

# **Qfsm User Manual**

**Stefan Duffner**

**Camille Decock**



---

# Table of Contents

1. Introduction .....

---

## List of Tables







---

# Chapter 2. Using Qfsm

## 2.1. Using the main menu

This section briefly explains the functions available through the main menu.

### 2.1.1. File

To create a new file:	select “New”. For more details, see Section 2.2,
-----------------------	--





## 2.1.5. State

To switch to the "add state" mode and create a new state:	select “New”. For more details, see
---	-------------------------------------





## 6. Simulate

Only one mode can be activated at a time. To change the mode, click on one of the icons in the middle of the toolbar.

When a diagram is larger the working area, you have the possibility to move the view of a diagram. To move the view of a diagram, click on the middle mouse button and drag the mouse pointer. As soon as you release the middle mouse button the application reverts to the selected mode.



Alternatively, you can select the respective menu entry or press the respective short cut. The active mode is indicated by a highlighted toolbar button. In some modes, the form of the mouse cursor also changes, e.g. a magnifier for the zoom mode.



2) Switch to the "add state" mode (see Section 2.3.4, "Using the Add State mode").

1) switch to the "select mode" (see Section 2.3.1, "Using the Select mode").

2) double-click on the respective state

Or

2) select one state and select *State->Edit* from the main menu.

## 2.5. Adding and modifying transitions

To create a new transition:

1) you need first to:

- create a FSM or

- open an existing filem [(.)] TJ 1 0 0 -2 -17 Tm [( State->Edi7812iS35725 2iS3572a,,56 14 1 h W n 0 G [] 06e,7eat6w ) se



Input	<p>If the condition type is binary, enter a string of zeros and ones representing the Mealy input that should trigger this transition. You can also use the character 'x' meaning: "don't care". You can also specify several alternative sets of inputs linked with " " (OR).</p> <p>If the condition type is ASCII enter a character or specify an expression in a specific format which is explained in detail in</p>
-------	--

## 2.6.1. Single character

This is the most simple form of condition. It contains one ASCII character, e.g. 'a' or 'z'.

Note that for special characters, e.g. '-' (minus sign) or the space character you need to use an escape sequence (see Section 2.6.3, “Escape sequences”).

- A-F0-9



There are two modes to send input data:

- click on the *Send*

2. VHDL
3. Verilog HDL
4. KISS

Some of the export functions open a dialog allowing you to specify additional export options. However, they should be self-contained for the users having experience with hardware description languages.

### **2.9.3. State Tables**

State tables can be exported in the formats: ASCII

The idea is that you create the action file only once at the beginning, and then edit this file manually to define the specific actions. You can then modify your FSM logic in Qfsm as many times as you want and regenerate the FSM file without losing your own action definitions.

*Example:* suppose you have created an ASCII FSM and you export it under the name

**java -jar /usr/local/bin/smc/bin/Smc.jar -java FSM1.sm**

replacing the path of Smc.jar corresponding to your smc installation. This creates the file





- start state,
- event,
- end state,
- action.

When selecting this export function from the menu a non-modal dialog box is opened that displays the resulting code. This is automatically updated as the diagram is modified. An additional reset transition can optionally be added for each state by checking the box at the bottom of the dialog. There you can also specify the name of the reset event and the name of the respective action to be triggered.

## 2.9.8. State Chart XML

A "free text" FSM can be exported in the State Chart XML format as proposed by the W3C. The specification of SCXML can be found here [<http://www.w3.org/TR/scxml/>]. Note that each transition is triggered by an event. And the text that is entered as transition input is used as the name of the event. The text that is entered as the transition output is interpreted as the name of an event that is to be sent when the transition is triggered. Thus, if the output linked to a transition is non-empty a corresponding `<send>` tag will be written.

## 2.10. Importing

Qfsm can import files written in the DOT language [<http://www.graphviz.org/content/dot-language>] using Graphviz [<http://www.graphviz.org>].

To import a DOT file (with extension .gv), select *File->Import->Graphviz...* in the main menu.

Qfsm uses Graphviz to transform the textual description of the FSM into a graphical layout of the state diagram.

When writing a DOT file make sure you respect the following syntax rules:

-

Qfsm automatically determines the type of FSM: binary, ASCII, or Free Text. In the above example, "LR\_0", "LR\_1", ... denote the states. It can be given any name by specifying the `label` attribute. If you don't do this, the node name (here: "LR\_1", ...) is used as the state name. The `moore` attribute is used to specify the Moore outputs of a state.

Transitions are written as

```
State1 -> State2;
```

or

```
State1 -> State2 [ label = "00###11" ];
```

where the `label`

Inversion descriptor	
----------------------	--

---



Hardware test code, 20

Hexadecimal, 16

HP-UX, 2

HTML

State table, 18

## I

I/O description, 21

IDE project file, 3

Import, 22

Include asynchronous output, 18

Initial state, 6, 15

Input

Transition property, 13

Installation, 2

binary, 2

from source, 2

Linux, 2

Mac OS X, 2

Windows, 2

Integrity check, 5, 15

IO view, 5

IR( Tm [(5)] TJ 1 0 0 e, )] TJ 1 0 0 -J 1 0 0 -1 0 275.32998 r 0 a57 T1 0 0 -1 0 82.0K900146 Tm [(I)] TJ /F5 10 T3 1 0

## **S**

- Save, 4
- Save as, 4
- SCXML, 22
- Select, 4
- Select all, 4
- Select mode, 13
- Send button, 17
- Shadows, 5
- Simulation, 5, 15
  - VHDL, 20
- Simulator dialog
  - Input, 16
  - Output, 16
  - State name and state code fields, 16
- Single character, 14
- SMC, 19
- Solaris,