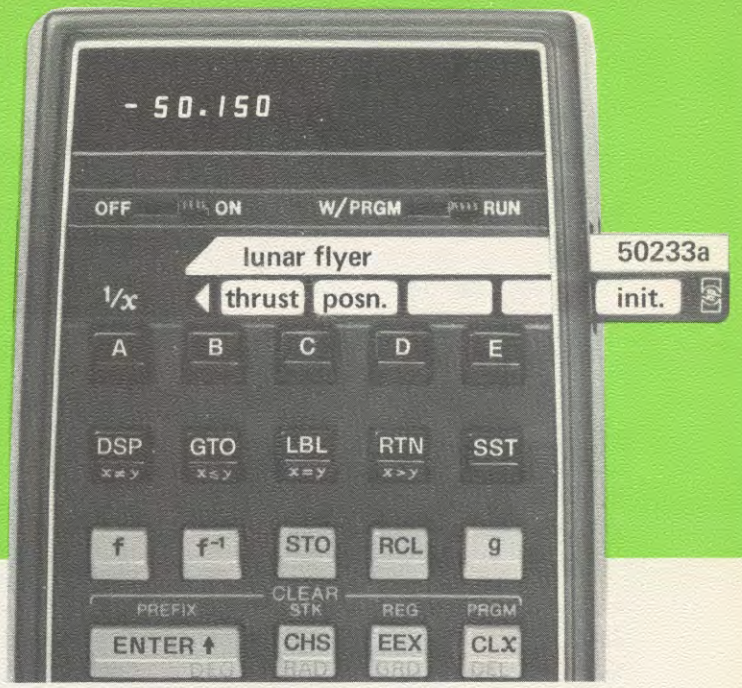
HP-65 USERS' NEWS

HEWLETT hp PACKARD



Volume 1
Number 2

HP-65 IN SPACE RENDEZ-VOUS

An HP-65 programmable pocket calculator – just like the one you own – played an important role in the historic Apollo/Soyuz July 1975 space rendezvous.

The calculator was used to calculate two critical mid-course correction manoeuvres just prior to the link-up between the U.S. Apollo and the Russian Soyuz spacecraft. These manoeuvres took place 12 and 24 minutes after terminal phase initiation (i.e. the closing stages of the flight before rendez-vous).

It was also used as a back-up for Apollo's onboard computer during the final lining up prior to rendezvous and docking. Here, the first job concerned the coelliptic manoeuvre (putting both spacecrafts into identical orbits) when the vehicles came within approximately 100 miles of each other. The second task covered the terminal phase initiation calculations when Apollo was 22 miles from Soyuz. In both instances, the HP-65's answers to the problems were compared with those of the onboard computer.

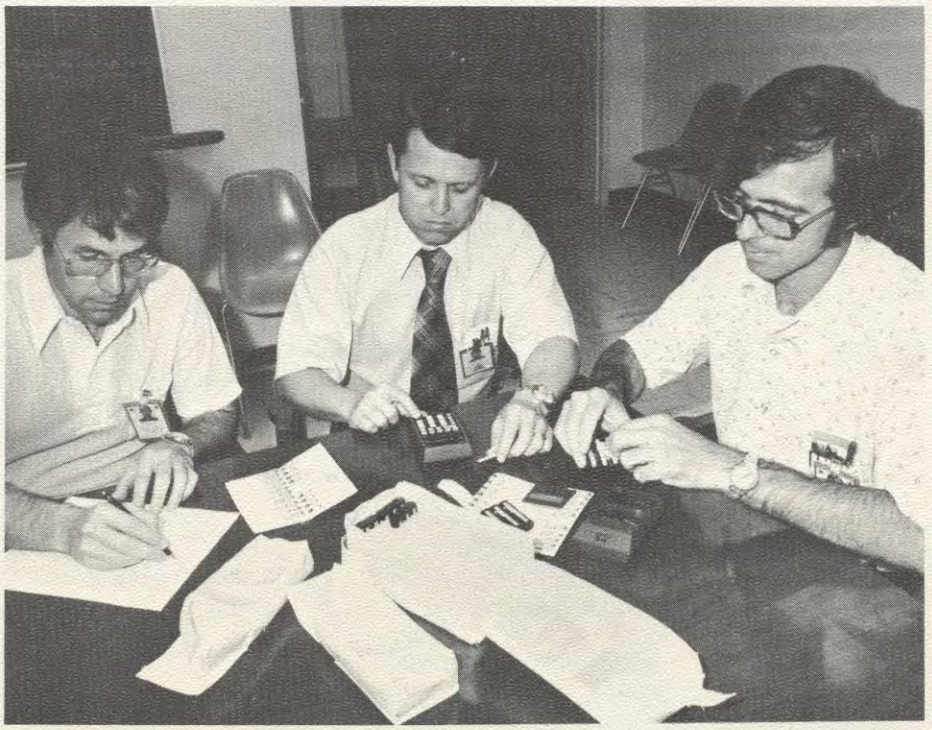
Had the onboard computer failed, however, the HP-65 remained the only available source for calculating the mid-course manoeuvres because – at that phase of the mission – the spacecraft was not in communication with ground stations.
A third set of calculations performed by the HP-65 enabled the astronauts to point Apollo's high-gain antenna precisely at an orbiting satellite to optimize communications with Earth.

NASA scientists wrote programs of up to 1,000 steps and recorded them on magnetic cards for the astronauts to feed into the HP-65 to perform the critical calculations automatically during flight.

Two HP-65's were taken on the space trip, along with four sets of programs and six spare battery packs. This is not the first HP pocket calculator to go in space: an earlier model, the HP-35, earned its space wings on the Skylab missions.

IN THIS ISSUE

User's Library Corner 2
Top Ten Programs 2
Seeing is Believing. 2
Programming Notes. 3
Multiple Key Functions 3
Splitting Merged Codes. 5
Display Power Consumption. 6
Not and error 6
How to Make an HP-65 Talk. 6
Private HP-65 Users Clubs. 7
Application Pac News 7
Two New Miracle Machines 8



A NASA technician (centre) and colleagues from McDonnell Douglas Corporation working on the HP-65 calculator carried aboard the Apollo Command Module during the recent Apollo/Soyuz link-up project. The 10-day mission tested a common docking system and performed joint experiments between American and Russian crew members. (NASA photo)

USERS' LIBRARY CORNER

On November 1st, 1975, the Users' Library celebrated its first birthday. In that one year, membership has grown from zero to several thousands.

Program exchange activities have grown in proportion. We are very pleased with the way so many of you have contributed hundreds of extremely interesting and stimulating programs. It is not surprising that demand for these programs has been so high among your fellow members. Even allowing for the relatively slow initial phase, we can look back with satisfaction on the progress made. Thanks to the support you and other members have given, the project is as successful as anticipated and its popularity continually increasing.

We have already taken steps to widen the Library's scope, by including a section covering its activities in a questionnaire sent to all HP-65 owners. The response is highly encouraging. These answers are currently being analysed on an 8K HP-9830 desk-top calculator for a clearer indication of user needs.

A quick appraisal has already shown that there is a requirement for more technical information on both machine capability and on programming. As a direct result this and subsequent issues of "USERS' NEWS" will carry a more extensive programming section to carry articles contributed by experienced HP-65 owners.

For those of you who develop new ideas, don't hesitate to submit them to the Library to be shared with your fellow members. We shall be happy to publish them in future issues of "USERS' NEWS'. 🕮

TOP TEN PROGRAMS

Addendum I included a fine crop of programs. The ten most requested programs are listed below in order of popularity:

1.	T-Statistic for 2 means (Variables not equal) Robert Gordon, Northwood, UK	No. 50057 A
2.	Determinent of a 4 x 4 Matrix Giorgio Merlani, Monza, Italy	No. 50002 A
3.	Test for zero correlation André Scholberg, Corseaux, Switzerland	No. 50021 A
4.	Confidence Limit for the correlation coefficient Andras Bekessy, Genève, Switzerland	No. 50130 A
5.	Logic Equations Thomas von Randow, Hamburg, Germany	No. 50134 B
6.	Two Variables correlation coefficient linear regression Michel Crolais, Fontenay, France	No. 50149 A
7.	Position of the sun in the sky, time of sunrise and sunset Hubertus Wöhl, Göttingen, Germany	No. 50007 A
8.	T-Statistic/Data Retrieval Robert Gordon, Northwood, UK	No. 50056 A
9.	Least squares fitting of a 2nd degree parabola Siegbert Förster, Rosstal, Germany	No. 50091 A
10.	NIM with 3 Piles Graham Komlosy, Addlestone, UK.	No. 50104 A

In appreciation of these fine submittals, we are offering an Application Pac of his choice to the person sending the most requested program, and a Pac of blank magnetic cards to the contributors of Nos. 2-10 on the list.

SEEING IS BELIEVING

Or is it ? The old saw has taken a knock with Jim Caldwell of Austin, Texas.

Ever wonder how you would use an electronic calculator if you were blind ? No real problem to memorize the keyboard but how would you read the display ? You can't touch the LED's in the display, and even if you could, you wouldn't be able to feel the numbers on them.

You'd probably give up. But not Jim. He is totally blind – as a result of an accident – but he **can** read the display on his HP-65 ! What's more, he has written many programs on his HP-65. He uses an Optacon*, an electronic device that converts the visual images of letters and numbers into tactile forms that he can identify through feeling.

Despite his blindness and partial paralysis, Jim was determined to do something with his life. A couple of years ago he realised that he couldn't compete in his chosen field without a calculator. So he wrote to several manufacturers suggesting ways to make a braille-type readout.

He got one reply. Hewlett-Packard's Advanced Product Division started Jim on a new track; the remarkable Optacon. So he set about investigating the possibility of using one to read the display on an electronic calculator. Then, to top things off, HP came out with the HP-65.

Although Jim already had an HP-80 on order, he was excited about the prospect of using the Optacon on the machine that offered him the most potential.

Today, Jim uses the Optacon with a special head adapted to fit the display of his HP-65 and is using the pair to help him prepare a doctorate in Management Science and Operations Research. 🕮

* (The Optacon is a battery powered, portable device produced by Telesensory Systems in Palo Alto, Calif.).

PROGRAMMING NOTES

This section is included to help you become better aquainted with programming techniques so that you may increase the effective power of your HP-65. See how the following ideas demonstrate efficient use of the 65's capabilities. You will be surprised how much more you can do with 100 steps ...

MULTIPLE KEY FUNCTIONS

The following explanations by Richard Nelson show the various techniques that have been used to achieve multiple functions from the keys A to E.

Flags

The first approach might be to use a flag. If the flag is unset, pressing 'A' causes one function to execute. If the flag is set, pressing 'A' causes another function to be executed. To simplify use, let's use 'E' to set the flag, and have the program unset the flag after completing execution. We can include multiple functions for 'B', 'C' and 'D' as well.

To use, press 'A' for first function, and 'E' 'A' for second function. Likewise, 'B' and 'E' 'B', 'C' and 'E' 'C', 'D' and 'E' 'D'. If the flag is set by pressing 'E', the next key ('A' to 'D') tests the flag, finds it set, branches to the numbered label, executes the alternate function, and then resets the flag. If the flag is not set, the branch instruction is skipped, and the first function is executed.

Such a program might look like this:

LBL	LBL	LBL	LBL	LBL
A	B	C	D	E
f	f	f	f	f
TF1	TF1	TF1	TF1	SF1
GTO	GTO	GTO	GTO	RTN
0	1	2	3	
first function	first function	first function	first function	
A	B	C	D	
RTN	RTN	RTN	RTN	
LBL	LBL	LBL	LBL	
0	1	2	3	
alternate function	alternate function	alternate function	alternate function	
A	B	C	D	
f^{-1}	f^{-1}	f^{-1}	f^{-1}	
SF1	SF1	SF1	SF1	
RTN	RTN	RTN	RTN	

Altogether, we get eight functions from five keys, **but** – let's look at all the 'housekeeping' steps (labels, Goto, flag tests, sets, resets, returns) we used in the process. Label A used 12 steps, as did B, C and D. Label E used five for a grand total of 53 steps. This leaves us only 47 steps for programming eight functions. If we default Label A by omitting the first two steps, we gain two more steps; but there has to be a better way !

DSZ

Another method takes advantage of the HP-65's built-in decrement feature (gDSZ). In a manner similar to flag testing, we substitute the gDSZ instruction which subtracts 1 from Register 8 (R8) and then skips the following two steps if the result in R8 is zero. If we store a 1 in R8 using key 'E', then the gDSZ test will skip. (Decrement, Skip if Zero). The advantage of this method is that nothing needs to be reset at the end of execution, saving steps. A possible disadvantage is the loss of R8 for data storage.

Such a program might look like this:

LBL	LBL	LBL	LBL	LBL
A	B	C	D	E
g	g	g	g	1
DSZ	DSZ	DSZ	DSZ	STO8
GTO	GTO	GTO	GTO	gR↓
0	1	2	3	RTN
alternate function	alternate function	alternate function	alternate function	
A	B	C	D	
RTN	RTN	RTN	RTN	
LBL	LBL	LBL	LBL	
0	1	2	3	
first function	first function	first function	first function	
A	B	C	D	
RTN	RTN	RTN	RTN	

The gR↓ in program 'E' can be eliminated if **none** of the programs require an input in the X register.

Execution of the above program is identical to the previous program ('A', 'E' 'A', etc); however, if we count our 'housekeeping' steps, we fare a little better than before. Labels 'A' through 'D' use 10 steps each and Label 'E' uses six steps for a total of 46 steps. If we default Label A, we use 44 steps. Either way, we save seven steps compared to flag testing.

Input or Calculate (I/C Flags)

Now, let's suppose we want a key to perform two different functions where the first function is very short; namely, store the content of X in a register and stop. A given key can be used to input data or, given other known factors, calculate an unknown.

Such a program using flags might look like this:

LBL	LBL	LBL	LBL	LBL
A	B	C	D	E
f^{-1}	f^{-1}	f^{-1}	f^{-1}	f
TF1	TF1	TF1	TF1	SF1
*STO1	*STO2	*STO3	*STO4	RTN
RTN	RTN	RTN	RTN	
calculate unknown	calculate unknown	calculate unknown	calculate unknown	
A	B	C	D	
f^{-1}	f^{-1}	f^{-1}	f^{-1}	
SF1	SF1	SF1	SF1	
RTN	RTN	RTN	RTN	

To input known data, press 'A' or 'B', etc. To calculate an unknown, press 'E' 'A', or 'E' 'B', etc. In terms of 'housekeeping', we fare still better. Labels A through D each use 8 steps, plus five more for Label E; a total of 37 steps . Again, we can default Label A, thereby using only 35 steps.

* (Not counted).

Input or Calculate gDSZ (I/C DSZ)

We can use the gDSZ method in a similar manner:

LBL	LBL	LBL	LBL	LBL
A	B	C	D	E
g	g	g	g	1
DSZ	DSZ	DSZ	DSZ	STO8
*STO1	*STO2	*STO3	*STO4	RTN
RTN	RTN	RTN	RTN	
calculate	calculate	calculate	calculate	
unknown	unknown	unknown	unknown	
A	B	C	D	
RTN	RTN	RTN	RTN	

Data is input or calculated in the same manner as the previous flag test method but, since there is no flag to unset, eight more steps are saved. 'Housekeeping' is now reduced to 29 steps. Again, if we default Label 'A', we cut 'housekeeping' to 27 steps.

* (Not counted).

Since the above program does not normally need any specific data in X at the time 'E' is pressed, a gR↓ step is not required in LBL 'E'.

Input or Calculate Comparative Tests

Comparative tests can also be used in a program to decide whether data is an input or not. Most commonly, zero is used as a reference. As an example:

LBL	LBL	LBL	LBL	LBL
A	B	C	D	E
*STO1	*STO2	*STO3	*STO4	*STO5
0	0	0	0	0
gX≠Y	gX≠Y	gX≠Y	gX≠Y	gX≠Y
RTN	RTN	RTN	RTN	RTN
g NOP	g NOP	g NOP	g NOP	g NOP
calculate	calculate	calculate	calculate	calculate
unknown	unknown	unknown	unknown	unknown
A	B	C	D	E
RTN	RTN	RTN	RTN	RTN

This program is convenient to use because a zero in X at the time A or B through E is pressed causes calculation of that function. Since the comparison test places 0 in X after each data entry (if data is not entered before pressing the next letterd key), that function is automatically calculated.

* (Not counted).

We also get an additional bonus: Key E is **not** used to control the calculator function, so we can use Label E for another function. For comparison purposes we will count the steps in A through to D only, a total of 28 steps. Again, a default label will reduce this to 26 steps, one less than the gDSZ approach. In addition, if Label E is needed for other purposes, it is available. One thing to note is that this approach assumes zero is never used as a valid entry.

Input or Calculate Polarity Tests

Another variation of comparison testing is to test for polarity (positive or negative value). Assuming all real data values are positive, data with a negative sign could be used to indicate that an alternate function is desired. Although more 'housekeeping' steps are used in this approach, for some applications it may be most ideal. Such a program might look like this:

LBL	LBL	LBL	LBL	LBL
A	B	C	D	E
ENT↑	ENT↑	ENT↑	ENT↑	ENT↑
g	g	g	g	g
ABS	ABS	ABS	ABS	ABS
gX≠Y	gX≠Y	gX≠Y	gX≠Y	gX≠Y
GTO	GTO	GTO	GTO	GTO
1	2	3	4	5
positive	positive	positive	positive	positive
function	function	function	function	function
A	B	C	D	E
RTN	RTN	RTN	RTN	RTN
LBL	LBL	LBL	LBL	LBL
1	2	3	4	5
negative	negative	negative	negative	negative
function	function	function	function	function
A	B	C	D	E
RTN	RTN	RTN	RTN	RTN

Note that the use of the ABS value always converts X to a positive value, eliminating the need to change sign later if X was negative.

Comparing this program with others (for 8 functions only) this approach uses 12 steps per label for a total of 48 steps. Not too efficient ! We can get some relief if we are willing to use Label E as a subroutine, thus:

LBL	LBL	LBL	LBL	LBL
A	B	C	D	E
E	E	E	E	ENT↑
gX≠Y	gX≠Y	gX≠Y	gX≠Y	g
GTO	GTO	GTO	GTO	ABS
1	2	3	4	RTN
positive	positive	positive	positive	
routine	routine	routine	routine	
RTN	RTN	RTN	RTN	
LBL	LBL	LBL	LBL	
1	2	3	4	
negative	negative	negative	negative	
routine	routine	routine	routine	
RTN	RTN	RTN	RTN	

Now only 46 steps are used which, by defaulting Label A, can be reduced to 44 steps.

Another program variation of polarity testing is used in Standard Pac 05A to determine whether input data is to be converted to another form before storing. Study the program listing in the back of your Standard Pac Manual. The method may be just what you need for that knotty program you have been working on.

The following table summarises the above methods:

Methods	Functions	Minimum Housekeeping
Flags	8	51
DSZ	8	44
I/C FLAGS	8	35
I/C DSZ	8	27
I/C COMPARE	8, 10	26, 33
POLARITY	8	44

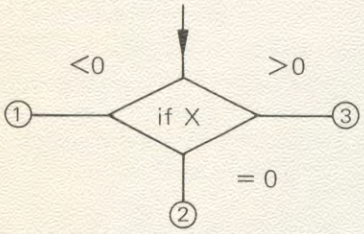
SPLITTING MERGED CODES

Dean Lampman, an HP-65 Users' Club member was examining the logic of the HP-65 in order to write a program in BASIC to simulate the HP-65 using a larger computer. During the logic study and program execution, it was found that, except for RTN and SST, all keys in Rows 2 and 3 on the keyboard can have their instructions and addresses separated by any number of logical compare instructions !

The following instructions may be split:

DSP	f^{-1}	STO +
GTO	STO	STO −
LBL ?	RCL	STO x
f	g	STO ÷

To illustrate this new 'instruction splitting' logic, consider a typical 'if' statement common to larger computers. The logic flow would be:



X negative, GTO 1
X zero, GTO 2
X positive, GTO 3

The concept of splitting merged codes to allow a form of 'indirect addressing' was first reported by Neil Murphy.

The applications of this technique are so varied that many practical applications are yet to be found. A few routines will serve as examples of how this new logic may be applied.

Three-Way Branch — *8 Steps*

This three-way branch compares the X register with zero and may be executed by:

1. 0	00	
2. gX=Y	35 07	
3. GTO	22	
4. gX≠Y	35 21)	The GTO instruction and its address is
5. gX>Y	35 24)	split by two logical compares !
6. 3	03	
7. 2	02	
8. 1	01	

Any number could be entered prior to Step 2.

Three-Way Branch — *6 Steps*

This routine is well suited for iterative problems where two consecutive answers (in X&Y) represent maximum accuracy and the program should stop. Answers above or below are iterated according to the undershoot or overshoot status.

1. GTO	22	
2. gX≠Y	35 07	
3, gX>Y	34 24	LBL 1 is undershoot routine.
4. 2	02	LBL 2 is overshoot routine.
5. R/S	84	
6. 1	01	

Note that the GTO instruction is ignored when the R/S is executed.

Three-Way Store — *6 Steps*

This routine will store in registers R1, R2 or R3 depending upon X and Y compares:

1. STO	33	
2. gX≠Y	35 21	The table below summarizes the logic.
3. gX>Y	35 24	Multiple addresses often cause un-
4. 3	03	wanted numbers in the display. Study
5. 2	02	the logic to see why.
6. 1	01	

Condition	Action	X Register	Y Register
X<Y	X stored in R1	unchanged	unchanged
X=Y	X stored in R2	1 (step 6)	unchanged
X>Y	X stored in R3	21 (steps 5&6)	X

Three-Way Store, X & Y Unchanged — *8 Steps*

1. STO	33	
2. gX≠Y	35 21	
3. gX>Y	35 24	This routine is the same as the one
4. 3	03	above except the X&Y registers are
5. 2	02	unchanged. Any digit may be used for
6. 1	01	Step 7.
7. 0	00	
8. R↓	35 08	

Answer Determines Scientific or Fixed Display — *5 Steps*

Based on a logical compare of X&Y, this routine will display the X register in scientific or fixed mode.

1. DSP	21	
2. gX=Y	35 23	if X=Y display scientific.
3. DSP	21	if X≠Y display fixed.
4. •	83	
5. n		(n is number of decimal places)

Root or Power — *5 Steps*

Using Y^x, the root is taken if X≠Y, otherwise the power n^n is calculated.

1. g	35
2. gX≠Y	35 21
3. 1/x	04
4. g	35
5. Y^x	05

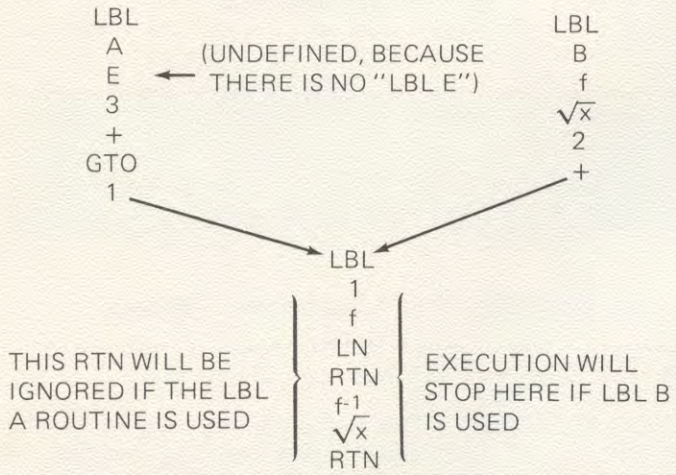
Sin or Cos. — *5 Steps*

If X=Y, compute sin of X; if X≠Y, compute cos of X.

1. g	31
2. gX=Y	35 23
3. SIN	04
4. R/S	84
5. COS	05

UNDEFINED SUBROUTINE CALL

This tip can be applied to save many steps in a long program: If an **undefined** subroutine (say, E) is called in a running program, then E will be ignored and the first RTN encountered will act like an NOP. For example, consider this problem: $(LN\ (x+3))^2$ or $LN\ (\sqrt{x}+2)$. It would look like this:



Now, when A is pressed, the pointer moves down through the program, ignores the first RTN, and stops at the **second** RTN. When B is pressed, the pointer stops at the **first** RTN.

There is, however, one caution: this will not work if a **defined** subroutine is called before a RTN is encountered.

DISPLAY POWER CONSUMPTION

Economy Considerations

Some users have expressed interest in the power consumption of the HP-65 with regard to battery life. The following measurements were made using a fully-charged battery with a terminal voltage of 3.82 volts. Measurements were made with a Simpson 260 volt-ohm meter.

DISPLAY	CURRENT
.	130 mA
12 8's	180 mA
Executing Program	150 mA (approx.)
Reading Program Card	275 mA

The digit 8 requires all seven segments on the display, and with all the 8's displayed the added current is 50 mA for the 84 segments, or approximately 620 μA per segment. One way of saving power is to press the decimal point. Relative increases in consumption for various operations are shown below, using the decimal point current as reference:

DISPLAY	% INCREASE
0.00	8.6
All 8's	40.1
Executing Program	15.4

These measurements, although approximate, do indicate that the standby power of the HP-65 is relatively high. Unless the display is at maximum consumption (all the 8's), the added power consumption due to the display alone is a small percentage of the total.

What is the trade-off between a machine at idle (decimal point only on display) against turning the machine off and then on again before re-entering a program card? The energy required to read a program card in 1.2 seconds is 2.75 x 3.82 x 1.2 = 1.25 watt seconds. To equal the same power energy, idling would require $\frac{1.25}{0.13 \times 3.82} = 2.52$ seconds.

Considering the peak currents involved in starting the motor, errors in measurement, etc., a good guide would be: if your machine is going to sit idle for more than 5 to 10 seconds, power is saved by turning it off, then on again, before re-entering the program!

NOT AN ERROR

Several members have reported what at first appears to be an error in the HP-65 operation. The apparent malfunction occurs when a polar-to-rectangular (f^{-1}, R→P) function is executed during a program. For angle values of ± 90°, and magnitude values less than Unity (1), the program halts. This experience can be disturbing because a completely debugged program which may have run without problems, suddenly stops with zero in the display. Corrective action: Press R/S.

This condition results because the X coordinate is zero and the calculator cannot handle numbers smaller than 10^{-99}, which is close to zero, but is not zero. This underflow condition is recognized by the calculator, and execution is halted as an indication to the user that an underflow condition exists.

To verify the above, run the following routine:

1.	LBL	23	
2.	E	15	
3.	f^{-1}	32	Converts Polar
4.	R→P	01	to Rectangular.
5.	f	31	* Converts Rectangular
6.	R→P	01	back to Polar.
7.	R/S	84	Displays magnitude.
8.	gX⇌Y	35 07	
9.	R/S	84	Displays Angle.

* (Halts if ± 90° and magnitude < 1).

Examples:

Case 1. 90 ↑ 1.5 'E'
Case 2. 90 ↑ 0.8 'E'

The first case will run through to completion, and the correct 1.5 will be in the display. R/S provides the correct angle. The second case will halt after instruction f^{-1} and R→P. Pressing R/S yields the correct 0.8; R/S again, the correct 90°

HOW TO MAKE AN HP-65 TALK

In its June 1975 issue, "Electronics Engineering" described how two Italian HP-65 users obtained X register output in audio form:

"The HP-65 pocket programmable calculator has no built-in provision to interface audibly with the outside world. However, its display generates a modulated rf noise that depends on the number displayed. A simple AM transistor radio receiver can pick up this noise and transform it into an audio signal.

In order to simplify recovery, a routine has been developed to convert a number, digit by digit, into a sequence of bursts of noise pulses. A burst is obtained by putting the number 8 888 888 888 in the X register and giving ten NOP instructions.

After being band-pass filtered at about 50 Hz, and then rectified and integrated, the signal is applied to an amplitude discriminator which has an appropriate threshold and a logic-level output.

With the code, one burst is given for digit 0, two bursts for 1, three for 2, and so on. The least significant digit is transmitted first. If scientific notation is used, a series of zeroes, corresponding to the exponent, is transmitted before the significant digits. The time interval between two consecutive digits is longer than that between the bursts signalling a single digit. A limitation of the program is that the sign and the decimal part of a number is ignored.

The conversion routine occupies 63 steps of the program memory, leaving 37 steps for a processing program. The processing program cannot use LBL 1, LBL 2 and LBL A. The STO 1, STO 2 and STO 8 registers can be used, but they are altered by transmission of the data. The processing program must end with the number in the X register; the last instruction must be A. If manual operation is used, the X register content is transmitted after Key A is pressed.

If the results of a calculation are expressed by more than one number, but less than eight, it is possible to store these numbers in STO 3, 4, 5, 6, 7 and 9, and in one of the STO 1, 2 or 8. The number stored in the last of these, if used, must be transmitted first.

If the processing program has more than 37 instructions, it can be recorded on a separate program card. After the process is completed, the resulting numbers must be stored, observing the above limitation, and the transmission routine card must be introduced. The 37 free steps can allocate a second routine which recalls the results one by one and supplies them to the transmission routine.

1.	LBL	23	22.	STO 1	33 01	43.	8	08
2.	A	11	23.	RCL 2	34 02	44.	DSP	21
3.	f	31	24.	f^{-1}	32.	45.	·	83
4.	INT	83	25.	INT	83	46.	0	00
5.	g	35	26.	1	01	47.	g NOP	35 01
6.	ABS	06	27.	0	00	48.	g NOP	35 01
7.	STO 1	33 01	28.	x	71	49.	g NOP	35 01
8.	LBL	23	29.	1	01	50.	g NOP	35 01
9.	1	01	30.	+	61	51.	g NOP	35 01
10.	RCL 1	34 01	31.	STO 8	33 08	52.	g NOP	35 01
11.	1	01	32.	LBL	23	53.	g NOP	35 01
12.	0	00	33.	2	02	54.	g NOP	35 01
13.	÷	81	34.	8	08	55.	g NOP	35 01
14.	0	00	35.	8	08	56.	g NOP	35 01
15.	g x=y	35 23	36.	8	08	57.	CLX	44
16.	RTN	24	37.	8	08	58.	g	35
17.	g NOP	35 01	38.	8	08	59.	DSZ	83
18.	g R↓	35 08	39.	8	08	60.	GTO	22
19.	STO 2	33 02	40.	8	08	61.	2	02
20.	f	31	41.	8	08	62.	GTO	22
21.	INT	83	42.	8	08	63.	1	01

Gianfranco Sinigaglia and Paolo Sinigaglia, University Physics Institute, 46 Irnerio, 40126 Bologna, Italy."

The fully documented program is now available through the Users' Library under reference No. 50416A.

PRIVATE HP-65 USERS CLUBS

Last year, some enthusiastic HP-65 owners in the United States formed their own club. Mostly Users' Library members, these enthusiasts exchange knowledge and experience in a casual way through Richard Nelson's monthly newsletter called "65 Notes".

A no less enthusiastic member of the HP-65 European Users' Library has now started a German Users Club. Heinrich Schnepf, from Köln, publishes a monthly newsletter in German called "Display" which he circulates to members in return for a subscription fee to cover his operating costs. If you are interested in becoming a member, please write to:

Heinrich Schnepf
Noldeweg 2
D – 5000 Köln.

Although this club is entierely independent of the Hewlett-Packard company, we very much applaud its aims and wish it every success.

APPLICATION PAC NEWS

To extend the all-round capability of the HP-65, we have developed the following three new Application Pacs for Statistics, Microwave Circuit Design, and Mechanical Engineering.

Machine Design Pac 1 *(00065-67052)*

Thirty-five programs (40 prerecorded cards) provide solutions for the machine designer in dynamics, vibrations, linkages, cams, gears, springs, power transmission, and machine geometrics.

Stat Pac 2 *(00065-67053)*

Thirty-one programs (40 prerecorded cards) provide solutions in the areas of general statistics, distribution functions, curve fitting, analysis of variance, test statistics, probability, quality control, and queing theory.

E.E. Pac 2 *(00065-67056)*

Twenty-seven programs (40 prerecorded cards) will assist the microwave circuit designer in making microwave measurements, designing transistor amplifiers, computing transmission line properties and certain system properties, and performing difficult related mathematical operations.

These new Application Pacs bring the total available for your HP-65 to 15, covering such diverse subjects as finance, mathematics, statistics, electrical engineering, chemical engineering, stress analysis, numerical control, surveying, medecine, aviation and marine navigation. Most Pacs are also available in French, German and Italian. For more details, please contact your local Hewlett-Packard Sales Office or Dealer.

TWO NEW MIRACLE BABIES JOIN THE HP FIRST FAMILY

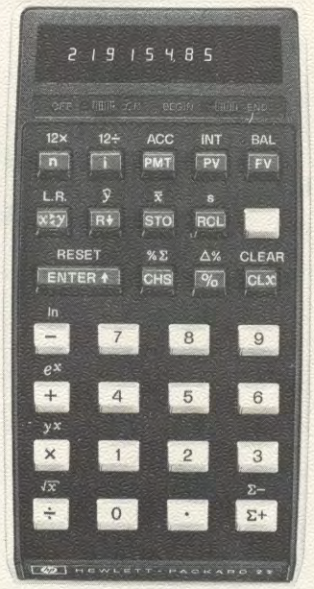
The secret of all HP's small miracles is the keys, and the array of special functions they perform. Like the rest of the family, these two new hand-held calculators feature HP's advanced design, 4-register stack, and RPN logic system. The small miracle is that each leads its class in power, weighs a mere 6 ounces, and is moderate in cost.

HP-25 Scientific Programmable Calculator

Preprogrammed to solve 72 scientific, engineering, and mathematical functions with keystroke ease, the HP-25 has the power you need to solve technical problems.

But that's just the beginning. Its comprehensive keystroke programming capability lets you solve repetitive and iterative problems — even some complicated ones that force you to wait in line for computer time — at your desk, in minutes or even seconds. The HP-25's programming power includes a 49-step memory; direct and conditional branching using as many as 8 relational tests; integer/fraction truncation and absolute value keys; and a full editing capability that lets you see what your program looks like, step by step, and lets you look at intermediate results by means of controlled pauses in the program.

The new HP-25 also has 8 addressable memories, each capable of register arithmetic. And it's the first hand-held machine to offer not only fixed and scientific but also engineering notation (i.e., exponent displayed as a multiple of ± 3, as in giga and nano).



HP-22 Business Management Calculator

Combining the preprogrammed power of four present-day calculators, this new model solves virtually every number problem that a business manager faces. Its easy-to-use keyboard includes:

* All typical commercial functions such as percent, mark-ups and discount, percent difference and percent of total.
* A complete range of financial keys for evaluating time-and-money problems, simple and compound interest, internal rate of return, ordinary annuity and annuity due and loans.
* Advanced statistical functions, linear regression and linear estimate and growth curves.
* A powerful range of mathematical functions such as log, antilog, exponentiation, and root extraction.

The HP-22 gives you the memory you will need for business management calculations. In addition to the 5 financial registers and the 4 operational stack registers, the HP-22 provides 10 addressable registers you can use to store data.

HEWLETT hp PACKARD

See the family of professionnal hand-held calculators at your local HP Dealer or contact the nearest HP Sales Office.

HP-65 Users' News is published periodically by HEWLETT-PACKARD S.A.,
HP-65 Users' Library-Europe
7, rue du Bois-du-Lan,
P.O. Box
1217 MEYRIN 2
Switzerland

Readers' comments or contributions will be most welcome. Don't hesitate to drop a line to the editor at the above address.