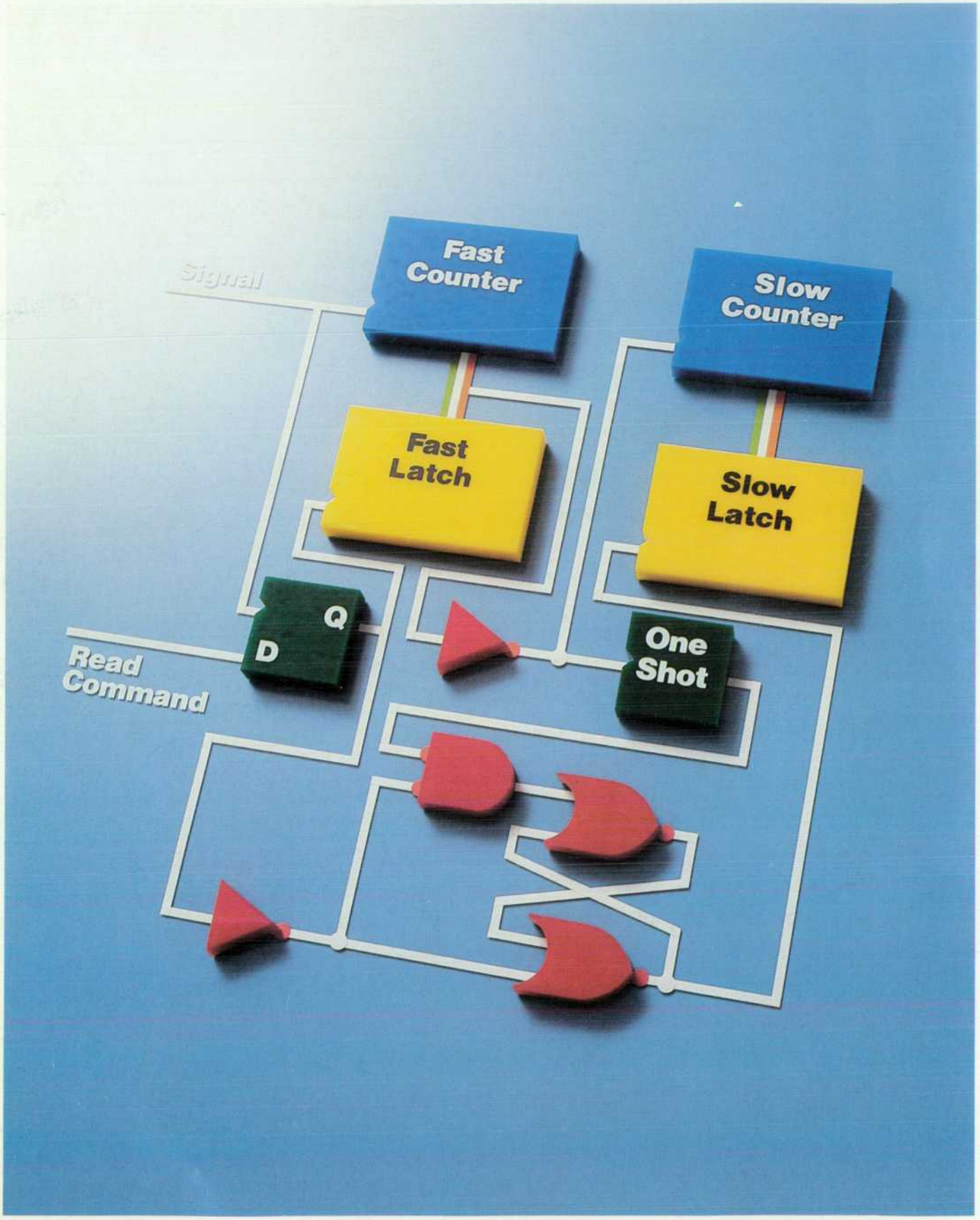
HEWLETT-PACKARD
JOURNAL

FEBRUARY 1989



HEWLETT-PACKARD JOURNAL

February 1989 Volume 40 • Number 1

Articles

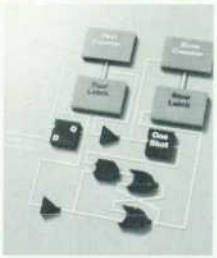
The **Hewlett-Packard Journal** is published bimonthly by the Hewlett-Packard Company to recognize technical contributions made by Hewlett-Packard (HP) personnel. While the information found in this publication is believed to be accurate, the Hewlett-Packard Company makes no warranties, express or implied, as to the accuracy or reliability of such information. The Hewlett-Packard Company disclaims all warranties of merchantability and fitness for a particular purpose and all obligations and liabilities for damages, including but not limited to indirect, special, or consequential damages, attorney's and expert's fees, and court costs, arising out of or in connection with this publication.

Subscriptions: The Hewlett-Packard Journal is distributed free of charge to HP research, design, and manufacturing engineering personnel, as well as to qualified non-HP individuals, libraries, and educational institutions. Please address subscription or change of address requests on printed letterhead (or include a business card) to the HP address on the back cover that is closest to you. When submitting a change of address, please include your zip or postal code and a copy of your old label.

Submissions: Although articles in the Hewlett-Packard Journal are primarily authored by HP employees, articles from non-HP authors dealing with HP-related research or solutions to technical problems made possible by using HP equipment are also considered for publication. Please contact the Editor before submitting such articles. Also, the Hewlett-Packard Journal encourages technical discussions of the topics presenting in recent articles and may publish letters expected to be of interest to readers. Letters should be brief, and are subject to editing by HP.

Please address inquiries, submissions, and requests to: Editor, Hewlett-Packard Journal, 3200 Hillview Avenue, Palo Alto, CA 94304, U.S.A.

In this Issue



The demand for new measurement and test capabilities often comes from the same source that gives us solutions—advanced technology. The two instruments whose designs are presented in this issue are advanced-technology solutions to advanced-technology problems. One is an analyzer that measures rapidly varying frequencies such as those found, for example, in spread-spectrum radar, communications, and navigation systems. The other is a signal synthesizer that generates the complex modulation formats used in many products and systems.

Chirp modulation, stepped frequency, and other spread-spectrum techniques make radar, communications, and navigation systems more immune to noise and more secure, but severely challenge measurement technology. The HP 5371A Frequency and Time Analyzer (pages 6 to 41) can measure the time-varying instantaneous frequency of such signals by a method known as continuous measurement technology. Most frequency counters measure for a time and then must stop while their data is read and processed. In the HP 5371A, zero-dead-time counters implemented with high-speed integrated circuits make one measurement after another without stopping. The counters are read on the fly and their data is stored to be processed later, say after 1000 measurements. The analyzer records the precise times at which the input signal crosses zero, a procedure known as phase progression digitizing (page 28), and from this data computes estimates of the instantaneous frequency. The technology is effective for dozens of frequency and timing measurements that were formerly difficult or impossible.

Applications for the HP 8904A Multifunction Synthesizer (pages 52 to 80) are found in navigation, commercial electronics, audio testing, communications, and other areas. For example, with the HP 8904A modulating a high-frequency signal generator, it's easy to generate the VOR (VHF omnirange) and ILS (instrument landing system) signals used by aircraft for navigation. Other examples, including FM stereo testing and telephone tone sequence generation, are described in the articles on pages 52 and 73. The complex modulation formats and signal combinations needed for these applications, which formerly required custom, one-of-a-kind generators, are created by the HP 8904A from six fundamental waveforms: sine, square, triangle, ramp, white noise, and dc. Low cost and high performance are achieved by calculating the waveforms digitally in real time. Doing all the math is a very large-scale integrated circuit, the digital waveform synthesis IC (pages 57 and 62), which is fabricated in HP's proprietary high-speed, high-density NMOS-III process. The 12-bit binary numbers generated by the synthesis IC are turned into the desired analog signals by an output system designed for excellent frequency response and low distortion (page 66).

The research report on page 42 is on a voice and data communications network architecture developed at HP Laboratories. Most networking technology in use today is based on ideas developed nearly twenty years ago, the authors tell us, while significant changes have occurred in recent years both in users' requirements and in the technology available for building networks. The architecture they describe is an attempt to address these changes. The architecture's main contributions are the true integration of voice and data (voice is transmitted as data), a single architecture for local, metropolitan, and wide area networks, high throughput with low overhead, very good cost/performance ratio, and compatibility with existing standards. The architecture project has been completed, and this report summarizes its results.

-R.P. Dolan

Cover

Sculpted in plastic is the circuit diagram of a zero-dead-time counter, a key component of the HP 5371A Frequency and Time Interval Analyzer. See Fig. 1 on page 33 for a more conventional diagram.

What's Ahead

Featured in the April issue will be several articles on the design of the HP 3458A Digital Multimeter, a state-of-the art instrument that's equally at home in high-speed automatic test applications or ultrahigh-resolution metrology laboratory applications. The VXIbus, a new industry standard interface for modular instrumentation, and an HP "starter kit" for VXIbus instrument designers will be described. There will also be several papers on software design from the 1988 HP Software Engineering Productivity Conference.

Characterization of Time Varying Frequency Behavior Using Continuous Measurement Technology

The HP 5371A Frequency and Time Interval Analyzer implements the continuous measurement technique to provide advanced capabilities for measuring frequency and time interval variations.

by Mark Wechsler

CHARACTERIZATION OF voltage-controlled oscillators and other signal sources with time varying frequencies is a difficult measurement problem that shows up in many seemingly disjoint applications. Spread-spectrum communication systems tax current measurement techniques. Digital devices are operating at increasingly higher data rates, making bit error rate measurements important for determining the quality of a digital system. Designers of information storage devices are pursuing ways to store more information in less physical space and access this information at higher rates. Digital communication systems are being developed that interact over many thousands of miles, causing stringent synchronization problems. In all these cases, the old model of a stable clock or a constant carrier frequency is oversimplified. Designers clearly need an improved model, one that includes wanted or unwanted frequency variations, and a method to measure this improved model accurately.

The continuous measurement technique promises to provide the needed capabilities for frequency and timing measurements. The technique is especially well-suited to the measurement of modulated or jittered signals. A time-domain trace of the modulation or jitter can be viewed directly. Measurement of continuous-wave, steady-state signals is not compromised by the technique.

The HP 5371A Frequency and Time Interval Analyzer, Fig. 1, implements the continuous measurement technique, giving users great flexibility and high performance in frequency and timing measurements. Proprietary HP ICs give the analyzer a frequency range of dc to 500 MHz. Single-shot time interval resolution is 200 ps. Continuous sampling is possible at rates up to 10 MHz. Up to 1000 consecu-

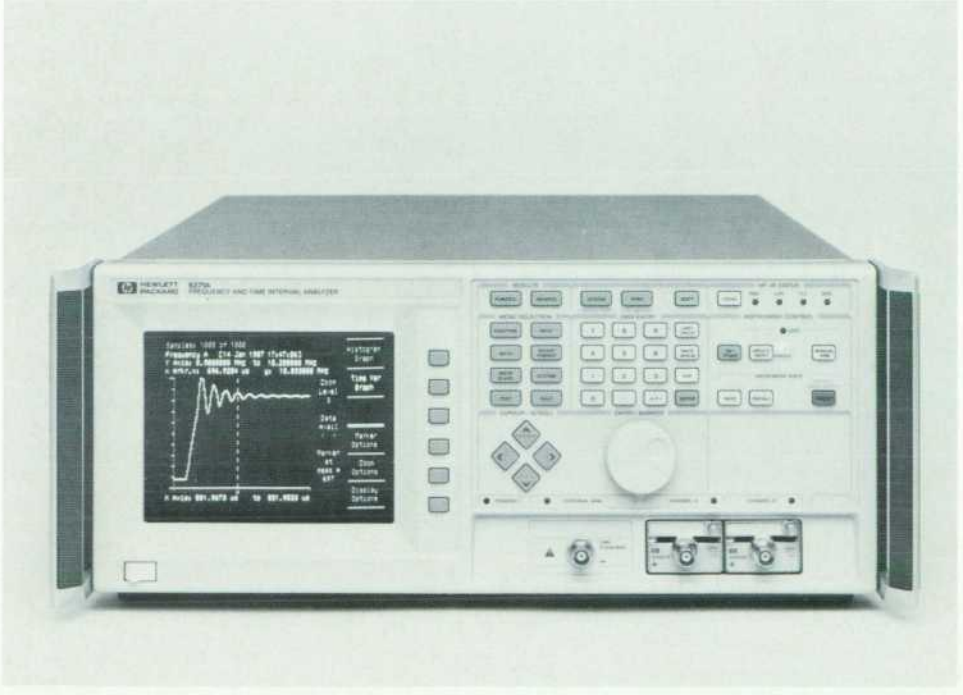


Fig. 1. *The HP 5371A Frequency and Time Interval Analyzer provides new capabilities for characterizing the frequency behavior of agile and other time varying sources. Its frequency range is dc to 500 MHz, but its capabilities can be applied to signals in the 2-to-18-GHz range using the HP 5364A Microwave Mixer/Detector.*

tive measurements can be acquired.

The input signal conditioning and arming capabilities exceed those of previously available counter products. Built-in analysis functions provide all of the features common to universal counters, along with graphical results, which are displayed on the built-in CRT.

The measurement and analysis capabilities of the HP 5371A can be applied to microwave and millimeter-wave signals, for example by down-converting these signals using the HP 5364A Microwave Mixer/Detector (see box, page 8). Designed as a companion instrument to the HP 5371A, the HP 5364A uses a mix-down technique that retains the phase and timing information of the original signal.

Continuous Measurement Capability

To illustrate the advantages of the continuous measurement technique, we can compare a traditional reciprocal counter with a continuous counter using a simple input signal, a steady-state sine wave (see Fig. 2).

A reciprocal counter (e.g., the HP 5335A or HP 5345A) opens the measurement gate (start sample), records event and time counts, and then closes the measurement gate after a predefined gate time (stop sample), again recording event and time counts (Fig. 2a). The measurement is complete at this point, and the frequency is computed using the frequency estimate:

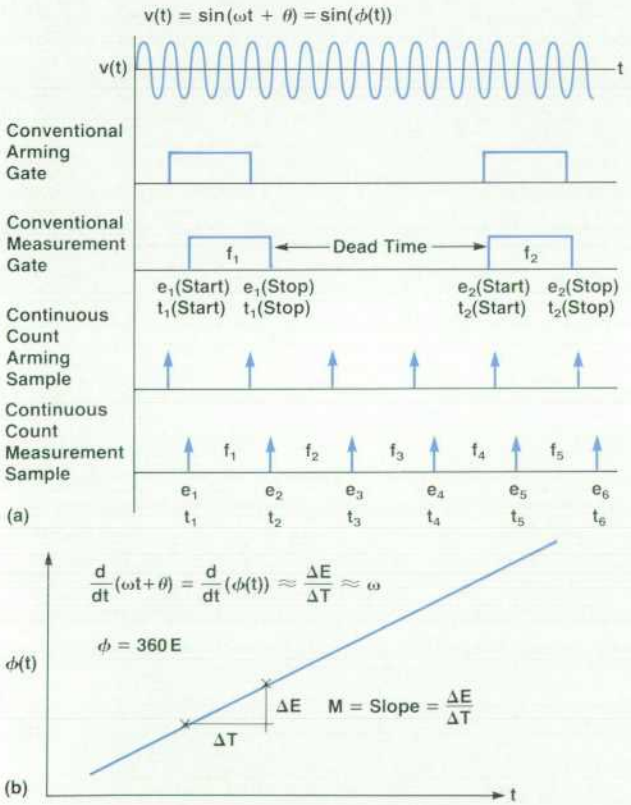


Fig. 2. *(a) A comparison of reciprocal counter operation and the continuous count technique. In continuous counting, the stop sample of one measurement is the start sample of the next. Measurements are made in blocks, and there is no dead time within a block. (b) The reciprocal counter estimates frequency as the derivative of phase, that is, the slope of a straight line connecting the start and stop sample values.*

$$f_{est} = \frac{e(\text{stop sample}) - e(\text{start sample})}{t(\text{stop sample}) - t(\text{start sample})}$$

This estimate can be understood using the definition of hertz, or cycles per second. We simply measure how many cycles occur in a specified time. The measurement gate is a version of the arming gate, synchronized to the positive transition of the input signal to be measured. This allows the event count to be an integral number, and the measurement error is entirely caused by the error in quantizing the time count.[1]

A more formal justification of the estimate is provided by a phase-versus-time graph of the signal. In Fig. 2b, the argument of $\sin(\omega t + \theta)$, $\omega t + \theta = \phi(t)$, is plotted as a function of time. For the simple case of a steady-state sinusoid, the plot is a line with slope ω. In general, $d(\phi(t))/dt = \omega$. A traditional counter computes the slope of this phase as the estimate of frequency (derivative of phase). The estimate is simply the slope of the straight line connecting the two sample values.

This frequency result is presented to the user numerically, and another measurement starts. It is clear that this technique has inherent dead time, during which changes in the input signal are not included in the mean-value result. The dead time is labeled in Fig. 2a. When the processing is occurring, a measurement is not permitted. Also, in principle (and usually in practice), the errors for different measurements are not correlated with one another.

Dead times not only interrupt signal measurements, but also destroy the timing relationship between gates. One approach to dealing with dead time is to measure it, either in the next pass,[2] or by using another counter operating in syncopation. However, with either method, the time scale generated is not truly continuous, being composed of small fragments whose systematic error may cumulate to a large value.

Again referring to Fig. 2a, we see that a continuous counter measures in a similar but more general fashion. Instead of interleaving measurement and processing cycles, measurements are performed back to back and are processed only after the last measurement. These back-to-back measurements are the essence of the continuous measurement technique. A series of continuous measurements is

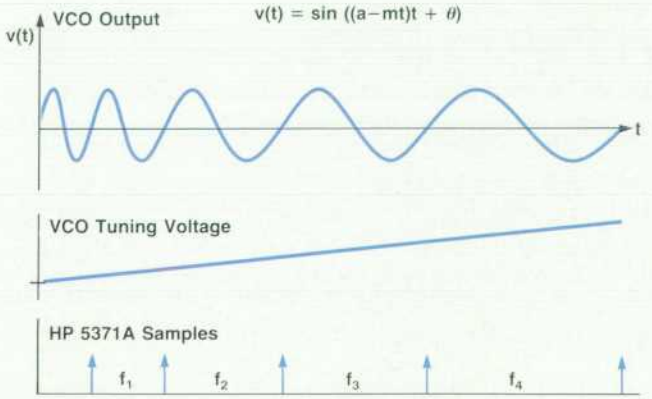


Fig. 3. *A voltage controlled oscillator with a linear tuning ramp, and the HP 5371A sampling points.*

Analyzing Microwave and Millimeter-Wave Signals

Microwave and millimeter-wave radar, communication, and navigation systems use spread-spectrum modulation to enhance system performance. Radars use chirp (linear FM pulse), stepped (pulse to pulse), and Barker (phase modulated pulse) modulation, and pulse repetition frequency (PRF) jitter or stagger. Random or pseudorandom frequency agility of the carrier is often implemented. Communication and navigation systems use FSK, PSK, or QAM modulation. Security is provided by pseudorandom coding or frequency hopping or both.

The measurement of these signals requires a wide bandwidth for frequency agility or hopping, an ability to capture the nonrepetitive random or pseudorandom modulation, and signal processing fidelity to preserve the modulation. The HP 5371A Frequency and Time Interval Analyzer has the wide-bandwidth baseband measurement capability to capture the nonrepetitive modulation.

Microwave and millimeter-wave signals can be measured by the HP 5371A if they are first brought within its frequency range of dc to 500 MHz by down-conversion, detection, or prescaling. The HP 5364A Microwave Mixer/Detector is designed for fundamental mixing, detection, and conditioning of microwave signals from 2 to 18 GHz for the HP 5371A's input. For millimeter-wave signals from 18 to 110 GHz a harmonic mixer (HP 11970 series) is used and the IF is amplified and filtered by the HP 5364A.

The HP 5364A mixer requires an input from a suitable local oscillator (LO) at +8 dBm. The HP 11970 millimeter-wave harmonic mixers require a +16-dBm signal from 3.0 to 6.1 GHz depending on the band. Fundamental mixing is implemented in the HP 5364A so that a user's system LO can be used for measurements that require a very low-phase-noise signal, such as Doppler radar. For frequency hopping communication systems the hopping LO can be used to prepare the signal for modulation analysis. The HP 8671B synthesized CW signal generator is an economical choice for use as an LO for the HP 5364A or HP 11970 mixers. (An HP 11975A amplifier is required to obtain the +16 dBm drive level.)

Fig. 1 shows a block diagram of the HP 5364A and its relationship to the LO and external mixer. To provide simultaneous IF and detected signals, the input signal is first adjusted for linear dynamic range by a 10-dB step attenuator. The signal is then resistively divided into the mixer and detector channels. The local oscillator drive is connected by semirigid coax to the front-panel input connector. Semirigid coax is used throughout the instrument to minimize phase or amplitude variations resulting from vibration or shock. A 10-dB fixed attenuator provides additional mixer padding and isolates the detector from LO leakage through the L-R path of the mixer. Broadband IF amplification from 10 to 500 MHz compensates for the power divider, pad, and mixer conversion losses and provides a nominal gain. Low-pass filtering is provided to ensure that the LO signal leakage (L-I) is suppressed by greater than 40 dB in the IF output to preserve measurement accuracy. The R-I path leakage is also suppressed by greater than 40 dB. IF group delay ripple is typically less than 1.0 ns from 20 to 500 MHz. The mixer noise figure is typically 8.5 dB but the power divider and pad add 16 dB of loss to the input port for a system noise figure of 24.5 dB.

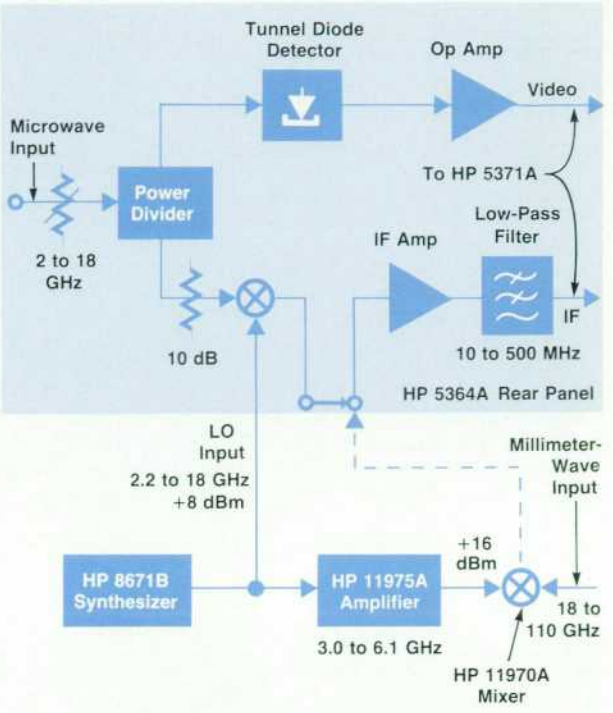


Fig. 1. *The HP 5364A Microwave Mixer/Detector is used with an external local oscillator to down-convert microwave signals for analysis by the HP 5371A Frequency and Time Interval Analyzer. For millimeter-wave signals, an external mixer is also required.*

Compared to IF detection, microwave detection provides arming and pulse measurement capability with minimum noncoherent timing jitter. A tunnel-diode detector has square-law response and less than 5 ns rise time. A wideband op amp conditions the signal to typically 80 mV with a −12 dBm input (<10 GHz). This direct coupled video signal can be used to measure jittered or staggered PRF or PRI signals, pulse width and variations, pulse rise and fall times, and pulse position coding.

The microwave sensitivity of the IF channel is −25 dBm and the maximum input (attenuator set at 50 dB) is +48 dBm peak pulse power. Video sensitivity is −12 dBm and the maximum input is +48 dBm peak pulse power.

The millimeter-wave sensitivity of the IF channel is typically −20 dBm at Q Band (measured at 44 GHz). The sensitivity for other bands is proportional to the millimeter-wave mixer conversion loss for the particular band.

Richard Schneider
Project Manager
Santa Clara Division

called a block. Inside a block, there is no possibility of missing information. For every measurement except the first and last, the start sample of the ith measurement is identical to the stop sample of the (i − 1)th. As a result, the errors for different measurements are correlated, which improves the averaging performance. Between blocks of mea-

surements the continuous counter has dead time during which these measurements are processed.

A continuous frequency estimate, actually a series of estimates, is calculated similarly to the conventional calculation:

$$f_{est}(i) = \frac{e(i) - e(i-1)}{t(i) - t(i-1)}$$

Any frequency instability of the signal is included in the continuous frequency estimate data. Also, for a steady-state signal, a continuous frequency estimate can be more accurate than a traditional estimate, given equivalent measurement times, because of the absence of dead time. The continuous measurement technique can average more measurements and thereby provide more frequency resolution (digits) per unit time, an important parameter in most systems.

Characterization of a voltage-controlled oscillator (VCO) provides insight into the benefits of continuous measurement in measurements on agile signals. In Fig. 3, the time-domain waveform of a VCO is shown. A voltage ramp applied to the tuning voltage input changes the frequency of the VCO linearly with time. The dynamic nature of this kind of signal makes VCO characterization using traditional techniques difficult. With a traditional counter, valuable information is lost during the measurement dead time. While it is possible to make the measurement using a sophisticated test setup and recording multiple passes of the data, the complete information is available directly using the continuous method. Additionally, all information is accumulated in a single pass. What we get is a straightforward result in a format that is easy to understand.

Fig. 4 shows how the HP 5371A measures the frequency of a frequency modulated signal.

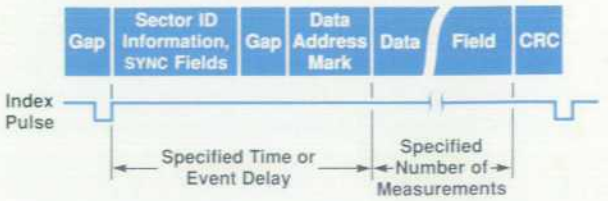


Fig. 5. *In a disc drive, a time or event holdoff can be used to delay measurements from the index pulse into the data portion of the sector.*

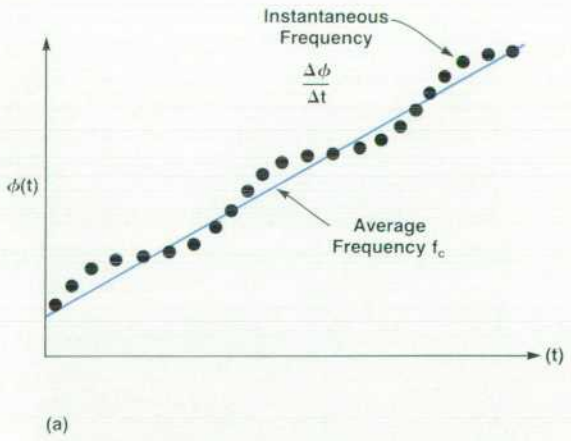
Arming and Gating

Modern communications signals use complex coding and multiplexing schemes to pack more information into less bandwidth. For example, in disc drives, information is stored in sectors, and each sector has a number of fields, such as header, data, and error-correction fields. In digital communications links, information belonging to many users is time-multiplexed onto a single channel, so header information is required to identify the users. In both these applications, it is advantageous to have the ability to measure only part of the entire waveform. This requires sophisticated arming and gating capability.

The HP 5371A provides this capability as a standard feature. There are four classes of capability: sampling, holdoff, holdoff/sampling, and holdoff/holdoff.

Sampling. Sampling provides a way to tune the measurement process to match the input waveform's frequency variations. The frequency resolution of the result is inversely proportional to the sampling interval. Thus, selecting the sampling interval is a trade-off between measurement rate and measurement resolution.

Sampling controls the interval during which distinct frequency or timing measurements are made. The HP 5371A is very efficient in this respect, with sampling intervals selectable from 600 nanoseconds to 4 seconds. Three modes



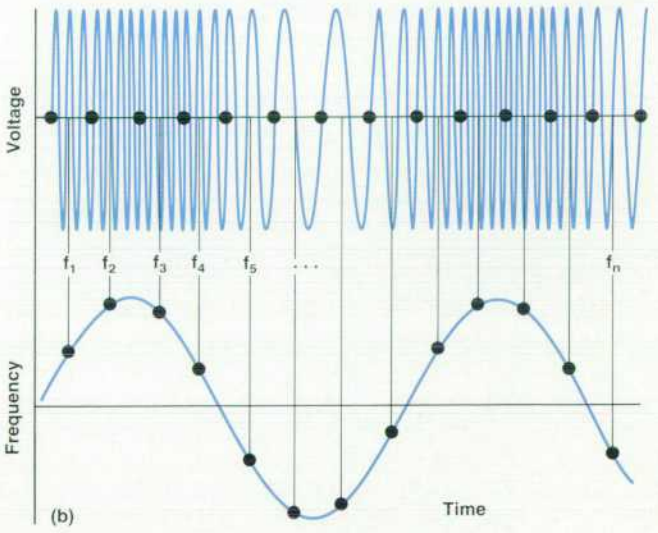


Fig. 4. *(a) The HP 5371A measures frequency by precisely measuring the times at which integer numbers of signal cycles have occurred. A phase progression plot of the measured data points reveals the instantaneous and average frequencies. (b) The number of cycles in each time interval divided by the length of the interval gives the instantaneous frequency estimate f_i for that interval.*

of sampling are supported. First, an internal signal is provided at a user-selectable, fixed interval from 600 ns to 4 s. Second, an edge on the input channels (A,B) or the external arming channel can be used. Third, sampling can occur at the maximum rate, which is either every 100 ns or controlled by the period of the input signal.

Holdoff. To measure a specific part of the input signal, holdoff arming is provided. Holdoff arming controls when a block of measurements begins. For a disc drive, the jitter on the data sequence can be measured without including the jitter in the header field, as shown in Fig. 5. In many cases the jitter is worse for certain known data patterns, so we don't want to include the information in the header field, which will not be equal to (in general) the worst-case data pattern.

As shown in Fig. 5, an edge is provided to identify the beginning of the data field. Two modes of delay arming are provided. If the edge is coincident with the data field, the edge can be used directly. If the data field occurs after the edge, a time delay can be used to synchronize the measurement with the correct area of the signal. Similarly, a delay of a specified number of events can be used to accomplish this synchronization. If holdoff arming is selected, the measurement block will be held off until the appropriate conditions are met. Each block will be held off in the same manner. Within the block, the continuous measurements are accumulated at the fastest rate possible, which is every input transition or every 100 ns, whichever is longer.

Holdoff/Sampling. The third type of arming provided is a combination of the first two. In many cases, holdoff is required and a slower sampling rate is desirable for increased frequency resolution. In this case, HP 5371A operation is shown in Fig. 6. This is the most general type of arming.

Holdoff/Holdoff. The fourth type of arming is used when two holdoffs are required. In this case the block size is, by definition, equal to one, and the HP 5371A functions much like a conventional counter (with arming). Time and event holdoffs are both supported, but cannot be mixed. HP 5371A operation in this mode is shown in Fig. 7. This class of sampling is useful for picking out a particular interval in a data pattern.

Hardware Design

An HP 5371A signal flow block diagram is presented in Fig. 8. Many of the sections are covered in detail in the accompanying articles.

Input Section. Two measurement channel inputs (A,B) and one external arming channel input are supplied. These

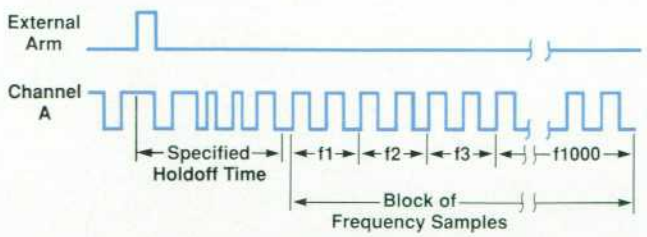


Fig. 6. *The time holdoff arming mode can be used to delay blocks of measurements by a specified time.*

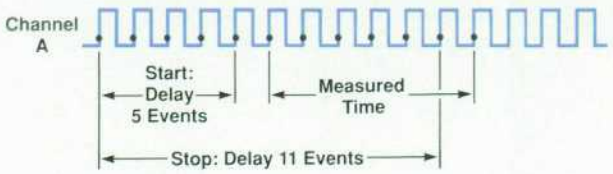


Fig. 7. *The start and end of a single measurement can both be defined by time or event holdoffs.*

input channels convert arbitrary-level input signals into fixed-level digital signals. The dynamic range of the measurement channels is ±45 mV p-p to ±2V p-p. These inputs are dc coupled and have 500-MHz bandwidth. The dynamic range of the arming channel is 140 mV p-p to 5V p-p. This input is dc coupled and has 100-MHz bandwidth. The measurement channels (A, B) can also be used as 500-MHz arming inputs.

Time Base Subsystem. The HP 5371A uses an HP 10811A 10-MHz precision ovenized crystal oscillator for its fundamental time reference. This signal is multiplied up to 500 MHz for use in the time sampling subsystem and the interpolator subsystem.

Arming. This system provides the capabilities described earlier. It is implemented using custom bipolar ICs developed in conjunction with HP's Santa Clara Technology Center. Some amount of discrete ECL logic is required to control the LSI custom circuits. The arming section basically determines when the latching system takes samples.

Latching and Memory System. Precision timing is accomplished in this section. Data from two interpolators and time and event registers is stored sequentially in memory to form blocks of measurements.

HP-IB System. A hierarchical command language was adopted to simplify programming. Three formats of output data are supported. In order of increasing throughput, they are ASCII, floating-point, and binary. In the binary mode, data can be transfered at a maximum rate of 20,000 measurements per second.

Microprocessor. A 68000 microprocessor system controls the functions of the HP 5371A. It includes 625K bytes of EPROM for program storage (550K bytes are used) and 256K bytes of system RAM (200K bytes are used). All processing and analysis tasks are performed in firmware.

Keyboard/Graphics Display. A 7-inch CRT displays both numeric and graphic results. It also supports the softkey

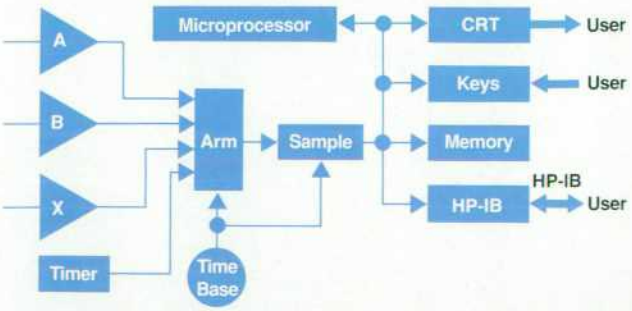


Fig. 8. *Signal flow block diagram of the HP 5371A. There are two measurement channel inputs and an external arming channel input.*

labels. A keypad provides 17 hard keys and 6 softkeys. A rotary pulse generator (knob) is provided to ease numeric entry and control display cursors and scrolling. Four up-down arrow keys are provided.

Power Supply. This module supplies +5V dc, −5.2V dc, −3.25V dc, ±15V dc, and −25V dc to the other hardware systems. The maximum ac power used by the instrument is 500 VA.

Data Analysis

The increased measurement capability of continuous count is complemented by an extensive set of analysis features. Graphic representation of information is especially important because a large number of measurements can be accumulated within blocks. A sequential list of measurements is available, but most often this is a time-consuming way of looking at the data. Graphics processing and display capability helps the user visualize the information contained in the list.

Statistics. Statistical computations offered in the HP 5371A include minimum value, maximum value, mean value, mean squared, root mean squared, Allan variance, and square root of the Allan variance. These features can be used for limit testing, pass-fail testing, and increasing resolution by averaging.

Histogram. A histogram analysis function is provided, which gives a graphical representation of the data, sorted according to measured result. An example is illustrated in Fig. 9.

Multiple distributions often occur and are difficult to characterize because several means and standard deviations may be required. The histogram display gives the user the ability to acquire several different results and compute the statistics of each distribution separately (Fig. 10).

A statistical representation of the results is often useful when determining margin on serial data links. Errors occur very infrequently, typically one error in 10,000,000 bit transfers. A histogram effectively provides the user with a probability density function for the random process. Mean value is important, but perhaps more important is the evaluation of the standard deviation and minimum and maximum values, and how these relate to timing margin (i.e., quality).

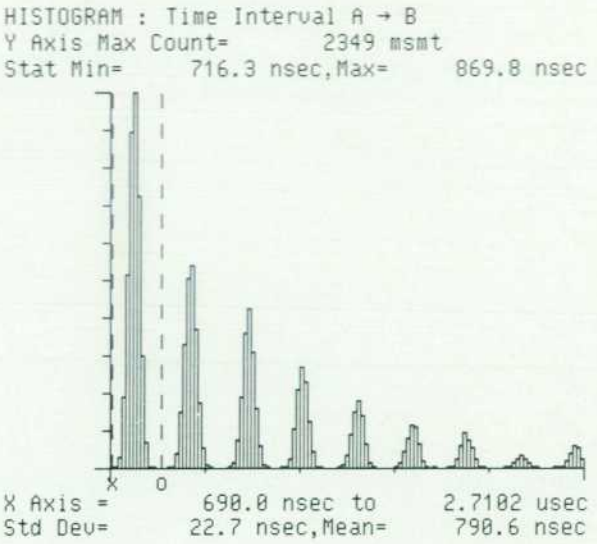


Fig. 10. *Histogram of compact-disc pulse width measurements with eight-of-fourteen modulation.*

An important characteristic of the histogram display is that it can be used over a large number of blocks. Consequently, statistical information can be accumulated over a very large number of samples.

Time Variation. The time variation display can be used to represent an input signal's instantaneous frequency or tim-

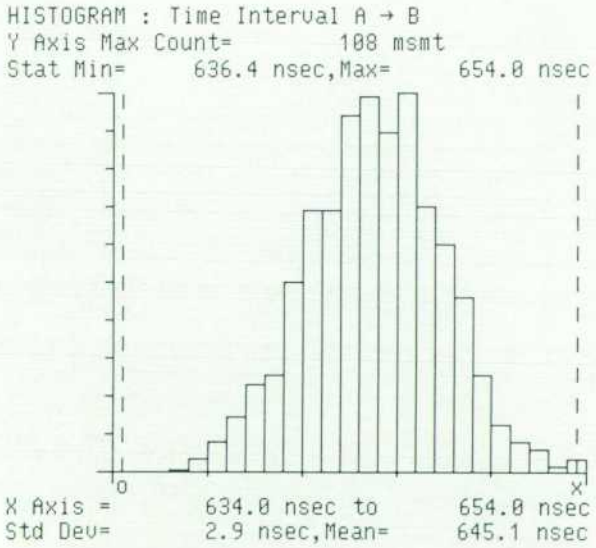


Fig. 9. *Histograms can be displayed on the CRT. Marker and zoom features are available.*

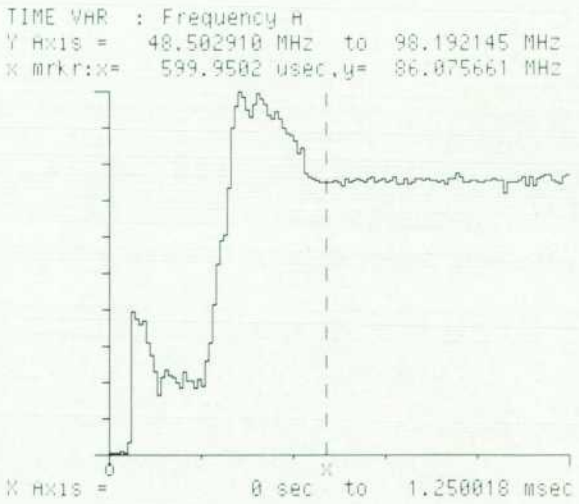


Fig. 11. *The step response of a VCO is shown by a time variation plot.*

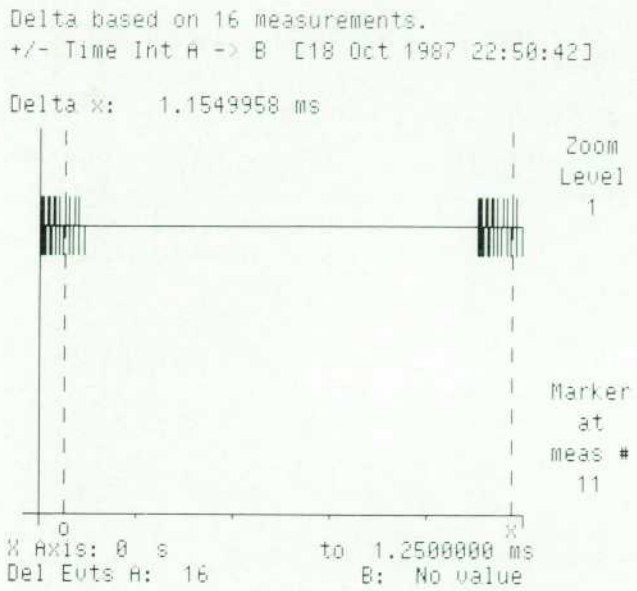


Fig. 12. *An event timing graph for two pulse bursts shows that 16 events occurred on the A channel between the markers and the marked events occurred 1.1549958 ms apart.*

ing behavior. This display is simply a graphical representation of the measurement result over time. The sample rate of the measurement must agree with the rate of change of the input signal characteristic. The step response of a voltage controlled oscillator shows the power of this capability (Fig. 11).

Event Timing. Another way of representing the instantaneous behavior of signals is the event timing graph. Transitions of the input signal are displayed in a manner very similar to a logic analyzer. This format is useful when the sequence of the input pulse stream or the relative location of two input pulse streams is desired. Fig. 12 demonstrates this capability for a pulsed signal.

Acknowledgments

I would like to thank Dan Hunsinger and Art Muto for their leadership and support throughout the product development. Dave Chu provided the concept of continuous measurement and gave technical guidance to the project team. Special thanks to Phil Deaver, who led the development effort. Bruce Greenwood and Alex Peake contributed both to product definition and to identifying the applications for this measurement technology. Kelly Spafford, Darryl Scroggins, and Al Heredia were key to a successful introduction to manufacturing. Carol Courville, Bo Garrison, and Deborah Faryniarz designed the mechanical package. To the entire team, R&D, manufacturing, marketing, and quality assurance, thanks for a job well done.

Reference

1. HP Application Note 200, *Fundamentals of the Electronic Counters.*

2. R.G. Huenemann, "Signal Processing Techniques for Automatic Transceiver Testing," *Hewlett-Packard Journal*, Vol. 24, no. 10, August 1973, pp. 8-13.

Firmware System Design for a Frequency and Time Interval Analyzer

Built-in control and analysis firmware tailors the continuous measurement technology of the HP 5371A to dynamic frequency and time interval applications.

by Terrance K. Nimori and Lisa B. Stambaugh

FREQUENCY COUNTERS offer the high resolution and measurement flexibility essential for examining frequency or timing stability. Traditional counters, however, lack measurement control and analysis capabilities for profiling changes in frequency or timing parameters. Although external processing is often used to reveal modulation or jitter components, software complexity and non-continuous sampling limit the effectiveness of this technique.

Based on continuous measurement technology, the HP 5371A Frequency and Time Interval Analyzer represents a significant contribution to dynamic signal analysis. Powerful analysis and graphics firmware provide enhanced performance capabilities for such applications as frequency modulation and timing jitter measurements.

This article examines the major firmware components of the HP 5371A, and focuses on their design and application to the continuous measurement concept.

Design Objectives

The design objectives of the firmware were strongly influenced by the potential of the continuous measurement technology. These objectives included:

- Supporting continuous measurement processing and statistical and graphical analysis functions
- Providing an enhanced user interface through a menu-driven front panel and a descriptive HP-IB programming language
- Partitioning the firmware system into functional modules that can be combined for extended capabilities
- Leveraging technologies from other HP products.

We concluded early in the design of the HP 5371A that while existing designs offered basic features, performance improvements and additional capabilities were often needed to address these objectives. An example that illustrates this contrast is the microprocessor system. While a conventional counter might use an 8-bit processor and typically 16K bytes of ROM, the HP 5371A uses a 32-bit 68000 microprocessor for overall control. A total of 550K bytes of ROM and 200K bytes of RAM are used. Unlike many existing designs, which were written in assembly language, most of the HP 5371A firmware (approximately 80%) was written in Pascal for maximum flexibility and development efficiency. Only time-critical functions, such as interrupt handlers and hardware interfaces, were written in assembly language.

System Architecture

The firmware system for the HP 5371A consists of five

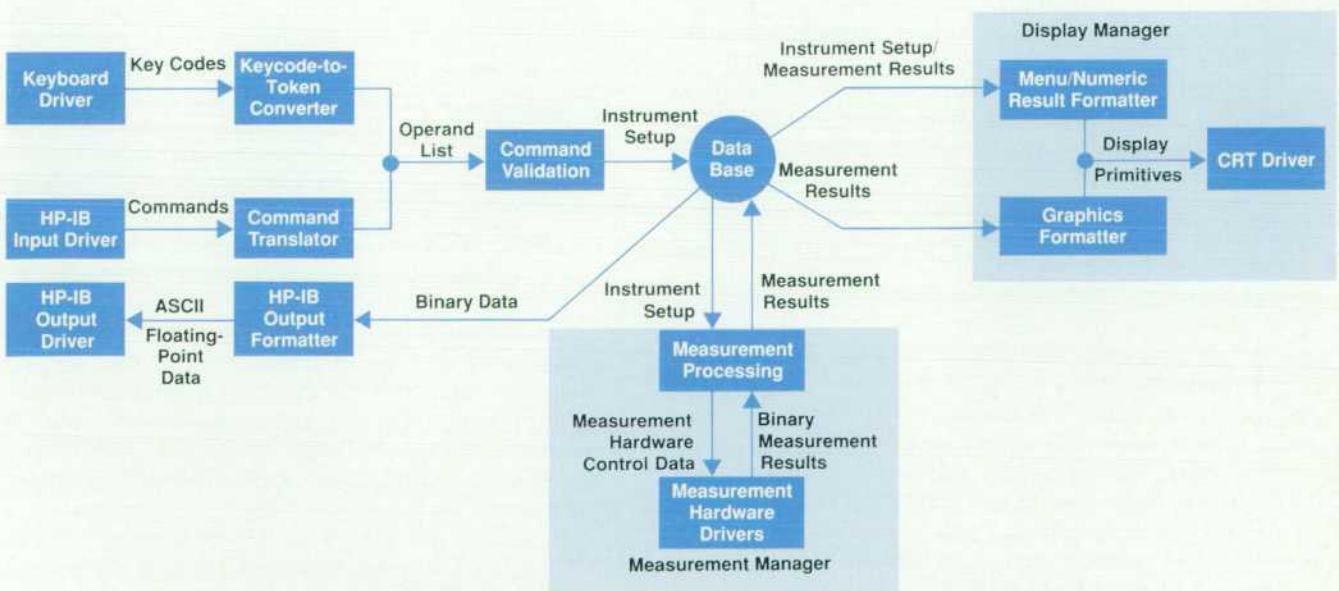


Fig. 1. *Firmware architecture of the HP 5371A Frequency and Time Interval Analyzer.*

major modules (Fig. 1):

- Command Translator
- Measurement Manager
- Display Manager
- HP-IB
- Central Data Base.

These modules are synchronized by a real-time operating system.

The command translator parses keyboard operations or HP-IB commands into instrument settings, which are saved in the central data base. These settings are used by the measurement manager to program the measurement hardware. The measurement manager is also responsible for retrieving and processing measurement results. Menu screens and graphics displays are controlled by the display manager. The HP-IB module builds input command strings and provides a variety of output data formats and status reporting capabilities for the automated test system environment.

Operating System

The HP 5371A operating system is divided into two sections: scheduling executive and command interpretation. The scheduler is responsible for scheduling tasks and for passing messages between these tasks. The command interpretation section includes the command translator and the action-taking tasks that carry out the commands requested.

Messages are used to communicate between tasks. One task creates a message and sends it to another task. The second task reacts to this message by performing some action and/or sending another message. Messages are used to send information from task to task, and to synchronize their operations. Messages are created by tasks and left at exchange points to be picked up or received by some other task. Tasks may wait at exchange points for messages for a specified or unspecified length of time. If a default return time is not specified, a task may wait indefinitely for a message. This ensures that the task will not continue execution without receiving the proper message.

Interrupts are also handled through this message structure. When an interrupt occurs, the appropriate message is sent to the task that handles that interrupt. Examples of typical interrupts include HP-IB, measurement hardware, and real-time clock interrupts. In some cases, the message is only a signal that an event has happened, such as the completion of a measurement or the occurrence of another clock cycle. In other cases the message contains additional information, such as the data byte sent via the HP-IB.

The message handler is responsible for all message processing procedures. To minimize complexity, especially in the area of garbage collection, we use fixed block sizes to create messages. These fixed-length blocks are joined in linked lists to obtain variable-length messages. The message handler takes care of allocating blocks of free space for messages and later returning blocks to free space when they are no longer needed.

All tasks have an assigned priority, and are queued for execution in order of priority. When one task must halt while waiting for a message, the scheduler can determine the next task to run based on this priority queue. Two queues take care of waiting and ready tasks, and a pointer to the running task indicates which task is currently running. When a message is being processed, the running task is inserted into the ready queue. If the action resulting from the latest message causes another task to move from the waiting state to the ready state, that task is also inserted into the ready queue. After the message has been processed, the task at the front of the ready queue becomes the new running task. If the running task goes to the waiting state, it is placed into the waiting queue and the first task in the ready queue becomes the running task (Fig. 2). The previous running task will run to completion when it again becomes the highest-priority task.

Conventional frequency counters are designed using an in-line executive scheme, that is, a single process flow defines the instrument's operation. The measurement process is typically organized as: measure, process data, display, and repeat. When interrupts occur (e.g., keyboard or HP-IB), the executive is suspended, an interrupt processing procedure handles the interrupt, and the executive is restarted. Since these counters have very few types of interrupts, this is not very complicated; there are usually only two or three separate interrupt processing routines.

In the HP 5371A, the executive is one task and the interrupt processing procedures are the other tasks in the system. In most conventional counters, the system is not sophisticated enough to resume operation in the middle of the executive after interrupt processing is complete. The interrupt always forces restarting of the executive, and therefore, possible loss of results. In the HP 5371A this is not a problem. Interrupt processing always takes precedence over the executive as the running task, and when the interrupt handling procedures are complete, the executive always becomes the running task again.

Command Translator

The command translator interprets all incoming commands to the instrument. The command translator task works together with the command translator sequencer task to interpret these commands and send messages to the tasks that act upon them. The command translator sequencer allows the command translator to process incoming commands as quickly as possible, leaving the communication details to the command translator sequencer.

The main body of the command translator is the parser, which examines a string of tokens (the keywords and data items in the incoming command) and generates a list of operands. This list of operands is passed to whatever task needs to act on the command. Some error checking is done within the command translator for invalid keyword sequences or syntactically incorrect commands such as misspelled keywords. Validity checking based on the instrument configuration is left to the action tasks. The command translator sends the operand list to a specified task. That task converts the operands into parameters, which may be passed to separate procedures, depending on the function.

Front-panel keypresses are converted to equivalent tokens in preparation for calling processing procedures that handle specific commands. When these procedures are called, there is no indication whether the command comes from the front panel or the bus; both cases are handled

identically. This allowed us great flexibility in implementing command actions. Once the command translator was running, it was a simple matter to add an HP-IB command to the grammar and test the functionality of a feature without implementing the menu processing necessary to choose that function or option via the front panel.

Conventional counters use state tables to process keystrokes and eventually arrive at a procedure call based on the current key. HP-IB commands are parsed separately, and eventually lead to the same final procedure call. Previous design teams traditionally coded all front-panel interaction first, leaving the HP-IB to the end of the implementation schedule. One major philosophical difference between our project and these earlier ones involved coding the HP-IB interface early enough to take advantage of the command input while still designing the instrument. Since all of the action processing routines were developed only once, we were certain that the HP-IB commands behaved exactly like the front-panel commands.

Data Base

The HP 5371A uses a central data base for global parameter storage. In this data base all variables that define the instrument configuration are declared in a contiguous block of memory. These global declarations take place in a separate file, which does not contain program code. These variables require about 350 bytes of RAM, and are stored in groups: all measurement variables together, all math processing variables together, and so on. Many of these variables are accessed by more than one module or file, and it would be difficult to determine which file owns that variable, or should rightfully declare it globally. By declaring all of them in a separate file in a contiguous section of memory, many of these problems are avoided. In development, when variables needed to be added to that group, only one file had to be recompiled.

This implicit order is used in several ways. When the instrument is powered down, a backup battery allows the recovery of the previous instrument state. Since these variables are already stored in order and we know the size of the block, we know exactly which ones are recalled, and which ones are to be reinitialized on power-up.

Nine saved configurations allow the user to set up the instrument for a particular application and store it for later recall. By using the same set of variables for saved configurations and to define the instrument state, we are able to use a duplicate buffer patterned after the original memory configuration. Using pointers, it is simple to copy the current configuration to a saved one without knowing the order or number of bytes to be stored (a constant contains the memory block size). When the configuration is recalled, a pointer is again used to index through the copied buffer,

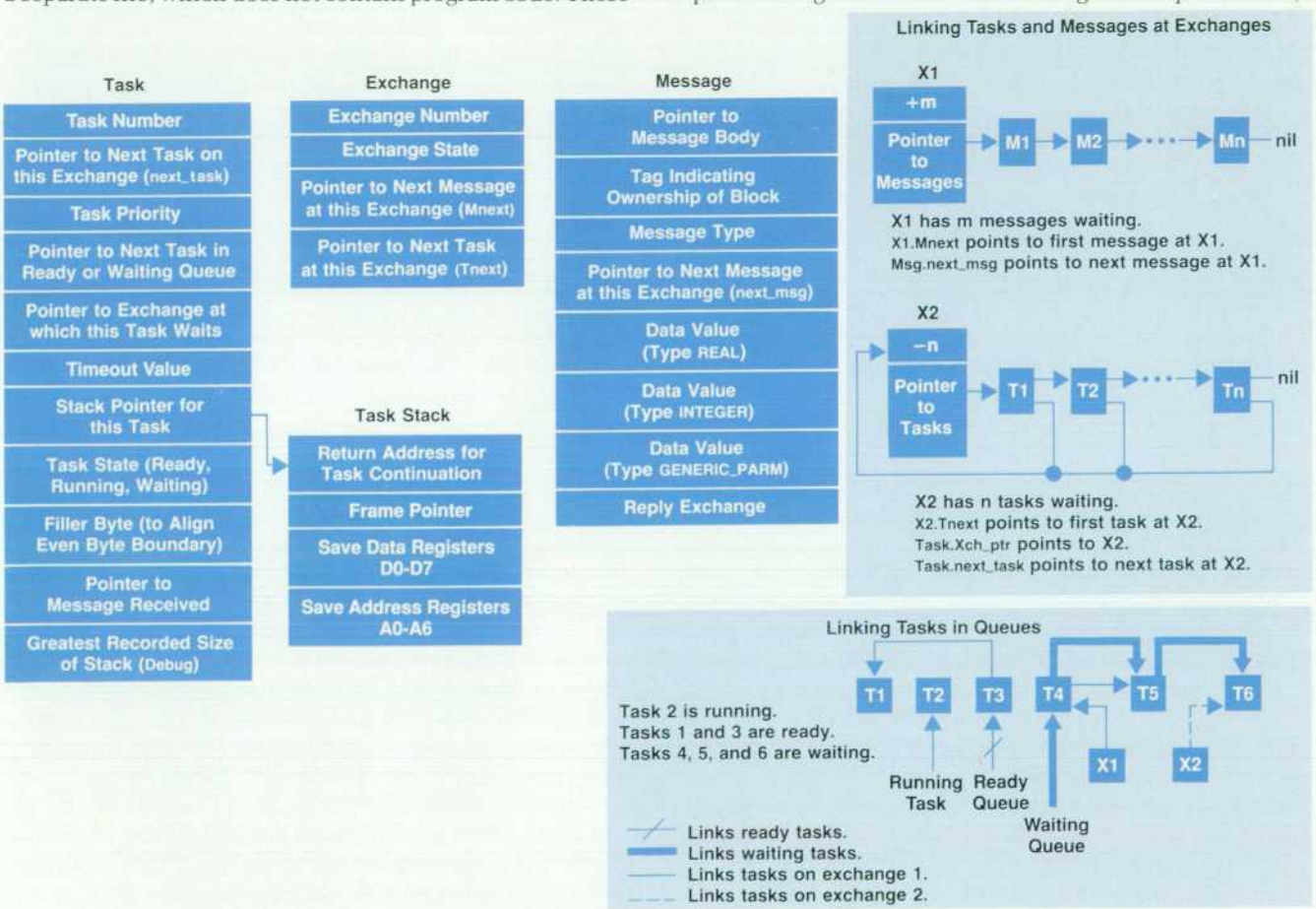


Fig. 2. *Operating system data structures.*

copying it back into the original memory configuration.

When the front-panel **PRESET** key is pressed, the current configuration is automatically copied into register 0, allowing later recovery. Again, a pointer makes the indexing and copying simple. By keeping all of these parameter subsets the same, we achieve consistency easily.

Measurement Configuration

The measurement manager directs measurement acquisition and processing. It initiates a measurement cycle by programming the input amplifiers and the counting hardware to perform the selected measurement. At the completion of the measurement cycle, it extracts the values of the counters, derives a frequency or time interval, and saves the results in measurement arrays. The measurement manager also performs postprocessing, such as statistics and limit testing.

Continuous Count Architecture. The HP 5371A is based on a continuous measurement architecture. The concept is straightforward: sample the value of a binary counter without disrupting the counting process. Until recently, this was difficult to achieve at high input frequencies and sample rates while maintaining useful resolution. In the HP 5371A, custom sequencer and zero-dead-time (ZDT) counter ICs provide this capability (see article, page 35).

In a typical measurement, input events (trigger edges of signals applied to channel A or B) and edges of a 2-ns reference clock are accumulated in ZDT counters. A ZDT counter contains a 16-bit presettable counter and dual 16-bit latches, all of which have 500-MHz bandwidth.

There are three ZDT counting chains, each consisting of two cascaded ZDT counters. Two ZDT chains can be directed to count events simultaneously on both input channels while the third chain counts the reference clock signal. Thus, simultaneous frequency measurements can be performed on two input channels.

Instantaneous values of these counters are latched in high-speed memory as often as every 100 ns without interrupting the counting process. Latch signals are provided by the sequencer IC, which routes the various clock and event signals to the appropriate ZDT counters.

Supporting this system are digital interpolators and the gate timer. Digital interpolation increases the effective resolution of the system by measuring the phase relationship between the reference clock and the start and stop input events. The time difference between the first or last input event and the following clock edge is quantized to 200 ps.

The gate timer generates sampling intervals and arming delays. It is analogous to the gate time control of a conventional counter. For continuous sampling, however, the timer does not necessarily close the measurement gate. Instead, it defines when the current values of the ZDT counters are to be sampled.

Programming the Sequencer. The sequencer multiplexes latch signals to the ZDT counters. It routes selected arming signals to internal gate flip-flops and implements two-stage sequential arming for precise control of the sampling rate. In general, a measurement cycle may be armed automatically, or by an edge of an input channel, or following a time delay or a number of events referenced to an input channel edge.

Eight 7-bit control registers are configured by the microprocessor to perform the selected measurement. These registers control two similar functional blocks consisting of arming and counting circuits. The arming block selects the sources of latch signals for the ZDT counters. The counting block selects the sources to be counted, such as an input channel or the reference clock, by routing these signals to the appropriate ZDT counters.

Most of the programming data is extracted from a table of control information. An element of this table corresponds to a particular combination of measurement function and arming mode. These settings are retrieved from the data base when the instrument configuration is changed or when a measurement cycle is restarted by the user. Control bits that correspond to specific channel qualifiers are appended.

Programming the ZDT Counters. The ZDT counters accumulate input events or reference clock pulses and output instantaneous or latched values to the microprocessor or DMA-configured measurement memory. They can be programmed to generate an interrupt signal when one or more predetermined conditions have been satisfied. Each ZDT counter interfaces to the microprocessor with a 13-bit control register that contains the latch control bits, interrupt mask, and binary divider ratio.

Three latch modes can be programmed. In the normal operating mode, a ZDT counter is configured to accept latch signals from the sequencer. After a latch signal is received, the value of the counter is latched on the next edge of the input signal or reference clock. If synchronization is unnecessary, the microprocessor can set a control bit that causes the current value of the counters to be latched asynchronously. This is referred to as a forced latch. Finally, the latch signal may propagate from another counter chain.

An interrupt mask defines when a ZDT counter should send an interrupt signal to the sequencer or measurement memory. For example, a ZDT counter can be preset to a value and count incoming events until it rolls over from its maximum value to an all-zero state, when it generates an interrupt signal. This condition is referred to as terminal count. It is primarily used as a qualifier in two-stage arming, such as an event holdoff. To hold off a measurement by a number of events, a pair of cascaded ZDT counters is programmed to its maximum value ($2^{32} - 1$) reduced by the number of events. After that number of holdoff events, the next event will cause the ZDT counters to roll over to zero and generate the terminal count signal, indicating to the sequencer that the number of events has been counted.

Precise time delays can also be generated with this technique. By counting a number of reference clock edges with a ZDT counter, a time holdoff between 2 ns and 8 seconds can be generated with 2-ns resolution.

A special case arises when fewer than 65,536 events are specified. Since the higher-order ZDT counter is not needed, it is preset with a value of zero, and the microprocessor sets a control bit to specify a cascaded latch signal. As events are received by the lower-order counter, they will be counted until a rollover occurs.

Alternatively, a ZDT counter can be programmed as a binary divider. Interrupts are generated whenever specific

multiples of the reference clock or an input signal are counted. In the HP 5371A, the user can select divide ratios of 2^4, 2^8, 2^{12}, ..., 2^{28}. While these ratios are more restrictive than the number of holdoff events that can be specified for the event arming mode, this cycle arming mode permits continuous measurement acquisition. A primary application of this feature is a constant-event frequency or period measurement.

The ZDT counters must be programmed before a block of measurements is acquired (unless the binary output mode is selected) or before every measurement if they are used to generate an event or time delay. In other cases, they remain free-running. Each ZDT counter is initially reset, loaded with a preset value, then reset again. A ZDT counter is preset even with a value of zero, since this allows a maximum time to elapse before an overflow occurs.

Programming the Gate Timer. Two gate timers generate repetitive sampling intervals for arming continuous measurements. These intervals are less precise than those generated by counting reference clock pulses with the ZDT counters. Unlike ZDT-generated delays, however, the gate timer does not require presetting between sampling intervals. Consequently, the interval arming mode permits continuous measurement acquisition, while the time arming mode is inherently noncontinuous.

The user-specified delay is translated into multiples of 200 ns and programmed as a binary value. The gate timer is initiated by a signal from the sequencer.

Measurement Memory. Binary count data is saved in 16-bit RAMs. A RAM is assigned to each ZDT counter. In addition, the 4-bit start and stop interpolator values are saved.

To maximize the measurement rate, a DMA controller facilitates high-speed memory operations. Before a measurement cycle is initiated, the microprocessor programs the DMA controller with the starting address of this memory. The starting address is determined by subtracting the number of measurement samples from the highest memory address. Samples are then saved at successively higher memory addresses. When the address counter reaches its maximum value, the DMA controller sends an interrupt to the microprocessor, signaling the completion of the measurement cycle.

Autotriggering. Autotriggering provides automatic selection of the trigger level of the channel A and B inputs. The trigger level defines a voltage threshold that, when crossed in a specified direction, constitutes an input event. Two modes are offered: single autotrigger and repetitive autotrigger. Trigger levels are set when a measurement cycle is restarted by the user or when a measurement or input parameter is changed. In the repetitive autotrigger mode, the trigger level is also set at the start of each block of measurements. This mode is useful in tracking amplitude variations, but it reduces the overall measurement rate because measurements are suspended during the autotrigger phase.

The autotrigger level is specified by the user as a percentage of the peak-to-peak amplitude of the input signal. The minimum and maximum peak signal amplitudes are determined by a binary search algorithm. The trigger level DAC is programmed for each level that is tested, and the output of the input comparator is directed to a ZDT counter. After a delay of approximately 1.8 ms to allow the DAC and auxiliary comparators to settle, the microprocessor performs a forced latch and records the initial value of the counter. To provide a time window for recognizing a transition of the input signal, the algorithm waits an additional 1.8 ms before latching the counter again. If the latched values differ, the trigger level is bounded by the previous and current test levels, and this process is repeated until the search converges on a minimum or maximum trigger level. If the latched values are identical, auxiliary comparators are checked to see if the trigger level is higher or lower than the value being tested.

Measurement Processing

Following an end-of-measurement interrupt, the microprocessor disables the DMA controller, ZDT counters, sequencer and gate timers. Event counts are retrieved from measurement memory, processed as requested, and saved as double-precision floating-point numbers in microprocessor RAM. Measurement processing includes the detection of overflows, calculation of a frequency or time interval result, and determination of measurement resolution.

Overflow Processing. Event counts are temporarily retained in internal latches before they are transferred to measurement memory. Two latches are coupled to each ZDT counter. Continuous time interval and single-channel frequency, period, and totalize measurements use a single latch. Other measurements require two latch signals. These measurements include dual-channel frequency, period and totalize measurements, time interval measurements, and noncontinuous measurements. This distinction is important because it affects overflow processing.

For example, event counts for single-latch and positive time interval measurements must increase with successive latches. Therefore, an overflow condition occurs if a particular event count is less than the previous event count. The overflow is resolved by adding a constant equal to one more than the maximum count value (in this case, 2^{32}) to that event count and to all latched counts that follow it. Subsequent overflow conditions are resolved with the same technique.

In other measurements, the event count does not necessarily increase. Consider the ±(time interval A→B) measurement, in which the channel A and B input signals control the first and second latches, respectively. If the channel B signal precedes the channel A signal, the event count recorded by the second latch will be less than the event count recorded by the first latch. To distinguish this situation from an overflow condition, the firmware compares the next pair of event counts with the previous pair. Since the respective event counts for each latch must increase, a discrepancy is interpreted as an overflow and is resolved in the manner previously described.

Frequency Calculation. Frequency is calculated by applying the reciprocal counting technique, based on the formula:

$$\text{Frequency} = \frac{\text{Number of Input Events}}{\text{Gate Time}}$$

In a continuous, single-channel frequency measurement, two ZDT counting chains accumulate input events and reference clock edges. Both counting chains are sampled

at a rate specified by the user. The number of input events is simply the difference in event counts between successive latches. Similarly, the gate time is the product of the difference in reference clock edges and the period of the clock. The gate time can be further refined to 200-ps resolution by including the start and stop interpolator data.

Therefore, assuming that ZDT1 counts input events and ZDT3 counts reference clock edges, the frequency associated with sample n can be expressed as:

$$\text{Frequency} = \frac{\Delta\,\text{Events}}{\Delta\,\text{Time}}$$

$$= \frac{\text{ZDT1}_{n+1,2} - \text{ZDT1}_{n,2}}{[(\text{ZDT3}_{n+1,2} - \text{ZDT3}_{n,2})(2\text{ ns})] - [(I_{stop} - I_{start})(200\text{ ps})]}$$

where $\text{ZDTi}_{m,n}$ denotes the event count recorded by ZDT counting chain i for sample m, using latch n. Start and stop interpolator counts are denoted by I_{start} and I_{stop}, respectively. Note that n + 1 latch operations are performed for a block of n measurements.

If the sampling rate is controlled by a ZDT chain, as in the time sampling or event sampling arming modes, the measurement is not continuous. In this case, both latches of ZDT1 and ZDT3 are used, to mark the start and end of each measurement. Frequency is calculated as before, except that 2n latch operations are performed for a block of n measurements.

Two-channel frequency measurements use all three ZDT chains. The channel A signal is counted by ZDT1 and the channel B signal is counted by ZDT2. As before, ZDT3 counts the reference clock signal. Channel A measurements are derived from the data saved in one pair of latches, while channel B measurements are derived from the other pair.

Time Interval Calculation. Time interval is a measurement of elapsed time between two input events. The measurement gate is controlled by two independent signals. Between these signals, reference clock pulses are accumulated. Time interval and ±(time interval) measurements are derived from the difference in the final and initial event counts, which are recorded by the pair of latches in each ZDT counter. Assuming that the ZDT3 counting chain counts the reference clock, the time interval of sample n can be expressed as:

$$\text{Time Interval} = [(\text{ZDT3}_{n,2} - \text{ZDT3}_{n,1})(2\text{ ns})] - [(I_{stop} - I_{start})(200\text{ ps})].$$

For a time interval measurement, this result will always be greater than zero. For a ±(time interval) measurement, this result may be positive, negative, or zero. These measurements are not contiguous because both latches are used. Since the ZDT3 counting chain is not reset between measurements, however, time correlation between measurements is maintained.

In contrast, the continuous time interval measurement offers continuous sampling in measurements of intervals greater than 100 ns. This is a single-latch measurement, and is calculated from the expression:

$$\text{Continuous Time Interval} = [(\text{ZDT3}_{n+1,2} - \text{ZDT3}_{n,2})(2\text{ ns})] - [(I_{stop} - I_{start})(200\text{ ps})].$$

The minimum time interval limitation of 100 ns corresponds to the maximum rate at which latched counts are transferred to measurement memory. By comparison, the time interval and ±(time interval) modes can measure smaller time intervals because there is a much smaller delay in latching the start and stop event counts into the internal latches.

Totalize Calculation. Totalize measurements count the number of events of an input channel. Unlike other measurements, totalize measurements are not synchronized to events of the input signal. Consequently, a latch signal may occur while events are cascading through a ZDT counting chain. To ensure that valid data is retrieved, the data from the ZDT counters is latched twice. If different values are latched, status bits are examined to determine which value is correct. Then, assuming that ZDT1 counts input events, the result is calculated from the expression:

$$\text{Total} = \text{ZDT1}_{n+1,2} - \text{ZDT1}_{n,2}.$$

Two-channel totalize measurements are calculated analogously to two-channel frequency measurements.

Resolution Processing. Several factors limit measurement resolution, primarily the frequency of the input signal, the gate time, and the resolution of the interpolators. To avoid overstating the resolution of a measurement result, several algorithms dynamically adjust the number of significant digits that are displayed. These algorithms return an integer value that represents the number of significant digits in the mantissa. This information is used by the numeric result formatters, which round measurement results to the specified number of digits.

The general expression for determining the number of significant digits is:

$$d = I(\log_{10} m) - I(\log_{10} s) + 1.$$

where m is the measurement result, s is the calculated resolution, and I is a function that returns the smallest integer value that is equal to or less than its argument.

For example, the resolution of a frequency measurement is given by:

$$s = (200\text{ ps})(m/t)$$

where m is the calculated frequency and t is the gate time. For instance, a frequency of 5.000 ... MHz can be resolved to 1-mHz resolution over a gate time of 1.0 second. Therefore, the number of significant digits is:

$$I(\log_{10} 5.0 \times 10^{6}) - I(\log_{10} 1.0 \times 10^{-3}) + 1 = 10\text{ digits}.$$

The displayed result is 5.000 000 000 MHz.

Display Manager

The display manager manages information directed to the built-in CRT. It transforms instrument settings into menu screens, and formats measurement results and status messages.

CRT Interface. The HP 5371A has a 7-inch-diagonal, raster CRT display with 408-by-304-pixel resolution. Each pixel is mapped to a display plane that is viewed as RAM by the microprocessor. Text and graphs are displayed by controlling bits of this memory. A pixel is turned on when the corresponding bit of the display plane is set and turned off when that bit is cleared.

Two display planes are visually superimposed to vary the intensity of each pixel. The intensified display plane is used to highlight status messages and softkey labels. This plane retains the image of the active graph, while an image of a saved graph is retained on the other display plane. Segmenting graphs in this manner permits rapid comparison, since they are displayed or blanked by enabling or disabling a display plane.

Several display procedures were developed to simplify the interface to the CRT driver circuitry. These procedures access the three types of display primitives, which are the building blocks of a screen: moves, lines, and text. These procedures provide the ability to:

- Clear the display
- Move the graphics "pen" to a specified pixel
- Draw a line from the current position of the graphics pen to another position
- Display a label
- Select the inverse video mode
- Control the intensity of a pixel.

The display procedures are patterned after the graphics commands of the HP 9000 Series 200/300 Computers. Their similarities made it possible to simulate a portion of the menu and graphics firmware on an HP 9000 Model 236 Computer before any instrument hardware was developed.

Menu Structure. An important design goal was a front panel interface tailored to the measurement technology while conforming to conventions established by previous counters. This goal implied the following guidelines:

- Instrument parameters should be visible at a glance, similar to the front panel of a conventional counter
- Parameters should be organized in a sequential and/or hierarchical structure to clarify their functions or interactions
- Menu levels should be minimized to provide direct access to parameters
- Prompts and help screens should be provided to reinforce user actions and clarify concepts.

These guidelines were used to evaluate softkey-driven and cursor-driven menu structures. A softkey-driven menu structure provides menu options at each lower level that specifically relate to options selected at previous levels. By guiding the user through options based on previous choices, it progressively converges to a limited set of options. This hierarchical structure can be perceived as a major drawback, however, because it restricts the range of user control at successive menu levels, and may require a considerable amount of menu traversal.

In contrast, a cursor-driven menu structure maintains a wide range of user control by minimizing menu levels. Related menu options are simultaneously displayed on a menu page, rather than on different menu levels. An option is modified by positioning the menu cursor to its corresponding data field and using softkeys to select other choices. Compared to the softkey-driven structure, it is loosely directed and often requires a large portion of the display area.

The cursor-driven structure was chosen primarily because it supports more descriptive menu screens. Cursor-driven menus are well-suited to describing instrument settings and showing hierarchical or chronological relationships between them. This was particularly helpful in presenting the sophisticated measurement arming capabilities of the HP 5371A (Fig. 3). Another benefit is that parameter interactions are more clearly visible. For example, conflicts between the measurement function and the arming mode can be seen without traversing through a series of softkey choices.

Parameter interdependencies also influenced the softkey structure. In the HP 5371A, valid choices for the selected parameter are mapped to softkeys. Pressing one softkey repetitively to cycle through a set of options would have been simpler to implement, but might force the user to cycle through an intermediate option that is coupled to another parameter. By permitting direct selection of each option, we did not have to be concerned about inadvertent changes to instrument settings that might result from this process.

These concepts resulted in a shallow menu structure that distributes instrument functions over eight menu pages. Each page consists of related functions such as measurement, input, or statistical parameters. To change a parameter, the user (1) presses a menu key to display the desired page, (2) uses the cursor keys to position the menu cursor, denoted by a highlighted inverse video field, at the parameter of interest, and (3) selects a new option for that parameter from a list of softkey options.

Field Processing. Menus and data screens are composed of parameter fields. A field is a segment of changeable text

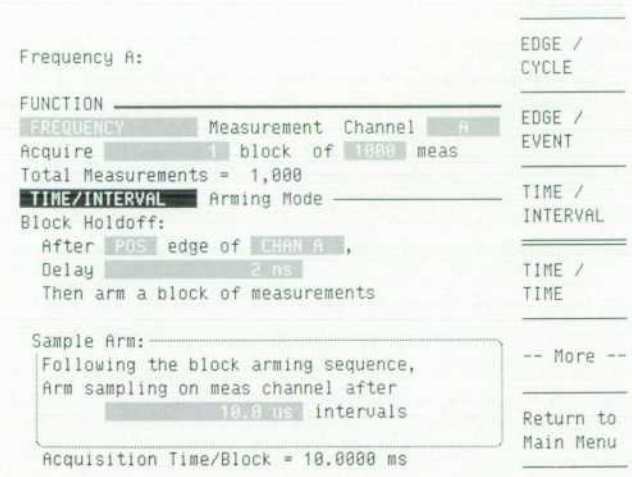


Fig. 3. *Menu for selecting the measurement function, sample size, and arming configuration. The cursor-driven structure facilitates the chronological arrangement of arming qualifiers.*

that has a predefined length and position. The function of the menu processing module is to translate instrument settings into data for these fields.

The settings of all parameters on the displayed menu are extracted from the data base. This technique is less efficient than examining only the parameter that has changed, but it isolates the menu processing module from parameter interactions. For example, selecting the measurement function may preset the measurement channel and arming mode. These interactions are resolved by the command translator, which updates the data base and sends a message to the display manager. Consequently, only one message is required and settings do not have to be specified as a parameter list.

The value of a non-numeric setting, represented as a Pascal enumerated type, provides an index into a list of character strings that describe each setting. If the setting is a numeric value, it is processed by a floating-point-to-ASCII formatter. The status of a field specifies its display characteristics, such as intensity or enhancement mode. The field's status is determined by examining related instrument settings. Modifiable parameters are displayed within inverse video fields, and the field currently pointed to by the menu cursor is highlighted.

Text and status data for every field are saved in an array. Because the field positions are invariant, they are placed in a lookup table that is mapped to each field.

Since updating an entire menu can be time-consuming, we decided to update only portions that have changed. To keep track of changes, a second array of field data is used to retain the previous menu configuration. By sequencing through corresponding elements of these arrays, differences are easily identified and displayed. Fig. 4 illustrates this process.

Softkey Processing. Softkeys are processed similarly. Two arrays define the text and status for each softkey label on the current and previous menu screens. As the menu cursor is moved, valid options for the selected instrument setting are identified and translated into data for this array.

HP-IB

HP-IB input and activities are handled by two tasks. The input task accepts characters and formats them into a string, which is passed to the command translator module for parsing and further processing. The output task coordinates the formatting and output of measurement results, status, and plotted and printed hard copy.

Programming Language. The HP 5371A's programming language is a hierarchical one, modeled after the HP Oscilloscope Language used in the HP 54100A Digitizing Oscilloscope. In this language structure, commands are divided into subsystems arranged by functional modules of the instrument. Examples include function, input, graphics, and processing subsystems. Command mnemonics are duplicated in different subsystems to provide consistency in meaning. One example is the SOURCE command, which is used in several subsystems to define the current source channel to which all parameter specifications apply. Programming is simplified through a minimal set of commands and the use of complete words as mnemonics.

In addition to the full mnemonics, short-form equivalents are accepted (e.g., SOUR for SOURCE). This flexibility allows the user to choose between readability and processing speed. For non-time-critical applications, use of the complete mnemonics provides self-documenting code, while use of the terse command set reduces HP-IB handshake time and improves throughput.

Output Formats. The output formatter translates measurement results into ASCII or floating-point formats before placing them in the output buffer. Formatting of results is confined to this single module, which coordinates the translation and packing of bytes into the output string.

To allow the user access to all displayed measurement results, including statistics, limit test status, and expanded data (missed events and gate times), the HP-IB output format parallels the current RESULTS screen display. That is, whichever result elements are displayed, those same elements are transferred over the bus. This enables the user to obtain all of the processed results easily.

The binary output mode provides direct access to the register values of time and event stamps. In addition to being the fastest method of transferring data from the HP 5371A to the controller, the binary output mode allows a larger number of measurements to be acquired and offers total flexibility in manipulating results. This mode is useful for collecting many measurements in external memory for later analysis, or for performing computations not provided in the HP 5371A.

Acknowledgments

The successful development of the HP 5371A's firmware resulted from the contributions of many dedicated individuals. The display circuitry, graphics drivers, DMA memory, and diagnostic firmware were designed by Leland Ho. Mark Wine was involved with the hardware/firmware interface, including the microprocessor system, counting circuitry, and HP-IB drivers. The measurement control and

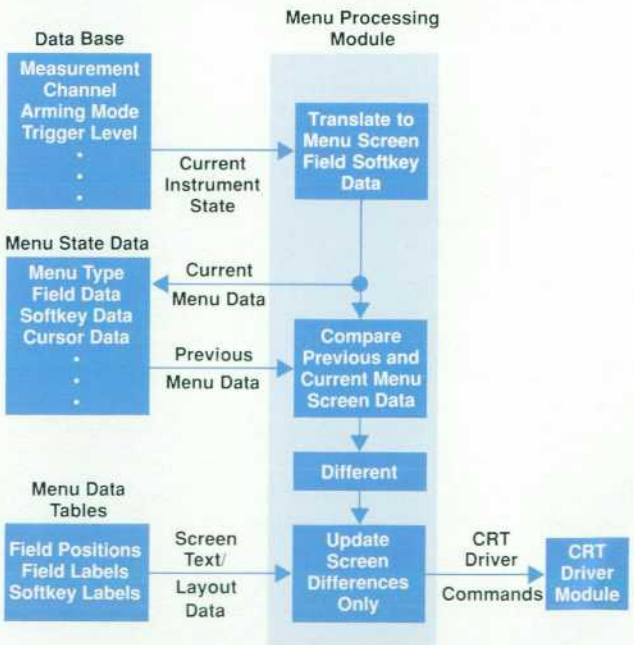


Fig. 4. *Process for generating menu screens.*

processing firmware was designed by David Block, who also contributed to the definition of the measurement set. Janelle Rose designed the graphics algorithms. Ron Capener contributed to the early definition of the firmware. We are also indebted to the ideas and feedback provided by Bruce Greenwood, product marketing engineer, who provided insights into user needs, and to Len Pilara, who spent many hours thoroughly explaining the measurement technique in the operating manual. Bob Bliven and Paul Reeder provided invaluable guidance in ergonomic issues. Finally, special thanks to the HP 51089A project team, who provided advice on processing algorithms and development tools, and to Steve Williams of the HP Signal Analysis Division who supplied the optimized math library.

Table-Driven Help Screen Structure Provides On-Line Operating Manual

The structure and firmware were designed for ease of reuse.

by Lisa B. Stambaugh

The HP 5371A HELP SCREEN feature is designed to give the novice user enough basic information to traverse the front panel, locate configuration information in the menus, become familiar with continuous count concepts, and get started using the instrument to make measurements.

The help screens are organized around a simplified instrument block diagram, allowing the user to choose an area of interest quickly. The main help screen shows this block diagram, along with softkeys for each main area (Fig. 1). The menu structure is only one level deep, allowing the user to return to the main level by pressing one key. Where possible, graphics are used to illustrate concepts, such as arming examples (Fig. 2).

Implementation

An objective for the implementation of the help screen feature was that it be easy to update, add, move, and delete screens. A table-driven structure made it easy for us to prototype screens for review and critique, rearrange screens within groups, and add new screens as needs were identified.

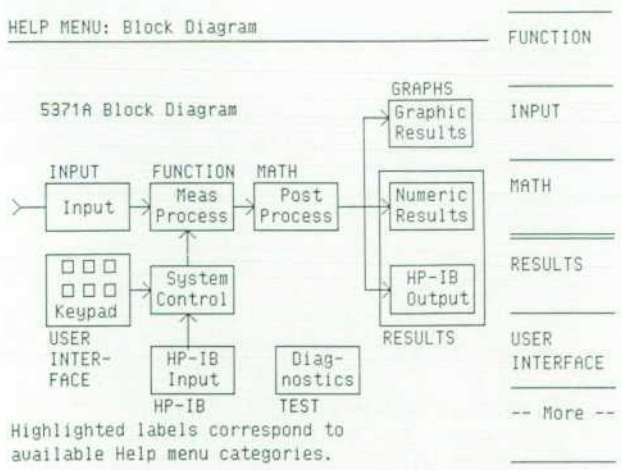


Fig. 1. *The basic HP 5371A help menu shows a block diagram of the instrument.*

In the final product, assembly language data tables contain about 80% of the 50K bytes of code required to generate the screens. Pascal code accounts for the remaining 20%, which handles operating system synchronization, reading and handling of the tables, and graphics within the screens (which could not be table-driven because of the individuality of nontext displays).

In addition to storing all screen text in tables, softkey labels, softkey highlighting, and screen titles are table-driven. The key to this structure is a single enumerated type that encompasses all screen types in a single list:

```
Help_menu_type = (H_MAIN_1,
                  H_MAIN_2,
                  H_MAIN_3,
                  H_FN_ARM_SUM_1,
                    ...
                  H_MAIN_USER_INT,
                  H_NONE);
```

This allows easy enumeration and CASE statements to deal with all screens at once. If another screen is added, another element is added to the enumerated types, and appropriate table entries are generated. It is as easy to add types in the middle of the list as at the end.

The majority of the screen text is contained in data tables that list the pixel locations and text for each line. These tables are used with a tool program, which interprets each command and calls CRT driver procedures to perform the action. In addition to text specification, the tool interprets commands to control line color, line style, inverse video, and character size.

Since all screens are generated in the same format, many

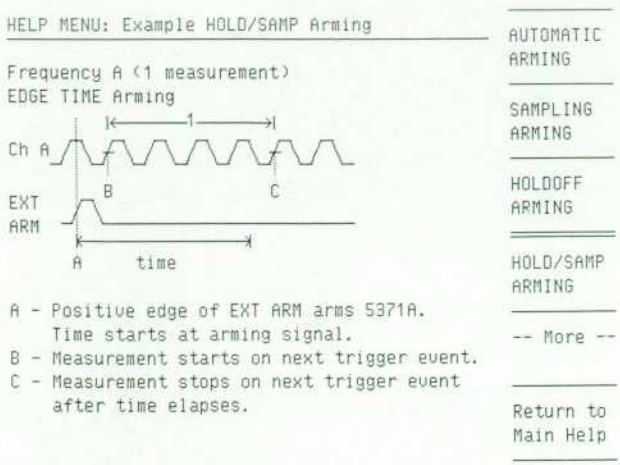


Fig. 2. *Some help screens illustrate concepts, but leave complex or lengthy explanations to the manual.*

of the initial steps required to draw a screen are duplicated for each case. A single additional table entry takes care of these actions for each screen.

```
HS_PREP_ALL  CLEAR_DISPLAY      ;clear the display
             IVOFF              ;turn off inverse video
             SET_LINE_STYLE 1   ;set solid line style
             SET_COLOR_2        ;bright green plane
             INT_MOVE 0,284     ;draw a line under the title
             INT_LINE 322,284
             SET_COLOR 1        ;color back to green
             TABLE_END          ;end of table
```

Here we can see some of the commands that are used in the data tables and are interpreted using the tool program. CLEAR_DISPLAY is interpreted to call the Pascal procedure Clear_display, INT_MOVE 0,284 results in a call to the Pascal procedure Int_move with the parameters 0 and 284, and so on. Int_move (0,284) moves to the pixel location (0,284), and Int_line (322,284) draws a line from that location to pixel location (322,284). Storage of these constants, with a single procedure interpreting them and calling the Pascal procedures, saves memory space and simplifies the screen generation process.

Separate tables control softkey labels, softkey placement within key spaces for a given screen, pointers to the next screen to be displayed when a key is pressed, which softkey is highlighted for a given screen, and the title to be displayed with that screen.

The table Help_new_skeys serves two purposes. It sets up the softkey labels associated with each screen and provides the index into the next screen when a softkey is pressed. There is a six-entry segment in the table for each of the types in Help_menu_type. The six entries are the types for each of the six softkeys. When a key is pressed, it is assigned a number between one and six. That number is used to index into the six-entry segment, giving the type of the next display screen. That type is then used to display the next set of softkeys, and so on.

```
Help_new_skeys
             DC.B H_MAIN_FN          ;main 1
             DC.B H_MAIN_INPUT
             DC.B H_MAIN_MATH
             DC.B H_MAIN_RESULTS
             DC.B H_MAIN_USER_INT
             DC.B H_MORE

             DC.B H_MAIN_GRAPHS      ;main 2
             DC.B H_MAIN_HPIB
             DC.B H_TEST
             DC.B H_NONE
             DC.B H_NONE
             DC.B H_MORE

             ...
```

In conjunction with this table, Highlight_help_tab is used to determine which softkey is to be highlighted, based on the current screen displayed. Again, it is ordered by the enumerated type Help_menu_type. The table entries are byte values from zero to six, corresponding to the key to be highlighted. Values one to six map directly to the softkeys, while zero signals that no softkey is to be highlighted for this screen.

```
Highlight_help_tab

        DC.B 0      ;main 1—no softkey highlighted
        DC.B 0      ;main 2—no softkey highlighted
        DC.B 0      ;main 3—no softkey highlighted

        DC.B 1      ;fn arm sum 1—first softkey highlighted
        DC.B 2      ;fn arm sum 2—second softkey highlighted
        DC.B 3      ;fn arm sum 3—third softkey highlighted
```

Following the same pattern, it is easy to generate tables to define softkey labels and titles, again using Help_menu_type to order the tables (irregular spacing in these strings is used to center text on two lines within the softkey space).

```
Help_key_labels

        help_str "MAIN          1"          ;main 1
        help_str "MAIN          2"          ;main 2
        help_str "MAIN          3"          ;main 3

        help_str "ARMING        OVERVIEW"   ;fn arm sum 1
        help_str "VALID ARM     OPTIONS"    ;fn arm sum 2
        help_str "VALID ARM     OPTS CONT." ;fn arm sum 3
```

```
Help_titles

        help_title_str  "HELP MENU: Block Diagram"
        help_title_str  "HELP MENU: Block Diagram"
        help_title_str  "HELP MENU: Block Diagram"

        help_title_str  "HELP MENU: Function Arming"
        help_title_str  "HELP MENU: Function: Valid Arming Options"
        help_title_str  "HELP MENU: Function: Valid Arming Options"
```

The macros help_str and help_title_str provide the ability to

generate string data tables with constant-length strings. This avoids having to declare space for blank characters to pad table entries to a constant length. The macro takes care of this automatically.

To facilitate table access from the Pascal code used to generate the menus, Pascal type declarations are used:

```
TYPE

    Help_key_line
        = ARRAY[1..6] OF Help_menu_type;
    Help_key_array
        = ARRAY [H_MAIN_1..H_NONE] OF Help_key_line;
    Help_highlight_array
        = ARRAY [H_MAIN_1..H_NONE] OF SIGNED_8;
    Help_key_lbl_array
        = ARRAY [H_MAIN_1..H_NONE] OF Chararray21;
    Help_title_array
        = ARRAY [H_MAIN_1..H_LEDS] OF Chararray45;
```

Then, when a particular table is to be accessed from a Pascal file, a corresponding variable declaration is used to define it:

```
VAR
    Help_softkey : Help_key_line;
    Help_new_skeys : Help_key_array;
    Highlight_help_tab : Help_highlight array;
    Help_key_labels : Help_key_lbl_ array;
    Help_titles : Help_title_array;
```

These variable names are exactly the ones used for the assembly language tables. They are declared globally in the assembly files, and only declared externally in the Pascal files using them.

Tools

Two tools were used extensively in developing the screens. The first is an assembly language program that reads and interprets CRT actions, which have been coded into constants in a table, using defined CRT macros. This allows all text screens to be generated via tables, without dealing with specific CRT-drawing interface issues. Use of this tool greatly reduced the code size of the help feature because the data files were reduced to simple constants and strings. Generating the screens using in-line Pascal code would have used much more ROM. The complete set of commands is fairly small, but the commands provide all of the necessary capability to generate the screens. The commands (with example parameters) are as follows:

```
SET_COLOR 2
SET_LINE_STYLE 1
INT_MOVE 0,288
INT_LINE 322,284
GTEXT "HELP MENU:"
IVON
IVOFF
SET_CHAR_SIZE 7
CLEAR_DISPLAY
TABLE_END
```

They correspond to these Pascal procedure calls:

```
Set_color (2)
Set_line_style (1)
Int_move (0,288)
Int_line (322,284)
Gtext ("HELP MENU:")
IVon
IVoff
Set_char_size (7)
Clear_display
```

CRT_INTERPRET is the assembly language procedure that interprets these commands. It is called using the following format from Pascal, which requires only the name of a pointer to a data table:

```
PROCEDURE CRT_INTERPRET (VAR table_ptr : UNSIGNED_16);
```

A string macro allows generation of string data tables with constant-length strings. This is especially useful for title strings, which are not of constant size and cover a wide range of lengths. The string macro is also used for softkey labels. Even though those labels are shorter (maximum of 20 characters), use of the string macro avoids declaring the extra padding characters within the data tables. Macros are easily generated using the following code segment. The only changes needed are the declaration of the length of the string and the macro name.

```
help_str    MACRO    &name

T1&&&&      DC.B     T2&&&& – T1&&&& – 1   ;string length
            ASCII    &name                 ;chars in the string
T2&&&&      EQU      $                     ;end mark of string
            REP      help_len – (T2&&&& – T1&&&&-1)
            DC.B     ' '                   ;add trailing blanks
            MEND                           ;end of macro
```

Benefits

The help screen structure has several benefits. It was easy to add new help screens throughout the final phases of the product. In addition, we designed the HP 5371A firmware with the intent of leveraging much of it for future products, and this structure allows flexibility in keeping some screens intact, modifying others, and adding others. A feature that helped make the implementation cleaner was the ability to develop screens independently without affecting existing code. Screens that were complete and accurate were not at all affected by changes to those under development. Furthermore, once the structure and mechanics of displaying a screen were finalized and functional, any screens under development were totally isolated from that structure. Layouts were developed by hand on a pixel grid and later converted to numeric locations and text strings.

Acknowledgments

A comprehensive set of help screens cannot be the product of one person's ideas. The feature would not have been as successful or useful without the input and suggestions

of the entire HP 5371A project team. Leland Ho deserves special thanks for writing CRT_INTERPRET. Without it, screen development would have been tedious and time-consuming, and the equivalent code to generate those screens would have been much more ROM-intensive.

Input Amplifier and Trigger Circuit for a 500-MHz Frequency and Time Interval Analyzer

Two thick-film hybrid circuits provide precise, stable high-frequency triggering in manual trigger, single autotrigger, and repetitive autotrigger modes.

by Johann J. Heinzl

THE HP 5371A FREQUENCY AND TIME INTERVAL ANALYZER operates on a signal by determining both its cycle count and the times of its zero crossings. The accuracy of the measurement depends directly upon the accuracy with which the times of the zero crossings are preserved. The function of the input amplifier and trigger circuit is to convert the analog input signal into a binary signal whose timing transitions are representative of the zero crossings of the original signal. The zero crossings can be defined around a user-selected voltage called the trigger level. The circuit design addresses the following major issues, the effects of which are seen even beyond the operating bandwidth of the instrument (dc to 500 MHz):

- Impedance conversion and probing
- Accuracy of the trigger level setting
- Sensitivity

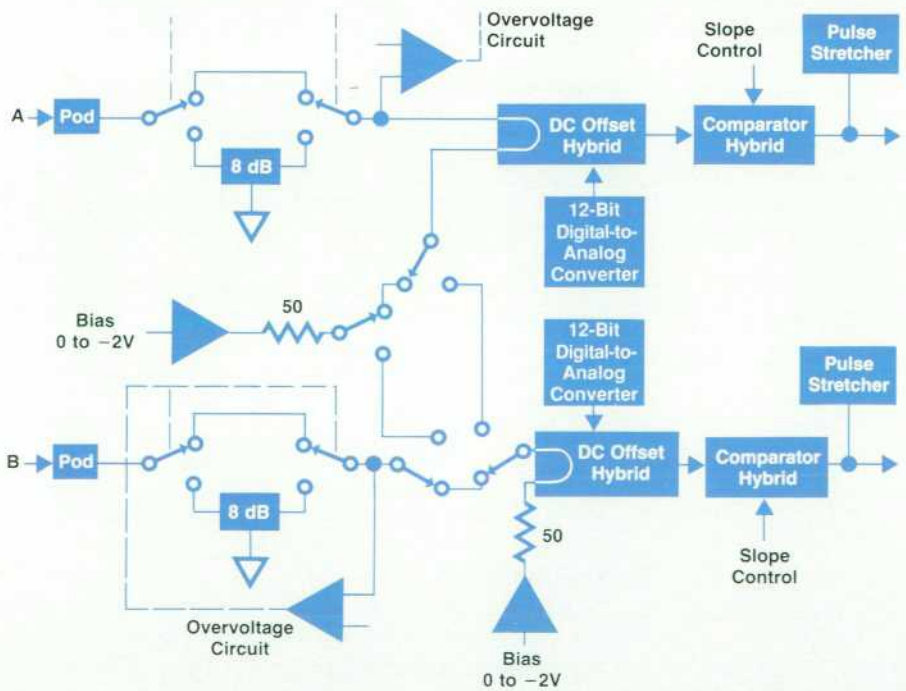


Fig. 1. *Simplified block diagram of the input amplifier of the HP 5371A Frequency and Time Interval Analyzer.*

- Preservation of timing precision at zero crossings.

Implementation

The input amplifier consists of three subblocks: the impedance conversion and input switching matrix, the dc offset hybrid and trigger level digital-to-analog converter (DAC), and the comparator hybrid and pulse stretcher. For a simplified block diagram, see Fig. 1.

Impedance Conversion and Probing

The HP 5371A offers a choice of three different input impedances: 50Ω, 10 kΩ, or 1 MΩ, using plug-in pods. These pods were designed for the HP 54100A oscilloscope series.[1] They are small plug-in units and can easily be exchanged. The pods have an output voltage range of ±2V and drive 50Ω. The outputs of the pods are connected to the HP 5371A input board by a 50-ohm coaxial cable. The model numbers are:

- HP 54001A. 10 kΩ in parallel with 2 pF, ±20V, 10:1
- HP 54002A. 50Ω input impedance, ±2V input limit (standard pod for the 5371A)
- HP 54003A. 1 MΩ in parallel with 8 pF, ±2V or ±20V with 10:1 probe.

The HP 54003A comes with a 10:1 voltage divider, which is factory adjusted to yield optimal pulse response. This is a significant advantage when used in a counter, since probe matching is difficult without a voltage-versus-time display. This probe's bandwidth is approximately 300 MHz.

The HP 5371A has two identical input channels, A and B. Measurements can be made on one input signal, on two input signals simultaneously, or with a signal applied to channel A and internally connected to both channels (the input impedance is unaffected). The configuration depends on the measurement. For instance, a frequency measurement requires only one channel but a frequency ratio measurement needs two separate channels. For a pulse width measurement, the signal on channel A is internally applied to both channels. This signal routing is accomplished by means of RF relays (see Fig. 1).

Additional features are provided when the standard 50Ω pod is used. An 8-dB attenuator can be selected to increase the dynamic and operating ranges from 2V p-p and ±2V, respectively, to 5V p-p and ±5V. The input signal can also be terminated to −2V through a 50Ω termination. This is useful for ECL signals.

DC Offset Hybrid and Trigger Level Control

The user controls the trigger level to specify the switching threshold of the HP 5371A by means of a thick-film dc offset hybrid circuit. The comparator, which follows the dc offset circuit and is described in the next section, switches at zero volts. The dc offset circuit makes it possible to offset the input signal by a known amount, thereby changing the trigger level.

Consider a pulse width measurement of a 2V p-p sine wave at the 90% level, which happens to be at +0.8V. If the user selects a trigger level of +0.8V dc, the dc offset hybrid shifts the signal by exactly −0.8V, so the 90% point is at zero volts, the comparator operating point. The offset voltage must be precisely controlled to maintain the timing accuracy of the zero crossing. Any point on the input waveform within the operating range of ±2V can be defined as the zero crossing point.

Fig. 2 shows a simplified block diagram of this circuit. The dc offset is achieved by regulating the dc current in transistor Q4 which in turn causes a proportional dc voltage drop in R6. The trigger level is generated in the 12-bit DAC

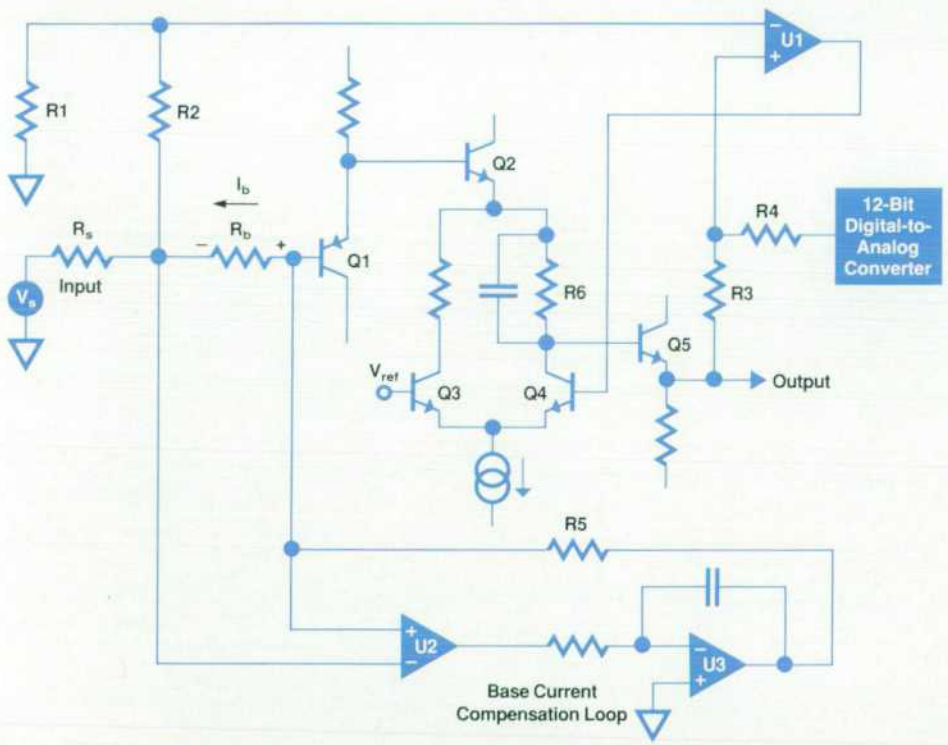


Fig. 2. *Simplified diagram of the dc offset circuit.*

and is applied to R4. The feedback mechanism consisting of R1 to R4 and op amp U1 ensures that the selected dc shift (trigger level) is precise and temperature stable. The trigger level resolution is 2 mV and the adjustment range is from −2V to +2V.

A second feedback loop consisting of R_b, R5, and op amps U2 and U3 cancels the base current of pnp transistor Q1 by sensing the voltage drop in R_b and injecting an equal and opposite current. If this base current compensation were not done, small trigger level errors would occur because the base current would develop an offset in the source resistor R_s. This offset would change with temperature and would result in different errors depending on the type of measurement. In the above-mentioned pulse width measurement, for example, it would change the width of the pulse. On the other hand, a frequency measurement is insensitive to this offset, providing that the comparator is still able to operate.

In addition to generating the trigger level accurately, it is also necessary to minimize distortion of the signal ahead of the comparator. The bandwidth of the dc offset hybrid circuit is approximately 1 GHz. Pulse overshoot and undershoot are less than ±10% when measured with a tunnel diode step generator that has less than 100 ps rise time (see Fig. 3). Typical isolation between channels is 55 dB at 500 MHz. This level of performance is necessary to achieve the 200-ps timing resolution.

Comparator Hybrid

The last major circuit block on the input board is the comparator hybrid. It consists of a diode bridge limiter with constant input resistance and a custom HP bipolar comparator IC. The comparator IC is a high-speed differential amplifier with hysteresis. The diode bridge limiter prevents the signal from exceeding the common and difference voltage limits of the differential amplifier. This ensures that propagation delay changes as a function of input amplitude are minimized, which directly affects the timing accuracy.

A phase inversion capability is also included in the comparator circuit. This can be used to specify either positive-going or negative-going transitions. The output signal of the comparator IC has logic levels of 0 and −0.8 volts (nominal).

Hysteresis is defined as the voltage the input signal has to exceed to generate change in the comparator output level (see Fig. 4). The hysteresis characteristic of the circuit resembles the familiar BH curve (flux versus magnetizing force) in magnetic cores. It is usually expressed in terms

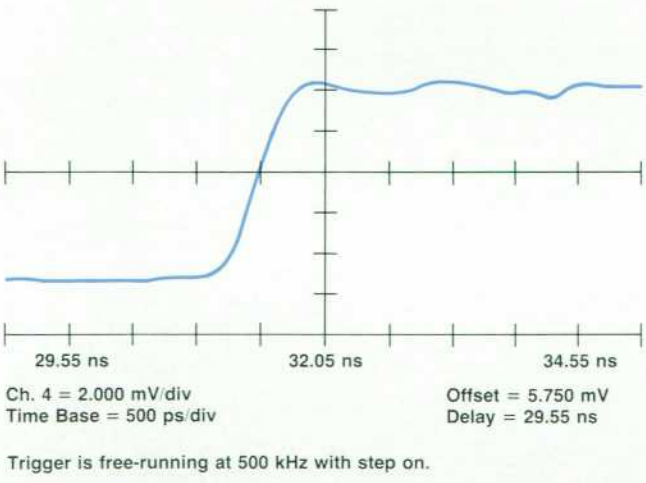


Fig. 3. *Pulse response of the dc offset circuit has less than 10% overshoot for an input pulse with 100-ps rise time.*

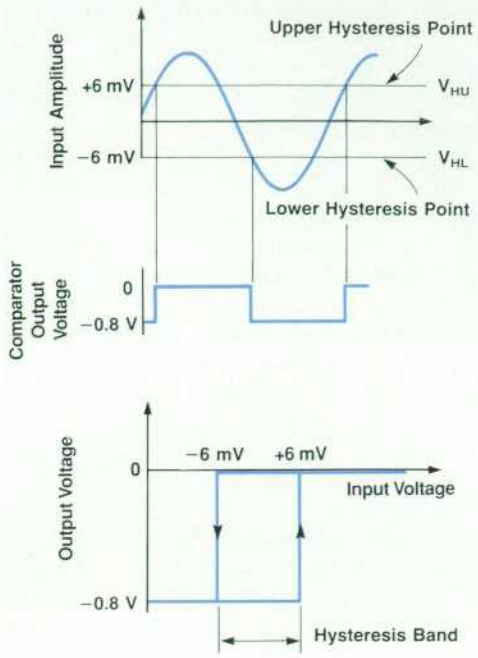


Fig. 4. *Comparator hysteresis.*

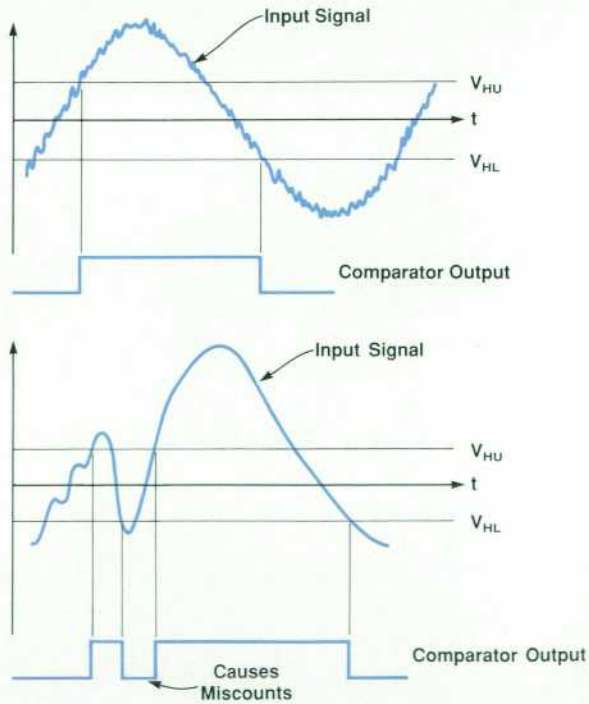


Fig. 5. *Noise greater than the hysteresis band causes miscounts.*

of millivolts at a specified frequency.

Hysteresis in the comparator IC affects two key performance specifications. The first is sensitivity, defined as the minimum signal amplitude at which the instrument meets the specifications for frequency measurements. The smaller the hysteresis the higher the sensitivity. Hysteresis also affects the noise performance. The smaller the hysteresis band the more sensitive the instrument is to noise. If the noise amplitude exceeds the hysteresis band, undesired transitions will occur (Fig. 5). Selection of an appropriate hysteresis band is always a trade-off between proper operation with very low-level signals and undesired transitions caused by triggering on noise. In the HP 5371A the sensitivity is 15 mV rms.

The comparator circuit is implemented as a thick-film hybrid to get the best high-frequency performance.

Trigger Considerations

There are three methods of setting the trigger level: manual trigger, repetitive autotrigger, and single autotrigger.

The manual trigger mode requires knowledge about the signal levels to select an appropriate trigger level. The signal to be measured does not have to be repetitive.

The input amplifier printed circuit board also drives an LED on the front panel, which flashes if the comparator output is switching. This helps the user find a good trigger level in the manual mode. Switching is detected using a pulse stretcher, which is basically a nonretriggerable one-shot (monostable multivibrator) that triggers on a positive edge from the comparator circuit.

In the repetitive autotrigger mode the trigger level is automatically set to $(V_{peak+} + V_{peak-})/2$. The signal must have no amplitude modulation and be repetitive with a frequency above 1000 Hz. A description of the algorithm can be found in the article on page 13. The autotrigger subroutine is followed by the measurement routine, and the two routines alternate thereafter.

The single autotrigger mode is new for HP counters and timing analyzers. It is provided because the HP 5371A has the ability to perform measurements continuously. The trigger subroutine is performed only once to set the correct trigger level and is followed by measurement cycles only. This offers the convenience of automatically setting the trigger level and freezing it. This improves throughput and keeps the trigger level constant over subsequent measurement blocks.

Caution must be used when using single or repetitive autotrigger on waveforms that have overshoots, since these routines search for the peak of the waveform. If the waveform to be measured has a 10% overshoot, the routine will conclude that the peak-to-peak amplitude is 120% and will calculate the trigger setting based on that information. In many cases this will not impact the measurement accuracy, but in some cases, for example rise and fall time measurements, it can lead to incorrect results. The analog circuits in the HP 5371A are designed to have excellent step response characteristics to minimize these effects.

Summary

The high performance of the input amplifier is mainly a result of optimizing the important specifications such as step response, cross talk between channels, noise floor, timing precision at zero crossings, and others. Circuit models were constructed for most of the components and the design was simulated and optimized using the HP Spice program.

Acknowledgments

The author would like to acknowledge Art Muto for his many contributions in terms of definition and circuit ideas, Jong Kim who designed the two thick-film hybrids, and Ken Rush and Ed Evel of the HP Colorado Springs Division for their many valuable ideas.

Reference

1. K. Rush, et al, "High-Performance Probe System for a 1-GHz Digitizing Oscilloscope," *Hewlett-Packard Journal*, Vol.37, no. 4, April 1986, pp. 11-19.

Phase Digitizing: A New Method for Capturing and Analyzing Spread-Spectrum Signals

By continuously counting and time-tagging zero crossings, a phase or time encoded signal can be digitized and analyzed with efficiency and precision.

by David C. Chu

PRECISE AMPLITUDE MEASUREMENT has been the basis of many traditional instruments. Oscilloscopes, spectrum analyzers, power meters, and voltmeters all focus on precise analog voltage as their basic measurement. Even a vector analyzer, which measures both amplitude and phase, does so by measuring two analog voltages: the I and Q modulation components.

This preoccupation with precise analog amplitude in instrumentation is at variance with modern modulation methods, which tend to deemphasize analog amplitude modulation in favor of frequency, phase, and time modulation for more reliable communication. Not only ordinary FM, PM, and pulse width modulation, but also modern FSK, PSK, and QPR in communications and Barker (binary phase), polyphase, and chirp modulation in radar, all deemphasize amplitude in favor of time-based parameters. Even QAM signals contain more information in the phase than in the amplitude. For these signals, fidelity is characterized by precision in frequency, phase, and time. Interest in amplitude is often connected with studying dropouts only.

The frequency agile signal, which is active over a wide frequency range in a short period of time, poses another challenge to many established practices in instrument design. Techniques such as quasistatic range switching to cover different frequency bands, searching for a signal with a sweeping narrowband receiver, or taking time to phase-lock to a signal are no longer appropriate with the agile signal. Another difficulty is that some signals are not repeatable, at least not well enough to be measured by equivalent-time techniques.* The output of a pulsed VCO and the waveform of one oscillator acquiring lock with another are examples of signals that are different on each occurrence. Often, it is statistical variations, such as jitter, and not just an average value. that are of interest. These can only be measured with repeated single-pass measurements, even though the signal is nominally repetitive. Measurement methods that require multiple passes for one measurement are not applicable.

In short, traditional narrowband, amplitude-based instruments requiring repetitive signals are suddenly found wanting when confronted with frequency agile, time-encoded or phase-encoded signals that do not repeat.

*For example, equivalent-time sampling builds a picture of a repetitive waveform by sampling it at a slightly different time in its cycle on each repetition.

Amplitude Digitizing

The waveform recorder (or high-sampling-rate digitizing oscilloscope) is wideband and can capture a transient spread-spectrum signal by digitizing its voltage, taking samples regularly spaced over time. While this is a good way to view the waveform, it is a difficult and inefficient method of characterizing the signal's modulation fidelity.

Consider a modulated signal with an agile frequency carrier randomly hopping about. To capture the waveform without aliasing requires sampling above the Nyquist rate. For such a signal, the sampling rate must be more than twice (closer to four times in practice) the highest carrier frequency plus modulation. Since the modulation (information) bandwidth is always less, and often much less, than the waveform bandwidth, sampling for full waveform recovery is grossly inefficient, resulting in unnecessary data bulk, if only the information is of interest. Apart from this, processing the data to uncover the modulation is a complex and difficult process since the data is in the form

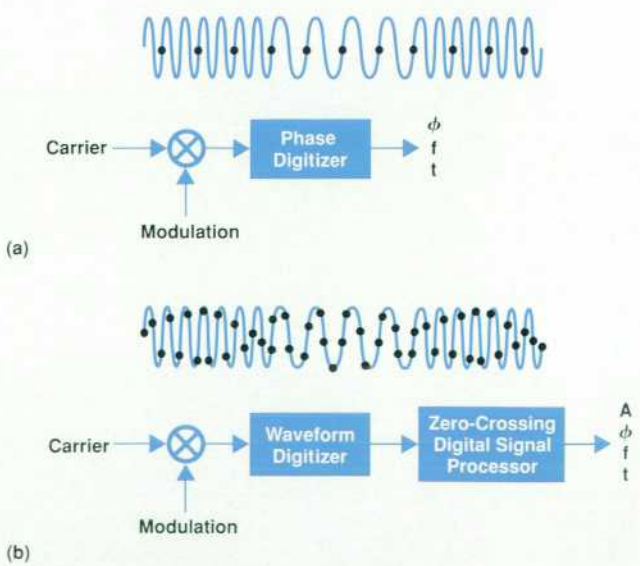


Fig. 1. *Sample locations for (a) phase digitizing and (b) amplitude digitizing.*

of voltage, that is, $v(t_i)$, where $v(t)$ may take on the form

$$v(t) = [V_o + V_a(t)]\sin[2\pi f(t)t + \theta(t)]$$

and any of the three—the amplitude $V_o + V_a(t)$, the frequency $f(t)$, and the phase $\theta(t)$, may change with time depending on the modulation type. Selecting the optimum band-limited functions $V_a(t)$, $f(t)$, and $\theta(t)$ to fit a set of $v(t_i)$ means iterative curve fitting to trigonometric functions with complicated and possibly discontinuous arguments, an incredibly messy operation "prone to undersirable convergence behavior because of the nonlinear behavior relationship between parameters."[1] This is true when V_a, f, and θ are constants. The complexity of fitting to variable parameters is simply staggering.

A less sophisticated but more effective approach is to find the locations of peaks and zero crossings. For a properly band-limited signal, interpolation between samples using digital signal processing allows computation of a voltage given a time. Computing a zero crossing time is the reverse; one computes a time given a voltage (zero). For slow-slewing signals, a combination of (sin x)/x interpolation and linear approximation may enhance the resolution of zero-crossing time measurements below one sampling period. Using this technique and counting the crossings in software, Nichols[2] simulated a frequency counter with a variable gate for frequency profiling. Unfortunately, for binary-voltage fast-switching signals, there is no timing resolution enhancement, the precision voltage measuring capability of the waveform recorder is not really being used, and the resulting counter performance is poor.

Phase Progression Digitizing

The signal digitizing method used in the HP 5371A Frequency and Time Analyzer is based on continuous counting and sampling only at signal zero crossings. The method bypasses the two extra steps, voltage digitizing and voltage-to-phase conversion, and directly digitizes the phase progression of the signal. The procedure may therefore be appropriately called "phase progression digitizing," or "phase digitizing" for short. Amplitude information is discarded. Because the data is already in the form of phase and time, trigonometric functions are totally avoided, replaced by simple functions like straight lines and parabolas, making analysis simple for even moderately complex modulation.

Phase digitizing is illustrated in Fig. 1a, which shows the sample locations on a sinusoidal signal of changing frequency. Samples occur at occasional upcrossings at a relatively constant rate. Each sample produces two increasing numbers, the total cycle count and the time stamp at that point. The cycle count comes from reading, on the fly, a counter counting the signal. Every cycle is counted, not just those on which a sample occurs. When a sample does occur, that particular cycle's upcrossing is gated by a synchronizer for time stamping, which is accomplished by reading (also on the fly) another counter counting a 500-MHz time base clock. With interpolation, described in the article on page 35, the resolution is improved from 2 ns to 0.2 ns. In counter jargon, the cycle count is called the event number and the time stamp is simply called time.

In contrast, amplitude digitizing is illustrated in Fig. 1b. The samples are regular in time and they fall on the signal wherever they may, not necessarily at zero crossings. There are always more samples in voltage digitizing than phase digitizing. There must be several samples per signal carrier cycle in voltage digitizing, regardless of the modulation. In phase digitizing, it is the other way around. There are usually several cycles per sample. The sampling rate must only exceed the Nyquist rate for the modulation, regardless of the carrier frequency.

The event number in phase digitizing is a measure of the phase progression of the signal. Because every cycle is counted, each event means the signal has progressed by one cycle, that is, 2π radians or 360 degrees. Consider a modulated signal $s(t)$ of the form

$$v(t) = A\sin(2\pi\phi(t))$$

where $\phi(t)$ is monotonically increasing. Sampling at upcrossings means samples are taken only at integer values of $\phi(t)$. The ith event sample e_i and the ith time sample t_i bear a simple mathematical relationship:

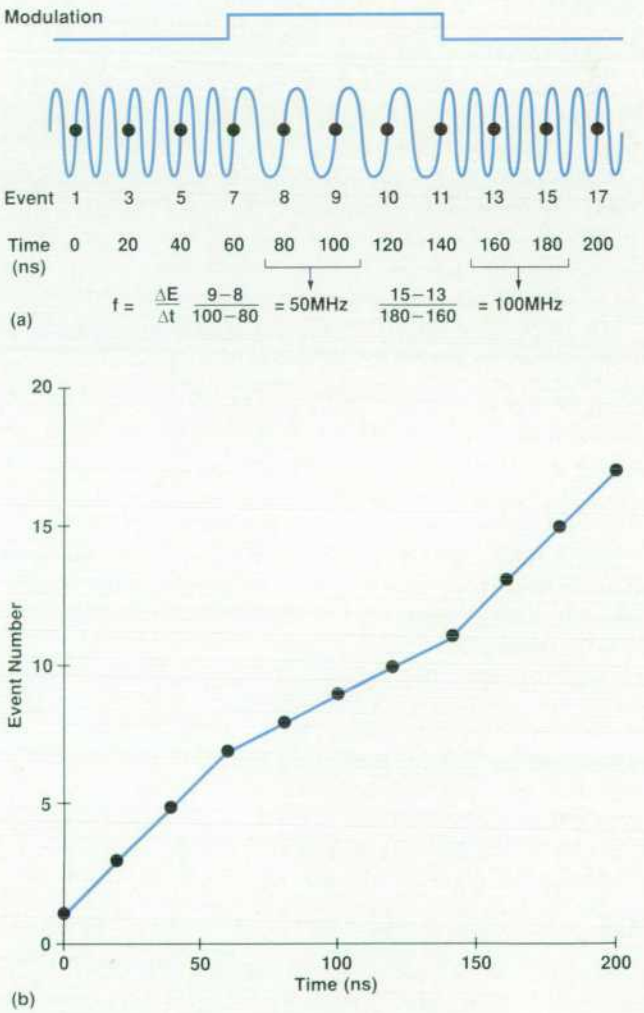


Fig. 2. *(a) Sample locations and recorded data for phase digitizing a BPSK signal. (b) A phase progression plot of the data.*

$$\phi(t_i) = e_i.$$

Both $\phi(t_i)$ and e_i are integer valued, and t_i is quantized to 200 ps. In short, the function $\phi(t)$ is digitized, although not uniformly in time as is customary.

Phase Progression Plot

In Fig. 2a, the samples obtained from phase digitizing a BPSK signal are shown with their t_i and e_i values. In Fig. 2b they are plotted using t_i and e_i as the X and Y coordinates. The two different frequencies are manifested in the apparently piecewise linear figure. The slope $d\phi(t)/dt$ of the line shown in the figure is the derivative of the phase function $\phi(t)$ and therefore indicates the frequency of the signal. Notice that the sudden change in signal frequency is effortlessly handled by the phase digitizing process. The sample rate is relatively constant, and the only clue indicating a frequency increase is a larger difference between consecutive event numbers.

A "modulation-domain" plot, a term coined by an enthusiastic colleague for a plot of instantaneous frequency as a function of time, can be obtained by digital differentiation of the event/time function. The simplest instantaneous frequency estimate is obtained by dividing the change in event number by the change in time from two adjacent samples, that is,

$$f_1(t_i) = \frac{e_{i+1} - e_i}{t_{i+1} - t_i}$$

A more sophisticated estimate is obtained by fitting a parabola through three consecutive points. The derivative of the parabola at the middle point is the instantaneous frequency there.

$$f_2(t_i) = \frac{(t_{i+1} - t_i)f_1(t_{i-1}) + (t_i - t_{i-1})f_1(t_i)}{t_{i+1} - t_{i-1}}$$

Neither of these digital differentiation methods depends on uniformly sampled data. Other differentiation algorithms can be used, generally trading bandwidth of the f(t) function for noise reduction.

In general, for differentiation using 2M points, one can use the following mathematical structure, choosing a set of weights $\{\lambda_i\}$ to achieve the desired filter response. The average frequency $\bar{f}$ at the average time $\bar{t}_i$ can be expressed as:

$$\bar{f}(\bar{t}_i) = \sum_{j=1}^{M} \lambda_j \left[\frac{e_{i+j} - e_{i-j+1}}{t_{i+j} - t_{i-j+1}} \right]$$

where the average time $\bar{t}_i$ is given by:

$$\bar{t}_i = \frac{1}{2} \sum_{j=1}^{M} \lambda_j (t_{i+j} + t_{i-j+1}).$$

The weights λ_j sum to unity. For odd numbers of points, $2M+1$, one can simply change $i-j+1$ to $i-j$ in the subscripts.

Linear Curve Fitting

Sometimes the average frequency is taken over the entire data array for an overall average.

Fig. 3 shows a phase progression plot of corrupted data. The corruption may be from many sources, such as quantization noise or modulation. The average frequency is to be determined from the corrupted data. One method, used traditionally by frequency counters, is to find the slope of the line joining the two end points, since intermediate points are not available. The estimate $\bar{f}_{ctr}$ in phase digitizing notation is:

$$\bar{f}_{ctr} = \frac{e_N - e_1}{t_N - t_1}$$

Better methods are available from statistics to find a linear best fit to the data. The slope of the fit is the average frequency estimate, and the Y intercept gives the phase relationship with other signals. A least-squares linear fit, which minimizes the sum of the squared error terms is given by:

$$\bar{f}_{lsq} = \frac{N\Sigma e_i t_i - \Sigma e_i \Sigma t_i}{N\Sigma t_i^2 - (\Sigma t_i)^2}$$

This estimate is computation intensive, involving sums of products and of squares.

Another linear fit that is almost as good, but much easier to compute than the least-squares fit is the bicentroid method. It does not involve squares or products. It partitions the number of samples into three roughly equal groups chronologically. The first group and the last group have the same number, say Q, of samples. The middle group is ignored. One then proceeds to compute the average e and average t for the two groups. The average frequency is estimated by the line joining these two "centroids." With equal numbers in each group, the normalizing factor (1/Q) for each term cancels out. The calculation is reduced to sums of data and one division,

$$\bar{f}_{2ctd} = \frac{\Sigma e_{\text{last Q}} - \Sigma e_{\text{first Q}}}{\Sigma t_{\text{last Q}} - \Sigma t_{\text{first Q}}}$$

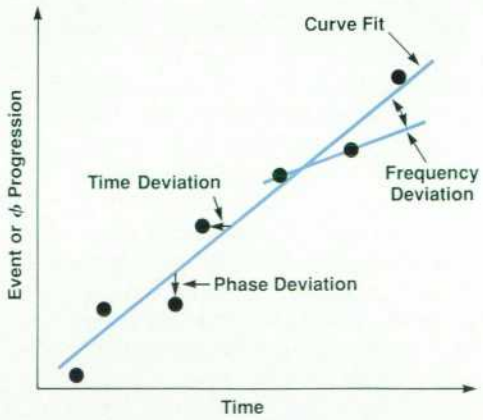


Fig. 3. *A straight-line fit to noisy data, showing phase, time, and frequency deviations.*

In the general equation given above, this corresponds to a choice of λ_j proportional to $t_{i+j} - t_{i-j+1}$ at the two ends of the data points and zero at the middle, where i is at the array center.

The average phase is best estimated by the average of all of the e_i and t_i, that is,

$$e_0 = (1/N) \sum_{i=1}^{N} e_i$$

$$t_0 = (1/N) \sum_{i=1}^{N} t_i$$

The equation for the curve fit $\hat{e}(t)$ becomes

$$\hat{e}(t) = e_0 + \bar{f} \times (t - t_0).$$

The frequency estimate $\bar{f}$ is obtained by one of the above methods or by any other method the user feels appropriate. We use $\bar{f}_{2ctd}$.

How Good Are the Estimates?

The standard deviations of these estimates can be computed by making some assumptions. We assume that the random time quantization error q is the predominant contributor of error, and that the sampling rate is approximately uniform. The standard deviation of these estimates can be expressed in terms of q, the gate time g (measurement time), and the number N of samples taken. Standard deviations of the fractional frequency for the three estimates—traditional counter, bicentroid, and least-squares—are:

Traditional counter:	$\sigma_{\delta f/f} = (q\sqrt{2})/g$
Bicentroid:	$\sigma_{\delta f/f} = (q\sqrt{13.5})/(g\sqrt{N})$
Least squares:	$\sigma_{\delta f/f} = (q\sqrt{12})/(g\sqrt{N})$

The timing quantization error q for the HP 5371A is 200 ps.

The smaller the standard deviation, the better the estimate. Notice a dramatic reduction of standard deviation for the bicentroid method over the traditional counter method for large N. Note also the meager improvement of the least-squares fit over the bicentroid fit, considering the extra amount of computation it requires.

Modulation Computation

Computing average parameters is one function of the curve fit. Another very important function is the computation of deviations, both intentional (modulation), and unintentional (error). Deviation is the difference between the data points and the curve fit. Three main kinds of deviation are of interest (see Fig. 3):

- Phase. The vertical distance from a data point to the curve is phase deviation. By computing this for every point, phase deviation as a function of time is obtained:

$$\delta_\phi(t_i) = e_i - \hat{e}(t_i).$$

- Time. The horizontal distance from a point to the curve is a time deviation. By computing this for every point, a graph of time deviation as a function of time is generated:

$$\delta_t(t_i) = -\delta_\phi(t_i)/\bar{f}$$

- Frequency. Subtracting the derivative of the curve fit from the instantaneous frequency produces frequency deviation as a function of time. When measuring a frequency modulated (FM) signal, the slope of the linear curve fit gives the carrier frequency and the frequency deviation is the modulation. Frequency deviation as a function of time is:

$$\delta_f(t_i) = f(t_i) - \bar{f}.$$

Frequency Agile Carrier

The measurement of modulation in an agile carrier is demonstrated in Fig. 4, which shows FSK modulation on a carrier hopping between 10 and 500 MHz. The top graph shows the instantaneous frequency within the 140-μs time frame. The dehopped FSK signal is shown in the bottom graph. The signal was phase digitized by the HP 5371A Frequency and Time Analyzer in one pass into 1000 data points, and analyzed by an external controller in accordance with the method described here.

Pulsed Frequency Bursts

Fig. 5 shows phase progression plots for two pulse bursts separated by a region of no activity. This signal generates two lines separated horizontally by a gap. This example illustrates a significant advantage of phase digitizing over waveform digitizing: there are no samples where there is no signal—a direct consequence of signal-triggered sampling. No memory is wasted on empty regions. This is particularly important for narrow pulse bursts at low pulse repetition rates, or for sparse transient capture with limited memory. A waveform recorder would fill the memory up with mostly zero data. On the other hand, should noise be present in supposedly quiet periods, samples will be generated.

Frequency Chirp

A chirp is a signal whose frequency varies linearly with

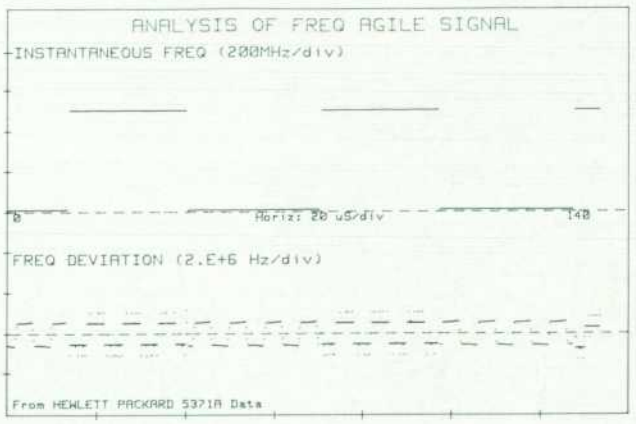


Fig. 4. *A measurement of FSK modulation on a frequency-hopped carrier.*

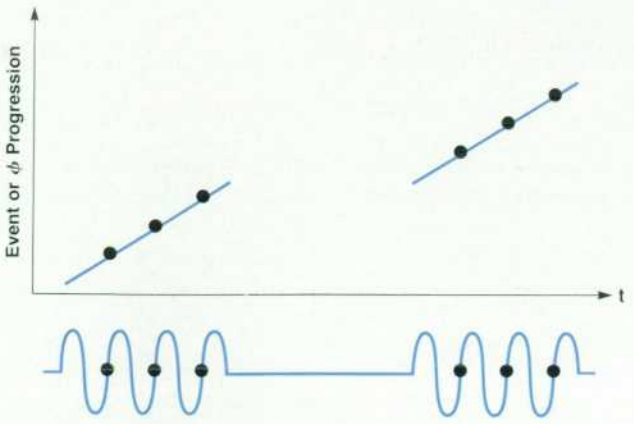


Fig. 5. *Phase progression plot for pulsed RF.*

time. Since phase is the integral of frequency, the phase progression of a chirp is quadratic with time. An ideal chirp would produce a parabola on the phase progression plot (Fig. 6). To measure chirp nonlinearity, the data can be compared with a best-fit parabola. The least-squares quadratic fit is well-described in the literature. It requires even more lengthy computations than the linear least-squares fit. A considerably simpler method, suggested by a colleague, partitions the data into five approximately equal groups chronologically, making the first, third, and fifth groups equal in number, say Q, where Q is about N/5. Assuming that the quadratic fit is given by:

$$\hat{e}(t) = at^2 + bt + c,$$

the coefficients a, b, and c can be computed from the following three equations:

$$a\sum_{1st\,Q} t_i^2 + b\sum_{1st\,Q} t_i + Qc = \sum_{1st\,Q} e_i$$

$$a\sum_{3rd\,Q} t_i^2 + b\sum_{3rd\,Q} t_i + Qc = \sum_{3rd\,Q} e_i$$

$$a\sum_{5th\,Q} t_i^2 + b\sum_{5th\,Q} t_i + Qc = \sum_{5th\,Q} e_i$$

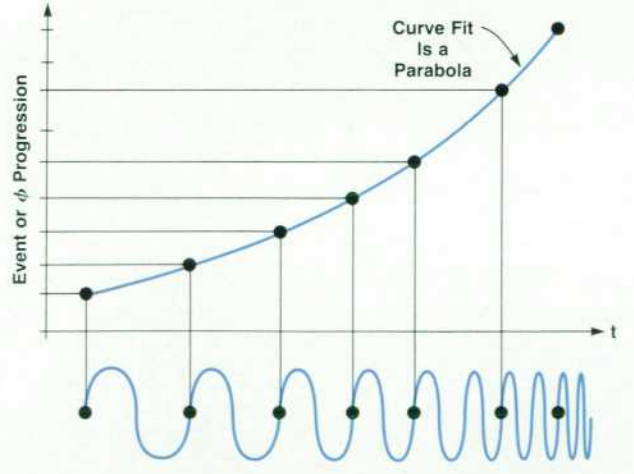


Fig. 6. *Phase progression plot for a linear frequency chirp.*

There are sums of squares, but no cross products. The solution is straightforward and is not reproduced here. The second and fourth groups are ignored.

Once the coefficients a, b, and c are computed, the usual deviation computation can be performed to generate chirp nonlinearity parameters.

- Phase nonlinearity $\delta_\phi(t_i)$ is given by $e_i - \hat{e}_i$, or

$$\delta_\phi(t_i) = e_i - (at_i^2 + bt_i + c).$$

- Time nonlinearity $\delta_t(t_i)$ is given exactly by:

$$\delta_t(t_i) = t_i - t_i'$$

where t_i' is the smaller root of the equation $ax^2 + bx + c = e_i$, or approximately by $-\delta_\phi(t_i)/(\text{curve-fit slope})$, that is,

$$\delta_t(t_i) \approx (at_i^2 + bt_i + c - e_i)/(2at_i + b).$$

Advanced phase means earlier in time, so $\delta_\phi(t_i)$ and t_i are always opposite in sign. Time nonlinearity is interpreted as the difference in time between the zero crossings of the signal and the zero crossings of the ideal chirp.

- Frequency nonlinearity is given by the difference between instantaneous frequency and the derivative of the quadratic fit:

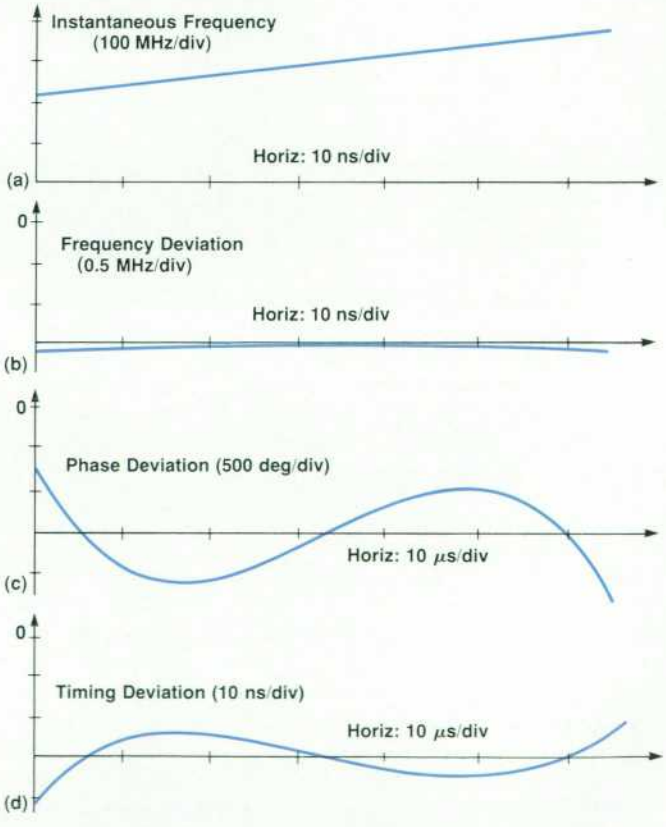


Fig. 7. *Analysis of a chirp signal using HP 5371A data. (a) Instantaneous frequency superimposed on the derivative of the best-fit parabola, which is not distinguishable because the chirp is nearly ideal. (b) Frequency deviation. (c) Phase deviation. (d) Time deviation.*

Reading a Counter on the Fly

A traditional binary ripple counter consists of a series of cascaded divide-by-two stages. With the frequency halved after each stage, the maximum count rate is effectively determined by the first few stages. Subsequent stages do not affect the maximum count rate, so they can be slow and numerous. During counting, carries ripple through a varying number of stages with different propagation delays. In a high-speed counter, the switching times of some of the slower stages may span many input cycles. For these reasons, there are few instants when all the bits are correctly aligned for reading. Therefore, to read such a counter accurately, it is necessary to slow down counting drastically if not stop it altogether.

With synchronous counting, a logic network adds 1 to the current count state so that the next event clocks the counter to the next state immediately. With careful design, the synchronous counter can be read on the fly, since all the bits theoretically change at once. With long count chains, however, the logic network becomes complex and unwieldy, and processing time lengthens. With a large fan-in, clock skew becomes more significant, causing switching to be farther from simultaneity. The maximum count rate is reduced and difficulty in reading increases. Since every bit must be as fast as the fastest bit, power consumption may soar. To summarize, short synchronous counters can count rapidly and can be read on the fly, but long synchronous counters are impractical and performance deteriorates.

Zero-Dead-Time Counter

The object is to design a binary-coded counter whose maximum count rate is independent of counter length like the ripple counter, but one that can be read on the fly repeatedly like the synchronous counter. The maximum sampling rate should be fast and independent of the input signal frequency. Fig. 1 shows such a counter with its data latch for capturing the counter contents in response to a read command.

In this example, the first M bits of the counter is a binary-coded synchronous counter with an M-bit latch. The Mth or most-significant bit is used to drive a slower binary counter in tandem. This slower binary counter may be either a ripple or a synchronous type. It has its own data latch register.

Upon a read command, the first M bits are latched in a straightforward manner from the synchronous counter. The read command, however, does not directly activate the latch for the slow counter. Instead, it is ANDed with a pulse generated by a monostable multivibrator triggered by the same Mth bit of the fast counter. A read command that falls within the pulse duration is inhibited by this pulse. Reading of the slow counter is effectively delayed until the pulse ends. Read commands that arrive at other times will activate the second latch immediately. An R-S flip-flop prevents multiple latching of the second counter by the rhythmic action of the monostable vibrator.

The reasoning behind this design is as follows. Since the second counter is triggered by the Mth bit, any motion will begin after the trigger and subside after a suitable length of time t_r. Reading of the counter should be postponed until this time passes. The one-shot, triggered by the same Mth bit, is designed to ensure just that by inhibiting reading during this critical period. The one-shot pulse width t_p is designed to be longer than the settling time but shorter than 2^M times the period of the maximum input frequency. The minimum M that satisfies this with a comfortable margin is chosen. The following summarizes the timing relationships:

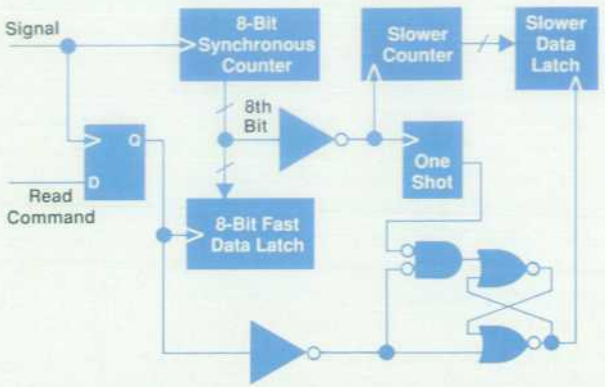


Fig. 1. *A high-speed part-ripple counter that can be read on the fly.*

$$t_r < t_p < 1/f_{s\,max}$$

$$t_p < 2^M/f_{max}$$

where t_r = settling time of the second counter
t_p = pulse width of the monostable vibrator
$f_{s\,max}$ = maximum sampling rate
f_{max} = maximum counting frequency
M = number of stages in the synchronous counter.

With this design, the objective is achieved. The maximum count rate is unaffected by the speed or length of the second counter, yet the entire counter can be read on the fly in normal binary code. Notice that as long as the sampling rate is below $1/t_p$, a new valid reading is guaranteed at the next event after each read command. A forced latch feature makes even the one event optional in case the signal has ceased. As a result, the allowable sampling rate is independent of the input signal. In contrast, some on-the-fly reading schemes (e.g., reference 1), require the sampling rate to be below a fixed submultiple of the current signal frequency. Such signal dependency makes these schemes unsuitable for measuring agile signals when the frequency is totally unknown.

The penalty for a slow second counter is a low maximum sampling rate. Therefore, the second counter should be designed to meet whatever top sampling rate is required.

We call this assembly a zero-dead-time counter or continuous counter since no dead time is created in signal monitoring when it is being read. In phase digitizing, two such counters are used: one monitoring the signal (event) and the other a time base clock (time). The synchronized edge from the event counter is used for the read command of the time counter. The two counters then produce the event and time data, respectively. Because the two are synchronized, the time data always reflects the time of the exact cycle given by the event counter.

Samples produced this way are not uniform in time, but the time data gives the actual times precisely for all samples.

Reference

1. J.O. Horsley, *High-speed counter with reliable count extraction system*, U. S. Patent 3,982,108, Sept. 21, 1976.

$$\delta_f(t_i) = f(t_i) - (2at_i + b)$$

where instantaneous frequency is computed as before.

Fig. 7 shows a chirp analysis from a measurement with an HP 5371A. The signal was sampled 700 times in one pass and analyzed. Fig. 7a shows the instantaneous frequency superimposed on the derivative of the fitted parabola, which is not really distinguishable since the chirp is near ideal. The signal chirped from 210 MHz to 370 MHz in just under 70 μs. Fig. 7b shows frequency deviation, Fig. 7c shows phase deviation, and Fig. 7d shows time deviation. Nonlinearity is clearly visible in the phase and time deviation plots.

Phase-Shift Keying

A BPSK signal is a constant-frequency signal, but its phase shifts by 180 degrees with a binary modulating signal. On the phase progression plot, such a signal generates a straight line of a constant slope which is periodically shifted down (or up) by half a count (see Fig. 8). Since BPSK signals phases are modulo 2π, 360 degrees is the same as zero degrees, and the analysis treats it as such. The curve fit for BPSK is a system of parallel lines alternating between 0 and ±180 degrees. Data points are mapped to the nearest line for demodulation. Deviation parameters are computed from differences between the data and the curve fit as usual. Generalization to MPSK is immediate: for QPSK, the parallel lines are 90 degrees apart, and so on. Fig. 9 shows an 8PSK signal switching among the eight allowable phase states. The phase deviation graph shows the switching to be within the resolution of the instrument. Phase resolution is computed as 360qf degrees, where q is 200 ps and f is the carrier frequency. The signal was captured in one pass, taking 1000 phase digitizing samples. The 10-MHz carrier frequency is derived from the data by curve fit. The phase resolution is 0.72 degree.

PSK signals may also be pulsed, as in the case of a pulsed Barker-coded (BPSK) radar signal. Such signals can be captured and analyzed in a similar manner.

Frequency Coverage Extension

The counting range for the HP 5371A is 0 to 500 MHz, and this represents the dynamic frequency range an agile carrier can operate over and still be measurable. With a frequency down-converter, this frequency agility range can be moved to microwave regions. Heterodyning preserves phase and translates frequency, so frequency-coded and phase-coded signals can be demodulated after down-conversion. The signal illustrated in Fig. 4 was down-converted from 3 GHz. The HP 5364A Microwave Mixer/Detector, for example, allows measurement from 2 to 18 GHz with a 500-MHz dynamic range by heterodyning with a stable local oscillator. The frequency of the local oscillator is arbitrary as long as the agility range is brought to within the relatively wide counting range of the HP 5371A. Efficient digitizing can begin immediately, before demodulation, with no data loss. Actual demodulation is accomplished by software homodyning with a computed IF carrier, operating on captured data. A hardware IF local oscillator is not needed.

In contrast, conventional carrier-recovery demodulation requires heterodyning to a narrow IF band. Using a Costas phase-locked loop,[3] a coherent IF carrier is generated. Only by hardware-homodyning the signal with the generated carrier can digitizing of the demodulated signal begin. With an agile carrier, it is almost impossible to heterodyne to a narrow IF window for all the channels. Also, significant data loss may result during the process of carrier-recovery phase locking.

The measurable frequency agility range is limited by the maximum count rate of the HP 5371A to 500 MHz. Using a high-frequency prescaler, the range can be extended. For example, a high-speed divide-by-four prescaler can quadruple the range of the instrument, that is, the 500-MHz range of the HP 5371A can be extended to 2 GHz. In contrast, extending the frequency range of a voltage digitizer is considerably more difficult.

The prescaler divides both frequency and phase by a fixed factor, four in this case, so both the carrier and the deviation results must be multiplied by four. For modulo 2π phase modulation, such as BPSK, the data should be treated as 8PSK for analysis. Then all phases are multiplied by four before the modulo 2π operation. For example, 8PSK phases of 1, 91, 181, and 271 degrees will all be corrected to the same 4-degree phase of the BPSK prescaler input signal.

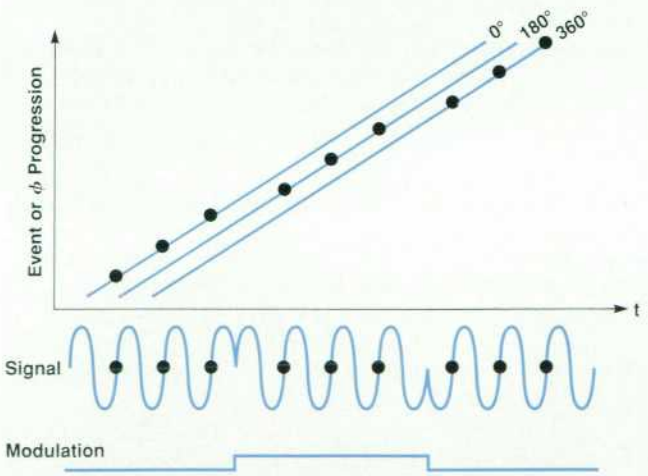


Fig. 8. *Phase progression plot for a BPSK signal.*

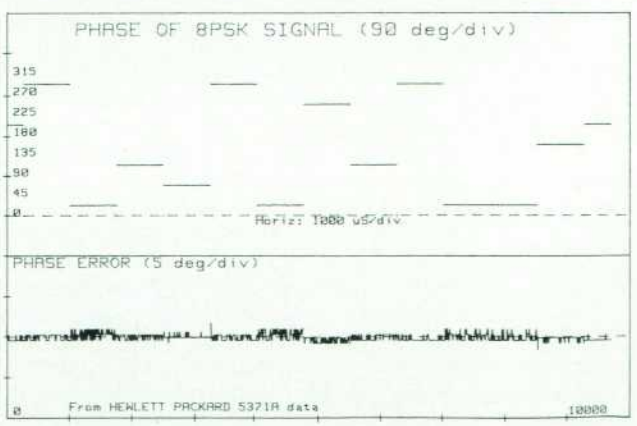


Fig. 9. *Phase and phase error of an 8PSK signal.*

Of course, prescaling and down-conversion can be combined to give a wide frequency agility range in the microwave region.

Conclusion

Phase digitizing is an effective method of capturing and analyzing many spread-spectrum signals. Compared with voltage digitizing, it offers great economy and precision and a wide frequency dynamic range in signal capture. Analyses of frequency-coded and phase-coded signals are vastly simplified because the data is in the form of phase and time. The agility range is extended relatively easily by prescaling. Microwave signals can be measured by down-conversion using a stable but arbitrary local oscillator. However, phase digitizing ignores amplitude and is not suitable for amplitude modulated signals.

Acknowledgments

Mike Ward worked with me in demonstrating the first operating continuous counters, and he later designed the first ASIC version with Dave DiPietro and Grady Hamlett. The contribution of John Flowers to the analysis software was indispensable. Sherry Read suggested the quadratic curve-fit method described in this paper. The support of management to develop such a nontraditional instrument took courage. Lastly, I would like to salute the entire HP 5371A development team for ably transforming a caterpillar into the beautiful butterfly the HP 5371A is.

References

1. B.J. Frohring, B.E. Peetz, M.A. Unkrich, and S.C. Bird, "Fixed-Frequency Sine-Wave Curve Fit," *Hewlett-Packard Journal*, Vol. 39, no. 1, February 1988, p. 48.
2. D.C. Nichols, "Precision Digitizing Oscilloscope Waveform, Analysis, Display, and Input/Output," *ibid*, p. 62.
3. J.P. Costas, "Synchronous Communications," *Proceedings of the IRE*, Vol. 44, no. 12, 1956, pp. 1713-1718.

Frequency and Time Interval Analyzer Measurement Hardware

Examples of measurements on a frequency agile radio are used to illustrate the design and operation of the measurement hardware of the HP 5371A analyzer.

by Paul S. Stephenson

FREQUENCY AGILE SYSTEMS pose many new measurement challenges. For example, how do you characterize modulation on a pseudorandom or unknown carrier? How do you capture relatively long time histories of frequency agile sources without using megabytes of memory? The design of the measurement hardware in the HP 5371A Frequency and Time Interval Analyzer had to deal with these challenges. This article will describe the design and operation of the measurement hardware, using a frequency-hopping radio measurement example.

Frequency Agile Measurements

The principal problem in frequency agile measurements is acquiring a time history of the source frequency. It is more efficient to count the number of signal periods during each sampling period while sampling at a rate tailored to the modulation bandwidth of the source under test than to record the entire waveform and postprocess it. The HP 5371A uses the former method to make frequency agile measurements, and thereby reduces the amount of data and processing required.

Other attempts to analyze frequency agile generators with standard instrumentation such as counters, spectrum analyzers, and modulation analyzers do not effectively handle frequency hopping signals in a dynamic environment. This stems from the constraints that standard instrument techniques impose on the measurement. These techniques require the hopping pattern to be repeated, slowed down, or known, thus limiting a user's understanding of the device's performance with respect to important parameters (e.g., switching transients, settling time, hopping distribution, and frequency or phase modulation riding on a hopping carrier). For more information on characterizing frequency agile sources, see reference 1.

Fig. 1 shows a schematic representation and typical functional characteristics of a hopping radio carrier used in spread-spectrum communications. The 1.6-ms blanked intervals between hops do not contain information and result in wasted memory when recorded with a waveform recorder or a digitizing oscilloscope.

To see the memory requirement differences between a waveform digitization method and the continuous count method of the HP 5371A, consider the three-hop waveform of Fig 1. If f_1, f_2, and f_3 are 48.5 MHz, 87.5 MHz, and 60

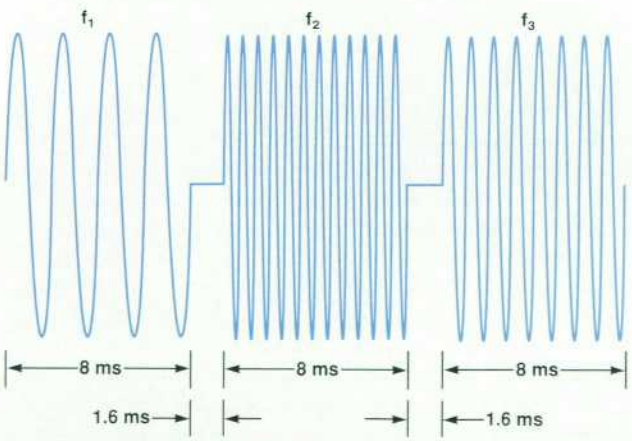
MHz, respectively, an eight-bit digitizer has to sample at twice the highest carrier frequency, or 175 MHz, for 27.2 ms to recover the modulated information. This results in 4.76 million samples at one byte per sample. The HP 5371A only has to sample at twice the modulation bandwidth (2×20 kbits/s) to recover the desired frequency-versus-time information. At this 40-kHz rate the HP 5371A uses 960 samples at 10 bytes per sample for a total of 9600 bytes, versus 4.76 Mbytes for the waveform digitizer, or approximately 500 times less memory to obtain the desired frequency-versus-time data.

Before the introduction of the HP 5371A, dynamic testing of frequency hopping radios was usually limited to transmitters and receivers connected in a back-to-back configuration. Such a configuration uses a "golden receiver" to verify transmitter performance, and vice versa. A golden receiver is a unit that has been proven empirically to meet functional specifications.

This transmitter/receiver test approach has several limitations. Verifying and calibrating the performance of golden receivers is difficult. Often only a few units are available, making field repair difficult. Golden receivers can be poor analyzers for identifying problems within subsystem modules. Finally, information on performance margins cannot be gathered with this kind of go/no-go testing.

Measurement Hardware

The HP 5371A measurement hardware can be thought of as all the hardware used to acquire raw data from the binary outputs of the input amplifier. For example, when measuring a frequency-hopping radio, the RF output is preconditioned by the input board, which outputs a binary signal to the measurement hardware. The binary signal edges occur at the points where the transmitter input has crossed through the trigger level; these points are called events. The HP 5371A counts these events for some period of time, and by taking the ratio of events to time, computes the average frequency over this sampling period.



Frequency Range: 30 to 87.975 MHz
Channel Spacing: 25 kHz
Hopping Rate: 9.6 ms per hop
Settling Time: 1.6 ms
Modulation Modes: FM voice or 20 kbits/s data
Modulation Deviation: ±7 kHz (FM voice and FSK data)

Fig. 1. *Output waveform and typical functional characteristics of a frequency-hopping radio.*

To avoid uncertainty in the number of events counted during the sampling time, a reciprocal counting technique is used to reduce measurement error.[2] The reciprocal technique measures the time interval between the first and last events in the sample period while simultaneously counting the number of events. Therefore, unlike constant-sampling-rate systems, the sampling period is defined by synchronizing the sampling clock with the measured signal. The nth sampling period is the period between the first event after the nth sample-arm signal and the first event after the $(n+1)$th sample-arm signal. This forces an integral number of events to occur during each sampling period, and thus makes the quantization error independent of input signal frequency. Second, it allows control, if desired, of the sampling period in real time by externally gating the instrument with a modulated sampling clock.

In this continuous count system, measurements are made in blocks, and within a block, the sampling periods are contiguous. Thus the last sample of the nth sampling period is the first sample of the $(n+1)$th sampling period.

In summary, the primary functions of the measurement hardware are counting events and measuring the time between the first and last events during a user-defined sampling period. This information is stored for processing after the completion of the data acquisition process.

System Block Diagram

Fig. 2 shows the HP 5371A system block diagram. As noted earlier, the input board preconditions the input signals and converts them to binary signal inputs for the measurement hardware. Other measurement hardware inputs include an external arming input (100 MHz) and an external frequency standard input (1, 2, 5, or 10 MHz). In addition to these signal inputs, the measurement hardware receives measurement setup and control instructions from the microprocessor.

Signal outputs from the measurement hardware include gates 1 and 2, arm delays 1 and 2, and a 10-MHz frequency standard output. The gate and arm signals are binary signals indicating when measurement samples occur and when certain arming conditions are satisfied. These signals make it possible to use the flexible arming and gating to trigger additional instrumentation. The frequency standard output

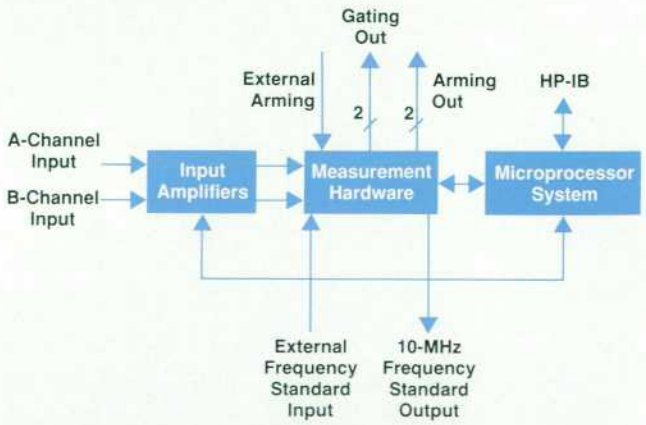


Fig. 2. *System block diagram of the HP 5371A Frequency and Time Interval Analyzer.*

is a buffered output of the instrument's 10-MHz ovenized crystal frequency standard. When an external frequency standard is applied, this output matches the external standard's stability characteristics at 10 MHz.

In addition to these signal outputs, the measurement hardware provides information to the microprocessor, including the state of the hardware and raw measurement data. Raw, unprocessed, binary data can also be rapidly acquired and transferred via the HP-IB (IEEE 488/IEC 625) to a host computer for subsequent processing.

Measurement Hardware Block Diagram

Fig. 3 shows the measurement hardware block diagram. The four major blocks are the sequencer, the event counters, the time base, and the memory system. The sequencer routes the input signals to the appropriate counting chain and generates the arming and latching signals for the event counters and the time base. The arming and latching signals depend on the input signal configuration, the sampling interval, and other arming qualifiers.

The two event counters count the events sent to them from the sequencer. They can be used for measuring and arming (e.g., event or time holdoff arming). The time base consists of two major subblocks: a counter chain that counts a synthesized, 500-MHz clock phase-locked to the internal or external frequency standard, and an interpolator which increases time resolution by a factor of ten over the 500-MHz time counter chain. The memory system consists of an 8K × 112-bit RAM operating at 10 MHz.

Measurement Example

We can see how each of these blocks functions during a measurement by using the example of a frequency-hopping radio. Suppose we have available from the radio under test both the RF output and the SYNC pulse defining the beginning of a psuedorandom hopping sequence. Assume that we want to examine, with one thousand samples, a 10-ms portion of the radio's hopping sequence, two seconds into the sequence. We must precisely delay the measurement for two seconds after the SYNC pulse. This can be accomplished by setting the HP 5371A to measure as follows:

```
FREQUENCY Measurement Channel A
Acquire 1 block of 1000 measurements

TIME/INTERVAL Arming Mode
Block Holdoff:
  After POSITIVE edge of EXT ARM
  Delay 2.000000000 s
  Then arm a block of measurements

Sample Arm 10 μs
Acquisition Time/Block 10 ms
```

In general, this setup allows us to delay to any point after the external arming signal with 2-ns time resolution (8-second range) and then take up to 1000 continuous frequency measurements with sample periods as short as 600 ns. In this example, we are delaying two seconds and the sampling period is 10 μs. The measurement is carried out by programming the sequencer to route the radio's RF output (connected to Channel A of the HP 5371A) to one of the event counters and the 500-MHz time base clock to the other event counter. The time base clock to the second counter is gated on after receipt of the radio's SYNC signal by the sequencer. Finally, the 10-μs sample-arm signal is selected as the qualifier for generating latching signals. The latching signals capture the event and time counts.

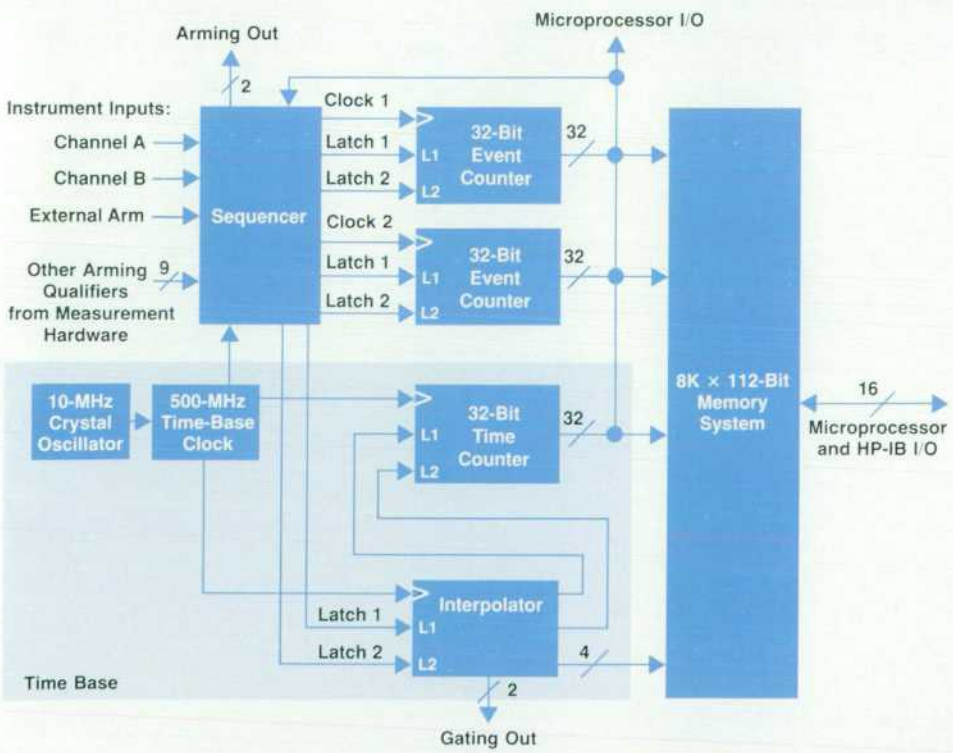


Fig. 3. *Simplified block diagram of the HP 5371A measurement hardware.*

The event counters perform two functions: one counts the number of events on channel A (the radio's RF output), and the other counts a predetermined number of cycles of the 500-MHz time base clock (this generates the delay after the radio's SYNC signal). The event counter is preset to its terminal count minus the desired time delay divided by 2 ns. Then, after the sequencer enables the 500-MHz clock to this counter, the terminal count signal defines the selected delay arming.

In short, the measurement sequence is as follows. The SYNC signal in the external arming channel goes to the sequencer, enabling the 500-MHz time base to clock the event counter. The terminal count signal from this counter defines the block holdoff time, signaling the sequencer to start the 10-μs sample-arm signal. Next, the sequencer synchronizes the sample-arm signal with the channel A events, creating latch signals for the RF input event counter and the time base. Finally, the latched values of the event counter and time base are stored in the memory system as raw data. After acquiring the data for all 1000 measurements, the processor assumes control and processes the raw data into measurement results. Fig. 4 shows the HP 5371A display of frequency versus time for this measurement.

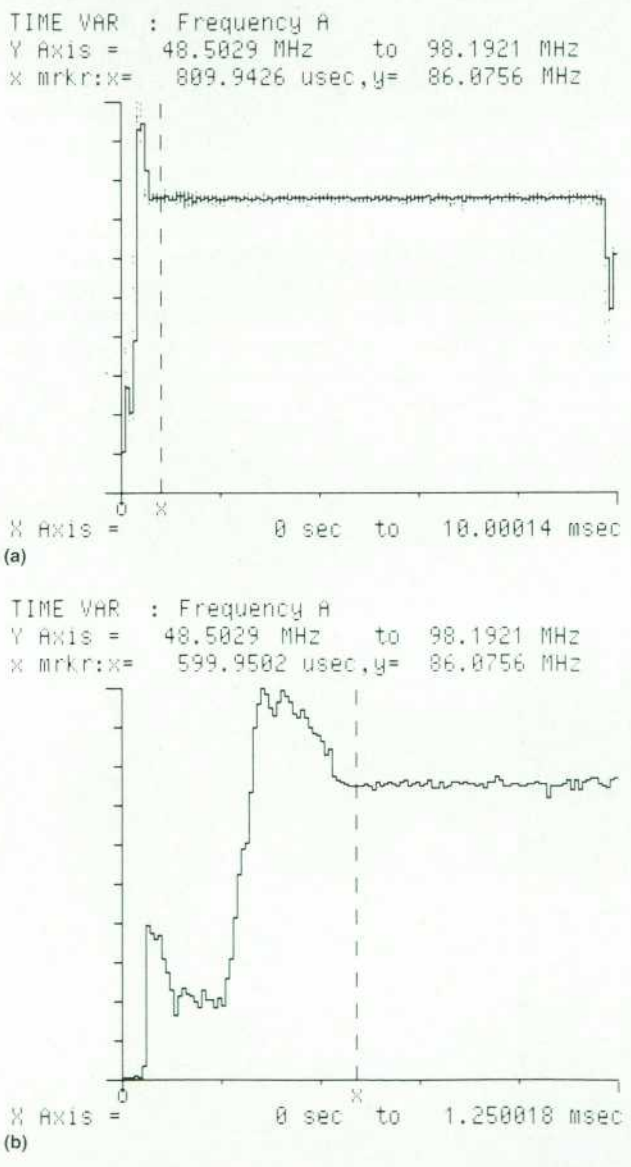


Fig. 4. *Complete (a) and zoomed-in (b) views of a frequency hopping radio's switching profile. The radio can hop from 48.500 MHz to 86.075 MHz in approximately 600 μs.*

This example outlines the basic hardware operation for the single-channel frequency measurement with time interval arming. For more information on other HP 5371A measurement configurations, see reference 3. The next section will cover the theory and operation of the four major subblocks of the measurement hardware in more detail.

Sequencer

The sequencer is one of two ASICs (application-specific integrated circuits) developed in-house for the HP 5371A. Fabricated using the HP5 process (a 5-GHz f_T, bipolar process), this digital IC operates at frequencies in excess of 500 MHz. Containing over 2000 transistors and dissipating 3W, the sequencer occupies a 4.56×4.74-mm die packaged in a 72-pin printed circuit pin-grid array, or PCPGA (see Fig. 5). Mounting the die on the PCPGA's gold-plated, copper heat slug and attaching a three-finned machined aluminum heat sink allows the sequencer to operate with only a 10°C/W junction temperature rise above ambient at the air intake. This low thermal resistance keeps the junction temperature below 90°C under worst-case operation and contributes to the reliability of the sequencer.

As illustrated in the preceding measurement example, the sequencer serves as the signal-routing hub of the multiple-chip continous count system, and performs the signal switching, arming, and gating for the two input channels. 55 control bits in the sequencer configure it for the selected measurement and provide two-stage sequential arming for

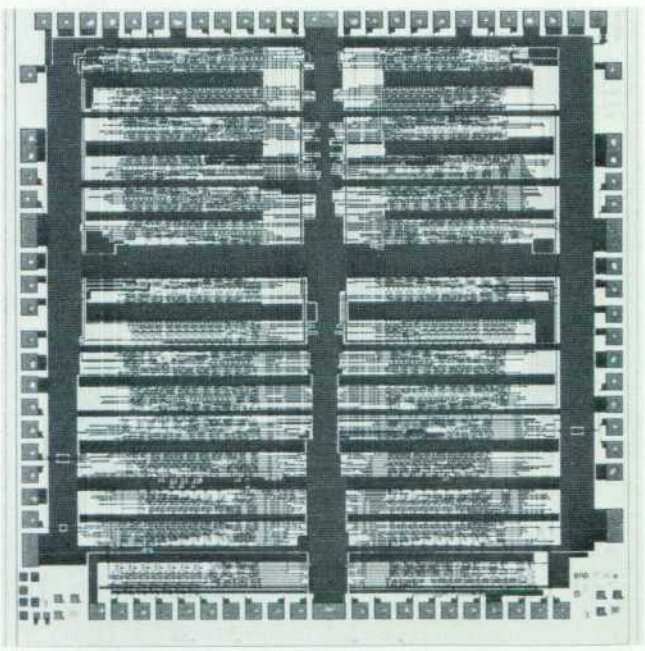


Fig. 5. *The sequencer is one of two application-specific ICs developed for the HP 5371A. It controls complex arming configurations for input signals to 500 MHz.*

both stop and start arming. For two-channel time interval measurements, a parity detector keeps track of events on both channels to measure edge-pair relationships consistently. This feature eliminates bimodal distributions.*

The sequencer architecture consists of two nearly identical channels, each containing an arming section and a clock section as shown in Fig. 6. The clock section selects the signal to be counted and routes it to the event counter. The arming section selects the sources of the latch signals for the event counters and the time base after the arming requirements for the measurement are satisfied. Designed to minimize crosstalk and jitter effects, the sequencer contributes less than 25 ps rms jitter to any measurement.

Event Counters

The HP 5371A has two 32-bit event counters. Each receives clocks and latches from the sequencer, and outputs latched count values on the fly, without resetting the counter. Each 32-bit counter consists of two 16-bit, cascadable, zero-dead-time (ZDT) counters. The ZDT counter chip is the other ASIC developed for the HP 5371A. Operating in excess of 500 MHz and, like the sequencer, fabricated in the HP5 process, the ZDT chip contains over 1500 transistors and dissipates 1.5W. It occupies a 4.56 × 3.43-mm die packaged in a 72-pin PCPGA. Fig. 7 shows the printed circuit board assembly containing the ASICs (six ZDT chips and one sequencer).

*In counter/timers without this feature, arming ambiguities cause ±(time interval) measurements under some conditions to give the desired time interval, and under other conditions to give the signal period minus the desired time interval, thereby producing bimodal measurement distributions.

As shown in the measurement example, the ZDT chip counts events and outputs latched data to the measurement memory system. The ZDT chip also outputs the terminal count signal, which the sequencer can use as an arming qualifier. The ZDT chip accepts 16 bits of preset data and a 13-bit control word that sets the binary scaling ratio and the ZDT counter operating mode. Three address lines plus a chip-select signal control I/O on the multiplexed, bidirectional, 16-bit ZDT bus.

The ZDT latches count data on the fly at input frequencies in excess of 500 MHz. It achieves this with a cascadable architecture that synchronizes the latch command with the clock and then maintains this relationship through all 16 bits (see Fig. 8). Latching rates are limited by the clock-to-data-valid time of the counter (>10 MHz with two cascaded ZDT chips forming a 32-bit counter). Two fully parallel latch paths in the ZDT allow two latches to be taken back-to-back over the full operating bandwidth of the ZDT chip (>500 MHz). This overcomes some of the latch-rate limitations imposed by the clock-to-data-valid time of the counter. In addition, time interval measurements through zero time can be made using only one ZDT chain, because the parallel latch paths handle two totally independent latches.

Time Base

The purpose of the time base is to time-tag every latched event count, thereby building a time history of the latched event counts. From this, frequency or time interval can be analyzed as a function of time. As shown in Fig. 3, the time base consists of the time counter, which counts a

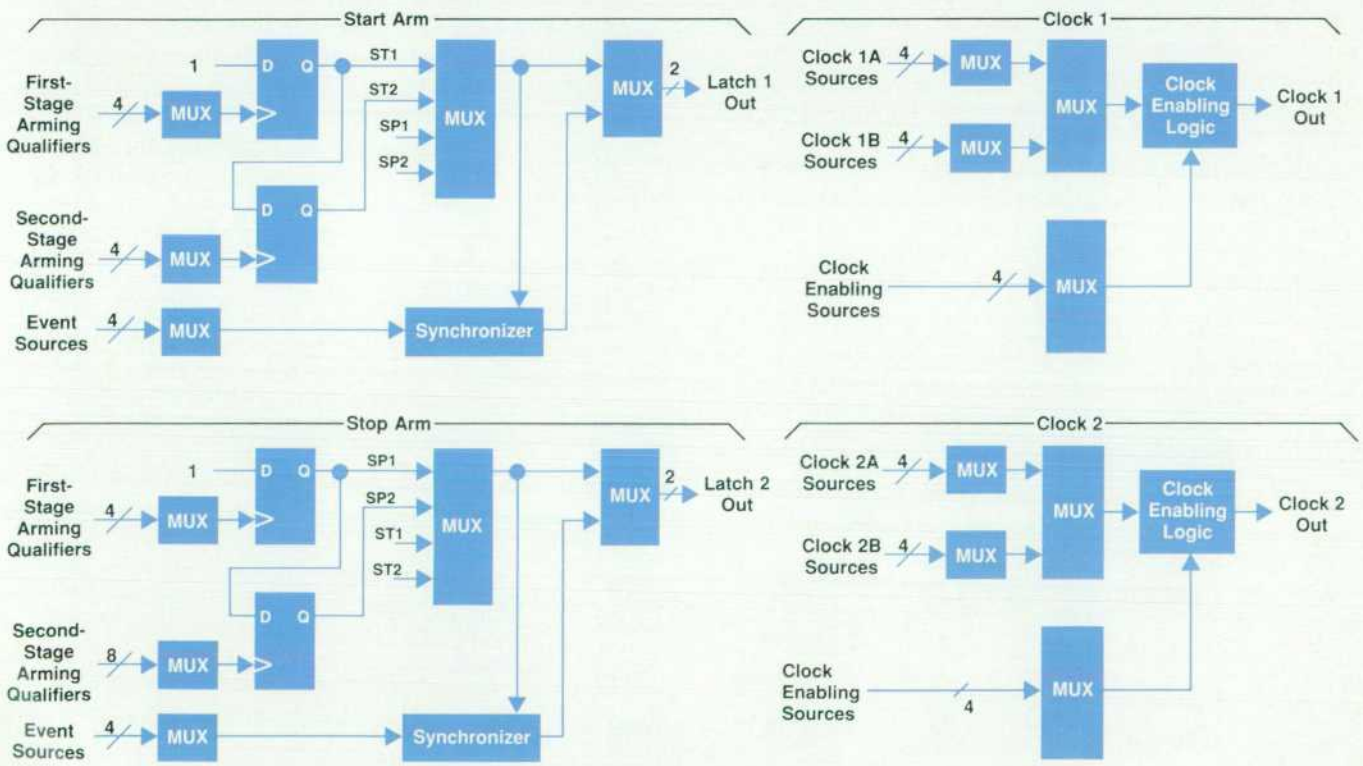


Fig. 6. *Simplified sequencer block diagram. The sources and qualifiers shown are the instrument inputs (channel A, channel B, external arm) and other arming qualifiers from the measurement hardware.*

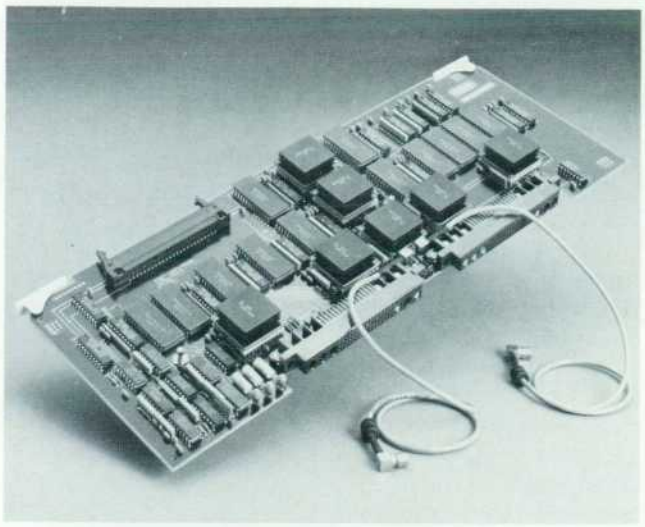


Fig. 7. *Printed circuit board assembly containing one sequencer and six zero-dead-time (ZDT) counter ICs.*

synthesized 500-MHz clock phase-locked to the frequency standard, and the interpolator. Two ZDT chips form the 32-bit time counter, which operates like an event counter dedicated to counting the 500-MHz time base clock.

Latches to the time counter come from the sequencer much like latches to the event counter. They differ, however, from latches to the event counter because they must first be synchronized with the 500-MHz time base clock. This sychronization occurs before interpolation. Fig. 9 illustrates this measurement timing relationship.

The interpolator assembly performs two functions: it synchronizes the event latch signal with the 500-MHz time base clock, transforming it into a time latch, and it quantizes the time interval between the event latch and the time latch. Each time-tag consists of the time counter result, with 2-ns resolution, and the interpolator result, which quantizes the time difference between the time latch and the event latch with 200-ps resolution. These results are combined during measurement processing, giving 200-ps resolution for each time-tagged event count.

Interpolation Technique

The interpolation technique is based on a controlled race condition between the event and time latches. As shown in Fig. 10, the two latch signals go to a bank of nine high-speed D flip-flops which determine the relative phase between the two signals at nine equally spaced stages in the race. The race spans the 2-ns period of the time base clock.

With the race fixed such that the event latch signal is progressively delayed by 200 ps per stage (one time quantum per stage), the flip-flop outputs will quantize the original time interval between the two raced latches. Logical one and zero outputs indicate event latch leading and time latch leading, respectively. The stage in the race where the relative phase changes from event latch leading to time latch leading determines the original time interval between the latches within 200 ps, and thereby quantizes it. The flip-flops output a thermometer code, which is converted into a binary code and stored with the event count and time count in the memory system. The advantages of this interpolation technique over other time-to-digital converters are higher conversion rates and elimination of time-to-amplitude conversion circuits.

Memory System

The HP 5371A can measure signals as long as 8 s with resolution to 200 ps, and can count up to four billion events on both channels with 1-event resolution. These large dynamic ranges, 4×10^{10} and 4×10^{9}, place heavy demands on the measurement memory width because the HP 5371A always stores full-precision raw data. Combining these raw data memory requirements with the measurement status-bit memory requirements results in a measurement memory width of 112 bits. The memory system also supports the 10-MHz maximum sampling rate, so the instrument has an information bandwidth potential in excess of 1 GHz.

The processed measurement memory depth depends on the mode of operation. The raw measurement memory depth is 8K samples. However, because of processing limitations, only 1000 individual processed measurements per block are available from the front panel (larger measurement sample sizes can be accumulated for statistical analysis by acquiring multiple blocks). Two-channel frequency, period, and totalize measurements require twice

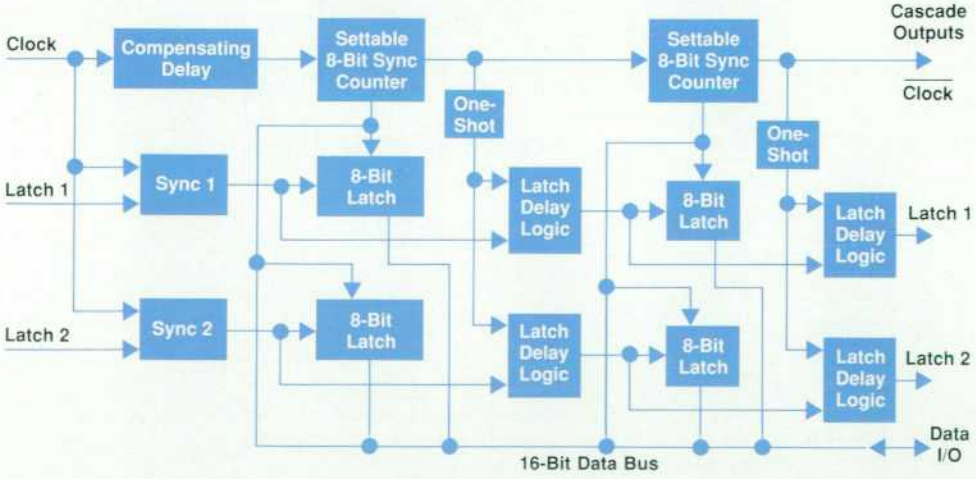


Fig. 8. *Simplified block diagram of the ZDT counter chip.*

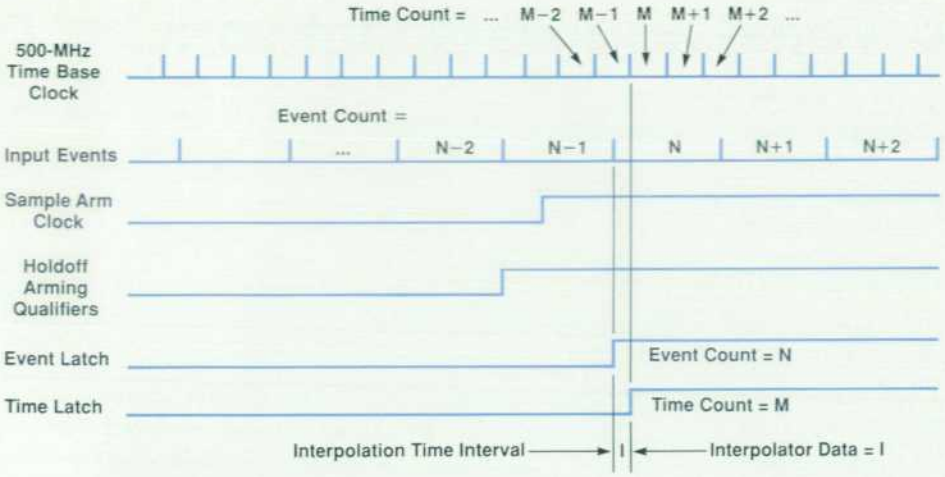


Fig. 9. *Timing diagram for event and time data acquisition and interpolation.*

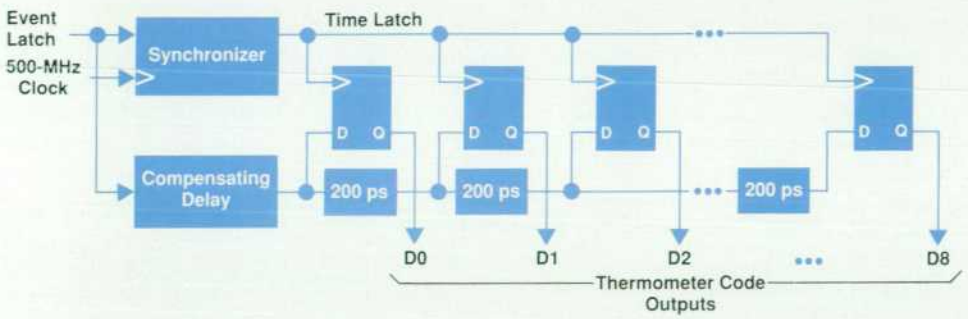


Fig. 10. *Simplified interpolator block diagram.*

the raw data of the single-channel measurement, so only 500 measurements per block are available.

A host computer, reading raw binary data over the HP-IB, overcomes the processing limitations and can access the full 8K measurement memory depth. This nets 4095 individual processed measurements per block for single-channel measurements (some measurements require two pieces of raw data per processed measurement). For two-channel measurements the net is 2047 measurements per block.

Acknowlegments

Many people contributed to the design of the HP 5371A measurement hardware. William Lam designed the ZDT/sequencer board and was responsible for the ZDT ASIC development. Mark Wine and Victor Prince handled the DMA memory support and measurement control hardware. Jim Cole designed the time base synthesizer. Steve Carroll and Ron Jensen contributed to the interpolator hardware and the power supply design, respectively. The project team would also like to thank everyone at the Santa Clara Technology Center who contributed to the development and release of the sequencer and ZDT ASICs.

References

1. Hewlett-Packard Application Note 358-1, *Characterization of Frequency-Agile Signal Sources*, November 1987.
2. Hewlett-Packard Application Note 200, *Fundamentals of Electronic Counters*, July 1978.
3. *HP Product Note/Specification Guide, HP 5371A Frequency and Time Interval Analyzer*, November 1987.

An Integrated Voice and Data Network Based on Virtual Circuits

Developed as an HP Laboratories research project, this network offers true integration of voice and data, a single architecture for local and wide area networks, high throughput, low host overhead, very good cost/performance ratio, and effective interfacing to existing standards.

by Robert Coackley and Howard L. Steadman

THE RAPID ACCEPTANCE of distributed data processing, workstations, and personal computers has made networking the critical unifying technology in modern computing environments. Most of today's networking technology is an extension of ideas developed in the late 60s and early 70s. However, within the last 15 years there has been significant change both in users' requirements and in the technology available to implement networks. This architecture was designed to address these changes and to deal with what we consider to be the main issues of an integrated voice and data network. The architecture was developed as a research project of HP Laboratories. Work on it has now ceased, and this paper is a summary of the knowledge gained. One product resulted from the project: HP StarLAN 10, a 10-Mbit/s local area network using twisted-pair wire.

The first design issue for an integrated voice and data network is the true integration of voice and data in all aspects of the network. While it may seem redundant to raise this as an issue, there is an important point to be made. Many systems and products provide both voice and data services, but are not integrated in their design. For example, many manufacturers' plans for ISDN envision functionally separate switches for voice and data. If the different services are provided by separate equipment, there is no opportunity to take advantage of the increased efficiency that the complementary nature of digital voice and digital data allows. The results are increased cost and lower performance.

The second issue is that of area coverage. Users need networks that cover not only local areas, but campus, metropolitan, national, and international areas as well. This is particularly important to a company such as HP, since many of our customers are geographically distributed, and often even a single location will exceed the limits of the current LAN technology.

The third issue is performance, particularly in the intermediate level protocols. Many of these protocols were developed in the late 60s and early 70s for wide area networks (WANs). Both the limitations of the transmission technology and the implementation techniques employed at that time dictated software intensive implementations of relatively complicated algorithms. While appropriate to those circumstances, these protocols are not readily adaptable to high speeds and are not well-matched to the characteristics of modern links such as T1 or satellite links. The result is that excessive demands are placed on the host, both in CPU cycles and memory required. Even when these demands are met, the network performance is often several orders of magnitude lower than required.

The fourth issue is cost. As networks move out of the laboratory and engineering departments and become pervasive within a company, cost becomes a major concern. This concern is driven both by the greatly increased number of connections and by the extension of these facilities into areas where capital investment is traditionally much lower.

In addition to these four issues, it is appropriate to understand the role of networking standards. While recognizing the importance of national and international standards, we determined that the objectives of the architecture could not be realized within the constraints of current standards. The approach taken was to develop internal standards to meet the design objectives while providing external interfaces compatible with international standards. These external standard interfaces contribute no performance degradation other than what is inherent in the external standard.

This is exactly the strategy of systems such as packet transport networks that use X.25 as the external interface. They give complete freedom to the implementor in the design of the internet. Thus a network with X.25 and X.75 interfaces is in every sense a standard network without any regard to the implementation of the internet.

The architecture outlined in this article addresses all of these issues and offers a connection-oriented network with the advantages of virtual circuit switching, together with error control and link failure recovery. The main contributions are:

- True integration of voice and data
- Single architecture for LANs, MANs, and WANs
- High throughput with low host overhead
- Very good cost/performance ratio
- Effective interfacing to existing standards.

Virtual Circuits

A *virtual circuit* is a communications circuit that appears to exist. There is no such physical circuit, but by careful arrangement of the flow of data it is possible to simulate the connection between any two endpoints. Fig. 1 shows the concept of virtual circuits in a simple network. Generally, communication is achieved by forming the data into packets of specified length including addressing information (Figs. 2 and 3). Data is routed on a by-packet basis to the appropriate nodes, but the main links in the network will be carrying many interleaved packets for a variety of destinations. There are no fixed end-to-end connections, yet the user appears to have a communications link. The important point to remember, in reference to this type of network, is that the addressing is specific to each packet and can therefore represent a substantial overhead that restricts the rate of flow of data. Additionally, it is possible for packets to arrive at nodes out of sequence. This doesn't particularly matter for data communication, but would be important for the communication of real-time services such as speech. In general, virtual circuits are a universal transport mechanism that consume transmission and switching resources only when required. This dynamic allocation of resources can benefit the user and the network substantially through either enhanced throughput or reduction of the network loading.

There are different ways of implementing the concept of virtual circuits. The method used in this network is to pass address information only at the start of communication and thus improve the efficiency of the data link.

Network Architecture

The network architecture developed in this research project combines the most desirable features of packet and circuit switching with new technology that makes implementation of the network practical. At the higher levels, the architecture provides a virtual circuit network, with its inherent simplicity for host computers and its transparency as viewed by the user. At the lowest level, it incorporates data framing on the individual transmission links to provide reliable transmission over error-prone links and to maintain multiplex synchronism between ends of the link. At the intermediate levels, a hardware-based architecture has been developed that overcomes the efficiency and performance problems inherent in a switching node implementation based on general-purpose processors. Central to this architecture is a special-purpose hardware structure that enables the system to perform the switching function with very good cost/performance ratio and with high absolute performance. From packet networks, the system design takes highly efficient utilization of link bandwidth and reliable data transport, and from circuit networks, good matching to user requirements, namely, a transparent bidirectional channel with small, bounded transmission delays. The architecture also places most of the communications-oriented processing within the network. Thus the host is relieved of the complexity of implementing those functions and of the requirements for the code space and computing power to execute them.

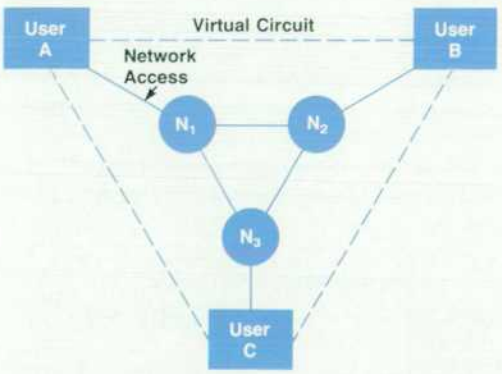


Fig 1. *Virtual circuits exist between A and B, B and C, and A and C, even though there are no direct links allocated to individual users.*

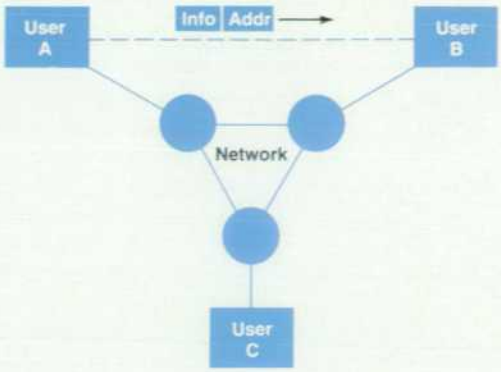


Fig 2. *Information communicated between users is enclosed in packets for transport through the network.*

Since the network is truly integrated for all aspects of transmission and switching, voice services are not handled as a special case and will therefore only be referred to occasionally. In particular, voice traffic is handled as a real-time circuit to provide the bounded delay required for acceptable performance, all other voice specific requirements being handled at the endpoints. In the simplest implementation, the endpoint processing consists only of analog-to-digital conversions and timing. It is interesting to note that advanced voice transmission techniques such as TASI (pause compression) and variable-rate encoding, can be implemented solely by changes to the terminal equipment and place no special demands upon the network.

The network provides, then, direct virtual circuit service to its users, with data transmitted transparently over each virtual circuit. Route determination and circuit routing are distributed and adaptive, and there are robust error control

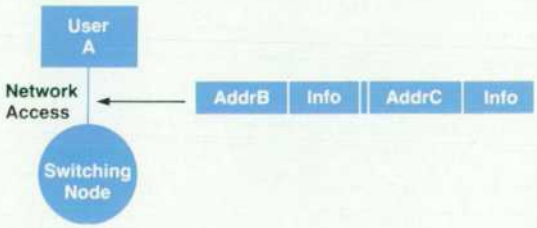


Fig. 3. *The access link to the network may be submultiplexed to give multiple virtual circuits per user.*

and data ordering schemes as part of a node-to-node protocol. The design also accommodates either link or switching node failures without introducing errors in the virtual circuits routed through the failed components.

Network Structure

The network structure is very simple, since it consists of identical switching nodes connected by data links in a network of arbitrary topology. All nodes provide circuit routing, network management, flow control, and error recovery functions as well as data transport. Additionally, nodes that connect to user equipment provide some level of command interpretation so that circuit connection and disconnection and control of services can be invoked.

Communication Links

Communication links within the network are either *access links* connecting user equipment to switching nodes, or *trunk links* used to interconnect nodes (Fig. 4). In this design the protocols used on trunk links are optimized for the operation of the network and are invisible to the user. Access links, however, may use standard protocols for the connection of commonly used equipment to the network or may have proprietary protocols where superior performance is required. Access links may also be used to interconnect with other networks such as those using X.25 protocols.

Since access links are one of the most cost-sensitive elements of the design, we have developed a proprietary link that optimizes the compromise between cost and performance. This link accommodates voice and data traffic and employs a digital transmission scheme to provide full-duplex transmission over single-pair, twisted wiring, as used in telephone systems. The technique is conceptually simple in that each direction of transmission is separated in time (Fig. 5), a scheme sometimes referred to as ping-pong or time compression multiplex (TCM). In this design the line rate used is 10 Mbits per second with Manchester encoding for transmission of the clock component. Data frames in each direction are of variable length up to a maximum of 256 bytes. The protocol is bit-oriented and has several features in common with the High-Level Data Link protocol (HDLC), including bit stuffing for data transparency, unique frame delimiters, and a sixteen-bit cyclic redundancy check (CRC) for error detection. The total overhead for each frame is five bytes, which yields a utilization efficiency in the range of 17% to 98% depending upon the length of the frame.

Network operation does not depend upon the nature of the trunk or access links, so a wide range of interconnect technologies such as optical fiber, microwave, satellite, or multiplexed cable can be accommodated. In particular, simple phone wiring offers many advantages for access links over short distances and would be the most common medium for connecting workstations to the network.

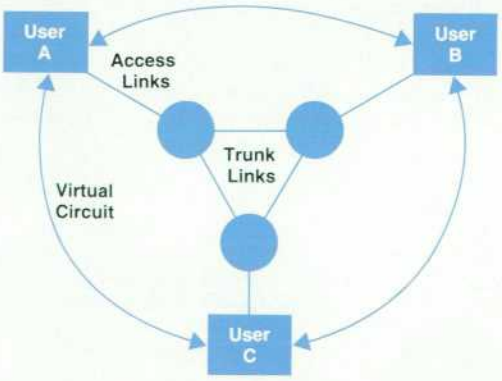


Fig. 4. *General network structure.*

Network Nodes

Node design is based on a hierarchical structure in which fast but simple hardware deals with the most frequently required task of switching data bytes. Lower-rate tasks such as those associated with frames of data on a link are handled by a processor dedicated to that link. Interactions between links are even less frequent and are handled by a processor supervising a number of links. The general physical structure is shown in Fig. 6. Alternative implementations of the architecture distribute the function of the switching engine to each link interface processor, where the storage array is provided by a common shared memory.

Each link processor implements the interface between shared switching memory in the node and a communication channel. Trunk link processors handle serial communications, while an access link processor may interface to a similar line, a parallel interface, or even a number of UARTs for asynchronous terminal connections. The common control processor deals with network addressing at the time each virtual circuit is established. It also manages several other tasks, including network status reporting, accounting, node diagnostics, failure reporting, and maintenance support. As mentioned earlier, there is a special hardware processor designed to optimize network performance. This is the frame engine (see Fig. 7.).

The frame engine consists of a storage array and various processors that together perform the functions of the switching engine of Fig. 6. The storage array contains data waiting to be passed through the node together with details of the linkages between the data elements. Tear/build engines construct or disassemble frames in close cooperation with the queue descriptor processor. The mapper and the service vector processor provide data to the queue descriptor processor to identify whether a virtual circuit has data waiting for transmission and to determine the details of the mapping across the node. Mapping is the linking of virtual circuits to queue descriptions for locating data within the storage array. Switching is achieved within the frame engine by linking different virtual circuits to queues in the storage array.

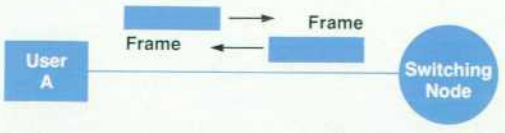


Fig. 5. *Time compression multiplexing (ping-pong). Data frames going in opposite directions are separated in time.*

It is possible to expand the capacity of a node by adding a frame engine interface card, which provides a bus link for additional frame engines to communicate with the appropriate queue descriptor processors and storage arrays.

Data Link Layer Design

The data link layer of the network design is fundamental to the operation of the network and specifies the frame structure and trunk link protocol. Here we describe the design with reference to a simple dedicated trunk link connecting two nodes. Link operation is full-duplex and symmetrical, so we only need to consider the operation of one half of the link.

Each frame is made up of three major sections: a header containing frame identification information, a body containing virtual circuit data and flow control information, and a check and acknowledgment field to verify reliable frame transmission.

The general format within the trunk link frame is shown in Fig. 8. Frame length is link specific, that is, the frame length is fixed for a given link, but can be optimized for particular link types to accommodate error rate or delay specifications. Frames are sent across the link and are acknowledged by the receiver. Each individual frame is acknowledged selectively by the use of sequence numbers. The sequence numbers are assigned cyclically, that is, modulo some number, using a parameter to specify the cyclical range, which is set large enough to eliminate ambiguity. The sequence number is a 16-bit binary number whose maximum value is defined by the window size at the receiver and determines the number of frames that can be sent before an acknowledgment must be received. The acknowledgment fields shown as 1st Ack and 2nd Ack in Fig. 8 contain 16-bit sequence numbers of frames that were correctly received from the sending node. Two acknowledgment fields are provided to allow time separation of multiple acknowledgments, thereby enhancing network reliability.

Error control is accomplished by means of a 32-bit CRC and a 16-bit flag using the generating polynomial specified by AUTODIN and Ethernet. The decision to use a 32-bit CRC and a 16-bit flag was based on our desire for high reliability. It has been shown[1] that significant improvements in link reliability require both the 32-bit CRC and a 16-bit flag. This can be done with no change in throughput for traffic patterns where frames exceed 1500 bits in length. Typical frame lengths for practical voice/data networks meet this criterion. Each frame is protected by the CRC generated by the sender and checked by the receiver, which acknowledges each correctly received frame. Frames not acknowledged within the period defined by the window size are retransmitted by the sender to achieve error control. The design also makes provision for multiple retransmissions to improve the probability of correct reception for retransmitted frames. In the present design, retransmitted frames are sent twice. These are separated by several other frames to reduce the chance that a single error burst would corrupt both attempts. The probability of errors in both of the retransmitted frames is negligibly low. These techniques improve the responsiveness and overall efficiency of link communication.

Trunk link frames are separated by a single flag which consists of a 16-bit pattern defined as leading and trailing zeros with 14 ones in between (0111111111111110). The flag pattern is unique and must not be allowed to occur accidentally within the frame bands. To avoid this, a technique known as bit stuffing is used. The transmitter inserts a zero bit after any sequence of 13 contiguous one bits whenever this pattern occurs in the data. At the receiver, whenever 13 contiguous one bits are received followed by a zero the receiver will discard the zero. If there are 14 contiguous one bits, the pattern will be interpreted as a flag marking the beginning or end of a frame.

The virtual circuit segment consists of data for one or more virtual circuits. The field is of fixed length for a given link and may be up to 512 bytes long. Its internal structure is formatted by the frame engine. To provide maximum transmission efficiency, we compress the virtual circuit number by using increments to identify individual virtual circuits. On many occasions the increment will be expressible in six bits, but if the increment is large, an increment extension of up to 12 bits is available. In processing a frame, the current virtual circuit number is assumed to start at 0, and the first few virtual circuits are used for control information. Fig. 9 shows the outline of the scheme.

Fig. 9a shows three virtual circuits: A-E, B-F, and C-G. These could represent links between hosts, between workstations, or between workstations and hosts. The frame structure would be as shown in Fig. 9b. Now assume that source B is quiescent for a while. The frame would then be as shown in Fig. 9c. Thus the main body of the trunk link frame is composed of segments for each virtual circuit

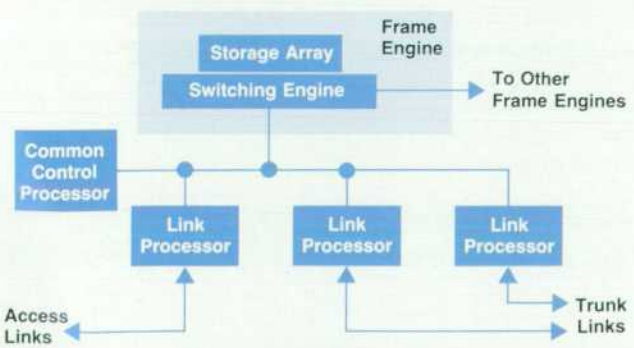


Fig. 6. *Network node structure.*

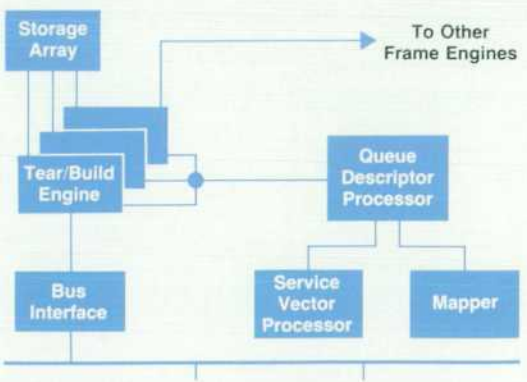


Fig. 7. *Frame engine block diagram.*

with a virtual circuit header as part of each segment. The header carries both the increment value and the length of the segment. The increment value is used to identify the virtual circuit number and is called ΔVC.

The network design provides for multiple classes of service. The present design provides control of delay for certain user services by defining two classes of service. The class of service is changed by means of a delta-class-of-service bit, which is used in association with the virtual circuit increment value. The two classes adopted in the prototype network are:

- Real-time service (includes voice services)
- Bulk data service.

To accommodate all this control, two forms of segment header are employed (see Fig. 10). The first is the normal two-byte header with six bits available for the virtual circuit increment value, and the second is an extended three-byte header with 12 bits for the virtual circuit increment value. Extended headers are identified by setting the leading bit to one, and additional protection from ambiguity with an end-of-frame byte is provided by setting the second bit to zero. The third bit in the extended header is the delta-class-of-service bit. The normal header is most commonly used because the algorithm does keep virtual circuit numbering densely packed, but the extended header must be used for a class-of-service change. Extended headers can be concatenated to provide for very large increment values. The need for such large increments is very rare.

Virtual circuit segment headers also contain three flow control bits (fc), which will be referred to later.

Flow and Congestion Control

Flow control is a mechanism whereby a slow receiver can hold off a fast sender. Congestion control deals with the problem of more traffic at some point in the subnet than there are resources available. In this network design the two controls are provided by a single mechanism, which obeys two guiding principles:

- Network resources should only be overallocated in a manner consistent with the planned grade of service.
- If service has been guaranteed, then that guarantee should be honored. If no guarantee has been given, then equal treatment should be given to all members of a given service class.

The flow/congestion control protocol chosen is based on the concept of *backpressure*. The basic principle is that only those virtual circuits feeding into a congested area are affected. Control then propagates to all the sources contributing to the congestion and reduces (or stops) their inputs, leaving other traffic sources undisturbed. The network applies selective flow control to each individual virtual circuit and operates on positive transmission permits. The issuance of transmission permits is based on the concept of *units of permission*, which are guarantees that buffer space is available in a particular node. A transmission permit indicates the willingness of a receiver to accept a quantum of bytes from the sender. These techniques provide compression of control information throughout the network and contribute to the adaptive nature of the architecture. In the prototype implementation we used 32 bytes as the unit of permission. All information is error-protected and the transmission permits are communicated within virtual circuit segments to gain the error protection afforded by the trunk link error control, thereby maintaining network reliability.

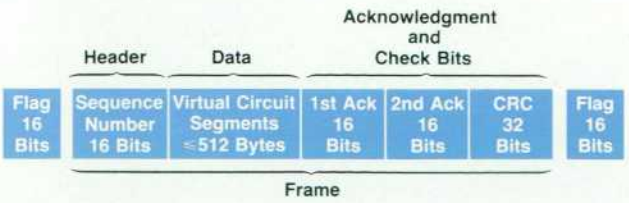


Fig. 8. *Trunk link frame format.*

As shown in Fig. 10, there are three flow control bits. Bit fc2 is the permission bit. Bits fc0 and fc1 are used as virtual circuit acknowledgments to provide for two-hop acknowledgment, allowing the network to deal with link or node failures.

Nodes implement the two-hop acknowledgment scheme by passing back an acknowledgment periodically on the return channel to the second-previous node (if there is one) along a virtual circuit, saying how many characters were received on that circuit since the last acknowledgment. When this acknowledgment is received, a node is free to discard the acknowledged characters in that virtual circuit buffer, thus making room for more data. All data is saved until an acknowledgment is received, so that when the intermediate node notices a communication failure and forces an attempt to reroute the virtual circuit, no data that was within the failed node is lost.

Tokens are employed to mark the start and end of a data stream that is being returned to the access node as a result of a communication failure. These tokens enable nodes to match data streams so that if there is an alternate route, then a contiguous stream will arrive at the destination node even when there is a link or node failure.

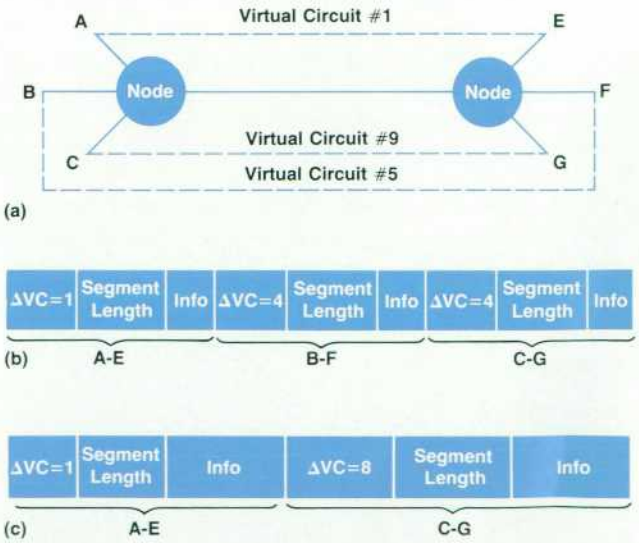


Fig. 9. *(a) Three virtual circuits. (b) Frame structure. (c) Frame structure after B is quiescent for a while.*

Network Routing

In complex networks, particularly in nonlocal ones where the cost of transmission is significant, the assignment of circuits to optimum or near optimum routes has a significant economic benefit. However, optimization is one of the more complex problems that any network has to deal with. It is further complicated by the dynamic nature of the effective topology of the network and the available resources. This occurs because of the variations in loading on network resources and the failure or impairment of links or nodes.

Evaluation of route costs is a linear-programming problem, which for small networks is a fairly simple task. However, as the size of the network grows then route evaluation can become a significant problem.

Some networks solve the problem by using a central supervisory machine, which interrogates the network to assess its topology and then computes the best routes. Others use manual assessment of alternatives followed by selection of a route based on the load and class of service. Our network design uses a decentralized adaptive approach, in which no single node or resource has total knowledge of the topology and loads. The routing algorithm adapts to topological changes and adjusts routings automatically as loadings vary. The network also accepts new nodes without manual intervention and recovers from link or node failures without introducing errors.

Routing Objectives and Strategy

In this network the routing objectives are:

- Provide near optimal routes for virtual circuits
- Distribute the traffic load
- Control congestion
- Maximize the number of potential virtual circuits.

The routing strategy has three basic components:

- A measurement process for determining values of network state parameters
- An analysis algorithm to evaluate routes
- A routing control mechanism for deciding how to route traffic.

The measurement process is solely a local function. Each node in the network is responsible for maintaining and updating information about the links attached to it and the resources available within the node for new virtual circuits.

Costs are route evaluation criteria. They are held in the route table. They may be developed from parameters such as delay, bandwidth available, or other route parameters that influence route selection. Cost comparisons are based on the product of cost values from the route table weighted by the service class.

The route analysis algorithm determines the minimum-cost path from each node in the network to a particular node. This analysis is carried out for every node in the network to produce minimum-cost paths between a given node and all other nodes. Similar route analysis algorithms have been published.[2,3]

It is important that route analyses should be carried out frequently enough to ensure that virtual circuit requests receive near optimal routings. Conversely, the route analysis algorithm should not be invoked too frequently, because this would consume network resources unduly. We used a routing simulator to develop a suitable compromise for the interval between route analyses.

Within the network, each node is responsible for initiating its own route analysis. There are several situations in which a node may initialize such a route analysis:

- Periodic updates as outlined earlier
- Disaster recovery within the network
- A large difference between real costs and the costs held in the route table.

A particular node N_i initiates route analysis so that all other nodes can determine the minimum-cost route to N_i. The algorithm is the same for each node except the originating node.

Route analysis begins when N_i sends a cost message to each of its neighbors. The cost message has three parameters:

- gen#. The current generation of route analysis for N_i.
- Destination. The originator of the cost message. This is used as the index for the route table.
- C_{ji}. The cost of the route from N_j to N_i, where N_j was the last node to receive the cost message.

Every node maintains a routing table indexed by destination node. Each entry contains the most recent generation number for route analysis initiated by that destination, the current minimum route cost to the destination, and the adjacent link along this route to the destination.

When a node N_j receives a cost message from a neighbor node N_i, two comparisons are made. If the gen# is the same as that stored, the stored cost value becomes the lower of the stored route cost and the received route cost. If the gen# is a new one, then the gen# and the cost value are both updated. To make the cost comparison, the node must add the cost of the adjacent link over which the message

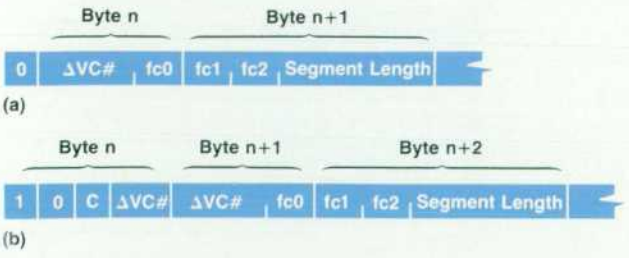


Fig. 10. *(a) Normal header format. (b) Extended header format.*

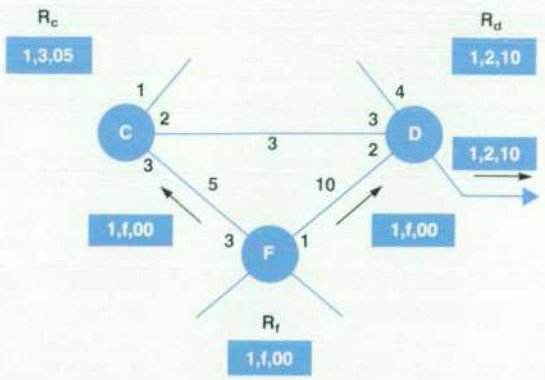


Fig. 11. *Messages and initial routing tables for routes to node F.*

was received to the cost contained in the message to arrive at a value for the total cost of the route. Generation numbers are cycled (modulo 16) to keep the gen# value within a limited range. However, the range is large enough to ensure that there is no ambiguity in gen# values.

For example, consider the partial network in Fig. 11. Node F sends out its initial routing cost message. This will consist of Destination = F and $C_{ji} = 0$ since the node is only speaking for itself. When D and C receive this message, each adds the cost of the link from F to the message, storing the complete message and the name of the link in its own routing table. During the next phase, both C and D transmit a routing cost message for the route to F with the information as collected in their routing tables. For simplicity, this is only shown on the diagram as being sent by node D, but in practice, both C and D would send routing messages to all their neighbors. Routing performance is improved by not sending messages back to originating nodes such as F. Further improvement is provided by not forwarding new messages when routing cost changes are trivial.

The message sent from D to C would then be 1,f,10, to which C would add the cost of the link to port 3 on D, giving a routing table entry of:

R_c = gen#, port of node C, total route cost (C to F)

or, in this case,

R_c = 1,2,13

Since R_c already has an entry of R_c = 1,3,05 and this is a lower-cost route, the data from D is discarded. Using this technique, the algorithm quickly converges and is stable. In practice, the route table is extended to include several of the most recent generations. This allows overlapping circuit setup and route computation.

The algorithm is shown in Fig. 12. It has the following attributes:

- The circuit is routed entirely according to a single cost generation. This avoids inconsistencies.
- Route determination is on a per-circuit basis. Additionally, the network provides a reroute-on-fail mechanism to protect against link or node failures. This mechanism can also be invoked if a routing becomes grossly suboptimal.
- Nodes disappear from the routing table if their information is old. That is to say, nodes are removed from the routing table if an update has not been received within a reasonable time limit. A limit of ten times the update period is sufficient, since users will not experience a connection failure because of an attempt to use a route that is no longer available. In such cases it is the setup that will fail.

Making a Connection

To establish a circuit, the originating device will assign an identification number to the connection. This is unique for the originating node and will be used to identify all subsequent programmatic requests. At the same time, an extended virtual circuit description is also prepared and a request is given for the access interface to send a connect message to the local node. The common control processor at the local node chooses the outbound link for the requested connection based on its route cost table, and selects the lowest available virtual circuit number for the circuit on that link. This is forwarded along the link command circuit with the connect message to the next node.

At each node, the connect request is checked to make sure that the service it requests can be satisfied. If not, then a reply is sent over the full-duplex link to the originator with a connect denial message. The connect denial message travels back over the outbound route, releasing reserved resources at each node. If a node can honor a connect request, it reserves resources, modifies the request to indicate the current route cost, and forwards the request towards the destination, using the local routing table.

When the connect message reaches a node having the destination address, the node completes an access link specific transaction with the addressed device to complete the circuit. If the connect request is accepted, the called node creates a connect grant message, which travels back along the route. The grant message carries the actual costs of the circuit as routed, so the originator can decide whether the circuit is acceptable. The circuit is then ready for full-duplex traffic.

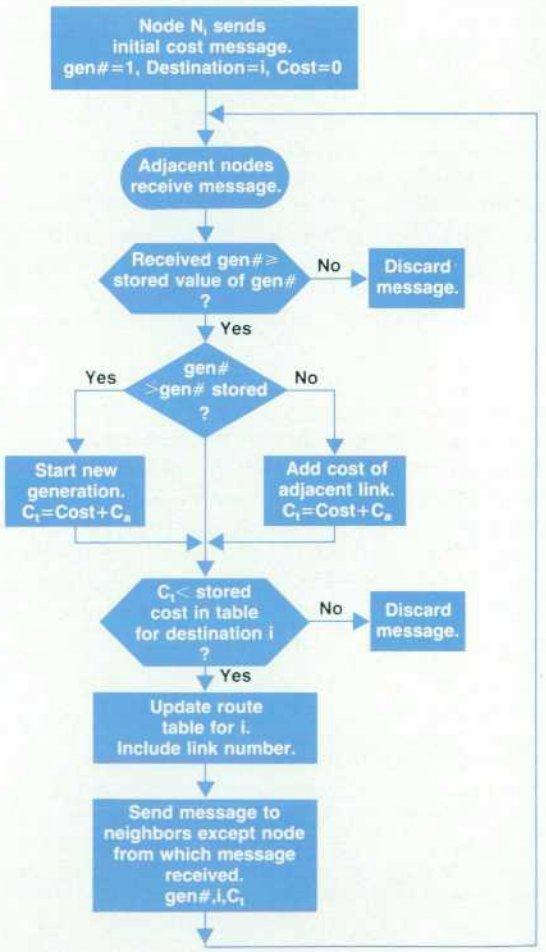


Fig. 12. *Routing algorithm.*

Automatic Reroute

Rerouting can take place either to improve service or to deal with failures encountered in links or nodes. In the latter case, the objective in rerouting will be to preserve the connection and to honor the promise of data delivery in sequence and without error. The commitment to delay time or bandwidth may not be met since this will depend upon the remaining topology and the traffic capacity of the remaining network.

The mechanism makes use of foldback nodes, which are the nodes detecting a broken link or a node failure between them. Fig. 10 shows the flow control bits (fc), which provide a two-hop acknowledgment mechanism. These flow control bits are part of the virtual circuit header, and thus provide control information between nodes through which the particular virtual circuit is routed. This allows for retention of data at a node even when a link or node farther along the virtual circuit has failed. By virtue of these acknowledgments, it is also possible to identify the point in the data stream at which a failure occurred, so that the foldback node can then transmit data from its buffers back to a point where a new connection can be established. When the buffer contents have been sent back to the point of departure from the original route, then the originating node will be allowed to resume data transmission, giving the effect of a perfect virtual circuit with just a short delay introduced by the retransmission period.

Routing Simulation

Extensive testing by simulation has shown that the dynamic routing algorithm is very effective in distributing load on alternate paths without introducing excessive control overhead. The convergence of the algorithm was proven both in theoretical arguments and in tests run on a simulator using several network examples. The number of update messages exchanged on each link during the routing table computation at all nodes has been shown to grow linearly with the number of nodes in a typical network. This corresponds to the theoretical lower bound that applies to any dynamic routing algorithm. However, the upper bound for this algorithm allows the number of messages to grow as fast as the square of the number of nodes. For this situation, we introduced a modification to the algorithm called a hold-down. The hold-down operates to delay the propagation of routing cost updates on slow (and generally more costly) links while transmitting the updates immediately on low-cost links. This technique does indeed help in pathological situations, but offers no advantages or penalties in those networks where the number of cost messages tends toward the lower bound. Judicious use of the hold-down can restrict the number of update cost messages to a linear growth.

The load leveling properties of the algorithm have been shown to use multiple paths very effectively. In simulation testing, the performance for multiple-path situations came within 15% of the theoretical upper bound. The ability to make good use of multiple paths depends on the rate at which the routing tables are refreshed. Intuitively, one expects that the routing update period should be small compared with the average user session duration. This was confirmed by experimental work: for an average session duration of 3 minutes, the throughput degraded progressively if the update intervals were increased from 5 seconds to 100 seconds. A good operating point in these tests was around 20 seconds.

Acknowledgments

The development of the architecture described in this article has been very much a team effort. All the members of the development team have made significant contributions that led to a successful implementation of a network using these principles. Specifically, the authors would like to thank David Means, Bruce Hamilton, Phil Curtis, and Alan Maloney for their contributions to the development of the architecture. Additionally, we wish to acknowledge the contributions made by Wayne Lichtenberger, Frank Weiss, Dennis Young, and Al Vigano in the design and implementation of the system.

References

1. J.S. Ma, "On the Impact of HDLC Zero Insertion and Deletion on Line Utilization and Reliability," *IEEE Transactions on Communications*, Vol. COM-30, no. 2, February 1982.
2. P. Merlin and A. Segall, "A Fail-Safe Distributed Routing Protocol," *IEEE Transactions on Communications*, Vol. COM-27, no. 9, September 1979.
3. A. Segall, "Advances in Verifiable Fail-Safe Routing Procedures," *IEEE Transactions on Communications*, Vol. COM-29, no. 4, April 1981.

Authors

February 1989

6 Time Varying Frequency Analyzer

Mark Wechsler

As a project manager for the HP 5371A Frequency and Time Interval Analyzer, Mark Wechsler coordinated measurement hardware development efforts. In an earlier staff position, he worked on the HP 5350A and 5351A, designing hardware for these Microwave Frequency Counters. He is now project manager for a new product. Mark came to HP as a student in the summer of 1979. He returned full-time a year later, when he had earned his BSEECS degree at the University of California at Santa Barbara. He was born in Scranton, Pennsylvania, and lives in Belmont, California. Mark's leisure activities include boardsailing, skiing, softball, and volleyball.

13 Firmware System Design

Lisa B. Stambaugh

Lisa Stambaugh worked at HP during summer recess and returned full-time after her graduation in 1980. Her BSEECS degree is from the University of California at San Diego and her MSCS degree is from Stanford University (1985). As software project manager, Lisa's responsibilities included the HP 5371A Analyzer. Her earlier projects involved software design for the HP 5350A Microwave Counter and HP 5344A Source Synchronizer. The use of firmware in instrumentation and software quality are among her main professional interests, and she has published an article in Microwave Systems News about firmware innovations in counters. Lisa was born in Binghamton, New York, is married, and has a son and a daughter. She lives in Fremont, California. In her spare time, she enjoys aerobics, dance, cooking, baking, and needlework.

Terrance K. Nimori

As a development engineer at HP's Santa Clara Division, Terry Nimori designed the configuration menus and numeric data screens, wrote the menu processing firmware, and contributed to the definition of the measurement capabilities of the HP 5371A Analyzer. He also helped develop a demonstrator box for simulating typical application signals for the instrument. Among past assignments, he designed firmware for the executive and HP-IB functions of the HP 5334A Universal Counter and taught classes on HP-IB fundamentals. User interfaces and instrument programming languages are his main professional interests. Terry joined HP in 1979, after receiving his BS degree in engineering from Harvey Mudd College. He was born in Honolulu and lives in Cupertino, California. As an amateur photographer, he recently netted several awards in newspaper-sponsored contests. He is also interested in astronomy, traveling, and ham radio (KH6HIQ).

21 Table-Driven Help

Lisa B. Stambaugh

Author's biography appears elsewhere in this section.

24 Input Amplifier/Trigger

Johann J. Heinzl

Since he came to HP in 1972, Johann Heinzl has contributed to the development of numerous products, including the HP 4935A, 4940A, 4942A, 4943A, and 4945A Transmission Impairment Measuring Sets. He designed the input amplifiers for the HP 5371A Analyzer. Before joining HP, he worked in the United Kingdom for Marconi Instruments. He is named coinventor in a patent for a modulation system which allows simultaneous passage of both measurement and control signals through a single cable. Johann has written and coauthored a number of articles for international journals, including the HP Journal. He describes his birthplace as a "small village" in Austria, is married, and lives in Los Gatos, California. His leisure interests are tennis, skiing, gardening, house remodeling, travel, and modern art.

28 Phase Digitizing

David C. Chu

David Chu originated the event/time continuous-count architecture and the continuous counter used in the HP 5371A Frequency and Time Interval Analyzer. He also produced the algorithms for processing event/time data. In the past, he worked on time interval averaging for the HP 5345A Electronic Counter and, as originator and project manager, on the HP 5370A Universal Time Interval Counter. David joined the Santa Clara Division of HP in 1962. During a sabbatical leave, he served as professor at Cuttington College in Liberia (West Africa). His BSEE degree is from the University of California at Berkeley (1961), and his MSEE (1962) and PhD (1974) degrees are from Stanford University. He has published some 20 technical papers on the subjects of timing, optics, circuits, signal processing, measurement architecture, and information theory and holds a number of patents in these fields. He is a member of the IEEE and the OSA. David was born in Hong Kong, is married, and has two children. He lives in Woodside, California. His favorite pastimes are sailing, boardsailing, and skiing.

35 Measurement Hardware

Paul S. Stephenson

Paul Stephenson joined the Santa Clara Division of HP in 1985, after earning his MSEE degree from Stanford University. His BSEE degree is from the University of Washington (1979). As a development engineer on the HP 5371A Frequency and Time Interval Analyzer, he worked on the design and development of the sequencer IC and interpolator assembly. In the years 1979 to 1985, Paul designed high-speed digital imaging systems for nuclear physics research at the Lawrence Livermore National Laboratory. He is a member of the IEEE and has coauthored articles about solid-state imaging and a voice communication device. Born in Corvallis, Oregon, Paul is married and lives in Pleasanton, California. In his off-hours, he enjoys music, boardsailing, and skiing.

42 Voice and Data Network

Howard L. Steadman

Howard Steadman joined HP Laboratories in 1977 after holding positions with major computer developers like Digital Equipment Corporation, Tymshare Incorporated, and SCI. System architecture and design are his main professional interests. As director of HPL's Communications Systems Laboratory, he lead the development of the Voice and Data Network architecture. He is the originator of two patents related to computer technology and is a member of the ACM. Howard's BA degree in mathematics is from the University of Colorado (1963). He is active in a homeowners association in Los Gatos, California, where he lives. He was born in Colorado Springs, Colorado, is married, and has a small daughter. In his spare time, he is building his own home.

Robert Coackley

Telecommunications, especially the convergence of telephony and computer networks, are Bob Coackley's main professional interests. Since he joined the Queensferry Telecom Division of HP in 1970, he has served in various managerial positions, and at different facilities in the U.K. and the United States. During development of the Voice and Data Network, he was a department manager at the HP Laboratories. He worked for British Telecom in a previous position. He holds four patents describing measurement technology for magnetics and telecommunications. Bob's BSc degree (1966) is in electrical engineering, and he is a fellow in the British IEE. He now is president of LC COM, a subsidiary of Tektronix, Incorporated. He was born in Stockport, England, is married, and has two grown children. He lives in Sunnyvale, California. In his off-hours, Bob teaches telecommunications at the Golden Gate University; he also likes sailing and house renovation.

52 Multifunction Synthesizer

Fred H. Ives

The digital waveform synthesis IC and the HP 8904A Multifunction Synthesizer were Fred Ives' primary responsibilities as project manager. He has held a similar position in the development of the DAC/sampler for the HP 8770A Arbitrary Waveform Synthesizer and has worked as an engineer on the design of the HP 8642A and HP 8662A Synthesized Signal Generators. He continues to head signal

generator projects. Fred earned his BS and MSEE degrees at the Massachusetts Institute of Technology in 1972, the year he also joined HP. He has coauthored an article about the HP 8770A Arbitrary Waveform Synthesizer and is named coinventor in several patents involving synthesized signal generators and high-speed IC design, his main professional interests. Fred was born in Margaretville, New York, is married, and lives in Veradale, Washington. His favorite pastimes include boating and other outdoor activities, landscaping, and the stock market.

57 DWSIC Architecture

Mark D. Talbot

Mark Talbot is an R&D development engineer at HP's Spokane Division. His list of past projects includes the HP 8656B and HP 8663A Synthesized Signal Generators in addition to the HP 8904A Multifunction Synthesizer discussed in this issue of the HP Journal. Before coming to HP in 1981, he worked at Burroughs Corporation as a test engineer and at ITT/Federal Electric Corporation as a senior engineer. Two pending patents are based on his designs, one of them on the digital waveform synthesis IC. Mark served in the U.S. Army at the rank of sergeant. He was born in Modesto, California, is married, and has two daughters. He lives in Liberty Lake, Washington, where he is active in his church. His off-hours activities include jogging, fishing, square dancing, and camping.

62 DWSIC Development

Dale R. Beucler

When he joined HP in 1978, development and production engineering on NMOS-III RAM ICs were among Dale Beucler's first projects. During development of the HP 8904A Multifunction Synthesizer, he worked on the digital waveform synthesis IC, and IC design continues to be his main professional focus. Dale's BSEE degree is from the California Polytechnic State University at Pomona (1978). He was born in West Covina, California, is married, and lives in Fort Collins, Colorado. As his leisure activities, he likes running, hiking, skiing, and tandem bicycling with his wife.

James O. Barnes

As a development engineer at HP's Colorado Integrated Circuit Division, Jim Barnes was responsible for the design of the digital waveform synthesis IC for the HP 8904A. In prior assignments, he helped develop the NMOS-III process and was active in marketing and training associated with HP's in-house IC technologies. His professional specialties, integrated circuit design and design methodologies, are also focal to his present activities. Before coming to HP in 1979, Jim worked for five years in the field of medical imaging research and served as visiting faculty member at Carnegie-Mellon University, his alma mater. He has a BS degree in engineering physics (1966) and a PhD degee in physics (1974). He has published nine papers and articles on the subjects of solid-state physics, medical imaging, and sensors. Jim was born in Chicago, Illinois, is married, and has three children. He lives in Fort Collins, Colorado. Fishing, photography, and playing soccer are his favorite pastimes.

Craig A. Heikes

As a development engineer, Craig Heikes was responsible for the design of the digital waveform synthesis IC used in the HP 8904A Multifunction Synthesizer. Among his many past projects are design of the HBD tool and the schematic-capture internal tool. He first came to the HP Boise Division as a summer employee in 1980, then joined the Colorado IC Division full-time in 1982. Craig's BSEE degree (1981) and MSEE degree (1982) are from Montana State University. He is a volunteer for Outdoor Colorado, an environmental organization. Born in Pocatello, Idaho, he lives in Fort Collins, Colorado. His off-hour activities include both water and snow skiing, hiking, backpacking, aerobics, and motorcyling.

66 Analog Output System

Thomas M. Higgins, Jr.

The reference loop and the floating output amplifier conversion were Tom Higgins' main responsibilities in the development of the HP 8904A Multifunction Synthesizer. He has worked on a gallium arsenide pulse modulator in the past, and recently started work on a new product development. Tom attended Montana State University, where he received his BSEE degree in 1985, the year he joined the Spokane Division of HP. Pursuing continued study, he is working toward an MSEE degree at Washington State University. Born in Alexandria, Virginia, Tom resides in Spokane, Washington. As his favorite pastimes, he lists wave-jumping in the Columbia Gorge, skiing, woodworking, and restoring old cars and his Victorian house.

70 Firmware Design

Mark D. Talbot

Author's biography appears elsewhere in this section.

73 Synthesizer Applications

Kenneth S. Thompson

A member of the marketing staff and a product planning engineer at HP's Spokane Division, Ken Thompson was responsible for the introduction of the HP 8904A Multifunction Synthesizer. He served in a similar capacity when the HP 8903B Audio Analyzer was developed and introduced. Ken joined HP in 1983, the year he received his BSEE degree from Brigham Young University. Ken's primary interests are in analog electronic design. He was born in Boise, Idaho, and now resides with his wife and three children in Spokane, Washington. He is active in his church and enjoys flying radio-controlled model planes, building stereo audio equipment, and playing his electronic synthesizer.

77 Testing and Process Monitoring

David J. Schwartz

While attending college, Dave Schwartz worked as an avionics technician. He joined HP in 1980, when he graduated from the California Polytechnic State University at San Luis Obispo with a BSEE degree. As a production engineer, he worked on such products as the HP 8901B Modulation Analyzer and the HP 8662A and HP 8663A Synthesized Signal Generators. When he was later assigned to test engineering, his projects included the HP 8642A, HP 8904A, and HP 8645A Synthesizers. Dave has written and coauthored articles for Microwave Systems News and the Microwave Journal. He was born in New Rochelle, New York, and lives in Post Falls, Idaho. He is an amateur pilot and also enjoys photography and sailing.

Alan L. McCormick

Alan McCormick attended the Spokane Technical Institute, where in 1983 he received an AAS degree in digital electronics. He came to HP in the same year. His assignments as a line technician included providing production liaison on the HP 8903A and HP 8903B Audio Analyzers and on the HP 8904A Multifunction Synthesizer. He also served as a programming instructor. Alan was born in Santa Ana, California. He resides in Spokane, Washington, is married, and has a daughter. His wife is a receiving supervisor at HP's Spokane Division. His outside interests include music and audio engineering, particularly sound reinforcement techniques.

Multifunction Synthesizer for Building Complex Waveforms

The HP 8904A uses digital synthesis and VLSI technology to provide a highly reliable tool for demanding applications like VOR, ILS, FM stereo, and communication signaling.

by Fred H. Ives

MODERN COMMUNICATIONS SYSTEMS employ complex modulation formats, sometimes using subcarriers or time varying signals to increase their capacity and usability. The divergent testing requirements imposed by these systems have been typically solved with one-of-a-kind custom solutions. This is a costly and sometimes unreliable solution. The HP 8904A Multifunction Synthesizer (Fig. 1) was developed to provide a low-cost, high-performance alternative to these application-specific complex waveform synthesis solutions.

The HP 8904A uses VLSI technology to create complex waveforms from six fundamental waveforms: sine, square, triangle, ramp, Gaussian white noise, and dc. Sine waves are provided in the range from 0.1 Hz to 600 kHz with 0.1-Hz resolution. The square, ramp, and triangle waveforms are available from 0.1 Hz to 50 kHz. All waveform values in the HP 8904A are digitally calculated in real time by HP's digital waveform synthesis IC (DWSIC). The use of this IC results in signals that have well-defined accuracy and exact repeatability.

The standard HP 8904A routes all signals to a single output, and with Option 002 a second floating output can be added, providing a separate signal for two-channel applications. Option 001 adds three more identical synthesizers (channels) which can either modulate the first synthesizer or be summed to the output (see Fig. 2). Frequency, amplitude, waveform, phase, and destination can be independently set for each synthesizer. Available modulation types for channel A include AM, FM, ϕM, DSBSC (double-sideband suppressed carrier), and pulse modulation. The HP 8904A provides fast hop in frequency, phase, and/or amplitude with Option 003. This feature can be used to create digital modulation formats. By hopping in frequency, FSK (frequency shift keying) and other frequency switching modulation formats with up to 16 frequencies can be generated. BPSC, QPSK, and other phase shifting formats can be made by hopping in phase.

The HP 8904A can be used in applications such as navigation, commercial electronics, audio testing, and communications signaling. For navigation applications the HP 8904A, used as a modulation source in combination with an RF signal generator, can generate VOR (VHF omnirange) and ILS (instrument landing system) signals (see Fig. 3). These signals are used by modern aircraft for navigation. For VOR, channel B is used to frequency modulate channel A, while channel C is summed with the modulated channel A. The bearing angle can be changed by altering the relative phase of channel C. The minimum bearing angle resolution is 0.1 degree. Since the entire VOR composite waveform is calculated in real time by the digital waveform synthesis IC, the HP 8904A Option 001 can deliver typical bearing accuracy on the order of ±0.05 degree. High accuracy can also be achieved in the generation of the instrument landing system localizer signal, which determines the center of the runway, and the glide slope signal, which determines the angle of the aircraft descent. These modulation signals are formed very precisely by the simple addition of 90-Hz and 150-Hz tones.

In conjunction with an RF signal generator, the HP 8904A can generate the signals required to test commercial FM stereo receivers. FM stereo multiplex composite signals require exceptional phase and amplitude linearity to maintain good stereo separation (see Fig. 4). The HP 8904A can generate stereo composite signals that typically maintain greater than 65 dB separation over the full 20-Hz-to-15-kHz

Fig. 1. *The HP 8904A Multifunction Synthesizer uses the latest VLSI technology to create complex waveforms from digitally synthesized sine, square, triangle, ramp, Gaussian white noise, and dc waveforms. In addition to the standard internal synthesizer (channel), three more identical internal synthesizers can be added. The option that adds the three addional synthesizers also provides AM, FM, ϕM, DSBSC, and pulse modulation. Application areas for the HP 8904A include audio testing, navigation, commercial electronics and communications.*

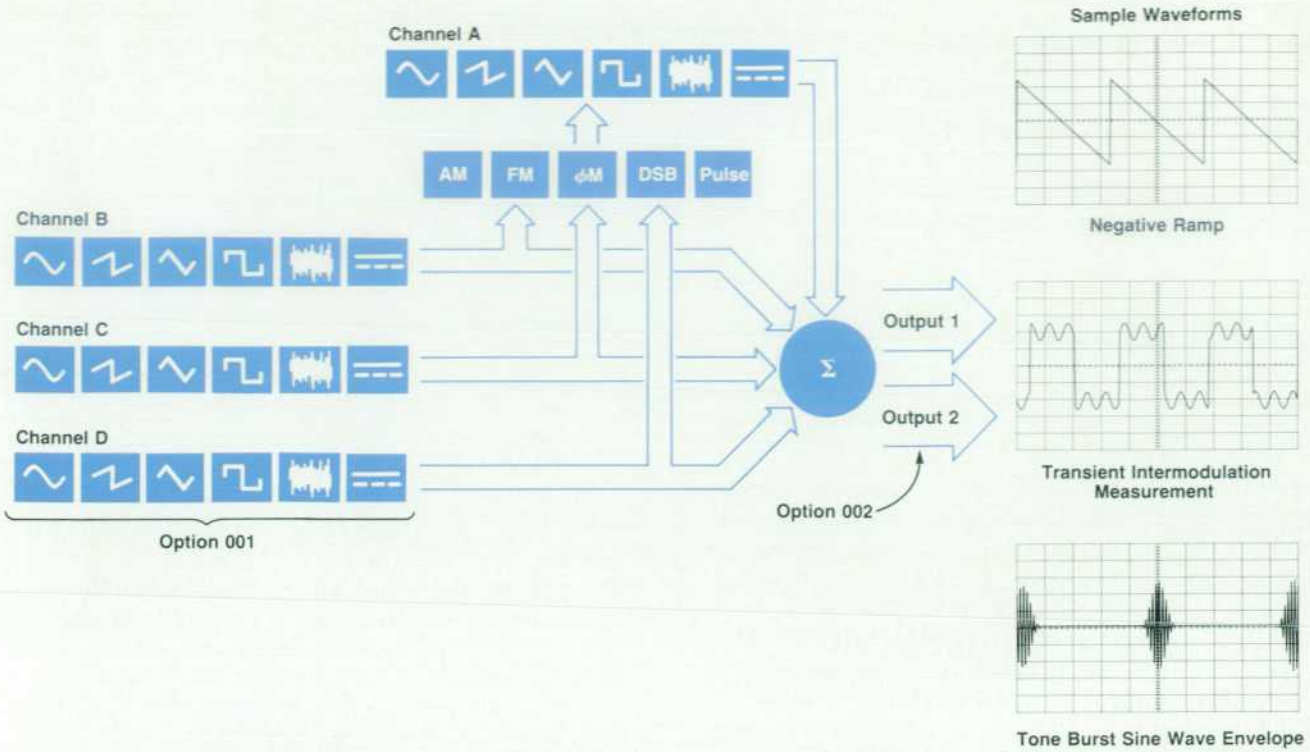


Fig. 2. *Block diagram of the internal HP 8904A synthesizers and some of the waveforms they can generate.*

audio bandwidth. Synthesizer accuracy gives an exact relationship between the 19-kHz pilot tone and the 38-kHz DSBSC subcarrier. Since the DSBSC modulation is calculated in the HP 8904A by the DWSIC, subcarrier suppression is greater than 70 dB. Unlike dedicated stereo encoders, the HP 8904A can vary frequency, phase, and amplitude for parametric testing. For example, the pilot tone amplitude can be varied to test phase-locked loop stereo decoder chips for lock range.

The HP 8904A can generate many types of signals used in audio applications. By summing or modulating with the four internal channels, the HP 8904A can generate signals that conform to international standards such as CCITT (International Consultative Committee for Telephone and Telegraph) twin tone, and SMPTE (Society of Motion Picture and Television Engineers) intermodulation test signals (see Fig. 5). Typical residual intermodulation distortion is less than −70 dB. Other useful audio signals that can be generated by the HP 8904A include the IHFM (Institute of High Fidelity Manufacturers) dynamic headroom test signal and a phase continuous linear sweep signal.

In addition to three internal channels, Option 001 also adds three sequence modes to the HP 8904A: tone sequence mode, dual-tone multifrequency (DTMF) sequence mode, and digital sequence mode. These sequence modes can be used to generate signals for paging radios, mobile telephones, and two-way radios. The tone sequencing mode allows entry of 16 different sine wave tones, each with an on time and an off time. From these 16 tones, sequences can be built up to a length of 250 tones. The minimum on or off time is 800 μs with 10-μs resolution, while the maximum value is 655.35 ms. Timing accuracy is better than ±20 μs.

DTMF (dual-tone multifrequency) mode allows generation of sequences of telephone-type signals up to 250 signals in length. In this mode the HP 8904A can generate the 16 standard frequency pairs defined by the Bell Telephone System (*Bell Technical Reference Publication No. 48005*). The minimum timing period for DTMF is 1 ms, with 10-μs resolution.

Digital sequence mode can generate digital bit streams up to 1000 bits in length. The minimum period in the digital mode is 100 μs with 10-μs resolution. On and off levels in the digital mode can be set to any value for stimulating different logic families and can be asserted high or low to follow standard logic conventions. For ease of use, data can be entered in binary, octal, or hexadecimal formats.

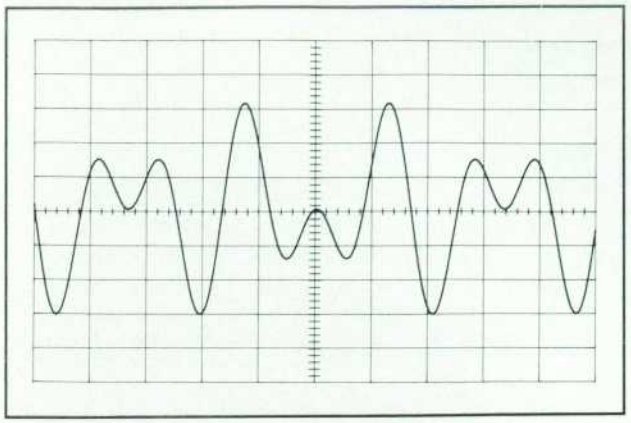
Digital Waveform Synthesis Integrated Circuit

The HP 8904A uses digital synthesis techniques, that is, direct mathematical calculation of the waveform point by point in real time. Digital hardware implements the mathematics. Fig. 6 shows the basic digital waveform synthesizer used in the DWSIC. At the heart of this synthesizer is a phase accumulator consisting of an N-bit binary adder and latch circuit. The output frequency of the phase accumulator is determined by the formula:

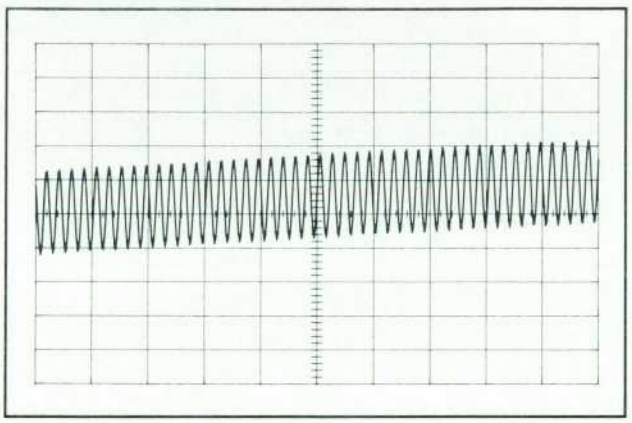
$$f_{out} = f_{in} \times f/2^N$$

Where f is the binary frequency input number, N is the accumulator length, and f_{in} is the clock rate of the accumulator latch. This formula shows that if $f_{in} = 0.1 \times 2^N$ hertz, the frequency resolution of this synthesizer is 0.1 Hz.

The additional adders in Fig. 6 cause the frequency and



(a)



(b)

Fig. 3. *Signals for navigation. (a) Instrument landing system (ILS) two-tone composite signal. (b) VHF omnidirectional range (VOR) composite signal.*

phase of the accumulator to vary with modulating signals. The phase ramp from the accumulator can be converted by a lookup read-only memory (ROM) or other logic to a sine wave or other waveforms. The multiplier placed after the sine ROM allows the overall amplitude of the waveform to be varied by a modulating signal. The final step after all the digital signal processing is the conversion of the signal back to an analog signal by a digital-to-analog converter (DAC).

The advantages of this digital implementation are many. The signals produced are exact in frequency, phase, and amplitude. Therefore, they are very repeatable. The analog drift problem is not there. Very precise component tolerances are not needed for low distortion and low drift, and the signal is the result of a precise and repeatable digital calculation.

Two basic topologies used in digital synthesis are the RAM-based lookup method and the accumulator-based calculation method. RAM-based lookup is used in arbitrary waveform generators, and has the advantage of allowing arbitrarily shaped waveforms to be generated. The low-frequency content and frequency accuracy of the signals produced are limited by the size of the RAM. This is because the RAM is clocked at a high speed to get an accurate representation of the signal in the time domain. Also, the

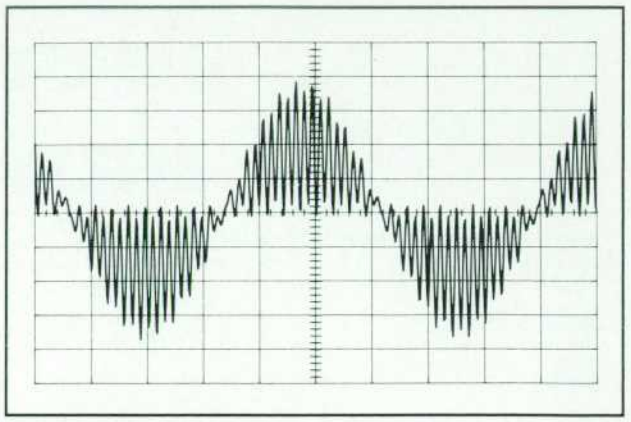


Fig. 4. *FM broadcast stereo composite signal.*

RAM must be loaded before the output waveform can be changed.

The digital waveform synthesis integrated ciruit uses accumulator-based waveform calculation because this method is particularly attractive for producing the signals used in communications. Many of these signals can be derived from sine, triangle, ramp, and square waves, and they can all be calculated in a very compact structure inside the DWSIC. The most complex signal to calculate is the sine wave. This signal is created by using a ROM. The ROM structure is much more compact than RAM and easily fits on the DWSIC.

The compact size of the synthesis circuits along with the high speed of the proprietary HP NMOS-III VLSI process[1] allows more features to be included. The accumulator design is fast enough to be used by one, two, or four synthesis channels with proper signal multiplexing. A very flexible control is included, allowing the DWSIC to operate in several different configurations. Also, the output signals can be changed quickly since the DWSIC can be set up quickly by simple microprocessor control. The design is compact enough to include other features also, such as a small RAM to allow faster switching between different operating states, and a noise generation circuit to produce broadband noise for testing communications circuits. See the article "Digital Waveform Synthesis IC Architecture" on page 57 for more details on the DWSIC.

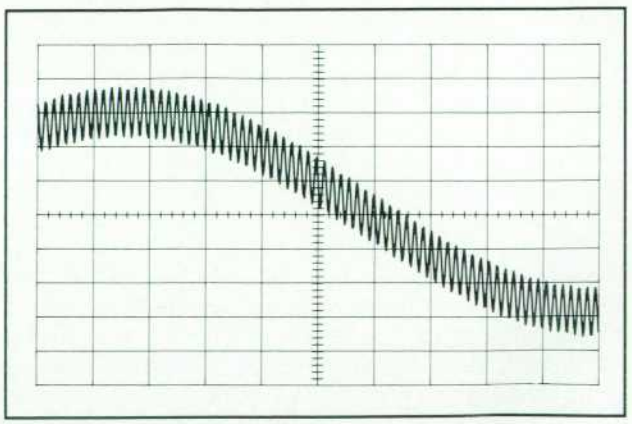


Fig. 5. *Intermodulation test signal.*

Mechanical Design of the HP 8904A

The product design objectives for the HP 8904A were to create a compact, lightweight instrument that has a low part count and is easy to assemble. A low-cost design and low audio noise from the fan were also very important specifications.

To achieve low cost, light weight, and minimum parts, several standard HP castings and sheet-metal parts were combined into folded sheet-metal parts. The four side strut castings and deck were combined into a single folded sheet-metal part, and the normally separate rear casting and rear panel were united into a single folded sheet-metal piece. Both of these fabricated parts are made from 0.9-mm-thick electrolytic zinc plated steel. The deck and rear panel come preassembled from the fabricator. The deck assembly consists of the rear panel riveted to the deck, four grounding posts, and 25 locking plastic printed circuit board mounting standoffs. Fig. 1 shows an exploded view of the mechanical construction for the HP 8904A.

To aid in assembly, the fan, line module, and voltage select switch are pop riveted onto the rear panel using pneumatic tools. Except for the serial number label, all warning and information labels are combined into a single label that covers the rear panel. Alignment marks are stamped into the fan grill area for easy, consistent location of the serial number label.

Since the rear panel already has several components riveted in place, the deck assembly is attached to the front casting by rivets. These rivets go through existing tapped holes in the front frame casting. We felt that it was unlikely that the casting would be replaced in the field. But, since the rivets are soft aluminum, they can be removed easily and the new cast frame can be reattached using screws and the existing tapped holes in the casting.

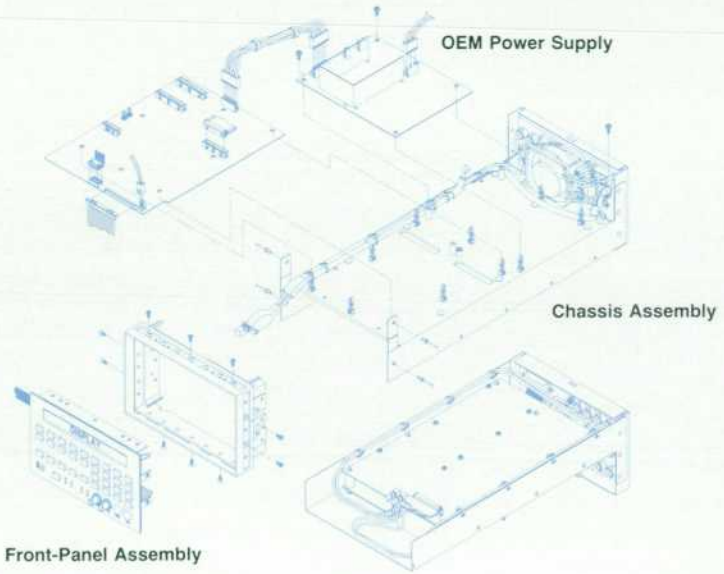


Fig. 1. *HP 8904A mechanical assembly.*

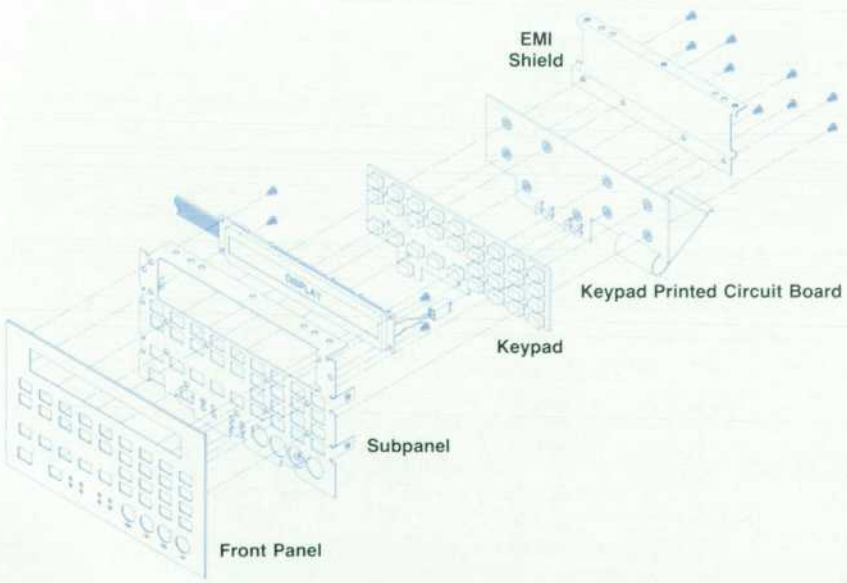


Fig. 2. *Front-panel assembly of the HP 8904A Multifunction Synthesizer.*

The front panel is designed to be built and aligned as a module. This module consists of a laminated polycarbonate dress panel and metal backing panel, a liquid-crystal display (LCD), a silicone rubber keypad, a keyboard, an EMI shield, and a mounting subpanel. The dress panel has the display window as an integral part of its construction. This assembly is shown in Fig. 2.

The new quieter noise level requirements were met through fan selection, design iterations, and testing. A low-power, high-efficiency, axial-flow fan and a stamped fan grill are used. The printed circuit boards and the sheet-metal deck then were moved away from the fan until the 5.0-dB specification was met. A good rule of thumb is that if you can hear the instrument running in a normal R&D lab environment, it is probably too loud to meet the 5.0-dB requirement.

The HP 8904A uses a custom OEM power supply. A switching supply was chosen over a linear supply for two reasons: size and efficiency. With a higher-efficiency supply, there is significantly less heat to remove from the instrument. The chief drawback to the switching supply is the amount of ripple present on the critical voltages. The supplier was able to achieve 15-millivolt ripple on those required voltages. Further postregulation is done on the analog and digital boards as required. The use of a vendor for a custom supply is Spokane Division's first attempt to develop an instrument power supply from a set of written specifications. This process went well and resulted in a reliable and efficient design.

Conducted and radiated emissions testing resulted in a few instrument modifications. A thin insulated shield is placed between the output boards. Additional screws secure the subpanel to the front casting and a shield is placed directly behind the LCD display. A series of small bumps are added to the rear sheet-metal panel to increase the number of grounding points to the instrument cover. Several passes were made at filtering the power line module and power supply until a good margin exists in the conducted emissions requirement.

Larry R. Wright
Product Designer
Spokane Division

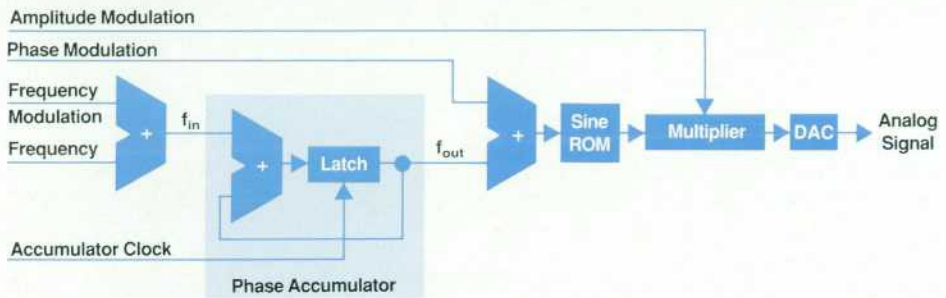


Fig. 6. *Digital synthesis, the accumulator method. All waveform values are digitally calculated in real time.*

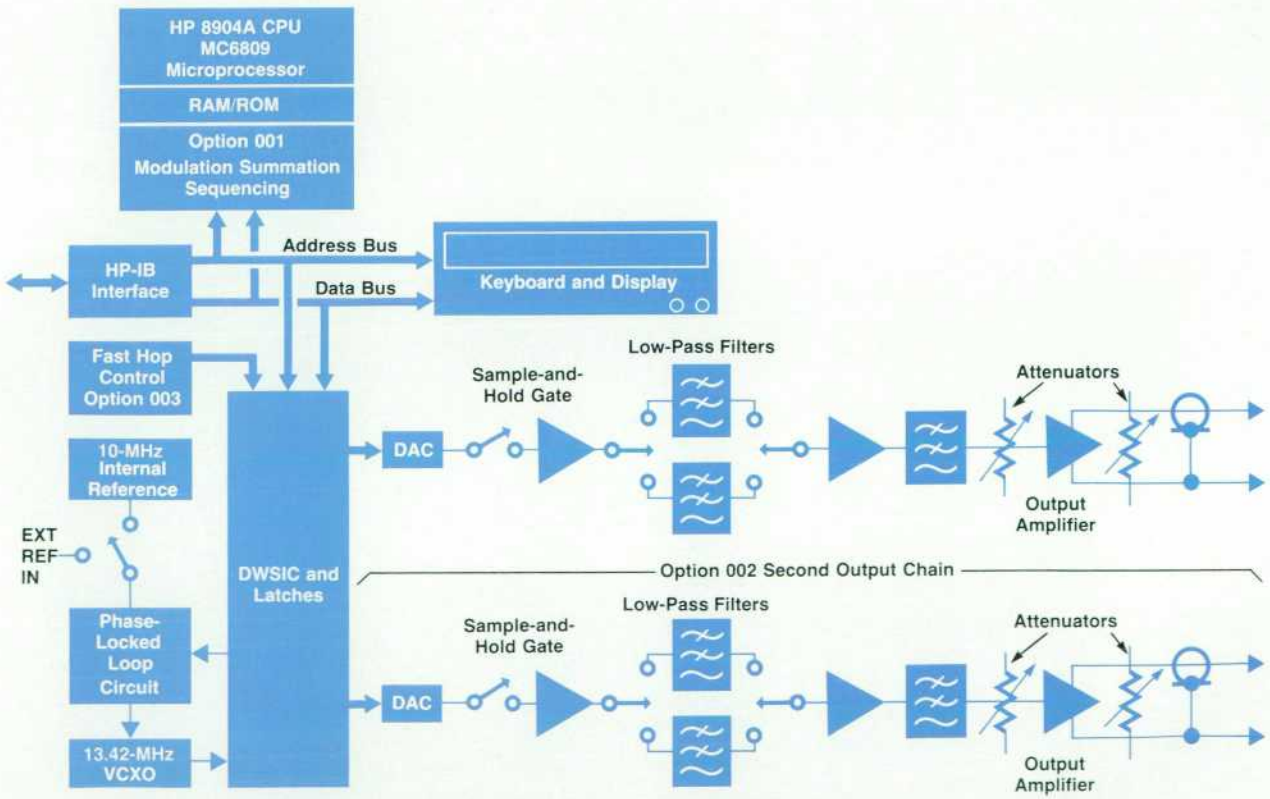


Fig. 7. *HP 8904A instrument block diagram.*

HP 8904A Instrument Design

The DWSIC made the design of the HP 8904A instrument very straightforward. Fig. 7 is a block diagram of the instrument showing the organization of control, digital signal generation, output channel signal conditioning, and external connections. The DWSIC takes care of digital generation of all the signals. The DWSIC's output is a 12-bit digital word. This word is clocked onto a bus connected to the channel's output section and signal conditioning circuits. The DWSIC does one other job; it contains an accumulator used to lock the reference phase-locked loop. This circuit allows the HP 8904A to be phase-locked to an internal or external 10-MHz reference.

The HP 8904A contains three different printed circuit boards: the digital board, the output board, and a board containing the keyboard contact closures for the flexible rubber keymat and four HP-IB status indicators. The digital board contains the microprocessor instrument controller with memory, the reference phase-locked loop, the DWSIC, and external interface circuitry.

The keyboard, display, and power supply connect to the digital board. The liquid crystal display module with backlight is supplied by a vendor with cables attached. The HP 8904A power supply was developed to our specifications by a vendor, and is supplied to us fully tested.

The output boards for channels A and B are identical and contain the digital-to-analog converter, sample-and-hold circuit, filters, amplifiers, attenuators, and overvoltage protection circuitry. A floating output amplifier is provided to enable the instrument to drive systems that are not at ground reference, and to minimize noise pickup.

In addition to the printed circuit boards, the HP 8904A has an HP-IB interface, a fast hop control bus, and reference clock inputs and outputs. This block diagram is very flexible and allows for several different configurations. All the options are mutually compatible and can be retrofit in the field if the customer needs to add capability at a later date.

Acknowledgments

The HP 8904A project team at HP's Spokane Division and the IC design team at HP's Colorado Integrated Circuit Division (CICD) developed the DWSIC to perform all the signal synthesis, modulation, and control in the HP 8904A. The Spokane Division's design team defined the external and internal specifications for the DWSIC, and the CICD design team used these specifications to design and develop the integrated circuit. I would like to express my gratitude to all the people involved in the DWSIC and the HP 8904A development at both divisions. Also, thanks to Ray Fried for his guidance and to Steve Holdaway for his support of the DWSIC development and the HP 8904A project.

Reference

1. J.M. Mikkelson, et al, "NMOS-III Process Technology," *Hewlett-Packard Journal*, Vol. 34, no. 8, August 1983, pp. 27-30.

Digital Waveform Synthesis IC Architecture

The digital waveform synthesis IC is the heart of the HP 8904A Multifunction Synthesizer. It provides a digital approach to the conventional analog functions of modulation and signal generation.

by Mark D. Talbot

THE DIGITAL WAVEFORM SYNTHESIS integrated circuit (DSWIC) is a digital waveform synthesizer on an IC. It incorporates many signal generation and control functions, is cost-effective, and has multiple uses. The design objectives for the DWSIC called for the following features and functions:

- Four concurrently operating channels with independent frequency, waveform, amplitude, and phase settings
- Amplitude modulation, double-sideband suppressed carrier modulation, frequency modulation, phase modulation, and pulse modulation on one of the channels
- Channels that can be selectively summed together
- Random access memory to provide rapid selection of up to 16 different settings of frequency, amplitude, and phase under internal or external control
- Modulation of one channel from another internal channel, from the sum of channels, or from an external input.

The digital waveform synthesis IC generates one, two, or four time-multiplexed channels (A, B, C, and D) of digitally synthesized waveforms. The output of the DWSIC is converted to analog signals by means of digital-to-analog converters (DACs). Fig. 1 shows a block diagram of the

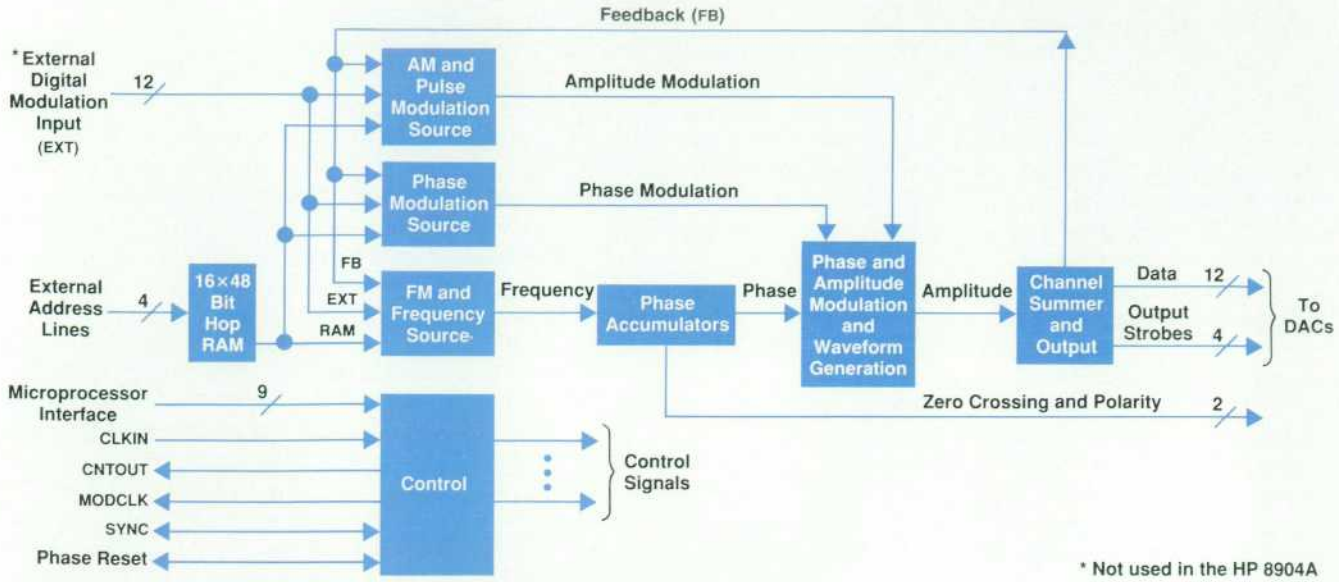


Fig. 1. *Digital waveform synthesis IC block diagram.*

DWSIC.

Control Block

The control block is the interface to a microprocessor and external control logic. With a six-bit data bus, one address line, a chip enable line , and a write enable line, the DWSIC appears to be a write-only peripheral to the controlling microprocessor. Phase reset and SYNC lines provide multiple DWSIC synchronization and fast external parameter change capabilities. The instrument clock input, CLKIN, runs at twice the required internal clock rate. The DWSIC outputs two clock signals: MODCLK and CNTOUT. MODCLK allows synchronization of external modulation inputs and data outputs, and CNTOUT is a fractional multiple of CLKIN.

A 24-bit register, PLLDIV, provides the denominator for the fractional multiply that generates the CNTOUT term. CNTOUT is the overflow from a 24-bit accumulator. The value for CNTOUT is determined by the formula: $\text{CNTOUT} = \text{CLKIN}/2 \times \text{PLLDIV}/2^{24}$. The CNTOUT term is used in a phase-locked loop to lock the CLKIN term to a reference in the HP 8904A.

The channel cycle rate can be selected as a function of the CLKIN rate. A number loaded into the clock divide register allows a range of CLKIN/2 to CLKIN/2048 to be selected.

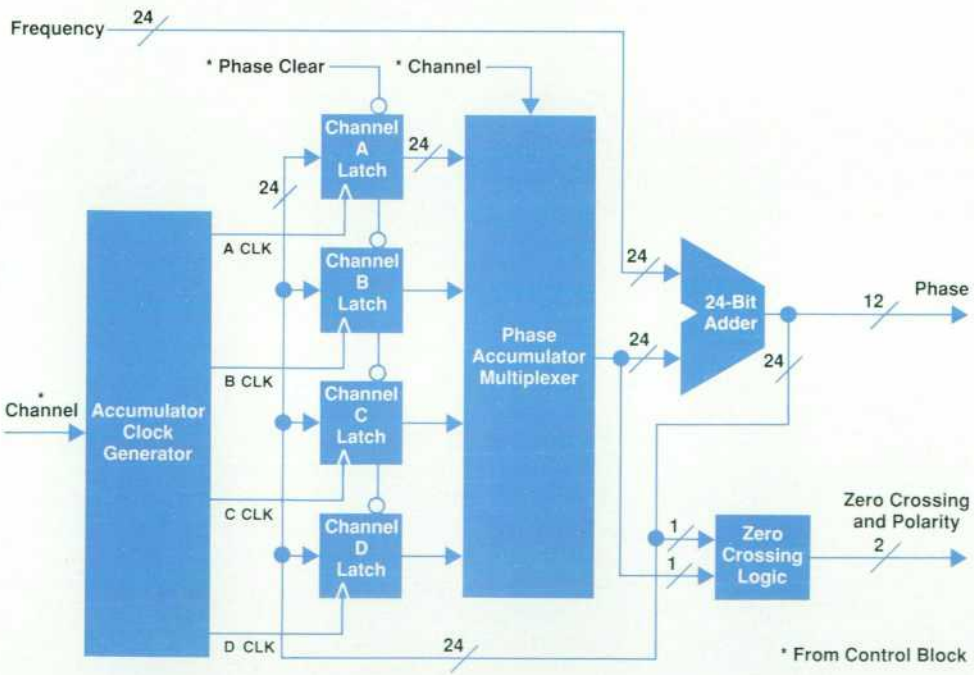


Fig. 2. *Phase accumulator diagram.*

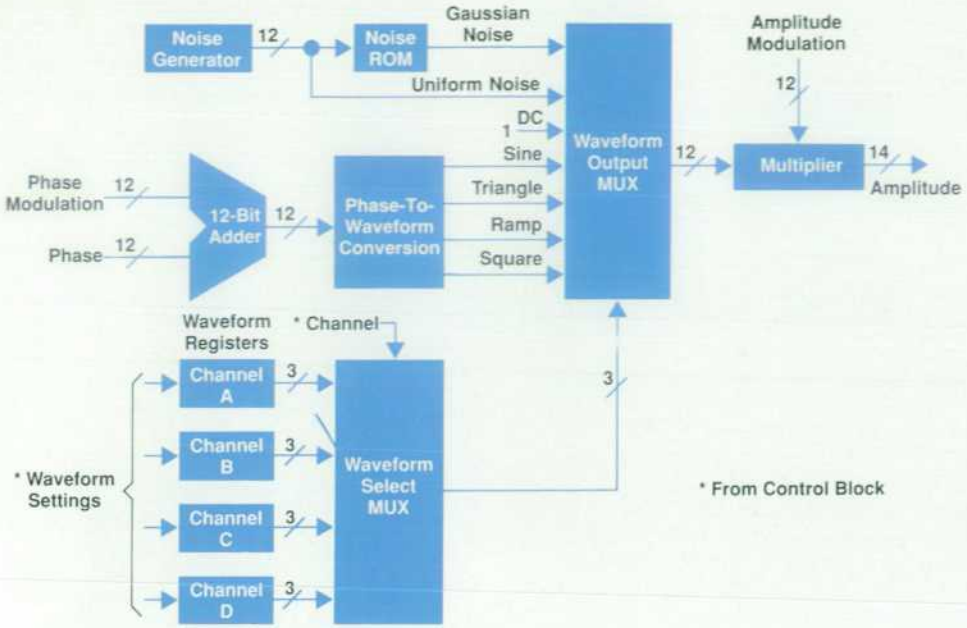


Fig. 3. *Amplitude and phase modulation and waveform generation diagram.*

Hop RAM

The hop RAM is a 16-word-by-48-bit RAM memory that enables the DWSIC to provide frequency, phase, and/or amplitude hopping for the HP 8904A. Up to 16 settings for frequency, amplitude, and phase can entered into the hop RAM for channel A. By changing the hop RAM address via the external address lines different hop settings can be selected at a very fast rate. Internal control logic handles the enable and disable settings of the three hop parameters. For example, phase and amplitude hopping can be disabled to produce only frequency hopping without having to remove the phase and amplitude settings from the hop RAM. This feature allows the generation of frequency shift keying signals such as those used in modems, pagers, and other tone signaling devices.

Phase Accumulator

The phase accumulator is the heart of the DWSIC (see Fig. 2). There are four phase accumulators, which are each 24 bits wide and provide 0.1-Hz frequency resolution. The channel timing signals are generated by the control block described above. These signals tell the DWSIC which channel is presently active. The DWSIC can be operated in one-channel mode (only channel A active), two-channel mode (channels A and B active), or four-channel mode (channels A to D active).

Under the control of the channel signals, the phase accumulator multiplexer selects the active channel's accumulator output. This is added by the 24-bit adder to the desired frequency value coming from the FM and frequency sources. The sum, which represents the present phase value, is clocked into the sourcing channel's latch and sent to the waveform generator. The clocks for the accumulators are generated by the accumulator clock generator, which is also controlled by the channel timing signals. The phase clear signal forces the contents of all channel phase latches to zero. This permits phase initialization of all four channels.

The zero crossing logic monitors the active channel and outputs two signals: a level indicating the polarity of the present phase value, and a pulse every time it crosses zero. Both of these signals are available at the output of the DWSIC.

Phase Modulation, Amplitude Modulation, and Waveform Generation

The phase modulation source shown in Fig. 1 provides

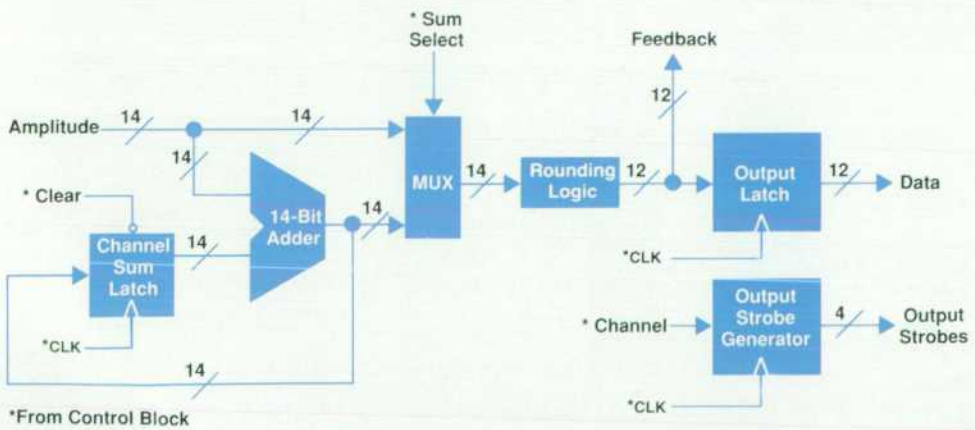


Fig. 4. *Channel summer and output diagram.*

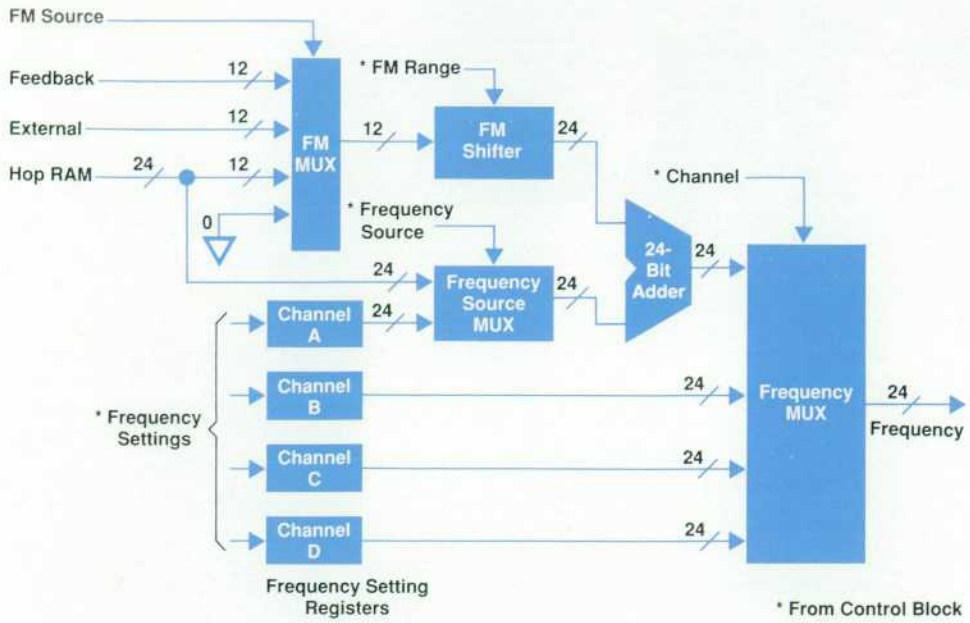


Fig. 5. *FM and frequency source.*

the phase offset values that are added to the accumulated phase values from the phase accumulator for each active channel (see Fig. 3). For channel A, this source can contain additional modulation data, and by adding this data to the accumulated phase values, the phase of channel A can be modified by a fixed or varying amount. This results in a phase offset and/or phase modulation for channel A only.

The resulting phase is converted to a magnitude value by the phase-to-waveform conversion logic. This block contains mathematical logic and one quadrant of a sine lookup ROM. The resulting waveforms are sine, triangle, ramp, and square. To generate dc waveforms, a constant value of one is used. Twelve bits of an independent 31-bit pseudorandom generator are used to provide uniform noise. This same data supplied to a uniform-to-Gaussian lookup ROM generates Gaussian noise data values. Seven bits of the uniform noise data are used to generate 11 bits of Gaussian data with an eighth bit used for polarity information.

Waveform settings, which come from the control block, are used to direct the desired waveform from the waveform output multiplexer to the multiplier. The multiplier is used to provide output level control for all channels and AM and double-sideband suppressed carrier modulation for channel A. The 14 most-significant bits of a 12×12 bit multiplication of the waveform data are sent to the channel summer.

Channel Summation and Output

The sum select, clock, and clear control signals allow the incoming amplitude values of the time multiplexed data to be selectively added and output to the rounding logic (see Fig. 4). Sums of channels A+B, A+C, A+D, B+C, B+D, C+D, A+B+C, A+B+D, A+C+D,

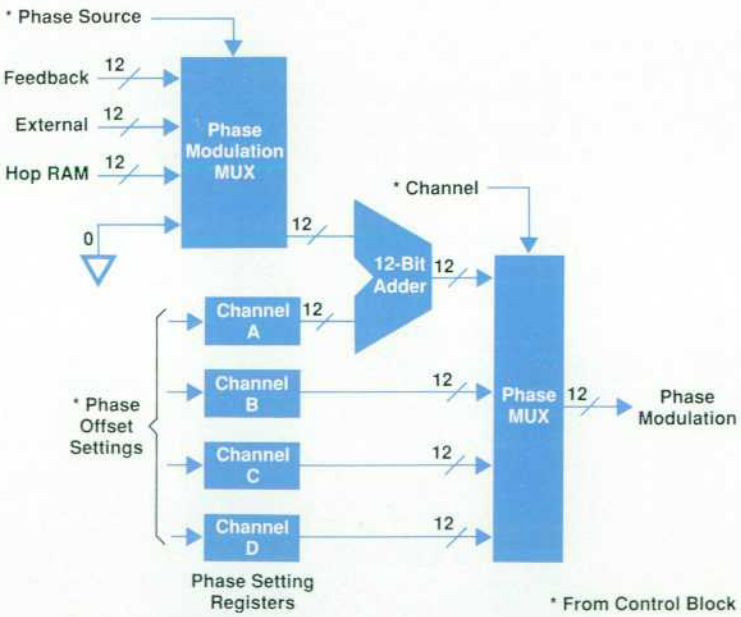


Fig. 6. *Phase modulation source.*

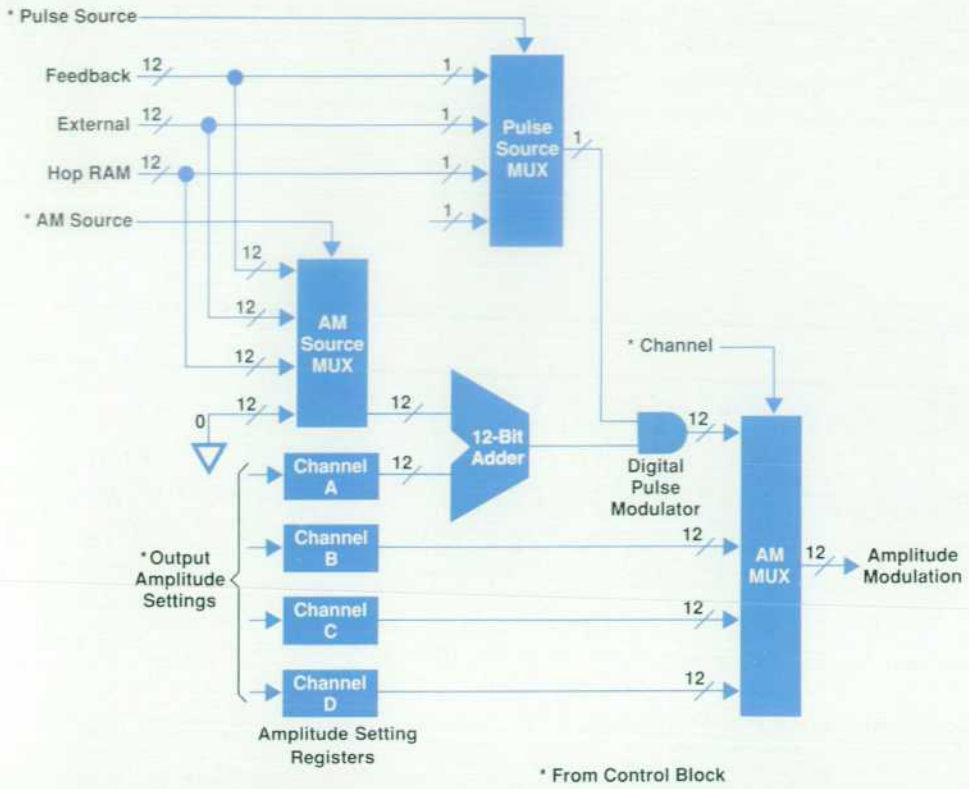


Fig. 7. *AM and pulse modulation source.*

A + B + C + D, and (A + B) followed by (C + D) are possible. A 14-bit adder is used to maintain the accuracy of the least-significant bits. The rounding logic strips out the lower two bits and converts either the summer output or the channel amplitude to 12 bits. This value is sent to the feedback bus and outside the DWSIC via the output latches and data lines.

Four output strobes are generated by the output strobe generator to qualify data on the data lines. Each of these strobes can be individually turned off or assigned to coincide with the output data of any one of the active channels.

Modulation Sources

The three modulation source blocks shown in Fig. 1 provide the internal AM, FM, phase modulation, and pulse modulation for the DWSIC.

Frequency and FM Sources. Fig. 5 shows the FM and frequency source logic. Frequency values stored in the frequency setting registers for channels B, C, and D are output to the phase accumulator logic via the frequency multiplexer. Channel A's value consists of the sum of the frequency source multiplexer output and the output of the FM shifter. Either twenty-four bits of hop RAM data or the channel A frequency setting are selectable by the frequency source control line. The FM source control signal selects either the feedback input generated by the channel summer, the external digital modulation input, 12 bits of hop RAM data, or zero. The 12-bit output of the FM multiplexer goes into the FM shifter. This block of logic aligns the 12-bit FM data and generates the 24 bits of frequency information. Its alignment is controlled by the FM range, which represents the amount of FM deviation desired. The 24-bit result is sign-extended for the most-significant bits and zero-padded for the lower bits. When added to the frequency multiplexer output the result is either a fixed or varying frequency value for channel A.

Phase Modulation Sources. The phase modulation multiplexer shown in Fig. 6 selects the value added to the channel A phase offset setting. As determined by the phase source signal, either the feedback signal from the channel summer, the external modulation input, 12 bits of hop RAM data, or zero can be selected. This sum (channel A) or the phase offset setting of channels B, C, or D constitutes the phase modulation output of the phase multiplexer as selected by the channel signal.

AM and Pulse Modulation Source. The AM and pulse modulation source (Fig. 7) is almost identical to the phase modulation source, but has an additional pulse source multiplexer. The pulse source multiplexer selects the most-significant bit from the feedback path, external modulation input, or hop RAM and provides pulse modulation data. Pulse modulation is accomplished by allowing the 12-bit sum of the channel A amplitude setting and the desired AM modulation source to pass unchanged or be forced to zero. In the case where pulse modulation is not desired, a value of one from the pulse source multiplexer can be selected to disable the digital pulse modulator.

SYNC Latches

In series with each of the frequency, phase, amplitude, and waveform setting registers of all four channels is a set of SYNC latches. In normal operation these latches operate in a transparent mode. By special configuration of the control logic, these latches can be configured to latch the setting register values. This permits the setting registers to have new values installed without affecting the present IC con-

figuration. At some point, if it is desired to change the DWSIC's parameters by either an internal control bit or the external SYNC control pin, the new contents of the setting registers can be latched into the latches.

Acknowledgments

Rob Dickerson, Bill Dickerson, Fred Ives, and Mike McNamee contributed to the the DWSIC block diagram. Pat Owsley and John Purviance modeled the noise generation circuitry. The signaling applications expertise of Gary Johnson led to the implementation of the SYNC latches and SYNC control logic. Special thanks to Ray Fried for his enthusiastic support of this project.

Development of a Digital Waveform Synthesis Integrated Circuit

The digital waveform synthesis IC is an excellent example of using custom VLSI in an instrument to reduce cost and increase functionality, accuracy, and reliablity.

by Craig A. Heikes, James O. Barnes, and Dale R. Beucler

CUSTOM DIGITAL VERY LARGE-SCALE INTEGRATED (VLSI) circuits are used extensively in many HP computer products, but are just beginning to show up in HP test instruments. The HP 8904A Multifunction Synthesizer represents an excellent example of using custom VLSI to reduce cost and increase functionality, accuracy, and reliability in a test instrument.

The digital waveform synthesis IC (DWSIC) is a digital approach to doing conventional analog functions of modulation and signal generation. The primary design challenges included implementing the highly parallel design as a data path layout, using behavioral modeling and verification to ensure accuracy, and designing new custom circuits (e.g., multiplier and ROM).

Chip Overview

Most computer chips can be characterized as having a bus-oriented architecture with large portions of the chip linked by a relatively small number of global buses and a few operations occurring simultaneously. In the DWSIC, data flows through four synchronized parallel pipelines containing 26 pipe stages, with 23 simultaneous operations taking place. The chip can run at a frequency of up to 16.7 MHz and perform 391 million logical and arithmetic operations per second. Four 12-bit channels of either sine, square, triangle, or ramp waveforms are generated with single-bit accuracy. In addition, the DWSIC provides for internal or external AM, FM, or phase modulation, uniform and Gaussian noise generation, and operation-point hopping. A more detailed description of the of the architecture of this chip can be found in the article on page 57.

HP's NMOS-III fabrication process has the density to allow this entire system to be integrated onto one chip.[1] Also important to the success of the project were the hierarchical block design (HBD) methodology and tools, and the extensive NMOS-III cell library.

A hardware emulator made up of MSI and LSI components was built early in the project to prove feasibility and to allow early firmware development. The emulator provides us with a chance to compare a discrete implementation with a custom integrated circuit solution. From Table

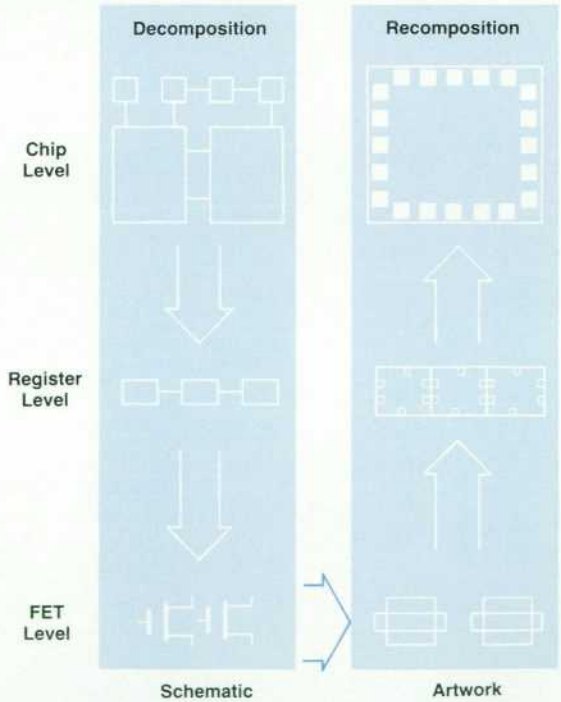


Fig. 1. *Hierarchical block design (HBD) design flow.*

I, which summarizes the comparison between the two implementations, it is clear that an integrated circuit solution has a definite advantage. Another important advantage of an integrated solution over a discrete solution is in reliability. A 100× improvement in reliability is not uncommon for a chip of the size and power given in Table I.

Table I
Emulator Chip Implementation

	Discrete Solution	Integrated Circuit Solution
Number of components	400 MSI, 2 LSI TTL	99,000 FET
Power	60 Watts	2.7 Watts
Relative Cost	50	<1
Volume	20,000 cm^3	25 cm^3

Design Methodology

The DWSIC was designed using a methodology called hierarchical block design (HBD). There are three main attributes of the HBD methodology: hierarchical blocks, data path design, and scan-path testing.

Using HBD, a chip is divided into hierarchical blocks. The logical design is first split into a few functional blocks, and then each functional unit is further divided. When this top-down decomposition is complete, the whole chip is represented as a hierarchical interconnection of simple blocks such as NAND gates, registers, or even MOS transistors. Recomposition takes place from the bottom up. A physical representation of each block is created and assembled in the reverse order from the logical decomposition. When the assembly process is complete, the chip is finished (see Fig. 1).

Chips designed with HBD are decomposed into at least two main function blocks called the data path and the programmable logic array (PLA). All data manipulations take place in the data path and all control and state machine operations take place in the PLA. Signal data from the chip pads is fed into the data path and control data is fed to the PLA. Additional functional blocks can also be included in the design.

The DWSIC has two physically distinct data paths, a PLA, a multiplier, and two ROMs. The parallel pipelines are built as two large data paths with all control handled by a PLA placed between them. External to the data paths, but still part of the data flow, are a 4K-by-11-bit ROM containing a sine function lookup table, an 11-by-12-bit

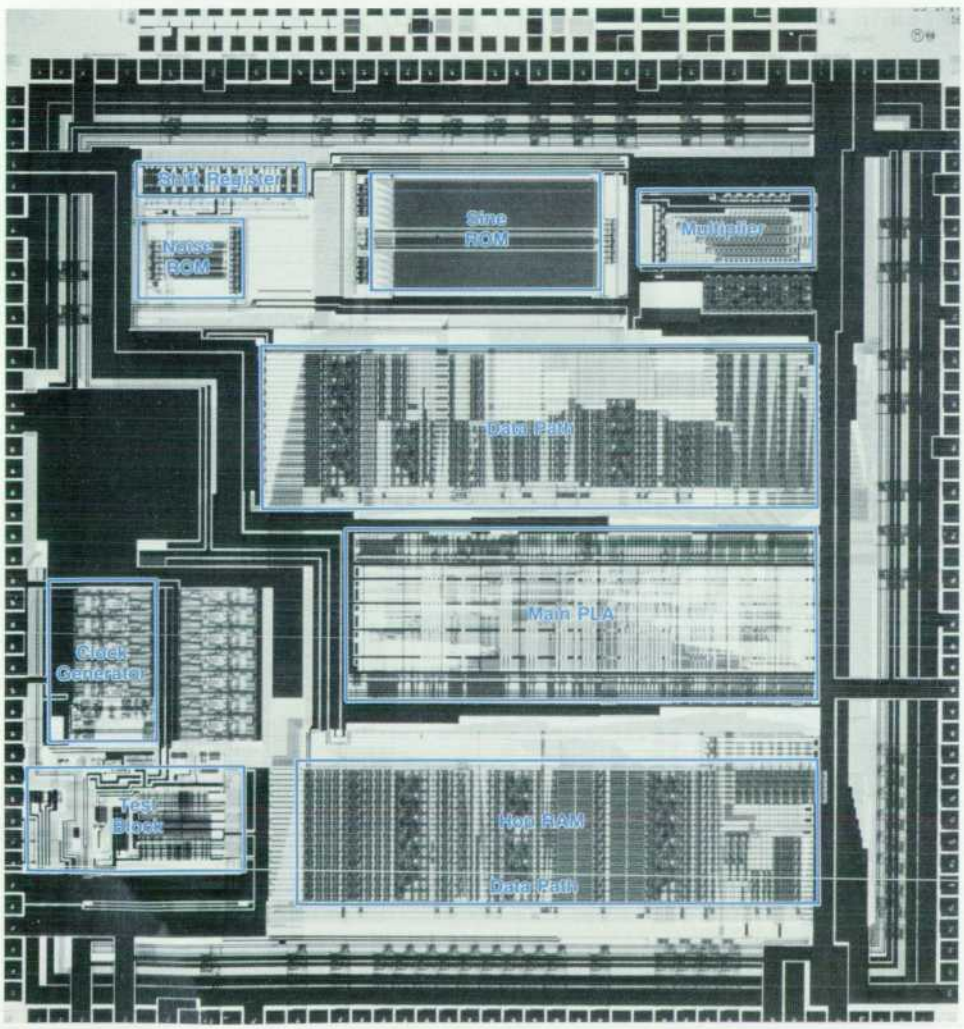


Fig. 2. *A photomicrograph of the DWSIC chip with the blocks labeled.*

multiplier to provide amplitude modulation, and a noise generation circuit. Fig. 2 identifies the major circuit blocks on a photomicrograph of the chip.

The third major feature of HBD is the scan-path test methodology. Key storage locations within the chip are connected by one or more serial shift scan paths. These scan paths are used only to test the chip and play no role in the chip's normal operation. The scan paths terminate at a diagnostic interface port (DIP), which consists of four I/O pads and a controller (see Fig. 3). An HP-developed IC tester is used to stimulate the four DIP pads. While the chip is halted, the tester loads every scanned node in the chip with an initial input value. The chip is then stepped one or more times at the full operating frequency and the scan path is shifted out. The IC tester checks to see if the scanned-out values are correct. With this test methodology every block in an HBD design is controllable and observable, and therefore fully testable.

Design Challenges

Design challenges introduced by the data flow architecture were achieving a regular space-efficient data path layout using existing NMOS-III data path library cells, and ensuring correct operation and testability of this relatively complex structure. The layout considerations required implementing the parallel pipe stages as a series of pipeline segments, with signals being bused relatively long distances over unrelated segments. The number of pass-through buses and the drive capabilities of the library cells are limited. Therefore, numerous design iterations were required before these limits were satisfied, and the flexibility of the library was taxed. Because of the complexity of the data path layout, a Pascal program was used to place the component blocks. This program takes a textual description as input and creates a labeled schematic and artwork representation (see Fig 4). This greatly facilitated modification of the design and helped ensure consistency between the schematic and artwork representations.

In addition to correct functionality, other important design parameters were power dissipation, packaged part cost, board space requirements, and reliability. All of these affect final instrument cost. In particular, power below about 2.7 watts would allow simple, inexpensive cooling techniques. One strategy for achieving this was to use lower-power circuits wherever speed was not critical, such as in the control registers. Packaged part cost was reduced by employing a printed circuit pin-grid array package, which also helped the cooling problem by providing a low thermal resistance. A further advantage of the pin-grid array package is efficient use of board space compared with dual in-line packages. Reliability is achieved in the NMOS-III process through the use of functional burn-in, continous monitoring of field failures, and process control.

Chip Modeling and Verification

The DWSIC has a complex architecture. Since the goal for any chip design is to be correct the first time, it is necessary to simulate the design extensively before casting it into silicon. The simulation methodology used for the DWSIC was to simulate the logical design behaviorally, extract test vectors from the simulation, and run these extracted vectors on the transistors from the layout using a switch-level simulator.

For behavioral simulation, a chip is modeled algorithmically at a high level to ensure that the architecture is correct. For each component block in the hierarchy, a behavioral description is written. This description algorithmically relates the inputs of the block to its outputs (see Fig. 5). Structural descriptions are used to describe how the behavioral blocks are connected. When the model is complete and input vectors are written, it is possible to simulate the operation of the chip.

Running the behavioral simulation causes the input vectors to produce streams of data at the model's outputs. These output streams are checked for numerical accuracy and converted into output assertion vectors for an HP developed switch-level simulator. A complete vector set consisting of inputs and assertions is run on the switch-level simulator. The switch-level simulator models each MOS transistor as a simple voltage-controlled switch and pro-

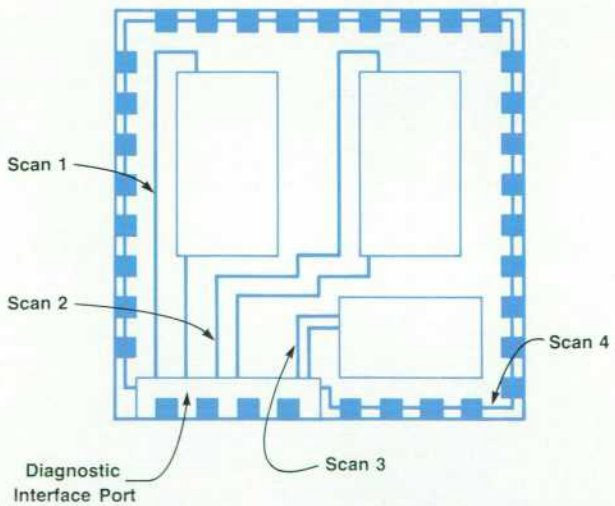


Fig. 3. *Scan paths on a chip terminating at a diagnostic interface port (DIP).*

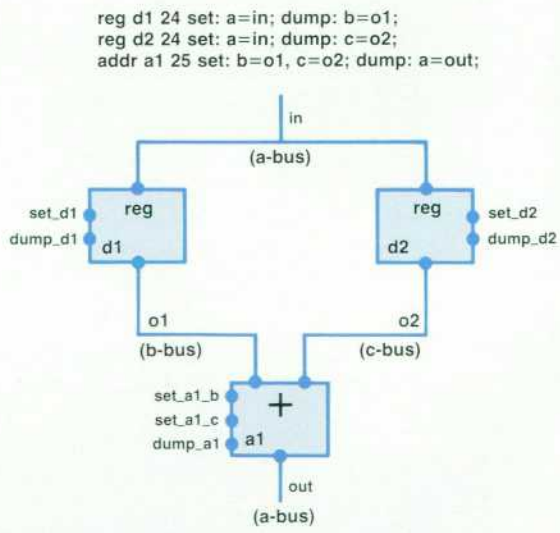
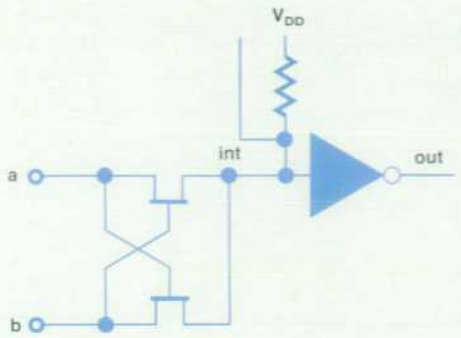


Fig. 4. *Textual description of data path layout and artwork representation.*




```
procedure xor (var a, b, out: integer);
var
  int: integer;
begin
  if (a=1) then int:=b
  else if (b=1) then int:=a;
  out:=not(int);
end;
```

Fig. 5. *Behavioral description for an exclusive-OR (XOR) component.*

vides a logical simulation for a chip at the transistor level. The goal of the vector-capture and simulation methodology is to verify that the transistor-level implementation of the chip exactly reproduces the function of the behavioral model.

New Circuits

About one third of the DWSIC design time was spent on new custom circuits to implement functions that had not been done before in NMOS-III. Two of the more interesting new circuits were an 11-by-12-bit integer multiplier and a 4K-by-11-bit ROM.

The design goal for the multiplier was to minimize power, design time, and circuit area. The system requires a product from the multiplier at every state, making propagation delay important also. Three multiplier architectures were considered to achieve these goals. The attributes of these architectures are summarized in Table II. All three approaches could have met the system speed requirements, but the Booth approach provided a significant speed margin and best satisfied the design goals. The implementation of the Booth approach is similar to that found in HP math chips.[2]

Table II

Attributes of Multiplier Architectures

	Full Array	Wallace Tree	Booth Encoded
Number of FETs	2900	3200	2600
Power (mW)	175	190	155
Area (mm^2)	0.60	0.70	0.55
Speed (number of gate delays)	25	15	16
Layout Complexity	low	high	moderate

Series transistor ROMs have been done in NMOS-III before, but the DWSIC represents the first parallel transistor ROM implementation. A series transistor ROM consists of a chain of transistors connected drain to source. This ROM implementation results in very small bit size but requires more complex sense amplifiers because of signal degradation in the chain. A parallel transistor ROM, where all the column transistors are connected with their sources and drains in parallel, was chosen for the DWSIC because design and access time were more important than ROM bit size. A parallel ROM has faster access time because the signals propagate through a single transistor instead of a transistor chain, and is simpler to design because elaborate sense amplifiers are not necessary since the signals don't degrade as much.

Summary

A custom digital VLSI circuit has been designed to increase the performance/price ratio and improve flexibility for a complex waveform generation product. The project proved that the NMOS-III process, design methodology, and circuit library created for the design of high-speed computer chips can be successfully employed in instrument applications.

The DWSIC project started in July 1985 and required 54 engineering-months over a 13-month period to complete. The resulting chip contains over 99,000 FETs and the first silicon was completely functional.

Acknowledgments

The authors wish to recognize the other members of the chip design team: James Stewart, Larry Arnstein, Tom Walley, Don Novy, and Marty Wilson, along with the IC fabrication and assembly groups.

References

1. J.M. Mikkelson, et al, "NMOS-III Process Technology," *Hewlett-Packard Journal*, Vol. 34, no. 8, August 1983, pp. 27-30.
2. W. McAllister and J. Carlson, "Floating-Point Chip Set Speeds Real-Time Computer Operation," *Hewlett-Packard Journal*, Vol. 35, no. 2, February 1984, pp. 17-23.

Analog Output System Design for a Multifunction Synthesizer

The analog output system for the HP 8904A takes the 12-bit data stream from the digital waveform synthesis IC and converts it to an analog signal with excellent frequency response and low distortion.

by Thomas M. Higgins, Jr.

THE WAVEFORMS GENERATED in the digital waveform synthesis IC (DWSIC) in the HP 8904A Multifunction Synthesizer exist only as 12-bit binary numbers at the output of the IC. The output system converts these binary numbers into the desired analog signals.

The first step in this conversion process is the digital-to-analog converter (DAC). Each binary number output by the DWSIC represents the instantaneous voltage of the waveform being generated. Each number represents a sample of the waveform, and the number of samples per second is determined by the clock that drives the DWSIC. The DAC takes each binary number and converts it to the voltage corresponding to that particular sample.

The output of the DAC is a stair-step approximation of the analog signal. If we look at a synthesized sine wave in the frequency domain (see Fig. 1a), we see the desired output signal, along with some high-frequency alias signals characteristic of sampled waveforms. The magnitude of each component is determined by the familiar (sin x)/x envelope. If the maximum frequency in the digitized signal is less than the Nyquist rate, or one half of the sample rate, then these alias signals can be filtered off, removing the stair-step shape. In the real world, ideal low-pass filters cannot be realized, so the required sample rate is somewhat higher than the Nyquist rate.

Because of the (sin x)/x envelope, the frequency response of the output signal is not flat. To flatten the response of the analog signal, the (sin x)/x roll-off must be compensated for, either by digital signal processing techniques while the signal is still in numerical form, or by using a filter with some peaking.

Real-world DACs introduce spurious signals not present in an ideal sampled waveform. The accuracy of a DAC's output is not specified until a given settling time has passed. During this settling time there can be glitches, overshoot, and ringing. In general, these transitions are not uniform, but depend on what two values the DAC is switching between. This adds a noise-like component to the output spectrum, as shown in Fig. 1b. Since much of this noise is present in the same band as the signal, it cannot be filtered out. To eliminate this noise, the output of the DAC can be sampled after it has settled, and the transitions are then determined by the sample-and-hold circuitry. A properly designed sample-and-hold circuit can greatly enhance the signal-to-noise ratio of a digitally synthesized signal.

The block diagram in Fig. 2 shows how these signal conditioning functions are performed by the output board in the HP 8904A. The 12-bit data from the DWSIC is first reclocked by the master clock into the input data latches. This provides proper timing between the DAC and the sampler, and removes any timing skew between the twelve data lines. The DAC converts this data to an analog signal, and the sampler samples the DAC output to remove noise caused by uneven DAC output transitions. The sampler output is buffered to drive one of the two anti-aliasing filters and then another amplifier boosts the voltage of the

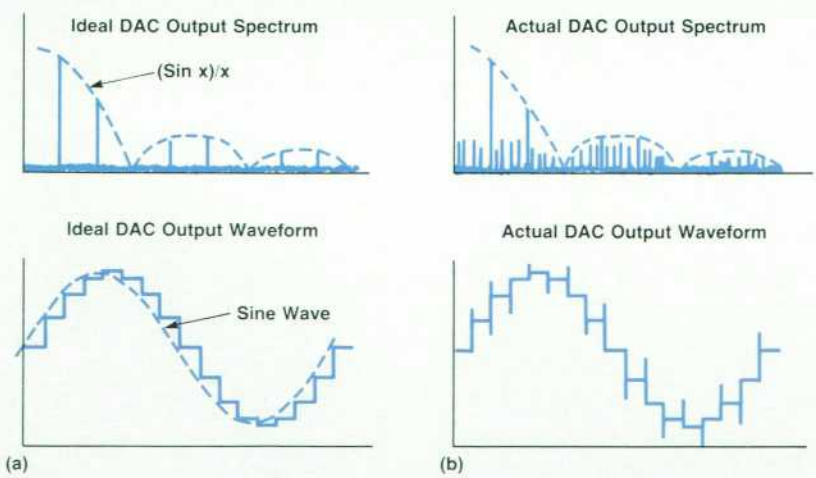


Fig. 1. *(a) Ideal DAC output spectrum and ideal DAC output waveform. (b) Actual DAC output spectrum and actual DAC output waveform.*

signal up to the desired ±10 volts. (Sin x)/x compensation removes the roll-off produced by the sampling, and a step attenuator allows control of signal level without sacrificing dynamic range. Finally, the floating output amplifier provides the current drive necessary for a 50-ohm source, and provides a floating output to avoid system ground-loop problems.

Sampler

As described earlier, the function of the sampler is to sample the output of the DAC after it has settled to its specified accuracy and hold this voltage until the next sample. This results in lowering the noise produced by inconsistent transitions in the DAC. The sample-and-hold circuit holds the previously sampled voltage for the first half cycle while the DAC settles, and for the second half cycle it tracks the DAC output. This 50% duty cycle provides maximum settling time for both the DAC and the hold circuitry.

The sampler driver is a limiting amplifier that provides a differential, high-current drive signal to the diode bridge. A transformer at the output of the sampler driver allows the sampling bridge to float and improves the balance of the drive signal. The combination of the differential driver and the transformer produces a very well-balanced, isolated drive signal and prevents high levels of the drive signal from getting onto the sampled signal. If high levels of the drive signal were to get through the sampler, the sampler buffer would be overloaded, causing severe distortion. The small amount of drive signal that does get through the sampler is filtered out at the output of the sampler buffer.

The sampler buffer is the most critical circuit on the output board. It must have an extremely high input impedance so as not to drain the 15-pF hold capacitor, and must have very low distortion for audio signals in the presence of high-level, high-frequency alias signals. Any dc offsets are magnified by ten in the following amplifier stages, so the buffer must have a very low offset. In addition, it must have an extremely flat frequency response and be able to drive the low and nonresistive impedance of the filters that follow it.

To achieve the required high input impedance with good high-frequency performance, the input stage of the sampler buffer is a single-ended cascode FET amplifier. The input stage drives a common emitter amplifier, which provides more gain. The combination of the two inverting stages provides a very large noninverting gain. The output of the common emitter stage is buffered by a monolithic emitter follower output stage. An unusual feature of this amplifier is the way negative feedback is applied (see Fig. 3a). The output is connected to the source of Q1 and the gate of Q2. This node is grounded in a typical cascode amplifier, but if the node is driven by a low-impedance source, the circuit will amplify the difference between the input signal at the gate of Q1 and the signal at this node—the circuit then functions as a differential amplifier. The input signal drives the noninverting input, and the output provides feedback to the inverting input.

One problem with this amplifier is a large dc offset, and to make matters worse, the offset has a large variation over temperature and from part to part. To solve this problem, a dc correction loop is used, which provides the amplifier with the dc performance of a good monolithic operational amplifier. This circuit is shown in Fig. 3b. An operational amplifier compares the output with the input inside a 0.1-Hz bandwidth and biases the FET input stage to drive the difference to zero. The offset of the entire amplifier is then set by the input offset of the monolithic operational amplifier. The input signal sense line and the gate drive for the input FET are bootstrapped to isolate the dc correction loop at high frequencies. This keeps the input impedance high at higher frequencies.

Anti-Aliasing Filters

The HP 8904A Multifunction Synthesizer offers two anti-aliasing filters: a 7-pole elliptic filter and a Gaussian filter. The elliptic filter provides a good approximation of an ideal filter because it has excellent in-band flatness and out-of-band rejection characteristics. It is used when maximum frequency-domain accuracy is required. The filter has less than 0.002 dB passband ripple, and uses inductors constructed of ferrite wound with Litz wire to achieve a Q of better than 300. The filter rolls off −0.06 dB by 600 kHz because of the finite Q, so an equalizer is used to improve the flatness to −0.01 dB at 600 kHz. The phase response of the filter is excellent, with less than 0.1 degree monotonic departure from linear by 100 kHz. This is very important for FM stereo (see HP 8904A applications article on page 73), since phase and amplitude accuracy determine the amount of stereo separation that can be achieved.

Since the highest allowed output frequency is 600 kHz and the clock frequency is 1.68 MHz, the lowest alias frequency is 1.08 MHz. Because of the (sin x)/x response, this is also the largest-amplitude alias signal—it is only 9 dB

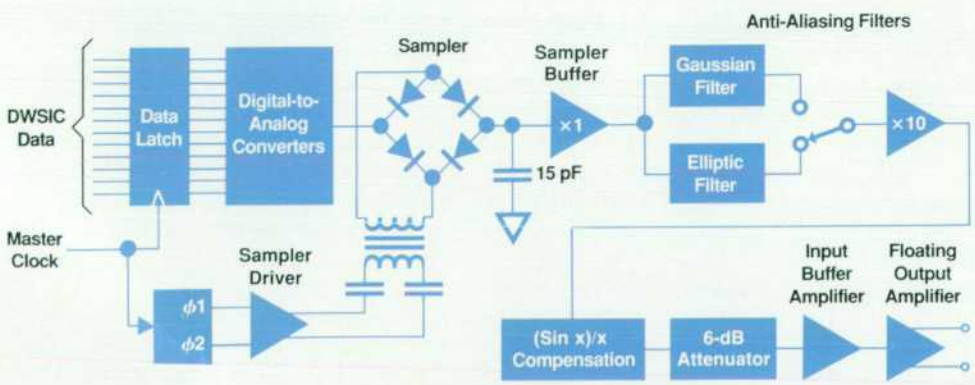


Fig. 2. *Output board block diagram.*

A Generating a Phase-Locked Binary Reference Frequency

The HP 8904A's precise control of phase is more useful if it can be phase-locked to other instruments. To accomplish this, an accumulator-based reference loop is included in the design. This loop performs the fractional division necessary to lock the master clock to an internal or external 10-MHz reference. Since the phase accumulator is 24 bits wide and the desired frequency resolution for each channel is 0.1 Hz, the clock for each channel must be 2^{24}(0.1 Hz), or 1.6772160 MHz. There are four channels, so this increases the clock frequency by a factor of four to 6.7108864 MHz. Finally, the DWSIC requires a 2× clock, so the master clock frequency to the DWSIC becomes 13.4217728 MHz (see Fig. 1).

Design goals were to lock to a 10-MHz reference with a frequency accuracy of 100 ppm or less, have spurs less than −43 dB, require no tweaks, and be very low-cost. Spurs on the main clock at −43 dB are divided down to −70 dB from a 600-kHz output signal, −85 dB from a 100-kHz signal, and −100 dB from a 20-kHz signal.

Since this loop synthesizes a single reference frequency, the voltage-controlled crystal oscillator (VCXO) needs enough tuning range to accommodate a ±100-ppm reference, and enough margin to allow for temperature drift and component variations. It also needs to be highly stable and very low-noise so that a narrow loop bandwidth of about 50 Hz can be used to filter spurs produced by the fractional division. A voltage-controlled crystal oscillator is ideal for this purpose. The oscillator designed for the reference loop is a Pierce oscillator with a tuning range of typically about ±300 ppm. With this much extra tuning range, the loop can compensate for component or temperature variations and still achieve the goal of locking to a reference with a frequency error of less than 100 ppm.

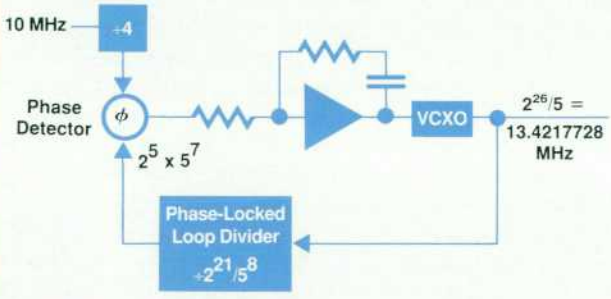


Fig. 1. *Reference loop.*

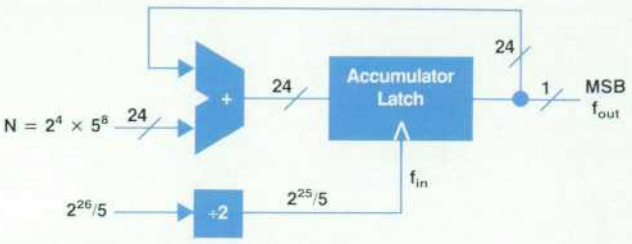


Fig. 2. *Phase-locked loop divider.*

To synthesize a frequency with 0.1-Hz resolution using a low-cost fractional-N phase-locked loop, a 24-bit accumulator was added to the DWSIC to perform the fractional division. This accumulator is separate from the main phase accumulator. Every clock cycle, a fixed number N is added to the contents of the phase-locked loop accumulator. The frequency spectrum of the most significant bit will consist of a large number of harmonics of the fundamental frequency of the accumulator, which is equal to the clock frequency divided by the accumulator size, which in this case is 2^{24}. The Nth harmonic will be the largest component, and by using a narrow loop bandwidth, the other harmonics can be filtered out. The output frequency is then:

$$f_{out} = f_{in} \times N/2^{24}$$

The clock input to the DWSIC is divided by two before driving the accumulator. In this application, $f_{in} = 2^{25}/5 = 6.7108864$ MHz, and the number being accumulated is $N = 5^8 \times 2^4$. Plugging these numbers into the above equation, we see that the output frequency is $2^5 \times 5^7$, or exactly 2.5 MHz. The output of the divider is a frequency that can be obtained simply by dividing the 10-MHz reference by 4. These two signals become the inputs to the phase detector.

The binary representation of the accumulator number N is 10111110101111000010000. Since there are four trailing zeros, the lower four bits of the accumulator are never exercized, and the effective size of the accumulator is 20 bits. As described earlier, there will be spurs spaced at multiples of the fundamental frequency of the accumulator:

$$f_{Spur} = f_{in}/2^{20} = 6.4 \text{ Hz}$$

Some of these spurs fall inside the 50-Hz loop bandwidth, but these close-in spurs are low enough that they are not a problem.

below the 600-kHz output signal. Therefore this is the most difficult alias signal to reject. Elliptic filters have stop-band zeros at the resonant frequencies of the parallel LC pairs, so the lowest of the three nulls for this filter is set at 1.08 MHz for maximum rejection of this worst-case alias signal. The rest of the stop band has at least 62 dB of attenuation.

Despite the accuracy of the elliptic filter, waveforms that contain large transients will overshoot. When using the elliptic filter, a sine wave will be accurately reproduced within the limits of the 12-bit system. However, a square wave would ring because the higher-order harmonics would be filtered off. The time-domain response is an accurate representation of a square wave passed through a finite bandwidth, but if the signal is being used as a digital source, the overshoot may be undesirable. A Gaussian filter has a Gaussian impulse response; therefore, the response to a step input has no overshoot. Technically, the time-domain response is not as accurate, but the output looks better for signals with large transients and is more useful in some applications. Since the Gaussian filter rolls off more slowly than the elliptic filter, the cut-off frequency of the Gaussian filter is set at 230 kHz to obtain adequate rejection of aliased signals.

The (sin x)/x compensation is performed by an underdamped two-pole low-pass filter at the output of the main amplifier. The filter provides 1.9 dB of peaking at 600 kHz,

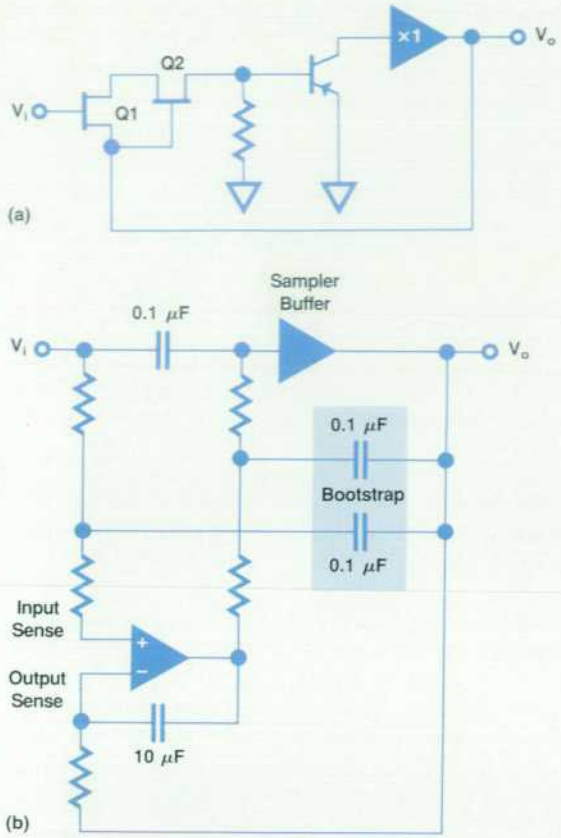


Fig. 3. *(a) Sampler buffer. (b) Dc correction loop.*

canceling the effect of the (sin x)/x roll-off. The component values were computer optimized to fit the peaked frequency response of the filter to the inverse of the (sin x)/x function from dc to 600 kHz. The main amplifier isolates this filter from the anti-aliasing filters.

Amplitude Scaling

The full-scale amplitude at the output of the filters is 1 volt peak. Since the desired output level is 5 volts peak into 50 ohms, 10 volts peak into an open circuit is required. The main amplifier is the main gain block in the instrument and provides the required gain of 10. The critical specifications for this amplifier are low distortion and good flatness. A discrete single-ended amplifier with two stages of gain provides a slew rate in excess of 300 volts per microsecond, a gain-bandwidth product of over 1 GHz, and an integrator-type roll-off for frequencies below 600 kHz. Above 1 MHz the gain-bandwidth product is reduced to about 150 MHz for stability. A 1-GHz gain-bandwidth product is necessary to provide 60 dB of loop gain at 100 kHz for an amplifier with a gain of 10. This corresponds to a maximum gain error of 0.1%. However, the integrator-type roll-off causes an additional 90 degrees of phase shift, putting the error signal in quadrature with the input signal. When the error signal and input signal are added vectorially, the gain error is much less than 0.1%. In practice, the flatness of the main amplifier is typically better than ±0.01%, or about ±0.001 dB, to 100 kHz. Good dc performance is assured by using this discrete amplifier as the high-frequency block of a composite amplifier, with a monolithic operational amplifier biasing the input stage. The crossover between the high-frequency and low-frequency blocks is at a frequency where there is still sufficient loop gain to swamp out the phase bump at the crossover. Unlike the sampling buffer, the dc correction is inside the feedback loop.

The multiplier in the DWSIC can be used to scale the signal amplitude, but each factor of two reduction in signal level reduces the effective resolution by one bit. This would cause low-level signals to have poor distortion and signal-to-noise ratio. To avoid this problem, only the upper 6 dB of scaling in the multiplier is used—this provides the fine amplitude control, while ensuring that the signal has at least 11 bits of resolution. An additional 96 dB of attenuation is available in 6-dB steps from the step attenuator.

Floating Output Amplifier

The input buffer amplifier provides a 1-kilohm load to match the impedance of the 6-dB step attenuator and a voltage drive for the floating output amplifier. The floating output amplifier converts this single-ended voltage drive to a differential, floating output using a combination of positive and negative feedback.[1] The amplifier topology is identical to the output amplifier in the HP 8903B. However, the two amplifier blocks were changed to meet the HP 8904A's requirements. The HP 8904A can produce a 10-volt-peak 600-kHz sine wave, while the maximum for the HP 8903B is 100 kHz at 6 volts rms. The required slew rate is almost ten times as high, so new higher-speed op amps are used. In addition, the HP 8904A can supply dc voltages, which more than doubles the power dissipation in the output transistors, so current limiting has been added.

Acknowledgments

Rob Dickerson and Fred Ives defined the output board. Rob Dickerson designed the majority of the output board circuitry, including the DAC and sample-and-hold circuits. Fred Ives designed the anti-aliasing filters and provided overall guidance and support throughout the project.

References

1. George D. Pontis, "Floating a Source Output," *Hewlett-Packard Journal*, Vol. 31, no. 8, August 1980, pp. 12-13.

Firmware Design for a Multiple-Mode Instrument

The firmware architecture of the HP 8904A Multifunction Synthesizer is designed to handle the existing operating modes efficiently and to facilitate evolutionary changes.

By Mark D. Talbot

THE HP 8904A MULTIFUNCTION SYNTHESIZER hardware offers many different operating modes, depending on the requirements of the user. The HP 8904A firmware was designed and implemented with the goal of efficiently handling the existing operating modes and allowing for changes within these modes and the addition of new modes. Three major firmware modules provide this required flexibility: the numeric data parser, the command parsing structure, and the interrupt handling scheme. Each of these modules provides hooks to customize or add to the HP 8904A firmware with little or no changes to the existing firmware.

Numeric Data Parser

The firmware of most HP signal generating instruments has the luxury of fixed data types, string lengths, and display window positions. In contrast, the HP 8904A, with its multiple modes, requires custom configuration of the firmware to handle the widely varying data types, string lengths, terminators, and display positions. The numeric data parser allows custom configuration of signaling sequence and mode command parameters (e.g., 10 kHz, AM, FM, etc.). A parser, as used in this article, refers to firmware functions written in the C language that check data or commands for their syntactic and semantic correctness.

The numeric data parser is a library function used by the front-panel and HP-IB command parsers described in the next section. Its role in parsing input data is illustrated in Fig. 1. The instrument is in the channel configuration mode and the user enters a frequency of 120 Hz from the front panel. When the **FREQ** key is pressed, the channel configuration mode parser calls the function np_init() to initialize the data structure used by the numeric data parser in subsequent key presses associated with setting frequency. When the user enters 120 Hz the string is sent to the numeric data parser. The numeric data parser uses the parameters established during initialization to parse the string.

Fig. 1b enumerates the five parameters that can be passed to the numeric parser initializer. The firmware designer builds the parsing rules for the numeric data parser by filling in the parameters shown in Fig. 1b. This collection of parameters is then passed to the np_init() function which sets up the data structure for the numeric data parser. For the example above, data_type = Real, and term_type = Frequency. The other parameters are used to position the value on the front-panel display.

Command Parsing Structure

The HP 8904A command parsing structure, which is shown in Fig. 2, allows the implementation of customized command modes or changes to existing modes without affecting the other modules in the parsing structure. Besides the numeric data parsers, there are four other parsers in the HP 8904A firmware. Each parser is tailored to a specific category of commands. The following is a partial list of HP 8904A commands and their associated parsers.

Commands or Data	Parser
Front-Panel Keystrokes	Keys-to-Command Parser
HP-IB Tokens	HP-IB Token-to-Command Parser
Generic Commands	Initial Command Parser
NEXT	
LAST	
MAIN	
LOCAL	
RESET	
ADRS FLOAT	Mini-Parsers
FILTER OUTPUT	
RECALL SPECIAL	
Mode-Specific Commands	Mode Command Parsers
AMPTD DESTN	
FREQ INCR SET	
PHASE SAVE	
WAVE FORM	
Mode-Specific Softkey Definitions (f_1...f_4)	
Main Level Commands	Top Level Parser
Mode Selection Keys (f_1...f_4)	

Front Panel and HP-IB Input Parsers. All inputs to the command parsers enter through the front-panel keys or the HP-IB. The keys-to-command parser and the token-to-command command parser shown in Fig. 2 handle front-panel input and HP-IB input, respectively. There is a numeric data parser contained in each of these parsers. These parsers perform syntax checks on the command or parameter input and then set up the commands for processing by the other parsers in the command parsing structure. Each command or parameter passing through the input parsers concludes with one of the following actions taken:

- The input is ignored because of an error or some out-of-sequence input.
- The input is not converted but stored as part of a yet to be completed input string.
- The input is successfuly converted to a command data item.

Initial Command Parser. After a command has been parsed by the front-panel and HP-IB bus parsers, a data item representing the command is passed to the initial command parser. Depending upon the type of command, the initial command parser performs one the following actions:

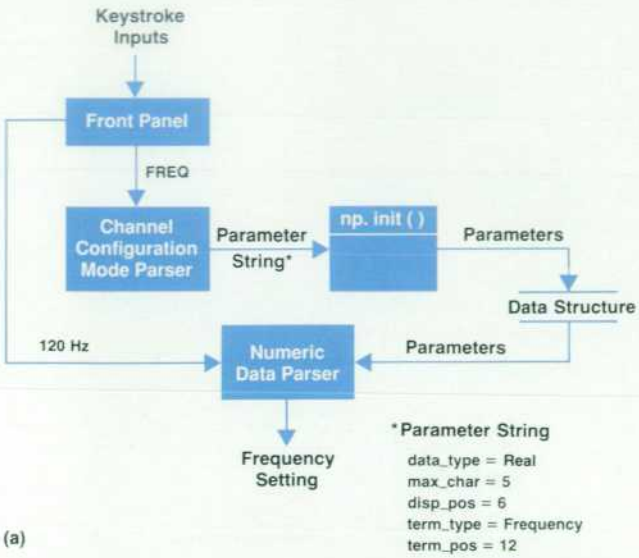
- If the command is one of the generic commands the initial command parser is responsible for handling, it processses the command in total or in part.
- If the command is one of the generic commands handled by a mini-parser and the mini-parser is not active it calls an initialization function to set a pointer to the command's mini-parser, which results in making the specific mini-parser active. Setting the pointer is illustrated by the switch shown in Fig. 2.
- If a mini-parser is active it is invoked.
- If the command is not found in the list of generic commands or a command needs further action, the active mode command parser is invoked.

A valid active mode command parser will always be present in addition to the initial command parser. This will be either the top level command parser (tlp_prs()) for main level commands, or a mode command parser for one of the mode-specific commands.

Mode Command Parsers. The mode command parsing section of the command parsing structure is divided into mode installation functions and the mode command parsers. The installation functions initialize and activate the mode command parsers, including setting a pointer to the selected mode command parser.

During a normal power-up sequence, the execution of the MAIN command, or an instrument PRESET, the install_tlp() (install top level command parser) function is performed. This function sets a pointer to the top level command parser function tlp_prs(). This causes the flow of commands not handled by the initial command parser to be directed to the top level command parser. The top level command parser allows the user to scroll through the available instrument command modes. When a new mode is selected, its installation function (install_modei(), i=1,2...N) is executed. The mode installation function configures the hardware for the selected mode, sets a pointer to the function to call when the mode is exited, and sets the pointer to the command's mode comand parser, which causes command flow to be directed to the selected mode command parser. Hardware initialization, display update, and softkey configuration are also performed.

The responsibility of the mode command parser is to test a command to see if the command is within its repertoire, and if so, to execute the command; otherwise, it is reported to the operator as an error. For example, the commands



(a)

Function Call

np_init(data_type, max_char, disp_pos, term_type, term_pos)

data_type = One of the following:

Type	Allowable Entry
Binary	0-1
Octal	0-7
Decimal	0-9
Hexidecimal	0-9, A-F
Phone	0-9, A-D, #, and *
Real	0-9, "." and "-"
P_Real	0-9, and "."
Destination	Out1, Out2, Am, Fm, Om, Dsb, Pulse, and Off
Waveform	Sine, Ramp, Trngl, Dc, Noise, and Square
On, Off	On, Off

max_char = The number of data keys that will be accepted, 0 to 10.

disp_pos = Display position for incoming data keys, 1 to 80. (2×40-Character LCD Display)

term_type = One of the following:

Type	Acceptable Characters
None	N/A as in Waveform
End of Buffer	Input = max_char
Enter Key	Enter
Angle	Deg, Rad
Volts	V, mV, or uV
Time	Sec, mSec
Frequency	kHz, Hz
Amount	%

term_pos = Display position for the data terminator, 1 - 80. (2×40 Character LCD Display)

(b)

Fig. 1. *(a) The role of the numeric data parser in the HP 8904A firmware parsing architecture. (b) Parameters passed to the numeric data parser initialization function* np_init().

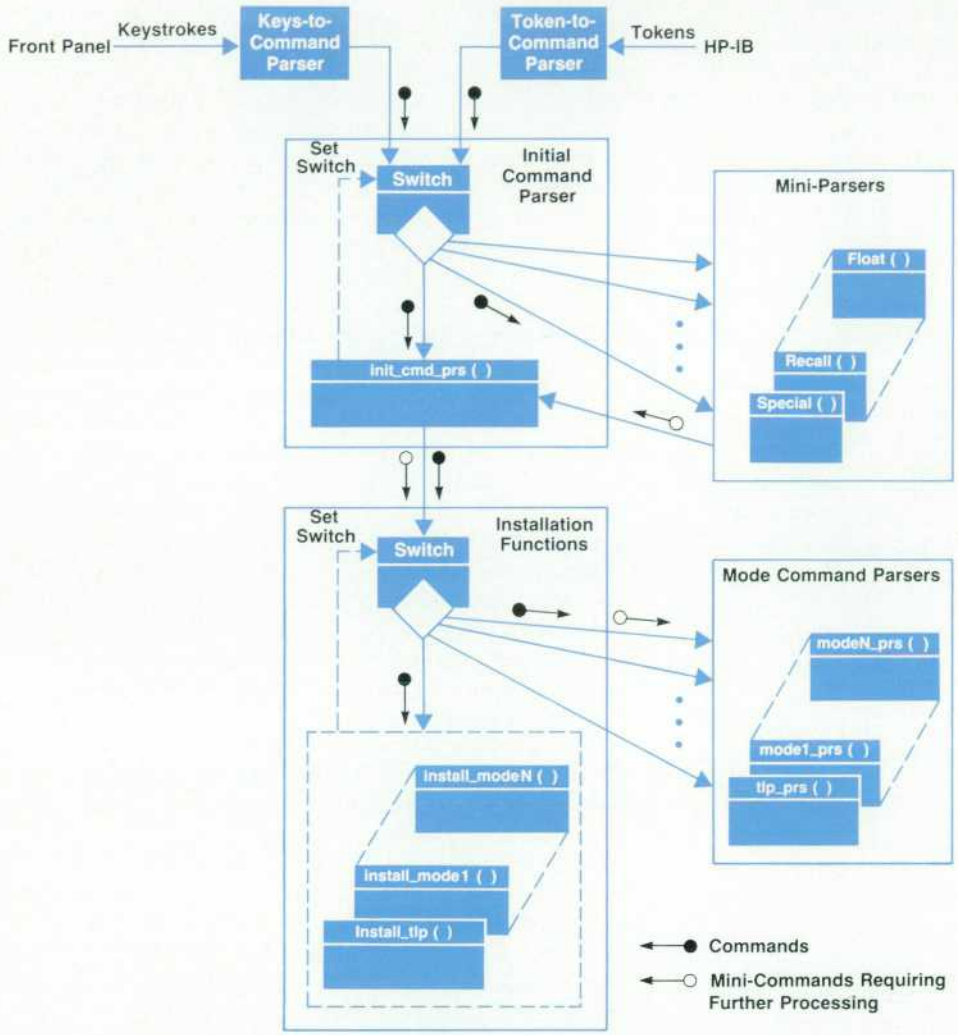


Fig. 2. *Command parsing structure for the HP 8904A.*

AMPTD (amplitude) and FREQ (frequency) would be processed successfully in the channel configuration mode, but the FREQ command in dual-tone multifrequency mode would cause an error. The same mode-specific command can be valid for several different modes but perform different operations from one mode to the next. For example, in the channel configuration mode the FREQ command sets the output frequency, whereas in the tone sequence mode the FREQ command sets the frequency for the selected tone. A mode exit function is called whenever the presently active mode is exited. This is done whenever a softkey function EXIT is selected, or when the MAIN or PRESET keys are pressed. The mode exit function will clean up its working environment, leaving it clean for the next mode to use. It also forces the top level command parser to be installed.

Mini-Parsers. While the instrument is in one of the instrument modes or at the main selection level, it is sometimes necessary to perform some other function momentarily without disrupting the present operation, and then resume the present operation when the momentary function is finished. Examples of these momentary functions include SAVE, RECALL, changing the HP-IB address, or turning off an output. These types of operations are performed by the mini-parsers. The mini-parser is a command parser that has a very limited functionality and is placed in series with the command path for a short period of time.

When a command to perform a function that requires the use of a mini-parser is received, a call to that mini-parser's initialization function is made. This function in-

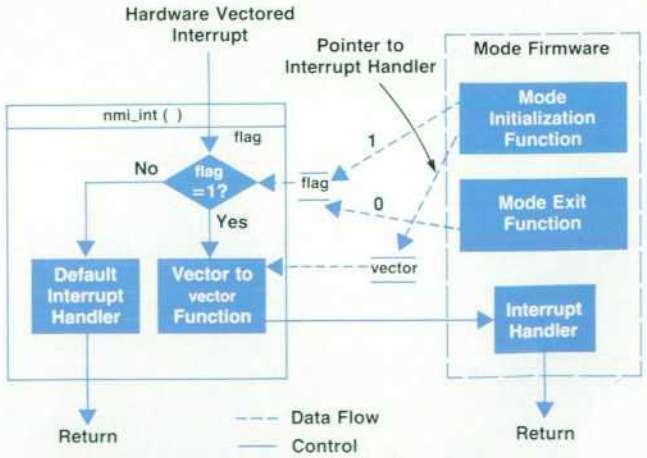


Fig. 3. *HP 8904A interrupt handling structure.*

stalls the specific mini-parser by setting a pointer to a mini-parser function. The mini-parser will execute the present command or pass it on for further processing by other parsers. When the mini-parser is finished or chooses to abort the operation because an invalid command was received, it will remove itself from active status so that subsequent commands are not passed to it.

Vectored Interrupts

For the HP 8904A real-time hardware, interrupts are generated by the timer IC and and the digital waveform synthesis IC control logic. During an interrupt a microprocessor typically directs program flow (vectors) to locations stored in fixed addresses in its program ROM. When the HP 8904A firmware was written the requirements for handling interrupts by all present and future modes were unknown. This was inconsistent with the fixed interrupt handler methodology. The solution was to develop an environment that allows each operating mode to install its own individual interrupt handlers. Fig. 3 shows the HP 8904A interrupt handling structure.

Depending on when and how the interrupt is to be used, the interrupt handler is installed by the mode initialization function or some other function before enabling the interrupt. This is done by setting the variable flag to one and placing a pointer to the desired interrupt handler function into the variable vector. When a hardware interrupt occurs, such as a nonmaskable interrupt (NMI), the microprocessor will vector to the nmi_int() function. The status of flag is checked to see if an alternate handler is installed. If the flag is set, the function pointed to by vector is called and if it is not set, default interrupt routines clear the interrupt. This latter situation occurs only if the interrupt was unintentionally or spuriously generated. The special interrupt handler function is removed by setting flag to zero. This is done by the mode command parser when it is finished, or when the mode exit cleanup function is invoked, or after PRESET or instrument power-up.

Acknowledgments

The author would like to thank Troy Beukema who wrote all of the tone, DTMF, and digital signaling firmware for the HP 8904A, Bill Dickerson who developed the overall instrument methodology for the channel configuration mode firmware, and Ken Thompson and Bob Rands who defined the requirements for the signaling modes.

Multifunction Synthesizer Applications

Application areas for the HP 8904A Multifunction Synthesizer include telecommunications, navigation, mobile radio communications, consumer electronics, sonar, and electromechanical systems.

by Kenneth S. Thompson

THE FLEXIBILITY AND ACCURACY afforded by the architecture of the digital waveform synthesis IC (DWSIC, see article page 57) make the HP 8904A Multifunction Synthesizer suited for a broad range of applications. These applications areas include telecommunications, navigation, mobile radio communications, consumer electronics, sonar, and electromechanical systems. Typically the conflicting requirements of these applications have been met by many specialized sources optimized for specific tasks or by expensive high-performance sources. The HP 8904A can compete with these application-specific sources and in many cases exceed their performance at a lower cost.

The five operational modes of the HP 8904A are:

- Channel configuration mode
- Tone sequence mode
- Dual-tone multifrequency (DTMF) sequence mode
- Digital sequence mode
- Hop RAM mode.

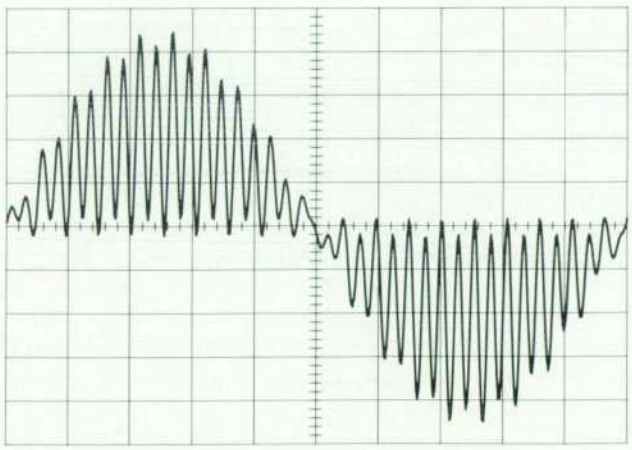


Fig. 1. *An FM stereo composite test signal generated by the HP 8904A Multifunction Synthesizer.*

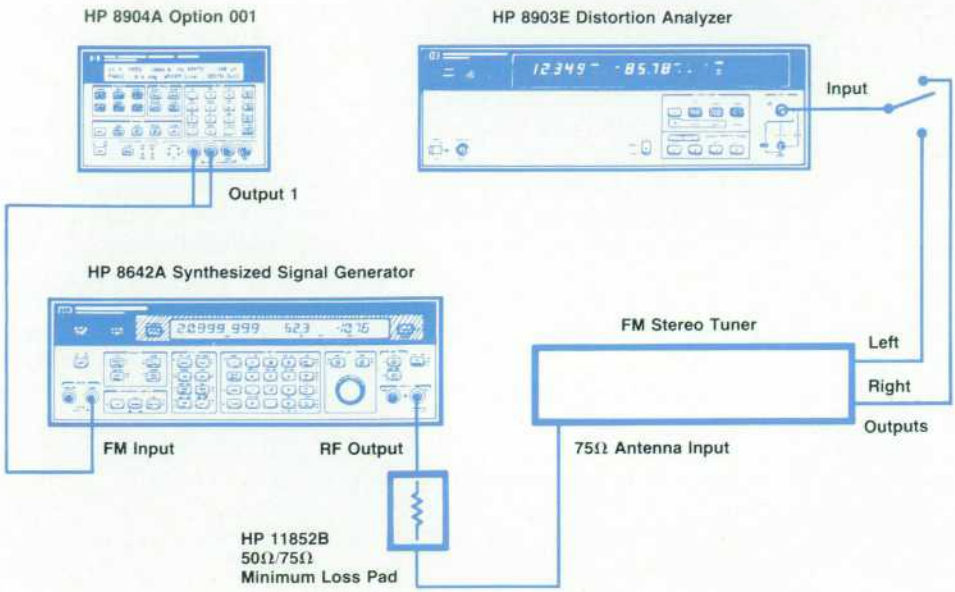


Fig. 2. *Test setup for performance testing of an FM stereo receiver using the HP 8904A.*

The channel configuration mode can be used to synthesize complex waveforms by using various combinations of the four synthesizers available in this mode (with Option 001). The four synthesizers can be freely mixed in various combinations by using the digital summing or modulation capabilities built into the digital waveform synthesis IC. Waveforms found in navigation, sonar, consumer electronics, and many other fields can be generated with digital precision and repeatability in this mode.

The tone sequence and digital sequence modes were specifically designed with mobile radio communication system requirements in mind. The DTMF sequence mode was designed for both telecommunications and mobile radio requirements. These three sequencing modes allow the user to enter a data string or sequence with specified timing. Once the specifications are entered, the HP 8904A can be instructed to output the sequence in a single burst or repetitively. These sequences are used in telecommunications and mobile radio to send data over a voice transmission channel. This data can be a telephone number, the address of a pocket pager, or ASCII text.

The hop RAM mode gives the HP 8904A the ability to hop frequency, amplitude, and phase in less than 8 μs. The switching is phase continuous and can be used to simulate many forms of digital modulation.

Channel Configuration Mode

In the channel configuration mode, the four independent synthesizers can be combined with digital precision to form complex signals. One such waveform is the stereo composite test signal used in the manufacture of FM stereo receivers. The FM stereo broadcast system uses a subcarrier system centered at 38 kHz to transmit stereo information along with the normal monophonic audio information occupying the frequency band from 20 Hz to 15 kHz. A pilot frequency of 19 kHz is used to indicate the presence of the stereo information and to aid in the synchronous demodulation of the stereo subcarrier. The HP 8904A can generate a stereo test signal with low distortion, excellent frequency flatness, and high separation. This signal consists of a sine wave audio signal, the 19-kHz pilot tone, and the 38-kHz subcarrier modulated by the audio signal. To implement this waveform with the HP 8904A, four sine waves are summed with the required frequency, amplitude, and phase settings. Table I gives the complete settings for a right-channel-only FM stereo signal, and Fig. 1 shows the composite signal generated. Channels A and B are summed to form the 38-kHz subcarrier, which is modulated at a rate of 1 kHz using double-sideband suppressed carrier modulation. Channel C is the 19-kHz pilot tone, and channel D is the 1-kHz audio tone. A critical specification for this type of system is the separation between the two audio channels. The separation of the FM stereo signals created by the HP 8904A is 65 to 70 dB, which is excellent.

Table I
Instrument Settings for an FM Stereo Composite Waveform

Channel	Destination	Frequency	Amplitude	Phase	Waveform
A	Output 1	39 kHz	1.25V	90°	Sine
B	Output 1	37 kHz	1.25V	270°	Sine
C	Output 1	19 kHz	0.3V	0°	Sine
D	Output 1	1 kHz	2.5V	0°	Sine

To test an FM receiver, the HP 8904A can be used to frequency modulate an RF signal generator operating from 88 to 108 MHz. This RF signal is sent to the receiver under test, and the audio output of the receiver is routed to an audio analyzer to measure the performance of the receiver. Fig. 2 shows a typical test setup for doing performance testing of FM stereo receivers.

Another example of a complex signal that can be made in the channel configuration mode is the VHF omnidirectional radio range (VOR) composite signal used in air navigation. The VOR system provides directional information to aircraft with respect to a specific VOR transmission site. A VOR receiver provides the pilot with a compass bearing to the VOR transmitter site. By tuning to two or more VOR

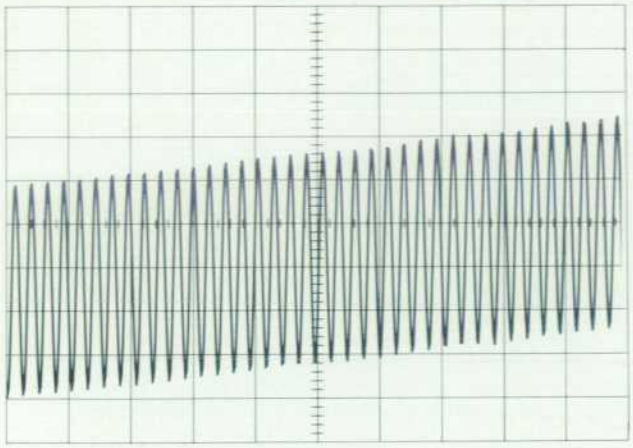


Fig. 3. *A composite VOR navigation signal generated by the HP 8904A Multifunction Synthesizer.*

transmitter stations, a pilot can determine the plane's exact location on an air chart by triangulation. The VOR system encodes the bearing information as phase offsets relative to a subcarrier reference phase. A subcarrier of 9960 Hz is frequency modulated by a 30-Hz sine wave to provide a precise phase reference signal. A second 30-Hz sine wave is summed with this subcarrier at a phase offset directly proportional to the desired bearing angle. The HP 8904A can generate the VOR composite signal with better than 0.044-degree bearing accuracy. This excellent performance is a direct result of the accuracy of the digital generation techniques used in the HP 8904A. Table II gives the HP 8904 settings for generating a VOR composite signal, and Fig. 3 shows the signal.

Table II
Channel Configuration Settings for a Composite VOR Navigation Waveform with a Bearing Angle of 30 Degrees
(Changing the phase of channel B will alter the bearing angle.)

Channel	Destination	Frequency	Amplitude	Phase	Waveform
A	Output 1	9960 Hz	1.25V	0°	Sine
B	FM	30 Hz	480 Hz	30°	Sine
C	Output 1	30 Hz	1.25V	0°	Sine
D	Off				

In a similar manner to the FM stereo example, the VOR composite signal generated by the HP 8904A can be used to modulate an RF signal generator to test a VOR receiver. In this case, the composite signal is used to amplitude modulate the RF signal generator. By programming the HP 8904A to generate VOR composite signals with different bearing angles, the accuracy of the VOR receiver under test can be verified.

Signal Sequencing Modes

The three sequence modes in the HP 8904A can be used to generate a wide range of signaling formats. These signaling formats are used in mobile communications to call pocket pagers, mobile radios, or cellular telephones. They can also be used to transmit data to mobile receivers, for example to send telephone numbers or messages to pocket pagers equipped with alphanumeric displays. The Motorola 5/6 tone signaling format is an example of a signaling scheme for the selective calling of pocket pagers. Using the tone sequence mode, the HP 8904A can generate this signaling format with synthesizer accuracy and phase continuous switching between consecutive tones. Table III gives the settings for generating the Motorola 5/6 tone format on the HP 8904A.

Once this data has been entered into the HP 8904A, a sequence can be built from the sixteen tones. A typical sequence might be "2751A." This message would specifically address pager number 27511. The A character stands for a repeat, which tells the pager that the code digit in the A position is the same as the preceding digit. Although this example only has a sequence length of five characters, the HP 8904A can have tone sequences of up to 250 tones. The HP 8904A can be directed to send the sequence once or to repeat the message continuously until it receives a stop command. Fig. 4 shows a typical setup for pocket pager testing with the HP 8904A, and Fig. 5 shows the pattern generated representing the message "2751A."

The DTMF (dual-tone multifrequency) sequence mode enables the HP 8904A to generate tone signaling sequences used worldwide for telephone switching and in mobile radio applications. In the DTMF sequence mode the sixteen standard frequency pairs of the DTMF system can be chained into sequences up to 250 tone pairs long. As in

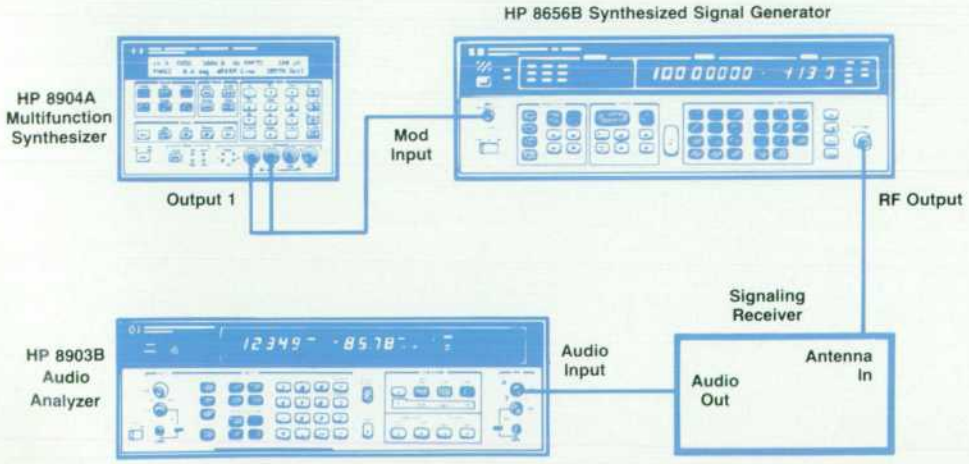


Fig. 4. *Test setup for performance testing of a pocket pager using the HP 8904A.*

the tone sequence mode, for each tone pair a specific on time and off time can be specified with 100-μs timing resolution. The HP 8904A provides synthesizer accuracy and fast, glitch-free DTMF signaling data streams. The DTMF mode in the HP 8904A can be used to test DTMF decoders, phones, DTMF-equipped mobile radios, and other telecommunications equipment.

Table III
Tone sequence information for the HP 8904A to Generate Motorola 5/6 Tone Sequential Signaling

	HP 8904A Multifunction Synthesizer Data			
Motorola 5/6 tone Symbol	HP 8904A Tone Number	Frequency	On Time (ms)	Off Time (ms)
0	0	600.0 Hz	33	0
1	1	741.0 Hz	33	0
2	2	882.0 Hz	33	0
3	3	1023.0 Hz	33	0
4	4	1164.0 Hz	33	0
5	5	1305.0 Hz	33	0
6	6	1446.0 Hz	33	0
7	7	1587.0 Hz	33	0
8	8	1728.0 Hz	33	0
9	9	1869.0 Hz	33	0
Repeat*	A	459.0 Hz	33	0
Group *	B	2151.0 Hz	33	0
Delay *	C	1.0 Hz	0	45
Gap *	E	1.0 Hz	0	52

*Special control codes for pagers.

The HP 8904A is transformed into a low-rate serial data generator when using the digital sequence mode. Many paging systems offer alphanumeric pagers capable of displaying a text message. These paging systems employ a serial digital data stream with error correction encoding to transmit this text information. In the digital sequence mode, the HP 8904A can generate serial data streams up to 1000 bits long in NRZ (non-return-to-zero) format at bit rates as fast as 10 kHz. Thus, the digital sequence mode can be used to modulate an RF signal generator with a known message for verifying the performance of pagers.

Hop RAM Mode

One benefit of the phase accumulator synthesis technique used in the digital waveform synthesis IC is very fast switching of frequency, phase, or amplitude. In effect the HP 8904A can change states in a single cycle of the DWSIC. To make this capability useful, the hop RAM mode was developed. The hop RAM mode provides sixteen memory locations which can each contain an amplitude, frequency, and phase setting. A special hop bus on the rear panel of the HP 8904A controls which of the sixteen hop states the instrument is in. Placing a four-bit, TTL-level control nibble on the hop bus will cause the HP 8904A to jump to the specified frequency, amplitude, and/or phase state. The switching is phase continuous and glitch free, and occurs in less than 8 μs. The hop RAM mode can be used for fast frequency switching applications and simulating modem signals. The HP 8904A can also be used as a direct digital modulator for PSK, FSK, multilevel FM and up to sixteen-state QAM (quadrature amplitude modulation).

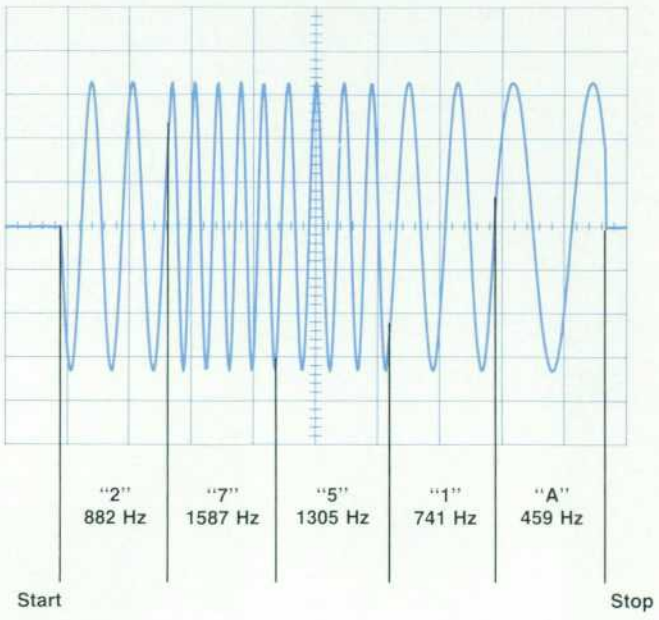


Fig. 5. *5/6 tone signaling as generated by the by the HP 8904A. This pattern represents the message "2751A" (on times have been reduced to 3 ms per tone to facilitate the plot).*

Conclusion

The HP 8904A can generate a wide range of complex signals with digital accuracy and low cost, and without the mathematical derivations required by most arbitrary waveform synthesizers. Waveforms are created by digitally summing the four synthesizers and/or by using the comprehensive modulation functions in the DWSIC. The design is optimized for low cost to compete with older-technology sources optimized for specific applications. A waveform catalog is available listing over seventy waveforms that can be generated with the HP 8904A using only the channel configuration mode. The waveforms in the catalog are a small sampling of the waveform generating capability of the HP 8904A.

Testing and Process Monitoring for a Multifunction Synthesizer

Ensuring the quality and reliability of the HP 8904A Multifunction Synthesizer required a twofold test strategy: understanding the critical characteristics of the instrument and process control.

by David J. Schwartz and Alan L. McCormick

THE TEST STRATEGY FOR THE HP 8904A required a twofold solution. The first step was to understand the instrument and its performance characteristics. Because of its state-of-the-art design, there are no reliable, fast, and automated means of directly measuring some of its critical parameters. Also, its versatility makes it impossible to measure even a significant fraction of the waveforms it is capable of producing. Understanding the instrument at this level made it possible for us to find a concise set of parameters that can be measured accurately and quickly to verify that the unit under test is working correctly, and allowed us to make small design changes that enhanced testability.

The second step of the solution was to develop a test strategy for the HP 8904A that emphasized process control, rather than intense end-of-the-line testing. By testing critical parts and modules and then verifying the assembly process, the performance of the unit under test is assured quickly and inexpensively. Fault isolation and repair are also easier because problems are identified earlier in the production process.

The production test strategy for the HP 8904A takes advantage of the instrument's straightforward block diagram and the capabilities of the HP 3065 Automated Board Test System, on which all testing is done. The output board and the digital board determine the performance of the HP 8904A. The interface between these two is clearly defined and simple to characterize. The HP 3065 gave us built-in process control routines and freedom from making numerous connections during the test process. Fig. 1 shows a block diagram of the production flow for the HP 8904A.

Two primary functions in the HP 8904A need to be tested: the generation of correct digital data and the conversion of this digital data to an analog waveform. The first function is performed entirely on the digital board. The second is done entirely on the output board. After each of these is verified, what remains is to ensure that they are connected correctly.

DWSIC and Digital Board Test

Testing of the DWSIC is done in two phases. The primary testing is done at HP's Colorado Integrated Circuit Division where the IC is tested at the functional block level. The second phase of testing is an incoming inspection, which is a simple go/no-go test intended to catch catastrophic part failures. Incoming inspection also provides us with a basis for monitoring our process to ensure that we are not damaging parts in handling them. For this test the device under test is installed in an HP 8904A test bed and instructed to produce a variety of waveforms chosen to exercize as much of the chip as possible within a reasonable amount of test time. The resulting digital data is captured by a logic analyzer and compared with stored good data.

In the remainder of the digital board testing, logic and microprocessor components are used well within their performance limits. The board's performance is verified by

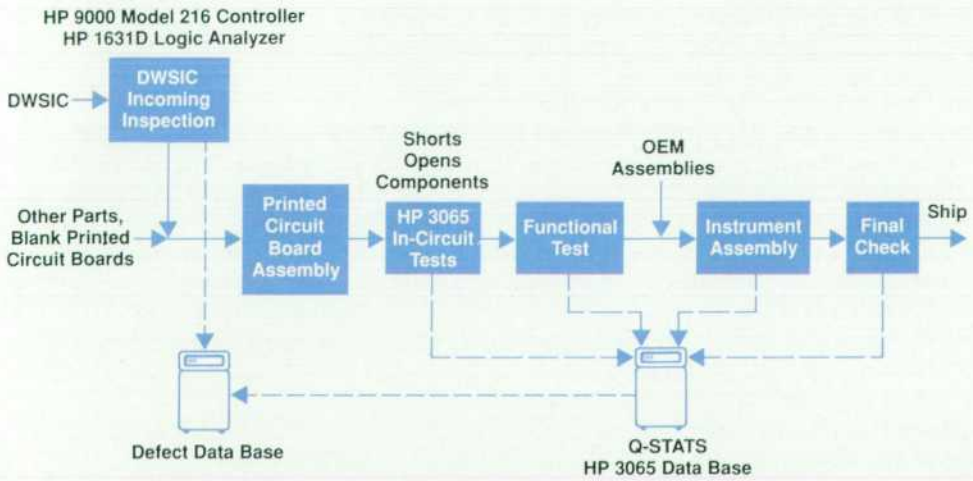


Fig. 1. *HP 8904A production flow and data collection block diagram.*

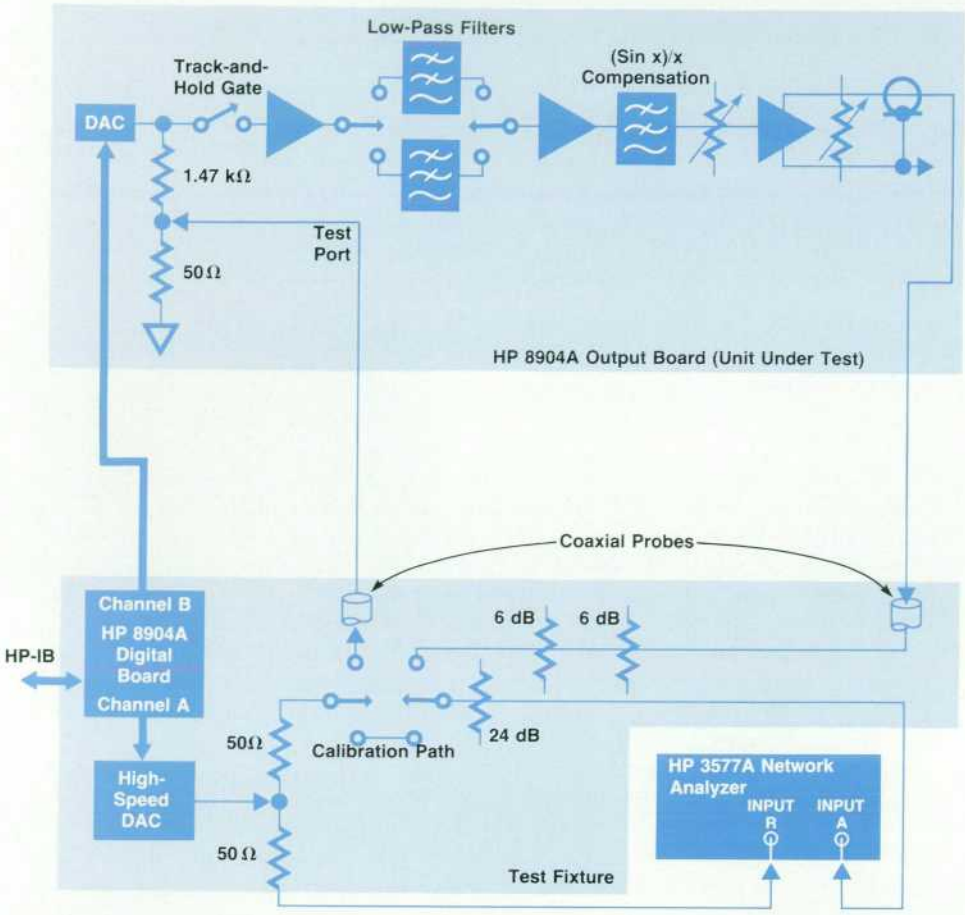


Fig. 2. *HP 8904A output board flatness test block diagram.*

conventional HP 3065 testing, the HP 8904A power-up self-check, and the ability of the board to pass the final check at the end of the line.

Output Board Test

Testing the analog output board begins with conventional HP 3065 in-circuit testing. Following in-circuit testing, a special test fixture equipped with coaxial probes is used on the HP 3065 to calibrate and verify the board under test. The tests are designed to verify the ability of the output board to convert any arbitrary stream of digital data into an analog waveform. Thus, one set of tests verifies the ability of the output board to reproduce all of the myriad waveforms that the DWSIC can generate. The output board test accomplishes this by measuring five primary characteristics: level accuracy, flatness as a function of frequency, amplitude linearity, noise and spurs, and delay distortion. From these five characteristics, the ability of the output board to reproduce a waveform can be predicted.

The biggest difficulty in making these measurements was to get the necessary accuracy. The test's measurement uncertainty had to be only a few hundredths of a percent out to 100 kHz for us to verify the amplitude flatness. At the same time, the test had to be fast enough and fully automated to meet our cost goals. This was achieved by making two-port transfer function measurements of amplitude and delay flatness with a network analyzer, using a high-speed reference DAC as a source driving the node between the HP 8904A's DAC and track-and-hold gate. The measurement is calibrated against a reference two-port network and a short length of 50-ohm coaxial cable, and gets its absolute reference from one-port voltage measurements. A separate two-port measurement is made to align the anti-aliasing filter. Two analyzers are used, the HP 3562A for frequencies below 100 kHz and the HP 3577A for those above 100 kHz. The test setup for the output board test is shown in Fig. 2.

Coaxial test probes and movable jumpers are designed into the output board for these measurements. The test station operator does not need to make any signal connections because the signals all come through the coaxial probes. The jumpers serve to isolate sections of the circuit and are moved according to prompts from the system. The reference DAC mentioned above and a switch matrix are built into the base of the output board test fixture.

Final Check

With the ability of the DWSIC to generate data confirmed by the chip tests, the ability of the digital board to issue the correct commands to the DWSIC verified by the HP 3065 test and the HP 8904A power-up self-check, and the data conversion performance of the output board verified by the special test fixture, what remains is to ensure that these pieces are assembled correctly. This is the job of final check. Performing a few fast, simple measurements on the outputs of the HP 8904A, this station provides us with a final process control check.

```
          *** REPORT CATEGORY:  Overall Production  ***

     REPORT TITLE:    MONTH END DEFECT REPORT

=====================================================================

PRODUCTION IMPACT REPORT:

                UNITS TESTED          =       2000
                UNITS FAILED          =          5
                YIELD                 =       99.8 %

                TOTAL PARTS TESTED    =     498877
                TOTAL FAILURES        =         12

                UNIT FAIL RATE        =         .4 %
                PPM FAIL RATE         =         24 ppm

                TOTAL TEST TIME       =       76.6 Hours
                AVERAGE TEST TIME     =        2.3 Minutes

                TOTAL REPAIR TIME     =        1.2 Hours
                AVERAGE REPAIR TIME   =        5.7 Minutes

=====================================================================

RESPONSIBILITY REPORT:

COMPONENT FAILURES          QTY   FAIL %   PPM FAIL RATE   REPAIR TIME
------------------------   -----  ------   -------------   -----------
Failed Components             2     16.7              4          .3 Hrs
Problem Isolation             1      8.3              2          .0 Hrs
Factory Select Values         1      8.3              2          .1 Hrs
------------------------   -----  ------   -------------   -----------
                  TOTAL       4     33.3              8          .4 Hrs

WORKMANSHIP / PROCESS       QTY   FAIL %   PPM FAIL RATE   REPAIR TIME
------------------------   -----  ------   -------------   -----------
WKCTR-399                     5     41.6             10          .3 Hrs
WKCTR-365                     2     16.7              4          .4 Hrs
WKCTR-419                     1      8.3              2          .1 Hrs
------------------------   -----  ------   -------------   -----------
                  TOTAL       8     66.7             16          .8 Hrs

=====================================================================
```

Fig. 3. *An excerpt from the overall production SQC report on failure data.*

Statistical Quality Control and Process Monitoring

One major advantage of using the HP 3065 test system is the availability of the HP 1000 as the central processor for the entire production line. The standard HP 3065 test system automatically provides excellent process control data through the built-in data base and statistical quality control (SQC) package. The standard information is based on parametric test data, and provides such information as boards tested, boards failed, first-pass yields, test times, and cycle times.

The missing link to total quality control in the standard package is the ability to identify root causes through failure data. Our production line developed utilities on the HP 1000 to provide this missing information. An HP 3065-compatible failure-data-collection system was developed to collect failure data and symptoms from all printed circuit board and instrument assembly operations. This utility was appended to the standard HP 3065 test plan to allow the operator to call the failure screens and enter defect information and remarks when necessary. The utility also runs on a stand-alone terminal to be used at all nonautomated processes. Several of these terminals are distributed throughout the HP 8904A production line.

To eliminate redundant data collection, the data collected is reformatted and uploaded to the division's failure data base on a daily basis. This reformatting utility runs automatically each night and is virtually invisible to the production line. In addition to the reformatting process, the utility also creates, for every board and instrument assembly operation, a report containing information on workmanship and process defects. This provides a tight feedback loop for each process concerning the nature of the defect and where in the process the defect was discovered.

A third utility was developed to provide SQC reports on the failure data. This utility can be run at predetermined intervals. The reports provide more exhaustive information about each process step than the daily report (see Fig. 3).

Assuring Reliability

The reliability of the HP 8904A Multifunction Synthesizer is assured by design and testing. Some methods are applied once during the design, while others continue into production.

Designing for Reliability

Each engineer is required to review the design to determine the electrical stresses of voltage, current, and power on each part. These stresses are entered into a computer data base program developed by the quality and reliability engineering department. The stresses are compared to the part specifications, base failure rates, and failure-rate-versus-stress curves. Failure rates are then predicted for each part and assembly, and for the whole instrument. Any parts with unusually high predicted failure rates are flagged for further analysis by the designer. Part temperatures are measured by thermocouple and infrared camera to confirm predicted temperature rises.

Integrating the digital waveform synthesis function into one IC contributes greatly to the low failure rate. A discrete logic implementation of this function would have a predicted failure rate nearly eight times the predicted failure rate of the DWSIC.

Reliability Verification

Class B is a standard HP environmental reliability verfication process. Included are regulatory compliance testing and tests that confirm operation to specifications over extremes of temperature, humidity, and vibration. This was done twice, first on prototype units to find problems early, and then on preproduction units for full verification.

The second round of tests was done in record time. Ten units were available so that tests could be done in parallel. Both the environmental chambers and the test system were set for timed control to run tests at night and on weekends without operator intervention. The test system checks all output specifications in less than fifteen minutes.

Strife (stress + life) tests were done. Increasingly extreme temperature limits, temperature slew rates, and vibration were applied to the product to find any design weak points.

Failure Tracking

The quality and reliability engineering failure tracking system was activated for the HP 8904A at the beginning of the project. Failures of any sort (components, processes, damage in test, etc.) are logged. An owner is assigned to the problem by quality and reliability engineering, and the problem's progress is tracked from its discovery until a solution is in place and verified. The tracking system keeps problems from being forgotten or lost, thus ensuring progress towards solutions. The process is transferred to production in a modified form for continuity of the knowledge gained.

Acknowledgments

My thanks to Dan Seely for all the work he did in these areas. He did over half the work and left his post to me with everything in place for the successful conclusion of the project.

Donald T. Borowski
Quality and Reliability Engineer
Spokane Division

This program provides some of the information about the nonautomated stations that already exists for the computer-controlled stations through the HP 3065 SQC package. Additional information includes process performance measures, total repair times, a Pareto diagram of contributions from all processes, a Pareto diagram of where defects are reported, top failures by reference designators, top failures by part number, and a Pareto diagram of the types of defects being reported. In addition, a detail report can be prepared upon request.

Finally, analysis teams have been assembled to analyze the defects and the corresponding processes involved to determine the root causes for the recurring defects. This team then submits a proposal of possible solutions to management for consideration, approval, and implementation.

This entire process has provided unprecedented process control at Spokane Division. Much of the process was implemented without additional costs using routines already in place. Continuing support costs are minimized by a paperless reporting scheme that eliminates filling out forms or retyping reports.

Acknowledgments

We would like to thank Scott Smith, Mark Secrist, and Jerry McCandless for their efforts in helping to make this a successful project.

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

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

HEWLETT-PACKARD JOURNAL

February 1989 Volume 40 • 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, 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

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.

5953-8573