LLVM AArch64

FOSDEM – LLVM dev room – February 2015

Kristof Beyls

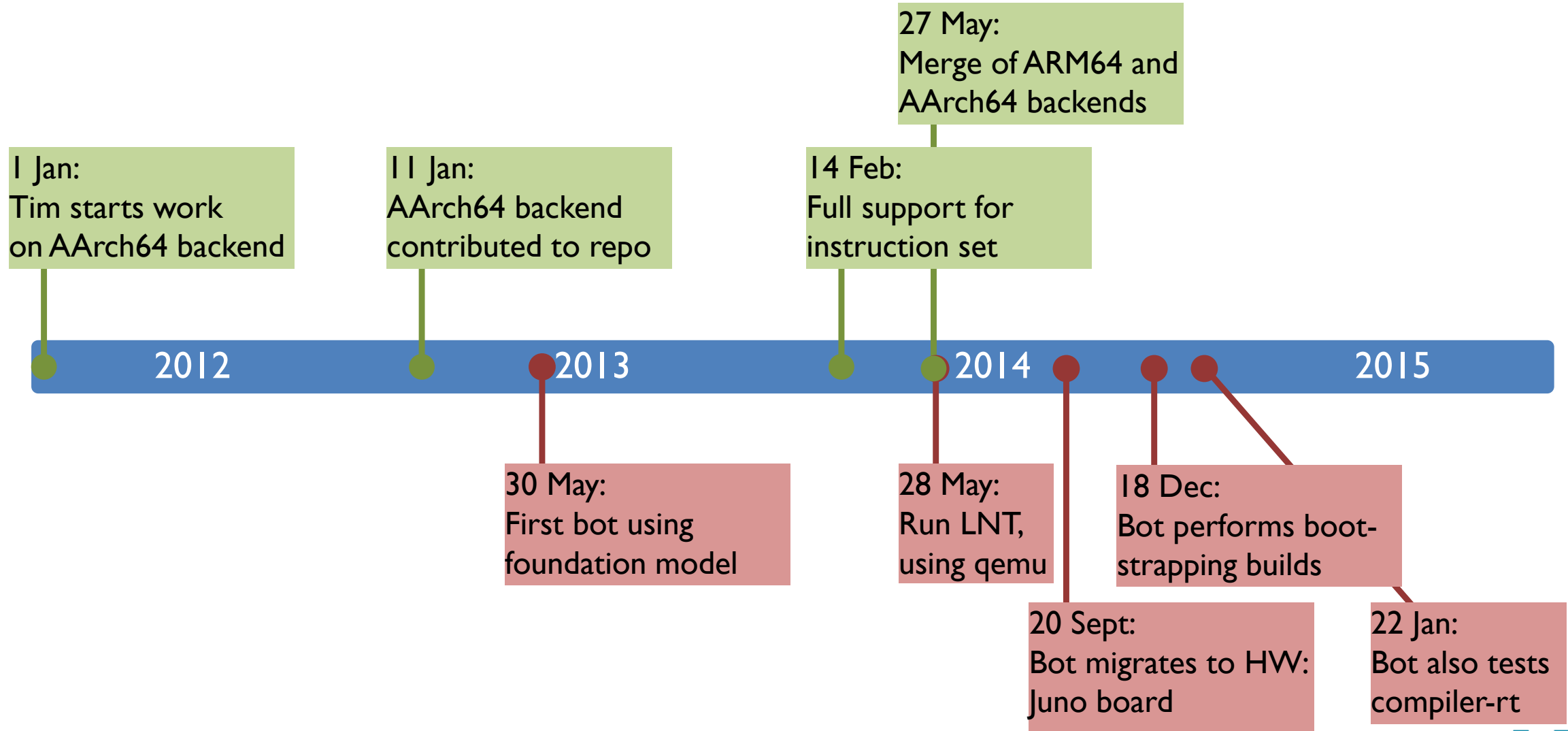The Architecture for the Digital World®

ARM®

Overview

- Some history of how AArch64 support got added to clang & llvm.
- Current status of support for AArch64

ARM®

What is AArch64?
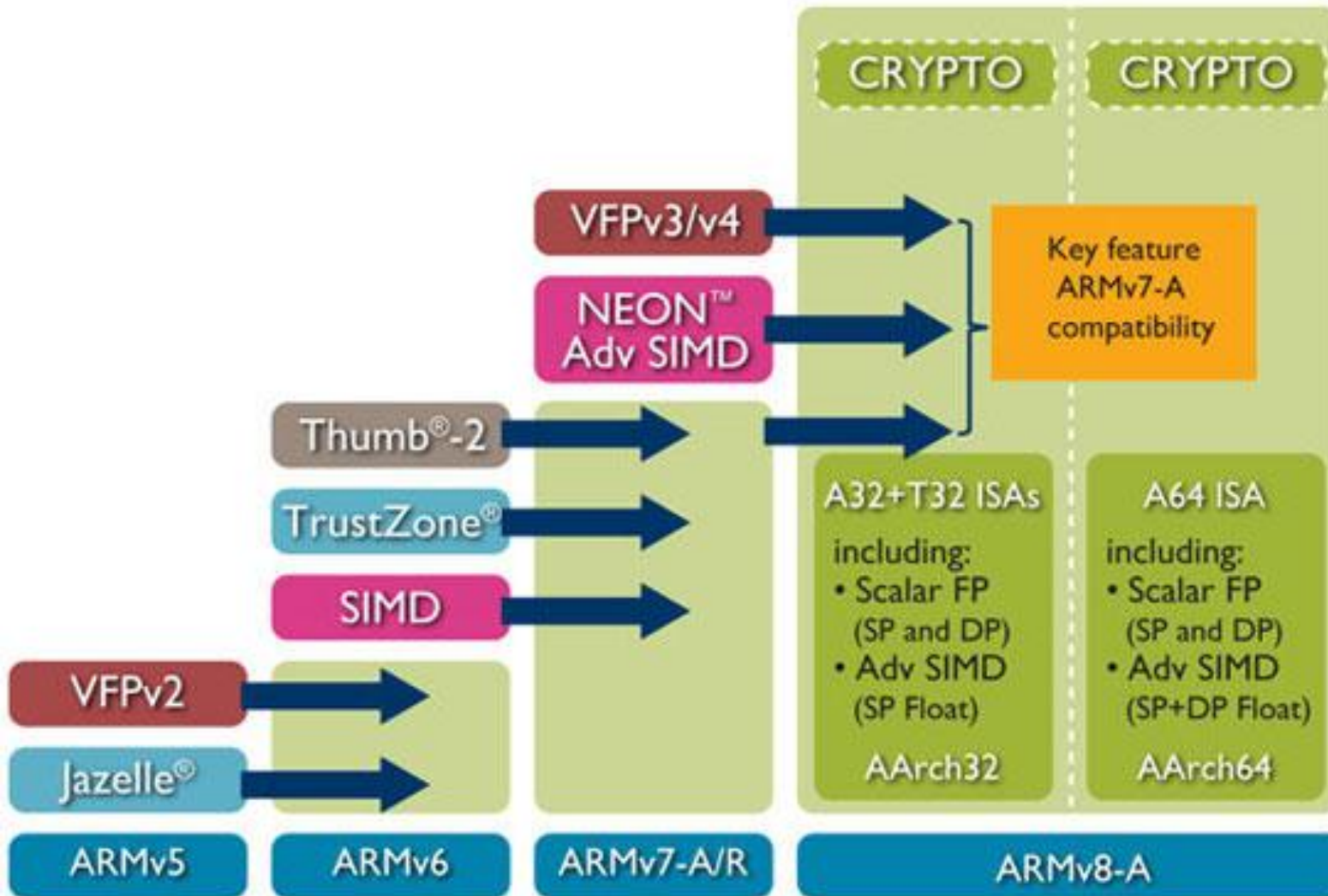
History of support for AArch64 in LLVM

ARM®

Timeline

27 May:
Merge of ARM64 and
AArch64 backends

1 Jan:
Tim starts work
on AArch64 backend

11 Jan:
AArch64 backend
contributed to repo

14 Feb:
Full support for
instruction set

2012   2013   2014   2015
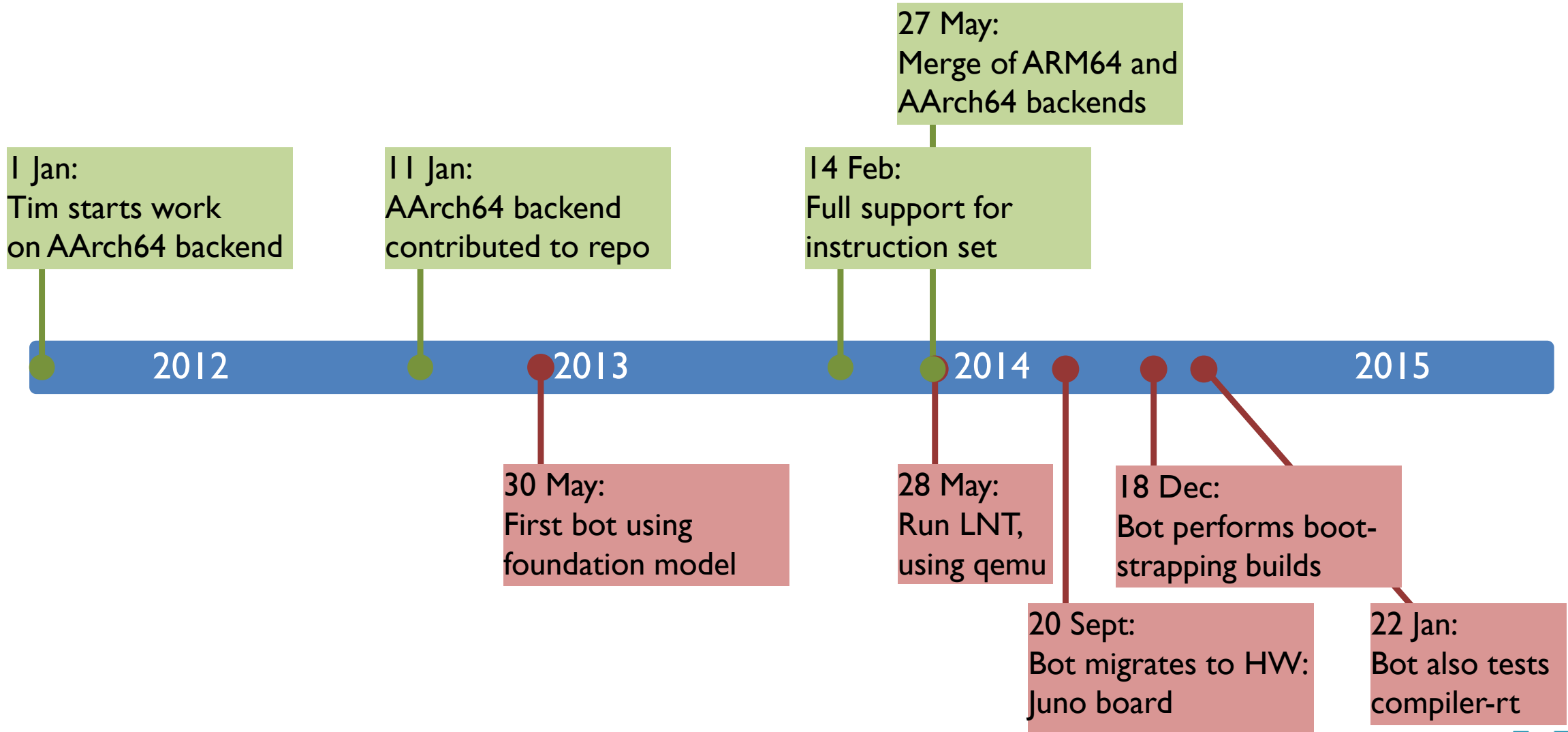
30 May:
First bot using
foundation model

28 May:
Run LNT,
using qemu

18 Dec:
Bot performs boot-
strapping builds

20 Sept:
Bot migrates to HW:
Juno board

22 Jan:
Bot also tests
compiler-rt

ARM®

Timeline

Focus on correctness.
LLVM isn't afraid of refactoring -> regression tests at least as important as code.

27 May:
Merge of ARM64 and AArch64 backends

1 Jan:
Tim starts work on AArch64 backend

11 Jan:
AArch64 backend contributed to repo

14 Feb:
Full support for instruction set

2012 **2013** **2014** **2015**

30 May:
First bot using foundation model

28 May:
Run LNT, using qemu

18 Dec:
Bot performs boot-strapping builds

Focus on correctness.
LLVM isn't afraid of refactoring; regression tests can't cover everything -> buildbots provide important additional test coverage

20 Sept:
Bot migrates to HW: Juno board

22 Jan:
Bot also tests compiler-rt

ARM®

Timeline

1 Jan:
Tim starts work
on AArch64 backend

11 Jan:
AArch64 backend
contributed to repo

27 May:
Merge of ARM64 and
AArch64 backends

14 Feb:
Full support for
instruction set

2012   **2013**   **2014**   **2015**

30 May:
First bot using
foundation model

28 May:
Run LNT,
using qemu

18 Dec:
Bot performs boot-
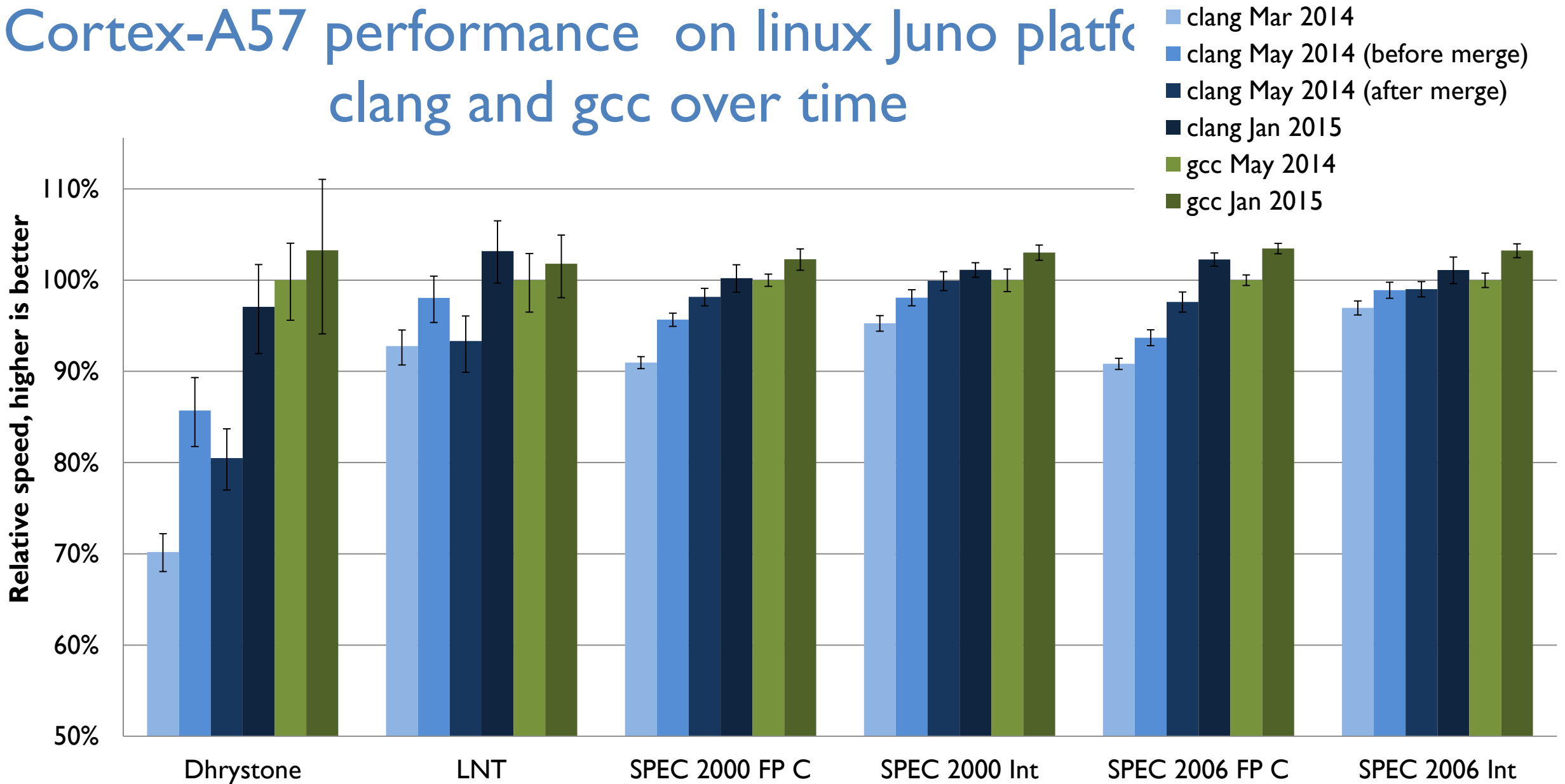strapping builds

20 Sept:
Bot migrates to HW:
Juno board

22 Jan:
Bot also tests
compiler-rt

ARM®

Cortex-A57 performance on linux Juno platform: clang and gcc over time

Legend:
- clang Mar 2014
- clang May 2014 (before merge)
- clang May 2014 (after merge)
- clang Jan 2015
- gcc May 2014
- gcc Jan 2015

Y-axis: Relative speed, higher is better (50% to 110%)

X-axis categories: Dhrystone, LNT, SPEC 2000 FP C, SPEC 2000 Int, SPEC 2006 FP C, SPEC 2006 Int

ARM®

Current status of support for AArch64

ARM®

What works?

- Lots
    - C & C++ generally just works
    - Little endian & Big endian
    - Neon intrinsics
    - Performance of generated code getting improved all the time.

ARM®

Some missing features

- Buildbots tracking performance of generated code

- Straightforward link time optimizations, due to lack of open source linkers integrating with LLVM for LTO and supporting AArch64 – *although AArch64 support in gold is actively being developed/almost done now?*

- ILP32

- More not-frequently-used features are discovered over time, as more software gets pushed through the backend. Two examples from last month:
  - no support for over-aligned stack objects
  - Thread-Local Storage: llvm produces relocations that linkers on AArch64-linux can't handle yet.

ARM®

Status of AArch64 support in other LLVM sub-projects

- Most projects use top-of-trunk LLVM

- Therefore, the best way to monitor the status of a given subproject is to look at the current status of the LLVM buildbots testing the particular subproject.

- For the main sub-projects other than clang (front end) or LLVM (mid-end and backend), there currently isn't any AArch64 buildbot, so the status is a bit unclear:

  - Compiler-rt (builtins/libgcc-equivalent): *buildbot is running the tests since very recently.*
  - Compiler-rt (sanitizers): *buildbot is running the tests since very recently.*
  - Libc++/libc++abi: should be largely working?
  - Lld: largely not working, but active development is going on?
  - Lldb: ?

ARM®

Summary

- AArch64 backend has been available in llvm.org for 2 years.

- Major steps forward last year
    - Basic full support for the architecture in front-end & back-end.
    - Generated code has reasonable performance.

- Status of support for AArch64 in the various LLVM sub-projects is a bit unclear at the moment.

ARM®

END

ARM®