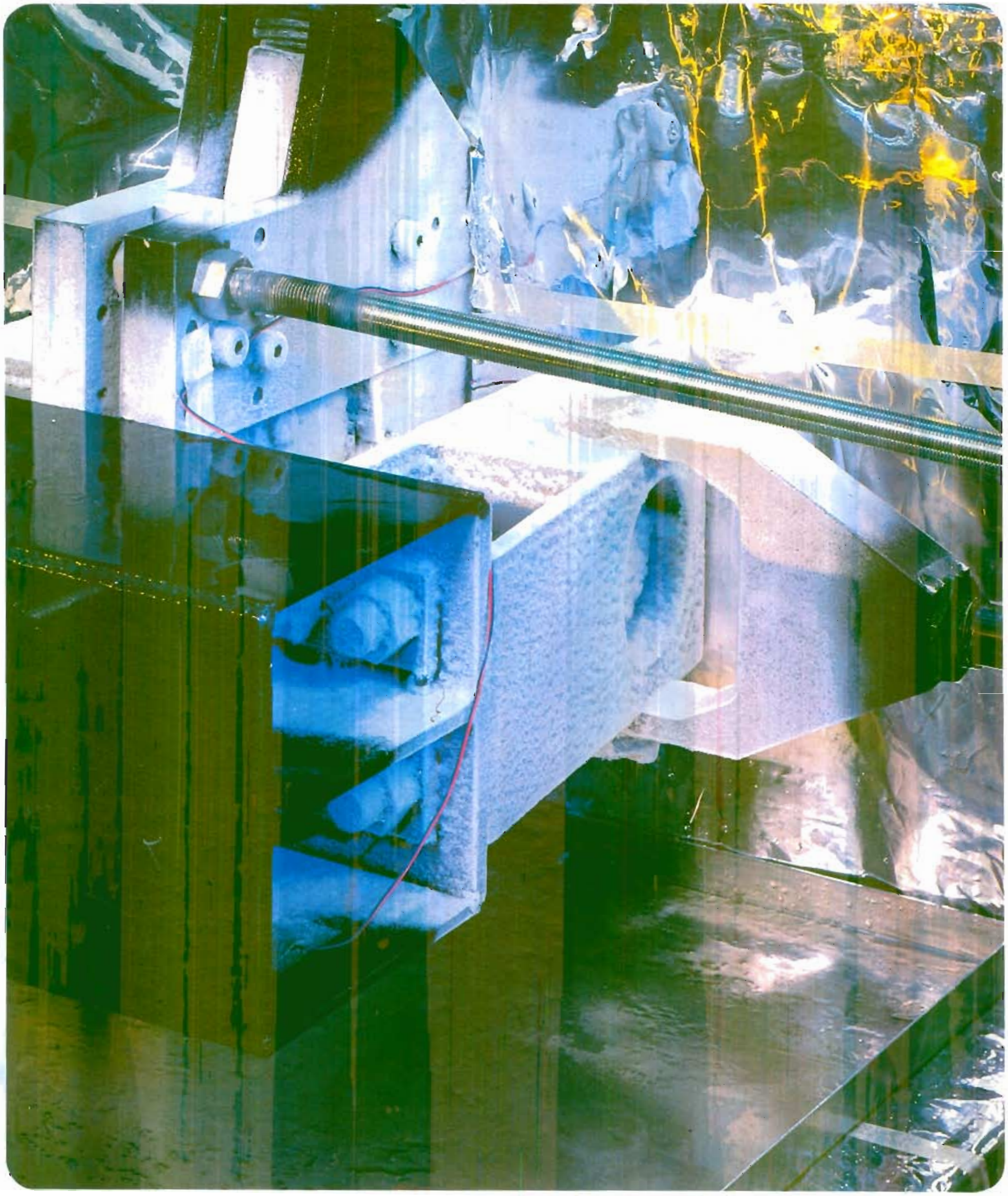
# Keyboard

Sept-Oct/79 A Publication of Hewlett-Packard Desktop Computer Division



Keyboard
September-October 1979

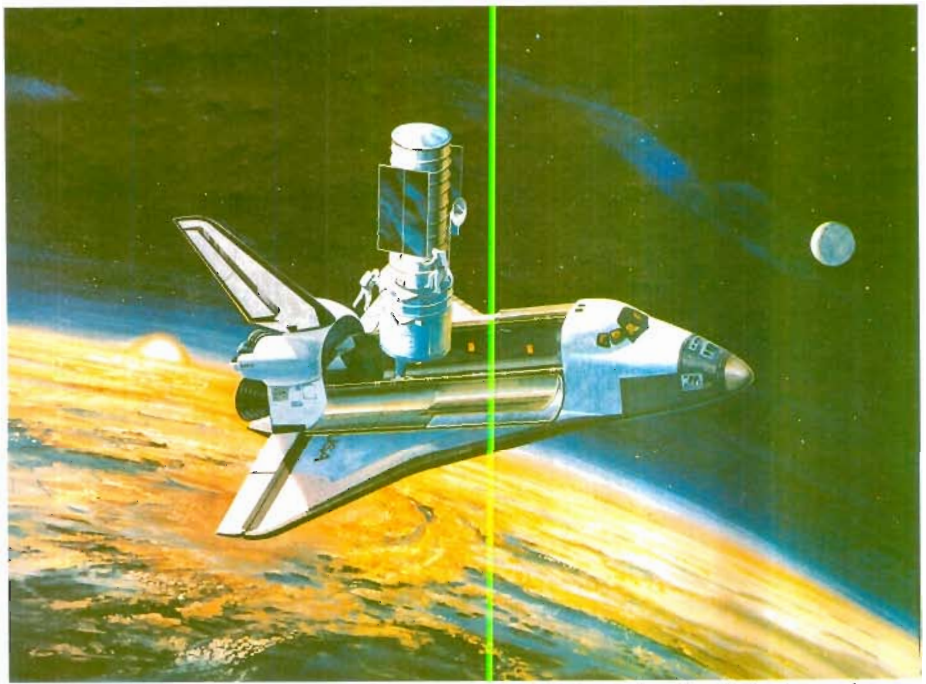
## Cover

An experimental version of one of the latches destined to hold payloads firmly in place within the space shuttle is shown on this issue's cover just after cold-temperature tests. During the tests, the latch assembly was doused with liquid nitrogen while stress tests were run. The cover story begins on page 1.

**Photo credits**

page 1, upper photo; page 3, upper photo — National Aeronautics and Space Administration, Washington, DC.
cover; page 1, lower photo; page 3, lower photo — Mel Ryan-Roberts, Ball Aerospace Systems Division
pages 4, 5 — Peter Lauener and W.F. Lauener Ltd., Thun, Switzerland

# Taking the shuttle home



On the way to orbit, space shuttle payloads will be secured in the payload bay by latches under development at Ball Aerospace. Elevons being tested at NASA are essential to the shuttle's safe return.

*by Court Adams and*
*Bill Sharp*

By the middle of next year, the first launching of a new class of spacecraft will open another era in the exploration of the universe beyond our planet's atmosphere.

The space shuttle developed for the U.S. National Aeronautics and Space Administration, Washington, DC, represents a significant advance over earlier space vehicles. The shuttle orbiter itself looks rather like a short, strubby-winged airplane. And this similarity is deliberate, for the craft will glide down to land on earth at the conclusion of each space mission.

In the effort to prepare the shuttle for the first launch, NASA and its contractors are at work completing portions of the space shuttle hardware. Some of the test controlling and data collecting and handling that is a part of these projects is being done with Hewlett-Packard Desktop Computers.

This article is divided into two parts. The first, by Court Adams of Ball Aerospace, describes how a System 45 is aiding the development and testing of a high-reliability payload retaining latch for use in the extreme environment endured by the space shuttle. The second part, by *Keyboard* Editor Bill Sharp, tells about a System 35 that is being used in the development of shuttle parts crucial to a winged spacecraft's reentry into earth's atmosphere, something which has never been attempted before.

## Designing latches

Ball Aerospace, Boulder, Colorado, U.S.A., is designing and building hardware that will hold payloads firmly in position inside the shuttle payload bay during transport to and from earth orbit. The hardware consists of remotely operated latches that will be used to secure payloads by grasping cylindrical trunnions mounted on each payload module. The latches, attached to rails within the orbiter bay, can release payload modules when the shuttle reaches earth orbit.

This multiple exposure of a shuttle latch model illustrates the mechanism's movement.

One design requirement is that each latch mechanism must be strong enough to withstand any shuttle crash that the pilots could survive without releasing the payload into the crew compartment. At the same time, the latch must allow some freedom of movement. The shuttle, like all modern aircraft, has a relatively flexible fuselage. Its payloads will tend to be either more or less rigid than the fuselage. This means that the latches must allow some movement, or the payload or shuttle could be damaged.

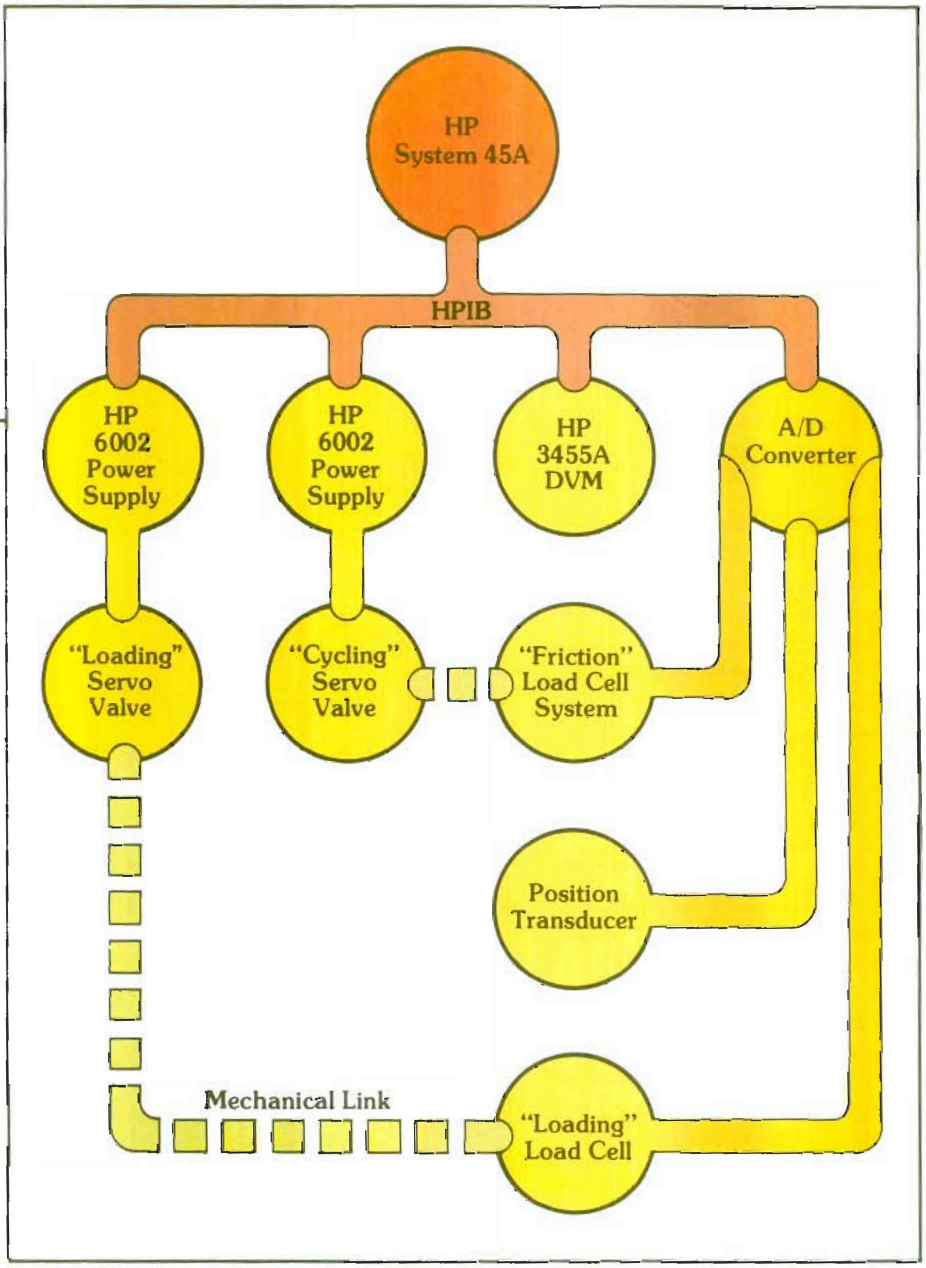
To ease movement of the payload trunnion in the grasp of each latch, Ball researchers have had to develop space-compatible lubricants. These must be as efficient in hard vacuum from −73° C to +177° C (−100° F to +350° F) as they are in ambient air.

Several materials appear to exhibit the necessary coefficient of friction of 0.1, which is approximately equivalent to smooth ice on ice. These materials now are being evaluated under simulated operating stresses at Ball.

## Simulating stress

The test setup has been constructed to simulate stresses and evaluate the lubricants. It includes a large loading frame, two hydraulic systems to apply loads, analog load cells to measure the forces, and a System 45 Desktop Computer operating in an HP-IB configuration. The System 45 controls test operations, stores and reduces test data and prints out test reports. Figure 1 depicts the test setup.

The test program sets the HP 6002 "loading" power supply to a desired



The flowchart above depicts interrelationships between the components of the test system used by Court Adams and his colleagues. Ball Aerospace researchers developed their own A/D converter.

load via the hydraulic servo valve. While the load is held constant, the 6002 power supply moves the cycling servo valve in and out. This causes the trunnion to slide back and forth through the latch bearings. The sliding friction force is measured by the cycling load cell system.

## Testing the hardware

Typically, a set of 10 cycles represents a data point. The force of friction, which is somewhat load-dependent, is determined by a least squares fit and stored on tape.

The program includes the following modules:

1. The loading subprogram applies selected static loads to the trunnion and provides several safety interlocks to prevent accidental dangerous overloads.
2. The cycling subprogram causes trunnion movement through the latch, measures the friction and stores friction data on tape.
3. The data reduction subprogram calculates the mean coefficient of friction, standard deviation and upper limits; generates the test report format and prints out the test report on the internal thermal line printer.

The entire program makes extensive use of graphics in instructions to the test operator in order that his be an involved, "hands on" role. Tape cassettes are used for all program and data storage.

Researchers found it necessary to design one piece of input/output equipment to interface with the analog instrumentation. They made their own eight-channel, analog-to-digital (A/D) converter. While the HP 3455 DVM is an extremely accurate instrument, they needed something with a sampling rate 10 times faster. Accordingly, one purpose of the custom converter is to sample voltages at a very rapid rate and store the data directly in memory via the 16-bit parallel I/O bus.

Variable scan time was another important factor. Typically, data is taken from five transducers. Ball researchers want the option of selecting the order in which the transducers are scanned, and the time duration of the scan. This feature is immediately available in the converter via program commands. Also, due to the high input impedance of the converter, researchers are able to scan live circuits of any impedance.

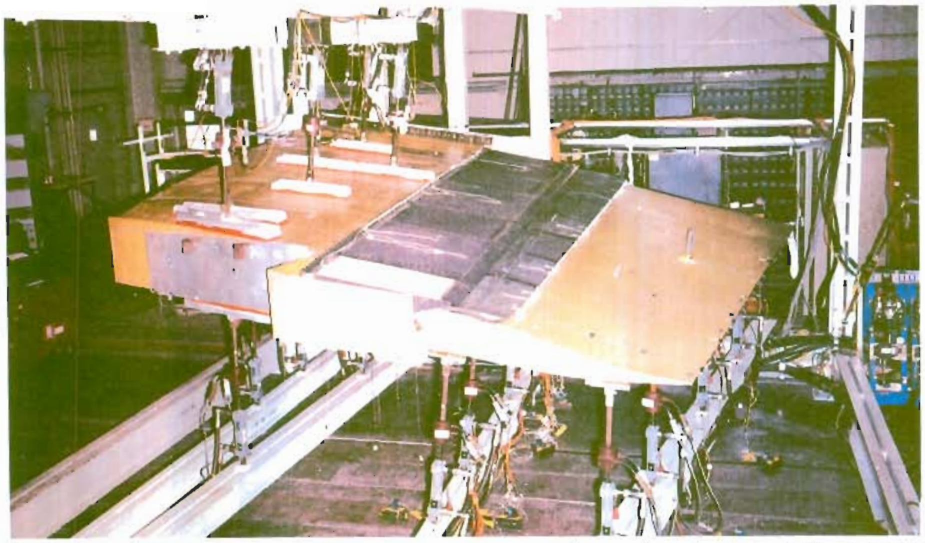
## Controlling descent

At the same time that researchers at Ball Aerospace are working to perfect the payload latch and its lubricants, researchers at NASA's Dryden Flight Research Center, Edwards, California, U.S.A., are conducting other tests.

Earlier reentry craft used in space programs all have lacked wing surfaces. They were designed simply to survive the fiery reentry heat and deploy parachutes to slow the craft's descent through the remainder of earth's atmosphere to a one-point landing on water or land.

Because it is designed to survive reentry heat with wing surfaces fully functional, the shuttle can glide to a smooth landing with a great deal of control on a fairly conventional runway.

But in order to execute a glide through the atmosphere and land safely, the shuttle must have movable


Elevon seal tests at NASA subject the working wing segment shown above to stresses by hydraulic jacks, heating and pressurization to make certain the seals will survive reentry.

wing surfaces to control its descent. And these surfaces must survive the heat and stress of reentry — something that has never been accomplished before.

### Studying performance

The shuttle has wing parts called elevons, which control both the up and down movement and the roll of the shuttle. But the elevons are a vulnerable point when the shuttle reenters earth's atmosphere. Sophisticated seals on the moving surfaces are designed to keep hot gases from damaging internal wing parts during reentry. Efforts using a System 35 at the NASA Flight Research Center are directed toward verifying that the elevon system of the shuttle can withstand the punishment of reentry.

In the test, NASA and Rockwell International researchers subject a portion of a shuttle wing including an elevon to simultaneous heating, loading and pressurization to simulate reentry conditions. Researchers at work on the shuttle wing tests are headed by Walter Sefic, manager of the Flight Loads Research Facility. Leon Weirather, instrumentation engineer, is responsible for the System 35 program.

Ed Hamlin, now a senior electronics engineer with Teleco Oil Field Services, Inc., Middletown, Connecticut, wrote the original program when he was with NASA. Hamlin explained that the data collection system then in existence did not have enough display capability to suit NASA's needs. While an older Systems Engineering Laboratory computer took data for research and control, they purchased the System 35 to monitor safety data to protect the wing structure being tested.

### Running tests

Experiment parameters are relayed through a Computer Products Inc. RTP 7471 low-level, analog controller. The System 35 provides researchers with real-time display of safety data. The desktop computer system includes a 9885A Flexible Disc, 9876A Thermal Printer, 9878A I/O Expander and 98034A HP-IB Interface.

Instant data feedback allows the operator to react quickly to any change. This could be a deterioration in the quality of raw data, or a dangerous change in the force produced by the hydraulic system, threatening the wing structure under test.

The System 35 displays a menu of seven major tasks that may be chosen by selecting various Special Function Keys, such as, "record calibration data," and "get calibration status." The operator is next presented with another menu to clarify where the operator is in the program and how he can proceed. There is no need to spend time referring to a manual.

The system is programmed to calibrate the test equipment by taking readings from all load cells while they are under zero load and under maximum load. These readings are stored, along with the slope and intercept of each. A/D output readings from the cells can be converted to pounds of force, assuming a direct relationship between the load and the readings taken.

Once calibration is complete, the program can go into a scan mode where data from each channel is displayed on the screen and updated every 1.6 seconds. The operator has a choice of several forms in which the data can be displayed.

By using the System 35 to follow the test in progress, the researchers have an independent monitor. It is an interactive source of information for the operator that does not necessitate tying into the control or data handling systems.

It is helpful to remember that, while these two efforts involve one of the most demanding fields of science, manned space exploration, they are being developed with the help of commonplace applications of desktop computers. The functions of the computers, examined in this light, should be familiar to most readers. The shuttle itself is, after all, little more than a carefully orchestrated assemblage of existing technologies.

K

Court Adams is a test engineer with the Integration and Test Department of Ball Aerospace Systems Division. Prior to his development work on the space shuttle payload latch, he worked on the star tracker navigation system for the shuttle. He received his BSEE from the University of Florida.

Court Adams
Ball Aerospace Systems Division
P.O. Box 1062
Boulder, Colorado 80302 U.S.A.

# Engineering for metal strip casting

*by Peter Lauener*

In the field of metal strip casting, there is a strong trend toward what is called continuous strip casting (CSC). This means that without interruption, the casting line will produce a metal strip beginning with liquid metal in the furnace, and ending with a coil that is suitable for transportation and further treatment.

As it can be of great advantage to cast a relatively thick strip, a modern CSC line will produce a strip 20 to 50 mm thick and 550 to 2000 mm wide. The desired coil at the end of the line has a strip thickness of 4 to 6 mm at the most. Special rolling mills operate as part of these lines in order to achieve the necessary reduction in metal thickness.

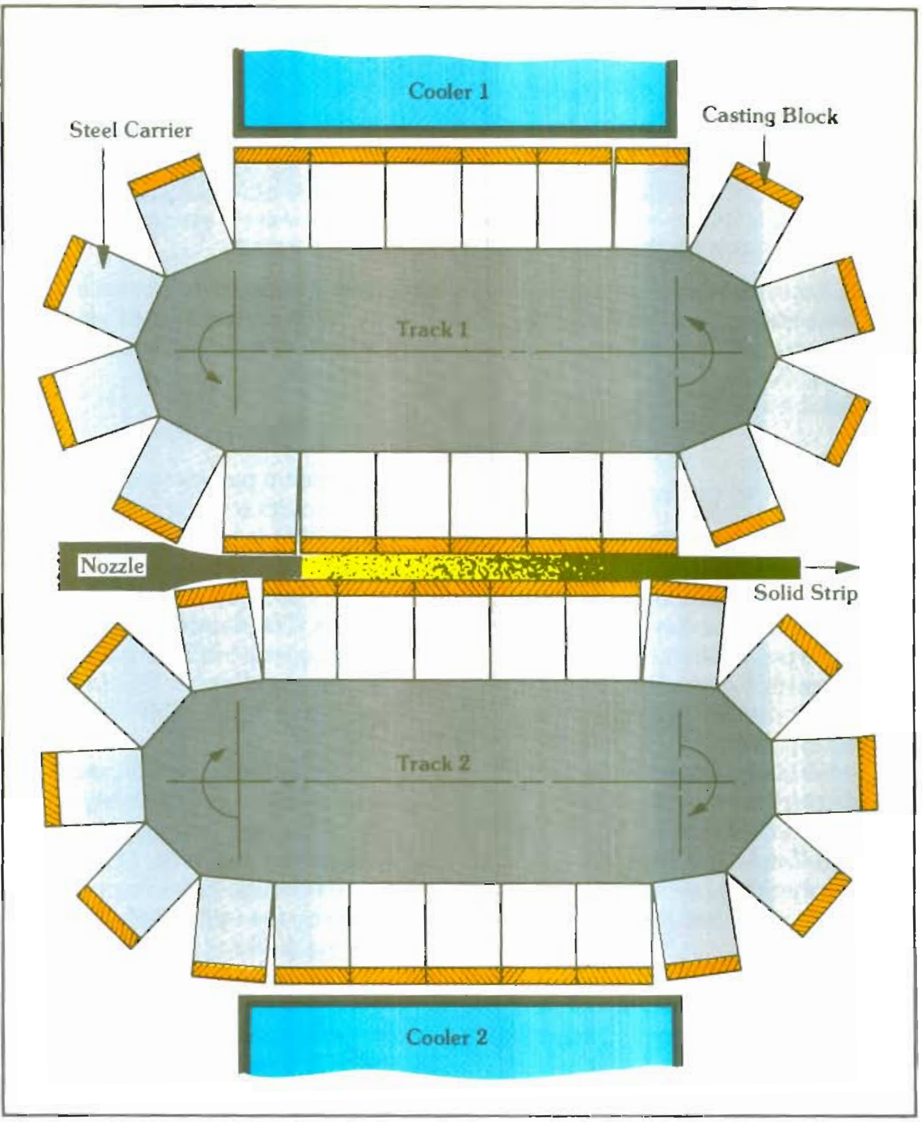
## Casting with moving blocks

The Swiss engineering company, W.F. Lauener Ltd., Thun, Switzerland, specializes in designing the various components of CSC, and has successfully developed, set up and put into operation several such production lines. Design engineering computations are performed using a Hewlett-Packard 9831A Desktop Computer.

One of the most recent developments is a so-called "block-caster," realized through close collaboration with the Swiss aluminum firm Alusuisse. To date, the block caster is unique because it allows the continuous strip casting of certain high-quality aluminum alloys.

The primary parts of this casting machine are the blocks, which come into contact with the liquid metal, thus taking up heat from the casting and cooling it to the solid state. The process can be described in two steps:

1. Metal casting blocks take heat from the liquid metal, which cools.
2. The casting blocks pass through a cooler, after which, at their initial temperature, they again come into contact with liquid metal.



The continuous strip casting line operates by producing a steady stream of molten metal at one end of the dual roller set, and rapidly cooling the metal to produce a solid strip at the other end.

The process of transferring heat from the casting to the cooling liquid by means of the casting blocks takes place continuously.

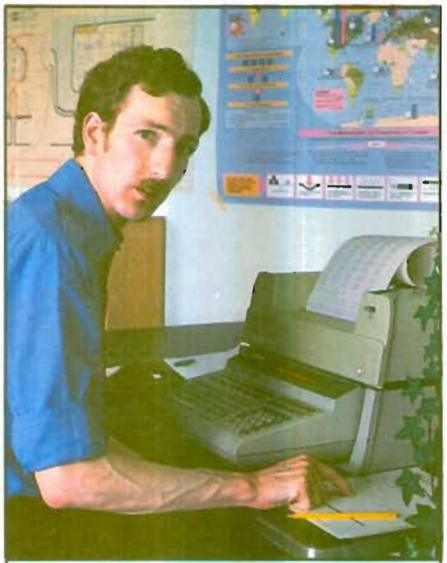
## Revolving tracks

Two revolving tracks make up the actual casting machine. Casting blocks mounted on steel carriers are the elements of the tracks.

Adjoining blocks on the track form a surface which moves parallel to the corresponding surface formed by blocks of the other track. The distance between the planes of the two tracks determines the thickness of the cast strip of metal.

Some of the blocks within each track thus partially form the actual mold, and move along with the solidifying metal. At the same time, other blocks on the opposite end of the track are moving through a cooler. The cooler brings the temperature of the hot blocks back down to what it was before the blocks entered the casting section and came into contact with liquid metal.

Workers install one of the roller sets for a strip casting line. The copper alloy plates which contact and cool the molten metal are clearly visible in the photograph.



Peter Lauener conducts theoretical engineering investigations related to the development of new machines such as planetary mills and the casting machine described here. He studied mechanical engineering at the Federal Institute of Technology in Zurich. He now works with W.F. Lauener Ltd., Thun, Switzerland.

Peter Lauener
LAREX AG
4565 Recherswil
Switzerland

### Casting block heating studies

In order to control the cooling process of the solidifying casting and achieve high quality not only on the surface, but also in the internal structure of the casting, it is extremely important to know more about heat flow, stress and deformation of the casting blocks.

Necessary calculations, unthinkable without a computer, were done with the aid of a 9831 Desktop Computer. Our computer is equipped with a 9866B Thermal Printer and 16K bytes of memory.

For these computations:

$T$ = temperature within the casting block at distance x from the surface and at time t
$T_0$ = surface temperature of the casting block when in contact with the liquid metal
$T_a$ = temperature of the casting block before it comes in contact with the liquid metal
$t$ = time of contact between casting block and cast metal
$a$ = thermal guide number
$s_0$ = thickness of the casting block
$n$ = positive integer

$$B = (T_a - T_0) \bullet 4/\pi$$

$$C = (2n-1)^2 \bullet a \bullet \pi^2 \bullet t/s_0^2$$

$$D = 2n-1$$

$$E = (-1)^{n+1}$$

$$F = \cos((2n-1) \bullet \pi \bullet x/s_0)$$

The heat flow equation

$$T = T_0 + B \bullet \sum_{n=1}^{\infty} (\exp(-C) \bullet E \bullet F/D)$$

was used as a basis for the applied program to compute the temperature distribution in the casting blocks. For a casting block cross section of 70 by 200 mm, the program output contains the temperature of 224 different points after any chosen contact time between the cast metal and the casting block.

### Temperature non-uniformity

The non-uniformity of the temperature increase or decrease within a casting block is considered when the stresses, and consequently the lines of equal strain (isopascals) within the casting block are calculated.

A further part of the output describes the shape of the casting block in three directions, assuming unhindered deformation under the computed heat distribution.

When making the calculations for a block with the above-mentioned cross section, the program requires 8 two-dimensional (14 by 16) arrays with split-precision accuracy and approximately 30 simple variables. The 16K byte version of the 9831 can easily handle problems up to twice the size of the one referred to in the example.

As our programs increase in number as well as in size, an expansion of the system must be considered. But for us the 9831 in combination with the thermal printer has proven to be a very reliable and efficient machine, especially with respect to scientific applications in the field of mechanical engineering. K

# Crossroads

## Random numbers — part II

by John Nairn,
Hewlett-Packard Company,
Desktop Computer Division

> *"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."*
>
> John Von Neumann

In a previous Crossroads article (1978/1 *Keyboard*) we looked at the nature of random number sequences and at some of their applications. In this article we will see how computers generate random sequences and look at ways of testing how "good" these sequences are.

We saw in the last article that an ideal random number sequence would contain an unlimited number of distinct values in a given range having a specified distribution. Usually a uniform distribution (all values equally likely) over a range $0 \leq R < 1$ is preferred, since other ranges and distributions can be derived from this one. In addition, the elements of the sequence should be independent of one another. The history of random number generators is a chronicle of the search for algorithms that produce sequences (sometimes called pseudo-random sequences) that closely approximate these properties.

In the final analysis, however, the question of how good a particular random number generator is depends entirely on the application. For example, in a random, computer-controlled light show, the main requirement would be that the human eye not recognize any repetition of the pattern. If, on the other hand, the sequence is used to simulate dealing cards in a game, uniformity of distribution would be more important. And if we were randomly sampling data points for statistical evaluation, we would want to ensure that there is no correlation between the random samples chosen and the type of analysis being performed. In short, a random number generator that is perfectly acceptable for one application may be totally unacceptable for another.

Part I of John Nairn's discussion of random number generators appeared in the 1978/1 issue of *Keyboard*. You can obtain a copy of this issue, and/or a copy of the paper mentioned in reference 2 by writing to *Keyboard* at the address on the back page.

So, how does a computer generate these pseudo-random sequences? The most common method is to apply some function or algorithm to the last element of the sequence, $R_n$, to obtain the next element, $R_{n+1}$. Symbolically, this can be written

$$R_{n+1} = f(R_n) \qquad (1)$$

The element that started the sequence, $R_0$, is known as the seed. Given a seed and an algorithm, the entire sequence is determined and can be regenerated at will.

One very popular function is the so-called linear congruential method (LCM). This rather forboding name simply indicates that the function being used is of the form

$$R_{n+1} = (A \bullet R_n + C) \bmod M \qquad (2)$$

where A, C and M are appropriately chosen constants and N mod M is the remainder after N is divided by M. The choice of constants depends on the type of arithmetic (integer, decimal, binary, etc.) the computer uses and on the desired properties for the sequence generated. An excellent reference in this area is Donald Knuth's series "The Art of Computer Programming" [1].

If the arithmetic indicated by equation (2) is done exactly, Knuth shows that many properties of the sequence such as uniformity, independence and sequence length can be determined analytically. And indeed, anyone having exacting requirements would want to employ a generator whose analytic properties are known. It is for this reason that Knuth states, "random numbers should not be generated with a method chosen at random."

But what if I make up my own generator, or I have one from a program library and I don't know its properties? Many HP desktop computers provide a function for generating random number sequences using a modification of the LCM technique [2]. Since its intent is merely to provide a convenient source of random sequences for simple applications, this modified LCM generator trades off an exact implementation for less ROM space and shorter computation time.

In such situations, theoretic determinations are replaced by empirical tests. For a given application, we merely need to determine what properties are required of the random sequence and test the proposed generator to determine if it has these properties.

Some tests are quite trivial. In the earlier example of the computer-controlled light show, the generator is satisfactory if the results "look good." Other tests are less trivial, but the method for performing

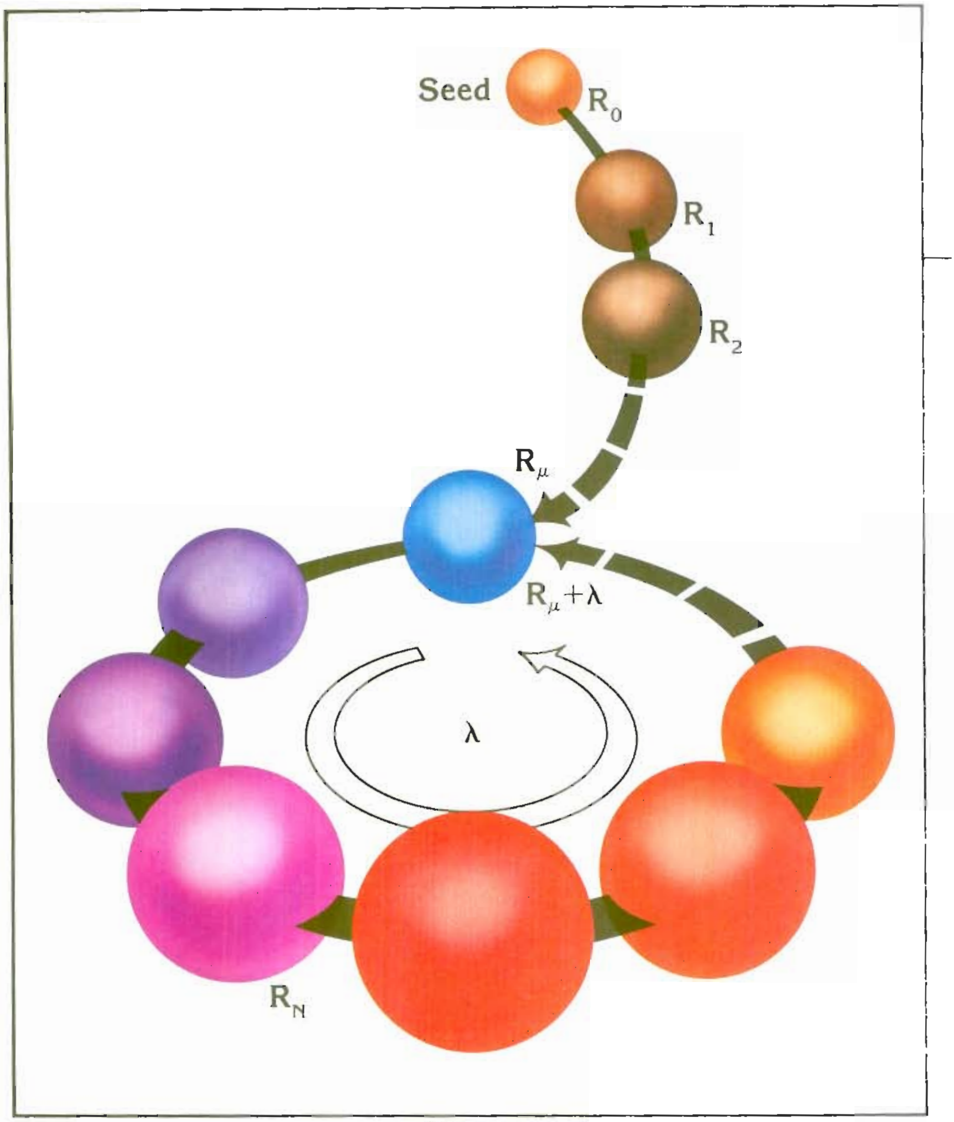


Figure 1

them is obvious. For example, if uniform distribution over the range is important for the application, we can divide that range into a number of intervals and count how many values of the sequence fall into each interval. The results will show whether the sequence is uniform, or consistently favors some intervals over others.

Still other tests may be non-trivial and not immediately obvious. A good example of this is the determination of the length of a sequence. Every random number generator of the type shown in equation (1) must eventually enter a loop. This structure of random sequences is shown in figure 1. Since a computer can only represent a finite number of values, eventually a number $R_\mu$ will be generated that has already occured earlier in the sequence. By equation (1) this number generates the same successor that it did previously, and the generator cannot escape the loop. I have seen many cases of programmers believing that they have run 100,000 random samples, only to learn that they actually ran the same 10,000 samples ten times.

How then do we determine the length of the sequence before it begins to repeat? If we try to compare each new value to all previous ones we will probably run out of memory (not to mention the enormous computation time) before finding $R_\mu$. If we save only every tenth or hundredth value, we could easily miss the repeated value.

If the generator used allows specifying the seed, it is a relatively easy matter to find an element guaranteed to be on the loop. Merely calculate in parallel the two sequences

A: $R_1, R_2, R_3, \ldots, R_N$
B: $R_2, R_4, R_6, \ldots, R_{2N}$

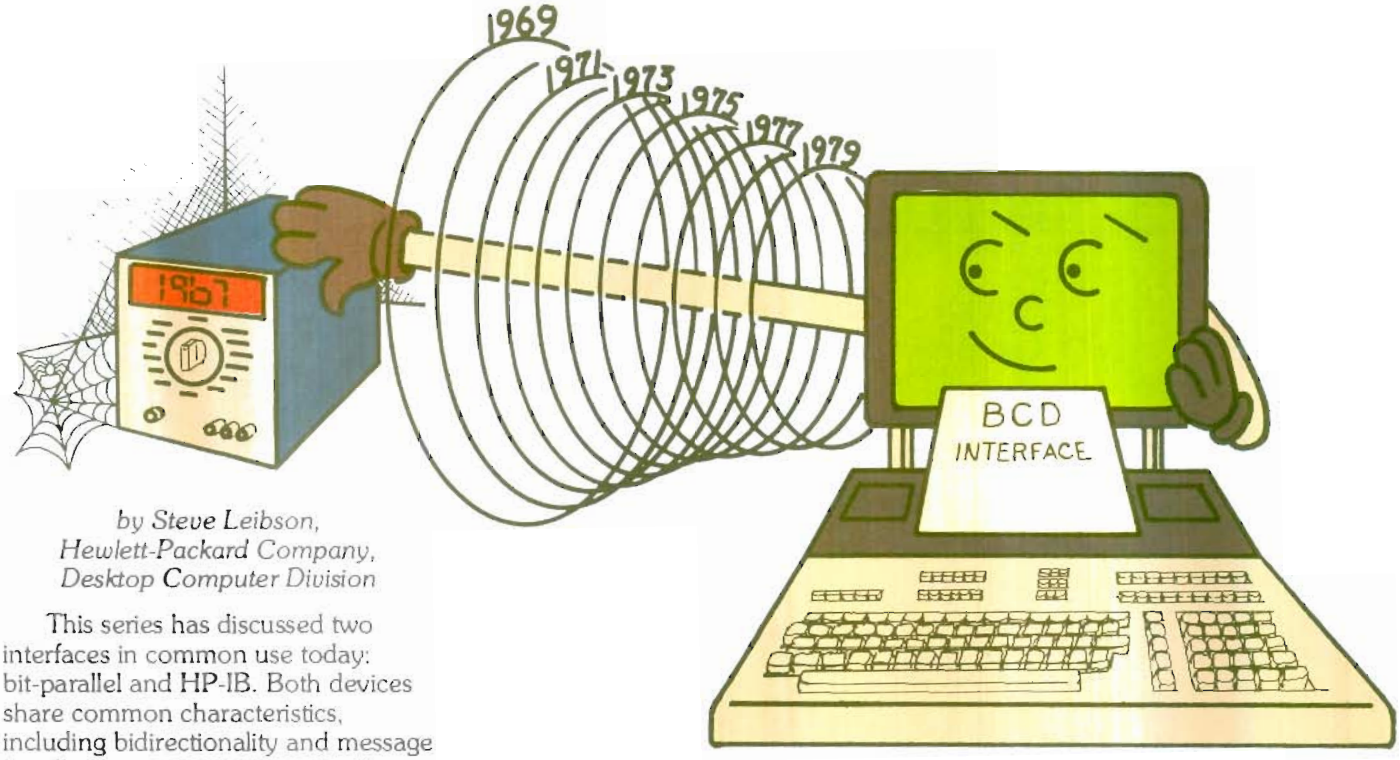
Sequence B is generated by setting the seed to the last number obtained and calling the generator twice, discarding the first value. At each step, merely compare the two values, $R_N$ and $R_{2N}$, to see if they are equal. No previous values need to be saved or compared. It can be shown that since sequence B is going around the loop twice as fast as sequence A, when they finally meet $\mu \leq N \leq \mu+\lambda$. That is, $R_N$ is guaranteed to be on the loop. Once $R_N$ is found it is a simple matter to traverse the loop once more counting the number of elements necessary to return to $R_N$, and thus obtaining the value of $\lambda$. The determination of $\mu$ is a little more difficult, but usually $\lambda$ is the important result. Remember also that $\mu$ and $\lambda$ are functions of the seed as well as the algorithm, so don't choose seeds randomly either.

In summary, one should realize that pseudo-random number generators do not produce ideally random sequences. Corollary #723 of Murphy's Law states that "the likelihood of a particular random number generator having a desired property is inversely proportional to the sensitivity of the given application to that property." When in doubt — test!

**References:**

1. D.E. Knuth, *The Art of Computer Programming*, Vol. 2 (Addison-Wesley, 1969).
2. J.H. Nairn, "9825/35/45 Random Number Generator." Copy available from *Keyboard* editor on request.

# The BCD interface



*by Steve Leibson,*
*Hewlett-Packard Company,*
*Desktop Computer Division*

This series has discussed two interfaces in common use today: bit-parallel and HP-IB. Both devices share common characteristics, including bidirectionality and message handling in portions (characters).

When instruments were first connected to computers, instruments by themselves did not have the electrical sophistication in their circuitry for either of these interfacing techniques. A different method of connection, called binary coded decimal (BCD), was used.

## Computer bears burden

This method allows the burden of the intelligence for the interface to reside with the computer, which somehow has to accept all of the information in parallel. But interface designers created the required circuits, and the BCD interface remains popular in instrumentation. It provides a link to older instruments that have been reliably turning out data for years. BCD also is generally simple to design into a current instrument.

Equipment which uses the BCD interface usually measures some physical parameter such as voltage, current or weight. These instruments send information to the computer, but do not receive information from it.

Thus, the BCD interface is unidirectional. Information flows only from the instrument to the computer. Control lines may be available from the computer to actuate ranges or control other aspects of the readings, but they are not used for messages.

## Using 10 binary codes

BCD is simply a coding method which takes the ten decimal numerals 0 through 9 and encodes these into 10 binary codes. The encoding is the binary sequence 0 through 9 as follows:

| Binary code | | | | Numeral |
|---|---|---|---|---|
| Bit 3 | Bit 2 | Bit 1 | Bit 0 | |
| 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 1 |
| 0 | 0 | 1 | 0 | 2 |
| 0 | 0 | 1 | 1 | 3 |
| 0 | 1 | 0 | 0 | 4 |
| 0 | 1 | 0 | 1 | 5 |
| 0 | 1 | 1 | 0 | 6 |
| 0 | 1 | 1 | 1 | 7 |
| 1 | 0 | 0 | 0 | 8 |
| 1 | 0 | 0 | 1 | 9 |

Note that the encoding is simple binary and that four binary digits (bits) are required to represent 10 numerals. Also note that codes 1010 through 1111 are not used. Because of this, BCD coding is not as efficient as pure binary coding. This inefficiency allows simple decoding for display to human operators.

An analogous situation is that of a calendar. Each page contains five weeks, which is more than enough to hold any month. The extra spaces for days are left blank, which is inefficient. But because people block the days of the year into months, the convenience of separating the months into different pages more than makes up for the blank spaces.

Each digit of a reading therefore requires four signal wires to transmit the binary values associated with that digit. Since all digits are available on the I/O connector simultaneously, the connector may have as many as 40 or 50 signal pins on it for a high-resolution instrument.

*At its inception, BCD interfacing was a big success. The tedious job of taking data was greatly simplified.*

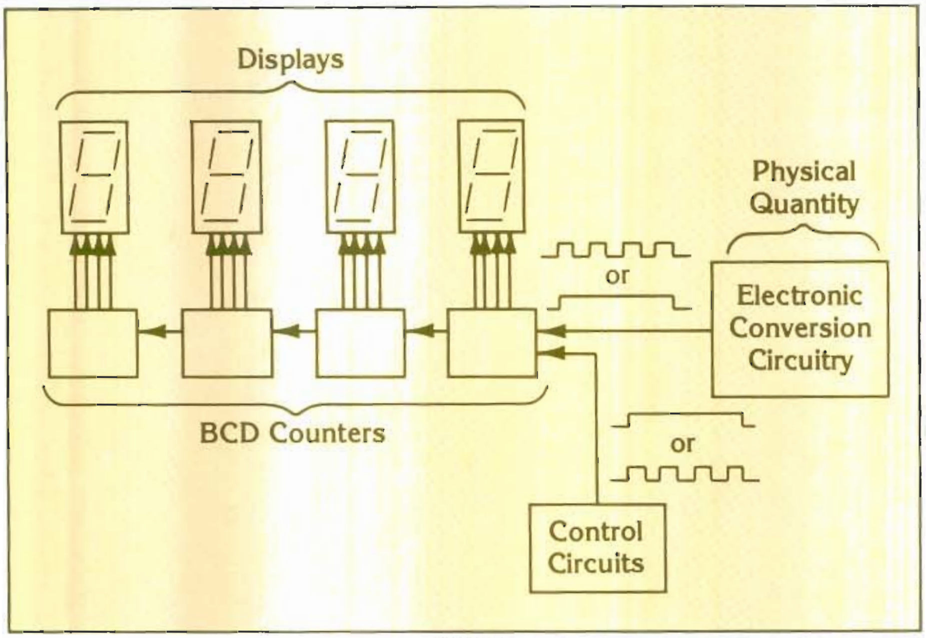


Figure 1

### Implementing BCD

Why is the BCD interfacing technique easy to implement in these measuring instruments? The answer lies in the circuitry used to actually do the measuring. A diagram of a typical measuring instrument is shown in figure 1.

Input to the instrument, such as voltage, current or weight, is fed to an electronic circuit that converts the physical parameter to a signal that represents that parameter. This signal may be either a square wave with a frequency that is controlled by the physical parameter, or a pulse with a duration that is controlled by the parameter.

If the signal is a frequency, it is fed to the input of a counter which counts for a precisely-controlled length of time. If the signal is a pulse, it controls a counter which counts the oscillations of a precise square wave.

In either case, a count is obtained which is proportional to the input parameter. A small input generates a small count, and a large input generates a large count. The count drives the digits in a display. If the count is binary, it is difficult to read because people don't think in binary, computers do. Fortunately, it is possible to build a BCD counter.

If several BCD counters are cascaded, they are able to produce outputs that are easily displayed. This works in much the same manner as the odometer of a car. Each wheel of the odometer has the numerals 0 through 9 printed on it. Each time a wheel on the odometer makes a full turn, it advances the wheel to the left of it by one count.

In a chain of BCD counters, each time a counter advances from 9 to 0, it advances by one the next counter on the chain. Each BCD counter has four signal lines that represent the state of the counter. The signal lines are used to drive one digit of a display. When the digits from all of the counters are combined, they form the complete reading of the instrument.

The preceding explanation is true for a wide range of measuring instruments that have digital displays. Most use a counting technique to convert a physical quantity into a digital display.

### Adding a printer

The first accessory instrument designers added to digital instrumentation was a printer. This made it possible for an unattended instrument to log its own readings. Signal lines are brought out from the counters in the instrument to drive the print wheels in the printer. Each digit has its own wheel in the printer. Signals from the BCD counter control the position of the print wheel when it hits the paper.

Extra signals are only required for this interface to tell the printer when the data on the BCD lines is valid (print command), and to allow the printer or other external device to control the rate at which readings are made (external trigger). These two wires form a handshake mechanism between instrument and printer.

### Automating experiments

At its inception, BCD interfacing was a big success. Experiments which had required an attendant to write down the readings could now be automated. Printed logs could be obtained for production testing. The tedious job of taking data was greatly simplified.

Now if a printer could do all that, just think what could be done by replacing the printer with a computer. Data would no longer have to be punched on cards or entered by hand. The eyes and ears of a computerized process control loop were about to come into being.

### BCD at HP

Let's look at a BCD interface for a Hewlett-Packard desktop computer (figure 2) and see how it works. The 98033A BCD Interface is used with the 9825A/S, System 35A/B and the System 45A/B. It has several inputs that connect to the instrument. There are enough signal wires for an eight-digit mantissa with a sign bit and a single-digit exponent, with a sign.

In addition, there is a bit to represent overload or overflow and four bits for a function code. When the computer takes a reading, the interface card scans its input lines and transforms the BCD digits, all available in parallel, into a string of ASCII characters which the computer reads one at a time.

Sixteen characters form one reading as shown in figure 2. In this way, the reading may have as many as eight digits in the mantissa, and a single-digit exponent. This is usually more than sufficient to handle a BCD instrument. Unused digits can be wired to always read zero.

### Interfacing flexibility

Since there is no standard for BCD interfacing, the 98033A provides flexibility in the interpretation of the signal wires. It can be configured for either positive true logic, where logic 0=0 volts and logic 1=+5 volts, or negative true logic, where logic 1=0 volts and logic 0=+5 volts. Note that the voltage levels 0 and +5 are TTL standard, a logic family introduced in the late 1960s and currently dominating logic design.

Character numbers 10, 13 and 16 ("E", "," and "LF") are generated within the 98033A Interface. They aid the computer in deciphering the meaning of individual digits coming from the instrument. The "E" is a prefix that indicates an exponent will follow. The comma separates the reading from the overload bit and the function code. "LF" is a line feed character that terminates the message.

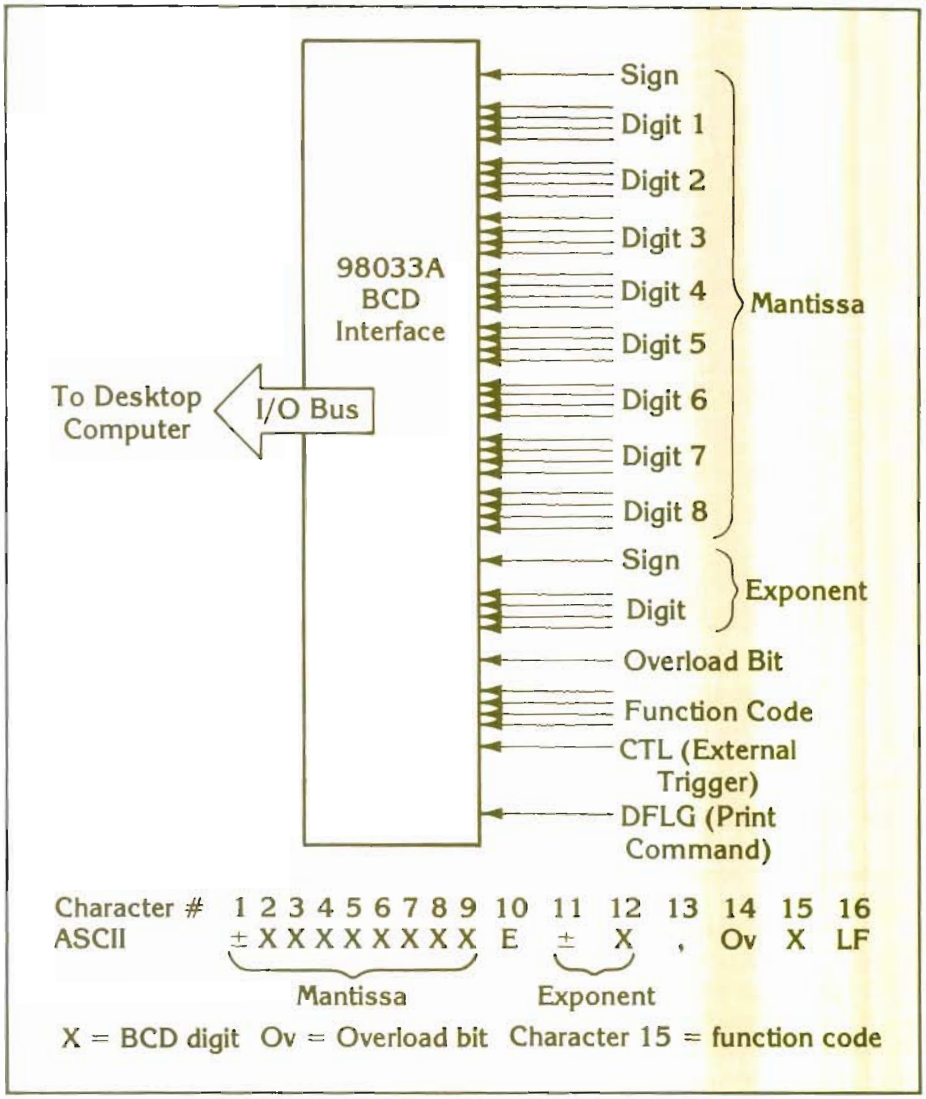


Figure 2

Intelligence in the 98033A replaces that which might otherwise be required in the instrument's interface. This interface serves as an electrical link to the past. It provides a data path between today's computers and earlier interfacing instruments. It also eases the burden for instrument designers who may not be able to justify development costs for a more complex interface. K

# Improving product quality

*by Brenda Hume,
Hewlett-Packard Company,
Desktop Computer Division*

The high cost of manufacturing electronic parts such as printed circuit boards can be reduced through effective quality control. Hewlett-Packard's Desktop Computer Division uses a combination of hardware and software known as the Parts Failure Information System (PFIS) to identify parts and components that have a high failure rate.

Management can use this information to determine if there is a problem with workmanship or with the vendor supplying the part. And corrective measures can be justified at any production level with information generated from PFIS data.

## Improving reliability

PFIS improves reliability of subassemblies used in the manufacture of HP 9815, 9825 and Systems 35 and 45 Desktop Computers. A subassembly is an assembled group of parts referred to with one part number, such as a circuit board with components mounted on it, or an interface cable.

PFIS consists of an HP 2100S minicomputer, a System 35 Desktop Computer and an HP 3000 minicomputer. The 2100 is the controller for the test. It has 175 programs, one to test each of DCD's 175 subassemblies. System 35 acts as a go-between and feeds data from the 2100 to the HP 3000. The 3000 maintains the master data file from which PFIS reports are taken.

## Transferring data

System 35 dramatically reduces the work of data transfer for PFIS. In three minutes it can enter data that would take four hours manually.

HP 3000 software is written in COBOL. The System 35 uses BASIC and Assembly; the program for accepting data from the 2100 is written in BASIC.

To interface the 35 with the 3000, a software emulator is required. The emulator, written in Assembly language, has proved to be eight times more efficient than using a detailed and lengthy BASIC program. Designers of PFIS chose the System 35 because it was the only HP desktop computer with both BASIC and Assembly language capability.

## Aiding management

The test system also includes an HP-IB 59309A Real Time Clock that measures the length of time the test takes for each subassembly. This provides valuable information for management. For example, if 200 000 subassemblies are tested in one year, and 400 000 subassemblies are projected to be tested the next year, two systems might be required to handle the workload.

Data from the real time clock helps forecast future hardware requirements. This same information can also be used to charge each department for the time required to test its parts.

## Testing subassemblies

The test begins when a technician places one of the many subassemblies on a computer-controlled vacuum device. This device pulls down the assembly, where it rests on a "bed of nails," with contacts that correspond to the electrical connections on the subassembly. Each different subassembly has a unique configuration of component parts. For each type there is a program written to conduct the test and read in data.

Each subassembly is identified by codes contained in the computer memory. After the subassembly is identified, the correct program is accessed, and data is collected.

Data from the test indicates whether the part meets specifications. If a part fails, the test shows where in the production cycle the failure occurred.

### Analyzing failures

Causes of failures may also be determined. This could be either the circuit application stress levels or receipt of parts having a high "infant mortality" rate (new parts that frequently fail). With this information, vendors supplying the defective parts are identified.

PFIS reports provide information to answer specific questions related to product-level failures, including: how many subassemblies were tested during the reporting period, how many defects occurred, what failed in these subassemblies, at what point in the product test cycle did the defect occur and how many production instruments were repaired during the period.

In addition, PFIS reports provide answers at the subassembly and component level. For example, data accumulated includes subassemblies tested and failures occurring during subassembly testing, circuit positions of failures and points in test cycles where component failures occurred.

In addition to the HP 2100S, System 35 and 3000, there are other test systems set up to transfer data to the 3000 master file. One of these uses a System 45 Desktop Computer in place of the System 35. DCD's Parts Failure Information System tests and collects data on an average of 32 000 subassemblies per month — a job that would require 40 technicians to do manually.

## System 45A support

Effective November 1, 1979, Hewlett-Packard will terminate production of the 9845A mainframe, which is part of our System 45. Since the introduction of the 9845B mainframe in April, demand for the 9845A mainframe has diminished rapidly. This was expected, since the 9845B mainframe provides users with new capabilities and additional software, and does this in the same price range as its predecessor.

For this reason, it is no longer practical for us to maintain production lines for the 9845A mainframe. But, while 9845A production will cease, we will continue to support the product through the availability of ROMs, software packs and field installation kits until November 1, 1980.

In accordance with standard Desktop Computer Division practice, technical support and consumables will be available for 10 years after termination of mainframe production. Full service will be available on-site for at least five years after production termination, and at any HP service facility for the remainder of the 10-year support period. Beyond 10 years, service will be available on a "best effort" basis.

External peripherals and interface cables are not affected in any way by this change. All System 45A peripherals and interface cables are used also with the System 45B, and will continue to be available.

You may want to advise your local HP sales office or field engineer of any need for additional 9845A mainframes. Upgrade kits are available to convert your System 45A to a System 45B configuration if you require additional capability.

# Programming Tips

## Transferring data from the 9830 to the System 45

This tip is to correct an error which appeared in a programming tip in the Jul/Aug 1979 *Keyboard*. The error concerns the improper use of the CARD ENABLE statement. I erroneously stated that to set the 98032 Opt. 30 card for data input to the System 45, the following commands should be used:

S=⟨select code of 98032 card⟩
CONTROL MASK S;1
CARD ENABLE S

In actuality, the CARD ENABLE S should be changed to:

WRITE IO S,5;1

Likewise, to set the I/O card for output from the System 45, the following statements appeared in the tip:

CONTROL MASK S;0
CARD ENABLE S

They should be replaced by:

CONTROL MASK S;0
WRITE IO S,5;0

The major points to remember concerning the CARD ENABLE statement are:

1. Use CARD ENABLE only for I/O transfers which utilize the interface card's interrupt capability.
2. Be careful when using CARD ENABLE on a high-order select code. If you try to access a tape cartridge when doing so, the system will hang up until an interrupt is received from the card.

For details, write to *Keyboard* at the address on the back page, and we will send you some additional information on this tip.

*by Martin Nielsen,*
*Hewlett-Packard Company,*
*Desktop Computer Division*

## READ/DATA capability in HPL

A deficiency of HPL as compared to BASIC is that it lacks the capability of storing data within the program section. A program that requires a large amount of constant data is awkward in HPL. You must either enter many lines like 1→A, or write a program to accept the data from the keyboard and put it on tape or floppy disc. BASIC has a "READ" statement that does a read-from-memory of data contained in "DATA" statements.

It is possible to duplicate this capability in HPL using a surprising property of the "list" statement as reported by Howard Rathbun in a DCD paper available from *Keyboard*. This involves using the Advanced Programming ROM's single-quote function to return a string to the General I/O Programming ROM's "list#" statement. This string is actually the name of a buffer set up using the Extended I/O Programming ROM. In this manner, the "list" statement is tricked into using a buffer, something not otherwise allowed by the syntax.

To synthesize a "READ/DATA" capability, another single-quote function is used, this time within a "red" statement. Besides returning the buffer name, this function searches through memory for dummy "data" statements, using the "list#" statement and another single-quote function.

The dummy "data" statement is simply a long label containing the characters "data" and a list of data items. When the function finds one, it blanks out the "data" and statement number and returns the buffer name to the "red" statement. The "red" statement uses this buffer as a fast read/write buffer and reads the data from it.

The following short program demonstrates this technique. The program requires the String, Advanced Programming and Extended I/O ROMs.

```
0: dim D$[100]
1: buf "data";D$;3
2: red "DATA";A;B;C
3: prt A;B;C
4: red "DATA";D;E;F
5: prt D;E;F
6: stp
7: "DATA":list #"dat";
   r0;r0
8: r0+1→r0
9: if (pos(D$;"'
   ''data")→r1)
   =0;buf "data";gto -2
 10: ""→D$[1;r1+7]
 11: ret "data."&char
     (p1+48)
 12: "dat":ret "data.1"
 13: "data 1.1;2.2;3.3":
 14: "data 5;6;7":
```

If you would like further information on this application, including an explanation of each line in this program, and a copy of Howard Rathbun's discussion of transferring a program to a string on the 9825, please write to *Keyboard* at the address listed on the back page. *Keyboard* will be happy to mail you this information.

*by Joseph D. Pepin,*
*Western Electric Engineering*
*Research Center, P.O. Box 900,*
*Princeton, New Jersey 00540 U.S.A.*

# Update

## BASIC Users' Club meeting

*by Ron Mora,*
*Hewlett-Packard Company,*
*Desktop Computer Division*

The BASIC Users' Club held another in its series of regional meetings June 27 at the HP Englewood, Colorado, sales office. The meeting was attended by more than 30 club members representing firms and organizations including Coors, Rockwell International, the U.S. National Oceanic and Atmospheric Administration, U.S. National Bureau of Standards and the U.S. Air Force.

Topics discussed by the group included services of the BASIC Users' Club itself, which provides a program exchange for users of HP System 35 and System 45 Desktop Computers, applications of desktop computers, programming training and graphics. Hardware demonstrations of desktop computers and other products were available for the members, with opportunity for hands-on experience.

One member commented about the meeting, "It's extremely valuable talking to other users . . . I've learned things here that would have taken me months to find out on my own."

The meeting was hosted by District Sales Manager Rod Waage, and his field team including Colin Campbell, Jim Jensen, Bill Haselmire, John Abegg and Bud Sloan. Factory personnel assisting included System 45 Instructor Donna Kimble, Keyboard Editor Bill Sharp, Programmer Kathy Osborne, as well as Dell Fischer and Ron Mora of the BASIC Users' Club. K

## Certified error-free data cartridge

At time of shipment, the new HP 98200A Data Cartridges are certified 100% error-free. Each cartridge is a compact magnetic tape device which provides reliable data storage for all 9800 series desktop computers and HP 264X series display terminals.

HP tests each cartridge automatically over its entire length by recording and reading back tightly-spaced data bits on both tracks.

Features of the cartridge include speeds up to 230 cm/s (90 in./s) acceleration rates up to 5080 cm/s$^2$ (2000 in./s$^2$) and as much as 5.4 megabits of unformatted data storage (1600 bits/in. on two tracks).

Model 98200 is available as one carton of five cartridges. Unlabeled versions and quantity discounts are available. K

## More transparencies

In the Update section of the May/Jun 1979 *Keyboard*, we incorrectly described the Overhead Transparency Kit, part number 17055A, which can be used with HP desktop computers and either HP 7221A or 9872A plotters to produce excellent overhead transparencies in up to seven colors. If that sounded good to you, it should sound even better now, because the kit includes 200 sheets of transparency film, not 100 as reported in that item. K

Keyboard

Hewlett-Packard, 3404 E. Harmony Road
Fort Collins, Colorado U.S.A. 80525
Phone: 303/226-3800

Publications Manager - Ed Bride
Editor - Bill Sharp
Editorial Assistant - Chris Stumbough
Art Director - Hal Andersen
Typography - Paula Dennee

For further information on HP products or applications, please contact your local Hewlett-Packard Sales and Service Office or write to Keyboard

**CHANGE OF ADDRESS:** To change your address or delete your name from our mailing list please send us your old address.
Send changes to Hewlett-Packard Keyboard, 3404 E. Harmony Road, Fort Collins, Colorado 80525, U.S.A.
For Europe: send changes to Hewlett-Packard NV, Post Box 529, 1180 AM Amstelveen, Holland

Sept. 1, 1979