

khs

instruments

khs-calc
User Manual

Rev. 1.1

Content

1.1) About This Manual	1
1.2) About khs-calc	1
2.1) Installing khscal on Windows NT, Windows95/98	1
2.2) Uninstalling khs-calc	1
3.1) First Steps	2
3.2) Stack Operations	3
3.3) First steps with Floating Point Operations:	3
3.4) In the integer world:	4
4.) Programming	6
Appendix A:	
Description of the Operations and the Syntax of the Script Interface	7
A.1) Mode Switching and Other Commands	7
A.2) Stack Operations	7
A.3) Memory Operations	8
A.4) Arithmetic Operations	8
A.5.1.2) Stack Operations	8
A.5.1.2) Logical Operations in HEX, INT, UINT, OCT and BIN mode only:	9
A.5.2.1) General Operations in ENG, SCI, and FD mode only:	9
A.5.2.2) Arithmetic Operations in ENG, SCI, and FD mode only:	9
A.5.3.1) General Trigonometric Operations in ENG, SCI, and FD mode only:	9
A.5.3.2) Trigonometric Operations in ENG, SCI, and FD mode only:	10
A.5.3.3) Inverse Trigonometric Operations in ENG, SCI, and FD mode only:	10
A.5.3.4) Hyperbolic Trigonometric Operations in ENG, SCI, and FD mode only:	10
A.5.3.5) Inverse Hyperbolic Trigonometric Operations	10

1.1) About This Manual

This manual describes the installation and removing of the khs-calc. This manual also describes first steps using the khs-calc.

1.2) About khs-calc

There was two reasons creating the khs-calc: The first was to test and improve the components of the software construction kit. The second reason was the need to simplify some calculation in our hardware design. Often there was to make some medium complex calculations - one example is shown in the programming example: Calculating the resistance of a temperature sensor at a certain temperature. If you looking for more examples, please visit the web site:

www.khs-instruments.com

and select the application section. To run the examples in the application notes, copy and paste the code into an editor and store the file under fx.run (x=1..6) in the current directory of the khs-calc. For more information please refer to the programming example in this manual.

2.1) Installing khscalC on Windows NT, Windows95/98 systems

All the required files are contained in the self-extracting zip file KHSCALC.ZIP. First, you have to extract the files using an utility such as WinZip or Pkunzip. The files contained in KHSCALC.ZIP will then be unpacked into the current directory.

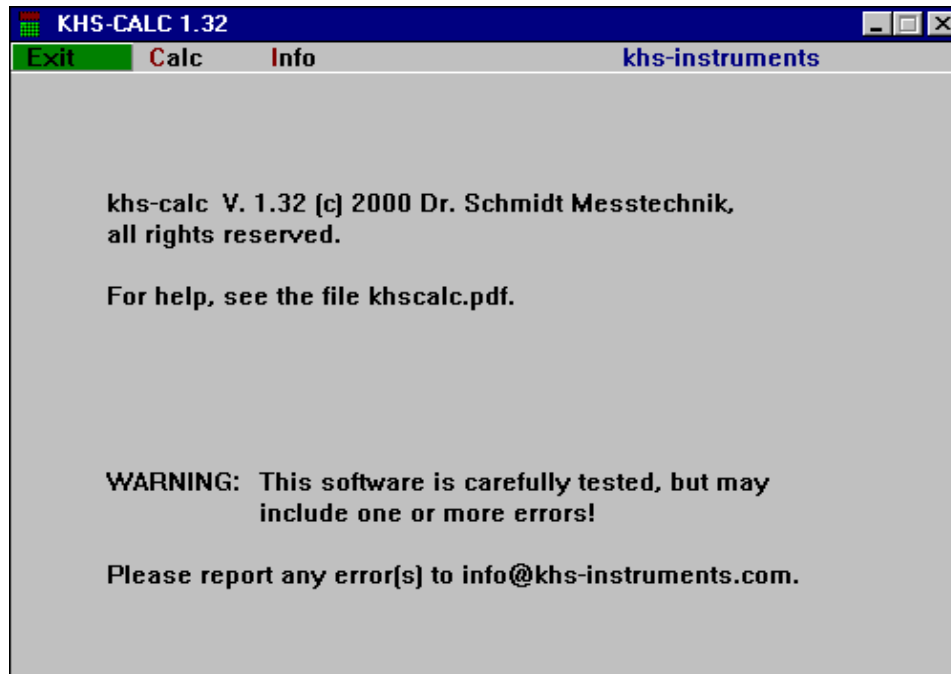
Once the files are unpacked, khs-calc is ready to run. You can do this by double clicking on the file CALC.EXE from Explorer, or drag CALC.EXE onto the desktop to create a shortcut. You can right-click on the Start button, select open, and drag CALC.EXE onto the Program folder (or another folder if you prefer) to add khs-calc to the Start menu.

2.2) Uninstalling khs-calc

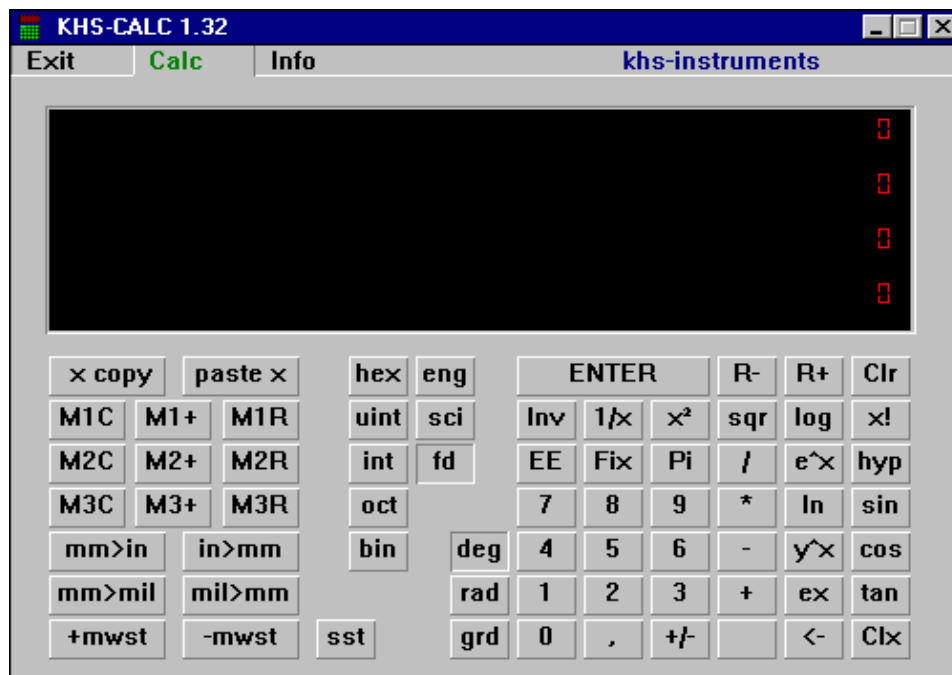
If you need to remove khs-calc from your system, simply delete the khs-calc directory and its contents. Khs-calc use no any registry settings or entries.

3.1) First Steps

After successful installation double click the khs-calc icon. You will see a screen like this:



Now move the cursor to the calc position and left click. Now you see the calculator in floating point mode: The 'Esc' key brings you back to the main menu. X,Y registers memory and stack are not cleared. Alternatively simply left-click 'calc' again.



The khs-calc is now ready for calculations.

Please note:

Before using khs-calc first, please note: The khs-calc uses the *reverse polish notation*. This means: First enter the operand and push onto the stack and afterwards enter the operation.

3.2) Stack Operations

The khs-calc provides four stack levels: One X-register, one Y-register and two additional registers. The 'ENTER' button transfers the data from the X-Register to the Y-Register. The old Y-register is pushed on the stack. The operations R- and R+ rotates the stack downwards respectively upwards.

The operations +, -, x,... uses the X and Y register. The result is placed into the X-register. If you enter a number after a such a operation, a 'ENTER' is inserted automatically.

3.3) First steps with Floating Point Operations:

To get familiar with the khs-calc first we want to make some floating point operations:

Lets calculate $\pi / 1.5 \text{ E } -4 + 3$. To do this:

- | | |
|-------------------------|---|
| 1) Press the Pi button. | (Pi is in the X Register) |
| 2) Press ENTER. | (ENTER shifts Pi into the Y register) |
| 3) Press 1.5 | |
| 4) Press EE | |
| 5) Press 4 | |
| 6) Press +/- | (1.5 E-3 is in the X Register) |
| 7) Press the / button. | (Calculates Y/X puts the result in the X register. You should get as a result 2094.4) |
| 8) Press 3 | 3 is now in the X register. The former result is shifted into the Y register) |
| 9 Press + | |

You should get as a result 20946.9510

Now we want to get less post decimal positions. To do this, press the 'Fix' button. Please note, the button stay pressed. Assume, we want only one post decimal positions. Simply press '1'.

Now we get as a result: 20947.0

Let us look how it looks in the scientific notation. To do this, press the 'sci' button. The result is:
2.1e4

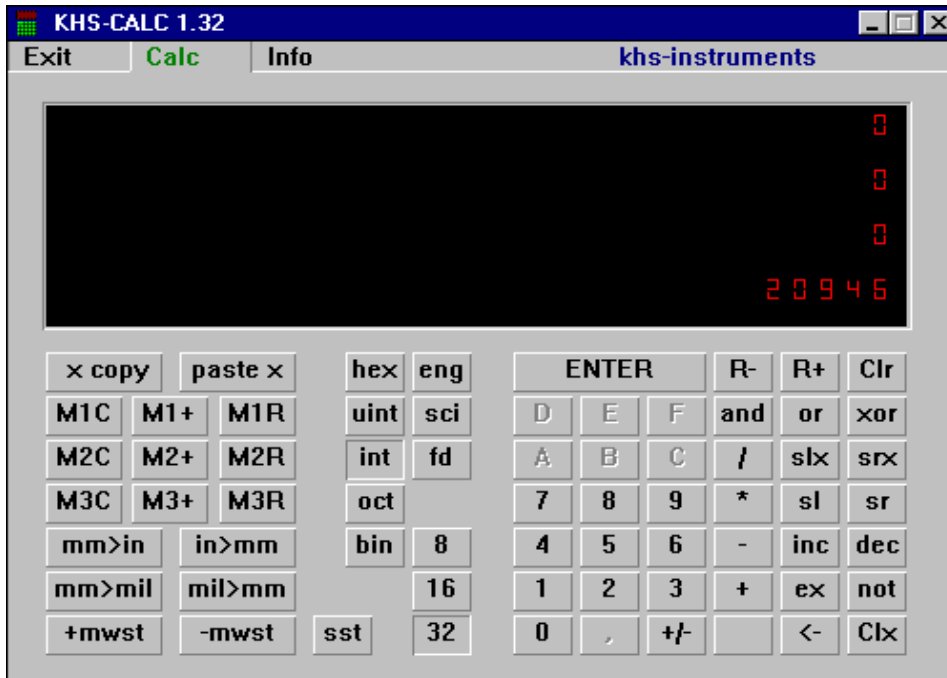
Sometime it's useful to get the result in engineering format. Simply press the 'eng' button. Now we get:
20.9e3

If we no longer want the fixed display we can switch it off. First press the 'Inv'. Please note, the 'Inv' button stay pressed. Now press 'Fix' and you should get as a result
20.947e3

The 'Inv' button works with the following functions: Fix, Log, Ln, sin, cos, tan.
The 'Hyp' Button work with the functions sin, cos, tan.

3.4) In the integer world:

But now we want to go into the world of bits, bytes and logical operations: To do this, press the '*int*' button. The display changes to:



Please note, there are no more '*ln*', '*cos*', and other functions. Instead there are '*and*', '*or*' '*srx*' and other integer functions: Now we are in the world of bits and bytes.

First we are curious, what's 20946 in hexadecimal? To get this information press the '*hex*' button. 51D2.

If you are not familiar with hexadecimal numbers try counting:

First clear the display with the '*Clr*' button. Now press the '*inc*' button several times. You can observe counting in hexadecimal numbers:

1, 2, .. 9, A, .. F, 10, and so on.

To count down press the '*dec*' button.

If you want, you can try counting in other display format like octal ('*oct*') and binary ('*bin*') numbers.

Now we want to test the logical functions. To do this clear ('*Clr*') the display and enter the binary mode ('*bin*').

Now we want to get 1101b or'ed with 10b. To get the result

press 1101 and then enter. This pushes 1101 onto the stack.
Now enter 10

You get the result by pressing '*or*'. You should get 1111. In the same way try '*and*' and '*xor*'.

If we want to shift a result, we have to press '*sl*' to shift left and '*sr*' to shift right. Enter 100 and try '*sl*' and '*sr*'.

But, if we want to shift '1111' eight times left, there is an easier way. Push the 1111 onto the stack, switch into the integer mode ('*int*') and enter 8. Switch back to binary mode '*bin*' and press '*slx*', shifting the value of the stack eight times left. You should get as result:

111100000000

With the +/- button you can calculate negative values: Press '+/-' and you get the negative value of 111100000000:

11111111111111111111000100000000

If you work with 16 bits only, you want to ignore the upper 16 bit if the result. Simply press the '16' button. The result is:

1111000100000000

This was a short overview of the khs-calc. Next step is learning how to program the khs-calc.

4) Programming

We want to calculate the resistance R_T of a resistance temperature sensor (in our case a KTY 10 from Siemens/Infineon) at the temperature T . The equation is:

$$R_T = R_{25} \cdot \left(1 + A \cdot \frac{T - 25}{B} + C \cdot \left(\frac{T - 25}{B} \right)^2 \right) \cdot \exp \left(\frac{D}{T + 273.15} \right)$$

$$\frac{T - 25}{B} = \frac{T_A - 25^\circ\text{C}}{B}$$

$$A = 7.68 \cdot 10^{-3} \cdot K^{-1}$$

$$D = 7.68 \cdot 10^{-3} \cdot K^{-1}$$

For programming we use the lettering on the buttons, the only thing we have to know, is we have to make a * before the command. (For a complete list of commands please refer to the next chapter) So lets calculate:

We assume, the temperature in °C is already in the X-register.

*REM	Resistance of an temperature sensor
*title R(t)	;shows R(t) on the F1 button
*ENTER	;First we have to subtract 25°C.
*NUM 25	
*_	
*STO_1	;store into the memory 1
*x²	;square
*ENTER	;onto stack
*NUM 1.88e-3	
**	;multiply by beta
*ENTER	;result onto stack
*RCL_1	;temperature
*ENTER	;onto stack
*NUM 7.68e-3	
**	;multiply by alpha
*+	;add with beta *T²
*ENTER	
*NUM 1	
*+	;add 1
*ENTER	;onto stack
*NUM 2000	;resistance at 25°C
**	;multiply

To use this program, use the copy function, copy this program, open the editor and paste it. Store the file into the current directory of the khs-calc and use the filename 'f1' .

Please note: The Windows stores the file under f1.txt. Use the windows explorer to change the filename into 'f1.run'

If you enter 25 and press the R(t) button, you should get as a resistance the value 2000.

Appendix A: Description of the Operations and the Syntax of the Script Interface

This section describes the program on the desktop (the buttons) and the operators of the khs-calc '*Fx.run*' files.

The optional script files '*F1.run*' to '*F6.run*' are called when the buttons F1 to F6 are pressed. You can mark your own title to the button F1 to F6, if you use the command `*title <string>` in the '*Fx.run*' file. This make the button Fx marked with `<string>`.

The ASCII- files F1.RUN to F6.RUN can be written with the Windows-editor.

All command begins with a '*' character, and all comments begin with a ';' or the *REM operator.

A1) Mode Switching and Other Commands:

Operator	Button	Key	Operation
*REM			for comments
*title str	F1 to F6		str marks the F button with <code><str></code> F1..F6 (max. 6 characters)
*INT	int		Input and display integer
*UINT	uint		Input and display unsigned integer
*HEX	hex		Input and display hexadecimal
*OCT	oct		Input and display octal
*BIN	bin		Input and display binary
*ENG	eng		Input and display
*SCI	sci		Input and display
*FD	fd		Input and display

A2) Stack operations:

Operator	Button	Key	Operation
*ENTER	ENTER	ENTER	0->R1; Y->R0; X->Y; X=0;
*R+	R+		R0->R1; Y->R0; X->Y; X=R1;
*R-	R-		X->R1 ; R1->R0; R0->Y; Y->X;
*EX	EX		X->Y; Y->X
*CLR	Clr		X=0;

A3) Memory Operations

Operator	Button	Key	Operation
*STO_1			memory 1 = X
*STO_2			memory 2 = X
*STO_3			memory 3 = X
*CLR_1	M1C		memory 1 = 0
*CLR_2	M2C		memory 2 = 0
*CLR_3	M3C		memory 3 = 0
*ADD_1	M1+		memory 1 = memory 1+X
*ADD_2	M2+		memory 2 = memory 2+X
*ADD_3	M3+		memory 3 = memory 3+X
*RCL_1	M1R		X = memory 1
*RCL_2	M2R		X = memory 2
*RCL_3	M3R		X = memory 3

A4) Arithmetic Operations

Operator	Button	Key	Operation
*+	+	+	$X=X+Y$
*-	-	+	$X=X-Y$
**	*	*	$X=X * Y$ (Product)
*/	/	/	$X=X/Y$
*CHS	+/-		$X=-X$
*NUM	(1 to 0, ',', ',')		$X=NUM$

A5.1.1) General Operations in HEX, INT, UINT, OCT and BIN mode only:

Operator	Button	Key	Operation
*8_BIT	8		Display with 8 bits
*16_BIT	16		Display with 16 bits
*32_BIT	32		Display with 32 bit

A.5.1.2) Logical Operations in HEX, INT, UINT, OCT and BIN mode only:

Operator	Button	Key	Operation
*AND	and		$X = X \& Y$
*OR	or		$X = X \text{ or } Y$
*XOR	xor		$X = X \text{ xor } Y$
*SR	sr		$X \gg 1$
*SL	sl		$X \ll 1$
*SRX	srx		$X = Y \gg X$
*SLX	slx		$X = Y \ll X$
*INC	inc		$X = X + 1$
*DEC	dec		$X = X - 1$
*NOT	not		$X = X \text{ xor } 0\text{xffffffff}$

A.5.2.1) General Operations in ENG, SCI, and FD mode only:

Operator	Button	Key	Operation
*fix n	fix		Enter number of digits
*inv	inv		
*fix	fix		Variable number of digits (Switch off 'Fix')

A.5.2.2) Arithmetic Operations in ENG, SCI, and FD mode only:

Operator	Button	Key	Operation
*1/x	1/x		$X = 1/X$
*x ²	x ²		$X = X * X$
*sqr	sqr		S = square root of X
*y ^x	y ^x		$X = Y^X$
*e ^x	e ^x		$X = e^X$
*ln	ln		$X = \ln(x)$ (base e)
*log	log		$X = \log(X)$ (base 10)
*inv	inv		
*log	log		$X = 10^X$
*x!	x!		$X = X!$

A.5.3.1) General Trigonometric Operations in ENG, SCI, and FD mode only:

Operator	Button	Key	Operation
*deg	deg		Angle = 0..360
*rad	rad		Angle = 0..2 Pi
*grd	grd		Angle = 0.100
*Pi	Pi		$X = 3.1415...$

A.5.3.2) Trigonometric Operations in ENG, SCI, and FD mode only:

Operator	Button	Key	Operation
----------	--------	-----	-----------

*sin	sin	$X=\sin(X)$
*cos	cos	$X=\cos(X)$
*tan	tan	$X=\tan(X)$

A.5.3.3) Inverse Trigonometric Operations in ENG, SCI, and FD mode only:

Operator	Button	Key	Operation
*inv	inv		
*sin	sin		$X=\sin(X)$
*inv	inv		
*cos	cos		$X=\cos(X)$
*inv	inv		
*tan	cos		$X=\tan(X)$

A.5.3.4) Hyperbolic Trigonometric Operations in ENG, SCI, and FD mode only:

Operator	Button	Key	Operation
*hyp	hyp		
*sin	sin		$X=\sinh(X)$
*hyp	hyp		
*cos	cos		$X=\cosh(X)$
*hyp	hyp		
*tan	cos		$X=\tanh(X)$

A.5.3.5) Inverse Hyperbolic Trigonometric Operations in ENG, SCI, and FD mode only:

Operator	Button	Key	Operation
*inv	inv		
*hyp	hyp		
*sin	sin		$X=\sinh(X)$
*inv	inv		
*hyp	hyp		
*cos	cos		$X=\cosh(X)$
*inv	inv		
*hyp	hyp		
*tan	cos		$X=\tanh(X)$