Franco Minucci

Franky’s vm

Design document

# Revision History

|  |  |  |
| --- | --- | --- |
| **Author** | **Data** | **Description** |
| F.Minucci | 28/04/2014 | Initial Release |

Table of Contents

[Revision History 1](#_Toc386469622)

[2 Introduction 2](#_Toc386469623)

[3 System requirements 2](#_Toc386469624)

[3.1 Configurable ISA and bytecode format 2](#_Toc386469625)

[3.2 Modularity 2](#_Toc386469626)

[3.3 Parallel processing support 2](#_Toc386469627)

[3.4 Configurable memory management 2](#_Toc386469628)

[3.5 Support for different operating systems 2](#_Toc386469629)

[4 Roadmap 2](#_Toc386469630)

[5 System description 2](#_Toc386469631)

[6 ISA/bytecode description document 2](#_Toc386469632)

[7 Standard modules 2](#_Toc386469633)

[8 Program interface 3](#_Toc386469634)

[9 License 3](#_Toc386469635)

[10 References 3](#_Toc386469636)

# Introduction

The Franky’s VM (FVM) is a general purpose virtual machine with a configurable instruction set. The scope of this project is to have a virtual system capable of:

* Testing new system concepts
* Emulate real architectures
* Have a VM for existing languages that can run on different systems

The project is open source and distributed with GPL License.

# System requirements

## Configurable ISA and bytecode format

The instruction set shall be user defined starting from a default starting point it should be easy to add or remove instructions as it is needed by the application.

The bytecode format shall be user defined in order to be adapted to the different user needs or architectural choices

## Modularity

The global design must be modular and component based. It shall be easy to write and plug or unplug different modules in order to adapt the functionality to the user needs.

## Parallel processing support

The system shall allow the execution of different processes or threads running at the same time. A real time option can be foreseen for the future.

## Configurable memory management

Memory management shall be configurable and adaptable with the possibility of allow or disable a garbage collector, chose different gc algorithms, allow unmanaged access to the memory if needed.

## Support for different operating systems

The FVM is designed around the write once run everywhere concept and shall be compiled and tested on different operating systems.

# Roadmap

# System description

# ISA/bytecode description document

# Standard modules

# Program interface

# License

# References