Status of GENIVI LAVA Automated Test

Automated Testing, a Collaborative Approach for the Industry

Stephen Lawrence (Renesas), GENIVI BIT Lead | October 2020

GENIVI®

Agenda

- Short recap
  - Why shared testing upstream matters
  - How GENIVI is contributing
- Update on the GENIVI Automated Testing Board Farm
  - Challenges of Android testing and solutions
- The road ahead
- Q&A and discussion

Shared testing upstream

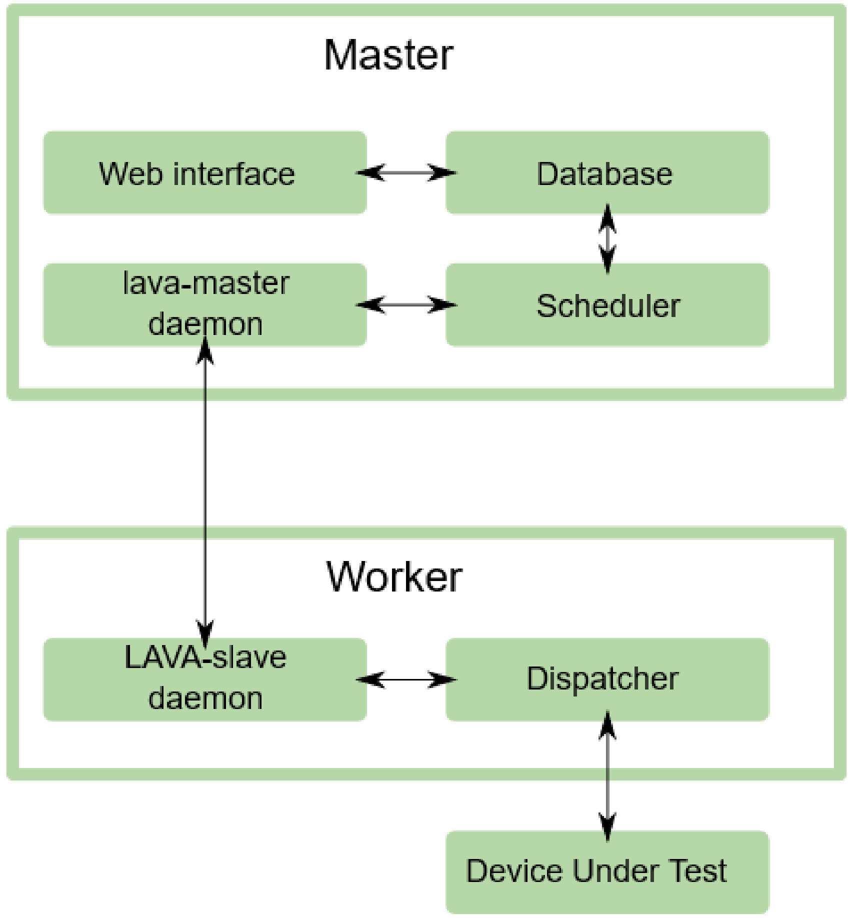
- OSS included in company-internal testing anyway so why share?
  - Pooling resources creates ability to test wider variation of versions and configuration than is normally done in a production or internal platform project (e.g.: Kernel CI)
  - In a complex stack trying to cover everything internally is very difficult
  - Development of test tools and test cases is time consuming and costly.
- Conclusions
  - Ability to look upstream for test results is a stronger basis for development
  - This is complimentary to your in-house testing
  - See my longer presentation at the last conference for a more detailed discussion of why this is really important

Automated test initiative: start-up recap

- Working mode: make a start, be flexible and open to collaboration with other orgs
- LAVA based test system to be connected to GENIVI GoCD CI (and other CI as needed)
  - Distributed system in wide use.
  - Strong support for complex deployment use cases on embedded hardware.
  - Proven to scale to large deployments.
  - Designed to automate validation during development
- Will use it for CIAT test of future GENIVI code emerging from Multi-OS

Automated test initiative: LAVA

- What is LAVA?
  - System for deploying OSs onto physical and virtual h/w to run tests.
  - Designed to automate validation during development
  - Wide device support
  - Extensive feature list
  - See Overview in LAVA documentation for full details
- Architecture
  - A LAVA instance consists of two primary components
    - LAVA Master (control server)
    - LAVA Worker (execute tests on boards) for QEMU and automotive hardware
  - YAML based test job descriptions



Automated test initiative: Last summit..

- Has been live and stable for some time now.
- Genivi LAVA Master (server)
- Genivi LAVA Worker (slave)
  - Renesas are hosting a lab currently containing the following DUTs (Device Under Test):
    - QEMU
    - R-Car M3 Starter Kit
    - R-Car H3 Starter Kit with Kingfisher expansion board fitted
  - More Workers/labs welcomed..
- Configuration
  - Running in Docker containers created using lava-docker from Kernel CI project
  - Leveraged work occurring in embedded industrial Civil Infrastructure Platform (CiP)
- OS support
    - Successfully proved Linux test cases using meta-ivi-test unit tests
    - Next step was investigating Android using the upcoming support for Android host tools in Docker containers

Challenges of Android testing and solutions

- Not typical approach of connect to terminal on DUT and directly execute tests
- Android includes host tools for flashing with fastboot and DUT control/test with ADB
- Some challenges:
  - Need way to handle different host setups and execute more than a terminal
  - DUT can be restarted multiple times presenting discovery challenges to host
- Solutions in LAVA:
  - LXC (just deprecated) and Docker (first introduced in v2020.02 and improved in later releases)
  - We use Docker
    - Create Docker container <foo> containing the host tooling
    - LAVA job executes <foo> container on host which communicates with the DUT

Status

- Life on the bleeding edge
  - Integrated and tested new LAVA Docker features as released (some pre-release)
  - Contributed back to lava-docker and LAVA projects with bug reports and code updates
  - Currently running LAVA 2020.07+ on LAVA Worker. Plan to update after summit.
- Initial AOSP support complete
  - Can flash AOSP using fastboot
  - Control of DUT via ADB
    - Execute shell commands, transfer files etc.
    - Example: use ADB to query Android for system start, capture screen and transfer capture.
- Completing integration of containerised AOSP build for reference systems into CI
- Next
  - Development of tests
  - CI pipelines to execute tests on AOSP builds and capture results

Thank you!

Visit GENIVI:

http://www.genivi.org

http://projects.genivi.org

Contact us:

help@genivi.org

GENIVI®