OpenHW Group

Proven Processor IP

Git 101 for CORE-V-VERIF

Mike Thompson

mike@openhwgroup.org

OPENHW GROUP™
PROVEN PROCESSOR IP

Contents

- A quick introduction to *git* and the way *git* and *GitHub* are used in OpenHW's core-v-verif repository.

Credits

- Thanks to the following members of the OpenHW Group for their contribution to these slides:
  - Alfredo Herrera, BTA
  - Wayne Beaton, Eclipse Foundation

git, GitHub, GitLab, Bitbucket

- "Git" is a revision/configuration control tool.
  - If you've ever used RCS, CVS, SVN, Clearcase, Perforce or Mercurial then you already get the general idea.
  - Written by Linus Torvalds (yes, *that* Linus Torvalds).
  - Free-and-open-source tool.
  - The Pro Git Book (https://git-scm.com/book/en/v2) is an excellent reference.
    - Doctor Google is the best git reference.
    - Both GitHub and GitLab have excellent documentation.

- GitHub, GitLab, Bitbucket are services built on top of git:
  - Typically provide repository hosting and browser-based user-interface.

TL; DR

- “Check-out” a working copy of a repository
  $ git clone https://github.com/openhwgroup/core-v-verif.git

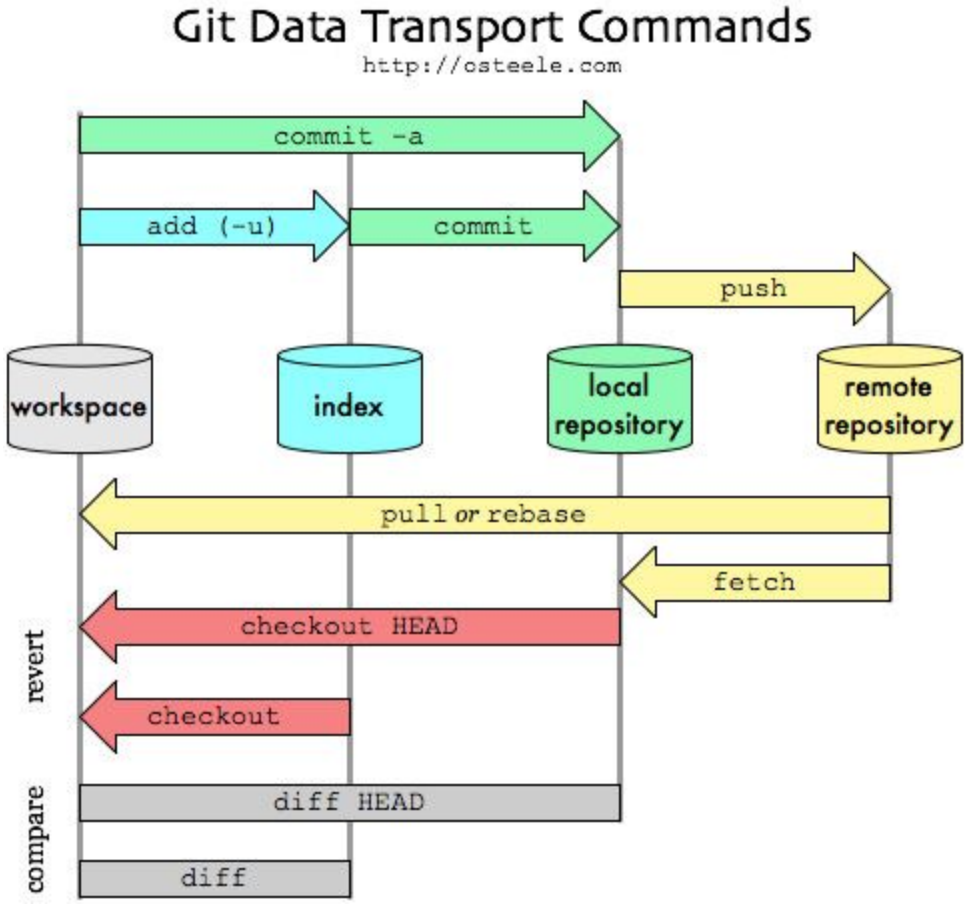
- “Check-in” new or modified files:
  $ git add modified_file.txt
  $ git commit -m ‘Fix issue #123’
  $ git push origin main

- Be careful!
  - The term “checkout” in git means something different than it does in other revision control tools.

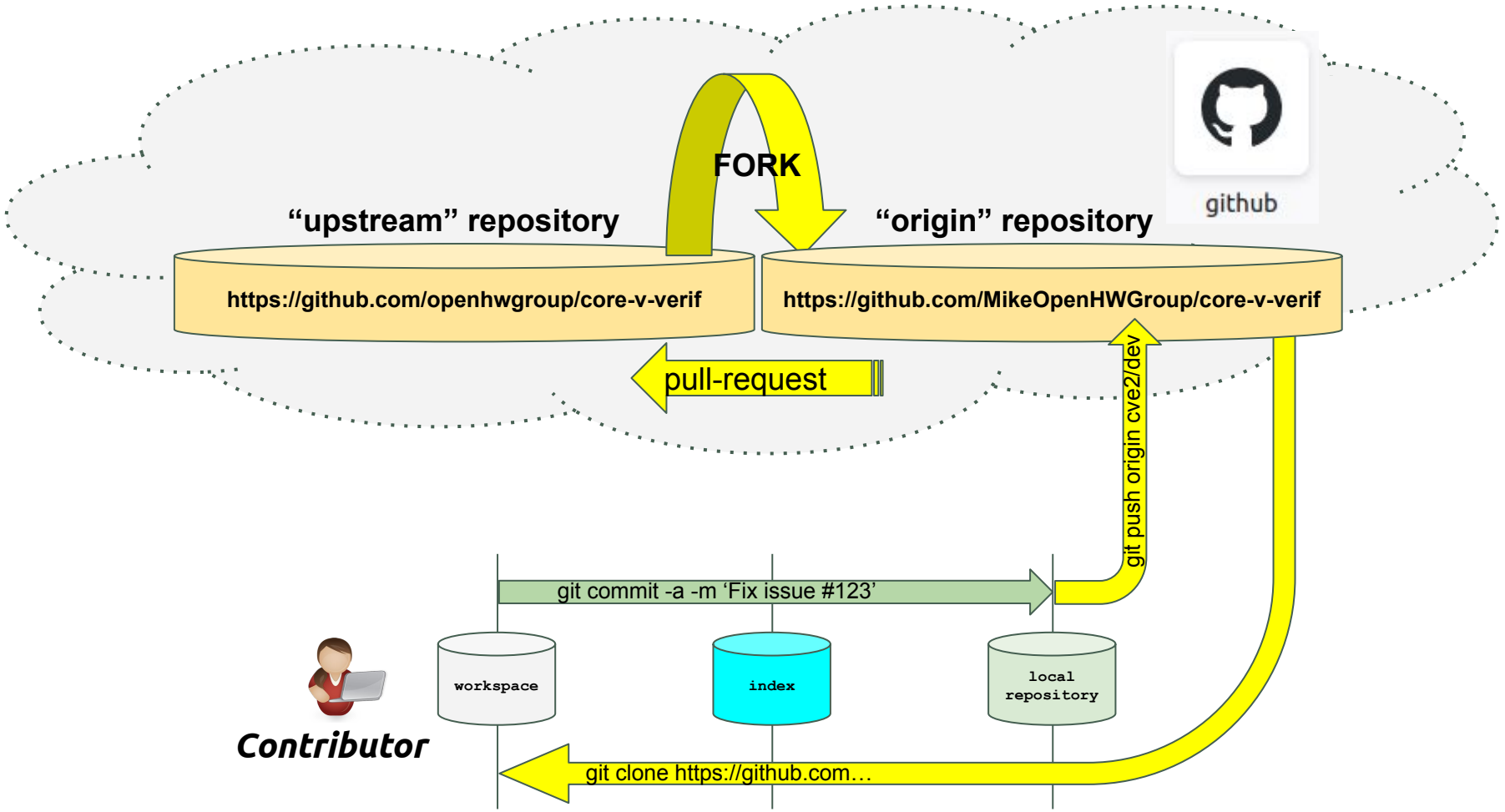
Repositories and Workspaces

- Typically, you will work with a *remote repository* that is used by a team to manage a single code-base.
  - This repo is typically called “origin”.
  - Note that “origin” is not special - there is no concept of a “central repository in git.

- The git clone command (not shown) populates a *local repository* and *workspace*:
  - You work with files on your workspace.



Working with Remotes



GitHub: Typical Usage Model

1. Browse to the URL of the Repo you want to work with:
   e.g. https://github.com/openhwgroup/core-v-verif.git
   - by convention we will call this repository the “upstream” repo.

2. Fork the repository:
   - by convention we will call this repository the “origin”.

3. Clone repository.

4. Make modifications as required.

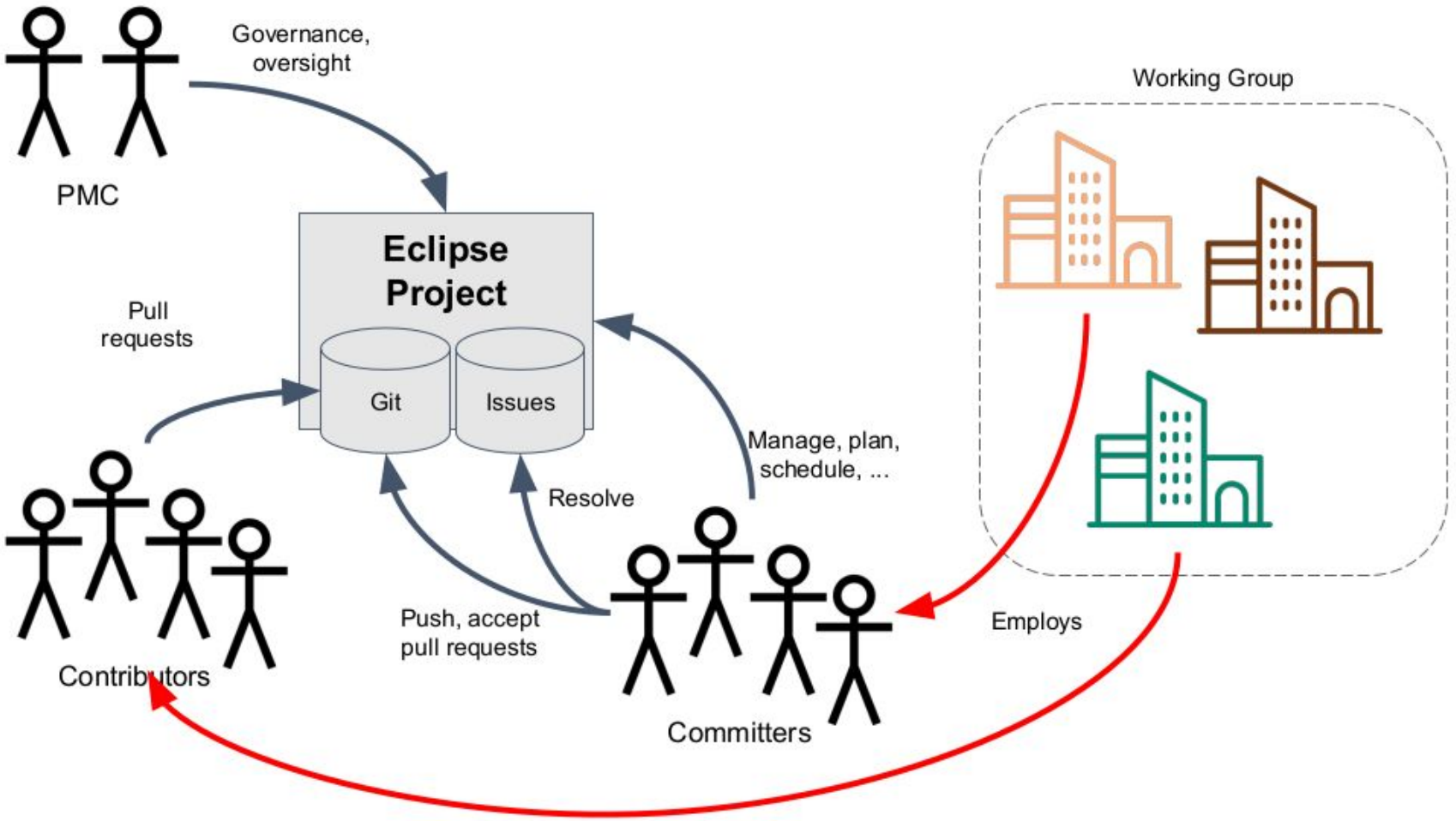
5. Push modifications back to the origin (your forked repository).

6. Create a pull-request from your repository to the origin to the upstream.

CORE-V-VERIF Usage of GitHub

- CORE-V-VERIF supports several “cores-under-verification”.
- Each of these cores works independently on their own set of branches:
  - cv32e40p/dev, cv32e40p/release, cva6/dev, etc.
- Typically, pull-requests are made to a “dev” branch.
- Most branches are covered by an automated CI flow.
- CORE-V-VERIF uses resources available in other repositories:
  - The RTL for the “core-under-verification” is the best example.
  - CORE-V-VERIF does not use git sub-modules for this.

The Eclipse Flow



Getting Started

- Create a Eclipse Account!
- Fork core-v-verif
- Use your browser to review MergeTest.md.
- Follow the instructions - I am eagerly awaiting your pull-request!

- You may also find GitCheats.md useful.

One More Thing…

- GitHub is moving to enforce ssh connections.
- This will impact users:
  - Git commands on the command-line will need to be of the form: git@github.com:openhwgroup/core-v-verif.git
  - You will need to register your public key with GitHub. https://docs.github.com/en/authentication/connecting-to-github-with-ssh/adding-a-new-ssh-key-to-your-github-account

Thank You