Don Rindsberg
The Bit Stop
PO Box 973
Mobile AL 36601

Here's HUEY!

. . . super calculator for the 6502

What hobbyist wouldn't like to have available the calculating power of FORTRAN in his system? Here is HUEY, written for the 6502 microprocessor, which does arithmetic with precision better than an IBM 360/370 or Univac 1108, unless they pull a dirty trick and switch to double-precision mode. What's more, HUEY (please don't call him Hewlett) operates from your ASCII keyboard like a calculator; will output through your routines to a TV screen or Teletype; is preprogrammed to do trig functions, natural and common logs, exponential functions and other goodies; and is programmable for many other functions (financial, accounting, mathematics, engineering, etc.) you would like to call at the press of a single key. Further, the routines can be called as subroutines by your own programs if you need the precision or the functions HUEY provides.

All this is contained in 2.5K of memory, with optional expansion to 3K by addition of your own functions.

The complete hexadecimal listing of the program is given in Table 1. The basic program occupies addresses 1000 through 19FF, but you will wish to reserve 1AOO through 1BFF if you intend to add other functions. The program uses page zero extensively for arithmetic registers and memory registers, and you should avoid using addresses 0020 through 009F for your video routines or other programs. The program itself sets all these locations at the values it wants, so there is no extensive entry into page zero required before the program can be used. Arithmetic overflow, division by zero, logs of negative numbers and other no-nos divert the program to software breaks. If you set your IRQ vector to go to address 1164, a break will give you an error code on your display. This code, shown in Table 2, is simply the address of the break, plus two. The error code will be followed by a string of zeros on the next line.

Entering the Program

For ease of entry, all of the basic user options in the program are contained in the first few bytes of the listing. Program A will allow you to select the options and provide entry to your input/output routines. Although the accesses to your input and output routines appear to be jumps, in all cases they originate deep within the inner workings of the program as jumps to subroutines. So, when your input or output routine is through doing its thing, however long it takes, it must end with op code 60 (return from subroutine). Just to be safe, set Y to zero just before returning, although I thought this kind of bug was exterminated prior to printing.

The request by the program to output a character is, in all cases, preceded by instructions within the program to place the ASCII character in the accumulator of the microprocessor. Similarly, when it asks for an input, it expects its ASCII equivalent to be in the accumulator. Your routines should restore the stack pointer, since the stack has what your mail should have on the upper left corner — namely, the return address. If you have Tiny BASIC, identical input/output routines can be used.

Press Go, Start, or Whatever

Your initialization routines can do whatever you like (clear the TV screen, set the IRQ vector or turn on the coffee), but when they are finished, there should be a

1006	4C,xx,yy	A jump to your input routine. xx is ADL, yy is ADH.
1009	4C,xx,yy	A jump to your output routine.
100C	5C	Choice of default character.
100D	2A	Choice of exponent symbol.
100E	09	Number of digits entered or displayed. Enter number between 02 and OD.
100F	08	Choice of back space code.
1010	00	Expansion.
1011	01	Delay time for typewriter carriage return. Use 01 for shortest time. FF for longest. FF gives 0.3 sec for 1 MHz clock.
1012	1B	Character used to TAB to exponent.
1013	00	Expansion.

Program A.

Code	Description
14F9	Overflow in Exponent calculation.
157D	Square root of negative number.
15C7	Natural log of negative number or zero.
16E6	Floating point overflow (number too large).
176F	Division by zero.

Table 2. Error codes.

```
1000  4C 14 10 4C 1B 10 4C 00 00 4C 00 00 5C 2A 09 08
1010  18 01 1B 00 A0 80 A2 20 20 94 13 A2 FF 9A D8 A9
1020  20 85 90 A9 60 85 93 20 07 17 20 56 17 20 73 17
1030  A0 00 84 9F A9 81 85 9E EA EA EA EA EA A9 3A 20
1040  09 10 20 06 10 A0 00 D8 29 7F CD 0F 10 F0 4A CD
1050  12 10 D0 03 4C E6 10 C9 2A 90 E7 C9 5B B0 E3 C9
1060  30 90 3E C9 3A B0 3D E9 2F 85 9D A5 9E C9 81 D0
1070  03 20 56 17 A5 9E 38 ED 0E 10 C9 83 F0 1D C9 80
1080  F0 11 A5 9E C9 81 38 D0 01 18 20 00 17 B0 B3 A9
1090  2E 90 AC 20 D2 17 38 B0 F1 C6 9E AE 0C 10 4C 93
10A0  11 18 69 0A 85 9D 20 34 17 20 07 17 A0 00 84 4E
10B0  A9 18 85 4F 20 00 11 A6 9D BD CC 14 AA 20 CE 10
10C0  A9 08 85 4E A9 18 85 4F 20 00 11 4C 30 10 29 03
10D0  18 69 18 85 4F 8A 29 FC 85 4E 4C 00 11 E6 4E D0
10E0  02 E6 4F 60 00 00 A9 81 18 6D 0E 10 C5 9E D0 06
10F0  20 D2 17 4C 42 10 A9 30 20 09 10 E6 9E D0 E7 EA

1100  A0 00 B1 4E F0 0C 20 13 11 38 E6 4E D0 02 E6 4F
1110  B0 EE 60 08 48 29 7E AA BD 00 12 85 91 BD 01 12
1120  85 92 68 28 30 06 4A B0 06 4C 90 00 4C AE 12 4C
1130  D0 12 A5 20 C9 60 B0 0B A9 F2 85 91 A9 12 85 92
1140  20 AE 12 60 A5 20 F0 03 4C 03 13 A0 10 A2 70 4C
1150  94 13 A5 72 D0 0B A2 73 D0 07 A2 20 A0 07 4C 94
1160  13 4C 03 14 68 68 85 9A 68 20 74 11 A5 9A 20 74
1170  11 4C 1B 10 48 4A 4A 4A 4A 20 89 11 20 09 10 68
1180  29 0F 20 89 11 20 09 10 60 C9 0A B0 03 69 30 60
1190  69 36 60 98 91 9E 8A EA EA 4C 3F 10 00 00 00 00
11A0  00 00 00 00 00 00 00 00 00 00 00 00 7F 40 00 00
11B0  00 00 00 82 40 00 00 00 00 00 7F 58 B9 0B FB E8
11C0  E6 7E 6F 2D EC 54 9B 92 80 5A 82 79 99 FC E4 80
11D0  52 B0 3D 80 00 00 81 AB 86 49 10 00 00 80 6A 08
11E0  65 10 00 00 86 57 6A E0 FF 0A 80 89 4D 3F 1D 00
11F0  00 00 7B 46 FA 70 00 00 00 83 4F A3 03 4D 00 00

1200  00 00 50 16 4A 16 77 16 66 17 77 15 85 15 73 17
1210  1C 16 80 12 BF 15 D5 14 F8 14 00 14 00 13 2C 16
1220  E8 16 55 15 5B 15 6A 15 DE 12 96 12 07 17 56 17
1230  32 11 DE 17 FA 17 00 00 00 00 00 00 00 00 00 00
1240  00 00 00 00 00 00 00 00 2A 15 AC 11 B3 11 37 00
1250  47 00 57 00 20 00 F2 12 F9 12 C0 14 C7 14 CE 14
1260  95 12 31 15 38 15 3F 15 46 15 4D 15 BA 11 C1 11
1270  C8 11 CF 11 D6 11 DD 11 E4 11 EB 11 F2 11 F9 11
1280  A2 06 B5 40 95 30 B5 50 95 40 B5 60 95 50 B5 67
1290  95 60 CA 10 ED 60 A2 06 B5 20 B4 30 95 30 B5 40
12A0  94 40 B4 50 95 50 94 20 CA 10 ED A0 00 60 A2 06
12B0  A0 06 B5 60 95 67 B5 50 95 60 B5 40 95 50 B5 30
12C0  95 40 B5 20 95 30 B1 91 95 20 88 CA 10 E4 C8 60
12D0  A2 26 A0 06 B5 00 91 91 CA 88 10 F8 C8 60 A2 06
12E0  B5 57 B4 20 95 20 94 57 CA 10 F5 60 00 00 00 00
12F0  00 00 00 00 00 00 00 00 00 00 80 40 00 00 00 00

1300  4C 44 11 EA EA EA A0 00 84 70 A5 21 10 05 C6 70
1310  20 8F 16 A0 00 84 71 A5 20 C9 7F F0 32 90 13 20
1320  1C 16 20 B4 13 20 9D 16 A5 71 20 A8 13 85 71 B8
1330  50 E5 C9 7A 90 06 20 5F 16 B8 50 DB 20 1C 16 20
1340  B4 13 20 77 16 A5 71 20 AE 13 85 71 B8 50 C8 A2
1350  72 20 92 13 A2 82 20 92 13 A9 05 85 82 A2 26 A0
1360  06 20 9D 13 A9 30 85 5F A2 26 A0 06 20 9D 13 C6
1370  5F F0 11 90 07 A2 7F A0 8F 20 C4 13 A2 82 20 E2
1380  13 B8 50 E4 A5 71 20 AE 13 85 71 EA EA EA EA EA
1390  EA 60 A0 0E A9 00 95 00 E8 88 D0 FA 60 16 00 CA
13A0  88 36 00 CA 88 D0 FA 60 18 F8 69 01 D8 60 38 F8
13B0  E9 01 D8 60 A2 20 A0 07 20 94 13 A9 83 85 20 A9
13C0  50 85 21 60 A9 0E 85 6F F8 18 B9 00 00 75 00 48
13D0  29 0F 95 00 68 29 10 F0 01 38 CA 88 C6 6F D0 EA
13E0  D8 60 D8 A9 0D 85 6F 56 00 90 08 E8 A9 09 75 00
13F0  95 00 CA E8 C6 6F D0 EF EA EA 60 00 00 00 00 00

1400  4C 52 11 EA EA EA A5 71 20 A8 13 85 71 A2 20 A0
1410  07 20 94 13 A9 80 85 20 4A 85 21 A5 71 A8 F0 23
1420  18 F8 69 50 C9 50 D8 90 0D 98 20 AE 13 20 99 14
1430  20 77 16 B8 50 E5 98 20 A8 13 20 99 14 20 9D 16
1440  B8 50 D8 20 1C 16 A2 20 A0 07 20 94 13 A2 67 A0
1450  07 20 94 13 A9 40 85 68 A9 2F 85 5E 20 A2 14 90
1460  0B A2 06 B5 67 15 20 95 20 CA D0 F7 18 A2 FA A9
1470  80 B0 01 0A 57 6E 15 6E 95 6E E8 D0 F2 C6 5E D0
1480  DB A9 7F 85 20 A5 70 30 0A 20 3E 16 20 77 16 EA
1490  EA EA 60 20 8F 16 B8 50 F3 85 71 20 1C 16 20 B4
14A0  13 60 A2 7F A0 0E F8 18 B5 00 75 00 48 29 0F 95
14B0  00 68 29 10 18 F0 01 38 CA 88 D0 EC D8 60 00 00
14C0  7F 80 00 00 00 00 00 81 40 00 00 00 00 00 81 60
14D0  00 00 00 00 00 A5 22 85 2F 38 E9 7C A5 21 E9 00
14E0  10 15 18 A5 22 69 78 A5 21 69 00 10 09 A9 00 A2
14F0  06 95 20 CA 10 FB 60 00 38 A5 2F 65 20 85 20 60

1500  1C 64 60 70 60 28 60 60 3C 60 38 B1 34 6D 60 55
1510  60 B0 60 65 60 48 60 45 80 60 50 60 14 E0 7C 01
1520  75 40 44 24 58 60 60 60 60 60 80 40 00 00 00 00
1530  20 82 50 00 00 00 00 00 82 90 00 00 00 00 00 83
1540  50 00 00 00 00 00 81 56 FC 2A 2C 51 5E 81 64 87
1550  ED 51 10 B2 00 A9 09 EA 85 97 60 A5 4E 85 95 A5
1560  4F 85 96 60 00 00 00 00 00 00 C6 97 F0 08 A5 95
1570  85 4E A5 96 85 4F 60 A5 21 10 01 00 A5 20 85 98
1580  A9 80 85 20 60 A9 00 85 99 46 98 90 02 E6 99 06
1590  98 A5 98 10 09 38 E9 80 4A 18 69 80 90 09 A9 80
15A0  38 E5 98 4A 18 65 98 85 20 A5 99 F0 11 A9 C8 85
15B0  91 A9 11 85 92 20 AE 12 20 77 16 20 80 12 60 A5
15C0  21 F0 02 10 01 00 20 1C 16 A2 80 46 3E A5 30 30
15D0  07 86 3E 38 A9 00 E5 30 86 30 38 E9 80 85 22 A9
15E0  00 85 21 20 2C 16 06 3E 90 0B A2 05 A9 00 F5 21
15F0  95 21 CA 10 F7 60 20 94 13 A9 8E 85 20 4C 3E 16

1600  18 A2 05 B5 21 75 31 95 21 CA 10 F7 60 06 94 20
1610  12 16 24 21 10 05 20 8F 16 E6 94 38 A2 07 94 26
1620  B5 1F B4 2F 94 1F 95 2F CA D0 F3 60 A2 23 A0 07
1630  4C F6 15 C6 20 06 26 A2 05 36 20 CA D0 FB A5 21
1640  0A 45 21 30 04 A5 20 D0 EA 60 20 8F 16 20 5D 16
1650  A5 30 C5 20 D0 F7 20 00 16 50 E3 70 05 90 BD A5
1660  21 0A E6 20 F0 7E A2 F4 A9 80 B0 01 0A 56 2D 15
1670  2D 95 2D E8 D0 F2 60 20 0D 16 65 20 20 CD 16 18
1680  20 66 16 90 03 20 00 16 88 10 F5 46 94 90 AF 38
1690  A2 06 A9 00 F5 20 95 20 CA D0 F7 F0 BC 20 0D 16
16A0  E5 20 20 CD 16 38 A2 05 B5 31 F5 27 48 CA 10 F8
16B0  A2 FA 68 90 02 95 37 E8 D0 F8 A2 06 36 20 CA D0
15C0  FB 06 36 A2 05 20 F8 16 88 D0 DA F0 BE 08 20 EF
16D0  16 28 EA B0 0D 30 04 68 68 90 B2 49 80 85 20 A0
16E0  2F 60 10 F7 00 20 5F 16 A5 20 C9 8E D0 F7 60 A0
16F0  06 96 20 88 D0 FB 60 EA 36 30 CA D0 FB B0 E5 60

1700  A5 9D 91 9E E6 9E 60 A9 0D 20 09 10 A9 0A 20 09
1710  10 AE 11 10 88 D0 FD CA D0 FA 60 EA EA EA EA A2
1720  81 A0 72 AD 0E 10 85 9B B5 00 99 00 00 E8 C8 C6
1730  9B D0 F5 60 20 1F 17 AE 0E 10 B5 81 0A 0A 0A 0A
1740  15 82 85 71 A5 80 85 70 B5 81 10 08 F8 38 A9 00
1750  E5 71 85 71 D8 60 A2 70 A0 20 20 94 13 60 EA EA
1760  09 30 20 09 10 60 A5 20 F0 03 4C 9D 16 00 00 00
1770  00 00 00 A9 20 A6 70 10 02 A9 2D 20 09 10 A5 72
1780  20 60 17 A9 2E 20 09 10 AE 0E 10 CA 86 3E A2 00
1790  86 3F A6 3F B5 73 20 60 17 E6 3F C6 3E D0 F3 AD
17A0  0D 10 20 09 10 A5 71 C9 50 90 0F A9 2D 20 09 10
17B0  38 F8 A9 00 E5 71 D8 B8 50 07 A9 20 20 09 10 A5
17C0  71 48 4A 4A 4A 4A 20 60 17 68 29 0F 20 60 17 4C
17D0  07 17 AD 0D 10 20 09 10 A9 20 20 09 10 60 20 DD
17E0  10 A0 00 B1 4E AA A5 4E 48 A5 4F 48 8A 20 CE 10
17F0  68 85 4F 68 85 4E 60 00 00 00 A9 03 85 97 60 00

1800  00 00 00 00 00 00 00 00 53 C8 06 1C 0E 12 28 00
1810  00 00 00 00 D4 D4 1A 12 00 00 00 00 D4 D4 12 12
1820  06 12 00 00 EE 00 00 00 D4 D4 12 12 08 12 00 00
1830  00 00 00 00 2C 2E 00 00 4F 00 00 00 CE 00 00 00
1840  EA 00 00 00 E8 00 00 00 D8 10 08 12 00 00 00 00
1850  D4 D4 1A 12 DA 06 12 00 10 00 00 00 00 00 00 00
1860  00 00 00 00 D4 D4 12 12 02 12 00 00 00 00 00 00
1870  D4 D4 12 12 04 12 30 00 00 00 00 00 2A 00 00 30
1880  D4 14 51 10 53 12 F0 04 12 D2 F0 02 12 08 12 D4
1890  53 06 12 F6 04 12 F4 10 08 12 F2 02 12 D2 06 12
18A0  CA 02 12 D0 02 12 EC 06 12 00 00 00 00 00 00 00
18B0  EC 08 12 D4 20 16 1F 04 12 51 D4 06 12 53 F8 02
18C0  12 FA 10 08 12 FC D2 06 12 10 04 12 FE 02 12 D0
18D0  04 12 D0 10 08 CA 02 12 18 12 00 00 00 00 00 00
18E0  0A 53 10 12 DC 22 24 51 10 12 D2 D0 08 12 D0 02
18F0  12 DC 08 12 26 0C 00 00 00 00 00 00 00 00 00 00

1900  53 D4 06 12 E6 08 12 E6 08 12 E6 08 12 CA 08 12
1910  D8 02 12 D8 10 08 12 E6 06 12 E4 02 12 D2 06 12
1920  DE 08 12 E2 08 12 53 D4 06 12 51 CC 06 12 CC 06
1930  12 CC E2 06 12 60 04 12 D0 06 12 E2 02 12 D2 06
1940  12 00 00 00 34 2A 24 53 C8 06 1C 0E 12 28 2A 26
1950  00 00 00 00 EA DC 08 12 10 04 12 32 01 00 00 00
1960  00 00 00 00 32 80 EE 06 12 00 00 00 EE 08 12 32
1970  B0 00 00 00 32 01 D4 D4 06 12 D8 10 04 12 32 E0
1980  08 12 00 00 00 00 00 00 00 00 00 00 00 00 00 00
1990  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
19A0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
19B0  32 DD 32 DD D4 06 DA 06 53 E2 CC 06 D8 04 51 D8
19C0  22 24 D2 06 D0 08 D0 DC 04 12 51 06 D8 02 26 D2
19D0  DA 06 12 32 E0 06 CC 06 00 00 00 00 D4 D4 06 12
19E0  D8 02 12 32 E0 D8 02 12 08 12 00 00 00 00 00 00
19F0  00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
```

Table 1. Complete hex listings of program.

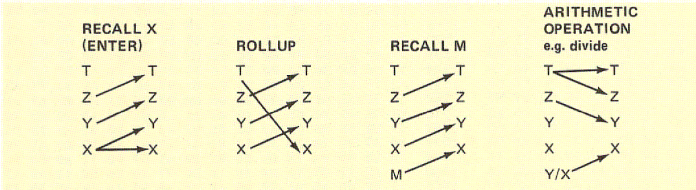

Fig. 1. Stack operations.

0020-26	X-register
0027-2D	E-register
002E	Exponent
002F	Integer
0030-36	Y-register
0037-3D	Memory M1
003E	Temporary used by In and line output
003F	Temporary used by line output
0040-46	Z-register
0047-4D	Memory M2
004E/4F	String pointer
0050-56	T-register
0057-5D	Memory M3
005E	Counter
005F	Counter
0060-66	U-register
0067-6D	V-register and scratch
006E/6F	Temporary for decimal conversion
0070-7F	Main decimal register D
0080-8F	Decimal scratch and line buffer
0090	20 (JSR op code)
0091/92	Address of next routine
0093	60 (RTS op code)
0094	Sign
0095/96	Return address
0097	Counter
0098	Exponent
0099	Flag for SQRT
009A	Temporary used by break routine
009B	Counter
009C	(Expansion)
009D	Last key
009E/9F	Line buffer pointer

Table 3. Page zero assignments.

ADDRESS	DATA	FUNCTION
1526	83	Sets Z key to call address 1B80
1B80	EA	Recall pi
1B81	08	Divide
1B82	12	Move stack down after arithmetic
1B83	32	Set up to call string as subroutine
1B84	EO	Square root string
1B85	00	End

Program B.

jump to HUEY's point of COLD START at address 1000. HUEY will set the stack the way he likes, clear his registers in page zero, output a line of zeroes and output a colon, which is his way of telling you that he's waiting for you to press a key. The screen or typewriter should look like this:

```
0.00000000* 00
:
```

By the way, to keep the program short, I had to limit HUEY's vocabulary to scientific notation. That means all entries should have the decimal point after the first digit, and the number that follows the star is the power of ten that multiplies the first part.

OK, let us determine what 1 divided by 3 is, and we will then go on to more glamorous experiments. Press 1; press P (to enter the *positive* number you have just pressed into the X-register). Your display will now show:

```
:1.P
 1.00000000* 00
:
```

Note that HUEY has forced you to use scientific notation by typing a decimal point immediately following your 1, and has displayed the contents of the X-register. HUEY displays the X-register after every function, just like your $9.95 pocket calculator (at this point you may ask why you paid 100 times this amount for your system, but read on!).

Next, press 3, and then P. The display should now show 3 in our notation. Now, let us divert for one moment to check our work. Let's press K to see what the HUEY stack of four registers contains. The display should be as follows, except that the X, Y, Z and T are not actually displayed.

```
  :K
T  0.00000000* 00
Z  0.00000000* 00
Y  1.00000000* 00
X  3.00000000* 00
  :
```

HUEY has just displayed the X, Y, Z and T registers in reverse order to enable you to examine them more naturally. The entry of 3 into X pushed the previously entered 1 into the Y-register. Now press / to divide Y by X, and the display should show 3.33333333*-01, which means 0.333333333 in ordinary terms, since the minus 01 exponent indicates the decimal point moves one place to the left.

Let's re-examine the stack at this point by pressing K. Note that both dividend and divisor are lost in the arithmetic process, and that Z has moved to Y; in the process T remains unchanged, but is duplicated in Z.

```
:K
 0.00000000* 00
 0.00000000* 00
 0.00000000* 00
 3.33333333*-01
:
```

These features of the stack may or may not have anything to do with HUEY's name. The stack operations, by the way, are outlined in Fig. 1. For clarity of display on a TV screen or typewriter, the ROLL operation has been reversed from that used by hand calculators of similar name. You may load up to four numbers into the stack and then perform your arithmetic functions by combinations of rolls, X/Y exchanges, stores in memory, recalls, interspersed with arithmetic commands; and any time you have lost track of what is in the stack, just press K. All arithmetic operations are done on X and Y, without losing Z and T. There is no need to clear the stack registers; just enter new numbers, and the old ones disappear off the top.

Want to enter an exponent without entering all those zeros? Press ESC (or whatever ASCII key you have selected for this function at 1012) and HUEY will fill in the zeros for you. Negative numbers? Just press N instead of P to enter your number as a negative value. Negative exponents? Subtract your desired negative value from 100 and enter the result. For example, if you want to enter -09 as an exponent, enter 91. HUEY will echo the value he read so you can be sure. In this case, the echoed exponent will read -09. Back

X-register 0020-0026	E-register 0027-002D	Flags, etc.
Y-register 0030-0036	Memory M1 0037-003D	
Z-register 0040-0046	Memory M2 0047-004D	
T-register 0050-0056	Memory M3 0057-005D	
U-register 0060-0066	V-register 0067-006D	
D-register (decimal) 0070-007F		
B Decimal scratch 0080-008F		
Flags, counters, and pointers 0090-009F		

Fig. 2. Page zero memory map.

The following functions are associated with number entry and stack manipulation:

P	Enter as positive number into X
N	Enter as negative number into X
@	Clear entry (use before P or N pressed)
K	Display stack contents
R	Roll the stack up (X to Y, Y to Z, Z to T, T to X)
X	Exchange X and Y

The following operate on X and Y and leave the result in X, with the stack dropping down to fill in; T is duplicated in Z:

+	X added to Y
-	X subtracted from Y
*	X multiplied by Y
/	Y divided by X

The following operate on X, leaving the other stack registers unchanged (arctan does bomb it):

A	Antilog X (base 10)
C	Cos X (radians)
E	Exponent (e raised to the X power)
G	Log X (base 10)
I	Inverse of X (1/X)
L	Natural logarithm of X
Q	Square root of X
S	Sin X (radians)
T	Tan X (radians)
?	Arctan X

The following functions are associated with memory and, in the case of recalls, push the stack up one notch:

Right Arrow	Store X in M
Left Arrow	Recall M into X
U	Recall pi into X (3.141592654)
V	Recall e into X (2.718281828)
W	Recall log e into X (0.434294481)

Table 5. Preprogrammed functions.

KEY	ACTION
Number Keys	Area
P	Enter as positive number
U	Recall pi
/	Divide to get A/pi
Q	Take square root

Example 1.

space works on both the number and the exponent, if you make a mistake, or you can press @ to clear your entry.

Sample Calculation

Suppose you have the area of a circle and want the radius. Simply use the sequence in Example 1.

If you have many values of radius you wish to calculate, you can preprogram, say, the Z key to do the entire operation with one keystroke, after the area is entered, as shown in Program B.

With these modifications to the program, you simply press number keys for the area, press P to enter as a positive number, and then your magic key Z. Repeat for as many calculations as you like. This simple illustration probably does not warrant preprogramming, but it illustrates the power of the system for more complicated calculations.

If you reset your system for any reason, you can re-enter HUEY at address 1003 for WARM START, which will not destroy the contents of the arithmetic registers.

HUEY recognizes, as functions, ASCII entries 2A through 2F (which includes the arithmetic functions) and 3A through 5A (all the upper-case letters and a few punctuation marks). All other ASCII entries are rejected, excepting, of course, the numbers and the special back space and tab functions. Preprogrammed functions are listed in Table 5.

Unused keys can be used for other functions. I have calculated such diverse things as compound interest, hyperbolic functions, 99-term power series and others by adding 12-60 byte programs to page 1A.

Numbers are limited in size to about 1.00000000*37.

The Inner Workings

The arithmetic operations occupying page 16 are a floating point package originally printed in *Dr. Dobb's Journal*, but the package has been modified to use 47-bit arithmetic instead of the original 23-bit. (IBM single precision is 24 bits and Univac is 27.) Our 47 bits gives a precision to arithmetic operations of about 13 equivalent significant figures in decimal. The algorithms for ln, exp. sin, cos, tan and arctan can be counted on for eight-place accuracy, with the trig functions limited to the range 0-90 degrees or 0-pi/2 radians. Square root is performed with accuracy equivalent to the arithmetic operations.

The high precision is, of course, obtained at some sacrifice of speed, but this program is intended for the person who has some serious calculating to do, and not for the game-player who is satisfied with a number system allowing no fractions and having numbers limited to -32K to +32K.

The routines limiting the speed are the conversion of decimal to binary (page 14) and conversion of binary to decimal (page 13). This can be observed by the slowness of entry and display of very large and very small numbers, e.g., 1.00000000*-37.

Memory Registers

Although only one memory (M1) is preprogrammed for keyboard access, two other registers, M2 and M3, are used by the program to store intermediate results. In addition, the stack, which appears to the user to be four registers, is actually six registers X, Y, Z, T, U and V. The two extras are used to simplify restoring the contents of the four "visible" registers. Register E is used by the floating point package. All of these registers are 7-byte binary registers. All arithmetic is done in X, Y and E, and their original contents are stored elsewhere if restoration is desired.

In addition, register D is a 16-byte register that holds a number formatted in decimal, and register B acts as a line buffer and scratchpad. Fig. 2 shows the register locations in page zero. You can set up other registers in page zero if necessary.

Constants, such as pi, 0, 1 and SQRT2, are squeezed into available space throughout the program. Each constant requires seven bytes, and is in binary (hex) form. Other constants you may need, such as the tax rate on incomes over $150,000, can be stored anywhere in your memory, and can be accessed by way of empty slots in the address table. The location and hex values of the preprogrammed constants are shown in Table 4.

Address	Constant	Value (hex)						
12F2	0	00	00	00	00	00	00	00
12F9	1	80	40	00	00	00	00	00
14C0	-1	7F	80	00	00	00	00	00
14C7	2	81	40	00	00	00	00	00
14CE	3	81	60	00	00	00	00	00
1531	5	82	50	00	00	00	00	00
1538	-7	82	90	00	00	00	00	00
153F	10	83	50	00	00	00	00	00
1546	e	81	56	FC	2A	2C	51	5E
154D	pi	81	64	87	ED	51	10	B2
11BA	ln 2	7F	58	B9	0B	FB	E8	E6
11C1	log e	7E	6F	2D	EC	54	9B	92
11C8	SQRT 2	80	5A	82	79	99	FC	E4
11B3	4	82	40	00	00	00	00	00
11AC	0.5	7F	40	00	00	00	00	00
152A	1+	80	40	00	00	00	00	20

Table 4. Preprogrammed constants.

What Happens When You Press a Key?

When a function key is pressed, the ASCII value is used to look up a coded address in the table starting at 1500. The coded address refers to every fourth address in pages 18, 19, 1A and 1B. I have preprogrammed functions in the first two of these pages, leaving pages 1A and 1B for your own functions. At selected addresses in pages 18 and 19 are strings of microinstructions, each string ending in 00 to signal the end of the string. In turn, each microinstruction refers to a subroutine call instruction, a recall memory instruction or a store instruction. If bit 7 of the microinstruction is 1, a memory register is recalled to X. If bit 0 is a 1, then X is stored in a memory register. Otherwise, the microinstruction is decoded as a subroutine call. In all cases, the middle bits (1-6) of the micro refer to the address table occupying the first half of page 12, where the address of the subroutine or memory is picked up. After the current micro is executed, the program picks up the next, until it reaches 00. After the execution of a string of microinstructions, a common output string (located at 1808) is called by the mainline program. This exit string recalls a rounding constant a little bigger than 1, multiplies by X, converts the product to decimal, outputs the scientific format and restores the original contents of the stack. Note that rounding is applied only to the number being displayed and not to any binary registers, so that accuracy is maintained. For the purist, rounding can be easily disabled, but I hate to decode a string of nines!

A common entrance string is called by the mainline program, but it is not required at present and is disabled by the 00 at address 1800.

Because of the unique method of assembly of this program, that is, completely by hand, anyone wishing to reassemble to another memory location should work in whole pages, since some portions must stay in the same relative position on the page. Reassembly to start at address 4000, say, is relatively easy, but reassembly to start at address 4080 would be quite a chore.

A Sophisticated Example

When you have seen what the basic program will do, you may wish to calculate other functions. Let's say you want to preprogram for sinh(X), obtained by using the equation in Example 2.

sinh(X) = (e**X - e**-X)/2

Example 2.

Suppose you wish to use the H key for this function, and you wish to enter the string of microinstructions at address 1A80. At 1514, the H position in the table, enter 82. This 82 is derived from 80, the ADL of the string address, plus 2, the page offset from the base page 18. Now, enter a suitable string of microinstructions such as in Program C to solve your equation.

```
1A80   32   Set up to call a string as subroutine
1A81   BO   Call exponent string to get e**X
1A82   D4   Enter (make a duplicate copy in Y)
1A83   32   Set up to call string as subroutine
1A84   48   Call invert string to get e**-X
1A85   04   Subtract
1A86   12   Move stack down after arithmetic
1A87   DC   Recall number 2
1A88   08   Divide
1A89   12   Move stack down after arithmetic
1A8A   00   End
```

Program C.

With this short string, you can get sinh(X) each time the H key is pressed, or whenever the string is called by another string. Strings can be debugged by replacing any micro with 00 to stop the action at that point, and then pressing K to examine the stack.

A complete manual giving the details of this system, and including a documented listing of the program, is available from The Bit Stop, P.O. Box 973, Mobile AL 36601, for $20 postpaid. Tapes are also available.

Encoding by use of microinstructions makes for a memory-efficient system, since each can call a machine-language subroutine using only one byte. Further, these microinstructions can set up loops to repeat a string as many times as desired, and can even call other complete strings as subroutines. All these features are illustrated in the arctan routine, which is about 60 bytes long, but results in calling some 200 machine-language subroutines before execution is complete.

My thanks go to Felton Mitchell, who got me off my can to write this, and who very kindly dumped off the hard copy. ■