Linaro Connect
MADRID 2024 | MAY 14-17 2024

Advanced features of Linaro Toolchain CI

Christophe Lyon
Thiago Bauermann
Laurent Alfonsi

Agenda

- Flaky tests support
- Multiple components
- Interesting cases
- Pre-commit CI
- Some statistics

Flaky tests

Flaky tests

- Some testsuites have ‘flaky’ tests: results are slightly different every time

- They cause various problems
  - Waste of time investigating on the CI maintainer side
  - Waste of time on the developers side
  - Lack of confidence in the automatic notifications
    - Developers tend to ignore them when there are too many false alarms
    - If the community loses trust in the CI, it’s very difficult to rebuild

Flaky tests support

- Sources of flaky tests
  - Timing assumptions / race conditions
  - Testing harness problems
  - DejaGnu problems
  - Sometimes only on a subset of architectures / environments
    - e.g., stable on hardware, flaky with QEMU

- Possible approaches to flaky tests
  - Reduce parallelism
  - Create and maintain lists of flaky tests
  - Remove or XFAIL flaky tests

- Our (automated) approach to flaky tests
  - Detect flaky tests
  - Add flaky test to the list of flaky tests
  - Ignore flaky tests for a few weeks
  - Re-detect flaky tests

Flaky tests detection

- Run testsuite
- Check failures against the list of expected FAILs and flaky tests (baseline results)
- If we have new [unexpected] FAILs, re-run testsuite
  - Only the offending subset
- If new FAILs are still FAILing – they are FAILs (i.e., regressions)
- If new FAILs are now PASSing – they are flaky tests
- Keep re-running progressively smaller subsets of the testsuite until there are no more unexpected FAILs.

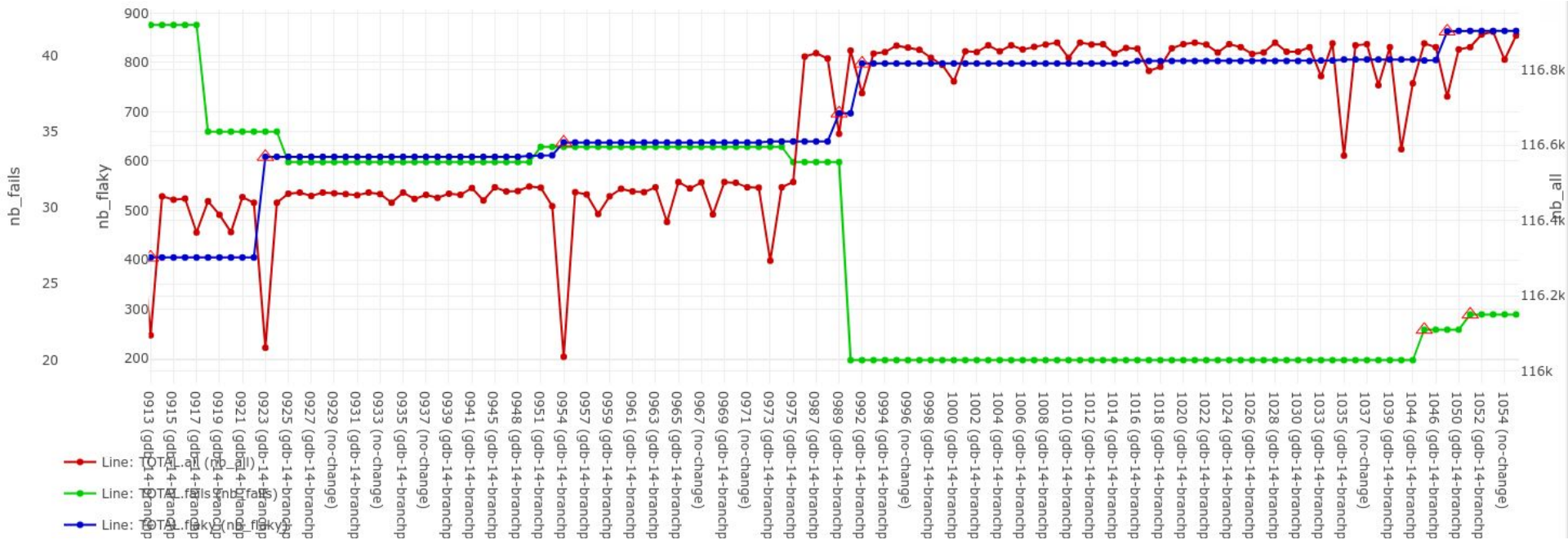
```
Using flaky fails file …/flaky.xfail
Using expected fails file …/baseline.xfail
Starting testsuite run #0.
FAIL: xxx1
FAIL: xxx2
   ⋮
Starting testsuite run #1.
PASS: xxx1
FAIL: xxx2
   ⋮
Starting testsuite run #2.
FAIL: xxx3
   ⋮
Starting testsuite run #3.
   ⋮
```

Detected new FAIL → PASS flaky test:
flaky,expire=20240807 | FAIL: xxx1

New flaky tests were found: do another run.

flaky,expire=20240807 | FAIL: xxx3

LNT Dashboard – GDB on aarch64-linux



Multiple components

Classic GNU toolchain single component CI

- <component> is **one** of Binutils, GCC, GDB, Glibc, QEMU
- Build <component>
  - git clone
  - make all
- Run testsuite
  - make check
- Compare results
  - Fetch baseline results
  - Compare new result summaries vs baseline result summaries

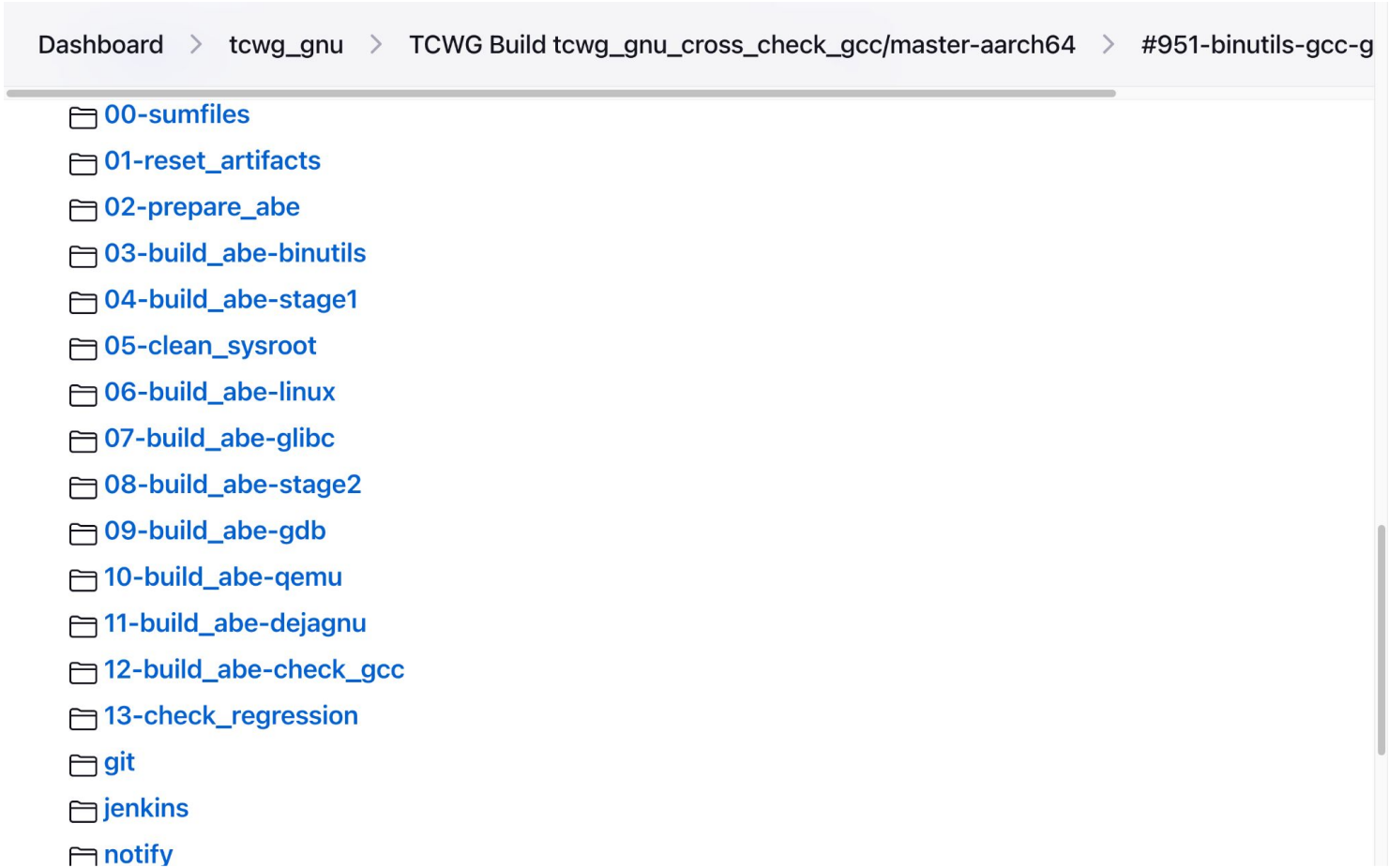
Full GNU toolchain CI: multiple components

- Take tip-of-trunk of Binutils, GCC, Glibc, **and** GDB
  - **and** tip-of-trunk Linux (for kernel headers)
  - **and** tip-of-trunk QEMU (for testing)

- Build tip-of-trunk toolchain

- Run testsuites

- Bisect any failures down to a single commit, in any of the components


Dashboard > tcwg_gnu > TCWG Build tcwg_gnu_cross_check_gcc/master-aarch64 > #951-binutils-gcc-g
00-sumfiles
01-reset_artifacts
02-prepare_abe
03-build_abe-binutils
04-build_abe-stage1
05-clean_sysroot
06-build_abe-linux
07-build_abe-glibc
08-build_abe-stage2
09-build_abe-gdb
10-build_abe-qemu
11-build_abe-dejagnu
12-build_abe-check_gcc
13-check_regression
git
jenkins
notify

Build → Reduce → Bisect → Report → Reset

- Build #(N-1): Pre-existing baseline results from previous good build

- Build #(N): Build **new** branches of all components
  - Build **new** Binutils,GDB,Glibc,Linux,QEMU and **new** GCC – FAIL
  - Trigger 2 builds to **reduce** failure to a single component

- Build #(N+1): Build **new** branch of Binutils,GDB,Glibc,Linux,QEMU
  - Build **new** Binutils,GDB,Glibc,Linux,QEMU and **baseline** GCC – PASS
  - Successful build updates **baseline** results

- Build #(N+2): Build **new** branch of GCC
  - Build **baseline** Binutils,GDB,Glibc,Linux,QEMU and **new** GCC – FAIL
  - Trigger bisect build to **bisect** commit range of GCC

Interesting cases

A few interesting cases

GNU-1188

- binutils patch causes a regression in Glibc tests
  - **`elf: Add _bfd_elf_link_m[un]map_section_contents`**
  - `FAIL: wcsmbs/tst-wcstod-round`
- Fixed in binutils

GNU-1136

- gdb patch causes a regression in GCC tests
  - **`Gdb: Only search types in cp_lookup_rtti_type`**
  - `FAIL: libstdc++-prettyprinters/cxx11.cc print ecmiaow`
  - `FAIL: libstdc++-prettyprinters/cxx11.cc print emiaow`
- Longstanding GDB issue, workaround applied to the testcase

From: ci_notify@linaro.org
To: tom@tromey.com
Cc: gdb-testers@sourceware.org
Subject: [Linaro-TCWG-CI] gdb-14-branchpoint-1411-g033bc67bdb0: FAIL: 2 regressions on arm
Date: Tue, 30 Jan 2024 19:03:33 +0000 (UTC) [thread overview]
Message-ID: <578038658.1014.1706641415312@jenkins.jenkins> (raw)

[-- Attachment #1: Type: text/plain, Size: 3204 bytes --]

Dear contributor, our automatic CI has detected problems related to your patch(es). Please find some details below. If you have any questions, please follow up on linaro-toolchain@lists.linaro.org mailing list, Libera's #linaro-tcwg channel, or ping your favourite Linaro toolchain developer on the usual project channel.

We appreciate that it might be difficult to find the necessary logs or reproduce the issue locally. If you can't get what you need from our CI within minutes, let us know and we will be happy to help.

We track this report status in https://linaro.atlassian.net/browse/GNU-1136 , please let us know if you are looking at the problem and/or when you have a fix.

In master-arm after:

```
  | commit gdb-14-branchpoint-1411-g033bc67bdb0
  | Author: Tom Tromey <tom@tromey.com>
  | Date:   Tue Sep 19 17:39:31 2023 -0600
  |
  |     Only search types in cp_lookup_rtti_type
  |
  |     This changes cp_lookup_rtti_type to only search for types -- not
  |     functions or variables.  Due to the symbol-matching hack, this could
  |     just use SEARCH_TYPE_DOMAIN, but I think it's better to be clear; also
  |     I hold on to some hope that perhaps the hack can someday be removed.
```

FAIL: 2 regressions

regressions.sum:

```
                === libstdc++ tests ===

Running libstdc++:libstdc++-prettyprinters/prettyprinters.exp ...
FAIL: libstdc++-prettyprinters/cxx11.cc print ecmiaow
FAIL: libstdc++-prettyprinters/cxx11.cc print emiaow
```

From: Jonathan Wakely <jwakely@redhat.com>
To: Tom Tromey <tom@tromey.com>
Cc: Maxim Kuvyrkov <maxim.kuvyrkov@linaro.org>,
        gdb-testers@sourceware.org,
         Linaro Toolchain Working Group
        <linaro-toolchain@lists.linaro.org>
Subject: Re: [Linaro-TCWG-CI] gdb-14-branchpoint-1411-g033bc67bdb0: FAIL: 2 regressions on arm
Date: Wed, 13 Mar 2024 23:57:16 +0000   [thread overview]
Message-ID: <CACb0b4=DNpT0r2GYPozJ1fomuybxukAm5SpFFakoefBoDVQ70A@mail.gmail.com> (raw)
In-Reply-To: <87h6h9x38t.fsf@tromey.com>

On Wed, 13 Mar 2024 at 20:53, Tom Tromey wrote:
>
> >> Just a guess, but maybe making the type global instead of a local type
> >> (with no linkage) will solve it:
> >>
> >> --- a/libstdc++-v3/testsuite/libstdc++-prettyprinters/cxx11.cc
> >> +++ b/libstdc++-v3/testsuite/libstdc++-prettyprinters/cxx11.cc
> >> @@ -63,6 +63,11 @@ struct datum
> >>
> >> std::unique_ptr<datum> global;
> >>
> >> +struct custom_cat : std::error_category {
> >> + const char* name() const noexcept { return "miaow"; }
> >> + std::string message(int) const { return ""; }
> >> +};
>
> gdb doesn't generally handle local types well -- a longstanding bug
> nobody has tried to fix.
>
> However my patch wasn't intended to introduce this problem.
> Perhaps it is at fault.

I've pushed the workaround to trunk now anyway, as r14-9457-ga8c7c3a40953e3

I haven't backported it yet.

Pre-commit CI

Pre-commit CI

- Instead of catching regressions (post-commit), why not test patches **before** they are merged?
  - Also gives useful information to reviewers/maintainers

- Interacts with patchwork
  - Fetch patches from patchwork
  - Apply patches on top of baseline revision
  - Run the same build/test recipe as post-commit CI
  - Update “check” status in patchwork
  - Notify author/submitter in case of regression

Patch	Series	rb/tb	S/W/F	Date	Submitter
arm: [MVE intrinsics] Fix support for predicate constants [PR target/114801]	arm: [MVE intrinsics] Fix support for predicate constants [PR target/114801]	- -	- - -	2024-04-26	Christophe Lyon
[2/2] Remove support for nontemporal stores with ssa_names on lhs [PR112976]	[1/2] Add verification of gimple_assign_nontemporal_move_p [PR112976]	- -	- - -	2024-04-26	Andrew Pinski
[1/2] Add verification of gimple_assign_nontemporal_move_p [PR112976]	[1/2] Add verification of gimple_assign_nontemporal_move_p [PR112976]	- -	- - -	2024-04-26	Andrew Pinski
vax: resolve long-standing documentation bugs re floating-point codegen [PR79646]	vax: resolve long-standing documentation bugs re floating-point codegen [PR79646]	- -	3 - -	2024-04-26	Abe Skolnik
aarch64: Use cinc for small constants instead of just add [PR112304]	aarch64: Use cinc for small constants instead of just add [PR112304]	- -	3 - -	2024-04-26	Andrew Pinski
aarch64: Fix normal returns inside functions which use eh_returns [PR114843]	aarch64: Fix normal returns inside functions which use eh_returns [PR114843]	- -	2 - 1	2024-04-26	Andrew Pinski
[committed] libstdc++: Do not apply localized formatting to NaN and inf [PR114863]	[committed] libstdc++: Do not apply localized formatting to NaN and inf [PR114863]	- -	- 2 -	2024-04-26	Jonathan Wakely
LoongArch: Add constraints for bit string operation define_insn_and_split's [PR114861]	LoongArch: Add constraints for bit string operation define_insn_and_split's [PR114861]	- -	4 - -	2024-04-26	Xi Ruoyao
middle-end/114734 - wrong code with expand_call_mem_ref	middle-end/114734 - wrong code with expand_call_mem_ref	- -	- - 2	2024-04-26	Richard Biener
[committed,gcc-14] libstdc++: Update status tables to refer to GCC 14 not mainline	[committed,gcc-14] libstdc++: Update status tables to refer to GCC 14 not mainline	- -	- - 2	2024-04-26	Jonathan Wakely
[committed,gcc-13] libstdc++: Update status tables to refer to GCC 13 not mainline	[committed,gcc-13] libstdc++: Update status tables to refer to GCC 13 not mainline	- -	- - 2	2024-04-26	Jonathan Wakely

aarch64: Fix normal returns inside functions which use eh_returns [PR114843]

Message ID	20240426170740.3001529-1-quic_apinski@quicinc.com
State	New
Headers	show
Series	aarch64: Fix normal returns inside functions which use eh_returns [PR114843] | expand

Checks

Context	Check	Description
linaro-tcwg-bot/tcwg_gcc_build--master-arm	success	Testing passed
linaro-tcwg-bot/tcwg_gcc_check--master-arm	fail	Testing failed
linaro-tcwg-bot/tcwg_gcc_build--master-aarch64	success	Testing passed
linaro-tcwg-bot/tcwg_gcc_check--master-aarch64	success	Testing passed

Commit Message

Andrew Pinski April 26, 2024, 5:07 p.m. UTC

```
  The problem here is that on a normal return path, we still restore the
eh data return when we should not.
Instead of one return path in the case of eh_return, this changes over
to use multiple returns pathes just like a normal function.
On the normal path (non-eh return), we need to skip restoring of the eh
return data registers.
```

Pre-commit CI challenges

- Patches do not apply or are incomplete
  - Patches are extracted from mailing lists
  - Patches have implicit dependencies
- Patches cannot be trusted
  - Testing environment needs to be containerized
  - ... but we need to extract diagnostics for email reports
- Testing bandwidth
  - Post-commit build can test hundreds of revisions at a time
  - Pre-commit build tests one patch at a time

Upstream feedback

Upstream feedback

- Initial feedback was positive but...
  - A few people complained about the notifications
  - We worked on improving the messages, and access to useful information

- Very positive feedback from GNU Cauldron presentations

- We then got more positive feedback
- We explicitly encourage developers to contact us

- Several messages explicitly saying Linaro Toolchain CI is a great improvement

- "**Linaro CI**" often mentioned by developers when discussing their patches

- Requests for new features / improvements
  - Verify ChangeLog, coding rules, etc...

Pre-commit CI and upstream

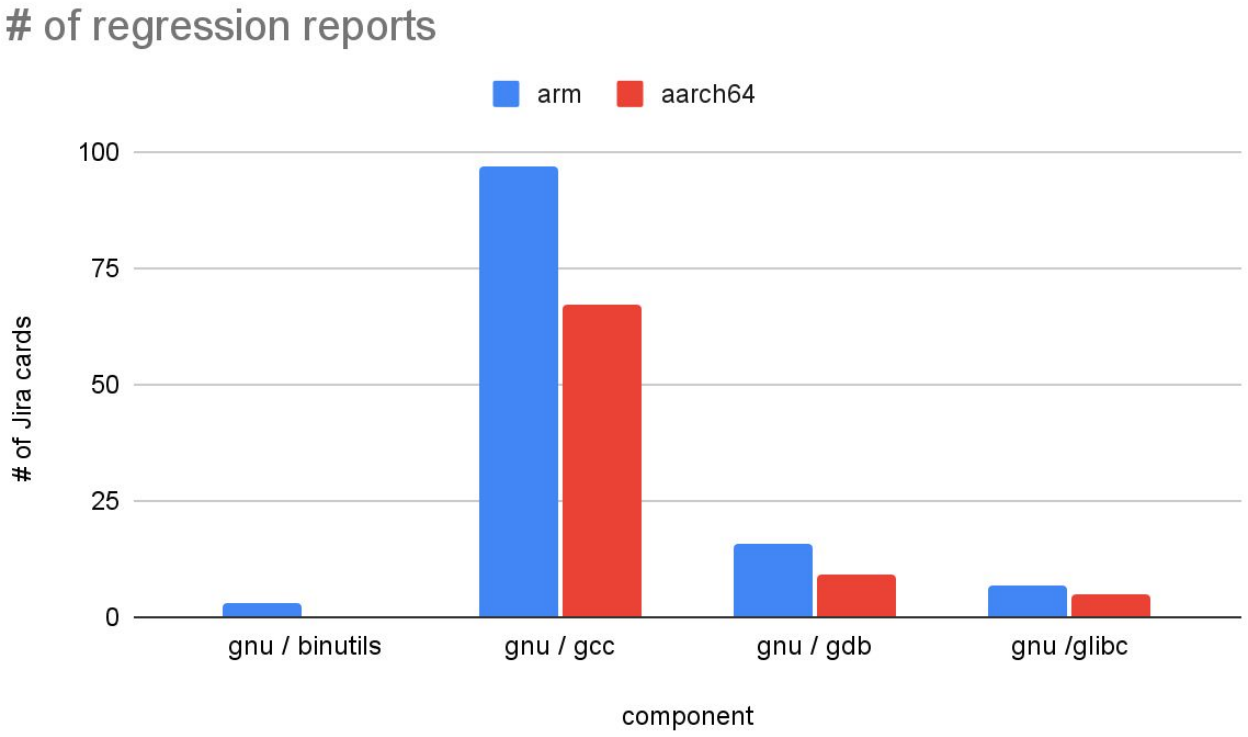
- Pre-commit CI not always taken into account by patch authors

- Possible improvements:
  - Reply to thread on the list to increase visibility
  - Policy change asking for revert?

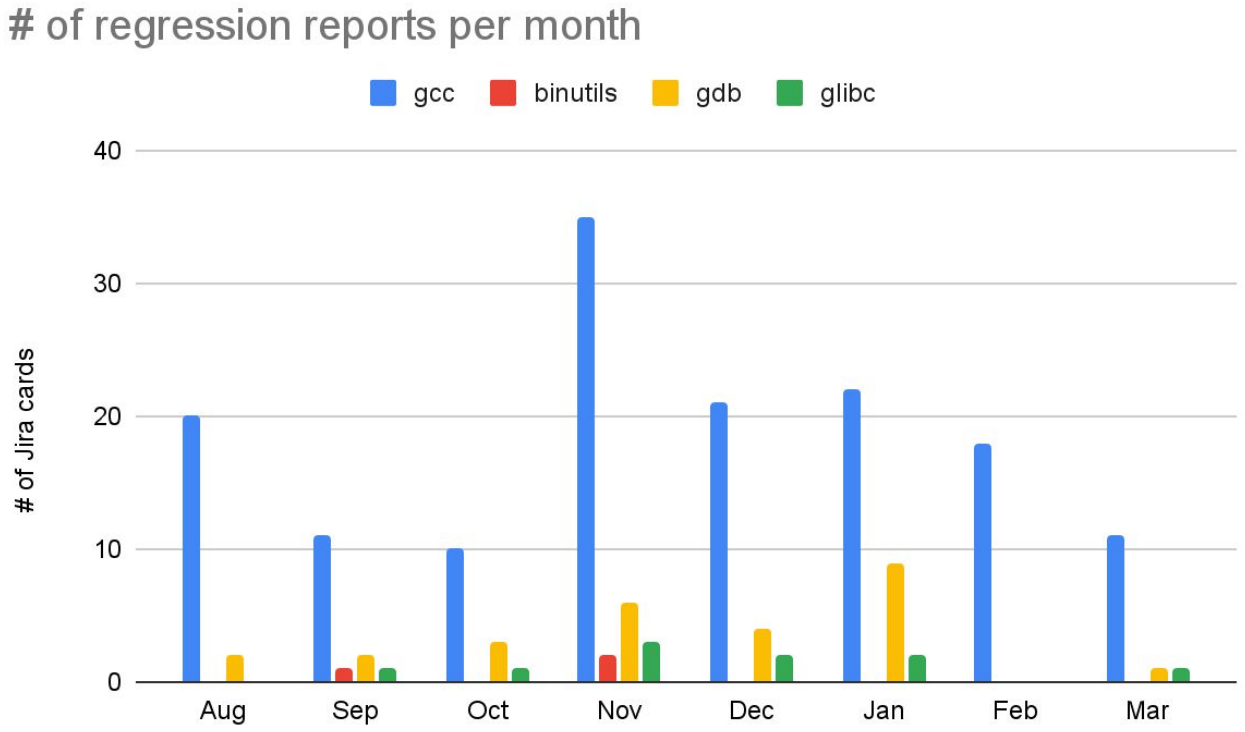
- Pre-commit CI and auto-regenerated files
  - GCC patches submission did not contain auto-regenerated files (`autoconf/automake`, but not only)
  - CI applies “incomplete” patches, leading to failures….
  - …. And to complaints
  - Worked to improve automation, but too complex to fully automate
  - Gathered arguments and **community agreed to a Policy change**
    - Patches should now be submitted with all auto-generated parts
    - Enables more pre-commit CI
    - Also gives reviewers more confidence

Statistics

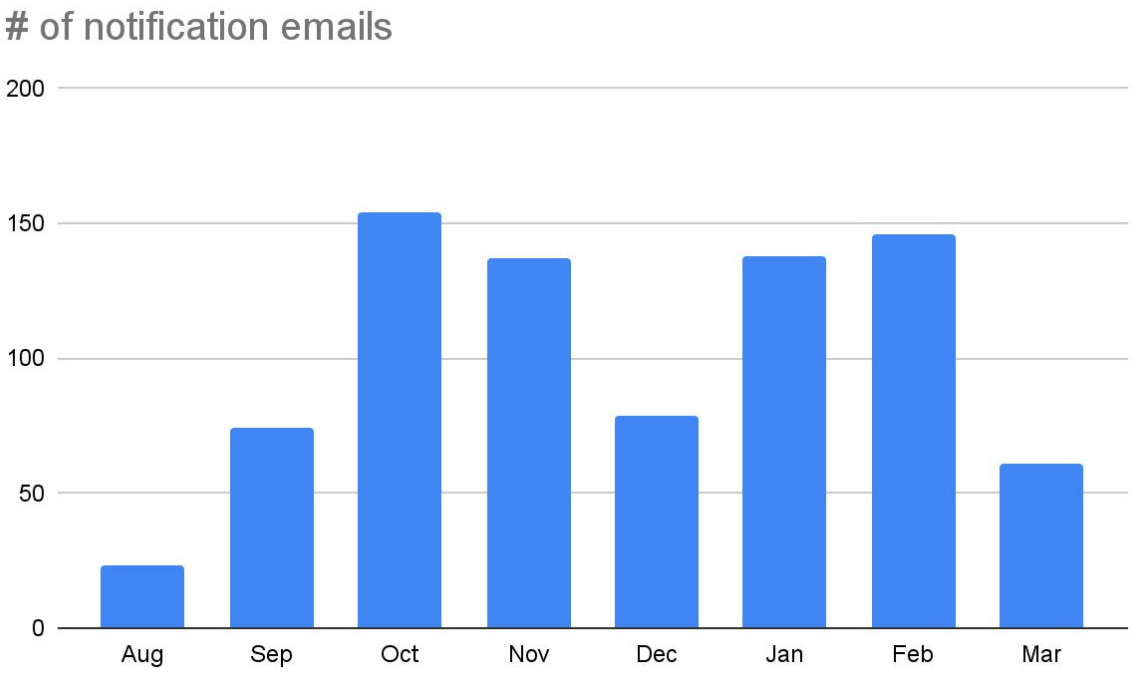
Post-commit CI stats between 2023-08-01 and 2024-03-30 (8 months)



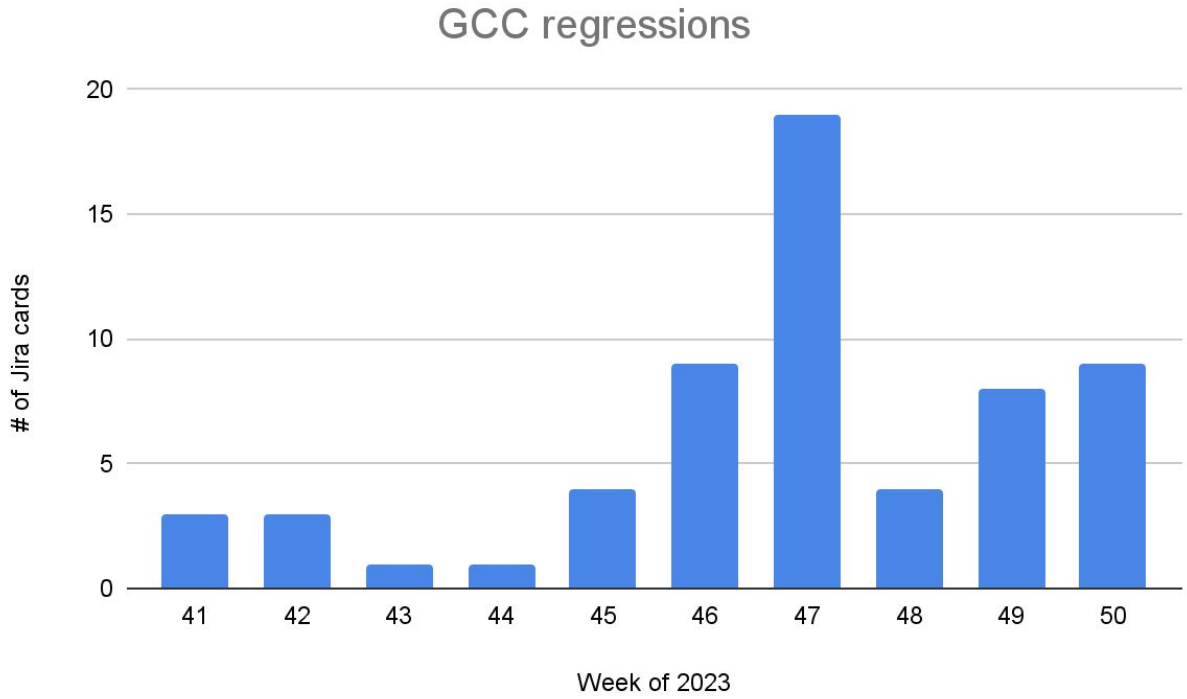
Post-commit CI stats between 2023-08-01 and 2024-03-30 (8 months)



Pre-commit notifications



That time we accidentally broke the pre-commit CI



Some quotes from the community

- Finally, [Patchwork] has been integrated with our CI systems (thanks Linaro!), so it can automatically pull reviews and run validations on them, then report the results back; often before I've even had time to look at the patch.
  — Richard Earnshaw – 23 Apr 2024

- For anyone reviewing this, I'm aware of the arm regressions (thanks again, linaro CI). I'm looking into them, but other reviews are still welcome!
  — Guinevere Larsen – 9 Apr 2024

- Bootstrapped and tested on x86_64-unknown-linux-gnu, will push if the linaro CI is happy.
  — Richard Biener – 16 Jan 2024

- Very many thanks (and a Happy New Year) to the pre-commit patch testing folks at linaro.org. Their testing has revealed that although my patch is clean on x86_64, it triggers some problems on aarch64 and arm. The issue (with the previous version of my patch) is that these platforms require a paradoxical subreg to be generated by the middle-end, where we were previously checking for truly_noop_truncation.
  — Roger Sayle – 31 Dec 2023

Linaro Connect
MADRID 2024 | MAY 12-17 2024

Thank you