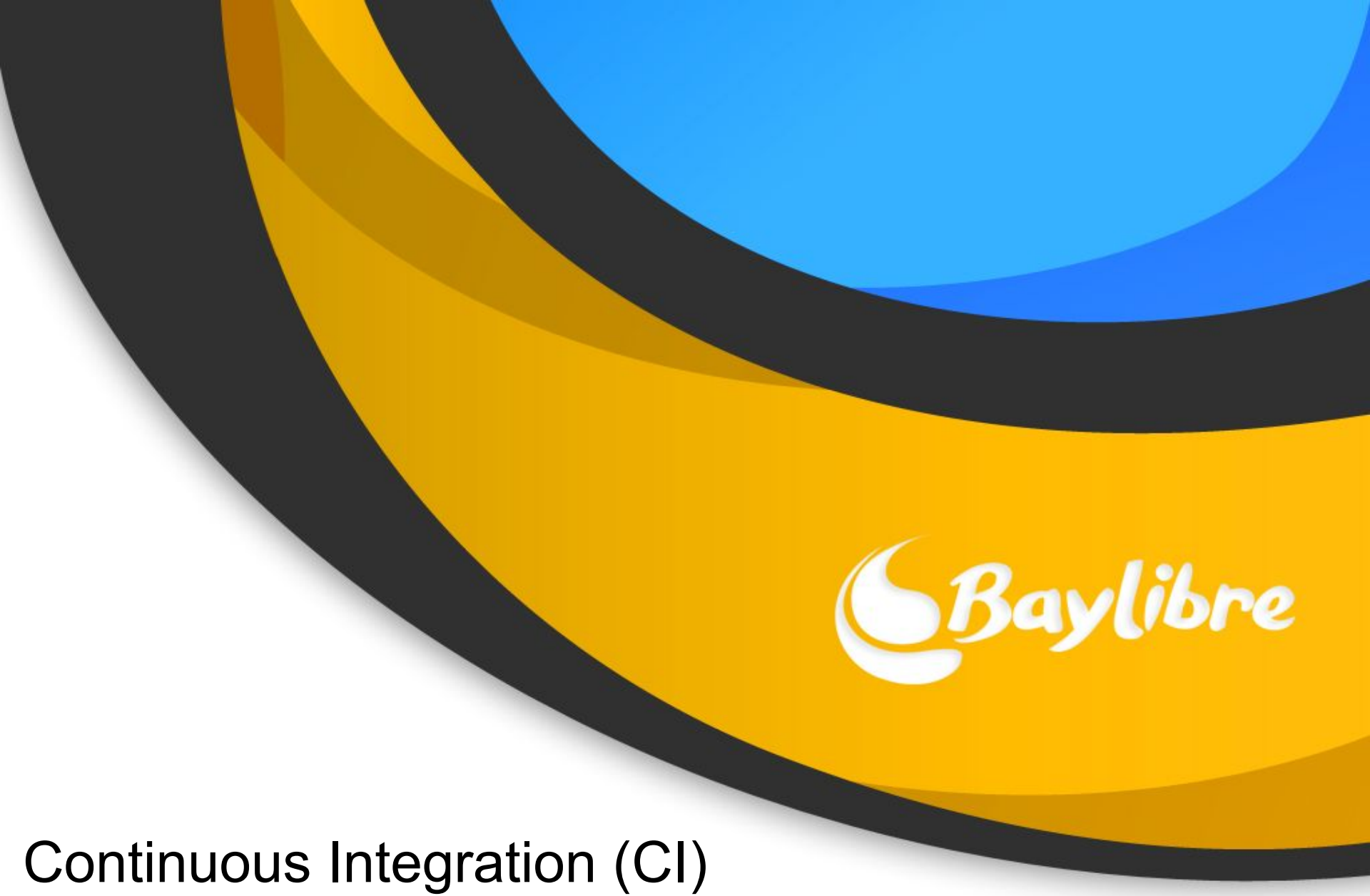
Introducing The “Lab in a Box” Concept

AGL AMM Dresden, October 2017

Patrick Titiano - Kevin Hilman, Baylibre.

Agenda

- Continuous Integration (CI)
  - Why so serious?
  - Typical CI Loop
  - LAVA, KernelCI & AGL CIAT Loop
- The “Lab in a Box”
  - Motivations, Challenges & benefits
  - Implementation
  - Success?
- Conclusions
- What’s next?
- Q & A



Continuous Integration (CI)

CI?

- Definition: the practice of merging all developer changes to a shared mainline several times a day, in combination with automated tests.
  - Introduced by Grady Booch (UML) in 1991, then adopted by Extreme Programming (XP) methodology

- Purpose: prevent integration problems, referred to as "integration hell"
  - Run tests in a given branch and verify they all pass before committing to the mainline
  - Use build and test servers to automatically run the tests periodically or after every commit, and report results to the developers.
  - Include unit / integration / static / dynamic tests as well as power / performance profiling

* Source: https://en.wikipedia.org/wiki/Continuous_integration

Why So Serious?

- Linux Kernel:
  - 4316 developers from 519 companies
  - Everyday:
    - 10 000 lines added
    - 2500 lines removed
    - 2100 lines modified
  - Every hour:
    - 8.5 changes per hour
- All Incoming patches shall be tested against all configurations
  - May break a different configuration than the one it is developed for

* Source: https://kernel-recipes.org/en/2017/talks/linux-kernel-release-model/

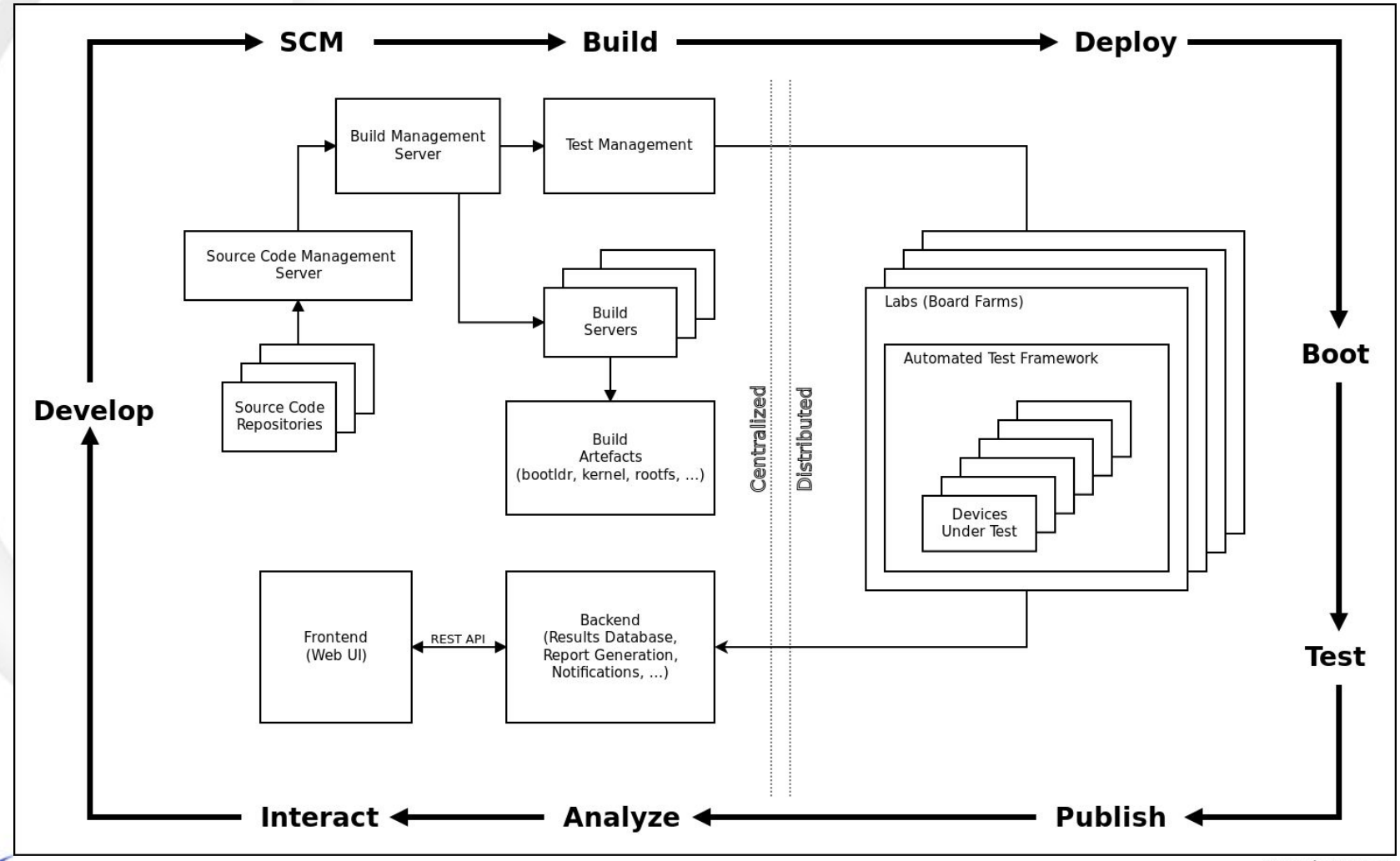
Why So Serious?

- **Manual testing model does not scale.**
  - Developers and maintainers cannot manually sustain that pace
  - Developers and maintainers cannot validate all patches physically on HW
  - Companies facing the same issue internally

- **We <u>absolutely</u> need to put in place CI loops that <u>automatically</u> test incoming patch series**

- **HW resources sharing** for CI is the **next step** into contributing to the community

Typical CI Loop



Generic CI Loop

LAVA Overview (1)

- **LAVA** (Linaro Automated Validation Architecture) : open source automated testing system which delivers device automation and result collection.
- Provides remote access to a set of devices, in order to
  - Deploy systems for testing,
  - Automate the operation of the boot and test
  - Coordinate / schedule multiple boots / tests
  - Collect the results.
- Involves the provision of HW and the SW to automate tests on that HW.

* Source: https://www.linaro.org/initiatives/lava/

LAVA Overview (2)

- The LAVA software includes
  - The lava-server component to
    - Schedule jobs,
    - Administer device configurations, and
    - Store results.
  - The lava-dispatcher component, which supports
    - Processing test jobs that can deploy Debian, Ubuntu, Open Embedded and Android images on supported development boards.
    - Physical connection to boards (e.g. serial console, USB for fastboot, etc.)

- Generic support can be extended and customised to support additional client types (development boards) and interface with external equipment.

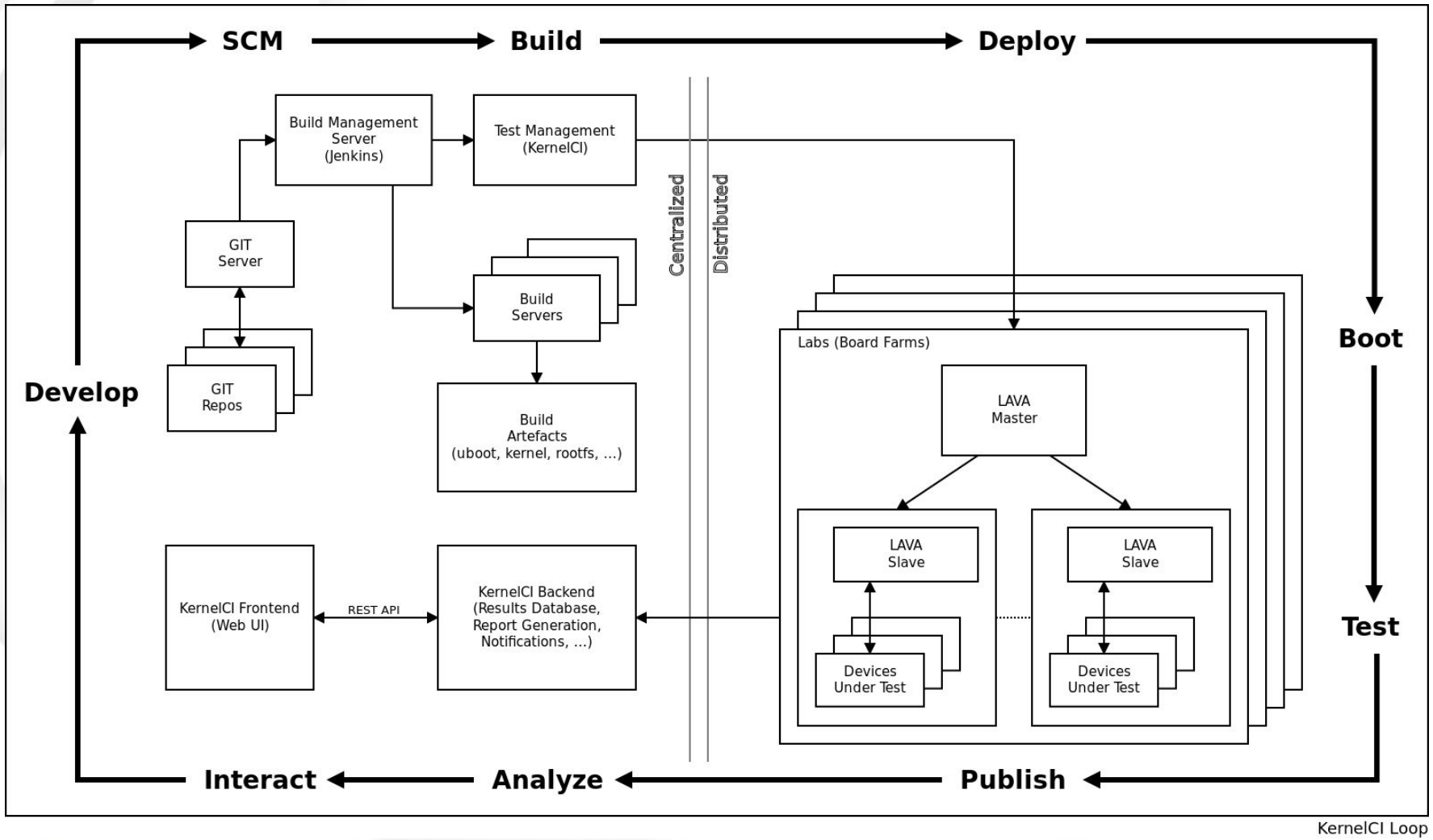
* Source: https://www.linaro.org/initiatives/lava/

KernelCI.org

- B**uild & Boot Test Automation System**
  - Focused on **upstream Linux kernel**,
  - Open Source, Community based,
    - (welcomes contributions like HW / Lab / infrastructure / resources)
  - Distributed, leveraging **LAVA**
- Since May 2014 :
  - Performed **3,325,676 boots** on **266 boards**, across **3 architectures** and **34 SoCs**. (**2,689 boots** per day.)
- Results reported via mailing lists and web site
- Much more likely that kernels will build… and run
  - v3.14: 51 failed configs
  - v4.1: 1 failed config

* Source: https://kernelci.org/, https://elinux.org/images/b/b9/Brown.pdf

KernelCI Loop



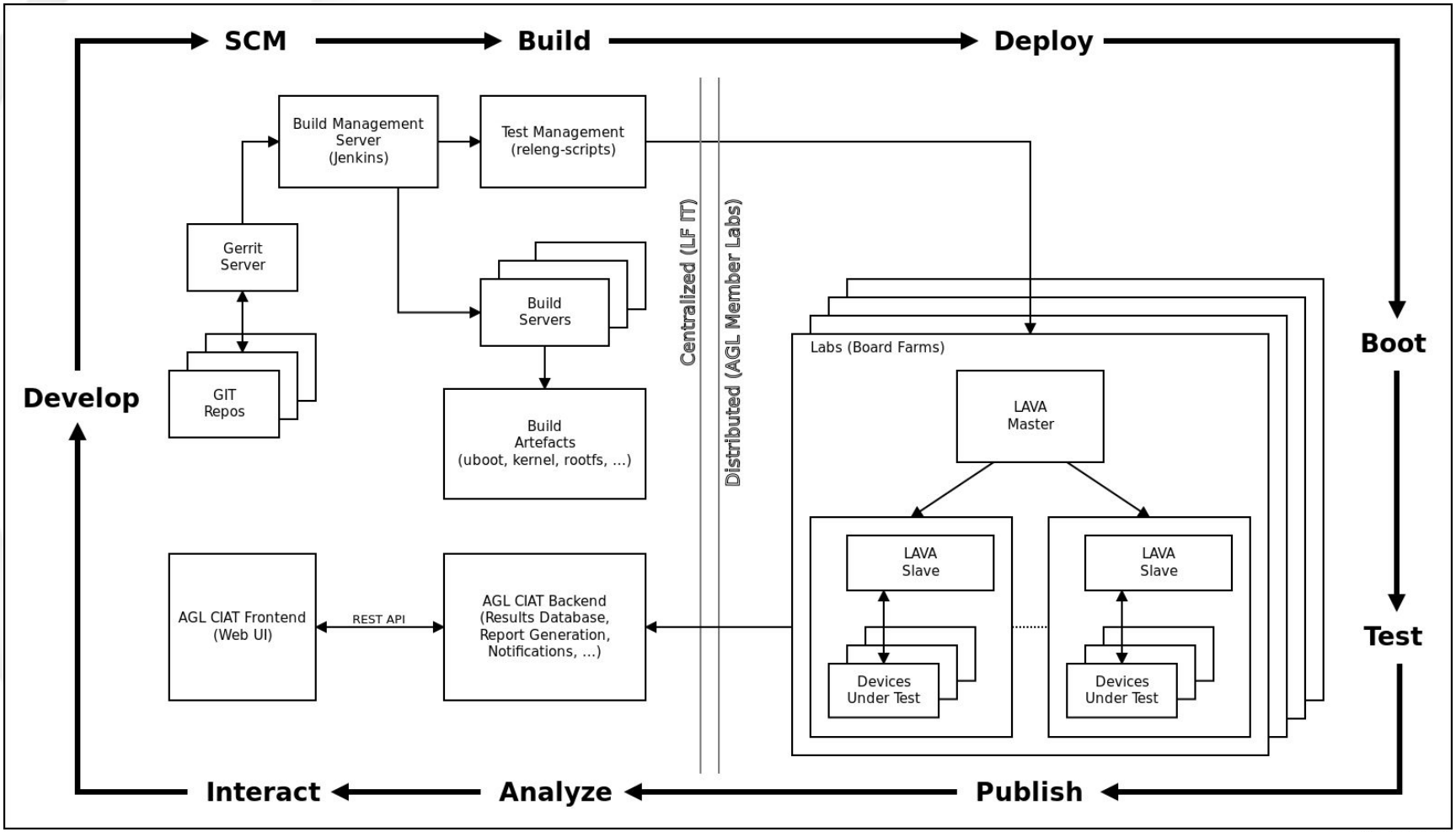
KernelCI Loop

AGL CIAT Loop

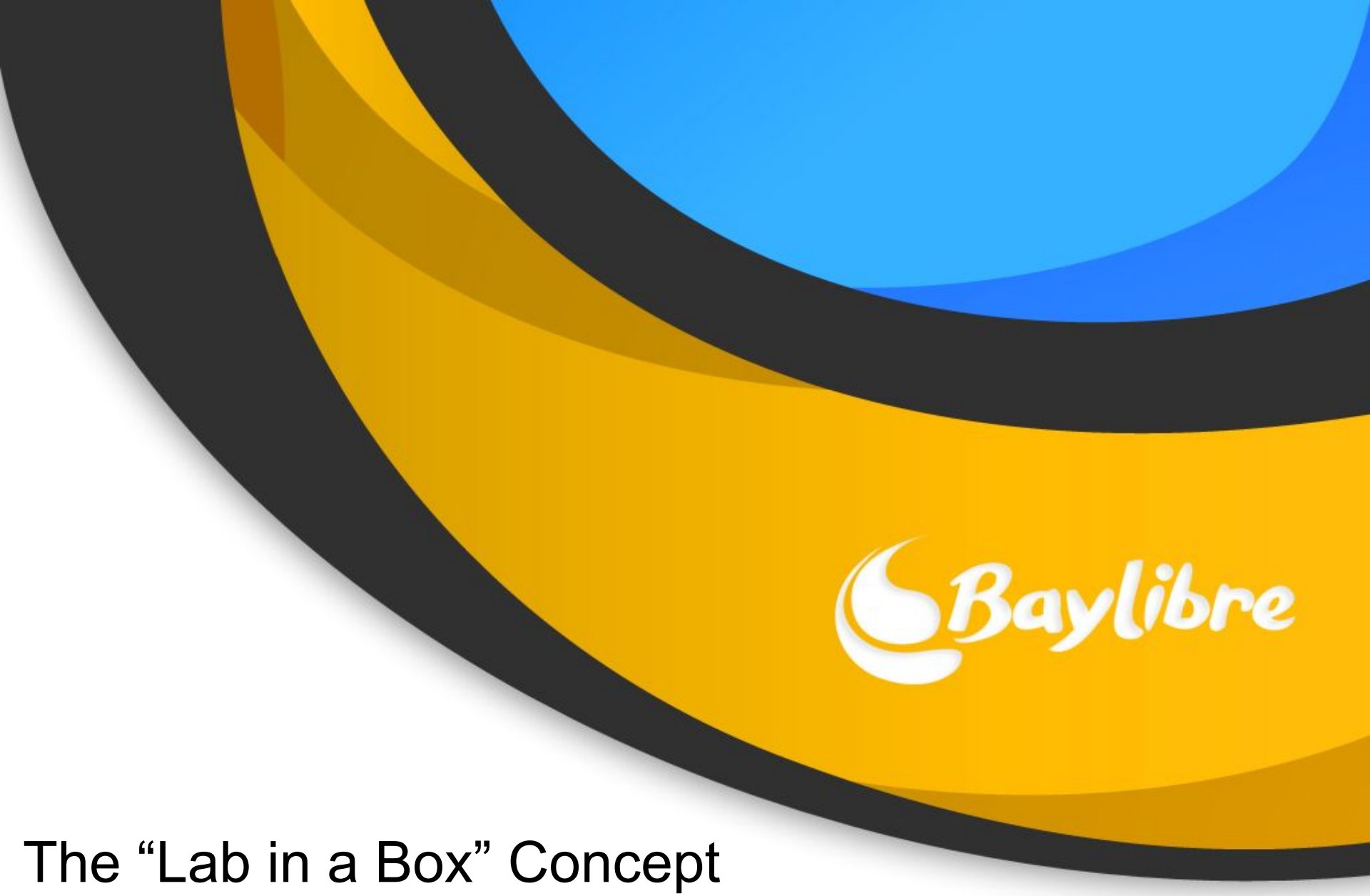
- Funded by Linux Foundation AGL Initiative
  - As part of the CIAT EG (Continuous Integration And Test Expert Group)
- Developed by Baylibre
- Leverages LAVA
- Leverages and extends KernelCI to
  - Test AGL releases, snapshots and per-commit (via gerrit)
  - Run any kind of test instead of only build and boot
    - Generic test suites,
    - AGL-specific test suites,
    - Automotive-specific test suites,
    - Power & Performance profiling,
    - ...
  -

* Source: https://wiki.automotivelinux.org/eg-ciat,

AGL CIAT Loop

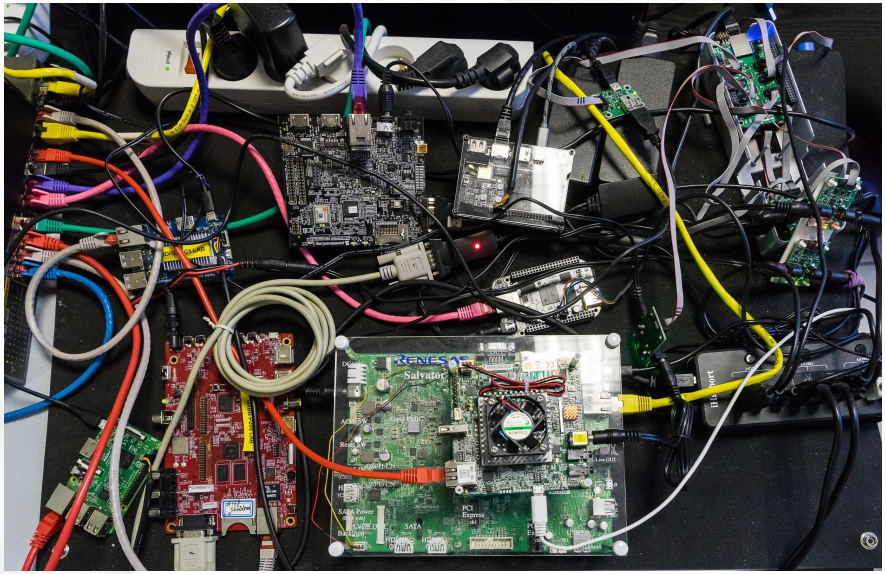


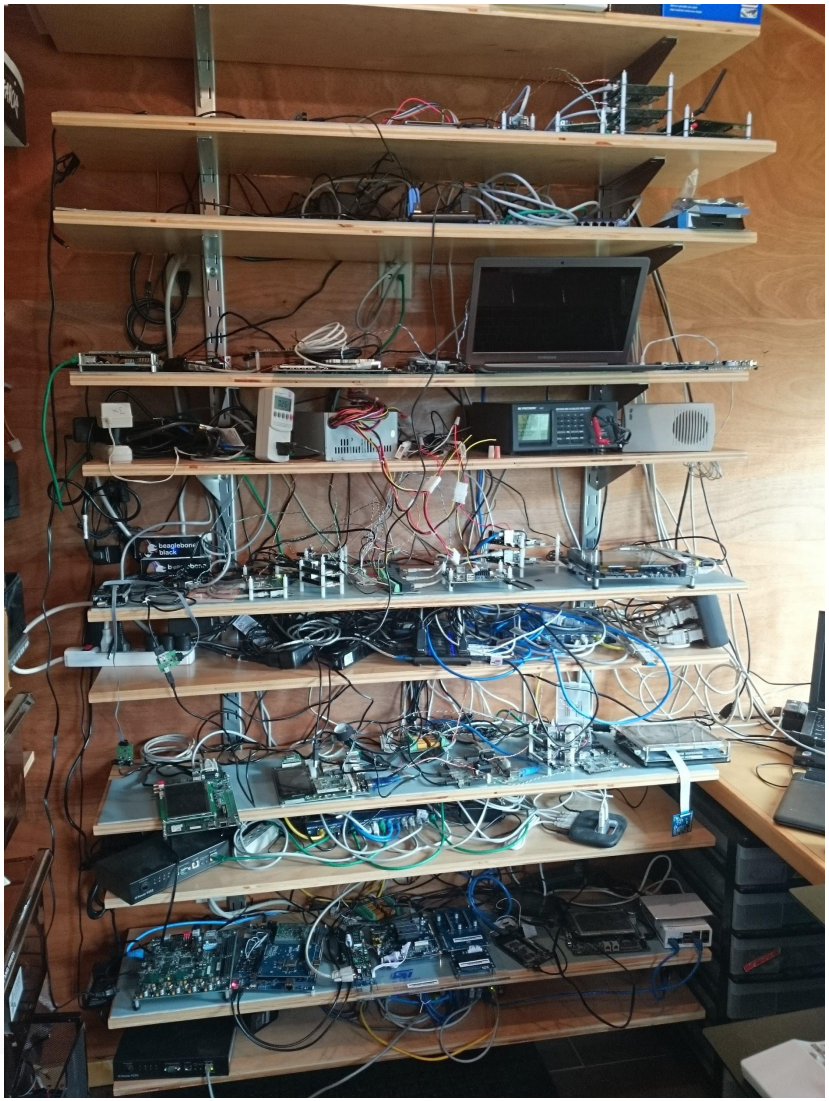
AGL CI Loop



The “Lab in a Box” Concept

Motivations





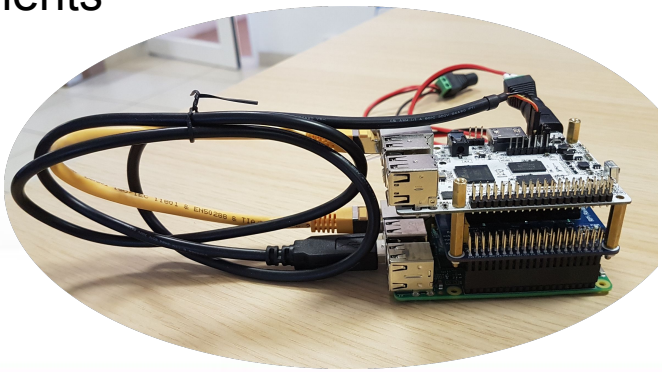
Time to go pro!

Motivations (2)

- Simplify Administration
  - LAVA: nice technology, but difficult to get into it
    - Installation process (now eased with Dockers)
    - Device-types
  - USB Serial debug ‘pairing’
  - Ultimately users shouldn’t be aware of the internal technologies to build and run a CI Lab
- Ease duplication / scalability
- Accelerate deployment

Requirements

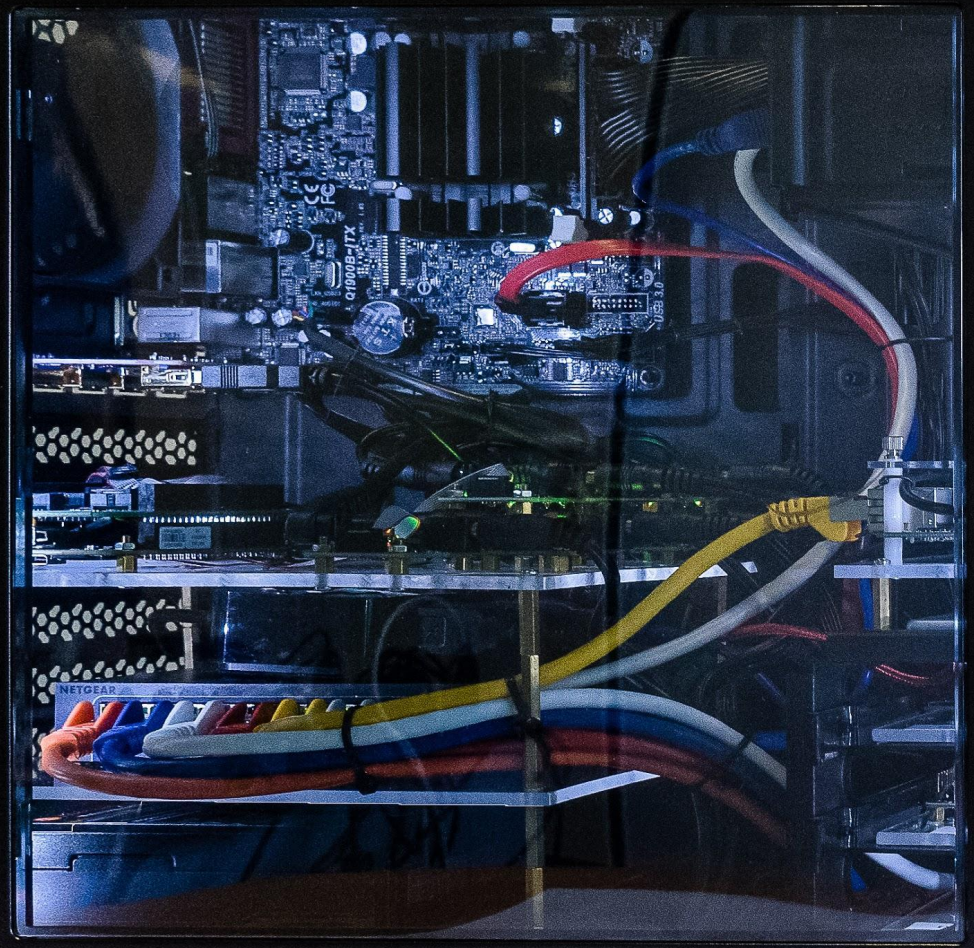
- “All in One” solution, integrating
  - LAVA master and dispatcher, Devices Under Test (DUT), power supplies for all DUT, connectivity / wiring (network, debug ports, power control, etc)
  - Reference & community AGL boards
- Lost cost
- Scalable / Reproducible
- Safe / Maintainable
- Easy installation (HW + SW)
  - Pre-installed / pre-configured SW components
  - Administration control panel
- Fits in an apartment (for home workers)
- Documented



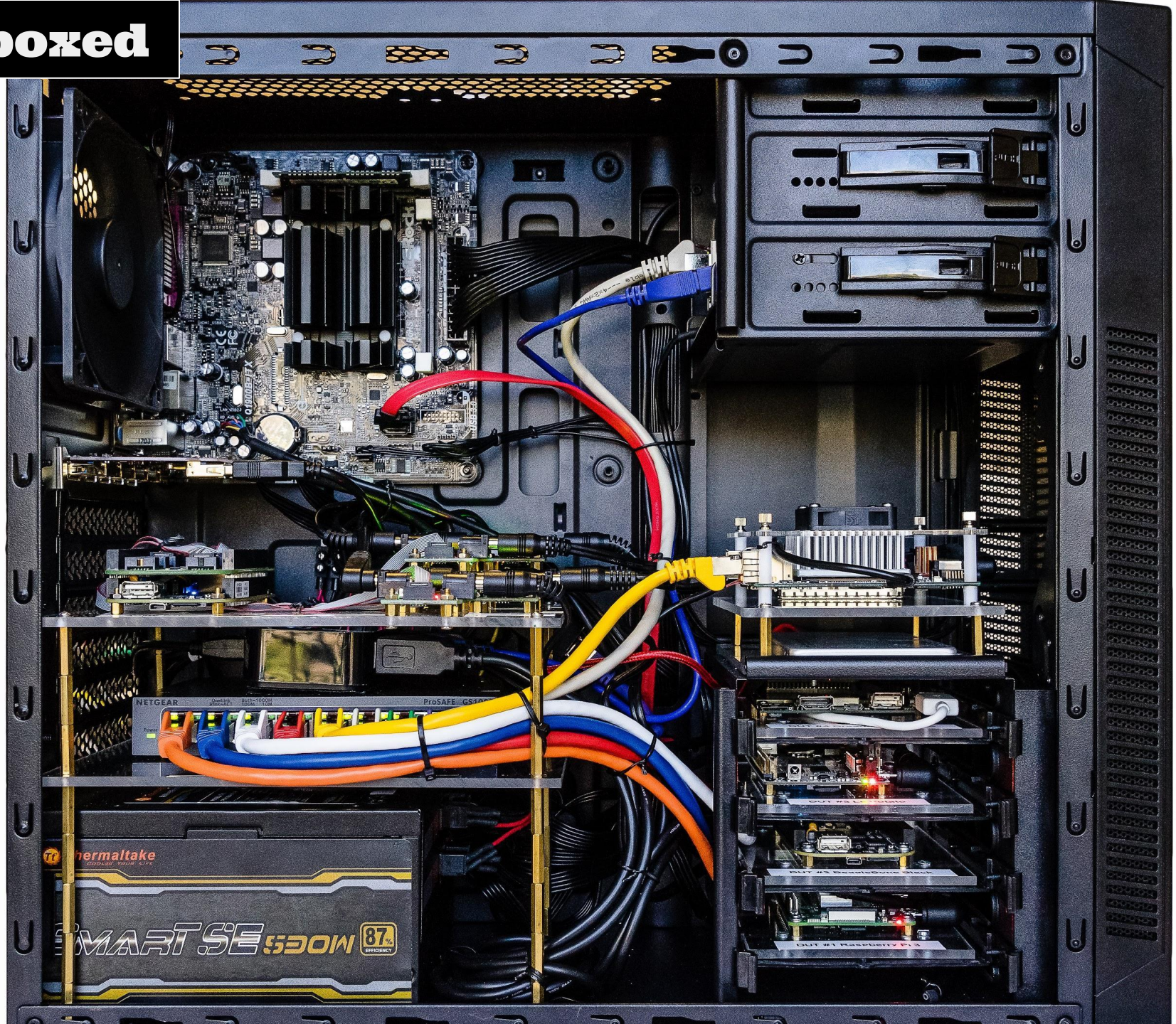
Challenges

- A lot of stuff to integrate in a single case
  - DUT
    - Custom size
    - Custom connections
  - Power Control unit
  - Lab Wiring
    - Network Switch
    - USB Hub
    - Per DUT
      - Power cable
      - Serial debug cable
      - Ethernet cable
- Maintenance
-

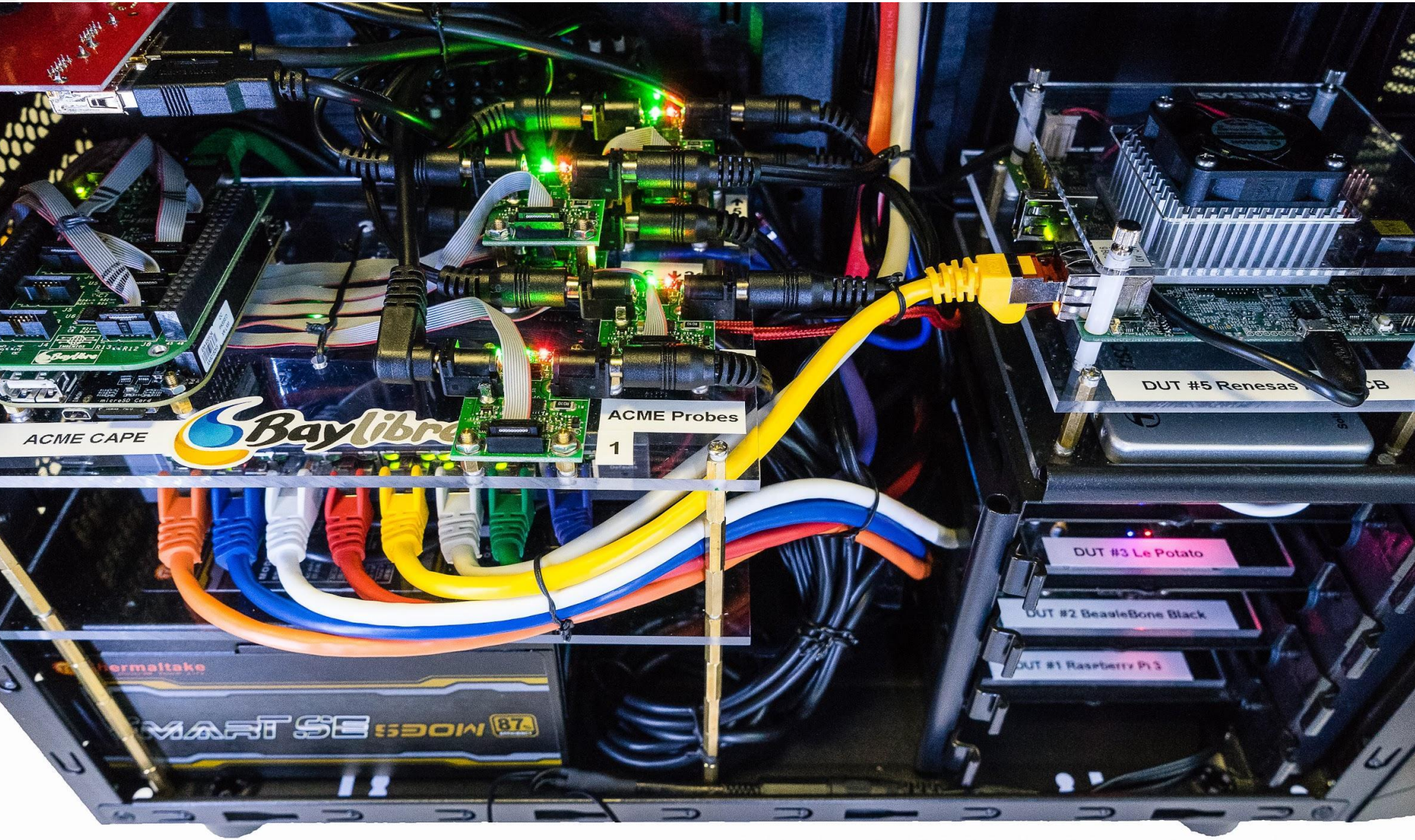
This is... LAVA box...



... unboxed


NETGEAR
ProSAFE GS108
Thermaltake
SMART SE 530W
87%
EFFICIENCY

Welcome inside...



Software: LAVA dispatcher (slave)

Manage all connections between boards and “real world”

Services

- DHCP
- TFTP
- NFS
- NBD

Power control

- BBB + ACME

https://baylibre-acme.github.io/

Serial consoles

- USB / serial cables (FTDI)
- udev rules
- conmux / cu

USB misc.

- fastboot
- gadget: ethernet, mass storage

Containerized using: lava-slave-docker project

https://github.com/kernelci/lava-slave-docker/

Software: LAVA server (master)

Web interface

Job scheduling, priorities

XML-RPC API

Board description

Board description: jinja2 templates

`device-type`

What all boards of this “type” have in common

- u-boot , fastboot, barebox, etc.
- Load addresses
- Bootloader environment

Can inherit/extend other device-types (e.g. base-uboot)

`device`

Specific to one instance of a board

- Select `device-type`
- How to connect to serial console
- PDU: how to power on/off
- Can override/extend settings from device-type

Containerized using: lava-docker project (from kernelCI)

https://github.com/kernelci/lava-docker/

Example: LAVA board description

device type descriptions: `/etc/lava-server/dispatcher-config/device-types/`

```
$ cat meson8b-odroidc1.jinja2
{% extends 'base-uboot.jinja2' %}

{% set console_device = console_device|default('ttyAML0') %}
{% set baud_rate = baud_rate|default(115200) %}
{% set device_type = "meson8b-odroidc1" %}
{% set bootloader_prompt = bootloader_prompt|default('odroidc#') %}
{% set interrupt_prompt = interrupt_prompt|default('stop autoboot') %}
{% set interrupt_char = interrupt_char|default('\n') %}

{% set bootm_kernel_addr = '0x21000000' %}
{% set bootm_ramdisk_addr = '0x22000000' %}
{% set bootm_dtb_addr = '0x21f00000' %}
{% set text_offset = '0x00208000' %}

{% set uboot_mkimage_arch = 'arm' %}
```

device descriptions: `/etc/lava-server/dispatcher-config/devices/`

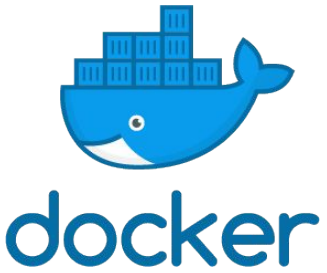
```
$ cat devices/meson8b-odroidc1-01.jinja2
{% extends 'meson8b-odroidc1.jinja2' %}
{% set connection_command = 'conmux-console lava-baylibre/meson8b-odroidc1' %}
{% set hard_reset_command = '/usr/local/bin/acme-cli -s lab-acme-1 reset 4' %}
{% set power_off_command = '/usr/local/bin/acme-cli -s lab-acme-1 switch_off 4' %}
{% set power_on_command = '/usr/local/bin/acme-cli -s lab-acme-1 switch_on 4' %}
```

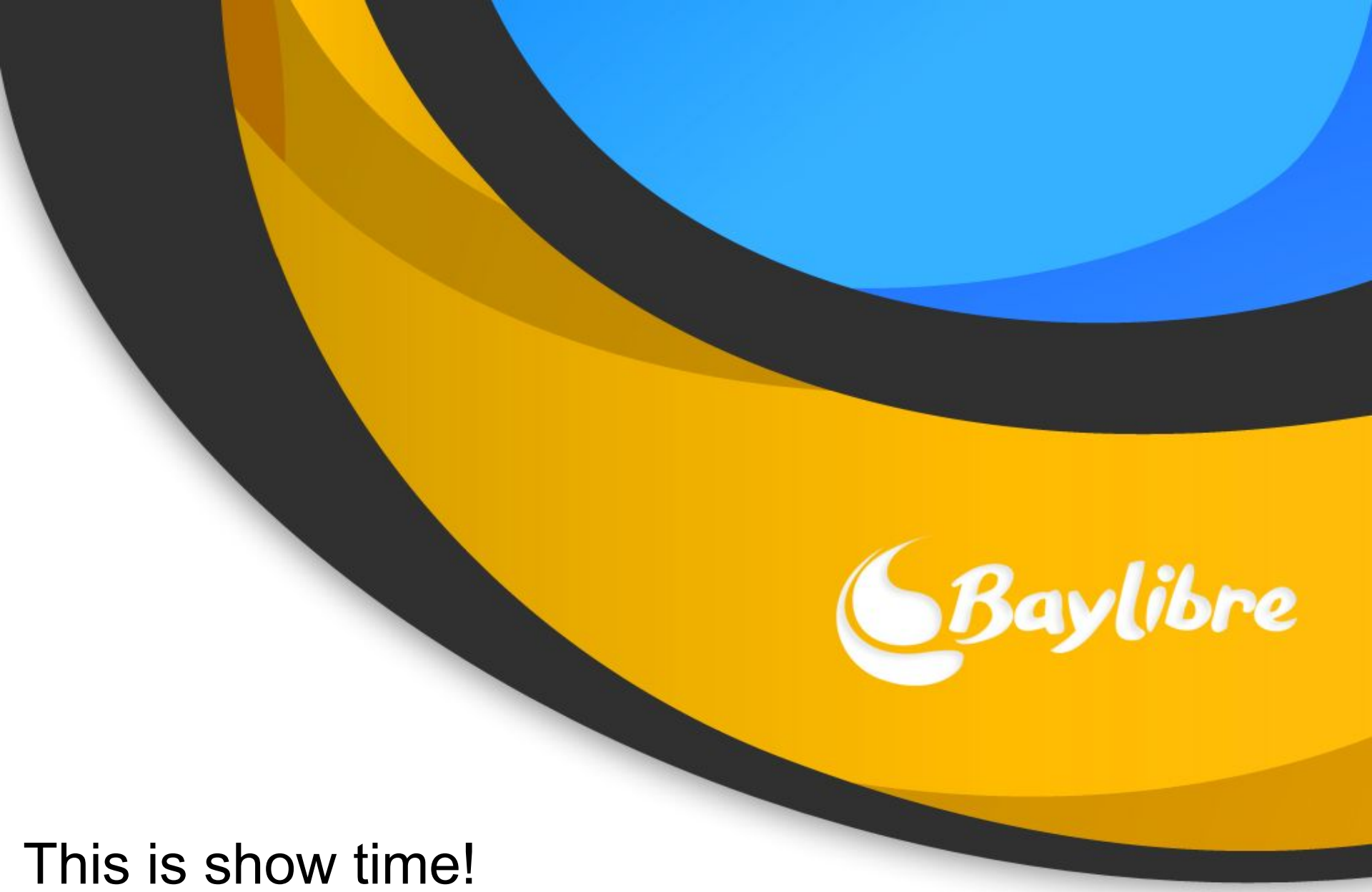
Software: Bringing it together

```
# cat docker-compose.yml
    services:
      lava-master:
        build: {context: lava-master}
        devices: ['/dev/kvm:/dev/kvm']
        hostname: lava-master
        ports: ['10080:80', '1022:22', '5555:5555', '5556:5556']
        restart: always
        stdin_open: true
        tty: true
        volumes: ['/boot:/boot', '/lib/modules:/lib/modules']
      lava-slave:
        build: {context: lava-slave}
        devices: ['/dev:/dev']
        environment: {LAVA_MASTER: lava-master}
        hostname: lab-slave-0
        links: [lava-master]
        ports: ['69:69/udp', '80:80', '55980-56000:55980-56000']
        restart: always
        stdin_open: true
        tty: true
      squid:
        build: {context: squid}
        hostname: squid
        ports: ['3128:3128']
        restart: always
        volumes: ['squid-cache:/var/spool/squid']
    version: '2.0'
```

Multi-container management:

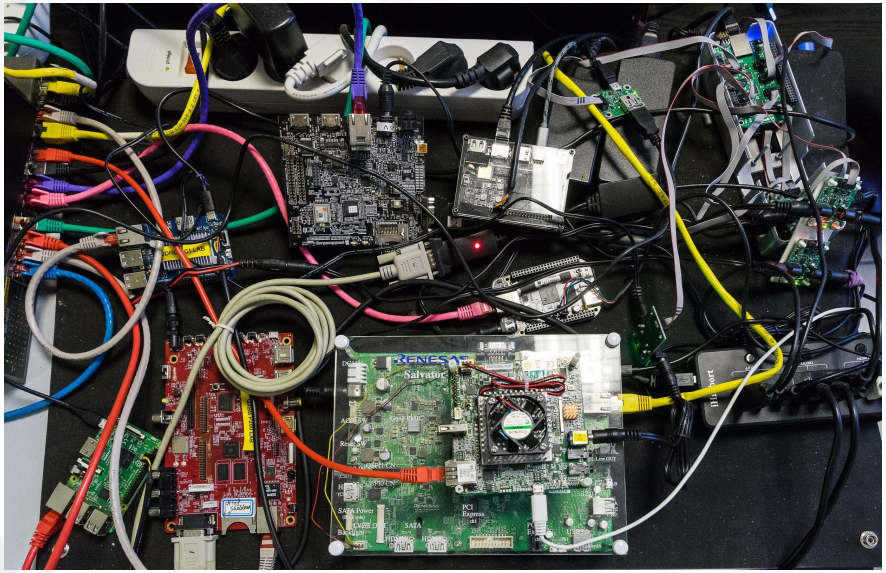
Docker compose

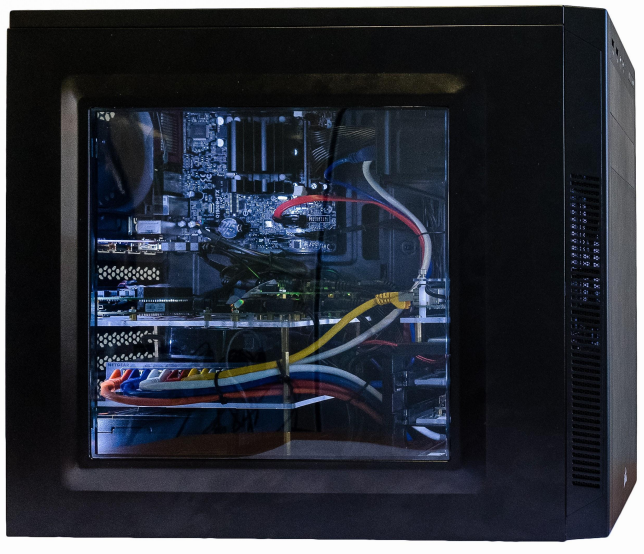




This is show time!

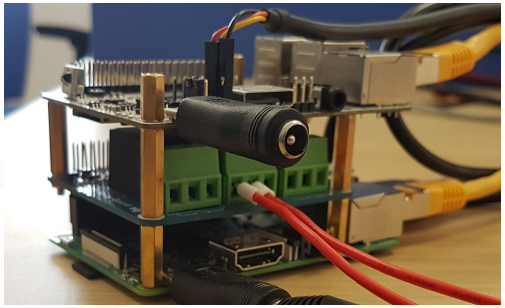
Achievements (1)





Achievements (2)

- Fully functional
- Complete CI LAVA lab integrated in single PC case
- No more wiring or boards laying on a desk / on shelves
- Fits well in our (small) appartments (for home workers)
- Good demonstrator for evangelising CI
- All DUT on drive trays, allowing easy maintenance
- Reasonable cost (400 euros)
  - Reduced when recycling PC / USB Hub / Network Switch / ...
- Partially Automated SW installations (still under work)



Limitations

- Tedious (long) to build / Difficult to “mass produce”
- Requires good tinkering (incl. soldering) skills
- Heavily packed
- DUT size limited (2x 5”¼, 5x 3”1/2)
  - Height
- Supports only +5V and +12V powered DUT
- DUT power consumption must be balanced across ATX connectors
  - Do not exceed 4A per pair of wires
- Using a larger PC case may not allow integrating many more DUT
  - Excessive internal wiring
- No standard “CI” connector
  - Custom wiring for each new DUT

What could be improved?

- Not recommended to use a “moderated” (and modular) power supply
  - Supports only +5V and +12V powered DUT
  - The more powerful the ATX power unit is, the more SATA/Molex connectors we get
- Integration of larger development boards
- Administration control panel

What's next?

- "Lab in a Box" was a first experimentation to validate the concept
  - Low-cost,
  - Targeting individuals
- Next:
  - Address Professional-grade "Lab in a Box"
  - More SW installation automation
  - More SW administration automation
    - Including administration control panel
  - Work with manufacturers to define standard CI connectors
  - Connectivity (Wi-Fi / BT)
  - Integrate standard test jobs

Q & A

THANK YOU!