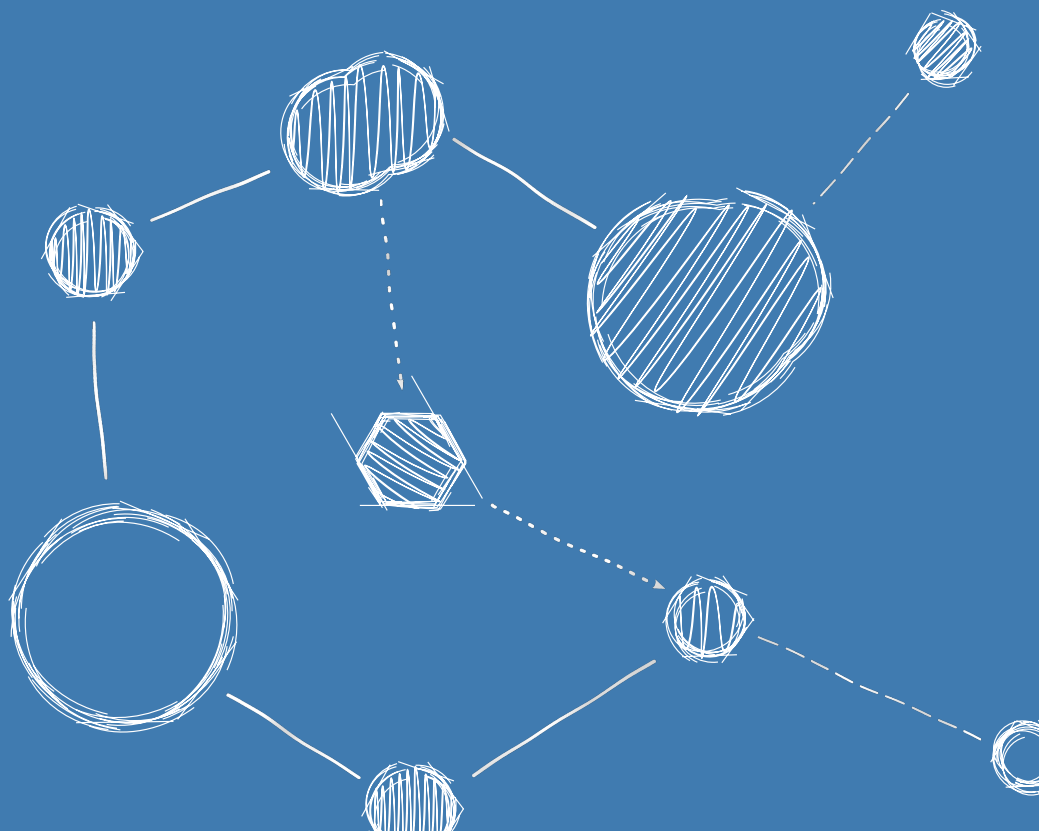


LEAN OS – AN OPERATING SYSTEM FOR THE SSDP

Software Requirements Specification



Software Requirements Specification

Reference: LEANOS-UVIE-SRS-001

Version: Issue 0.2, May 2, 2016

Prepared by: Armin Luntzer¹

Checked by: Roland Ottensamer¹

Approved by: Franz Kerschbaum¹

¹ Department of Astrophysics, University of Vienna

Copyright ©2016

Permission is granted to copy, distribute and/or modify this document under the terms of the GNU Free Documentation License, Version 1.3 or any later version published by the Free Software Foundation; with no Front-Cover, no Logos of the University of Vienna.

Contents

1	Introduction	6
1.1	Purpose of the Document	6
1.2	Scope of the Software	6
2	Applicable and Reference Documents	7
3	Terms, Definitions and Abbreviated Items	8
3.1	Acronyms	8
3.2	Glossary	9
4	Software Overview	13
4.1	Function and Purpose	13
4.2	Environmental considerations	13
4.3	Relation to other systems	13
4.4	Constraints	13
5	Requirements	15
5.1	General	15
5.2	Functional Requirements	16
5.3	Performance Requirements	20
5.4	Interface Requirements	21
5.5	Operational Requirements	21
5.6	Resources Requirements	21
5.7	Design Requirements and Implementation Constraints	22
5.8	Security and Privacy Requirements	22
5.9	Software Quality Requirements	23
5.10	Software Reliability Requirements	23
5.11	Software Maintainability Requirements	23
5.12	Software Safety Requirements	23
5.13	Software Configuration and Delivery Requirements	24
5.14	Data Definition and Database Requirements	24
5.15	Human Factors Related Requirements	24
5.16	Adaptation and Installation Requirements	24
6	Validation Requirements	25
7	Traceability	26
8	Logical Model Description	27

List of Requirements

R-GEN-0001	15
R-GEN-0002	15
R-GEN-0003	15
R-GEN-0004	15
R-GEN-0006	16
R-GEN-0601	16
R-GEN-0602	16
R-FUN-0007	16
R-FUN-0803	16
R-FUN-0008	17
R-FUN-0011	17
R-FUN-0012	17
R-FUN-0013	17
R-FUN-0014	17
G-FUN-0015	17
R-FUN-0016	18
R-FUN-0017	18
R-FUN-0019	18
R-FUN-0020	18
R-FUN-0021	18
R-FUN-0022	18
R-FUN-0804	19
R-FUN-0023	19
R-FUN-0024	19
G-FUN-0025	19
R-FUN-0026	19
R-FUN-0027	20
R-GEN-0009	20
R-GEN-0018	20
R-GEN-0010	20
R-GEN-0028	21
R-GEN-2001	21
R-GEN-0101	21
R-GEN-0801	22
R-GEN-0802	22
R-GEN-0805	22
G-GEN-0201	23
R-GEN-0301	23

R-GEN-0401	23
R-GEN-0402	23
R-GEN-0403	24
R-GEN-0005	24
R-GEN-1001	25
R-GEN-1002	25

Revision History

Revision	Date	Author(s)	Description
0.0	16.07.2015	AL	draft requirements created based on NGAPP
0.1	16.03.2016	AL	initial version with specifications from MPPBv2
0.2	02.05.2016	AL	revised after internal design review

1. Introduction

1.1 Purpose of the Document

This document specifies the software requirements for the operating system kernel LeanOS. LeanOS targets the [Scalable Sensor Data Processor \(SSDP\)](#), and to a lesser extent, its compatible predecessor, the [MPPB v2.x](#) [2]. It is intended to be used in unmanned space applications of at least Software Criticality Level C. Readers must be familiar with the basic concepts of event driven real time operating systems and the target hardware.

This document follows the document structure for software requirement specifications found in Annex D of ECSS-E-ST-40C [1].

1.2 Scope of the Software

LeanOS is a lightweight operating system targeting the particular characteristics of the [SSDP](#) and is focused on driving the [Network On Chip \(NoC\)](#) and its attached [Xentium Digital Signal Processor \(DSP\)](#) cores.

2. Applicable and Reference Documents

- [1] *ECSS-E-ST-40C Space engineering - Software*. ESA Requirements and Standards Division, 2009.
- [2] *Massively Parallel Processor Breadboarding Datasheet*. 2016.

3. Terms, Definitions and Abbreviated Items

3.1 Acronyms

ADC	Analog to Digital Converter 8, 10, <i>Glossary: Analog to Digital Converter</i>
API	Application Programming Interface 8, 10, 15, <i>Glossary: Application Programming Interface</i>
BSP	Board Support Package 8, <i>Glossary: Board Support Package</i>
CPU	Central Processing Unit 8, 10–12, 18, 26, <i>Glossary: Central Processing Unit</i>
DAC	Digital to Analog Converter 8, 10, <i>Glossary: Digital to Analog Converter</i>
DMA	Direct Memory Access 8, 14, 26, <i>Glossary: Direct Memory Access</i>
DSP	Digital Signal Processor 6, 8, 11–14, <i>Glossary: Digital Signal Processor</i>
ELF	Executable and Linkable Format 8, <i>Glossary: Executable and Linkable Format</i>
FIFO	First In - First Out 8, <i>Glossary: First In - First Out</i>
FPU	Floating Point Unit 8, 20, <i>Glossary: Floating Point Unit</i>
GCC	GNU Compiler Collection 8, <i>Glossary: GNU Compiler Collection</i>
ILP	Instruction Level Parallelism 8, 13, <i>Glossary: Instruction Level Parallelism</i>
ISR	Interrupt Service Routine 8, 20, 21, <i>Glossary: Interrupt Service Routine</i>
MMU	Memory Management Unit 8, <i>Glossary: Memory Management Unit</i>
MPPB	Massively Parallel Processor Breadboarding system 5, 6, 8, 11, 14, 21, <i>Glossary: Massively Parallel Processor Breadboarding system</i>
NGAPP	Next Generation Astronomy Processing Platform 5, 8, 14, <i>Glossary: Next Generation Astronomy Processing Platform</i>
NoC	Network On Chip 6, 8, 11, 12, 26, <i>Glossary: Network On Chip</i>
POSIX	Portable Operating System Interface 8, 15, <i>Glossary: Portable Operating System Interface</i>
PUS	Packet Utilisation Standard 8, <i>Glossary: Packet Utilisation Standard</i>
RAM	Random-Access Memory 8, 10, <i>Glossary: Random-Access Memory</i>
RISC	Reduced Instruction Set Computing 8, 12, <i>Glossary: Reduced Instruction Set Computing</i>
RMAP	Remote Memory Access Protocol 8, <i>Glossary: Remote Memory Access Protocol</i>
RR	Round Robin 8, <i>Glossary: Round Robin</i>
RSA	RUAG Space Austria 14

- SMP** Symmetric Multiprocessing 8, 11, 14, 18, *Glossary: [Symmetric Multiprocessing](#)*
- SoC** System On Chip 9, 11, *Glossary: [System On Chip](#)*
- SSDP** Scalable Sensor Data Processor 6, 9, 14, 21, 26, *Glossary: [Scalable Sensor Data Processor](#)*
- UVIE** University of Vienna 14
- VLIW** Very Long Instruction Word 9, 13, *Glossary: [Very Long Instruction Word](#)*

3.2 Glossary

Analog to Digital Converter

An Analog to Digital Converter is a system that converts an analog signal into a quantized digital signal. Its counterpart is the [Digital to Analog Converter \(DAC\)](#).

Application Programming Interface

The Application Programming Interface defines how a developer can write a program that requests services from an operating system or application. [Application Programming Interfaces \(APIs\)](#) are implemented by function calls composed of verbs and nouns, i.e. a function to execute on an object.

Board Support Package

A Board Support Package is the implementation of a specific interface defined by the abstract layer of an operating system that enables the latter to run on the particular hardware platform.

Central Processing Unit

The Central Processing Unit is the electronic circuitry that interprets instructions of a computer program and performs control logic, arithmetic, and input/output operations specified by the instructions. It maintains high-level control of peripheral components, such as memory and other devices.

Digital Signal Processor

A Digital Signal Processor is a specialised processor with its architecture targeting the operational needs of digital signal processing.

Digital to Analog Converter

A Digital to Analog Converter is a system that converts a quantized digital signal into an analog signal. Its counterpart is the [Analog to Digital Converter \(ADC\)](#).

Direct Memory Access

Direct Memory Access is a feature of a computer system that allows hardware subsystems to access main system [Random-Access Memory](#) directly, thereby bypassing the [Central Processing Unit](#).

Executable and Linkable Format

The Executable and Linkable Format is a common standard file format for executables, object code, shared libraries, and core dumps.

First In - First Out

In FIFO processing, the "head" element of a queue is processed first. Once complete, the element is removed and the next element in line becomes the new queue head.

Floating Point Unit

A co-processor unit that specialises in floating-point calculations.

GNU Compiler Collection

The GNU Compiler Collection is a compiler system produced by the GNU project. It is part of the GNU toolchain collection of programming tools.

Instruction Level Parallelism

Instruction-level parallelism (ILP) is a measure of how many instructions in a computer program can be executed simultaneously by the [Central Processing Unit \(CPU\)](#).

Interrupt Service Routine

An Interrupt Service Routine is a function that handles the actions needed to service an interrupt.

LEON2

The LEON2 is a synthesisable VHDL model of a 32-bit processor compliant with the SPARC V8 architecture. It is highly configurable and particularly suitable for [System On Chip \(SoC\)](#) designs. Its source code is available under the GNU LGPL license

LEON3

The LEON3 is an updated version of the [LEON2](#), changes include [Symmetric Multiprocessing](#) support and a deeper instruction pipeline

LEON3-FT

The LEON3-FT is a fault-tolerant version of the [LEON3](#). Changes to the base version include autonomous error handling, cache locking and different cache replacement strategies.

Massively Parallel Processor Breadboarding system

The Massively Parallel Processor Breadboarding system is a proof-of-concept design for a space-hardened, fault-tolerant multi-DSP system with various subsystems to build a powerful digital signal processing system with a high data throughput. Its distinguishing features are the [Network On Chip](#) and the [Xentium DSPs](#) controlled by a [LEON2](#) processor. It was developed under ESA contract 21986 by Recore Systems B.V.

Memory Management Unit

A Memory Management Unit performs address space translation between physical and virtual memory pages and protects unprivileged access to certain memory regions.

Network On Chip

A Network On Chip is a communication system on an integrated circuit that applies (packet based) networking to on-chip communication. It offers improvements over more conventional bus interconnects and is more scalable and power efficient in complex [System On Chip](#) designs.

Next Generation Astronomy Processing Platform

Next Generation Astronomy Processing Platform was an evaluation of the [MPPB](#) performed in a joint effort of RUAG Space Austria and the Department of Astrophysics of the University of Vienna. The project was funded under ESA contract 40000107815/13/NL/EL/f.

Packet Utilisation Standard

The Packet Utilisation Standard addresses the end-to-end transport of telemetry and telecommand data between user applications on the ground and applications onboard a satellite. See also ECSS-E-70-41A.

Portable Operating System Interface

The Portable Operating System Interface is a family of standards specified by the IEEE Computer Society for maintaining compatibility between operating systems.

Random-Access Memory

Random-Access Memory is a type of memory where each memory cell may be accessed directly via their memory addresses.

Reduced Instruction Set Computing

RISC is a [CPU](#) design strategy that intends to improve performance by combining a simplified instruction set with a microprocessor architecture that is capable of executing an instruction in a smaller number of clock cycles.

Remote Memory Access Protocol

The Remote Memory Access Protocol is a form of [SpaceWire](#) communication that transparently communicates writes to memory mapped regions between different hardware devices.

Round Robin

Round Robin is a scheduling algorithm where time slices are assigned in equal portions and in circular order. In the context of threads, priorities are usually only used to control re-scheduling order when a mutex is accessed by a thread.

Scalable Sensor Data Processor

The Scalable Sensor Data Processor (SSDP) is a next generation on-board data processing mixed-signal ASIC, envisaged to be used in future scientific payloads requiring high-performance on-board processing capabilities. It is built upon a heterogeneous multicore architecture, combining two [Xentium DSP](#) cores with a general-purpose [LEON3-FT](#) control processor in a [Network On Chip](#).

SpaceWire

SpaceWire is a spacecraft communication network based in part on the IEEE 1355 standard of communications.

SPARC

SPARC ("scalable processor architecture") is a [Reduced Instruction Set Computing](#) instruction set architecture developed by Sun Microsystems in the 1980s. The distinct feature of SPARC processors is the high number of [Central Processing Unit](#) registers that are accessed similarly to stack variables via "sliding windows".

Symmetric Multiprocessing

Symmetric Multiprocessing denotes computer architectures, where two or more identical processors are connected to the same periphery and are controlled by the same operating system instance.

System On Chip

A System On Chip is an integrated circuit that combines all components of a computer or other electronic system into a single chip.

Very Long Instruction Word

Very Long Instruction Word is a processor architecture design concept that exploits [Instruction Level Parallelism](#). This approach allows higher performance at a smaller silicon footprint compared to serialised instruction processors, as no instruction re-ordering logic to exploit superscalar capabilities of the processor must be integrated on the chip, but requires either code to be tuned manually or a very sophisticated compiler to exploit the full potential of the processor.

Xentium

The Xentium is a high performance [Very Long Instruction Word DSP](#) core. It operates 10 parallel execution slots supporting 32/40 bit scalar and two 16-bit element vector operations.

4. Software Overview

4.1 Function and Purpose

In the course of the [NGAPP](#) activities, an evaluation of the [MPPB](#) was performed in a joint effort of [RUAG Space Austria \(RSA\)](#) and the Department of Astrophysics of the [University of Vienna \(UVIE\)](#). While the original intent of the work of [UVIE](#) was to quantify the performance of the [Xentium DSPs](#) and the [MPPB](#) as a whole with regard to on-board data treatment and reduction in an astronomical mission setting, it was found that, given the highly innovative nature of this new processing platform, a novel approach was needed concerning the management of system resources, [Direct Memory Access \(DMA\)](#) mechanics and [DSP](#) program design for best efficiency and turnover rates. Consequently, [UVIE](#) developed an experimental operating system to stably drive the [DSP](#) cores and the [MPPB](#) close to its performance limit. LeanOS is a development based on this operating system concept. Along with its intended functionality, it provides full software test coverage, example applications and good documentation that supports all aspects of the software.

4.2 Environmental considerations

LeanOS is only required to support the [SSDP](#), but will also support the [MPPB](#) v2.x. Ultimately, this does allow reuse of the operating system core for [LEON2](#) and [LEON3](#) based platforms with minor adaptations. Even though the [SSDP](#) does not contain multi-core [LEON3](#) processor, [Sym-metric Multiprocessing \(SMP\)](#) support will be prepared in LeanOS, as this will likely change with newer versions of the [SSDP](#). LeanOS is released under an open source license, so compatible development tools should be available under such licenses as well. Note that it might still be necessary to use a commercial product to target particular configurations or proprietary hardware functionality.

4.3 Relation to other systems

LeanOS is a stand-alone software product.

4.4 Constraints

LeanOS primarily targets the [SSDP](#), which effectively limits the hardware support the the characteristics of this platform.

The source code is written in C and if required, [SPARC](#) v8 compatible assembly. This is necessary as hardware interaction of a machine level requires programming languages designed for that purpose.

5. Requirements

5.1 General

R-GEN-0001	Short Text	Software Requirement
	No external dependencies	LeanOS shall not make use of external libraries, including C-Runtime libraries.

R-GEN-0002	Short Text	Software Requirement
	Custom libc functionality	LeanOS shall, as part of its code-base and as needed, provide its own implementations of typical C-library functions with an API conforming to their POSIX definitions.

Comment

The collection of these functions can also form the base or a subset of a C-Library to be used by applications.

R-GEN-0003	Short Text	Software Requirement
	Application libc usage	LeanOS shall optionally call a function that initialises a C-Library on startup. It is up to the user to define and link the implementation of this function and the according C library.

Comment

The function `__libc_start_main()` is declared as a weak symbol and tested before execution.

R-GEN-0004	Short Text	Software Requirement
	Fatal Error Management	If a non-recoverable error state is detected in LeanOS, it shall optionally call an error handling routine provided by the user before issuing a reboot. LeanOS shall provide a directive to register such a function.

Comment

This gives the user the possibility to perform a clean shutdown of critical tasks in their particular environment.

R-GEN-0006	Short Text	Software Requirement
	Core runtime	In its basic configuration, LeanOS shall restrict itself to the initialisation of its core services on a single processor, thereby configuring traps, memories and timers. All other services or device drivers shall be configured and added via the build system.

R-GEN-0601	Short Text	Software Requirement
	Error Logging	LeanOS shall maintain an error message log that is readable by the user.

R-GEN-0602	Short Text	Software Requirement
	Coding Style	LeanOS code shall use the Linux kernel coding style. Source files and patches shall be checked using the check-patch.pl utility found in the Linux kernel source tree.

Comment

A major point of LeanOS is that it does not only come with an open source license, but should ideally only use tools that are distributable under a similar license. The Linux kernel is used in billions of devices world-wide, its coding style is hence arguably well-suited for use in successful software.

5.2 Functional Requirements

R-FUN-0007	Short Text	Software Requirement
	SMP support	LeanOS shall be able to run on a multi-core configuration of a LEON3 processor.

R-FUN-0803	Short Text	Software Requirement
	MMU support	LeanOS shall support the use of the MMU of the LEON3 processor.

R-FUN-0008	Short Text	Software Requirement
	Supervisor Mode	The LeanOS kernel shall execute with the SPARC supervisor mode enabled. Application code shall run with supervisor mode disabled.
R-FUN-0011	Short Text	Software Requirement
	Timing	LeanOS shall provide access to typical time keeping, time taking and delay timing functionality expected by an operating system.
R-FUN-0012	Short Text	Software Requirement
	Task Support	LeanOS shall provide means to create new tasks.
R-FUN-0013	Short Text	Software Requirement
	Task priorities and deadlines	LeanOS shall support the assignment of a priority and a deadline to a task.
R-FUN-0014	Short Text	Software Requirement
	Task scheduling	LeanOS shall offer a fixed priority scheduler with priority inversion handling.
G-FUN-0015	Short Text	Software Requirement
	Other schedulers	LeanOS should offer an Earliest Deadline First scheduler supporting priority execution in overload conditions.
Comment	Along with dynamic ticking, this is an option to optimise thread CPU utilisation with the added benefit of predictable execution for certain high-priority threads in conditions where the total load unexpectedly would exceed 100%.	

R-FUN-0016	Short Text	Software Requirement
	Kernel Ticks	LeanOS shall operate in tickless (non-periodic) timed wakeup mode by default.

Comment

Tickless timing avoids unnecessary wake-ups of the CPU if no task is running and improves performance by only switching to kernel mode from regular tasks when absolutely necessary.

R-FUN-0017	Short Text	Software Requirement
	Tickless Timing	LeanOS timing functionality shall be able to operate in tickless mode, where a queue of wakeup times is maintained and a hardware timer is used in such a way that its next underflow (resulting in an interrupt) is programmed to coincide with the next wakeup time. New wakeup times shall be inserted into the queue as needed to maintain the desired timeline of events and the hardware timer be readjusted accordingly.

R-FUN-0019	Short Text	Software Requirement
	Semaphores and Mutexes	LeanOS shall provide semaphores and mutexes as part of its task functionality. Task priorities shall be protected by the priority ceiling protocol.

R-FUN-0020	Short Text	Software Requirement
	Tasks and SMP	LeanOS shall support task migration between CPUs and track and ensure atomicity of related functions (e.g. mutexes) across multiple processors.

R-FUN-0021	Short Text	Software Requirement
	Message Queues	LeanOS shall support message queues for inter-process communication. Atomicity of queues shall be ensured across multiple processors.

R-FUN-0022	Short Text	Software Requirement
	Kernel Modules	LeanOS shall offer loadable module support infrastructure.

R-FUN-0804	Short Text	Software Requirement
	Kernel- Userspace Initialisation	LeanOS shall offer a configurable initialisation point that executes a user-space setup procedure.
Comment	This is the point where the application software is loaded.	

R-FUN-0023	Short Text	Software Requirement
	System Control Interface	LeanOS shall offer a way for drivers or other functional modules to export configuration or state variables into an logical tree maintained by the operating system. This logical tree shall be accessible by other modules and the user as a character-based interface.
Comment	This is supposed to be a centrally organised, generic parseable interface to get or set configuration states or system statistics.	

R-FUN-0024	Short Text	Software Requirement
	Binary System Control Interface	LeanOS shall offer the possibility to install binary exchange nodes within the System Control Interface tree that may be used for larger amounts of binary data. The binary format shall be defined by the creator of the node. Any users of the node shall be responsible to read or write in the correct format expected by the node.
Comment	Sometimes, a binary dump to or from a subsystem is needed that is only parseable with special knowledge. This could, for example, be an internal memory dump or some calibration data.	

G-FUN-0025	Short Text	Software Requirement
	Device Drivers	LeanOS should come with device drivers for all hardware functionality of the SSDP.

R-FUN-0026	Short Text	Software Requirement
	Xentium Scheduler	LeanOS shall offer a way to define and load Xentium data processing code.

R-FUN-0027	Short Text	Software Requirement
	Xentium Data Buffers	LeanOS shall provide packet-driven meta-data buffers to Xentium programs that can link to arbitrary data sets and routing tables that define the path of a data packet through a series of Xentium program nodes.
Comment	Datagrams propagate through processing nodes as defined in their routing table, which represents their individual "processing chain".	

5.3 Performance Requirements

R-GEN-0009	Short Text	Software Requirement
	Traps/Interrupt:	LeanOS trap entry and exit code shall not exceed 500 instructions.
Comment	This does not include the callback functions for a trap or interrupt service routine.	

R-GEN-0018	Short Text	Software Requirement
	Deferred saving of FPU registers	In trap mode, LeanOS shall defer saving of FPU registers until it is accessed.
Comment	This approach saves many clock cycles, as the FPU is not typically used as part of an ISR.	

R-GEN-0010	Short Text	Software Requirement
	Interrupt downtime	ISRs that execute longer than 50 μ s shall be set to be executed in deferred mode at a later time if feasible given the type and rate of an ISR.
Comment	Interrupt mode should be left as fast as possible, so regular processing can continue. Most ISRs requiring long processing times perform actions on data, which can typically be moved into a dedicated thread, with the ISR acting as a signalling function.	

R-GEN-0028	Short Text	Software Requirement
	Real-Time Thread Support	LeanOS shall offer support for a class of real time threads that may also preempt the operating system with the exception of <i>ISRs</i> .
Comment	Some application real time tasks may be so timing sensitive, that even operating system code must be preemptible to guarantee a timely response.	

5.4 Interface Requirements

R-GEN-2001	Short Text	Software Requirement
	Interface Documentation	The internal and external interfaces of LeanOS shall be described as part of its source code in Doxygen markup.
Comment	The interfaces of a complex piece of software such as an operating system often change over time, as it is adapted to new circumstances or improved implementation of particular functionality or just subtle changes that may not always propagate into other documents as they should. It is therefore better to maintain interface documentation together with the code it describes, where it will be more likely to be updated on the spot.	

5.5 Operational Requirements

No operational requirements have been identified.

5.6 Resources Requirements

R-GEN-0101	Short Text	Software Requirement
	Target Platform	LeanOS shall execute on the <i>SSDP</i> , in particular the <i>LEON3-FT</i> processor of the platform. It should also execute on the <i>MPPB v2.x</i> .

5.7 Design Requirements and Implementation Constraints

R-GEN-0801	Short Text	Software Requirement
	Modular Design	All components of LeanOS shall be written such that a particular component does contain its intended functionality as much as possible.

Comment

If something is mostly self-contained, it is easier to modify and re-use in another software project.

R-GEN-0802	Short Text	Software Requirement
	Software Hierarchy	All components shall make use and rely only on functionality that is lower in hierarchy. The use of functionality that is hierarchially equal or higher shall be explicitly forbidden.

Comment

Note that hierarchy refers to the abstraction level of a component. An [ISR](#) is of higher level than the interrupt dispatcher. Even though it is called by the latter, the actual registration of the [ISR](#) is done by a higher level component. Such constructs are legal, the reliance on user-space provided functionality, e.g. an error reporting function, which sends messages via some packet interface on the other hand, is not.

R-GEN-0805	Short Text	Software Requirement
	Programming Language	The programming language shall be C. SPARCV8 compatible assembly shall be used when necessitated by performance or timing constraints and interface requirements.

5.8 Security and Privacy Requirements

No security or privacy requirements have been identified.

5.9 Software Quality Requirements

G-GEN-0201	Short Text	Software Requirement
	Product Metrics	LeanOS should have a cyclomatic complexity of at most 15 and a nesting level of at most 6 per function. Each function shall have a single exit point for the nominal case.

5.10 Software Reliability Requirements

No software reliability requirements have been identified.

5.11 Software Maintainability Requirements

R-GEN-0301	Short Text	Software Requirement
	Version Identification	It shall be possible to identify the version of compiled binary software components by reading their identifier from a special memory segment.

Comment

The build identifier or version number should be set and defined when building the binary.

5.12 Software Safety Requirements

R-GEN-0401	Short Text	Software Requirement
	Stack Pointer Checks	The stack pointer of a task shall be checked for feasibility before scheduling the latter.

R-GEN-0402	Short Text	Software Requirement
	Correctable Traps	LeanOS shall provide handlers for correctable traps caused by kernel or user code and either correctly execute the desired operation (e.g. unaligned access) or replace the result with a default value (e.g. division by zero) and skip the offending code instruction to continue.

R-GEN-0403	Short Text	Software Requirement
	Trap Error Reporting	LeanOS shall make an entry into its error message log when a trap event occurs, describing the nature and source of the trap.

5.13 Software Configuration and Delivery Requirements

R-GEN-0005	Short Text	Software Requirement
	Build configuration	LeanOS shall make use of the Linux Kernel Build System (kbuild) for its configuration.

5.14 Data Definition and Database Requirements

No data definition or database requirements have been identified.

5.15 Human Factors Related Requirements

No human factors related requirements have been identified.

5.16 Adaptation and Installation Requirements

No adaptation and installation requirements have been identified.

6. Validation Requirements

R-GEN-1001	Short Text	Software Requirement
	Verification Method	The verification method and verification activity shall be specified in a Software Qualification Test Plan for each requirement.

R-GEN-1002	Short Text	Software Requirement
	Qualification Testing	SW qualification testing shall be performed with various configurations of LeanOS.

7. Traceability

The requirements in this document are both user and system requirements. Traces from design and test to the software requirements are given in the respective documents.

8. Logical Model Description

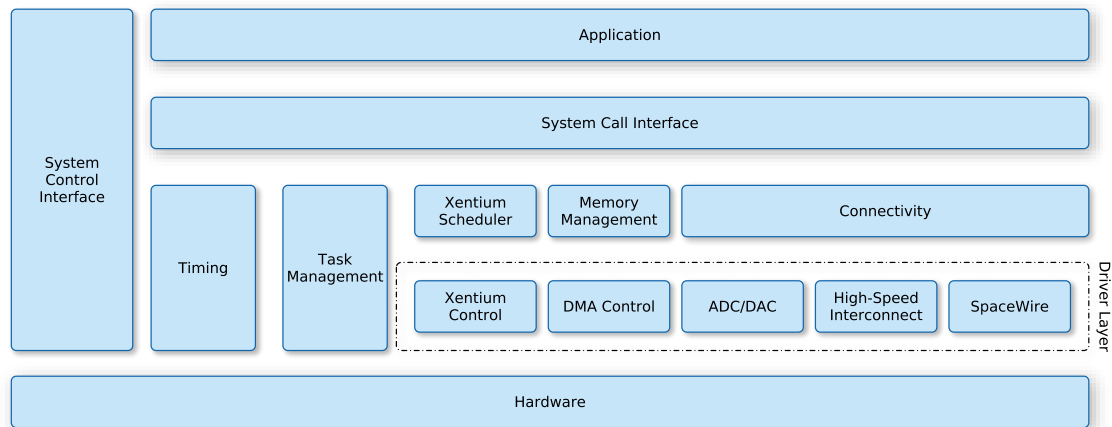


Figure 8.1: The logical model of LeanOS. Here, the “hardware” layer represents both the hardware and the hardware abstraction layer of the software.

In LeanOS, the [SSDP](#) hardware is accessed in multiple layers of abstraction (see Figure 8.1). Typical [CPU](#) tasks such as thread/task management and timer operation are used as part of the operating system kernel and are also accessible by user applications via a system call interface. Other functional hardware components of the [SSDP](#) such as the [NoC DMA](#) have their own driver modules. These are in turn used by the [Xentium](#) scheduler and other higher level modules in the operating system. Configuration of and access to the latter from user space is done via a system call interface. The system control interface serves as an intermediate between all layers of the operating system, where system or module states and hardware modes or usage statistics may be exported by individual components for external (user) access.