KERNKONZEPT

OBTAINING SAFETY & SECURITY CERTIFICATIONS FOR L4Re

Marcus Hähnel

In this talk

01 What is New in L4Re Land?

02 Safety & Security

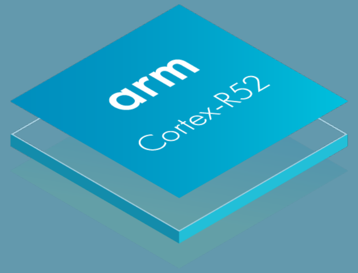
03 The L4Re Micro Hypervisor

04 What Will The Future Hold?

WHAT IS NEW IN L4Re LAND?

01

Evolving the Technology (Highlights)



- **RISC-V Support (incl. H-Extension)**
- **L4Re Micro Hypervisor for Cortex R-52/82**
- **Extended virtio support in Uvmm**
  - Virtio GPU Demo with simultaneous Android and Linux instances
- **ARM SystemReady**
  - arm64 UEFI Platforms
  - Example: AWS Graviton

The Arm SystemReady logo is a trademark of Arm Limited (or its subsidiaries) in the US and/or elsewhere. All rights reserved. The RISC-V logos are trademarks of RISC-V International.

Certified Digital Sovereignty

- **Secure Operating System**
  - German GEHEIM since January 2024
  - CC EAL4+ on the finish line
- **Safe Operating System**
  - L4Re-based EB corbos Hypervisor
  - ASIL-B by TÜV SÜD

SAFETY & SECURITY

Obtaining certifications with a pre-existing code-base

02

L4Re Certifications and Accreditations

GERMAN
GEHEIM

EU
SECRET

NATO
SECRET

- **2013**: first L4Re **VS-NfD** product
- **2017**: first L4Re **GEHEIM** product
- **2024**:
  - First operating system accredited by BSI for **GEHEIM**
  - **ASIL-B & SIL-2** certification for L4Re based product
- **2025: EAL4+** certification imminent

ISO26262

Many Documents and a bit of Code

- L4Re is basis of EB corbos Hypervisor by Elektrobit
- ASIL-B & SIL-2
- Certified as Safety Element out of Context (SEooC)
- Many improvements contributed back to Open Source release

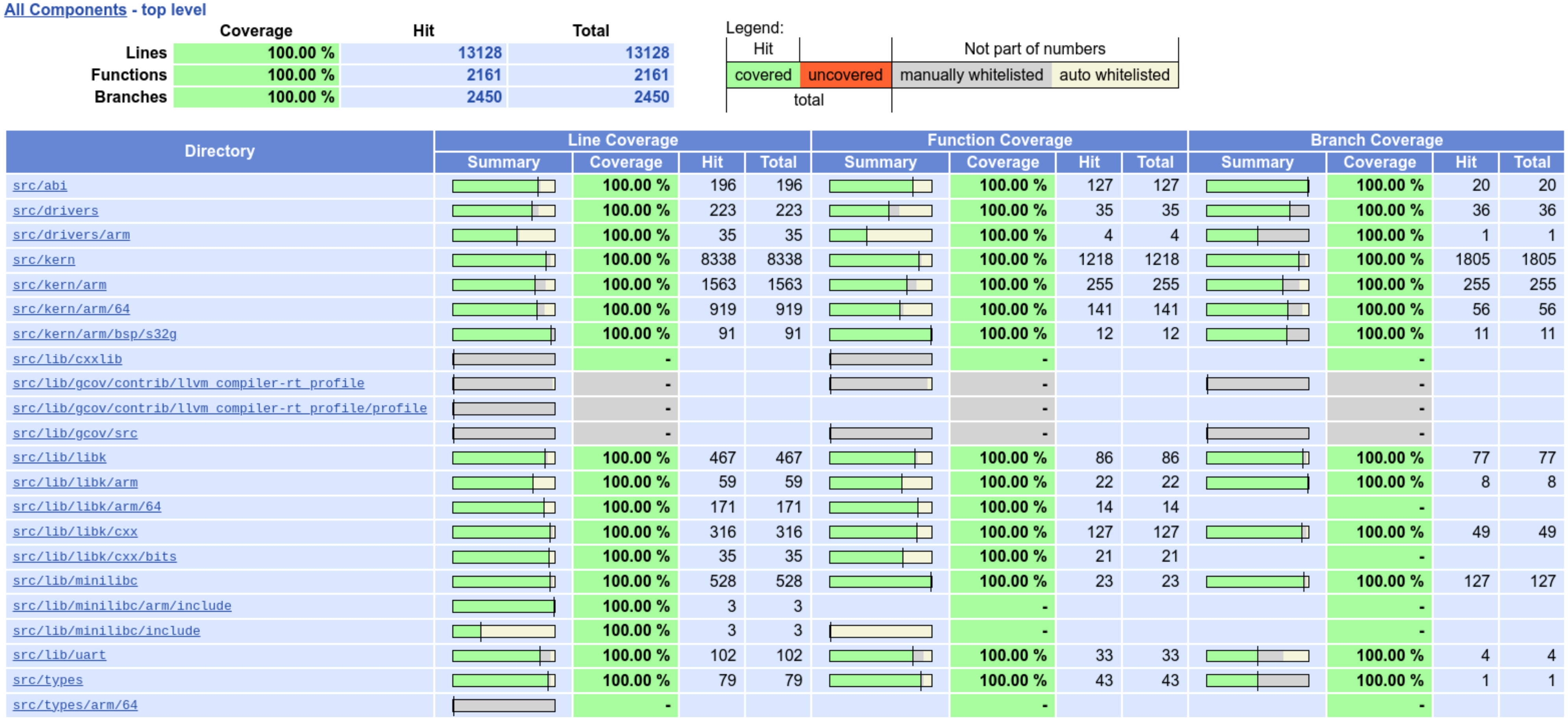
Static Analysis and MISRA

- **MISRA C++ 2008**
  - Reasonable subset
  - C++ 2023 better suitable
  - challenges to adapt to system software
- **Warning Free**
  - For a reasonable subset (-Wall and some more)
- **Static Analyzers are a challenge**

Tests, Documentation & Requiremements

- **LLD & SwAD**
  - 127 Pages SwAD (Core components)
  - 250 Pages LLD (Safety Components)
- **Requirements**
  - 104 architecture requirements
  - 525 detailed requirements
- **Tests**
  - >1000 tests in ~7000 test instances
  - >130k lines of test code
  - all linked to requirements

Code Coverage



All Components - top level

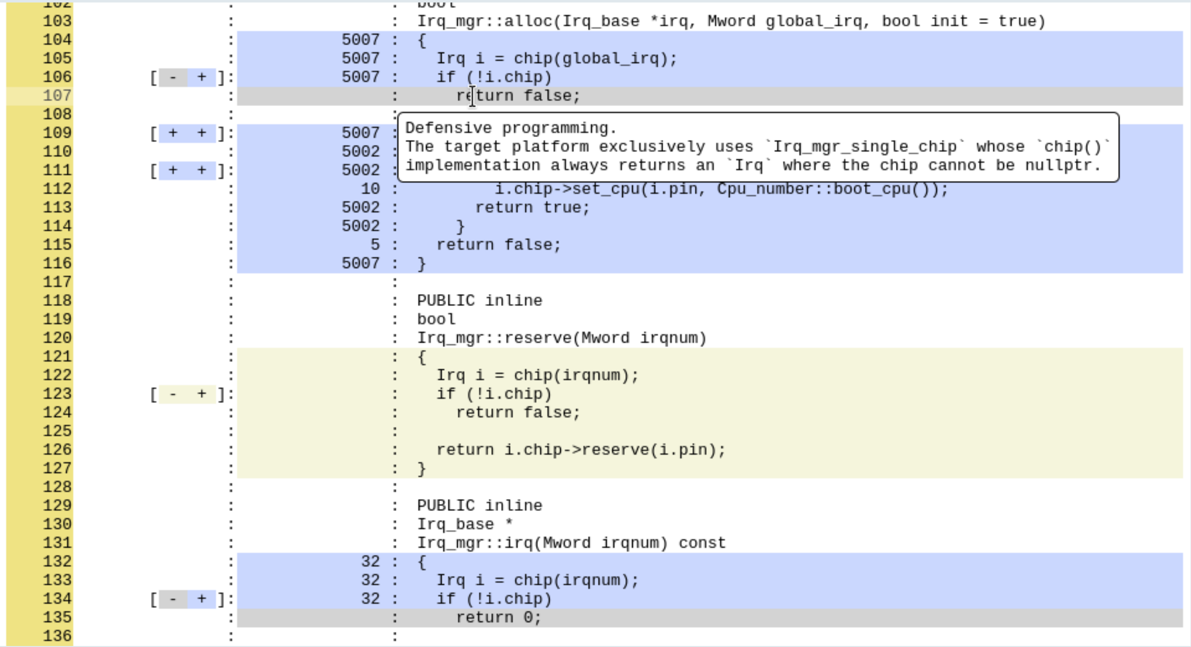
	Coverage	Hit	Total
Lines	100.00 %	13128	13128
Functions	100.00 %	2161	2161
Branches	100.00 %	2450	2450

Legend:

Hit		Not part of numbers	
covered	uncovered	manually whitelisted	auto whitelisted
total			

Directory	Line Coverage				Function Coverage				Branch Coverage			
	Summary	Coverage	Hit	Total	Summary	Coverage	Hit	Total	Summary	Coverage	Hit	Total
src/abi		100.00 %	196	196		100.00 %	127	127		100.00 %	20	20
src/drivers		100.00 %	223	223		100.00 %	35	35		100.00 %	36	36
src/drivers/arm		100.00 %	35	35		100.00 %	4	4		100.00 %	1	1
src/kern		100.00 %	8338	8338		100.00 %	1218	1218		100.00 %	1805	1805
src/kern/arm		100.00 %	1563	1563		100.00 %	255	255		100.00 %	255	255
src/kern/arm/64		100.00 %	919	919		100.00 %	141	141		100.00 %	56	56
src/kern/arm/bsp/s32g		100.00 %	91	91		100.00 %	12	12		100.00 %	11	11
src/lib/cxxlib		-				-				-		
src/lib/gcov/contrib/llvm_compiler-rt_profile		-				-				-		
src/lib/gcov/contrib/llvm_compiler-rt_profile/profile		-				-				-		
src/lib/gcov/src		-				-				-		
src/lib/libk		100.00 %	467	467		100.00 %	86	86		100.00 %	77	77
src/lib/libk/arm		100.00 %	59	59		100.00 %	22	22		100.00 %	8	8
src/lib/libk/arm/64		100.00 %	171	171		100.00 %	14	14		-		
src/lib/libk/cxx		100.00 %	316	316		100.00 %	127	127		100.00 %	49	49
src/lib/libk/cxx/bits		100.00 %	35	35		100.00 %	21	21		-		
src/lib/minilibc		100.00 %	528	528		100.00 %	23	23		100.00 %	127	127
src/lib/minilibc/arm/include		100.00 %	3	3		-				-		
src/lib/minilibc/include		100.00 %	3	3		-				-		
src/lib/uart		100.00 %	102	102		100.00 %	33	33		100.00 %	4	4
src/types		100.00 %	79	79		100.00 %	43	43		100.00 %	1	1
src/types/arm/64		-				-				-		

Code Coverage

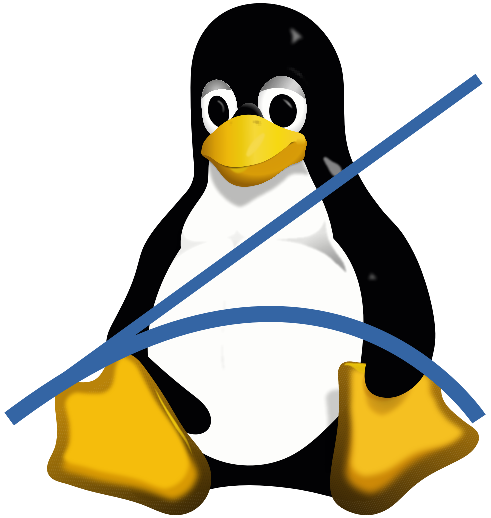


```
           :         :  bool
           :         :  Irq_mgr::alloc(Irq_base *irq, Mword global_irq, bool init = true)
           :    5007 :  {
           :    5007 :    Irq i = chip(global_irq);
  [ -  + ] :    5007 :    if (!i.chip)
           :         :      return false;
           :         :
  [ +  + ] :    5007 :
           :    5002 :
  [ +  + ] :    5002 :
           :      10 :          i.chip->set_cpu(i.pin, Cpu_number::boot_cpu());
           :    5002 :        return true;
           :    5002 :      }
           :       5 :    return false;
           :    5007 :  }
           :         :
           :         :  PUBLIC inline
           :         :  bool
           :         :  Irq_mgr::reserve(Mword irqnum)
           :         :  {
           :         :    Irq i = chip(irqnum);
  [ -  + ] :         :    if (!i.chip)
           :         :      return false;
           :         :
           :         :    return i.chip->reserve(i.pin);
           :         :  }
           :         :
           :         :  PUBLIC inline
           :         :  Irq_base *
           :         :  Irq_mgr::irq(Mword irqnum) const
           :      32 :  {
           :      32 :    Irq i = chip(irqnum);
  [ -  + ] :      32 :    if (!i.chip)
           :         :      return 0;
           :         :
```

```
Defensive programming.
The target platform exclusively uses `Irq_mgr_single_chip` whose `chip()`
implementation always returns an `Irq` where the chip cannot be nullptr.
```

- **314 whitelist entries in 9 categories**
- **Custom tooling to automatically determine optimized out code**
- **LLVM coverage vs. gcov**

A Way to a Safe Linux

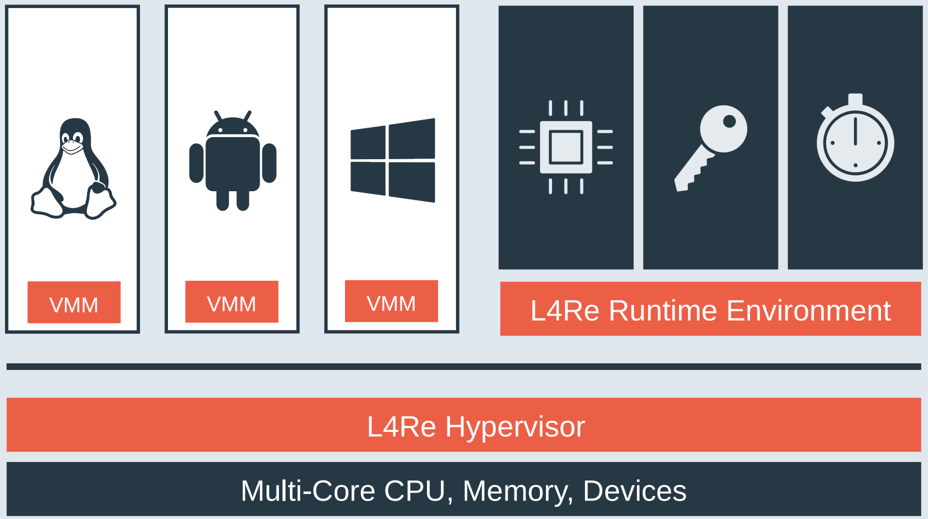


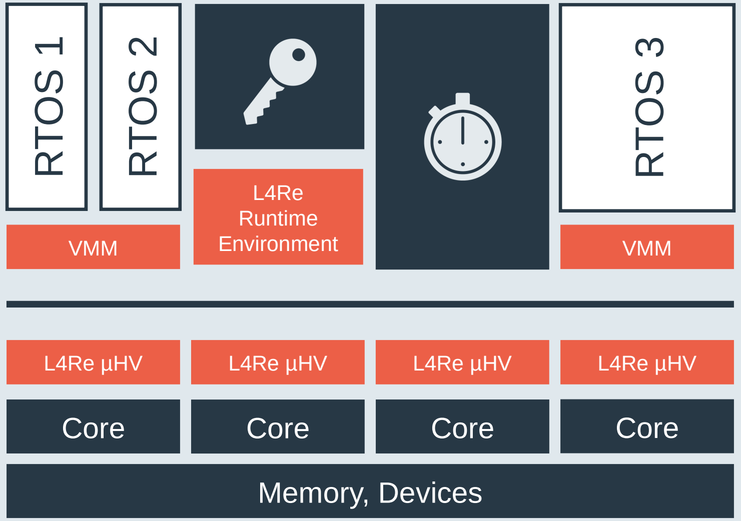
INTRODUCING: THE L4Re MICRO HYPERVISOR

Bringing the flexibility of MMU-systems to the Safety & Real-time MPU Domain

03

An L4Re for Real-time and Safety Hardware

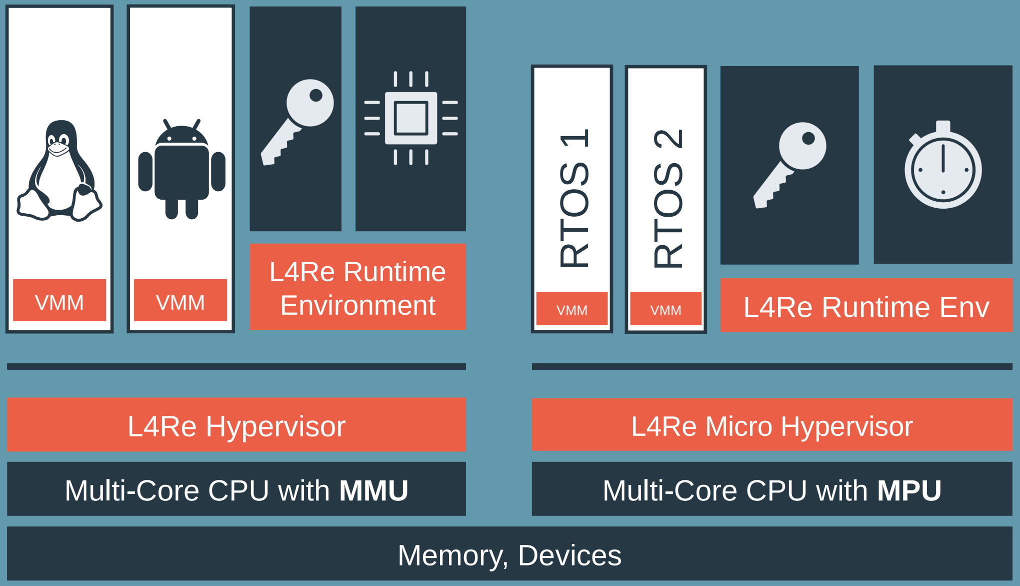




Micro Hypervisor on Cortex-R52 (NXP S32Z)

- L4Re for MPU-based systems
- Support for non-cache-coherent multi-core systems
- Cortex-R-based virtualization (Cortex-R52 and beyond)
- VM co-location on a core
- Size and feature optimized variant for memory-constraint systems

Flexible Deployment



- **L4Re scales from tiny embedded systems to large multi-core HPC systems**
- **Arm Cortex-A- & Cortex-R-based virtualization**
- **Flexible placement of applications on MMU or MPU system**
- **Upcoming cross-system communication**

WHAT THE FUTURE WILL HOLD

04

Technology on the Horizon

- Native L4Re cross-toolchains
- Rust Cross-Compiler
- ADA Support
- Improved public documentation and tutorials
- Tickless Kernel

Keeping the Certification Alive

- Contributors may not have safety as primary interest
- How to keep the required artifacts in sync?
  - Tests / Coverage
  - Requirements and links
- Ensure safety analysis
- All this without impeding outside contribution
- Open Source and Safety – It can be done!

KERNKONZEPT

THANK YOU

Marcus Hähnel | marcus.haehnel@kernkonzept.com

github.com/kernkonzept
www.l4re.org
www.kernkonzept.com