FrameSoC Workbench User Guide

Generoso Pagano

INRIA REPORT

June 2014

Project-Team MESCAL

FrameSoC Workbench User Guide

Generoso Pagano [*]

Project-Team MESCAL

Inria Report — June 2014 — 17 pages

Abstract: This guide describes the FrameSoC workbench, delivered within the SETv1.0.

[*] INRIA, generoso.pagano@inria.fr

RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée
655 avenue de l'Europe Montbonnot
38334 Saint Ismier Cedex

Table of contents

1 Introduction

FrameSoC, the SoC-Trace project [1] trace management infrastructure, is composed by three main layers, as shown in Figure 1.

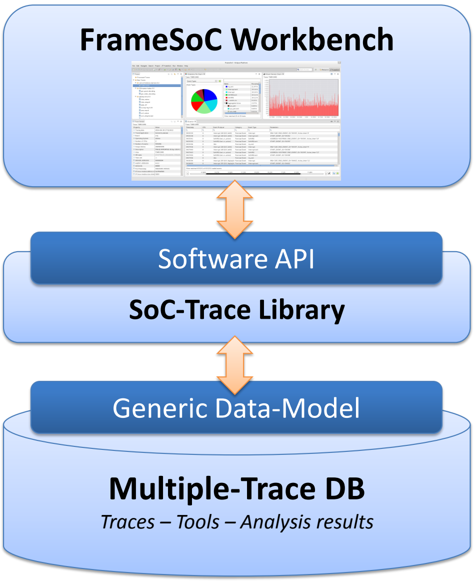
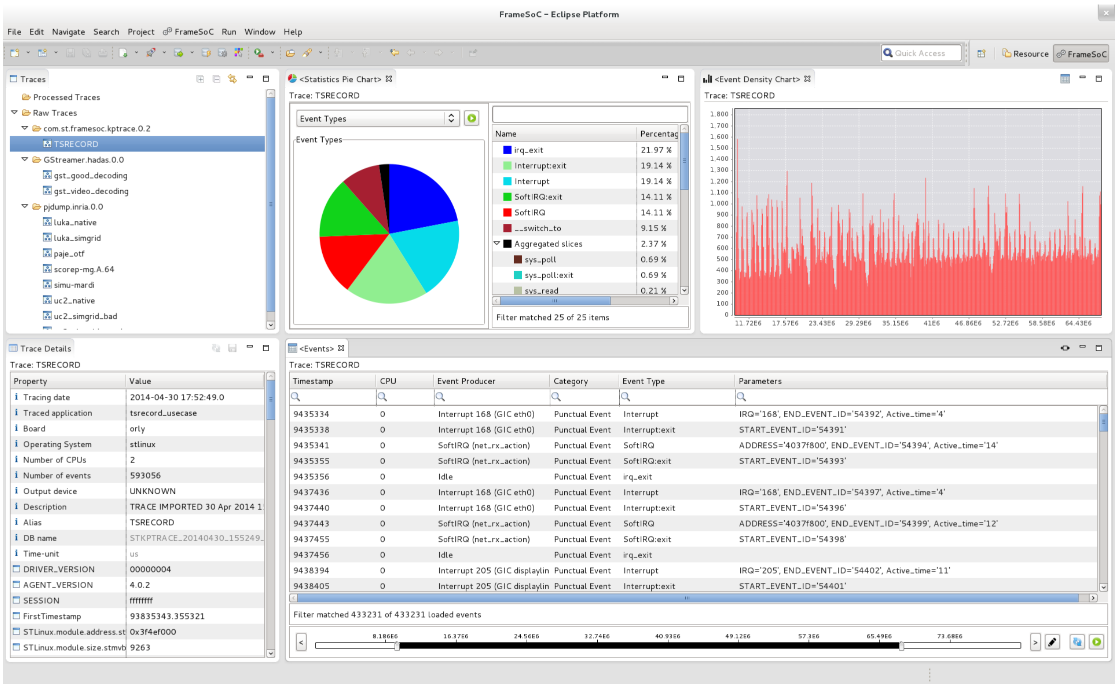


Figure 1 – FrameSoC Architecture

At the lowest level, there is a Multiple-Trace database, which stores all the data concerning traces, analysis tools and the analysis result produced by thees tools, using a generic data-model. Above this layer, the SoC-Trace library provides a software API to facilitate the interaction with the data-model and help the implementation of trace tools. On top of this software library, the FrameSoC workbench provides a graphical user environment, facilitating trace management, basic trace analysis and tool management.

This guide describes the FrameSoC workbench from the user perspective. More information about the lowest to layers of FrameSoC infrastructure can be found in the technical report RT-427 [2] and the research report RR-LIG-046 [3].

2 FrameSoC Perspective



Name	Percentag
🟦 irq_exit	21.97 %
🟩 Interrupt:exit	19.14 %
🟦 Interrupt	19.14 %
🟩 SoftIRQ:exit	14.11 %
🟥 SoftIRQ	14.11 %
🟥 __switch_to	9.15 %
■ Aggregated slices	2.37 %
🟫 sys_poll	0.69 %
🟦 sys_poll:exit	0.69 %
■ sys_read	0.21 %

Filter matched 25 of 25 items

Trace: TSRECORD

Property	Value
Tracing date	2014-04-30 17:52:49.0
Traced application	tsrecord_usecase
Board	orly
Operating System	stlinux
Number of CPUs	2
Number of events	593056
Output device	UNKNOWN
Description	TRACE IMPORTED 30 Apr 2014 1
Alias	TSRECORD
DB name	STKPTRACE_20140430_155249_
Time-unit	us
DRIVER_VERSION	00000004
AGENT_VERSION	4.0.2
SESSION	ffffffff
FirstTimestamp	93835343.355321
STLinux.module.address.st	0x3f4ef000
STLinux.module.size.stmvb	9263

Trace: TSRECORD

Timestamp	CPU	Event Producer	Category	Event Type	Parameters
9435334	0	Interrupt 168 (GIC eth0)	Punctual Event	Interrupt	IRQ='168', END_EVENT_ID='54392', Active_time='4'
9435338	0	Interrupt 168 (GIC eth0)	Punctual Event	Interrupt:exit	START_EVENT_ID='54391'
9435341	0	SoftIRQ (net_rx_action)	Punctual Event	SoftIRQ	ADDRESS='4037f800', END_EVENT_ID='54394', Active_time='14'
9435355	0	SoftIRQ (net_rx_action)	Punctual Event	SoftIRQ:exit	START_EVENT_ID='54393'
9435356	0	Idle	Punctual Event	irq_exit	
9437436	0	Interrupt 168 (GIC eth0)	Punctual Event	Interrupt	IRQ='168', END_EVENT_ID='54397', Active_time='4'
9437440	0	Interrupt 168 (GIC eth0)	Punctual Event	Interrupt:exit	START_EVENT_ID='54396'
9437443	0	SoftIRQ (net_rx_action)	Punctual Event	SoftIRQ	ADDRESS='4037f800', END_EVENT_ID='54399', Active_time='12'
9437455	0	SoftIRQ (net_rx_action)	Punctual Event	SoftIRQ:exit	START_EVENT_ID='54398'
9437456	0	Idle	Punctual Event	irq_exit	
9438394	0	Interrupt 205 (GIC displaylin	Punctual Event	Interrupt	IRQ='205', END_EVENT_ID='54402', Active_time='11'
9438405	0	Interrupt 205 (GIC displaylin	Punctual Event	Interrupt:exit	START_EVENT_ID='54401'

Filter matched 433231 of 433231 loaded events

FIGURE 2 – FrameSoC Eclipse Workbench

The FrameSoC workbench is an Eclipse application providing a perspective [1] for trace management and analysis. The FrameSoC perspective (Figure 2) contains the following elements:

- A FrameSoC menu (on top).
- A toolbar (the second toolbar, from the left).
- A set of views for trace management and analysis. We find on the left the two *management views*: a trace browser on top and a trace metadata viewer at the bottom. On the right there are three *analysis views*: a statistics pie chart and an event density chart on top, and an event table at the bottom.

The management views (trace browser and metadata viewer) refer to the whole system and are the entry point for trace analysis. Therefore they cannot be closed and there can be only a single instance of each of them. On the contrary, all the analysis views refer to a single trace, so they can be opened and closed as needed. For each trace, there can be an instance of each type of analysis view. The maximum number of open instances for a given type of analysis view is configurable (Appendix A). When a trace is selected in the trace browser, the corresponding metadata are shown in the metadata viewer and all the analysis view referring to that trace (if any) are highlighted. Namely, the view name is surrounded by < and >. In the example shown in Figure 2, all analysis views refer to the selected trace `scorep-mg.A.64`). On the other hand, when an analysis view is given focus, the trace shown in that view becomes the selected trace in the trace browser, so the metadata viewer is updated accordingly and all the analysis views are

[1] Within an Eclipse application, a perspective defines an initial set and layout of views, menu and toolbars. The FrameSoC Eclipse perspective can be activated by selecting it from the **Window** > **Open Perspective** > **Other...** menu.

consistently highlighted or unhighlighted.

The following sections describe in detail the FrameSoC management and analysis views, as well as the functionalities accessible via the FrameSoC menu and toolbar.

3 Management Views

3.1 Trace Browser

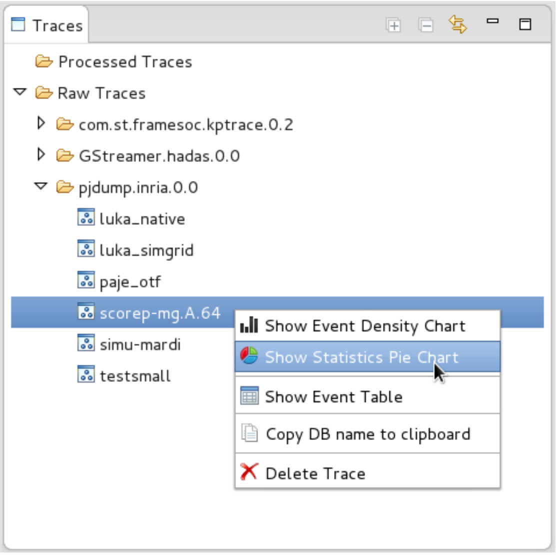


FIGURE 3 – Traces view

The *Traces* view (Figure 3) is the FrameSoC perspective trace browser. Traces are presented in a tree viewer with a two-level hierarchy. The first level distinguishes processed traces from raw traces [2]. Then, in each of these two categories, traces are grouped by type. The type may relate to the trace format or to the tool that created the trace. The viewer presents a trace alias for each trace.

Right-clicking on a single trace gives access to the following functionalities:

- Show Event Density Chart: open the density chart view for this trace (see Subsection 4.2).
- Show Statistics Pie Chart: open the pie chart view for this trace (see Subsection 4.1).
- Show Event Table: open the event table view for this trace (see Subsection 4.3).
- Copy DB name to clipboard: copy to the clipboard the name of the database containing the trace raw data.
- Delete Trace: delete the trace from the system.

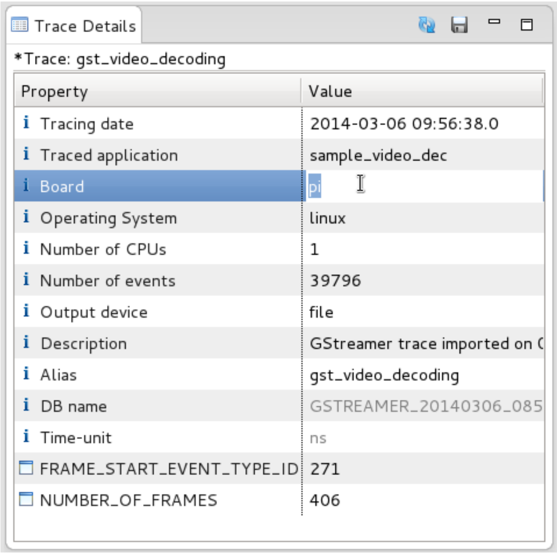
If more traces are selected, only the *Delete Trace* entry is displayed in this context menu. Note that if one or more traces are deleted, an update notification is sent to the other views. This allows a view to do the necessary actions if the trace it displays has been deleted.

The view toolbar contains three buttons: a couple of buttons to expand/collapse the trace hierarchy and a button to manually resynchronize the displayed traces with the information

[2] As described in the technical report RT-427 [2], a processed trace is a trace created by an analysis tool as the result of an analysis on another trace.

contained in the FrameSoC System DB (see the technical report RT-427 [2] for further details on the database architecture).

3.2 Trace Metadata Viewer/Editor



*Trace: gst_video_decoding

Property	Value
i Tracing date	2014-03-06 09:56:38.0
i Traced application	sample_video_dec
i Board	pi
i Operating System	linux
i Number of CPUs	1
i Number of events	39796
i Output device	file
i Description	GStreamer trace imported on (
i Alias	gst_video_decoding
i DB name	GSTREAMER_20140306_085
i Time-unit	ns
□ FRAME_START_EVENT_TYPE_ID	271
□ NUMBER_OF_FRAMES	406

FIGURE 4 – Trace Details view

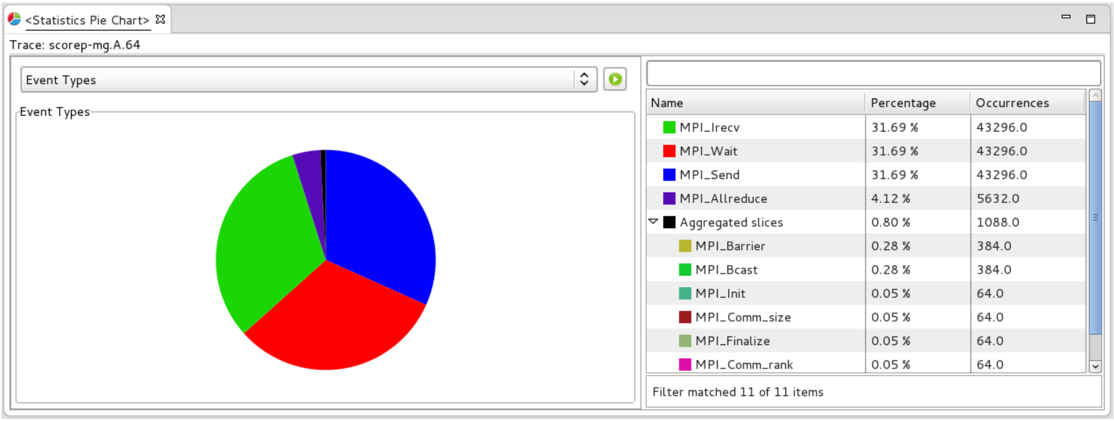
The *Trace Details* view (Figure 4 is the FrameSoC perspective trace metadata viewer and editor. When a trace is open or selected in the *Traces* view, the trace metadata are displayed in a table viewer containing two columns: the property name and the property value. The different properties are grouped in two different categories: predefined properties (displayed first) and custom properties (displayed last). Those categories are identified by two different icons. The *Value* column is normally editable, with the exception of some properties that are read only (the database name and the time unit). In order to edit an editable property value, you simply have to click the value and modify it (as shown in Figure 4 for the *Board* property). When one or more values have been modified, a star (*) is displayed before the trace alias, on top of the table, and the two view toolbar buttons are enabled. The *reset* button restores all non-saved edited properties to their previous value. The *save* button stores the changes. If the *Alias* predefined property of a trace is persistently modified, other views are notified in order to take the necessary actions (e.g., update their label for the trace).

Note that if more than one trace is selected in the *Traces* view, the metadata view displays only the properties having the same name and the same value. For these properties the editing support is still working, with the note that all the selected traces metadata are modified.

4 Analysis Views

4.1 Statistics Pie Chart

The *Statistics Pie Chart* view (Figure 5) is a FrameSoC analysis view presenting several metrics in the form of a pie chart. The view is composed of two parts. On the top left, there



<Statistics Pie Chart>

Trace: scorep-mg.A.64

Event Types

Event Types

Name	Percentage	Occurrences
🟩 MPI_Irecv	31.69 %	43296.0
🟥 MPI_Wait	31.69 %	43296.0
🟦 MPI_Send	31.69 %	43296.0
🟪 MPI_Allreduce	4.12 %	5632.0
▽ ⬛ Aggregated slices	0.80 %	1088.0
🟨 MPI_Barrier	0.28 %	384.0
🟩 MPI_Bcast	0.28 %	384.0
🟩 MPI_Init	0.05 %	64.0
🟥 MPI_Comm_size	0.05 %	64.0
🟩 MPI_Finalize	0.05 %	64.0
🟪 MPI_Comm_rank	0.05 %	64.0

Filter matched 11 of 11 items

FIGURE 5 – Statistics Pie Chart view

are the metric selector and a *play* button to load the metric. At the bottom left, there is the actual pie chart. On the right, a table viewer displays the same information as the pie. This table has three columns: the pie-slice name, the percentage value and the actual value. The name column cells contain a small square icon, filled with the color corresponding to the slice. Each column header, when clicked, triggers the sorting of the rows according to the values in the corresponding column. On the top of this table, there is an editable field, acting as a filter on table rows, working on the first column (*Name*). The number of items matched by the filter, over the totality of items, is shown in the status bar under the table viewer.

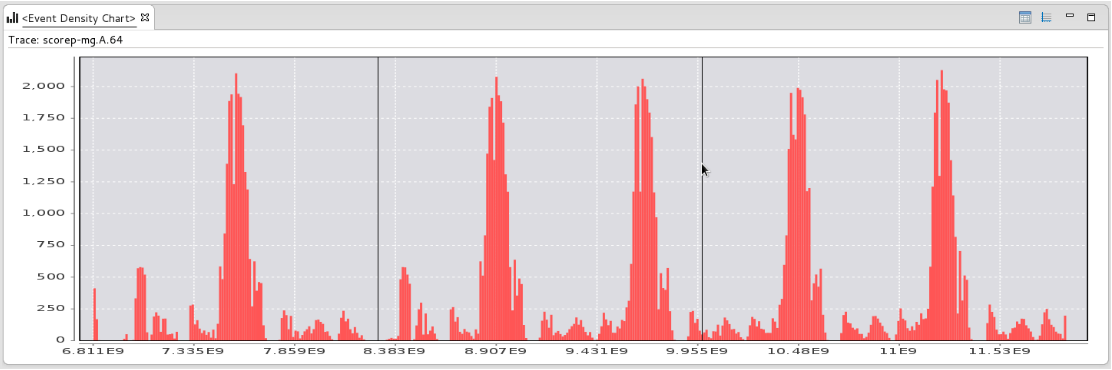
For the time being, two metrics are available: the *Event Producer* instances and the *Event Type* instances. Each slice represents the number of events having a given event producer or a given type respectively.

From Figure 5 it is possible to note that there can be a special slice in the pie: the *Aggregated slice*. This slice aggregates all the slices whose value is smaller than a given threshold, being therefore difficult or impossible to see. This threshold has been empirically set to 1%, taking into account user ergonomics and screen limitations. In the table on the right, the *Aggregated slice* corresponds to a folder entry, whose sub-entries are the actual slices. All the detailed information is thus kept and available in the tabular representation.

Note that when a *Statistics Pie Chart* view is opened for a given trace using the context menu in the trace browser, no pie chart is actually displayed, since the user has to select the metric of interest first, then press the *play* button.

4.2 Event Density Chart

The *Event Density Chart* view (Figure 6) is a FrameSoC analysis view displaying the event density over time in the form of a histogram. The x axis represents the time, while the y axis represents the number of events. The histogram number of bins is fixed and has been chosen in order to ensure a clear visualization, taking into account the number of pixels actually present on a screen. The user can zoom and dezoom portions of the chart, using the mouse as shown in Figure 6. In particular, to zoom a portion of the chart, the user has to click to the start of the portion, drag the mouse going to the right up to the end of the portion, then release the click. To completely dezoom, the user has to do the same as above, but dragging the mouse to the left this time. Hovering the mouse on a bin, some information about the bin is displayed (the central timestamp and the number of events).



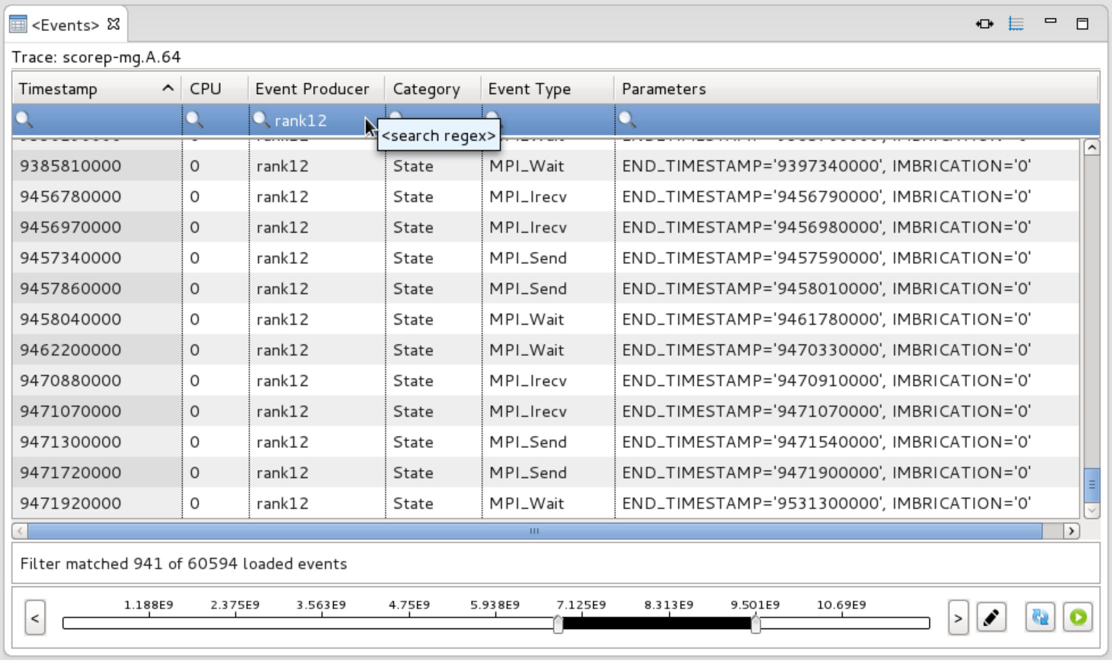
Trace: scorep-mg.A.64

FIGURE 6 – Event Density Chart view

The view toolbar contains a *table* button, which triggers the visualization of the displayed portion of trace in the table view.

Note that, for the computation of the event density, all the events of the trace are considered.

4.3 Event Table View



Trace: scorep-mg.A.64

Timestamp	CPU	Event Producer	Category	Event Type	Parameters
		rank12			
9385810000	0	rank12	State	MPI_Wait	END_TIMESTAMP='9397340000', IMBRICATION='0'
9456780000	0	rank12	State	MPI_Irecv	END_TIMESTAMP='9456790000', IMBRICATION='0'
9456970000	0	rank12	State	MPI_Irecv	END_TIMESTAMP='9456980000', IMBRICATION='0'
9457340000	0	rank12	State	MPI_Send	END_TIMESTAMP='9457590000', IMBRICATION='0'
9457860000	0	rank12	State	MPI_Send	END_TIMESTAMP='9458010000', IMBRICATION='0'
9458040000	0	rank12	State	MPI_Wait	END_TIMESTAMP='9461780000', IMBRICATION='0'
9462200000	0	rank12	State	MPI_Wait	END_TIMESTAMP='9470330000', IMBRICATION='0'
9470880000	0	rank12	State	MPI_Irecv	END_TIMESTAMP='9470910000', IMBRICATION='0'
9471070000	0	rank12	State	MPI_Irecv	END_TIMESTAMP='9471070000', IMBRICATION='0'
9471300000	0	rank12	State	MPI_Send	END_TIMESTAMP='9471540000', IMBRICATION='0'
9471720000	0	rank12	State	MPI_Send	END_TIMESTAMP='9471900000', IMBRICATION='0'
9471920000	0	rank12	State	MPI_Wait	END_TIMESTAMP='9531300000', IMBRICATION='0'

Filter matched 941 of 60594 loaded events

FIGURE 7 – Events view

The *Events* view (Figure 7) is a FrameSoC analysis view showing a tabular representation of trace events. The main element of this view is a table viewer, displaying a distinct row for each event of the trace. This table has the following columns:

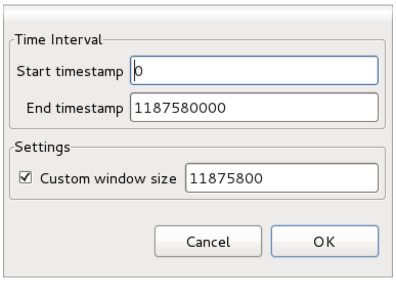
- Timestamp: the event timestamp.

- CPU: the number of the CPU on which the event has been produced.
- Event Producer: the name of the entity producing the event.
- Category: the event category (*Punctual Event*, *State*, *Link* or *Variable* [4]).
- Event Type: the event type name.
- Parameters: the list of the event custom parameters, with the format NAME='VALUE'. For the events whose category is not *Punctual Event*, this list is preceded by the two parameters that qualify a given category [4], always presented with the same format.

Each column header, when clicked, triggers the sorting of the rows according to the corresponding column. The first row of the table contains an editable filter for each column. These filters accept regular expressions. For example, in Figure 7 only the events produced by a given producer are filtered. The number of events matched by the filter, over the totality of loaded events, is shown in the status bar under the table viewer.

At the bottom of the view, there is a time management bar. This bar contains a double range time slider (representing the whole trace duration) surrounded by two arrow buttons, and three more buttons on the right (*edit*, *reset* and *play* buttons). The two knobs of the double range slider identify the portion of the trace (colored in black) actually loaded in the table. This way, the user always keeps a global visibility on the whole trace, while loading only the information he is interested in. In order to change the time window loaded in the table, one needs to change the width of the black bar. However, in order to avoid spurious and useless data transfers (from disk to memory), the change is executed only when the user activates the *play* button on the right of the bar. The *reset* button resynchronizes the time bar with the time window actually loaded in the table, if they differ. In order to modify the time window visualized in the double range slider the user has several possibilities.

- Graphical selection: it is the most intuitive way and it involves using the two knobs, in order to graphically set the two bounds of the time interval.
- Time window navigation: this possibility involves using the two arrow buttons placed respectively on the left and on the right of the time bar, in order to select a time window that has the same size of the currently visualized one, but is located immediately before or immediately after the current one.
- Manual selection: by pressing the *edit* button placed immediately beside the time bar on the right, the user access a dialog (Figure 8) where it is possible to explicitly put the exact values of the time interval start and end timestamps. This dialog enables also a manual definition of the size of the time window to be used when performing the time window navigation described above.



Time Interval

Start timestamp 0

End timestamp 1187580000

Settings

☑ Custom window size 11875800

FIGURE 8 – Time window dialog

The view toolbar contains an *adjust* button, which triggers column width resizing in order to fit the actual content.

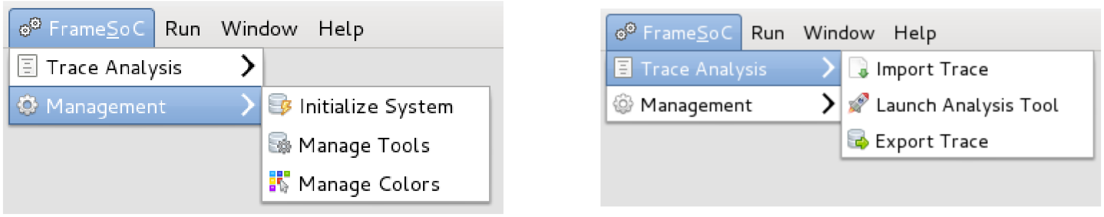
5 FrameSoC Menu and Toolbar

When the FrameSoC perspective is activated, the FrameSoC menu (Figure 9) and the corresponding toolbar (Figure 10) are visible.

In the menu, the different functionalities are grouped in two categories: management and trace analysis. The management menu (Figure 9a) allows the user to access system configuration, tool management and color management. The trace analysis menu (Figure 9b) allows the user to launch importers, analysis tools and exporter tools.

In the toolbar, the different buttons are simply shortcuts for the above functionalities, where corresponding icons mean corresponding functionalities. The added value of the toolbar is that, beside each of the three buttons used to launch the tools of the various category (import, analysis, export), there is a drop down menu containing the list of the tools of that category, useful to directly launch a tool.

In the following subsections, the functionalities accessible from the menu or the toolbar are explained in detail.



(a) Management Menu

(b) Trace Analysis Menu

FIGURE 9 – FrameSoC Menu

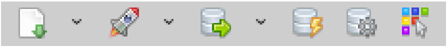


FIGURE 10 – FrameSoC Toolbar

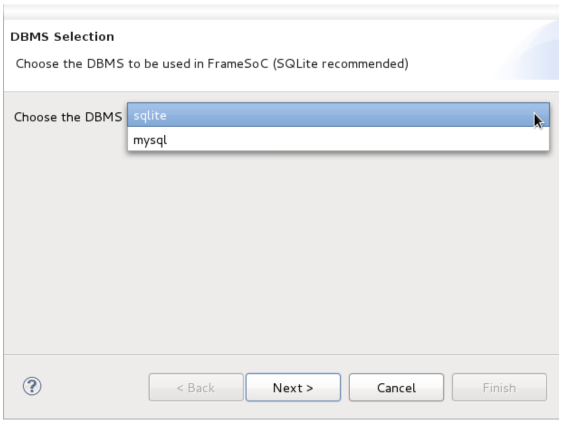
5.1 System Initialization

At system initialization, the user accesses a configuration wizard, whose first page is shown in Figure 11. This page allows the user to select the DBMS [3] to be used for trace storage. In fact, as described in the technical report RT-435 [5], FrameSoC can work with several DBMS. At the moment, the support has been implemented for SQLite (recommended) and MySQL.

Once the user has chosen the DBMS and pressed *Next*, he comes to the DBMS configuration page (Figure 12), which is different for each DBMS. If SQLite is selected (Figure 12a), the user has simply to enter the directory where he wants the database files to be kept. Otherwise, if MySQL is chosen (Figure 12b), the user has to specify some connection parameters (user-name, password, URL).

In both cases, after pressing *Finish*, the FrameSoC storage subsystem is correctly configured. If a System DB already exists, it is reused, otherwise a new one is created. This configuration is saved in the FrameSoC configuration file (Appendix A).

[3] Data Base Management System

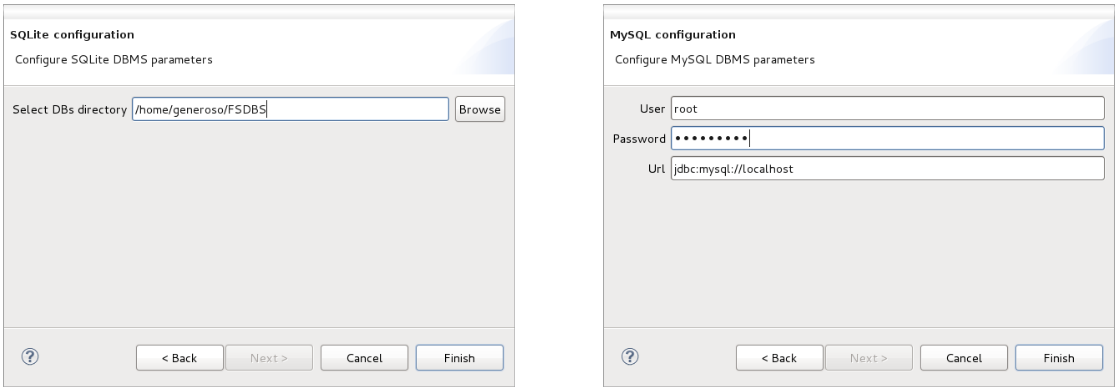


DBMS Selection

Choose the DBMS to be used in FrameSoC (SQLite recommended)

Choose the DBMS sqlite

FIGURE 11 – System Initialization: DBMS selection dialog



SQLite configuration

Configure SQLite DBMS parameters

Select DBs directory /home/generoso/FSDBS

MySQL configuration

Configure MySQL DBMS parameters

User root

Password •••••••••

Url jdbc:mysql://localhost

(a) SQLite configuration dialog

(b) MySQL configuration dialog

FIGURE 12 – System Initialization: DBMS configuration

Note that after the initialization, in the case of existing System DB, if there is a mismatch between the tools registered in this System DB and the tools actually present in the FrameSoC Eclipse runtime [4], this is automatically fixed:

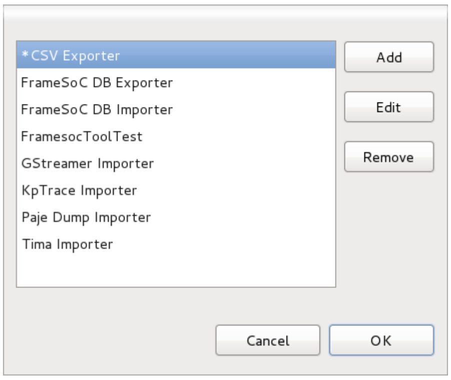
- If a tool exists in the DB but not in the runtime, the tool is removed with its results (if any), after user confirmation.
- If a tool exists in the runtime but not in the DB, the tool is automatically registered.

Note also that at each FrameSoC startup, the system automatically checks that the storage configuration is good. If it is not the case, the system initialization wizard is automatically launched. A control for mismatch between runtime tools and System DB tools is equally done at each startup, with the same policy as described above.

[4] As described in the technical report RT-435 [5] the preferred way to add tools to FrameSoC is to create an Eclipse plugin, extending a specific extension point defined by FrameSoC. For this reason a tool is typically a plugin in the FrameSoC Eclipse runtime.

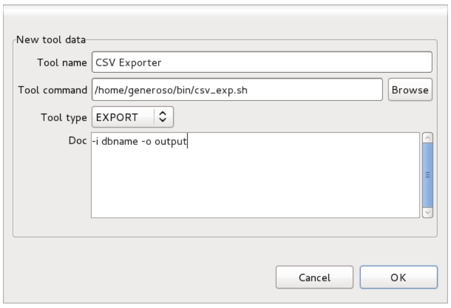
5.2 Tool Management

The tools management dialog (Figure 13) simply displays the list of tools registered to the system, enabling the installation, modification and removal of external *black-box* tools (tools that are not eclipse plugins).



*CSV Exporter
FrameSoC DB Exporter
FrameSoC DB Importer
FramesocToolTest
GStreamer Importer
KpTrace Importer
Paje Dump Importer
Tima Importer

FIGURE 13 – Tool manager



New tool data
Tool name: CSV Exporter
Tool command: /home/generoso/bin/csv_exp.sh
Tool type: EXPORT
Doc: -i dbname -o output

FIGURE 14 – Black-box tool dialog

Adding or modifying a *black-box* tool requires the user to edit the different fields of the dialog displayed in Figure 14. In particular, the user has to pick a unique tool name, specify the launching command, select the tool type (import, analysis, export), write the launching documentation.

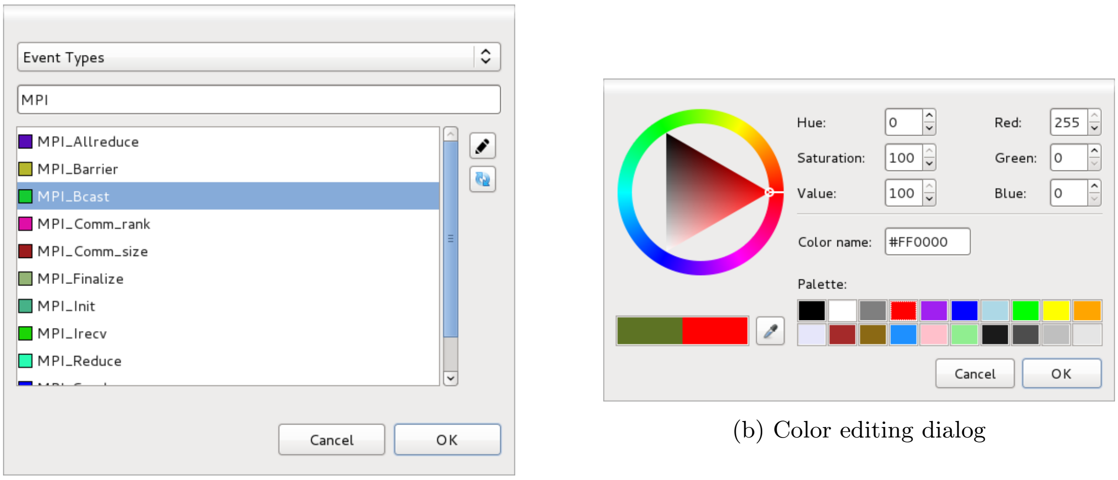
Note that the use of external *black-box* tools is discouraged, since most of the functionalities available using Eclipse plugins are not available.

Note also that plugin tools cannot be added, edited or removed via the manage tool dialog, since they are managed as standard Eclipse plugins. Plugin tools are installed and removed using the normal Eclipse procedure (Help > Install New Software... menu).

5.3 Color Management

The color management dialog (Figure 15a) allows the user to modify the colors associated to event producers and event types in a centralized way for the whole workbench. The combo box at the top of the dialog enables the selection of the entity (event producer or event type). The list below enumerates all the entities, preceded by a small squared icon filled with the entity color.

The editable text field on top of this list can be used as a filter on the list. When an entity is selected, pressing the *edit* button gives access to the color edition dialog (Figure 15b). Pressing *OK* saves all changes. Pressing the *reset* button (under the *edit* button), reverts unsaved changes. The color configuration for a given entity is physically stored in a configuration file located in the `configuration/fr.inria.soctrace.framesoc.ui` sub-folder of the eclipse install directory. For example, the relative path to eclipse install directory of the event type color configuration file is: `configuration/fr.inria.soctrace.framesoc.ui/event_type_colors`.



Event Types

MPI

- MPI_Allreduce
- MPI_Barrier
- MPI_Bcast
- MPI_Comm_rank
- MPI_Comm_size
- MPI_Finalize
- MPI_Init
- MPI_Irecv
- MPI_Reduce

Hue: 0
Saturation: 100
Value: 100
Red: 255
Green: 0
Blue: 0
Color name: #FF0000

(b) Color editing dialog

(a) Color management dialog

FIGURE 15 – FrameSoC Color Management

Note that when OK is pressed after changing some colors, the workbench modules are notified and the views may react by updating their colors in real-time (as done by the Pie Chart for example).

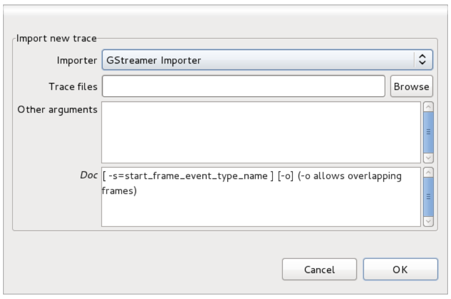
5.4 Trace Import

The trace import dialog (Figure 16) allows the user to import a new trace into the system using one of the registered importer tools. The user selects the importer from the combo box at the top, then he specifies the trace files (if more than one file is needed, all the files should be selected in the browser dialog opened when the *Browse* button is pressed). If the importer requires additional parameters (the *Doc* field normally provides this information), the user specifies these parameters too.

Finally, after pressing *OK*, the import process is launched. At the end of this process, the trace browser view is automatically updated with the new trace information.

5.5 Launch Analysis Tool

The launch analysis tool dialog (Figure 17) allows the user to launch one of the analysis tools registered to the system. The user selects the analysis tool from the combo box at the top. If the tool requires additional parameters (the *Doc* field normally provides this information), the user specifies these parameters too. Finally, after pressing *OK*, the tool is launched.

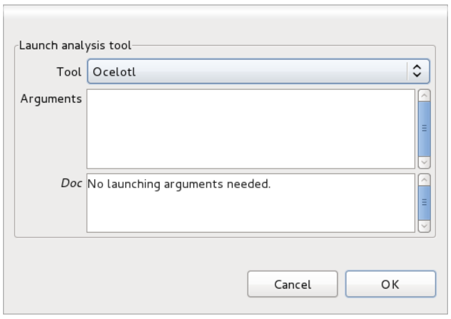


Import new trace

Importer: GStreamer Importer

Doc: [-s=start_frame_event_type_name] [-o] (-o allows overlapping frames)

FIGURE 16 – Import trace dialog



Launch analysis tool

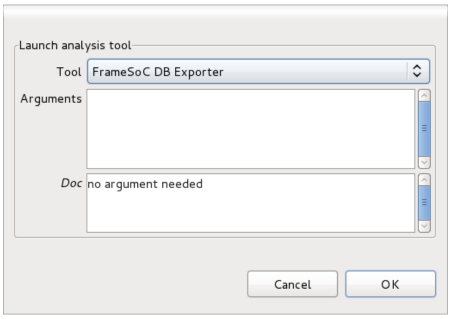
Tool: Ocelotl

Doc: No launching arguments needed.

FIGURE 17 – Launch analysis dialog

5.6 Trace Export

The trace export dialog (Figure 18) works exactly as the launch analysis tool dialog, with the difference that this time the user can launch one of the exporter tools registered to the system.



Launch analysis tool

Tool: FrameSoC DB Exporter

Doc: no argument needed

FIGURE 18 – Export trace dialog

A FrameSoC Configuration File

The FrameSoC configuration file (`.soctrace.conf`) is located in the user home directory. It is automatically generated and mostly configurable using the GUI.

The file contains the following parameters:

soctrace_dbms : DBMS used by FrameSoC. The accepted values are: sqlite, mysql.

mysql_db_user : MySQL database user name.

mysql_db_password : MySQL database user name.

mysql_base_db_jdbc_url : MySQL database connection URL.

sqlite_db_directory : Directory containing the SQLite database files.

soctrace_db_name : Name of the FrameSoC System DB.

max_view_instances : Maximum number of instances for a given type of analysis views.

References

[1] SoC-TRACE project. `http://tinyurl.com/minalogic-soc-trace`.

[2] Generoso Pagano and Vania Marangonzova-Martin. SoC-Trace Infrastructure. Rapport Technique RT-0427, INRIA, November 2012.

[3] Vania Marangozova-Martin and Generoso Pagano. Gestion de traces d'exécution pour le systèmes embarqués : contenu et stockage. Research Report RR-LIG-046, LIG, Grenoble, France, 2013.

[4] Alexis Martin, Generoso Pagano, Jérôme Correnoz, and Vania Marangozova-Martin. Analyse de systèmes embarqués par structuration de traces d'exécution. In *ComPAS'2014*, Neuchâtel, Switzerland, April 2014.

[5] Generoso Pagano and Vania Marangozova-Martin. SoC-Trace Infrastructure Benchmark. Rapport Technique RT-0435, INRIA, June 2013.

RESEARCH CENTRE
GRENOBLE – RHÔNE-ALPES

Inovallée
655 avenue de l'Europe Montbonnot
38334 Saint Ismier Cedex