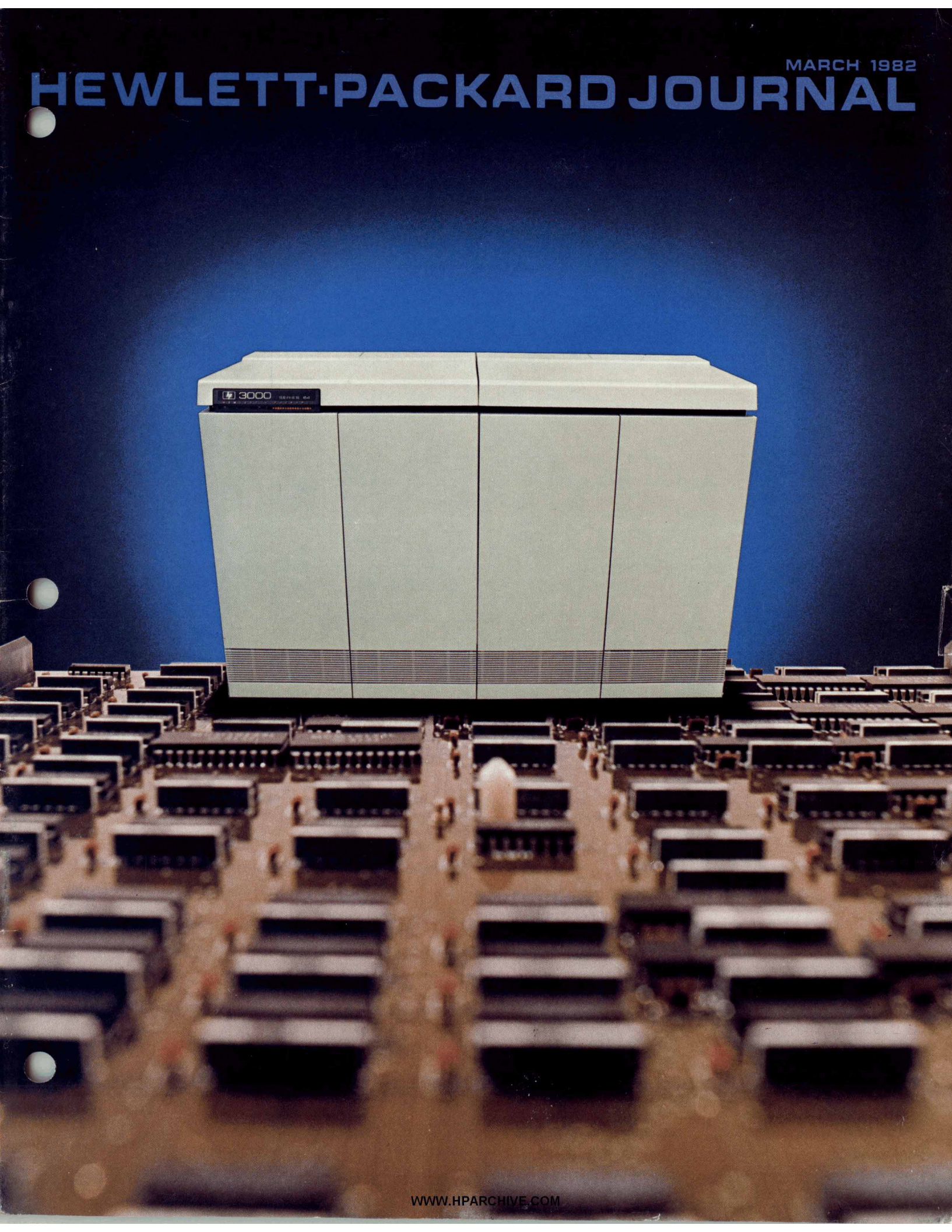
MARCH 1982

HEWLETT-PACKARD JOURNAL

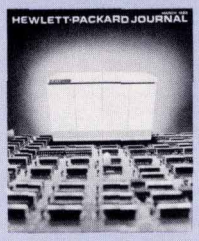


HEWLETT-PACKARD JOURNAL
Technical Information from the Laboratories of Hewlett-Packard Company

MARCH 1982 Volume 33 • Number 3

Contents:

In this Issue:



In the world of computers, faster is generally better, everything else being equal. A computer that executes more instructions per second can process more data in a given amount of time, serve more users at the same time, and respond more quickly to changes in its input data. Again, everything else being equal, a computer that executes more instructions per second is a more powerful machine and therefore a "larger" computer, although it may be physically smaller than some slower and less capable machine, and may cost less, too, since over the years advancing technology has steadily given us larger and larger computers for the same amount of money.

The subject of this month's issue is Hewlett-Packard's largest computer, the HP 3000 Series 64. Large enough to handle the entire data processing needs of a good-sized company, this newest member of HP's business computer family has 2½ times the processing power of the HP 3000 Series 44, previously HP's largest, and nearly ten times the power of the HP 3000 Series 30, the smallest member of the family. For example, the Series 64 can have 144 terminals attached to it, while the Series 44 can accept only 64. The main reasons for the Series 64's greatly improved performance are two: faster operation and parallel operation (doing more than one thing at a time.) Faster operation comes from the use of emitter coupled logic (ECL), one of the fastest commercially available integrated circuit logic families, and from some advanced memory techniques. Parallel operation is made possible by a pair of arithmetic logic units, or ALUs, that share the calculating and decision making that are the basic functions of a computer. With its combination of high speed and parallel operation, the Series 64 can execute well over a million instructions per second (MIPS, in computer jargon), a very creditable number and a real bargain at the Series 64's price, although far from world-championship performance. (There are several computers today in the 10-15 MIPS class and a few that approach 50 MIPS.)

Since it is an HP 3000, the Series 64 can run programs written for other HP 3000s. Because it is a highly complex machine, its designers went to great lengths to make it reliable and easy to service, and it qualifies for HP's money-back guarantee that it will be operational at least 99% of the time. Pictured on this month's cover is the Series 64 system processing unit superimposed on a photograph of a printed circuit board loaded with ECL circuits. The board carries part of the ALU section of the computer.

-R. P. Dolan

Editor, Richard P. Dolan • Associate Editor, Kenneth A. Shaw • Art Director, Photographer, Arvid A. Danielson • Illustrator, Nancy S. Vanderbloom
Administrative Services, Typography, Anne S. LoPresti, Susan E. Wright • European Production Supervisor, Henk Van Lammeren

High-Performance Computing with Dual ALU Architecture and ECL Logic

This largest and fastest HP 3000 Computer System can handle all of the data processing needs of many companies.

by Frederic C. Amerson, Mark S. Linsky, and Elio A. Toschi

HEWLETT-PACKARD'S HP 3000 COMPUTER System family offers users a choice of compatible interactive business systems of various sizes, prices, and performance levels, all using HP's MPE (Multiprogramming Executive) operating system. The new highest-performance member of this family is the HP 3000 Series 64, Fig. 1. The Series 64 provides over 2.5 times the processing power of the Series 44, HP's previous performance leader. This new high-end machine now extends the performance of the HP 3000 Computer family into the one million instructions/second class. Although basically a 16-bit-word machine like other HP 3000s, it is capable of emulating a 32-bit-word machine in many applications.

The Series 64 is expected to be used in all types of EDP (electronic data processing) and distributed data processing applications. As a stand-alone computer system, it can perform a wide range of tasks for a division of a large company and can handle all the EDP needs of a medium-size company. In a distributed processing environment, the Series 64 can serve either as a major node or as the central computer in distributed networks.

The increased system performance of the Series 64 comes primarily from faster operation and performing more calculations in parallel. The methods for achieving these results are the use of dual ALUs (arithmetic logic units), WCS (writable control store), cache memory, 32-bit memory addresses and ECL (emitter coupled logic) technology. Fig. 2 is a block diagram of the Series 64 CPU (central processing unit).

A dual ALU design is effective only if parallel operations can replace operations that were sequential and therefore took longer to complete. The rich instruction set of the HP 3000 lends itself to more parallelism than simpler instruction sets. The traditional method of achieving parallelism is pipelining. The design of the Series 64 CPU incorporates this traditional approach into the dual ALU design to achieve nearly twice the efficiency of a single ALU. Pipelining is a design organization that moves data to be processed through a sequence of hardware operations. A piece of data enters this pipe each clock cycle and proceeds through each stage or operation on successive clock cycles until it is completely processed.

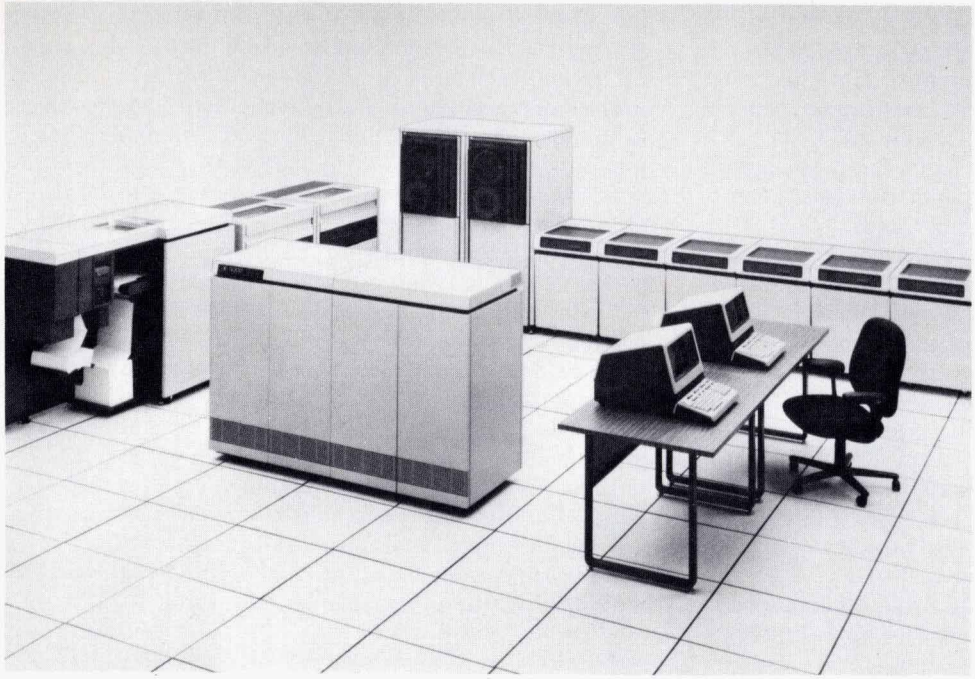


Fig. 1. *The HP 3000 Series 64 Computer System can handle all the data processing needs of a medium-size company or a division of a large company. In networks it can serve as a major node or as the central computer. Its performance is in the one million instructions/second class.*

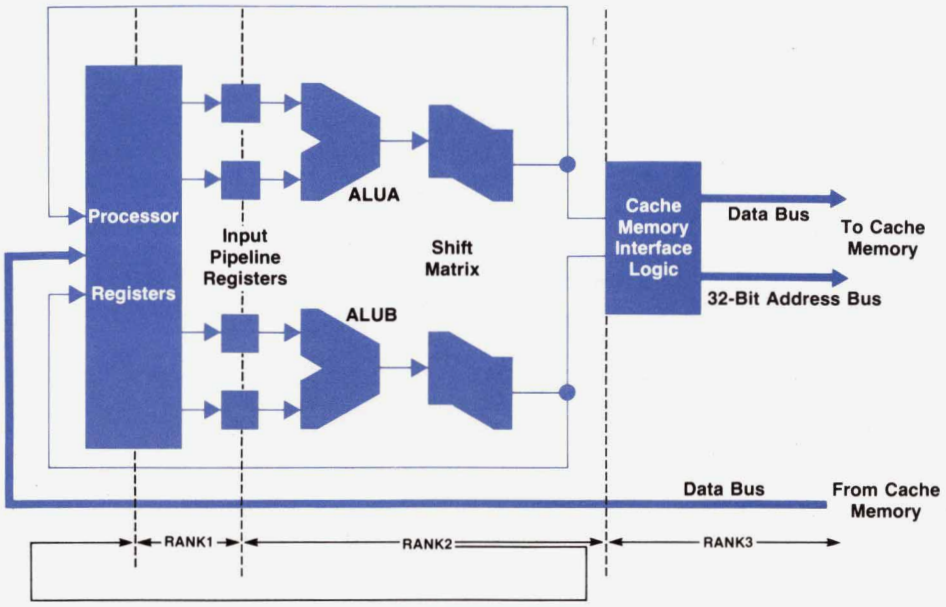


Fig. 2. *The HP 3000 Series 64 central processing unit achieves high performance by means of dual arithmetic logic units, a cache memory, a three-rank pipelined data path, and high-speed emitter coupled logic.*

Writable control store provides faster operation than read-only memory (ROM) for storing and reading the microcode that implements the instruction sets of most modern computers. The static RAMs (random-access memories) used to design the WCS are much faster than ROMs of similar and compatible technologies.

For increased memory bandwidth, both cache memory and a wide memory bus are used. A cache, or small buffer memory, provides high-speed local memory for the CPU while interfacing to a larger but slower main store. The goal is to design the cache such that, most of the time, requests from the CPU for data or instructions will not require the cache to go to main memory to get the particular address needed. This depends very much on the software operating on the system and on the hardware organization of the cache. For the Series 64 running its expected job mix, the CPU cache, with 8K bytes of data storage, provides the CPU with about 95% of its requests in one clock cycle. The rest of the time accesses take longer because the information must come from main memory. However, the average memory access time is still less than two clock cycles.

The interface to the memory system for the cache and I/O system is a 32-bit-wide bus that operates at transfer rates as high as 18M bytes/second (limited by memory speed). The 16-bit I/O buses of the Series 64 are interfaced to this high-speed central bus by input/output adapters, or IOAs, each of which has a 64-byte buffer. These buffers allow these I/O ports to match the slower 16-bit I/O buses to the higher-speed, 32-bit central system bus.

One of the highest-performance logic families in wide use is emitter coupled logic (ECL). In the Series 64, both 10k and 100k ECL techniques were selected for those design areas where the speed and functions available are well matched to the requirements of the design. STTL (Schottky transistor-transistor logic), another fast logic family, is used for the memory array and I/O interfaces. For STTL, 10k ECL, and 100k ECL, typical delays are 3, 2, and 1 nanoseconds per gate, respectively, and each technology offers functions not available in the others.

Design Objectives

Although the main objective of the Series 64 was to extend HP 3000 price/performance to a new high level, there were several other important design objectives. These included software compatibility with existing MPE-based systems, I/O system compatibility with HP 3000 HP-IB* peripherals, and availability of guaranteed uptime service (GUS).

The Series 64 is software compatible with previous HP 3000 family members. MPE, the Multiprogramming Executive, is the operating system for the HP 3000 product line. Object code compatibility is preserved such that any application program written on other HP 3000s in any of six languages—COBOL, FORTRAN, Pascal, RPG, BASIC, and SPL—will run on the Series 64.

The I/O system is the same as that found in the Series 44 and other HP 3000s using the HP-IB. As in these other systems, the HP-IB communicates with the CPU and memory through a channel controller board, which is installed on the intermodule bus (IMB). In the Series 64, however, the memory and CPU are not on the IMB. Communication is accomplished through an I/O adapter (IOA) which interfaces the IMB to the central system bus (CSB). Multiple IOAs can be supported on the CSB (currently a maximum of two), significantly increasing the I/O bandwidth. This I/O system design provides customers with an upgrade path for their HP-IB peripherals, HP-IB device controllers, and channel controllers.

For GUS to be available on the Series 64, high reliability and supportability were important design goals. Guaranteed uptime service is Hewlett-Packard's money-back guarantee that the CPU, cache, main memory, and I/O system including one or two system discs are operational at least 99% of the time. To be able to offer GUS, it is imperative that the system have a high level of reliability, or a high mean time between failures (MTBF), and a high level of supportability, or a low mean time to repair (MTTR).

(continued on page 7)

*HP-IB is Hewlett-Packard's implementation of IEEE Standard 488-1978.

Dual ALU Micromachine Has Powerful Development Tools

by Richard D. Murillo

The central processing unit (CPU) of the HP 3000 Series 64 is microprogrammed to interpret the dual ALU architecture of this machine. A single line of microcode controls the execution of two parallel processing units. Each unit controls its own execution on subsequent clocks and may specify the next line of microcode to be executed. The two processing units are referred to as ALUA and ALUB. Each processor consists of an arithmetic logic unit (ALU) and shifter, two source operand data paths, a target data path, logic to perform special functions such as reading and writing memory, and logic to control the sequence of microcode execution.

The Series 64 microcode processor uses a three-stage pipeline to execute microcode. The three stages are referred to as RANK1, RANK2, and RANK3. During RANK1, source operands for the two ALUs are clocked into the registers that serve as input to the ALUs. During RANK2, the arithmetic operations are performed and the results are stored into the destination register. Special operations and skip tests are also performed during RANK2. Memory reference operations initiated during RANK2 are performed during RANK3.

Three speeds of jumps are built into the Series 64 processor. A fast jump (unconditional jump) is taken from RANK1. The target of a fast jump is the next line of microcode to enter the pipeline after the jump line. Medium-speed jumps are dependent on information known at the beginning of RANK2, and are taken during RANK2. In this case, the next sequential line enters the pipeline before the target of the jump. The execution of this line is inhibited when it enters RANK2. Slow jumps depend on the output information of the ALUs during RANK3. In this case, the next two sequential lines of microcode entering the pipeline are inhibited during RANK2. Inhibition of the RANK2 execution occurs when the NOP flip-flop is set for the respective ALU. Accordingly, the term NOPed means inhibition of RANK2 execution. This inhibition has an effect only on the storing of information from the ALU and the special/skip functions. The ALU operations are still performed and the output of the ALU may be used on subsequent lines.

This pipelining of the microcode processor contributes significantly to the performance of the Series 64 CPU. The microprogrammer, however, must be aware of the effects of the pipeline on the execution of microcode. For example, new data stored into some registers cannot be accessed on the following line of microcode. The store into the register on the first line takes place in RANK2, and the read on the second line takes place in RANK1, both during the same clock period. Therefore, the new data is not clocked into the register until after it is read.

The microcode format field is divided into fourteen fields across two ALUs on a single line of microcode. These fields include two input sources, operation functions and store target, special CPU functions, and microcode skip conditions. A subfunction field may be used to specify shifting of the ALUs or a target of a microcode jump. One of the input source fields on either ALU can be used to specify a short or long hexadecimal literal. Because the microcode is Huffman-encoded, use of the subfunction or literal options will disallow the use of some fields during the assembly of the microcode line. For example, specifying a long (16-bit) literal will disallow the use of special and skip fields and one of the input source fields.

Development System

The HP 3000 Series 64 microcode development system (MDS) is an interactive software tool used for the overall design and development of system and diagnostic microcode. The development system includes a microcode assembler and editor, a system (hardware level) simulator, and a set of symbolic debugging tools.

The microcode development system runs as an interactive program under the HP 3000 Multiprogramming Executive (MPE) operating system. It has a complete set of commands for entering and saving microcode source files, editing microcode fields and source lines, assembling microcode, and running a system simulation (see Fig. 1). It also has a set of debugging tools and commands. These commands consist of one or more letters and can be uppercase or lowercase. Multiple commands on one line may be entered by separating them with semicolons. Many commands have parameters that may be entered as general expressions. The system also provides a method of terminating command execution.

The microcode development system provides a wide range of commands for working with microcode source files. These files are compatible with HP 3000 EDITOR source files. The system provides the capability of entering microcode source files into a work file, keeping the text in a permanent MPE file, and renumbering the microcode text. Editing features of the system include adding and deleting of text, modifying, listing, and insertion of microcode lines, and the addition of comments into the text.

Using the BATCH command allows the microprogrammer to assemble source microcode. This assembly process takes a microcode source file as input and produces a microcode format-

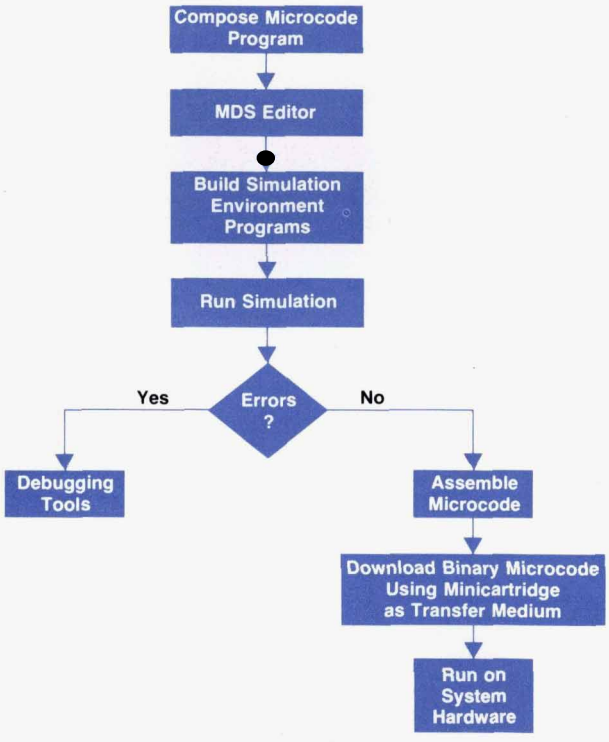


Fig. 1. *Development sequence using MDS, the HP 3000 Series 64 microcode development system.*

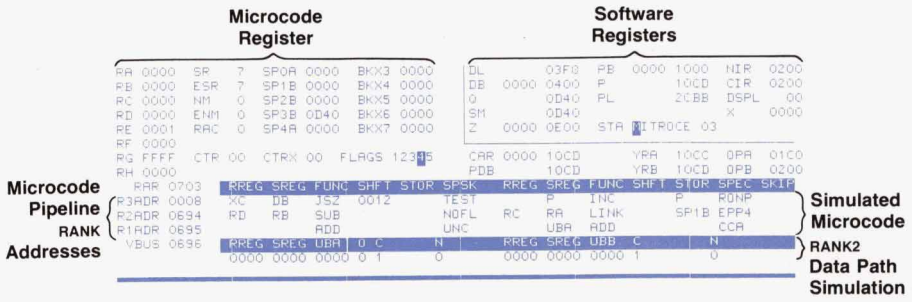


Fig. 2. *The programmer can step through a microcode program one line at a time and observe the results on a CRT terminal. The screen is divided into two main areas, register display and simulated microcode display. Register contents are shown in hexadecimal notation and inverse video is used to show different flag states.*

ted binary file. The microcode assembler also produces a listing of the microcode text, a cross-reference map (labels mapped to addresses), a listing of writable control store (WCS), which is the layout of the microcode memory area in binary, and a listing of the lookup tables (LUT), which is a layout of the system macroinstruction-to-microcode-instruction mapping. The assembler uses imbedded commands in the source file as options to control these listings. There are also options to print top-of-page headings and the setting of the control store address.

The LUT command allows the microprogrammer to set up the lookup table entry for a particular system macroinstruction. It allows the programmer to specify the binary format of the macroinstruction, the preadjustment value of the top-of-stack registers, the displacement value, and indexing and indirect options for memory reference macroinstructions.

Simulation Package

The microcode development system has a simulation package that allows the microprogrammer to simulate the execution of a microprogram so that it can be checked out before the program is placed into the system control store. Program execution can be simulated a single line at a time or as a free-run execution.

Execution of a microprogram is accomplished by entering the microcode into a development system work file and using the EXECUTE command. The execution is divided into three types: single-line (wait) manual control execution, single-line (pause) automatic control execution, and free-run execution of the microprogram.

Single-step execution of a microprogram allows the microprogrammer to step through each line of microcode through the CPU pipeline. The programmer can examine each step on the display screen of a CRT terminal (Fig. 2). The screen gives the programmer a visual picture of the CPU hardware buses and registers after each line of microcode is executed. The programmer can also examine the effects of each microcode line as it passes through the ranks in the pipeline. In single-step mode, the screen is updated after each microcode line is executed either manually (programmer hits the carriage return key on the terminal) or automatically (system updates after one second).

The microprogrammer can use the EXECUTE command to specify the control store address where microcode execution will start, and can specify the number of simulated clock cycles to excute before terminating the simulation. The programmer can also execute the next overhead (macroinstruction fetch/decode) microcode line for software simulation and print a trace of the simulation to a hard-copy device.

The simulator contains a memory image area for loading and executing software programs and data. This environment area allows the microprogrammer to simulate the execution of diagnostic programs or special instruction test routines. Using the STORE/RESTORE commands, a microprogrammer can store an environment as an MPE file or restore an environment into the simulator. This debugging tool allows the microprogrammer to correct problems in microcode and restore test routines without having to rebuild the environment.

Debugging Tools

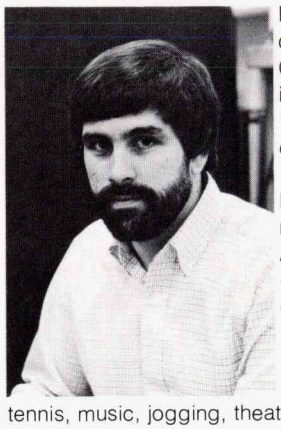
The microprogrammer is also supplied with a complete set of commands for use in debugging the microcode. In addition to single-step execution and environment test routines, the programmer can set and clear microcode breakpoints in control store, display and modify memory locations at specified addresses, and evaluate arithmetic expressions in three different bases: octal, decimal, and hexadecimal. The microprogrammer can set up to 32 different breakpoint addresses for tracing the path of the microcode and can indicate a count for the number of times the breakpoint is executed before the break is taken. Memory can be displayed at the terminal or printed out on a hard-copy device. The system also contains an expression evaluator which allows the microprogrammer to examine and/or evaluate registers, constants, and symbolic microcode labels values in an expression.

The microcode development system was used in the overall development of the Series 64 CPU design and for development and coding of the HP 3000 machine instruction set, system microcode diagnostics, and channel program microcode. This powerful development tool was instrumental in the discovering of hardware/firmware design problems in the early stages of the project and the debugging of the system microcode while the hardware was being built. The system was also used for simulating microcode test programs, for CPU software diagnostic routines, and for determining hardware design faults. It is hoped that the microcode development system can be enhanced or modified for future system designs and machine architectures.

Acknowledgments

The author wishes to thank Cherie Kushner for major contributions to the overall design of the microcode simulator and Mike Mason and Kevin McGrath for their technical assistance.

Richard D. Murillo



Rick Murillo received BS and MS degrees in computer science from California State University at Chico in 1974 and 1977. After joining HP in 1977, he developed system microcode for two years, and since 1979 has been project manager for HP 3000 Series 64 diagnostics and microcode. He's a member of the ACM and the IEEE Computer Society and teaches business data processing at a local community college. Born in Oakland, California, he now lives in Los Gatos, California and is interested in photography, tennis, music, jogging, theater, and old-time movies.

Design for Reliability

Increased reliability is the result of an improved interconnect scheme, a highly effective cooling system, system design to protect against electromagnetic susceptibility, and careful component selection and qualification. Careful consideration of each of these areas was necessary to provide an optimal system design.

The large number of interconnections in the CPU cardcage demanded a dense, reliable, and cost-effective connector scheme. With a maximum of 8M bytes of memory in the system, there are over 8000 high-speed backplane connections and over 1000 for the frontplane. For added reliability, two-piece connectors were chosen over printed circuit board edge connectors for the backplane and frontplane, as well as for all frontplane connectors for cables. This choice also means that gold-plated printed circuit boards are not required, which results in a considerable cost savings.

The cooling design is very critical to the reliability of the system. The results are impressive. Even though the power dissipated in the CPU cardcage is three to eight times that found in other HP 3000 Computer Systems, the temperature rise for a system at room temperature is on the average, 50% lower than for these same systems. This reduction in temperature rise inside the system leads to a much lower junction temperature for the components, which of course means higher reliability.

System reliability with respect to electromagnetic susceptibility requires that the system be immune to certain levels of electrostatic discharge (ESD), radiated and conducted electrical waves, power line transient noise, and magnetic waves. The mainframe and power system design must be done at the system level to guarantee that the system is insensitive to these types of externally generated interference. Grounding and shielding of the Series 64 processing unit minimizes the effects of static discharge. Immunity to power line transients is another important design objective, and the design of the Series 64 ac power system makes it resistant to injected transients of up to 1000V.

The selection and qualification of quality components plays an integral part in the design of a reliable system. All semiconductor devices used in the Series 64 are covered by Hewlett-Packard's general semiconductor specification, which defines those standards and requirements that must be met by suppliers and devices to meet minimum quality and reliability levels. Since ECL and 64K dynamic RAMs were critical to the success of the Series 64, extensive testing was done to evaluate these components thoroughly.

Three main ECL families were tested: 10k ECL, 100k ECL, and 10k ECL RAMs. These groups were further subdivided into the processes used to fabricate the devices. A representative part was chosen based on complexity and use from each family and process for qualification. The qualification tests consisted of dynamic operating life at elevated temperatures with parametic measurements recorded at intermediate points from start to 1000 hours. This type of testing is concerned with stability and allows for trend analysis. Hewlett-Packard was then able to communicate to the vendors any problems encountered with enough data so that proper action could be taken.

HP has developed much experience as an aggressive user of semiconductor memories. Through the cooperation of the using divisions, a corporate specification was developed. It was also possible to establish efficient and effective evaluation and qualification procedures for 64K RAMs. Aggressive soft error rates were specified and numerous RAMs tested to determine the rates at the system level and for accelerated tests. A key goal is a rate of <1000 fits (failures per 10^9 hr) in system operation. The results were the early establishment of qualified suppliers and devices for Hewlett-Packard and the Series 64.

Assuring Supportability

The other aspect of system availability is supportability: the length of time it takes to isolate a failure and fix it. Serviceability in the Series 64 is enhanced by a sophisticated diagnostic control processor and comprehensive fault-locating diagnostics. The diagnostic control unit or DCU is a separate processor that has the capability to access CPU registers, monitor line voltages and system temperatures, conduct self-tests, and log errors on the system. With extensive microdiagnostics, it can isolate hardware failures down to the functional board level. Since microcode is stored in RAM and not in ROM, there is no real limitation on the amount of storage available for microdiagnostics. It was therefore possible to develop tests that perform a hierarchical diagnosis of the hardware, allowing quicker and more accurate isolation of faults. As in other HP 3000 Computer Systems, remote diagnosis is also provided.

Dual ALU Design

Other HP 3000s have used more than one arithmetic logic unit (ALU) to perform the arithmetic functions necessary for the execution of instructions. However, these additional ALUs have been very special-purpose, allowing little flexibility. In particular, one ALU always added together the index register and the displacement field of the instruction. When the instruction was not indexed, it added zero to the displacement. This logic then provided the sum to the microprogrammer whenever it was needed. Usually, the sum was needed only once during an instruction, so most of the time this ALU was performing needless work. In the approach taken by the Series 64, this second ALU is general-purpose, so that it is useful throughout the instruction. Thus the microprogrammer is allowed direct control of the function of this ALU, thereby greatly increasing its efficiency.

Another approach to improving performance is adding processors to a system. Unfortunately, the second processor of a multiprocessor system usually requires considerable sophisticated software to achieve the hoped-for performance. Many complicated interactions between processors can occur that are not problems in a uniprocessor system. Also, some functions do not behave as they do in a uniprocessor. For example, what does the second processor do when interrupts are disabled by the first processor? This must be dealt with in the software design. For another example, the function of disabling process switching no longer prohibits certain actions, as it does in a uniprocessor system, since there still may be more than one process running. Thus other more expensive mechanisms must be devised to control the interaction between software mod-

ules which might be running on different processors. In general, this results in the second and successive processors of a multiprocessor system being much less than 100% effective.

By putting the second processor under microcode (firmware) control rather than software control, performance is achieved without complicated multiprocessor software. System performance benefits as if there were more than one processor, which is true, but the software does not see the added complexity; it is hidden in the microcode. The system benefits from the additional hardware without suffering the performance degradation of multiprocessor software.

Parallelism in microcode means that each instruction runs faster than in a system with parallelism in software. Thus instruction execution time is reduced. A stream of single instructions benefits from the additional hardware as much as multiple timeshare jobs. In a multiprocessor system, this performance benefit is not realized until there is enough work to be done in parallel to keep all of the processors busy. The richness of the HP 3000 instruction set provides the basis for a contribution in parallelism. In a simpler instruction set, there may not be enough work to keep two parallel ALUs busy. The general bounds checking of the HP 3000 and powerful instructions like procedure calls benefit from the power of two ALUs. In fact, because of some other minor improvements, a procedure call on the Series 64 takes fewer than half the cycles of previous HP 3000 systems.

A wide microcode word makes options available to the microprogrammer that were not available on earlier systems. In the past, only certain registers could be used to store addresses that were also sent to the memory system. That has been expanded to allow any register to be used to save the address being sent to memory.

One of the interesting anomalies of ECL logic is the relationship between read-only memory parts (ROMs) and random-access memories (RAMs). In past HP 3000 systems, the microprogram has been stored in ROM because this was more cost-effective. However, for ECL, it is more cost-effective to use RAM, or writable control store (WCS). The flexibility obtained makes it easy to update the microcode when bugs are found or performance enhancements made. The capability of loading microdiagnostics and executing them at the customer site in the WCS gives the Series 64 the best fault-locating diagnostics available on any of Hewlett-Packard's systems today.

The design approach is dubbed dual ALU even though much more than the ALUs are paired. When the decision was made to make the second ALU general-purpose, it was necessary to provide support for it so that it would be as useful as the first. Original estimates for the efficiency of this second ALU were in the vicinity of seventy percent, but with the right support, the actual efficiency has been well over eighty-five percent.

ALU Capabilities

Each of the two ALUs has access to a subset of the registers available in the processor. Some are available as one source to each of the ALUs while others may be a source operand to either input of both ALUs. Most registers can be altered by one or the other ALU, but not both. The exception to this is the top-of-stack cache registers (TOS) which are available to both ALUs as either operand and may be altered by both ALUs. This makes these registers extremely valuable as scratchpad registers because they are useful for passing data between the ALUs. Unfortunately their generality also makes them expensive to implement, so there are only eight of them. The output of either ALU may be passed to its own input or that of the other ALU; this is also a frequent method of exchanging data between the ALUs. Scratchpads and software environment registers may be altered by one of the ALUs and read by one or both of them. In the case of the environment registers, this is particularly useful since it allows both the upper and lower limits of an address to be checked in a single microcycle. Each ALU has a bank of 512 registers that are available only to it. By pairing these registers it is possible to read, store, and operate on thirty-two bit data, even though the ALUs themselves are only sixteen-bit units.

Either ALU may specify any of a variety of special options. These perform such useful functions as setting condition codes on data, checking for operands that may be in the TOS registers, setting and clearing flags, and controlling the loop counter. Memory reads and writes are also initiated in these fields.

Each ALU may independently skip the execution of its half of the following line of microcode by use of its skip field. This very powerful technique keeps both of the ALUs busy executing useful code as much of the time as possible, since it is necessary to skip the effect of only half the line of microcode rather than the entire line. Also, either ALU may specify transfer of control to another point in the microcode by a conditional or unconditional jump instruction.

Tight control of the two ALUs is maintained by keeping them in lock step with a single microaddress register. When either ALU specifies a jump, both ALUs must jump to the new location. Thus, synchronization problems that would occur if each could be executing independently are eliminated. Although the control of the microsequencer is more complex than it might be for a single ALU, it is far less complex than the mechanism that would be required to keep two independent streams of instructions from causing total chaos. It is impossible to get the ALUs out of synchronization since they are both controlled by the same microcode word. The control store is addressed by a single register. It may be modified by either ALU, but there is only one register. By looking at only one line of microcode it is possible to determine precisely what each ALU will do.

Since it is possible for both ALUs to specify a jump on the same line of microcode, a priority mechanism is necessary to resolve conflicts. One approach would require that the microcode never specify two jumps that could both be taken on a single line. This would give a straightforward method of determining the next line of microcode, but would restrict the programmer and limit flexibility. A more powerful approach assigns a priority mechanism if both ALUs specify a jump at the same time. If neither of the jump conditions is met, execution continues in sequence. If only one of the conditions is met, execution transfers to the location specified by that ALU. But, if both conditions are met, then one of the ALUs has priority over the other. This

structure makes it possible to perform a multiway branch in a single line of microcode. By coupling with the skip conditions of the preceding line, a three-way branch can be performed on four different conditions. An insightful coder can use this for tremendous leverage.

Simultaneous access to the cache memory is perceived by the ALUs although the cache interface has only a single port. It is possible for either or both ALUs to specify a memory read or write on any line of microcode. The cache memory, however, will accept only a single request from the processor in any cycle. To reconcile the CPU to the cache, a strict priority mechanism sorts out all accesses and sends them to the cache in priority order. Thus, if both ALUs request a memory access at the same time, the lower-priority one will have its request queued until the higher-priority one completes. In general, no more than one request to memory is needed for each cycle in the processor, but it is not always convenient to specify the requests on separate cycles. The priority scheme, like that of the jumps, allows greater flexibility to provide increased performance.

Accesses to the cache that cannot be satisfied immediately are deferred in hardware so that until the data is actually needed the microprocessor does not stop and wait. There is no speed advantage to allowing both ALUs to specify a memory access on the same line if processing halts until both requests can be sent to the cache. It is therefore necessary to buffer the requests in a holding register and allow the processor to continue execution. Only when more requests are made than can be buffered must processing be suspended to wait for completion. Up to three requests can be in process without overflowing the buffer.

The two sixteen-bit ALUs can be linked together to perform 32-bit calculations. In cases where it is useful to perform more than sixteen-bit calculations, a special option in the microcode, called LINK, ties both of the ALUs together to perform 32-bit arithmetic. There are many instructions in the HP 3000 architecture that benefit from this capability. It is particularly useful in some of the multiply and divide instructions which are simplified by not having to piece together so many sixteen-bit partial results.

ECL Logic Design Considerations

As mentioned earlier, the obvious reason for using emitter coupled logic (ECL) is its high speed, but other advantages come from ECL's low-voltage logic swings. These advantages are low noise induced into power supplies, low radiated noise, and low crosstalk. However, these small logic voltage swings also result in low noise margins. Noise margin is reduced by temperature differentials between drivers and receivers, voltage differences caused by power distribution, voltage noise on power buses and power planes, and signal voltage ringing because of impedance mismatch.

A temperature differential between driver and receiver causes loss of noise margin if the receiver does not track the temperature of the driver. The Series 64 limits the temperature differential across a printed circuit board and from board to board by using fast-moving air from a common plenum chamber to cool the printed circuit boards. For some signals, limiting temperature was not quite enough; for these signals, differential drivers and receivers are used with a common reference voltage for each driver-receiver pair.

ECL operates with a potential difference of 5.2 volts for 10k ECL and 4.5 volts for 100k ECL. The most positive voltage is called V_{CC}. Any voltage difference between V_{CC}s in the system can subtract directly from the noise margins. The Series 64 makes V_{CC} the common rail or signal ground. The CPU printed circuit boards, the frontplane, and the backplane use two thick copper planes close to the signal plane to distribute ground. Ground connections are distributed along the board connectors to the frontplane and the backplane, reducing the effects of voltage differences between various parts of the system. V_{EE}, the most negative potential across the ECL ICs, is at -5.2 volts. V_{EE} is very heavily bypassed to ground on each board by placing a capacitor next to every other IC on the board. V_{EE} is distributed to the CPU by two thick copper planes on the backplane and one thick copper plane on each printed circuit board. Changes in V_{EE} reduce noise margins by twenty-five percent of the change in V_{EE}. V_{TT} (termination potential, the return for the termination resistors) is distributed to the CPU like V_{EE}. It has much less effect on noise margins than V_{EE}.

The Series 64 CPU uses transmission line techniques with closely controlled characteristic impedance on all signal nets to maintain signal integrity and speed. In TTL designs, it is possible for the signal to propagate up and down the signal path three times before the signal is stable. Signal paths with very long stubs (a branch off the main path) will cause reflections, thus making propagation delay longer than necessary. At higher bit rates and faster edge speeds these reflections may combine to cause loss of noise immunity. TTL wiring techniques use an input diode clamp built into the IC to reduce the amplitude of the undershoot or ringing. ECL techniques approach the problem by matching the characteristic impedance, thus controlling reflections. The Series 64 uses microstrip lines for controlled characteristic impedance. A microstrip line is a conductor strip or trace separated from a ground plane of conductive material by a dielectric. The characteristic impedance (Z_0) can be controlled by arriving at a balance between the trace geometries, board materials, board thickness, and power consumption. To reduce signal ringing and maintain signal integrity, the termination of the line must match the Z_0 of the printed circuit board. ECL is

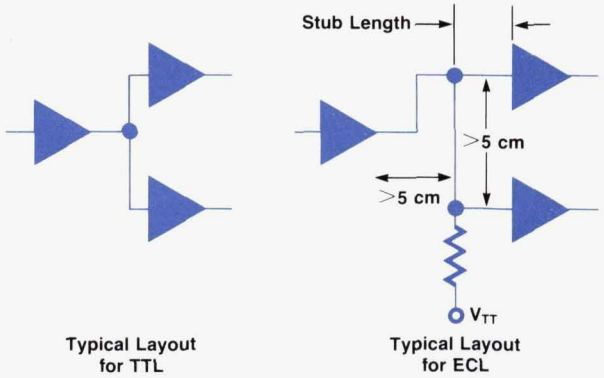


Fig. 3. *Comparing ECL and TTL design techniques.*

designed to drive lines of 50 ohms or greater. The Series 64 uses 68 ohms as the nominal value of termination resistors except for special cases. Even in these cases, controlled impedance rules are followed.

A thorough understanding of signal propagation is very important in ECL board design. Trace layout must be done in an orderly fashion. The designer must have complete control of all traces so that stubs do not occur. A good reporting scheme for the actual board layout is necessary, so that timing can be accurately simulated for analysis.

Control of Clock Skew

A fast cycle time requires tight control of clock skew and system synchronization. The Series 64 has a cycle time of 75 ns. The clock skew was held to 5 ns worst-case. This is not very significant in itself except that no special manufacturing adjustments or considerations are necessary. This was assured by using equal traces from the point where clocks are generated to the point where clocks are received, by using low-propagation-delay 10k ECL and MECL III parts, and by distributing clocks to each board in the system by complementary pairs. The clock distributed to each board has a period of 37.5 ns. Each board receives a sync signal from the DCU (diagnostic control unit) that locks in-phase a divide-by-two circuit on each board. The absence of the sync signal stops clock generation on that board. The 75-ns clock derived on each board is then distributed to all ICs on the board that require a clock. Each clock has no more than four loads. A bonus of distributing the clock in this manner is that we are able to generate four phases of 18.75 ns for minor cycles in the system. These are used mostly in the memory and I/O systems.

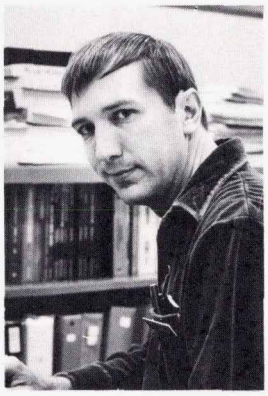
There are certain parts of the CPU that require close clock-to-clock tolerance. In these cases clock pairs are chosen such that the clock signals requiring tight tolerance are driven from the same IC. Distributing differentially driven clock signals removes skew caused by temperature and voltage changes between boards.

Partitioning and Propagation Delay

Efficient use of each clock means performing as many logical functions in each cycle as practical, and in the ideal case having the same propagation delay for all paths in the system. No path should have to wait for any other path. This is where propagation delay becomes an important factor in influencing partitioning of the circuitry, even more than logical functions. ECL works best when signal paths flow from one point to one or more receivers, as shown in Fig. 3. Stubs must be short and lines should be distributed such that the distance between loads is greater than 5 cm. This allows the line to approximate a distributed capacitive loading rather than a lumped capacitance which would cause reflections. The Series 64 CPU accomplishes this by bit-slicing the data path of the system into four bits for ALUA and four bits for ALUB on each of four printed circuit boards, called RALU boards. This enables each individual bit line to be completely contained on one printed circuit board. It is much easier to adhere to ECL rules if each line is contained on one board. The four RALUs contain the data paths for RANK1 and RANK2 (see Fig. 2). The data signals used for carrys and shifts are routed across the frontplane to other RALUs. The frontplane is an added signal plane that ties six of the eleven CPU boards together.

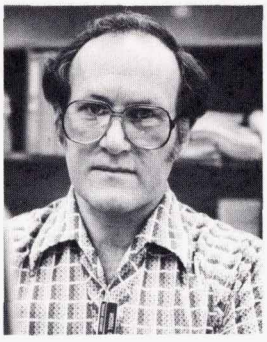
Higher Data Path Performance

Higher performance and increased functionality are obtained by using high-speed 100k logic in the main data path of the CPU. The use of 100k logic decreases the propagation delay through the IC to one-half of what it would be for 10k ECL. Additional care had to be taken in board layout because of the increased rise times and reduced voltage swings experienced with 100k ECL. The RANK1 and RANK2



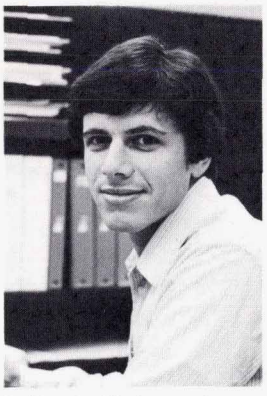
Frederic C. Amerson

Rick Amerson received his BEE degree from Georgia Institute of Technology and joined HP in 1972. He designed a plotter interface for the HP 3000, did CPU design for the HP 3000 Series II and Series 64, and was project manager for portions of the Series 64 design. He's now a project manager with HP's Computer Systems Division. A resident of Santa Clara, California, Rich is treasurer of his church, an instrument-rated private pilot, and a member of the International Brotherhood of Magicians. He enjoys cooking and downhill skiing.



Elio A. Toschi

Elio Toschi has been designing computer hardware for HP since 1965. He has served as a design engineer and as project leader for various I/O, communications, and memory systems including those of the HP 3000 Series 64. Before coming to HP, he designed avionics equipment for eight years. Elio was born in San Jose, California and received his BSEE degree in 1957 from San Jose State University. He's married, has two children, loves skiing of any variety, and still lives in San Jose.



Mark S. Linsky

A native of Gloucester, Massachusetts, Mark Linsky received his BSEE and MSEE degrees from Massachusetts Institute of Technology in 1972 and 1973. He joined HP in 1973, designed a handheld-calculator power system, contributed to the design of the HP 3000 Series 64 memory system, and served as project manager for the Series 64 memory and I/O systems. In 1976 he received his MBA degree from the University of Santa Clara. One patent—on a battery recharger—has resulted from his work. Mark is married, has a daughter, and lives in Saratoga, California. He keeps busy playing softball and racquet sports and working on his home in Saratoga and his vacation home near Lake Tahoe.

data path is implemented in 100k logic, except for the store. The store path, UBUS, uses 10k logic. Since 10k logic should not be driven by 100k logic because of their different threshold voltages, the CPU uses quad line drivers to interface between 100k and 10k logic.

The increased speed and functionality of 100k ECL is used by the main data path of the ALU, the R and S operand registers, the R and S multiplexers and the carry lookahead circuit. The major advantages in functionality are in the multiplexers and the carry lookahead ICs, where a two-for-one reduction in parts count is achieved. The ALU ICs perform eight logic operations and eight arithmetic operations. In addition to performing binary arithmetic, the circuit contains the necessary correction logic to perform BCD addition and subtraction. The ability to do decimal arithmetic is not available in standard 10k ECL without extensive correction circuitry, which increases critical path delay.

Acknowledgments

The success of the Series 64 was the result of the hard work of many people. We especially wish to thank Bob Horst, Bill Bryg, and Bill Putzier for their contributions to the design of the CPU, Dave Cantrell for his efforts in building and debugging CPU hardware, Ed Miller for his design of the power system and EMC compatibility, Jim Brannan and Andre Shaari for their help in designing quality into the system, and Peter Rosenbladt for his leadership in getting the product to market.

Powerful Diagnostic Philosophy Reduces Downtime

by David J. Ashkenas and Richard F. DeGabriele

AN IMPORTANT ATTRIBUTE of any computer system is availability. Because we rely on computers so heavily to increase our productivity, it is vital that they function properly all the time. From an availability standpoint, an ideal computer is one that never fails. Since this goal is difficult to attain, the next best computer is one that rarely fails and can be quickly repaired when it does.

A key factor to be considered in designing such a machine is that it is generally difficult to isolate a faulty component but it is usually easy to replace it. One method of solving this problem is to increase the size of the smallest field-replaceable unit. With fewer subassemblies to consider, the process of determining which one to replace becomes easier. If this approach is carried to its logical conclusion, the entire computer can be made one unit. For economic and other practical reasons, however, the smallest field-replaceable unit of the HP 3000 Series 64 is a printed circuit assembly or a power supply. An exception is the main memory RAM integrated circuits, which can be individually replaced.

Because of the tremendous complexity of the Series 64 (there are over 3000 ICs in the CPU alone), conventional methods of fault isolation, such as board swapping, are both time-consuming and expensive. Therefore, an innovative failure diagnosis philosophy was formulated and adhered to throughout the design and development of the computer. The resultant system is easy to use and provides for remote diagnosis and board-level isolation of mainframe failures.

The field diagnostics used in previous HP 3000 Computers do not diagnose; rather, they test and then simply halt if a failure occurs. The customer engineer (CE) must then correlate the particular test that failed with a specific circuit function being tested, determine which board performs that function, and replace it. This requires that the CE be familiar enough with the diagnostic to know precisely which circuit function it was testing. In addition, the CE must be familiar with the machine under test at the detailed block diagram level to know how that circuit function is partitioned over the set of boards in the system. Without this detailed knowledge, troubleshooting is reduced to swapping boards on a best-guess basis until the failing board is isolated. It is expensive to train the CE to know the machine at the detailed block diagram level. However, board swapping is also costly; it takes time and large spare parts inventories must be carried.

To reduce repair time and inventory costs and to increase field productivity, the Series 64 diagnostics have been designed to indicate which board(s) in the system could contain the fault. To help achieve this objective, the diagnostics are written in microcode to exercise maximum isolation and control of the hardware being tested.

The fault-locating diagnostics are packaged so that they obtain failure information while requiring no special training or knowledge to use. In fact, the diagnostics are designed to be run either on-site by the customer or remotely from a field office so that the CE has an understanding of which replacement boards may be needed before traveling to the computer site. This is possible because the diagnostics themselves identify suspected boards in order of fault probability (see Fig. 1). Hence no special training or knowledge of the hardware is necessary to interpret the test results.

All of the functions available at the system console are also available at a remote console via a modem and tele-

```
HP 3000 SERIES 64  FAULT LOCATING DIAGNOSTICS
Diagnostic Menu - Press the corresponding softkey
f1 - KER/MICR -- Run Kernel and all Microdiagnostics.
f2 - ALL MICR -- Run all Microdiagnostics (Section 1-5).
f3 - MEMORY -- Run Microdiagnostics (Section 4-5).
f4 - I/O -- Run Microdiagnostics (Section 5).
f5 - WCS PART I -- Addresses 0100H - 13FFH
f6 - WCS PART II - Addresses 0000H - 12FFH
f7 - IOMAP -- Run Microdiagnostic IOMAP
f8 - RESTART -- Rerun currently loaded Microdiagnostics.
```

```
KERNEL DIAGNOSTIC
REVISION A-2123
TESTING TIME = 4 MIN.
PAGE ONE COMPLETED
PAGE TWO COMPLETED
PAGE THREE COMPLETED
PAGE FOUR COMPLETED
END OF KERNEL DIAGNOSTIC


FAULT LOCATING MICRODIAGNOSTIC REV A-2126
SECTION 0001 LOADING
SECTION 0001 EXECUTING--TESTING TIME 00010 SECONDS
SECTION 0001 COMPLETED, 00192 PASSES
SECTION 0002 LOADING
SECTION 0002 EXECUTING--TESTING TIME 00040 SECONDS
MICRODIAGNOSTIC FAULT--TEST NUMBER 059.3, PASS COUNT 00000
SUSPECTED ASSEMBLIES IN ORDER OF FAULT RANK:
1 CTLB
2 RAL2
3 SKSP
```

Fig. 1. *Fault-locating diagnostics menu and a typical test result showing a successful diagnosis.*

phone line. In addition to normal console capabilities, a CE can load any diagnostic, monitor the ac and dc power systems, and run the fault-location diagnostics without leaving the field office. Hence a customer's computer can be fully diagnosed before any trips to the site have been made.

Diagnostic Control Unit

The most powerful diagnostic tool of the Series 64, the diagnostic control unit or DCU, is completely built into the CPU. The DCU is a Z80 microprocessor-based board that provides the sole interface between the user and the mainframe (see Fig. 2). For the first time on an HP 3000, there are no separate front-panel controls and maintenance panels. All these functions can be executed from the system console, which is connected directly to the DCU via a standard RS-232-C serial data channel. The ROM-based program for the microprocessor configures the DCU and the console to allow maintenance, diagnostic, and system operator functions to be performed while remaining transparent to the user.

The DCU has access to all other printed circuit boards in the CPU/MEM cardcage through the use of serial shift strings. Each board is designed such that many of its state-determining elements (e.g. registers, flags, flip-flops, etc.) are implemented using shift register ICs. These elements are connected to form a large circular shift register known as a shift string. There is one string for each board. The DCU can freeze the machine, shift data out from a selected board for examination and/or modification, and then return the shift string to the board (see Fig. 3).

The DCU also controls the clocks for the Series 64 system. It can clock the entire system or any subset of its boards from 1 to 255 contiguous clocks. The shift string and clock control capabilities of the DCU form the basis of a number of system control and diagnostic functions, including: loading and reading the WCS (writable control store) and memory, system initialization, maintenance panel (i.e., register and flag) displays, and initial microprogram loading (power-up and cold-load sequences).

When not performing any of the above tasks, the DCU operates the power system control (PSC) board, an interface between the DCU and the power system of the Series 64. It is via the PSC that the DCU can monitor the ac line voltage, dc power supply voltages and currents, and other parameters. This information can be displayed on both the system console and a light-emitting diode display on the PSC. Hence the CE can use the DCU to identify potential problems in the power system.

Diagnostic Tests

An equally important component of the Series 64 diagnostic philosophy is the set of diagnostic tests used to verify the proper operation of all mainframe hardware and to

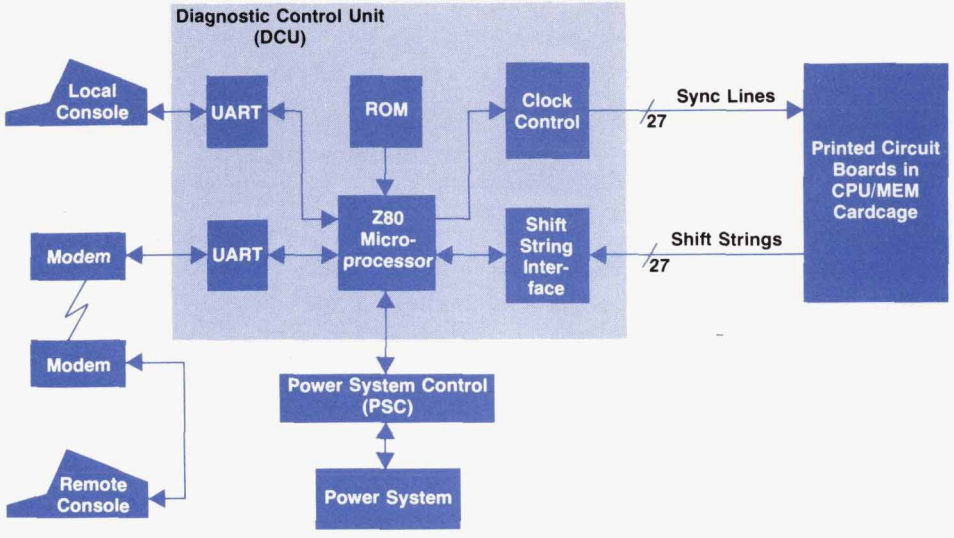


Fig. 2. *Diagnostic control unit block diagram. The DCU provides the sole interface between the user and the mainframe.*

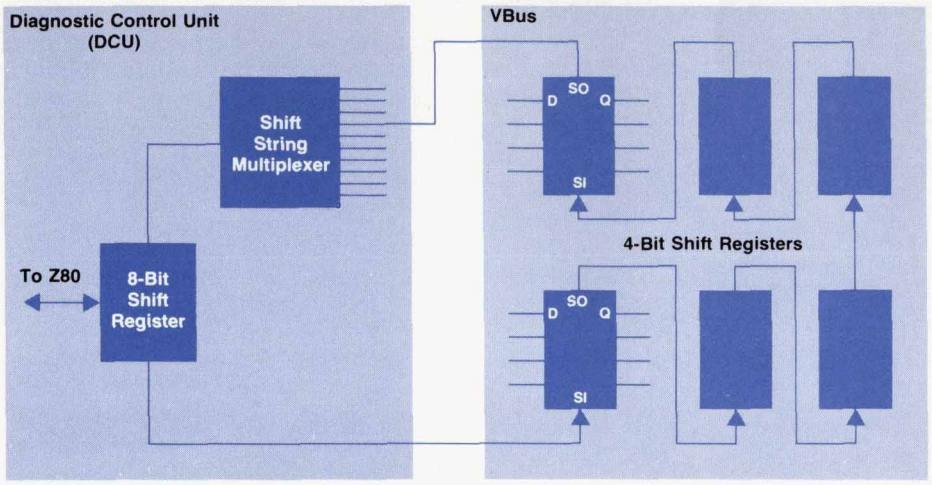


Fig. 3. *Each CPU or memory board is designed such that most of its flip-flops, flags, and registers form a large circular shift register or "shift string." This is a diagram of a shift string and the corresponding console display. The string is broken up into fields determined by the designer.*

detect and isolate faults when they occur. The diagnostics are designed to test the computer in a hierarchical fashion, with three distinct levels of diagnosis (see Fig. 4).

This hierarchy is designed to take maximum advantage of the built-in diagnostic features of the Series 64 to provide board-level isolation of failures. The procedure for testing the computer involves applying the lowest-level tests first, and then working upward through the heirarchy, adding more known good hardware in small increments. Thus when a test fails, the faulty circuitry is implicitly isolated in most cases, since all hardware required to run lower-level tests has already been checked.

Level 1—Kernel Hardware Verification

The first level of testing is designed to check the essential core, or kernel, of hardware that is required to perform all subsequent tests. Level 1 is composed of two separate diagnostics: the DCU self-test and the kernel hardware diagnostic.

The DCU self-test is a program stored in ROM on the DCU. Initiated on power-up or via a command typed on the console, it verifies most of the DCU and PSC hardware and provides a pass/fail indication on both the console and a group of LEDs on the DCU board.

The kernel hardware diagnostic verifies that portion of the CPU hardware needed to load and run the next level of tests, the fault-locating microdiagnostics. The kernel hardware diagnostic consists of a set of DCU commands that are stored on flexible disc and loaded into the DCU via the system console for execution. In this diagnostic, fundamental CPU operations (microsequencing, basic microinstruction decoding, and main data paths) are checked out. Essentially this is a single-cycle test that is both applied and observed by the DCU. The DCU forces lines of microcode into the CPU, clocks the system, and then checks the results. In this manner, the basic CPU kernel is verified by an independent processor.

Level 2—Fault-Locating Microdiagnostics

Fault-locating microdiagnostics perform the bulk of the testing done on the Series 64 mainframe. They are used to isolate faults down to the board level. Primarily designed for field use, these diagnostics will, upon a test failure, give the user a list of boards that could contain the fault. Usually this list will contain four or fewer boards.

Like the kernel hardware diagnostic, these tests are stored on flexible disc and are loaded by the DCU. Unlike the kernel diagnostic, they are written in CPU microcode. The DCU's job is to transfer the tests to WCS, initiate them, and interpret the results. The advantage of testing in microcode is that hardware can be tested in small increments. Because the microdiagnostics are quite large, they are broken into five sections and the hardware is tested one subsystem at a time.

To assure board-level isolation, testing is done on a functional circuit basis. A functional circuit is defined as several gates connected to implement a given logical function. With the knowledge of which functional circuit is under test and how it is partitioned across the boards in the system, it becomes possible to specify which set of boards must contain the fault. This information is embedded within each test. When a failure occurs, the DCU extracts this data from the microcode and displays the suspected printed circuit assemblies in order of fault rank.

The DCU is also used to enhance the isolation capability of the fault-locating microdiagnostics. Some tests use the DCU to access hardware circuits that are otherwise inaccessible to the microcode. This is accomplished by embedding

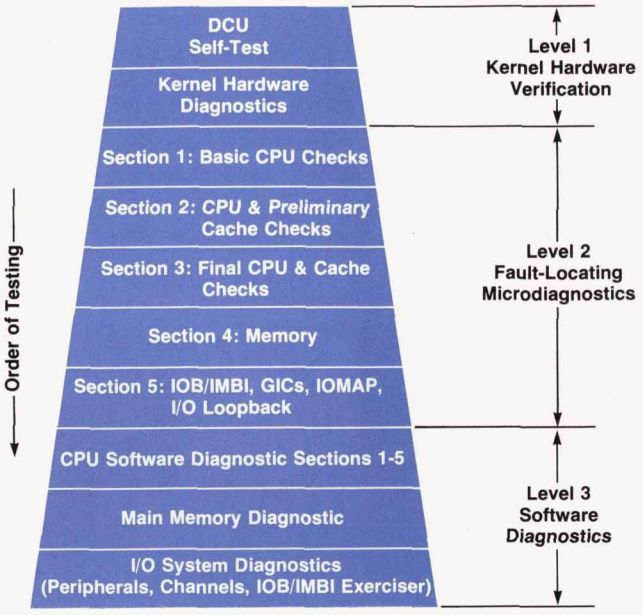


Fig. 4. *The diagnostics test the computer in hierarchical fashion. Successive tests involve increasing amounts of hardware.*

DCU requests in the microdiagnostic. Upon receiving a request, the DCU freezes the system, retrieves a command from a designated register, and restarts the microprogram after the command has been completed.

Level 3—Software Diagnostics

The first two levels of diagnostic tests do a thorough check of all the CPU hardware. However, they do not rigorously test all the CPU instructions, nor do they exercise all of main memory, nor do they verify the proper operation of all the I/O cards and peripherals. To test all this hardware, many software diagnostics have been developed or adapted from other HP 3000 Computers. These are written in higher-level languages rather than microcode.

The CPU software diagnostics perform an extensive test of all the CPU machine instructions, expecially those unique to the Series 64. These tests do not isolate faults to the board level, but rather give a pass/fail indication.

The main memory diagnostic tests the main memory arrays as well as error correction and logging circuitry. Any array failures are isolated to the faulty ICs, which can be replaced in the field.

The I/O system diagnostics are test programs that have been adapted from those written for the HP 3000 Series 33, 40, and 44. They check the entire I/O system of the Series 64, including all boards that can be plugged into the I/O card cage. There are also diagnostics for a number of tapes, discs, and printers that can be connected to the mainframe.

All software diagnostics are stored on magnetic tape and are loaded and run by using the diagnostic utility system (DUS), a simplified operating system. The role of the DCU in these tests is very small. These diagnostics are intended for use by the CE rather than the customer and require some specialized knowledge to configure and execute.

One Hour to Repair

This combination of diagnostic hardware and microcode allows unprecedented supportability in the field. An indication of the success of this effort has been the achievement of one of the primary design goals of the Series 64: an MTTR (mean time to repair) of less than one hour. Hence the Series 64 may be regarded as a computer with very high availability.

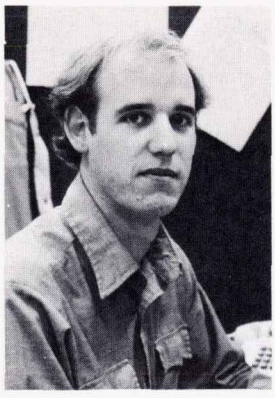
Acknowledgments

The authors wish to call special attention to the contributions of Kevin McGrath, who laid the foundations for the diagnostic philosophy and was instrumental in the development of the microdiagnostics. Special thanks are also due W. Grant Grovenburg for his work on the initial design of the DCU, Norm Galassi, Randy Jones, and Tom Graham for their ability to develop 56K bytes of DCU firmware against all odds, and Mehraban Jam for his outstanding work on the final design of the PSC.

Richard F. DeGabriele

Rick DeGabriele is a native of Sacramento, California and attended California State Polytechnic University (San Luis Obispo) where he received the BSET degree in 1976. He joined HP that year and was the diagnostic group project leader for the HP 3000 Series 64. Rick recently transferred to HP's Roseville Division to work in the 2250 system manufacturing group. He is married and lives in Citrus Heights, California. He enjoys water skiing, football, and baseball, and is restoring a 1966 Corvette.

David J. Ashkenas



David Ashkenas was a summer intern at HP's Stanford Park Division in 1977 while studying for the BSEE degree at Stanford University. He received that degree in 1978 and joined HP full time. He was responsible for the initial design of the HP 3000 Series 64 power system controller and the hardware design of the diagnostic control unit. In 1979 he received his MSEE degree from Stanford. David was born in Ithaca, New York and raised in Pasadena, California. He lives in Sunnyvale, California and is interested in sailing, international travel, and Italian sports cars.

A High-Performance Memory System with Growth Capability

by Ken M. Hodor and Malcolm E. Woodward

IN THE BEGINNING, computers consisted of a simple central processing unit with a few registers, some memory, and an input/output system (Fig. 1). As technology advanced, the central processing unit grew and acquired more registers and the RAM (random-access memory) size grew as new higher-density RAMs became available.

The overriding goal of the central processor became high speed, and the overriding goal of RAM design became low cost. As RAM became a larger portion of the cost of a computer system, cheaper RAMs were developed. More registers were used to fill the need for fast access to data.

As the conflict between memory and CPU grew, there evolved the cache memory. This memory contains a small subset of the information in the main memory array, but its speed of access is much closer to what the central processor requires. However, the cache integrated circuits are relatively expensive compared to the integrated circuits used in the main memory array.

While this was going on, computer input/output systems were developing a need for larger and larger buffer storage areas. I/O buffers attempt to match the speed of the I/O devices with the speed of the main memory array. As the speed of the CPU has increased, the demand for more I/O memory has also increased.

This is where the HP 3000 Series 64 is today. Fig. 2 is a basic block diagram of the system. The basic modules are the central processor and cache memory, the input/output adapters (IOA), and the memory module (MEM). These modules communicate with one another over the central system bus (CSB).

The central system bus is a synchronous, general-purpose, 14-MHz bus. Data transfers to and from memory are on a 16-byte block basis. Addresses, data, and messages share the same 32-bit path with two parity bits. Transfer rates as high as 56 megabytes per second can be achieved, although the current memory modules limit this to 18 megabytes per second.

Basic Information Transfer

The basic information transfer on the CSB is either to read data from memory or to write information to memory. To do a write to memory, a module sends the address followed by the block of data to be written to memory. This takes a total of five clock cycles to go across the CSB, one clock cycle for the address and four for the four 32-bit data words.

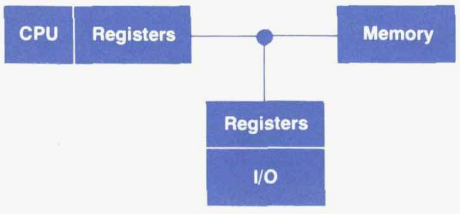


Fig. 1. *Simple computer block diagram.*

For a read operation, data can be in the memory module or in any of the caches in the system. For a read cycle a module sends an address out to memory via the CSB and every module looks to see if it has the most current data at that address. Only the copy held by the last module to access the data will be flagged as valid.

If memory has the only copy of the information, the memory module provides it to the requesting module. If one of the caches on the CSB has a more up-to-date copy of the data requested, this module must give it up and supply it to the requesting module. A module is said to abort the memory request and become memory, supplying the data. This feature increases the effective memory bandwidth, thus increasing the system performance.

The CSB is also used for sending messages. These messages may be originated by the CPU or by one of the system modules. Messages are used by the CPU to request the status of the various modules and to control their operations. Messages are used by the system modules to report status changes and to reply to CPU requests. One example of a message is the CPU's requesting the memory module to report memory size. Another example of a message transaction is the I/O adapter's reporting that a device is requesting service.

Messages are also used extensively by the microdiagnostics. The diagnostics use messages to set up conditions for various test cases. They also use them in the same manner that the system would during normal operations.

Memory Module

Fig. 3 is a block diagram of the memory module. To read from memory an address is sent over the CSB and through the common bus interface (CBI), through the buffers on the memory module and out to the main memory arrays (MMAs). Each MMA responds to an address range. The

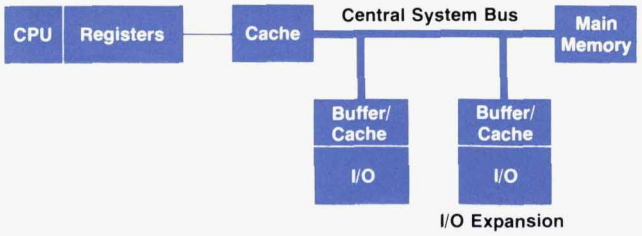


Fig. 2. *HP 3000 Series 64 block diagram showing cache and I/O buffer memories used to match modules that operate at different speeds.*

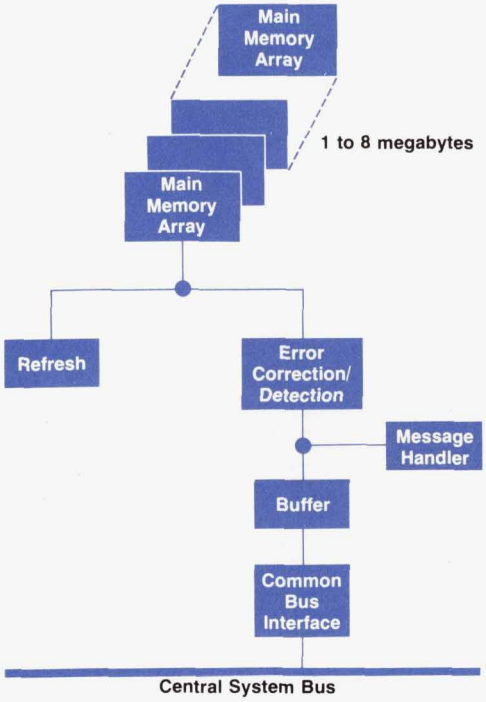


Fig. 3. *Memory module block diagram.*

addresses are generated by adders on the arrays. Each MMA added to the system adds the array size coming from the board next to it to the amount of memory available on its board. This allows MMAs with various memory sizes to be mixed within a system. The memory address goes out to all of the MMAs and the MMA that has the correct address range will respond with the data. The data will come out of the array in four 39-bit words, each consisting of 32 bits of data and seven bits for error detection and correction.

As the information goes through the error correction and detection logic, single-bit errors are corrected and logged in the error-logging RAMs. Double-bit errors are not corrected, but are sent back to the requesting module as they came from the MMA. An error condition is sent to the requesting module with the bad word. To get to the requesting module they must go through the CBI.

For a write to memory the information comes through the CBI and into the buffer. The RAMs cannot handle the information as fast as the CSB can deliver it, so buffering is used to handle the mismatch in speed. For a write the address is sent to the MMAs followed by the four 39-bit words, 32 bits of data and seven syndrome bits for error correction and detection. The seven syndrome bits are generated in the error correction and detection circuitry and sent to the MMAs.

The RAMs are dynamic and must be refreshed periodically, so there is refresh logic within the memory module. A refresh takes priority over an access of memory. If a refresh is needed it can hold off the reading or writing of data in the MMAs. Thus the amount of time to access memory is variable. Refresh has a minimal effect on the effective memory bandwidth.

The Series 64 has automatic power-fail/auto-restart capability. If the power to the machine should fail, all of the information in the caches will be sent to memory within 5 milliseconds. Memory has battery backup power, which is supplied to the refresh logic, RAMs, and other necessary circuitry to keep the information alive in memory. When the power returns, the system automatically resumes where it left off.

Messages sent to memory are used for determining memory size, to respond to various error conditions, for diagnostic purposes to check out various paths, and to find out if any read of memory has caused a single-bit error and which RAM caused the error condition.

Cache Module

The cache module consists of the cache memory array (CMA) and the cache array controller (CAC) printed circuit boards (see Fig. 4). The cache serves two functions. First, it is a high-speed buffer between the CPU and the main memory module. Second the cache is a communications path between the CPU and all of the other system modules.

There are five data paths to and from the cache and one between the CAC and CMA. These are:

- The processor data bus (PDB)
- The cache address bus (CAB)
- The address and data bus (ADATA)
- The data bus (DATA)
- The intercache bus (ICB)
- The cache data bus (CDB).

The processor data bus provides data to the cache memory array and commands to the cache array controller. The cache address bus provides the cache with address information from the CPU and messages from the CPU to other system modules. The ADATA bus provides the cache with addresses and data from the central system bus. The cache data bus provides the CPU with data from the cache and the central system bus. The DATA bus provides addresses and data to the central system bus from the cache. The intercache bus provides an address path between the CAC and the CMA.

The cache memory array consists of 8K bytes of high-speed ECL RAM which serves as a high-speed buffer be-

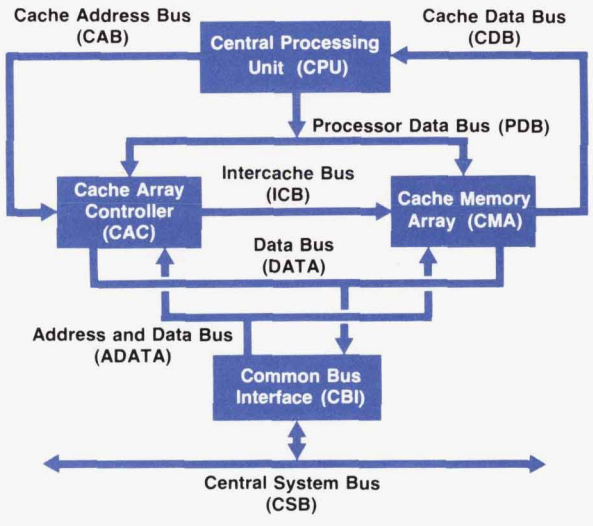


Fig. 4. *Cache module block diagram.*

tween main memory and the CPU. The array is organized into two data sets. Each data set contains 4K bytes of data arranged into 32-bit words. The CMA also provides the data path for incoming messages from the system modules to the CPU.

The cache array controller contains the control logic for the cache. The CAC keeps track of what data is contained in the cache memory array. When a memory request is issued by the CPU or any of the system modules, the CAC determines whether the CMA contains the requested information. If it does, the CAC takes the necessary action to supply the data to the requester. If the CMA doesn't contain the requested data, the actions taken by the CAC depend upon the requester. If the requester is another system module, no action is taken by the cache array controller. If the requester is the CPU, the CAC tells the CPU that the data is not available and initiates a memory request for the memory block (16 bytes) that contains the requested data. When the cache memory array receives the requested block, the CAC instructs the CMA to supply the requested data to the CPU via the cache data bus and informs the CPU that the requested data is available.

Bus Control Commands

Communications between the CPU and other system modules, including the cache, take the form of messages. The CPU initiates these messages by sending bus control (BUSC) commands to the cache array controller. BUSC commands are used in normal system operations and in diagnostic operations. When the CPU issues a BUSC command, the CAC determines whether the message is to be sent to another system module or to be acted upon by the cache. The messages the cache acts upon are:

- Read cache status
- Write cache status
- Write tag data
- Verify tag data
- Read incoming message.

The send word message, SNDWRD, is sent to the addressed system module via the central system bus. The SNDWRD message is addressed to main memory, one of the I/O modules, or the cache. The messages that are sent to the other system modules are used to:

- Determine system configuration
- Report status changes of modules to the CPU
- Read error conditions and the memory error log
- Perform diagnostic exercising.

The cache may interrupt the CPU in two ways. The first is a message interrupt. This interrupt occurs when the cache array controller detects an incoming message from one of the system modules. The CAC responds to an incoming message by issuing the MSGINT signal to the CPU and holding the message word until the CPU has had time to read it. MSGINT causes the CPU to branch to the proper place in microcode and read the incoming message. The CPU responds to the message interrupt by issuing BUSC commands to read the incoming message.

The second interrupt is an error condition (CACHE ERROR) which is generated by the CAC when a catastrophic error has been detected in the cache memory array data or in the tag and status information relating to the CMA data. The response to a CACHE ERROR interrupt is a system halt.

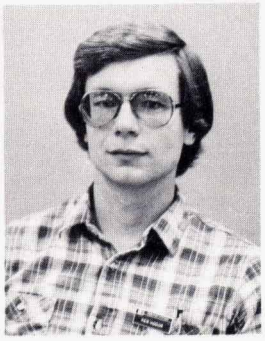
I/O Adapter

The I/O adapter, which interfaces the Series 64 to an I/O bay, has a small cache used as a buffer. The I/O cache is structured as a four-set associative memory. Each set is one block deep. The article on page 18 goes into more detail.

The I/O adapter also uses the central system bus for the communications with the other modules. This module also accesses the CSB via a common bus interface.

Acknowledgments

The authors would like to acknowledge the contributions made by Rick Amerson, Mark Linsky, Karen Meinert, Stanley Anderson, Jim Socha, Charles Andrews, Skip LaFetra, John Figueroa, Paul Rogers, Anthony Boresch and Fred Bird to this part of the Series 64 project.



Ken M. Hodor

Born and raised in Hammond, Indiana, Ken Hodor received his BSE degree from Purdue University and joined HP in 1973. He has designed calculator ICs, electronics for a laser interferometer, and the central system bus and memory for the HP 3000 Series 64. Ken is a scuba diver, a home computer hobbyist, a cyclist and a swimmer. He's married, has two daughters, lives in Sunnyvale, California, and is currently helping introduce microcomputers at a local elementary school.

Malcolm E. Woodward

Woody Woodward started his HP career in 1972 as a technician on the 2100 Computer manufacturing line. He soon moved into the R&D lab as a technician and later became a design engineer, contributing to the design of the HP 300 and HP 3000 Series 64 Computers. Before coming to HP, he served in the U.S. Marine Corps for over six years as an electronics technician, instructor, and technical supervisor. Born in Ontario, Oregon, Woody is married, has four children, and lives in San Jose, California. His hobbies include amateur radio, automobile mechanics, and pistol and rifle shooting. He serves as amateur radio emergency coordinator for the City of San Jose and several county agencies.

An Input/Output System for a 1-MIPS Computer

by W. Gordon Matheson and J. Marcus Stewart

WHEN THE HIGH-SPEED CPU and central system bus for the HP 3000 Series 64 were first conceived, considerable attention was given to the goals and design approach for the I/O system hardware. What emerged was a system that is compatible with earlier systems, yet provides for expansion and future growth.

On the HP 3000 Series 30, 33, 40, and 44 Computer Systems, all I/O channels, the CPU, and main memory communicate on the intermodule bus (IMB). The IMB is an asynchronous TTL-technology backplane bus that will support up to 15 I/O ports, depending on total length and configuration (see system diagram, Fig. 1, and IMB summary in Table 1). Previously there have been two types of I/O ports available for the IMB: the 31262A General I/O Channel, which connects HP-IB (IEEE 488) devices to the system, and the 31642A Asynchronous Data Communication Channel for RS-232-C terminals and modems. Many kinds of peripherals attach to these two channels, including disc memories, magnetic tapes, printers, printer interfaces, laser printers, and CRT terminals.

The input/output hardware of the HP 300 Computer is similar to that of the HP 3000 family members mentioned above.[1]

Series 64 I/O System Goals

Since the HP 3000 Series 64 is a member of a family, and since the goal was to support many of the same peripherals that are available for the other family members, the strategy for the I/O system was to retain the IMB as the primary attachment of I/O to the system if it could be made compatible with other system requirements. This allows an easier upgrade path for users, who can use many of the same cables, I/O channels, and some IMB-compatible device controllers that are used in earlier systems. Of course, it also allowed HP to implement a system that had been proven in other HP computers, and to minimize software driver development, hardware design, and control program algorithm development. Other goals of the HP 3000 Series 64 I/O system were improved I/O throughput and support of larger numbers of peripherals than on the next fastest family member, the Series 44.

To help achieve these goals and assure that the higher CPU performance is augmented by higher I/O performance, three major improvements have been made:

1. The new 30144A Advanced Terminal Processor (ATP) is used for datacomm interfacing (see article, page 22). This eliminates much of the CPU intervention required by previous products for datacomm transfers. It allows attachment of up to 96 datacomm connections per channel on the intermodule bus. The ATP has an extensive DMA (direct memory access) facility, whereas the older 31264A Asynchronous Data Communication Channel required channel program intervention by the processor for every byte transferred.
2. A round-robin approach was adopted for servicing the channel programs for devices on the general I/O channels. This is a rotating priority scheme that gives equal service to all devices. With large system configurations having large numbers of high-use discs, round-robin servicing eliminates the lockout of lower-priority devices by higher-priority devices.
3. Provisions for multiple I/O buses. The throughput of the central system bus is maximized if several intermodule buses are able to attach to the CSB and perform DMA simultaneously. The throughput of the IMB is much lower than that of the CSB, and the CPU averages a small portion of the maximum bandwidth of the CSB, so multiple IMBs can provide enhanced DMA throughput and allow for attachment of larger numbers of I/O channels than could be supported by a single IMB.

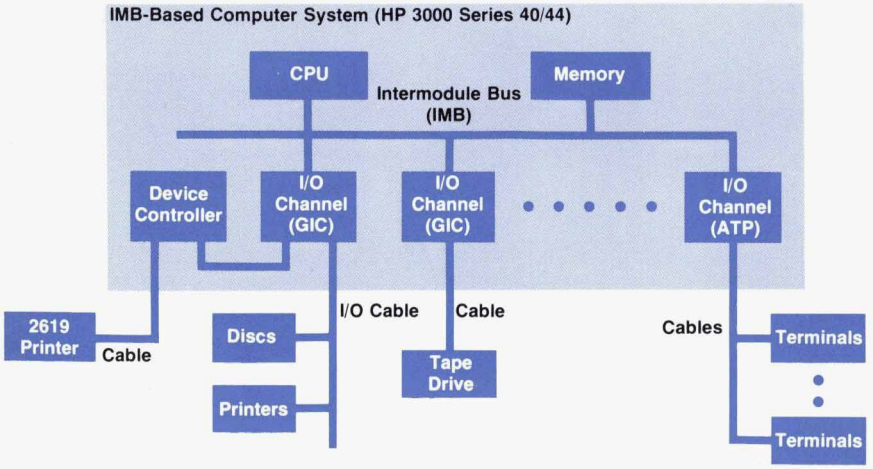


Fig. 1. *HP 3000 family input/output system diagram. All I/O channels, the CPU, and main memory communicate on the intermodule bus (IMB).*

While the I/O subsystem for the HP 3000 Series 64 is designed for optimized attachment to the IMB, it was developed under a consistent set of system rules for I/O operations and hardware interface. Non-IMB types of I/O hardware may be attached to the system if the need arises, and the HP 3000 Series 64 will be able to keep pace with future I/O system developments.

A final and extremely important goal was to make I/O hardware problems easy to diagnose accurately and fast to repair. This means that it must be possible to check each level of hardware interface quite thoroughly by diagnostics before extending testing to levels farther out. For example, an untested logical block on an I/O channel could cause a channel failure to be diagnosed as a device failure. To achieve the goal of diagnosability, a significant part of each hardware component consists of diagnostic features that allow on-board loopback of data and control bits, checking of error detection logic, and simulating operations that normally involve the next level of hardware.

I/O Adapters

The CSB was developed for very high-speed data transfers and synchronized operation. Its implementation was necessary to achieve the memory and CPU performance goals for the system. Unfortunately, the CSB can't be used for direct attachment of input/output channels. The main difficulty is that it is necessary to restrict the physical length of the bus to maintain the capability of high transfer rates; otherwise, it would take too long for a signal to propagate from one end of the bus to the other. Physically, the CSB is less than 15 cm long. This places severe restrictions on the number of printed circuit assemblies that can be plugged into the bus. For this and other reasons, a separate I/O bus is required for the HP 3000 Series 64, and the IMB was chosen to do the job. All I/O channels (general I/O channels, advanced terminal processors, intelligent network processors, etc.) plug into an IMB-type backplane in the I/O cardcage, and an I/O adapter has been developed to provide for communication between the two buses.

An I/O adapter, in general, is any interface between the CSB, which supports the CPU and main memory, and another bus that supports I/O channels and devices. On the HP 3000 Series 64, the present I/O bus is the IMB, but some other bus could be used so long as the proper I/O adapter is provided.

As an integral part of the Series 64, the I/O adapter is required to perform the following functions:

- Electrical translations. The CSB is an ECL-level bus. If the I/O bus uses other logic levels (the IMB, for example, is a TTL bus), it is up to the I/O adapter to provide translation between the two.
- Command and service request translations. On the CSB, all commands and requests are sent in the form of messages. An I/O bus may have a similar scheme, or it may use separate lines for service requests and for issuing commands, as the IMB does. In either case, the I/O adapter must translate requests for service on the I/O bus, whatever their form, to messages addressed to the CPU, and messages from the CPU must be properly translated to commands on the I/O bus that use the proper handshaking protocol.
- Memory buffer. All memory transactions on the CSB take place in 16-byte chunks in the form of four contiguous 32-bit words transferred in four successive clock cycles. If the I/O bus uses any other size as its minimum addressable data block (on the IMB, it is one 16-bit word), the I/O adapter must provide some sort of buffer to convert one size to the other.
- Synchronization to system clock. The CSB is a fully synchronous bus. The I/O bus may be either asychronous as the IMB is, or synchronized to another clock. In either case, it is up to the I/O adapter to synchronize the I/O bus to the CSB.

The Series 64 system currently supports up to two I/O adapters, making it possible to increase significantly the I/O bandwidth of the system, since two separate I/O buses can transfer data independently at the same time.

Interfacing the IMB to the Series 64

The IMB I/O adapter for the Series 64 consists of three printed circuit assemblies and the associated interconnecting cables. The three printed circuit assemblies are the common bus interface (CBI), the I/O buffer (IOB), and the IMB interface (IMBI) assemblies. They are connected as shown in Fig. 2. The IOB and CBI both reside in the main (CPU) cardcage and the IMBI resides in the IMB cardcage. The common bus interface is the universal assembly that interfaces modules on the central system bus to the CSB itself. Communication signals between the I/O buffer and the common bus interface travel along the backplane and through a very short 2-cm frontplane cable. The I/O buffer is connected to the intermodule bus interface with two 1.5-m transmission line cables. Each of these cables carries 30

Table I

Intermodule Bus (IMB) and Central System Bus (CSB) Characteristics

Characteristic	IMB	CSB
Technology	TTL logic	ECL logic
Address bus	24 bits	32 bits, multiplexed
Data bus	16 bits	
Handshake	asynchronous, separate address, data	synchronous, multiplexed address, data
Memory data	16-bit word	blocks of four 32-bit words
Memory operations	read, write word	variety of read/write block
I/O operations	I/O read, write (broadcast/selective)	two types of 32-bit messages
DMA access	slot number priority	rotating priority
Channel program request	asynchronous signal	unsolicited messages
Interrupt Request	asynchronous signal	

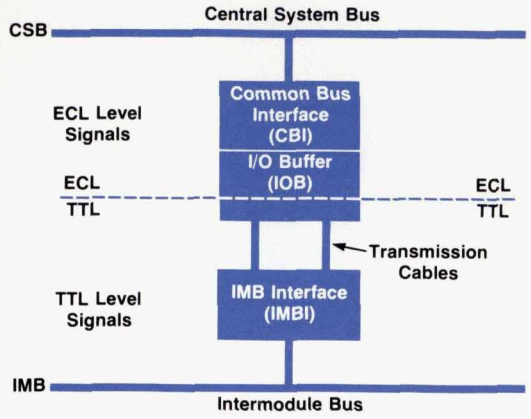


Fig. 2. *The I/O adapter interfaces the Series 64 to the IMB. It consists of three printed circuit assemblies.*

signal and 60 associated ground conductors, providing balanced 92-ohm paths for the TTL-level interface signals between the IOB and IMBI.

In the IMB I/O adapter, signal level translation between the central system bus and the intermodule bus is done by the I/O buffer. Referring again to Fig. 2, the broken line across the IOB indicates the boundary between ECL-level signals and TTL-level signals. Except for the level translators, the IOB's internal logic is implemented with ECL devices.

Another general requirement of an I/O adapter is the translation of the sixteen-byte memory transfers on the CSB to whatever size the I/O bus uses. The IMB works with one sixteen-bit word at a time, and the IOB takes care of the buffering between the two. The IOB uses a four-set, fully associative cache with a two-cycle (150 nanosecond) access time as the memory buffer. "Fully associative" means that the IOB can store copies of up to four sixteen-byte blocks from main memory at the same time, and any of the four blocks can be associated with any arbitrary memory address. In terms of performance, this means the IOB can support up to four simultaneous high-speed direct memory accesses (DMAs) at the same time without any thrashing (excessive swapping) in the cache. Besides acting as memory to the intermodule bus, the IOB supports central system bus memory activity by checking addresses on the CSB and providing any blocks requested for which it has a valid copy.

Still another requirement of the I/O adapter is translation of commands between the CSB and the I/O bus (see Fig. 3). The IMB I/O adapter's specific task is to translate channel program service and interrupt requests on the IMB to messages to the CPU, and to translate messages from the CPU into global or addressed commands on the IMB. Most of this is done by the IMBI, with the IOB acting as a level translator and traffic controller. When a request line on the IMB is asserted, the IMB I/O adapter formulates a 32-bit unsolicited message and transmits it to the CPU, then disables further messages until they are reenabled by commands from the CPU. IMB and I/O adapter commands are sent as a single 32-bit message from the CPU. When the I/O adapter completes the handshaking of the command it formulates a 32-bit response message containing 16 bits of status (parity error on command, IMB handshake timeout, etc.) and the contents of the IMB data bus when the command execution is completed.

One of the biggest challenges for the IMB I/O adapter is that of synchronization. Since the IMB is a totally asynchronous bus and the CSB is totally synchronous, all transactions on the IMB have to be synchronized with the Series 64's master clock before any interaction with the CSB can take place. All of this is done by the IMBI, which is itself synchronized with the system clock. What makes this task challenging is the high speed of the Series 64's master clock: the cycle time is only 75 ns. The normal method for synchronizing any signal is to run it through a series of D-type flip-flops, with each successive stage minimizing the chance of an unstable output at the end of the chain. Normally, the higher the clock speed, the more levels of synchronization are needed to assure stability. However, each level of synchronization adds delay to any signal that must be synchronized and reduces the bandwidth of the bus. The IMBI, therefore, uses a specially designed state transition system that imposes minimum extra delay on the signal being synchronized by being tolerant of metastable states. Thus the IMBI is designed with only one level of synchronization on each signal, with the state machine adding the extra levels needed for high reliability.

Expansion and Growth

To support more than one IMB for the performance and configurability issues discussed earlier, a few modifications were necessary both to software and to the IMB.

The I/O bay is wide enough to support a 24-slot IMB backplane, which is divided into two IMBs of eight slots and sixteen slots respectively (see Fig. 4). The IMBI resides in the slot nearest the CPU bay of each IMB. Channel and device controller boards may be inserted in the remaining slots with a few configuration guidelines. All cables leaving the system do so through a junction panel, which provides a solid chassis ground for them. The larger IMB accommodates

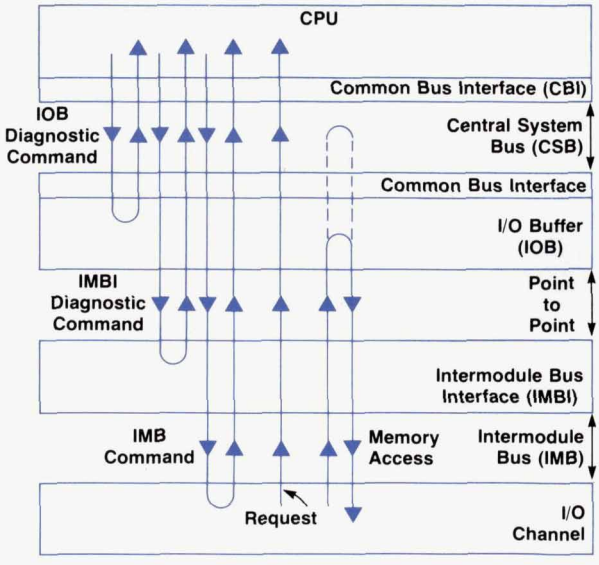


Fig. 3. *The I/O adapter translates various commands between the central system bus (CSB) and the I/O channel.*

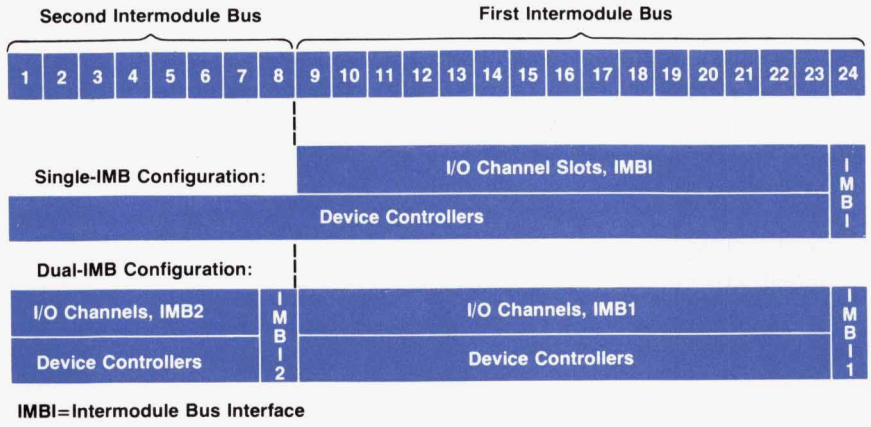


Fig. 4. *The Series 64 I/O bay holds two intermodule buses (IMBs) of eight and sixteen slots.*

a reasonable system I/O configuration before space or performance requirements indicate the need for splitting the bandwidth or adding more channels than can fit onto a single IMB. The closer an I/O channel is to the IMBI, the higher its priority for memory access.

The Series 64 has a device reference table structure similar to the other members of the family. That is, for each of the eight devices per channel, a four-word entry in main memory provides the following information:

1. The address of the channel program for that device.
2. The address of the channel program variable area for the device (to save parameters relating to execution of the channel programs).
3. The label of the interrupt handler code segment for external interrupts from the device.
4. A word for run-time execution state information for the device (e.g., the reason a channel program is suspended).

The total device reference table length has been expanded from 120 to 512 entries to accomodate up to four I/O adapters for future growth, and its starting location is not fixed in hardware, but has been made indirect through a pointer in reserved main memory.

The RMSK and SMSK instructions (read mask and set mask) specify which I/O channels are allowed to generate interrupts to the CPU. These instructions also had to change, since software is providing for four I/O adapters, each of which can have 15 channels (channel 0 is not used on the IMB). The interrupt masks are now contained in words 32-35 of memory.

Similarly, all of the I/O instructions have been expanded to include specification of the I/O adapter number.

It is a testimonial to the modularity of the I/O system software that these changes were implemented with minimal time and effort, since only a few common system subroutines required modification. The MPE-IV operating system will work equally well with single-IMB systems or with the Series 64, since it can distinguish the hardware system under which it is running.

An intelligent processor on an IMB might need to know where its device reference table is, if it is executing its own channel programs. Therefore, two extra lines were added to the IMB to let the smart channels recognize the IMB to which they are attached.

Summary

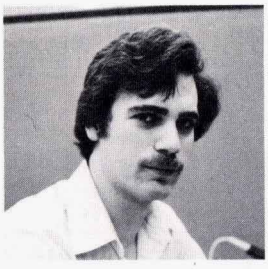
The HP 3000 Series 64 I/O system uses much of the same hardware and supports the same peripherals as the Series 30, 33, 40, and 44 while providing an expansion path to meet the needs of users of larger and faster systems. This was also accomplished with relatively minor changes in the I/O software. Yet there is also provision for adaptability to future requirements in the I/O area.

Acknowledgments

Series 64 I/O development was managed by Mark Linsky. Neil Wilheim developed the original design concepts for the I/O adapter. Paul Chang developed the CPU firmware to support all I/O functions. Rick DeGabriele and Tom Graham wrote the I/O microdiagnostics. Skip LaFetra assisted in several phases of the project and contributed provisions for performance enhancements.

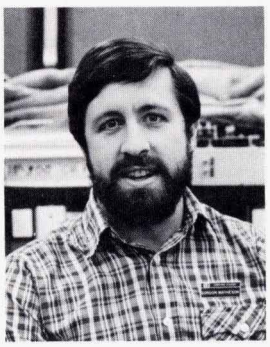
Reference

1. W.G. Matheson, "A Computer Input/Output System Based on the HP Interface Bus," Hewlett-Packard Journal, July 1979.



J. Marcus Stewart

Marc Stewart received his BSEE degree from the University of California at Davis in 1979. He joined HP the same year as a production engineer and a year later became a development engineer on the HP 3000 Series 64 project. He is married, lives in San Jose, California, is active in his church, and enjoys skiing, bicycling, swimming, and model airplanes.



W. Gordon Matheson

With HP since 1973, Gordon Matheson has done CPU, DMA, and microcode design for HP 1000 Computers, developed hardware and firmware for the HP 300 Computer I/O system, and developed the I/O adapter for the HP 3000 Series 64. He's also worked on local data networks and I/O standards (ANSI X3.T9.2). Born in Glendora, California, he now lives in San Jose. He's married, has four children between 2 and 7 years old, plays piano, and considers himself a full-time parent.

The Advanced Terminal Processor: A New Terminal I/O Controller for the HP 3000

by James E. Beetem

THE ADVANCED TERMINAL PROCESSOR (ATP) is a new terminal I/O controller designed specifically for the HP 3000 Series 64. Although the basic ATP design is compatible with any HP 3000 that uses the intermodule bus (IMB), its design center is a large system with the processing power of the Series 64.

The HP 3000 Terminal I/O Environment

The first step in designing a new terminal I/O controller is to establish the maximum number of terminals to be supported and the demands that each terminal will place on the system. The terminal I/O loads expected on the Series 64 can be predicted by projecting the same loads on the widely used HP 3000 Series III to a system with four to six times its processing power. The resulting numbers can be adjusted further to allow for the trends that are apparent in the HP 3000 applications environment.

A typical HP 3000 Series III has 35 terminals connected, three of them via modems. Usually, during the busiest part of the day, 25 of the terminals have active sessions and 20 terminals are actually in use. About two-thirds of the terminals are supplied by HP and the rest come from a wide variety of vendors.

The maximum supported number of terminal I/O ports for the Series III is 64. All 64 may be operated as modem ports, although the typical system has only three modem ports.

The number of terminals that an HP 3000 can operate with acceptable user response times is strongly dependent upon the demands the application package places on the system. Customers have reported satisfactory response times with their application package supporting as few as ten and as many as 48 terminals. The typical numbers chosen for this analysis represent a 75th percentile case—only about 25% of HP 3000 customers have more terminals connected to their systems.

On the typical HP 3000 Series III, the average I/O load per terminal during peak hours is about 20 characters per second. The load is generated by processing one user transaction every 20 seconds at each terminal. A transaction involves writing a message to the screen of the terminal and reading in the operator's reply. Usually, the output is about 300 characters and the input is about 100 characters. The transaction time and the number of I/O characters vary over a very wide range and are strongly dependent upon the specific application being run.

The actual I/O load in any short period of time varies widely from the average. Peaks of up to 1000 characters per second occur occasionally when several terminal devices operate concurrently.

Several trends are evident that will have an impact on terminal I/O. First, as terminal prices decline, more and more terminals that are lightly used are attached to the system. These terminals demand much less of the system's resources. On the other hand, the new intelligent terminals have substantial processing power and frequently move masses of data to and from the system. These devices can generate very heavy I/O loads.

Another class of devices placing significant terminal I/O loads on the system consists of character printers such as the HP 2631B and HP 2601A. When in operation, these devices may require as many as 180 characters per second. Because these devices tend to operate during the system's peak loads, they can sharply increase the average I/O loads.

Most of the newer terminal devices allow higher line speeds. Almost all terminals now offer a maximum line

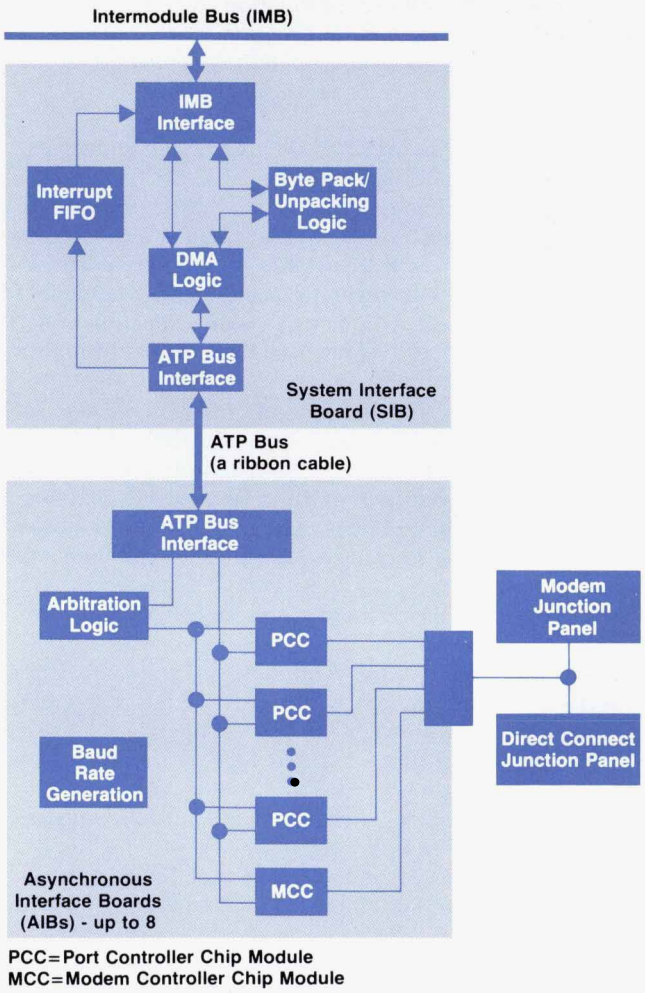


Fig. 1. *Block diagram of the advanced terminal processor (ATP) for the HP 3000 Series 64. Up to 256 terminals can be connected.*

speed of 9600 bits per second (bps). New designs call for a speed of 19,200 bps. These higher data rates will cause fewer but higher peak I/O loads.

Based on this analysis, the ATP for the Series 64 was designed to allow a maximum configuration of 256 terminals. It expects to have about 200 terminals connected in a typical installation. About 125 devices will have active sessions with 100 actually in use at any given time. The average I/O load is expected to be about 4000 characters per second with occasional peaks of 20,000 characters per second. However, it is important that the reader understand that the numbers developed in this section are design center and maxima for a subsystem within the Series 64. Constraints external to the terminal I/O subsystem may not allow it to achieve its design maxima.

Other Design Considerations

The trend to higher line speeds causes another problem. The RS-232-C standard used to connect terminals to the HP 3000 has a limited speed/distance relationship. In theory, devices must be within 16 metres of the system. In actual installations, these limitations are often exceeded without problems at slower line speeds. At 9600 and 19,200 bps, the speed/distance relationship is a real limitation, restricting successful high-speed operation to terminals that are relatively close to the system.

A new modem interface standard is emerging: RS-449. When these new devices become widely used, the HP 3000 Series 64 will need to support them. Provisions are made in the new controller to allow customers to operate both RS-232-C and RS-449 types of modems on a single Series 64. Flexible junction panels support both modem standards, the existing direct connection standard, and a new direct connection standard with an improved speed/distance relationship.

Another consideration is how terminals are operated. Current HP 3000 software operates terminals in one of two modes. Each mode makes sharply differing demands upon the terminal I/O system. In character mode, a human operator is typing characters and using the editing facilities provided by the host computer system to prepare input data strings. Characters arrive at the computer infrequently, but each character must be compared to a large special character set and the appropriate actions must be taken.

In block mode, the terminal provides the editing facilities. When the input data string is ready, the terminal transmits it as one continuous block of data. The data arrives as fast as the terminal can transmit it, usually close to the line speed. At 9600 bps, one character arrives each millisecond. Very little processing is required, only a check for the character that marks the end of the block.

Under the current MPE (Multiprogramming Executive) terminal I/O system, the terminal I/O controller sometimes does not know which mode is in effect. Therefore, the ATP is designed to handle the character mode processing load at the block mode data rate.

ATP Design

The key element in the ATP design is the 6801 single-chip microcomputer. This device is programmed to provide the functions needed to support an HP 3000 terminal port. It is called the port controller chip (PCC). One of these chips is used for each terminal port.

The ATP hardware is designed around the PCC, providing the interfaces to the intermodule bus and to the terminal or modem (Fig. 1). The ATP software is designed for flexibility, to allow easy implementation and support of a wide range of facilities.

Port Controller Chip (PCC)

The 6801 PCC contains an enhanced 6800 microprocessor, 2K bytes of read-only memory (ROM), 128 bytes of random-access memory (RAM), a universal asynchronous receiver/transmitter (UART), and a timer with an edge-triggered counter. The UART provides the bit-serial interface to the terminal, eliminating the need for a separate UART chip. The microprocessor and its memory provide the processing power to do character handling without burdening the system processor. The PCC can handle the full line speed of 19,200 bits per second (1920 characters per second) in all modes of operation.

Since the PCC's microprocessor is dedicated to one port, the ROM-based software that operates the PCC can be very simple. There is no need for software to share resources or to resolve contention problems. The PCC is a simple slave processor, dedicated to doing a single task very quickly and efficiently.

The PCC checks the input and output data streams for special characters. System software may define two input edit sets and enable either of them for comparison with the input data stream. Two output sets are defined, one to scan the output stream and one to scan the input data stream during an output. These facilities give software complete control of the full-duplex operation of the PCC without impact on the system processor.

The PCC handles both the ENQ/ACK and X-ON/X-OFF flow control handshakes that prevent data overruns at the terminal. Also, it generates the time delays required by unbuffered teletypewriter devices that must perform physical movement of the carriage or print mechanism. Both the ENQ and ACK characters may be redefined by system software.

The PCC generates output parity and checks input parity when parity is enabled. In the 7-bit data mode, it clears the eighth input bit to zero and sets it to zero or one on output.

The PCC is provided with three paths to or from system memory. One path is used to send control information (orders) to the PCC from system software. This facility replaces the channel programs used by other intermodule-bus-based I/O devices and eliminates the need for channel program processing facilities.

The remaining two paths provide an input path to and an output path from system memory for data. Having two unidirectional paths rather than one bidirectional path allows the PCC to perform both input and output of data as the result of a single control order. Also, it allows the switch from output to input to occur very quickly, minimizing the ATP's response time.

The PCC does not have access to the address registers for these paths. This implementation simplifies the hardware design, but requires the PCC to interrupt the system processor to handle the backspace and delete-line special characters. These line-edit functions occur rarely and usually at human typing speeds, so the workload they generate for the

system processor is very low.

The PCC's timer is used for most terminal timeouts while the edge-triggered counter allows the PCC to do speed sensing accurately at all supported line speeds by measuring the width of the asynchronous protocol's character start bit.

A portion of the PCC's RAM is used for a 16-character input buffer. An input buffer of this size allows the ATP subsystem a full 16 character times to respond to a software interrupt (or eight milliseconds at the 19,200 bps line speed) before a data overrun can occur.

ATP Hardware

There are three elements in the ATP hardware design. The first element is an interface on the intermodule bus. This is provided by the system interface board (SIB). Then there is logic to resolve the asynchronous contention of many PCC's for the shared facilities. The board that contains this logic is called the asynchronous interface board (AIB). Last, there are junction panels to provide the terminal and modem interfaces (see Fig. 2).

System Interface Board (SIB)

Functionally, the SIB is a byte multiplexer channel optimized for the port controller chip (PCC) and terminal I/O. On one side, the SIB provides an interface to the intermodule bus (IMB), the standard I/O bus for the HP 3000. On the other side, it controls the ATP bus, which connects the SIB to as many as eight asynchronous interface boards (AIBs). This design allows one ATP subsystem to provide 96 terminal I/O ports while occupying one IMB channel address and nine IMB I/O slots. These are important considerations for a Series 64 system.

The SIB provides an interface to the software for control of the PCCs. It generates IMB requests for the PCCs and performs the tasks required to manage the asynchronous nature of the interface.

Since the PCCs and the ATP bus are byte-oriented while the IMB is word-oriented (two bytes), the SIB performs the byte packing and unpacking to translate from one bus to the other. It also allows data transfers to start or stop in either the left or right bytes of the two-byte words in system memory.

As the controller of the ATP bus, the SIB polls each AIB many times each second to see if one of the PCCs has data for or needs data from system memory or wants to generate a software interrupt. The SIB generates software interrupts upon request of a PCC. The PCC's identity number and a byte of status information are presented to software at the IMB interface. A first-in, first-out (FIFO) queue is maintained to smooth out peaks in the interrupt workload. To prevent interrupt overruns, polling on the ATP bus is suspended when the FIFO is full. The worst-case maximum data rate on the ATP bus is about 150,000 characters per second.

Each PCC is provided with three paths into system memory. The SIB implements these paths via the IMB's direct memory access (DMA) facility. This implementation allows the ATP to handle heavy I/O loads with very low overhead. All of the overhead on an ATP I/O transfer is incurred to start and stop the transfer. Once the transfer starts, the only overhead is one IMB DMA cycle for each two characters.

The SIB is designed to be easy to manufacture and support. The basic design is conservative, making little use of exotic or high-speed parts. The ability to diagnose a failed board has been designed into the ROM-based state machine that operates the board.

Asynchronous Interface Board (AIB)

Each AIB contains an interface to the ATP bus, twelve PCC modules, one modem controller chip (MCC) module, and the circuits that allow the PCC modules to share the ATP bus and the MCC. The AIB also contains the circuits that generate the UART baud rates and drive the cables to the junction panels.

The PCC and MCC modules and the ATP bus interface are attached to a bus on the AIB. Access to this bus is controlled by a ROM-based state machine. The state machine arbitrates

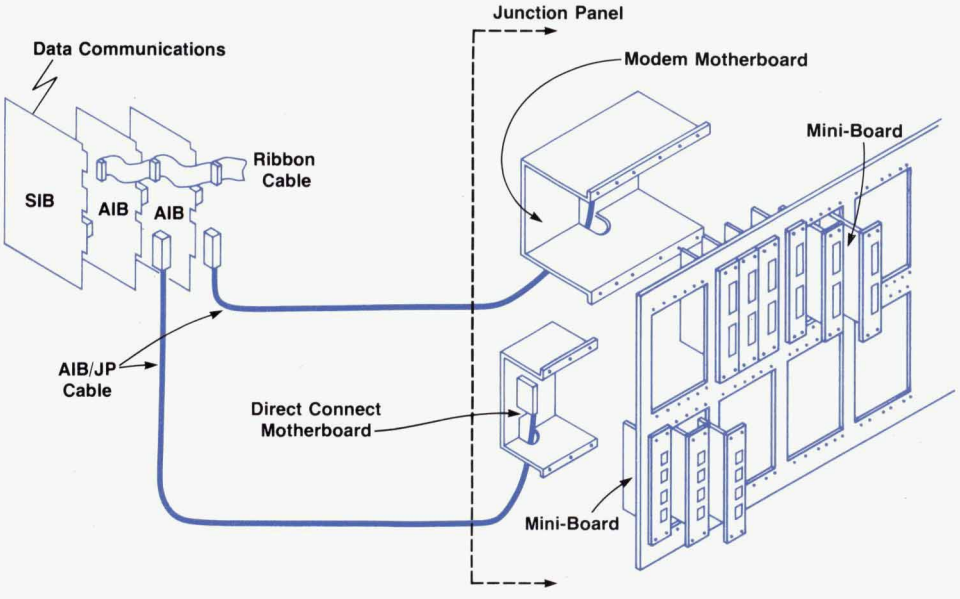


Fig. 2. *Advanced terminal processor junction panel mini-boards carry four direct connection ports or two modem ports each.*

requests for the AIB bus from the PCC modules, the MCC module and the ATP bus interface.

The MCC is another 6801 microcomputer. Its function is to allow each PCC to control the signals that operate a device connected to that port, usually a modem. Each PCC may read seven input signals and set eight output signals. This capability allows the PCC to control almost all devices offering a serial I/O capability.

The UART on the MCC is used to multiplex the 180 device control signals between the MCC and the junction panel. Use of the UART as a multiplexer eliminates the need for external multiplexer logic or a very large and expensive cable.

The AIB is designed to take advantage of the PCCs and the MCC to diagnose much of its own logic. Extensive use is made of the 6801 microcomputers to loop data from the system memory through the AIB's logic and back to system memory. This technique provides good troubleshooting facilities without adding special logic for testing.

Junction Panels

The ATP provides two types of junction panels. Direct connect junction panels are used to connect terminals in the local area. Modem junction panels provide the control signals required to operate full-duplex modems, allowing remotely located terminals to be connected to the HP 3000 via the public telephone system.

The ATP design calls for a high-speed, long-distance direct connection facility. This facility is implemented using EIA standard RS-422. The standard allows operation at a line speed of 100,000 bps at 4000 feet (1200 meters). However, since most terminals now available offer only RS-232-C interfaces, the ATP junction panels allow both types of interfaces, and interfaces can be easily mixed.

The ATP direct connection junction panels are implemented as small boards that plug into a motherboard. Each mini-board carries four ports, either RS-422 or RS-232-C.

To allow the maximum number of ports in the least possible space, a new more compact connector was designed to replace the 25-pin D subminiature connector in common use. The new connector uses only three wires for RS-232-C and five wires for RS-422. It is fully shielded and carefully grounded to insure that no radio frequency interference (RFI) is generated and that the system has the minimum possible susceptibility to external electrical influences.

The modem junction panels can also support a mixture of two interface standards. The modem junction panel uses the same mini-board concept as the direct connection panels, except that the modem mini-boards can carry only two ports per board because of the large size of the standard modem connectors.

On the modem junction panel, another 6801 demultiplexes the signals from the modem controller chip. This microcomputer is called the modem scanner chip. The port controller chip notifies the modem scanner chip (through the MCC) which input signals to examine, the expected state of these signals, and how to set the output latches.

The modem scanner chip scans the modem input signals, notices and debounces any changes and reports the changes to the MCC, which passes them on to the PCC.

ATP Software

The principal objective of the ATP software design is to provide a structure that will be easy to support and enhance. Most of the costs associated with a software product are incurred to fix problems in the original implementation and to add new features after the initial release. The ATP software was designed with a modular structure to facilitate this process.

The most desirable enhancement to HP 3000 terminal I/O has been a more flexible terminal-type facility. To achieve this goal in the ATP, the control of a port's characteristics is table-driven. Each of the current terminal types corresponds to one table. Changing one of these tables or adding a new table is a simple task.

In addition, the ATP software is designed to be fail-safe. When the MPE software detects a serious problem, all processing is stopped and a system failure is reported. When the ATP software detects a problem that affects only one port, only that port's operations are stopped. All other processing continues. An on-line facility allows the system manager or the HP customer engineer to record the state of that port for later analysis. The on-line facility also allows the port to be reset without requiring the system to be warm-started.

The ATP does not support some obsolete features of the earlier controllers, including half-duplex modems, tape mode, and several obsolete HP terminals.

Acknowledgments

A project of the size and scope of the ATP requires the skills of many people. Phil Ho contributed to many aspects of the design and wrote the PCC ROM code that is the heart of the product. Phil Curtis designed the SIB and acted as the hardware lead. Nancy Madison designed the AIB and many of the diagnostic tools. John Starita designed the junction panels, was responsible for the new connectors, and wrote the MCC and MSC ROM code. Steve Couch and Jon Hewitt designed the software structure and did most of the coding, with help from Cathy Smith and Curt Motola.

James E. Beetem

Jim Beetem received the SB degree in mechanical engineering from the Massachusetts Institute of Technology in 1962. He then served in the U.S. Air Force until 1967, attaining the rank of captain. Jim continued his education at Stanford University, earning an MBA degree in 1969 and doing post-graduate work in computer science. He joined HP in 1969 and has managed production control and information systems activities. Now a project manager for data communications, he is a member of the Association for Computing Machinery and has presented various papers on project management and terminal I/O at technical conferences. Jim was born in Lexington, Kentucky and now lives in San Jose, California. He enjoys white-water rafting, photography, camping, and backpacking. He is married and has two small children.

GUEST—A Signature Analysis Based Test System for ECL Logic

by Edward R. Holland and James L. Robertson

TESTING OF LOADED PRINTED CIRCUIT BOARD assemblies is vital in the production of any computer system. For effective testing of HP 3000 Series 64 boards, a special tester was designed. This system, the Gemini Universal ECL Signature Test system, or GUEST, is able to capitalize on the HP 5005 Signature Multimeter[1] and the built-in shift string organization of the flip-flops on the boards under test (see article, page 11).

Since all ECL networks in the Series 64 must be terminated in their characteristic impedances to suppress reflections, a tester running at slow clock speeds, as most board testers do, would not detect termination and other timing problems. A tester that functions at real-time clock rates was required and the GUEST system meets that need, being able to operate at clock rates up to 25 MHz.

In board testing, a test vector is a set of input states applied by a tester to a unit being tested. Generation of test vectors for a given unit under test (UUT) is often a difficult and time-consuming operation. In the GUEST system, vectors are generated algorithmically in real time by hardware. Therefore, preparation of test programs is reduced from weeks or months to a few hours. This has made the GUEST test system a useful tool throughout the prototype design process of the Series 64. By contrast, most test systems are usable only after the design is complete and all the test vectors have been generated, a point that is reached typically very late in a project.

Computer-Driven Test System

The GUEST test system is interfaced to the controlling HP 1000 Computer though the HP-IB (IEEE 488) interface bus (see Fig. 1). The primary functions of the computer system are to initiate testing and to guide the operator in probing of a defective UUT. One computer is able to handle up to six GUEST test stations, or a combination of GUEST test stations and DTS-70[2] test stations with all stations active at the same time.

The GUEST test system is interfaced to the UUT by a personality board as shown in Fig. 2. Building the personality board for ECL boards requires simply loading wire jumpers on a universal board. There is one set of jumpers for UUT input pins and another set of jumpers for pins that require terminations.

Hardware Test Vector Generation

The test vectors used in the GUEST system are generated as a serial bit stream by a 13-bit linear feedback shift register, which produces an 8191-bit-long pseudorandom sequence (see Fig. 2). This bit stream is shifted though a 588-stage shift register on the GUEST board and then via the personality board through the shift string of the UUT. Shifting continues for 1023 clock cycles. Then on the 1024th clock cycle, all of the shift register elements on both the GUEST board and the UUT are parallel-loaded under control of the diagnostic control unit shift line. The next 1023 clock cycles then cause the data loaded in the parallel operation to be shifted out of the combined registers past the GO/NOGO point. The test runs for a total of 8,379,393 (8191×1023) shifting clock cycles so that each of the inputs to the board is driven by all of the 8191 bits of the

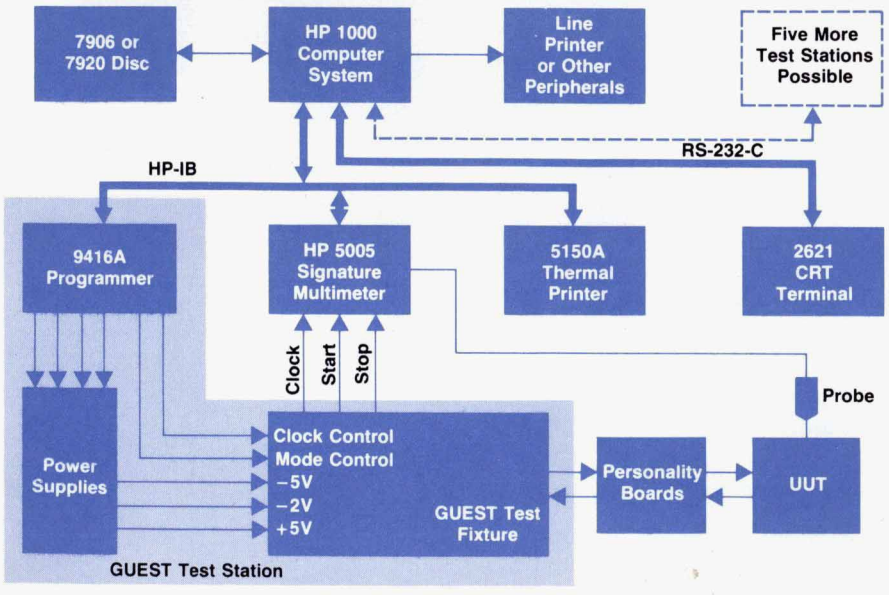


Fig. 1. *GUEST is interfaced to the controlling HP 1000 via the HP-IB (IEEE 488), and to the unit under test by a personality board.*

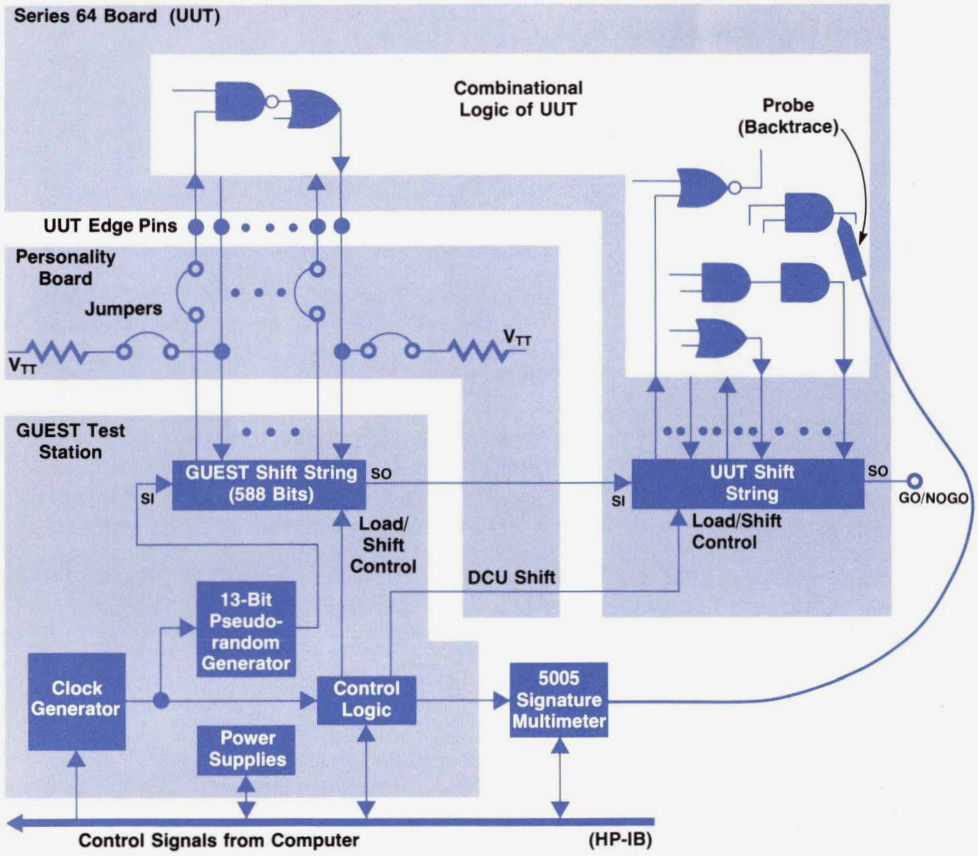


Fig. 2. *Test vectors are generated algorithmically by GUEST and loaded into the shift string of the HP 3000 Series 64 board being tested. The data is then shifted past the GO/NOGO point, where the 5005 Signature Multimeter generates a signature that tells whether or not a problem exists on the board.*

pseudorandom sequence and each output of every shift string flip-flop on the board is also driven by all 8191 bits.

Go/No-Go Testing

As the data stored in the shift registers on the 1024th clock cycle is shifted out past the GO/NOGO test point, the HP 5005 Signature Multimeter computes the signature of the data stream and the computer checks this signature against the expected signature stored for the board type being tested. Since both the GUEST and the UUT shift registers are shifted past this test point, any incorrect value from any part of the UUT causes an incorrect GO/NOGO signature. If the board passes the GO/NOGO test it is ready for installation in a system. If the board fails this test, additional testing is done on the GUEST tester.

The control logic of the GUEST tester can limit the signature analyzer to only the last N data outputs of the shift string, where N is specified by the computer. When N is the maximum value, the signature analyzer clock is always enabled and the signature depends on all bits in the shifted outputs. As N is decreased, fewer and fewer outputs enter into the signature. At some value N=n the signature will be incorrect, while at N=n−1 the signature will be found to equal the expected value. Since there is a one-for-one mapping of the value of N and a point in the overall shift string, it is possible to pinpoint the fault to an I/O pin or an input to a shift string bit on the UUT. The computer is programmed to use a binary search algorithm and takes ten signatures or less to find the correct point in the shift string to begin further backtracing.

Backtracing

While backtracing faults to the component level, the signature at each node is dependent only on combinational functions of the pseudorandom test vectors applied to the UUT input pins and shifted into the internal shift register states of the UUT. The signatures are measured by moving the signature analyzer probe to each node as guided by the computer.

Backtracing starts at the node determined by the GO/NOGO binary search. When a node is found that has an incorrect signature but all inputs affecting that node have correct signatures, the faulty node has been isolated.

Feedback Loops

To make effective use of signature analysis, all feedback loops should be broken. If a feedback loop is not broken, a bad signature anywhere in the loop causes all other nodes in the loop to have bad signatures, and fault isolation to the component level is impossible. Feedback loops are broken during backtracing by the internal UUT shift registers. This is accomplished by clocking the signature analyzer when the outputs of each shift register are dependent only on its serial input and not on its parallel data inputs. Since bad signatures cannot propagate through these shift registers, any feedback path from an output to a parallel data input is broken.

Clock and Shift Control Faults

Another complication in the fault isolation process is that of diagnosing faults in the clock and shift control circuits of the UUT. Since data in these areas is normally synchronous

Designing for Testability with GUEST

by Karen L. Meinert

The HP 3000 Series 64 and the GUEST system were designed at the same time to work together. It was discovered that certain design issues had to be considered when designing a Series 64 printed circuit assembly (PCA) if maximum GUEST testability was to be achieved.

High Fan-in

Fan-in is defined as the number of inputs to a circuit that is needed to produce an output. Because GUEST outputs only 8191 predefined test vectors, a fan-in of 14 or greater makes it impossible to sequence through all combinations (2^{14} is greater than 8191). If the fan-in is less than 10 it is safe to assume that all combinations will appear in the test sequence. Modifications to the board or the personality board can be made to assure that all combinations are tried when the fan-in is between 11 and 13. There are also several signals (mostly high, mostly low, pulse high, pulse low) available on the personality board; these can be used to drive inputs to increase a circuit's effective test vectors.

It is important that the 13 inputs to a high-fan-in circuit be driven by contiguous bits from GUEST. The bits in the shift string and on the edge connectors of GUEST are just shifted versions of the 13 bits that guarantee the 8191 test vectors. If 13 bits are selected at random, there is no guarantee that all 8191 combinations will appear.

Feedback Loops and Shift Strings

For a PCA to be GUEST-testable it is not necessary that it have shift strings (see article, page 11). The shift strings are an effective means to break up feedback loops and initialize memory elements. If no feedback loops exist on a board and memory can be initialized, there is no need for shift strings.

Nonstandard Clocking

Both the GUEST shift register and the signature analyzer are clocked on the standard system clock. If other phases or edges of the clock are used, testing problems may occur. Feedback loops are not necessarily broken if shift string registers are clocked on phases other than that used by GUEST. Also, there could be problems in getting information between the PCA and the GUEST system in time if other phases are used. In general, clocks other than the standard system clock should be avoided when designing PCAs to be tested by GUEST.

Non-ECL Circuitry

GUEST was designed to be an ECL circuit tester. To test circuits other than ECL, translator packs are used on special personality boards that adapt each tested board to GUEST. In addition, three-state buses should not be allowed to float to the high-impedance state or unstable signatures will result. The solution is to have GUEST drive the bus whenever the board is not. This is accomplished by bringing the three-state bus enable signal to an unused I/O pin on the PCA. This signal is inverted and used on the personality board to enable the translator direction (ECL to TTL or vice versa).

TTL signals can be probed by the HP 5005 Signature Multimeter, since this analyzer is capable of changing its input voltage threshold to allow for non-ECL signals. This can be done either manually or under automatic control through the HP-IB interface.

RAMs

To test a RAM completely each location must be addressed five times: once each to write one, write zero, read one, read zero, and deselect. Since GUEST outputs a total of only 8191 test vectors, RAMs larger than 1024 bits deep cannot be tested completely. To obtain known signatures on the outputs of RAMs, any location that is read must first have been written. If a location is read that has not been written, the read data is whatever was in the RAM at power-up. This is random data and therefore has a random signature. A signal called RAMINIT (RAM INITialize) is generated by GUEST to assure that all read locations are initialized. GUEST generates a pseudorandom number sequence with a length equal to half of a signature analyzer cycle. This sequence is repeated during the second half of the cycle. RAMINIT is high for the first half-cycle and low for the second half-cycle during the time that the RAMs can be read or written into. RAMINIT is used in the write enable circuitry to force a write when high and to enable reads and writes when low. In this way every location that is addressed in the first half is written with known data. When these addresses repeat themselves in the second half (they are part of the repeated pseudorandom sequence), where reads may occur the locations have been initialized, and known, repeatable signatures will occur.

Karen L. Meinert

Karen Meinert is a native of the State of Iowa and attended Iowa State University, earning the BSEE degree in 1979. She joined HP that same year and is a design engineer for the main memory subsystem used in the HP 3000 Series 64. Karen lives in Santa Clara, California. When she isn't busy exercising her golden retriever, she enjoys sports, scuba diving, camping, and woodworking.

with the signature analyzer clock, only SA0 and SA1 (stuck at 0 and stuck at 1) signatures would be found and normal backtracing would be impossible. To isolate this type of fault, the first backtrace signature measured after the GO/NOGO test fails is at the GO/NOGO test point. A correct signature here indicates that the UUT clock and shift control circuits are functioning. If the test fails, the shift register serial outputs are probed using a binary search algorithm until a good signature is found and the following device's serial output is bad. Clock and shift control inputs to this earliest failing shift register are probed while pseudorandom data is applied to the clock and shift inputs of the UUT. If no bad inputs are found, the shift register is suspected to be bad. If a clock or shift control input fails, backtracing starts at that point, with pseudorandom data still applied to the clock and shift inputs of the UUT, and continues until the failing node is isolated.

Test Report

When the faulty node has been isolated, three additional measurements are made on the suspected node. They are: +

peak voltage, − peak voltage, and resistance to ground. A test report with information pertinent to the failing node is then printed. The failure descriptions can be: NODE MOVES, NODE STUCK, NODE UNSTABLE, or OPEN TRACE. The information on the test report can be used to further diagnose the fault (shorted nodes, open terminations, etc.).

Two assumptions made so far are that the probe has made good contact with the node being tested, and that the operator probes the correct point when prompted. This is not always the case, so provisions have been made for recovery from probing contact errors. To minimize operator misprobes, all pertinent IC pins are probed in sorted ascending order before moving on to the next IC. Thus, pin-to-pin and IC-to-IC movement is minimized. Each node is probed at least twice in different locations to diagnose misprobes or open traces. If a misprobe is discovered before it is diagnosed by the computer, the operator can ask to reprobe the point where the error was made.

Edge connector pins, resistor pins, or any other point that is difficult to locate or probe can be declared inaccessible. During backtrace, the operator will not be prompted to probe these points. When the test report is printed, any inaccessible point that could have caused the fault is listed as not checked.

Creating the Test File

Before using GUEST, a UUT test file must be created by TESTAID.[3] To use TESTAID, topology information (a description of the types and interconnections of all integrated circuits on the UUT) must be supplied by the test programmer. This data is available from the computer routing and placement program used in the design of Series 64 boards, and is converted by a utility program to the format required by TESTAID. No other input data is required for TESTAID simulation because all test vectors are generated by the GUEST hardware.

After the preliminary test file has been created by TESTAID, UUT-dependent test setup parameters need to be added (clock period, reference voltages, special node definitions, etc.). Signatures can then be added as measured from a known good UUT. The addition of all GO/NOGO signatures is automatic and requires approximately two hours run time. Backtrace signatures are added by manually probing each internal node as guided sequentially by the computer. This process also requires approximately two hours.

Once generated, the UUT test file should be verified for accuracy of fault isolation and fault coverage. Fault isolation accuracy can be verified by inserting a few known faults in critical areas of the UUT and determining whether they are diagnosed correctly. Fault coverage is verified by measuring the GO/NOGO signature repeatedly while probing internal nodes, as guided by the computer, with a special probe that forces each node both high and low. If the GO/NOGO test passes, the stuck at 1 or stuck at 0 node fault is marked as not detected. When probing is complete, the fault detection percentage is computed and saved in the UUT test file. Typical fault coverage is in the order of 99% for ECL boards.

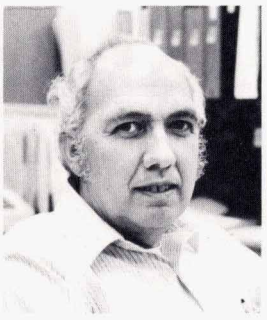
The final step to be done when creating a test file is documentation. By entering the DOCALL command, complete documentation of the UUT test file (all node drivers, receivers, terminations, special setup, signatures, etc.) can be listed on the line printer.

Acknowledgments

The authors would like to acknowledge the creative work of Bob Horst and Rick Amerson for proposing and designing the early GUEST test system. Credit should go to Jim Tseng, Harish Joshi, and Bob Cook for the final translation of the hardware into a reliable and highly usable system. Mike Phillips deserves much credit for working out bugs in the process of personality board design and a lot of helpful feedback in the programming process. John Shing and Gary Gitzen deserve thanks for their helpfulness in making available modified 5005A Signature Multimeters for use in the GUEST system.

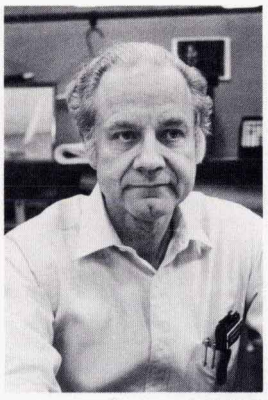
References

1. R.A. Frohwerk, "Signature Analysis: A New Digital Field Service Method," Hewlett-Packard Journal, May 1977.
2. P.S. Stone and J.F. McDermid, "Circuit Board Testing: Cost-Effective Production Test and Troubleshooting," Hewlett-Packard Journal, March 1979.
3. K.P. Parker, "Software Simulator Speeds Digital Board Test Generation," Hewlett-Packard Journal, March 1979.



James L. Robertson

Jim Robertson joined HP in 1958 with previous experience as an avionics test engineer. He has been a standards lab manager and a pretest supervisor, and now works in electronic tooling. Jim is a native of Los Angeles, California and received the AS degree in electrical engineering from Foothill College, California in 1968. He is married, has two sons and two married daughters, and enjoys camping, hiking, and travel. He lives in Los Altos, California.



Edward R. Holland

Ed Holland was born in Detroit, Michigan and attended Michigan State University where he earned the BS degree in electrical engineering. After service in the U.S. Naval Reserve and work on radar and test system design, Ed continued his education at Stanford University, earning an MS degree in electrical engineering in 1960. He joined HP in 1961 and has contributed to a number of products that include the 3440, 3460, and 3462 Voltmeters, the 2114, 2115, and 2116 Computers, and HP 3000 Computers. Currently, Ed is project manager for the Series 64 CPU, cache memory, and GUEST tester hardware. He is the author of a paper on computer I/O and peripherals and is named inventor on a patent related to the basic A-to-D conversion concept used in the 3460 DVM. Ed is a member of the IEEE and the Planetary Society. He lives in Palo Alto, California, is married, and has two children. He enjoys backpacking, bicycling and sailing and works at promoting space exploration and the search for extraterrestial intelligence.

Packaging the HP 3000 Series 64

by Manmohan Kohli and Bennie E. Helmso

THE HP 3000 SERIES 64 is the new highest-performance member of the HP 3000 Computer family. It has been designed to retain certain visual aspects of current HP systems and peripherals for visual compatibility, but also incorporates new directions for future products. The Series 64 is intended to be used in an EDP environment and a cost-effective package design has been applied to optimize reliability and serviceability.

New tubular welded frames, preassembled formed cardcages, bus bars and modular junction panels were developed as part of this cost-effective package. New two-piece press-fit connectors for printed circuit boards and the backplane and an effective cooling scheme were developed for higher reliability.

The cabinet has two open frames that are bolted together in the front and rear. These provide complete access to various assemblies including backplanes for servicing (Fig. 1). A 36-slot cardcage contains the CPU, memory and I/O adapter cards and a second 24-slot cardcage contains I/O device cards. The power system is designed to support the maximum configuration.

Separate cooling systems are used to cool printed circuit boards and power supplies. Cooling air is drawn from the rear and is then forced through the cardcages for proper cooling of the printed circuit boards. Bus bars are used between backplanes and power supplies to minimize cable harnesses and to facilitate replacement of power supplies. I/O junction panels are modular and provide maximum flexibility for various system configurations.

The system cabinet is designed to meet HP Class C environmental specifications. This requires the system to operate at extreme temperature and humidity, between sea level and high altitude, and exposed to shock and vibration.

Cooling System

The cooling system is designed to meet two major objectives under worst-case conditions:

- Transistor junction temperatures not to exceed 85°C for higher reliability
- Air temperature rise not to exceed 10°C for adequate noise margin.

The CPU boards contain mostly ECL chips with an average heat dissipation of 0.5 watt/chip and 45 watts/board. However, the total heat dissipation from the maximum-configured CPU, memory, and I/O adapter is 1300 watts. These parameters presented some challenges in designing the cooling system. To meet the above objectives, five 16.5-cm-diameter tubeaxial fans are used to deliver high-velocity air at 2.3 m/s for cooling the ECL chips.

Cardcages are cooled by taking air in from the rear and then forcing it up through the cardcage. A plenum is used between the fans and the cardcage to help distribute air across the cardcage more evenly. The fans are inside the plenum, which is lined with acoustic foam to reduce noise.

Two sets of overtemperature switches are mounted on the two cardcages. When the temperature reaches the low set point (40°C), both audio and visual alerts to the operator are initiated. The system is shut down if the cardcage temperature reaches the high set point (50°C).

Electromagnetic Compatibility Design

To decrease emission of electromagnetic energy and susceptibility to it, the logic ground is isolated from the frame ground. This is achieved by electrically isolating all printed circuit boards and backplanes from the sheet-metal cardcages. Radiated emissions are minimized by providing and connecting ground planes in the printed circuit boards

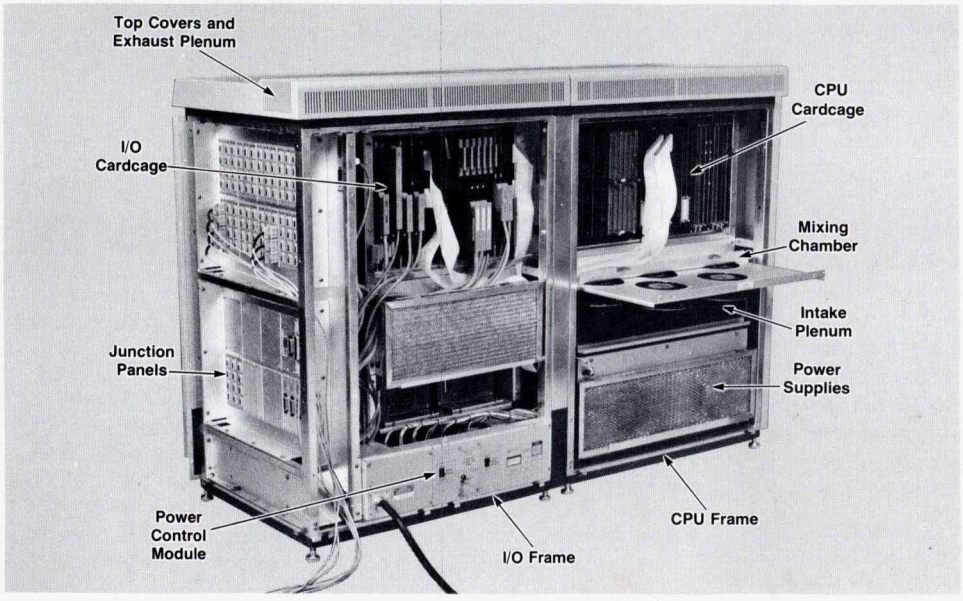


Fig. 1. *HP 3000 Series 64 cabinet design.*

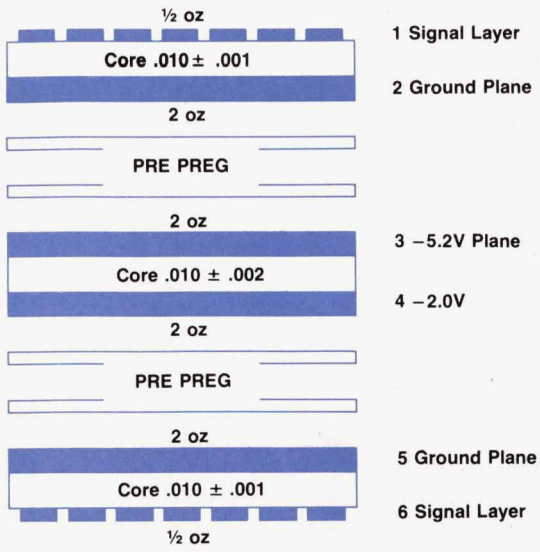


Fig. 2. *Typical HP 3000 Series 64 printed circuit board construction. Dimensions are in inches.*

and backplanes. Susceptibility is minimized by grounding the frames, exterior panels, and internal assemblies to each other through plated interfaces. Beryllium-copper clips mounted on all exterior panels make contact with the plated frame members to provide grounding to reduce susceptibility to electrostatic discharge. Three isolation transformers, one for each phase, provide good common-mode noise rejection, lessen susceptibility to electromagnetic interference, and insure that ac line leakage current is well within safety requirements.

Boards and Connections

Designing and making large, high-density, controlled-impedance-interconnect printed circuit boards presented some new challenges. The design parameters that were traded off to arrive at final design rules were board size, minimum trace width and spacing, IC package density, trace routability, and raw board testability. What resulted were boards approximately 36 cm square with an average of 120 IC packages per board. Minimum trace width and spacing was set at 0.18 mm (0.007 in). A 2.5-mm (0.100-in) grid system for component holes was employed to facilitate testing of unloaded boards before assembly.

For density and routability considerations only two signal layers are used. These are on the outer layers, leaving the inner layers for power and ground distribution. The typical board construction is shown in Fig. 2. It should be noted that the signal lines are in the standard microstrip configuration with nominal characteristic impedance of approximately 68 ohms.

For the interconnect design of the large number of individual circuits, manual design was ruled out, largely because of time constraints. A sophisticated design automation approach was selected to provide automated placement of components and routing of traces after initial data entry of schematic and mechanical information.

Interconnecting all the boards in a system this large also presented some special problems. Board-to-board interconnection in the CPU and memory sections requires just under 10,000 pins. A pin-and-socket (two-piece) connector system was selected. It was felt that the conventional card-edge (one-piece) system had some problems (such as board edge contamination from soldering and handling, gold plating cost, thickness and quality control) which the two-piece system minimizes. For example, all connector plating is done by the connector manufacturer, and all gold plating is eliminated from the printed circuit boards.

The board-to-board interconnection is accomplished by means of a backplane assembly. In addition to providing for the interconnection of signal lines, the backplane also distributes dc power to the individual circuit boards. The construction details of the backplane are as shown in Fig. 3. There are four signal layers, two ground planes, and four power planes. The outer layers are microstrip transmission lines and the inner signal layers (layer 3 and 8) are conventional striplines. The power and ground planes are duplicated to aid in power distribution. Because of the number

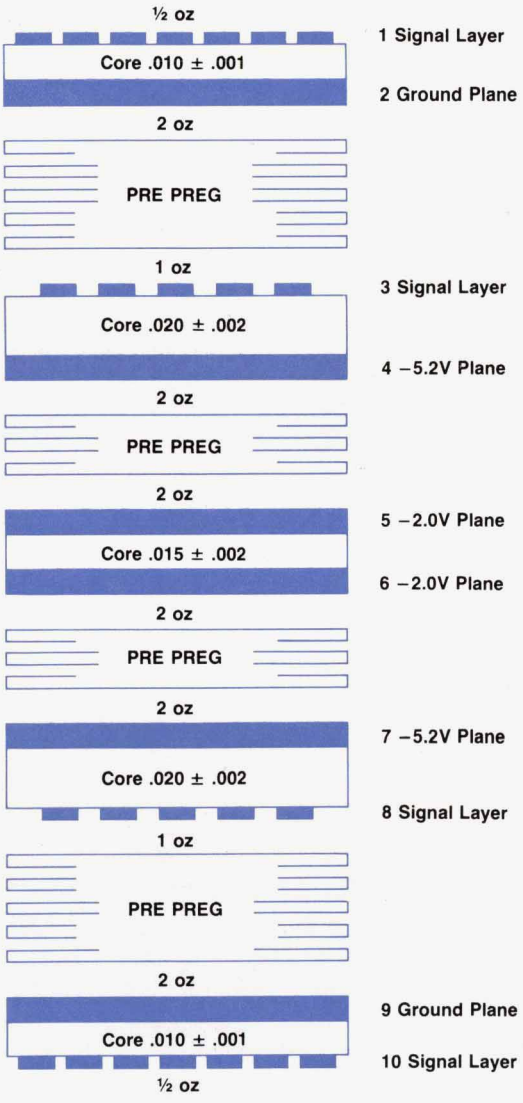


Fig. 3. *HP 3000 Series 64 backplane construction. Dimensions are in inches.*

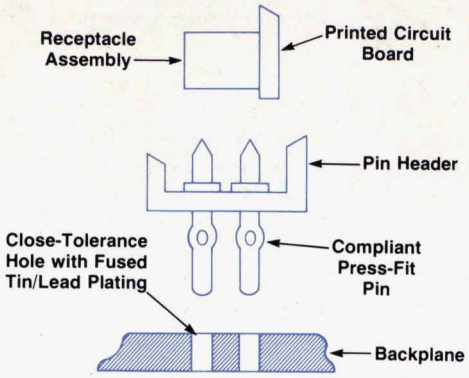


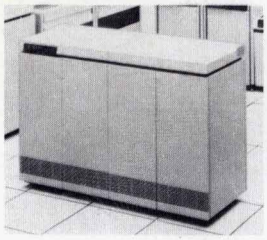
Fig. 4. *Board-to-backplane connection method is gas-tight and eliminates soldering.*

of layers and the large amount of copper in the inner planes, conventional connectors soldered into the backplane were not practical. The problems of thermal stresses to the backplane assembly during soldering and inability to remove or replace connectors because of the heat-sink effect of the inner planes led to the selection of press-fit connections between the connectors and the backplane. An interference fit between the connector pins and holes in the backplane produces a gas-tight connection and eliminates the need for soldering the pins to the backplane. The board-to-backplane connection is shown in Fig. 4.

Acknowledgments

The development of the new cabinet and packaging required contributions from many people. Contributions were made by Bob Cook on packaging and serviceability, Steve Spelman on industrial design and plastic parts, George Canfield on power distribution, and Katie Torres for most of the drawings and documentation. Jim Brannan was responsible for most of the product qualification testing, Barb Gee provided the manufacturing support and Toby Huff the service engineering support. Guidance and support was provided by Peter Rosenbladt, R&D section manager.

PRODUCT INFORMATION



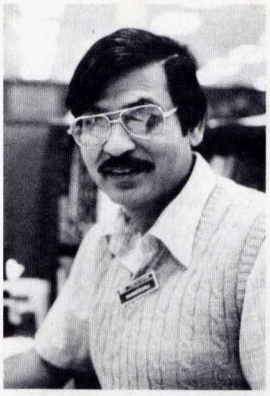
HP 3000 Series 64 Computer System

MANUFACTURING DIVISION:
Computer Systems Division
19447 Pruneridge Avenue
Cupertino, California 95014 U.S.A.

SPECIFICATIONS: HP Publication No. 5953-0656

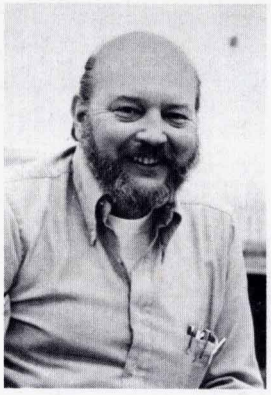
PRICES IN U.S.A.: 32460A HP 3000 Series 64 System Processing Unit, $164,700. 30143A I/O Adapter Module, $10,000. 30142A 1M-Byte Memory Module, $16,000. Disc Drives, terminals, and other peripherals additional.

Manmohan Kohli



Manny Kohli is a native of Punjab, India and received the BSME degree from Punjab University in 1964. He earned the MSME degree at the University of California at Berkeley in 1967 and then did mechanical design of impact printers and high-speed mechanisms and stress analysis of aircraft interiors before joining HP in 1974. Manny designed the cabinet for the HP 3000 Series 33/44, worked on the thermal printer used in the HP-91 and HP-97 Calculators, and is project manager for the industrial/product design of the HP 3000 Series 64. Manny enjoys working on home projects, tinkering with cars, camping, and reading. He is married, has two children, and lives in San Jose, California.

Bennie E. Helmso



Ben Helmso joined HP in 1960 after receiving the BSEE degree from the University of California at Berkeley. During his more than twenty years at HP he has worked on oscilloscope, microwave sweeper, and wristwatch product designs and managed manufacturing engineering and quality assurance groups. At present, Ben is doing RF signal generator product design. He is named inventor on a patent related to the HP-01 watchband. Ben was born in Los Angeles, California and served two years in the U.S. Navy before studies for his BSEE degree. He is married, has two children, and lives in Spokane, Washington. His interests include camping, boating, hunting, and fishing.

Hewlett-Packard Company, 3000 Hanover Street, Palo Alto, California 94304

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

HEWLETT-PACKARD JOURNAL

MARCH 1982 Volume 33 • Number 3
Technical Information from the Laboratories of Hewlett-Packard Company
Hewlett-Packard Company, 3000 Hanover Street
Palo Alto, California 94304 U.S.A.
Hewlett-Packard Central Mailing Department
Van Heuven Goedhartlaan 121
1181 KK Amstelveen, The Netherlands
Yokogawa-Hewlett-Packard Ltd., Suginami-Ku Tokyo 168 Japan
Hewlett-Packard (Canada) Ltd.
6877 Goreway Drive, Mississauga, Ontario L4V 1M8 Canada