PRELIMINARY
WP 31S OWNER'S MANUAL

This manual documents *WP 31S*, a free software replacing the original firmware of an *HP-20b* or *HP-30b* business calculator. It is an offspring of the *WP 34S* project. You can redistribute *WP 31S* and / or modify it under the terms of the GNU General Public License as published by the Free Software Foundation, either version 3 of the License, or (at your option) any later version.

WP 31S is published and distributed in the hope that it will be useful, but without any warranty; without even the implied warranty of merchantability or fitness for a particular purpose. For more details, see the GNU General Public License at *http://www.gnu.org/licenses/* .

This manual documents build 3626. We reserve the right to modify *WP 31S* at any time; its principles will stay constant, however. Stay informed by watching *http://sourceforge.net/p/wp34s/code/* .

The diagrams on p. 61 are based on material published in *Wikipedia*.

Internet addresses are specified as found and verified at 2014-4-10. Please note such addresses may change without notice at any time.

WP 31S would not have been created without our love for *Classics, Woodstocks, Stings, Spices, Nuts, Voyagers*, and *Pioneers*. Thus we want to quote what was printed in *Hewlett-Packard* pocket calculator manuals until 1980, so it will not fade:

> *"The success and prosperity of our company will be assured only if we offer our customers superior products that fill real needs and provide lasting value, and that are supported by a wide variety of useful services, both before and after sales."*
>
> *Statement of Corporate Objectives.*
> *Hewlett-Packard*

Just in Case You Do Not Have Your Own *WP 31S* Calculator Yet …

WP 31S runs on an *HP-20b Business Consultant* or an *HP-30b Business Professional*. Both are financial pocket calculators made for *Hewlett-Packard* and available for less than 40 US$ today. So get hold of one of them – preferably an *HP-30b* since it features significantly better keyboard mechanics. Then turn to *Appendix A* of this manual for instructions how to convert it into a *WP 31S*.

For the following, we assume the conversion is done and you hold your own *WP 31S* in your hands.

TABLE OF CONTENTS

WELCOME!

Dear user, we congratulate you on your *WP 31S*. It will serve you well, being designed based on the experience we gained with the *WP 34S RPN Scientific* since 2011. It uses the algorithms we elaborated for that high end calculator, now refined and packed in a lean and clean design for your number crunching *WP 31S*, laid over of the mechanics and hardware of an *HP-20b Business Consultant* or an *HP-30b Business Professional*. So you benefit from their excellent processor speed. With an *HP-30b* you also feel the famous rotate-and-click keys appreciated in vintage *Hewlett-Packard* pocket calculators since 1972.

Firmware and user interface of your *WP 31S* were thoroughly thought through, discussed, designed, written, and tested by us. We did that to create **a new straight and compact technical problem solver** that fits comfortably in your shirt pocket and uses *RPN*[1] – a serious scientific instrument. It readily offers you:

- A complete set of mathematical functions for solving engineering and scientific problems.
- A full-fledged UNDO (for the first time ever on an *RPN* pocket calculator).
- Statistical operations, including curve fitting and forecasting.
- Probability distributions like *Gaussian*, *Fisher's F*, *Student's t*, *chi-square*, *Poisson*, *binomial*, *Weibull*, and more*.*
- More than 50 fundamental physical constants as accurate as they are used today by national standards institutes such as *NIST* or *PTB*, plus a selection of important constants from mathematics, astronomy, and surveying.
- 86 unit conversions, mainly from old *British Imperial* to universal *SI* units and vice versa.
- Battery-fail-safe on-board backup memory.

[1] *RPN* stands for *reverse Polish notation*, a very effective and coherent method for the most efficient solutions to complicated problems. See *Section 1*.

Furthermore, your *WP 31S* provides ample space:

- a choice of 4 or 8 *stack* levels for your calculations,
- up to 17 general purpose registers for permanently storing your data, and
- 14 dedicated statistics registers for whatever you want to accumulate.

The *WP 31S* is a spin-off of the *WP 34S* firmware: it was created by Sanjeev Visvanatha (Canada) and Jonathan Cameron (USA) supported by the *WP 34S* team – Paul Dale (Australia), Marcus von Cube and me (both from Germany). Like its antecessor, also your *WP 31S* is a hobbyist's project still. It was introduced and discussed on the forum of the online *Museum of Hewlett-Packard Calculators* (http://www.hpmuseum.org/). We thank all the international forum members who contributed their ideas and lent their support throughout this project. We greatly appreciate your help! On said forum, you will find all the news about *WP 31S* in the future as well.

All aspects of *WP 31S* were carefully checked, so we have confidence it is free of severe bugs. This cannot be guaranteed, however, so we promise to continue improving *WP 31S* whenever necessary. If you discover any strange results, please report them to us. If they turn out to be caused by internal errors, we will provide you with an update as soon as it is available. Just as we did with the *WP 34S*, we will maintain short response times.

Enjoy!

Walter Bonin

Print Conventions and Common Abbreviations

- Throughout this manual, standard font is Arial. Emphasis is added by underlining or **bold** printing. *Specific terms, titles, trademarks, names* or *abbreviations* are printed in italics, *hyperlinks* in blue underlined italics. The latter cannot beam you to its target in a printed copy for obvious reasons, thus either a page number or the full external address is given or the link points you to the *Table of Contents*.
- Bold italic Arial letters such as ***n*** are used for variables; bold normal letters for constant **sample** values (e.g. labels, numbers, or characters). Calculator COMMANDS and CATALOGS are generally called by their names, printed in capitals in running text.
- This KEY font is taken for explicit references to calculator keys. Alphanumeric or numeric output (such as Hello! and 1234) are quoted using the respective calculator fonts.
- Courier is employed for `file names` and for describing numeric formats.
- Times New Roman bold capitals are used for register **ADDRESSES**, lower case bold italics for register ***contents***. For example, the value ***y*** lives in register **Y** and ***r5*** in **R5**. Overall *stack* contents are generally quoted in the order [***x***, ***y***, ***z***, …] .
- Lower case normal Times New Roman italics are for *units*.
- We will use decimal points in most parts of this manual (but you may set your *WP 31S* to decimal commas as well, of course). Although that point is definitely less visible than a comma, 'comma people' seem to be more tolerant against decimal points than vice versa (based on the number of complaints read).

All this holds unless stated otherwise locally.

The following abbreviations are used throughout this manual:

CDF = _**c**umulated **d**istribution **f**unction_ (…).

FM = _**f**lash **m**emory_ (a special kind of *RAM*).

HP = _**H**ewlett-**P**ackard._

IOP = _**I**ndex of **Op**erations_ (…).

LCD = _**l**iquid **c**rystal **d**isplay_.

PDF = _**p**robability **d**ensity **f**unction_ (…).

PMF = _**p**robability **m**ass **f**unction_ (…).

QF = _**q**uantile **f**unction_ (…).

RAM = _**r**andom **a**ccess **m**emory_, allowing read and write operations.

RPN = _**r**everse **P**olish **n**otation_ (see p. 17).

SI = _**s**ystème **i**nternational d'unités_, a coherent system of units of measurement agreed on internationally and adopted by almost the entire world – only Liberia, Myanmar, and the USA are missing still. We do not know what they are afraid of, and they seem not being aware of what they miss.

USB = _**u**niversal **s**erial **b**us_, a specific interface.

Some more abbreviations may be used (and explained) locally.

…

GETTING STARTED

At its heart, your *WP 31S* is a powerful problem-solving tool at your fingertips. Using the calculator's *RPN* logic system, you can work through complex problems while seeing intermediate answers every step of the way. The advantages of *RPN* become particularly apparent when dealing with exploratory type problems where intermediate answers are an important part of the problem solving process.

If you know how to deal with a good old *HP RPN* scientific calculator, you can start using your *WP 31S* right away. Browse this manual to get information on some basic design concepts that put your *WP 31S* ahead of many previous scientific pocket calculators. Continue using this manual for reference.

> On the other hand, if this is your first *RPN* calculator, we recommend you also read sections 1 to 2 of this manual thoroughly.
>
> *WP 31S* stands in the tradition of a long line of vintage *HP* calculators from 1968 on. *HP* launched the world's first scientific pocket calculator in 1972 and were famous for their quality calculators as long as they made them. You will find almost complete information about those devices, including the respective manuals, on *DVDs* distributed by the *Museum of HP Calculators* (see http://www.hpmuseum.org/cd/cddesc.htm) at low cost.

Almost all commands work on your *WP 31S* as they do on the *WP 34S* mentioned above. So knowing the *WP 34S* is a significant advantage. Nevertheless, this manual provides all the necessary information about the features of your *WP 31S*, including some equations and technical explanations. This manual is not intended, however, to replace textbooks on mathematics, statistics, physics, nor engineering, nor is it a hypothetical '*Beginner's Guide to RPN Calculating*'.

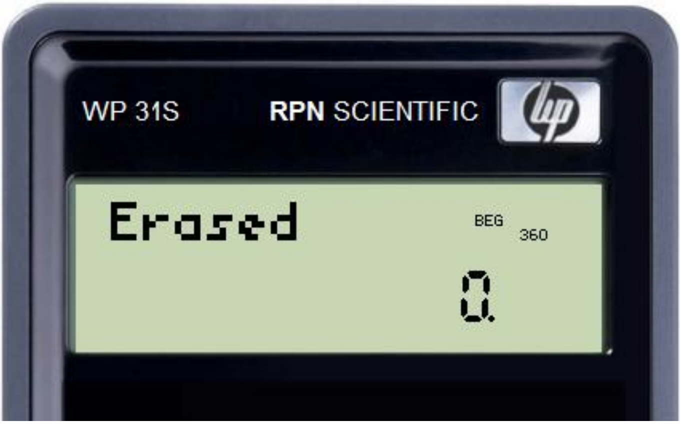
The following text starts with presenting the keyboard of your *WP 31S*, so you learn where you will find what you are looking for. It continues demonstrating basic calculation methods, the memory of your *WP 31S* and addressing objects therein, the display and indicators which give you feedback about what is going on. Then a major part of this manual is taken by an index of available operations, what they do, and how to call them. This manual closes with appendices covering special topics, e.g. the error messages your *WP 31S* will return if abnormal conditions prevent it from executing your command as expected. There you will also find instructions for keeping your *WP 31S* up-to-date when new firmware revisions will be released.

Before diving into it, here is something we ask you to remember:

> Your *WP 31S* is designed to help you solving problems. But it is just a tool – it cannot think for you nor can it check the sensibility of a problem you apply it to. Do not blame us nor your *WP 31S* for errors you may make. Gather information, think before and while keying in and calculating, and check your results: these tasks will always remain your responsibilities.

Problem Solving, Part 1: First Steps

Start exploring your *WP 31S* by turning it on: Press its bottom left key – notice that ON is printed below that key. Doing this the very first time, you will get a display like this:

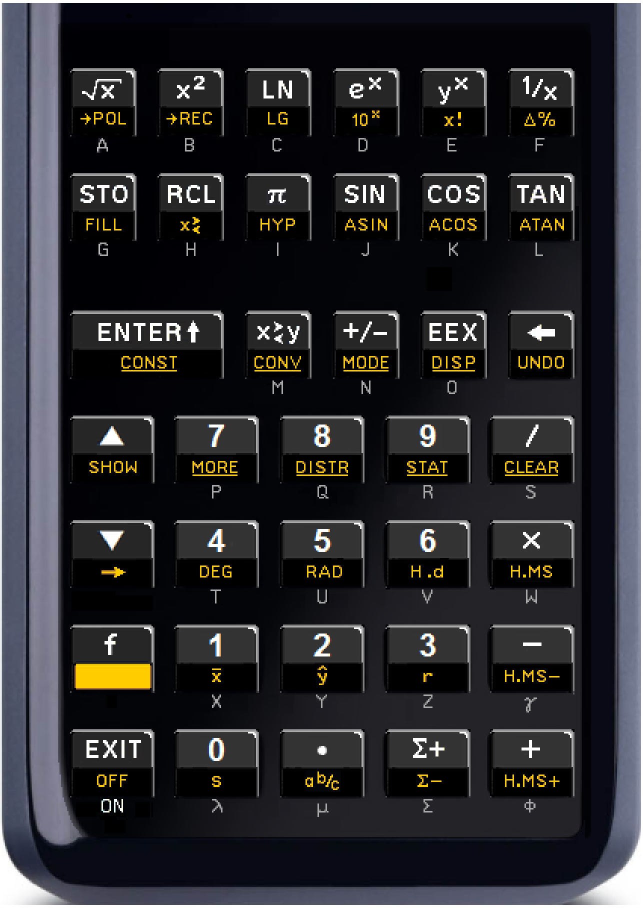


To turn your *WP 31S* off again, press f (notice a little f showing up left in display), then EXIT (which has OFF printed on its lower slanted face). Since your *WP 31S* features *Continuous Memory*, turning it off does not affect the information it contains. To conserve battery power, your *WP 31S* will shut down automatically about five minutes after you stop using it – turn it on again and you can resume your work right where you left off.

To adjust display contrast, hold down the key ON and press + or − repeatedly until it suits your needs.

All that works as on preceding *RPN* pocket calculators (like *HP-42S* or *WP 34S*). Your new *WP 31S*, however, looks cleaner than a *WP 34S.* This is due to your *WP 31S* featuring just one *prefix* key.

Thus you get up to two functions per key. Looking at an arbitrary key, white print is for its *primary* function. For additional functions,



golden labels are put on the slanted fronts of 36 keys, grey characters are printed below 29 of the keys.

For accessing a white label, just press the corresponding key (thus it is called its *primary* function). For a golden label, press $\boxed{f}$ first, then the corresponding key.

Take the key[2] $\boxed{1}$, for **example:** Pressing ...

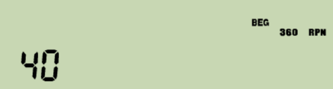
- $\boxed{1}$ alone will enter the digit **1**,
- $\boxed{f}$ + $\boxed{1}$ will call $\boxed{\bar{x}}$ computing the *mean value* (of previously entered statistical data).
- The grey letter X will become relevant in *addressing* (see further below).

[2] For better readability on paper, we refer to keyboard labels using dark print on white from here on, like e.g. $\boxed{2}$ or $\boxed{\bar{x}}$, omitting the *prefix* $\boxed{f}$ for the latter since the printed color makes it redundant.

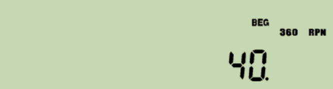
Time for a little problem solving **example:**

Let us assume you want to fence a little rectangular patch of land, 40 *yards* long and 30 *yards* wide.[3] You have already set the first corner post (A), and also the second (B) in a distance of 40 *yards* from A. Where do you set the third and fourth posts (C and D) to be sure that the fence will form a proper rectangle? Simply key in

4 0

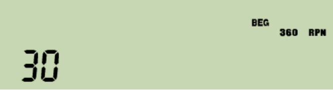

BEG 360 RPN
40

ENTER↑

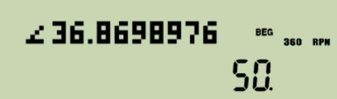

BEG 360 RPN
40.

This ENTER↑ separates the two numbers in input here.[4]

3 0


BEG 360 RPN
30

→POL


∠36.8698976 BEG 360 RPN
50.

Note that →POL is reached by pressing f + √x top left on the keyboard.[5]

Now all you need is a friend and 80 *yards* of rope. Ask your friend to hold both ends of the rope for you, take the loose loop and walk away as far as you can – when the loop is stretched,

[3] Most of you are fond of *SI*. Despite this fact, we use (old *British*) *Imperial* units here so our US-American readers can follow. But the example will work with *meters* as well.

[4] Note the number 40 is adjusted to the right and a radix mark is added. This indicates input being closed for this number. Further effects of ENTER↑ will be discussed below.

[5] The characters displayed in the top row here are not relevant for solving this problem; they will be explained later.

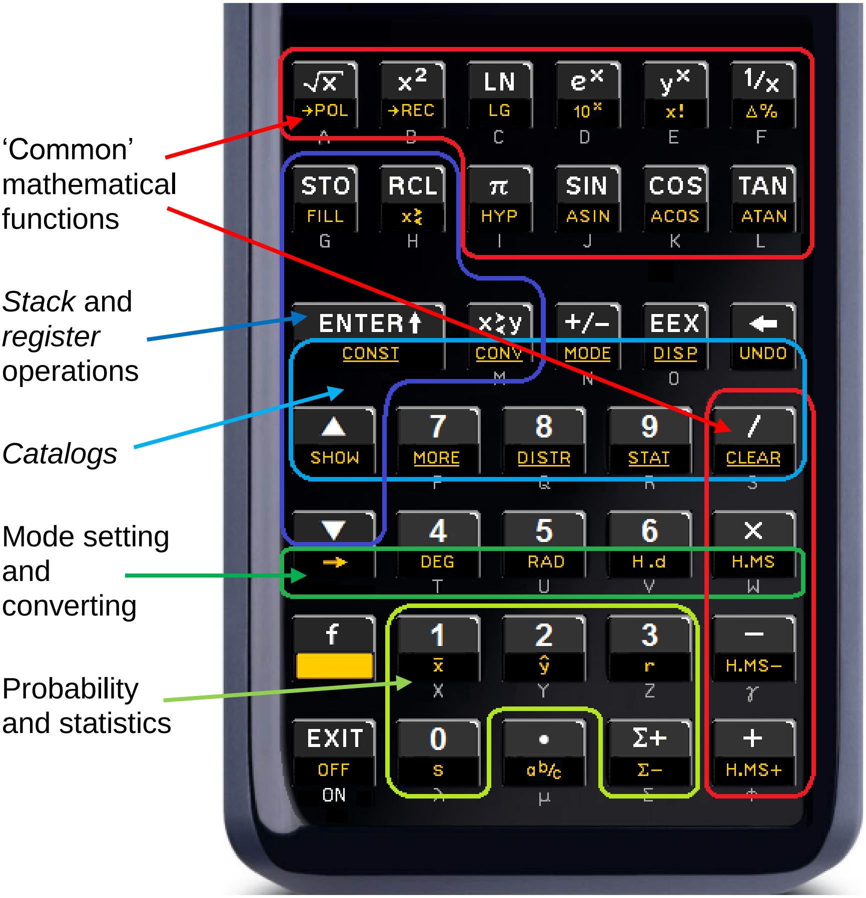
mark that position on the rope and return to your friend. Ask him or her to hold this point of the rope as well, fetch the two loose loops and walk again as far as possible – when the loops are stretched, mark both positions on the rope. Return again; hand over the two new points and walk once more, now with four loose loops. After marking as before, your rope has got marks every 10 *yards* now.

Nail its one end on post A now and its other end on B, fetch the loose loop and walk 3 marks away. When both sections of the rope are tightly stretched, stop and place post C there. You may set post D the same way.

This method works for arbitrary rectangles, whatever other distances may apply (you may need a tapeline in the general case). As soon as you press →POL, your *WP 31S* does the necessary calculation of the length of the diagonal automatically for you. You just provide the land, posts, rope, hammer and nails. And it will be up to you to set the posts!

How the Keyboard is Organized

You will have recognized the labels on your *WP 31S* are grouped according to their purposes. There are four larger sets:



Getting used to these sets, you will very soon find functions on your *WP 31S* easily. Functions and their inverses – if inverses exist – are placed next to each other.

Before proceeding to the next chapter, let us return to our little introductory land fencing problem solving example with two general remarks:

1. There is absolutely no need to enter units in your calculations. Just stay with a consistent set of units and you will get meaningful results within this set.[6] If you need to convert inputs into units being part of such a set or you want results in special units, however, CONV will help (…).
2. Although we entered integer numbers only for both sides of our little patch of land, your *WP 31S* calculated the diagonal using real (floating point) numbers. It lets you enter decimal *yards* as well. Alternatively you may enter fractions such as e.g. 6 ¼ if you need them. We will introduce fraction mode further below.

Now, let us show you more general things first: how to enter numbers and commands, and how to deal with them in typical calculations.

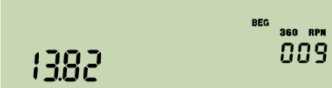
How to Enter Numbers

This is as easy as typing: e.g. for entering 12.34 simply press 1 2 • 3 4. You may put in up to 12 digits and a radix mark for the mantissa, optionally multiplied by an up-to-three-digit power of ten. Any digit mistyped may be deleted immediately by ⬅ and entered correctly thereafter.

For negative numbers such as –5.6 , enter 5 +/– • 6 or 5 • +/– 6 or 5 • 6 +/– ; pressing +/– changes the sign of the number being entered.

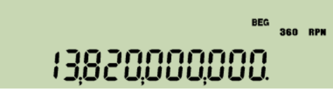
[6] The big advantage of *SI* is that it is the largest consistent set of units available.

For a really big figure such as the age of our universe as we know it, enter 1 3 • 8 2 EEX 9 (years) which is echoed



1382 009

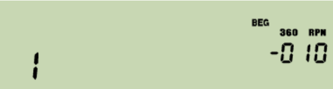
When you complete input by pressing ENTER↑ [7] you will see the equivalent:



13,820,000,000.

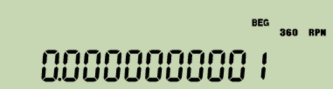
in the startup default display format.

Really tiny numbers such as the typical diameter of an atom (i.e. 0.000 000 000 1 *m* – with ten zeroes leading the 1) are entered in full analogy: EEX +/- 1 0 [8]



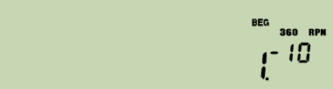
1 -010

will be displayed like



0.0000000001

as it should be after completing input. It may be, however, shown more compact as



1.-10

with other settings.

[7] Before completing it, input is generally just echoed in the lowest numeric line. Input is closed and released for interpretation by a command – e.g. by ENTER↑ as shown here.

[8] Note you do not have to enter 1 EEX +/- 1 0 here.: if there is no open numeric input heading EEX, 1 is assumed for the mantissa. And a +/- entered after EEX will always change the sign of the exponent – if you want to change the sign of the mantissa instead, do it before pressing EEX or after closing input.

How to Access and Execute Commands

Just enter the sequence of keystrokes required to access the command you wish to execute. Pending input will be echoed top left of the screen until the command is completed. The active *prefix* is displayed by f for visual feedback, if applicable.

For many commands, that will be the only echo you will see during input since the next keystroke will terminate entry, execute the command, and show the result (as with →POL above).

Some commands (like STO and RCL), however, require one or more trailing parameters and will thus stay displayed for longer. Examples will be shown at the end of this section.

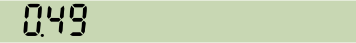
Problem Solving, Part 2: Elementary *Stack* Mechanics

Almost all of the commands of your *WP 31S* are mathematical operations or functions taking and returning real numbers such as 1 or -2.34 or 3.141 592 653 59 or 5.6×10^{-7}. Note that integer numbers like 3, 10, or –1 are just a subset of real numbers.

One-number (*monadic*) functions: Many real number functions provided operate on <u>one</u> such number only.

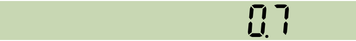
Example:

Enter . 4 9



0.49

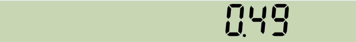
and press √x for calculating the square root. You will get



0.7

in the startup default display format.

Pressing x^2 returns



0.49 .

Generally speaking, *monadic* functions replace the value displayed in the lowest numeric line (this value is called ***x***) by the numeric function result **f(*x*)** – e.g. by **f(*x*)** $= x^2$ in the last example. The vast majority of electronic calculators works this way, so this is no real surprise.

Besides $\sqrt{x}$ and x^2, there are many more *monadic* functions provided on the keyboard of your *WP 31S*:

- LN (for $ln(x)=log_e(x)$) and e^x,
- LG (for $lg(x)=log_{10}(x)$) and 10^x,
- the reciprocal 1/x,
- the factorial x!,
- the trigonometric functions and their inverses:
  - SIN for $sin(x)$ and ASIN for $arcsin(x)$,
  - COS for $cos(x)$ and ACOS for $arccos(x)$,
  - TAN for $tan(x)$ and ATAN for $arctan(x)$,
- the hyperbolic functions and their inverses:
  - HYP SIN for $sinh(x)$ and HYP ASIN for $arsinh(x)$,
  - HYP COS for $cosh(x)$ and HYP ACOS for $arcosh(x)$,
  - HYP TAN for $tanh(x)$ and HYP ATAN for $artanh(x)$,
- and the angular conversions → DEG and → RAD.

The *Index of Operations (IOP)* below may contain further information about each of these functions.

Two-number (*dyadic*) functions: Some of the most popular mathematical functions, however, operate on two numbers instead. Think of + and – , for instance.

Example: Assume having an account of 1,234 US$ and taking 56.7 US$ away from it. What will remain? One easy way to solve such a task works as follows:

On a piece of paper		On your *WP 31S*		
Write down the first number:	1234	Key in the first number:	1 2 3 4	1234
Start a new line.		Close 1st input:	ENTER↑	
Write down the second number; draw a line below:	56.7	Key in the 2nd number:	5 6 . 7	56.7
Subtract:	1177.3	Subtract:	−	1177.3

This is the essence of *RPN*:

Provide the necessary operands, and then execute the requested operation.

And a major advantage of *RPN* compared to other calculator operating systems is that it sticks to this basic logic – always.[9]

As the paper holds your operands before you start calculating manually, a place holding your operands on your *WP 31S* is required, too. The *stack* does that.

[9] Some people claim this being true for *RPL* only. *RPL* is a language developed from *RPN* in the 1980's. Maybe those people are even right. In my opinion, however, *RPL* strains the underlying *postfix* principle beyond the pain barrier, exceeding the limit where it becomes annoying for human brains. Not for everybody, of course, but also for many scientists and engineers. Thus we stick to *RPN* on our pocket calculators.

Think of it like a pile of registers.[10] They are traditionally named **X, Y, Z,** and **T,** optionally followed by **A, B, C,** and **D** on your *WP 31S*.

New input is always entered in the bottom register **X** and is displayed in the bottom line. Also ***y*** may be shown on the screen on demand.

ENTER↑ separates two input numbers following each other immediately by closing the first number ***x*** and copying it into **Y**, so **X** can take the second number then without losing information (compare p. 12).[11]

	***Stack* register**	**content**
	D	***d***
	C	***c***
	B	***b***
	A	***a***
	T	***t***
	Z	***z***
	Y	***y***
Display	**X**	***x***

After having keyed in the second number (the new ***x***) in the account example above, pressing − subtracts this ***x*** from the first number (now in **Y**) and puts the result **f(*x, y*)** = ***y*** – ***x*** in **X**. This procedure applies to most *dyadic* functions featured:

Put both operands on the *stack*
then execute the operation f(*x, y*)
and the result will be displayed in X.

…

There is also a **three-number (*triadic*) real function** featured by your *WP 31S*: %MRR replaces ***x*** by **f(*x, y, z*)**; then ***t*** drops into **Y** and so on, and the content of the top *stack* level is repeated twice.

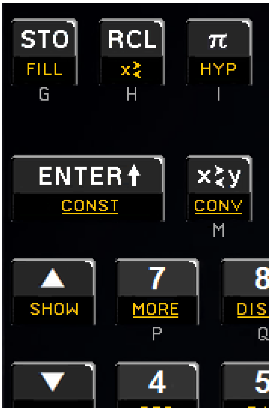
[10] Learn more about all the registers your *WP 31S* provides in next chapter.

[11] This is the classic way ENTER↑ also worked from the *HP-35* of 1972 until the *HP-42S* ceased in 1995. It is often said ENTER↑ 'pushes ***x*** on the *stack*' (although it pushes ***x*** under the *stack* in our picture). The higher *stack* contents are lifted out of the way before: in a 4-level *stack*, ***z*** is put into **T** and ***y*** into **Z** before ***x*** is stored in **Y**. See pp. 26ff for details and pictures.

Some real functions (e.g. DECOMP) operate on one number but return two. Other operations (such as RCL or SUM) do not consume any *stack* input at all but just return one or two numbers. Then these extra numbers will be pushed on the *stack*, taking one level each (see p. 23).

Looking Closer at the *Stack*

To understand the genius of *RPN*, we will look a bit closer at the functions operating on the *stack*. In addition to one-, two-, and three-number functions explained above, there are some dedicated *stack* commands:



The traditional *stack* and register control operations ENTER↑, x≷y, R↓ [12], STO, RCL, LASTx, R↑, x≷, and SHOW have been known for decades and are almost all found within this small area of the keyboard here; six new commands were added to your *WP 31S*: FILL, SSIZE4, SSIZE8, SSIZE?, YDOFF, and YDON. All these but FILL are found in MORE (see p. 48).

Your *WP 31S* is the second calculator model worldwide offering you a choice of four or eight *stack* levels (see the commands SSIZE4 and SSIZE8). [13] Four levels are for full compatibility with four decades of *HP* calculators, eight levels are for worry-free calculations of even the most complex real world formulas.

Thus, the fate of *stack* contents depends on the particular operation executed as well as on the *stack* size chosen.

[12] This traditional command label stands for **r**olling the *stack* one level down. There is another command rolling the *stack* up. See the next page to find out why they carry those names.

[13] The world's first eight-*stack*-level calculator was the *WP 34S* in 2011.

Operations in the 4-level *stack* work as known from vintage *HP RPN* calculators for decades. In the optional larger *stack* of your *WP 31S* everything works in analogy – just with twice the levels available for intermediate results.

Look below to see what ENTER↑, FILL, x≷y, R↓, and R↑ do in detail on the *stacks* of either size (startup default display format is assumed for these examples). Turn to next page for recalling, *monadic*, and *dyadic* functions. See the IOP for detailed information about the other commands mentioned above.

			Stack contents after executing …				
	Stack level	Assumed *stack* contents at start:	ENTER↑	FILL	x≷y	R↓ = ◀	R↑ = ▶
With **4** *stack* levels	T	*t* = 4.	3.	1.1	4.	1.1	3.
	Z	*z* = 3.	2.	1.1	3.	4.	2.
	Y	*y* = 2.	1.1	1.1	1.1	3.	1.1
	X	*x* = **1.1**	**1.1**	**1.1**	**2.**	**2.**	**4.**
With **8** *stack* levels	D	*d* = 8.	7.	1.1	8.	1.1	7.
	C	*c* = 7.	6.	1.1	7.	8.	6.
	B	*b* = 6.	5.	1.1	6.	7.	5.
	A	*a* = 5.	4.	1.1	5.	6.	4.
	T	*t* = 4.	3.	1.1	4.	5.	3.
	Z	*z* = 3.	2.	1.1	3.	4.	2.
	Y	*y* = 2.	1.1	1.1	1.1	3.	1.1
	X	*x* = **1.1**	**1.1**	**1.1**	**2.**	**2.**	**8.**

Stack level	Assumed *stack* contents at start:	*Stack* contents after executing …			
		RCL L	s [14]	x^2 [15]	+ [16]
T	***t*** = 4.	3.	2.	4.	4.
Z	***z*** = 3.	2.	1.1	3.	4.
Y	***y*** = 2.	1.1	s_y	2.	3.
X	***x* = 1.1**	***last x***	s_x	**1.21**	**3.1**

D	***d*** = 8.	7.	6.	8.	8.
C	***c*** = 7.	6.	5.	7.	8.
B	***b*** = 6.	5.	4.	6.	7.
A	***a*** = 5.	4.	3.	5.	6.
T	***t*** = 4.	3.	2.	4.	5.
Z	***z*** = 3.	2.	1.1	3.	4.
Y	***y*** = 2.	1.1	s_y	2.	3.
X	***x* = 1.1**	***last x***	s_x	**1.21**	**3.1**

RCL L represents the vintage command LAST***x*** (…).

Using the *stack* this way, *RPN* completely eliminates the need for any parenthesis keys!

[14] This stands for an arbitrary function returning two values.
[15] This represents an arbitrary monadic function.
[16] This represents an arbitrary dyadic function.

See this **example**, showing a slightly more complicated formula and the keystrokes used for solving it step by step (where the colors represent the stack levels involved):

$$\sqrt{\frac{1+\left|\left(\frac{30}{7}-7.6\times 0.8\right)^{4}-\left(\sqrt{5.1}-\frac{6}{5}\right)^{2}\right|^{0.3}}{\left\{sin\left[\pi\left(\frac{7}{4}-\frac{5}{6}\right)\right]+1.7(6.5+5.9)^{3/7}\right\}^{2}-3.5}}$$

RAD 7 ENTER↑ 4 / 5 ENTER↑ 6 / − π × SIN	$sin\left[\pi\left(\frac{7}{4}-\frac{5}{6}\right)\right]$
6.5 ENTER↑ 5.9 + 3 ENTER↑ 7 / y^x 1.7 ×	$1.7\times(\ldots)^{3/7}$
+ x^2 3.5 −	complete denominator
7.6 +/− ENTER↑ .8 × 30 ENTER↑ 7 / + 4 y^x	$(30/7-7.6\times 0.8)^4$
6 ENTER↑ 5 / 5.1 $\sqrt{x}$ − x^2	$\left(\sqrt{5.1}-6/5\right)^2$
− x^2 $\sqrt{x}$ [17] .3 y^x 1 +	complete numerator
x≷y / $\sqrt{x}$	complete result (0.35)

Even solving this formula requires only four *stack* levels. Note there are no *pending operations* – each operation is executed individually, one at a time, allowing **perfect control of each and every intermediate result.** This is another characteristic advantage of *RPN*.

[17] The sequence x^2 $\sqrt{x}$ calculates the absolute value of x. In real life, you see whether x is negative here and can turn it positive by +/− if necessary.

Calculating such a formula from the inside out is always a wise strategy. If you had started with its numerator straight ahead, you would have needed five levels for its complete solution instead. With eight levels as provided on your *WP 31S*, you will be on the safe side even with the most advanced equations you will meet in your life as scientist or engineer.

For decades, solving the following formula for the *Mach* number as a function of aircraft speed in *knots*[18] (here: 350) and altitude in *feet* (here: 25 500) was used for demonstrating the simplicity and coherence of RPN:

$$\sqrt{5\left(\left[\left\{\left(1+0.2\left[\frac{350}{661.5}\right]^{2}\right)^{3.5}-1\right\}\{1-6.875\times10^{-6}\times25500\}^{-5.2656}+1\right]^{0.286}-1\right)}$$

Start with the innermost parenthesis and solve the total expression like this (we will need only three *stack* levels here):

350 ENTER↑ **661.5** / x^2

.2 × **1** + **3.5** y^x

1 −

6.875 EEX +/– **6** ENTER↑ **25500** × +/–

1 + **5.2656** +/– y^x

×

1 + **.286** y^x

1 −

5 × $\sqrt{x}$, resulting in approximately 0.84, i.e. 84% of the speed of sound.

[18] The *knot* is an ancient *Imperial* unit surviving in aviation business. $1\ knot = 1\ {}^{nautical\ mile}/_{hour} = {}^{463}/_{900}\ {}^{m}/_{s} \approx 0.5144\ {}^{m}/_{s} \approx 1.85\ {}^{km}/_{h}$. And a *foot* is another unit taken from that heap of old units (see CONV). We quote the above formula without checking it.

At the bottom line, we recommend:

**Let your *WP 31S* care for the arithmetic
while you care for the mathematics!**

Error Recovery

Nobody is perfect – errors will happen. With your *WP 31S*, however, error recovery is as easy as pressing ⌫ or UNDO .

1. If you got an **error message** in response to your function call, just press ⌫ to erase that message and return to the state before that error happened. Now do it right!
2. If you have erroneously called a **wrong command**, just press UNDO to undo it. This command will restore the *stack* and all other registers exactly as they were before the last command was executed. Since your *WP 31S* features *RPN*, that is all you need for resuming your work as if said error did not happen at all.

Don't worry – be happy!

…

Reaching Commands That Are Not Printed on the Keys

Your *WP 31S* features over 340 different commands. 72 labels are printed on the keys. So how do you learn about the other commands? Where are they and how can you call them?

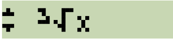
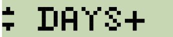
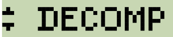
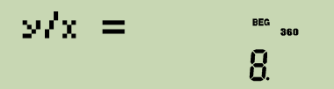
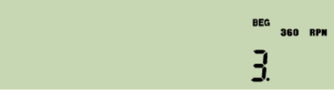
The answer to the first question is easy: read! The *IOP* contains everything.

The answer to the second question is less obvious, but easy as well: the 'hidden commands' are stored in *catalogs*. Remember all underlined labels on the keyboard point to such collections of commands. You will find them on the slanted fronts of eight keys.

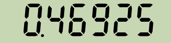
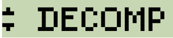
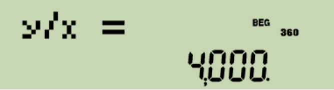
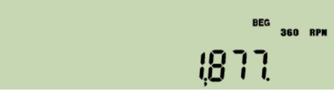
Example: Assume you want to learn about DECOMP – this command was mentioned on p. 21 but is not printed on any key. You return to the *Table of Contents*, look for the *Index of Operations*, jump directly to the letter **D** therein, and look up DECOMP to get the necessary information where this command lives and what it does. OK, enter **0.375** and check it – calling it is even easier than looking it up:

You have read DECOMP lives in MORE.

So just key in … and your *WP 31S* displays …

MORE	 ⇕ ³√x	being the first command stored in MORE.
D	 ⇕ DAYS+	for obvious reasons.
▼	 ⇕ DECOMP	voilà! Now execute it by pressing …
ENTER↑	 y/x = BEG 360 8.	
x≷y	 BEG 360 RPN 3.	since 0.375 = $^3/_8$.

Expected that. But what is this:

0.46875	 0.46925	?
MORE	 ⇕ DECOMP	oh, good, that function is memorized!
ENTER↑	 y/x = BEG 360 4,000.	
x≷y	 BEG 360 RPN 1,877.	meaning 0.469 25 = 1 877 / 4 000 .

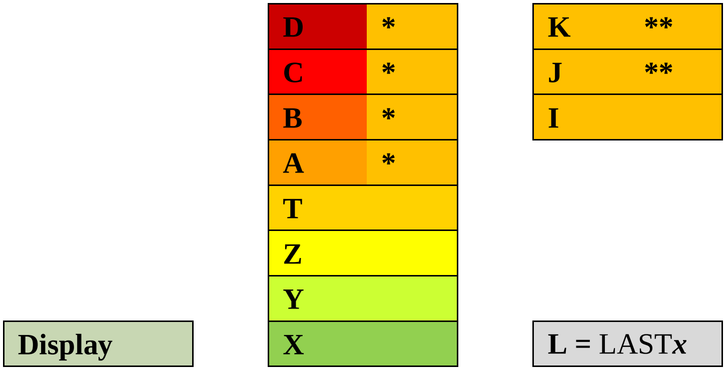
As ▼ browses forwards in MORE, ▲ will browse backwards. And EXIT will leave the *catalog* without executing anything.

That's all you need to know about calling commands and real calculations on the *stack*. There are, however, more places in your *WP 31S* where data may be stored and saved. Let us show them to you.

Addressing Objects in *RAM*

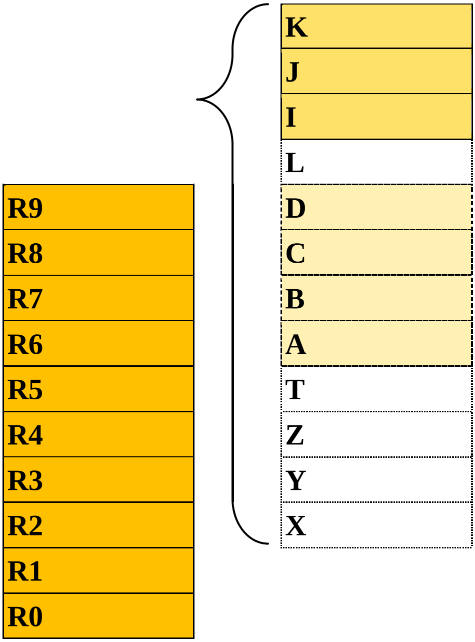
The picture below and the one overleaf show the address space of your *WP 31S*. For addressing those registers, see the table below.

Special registers and *stack*



Depending on *stack* size chosen, either **T** or **D** will be the top *stack* level. **A – D** will be allocated for the *stack* if required. **J** and **K** may carry parameters of statistical distributions. Unless required for the purposes just mentioned, **A, B, C, D, I, J,** and **K** are available as additional general purpose registers (see overleaf).

General purpose registers[19]

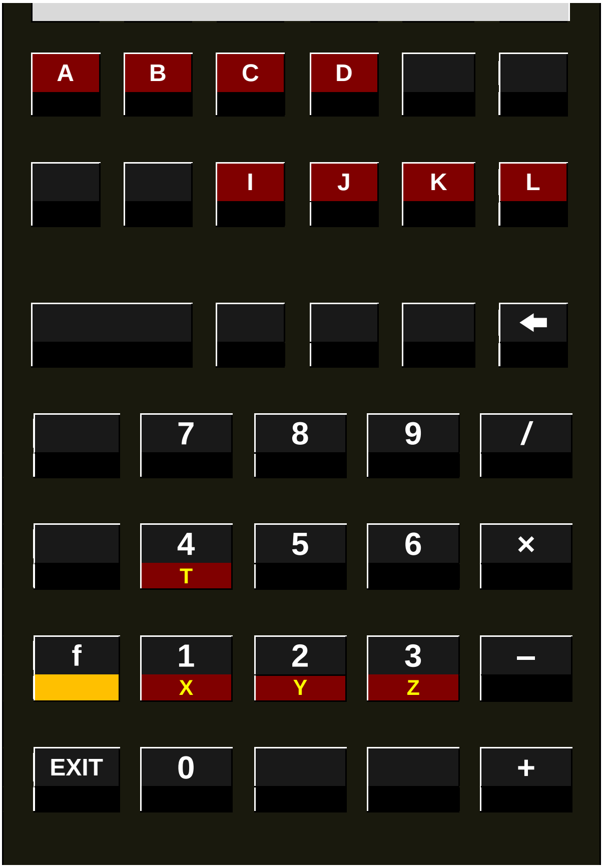


During input processing in memory addressing, e.g. while entering parameters for storing, recalling, swapping, copying, clearing, or comparing, you will not need all the labels presented on the keyboard. Just 28 keyboard labels plus the *prefix* will do instead.

The calculator mode supporting exactly these labels (and no more) is called *transient alpha mode* (α_T). As shown in examples on the next pages, it may be automatically set in addressing.

Entering α_T, the operational keyboard is temporarily reassigned as pictured overleaf. There, **dark red background** is used to highlight changed key functionality – e.g. the top left key will access register (or *stack* level) **A** in α_T directly.

[19] Your *WP 31S* features 14 more registers, dedicated to statistical sums. They are accessed differently as explained in the *IOP*, and are not part of this address space.



Note that you will not need f in α_T except for addressing the four lowest *stack* registers.

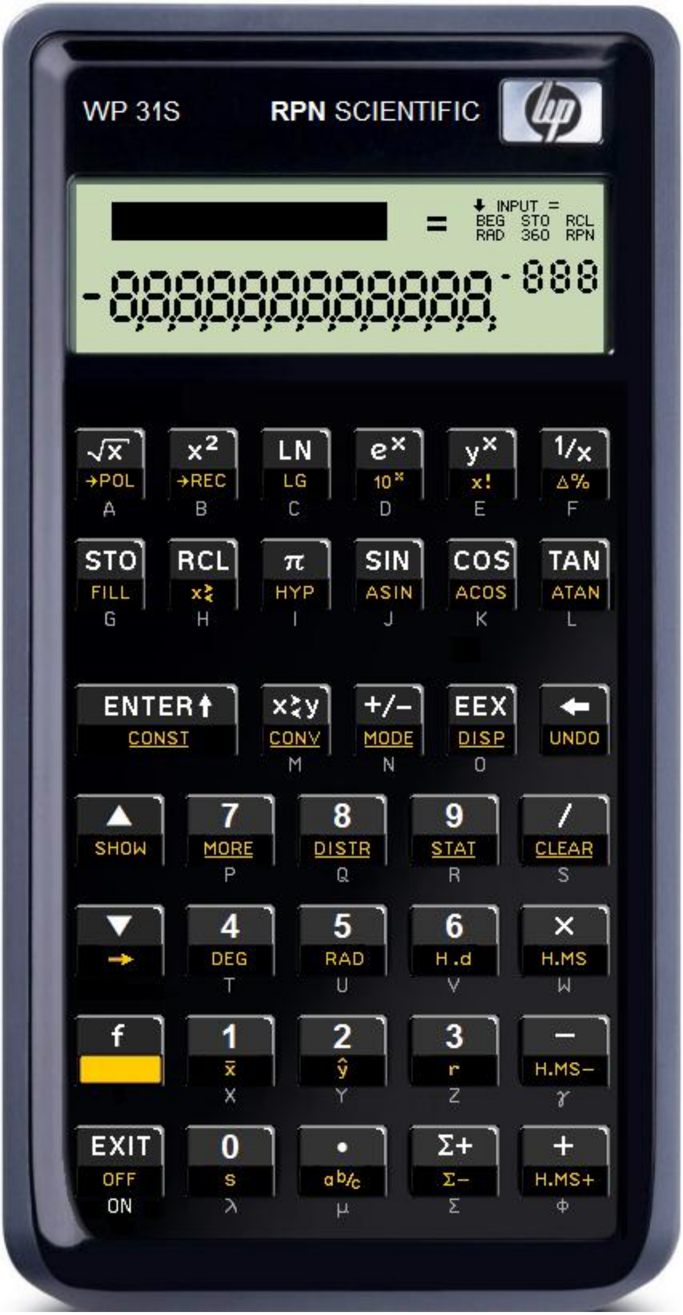
α_T will be terminated (automatically returning to the mode set before) as soon as sufficient characters are entered for the respective operation. You may delete pending input character by character using ← – or just abort the pending command by EXIT; the latter will leave α_T immediately.

Addressing Table

This table shows the stages of entering storage and recall commands to address registers. You will see that STO and RCL may be combined with the four arithmetic operators to perform calculations during storage or recall operations.

1	User input	RCL, STO, or x≷.	
	Echo	**OP** _ (with mode α_T set), e.g. x⇆_	
2	<u>Optional</u> user input for RCL or STO [20]	+, −, ×, or /	
	Echo	**OPo** _ (with α_T set), e.g. RCL+ _	
3	User input	*Stack level* or *lettered register* X, Y, Z, … , I, J, K	*Register* ***number*** 0, 1, 2, … , 9
	Echo	**OP *x*** e.g. STO− K	**OP *n*** e.g. x⇆ 4

[20] Entering such a step-2 operator twice will remove it, so e.g. RCL / / equals RCL.
Note this so-called store and recall arithmetic may operate on any register, also on the *stack* and even on **L**. See RCL and STO in the *IOP* for more about it. There are a few true adepts on this planet who have learned some very nice tricks making use of these particular arithmetic operations.


WP 31S
RPN SCIENTIFIC
INPUT =
BEG STO RCL
RAD 360 RPN
√x →POL A
x² →REC B
LN LG C
eˣ 10ˣ D
yˣ x! E
1/x Δ% F
STO FILL G
RCL x≷ H
π HYP I
SIN ASIN J
COS ACOS K
TAN ATAN L
ENTER↑ CONST
x≷y CONV M
+/− MODE N
EEX DISP O
← UNDO
▲ SHOW
7 MORE P
8 DISTR Q
9 STAT R
/ CLEAR S
▼ →
4 DEG T
5 RAD U
6 H.d V
× H.MS W
f
1 x̄ X
2 ŷ Y
3 r Z
− H.MS− γ
EXIT OFF ON
0 s λ
• a b/c μ
Σ+ Σ− Σ
+ H.MS+ Φ

DISPLAYING SOLUTIONS ON YOUR *WP 31S*

The *LCD* is your window to your *WP 31S* – there you see what is going on and what the results are. This display sports a total of 400 elements in 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 on the top right side are called *annunciators.*

The ten annunciators are for indicating modes. Not all of them are used by your *WP 31S*.

The numeric section in the lower part of the *LCD* is used for displaying numbers in different formats, status data, or message parts. See the examples below for more.

The dot matrix section above is used for passing additional information to the user.

Example: 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 and → are shown until they are resolved. If you pressed such a prefix erroneously, recovery is as easy as follows:

○ f f = → → = NOP

○ f → ⌫ = f → EXIT = NOP

In addressing registers, progress is recorded as explained on p. 31 in detail. You may edit such pending operations by ⌫ or cancel them by EXIT (…).

If two or more requests compete for display space, the priorities are:

1. error messages as listed in *Appendix C*,
2. special information as explained below,
3. the contents of **Y** if YDON is set,
4. information about the modes your *WP 31S* is running in.

Recognizing Calculator Modes

The **annunciators** or specific characters in the dot matrix indicate most modes and system states:

Indicator	Set by	Cleared by	Explanation, remarks
INPUT	entering a *catalog*	leaving it	Allows for entering letters for fast access to commands stored in a *catalog*. See pp. 46ff for more about browsing *catalogs*.
▬	battery low	battery voltage > 2.5 *V*	Low battery will reduce processor speed. Your *WP 31S* will shut off when voltage drops below 2.1*V*.
BEG	always on		
RAD	RAD	DEG	Angular mode (see p. 39)
360	DEG	RAD	
RPN	almost every command	a *temporary message*	See p. 36 for handling of *temporary messages* in general.

The indicators following below are all lit in the dot matrix exclusively:

Indicator	Set by	Cleared by	Explanation, remarks
f	f	see pp. 33f	Transient signal of pending prefix.[21]
f →	→		
÷	entering a *catalog*	leaving it	See pp. 46ff for more about *catalogs*.
m.dy	M.DY	any other date setting	See pp. 39ff for more about date modes. Note default D.MY is not indicated. No date mode will be shown as long as YDON is set.
y.md	Y.MD		
A number in the dot matrix	YDON	YDOFF	Permanent display of y may help keeping track of the stack. Note numbers will appear up there temporarily after →POL and →REC in any case.

Radix marks and separators are seen in the numeric display immediately. The numeric format of fraction mode is unambiguous as well; check the examples shown on pp. 42ff.

All keyboard input will be interpreted when the input is completed, according to the modes set at that time.

[21] Note that → automatically sets f for the next keystroke. There is no need for pressing f once more.

Common Commands Returning Specific Displays

Some common commands use the *LCD* for presenting ***temporary messages*** as defined here:

> Whenever anything different from the actual contents of **X** in current mode is displayed or any additional information is shown in the dot matrix, this extra information is considered a *temporary message*. This is further indicated by the annunciator **RPN** turned off as mentioned on p. 34.
>
> If such extra information is displayed outside of a *catalog*, it will vanish with the next keystroke. Pressing EXIT or ⬅ will just clear the *temporary message* and return to the normal display, any other key will be executed in addition.

Various commands will result in special information displays:

1. CLALL will ask you for confirmation before executing. Answer Y (by pressing 2) or N (by pressing +/-). Also EXIT or ⬅ will be read as N, all other input will be ignored.
2. CATREG shows register contents as *temporary messages* (see pp. 46f).
3. VERS generates a *temporary message* showing you version and build of the firmware running on your *WP 31S* (like the display on the title page of this manual).
4. Any error condition will result in a (*temporary*) error message displayed (see *App. B*). Deal with it as explained on p. 26.
5. Browsing an arbitrary *catalog* will display the commands contained therein as *temporary messages* (see pp. 46ff for more).

Displaying Numbers

Floating point modes cover the ‘usual’ numbers you calculate with: decimal real numbers, fractions, measured or counted values, angles, times and dates. Thus, DECM is the startup default mode of your *WP 31S*.

You can set display preferences according to your personal and your region's practices using the commands RDX**,** or RDX**.**, TSON or TSOFF, as well as D.MY, M.DY, or Y.MD.

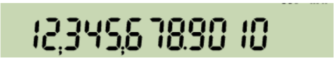
For **floating point decimal numbers**, startup default displays all digits as long as they fit the display width; it will switch to SCIentific (i.e. mantissa plus exponent) notation to avoid flowing over the display limits. This format is ALL 0. Besides ALL and SCI, there are two more numeric display formats, FIX and ENG. Their effects can be most easily demonstrated and distinguished using an **example:**

Format Input	ALL 00	FIX 4	SCI 3	ENG 3
107.123456	107.123456	107.1235	1.071 ^{2}	107.1 ^{0}
1/x **2** ×	1.86700473891 $^{-2}$	0.0187	1.867 $^{-2}$	18.67 $^{-3}$

Within FIX, the radix mark will always stay at the FIXed position defined. The radix mark floats in the other notations, where e.g. 1.071 ^{2} represents $1.0712 \cdot 10^{2}$, while 18.67 $^{-3}$ V means $18.67 \cdot 10^{-3}\ V = 18.67\ mV$. Within ENG, the exponent will always be a multiple of three corresponding to the unit prefixes in *SI* – thus it is called the ENGineer’s notation.

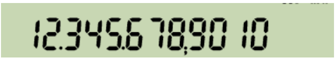
As soon as a numeric entry is completed, the mantissa will be displayed adjusted to the right, the exponent to the left. Within the mantissa, either points or commas may be selected as radix marks, and additional marks may be chosen to separate thousands.

Example: Assume the format is set to FIX 4 again. Key in 12345678 [.] 901 [**ENTER↑**], and you will get



12,345,678.90 10

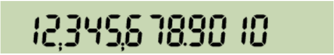
or



12.345.678,90 10

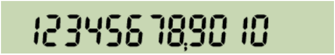
with TSON.[22]

Without these separators (i.e. with TSOFF), the same number will look like this



12,345,678.90 10

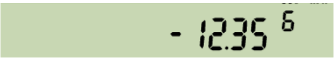
or



12345678,90 10

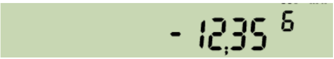
.

With ENG 3 and after [+/_], you will see



- 12.35 6

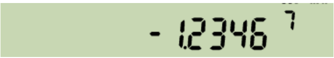
or



- 12,35 6

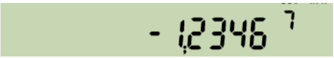
.

With SCI 4 , the same number will look like this



- 1.2346 7

or



- 1,2346 7

.

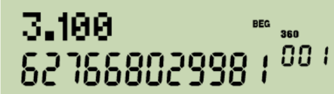
See also DISP in the *IOP*.

Floating point numbers within $10^{-383} \le |x| < 10^{+385}$ may easily be entered directly in the default DECM mode. Within this range, your *WP 31S* calculates with 16 digits and will display up to twelve usually (see the next point for an exception). Values < 10^{-398} are set to zero. For results $|x| \ge 10^{+385}$, error 4 or 5 will appear (see *App. B*).

SHOW displays the full number stored in $\mathbf{X}$, i.e. all 16 digits present internally in the mantissa of standard floating point numbers and all digits of the exponent almost like in scientific display format. All this is shown in one *temporary message*.

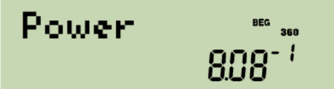
[22] These separators may also be beneficial in fraction mode described below.

E.g. π **3** y^x SHOW returns



reading 3.100 627 668 029 981 · 10^1 ,
i.e. 31.006 276 680 299 81.

For fitting measured and accumulated data points with a regression curve, four mathematical models are provided as in the *HP-42S*. See the commands EXPF, LINF, LOGF, and POWERF in the *IOP*. As shown in this **example**



,

the fit model applied is displayed *temporarily* after each command related to fitting (i.e. after CORR, COV, L.R., s_{XY}, $\hat{x}$, $\hat{y}$).

The command BESTF will set your *WP 31S* to select the model resulting in the greatest absolute *correlation coefficient* (see CORR). Like with all other auto-functionality, you should know what you are doing here.

Angles, Times, and Dates

There are two **angular modes** featured: DEG and RAD. And *degrees* may be displayed in decimal numbers as well as in *hours*, *minutes*, *seconds* and hundredth of *seconds* (H.MS). The usual angular conversions are provided:

Example: Set the display to FIX 4. Then enter the following: and you will see:

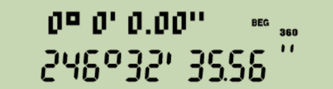
RAD — Choose *radians* as current angular mode.

π 300 /	0.0105	So π/300
→ DEG	0.6000	are exactly 0.6°
→ H.MS	0.3600	or 0°36'0".

The full spectrum of angular conversions provided is seen in the table following (see the *IOP* for more):

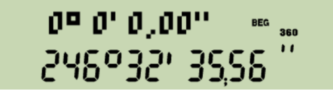
From	*degrees* H.MS	decimal *degrees*	*radians*	current angular mode
… to *degrees* H.MS	—	→ H.MS	—	—
… to decimal *degrees*	→ H.d	—	CONV rad→°	→ DEG
… to *radians*	—	CONV °→rad	—	→ RAD

In **H.MS display mode**, entered via H.MS, decimal numbers are converted and displayed as a *temporary message* formatted `hhhh°mm'ss.dd"`. With ***y*** = *0* and ***x*** = *246.54321* this may look like



0° 0' 0.00" DEG 360
246°32' 35.56 "

or

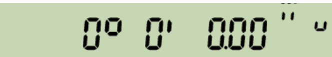


0° 0' 0,00" DEG 360
246°32' 35,56 "

for RDX**.** or RDX**,** , respectively.

For decimal times less than 5*ms* or 0.005 *angular seconds* but greater than zero, a u for underflow will be lit in the exponent section.

Example: EEX +/- 7 → H.d H.MS displays

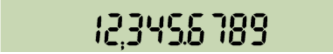

0° 0' 0.00" u

.

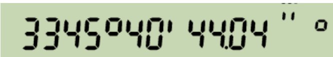
Note there are no leading zeroes in the *hours*, *minutes*, and *seconds* sections.

The number of *hours* or *degrees* is limited to 9 000 in this mode. For times or angles exceeding 9 000, an o is shown in the exponent section signaling an overflow, and the value is displayed modulo 9 000.

For **example**,

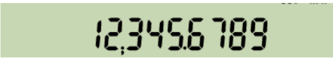

12,345.6789

will become after H.MS


3345°40' 44.04" o

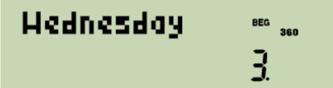
until the next key is pressed.

Then it returns to


12,345.6789

.

WDAY returns a display looking like the following for an input of 13.01201 in default D.MY mode (equivalent to inputs of 2010.0113 in ISO[23] Y.MD or 1.13201 in M.DY):


Wednesday BEG 360

3.

.

Expect similar displays after DAYS+.[24]

[23] ISO stands for the International Organization for Standardization. Among a lot of things standardized in the world of today (think of e.g. nuts and bolts – you do not have to cut your very own ones anymore but can get them from an arbitrary hardware store instead), date format is, too. It may help in sorting files properly. See the standard ISO 8601 in this matter.

[24] Calculation of weekdays for the past depends on the calendars used at that time – there may be different true results for different countries depending on the date the particular country introduced the Gregorian calendar. Officially, that calendar became effective in October 1582. Large parts of the world took

Dates will be displayed numerically according to the date mode chosen – and this mode (except startup default D.MY) will be indicated in the dot matrix as shown on pp. 34f unless YDON is selected.

Fractions

Fractions are handled like in previous *RPN* calculators (i.e. in *HP-32SII*, *HP-33S*, *HP-35S*, and *WP 34S*). In particular, DENMAX sets the maximum allowable denominator (see the IOP). Fraction mode is most easily entered via a b/c or by keying a second radix mark in numeric input; it is left towards floating point decimal mode via ALL, FIX, SCI, ENG, H.d, or H.MS.

The display will look like the examples below – fractions are adjusted to the left. If the fraction is exactly equal, slightly less, or greater than the floating point number converted, =, Lt, or Gt is indicated in the exponent, respectively. On your *WP 31S*, fraction mode can handle numbers with absolute values less than 100,000 and greater than 0.0001. The maximum denominator is 9 999. Underflows and overflows will be displayed in the format set before fraction mode was entered.

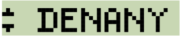
their time and switched in 1752, others joined even later (until 1949). See the commands JG1582 and JG1752 taking care of most cases. Luckily, today each and every country is in.
Dates before the year 8 A.D. may be indicated differently than they were at that time due to the inconsistent application of the leap year rule before then. We count on your understanding and hope this shortcoming will not affect too many calculations.
Note that 8 A.D. should be better written A.D. 8 or even A.D. VIII correctly – quite some false Latin is found in the English language. Nobody, however, counted years this way at that time – around the Mediterranean Sea, it was the year DCCLXI A.V.C. then. Also note the Julian calendar was introduced and became valid not earlier than DCCVIII A.V.C. – before, months were organized differently. Julius Caesar was murdered in DCCIX A.V.C.; calendars are a sensitive topic still.

Examples showing some displays you may see using your *WP 31S* in fraction mode:

Enter the following: and you will see:

0 FILL

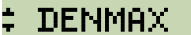
MODE D



⇕ DENANY

ENTER↑ initializes fraction mode.

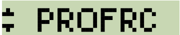
MODE D ▼ ▼ ▼



⇕ DENMAX

ENTER↑ sets maximum precision.

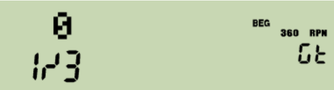
MODE P R



⇕ PROFRC

ENTER↑ sets proper fraction mode.

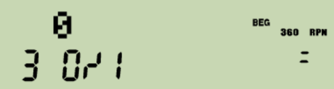
3 1/x



```
0                BEG 360 RPN
1/3                       Gt
```

since 1/3 > 0.333 333 333 333.

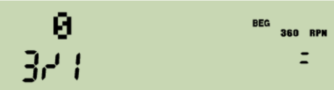
1/x results in



```
0                BEG 360 RPN
3 0/1                      =
```

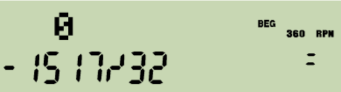
since this is exact.

Use MODE I ENTER↑ to display it as an *improper* fraction:



```
0                BEG 360 RPN
3/1                        =
```

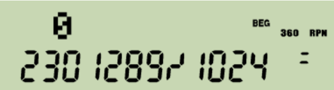
50 . **40625** −



```
0                BEG 360 RPN
-15 17/32                  =
```

– exact as well.

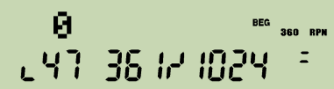
x^2 returns



```
0                BEG 360 RPN
230 1289/1024              =
```

.

Now, enter MODE P R ENTER↑ for converting that into a *proper* fraction.[25] You will get



with a little hook left of the first digit shown. This indicates the leading integer number is displayed incompletely – there are at least two digits preceding 47 here but no more display space. Press SHOW (see p. 38) to unveil the integer part of this proper fraction as 2247.

Numeric input in fraction mode is straightforward and logically coherent. If you enter two radix points, the first is interpreted as a space, the second as a fraction mark. Assuming proper fraction mode is chosen, see these **examples:**

Key in:	and get:	
1 2 . 3 . 4 ENTER↑	12 3⌐4	
1 . 2 ENTER↑	1 1⌐5	(decimal input)
. 1 . 2 ENTER↑	1⌐2	
. 1 2 ENTER↑	3⌐25	(decimal input)
1 . . 2 ENTER↑	1 0⌐1	(= 1 $^0/_2$) [26]

[25] Translator's note: *Proper fractions* cover both "echte Brüche" (like ¾) and "gemischte Brüche" (like 2 ½) in German.

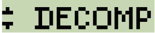
[26] This display of a pure integer number tells you unambiguously your *WP 31S* is in proper fraction mode. For comparison, note the *HP-32SII* reads 1 . . 2 as ½ – though this is not consistent with its other input interpretations in fraction mode (and does not even save keystrokes but adds confusion only).

By the way, please note that DECOMP indicates the precision of its results in a quite similar way as explained two pages above:

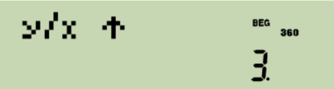
Enter the following: and you will see:

0.33333 .d

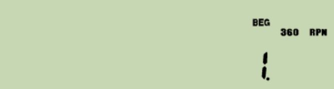
MORE D E



ENTER↑



x≷y



meaning $^1/_3$ > 0.333 33.

…

***CATALOGS* AND A BROWSER**

Due to the large set of operations your *WP 31S* features, most of them are stored in *catalogs* as was discussed from p. 26 on. The *catalog* CONST is special – it contains numeric constants instead.

Opening a *catalog* on your *WP 31S* will activate the grey letters on the keyboard to allow for typing the first character(s) of the item desired for rapidly accessing it. Then,

▲ and ▼ will browse the open *catalog*.

ENTER↑ selects the item displayed, recalls or executes it, and leaves the *catalog*.

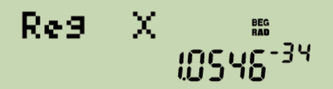
EXIT or ⬅ exit the *catalog* without executing anything, i.e. they cancel the *catalog* call.

You may switch *catalogs* easily by just calling a new one directly from the *catalog* you are browsing – no need to EXIT first.

When the last *catalog* called is reopened, the last item viewed therein is immediately displayed for easy repetitive use.

There is also a register browser featured: CATREG, meaning *catalog* of registers. It allows for viewing all 22 registers of your *WP 31S*, starting with **X**, and is operated in close analogy to the *catalogs* provided:

Calling CATREG, the first screen you see is (the numeric value may deviate):



Reg X

1.0546^{-34}

Within CATREG, the following keys will work exclusively:

▲ goes up the *stack*, continuing with the other lettered registers until **K**, then with **R0**, **R1**, etc.

▼ browses the registers going down from **R9** to **R0**, and then continuing with **K**, **J**, etc.

Input of any legal letter or any digit jumps directly to the corresponding register (see p. 30) and shows its contents.

ENTER↑ or RCL recall the register displayed and leave CATREG.

EXIT or ⬅ will exit CATREG without recalling anything.

See the next pages for full contents of the eight *catalogs* provided. Items are sorted alphabetically within each *catalog* (see p. … for the sorting order). You may access a particular item quickly by typing the first character(s) of its name – see the *IOP* for examples.

Catalog Contents in Detail

CLEAR

CLALL
CLREGS
CLΣ
RESET

CONST

See p.

CONV

See p.

DISP

ALL
DSP
ENG
FIX
RDX,
RDX.
SCI
TSoff
TSon
YDOFF
YDON

DISTR

Binom
Binom$_P$
Binom$_u$
Binom $^{-1}$
Cauch
…
Expon
…
F$_P$(x)
F$_u$(x)
F(x)
F $^{-1}$(p)
Logis
…
Norml
…
Poiss
…
t$_P$(x)
t$_u$(x)
t(x)
t $^{-1}$(p)
Weibl
…
$\boldsymbol{\chi}^2$
χ^2 **I**NV
$\chi^2{}_p$
$\chi^2{}_u$

MODE

BestF
DENANY
DENFAC
DENFIX
DENMAX
D.MY
ExpF
IMPFRC
JG1582
JG1752
LinF
LogF
M.DY
PowerF
PROFRC
SSIZE4
SSIZE8
Y.MD

MORE

$\sqrt[3]{x}$
ANGLE
BATT
CATREG
DAYS+
DECOMP
FP
GCD
IP
LCM
LOAD
MOD
NEXTP
PRIME?
RMDR
SAVE
SSIZE?
VERS
WDAY
WHO
$\sqrt[x]{y}$
ΔDAYS
%
%MRR
| |

STAT

COMB
L.R.
nΣ
PERM
SERR
SERR$_W$
SUM
$\mathbf{s_w}$
x̄$_W$
Σln^{2}x
Σln^{2}y
Σlnx
Σlnxy
Σlny
Σx
Σx^2
Σx^{2}y
Σxlny
Σxy
Σy
Σy^2
Σylnx

APPENDIX A: CREATING AND UPDATING YOUR *WP 31S*

Your *WP 31S* is a spin-off of the *WP 34S* as mentioned above. Thus, transmitting the *WP 31S* firmware to the *FM* of an *HP-20b* or an *HP-30b* calculator will work exactly as it does for the *WP 34S*. The one and only difference is the location of the binary files to load on your computer. …

How to Flash Your *HP-20b* or *HP-30b*

You cannot buy a *WP 31S* pre-flashed; but flashing is no witch work – you can do it yourself. You need an *HP-20b* or *HP-30b* calculator, a special cable, a binary file to load on your computer, and software for the transmission to your calculator.

- The transmission software is called *MySamBa*. Open http://sourceforge.net/projects/wp34s/files/FlashTool/ , download `MySamBa.zip` and unpack it.

- The binary file you need to transmit to your calculator to make it your *WP 31S* is called `calc.bin` ; download it from http://sourceforge.net/p/wp34s/code/HEAD/tree/branches/wp31s/realbuild/

- For the cable, there are two alternatives as described in the *WP 34S* manual available.

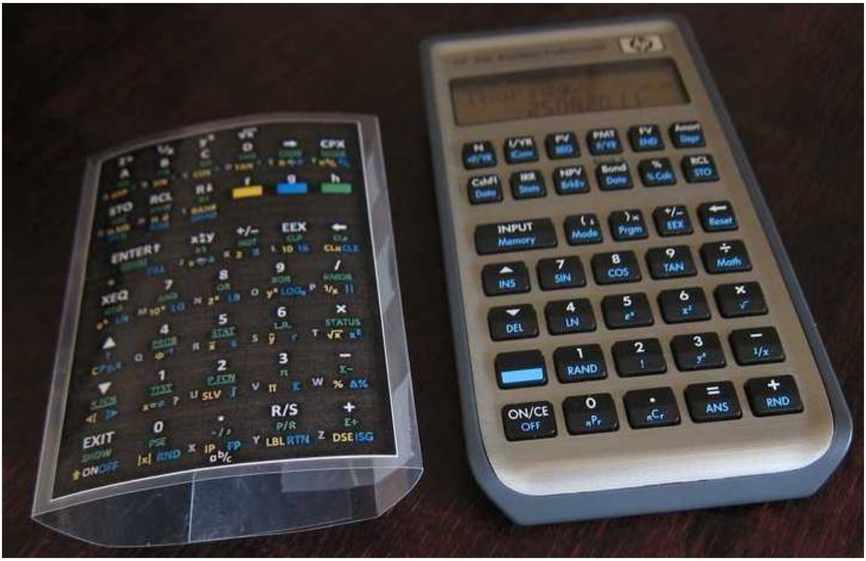
…

Having prepared your calculator, computer, software, file, and cable, follow one of the two alternative procedures A or B as described in the *WP 34S* manual available for transforming an *HP-20b* or *HP-30b* into a *WP 31S*.

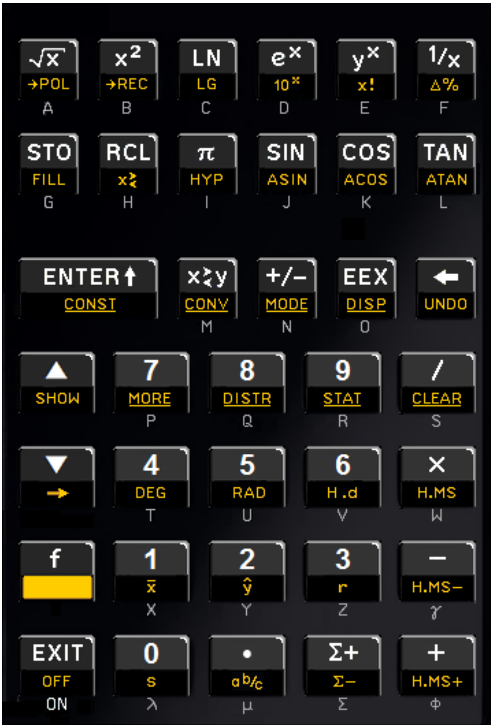
Overlays

At the time this text was written, no dedicated *WP 31S* vinyl overlays were available yet. As long as it will take to get them, you may help yourself by taking a picture like the one shown on next page. Cut it out (it shall be 64.5 *mm* wide), span it over your *WP 31S* using transparent adhesive tape, and you are done – the flexibility of this setup allows actuating the keys though there are no holes in your overlay.

If you want to add some extra durability to your paper overlay, you can laminate it. By folding and closing it as shown below,[27] you can create a 'sleeve' for your former financial calculator. It will be considerably stiffer than pure paper but may serve you well until the vinyl overlays will become available and arrive in your mailbox, even if that will take a little longer. You may want to check www.hpmuseum.org/forum periodically to learn about availability of overlays.



[27] Note that photograph is of a laminated overlay for a *WP 34S*.

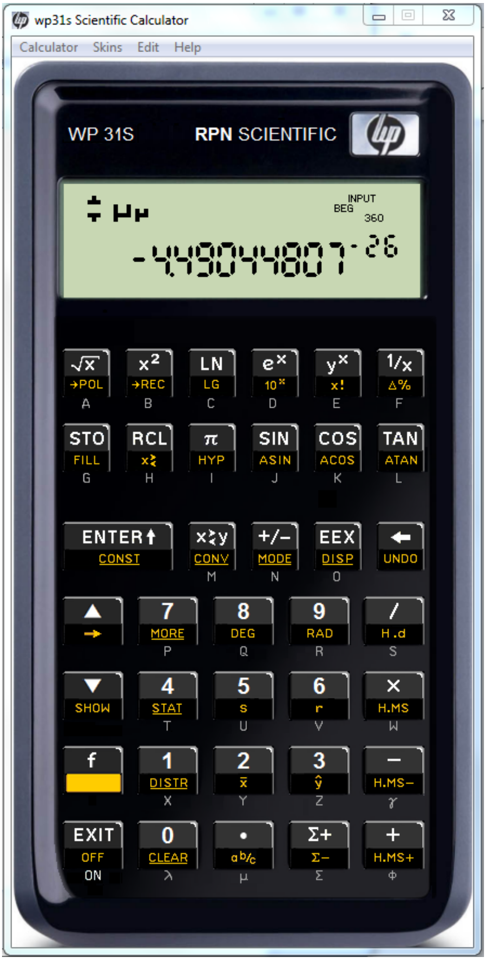

√x →POL A
x² →REC B
LN LG C
eˣ 10ˣ D
yˣ x! E
1/x Δ% F
STO FILL G
RCL x⇄ H
π HYP I
SIN ASIN J
COS ACOS K
TAN ATAN L
ENTER↑ CONST
x⇄y CONV M
+/− MODE N
EEX DISP O
← UNDO
▲ SHOW
7 MORE P
8 DISTR Q
9 STAT R
/ CLEAR S
▼ →
4 DEG T
5 RAD U
6 H.d V
× H.MS W
f
1 x̄ X
2 ŷ Y
3 r Z
− H.MS− γ
EXIT OFF ON
0 s λ
• ab/c μ
Σ+ Σ− Σ
+ H.MS+ Φ

EMULATING A *WP 31S* ON YOUR COMPUTER

If you own or operate a Windows-PC, you may download

http://sourceforge.net/p/wp34s/code/HEAD/tree/branches/wp31s/windows/bin/wp31sgui.exe

Download the files `emulator.dll`, `wp31s_medium.bmp`, `4 Medium 31s.skin` of the same build number as well.[28] Then just start the exe file by double clicking. This is our test application featuring an identical function set as your WP 31S calculator. That emulator goes along with its development.



Typically the emulator is operated by the computer mouse. You click on the corresponding areas of the keyboard image instead of pressing keys. Right-clicking on a key calls its shifted function. Right-clicking on the WP symbol opens a menu. Everything else is pretty self-explanatory. Just try it out.

There are also (so-called Qt) emulators available working for other operating systems (e.g. Linux, iOS).

[28] Note the name of the skin file contains two blanks.

RELEASE NOTES

	Date	Release notes
	25.1.14	Project start with first publication on the forum.
1.0	1.2.14	Manual setup based on the manual of the *WP 34S*.
1.1	22.2.14	Revisions according to the new layout featuring UNDO.
1.2	10.3.14	Layout change to move CONST to ENTER and put x! on a key. Put all catalogs in the two central rows of keys.
	1.5.14	This text is an extract of the manual to be published. Note the ellipses and some red print.