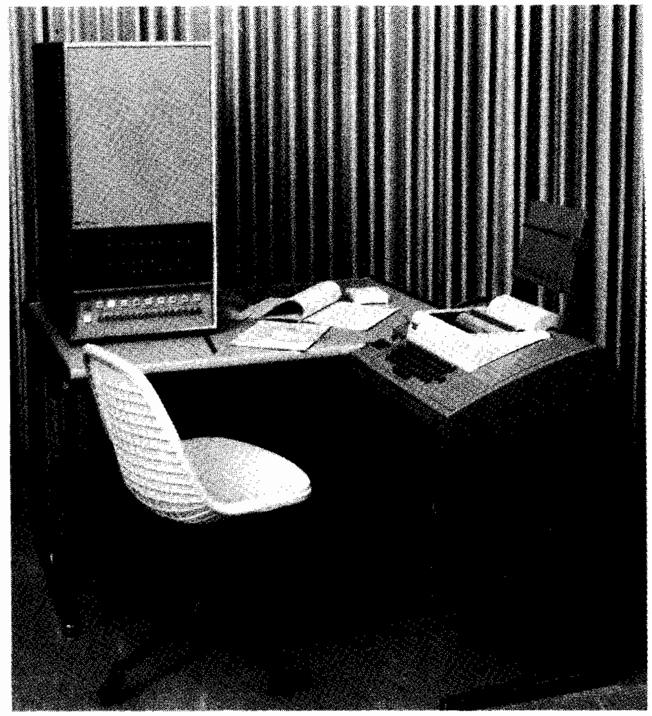
HEWLETT-PACKARD JOURNAL

Cover: LOGIC MODULES FROM A NEW INTEGRATED-CIRCUIT COMPUTER FOR INSTRUMENTATION SYSTEMS: see page 2

MARCH 1967



A Computer for Instrumentation Systems

Problems of interconnection, programming, and environment arise in the design of systems containing both computers and instruments. They are solved in advance by this new integrated-circuit computer.

DIGITAL COMPUTERS AND LABORATORY INSTRUMENTS ordinarily make strange bedfellows. They not only have difficulty making contact with each other and working together, but most computers don't even feel comfortable in the unsympathetic everyday world in which instruments live.

In recent years, however, a growing need has arisen for a computer that can work efficiently in instrumentation systems. These systems are becoming more numerous and more complex. Many of them would not be feasible without the data processing and flexible control that a computer can provide.*

Because the need for an instrumentation computer was not being filled by others, -*hp*- decided to go ahead and build one, feeling that a computer to work with instruments could probably be built best by an instrumentation manufacturer. Our new computer is an integrated-circuit machine which has the computing power and special capabilities needed to work well in instrumen-

* See p. 11 for descriptions of some of these systems.

tation systems. It does away with three discouraging problems that have plagued systems designers in the past:

—the interface problem, or how to connect the instruments with the computer so that efficient data communication can take place

—the software problem, or how to program the computer efficiently, using programming systems that were not designed with instruments in mind

—the environmental problem, or how to keep the computer working in the unfriendly environments in which instruments operate.

A closer look at these problems and at how the instrumentation computer eliminates them will be the best introduction to the new machine and what it can do for the systems designer.

Convenient Computer/Instrument Interfaces . . .

Laboratory instruments are unfamiliar input/output devices for most computers. Seldom is there any ready-made, convenient means for connecting an instrument to a computer; it usually takes an expensive, custom-designed interface. This is true even of some common computer input/output devices. For example, if a computer is not specifically designed to work with a magnetic tape unit, and many are not, it may take several thousand dollars worth of engineering talent and equipment, plus several months, to produce an interface that will allow the computer and a tape unit to work together. If a number of instruments have to be interfaced, or if the system configuration has to be changed, the user's problems are multiplied many times, of course.

. . . Provided by Flexible Input/Output System

The new instrumentation computer is different; it has an extremely flexible input/output system which greatly simplifies interfacing to a large number of instruments. Within the main frame of the basic computer is a box which accepts up to 16 plug-in interface cards (Fig. 1). Most instruments and input/output devices can be interfaced to the computer with one card each; a few require two cards. Accessory modules raise the maximum number of interface cards to 48.

Cover

To get this striking effect for our March cover, photographer Hal Smith placed his lights underneath one of the plug-in logic boards from the new *-hp-* Model 2116A Instrumentation Computer. Component density on these boards is quite high for two-sided (not multi-layer) boards — about 200 integrated-circuit flatpacks per square foot. Each flatpack is the equivalent of 50 to 100 discrete components.

Eventually, there will be plug-in interface cards for all *-hp-* instruments which either provide a digital output or can be programmed.* At present, cards have been designed for more than 20 instruments, including counters, nuclear scalers, electronic thermometers, digital voltmeters, ac/ohms converters, data amplifiers, and input scanners. Interface cards have also been designed for most kinds of input/output devices, such as magnetic tape recorders, teletypewriters, paper tape readers and punches, and dataphones. Interfaces for printers and card readers and punches are under development.

Efficient Programming . . .

Programming systems, or 'software' in computer jargon, which have not been designed to work with instruments can be made to do so only reluctantly and inefficiently. The inability of previously available software to cope with instruments is never more obvious than when the computer and its instruments have to exchange data or control signals.

* Instruments which provide analog outputs can be interfaced with the computer via a digital voltmeter or an analog-to-digital converter. Instruments which require analog programming can be interfaced via a digital-to-analog converter.

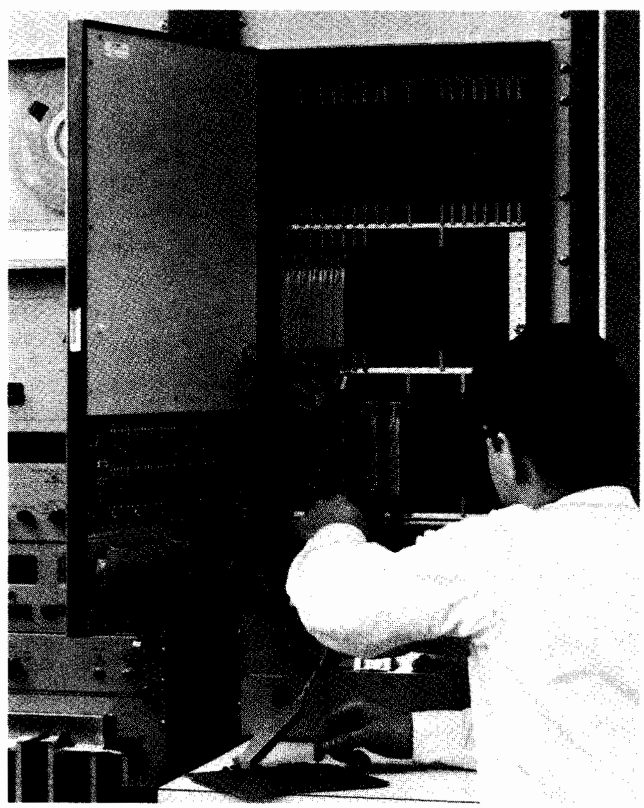


Fig. 1. *Operator of new -hp- 2116A Computer changes one of the plug-in interface cards which make it easy to connect the computer with laboratory instruments and other input/output devices. This and other features of the new machine greatly simplify the task of augmenting an instrumentation system with a general-purpose computer.*

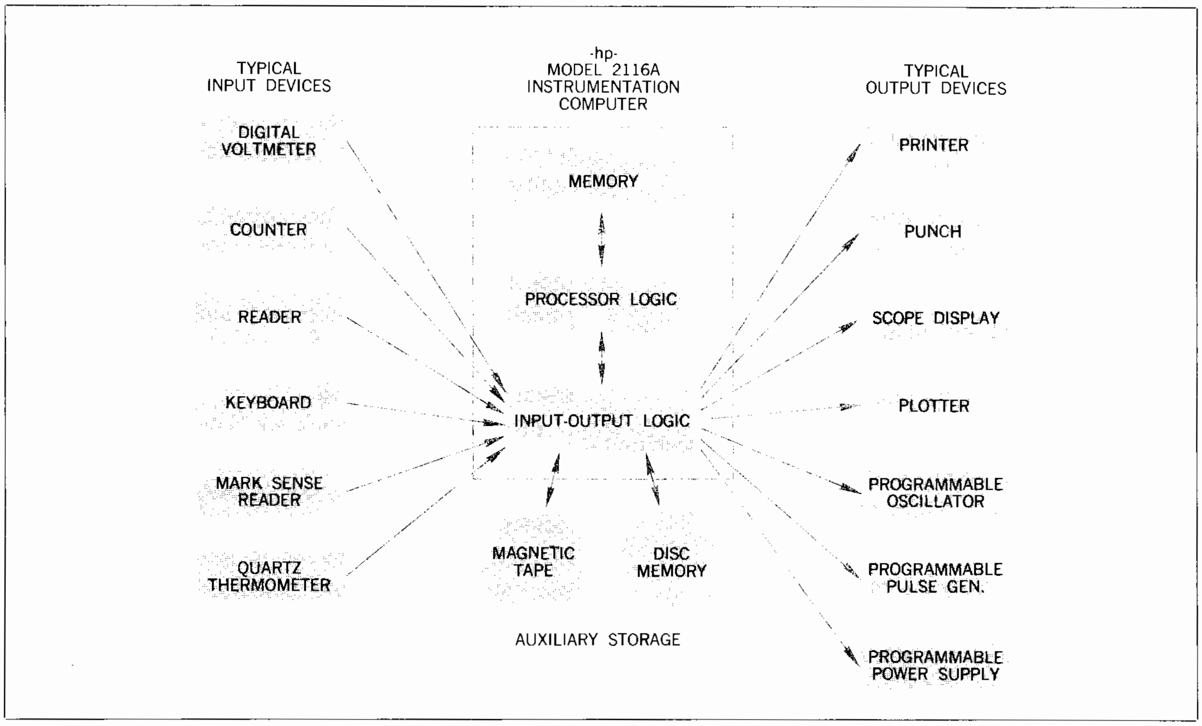


Fig. 2. *Some of the input/output devices that can be connected to –hp– 2116A Computer by means of plug-in cards. More are listed in text.*

Typically, computers in instrumentation systems process data in real time — as soon as it is acquired — rather than at some later time. The computer is 'on line' continually, processing data according to the program prepared by the user. When an instrument is ready to transmit data to the computer, the computer must interrupt its regular program, determine which instrument is requesting service, and load the new data into the proper place in the memory. Although this seems simple enough, it is not. Unless the computer has been designed to work with instruments, it takes a surprisingly long time for the computer to scan all of the instruments and find the one which has the data to transmit. This time can usually be put to better use. What's more, someone — a programmer — has to write the programs which tell the computer how to talk to each instrument. Data formats and codes often differ from instrument to instrument, so programs for different instruments may have to be quite different. In fact, the software problem may outweigh the interface problem when a system is being assembled or changed.

. . . Provided by Multi-Channel Interrupt System . . .

To allow all of its instruments to be serviced rapidly and in real time, the instrumentation computer has a multi-channel priority interrupt system. 'Multi-channel' is the key word here. The priority of each instrument is determined by the slot occupied by its interface card, and each slot, or channel, is assigned a different location in the computer's memory. When one or more instruments signal the computer that they are ready to transmit data, the computer does not have to scan the instruments in order of their priorities to find out which one has produced the interrupt. While other machines may require a millisecond or more just to determine which instrument is requesting service, the instrumentation computer locates the highest-priority interrupting instrument and its service subroutine in a few microseconds, transfers the data, and goes on to the next-lower-priority instrument. When all instruments requesting service have been serviced, the computer returns to its regular program, having spent a minimum of time away from it. Alternatively, interrupt signals from one or more instruments can be inhibited by the computer, or the computer can signal an instrument to make a measurement.

For those special occasions when even a few microseconds can't be spared, the new computer will soon have direct memory data channels. One of these direct channels can be assigned under program control to any input/output channel, permitting data to be transferred directly to memory without going through the normal channels (i.e., through one of the two accumulator registers). The maximum data transfer rate for a direct memory channel is 600,000 16-bit words per second, whereas the maximum rate for normal channels is about 60,000 words per second.

. . . and Compatible Software

Software for the instrumentation computer has been designed to make full use of the flexibility of the input/output (I/O) hardware. A modular control system allows programs to be written without concern for the specific operating requirements of individual I/O devices, and a 'software configurator' is furnished which allows the user to modify his control system easily to fit different I/O hardware configurations. Systems can be upgraded (say by switching from a low-speed to a high-speed tape punch) without changing the program. In other words, programming of the instrumentation computer is very nearly independent of the I/O devices used.

Programs for the new computer can be written in either of two programming languages, FORTRAN or assembly language. The computer comes equipped with a FORTRAN compiler which operates in the basic memory of 4096 16-bit words. The compiler is a program which converts any program written in ASA Basic FORTRAN — a universally accepted programming language — to the binary machine language that the computer understands. Actually, the new computer's FORTRAN is an augmented version of ASA Basic FORTRAN, allowing more flexibility in programming.

Assembly language, the other programming system, is a symbolic language which is closely related to the computer's hardware. The assembler, a program which converts assembly language programs to binary machine language, also operates in the basic 4096-word core memory.

The example below illustrates the two programming languages.

An interesting fact not shown in the example is that the instrumentation computer's FORTRAN compiler can produce an assembly-language listing of a FORTRAN program at the same time that it translates the program to machine language. This unusual capability makes it easy for a programmer to write part of a program in FORTRAN and part in assembly language, and then combine the two parts. It is often most efficient to write part of a program for the specially-designed instrumentation computer in assembly language, which is more closely related to the design of the machine than is FORTRAN, a machine-independent language.

The new computer's modular basic control system includes the following software modules:

—a relocating loader, which loads, combines, and initiates the execution of programs or parts of programs prepared by the FORTRAN compiler and the assembler.

—an I/O control, a general I/O device control program

—I/O drivers, which control specific I/O devices.

Programming the Instrumentation Computer: An example

PROBLEM: Read magnitude and phase, then convert to rectangular coordinates.

Programmer writes this or this

FORTRAN program: Written by programmer and punched on paper tape for computer input.

```
READ (10) R
READ (11) THETA
Y=R*SIN (THETA)
X=R*COS (THETA)
```

FORTRAN compiler: Supplied with computer on paper tape. Loaded into memory, it causes computer to convert input program written in FORTRAN to machine language.

Assembly language program: Written by programmer and punched on paper tape for computer input.

```
LIA    10
JSB    CONV
STA    R
LIA    11
JSB    CONV
STA    THETA
JSB    SIN
LDB    R
JSB    MPY
STA    Y
LDA    THETA
JSB    COS
LDB    R
JSB    MPY
STA    X
```

Assembler: Supplied with computer on paper tape. Loaded into memory, it causes computer to convert input program written in assembly language to machine language.

```
1 0 0 0 0 1 1 1 0 1 0 0 1 0 0 0
0 0 0 1 1 1 0 0 0 0 0 1 0 0 0 0
0 1 1 1 0 1 0 0 1 0 0 0 0 0 0 0
    •     •     •
    •     •     •
    •     •     •
```

Machine language program: Punched on paper tape by computer operating under control of assembler or FORTRAN compiler. Loaded into memory to cause computer to operate on data to solve problem.

Two other software packages provided with the basic control system are:

—a 'prepare control system' program, used to combine or modify the elements of the basic control system.

—debugging routines.

Other software elements which have been designed for the instrumentation computer include hardware diagnostic programs for troubleshooting, a symbolic editor program which makes it easy to edit or change any program, and a library of subroutines for various mathematical operations. All of this software is provided on punched paper tape. All of it works in the computer's basic memory of 4096 16-bit words, although the memory can be expanded to 16,384 words if desired and the software will take full advantage of the additional memory.

Environment Not a Problem

Unfriendly environments in which instruments must commonly work are no problem for the instrumentation computer. Unlike most computers, it will not balk at temperatures ranging from 0°C to 55°C, line voltage varying ±10%, line frequency varying between 50 and 70 Hz, and humidity up to 95%. The computer needs no air conditioning in order to operate in such environments. It can also function normally under conditions of electromagnetic interference and vibration that would seriously hamper most computers.

What It Can Do

Some general categories of problems which can be solved easily by the new computer and an appropriate combination of standard -*hp*- instruments are:

1. **Data reduction problems,** in which a large number of data points are taken, but only a few answers are required. A computer can perform mathematical operations, such as integration and convolution, thereby greatly reducing the amount of data requiring human analysis later on. Time savings are possible too, because the computer can be on line, processing data while measurements are being made.

2. **Data transformation and report preparation.** Frequently the data produced by an instrument is not in the best form for the user. The computer can perform operations such as curve-fitting and linearization, and then present the data graphically, or in other required forms.

3. **Problems which require fast results** for feedback during a test. When feedback is available, the amount of data taken can often be reduced substantially by not taking redundant data. The computer can decide on the basis of previous data what new measurements should be made, and then direct the proper instruments to make the measurements.

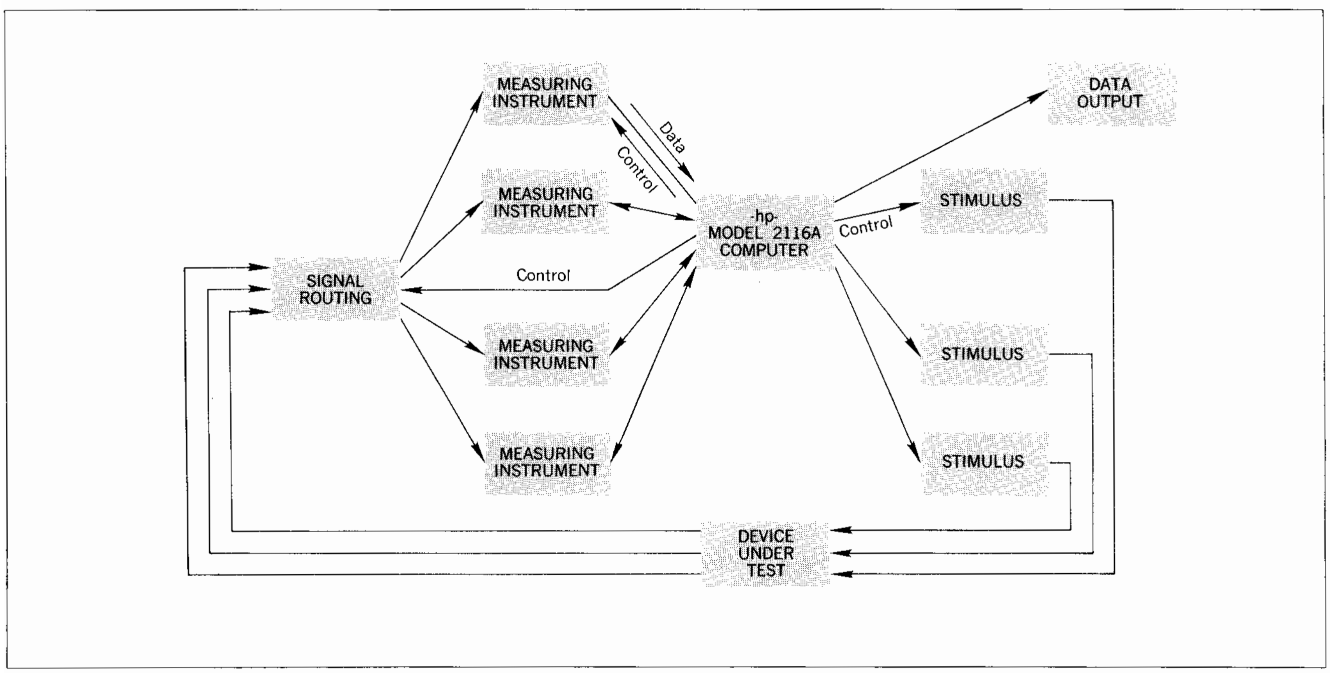


Fig. 3. *Typical application for –hp– 2116A Computer is controlling detailed tests of transistors, printed circuit cards, or other devices, then presenting processed data in any required form.*

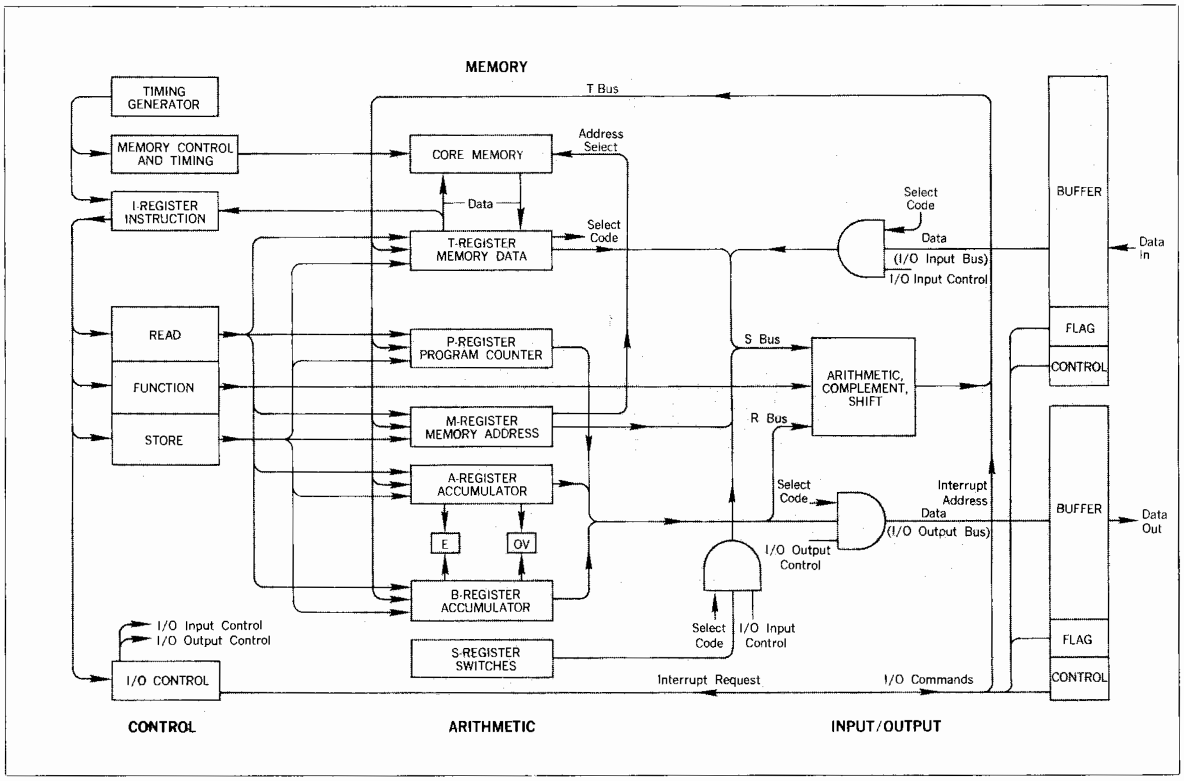


Fig. 4. *Block diagram of –hp– 2116A Computer.*

4. **System and process control.** The computer can receive data from instruments which are monitoring a system or process, operate on this data, and make adjustments to the process in order to optimize its performance. Computer control is automatic and continuous, and the computer program can be changed easily when necessary.

5. **Automatic testing.** The computer can cause a device under test to be stimulated, monitor its response, and reduce the results to any desired form (Fig. 3). It can perform complex tests quickly and automatically, and can even keep records of test results for statistical analysis.

Instrumentation computers have been integrated into several different systems to perform these and other kinds of tasks. Descriptions of four of these systems can be found on page 11.

What's Inside

So far, I've tried to describe the new computer from the point of view of a designer of instrumentation systems, emphasizing how this computer differs from others, why it differs, and what it will do for the systems designer. Now, I'd like to take you inside the machine for a closer look at certain aspects of its design.

Fig. 4 is a block diagram of the instrumentation computer, showing its four major sections, which are:

1. an arithmetic section
2. a memory
3. a control unit
4. an input/output (I/O) system.

Arithmetic operations and temporary storage of data and instructions are accomplished in the nine internal registers indicated in Fig. 4. Eight of these are flip-flop (integrated circuit) registers. The ninth is a row of toggle switches for manual data entry. The contents of all but one of the flip-flop registers are available to the programmer, and are displayed on the front panel.

The A and B registers, called accumulators, execute and hold the results of the arithmetic and logical operations called for by programmed instructions. These registers operate independently, giving the programmer considerable freedom in program design. (Many small com-

Spring Joint Computer Conference

Hewlett-Packard's new 2116A Computer will be at the Spring Joint Computer Conference in Atlantic City, New Jersey, April 18–20, 1967, operating with a variety of instruments and peripheral devices. Stop in and see us at booths E7–E11.

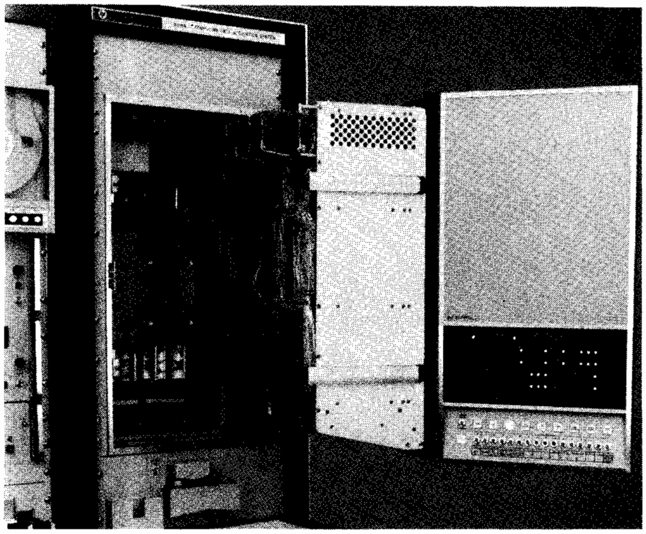


Fig. 5. *All elements of –hp– 2116A Computer are accessible from front of cabinet. Memory modules (the two cube-shaped objects protruding from top of backplane) can be changed in five minutes. Computer is fully enclosed for bench use, weighs only 230 lbs. (10.4 kg) and stands 31½ inches high (782.4 mm). It can be rack mounted.*

puters have only one accumulator; others have two, but they are not independent.) These two registers can be addressed by any memory reference instruction as memory locations 00000 and 00001, thus permitting inter-register operations, such as 'add (B) to (A), 'compare (B) with (A),' etc., using a single-word instruction.

In addition to the nine internal registers, there is an input/output buffer register for each instrument or device connected to the computer. These I/O buffers are located on the plug-in interface cards. Each interface card also contains a flag flip-flop, which is set by the external device to indicate that it is ready to transmit data, and a control flip-flop, which is set by the computer to inhibit or request data transfer, or to cause the external device to do whatever it is supposed to do.

Memory

The instrumentation computer has a coincident-current core memory system capable of storing 4096 16-bit words. A second 4096-word memory module can be installed within the computer main frame (Fig. 5), and external modules can be added to expand the memory capacity to 16,384 words. Another memory option is a seventeenth bit in each word, to be used for parity checking (error detection). Cycle time of the memory, the time required to read and write one word, is 1.6 microseconds.

Cores used in the memory have low temperature coefficients, and are also temperature compensated in order to meet the wide environmental specifications necessary for operation in instrumentation systems. Lithium core material is used because of its insensitivity to temperature variations.

The memory is organized into 'pages' of 1024 words each. One page is designated the 'base' page. There is only one base page, regardless of how many 4096-word memory modules there are. A one-word program instruction stored on one page can order the computer to do something with any other word stored on either the same page or the base page, but not with words stored on the other pages. Words stored on the other pages have to be called out, or 'addressed,' indirectly, using more than one word. Thus it is always possible to address 2048 words (two pages) directly. This is an unusually large direct-addressing capability for this size of computer, and it makes for more efficient, faster-running programs.

Instruction Repertoire

Choosing a set of instructions is the most critical decision in the design of any computer. The computer will be wired to respond to whatever instructions are chosen, and it will respond only to these instructions. If a machine is easy to program, if it carries out programs efficiently, it is because the instructions have been well designed. A look at the instruction repertoire, therefore, will tell a knowledgeable person more about a computer than anything else.

There are 68 basic instructions in the instrumentation computer's repertoire. Fourteen are memory reference instructions, which are used when information is to be obtained from the memory or stored there, forty-one are register reference instructions, used to alter or test the contents of the registers, and the last thirteen are input/output instructions.

Diagrams showing the 68 instructions and how they are coded into 16-bit words are presented on page 9. There isn't space in this article to discuss all of the instructions in detail, but the diagrams shouldn't be difficult to understand, especially if you know something about computers. Notice that each instruction has a three-letter symbol (mnemonic) which is used in writing assembly-language programs.

Of major significance in the design of the instructions is the manner in which several register reference instructions can be combined into a single 16-bit word. The in-

CORRECTION

In the article 'S-parameter Techniques for Faster, More Accurate Network Design,' Feb., 1967, two equations in the table on pages 23 and 24 contained incorrect signs. The correct equations are

$$s_{11} = \frac{(1 - y_{11})(1 + y_{22}) + y_{12}y_{21}}{(1 + y_{11})(1 + y_{22}) - y_{12}y_{21}}$$

$$s_{22} = \frac{(1 + y_{11})(1 - y_{22}) + y_{12}y_{21}}{(1 + y_{11})(1 + y_{22}) - y_{12}y_{21}}$$

-hp- Model 2116A

Instrumentation Computer Instruction Repertoire and Coding

MEMORY REFERENCE INSTRUCTIONS

Execution Time: 3.2μs except JMP 1.6μs and ISZ 3.6μs

15	14 13 12 11	10	9 8 7 6 5 4 3 2 1 0 (Bit No.)
Direct or Indirect Address	Instruction	Base page or current page	Memory Address

AND	'And' (M) to A; result in A
XOR	Exclusive 'or' (M) to A; result in A
IOR	Inclusive 'or' (M) to A; result in A
JSB	Jump to subroutine
JMP	Jump, unconditionally
ISZ	Increment (M); skip if result zero
ADA/B	Add (M) to A or B; result in A or B
CPA/B	Compare (M) with A or B; skip if not equal
LDA/B	Load (M) into A or B
STA/B	Store (A) or (B) into M; A, B unchanged

(M) = Contents of memory location given by Memory Address

REGISTER REFERENCE INSTRUCTIONS

Execution Time: 1.6μs per 16-bit word

Shift-Rotate Group

15	14	13	12	11	10	9 8 7 6	5	4	3	2 1 0 (Bit No.)
0	0	0	0	A or B Register	0	First Instruction				Second Instruction

NOP	No operation
CLE	Clear E-Register
SLA/B	Skip if least significant bit of A/B is zero
A/BLS	A or B arithmetic left shift one bit
A/BRS	A or B arithmetic right shift one bit
RA/BL	Rotate A or B left one bit
RA/BR	Rotate A or B right one bit
A/BLR	A or B left shift one bit (sign cleared)
ERA/B	Rotate E right one bit with A or B
ELA/B	Rotate E left one bit with A or B
A/BLF	Rotate A or B left four bits

INPUT/OUTPUT INSTRUCTIONS

Execution Time: 1.6μs

15	14	13	12	11	10	9	8 7 6	5 4 3 2 1 0 (Bit No.)
1	0	0	0	A or B Register	Macro Call or I/O Instruction	H/C	Instruction	Device Address

H/C = Hold or Clear Flag

I/O Instructions

H/C	HLT	Halt program
0	STF	Set flag bit of selected channel
1	CLF	Clear flag of selected channel
0	SFC	Skip if flag clear
0	SFS	Skip if flag set
H/C	MIA/B	Merge contents of selected channel into A or B
H/C	LIA/B	Load contents of selected channel into A or B
H/C	OTA/B	Output from A or B to selected channel
H/C	STC	Set control bit of selected device
H/C	CLC	Clear control bit of selected device

Register Reference Instructions Coded With I/O Instructions. Device Address = 000001

0	STO	Set arithmetic overflow
1	CLO	Clear arithmetic overflow
H/C	SOC	Skip if arithmetic overflow clear
H/C	SOS	Skip if arithmetic overflow set

REGISTER REFERENCE INSTRUCTIONS

Execution Time: 1.6μs per 16-bit word

Alter-Skip Group

15	14	13	12	11	10	9 8	7 6	5	4	3	2	1	0 (Bit No.)
0	0	0	0	A or B Register	1	First Instruction	Second Instruction						

CLA/B	Clear A or B
CMA/B	Complement A or B (ones complement)
CCA/B	Clear, then complement A or B
CLE	Clear E-Register
CME	Complement E-Register
CCE	Clear, then complement E-Register (sets E to 1)
SEZ	Skip if E-Register is zero
SSA/B	Skip if sign of (A) or (B) is zero (A/B positive)
SLA/B	Skip if least significant bit of (A) or (B) is zero
INA/B	Increment (A) or (B) by one
SZA/B	Skip if (A) or (B) is zero
RSS	Reverse skip sense

Overflow Group (Coded With Input/Output Instructions)

STO	Set arithmetic overflow
CLO	Clear arithmetic overflow
SOC	Skip if arithmetic overflow clear
SOS	Skip if arithmetic overflow set

structions in each word are executed in time sequence, reading from left to right in the diagrams above, and it takes only one memory cycle, 1.6 microseconds, to execute all of the instructions in a word. This powerful capability of combining instructions makes the instrumentation computer's repertoire equivalent to over 1000 useful one-word, one-cycle instructions. It lets the programmer write extremely compact, efficient programs.

Several of the individual instructions are tailor-made for instrumentation work. For example, one of the 'shift-rotate' group of register reference instructions is ALF (or BLF), which causes the word in the A (or B) register to be shifted four places to the left in one memory cycle. This operation is needed often when translating data from the binary-coded-decimal format favored by instrument designers to the straight binary format used by the computer.

If this instruction (ALF) is written twice in a row, it will fit twice into one 16-bit word and will cause the computer to exchange the most significant 8 bits of a word with the least significant 8 bits in only one memory cycle. This operation is needed in changing from 16-bit words to the 8-bit characters used by teletype or punched-tape units, and vice versa. To be able to do it in one memory cycle instead of eight represents quite a lot of time saved.

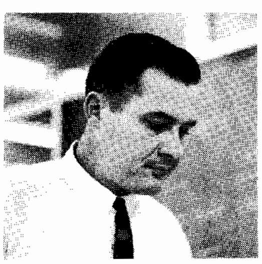
Acknowledgments

So many people have contributed to the design of the instrumentation computer that it would be impossible to list them all individually. I am most grateful for the dedication and the creative efforts of all my colleagues in the following groups:

Group	Leaders
Logic Design	*Edward R. Holland and Eugene R. Stinson*
Memory	*Robert L. Gray and Joseph Olkowski, Jr.*
Input/Output	*Richard C. Reyna*
Mechanical Design	*Tor Larsen*
Programming	*Roy L. Clay*
Applications	*John Koudela, Jr.*

I also wish to acknowledge the valuable consulting services of Samuel N. Irwin, and the support of Henry A. Doust, Jr.'s engineering services group, Joseph B. Dixon's prototype shop, and Norman A. Day, Jr.'s layout group.

—*Kay B. Magleby*



Kay B. Magleby

Kay Magleby received his BS degree from the University of Utah in 1957, and his MS and PhD degrees from Stanford University in 1960 and 1964, respectively. He came to *-hp-* in 1958 and became the first *-hp-* engineer to work on sampling oscilloscopes. He subsequently served as project leader for the development of several sampling-oscilloscope plug-ins. Then, after moving from the oscilloscope division to the advanced research and development group, he contributed to the 8405A Vector Voltmeter project and to research projects in digital techniques, eventually starting the 2116A Computer project. In 1965, he transferred to the *-hp-* Dymec Division as head of the computer development group.

Kay holds several patents in sampling and digital instrumentation. He is active in the IEEE Instrumentation and Measurements Group and is a member of Phi Kappa Phi, Tau Beta Pi, Sigma Xi, and Eta Kappa Nu. He has a wife and three children, and enjoys skiing and woodworking.

SPECIFICATIONS
-hp- Model 2116A COMPUTER

TYPE

General-purpose digital computer, with input/output system and modular software organized for flexible application in on-line instrumentation systems.

MEMORY

TYPE. Magnetic core.

SIZE: 4096 16-bit words. Expandable to 8192 words (in main frame) with plug-in 4096-word module and associated cards, Option M4. Maximum memory size 16,384 words. (Parity bit included in standard stack for use with Option M2, Memory Parity Check.)

ADDRESSING: Memory is organized in 1024-word pages, 2048 words directly addressable.

SPEED: 1.6 microsecond cycle time.

LOADER PROTECTION: Last 64 locations of memory reserved for Basic Binary Loader. Front panel switch, in 'Protect' position, prevents alteration of contents of these locations.

MEMORY PARITY CHECK (Option M2): Permits parity checking within memory. Consists of one plug-in card for each 4K of memory.

MEMORY TEST (Option M3): Enables memory to be tested independently of program control. Consists of one plug-in card.

ARITHMETIC Parallel, binary, fixed point, two's complement.

SPEED

Add	3.2 μs
Subtract	4.8 μs
Multiply	130 μs
Divide	200 μs
Floating point add	375 μs
Floating point subtract	375 μs
Floating point multiply	750 μs
Floating point divide	1.1 ms

(Above are subroutine operations except for Add. Times shown are approximate.)

REGISTERS

Eight internal hardware (flip-flop) registers and Switch register. Contents of all registers except Instruction and Switch register displayed by front panel lamps.

A-REGISTER. Accumulator, input/output. (16 bits.)

B-REGISTER: Accumulator, input/output. (16 bits.)

E-REGISTER: Extend register, links A and B register; indicates carry from A or B register. (1 bit.)

OV-REGISTER: Overflow register, indicates overflow from A or B register. (1 bit.)

T-REGISTER: Transfer register, temporarily holds data transferred in or out of memory. (16 bits.)

P-REGISTER: Program counter. (15 bits.)

M-REGISTER: Memory address register, holds address of next memory location to be accessed. (15 bits.)

I-REGISTER: Instruction register, decodes Memory Reference instructions, holds indicators for zero/current page and direct/indirect addressing. (6 bits, 10-15.)

S-REGISTER: Toggle switches on front panel for manual data entry. Contents of register indicated by switch positions. (16 bits.)

INSTRUCTIONS

68 basic, one-word instructions, in three types:

Memory Reference	(2-cycle)	14
Register Reference	(1-cycle)	41
Input/Output	(1-cycle)	13

Register Reference instructions are micro-operations, can be combined to form over 1000 one-word, single-cycle instructions.

INPUT/OUTPUT

NUMBER OF CHANNELS: 48, 16-bit parallel interrupting channels, with priority control, utilized through plug-in I/O interface cards (1 per channel). 44 channels available for I/O devices; 4 channels reserved for processor options.

MAIN FRAME CAPACITY: 16 channels for I/O devices. Power for interface cards provided from internal supply. (Peripherals draw power directly from 115/230v line.)

INTERRUPT RESPONSE: Servicing of interrupt request (execution of first useful instruction) begins within 3 μs with one I/O channel in use, or within 7 μs for highest priority channel in multiple-channel system.

DATA FORM

PUNCHED TAPE: ASCII. Parity not used, 8th level always punched. (1-inch tape.)

MAGNETIC TAPE: IBM-compatible, 7-channel NRZI. (½-inch tape).

SOFTWARE

Software (punched tape) available consists of:
Compiler, ASA Basic FORTRAN (Extended)
Assembler
Symbolic Editor
Basic Control System
I/O Device Handling Routines
Cross-reference Symbol Table Generator
Hardware Diagnostics
Basic Control System is modular, includes configurator (Prepare Control System) to permit adaptation by user to different I/O arrangements. Also includes Debugging Routines.)

PRICES: 2116A Computer (4096-word memory, no I/O options) $22,000
Memory parity check. Option M2, $1000
Memory test, Option M3, $420
8192-word memory (basic 4K plus 4K additional), Option M4, $8000
I/O Options, $1000 to $15,000

MANUFACTURING DIVISION: *-hp-*-Dymec Division
395 Page Mill Road
Palo Alto, California 94306

Successful Instrument-Computer Marriages

Instrumentation computers are designed to be easy to incorporate into any system which contains electronic, chemical, or medical instruments. Here are four remarkably varied examples of how these computers are being used.

Computing Data Acquisition System

Data acquisition systems are the simplest type of instrumentation system. An elementary data acquisition system converts data from a number of inputs to a form suitable for printing by an output recorder. In more complex systems, some processing of the raw data is done, and the operation of the system may be controlled to some extent by the data.

Because it can easily carry out complex programs, and because its programs can be changed easily, an instrumentation computer in a data acquisition system makes control of the system extremely flexible. It also provides rapid, local data processing, thereby eliminating the loss of time inherent in remote data processing.

A typical computing data acquisition system is shown in Fig. 1. Such systems are used, for example, in testing jet engines: the analog inputs are physical parameters such as pressure, temperature, fuel flow, and engine speed, and the computer outputs are operating parameters such as efficiency and power. The computer not only provides immediate results to help the operator set up the test, but also controls some portions of the test, thereby making the checkout more automatic.

Fig. 2 is a block diagram of the system of Fig. 1, illustrating its use in jet engine testing. Surprisingly, this computing system costs little more than a less flexible

Fig. 1. *Computing data acquisition system can be used in a variety of scientific and industrial applications, including jet engine testing (see Fig. 2).*

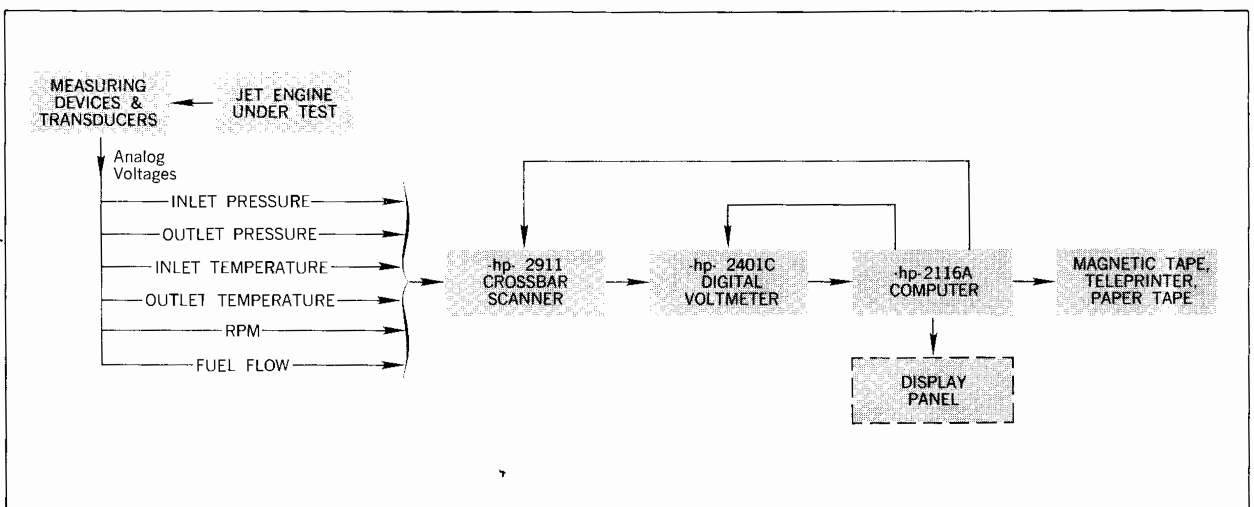


Fig. 2. *Block diagram of computing data acquisition system used for testing jet engines. Computer calculates engine efficiency, power, and other performance data.*

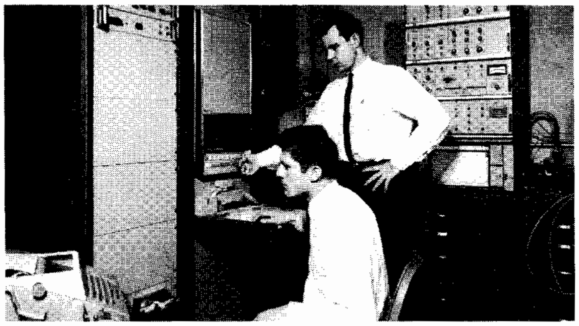


Fig. 3. *–hp– 2116A Computer speeds gas chromatography by analyzing outputs of several chromatographs (one is visible at right), freeing chemist to do less tedious tasks.*

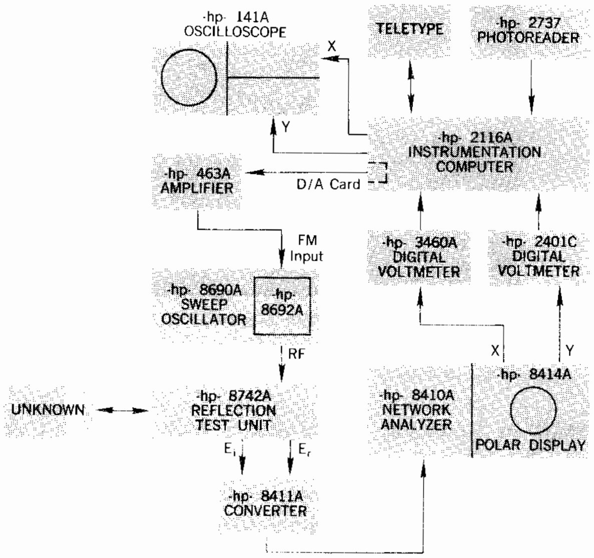


Fig. 4. *Block diagram of microwave impedance-measuring system including instrumentation computer. Computer controls test instruments, refines measurements, and presents results in several forms.*

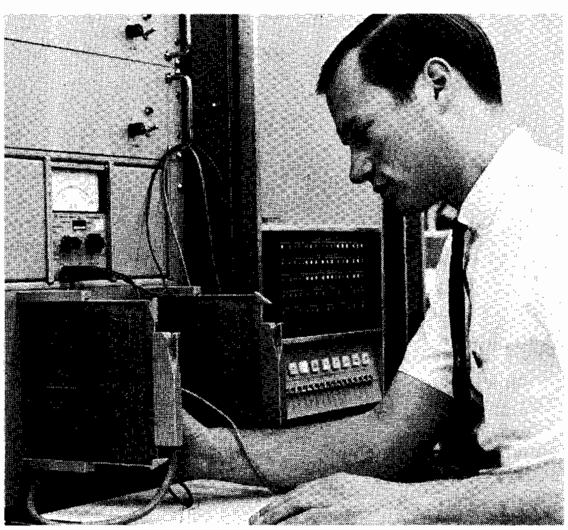


Fig. 5. *–hp– 2116A Computer pretests logic cards for other –hp– 2116A Computers. Tests take only minutes per card, would take months without computer. Computer also keeps statistics for quality control.*

noncomputing system capable of performing some, but not all, of the same tasks.

Gas Chromatograph System

A gas chromatograph is a versatile chemical instrument which provides an analytical chemist with information about the composition of an unknown sample of material. However, it takes a considerable amount of interpretation and analysis to extract this information from the output of the chromatograph, and analytical chemists who do this type of work spend a large portion of their time on data reduction. An instrumentation computer in a chromatograph system can not only free the chemist from time-consuming data reduction, but because it can analyze the outputs of many instruments, it can also greatly increase the number of samples that a chemist can test in a day. Fig. 3 shows part of a developmental system in which an instrumentation computer will be used to analyze the outputs of up to 36 gas chromatographs.

Microwave Impedance-Measuring System

A block diagram of an impedance-measuring system including an instrumentation computer and a network analyzer (Hewlett-Packard Journal, Feb., 1967) is shown in Fig. 4. This system measures the reflection coefficient of an unknown device as a function of frequency, then calculates impedance, admittance, standing wave ratio, return loss, and mismatch loss. Residual errors in the system are measured with a calibrating short in place of the unknown, and the computer automatically subtracts these errors from the measurements. The refined results are stored in the computer memory and displayed on the oscilloscope. The network analyzer provides a Smith Chart display of the raw data.

Logic Module Test System

Final testing of a complex system (e.g., a computer) can be greatly simplified by pre-testing the modules or cards that make up the system. However, manual testing of logic modules can be extremely expensive and time consuming, because there are so many inputs and outputs to be checked. A computer makes pre-testing practical, because it can automatically stimulate the modules and monitor their responses. A complete test of a module with 16 inputs would require 2^{16}, or 64,000, different tests. At one minute per test, it would take a technician six months to test one module. An instrumentation computer can do it in less than one minute. The computer can also keep statistics on the tests for quality control.

Logic modules for *–hp–* instrumentation computers are tested by the computer system illustrated in Fig. 5. The response of the module under test is compared with that of a reference circuit and the operator is alerted if the test device's response is not within specified limits. ■