Quick introduction to FLINT

How to compile and contribute

Albin Ahlbäck

LIX, CNRS, École Polytechnique

27th of January, 2025

FLINT development workshop, Palaiseau

What is FLINT?

- FLINT stands for **F**ast **Li**brary for **N**umber **T**heory.
- Written in C.
- Base functionality for integers, rational numbers, modular arithmetic, floating-point arithmetic, ball arithmetic, finite fields, ...
- Extends functionality to vectors, matrices, univariate polynomials, multivariate polynomials (including factorization), special functions, embeddings of finite fields, number fields, algebraic numbers, ...

News

Since the workshop in Bordeaux 2024, we released FLINT 3.2.0. This includes:

- New module `mpn_mod` for packed fixed-size modular arithmetic
- New module `nfloat` for packed fixed-precision floating-point arithmetic
- Major work on the `gr` modules
- Faster single-word modular arithmetic
- Improved low-level routines, mainly multiplication
- Streamlining test code, documentation fixes, bug fixes, CI improvements, ...

Interacting with FLINT

- FLINT provides a binary library (such as `libflint.so`) along with headers.
- Library can be linked and called in other software, such as in `Oscar.jl`:

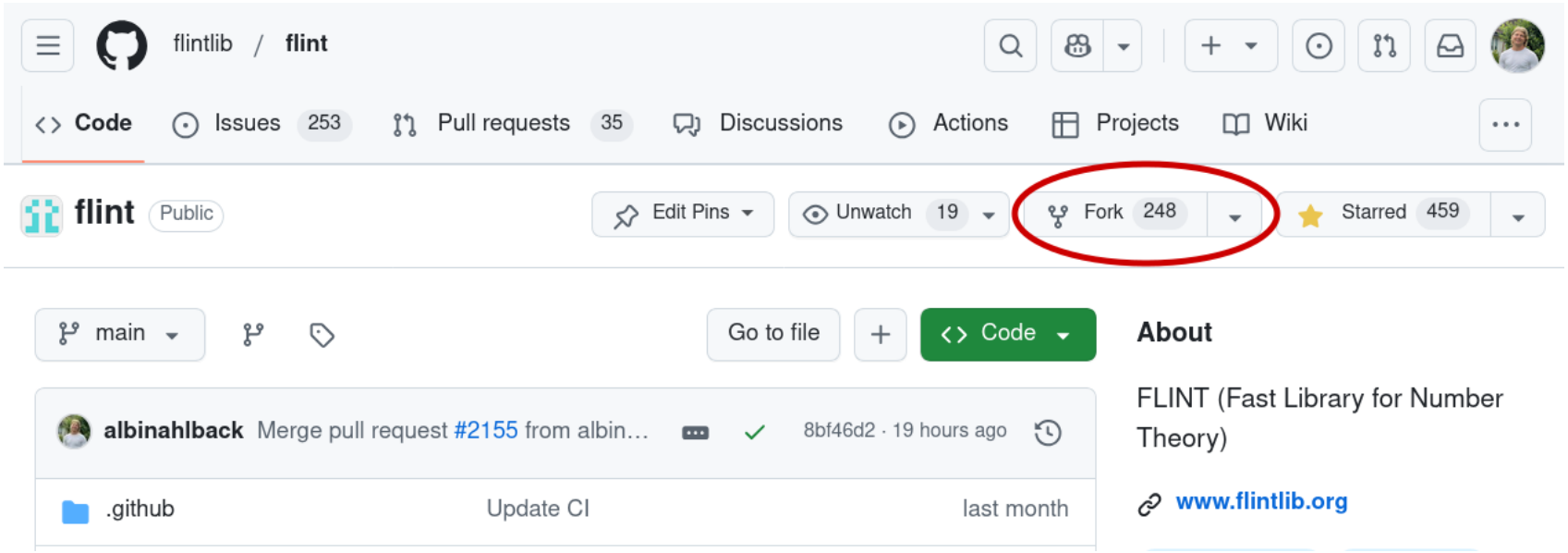
```
@ccall libflint.fmpz_add(z::Ref{ZZRingElem},
     x::Ref{ZZRingElem}, y::Ref{ZZRingElem})::Nothing
```

Useful links

- FLINT on Github: https://github.com/flintlib/flint/
- FLINT's website: https://flintlib.org/
- Zulip chat for FLINT: https://sagemath.zulipchat.com/#narrow/stream/408539-flint
- FLINT mailing list: https://groups.google.com/g/flint-devel

Working with Github

To contribute to FLINT, you probably want to create your own fork on Github:



Feel free to star it as well!

Pulling FLINT from official Git server

Assuming that you have forked FLINT on Github, you can set up FLINT on your local computer via

```
git clone git@github.com:MyUsernameOnGithub/flint.git
cd flint/
git remote add upstream git@github.com:flintlib/flint.git
```

Now your own remote repository is called `origin`, and flintlib's repository is called `upstream`.

Updating FLINT via Git

During the workshop, FLINT will be updated continuously, and you can become outdated quite quickly. To update your FLINT, write

```
git checkout main
git fetch upstream
git rebase upstream/main
git push
```

and now the `main` branch your local repository and your fork on Github should be up-to-date with the official repository's `main` branch.

Develop a new branch

You have a great idea and want to try it out, and eventually have it merged in FLINT. Then create a new branch by the following:

```
git checkout -b mybranch main
...  // edit some files
git add myfile1 myfile2 myfile3
git commit  // ensure that commit message is descriptive
git push -u origin mybranch
```

Now `mybranch` should in on your repository on Github, and you can create a pull request.

Prerequisites

On Unix-type systems (macOS, Linux, *BSD), we need the following:

- GMP and MPFR. On Debian:

  ```
  sudo apt install libgmp-dev libmpfr-dev
  ```

- GNU Make and GNU Autotools. On Debian:

  ```
  sudo apt install make autoconf libtool-bin automake
  ```

If you use Windows and need help with installation, please talk with Albin afterwards.

Configuring and building FLINT

- Generate configure script:

  ```
  ./bootstrap.sh
  ```

- Configure FLINT:

  ```
  ./configure
  ```

  Type `./configure --help` to see a full list of options.

- To build FLINT, run:

  ```
  make -j $(expr $(nproc) + 1)
  # $(nproc) is the number of threads on your system.
  # This builds FLINT the fastest.
  ```

Testing FLINT

- Testing FLINT:

  ```
  make check
  ```

  Can be done multithreaded by passing option `-j` to `make`.

- Examples of testing specific modules:

  ```
  make check MOD=fmpz
  make check MOD="fmpq nmod_poly"
  ```

Installing FLINT

- To install FLINT, run:

  ```
  make install
  ```

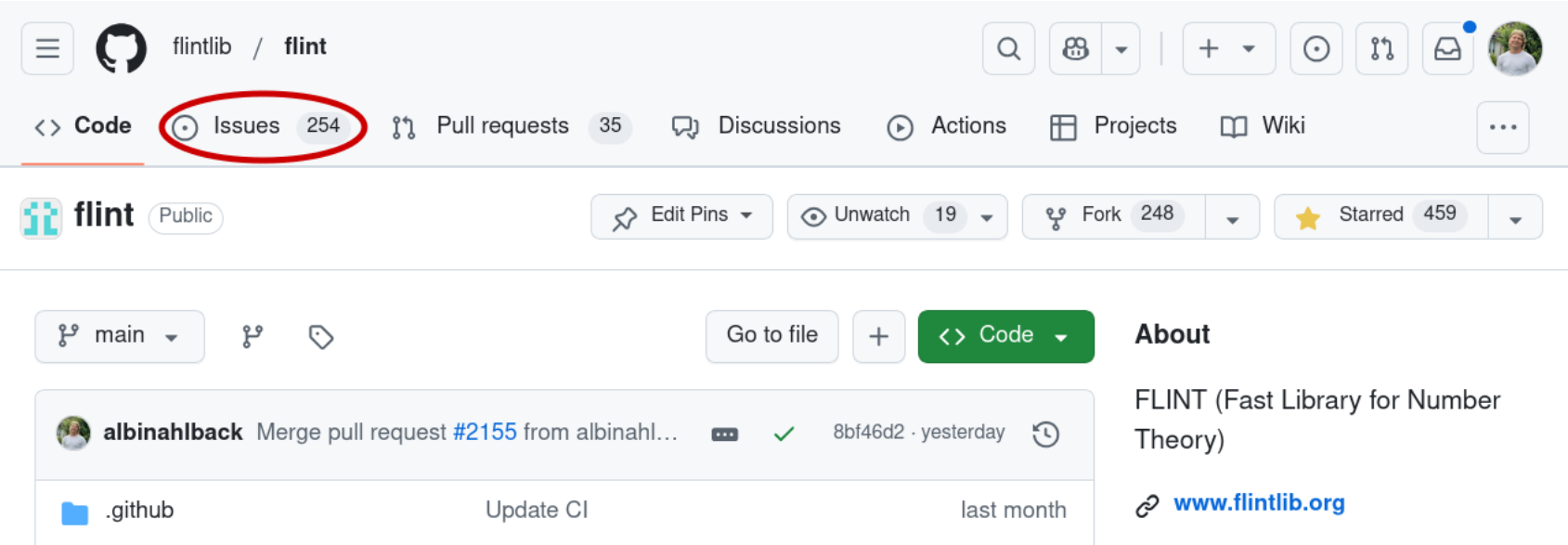
  The destination is set during configuration, with default being `--prefix=/usr/local`.

File layout

```
flint/
  doc/
    source/
      fmpz.rst  // Documentation for fmpz module
  examples/
    factor_integer.c
  src/          // Source code
    flint.h
    fmpz.h      // One header per module
    fmpz/       // Module's sources in its own directory
      add.c
      test/     // Test directory for this module
        main.c
        t-add.c
```

Reporting bugs

To report a bug in FLINT, preferably open up an issue at Github:



or you can send an email to the FLINT mailing list (link in the beginning of this presentation).

Some ideas for the workshop

- General development
- Finding bugs
- Bug fixes
- Improve the usage of FLINT in other software
- Improve documentation
  - Is docstring of X clear?
  - Does it agree with how people usually define it (say Mathlib)?
- Streamlining of FLINT – simplify and "coherentify" everything
  - Can test code be unified, yet still be expressive?
  - Documentation can definitely be more coherent! Do we really need n independent docstrings for addition of (exact) rings? Write a macro for, say, addition?