

Intel® System Debugger 2018 - System Trace User Guide

Copyright © 2017 Intel Corporation
All Rights Reserved
Revision: 20170710



Contents

| | | |
|--------|---|----|
| 1 | Legal Information | 5 |
| 2 | About this Document | 6 |
| 3 | Installing from a Web Package | 7 |
| 3.1 | Pre-requisites | 7 |
| 3.2 | Installation..... | 7 |
| 3.3 | Installation Troubleshooting | 13 |
| 4 | Installing the System Trace Eclipse* Plugin Manually | 14 |
| 4.1 | Pre-requisites: | 14 |
| 4.2 | Installation..... | 14 |
| 4.3 | Plugin Installation Troubleshooting: | 19 |
| 5 | Starting System Trace in Eclipse* | 20 |
| 5.1 | Importing Example Rule Sets | 22 |
| 5.2 | Startup Troubleshooting: | 22 |
| 6 | Hardware Setup and Configuration | 23 |
| 6.1 | Connecting Hardware..... | 23 |
| 6.2 | Setting up BIOS..... | 23 |
| 6.3 | Setting up the Target for Event Trace for Windows (ETW) Tracing | 24 |
| 6.4 | Configuring the Target Connection | 27 |
| 6.5 | Configuring and Selecting Trace Sources Using Example Projects..... | 32 |
| 6.6 | Select Trace Sources..... | 33 |
| 6.6.1 | Enabling and Configuring BIOS and CSME traces | 35 |
| 6.6.2 | Enabling and Configuring Architectural Event Traces (AET) | 35 |
| 6.7 | Select Trace Destination..... | 37 |
| 6.7.1 | Enabling and Configuring ETW traces | 39 |
| 6.8 | Select Trace Destination..... | 42 |
| 6.9 | Starting Live Tracing | 43 |
| 6.9.1 | Start Capture without Configuration | 44 |
| 6.10 | Stop Live Tracing | 47 |
| 6.10.1 | Stop Capture without Configuration | 47 |
| 6.11 | Disconnect from Target..... | 48 |
| 6.12 | Hardware Setup and Configuration Troubleshooting:..... | 48 |
| 6.13 | Exporting Trace Configuration | 50 |
| 7 | Trace Analysis Use-Cases | 52 |



| | | |
|--------|---|-----|
| 7.1 | Importing an Existing Trace Capture Archive/Trace Capture File (bin-file) | 52 |
| 7.2 | Stopping an Ongoing Decode | 55 |
| 7.3 | Trace Hardware Tests | 56 |
| 7.4 | Navigating a Trace using Graphical Representation of the Event Distribution | 58 |
| 7.5 | Basic Trace Analysis: Selecting Trace Fields to be Displayed | 63 |
| 7.6 | Basic Trace Analysis: Message View Column Presets | 66 |
| 7.6.1 | Create new Column Preset | 66 |
| 7.6.2 | Export Column Preset | 68 |
| 7.6.3 | Import Column Preset | 68 |
| 7.6.4 | Delete Column Preset | 70 |
| 7.6.5 | Restore default Column Presets | 71 |
| 7.7 | Quick filtering a trace | 72 |
| 7.8 | Basic Trace Analysis: Detailed View of a Trace Entry | 73 |
| 7.9 | Basic Trace Analysis: Marking Trace Entries | 75 |
| 7.10 | Basic Trace Analysis: Message Search | 79 |
| 7.10.1 | Packets Constraints Setting | 79 |
| 7.10.2 | Field Constraints Setting | 80 |
| 7.10.3 | Search Direction Setting | 81 |
| 7.10.4 | Search Type (Search, Filter, Mark) Selection | 81 |
| 7.10.5 | Boundary Setting | 82 |
| 7.10.6 | Search Activation | 82 |
| 7.10.7 | Search Specification Saving | 82 |
| 7.10.8 | Search History | 84 |
| 7.11 | Advanced Trace Analysis: TRAM - TRace Analysis & Mining | 84 |
| 7.11.1 | What Is TRAM? | 84 |
| 7.11.2 | TRAM View | 85 |
| 7.11.3 | Scenario Specification | 87 |
| 7.11.4 | Scenario Detection | 93 |
| 7.11.5 | Scenario Sets | 99 |
| 7.11.6 | Quick Search | 101 |
| 7.11.7 | Trace Analysis preferences | 103 |
| 7.12 | Advanced Trace Analysis: Master/Channel Filtering | 105 |
| 7.13 | Advanced Trace Analysis: Trace Capture File Size Limit | 106 |
| 7.14 | Analyzing Chipset Firmware Flash Logs | 107 |
| 7.14.1 | Use Case Overview | 107 |
| 7.14.2 | Decoding Firmware Flash Logs | 107 |
| 7.15 | Advanced Trace Analysis: Decode Trace with a Different Decode Network | 109 |
| 7.16 | Trace Analysis: Hints | 111 |
| 7.16.1 | Multiple Rules | 111 |
| 7.16.2 | Copying & Pasting of Trace Entries | 111 |
| 7.16.3 | Exporting Traces | 111 |
| 7.16.4 | Showing the Source of Traces | 112 |
| 7.17 | Accessing the Online Help | 113 |



| | | |
|-------|---------------------------|-----|
| 8 | TRAM Language | 114 |
| 8.1 | Events | 114 |
| 8.1.1 | Event Syntax: | 114 |
| 8.1.2 | Packet Constraints | 114 |
| 8.1.3 | Field Constraints | 115 |
| 8.1.4 | Format Constraints | 116 |
| 8.1.5 | Message Macro | 117 |
| 8.2 | Scenario Definition | 118 |
| 8.2.1 | Event Declaration | 118 |
| 8.2.2 | Sequence Definition | 118 |
| 8.2.3 | FSM Definition | 120 |
| 8.3 | Stitch Variables | 121 |



1 Legal Information

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.

Intel disclaims all express and implied warranties, including without limitation, the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, as well as any warranty arising from course of performance, course of dealing, or usage in trade.

This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.

The products and services described may contain defects or errors known as errata which may cause deviations from published specifications. Current characterized errata are available on request.

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at Intel.com, or from the OEM or retailer.

Copies of documents which have an order number and are referenced in this document may be obtained by calling 1-800-548-4725 or by visiting www.intel.com/design/literature.htm.

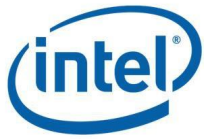
Intel, the Intel logo, Xeon, and Xeon Phi are trademarks of Intel Corporation in the U.S. and/or other countries.

Optimization Notice: Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness of any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice.

Notice Revision #20110804

*Other names and brands may be claimed as the property of others

© 2017 Intel Corporation.



2 About this Document

This User Guide describes how to install the product as well as the most important use-cases to get started with the trace examples shipped with this version.



3 Installing from a Web Package

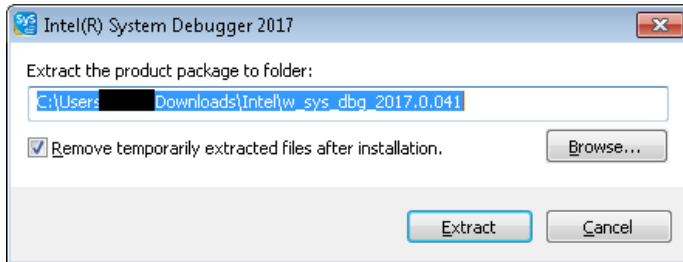
3.1 Pre-requisites

- A working Eclipse* **64-bit** Mars.2 for C/C++ developers
https://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/mars/2/eclipse-cpp-mars-2-win32-x86_64.zip
- Java* RE version 1.8 **64-bit** or higher
- .NET Framework
 - Windows 7: .NET Framework 4.0 and 4.5
 - .NET 4.0 download
<http://www.microsoft.com/en-US/download/details.aspx?id=17718>
 - .NET 4.5 download
<http://www.microsoft.com/en-US/download/details.aspx?id=40779>
 - Windows 8.x: .NET Framework 3.5 and 4.5
 - .NET 3.5 download
<http://www.microsoft.com/en-us/download/details.aspx?id=5007>
(Please be aware that this installer might want to download files from the internet, even it is an offline installer).
 - .NET 4.5 download
<http://www.microsoft.com/en-US/download/details.aspx?id=40779>
- Optional: a working Internet connection, as the Eclipse* plugin installer might need to install missing dependent plugins

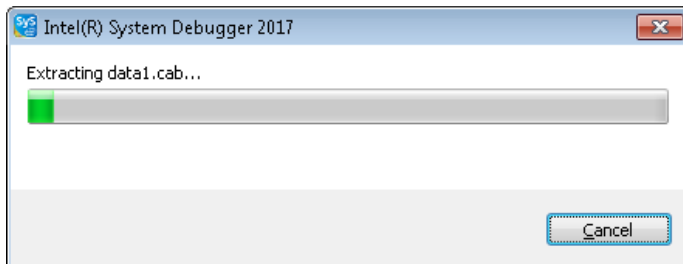
3.2 Installation

To install the Intel® System Debugger 2018 from a downloaded web *package*, do the following:

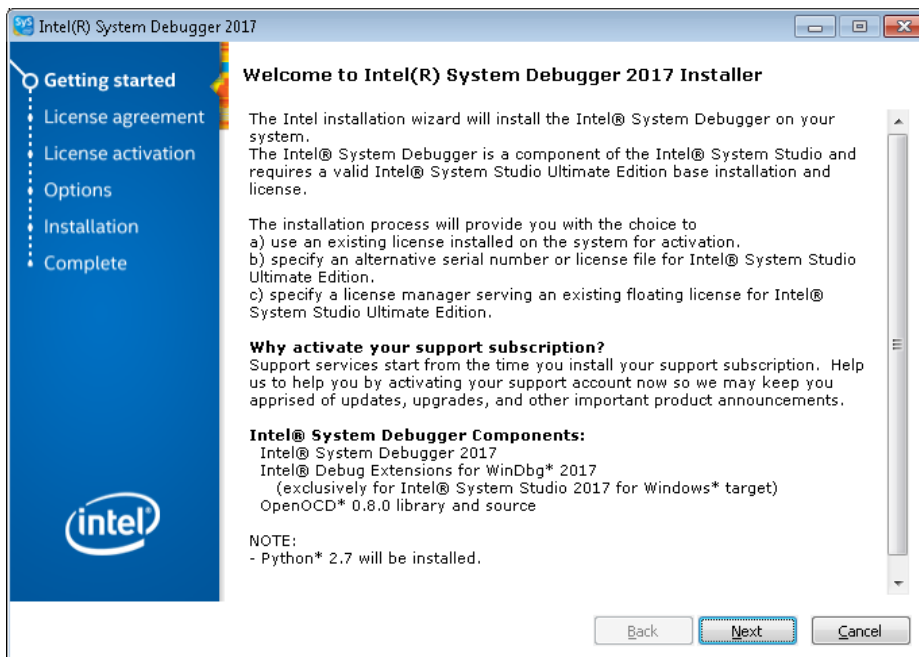
1. To start the installation, double-click the web package installation executable. (Enter or browse to the directory where the installer shall extract the setup files.



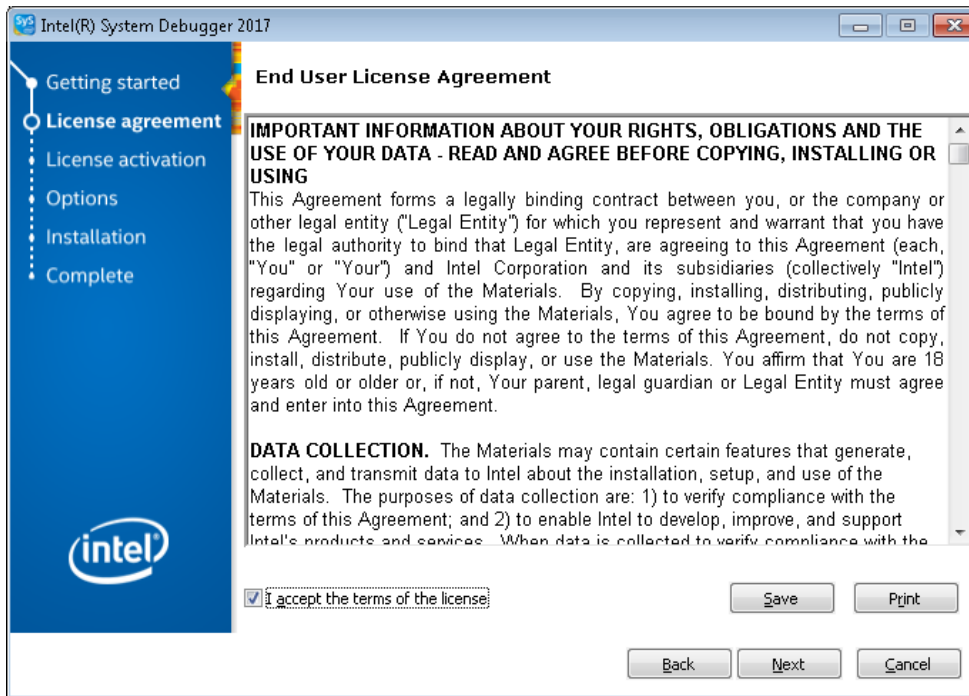
2. Click **Extract**. The installer prepares the installation procedure.



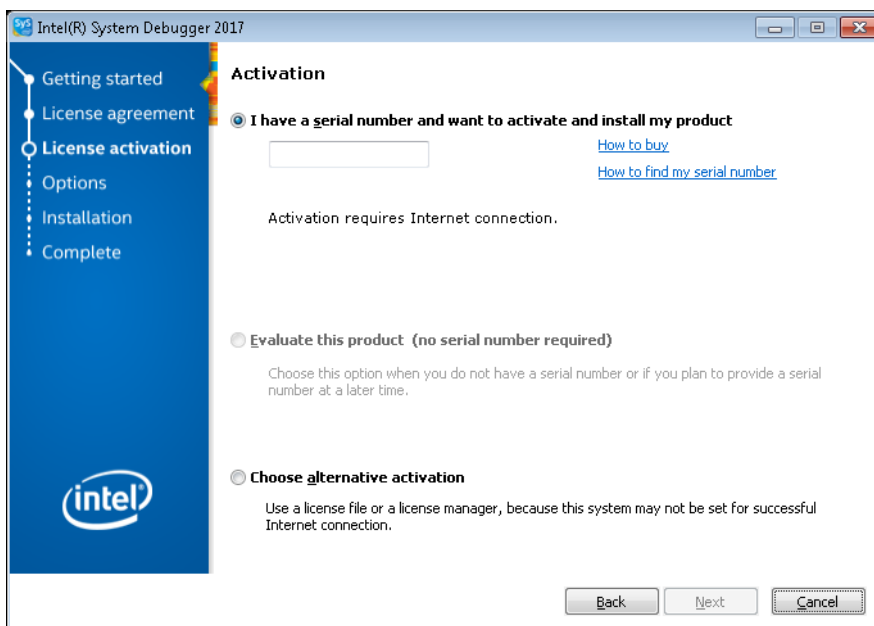
After the extraction is completed, the main setup starts and the following dialog box is displayed.



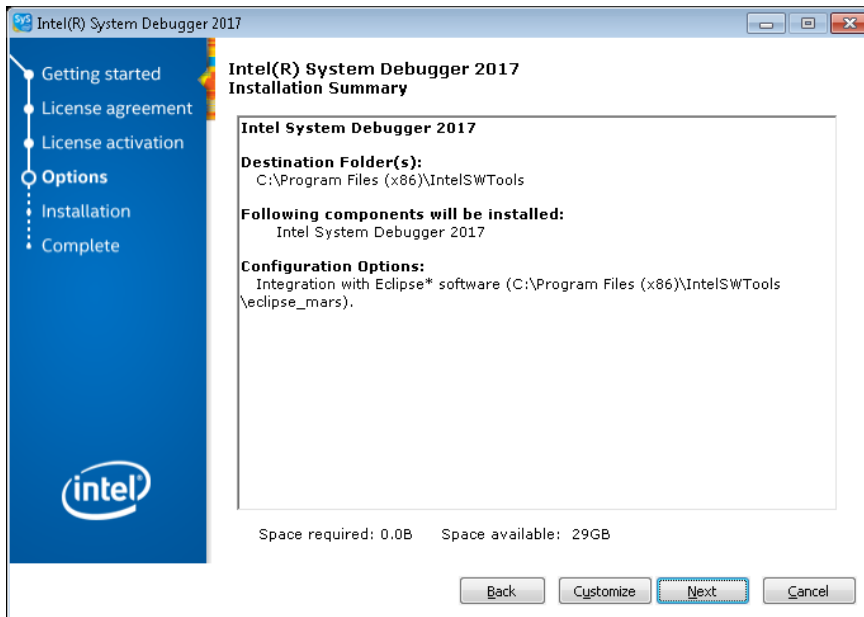
3. Click **Next**.
The Software License Agreement page is displayed.



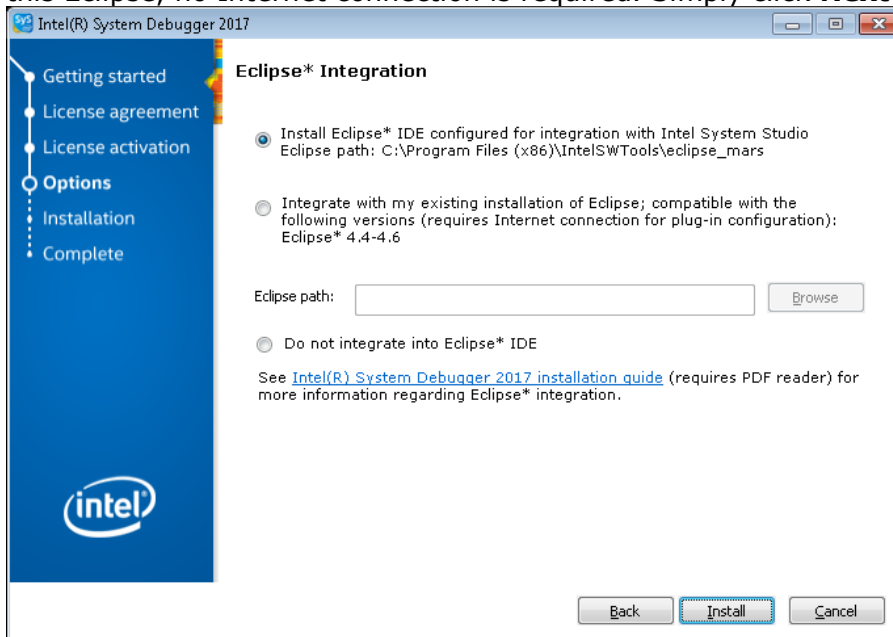
4. If you have accepted the software license agreement, click **Next**.
The Activation page is displayed.



5. Choose the desired option to provide your license and click **Next**.
You will see the installation summary page:



6. In the next step the installer asks for an Eclipse* installation, to integrate the System Trace plugins. If you do not have a compatible Eclipse* installation already (see pre-requisites) on your system, the ISS package comes with an Eclipse* installation, which fulfills all pre-requisites required for the System Trace plugin integration. This pre-configured version is set as Eclipse* location by default in the dialog shown below. Using this Eclipse, no Internet connection is required. Simply click **Next** in this case.





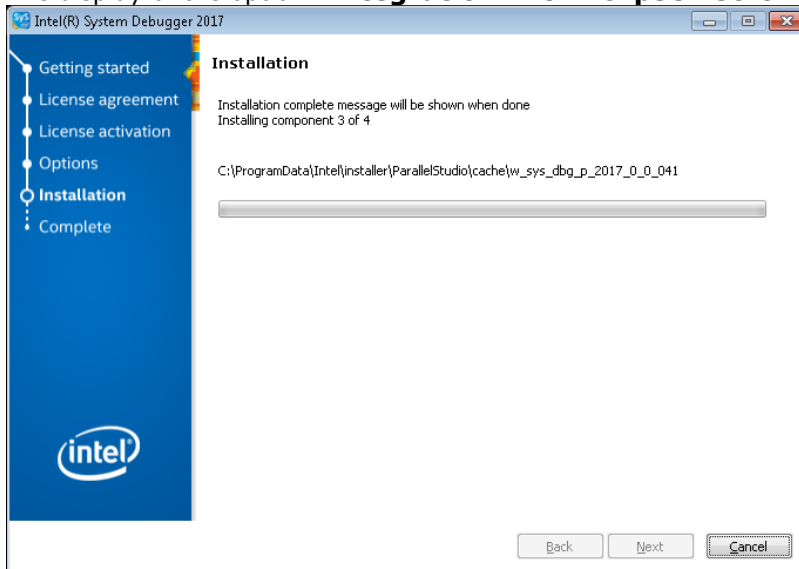
If you want to integrate into your own Eclipse* installation, choose the location of your Eclipse* installation, for example, C:\eclipse:

Please note that a working Internet connection is required as your Eclipse* installation might need to download dependent plugins.

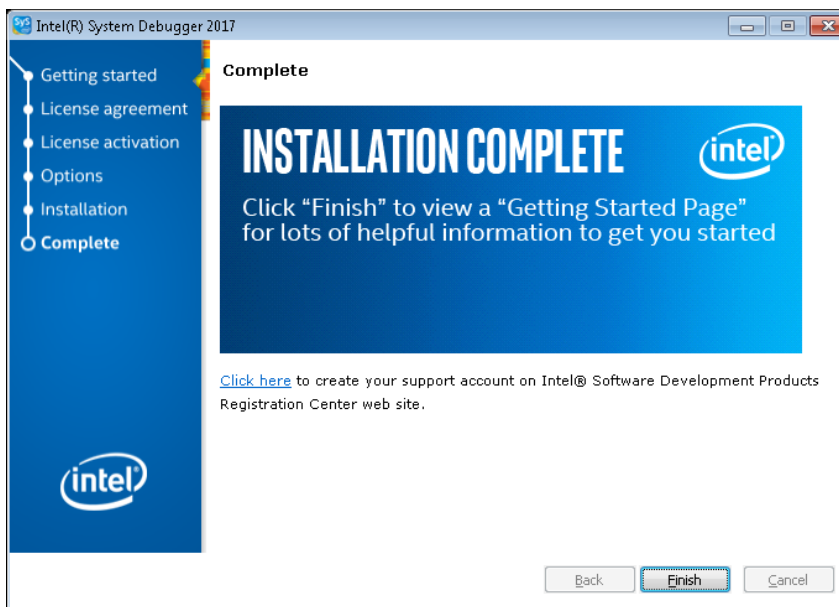
If you are not interested in the Eclipse* integration, select **Do not integrate into Eclipse* IDE.**

7. After clicking **Install**, the installation starts.

NOTE: The display of the option: **Integration with Eclipse* software** depends on your selection.



When the installation is finished, the following page indicates the installation success.



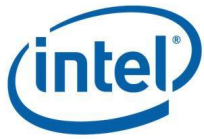


8. Click **Finish** to close the installation dialog box.



3.3 Installation Troubleshooting

| Symptom | Solution |
|---|--|
| <i>The Eclipse* installation does not work or the installation takes very long time.</i> | Make sure that you have the correct Eclipse* version installed (Eclipse* Mars.2) and you have a working Internet connection. If you are behind a firewall and need a proxy for Internet access, make sure this proxy is correctly set in Eclipse* (Window > Preferences > General > Network Connections). |
| <i>Eclipse* is not showing the System Trace perspective in the Open View window.</i> | Make sure your system fulfills the requirements, including Java* 1.8 64bit and Eclipse* Mars.2 64bit. |
| <i>The installation cannot write to the default installation location</i> | Make sure you install the Intel® System Debugger 2018 package with administrator rights. |
| <i>My old System Trace workspace/project does not work with the newly installed Intel(R) System Studio.</i> | Make sure you have created a new System Trace project. If this does not work, please create a new Eclipse* workspace and a new System Trace project. |
| <i>A workspace in the installation directory does not work</i> | User files, such as Eclipse* workspaces, should never go to system sensitive directories. Please create your workspace in your user home directory or any other directory where the current user has write access. |
| <i>The installation aborts and no meaningful error message is displayed.</i> | <p>If this is the case, the installer has most likely written log-files, which helps identifying the problem. The log-files can be found at the following location:</p> <pre>C:\Users\<username>\AppData\Local\Temp\ pset_tmp_ISS_2018_<username>\</pre> <p>Every install run will create a sub directory marked with date and time. Inside you will find log-files, which provide information on the possible root cause.</p> |



4 Installing the System Trace Eclipse* Plugin Manually

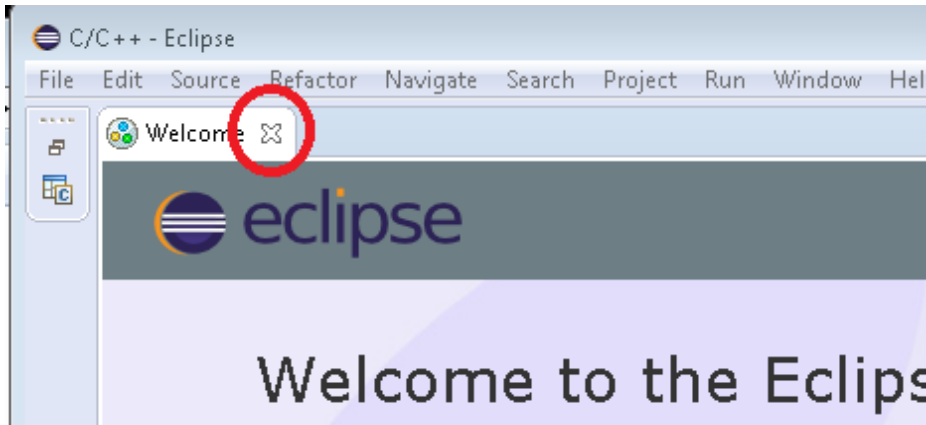
If the Eclipse* plugin integration of the Intel® System Debugger 2018 fails, you can install the System Trace Eclipse* Plugin manually as described in the subsequent sections.

4.1 Pre-requisites:

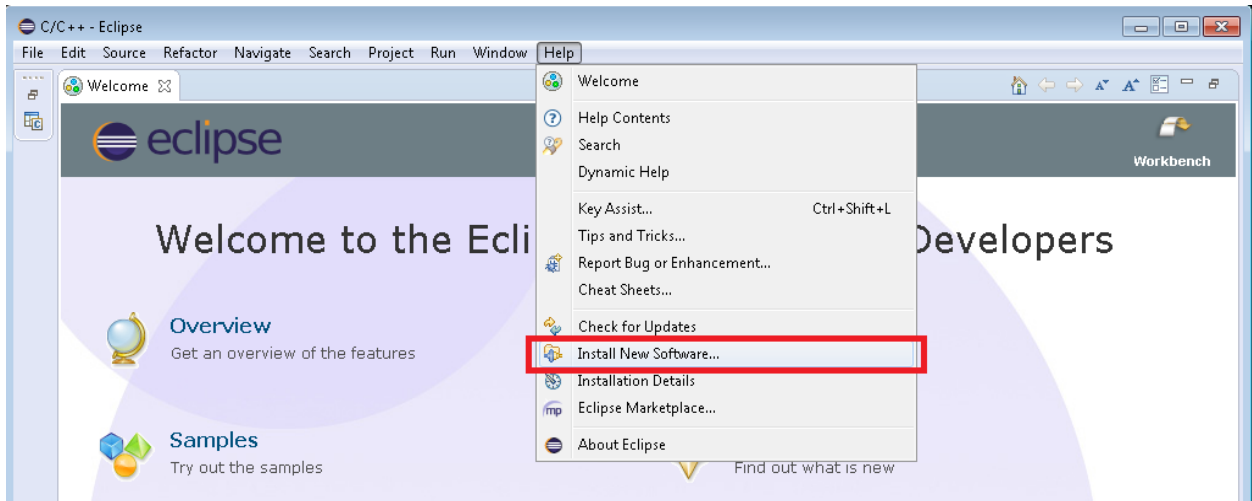
- A working Eclipse* **64-bit** Mars.2 for C/C++ developers
https://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/mars/2/eclipse-cpp-mars-2-linux-gtk-x86_64.tar.gz
or
a working Eclipse* **64-bit** Neon.1a for C/C++ developers
http://www.eclipse.org/downloads/download.php?file=/technology/epp/downloads/release/neon/1a/eclipse-cpp-neon-1a-linux-gtk-x86_64.tar.gz
- Java* RE version 1.8 64-bit or higher

4.2 Installation

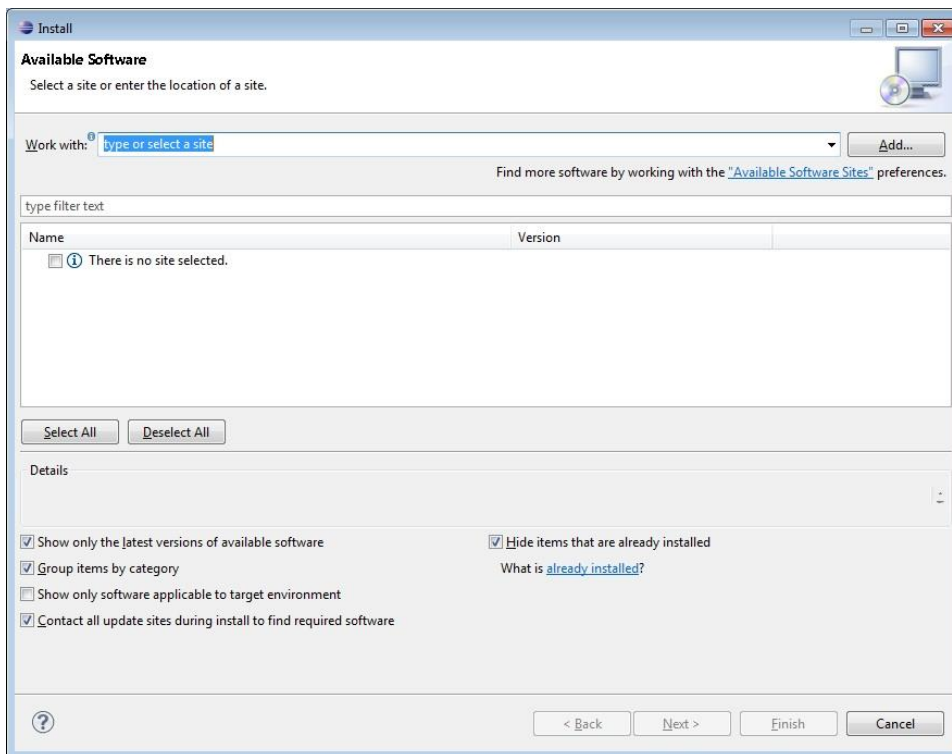
1. Start Eclipse* and close the **Welcome** page.



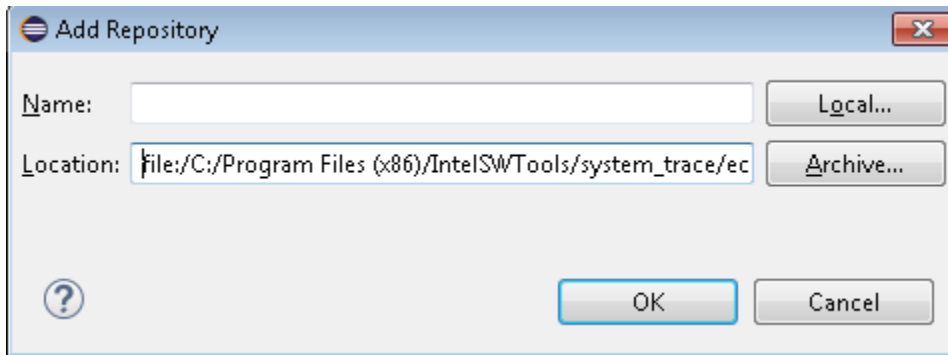
2. Select **Help** > **Install New Software**.



The following **Install** dialog box is displayed.

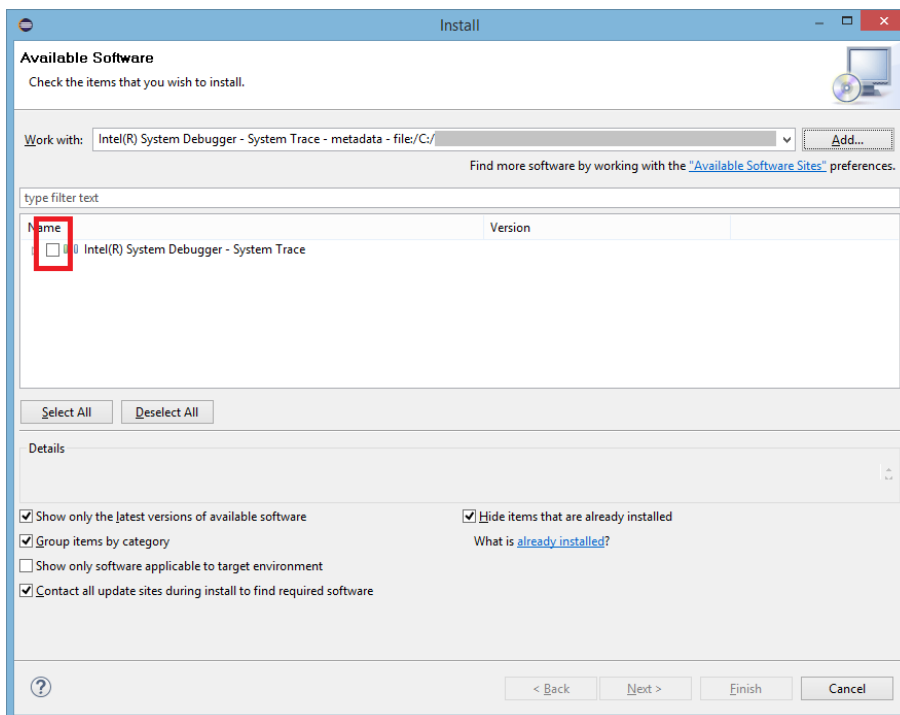


3. Click **Add ...** at the top right corner of the dialog and select the Intel® System Debugger 2018 plugin source. The following dialog box is displayed.

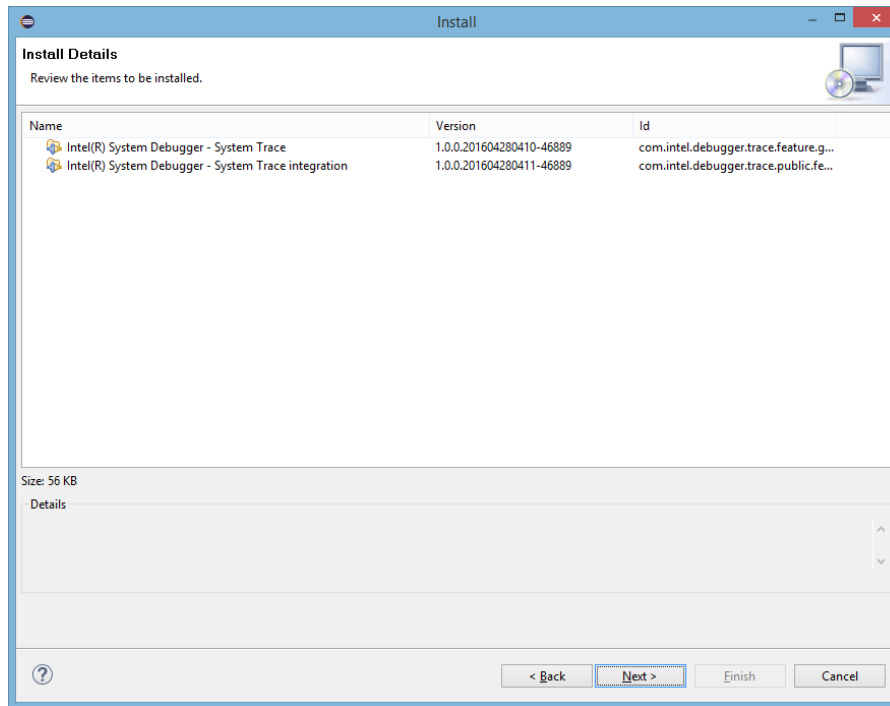


4. Click **Local ...** to add a local Eclipse* repository. Select the following path: <INSTALL-ROOT>\system_trace\eclipse_repository
5. Click **OK**.

A dialog box is displayed, where you can select the plugin "System Trace" and "System Trace, Skylake Trace". Make sure you have set the checkboxes exactly like shown in the screenshot below, otherwise you might not see the "System Trace, Skylake Trace" component.

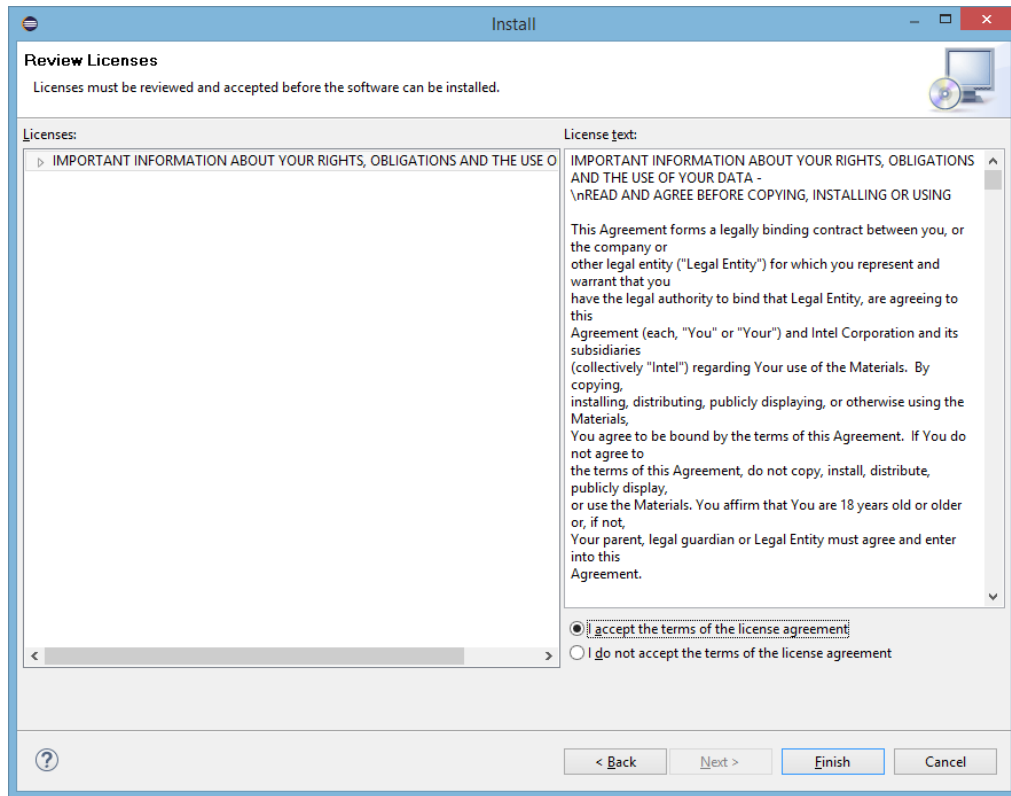


6. Click **Next**.
- The following dialog box is displayed.



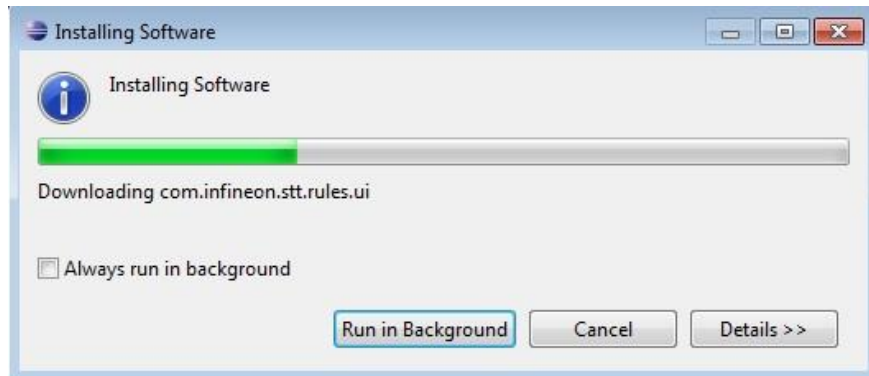
7. Click **Next**.

The license is displayed.



8. Accept the license and click **Finish**.

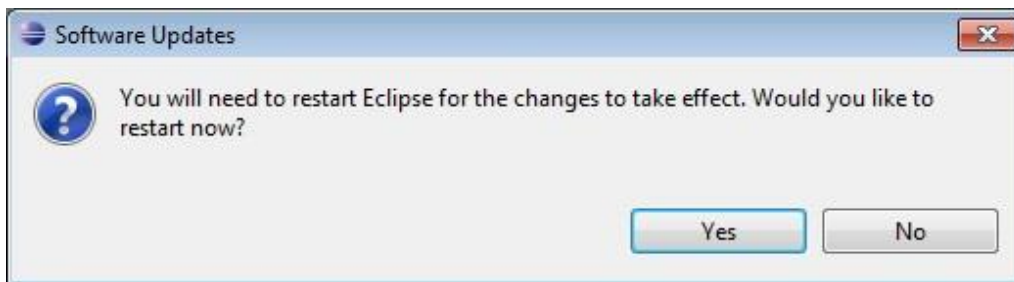
The installation starts.



9. Confirm the installation of unsigned 3rd party plugins with **OK**.

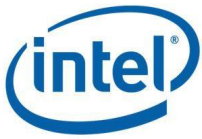


10. After successful installation, restart Eclipse*.



4.3 Plugin Installation Troubleshooting:

| Symptom | Solution |
|---|--|
| <i>The System Trace does not appear after selecting the perspective. Instead the Eclipse* Welcome page is shown</i> | Close the Eclipse* Welcome page by clicking the "X" right to the Welcome tab. Open the System Trace perspective as described in section 10. |
| <i>Eclipse* does not install the plugins due to unsigned content</i> | Confirm any dialog boxes about unsigned content during the installation of System Trace with OK . |
| <i>I cannot find the local Eclipse* repository to install the plugins manually</i> | Make sure you installed the Intel® System Debugger 2018 package correctly. Try to re-install the package and skip the Eclipse* integration step during the installation. |
| <i>The installation takes very long time</i> | If you integrate into your own Eclipse* installation it can be that some plugins need to be fetched from the Internet. If your internet connection is or does not work at all, this step might take long until Eclipse* runs into a timeout. If you are behind a firewall and need a proxy for Internet access, make sure this proxy is correctly set in Eclipse* (Window > Preferences > General > Network Connections) |



5 Starting System Trace in Eclipse*

This chapter shows you how to start the System Trace.

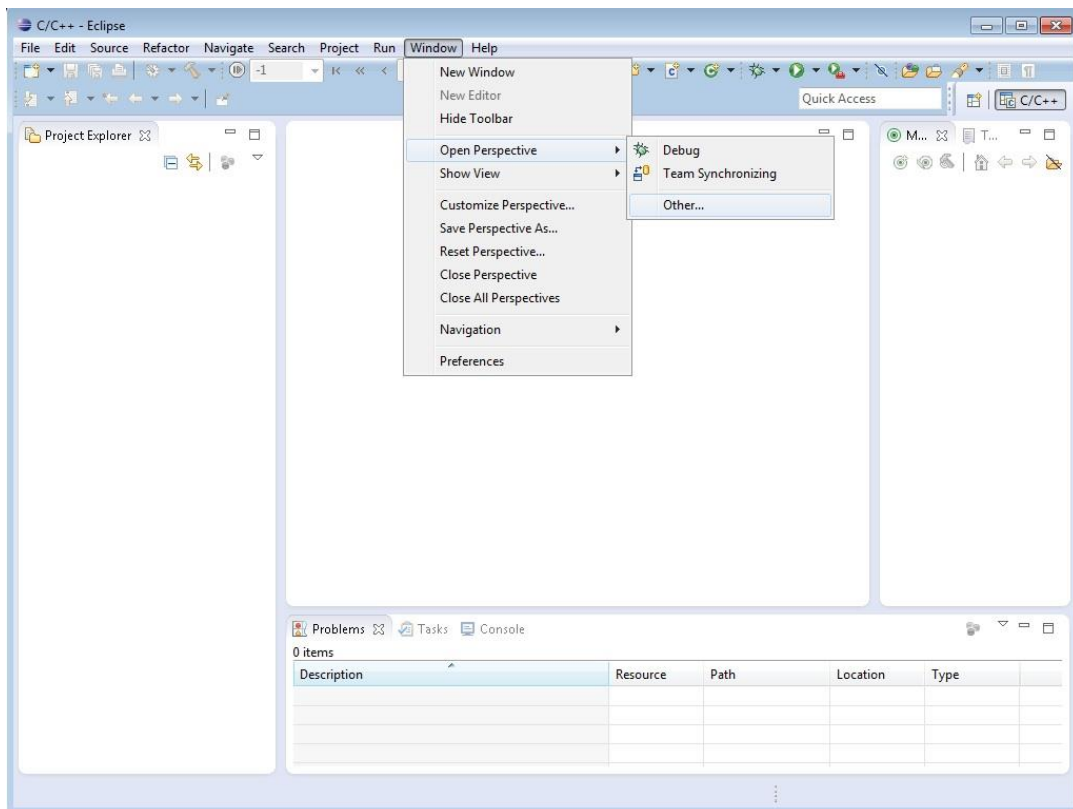
1. Start Eclipse*. **The pre-configured default Eclipse* installation can be found in <INSTALL-DIR>\eclipse_mars, e.g.:
C:\Program Files (x86)\IntelSWTools\eclipse_mars\eclipse.exe.**

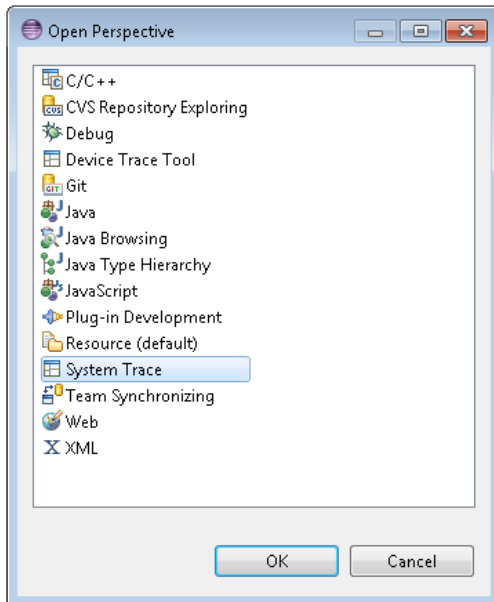
Important note:

If you have an existing workspace containing an old System Trace project this project is probably not compatible with the newly-installed System Trace anymore.

Recommended procedure: Whenever you install a newer version of Intel® System Studio including the System Trace, please create a new Eclipse* workspace or at least a new System Trace project.

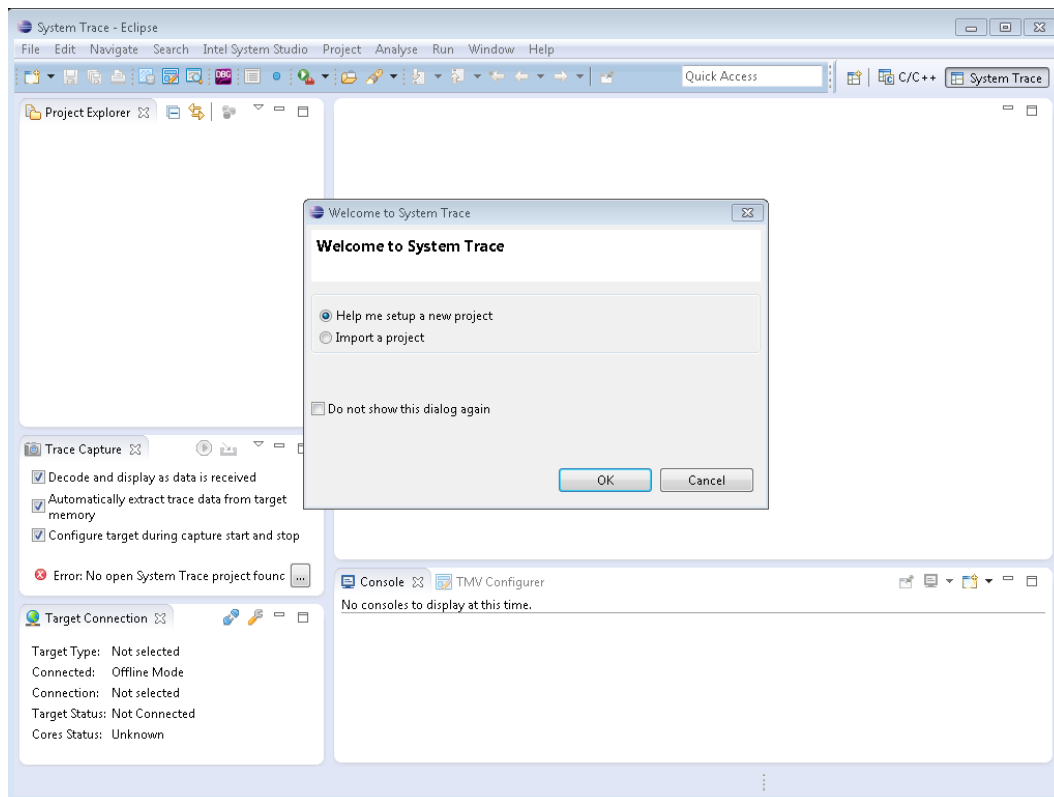
2. To open the System Trace plugin, go to **Window > Open Perspective > Other.**





3. Choose **System Trace** as perspective.

The System Trace default window is displayed. The first time you launch the System Trace will show a welcome dialog, which guides you through the first steps.



Options available on the **Welcome to System Trace** dialog:



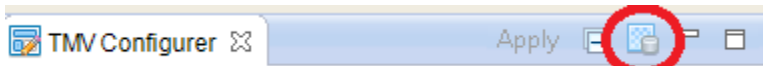
- **Help me setup a new project**
This option will guide you through the process of creating a trace project.
- **Import a project**
This option will open the Eclipse* import dialog to import existing trace projects.

5.1 Importing Example Rule Sets

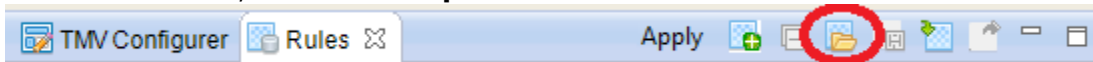
The Intel® System Studio provides example rule sets for supported trace sources. For example, to mark trace messages based on their origin. The example rules can be found in <INSTALLDIR>\system_trace\examples\rules

To import these example rule sets do the following:

- 1) In the **TMV Configurer** view, open the rules view by clicking on the **Open Rules View** icon:



- 2) In the Rules view, click on the **Open Rules** button.



- 3) In the Import Rule(s) Settings dialog browse to a rules file in the location mentioned above.
- 4) Click **OK** and you should see the imported rules in the **Rules** view.
- 5) In the **TMV Configurer** view, you can now use the imported rules to filter, search or mark trace messages.

5.2 Startup Troubleshooting:

| Symptom | Solution |
|---|---|
| <i>Eclipse* is not showing the System Trace perspective in the Open View window.</i> | Make sure your system fulfills the requirements, including Java* 1.8 64bit and Eclipse* Mars.2 64bit. |
| <i>System Trace displays error messages when restarting Eclipse* after installation</i> | Make sure you have the right Java* RE installed. Make sure you have Java* version >= 1.8 64bit installed. |

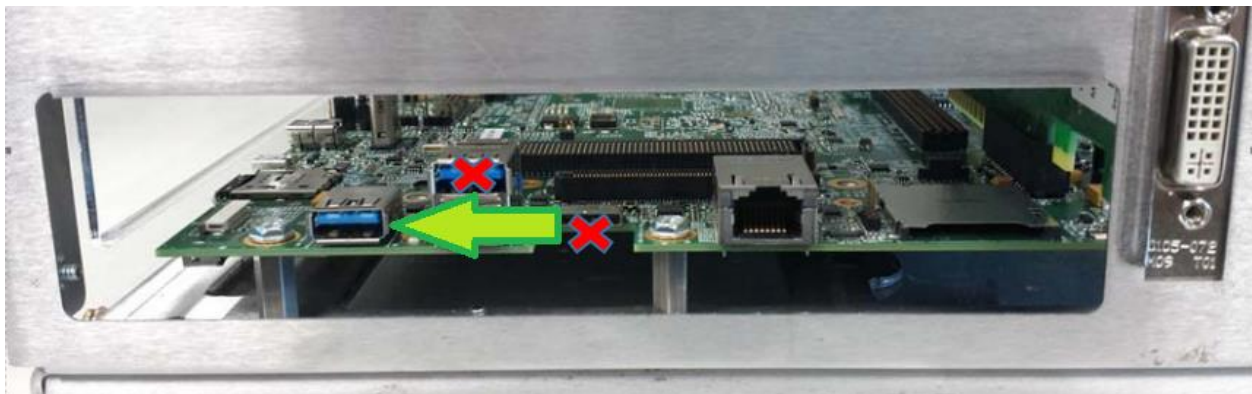
6 Hardware Setup and Configuration

This chapter describes how to configure the target for doing live streaming using the System Trace. Furthermore, it describes how to start and stop live tracing.

Note: *If you experience problems during the connection procedure, please refer to the trouble shoot section at the end of this chapter.*

6.1 Connecting Hardware

Connect a USB cable to target side of the Closed Chassis Adapter (CCA) and to the solitary USB port on the target board. Connect a USB cable to the host side of the CCA and to the host running the Intel® System Debugger 2018.



6.2 Setting up BIOS

The Intel® Trace Hub needs to be enabled in the BIOS settings before tracing is possible. To enable this functionality enter the BIOS and enable the following settings:

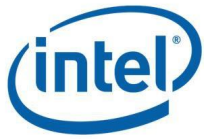
Intel Advanced Menu → PCH-IO Configuration → Trace Hub Configuration Menu → Trace Hub Enable Mode → Set to value **<Host Debugger>**

Intel Advanced Menu → PCH-IO Configuration → DCI Enable (HDCIEN) – Set to 'Enabled'

Intel Advanced Menu → CPU Configuration → Debug Interface → Set to value **<Enabled>**

Intel Advanced Menu → CPU Configuration → Direct Connect Interface → Set to value **<Enabled>**

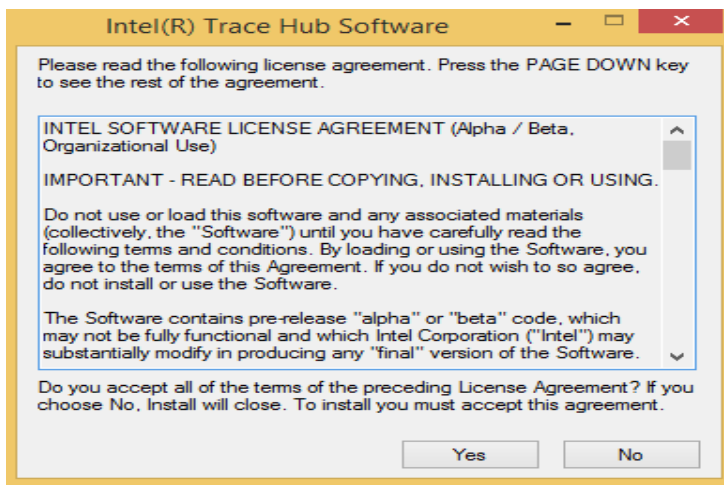
Save and exit the BIOS.



6.3 Setting up the Target for Event Trace for Windows (ETW) Tracing

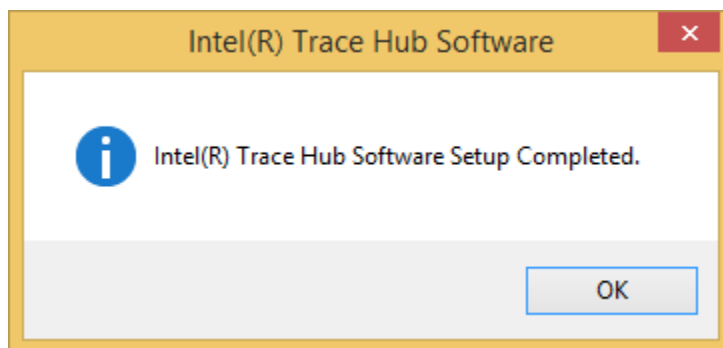
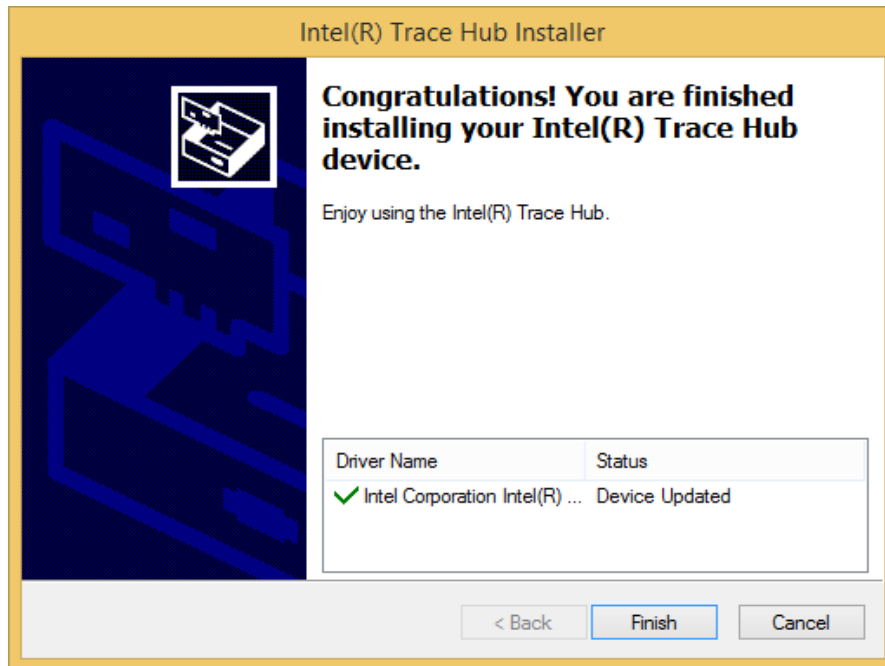
This section can be skipped if ETW is not of interest. Once Windows is installed on the target the ETW drivers need to be installed on the target side. The drivers can be downloaded from:
<https://platformsw.intel.com>

Driver and service installation package: the driver and service installation package comes as a single setup.exe package. Running the file installs both components. Select the setup.exe file and click **Run as Administrator** option to start the installation.



Read the agreement and click Yes to continue with the installation.

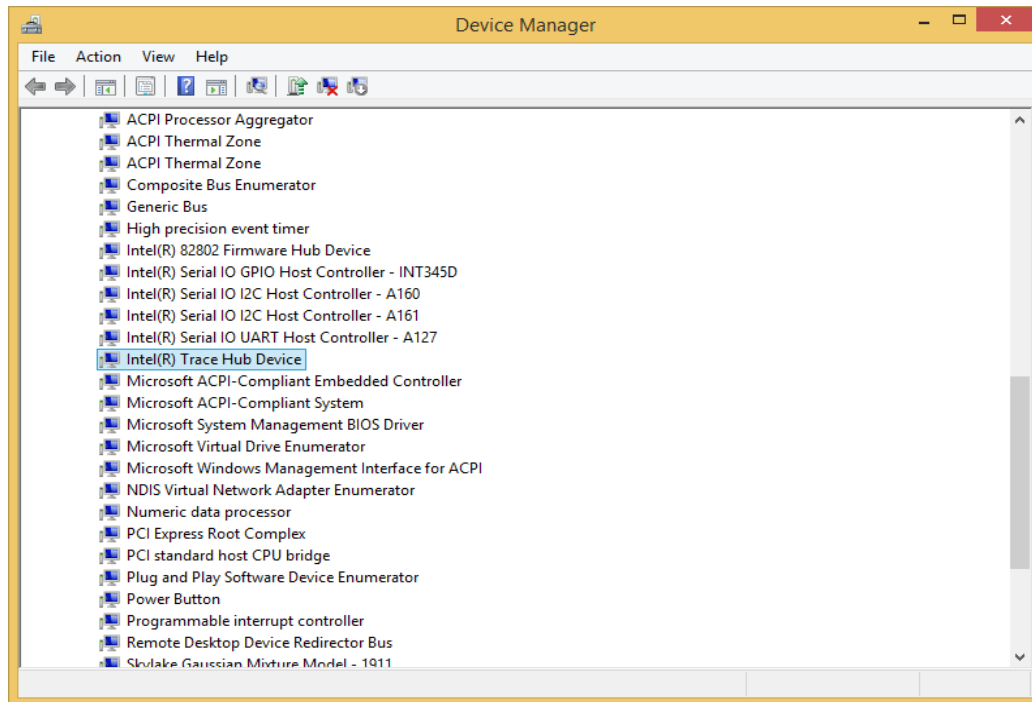
Go through the guided installation and the final screen will appear as below.



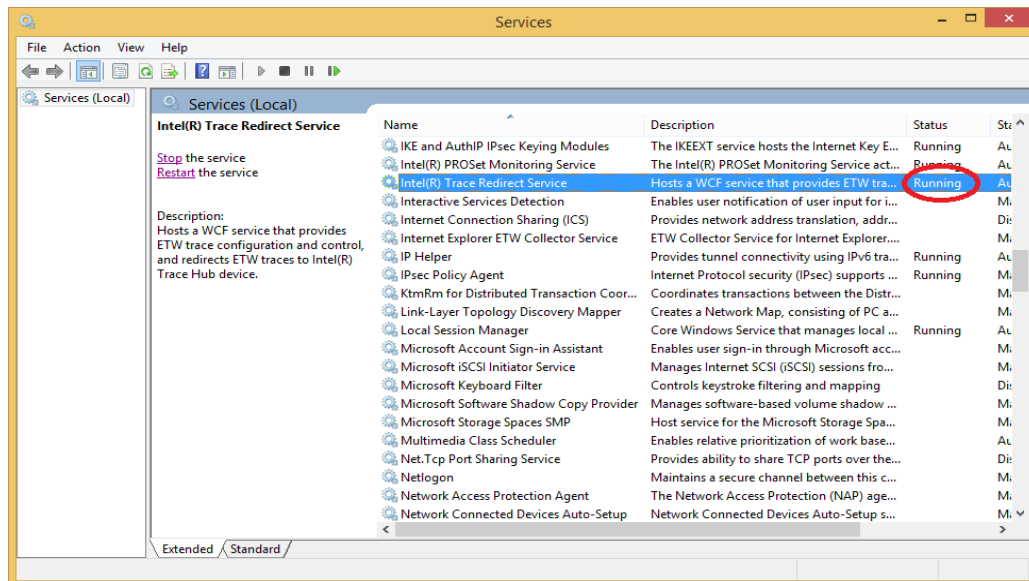
The installer displays a message after the installation process is complete. Indicating the Intel® Trace Hub Driver is updated and Intel® Trace Redirect service is created. Click **OK** to close the window.

To verify that the setup was successful please execute the next three steps.

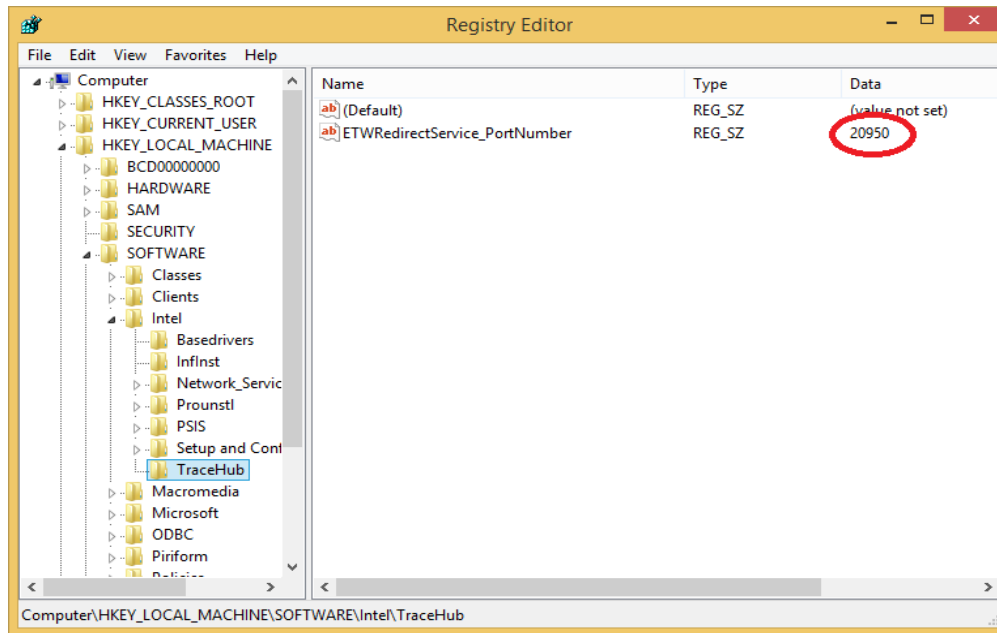
Step 1: Go to the Windows* device manager and check that the Intel® Trace Hub Device is listed correctly under the **System Devices** hive without any error indications:



Step 2: Now open the Windows* Services dialog (run services.msc) and verify that the Intel® Trace Redirect Service status is “Running”. If not, start the service by right clicking on the entry and select **Start**.



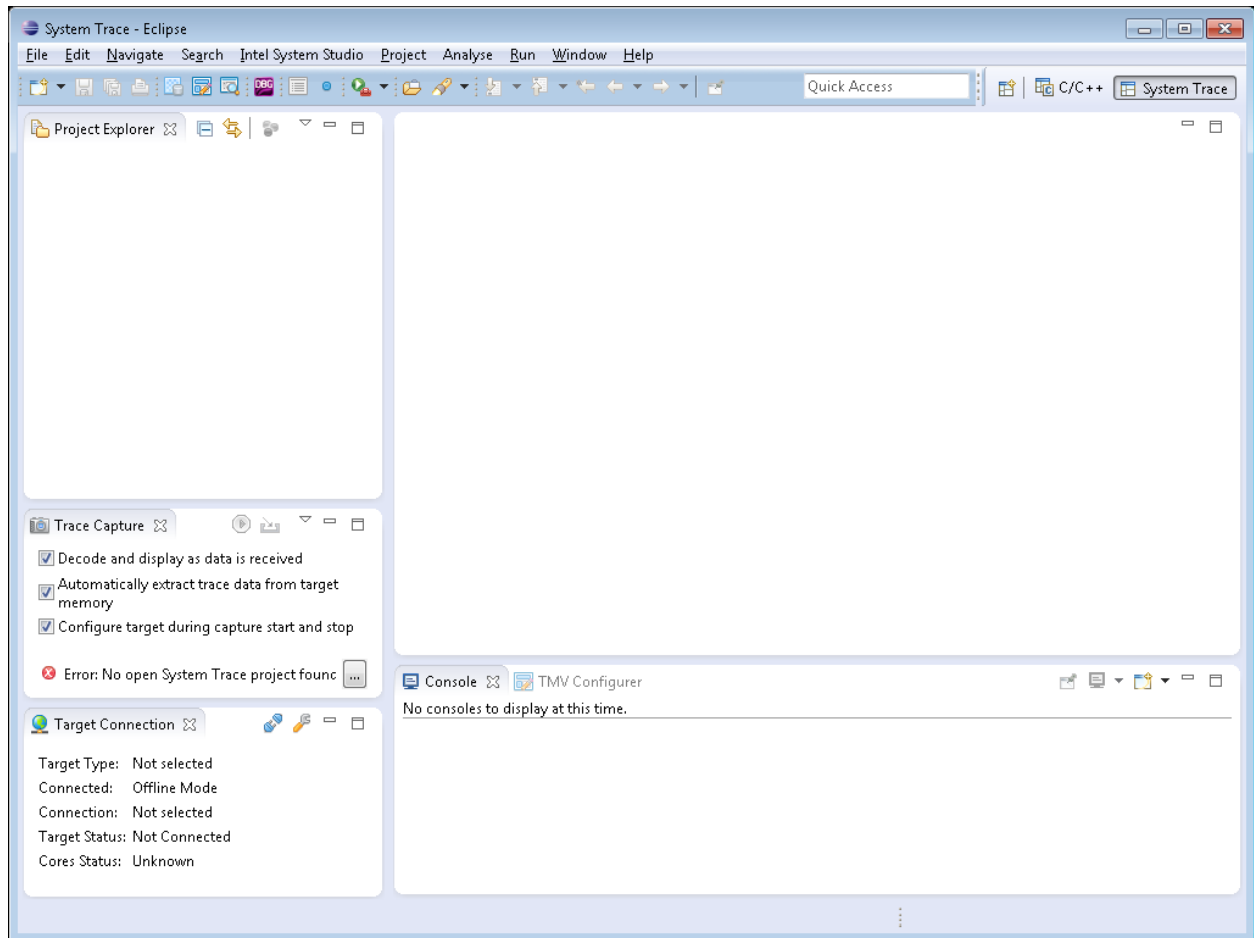
Step 3: Open the Registry Editor by executing “regedit” and verify if the folder Trace Hub is created under “HKEY_LOCAL_MACHINE\SOFTWARE\Intel” and the value ETWRedirectService_PortNumber is set to 20950. The target is now ready for ETW tracing.



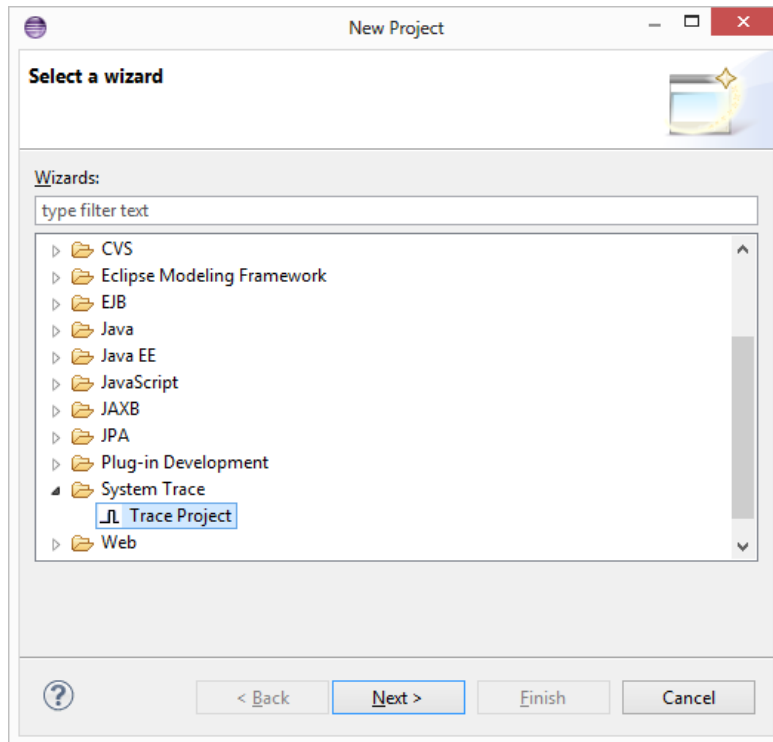
6.4 Configuring the Target Connection

Open the System Trace as described in the previous chapters.

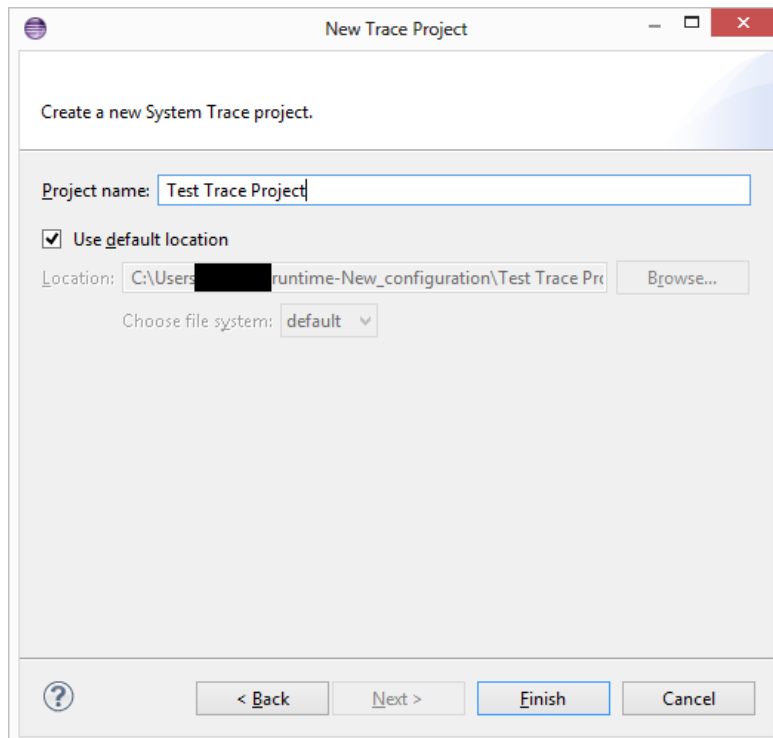
You should see the following screen (System Trace startup screen):



In order to configure the target, you need to create a **Trace Project** first. To do this select **File > New > Project...** and browse to **System Trace > Trace Project**:

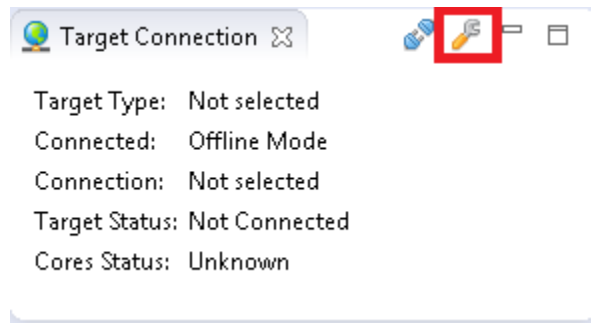


Click Next and enter a project name in the following dialog.

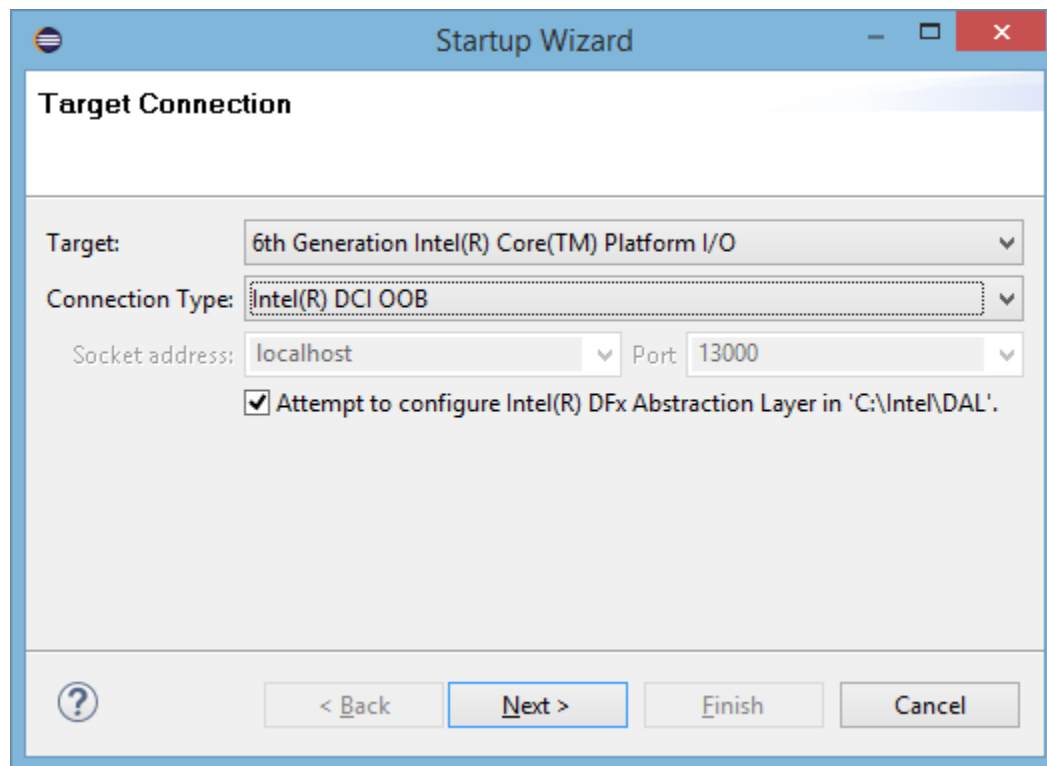




After creating the Trace Project, click on the **Configure Target Connection** button to specify the target you are using:



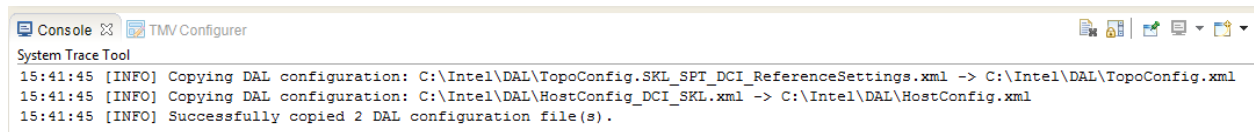
In the following dialog, specify the target and the connection type:



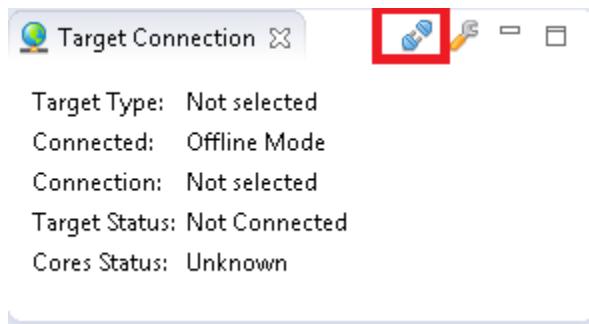
If you want to configure the DAL installation automatically, make sure **Attempt to configure DAL installation** is checked.

Confirm this dialog with **Finish**.

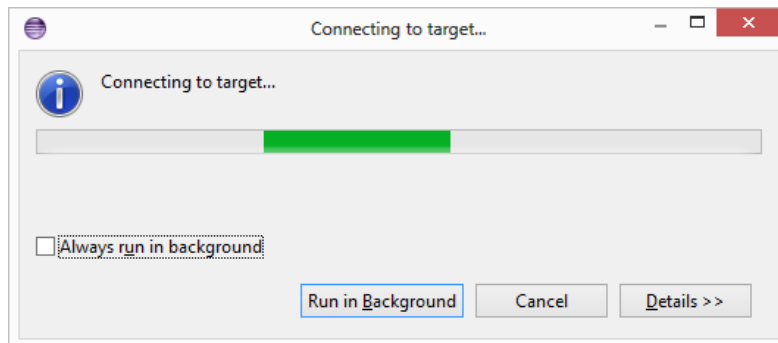
Now the connection type is set and in case **Attempt to configure DAL installation** is checked, the right DAL configuration files are selected automatically, which is indicated in the Eclipse* console view, e.g.:



To connect to the target, click on the **Connect** button. It is shown as un-plugged connector, which means the current state is disconnected.



The System Trace is now trying to connect to your target using the connection type you specified just before.



NOTE: Connecting for the first time can take up to 5 minutes as the CCA firmware might need to be updated. Check the messages in the console window to see if such an update is still progressing.

Once this is done, the System Trace will indicate that the target is now connected by updating the status in the **Trace Capture** view to **Connected**. The connection icon also changed to a plugged connector to indicate connected state.

NOTE: The state **Connected** means that the System Trace is connected to the selected target system. Connected does not mean that the Intel® Trace Hub is already configured. The configuration of the Intel® Trace Hub is done when the capturing starts (see next paragraphs).



The Target Connection dialog also shows the current Target Status and Core Status:

Connected: Connected
Connection: Intel(R) DCI OOB
Target Status: Ok
Cores Status: Running

6.5 Configuring and Selecting Trace Sources Using Example Projects

The Intel® System Studio provides example projects for supported trace sources. These examples are located in the following directory:

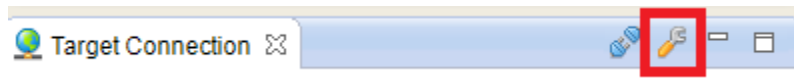
<INSTALLDIR>\system_trace\examples\projects

To import such a project do the following in Eclipse*:

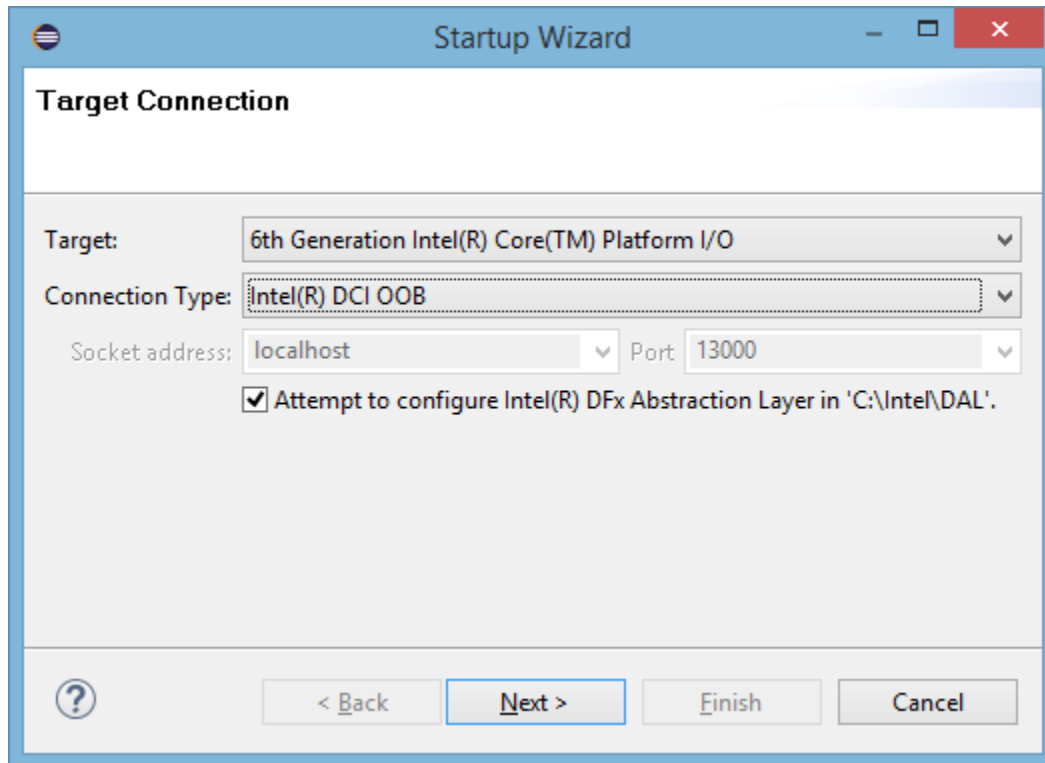
- 1) Open the System Trace as described in the previous chapters
- 2) Select **File > Import ...**
- 3) Select **General > Existing Projects into Workspace** and click **Next**
- 4) Select as archive file one of the example projects in the location mentioned above.
- 5) Click Finish

The example trace project should occur in the Project Explorer. Inside this project, you will find an example configuration in the folder configuration. Open this configuration.

To start tracing you only need to specify the target you are using and click **Connect**.

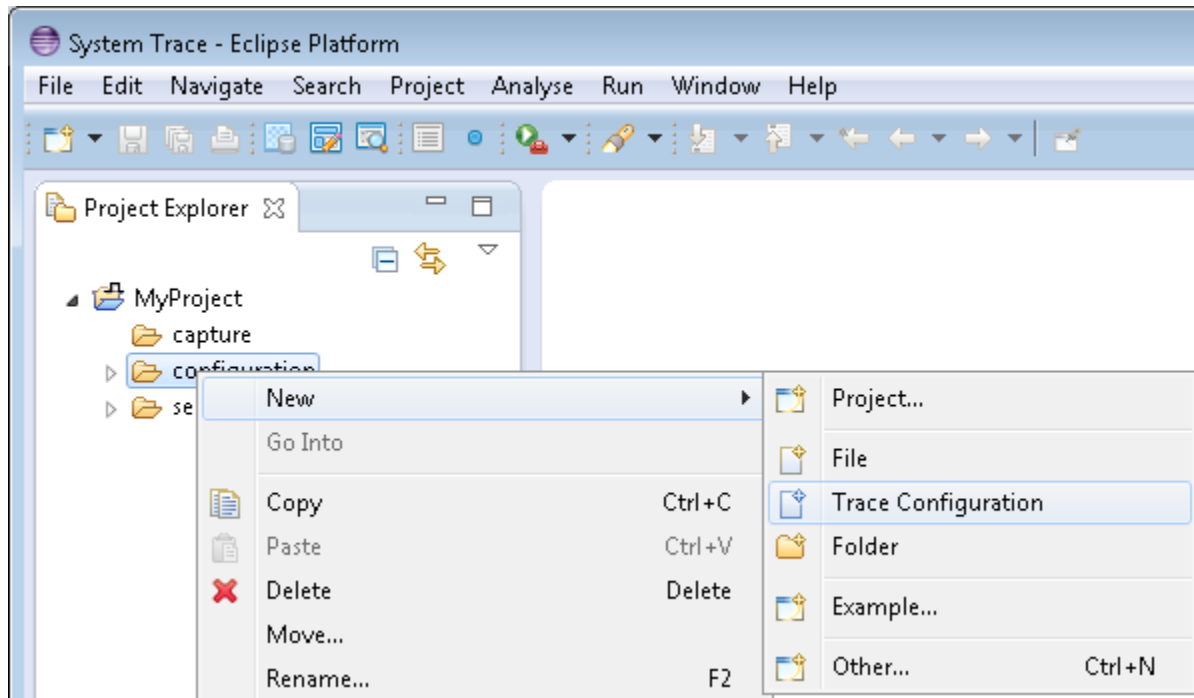


In the following dialog specify the target and the connection type.

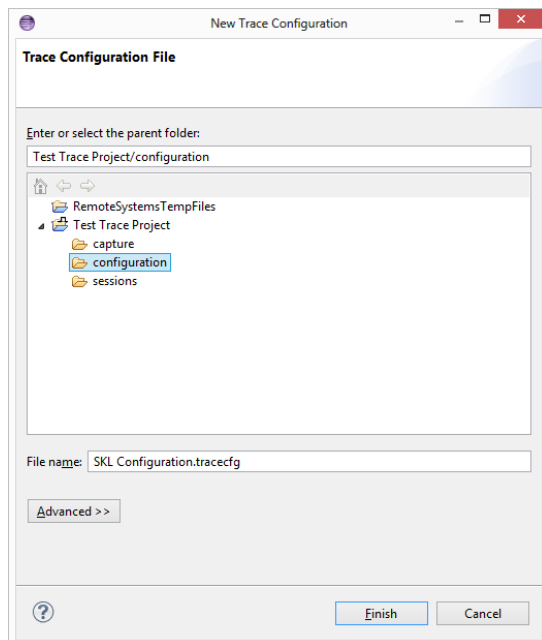


6.6 Select Trace Sources

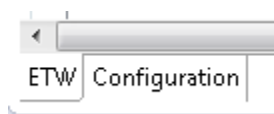
A **trace configuration** needs to be created in order to specify the **trace sources** you want to enable. Right-click on the configuration node in the Project Explorer view and select **New > Trace Configuration**.



Specify a name for the new trace configuration in the **New Trace Configuration** dialog and click on **Finish**.



At the bottom of this editor, you will find different trace source configuration tabs, which provide trace source specific configuration parameters, such as:



Inside these tabs, you can select/configure different trace sources.

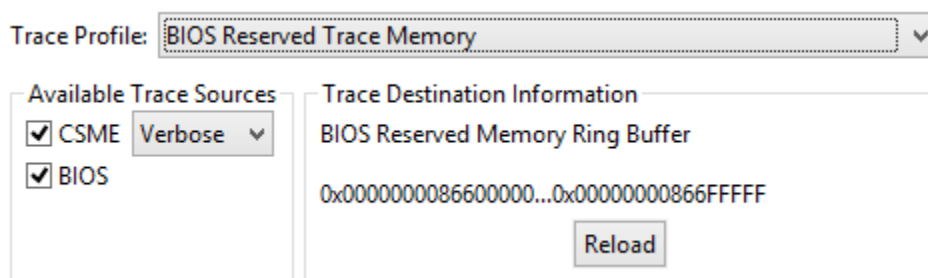
When the configuration is finished, save the file by selecting **File > Save**.

NOTE: It is important that to save your configuration at the end. Otherwise, the System Trace cannot configure the target correctly.

6.6.1 Enabling and Configuring BIOS and CSME traces

You only need to follow the steps in this sub-chapter if you are interested in BIOS and CSME traces.

To enable and configure BIOS and CSME trace sources, go to the **Configuration** tab in the **Configuration** editor and select the BIOS and CSME checkboxes.



CSME tracing verbosity can be set to Verbose or Normal.

Also, select the trace profile. The trace profile specifies where the traces will be sent. The default is Streaming to Intel(R) DCI OOB.

Save the configuration by selecting **File > Save**.

6.6.2 Enabling and Configuring Architectural Event Traces (AET)

You only need to follow the steps in this sub-chapter if you are interested in AET traces.

AET gives the user core level P- and C-state transitions. It also provides visibility into all of the power management MSRs (e.g. energy and perf bias) and MWAIT requests. Therefore, AET allows the user to see what the OS is doing to the system behind the scenes. Since the cores are down during package level c-states, no visibility is provided there. Also included are interrupt/iret and exception/fault/trap tracing, so the user can see e.g. what events are influencing performance, waking cores from sleep, etc.



Refer to the AET introduction slide set to get more information about the AET technology. The slide set can be found in
<INSTALLDIR>\documentation_2018\en\debugger\iss2018\system_debugger\system_trace\ aet_introduction.pdf

To enable AET traces please do the following:

- 1) Check the **Architectural Event Trace** checkbox in the main configuration panel, like described in chapter 6.6, Select Trace Sources:

Trace Profile: (none) ▼

Available Trace Sources

- ☐ CSME Verbose ▼
- ☐ BIOS
- ☒ Architectural Event Trace

Trace Destination Information

Streaming Mode

Intel(R) DCI OOB

- 2) Select the AET tab on the bottom of the configuration editor:

Source & Destination AET ETW

- 3) If the **AET** tab was selected, the AET configuration should show up:

☒ Enable AET

Config 0

☐ Enable RIC ☒ Enable LIP

| Event | Description | Enable | LBR |
|-----------|-------------------------------|--------------------------|--------------------------|
| HW_INTR | HW Interrupt trace enable | <input type="checkbox"/> | <input type="checkbox"/> |
| IRET | IRET trace enable | <input type="checkbox"/> | <input type="checkbox"/> |
| EXCEPTION | Exception trace enable | <input type="checkbox"/> | <input type="checkbox"/> |
| MSR | MSR trace enable | <input type="checkbox"/> | <input type="checkbox"/> |
| POWER | Power management trace enable | <input type="checkbox"/> | <input type="checkbox"/> |
| IO | IO trace enable | <input type="checkbox"/> | <input type="checkbox"/> |
| SGX | Enclave event enable | <input type="checkbox"/> | <input type="checkbox"/> |
| WBINVD | WBINVD trace enable | <input type="checkbox"/> | <input type="checkbox"/> |
| SMI | SMI trace enable | <input type="checkbox"/> | <input type="checkbox"/> |
| MWAIT | MONITOR/MWAIT trace enable | <input type="checkbox"/> | <input type="checkbox"/> |

Reset Defaults

Thread Assignment

- ☐ TARGET
 - ☐ PKG 0
 - ☐ C0
 - ☐ T0 [P0]
 - ☐ T1 [P1]
 - ☐ C1
 - ☐ T0 [P2]
 - ☐ T1 [P3]

New Configuration Clone Configuration



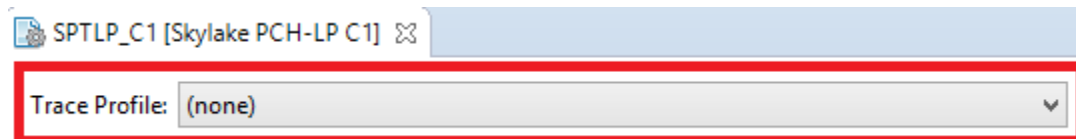
Please make sure that the **Enable AET** checkbox is selected. In the AET configuration panel, you can turn on or turn off specific AET events.

NOTE: If you stream via Intel(R) DCI OOB, please note that the bandwidth is limited. When enabling all AET events in the configuration panel, AET might generate a high volume of data, which might lead to errors in STT, due to the maximum available Intel(R) DCI OOB bandwidth.

- 4) Save your configuration by clicking the save icon in the main Eclipse toolbar or use CTRL-s.
- 5) Start Live Tracing, like described in chapter 6.9

6.7 Select Trace Destination

In the trace configuration editor, you can select the trace destination by selecting a Trace Profile:



The default is Streaming to Intel(R) DCI OOB. Each trace destination has different implications regarding the trace scenario that is being run. **It is important to understand these implications before selection a destination.** The following options are available:

- **Streaming to Intel(R) DCI OOB**
This streaming profile setting writes all trace events to the Intel(R) Direct Connect Interface (Intel(R) DCI) USB port that is also used for connecting and configuring the target system. This profile supports tracing through reboots of the target system. This profile implies that the trace tool is connected to the DUT over Intel(R) DCI OOB at all times. The data bandwidth of the trace is limited by the USB data path. Too high data rates can result in loss of trace events. Try using BIOS allocated memory if this happens
- **BIOS Reserved Trace Memory**
The BIOS reserved memory buffer profile requires BIOS setup for memory allocation and programming of the Intel® Trace Hub destination addresses. Review the Trace Hub section in the BIOS setup for reserving trace memory. This method is only available after BIOS has booted and cannot be used to analyze reset issues, as the memory gets reset and cleared on a target reset as well. The benefits of using this method are the virtually unlimited data bandwidth and the fact that a system can collect traces without an active connection to the trace tool. A connection to the



trace tool is only needed when the trace is captured from the memory buffer.

- **Intel(R) Trace Hub Memory**

The Intel® Trace Hub has a small on-chip memory buffer that is always accessible. Using this profile supports tracing through reboots of the target system. The main benefit of this profile is the analysis of early boot failures of a platform or a fallback when other destinations cannot be used due to hardware issues (for example no system memory or USB access). However, since this buffer is small, it can only hold a few trace events before the older ones get overwritten.

Once you selected the trace destination, save the configuration by selecting **File > Save**.

6) Starting Live Tracing.

NOTE: When analyzing the live traces, you might want to switch on the AET specific columns. To do this open the **Message View Select Columns** dialog, by clicking on the following icon in the Message View toolbar:

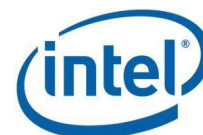


You will find a section called AET in the column tree, where you can enable columns you are interested in. For more information about Message View columns, please refer to chapter 7.5

Basic Trace Analysis: Selecting Trace Fields to be Displayed.

The following table describes AET events, which can be enabled including references to the Intel® 64 and IA-32 Architectures Software Developer's Manual for more information. The document can be downloaded here: <http://www.intel.com/content/www/us/en/architecture-and-technology/64-ia-32-architectures-software-developer-manual-325462.html>

| | |
|------------------|---|
| HW_INTR | Enable events triggered by hardware interrupts, those are interrupts #32..255 across all IA processors in the system. Please refer to the Intel(R) Software Development Manual, volume 3, chapter 6 (Interrupt and Exceptions Handling) |
| IRET | Enable events triggered by execution of the Interrupt Return Events (IRET) instruction. This is used to return from interrupt/exception/fault handler. Please refer to the Intel(R) Software Development Manual, volume 2, chapter 3 (Instruction Set Reference, A-M) |
| EXCEPTION | Enable events triggered by Exceptions or INTn. Exceptions are the first 32-interrupt vectors, which are reserved by Intel to report specific instruction execution error conditions like page-faults. Please refer to the Intel(R) Software Development Manual, volume 3, chapter 6 (Interrupt and Exceptions Handling) |



| | |
|---------------|---|
| MSR | Enable events triggered by CPU Model Specific Register (MSR) read/write accesses using RDMSR/WRMSR instructions. Please refer to the Intel(R) Software Development Manual, volume 3, chapter 35 (Model Specific Registers) and volume 2, chapter 4 (Instruction Set Reference, N-Z) |
| POWER | Enables events triggered by thread-level C-state activity. Please refer to the Intel(R) Software Development Manual, volume 3, chapter 14 (Power and Thermal Management) |
| IO | Enable events of data transfers to and from input/output port(s) issued by the IN and OUT instructions. Please refer to the Intel(R) Software Development Manual, volume 1, chapter 17 (Input/Output) |
| SGX | Enable events triggered by Intel(R) Software Guarded Extensions (Intel(R) SGX) entries and exits. Please refer to the Intel(R) Software Development Manual, volume 3, chapter 37 (Introduction to Intel(R) Software Guard Extensions) |
| WBINVD | Enable Write Back and Invalidate Cache events. Please refer to the Intel(R) Software Development Manual, volume 2, chapter 4 (Instruction Set Reference, N-Z) |
| SMI | Enable System Management Interrupt (SMI) events of entries and exits from System Management Mode (SMM). Please refer to the Intel(R) Software Development Manual, volume 3, chapter 34 (System Management Mode) |
| MWAIT | Enable Monitor Wait (MWAIT) events, e.g. triggered by OS requests for lower power states. Please refer to the Intel(R) Software Development Manual, volume 2, chapter 3 (Instruction Set Reference, A-M) or volume 3, chapter 8.10.4 (MONITOR/MWAIT Instruction) |

6.7.1 Enabling and Configuring ETW traces

You only need to follow the steps in this sub-chapter if you are interested in ETW traces.

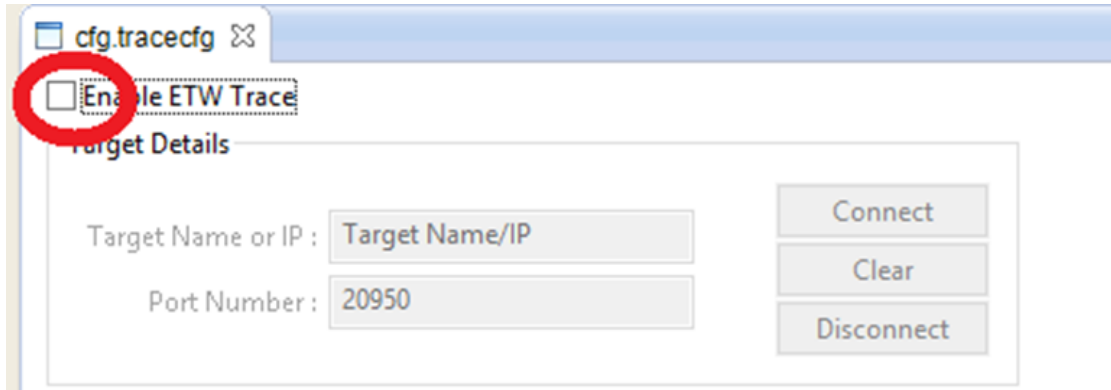
ETW (Event Tracing for Windows) is the event tracing mechanism, available on windows for capturing trace logs from drivers and applications. ETW can be configured in the ETW Panel.

NOTE: For capturing ETW traces, the target needs to be setup first. To do so, install the Intel® Trace Hub Driver, which is part of the Intel release drivers, which can be downloaded here:
<https://platformsw.intel.com>

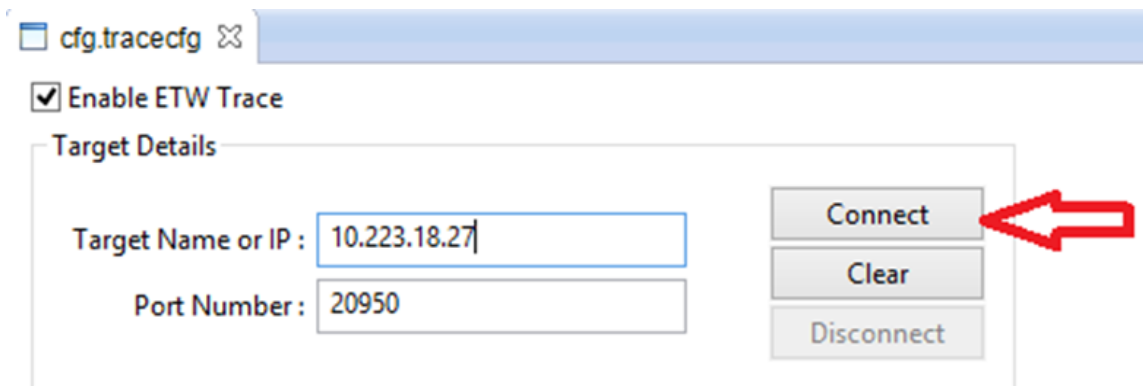


The setup is described in chapter Setting up the Target for Event Trace for Windows (ETW) Tracing.

Choose the ETW Tab in the STT configuration editor for configuring ETW traces.



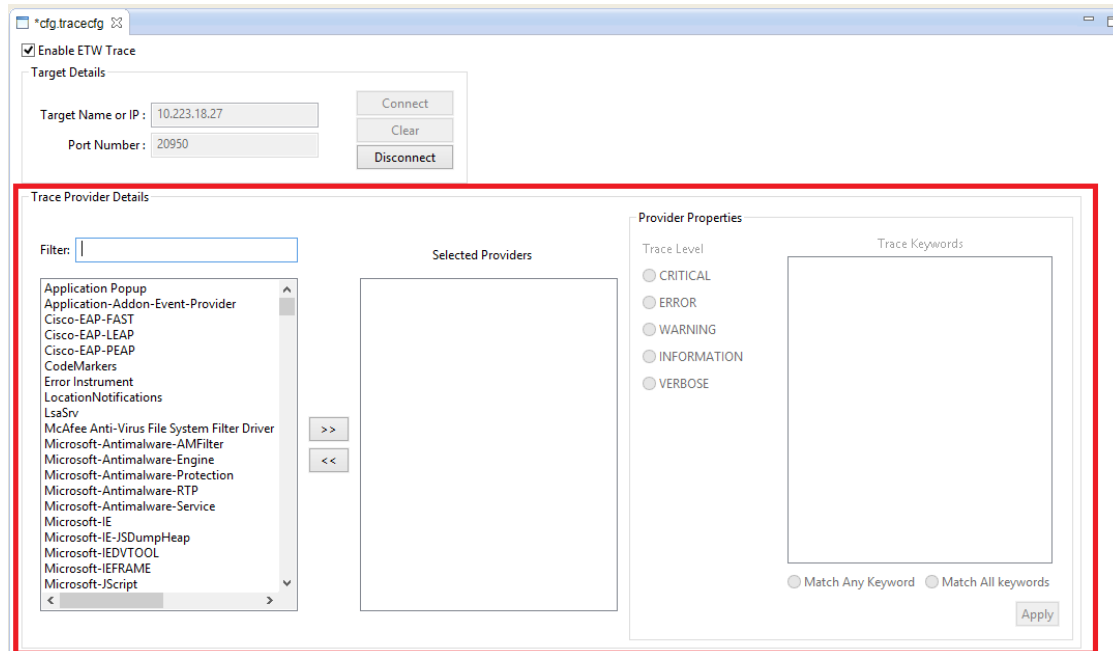
Select the Enable **ETW Trace** checkbox.



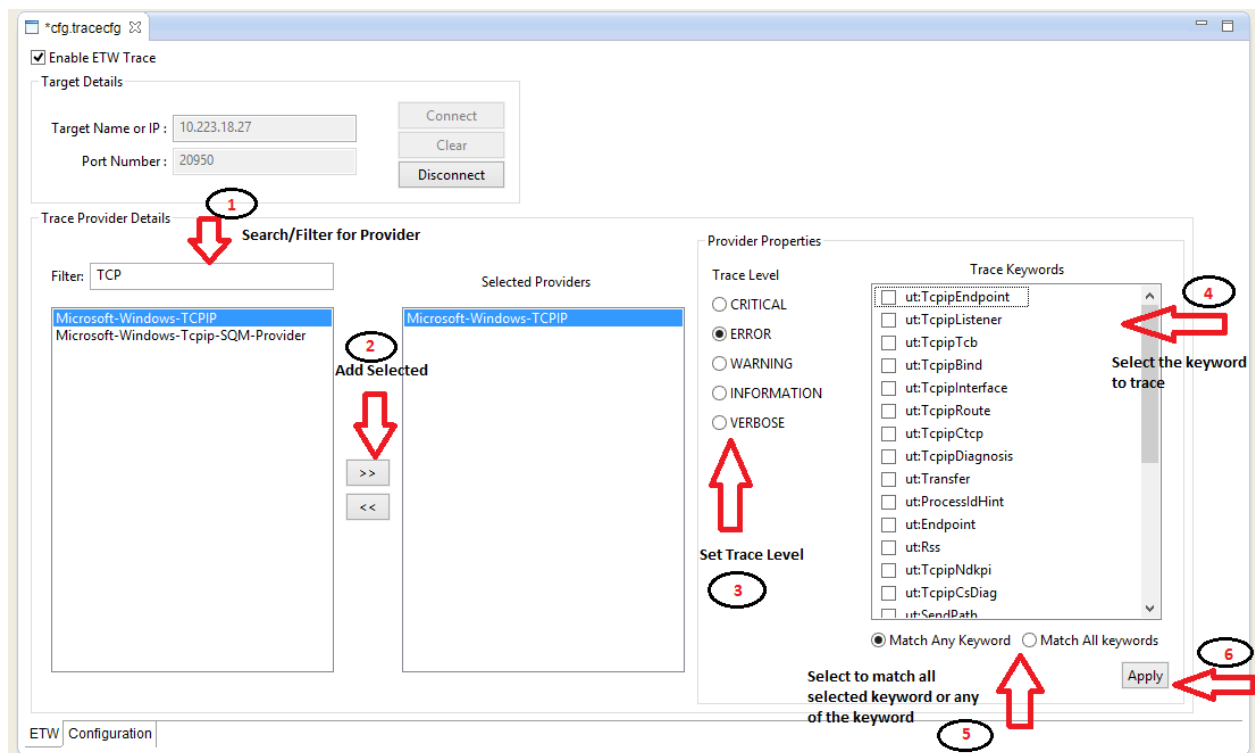
Enter the IP address or DNS name of the target for connecting to do the trace configuration. Click on connect. The target will listen to TCP port 20950 by default.

If you want to use a different port then you should change the target side port as well. Please follow the IntelTraceHub Driver installation document.

When the connection to the target is established, the view lists available ETW trace providers. Select the trace providers you are interested in.



Follow the steps as displayed in the image below. When you are done with selecting and configuring a particular trace source, you can repeat the steps in order to select another trace source.

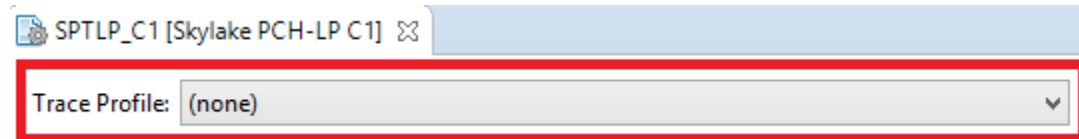




Save the configuration by selecting **File > Save**.

6.8 Select Trace Destination

In the trace configuration editor, you can select the trace destination by selecting a Trace Profile:



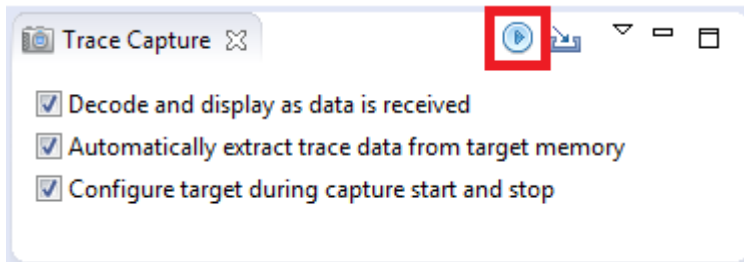
The default is Streaming to Intel(R) DCI OOB. Each trace destination has different implications regarding the trace scenario that is being run. **It is important to understand these implications before selection a destination.** The following options are available:

- **Streaming to Intel(R) DCI OOB**
This streaming profile setting writes all trace events to the Intel(R) Direct Connect Interface (Intel(R) DCI) USB port that is also used for connecting and configuring the target system. This profile supports tracing through reboots of the target system. This profile implies that the trace tool is connected to the DUT over Intel(R) DCI OOB at all times. The data bandwidth of the trace is limited by the USB data path. Too high data rates can result in loss of trace events. Try using BIOS allocated memory if this happens
- **BIOS Reserved Trace Memory**
The BIOS reserved memory buffer profile requires BIOS setup for memory allocation and programming of the Intel® Trace Hub destination addresses. Review the Trace Hub section in the BIOS setup for reserving trace memory. This method is only available after BIOS has booted and cannot be used to analyze reset issues, as the memory gets reset and cleared on a target reset as well. The benefits of using this method are the virtually unlimited data bandwidth and the fact that a system can collect traces without an active connection to the trace tool. A connection to the trace tool is only needed when the trace is captured from the memory buffer.
- **Intel(R) Trace Hub Memory**
The Intel® Trace Hub has a small on-chip memory buffer that is always accessible. Using this profile supports tracing through reboots of the target system. The main benefit of this profile is the analysis of early boot failures of a platform or a fallback when other destinations cannot be used due to hardware issues (for example no system memory or USB access). However, since this buffer is small, it can only hold a few trace events before the older ones get overwritten.

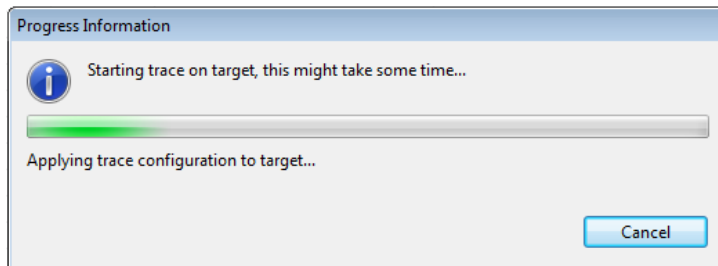
Once you selected the trace destination, save the configuration by selecting **File > Save**.

6.9 Starting Live Tracing

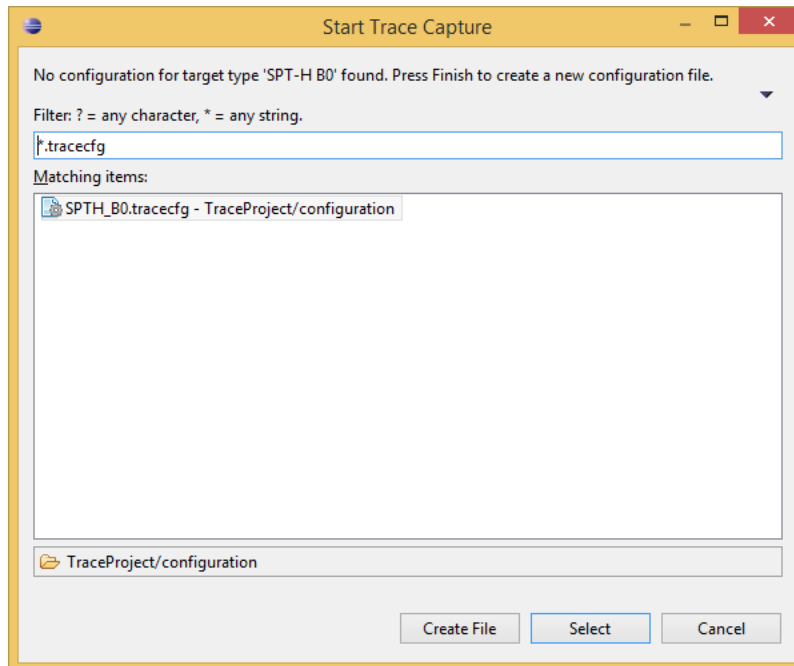
After completing the steps in the previous chapters, the **Trace Capture** view enables the **Begin capturing trace data from target button**, like shown below:



Click on the **Begin capturing trace data from target** button to start the trace capture process. The following loading screen appears:



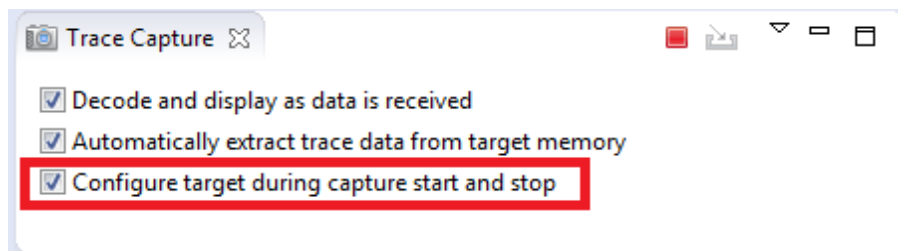
In case no trace configuration was opened before, the following dialog pops up after clicking the **Begin capturing trace data from target** button:



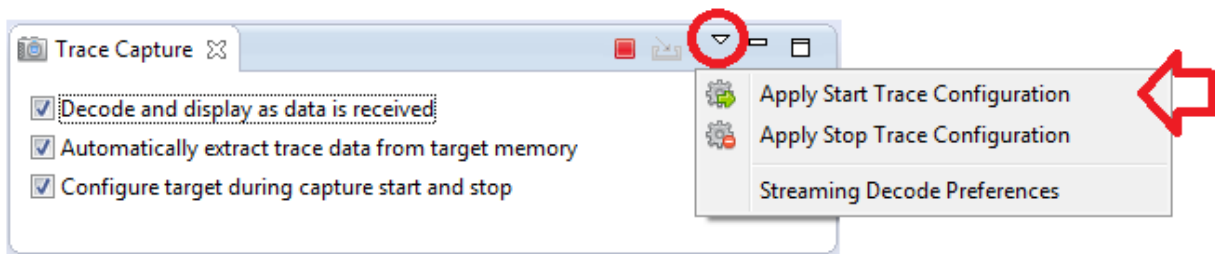
Choose the trace configuration you want to apply and click **Select**. If you do not have a trace configuration file click **Create File** to create a one.

6.9.1 Start Capture without Configuration

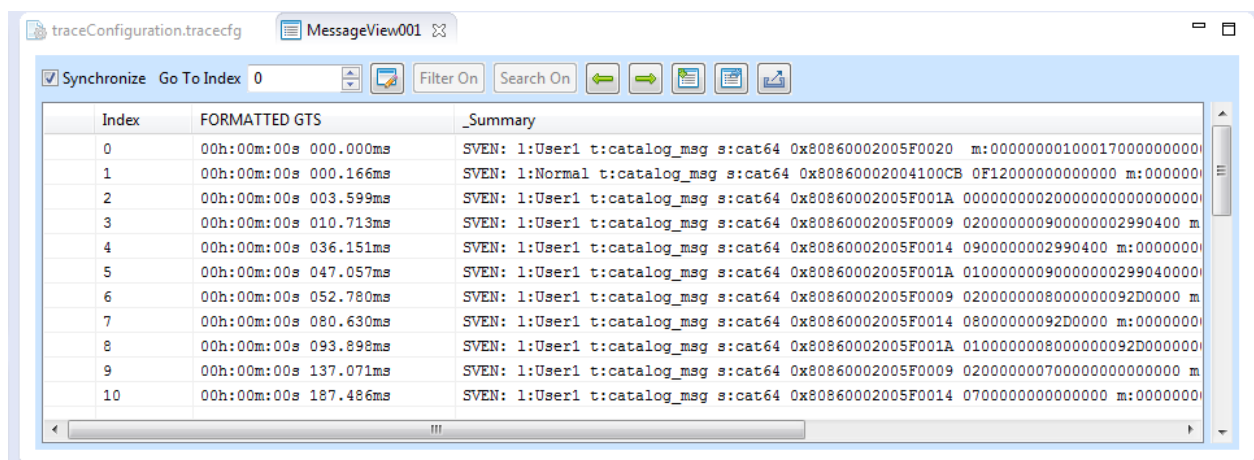
By default, the trace viewer configures the Intel® Trace Hub hardware for tracing and starts the capture of trace data at the same time. This overrides any existing trace configuration. To capture just the data without changing the hardware configuration setup earlier or by some other tool, uncheck this option in the **Trace Capture** view:



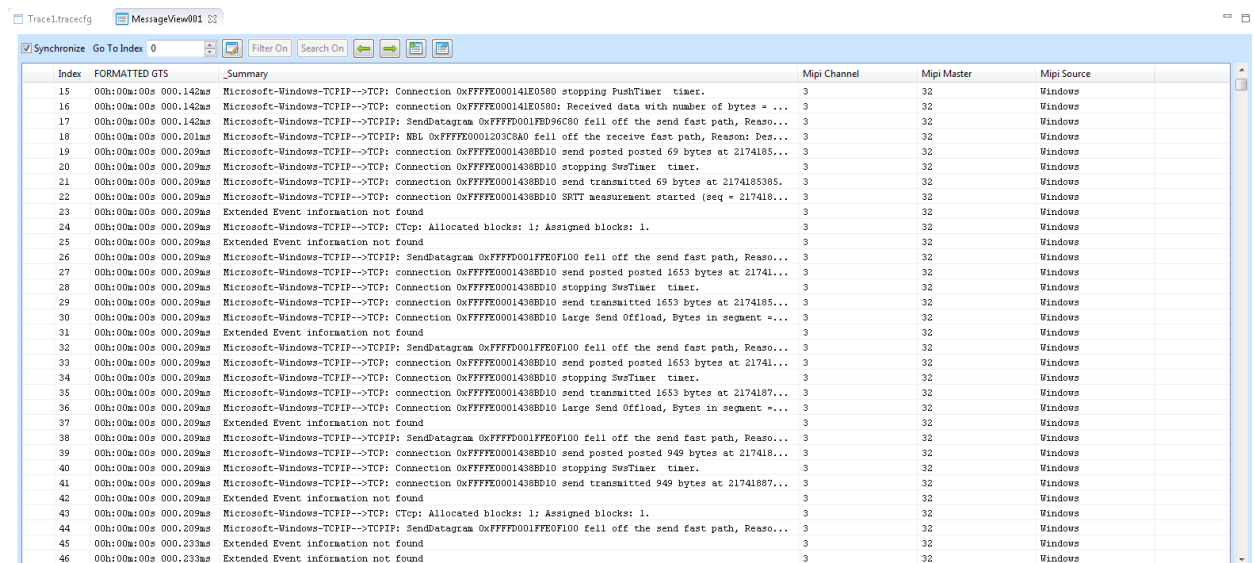
Note: You can still apply the start capture configuration settings manually during a running capture using the context menu of the Trace Capture view.



After completion of the connection procedure, trace messages received from the target are displayed in the system trace, for example, for BIOS and CSME traces.

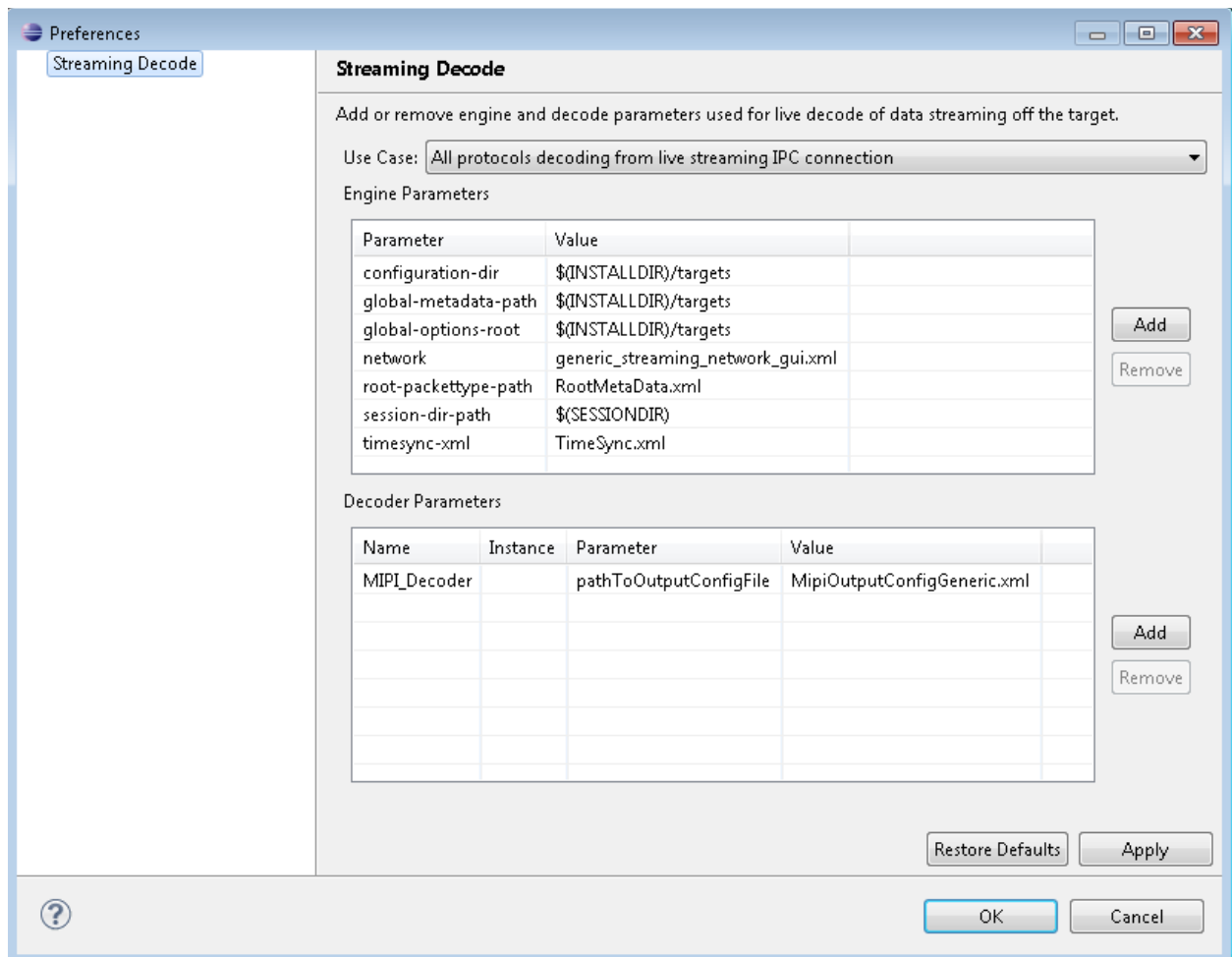
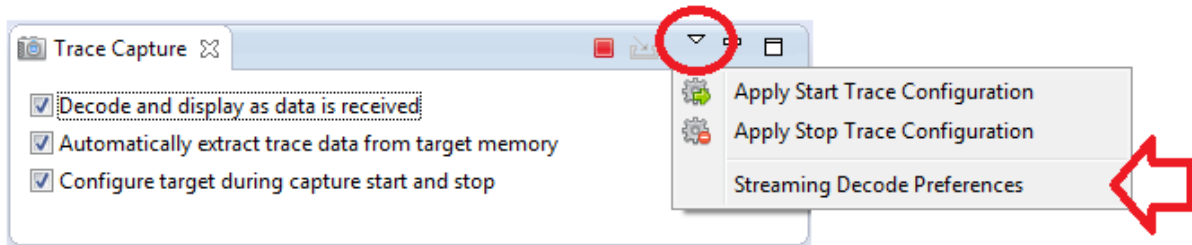


Example for ETW traces:





If you want to change the decoder parameter for streaming, open the streaming decode preferences dialog and set your preferred options.

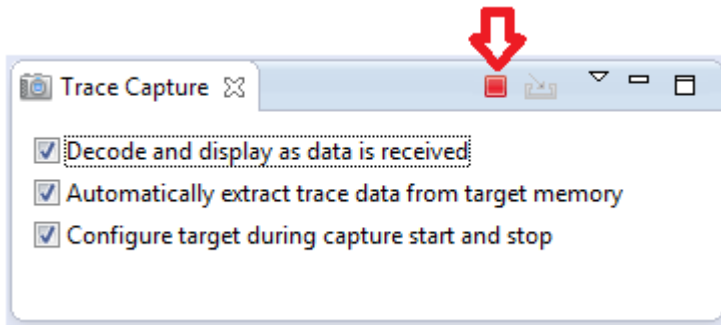


Close the dialog with **OK** and start live tracing.

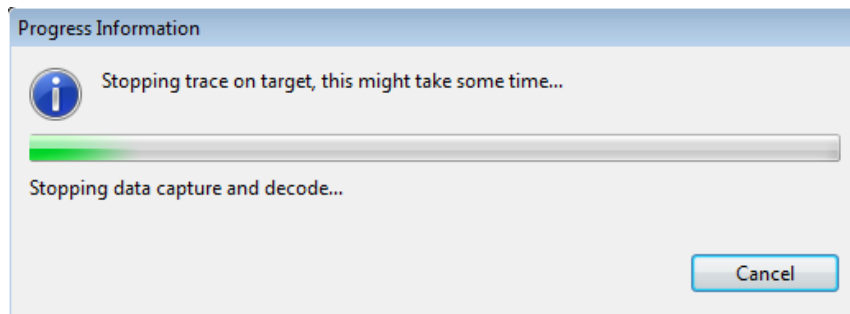
NOTE: Changing streaming parameters is an expert setting and requires detailed knowledge about decoders.

6.10 Stop Live Tracing

To stop the live tracing, click the **Stop capturing** button in the Trace Capture view.

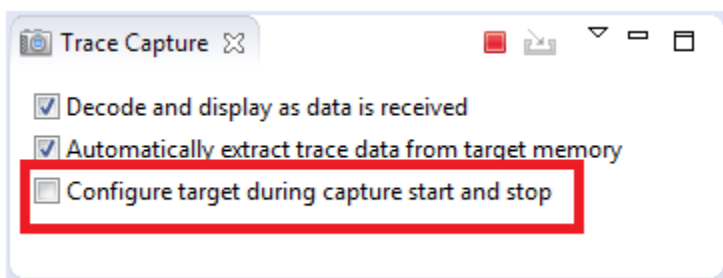


The System Trace stops the live streaming.



6.10.1 Stop Capture without Configuration

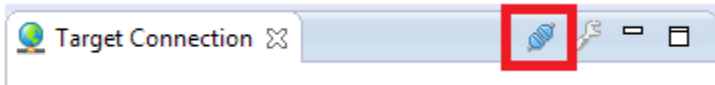
By default, the trace viewer configures the Intel® Trace Hub hardware to stop tracing and the capture of trace data at the same time. This overrides any existing trace configuration. To avoid changing the hardware configuration on capture actions, disable this option in the **Trace Capture** view:





6.11 Disconnect from Target

To disconnect from a target, click the **Disconnect** button in the **Target Connection** view:



The System Trace then will be disconnected from the configured target.

6.12 Hardware Setup and Configuration

Troubleshooting:

| Symptom | Solution(s) |
|--|---|
| <i>The connection from the CCA adapter to the target using USB3 is not possible. The connection procedure runs into a timeout.</i> | <ol style="list-style-type: none">1) Use a USB2 (non-SuperSpeed) Type A to Type B cable to connect the CCA to the host.2) The CCA does not use the USB protocol to talk to the target, instead it uses the Intel(R) DCI OOB (formerly known as Boundary Scan Sideband Interface (BSSB) protocol), which uses both single-ended and differential signaling. It also uses the SuperSpeed wires in a USB cable to transmit the Intel(R) DCI OOB. Therefore, an off-the-shelf USB cable will likely not work. Please be sure to use the ~9" USB SS cable that shipped with the CCA.3) Connect the CCA to the target only using USB3 |
| <i>The connection using the connect button in the System Trace does not work.</i> | <p>Some target steppings (SPT A0, and likely B0) require the following procedure:</p> <ol style="list-style-type: none">1) Turn off the target2) Connect the CCA adapter to the target and the host.3) Click connect in the System Trace panel4) Wait 3-4 seconds, then switch on the board |
| <i>The CCA adapter will not be recognized by the Windows operating system.</i> | <p>Make sure the Intel(R) Direct Connect Interface (Intel(R) DCI) drivers are installed probably. Please check the Windows device manager if you find a</p> |



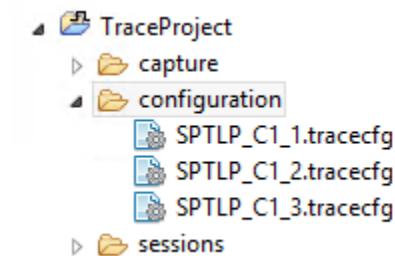
| | |
|--|---|
| | device called <i>Intel USB CCA Debug Class Devices</i> . |
| <i>The connection to the target takes long time and the fails with a timeout message</i> | <p>If the Intel® Dfx Abstraction Layer library (DAL) detects an old firmware on the CCA, the firmware will be updated by DAL automatically during the first connect to the target. In some cases the connect procedure can run into a timeout and display error messages on the Eclipse* console. The firmware will only be updated at the first connect and when an old firmware is detected. All subsequent connects should run smooth.</p> <p>If you ran into this timeout, please do the following:</p> <ol style="list-style-type: none"> 1) Close Eclipse* 2) Unplug the CCA adapter 3) Kill the DAL MasterFrame process (MasterFrame.HostApplication.exe) using the Windows* task manager. 4) Plug-in the CCA adapter 5) Start Eclipse* and follow the connect procedure described in the user guide again. |
| <i>No trace events appear</i> | <ol style="list-style-type: none"> 1) Check if the configuration has the necessary trace sources enabled 2) Check if "Configure target during capture start and stop" was disabled by mistake. The viewer skips any trace configuration if this option is unchecked 3) Chose "Apply Start Trace Configuration" from the Trace capture context menu |
| <i>ETW: Error in Target Connection. Network issues.</i> | Make sure the target is connected to the network and has valid IP address. Make sure you can ping the target from the host and vice versa. Make sure the ETW service is running on target with a valid port (e.g. default 20950). |
| <i>ETW:Error in Start Trace Capture</i> | Make sure that the Intel® Trace Hub device is displayed in Windows Device Manager. If it is not displayed, set the configuration in BIOS to enable the Trace Hub device. |



6.13 Exporting Trace Configuration

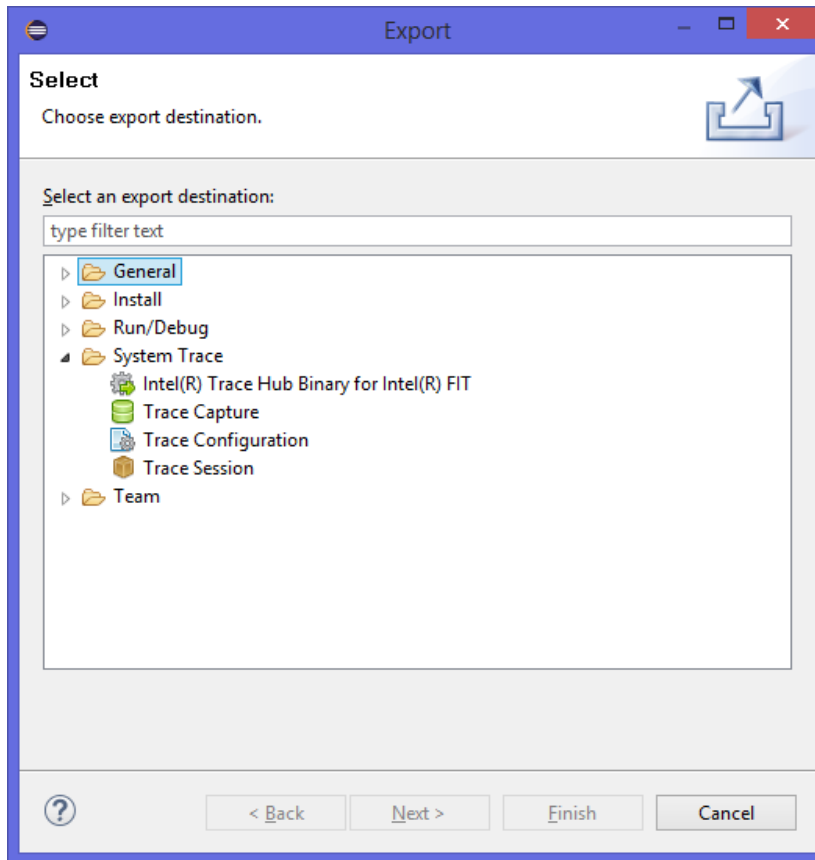
The Trace Configuration represents selected trace sources and trace destination described in sections: 6.6 and 6.7. The current state of the selected sources and trace destination is saved within the **Trace Configuration**. The Trace Configuration can be exported into a specified folder in two ways: exporting the Trace Configuration file (.tracecfg) or exporting it to Intel(R) Trace Hub binary file (.bin). The “Intel(R) Trace Hub Binary for Intel(R) FIT” can be used to set Intel(R) Trace Hub registers from the firmware when the platform boots. For more information on Intel(R) FIT and how to generate a firmware with these settings, contact your Intel representative.

A Project can have multiple configuration files. A project with three Trace Configurations will be used as an example below.



Exporting to a folder can be done in two different ways:

1. Right-click on a .tracecfg file and go to **Export** in the context.
This option is available even if multiple configuration files are selected.
2. Right-click in the project explorer window and then in the context menu click on **Export** to see the following Export wizard.



Under the **System Trace** folder, select preferred export method: Intel(R) Trace Hub Binary for Intel(R) FIT or **Trace Configuration**.

Clicking on **Next** navigates you to **Export** dialog where user must mention the destination folder for the export.

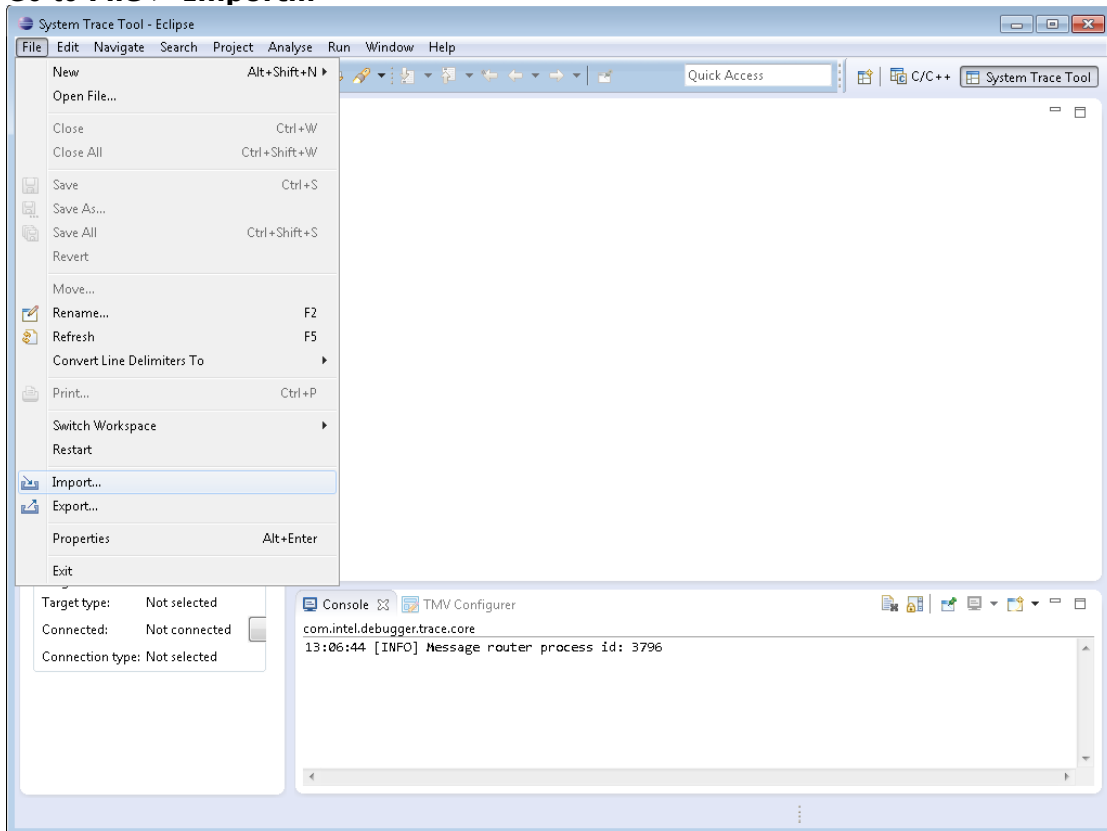
7 Trace Analysis Use-Cases

This chapter describes the most important use-cases for the System Trace and how they can be used.

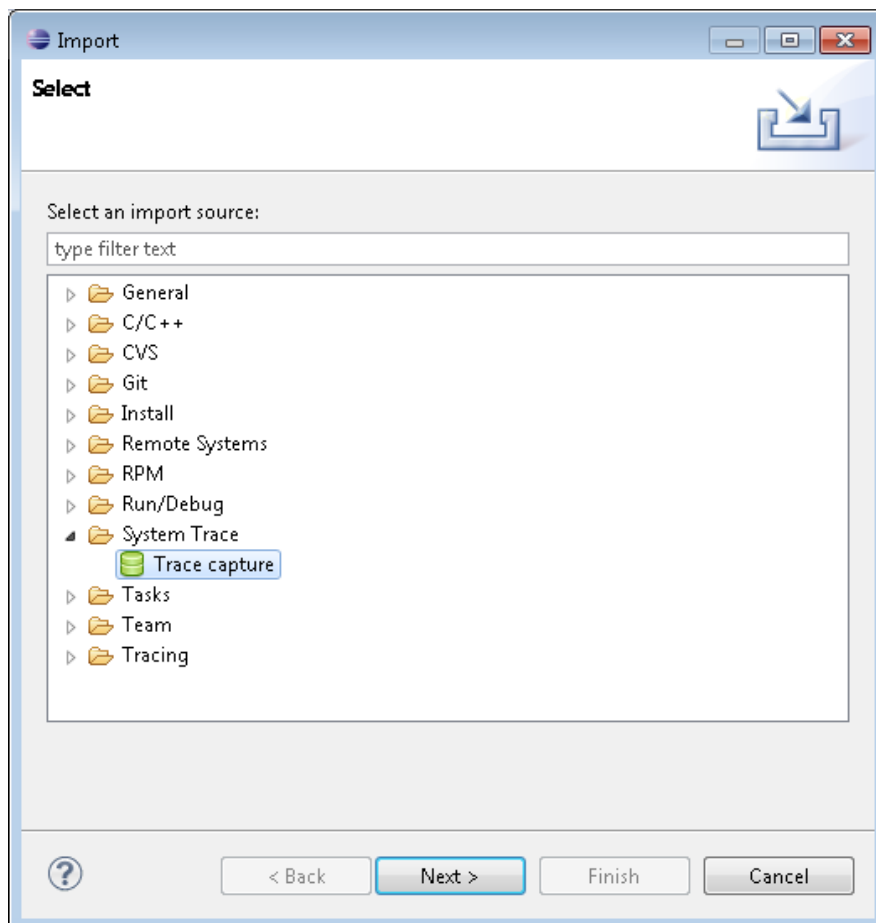
7.1 Importing an Existing Trace Capture Archive/Trace Capture File (bin-file)

To import an existing Trace Capture Archive file, such as, one of the examples, which can be found in `<INSTALLDIR>\system_trace\examples\input`, you need to import such a file into your trace viewer. To import a sample System Validation Events (SVEN) trace included in a Trace Capture Archive or in a plain bin-file:

1. Open System Trace integrated in Eclipse*.
2. Switch to the trace perspective.
3. Go to **File > Import...**



4. Browse to **System Trace > Trace capture**.

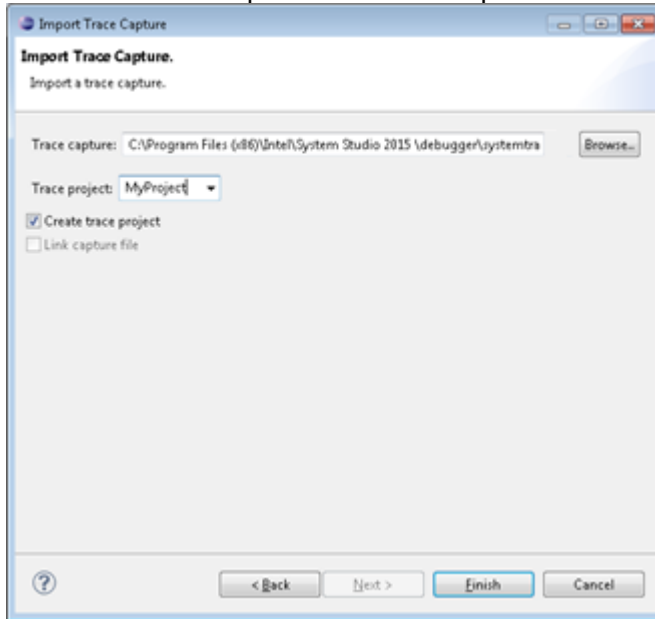


Click **Next**.

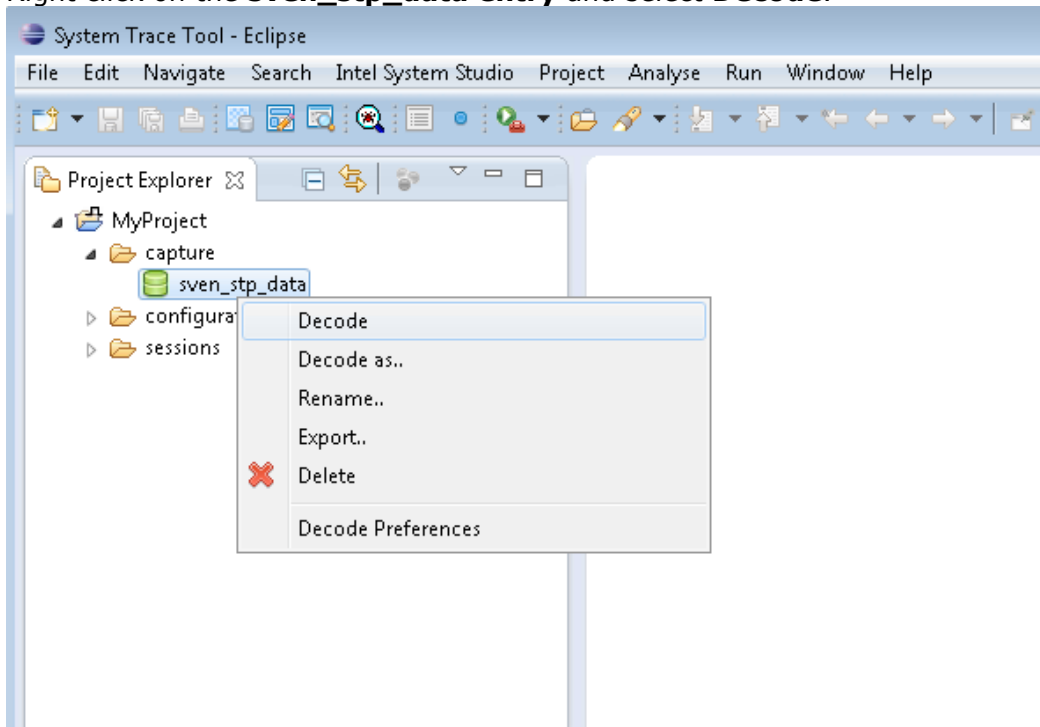
5. Select the Trace Capture Archive by clicking **Browse...**
Locate sven_stp_data.tracecpt in the directory
<INSTALLDIR>\system_trace\examples\input
If you want to import a bin file: select the *.* file filter on the bottom right corner of the file browse dialog.
6. Select the Trace Project you want to import the data. If you do not have a project, type in a name for your new project and select **Create trace project**.



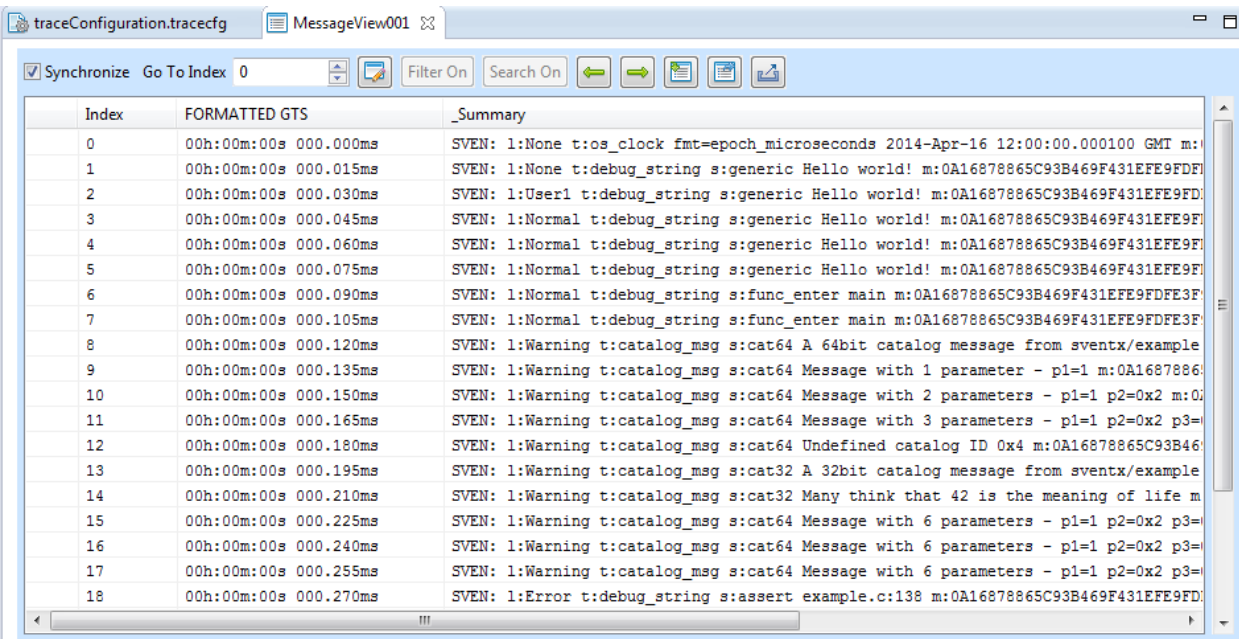
- Click **Finish** to import the Trace Capture Archive.



- In the Project Explorer, expand the **capture** node. You should see a **sven_stp_data** entry.
- Right click on the **sven_stp_data** entry and select **Decode**.



The sample trace is displayed in the trace viewer.



| Index | FORMATTED GTS | _Summary |
|-------|-----------------------|---|
| 0 | 00h:00m:00s 000.000ms | SVEN: 1:None t:os_clock fmt=epoch_microseconds 2014-Apr-16 12:00:00.000100 GMT m: |
| 1 | 00h:00m:00s 000.015ms | SVEN: 1:None t:debug_string s:generic Hello world! m:0A16878865C93B469F431EFE9FDF |
| 2 | 00h:00m:00s 000.030ms | SVEN: 1:User1 t:debug_string s:generic Hello world! m:0A16878865C93B469F431EFE9FD |
| 3 | 00h:00m:00s 000.045ms | SVEN: 1:Normal t:debug_string s:generic Hello world! m:0A16878865C93B469F431EFE9F |
| 4 | 00h:00m:00s 000.060ms | SVEN: 1:Normal t:debug_string s:generic Hello world! m:0A16878865C93B469F431EFE9F |
| 5 | 00h:00m:00s 000.075ms | SVEN: 1:Normal t:debug_string s:generic Hello world! m:0A16878865C93B469F431EFE9F |
| 6 | 00h:00m:00s 000.090ms | SVEN: 1:Normal t:debug_string s:func_enter main m:0A16878865C93B469F431EFE9FDFE3F |
| 7 | 00h:00m:00s 000.105ms | SVEN: 1:Normal t:debug_string s:func_enter main m:0A16878865C93B469F431EFE9FDFE3F |
| 8 | 00h:00m:00s 000.120ms | SVEN: 1:Warning t:catalog_msg s:cat64 A 64bit catalog message from sventx/example |
| 9 | 00h:00m:00s 000.135ms | SVEN: 1:Warning t:catalog_msg s:cat64 Message with 1 parameter - p1=1 m:0A1687886 |
| 10 | 00h:00m:00s 000.150ms | SVEN: 1:Warning t:catalog_msg s:cat64 Message with 2 parameters - p1=1 p2=0x2 m:0 |
| 11 | 00h:00m:00s 000.165ms | SVEN: 1:Warning t:catalog_msg s:cat64 Message with 3 parameters - p1=1 p2=0x2 p3= |
| 12 | 00h:00m:00s 000.180ms | SVEN: 1:Warning t:catalog_msg s:cat64 Undefined catalog ID 0x4 m:0A16878865C93B46 |
| 13 | 00h:00m:00s 000.195ms | SVEN: 1:Warning t:catalog_msg s:cat32 A 32bit catalog message from sventx/example |
| 14 | 00h:00m:00s 000.210ms | SVEN: 1:Warning t:catalog_msg s:cat32 Many think that 42 is the meaning of life m |
| 15 | 00h:00m:00s 000.225ms | SVEN: 1:Warning t:catalog_msg s:cat64 Message with 6 parameters - p1=1 p2=0x2 p3= |
| 16 | 00h:00m:00s 000.240ms | SVEN: 1:Warning t:catalog_msg s:cat64 Message with 6 parameters - p1=1 p2=0x2 p3= |
| 17 | 00h:00m:00s 000.255ms | SVEN: 1:Warning t:catalog_msg s:cat64 Message with 6 parameters - p1=1 p2=0x2 p3= |
| 18 | 00h:00m:00s 000.270ms | SVEN: 1:Error t:debug_string s:assert example.c:138 m:0A16878865C93B469F431EFE9FD |

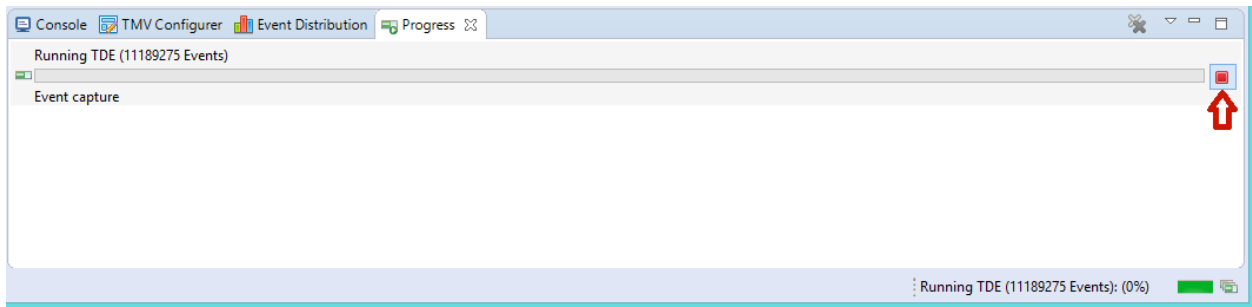
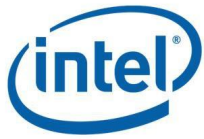
7.2 Stopping an Ongoing Decode

Large capture files may take some time to decode completely. Following steps show you how to stop a long running decode.

1. Open the task in the lower right corner while a decode is ongoing.



2. Press the stop button to cancel the decode.



7.3 Trace Hardware Tests

The **Trace Hardware Tests** functionality can be used to make sure the target connection is working and basic features on the target are working correctly to be able to start tracing.

Trace Hardware Tests can be launched by connecting to the target using the connect icon in the **Target Connection** view:

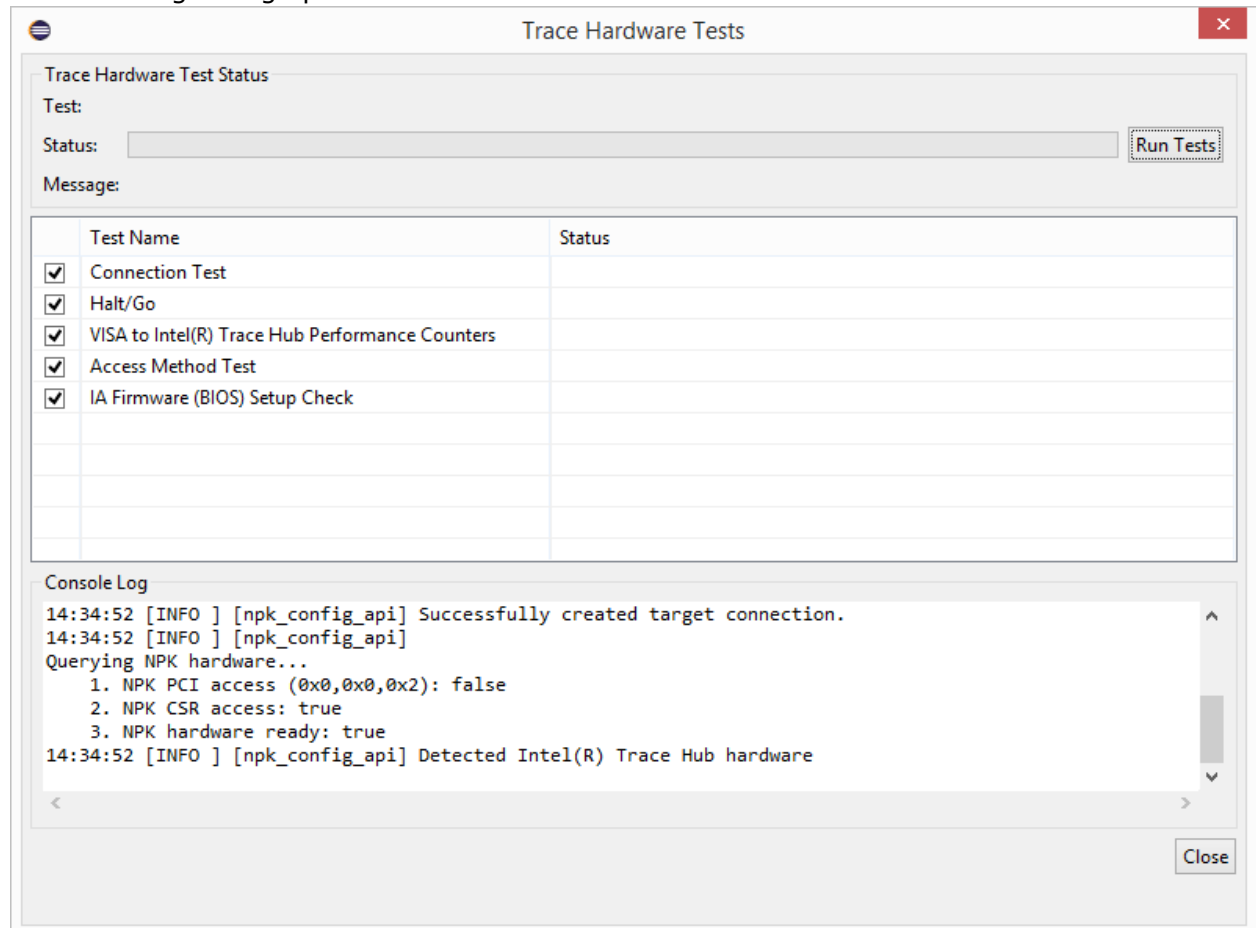


Once you are connected, click on the **Trace Hardware Tests** icon in the **Target Connection View**:





The following dialog opens:



The dialog lists all the tests, which are currently available. These are:

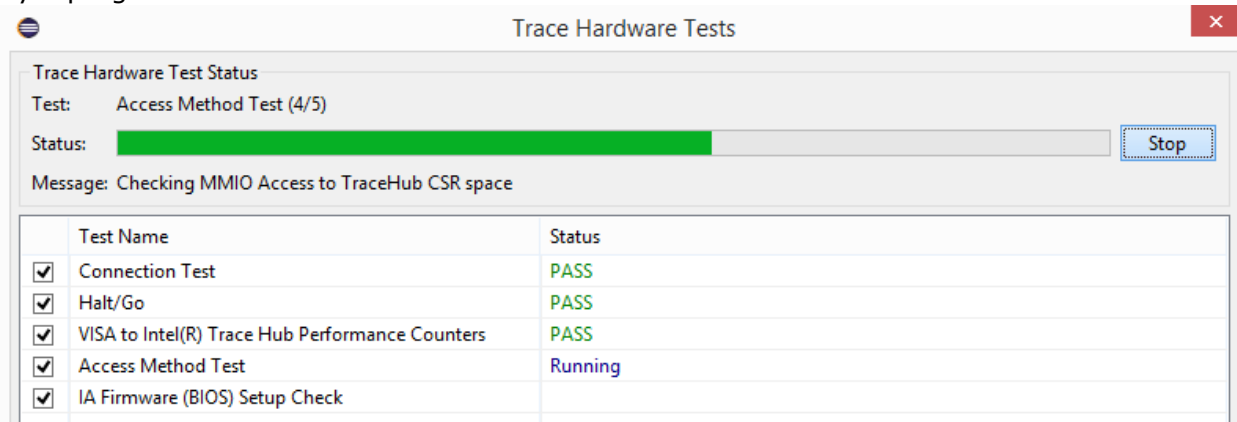
- **Connection Test**
Checks that the connection to the probe is available.
- **Halt/Go**
Checks that the platform can halt/go correctly. An error here indicates that some sources, e.g. AET, may not be configurable.
- **VISA to Intel(R) Trace Hub Performance Counters**
This is the Intel(R) Trace Hub internal check that verifies the validity of the hardware VISA observation paths to Intel(R) Trace Hub are able to be configured and that performance counters are useable.
- **Access Method Test**
Tests the various access methods that are supported. If an access method fails, it could be that some trace sources may not be configurable.



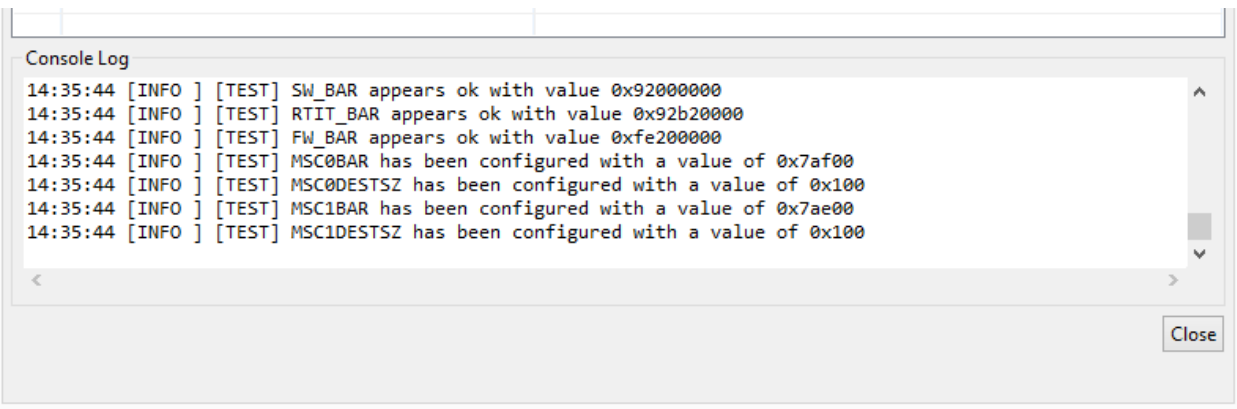
- **IA FW BIOS check**

Check to see that BIOS configured the Intel(R) Trace Hub hardware correctly. If this test fails, it means that BIOS did not setup Intel(R) Trace Hub and the user should enable Intel(R) Trace Hub in the BIOS menus.

To start the tests click on the **Run Tests** button. The status of the test run will be indicated by a progress bar:



The result of the various tests will be shown as status PASS or FAIL. More descriptive status messages during test execution will be available in the **Console Log**:

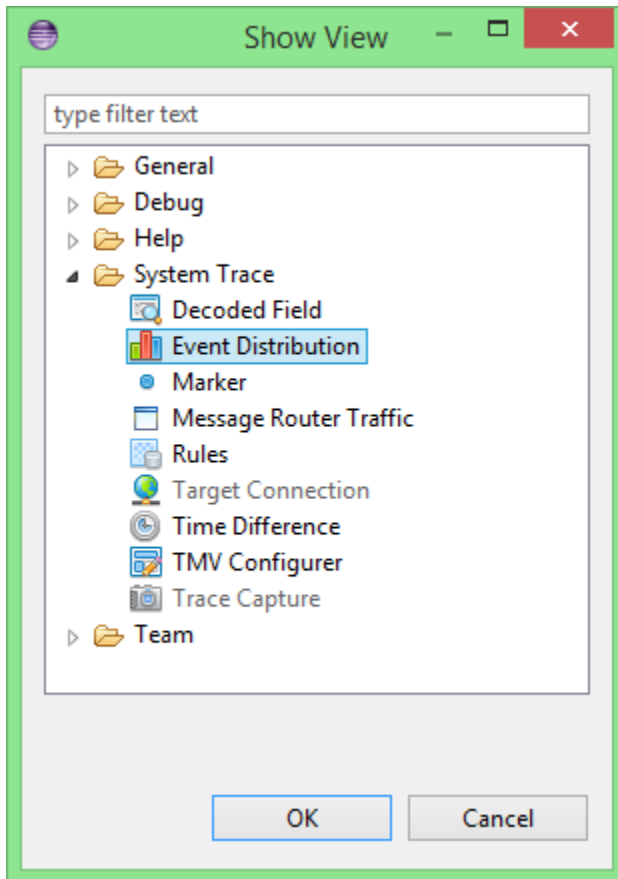


7.4 Navigating a Trace using Graphical Representation of the Event Distribution

The **Event Distribution** view visualizes the distribution of messages over time to provide a better overview of the trace and assist you in identifying interesting regions. This can be a good starting point for trace analysis.

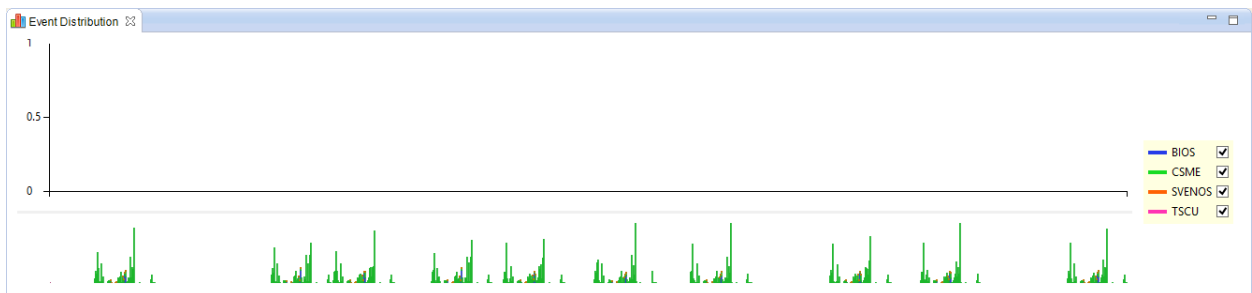
1. Select **Window > Show View > Other...** from the menu bar.

The **Show View** window is opened.



2. Select **System Trace > Event Distribution** and click **OK**.

The **Event Distribution** view is opened.



The **Event Distribution** view displays two graphs on the left and the **Source Selection** (the legend box) on the right.

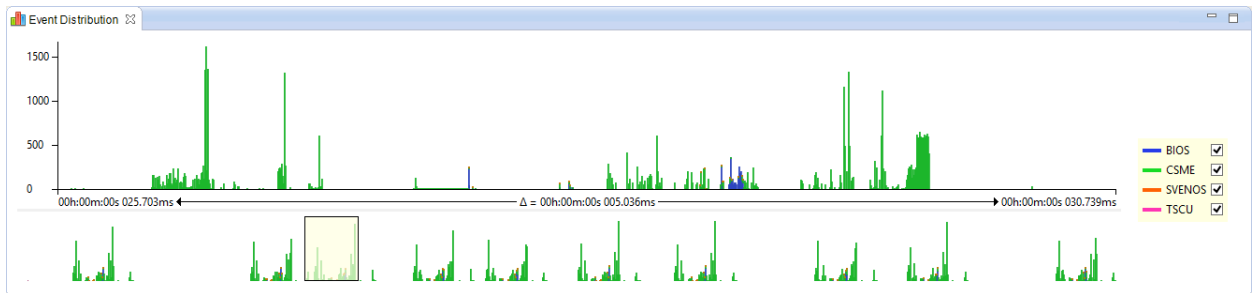
The **Full View** (the lower graph) always shows the full range of the trace and the **Zoomed View** (the upper graph) shows the selected range if set.

The **Source Selection** lists all sources present in the trace with the corresponding color code in plot. Setting the checkbox shows or hides the source in the graphs accordingly.

A point or a pixel on the horizontal axis in the graph represents a time frame known as **Slice** and the value on the vertical axis is the total number of events within that **Slice**.



3. To select a region of the trace to zoom into, left click on the intended starting point in the **Full View** and drag to the intended end point.

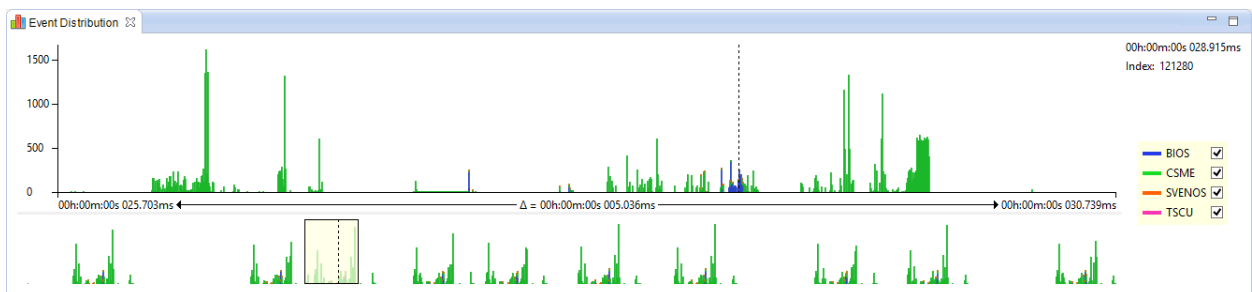


The **Zooming Range Selection** (the semi-transparent yellow box on the **Full View**) represents the portion of the full trace that the **Zoomed View** is showing.

The **Zooming Range Selection** can be updated using the following controls:

- | | |
|-----------------------------|---|
| Select Zooming Range | left mouse click on the Full View and drag to left or right |
| Resize Zooming Range | left mouse click on the left or right edge of the Zooming Range Selection and drag to left or right |
| Zoom In | scroll wheel up up arrow key |
| Zoom Out | scroll wheel down down arrow key |
| Pan Left | left mouse click in the Zooming Range Selection and drag to left middle mouse click on the Full View and drag to left left arrow key |
| Pan Right | left mouse click in the Zooming Range Selection and drag to right middle mouse click on the Full View and drag to right right arrow key |

4. To see more information for a particular **Slice**, hover the mouse pointer over the **Zoomed View** or the **Full View**.



The **Cursor** (the dotted line) is displayed at where the mouse pointer is hovering.

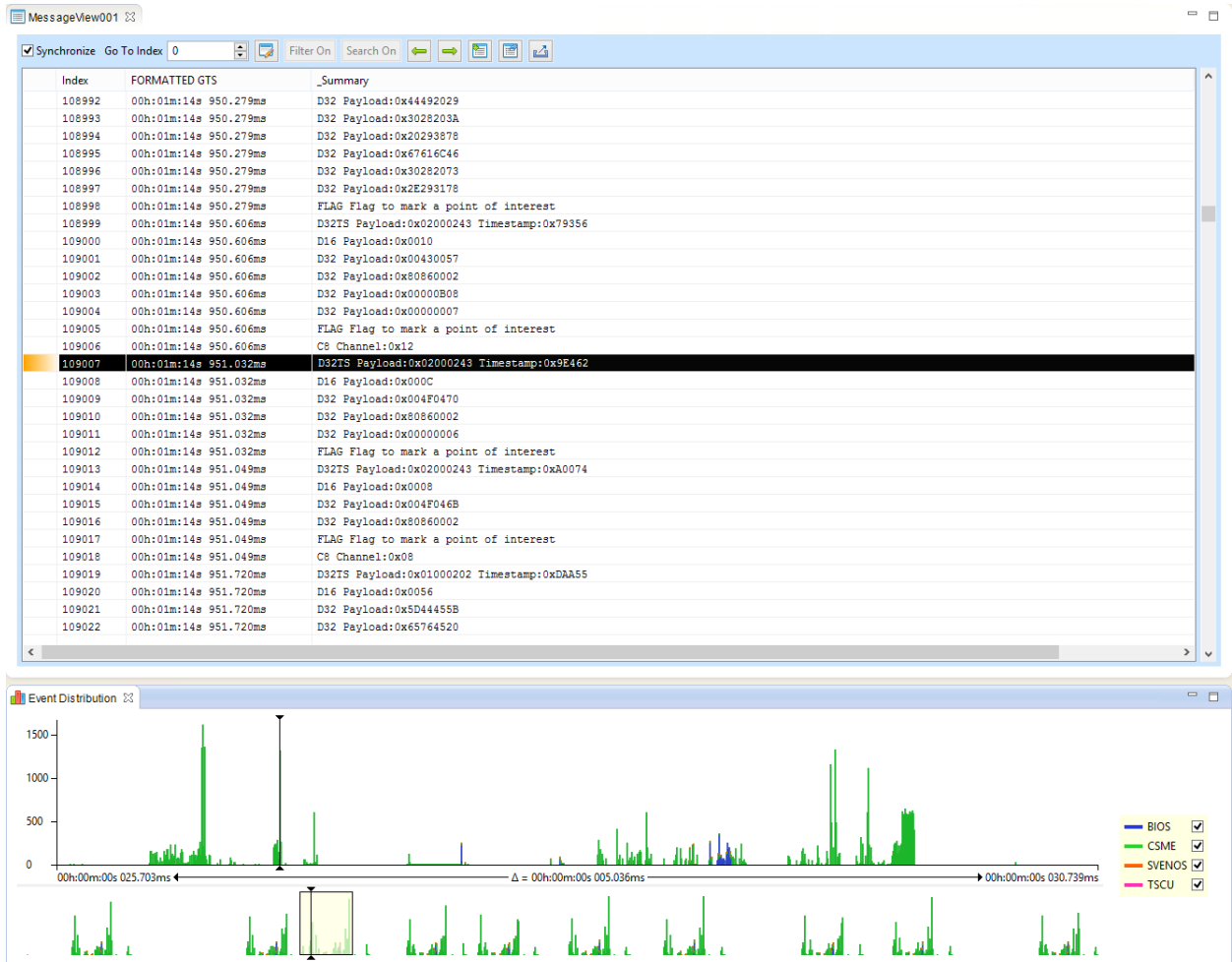


If the mouse pointer is hovering in the **Zoomed View**, a **Cursor** is always shown at the corresponding **Slice** in the **Full View**.

If the mouse pointer is hovering in the **Full View**, a **Cursor** is shown at the corresponding **Slice** in the **Zoomed View** only if it is within the selected range.

More information of the **Slice** is shown in the top right corner of the view, including the timestamp and the index of the first message in the **Slice**.

- To select the first message of a **Slice** in the **Trace Message** view, double click on the **Slice** in the **Zoomed View** or the **Full View**.

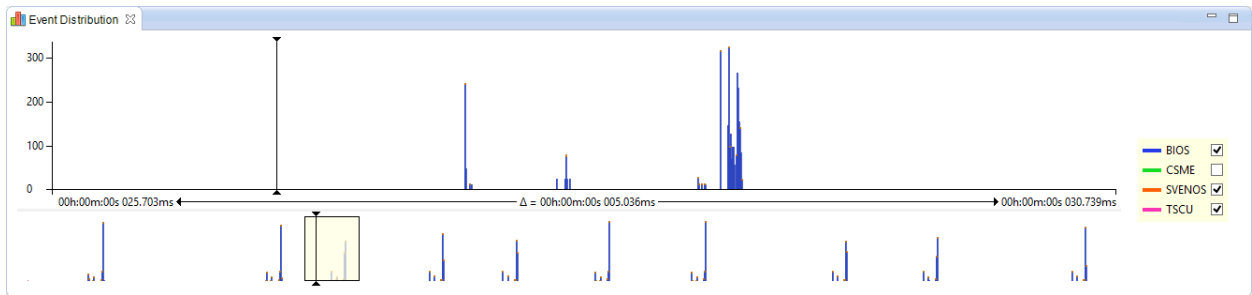
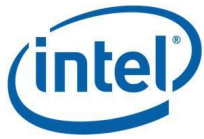


The **Marker** is displayed in the **Full View**, and the **Zoomed View** if applicable.

The next nearest message from the point marked in the graphs is selected in the **Trace Message** view.

On the other hand, selecting a message in the **Trace Message** view positions the **Marker** in the **Event Distribution** view on the **Slice** that contains the selected message.

- To show or hide a source in the graphs, set or clear the checkbox for the source in the **Source Selection**.



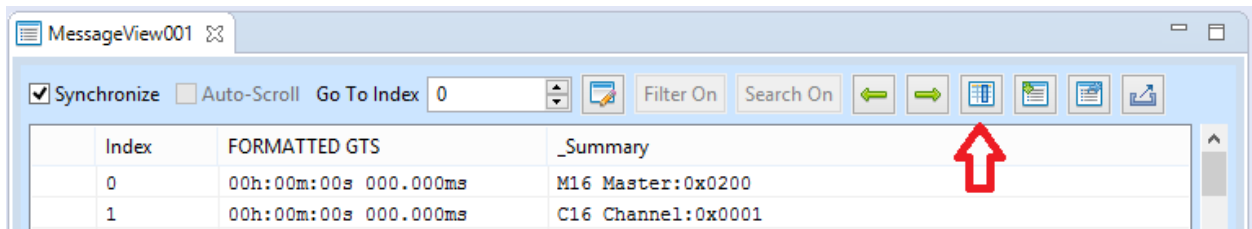
In this example, the source that is overwhelming the distribution data of the other sources is hidden in the graphs. This allows you to study the other sources more effectively.

The **Cursor** and the **Marker** ignore the hidden sources. For instance, setting the **Marker** in the **Event Distribution** view selects the next nearest message of visible sources only in the **Trace Message** view.

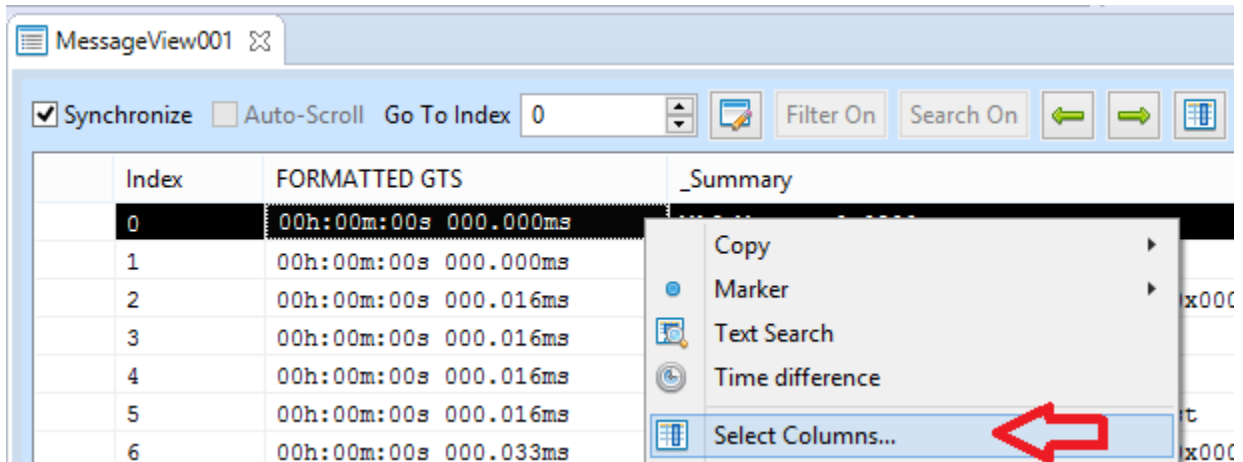
7.5 Basic Trace Analysis: Selecting Trace Fields to be Displayed

In many cases, you are interested in only a certain set of trace fields, which are shown as columns in the trace viewer. You can customize the column display of the trace viewer with the **Select Columns** dialog in **Message View**.

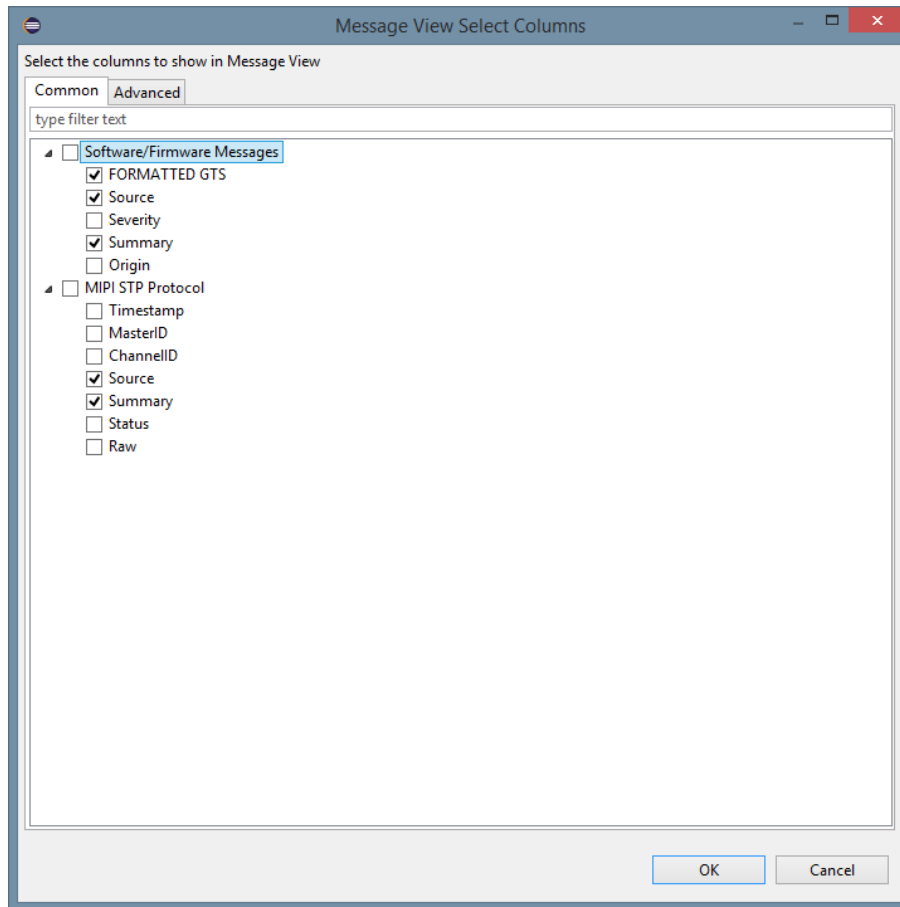
- Open **Select Columns** dialog from the **Message View** toolbar :



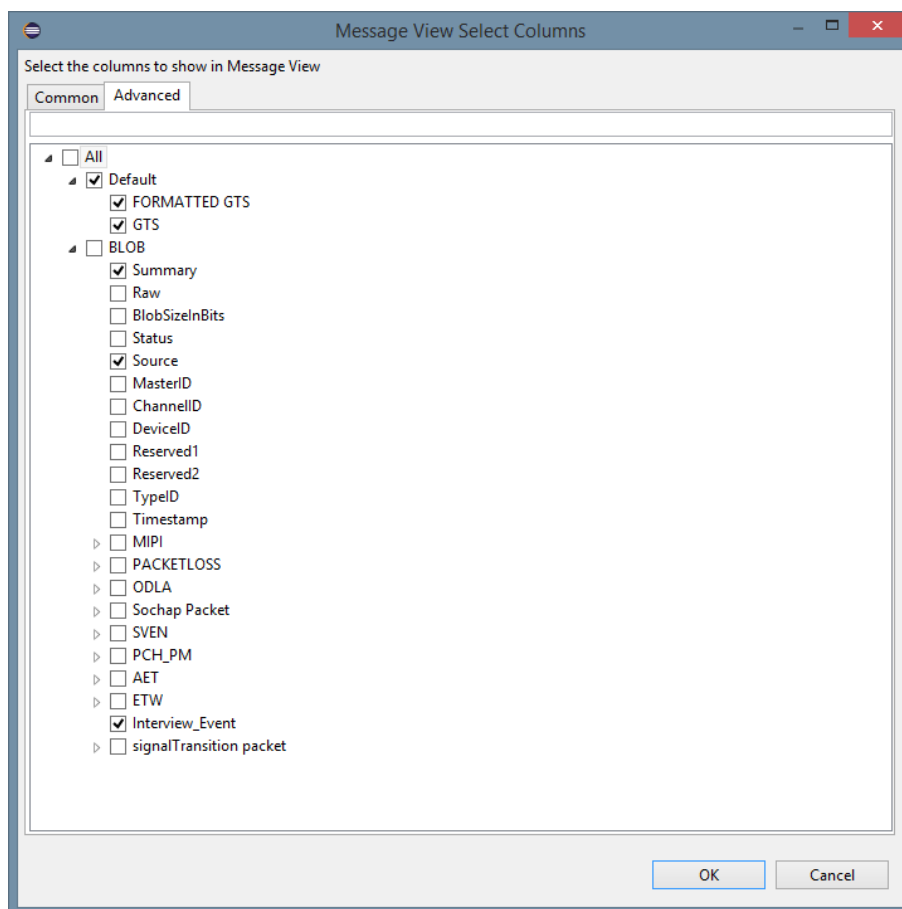
Or right click in message view and pick "**Select Columns...**" item from context menu.



- The **Message View Select Columns** dialog is opened.
The dialog contains two tabs: **Common** and **Advanced**
The selection is synchronized between the tabs.



The **Common** tab displays commonly used fields and grouped by use cases or protocols.



The **Advanced** tab displays every selectable field for a session, where the metadata fields retain their hierarchical structure within **BLOB** as the root node of the metadata tree. The **Default** node contains additional fields that are not part of the metadata.

The **All** node is great for clearing selection, but it is not recommended to simply enable all fields in **Advanced** tab. A maximum of 32 fields can be selected.

NOTE: Some fields are common fields are shared across protocols. When you enable or disable one of these fields, it will be enabled or disabled across all protocols.

9. To make your selection effective, click **OK** Button of the Message View Select Columns dialog.

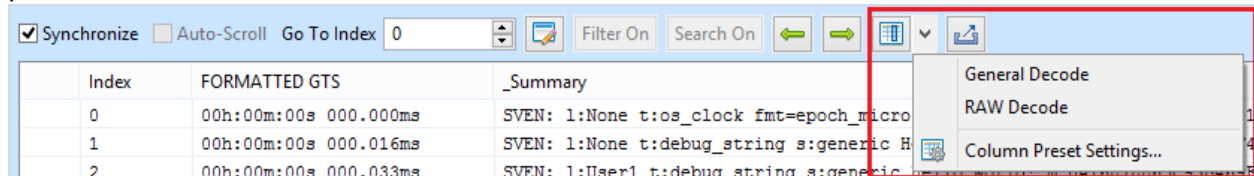
The trace viewer now displays all selected fields of the session, such as:



| Index | FORMATTED GTS | Summary | Mipi Channel | Mipi Master | Mipi Source | PacketStatus | PacketType | Raw |
|-------|-----------------------|---|--------------|-------------|--------------|--------------|---------------------|------------------|
| 0 | 00h:00m:00s 000.000ms | SVEN: 1:None t:debug_string s:gener... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | 8210FF0100000000 |
| 1 | 00h:00m:00s 000.000ms | SVEN: 1:Normal t:debug_string s:gen... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | C212FF0100000000 |
| 2 | 00h:00m:00s 000.000ms | SVEN: 1:Normal t:debug_string s:gen... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | C213FF0100000000 |
| 3 | 00h:00m:00s 000.000ms | SVEN: 1:Normal t:debug_string s:gen... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | C213FF0100000000 |
| 4 | 00h:00m:00s 000.000ms | SVEN: 1:Normal t:debug_string s:gen... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | C213FF0100000000 |
| 5 | 00h:00m:00s 000.000ms | SVEN: 1:Normal t:debug_string s:fun... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | C212FF0200000000 |
| 6 | 00h:00m:00s 000.000ms | SVEN: 1:Normal t:debug_string s:fun... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | C213FF0200000000 |
| 7 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B313FF0200000000 |
| 8 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B312FF0200000000 |
| 9 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B312FF0200000000 |
| 10 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B312FF0200000000 |
| 11 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B312FF0200000000 |
| 12 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B312FF0100000000 |
| 13 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B312FF0100000000 |
| 14 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B313FF0200000000 |
| 15 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B313FF0200000000 |
| 16 | 00h:00m:00s 000.000ms | SVEN: 1:Warning t:catalog_msg s:cat... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | B313FF0200000000 |
| 17 | 00h:00m:00s 000.000ms | SVEN: 1:Error t:debug_string s:stare... | 69 | 1604 | 0x0000450644 | NONE | BLOB.SVEN.EXAMPL... | A213FF0700000000 |

7.6 Basic Trace Analysis: Message View Column Presets

The column preset feature allows user to save column settings and load them very easily. In the Message View click on the button drop down menu to show the available column presets:

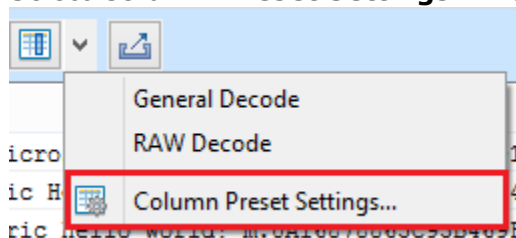


The Intel(R) System Debugger already comes with some predefined column sets. By clicking on the column presets (e.g. **General Decode** or **RAW Decode**), the Message View changes the column layout according to the selected preset.

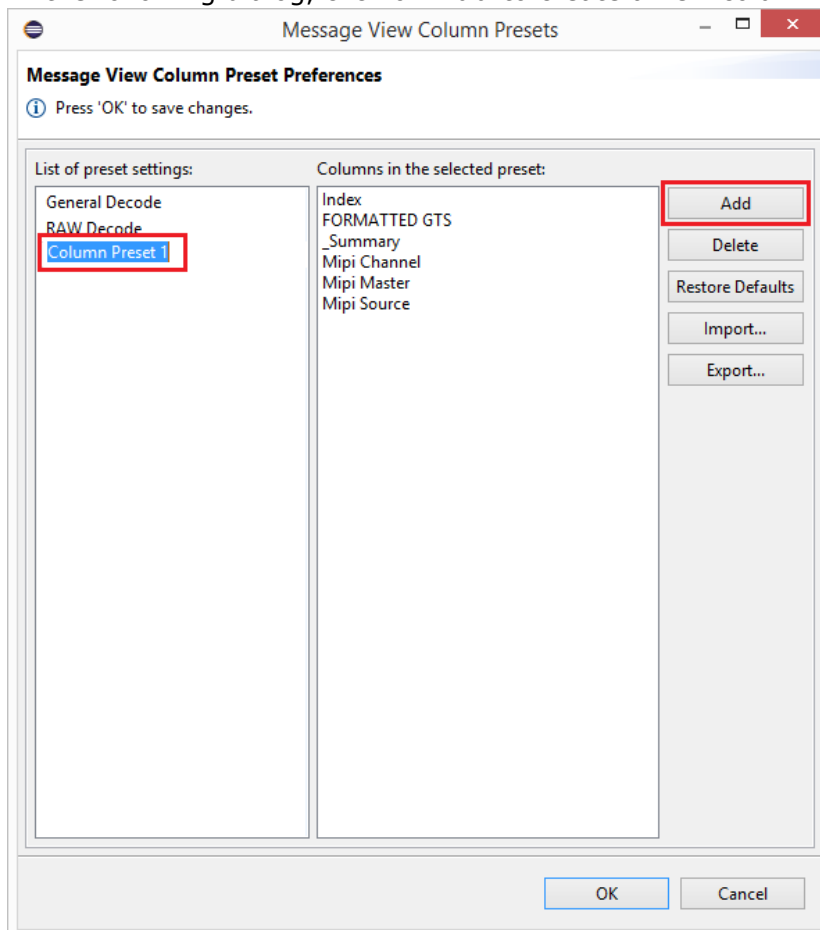
7.6.1 Create new Column Preset

To create a new column preset, do the following:

- 1) Enable all columns in the Message View you want to have in your new preset. To do this follow chapter **7.5 Basic Trace Analysis: Selecting Trace Fields to be Displayed**.
- 2) Select **Column Preset Settings...** from the column picker drop down menu.



- 3) In the following dialog, click on **Add** to create a new column preset.

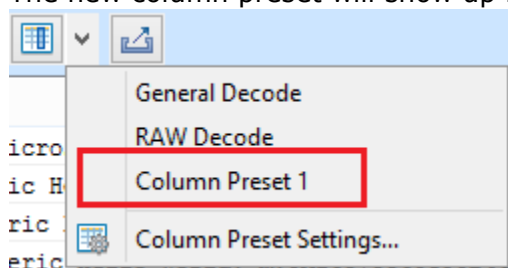


On the left column (**List of preset settings**), you can type in a new name for the column preset. The right column (**Columns in the selected preset**) shows the columns you have activated in the Message View.

If you want to rename the new column preset, right click on the entry and choose **Rename**.

- 4) Click **OK** to save the preset in your workspace.

- 5) The new column preset will show up in the column picker drop down menu:

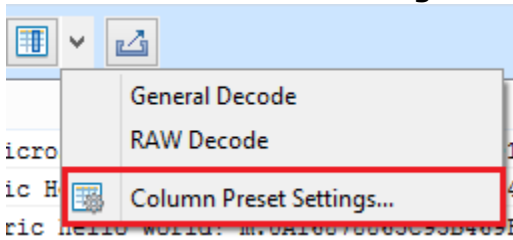




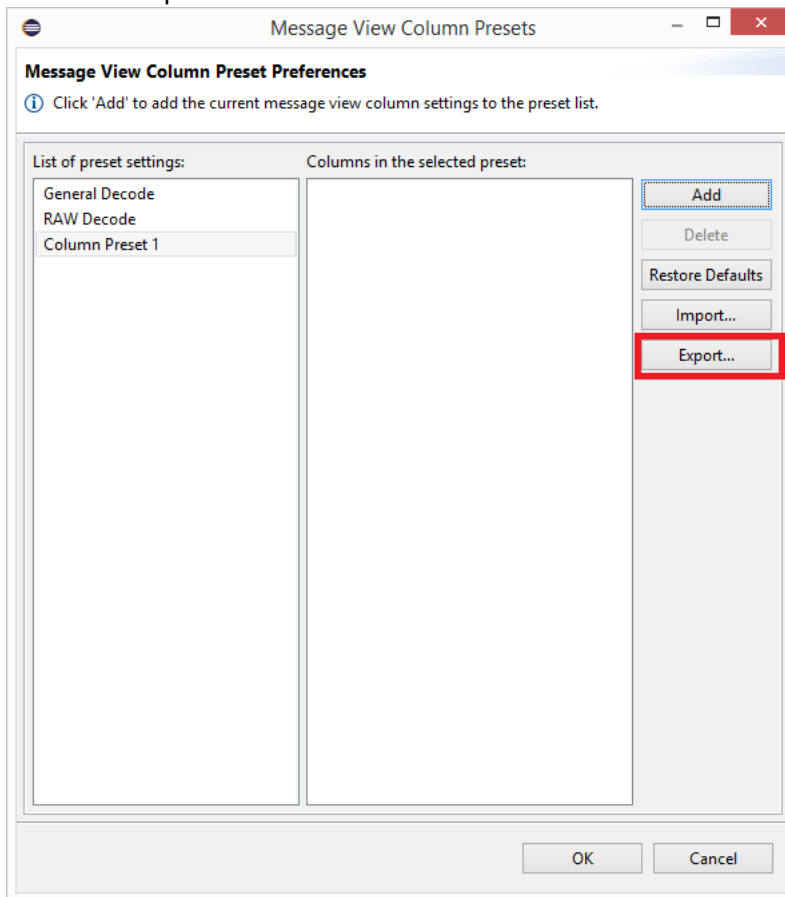
7.6.2 Export Column Preset

To export column presets, do the following:

- 1) Select **Column Preset Settings...** from the column picker drop down menu.



- 2) Click the Export... button:

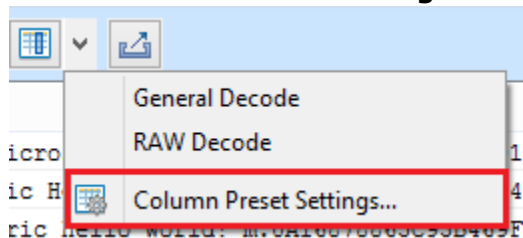


- 3) Choose a location and a filename in the file picker and click **Save**.

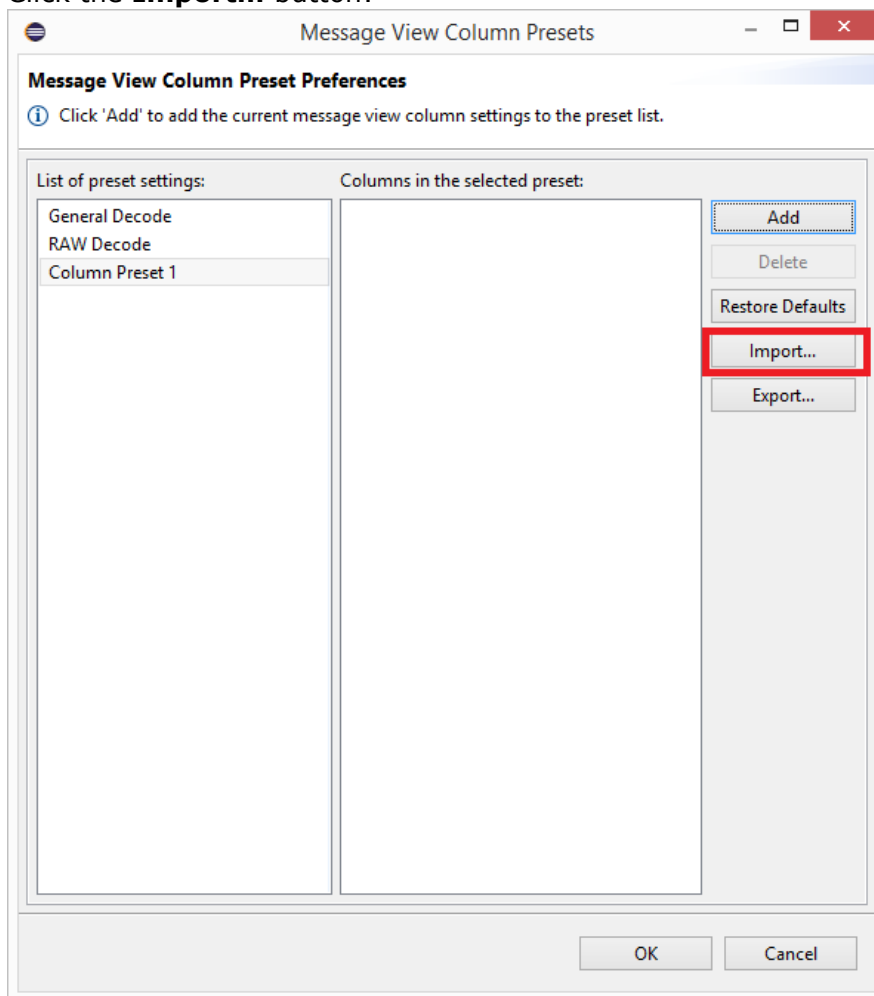
7.6.3 Import Column Preset

To import column presets, do the following:

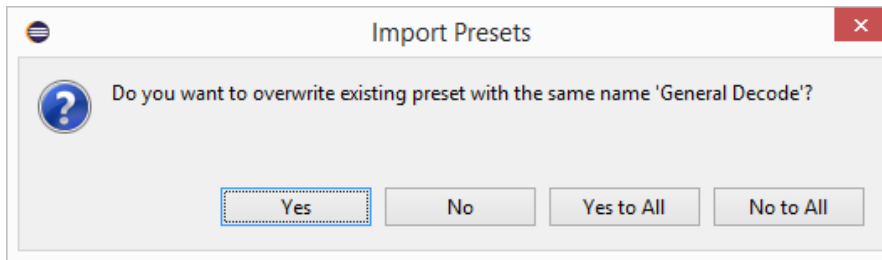
- 1) Select **Column Preset Settings...** from the column picker drop down menu.



- 2) Click the **Import...** button:



- 3) Select the .columnOrder file you want to import and click **Open**.
- 4) If the **Message View Column Presets** dialog detects that you try to import existing settings, it will ask if the column preset should be overwritten or not:



Select **Yes** to overwrite the existing column preset in the list.

Select **No** to skip importing the existing column preset.

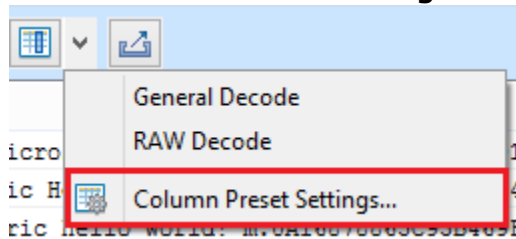
Select **Yes to All** to overwrite all existing column presets in the list from the import file.

Select **No to All** to skip importing all existing column presets.

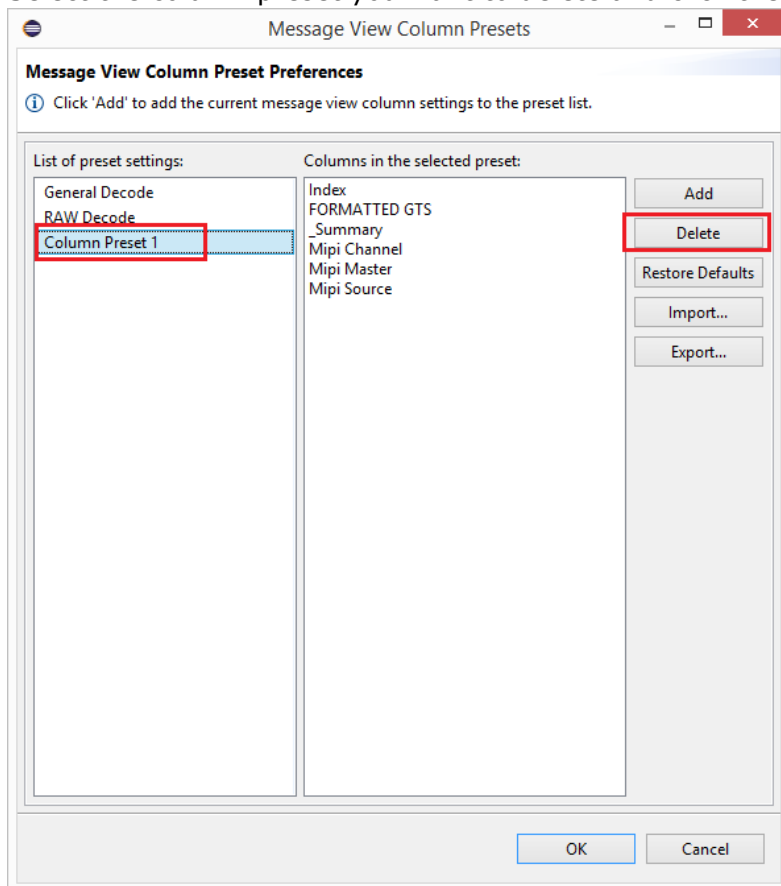
7.6.4 Delete Column Preset

To delete a column preset, do the following:

- 1) Select **Column Preset Settings...** from the column picker drop down menu.



- 2) Select the column preset you want to delete and click the **Delete** button

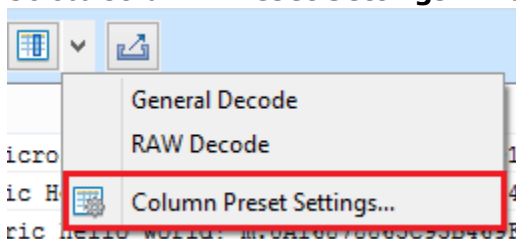


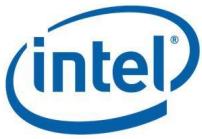
- 3) Click **OK** to save your changes.

7.6.5 Restore default Column Presets

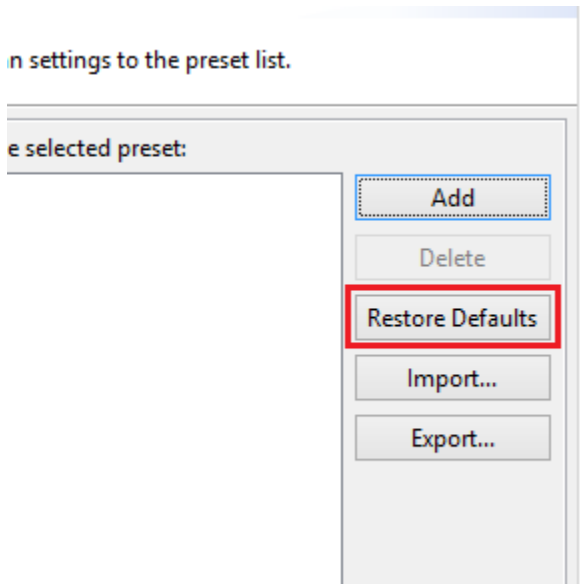
The column presets can be restored to default, by doing the following:

- 1) Select **Column Preset Settings...** from the column picker drop down menu.





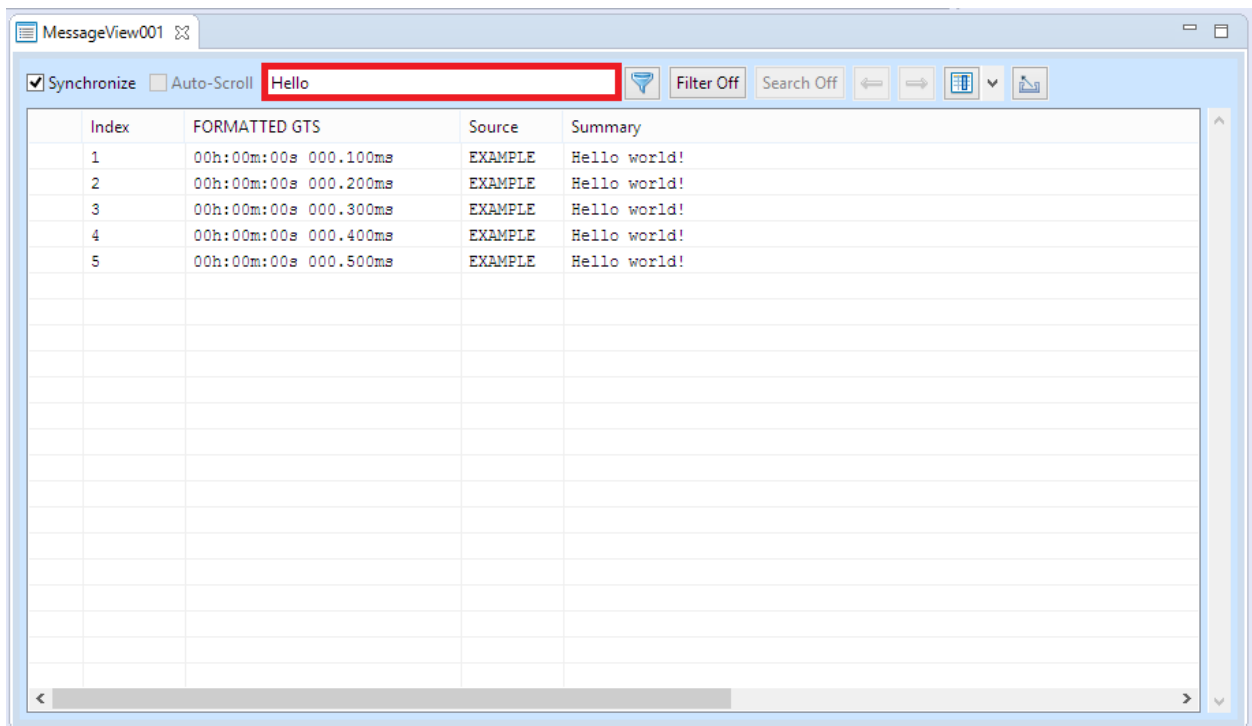
- 2) Click the **Restore Defaults** button.



Warning: this will remove you custom column presets.

7.7 Quick filtering a trace

To quickly find events you are interested in, type a text matching criteria into the quick filter text box in the Message View and press enter:

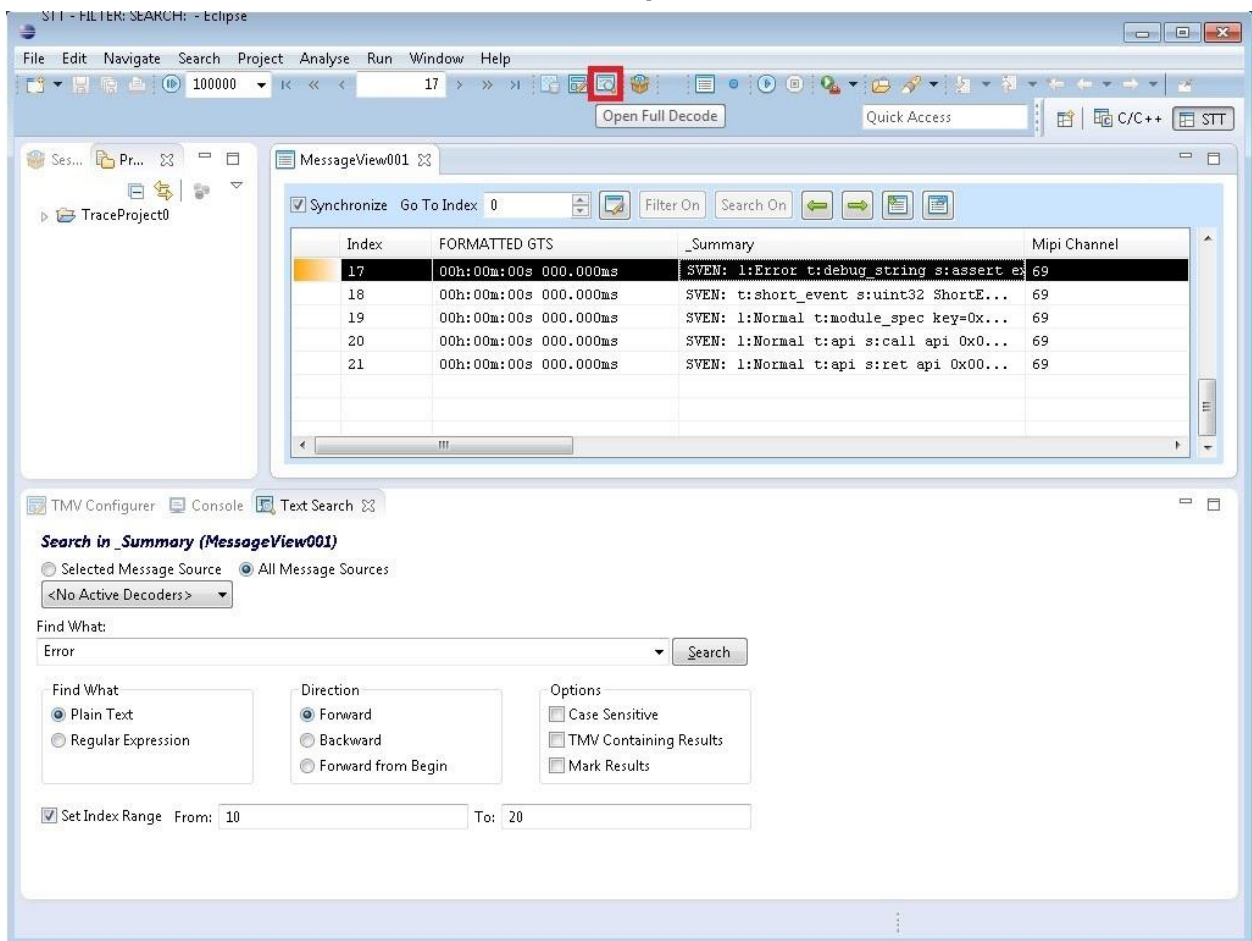


The view will display only the messages, which have visible fields that match the given criteria. To display all messages again, click on "Filter Off". For more complex criteria, refer to section 7.10, Basic Trace Analysis: Message Search.

7.8 Basic Trace Analysis: Detailed View of a Trace Entry

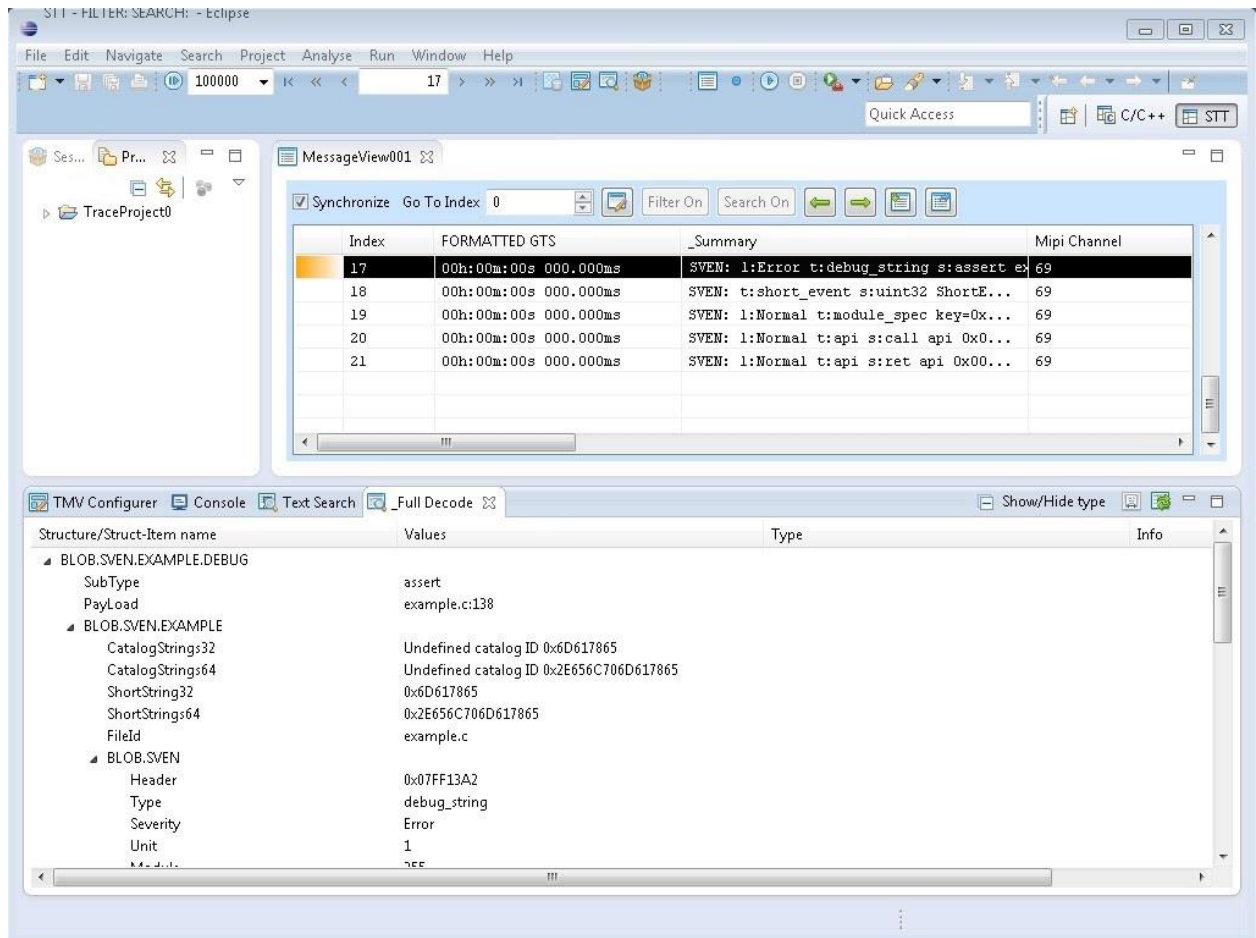
This section explains to examine the trace entry for the SVEN Error event in detail.

1. With the SVEN Error event selected, click the **Open Full Decode** icon in the toolbar:



The tree of the decoded event is displayed, with all decodable fields and values. Note that this tree displays the most-detailed or most protocol-specific decode on the top. Farther down the information becomes more general – until it reaches the base Mobile Industry Processor Interface (MIPI) record level ("Blob") at the very bottom. Blob stands for Binary Large Object.

For the SVEN Error, the tree is displayed such as:



The view displays the SVEN Error trace type ("assert"), the erroneous file ("example.c") and the erroneous code line ("138").

NOTE: One hierarchy level down, you can see that this SVEN Error trace entry did not have a valid catalog file. This means that the given number could not be decoded to a text string. If a valid catalog file would have been in place, the text string from the catalog would have been printed, instead of "Undefined catalog ID <messageID>". Two hierarchy levels down, you can also inspect which software unit and module caused the trace entry, which GUID was assigned to this entry - these are defined by the particular instance of SVEN that generated the trace and may vary.

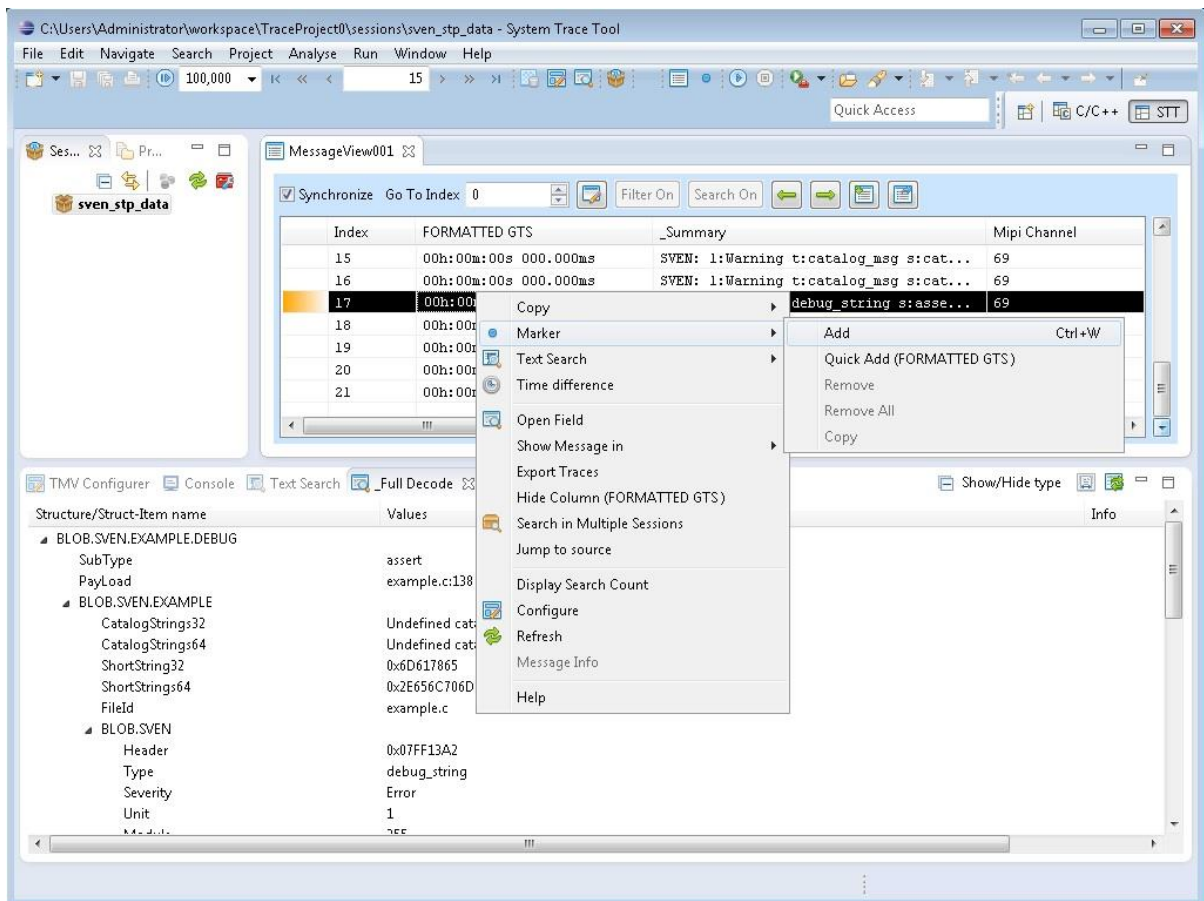
To recognize quickly trace messages in a later stage of your project, for example for team collaboration activities, it is possible to mark them. This is described in the next section.

7.9 Basic Trace Analysis: Marking Trace Entries

You can mark trace entries for later use. This is useful for recognizing them in a later stage of your project, for example if you wish to examine them in a team collaboration effort.

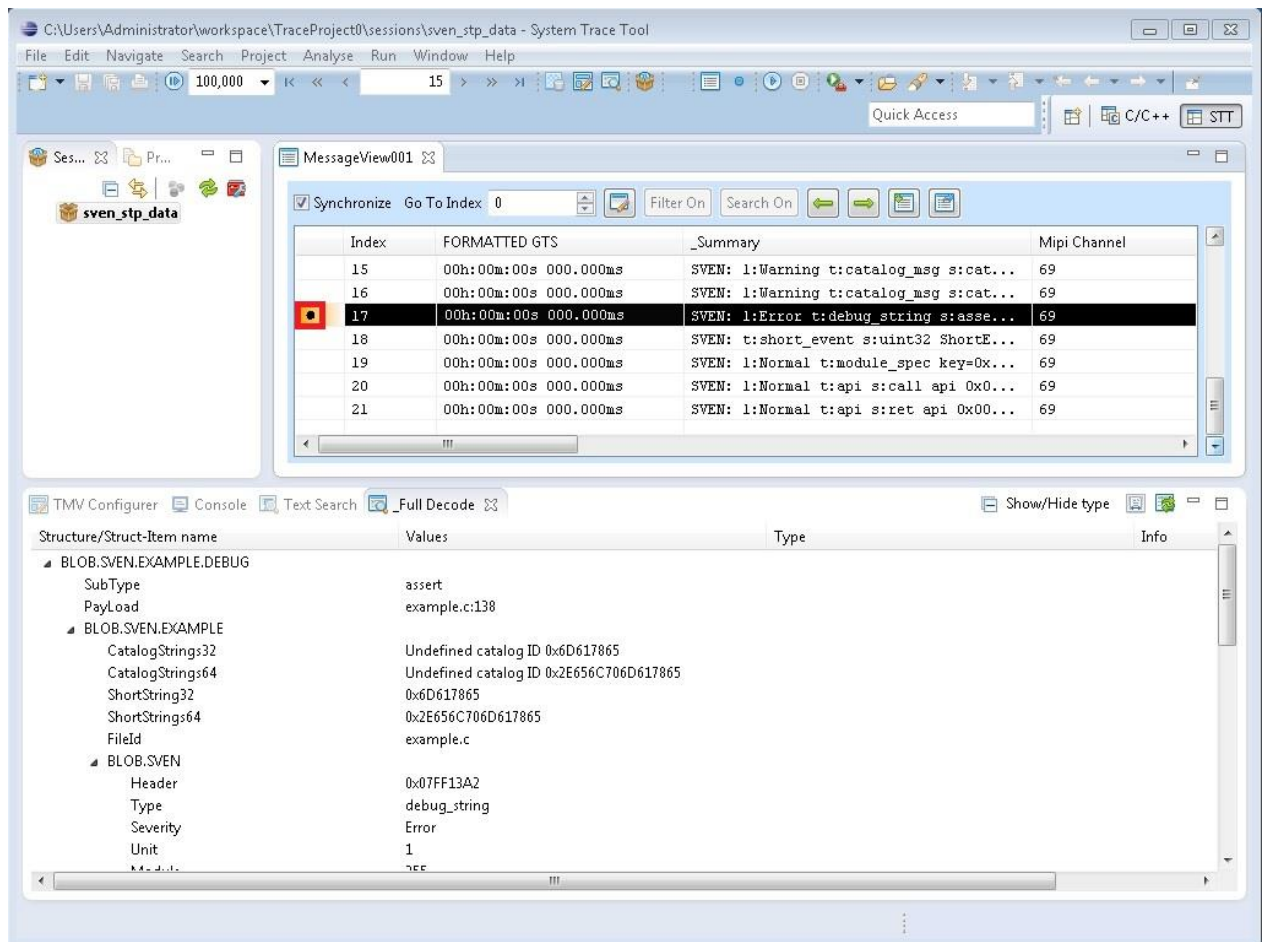
1. Assuming the SVEN Error trace message is selected, right-click the message.

The trace entry's context menu is displayed.



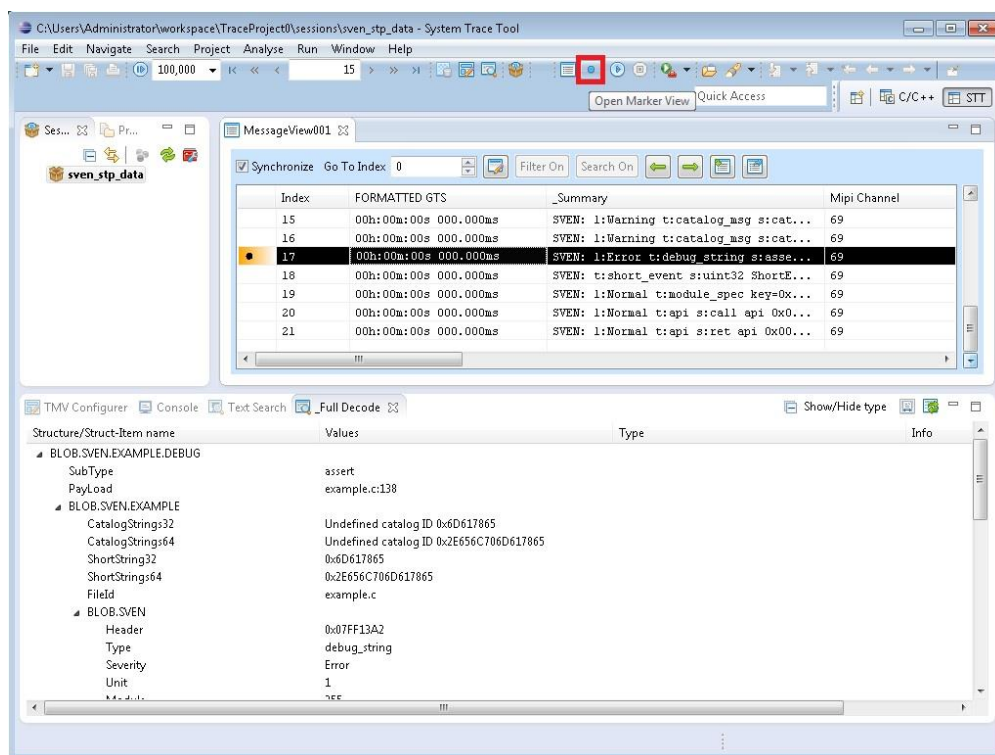
2. Select **Marker > Add** from the context menu.

A marker bubble is added in the very first column of the trace view:

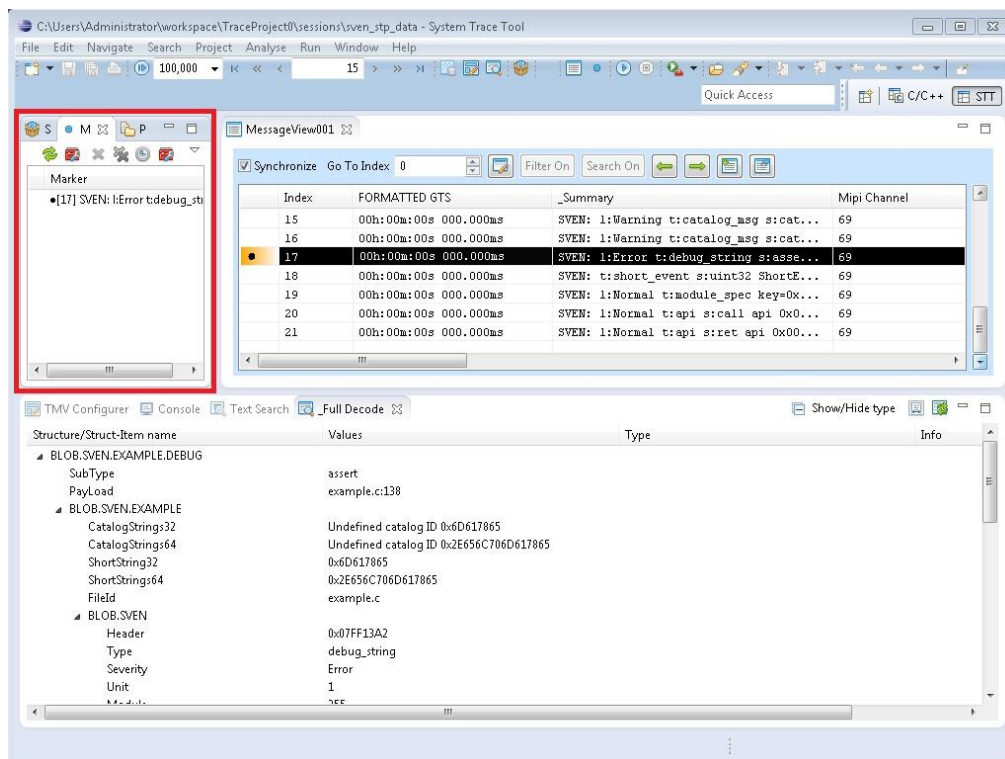


This works well for small numbers of markers in a trace. If there are more markers, a dedicated view displays all markers set in the trace.

3. To open this **Marker** view, click the **Open Marker View** icon in the toolbar.



The **Marker** view is displayed on the top-left tabs.

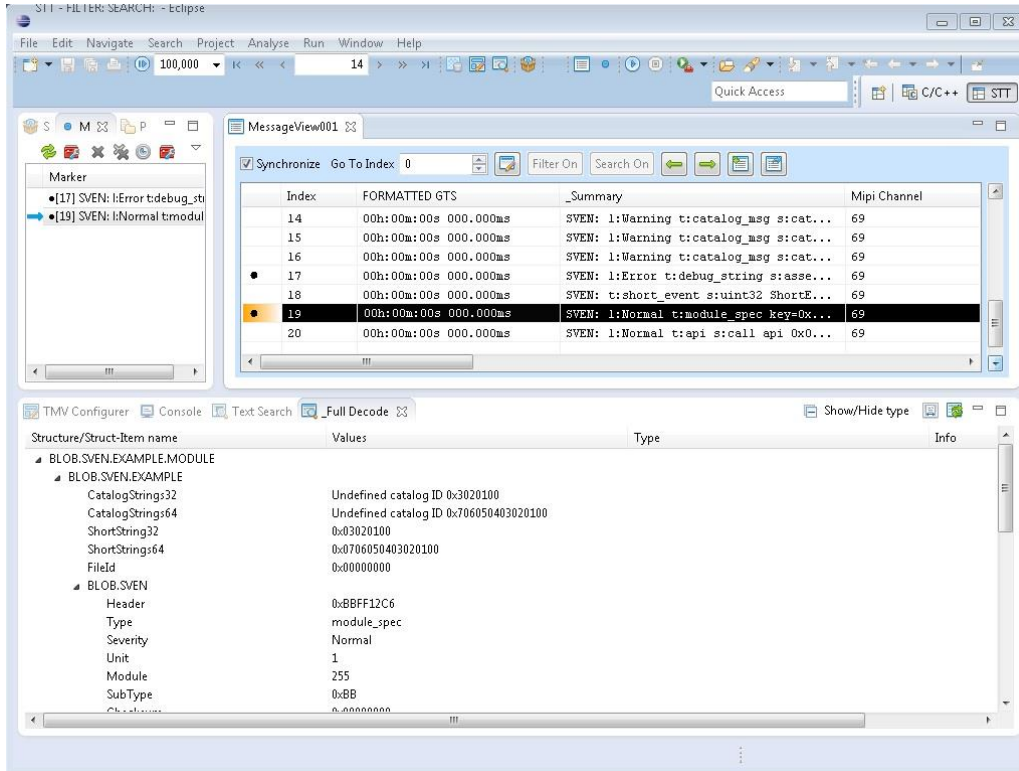


The **Marker** view, displays a list of all markers in a trace message view.



4. Add another marker on index 19 by selecting the respective trace entry with a left-click.
5. Open the context-menu by right-click and select **Marker > Add**.

The **Marker** view now contains two entries, such as:



You can now use the trace marker entries in the Marker view to scroll quickly the trace to display interesting entries.

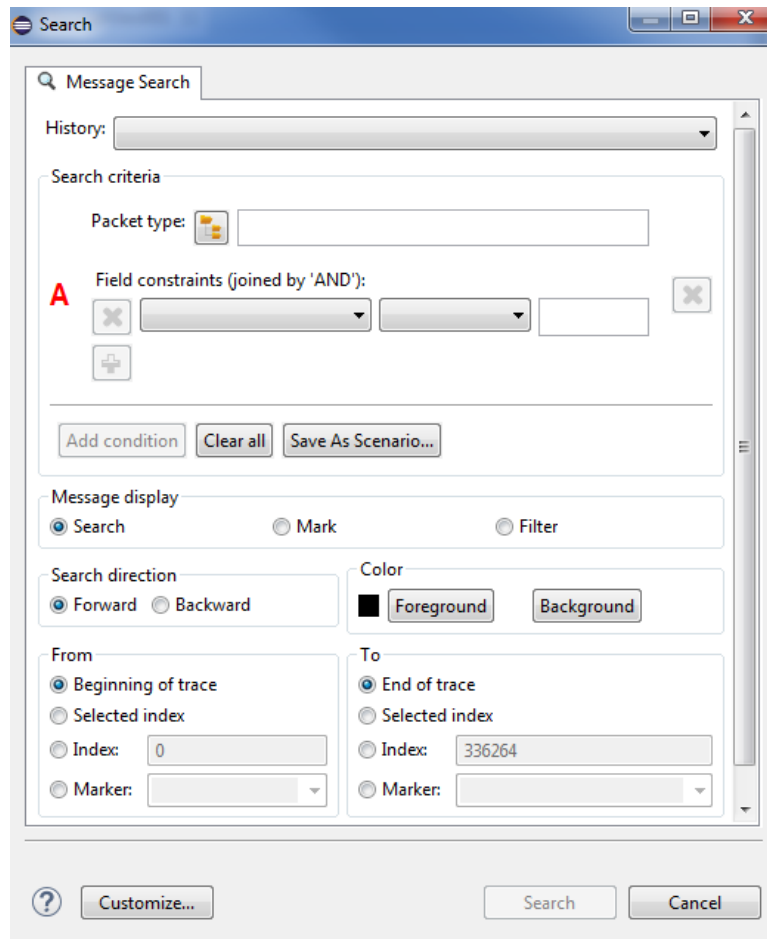
NOTE: If your trace contains timestamp information (the SVEN example does not), you can also examine, for example, the frequency/time interval in which a certain event occurs. To do so, select Time Difference from a marker's context menu and select a second entry by holding the <CTRL> key.

Besides the basic trace analysis, there is also a more advanced way of searching, marking and filtering trace entries based on user-defined rules. The subsequent trace analysis sections in this document describe trace analysis based on rules.

7.10 Basic Trace Analysis: Message Search

The **Message Search** capability enables you to perform search, filter and marking on a given trace session according to advanced search criteria.


To open the **Message Search** dialog, select **Search -> Search ...** from the Eclipse* main menu, or press “**CTRL-F**” when focus is on the trace viewer.



See the following sections for details on each section.

7.10.1 Packets Constraints Setting

Click the “**packet explorer**” icon and select the relevant packets:



Packet type: 

You may also type in the packet name prefix and use the auto-completion feature. In this case, type in the “OR” operator when multiple packets are chosen.



You may leave the packet selector empty denoting all packets.

7.10.2 Field Constraints Setting

- Click the field constraints drop-down menu and select the desired field. Fields that belong to at least one of the selected packets will be shown.
- Set constraints to the selected fields.
- Click the "plus" button to add field constraints. 
- Click the "x" button to remove a field constraint. 
- An **"AND"** logics will be applied to multiple field constraints.
- Click the **"Add Condition"** button to add additional packet-field constraints to your search. An **"OR"** logics will be applied.

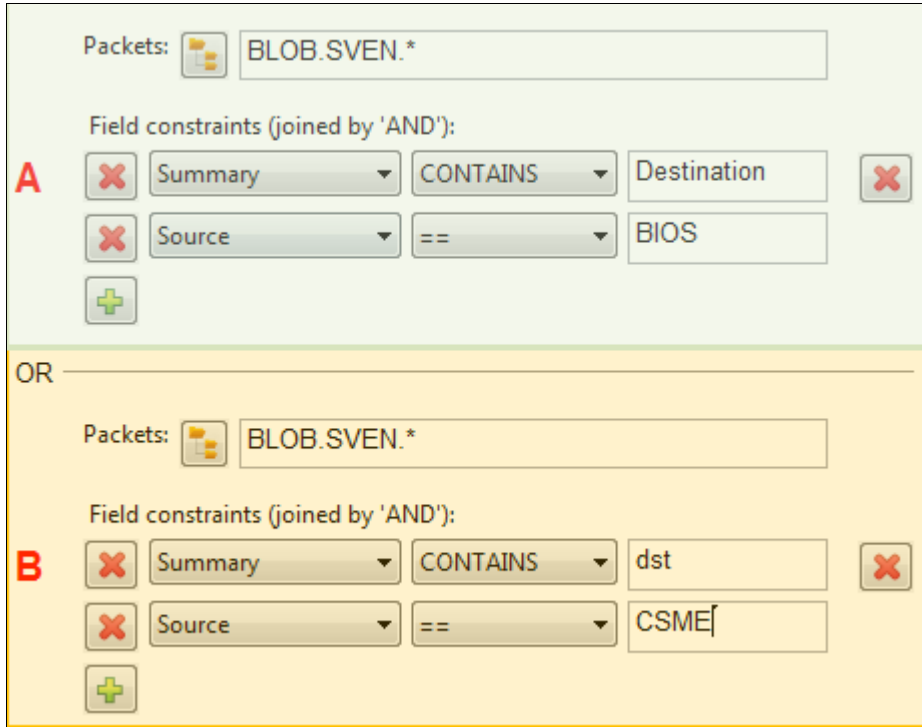
Examples:

- Search for messages of the type "BLOB," with *Severity of Fatal or Error or Warning*:

- Search for messages of the type "BLOB" coming from the BIOS where *POSTCODE* appears in the Summary:

- Multiple search criteria – search for one of the following options:

- BLOB.SVEN messages coming from the *BIOS* where the text contains "Destination"
- BLOB.SVEN messages coming from the *CSME* where the text contains "dst"



The screenshot shows a search filter configuration window with two filter rules, A and B, separated by an OR operator.

Filter A (Green background):

- Packets: BLOB.SVEN.*
- Field constraints (joined by 'AND'):
 - Summary CONTAINS Destination
 - Source == BIOS

Filter B (Yellow background):

- Packets: BLOB.SVEN.*
- Field constraints (joined by 'AND'):
 - Summary CONTAINS dst
 - Source == CSME

7.10.3 Search Direction Setting

Sets the search direction to forwards or backwards. Forward searches will start from the beginning of the selected range (see section 1.5), and backwards searches will start from the end of the selected range. This option is ignored for filters and markers.

7.10.4 Search Type (Search, Filter, Mark) Selection

Select one of the following actions to be performed:

- **Filter** – shows only trace entries that match the criteria.
- **Search** – finds all trace entries that match the criteria starting from the lower bound for "forward" search or from the upper bound for 'backward' search.
- **Marker** – sets markers to matching trace entries. It is recommended to delete all Markers from previous steps before continuing with this analysis example. To do so, perform the following steps, right-click on the **Marker** view and select **Remove All Markers** from the context menu.




7.10.5 Boundary Setting

To limit the search to a certain window, choose the upper-bound and lower-bound using the “**From**” or “**To**” options. If you defined markers, they will be displayed in the Marker drop-downs, allowing you to use them as boundaries.

The screenshot shows the 'Boundary Setting' dialog box. It is divided into two main sections: 'From' and 'To'. Each section contains radio buttons for different search boundaries: 'Beginning of trace', 'Selected index', 'Index' (with a text input field), and 'Marker' (with a dropdown menu). In the 'From' section, the 'Marker' option is selected, and the dropdown shows '[3] Strat of PM transaction'. In the 'To' section, the 'Marker' option is also selected, and the dropdown shows '[21370] End of PM transaction'. A dropdown menu is currently open for the 'To' section's 'Marker', displaying two options: '[3] Strat of PM transaction' and '[21370] End of PM transaction', with the latter being highlighted.

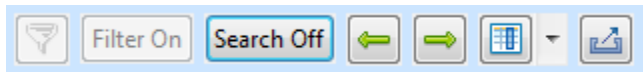
7.10.6 Search Activation

Click the “**Search**” button to activate the search.

To search for the next packet, use the forward or backward buttons at the Message View toolbar .

To cancel a search, click the “**Search Off**” button at the Message View toolbar.

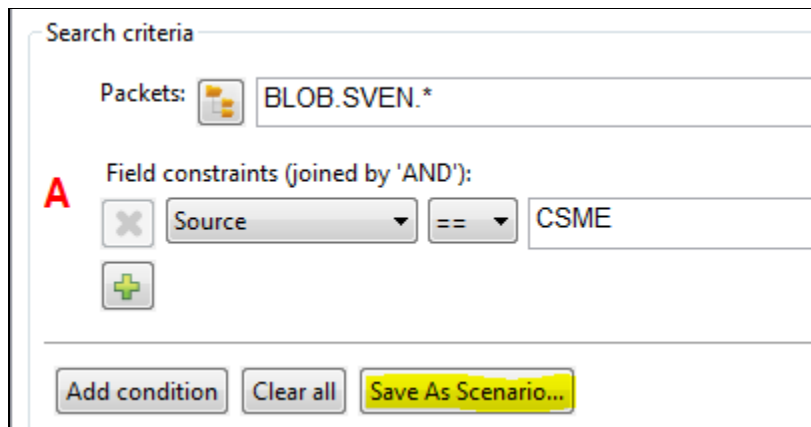
To cancel a filter, click the “**Filter Off**” button at the Message View toolbar. Clicking “**Filter On**” will reactivate the last filter.



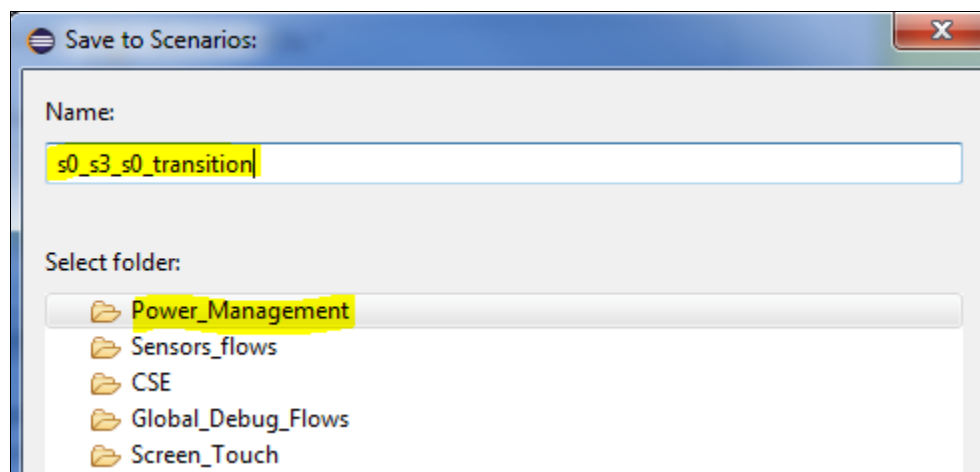
7.10.7 Search Specification Saving

You may save the search specification in TRAM scenario explorer and TRAM scenario repository. This will enable you to use the search specification later on or share it with your colleagues (see Scenario Specification).

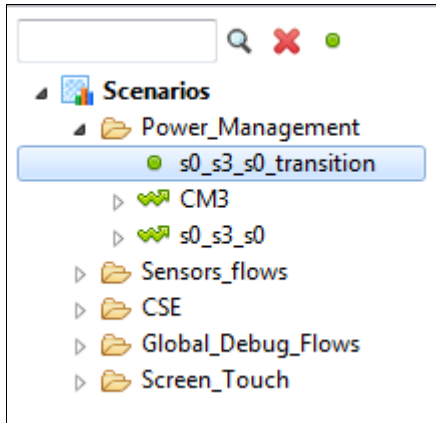
To save your setting for future use, click the “**Save As Scenario**” button.



A dialog showing the scenario tree will appear enabling you to choose the desired scenario folder. Choose the desired folder, set a name to the search criteria, and click “OK”.

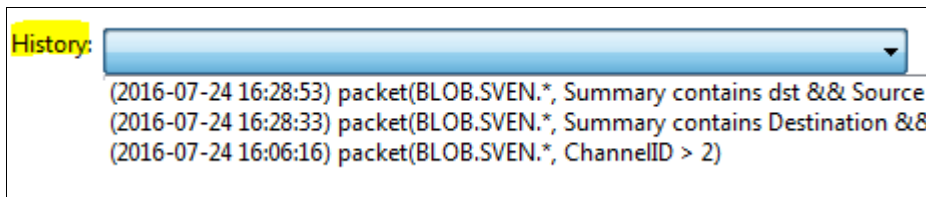


The scenario will be displayed in TRAM scenario explorer. You will be able to edit your scenario or activate it. See section 7.11.3 for details.



7.10.8 Search History

The history drop-down displays all previous search patterns (ordered by activation time), represented in the TRAM language. Selection of a search pattern populates the dialog with the search definition, enabling you to re-activate the search.



7.11 Advanced Trace Analysis: TRAM - TRace Analysis & Mining

7.11.1 What Is TRAM?

The **TR**ace **A**nalysis & **M**ining (TRAM) feature refers to a set of capabilities that deal with trace based analysis.

TRAM increases the productivity and quality in validation and debug (HW, SW) by providing the following capabilities:

- **Flow Detection:** Detection of complex scenarios (flows) on large traces (w/ missing data)
- **Advanced Trace Search:** Filter and search according to user defined criteria (simple or complex).

- **Measurements:** Measurements of latency, performance, coverage relating to a detected scenario on a trace.
- **Abstraction:** Abstraction of traces and stitching of related trace data.
- **Feedback:** Feedback on broken flows and trace quality (missing data, ambiguous data, etc.).

The input to TRAM is a **trace session** to be analyzed and **scenarios**, defined in TRAM language, representing a system behavior to be detected and measured.

7.11.2 TRAM View

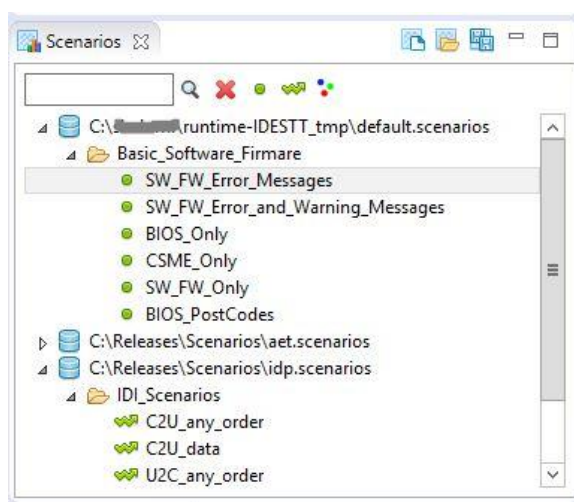
To open **TRAM** view, click the “**Trace Analysis**” icon in the toolbar.

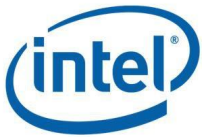


If this is the first time that you are using TRAM and no scenarios are defined, TRAM will create a default scenario file (default.scenarios) with a few examples and will load it into TRAM View.

TRAM GUI is composed of three areas:

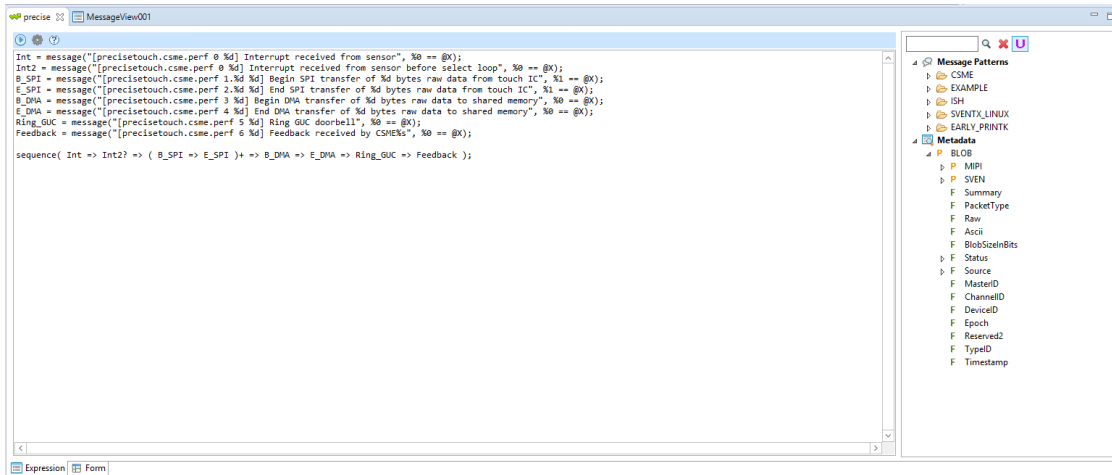
1. **Scenario Explorer** (left pane)
 - Enables you to define scenarios and organize them in an explorer





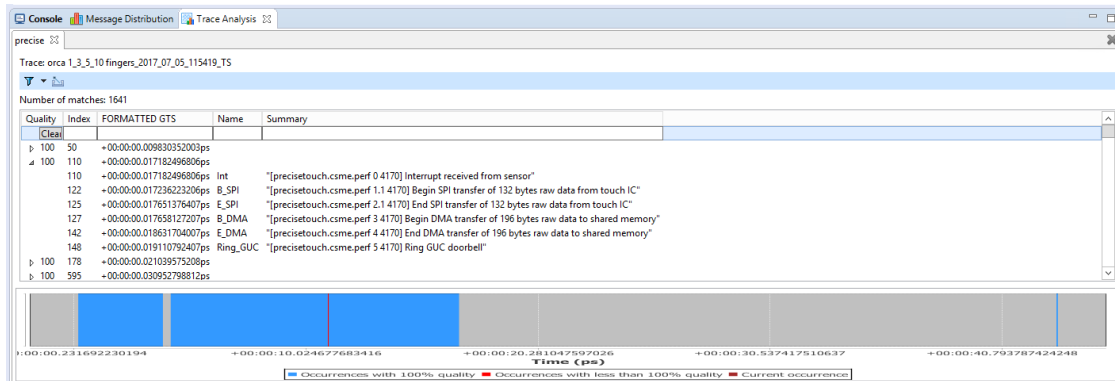
2. Scenario editor (top section)

- Enables you to define either a scenario using TRAM language (expression tab) or using a designated graphical form (form tab) or a scenarios' set using a form. See sections 7.11.3.6 and 7.11.5 for more details.
- The expression editor contains also the **Message and Packets Explorer** (right):
 - Displays the messages and packets that are the building blocks for scenario definition. This enables you to drag & drop messages, packet names, field names, and mnemonic values into the expression editor and create scenarios.
 - **Note:** only messages that were captured in advance are displayed. You may define your scenario on top of messages that are not in the explorer.



3. Results area (bottom section)

- Display the results of scenario detection and measurements. See section 7.11.4 for more details.



7.11.3 Scenario Specification

TRAM enables you to define **Scenarios** for analysis, organize them in an explorer, and save them in a scenarios repository file.

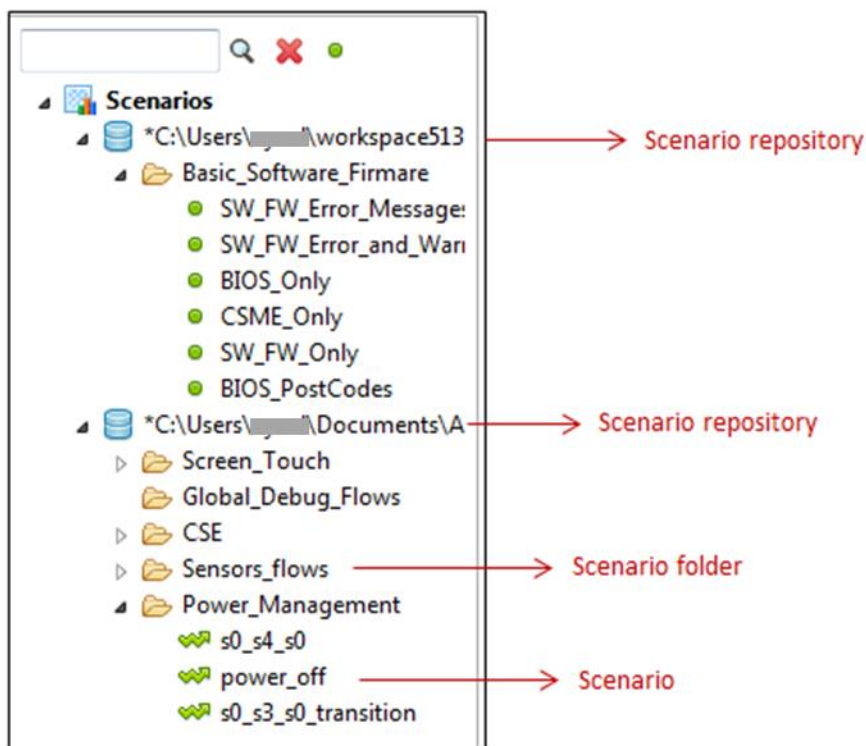
7.11.3.1 TRAM Scenarios

Scenarios represent any system behavior to be detected, measured, and filtered on top of traces. Scenarios can represent a simple trace pattern to be searched/filtered or an entire complex flow such as a sequence or a state-machine.

Scenarios are defined in TRAM language. For more details on TRAM language, see section 8.




7.11.3.2 TRAM Scenario Explorer

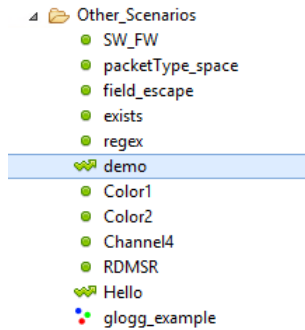
The TRAM scenarios explorer displays the TRAM scenarios and scenarios sets. This view enables you to open multiple scenarios repository files, organize your scenarios in folders hierarchy for better navigation and management, and save them to a TRAM scenario file (.scenarios) for reuse.




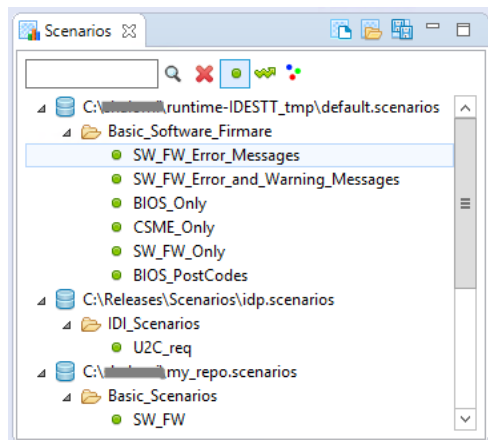
There are three types of objects in the explorer:




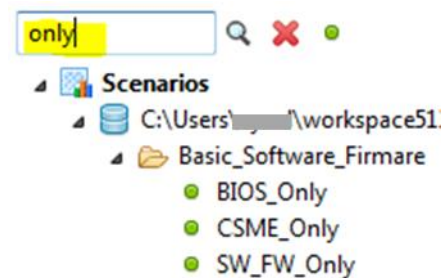
1. **Basic scenarios**  : a single event, often used for trace and search filter
2. **Complex scenarios**  : a sequence of events over time
3. **Scenarios sets**  : a set of scenarios with different search/filter/mark actions



Use the different types icons at the top to filter the tree by one or more of the types. Use the cancel button  to remove all filters.



To **search** for a certain scenario, use the search entry. TRAM will display and expand folders that match the search pattern or folders having a scenario that matches the search pattern. Empty folders will not be displayed even if they match the search pattern. Use the cancel button  to remove this filter.



You may also combine textual filter with filtering by type to get only objects that match the text of the type selected.

7.11.3.3 Using a Default Scenario File

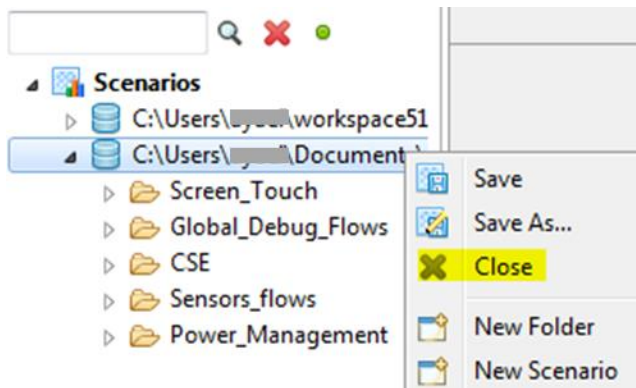
If you are using TRAM for the first time, a **default.scenarios** file will be copied over into your workspace and will be opened and displayed in the Scenario Explorer. The default repository file contains a few example scenarios. You may modify and save this file (recommended with a different name).

7.11.3.4 Using a Predefined Scenario File

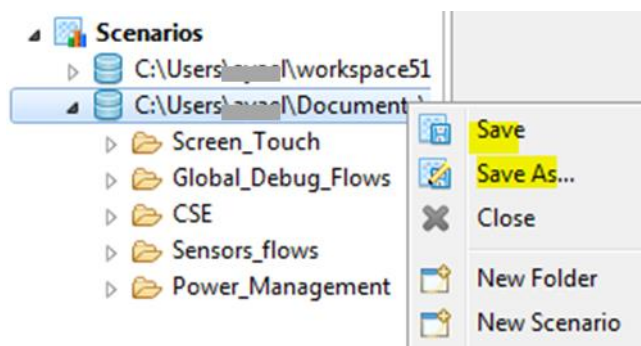
To open a predefined scenario file, click the **Open** button at the Scenarios view toolbar and choose the scenario specification file. This file has a ".scenarios" suffix, and it must follow the TRAM format (typically created by using the **Save** option).



To close a scenario file select the scenario repository from the explorer and then select the **Close** option from the context menu.



To save changes done to the scenario repository, select the scenario repository from the explorer and then select the **Save** or **Save As** option from the context menu.



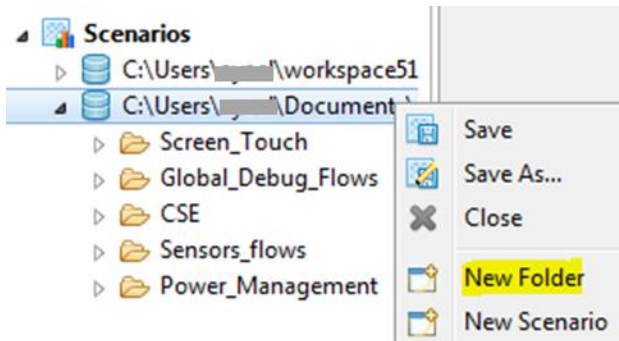


To save changes done to all scenario repositories, click the **"Save All"** button at the TRAM Toolbar (right side).



7.11.3.5 Creating a Scenario Folder

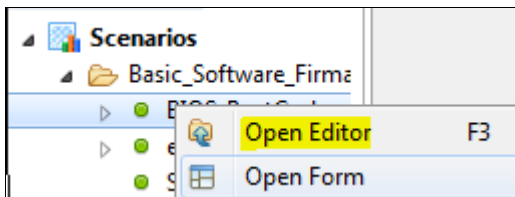
To create a folder in the **Scenarios Explorer** select **"New Folder"** from the context menu (right click). You can create a new folder under a repository or under a parent folder.



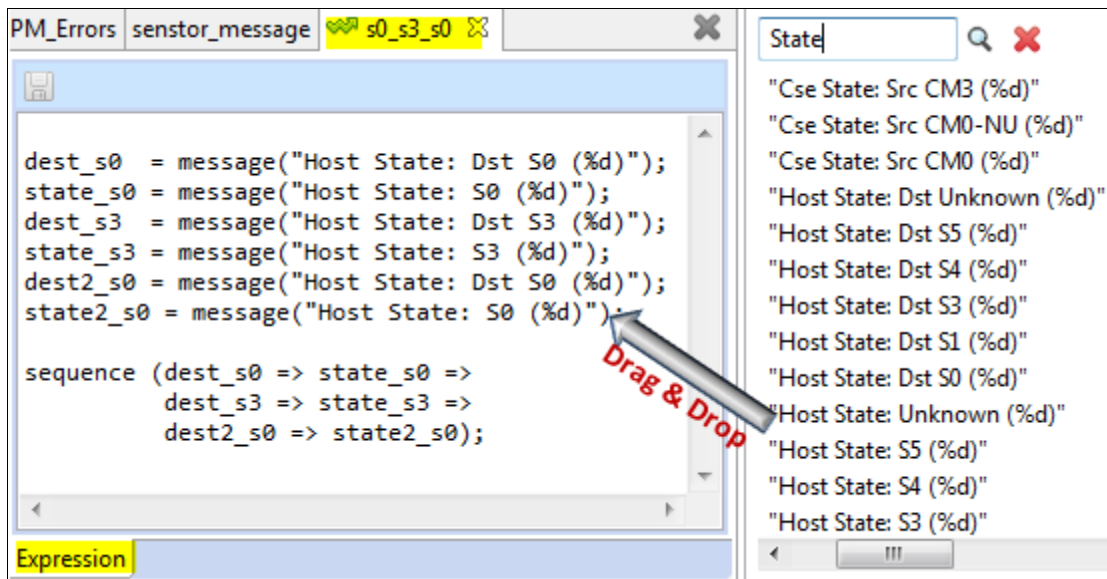
7.11.3.6 Creating a Scenario

To create a scenario follow these steps:

1. Create a new scenario
Choose the desired folder or repository and select **"New Scenario"** from the context menu. Rename your scenario if needed.
2. Define the scenario expression using the text editor
To define the scenario expression, double-click the scenario or select **"Open Editor"** from the context menu to open the expression tab for editing. The scenario is defined in TRAM language (see section 8).



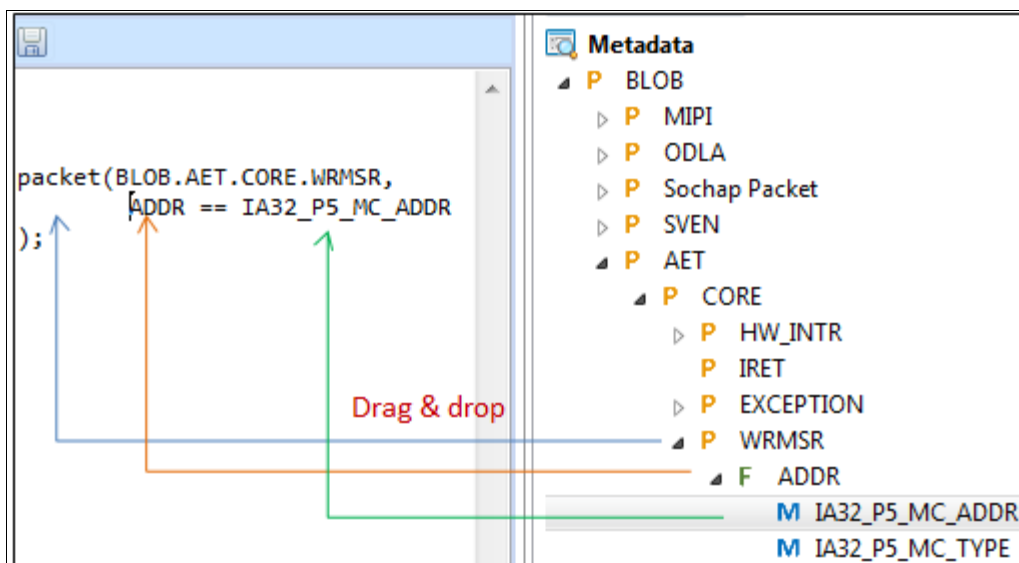
To create an event using the message macro, drag & drop messages from the Message Explorer into the **Expression**. Note: only messages that are known in advance are displayed. You may use messages that are not in the explorer. For example, you may copy patterns from the Message View.



To create a packet based event drag & drop the packet name from the metadata explorer into the **Expression area**.

To create a constraint on a field drag & drop the field name.

To use a mnemonic value as a constraint value, drag & drop the mnemonic name.

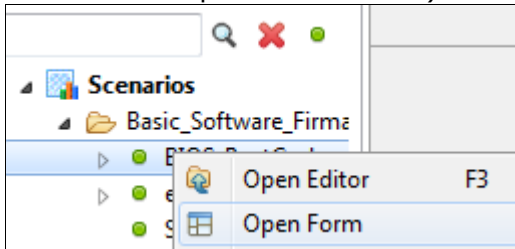


3. Define the scenario expression using the form editor

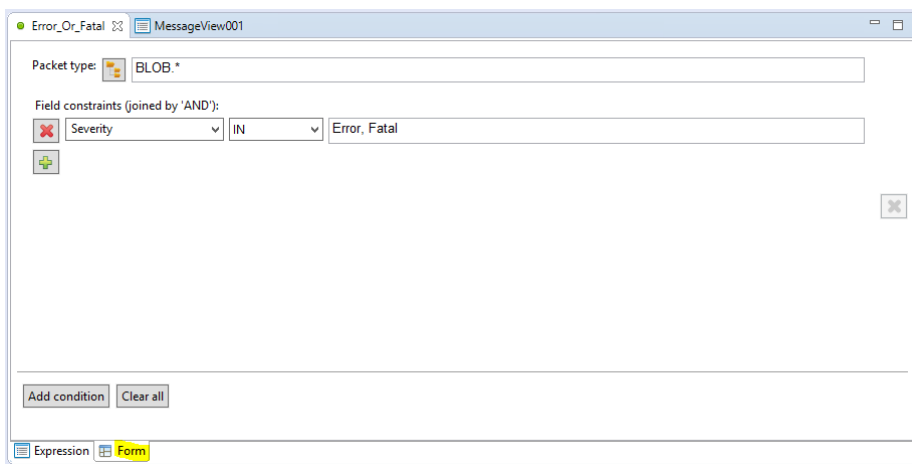
To create a **basic** scenario, select the **"Open Form"** option from the context menu. This



will provide you with a friendly GUI to define the search pattern (see section 7.10 for a detailed description of the form).



Note both the expression editor and the form are synchronized once you click the **"Save"** button.

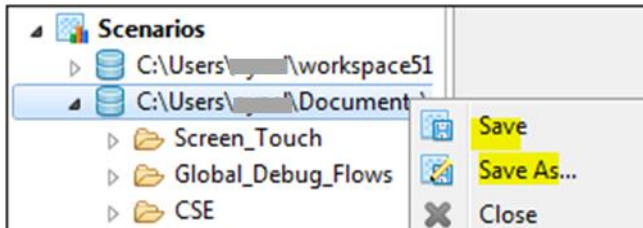


4. Validate and save the scenario

Click the editor's **"Save"** button to save the scenario. If you want to ensure that your scenario is compliant with TRAM language, select the scenario in the scenarios explorer and click the **"Validate"** option from the context menu. Save action will also validate the scenario.

5. Save the scenario in the TRAM repository file

Select the scenario repository from the explorer and click the **Save** or **Save As** from the context menu. You may e-mail the scenario file to others to share scenarios.






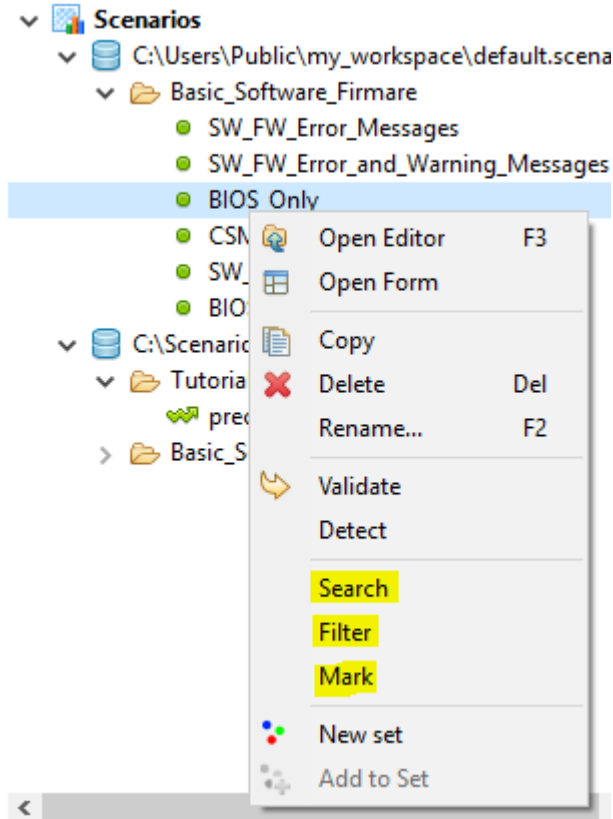
7.11.4 Scenario Detection

Once you define a scenario, you may detect it in the trace and perform measurement, as follows:

- Search/Filter/Markers (for basic scenarios only)
 - Search for a scenario in a given time window
 - Filter (keep) only messages that match the search criteria
 - Set markers for all messages that match the search criteria
- Detection
 - Detection of all occurrences of the scenario, display them and provide hyperlinks from results to the trace
- Feedback
 - Display the detection quality - percentage of events that occurred in each scenario occurrence.
 - Indication of what is missing in a scenario that is not of 100% quality
- Measurements
 - Information about occurrences of each event of each scenario
 - Information about occurrences of a scenario
 - Information on the latency between events within a scenario occurrence

7.11.4.1 Scenario Search, Filter, Marker

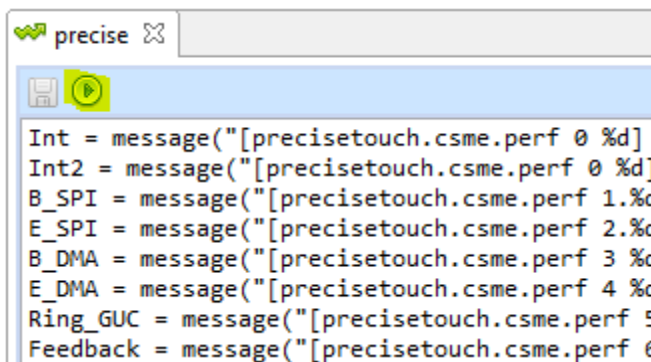
You may apply the operation of Search, Filter, Mark on **Basic** scenarios (denoted by the circle icon ).



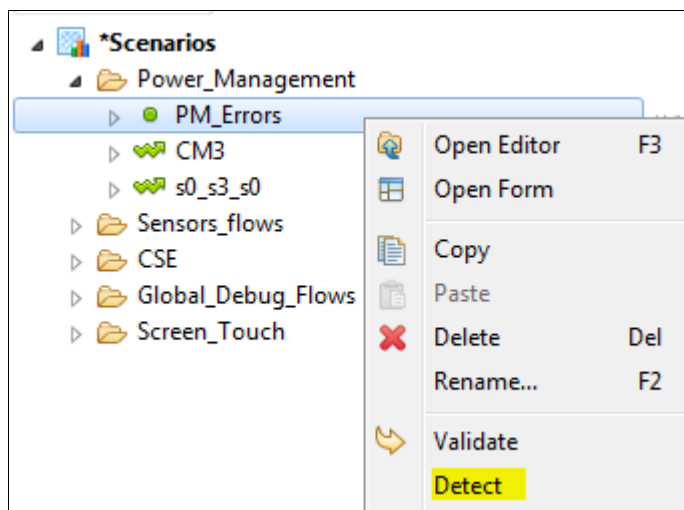
- . **Search:** finds all trace entries that match the criteria
- a. **Filter:** shows only trace entries that match the criteria
- b. **Mark:** sets markers to matching trace entries

7.11.4.2 Scenario Detection activation

To detect a scenario (Basic or not) in your trace click the **"Detect"** button in the scenario editor toolbar



or select the scenario from the scenario explorer and choose “**Detect**” from the context menu (right-click).

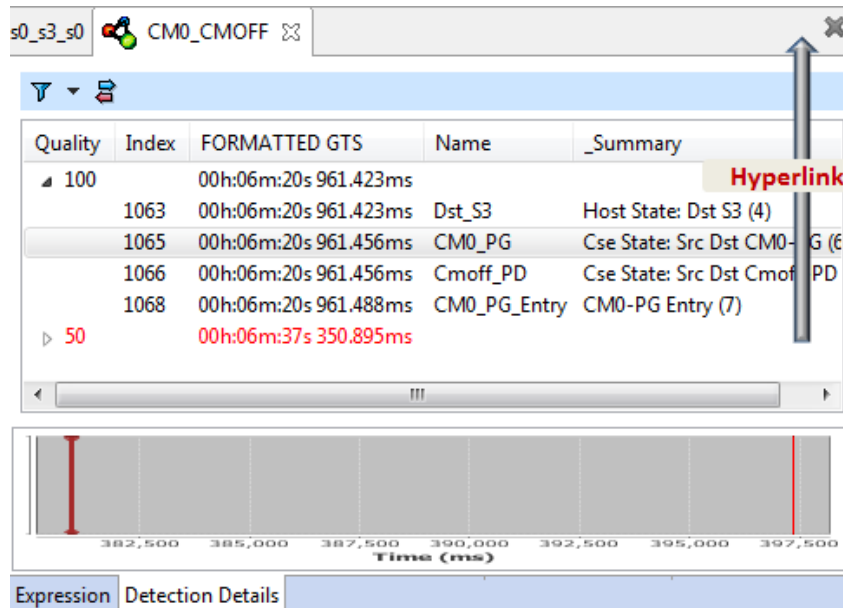



The processing happens in the background and progress is reported in the progress bar in the bottom-right corner. Once processing completes, you can view the results in the sub-tabs next to the expression of the scenario.

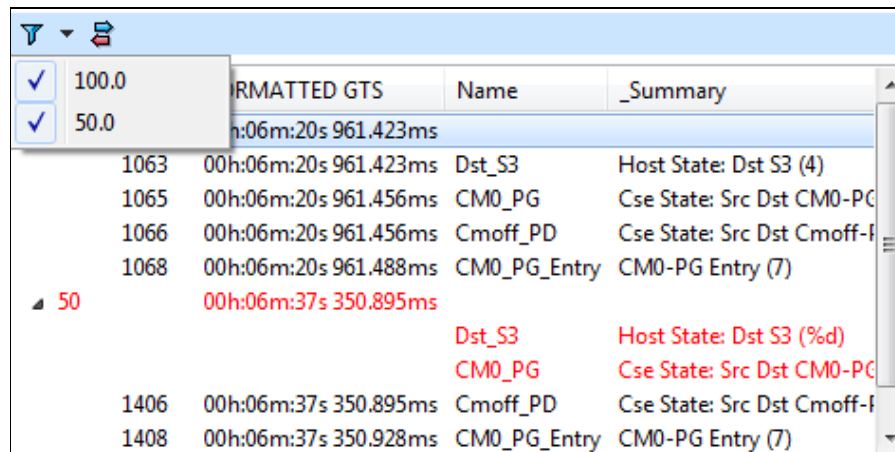
7.11.4.3 Detection Details Tab

This tab lists the occurrences of the scenario aside to a percentage denoting the quality of the scenario - the percentage of events (of the scenario) detected in the specific occurrence.

To see the occurrence of events within a specific scenario, expand the occurrence by clicking the small arrow next to the percentage. Clicking on an event will select the matching even in the Message View.



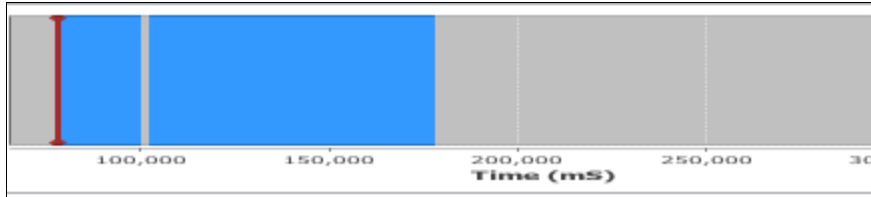
To view scenarios with quality less than 100%, click the filter button  and choose those that are not 100%. The events that are missing within a scenario (were not detected in the trace) are colored red. Events that exceeded defined time limit are colored in orange (see 7.11.7).



The scenario occurrence graph at the bottom displays the occurrences of the scenario over time.

- A blue line denotes a scenario having a quality of 100%.
- A red line denotes a scenario having a quality less than 100%.

Clicking on a certain location in the graph shows the corresponding occurrence in the result table



7.11.4.4 Timing Table Tab

This tab displays statistics on event and scenario occurrences.

- The **Events** table displays the number of occurrences (coverage) of each event within a scenario.
- The **Transitions** table displays statistics on transitions between different events within a scenario, including minimum, maximum and average delay, forward probability (how often a given branch path was taken) and number of occurrences.

Console Message Distribution Trace Analysis

precise

Trace: orca_1_3_5_10_fingers_2017_07_05_115419_TS

| Name | Occurrences |
|----------|-------------|
| Int | 1641 |
| Int2 | 0 |
| B_SPI | 2845 |
| E_SPI | 2845 |
| B_DMA | 1641 |
| E_DMA | 1641 |
| Ring_GUC | 1641 |

| Source | Target | Min Delay (ps) | Max Delay (ps) | Avg Delay (ps) | Delay (ps) | Fwd Prob. | Occurrences |
|--------|----------|----------------|----------------|----------------|------------|-----------|-------------|
| Int | Int2 | | | | | 0 | 0 |
| Int | B_SPI | 2038000 | 574319200 | 445533852 | | 1 | 1640 |
| Int2 | B_SPI | | | | | 0 | 0 |
| B_SPI | E_SPI | 256166400 | 2639016801 | 1685816229 | | 1 | 2844 |
| E_SPI | B_SPI | 144998001 | 1055703600 | 310302883 | | 0.423 | 1204 |
| E_SPI | B_DMA | 4018000 | 6101600 | 6615681 | | 0.577 | 1641 |
| B_DMA | E_DMA | 338497600 | 1285194400 | 360022607 | | 1 | 1641 |
| E_DMA | Ring_GUC | 329714001 | 896560001 | 338233685 | | 1 | 1641 |

Detection Details Timing Table Delay Between Occurrences Transitions Latency

- Expand transitions with occurrences > 0 to view detailed list of all occurrences.

Transitions

| Source | Target | Min Delay (ps) | Max Delay (ps) | Avg Delay (ps) | Delay (ps) | Fwd Prob. | Occurrences |
|--------|--------|----------------|----------------|----------------|------------|-----------|-------------|
| Int | Int2 | | | | | 0 | 0 |
| Int | B_SPI | 2038000 | 574319200 | 445533852 | | 1 | 1640 |
| Int2 | B_SPI | | | | | 0 | 0 |
| B_SPI | E_SPI | 256166400 | 2639016801 | 1685816229 | | 1 | 2844 |
| 63 | 66 | | | | 2639016801 | | |
| 43066 | 43104 | | | | 2215903201 | | |
| 69308 | 69346 | | | | 2212918401 | | |
| 252624 | 252662 | | | | 2212599200 | | |

Detection Details Timing Table Delay Between Occurrences Transitions Latency

- Sort the list of occurrences by delay, source or target (ascending or descending) by clicking on the corresponding header.
- Click on source or target entry to jump to the corresponding index in the message view.
- Transitions from scenarios with quality < 100% are colored in red.



7.11.4.5 Delay Between Occurrences Tab

This tab displays a bar chart with the delay (latency) between the occurrences of consecutive scenarios.



- The chart includes only occurrences of 100% quality.
- Use the popup menu or drag the mouse to the left/right to zoom-in/out.
- Hover with your mouse on the bars to see the actual values in tooltips.
- Click on the bars to jump to the occurrence's first index in the message view.

7.11.4.6 Transitions Latency tab


This tab displays a stacked bar chart of the occurrences of each scenario and the delay between the events within a scenario.



- The chart includes only occurrences of 100% quality.
- Use the popup menu or drag the mouse to the left/right to zoom-in/out.
- Hover with your mouse on the bars to see the actual values in tooltips.
- Click on the bars to jump to the occurrence's first index in the message view.

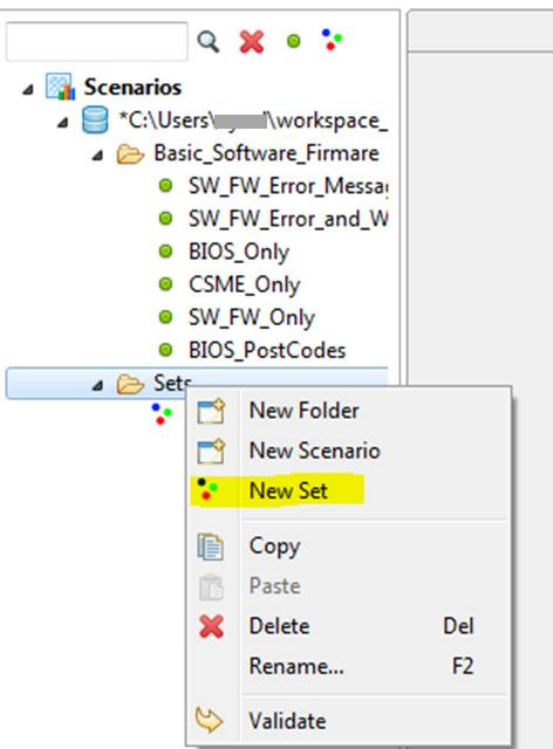
7.11.5 Scenario Sets


TRAM enables you to define a **set** of basic scenarios and then to associate different actions on top of them (Filter, Search, Mark).

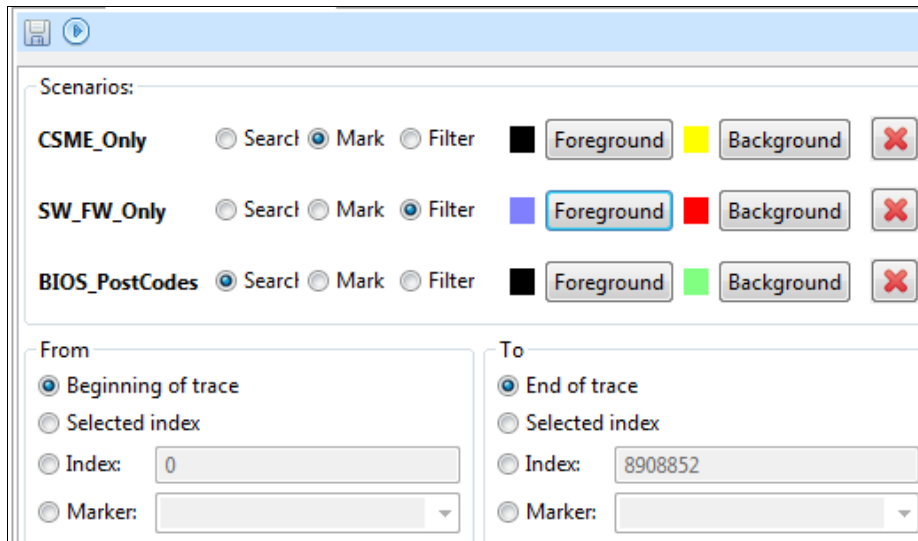
Basic scenarios are denoted by the circle icon  and are related to one trace entry and not to a flow.

To create a set of scenario do the following:

1. Select the desired folder
2. Select the **"New Set"** option from the context menu and give the set a name.



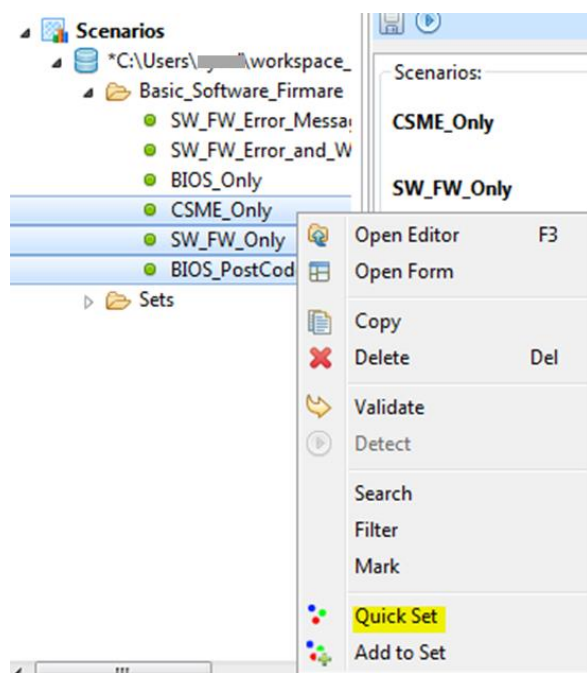
3. Open the set editor - select the set from the explorer and choose **"Edit"** from the context menu (or double click on the set name).
4. Drag and Drop the desired scenarios from the explorer into the set editor. Alternatively, select the desired scenarios and then select the **"Add to Set"** option from context menu. Remove scenarios from the set by clicking the delete button (.



5. Select the desired operation for each scenario within the set.
6. Select the desired coloring for each operation (optional).
7. Restrict the trace boundaries for your set operations to a certain time window (optional)
8. Click the **Save** button to save the set
9. Click the run button (▶) to apply your set to the trace displayed. Results will reflect the defined rules.

TRAM enables you to define a Quick Set without saving it in the explorer.

To create a Quick Set select the desired basic scenarios from the explorer and then select the "Quick Set" option from the content menu.

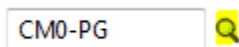


TRAM will create a temporary set, enabling you to select the desired operation, set colors and boundaries as described above, add more scenarios to the set, and run it as described above. You may save the set you created to the repository using the editor's "Save" button.

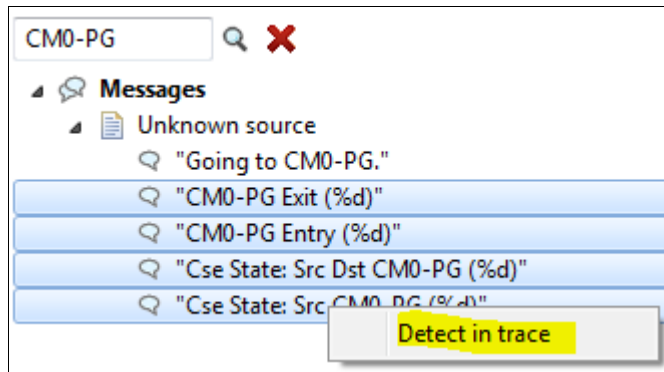
7.11.6 Quick Search

To search for message/messages in your trace (without defining a scenario) do the following:

- Locate the desired messages in the Message Explorer. You may use the explorer search for this.



- Select the desired messages.
- Choose "**Detect in trace**" from the context menu.



Your search results will appear under the **Search Result** tab with the following information:

- Summary of occurrences (the "**Summary**" tab) - Provides a table that elaborates the number of occurrences of each message

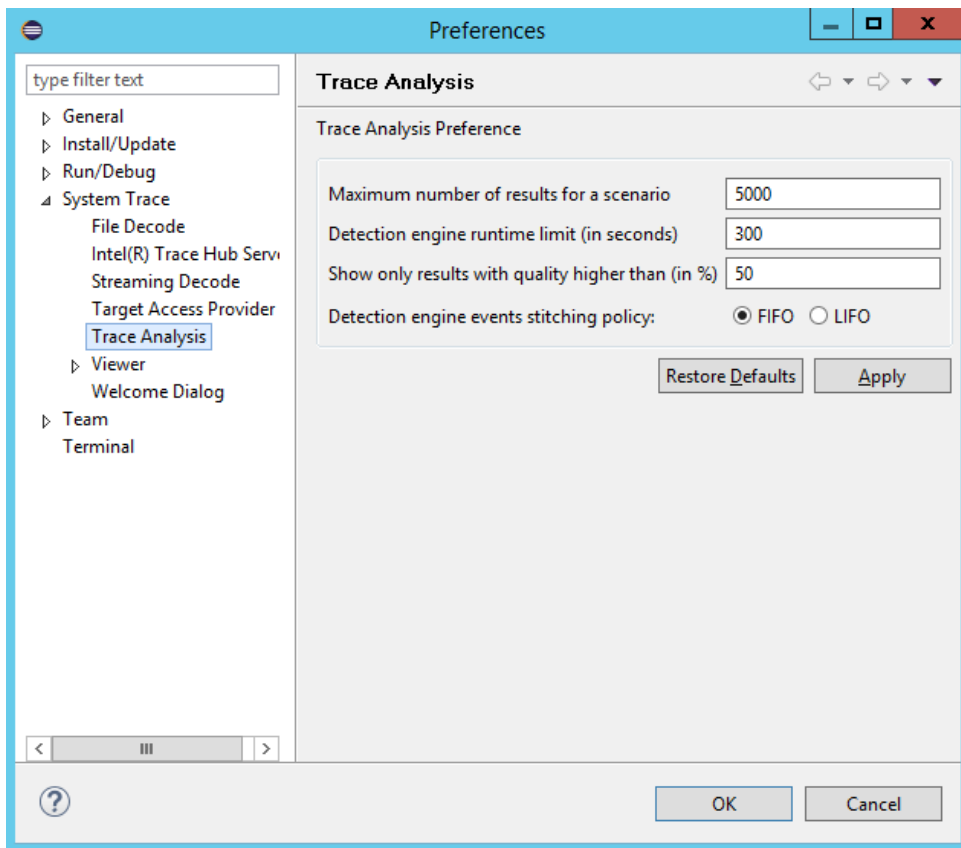
| s0_s3_s0 | CM0_CMOFF | Find messages 1 | |
|----------------------------------|-----------|-----------------|--|
| Data | Total | | |
| "CM0-PG Entry (%d)" | 2 | | |
| "CM0-PG Exit (%d)" | 1 | | |
| "Cse State: Src CM0-PG (%d)" | 0 | | |
| "Cse State: Src Dst CM0-PG (%d)" | 1 | | |

- Details on occurrences (the "**Details**" tab)
 - Lists each occurrence of each message (color coded)
 - To hyperlink to the relevant trace location, click the desired occurrence.

| s0_s3_s0 | CM0_CMOFF | Find messages 1 | |
|----------|-----------------------|-------------------------------|-----------|
| Index | FORMATTED GTS | Message | Hyperlink |
| 1065 | 00h:06m:20s 961.456ms | Cse State: Src Dst CM0-PG (6) | |
| 1068 | 00h:06m:20s 961.488ms | CM0-PG Entry (7) | |
| 1408 | 00h:06m:37s 350.928ms | CM0-PG Entry (7) | |
| 1563 | 00h:13m:36s 053.747ms | CM0-PG Exit (8) | |

7.11.7 Trace Analysis preferences

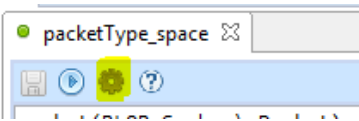
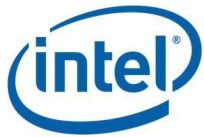
To configure TRAM detection default behavior, go to the Window -> Preferences -> System Trace -> Trace Analysis page:



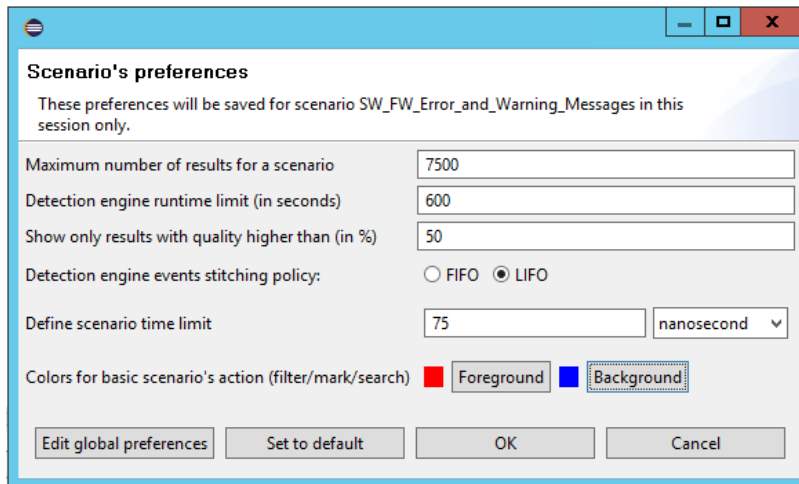
Here you can control the following parameters:

- Maximum number of detection results displayed - default set to 5000, max permitted - 50,000.
- Limit for detection engine runtime in seconds - default set to 300.
- Minimum threshold for detection results quality measured by percentage of scenario's events detected (see 7.11.4.3) - default set to 1% (show all).
- Detection stitching policy - stitch together events to same scenario by First-In-First-Out (FIFO) or Last-In-First-Out (LIFO) - default set to FIFO.

To override your default preferences for a specific scenario click the scenario's settings button in the scenario expression tab's toolbar:



This will open the scenario preferences dialog, where you can set specific values for this scenario:



These settings will be saved for your current session for the selected scenario.

There are two more preferences here in addition to the options mentioned above:

1. **Scenarios time limit** (optional): define a time limit for the scenario to complete. Occurrences that exceeded the specified time limit will be marked in orange in the detection details tab, as well as the specific events within that scenario's occurrence.

| Quality | Index | FORMATTED GTS | Name | Summary |
|---------|-------|----------------------------|----------|--|
| 100 | 34150 | 00:16:36.785,372,211,494ps | B_SPI | "[precisetouch.csme.perf 0 4341] Interrupt received from sensor" |
| 100 | 34352 | 00:16:36.795,382,107,898ps | B_SPI | "[precisetouch.csme.perf 1.1 4341] Begin SPI transfer of 3712 bytes raw data from touch IC" |
| 100 | 34556 | 00:16:36.805,369,850,702ps | B_SPI | "[precisetouch.csme.perf 2.1 4341] End SPI transfer of 3712 bytes raw data from touch IC" |
| 100 | 34758 | 00:16:36.824,255,842,309ps | B_SPI | "[precisetouch.csme.perf 1.2 4341] Begin SPI transfer of 3308 bytes raw data from touch IC" |
| 100 | 34758 | 00:16:36.824,255,842,309ps | Int | "[precisetouch.csme.perf 2.2 4341] End SPI transfer of 3308 bytes raw data from touch IC" |
| 100 | 34787 | 00:16:36.824,810,897,510ps | B_DMA | "[precisetouch.csme.perf 3 4341] Begin DMA transfer of 7084 bytes raw data to shared memory" |
| 100 | 34790 | 00:16:36.826,845,497,910ps | E_SPI | "[precisetouch.csme.perf 4 4341] End DMA transfer of 7084 bytes raw data to shared memory" |
| 100 | 34801 | 00:16:36.827,029,634,311ps | B_SPI | "[precisetouch.csme.perf 5 4341] Ring GUC doorbell" |
| 100 | 34806 | 00:16:36.828,871,395,911ps | E_SPI | |
| 100 | 34806 | 00:16:36.828,877,947,911ps | B_DMA | |
| 100 | 34811 | 00:16:36.829,235,538,711ps | E_DMA | |
| 100 | 34817 | 00:16:36.829,571,995,111ps | Ring_GUC | |
| 100 | 34834 | 00:16:36.831,673,795,512ps | | |
| 100 | 35155 | 00:16:36.841,996,323,116ps | | |
| 100 | 35371 | 00:16:36.853,292,273,521ps | | |

2. **Color preferences:** For basic scenarios you will also have the option to determine foreground and background colors for basic scenarios actions, such as search, filter and mark (see 7.11.4.1 above). These colors will also be used

as default colors for this scenario when added to a scenarios' set (see 7.11.5). The colors settings will be saved for the scenario between sessions as well.

7.12 Advanced Trace Analysis: Master/Channel Filtering

Traces from the specific masters and channels can be filtered by setting a decoder parameter in **File Decode Preferences** dialog and/or in the **Streaming Decode Preferences** dialog. Filtering channels might be important if a trace source has noisy traffic, where not all information is relevant for the user.

The filtering can be enabled by adding a decoder parameter to the MIPI decoder. The decoder parameter is named **masterChannelfilter** and has the following syntax:

<master>:<channels-to-filter>

The following expressions for <master> and <channels-to-filter> are valid:

- Single number
- Range using the "-" operator: <number-1>-<number-n>

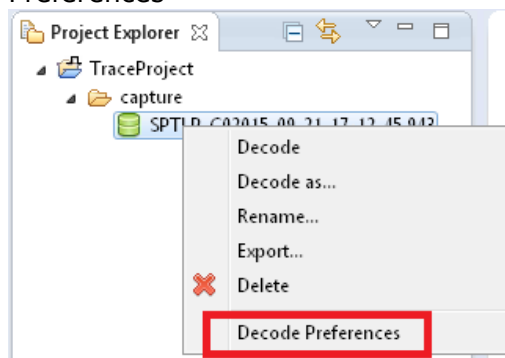
It is also allowed to combine multiple expressions by separating them with a comma, for example:

<master>:<channel>,<master>:<channel-1>-<channel-n>,<master-1>-<master-n>:<channel>,<master-1>-<master-n>:<channel-1>-<channel-n>

Example on master 3, which uses channels 1 to 10, filter out channels 5 to 8. To do this is to add a new **MIPI_Decoder masterChannelfilter parameter** using following steps:

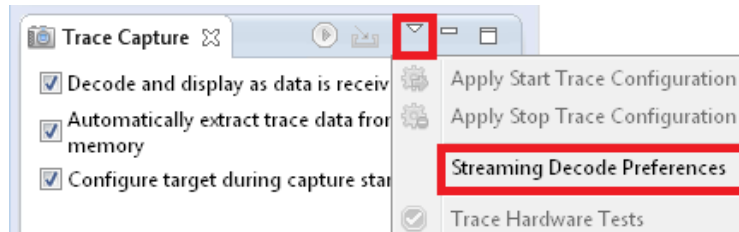
1. Open File Decode Preferences or Streaming Decode Preferences depending on the streaming mode:

Offline decode: Right-click on the capture file and then click on Decode Preferences

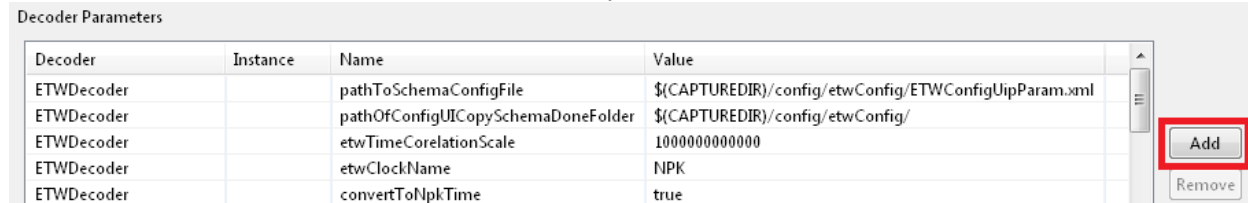




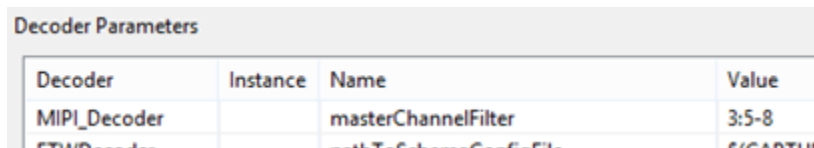
Live decode: Right-click on view menu “triangle” and then click on **Streaming Decode Preferences**



2. Press the **Add** button to add a new decoder parameter



7. Add a following line: MIPI_Decoder | filter | 3:5-8 as shown in the picture below

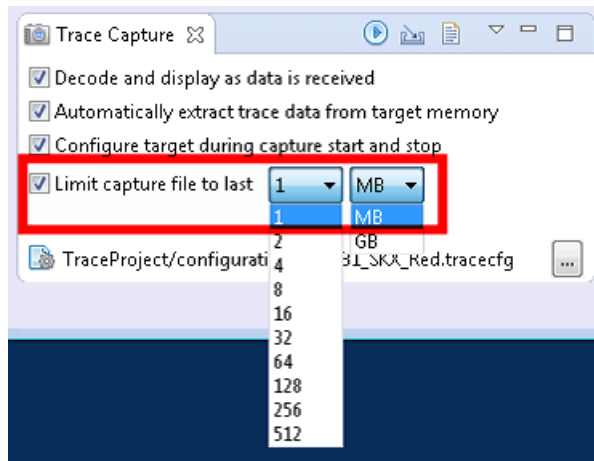


The value of the masterChannelfilter parameter for the MIPI decoder is 3:5-8, meaning that for master 3 channels 5, 6, 7 and 8 are filtered out and there will be no traces from these channels visible. More complex example would be 2-4:5-8 which filters on master 2 channels 5 to 8, on master 3 channels 5 to 8 and on master 4 channels 5 to 8.

Please note that filtering on channel 0 is not supported.

7.13 Advanced Trace Analysis: Trace Capture File Size Limit

Captured file size during live decode could be limited to a specific size. This is especially important for long running live traces where gigabytes of data are received. The user can choose to store on the hard drive only the last X Mbytes of the received data. Before starting capture live trace, select the desired capture file size in the **Trace Capture** view. When using a streaming destination, capture file size limit is implemented using a circular file buffer. When tracing to memory, only the specified amount of trace data is actually extracted, greatly improving extraction speed when running long traces.



7.14 Analyzing Chipset Firmware Flash Logs

The chipset firmware (FW) is able to store fatal FW events into SPI flash memory in addition to send them out to the Intel® Trace Hub. This mechanism was developed to provide post mortem access to firmware fatal errors even if no active capture using the Intel® Trace Hub was performed.

The chipset firmware Image Analyzer is parsing the SPI flash log contents and decodes any included fatal error messages.

7.14.1 Use Case Overview

1. A system in the manufacturing line is showing erratic behavior and a problem with chipset firmware of the management engine is suspected
2. The SPI flash content of this system is captured into a file using for example a Dediprog* programmer or the Intel ® Flash Programming Tool, which is part of the system tools in the Firmware kit.
3. The captured SPI flash image file can be added to the trace viewer for showing any fatal firmware messages that got stored there from former system runs.

7.14.2 Decoding Firmware Flash Logs

The chipset firmware Image Analyzer embedded in the trace viewer is used to parse SPI flash log and decode included fatal error messages. This is done in 2 steps:

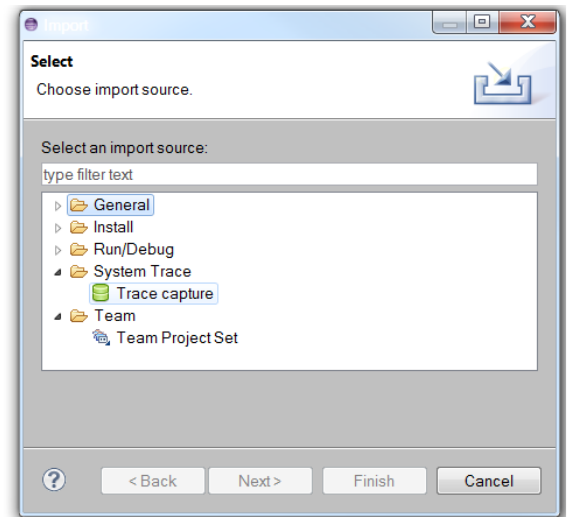
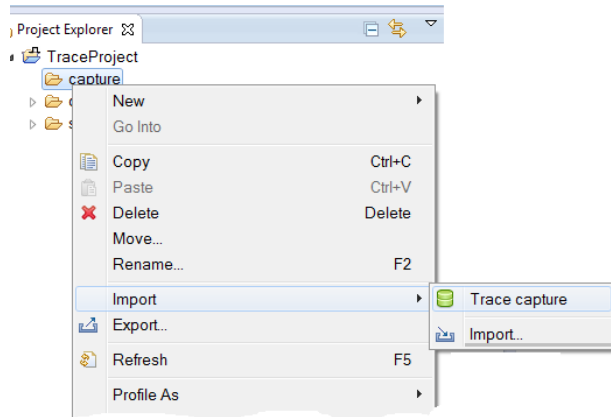
- Adding the firmware image file as a trace capture
- Running a custom decode on the Firmware image file

Adding the Firmware Image file as a trace capture:

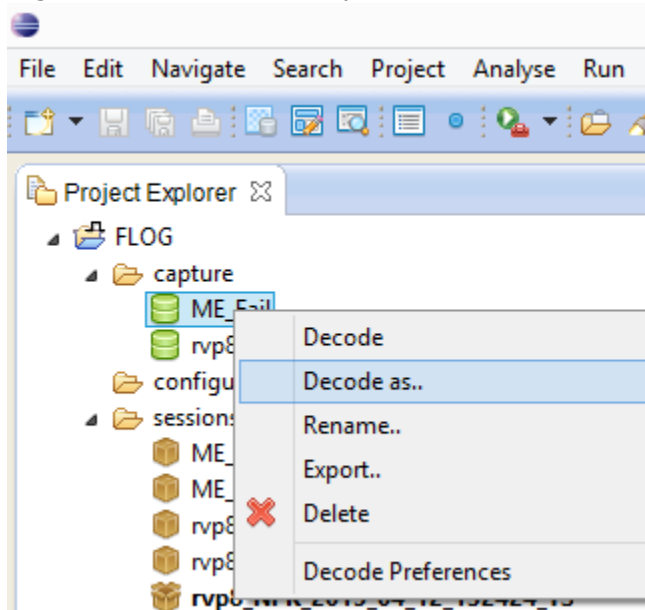


1. Select "Import" from the context menu of the Trace Project capture node or choose "File → Import → System Trace → Trace Capture"

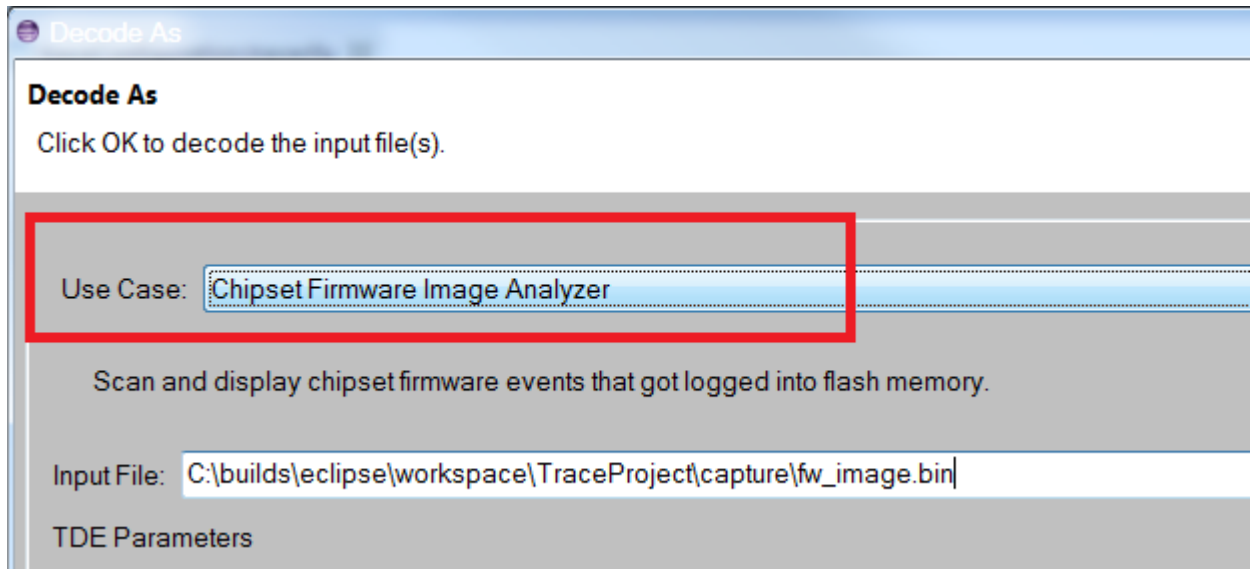
from the main menu.



2. Right click on the new capture and select **Decode as...**



3. In the **Decode As** dialog, select the "Chipset Firmware Image Analyzer" use case.

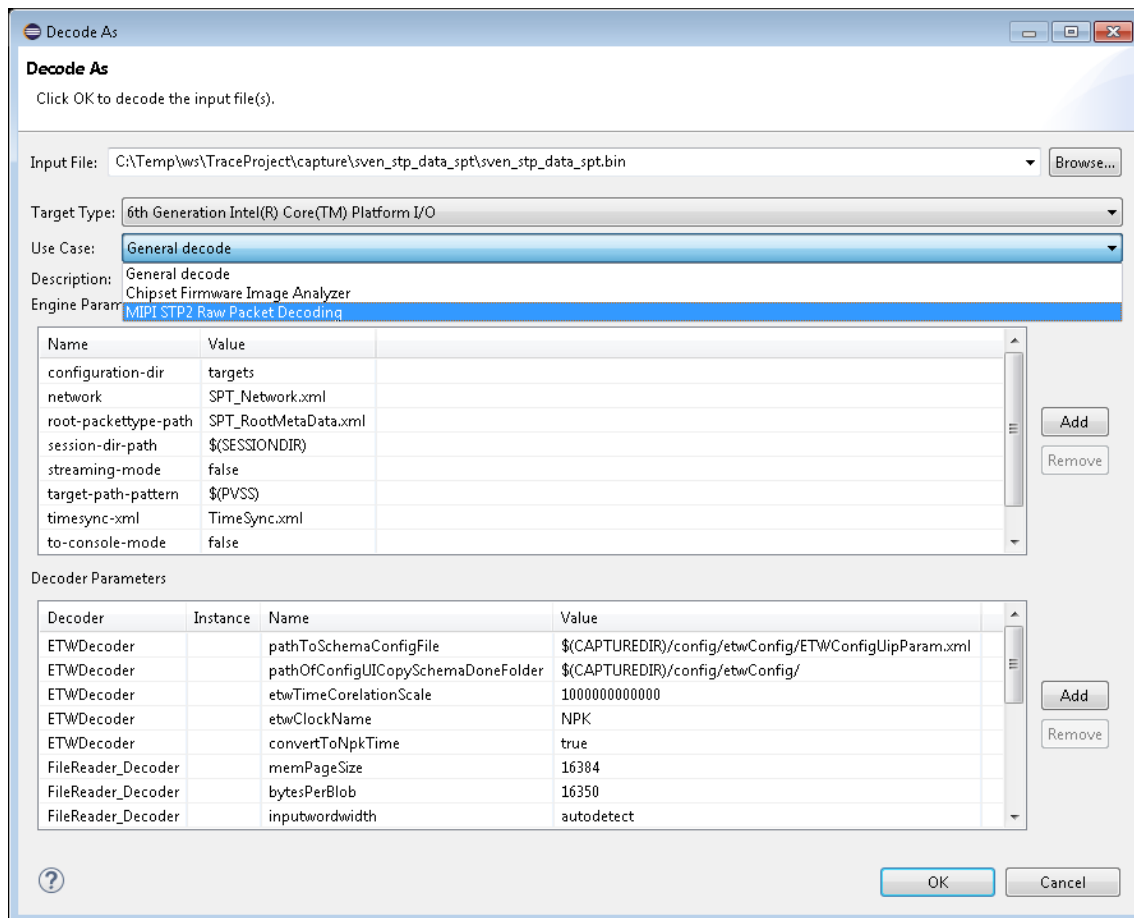
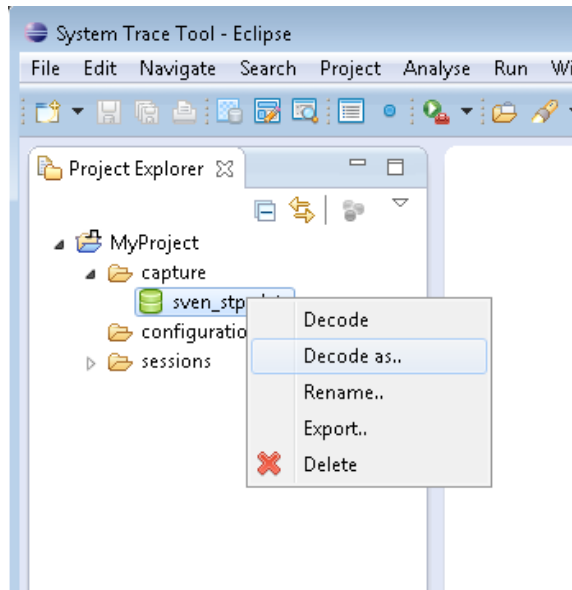


Select OK, the MessageView will contain all SPI flash log fatal entries.

7.15 Advanced Trace Analysis: Decode Trace with a Different Decode Network

To decode your trace with a different decode network, e.g. to decode just the raw MIPI packages, do the following:

1. In the **Project Explorer**, right-click on the captured trace file and select **Decode as...**
2. In the next dialog, select the use-case (network) you want to use and click OK.



7.16 Trace Analysis: Hints

7.16.1 Multiple Rules

Multiple rules can be active simultaneously, for example, for displaying all “Error” packets in red and all “Warning” packets in orange, or for filtering only for those two types of trace entries. When using multiple rules, make sure to use the “AND” and “OR” logical operators above each rule section in the **TMV Configurer** as required for your application.

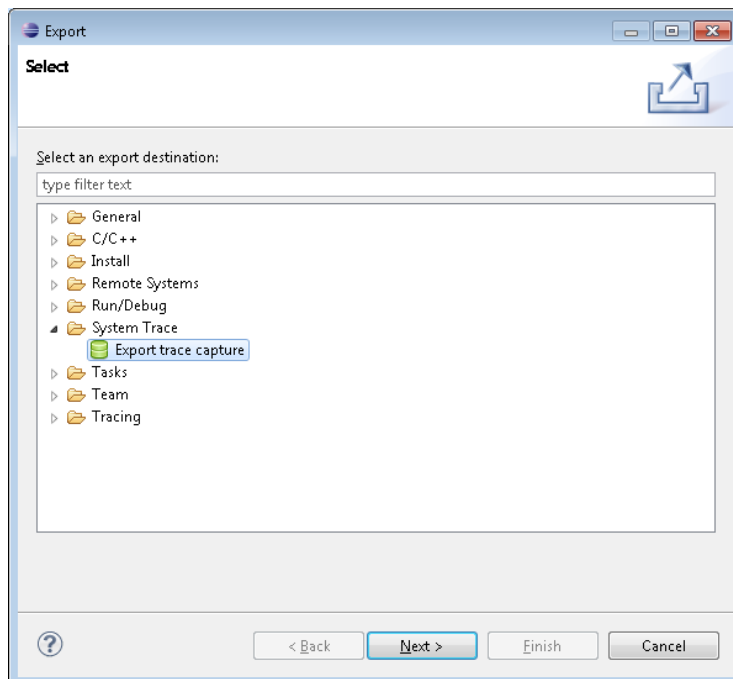
7.16.2 Copying & Pasting of Trace Entries

After filtering your trace, you may be interested in copying and pasting the resulting trace entry set into another application. To do so, select one or more trace entries, right click on the trace entries, select **Copy** and pick the method that best matches your use-case.

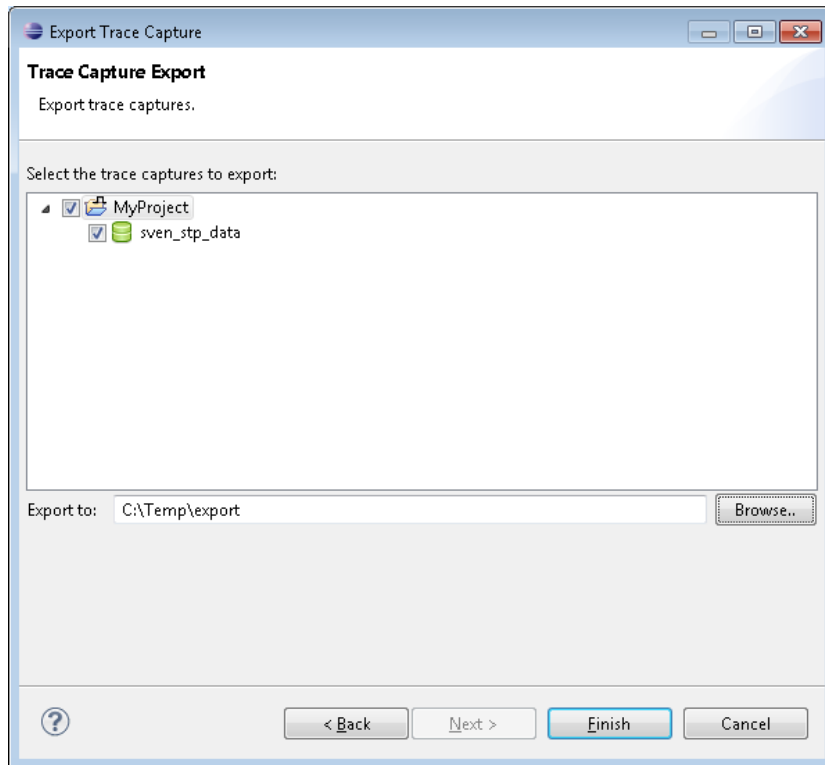
7.16.3 Exporting Traces

For further analysis, it is also possible to export traces by selecting **File > Export ...** from the menu bar.

In the **Export** dialog, browse to **System Trace > Export trace capture**.



Click Next and select the project you want to export and provide the location where you want to export the trace archive.

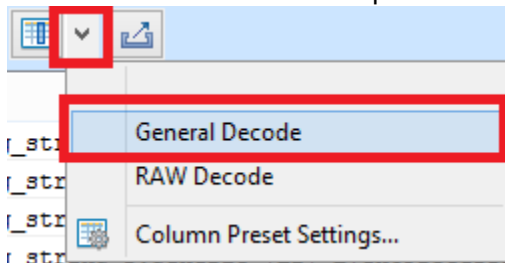


Click **Finish** to start the export. You should find a trace archive file (.tracecpt) in the directory you specified in the **Export** dialog.

7.16.4 Showing the Source of Traces

To verify from which trace source the decoded messages in the Message View are coming from, please do the following (assuming the Message View is open and displays messages):

- 1) In the Message View toolbar, click the small arrow near the column picker button to show the available column presets and select **General Decode**.



- 2) This column preset will show the **Mipi Source** column, which displays the origin of a trace message, e.g. in this case BIOS and CSME:

| <input checked="" type="checkbox"/> Synchronize <input type="checkbox"/> Auto-Scroll Go To Index 0 Filter On Search On | | | |
|--|-----------------------|-------------|--|
| Index | FORMATTED GTS | Mipi Source | _Summary |
| 671 | 00h:00m:07s 335.726ms | CSME | SVEN: 1:Normal t:debug_string s:generic [ED] RegisterProcess Ioctl |
| 672 | 00h:00m:07s 335.741ms | CSME | SVEN: 1:Normal t:debug_string s:generic [ED] "dal_lnc" RegisterProc |
| 673 | 00h:00m:07s 335.761ms | CSME | SVEN: 1:Normal t:debug_string s:generic [ED] Added Name "dal_lnc" F |
| 674 | 00h:00m:07s 336.436ms | CSME | SVEN: 1:Normal t:debug_string s:generic [LDR] LLM: "dal_lnc" proce |
| 675 | 00h:00m:07s 340.587ms | CSME | SVEN: 1:User1 t:debug_string s:generic [DAL_LNCH] Registered HECI c |
| 676 | 00h:00m:07s 343.177ms | CSME | SVEN: 1:Normal t:debug_string s:generic [ED] "dal_lnc" PID (8222) I |
| 677 | 00h:00m:07s 343.220ms | CSME | SVEN: 1:Normal t:debug_string s:generic [ED] "dal_lnc" PID (8222) S |
| 678 | 00h:00m:07s 343.684ms | CSME | SVEN: 1:Normal t:debug_string s:generic [ED] RegisterForEvents Ioctl |
| 679 | 00h:00m:07s 471.787ms | BIOS | SVEN: 1:Normal t:debug_string s:generic POSTCODE=<0000004F> m:BAE7A |
| 680 | 00h:00m:07s 622.611ms | BIOS | SVEN: 1:Normal t:debug_string s:generic POSTCODE=<00000035> m:BAE7A |
| 681 | 00h:00m:07s 637.245ms | BIOS | SVEN: 1:Normal t:debug_string s:generic POSTCODE=<00000036> m:BAE7A |
| 682 | 00h:00m:07s 638.422ms | BIOS | SVEN: 1:Normal t:debug_string s:generic SmmBaseRelocate: Start m:BA |
| 683 | 00h:00m:07s 671.711ms | CSME | SVEN: 1:Normal t:debug_string s:generic [ED] RegisterProcess Ioctl |
| 684 | 00h:00m:07s 671.726ms | CSME | SVEN: 1:Normal t:debug_string s:generic [ED] "dal_ivm" RegisterProc |
| 685 | 00h:00m:07s 671.817ms | CSME | SVEN: 1:Normal t:debug_string s:generic [ED] Added Name "dal_ivm" F |
| 686 | 00h:00m:07s 678.876ms | CSME | SVEN: 1:Normal t:debug_string s:generic [LDR] LLM: "dal_ivm" proces |

7.17 Accessing the Online Help

The System Trace provides an online help directly available in Eclipse*. To access this help go to **Help > Help Contents**. Browse to **System Trace** to show the online help.

8 TRAM Language

TRAM language enables you to define scenarios (system behaviors) from simple ones relating to complex flows and use them for search, filter, or advanced analysis.

The language is compositional and it is built on the following levels

1. **Events** – a single packet/message with constraints to its fields/parameters
2. **Scenarios** – flow over time on top of events defined by means of sequence representation or FSM representation
3. **Scenarios on top of scenarios** – use scenarios as building blocks to build other scenarios

8.1 Events

An event is a special case of a scenario referring to a single message or packet.

8.1.1 Event Syntax:

The syntax to define an event is:

```
<event_name> = packet (<packet constraints >, < field constraints >);
```

- Field constraints are optional
- event_name is an alphanumeric identifier.

Example:

```
E1 = packet (Blob || ETW, Status == Error && Severity == 7);
```

8.1.2 Packet Constraints

Packet constraints represent a pattern denoting the packet type.

- The packet constraints supports '*' token as "all".
- You may use "*" to denote any packet.
- Use the OR sign ("||") to denote multiple packets.

Example:

```
//ETW or BLOB packets with error status
E1 = packet (ETW || BLOB, PacketStatus == Error);

//all types SVEN packets
E1 = packet ( *SVEN*, Severity != User99 );

//all packets with error
E5 = packet ( * , Severity == Error );
```

8.1.3 Field Constraints

A constraint on a field is defined by one of the following:

- field_name operation <value>
 - Operation is one of: == , != , >= , <= , > , <
- field_name in [x : y]
 - [x : y] range of values
- field_name in [set of values separated by comma]
- field_name contains <string>
- field_name begins-with <string>
- field_name ends-with <string>
- field_name =~ /<reg-exp>/
- exists(field_name)
- format (<field_name>, "<field-format >", <format constraints>)
 - Comes to set constraints to "printf" fields (see section 8.1.4)

Notes:

- The Boolean operators allowed between field constraints are "&&" (and), "||" (or), and "!" (not).
- Numerical values are treated as unsigned integers.
- <value> could be numeric, string, or mnemonic (for fields that have mnemonic value).
- <value> could be a name of another field.



- <String> is a quoted string using single quote or double quote.
- field_name may include a bit range (for vector fields)
 - field_name[upper_bit:lower_bit]
 - field_name[bit]

Examples:

```
//all AAA packets with ADDR in a given range
E3 = packet (AAA*, ADDR in [PAGE1 : PAGE7]);

//all BLOB packets with error status due to halt or shutdown
E5 = packet (BLOB, REASON in [HLT, SHUTDOWN] && PacketStatus == Error);

//message that begins with ERR
E6 = packet (BLOB.*, Summary begins-with "ERR");
```

8.1.4 Format Constraints

A format constraint enables you to set constraints to formatted fields (e.g., `printf`).

For example, set constraints to restrict the second parameter of the following `printf` **"Printer number %d transfers %d bytes"**

The syntax for format constraints is:

```
format (<field_name>, "<field_format >", <format_constraints>)
```

- <Field_name>
 - The name of the field
- <field_format>
 - The formatted string that the field should match
- <format_constraints>
 - Is a Boolean expression on top of parameter constraints
 - The parameters are those that appear in the field-format.



- A parameter is identified by a “%i”, where “i” is the position of the parameter (%0 for the first parameter, %1 for the second parameter, etc.).
- The following parameter constraints are allowed:
 - Parameter_number **operation** <value>
 - Operation is one of: **==** , **!=** , **>=** , **<=** , **>** , **<**
 - Parameter_number **in** [X:Y]
 - [x : y] range of values
 - Parameter_number **in** [set of values separated by comma]
 - Parameter_number **contains** <string>
 - Parameter_number **begins-with** <string>
 - Parameter_number **ends-with** <string>
 - Parameter_name **=~** /<reg-exp>/

Examples:

```
//number of bytes transferred is greater than 0.
E8 = packet (BLOB*, format(payload, Printer number %d transfers %d bytes," %1 > 0)
&& Severity == USER1)
```

8.1.5 Message Macro

The **message** macro streamlines the definition of events that refer to SVEN packets with constraints on the payload field.

The format is:

```
<event_name> = message(<printf_message>, < printf parameter constraints >);
```

For <printf parameter constraints>, see section 8.1.4.

Example:

```
//two equivalent events
E1 = packet(*SVEN*, format(payload, "[Printer number %d transfers %d bytes," %1
```



```
== 7));  
E2 = message ("Printer number %d transfers %d bytes", %1 == 7);
```

8.2 Scenario Definition

A scenario is defined by a series of declarations, defining the relevant events, and a flow definition specifying the time relationship between the previously declared events. A flow can be a sequence or an FSM. If omitted, linear order is assumed between the events in the declaration part.

8.2.1 Event Declaration

Event declaration is a list of events definitions separated by ";".

- <event_name> = <inline_event_specification>;
- <event_name> = reference(<predefined_scenario_name>);

Example:

```
//event declaration  
E1 = message ("printer is preparing to restart");  
E2 = message ("printer is killing all active jobs");  
E3 = message ("printer is restarting");  
E4 = reference (/camera/power-flows/camera-turns-off);
```

8.2.2 Sequence Definition

The syntax is as following:

```
sequence (sequence_expression);
```

Sequence_expression is as follows:

| | Semantics | Constructs |
|--------------------|-----------------------------|--|
| E | E occurs | E is an event defined in the declaration section |
| E1 => E2 | E2 occurs after E1 | E1 and E2 are sequences |
| E* | E occurs zero or more times | E is a sequence |
| E* | E occurs one or more times | E is a sequence |
| E? | E occurs zero or one time | E is a sequence |
| E1 E2 | E1 or E2 occur | E1 and E2 are sequences |

The order of evaluation is:

- *,+,?
- =>
- ||

Known limitations:

1. Repeat operators (*, +, ?) cannot be located at the end of a sequence.
2. Any operator other than sequence (=>) does not work properly on top of *reference* event.
3. Trace entries will match no more than one event. If an entry matches more than one of the sequence's events, behavior may be ambiguous.

Example:

```
//event declaration

E1 = message ("printer is preparing to restart");
E2 = message ("printer is killing all active jobs");
E3 = message ("printer %d is restarting sending %d packets");

//sequence definition

//printer may send several message during preparation to restart
sequence (E1+ => E2 => E3);
```



```
//event declaration  
E1 = message ("prepare to send data");  
E2 = message ("start send packet");  
E3 = message ("end send packet");  
E4 = message ("data transaction is done");  
  
//sequence definition  
//loop of start-end of packet sending  
sequence (E1 => (E2 => E3)+ => E4;
```

8.2.3 FSM Definition

An FSM is defined by the following construct:

```
fsm (<transitions list>)
```

The transition list is of the following form:

```
<from_state_name> ( transition_condition ) => <to_state_name>
```

- <from_state_name> is any alphanumeric name denoting the state name
- <to_state_name> is any alphanumeric name denoting the state name.
- The transition_condition is an event name declared in the declaration section.
- The start state transition will not have a from_state.
- The end state transition will not have a to_state.
- An initial state can also be a final state.

Example:

```
//event declaration  
E1= message ("prepare to send data");  
E2= message ("start send packet");  
E3= message ("end send packet");  
E4= message ("data transaction is done");
```

```
//FSM definition
```

```
fsm(
```

```
    => S0,
```

```
    S0 (E1) => S1,
```

```
    S1 (E2) => S2,
```

```
    S2 (E3) => S3,
```

```
    S3 (E4) => S4,
```

```
    S3 (E3) => S2,
```

```
    S4 =>
```

```
)
```

8.3 Stitch Variables

Stitch variables define a relationship between parameters of a message or fields. The format is as follows:

```
field_parameter_name == @<stitch_variable_name>
```

- variable_name is alphanumeric string.
- First usage must use the "==" operator (where the variable value is **assigned**).
- Further usages of the variable keep the first assigned value.
- A field/parameter name can be constraints using a stitch variable using the following:
 - Field/parameter_name == <stitch_variable_name>
 - Field/parameter_name != <stitch_variable_name>
 - Field/parameter_name >= <stitch_variable_name>
 - Field/parameter_name <= <stitch_variable_name>
 - Field/Parameter_name > <stitch_variable_name>
 - Field/Parameter_name < <stitch_variable_name>

Examples:



```
// reading and writing data to the same location
E3 = message ("sending data to location %d," %0==@X);
E4 = message ("reading data from location %d," %0 == @X);

//inconsistent number of bytes (to be used to detect bugs)
E3 = message ("server %d is waiting for %d bytes," %0==@X, @1 == Y);
E4 = message ("server %d received %d bytes," %0 == @X, @1 != Y);

//printer transaction - stiching on a certain printer, certain server
E1 = message ("printer %d is preparing to restart," %0==@P);
E2 = message ("printer %d is killing all active jobs," %0==@P);
E3 = message ("printer %d is sending %d bytes to server %d," %0==@P && %1 == @S);
E4 = message ("server %d received %d bytes from printer %d," %0==@S && %1 == @P);
E5 = message ("server %d approves restart of printer %d," %0==@P);
E6 = message ("printer %d is restarting," %0==@P);
sequence (E1 => E2 => E3 => (E4 => E5) +=> E6);

//reading and writing data to the same location
E3 = message ("sending data to location %d," %0==@X);
E4 = message ("reading data from location %d," %0 == @X);

//inconsistent number of bytes (alert for bug)
E3 = message ("server %d is waiting for %d bytes," %0==@X, @1 == Y);
E4 = message ("server %d received %d bytes," %0 == @X, @1 != Y);
```