

# CodeScape 2.1.1. Beta, build 92

## Known Problems

### Build 91

The Profiler does not work correctly. The fix added to the Profiler to support multi-threaded applications did not fix the problem. A dialog box appears, informing the user about an error in matching the return with the calling address.

## New Features

### Exception reporting (build 91 - not yet documented in the help)

*Exception Reports* is a new menu option on the Debug menu and on the Target window shortcut menu. Select this option to toggle exception reporting on or off. If this option is on and an exception occurs, a dialog appears and tells you the exception category, type, and where it occurred.

### Call Stack step out to any level (build 91 - not yet documented in the help)

In the Call Stack region, use Run to Cursor to return to a specific function outside of the current one.

### Simulator as of build 87 - alpha feature

#### Bug Fixed:

1. On closing the simulator several incorrect register lock error messages can be produced.

#### Known Bugs:

1. The following instructions are not implemented:  
SLEEP  
LDTLB  
MOVCA.L  
OCBI  
OCBP  
OCBWB
2. The following instruction is only partially implemented:  
PREF @Rn - Prefetches OK, Will not save store queues
3. The processor store queues are not modeled. PREF write feature not implemented.
4. Cache operation is not completely tested:
  - a). Caches are not initialized to the real processor status on simulator startup. The caches are purged on simulator startup. The CCR is read from the CPU on startup.
  - b). Changes to CCR register only noticed on a 32bit write.
5. MMU not modeled
6. Tracing (Stepping) whilst viewing the simulator output can be confusing because during a dual issue the first of the two instructions dual issued can hit the breakpoint the simulator currently stop immediately but the partial operation of a slot is not reported until the whole slot completes.
7. UI is the same as the SH2 & will not permit color / font changes
8. None of the reported SH4 processor bugs are simulated.

### New editor feature (build 68)

The editor now has rudimentary IME support. This is not perfect, but at least allows Japanese to be entered.

### **Editor features (build 67)**

Added a status bar showing current line & column  
Added syntax highlighting  
Added a toolbar  
Added bookmarks  
Added a recent file list (this list is per editor and NOT global to all editors)  
Added an option to configure the syntax highlighting  
Fixed the Shift+F10 problem  
Fixed the cancel session close problem

### **Setup Editor Dialog (build 67)**

Added %f place holder for filename to be loaded into editor  
Fixed problem with corrupted names  
Fixed problem with duplicate names  
Grayed delete button when no editors to delete  
Grayed add button when maximum number of editors entered

### **Shortcuts (build 67)**

Added shortcuts for creation of windows (Alt 1-9)  
Added shortcuts for hard and soft reset (Alt/Ctrl F2)

## **Fixes**

### **The Target window Right Click Menu options do not work as expected (build 92)**

Fixed.

### **Toggle breakpoint does not work in memory region (build 91)**

OnCmdUI update changed in base class & stopped the memory region working. Added local version to memory region.

### **Locking of Disassembly region (build 91)**

Remove 3 lines of code that setup the lock flag before the dialog was initiated. The values were set again if the user pressed ok.

### **Disappearing breakpoints in disassembly region (build 91)**

The recalculation of the breakpoint position in the buffer was only done on a refresh\_info or greater state (i.e. data got from the target), but they could move on a refresh\_none (e.g. buffer origin moved within the buffer). Fix was to call the breakpoint refresh function on every refresh.

### **Cursor in disassembly region (build 91)**

Added support for cursor movement to top when window locked.

### **Call Stack step out to any level (build 91)**

Now can use run to cursor in call stack region that works like the step out except instead of going to the immediate caller it will go the level indicated by the call stack region's cursor.

### **Dialog pops up informing user of exception / interrupt (build 91)**

Added menu option to debug and target window menus to enable/disable reporting and if enabled and an exception occurs a dialog pops up informing the user of the exception.

**Symbol Completion Dialog (build 90)**

Filenames now don't appear in the symbol list.

**New Options on Build Menu (build 90)**

Added two options to the Build and Target Windows. Options are Hide and Allow Docking

**Project make options disappear when makefile field changed (build 89)**

Filled in remaining fields from those already present in dialog.

**Removal of breakpoints on load program file (build 89)**

Checks for newer or different program file and if any breakpoints present prompts to remove them. Note the prompt comes before the program file is loaded since if run to main etc. is set it is possible to hit an old breakpoint before main is reached.

**Add a cancel option to shutdown (build 89)**

Added a cancel button to save session dialog. Note did not add a plain shutdown query, if nothing has changed nothing will be lost, seems standard behavior in Windows apps.

**Set PC to Cursor does not work in the memory window (build 89)**

A related problem occurred in other windows where set cursor/pc to pc/cursor options were available but should have been disabled, set pc to cursor is now only active in source/disassembly windows, and set cursor to pc only available in source/disassembly/memory regions.

**If the make in the spawned process fails, there is no way for the user to shutdown CodeScape (build 89)**

The failure occurs (95/98 not NT) because of a duff setting in the environment variable CMDLINE, before launching the build CodeScape checks the environment and throws away any spurious CMDLINE entries.

**Re-draw of background whilst loading a large elf (build 89)**

Only happens in Japanese Win95. Changes the pump message loops  
Made no difference to US builds but the Japanese version now re-draws eventually.

**Focus not shown on Region Titlebar (build89)**

Ensured focus gets set when frame is activated.

**Corrupt source file list by loading a p-file for the second time (build 89)**

Was not handling cancellation properly. There are two phases to loading: Debug Info and Binary Data. It now can tell which phase has been cancelled. If the debug stuff was interrupted it will clean up any half-processed info and remember that it had tried to do the debug but got cancelled. Then when you restart, it knows to try the debug info again. Once the debug info has been successfully and completely processed, future restarts will not process the debug stuff. If a cancel happens during the binary load part, any debug info remains intact and only binary data will be reloaded on subsequent restarts.

**No One shot after lock to address (build89)**

Added a one shot to the case where the lock becomes active.

**CodeScape Build86 with ASMSH (build89)**

Fix in Salvador, The macro was being tagged as a valid source address.

**Cannot open file in dos box (build 89)**

This was due to one of several places where a program file was (re)loaded. These were not taking into account the section klagging and run from copy flags. They now do so.

This should also affect reloads when you power cycle the target as well as restart

**Memory region vertical toolbar on startup (in a very small window) does not draw the down arrow (build89)**

Ensured size of scroll bar recalculated after initially configuring the view

**Docking Project & Target Windows (build 87)**

The project & Target windows now dock / size correctly.

**Locking Program File (build 87)**

Option now exists on load program file dialog to load the program (elf / coff) file without locking external access. This is slower than the default action when loading the file, once loaded performance is as before.

**Exceptions Reporting (build 87)**

- 1) Fix to exception reporting from target. Build 86 fix for floating point enables bit, broke the reporting of other exceptions.
- 2) Stub version 2.8.0a had a bug that effected FPU exception reporting
- 3) More descriptive exception reporting including FPU sub-type.

**Exception response code modified (build 87)**

This result in windows not being drawn during intermediate states. Effect are speedup in:

- 1) Tracing (especially source level ).
- 2) Response to incorrect FPU exceptions.
- 3) Conditional breakpoints.

**Caching of expanded expressions in watch / local (build 87)**

The state of upto 64 expanded symbols is retained. Therefore if you re-enter a function that had an expanded symbol, the symbol will appear in the state it was left in.

**Status Information On Title (build 87)**

The status information for the active target / processor is replicated on the CodeScape title bar.

**Lost Accelerator Keys (build 87)**

Lost accelerator keys found. The problem resulted from the ON\_UPDATE\_COMMAND\_UI messages not being sent to sub-menus for accelerators. This is reported in Microsoft known problems Q120598. The fix was to add your own OnInitMenuPopup() function overriding the default behavior.

**Elf Load Speed (build 86)**

The Initialisation of Lines on very large elf's could cause the cache to thrash. This produced the effect of the load stopping. Speed improved.

**Floating Point Exceptions (build 86)**

Workaround for the SH4 FPU exception 'characteristic'. The SH4's floating point exception handling requires software assistance when either the V, O, U, or I bits in the FPSCR.enable field are enabled. An FPU exception will be raised regardless of whether an exception has occurs, for a large number of floating point op-codes (e.g. fadd, fsub, fmul etc.).

This means that once any of these bits are set the CPU can stop with an exception on a floating point instruction such as 'fmul r4, r5' when the values in r4 & r5 are 1.5 & 1.5.

CodeScape (Build 86) plus DA debug stub (version 2.8.0a) has a workaround to solve the problem. The work around operates as follows:

- 1) **The floating point enable bits are set.**
- 2) **A floating point exception occurs when the op-code is encountered.**
- 3) **User Exception Handler () - Optional**  
This must pass the FPU exception on to the stub using the pass-back method  
To implement the exception 'passback sequence' the following code must be issued from within the users exception handler  
BRK  
RTE  
RTE  
NOP  
This code is executed for exceptions that the user does not want to handle implicitly. Note: The EXPEVT register must not be changed prior to calling the 'passback sequence'
- 4) **The Debug Stub -Default Exception Handler**  
The debug stub now handles this exception by analysing the FPU exception to determine whether the instruction caused a valid exception or not. If a valid exception had occurred then the code execution is suspended and an appropriate message is displayed via CodeScape. However, if a unwanted exception was detected then the debug stub will perform the following:
  - a) The FPSCR register is saved.
  - b) The enable flags in the FPSCR are cleared
  - c) The FP instruction that caused the exception is re-executed
  - d) The FPSCR is restored to its saved value.
  - e) The interrupted code is resumed.In this way the problem is masked from the user and the code runs as if no exception has taken place.  
  
NOTE:  
The debug stub can not handle FP instructions executed in slots. Currently, unwanted exceptions resulting from an FP instruction in a slot will cause the exception to be thrown to CodeScape.
- 5) **CodeScape FPU Exception Handler**  
CodeScape receives the more complex FPU exception (the real one + the slotted one). If a real one is encountered CodeScape will stop & display the exception on the status line. If an unwanted slotted exception occurs CodeScape will trace around the exception & begin the program running again.

NOTE:

If an unwanted exception is handled by the debug stub a large loss of performance will be incurred (~several hundred clocks). If the unwanted exception is passed all the way out to CodeScape the loss of performance is huge. ***It is recommended that the floating point enable bits are not used when the game performance is required.***

### **Debugging mode (build 86)**

The selection of the debugging mode (OS v CPU) only happens once per target the first time CodeScape runs after a re-flash. To change the mode between re-flash use DA-check to switch the mode.

### **Tooltips in watch, locals & call stack windows (build 86)**

Expands to show the full text of a cell that is only partially visible.

**Fix to disassembler to calculate SEA & DEA (build 86)**

The disassembler incorrectly calculated the SEA & DEA on certain mov & fmov op-codes.

**Unstep (build 86)**

The unstep mechanism has been improved + the fix for the disassembly of SEA & DEA also improved this feature.

**Register window shows SEA & DEA for 'fmov' (build 86)**

The register window now shows the SEA & DEA values for 'fmov' op-codes.

**Profiler Break Points (build 84)**

Added Profiler Break Points. These are used to Start / Stop Trace Profiling.

**Elf load problem (build 84)**

Main.elf now loads correctly.

**Reflashing multiple targets (build 84)**

Now does not halt.

Session load problems (build 84)

Profile added unsupported data to session files, additional checking & removal.

**Profiler Break Points (build 82)**

Added Profiler 'Interrupt Profile Trace Filter'. This prevents the Profiling of subroutines of an Interrupt / Exception routine.

**Go to next error (build 81)**

Fixed problems due to relative path names.

**Source Window "List files in Program File" (build 81)**

A switch now available to show only files that contain valid source.

**GPF on Session load (build 81)**

Resulted in profiler view identifier changing between build. More rigorous testing performed on identifiers.

**GPF in Source Window (build 81)**

fixed.

**Project build (build 81)**

Now uses a pipe to show output as build proceeds.

**Fast keys in several dialogs (build 81)**

Changed to remove duplicates.

**Warning on startup when salsa fails to initialize (build 80)**

The reason SALSA failed to initialize is now displayed as a message box. This can happen more often in the case of an old version of ASPI (version 1.0).

**Browse Button added to Editor Setup (build 80)**

Browse button added to editor setup (user request).

**Mouse Position Selection In Bit Region (build 80)**

Cursor position can now be moved with the mouse on to members of the bit field in the register region.

**GPF when switch between source in the same region (build 80)**

Fixed dangling pointer

**Source Files Remained Open (build 80)**

Source files could be left open (stopping external editors from opening / saving). This happened when the source region was hidden or zero sized (no rows visible).

**Memory region overlapping (build 78)**

Sometimes it says ""Internal Error - Memory Validity ranges overlap"" and be hung up - altered to do the scan of the memory definitions on startup & to report any error with more concise error message

**Register Region Inc/Dec Value (build 78)**

Inc/dec value can now accept floating point value and Hexadecimal or symbols

**Static Member Variables (build 78)**

Static member variables (GNU - ELF/DWARF) now evaluate correctly.

**CTRL+Right Mouse (build 78)**

The CTRL+Right mouse problems from build 77 have been fixed.

**Project Build Window (build 78)**

Width & height of the undocked Project Build Window is now stored into the registry .

**Processor Status Information (build 78)**

The processor status information now says when a processor is disabled

**Setting Up Command line for External Editor (build 78)**

The command line now supports %f+%l for the editor startup

**Interrupt Pass-back Implemented (build 77)**

Interrupt pass-back implemented to supplement Exception pass back (See firmware history.txt.)

**Load P-File (build 77)**

Load program file substantially speeded up.

**Goto Next Error after Make (build 77)**

F4 cycles through errors resulting from a make.

**Start Address of Memory Region (build 77)**

On re-loading a session start address of memory windows could be incorrectly loaded.

**Breakpoints On Shutdown (build 77)**

When CodeScape removed HBC & UBC breakpoints on shutdown it stopped target if it was running. This bug has now been fixed.

**DALI.CFG changed (build 77)**

The area of memory used by CodeScape for the debug stub has now been removed from the .cfg file. This is to prevent accidental corruption.

**Editor IME Support (build 75)**

Improved Cursor Movement.

Drag and drop Fixed

Caret halfway through a multi-byte character fixed.

*Known Limitations*

- 1) There is a small glitch when new characters are being typed (half a character sometimes disappears but it comes back when enter is hit).
- 2) Overwrite mode does not work - none of the Japanese applications tested appears to work.
- 3) Select work isn't "intelligent" but this could be lots of work and Notepad doesn't bother. (Word "knows" Japanese, or uses IME features, to allow words to be selected with double click).

**Color / Font (build 75)**

Default colour / font information now stored in registry (removed from session). Window specific information still stored in session.

**Registry (build 75)**

Defaults are now stored in registry & are version specific.

**DALI.CFG (build 75)**

The stub area of memory is not included in the user memory space. Aimed at preventing user from overwriting stub area.

**Editor IME Support (build 73)**

Candidate window now appears to allow selection of words.

Problems still exist in drag and drop, it is also possible to find the caret halfway through a multi-byte character.

**Range on dali.cfg (build 73)**

Specified range on dali.cfg is 1 byte shortened. With default DALI.CFG, 0xcfffffff is not assessable

**Watch Re-draw (build 73)**

Watch window did not update immediately after bitfield value edited if window updates off.

**Removal of breakpoints after re-build (build 73)**

When finishing the reloading of ELF after it is rebuild via CodeScape's project making, dialog inquires about removing All Breakpoints

**Removal of breakpoints On Session Load (build 73)**

If during reloading of program file as part of a session, CodeScape detects that the time stamp within the session file is different to that of the program file being loaded, a dialog inquires about removing All Breakpoints.

**Build Fails Under NT (build 73)**

NMAKE fails under NT

**Line Truncation in Disassembler Region (build 73)**

The lines could be truncated in assembly region if large labels are used, column now auto formats.

**Errors During Make (build 73)**

When double clicking on an error, CodeScape failed to inform the editor if network path required.

**Warnings During Make (build 73)**

CodeScape treated warnings with the same severity as error, i.e. it did not load the program file.

**Source Region Line Numbers (build 73)**

In source region, line numbers only appear for executable lines. This means that comment lines show no line numbers. All lines now show numbers.

**Load Program File (build 73)**

Option now on load program file dialog to optimize load times, by joining together adjacent binary block.

**Button Label Change (build 73)**

In editor Setup dialog "Delete" option renamed "Remove"

**DA Rev A Fix (build 71)**

Build 68 did not except rev A DA boards .

**Editor features (build 71)**

The editor's IME support has been improved extensively.

**Project Build (build 71)**

'Any (\*.\*)' added to option on browse button.

**Toolbar & menu status updates (build 70)**

Toolbar & menu status updates modified so that it reflects the status of the window with the focus.

**Breakpoints (build 68)**

A small bug was found and corrected in the breakpoint mechanism. As a result, tracing has been speeded up.