Next - Gen CI/CD Factory for Embedded Systems

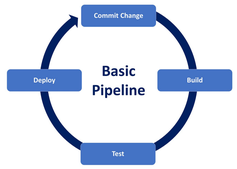
STUDENTS: Anuja Kalekar , Danielle Fung, and Karan Chauhan
INDUSTRY MENTOR: Adrien Leravat
FACULTY MENTOR. : Payman Aranshahi

Witekio

Problem Statement

Background:

CI/CD, or Continuous Integration Continuous Delivery, is a process to automate the lifecycle of an application, which includes building, testing, and deploying its software. This process plays an important role in saving developers time from manually completing tasks. It also quickly catches errors that occur from new changes made to an application's codebase.



Problem:

With CI/CD becoming standard practice in software engineering, there needs to be a solution for embedded systems that can support various testing frameworks used for embedded software.

Goal:

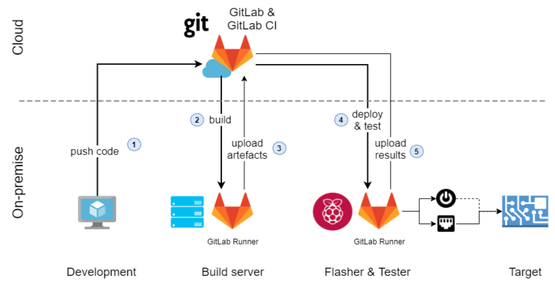
Design and implement a fully automated Continuous Integration/Continuous Delivery (CI/CD) solution for embedded systems for performing tests on target platform.

Requirements

- Flasher must:
  - Be controlled remotely from an online CI
  - Build an application's software
  - Flash software onto a target board
  - Run tests on the software
  - Return the results to the online CI
- The test systems available must include:
  - Witekio's Automation Lab
  - GoogleTest
  - Fuego
- The test systems and their dependencies must be installed and configured onto Docker images as appropriate so tests can immediately run
- Communication Requirements
  - Users shall communicate to the flasher via online CI
  - The flasher shall communicate to the target board via SSH
- Data Storage, Format, and Security Requirements
  - All data shall be stored in an online git repository
  - Any files flashed onto the target board must be deleted after tests are complete
- Power and switching control
  - The target board should be provided power when flasher is ready to flash and after testing it should be turned off using USB switch hub or controllable power supply

Implementation

System Overview



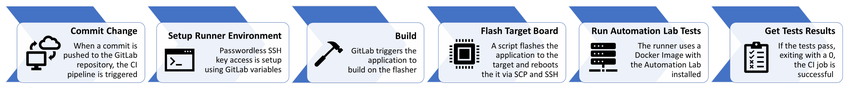
The system is structured with GitLab's CI Pipeline. Raspberry Pis are used as Runners, which build and flash applications to the target and runs various test.

Supported testing:
- Automation Lab
- Fuego
- GoogleTest

Image Generation
- Yocto-Mender

Automation Lab

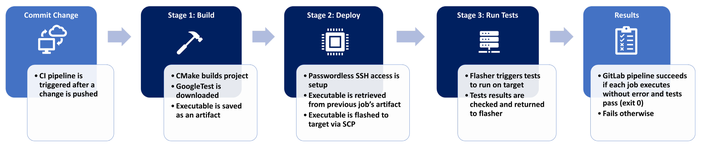
- The Automation Lab is Witekio's solution to automate various tests - including audit, non-regression, and functionality tests - for both hardware and software
- It is integrated into the pipeline by creating a Docker image with the Automation Lab and its dependencies installed. This image is used on the runner to perform tests.



Fuego

- Test framework used for running tests on target (linux) embedded platform.
- Useful for scheduling of automated tests on target using Fuego (ftc) commands with a docker image and Jenkins based UI, used on host machine for easability in configuration.
- Benchmark results and logs of target board can be obtained after successful completion of tests.

GoogleTest



- GoogleTest (or gtest) is a Google's unit testing framework for C++
- In this system's pipeline, CMake is used to compile the project. During the build's configuration step, GoogleTest is downloaded so it can be built with the rest of the project

Yocto-Mender

- For the custom raspberry pi image, warrior branch of Yocto is used.
- Parameters for mender server and other functionalities were added in the Yocto config file.
- A basic Yocto image was synced. The configurations for raspberry pi and mender were added run-time.
- Among the two images generated, *.mender image was used to deploy on the board.

Results/Conclusion

The goal of our project is to create a CI/CD automation that supports Witekio's Automation Lab in addition to other testing frameworks and configurations.

Outcomes:
- The automation is successfully setup on the boards using the Gitlab CI pipeline.
- Witekio's Automation Lab is dockerized and setup to use on a GitLab runner to perform tests on the target board.
- Fuego setup on the flasher using the docker image was successful along with adding the target board using the SDK toolchain of arm boards for cross compilation.
- Jenkins interface using Fuego for simplicity in creating and scheduling tasks was implemented.
- GoogleTest successfully downloads during CMake's build and can be used to compile a project tested with gtest.
- The custom image generation through Gitlab CI is successful and the image can be extracted afterwards. The extracted image can be deployed on the target board using mender server.

Future Work

- Including various target boards and tests for those boards.
- Using just Fuego commands in a script format that can be deployed on the GitLab runner instead of the Raspberry Pi as a flasher device.
- Integration of Fuego with GoogleTest and Automation Lab for scheduling any types of tests.
- For the image deployment, current way is to use Mender server i.e. upload an artifact there and choose an appropriate registered board for the deployment. This can be achieved by REST commands and integrated in the Gitlab CI pipeline.

Acknowledgements

Adrien Leravat - Industry Mentor
Merlin Webster - Automation Lab

References

1. https://docs.gitlab.com/ee/ci/introduction/
2. https://www.witekio.com/ready-to-use-iot-and-embedded-software-bricks/
3. https://elinux.org/images/e/ec/ELCE2015-LTSI_Test_Project_ibe.pdf
4. http://fuegotest.org/wiki/FrontPage
5. https://github.com/google/googletest
6. https://jumpnowtek.com/rpi/Raspberry-Pi-4-64bit-Systems-with-Yocto.html
7. https://docs.mender.io/2.3/apis/open-source/device-apis
8. System Overview Diagram - Adrien Leravat