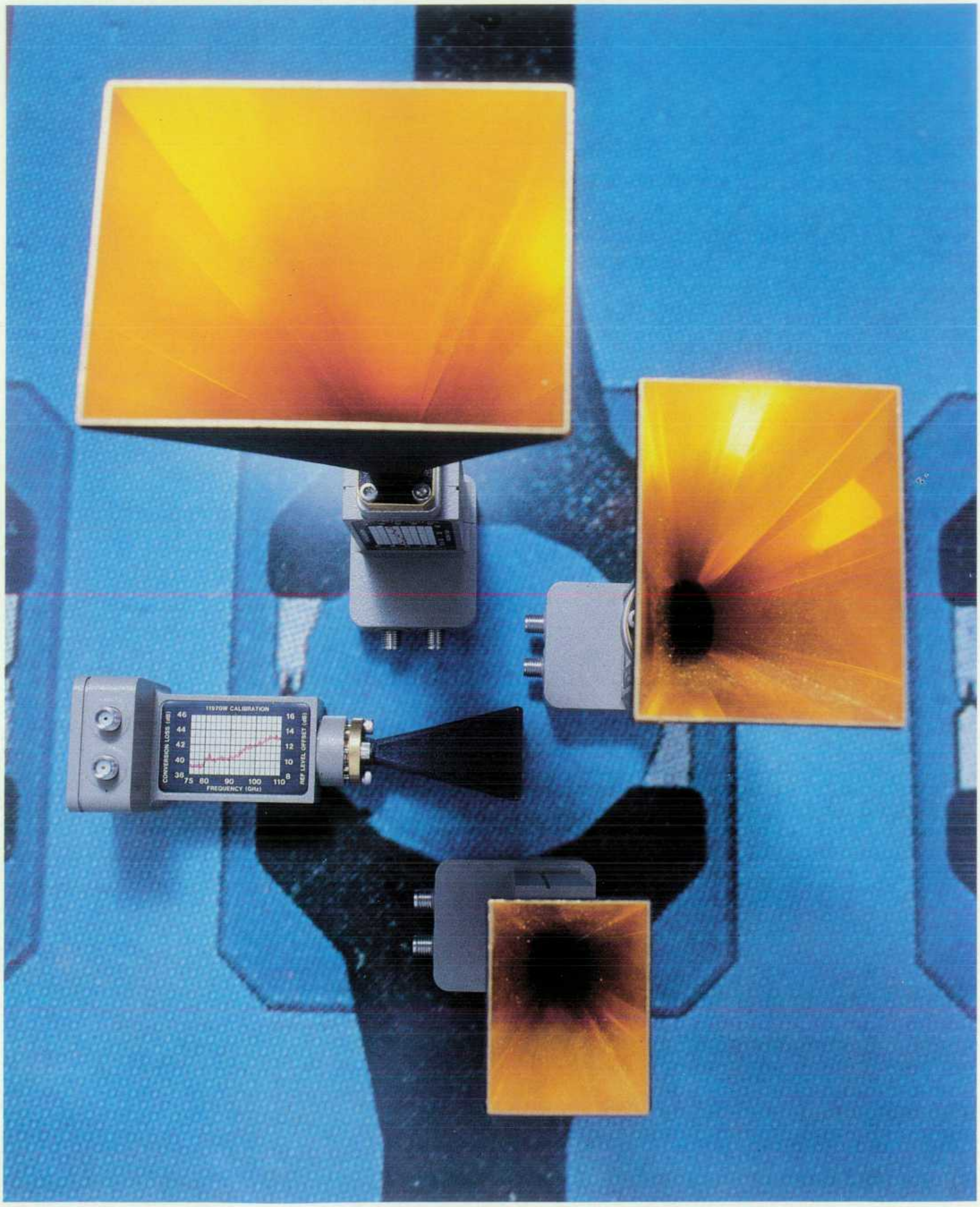
HEWLETT·PACKARD JOURNAL

NOVEMBER 1986



HEWLETT·PACKARD JOURNAL

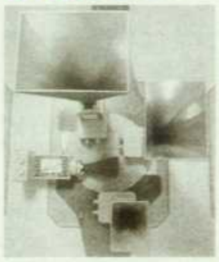
November 1986 Volume 37 • Number 11

Articles

Departments

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

In this Issue



When engineers talk about millimeter waves, they're referring to electromagnetic energy in the frequency range of 26 to 300 gigahertz, more or less. Although this region of the electromagnetic spectrum is beginning to be used for high-speed communications, high-resolution radar, guidance systems, and radio astronomy, circuit elements such as transistors are available only for the low end of this frequency range, and integrated circuits are hard to come by. In the last few years, Hewlett-Packard Laboratories and the HP Microwave Technology Division have achieved notable success in developing small-scale integrated and hybrid circuits that operate at 100 GHz and beyond. Using gallium arsenide and related compounds and a process known as molecular beam epitaxy (MBE), HP engineers have fabricated Schottky and modified barrier diodes, along with ICs and hybrids based on these diodes. Mainly useful for nonlinear and frequency translation applications such as detectors, samplers, multipliers, and mixers, they are now available in several new millimeter-wave products from HP. On page 4, Doug Collins of HP Laboratories tells us how MBE works. Modified barrier diodes and their uses in products are discussed in the article on page 10. Fabrication of ICs and hybrids is the subject of the article on page 14, and the article on page 22 describes a series of mixers that use the new GaAs diodes to extend the range of microwave spectrum analyzers to 110 GHz. The cover photograph shows several of these mixers with their horn antennas of various sizes for operation in different frequency bands.

The remainder of this issue is devoted to several papers on unusual software tools for the support of HP computer systems in the field. These are forward-looking efforts, aiming to predict failures before they happen and to use artificial intelligence (AI) technology in expert systems for troubleshooting and configuring computer systems. Predictive Support (page 30) is a software package that lives on a customer's system, counts soft errors, and notifies HP support personnel when it spots an ominous trend. Reports indicate that it works and customers like it. AIDA (page 34) is an expert system that helps HP support personnel analyze HP 3000 Computer memory dumps. A dump is a last-resort method of finding a computer problem. It may contain many megabytes of information and is hugely complex to analyze. AIDA's major contribution is its formatting capability, which helps the human expert organize all that data. That it can also find corrupt data automatically is considered a bonus by many HP troubleshooters. Schooner and IPT (pages 42 and 48) are expert systems for troubleshooting datacom links and disc drives, respectively. They've proved useful for troubleshooters learning the business and for solving the easier problems, leaving experienced personnel more time to tackle the tougher problems. Mycon (page 54) is a prototype expert configurator for computer systems. Worthy of note is its concept of multilevel constraints, which supplement and refine traditional approaches to the design of expert configurators.

R. P. Dolan

What's Ahead

The December issue will be another in our series covering HP Precision Architecture topics. Subjects included are the HP-UX operating system for technical and real-time HP Precision Architecture computers such as the HP 9000 Model 840, and ALLBASE, the data base management system for both technical and commercial HP Precision Architecture computers. December is also our annual index issue.

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

Molecular-Scale Engineering of Compound Semiconductor Materials

The ever increasing sophistication of semiconductor electronic devices and integrated circuits continues to place increasing demands on the precision with which the underlying semiconductor materials are produced. The development of molecular beam epitaxy allows the highly repeatable growth of compound semiconductor epitaxial films (such as GaAs and $Al_xGa_{1-x}As$) with atomically abrupt changes in alloy composition and doping and with excellent uniformity.

by Douglas M. Collins

SEMICONDUCTING MATERIALS provide the foundation upon which today's high-technology electronics industry is built. While silicon is the most widely used semiconducting material, and that for which the most advanced technology has been developed, its physical properties are not suitable for many of today's device needs. Thus, the use of other semiconducting materials whose electronic band structure leads to specific properties that make them more suitable than silicon for specific applications is becoming more and more common.

One such class of semiconductors is that known as the group III-V compound semiconductors. This name is derived from the fact that these materials are compounds of elements found in columns III and V of the periodic table. The most common example is gallium arsenide (GaAs), which is widely used in microwave and optoelectronic devices.

To produce high-quality compound semiconductor materials for high-performance electronic or optoelectronic devices, it is necessary to use epitaxial growth techniques. This is especially true for those devices that require heterojunctions (for a discussion of heterojunctions, see box on page 6). In general, an epitaxial growth process is any process in which chemical elements or compounds are deposited onto a single-crystal substrate under conditions such that the deposited materials become precisely arranged upon the substrate, yielding a single-crystal deposited, or epitaxial layer. In fact, the word epitaxy is derived from the Greek words *epi* (meaning "on") and *taxis* (meaning arrangement).

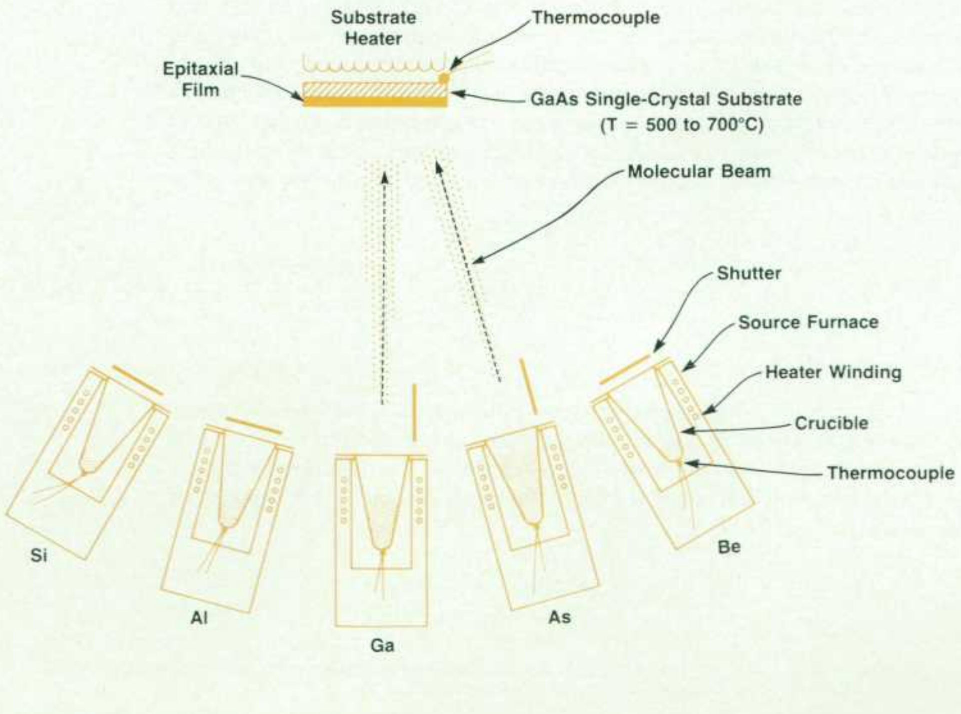


Fig. 1. *Schematic representation of the molecular beam epitaxy (MBE) technique.*

To push devices to higher and higher levels of performance, it is necessary to control the epitaxial growth process precisely. The molecular beam epitaxy technique is one that provides for the precise growth of complex III-V compound semiconductor structures which are particularly well-suited for high-speed (e.g., millimeter-wave) device and integrated circuit applications.

The MBE Technique

Molecular beam epitaxy (MBE) can best be described as a highly controlled ultrahigh-vacuum evaporation/deposition process. The apparatus used for the MBE growth of GaAs and $Al_xGa_{1-x}As$ is shown schematically in Fig. 1. (For a discussion of alloy semiconductors such as $Al_xGa_{1-x}As$, see box on page 8.) The principal components in this system are the resistance-heated source furnaces, the source furnace shutters, and the heated substrate station. Three of the source furnaces contain the elements Ga, Al, and As which, when reacted on the heated single-crystal substrate, produce the epitaxial semiconductor layer. The other two sources contain silicon and beryllium, the two most commonly used dopants in the MBE growth of GaAs and $Al_xGa_{1-x}As$. Silicon substitutes for the gallium or aluminum atoms on the group-III sublattice resulting in one extra electron; hence silicon is an n-type, or donor, impurity in these materials. Beryllium also substitutes for the aluminum or gallium atoms, but with one less electron; hence beryllium is a p-type, or acceptor, impurity in these materials.

The rate of evaporation of the source materials, and thus the flux of atoms or molecules arriving at the substrate surface, is determined by the temperature of each of the sources. Each source temperature is individually controlled to within ±0.2°C, which results in less than ±1% variation in the source flux.[1] The temperatures (and thus the fluxes) of gallium and aluminum (both of which evapo-

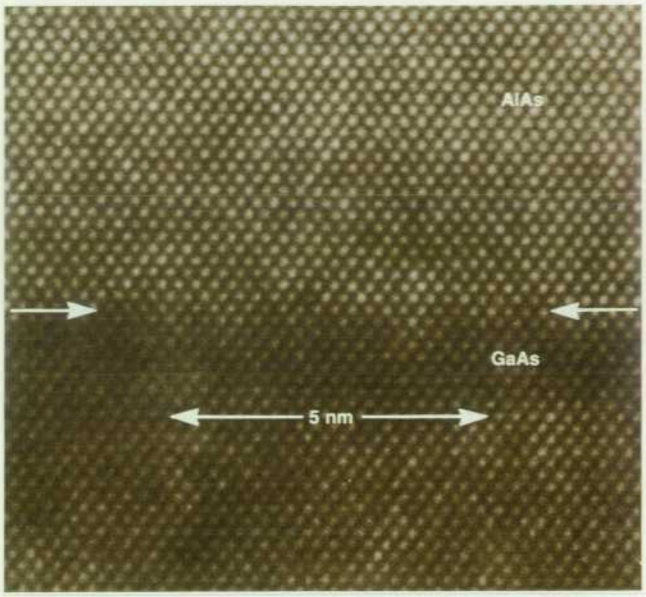


Fig. 2. *High-resolution transmission electron micrograph of atomically abrupt GaAs/AlAs heterojunction formed by MBE.*

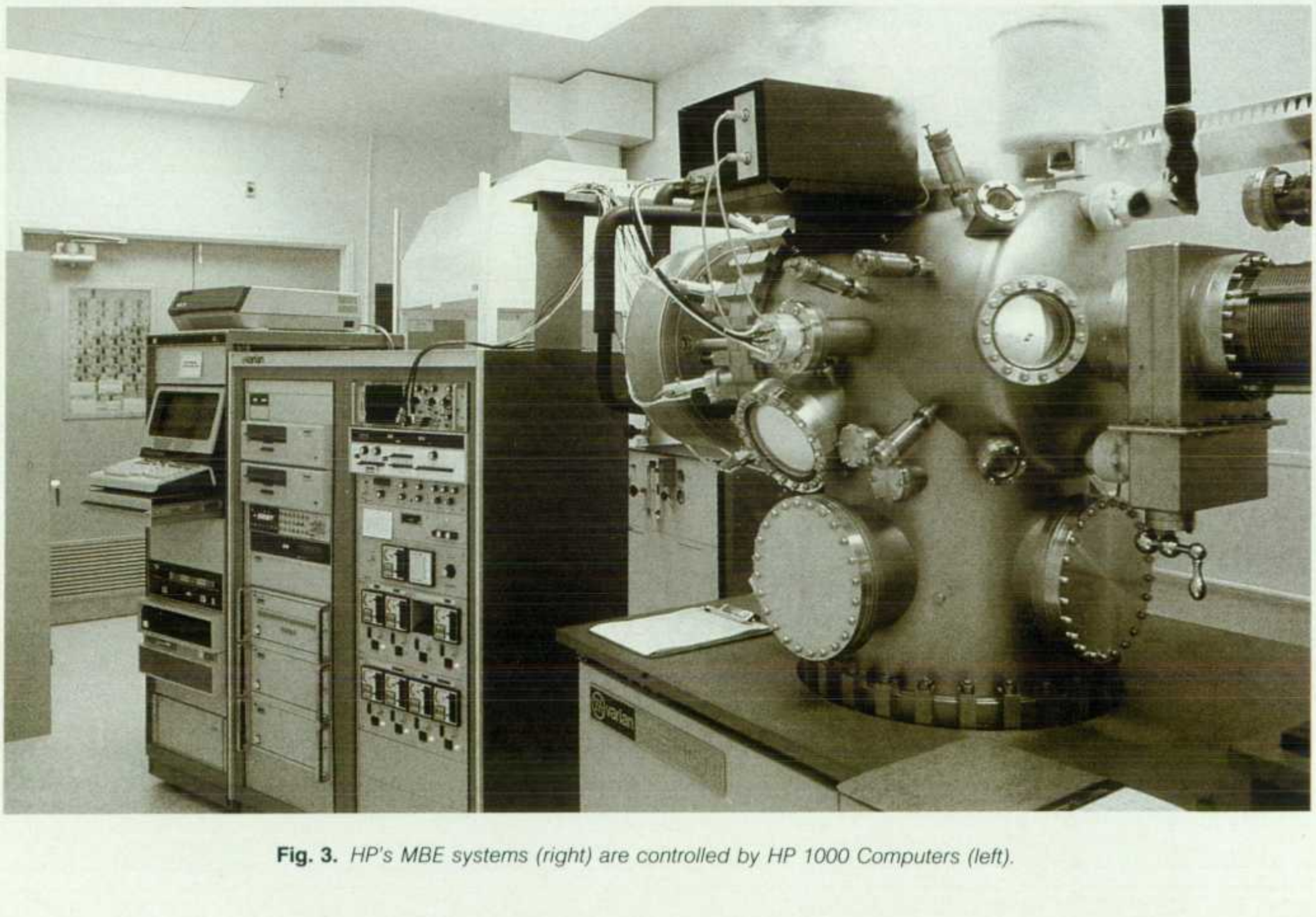


Fig. 3. *HP's MBE systems (right) are controlled by HP 1000 Computers (left).*

Compound Semiconductor Alloys and Heterojunctions

The properties of semiconducting materials that are of practical interest depend on the material's crystal structure, the interatomic spacing in the crystal, the electronic band structure, and the thermodynamics of defect formation, in particular the incorporation of impurity or dopant atoms that alter the electronic properties in a desired fashion. These factors combine to determine the bandgap energy E_g of the semiconductor (and whether the bandgap is "direct" or "indirect"), the transport properties of electrons and holes (in particular the mobilities of these charge carriers), and the practical matter of being able to control the electron and hole concentrations over the ranges necessary for use in devices.

Silicon crystallizes into the so-called "diamond" structure (Fig. 1a), which can be described as two interleaved face-centered cubic lattices (an FCC lattice has an atom at every corner of a cube in addition to an atom at the center of every face of the cube). One FCC lattice has its origin at 0, 0, 0 and the other FCC lattice has its origin at a/4, a/4, a/4 (only a portion of the second lattice is shown in Fig. 1). This results in a tetrahedral coordination between each silicon atom and its four nearest neighbors.

Most of the III-V compound semiconductors, including GaAs, crystallize into the so-called "zincblende" structure (Fig. 1b). This structure is remarkably similar to that of silicon. In fact, it is the diamond structure, except that one of the two interleaved FCC sublattices is composed of group-III atoms (e.g., gallium) while the other is composed of group-V atoms (e.g., arsenic). Thus each group-III atom is tetrahedrally bonded to four nearest-neighbor group-V atoms, and each group-V atom is tetrahedrally bonded to four nearest-neighbor group-III atoms.

The additional complexity of the zincblende structure offers a degree of freedom not available in elemental semiconductors such as silicon and germanium. This freedom is the ability to mix different group-III elements on the group-III sublattice as well as different group-V atoms on the group-V sublattice. Thus we can form ternary (e.g., $Al_xGa_{1-x}As$) and quaternary (e.g., $In_xGa_{1-x}As_yP_{1-y}$) alloy semiconductors, where $0 \leq x \leq 1$ and $0 \leq y \leq 1$. The subscripts x and y in these chemical formulas indicate the fraction (commonly called the mole fraction) of each sublattice that is occupied by each element. Since, in general, different semiconductors have different bandgap energies and different electron and hole transport properties, the ability to form alloy semiconductors provides the materials scientist with the freedom to engineer the material to the needs of the device physicist. Thus it is possible, for example, to adjust the bandgap of a semiconductor to give the desired wavelength of emission for a light-emitting diode.

In addition to providing the flexibility of engineering semiconductor material properties by forming these alloys, compound semiconductor technology provides an added capability with far-reaching consequences—the ability to join two different semiconductor materials intimately together to form a semiconductor heterojunction. This is conceptually very similar to joining together p-type and n-type silicon to form a semiconductor pn homojunction. However, it leads to many more potential applications of III-V compound semiconductors. For example, if a thin layer (approximately 0.2 micrometer) of one semiconductor is sandwiched between two layers of a second semiconductor that has a larger bandgap and a smaller index of refraction, the center layer will serve to confine both charge and light, thus providing two of the necessary conditions for the formation of a population inversion in this layer, which leads to semiconductor laser action.[1] Such structures are known as double heterojunctions. Another technologically important heterojunction is the modulation-doped heterojunction (see box, page 8), which is important to the very high-speed operation of the modulation-doped field effect transistor (MODFET, also known as the high-electron-mobility transistor, or HEMT).[2]

To form semiconductor heterojunctions with a high degree of structural perfection, the two semiconductors must have identical (or compatible) crystal structures and nearly equal interatomic spacings, or lattice constants (dimension a in Fig. 1). This requirement can be met by using compound semiconductor alloys that have the same crystal structure while simultaneously choosing alloy compositions (x and/or y in the above chemical formulas) that result in the same lattice constants. This is a condition which is referred to as lattice matching. In the GaAs/$Al_xGa_{1-x}As$ material system discussed in the accompanying article, the lattice match is nearly perfect for all values of x.

References

1. H.C. Casey and M.B. Panish, *Heterostructure Lasers: Part A, Fundamental Principles, and Part B, Materials and Operating Characteristics*, Academic Press, New York, 1978.
2. H. Morkoç and P.M. Solomon, "The HEMT: A Superfast Transistor," IEEE Spectrum, Vol. 21, no. 2, February 1984, pp. 28-35.

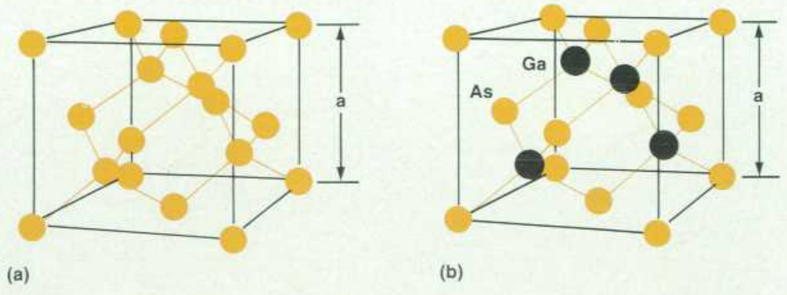


Fig. 1. *Semiconductor crystal structures (a = lattice constant). (a) Diamond structure for silicon, carbon, germanium, etc. (b) Zincblende structure for GaAs, GaP, InSb, etc. Bonds between nearest neighboring atoms are shown in color.*

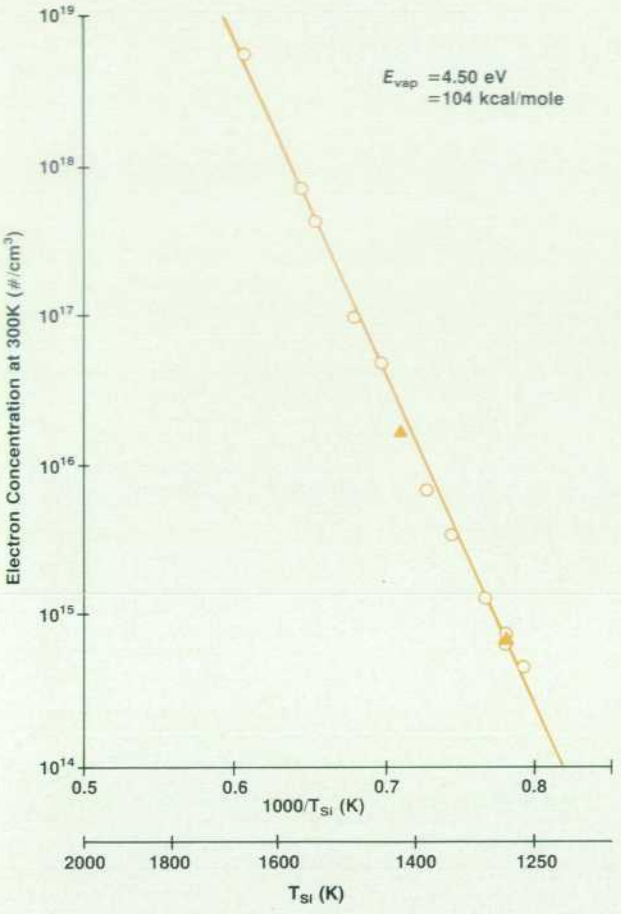


Fig. 4. *Free electron concentration at room temperature (300K) for Si-doped MBE GaAs as a function of silicon source temperature.*

rate as monatomic beams) are chosen to give the desired growth rate and alloy composition. The temperature of the arsenic source (arsenic evaporates as the As_4 molecule) is chosen such that the As_4 flux is slightly higher than that required for growth of stoichiometric GaAs or $Al_xGa_{1-x}As$ (i.e., where the number of gallium and aluminum atoms incorporated into the film is equal to the number of arsenic atoms incorporated into the film). The small excess of arsenic is used to ensure nearly perfect stoichiometry since, at the substrate temperatures used (typically 500 to 700°C), the arsenic will bond only to gallium or aluminum and not to itself. The excess arsenic evaporates from the film surface.[2] Similarly, the temperatures of the silicon and beryllium sources are chosen such that the fluxes of these elements lead to the desired concentrations of n-type and p-type doping, respectively.

The temperatures of the gallium and aluminum sources are typically chosen to result in a growth rate of 1.0 μm/hour. This corresponds to only one molecular layer of GaAs or $Al_xGa_{1-x}As$ per second. Since each source furnace is equipped with its own individual shutter that can be opened or closed in a small fraction of a second, it is a simple matter to control the composition or doping in an epitaxial film on an atomic scale. An example of an atomically abrupt GaAs/AlAs heterojunction (grown by closing the gallium shutter and opening the aluminum shutter simultaneously) is shown in the high-resolution transmission electron micrograph of Fig. 2.

The MBE systems in operation at Hewlett-Packard Company (e.g., see Fig. 3) are interfaced with HP 1000 Computers that control the temperature setpoints for the source furnaces and the substrate, the source shutter operation, and the growth times for all layers in multilayer films. In addition to having the capability for abruptly changing the composition or doping of MBE films, the computer is also used to contour alloy compositions and doping profiles programmably to achieve specific materials structures for use in particularly high-performance devices.[3]

For the high degree of control offered by MBE to be of practical use, it is necessary that the undesirable background impurity levels in MBE films be as low as possible and that the films be highly uniform. These conditions are easily met in the MBE growth of GaAs. The ultrahigh vacuum system used in MBE is pumped by ion and titanium sublimation pumps and includes extensive liquid-nitrogen cryoshrouding to pump condensable gases such as water vapor. Background partial pressures of undesirable gases are typically below 10^{-11} torr. This allows undesirable impurity levels as low as ten parts per billion to be achieved. Continuous rotation of the substrate during growth leads to demonstrated uniformities in thickness (or growth rate) and doping concentration of better than ±2% across a two-

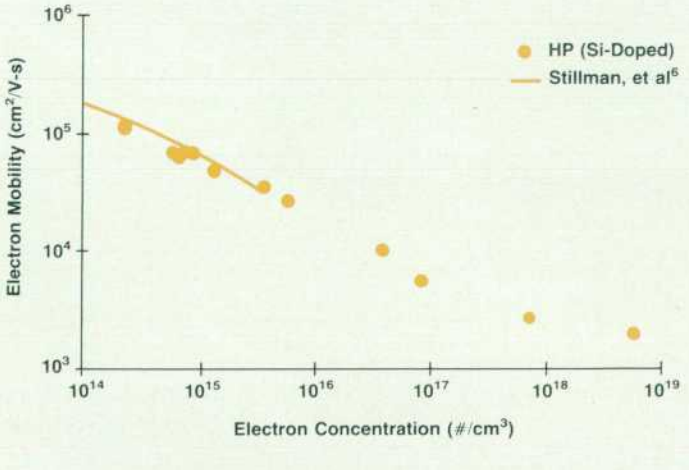


Fig. 5. *Mobilities for Si-doped MBE GaAs as a function of free electron concentration at 77K.*

The Modulation-Doped Heterojunction

The accompanying article demonstrates that MBE is capable of producing GaAs and $Al_xGa_{1-x}As$ films with a high degree of control over doping concentration and alloy composition and with extremely abrupt heterojunction interfaces. All of these capabilities are important for the realization of the high-speed device known as the modulation-doped FET. The layered material structure upon which this device is based is commonly referred to as a selectively doped, or modulation-doped heterojunction (see Fig. 1).[1] It has been most commonly produced in the GaAs/$Al_xGa_{1-x}As$ material system, although other semiconductor heterojunctions have also been used.[2] The epitaxial layer structure is grown on a high-resistivity (10^7 ohm-cm) GaAs substrate. The first layer is undoped GaAs approximately one micrometer thick, which must be of very high purity (typically less than 10^{15} ionized impurities per cubic centimeter). The second layer is undoped $Al_xGa_{1-x}As$ ($x = 0.3$) which, depending on the desired properties of the structure, will typically be between 2 and 15 nanometers thick. This corresponds to only 8 to 60 atomic layers. The atomic scale abruptness of the heterojunction between these two layers and the precise control of the undoped $Al_{0.3}Ga_{0.7}As$ "spacer" layer thickness are critical to the proper performance of this device. The third layer is silicon-doped ($N_{Si} = 2\times10^{18}$ cm^{-3}) $Al_xGa_{1-x}As$ with x typically the same as that in the undoped spacer layer which, again depending on the desired properties of the structure, will be between 10 and 50 nanometers thick. The fourth, and last, layer is n-type GaAs ($n = 2\times10^{18}$ cm^{-3}) approximately 20 nanometers thick. This layer facilitates making low-resistance ohmic contacts to the structure.

The special properties of this structure that make it of interest for high-speed device applications are best understood by referring to the energy band diagram shown in Fig. 2. Here we see that a conduction band discontinuity is formed at the abrupt GaAs/$Al_xGa_{1-x}As$ interface. This is accompanied by depletion of electrons from the $Al_xGa_{1-x}As$ and accumulation of these electrons in the GaAs. The potential "notch" in which these electrons reside confines the electrons to a region so thin that they behave as a two-dimensional system. This system is commonly referred to as a two-dimensional electron gas. This electron gas has many similarities to the two-dimensional electron gas in the gate channel of a silicon MOSFET. However, there are two very important differences. First, the GaAs/$Al_xGa_{1-x}As$ heterojunction is very nearly structurally perfect and thus electrons traveling parallel to this heterojunction interface are unlikely to scatter from structural defects. Second, since the GaAs is of very high purity and the ionized silicon donors in the $Al_xGa_{1-x}As$ are set back from the heterojunction interface by the spacer layer thickness, scat-

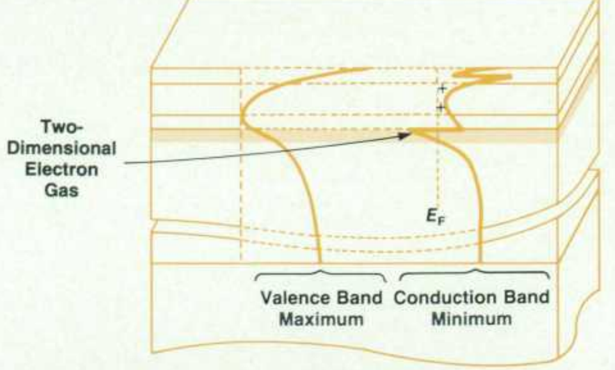


Fig. 2. *Energy band diagram for structure of Fig. 1. The two-dimensional electron gas formed in the structure is ideal for the conducting channel of a FET.*

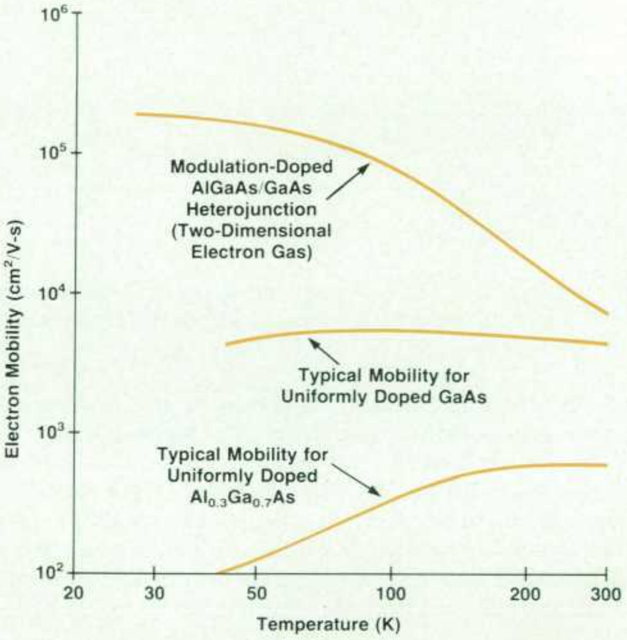


Fig. 3. *Comparison of electron mobilities for a modulation-doped heterojunction and the two uniformly doped materials used to form the heterojunction.*

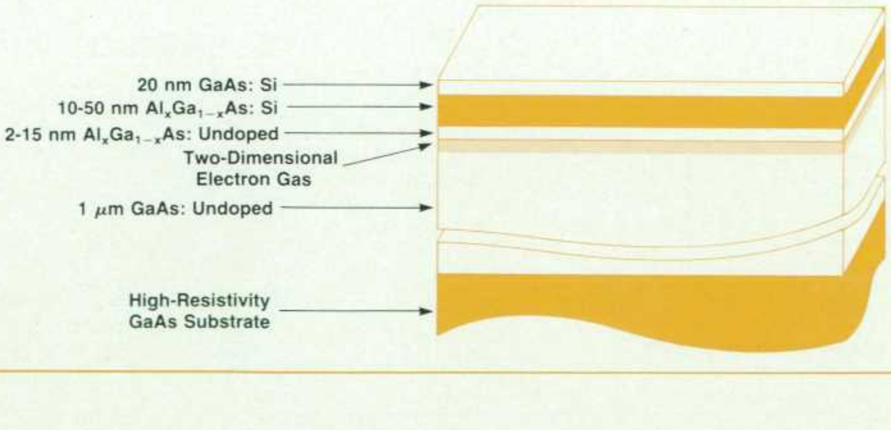

20 nm GaAs: Si
10-50 nm $Al_xGa_{1-x}As$: Si
2-15 nm $Al_xGa_{1-x}As$: Undoped
Two-Dimensional Electron Gas
1 μm GaAs: Undoped
High-Resistivity GaAs Substrate

Fig. 1. *Cross section of a modulation-doped heterojunction.*

tering from the silicon donor atoms is minimized. These two factors result in excellent transport properties for electrons in this modulation-doped heterojunction structure.

The effect on electron transport properties is shown in Fig. 3 where the significantly enhanced mobilities found in these heterojunction structures (top curve) are compared with those observed in uniformly doped GaAs and $Al_{0.3}Ga_{0.7}As$ with comparable sheet electron concentrations. (In this case, $n_s = 6\times10^{11}$ cm^{-2}, which is comparable to the sheet carrier concentration in the channel of an FET). This data shows that the mobility in the modulation-doped structure is a much stronger function of temperature than the mobilities in the uniformly doped films. This occurs because the mobility at higher temperatures in the modulation-doped structure is still dominated by phonon scattering (i.e., lattice vibrations). However, even at room temperature, the modulation-doped structure exhibits a mobility that is a factor of 1.5 to 2 higher than in the uniformly doped GaAs material. In short-gate-length MODFETs this results in an increase in the electron peak velocity by a factor of 1.5 to 2 and thus a comparable increase in the speed of operation of the device.[3]

References

1. R. Dingle, et al, "Electron Mobilities in Modulation-Doped Semiconductor Superlattices," *Applied Physics Letters*, Vol. 33, 1978, pp. 665-667.
2. T.P. Pearsall, et al, "Selectively-Doped AlInAs/GaInAs Heterostructure Field Effect Transistor," *IEEE Electron Device Letters*, Vol EDL-4, no. 1, 1983, pp. 5-8.
3. M. Hueschen, et al, "Pulse Doped MODFETs," *1984 International Electron Devices Meeting Technical Digest*, pp. 348-351.

inch diameter substrate.

MBE Grown GaAs and $Al_xGa_{1-x}As$

The realization of the precise control of epitaxial film parameters promised above can best be demonstrated by presenting results of doping and alloy composition control for MBE films. Fig. 4 shows the free electron concentration (n, as measured by the Hall effect[4]) at room temperature for Si-doped MBE GaAs as a function of the reciprocal of the silicon source temperature. Several important points may be made from this data. First, it is clear that we have the capability of varying the n-type doping of MBE GaAs over a wide range (from less than 10^{15} cm^{-3} to 5×10^{18} cm^{-3}, which corresponds to approximately 0.02 to 100 parts per million) with a high degree of control. In fact, the slope of the precisely exponential variation in n with $1/T_{Si}$ corresponds closely to the activation energy for the evaporation of silicon, indicating that the silicon incorporation rate is directly proportional to the silicon flux.[5] The two different symbols in Fig. 4 correspond to two different series of MBE growth runs, the second having been carried out after refilling the gallium and arsenic source furnaces. Thus excellent control and repeatability are readily achieved in the MBE growth of GaAs.

To achieve superior performance of electronic devices, it is necessary not only to control carrier concentration precisely, but also to assure good carrier transport properties. A common measure of the quality of n-type GaAs is the electron mobility. Mobility, as the term suggests, is a measure of the ease with which the charge carriers move through the crystal lattice. In particular, high mobilities in lightly doped n-type GaAs at low temperatures (e.g., liquid-nitrogen temperature of 77K) are indicative of low concentrations of background impurities since, under these conditions, the mobility is dominated by scattering of electrons by impurities in the crystal. Mobilities for Si-doped, MBE-grown GaAs are shown in Fig. 5 as a function of the free electron concentration. This data was obtained at a temperature of 77K, where impurity scattering dominates. The solid line shown for low doping levels (below 5×10^{15}

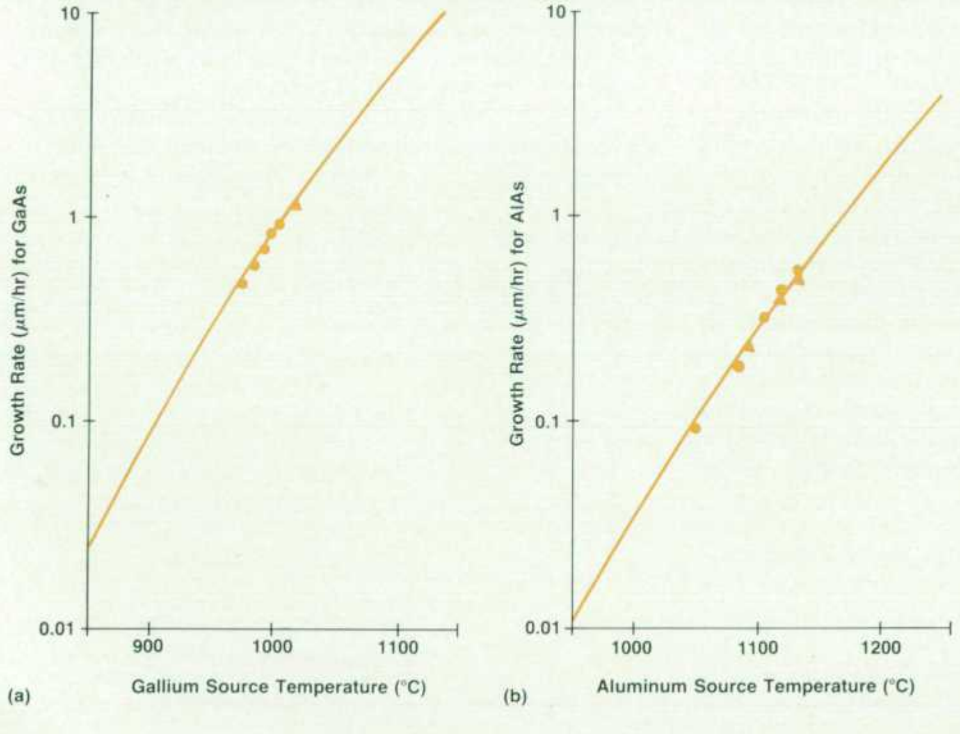


Fig. 6. *Growth rate as a function of (a) gallium and (b) aluminum source temperatures. The symbols represent data for different growths and the solid lines represent the relative temperature dependencies of the vapor pressures of (a) gallium and (b) aluminum.*

cm^{-3}) represents mobilities for extremely high-purity GaAs grown by the $AsCl_3$ vapor phase epitaxy (VPE) technique.[6] It is clear that the MBE GaAs compares very favorably with the high-purity VPE material.

The control of the MBE growth of $Al_xGa_{1-x}As$ is demonstrated in Fig. 6 where growth rate is shown as a function of gallium (Fig. 6a) and aluminum (Fig. 6b) source temperatures. The symbols represent experimental data for different growths. The solid curves represent the relative temperature dependencies of the vapor pressures of gallium and aluminum. Thus, by appropriate choices for the gallium and aluminum source temperatures, both growth rate and alloy composition of GaAs and $Al_xGa_{1-x}As$ can be readily controlled and reproduced using MBE.

Acknowledgments

The successful development and application of a technology such as MBE can only be achieved through effective teamwork. I am very grateful to Midori Kanemura, Marge Pustorino, Dan Mars, Jeff Miller, Bob Lacoste, and Tom Low of Hewlett-Packard Laboratories and George Patterson of HP's Microwave Technology Division for their contributions to these efforts.

References

1. P.E. Luscher and D.M. Collins, "Design Considerations for Molecular Beam Epitaxy Systems," *Progress in Crystal Growth and Characterization*, Vol. 2, 1979, pp. 15-32.
2. A.Y. Cho and J.R. Arthur, "Molecular Beam Epitaxy," *Progress in Solid-State Chemistry*, Vol. 10, Part 3, 1975, pp. 157-191.
3. S. Kofol, et al, "GaAs/AlGaAs Heterojunction Bipolar Transistors with Very Low Base Sheet Resistance," *1984 IEEE International Electron Devices Meeting Technical Digest*, pp. 198-200.
4. S.M. Sze, *Physics of Semiconductor Devices*, Wiley-Interscience, New York, 1969, pp. 42-46.
5. D.M. Collins, "On the Use of 'Downward-Looking' Sources in MBE Systems," *Journal of Vacuum Science and Technology*, Vol. 20, no. 2, 1982, pp. 250-251.
6. G.E. Stillman and C.M. Wolfe, "Electrical Characterization of Epitaxial Layers," *Thin Solid Films*, Vol. 31, 1976, pp. 69-88.

Extending Millimeter-Wave Diode Operation to 110 GHz

by Eric R. Ehlers, Sigurd W. Johnsen, and Douglas A. Gray

IN THE LATE 1970s, silicon Schottky diodes and point contacts dominated diode applications in the microwave and low millimeter-wave range of frequencies. By using carefully designed microwave structures, performance well into the millimeter-wave range could be achieved. This performance, however, did not come easily. These structures were difficult to reproduce and as a result were very expensive. The small diode structures were also sensitive to damage by electrostatic discharge (ESD). Although Schottky diode pairs and quads were readily available, diodes integrated with other circuit elements could not be obtained as off-the-shelf components.

When HP launched its effort to develop a family of diodes particularly suited to millimeter-wave applications, we wanted a device that could be readily integrated into a microwave structure, was rugged and capable of handling high drive levels for multiplier applications, and had at least the sensitivity of presently available diodes.

Diode Technology

To meet the project goals of designing low-capacitance, high-power-handling diodes suitable for small-scale integrated circuits, the selection of material technology was of paramount importance. Although silicon technology was very mature and some silicon devices did exist for applications up to 40 GHz, the high parasitic capacitance of silicon substrates made it difficult to integrate other circuit elements and beam leads without sacrificing high-frequency performance or power handling. It was clear that the design and fabrication of a useful device up to 110 GHz (W-band) required a different material technology.

The III-V compound semiconductor gallium arsenide (GaAs) offers several advantages over silicon. The intrinsic doping in GaAs is two orders of magnitude lower than

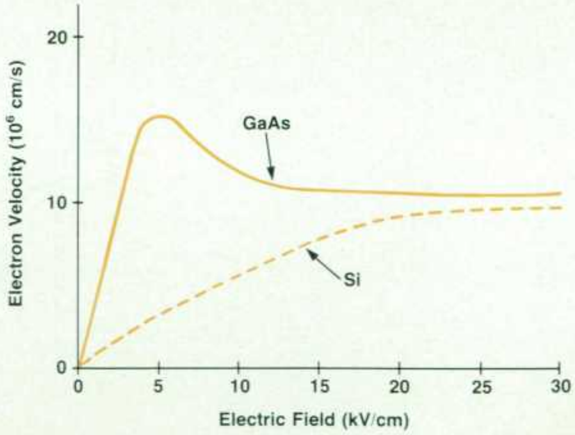


Fig. 1. *Electron velocity in gallium arsenide and silicon.*

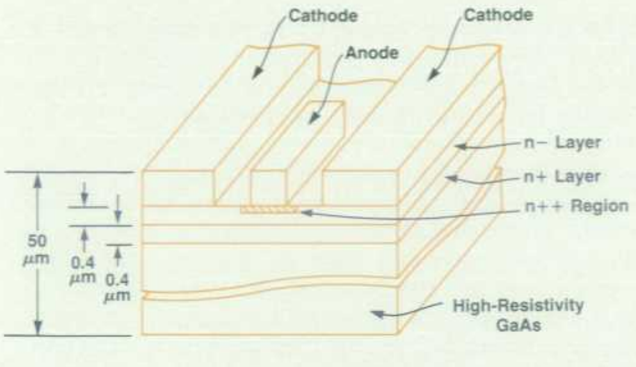


Fig. 2. *Cross section of sintered Schottky barrier diode.*

silicon. This results in a much higher substrate resistivity and lower parasitic capacitance effects. The lower parasitic capacitance of GaAs substrates is critical. It allows the design of larger devices with increased power handling capabilities without degrading frequency performance.

Another advantage of GaAs is its higher electron mobility (see Fig. 1). This higher mobility translates up to a 5× higher electron velocity in GaAs compared to silicon. This leads to much lower carrier transit times and makes it possible to design devices exhibiting cutoff frequencies in excess of 1000 GHz.

For these reasons GaAs was the material of choice for the millimeter-wave diode project. There was a major disadvantage, however, because a primary goal of the project was to design low-barrier and medium-barrier diodes which are commonly used in mixer and detector applications. Although HP has been very active in GaAs research and development (HP instrumentation has been shipped using internal GaAs FET and diode technology since the early 1970s), a reliable method of repeatably fabricating a modified barrier diode on GaAs presented a major technical challenge.

The difficulties in fabricating modified barrier diodes on GaAs were overcome by a two-phase approach. Standard Schottky barrier diodes are fabricated using liquid-phase epitaxy to grow the double n-on-n+ layers required for millimeter-wave IC fabrication. Low-barrier Schottky diodes were originally fabricated using the same epitaxial layers, but with a proprietary Schottky barrier metallization (Fig. 2). The low-barrier diode was formed by a sintering process after the Schottky metal was patterned. The sintering formed a thin, heavily doped n+ layer under the anode which lowered the barrier height to the desired value (see Fig. 3). This process led to the introduction of a 26-GHz-to-40-GHz zero-biased detector fully integrated on one GaAs chip (see box on page 13). The sintering process was, however, very difficult to control and the devices were extremely sensitive to ESD, which resulted in relatively low yields during microcircuit assembly. However, even with the low yields, this device represented a significant advance in the state of the art of millimeter-wave device technology.

The second generation of modified barrier diodes was made possible by the capabilities provided by molecular beam epitaxy (MBE, see article on page 4). MBE technology provides the means for very controlled growth of the epitaxial layers required for the fabrication of planar doped barrier diodes (also called modified barrier diodes by HP). These diodes are formed by growing a multilayer GaAs epitaxial structure sandwiched between two low-resistance contacts (see Fig. 4). The design of the epitaxial structure

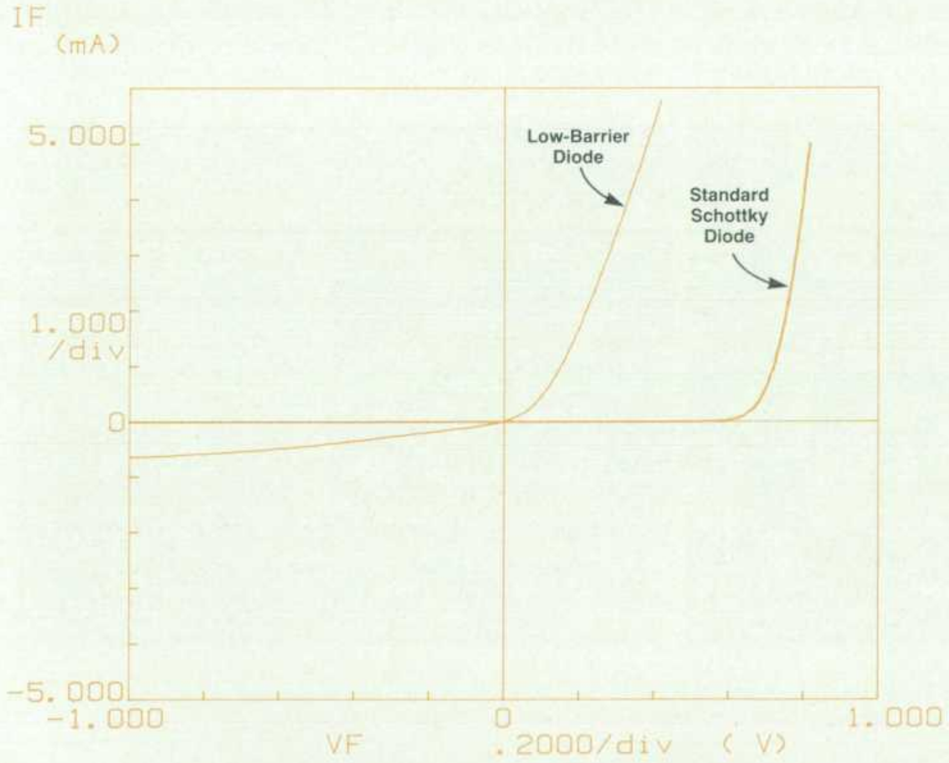
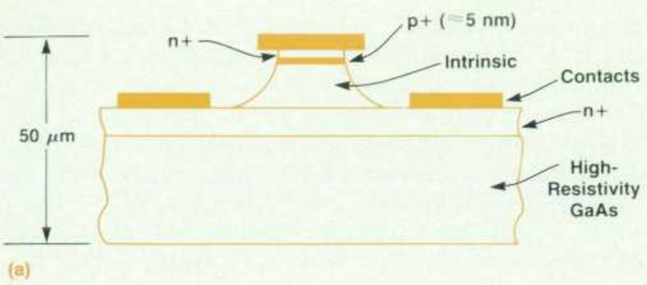


Fig. 3. *I-V characteristics for standard Schottky diode and sintered low-barrier version.*



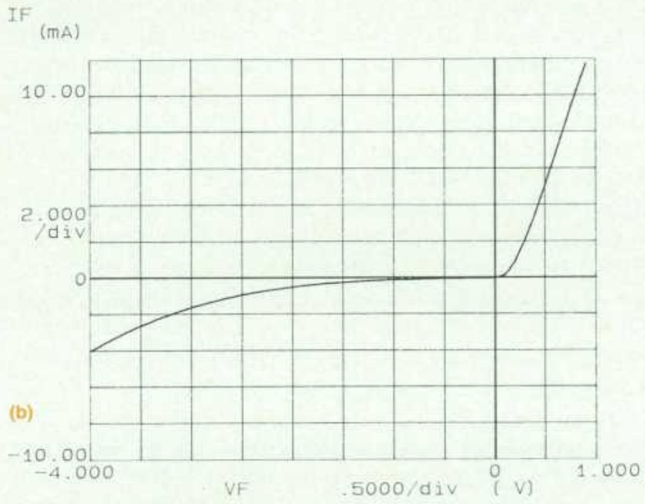


Fig. 4. *(a) Cross section of planar doped barrier diode fabricated by molecular beam epitaxy (MBE). (b) I-V characteristic.*

determines the barrier height of the diode. Therefore, extremely tight control of epitaxial layer doping and thickness is required. Epitaxial layers as thin as 4 nm must be grown with abrupt transitions between layers. This critical step in the fabrication of modified barrier diodes has been moved out of the process lab and placed on the shoulders of the computer-controlled MBE system. The state of MBE technology at HP has demonstrated its consistent performance to these demanding specifications, enabling the design and fabrication of device structures for both low-barrier and medium-barrier diode applications.

With the technology in place for manufacturing GaAs diodes with varying barrier heights, the next step was to provide the capability for fully integrated solutions of some standard millimeter-wave circuit functions. Most of the millimeter-wave circuits used in instrumentation are samplers, mixers, multipliers, and detectors. At frequencies above 26 GHz, the repeatability of hybrid assembly techniques is critical. A slight variation in a bond-wire length can be a significant fraction of a wavelength at these frequencies. The integration of a millimeter-wave diode, resistors, and capacitors on a single GaAs chip with beam leads for thin-film circuit interconnects offers superior performance in phase matching for samplers and greatly reduces the complexity and assembly costs of millimeter-wave hybrid circuits. The integration process developed provides the circuit designer with the flexibility for integrating silicon nitride capacitors with values up to 20 pF and tantalum nitride resistors with a sheet resistivity of 50 $\Omega/\square$.

The technology is directed toward allowing design engineers working on new millimeter-wave test instruments to design their own custom integrated circuits. There are some roadblocks, however. A designer must work very closely with the IC facility to make sure process guidelines or limits are not violated by the design. This can be the most time-consuming and frustrating part of the design stage. There is also a fear of the high development costs that are normally associated with a custom IC design. This factor is especially a concern at millimeter-wave frequencies where current computer-aided-design techniques are not sufficient to ensure a successful circuit design on the first try.

To lessen some of these difficulties and to encourage the use of this technology at HP, it was decided to offer it in a prototype mode. In this approach, circuit designers can

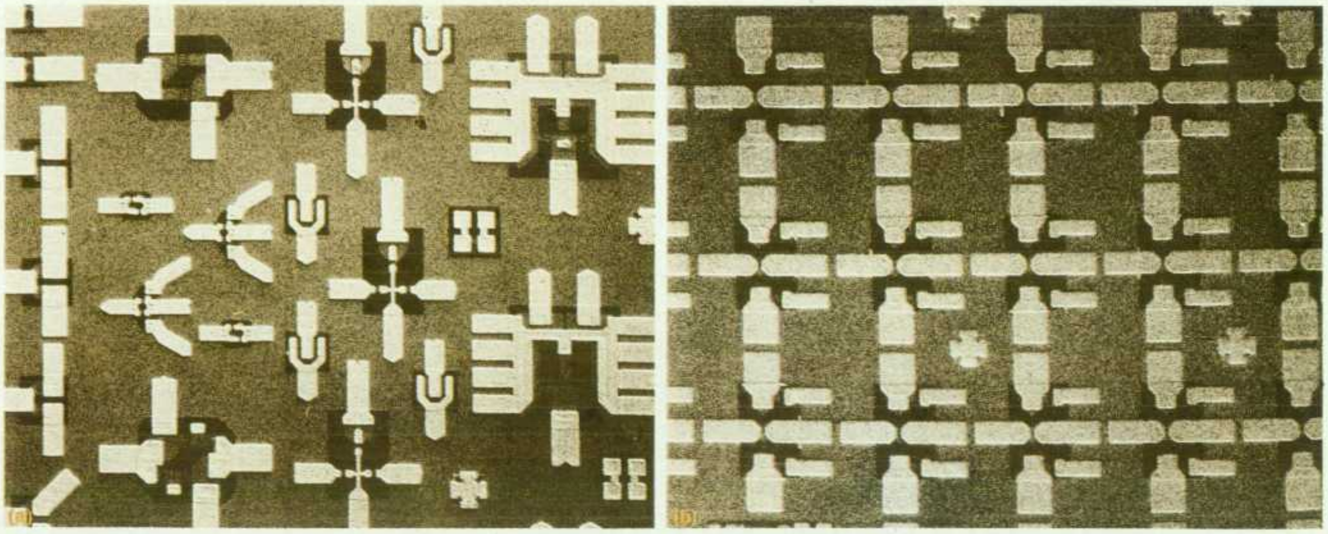
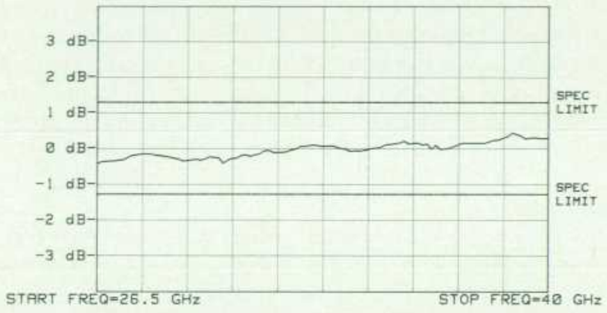


Fig. 5. *(a) Composite mask for prototyping modified barrier diode IC designs. (b) Production mask used for volume production of successful designs.*

26.5-to-40-GHz Waveguide Detector

In 1983 Hewlett-Packard introduced the HP 11664D 26.5-to-40-GHz waveguide detector for the HP 8756A Scalar Network Analyzer. The HP 11664D uses a Schottky diode with a barrier height modified by a sintering process in an integrated detector circuit.[1] This circuit has proven reliable in three years of use in the field. However, manufacturing costs of the product are high because of difficulties in controlling the device fabrication process and losses in assembly caused by static electricity discharges. The new diodes, described in the accompanying article, are significantly easier to manufacture, and losses during assembly are reduced by a more rugged structure.



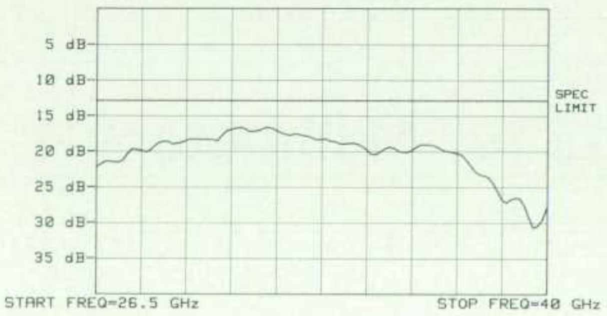


Fig. 1. *Typical return loss (a) and flatness (b) of HP 11664D Detector.*

Experience

To date, manufacturing losses during assembly with the sintered diodes has occasionally run as high as 50%. Under the same conditions, there have been no failures of the new modified barrier diodes. This is attributed to the tenfold reduction in electrostatic susceptibility of these diodes. Slightly lower capacitance has also led to an improvement in the frequency response of the HP 11664D. The lower capacitance allowed a small assembly change that resulted in a 1.0-dB improvement in flatness as a function of frequency. These diodes also have a more predictable voltage response as a function of input power. This allows improved dynamic accuracy for the HP 11664D. Fig. 1 shows typical return loss and flatness of the HP 11664D Detector.

Reference

1. C.C. Chang, et al, "A Zero-Bias GaAs Millimeter-Wave Integrated Detector Circuit," *IEEE Microwave Theory and Techniques Symposium Digest*, 1982.

Herb Upham
Development Engineer
Microwave Technology Division

design custom devices using the components supported in the integrated diode process and optimize the design for their specific application. The design is first placed on a composite mask set (Fig. 5a) with many other custom integrated diodes. This composite mask set is fabricated to supply the various designers with a small number of samples of their design for evaluation. If another design iteration is required, the redesigned device is placed on another composite mask for another cycle of prototype samples. This spreads out the development cost of the custom devices to all the designers using a composite mask. It makes it more affordable to do several design iterations or even try a high-risk design idea. Once a design is optimized and assigned to an instrument project, it is stepped onto a dedicated mask set (Fig. 5b) for higher-volume production.

The development of this integrated diode technology was very closely linked with several instrument development projects. The HP 8510A Network Analyzer and the HP 5350A/51A/52A Frequency Counters[1] rely on GaAs integrated samplers for down-conversion. The design of the HP 11970/71 Harmonic Mixers and the HP 83554A/5A/6A Millimeter-Wave Sources, which both cover the waveguide bands up to 60 GHz, also relies heavily on these Schottky barrier diode ICs. A major milestone, however, was the development of the planar doped barrier diode which enabled HP to design the HP 11970V/W Harmonic Mixers that allow spectrum measurements up to 110 GHz.

Planar Doped Barrier Diode Applications

Special features of these modified barrier diodes make them ideally suited for certain key millimeter-wave products. Before discussing these applications, however, it should be pointed out that in terms of their terminal voltage and current characteristics at low frequencies, these diodes are very similar to conventional pn junction diodes and metal-semiconductor Schottky diodes. In forward bias, they closely follow the ideal diode equation as illustrated in Fig. 3:

$$I = I_o\,[\exp\,((V-IR_s)/nV_t)-1]$$

where I_o is the saturation current, n is the ideality factor (typically 1.2 for these modified barrier diodes), R_s is the series resistance, and V_t is the thermal voltage (0.026V at 300K). In reverse bias, the low-barrier versions of these modified barrier diodes deviate from this ideal equation and exhibit a "soft" breakdown characteristic caused by the variation of barrier height with bias (see Fig. 3).

The most important feature of the planar doped barrier diodes is that their barrier heights are adjustable by controlling the thickness and doping of the epitaxial layers. Changing the barrier height is equivalent to varying the saturation current and hence the turn-on voltage of the diode. Controlling the turn-on voltage is highly desirable for certain applications. One of these applications is the unbiased harmonic mixer (see article on page 22). For this application it is critical that the conduction angle (the number of degrees during each local oscillator (LO) cycle that the diode is on) be controlled precisely, and for a given LO power the conduction angle is controlled by the barrier height.

Another application that requires a low barrier height is the broadband unbiased detector. Diode detectors are widely used at microwave and millimeter-wave frequencies as power meter sensors, scalar network analyzer detectors, and parts of leveling circuits for sources. To work well as an unbiased detector, a diode must essentially be turned on at zero volts. This implies a high saturation current and a low barrier height. A critical specification for these diodes is the video resistance R_v, which can be defined as the incremental resistance of the diode at a bias of zero volts. For most detector applications R_v must be between 500Ω and 5 kΩ. This requires precise control of barrier height and is an ideal application for planar doped barrier diodes.

An important parameter for microwave and millimeter-wave diodes is junction capacitance, which must be kept as small as possible. Like a Schottky diode, a modified barrier diode has no minority charge storage capacitance, so the junction capacitance is quite low. The reverse-bias capacitance of a modified barrier diode is determined primarily by the dielectric constant of GaAs and the thickness of the intrinsic layer and has less variation with bias than Schottky diodes. The typical junction capacitance of a modified barrier diode with an active area of 40 μm^2 is only 25 femtofarads at a bias of zero volts, and with an effective series resistance of about 20 ohms, these diodes work well in detector applications at frequencies up to 110 GHz.

The modified barrier diodes are remarkably rugged devices. Physically, they have good beam-lead pull strengths of over 5 grams and a passivation that allows them to be used in nonhermetic packages. Electrically, they can reliably operate at high current densities; the diodes used in the millimeter-wave mixers described in the article on page 22 are rated to operate continuously at 16 dBm of LO power, which corresponds to average currents through the diodes of over 16 mA. These diodes are also relatively insensitive to damage caused by electrostatic discharge (ESD).

The fact that modified barrier diodes are available with integrated resistors and capacitors enhances their usefulness to the design engineer. Integrating resistors and capacitors with closely matched diodes can, in many cases, improve the performance of a product and lower its cost by decreasing parts count and shortening assembly times. The development and production of these modified barrier diode circuits provides the millimeter-wave designer with a new tool for the design of detectors and mixers up to 110 GHz.

Reference

1. S.R. Gibson, "Gallium Arsenide Lowers Cost and Improves Performance of Microwave Counters," *Hewlett-Packard Journal*, Vol. 37, no. 2, February 1986.

Diode Integrated Circuits for Millimeter-Wave Applications

GaAs diode integrated circuits based on metal-semiconductor (Schottky) or modified barrier diodes have now extended the operating frequency range of small-scale ICs beyond 100 GHz. These circuits, which form the basis for many of HP's new millimeter-wave instruments, are useful for nonlinear and frequency-translation applications.

by Mark P. Zurakowski, Domingo A. Figueredo, Scott S. Elliott, George A. Patterson, William J. Anklam, and Susan R. Sloan

GALLIUM ARSENIDE (GaAs) has become the material of choice for building devices or integrated circuits for operation at frequencies higher than a few gigahertz. Several manufacturers, or foundries, are now offering design and processing services for small-scale to medium-scale integrated circuits on GaAs operating to frequencies as high as 10 GHz and HP has developed a technology for the design and fabrication of diode integrated circuits on GaAs that operate to above 100 GHz. These circuits contain no linear gain elements such as transistors; instead they are composed of diodes, resistors, capacitors, and conductive transmission lines. Their design makes them very useful for signal detection, mixing, multiplication, sampling, power limiting, and other frequency-translating or nonlinear circuits in the millimeter-wave frequency range.

The millimeter-wave region of the electromagnetic spectrum is generally considered to span the frequency range

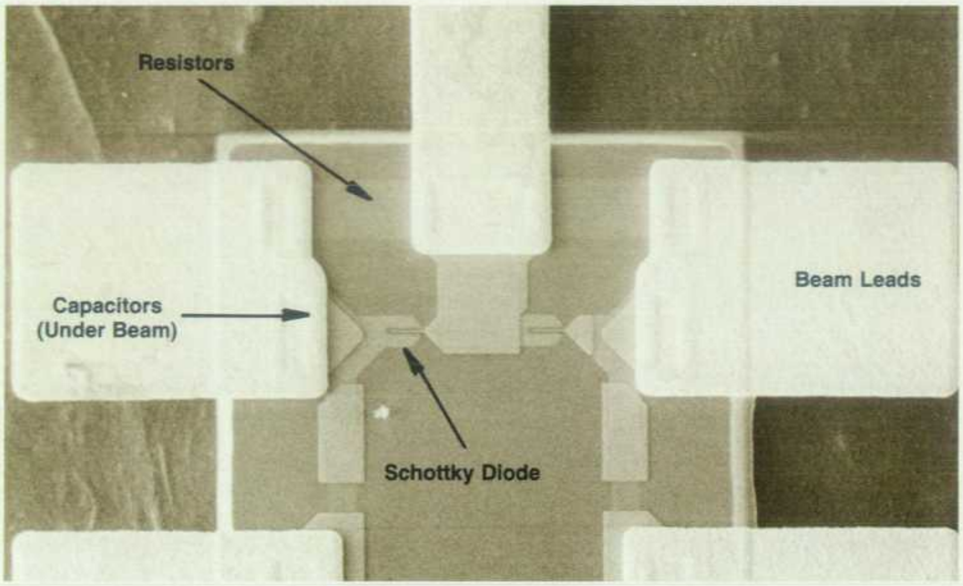


Fig. 1. *Scanning electron microscope (SEM) photomicrograph of a GaAs Schottky barrier diode integrated circuit.*

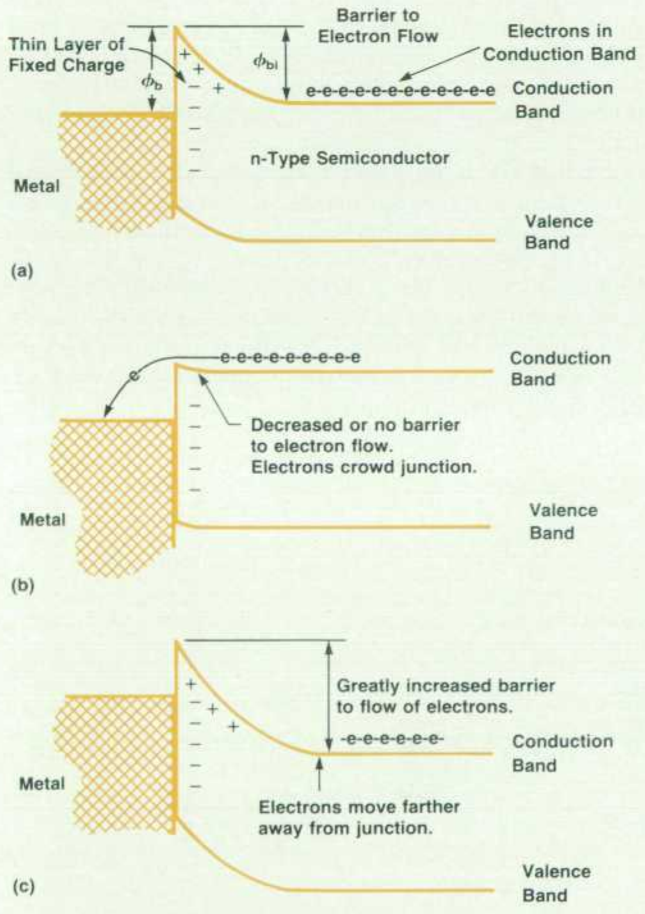

Thin Layer of Fixed Charge
Barrier to Electron Flow
Electrons in Conduction Band
ϕ_{bi}
ϕ_b
e-e-e-e-e-e-e-e-e-e-e-e-e
Conduction Band
n-Type Semiconductor
Metal
Valence Band
(a)
e-e-e-e-e-e-e-e-e
Conduction Band
Decreased or no barrier to electron flow. Electrons crowd junction.
Metal
Valence Band
(b)
Greatly increased barrier to flow of electrons.
-e-e-e-e-e-e-
Conduction Band
Metal
Electrons move farther away from junction.
Valence Band
(c)

Fig. 2. *Energy band diagrams of a metal-semiconductor (Schottky) junction for various bias conditions. (a) Zero bias. (b) Forward bias. The barrier height is decreased and electrons flow easily across the junction. (c) Reverse bias. The barrier becomes more pronounced and the electrons move farther away from the junction.*

of 26 GHz to about 300 GHz, a region in which the wavelength of electromagnetic radiation is of the order of a few millimeters in free space. This region is rapidly gaining importance for uses such as high-speed digital communication, high-resolution radar, military guidance systems, and radio astronomy. Unfortunately, circuit elements such as transistors are available only for the very low end of this frequency band and even small-scale integrated circuits have been virtually unobtainable.

Millimeter-wave frequencies are so high that microwave-type hybrid circuits are difficult to construct. Even the smallest of elements or bond wires can be a significant fraction of a wavelength long and therefore can no longer be modeled as a lumped element. The resulting parasitic capacitances and inductances can completely alter the electrical performance of a hybrid circuit, rendering it unrepeatable, unreliable, and sensitive to its environment. In addition, some applications require symmetry and phase-matching, which are very difficult to achieve in hybrid circuits. Hence, most designs at millimeter-wave frequencies have been done in metal or dielectric waveguide using expensive machined transitions to do the signal processing.

To date, the most important signal processing circuits for millimeter-wave applications are frequency-translating circuits—those circuits used to impress or multiplex lower-frequency information on a millimeter-wave carrier signal, or to shift the information from the millimeter-wave band down to a lower frequency where sophisticated signal processing such as amplification or filtering can be done. These translation functions can be handled very elegantly using diode integrated circuit technology.

A diode integrated circuit is composed of one or more diodes in combination with resistive strips, conductive lines, and capacitive elements patterned in accordance with a set of design rules. Two layers of metallization are presently available, allowing the realization of nonplanar circuit geometries. The completed component is a small pellet of GaAs with beam leads which allow high-strength, low-parasitic bonding to a waveguide or substrate. A photo-

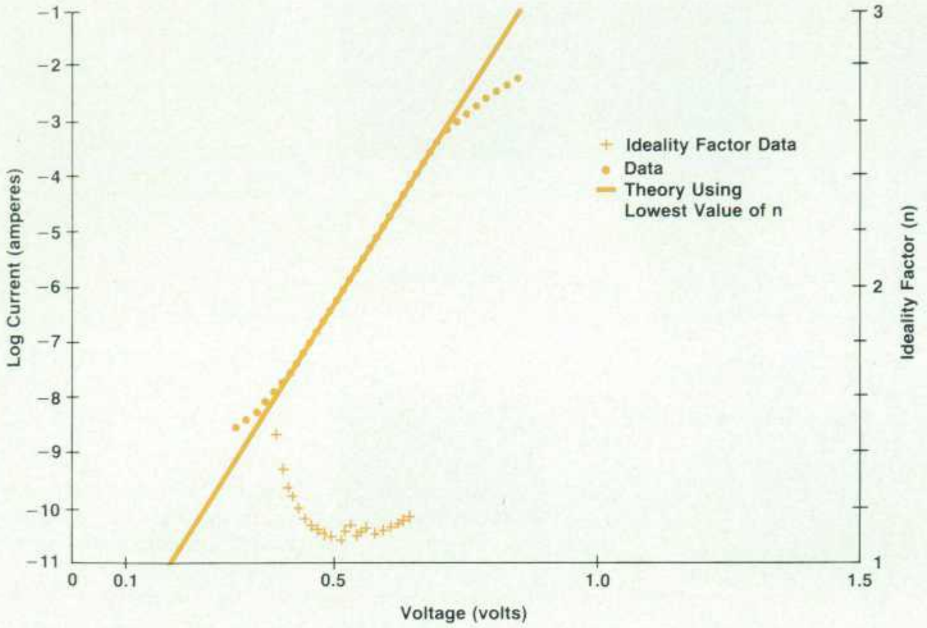


Fig. 3. *Typical Schottky diode I-V characteristic. The straight line illustrates classic behavior. The deviation from the line for actual devices is caused by leakage current for low-current levels and by series resistance for high-current levels.*

micrograph of a sample chip is shown in Fig. 1.

Schottky Barrier Diodes

All of the circuit applications mentioned above require one or more nonlinear circuit elements that operate at millimeter-wave frequencies. GaAs Schottky barrier diodes are a good choice for such high-frequency operation because of the higher velocities at which electrons travel in GaAs combined with the low charge storage effects exhibited by these diodes.

A Schottky barrier is formed whenever a metal makes contact with a semiconductor surface. Ideally, equalization of the metal and semiconductor (n type) Fermi levels causes the transfer of electrons from the semiconductor into the metal, depleting the semiconductor of mobile charge carriers and generating a positive space charge region. This causes the energy bands in the semiconductor (Fig. 2) to bend up at the interface, which creates a barrier potential dependent on the work function of the metal. Besides this fundamental process, if there are large numbers of surface states present on the semiconductor, then instead of electron transfer to the metal, it is electron transfer to the surface states that causes carrier depletion in the semiconductor and band bending near its surface. Fig. 2a shows the negatively charged surface states as minus signs at the metal/semiconductor interface. If the density of surface states is sufficiently large, the barrier potential becomes virtually independent of the metal's work function. GaAs falls into this category; the surface states are generated by damag caused by the metal deposition.

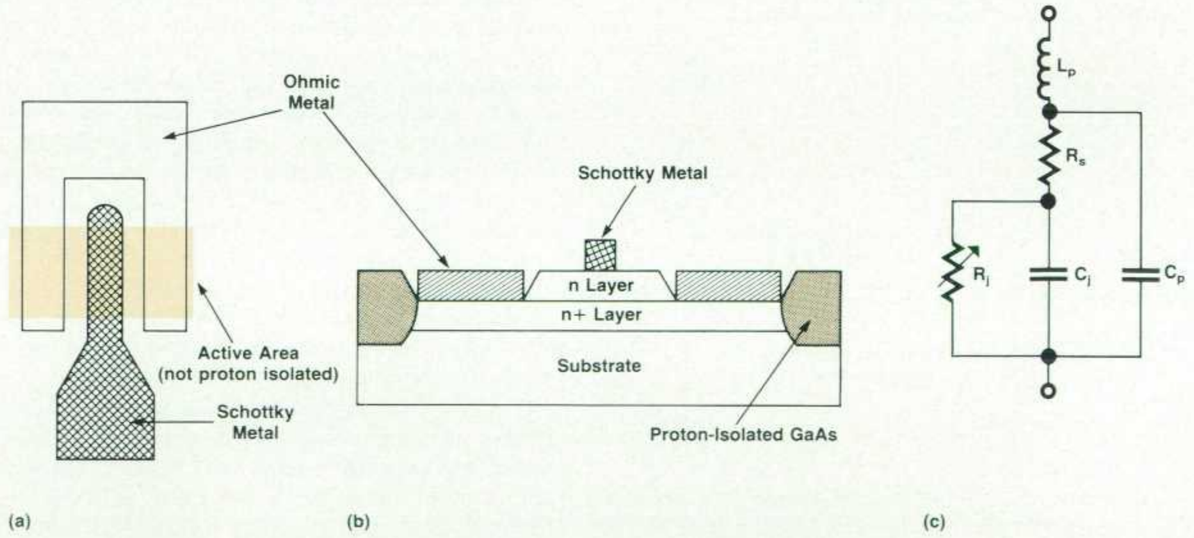


Fig. 4. *Single Schottky diode. (a) Top view. (b) Cross section. (c) Equivalent circuit. R_j represents the nonlinear, exponential part of the circuit, R_s the series resistance, C_j the junction capacitance, C_p the parasitic capacitance, and L_p the beam lead inductance.*

An externally applied voltage will alter the barrier potential for electrons flowing from the semiconductor into the metal. Under forward bias, the conduction band is bent upward, allowing electrons to flow freely into the metal (Fig. 2b). Under reverse bias, the Schottky barrier potential prevents electron flow from the metal to the semiconductor (Fig. 2c). The current flow is governed by thermionic emission of electrons from the semiconductor into the metal with a current-versus-voltage relationship described by:

$$J = J_o[\exp(qV/nkT)-1] \qquad (1)$$

where J is the current density in A/cm^2, J_o is the reverse saturation current density, T is the temperature in K, q is the electron charge in coulombs, V is the applied voltage, and k is Boltzmann's constant. The ideality factor n, typically 1.1 to 1.2, accounts for tunneling currents and other nonideal behavior. J_o is given by the expression: $J_o = A^*T^2 \exp(-q\phi_b/kT)$ where ϕ_b is the barrier height and A^* is Richardson's constant given by $A^* = 4\pi qm^*k^2/h^3$ where m^* is the effective mass of the charge carrier (in this case an electron) and h is Planck's constant. A typical I-V characteristic is shown in Fig. 3 along with a plot of the ideality factor n as a function of bias.

A single Schottky diode and its equivalent circuit are shown in Fig. 4. The nonlinear I-V characteristic of equation 1 is modeled by a voltage dependent nonlinear resistor R_j. The junction capacitance C_j of the diode depletion region is voltage dependent and for uniformly doped epitaxial layers is given by:

$$C_j = C_{jo}/\sqrt{1-(V/\phi_{bi})}$$

and

$$C_{jo} = A\sqrt{q\epsilon N_d/2\phi_{bi}}$$

where ϕ_{bi} is the built-in potential (Fig. 2), C_{jo} is the zero-bias capacitance, A is the area of the junction, N_d is the doping of the epitaxial layer, and ϵ is the dielectric constant of the semiconductor. The parasitic elements are the series resistance R_s and the beam-lead/Schottky-ohmic inductance L_p and capacitance C_p. Major contributors to R_s are contact resistances, the undepleted epitaxial layer resistance, and metal layer resistances. Minimizing the effects of the parasitic elements is extremely important since, with C_j, they set an upper limit to the frequency of operation.

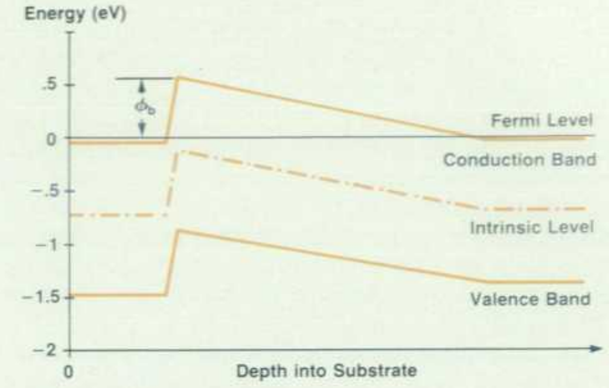
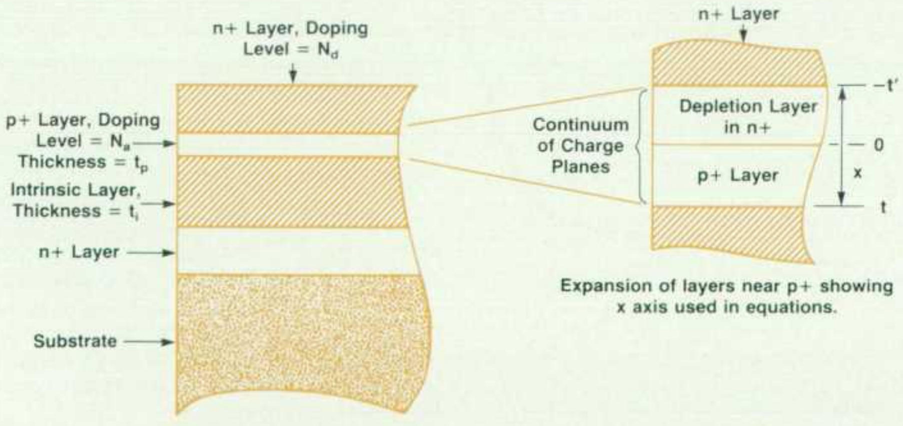


Fig. 5. *Energy band diagram of a modified barrier diode at zero bias.*

Modified Barrier Diodes

In many product applications, high-speed, variable-barrier-height diodes are very desirable because of the added flexibility they provide in choosing optimum response, bias conditions, and power-handling capability. In the case of standard GaAs Schottky diodes, this flexibility is almost nonexistent because the barrier height is virtually independent of the metal system used for the Schottky contact. To address the variable-barrier needs of HP's millimeter-wave customers, a new device called a modified barrier diode has been developed that allows a designer to choose diode barrier heights between 0.2 eV and 1.2 eV. Production of modified barrier diodes requires a technique called molecular beam epitaxy (MBE, see article on page 4) to grow extremely thin, highly doped layers. The major advantage provided by MBE is that the number of fixed charges introduced into the layer determining the barrier height is very accurately controlled by this layer's doping and thickness.

The energy band diagram for a modified barrier diode is triangular in shape as depicted in Fig. 5. It is very similar to the diagram for a Schottky diode and, with good approximation, Equation 1 can also be used for modified barrier diodes. All that is needed is an equation relating doping levels and layer thickness to barrier height. The epitaxial



n+ Layer, Doping Level = N_d
p+ Layer, Doping Level = N_a, Thickness = t_p
Intrinsic Layer, Thickness = t_i
n+ Layer
Substrate
n+ Layer
Continuum of Charge Planes
Depletion Layer in n+
p+ Layer
−t'
0
x
t
Expansion of layers near p+ showing x axis used in equations.

Fig. 6. *Cross section of modified barrier diode fabricated using molecular beam epitaxy (MBE).*

layers forming the modified barrier diode and the variables that will be used in the following simple theory of operation are shown in Fig. 6. Gauss' Law is applied using the following assumptions:

- Depletion approximation. That is, all space-charge regions are fully depleted (no mobile charge carriers) and have abrupt boundaries.
- Infinite planes of charge.
- All dopings are uniform for simplicity of integration.
- Charges are not discrete but spread throughout the layers.
- The bottom layer is far from the barrier and does not enter into the calculation.

The last assumption allows the I-V equation for the modified barrier to have the same form as for the Schottky barrier.

Gauss' Law:

$$\oint \mathbf{E}\cdot\mathbf{n}ds = \frac{1}{\epsilon}\sum_{i=1}^{\infty} q$$

yields $E=\sigma/2$ for an infinite plane of charge where **E** is the electric field, **n** is the normal to the plane, σ is the sheet charge density, and ϵ represents the semiconductor dielectric constant.

For the structure shown in Fig. 6, there is a continuum of charge planes. Thus:

$$E = (2qN_at - 2qN_ax)(1/2\epsilon) \text{ for } 0\leq x\leq t$$

$$E = (2qN_at+2qN_dx)(1/2\epsilon) \text{ for } -t'\leq x\leq 0$$

where N_a is the volume density of acceptor ions, N_d is the volume density of donor ions, t is the thickness of the p+ layer, and t' is the thickness of the n+ layer.

Integrating E from $-t'$ to t and using $t'=t(N_a/N_d)$ from charge balance, we obtain:

$$-\phi_b = (1/2\epsilon)\,[N_a + (N_a^2/N_d)]t^2 \qquad (2)$$

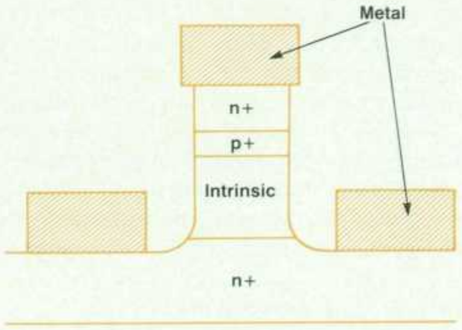


Fig. 8. *Cross section of modified barrier diode. Equivalent circuit and element definitions same as for Fig. 4.*

This equation gives the approximate barrier height of a modified barrier diode and can be substituted directly into Equation 1 for the Schottky diode I-V characteristic.

Comparison of a measured I-V characteristic with a curve predicted by the simple theory presented above is shown in Fig. 7. The reverse characteristics are not modeled well by the simple theory for low-barrier diodes because the simple theory neglects the effect of the electrons in the intrinsic region on the barrier height. In reality the barrier height is lowered slightly in the reverse bias direction and raised slightly in the forward bias direction. This can be taken into account using Stanford University's SEDAN (SEmiconductor Device ANalysis) modeling program. SEDAN is a one-dimensional numerical simulator which solves simultaneously the continuity equations and Poisson's equation for the electrostatic potential and carrier concentrations as a function of space and time. The third I-V characteristic in Fig. 7 is predicted by SEDAN. Notice that it predicts much higher leakage current in the reverse direction and more closely follows the actual I-V characteristic of the measured diode. The I-V characteristics for the simple theory, SEDAN, and the measured diodes all agree for the higher barrier heights since the effect of the

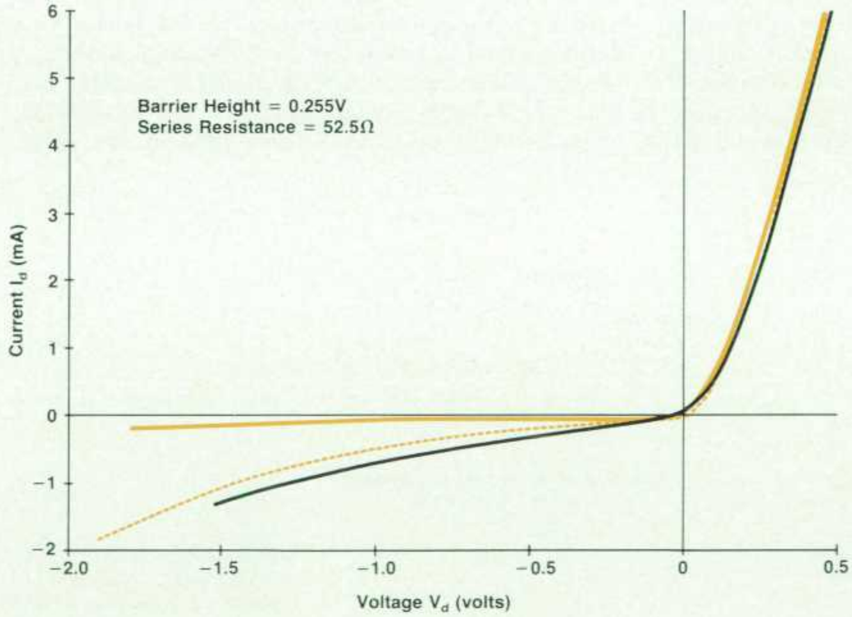
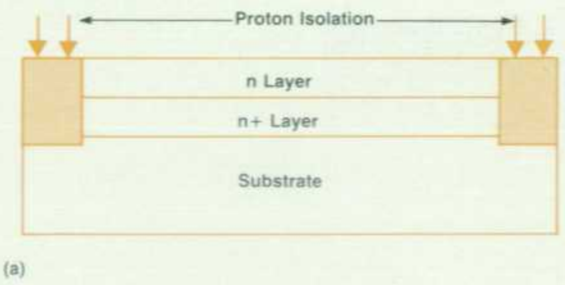
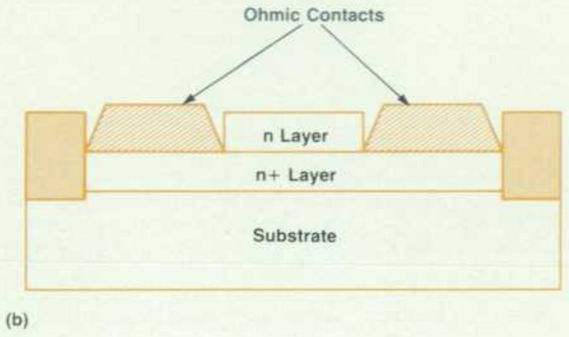
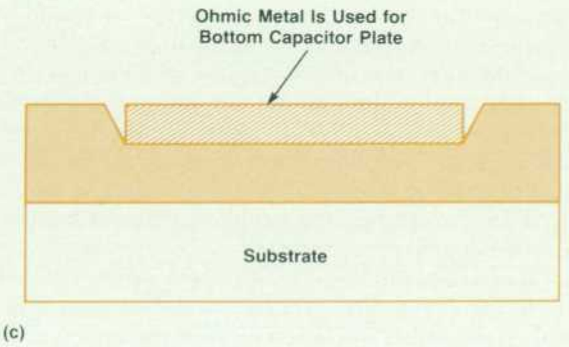


Fig. 7. *I-V characteristic for modified barrier diode. Solid color curve is that predicted by the simple theory discussed in the text, dashed color curve is that predicted by Stanford University's SEDAN modeling program, and the solid black curve is measured data.*







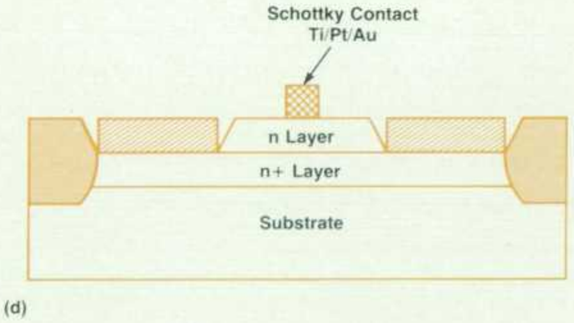


Fig. 9. *Schottky barrier diode fabrication. (a) Nonconductive areas formed by proton isolation (hydrogen ion implantation). (b) Ohmic contacts formed by etching top layer, depositing metal in holes, and alloying metal to bottom layer. (c) The steps for (b) are used at the same time to form the bottom plates of the capacitors in the proton-isolated regions. (d) The Schottky contact is formed using a trimetal process of titanium, platinum, and gold.*

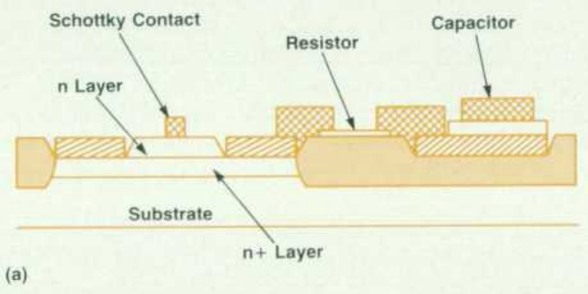
barrier lowering is a much smaller portion of the intrinsic barrier height.

Fig. 8 shows a cross section of a modified barrier diode. Its corresponding circuit model is the same as that shown in Fig. 4 for a standard Schottky barrier diode, but in this case C_j is no longer dependent on the applied voltage, except for the lowest barrier diodes. For a modified barrier diode, C_j depends only on the area and intrinsic layer thickness.

Diode IC Fabrication

Diode integrated circuits are fabricated using standard photolithographic techniques. The substrates are semi-insulating gallium arsenide wafers with reasonably low defect density. In the case of Schottky diode ICs, liquid-phase epitaxy (LPE) is used to grow single-crystal layers of doped GaAs. The first layer consists of about 300 nm of heavily doped n-type GaAs (about $10^{18}/cm^3$) to serve as a bottom conducting surface for the diodes. The top layer is designed to minimize the zero bias junction capacitance while maintaining reasonably low series resistance. We typically choose this layer to be about 300 nm of $10^{17}/cm^3$ n-type material.

The first step in the process is to select and isolate regions of the substrate to be used for diodes. The technique of proton isolation, or ion implantation with hydrogen ions, is used to destroy the conductivity of the epitaxial layers in all areas of the wafer except where diodes are to be located (see Fig. 9a). Holes are then etched through the top layer to the heavily doped n+ layer and metal is deposited in these holes. The metal is alloyed through a sintering step to form a low-resistance (ohmic) contact to the bottom



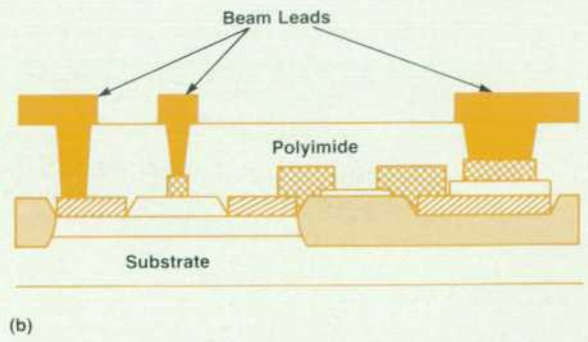


Fig. 10. *(a) Cross section of integrated structure. (b) Structure of (a) with polyimide insulating layer and beam leads.*

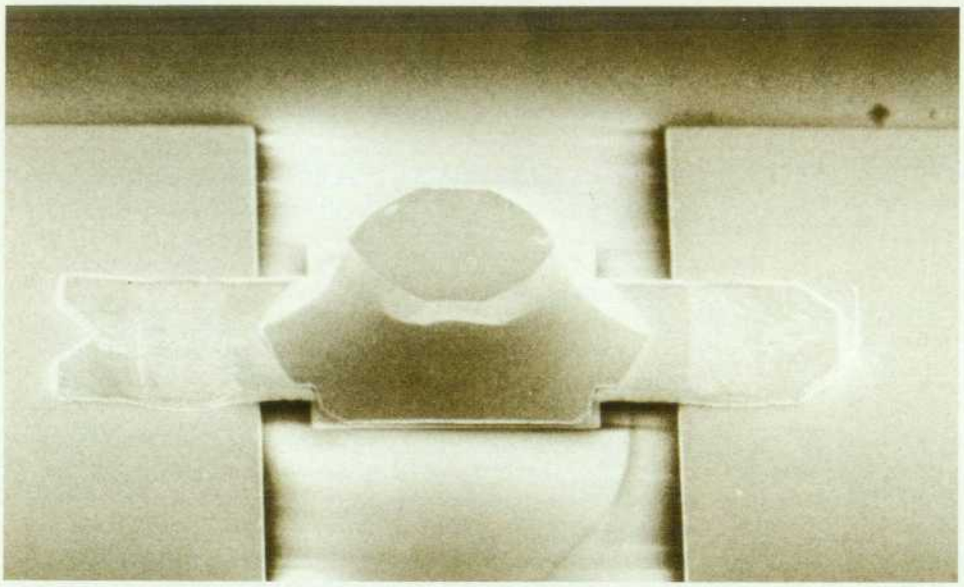


Fig. 11. *Photomicrograph of completed diode IC chip mounted upside down on a microwave carrier.*

layer, and thus to the bottom of the diodes (Fig. 9b).

The same etch and deposition steps used to form the ohmic contacts are also used to form the bottom plates for capacitors as shown in Fig. 9c. The GaAs etch is done in a proton isolated region of the substrate, so that the metal does not form an ohmic contact in this case. Resistive strips of tantalum nitride are formed next on the semi-insulating region of the GaAs by field deposition and selective etching. The resistivity of these films is nominally 50Ω/□. The capacitor dielectric, consisting of about 100 nm of silicon nitride, is patterned on top of the metal bottom plate to yield a nominal capacitance density of 0.67 fF/μm^2.

The critical Schottky barrier contact is now formed. A trimetal system of titanium-platinum-gold is used to form a stable and highly reliable contact that exhibits low parasitic resistance and a low potential for electromigration. The titanium adheres well to GaAs and does not easily form alloys with it. The gold is highly conductive and extremely resistant to corrosion or electromigration, but it does interdiffuse easily with both titanium and GaAs. Therefore, a thin layer of platinum is used to separate the titanium from the gold and adheres very well to both.

Photoresist is first applied to the whole wafer and windows are defined where the metals will eventually be located. The three metals are applied sequentially in the same evaporation system, and the unwanted regions of metal are lifted off by dissolving the photoresist underneath. This process allows very good control of metal strip geometries down to around 800 nm, which is near the lower limit for optical lithography. A cross section of the resulting Schottky contact finger is shown in Fig. 9d. The same metal deposition and lift-off procedure is used to form the top plates of the capacitors and to make connections between all of the various circuit elements (see Fig. 10a).

The entire circuit is passivated and planarized by a 1-μm

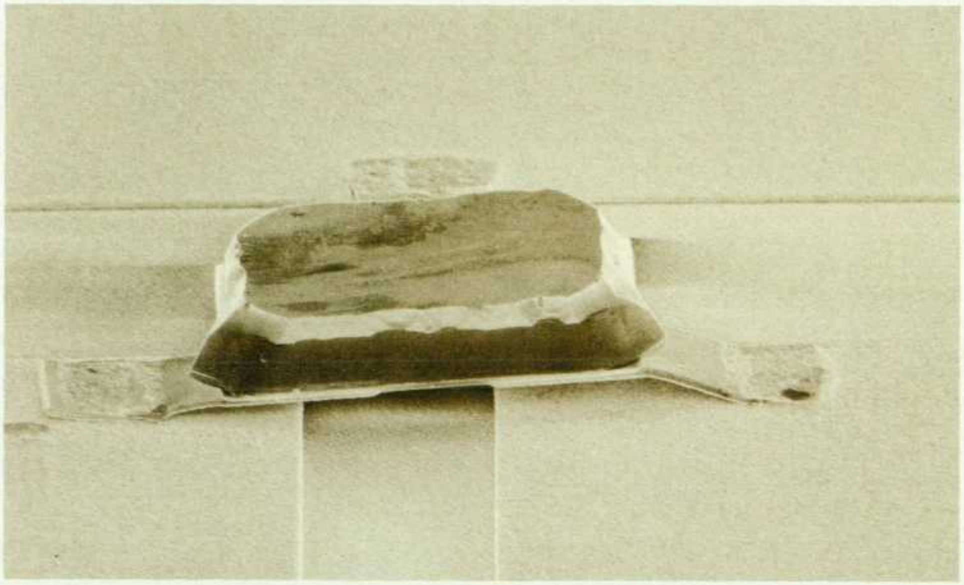


Fig. 12. *Photomicrograph of single beam-lead diode mounted on a coplanar waveguide structure for nonlinear and linear testing at microwave frequencies.*

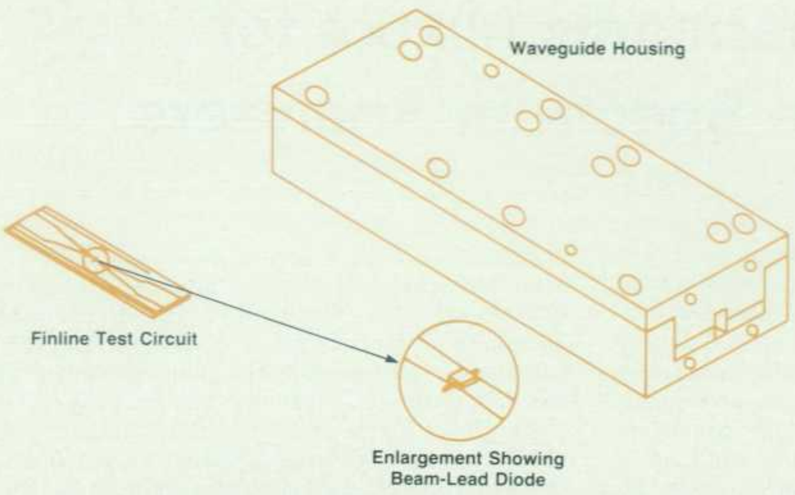


Fig. 13. *Diagram of finline test package used to test beam-lead diodes at millimeter-wave frequencies.*

thick coating of polyimide which is spun on and cured. Contacts to the outside world are made by selectively etching vias (holes) through the polyimide to the desired metal areas. Added areas of metal are photolithographically formed on top of the polyimide and plated to several micrometers thick with gold to form beam leads or a second contact metal layer (see Fig. 10b).

The wafer is mounted circuit-side down on a quartz wafer and backlapped carefully to a thickness of 50 μm. A backside alignment to the patterned front surface is performed using an infrared aligner to expose photoresist which defines etch channels. A chemical etchant is then used to separate the chips and expose the gold beams. A completed chip, mounted upside down on a microwave carrier, is shown in Fig. 11.

Fabrication of modified barrier diode integrated circuits is quite similar except for the addition of a mesa etch to define the diodes and replacement of the Schottky metal deposition with a low-resistance top metal deposition and alloy.

GaAs Diode IC Characterization

During fabrication the wafers are monitored by periodically checking standard test patterns included on each mask set. The patterns are probed and dc-to-1-MHz tests are done as the IC is fabricated. From these tests we can evaluate the quality of the circuit elements at various steps in the process. Upon completion of the fabrication process, each diode IC is tested with an automated test system which screens the ICs for specified dc parameters.

Some applications require testing of parameters at microwave and millimeter-wave frequencies. This testing is more difficult and is usually done on a sample basis. A fixture useful for both linear and nonlinear testing at microwave frequencies is shown in Fig. 12 in which a single beam-lead diode is mounted on a coplanar waveguide structure. A network analyzer can be used to measure the four s-parameters of this configuration. This data can be compared to that predicted by a linear model of the device to find the element values in the model that most closely fit. Nonlinear parameters such as harmonic generation efficiency or series power limiting can also be measured.

A fixture that has proven useful for millimeter-wave frequencies is shown in Fig. 13. This circuit consists of a finline substrate in a waveguide housing. The finline is composed of a thin film of gold on a fused silica or sapphire substrate that tapers to a narrow gap at the center. It provides a broadband, low-VSWR transition from the waveguide mode to a finline mode. A diode or diode integrated circuit can be mounted at the neck of this circuit as shown. Again, full two-port s-parameter measurements can be made over the waveguide band using a network analyzer, or nonlinear measurements can be performed.

Conclusion

Hewlett-Packard has developed technologies for fabrication and design of custom diode integrated circuits that operate beyond 100 GHz. These ICs form the basis for many of HP's new millimeter-wave instruments, but are not available as separate products.

Acknowledgments

Many of the original concepts of GaAs ICs using Schottky barrier diodes were developed by C.C. Chang, Gary Roberts, and Dennis Lynch. The process development work was partially due to the efforts of Nina Maddox, Mimi Planting, and Mary Stone. Much of the characterization work was done by Eric Ehlers, Tim Bagwell, and Virginia Cox. The transition to production was made possible by Mark Buswell, Janice Pryst, Sig Johnsen, and Selena Vella.

Unbiased Subharmonic Mixers for Millimeter-Wave Spectrum Analyzers

by Robert J. Matreci

EXTERNAL WAVEGUIDE MIXERS are used to extend the frequency range of a microwave spectrum analyzer beyond its frequency limit for a coaxial connector input. Since a microwave analyzer's local oscillator (LO) has a typical range of 2 to 6.2 GHz, the mixers must operate on higher-order harmonics of the LO. To maintain the analyzer's amplitude measurement accuracy, individual calibration and flat frequency response of the mixer are mandatory. If there are no electrical or mechanical adjustments to be made as a function of frequency (e.g., bias current or backshort position), then wideband and automated spectral measurements are possible. High burnout level and ruggedness are required for reliability.

Even-Harmonic Mixer Design

The HP 11970 family of waveguide mixers[1] is designed for use with the HP 8566B Spectrum Analyzer or the HP 70000 Modular Measurement System (Fig. 1). Six waveguide bands from 18 to 110 GHz are covered.

In the schematic of Fig. 2, a microstrip diplexer separates the analyzer's LO from the intermediate frequency (IF) mixing product. The RF signal enters via a tapered waveguide section which is terminated by the diodes. The antiparallel-pair diode configuration[2] gives improved conversion loss over single-diode circuits since the conductance waveform symmetry produces mixing products caused only by even harmonics of the LO. Thus, less energy is lost to unused harmonics. The back-to-back diode connection also increases electrostatic voltage damage resistance.

The diodes, Fig. 3, are a monolithic GaAs pair[3] produced by HP's Microwave Technology Division. A 250-μm-diameter loop connects the two diodes and contains the even-order current harmonics. The zero-bias capacitance C_{jo} is 13 fF per diode and the series resistance is 24Ω. A polyimide layer passivates the diodes and contributes to the excellent beam-lead pull strength of 15 grams.

A key factor in realizing constant conversion loss versus frequency is the control of the odd harmonics of the LO produced in the diodes. Although the conductance (mixing) waveform contains only even harmonics of the LO, the input current contains only odd harmonics. If these odd harmonics propagate away from the diode's location and then reflect back from any internal or external element, the desired even harmonic can be severely weakened by the reentering signals. In previous harmonic mixers, these destructive interference effects led to numerous spike-like increases of conversion loss versus frequency, and a change in bias current or backshort position was required to move

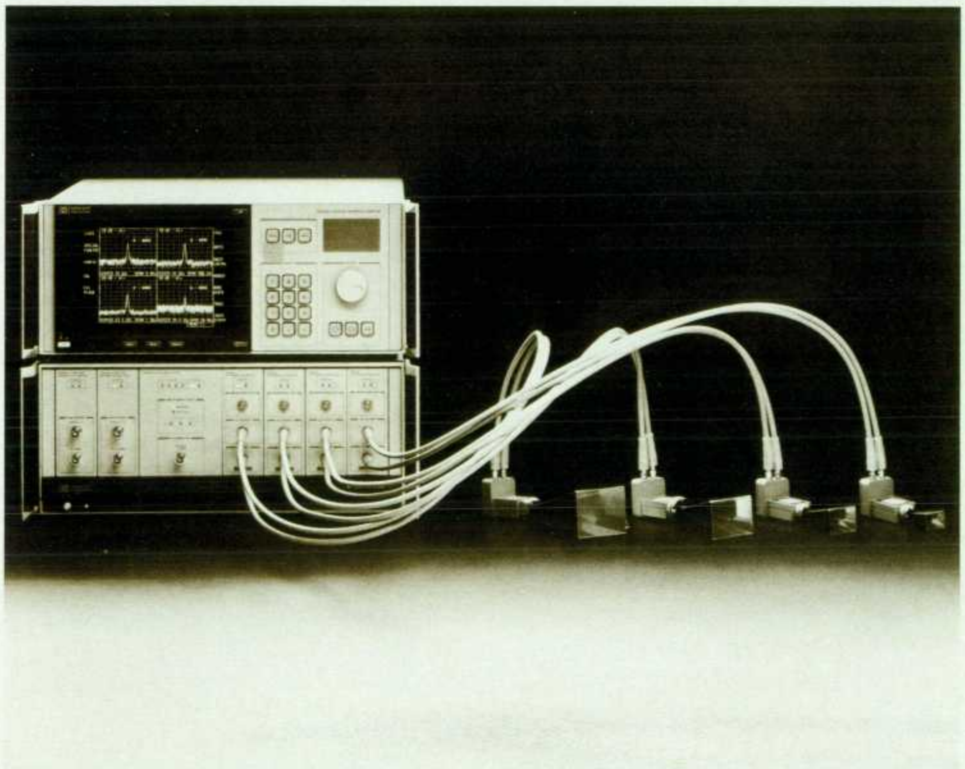


Fig. 1. *Four HP 11970 waveguide harmonic mixers connected to the HP 70000 Modular Measurement System.*

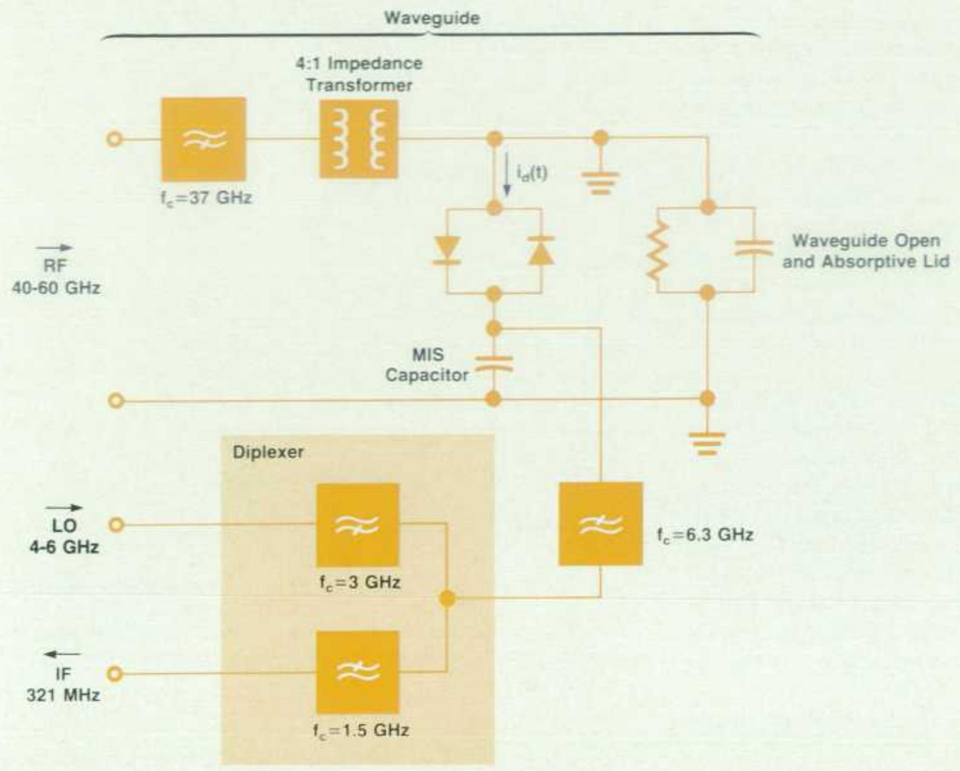


Fig. 2. *Schematic of U-band (40-to-60-GHz) tenth-harmonic mixer.*

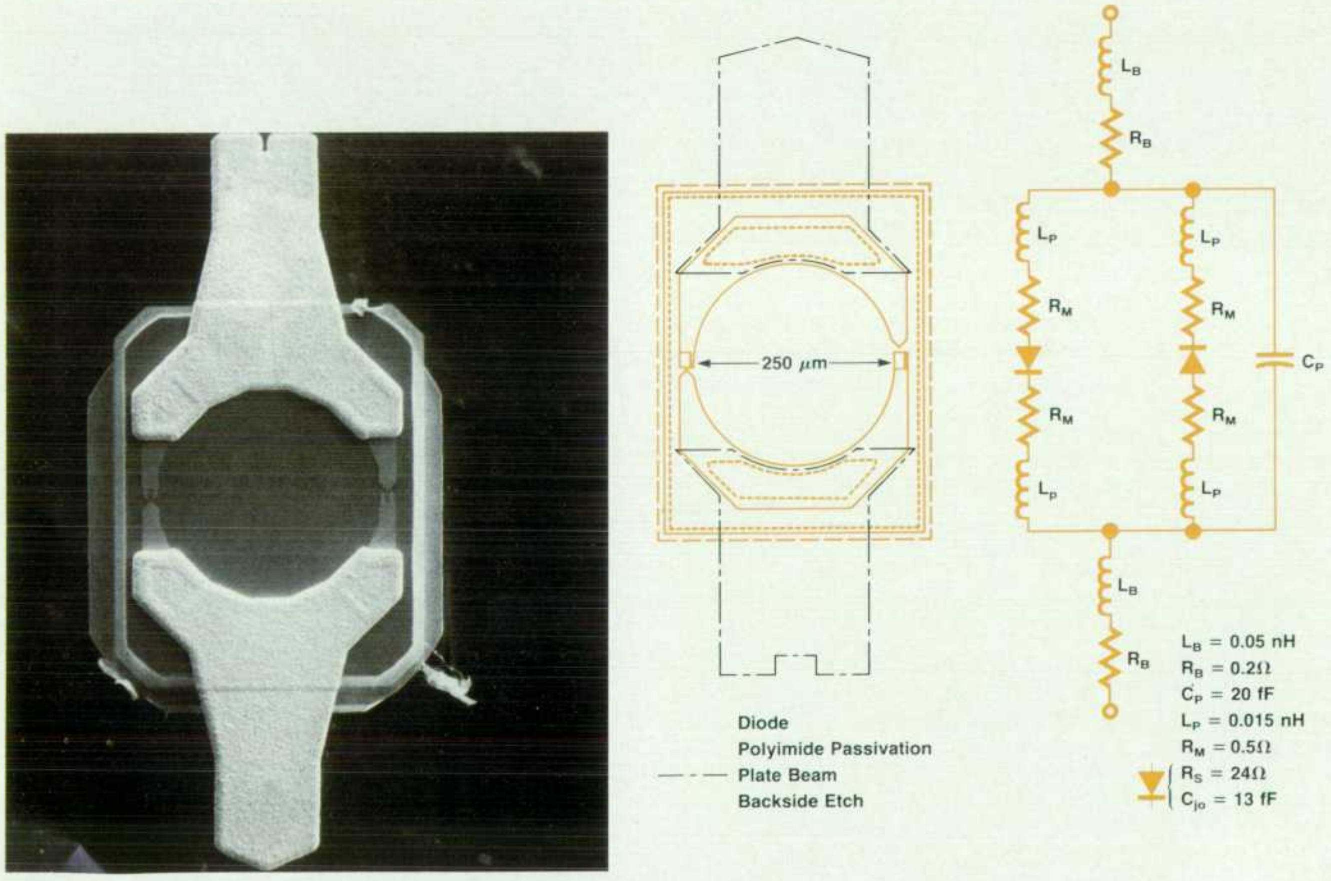


Fig. 3. *Beam-lead GaAs diode. (left) Photomicrograph of diode. (center) Layout of diode. (right) Schematic representation of diode.*

the reconstruction phase away from cancellation.

For the HP 11970 Mixers, odd-harmonic reflection from the diplexer and the LO source system are eliminated by the short produced at the diodes by the lumped first element of the 6.3-GHz low-pass filter (LPF, see Fig. 2). This metal-insulator-semiconductor (MIS) capacitor is also in series with the RF signal, so it must maintain its shorted condition throughout the RF band so that the entire RF voltage can appear across the diode. The capacitor's pad dimensions are 75 μm by 75 μm and its silicon-dioxide (SiO_2) dielectric layer thickness is 440 nm. These small sizes push the capacitor's self-resonance frequency to more than 110 GHz.

The waveguide 37-GHz high-pass filter (Fig. 4) follows a symmetrical $\exp(\cos^3\theta)$ taper,[4] and prevents odd LO harmonics from reflecting from the out-of-band source mismatch of the system being measured. Early in the development project, a rectangular-to-double-ridge taper was tried, but the large variation in cutoff frequency along the taper was itself a source of odd-harmonic reflection. Other especially troublesome sources are components such as bends, twists, and even the terminated arms of directional couplers, all of which present significant reflections at or below their cutoff frequency.

The high-pass filter's width taper is integrated into a

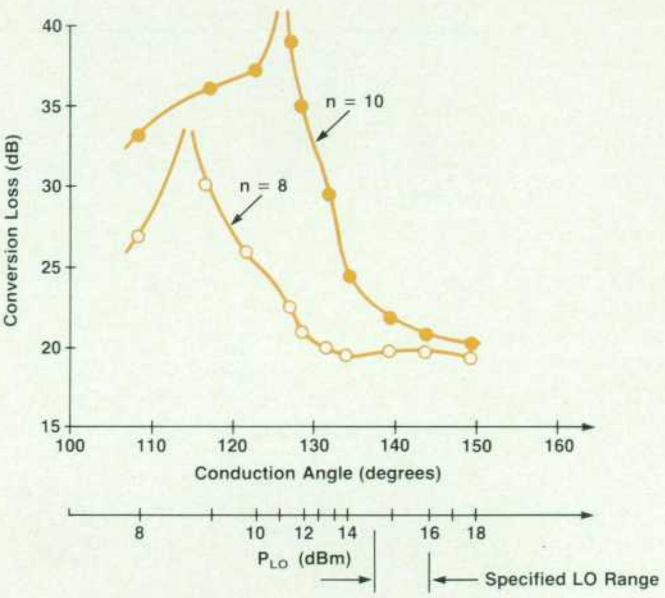


Fig. 5. *Conversion loss versus conduction angle and LO power for the eighth and tenth harmonics of a high-barrier diode.*

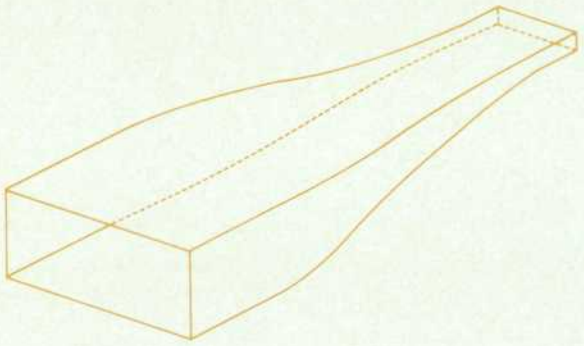
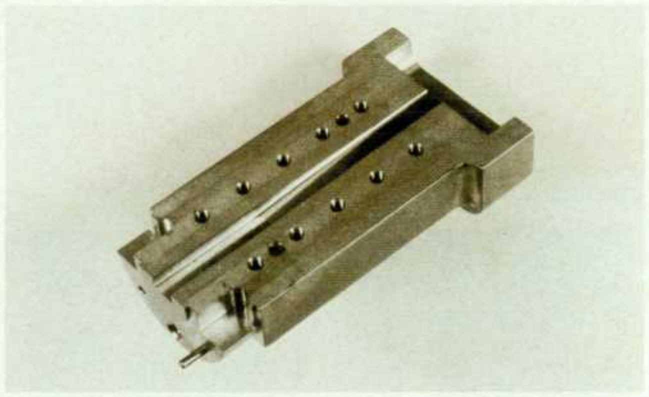



Fig. 4. *A special taper (left) in the waveguide (right) acts as a 37-GHz high-pass filter and 4:1 impedance transformer. The taper is formed by numerically controlled milling machines according to equations describing the taper's dimensions.*

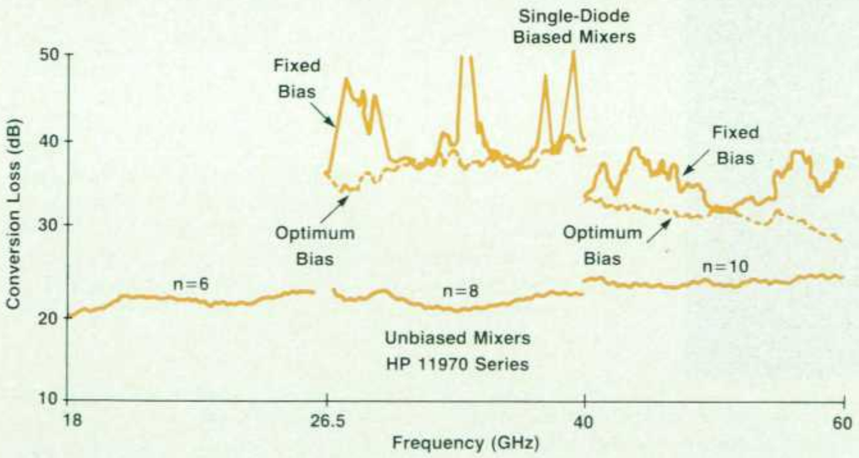
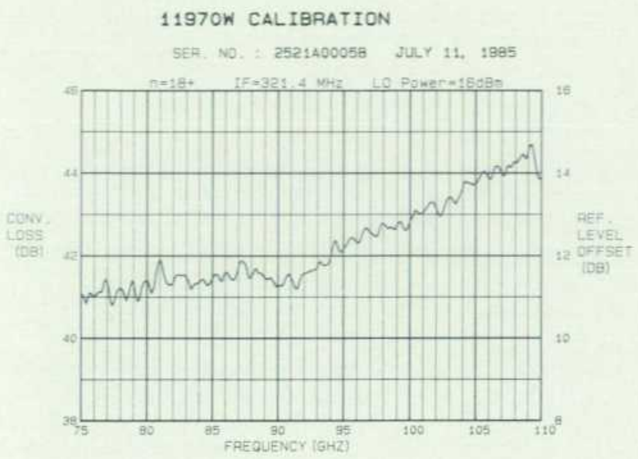
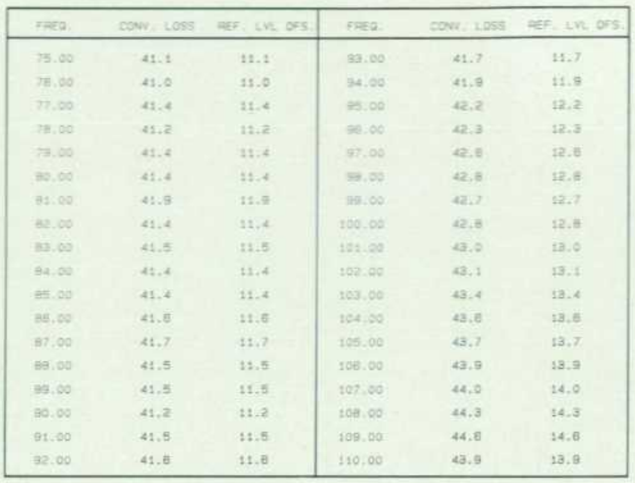


Fig. 6. *Conversion loss versus frequency for the 18-to-60-GHz range.*





FREQ.	CONV. LOSS	REF. LVL OFS.	FREQ.	CONV. LOSS	REF. LVL OFS.
75.00	41.1	11.1	93.00	41.7	11.7
76.00	41.0	11.0	94.00	41.9	11.9
77.00	41.4	11.4	95.00	42.2	12.2
78.00	41.2	11.2	96.00	42.3	12.3
79.00	41.4	11.4	97.00	42.6	12.6
80.00	41.4	11.4	98.00	42.6	12.6
81.00	41.9	11.9	99.00	42.7	12.7
82.00	41.4	11.4	100.00	42.8	12.8
83.00	41.5	11.5	101.00	43.0	13.0
84.00	41.4	11.4	102.00	43.1	13.1
85.00	41.4	11.4	103.00	43.4	13.4
86.00	41.6	11.6	104.00	43.6	13.6
87.00	41.7	11.7	105.00	43.7	13.7
88.00	41.5	11.5	106.00	43.9	13.9
89.00	41.5	11.5	107.00	44.0	14.0
90.00	41.2	11.2	108.00	44.3	14.3
91.00	41.5	11.5	109.00	44.6	14.6
92.00	41.6	11.6	110.00	43.9	13.9

Fig. 7. *Each HP 11970 Mixer is individually calibrated for conversion loss across its full frequency band. Data is entered into the mainframe spectrum analyzer to provide corrected amplitude measurements in the millimeter-wave bands.*

modified exponential waveguide height taper,[5] which in a minimum length lowers the waveguide impedance by a factor of 4 to match the time-average impedance of the diode.

The diodes are thermosonically bonded across the reduced waveguide opening height (0.5 mm), one terminal to the bottom wall and the other onto the MIS capacitor attached with epoxy adhesive to the top wall.

The relatively large local oscillator drive level required (14.5 dBm to 16 dBm) is a result of the high diode barrier height (V_f at 1 mA = 0.73V) and the need for a large conduction angle. Fig. 5 shows that only conduction angles greater than 140 degrees will avoid nulls in the eighth or tenth harmonic of the conduction waveform.

The eighteenth harmonic is required to reach the 75-to-110-GHz band, but the conduction angle null occurs close to the standard 14.5-dBm-to-16-dBm LO power range. The conduction angle can be altered by using more LO power or reducing the barrier height of the diodes. Since the various harmonic mixers are required to be compatible with the LO drive available in the spectrum analyzer system, the medium-barrier diode (V_f at 1 mA = 0.28V) described in the articles on pages 10 and 14 was developed. This diode provides the same type of flat response and insensitivity to LO variation at 110 GHz (n = 18) as the high-barrier diode does at 18 GHz (n = 6).

Performance Measurements

Conversion losses for each waveguide band from 18 to

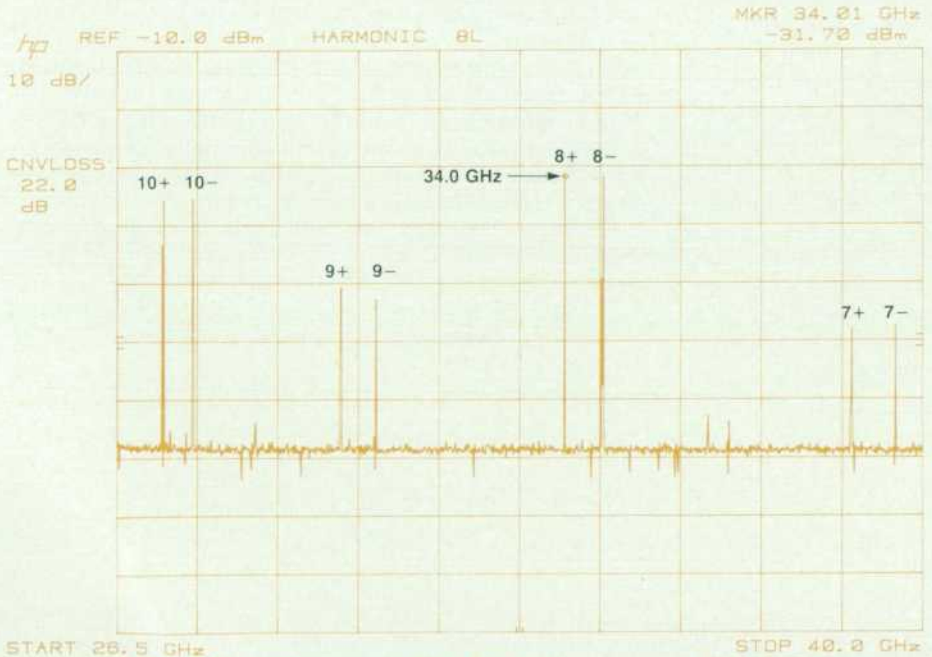


Fig. 8. *A-band spectrum analyzer display of a single input frequency at 34 GHz for mixer with 321-MHz LO. Close grouping of image responses (N− harmonics) and odd-order suppresion are shown.*

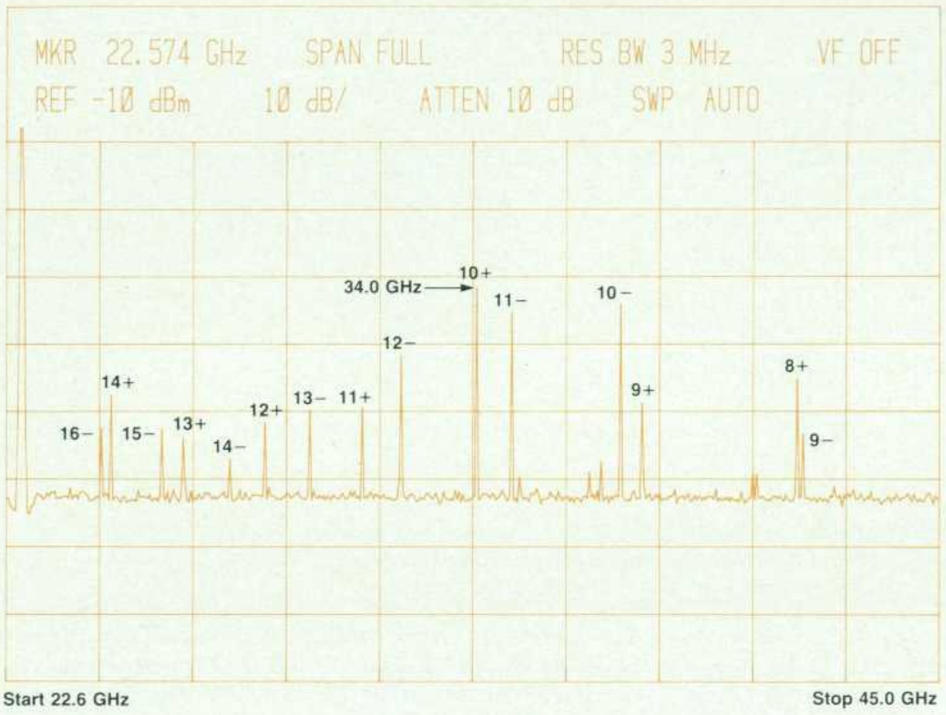


Fig. 9. *A-band spectrum analyzer display for older 2.05-GHz IF harmonic mixers. Note that the image responses (N− harmonics) interleave with desired N+ harmonic responses.*

60 GHz are shown in Fig. 6. To demonstrate the improved flatness and sensitivity of the unbiased HP 11970 Mixers, the plot for a single-diode biased mixer is shown for the cases of bias optimized at each frequency and a fixed bias optimized at only the center of the band. An example of the calibration chart that accompanies each HP 11970 Mixer is shown in Fig. 7 for the 75-to-110-GHz band.

The suppression of odd-harmonic mixing products and the choice of a relatively low IF frequency (321 MHz versus 2.05 GHz used previously) aids in reducing the spectrum analyzer's display clutter. Because no RF preselectors are available for these bands, responses from harmonics adjacent to the one desired also appear. The antiparallel-pair configuration suppresses the adjacent odd responses and the low IF frequency causes the wanted N+ harmonic response and its N− harmonic image response to be grouped closely with no interleaving. The resulting full band display caused by a single input frequency has a certain "order" to it (see Fig. 8). Fig. 9 shows the same display for the earlier 2.05-GHz-IF mixers.

Acknowledgments

Kudos to C.C. Chang, Gary Roberts and Dennis Lynch for the original high-barrier diode. The medium-barrier diode group, whose articles appear in this issue, did an excellent and fast job. Vonnie Anderson aided with computer modeling. Jimmie Yarnell designed the mechanical package and processes (besides creating valuable artists' renditions). Helen Ericson and Dorothy Yanez handcrafted the prototypes. Vinh-Khuong Nguyen, Ken Lew, and Robert Charlton molded the efforts into a producible product with automated testing. Project managers Frank David and Roger Stancliff reviewed the design and Roger Kahn and Ron Rausch marketed the devices.

References

1. R. Matreci and F. David, "Unbiased, Subharmonic Mixers for MMW Spectrum Analyzers," *1983 IEEE MTTS Symposium Digest*, pp. 130-132.
2. M. Cohn, J. Degenford, and B. Newman, "Harmonic Mixing with an Antiparallel Diode Pair," *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-23, August 1975, pp. 667-673.
3. G. Anderson, et al, "GaAs Beam Lead Antiparallel Diodes for MMW Subharmonic Mixers," *IEEE International Electron Devices Meeting*, Washington D.C., December 1981.
4. C. Tang, "Nonuniform Waveguide High-pass Filters with Extremely Steep Cutoff," *IEEE Transactions on Microwave Theory and Techniques*, Vol. MTT-12, May 1964, pp. 300-309.
5. F. David, "Analysis and Synthesis of Exponentially Tapered, Nonuniform Transmission Line Impedance Transformers," MSEE Thesis, Oregon State University, 1975.

Authors

November 1986

4 — Molecular-Scale Engineering

Douglas M. Collins

Doug Collins is an alumnus of Montana State University (BSEE 1971) and Stanford University (MSEE 1972 and PhD EE 1977). With HP Laboratories since 1979, he has held several staff and management positions and is currently a department manager in the high-speed devices laboratory. His department is responsible for optical and electrical characterization of III-V compound semiconductors in addition to the MBE and organometallic vapor phase epitaxial growth of these materials. Before coming to HP he worked on MBE development for III-V semiconductors at Varian Associates. He's the author or coauthor of over 25 papers on various topics related to materials research and is active in several professional societies, including IEEE, the American Physical Society, the American Vacuum Society, and TMS/AIME. Doug lives in Sunnyvale, California, far away from his place of birth—Belo Horizonte, Brazil. He and his wife have two sons. When not at work, he keeps busy with home improvements and woodworking and likes camping with his family, skiing, and amateur radio.

10 — HP Diode Operation

Sigurd W. Johnsen

Sig Johnsen is applications engineering manager for GaAs diodes, FETs, and silicon bipolar devices at HP's Microwave Technology Division. He has held a number of engineering positions since joining HP in 1979. He's a 1977 graduate of the Rochester Institute of Technology (BS physics) and was a test equipment engineer at General Electric Aerospace before coming to HP. Born in White Plains, New York, Sig now lives in Santa Rosa, California. He and his wife have two sons. His interests outside work include skiing, hiking, backpacking, and gardening.

Eric R. Ehlers

Eric Ehlers came to HP in 1979 and has been an applications engineer for GaAs FETs and diode integrated circuits. He's currently an R&D engineer working on millimeter-wave products. Born in San Diego, California, he studied biology at the California Institute of Technology (BS 1973) and worked as a biochemist at Scripps Clinic. He continued his studies at the University of California at Santa Barbara (BSEE 1977 and MSEE 1979). He's the author or coauthor of four papers on microwave measurements and circuit design. Eric and his wife live in Santa Rosa, California and have two sons. He's involved in youth soccer and teaches courses at a local college. Other interests include karate, hiking, and science fiction.

Douglas A. Gray

A New York City native, Doug Gray completed his studies for a BSEE degree from the Polytechnic Institute of New York in 1960 and for an MSEE degree from Stanford University in 1962. With HP since 1960, he was R&D manager for the Microwave Technology Division before his recent retirement. He has contributed to the development of microwave modulators, signal generators, and network analyzers and sweepers, among other products. He has written several papers on microwave components and is named author on a patent for a Gunn oscillator design. He and his wife live in Santa Rosa, California and have four children. One of his sons is an HP engineer. Sailing is his favorite recreational activity. He sails his 42-foot cutter around San Francisco Bay and recently sailed to Hawaii. He also enjoys photography and scuba diving.

14 — Diode Integrated Circuits

Susan R. Sloan

With HP since 1985, Susan Sloan is an R&D engineer at the Microwave Technology Division and has contributed to the development of fabrication processes for high-frequency diodes. She was born in Madison, Wisconsin and attended the University of Wisconsin at Madison. Her BS degree in physics and English was awarded in 1981. She continued her studies at the University of Colorado, completing work for an MS degree in physics in 1984. Susan lives in Petaluma, California and enjoys skiing, bicycling, hiking, reading, and knitting. She's also a Beatles music fan.

Mark P. Zurakowski

A project manager at the Microwave Technology Division, Mark Zurakowski has held a number of engineering positions at HP since 1978. His most recent project work has been on GaAs diode integrated circuits. He is named inventor for a patent application on a modified barrier diode and is author or coauthor of three papers related to device design and processing, semiconductor physics, and millimeter-wave devices. Mark has a 1978 BS degree in mechanical engineering and materials science from the University of California at Davis. He also earned an MS degree in materials science and engineering from Stanford University in 1982. Born in Lansing, Michigan, Mark is a resident of Santa Rosa, California. He's married and has three children. He enjoys traveling, camping, and hiking with his family and likes woodworking and sports.

William J. Anklam

Bill Anklam was born in Columbus, Ohio but grew up in Muscatine, Iowa. He's a graduate of Iowa State University (BSEE 1982) and of Stanford University (MSEE 1983 and EE 1986). Josephson devices were the subject of his thesis research. With HP since 1985, he has been responsible for the design and characterization of new devices and circuits for millimeter-wave applications. He's coauthor of four papers on Josephson junction fabrication and characterization and is interested in device physics, ultrafast phenomena, picosecond electronics, and superconductivity. A resident of Santa Rosa, California, Bill is a runner and likes backpacking, motorcycling, softball, golf, and music. He's also a voracious reader.

Domingo A. Figueredo

Domingo Figueredo joined HP in 1981 upon getting his MSEE degree from the University of California at Santa Barbara. His BSEE degree was awarded in 1979 by the Florida Institute of Technology. His contributions at HP include work on bipolar transistors, on GaAs MES-

FETS, and on modified barrier diode technology. He's presently a project manager for millimeter-wave transistors. His work on planar doped transistors is the subject of a patent application and he's the author or coauthor of four papers on microwave semiconductor device theory, processing, and applications. His other professional experience includes research on molecular beam epitaxy at the University of California. Born in Barcelona, Venezuela, Domingo now lives in Rohnert Park, California. He's married and has five children. His leisure activities include weight training and jogging.

Scott S. Elliott

Scott Elliott is the section manager for the development of integrated diode circuits and other technology products at HP's Microwave Technology Division. Born in Aberdeen, Washington, he studied electrical engineering at the University of California at Berkeley, receiving his BS degree in 1969 and his MS degree in 1971. After working for four years on microwave components, he continued his studies at the University of California at Santa Barbara and completed work for a PhDEE in 1978. With HP since 1978, Scott has been a development engineer, a project manager, and a section manager. He was one of the developers of the first surface-acoustic-wave resonators and low-loss filters to be used in commercial instruments. He has published or presented over 30 papers in the areas of lasers, microwave components, and microwave acoustics. He and his wife and two children live in Santa Rosa, California.

George A. Patterson

With HP since 1975, George Patterson was instrumental in determining the feasibility of using MBE technology for devices for the Microwave Technology Division and is responsible for the continuing development of MBE. Before working on MBE he was a process development engineer for silicon and GaAs devices. Born in Mineola, New York, he served in the U.S. Army Signal Corps and is an alumnus of Michigan State University (BSEE 1969). He also attended Stanford University and worked as a silicon process engineer at Fairchild Semiconductor Corporation before coming to HP. He is coauthor of a paper on using ion implantation for doping GaAs. A resident of Santa Rosa, California, George is married and has a son. His wife is an HP computer systems administrator. He's an avid mountain climber and skier and has climbed most of the mountains in the Cascade Range in the western United States. He and his son climbed the east side of Mount Whitney, California last year. He's also a pilot and has instrument, commercial, and multi-engine ratings.

22 Subharmonic Mixers

Robert J. Matreci

Born in Chicago, Illinois, Bob Matreci is an alumnus of Wichita State University (BSEE 1969) and the University of Kansas (MSEE 1973). He worked on avionics test equipment before coming to HP in 1978. He contributed to the development of the HP 11970 Series Harmonic Mixers and is now a project manager for millimeter-wave components and systems. His work on a millimeter-wave component is the subject of a patent application. Bob and his wife live in Santa Rosa, California and he's active in his church. He's a sailing enthusiast and was navigation and communications consultant for Peter Bird's 1982-83 solo row across the Pacific Ocean. He has also been finishing some work on his new house.

30 Predictive Support

Bruce J. Richards

A section manager at HP's Knowledge Systems Laboratory, Bruce Richards has been responsible for several computer system support tools, including Predictive Support software for the HP 3000 Computer. He joined the Neely Sales Region in 1976 and held various field support technical and management positions before moving to his current job. A California native, Bruce was born in Long Beach and educated at the University of California at Irvine (BSEE 1976). Bruce and his wife live in San Jose and are the parents of a new baby. An avid photographer, he also likes skiing and tasting and collecting wine.

David B. Wasmuth

Dave Wasmuth is a software engineer and has worked mainly on HP Predictive Support software since coming to HP in 1984. He worked in software integration and testing for the System Development Corporation and for the IBM Corporation before joining HP. Dave was born in Dallas, Texas and graduated from California State University at Chico with a BS degree in computer science in 1984. A resident of Los Altos, California, he likes all kinds of sports.

34 AIDA

Craig M. Myles

An HP 3000 Computer systems specialist in HP's St. Louis, Missouri office, Craig Myles came to HP in 1979. He specializes in solving MPE operating system problems, often writing software and hardware diagnostics. He was the MPE specialist for the development of AIDA. He was born in Berea, Ohio, and graduated from The Principia College with a BS degree in physics in 1979 after serving for four years in the U.S. Navy. Craig and his wife live in St. Louis. His outside interests include scuba diving, bird-watching, swimming, and astronomy.

Lynn R. Slater, Jr.

Born in El Paso, Texas, Lynn Slater attended Texas A&M University and received his BS degree in computer science in 1983. He also gained programming experience by working for several organizations while finishing his degree. With HP since 1983, he has contributed to an internal HP software design and development environment and is now project leader for the AIDA expert system. He's a member of IEEE and ACM. A resident of Fremont, California, Lynn is active in local politics. His outside interests include target shooting and motorcycling.

Keith A. Harrison

Keith Harrison was born in Burton On Trent, Staffordshire, United Kingdom. He studied mathematics at Pembroke College, Oxford University, and graduated with BA and MA degrees in 1975. Before coming to HP in 1980 he developed computer-aided design tools and wrote data processing software for the Kingdom of Saudi Arabia. At HP he has contributed to HP NLS/3000 and the AIDA expert system and now develops ways to deliver knowledge-based services to customers. Keith is a resident of Wokingham, Berkshire, U.K. His outside interests include compiler design, language design, and playing recorder and classical guitar.

42 Schooner

Diane M. Ahart

Born in Ilion, New York, Diane Ahart attended the University of Minnesota, from which she received a BS degree in computer science in 1985. She has been with HP since 1985 and is working on an example-based knowledge acquisition and delivery tool. She also maintains and supports the Schooner expert system. Before coming to HP she worked at a Honeywell, Inc. artificial intelligence laboratory. Diane lives in Foster City, California and enjoys skiing and camping.

R. Michael Young

Michael Young was born in Salinas, California and worked as an actor and teacher before studying computer science at California State University at Sacramento (BS 1984). He came to HP the same year and has contributed to the development of the Schooner expert system and to Common Lisp for HP Precision Architecture. He's currently on a leave of absence from HP to complete a master's degree in computer science from Stanford University. Michael's professional interests include theories of automated reasoning, nonmonotonic logics, and modeling of processes. He lives in Sunnyvale, California and enjoys windsurfing and playing squash.

Brian T. Button

Brian Button was educated at the University of California at Berkeley and received his BS degree in electrical engineering and computer science in 1982. After joining HP's Computer Systems Division the same year he worked on system microcode and then was a project leader for the Schooner expert system. He's now a product manager for the multivendor portion of the network support program and is interested in computer languages and knowledge representation. A native of California, he was born in Richmond and lives in San Jose. Recently married, he's active in the Junior Chamber of Commerce and likes skiing and backpacking.

48 IPT

Roy M. Vandoorn

Born in Edmonton, Canada, Roy Vandoorn studied mathematics and computer science at San Jose State University. He completed work for his BA degree in 1978 and for his MS degree in 1980. He has been with HP since 1980 and is an R&D project manager for advanced diagnostic tools at the Knowledge Systems Laboratory. He has worked on IPT and on a software defect tracking system. He's also a member of the American Association for Artificial Intelligence. A resident of San Jose, California, Roy is a scuba diver and helps to train other divers.

George R. Gottschalk

An R&D software engineer, George Gottschalk has been with HP since 1984. He has contributed to the development of IPT and continues to work on other software support tools. George was born in Hamamatsu, Japan and attended San Jose State University. He studied mathematics and computer science and received his BA degree in 1984. He's now living in Palo Alto, California and enjoys hiking, backpacking, and drawing cartoons. He says he's also working on an epic poem about IPT.

54 Mycon

Robert I. Marcus

Bob Marcus was born in Brooklyn, New York and studied mathematics at the Massachusetts Institute of Technology (BS 1963) and New York University (PhD 1972). He was a university professor before coming to HP in 1984 and recently left the company. An R&D software engineer, he worked on a configurator for the HP 3000 Computer and on an interactive information structuring tool. He has published five papers on stochastic partial differential equations and three papers on computer science and is a member of AAAI, ACM, IEEE, and the Society for Industrial and Applied Mathematics. Bob and his wife live in the Seattle, Washington area and have one child.

Predictive Support: Anticipating Computer Hardware Failures

Predictive Support software for the HP 3000 Computer lives on the customer's system and notifies appropriate personnel of impending failures.

By David B. Wasmuth and Bruce J. Richards

IN THE BUSINESS of customer support it is very desirable to identify and repair weak or failing products before they degrade to the point where the impact on system users is significant. To this end, Predictive Support software for the HP 3000 Computer was created.

Briefly stated, Predictive Support is an HP 3000 software product that lives on the customer system and periodically examines the soft error rate of the various system components. When these rates approach uptime threatening levels, the Predictive Support system automatically notifies the appropriate person so that corrective action can be taken. The current implementation of Predictive Support covers all system disc drives, magnetic tape drives, and system memory.

Products being considered for predictive analysis must have a high degree of internal error detection and reporting capability for Predictive Support to be effective. Once a candidate has been selected, product experts from HP's field Response Centers and manufacturing divisions model the degradation and failure modes of the product. The results are reduced to a set of product independent rules to be incorporated into the Predictive Support product. The definition of the rules is a dynamic process, so the Response Center experts must pay constant attention to the effectiveness of the rules established for each product.

Predictive Support Software Operation

After Predictive Support is distributed to the customer system, the analysis begins. Predictive Support executes in four basic phases. First, error data is collected. Second, the error data is reduced to a generic format. Third, trend detection is performed, and finally, if necessary, the appropriate actions are taken to solve any problems. Overall processing is controlled by the predictive monitor process. Fig. 1 shows a graphic representation of the Predictive Support architecture.

Predictive Support uses special utility programs to collect error data. Each utility is launched as a child process and retrieves the error data for a specific class of products. Predictive Support currently uses three utility programs to collect data from system-maintained log files, internal disc drive logs, and processor memory logs.

In the second phase of processing, the utilities translate the myriad of error data formats into a common message format, so the predictive monitor can use the same trend detection algorithm to process the error data to determine whether a failure is imminent on the specific system component. The messages contain information identifying the product and the class of error involved. In addition, error class specific information that is not needed for trend detection, but is necessary for further definition of the particu-

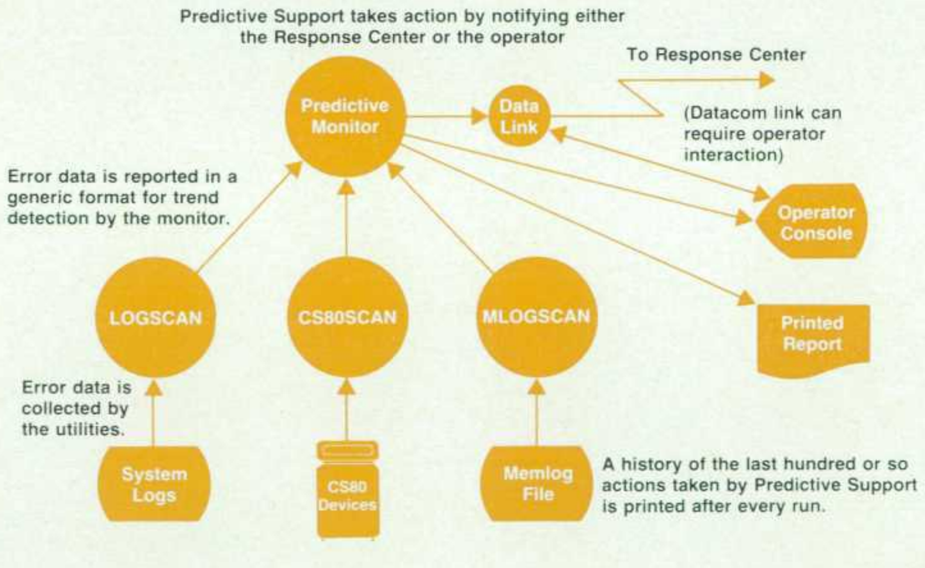

Predictive Support takes action by notifying either the Response Center or the operator
To Response Center
Predictive Monitor
Data Link
(Datacom link can require operator interaction)
Error data is reported in a generic format for trend detection by the monitor.
Operator Console
LOGSCAN
CS80SCAN
MLOGSCAN
Printed Report
Error data is collected by the utilities.
System Logs
CS80 Devices
Memlog File
A history of the last hundred or so actions taken by Predictive Support is printed after every run.

Fig. 1. *Structure of the Predictive Support software.*

lar error, is appended to the message and passed on to the eventual troubleshooter.

When the messages are received by the predictive monitor process for third-phase processing, the error data is passed through a trend detection algorithm. If the results indicate that an undesirable trend has been established, the appropriate action is triggered. For most products, the vast majority of soft errors are normal and transparent to the system users. The trend detection algorithm is described in more detail later.

Predictive Support Output

Taking the specified action is the fourth and final phase. The predictive monitor notifies the appropriate person so that corrective action can be carried out by the customer or by HP support personnel. In cases where the customer is notified (e.g., a specific medium is marginal, or a drive needs cleaning), the predictive monitor sends a message to the console, informing the operator about the problem. In cases where HP needs further investigation of a potential problem, Predictive Support uses a communication link to transfer the information directly to the Response Center for investigation. At the end of every run of Predictive Support, regardless of the actions taken, a report is generated listing all of the messages output during this and previous runs of Predictive Support. This provides a hard-copy record of Predictive Support activity.

The data communication link between the customer system and the Response Center uses the remote support modem installed on most of the HP 3000 systems in the U.S.A. At the end of the run, if messages have been generated for HP support, Predictive Support requests the system operator's permission to use the modem. Once permission is granted, Predictive Support autodials the modem, if possible and allowable, calling into a secured port on the Response Center job management system. A single file is then transferred (with error checking) from the customer system to the Response Center. At each step of the transfer the customer has complete control over the connection.

At this point Predictive Support has completed its execution. In the cases where it has transferred action data to the HP Response Center, the HP engineers may investigate the problem further and recommend the appropriate course of action. Often this recommendation requires on-site action by the field customer engineer, but the downtime will be scheduled at a convenient time. This advance warning saves the customer the inconvenience of unscheduled downtime and allows HP to schedule customer engineering resources more productively.

Trend Detection with Predictive Support

The objective of the Predictive Support software is to monitor trends in computer system operation as they occur, notifying the appropriate person when a trend indicates that there may be a problem. It was obvious that some statistics must be gathered and analyzed, but the questions of how to keep these statistics and what to count had to be resolved.

To analyze system trends, we need to identify entities that can be monitored in the hope of detecting signs of impending failures. The most familiar source of failures on a computer system is the hardware, e.g., peripheral devices. The entities that are monitored by Predictive Support are referred to generically as devices. Nonperipheral devices such as memory are referred to as pseudodevices. For each type of device that is to be monitored, some set of significant programmatically observable events must be identified so their occurrence can be tabulated and analyzed. The best example of an event to be monitored is an I/O error condition, such as a recovered read error on a disc drive. As implied by the designation, these errors are detected and corrected by the drive and do not have a noticeable effect on performance or integrity. A single occurrence of such an event would not indicate a problem, but when the frequency of recovered error occurrence reaches a certain level, a pending failure may be indicated.

For trend detection, the frequency of occurrence of these significant events must be calculated for each device being monitored. Merely tabulating the occurrences of an event will not, however, yield the frequency of occurrence. We therefore need to track some other factor to weight the occurrence of the significant event. Since simple continuous tracking of event occurrence and weighting factor will lead to a dilution of the statistics, the accumulation of the weighting factor must be limited to create a sample across which the frequency of occurrence can be analyzed. The sample can count something related to the event, such as the number of disc accesses, or a time interval. To identify an undesirable trend, a threshold must be defined where the occurrence value, relative to the defined sample size, is considered undesirable.

In summary, to detect abnormal system trends, we need: 1) entities to monitor, that is, devices and pseudodevices, 2) definition of significant events that occur on the devices, and 3) rules defining both the samples across which event frequencies are monitored and the relative thresholds of event occurrence where problems may be indicated.

Trend Detection Data Needs

The Predictive Support software system uses two major data structures to manage the collection and analysis of the statistics kept for detecting system trends. One is the trend log, a repository for collected event occurrence data. The other is the normalization matrix, which contains a set of rules governing the logging of event data in the trend log and the interpretation of the resulting trend data. The matrix makes the trend detection process table-driven. There is a unique set of events defined for each type of device monitored and one or more rules in the matrix for each event. A single element of the trend log is referred to as a bucket and contains the current event occurrence and sample values, along with a time stamp associated with the last time the bucket was updated. There is logically one bucket in the trend log for each normalization rule associated with each actual device or pseudodevice configured on the system. The buckets are grouped into sets by device type. The set size is determined by the number of devices configured as the given type, leading to a one-to-one correspondence between rules and bucket sets.

The rule has two parts that control the collection and analysis of statistics in the buckets of its set: a bucket definition and an action specification. There are fields in the

Systems Design for Worldwide Delivery of Customer Support

Customers using HP products count on fast, effective support to maximize the use of those products. HP meets this commitment by providing support services for hardware and software products locally from field offices and remotely from centralized Response Centers throughout the world. This cooperative approach to support maximizes the use of HP resources and returns customers to operation as quickly as possible. Fast delivery of solutions, in turn, yields the parallel results of increased resource availability and decreased cost of ownership.

The development of systems and tools to be used in the problem-solving process—including the software described in this issue on pages 30 to 56—is the charter of the Knowledge Systems Laboratory, a part of HP's Customer Support Operation. Another part of the operation, the Support Systems Laboratory, develops the management information systems used in the process.

In evaluating possible projects for the Knowledge Systems Laboratory, we identify several areas of focus. To increase resource availability, we concentrate on the following approaches to problem solving: predicting failures before they occur, responding quickly when they occur, repairing products rapidly, and diagnosing problem causes accurately. To decrease cost of ownership, we concentrate on solving problems quickly, solving problems *once*, managing both human and material resources efficiently, minimizing training costs, and sharing information on a worldwide basis. Since the problem-solving focus areas interrelate, many proposed projects make improvements in more than one area.

Although all software developed by our laboratory is intended for internal use only, it is subject to the same standards of quality as external HP software. All projects selected for development must be designed to user specifications for functionality, usability, performance, reliability, and supportability. Expert systems tools must meet the additional criterion of an easy-to-update knowledge base for long-term usability. Updatability thus becomes another criterion guiding the design of our projects. Some applications may call for ultimate delivery as self-contained or closed systems for which we maintain the contents while others are best delivered as operating shells for which we provide only the structure and the user maintains the contents.

One project that promised positive results in both resource availability and support costs was the capability to predict failures on key elements of an HP 3000 system. Since electromechanical devices account for a significant amount of system downtime (and a corresponding amount of on-site calls), these products were identified as likely candidates for proactive support techniques. Error detection and reporting capability proved to be equally important features for products suited to this type of support, and disc drives, tape drives, and system memory were ultimately selected as subjects for our Predictive Support software project. The software system revolves around a single algorithm delivered as a closed system to customers. Predictive Support software has proven successful in increasing customer system availability, scheduling on-site visits, and centralizing customer system information.

To improve results in the areas of rapid repair and accurate diagnosis, another project team focused on the area of expert systems for use by field and Response Center engineers. Since success with these systems depends on a clearly defined domain, the specialties of HP 3000 dump analysis, asynchronous terminal-to-HP 3000 datacom links, peripheral troubleshooting, and system configuration were selected as the domains of four separate projects. As some of the first practical applications of knowledge engineering techniques, the projects have made contributions to the training of engineers with limited experience as well as to the skill level of engineers whose expertise lies outside of these domains.

Blenda Mariani
Product Support Manager
Knowledge Systems Laboratory

bucket definition that define the bucket's sample value in terms of the normalization method and maximum size. The method specifies whether the sample is a time interval or an accumulation of some other event (e.g., disc accesses). The sample value in the bucket cannot exceed the maximum size defined in the rule. If the logging of event data causes this condition to occur, the occurrences relative to the obsolete portion of the sample must be discarded before the new event occurrences can be added in. The formula used to reduce the occurrence value when this overflow condition arises is described in detail in the box on the next page. The bucket definition also includes the event occurrence threshold. If after event data is logged the bucket's event occurrence value is at or above the threshold, the action defined by the action specification part of the rule will be taken.

The actions of the Predictive Support system will frequently involve communication with human users, namely the customers and/or HP's support engineers. After the message is sent to the appropriate person, there are three options: the bucket values (time stamp omitted) can be left alone, reset to zero, or reset with event processing suppressed (only if the bucket has a time normalization method). For example, some disc devices have their own microprocessor controller for which the software is held in ROM. The utility for examining their logs only supports the current (and future) ROM revision, and it checks the revision level before reading the drive logs, reporting an event if the customer's drive is using an unsupported version. This condition is then reported to the Response Center, and a service call is initiated to upgrade the ROM. Service calls of this nature are usually scheduled weeks in advance, so there is no need to continue reporting it each time Predictive Support is run. For this reason, event processing is suppressed for that bucket for the length of the sample size (3 weeks).

Predictive Event Processing

During event processing, the predictive monitor launches each utility, one at a time, and waits for it to send a message. It wakes up when a message is received, and if it is an event record, the following algorithm is performed to log the trend data:

- Use the device identification information to find the first rule in the normalization matrix.
- Read the trend log values associated with the rule, and use the time stamp, along with the event record's time stamp, to compute the interval to be used as the event record's sample value for rules with a time normalization method.
- FOR EACH rule associated with the event number and device type specified in the event record, log the event record's trend data into the trend log as follows:

```
IF the rule has a suppress action,
AND the bucket is suppressed,
AND the time since last update is less than the rule's maximum
    sample,
  THEN skip the following steps, continuing with the next
    rule.
Use the algorithm described in the box on the right to log the event
    data into the trend log.
IF the bucket's new event occurrence value is greater than or
    equal to the threshold defined in the rule,
  THEN take the specified action by outputting the message
    to the appropriate destination and optionally resetting the
    bucket values and suppressing further event processing.
```

- Continue with the next rule and, when all rules have been applied, wait for the next message.

Conclusion

Using the files and algorithms described in this article, the Predictive Support software system collects meaningful system data. This data is gathered and analyzed in terms of statistics that provide insight into the trends of computer system operation. This information can then be used to detect abnormal system trends in an effort to identify and repair weak or failing products before their impact on system users becomes significant. Clearly, not all failure modes of products manifest themselves as degradations before eventual failure. But for those frequent cases where this does occur, Predictive Support is quite effective.

Acknowledgments

Both authors would like to thank Milica Tutush, Robert Henely, Zahid Khan, and Russ Lown for their contributions as members of the Knowledge Systems Laboratory project team. And in recognition of the extended nature of the team, we would also like to extend our appreciation to John Hodgson, Mike Byce, and Tom Moore of HP's Disc Memory Division for their help in setting the rules affecting disc drives, to Chris Anderson, Dick Schafer, Bob Getzfred, Marge Mortenson and many others at the Response Centers for their assistance in planning for the operational impact of the project as well as for implementing it in the field, and to Joe Ziegler, Ed Canfield, Steve Fiedler, Craig Beutelspacher, Mike Phillips, Carol Ling, and Jon Edwards for their commitment to working with us to write the utility processes and datacom code that form the operating core of the system.

Logging Event Data in the Trend Log

Logging event data involves adding the event record values to the current trend log values. The resulting event occurrence value is then compared with the threshold to determine whether to take action.

Three pairs of values are used for trend detection. X values are associated with event occurrences and Y values are associated with the sample values. In this description, each of the pairs will have a subscript designation. The count and weight values from the event record use the subscript e. For example, if the normalization method is time, Y_e = time since last update. The trend log values for current event occurrence and sample values stored in a bucket use the subscript b. The threshold and maximum sample values defined in the normalization rule use the subscript r.

When an event record's values are logged, one of three mutually exclusive conditions exists. The equations used to set the new bucket values in each case are as follows:

Case 1: $Y_e \geq Y_r$ (event sample larger than maximum)
$X_b \leftarrow (X_e/Y_e)Y_r$
$Y_b \leftarrow Y_r$

Case 2: $(Y_e + Y_b) \leq Y_r$ (bucket sample will not overflow)
$X_b \leftarrow X_b + X_e$
$Y_b \leftarrow Y_b + Y_e$

Case 3: $(Y_e + Y_b) > Y_r$ (bucket sample will overflow)
$X_b \leftarrow X_b - (X_b/Y_b)(Y_b + Y_e - Y_r) + X_e$
$Y_b \leftarrow Y_r$

Under Case 1 the event record values cover the entire sample, so the old statistic is discarded and the ratio of the event record values is multiplied by the maximum sample, resulting in the new X_b value.

Under Case 3 an attempt is made to discard obsolete occurrences, so the statistic is evaluated only for the sample specified in the rule. The X_b value must be normalized before X_e is added. The amount of overflow is multiplied by the current ratio of the bucket (i.e., frequency of occurrence), and the result is subtracted from X_b before X_e is added.

AIDA: An Expert Assistant for Dump Readers

This expert-system-based program increases human readers' productivity and success rate in HP 3000 memory dump analyses.

by Lynn R. Slater, Jr., Keith A. Harrison, and Craig M. Myles

AIDA (Automated Interactive Dump Assistant) is an application that is designed to assist the human dump reader in the analysis of memory dumps taken from the HP 3000 family of computers.

Dumps (see box, next page) are read for two main reasons. The support engineer may simply want to know what the problem was, to provide a workaround in as short a period of time as possible. Alternatively, the engineer may wish to determine the precise sequence of events that caused the failure, to effect a cure.

Dump analysis requires an unusual set of skills. The dump reader must be very familiar with the details of the operating system and its subsystems to relate memory locations to memory contents at the time of the dump. The dump reader needs to be able to scan dynamic links, identify the role of each memory portion, determine what was happening at the time of the dump, determine what is unusual, and from this determine the general nature or exact cause of the problem.

In analyzing a dump, a dump reader must know many details. Many operating system data structures are variable in format. Normally legal values may be inappropriate depending upon other parts of the dump, and normally invalid values may be appropriate for short times as the operating system updates other values. Dumps may be the expected result of an improper configuration, may be a known problem with complex symptoms, or may be unique. Also, the direct cause of a problem may have been the result of memory improperly modified by some other cause. The initial cause of the problem may not be in memory or may have been overwritten by the effects of the problem.

Dump readers are hindered by the lack of any single source for their detailed knowledge. Within Hewlett-Packard, the dump reader's task is further complicated by our commitment to supporting multiple versions of the operating system. Our dump readers must master differing sets of detailed knowledge for each version. An expert system provides a central repository for this information and also guarantees that it will not be forgotten or lost as experts move to newer releases or change jobs.

We cannot reduce the inherent complexity of the dump, but we can try to encapsulate the detailed knowledge, which can then be thought of as rules,[1] in a tool. Reducing the amount of detailed knowledge needed to read a dump both speeds dump analysis and increases the pool of dump readers. We can further reduce the time needed to read a dump by allowing user management of the information presented.

All this detailed knowledge must then be applied to every significant portion of memory. Any portion may be corrupted, and any corruption is a clue to the nature of the problem. Humans cannot apply their expertise to every portion of the dump and still complete the analysis in a reasonable amount of time. Instead, they must start from a few leads and try to discover the portions of memory worth detailed analysis. On the other hand, an automatic tool can scan each portion of memory as it is first encountered and report unusual conditions to the dump reader. Human dump readers, relying on personal experience and knowledge unobtainable (or uncodable) to the tool, can then decide the significance of the corruptions.

```
Corruption Detected in : CONFIGURATION
Because : Ldev 11's CPVA(0) word (142036) <> 0
**** DMA Abort - CHREG B = 002036  Mem Addr 4.150000

Corruption Detected in : CONFIGURATION
Because : Ldev 100 is a real device with a DIT pointer of 0

Corruption Detected in : CONFIGURATION
Because : Warning: Ldev 163 default output is to ldev 162

Corruption Detected in : CONFIGURATION
Because : Warning: Ldev 183 default output is to ldev 182

Corruption Detected in : CONFIGURATION
Because : Gic 4 on IMB 0 only contains INP/Translator Cards.
Please check that there is no HPIB hood cable on the GIC card,
as this can cause memory corruption.

Corruption Detected in : CONFIGURATION
Because : IMB 1 has 3 high speed Gics configured. 2 is the maximum

Corruption Detected in : DST-70
Because : Number of DRQ table entries (0) <= 0
Assuming 1130 entries

Corruption Detected in : MEMORY-4-150000
Because : Found a region of length = 0
```

Fig. 1. *Typical corruptions detected by AIDA.*

What Is a Memory Dump?

A memory dump is a copy of the contents of a computer's dynamic memory along with other key information such as register contents, I/O channel programs, and virtual memory. In effect, a memory dump is a snapshot of what the computer was doing at the time of the dump. The dump is often made by special-purpose programs that either always reside in memory or are supported by firmware. These programs are normally designed to make no use of the operating system and write the contents of memory directly to a tape.

Many tools exist to find a problem if a regular application program fails. These tools may run as independent processes that examine or modify the data or code of the application program. However, if the operating system fails, diagnostic programs will not be able to run until the system has been restarted and the evidence of the problem destroyed. Therefore, a memory dump is taken to capture the state of the machine at the moment of the failure. Memory dumps preserve the evidence of the problem for later diagnosis; often this is the only evidence. Without dumps, experts can only make educated guesses about what might have happened. Even if the problem is theoretically solvable without dump analysis, the dump is often a good first clue to the cause.

Operating systems may fail for many reasons. The CPU may encounter an illegal instruction in privileged code, external interrupts may occur faster than they are resolved, the operating system might believe that it has no work to do or be waiting for something that will never happen, or internal routines might detect abnormalities of such seriousness that they call system failure routines rather than risk destroying still valid data with continued processing. Most often, these are the final steps in a problem that has already happened.

In addition to AIDA, there are two tools for the dump reader:

- DPAN is a program that formats certain tables and memory from the dump tape and produces a line printer listing. Users are faced with two options. They can print the whole dump, an unattractive option particularly for the larger machines, or they can second-guess the problem, print certain tables, and then, if more information is needed, print additional information as required.
- IDAT is an interactive version of DPAN. It offers a different set of formatted tables than DPAN, but presents information interactively upon demand. IDAT is also capable of accessing a dump tape on a remote machine. To use IDAT successfully, the user needs to be very skilled in dump analysis.

Automatic corruption detection alone will rarely solve a dump. Most often it is just the first step. The corruptions indicate why the system failed, but you must know why the corruptions occurred to solve the problem. This is the real task of dump reading. Experienced dump readers often scan source code or uncorrupted memory for clues. Even if some of this process could be automated, there will still be portions of dump analysis that will depend upon a person's ability to solve problems never before encountered. Until a tool can guarantee an automatic solution to any memory dump, human dump readers will be involved. Therefore, to allow the user to pursue the problems the tool did not solve, a good dump reading tool should act as an expert formatter as well as an automatic analyzer.

AIDA Overview

AIDA provides the following capabilities:

- The automatic detection of simple data structure or subsystem corruption
- The automatic analysis of some types of class problems
- Ergonomic presentation of data from the dump.

AIDA is an expert-system-based application intended for use by support engineers in the HP Response Centers, HP Customer Escalation Centers, or HP product divisions to solve customer problems in less time and with more confidence that the problem will not recur. It can also be used by product divisions to debug new operating system components.

AIDA concentrates on trying to identify the problem automatically while providing dump readers with the tools to pursue their own lines of inquiry.

AIDA automatically detects standard problems such as resource deadlocks, and eliminates the need for the human dump reader to process these well-defined problems. AIDA also detects most forms of corruption in MPE tables or subsystems. (MPE is the Multiprogramming Executive, the HP 3000 operating system.) This automation frees the dump reader to concentrate on the more challenging aspects of the dump. However, AIDA cannot automatically examine the entire dump because of the dump's large size and the need for dump readers to have fast interactive results. Fig. 1 shows typical corruptions detected by AIDA.

While AIDA cannot guarantee an automatic solution to any problem presented, it is a superior formatter which makes it easy for dump readers to pursue independent lines of inquiry without changing their environment or their thought flow.

Assuming that the problem is not automatically detected by AIDA, or the user wishes to determine why the problem occurred, the reader can display any table or data structure resident in memory at the time the dump was taken. This data is displayed in as informative a way as possible. As the user directs AIDA to bring in more information from the dump, this new information is automatically scanned for errors or inconsistencies; these discoveries are added to a special corruptions data base and reported. The corruptions can often guide dump readers to a problem or symptom that otherwise may have gone unnoticed.

In practice, human dump readers do not normally completely analyze every dump they process. At times they do not invest the effort needed to follow the complicated chains that AIDA does and either give up or need a second dump for comparison. A major contribution of AIDA is its superior formatting, which allows it to follow these chains automatically. Hence, with AIDA the solution rate goes up. It does not just speed up the process, encapsulate knowl-

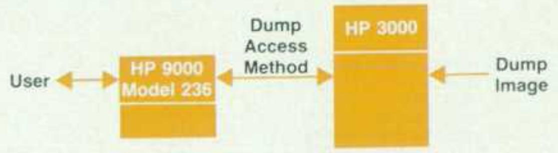


Fig. 2. *AIDA runs on an HP 9000 Series 200 Workstation. A process on the HP 3000 called the dump access method sees the workstation as a terminal.*

edge, and let nonexperts read dumps, it makes problems solvable that were not solvable before.

Environment

AIDA runs on an HP 9000 Series 200 Workstation running under the Pascal 2.1 operating system with a minimum disc size of 45 megabytes. This configuration was chosen because it was compatible with the prototype version of Hewlett-Packard's AI development environment.[1]

To obtain information from the HP 3000 dump tape, the strategy depicted in Fig. 2 was adopted. A process on the HP 3000 (the dump access method) sees the workstation as a terminal. Whenever the workstation requires information from the dump it sends a message to the dump access method, which parses the request, extracts the relevant data from the disc image of the dump tape, compresses the data, and writes it out to the workstation. All data transfers are checksummed and numbered for safety.

The HP 9000 portion of AIDA runs in a Lisp-based environment which provides compiled and interpreted Portable Standard Lisp,[2] objects,[3] and an EMACS-like editor called NMODE.[1,4] The NMODE editor supplied the code development environment as well as the foundation of the user interface. AIDA can be thought of as an NMODE extension that generates and manipulates data in ways particularly suitable for dump reading. Most of the information management described later in this article is taken from NMODE. By building on top of NMODE, AIDA is able to support any number of independent buffers, any number of windows on a large number of devices, infinite scrolling horizontally as well as vertically, context sensitive on-line help, list browsing with filters, and other features with very little development effort. These features are all benefits of the standard user interface utilities supplied by NMODE. Without these features, AIDA's contribution would be significantly lessened.

Knowledge Acquisition

When human dump readers start to read a dump, they

```
   *** MPE Logging ***
LOG FILE                       LOG0796
MPE LOGGING FLAG               000004
RECORDS MISSED                 #12
BUFFER 0 DST                   37 - Empty
BUFFER 1 DST                   40 - Current
BUFFER SIZE (W)                #512
BUFFER 0 RECORDS               #0
BUFFER 1 RECORDS               #3
LOGGING IS SUSPENDED           NO
LOGGING ENABLED                ON
JOB INITIATION                 ON
JOB TERMINATION                ON
PROC TERMINATION               ON
FILE CLOSE                     ON
SYSTEM SHUTDOWN                ON
POWER FAIL                     ON
SPOOLING                       ON
LINE DISCONNECT                ON
LINE CLOSE                     ON
I/O ERROR                      ON
VOLUME MOUNT                   ON
VOLUME SET MOUNT               ON
TAPE LABELS                    ON
CONSOLE                        ON
PROG FILE EVENT                ON
CALL PROGRESS SNGLS            ON
DCE PROVIDED INFO              ON
CS80 MAINT REQ                 OFF
DCU LOGGING                    OFF
```

Fig. 3. *Formatted header section of the MPE logging table.*

```
        ====================LST Directory=========================
        ENTRY TYPE LENGTH LDEV @ SECT ADDR PIN  CSTBLK PROCESS
                                                       COUNT
000001  Sharer        #7 #4@516068         355
000002  Garbage       #6
000003  Prog File    #40 #4@516068              000015      #2
000004  Garbage      #66
000005  Prog File    #64 #2@70270               000023      #2
000006  Prog File    #63 #1@126129              000016      #2
000007  Prog File    #84 #1@124390              000022      #2
000010  Sharer        #7 #2@41134          34
000011  Garbage       #6
000012  Prog File    #61 #2@41134               000017      #2
000013  Garbage       #7
000014  Sharer        #7 #3@704806         237
000015  Garbage       #7
000016  Sharer        #7 #2@655067         55
000017  Sharer        #7 #2@41134          171
000020  Garbage      #65
000021  Sharer        #7 #1@50303          27
000022  Garbage       #6
000023  Prog File    #34 #1@50303               000011      #2
000024  Sharer        #7 #2@36167          32
000025  Garbage       #6
000026  Sharer        #7 #1@50443          33
000027  Garbage      #13
000030  Loadproc     #42                   5
000031  Garbage       #6
000032  Loadproc     #50                   1
000033  Sharer        #7 #1@50443          31
000034  Garbage       #6
000035  Prog File    #35 #1@50443               000012      #2
000036  Prog File    #35 #2@36167               000013      #2
000037  Loadproc     #44                   33
000040  Prog File    #33 #2@982738              000014      #2
000041  Sharer        #7 #2@982738         35
000042  Sharer        #7 #2@36167          36
000043  Sharer        #7 #2@36167          40
000044  Garbage      #76
000045  SL File      #66 #4@107379
000046  Garbage     #301
```

Fig. 4. *A portion of the formatted data section of the loader segment table.*

scan the dump looking at the registers, process control block table, and other areas, trying to get a "feel" for the state of the machine. If they spot something that doesn't feel right, they direct their energies to trying to explain their observation before resuming with their initial search. The AIDA project determined and subsequently modeled the knowledge of the operating system, machine architecture, and typical search order by taking protocols of human dump readers analyzing selected dumps. A mixture of standard test cases and random dumps supplied by customers was used. It was found beneficial to choose the experts carefully. The most useful were engineers who were able to communicate easily. Also, the less experienced engineers were preferable because they still performed their analyses deliberately. As they became more competent they tended to work more instinctively and it was harder to determine why they worked the way they did.

Analysis Phase 1

The first phase of analysis is designed to emulate the human dump reader's initial scan; it gathers the minimum amount of initial facts needed to begin processing. The connection to the dump access method is created and the dump file opened. AIDA checks the file type and a few memory locations that allow AIDA to decide whether the file is a dump file taken from MPE V/E or later. AIDA then extracts and locally caches a predefined set of data via the dump access method. As this data is brought in, it is scanned by the automatic corruption detectors. It is possible that nothing unusual will be found during this first pass. At the end of this phase, the user knows miscellaneous data such as CPU model, operating system version, time of failure, system failure number, current instruction (automatically decompiled), console device, and last system welcome messages. Also, if the dump is from a system failure, explanations of the failure, probable causes and resolutions, and the calling modules or procedures (and the line of code within them) are extracted from separate master lists maintained by the MPE development laboratory, the Response Centers, and each dump reading site. This process takes less than 90 seconds with a 1200-baud phone link. The dump reader can direct AIDA to verify the HP 3000's configuration or scan and verify key tables such as the CST, DST, and PCB during this initial phase. This additional processing requires extra time.

Analysis Phase 2

Once the initial phase of data collection and analysis has been completed, all discovered dump inconsistencies are reported. The dump reader can now scan these messages and the miscellaneous facts derived from the dump to decide what to do next. The dump reader can have AIDA execute a user-defined suite of commands that may uncover further corruption while the reader works at some other task.

At this point, the dump reader is using an interactive formatting tool. Each time new data is requested, AIDA automatically scans it for corruptions. This fetch and scan on demand strategy is necessary because the size of a dump prevents automatic scanning of all of the dump in a time acceptable to an interactive user.

Internal Structure

AIDA keeps the formatting and corruption detection task separate from the control, presentation, and information management (user interface) task. This helps keep the knowledge of MPE separate from the user interface design. The tasks are interdependent and both tasks are always active.

Formatting and Corruption Detection

There are many tables and subsystems used in MPE, but generally only relatively few are significant to any particular dump. To reduce our memory requirements, AIDA has adopted a load on demand strategy: the AIDA code required to format or analyze a table or subsystem is brought into memory only as that table or subsystem is requested by the user. Besides reducing our memory requirements, this strategy has helped us keep our code very modular and structured so that additional tables or subsystems are easily added. AIDA does not have the capability to unload code once the formatting or corruption detection is complete. While this could theoretically cause AIDA to run out of memory as it reads a dump, this has only happened under maximum test conditions on our smallest workstations.

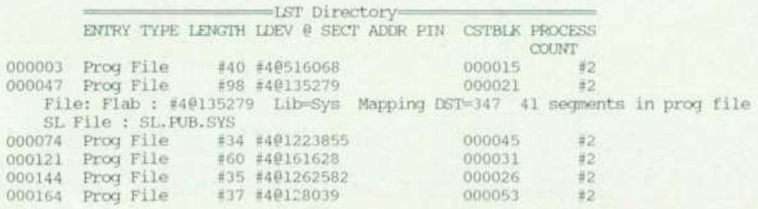
To add a new table or subsystem formatting capability to AIDA, entries are added to internal AIDA tables. These tables tell AIDA what command or commands use the code in the new module, the corruption checks performed by the module, and how to find the module. Each module also lists what other modules it needs loaded. The user interface portion of AIDA keeps track of the loaded modules and ensures that all the appropriate code is available before the execution of any command.

All modules call upon a set of fundamental utilities that access memory, format fields, validate value ranges, manipulate internal data structures or objects, supply data to the user interface, extract or set particular bits, or perform other low-level internal manipulations. All routines are written with the expectation that their arguments may have been derived from a portion of the dump with undetected corruption and thus may be nonsensible. The utilities return special innocuous values that will let the corruption become visible to the user without aborting any part of AIDA. This lets AIDA handle dumps with corruptions not anticipated by the expert who wrote the module. In the few cases where a routine does terminate abnormally, the structure of the Lisp environment and the NMODE editor underlying AIDA allows a graceful recovery where a command, column, or row is inaccessible, but the rest of AIDA still runs. Users can always display the unformatted raw memory to discover the nature of the corruption.

Table and memory reference routines are among the utilities available to the formatting modules. These are the only routines that actually access the dump data. These routines keep track of which portions of the dump have been accessed and cause the appropriate consistency and corruption checks to execute whenever new portions of the dump are accessed. They are, in effect, forward chaining rules whose firing is based on the availability of data. Thus every portion of the dump is checked and corruptions reported as soon as the data becomes available to AIDA. Corruption checks that depend upon the data from more than a single portion of memory may bring in all desired portions of memory whenever any portion is brought in, may wait until all portions are brought in, or may execute when most of the memory is brought in. AIDA MPE experts used all these strategies in coding different corruption checks.

There are basically two kinds of formatting: tables and subsystems.

Table Formatting. The formatting of a table involves displaying a single logical MPE structure as it appears to MPE. MPE is a table-driven operating system and the examination of tables is vital to dump readers. A table consists of a header section and a data section. Entries in the header



```
               ==================LST Directory==================
         ENTRY TYPE LENGTH LDEV @ SECT ADDR PIN  CSTBLK PROCESS
                                                         COUNT
000003   Prog File     #40 #4@516068              000015      #2
000047   Prog File     #98 #4@135279              000021      #2
     File: Flab : #4@135279  Lib=Sys  Mapping DST=347  41 segments in prog file
     SL File : SL.PUB.SYS
000074   Prog File     #34 #4@1223855             000045      #2
000121   Prog File     #60 #4@161628              000031      #2
000144   Prog File     #35 #4@1262582             000026      #2
000164   Prog File     #37 #4@128039              000053      #2
```

Fig. 5. *Same portion of the formatted data section of the loader segment table as shown in Fig. 4, but filtered to show only program file entries, and filtered again to show only entries from disc #4. The second entry is expanded.*

section apply to the table as a whole and may contain information such as number of entries, entry size, first free entry, etc. The data section is divided into any number of entries, each of which appears as a row and relates to some internal MPE logical entity. Each entry is further subdivided into any number of fields, which appear as the columns of the table. Some tables do not have both sections.

Our observations of expert dump readers showed that they are concerned either with some aspect of the whole table or with just some particular entries. They rarely examine the header and the data portions of a table at the same time. AIDA displays these sections independently. Fig. 3 shows a formatted table header section and Fig. 4 shows a formatted data section.

All table formatting is driven by a standard formatter. For the header section, the table specific code just specifies the labels of the entries and later the formatted values of the entries for each particular instance. For the data section, the table specific code specifies the labels of the fields (which become the columns of the table) and later the formatted values of the fields for each row. The transformation of this internal data into a display is handled by the control, presentation, and information management portions of AIDA. A key point of both table sections is that the table specific formatting routines are concerned only with the MPE specific logical format of the table and not with any of the implementation details such as screen width or whether the table is to be printed on paper.

AIDA often displays data that does not actually exist in a table but is logically related. If a table contains information about another table, this information may be displayed in the header section of the other table even though MPE does not physically store it there. Similarly, rows may have columns added where the data is taken from other tables. Where there is a great deal of related data for a row, and where this data would overwhelm the user if presented for all rows, AIDA will display a simple form of the rows but will let the user expand any row (Fig. 5). When a row is expanded, formatter routines are called with the row and table as arguments and can return any amount of data taken from any number of dump locations. The nature of expansions varies greatly with each table. For example, code segment table entries expand into decompilations of the actual code segment, and device information table entries expand into the channel programs. MPE tables were designed for customer performance and not for dump reading; these logical enhancements to the actual table structures make MPE more readable.

Subsystem Formatting. The formatting of a subsystem involves retrieval of data from many portions of the dump. AIDA includes in subsystem formatting the presentation of data that, while not a table or subsystem to MPE, is seen as logically related by a dump reader (Fig. 6). The display of DS users is an example of formatting of a true MPE subsystem (Fig. 7). The display of all of the executing programs along with user name, job or session number, and capabilities is an example of useful information that is not properly a subsystem to MPE. Unlike the tables, there is no common structure for subsystem formatting.

Control, Presentation, and Information Management

The user interface for AIDA presents some special challenges. The human dump reader, analyzing a formatted printout of the dump, is generally juggling many pieces of information, using pens, rulers, paper clips, etc. to mark places in the dump. Ideally, a user interface should provide the same capability. The user interface should assist the dump reader in the location and interpretation of the data. The user should not have to refer to an external architectural design document such as the MPE Tables Manual.

Multiple Buffers and Windows. Any number of tables or subsystems may need to be visible simultaneously. Initially, AIDA offered a table selection menu and graphics tablet to select an MPE table and an icon representing the desired action. However, the combination of pen and keyboard proved cumbersome and the keyboard alone became the input device.

AIDA adopted from NMODE the ability to display any number of buffers with independent data in any number of windows. Each buffer can be thought of as a separate dialog between AIDA and the user. This independence allows the user to pursue many trains of thought. These buffers are very similar to the ones used in the EMACS editor. The underlying NMODE editor allows quick and natural motion from buffer to buffer. Except for textual commands, users can navigate through AIDA and the dump with single-keystroke commands. For performance reasons, AIDA limits users to two windows on devices that are not bit mapped.

One of the buffers, the MPE tables browser, contains a

```
Table     Table Entries  Max In Use   Cur. In Use   Max %   Cur %   Size/Wds

DST        1023           580           445          56      43        4
CST        575            245           242          42      42        4
XCST       1047           559           413          53      39        4
PCB        255            255           80           100     31        21
IOQ        383   (378)    54            19           14      5         12
DRQ        -1    (585)    197           15           33      2         17
TBuf-123   240            42            22           17      9         69
TBuf-124   180            24            7            13      3         69
SBuf       35    (34)     8             3            23      8         129
TRL        510            -             71           -       13        4
ICS        2047           591           220          28      10        1
CSTBlk     128            -             46           -       35        1
JPCnt      100            -             25           -       25        1
Swap       2048           555           421          27      20        6
G Rins     1024           -             1            -       0         3
L Rins     1024           -             1            -       0         3
F Rins     1024           -             16           -       1         3
VMem       46080          33976         24816        73      53        Sectors
Spool      10240000       -             18120        -       0         Sectors
Memory     61 Avail Regions - Largest is 6528 words
Direc      11971 Sectors
```

Fig. 6. *An example of subsystem formatting. This shows tuning and table use information.*

list of all MPE tables available in the dump (Fig. 8). Users move the cursor to pick a table and use single-keystroke commands to read it. Another buffer accepts subsystem formatting commands and creates new buffers to display each subsystem. Another buffer contains optional values that the user can edit to control AIDA's operation.

Information Reduction. A major problem in any dump analysis is information overload. Most of the dump is of no significance to the problem. Normally, a dump reader analyzes a dump in a constrained and fleeting manner. It is normal to consider only one or two words of a table before going to a different table. The dump reader's task is to find and isolate the significant portions of each table or subsystem. Tools that just list formatted memory (either on paper or to a screen) are of no assistance here; the significant data is mixed with the insignificant. Rather than just listing to a screen, the user interface of AIDA receives tables from the formatter in a data structure that preserves rows and columns. Initially, the whole table is displayed, but facilities exist for the user to control which columns are displayed and to filter down to those entries that have a field equal to a given value (Fig. 5). This filtered list can be subsequently filtered. For example, when given a formatted process control block table 314 characters wide with 500 entries, a user might choose to look only at the wakemask (a column of the PCB table) of those entries that represent nonuser processes that are critical, hold an SIR, and are impeded. Instead of searching a display of 314 characters by hundreds of lines, the dump reader examines only two lines of 71 characters. This column and row specific filtering can be applied to any AIDA table format. The table-driven nature of MPE means that the user interface does not need any understanding of the semantics of the fields for the filtering to be significant in most cases. Also, for most tables it is possible to select an entry for a more detailed explanation.

The varied nature of subsystem formatting prevents AIDA from providing a general information management scheme such as the column selections and filters used on tables. Instead, the formatting is initiated by textual commands from the user (Fig. 9). These commands accept arguments that control the nature and depth of detail of the resulting display. Generally, a fresh buffer is created for each command, but AIDA avoids buffer overpopulation by combining the results of short formatting commands into a single buffer known simply as OUTPUT. The subsystem formatting interface is designed to simulate an interactive IDAT session. This design eases the dump readers' transition into using AIDA.

On-Line Help. The usability of AIDA is enhanced by a context sensitive on-line help system. The user can ask for help from anywhere in AIDA. The underlying NMODE editor knows what buffer the user was in and from that determines the purpose of the buffer and the appropriate help chapter. A buffer is created containing this chapter and substituted for the buffer the user was in. If the chapter does not contain the necessary information, the user can go to the on-line help table of contents and choose another. Users are returned to the original buffer when they leave the help system. MPE information useful to dump readers is also included in the on-line help system. This includes known entries into system failure routines, probable causes of system failures, and the meaning of each field in any table that AIDA formats.

Preservation of AIDA Sessions. Dump readers do not always complete their analysis of a dump in a single session. AIDA provides the ability to save a partially analyzed dump in the current state. All processing done, all dump data read, and all corruptions found are saved to disc. This feature allows a dump reader to respond to interruptions or wait for a response from another location and later continue from the same point.

Rules in AIDA

Readers familiar with expert systems might have noticed that the changing detailed knowledge required by experts to read dumps seems to offer an ideal application for rules in a traditional expert system. AIDA started from this observation and initially used techniques similar to those described in the articles on Schooner and IPT (pages 42 and 48). The results were acceptable except for performance.

Upon examination it turned out that formatting and most error checks follow a simple path with few control branches. Rules and traditional inference engines search a large rule domain for applicable rules to fire. This "hunt and peck" approach is very flexible and suitable for systems where there may be any number of rules in arbitrary order that become ready to fire in an unpredictable order. However, consider the case where rule B will always fire after rule A and at no other time. Unless the rule selection strategy is perfect, execution is faster if rule A and B are combined. Consider the case where rule C or rule D will fire after rule B and at no other times. These rules could be stated with a reformulation of rules A and B as part of their predicate along with whatever condition selects rule C over D or vice versa. Traditional inference engines would search the entire rule domain, evaluating the predicate of all rules encountered until rule C or D is discovered. Again, unless the rule search strategy is perfect, it would be faster to combine rules A, B, C, and D. (Also, many inference engines would have reevaluated the portions of rules C and D that had already been evaluated when deciding to fire rules A and B.) As these rules are combined, procedures are created; the longer the chain, the longer the procedure. The merger of rules A, B, C, D can be expressed as follows:

```
JobNum  Pin   Stack JobName           Ldev  DS-DST  Index

#S298   214   710   FIELD.DUMPS       40    401      0
#S298   315   1042  FIELD.DUMPS       40    701      0

DSMONX  Ldev: 180  Pin: ?   (No active Pin)       INP 17 has been downloaded
DSMONX  Ldev: 160  Pin: ?   (No active Pin)       INP 16 has been downloaded
DSMON   Ldev: 150  Pin: 40  DSMON  B.51.04 000  INP 13 has been downloaded
DSMON   Ldev: 50   Pin: 36  DSMON  B.51.04 000  INP 14 has been downloaded
DSMON   Ldev: 40   Pin: ?   (No active Pin)       INP 11 currently down loading
```

Fig. 7. *An example of subsystem formatting showing DS users.*

```
*** Available MPE Tables ***  << AIDA (A.02.05) >>

 LXR    Table-name    Width Length
 ___  ______________  _____ ______
 *X  Caching            78  14568
 *   CSDST-driver       36    460
 *   CSDST-ldtx        199    460
 *X  CST                69   2304
 *   CSTBlk             15    129
 *X  CSTX               87   4192
 *X  DCT                48   1144
 *   DDS               ???
 *   DirSpace          ---
 *X  DIT                78   3969
 *   DLT                85    216
 *X  DRQ-disabled       79     17
 *X  DRQ-free           79     17
 *X  DRQ-inuse          79     17
 *   DRQ-logical        77     17
 *X  DRQ-serial         79     17
 *X  DRT                36    828
 *   DS Devices         35
 *   DS Global         ---
 *   DS VPins           31
 *   DS VTrace          46
 *   DS XRef            75
 *   DST                78   4096
 *   FMAVT              54    576
 *   ICS               ---     64
 *   IDD               139   1664
 *X  ILT               115   2824
 *   IOQ-free           75   4620
 *   IOQ-inuse          75   4620
 *   IORT              ???
 *   JCut              ???
 *   JMat              118   1406
 *   JPCnt             ???
 *   LDT                52   1491
 *   LDTX               46   1065
 *   LPDT               56    852
 *X  LST                62   8965
 *   MeasTab           ???
 *X  Monitor            84   1024
 *X  MPE Logging        45   1036
 *X  Msg Port          105   1220
 *   MSGBUF            270   4100
 *   ODD               139   1920
 *   PCB                72   5376
 *   PJXRef            ???
 *X  Port Harbor       134   5261
 *   PVUser            ???
 *X  RIN                53  15367
 *   SBuf              270   4656
 *X  Shared-FCB         16      0
 *X  SIR                56    572
 *   SJDT
 *   SJIT
 *   SRT                99   2820
 *   SwapTab            81  12312
 *X  TDT                48   1144
 *X  TLT               113    455
 *   TRL                68   2052
 *   UCRQ               19    196
 *   VMem               77    804
 *   VTab               94    126
```

Fig. 8. *The MPE tables browser, a list of all tables available in the dump, is the starting point for all table access.*

```
IF <Predicate of rule A>
  THEN BEGIN
    <Action of rule A>
    IF <Predicate of rule B except those parts stated in rule A>
      THEN BEGIN
        <Action of rule B>
        IF <Condition that distinguishes rule C from rule D>
          THEN <Action for rule C>;
          ELSE <Action for rule D>;
          ENDIF;
        END;
        ENDIF;
    END;
    ENDIF;
```

AIDA is fortunate that in MPE each table or subsystem requires a largely independent body of knowledge. Dump analysis is a large problem domain, but the domain is divided into well-established, distinct subdomains—process hangs, system failures, flying bytes, etc. The knowledge base is compartmentalized and in each compartment the knowledge is easily expressible as procedures. This subdivision makes the problem manageable.

Initially, AIDA held detailed knowledge exclusively in rules, but these were gradually condensed into procedures until AIDA had very few rules left. These remaining rules were incorporated into procedures in the appropriate modules and all new tables or subsystems were created with procedural corruption detection schemes. The strategy of scanning for corruption on the first access to any memory portion serves to make these procedures essentially forward chaining rules. As we did this, performance improved dramatically. For example, a resource deadlock check that had taken over five minutes was reduced to less than two seconds.

Lessons Learned

Rules are not a necessary component of an expert system.

The history of rules in AIDA is probably the most significant lesson for others thinking of developing expert systems. Rules are not a requirement for a expert system. AIDA is no less expert with procedures than it was with rules and seems to be just as easy to modify. Basically, rules do not provide acceptable performance for low-level detailed checks performed many times. Applications with well-compartmentalized knowledge may do best to avoid rules.

We do expect rules to reappear in the next phase of AIDA. Now that we have this foundation of accessible dump knowledge, we can add rule systems to perform tasks such as deciding what portions of memory are likely to be significant and filtering for them, or deciding what AIDA command an expert user would use next given what AIDA has already discovered. For this phase, we will probably use a rules system based on the same principles as OPS5.[5,6]

Supply more than expertise.

The other lesson learned from AIDA is that an expert system that does not solve every problem presented will not gain acceptance unless it also acts as a tool for the user to solve the remaining problems. If AIDA had not been an expert formatter, it would not have been used because it could automatically solve relatively few special cases at first. By being part of the normal work flow, AIDA can make incremental contributions in each step by relieving the dump reader of having to check all the fields and tables manually and by sometimes finding a problem on its own. If AIDA were not an expert formatter, each dump not automatically solved would count against it. Instead, each dump AIDA automatically solves is justification for further use.

Acknowledgments

Many individuals and organizations in HP have contributed in numerous ways to the success of AIDA. However,

```
                    Process Related Functions
F PINS                    Display a summary of all processes.
F PINS, IMP               Display a summary of all impeded processes.
F PCB[,pin]               Display a summary of one PCB entry.
F PCB pin LSTT            Display the LSTT for the PCB entry.
F LSTT[,pin]              Display the LSTT for the PCB entry.
W ACTIVE                  Set current working process to active process.
W pin                     Set the current working process.
W                         Display the current working process.
DR                        Display the registers for the working pin
F DR pin                  Display the registers for the PIN.
F REGS                    Display the system register summary page.


            User Code/Data Segment Related Functions
F STACK[,dst[,level]]     Display a summary of a specified stack.
F PCB pin STACK[,level]   Display the stack for the PCB entry.
T                         Display stack markers for the working process.
F PCB pin MARKERS         Display the stack markers for the PCB entry.
F CO  cst,STT[,n]         Format the CST's STT (or just one STT entry)
F COX cst,STT[,n]         Format the CSTX's STT (or just one STT entry)
F CHECKSUM[,cst]          Compare the checksum value with calculated value
F LMAP[,NEW]              Display or change the current loadmap.
F LMAP[,BUILD]            Build a loadmap from code segments in memory.
F PROGS                   Display a summary of all running programs.
F CI                      Display the last CI command of each session.
F ICS                     Display the ICS stack markers.


                   I/O Related Functions
F DEVINFO                 Display the configuration.
F SHOWDEV[,ldev/class]    Print the device status for ldev/classname.
F TERM[,ldev]             Display information for the given terminal.
F TBUFS                   Display all terminal buffers.


                   MPE Utility Functions
F BANK[,n]                Print a list of bank objects and owners.
F DFS                     Display the disc free space details.
F JOBS                    Display a showjob listing
F MEM keyword             Display memory region headers.
F MISC                    Display miscellaneous dump information.
F SHOWCACHE               Display the current caching stats.
F SHOWDS[,dsline]         Display all users of DS/3000.
F SIRS                    Display any sirs that are held
F SYSGLOB                 Display System Global Definitions
F TRL                     Display the TRL active entries chronologically.
F TUNER                   Display table usage statistics.
:<command>                Execute the MPE Command Interpreter.
Search [loc][,pattern]    Searches memory for specific pattern of data


                   Debug Related Commands
D DA  xds  + offset , length , format
D EA  bank + offset , length , format
D CO  cst  + offset , length , format
D COX cstx + offset , length , format
D SY  offset , length , format
D SYX offset , length , format
D A   offset , length , format
DV ldev + sector , length , format
= expression , format
#[format[ width]]       Change width of display.
$                       Print out the current user-defined registers.
$n expression           Assign value to user-defined register n.


                   AIDA Utility Functions
F BROWSERS              Enter a browser of all system browsers.
F BUFFERS               Enter a browser of all created buffers.
F ERRORS                Re-Display all known corruption messages
F OPTIONS               Re-Enter the AIDA options browser.
SAVE-DUMP               Store the AIDA session, log off the 3000
F DUMPS                 Browse saved AIDA sessions and Volumes.
OPENLINE [linespeed]    Re-establish RS232 between AIDA and the host 3000.
CLOSELINE               Stop communication between AIDA and the host 3000.
TERM                    Enter the Terminal emulator.
FILER                   Enter the Pascal Operating system Filer subsystem.
BYE                     Terminate the AIDA session and log off the HP3000.
V                       Print version level of each loaded AIDA  module.
```

Fig. 9. *Subsystem formatting commands accepted by AIDA (extracted from the on-line help facility).*

a few of them deserve special recognition. Foremost is Stan Vierhaus who developed the dump access method. Accessing the dump data is only touched on in this article but is a prerequisite to any dump analysis. The St. Louis Customer Escalation Center has been most supportive by loaning some of their best MPE experts. The authors would particularly like to thank the Application Technology group of HP Laboratories and all others who created the excellent Lisp development environment that allowed the AIDA project to focus on dump analysis rather than the environment. Martin Griss and others brought Lisp to HP and provided the foundation and tools from which AIDA and similar projects are built. Marty Cagan, Mike Creech, and Craig Zarmer provided continuous technical expertise and support. Finally, the authors would like to thank Ira Goldstein who has inspired numerous projects as well as AIDA.

References

1. M.R. Cagan, "An Introduction to Hewlett-Packard's AI Workstation Technology," *Hewlett-Packard Journal*, Vol. 37, no. 3, March 1986, pp. 4-14.
2. M.L. Griss, E. Benson, and G.Q. Maquire, "PSL, A Portable Lisp System," *1982 ACM Symposium on Lisp and Functional Programming*, August 1982.
3. A. Snyder, *Object-Oriented Programming for Common Lisp*, HP Laboratories Technical Report ATC-85-1, February 1985.
4. R.M. Stallman, "EMACS: The Extensible, Customizable, Self-Documenting Display Editor," *Interactive Programming Environments*, Barstow, Shrobe, and Sandewall, eds., McGraw-Hill, 1984.
5. C.L. Forgy, *On the Efficient Implementation of Production Systems*, Department of Computer Science, Carnegie-Mellon University, 1979.
6. L. Brownston, R. Farrell, E. Kant, and N. Martin, *Programming Expert Systems in OPS5*, Addison-Wesley, 1985.

A Troubleshooting Aid for Asynchronous Data Communications Links

Schooner is an expert system for fault diagnosis and personnel training on point-to-point datacom links.

by Brian T. Button, R. Michael Young, and Diane M. Ahart

THE DATA COMMUNICATIONS market is among the fastest growing markets of the computer field, and data communications support has a rapidly increasing need of expertise. Solutions in this area are perceived by computer vendors and customers to be critical for a successful total solution. The trend toward personal computers and office automation has given increasing importance to easy installation and support of data communications links.

To address a portion of this problem, Hewlett-Packard initiated the Schooner project. Schooner's principal objective is to maximize leverage of HP's expertise in asynchronous data communications and reduce the time it takes to solve datacom problems for HP's customers. It attempts to do this by solving the easier problems, thereby reducing the workload on HP's more experienced engineers and freeing them to tackle other problems. Schooner also serves as a training tool for less experienced personnel.

Overview

Schooner combines an inference engine and a knowledge base[1] to provide expert-level assistance with asynchronous, point-to-point data communications problems for fault diagnosis and personnel training. It verbally guides Response Center engineers, field support personnel, or other users through the solution of problems in this area.

The present knowledge base is oriented towards diagnosis of problems in RS-232-C links connecting a terminal to an HP 3000 Computer either directly or with modems. When initiating a troubleshooting session with Schooner, the user designates the configuration by specifying any combination of terminal type, port type, operating system, connection type (cable or modem), and if applicable, the types of modems at the terminal and computer sides of the connection.

After acquiring this information, Schooner goes to the main investigation phase of the session, asking the user to relay observations and perform manipulations. Since the configuration description determines the rule set used in this phase, the session proceeds differently for modem-connected data links than for direct (cable-connected) links. Schooner understands problems and characteristics specific to the makes and models of devices in the configuration.

Schooner Tests

In Schooner, tests are the basic units of inference corresponding to rules in classic rule-based systems. Tests describe a state in which observations may be made, observations to make, and a translation from this information to beliefs about potential faults and the state of the system being debugged.

Although tests are the basic units of inference, there are important differences between tests and rules. Tests do not chain. Each test is a unit and is applied singly without direct interaction with other tests. Unlike a rule,[2] once selected and applied, a test is no longer eligible for selection unless disapplied.

Under appropriate circumstances, an applied test can be disapplied. All inferences resulting from the application of the test are retracted and the test is made eligible for reapplication.

The application of a test has three stages. Each of these stages has corresponding fields, which allow the user to specify what should occur there.

- Preliminaries. Perform the manipulations to the system necessary before the application of the query and record assumptions made for the test.
- Query. Obtain the desired information.
- Inferences. Update beliefs about the system based on the information obtained.

```
Test:          Local-Loopback-Thru-Modem
Bindings:      ?config  terminal                   ?term
                                  /* ?config is prebound */
               ?config  datalink                   ?link
               ?link    t-modem                    ?modem
                                  /* concerned with only one modem */
Requirements:  ?term    in-working-state           yes
               ?config  term-link-connected        yes
               ?modem   in-working-state           yes
Preconditions: ?modem   in-local-loopback          yes
Query:         ?term    do-typed-chars-appear      ?typed-chars-appear
Inferences:    if    ?typed-chars-appear is no
               then { FAULT: ?term    terminal-fault          indicated
                      FAULT: ?config  term-not-connected      indicated }
               if    ?typed-chars-appear is yes
               then { FAULT: ?term    terminal-fault          eliminated
                      FAULT: ?config  term-not-connected      eliminated
                      FACT:  ?config  term-link-connected     yes
                      FACT:  ?term    in-working-state        yes }
End-Test:
```

Fig. 1. *An example of a Schooner test. This test only applies to modem links. It consists of putting the terminal in a normal state to transmit, putting the modem in local loopback mode, and then typing on the terminal to see if characters come up on the screen. This test is used to verify the link between the terminal and the local modem.*

Preliminaries

Two fields in a test, the precondition and requirement clauses, are provided to allow the user to manipulate the system into the state necessary to apply the query. They are both ordered lists of simple clauses, but they differ substantially in intent and use.

In the test shown in Fig. 1, the precondition clause:

```
?modem in-local-loopback yes
```

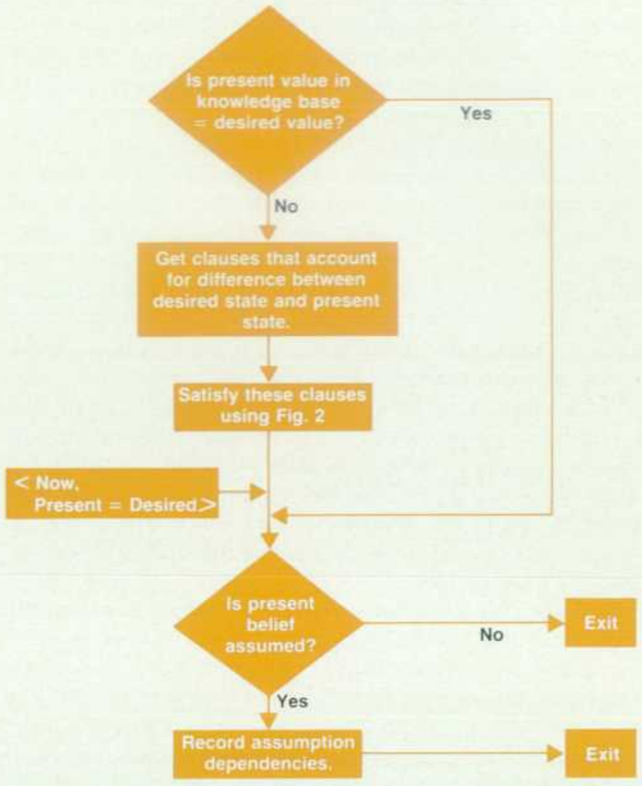
specifies a state of the modem that must be satisfied before application of the query. After the precondition clause is satisfied (Fig. 2), the modem will be in local loopback, regardless of its state before the test was selected.

Requirement clauses are used to establish an assumption about the state of the system. To satisfy a requirement clause, Schooner must bring the system to a state where it can assume that the clause is satisfied (Fig. 3). Having done so, Schooner must then tag the assumption so that the test information and application can be retracted if the assumption later turns out not to be valid.

In the test in Fig. 1, the requirement clause:

```
?term in-working-state yes
```

specifies that Schooner must put the terminal in a state in which it can be assumed that it works. If some previous test has put the terminal in local mode, then Schooner must ask the user to put the terminal in remote mode. Once this has been done, Schooner can assume that the terminal is in a working state, tag the dependency, and proceed to the next clause in the test.



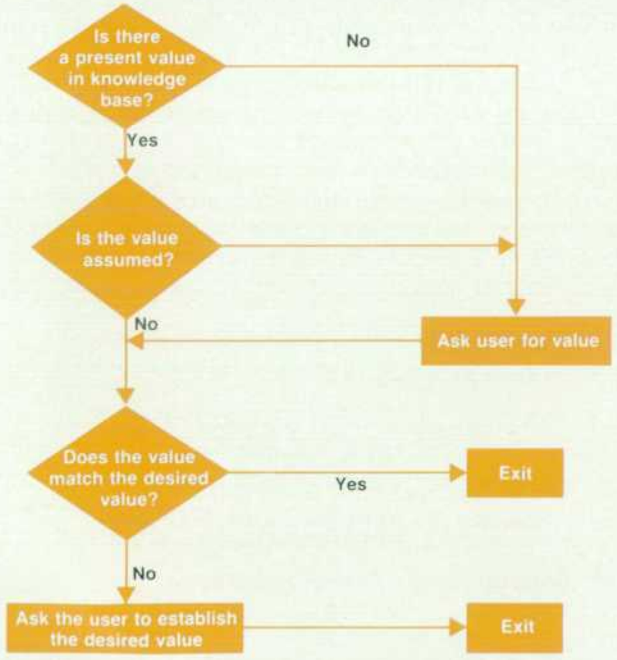
Is present value in knowledge base = desired value?
Yes
No
Get clauses that account for difference between desired state and present state.
Satisfy these clauses using Fig. 2
< Now, Present = Desired >
Is present belief assumed?
No
Exit
Yes
Record assumption dependencies.
Exit

Fig. 3. *Algorithm for satisfying a requirement clause.*

Queries and Inferences

The query is used to obtain information once the system is in a state appropriate to make the observation. A query is never used to achieve a state, but only to obtain information. Hence, the item in the third position in a query clause is always a variable (in Schooner, variables start with a question mark) rather than a desired value. The user is asked for the information needed by the query and the variable in the third position is set to that value. During the inferences, that value is substituted for the variable whenever it appears.

The inferences use the binding of variables set in the query to update Schooner's beliefs. Inferences may either assert beliefs about potential faults in the system or assert beliefs about the state of the system configuration.



Is there a present value in knowledge base?
No
Yes
Is the value assumed?
No
Ask user for value
Does the value match the desired value?
Yes
Exit
No
Ask the user to establish the desired value
Exit

Fig. 2. *Algorithm for satisfying a precondition clause.*

Inference Engine

The Schooner inference engine is designed specifically for troubleshooting. It uses a general troubleshooting process (Fig. 4) to investigate potential faults in the system being debugged. Schooner applies tests to the system to determine more about faults that may exist. The result of each test increases or decreases belief in various possible faults in the system being debugged. Subsequent tests are selected on the basis of such beliefs and other information provided by previous tests.

Data Structures

The Schooner inference engine uses several partially interlinked data structures. These provide an internal representation of the data link being debugged and maintain a consistent set of beliefs about the faults therein. The most important data structures are the configuration, the faults, the tests, and the linkage.

Configuration. The configuration (see Fig. 5) is a hierarchical structure that represents the system being debugged. See the box on page 46 for a discussion of hierarchies. Each node in the configuration, called an entity, is a frame[3] representing a component of the system.

Slots in the entity frame describe aspects of the state of that component. During the specification stage each slot contains the following information: data type, possible values, actual values, default value, and cost of determining value. During the troubleshooting session, each frame also holds the current beliefs regarding that aspect of the entity as well as other pertinent information such as how, when, and where those beliefs were obtained.

In addition to the normal slots, each entity has an operating summary slot named in-working-state. This slot reflects the state of all the slots in the entity that are operating. For example, a value of yes in the operating summary slot of the data link in Fig. 5 indicates that Schooner was able to assume that the data link, the phone link, and both modems are in an operating state.

The operating summary provides a powerful capability for describing test dependencies. A test whose requirements assume that the entire configuration was in a working state doesn't have to specify each individual entity to guarantee useful backtracking. This characteristic allows reference to the operability of the configuration at any functional level. In the example of Fig. 5, operating summary slots could be used to refer to the operability of the configuration, the data link, either modem, or in general, any entity in the configuration.

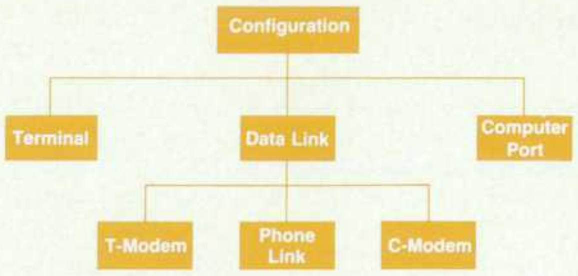


Fig. 5. *Configuration data structure.*

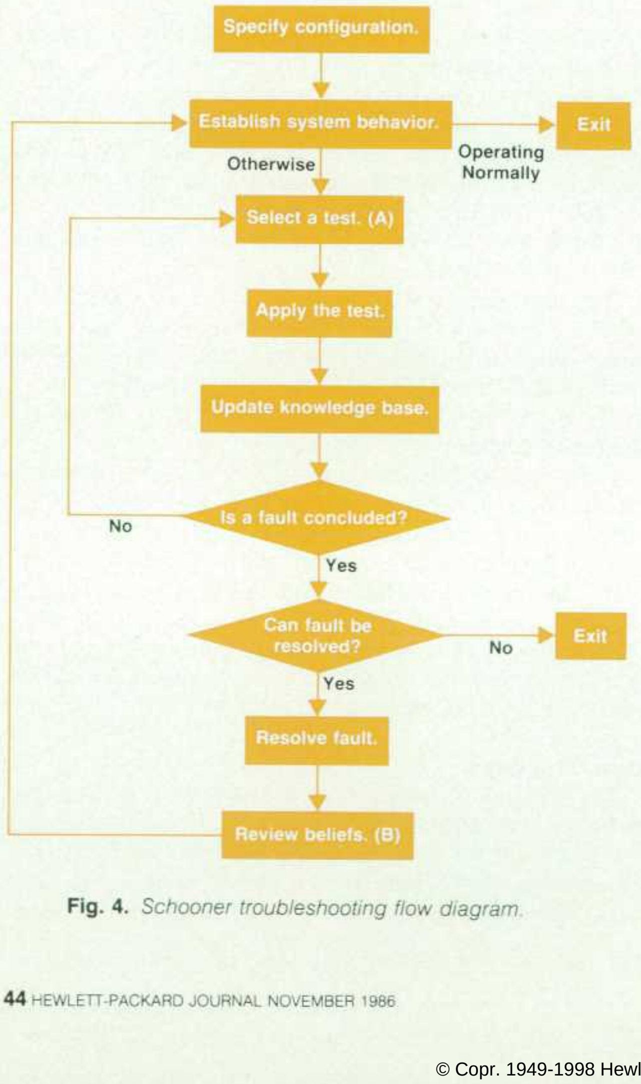


Fig. 4. *Schooner troubleshooting flow diagram.*

Faults. Faults are used for characterizing and manipulating failures that can occur in a system. They are also used for heuristic test selection (see below) and provide an explicit structure for holding current beliefs of what might be wrong in the system. During the specification stage, a fault simply contains a description and a list of tests that point to it (Fig. 6).

Faults are integrated into the configuration by their attachment to entities in the configuration. Faults are attached to the entity in which they occur (Fig. 7). For example, a fault representing a malfunction in the terminal would be attached to the terminal entity, but a fault representing a baud rate discrepancy between the terminal and the port would be attached to the configuration entity.

Tests. Tests are pointed to by faults. The outcome of a test can either confirm or deny a fault.

Linkage. The three data structures above describe the essentials necessary for troubleshooting a physical system. Schooner contains an internal description of the system being diagnosed, a set of beliefs of potential faults, and a way of describing tests to be performed on the system. The linkage data structure (Fig. 7) ties these three elements together to provide useful access when and where necessary. It provides links from an entity in the configuration to faults that may exist in the entity, and it provides links from a fault to tests that might tell about it. Once a test is applied, it finds the faults to which to assign status.

In summary, the main duties of the linkage data structure are to ensure knowledge consistency, guarantee uniqueness, and tie the whole thing together.

Fault:	Keyboard-Card-Disconnected
Frequency:	Rare
Description:	Is the keyboard card disconnected?
Resolved:	Plug-in-the-keyboard-card
Concluded:	Inspect-keyboard-card
Eliminated:	Do-the-cursor-control-keys-move the-cursor? Hit-carriage-return
Indicated:	Does-control-g-ring-bell?
End-Fault:	

Fig. 6. *An example of a fault in Schooner. This fault occurs when the card for the keyboard inside the terminal becomes loose in its slot.*

Heuristic Test Selection

With an understanding of the essentials of the Schooner data structures, it is now possible to describe more clearly some of the other processes in the Schooner troubleshooting flow diagram, Fig. 4.

The test selection process (the box marked A in Fig. 4) is critical for rational behavior and proficient troubleshooting. Poor selection of tests will irritate the user, make troubleshooting arduous, and create an impression of irrationality. This aspect of the system can single-handedly determine the difference between acceptability and unacceptability of an expert system.

Each state change or belief change caused by a system manipulation or test completion causes Schooner to do a complete reevaluation of all possible tests to ensure an action appropriate to the new situation (Fig. 8). Each potential fault in the system is given a numeric pursuit desirability, which is a combination of its frequency (a static value assigned at specification) and its likelihood (a dynamic value received from tests during the session). Each test that can further indicate or disindicate the fault is given a numeric cost (see below). Schooner then selects the test-fault combination that best combines the attributes of low user effort (in the test) with high pursuit desirability (in the fault).

The total cost of a test is the effort to do the query plus the effort required to put the system in a state to do the test. The resulting value is a function of the effort required to perform the test from the present state of the system. This technique for determining cost tends to minimize manipulations to the system by making Schooner take maximum advantage of the present state. For example, if the user is asked to run a diagnostic to look at the state of a port, the effort required to do a system dump will cause Schooner to obtain all the information possible from doing the dump before going to another area of investigation.

Review Beliefs

When a fault is discovered and subsequently resolved in the system being diagnosed, Schooner reviews the information and consequent beliefs that have accumulated during the session. Typically, some of the information gained from previous tests and observations is no longer valid. Schooner must find the invalid beliefs and retract them. The nature of the Schooner data structures makes this process fairly simple. Schooner merely has to determine (according to procedures described below) which tests are no longer valid and "disapply" them. This procedure is what occurs in the review beliefs process (the box marked B in Fig. 4).

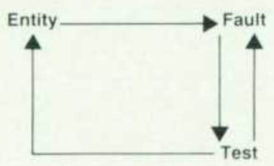


Fig. 7. *Linkage data structure. Faults are attached to an entity. A fault's likelihood results in a test's selection. The test manipulates the entity and infers about the fault.*

```
For each Fault in possible faults
    Evaluate Fault pursuit desirability
    For each Test that can tell about Fault
        Evaluate Test Cost
        Test-Desirability =
            Fault-Desirability/Test-Cost
    End For
End For
Select Test with Maximum Desirability
```

Fig. 8. *Schooner heuristic selection algorithm.*

When a test is disapplied, all its results are retracted, that is, all status assigned to faults as a result of applying the test is retracted. Additionally, the test is made available for reapplication.

There are two reasons for Schooner to disapply a test. The first occurs when the test indicated a fault that was subsequently resolved. Since the test results were based on the existence of that fault, such actions are necessary to maintain consistency. The reapplication of the test would be very likely to yield different results, since the fault has been remedied.

The second reason for disapplying a test occurs when the test depends on an assumption specified in a requirements clause that the discovery of the fault has contradicted. In the example test shown in Fig. 1, the clause

```
?term in-working-state yes
```

required an assumption, since it wouldn't be known that the terminal was in a working state. The subsequent discovery of a resolvable fault in the terminal (for example, being in local mode) would cause Schooner to instruct the user to resolve the fault. Schooner would then go into the review beliefs process, which would notice that the test do-local-loop-back-at-modem made an invalid assumption. All beliefs that the test asserted would be retracted and it would be made eligible for reapplication.

The resulting behavior is a natural sort of backtracking and attention control. As Schooner applies tests that indicate a failure in an area of the link, faults in this area are concentrated on. If a fault is discovered and resolved, then Schooner backtracks, verifying symptoms that were discovered earlier to determine the present characteristics of the link being diagnosed and to decide where to investigate next.

These dependencies are a natural representation of the use of tests as a unit of knowledge. There are no special requirements for the specification of dependencies. All the right information gets backed out when it should be.

Behavioral Results

Schooner has turned out to be a competent expert system in the domain in which it has been applied. It is effective at discovering faults for which it has knowledge. Knowledge acquisition and formalization, although cumbersome, allows Schooner more or less unlimited growth of expertise

Hierarchies

Hierarchies are a potent tool in artificial intelligence applications. A hierarchy is an n-ary tree with a root node. Each node is represented as a frame. The frames are connected by pointers that point from parent node to child. In a hierarchy, each parent (ascendant) node is a generalization of its descendant nodes. To explain this more clearly, this article will discuss two types of hierarchy.

Inheritance Hierarchies

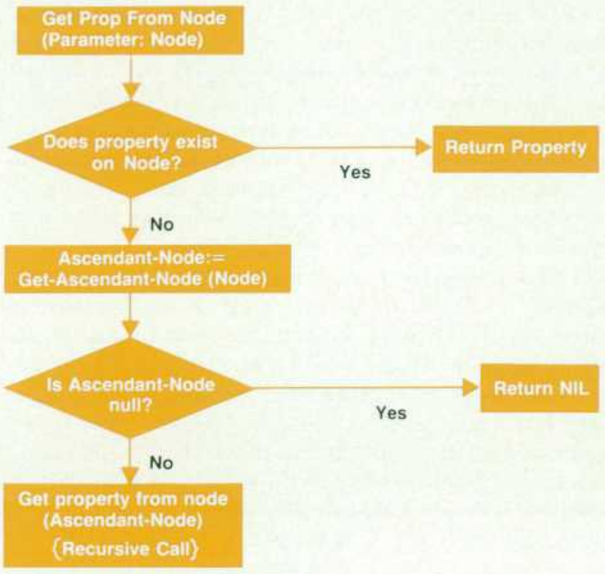
Inheritance hierarchies are used to describe objects in the real world. In an inheritance hierarchy (see Fig. 1), AKO (a kind of) tree, higher nodes on the tree represent more general categories while lower nodes are more specific. At each level in the hierarchy, nodes contain properties that distinguish them from their siblings. Information common to all siblings is inherited from parent nodes. If the user queries a property of a node in the hierarchy, access routines (Fig. 2) search for the requested information, looking first in the specified node and then in ascendant nodes.

For example, in Fig. 1, the mammal node would hold the property of being warm-blooded. Any queries about this property to the ape, dog, or bat nodes (or nodes descendant from them) would result in the correct value being returned.

This method for representing objects in the real world is used widely in artificial intelligence. It has proved to be a very powerful representation technique for several reasons. It provides enormous economy of data; each internal representation of every kind of bird does not contain information that it breathes, eats, mates, flies, etc. This information logically, conceptually, and sensibly belongs at higher levels in the hierarchy. Adding new objects is very easy. The addition of a new type of hawk called the kitty hawk would automatically result in the right sort of assumptions made about it—it flies, is carnivorous, etc. Only information that differentiates the kitty hawk from other types of hawks would actually have to be added to the kitty hawk node.

Functional Hierarchies

For troubleshooting, expert systems need sophisticated techniques for representing the systems they are debugging. Schooner and IPT (see article, page 48) represent devices they are debugging by creating an internal representation that mirrors the device's functional properties. Each device (see example in Fig. 2 on page 49) is shown as the union of its major subsystems. Each subsystem, in turn, is divided into smaller subsystems until it reaches the smallest size of component the troubleshooting system can reason about. This type of representation of a device is called a functional hierarchy.



Get Prop From Node (Parameter: Node)
Does property exist on Node?
Yes
Return Property
No
Ascendant-Node:= Get-Ascendant-Node (Node)
Is Ascendant-Node null?
Yes
Return NIL
No
Get property from node (Ascendant-Node) (Recursive Call)

Fig. 2. *When a user queries a property of a node, access routines search for the requested information.*

Functional hierarchies offer several advantages for device representation. First, they allow a divide-and-conquer troubleshooting methodology. Instead of verifying the operability of each subcomponent at the lowest level, the expert (or expert system) can determine the operability of one of the subsystems. Only when a fault has been isolated to a subsystem does the troubleshooter then investigate the nodes that lie below the subsystem.

Another advantage of functional hierarchies that is used by Schooner is that the operability of a node at any level can be viewed as the sum of the operability of all nodes descendant from it. Schooner automatically maintains this relationship, updating its beliefs of the operability of nodes at intermediate and upper levels based on the discovery of faults at lower levels. As a result, knowledge for Schooner can refer to the operability of a device with assurance of appropriate results. For example, if the electrical system in Fig. 2 on page 49 was determined to be in a working state, Schooner would merely have to assert that fact rather than eliminate all the faults in or below that node in the diagram. This presents a concise way of giving information and preserves accuracy even when faults are subsequently added or removed.

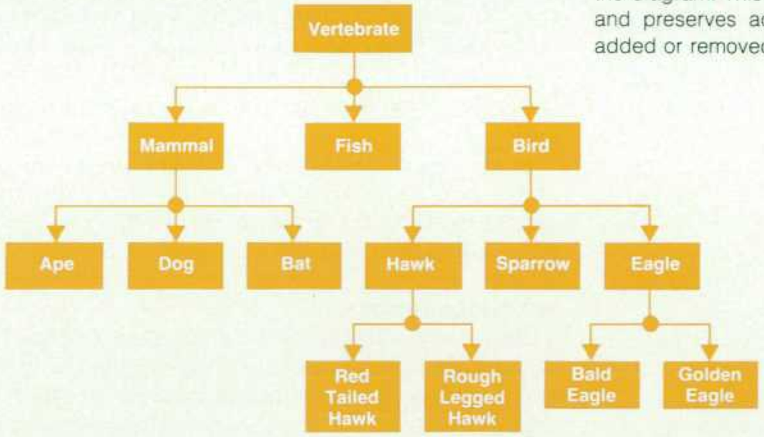


Fig. 1. *Animal characterization hierarchy.*

with few expected changes to the inference engine or knowledge representation technique.

The following transcript from a sample session demonstrates the knowledge base and provides a picture of Schooner's behavior with respect to the concepts described above.

Sample Session

This example demonstrates simple backtracking.

Problems: remote mode key up and terminal disconnected from cable.

```
                    WELCOME TO SCHOONER!

   <<Debugger for asynchronous, point to point, direct connect,
                     terminal problems.>>

Describe the type of problem you are experiencing.
→ inability-to-log-on
Device name for reference: PORT-3, CABLE-3 and TERMINAL-3
→ acknowledge
Using the cursor control key, move the cursor out from the left margin.
Depress the return key, what appears on the screen?
→ cr-only
Is TERMINAL-3's remote mode key in the down position?
→ no
Depress TERMINAL-3's remote mode key.
→ acknowledge
Using the cursor control key, move the cursor out from the left margin.
Depress the return key, what appears on the screen?
→ nothing
```

Schooner believed a problem had been solved. It then found out what the symptoms were and discovered a problem still existed and the symptoms are now different.

```
Is TERMINAL-3's block-mode key in the down position?
→ no
Hold down the ctrl key and press q.
→ acknowledge
Is any text coming up on the screen?
→ no
Turn off TERMINAL-3, wait 5 seconds, then turn it back on again.
→ acknowledge
Using the cursor control key, move the cursor out from the left margin.
Depress the return key, what appears on the screen?
→ nothing
What is displayed on the first line of TERMINAL-3?
→ terminal-ready
Is TERMINAL-3 connected securely to CABLE-3?
→ no
Connect TERMINAL-3 to CABLE-3.
→ acknowledge
Using the cursor control key, move the cursor out from the left margin.
Depress the return key, what appears on the screen?
→ lf-cr-and-colon
Problem(s):
    TERMINAL-DISCONNECTED
    TERMINAL-IN-LOCAL-MODE
resolved.
THANK-YOU
```

Acknowledgments

The authors would like to thank Michael Shorts, Jack Hymer, and Joe Gates for acting as domain experts during the development of this product. Their combined expertise forms the knowledge base that allows Schooner and its users to troubleshoot asynchronous datacom problems.

References

1. *Building Expert Systems*, F. Hayes-Roth, D.A. Waterman, and D.B. Lenat, eds., Addison-Wesley, 1983.
2. J. Doyle, "A Truth Maintenance System," *Artificial Intelligence*, Vol. 12, no. 3, 1979.
3. M. Minsky, "Steps toward Artificial Intelligence," *Computers and Thought*, E.A. Feigenbaum and J. Feldman, eds., McGraw-Hill, pp. 406-450.

A Rule-Based System to Diagnose Malfunctioning Computer Peripherals

The Intelligent Peripheral Troubleshooter, an expert system, currently diagnoses malfunctions in HP disc drives, but other devices will be easy to add to its repertoire.

by George R. Gottschalk and Roy M. Vandoorn

AS BUSINESSES PLACE more and more reliance on their computer systems, downtime has an ever greater impact on their profits. In addition to the efforts of HP product divisions to make HP products more reliable, HP's Knowledge Systems Laboratory has taken a dual approach to minimizing customer downtime, aiming first to predict failures before they happen,[1] and second, to improve the accuracy of the diagnosis.

To diagnose malfunctioning peripherals more effectively, the Intelligent Peripheral Troubleshooter (IPT) has been developed. IPT is an expert system that performs a diagnosis of a malfunctioning peripheral based on aspects of its current state.

IPT is geared to work in three different environments. It can be used as a Response Center tool, as an aid to the HP service person on-site as an escalation tool, and as a training aid. When a customer places a service call, an engineer located in one of the Response Centers contacts the customer to collect as much information as possible about the problem and determines if an on-site visit is actually required. During this session, IPT is used to determine the most likely cause of the problem. This helps ensure that the service person going on-site will have the correct parts, thus reducing the number of multiple-trip fixes.

Once an HP service engineer is on-site, IPT may request more precise information that can only be obtained by a trained engineer. Examples are using a disc service unit, measuring voltages across test points, and measuring seek times. Like human experts, IPT will be able to make a more precise diagnosis if it is given this additional information. IPT can also be used by the Customer Engineer Assist Group in the Response Center when their expertise is required because of problem escalation.

For any expert system that attacks a class of problems to be successful, enough generality must be built in so that an entirely new system is not necessary for every variation.[2] For IPT to be successful, the underlying approach must be general enough to be able to deal with all HP peripherals. For this reason, IPT has the following three characteristics:

A peripheral independent inference engine

A system and component representation of a peripheral

A knowledge base built by an interactive rule-maker.

IPT follows the classic expert system model of separation of knowledge and the inference engine as shown in Fig. 1. This allows the inference engine to be totally device independent.

Given the relevant knowledge, IPT could troubleshoot a wide range of devices, even household appliances such as toasters. In this paper, all examples will deal with a fictional toaster. This substitution enables this paper to ignore the technical details of disc drives.

What Fig. 1 does not show is that there are actually three types of knowledge, which are directly correlated with the three phases of the inference engine. The three phases can be thought of as: narrow the problem down to a logical group of failures, explore each of these failures to see which is most likely, and perform the repair.

IPT Knowledge Representation

In previous attempts to develop systems that could troubleshoot malfunctions, commonly referred to as symptom-cure systems, the diagnoses were reached by a best-fit correlation between the symptoms and the possible cures. For IPT to be a useful tool, it needed to be able to deduce the problem much like a human expert. The difficulty, as in all expert system development, was to determine what information is actually useful and how to represent it.

It turns out that describing troubleshooting in terms of symptoms and cures is too limiting. To describe the process better, components and observables are more appropriate terms. Components are field replaceable or adjustable units. Observables are determinable states of the device, such as the state of the bread after toasting, and the setting of the knob.

In troubleshooting scripts, isolating the failing component is not a one-step process. Therefore, the components are grouped into their logical systems. For example, in the

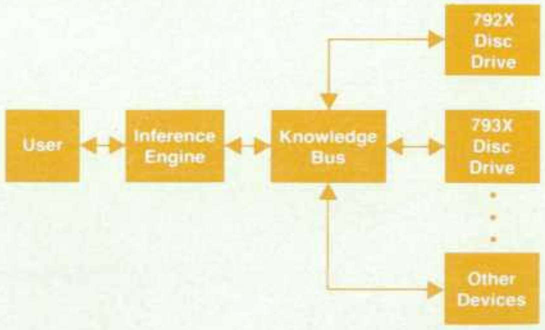


Fig. 1. *Separation of IPT's knowledge base and inference engine.*

toaster diagrammed in Fig. 2 there are three systems: the electrical system, the mechanical system, and the timer system. One of the interesting features of systems is that one system can be completely contained within another system. For example, the components of the timer system (the clock and the timer knob) are both completely contained within the mechanical system. (For simplicity we will assume that the timer system will work even if the toaster is not plugged in.) Thus, depending on our goals, we could solve for the clock as an element of either the mechanical system or the timer system. Systems are the logical groups that IPT solves for in phase 1 of a troubleshooting operation.

There are distinct advantages in using systems to group components. First of all, the rules are easier to enter because we can think of observables as indicating a typical electrical problem rather than specifying the effects on every component within the electrical system. Next, using systems reduces our search space when we try to solve for a component. If IPT believes a problem is in the electrical system, for example, it won't waste its time asking questions about the timer. IPT will only go after those problems it considers interesting enough to solve.

A further advantage is that this representation brings us one step closer to the real world. Artificial intelligence applications often get their results by performing clever numerical or logical manipulations that have no underlying theory in reality. We can say, however, that IPT understands that the clock is part of the timer system and that it is related to the mechanics of the toaster. Although IPT, of course, has no real grasp of the concept of a toaster, and the words clock and timer have no meaning to IPT, it is hoped that someday IPT will be able to diagnose hardware by looking at a device on a functional level. Getting IPT to recognize that certain components are related in a special way is a good start.

Most observables point to multiple components and each component is implied by multiple observables. When an observable does point to multiple components, it is possible to order the components in order of likelihood. A ranking strategy was developed to reflect this relationship:

Always	Given that this observable occurs, then this component will always be at fault.
Consistently	Given that this observable occurs, then this component will consistently be at fault.
Frequently	Given that this observable occurs, then this component will frequently be at fault.
Occasionally	Given that this observable occurs, then this component will occasionally be at fault.
Rarely	Given that this observable occurs, then this component will rarely be at fault.

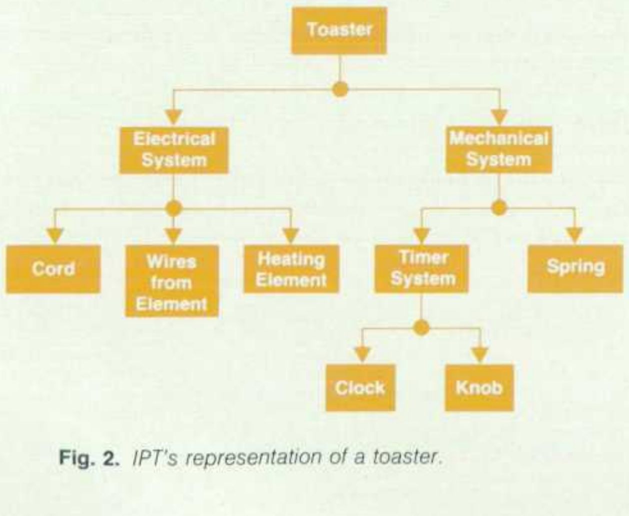


Fig. 2. *IPT's representation of a toaster.*

Words are used to describe the categories to the experts since it is much more natural than dealing with probabilities. Computers, however, are designed to think in terms of exact numbers, not vague terms. Therefore, it was necessary to convert the logical categories to probabilities. IPT actually goes one step further and uses certainty ratios.[3]

The advantages of certainty ratios are:

- Comparisons are manipulated more easily
- All observables are weighted equally
- Certainty ratios are easier to maintain
- Results are mathematically correct.

The certainty ratios are hidden from the user, since intuitive probabilities are easier for humans to understand. A 70% probability means something, but a corresponding certainty ratio of 2.33 has no intuitive meaning. IPT uses probability to rank components and systems. It does not consider probabilities to be an absolute measure of certainty. Thus the diagnosis may not necessarily be halted when a component reaches a probability level of 99%. Instead, probability is used to choose the most likely path to investigate and to halt diagnosis when one component is significantly more indicated than all the other components. This method has the major advantage that as long as the relative ranking of the components is correct, the exact category in which the component appears is not very important. This approach also has the consequence that when we can use multiple experts, the original work of the first expert does not need to be totally redone when the next expert adds knowledge.

Nearly every time something new is observed, that observation affects the likelihood that some system or component is at fault. Thus every time an observable is determined, the certainty ratios should be updated. Therefore, at any time during the diagnosis, it is possible to step away and see how strongly IPT believes a particular component is at fault.

The effort involved in determining the state of an observable can range from trivial to major. It is important that during the troubleshooting script the easy questions are asked first and the more difficult questions are asked only as a last resort. The categories IPT uses to sort observables are: trival, easy, hard, and very hard. The guidelines used for the categories are:

Trival	Can be done by looking at the peripheral
Easy	Can be done by looking at the logs
Hard	Can be done by running a diagnostic
Very Hard	Requires swapping out a board.

These difficulty rankings are called the *cost* of the observable.

Within the three types of knowledge, both forward chaining and backward chaining rules are used. The first type of knowledge, which is used to determine the system at fault, is called initial observables. These rules can be thought of as the reason the user perceives there is a problem. They are implemented via forward chaining rules because forward chaining rules provide the control for the follow-up questions. Examples of initial observables for a toaster are:

Toast pops up but cold
Toast won't pop up and no heat
Toast imitating charcoal

If the user selects the last choice, toast imitating charcoal, then we want to investigate the mechanical system. By examining the toaster diagram, Fig. 2, it can be seen that the determination is between the timer system and the spring. Since the objective of this phase is to determine the system in which the fault is located, for this example this phase is completed. If the user selected toast pops up but cold, follow-up questioning is necessary to determine the faulty system.

The second phase attempts to determine the faulty component. Associated with all components are backward chaining rules. Each of these rules within the system determined in phase 1 needs to be investigated. The difficulty ranking for the observables now determines which rules are solved for first. In our example, it was agreed that the cost of checking the setting of the knob is trival; therefore, that is the first line of reasoning.

Associated with each backward chaining rule is a forward chaining rule that causes the certainty ratios to be updated based on the user's response. Forward chaining rules are used here since an unexpected response from the user may satisfy another rule. Since forward chaining rules automatically fire, the certainty ratios are properly updated. In other words, these rules have been implemented so that the backward chaining rule currently being attempted need not correspond to the forward chaining rule that ultimately fires.

Once the diagnosis has been completed, IPT instructs the user how to proceed with the repair. These are called hint rules. Hint rules are forward chaining rules and their premises are either an indicated component or a set of observables.

The next section examines how the inference engine uses these three types of rules.

Device Independent Inference Engine

IPT is based on the troubleshooting strategy of actual experts. As discussed earlier, troubleshooting is not a one-step process. Our experts find it easier to solve problems by thinking of components in terms of a system. For instance, if the heating element of a toaster does not turn red, then all components that have to do with electrical connections are suspect. Solving the problem now becomes easier if we concern ourselves with only those components.

Although much of the information is device specific, an underlying core of knowledge about the troubleshooting of devices exists. This core was coded into the IPT inference engine.

Several terms have been used to describe the code portion of IPT. It has been called the driver, the engine, and the inference engine. What all of these terms have in common is that they imply that the engine is the force that underlies IPT, and in this respect they are accurate. IPT's knowledge of a device is merely data. It is the driver that manipulates that data to derive a diagnosis. As we saw earlier, looking directly at the knowledge is similiar to looking at a dump of a data base. It does not have meaning until the engine interprets the data. Therefore, the only connection the user has with IPT is through the inference engine.

Of course, if the inference engine accessed the knowledge by name, it wouldn't be device independent. Instead, we have defined a protocol that allows us to access the knowledge indirectly. The foundation for this protocol was provided by HP-RL (Hewlett-Packard Representation Language[4,5]) and with our extensions constitutes the knowledge bus. At present, however, the knowledge bus is very primitive and is dedicated to one device at a time. All of this complicated structure does have a purpose: it allows IPT to be device independent except for the knowledge.

There are many conflicting demands on IPT's engine. The major concern was to limit the search space as quickly as possible by eliminating unnecessary questions and asking the remaining pertinent questions in a logical order. To implement this strategy, IPT's engine is split into three distinct phases, as shown in Fig. 3.

The first phase is the system determination phase. The purpose of this phase is to ask general, easy questions until either a component or a system is indicated. If a component is indicated, then the third phase is entered and the second phase is skipped.

If a component is not indicated, then IPT enters the second phase. Since a system is by definition indicated at the start of the second phase, the purpose of this phase is to determine the component at fault. Because the component determination phase is much more confident of the area to search, it can begin asking harder questions of the user.

Finally, the repair procedure phase is entered. This phase generates output from IPT, such as repair hints, the relative standings of the various components and systems, and a recap of the diagnosis.

One of the important points to keep in mind is the difference in the paths between IPT and traditional troubleshooting flowcharts (see Fig. 4). A traditional troubleshooting flowchart emphasizes tests that tend to eliminate areas of search space very slowly, and that perform exhaustive searches on the components in question. IPT, on the other hand, maintains its flowcharts in a dynamic environment, and can choose its next question based on the latest information. It can also choose to perform the easiest possible test that will solve for a particular component. Maintaining this type of flexibility is an important factor in keeping IPT

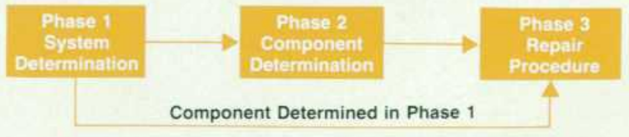


Fig. 3. *IPT's troubleshooting strategy.*

efficient.

Phase 1: System Determination

The first phase that IPT enters is the system determination phase. This phase tries to indicate either a system or a component by collecting broad and easily gathered information. What IPT is trying to do is to get an idea of where the problem lies. To do this, IPT maintains a list of questions to be asked. Forward chain rules are attached to the answer of each question so that the probability levels of the various components and systems are updated after each question.

Let's assume that a customer calls the Response Center with a problem. In this case, IPT should try to emulate what a customer engineer does when the CE first enters the customer's kitchen. A good example would be to ask

Which of these best describes the problem with your toaster?
- The toast won't pop up and no heat
- The toast pops up but is cold
- Toast imitating charcoal.

Now let's suppose that the customer indicates that the problem involves toast that pops up but is cold. When this observation is entered into IPT, the forward chain rule associated with cold toast is fired. Since the toast does in fact pop up, the probability that the problem is with the spring is downgraded. Furthermore, the electrical and timer systems have added support. At this time, however, neither system is indicated enough to leave this phase and search for components.

Instead, the time has come to search for the next question. The question of whether a fork will pop up the toast is discarded because we know that the toast has already popped up, and IPT only asks necessary questions. IPT then finds that the next thing to ask is

Does the heating element glow bright orange when toast is inserted in the toaster?
- Yes
- No

Let's assume that the customer says the heating element does not glow. The fact that the heating element does not glow seems to indicate the problem is in the electrical system, and that is sufficient information to leave this phase and enter the component indication phase.

Although toasters are much simpler devices than disc drives, this question is typical of those asked in this phase. In the case of disc drives, of course, the user will also be asked to obtain the logs so a history of the drive's errors can be reconstructed. The general idea of this section is to gather only enough information to point to a specific area and then leave. Therefore, it is not appropriate to ask the customer to take an ohmmeter and measure the resistance across the heating element. Also, these questions will not usually be able to solve uniquely for a component but will only affect the probability levels. These types of behavior are left for the next phase.

Phase 2: Component Determination

The purpose of the component determination phase is to solve for a failing component given an indicated system. Of course, this is not always possible, so this phase does have the ability to exit given certain conditions. What really happens is that the component determination phase attempts to solve for a unique component, and if the chain

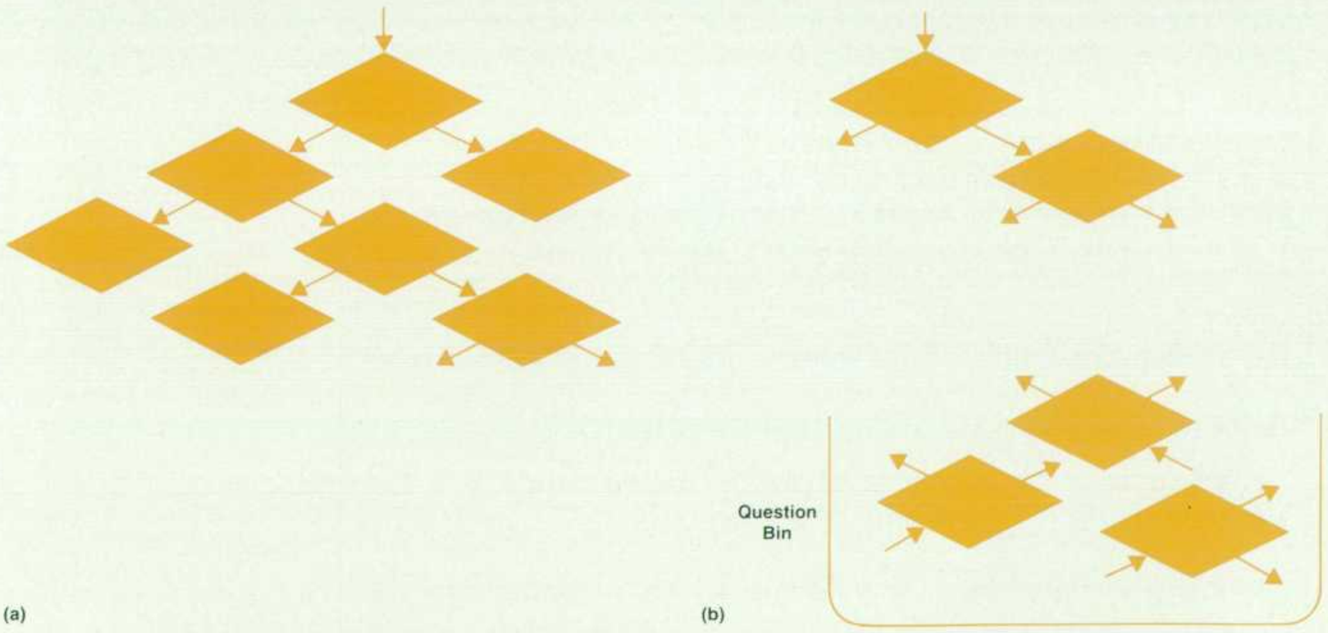


Fig. 4. *(a) In a traditional troubleshooting tree, the questions are set up in advance and are selected solely by the answers to previous questions. (b) IPT's dynamic troubleshooting tree chooses the next question from the question bin based on both the current environment and the answers to previous questions.*

of reasoning used to attempt the solve is completed, then a conclusion is asserted.

This is why it was decided to use backward chaining rules for this part of the knowledge. Within the indicated system, each component is sorted according to its probability factor. Subject to a difficulty ranking of the observable, these components are attempted from the most likely down to the least likely via backward chaining rules. When a component is "proved" to be at fault, phase 2 is completed.

In the toaster example after phase 1, the electrical system was indicated because the heating element did not glow. Clearly, IPT should now determine what is wrong with the electrical system. Let's assume that IPT has the following rules concerning the electrical system:

```
Cord-Rule: IF the cord is not plugged in
           THEN the problem is in the cord
           DIFFICULTY RANKING—Trival

Heating-Element-Rule: IF the cord is plugged in
                      AND IF there is infinite resistance across
                        the heating element
                      THEN the problem is in the heating element
                      DIFFICULTY RANKING—Hard

Heating-Element-Rule-1: IF the cord is plugged in
                        AND IF the customer declines to state
                          what the resistance across the
                          heating element is
                        THEN the problem is in the heating
                          element or the wires connecting the
                          heating element to the cord
                        DIFFICULTY RANKING—Hard
```

Finally, let's assume that at this point the heating element is slightly more indicated than the power cord.

First of all, IPT examines the three rules. Because the heating element is more indicated, it would like to use one of the heating element rules, but the difficulty ranking implies that it should ask the cord rule first because it is easier to ask.

```
Is the toaster plugged in?
  Yes
  No
```

If the cord wasn't plugged in then IPT would be able to conclude that the cord should have been plugged in.

To make things more interesting, however, assume that the cord was plugged in. Next, IPT examines the Heating-Element-Rule and sees that the first premise is satisfied.

```
Using an ohmmeter please measure the resistance across
  the heating element.
Is the resistance infinite?
    Yes
    No
    Dont know
```

If the user answers that the resistance is infinite the rule is satisfied and this phase is completed.

Suppose, on the other hand, that the user didn't have an ohmmeter handy. At this point the Heating-Element-Rule-1 is satisfied and gets asserted, even though IPT wasn't considering this rule at the time. This flexibility illustrates why there is no logical connection between the forward chaining and backward chaining rules.

The component determination phase is crucial to IPT's success. Using backward chaining rules for agenda control allows us to eliminate large areas of the search space. Because we have narrowed down the possibilities so far, we feel confident in asking hard questions, and can isolate components to a high degree of certainty. Finally, IPT maintains a great deal of control over when rules are attempted

```
+---------------------------------------------------------------------------+
|                                                                           |
| IPT was unable to obtain an exact diagnosis                               |
| It did, however, find some likely candidates                              |
| %%%%%%%%%%%%%%%%%%%%%%%%%%% REPAIR PROCEDURES %%%%%%%%%%%%%%%%%%%%%%%%%%%%% |
| If the toaster has a serial number below 687, then verify that the        |
| orignal 10 amp fuse has been replaced with a 20 amp fuse.                 |
|                                                                           |
|                                                                           |
|                                                                           |
| %%%%%%%%%%%%%%%%%%%%%%%% COMPONENT PROBABILITIES %%%%%%%%%%%%%%%%%%%%%%%%%% |
|                                                                           |
|                                                                           |
| The following components have the indicated probability                   |
|                                                                           |
|     VERY LIKELY              LIKELY             SOMEWHAT LIKELY           |
|                                                                           |
| WIRES-FROM-ELEMENT       HEATING-ELEMENT                                  |
|                                                                           |
|                                                                           |
| %%%%%%%%%%%%%%%%%%%%%%%%%% SYSTEM PROBABILITIES %%%%%%%%%%%%%%%%%%%%%%%%%%% |
|                                                                           |
| The following systems have the indicated probability                      |
|      VERY LIKELY              LIKELY             SOMEWHAT LIKELY          |
|                                                                           |
| ELECTRICAL-SYSTEM                                                         |
|                                                                           |
|                                                                           |
+---------------------------------------------------------------------------+
```

Fig. 5. *An IPT diagnosis.*

so that the cost of performing a test and the present likelihood of a component are both taken into account.

Phase 3: Repair Procedure

When the third phase is entered, IPT's diagnostic process is over. The purpose of the repair procedure phase is to take the information gathered during the diagnosis and print it out in a usable format. In most traditional troubleshooting programs this phase is usually trivial; a message or a number may be output on some device. In IPT's case, however, there may be certain instances where additional reasoning must be performed.

The first thing that is printed out is a list of repair hints. Repair hints may contain the necessary steps for a repair, additional probability information, safety hints, or relevant information from a service note.

Assume in the toaster example that all toasters with a serial number below 687 were built with a 10-ampere fuse instead of a 20-ampere fuse. Because so few of the toasters now have the 10-ampere fuse, it wouldn't make sense to include this as a component determination rule. Instead, we decide to make it a hint rule that only prints out when we haven't isolated the problem but know it's in the electrical system. Fig. 5 shows what the IPT output might look like.

The important thing about hint rules is the intelligence they must display. If IPT were to print out all of the hint rules for a particular device all of the time, there would be so many that the user would refuse to read them. By making hint rules fire only under certain conditions, IPT is able to make this portion of the output both important and concise.

If the component determination phase was successful, the next item to be printed out is the indicated diagnosis. Since the component information phase cannot always uniquely determine a component, after the indicated diagnosis is a list of components grouped according to their probability. The components are divided into three groups: most likely, likely, and somewhat likely. These groupings indicate the order in which IPT would like the components to be considered.

This time, however, the grouping also takes the difficulty of repair into account. For example, in the diagnosis shown in Fig. 5, the heating element is probably more likely than the wires connecting the element to the cord, but since checking the wires requires only a visual inspection, IPT would suggest checking the wires first.

Finally, the sections that illuminate IPT's reasoning process are printed. The first of these sections is the probability ratings of the systems. Next, the justification of any component determination rules is output. Finally, a recap of the observations entered into the system is printed. The purpose of these three sections is to give the users of IPT a way of determining why IPT made its diagnosis.

Conclusion

The Intelligent Peripherial Troubleshooter has proved the feasibility of developing knowledge-based systems to troubleshoot peripherals. Currently its diagnostic scope—it can diagnose HP 792X and HP 793X Disc Drives—is limited, but IPT will grow to cover more and more devices. The device independent inference engine makes it possible to add devices without the need of developing an entirely new strategy.

Acknowledgments

The authors would like to thank Phil Haseltine and Dick Schafer, North American Response Center domain experts. Without the expertise supplied by these people and others, IPT would not have the knowledge necessary to do effective troubleshooting of disc drives.

References

1. D.B. Wasmuth and B.J. Richards, "Predictive Support: Anticipating Computer Hardware Failures," *this issue.*
2. F. Hayes-Roth, D.A. Waterman, and D.B. Lenat, eds., *Building Expert Systems*, Addison-Wesley, 1983.
3. P.H. Winston, *Artificial Intelligence*, Addison-Wesley, 1984, p. 191.
4. S.T. Rosenberg, "HP-RL: A Language for Building Expert Systems," *Proceedings of IJCAI*, Karlsruhe, 1983, pp. 215-217.
5. M.R. Cagan, "An Introduction to Hewlett-Packard's AI Workstation Technology," *Hewlett-Packard Journal*, Vol. 37, no. 3, March 1986.

Multilevel Constraint Based Configuration

The goal of Mycon, a prototype expert system for configuring computer systems, is to relieve the support engineer of the tedious task of configuration of a customer order.

by Robert I. Marcus

A NEW ALGORITHM for handling computer system configuration problems has been implemented in a prototype called Mycon developed at the Hewlett-Packard Knowledge Systems Laboratory.

The original Mycon was written in Portable Standard Lisp using the HP AI Workstation environment[1] consisting of an objects package, a frame-based package, and an EMACS-type editor,[2] NMODE. This enviroment made it possible to construct an interface in which all the knowledge about the devices, constraints, and configurations of a system is easily accessible to the user. This first prototype focused on placing devices on I/O channels.

This prototype has been converted into HP standard Pascal and is running on the HP 3000, configuring the HP 3000 Series 68. The latest version also performs slot, junction panel, and cable configuration. The ultimate goal is to produce a program that can free the support engineer from the tedious task of logical configuration of a customer order.

The fundamental idea of the Mycon algorithm is very simple, but the algorithm appears to have a wide range of applications. The basic principle is that at every stage of the configuration process there are explicit constraints that cannot be violated.

The basic problem solved by Mycon is to construct a system configuration that satisfies a series of constraints supplied by an expert. The constraints fall into two different classes: required and suggested. A configuration is feasible only if it satisfies all of the required constraints. The suggested constraints are guides for selecting the best choice among the feasible configurations.

The prototype performs the logical configuration of a computer system. This problem consists of taking a set of devices and attaching them to I/O channels to maximize performance at a reasonable cost. Typical required constraints are The maximum number of devices that can be attached to a channel is 6 and High-speed devices must be attached to high-speed channels. Typical suggested constraints are Spread the high-speed devices evenly and Keep low-speed devices on low-speed channels. The difficult part of the problem is to use the knowledge embedded in the suggested constraints to control and evaluate the system configuration. The method of multilevel constraints was developed to handle this part of the problem.

Previous work on configuration problems, such as the R1 system developed by Digital Equipment Corporation[3] has used forward chaining rules to guide the configuration. The advantage of a rule-based or constraint-based system is the ease of maintenance and updating. In this paper, multilevel constraints are discussed as an alternative or supplement to forward chaining. The method can be extended to any problem in which there are qualitative preferences among a range of feasible solutions.

Basic Method as Used by Mycon

The steps in the basic method are:

1. Choose an initial algorithm that can generate candidate partial configurations as steps towards a final configuration.
2. Use the required constraints as a pruning mechanism to ensure that the initial algorithm will produce only feasible configurations.
3. Add suggested constraints as necessary to maximize the quality of the final configuration and to guide the configuration away from unsuccessful partial configuration that will require backtracking.
4. If the initial algorithm generates a sufficient number of possibilities, and all the required constraints are stated, and the suggested constraints are chosen well, then the method provides an efficient algorithm for optimal configuration.

The idea of multilevel constraints is to assign each constraint to a level determining when it is active and must be satisfied. In Mycon, the required constraints are assigned a level of 100. Suggested constraints are assigned lower levels ranging from 0 to 90 in steps of 10. The program begins attaching devices to channels keeping all constraints active. If a feasible configuration cannot be completed, then constraints at level 0 are dropped, the level is incremented by 10, and an attempt is made to attach the remaining devices. This process of dropping constraints is continued until the configuration is successfully completed or the level reaches 100.

This method offers several advantages over a strictly forward chaining approach. The number of constraints is smaller than would be necessary with forward chaining. It is not necessary to introduce additional predicates to resolve conflicts and to prevent the premature firing of rules. It is possible to evaluate the results of the configuration by keeping track of the suggested constraints it violated. Finally, the user of the system can suggest an alternate configuration for the program to evaluate and compare with the original result. This last feature is very important, because it allows a user to determine why the program has selected a particular configuration and which alternatives are equally suitable.

The Mycon Prototype

The user interface to Mycon is an interactive version of the form used by salespersons when taking a customer order. The form will only accept orders that satisfy all the basic system requirements regarding maximum and minimum numbers of devices and peripherals. Mycon can also determine if an order contains too many devices to be configured legally and will advise the user on how to reduce the order.

When a correct order has been entered, the program chooses the number of high-speed I/O channels and the system disc following guidelines suggested by an expert in configuration. The user is informed of the reasons governing these choices and is allowed to overrule them.

Next, Mycon attempts to place devices on the I/O channels using the active constraints to determine if the placement can be done. The devices are arranged in an order based on heuristics used by experts. In general, devices that are subject to more constraints, such as the system disc, are placed first. The user is able to watch the devices being placed one at a time with levels reported if desired.

A device is placed on the first I/O channel for which no active constraint is violated. If a device can't be placed at a level, then no further attempt is made at that level to place devices of the same type. When all devices possible have been attached at a given level, Mycon checks to see if the configuration is completed. If any devices remain to be placed, the level is raised and constraints below the current level are dropped. If the level reaches 100 without a successful configuration, the user is informed that it is necessary to reduce the number of high-speed I/O channels, which will lower the performance of the system. If the user decides to continue, the number of high-speed I/O channels is reduced, all devices are removed from the channels, the level is set at 0, and the configuration is restarted.

This process is repeated until a successful configuration is produced or no further reduction is possible. The last possibility has been eliminated in practice by using a maximal packing estimator on the configurability of the original order before starting configuration.

Finally, the user can reconfigure the system by adding, moving, and deleting devices. Mycon will evaluate the user's configuration and compare it with the original result.

Examples of Multilevel Constraints

Multilevel constraints serve a dual purpose in configuration. In one sense they are a means of knowledge representation that is easily understood. The levels convey an estimate of the importance attached to satisfying a constraint. However, the levels are also a control mechanism which can be used to govern the dynamics of the configuration process. For example, to ensure that high-speed devices are spread evenly on high-speed channels, the following constraints and levels are used:

Maximum of one device on high-speed channels	Level = 10
Maximum of two devices on high-speed channels	Level = 20
Maximum of three devices on high-speed channels	Level = 30
Maximum of four devices on high-speed channels	Level = 40
Maximum of five devices on high-speed channels	Level = 50

A fundamental constraint of the system is:

Maximum of six devices on high-speed channels Level = 100

The placement of a low-speed device on a high-speed channel at an early stage of the configuration would not strongly affect the performance of a successful configuration. However, it might stop the later placement of a high-speed device and prevent the configuration process from succeeding. Therefore, it is important that no low-speed devices be placed on high-speed channels until all high-speed devices have been attached. Since the constraints above may prevent this until the level is greater than 50, the following constraint is enforced:

No low-speed devices on high-speed channels Level = 60

It is possible to have Mycon maintain its knowledge base by insisting that certain inequality relations between constraint levels be enforced. For example:

Level of (No low-speed devices on high-speed channels) > level of (Maximum of five devices on high-speed channels).

The constraints can be chosen to avoid having devices placed on the same channel. For example:

Minimum of (Number of system discs attached to channel) AND (Number of magnetic tape drives attached to channel) = 0. Level = 90
(Number of high-speed master discs attached to channel) <= 1. Level = 30

It is also possible to insist that certain devices not be placed with those of another type. For example:

Number of slave discs attached to channel = 0 OR Number of master discs attached to channel = 0. Level = 100

Observations

It has been possible to translate all of the required and suggested constraints for device placement into the form of multilevel constraints easily. The selection of levels for the suggested constraints required some analysis and experimentation. The easy-to-understand form of the constraints facilitates addition, deletion, modification, and testing by a configuration expert untrained in programming.

The expansion of the prototype to handle placement into slots and junction panels greatly increased the knowledge required by Mycon. This was accommodated by adding constraints to the device placement phase of the program. Using these additional constraints made possible a procedural implementation of the slot and junction panel configuration.

The use of constraints also permitted an easy transformation of the program from Lisp to Pascal. Using Lisp as a development language hastened development of the prototype by providing flexibility in choosing data structures and an excellent debugging environment. The conversion into Pascal was done for reasons of portability and maintainability.

Conclusion

The method of multilevel constraints has proven very effective for logical configuration of computer systems. It combines simplicity and power in knowledge representation and control. The method can be extended to many problems in which a system is constructed systematically using knowledge of a domain. It can be used as a supplement or alternative to procedural or forward chaining methods. The constraints complement these methods by telling the program what actions not to take.

Constraints seem to be a natural way for experts to express their knowledge of a domain. By tracing the constraints and the levels it is possible to monitor very closely the progress of the configuration and detect gaps or errors in the knowledge.

The real power of the algorithm has not been fully tested by the logical configuration problem. For example, expressing multilevel constraints in mathematical form allows the introduction into expert systems of the tremendous amount of software constructed for linear and mathematical programming. In the field of artificial intelligence, there has been extensive work on constraint satisfaction and propagation; this could be used to extend the present algorithm. In the future, it seems possible that the method of multilevel constraints could become a standard tool of knowledge engineering.

Acknowledgments

Keith Harrison introduced me to the configuration problem and suggested many improvements in the user interface.

References

1. M.R. Cagan, "An Introduction to Hewlett-Packard's AI Workstation Technology," *Hewlett-Packard Journal*, Vol. 37, no. 3, March 1986.
2. R.M. Stallman, *EMACS—The Extensible, Customizable, Self-Documenting Display Editor*, Memo No. 519, Artificial Intelligence Laboratory, Massachusetts Institute of Technology, June 1979.
3. J. McDermott, "R-1: A Rule-Based Configurer of Computer Systems," *Artificial Intelligence*, Vol. 11, no. 1, 1982.

HEWLETT·PACKARD JOURNAL

November 1986 Volume 37 • Number 11

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

HEWLETT PACKARD

5953-8554