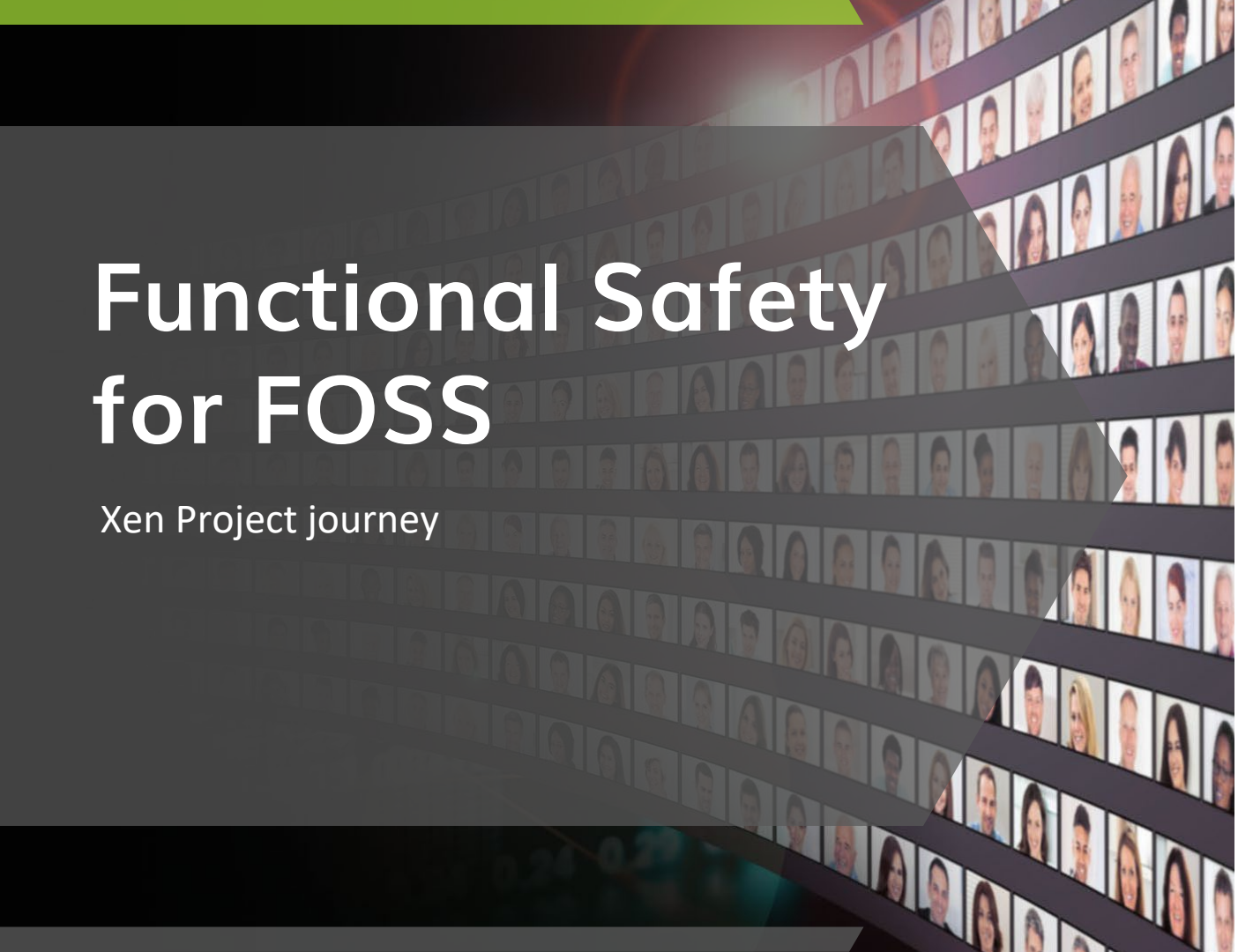
Functional Safety for FOSS

Xen Project journey

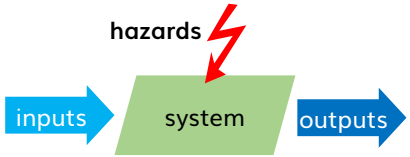


Linaro Connect
VIRTUAL • SPRING 2021

Background

What is Functional Safety?

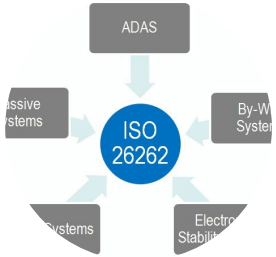
- Safety is a freedom from unacceptable risk of a harm
- Risk is a combination of probability and severity of a harm
- Harm means injury to people or damage to equipment or environment
- Functional Safety is absence of unreasonable risk due to hazards (potential source of harm) caused by malfunctioning behavior of the complex electronic systems



Functional Safety is regulated

IEC 61508

Generic Functional Safety standard applicable to (almost) any industry vertical



ISO 26262

Adaptation of IEC 61508 for Automotive systems. Many others exist e.g. for Medical, Railway, etc.

MISRA

Coding guidelines for automotive still widely used in mission-critical embedded applications

What do regulations demand?

- The product (safety) requirements are fully defined
- Able to demonstrate that requirements are correctly implemented by architecture, unit design, code
- The requirements, architecture, unit design, code and testing comply to the best practices described by the safety regulations
- The development process complies with regulations
- Able to demonstrate that all processes were strictly followed

Can existing FOSS be used for FuSa?

- Major features of the software that are uncommon for FOSS
  - Determinism
  - Fault processing
  - Defensive development
- Certain engineering practices not usually good in FOSS
  - Documentation
  - Engineering processes
- Special non-free tools, infrastructure and expertise

- Significant changes in how generic FOSS projects work and managed
- Until recently it was assumed two worlds cannot work together

Possible, but there's a catch

- It is extremely hard to strictly follow all requirements, so tailoring is possible, allowed and widely practiced
- The more you tailor, the higher the risk that the safety case does not pass and the higher the upfront cost
  - Tailoring = funding a specialist consultancy from special assessing company
- In the end, verification must demonstrate that everything has been done correctly
  - … or argue that what you have done is as good within what the standard requires
- Can only be done if assessors are actively involved in the development process

So, existing FOSS project like Xen

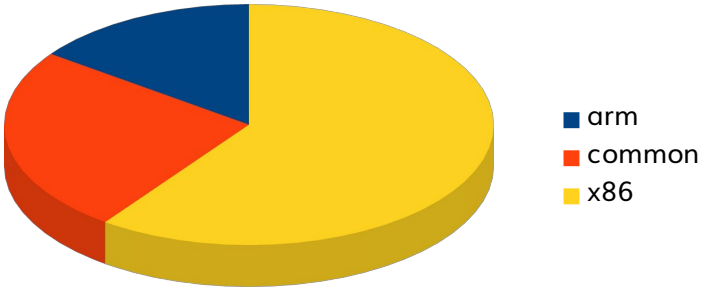
… shall at least:

- Somewhat re-work the codebase: interfaces, modularity, complexity, determinism, fault tolerance & processing
- Add any required missing artifacts: specifications, testing, etc. ensuring full traceability between them and code
- Strictly define development process and extend/modify where there are gaps
- Work with assessors on tailoring safety regulations
- Enforce the safety processes rigorously

Approach

Previous work

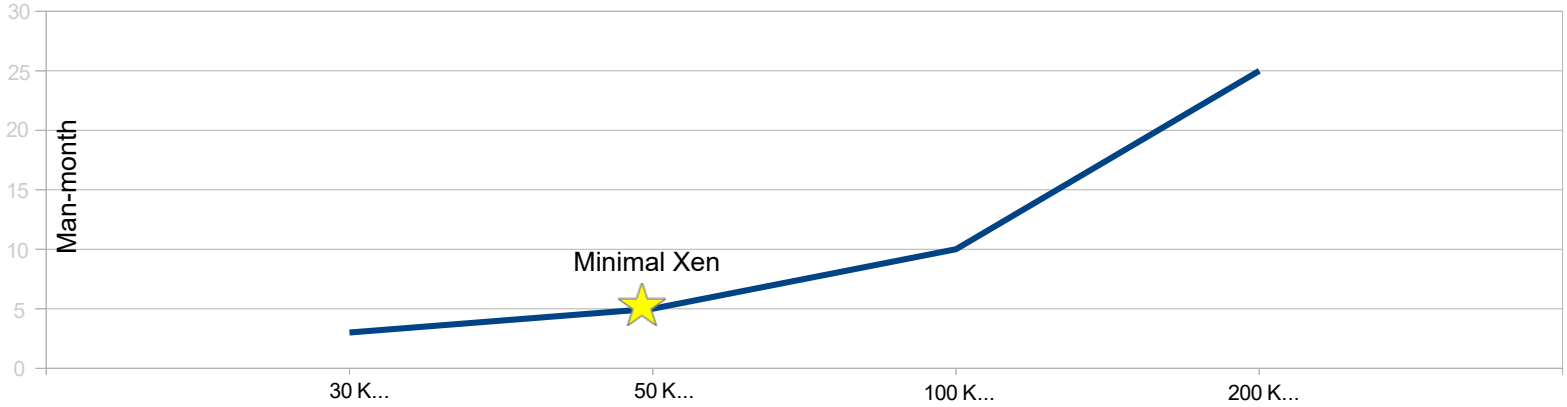
- Proprietary fork by safety-trained engineers
- DO-178 / DAL-C certification of Xen for Aerospace
- Effort of 5-15 man-years was needed for 1-off certification



Effort heavily depends on code size, so it make sense to certify some minimized codebase including only application essential functions

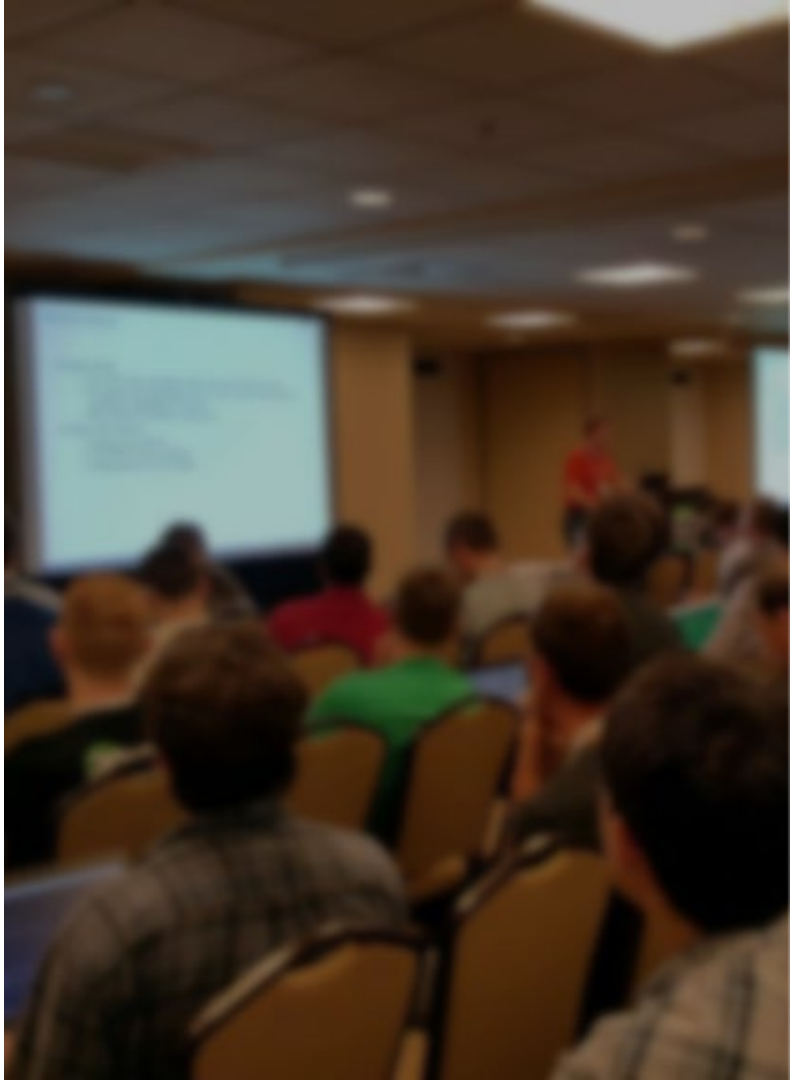
Community effort estimation

- Annual community investment is around 30 man-years
- Much higher than 5 man-years needed for minimal Xen
- But can FuSa be maintained in an adapted FOSS model?



Xen FuSa SIG

- Long term community work
- Leaders – Arm, EPAM, Xilinx
- Assessors – Exida, MIRA, TÜV
- Xen Project leadership team
- Kate Stewart from Linux Foundation as observer / advisor
- Industry representatives



High-Level Agreements

There must be a clear benefit the non-FuSa community

- Split development model with an open and a closed part
- There must be no uncertainty for the community why some specific changes (e.g. MISRA related) are pushed
- Changes to the open development workflow are minimal
- The workflow is git centric and there should not be no parallel universe for open part
- Anything we do for safety can only be done if there is agreement to implement changes in common code

Organization

Accountability – Commercial vendors

- Community cannot be responsible for the safety sign-off for the project
- Possible Split Development Model with accountable commercial vendor or group
- Commercial reference stacks for safety use-cases

Partnership – Linux Foundation projects

- **ELISA:** Founding members: Arm, BMW, KUKA, Toyota. Provide tools, processes and patterns for using Linux kernel in FuSa
- **Zephyr:** Goal set by Intel and some partners to establish Zephyr as a safety certifiable open source RTOS

Certification path & work scope

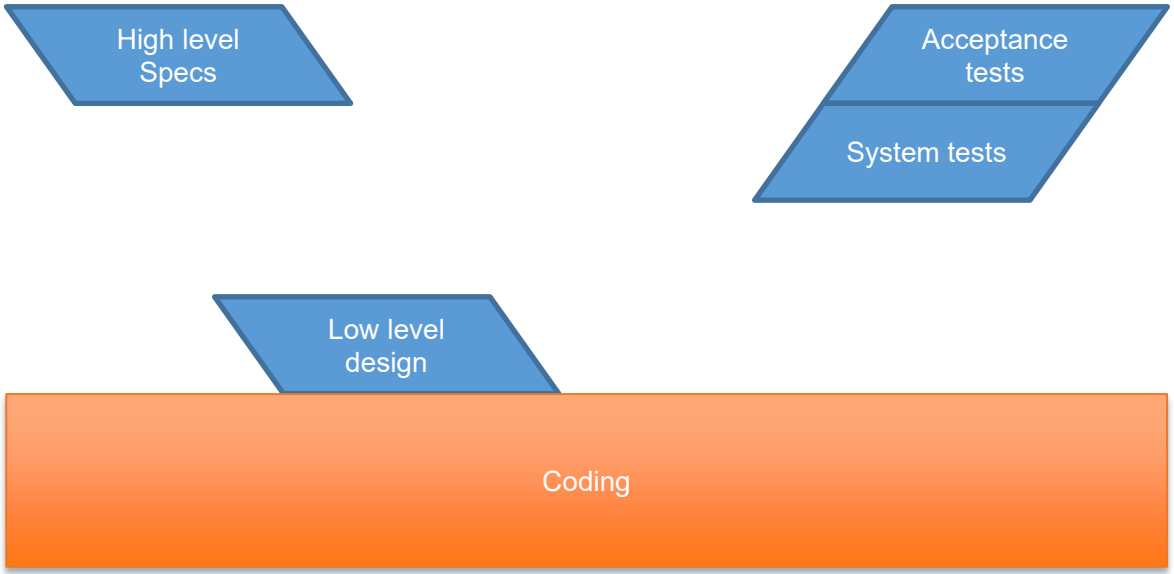
Certify Xen as ISO 26262 SEooC for ASIL B / via ISO 61508 route 3S

- Identify & eliminate gaps in requirements & documentation
- Identify & eliminate gaps in validation & verification
- Establish infrastructure for documentation maintenance & testing
- Implement fault processing and deterministic behavior
- Implement some defensive programming techniques and waive against unapplicable
- Support necessary certified tools
- Have safety management processes and tools in place

Requirements and Documentation

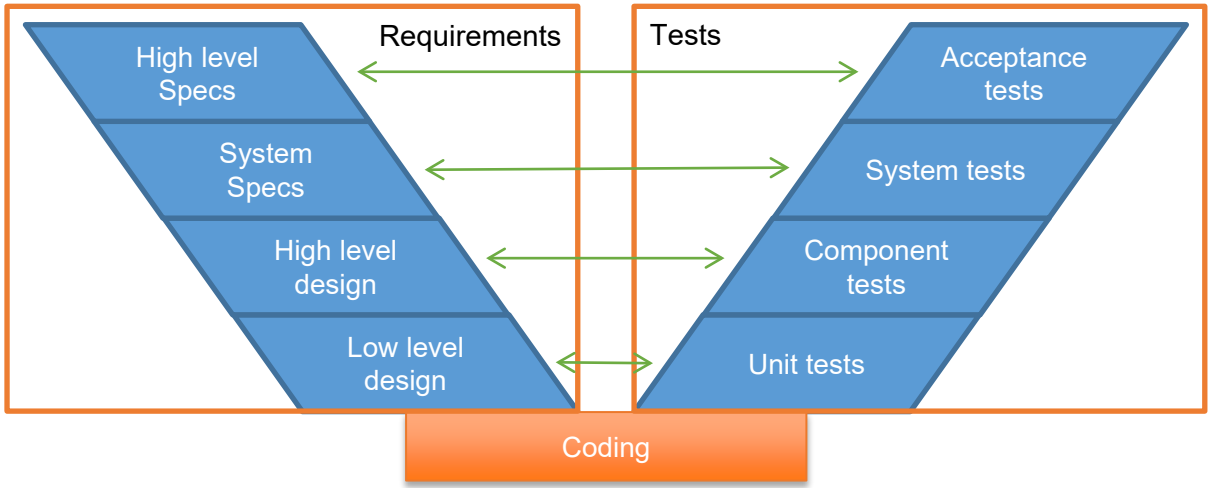
Requirements: V Model

How do we develop ?



Requirements: V Model

The theory



Requirements: What is it ?

- Several characteristics
  - Single function/feature
  - Testable/Measurable
  - Clear/Without ambiguity
- Examples:
  - "Xen shall never crash“ --- Not ambiguous but NOT TESTABLE
  - "Xen shall support up to 64 cores“ --- Clear, Testable and measurable
- Not completely documentation
  - information interesting for the user
  - Wording a bit specific: why "shall support" and not "supports" ?
- Could the documentation wording be acceptable as requirement ?
  - Yes if the wording is defined
  - and requirements are distinguished from comments or documentation

Requirements: Linking

- Goals:
  - From one test find the high level requirement
  - From one line of code find all the tests
  - From one high level requirement find all the code
  - With all intermediate steps (design or tests) from the V model
- Very consuming task
- Most common certification authority check

Requirements: Where to put them ?

- Standard: Specific "database"
  - Handles linking
  - proprietary software with licenses per user
  - Hard to maintain, especially for open source
- Chosen solution
  - High level requirements in a text file/table
    - Might be turned into proper documentation
  - Rest directly in code
    - Extracted using doxygen
    - Maintained with the code directly
  - Use doxygen to define and maintain the links

Requirements: Current status

- Work is done in collaboration with Zephyr project
  - Goal is to use same strategy and tools on both projects
  - Regular sync and share with Zephyr project
- Move Xen documentation to doxygen
  - Will improve the current Xen documentation
  - Preparation for Requirement extraction
- Write high level requirements
  - Agree on the terminology
  - Generate first examples to setup the doxygen tools

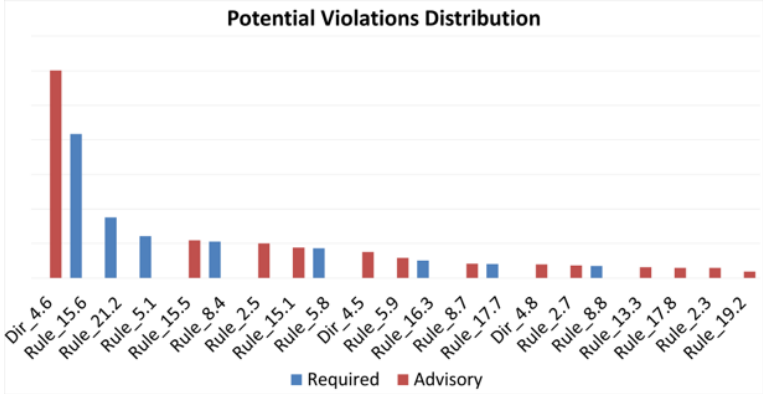
MISRA C

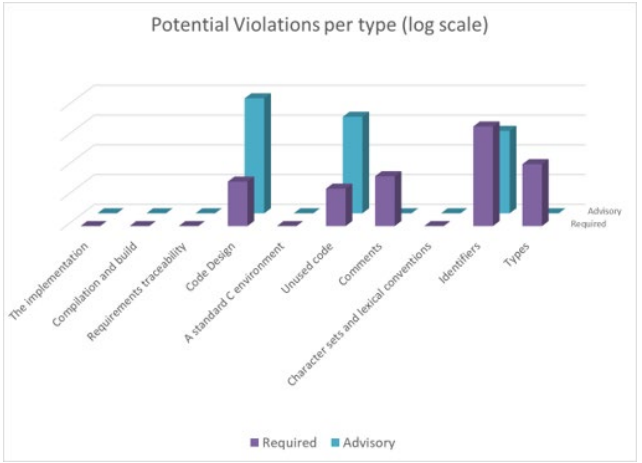
MISRA-C

- Safety guidelines for software written in C
- Rules and Directives
- Mandatory, Required, Advisory
- Deviations
- MISRA-C Static Code Analysis Tools

Misra-C and Xen: initial evaluation

- Feasible
- Top violations are false positives





MISRA-C and Xen: next steps

- Which guidelines make sense for Xen?
  - List all guidelines that apply (in collaboration with Zephyr)
  - How to check each of them for violations
- Find the right MISRA-C checker
- Integrate the MISRA-C checker into Xen Project's Gitlab infrastructure

Example: Rule 15.6

https://gitlab.com/MISRA/MISRA-C/MISRA-C-2012/Example-Suite/-/blob/master/R_15_06.c

Example: Rule 15.6

The real issue is:

```
if ( one )
    if ( two )
        foo();
    else
        bar();
```

Solution: `GCC --Wmisleading-indentation`

Example: Rule 5.1

https://gitlab.com/MISRA/MISRA-C/MISRA-C-2012/Example-Suite/-/blob/master/R_05_01_2.c

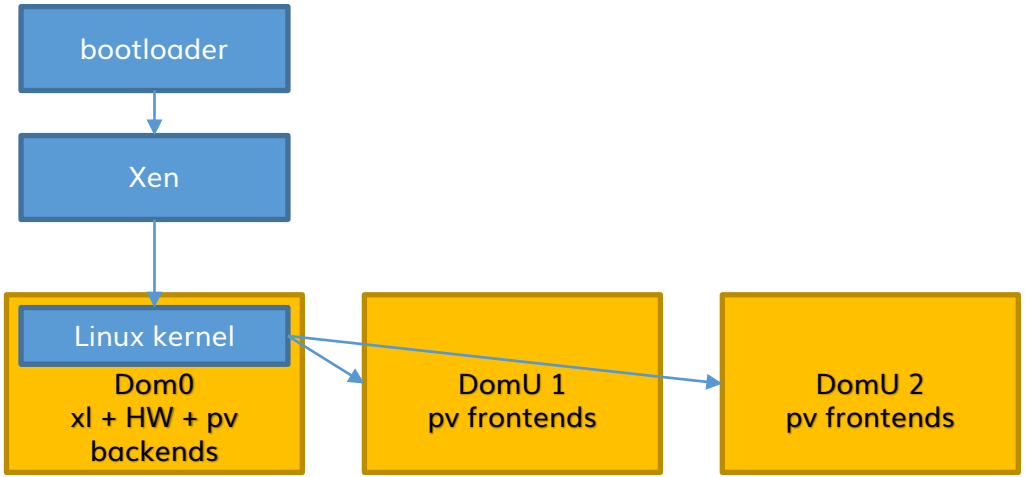
Example: Rule 5.1

- C99 requires first 31 characters to be significant
- Solution: document restrictions on the compiler used

Certifiable Xen Configurations

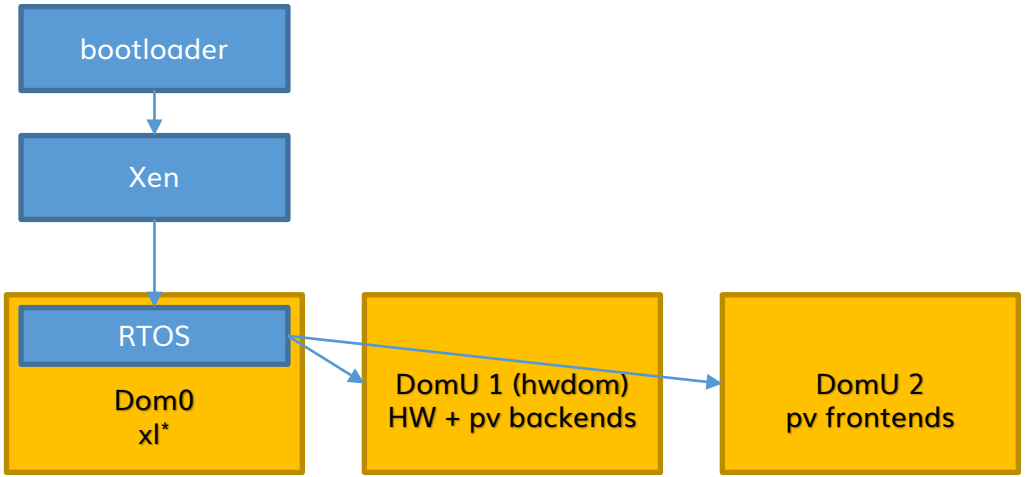
Traditional Xen boot

Dom0: Linux kernel + GNU tools
- Hard to comply with FuSa
- All HW in a single point of failure

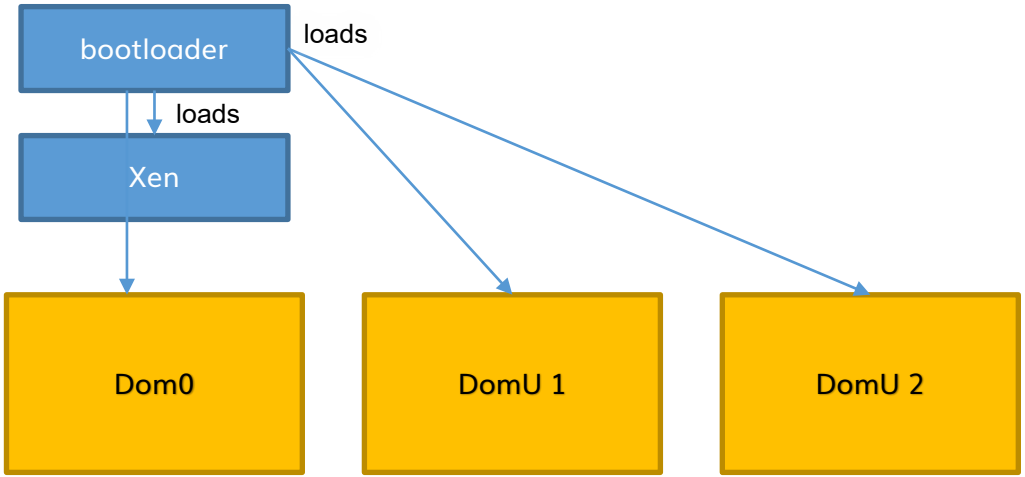


“Thin” dom0

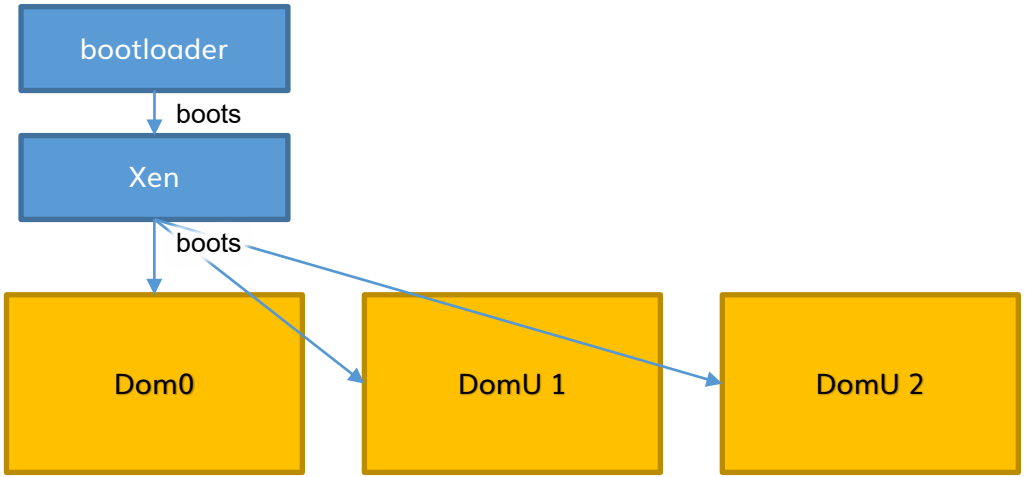
Dom0: FuSa compliant RTOS
- FuSa compliant rewritten xl
- No HW (except FuSa-related)



Dom0less



Dom0less



Dom0less

- Parallel Boot
  - shorter boot times
  - independent from Dom0
- No requirements on Dom0 running
- "True Dom0less": Dom0 is absent
- No Dom0 to safety-certify

Xen RT & IRQ Latency

POC

- *Zephyr OS build zephyr-v2.4.0-2750-g0f2c858a39fc*
- *Counter freq is 33280000 Hz, period is 30 ns*
- *Basic Xen support added to Zephyr OS upstream*
- *RTDS / 1x pCPU*

https://lists.xenproject.org/archives/html/xen-devel/2021-02/msg01404.html
https://lists.xenproject.org/archives/html/xen-devel/2021-01/msg00781.html

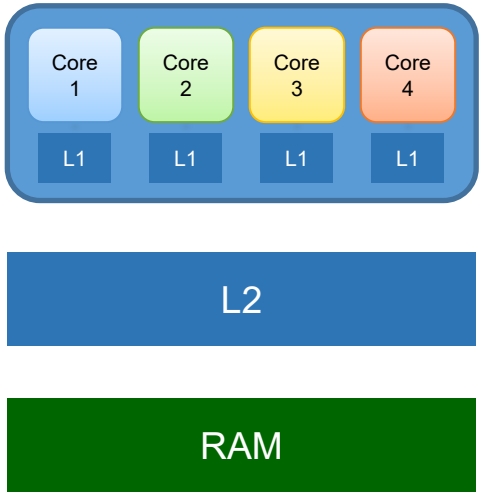
mean: 488 (14640 ns)
stddev: 1656 (49680 ns)

mean: 288 (8640 ns)
stddev: 20 (600 ns)

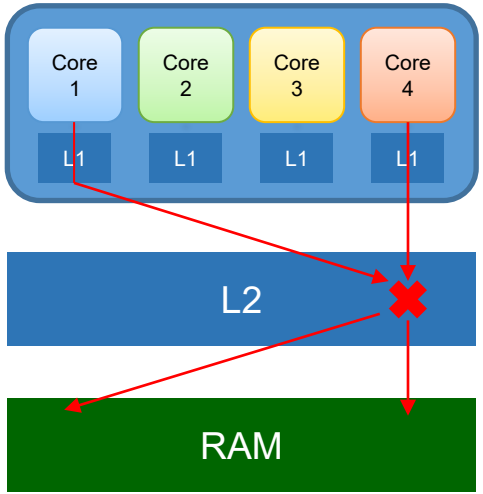
TODO

- Fix Arm's long-running hypercalls (memory partitioning)
  - Maximum latency is about 2ms!
  - Used only on domain creation
- Preemption of generic hypercalls
- Time accounting for Xen internals

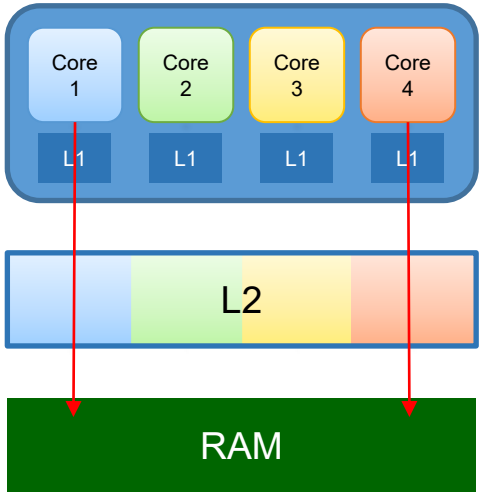
Cache Coloring



Cache Coloring

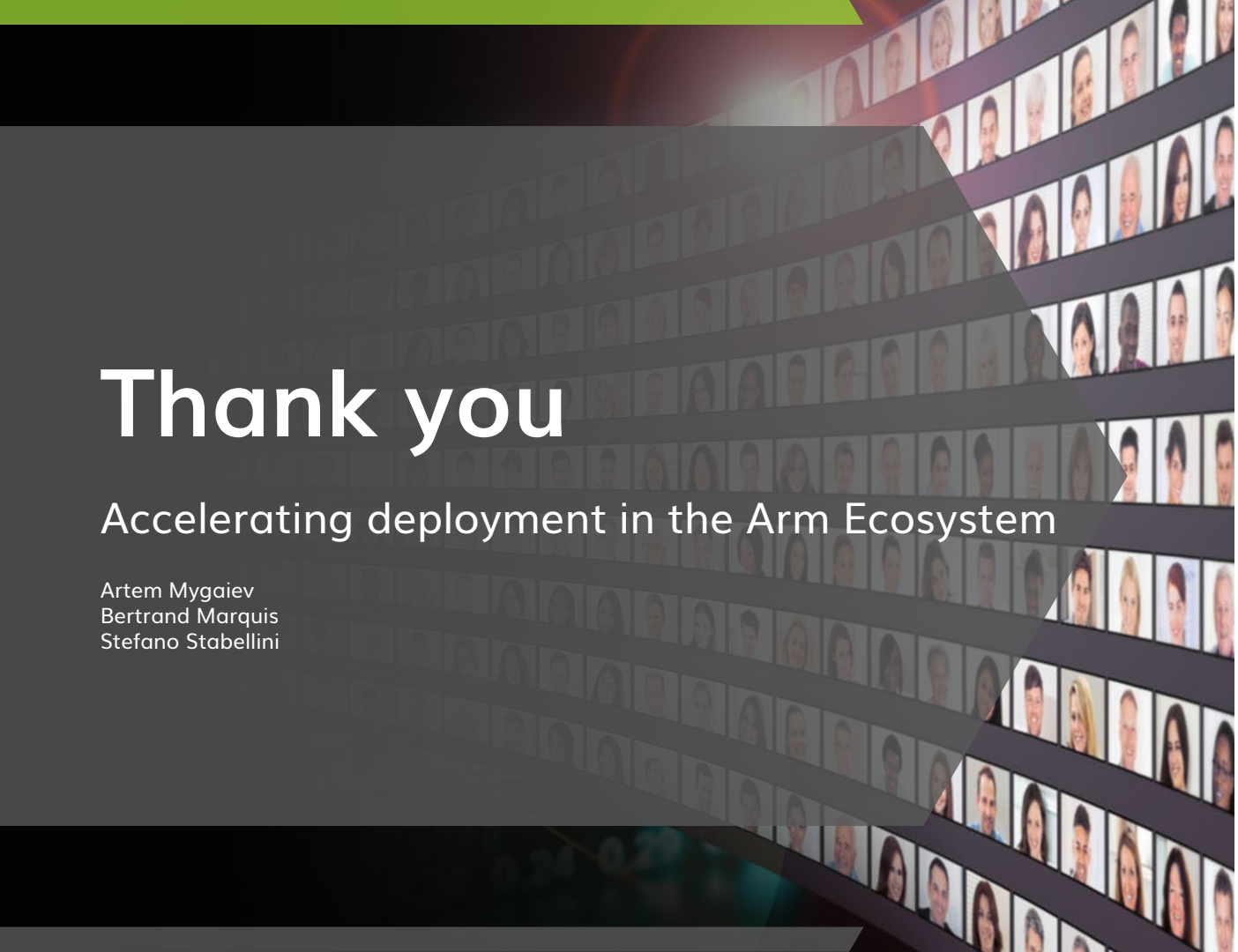


Cache Coloring



Cache Coloring

- Cache partitioning in software
- Fully dedicated cache lines to each VM
- Prevent cache interference between VMs
- Excellent interrupt latency and execution time improvements
    - IRQ latency < 5us



Thank you

Accelerating deployment in the Arm Ecosystem

Artem Mygaiev
Bertrand Marquis
Stefano Stabellini

Linaro Connect
VIRTUAL • SPRING 2021