



New Rod Firmware repository, deploy system and workflow

Nico Giangiacomi

New Firmware repositories

New git group path:

https://gitlab.cern.ch/atlas-pixel/daq/pixelrod_firmware

The screenshot displays the GitLab web interface for the 'PixelROD_Firmware' group. On the left, a sidebar menu includes options like 'Group overview', 'Details' (selected), 'Activity', 'Contribution Analytics', 'Issues' (1), 'Merge Requests' (3), 'Kubernetes', 'Members', and 'Settings'. The main content area shows the group's details, including its name, ID (26019), and a description: 'Groups hosting Pixel ROD firmware for PRM, Master and Slave'. Below this, there are tabs for 'Subgroups and projects' (selected), 'Shared projects', and 'Archived projects'. A search bar and a 'Last created' filter are present. A table lists three subgroups: 'RodMaster', 'RodSlave', and 'RodPrm', each with 0 stars and creation dates of 2, 3, and 3 days ago respectively.

Subgroups and projects	Shared projects	Archived projects	Search by name	Last created
RodMaster				★ 0 2 days ago
RodSlave				★ 0 3 days ago
RodPrm				★ 0 3 days ago

RodPrm

Git repo path:

https://gitlab.cern.ch/atlas-pixel/daq/pixelrod_firmware/rodprm

Two main branches (protected, cannot push)

- 1) **master** → 2018 style branch + latest fixes (minor changes). **Tagged as V0.x**
- 2) **user/bologna/Firmware_UBP** → latest modification by Gabriele (major changes). **Tagged as V1.x** (timing failure)

Secondary branches

- **FixTiming** (from user/bologna/firmware_UBP) → empty, to be used to fix Firmware_UBP timing

RodMaster

Git repo path:

[*https://gitlab.cern.ch/atlas-pixel/daq/pixelrod_firmware/RodMaster*](https://gitlab.cern.ch/atlas-pixel/daq/pixelrod_firmware/RodMaster)

Two main branches (protected, cannot push)

- 1) **master** → 2018 style branch + latest fixes (minor changes). **Tagged as V0.x**
- 2) **New_Features** → latest modification by Gabriele (major changes). **Tagged as V1.x**

Secondary branches

- **newMasterTTCEmu** (from master)→ new TTC emulator (to be properly validated yet)

RodSlave

Git repo path:

https://gitlab.cern.ch/atlas-pixel/daq/pixelrod_firmware/RodSlave

Two main branches (protected, cannot push)

1) **master** → 2018 style branch + latest fixes (minor changes). **Tagged as V0.x**

2) **Firmware_separated** → calibration VS data taking + fix on terminations.

Tagged as V1.x

Secondary branches

- **gitSha** (from Firmware_separated) → attempt to get SHA for dataTaking FW without MB
- **NewMerger** (from Firmware_separated) → new slave merger for dataTaking
- **Firmware_SmartL1IDAlgorithm** (from Firmware_separated) → smart L1ID algorithm for Pixel and Slave dataTaking fw
- *user/ngiangia/newTimeout* → very old attempt to implement new timeout mechanism (not working)

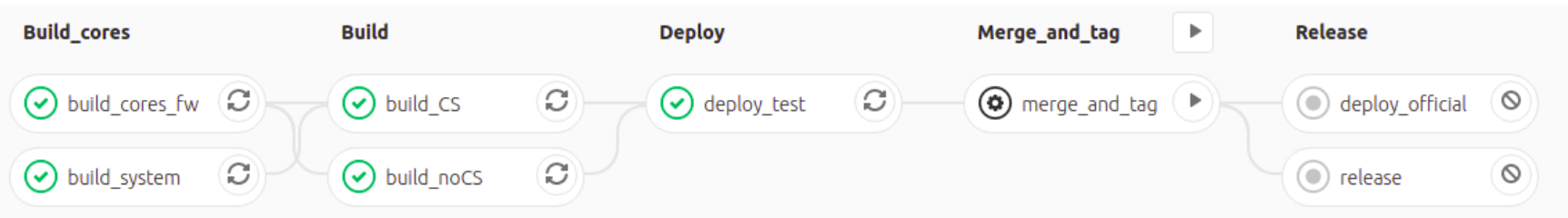
Workflow

When adding/modifying something:

- 1) **create new branch** (from the one that needs to be modified)
- 2) work normally
- 3) **push** your branch and create **merge request**
- 4) **Continuous Integration** builds firmware and deploy it to **cvmfs (test)**
- 5) **Test** the firmware
- 6) If tests are successful, **resume Continuous Integration stages** (merge and tag) to automatically **tag** the new firmware, create **release** and **deploy to cvmfs (official)**



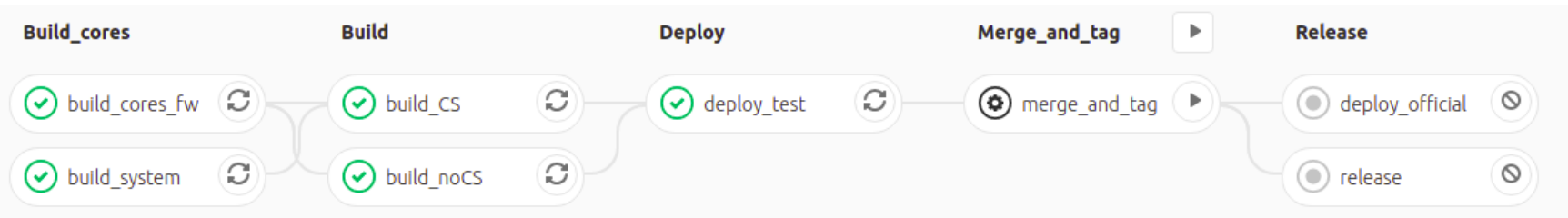
CI Stages



5 stages:

- 1) **build cores** → regenerates the fw cores. Precompiled cores (associated to their SHA) are stored in eos and retrieved to speed up this process
- 2) **build** → generates the fw (with CS and without CS). Fails if timing errors. CS firmwares CAN fail
- 3) **deploy** → firmwares are copied to **cvmfs** in: rod/firmware/rodXXXX/test/\$SHA/...
- 4) **Merge and tag** (to be **manually triggered** after testing fws) → automatically merges the branch (it should close the MR) and creates a new tag
- 5) **Release** (only after merge_and_tag) → creates **new release** with MR description as changelog and with **binaries**, copy binaries to **cvmfs** in rod/firmware/rodXXXX/official/\$tag/... (the very same firmware is copied)

CI Stages



5 stages:

- 1) **build cores** → regenerates firmwares (firmwares are stored in eos and retrieved by their SHA) are
- 2) **build** → generates the fw (firmwares CAN fail)
- 3) **deploy** → firmwares are copied to **cvmfs** in: rod/firmware/rodXXXX/test/\$SHA/...
- 4) **Merge and tag** (to be **manually triggered** after testing fws) → automatically merges the branch (it should close the MR) and creates a new tag
- 5) **Release** (only after merge_and_tag) → creates **new release** with MR description as changelog and with **binaries**, copy binaries to **cvmfs** in rod/firmware/rodXXXX/official/\$tag/... (the very same firmware is copied)

NOTE: the merged branch is not closed. Must be closed manually

RELEASES

RodMaster

- Project overview
 - Details
 - Activity
- Releases
- Repository
- Issues 1
- Merge Requests 1
- CI / CD
- Operations
- Packages & Registries
- Analytics
- Settings

atlas-pixel > PixelROD_Firmware > RodMaster > Releases

v1.1

Assets 4

Source code

"New deploy mechanism for New Features branch"

- binaries.tar.gz
- binaries.zip

459353d7 v1.1 Created 1 day ago by

v0.1

Assets 4

Source code

"New deploy procedure for RodMaster master branch"

- binaries.tar.gz
- binaries.zip

eeb954cd v0.1 Created 1 day ago by

Example: current RodMaster releases

v1.1: first release based on **New_Features** branch (v1.x)

v0.1: first release based on **master** branch (v0.x)