34S OWNER'S MANUAL

TABLE OF CONTENTS

WELCOME

Dear user, you hold in your hands the result of careful customizing. Mechanics and hardware of your 34S are of the new HP-30b Business Professional as is: so you get its unexcelled processor speed and also the famous rotate-and-click keys with tactile feedback as known and appreciated in vintage Hewlett-Packard calculators for decades. On the other hand, firmware and user interface of the 34S are newly designed and written from scratch to give you a **fast and compact <u>scientific</u> calculator like you have never had before**.

Its function set is based on the one of the renowned HP-42S RPN Scientific, the most powerful RPN calculator built so far [1]. We extended this set, incorporating completely the functionality of the famous programmer's calculator HP-16C, the fraction mode of the HP-32SII, probability distributions as featured by the HP-21S, and even **more functions for mathematics, statistics, physics, engineering, programming etc.** like

✚ Euler's Beta function, Fibonacci number calculation, Lambert's W (all these in real and complex domain), incomplete regularized Beta and Gamma, testing for prime numbers,

✚ the error function as well as many statistical distributions and their inverses like Poisson, binomial, geometric as well as exponential, Weibull for reliability analysis, and Gaussian with arbitrary mean and standard deviation,

✚ programmable sums and products,

✚ extended date and time calculations based on a real time clock,

✚ financial operations like mean rate of return or margin calculations,

✚ nearly 50 fundamental physical constants as precise as known today by standards institutes like NIST,

✚ over 60 conversions, dominantly between universal SI and old Imperial units,

✚ complete Greek and extended Latin letter fonts covering many languages (upper and lower case in two font sizes each).

The 34S is the first RPN calculator overcoming the limits of a 4-level stack – forget worries about stack overflow in calculations. It features a choice of two stack sizes expanded by a complex LASTx register: traditional 4 stack levels for HP compatibility, 8 levels for convenient calculations in complex domain, for more advanced real formulas, or for whatever application you have in your mind. You get a full command set for navigation in either size. Furthermore, the 34S features over 100 general purpose registers, 100 user flags, 476 program steps, 3 programmable hotkeys for your favourite programs, and a 31 byte alpha register for message generation.

If you know how to deal with a good old HP RPN calculator, you can start with your 34S right away. To show you the features of the 34S completely, however, we wrote this little manual. It starts with a survey of the active keyboard in various modes, so

[1] Though the HP-42S was sold in 1988 already, this statement holds still. – Due to display restrictions, matrix math cannot be supported by the 34S. Sorry for this.

you know where to find what you are looking for. It continues with tables about addressing, browsing the catalogues, and a paragraph about the display and indicators used to tell you what's going on. The major part of this booklet is taken by the index of operations, catalogue contents, constants and conversions featured. It closes with a list of messages the 34S will display if special input conditions prevent it from executing your command as expected.

Your 34S is the result of an intercontinental collaboration of two individuals, an Australian and a German, though we did this in our free time, so you may call it our hobby to some extent. We baptized it 34S in honour of one of the most powerful LED pocket calculators, the HP-34C, and since it is our humble approach – with the hardware given – to a future 43S we can only dream of so far becoming the successor of the HP-42S.

We have checked everything we could think of carefully to our best knowledge, so our hope may be justified the 34S is bug-free. We cannot guarantee this, however, nor can we bear any liability for errors in calculations nor their possible consequences. Nevertheless, we promise we will improve the 34S whenever it will turn out being necessary – so if you ever discover any strange result, please report it to us, and if it is unveiled being an error we will provide you with an update as soon as we have one.

Enjoy!

Paul Dale and Walter Bonin

PRINT CONVENTIONS

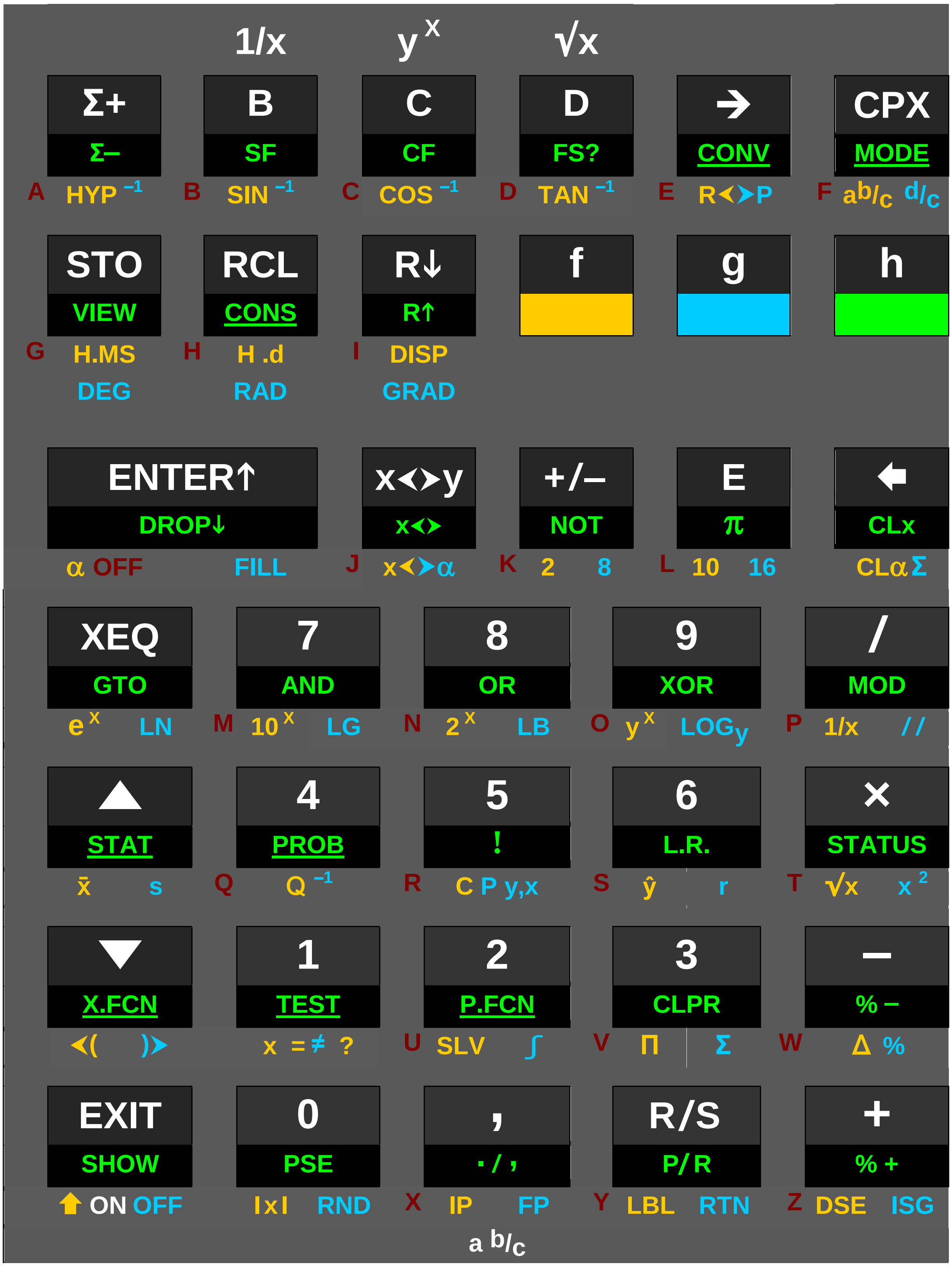
Throughout this manual, commands are generally called by their names, usually written in CAPITALS.

This CPX font is taken for explicit references to keys.

Register addresses are printed using Times New Roman. Lower case italic letters of this font are taken for register contents (e.g. *y* or ***r45*** or ***alpha*** for contents of stack level **Y** or general purpose register **R45** or the alpha register, respectively). Lower case bold italic Arial letters like ***n*** are used for variables.

All this holds unless stated otherwise explicitly.

KEYBOARD



Generally, white labels execute the *default primary function* of the respective key. To access a golden, blue, or green label, use *prefix* f, g, or h, respectively. Any label underlined opens a *catalogue*. For example, RCL preceded by

- f will set calculator mode to floating point decimal numbers via .d,
- g will set angular mode to radians via RAD,
- h will call the catalogue of constants via CONS.
- The dark red letter **H** will become relevant in *alpha mode* (see below).

Further remarks:

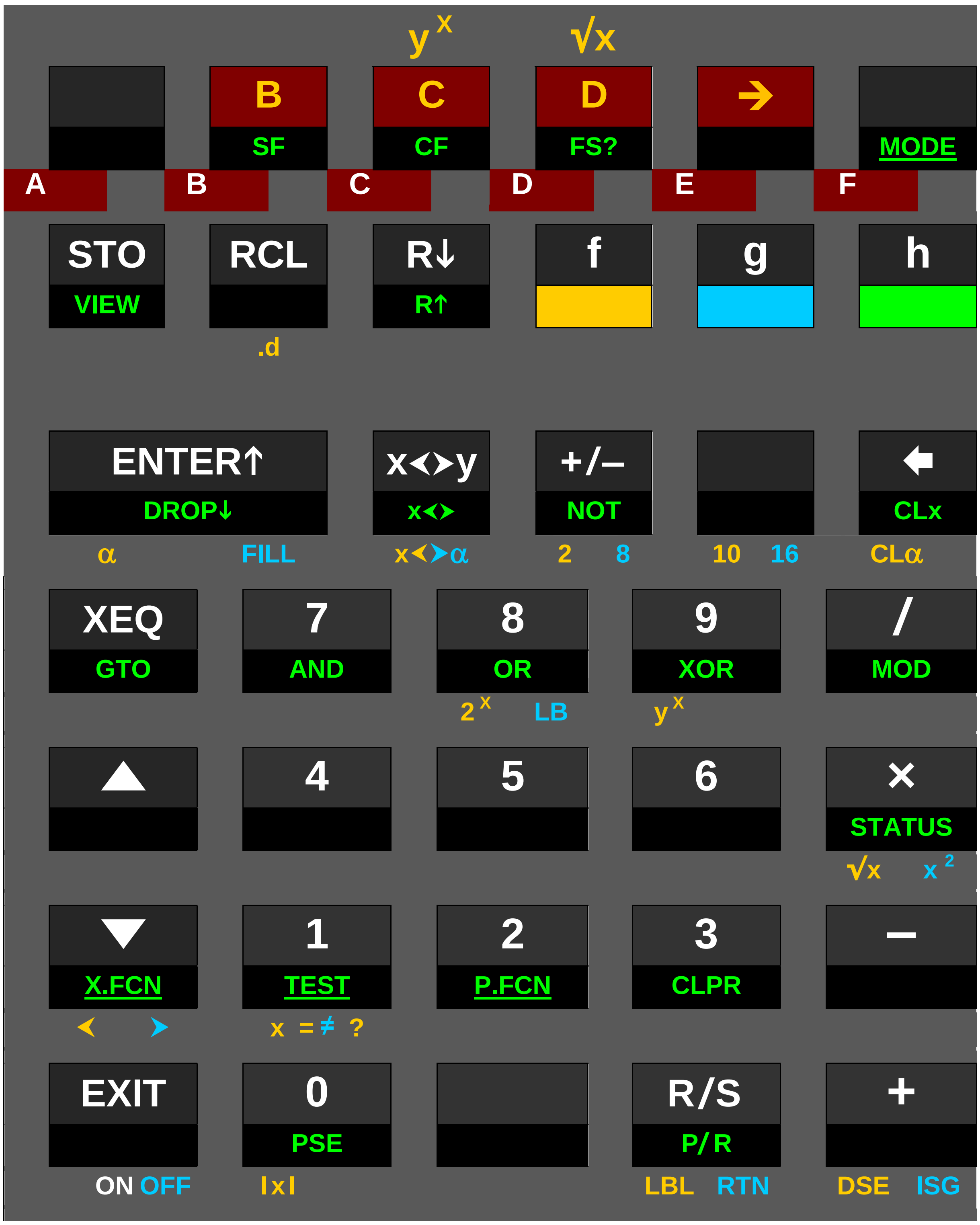
- The *hotkeys* B, C, and D in top row directly call the user programs carrying these labels if defined, else they act as 1/x, y^x, or $\sqrt{x}$, respectively.
- Prefix → combined with H.MS, H.d, DEG, RAD, GRAD, 2, 8, 10, and 16 converts x, while R◀▶P converts polar and rectangular coordinates in both x and y. So the latter switches representations of complex numbers, too.
- Prefix CPX may be used for calling functions in complex domain. Then names will be merged, e.g. CPX f COS will be displayed as ^{C}COS. Generally, if an arbitrary real function **f** works with x only, its complex sibling ^{C}**f** will work with $x_c = x + i\,y$. If **f** operates on one register, e.g. **R12**, then ^{C}**f** will operate on **R12** and **R13**. If **f** uses x and y then ^{C}**f** will use x, y, z and t. Please note all complex functions work with rectangular coordinates exclusively.
- Most one-number real functions replace x by the result $\mathbf{f}(x)$ stored in **X** again. In analogy, respective complex functions replace x by the real part and y by the imaginary part of the complex result $^C\mathbf{f}(x_c)$. Higher stack levels remain unchanged. Such functions are ^{C}1/x, ^{C}ABS, ^{C}FIB, ^{C}FP, ^{C}IP, ^{C}ROUND, ^{C}SIGN, ^{C}W, ^{C}W^{-1}, ^{C}x!, ^{C}x^2, $^C\sqrt{\ }$, ^{C}**+/−**, $^C\Gamma$(x), logarithmic and exponential with bases 10, 2 and e, as well as hyperbolic, trigonometric, and their inverses.

  Some real functions, e.g. DECOMP, operate on one number but return two. Other operations do not consume any stack input at all but return one or two numbers, like RCL or SUM. Then the extra number(s) will be pushed on the stack, taking one level per real or two per complex number, respectively.
- Two-number real functions replace x by the result $\mathbf{f}(x, y)$. Level **Y** is filled with the content of the next higher level, i.e. z. This goes on for higher levels, only the number on top is repeated as shown *below*.

  In analogy, respective complex functions replace x by the real part and y by the imaginary part of the complex result $^C\mathbf{f}(x_c, y_c)$. The next stack levels are filled with the contents of higher levels, and the complex number in the top two levels is repeated as shown *below*. Such complex functions are ^{C}LOGy, ^{C}y^x, $^C\beta$(x,y), ^{C}//, and the basic arithmetic operations.
- There are two three-number real functions included – Iβ and %MRR – replacing x by the result $\mathbf{f}(x, y, z)$. Then **Y** is filled with t and so on, and the content of the top level is repeated twice. No such complex functions are featured.
- If . is used twice in input, the 34S enters fraction mode. Calculator modes in general are as described in the *separate paragraph* below.

Please see the *index of operations* for a complete list of all the operations provided.

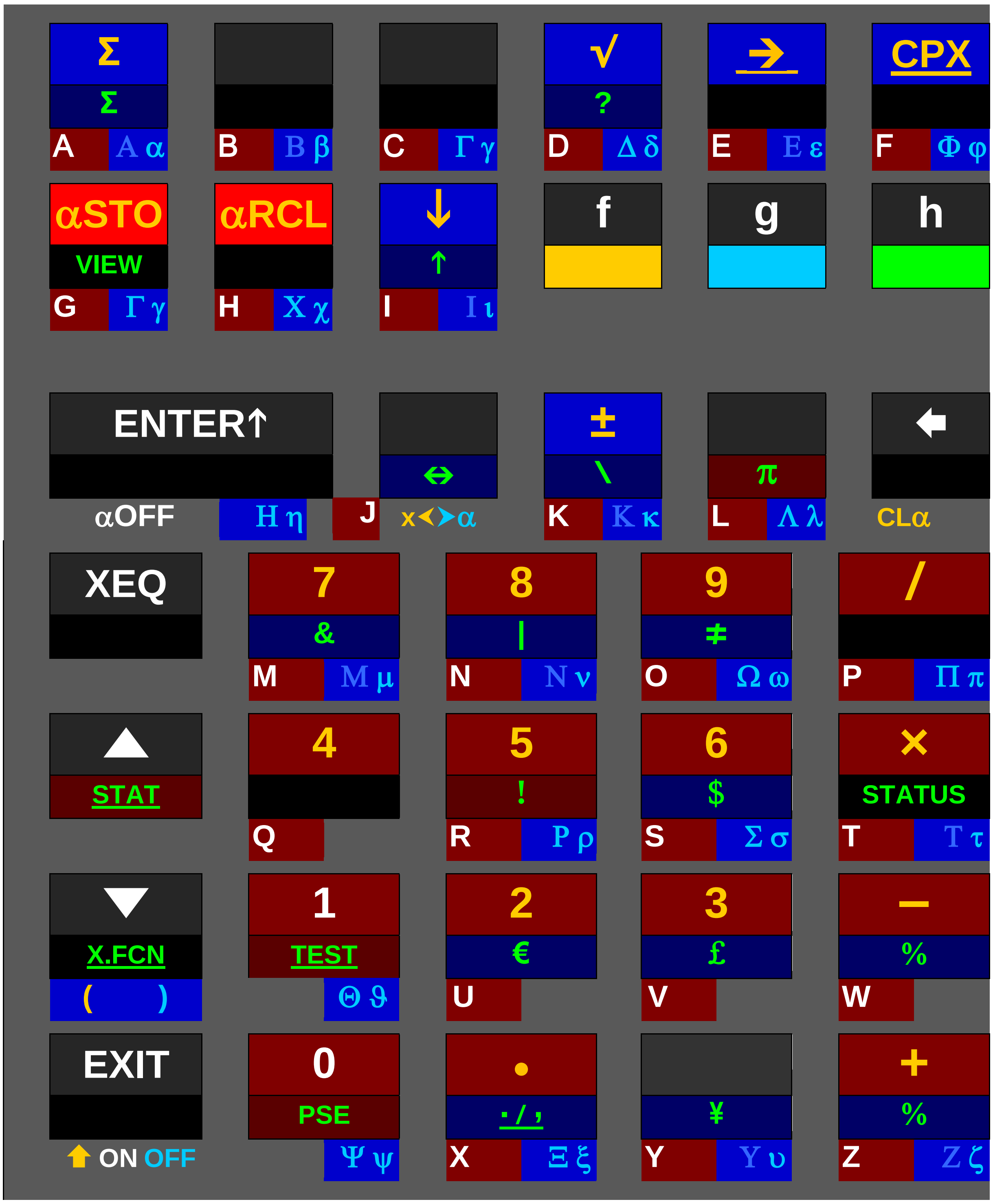
Virtual active keyboard in **hexadecimal mode**:



Primary functions of the top six keys will be numeric input, so their default primary functions are accessed using f . The key ➔ is exclusively for addressing and temporary display in other bases (see *addressing tables* and *index of operations* below).

For smaller integer bases, the active keyboard will look alike, but those top keys not needed for numeric input there will keep their default primary functions, except Σ+ and CPX . Attempts to enter an illegal digit will throw an *error*.

Virtual active keyboard in **alpha mode:**



In this mode, ***alpha*** is displayed in the dot matrix, and the numeric line is accessible by commands only. All labels printed on dark red or blue background in this picture append characters to ***alpha*** immediately or via alpha catalogues; those on blue deviate from the prints on the 34S at these locations.

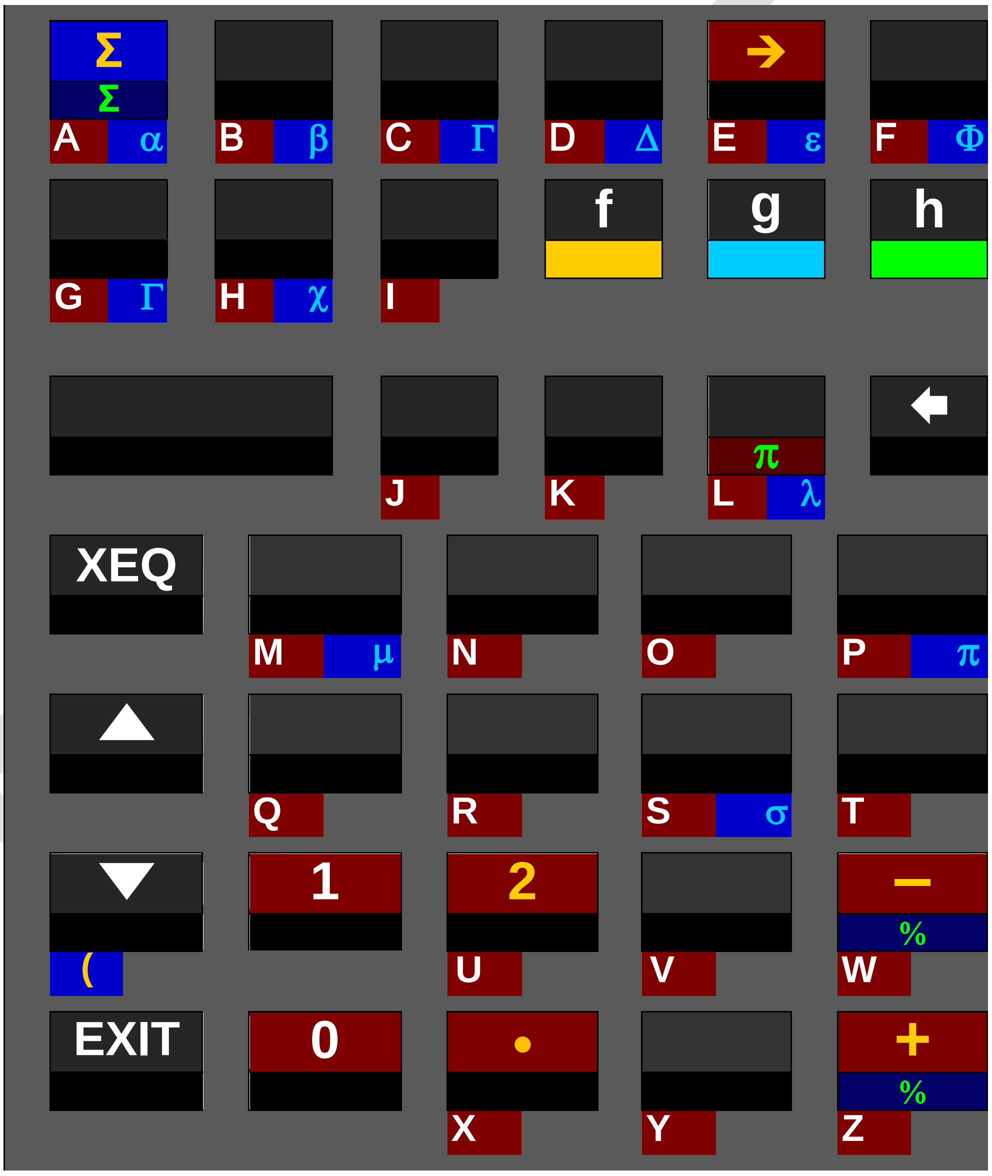
Generally, ↑ toggles upper and lower case, and PSE appends a space. Primary function of most keys is appending the letter printed bottom left of this key – dark red on the key plate. Then f is used for reaching the key tops in alpha mode there, and g leads to homonymic Greek letters where applicable [2].

Some logic symbols are accessible via the Boolean operations. Four currency symbols are located next to the %-sign as follows: **$** at the letter S, **€** at U for Euro, **£** at V, and **¥** at Y for Yen or Yuan. The catalogues h STAT, f ➡, f CPX, h TEST, and h ./, feature even more characters (see *below*).

If ***alpha*** is going to exceed 31 characters, the leftmost character(s) will be discarded.

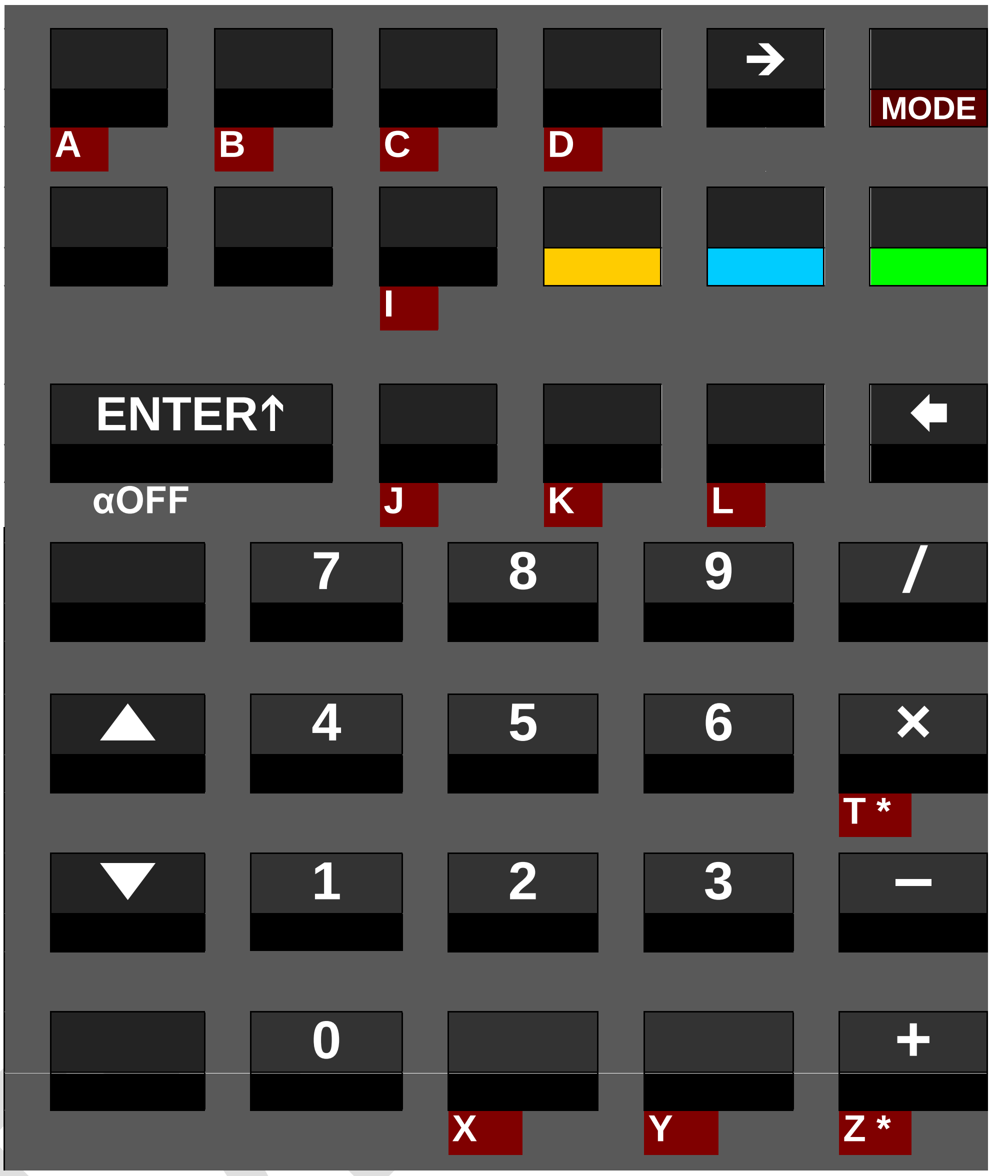
See the *index of operations* for αSTO and αRCL and many more alpha operations.

A subset of these characters is sufficient for **catalogue browsing:**

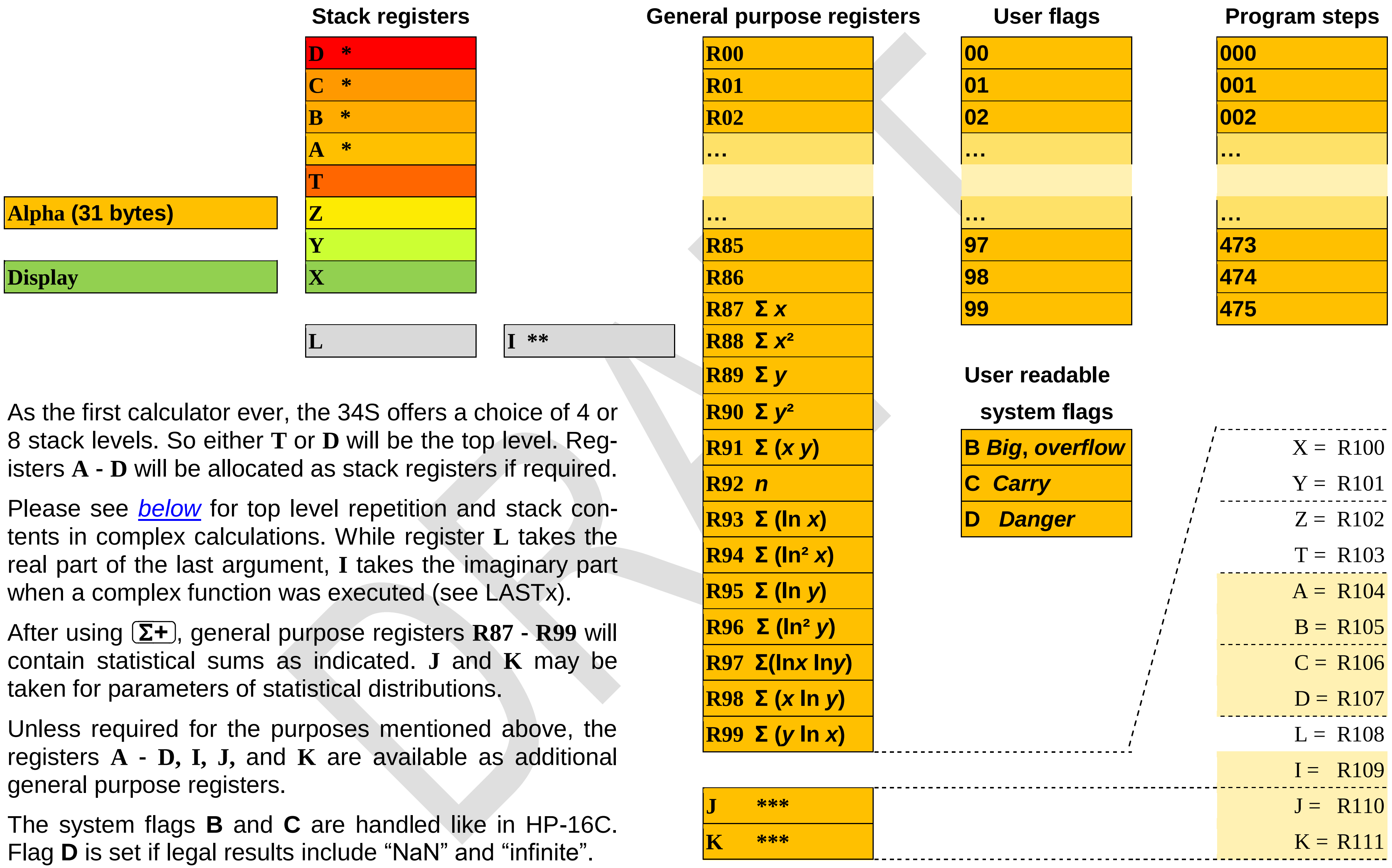


[2] "Homonymic" according to ancient Greek pronunciation. Three Greek letters require special handling: **Psi** is accessed via g 0 (below PSE), **Theta** via g 1 (below TEST), and **Eta** via g ENTER↑ . **Omicron** is not featured since looking exactly like **O** in either case. And we assigned **Gamma** also to **C** due to the alphabet, and **Chi** to **H** since this letter is next in pronunciation. Where we printed Greek capitals with lower contrast on page 7, they look like the respective Latin letters in our fonts. Greek professors, we hope for your understanding.

A **temporary alpha mode** is active during input processing in comparisons and addressing. See the virtual active keyboard here and find more about this mode below.



MEMORY



As the first calculator ever, the 34S offers a choice of 4 or 8 stack levels. So either **T** or **D** will be the top level. Registers **A - D** will be allocated as stack registers if required.

Please see *below* for top level repetition and stack contents in complex calculations. While register **L** takes the real part of the last argument, **I** takes the imaginary part when a complex function was executed (see LASTx).

After using Σ+, general purpose registers **R87 - R99** will contain statistical sums as indicated. **J** and **K** may be taken for parameters of statistical distributions.

Unless required for the purposes mentioned above, the registers **A - D, I, J,** and **K** are available as additional general purpose registers.

The system flags **B** and **C** are handled like in HP-16C. Flag **D** is set if legal results include "NaN" and "infinite".

STACK MECHANICS

What happens with the contents of particular stack levels depends on the function executed, its domain (integer/real or complex) and the stack size chosen.

Real and integer functions in a 4-level stack work as known for decades. Everything works alike in a larger stack on the 34S – just with more levels for intermediate results. Calculating formulas from inside out stays a wise strategy in either size. With more levels, however, stack overflow will hardly ever happen, even with the most advanced formulas you compute in your life as a scientist or engineer.

Calculating with complex numbers uses 2 registers or levels for each such number as explained above and shown here:

	Level	Assumed contents at the beginning:	Stack contents after executing … … the complex stack register operations							… **complex** functions of		… **integer/real** functions of 2 numbers like /	
			^{C}ENTER,	^{C}FILL	^{C}DROP	^{C}x↔y,	^{C}R↓,	^{C}R↑	^{C}LASTx	… 1 number like ^{C}x^2	… 2 numbers like ^{C}/	Before	After
With SSIZE4:	**T**	t = Im(y_c) = Im(t_c)	Im(x_c)		Im(y_c)	x_c			x_c	$y_c = t_c$	$y_c = t_c$	t	t
	Z	z = Re(y_c) = Re(t_c)	Re(x_c)		Re(y_c)							z	t
	Y	y = Im(x_c)	Im(x_c)		Im(y_c)	y_c			$lastx_c$	Im($(x_c)^2$)	Im(y_c / x_c)	y	z
	X	x = Re(x_c)	Re(x_c)		Re(y_c)					Re($(x_c)^2$)	Re(y_c / x_c)	x	y / x
With SSIZE8:	**D**	d = Im(t_c)	z_c	x_c	t_c	t_c	x_c	z_c	z_c	t_c	t_c	d	d
	C	c = Re(t_c)										c	d
	B	b = Im(z_c)	y_c	x_c	t_c	z_c	t_c	y_c	y_c	z_c	t_c	b	c
	A	a = Re(z_c)										a	b
	T	t = Im(y_c)	x_c	x_c	z_c	x_c	z_c	x_c	x_c	y_c	z_c	t	a
	Z	z = Re(y_c)										z	t
	Y	y = Im(x_c)	x_c	x_c	y_c	y_c	y_c	t_c	$lastx_c$	$(x_c)^2$	y_c / x_c	y	z
	X	x = Re(x_c)										x	y / x

So, an 8-level stack gives you the same flexibility in complex domain you are used to with a 4-level stack in real domain.

ADDRESSING AND COMPARING REAL NUMBERS

1	User input	x= ?, x≠ ?, x< ?, x≤ ?, x≥ ?, or x> ?				RCL, STO, αRCL, αSTO, VIEW, x⇄, DSE, ISG, DSZ, ISZ, FIX, SCI, ENG, DISP, BASE, CB and many more bit commands, or CF and the other flag commands		
	Dot matrix display	**OP _** (with temporary alpha mode set) e.g. ▫x > _				**OP _** (with temporary alpha mode set) e.g. RCL _ [3]		
2	User input	0 or 1	*Stack level or named reg.* X, Y, ...	ENTER↑ [4] leaves temp. alpha mode.	→ opens indirect addressing.	*Stack level or named register* X, Y, Z,... , K [5]	***Number** of register or flag or bit(s) or decimals* [6]	→ opens indirect addressing.
	Dot matrix display	**OP *n*** e.g. x ≤ 0 ?	**OP *x*** e.g. x ≥ y ?	**OP r_**	**OP →_**	**OP *x*** e.g. SCI Z	**OP *nn*** e.g. SF 15	**OP →_**
3	User input	Compares ***x*** with the number **0**.	Compares ***x*** with the number on stack level **Y**.	*Register no.* 00 … 99	Look right for more about indirect addressing.	Sets scientific display with the number of decimals specified in stack level **Z**.	*Stack level etc.* X, Y, Z,... , K	*Register number* 00 … 99
	Dot matrix display			**OP r *nn*** e.g. x ≠ r23?			**OP → *x*** e.g. VIEW →L	**OP → *nn*** e.g. STO →45
				Compares ***x*** with the number stored in **R23**.			Shows the content of the register where **L** is pointing to.	Stores ***x*** into the location where **R45** is pointing to.

[3] For RCL and STO, any of +, −, ×, /, ▲, or ▼ may precede step 2. See the index of operations.

[4] You may skip this for register numbers >19.

[5] Exceptions: RCL T, RCL × T, RCL Z, RCL + Z require an ENTER↑ previous to T or Z, e.g. RCL + ENTER↑ Z for the latter. This holds for STO as well.

[6] Register and flag numbers may be 00 … 99, number of decimals 0 … 11, integer bases 2 … 16, bit numbers 0 to 63, and integer word size up to 64 bits. For numbers <10, you may key in e.g. 5 ENTER↑ instead of 0 5. There are three additional flags addressed via B, C, and D. – Take into account some registers may be allocated to special applications.

ADDRESSING AND COMPARING COMPLEX NUMBERS

1	User input	$\boxed{\text{CPX}}$ $\boxed{x=?}$ or $\boxed{x\neq ?}$				$\boxed{\text{CPX}}$ $\boxed{\text{RCL}}$, $\boxed{\text{STO}}$, or $\boxed{x\gtrless}$		
	Dot matrix display	**OP _** (with temporary alpha mode set) e.g. ▪x = _				**OP _** (with temporary alpha mode set) e.g. ▪RCL _ [7]		
2	User input	$\boxed{0}$ or $\boxed{1}$	*Stack level or named register* $\boxed{X}$, $\boxed{Z}$, $\boxed{A}$, $\boxed{C}$, $\boxed{L}$, or $\boxed{J}$	$\boxed{\text{ENTER}\uparrow}$ [8] leaves temp. alpha mode	$\boxed{\rightarrow}$ opens indirect addressing.	*Stack level or named register* $\boxed{Z}$ [9], $\boxed{A}$, $\boxed{C}$, $\boxed{L}$, or $\boxed{J}$	*Register number* $\boxed{0}\boxed{0}$ .. $\boxed{9}\boxed{8}$ [10]	$\boxed{\rightarrow}$ opens indirect addressing.
	Dot matrix display	**OP *n*** e.g. ▪x = 0 ?	**OP *x*** e.g. ▪x ≠ z ?	**OP r_**	**OP →_**	**OP *x*** e.g. ▪RCL L	**OP *nn*** e.g. ▪STO 18	**OP →_**
3	User input	Compares ***x*** + *i* ***y*** with the real number **0**.	Compares ***x*** + *i* ***y*** with ***z*** + *i* ***t*** .	*Register number* $\boxed{0}\boxed{0}$ … $\boxed{9}\boxed{8}$	Look right for more about indirect addressing.	This is ^{C}LASTx.	*Stack level or named register* $\boxed{X}$, $\boxed{Y}$, ... , $\boxed{K}$	*Register number* $\boxed{0}\boxed{0}$ … $\boxed{9}\boxed{9}$
	Dot matrix display			**OP r *nn*** e.g. ▪x ≠ r26?			**OP → *x*** e.g. ▪x<> →Z	**OP → *nn*** e.g. ▪STO →45
				Compares ***x*** + *i* ***y*** with ***r26*** + *i* ***r27*** .			Swaps ***x*** with the contents of the register where **Z** is pointing to, and ***y*** with the contents of the next one.	Stores ***x*** + *i* ***y*** into 2 consecutive registers, starting with the one where **R45** is pointing to.

[7] For $\boxed{\text{RCL}}$ and $\boxed{\text{STO}}$, any of $\boxed{+}$, $\boxed{-}$, $\boxed{\times}$, or $\boxed{/}$ may precede step 2. See the index of operations.

[8] You may skip this keystroke for register numbers >19.

[9] Exceptions: ^{C}RCL Z, ^{C}RCL + Z, ^{C}STO Z, and ^{C}STO + Z require an $\boxed{\text{ENTER}\uparrow}$ previous to $\boxed{Z}$, e.g. $\boxed{\text{CPX}}$ $\boxed{\text{STO}}$ $\boxed{+}$ $\boxed{\text{ENTER}\uparrow}$ $\boxed{Z}$ for the latter.

[10] You may key in e.g. $\boxed{8}$ $\boxed{\text{ENTER}\uparrow}$ instead of $\boxed{0}$ $\boxed{8}$. Take care of pairs, since a complex operation will always affect two registers: the one specified and the one following this. We recommend storing complex numbers with their real parts at even register numbers. – Take into account some registers may be allocated to special applications.

ADDRESSING LABELS

1 User input Dot matrix display	B, C, or D **XEQ '*label*'** e.g. `XEQ 'A'`	XEQ, GTO, LBL, SLV, ∫, π or Σ **OP _** e.g. `GTO _`				
2 User input Dot matrix display	Calls the function labeled **A**.	B, C, or D **OP '*label*'** **e.g.** `Σ 'B'`	ENTER↑ sets alpha mode. **OP '_**	→[11] opens indirect addressing and sets temporary alpha mode. **OP →_**		*2-digit numeric label* 0 0 … 9 9 **OP *nn*** e.g. `LBL 07`
3 User input Dot matrix display		Sums up the function labeled **B**.	***Alphanumeric label*** (≤ 3 characters [12]) **OP '*label*'** e.g. `SLV'F1µ'`	*Stack level or named register* X, Y, Z, ... , K **OP → *x*** e.g. `∫ →T`	*Register number* 0 0 … 9 9 [13] **OP → *nn*** e.g. `XEQ →44`	
			Solves the function **F1µ** (with **F1µ** keyed in).	Integrates the function which's label is on stack level **T**.	Executes the routine which's label is in **R44**.	

[11] Works with all these operations except LBL.

[12] The 3rd character terminates entry and closes alpha mode. Shorter labels need a closing ENTER↑ .

[13] Some registers may be allocated to special applications. Please check the memory table above.

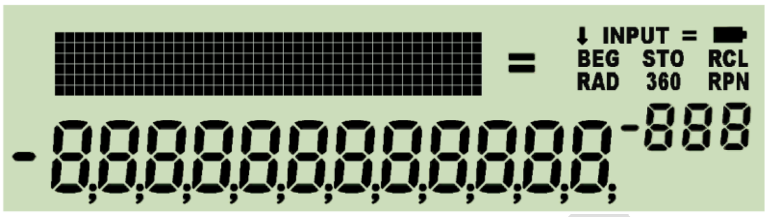
ADDRESSING CATALOGUE ITEMS

1	User input	CONS, CONV, MODE, P.FCN, STAT, TEST, X.FCN	CPX or STAT in alpha mode	TEST, ./., or ➡ in alpha mode
	Dot matrix display	**Shows 1st item in selected catalogue.** (e.g. BC? in P.FCN) Alpha mode is set.	(e.g. Á in CPX)	(e.g. , in PUNCT)
2	User input	XEQ, ▼, ▲, EXIT, or 1st character (e.g. F)	XEQ, ▼, ▲, EXIT, or letter (e.g. O)	↓
	Dot matrix display	**Shows 1st item starting with this character *)** (e.g. FB)	**Shows 1st item starting with this letter *)** (e.g. Ó)	↓
3	User input	XEQ, ▼, ▲, EXIT, or 2nd character (e.g. S)	↓	↓
	Dot matrix display	**Shows 1st item starting with this sequence *)** (e.g. FS?)	↓	↓
4	User input	XEQ, ▼, ▲, or EXIT (e.g. ▼)		
	Dot matrix display	**Shows next item in this catalogue** (e.g. FS?C)	(e.g. Ò)	(e.g. ")
…		Continue browsing this way until reaching the item desired (e.g. FS?F).	(e.g. Ö).	(e.g. :).
n	User input	XEQ Calculator leaves the catalogue returning to the mode set before … and executes or inserts the command chosen.	… and appends the selected character to ***alpha***.	
	Dot matrix display	**Result**	**Contents of alpha register** (e.g. Östl. Seite:)	

*) If a character or sequence specified is not found in this catalogue then the first item following alphabetically will be shown.

DISPLAY

The display features three sections: numeric, dot matrix and fixed symbols. The numeric section features a minus sign and 12 digits for the mantissa, as well as a minus sign and 3 digits for the exponent. The dot matrix is 6 dots high and 43 dots wide, allowing for some 7 to 12 characters, depending on their widths. The fixed symbols (except the big “=”) are called *annunciators*, and are for indicating modes.



The dot matrix section above is used for

1. indicating some more modes than the annunciators allow, adjusted to the right,
2. passing additional information to the user, adjusted to the left.

The numeric section in the lower part of the LCD is used for displaying numbers in different formats, status, or messages.

If two or more requests concur for display space, the items will be shown according to their priorities as follows:

1. error messages as described in a paragraph further below,
2. special information as explained below,
3. information about the modes the calculator is running in.

The *annunciators* or specific characters in the display signal the modes:

Signal	***INPUT***	**b**	**d**	**h**	**o**		***STO***
Mode name if different	**α**	**2**			**8**	**FLOAT**	**PRG**
Set by …	αON	BASE2	BASE10	BASE16	BASE8	BASE0	PRGON
Cleared by …	αOFF	any other BASE setting IMPFRC, PROFRC, H.MS, TIME, →H.MS					PRGOFF

Signal	***360***	***RAD***	**G**		
Mode name if different				**H.MS**	**FRC**
Set by …	DEG	RAD	GRAD	H.MS, TIME →H.MS	BASE1 IMPFRC, PROFRC 2nd . in input
Cleared by …	GRAD RAD	DEG GRAD	DEG RAD	BASE, →H COS, SIN, TAN IMPFRC, PROFRC	BASE ≠1 H.MS, TIME →H.MS

A running program is signaled by a flashing ***RCL*** annunciator. ***RPN*** may be lit permanently. Time modes (12h / 24h) are seen in the time string directly. The numeric formats of H.MS and fraction modes are unambiguous as well. Further settings are signaled in the dot matrix section, like the different date modes being indicated there by **D.MY** or **M.DY**. Defaults Y.MD and FLOAT are not indicated. Please check the examples below.

Some mode and display settings may be stored and recalled collectively by STOM and RCLM. The command RCLM recalls a 18-bit word containing mode data packed as follows, starting with the least significant bit:

Bits	Meaning	Values and corresponding settings
0, 1	Display format for real numbers	0 = ALL 1 = FIX 2 = SCI 3 = ENG
2 … 5	Number of decimals	0 … 12
6, 7	Angular mode	0 = DEG 1 = RAD 2 = GRAD 3 = DEG H.MS
8, 9	Date display format	0 = Y.MD 1 = D.MY 2 = M.DY
10	Time display format	0 = 24h 1 = 12h
11	Radix mark	0 = point 1 = comma
12 … 14	Curve fit model	0 = LinF 1 = ExpF 2 = PowerF 3 = LogF 4 = BestF
15, 16	Integer sign mode	0 = 2COMPL 1 = 1COMPL 2 = UNSIGN 3 = SIGNMT
17	Stack depth	0 = 4 levels 1 = 8 levels

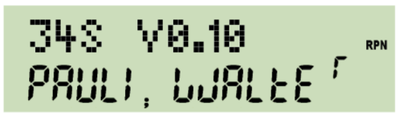
E.g. the start-up default with 4 stack levels,
FIX 4, DEG, Y.MD, 24h, decimal point,
LinF, 2COMPL is $000000000000010001_2 = 17_{10}$

Settings for 8 stack levels, SCI 2, RAD,
D.MY, 12h, decimal comma, BestF,
UNSIGN correspond to $110100110101001010_2 = 445770_{10}$

STOM takes such a number and sets the calculator modes accordingly. Please see the *index of operations* for more information about changing modes.

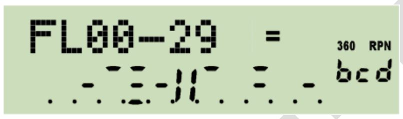
Some commands and modes use the display in a special way. They are listed below in order of falling priority:

1. **VERS** generates a display like this:

   This tells you have a 34S with firmware version 0.10 – the display on your own 34S may deviate from this example. Pressing any key will delete this message and return to previous state.

2. **STATUS** displays the status of 30 flags very concisely, allowing an immediate status overview after some training. If e.g. flags 2, 3, 5, 7, 11, 13, 14, 17, 19, and 23 are set, and labels B, C, and D are defined in program memory, STATUS will display this:

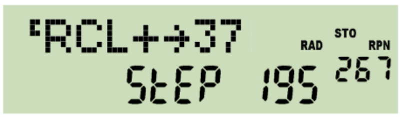
   Within the numeric section, each row of horizontal bars in the mantissa shows the status of 10 flags. When a flag is set, the respective bar turns black. So here the top row of bars indicates flags 0 and 1 are clear, 2 and 3 set, and flag 4 clear. Then, the divider II separates the first group of five flags from the next. Top row bars on its right side indicate flags 5 and 7 are set. Next row of bars shows flags 11, 13, 14, 17, 19 are set, and in the lowest row only flag 23 is set. All other flags in the range from 10 to 29 are clear.

   Scrolling down by ▼ will display flags 10 - 39, then 20 - 49 etc. until 80 - D. Scrolling up by ▲ reverts this. Alternatively, pressing a digit, e.g. 5, will show 30 flags starting with 10 times this digit, e.g. flags 50 - 79. The numeric exponent always indicates the status of the 3 hotkeys top left on the keyboard.

   The status will be displayed until any key is pressed but ▼, ▲, or a digit < 9.

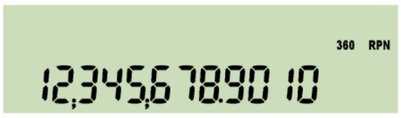
3. During **command input**, the dot matrix displays the command chosen until input is completed, i.e. until all required trailing parameters are entered. The prefixes f, g, and h are shown until they are resolved. In addressing, progress is recorded as explained in the *addressing tables above* in detail.

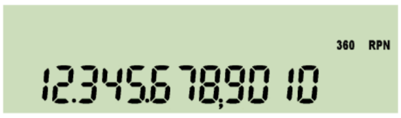
4. In **programming mode**, the numeric display indicates the program step (001 – 476) in the mantissa and the number of free steps in the exponent, while the dot matrix shows the command contained in the respective step, e.g.:

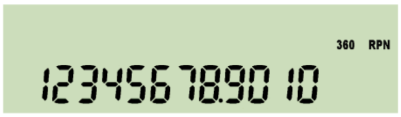
5. For **floating point numbers**, the mantissa will be displayed adjusted to the right, the exponent to the left. Within the mantissa, either points or commas may be selected as radix marks[14], and additional marks may be chosen to separate thousands. Assume the display set to FIX 4, then 12.345678901 millions may look like:

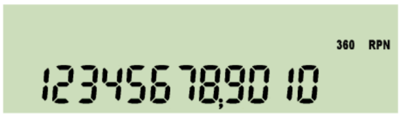
[14] Starting here, decimal input is written using a point as radix mark throughout this manual, although significantly less visible, unless specified otherwise explicitly. By experience, the „comma people“ are more capable to read radix points and interpret them correctly than vice versa.



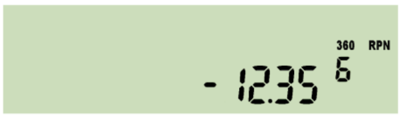
or


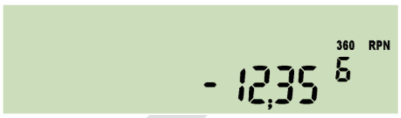
with thousands separators on, and without them like:



or


With ENG 2 and after changing the sign, the same number looks like this:

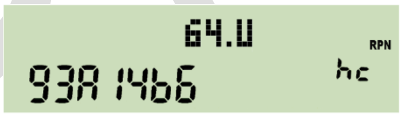


or


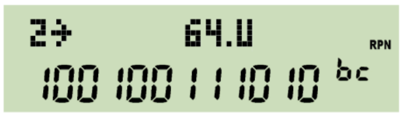
6. In **integer modes**, numbers are displayed adjusted to the left. Word size and complement setting are indicated in the dot matrix using a format WW.C, with C being 1 or 2 for 1's or 2's complement, respectively, U for unsigned, or S for sign-and-mantissa mode. Sign and 1st digit of the exponent show the base, a "c" in the 2nd digit signals a carry bit set, an "o" in the 3rd an overflow. Integer bases are indicated as follows:

Base	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Sign and 1st digit of exponent displayed	**b**	**3**	**4**	**5**	**6**	**7**	**o**	**9**	**d**	**-1**	**-2**	**-3**	**-4**	**-5**	**h**

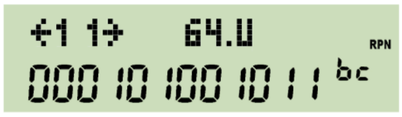
The example shows the 34S in unsigned hexadecimal mode with word size 64 and carry set:



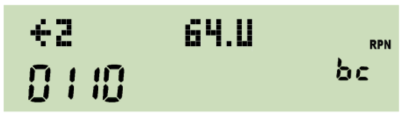
The same number displayed in binary representation will need more than 12 digits, being 1001001110100001010010110110. So, the display will show the 12 most significant digits together with an indication that there is more and where to look for it:



Now press g ▶ and you will get the next 12 digits:



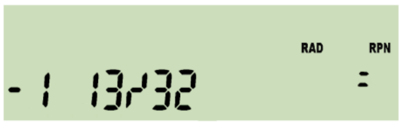
Press g ▶ again to show the least significant digits:

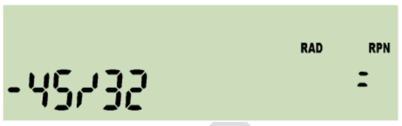


Please note the window will also change when 12 digits are keyed in, so the rightmost digits may fill the window incompletely. The digit shown in the dot matrix section next to the arrow indicates the number of windows found in the direction shown. Windows to the left contain 12 digits always.

7. In **fraction mode**, the fraction will be shown in the numeric display, adjusted to the left. "=", "Lt", or "Gt" is indicated in the exponent if the fraction is exactly equal, slightly less, or greater than the floating point number converted, respectively.

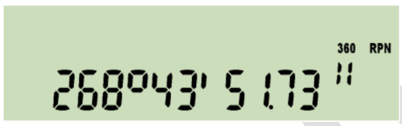
   E.g. -1.40625 will be displayed as follows:

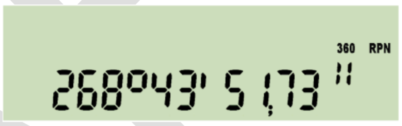
   

   or
   

   depending on the setting for proper fractions and assuming DENMAX ≥ 32. Fraction mode can handle numbers with absolute values <100,000. Maximum denominator is 9999. Some fractions featuring large numerators or denominators may exceed the display window. Then the same rules apply as in integer modes. Please note integers like 123 will be displayed as "123 0/1" or "123/1" in fraction mode, respectively.

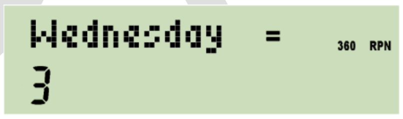
8. In **H.MS mode**, input format is `hhhh°mm'ss.dd"` with the number of hours or degrees limited to 9000. Output is adjusted to the right. It may look like this:

   or
   

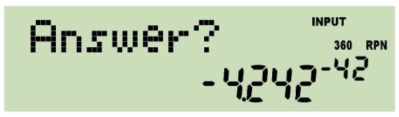
   depending on the radix setting.

9. Output of the function **DAY** will look as follows for an input of 1.13201 in M.DY mode (equivalent to inputs of 13.01201 in D.MY or 2010.0113 in Y.MD):

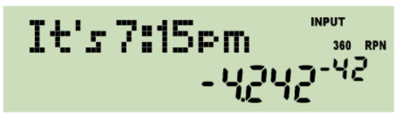
   

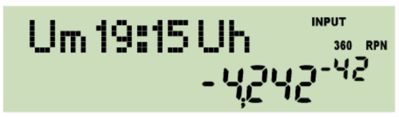
   The display may look similar for a result of DAYS+.

10. In **alpha mode**, the alpha register is displayed in the dot matrix while the numeric section keeps the result of the last numeric operation, e.g.:

    Different information may be appended to ***alpha***. See the commands starting with "α" in the index of operations below. E.g. αTIME allows creating texts like

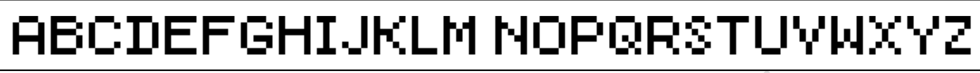
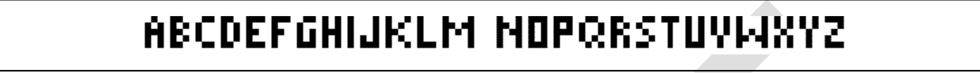
    or
    

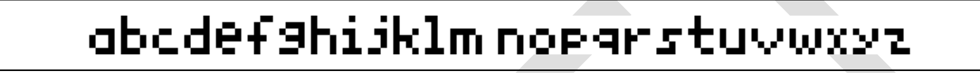
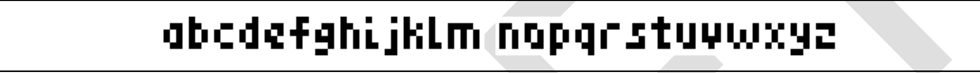
    depending on the time mode setting (12h / 24h).

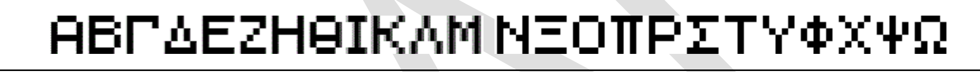
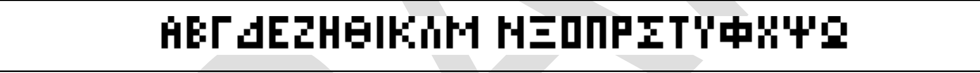
All keyboard inputs will be interpreted according to the modes set at input time.

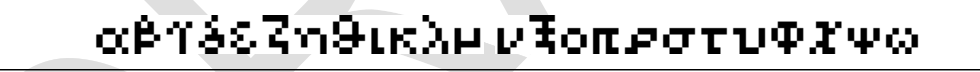
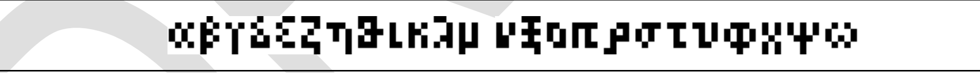
FONTS

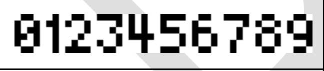
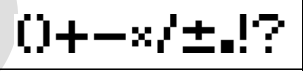
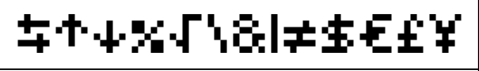
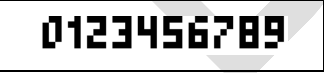
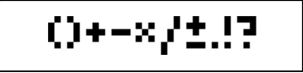
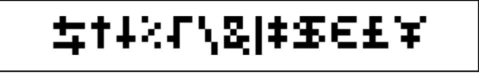
The 34S features a big and a small font. Both are based on Luiz Viera's fonts as distributed in 2004. Some letters were added and some modified for better legibility, since the dot matrix is only 6 pixels high. The following tables show the characters directly accessible through the keyboard. Those contained in the alpha catalogues are found below.

A B C D E F G H I J K L M N O P Q R S T U V W X Y Z



a b c d e f g h i j k l m n o p q r s t u v w x y z



Α Β Γ Δ Ε Ζ Η Θ Ι Κ Λ Μ Ν Ξ Ο Π Ρ Σ Τ Υ Φ Χ Ψ Ω



α β γ δ ε ζ η ϑ ι κ λ μ ν ξ ο π ρ σ τ υ φ χ ψ ω



0 1 2 3 4 5 6 7 8 9	( ) + - * / ± . ! ?	↔ ↑ ↓ % √ \ & | ≠ $ € £ ¥
		
		

INDEX OF OPERATIONS

This lists all functions available on the 34S with their names and keystrokes necessary. Names printed in **bold** face therein belong to commands directly accessible on the keyboard, the others are accessible via catalogues. These names will show up in program listings as well. Sorting is case insensitive and works as follows: 0 … 9, A … Z, α … ω, (,), +, -, *, /, ±, ",", ".", !, ?, ↔, ←, ↑, ↓, →, <, ≤, =, ≠, ≥, >, #, °, %, √, ∫, ∞. Super- and subscripts are handled like normal characters in sorting.

Generally, functions and keystroke programming will work as on the HP-42S, bit and integer functions as on the HP-16C, unless stated otherwise under remarks. Especially, all **tests** will return "Yes" or "No" in the dot matrix if called from the keyboard; if called in a program, they will lead to execution of the next program line if the test is true, else skip this line. We recommend you have the manuals of the vintage calculators mentioned ready to hand, e.g. on the DVD distributed by *www.hpmuseum.org*.

Functions available on the 34S for the first time on an RPN calculator are highlighted under remarks, as are operations carrying a familiar name but deviating in their functionality here.

Parameters will be taken from the lowest stack levels unless being mentioned explicitly in the 2nd column. Then they must follow the command. If ***underlined***, they may also be specified using indirect addressing, as shown in the *tables* above. Some parameters of statistical distributions must be given in registers **J** and **K** if specified.

Each function is listed stating the mode(s) it will work in, abbreviated by their *indicators*. In this column an "&" stands for a Boolean AND, a comma for an OR, and a backslash for "all but". So e.g. 2^X works in all modes but alpha. "FLOAT^{H}" stands for "FLOAT, H.MS". All operations will also work in mode PRG unless stated otherwise explicitly.

Name	Keys to press	in modes	Remarks
c…	CPX …	FLOAT	Indicates a complex operation (see *above*). CPX may be combined with many functions which's *names are printed in italics here* .
0 … 9	0 … 9	\α	Standard numeric input. For integer bases <10, input of illegal digits throws an *error message*.
		in addressing	Register input. See the *addressing tables* above for more.
	0, 1, f 2 … f 9	α	Appends the respective digit to ***alpha***.
10^x	f 10^x	FLOAT	
12h	h MODE 12h	\α	Sets 12h time display. 21:34 becomes 9:34 p.m.
1COMPL	h MODE 1COMPL	\α	Sets 1's complement mode like in HP-16C.
1/x	f 1/x	FLOAT	
	B	FLOAT	Shortcut as long as label B is not defined yet.

Name	Keys to press	in modes	Remarks
24h	h MODE 24h	\α	Sets 24h time display. 1:23 p.m. becomes 13:23.
2COMPL	h MODE 2COMPL	\α	Sets 2's complement mode like in HP-16C.
2^x	f 2^x	\α	
A … F	A … F (red print)	-1, -2, -3, -4, -5, h	Numeric input for digits >10. See page 6 for more information.
A … D, I … L etc.	A … D, I … L, T, X … Z (red print)	in addressing	Register input. See the *addressing tables* above for more.
A … Z	A … Z (red print)	α	Alphabetic input. See page 7 for more.
ABS	f |x|	\α	^{C}ABS returns the magnitude $r=\sqrt{x^2+y^2}$ in **X** and clears **Y**.
ACOS	g COS^{-1}	$FLOAT^H$	
ACOSH	g HYP^{-1} COS	FLOAT	
ALL	h MODE ALL	\α	Selects the format displaying "all" digits.
AND	h AND	Integer	Works bitwise as in HP-16C.
		FLOAT	Works like AND in HP-28S, i.e. ***x*** and ***y*** are interpreted before executing this operation. 0 is "false", any other real number is "true".
ANGLE	h X.FCN ANGLE	FLOAT	Calculates the angle between positive x-axis and the straight line from the origin to the point ***(x, y)*** , returns this angle in **X** and clears **Y**.
ASIN	g SIN^{-1}	$FLOAT^H$	
ASINH	g HYP^{-1} SIN	FLOAT	
ASR	h X.FCN ASR ***n***	Integer	Works like ***n*** (up to 63) consecutive ASRs in HP-16C. ASR 0 executes as NOP.
ATAN	g TAN^{-1}	$FLOAT^H$	
ATANH	g HYP^{-1} TAN	FLOAT	
BASE	h MODE BASE ***n***	\α	Sets the base for integer calculations, with 2 ≤ ***n*** ≤ 16. Popular bases are directly accessible on the keyboard. Current integer base setting is indicated in the exponent as explained *above*. Furthermore, BASE0 equals FLOAT, and BASE1 calls PROFRC.
BASE10	f 10		
BASE16	g 16		
BASE2	f 2		
BASE8	g 8		

Name	Keys to press	in modes	Remarks
BC?	h TEST BC? ***n***	Integer	Tests the specified bit in x .
BestF	h STAT BestF	FLOAT	Selects the best curve fit model, maximizing the correlation like BEST does in HP-42S.
BS?	h TEST BS? ***n***	Integer	Tests the specified bit in x .
B(x)	h PROB B(x)	FLOAT	= BINOMDIST(***x; j; k;* 1**) in MS Excel, with the sample size j and the gross error probability k. B^{-1}(p) returns the number of successes g for a given probability p in **X**.
B $^{-1}$(p)	h PROB B^{-1}(p)		
CB	h X.FCN CB ***n***	Integer	Clears the specified bit in x .
CEIL	h X.FCN CEIL	FLOAT	Returns the smallest integer ≥ x .
CF	h CF ***n***	\α	Clears the flag specified.
CLFLAG	h P.FCN CLFLAG	\α	Clears all user flags.
CLREG	h X.FCN CLREG	All	Clears all general purpose registers.
CLSTK	0 g FILL	\α	Clears all stack registers.
CLx	h CLX	\α	^{C}CLx clears both **X** and **Y**.
CLα	f CLα	All	Clears the alpha register like CLA in HP-42S.
CLΣ	g CLΣ	FLOAT	Clears all statistical sums.
COMB	f Cy,x	FLOAT	Returns the number of possible <u>sets</u> of y items taken x at a time. No item occurs more than once in a set, and different orders of the same x items are <u>not</u> counted separately. Formula: $C_{y,x} = \binom{y}{x} = \frac{y!}{x! \cdot (y-x)!}$
CONJ	CPX h X.FCN CONJ	FLOAT	Changes the sign of y .
CORR	g r	FLOAT	Returns the correlation coefficient for the current statistical data and curve fitting model.
COS	f COS	FLOAT^{H}	
COSH	f HYP COS	FLOAT	
DATE	h X.FCN DATE	FLOAT	Recalls the date from the real time clock and displays it in the numeric section in the format selected. See D.MY, M.DY, and Y.MD. The function DATE in HP-12C corresponds to DAYS+ here (see below).

Name	Keys to press	in modes	Remarks
DAY	h X.FCN DAY	FLOAT	Takes x as a date in the format selected and returns the name of the day in the dot matrix and a corresponding integer in the numeric display (Monday = 1, Sunday = 7).
DAYS+	h X.FCN DAYS+	FLOAT	Works like DATE in HP-12C, adding x days on a date in **Y** in the format selected and displaying the resulting date including the day of week in the same format as DAY does.
DBLR	h X.FCN DBLR	Integer	Double precision commands like in HP-16C.
DBL ×	h X.FCN DBL×		
DBL /	h X.FCN DBL/		
DECOMP	h X.FCN DECOMP	FRC	Decomposes x (after converting it into an improper fraction, if applicable), resulting in a stack [***numerator(x), denominator(x), y, z***] or [***num(x), den(x), y, z, t, a, b, c***] , respectively. Reversible by division.
DEG	g DEG	FLOAT	Sets angular mode to degrees.
DENANY	h MODE DENANY	\α	Sets default fraction format like in HP-35S, allowing maximum precision. Any denominator up to the value set by DENMAX may appear.
DENFAC	h MODE DENFAC	\α	Sets "factors of the maximum denominator". With e.g. DENMAX = 60, possible denominators are 2, 3, 4, 5, 6, 10, 12, 15, 20, 30, and 60.
DENFIX	h MODE DENFIX	\α	Sets fixed denominator format, i.e. the denominator equaling DENMAX always.
DENMAX	h MODE DENMAX	\α	Works like /c in HP-35S, but maximum value settable is 9999. The max. denominator will be set to 9999 if $x < 1$ or $x > 9999$ at execution time. For $x = 1$ the current setting is recalled.
DISP	f DISP ***n***	FLOAT	Changes the number of decimals while keeping the display format (FIX, SCI, ENG) as is.
DROP	h DROP⬇	\α	Drops x , changing stack contents to [***y, z, t, t***] or [***y, z, t, a, b, c, d, d***] , respectively. See *above* for ^{C}DROP.
DSE	f DSE ***r***	PRG	Given cccccc.fffii in ***r*** , this function decrements ***r*** by ii, skipping next program line if then ccccccc ≤ fff for DSE, or = 0 for DSZ
DSZ	h P.FCN DSZ ***r***		
D.MY	h MODE D.MY	\α	Sets the format for date display.

Name	Keys to press	in modes	Remarks
D→J	h X.FCN D→J	FLOAT	Takes x as a date in the format selected and converts it to a Julian day number.
D→R	h X.FCN D→R	FLOAT	Takes x as degrees and converts them to radians. Angular mode is kept.
E	E (the key)	FLOAT	Like EEX in vintage calculators.
E3OFF	h MODE E3OFF	\α	Toggle the thousands separator (either a point or a comma depending on the radix setting).
E3ON	h MODE E3ON		
ENG	h MODE ENG ***n***	\α	Sets engineering display format.
ENTER↑	ENTER↑	\α	See above for ^{C}ENTER.
ERF	h STAT ERF	FLOAT	Calculates the error function erf(x).
e^x	f e^x	FLOAT	
ExpF	h STAT ExpF	FLOAT	Selects the exponential curve fit model.
Ex(t)	h PROB Ex(t)	FLOAT	= EXPONDIST(x; j; 1) in MS Excel, with J containing the rate λ. Ex^{-1}(p) returns the survival time t_s for a given probability p in X.
Ex^{-1}(p)	h PROB Ex^{-1}(p)		
e^x -1	h X.FCN e^x-1	FLOAT	
FB	h X.FCN FB ***n***	Integer	Inverts ("flips") the specified bit in x .
FC?	h TEST FC? ***n*** etc.	\α	Tests the flag specified. Clears, flips, or sets this flag after testing, if applicable.
FC?C			
FC?F			
FC?S			
FF	h P.FCN FF ***n***	\α	Flips the flag specified.
FIB	h X.FCN FIB	\α	Calculates the Fibonacci number F_x .
FILL	g FILL	\α	Copies x to all other stack levels. See above for ^{C}FILL.
FIX	h MODE FIX ***n***	\α	Sets fixed point display format.
FLOAT	f H.d	\α	Works like DECM in HP-42S. See BASE0.
FLOOR	h X.FCN FLOOR	FLOAT	Returns the largest integer $\leq x$.
FP	g FP	FLOAT	Returns the fractional part of x .
FS?	h FS? ***n***	\α	Tests the flag specified.

Name	Keys to press	in modes	Remarks
FS?C	h TEST FS?C ***n*** etc.	\α	Tests the flag specified. Clears, flips, or sets this flag after testing, if applicable.
FS?F			
FS?S			
F(x)	h PROB F(x)	FLOAT	F works like Q(F), F^{-1} like F_P in HP-21S. The degrees of freedom are given in **J** and **K**.
F^{-1}(p)	h PROB F^{-1}(p)		
GCD	h X.FCN GCD	\α	Returns the Greatest Common Divisor of ***x*** and ***y*** .
Ge(x)	h PROB Ge(x)	FLOAT	Geometric distribution: $Ge(x)=1-(1-p_0)^x$ is the probability for a 1st success after ***x*** Bernoulli experiments. The probability $\boldsymbol{p_0}$ for a success in each such experiment must be given in **J**. Ge^{-1}(p) returns the number of failures ***f*** before the 1st success for a given probability ***p*** in **X**.
Ge^{-1}(p)	h PROB Ge^{-1}(p)		
GRAD	g GRAD	FLOAT	Sets angular mode to gon or grads.
GTO	h GTO ***label***	PRG	Inserts an unconditional branch to ***label***.
		\PRG, \α	Positions the program pointer to ***label***.
	h GTO . ***nnn***	\α	Positions the program pointer to line ***nnn*** or to line 000, respectively (not programmable).
	h GTO . .		
H.MS	f H.MS	FLOAT	Sets H.MS mode for time calculations. Display is formatted as shown above.
H.MS+	+	H.MS	Assumes **X** and **Y** containing times in the format `hhhh°mm'ss.dd"` , and adds or subtracts them, respectively.
H.MS–	−		
IMPFRC	g d/c	\α	Sets fraction mode allowing improper fractions in display (i.e. 5/3 instead of 1 2/3). Converts ***x*** according to the settings by DEN… Absolute decimal equivalents of ***x*** must be >1E-5 and <1E5. Compare PROFRC.
		FRC	Allows displaying improper fractions. Thus converts a proper fraction in **X** into the equivalent improper fraction, if applicable.
IP	f IP	FLOAT	Returns the integer part of ***x*** .
ISG	g ISG ***r***	PRG	Given `cccccc.fffii` in ***r*** , this function increments ***r*** by `ii`, skipping next program line if then `ccccccc` > `fff` for ISG, or = 0 for ISZ.
ISZ	h P.FCN ISZ ***r***		

Name	Keys to press	in modes	Remarks
I β	h X.FCN Iβ	FLOAT	Calculates the regularized incpl. beta function $\frac{B(x,y,z)}{\beta(y,z)}$ with $B(x,y,z)=\int_0^x t^{y-1}(1-t)^{z-1}dt$.
I Γ	h X.FCN IΓ	FLOAT	Calculates the regularized incomplete gamma function $\frac{\gamma(x,y)}{\Gamma(x)}$ with $\gamma(x,y)=\int_0^y t^{x-1}e^{-t}dt$.
J→D	h X.FCN J→D	FLOAT	Takes ***x*** as a Julian day number and converts it to a date in the format selected.
LASTx	RCL L	\α	See *above* for ^{C}LASTx .
LBL	f LBL ***label***	PRG	Identifies programs and routines for execution and branching. See opportunities for specifying ***label*** in the table *above*.
LCM	h X.FCN LCM	\α	Returns the Least Common Multiple of ***x*** and ***y***.
LEAP?	h TEST LEAP?	FLOAT	Takes ***x*** as a date in the format selected, extracts the year, and tests for a leap year.
LinF	h STAT LinF	FLOAT	Selects the linear curve fit model.
LJ	h X.FCN LJ	Integer	
LN	g LN	FLOAT	
LN1+X	h X.FCN LN1+X	FLOAT	
LN β	h X.FCN LNβ	FLOAT	Calculates the natural logarithm of β(***x***, ***y***) or Γ(***x***), respectively. See these functions.
LN Γ	h X.FCN LNΓ		
LOG₁₀	g LG	FLOAT	
LOG₂	g LB	\α	Calculates the logarithm of ***x*** for base 2.
LogF	h STAT LogF	FLOAT	Selects the logarithmic curve fit model.
LOGy	g LOGy	FLOAT	Calculates the logarithm of ***x*** for base ***y*** .
	CPX g LOGy	FLOAT	Calculates the logarithm of the complex number ***x*** + *i* ***y*** for the complex base ***z*** + *i* ***t*** .
L.R.	h L.R.	FLOAT	Calculates the parameters ***a1*** and ***a0*** of the fit curve through the data points accumulated, according to the model selected, and pushes them on the stack. For a straight regression line, ***a0*** is the y-intercept and ***a1*** the slope.

Name	Keys to press	in modes	Remarks
MASKL	h X.FCN MASKL ***n***	Integer	Work like MASKL and MASKR on HP-16C, but with the mask length following the command instead of taken from **X**.
MASKR	h X.FCN MASKR ***n***		
MAX	h X.FCN MAX	\α	Returns the maximum of ***x*** and ***y*** .
MIN	h X.FCN MIN	\α	Returns the minimum of ***x*** and ***y*** .
MIRROR	h X.FCN MIRROR	Integer	Reflects the bit pattern in ***x*** (e.g. 000101 becomes 101000 for word size 6).
MOD	h MOD	\α	MOD of HP-42S equals RMD of HP-16C.
M.DY	h MODE M.DY	\α	Sets the format for date display.
NAND	h X.FCN NAND	\α	Works in analogy to AND.
NaN?	h TEST NaN?	\α	Tests ***x*** for “Not a Number”.
nBITS	h X.FCN nBITS	Integer	Counts bits set in ***x*** like #B does on HP-16C.
NOP	h P.FCN NOP	PRG	
NOR	h X.FCN NOR	\α	Works in analogy to AND.
NOT	h NOT	\α	Works in analogy to AND.
nΣ	h STAT nΣ	FLOAT	Recalls the number of accumulated data points. Necessary for basic statistics.
N(x)	h PROB N(x)	FLOAT	= NORMDIST(***x; j; k;* 1**) in MS Excel, with the mean ***j*** and the standard deviation ***k***.
$N^{-1}(p)$	h PROB $N^{-1}(p)$	FLOAT	= NORMINV(***x; j; k***) . See N(x) for more.
OR	h OR	\α	Works in analogy to AND.
PAUSE	h PSE	PRG	Pauses program execution for about 1 s.
PERM	g Py,x	FLOAT	Returns the number of possible <u>arrangements</u> of ***y*** items taken ***x*** at a time. No item occurs more than once in an arrangement, and different orders of the same ***x*** items <u>are</u> counted separately. Formula: $P_{y,x} = x! \cdot C_{y,x}$, see COMB and FACT.
PowerF	h STAT PowerF	FLOAT	Selects the power curve fit model.
PRIME?	h TEST PRIME?	\α	Checks if the absolute value of the integer part of ***x*** is a prime number. Exact for $x < 66049$, Miller-Rabin with 40 iterations otherwise, with the probability ***P*** for erroneously claiming a composite is prime being $P \approx 2^{-80} \approx 10^{-24}$.

Name	Keys to press	in modes	Remarks
PROFRC	f a b/c	FLOAT	Sets fraction mode like in HP-35S, allowing only proper fractions or mixed numbers in display. Converts x according to the settings by DEN… Absolute decimal equivalents of x must be >1E-5 and <1E5. Compare IMPFRC.
		FRC	Allows displaying only proper fractions. Thus converts an improper fraction in **X**, e.g. 5/3 into 1 2/3, if applicable.
PROMPT	h P.FCN PROMPT	PRG	Displays ***alpha*** and stops program execution (equaling αVIEW followed by STOP actually). With a program running, enter the value requested and press R/S to continue.
P(x)	h PROB P(x)	FLOAT	= POISSON(***x; j*k;*** **1**) in MS Excel, with the gross error probability ***j*** and the sample size ***k***. Alternatively, the Poisson parameter λ may be in **J** if ***k*** = 1.
$P^{-1}(p)$	h PROB $P^{-1}(p)$	FLOAT	P^{-1} returns the number of successes ***g*** for a given probability ***p*** in **X**. See P(x) for more.
Q(x)	f Q	FLOAT	Works like Q in HP-32E and Q(z) in HP-21S.
Q^{-1}(p)	g Q^{-1}	FLOAT	Works like Q^{-1} in HP-32E and z_P in HP-21S.
RAD	g RAD	FLOAT	Sets angular mode to radians.
RAND#	h STAT RAND#	FLOAT	Returns a random number between 0 and 1 like RAN in HP-42S.
	h X.FCN RAND#	Integer	Returns a random bit pattern within the word size given.
RCL	RCL s	\α	See the *addressing table above* for ^{C}RCL.
RCLM	RCL MODE	\α	Recalls selected mode settings into **X**. See the paragraph about *indicators* above.
RCL+	RCL + s	\α	Recalls the content of address ***s***, executes the specified operation on it and pushes the result on the stack. E.g. RCL–12 recalls ***r12***, subtracts x from it and displays the result. RCL↑ (↓) recalls the maximum (minimum) of the values in ***s*** and **X**. See the *addressing table above* for ^{C}RCL.
RCL–	RCL – s		
RCL×	RCL × s		
RCL/	RCL / s		
RCL↑	RCL ▲ s		
RCL↓	RCL ▼ s		

Name	Keys to press	in modes	Remarks
RDX, / **RDX.**	h ./,	FLOAT	Toggle the radix mark.
RJ	h X.FCN RJ	Integer	Works in analogy to LJ.
RL	h X.FCN RL ***n***	Integer	Works like ***n*** consecutive RLs / RLCs on HP-16C. For RL, 1 ≤ ***n*** ≤ 63. For RLC, 1 ≤ ***n*** ≤ 64. RL 0 and RLC 0 execute as NOP.
RLC	h X.FCN RLC ***n***		
ROUND	g RND	FLOAT	Rounds ***x*** using the current display format, like RND in HP-42S.
		FRC	Rounds ***x*** using the current denominator, like RND in HP-35S.
ROUNDI	h X.FCN ROUNDI	FLOAT	Rounds ***x*** to next integer. ½ rounds to 1.
RR	h X.FCN RR ***n***	Integer	Works like ***n*** consecutive RRs / RRCs on HP-16C. See RL / RLC for more.
RRC	h X.FCN RRC ***n***		
RTN	g RTN	\PRG	Entered from the keyboard: Moves the program pointer to the first line of the routine observed. Compare GTO. In program execution: Returns control to the calling routine, i.e. moves the program pointer to the line following the most recent XEQ instruction encountered. If there is no matching XEQ, program execution halts.
		PRG	Last command in a routine. Will return control to the calling routine in program execution.
RTN+1	n/a	PRG	Internal support routine.
R-CLR	h P.FCN R-CLR	FLOAT	Interprets ***x*** in the form `ss.nn` . Clears ***nn*** registers starting with number ***ss*** . E.g. for ***x*** = 34.56, R-CLR will clear **R34** through **R89**.
R-COPY	h P.FCN R-COPY	FLOAT	Interprets ***x*** in the form `ss.nndd` . Takes ***nn*** registers starting with number ***ss*** and copies their contents to ***dd*** . E.g. for ***x*** = 7.0345678, ***r07, r08, r09*** will be moved into **R45, R46, R47**, respectively.
R-SORT	h P.FCN R-SORT	FLOAT	Interprets ***x*** in the form `ss.nn` . Sorts the contents of ***nn*** registers starting with number ***ss*** . Assume ***x*** = 49.026 , ***r49*** = 1.2 , ***r50*** = -3.4 ; then R-SORT returns ***r49*** = -3.4 , ***r50*** = 1.2.

Name	Keys to press	in modes	Remarks
R-SWAP	h P.FCN R-SWAP	FLOAT	Works like R-COPY but swaps the register contents of source and destination.
R↑	h R↑	\α	Rotates the stack contents one level up or down, respectively. See above for complex rotations.
R↓	R↓		
R→D	h X.FCN R→D	FLOAT	Takes x as radians and converts them to degrees. Angular mode is kept.
s	g s	FLOAT	Calculates the standard deviations s_y and s_x and pushes them on the stack.
SB	h X.FCN SB ***n***	Integer	Sets the specified bit in x .
SCI	h MODE SCI ***n***	\α	Sets scientific display format.
SEED	h STAT SEED	FLOAT	Stores a seed for random number generation.
SERR	h STAT SERR	FLOAT	Pushes $s_y / \sqrt{n}$ and $s_x / \sqrt{n}$ on the stack.
SETDAT	h X.FCN SETDAT	FLOAT^H	Sets the date or time, respectively, for the real time clock.
SETTIM	h X.FCN SETTIM		
SF	h SF ***n***	\α	Sets the flag specified.
SIGN	h X.FCN SIGN	\α	Returns 1 for $x > 0$, –1 for $x < 0$, and 0 for $x = 0$ or non-numbers.
	CPX h ...	FLOAT	Returns the unit vector of $x + i\,y$ in $\mathbf{X}$ and $\mathbf{Y}$.
SIGNMT	h MODE SIGNMT	\α	Sets sign-and-mantissa mode for integers.
SIN	f SIN	FLOAT^H	
SINC	h X.FCN SINC	FLOAT	Calculates $\frac{\sin(x)}{x}$.
SINH	f HYP SIN	FLOAT	
SL	h X.FCN SL ***n***	Integer	Works like ***n*** (up to 63) consecutive SLs on HP-16C. SL 0 executes as NOP.
SLV	f SLV ***label***	FLOAT	Solves the equation f(x) = 0, with f(x) calculated by the routine specified. Two initial estimates of the root must be supplied in $\mathbf{X}$ and $\mathbf{Y}$ when calling SLV. For the rest, the user interface is as in HP-15C.
SR	h X.FCN SR ***n***	Integer	Works like ***n*** consecutive SRs on HP-16C. SR 0 executes as NOP.

Name	Keys to press	in modes	Remarks
SSIZE4	h MODE SSIZE4	\α	Sets the stack size to 4 or 8 levels, respectively. If stack size grows, the top level contents will be copied into the new levels. If the stack shrinks, previous top levels will be lost. – The same will happen if the stack size is changed via STOM.
SSIZE8	h MODE SSIZE8		
SSIZE?	h TEST SSIZE?	\α	Returns the number of stack levels accessible.
STO	STO ***d***	\α	See the *addressing table above* for ^{C}STO.
STOM	STO MODE	\α	Sets selected modes as encoded in $\boldsymbol{x}$. See the paragraph about *indicators* above.
STOP	R/S	PRG	Stops program execution.
STO+	STO + ***d***	\α	Executes the specified operation on the content of address ***d*** and stores the result into said address. E.g. STO–12 subtracts $\boldsymbol{x}$ from $\boldsymbol{r12}$, and stores the result in $\mathbf{R12}$ again. STO↑ (↓) takes the maximum (minimum) of the values in ***d*** and $\mathbf{X}$ and stores it. See the *addressing table above* for ^{C}STO.
STO–	STO − ***d***		
STO×	STO × ***d***		
STO/	STO / ***d***		
STO↑	STO ▲ ***d***		
STO↓	STO ▼ ***d***		
SUM	h STAT SUM	FLOAT	Recalls the linear sums $\boldsymbol{\Sigma y}$ and $\boldsymbol{\Sigma x}$. Useful for basic vector algebra.
TAN	f TAN	FLOAT^{H}	
TANH	f HYP TAN	FLOAT	
TIME	h X.FCN TIME	FLOAT^{H}	Recalls the time from the real time clock.
t(x)	h PROB t(x)	FLOAT	t works like Q(t) , t $^{-1}$ like tp in HP-21S. The degree of freedom is stored in $\mathbf{J}$.
t $^{-1}$(p)	h PROB t $^{-1}$(p)		
UNSIGN	h MODE UNSIGN	\α	Sets unsigned mode for integers.
VIEW	h VIEW ***s***	All	Views the contents of address ***s*** .
W	h X.FCN W	FLOAT	W returns Lambert's W for given $\boldsymbol{x}$ ≥ -1/e , while W^{-1} returns $\boldsymbol{x}$ for given W (≥ -1).
W $^{-1}$	h X.FCN W^{-1}		
Wb(t)	h PROB Wb(t)	FLOAT	= WEIBULL($\boldsymbol{x; j; k;}$ **1**) in Excel, with the *shape parameter* ***j*** and the *characteristic lifetime* ***k***.
Wb $^{-1}$(p)	h PROB Wb^{-1}(p)	FLOAT	Wb^{-1} returns the survival time $\boldsymbol{t_s}$ for given probability ***p*** . See Wb(t) for more.

Name	Keys to press	in modes	Remarks
WSIZE	h MODE WSIZE ***n***	\α	Works like WSIZE on HP-16C, but with the parameter following the command instead of taken from **X**. WSIZE 0 sets the word size to maximum, i.e. 64 bits.
WSIZE?	h TEST WSIZE?	\α	Recalls the word size set.
x^2	g x^2	\α	
XEQ	XEQ ***label***	PRG	Calls the respective subroutine.
		\PRG, \α	Executes the respective program.
	B , C , or D (you may need f for accessing these hotkeys in integer bases >10.)	PRG	Calls the respective subroutine, so e.g. XEQ C will be inserted when C is pressed.
		\PRG, \α	Executes the respective program if defined.
XNOR	h X.FCN XNOR	\α	Works in analogy to AND.
XOR	h XOR	\α	Works in analogy to AND.
x !	h !	FLOAT	
x ⇔	h x≷ ***r***	\α	Swaps the contents of **X** and ***r*** . See *above* for complex x⇔ .
x ⇔y	x≷y	\α	Swaps ***x*** and ***y*** , performing Re⇔Im if a complex operation was executed immediately before. See *above* for ^{C}x⇔y .
x → α	g x◀▶α	All	Interprets ***x*** as a code of up to 6 characters. Appends these characters to ***alpha***, similar to XTOA in HP-42S.
x < … ?	h TEST x < ? ***a***	\α	Compare ***x*** with ***a***. The three dots will be replaced in the listing by ***a*** according to the examples given in the *addressing table above*. CPX f x = ? ***a*** and CPX g x ≠ ? ***a*** compare the complex number ***x*** + *i* ***y*** as explained in the *addressing table above*.
x ≤ … ?	h TEST x ≤ ? ***a***		
x = … ?	f x = ? ***a***		
x ≠ … ?	g x ≠ ? ***a***		
x ≥ … ?	h TEST x ≥ ? ***a***		
x > … ?	h TEST x > ? ***a***		
x̄, ȳ	f x̄	FLOAT	Pushes $\frac{1}{n}\sum y$ and $\frac{1}{n}\sum x$ on the stack.
x̄w	h STAT x̄w	FLOAT	Returns the weighted mean $\sum xy \big/ \sum y$.

Name	Keys to press	in modes	Remarks
x̂	h STAT x̂	FLOAT	Returns a forecast ***x*** for a given ***y*** (in **X**) following the fit model chosen. See L.R. for more.
y^x	f y^x	\α	In integer modes ***x*** must be ≥ 0.
	C	\(α, -3, -4, -5, h)	Shortcut working as long as label C is not defined yet.
Y.MD	h MODE Y.MD	\α	Sets the format for date display.
ŷ	f ŷ	FLOAT	Returns a forecast ***y*** (in **X**) for a given ***x*** following the fit model chosen. See L.R. for more.
αDATE	h X.FCN αDATE	\integer	Takes ***x*** as a date and appends it to ***alpha***.
αDAY	h X.FCN αDAY	\integer	Takes ***x*** as a date, recalls the name of the respective day and appends its first 3 letters to ***alpha***.
αIP	h X.FCN αIP	All	Appends the integer part of ***x*** to ***alpha***, similar to AIP in HP-42S.
αLENG	h X.FCN αLENG	All	Returns the number of characters found in ***alpha***, like ALENG in HP-42S.
αMONTH	h X.FCN αMONTH	\integer	Works like αDAY, but processing the month.
αOFF	h P.FCN αOFF	PRG & α	Work like AOFF and AON in HP-42S.
αON	h P.FCN αON	PRG & \α	
αRCL	f RCL ***s***	α	Interprets the content of the source ***s*** as characters and appends them to ***alpha***.
	h X.FCN αRCL ***s***	\α	
αRC#	h X.FCN αRC# ***s***	All	Interprets the content of ***s*** as a number, converts it to a string in the format selected, and appends this to ***alpha***.
αRL	h X.FCN αRL ***n***	All	Rotates ***alpha*** by ***n*** characters like AROT in HP-42S, but with a positive parameter trailing the command instead of taken from **X**. αRL 0 executes as NOP. αRR works like αRL but rotates to the right.
αRR	h X.FCN αRR ***n***		
αSL	h X.FCN αSL ***n***	All	Shifts the ***n*** left-most characters out of ***alpha***, similar to ASHF in HP-42S. αSL 0 executes as NOP.
αSR	h X.FCN αSR ***n***	All	Works like αSL but takes the ***n*** right-most characters instead.

Name	Keys to press	in modes	Remarks
αSTO	f STO ***d***	α	Stores the first 6 characters in the alpha register into destination ***d*** .
	h X.FCN αSTO ***d***	\α	
αTIME	h X.FCN αTIME	FLOAT^H	Takes x as a time and appends it to ***alpha*** in the format selected
αVIEW	h X.FCN αVIEW	\α	Displays ***alpha***. In programs, use αVIEW followed by PAUSE for message output.
α → x	f x◀▶α	All	Returns the character code of the left-most character in ***alpha*** and deletes this character, like ATOX in HP-42S.
β(x,y)	h X.FCN β(x,y)	FLOAT	Calculates Euler's Beta $B(x,y)=\frac{\Gamma(x)\cdot\Gamma(y)}{\Gamma(x+y)}$ with $\operatorname{Re}(x)>0$, $\operatorname{Re}(y)>0$.
Γ(x)	h STAT Γ(x)	FLOAT	
	h X.FCN Γ(x)		
ΔDAYS	h X.FCN ΔDAYS	FLOAT	Assumes $\mathbf{X}$ and $\mathbf{Y}$ containing dates in the format chosen and calculates the number of days between them. Works like in HP-12C.
Δ%	f Δ%	FLOAT	Calculates $100\cdot\frac{x-y}{y}$ like %CH in HP-42S.
π	h π	FLOAT	Complex version copies π in $\mathbf{X}$ and clears $\mathbf{Y}$.
Π	f π ***label***	FLOAT	Computes a product with the routine specified by ***label***. Initially, $\mathbf{X}$ contains the loop control number in the format `cccccc.fffii` and the product is set to 1. Each run through the routine specified computes a factor. At its end, this factor is multiplied with said product; the operation then decrements `ccccccc` by `ii` and runs said routine again if `ccccccc` is then > `fff` , else returns the resulting product in $\mathbf{X}$.
Σ	g Σ ***label***	FLOAT	Computes a sum with the routine specified by ***label***. Initially, $\mathbf{X}$ contains the loop control number in the format `cccccc.fffii` and the sum is set to 0. Each run through the routine specified computes a summand; at its end, this is added to said sum; the operation then decrements `ccccccc` by `ii` and runs said routine again if `ccccccc` is then > `fff`, else returns the resulting sum in $\mathbf{X}$.

Name	Keys to press	in modes	Remarks
σ	h STAT σ	FLOAT	Works like s but calculates the standard deviation of the population instead.
$\Sigma\ln^2x$	h STAT $\Sigma\ln^2x$ etc.	FLOAT	Recall the respective statistical sums. These sums are necessary for curve fitting models beyond pure linear. Calling them by name enhances readability of programs significantly.
$\Sigma\ln^2y$			
$\Sigma\ln x$			
$\Sigma\ln xy$			
$\Sigma\ln y$			
$\Sigma x\ln y$			
$\Sigma y\ln x$			
Σx	h STAT Σx etc.	FLOAT	Recall the respective statistical sums. These sums are necessary for basic statistics and linear curve fitting. Calling them by name enhances readability of programs significantly.
Σx^2			
Σxy			
Σy			
Σy^2			
Σ+	Σ+	FLOAT	
Σ–	h Σ–		
$\chi^2(x)$	h PROB $\chi^2(x)$	FLOAT	χ^2 works like $Q(\chi^2)$, the inverse like χ^2_p in HP-21S. The degree of freedom is given in **J**.
χ^2INV	h PROB χ^2INV		
+, –, ×, /	+, –, ×, /	\α	
+/–	+/–		
/ /	g / /	FLOAT	Calculates $\left(\frac{1}{x}+\frac{1}{y}\right)^{-1}$.
[.] or **[,]**	.	FLOAT	Inserts the radix mark as selected.
		α	Inserts a point.
[.]	.	D.MY etc.	Separates the leading unit in date modes. The user may decide where a number represents a date.
[ ] or **[/]**	.	FRC	First . is interpreted as a space, 2nd as a fraction mark. E.g. 2 . 3 . 4 results in 2 ¾ in the dot matrix display. Improper fractions may be entered starting with a . .

Name	Keys to press	in modes	Remarks
[°]	.	H.MS	Separates degrees (hours) from minutes and seconds, so input format is `hhhh.mmssdd`.
%	g %	FLOAT	Calculates $\frac{x \cdot y}{100}$.
%MG	h X.FCN h % MG	FLOAT	Calculates the margin [15] $100 \cdot \frac{x-y}{x}$ in % for a price x and cost y , like %MU-Price in HP-17B.
%MRR	h X.FCN h % MRR	FLOAT	Calculates the mean rate of return in % per period, i.e. $100 \cdot \left[\left(\frac{x}{y} \right)^{1/z} - 1 \right]$ with x = FV = future value after z periods, y = PV = present value. For $z = 1$, Δ% returns the same result easier.
%T	h X.FCN h % T	FLOAT	Calculates $100 \cdot \frac{x}{y}$, i.e. percent of total FWIW.
%Σ	h X.FCN h % Σ	FLOAT	Calculates $100 \cdot \frac{x}{\sum x}$.
%+	h %+	FLOAT	Adds a markup of x % to a price y , calculating $y \cdot \left(1 + \frac{x}{100} \right)$ like in %MU-Cost of HP-17B.
%+MG	h X.FCN h % +MG	FLOAT	Adds a margin [15] of x % to the cost y , calculating a sales price $y \Big/ \left(1 - \frac{x}{100} \right)$, as %MU-Price does in HP-17B.
%–	h %–	FLOAT	Subtracts a discount of x % from the price y , calculating $y \cdot \left(1 - \frac{x}{100} \right)$.
√‾	f √x	\α	
	D	\(α, -4, -5, h)	Shortcut working as long as label D is not defined yet.
∫	g ∫ ***label***	FLOAT	Integrates the function given in the routine specified. Lower and upper integration limits must be supplied in **Y** and **X**, respectively. Otherwise, the user interface is as in HP-15C.

[15] Margin corresponds to „Handelsspanne“ in German.

Name	Keys to press	in modes	Remarks
∞?	h TEST ∞?	\α	Tests x for infinity.
→DEG	→ g DEG	FLOAT	Takes x as an angle in the angular mode set and converts it to degrees. Angular mode is kept.
→GRAD	→ g GRAD	FLOAT	Works like →DEG, but converts to grads.
→H	→ f H.d	H.MS	Takes x as hours or degrees in the format `hhhh°mm'ss.dd"` and converts them into decimal numbers.
→H.MS	→ f H.MS	FLOAT	Takes x as *decimal* hours or degrees and converts them into `hhhh°mm'ss.dd"`.
→POL	g R◀▶P	FLOAT	Assumes **X** and **Y** containing Cartesian coordinates (x, y) and converts them to the respective cylinder coordinates (r, ϑ) .
→RAD	→ g RAD	FLOAT^H	Works like →DEG, but converts to radians.
→REC	f R◀▶P	FLOAT	Assumes **X** and **Y** containing cylinder coordinates (r, ϑ) and converts them to the respective Cartesian coordinates (x, y) .

Non-programmable control, clearing and information commands:

Name	Keys to press	in modes	Remarks
	g OFF	All	Turns calculator off.
	ON	Calc. off	Turns calculator on.
	▲ / ▼	Status open	Goes to previous / next set of flags.
		Cat. open	Goes to previous / next item in this catalogue.
		α	Shifts the display window to the left / right in ***alpha*** if possible. Useful for longer strings.
		Else	Acts like BST / SST in HP-42S.
	f ◀(/ g)▶	Integer	Shifts the display window to the left / right like in HP-16C. Useful while working with small bases.
		α	Appends a left / right parenthesis to ***alpha***.
	f ⇧	α	Toggles upper and lower case.

Name	Keys to press	in modes	Remarks
	⬅	Input pending	Deletes the last digit or character put in.
		α	Deletes the rightmost character in ***alpha***.
		PRG	Deletes current step.
		Else	Acts like CLx.
	f α	\α	Toggle alpha mode for keyboard entry.
	ENTER↑	α	
CLALL	h X.FCN CLALL	\PRG	Clears all registers and programs if confirmed.
CLPR	h CLPR	\α	Clears the current program (i.e. the one the program pointer is in) after confirmation.
EXIT	EXIT	All	Exits catalogues and other operations with pending input, canceling the execution of said operation.
PRGOFF	h P/R	PRG	Leaves programming mode.
PRGON	h P/R	\PRG, \α	Enters programming mode.
RESET	h X.FCN RESET	All	Clears all registers and programs if confirmed, and resets all modes to start-up default, being 24h, 2COMPL, DEG, DENANY, DENMAX 9999, FIX 4, FLOAT, LinF, PROFRC, SSIZE4, WSIZE 64, Y.MD.
R/S	R/S	\PRG, \α	Runs a program (starting with the current program line) or stops a running program immediately.
SHOW	h SHOW	FLOAT & \PRG	Shows the full mantissa until the key EXIT is released.
		PRG	Displays a CRC-32 checksum of program memory contents (8 hex digits), allowing validation of program integrity.
STATUS	h STATUS	\PRG	Shows the status of user flags, similar to STATUS on HP-16C. See above.
VERS	h X.FCN VERS	\PRG	Shows the firmware version.
→BIN	→ f 2	\α	These commands show x in target integer representation until the key +/_ or E is released, respectively. Base is kept as set. If used in integer bases 15 and 16, prefix f must precede the key →.
→DEC	→ f 10		
→HEX	→ g 16		
→OCT	→ g 8		

Catalogues (not programmable):

Calling a catalogue will set temporary alpha mode to allow for typing the first 1 or 2 characters of the item wanted. ▲ and ▼ browse the catalogue, XEQ selects the item displayed and exits, while EXIT leaves the catalogue without executing anything, returning to the mode as set before. See the *table above about addressing catalogued items*, and the *next paragraph* for detailed item lists.

Name	Keys to press	in modes	Contents
CONST	h CONS	FLOAT	Constants like in HP35s. See them listed in a *table below*. CPX h CONS will clear **Y** in recalling the constant selected.
CONV	h CONV	FLOAT	Conversions as listed in a *table below*.
CPX	f CPX	α	"Complex" letters mandatory for many languages. Upper or lower case will be displayed according to setting (see f ⇧ above).
MODE	h MODE	\α	Mode setting functions.
PROB	h PROB	FLOAT	Extra probability distribution functions.
P.FCN	h P.FCN	\α	Extra programming functions.
STAT	h STAT	FLOAT	Extra statistical functions.
		α	Special letters for statistics, and subscripts.
TEST	h TEST	\α	All tests except the two on the keyboard.
		α	Comparison symbols and brackets. Parentheses are called by f (and g), respectively.
X.FCN	h X.FCN	FLOAT	Extra real functions.
		Integer	Extra integer functions.
		α	Extra alpha functions.
	CPX h …	FLOAT	Extra complex functions.
. / ,	h ./.	α	Punctuation marks and text symbols.
→	f →	α	Arrows, mathematical symbols, and superscripts.

DETAILED CATALOGUE CONTENTS

The operations contained in the catalogues MODE, STAT, PROB, P.FCN, TEST, and X.FCN are listed in the following table. A single function, e.g. CB, may be contained in more than one catalogue. The characters necessary to access a specific function in the respective catalogue are printed bold in this table – ▼ has to be pressed once for each character printed red – if even the last letter of a function name is red, one may need more strokes of ▼ to access this function. See also the catalogues CONST and CONV in separate paragraphs below.

The alpha contents of catalogues are found on the page after next page.

MODE	STAT	PROB	P.FCN	TEST
12h	**B**estF	**B**(x)	**C**LFLAG	**B**C?
1COMPL	**E**RF	**B** $^{-1}$(p)	**D**SZ	**BS**?
24h	**E**xpF	**E**x(x)	**F**F	**F**C?
2COMPL	**L**inF	**Ex** $^{-1}$(p)	**I**SZ	**FC**?C
ALL	**L**ogF	**F**(x)	**N**OP	**FC?**F
BASE	**n**Σ	**F** $^{-1}$(p)	**P**ROMPT	**FC?S**
DENANY	**P**owerF	**G**e(x)	**R**CLM	**FS**?C
DENFAC	**R**AND#	**Ge** $^{-1}$(p)	**R-**CLR	**FS?**F
DENFIX	**SE**ED	**N**(x)	**R-C**OPY	**FS?S**
DENMAX	**SER**R	**N** $^{-1}$(p)	**R-SO**RT	**L**EAP?
D.MY	**SU**M	**P**(x)	**R-SWA**P	**N**aN?
ENG	**x̄**w	**P** $^{-1}$(p)	**S**TOM	**P**RIME?
E3OFF	**x̂**	**t**(x)	αOFF	**S**SIZE?
E3ON	**Γ**(x)	**t** $^{-1}$(p)	α**O**N	**W**SIZE?
FIX	**σ**	**W**b(t)		**x** < ?
M.DY	**Σl**n^{2}x	**Wb** $^{-1}$(p)		**x ≤** ?
SCI	**Σln**^{2}y	χ^2(x)		**x ≥ ?**
SIGNMT	**Σlnx**	χ^{2}INV		**x > ?**
SSIZE4	**Σlnxy**			**∞**?
SSIZE8	**Σlny**			
UNSIGN	**Σx**			
WSIZE	**Σx^2**			
Y.MD	**Σxl**ny			
	Σxy			
	Σy			
	Σy^2			
	Σylnx			

X.FCN varies with the mode set; it contains in …				
… alpha mode:	… FLOAT:		… integer modes:	
	ANGLE	SIGN	ASR	RJ
CLALL	CEIL	SINC	CB	RL
CLREG	CLALL	TIME	CLALL	RLC
RESET	CLREG	VERS	CLREG	RR
VERS	DATE	W	DBLR	RRC
αDATE	DAY	W^{-1}	DBL*	SB
αDAY	DAYS+	XNOR	DBL/	SIGN
αIP	DECOMP	αDATE	FB	SL
αLENG	D→J	αDAY	FIB	SR
αMONTH	D→R	αIP	GCD	VERS
αRC#	e^x -1	αLENG	LCM	XNOR
αRL	FIB	αMONTH	LJ	αIP
αRR	FLOOR	αRCL	MASKL	αLENG
αSL	GCD	αRC#	MASKR	αRCL
αSR	Iβ	αRL	MAX	αRC#
αTIME	IΓ	αRR	MIN	αRL
	J→D	αSL	MIRROR	αRR
	LCM	αSR	NAND	αSL
	LN1+x	αSTO	nBITS	αSR
	LNβ	αTIME	NOR	αSTO
	LNΓ	αVIEW	RAND#	αVIEW
	MAX	β(x,y)	RESET	
	MIN	Γ(x)		
	NAND	ΔDAYS		
	NOR	%MG		
	RESET	%MRR		
	ROUNDI	%T		
	R→D	%Σ		
	SETDAT	%+MG		
	SETTIM			

CPX X.FCN
^{C}CONJ
^{C}e^x -1
^{C}FIB
^{C}LN1+x
^{C}LNβ
^{C}LNΓ
^{C}SIGN
^{C}SINC
^{C}W
^{C}W^{-1}
^{C}β(x,y)
^{C}Γ(x)

Here are the contents of the alpha catalogues – big font in left column or upper row, small font right or lower – making the 34S the most versatile global calculator known:

CPX					
À	À	À	à	à	à
Á	Á	Á	á	á	á
Â Ã Ā Ă	Ā	Ā	â ã ā ă	ā	ā
Ä	Ä	Ä	ä (ă)	ä	ä
Å	Å	Å	å	å	å
Ć	Ć	Ć	ć	ć	ć
Č	Č	Č	č	č	č
Ç	Ç	Ç	ç	ç	ç
È	È	È	è	è	è
É	É	É	é	é	é
Ê Ē Ĕ Ě	Ē	Ē	ê ē ĕ ě	ē	ē
Ë	Ë	Ë	ë (ĕ)	ë	ë
Ì	Ì	Ì	ì	ì	ì
Í	Í	Í	í	í	í
Î Ĩ Ī Ĭ	Ī	Ī	î ĩ ī ĭ	ī	ī
Ï	Ï	Ï	ï (ĭ)	ï	ï
Ñ Ň	Ñ	Ñ	ñ ň	ñ	ñ
Ò	Ò	Ò	ò	ò	ò
Ó	Ó	Ó	ó	ó	ó
Ô Õ Ō Ŏ	Ō	Ō	ô õ ō ŏ	ō	ō
Ö	Ö	Ö	ö (ŏ)	ö	ö
Ř	Ř	Ř	ř	ř	ř
Š	Š	Š	š	š	š
			ß	ß	ß
Ù	Ù	Ù	ù	ù	ù
Ú	Ú	Ú	ú	ú	ú
Û Ũ Ū Ŭ	Ū	Ū	û ũ ū ŭ	ū	ū
Ü	Ü	Ü	ü (ŭ)	ü	ü
Ů	Ů	Ů	ů	ů	ů
Ý	Ý	Ý	ý	ý	ý
Ÿ	Ÿ	Ÿ	ÿ	ÿ	ÿ
Ž	Ž	Ž	ž	ž	ž

STAT
x̄ x̂ ȳ ŷ 0 1 2 A B c e k m n p u µ ∞
x̄ x̂ ȳ ŷ 0 1 2 A B c e k m n p u µ ∞
x̄ x̂ ȳ ŷ 0 1 2 A B c e k m n p u µ ∞

→
→ ← ∫ ∞ ℏ ^ 0 -1 2 X
→ ← ∫ ∞ ℏ ^ 0 -1 2 X
→ ← ∫ ∞ ℏ ^ 0 -1 2 X

TEST
< ≤ = ≥ > [ ] { }
< ≤ = ≥ > [ ] { }
< ≤ = ≥ > [ ] { }

. / ,
, : ; “ # ‘ * @ _ ~
, : ; " # ' * @ _ ~
, : ; " # ' * @ _ ~

The letters provided here allow for correct writing Afrikaans, Català, Cebuano, Česky, Cymraeg, Deutsch, Eesti, English, Español, Euskara, Français, Gaeilge, Galego, Bahasa Indonesia, Italiano, Basa Jawa, Kiswahili, Kreyòl ayisyen, Magyar, Bahasa Melayu, Nederlands, Português, Quechua, Shqip, Slovenčina, Slovenščina, Basa Sunda, Suomeksi, Svenska, Tagalog, Winaray, Zhōngwén (with a little trick explained below), and almost Hrvatski and Srpski (sorry, no đ) as well as Dansk and Norsk (neither æ nor ø). If you know further languages covered, please tell us.

Mandarin Chinese (Zhōngwén) features four tones, usually transcribed like e.g. mā, má, mă, and mà. So you need different letters for ā and ă here, and for e, i, o, and u, too. With 6 pixels total character height we found no way to display them in both fonts nicely, keeping letters and accents separated. For an unambiguous solution, we suggest using dieresis (else not employed in Hànyŭ pīnyīn) for the third tone here.

CONSTANTS

This lists the contents of the catalogue CONST. Values of physical constants (*incl. their relative standard deviations given in parentheses below*) are from CODATA 2006, copied in August 2010. Green background denotes exact or almost exact values. The more the background turns to red, the less precise the respective constant is known.

The characters necessary to get to a specific function in the catalogue are printed bold in this index – ▼ has to be pressed once for each character printed red.

For the units, remember Tesla with $1T = 1\frac{Wb}{m^2} = 1\frac{V \cdot s}{m^2}$, Joule with $1J = 1N \cdot m = 1\frac{kg \cdot m^2}{s^2}$ and on the other hand $1J = 1W \cdot s = 1V \cdot A \cdot s = \frac{1}{e} eV \approx 6.24 \cdot 10^6 TeV$. Thus $1\frac{J}{T} = 1A \cdot m^2$.

	Numeric value	Unit	Remarks
a	365.2425	d	Gregorian year (per definition)
a$_0$	5.2917720859**E**-11 *(6.8**E**-10)*	m	Bohr radius $= \frac{\alpha}{4\pi \cdot R_\infty}$
c	2.99792458**E**8	m/s	Vacuum speed of light (per definition)
c$_1$	3.74177118**E**-16 *(5.0**E**-8)*	$m^2 \cdot W$	First radiation constant $= 2\pi \cdot h \cdot c^2$
c$_2$	0.014387752 *(1.7**E**-6)*	$m \cdot K$	Second radiation constant $= hc/k$
e	1.602176487**E**-19 *(2.5**E**-8)*	C	Electron charge $= \frac{2}{K_J R_K} = \Phi_0 G_0$
eE	2.718281828459045…	1	Euler's e. Please note the letter e is used for the electron charge elsewhere in this table.
F	96485.3399 *(2.5**E**-8)*	$\frac{C}{mol}$	Faraday's constant $= e N_A$
g	9.80665	m/s^2	Standard earth acceleration (per definition)
G	6.67428**E**-11 *(1.0**E**-4)*	$\frac{m^3}{kg \cdot s^2}$	Newton's gravitation constant
G$_0$	7.7480917004**E**-5 *(6.8**E**-10)*	$1/\Omega$	Conductance quantum $= 2e^2/h = 2/R_K$ with the von Klitzing constant R$_K$ = 25812.807557 Ω
g$_e$	2.0023193043622 *(7.4**E**-13)*	1	Landé's g-factor

	Numeric value	Unit	Remarks
h	6.62606896**E**-34 *(5.0**E**-8)*	$J\ s$	Planck constant
$\hbar$	1.054571628**E**-34 *(5.0**E**-8)*		$= h / 2\pi$
k	1.3806504**E**-23 *(1.7**E**-6)*	J / K	Boltzmann constant $= R / N_A$
m$_e$	9.10938215**E**-31 *(5.0**E**-8)*	kg	Electron mass
m$_n$	1.674927211**E**-27 *(5.0**E**-8)*		Neutron mass
m$_p$	1.672621637**E**-27 *(5.0**E**-8)*		Proton mass
m$_u$	1.660538782**E**-27 *(5.0**E**-8)*		Atomic unit mass $= 10^{-3}\ kg / N_A$
m$_\mu$	1.88353103**E**-28 *(5.6**E**-8)*		Muon mass
N$_A$	6.02214179**E**23 *(5.0**E**-8)*	$1 / mol$	Avogadro's number
p$_0$	101325	Pa	standard atmospheric pressure (per definition)
R	8.314472 *(1.7**E**-6)*	$\frac{J}{mol \cdot K}$	Molar gas constant
r$_e$	2.8179402894**E**-15 *(2.1**E**-9)*	m	Classical electron radius $= \alpha^2 \cdot a_0$
R$_\infty$	1.0973731568527**E**7 *(6.6**E**-12)*	$1 / m$	Rydberg constant $= \alpha^2 m_e c / 2h$
T$_0$	273.15	K	= 0°C, standard temperature (per definition)
t$_p$	5.39124**E**-44 *(5.0**E**-5)*	s	Planck time $= \sqrt{\hbar G / c^5}$
V$_m$	0.022413996 *(1.7**E**-6)*	m^3 / mol	Molar volume of an ideal gas at standard conditions $= RT_0 / p_0$
Z$_0$	376.730313461…	Ω	Characteristic impedance of vacuum $= \sqrt{\frac{\mu_0}{\varepsilon_0}} = \mu_0 c$
α	7.2973525376**E**-3 *(6.8**E**-10)*	1	Fine-structure constant $= \frac{e^2}{4\pi\varepsilon_0 \hbar c} \approx \frac{1}{137}$
γEM	0.57721566490153286…	1	Euler-Mascheroni constant
γ$_p$	2.675222099**E**8 *(2.6**E**-8)*	$\frac{1}{s \cdot T}$	Proton gyromagnetic ratio $= 2\mu_P / \hbar$

	Numeric value	Unit	Remarks
ε_0	8.854187817…**E**-12	$\frac{A \cdot s}{V \cdot m}$ or F/m	Electric constant, vacuum permittivity $= \frac{1}{\mu_0 c^2}$
λ_c	2.4263102175**E**-12 *(1.4**E**-9)*	m	Compton wavelength of the electron $= h/m_e c$
λ_{cn}	1.3195908951**E**-15 *(1.5**E**-9)*		Compton wavelength of the neutron $= h/m_n c$
λ_{cp}	1.3214098446**E**-15 *(1.9**E**-9)*		Compton wavelength of the proton $= h/m_p c$
μ_0	1.2566370614…**E**-6	$\frac{V \cdot s}{A \cdot m}$	Magnetic constant, also known as vacuum permeability $= 4\pi \cdot 10^{-7} \frac{V \cdot s}{A \cdot m}$ (per definition)
μ_B	9.27400915**E**-24 *(2.5**E**-8)*	J/T or $A \cdot m^2$	Bohr's magneton $= e\hbar/2m_e$
μ_e	-9.28476377**E**-24 *(2.5**E**-8)*		Electron magnetic moment
μ_n	-9.6623641**E**-27 *(2.4**E**-7)*		Neutron magnetic moment
μ_p	1.410606662**E**-26 *(2.6**E**-8)*		Proton magnetic moment
μ_u	5.05078324**E**-27 *(2.5**E**-8)*		Nuclear magneton $= e\hbar/2m_p$
μ_μ	-4.49044786**E**-26 *(3.6**E**-8)*		Muon magnetic moment
π	3.141592653589793…	1	
σ_B	5.6704**E**-8 *(7.0**E**-6)*	$\frac{W}{m^2 K^4}$	Stefan-Boltzmann constant $= \frac{2\pi^5 k^4}{15 h^3 c^2}$
Φ	1.61803398874989485…	1	Golden ratio $= \frac{1+\sqrt{5}}{2}$
Φ_0	2.067833667**E**-15 *(2.5**E**-8)*	V s	Magnetic flux quantum $= h/2e = 1/K_J$ with the Josephson constant $K_J = 4.83597891 \cdot 10^{14}\, Hz/V$
∞		1	Infinity (may the Lord of Mathematics forgive us calling this a constant)

CONVERSIONS

These are the contents of CONV [16]. The characters necessary to access a specific conversion there are printed bold in this index – ▼ has to be pressed once for each character printed red. The constant **T$_o$** may be useful for conversions, too; it is found in the *catalogue CONST*. The conversion factors or divisors listed in this table will not be seen when executing a conversion.

Conversion		Remarks	Class
acres→ha	* 0.4046873	1 ha = 10^4 m²	Area
atm→Pa	* 1.01325**E**5	Exactly	Pressure
au→km	* 1.495979**E**8	Astronomic units	Length
bhp→W	* 745.6999	British horse power	Power
Btu→J	* 1055.056		Energy
cal→J	* 4.1868	Exactly	Energy
cft→*l*	* 28.31685	Cubic feet	Volume
cm→inch	/ 2.54	Exactly	Length
feet→m	* 0.3048	Exactly	Length
flozUK→*ml*	* 28.41306	1 *ml* = 1 cm^3	Volume
flozUS→*ml*	* 29.57353		Volume
galUK→ *l*	* 4.54609		Volume
galUS→ *l*	* 3.785418		Volume
g→oz	/ 28.34952		Mass
g→tr oz	/ 31.10348		Mass
ha→acres	/ 0.4046873		Area
HP$_e$→W	* 746	Exactly	Power
inch→cm	* 2.54	Exactly	Length
in.Hg→Pa	* 3386.389		Pressure
J→Btu	/ 1055.056		Energy
J→cal	/ 4.1868	Exactly	Energy
J→k Wh	/ 3.6**E**6	Exactly, since 1 h = 3600 s	Energy

[16] For most European readers, many of the units appearing here may look obsolete at least. They die hard, however, in some corners of this world. Anyway, this catalogue provides the means to convert them to real units.

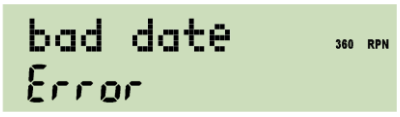
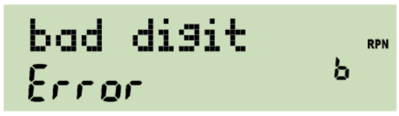
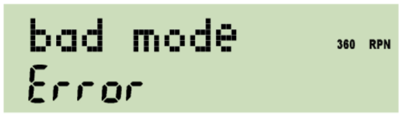
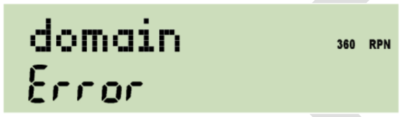
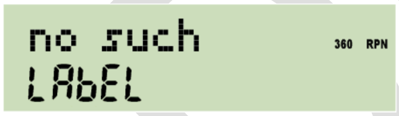
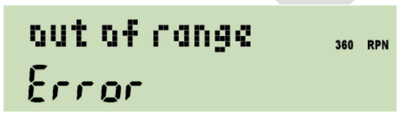
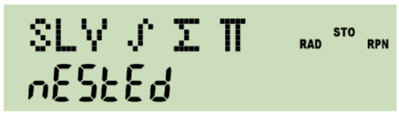
Conversion		Remarks	Class
kg→lbm	/ 0.4535924		Mass
km/h→m/s	/ 3.6	Exactly, since 1 h = 3600 s	Velocity
km→au	/ 1.495979**E**8	Astronomic units	Length
km→***l***.*y*.	/ 9.460730**E**12	Light years	Length
km→**mi**	/ 1.609344	Exactly	Length
km→**nmi**	/ 1.852	Nautical miles, exactly	Length
km→**pc**	/ 3.085678**E**16	Parsec	Length
kWh→J	* 3.6E6	Exactly	Energy
lbf→N	* 4.448222		Force
lbm→kg	* 0.4535924		Mass
***l*.*y*.**→km	* 9.460730**E**12	Light years	Length
l →cft	/ 28.31685	Cubic feet	Volume
l →**g**alUK	/ 4.54609		Volume
l →**ga**lUS	/ 3.785418		Volume
mbar→Pa	* 100	Exactly	Pressure
mi→km	* 1.609344	Exactly	Length
m*l*→flozUK	/ 28.41306		Volume
m*l*→flozUS	/ 29.57353		Volume
m/s→km/h	* 3.6	Exactly	Velocity
m→feet	/ 0.3048	Exactly	Length
m→**y**ards	/ 0.9144	Exactly	Length
nmi→km	* 1.852	Nautical miles, exactly	Length
N→lbf	/ 4.448222		Force
oz→g	* 28.34952		Mass
Pa→atm	/ 1.01325**E**5	Exactly	Pressure
Pa→in.Hg	/ 3386.389		Pressure
Pa→mbar	/ 100	Exactly	Pressure
Pa→**p**si	/ 6894.757		Pressure
Pa→**to**rr	/ 133.3224		Pressure

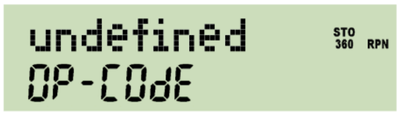
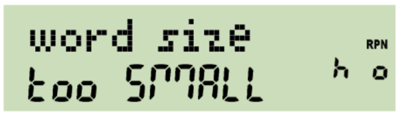
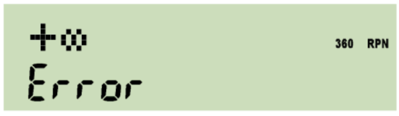
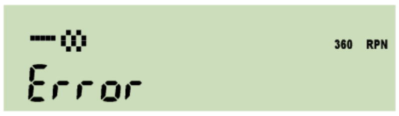
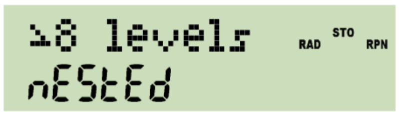
Conversion		Remarks	Class
pc→km	* 3.085678E16	Parsec	Length
PS(hp)→W	* 735.4988		Power
psi→Pa	* 6894.757		Pressure
sh ton→t	* 0.9071847	1 t = 10^3 kg	Mass
ton→t	* 1.016047		Mass
torr→Pa	* 133.3224	1 torr = 1 mm Hg	Pressure
tr oz→g	* 31.10348		Mass
t→sh ton	/ 0.9071847		Mass
t→**t**on	/ 1.016047		Mass
W→bhp	/ 745.6999		Power
W→HP$_e$	/ 746	Exactly	Power
W→PS(hp)	* 735.4988		Power
yards→m	* 0.9144	Exactly	Length
°C→°F	* 1.8 + 32	Exactly	Temperature
°F→°C	- 32) / 1.8	Exactly	Temperature

MESSAGES

There are some commands generating messages also in the dot matrix section of the display. Four of them, DAY, DAYS+, STATUS, and VERS, were introduced above in the *paragraph about display* already. Others are PROMPT, αVIEW and many more alpha commands, and the test commands.

Furthermore, there are a number of error messages. Depending on error conditions, the following messages will be displayed:

Message	Mode(s) allowing this message	Explanation and Examples
 bad date Error 	FLOAT	Invalid date format or incorrect date in input, e.g. month >12, day >31 etc.
 bad digit Error 	Integer	Invalid digit in integer input, e.g. 2 in binary, 9 in octal, or +/- in unsigned mode.
 bad mode Error 	All	Caused by calling an operation in a mode where it is not defined, e.g. SIN in hexadecimal.
 domain Error 	\α	An argument exceeds the domain of the mathematical function called. May be caused by roots or logs of negative numbers (if not preceded by CPX), by 0 / 0, LN(0), Γ(0), TAN(90°) and equivalents, ATANH(x) for $\left\|\mathrm{Re}(x)\right\| \geq 1$, ACOSH(x) for $\mathrm{Re}(x) < 1$, etc.
 no such LAbEL 	All	Attempt to address an undefined label.
 out of range Error 	All	• A number exceeds the valid range. Caused e.g. by specifying decimals >11, word size >64, negative flag numbers, integers ≥2^{64}, hours or degrees >9000, invalid times, denominators ≥9999 etc. • A register address exceeds the valid range. May also happen in indirect addressing. • An R-operation (e.g. R-COPY) attempts exceeding valid register numbers (0 … 99).
 SLV ∫ Σ Π nEStEd 	PRG	Nested use of solve, integrate, sum or product is not allowed.

Message	Mode(s) allowing this message	Explanation and Examples
 undefined OP-CODE	All	An instruction with an undefined op-code occurred (should never happen, but who knows).
 word size too SMALL	Integer, \PRG	Stack or register content is too big for the word size set.
 +∞ Error  −∞ Error	\α, \PRG	• Division of a number > 0 (or < 0) by zero. • Divergent sum or product or integral. • Positive (or negative) overflow in FLOAT.
 >8 levels nESTEd	PRG	Subroutine nesting exceeds 8 levels.

Any key pressed will erase the error message displayed and execute with the stack contents present.

Edit.	Date	Release notes
1	9.12.08	Start
1.1	15.12.08	Added the table of indicators; added NAND, NOR, XNOR, RCLWS, STOWS, //, N, SERR, SIGMA, < and >; deleted HR, INPUT, 2 flag commands, and 2 conversions; extended explanations for addressing and COMPLEX & …; put XOR on the keyboard; corrected errors.
1.2	4.1.09	Added ASRN, CBC?, CBS?, CCB, SCB, FLOAT, MIRROR, SLN, SRN, >BIN, >DEC, >HEX, >OCT, BETA, D>R, DATE, DDAYS, D.MY, M.DY, Y.MD, CEIL, FLOOR, DSZ, ISZ, D>R, R>D, EMGAM, GSB, LNBETA, LNGAMMA, MAX, MIN, NOP, REAL, RJ, W and WINV, ZETA, %+ and %-; renamed the top left keys B, C, and D, and bottom left EXIT.
1.3	17.1.09	Added AIP, ALENG, ARCL, AROT, ASHF, ASTO, ATOX, XTOA, AVIEW, CLA, PROMPT (all taken from 42S), CAPP, FC?C, FS?C, SGMNT, and the …# commands; renamed NBITS to BITS and STOWS to WSIZE; specified the bit commands closer; deleted the 4 carry bit operations.
1.4	10.2.09	Added CONST and a table of constants provided, D>J and J>D, LEAP?, %T, RCL and STO ▲ and ▼, and 2 forgotten statistics registers; deleted CHS, EMGAM, GSB, REAL and ZETA; purged and renamed the bit operations; renamed many commands.
1.5	5.3.09	Added RNDINT, CONV and its table, a memory table, the description of XEQ B, C, D to the operation index, and *a* and g_e to the table of constants; put CLSTK on a key, moved CLΣ and FILL, changed the % and log labels on the keyboard, put CLALL in X.FCN; checked and cleaned alpha mode keyboard and added a temporary alpha keyboard; rearranged the alphabet to put Greek after Latin, symbols after Greek consistently; separated the input and non-programmable commands; cleaned the addressing tables.
1.6	12.8.09	Added BASE, DAYS+, DROP, DROPY, E3OFF, E3ON, FC?F, FC?S, FIB, FS?F, FS?S, GCD, LCM, SETDAT, SETTIM, SET24, SINC, TIME, VERS, αDAY, αMONTH, αRC#; %Σ, as well as F-, t-, and χ^2-distributions and their inverses; reassigned DATE, modified DENMAX, FLOAT, αROT, and αSHIFT; deleted BASE arithmetic, BIN, DEC, HEX, and OCT; updated the alpha keyboards; added flags in the memory table; included indirect addressing for comparisons; added a paragraph about the display; updated the table of indicators; corrected errors.
1.7	9.9.09	Added P.FCN and STAT catalogues, 4 more conversions, 3 more flags, Greek character access, CLFLAG, DECOMP, DENANY, DENFAC, DENFIX, Iβ, IΓ, αDATE, αRL, αRR, αSL, αSR, αTIME, 12h, 24h, fraction mode limits, normal distribution and its inverse for arbitrary μ and σ, and Boolean operations working within FLOAT; deleted αROT, αSHIFT, the timer, and forced radians after inverse hyperbolics; renamed WINV to W^{-1}, and beta and gamma commands to Greek; added tables of catalogue contents; modified label addressing; relabeled PRGM to P/R and PAUSE to PSE; swapped SHOW and PSE as well as Δ% and % on the keyboard; relabeled Q; corrected CEIL and FLOOR; updated X.FCN and alpha commands; updated the virtual alpha keyboard.
1.8	29.10.09	Added R-CLR, R-COPY, R-SORT, R-SWAP, RCLM, STOM, alpha catalogues, 1 more constant and some more conversions, a table of error messages, as well as the binomial, Poisson, geometric, Weibull and exponential distributions and their inverses; renamed some commands; put √‾ instead of π on hotkey D.
1.9	14.12.09	Added two complex comparisons; swapped and changed labels in the top three rows of keys, dropped CLST; completed function descriptions in the index.
1.10	19.1.10	Added IMPFRC, PROFRC, ^{C}ENTER, αBEG, αEND, and an addressing table for items in catalogues; updated temporary alpha mode, display and indicators, RCLM and STOM, alpha-commands and the message table; renamed the exponential distribution; wrote the introduction.
1.11	21.9.10	Changed keyboard layout to bring Π and Σ to the front, relabeled binary log, swapped the locations of π, CLPR, and STATUS, as well as SF and FS?; created a menu TEST for the comparisons removed and the other programmable tests from P.FCN; added %MG, %+MG, %MRR, RESET, SSIZE4, SSIZE8, SSIZE?, ^{C}DROP, ^{C}FILL, ^{C}R↓, ^{C}R↑, registers J and K, a table of contents and tables for stack mechanics and addressing in complex operations; updated memory and real number addressing tables, DECOMP, αOFF, αON, Π, and Σ; renamed ROUNDI, WSIZE?, β(x,y), Γ(x) and the constant p_0 ; deleted DROPY (use x⇔y, DROP instead), αAPP, αBEG, αEND, and the "too long error" message; deleted Josephson and von Klitzing constants (they are just the inverses of other constants included in CONST already); brought more symbols on the alpha keyboard.
1.12	22.12.10	Modified keyboard layout; added the catalogues MODE and PROB; changed mode word, catalogue contents and handling (XEQ instead of ENTER), as well as some non-programmable info commands; expanded IMPFRC and PROFRC; added a paragraph about the fonts provided and explained alpha catalogues in detail; added PRIME? and some conversions; deleted FRACT, OFF and ON.

New (and existing) functions and their sizes, for information only:

Jacobi elliptic functions SN, CN & DN	1780 bytes	Re, Cx
Bessel functions of first and second kinds	4470 bytes	Re, Cx

Jn & In: real and complex (argument and order).
Yn & Kn: real and complex (argument and order).

Remember $J_n(x) = \sum_{r=0}^{\infty} \frac{(-1)^r \cdot \left(x/2\right)^{2r+n}}{r! \cdot \Gamma(n+r+1)}$

Digamma function (PSI)	1384 bytes	Re, Cx

needed for Bessel functions of second kind of integer order.

Zeta	2012 bytes	Re, Cx
~~cube / cube root~~	~~576 bytes~~	~~IN, Re, Cx~~
x!!	288 bytes	Re, Cx
~~EASTER~~	~~384 bytes~~	
Fused multiply/add	96 bytes	IN, Re

The real version can be replaced by complex multiply.
x+y*z can be done via (y, x) * (z, -1) at a pinch.

AGM	528 bytes	Re

limit of arithmetic geometric mean.

PRIME?	**576 bytes**	**IN, Re**

Conditional test for X being prime.
Exact for x < 66049, Miller-Rabin with 40 iterations otherwise.
P(claiming a composite is prime) = 2^-80 (approximately)
This routine also includes overflow resistant code for (a * b) mod c and (a ^ b) mod c which could also be exposed if required.

None of these are counting the catalogue and function table overheads.
Two bytes for a catalogue entry (one for each catalogue it is in) and 12-20 bytes for a function table entry (but only one of these), i.e. not terribly significant.

These are all moderately useful functions.

Paul's preferences for these would be to include the first five and PRIME?