Linaro connect Hong Kong 2018

HKG18-TR10: Best Practices For Getting Devices Into LAVA

Steve McIntyre

Agenda

- Introduction
- LAVA
- CI
- Device Requirements for Automation
- Recommendations

Introduction

Over the seven years of automation in Linaro, the LAVA team have identified a series of automated testing failures which can be traced to decisions made during hardware design or firmware development.

Here is a summary of our experience. The aim is to provide background information about why common failures occur, and recommendations on how to design hardware and firmware to reduce problems in the future.

We describe some device design features as hard requirements to enable successful automation, and some which are guaranteed to block automation.

History of LAVA

Linaro created the LAVA project in 2010 to automate **testing of software** using **real hardware**

Linaro

Automated

Validation

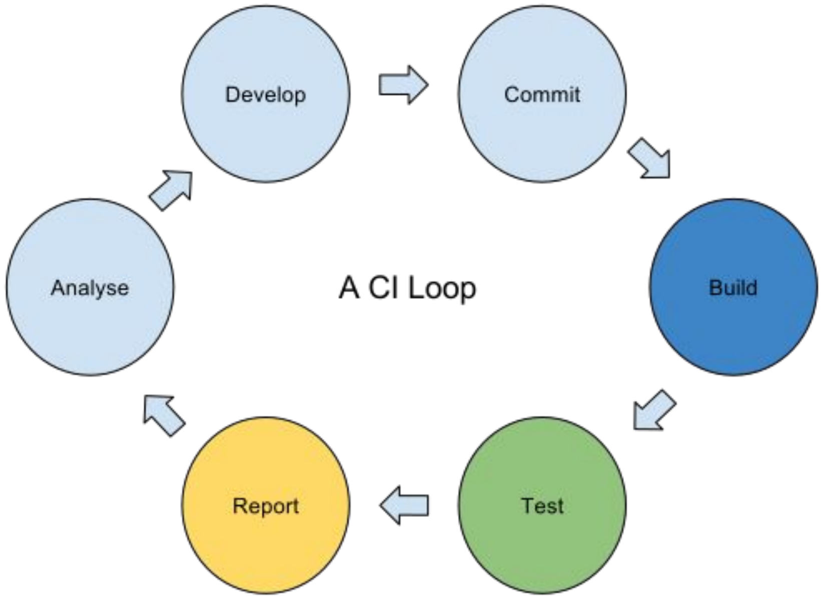
Architecture

Hardest part of the job has always been integrating new device types

Issues relate to hardware design and firmware implementations

- Millions of test jobs
- 100+ **very different** types of devices
  - ARM
  - x86
  - emulated
- Varied primary boot methods
  - U-Boot
  - UEFI
  - Fastboot
  - IoT
  - PXE
- 150+ devices in the Linaro lab, covering more than 40 different device types
- MultiNode and VLAN support

The CI process



A developer makes a change; that change triggers a build; that build triggers a test; that test reports back to the developer how that change worked. When looking for failures, beware:

- **False negatives**: not enough of the software is fully tested, or if the testing is not rigorous enough to spot all problems.
- **False positives**: the test itself is unreliable because of the test software, the test infrastructure or the test hardware.

Goals for LAVA

LAVA is one part of the CI loop - it provides device automation for testing. To work well, we have to consider:

- Reliability in the CI loop
  - Reliable software
  - Reliable infrastructure
  - Reliable test devices
- Scalability of the CI loop
  - Lots of devices
  - Lots of tests
  - In parallel
- Cost
  - Minimise expensive hardware
  - Minimise expensive admin intervention

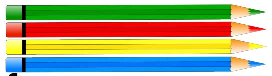
Device Requirements for Automation

- Reliability
- Uniqueness
- Scalability
- Deployment

Reliability

- Test jobs need to run reliably
  - Intermittent faults easily consume months of engineering time
  - False positives reduce engineer trust
- Infrastructure failures need to be identified correctly
  - Distinguish between infrastructure and test failures
- Infrastructure failures need to be minimised
  - 3 or 4 test job failures per week is too many
  - Devices may each run hundreds of test jobs in sequence
  - Do not let failures cascade
- All devices need to reach the minimum standard of reliability
  - It is not possible to automate all devices
  - It is not always possible to test all use cases of supported devices
  - Poor thermal control and stability issues under load will lower reliability
- Continuous Integration strategy remains the same for all devices
  - The problem is the same, regardless of underlying considerations
  - If the CI loop is not reliable, the work is pointless

Uniqueness



- Every unit must expose a distinct identifier across all exported interfaces unique to the DUT - can be set later by admins OTP or write protected
  - Serial, fastboot, MAC, USB mass storage
- Identifiers must be independent of the software deployed on the device
  - Separate method of modifying identifiers from normal deployment / flashing tools
- Identifiers must be stable across multiple reboots and test jobs
  - Identifiers generated randomly at boot time are never suitable
- USB serial connections
  - If multiple UARTs exposed, all identifiers must be programmed to be unique
- MAC
  - Should be unique already, if modifiable then only by admins

What works on the developer’s desk is not sufficient when connected in a rack with other similar devices.

Scalability

- Homogenous test platform using heterogeneous devices
    - Scalable infrastructure
- Do **not** complicate things
    - Try to avoid extra customised hardware
    - Relays, hardware modifications and mezzanine boards all increase complexity
    - More complexity raises failure risk nonlinearly
- Avoid unreliable protocols and connections
    - e.g. WiFi is nota reliable deployment method, especially inside a large lab with lots of competing signals and devices
- Not demanding enterprise or server grade support
    - But automation cannot scale with unreliable components
    - Server support typically includes automation requirements as a subset: RAS, performance, efficiency, scalability, reliability, connectivity and uniqueness
    - Automation racks have similar requirements to data centres - must work reliably at scale

Deployment

- Must provide a reliable and automated method to deploy, boot and test new software
  - e.g. automatic deployment over the network using TFTP
  - Manually writing software to SD is not suitable for automation
- Must always be able to interrupt a device at a level lower than the software being tested
  - e.g. to test U-Boot, must be able to deploy U-Boot without relying on a currently working U-Boot.
  - Consider how the device can be recovered automatically
- Automation typically requires controlling the device over serial
  - Not via a touchscreen or other human interface device
  - Not possible to drive firmware and kernel through a graphical interface
  - Parsing graphical menus is difficult and likely to be unreliable
- May not be able to automate lower levels - chain load?
- Consider how deployment tools interact

Recommendations

- Some ideas to consider during device design:
  - Hardware
  - Firmware
  - Overall
- Recommendations, not hard and fast rules!
  - Every device is different
  - Not every device is destined to be automated
  - If you want to automate things, there are some obvious ideas
  - And there are some that only come after hard-won experience
- Production hardware is often unsuitable for automation
  - Typically lacks connectivity or includes features which need workarounds
- Testing will typically work differently to expected customer usage
  - Will stress different interfaces and methods
  - May break all your assumptions

Hardware Design for Automation

- Keep It Simple...
- Serial Ports
  - Either real serial ports connecting to a serial console server, or **good** USB Serial
- Remote power control
  - Non-removable batteries are problematic
  - Immediate boot when power is applied, and shut down cleanly when removed
- Standard interfaces and form factors
  - Physical connections like RJ45, USB
  - Electrical specifications like power supply and serial voltage
- Onboard Networking
  - Physical ethernet with unique, stable MAC
- Storage reliability
  - Hundreds of test jobs per device may reduce the useful life of the device
- Protect essential components
- Don't over-optimise for board cost!

Firmware Design for Automation

- Adopt existing solutions where these exist
  - Makes it easier to work with the device
  - Do **not** write your own bootloader
- Make your firmware robust against mistakes
  - Do not depend on settings or data written by users
  - Make it easy to recover
- Serial support
- Ensure connectivity is available in the firmware
  - Firmware and bootloader drivers for networking and storage
- Consider chain loading
- Unique and stable identifiers for all exposed interfaces
  - e.g. stable MAC address
- Unique and stable meanings for settings
  - For both hardware and software

Overall Design for Automation

- Design for Testing
- Consider reliability from the beginning
- Avoid NIH - try to avoid custom tools
  - Push support for your devices upstream into existing tools
- BMC support?
  - Has a cost, but can make a huge difference for some types of device
- **Consider how users are going to use your hardware**
  - Might be **very** different than your expected use cases!

More information about LAVA

https://validation.linaro.org/static/docs/v2/

Linaro connect Hong Kong 2018

Thank You

#HKG18

HKG18 keynotes and videos on: connect.linaro.org

For further information: www.linaro.org