Primer: Testing Your Embedded System - What is a ptest, Lava, Fuego, KernelCI and...?

Jan-Simon Möller
AGL Release Manager
jsmoeller@linuxfoundation.org
Overview / Scope
Frameworks to test your embedded ...

- Yocto Project / ptest
- Fuego
- LAVA
- KernelCI
- labgrid
- r4d
- of course there are more
Yocto Project's ptest
ptest – Fact sheet

- **Name:** ptest
- **Project:** The Yocto Project [www.yoctoproject.org](http://www.yoctoproject.org)
- **URL:** https://wiki.yoctoproject.org/wiki/Ptest
- **TLDR:** Packaging and Execution of 'make test' style testsuites on the DUT
ptest

- ptests are
  - sub-packages (foo-ptest)
  - a output format ("$RESULT: <testname>")
    - PASS/FAIL/SKIP
  - convention to call them ( run-ptest script )
    - ptest-runner on the target to start them
from zlib_1.2.11.bb:

SRC_URI += "file://run-ptest"

inherit ptest

do_compile_ptest() {
    oe_runmake test
}

do_install_ptest() {
    install ${B}/Makefile   ${D}${PTEST_PATH}
    install ${B}/example    ${D}${PTEST_PATH}
    install ${B}/minigzip   ${D}${PTEST_PATH}
    install ${B}/examplesh  ${D}${PTEST_PATH}
    install ${B}/minigzipsh ${D}${PTEST_PATH}

    # Remove buildhost references...
    sed -i -e "s,--sysroot=${STAGING_DIR_TARGET},,g" \
              -e 's|${DEBUG_PREFIX_MAP}||g' \
              ${D}${PTEST_PATH}/Makefile
}

RDEPENDS_${PN}-ptest += "make"

---

wrapper script for target

compilation procedure for testsuite

install test binaries

adapt scripts/path to target execution if necessary

declare (undetectable) runtime dependencies for tests (e.g. make)
To add package testing to your build, set the `DISTRO_FEATURES` and `EXTRA_IMAGE_FEATURES`.

- `DISTRO_FEATURES_append = "ptest"`
- `EXTRA_IMAGE_FEATURES += "ptest-pkgs"`
+ supports cross-compilation of the test-suite ahead of time
+ well integrated into the bitbake procedures
+ can be combined with the "testimage" class
- large log output, full run takes quite long
- Result visualization needs postprocessing
References:
- https://wiki.yoctoproject.org/wiki/Ptest
- http://bit.ly/2S5JNtA
Fuego
Fuego – Fact sheet

- **Name:** Fuego
- **Project:** Fuego Test System
- **URL:** http://fuegotest.org/
- **TLDR:** automated testing of embedded targets from a host system with 100 pre-packaged tests
Fuego

- Fuego is
  - a Jenkins instance preloaded with
    - a lot of tests (ranging from LTP to )
    - scripts to compile the test for the target
    - upload and run on the target
    - grab the results and parse them
    - partially visualize the results
<table>
<thead>
<tr>
<th>S</th>
<th>W</th>
<th>Name</th>
<th>Last Success</th>
<th>Last Failure</th>
<th>Last Duration</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.cyclistest</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.dbench</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.Dhrystone</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.hackbench</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.harness</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.interbench</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.linux</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.imbench2</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.NETBEANS</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.reboot</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.signedtest</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.threads</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.functional</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
<tr>
<td></td>
<td></td>
<td>beadbone.default.Benchmark.functionalcrashme</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
</tr>
</tbody>
</table>
Fuego

+ Large set of tests out of the box !!
+ No prerequisites on the target beside ssh
+ Result parsers so graphing is possible
- Assumes board is local (ssh)
- Assumes board is deployed with filesystem
- Each board needs a separate configuration
Fuego

- References:
  - http://fuegotest.org/wiki/FrontPage
LAVA
• Name: LAVA
• Project: LAVA Software Community Project
• URL: http://www.lavasoftware.org/
• TLDR: Device automation and test execution framework
LAVA does

- manage deployment of the filesystem
- power-on & booting & test exec on the DUT
- support multiple devices of the same type
- have templates for $\geq 150$ types of boards
- support devices to be remote (master/worker)
## All Devices

<table>
<thead>
<tr>
<th>Hostname</th>
<th>Worker Host</th>
<th>Device Type</th>
<th>state</th>
<th>Health</th>
<th>Submissions restricted to</th>
<th>Tags</th>
</tr>
</thead>
<tbody>
<tr>
<td>arndale01</td>
<td>dispatcher01.lavalab</td>
<td>arndale</td>
<td>Retired</td>
<td></td>
<td>Retired: no submissions possible.</td>
<td></td>
</tr>
<tr>
<td>arndale02</td>
<td>dispatcher02.lavalab</td>
<td>arndale</td>
<td>Retired</td>
<td></td>
<td>Retired: no submissions possible.</td>
<td></td>
</tr>
<tr>
<td>arndale03</td>
<td>dispatcher03.lavalab</td>
<td>arndale</td>
<td>Retired</td>
<td></td>
<td>Retired: no submissions possible.</td>
<td></td>
</tr>
<tr>
<td>arndale04</td>
<td>dispatcher04.lavalab</td>
<td>arndale</td>
<td>Retired</td>
<td></td>
<td>Retired: no submissions possible.</td>
<td></td>
</tr>
<tr>
<td>arndale05</td>
<td>dispatcher05.lavalab</td>
<td>arndale</td>
<td>Retired</td>
<td></td>
<td>Retired: no submissions possible.</td>
<td></td>
</tr>
<tr>
<td>b2260-01</td>
<td>dispatcher01.lavalab</td>
<td>b2260</td>
<td>Bad</td>
<td></td>
<td>Health check failed: no test jobs will be scheduled.</td>
<td></td>
</tr>
<tr>
<td>b2260-02</td>
<td>dispatcher02.lavalab</td>
<td>b2260</td>
<td>Idle</td>
<td>Good</td>
<td>group kernel-ci</td>
<td></td>
</tr>
<tr>
<td>beaglebone-black01</td>
<td>dispatcher01.lavalab</td>
<td>beaglebone-black</td>
<td>Idle</td>
<td>Good</td>
<td></td>
<td></td>
</tr>
<tr>
<td>beaglebone-black02</td>
<td>dispatcher01.lavalab</td>
<td>beaglebone-black</td>
<td>Idle</td>
<td>Good</td>
<td></td>
<td></td>
</tr>
<tr>
<td>beaglebone-black03</td>
<td>dispatcher02.lavalab</td>
<td>beaglebone-black</td>
<td>Idle</td>
<td>Good</td>
<td></td>
<td></td>
</tr>
<tr>
<td>beaglebone-black04</td>
<td>dispatcher02.lavalab</td>
<td>beaglebone-black</td>
<td>Idle</td>
<td>Good</td>
<td></td>
<td></td>
</tr>
<tr>
<td>beaglebone-black05</td>
<td>dispatcher03.lavalab</td>
<td>beaglebone-black</td>
<td>Retired</td>
<td></td>
<td>Retired: no submissions possible.</td>
<td></td>
</tr>
</tbody>
</table>
+ Multiple instances for each DUT-type
+ Master/Worker split allows multiple labs
+ Scales up

- Initial setup (improved by lava-docker)
- Less detailed parsing and presentation of results compared to Fuego
LAVA

• References
  • lava documentation:
    • https://validation.linaro.org/static/docs/v2/
  • lava-docker:
    • https://github.com/kernelci/lava-docker
KernelCI
KernelCI

- **Name:** KernelCI
- **Project:** KernelCI project
- **URL:** http://kernelci.org
- **TLDR:** Test aggregation and visualization
KernelCI

- KernelCI
  - is a database and webfrontend
  - lets you upload results in json
  - visualizes the results
Details for Tree «mainline» - v4.19-rc8-109-gc7b70a641df2

SoC  omap2
Tree  mainline → 
Git branch  master
Git describe  v4.19-rc8-109-gc7b70a641df2 → 
Git URL  http://git.kernel.org/pub/scm/linux/kernel/git/torvalds/linux.git
Git commit  c7b70a641df26002e8f26e2b8122fcb6a1d815a1
Date  2018-10-19

Unique boards  5
Unique SoCs  1
Unique defconfigs  3 out of 199

Conflicting Boot Reports

Boot report conflicts have been detected.

These are likely not failures since other boot labs are reporting a successful state: they need to be reviewed.

<table>
<thead>
<tr>
<th>Architecture</th>
<th>Defconfig</th>
<th>Board</th>
<th>Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>arm</td>
<td>multi_v7_defconfig</td>
<td>omap3-beagle-xm</td>
<td>2</td>
</tr>
</tbody>
</table>

Boards Tested

Lab «lab-baylibre» (6 – 2 / 4 / 0) (1 architecture / 2 boards / 1 SoC / 3 defconfigs)

<table>
<thead>
<tr>
<th>dra7-evm</th>
<th>omap2plus_defconfig</th>
<th>arm</th>
</tr>
</thead>
</table>
KernelCI

+ can aggregate results from multiple sources
+ multiple tools can upload results (LAVA/Fuego)
- setup (but kernelci-docker)
- UI adaptations not easy
KernelCI

• References:
  • https://github.com/kernelci/kernelci-docker
  • https://github.com/kernelci/kernelci-admin
  • http://powerci.org
Name: labgrid
Project: labgrid-project
URL: https://labgrid.readthedocs.io/en/latest/

TLDR: abstraction of the hardware control layer needed for testing of embedded systems
Labgrid – Remote Control

Coordinator

Client

Access ➔

Direct resource access ➔

Exporter

Exporter

Exporter

Exports ➔

Exports ➔

Exports ➔
labgrid

- labgrid lets you
  - expose the DUT resources
    - to $TESTTOOL
    - to $DEVELOPER
  in a unified manner
labgrid

+ allows automated and developer access to the DUTs
+ abstracts the HW specifics
- integration with testtools (Lava/Fuego)
- setup
• References:
  • https://labgrid.readthedocs.io/en/latest/
  • https://github.com/labgrid-project
Name: r4d
Project: ci-rt (Real-Time LINUX)
URL: https://github.com/ci-rt/r4d
TLDR: infrastructure for power-control and console access that plugs into libvirt
Name: r4d
Project: ci-rt (Real-Time LINUX)
URL: https://github.com/ci-rt/r4d
TLDR: infrastructure for power-control and console access that plugs into libvirt
$ r4dcfg --add-rack ci-rt-1 room209

$ r4dcfg --add-power ci-rt-1 pc8210 pc-ci-rt-1.lab.linutronix.de

$ r4dcfg --add-serial ci-rt-1 PS810 ds-ci-rt-1.lab.linutronix.de

$ r4dcfg --add-board ci-rt-1 6 seattle
r4d

- r4d
  - will manage power/serial
  - will allow remote control
  - plugs into libvirt (Jenkins!)
+ libvirt (!)
+ small
- only selected serial/power switches supported
- libvirt patches not yet upstream (but debian pkgs)
Other
For all I missed ...

- There are of course more / other frameworks
- in the time we could just not cover all
- please speak-up in the Q/A and lets discuss
Wrap-up and Q/A
Wrap-up

• Different frameworks have different strengths
• But what is important:
  • Collaboration
  • Test Results
    • Aggregation, Evaluation, Visualization
• More and more boards → LABs
• Questions?
• Notes?
End

- Thank you for joining!