Code Coverage on Nohau's HCS12 and HC12 Full-Featured Emulator

By: Doron Fael – Nohau Corporation

September 1, 2004

This document briefly shows how code coverage is achieved using the Nohau full-featured emulator for the Motorola/Freescale HCS12 and HC12 family.

Code-Coverage is the emulator's ability to analyze which source lines and code lines are executed, which lines are not executed and which lines are partially executed. Code coverage is needed in applications that require very high reliability, and need to be verified so that every source line has been successfully exercised at least once, in order to comply with international standards of software integrity and reliability.

Code-coverage information is collected by the trace-unit of the full-emulator. The trace monitors the bus activity and decodes the instruction pipeline of the target HCS12 or HC12 processor as it executes the application under test. The trace analyzes which instructions are executed and which are not in run-time and writes this information to a dedicated memory in the trace in run-time. The analysis is aware of the HCS12 PPAGE register and can accurately analyze both non-paged and paged application up to 1Mbyte in size. (On the S12X Family it can analyze applications up to 8Mbyte in size)

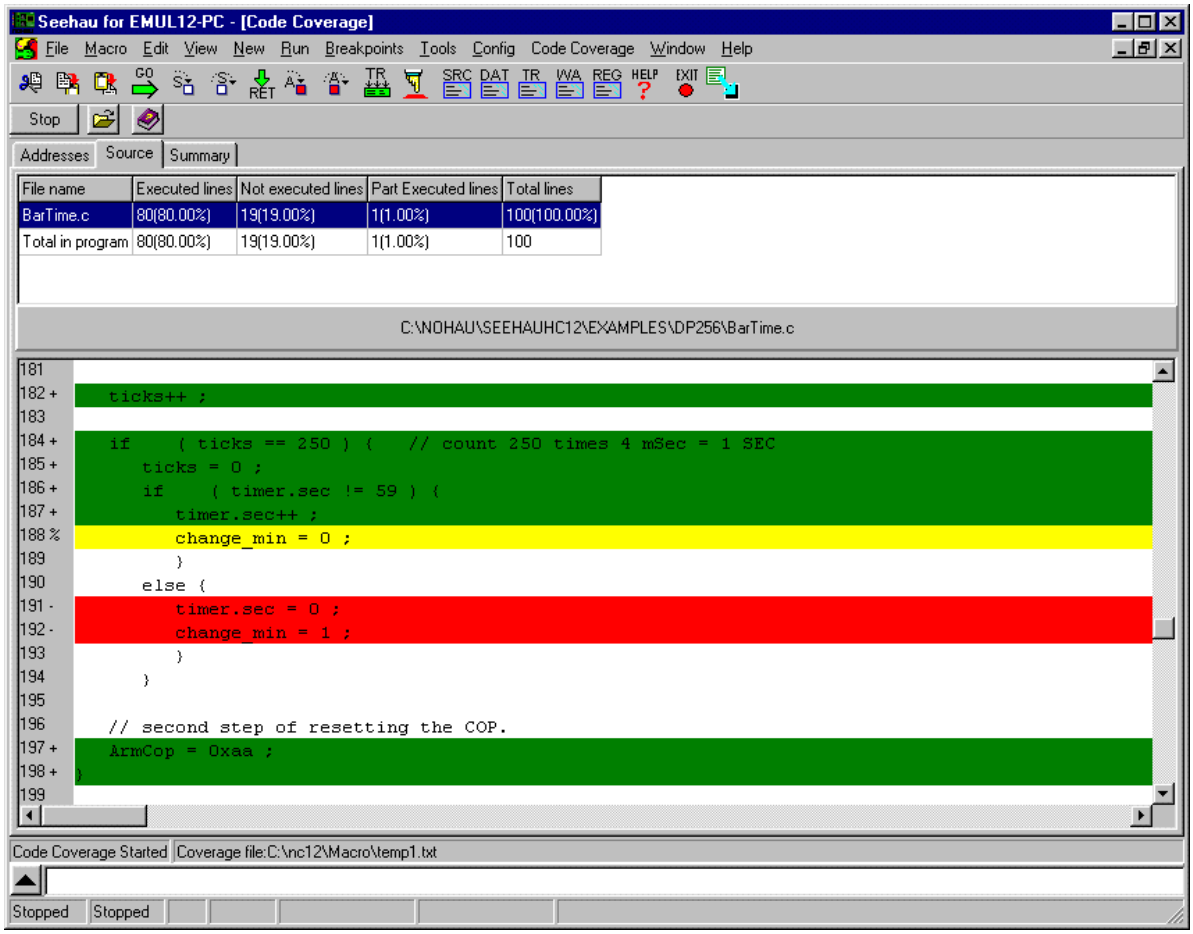
When the collection of the code-coverage information is completed the user can stop the code coverage. The code-coverage information is then automatically read from the trace memory to the PC, and the user is asked to input a file-name to save the results of this iteration of code-coverage information.

The code coverage information is then displayed to the user as shown in the screen-shots in the following pages:
- Green source lines have been fully executed
- Red source lines have not been executed at all
- Yellow source lines have been partially executed
- White source lines did not produce any assembly code and therefore are not analyzed.

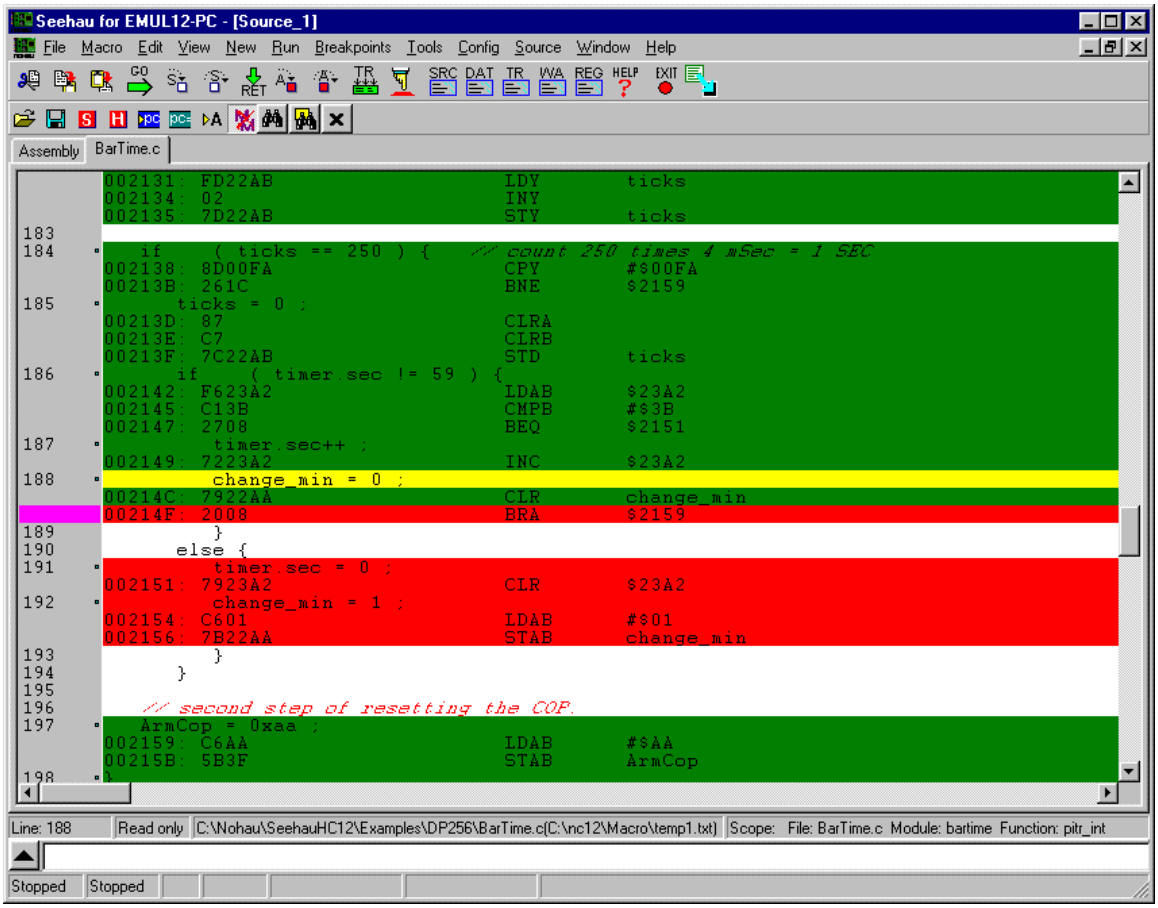
The Seehau software also allows Code-Coverage information from several iterations to be merged together, to create a complete code-coverage report of many code-coverage iterations.

In the following screen shot we can see fully executed source lines (Green), non-executed source lines (Red), partially executed source lines (Yellow), and source line with no assembly code (White).

As can be seen below, the portion of code that should take care of a change in the minutes has not been executed in this case, since the code-coverage was ran only for a few seconds before a transition of the minutes took place. If the user would run the code-coverage for a longer period, we would see these red (unexecuted) lines becoming green (executed).



The following screen-shot displays the code-coverage results in the source-window in mixed-mode display where both the C source lines and the assembly instructions are intermixed together.
The yellow source line “change_min = 0 ;” is partially executed, as only the first of the two assembly instructions that create this source line has been executed, and the second assembly instruction was not executed (in the particular case shown here because of a breakpoint).



The Code-Coverage information can also be displayed in raw mode as seen below listing the address ranges of the instruction that have been executed and supplying a summary as seen below. This information is also available in a text file, to allow further processing outside the Seehau software if needed (further processing can also be achieved using Seehau's built in macro language).