MICROSAR Safe Silence Verifier

Technical Reference

Version 1.4

Authors	Markus Groß, Patrick Markl
Status	Released

Document Information

History

Author	Date	Version	Remarks
Markus Groß	2012-07-24	1.0	Initial version
Patrick Markl	2012-08-22	1.1	Changes after review
Markus Groß	2012-11-15	1.2	Add information about third party libraries
Markus Groß	2013-01-15	1.3	Update to reflect changes
Patrick Markl	2014-03-03	1.4	Added restrictions chapter

Reference Documents

No.	Source	Title	Version
[1]	ISO	ISO/IEC 9899:1990, Programming languages -C	Second edition

Contents

Tables

1 Introduction

MICROSAR Safe Silence Verifier (MSSV) is a command line tool delivered as part of Silent BSW packages. MSSV checks based on rules the consistency of generated configuration files of the BSW modules. The result is written to a HTML report. The report is part of the proof that the BSW modules fulfill the Freedom from Interference criteria.

Reference
For all required steps to be performed as part of the Silent BSW integration see the project specific Safety Manual.

1.1 Intended audience

This document is relevant for developers who integrate Silent BSW into their ECU.

2 Functional Description

This chapter describes the tool MSSV and how it can be used to check the consistency of the generated configuration data for SilentBSW modules. MSSV supports configuration via command line parameters. These are described in the following section together with the report which is the output of MSSV.

2.1 Required Environment

MSSV supports the following operating systems:

- Windows XP SP3 (32Bit)
- Windows 7 (32Bit)
- Windows 7 (64Bit)

2.2 Restrictions

MSSV uses a Compiler front end in order to compile the input source files. This Compiler front end requires ANSI-C 90 [1] conform source code. Some target compilers implement specific language extensions which might prevent MSSV from compiling the code successfully. The Vector BSW code does not contain such language extensions. However, these extensions may be included via customer header files. In such a case the customer shall take care that these language extensions are encapsulated via the preprocessor for the MSSV execution. The corresponding preprocessor switches can be specified via the command line when calling MSSV.

2.3 Command Line Parameters

MSSV is configured by means of command line parameters. This chapter describes the available command line parameters and their meaning.

Command Line Parameter	Description
Optional Parameters	
`-h, --help`	Display available options.
`--version`	Prints the version and exits.
`-v, --verbose`	Enables verbose output of MSSV.
`--crcCheck`	Only perform CRC32 checks and then exit.
`--openReport`	Open the report file when finished.
`--stats`	Display timing statistics when MSSV is finished.
`-l, --logFile <file>`	Specifies the filename of a logfile. If this parameter is missing MSSV writes the log to stdout.
`-r, --reportFile <file>`	Specifies the filename of the report. If the parameter is just a path the report is written with the default report file name. Otherwise the specified report file name is used.

Command Line Parameter	Description
Optional Parameters	
	If this parameter is missing MSSV writes the report to the current working directory.
`-p, --pluginDir <directory>`	Specifies the path of the plugin directory. By default this is the subfolder “plugins” in the directory of MSSV.
`-D, --define <symbol>`	Additional defines for the compiler.
Required Parameters	
`-i, --inputDir <directory>`	One or more input directories.

Table 2-1 Command Line Parameters

2.3.1 Option -h, --help

This command line option prints the help of MSSV on the console. The help lists all command line parameters as displayed in Table 2-1.

2.3.2 Option --version

This parameter displays the version of MSSV on the command line.

2.3.3 Option -v, --verbose

The verbose option enables a verbose output of messages from MSSV. As a default the verbose mode is not active. This means that MSSV only displays warnings, errors and fatal errors as well as some selected note messages which might be of interest for the user. If the verbose mode is enabled MSSV will display all note messages.

2.3.4 Option --crcCheck

The CRC check option can be used to check the CRC32 checksums of the plugins. Then the report file contains a report about all plugins and their checksums. If the report is green all plugins are valid. If the report is red at least one plugin is invalid.

2.3.5 Option --openReport

This command line parameter instructs MSSV to open the resulting report HTML file after it has finished. Please make sure that HTML files are opened with a suitable program by default when using this option (e.g. if you double click an HTML file in the explorer it should open a web browser).

2.3.6 Option `--stats`

The statistics option is not particularly interesting. Passing this option to MSSV will display some timing statistics. These statistics include how much time was spent on each plugin and to parse a file etc.

2.3.7 Option -l, --logFile

By default MSSV displays all messages on the standard output (stdout). Using this option MSSV will redirect its output to a file instead. Please note that the file which is specified using this option should be writable by MSSV and be located inside a directory which is accessible for MSSV.

The resulting log file will have the file extension “.log” regardless what extension the file had which was specified as the command line parameter.

2.3.8 Option -r, --reportFile

MSSV writes a report file at the end of its analysis. This report file is named “report.html” by default and will be stored inside the working directory from which MSSV was called.

Using this command line option it is possible to specify a custom filename and location for the report file. Please note that the file which is specified using this option should be writable by MSSV and be located inside a directory which is accessible for MSSV.

The resulting report file will have the file extension “.html” regardless what extension the file had which was specified as the command line parameter.

2.3.9 Option -p, --pluginDir

This option enables the user to switch the plugin directory MSSV is using to perform the consistency checks. This is useful if multiple plugin directories exist.

2.3.10 Option -D, --define

Internally MSSV uses a compiler frontend to parse the generated source code and preprocess it. With this option it is possible to define symbols for the preprocessor without adding them to a header file. This parameter is similar to the “-D” parameter known from compilers such as GCC or Clang.

This command line option can be specified multiple times to define an arbitrary amount of symbols.

2.3.11 Option –i, --inputDir

MSSV requires one or more input directories to operate correctly. These input directories are scanned for source code files, which are processed by the plugins.

To specify multiple input directories the parameter can be passed multiple times with different values. The input directories are not scanned recursively by default. However, if a directory ends with “`\*`” MSSV scans this input directory recursively.

2.3.12 Command Line Usage

2.3.12.1 Option Only Parameters

Option only command line parameters can be passed to MSSV and do not require a value. Option only command line parameters are:

- `-h / --help`
- `--version`
- `-v / --verbose`
- `--crcCheck`
- `--openReport`
- `--stats`

These options can be passed as command line option separated by spaces.

For example:

```
MSSV.exe --help
```

or

```
MSSV.exe -v --openReport ...
```

2.3.12.2 Parameters Requiring A Value

Parameters which require a value are:

- `-l / --logFile`
- `-r / --reportFile`
- `-p / --pluginDir`
- `-D / --define`
- `-i / --inputDir`

These parameters require a value if they are specified on the command line. The value is separated from the identifier using a space.

For example:

```
MSSV.exe --reportFile myreport.html --define DEBUG ...
```

2.3.12.3 Examples

Display MSSV Help

To display the MSSV help use:

```
MSSV.exe --help
```

Run MSSV with minimum parameters

Assume that the root path of the delivery is `D:\delivery`. To run MSSV with the minimum required parameters one can use:

```
MSSV.exe --inputDir D:\delivery\*
```

This will instruct MSSV to scan the root path of the delivery recursively.

Log the output of MSSV to a file

Based on the example above, one can extend the command line to log the output of MSSV to a file. This time we use the short form parameters:

```
MSSV.exe --inputDir D:\delivery\* -l mylog.log
```

Write the report to another file and enable verbose mode

The command line from the previous example is adapted to write the report to the file specified and to enable verbose output:

```
MSSV.exe --inputDir D:\delivery\* -r myreport.html -v
```

3 Analysis Report

MSSV generates a report file after performing its consistency checks. By default the report file is named “report.html” and is located inside the working directory where MSSV was executed.

3.1.1 Structure

This report file has a simple structure to display the most important parts at a glance. The parts are described in the next sections.

3.1.1.1 Header

The header displays whether the MSSV consistency check was successful or not.

In case the check was successful and no inconsistencies or errors occurred, the header displays “Overall Check Result: Passed”.

If an inconsistency was detected or other errors occurred, the header displays “Overall Check Result: Fail”.

Additionally there may be a list of plugins which were skipped. This can happen both if the overall check result is pass or fail. In case one or more plugins were skipped, “Skipped Plugins: …” will be displayed directly under the overall check result. This list of skipped plugins needs to be manually reviewed as to why they were skipped and if this has any impact on the consistency of the generated data.

For example if a plugin was skipped because the module it is checking is not enabled and not used, this would most likely not affect the consistency of the generated data. For more information refer to the project specific Safety Manual.

3.1.1.2 Information about Environment

The second section of the report displays some information about the environment of MSSV. This includes for example the current windows user and the number of found errors and warnings.

3.1.1.3 Detailed Log Output

The detailed log output contains all messages from MSSV. This also includes messages which are only visible if the verbose mode is active.

Note
The information found in the detailed log output is complex and not intended to be checked. The only relevant information for the user is the overall test result which says if the check was successful or not. The detailed information becomes relevant if any errors occurred.

3.2 Error Messages

MSSV emits errors either in the detailed log output section of the report file or displays them as message boxes. Errors are only displayed using message boxes if the report could not be opened or written and hence cannot be emitted to the report.

This leads to a simple workflow of MSSV. If an error message was displayed using a message box a report file does not exist. If no message box was displayed all potential errors can be found in the report alongside the overall verdict of the MSSV consistency check.

There exist four classes of MSSV messages:

- Note,
- Warning,
- Error,
- Fatal error.

Notes are emitted very often and are not of importance for the user of MSSV, but can help to trace a problem if one occurred.

Warnings indicate that a minor issue occurred which are not relevant enough to issue an error. For example if a source code file for a plugin was not found, a warning is emitted. An error is not emitted because the user may have disabled the module intentionally in the generator and thus the consistency of the project is not affected by this module. However, the plugin for which the source code file was not found is skipped and added to the list of skipped plugins in the report. This list needs to be reviewed manually.

An error can occur during the consistency check of a BSW module if e.g. an inconsistency was detected. Errors do not prevent MSSV from continuing its consistency check and help the user to identify the underlying issue.

A fatal error indicates that a serious Issue occurred that prevents MSSV from performing the consistency check.

The return value of the `MSSV.exe` is determined by these four classes of messages. The class of the highest occurred message defines the return value.

Message Class	Value
Note	0
Warning	1
Error	2
Fatal Error	3

Table 3-1 Message classes and their value

For example if a warning occurred and no errors of any kind the return value of MSSV is 1. If an error and a fatal error were emitted the return value is 3. If only notes were emitted MSSV returns 0.

3.3 Steps if the Analysis Fails

In case the MSSV analysis reports a failure Vector has to be informed about this issue. Please use the contact information found in the project specific Safety Manual.

4 Integration

Since MSSV checks the consistency of the generated data it is convenient to run MSSV automatically after the data is generated. To integrate MSSV into the MICROSAR 3 workflow a GENy post generation application can be used. For integration into MICROSAR 4 DaVinci Configurator Pro 5 supports external generation steps to allow easy integration.

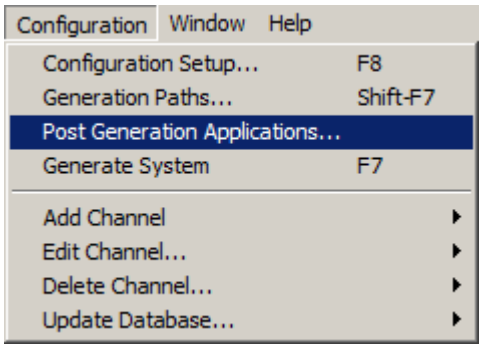
4.1 Deliverables

The following table describes which parts are delivered with the MSSV tool and where to find them.

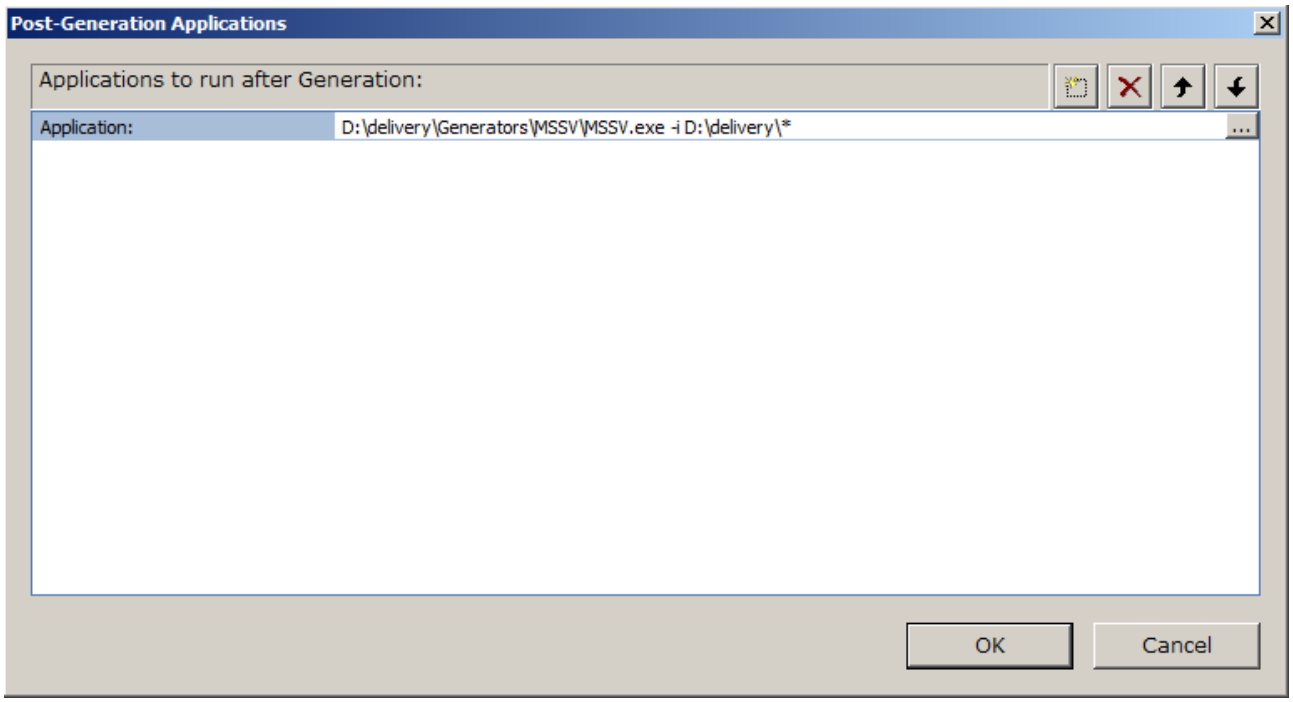
Deliverable	Location
MSSV.exe	.\Generators\MSSV\MSSV.exe
Technical Reference	.\Doc\TechnicalReferences\TechnicalReference_MSSV.pdf

Table 4-1 Locations of Deliverables in an SIP

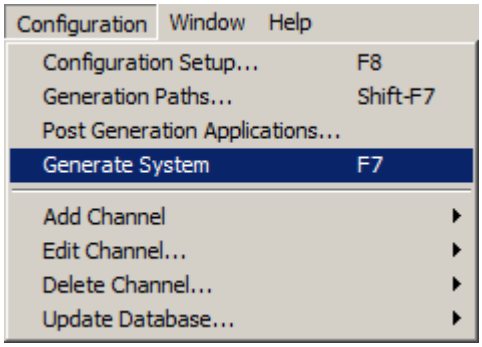
4.2 GENy



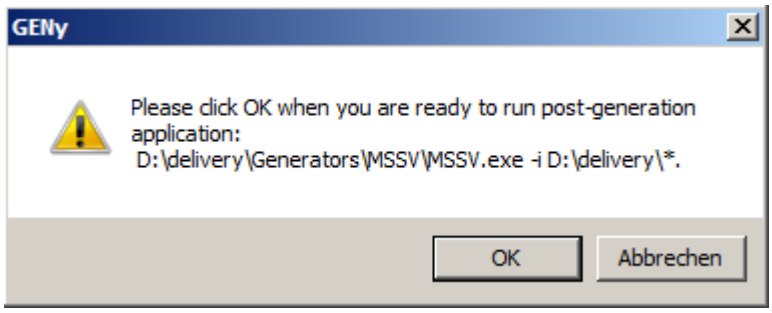
Start GENy and select the menu “Configuration”. Next select the menu item “Post Generation Applications...”. This will display the following window:



In this window you can enter the MSSV command line. First enter the path to the MSSV executable and then add any parameters to it. The input directory is specified as absolute path using the respective command line parameter.



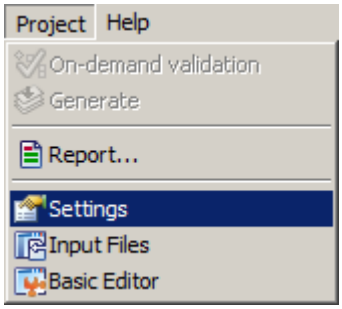
To test the post generation application select again the menu “Configuration” and click the item “Generate System”. Alternatively you can press F7. This will trigger GENy to generate the data for the current configuration.



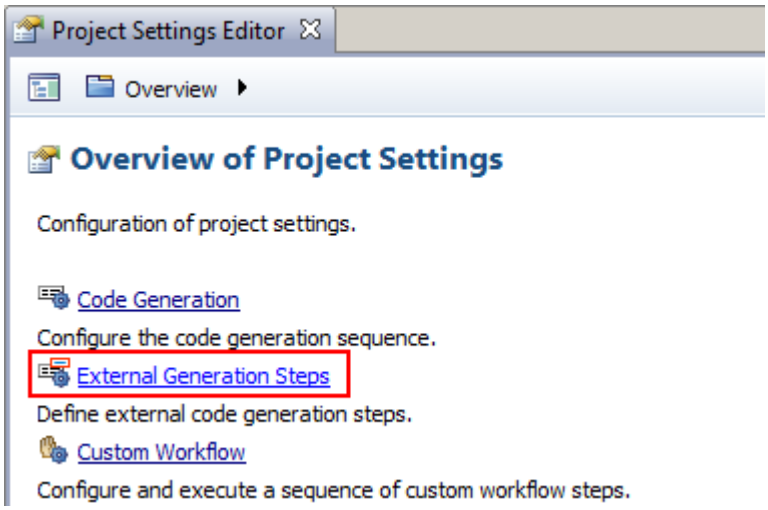
After the Generation has finished generating the data a messagebox will be displayed. This messagebox displays the previously set up post generation application and shows exactly what command line will be executed if the button “OK” is clicked.

To run the post generation application click the “OK” button.

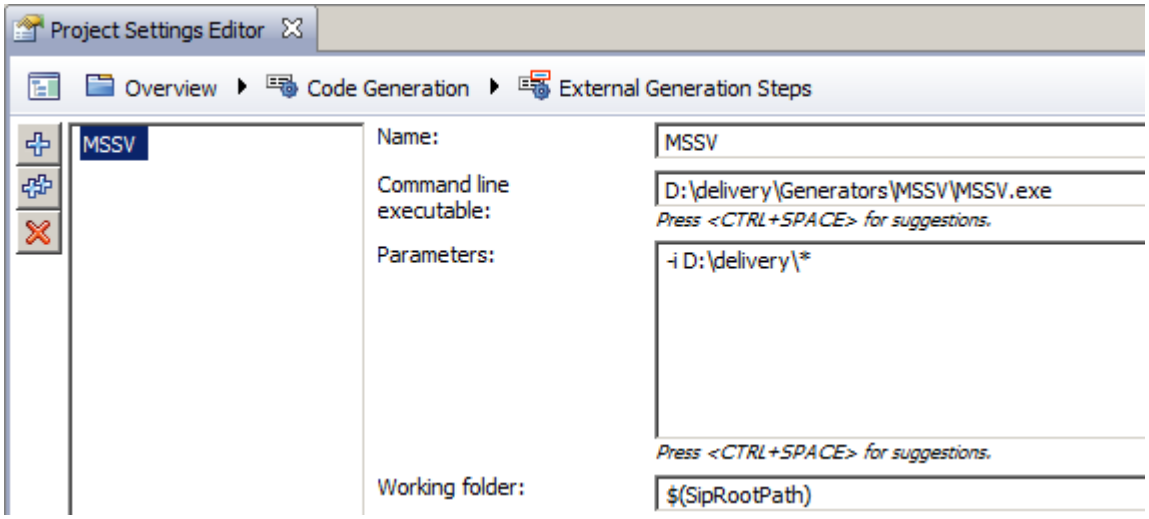
4.3 DaVinci Configurator Pro 5



Start DaVinci Configurator Pro 5 and select the menu “Project”. Next select the menu item “Settings”.

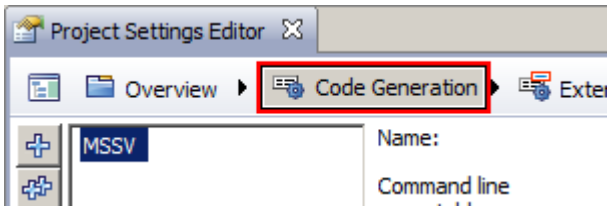


To add a new external generation step, select “External Generation Steps”. This will display the following window:

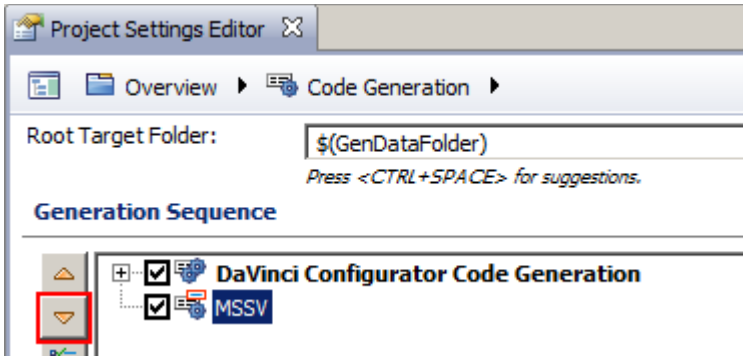


Click on the Add button with the “+” symbol and enter the MSSV path and command line arguments.

Note
It is required to set a working directory for a post generation step.



Now the external generation step needs to be configured to be run after the DaVinci Generators. To configure this click on the item “Code Generation” in the location bar in the same window.



Now select the MSSV Generation Step and enable it by checking the check box in front of it. Additionally MSSV should be run after DaVinci Configurator Pro generated the data. Therefore it is necessary to move it after the DaVinci Code Generation using the Down button with the “▼” symbol.

Now MSSV will be automatically executed after the DaVinci Configurator Pro has generated the data.

Note
MSSV will also be executed if the data was not successfully generated.

5 Third Party Libraries

5.1 Boost

Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization
obtaining a copy of the software and accompanying documentation covered by
this license (the "Software") to use, reproduce, display, distribute,
execute, and transmit the Software, and to prepare derivative works of the
Software, and to permit third-parties to whom the Software is furnished to
do so, all subject to the following:

The copyright notices in the Software and this entire statement, including
the above license grant, this restriction and the following disclaimer,
must be included in all copies of the Software, in whole or in part, and
all derivative works of the Software, unless such copies or derivative
works are solely in the form of machine-executable object code generated by
a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

5.2 ChaiScript

Copyright 2009-2012 Jason Turner and Jonathan Turner. All Rights Reserved.
Redistribution and use in source and binary forms, with or without
modification, are permitted provided that the following conditions are
met:

* Redistributions of source code must retain the above copyright
  notice, this list of conditions and the following disclaimer.
* Redistributions in binary form must reproduce the above
  copyright notice, this list of conditions and the following
  disclaimer in the documentation and/or other materials provided

5.3 LLVM/Clang

* Redistributions of source code must retain the above copyright notice,
  this list of conditions and the following disclaimers.

* Redistributions in binary form must reproduce the above copyright notice,
  this list of conditions and the following disclaimers in the
  documentation and/or other materials provided with the distribution.

* Neither the names of the LLVM Team, University of Illinois at
  Urbana-Champaign, nor the names of its contributors may be used to
  endorse or promote products derived from this Software without specific
  prior written permission.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE CONTRIBUTORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS WITH THE SOFTWARE.

5.4 OpenBSD regex

$OpenBSD: COPYRIGHT,v 1.3 2003/06/02 20:18:36 millert Exp $

Copyright 1992, 1993, 1994 Henry Spencer. All rights reserved.
This software is not subject to any license of the American Telephone
and Telegraph Company or of the Regents of the University of California.

Permission is granted to anyone to use this software for any purpose on
any computer system, and to alter it and redistribute it, subject
to the following restrictions:

1. The author is not responsible for the consequences of use of this
   software, no matter how awful, even if they arise from flaws in it.

2. The origin of this software must not be misrepresented, either by
   explicit claim or by omission. Since few users ever read sources,
   credits must appear in the documentation.

3. Altered versions must be plainly marked as such, and must not be

misrepresented as being the original software. Since few users
ever read sources, credits must appear in the documentation.

4. This notice may not be removed or altered.

```
=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=-=
/*-
 * Copyright (c) 1994
 *	The Regents of the University of California.  All rights reserved.
 *
 * Redistribution and use in source and binary forms, with or without
 * modification, are permitted provided that the following conditions
 * are met:
 * 1. Redistributions of source code must retain the above copyright
 *    notice, this list of conditions and the following disclaimer.
 * 2. Redistributions in binary form must reproduce the above copyright
 *    notice, this list of conditions and the following disclaimer in the
 *    documentation and/or other materials provided with the distribution.
 * 3. Neither the name of the University nor the names of its contributors
 *    may be used to endorse or promote products derived from this software
 *    without specific prior written permission.
 *
 * THIS SOFTWARE IS PROVIDED BY THE REGENTS AND CONTRIBUTORS ``AS IS'' AND
 * ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE
 * IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE
 * ARE DISCLAIMED.  IN NO EVENT SHALL THE REGENTS OR CONTRIBUTORS BE LIABLE
 * FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL
 * DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS
 * OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION)
 * HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT
 * LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY
 * OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF
 * SUCH DAMAGE.
 *
 *	@(#)COPYRIGHT	8.1 (Berkeley) 3/16/94
 */
```

6 Contact

Visit our website for more information on

- News
- Products
- Demo software
- Support
- Training data
- Addresses

www.vector.com