



SpeedUI More Info

- [Intro](#)
- [Conversion Tool](#)
- [CF.LIB](#)
- [UI.LIB](#)
- [UF.LIB](#)
- [AB.LIB](#)
- [Full Screen Browser](#)
- [Memory Requirements](#)
- [Variables in the Hidden Directory](#)
- [Features of the individul Libraries](#)
- [Obsolete Keyboard Libraries](#)
- [The new Library Catalog](#)
- [MsgBox & Choose Libraries](#)
- [Multi Equation Solver](#)
- [SpeedUI header line options](#)
- [Credits](#)

Intro:

This page contains some technical details about **SpeedUI**, the modular and speedy user interface replacement for the HP-48 G series. Be sure to have read the **Start Here** page.

Settings Conversion Tool:

From SpeedUI version 12.04 on, all SpeedUI settings will be stored in a single variable named SUIpar. SUIpar replaces all individual vars used in older versions of SpeedUI. The only exception is the dynamic clipboard, which will be stored in a separate variable as before. The program ToSUIpar reads all the individual settings, and combines them into the new single var named SUIpar. ToSUIpar also offers to delete the old variables, which the user should answer with YES, since in SpeedUI versions from 12.04 on, the old variables will not be used anymore.

If SUIpar doesn't exist, it will be created on first access using default values. The SpeedUI stack environment will store some temporary settings into SUIpar upon regular exit.

Since SUIpar holds all relevant SpeedUI settings, even those not held in normal vars so far, it will be much easier to backup/restore SpeedUI settings from now on.

Naming conventions regarding CF.LIB and UI.LIB:

CF.LIB and CF_lite.LIB have the same basic functionality. They only differ in the speedup rate of some functions, and the advanced features included, and therefore the program size. The same applies to UI.LIB and UI_lite.LIB . In this document the name CF.LIB stands for either CF.LIB or CF_lite.LIB, and the name UI.LIB stands for UI.LIB or UI_lite.LIB.

CF.LIB:

CF.LIB configures the system to use the replacement libraries instead of the built-in libs. This is because the built-in system configuration code of the G will not activate the replacement library if stored in a port != 0, or in some system configurations with cards in either port 1 and/or 2. Instead, the built-in version is activated. So CF.LIB helps us out of this limitation. It guarantees that the replacement libraries will be activated, regardless of their location and system memory configuration. CF.LIB does not have any user accessible keywords, these are in UF.LIB (see below) , but includes routines shared by other SpeedUI components. Additionally, the user can activate the SpeedUI boot configuration utility by holding the backspace key during a warmstart. The user will first be asked if he/she really wants to activate the SpeedUI config utility.

Active keys at this point are:

NO or CANCEL normal startup will proceed

YES SpeedUI config utility will be activated

Active keys in the SpeedUI boot config utility are:

DownArrow Move highlight to next item

UpArrow Move highlight to previous item

RightArrow Toggle status of highlighted item

ENTER/OK Accept chosen settings and continue startup

CANCEL/ATTN Continue startup w/o saving changes

The individual entries and their default settings are:**Entry Default Choice**

BOOT Cfg all Built-in only

WARP ON OFF

The (default) selected entry on starting the config util is WARP.

BOOT: can act as a config suppressor.

If set to 'Built-in only', no external library (including CF.LIB), will be configured during this startup. This feature is useful for accessing the calc when an ill-behaved library locks the machine. This setting will revert to default at the next warmstart.

WARP: toggles the global WARP switch.

If WARP is ON, all installed SpeedUI components will be activated. If UI.LIB is installed, CF.LIB will also try to start the SpeedUI shell at startup. If WARP is OFF, the built-in libs will be activated, and CF.LIB doesn't try to start the SpeedUI shell at startup. The SpeedUI shell can still be started manually, using the appropriate menu key in UI.LIB . If UF.LIB is installed, WARP status can also be toggled by entering WARP into the command line, followed by pressing ENTER, or just using the appropriate menu key in UF.LIB .

In the lite version, several features are disabled. These include:

- No ML decompiler. This means decompiling objects for display/edit will be done by the slow internal decompiler
- Multiline object display in SpeedUI stack will be considerably slower than in full version

UI.LIB:

UI.LIB is the fast stack replacement, and offers a user interface for certain SpeedUI management functions.

UI.LIB has the following user keywords:

<u>Name</u>	<u>Lib-ID</u>	<u>Description</u>
S4567	805.000	Toggles Stack environment (-->)
DISPRN	805.001	Display string in line r in font n (\$ row fontnumber -->)
TFNT	805.002	Toggle Font by ascending font size
THL	805.003	Toggle Header Line count by descending count
->FNTN	805.004	Select font indicated by number (fontnumber -->)
FNTN->	805.005	Get number of selected font (--> fontnumber)
->HDRN	805.006	Select header line count indicated by number (headerlines -->)
HDRN->	805.007	Get number of selected headerlines (--> headerlines)
CAT	805.008	Display the keyword catalog, with optional echo to cmd line
->I	805.009	Convert either Real number or HXS string into BINT
->R	805.010	Convert either BINT or HXS string into real number
->H	805.011	Convert either Real number or BINT into HXS string
INEG	805.012	Like NEG, but handles BINTs, too
I+	805.013	Like +, but handles BINTs, too
I-	805.014	Like -, but handles BINTs, too
I*	805.015	Like *, but handles BINTs, too
I/	805.016	Like /, but handles BINTs, too
USAG	805.017	Shows parameter types and order ({keyword} -->)
VV	805.018	Fast Object Viewer (Ob --> Ob)
AGROB	805.019	Object to GROB converter (Ob fontnumber --> GROB)

In the lite version, several features are disabled. These include:

- No BINTS submenu in MTH menu and no direct BINT/HXS/Real arithmetic and conversion
- No "Pretty Print", and thus no VV and AGROB functionality
- No error message display format option. Only default message format

The associated keywords stay there to preserve the internal library structure. In the lite version, these keywords either beep or just do nothing.

UF.LIB:

UF.LIB offers a user interface for certain SpeedUI management functions.

UF.LIB has the following user keywords:

<u>Name</u>	<u>Lib-ID</u>	<u>Description</u>
SETUPD	809.000	Starts the SpeedUI configuration special 5-line input form
QSM	809.001	Launch Quick Start Menu (QSM)
Usag	809.002	Launch USAG command, which shows parameters for all HP-48 keywords
MRCSTO	809.003	Set initial row/column coordinates for MatrixWriter replacement
MRCRCL	809.004	Get current row/column coordinates for MatrixWriter replacement
RPSetup	809.005	Starts the Repeater setup input form
RPSTO	809.006	Set delay for repeater ({ %d1 %d2 %d3 } -->)
RPRCL	809.007	Get current delay for repeater (--> { %d1 %d2 %d3 })
SUIpSTO	809.008	Set SUIpar (\$ -->)
SUIpRCL	809.009	Get current SUIpar (--> \$)
INVRECT	809.010	Invert a rectangular section of a GROB
SWAPLIB	809.011	Activates the next/prev inactive Lib with same ID
WARP	809.012	toggles WARP (-->)
NSBint	809.013	toggles the NoShift BINT manipulation feature on and off
NSAlpha	809.014	toggles the NoShift Alpha lock feature on and off
RSAlpha	809.015	toggles the RightShift repl. lib on and off
WRP?	809.016	returns state of variable _warp (--> %1 %0)

AB.LIB:

AB.LIB offers fast GROB and animation viewing, and some related management functions.

AB.LIB has the following user keywords:

<u>Name</u>	<u>Lib-ID</u>	<u>Description</u>
ABR	1200.000	Animation Browser (-->)
APLAY	1200.001	Play Animation from stack (? --> ?)
GV	1200.002	Graphics Viewer (? --> ?)
UPGROB	1200.003	Extract RLE packed GROB (pGROB --> GROB)
PGROB	1200.004	Pack GROB into RLE packed GROB (GROB --> pGROB)
->arr	1200.005	Convert meta list into array (Obn..Ob1 n --> [..])
arr->	1200.006	Convert array into meta list ([..] --> Obn..Ob1 n)

ABR searches the current directory for variable names with an *ARY extension, and lists the found variables using the SpeedBrowser. Pressing ENTER on the highlighted item will start the animation.

While running, ABR has the following key assignments:

<u>Key</u>	<u>Description</u>
UpArrow	Increase animation speed
DownArrow	Decrease animation speed
LeftArrow	Switch to reverse animation direction
RightArrow	Switch to normal animation direction
ON	Exit animation

GV takes one of the following parameters:

<u>Parameter</u>	<u>Description</u>
GROB	Graphics Object
pGROB	RLE packed GROB
{GROB/pGROB x y}	GROB with initial start coordinates
?	any combination of the above

GV has the following key assignments:

<u>Key</u>	<u>Description</u>
Menu key A (F1)	Start AnimationBrowser with current ob as input (if applicable)
Cursor keys	Move displayed window around
Next	Show next GROB in list
' (Tick)	Set upper left corner to initial or default coordinates
EVAL	Run predefined script from current directory
ENTER	Switch to GBUFF at curr. window position (if possible). Uses Coords from PPAR
+/-	Choose keyboard repeat delay
7	Move displayed window one pixel up and one pixel left
8	Move displayed window one pixel up. Same as UpArrow
9	Move displayed window one pixel up and one pixel right
4	Move displayed window one pixel left. Same as LeftArrow
5	Center GROB in display
6	Move displayed window one pixel right. Same as RightArrow
*	Set current upper left corner as default
1	Move displayed window one pixel down and one pixel left
2	Move displayed window one pixel down. Same as DownArrow
3	Move displayed window one pixel down and one pixel right
-	Switch to previous keyboard repeat delay . This key is a repeater
ON	Exit program
0	Move displayed window to coordinates 0,0
.	Set upper left corner to coordinates stored by * (initially 0,0)
Space	Show current window coords and GROB dimensions
+	Switch to next keyboard repeat delay . This key is a repeater
LS+Cursor keys	Move displayed window around pagewise
LS+Next	Show previous GROB in list
LS+ENTER	Switch to GBUFF at curr. window position (if possible). Coords are pixel coords
RS+Cursor keys	Move displayed window around to respective corners

Full Screen Browser:

SpeedUI up to version 11.04 was shipped with a standalone version of the SpeedBrowser, which is a fast replacement for the slow built-in full screen browser. It's called from the equation library, constant library, multi equation solver, periodic table, and some other programs. From SpeedUI 11.04 on, the SpeedBrowser versions shipped with SpeedUI are not standalone anymore, but require the presence of the SpeedUI core library CFLIB . SpeedBrowser now uses some functionality from the core library. The goal was to reduce program size and overhead, and thus the SpeedUI SpeedBrowser size could be reduced from about 6.7KB to about 4.2KB . Better yet, as a very compact alternative SpeedBrowser Ultra Compact is provided, which is only 1.4KB in size. The latter doesn't use the SpeedBrowser engine, but acts as a wrapper or translator for the G built-in Choose engine. This means all programs which call the full screen browser will be redirected from SBUC to the built-in Choose engine. Normally all programs which use the officially supported entry points of the full screen browser should work with SBUC, too. If an application doesn't behave well with SBUC, one can still use SB.LIB , which should work in any case. The long-term target is to move away from the SpeedBrowser to the Choose engine, or more exactly to my replacement of the Choose engine. The Choose engine is much more flexible than the old full screen browser, and also supports UFL fonts, so programs which utilize a browser should use the Choose engine API.

Nonetheless the old standalone GX/G/G+ SpeedBrowser is available separately, as well as a special version for the SX/S .

Memory requirements:

SpeedUI consists of various components, of which only one (CF.LIB) is always needed. If you're low on memory or using an HP 48G with only 32K RAM, you could use the lite versions of CF and UI, together with UFL1 or UFL3. However, even with the lite versions the performance gain is noticable instantly;-) Of course you can use SpeedUI components without using the accelerated stack interface. If you only want to have faster input forms, you'll only need CF.LIB and B0.LIB. If you only want to have faster choose boxes, you'll only need CF.LIB and B3.LIB. The presence of an UFL.LIB is optional in most cases, however at least UFL FNT1 is mandatory when using UI.LIB . But with UFL.LIB you'll gain somewhat lower case char display support for the small font in all SpeedUI components (including the new SpeedBrowser compact)

RAM usage:

Each SpeedUI display oriented component caches the currently chosen UFL font for faster access while running. The SpeedUI stack environment caches up to three different UFL fonts while running. so if the UFL library is in an uncovered port (0 or 1), memory usage for the fonts will be 15 bytes. If UFL is in a covered port, temp storage for the fonts will be between 2KB and 5KB (UI.LIB) . If UFL is not installed, most SpeedUI components (except UI.LIB) will use the internal fonts without caching. So if you're low on memory, but still want to use certain SpeedUI components (except UI.LIB), you could omit UFL.LIB .

The temporary memory needs for most of the SpeedUI components is nearly negligible (a few 100 bytes), except for the fonts, as described above, and for the cache for pretty-print expressions, which may take several KB depending on the complexity and amount of cached items.

The preliminary SysRPL display support using JAZZ 6.8+ will need _much_ temporary memory.

Variables in the hidden directory:

Old: Up to SpeedUI 12.03:

_Warp	used by CF
_na	used by UI
_clp	used by UI. Holds the clipboard item list.
_ra	used by UI
_nb	used by UI. Indicates whether NoShift direct BINTs entry is enabled or not.
_clpMode	used by UI. Holds the selected clipboard mode.
_clpMark	used by UI. Holds the current clipmark position.
_dly	not used so far. Holds the keyboard longhold delay.
_emm	used by UI. Holds the current error message display mode.
_rc	holds the start col/row for my MW replacement.

Most of these vars contain a bint, which is either #1 (enabled) or #0 (disabled). If one or more of these vars don't exist, the corresponding feature is assumed to be disabled.

New: From SpeedUI 12.04 on:

SUIpar	Holds all settings except clipboard data
_clp	used by UI. Holds the clipboard item list.

SUIpar is a string of 24 data bytes, which holds many SpeedUI related settings. These are in detail:

SUIpar holds the following settings:

<u>Pos</u>	<u>Length</u>	<u>Description</u>
1	1	Warp NoShift_Alpha_Lock RightShift_Alpha_Lock NoShift Bint_Entry
2	1	ErrMsgMode ClipboardMode (each entry uses 2 bits)
3	1	Internal Internal fSymMatrix fCommas
4	1	fSmallAGrob fED fEDIT fSRPL
5	1	fAGrobOK fNoOvalCST fOval fStk6
6	1	HeaderLines in pixels rows
7	1	FontNumber 1 - 4 . Uses up to 3 bits of the nib
8	1	FontHeight
9	1	MaxDispRow Max stk levels that can currently be displayed in curr. font
10	1	DispRowPart This is zero if there are no remaining pixel rows,
11	1	UITempFlags As name implies, this is a temporary storage
12	1	SUI_LegacyF Holds up to 4 legacy SpeedUI flags. Currently: fSmlFnt fSlider
13-17	5	Delay1
18-22	5	Delay2
23-27	5	Delay2
28-32	5	Delay4 Press/and/Hold Delay (not implemented, but reserved space)
33-37	5	Matrix row
38-42	5	Matrix column
43-47	5	Clipboard mark position

Nib 1 holds 4 individual status bits, nib2 holds two 4-state values, nibs 3,4,5 hold shadow copies of SpeedUI user flags, nibs 6 - 10 hold single-digit display metric values, nib 11 holds a temporary backup of certain SpeedUI settings, which is compared to the current value after certain stack manipulation operations, nib 12 holds up to 4 individual status bits. Nibs 13 - 27 hold the delays for the configurable key repeater, nibs 28 - 32 hold the longhold delay for certain keys (not implemented yet), nibs 33 - 42 hold the initial matrix coordinates, and nibs 43 - 47 hold the clipboard mark position within the current edit line. Nibs 3 - 12 are updated only on normal exit of the SpeedUI stack replacement. If for any reason the stack replacement doesn't exit normally, the RAM-based nibs 3 - 12 will NOT be updated. In a later version of SpeedUI, the enhanced stack interface could also read the settings of nibs 3 - 12 from SUIpar on startup. However currently nibs 3 - 12 are only shadow copies of the respective user flags and status RAM locations. Nibs 3 - 12 are not read from within the stack interface so far.

Features overview of the various libraries:

A1,A2,A3

These libraries are no longer supported! Please see below how to uninstall these libraries.

AB

The AnimationBrowser.

B0

fast replacement for the built-in form engine. Enabled via WARP toggle switch in config. As a nice feature hot key searches in long lists can now be interrupted, which wasn't possible in the orig. version. BE SURE TO STORE B0.LIB IN EITHER PORT 0 OR 1.

B1

fast replacement for the built-in MsgBox drawing mechanics. The somewhat faster User input form preprocessor is included, too. In one of the next versions, it may be expanded to include extended features like check fields and choose fields, if there's enough positive feedback.

B2

Very fast replacement for the built-in charset browser, with char echo capability, of course. Also shows keyboard shortcuts for direct char entry. Enabled via WARP toggle switch in config. This library catches the Last Command key combination, and simply adds a title bar to the Choose box.

B3

fast replacement for the built-in Choose box engine. With switchable font size and Sliders.

CF

configures and enables/disables all replacement libs. CF contains the ML decompiler and acts as base library for functions shared with other SpeedUI components. The SpeedUI BOOT config menu can be accessed when holding the backspace key during warmstart. Full SpeedUI configuration can be accessed via SETUPD in UF.LIB (requires an UFL.LIB containing FNT1) .

CF_lite

configures and enables/disables all replacement libs. CF_lite lacks the ML decompiler and acts as base library for functions shared with other SpeedUI components. The SpeedUI BOOT config menu can be accessed when holding the backspace key during warmstart. Full SpeedUI configuration can be accessed via SETUPD in UF.LIB (requires an UFL.LIB containing FNT1) .

CMD

implements the command catalog and USAG functionality.

DM

implements the lower case menus. Before SpeedUI 15.01 , this feature was included into CF.LIB . From SpeedUI 15.01 on, this feature is a separate plug-in library.

EA

Very fast MatrixWriter replacement. Fully compatible to the built-in MatrixWriter, but much faster display and key response. New features so far: Up to six visible columns (the orig. MW offered only 5) Up to seven visible rows (the orig. MW offered only 5) Font switching. You can switch between the normal (medium) font, and the small 3x5 Font, even while you're in the MW. The MW starts with the font (medium or small) which has been selected in the SETUP of the Config library (CF.LIB) . Border wrapping. This means if you're at the very top, left, bottom, right border of a matrix, you can go one step beyond the border, and you're at the opposite border. Also works with repeating keys. BE SURE TO STORE EA.LIB IN EITHER PORT 0 OR 1.

E4

Multi Equation Solver (MES) replacement. Fully compatible to the built-in MES, but slightly faster. This replacement circumvents some functional restrictions of the orig. version. SpeedUI users who also use MES should use this replacement.

MB

MessageBrowser. Sample application for the SpeedBrowser. Shows all HP-48G built-in messages in categories. Works with the built-in browser, too.

SBC

The SpeedBrowser Compact edition. See separate doc for details.

SBUC

The SpeedBrowser Ultra Compact edition. See separate doc for details.

FC

The very fast FlagBrowser shows descriptions for system or user flags. If FC.LIB and B3.LIB are installed, the MODES input form will call FC when pressing the [FLAGS] menu key. FC.LIB also has the keyword FC to be called from the cmd line or in programs.

IB

The fast interactive Input Form Builder for the HP-48 See separate doc for details.

LB

The list browser. Like James Donnelly's list browser, but faster and more compact.

UF

User frontend for various SpeedUI settings. Contains user keyword SETUPD for the SpeedUI config menu. Requires UFL.LIB Level F.

UFL1

Universal Font Library level 1 containing FNT1

UFL3

Universal Font Library level 3 containing FNT1 and FNT3

UFLF

Universal Font Library level F containing 4 different fonts (FNT1,FNT2,FNT3,FNT4)

UI

The SpeedUI stack replacement component. Greatly enhances and accelerates every aspect of the stack interface. Shows amount of free memory and port status. If a multi-bank card is placed in port 2 it will show the amount of banks in that card. Includes switchable 4,5,6,7-level interactive stack, with switchable header line (0,1, or 2). Now the input line has up to 9 visible rows. This replacement has the fastest possible status display, and the fastest command line and multiline editor. The editor now features a second menu page, with COPY/PASTE, clipboard browser, and online command catalog with parameter help. Also includes a Quick Start Menu for certain applications and some UI mangement keyboard shortcuts. See separate doc for details.

UI_lite

The SpeedUI stack replacement component. Greatly enhances and accelerates every aspect of the stack interface. Like UI, but lacks Pretty Print and some other features.

Obsolete key libraries (A1,A2,A3):

The replacement keyboard libraries are OBSOLETE since SpeedUI 8.xx, and are no longer supported!

If you have one or more of the keyboard libraries installed, you'll need to uninstall them to avoid any interferences with the new UI library.

Here's a method to safely uninstall them. To uninstall A1, A2, A3 do the following:

You'll need to have the latest version of CFLIB installed.

Then create a list in the HOME directory: { 161 162 163 } 'NoCfg' STO . Do a warmstart.

Use the 2048-Trick described in StartHere.pdf for deleting libs 161, 162 and 163.

After having removed all replacement keyboard libraries, please perform a warmstart.

The new Library Cat:

There is a minor 'restriction' when using the new library cat. Unlike the built-in version, the replacement doesn't check whether a library has been removed while displaying it's top level menu. So if you remove a library while displaying it's library cat menu entry, and then press the appropriate soft key, you'll simply get an empty menu. If you're in a specific libraries main menu, and then remove that library without changing the displayed menu, the menu will not disappear, but pressing one of the menu keys for that library will simply generate a 'Library closed' message. However the things mentioned above are just display limitations, there are no functional restrictions here. And you'll get the fastest library menu ever seen on an HP-48. The library cat can be updated by simply pressing rs+[LIBRARY] again.

MSGBOX & CHOOSE libraries:

B1.LIB and B3.LIB should both be installed, or more exactly, B1.LIB could be installed without B3.LIB , but if you want to install B3.LIB (which is highly recommended), you should also install B1.LIB . This is because the MSGBOX command relies on the predefined frame grobs in B3.LIB, most of which I omitted. However, it'll work with B3.LIB replaced only, but MSGBOX applications with more than three lines of text will write some text over the frame border, which doesn't look too good.

Multi Equation Solver (MES):

The G Series built-in MES is a powerful tool for solving sets of equations. However the built-in version has some limitations when used from an alternative stack environment, like SpeedUI or Java. In an error case, or when the solving process is interrupted via the ON/Cancel key, the user would expect a suitable message ("Interrupted"). This is not the case when using a stack replacement. When the above happens, MES aborts SpeedUI, leaving some trash on the stack. Similar with Java, MES aborts Java, and doesn't allow starting Java again. Therefore I made a standalone version of the MES, which can be stored in any port, and which solves the above problem. So if a user plans to use the MES, it is strongly recommended to install my replacement version.

SpeedUI header line options:

SpeedUI allows to choose between zero, one, or two header lines. Applications which rely on the classic header of two character rows height, may have problems displaying their messages when the SpeedUI setting is one or zero lines.

If you use applications which rely on status area messages, like the MES or other solvers, it is recommended to set HeaderLines to 2.

Credits:

The HP Corvallis Team for their ingenious arts in general, and for the HP-48 in particular

Will Laughlin for the extraordinary ML decompiler

Richard Steventon for the GROB Scroller & ?Dispmenu

JAVA Team for the AGROB Algebraic-to-GROB converter

Mika Heiskanen for the VV viewer

Heiko Arnemann for intensive beta-testing

Robert Ryan for bug reports and testing, and convincing me to include the 'Pretty Print' feature AGROB offers

Joe Donnini for bug reports and SpeedUI support

Javier Goizueta for bug reports

Rudy P for bug report

Timo Ruprecht for bug reports

and some other people who sent me suggestions, bug reports. Thank you all!

[\[Back to Top\]](#)

Any questions or suggestions? Feel free to contact me. Have fun.

Raymond Del Tondo (Magic48ges@gmx.de)