Linaro Connect
MADRID 2024 | MAY 14-17 2024

Overview of Linaro Toolchain CI configurations

Maxim Kuvyrkov
Linaro Toolchain Working Group

Intro

and philosophy

Linaro Toolchain WG CI: Principles

1. Engineer’s time is precious

2. Machine time is not precious, but limited

3. Upstream projects move faster than we can build and test

Consequences:

- Engineers are notified only about single-commit regressions
- Before-and-after comparisons are optimal for investigation
- Absolute results are interesting, but less useful for CI

Disclaimers

- Linaro Toolchain WG CI != Linaro CI
  - Linaro has many other CI projects
  - Linaro TCWG team alone has three (!) separate CI efforts

- There is no best CI
  - CI is a tool for a job

- We are toolchain engineers
  - Our CI has front-ends, middle-end and back-ends

CI Approaches

Build - Build - Build

- Test commits as quickly as possible
- The faster the test, the smaller the commit range

Pros

- Relatively simple
- Low-latency feedback
- Perfect for slow-moving projects

Cons

- Fast-moving projects get large commit ranges
- O(n) complexity

Build - Reduce - Bisect - Report

- Test check-point commits
- Reduce and bisect bad builds down to a single commit

Pros

- Single-commit ranges
- Handles many fast-moving projects
- O(log(n)) complexity

Cons

- More scripting to support reduce and bisect steps
- High-latency feedback

Regressions and Metrics

Regressions vs Failures

- Regression is a new failure that does not appear in baseline results
- Regressions are relative, failures are absolute
  - Detecting regressions requires a baseline
- As engineers we are fixing failures, but as maintainers we are interested in regressions

Single-commit regressions are the main goal of Linaro Toolchain CI

Comparing CI results

- Regressions are relative
- Comparison against a baseline
- Metric is a comparable property of CI result
  - Two CI results can be compared with “same”, “better” or “worse” output
- Every CI loop tracks a single metric
  - Minor consequence: we get hundreds of CI loops

Build/Testsuite metrics

- Number of successful build steps
  - Build GNU Toolchain
  - Build LLVM Toolchain
- Presence of new FAILs/ERRORs/etc.
  - Test Binutils, GCC, Glibc, GDB, QEMU
- List of binaries built
  - Build-test of Android/AOSP
- Number of object files built
  - Build-test of Linux kernel

Benchmarking metrics

- Number of CPU cycles
  - Single-threaded benchmarks
- Number of execution seconds
  - Multi-threaded benchmarks
- Size of .text segment
- Number of NEON-vectorized loops
- Number of SVE-vectorized loops

CI Projects / CI Configs

CI Projects

1. tcwg_aosp
2. tcwg_binutils, tcwg_bootstrap, tcwg_gcc, tcwg_gdb, tcwg_glibc
3. tcwg_gnu_cross, tcwg_gnu_embed, tcwg_gnu_mingw, tcwg_gnu_native
4. Tcwg_bmk-code_size, tcwg_bmk-code_speed, tcwg_bmk-code_{sve, vect}
5. tcwg_bmk-fujitsu_speed, tcwg_bmk-qc_speed
6. tcwg_kernel

tcwg_aosp

1. tcwg_aosp-build-aosp/oriole
2. tcwg_aosp-code_size-aosp/oriole
3. tcwg_aosp-code_size-dalvikvm/oriole
4. tcwg_aosp-code_size-dex2oat/oriole
5. tcwg_aosp-code_size-surfaceflinger/oriole

tcwg_{binutils, bootstrap, gcc, gdb, glibc}

1. tcwg_*_build
   a. tcwg_binutils_build
   b. tcwg_gcc_build
   c. tcwg_gdb_build
   d. tcwg_glibc_build
   e. tcwg_bootstrap_build
2. tcwg_binutils_check
3. tcwg_bootstrap_check
4. tcwg_gcc_check
5. tcwg_gdb_check
6. tcwg_glibc_check

tcwg_*

1. aarch64
2. arm

tcwg_bootstrap_*

1. bootstrap_debug
2. bootstrap
3. bootstrap_lto
4. bootstrap_O1
5. bootstrap_O3
6. bootstrap_profiled
7. bootstrap_profiled_lto
8. bootstrap_profiled_lto_lean
9. bootstrap_ubsan

tcwg_gnu_{cross, embed, mingw, native}

1. tcwg_gnu_*_build
   a. tcwg_gnu_cross_build
   b. tcwg_gnu_embed_build
   c. tcwg_gnu_mingw_build
   d. tcwg_gnu_native_build
2. tcwg_gnu_cross_check_binutils
3. tcwg_gnu_cross_check_gcc
4. tcwg_gnu_embed_check_binutils
5. tcwg_gnu_embed_check_gcc
6. tcwg_gnu_mingw_check_binutils
7. tcwg_gnu_mingw_check_gcc
8. tcwg_gnu_native_check_binutils
9. tcwg_gnu_native_check_gcc
10. tcwg_gnu_native_check_gdb
11. tcwg_gnu_native_check_glibc

1. aarch64
2. arm
   a. arm_eabi
   b. arm_v7a_softfp_eabi
   c. thumb_m0_eabi
   d. thumb_m0_soft_eabi
   e. thumb_m23_eabi
   f. thumb_m23_soft_eabi
   g. thumb_m33_eabi
   h. thumb_m33_hard_eabi
   i. thumb_m3_eabi
   j. thumb_m3_softfp_eabi
   k. thumb_m55_hard_eabi
   l. thumb_m7_eabi
   m. thumb_m7_hard_eabi
   n. Thumb_v8a_hard_eabi

tcwg_bmk-code_{size, speed, sve, vect}

1. tcwg_bmk-code_size-coremark
2. tcwg_bmk-code_size-cpu2017fast
3. tcwg_bmk-code_size-cpu2017rate
4. tcwg_bmk-code_size-spec2k6
5. tcwg_bmk-code_speed-coremark
6. tcwg_bmk-code_speed-cpu2017rate
7. tcwg_bmk-code_speed-spec2k6
8. tcwg_bmk-code_sve-cpu2017fast
9. tcwg_bmk-code_vect-cpu2017fast
10. tcwg_bmk-code_vect-cpu2017rate
11. tcwg_bmk-code_vect-spec2k6

1. gnu
2. llvm

1. aarch64
2. arm

1. O2
2. O3
3. Os
4. Oz

1. no-LTO
2. LTO

tcwg_bmk-fujitsu_speed, tcwg_bmk-qc_speed

1. tcwg_bmk-fujitsu_speed-cpu2017speed/gnu-aarch64-O2
2. tcwg_bmk-fujitsu_speed-cpu2017speed/gnu-aarch64-O3
3. tcwg_bmk-fujitsu_speed-cpu2017speed/llvm-aarch64-O2
4. tcwg_bmk-fujitsu_speed-cpu2017speed/llvm-aarch64-O3
5. tcwg_bmk-qc_speed-cpu2017rate/llvm-aarch64-O2
6. tcwg_bmk-qc_speed-cpu2017rate/llvm-aarch64-O2_LTO
7. tcwg_bmk-qc_speed-cpu2017rate/llvm-aarch64-O3
8. tcwg_bmk-qc_speed-cpu2017rate/llvm-aarch64-O3_LTO

tcwg_kernel

1. gnu
2. llvm

1. aarch64
2. arm

1. lts
2. stable
3. mainline
4. next

1. defconfig
2. allmodconfig
3. allnoconfig
4. allyesconfig

Let’s look at some of these!

What are Your Questions?

Linaro Connect
MADRID 2024 | MAY 12-17 2024

Thank you