THALES

embedded brains

CISTER
Research Centre in
Real-Time & Embedded
Computing Systems

Lero THE IRISH SOFTWARE
RESEARCH CENTRE

jenaoptronik

Trinity College Dublin
Coláiste na Tríonóide, Baile Átha Cliath
The University of Dublin

Critical
software

AIRBUS

RTEMS SMP
Ready to Fly

RTEMS Qualification Data Package Software Configuration File [sparc/gr740/smp/5]
Release 6

CONTENTS

© 2020, 2021 embedded brains GmbH   ii

Identification, Copyrights and License

© 2020, 2021 embedded brains GmbH

Action	Name	Organization	Signature
Written by	Sebastian Huber	embedded brains GmbH	
	Rute Mateus	EDISOFT	
	Sofia Pacheco	EDISOFT	
	José Valdez	EDISOFT	
	ISVV Team	Critical Software	
Verified by	Frank Kühndel	embedded brains GmbH	
Approved by	Nuno Ramos	EDISOFT	

Release: 6, Date: 2023-06-20, Status: Approved

This is a release issued as a result of the ISVV activity.

The following ISVV findings and changes were implemented:

- RTEMS-SMP-REQ-VER-041 - Units for data representing a physical quantity.

- Logos and layout were updated.

- Updated *Change List* and *Possible Problems and Known Errors*.

Release: 5, Date: 2022-05-18, Status: Replaced

This is a warranty release issued for project *Qualification of RTEMS Symmetric Multiprocessing (SMP)* ESA Contract No. 4000125572/18/NL/GLC/as. The following changes were implemented:

- In *Change List*, describe changes for package version 4.

- Add José Valdez (EDISOFT) as a writer of the document.

Release: 4, Date: 2021-12-10, Status: Replaced

This is the final release issued for project *Qualification of RTEMS Symmetric Multiprocessing (SMP)* ESA Contract No. 4000125572/18/NL/GLC/as.

Release: 3, Date: 2021-10-01, Status: Replaced

Release issued for the *QR*/2.

Release: 2, Date: 2021-05-28, Status: Replaced

Release issued for the *QR*/1.

Release: 1, Date: 2021-03-12, Status: Replaced

Release issued for the *CDR*.

CHAPTER

ONE

INTRODUCTION

The *RTEMS Qualification Data Package* (*QDP*) provides you with a comprehensive set of tools, libraries, header files, sources, and documentation to build applications using the *RTEMS* real-time operating system. Some components are pre-qualified for use in criticality category B systems according to *ECSS* standards for software products (ECSS-E-ST-40C [ECS09a] and ECSS-Q-ST-80C Rev.1 [ECS17]).

The QDP is shipped in the separate archive file `rtems-6-sparc-gr740-smp-5.tar.xz`[1]. Please read the *Installation Instructions* section before you unpack this archive. You may obtain this document and the QDP from the following ESA web site: https://rtems-qual.io.esa.int/.

The *ECSS Software Documentation Overview*, *Technical Notes Overview*, *RTEMS Documentation Overview* and *Formal Methods Documentation Overview* sections give you a structured overview and quick access to all documents of the QDP.

For help to migrate existing applications from a previous RTEMS version to the RTEMS version delivered by the QDP, see the *Migration Hints for Applications*. Migrating applications and a qualification according to ECSS standards is a complex task, especially if you migrate also from a uniprocessor system to an *SMP* system. Do not hesitate to contact a *Service Provider* for the QDP and profit from their experience.

Guidance for RTEMS Qualification in User's Environment is provided in a dedicated section, describing the necessary steps to qualify the QDP in its own hardware platform.

Important Notice

The QDP contains several open source software products which are covered by various open source licenses. You find all sources and licenses in the `/opt/rtems-6-sparc-gr740-smp-5/src` directory in case the QDP was unpacked in `/opt`. The QDP is provided by the copyright holders and contributors "as is" and any express or implied warranties, including, but not limited to, the implied warranties of merchantability and fitness for a particular purpose are disclaimed. In no event shall the copyright owner or contributors be liable for any direct, indirect, incidental, special, exemplary, or consequential damages (including, but not limited to, procurement of substitute goods or services; loss of use, data, or profits; or business interruption) however caused and on any theory of liability, whether in contract, strict liability, or tort (including negligence or otherwise) arising in any way out of the use of this software, even if advised of the possibility of such damage.

---

[1] The archive file `rtems-6-sparc-gr740-smp-5.tar.xz` has an SHA512 digest of cc74a2db2ae2341f32a4aee09dc772dcd31be82643f98092150474783cd5d108efd29df04c68c065f4b841ab52704c46d98665237975aacc332018a52d1c6160.

CHAPTER

TWO

APPLICABLE AND REFERENCE DOCUMENTS

2.1 Applicable Documents

There are no *applicable documents*.

2.2 Reference Documents

For reference documents see the *bibliography*.

CHAPTER

THREE

TERMS, DEFINITIONS AND ABBREVIATED TERMS

ABI This term is an acronym for Application Binary Interface.

API This term is an acronym for Application Programming Interface.

applicable document This term is defined by ECSS-S-ST-00-01C as a "document that contains provisions which, through reference in the source document, constitute additional provisions of the source document".

Board Support Package A collection of device initialization and control routines specific to a particular type of board or collection of boards.

BSP This term is an acronym for *Board Support Package*.

C++11 The standard ISO/IEC 14882:2011.

C11 The standard ISO/IEC 9899:2011.

CDR Critical Design Review

ECSS This term is an acronym for European Cooperation for Space Standardization.

ESA European Space Agency

FCV Functional Configuration Verification

GCC This term is an acronym for GNU Compiler Collection.

GNU This term is an acronym for GNU's Not Unix.

IRD This term is an acronym for Software Interface Requirements Document.

OpenMP This term is an acronym for Open Multi-Processing.

PA Product Assurance

PCV Physical Configuration Verification

POSIX This term is an acronym for Portable Operating System Interface.

QDP This term is an acronym for Qualification Data Package.

QR Qualification Review

RB This term is an acronym for Requirements Baseline.

RSB RTEMS Source Builder

RTEMS This term is an acronym for Real-Time Executive for Multiprocessor Systems.

SMP This term is an acronym for Symmetric Multiprocessing.

SPR Software Problem Report

SSS This term is an acronym for Software System Specification.

TLS This term is an acronym for Thread-Local Storage [Dre13]. TLS is available in *C11* and *C++11*. The support for TLS depends on the CPU port [con21a].

CHAPTER

FOUR

SOFTWARE CONFIGURATION ITEM OVERVIEW

Documents generated specifically for a QDP contain a package variant identification and version, for example [sparc/gr740/smp/5]. The package variant may have an impact on the content of documents. For example, the SRS generated for an SMP configuration will contain other requirements compared to the SRS generated for an uniprocessor configuration. Another example are the memory benchmarks and object sizes presented in this document, see *Memory Usage Benchmarks* and *Memory Usage of Objects*. In addition to the package variant identification and version, each document has a revision which is defined in the document specific changes information placed after the table of contents.

4.1 ECSS Software Documentation Overview

Before trying to access the documents, please read the *Installation Instructions* section and unpack the QDP archive.

4.1.1 Requirements Baseline

The Requirement Baseline (*RB*) which consists of the Software System Specification (*SSS*) and the Software Interface Requirements Document (*IRD*) is not contained in the QDP. The system requirements are to be defined by you, the end user of the QDP. RTEMS is designed as a reusable software product which can be utilized by application designers to ease the development of their applications. The requirements of the end system (system requirements) using RTEMS are only known to the application designer. RTEMS itself is developed by the RTEMS maintainers and they do not know the requirements of a particular end system in general. RTEMS is designed as a real-time operating system to meet typical system requirements for a wide range of applications. Its suitability for a particular application must be determined by the application designer based on the technical specification provided by the QDP accompanied with performance data for a particular target platform.

4.1.2 Technical Specification (TS)

The QDP includes the following documents of the Technical Specification (TS):

- Software Requirements Specification (SRS) [HK+23b]

    – /opt/rtems-6-sparc-gr740-smp-5/doc/ts/srs/srs.pdf

    – /opt/rtems-6-sparc-gr740-smp-5/doc/ts/srs/html/index.html

- Interface Control Document (ICD) [HK+23a]

    – /opt/rtems-6-sparc-gr740-smp-5/doc/ts/icd/icd.pdf

    – /opt/rtems-6-sparc-gr740-smp-5/doc/ts/icd/html/index.html

4.1.3 Design Definition File (DDF)

The QDP includes the following documents of the Design Definition File (DDF):

- Software Configuration File (SCF)

    The SCF is this document. It is an exception, since it is contained outside the QDP. The reason for being not included in the QDP is that the SCF may be used to verify the authenticiy and integrity of the QDP, see *Inventory of Software Configuration Item*.

- Software Detailed Design (SDD) [con23]

    – /opt/rtems-6-sparc-gr740-smp-5/doc/ddf/sdd/html/index.html

- Sofware Release Document (SRelD) [H+23]

    – /opt/rtems-6-sparc-gr740-smp-5/doc/ddf/sreld/sreld.pdf

    – /opt/rtems-6-sparc-gr740-smp-5/doc/ddf/sreld/html/index.html

- Software User Manual (SUM)

    The QDP does **not** provide a SUM which strictly follows the ECSS clauses. Please refer to the SCF (this document) and the *RTEMS Documentation Overview*.

4.1.4 Design Justification File (DJF)

The QDP includes the following documents of the Design Justification File (DJF):

- Software Reuse File (SRF)

    – /opt/rtems-6-sparc-gr740-smp-5/doc/djf/srf/SRF-024.pdf

    – /opt/rtems-6-sparc-gr740-smp-5/doc/djf/srf/html/index.html

- Software Unit and Integration Test Plan (SUITP) [HK+23c]

    – /opt/rtems-6-sparc-gr740-smp-5/doc/djf/suitp/suitp.pdf

    – /opt/rtems-6-sparc-gr740-smp-5/doc/djf/suitp/html/index.html

- Software Unit and Integration Test Report (SUITR)

  The Software Unit and Integration Test Report is a part of the Software Verification Report (SVR) [HK+23e].

- Software Validation Specification (SVS) with Respect to TS [HK+23d]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svs/svs.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svs/html/index.html

- Software Validation Report (SValR)

  The Software Validation Test Report is a part of the Software Verification Report (SVR) [HK+23e].

- Software Verification Report (SVR) [HK+23e]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svr/svr.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svr/html/index.html

4.1.5 Management File (MGT)

The QDP includes the following documents of the Management File (MGT):

- Software Development Plan (SDP) [S+21]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/sdp/SDP-000.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/sdp/html/index.html

- Software Review Plan (SRevP) [Ram21]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/srevp/SRevP-018.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/srevp/html/index.html

- Software Configuration Management Plan (SCMP) [Sil21]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/scmp/SCMP-001.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/scmp/html/index.html

4.1.6 Maintenance File (MF)

The maintenance software life cycle state is not covered by the QDP. The RTEMS development will continue independently. One goal of the project which delivered this QDP was to establish procedures in the RTEMS community so that the quality level achieved by the activity can be maintained in the future development of RTEMS.

4.1.7 Operational (OP)

The operational software life cycle state is not covered by the QDP. You, as an end user of the QDP, will hopefully have an operational phase if the software product delivered by the QDP is used in your applications.

4.1.8 Product Assurance File (PAF)

The QDP includes the following documents of the Product Assurance File (PAF):

- Software Product Assurance Milestone Report (SPAMR) [KGV+23]

    - /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spamr/spamr.pdf

    - /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spamr/html/index.html

- Software Product Assurance Plan (SPAP) [Mat21]

    - /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spap/SPAP-002.pdf

    - /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spap/html/index.html

4.2 Technical Notes Overview

Before trying to access the documents, please read the *Installation Instructions* section and unpack the QDP archive.

The QDP includes the following technical notes:

- QT-109 Technical Note: RTEMS SMP Qualification Target [HVM+21]

    - /opt/rtems-6-sparc-gr740-smp-5/doc/technical-notes/tn-qt/QT-109.pdf

- Technical Note: Space Profile [HV21]

    - /opt/rtems-6-sparc-gr740-smp-5/doc/technical-notes/tn-sp/tn-space-profile.pdf

- TI-003 Tools Identification [VP21]

    - /opt/rtems-6-sparc-gr740-smp-5/doc/technical-notes/tn-ti/TI-003.pdf

4.3 RTEMS Documentation Overview

Before trying to access the documents, please read the *Installation Instructions* section and unpack the QDP archive.

The QDP includes the following manuals of the RTEMS documentation set:

- RTEMS User Manual [con21e]

    - /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/user/user.pdf

    - /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/user/user/index.html

- RTEMS Classic API Guide [con21c]

- /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/c-user/c-user.pdf

- /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/c-user/c-user/index.html

- RTEMS Software Engineering [con21d]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/eng/eng.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/eng/eng/index.html

- RTEMS CPU Supplement [con21b]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/cpu-supplement/cpu-supplement.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/cpu-supplement/cpu-supplement/
    index.html

4.4 Formal Methods Documentation Overview

Before trying to access the documents, please read the *Installation Instructions* section and un-pack the QDP archive.

The QDP includes the following Formal Methods documentation set:

- Formal Verification Plan (Algorithms, Requirements, Approach) [BH21]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/fm/fvp/FV1-200.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/fm/fvp/index.html

- Formal Verification Artefacts (Architecture, Models, Assumptions, Traceability, Supporting Tests) [BT23]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/fm/fva/FV2-201.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/fm/fva/index.html

- Formal Verification Report [But21]

  - /opt/rtems-6-sparc-gr740-smp-5/doc/fm/fvr/FV3-202.pdf

  - /opt/rtems-6-sparc-gr740-smp-5/doc/fm/fvr/index.html

4.5 ISVV Documentation Overview

Before trying to access the documents, please read the *Installation Instructions* section and un-pack the QDP archive.

ISVV Documents Copyrights and License

© 2022 Airbus Defence and Space Limited
© 2021, 2023 Critical Software SA

The copyright holders listed above grant that the ISVV documents

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/gls/CSW-RTEMSISVV-2021-GLS-04956_4-rtems-smp-isvv-glossary-and-acronyms.pdf

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivep/CSW-RTEMSISVV-2023-PLN-01130_2-rtems-smp-isvv-ive-plan.pdf

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/iver/CSW-RTEMSISVV-2021-RPT-04960_7-rtems-smp-isvv-ive-report.pdf

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/iver/CSW-RTEMSISVV-2021-RPT-04960_7-rtems-smp-isvv-ive-report-code-annex.pdf

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/iver/CSW-RTEMSISVV-2021-RPT-04960_7-rtems-smp-isvv-ive-report-code-annex.xlsx

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivap/CSW-RTEMSISVV-2021-SVP-04961_6-rtems-smp-isvv-iva-plan.pdf

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivar/CSW-RTEMSISVV-2021-RPT-04962_4-rtems-smp-isvv-iva-report.pdf

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivar/CSW-RTEMSISVV-2021-RPT-04962_4-rtems-smp-isvv-iva-report-execution-logs-annex.zip

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivar/CSW-RTEMSISVV-2022-TCS-02711_9-rtems-smp-isvv-iva-test-specification-worksheet.pdf

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivar/CSW-RTEMSISVV-2022-TCS-02711_9-rtems-smp-isvv-iva-test-specification-worksheet.xslx

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/iver/CSW-RTEMSISVV-2023-LST-01297_3-rtems-smp-isvv-all-findings-except-for-code-analysis.pdf

- /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/iver/CSW-RTEMSISVV-2023-LST-01297_3-rtems-smp-isvv-all-findings-except-for-code-analysis.xlsx

The QDP includes the following Independent Software Verification and Validation (ISVV) documentation set:

- Glossary and Acronyms

  - /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/gls/CSW-RTEMSISVV-2021-GLS-04956_4-rtems-smp-isvv-glossary-and-acronyms.pdf

- Independent Verification Plan

  - /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivep/CSW-RTEMSISVV-2023-PLN-01130_2-rtems-smp-isvv-ive-plan.pdf

- Independent Verification Report (and Code Analysis Annex)

  - /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/iver/CSW-RTEMSISVV-2021-RPT-04960_

    7-rtems-smp-isvv-ive-report.pdf

- – /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/iver/CSW-RTEMSISVV-2021-RPT-04960_
    7-rtems-smp-isvv-ive-report-code-annex.pdf

- Independent Validation Plan

  – /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivap/CSW-RTEMSISVV-2021-SVP-04961_
    6-rtems-smp-isvv-iva-plan.pdf

- Independent Validation Report and Artefacts (Test Specification and Results, Test Proce-
  dures, Execution Logs)

  – /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivar/CSW-RTEMSISVV-2021-RPT-04962_
    4-rtems-smp-isvv-iva-report.pdf

  – /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivar/CSW-RTEMSISVV-2022-TCS-02711_
    9-rtems-smp-isvv-iva-test-specification-worksheet.pdf

  – /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivar/CSW-RTEMSISVV-2022-TSP-05453_
    1-rtems-smp-isvv-iva-test-procedures.zip

  – /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/ivar/CSW-RTEMSISVV-2021-RPT-04962_
    4-rtems-smp-isvv-iva-report-execution-logs-annex.zip

- List of All Findings (except for code analysis)

  – /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/iver/CSW-RTEMSISVV-2023-LST-01297_
    3-rtems-smp-isvv-all-findings-except-for-code-analysis.pdf

4.6 RTEMS Repositories

The QDP contains the following RTEMS Git repositories. Optionally, you can fetch the current
RTEMS Project repository state with the `git fetch origin` command. This gives you access to
the complete source code history of the RTEMS Project. Since the source code is delivered in
Git repositories you know exactly which changes were done for the QDP on top of the software
baseline provided by the RTEMS Project. You can directly use the QDP branches to do project-
specific customizations.

4.6.1 Git Repository: src/rsb

This repository contains the RTEMS Source Builder. It was used to provide the RTEMS tool
suite shipped with the QDP. The QDP contains a binary distribution of the RTEMS tools suite.
The `rtems/patches` and `rtems/sources` directories in the RSB directory contain the patches and
sources of the tool suite used to build the binary distribution. This helps you to meet the GPL
requirements of some tools. You do not have to run the RSB on your own. For example, you
may run the RSB to adopt the QDP to a different host computer platform.

The qdp branch with commit `889cf95db0122bd1a6b21598569620c40ff2069d` was used to build
the QDP. This branch is checked out after unpacking the archive. It is based on commit
889cf95db0122bd1a6b21598569620c40ff2069d of the `master` branch of the `origin` remote
repository.

4.6.2 Git Repository: src/rtems

This repository contains the RTEMS sources. It is used to provide the BSPs shipped with the QDP.

The qdp branch with commit 89499c4002fb5f8b0e1cfd7c1edac2e78076c835 was used to build the QDP. This branch is checked out after unpacking the archive. It is based on commit 89c498ccdae67ae817f8a141183b771948a817b9 of the master branch of the origin remote repository.

4.6.3 Git Repository: src/rtems-docs

This repository contains the RTEMS Documentation sources. It is used to provide the RTEMS Documentation shipped with the QDP.

The qdp branch with commit 9082f4d92c5da5f02b83a1e1203192094ea111fa was used to build the QDP. This branch is checked out after unpacking the archive. It is based on commit c16c8c6fd462d2c12301acaa8bfec7e98f05de6b of the master branch of the origin remote repository.

4.6.4 Git Repository: src/rtems-tools

This repository contains the RTEMS Tools sources. It is provided for reference.

The qdp branch with commit 2251c5a878c24e9120d9dd53751f69790e3d7387 was used to build the QDP. This branch is checked out after unpacking the archive. It is based on commit 2251c5a878c24e9120d9dd53751f69790e3d7387 of the master branch of the origin remote repository.

4.7 Guidelines for the Application Build

4.7.1 Recommended Optimization Flags

It is recommend to use the following optimization flags to compile C code of applications and libraries:

```
-O2 -ffunction-sections -fdata-sections -g
```

These flags are used to compile the production BSPs provided by the QDP. To link executables, it is recommended to enable the linker garbage collection with `-Wl,--gc-sections`. The test programs of the BSP were linked with this option. The rationale for using `-O2` is that this optimization level is commonly used.

Using function and data sections has an impact on the code generation since locations in translation units are unknown until link time. However, together with the linker garbage collection, smaller statically-linked executables can be produced. This helps to include only code in the executable that is actually used by the application. It avoids shipping executables with dead (or deactivated) code, which just wastes the typically expensive memory in the space domain. In

addition, it may reduce the bandwidth requirements when flight software needs to be patched or uploaded in flight.

The application may use other optimization flags if necessary. However, the *ABI* shall be compatible to the BSP. See also *Build an Example Application*.

4.7.2 Mandatory Compiler/Linker Flags

It is absolutely mandatory to adhere to the *ABI* defined by the BSP. It is recommended to use exactly the ABI relevant flags (machine flags) defined by the BSP for the application. You can query the ABI flags, `CFLAGS`, and linker flags of the desired BSP with the `pkg-config` command line tool, for example:

```
$ pkg-config --variable=ABI_FLAGS /opt/rtems-6-sparc-gr740-smp-5/lib/pkgconfig/
↪sparc-rtems6-gr740-qual-only.pc
-mcpu=leon3

$ pkg-config --cflags /opt/rtems-6-sparc-gr740-smp-5/lib/pkgconfig/sparc-rtems6-
↪gr740-qual-only.pc
-mcpu=leon3 -I/opt/rtems-6-sparc-gr740-smp-5/sparc-rtems6/gr740-qual-only/lib/
↪include

$ pkg-config --libs /opt/rtems-6-sparc-gr740-smp-5/lib/pkgconfig/sparc-rtems6-
↪gr740-qual-only.pc
-B/opt/rtems-6-sparc-gr740-smp-5/sparc-rtems6/gr740-qual-only/lib -qrtems -Wl,--
↪gc-sections -nodefaultlibs -Wl,--start-group -lrtemscpu -lrtemsbsp -lgcc -Wl,--
↪end-group
```

Using `pkg-config` to query the tool flags of the BSP avoids having hard coded flags in the application build system. It is recommended to place the linker flags provided by `pkg-config` to the end of the linker command line used to link the application executable. See also *Build an Example Application*.

4.8 Memory Usage Benchmarks

The memory usage benchmarks help you to get a rough estimate of the static memory usage of selected RTEMS features.

In the tables below, the `.text` section covers the benchmark program code. The `.rodata` section covers the read-only data used by the benchmark program. The `.rtemsstack` section covers statically allocated stacks. For example, stacks for the idle threads and interrupt processing. The `.data` section covers the statically allocated and initialized read-write data used by the benchmark program. The `.bss` section covers the statically allocated and zero initialized read-write data used by the benchmark program. The first data row of the table refers to a basic memory benchmark program which is used as a reference for the following memory benchmark programs. This gives you a hint how the statically allocated memory usage changes if certain features are added on top of the basic memory benchmark program.

4.8.1 Benchmarks Based on: spec:/rtems/val/mem-basic

The following memory benchmarks are based on the memory benchmark defined by
spec:/rtems/val/mem-basic.

spec	.text	.rodata	.rtemsstack	.data	.bss
/rtems/val/mem-basic	35536	96	8192	560	9904
/bsp/val/mem-clock	+2848	+0	+0	+48	+64
/rtems/barrier/val/mem-wait-rel	+1424	+0	+0	+64	+64
/rtems/barrier/val/mem-wait-rel-del	+1584	+0	+0	+64	+64
/rtems/clock/val/mem-get-uptime	+336	+0	+0	+16	+0
/rtems/clock/val/mem-set	+3024	+0	+0	+64	+16
/rtems/clock/val/mem-set-get-tod	+5280	+0	+0	+64	+16
/rtems/event/val/mem-snd-rcv	+1280	+0	+0	+16	+0
/rtems/fatal/val/mem-fatal	+0	+0	+0	+0	+0
/rtems/message/val/mem-bcst-rcv	+2896	+0	+0	+64	+0
/rtems/message/val/mem-snd-rcv	+3040	+0	+0	+64	+0
/rtems/message/val/mem-snd-rcv-del	+3216	+0	+0	+64	+0
/rtems/message/val/mem-ugt-rcv	+3008	+0	+0	+64	+0
/rtems/part/val/mem-get-ret	+752	+0	+0	+64	+64
/rtems/part/val/mem-get-ret-del	+944	+0	+0	+64	+64
/rtems/ratemon/val/mem-period	+2016	+0	+0	+48	+192
/rtems/ratemon/val/mem-period-del	+2272	+0	+0	+48	+192
/rtems/val/mem-smp-1	+17808	+0	+0	+0	-3008
/rtems/val/mem-smp-global-2	+17824	+0	+8192	+0	-1536
/rtems/val/mem-smp-global-4	+17840	+0	+24576	+0	+1280
/rtems/val/mem-smp-part-2	+17936	+0	+8192	+0	-784
/rtems/val/mem-smp-part-4	+18176	+0	+24576	+0	+4992
/rtems/scheduler/val/mem-add-cpu	+18304	+0	+0	+0	-3008
/rtems/scheduler/val/mem-rm-cpu	+19072	+0	+0	+0	-3008
/rtems/sem/val/mem-obt-rel	+7296	+0	+0	+48	+128
/rtems/sem/val/mem-obt-rel-del	+7648	+0	+0	+48	+128
/rtems/signal/val/mem-catch-snd	+1200	+0	+0	+16	+0
/rtems/task/val/mem-delete	+1488	+0	+0	+16	+0
/rtems/task/val/mem-exit	+112	+0	+0	+16	+0
/rtems/task/val/mem-get-affinity	+400	+0	+0	+0	+0
/rtems/task/val/mem-get-priority	+672	+0	+0	+0	+0
/rtems/task/val/mem-get-scheduler	+176	+0	+0	+0	+0
/rtems/task/val/mem-mode	+688	+0	+0	+16	+0
/rtems/task/val/mem-restart	+1200	+0	+0	+16	+0
/rtems/task/val/mem-set-affinity	+512	+0	+0	+16	+0
/rtems/task/val/mem-set-priority	+992	+0	+0	+16	+0
/rtems/task/val/mem-set-scheduler	+1712	+0	+0	+16	+0
/rtems/task/val/mem-sus-res	+224	+0	+0	+16	+0
/rtems/task/val/mem-wake-after	+448	+0	+0	+16	+0
/rtems/task/val/mem-wake-when	+1072	+0	+0	+16	+0
/rtems/timer/val/mem-after	+864	+0	+0	+48	+0

Table 1 – continued from previous page

spec	.text	.rodata	.rtemsstack	.data	.bss
/rtems/timer/val/mem-cancel	+480	+0	+0	+48	+0
/rtems/timer/val/mem-delete	+512	+0	+0	+48	+0
/rtems/timer/val/mem-reset	+736	+0	+0	+48	+0
/rtems/timer/val/mem-srv-after	+1376	+0	+0	+48	+0
/rtems/timer/val/mem-srv-init	+1520	+0	+0	+64	+32
/rtems/timer/val/mem-srv-when	+2032	+0	+0	+64	+0
/rtems/timer/val/mem-when	+1504	+0	+0	+64	+0
/rtems/userext/val/mem-create	+208	+0	+0	+64	+144
/rtems/userext/val/mem-delete	+752	+0	+0	+64	+144

4.8.2 Benchmarks Based on: spec:/rtems/val/mem-smp-1

The following memory benchmarks are based on the memory benchmark defined by *spec:/rtems/val/mem-smp-1*.

spec	.text	.rodata	.rtemsstack	.data	.bss
/rtems/val/mem-smp-1	53344	96	8192	560	6896
/rtems/val/mem-smp-global-2	+16	+0	+8192	+0	+1472
/rtems/val/mem-smp-global-4	+32	+0	+24576	+0	+4288
/rtems/val/mem-smp-part-2	+128	+0	+8192	+0	+2224
/rtems/val/mem-smp-part-4	+368	+0	+24576	+0	+8000
/rtems/scheduler/val/mem-add-cpu	+496	+0	+0	+0	+0
/rtems/scheduler/val/mem-rm-cpu	+1264	+0	+0	+0	+0

4.8.3 Benchmark: spec:/rtems/val/mem-basic

This static memory usage benchmark program facilitates a basic application configuration.

This resource benchmark is configured for exactly one processor, no clock driver, no Newlib reentrancy support, and no file system.

4.8.4 Benchmark: spec:/bsp/val/mem-clock

This static memory usage benchmark program facilitates a basic application configuration with the clock driver enabled (`CONFIGURE_APPLICATION_NEEDS_CLOCK_DRIVER`).

4.8.5 Benchmark: spec:/rtems/barrier/val/mem-wait-rel

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_BARRIERS` defined to one and calls to `rtems_barrier_create()`, `rtems_barrier_wait()`, and `rtems_barrier_release()`.

4.8.6 Benchmark: spec:/rtems/barrier/val/mem-wait-rel-del

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_BARRIERS` defined to one and calls to `rtems_barrier_create()`, `rtems_barrier_wait()`, `rtems_barrier_release()`, and `rtems_barrier_delete()`.

4.8.7 Benchmark: spec:/rtems/clock/val/mem-get-uptime

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_clock_get_uptime()`.

4.8.8 Benchmark: spec:/rtems/clock/val/mem-set

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_clock_set()`.

4.8.9 Benchmark: spec:/rtems/clock/val/mem-set-get-tod

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_clock_set()` and `rtems_clock_get_tod()`.

4.8.10 Benchmark: spec:/rtems/event/val/mem-snd-rcv

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_event_send()` and `rtems_event_receive()`.

4.8.11 Benchmark: spec:/rtems/fatal/val/mem-fatal

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_fatal()`.

4.8.12 Benchmark: spec:/rtems/message/val/mem-bcst-rcv

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_message_queue_construct()`, `rtems_message_queue_broadcast()`, and `rtems_message_queue_receive()`.

4.8.13 Benchmark: spec:/rtems/message/val/mem-snd-rcv

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_message_queue_construct()`, `rtems_message_queue_send()`, and `rtems_message_queue_receive()`.

4.8.14 Benchmark: spec:/rtems/message/val/mem-snd-rcv-del

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_message_queue_construct()`, `rtems_message_queue_send()`, `rtems_message_queue_receive()`, and `rtems_message_queue_delete()`.

4.8.15 Benchmark: spec:/rtems/message/val/mem-ugt-rcv

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_message_queue_construct()`, `rtems_message_queue_urgent()`, and `rtems_message_queue_receive()`.

4.8.16 Benchmark: spec:/rtems/part/val/mem-get-ret

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_PARTITIONS` defined to one and calls to `rtems_partition_create()`, `rtems_partition_get_buffer()`, and `rtems_partition_return_buffer()`.

4.8.17 Benchmark: spec:/rtems/part/val/mem-get-ret-del

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_PARTITIONS` defined to one and calls to `rtems_partition_create()`, `rtems_partition_get_buffer()`, `rtems_partition_return_buffer()`, and `rtems_partition_delete()`.

4.8.18 Benchmark: spec:/rtems/ratemon/val/mem-period

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_PERIODS` defined to one and calls to `rtems_rate_monotonic_create()` and `rtems_rate_monotonic_period()`.

4.8.19 Benchmark: spec:/rtems/ratemon/val/mem-period-del

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_PERIODS` defined to one and calls to `rtems_rate_monotonic_create()`, `rtems_rate_monotonic_period()`, and `rtems_rate_monotonic_delete()`.

4.8.20 Benchmark: spec:/rtems/val/mem-smp-1

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_PROCESSORS` defined to one using the SMP EDF scheduler (`CONFIGURE_SCHEDULER_EDF_SMP`).

4.8.21 Benchmark: spec:/rtems/val/mem-smp-global-2

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_PROCESSORS` defined to two using the global SMP EDF scheduler (`CONFIGURE_SCHEDULER_EDF_SMP`).

4.8.22 Benchmark: spec:/rtems/val/mem-smp-global-4

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_PROCESSORS` defined to four using the global SMP EDF scheduler (`CONFIGURE_SCHEDULER_EDF_SMP`).

4.8.23 Benchmark: spec:/rtems/val/mem-smp-part-2

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_PROCESSORS` defined to two using one SMP EDF scheduler for each configured processor (`CONFIGURE_SCHEDULER_EDF_SMP`).

4.8.24 Benchmark: spec:/rtems/val/mem-smp-part-4

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_PROCESSORS` defined to four using one SMP EDF scheduler for each configured processor (`CONFIGURE_SCHEDULER_EDF_SMP`).

4.8.25 Benchmark: spec:/rtems/scheduler/val/mem-add-cpu

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_scheduler_add_processor()`.

4.8.26 Benchmark: spec:/rtems/scheduler/val/mem-rm-cpu

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_scheduler_remove_processor()`.

4.8.27 Benchmark: spec:/rtems/sem/val/mem-obt-rel

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_SEMAPHORES` defined to one and calls to `rtems_semaphore_create()`, `rtems_semaphore_obtain()`, and `rtems_semaphore_release()`.

4.8.28 Benchmark: spec:/rtems/sem/val/mem-obt-rel-del

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_SEMAPHORES` defined to one and calls to `rtems_semaphore_create()`, `rtems_semaphore_obtain()`, `rtems_semaphore_release()`, and `rtems_semaphore_delete()`.

4.8.29 Benchmark: spec:/rtems/signal/val/mem-catch-snd

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_signal_catch()` and `rtems_signal_send()`.

4.8.30 Benchmark: spec:/rtems/task/val/mem-delete

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_delete()`.

4.8.31 Benchmark: spec:/rtems/task/val/mem-exit

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_exit()`.

4.8.32 Benchmark: spec:/rtems/task/val/mem-get-affinity

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_get_affinity()`.

4.8.33 Benchmark: spec:/rtems/task/val/mem-get-priority

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_get_priority()`.

4.8.34 Benchmark: spec:/rtems/task/val/mem-get-scheduler

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_get_scheduler()`.

4.8.35 Benchmark: spec:/rtems/task/val/mem-mode

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_mode()`.

4.8.36 Benchmark: spec:/rtems/task/val/mem-restart

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_restart()`.

4.8.37 Benchmark: spec:/rtems/task/val/mem-set-affinity

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_set_affinity()`.

4.8.38 Benchmark: spec:/rtems/task/val/mem-set-priority

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_set_priority()`.

4.8.39 Benchmark: spec:/rtems/task/val/mem-set-scheduler

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_set_scheduler()`.

4.8.40 Benchmark: spec:/rtems/task/val/mem-sus-res

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_task_suspend()` and `rtems_task_resume()`.

4.8.41 Benchmark: spec:/rtems/task/val/mem-wake-after

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_wake_after()`.

4.8.42 Benchmark: spec:/rtems/task/val/mem-wake-when

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_task_wake_when()`.

4.8.43 Benchmark: spec:/rtems/timer/val/mem-after

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_timer_create()` and `rtems_timer_fire_after()`.

4.8.44 Benchmark: spec:/rtems/timer/val/mem-cancel

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_timer_create()` and `rtems_timer_cancel()`.

4.8.45 Benchmark: spec:/rtems/timer/val/mem-delete

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_timer_create()` and `rtems_timer_delete()`.

4.8.46 Benchmark: spec:/rtems/timer/val/mem-reset

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_timer_create()` and `rtems_timer_reset()`.

4.8.47 Benchmark: spec:/rtems/timer/val/mem-srv-after

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_timer_create()` and `rtems_timer_server_fire_after()`.

4.8.48 Benchmark: spec:/rtems/timer/val/mem-srv-init

This static memory usage benchmark program facilitates a basic application configuration with a call to `rtems_timer_initiate_server()`.

4.8.49 Benchmark: spec:/rtems/timer/val/mem-srv-when

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_timer_create()` and `rtems_timer_server_fire_when()`.

4.8.50 Benchmark: spec:/rtems/timer/val/mem-when

This static memory usage benchmark program facilitates a basic application configuration with calls to `rtems_timer_create()` and `rtems_timer_fire_when()`.

4.8.51 Benchmark: spec:/rtems/userext/val/mem-create

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_USER_EXTENSIONS` defined to one and a call to `rtems_extension_create()`.

4.8.52 Benchmark: spec:/rtems/userext/val/mem-delete

This static memory usage benchmark program facilitates a basic application configuration with `CONFIGURE_MAXIMUM_USER_EXTENSIONS` defined to one and calls to `rtems_extension_create()` and `rtems_extension_delete()`.

4.9 Memory Usage of Objects

This section gives an overview of the application configuration dependent memory requirements of RTEMS provided objects.

4.9.1 Barrier Manager

If the Barrier Manager is used by the application, then 48 bytes are required to maintain the objects of this manager. The size of an object of this manager is independent of the application configuration. Each object of this manager has a size of 52 bytes.

4.9.2 Message Manager

If the Message Manager is used by the application, then 48 bytes are required to maintain the objects of this manager. The size of an object of this manager is independent of the application configuration. Each object of this manager has a size of 88 bytes. The memory area used for the messages managed by a message queue object is user-provided.

4.9.3 Partition Manager

If the Partition Manager is used by the application, then 48 bytes are required to maintain the objects of this manager. The size of an object of this manager is independent of the application configuration. Each object of this manager has a size of 60 bytes. The memory area managed by a partition object is user-provided.

4.9.4 Rate Monotonic Manager

If the Rate Monotonic Manager is used by the application, then 48 bytes are required to maintain the objects of this manager. The size of an object of this manager is independent of the application configuration. Each object of this manager has a size of 188 bytes.

4.9.5 Schedulers

The memory used for global scheduler data structures depends on the application configuration. Each scheduler has different memory requirements.

Each configured scheduler needs 112 bytes of read-only data.

The Deterministic Priority Scheduler needs 48 bytes. In addition, it needs 12 bytes for each configured priority level.

The Earliest Deadline First SMP Scheduler needs 76 bytes. In addition, it needs 20 bytes for each configured processor.

4.9.6 Semaphore Manager

If the Semaphore Manager is used by the application, then 48 bytes are required to maintain the objects of this manager. The size of an object of this manager depends on the application configuration. Let S be the count of configured schedulers. Each object of this manager has a size of $\max\{76, 68 + 8S\}$ bytes.

4.9.7 Task Manager

There are 60 bytes required to maintain the objects of this manager. The size of an object of this manager depends on the application configuration. Each object of this manager has at least a size of 656 bytes.

Additional 64 bytes are required for each configured scheduler.

If at least one user extension is configured, then additional 4 bytes are required for each configured user extension plus one.

Let N be the maximum of

- 144, if the Deterministic Priority Scheduler is configured, and

- 152, if the Earliest Deadline First SMP Scheduler is configured.

Additional N bytes are required for each configured scheduler.

If the Newlib reentrancy support is enabled, then additional 344 bytes are required.

Due to alignment requirements of the individual components additional memory is required for structure internal padding. The memory area used for the task storage used by a task object is user-provided.

4.9.8 Timer Manager

If the Timer Manager is used by the application, then 48 bytes are required to maintain the objects of this manager. The size of an object of this manager is independent of the application configuration. Each object of this manager has a size of 76 bytes.

4.9.9 User Extensions Manager

If the User Extensions Manager is used by the application, then 48 bytes are required to maintain the objects of this manager. The size of an object of this manager is independent of the application configuration. Each object of this manager has a size of 76 bytes.

4.10 Migration Hints for Applications

This section covers some topics which need to be considered when migrating an application from a previous RTEMS versions to the RTEMS version delivered by the QDP. The focus is on a migration from RTEMS 4.8 and the EDISOFT RTEMS Improvement. For migration help from other versions, see also the RTEMS User Manual.

4.10.1 No -specs bsp_specs GCC Option

The `-spec bsp_specs` GCC Option is no longer needed to build RTEMS applications and there is no `bsp_specs` file installed. If you use this option, then you get an error like this:

```
sparc-rtems6-gcc: fatal error: cannot read spec file 'bsp_specs': No such file or␣
↪directory
```

You can remove this GCC option from your build to fix this error. Alternatively, you can add an empty `bsp_specs` file.

4.10.2 No Manager Stubs

Older RTEMS versions provided stub files for some RTEMS Managers. These stub files are no longer needed and they are not installed. Remove the support for them in your build system.

4.10.3 rtems_task_create() vs. rtems_task_construct()

The rtems_task_create() directive is pre-qualified, however, it is only available if a custom stack allocator is configured. It is recommended to use rtems_task_construct() instead, for example:

```c
#define MAX_TLS_SIZE RTEMS_ALIGN_UP( 64, RTEMS_TASK_STORAGE_ALIGNMENT )

#define TASK_ATTRIBUTES RTEMS_DEFAULT_ATTRIBUTES

RTEMS_ALIGNED( RTEMS_TASK_STORAGE_ALIGNMENT )
static char task_storage[
  RTEMS_TASK_STORAGE_SIZE(
    MAX_TLS_SIZE + RTEMS_MINIMUM_STACK_SIZE,
    TASK_ATTRIBUTES
  )
];

static const rtems_task_config task_config = {
  .name = rtems_build_name( 'T', 'A', 'S', 'K' ),
  .initial_priority = 123,
  .storage_area = task_storage,
  .storage_size = sizeof( task_storage ),
  .maximum_thread_local_storage_size = MAX_TLS_SIZE,
  .initial_modes = RTEMS_DEFAULT_MODES,
  .attributes = TASK_ATTRIBUTES
};

void construct( void )
{
  rtems_status_code sc;
  rtems_id          id;
```

(continues on next page)

(continued from previous page)

```
  sc = rtems_task_construct( &task_config, &id );
  if ( sc != RTEMS_SUCCESSFUL ) {
    oops();
  }
}
```

4.10.4 Task Storage Area for Initialization Task

Since the rtems_task_create() directive is not supported without a custom stack allocator in the pre-qualified feature set of RTEMS, the task storage area for the initialization task shall be provided by the application. The application shall define the application configuration option CONFIGURE_INIT_TASK_CONSTRUCT_STORAGE_SIZE and the related options:

```
#define MAX_TLS_SIZE RTEMS_ALIGN_UP( 64, RTEMS_TASK_STORAGE_ALIGNMENT )

#define TASK_ATTRIBUTES RTEMS_DEFAULT_ATTRIBUTES

#define TASK_STORAGE_SIZE \
  RTEMS_TASK_STORAGE_SIZE( \
    MAX_TLS_SIZE + RTEMS_MINIMUM_STACK_SIZE, \
    TASK_ATTRIBUTES \
  )

#define CONFIGURE_MAXIMUM_THREAD_LOCAL_STORAGE_SIZE MAX_TLS_SIZE

#define CONFIGURE_RTEMS_INIT_TASKS_TABLE

#define CONFIGURE_INIT_TASK_ATTRIBUTES TASK_ATTRIBUTES

#define CONFIGURE_INIT_TASK_CONSTRUCT_STORAGE_SIZE TASK_STORAGE_SIZE
```

The maximum thread-local storage size of the application and the initialization task must be consistent. In the example above, this is ensured by using MAX_TLS_SIZE for all definitions. For your application, replace the example value of 64 with the maximum thread-local storage size required by your application. The task attributes of the initialization task shall be consistent with the task attributes used to define the initialization task storage area. In the example above, this is ensured by using TASK_ATTRIBUTES for all definitions. Your application may use different attributes.

For an example which uses these application configuration options, see *Build an Example Application*.

© 2020, 2021 embedded brains GmbH   30

4.10.5 rtems_message_queue_create() vs. rtems_message_queue_construct()

The rtems_message_queue_create() directive is not pre-qualified. As a pre-qualified alternative use rtems_message_queue_construct() instead, for example:

```c
#define MSG_SIZE sizeof( int )

static RTEMS_MESSAGE_QUEUE_BUFFER( MSG_SIZE ) msg_buffers[ 13 ];

static const rtems_message_queue_config msg_queue_config = {
  .name = rtems_build_name( 'M', 'S', 'G', 'Q' ),
  .maximum_pending_messages = RTEMS_ARRAY_SIZE( msg_buffers ),
  .maximum_message_size = MSG_SIZE,
  .storage_area = msg_buffers,
  .storage_size = sizeof( msg_buffers ),
  .attributes = RTEMS_DEFAULT_ATTRIBUTES
};

void construct( void )
{
  rtems_status_code sc;
  rtems_id          id;

  sc = rtems_message_queue_construct( &msg_queue_config, &id );
  if ( sc != RTEMS_SUCCESSFUL ) {
    oops();
  }
}
```

4.10.6 No free()

You can allocate memory using the rtems_malloc(), rtems_calloc(), and posix_memalign() functions. However, the free() function is not pre-qualified. There is also no malloc(), calloc(), realloc(), and aligned_alloc() available since these functions depend on errno which is not pre-qualified.

4.10.7 No Filesystem Support

The file systems supported by RTEMS are not pre-qualified. Therefore, the application configuration options CONFIGURE_MAXIMUM_FILE_DESCRIPTORS and CONFIGURE_APPLICATION_DISABLE_FILESYSTEM are mandatory for each application which is restricted to only use pre-qualified interfaces of RTEMS:

```c
#define CONFIGURE_MAXIMUM_FILE_DESCRIPTORS 0

#define CONFIGURE_APPLICATION_DISABLE_FILESYSTEM
```

If you do not have these option defined, then you get linker errors for an undefined reference to `rtems_filesystem_initialize`. For an example which uses these application configuration options, see *Build an Example Application*.

4.10.8 No Newlib Reentrancy Support

The Newlib reentrancy support is not pre-qualified. Therefore, the application configuration option CONFIGURE_DISABLE_NEWLIB_REENTRANCY is mandatory for each application which is restricted to only use pre-qualified interfaces of RTEMS:

```
#define CONFIGURE_DISABLE_NEWLIB_REENTRANCY
```

If you do not have this option defined, then you get linker errors for undefined references to `newlib_create_hook` and `newlib_terminate_hook`. For an example which uses this application configuration option, see *Build an Example Application*.

4.10.9 Linker Command File Requirements

The RTEMS version of the QDP supports thread-local storage (*TLS*) and uses a linker set based system initialization. For this some linker output section definitions are required to be present in the linker command file. It is recommended to use the linker command file installed by the BSP.

By default, RTEMS is compiled with function and data sections enabled (`-ffunction-sections` and `-fdata-sections`). Applications should be linked with the linker garbage collection enabled (`-Wl,--gc-sections`). For this it is necessary that some linker input sections (for example the static constructor and destructor sections) use the `KEEP()` directive. For a reference, see the linker command file installed by the BSP. Previous RTEMS versions did not use these options, so the `KEEP()` directive may be missing in custom linker command files.

The linker output section definitions presented next are mandatory in linker command files used for the RTEMS version delivered by the QDP and may be missing in custom linker command files which worked with previous RTEMS version. There shall be the following linker output section definition in a read-only memory area to support thread-local storage:

```
.tdata : {
  _TLS_Data_begin = .;
  *(.tdata .tdata.* .gnu.linkonce.td.*)
  _TLS_Data_end = .;
}
.tbss : {
  _TLS_BSS_begin = .;
  *(.tbss .tbss.* .gnu.linkonce.tb.*) *(.tcommon)
  _TLS_BSS_end = .;
}
_TLS_Data_size = _TLS_Data_end - _TLS_Data_begin;
_TLS_Data_begin = _TLS_Data_size != 0 ? _TLS_Data_begin : _TLS_BSS_begin;
_TLS_Data_end = _TLS_Data_size != 0 ? _TLS_Data_end : _TLS_BSS_begin;
```

(continues on next page)

(continued from previous page)

```
_TLS_BSS_size = _TLS_BSS_end - _TLS_BSS_begin;
_TLS_Size = _TLS_BSS_end - _TLS_Data_begin;
_TLS_Alignment = MAX (ALIGNOF (.tdata), ALIGNOF (.tbss));
```

The section for the thread-local storage data can and should be read-only since it contains only loadable initialization data. This data is used to initialize the thread-local storage area of a thread during its creation.

There shall be the following linker output section definition in a read-only memory area to support the read-only RTEMS linker sets:

```
.rtemsroset : {
  KEEP (*(SORT(.rtemsroset.*)))
}
```

There shall be the following linker output section definition in a read-write memory area to support the read-write RTEMS linker sets:

```
.rtemsrwset : {
  KEEP (*(SORT(.rtemsrwset.*)))
}
```

There shall be the following linker output section definition in an uninitialized read-write memory area to support the statically allocated RTEMS tasks:

```
.rtemsstack (NOLOAD) : {
  *(SORT_BY_ALIGNMENT (SORT_BY_NAME (.rtemsstack*)))
}
```

The `NOLOAD` output section type just marks the section as not loadable. This section is similar to the `.bss` section except that the content of the `.rtemsstack` section is not zero initialized. It is not initialized at all and may have arbitrary content. This reduces the time to boot the application. The section still uses memory space and shall not overlap with other sections.

Make sure that your MMU initialization covers the new sections.

4.10.10 No EDISOFT RTEMS Improvement Error Reporting

A unique feature of EDISOFT RTEMS Improvement is that fatal and non-fatal errors can be reported, stored, and retrieved. The error reporting is done by all directives. Applications can report errors with the `rtems_error_report()` directive and retrieve reported errors through the `rtems_error_get_latest_non_fatal_by_offset()` and `rtems_error_get_latest_fatal_by_offset()` directives. These directives are not supported by any RTEMS version of the RTEMS Project. They are also not supported by the RTEMS version delivered by the QDP. Applications should do the error reporting if needed on their own. The RTEMS Project offers alternative tracing solutions, see also the RTEMS User Manual.

4.10.11 Customize the Inter-Processor Interrupt Number

For the SMP support of RTEMS, an inter-processor interrupt is required. By default, interrupt number 14 is used for the inter-processor interrupt on the LEON3 BSPs. You can define the interrupt number in your application configuration with this code, for example if the default value conflicts with one of your device drivers. The inter-processor interrupt should have a low priority since it performs no timing critical activities.

```
#include <bsp.h>

const unsigned char LEON3_mp_irq = 13;

/* Your application configuration options */

#include <rtems/confdefs.h>
```

4.10.12 Hardware Errata

The pre-qualified items provided by the QDP address all hardware errata published before 2021-12-10. It is recommended to consult the documentation provided by the hardware vendor with respect to hardware errata workarounds and limitations.

4.11 Pre-Qualified Interfaces

The following sections list the pre-qualified functions, macros, and application configuration options which may be directly used by applications.

4.11.1 Application Configuration Information

- `rtems_configuration_get_idle_task()`
- `rtems_configuration_get_idle_task_stack_size()`
- `rtems_configuration_get_interrupt_stack_size()`
- `rtems_configuration_get_maximum_barriers()`
- `rtems_configuration_get_maximum_extensions()`
- `rtems_configuration_get_maximum_message_queues()`
- `rtems_configuration_get_maximum_partitions()`
- `rtems_configuration_get_maximum_periods()`
- `rtems_configuration_get_maximum_ports()`
- `rtems_configuration_get_maximum_processors()`
- `rtems_configuration_get_maximum_regions()`
- `rtems_configuration_get_maximum_semaphores()`

- rtems_configuration_get_maximum_tasks()

- rtems_configuration_get_maximum_timers()

- rtems_configuration_get_microseconds_per_tick()

- rtems_configuration_get_milliseconds_per_tick()

- rtems_configuration_get_nanoseconds_per_tick()

- rtems_configuration_get_stack_allocate_for_idle_hook()

- rtems_configuration_get_stack_allocate_hook()

- rtems_configuration_get_stack_allocator_avoids_work_space()

- rtems_configuration_get_stack_free_hook()

- rtems_configuration_get_ticks_per_timeslice()

4.11.2 Barrier Manager

- rtems_barrier_create()

- rtems_barrier_delete()

- rtems_barrier_ident()

- rtems_barrier_release()

- rtems_barrier_wait()

4.11.3 Base Definitions

- RTEMS_ALIAS()

- RTEMS_ALIGN_DOWN()

- RTEMS_ALIGN_UP()

- RTEMS_ALIGNED()

- RTEMS_ALIGNOF()

- RTEMS_ALLOC_ALIGN()

- RTEMS_ALLOC_SIZE()

- RTEMS_ALLOC_SIZE_2()

- RTEMS_ARRAY_SIZE()

- RTEMS_COMPILER_MEMORY_BARRIER()

- RTEMS_CONCAT()

- RTEMS_CONTAINER_OF()

- RTEMS_DECLARE_GLOBAL_SYMBOL()

- RTEMS_DECONST()

- RTEMS_DEFINE_GLOBAL_SYMBOL()

- RTEMS_DEQUALIFY()

- RTEMS_DEQUALIFY_DEPTHX()

- RTEMS_DEVOLATILE()

- RTEMS_EXPAND()

- RTEMS_HAVE_MEMBER_SAME_TYPE()

- RTEMS_OBFUSCATE_VARIABLE()

- RTEMS_PREDICT_FALSE()

- RTEMS_PREDICT_TRUE()

- RTEMS_PRINTFLIKE()

- RTEMS_RETURN_ADDRESS()

- RTEMS_SECTION()

- RTEMS_STATIC_ASSERT()

- RTEMS_STRING()

- RTEMS_SYMBOL_NAME()

- RTEMS_TYPEOF_REFX()

- RTEMS_UNREACHABLE()

- RTEMS_WEAK_ALIAS()

- RTEMS_XCONCAT()

- RTEMS_XSTRING()

4.11.4 C and POSIX

- clock_nanosleep()

- flsl()

- memcpy()

- memset()

- posix_memalign()

4.11.5 CPU Usage Reporting

- rtems_cpu_usage_reset()

4.11.6 Cache Manager

- rtems_cache_disable_data()

- rtems_cache_disable_instruction()

- rtems_cache_enable_data()

- rtems_cache_enable_instruction()

- rtems_cache_flush_entire_data()

- rtems_cache_flush_multiple_data_lines()

- rtems_cache_get_data_line_size()

- rtems_cache_get_data_cache_size()

- rtems_cache_get_instruction_line_size()

- rtems_cache_get_instruction_cache_size()

- rtems_cache_get_maximal_line_size()

- rtems_cache_instruction_sync_after_code_change()

- rtems_cache_invalidate_entire_data()

- rtems_cache_invalidate_entire_instruction()

- rtems_cache_invalidate_multiple_data_lines()

- rtems_cache_invalidate_multiple_instruction_lines()

4.11.7 Classic API Configuration

- CONFIGURE_MAXIMUM_BARRIERS

- CONFIGURE_MAXIMUM_MESSAGE_QUEUES

- CONFIGURE_MAXIMUM_PARTITIONS

- CONFIGURE_MAXIMUM_PERIODS

- CONFIGURE_MAXIMUM_SEMAPHORES

- CONFIGURE_MAXIMUM_TASKS

- CONFIGURE_MAXIMUM_THREAD_LOCAL_STORAGE_SIZE

- CONFIGURE_MAXIMUM_TIMERS

- CONFIGURE_MAXIMUM_USER_EXTENSIONS

- CONFIGURE_MINIMUM_TASKS_WITH_USER_PROVIDED_STORAGE

4.11.8 Classic API Initialization Task Configuration

- CONFIGURE_INIT_TASK_ARGUMENTS

- CONFIGURE_INIT_TASK_ATTRIBUTES

- CONFIGURE_INIT_TASK_CONSTRUCT_STORAGE_SIZE

- CONFIGURE_INIT_TASK_ENTRY_POINT

- CONFIGURE_INIT_TASK_INITIAL_MODES

- CONFIGURE_INIT_TASK_NAME

- CONFIGURE_INIT_TASK_PRIORITY

- CONFIGURE_RTEMS_INIT_TASKS_TABLE

4.11.9 Clock Manager

- rtems_clock_get_boot_time()

- rtems_clock_get_boot_time_bintime()

- rtems_clock_get_boot_time_timeval()

- rtems_clock_get_monotonic()

- rtems_clock_get_monotonic_bintime()

- rtems_clock_get_monotonic_coarse()

- rtems_clock_get_monotonic_coarse_bintime()

- rtems_clock_get_monotonic_coarse_timeval()

- rtems_clock_get_monotonic_sbintime()

- rtems_clock_get_monotonic_timeval()

- rtems_clock_get_realtime()

- rtems_clock_get_realtime_bintime()

- rtems_clock_get_realtime_coarse()

- rtems_clock_get_realtime_coarse_bintime()

- rtems_clock_get_realtime_coarse_timeval()

- rtems_clock_get_realtime_timeval()

- rtems_clock_get_ticks_per_second()

- rtems_clock_get_ticks_since_boot()

- rtems_clock_get_tod()

- rtems_clock_get_uptime()

- rtems_clock_set()

4.11.10 Device Driver Configuration

- CONFIGURE_APPLICATION_DOES_NOT_NEED_CLOCK_DRIVER

- CONFIGURE_APPLICATION_NEEDS_CLOCK_DRIVER

4.11.11 Directive Status Codes

- rtems_are_statuses_equal()

- rtems_is_status_successful()

- rtems_status_text()

4.11.12 Dynamic Memory Allocation

- rtems_calloc()

- rtems_malloc()

4.11.13 Event Manager

- rtems_event_receive()

- rtems_event_send()

4.11.14 Fatal Error Manager

- rtems_fatal()

4.11.15 Filesystem Configuration

- CONFIGURE_APPLICATION_DISABLE_FILESYSTEM

4.11.16 General Scheduler Configuration

- CONFIGURE_MAXIMUM_PRIORITY

- RTEMS_SCHEDULER_ASSIGN()

- CONFIGURE_SCHEDULER_ASSIGNMENTS

- CONFIGURE_SCHEDULER_EDF_SMP

- CONFIGURE_SCHEDULER_NAME

- CONFIGURE_SCHEDULER_PRIORITY

- CONFIGURE_SCHEDULER_TABLE_ENTRIES

4.11.17 General System Configuration

- CONFIGURE_DISABLE_BSP_SETTINGS

- CONFIGURE_DISABLE_NEWLIB_REENTRANCY

- CONFIGURE_INITIAL_EXTENSIONS

- CONFIGURE_INTERRUPT_STACK_SIZE

- CONFIGURE_MAXIMUM_FILE_DESCRIPTORS

- CONFIGURE_MAXIMUM_PROCESSORS

- CONFIGURE_MICROSECONDS_PER_TICK

- CONFIGURE_MINIMUM_TASK_STACK_SIZE

- CONFIGURE_TICKS_PER_TIMESLICE

4.11.18 Idle Task Configuration

- CONFIGURE_IDLE_TASK_BODY

- CONFIGURE_IDLE_TASK_INITIALIZES_APPLICATION

- CONFIGURE_IDLE_TASK_STACK_SIZE

4.11.19 Interrupt Manager

- rtems_interrupt_clear()

- rtems_interrupt_entry_initialize()

- RTEMS_INTERRUPT_ENTRY_INITIALIZER()

- rtems_interrupt_entry_install()

- rtems_interrupt_entry_remove()

- rtems_interrupt_get_affinity()

- rtems_interrupt_get_attributes()

- rtems_interrupt_handler_iterate()

- rtems_interrupt_is_pending()

- rtems_interrupt_local_disable()

- rtems_interrupt_local_enable()

- rtems_interrupt_lock_acquire()

- rtems_interrupt_lock_acquire_isr()

- RTEMS_INTERRUPT_LOCK_DECLARE()

- RTEMS_INTERRUPT_LOCK_DEFINE()

- rtems_interrupt_lock_destroy()

- rtems_interrupt_lock_initialize()

- RTEMS_INTERRUPT_LOCK_INITIALIZER()

- rtems_interrupt_lock_interrupt_disable()

- RTEMS_INTERRUPT_LOCK_MEMBER()

- RTEMS_INTERRUPT_LOCK_REFERENCE()

- rtems_interrupt_lock_release()

- rtems_interrupt_lock_release_isr()

- rtems_interrupt_raise()

- rtems_interrupt_raise_on()

- rtems_interrupt_set_affinity()

- rtems_interrupt_vector_disable()

- rtems_interrupt_vector_enable()

- rtems_interrupt_vector_is_enabled()

4.11.20 Kernel Character I/O Support

- getchark()

- rtems_put_char()

- rtems_putc()

4.11.21 Message Manager

- rtems_message_queue_broadcast()

- RTEMS_MESSAGE_QUEUE_BUFFER()

- rtems_message_queue_construct()

- rtems_message_queue_delete()

- rtems_message_queue_flush()

- rtems_message_queue_get_number_pending()

- rtems_message_queue_ident()

- rtems_message_queue_receive()

- rtems_message_queue_send()

- rtems_message_queue_urgent()

4.11.22 Newlib

- `_Futex_Wait()`

- `_Futex_Wake()`

- `_Mutex_Acquire()`

- `_Mutex_Acquire_timed()`

- `_Mutex_recursive_Acquire()`

- `_Mutex_recursive_Acquire_timed()`

- `_Mutex_recursive_Release()`

- `_Mutex_recursive_Try_acquire()`

- `_Mutex_Release()`

- `_Mutex_Try_acquire()`

4.11.23 Object Services

- rtems_build_name()

- rtems_object_get_local_node()

4.11.24 Partition Manager

- rtems_partition_create()

- rtems_partition_delete()

- rtems_partition_get_buffer()

- rtems_partition_ident()

- rtems_partition_return_buffer()

4.11.25 Rate-Monotonic Manager

- rtems_rate_monotonic_cancel()

- rtems_rate_monotonic_create()

- rtems_rate_monotonic_delete()

- rtems_rate_monotonic_get_status()

- rtems_rate_monotonic_ident()

- rtems_rate_monotonic_period()

4.11.26 Scheduler Manager

- rtems_scheduler_add_processor()

- rtems_scheduler_get_maximum_priority()

- rtems_scheduler_get_processor()

- rtems_scheduler_get_processor_maximum()

- rtems_scheduler_get_processor_set()

- rtems_scheduler_ident()

- rtems_scheduler_ident_by_processor()

- rtems_scheduler_ident_by_processor_set()

- rtems_scheduler_remove_processor()

4.11.27 Semaphore Manager

- rtems_semaphore_create()

- rtems_semaphore_delete()

- rtems_semaphore_flush()

- rtems_semaphore_ident()

- rtems_semaphore_obtain()

- rtems_semaphore_release()

- rtems_semaphore_set_priority()

4.11.28 Signal Manager

- rtems_signal_catch()

- rtems_signal_send()

4.11.29 Support Services

- rtems_is_name_valid()

- rtems_name_to_characters()

4.11.30 Task Manager

- rtems_task_construct()
- rtems_task_create()
- rtems_task_delete()
- rtems_task_exit()
- rtems_task_get_affinity()
- rtems_task_get_priority()
- rtems_task_get_scheduler()
- rtems_task_ident()
- rtems_task_is_suspended()
- rtems_task_iterate()
- rtems_task_mode()
- rtems_task_restart()
- rtems_task_resume()
- rtems_task_self()
- rtems_task_set_affinity()
- rtems_task_set_priority()
- rtems_task_set_scheduler()
- rtems_task_start()
- RTEMS_TASK_STORAGE_SIZE()
- rtems_task_suspend()
- rtems_task_wake_after()
- rtems_task_wake_when()

4.11.31 Task Modes

- RTEMS_INTERRUPT_LEVEL()

4.11.32 Task Stack Allocator Configuration

- CONFIGURE_TASK_STACK_ALLOCATOR

- CONFIGURE_TASK_STACK_ALLOCATOR_FOR_IDLE

- CONFIGURE_TASK_STACK_DEALLOCATOR

- CONFIGURE_TASK_STACK_ALLOCATOR_AVOIDS_WORK_SPACE

4.11.33 Timer Manager

- rtems_timer_cancel()

- rtems_timer_create()

- rtems_timer_delete()

- rtems_timer_fire_after()

- rtems_timer_fire_when()

- rtems_timer_ident()

- rtems_timer_initiate_server()

- rtems_timer_reset()

- rtems_timer_server_fire_after()

- rtems_timer_server_fire_when()

4.11.34 User Extensions Manager

- rtems_extension_create()

- rtems_extension_delete()

- rtems_extension_ident()

CHAPTER

FIVE

INVENTORY OF MATERIALS

There are no physical media included in the QDP. In particular, the boards used to perform the test runs are not included.

CHAPTER

SIX

BASELINE DOCUMENTS

The QDP contains all project-specific baseline documents. Please refer to *ECSS Software Documentation Overview*, *Technical Notes Overview*, *RTEMS Documentation Overview*, *Formal Methods Documentation Overview*, and *ISVV Documentation Overview*.

CHAPTER

SEVEN

INVENTORY OF SOFTWARE CONFIGURATION ITEM

This document may be digitally signed. Check that the signatures are valid and trustworthy. This document corresponds to the QDP archive file `rtems-6-sparc-gr740-smp-5.tar.xz` which has an SHA512 digest of `cc74a2db2ae2341f32a4aee09dc772dcd31be82643f98092215` `0474783cd5d108efd29df04c68c065f4b841ab52704c46d98665237975aacc332018a52d1c6160`. You should have obtained the archive file from the same distribution channel as this document. Check that the SHA512 digest of the QDP archive file matches with the SHA512 digest documented in this document. If the digests do not match, then the archive file is not the right one.

```
$ sha512sum rtems-6-sparc-gr740-smp-5.tar.xz
cc74a2db2ae2341f32a4aee09dc772dcd31be82643f98092150474783cd5d108efd29df04c68c065f4
b841ab52704c46d98665237975aacc332018a52d1c6160
```

For an overview of the QDP archive content, see *Software Configuration Item Overview*. The QDP archive contains the package verification script `verify_package.py` which has an SHA512 digest of `d7740026fa29e26d6c6a7a1b008ca098fa8f8f817b20f3c2af8f92c66cad50244bde76fac9f7` `a4902612a718139672eb121e92ecca6bfe1d4b68388d8430d6b8`. This script may be used to verify the files of the QDP after unpacking the archive. It may also be used to list the files of the QDP or to list the files of the QDP with an SHA512 hash value of each file content.

```
$ cd /opt/rtems-6-sparc-gr740-smp-5
$ ./verify_package.py --help
usage: verify_package.py [-h]
                         [--log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL}]
                         [--log-file LOG_FILE] [--list-files]
                         [--list-files-and-hashes]

optional arguments:
  -h, --help            show this help message and exit
  --log-level {DEBUG,INFO,WARNING,ERROR,CRITICAL}
                        log level
  --log-file LOG_FILE   log to this file
  --list-files          list the files of the package
  --list-files-and-hashes
                        list the files of the package with the SHA512 digest
                        of each file
```

CHAPTER

EIGHT

MEANS NECESSARY FOR THE SOFTWARE CONFIGURATION ITEM

The QDP needs a host computer or a Docker container, see *Prepare the Host Computer* and *Use QDP in a Docker solution*. The QDP can be used to build applications using RTEMS for the target system. You have to provide the target system and any means to deploy executables on it.

CHAPTER

NINE

INSTALLATION INSTRUCTIONS

The getting started section gives you step by step instructions which enable you to build a simple example application using the QDP.

9.1 Prepare the Host Computer

The QDP contains a binary distribution of tools such as the cross *GNU* Binutils and *GCC*. The tools were built on a Debian 10 machine. The tools should run on any Linux distribution which provides a glibc 2.27.

9.1.1 Debian

The following packages are required:

```
$ apt-get install libncurses5 make pkg-config
```

9.1.2 openSUSE

On openSUSE distributions install the *C/C++ Development* pattern.

9.2 Check the QDP Integrity

This document may be digitally signed. Check that the signatures are valid and trustworthy. This document corresponds to the QDP archive file `rtems-6-sparc-gr740-smp-5.tar.xz` which has an SHA512 digest of `cc74a2db2ae2341f32a4aee09dc772dcd31be82643f98092150474783cd5d108efd29df04c68c065f4b841ab52704c46d98665237975aacc332018a52d1c6160`. You should have obtained the archive file from the same distribution channel as this document. Check that the SHA512 digest of the QDP archive file matches with the SHA512 digest documented in this document. If the digests do not match, then the archive file is not the right one.

```
$ sha512sum rtems-6-sparc-gr740-smp-5.tar.xz
cc74a2db2ae2341f32a4aee09dc772dcd31be82643f98092150474783cd5d108efd29df04c68c065f4
b841ab52704c46d98665237975aacc332018a52d1c6160
```

9.3 Unpack the QDP Archive

It is recommended to unpack the QDP archive in /opt. Other locations may work, but this use case was not tested and is not documented.

```
$ cd /opt
$ sudo tar xf rtems-6-sparc-gr740-smp-5.tar.xz
```

9.4 Build an Example Application

The QDP contains an example application for the pre-qualified BSP. Build it to check that the tools work on your host computer.

```
$ cd /opt/rtems-6-sparc-gr740-smp-5/src/example
$ make
```

This should produce the following output:

```
mkdir b-gr740-qual-only
sparc-rtems6-gcc  -MT b-gr740-qual-only/init.o -MD -MP -MF b-gr740-qual-only/init.
↪d -Wall -Wextra -mcpu=leon3 -I/opt/rtems-6-sparc-gr740-smp-5/sparc-rtems6/gr740-
↪qual-only/lib/include -O2 -g -ffunction-sections -fdata-sections -c init.c -o b-
↪gr740-qual-only/init.o
sparc-rtems6-gcc -MT b-gr740-qual-only/app.exe -MD -MP -MF b-gr740-qual-only/app.
↪d -Wall -Wextra -mcpu=leon3 -I/opt/rtems-6-sparc-gr740-smp-5/sparc-rtems6/gr740-
↪qual-only/lib/include -O2 -g -ffunction-sections -fdata-sections -Wl,-Map,b-
↪gr740-qual-only/app.map b-gr740-qual-only/init.o -B/opt/rtems-6-sparc-gr740-smp-
↪5/sparc-rtems6/gr740-qual-only/lib -qrtems -Wl,--gc-sections -nodefaultlibs -Wl,
↪--start-group -lrtemscpu -lrtemsbsp -lgcc -Wl,--end-group -o b-gr740-qual-only/
↪app.exe
```

If you get a different output, then there is something wrong with the unpacked QDP or your host computer setup.

Try to run the example application b-gr740-qual-only/app.exe on a simulator or a target platform. The QDP provides a simulator, which can be used to run this application as follows:

```
$ cd /opt/rtems-6-sparc-gr740-smp-5/src/example
$ make run
```

This should produce the following output:

```
sparc-rtems6-sis -gr740 -dumbio -r s b-gr740-qual-only/app.exe

 SIS - SPARC/RISCV instruction simulator 2.29,  copyright Jiri Gaisler 2020
 Bug-reports to jiri@gaisler.se

 GR740/LEON4 emulation enabled, 1 cpus online, delta 50 clocks
```

(continues on next page)

<div align="right">(continued from previous page)</div>

```
 Loaded b-gr740-qual-only/app.exe, entry 0x00000000
Hello, world!
cpu 0 in error mode (tt = 0x80)
    83178  00008660:  91d02000   ta  0x0
```

If you get a different output, then there is something wrong with the unpacked QDP or your host computer setup.

9.5 Build a C++ Example Application

The QDP contains a C++ example application for the pre-qualified BSP. Build it to check that the tools work on your host computer.

```
$ cd /opt/rtems-6-sparc-gr740-smp-5/src/example++
$ make
```

This should produce the following output:

```
mkdir b-gr740-qual-only
sparc-rtems6-g++  -MT b-gr740-qual-only/init.o -MD -MP -MF b-gr740-qual-only/init.
↪d -Wall -Wextra -mcpu=leon3 -I/opt/rtems-6-sparc-gr740-smp-5/sparc-rtems6/gr740-
↪qual-only/lib/include -O2 -g -ffunction-sections -fdata-sections -fno-
↪exceptions -c init.cc -o b-gr740-qual-only/init.o
sparc-rtems6-g++ -MT b-gr740-qual-only/app.exe -MD -MP -MF b-gr740-qual-only/app.
↪d -Wall -Wextra -mcpu=leon3 -I/opt/rtems-6-sparc-gr740-smp-5/sparc-rtems6/gr740-
↪qual-only/lib/include -O2 -g -ffunction-sections -fdata-sections -fno-
↪exceptions -Wl,-Map,b-gr740-qual-only/app.map b-gr740-qual-only/init.o -B/opt/
↪rtems-6-sparc-gr740-smp-5/sparc-rtems6/gr740-qual-only/lib -qrtems -Wl,--gc-
↪sections -nodefaultlibs -Wl,--start-group -lrtemscpu -lrtemsbsp -lgcc -Wl,--end-
↪group -o b-gr740-qual-only/app.exe
```

If you get a different output, then there is something wrong with the unpacked QDP or your host computer setup.

Try to run the example application `b-gr740-qual-only/app.exe` on a simulator or a target platform. The QDP provides a simulator, which can be used to run this application as follows:

```
$ cd /opt/rtems-6-sparc-gr740-smp-5/src/example++
$ make run
```

This should produce the following output:

```
sparc-rtems6-sis -gr740 -dumbio -r s b-gr740-qual-only/app.exe

 SIS - SPARC/RISCV instruction simulator 2.29,  copyright Jiri Gaisler 2020
 Bug-reports to jiri@gaisler.se
```

<div align="right">(continues on next page)</div>

(continued from previous page)

```
 GR740/LEON4 emulation enabled, 1 cpus online, delta 50 clocks


 Loaded b-gr740-qual-only/app.exe, entry 0x00000000
Hello, C++ world!
cpu 0 in error mode (tt = 0x80)
    21319  00004e00:  91d02000    ta  0x0
```

If you get a different output, then there is something wrong with the unpacked QDP or your host computer setup.

9.6 Use QDP in a Docker solution

This section explains how to set-up a dockerized solution to use the QDP, which allows the usage of the QDP in any operating system. The following steps shall be followed for this set up:

1. Extract the QDP archive file `rtems-6-sparc-gr740-smp-5.tar.xz` into a folder, ex: docker-for-rtems.

```
$ ~/docker-for-rtems
```

2. Place the dockerfile on the above directory, on the same level as the QDP extracted directory.

```
$ ~/docker-for-rtems$ ls
Dockerfile.rtems-qdp  rtems-6-sparc-gr740-smp-5
```

The Dockerfile.rtems-qdp file has the following content:

```
FROM debian:buster
RUN apt-get update
RUN apt-get -y install libncurses5
RUN apt-get -y install pkg-config
RUN apt-get -y install make
RUN mkdir -p /opt/rtems-6-sparc-gr740-smp-5
```

Note: This Dockerfile will allow to use the QDP, but not running the RTEMS qualification as presented in *Guidance for RTEMS Qualification in User's Environment*. The necessary Dockerfile to do qualification activity is described in the mentioned section.

3. Open the command line in the ~/docker-for-rtems folder and process the Dockerfile with

```
$ ~/docker-for-rtems$ docker build -f Dockerfile.rtems-qdp -t rtems_qdp .
```

4. Run the docker image with

```
$ ~/docker-for-rtems$ docker run --rm -v ~/docker-for-rtems/rtems-6-sparc-gr740-
↪smp-5:/opt/rtems-6-sparc-gr740-smp-5 -it rtems_qdp /bin/bash
```

5. The above command will map the QDP content on ∼/docker-for-rtems/rtems-6-sparc-gr740-smp-5 to the /opt/rtems-6-sparc-gr740-smp-5 inside docker. This means that the operations with QDP done on /opt/rtems-6-sparc-gr740-smp-5 will be available to the host system. Additionally, the user can create a another folder inside rtems-6-sparc-gr740-smp-5, to contain the user's work:

```
$ root@c5cf83823b8f:/opt/rtems-6-sparc-gr740-smp-5# ls
bin  doc  include  lib  libexec  make  share  sparc-rtems6  src  verify_package.
↪py  workdir
```

6. Compile and execute the example as described in the previous section. If everything runs as expected, the docker set up is correct.

CHAPTER

TEN

CHANGE LIST

For changes not incorporated yet but affecting the software configuration item please refer to *Possible Problems and Known Errors*.

10.1 Package Version 1

The first version of the QDP contains a development snapshot which was issued for the *CDR*.

10.2 Package Version 2

The second version of the QDP contains a development snapshot which was issued for the *QR*/2.

10.3 Package Version 3

The third version of the QDP contains the final release issued for project *Qualification of RTEMS Symmetric Multiprocessing (SMP)* ESA Contract No. 4000125572/18/NL/GLC/as.

10.4 Package Version 4

The fourth version of the QDP contains a warranty release issued for project *Qualification of RTEMS Symmetric Multiprocessing (SMP)* ESA Contract No. 4000125572/18/NL/GLC/as. There are the following changes compared to package version 3:

- The RTEMS baseline changed. In package version 3, we had about 120 QDP-specific patches on top of the RTEMS baseline. Now, there are only 46 patches left. These patches fall roughly into three categories

   - BSP-specific changes,

   - changes in the build system to support building the pre-qualified only feature set,

   - performance and formal methods tests.

- Most of the validation tests are now included in the RTEMS baseline. Test runs on ARM and PowerPC platforms were performed showing good results comparable to the current QDP platforms.

- The relicensing to BSD-2-Clause of the RTEMS sources by the RTEMS Project progressed significantly compared to package version 3.

- The previous RTEMS baseline contained a bug which could result in a non-monotonic `CLOCK_MONOTONIC`. This bug is fixed in this package version. See also RTEMS ticket #4617.

- The specification and validation of some application configuration options was improved. Now only test cases and validations by inspections are used as validation methods for application configuration options. Entire test suites are no longer used as a validation evidence for a particular requirement.

- Significant changes in the Software Requirement Specification (SRS) [HK+23b] and Software Verification Report (SVR) [HK+23e], see the change log of the documents.

10.5 Package Version 5

The fifth version of the Qualification Data Package (QDP) adds several corrections as a result of the Independent Software Verification and Validation (ISVV) activities as in the table below:

Table 1: Findings already addressed

ISVV Finding[Page 64, 2]	Finding Title	Notes
REQ-VER-008	Undefined behaviour	Following the independent validation, requirements related with qualified items were updated as to better describe behaviour upon invalid inputs/combinations.
REQ-VER-010	No requirements	Default text improved.
REQ-VER-022	Performance limits	Added units and filtered performance limits according to board.
REQ-VER-024	References to incorrect board	Updated section 5 of the SRS.
REQ-VER-041	Units for data representing a physical quantity	Added units for the values referring to size.
REQ-VER-043	Indexed bookmarks	Added indexed bookmarks to QDP documents.
FM-001	Invalid Thread wait flags used	Corrected sources.
FM-002	Run function prototype name not matching with call	Corrected sources.
FM-003	Unfinished work is deployed and causing compilation issues	Corrected sources.
TS-VER-009	Boundary tests for TS validation	Updated sections 5.1 and 7.1 of the SUITP and section 5.1 of the SVS as to clarify the approach.

Table 1 – continued from previous page

ISVV Finding[Page 64, 2]	Finding Title	Notes
TS-VER-010	Validation Plan contains limited elaboration	Addressed as part of the IVA test campaign.
TS-VER-011	Testing multiple task interactions	Addressed as part of the IVA test campaign.
TS-VER-013	Incomplete SUITP strategy	Updated sections 5.1 and 7.1 of the SUITP and section 5.1 of the SVS as to clarify the approach.
TS-VER-019	Unit test objectives reached through TS tests	Updated sections 5.1 and 7.1 of the SUITP and section 5.1 of the SVS as to clarify the approach.
DESIGN-VER-010	No architectural design	SDD now contains an Architectural Design section rather than just the Detailed Design generated from Doxygen.
CODE-VER-014	**Macros wrongly defined**	**Partially addressed.** *The generation (from RTEMS specifications) of ftmctrl-regs.h, grgpio-regs.h, spwpnp-regs.h, and spwrmap-regs.h is fixed.* Eventual issues in dsu4-regs.h and gr1553b-regs.h are, as of the time of writing, still under discussion/clarification – please refer to #4842 - Registers definitions wrongly defined when there are reserved bits.
CODE-VER-032	**Output value not assigned**	**Partially addressed.** *For bsp_interrupt_* functions, it has been clarified that the presence of error conditions is implementation-defined.* Still to be addressed for leon3_install_inter_processor_interrupt.
CODE-VER-036	Unchecked boundaries	Added note within rtems_message_queue_send specification/ICD and added new _Assert() in the code.
CODE-VER-041	Variable not initialized in any file	Implemented the proposed correction.
CODE-VER-043	Global variable declared/defined in the wrong file	Moved the definition of this object to start.S and used a local symbol .Lbootcpuindex for it. Use a compare-and-swap instruction to assign the boot CPU. This allows a concurrent initialization.
CODE-VER-057	Function returning unchanged function input	Removed unused return values from functions' signatures.
CODE-VER-058	Incomplete statement	Not used a chained assignment for code clarity.
CODE-VER-074	*goto statements*	The goto label was directly after the loop, so it has been replaced with a break.

continues on next page

Table 1 – continued from previous page

ISVV Finding[Page 64, 2]	Finding Title	Notes
CODE-VER-076	"!", "&&" or "||" operand should have type "bool"	Simplified _Objects_Is_api_valid.
CODE-VER-079	Hazardous cast	Added TM27_USE_VECTOR_HANDLER to select the interrupt handler type used by the <tm27.h> implementation, avoiding function pointer casts.
CODE-VER-083	Requirement not fully implemented	Condition excluding nested usage removed from requirement.
VAL-002	Use of macros not in the pre-qualified set	Removed directive-not-pre-qualified from these interfaces/macros and improved requirements specifications of these clock manager directives.
VAL-004	Documentation refers to undefined behaviour on cases where an error is to be expected	Following the independent validation, documentation has been updated as to better describe behaviour upon invalid inputs/combinations.

Moreover a new set of ISVV related documents were added, namely verification and validation plans and reports and the overall list of ISVV findings.

In a nutshell, the previous Qualification Data Package (QDP) release 4 (which was already an outcome of the RTEMS qualification efforts) has been used as the baseline for the improvements identified during the ISVV activities. Although these QDPs have been based on the RTEMS 6, other meanwhile changes (not related to ISVV outcomes) on the RTMES public mainline have not been integrated into the QDP not to compromise/jeopardise the QDP qualification that has been performed using this baseline.

It must be noticed that not all of the identified and confirmed ISVV Findings have been corrected. Please refer to *Possible Problems and Known Errors* for the list of known problems not addressed.

For further details one can also refer to the RTEMS SMP ISVV Verification Report [F+23] within the QDP.

---

2 Consider these Findings' IDs preceded by "RTEMS-SMP-".

CHAPTER

ELEVEN

AUXILIARY INFORMATION

11.1 Service Providers

11.1.1 embedded brains

The embedded brains GmbH offers a wide range of RTEMS expert support services. This includes the entire tool suite (for example Binutils, GDB, Newlib, GCC). The company experts are active members in the RTEMS community. The company is a major contributor to the RTEMS Project and leading in the design and development of the SMP support for RTEMS. It offers training, consulting, and development services.

11.1.2 Thales Portugal

Thales Portugal (former EDISOFT) is managing the RTEMS CENTRE since 2006, a centre to support space missions with the RTEMS Improvement version, a space pre-qualified RTEMS version used in a large variety of ESA missions. Thales Portugal provides helpdesk/technical support, delta-qualification services, including Board Supported Packages, and training.

11.2 Frequently Asked Questions

11.2.1 How can I adopt the QDP to a new BSP, for example leon2?

The QDP provides you with everything to adopt it to a new *BSP*. The main RTEMS repositories are included in the QDP. This gives you full access to the project history in all details. With the *RSB* you are able to build the RTEMS tool suite for other targets or make updates of the tools. RTEMS and its ecosystem is documented in the RTEMS documentation shipped with the QDP, see *RTEMS Documentation Overview*. The tool chain to build a customized QDP is also available with sources and documentation under open source licenses. However, you should think about whether it is your core business to build up expert knowledge in this domain. It is probably faster and more cost effective to involve one of the *Service Providers* to help you with this task. The QDP and its generation process is complicated enough to require expert knowledge and training.

11.2.2 Is C++ supported?

C++ is supported in general, however, for a more complete C++ runtime and standard library support, more *POSIX APIs* need to be pre-qualified. In particular, support for condition variables is recommended. For a basic C++ example application, see *Build a C++ Example Application*.

11.2.3 Is OpenMP supported?

OpenMP provided by *GCC* is excellently supported by mainline RTEMS. For a pre-qualified RTEMS support for OpenMP, more *POSIX APIs* need to be pre-qualified. Also some restructuring in the Newlib C library is necessary to avoid a dependency on the C file streams. The major work is done, just ask one of the *Service Providers* to get is supported.

11.2.4 Is Ada supported?

For Ada support, more *POSIX APIs* need to be pre-qualified.

11.3 Guidance for RTEMS Qualification in User's Environment

The RTEMS software product included in the QDP is provided with pre-qualification tests already performed. However, it does not have baseline requirements, so the product assurance should assess the mission requirements. RTEMS is provided with a test package run (using the ESA hardware), however, after downloading the QDP, the user needs to repeat the testsuite execution and check the new report (Software Verification Report (SVR)), to make sure that the with its own hardware all tests pass, all performance requirements are still met and the coverage is still 100%. Once this is done, the QDP can be considered as qualified for the user hardware and can be used for its application.

The next sub-section will describe the activities that should be performed by both engineering team and PA to qualify the QDP on the users hardware. In case of any problem (ex: a test failure), please contact any of the *Service Providers*.

11.3.1 Engineering activities

The QDP provides the set of the necessary qualification tools to regenerate the Software Verification Report (SVR) in the *qual-tool* folder of the QDP. The first step that needs to be done by the user is to connect the hardware the PC where the QDP is installed or prepare its own setup (ex: a remote connection using the gdb). The file *grmon_command_gr740_smp.sh* should contain the command which is used to connect to the board. For a direct connection of the hardware to the PC, an example is:

```bash
#!/bin/bash
grmon -nb -nswb -abaud 115200 -ftdi -u -e "load $1;run;quit"
```

For a gdb connection, an example is as follows:

```bash
#!/bin/bash
/opt/rtems-6-sparc-gr740-smp-5/bin/sparc-rtems6-gdb --batch --command /opt/rtems-
↪6-sparc-gr740-smp-5/qual-tool/batch.gdb --args  $1
```

Where the *batch.gdb* script is as follows:

```
load
run
quit
```

Another alternative to run the *qual-tool* with a gdb connection is to adapt *config-variants/sparc-gr740-smp-user-qual.yml* by adding the following blocks in **post-process-items** items field:

```yaml
- uid: /steps/run-local-board-target-qual-only
  path: /config-values
  action: set
  value:
  - key: bsp
    value: ${../variant:/bsp}
  - key: arch
    value: ${../variant:/arch}
  - key: tester
    value: '%{_rtscripts}/gdb.cfg'
  - key: gdb_script
    value: 'bsp_gdb_script'
  - key: bsp_gdb_script
    value: |
      target extended-remote 10.128.9.126:2222
                      mon reset
                      load
                      run

  - key: max_test_period
    value: '3600'
```

```yaml
- uid: /steps/run-local-board-target-qual-only-coverage
  path: /config-values
  action: set
  value:
  - key: bsp
    value: ${../variant:/bsp}
  - key: arch
    value: ${../variant:/arch}
  - key: tester
    value: '%{_rtscripts}/gdb.cfg'
  - key: gdb_script
    value: 'bsp_gdb_script'
  - key: bsp_gdb_script
```

(continues on next page)

(continued from previous page)

```
    value: |
      target extended-remote 10.128.9.126:2222
                      mon reset
                      load
                      run


  - key: max_test_period
    value: '3600'
```

The following files contain the configuration for executing the testsuite run:

- config/steps/run-local-board-target-qual-only.yml - configuration for standard run

- config/steps/run-local-board-target-qual-only-coverage.yml - configuration for coverage run

The most relevant fields that the user might want to customize in this configuration files are the **commands**, which is the RTEMS Tester command, **config-values**, which is the RTEMS Tester .ini file (as shown above to set up the .ini file for testsuite runs with gdb), or the **ignore-tests**, which is the ignore pattern for test executables (for example the coverage does not run the performance tests). As already shown, to change these values the user should use the *config-variants/sparc-gr740-smp-user-qual.yml*, which is the file used to change the default configuration. Example to change the maximum test timeout (from the default 1 hour to 2 hours) you would need to add the following blocks in **post-process-items** items field:

```
- uid: /steps/run-local-board-target-qual-only
  path: /commands
  action: set
  value:
  - - '${../variant:/deployment-directory}/bin/rtems-test'
    - '--user-config=${.:config-directory}/${.:config-file}'
    - '--rtems-bsp=${.:config-variant}'
    - '.'
    - '--log-mode=all'
    - '--jobs=1'
    - '--timeout=7200'
    - '--report-format=yaml'
    - '--report-path=${.:executables-directory}/log-run-rtems-qual-only-board'
```

```
- uid: /steps/run-local-board-target-qual-only-coverage
  path: /commands
  action: set
  value:
  - - '${../variant:/deployment-directory}/bin/rtems-test'
    - '--user-config=${.:config-directory}/${.:config-file}'
    - '--rtems-bsp=${.:config-variant}'
    - '.'
    - '--log-mode=all'
    - '--jobs=1'
```

(continues on next page)

(continued from previous page)

```
    - '--timeout=7200'
    - '--report-format=yaml'
    - '--report-path=${.:executables-directory}/log-run-rtems-qual-only-board-cov'
```

```
- uid: /steps/run-local-board-target-qual-only
  path: /config-values[5]/value
  action: set
  value: '7200'
```

```
- uid: /steps/run-local-board-target-qual-only-coverage
  path: /config-values[5]/value
  action: set
  value: '7200'
```

The new SVR document will be generated in the folder *user_doc*, as configured in *config-variants/sparc-gr740-smp-user-qual.yml* (which means that the original SVR will not be over-ridden). Note also, the production of the SVR, will also produce the Software Design Document (SDD) document due to a necessary dependency (but it will be exactly the same as the already QDP provided SDD). Since this *qual-tool* folder is a sub-set of the Qualification Toolchain used to generate the QDP, in case of doubts about the configuration files, see the Qualification Toolchain User Manual, [EDI20], which contains the complete explanation about the configuration. Additionally, the user might also want to change some manual parts of the document. An example could be to include the company logo (this user SVR is generated with no logos by default) or to complete the change records or authors table. In such a case, the *config_user.yml* and *.rst* files in *docs/rtems/djf/svr* should be updated accordingly and also, to include logos and update the document header/footer, the *docs/common/rtemssmp_user.sty* shall be updated (see as example the originals *config.yml* and *docs/common/rtemssmp_user.sty*, which are used to generate the QDP delivered SVR). Again, more details in the Qualification Toolchain User Manual (also available in *qual-tool* folder, under *docs/qt/SUM-303/delivery*).

As already referred, the *qual-tool* is a subset of the Qualification Toolchain used to produce the QDP. Hence, the user needs to install the same packages as referred in the Software Reuse File (SRF), [EDI19]:

- gcc

- python3-pip

- python3-sphinx

- texlive

- texlive-latex-extra

- texlive-fonts-extra

- pdftk

- doxygen

- python3-xlrd

- git

- pkg-config

- git-lfs

- clang-tools

- cppcheck

- python3-venv

- libpython2.7-dev

- latexmk

- flex

- bison

- texinfo

- graphviz

- libncurses5

- xz-utils

Note that above are listed the packages for Debian 10. If another Operating System is used, the user needs to install the same packages, but for the respective Operating System.

It may be the case that the Operating System of the user is not compatible with the *qual-tool*, for example, Ubuntu 18.04 (LTS version) is known that does not work with the tool. In that situation the user shall use a docker solution. To run the *qual-tool*, the user will need the same packages as for the Qualification Toolchain, hence the Dockerfile to use shall be as follows (to set up the Docker environment, please follow the instructions described in *Use QDP in a Docker solution*):

```
FROM debian:buster
RUN apt-get update
RUN apt-get -y install netcat net-tools wget lsb-release
RUN echo "Acquire::http::Pipeline-Depth 0;" >> /etc/apt/apt.conf.d/30proxy
RUN echo "Acquire::http::No-Cache true;" >> /etc/apt/apt.conf.d/30proxy
RUN echo "Acquire::BrokenProxy true;\n" >> /etc/apt/apt.conf.d/30proxy
RUN apt-get update
RUN apt-get -y install gcc python3-pip python3-sphinx texlive texlive-latex-extra␣
↪texlive-fonts-extra pdftk
RUN apt-get -y install doxygen python3-xlrd git pkg-config git-lfs clang-tools␣
↪cppcheck python3-venv libpython2.7-dev
RUN apt-get -y install latexmk
RUN apt-get -y install flex bison texinfo
RUN apt-get -y install graphviz libncurses5 pkg-config xz-utils
RUN chmod -R 777 /opt
```

After all this configuration is set, the tool is ready to be run, to re-execute the RTEMS testsuite and generate the user Software Verification Report (SVR). The following commands shall be executed:

```
$ cd /opt/rtems-6-sparc-gr740-smp-5/qual-tool/
```

```
/opt/rtems-6-sparc-gr740-smp-5/qual-tool$ make env
```

```
/opt/rtems-6-sparc-gr740-smp-5/qual-tool$ . env/bin/activate
```

```
/opt/rtems-6-sparc-gr740-smp-5/qual-tool$ ./qdp_config.py config-variants/sparc-
→gr740-smp-user-qual.yml
```

```
/opt/rtems-6-sparc-gr740-smp-5/qual-tool$ ./qdp_build.py --log-level=DEBUG build-
→sparc-gr740-smp-user-qual/ 2>&1 | tee log.txt
```

The first commands will install and activate the python3 virtual environment, the *qdp_config* command will configure the tool for the chosen configuration (*config-variants/sparc-gr740-smp-user-qual.yml*) and the *qdp_build* command will execute the tool and produce the SVR. Due to the coverage testsuite execution, this will take about 5 hours. During the tool execution, the following output will be produced:

- *build-sparc-gr740-smp-user-qual*: will contain sub-products of the tool execution (see Qualification Toolchain User Manual for more details).

- *log.txt*: will contain the logging of the execution

In addition to running the tests, re-generate the report and verifying that in the user hardware execution everything is OK, the QDP users will need to make the traceability of the project requirements to the RTEMS requirements.

11.3.2 Product Assurance activities

Taking into account the specifics for the RTEMS pre-qualification project, the EDISOFT *PA* team suggests the following activities.

Verify if there are changes in the original plans (record tracking). Verify if the changes are in accordance with the original plans and, if not, they are justified. The PA should look at the following documents, considered as plans for the RTEMS pre-qualification project:

- *ECSS* documentation (ECSS-E-ST-40C [ECS09a], ECSS-Q-ST-80C Rev. 1 [ECS17], ECSS-M-ST-40C Rev.1 [ECS09b]) – Always applicable for *ESA* projects

- Software Development Plan (SDP)

    – /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/sdp/SDP-000.pdf

    – /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/sdp/html/index.html

- Software Configuration Management Plan (SCMP)

    – /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/scmp/SCMP-001.pdf

    – /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/scmp/html/index.html

- Software Product Assurance Plan (SPAP)

  – /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spap/SPAP-002.pdf

  – /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spap/html/index.html

- QT-109 Technical Note RTEMS SMP Qualification Target

  – /opt/rtems-6-sparc-gr740-smp-5/doc/technical-notes/tn-qt/QT-109.pdf

  The technical note contains the tailoring of ECSS documents for the RTEMS pre-qualification project (chapter 4) and other relevant technical information that could be used by the PA:

  – Requirements, interfaces and test plans format (see chapter 6)

  – ECSS statement of compliance (see chapter 9)

Verify the affected parts of the QDP deliverables, according below:

- General verifications (transversal to all documentation):

  – Verify document correctness against ECSS and QT-109

  – Verify proper tracking update (change records)

  – Verify traceability (for example requirements to tests)

- Verifications specific to the SRS:

  – /opt/rtems-6-sparc-gr740-smp-5/doc/ts/srs/srs.pdf

  – /opt/rtems-6-sparc-gr740-smp-5/doc/ts/srs/html/index.html

  – Verification following the SDP, section 5.3 (for requirement specifications)

  – Verification following the SPAP, section 6.7.1

- Verifications specific to the ICD:

  – /opt/rtems-6-sparc-gr740-smp-5/doc/ts/icd/icd.pdf

  – /opt/rtems-6-sparc-gr740-smp-5/doc/ts/icd/html/index.html

  – Verification following SDP, section 5.3 (for interface specifications)

  – Verification following SPAP, section 6.7.1

- Verifications specific to the SDD:

  – /opt/rtems-6-sparc-gr740-smp-5/doc/ddf/sdd/html/index.html

  – Verification following SDP, section 5.3

  – Verification following SPAP, section 6.7.2

- Verifications specific to the SVS and SUITP:

  – /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svs/svs.pdf

  – /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svs/html/index.html

  – Verification following SDP, section 5.3 (for test specifications)

  – Verification following SDP, chapter 6 (only the parts applicable to RTEMS)

  – Verification following SPAP, section 6.7.4

- Verifications specific to the software product source code and test suite:

    - Verification following SPAP, section 6.7.3

- Verifications specific to the SVR - **the one generated for the own hardware, according with the previous section**:

    - /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svr/svr.pdf

    - /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svr/html/index.html

    - Verification following SDP, section 10 (only the parts applicable to RTEMS)

    - Verification following SPAP, section 6.7.3

    - Verification following SPAP, section 6.7.4

    - Verification of test results

- Verifications specific to the SRelD:

    - Verify if the delta closes any *SPR*.

- Verifications specific to the SPAMR:

    - /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spamr/spamr.pdf

    - /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spamr/html/index.html

    - Verification following SPAP, section 6.5

    - Verification following SPAP, section 5.5 (verification of software metrics)

    - Verification following SPAP, section 6.7.4.1

- Verifications regarding Independent Software Verification and Validation (ISVV):

    - Revisit and re-assess as needed the documentation listed within *ISVV Documentation Overview* regarding the ISVV performed as part of the RTEMS SMP pre-qualification, namely in what respects ISVV postponed findings that might have been addressed in the meanwhile (e.g. by the RTEMS community)

- Verifications specific to this document:

    - Follow the instructions and try to replicate the software utilization, specifically the PA should be able to decompress the QDP and correctly compile the RTEMS example application.

    - Assess the possible impacts of the listed *Possible Problems and Known Errors*

- Other verifications:

    - Verify if updates on Sphinx documentation are according with SDP, section 8

    - Perform the activities as described in SPAP sections 6.2.1 and 6.4

    - Verification of Certificate of Conformance, according with SPAP, section 6.7.4.2

    - Verification of audits (*FCV*/*PCV*), according with SCMP, section 5.8

CHAPTER

TWELVE

POSSIBLE PROBLEMS AND KNOWN ERRORS

Refer to Software Release Document (SRelD) for a more detailed list.

The following issues are currently open:

- QDP and Coverity Results contain personal information
- Convert Coding Style to Sphinx Document
- Add rtems_rate_monotonic_deadline()
- RTEMS Pre-Qualification (ECSS) for SMP
- Technical Specification (TS) for space profile
- Unit, integration and validation tests for space profile
- Add traceability information to the application configuration options
- Add build specification item verification
- SPDX Licenses and Header File Formatting
- Header Source files format "extern C" missing
- Two consecutive paragraphs
- Incorrect Formatting of "if", "while", or "for" Statements
- Incorrect Formatting of "!" Statements
- Unfinished TODO section
- Non-compliant Comments with Templates
- Discrepancy between Code and Comments
- Inaccurate comments at the beginning of file
- Incorrect Copyright License Information
- Function defined in header file
- Wrong data type
- Unclear #IF 1 Clause in File: Possible Error
- Section left to be fixed in kern_tc.c and tls.h (FIXME)
- Unclear comments

- Output value not assigned

- #pragmas to ignore warnings

- Unchecked boundaries

- Code for test software

- "Timer Server" Cannot be Used in Systems with Static Memory Allocation

- Deprecated Functions

- score/ & bsps/: Unused input parameters

- Bitwise operator applied to a signed operand

- LEON3FT - Power-down workaround

- Some functions are listed as unspecified on ICD

- libdl does not support TLS

- Deal with GR740 errata: Level-2 Cache Issues H1 2023 (GRLIB-TN-0021)

- Upgrade Version of Sphinx to 7

- Monitor GCC Bug 108658

- Building sphinx 7.0.1 fails (cloned)

- Document and test CONFIGURE_JFFS2_DELAYED_WRITE_TASK_PRIORITY

- New build configuration option RTEMS_QUALIFIED

12.1 Postponed ISVV Findings

There were Independent Software Verification and Validation (ISVV) Findings that although confirmed as such, were not yet addressed. The table below summarises the most relevant ones (for full details, please refer to the documentation listed within *ISVV Documentation Overview*).

While using the upcoming RTEMS SMP Qualification Data Package (QDP) release 5, one shall be aware of these know issues, and assess the risk they impose to the scope/mission/application on which the RTEMS SMP QDP is being used on.

Table 1: Most relevant postponed Findings

ISVV Finding [2]	Finding Title	Further Details
REQ-VER-004	Incomplete requirements	The requirements/specifications of "functional-type: action" have an incomplete text for these to really be understood as such and represent RTEMS implementation.

Table 1 – continued from previous page

ISVV Finding [2]	Finding Title	Further Details
REQ-VER-037	Missing interfaces	One needs to check if interfaces within spec:/rtems/type/if/group needs to be updated not to mention interfaces not described in the ICD, or if the missing interfaces need to be added to ICD.
DESIGN-VER-004	No SW User Manual (SUM) which follows ECSS clauses	Some typical SUM sections/information might be missing, which might be due to the generic nature of RTEMS. Further assessment is needed.
DESIGN-VER-008	Memory usage benchmarks	Within section 4.8 of SW Configuration File (SCF)/SUM, one needs to add units for the values, and further contextualising information, otherwise one can't interpret the values.
VAL-001	"Timer Server" not directly usable	For the current pre-qualified feature set of RTEMS, if one wants to use rtems_timer_initiate_server(), then it has to provide a custom stack allocator. This makes it inconvenient to use. One would need to provide a new directive to construct the timer server, e.g., rtems_timer_server_construct().
VAL-003	Some functions are listed as unspecified on ICD	One needs to improve the specification of some interfaces which are currently set to unspecified interface-type.
VAL-005	Missing NULL pointer check for clock manager directives	Although it might be considered as user responsibility, one could add logic that tests for NULL pointers within the clock manager directives, returning RTEMS_INVALID_ADDRESS where applicable.
TS-VER-001	No test cases descriptions	Through out all the SVS, several test cases do not have a description. A testcase description should be added for each test case.
QT-QTU-SIM-008	SPAMR - Fixed date and signature for variable data	There is variable data on a signed document with fixed date.
QT-QTU-SIM-015	sphinxbuilder.py - bsp names with underscore	If the BSP name contains underscores, then the document build process will fail because the underscore, due to its special meaning in latex. This is most noticeable on the document titles.
QT-QTU-SIM-016	ARM architecture's documents populated with SPARC specific text	When building an ARM QDP, the ICD, SRS and SDD documents are populated with SPARC specific text. This is most noticeable on the document titles.

continues on next page

Table 1 – continued from previous page

ISVV Finding [2]	Finding Title	Further Details
CODE-VER-010	TODO section left undone	#4810 - Unfinished TODO section
CODE-VER-011	FIXME	#4827 - Section left to be fixed in kern_tc.c and tls.h (FIXME)
CODE-VER-014[3]	Macro wrongly defined	#4842 - Registers definitions wrongly defined when there are reserved bits
CODE-VER-018	Deprecated Functions	#4852 - Deprecated Functions
CODE-VER-023	Code not in accordance with comments	#4812 - Discrepancy between Code and Comments
CODE-VER-026	Function defined in header file	#4815 - Function defined in header file
CODE-VER-028	Unused input parameter	#4862 - Unused input parameters
CODE-VER-029	Input validity unchecked	#4855 - Input validity unchecked
CODE-VER-030	Wrong data type	#4816 - Wrong data type
CODE-VER-032[Page 78, 3]	Output value not assigned	#4830 - Output value not assigned
CODE-VER-044	Code for test software	#4846 - Code for test software
CODE-VER-052	#pragmas to ignore warnings	#4831 - #pragmas to ignore warnings
CODE-VER-071	Confusing #IF 1 clause	#4819 - Unclear #IF 1 Clause in File: Possible Error
CODE-VER-073	Power-down workaround	#4875 - Power-down workaround
CODE-VER-078	Bitwise operator applied to a signed operand	#4864 - Bitwise operator applied to a signed operand

All of the Postponed ISVV Findings have already been fully discussed with ESA and the actions to address these are already fully defined within the Excel files where these are listed. For further details, one can refer to the documentation listed within *ISVV Documentation Overview*.

12.2 Waiver

The following requests for waivers were issued and accepted.

---

[3] Findings RTEMS-SMP-CODE-VER-014 and RTEMS-SMP-CODE-VER-032 are partially addressed. Please refer to section 3 of the RTEMS SMP ISVV Verification Report [F+23] for details.

Request for Waiver

Request for waiver as defined in ECSS-M-ST-40C Rev. 1 from 6 March 2009. See table J-1 on page 75 for a description of fields.

Organization: embedded brains GmbH	**Number:** RfW-QDP-01
Project: Qualification of RTEMS Symmetric Multiprocessing (SMP)	**Issue:** 1
Business agreement: 4000125572/18/NL/GLC/as	**Date:** 2021-Nov-17
Order:	**Item designation:** Directive rtems_rate_monotonic_deadline() as mentioned in the Space Profile
Originator site: embedded brains GmbH, Dornierstr. 4, 82178 Puchheim, Germany	**Affected item(s):** Pre-qualified RTEMS SMP operating system software
Classification: Minor	**Effectivity:** Affected versions: All

Affected document: Technical Note Space Profile, release 5, section 3.14
Short description: Function rtems_rate_monotonic_deadline() missing
Detailed description: The directive rtems_rate_monotonic_deadline() shall not be implemented in the pre-qualified RTEMS SMP operating system despite it is required by the Space Profile.
Reason for request: No such directive exists in the public RTEMS OSS project. It would be a completely new directive which has no documentation, specification, implementation, and tests yet. Moreover, it is not required for the use case of Jena-Optronik.
NCR: ESTEC GitLab Issue #664 <https://gitrepos.estec.esa.int/external/rtems-smp-qualification/-/issues/664>
NRB: 240900028-20211015-01.MOM ("QR#2 Minutes of Meeting")
Adverse effects: rtems_rate_monotonic_deadline() directive not available in the API.
Limitation of use: The directive was historically envisioned for a special use case which never surfaced in practice. Consequently, there is currently no known impact on usage.
Proposed disposition: The software should be used without the rtems_rate_monotonic_deadline() function (use-as-is).

embedded brains:	EDISOFT Approval: ☐ approved ☐ rejected		ESA Approval: ☐ approved ☐ rejected	
Submitter:	**Project Manager:**	**Product Assurance and Safety Manager:**	**Technical Officer:**	**Product Assurance and Safety Manager:**
Frank Kühndel **Date:** 2021-Dec-01	Nuno Ramos **Date:**	Rute Mateus **Date:**	Javier F. Salgado **Date:**	Cora Janse **Date:**

Request for Waiver

Request for waiver as defined in ECSS-M-ST-40C Rev. 1 from 6 March 2009. See table J-1 on page 75 for a description of fields.

Organization: embedded brains GmbH	**Number:** RfW-QDP-02
Project: Qualification of RTEMS Symmetric Multiprocessing (SMP)	**Issue:** 1
Business agreement: 4000125572/18/NL/GLC/as	**Date:** 2021-Nov-17
Order:	**Item designation:** Source code of pre-qualified RTEMS
Originator site: embedded brains GmbH, Dornierstr. 4, 82178 Puchheim, Germany	**Affected item(s):** Pre-qualified RTEMS SMP operating system software
Classification: Minor	**Effectivity:** Affected versions: All

Affected document:
ECSS-Q-HB-80-04A, 30 March 2011, annex A.3.3.11 till A.3.3.13
Software Product Assurance Plan, Release 7, Chapter 7

Short description:
Source code does not comply with cyclomatic number, nesting or lines of code

Detailed description:
From 383 files, 13 have too many lines of code. From 2090 functions, 16 do not meet the required cyclomatic number of this metric.

Reason for request:
The code is not newly written. The time tested software stems from the public RTEMS OSS project. The 16 functions have been visually inspected. Those originating from RTEMS are all false positives. Those originating from external libraries/FreeBSD (strchr.c, qsort.c, iovprintf.c, kern_tc.c) should not be changed to warrant compatibility with the original for updates and bug fixes.

NCR:
ESTEC GitLab Issue #641 <https://gitrepos.estec.esa.int/external/rtems-smp-qualification/-/issues/641>

NRB:
240900028-20211015-01.MOM ("QR#2 Minutes of Meeting")

Adverse effects:
No adverse effects. The code from external libraries/FreeBSD is not expected to be maintained by RTEMS and through its heavy use in other projects well tested.

Limitation of use:
There is no limitation of use.

Proposed disposition:
The pre-qualified RTEMS source code should be used as provided (use-as-is).

embedded brains:	**EDISOFT Approval:** ☐ approved ☐ rejected		**ESA Approval:** ☐ approved ☐ rejected	
Submitter: Frank Kühndel **Date:** 2021-Dec-01	**Project Manager:** Nuno Ramos **Date:**	**Product Assurance and Safety Manager:** Rute Mateus **Date:**	**Technical Officer:** Javier F. Salgado **Date:**	**Product Assurance and Safety Manager:** Cora Janse **Date:**

Request for Waiver

Request for waiver as defined in ECSS-M-ST-40C Rev. 1 from 6 March 2009. See table J-1 on page 75 for a description of fields.

Organization: embedded brains GmbH	**Number:** RfW-QDP-03
Project: Qualification of RTEMS Symmetric Multiprocessing (SMP)	**Issue:** 1
Business agreement: 4000125572/18/NL/GLC/as	**Date:** 2021-Nov-17
Order:	**Item designation:** Application Programming Interface (API) of pre-qualified RTEMS
Originator site: embedded brains GmbH, Dornierstr. 4, 82178 Puchheim, Germany	**Affected item(s):** Pre-qualified RTEMS SMP operating system software
Classification: Minor	**Effectivity:** Affected versions: All

Affected document: Technical Note Space Profile, release 5, chapters 3 and 4 Interface Control Document (ICD), release 1, chapter 5 Software Design Document (SDD), release 1
Short description: ICD contains interface specifications not in the Space Profile
Detailed description: The ICD and the SDD define and describe functions and `#define`s which are not part of the Space Profile. A few of them are explicitly excluded by Space Profile section 4.5.3. Furthermore, linker symbols for such functions appear in the pre-qualified library.
Reason for request: Some functions/managers were explicitly excluded in the hope to ease the realization of the project at the beginning. During the project it turned out that it was easier to include some of them than to exclude them. Yet, most functions where added because without them it would not be possible to reach 100% code coverage.
NCR: ESTEC GitLab Issue #662 <https://gitrepos.estec.esa.int/external/rtems-smp-qualification/-/issues/662> ESTEC GitLab Issue #665 <https://gitrepos.estec.esa.int/external/rtems-smp-qualification/-/issues/665>
NRB: 240900028-20211015-01.MOM ("QR#2 Minutes of Meeting")
Adverse effects: No adverse effects. The future user of pre-qualified RTEMS SMP will have more API functions available than initially planed. Functions not used by a programmer are not linked to the application. Hence the executable code size does not increase by these additional functions.
Limitation of use: There is no limitation of use. A few of these additional functions are not pre-qualified. Their descriptions in the ICD and SDD contain a "constraint" noting that they are not pre-qualified. Any attempt to use such a non-pre-qualified function from within an application build with the pre-qualified RTEMS library will result in a linker error.
Proposed disposition: The software should be used with these additional functions (use-as-is).

embedded brains:	EDISOFT Approval: ☐ approved ☐ rejected		ESA Approval: ☐ approved ☐ rejected	
Submitter:	**Project Manager:**	**Product Assurance and Safety Manager:**	**Technical Officer:**	**Product Assurance and Safety Manager:**
Frank Kühndel **Date:** 2021-Dec-01	Nuno Ramos **Date:**	Rute Mateus **Date:**	Javier F. Salgado **Date:**	Cora Janse **Date:**

BIBLIOGRAPHY

[But21]   Andrew   Butterfield.   *Formal   Verification   Report*.   2021.   URL: /opt/rtems-6-sparc-gr740-smp-5/doc/fm/fvr/FV3-202.pdf.

[BH21]   Andrew Butterfield and Mike Hinchey. *Formal Verification Plan*. 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/fm/fvp/FV1-200.pdf.

[BT23]   Andrew Butterfield and Frédéric Tuong. *Formal Verification Artefacts*. 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/fm/fva/FV2-201.pdf.

[con21a]   The RTEMS Project contributors. RTEMS CPU Architecture Supplement, Git Commit 9082f4d92c5da5f02b83a1e1203192094ea111fa. 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/cpu-supplement/cpu-supplement.pdf.

[con21b]   The RTEMS Project contributors. *RTEMS CPU Architecture Supplement, Git Commit 9082f4d92c5da5f02b83a1e1203192094ea111fa*. embedded brains GmbH, Dornier-straße 4, 82178 Puchheim, Germany, 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/cpu-supplement/cpu-supplement.pdf.

[con21c]   The RTEMS Project contributors. *RTEMS Classic API Guide, Git Commit 9082f4d92c5da5f02b83a1e1203192094ea111fa*. embedded brains GmbH, Dornier-straße 4, 82178 Puchheim, Germany, 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/c-user/c-user.pdf.

[con21d]   The RTEMS Project contributors. *RTEMS Software Engineering, Git Commit 9082f4d92c5da5f02b83a1e1203192094ea111fa*. embedded brains GmbH, Dornier-straße 4, 82178 Puchheim, Germany, 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/eng/eng.pdf.

[con21e]   The RTEMS Project contributors. *RTEMS User Manual, Git Commit 9082f4d92c5da5f02b83a1e1203192094ea111fa*. embedded brains GmbH, Dornier-straße 4, 82178 Puchheim, Germany, 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/rtems/user/user.pdf.

[con23]   The RTEMS Project contributors. *RTEMS Qualification, Software Design Document [sparc/gr740/smp/5], Git Commit 89499c4002fb5f8b0e1cfd7c1edac2e78076c835*. embedded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/ddf/sdd/html/index.html.

[Dre13]   Ulrich Drepper. *ELF Handling For Thread-Local Storage*. 2013. URL: http://www.akkadia.org/drepper/tls.pdf.

[ECS09a] ECSS. *ECSS-E-ST-40C - Software general requirements*. European Cooperation for Space Standardization, 2009. URL: https://ecss.nl/standard/ecss-e-st-40c-software-general-requirements/.

[ECS09b] ECSS. *ECSS-M-ST-40C Rev.1 - Configuration and information management*. European Cooperation for Space Standardization, 2009. URL: https://ecss.nl/standard/ecss-m-st-40c-rev-1-configuration-and-information-management/.

[ECS17] ECSS. *ECSS-Q-ST-80C Rev.1 - Software product assurance*. European Cooperation for Space Standardization, 2017. URL: https://ecss.nl/standard/ecss-q-st-80c-rev-1-software-product-assurance-15-february-2017/.

[EDI19] EDISOFT. *Software Reuse File*. 2019.

[EDI20] EDISOFT. *Qualification Toolchain Software User Manual*. 2020.

[F+23] Xavier Ferreira and others. *RTEMS SMP Independent Verification Report*. Critical Software, 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/isvv/iver/CSW-RTEMSISVV-2021-RPT-04960_7-rtems-smp-isvv-ive-report.pdf.

[HK+23a] Sebastian Huber, Frank Kühndel, and others. *RTEMS Qualification, Interface Control Document [sparc/gr740/smp/5], Release 3*. embedded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/ts/icd/icd.pdf.

[HK+23b] Sebastian Huber, Frank Kühndel, and others. *RTEMS Qualification, Software Requirement Specification [sparc/gr740/smp/5], Release 4*. embedded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/ts/srs/srs.pdf.

[HK+23c] Sebastian Huber, Frank Kühndel, and others. *RTEMS Qualification, Software Unit and Integration Test Plan [sparc/gr740/smp/5], Release 3*. embedded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/djf/suitp/suitp.pdf.

[HK+23d] Sebastian Huber, Frank Kühndel, and others. *RTEMS Qualification, Software Validation Specification [sparc/gr740/smp/5], Release 3*. embedded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svs/svs.pdf.

[HK+23e] Sebastian Huber, Frank Kühndel, and others. *RTEMS Qualification, Software Verification Report [sparc/gr740/smp/5], Release 7*. embedded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/djf/svr/svr.pdf.

[H+23] Sebastian Huber and others. *RTEMS Qualification, Software Release Document [sparc/gr740/smp/5], Release 4*. embedded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/ddf/sreld/sreld.pdf.

[HV21] Sebastian Huber and José Valdez. *Technical Note: Space Profile, Release 6*. embedded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, Jun 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/technical-notes/tn-sp/tn-space-profile.pdf.

[HVM+21] Sebastian Huber, José Valdez, Cláudio Maia, Ting Peng, Joel Pinto, and others. *QT-109 Technical Note: RTEMS SMP Qualification Target, Release 6*. embed-

ded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, Jun 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/technical-notes/tn-qt/QT-109.pdf.

[KGV+23] Frank Kühndel, Matthias Göbel, José Valdez, and others. *RTEMS Qualification, Software Product Assurance Milestone Report [sparc/gr740/smp/5], Release 7*. embedded brains GmbH, Dornierstraße 4, 82178 Puchheim, Germany, 2023. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spamr/spamr.pdf.

[Mat21]   Andre Ribeiro; Rute Mateus. *SPAP-002 Software Product Assurance Plan, Release 7*. EDISOFT, Rua Calvet Magalhães 245, 2770-153 Paço de Arcos, Portugal, Jun 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/paf/spap/SPAP-002.pdf.

[Ram21]   Nuno Ramos. *Software Review Plan, Release 9*. EDISOFT, Rua Calvet Magalhães 245, 2770-153 Paço de Arcos, Portugal, Dec 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/srevp/SRevP-018.pdf.

[Sil21]   Helder Silva. *SCMP-001 RTEMS Software Configuration Management Plan, Release 7*. EDISOFT, Rua Calvet Magalhães 245, 2770-153 Paço de Arcos, Portugal, Oct 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/scmp/SCMP-001.pdf.

[S+21]   Helder Silva and others. *SDP-000 RTEMS Software Development Plan, Release 8*. EDISOFT, Rua Calvet Magalhães 245, 2770-153 Paço de Arcos, Portugal, Oct 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/mgt/sdp/SDP-000.pdf.

[VP21]   José Valdez and Sofia Pacheco. *TI-003 Tools Identification, Release 6*. EDISOFT, Rua Calvet Magalhães 245, 2770-153 Paço de Arcos, Portugal, Jan 2021. URL: /opt/rtems-6-sparc-gr740-smp-5/doc/technical-notes/tn-ti/TI-003.pdf.