Establish an automated testing lab for AGL

Jun 21, 2018
Automotive Linux Summit, Tokyo

Liu Wenlong
Nanjing Fujitsu Nanda Software Technology Co., Ltd (FNST)
Self Introduction

- Liu Wenlong(liuwl.fnst@cn.fujitsu.com)
- Linux Software Engineer (2015 ~ )
- AGL CIAT Member (2017 ~ )
  - focus on CIAT for AGL
  - Drivers test, LTP, Fuego and LAVA
Agenda

- Preface
- LAVA Quickstart
- Fuego Quickstart
- Fuego&LAVA solution
- LAVA tests in Fuego
- Future Work
Preface

CIAT (Continuous integration automated testing)

Existing tools,

- Autotest
  https://autotest.github.io/
- BuildStream
  https://wiki.gnome.org/Projects/BuildStream/
- LAVA
  http://validation.linaro.org
- Fuego
  http://fuegotest.org/
- KernelCI
  https://kernelci.org/
## Preface

Test tools comparison,

<table>
<thead>
<tr>
<th>Feature</th>
<th>LAVA</th>
<th>Fuego</th>
</tr>
</thead>
<tbody>
<tr>
<td>Build farm</td>
<td>No</td>
<td>Can be</td>
</tr>
<tr>
<td>Pre-packaged tests</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Jobs management</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Reporting(Results parser)</td>
<td>Yes</td>
<td>Strong</td>
</tr>
<tr>
<td>Distributed lab support</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Board management</td>
<td>Strong</td>
<td>Yes</td>
</tr>
<tr>
<td>System deployment(auto boot)</td>
<td>Yes</td>
<td>No</td>
</tr>
<tr>
<td>Web interface</td>
<td>Yes</td>
<td>Yes</td>
</tr>
<tr>
<td>Easy to extend</td>
<td>No</td>
<td>Yes</td>
</tr>
<tr>
<td>Easy to install</td>
<td>No</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Fuego+LAVA will be a good solution for our testing lab and AGL CIAT framework.
Let’s see how to establish an automated testing lab.

Refer to: https://schd.ws/hosted_files/aglammeu17/5c/jsmoeller_How_to_Write_Tests_for_the_AGL_HW_Test_Infra.pdf
LAVA Quickstart

LAVA installation and configurations,

- LAVA installation,
  https://www.validation.linaro.org/static/docs/v2/installing_on_debian.html
  Or install with docker https://github.com/kernelci/lava-docker (2018.4)

- LAVA configurations,
  - Add new devices
    https://www.validation.linaro.org/static/docs/v2/first-devices.html
  - Devices control(ser2net/PDU)
    https://www.validation.linaro.org/static/docs/v2/first-installation.html

- Do LAVA tests,
  - Write lava tests
    https://www.validation.linaro.org/static/docs/v2/developing-tests.html
  - Submit lava tests with “lava-tool”
    https://www.validation.linaro.org/static/docs/v2/lava-tool.html

Let’s have a look at the LAVA test definition to lay the groundwork for the later parts.
LAVA Quickstart

LAVA test definition (YAML file),

```
metadata:
  git.branch: agl-branch
  image.type: AGL
  ......

device_type: r8a7796-m3ulcb
  ......

actions:
  - deploy:
        to: nbd
        os: oe
        kernel:
          url: http://xxx/Image
        ......

  - boot:
        timeout:
          minutes: 10
        method: u-boot
        ......

  - test:
        definitions:
          - repository: https://git_qa_repo
            from: git
            path: test-suites/yocto-ptest.yaml
            name: yocto-ptest

        - test:
            definitions:
              - repository: ......
                run:
                  steps:
                    - lava-test-set start set-pass
                  ......
                    - lava-test-set stop set-fail
                from: inline
                name: health-test
                path: inline/health-test.yaml
```

We can easily do different tests on different HWs by customizing different sections.

[1] https://git.automotivelinux.org/src/qa-testdefinitions

For details, refer to: [https://www.validation.linaro.org/static/docs/v2/dispatcher-actions.html]

About those “sections” above, refer to:
[https://schd.ws/hosted_files/aglammmez17/5c/jsmoeller_How_to_Write_Testing_for_the_AGL_HW_Test_Infra.pdf]
LAVA connections,

Tester/developer side:

- LAVA Web interface

- Write/Submit lava tests
- Check test results
- Boards management
  - ...

LAVA HOST:

- LAVA master
- LAVA worker ...

Power Management

Serial management

Services: dhcpd/tftpd/nbd/nfs/…

Board farm:

Board Power Supply

serial/network

Relay/PDU

serial

DUT 01

DUT 02

……

serial

WAN ...

LAN ...

We can use LAVA in our auto-testing lab now.
LAVA Quickstart

LAVA test results,

Test results in LAVA

<table>
<thead>
<tr>
<th>ID</th>
<th>ID</th>
<th>Status</th>
<th>Actions</th>
<th>ID</th>
<th>Actions</th>
<th>ID</th>
<th>Actions</th>
<th>ID</th>
<th>Actions</th>
<th>ID</th>
<th>Actions</th>
<th>ID</th>
<th>Actions</th>
<th>ID</th>
<th>Actions</th>
</tr>
</thead>
<tbody>
<tr>
<td>336</td>
<td>336</td>
<td>Complete</td>
<td></td>
<td>335</td>
<td>Complete</td>
<td>334</td>
<td>Complete</td>
<td>333</td>
<td>Complete</td>
<td>332</td>
<td>Complete</td>
<td>331</td>
<td>Complete</td>
<td>330</td>
<td>Complete</td>
</tr>
<tr>
<td>336</td>
<td>336</td>
<td>Complete</td>
<td></td>
<td>335</td>
<td>Complete</td>
<td>334</td>
<td>Complete</td>
<td>333</td>
<td>Complete</td>
<td>332</td>
<td>Complete</td>
<td>331</td>
<td>Complete</td>
<td>330</td>
<td>Complete</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>336</td>
<td>336</td>
<td>Complete</td>
<td></td>
<td>335</td>
<td>Complete</td>
<td>334</td>
<td>Complete</td>
<td>333</td>
<td>Complete</td>
<td>332</td>
<td>Complete</td>
<td>331</td>
<td>Complete</td>
<td>330</td>
<td>Complete</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>328</td>
<td>328</td>
<td>Complete</td>
<td></td>
<td>329</td>
<td>Complete</td>
<td>330</td>
<td>Complete</td>
<td>331</td>
<td>Complete</td>
<td>332</td>
<td>Complete</td>
<td>333</td>
<td>Complete</td>
<td>334</td>
<td>Complete</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
LAVA Quickstart

LAVA features and diagram,

- Testing changes on multi-HW
- Boot testing, bootloader testing and system level testing
- Not a test lab
- Not a build farm
- Not a complete CI solution
- Not a set of tests
- Distributed test lab
- Support various bootloader

For details, please refer to, [https://www.validation.linaro.org/static/docs/v2/contents.html](https://www.validation.linaro.org/static/docs/v2/contents.html)
Fuego Quickstart

Fuego installation and configurations,

- Fuego installation steps

- Add a board
  [http://fuegotest.org/wiki/Adding_a_board](http://fuegotest.org/wiki/Adding_a_board)

- Add a toolchain
  [http://fuegotest.org/wiki/Adding_a_toolchain](http://fuegotest.org/wiki/Adding_a_toolchain)

- Add a test
  [http://fuegotest.org/wiki/Adding_a_test](http://fuegotest.org/wiki/Adding_a_test)

- How to write good tests?
  [http://fuegotest.org/wiki/Presentations](http://fuegotest.org/wiki/Presentations)
Fuego Quickstart

Fuego connections,

**Tester/developer side:**
- Jenkins web interface

**Jenkins**
- job management
- job trigger
- test results display
- etc

**Fuego Server:**
- Jenkins
- File server:
  - share test logs
  - documents
  - etc
- Fuego Framework
  - test tarball management
  - test scripts (build/run/analysis)
  - boards/toolchain configuration
  - etc

**WAN …**
**LAN …**

**Board farm:**
- Board 01
- Board 02
- …
- SSH/…
- ... Board Power Supply

**Users can,**
- Job management
- Run tests
- Check test results

We can use Fuego in our auto-testing LAB now.
Fuego Quickstart

Fuego test results report,

Jenkins

Test results in Fuego

Detailed testcase outputs
Fuego Quickstart

Fuego features and diagram,

- Highly customizable
- Unified test outputs
- Flexible test configuration
- Running tests in batches
- Pre-packaged tests
- Jenkins based
- Do tests with command lines
- Board setup is simple & flexible

Source from http://fuegotest.org/wiki/Architecture

For details, please refer to, http://fuegotest.org/wiki/Architecture
Fuego&LAVA solution

To do more with LAVA and Fuego,

- If we want to do LAVA+Fuego test, how?
  - Use LAVA as a board handler layer (existing feature)
    - Do LAVA job submit within Fuego
    - Enhanced board management for Fuego
    - LAVA board lab can be used in Fuego
    - Fuego has the ability to do OS upgrade

  - Use Fuego as LAVA test wrapper and build farm (new)
    - Do specified OS deployment/upgrades on real hardware automatically
    - With the help of Fuego, LAVA has the ability to make test results meaningful
    - Over 100+ Fuego pre-packaged tests can be used in LAVA
    - Fuego has the ability to do tests with remote board lab

Let’s see the existing “Fuego+LAVA” solution first.
Fuego&LAVA solution

Existing “Fuego+LAVA” feature (work flow),

**Fuego HOST**

- **Engine scripts:**
  - `<target>_prolog.sh`  
  - `pre_test: stage 1`  
    - (wait until board is up)
  - `pre_test: stage 2`
  - `build`
  - `deploy`
  - `run`
  - `get_testlog`
  - `post_test`
  - `cleanup`
  - `Processing`

- **Target board (DUT)**
  - **LAVA HOST**
    - Do hacking test:
      - [https://git.linaro.org/lava-team/hacking-session.git](https://git.linaro.org/lava-team/hacking-session.git)
      - Test: `hacking-session-xx.yaml`
    - **LAVA job submit**
    - Check if board is up
    - Check board status
    - Do some preparations
    - **Transfer test program**
    - **Execute test program**
    - **Test outputs**
    - **Cleanup, get syslogs**
    - **Stop hacking session**
    - **LAVA: finished until hacking stopped**

- **Target board (DUT)**
  - **Board: power on**
  - **LAVA: start hacking session**
  - **syslog(before)**
  - **Test program**
  - **syslog(after)**

**Target board (DUT)**

Refer to, [http://fuegotest.org/wiki/Architecture#fuego_test_phases](http://fuegotest.org/wiki/Architecture#fuego_test_phases)
Fuego&LAVA solution

Existing “Fuego+LAVA” feature (quickstart),
(example with “Intel UP2” board)

- Enable the following variables in board file:
  - TARGET_SETUP_LINK="fuego-lava-target-setup"
  - TARGET_TEARDOWN_LINK="fuego-lava-target-teardown"

- Add a “upsquared.lava” and a “upsquared.lava.yaml”:
  - file “upsquared.lava” holds env variables credentials and values for KERNEL, ROOTFS, LAVA_HOST, etc
  - file “upsquared.lava.yaml” holds a template for the lava test job

We can do Fuego+LAVA tests on Intel UP2 board with those steps above.

Refer to: https://bitbucket.org/tbird20d/fuego/commits/7d6a953455ada9b79d8a3a7ede523cd79647de5c
Fuego&LAVA solution

Existing “Fuego+LAVA” feature (test results), (example, “Functinal.bc” test on Intel UP2 board)

Console outputs
Fuego&LAVA solution

New “Fuego+LAVA” feature,

About the current “Fuego+LAVA” solution, it CANNOT,

- Not support remote board lab
- Cannot save test duration
- Cannot customize LAVA tests

We have those demands above, so how?

- Add a new job to generate yaml test files to support remote board lab
  - To use AGL “releng-scripts”[1] will be a nice choice
- LAVA tests customization(do multi-tests at one DUT boot)
  - add new tests in AGL qa-testdefinitions repo[2] to yaml test files

Let’s have a look at the new feature next.

[1] https://git.automotivelinux.org/AGL/releng-scripts
Fuego&LAVA solution

New “Fuego+LAVA” feature (work flow),

**Fuego HOST**

Engine scripts:
- `<target>_prolog.sh`
- `pre_test` (skipped)
- `build`
- `deploy` (skipped)
- `run`
  - Submit LAVA job
  - Get job outputs
  - Job status check
- `post_test` (skipped)

Processing

**LAVA HOST**

Run tests:
- Prepare artifacts from file server
- Fetch tests from upstream git repo
- Boot the target board (DUT)
- Deploy/run tests
- Test results collection
- ……

**Target board (DUT)**

- Board: power on
- LAVA: start testing
- test: health test
- ……
- LAVA: test finished
- Board: power off

[1] [https://git.automotivelinux.org/AGL/releng-scripts/](https://git.automotivelinux.org/AGL/releng-scripts/)
Refer to, [http://fugotest.org/wiki/Architecture#fuego_test_phases](http://fugotest.org/wiki/Architecture#fuego_test_phases)
Fuego&LAVA solution

New feature: add a new test to generate yaml test files,

- Add a new test, (named “Functional.lava” in this example)
  - Download the releng-scripts[1] to “Functional.lava/releng-scripts.tar.gz”

- LAVA tests generation and customization,

  ```
  $ cat /fuego-core/engine/tests/Functional.lava/fuego_test.sh
tarball=releng-scripts.tar.gz

  ...... 
  function test_build {
    # generate lava yaml test file.
    ./utils/create-jobs.py --machine $NODE_NAME --test ${FUNCTIONAL_LAVA_TESTS} ¥
    --boot ${FUNCTIONAL_LAVA_BOOT_TYPE} ¥
    --url ${FUNCTIONAL_LAVA_FETCH_URL} > test.yaml
  
  ...... 
  }
  ```

Now, AGL only support health-test, smoke,yocto-ptest in upstream repo[1]. But we can specify whatever other tests from github or somewhere and add it to yaml test file.

[1] [https://git.automotivelinux.org/AGL/releng-scripts/tree/templates/tests](https://git.automotivelinux.org/AGL/releng-scripts/tree/templates/tests)
Fuego&LAVA solution

New feature: jobs submit and results reporting,

- Job submit with lava-tool,

```bash
$ lava-tool submit-job https://${LAVA_USER}@${LAVA_HOST} test.yaml
# lava-tool can also be used to check the job status and get test outputs from LAVA.
```

- LAVA outputs analysis,

```
"case": "testcase",
"result": "result",
"set": "testset"
```

Easy to analyse and split the test outputs from LAVA.
Test name and result can be easily generated by the following key-value in the test outputs,
- “case”: testcase
- “result”: result
- “set”: testset
Fuego & LAVA solution

Test results with new “Fuego+LAVA” feature (Fuego side),

LAVA tests can be triggered and its testcases can also be parsed and displayed in Fuego.

Test case outputs in detail
Fuego&LAVA solution

Test results with new “Fuego+LAVA” feature (LAVA side),

LAVA test results (LAVA side) i.e. health-test

Results for test suite 0_health-test - Test Job 371

Fuego results in the next slide.
LAVA tests in Fuego

What is LAVA test,

The generic lava tests,

- Usually hosted in git repo[1]
  - yaml file describing the (set of) test(s) and scripts to execute the test(s)[2]
- Tests from,
  - from inline/git

How can we do Functional/Benchmark tests with LAVA? i.e. Functional.LTP.
- Scenario 1: Pre-installed LTP on DUT (i.e. agl-demo-platform-qa)
- Scenario 2: LTP not installed on the DUT (i.e. agl-demo-platform)

Different solution will be used for different scenarios. Let talk about this in the next slide.

[1] https://git.automotivelinux.org/src/qa-testdefinitions
LAVA tests in Fuego

Functional/Benchmark tests for LAVA. (Example, Functional.LTP)

- About scenarios 1 & 2, yaml test files from git/inline can be as below[1],

```yaml
......
params:
   TESTS: math
......
run:
   steps:
   - if [ ! -e /opt/ltp ]; then wget FUEGO_BUILD_SERVER/TARBALL & & tar --strip-components=1 -xzf TARBALL; else cd /opt/ltp; fi
   - ......
```

The TARBALL above is the pre-built LTP binaries(built in Fuego with ftc tool).

- About scenarios 2,
  - test from tar will be a better solution,
    BUT, LAVA said “Support is planned for tar and url”[2].

  - LAVA transport support will also be a good solution(more Fuego style) (WIP)

[2] https://www.validation.linaro.org/static/docs/v2/actions-test.html#from
LAVA tests in Fuego

LTP test with “Fuego+LAVA” solution (Fuego side),

LTP Test results
LAVA tests in Fuego

LTP test with “Fuego+LAVA” solution (LAVA side),

Results for test suite 1_LTP-test - Test Job 291

<table>
<thead>
<tr>
<th>Name</th>
<th>Test Set</th>
<th>Result</th>
<th>Measurement</th>
<th>Units</th>
<th>Logged</th>
<th>Bug Links</th>
</tr>
</thead>
<tbody>
<tr>
<td>abs01</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
<tr>
<td>atof01</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
<tr>
<td>float_bessel</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
<tr>
<td>float_exp_log</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
<tr>
<td>float_iperb</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
<tr>
<td>float_power</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
<tr>
<td>float_trigo</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
<tr>
<td>fittest01</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
<tr>
<td>fptest02</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
<tr>
<td>nextafter01</td>
<td>—</td>
<td>✔ pass</td>
<td>—</td>
<td>—</td>
<td>05/16/2018 2:56 p.m.</td>
<td>[0]</td>
</tr>
</tbody>
</table>

LTP test results
Future Work

About LAVA+Fuego:

- Run build phase separately and make tar for built artifacts
- Avoid false positive for test outputs from LAVA
- Watch the new features about LAVA.
- Take part in “LAVA Transport Support” related works
  (Tim Bird/Daniel Sangorrin/Jan-Simon are working on this)

OTHERS:

- New tests for AGL qa-testdefinition
- Benchmark test results displayed in html table
- Fuego failed tests investigation for R-Car M3 and Intel UP2
- Kernel LTSI test(BSP tests, drivers tests, OSS tests)
Introduction of Demo showcase

Welcome to our Demo Showcase…

Fuego and LAVA HOST

Hub

Screen

Internet

Relay

Targets

Fuego and LAVA HOST
Thank you!

liuwli.fnst@cn.fujitsu.com
shaping tomorrow with you