**HERCULES:**
**High-Performance Real-time Architectures for Low-Power Embedded Systems**

| | |
|---|---|
| Project title: | High-Performance Real-time Architectures for Low-Power Embedded Systems |
| Acronym: | HERCULES |
| Project ID: | 688860 |
| Call identifier: | H2020 - ICT 04-2015 - Customised and low power computing |
| Project Coordinator: | Prof. Marko Bertogna, University of Modena and Reggio Emilia |

# D6.3: Report on Open-Source Strategy Plan

| | |
|---|---|
| Document title: | Open-Source Strategy Plan |
| Version: | 1.0 |
| Deliverable No.: | 6.3 |
| Lead task beneficiary | EVI |
| Partners Involved: | All |
| Author: | Claudio Scordino |
| Status: | Final |
| Date: | December 31st 2016 |
| Nature[1]: | R |

| **Dissemination level** |
|---|

[1] **For deliverables: R** = Report; **P** = Prototype; **D** = Demonstrator; **S** = Software/Simulator; **O** = Other
**For milestones: O** = Operational; **D** = Demonstrator; **S** = Software/Simulator; **ES** = Executive Summary; **P** = Prototype

| | | |
|---|---|---|
| **PU** | Public | X |
| **PP** | Restricted to other programme participants (including the Commission Services – CS & IAB) | |
| **RE** | Restricted to a group specified by the consortium (including the IAB) | |
| **CO** | Confidential to consortium (including CS & IAB) | |

*This Project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement: 688860*

## Document history:

| Version | Date | Author | Comments |
|---|---|---|---|
| **0.1** | 2016-12-13 | EVI | Preliminary version |
| **0.2** | 2016-12-19 | EVI | Fix typos |
| **0.3** | 2016-12-20 | EVI, PIT | Fix typos and add references |
| **1.0** | 2016-12-22 | EVI | Final version |

*This Project has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement: 688860*

## Table of contents

## GLOSSARY

| Item | Description |
|---|---|
| ADAS | Advanced driver assistance systems |
| DoW | Description of Work |
| ECU | Electronic Control Unit |
| GPL | General Public License |
| IOMMU | Input-Output Memory Management Unit |
| IPR | Intellectual Property Rights |
| NDA | Non Disclosure Agreement |
| RTOS | Real-Time Operating System |

# 1. EXECUTIVE SUMMARY

# 2. INTRODUCTION

## 2.1. Objectives

This deliverable aims at a preliminary selection of the HERCULES components that can be released as open-source software and, in case, of the licensing options that can guarantee a sustainable business model for the involved partners.

The deliverable results from the activity carried out in WP6, especially from a set of physical meetings held by the consortium during the first year of the project. It will be followed by a further document (i.e., D6.4 released at month M36) which will refine and finalize the open-source strategy plan of the consortium.

## 2.2. References

[1] Open Source Initiative, https://opensource.org/

[2] Free Software Foundation, https://www.fsf.org/

[3] A Quick Guide to GPLv3, https://www.gnu.org/licenses/quick-guide-gplv3.en.html

[4] GPL, https://www.gnu.org/licenses/old-licenses/gpl-2.0.html

[5] LGPL, https://www.gnu.org/licenses/old-licenses/lgpl-2.1.html

[6] GNU Classpath license, https://www.gnu.org/software/classpath/license.html

[7] GPL Linking exception, https://en.wikipedia.org/wiki/GPL_linking_exception

[8] BSD licenses, https://en.wikipedia.org/wiki/BSD_licenses

[9] RedHat, https://www.redhat.com

[10] CentOS project, https://www.centos.org

[11] The Linux Kernel Archives, https://www.kernel.org

[12] Das U-Boot -- the Universal Boot Loader, http://www.denx.de/wiki/U-Boot/WebHome

[13] Mandriva, https://en.wikipedia.org/wiki/Mandriva

[14] Qt, https://www.qt.io/

[15] ERIKA Enterprise, http://erika.tuxfamily.org/

[16] Qt Contribution agreement, https://www.qt.io/contributionagreement/

[17] Jailhouse, https://github.com/siemens/jailhouse

[18] Multi-licensing, https://en.wikipedia.org/wiki/Multi-licensing

[19] AUTOSAR RTE, https://www.autosar.org/about/technical-overview/ecu-software-architecture/autosar-runtime-environment/

[20] The LLVM Compiler Infrastructure, http://llvm.org/

[21] University of Illinois/NCSA Open Source License, https://en.wikipedia.org/wiki/University_of_Illinois/NCSA_Open_Source_License

[22] The SolderPad hardware license, http://solderpad.org/licenses/

[23] Parallel Ultra Low Power (PULP), http://www.pulp-platform.org/

[24] Input–output memory management unit, https://en.wikipedia.org/wiki/Input%E2%80%93output_memory_management_unit

[25] clang: a C language family frontend for LLVM, http://clang.llvm.org/

[26] OpenMP 4.0 on NVIDIA CUDA GPUs, https://parallel-computing.pro/index.php/9-cuda/43-openmp-4-0-on-nvidia-cuda-gpus

[27] MIT License, https://en.wikipedia.org/wiki/MIT_License

[28] Black Duck software, Top Open-Source licenses, https://www.blackducksoftware.com/top-open-source-licenses

[29] Apache license 2.0, https://www.apache.org/licenses/LICENSE-2.0

[30] Choose an open source license, http://choosealicense.com/licenses/

[31] Pellizzoni, R., Betti, E., Bak, S., Yao, G., Criswell, J., Caccamo, M., Kegley, R. *A predictable execution model for COTS-based embedded systems*. In 17th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), pp. 269-279, 2011.

[32] Yun, H., Yao, G., Pellizzoni, R., Caccamo, M., Sha, L. *MemGuard: Memory bandwidth reservation system for efficient performance isolation in multi-core platforms*. In 19th IEEE Real-Time and Embedded Technology and Applications Symposium (RTAS), pp. 55-64, 2013.

# 3. OPEN-SOURCE LICENSING

## 3.1. Most common licenses

Despite the large amount of licenses available nowadays [1], most open-source projects still choose to release the code under the most popular GNU or MIT licenses, to let users figure out at a first glance what is allowed and what is not. In particular, the most popular licenses are:

- **GNU General Public License (GPL)** [4]

  This is a license published by the Free Software Foundation [2] allowing one to inspect, modify, redistribute (even sell) the source code as long as the recipient maintains the same rights. This is the less industry-friendly license among the ones listed in this document. In fact, it forbids proprietary applications either to directly include existing GPL source code or to be linked against a GPL library. If they do, then license starts affecting the proprietary code, and therefore its source code must be released as well (which is what industrial companies usually do not want to).
  According to a recent survey [28], this is currently the second most used license.

- **GNU Lesser General Public License (LGPL)** [5]

  Similar to the more popular GPL, this license allows linking proprietary code to a LGPL library as long as the recipient is given the possibility of re-linking such proprietary code against a newer or modified version of the same library. Unlike GPL, in fact, the linked proprietary code does not inherit the open-source license.
  Although more permissive than the standard GPL, this license is still unacceptable in some industrial contexts (e.g., biomedical, automotive) where the recipient must be prevented from accessing or modifying the software executed by the system.

- **GNU General Public License linking exception (GPL-LE)** [7]

  Also known as ClassPath [6] (from the first project that used it), this license adds an exception to the standard GPL to allow linking of a proprietary object against a GPL-LE library. At the same time, it does not impose the restrictions of the LGPL (i.e., possibility of re-linking). This is therefore the most permissive and industry-friendly license among the GPL licenses listed in this document.

- **MIT license** [27]

  This is a family [27] of licenses by Massachusetts Institute of Technology (MIT) aiming at imposing minimal restrictions on the redistribution of the software. The license is very short and simple. With respect to the GPL licenses, this family of licenses also allows including source code into a proprietary application by just acknowledging the original author of the code. According to a recent survey [28], this is currently the most used license.

- **BSD license** [8]

  BSD is a family of licenses made by the Berkeley University. The constraints are almost identical to the ones by the MIT license. The original “4-clause” BSD license contained a controversial "advertising clause" imposing to credit the original author in all advertising material. This license has been dropped in favor of the simpler “3-clause” or “2-clause” versions.

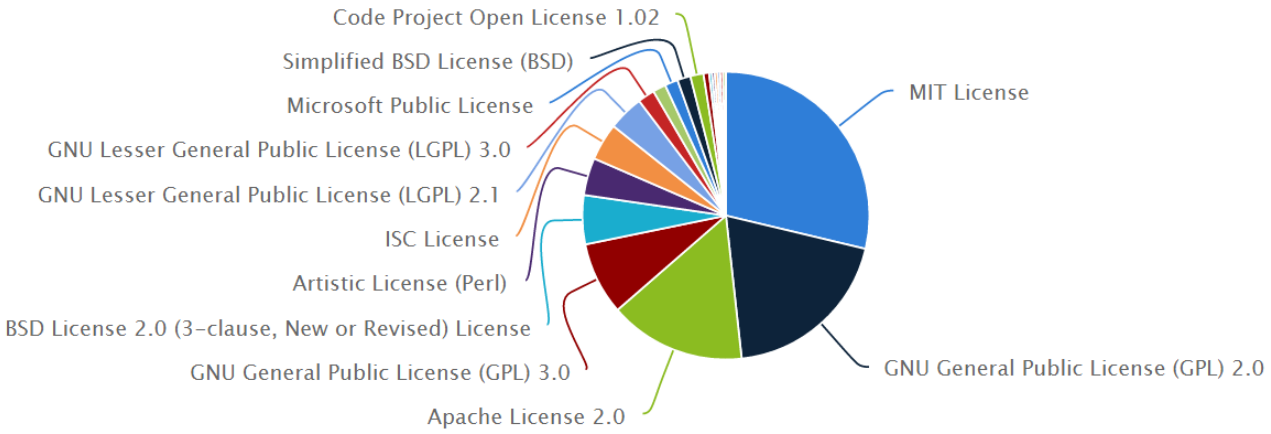
- **Apache license** [29]

  The Apache license, currently at version 2.0, is a license by the Apache Software Foundation (ASF). It is similar to MIT with a few more restrictions related to the acknowledgement of the original authors and the usage of the project name. This license is notable in the open-source community for explicitly dealing with software patents. According to a recent survey [28], this is currently the third most used license.

In addition to the distinction given by the different licenses, there are also some differences given by the specific version of each license. In particular, the version 3 [3] of the GPL licenses has introduced a set of further restrictions. Namely, these restrictions are: no patent protection (i.e., the final recipient automatically is granted any patent that might be needed to install and run a GPLv3 software); no hardware-based technologies to prevent execution of modified GPLv3 software; no Digital Rights Management (DRM), thus cracking GPLv3 code is fully legal. Several open-source projects (e.g., U-Boot [12], Linux kernel [11], etc.) have perceived this new version of the GPL licenses as too restrictive. The main concern has been related to the prohibition of hardware lockdown technologies in the context of safety-critical systems. They have therefore deliberately chosen to stick to the previous version. Indeed, a recent survey [28] shows that the amount of open-source projects under version 3 of GPL is less than half the amount of the projects under version 2.

All the above licenses also include a “no warranty” clause to secure the developer from any prosecution resulting from damages when running the software.

The following figures shows the percentage of usage of the various open-source licenses according to the Black Duck KnowledgeBase [28].




| License | Percentage of adoption |
|---|---|
| MIT | 29% |
| GPL 2.0 and 3.0 | 27 % |
| Apache 2.0 | 15 % |
| LGPL 2.1 and 3.0 | 6 % |
| BSD (all versions) | 6% |

The following figure summarizes the constraints of the various licenses:

| | Include code | Linking |
|---|---|---|
| **GPL** | X | X |
| **LGPL** | X | √ |
| **GPL-LE** | X | √ |
| **MIT** | √ | √ |
| **BSD** | √ | √ |
| **Apache** | √ | √ |

## 3.2. Misconceptions

There are two typical misconceptions about open-source software:

- *"The software must be given for free and cannot be sold".*

  None of the most popular open-source licenses forbid to sell the software. As an example, consider RedHat [9], who sells its own commercial distribution and, at the same time, makes the source code of the various components publicly available.
  Of course, the open-source license gives the final recipient the permission to re-distribute the received software even for free. Concerning the case of RedHat, this is similar to what happens with the CentOS Project [10] which distributes a free copy of the distribution re-built from the original sources. This kind of action cannot be fully prevented, but it can be discouraged by improving the commercial offer (e.g., by adding technical support, training or additional commercial tools). A further (but unusual) practice is to interrupt the provisioning of these additional services in case the recipient re-distributes the received software.

- *"The source code must be released to the public".*

  This is a misconception as well, since the above licenses only impose to release the open-source code to the recipients of the binary. It is true that, for ease of maintenance, several projects (e.g., the Linux kernel [11]) follow the common practice of releasing the source code through a public web-site. However, this practice is not stated by the licenses.

## 3.3. Business models

There are several advantages in releasing the source code under an open-source license, including:

- Cost reduction by re-using existing code from similar open-source projects
- Free help and support from a community of developers.
- Positive image of the company, making easier to hire talented people.
- Better code thanks to (free) review and improvements.

At the same time, it is of paramount importance to release the code according to a solid business model ensuring the needed revenues for the company. There are, in fact, cases of open-source companies who went bankrupt due to a wrong strategy plan [13].

To meet both goals (i.e., release under an open-source license and guaranteeing revenues) some companies choose to release the full code as open-source and rely on a business model exclusively based on additional services (e.g., customization, technical support). This is, for example, the case of the U-Boot bootloader [12]. It is also the strategy used by partner EVI with the legacy version (i.e., version 2) of its RTOS ERIKA Enterprise [15].

Other companies (e.g., Qt [14]) prefer to adopt a multi-licensing model [18]:

- The code released under the open-source license usually contains some kind of constraint. Often it has a reduced set of functionalities. In other cases, the usage is allowed only for academic or research projects. In further cases, the open-source license itself does not allow a proper usage in some industrial contexts (this is the case of Qt, released under the GPL or LGPL licenses).
- The code released under the commercial license, instead, is full-fledged. Often the commercial agreement also includes a set of additional services, like:
    - Technical support
    - Customization
    - Training
    - Bug fixing
    - Additional components

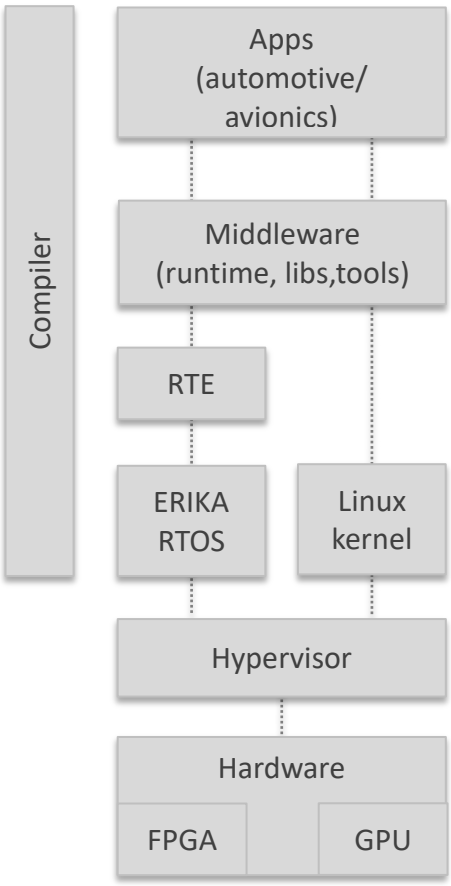
It is important to point out that to be able of releasing the code under a commercial license, the author must own the Intellectual Property Rights (IPR) on the software. In case of external contributions, a “contribution agreement” could be needed to transfer such IPR and, ultimately, allow the company to release the code under a different license. This practice is used, for example, by Qt, which asks external developers to sign the contribution agreement available at [16] before including their own code into the official library.

In the case of the HERCULES framework, it is therefore mandatory **to keep the IPR within the consortium** to leave room for a dual licensing model or re-licensing. In case of external contributions, the consortium will therefore need to set-up a process for obtaining the IPR on the modified code.

# 4. THE HERCULES FRAMEWORK

## 4.1. Architecture

The different components of the HERCULES framework are summarized in the following figure:




We will now analyze each single component to understand if it can be released under an open-source license, the kind of possible licenses, and the licensing currently favorite by the consortium.

## 4.2. Analysis of the single components

### Custom hardware

During the project, partner Magneti Marelli will develop a custom many-core board targeting the automotive market. The IPR owner of the hardware design will be the partner itself. It will be therefore possible to release such design under a “open hardware” license, as well as adopting a dual licensing business model. However, partner Magneti Marelli has already expressed its need of keeping the hardware design **proprietary** in order to create an advantage with respect to its competitors.

### FPGA blocks

Within the HERCULES project, partner ETHZ will implement general FPGA logic to wrap accelerators providing a shared memory interface with functionalities similar to iommu. The IPR of these FPGA blocks will be owned by such partner which will be therefore free to choose any licensing (including open-source and commercial). In particular, ETHZ has already expressed its willingness of adopting a **dual licensing** model: a basic version of the FPGA blocks will be released under the SolderPad license [22] (already used for the PULP platform [23]), while an enhanced version of the blocks will be provided under a proprietary commercial license.

### Hypervisor

**Jailhouse** [17] is a hypervisor targeting the embedded and safety-critical domains. It is developed by Siemens and released as open-source software under the GPLv2 license through the GitHub platform. Being an open-source initiative without the need of any "contribution agreement", the IPR ownership belongs partly to Siemens and partly to the development community.
The HERCULES consortium does not have the IPR on this software. Therefore, any change or improvement (e.g., support for the PREM model [31]) must be necessarily released under the same **GPLv2 license**, without any possibility of adopting a dual licensing strategy. The consortium can only choose if releasing the source code publicly (e.g., through a website or a GitHub account) or only to the HERCULES end-users. Currently, the consortium is oriented to release publicly only the low-level support for additional boards (e.g., NVidia, Xilinx, etc.) and release any additional functionality developed within the project only to the HERCULES users who will pay a fee.

**Vibrante** is another hypervisor, developed by NVidia for its own hardware. The source code is proprietary, and the IPR ownership belongs to NVidia. The HERCULES consortium has received a copy of the source code after signing a NDA. The consortium, in fact, aims at a commercial collaboration to improve the hypervisor by adding innovative functionalities. Since the hypervisor is property of NVidia, the functionalities developed within HERCULES cannot be released as open-source software; nor it is possible to adopt a dual licensing model. The functionalities will therefore remain **proprietary** and will be sold to NVidia through specific commercial agreements.

### ERIKA Enterprise

ERIKA Enterprise [15] is a RTOS developed by partner Evidence and released as open-source software under the GPL-LE license. The legacy version 2 of the RTOS contains some external contributions; this means that Evidence does not own the IPR on all code and cannot release the code under a dual licensing model.

During the recent P-SOCRATES FP7 project, the RTOS went through a complete re-design, aiming at improving the support for multi-core platforms. In particular, the internal data structures have been re-designed to allow a single kernel image across multiple cores. Evidence owns full IPR on this new version (also known as "version 3") which can therefore be released under **dual-licensing** model. In particular, partner Evidence has expressed its willingness of releasing parts of the code under the same GPL-LE license used for the legacy version; other parts will be released under a commercial proprietary license including technical support and additional components.

### Linux kernel

The Linux kernel is the biggest collaborative project developed by an open-source community. This piece of software is the basic brick of millions of devices running Linux or Android distributions, and it currently consists

of about 20 millions of lines of code under the GPLv2 license. Since the kernel image and the user-space applications are not linked together, the GPL license does not affect the other components of the system.
The IPR ownership belongs to the Linux kernel community, consisting of thousands of developers who have collaborated to the project over the past 25 years. The HERCULES consortium does not have therefore the permission of releasing the additional functionalities (e.g., improvement to the SCHED_DEADLINE CPU scheduler or its integration with the power management subsystem) under a different license; nor it has permission of adopting a dual licensing model. The consortium will therefore release such functionalities under the same open-source software of the rest of the kernel (i.e., **GPLv2**). Such functionalities will be released publicly (e.g., through the Linux Kernel Mailing List) to strengthen the on-going collaboration with the Linux kernel community.

### Run-Time Environment (RTE)

The AUTOSAR Run-Time Environment (RTE) [19] is an abstraction layer for communications among AUTOSAR software components. In practice, it hides the communication details under an interface which is identical for inter-ECU (e.g., CAN, LIN, FlexRay, MOST, etc.) and intra-ECU channels.
ERIKA Enterprise has not currently associated any RTE mechanism. Although not explicitly requested by the DoW, partner Evidence will start the development of this component within HERCULES in order to ease the adoption of ERIKA Enterprise in the use-case by partner Magneti Marelli.
The IPR ownership of this new component will belong to partner Evidence. Despite the freedom of choosing any open-source license or adopting a dual licensing model, partner Evidence currently plans to release this new component exclusively under a commercial **proprietary license**.

### The compiler

The LLVM compiler [20] is an open-source project started by the University of Illinois, which released the code under a proprietary open-source license (i.e., University of Illinois Open Source License [21]). This license combines text from both the MIT and BSD licenses and therefore allows re-licensing the code as long as the original author is acknowledged.
Partner ETHZ plans to enhance this compiler adding support for the PREM model [31]. The developed code will be released under **dual license**: a subset of the features will be released publicly under the same open-source license; an enhanced version of the features will be released through a commercial proprietary license.

### Middleware: OpenMP runtime for NVidia

The CLANG project [25] provides a OpenMP run-time for NVidia GPUs [26] under the same open-source license of the parent LLVM project (i.e., University of Illinois Open Source License). As explained above, this license allows re-licensing and dual licensing the modified code, as long as the copyright notice is reproduced. Partner ETH plans to enhance such support and to release the new functionalities under a **dual licensing** model.

In addition, the HERCULES partners may implement a set of new libraries and tools for the Linux user-space. These additional components may e.g. add additional functionalities for the PREM [31], the MemGuard [32] or the energy awareness infrastructures. These tools are expected to be developed from scratch. The IPR ownership will therefore belong to the specific partners who collaborated to the design and the implementation. Such partners will be free of licensing the code under any license (including open-source and commercial). At the moment, the consortium plans to release the code under a **dual licensing** model.

### The applications

Within the HERCULES project, the partners Magneti Marelli, Airbus and Pitom will develop the use-case applications for the automotive, avionics and ADAS domains, respectively. The IPR of such applications will belong to the respective partner. Despite the freedom of releasing the code as open-source or through a dual licensing model, the partners have already chosen to keep the property of the developed software in order to have an advantage with respect to their competitors.

## 5. CONCLUSIONS

The following table summarizes the possibilities and the current choices of the consortium for the single components:

| | Open-source released to public | Open-source released to end-users | Proprietary |
|---|---|---|---|
| **Custom hardware** | – | – | √ |
| **FPGA blocks** | √ | – | √ |
| **Hypervisor – Jailhouse** | √ | √ | X |
| **Hypervisor – Vibrante** | X | X | √ |
| **ERIKA Enterprise 3** | √ | – | √ |
| **RTE** | – | – | √ |
| **Linux kernel** | √ | – | X |
| **Compiler** | √ | – | √ |
| **Middleware** | √ | – | √ |
| **Apps** | – | – | √ |

where:

- the red box is a choice forbidden by the original background license;
- the yellow box is a choice allowed by the original background license but currently discarded by the consortium;
- the green box is the current choice of the consortium.

From the table it is clear that the consortium aims at releasing most of the components under a dual licensing model, mixing the benefits of an open-source strategy with the need of ensuring the revenues for the work carried out in the project.