Linaro Automated Validation on ARM

Paul Larson <paul.larson@linaro.org>
Some Quick Background

- Working Groups
- Landing Teams
- Platform Team
Engineering units

Working Groups
- Kernel Consolidation
- Toolchain
- Graphics
- Power Management
- Multimedia

Platform Engineering
- Validation & Benchmarking
- Evaluation Builds (Android, Ubuntu, Chrome, IVI...)
- Infrastructure
  - Release management

Validation, benchmarking, release management

SoC support and optimization

Landing Teams
- Samsung LSI
- Texas Instr.
- ST-Ericsson
- Freescale

Optimization & Innovation
Linaro Evaluation Builds

- **Ubuntu Desktop** – Linaro image based on Ubuntu Desktop
- **Nano** - small lightweight image for board validation
- **Developer** - focused image with console based developer tools.
- **ALIP** - (ARM Linux Internet Platorm) minimal X based image for internet access
- **Others**
- **Android**
Goals

• Deploy Linaro images across a variety of supported devices, without user intervention
• Make no assumptions about whether networking works or not
• Control the boot process
• Keep a good, recovery image
• Recover from hangs and crashes without manual intervention
• Monitor and retain serial log
• Execute tests and store results
LAVA Components

• Driver
• Scheduler
• Job Dispatcher
• Results Dashboard
Architecture Overview

LAVA Output Artifacts
- Output: *dashboard content
- Output: *result summary
- Output: *result bundle

LAVA Driver
- Input: *hwpacks
- Input: *images
- Input: *packages
- Input: *tests
- Output: *Job Bundle

LAVA Scheduler
- Input: *test description
- Input: *ready to test images

LAVA Runner
- Input: *Job Bundle

LAVA Client
- Output: *serial log
- Output: *system logs
- Output: *test results

LAVA Input Artifacts
- Input: *serial control stream
- Input: *image from net
- Input: *test commands
LAVA Components

- Driver
- Scheduler
- Job Dispatcher
- Results Dashboard
Driver

• Receive events from other systems
  • Continuous integration (jenkins, buildbot, etc)
  • Image build system (offspring)
  • Android build system
• Decide on tests to run based on input stream
• Submit templated jobs to the Scheduler
LAVA Components

- Driver
- **Scheduler**
- Job Dispatcher
- Results Dashboard
Scheduler

- Django app
- Web UI for human job creation
- XML-RPC interface for CLI job and automated job submission
- Support pools of similar hardware
  - Schedule to pool or to a specific device
- Scheduler Daemon
  - Process Job Queue
  - Launch dispatchers
  - Handle job (in)completion
  - Handle job timeouts
LAVA Components

- Driver
- Scheduler
- **Job Dispatcher**
- Results Dashboard
Job Dispatcher

- Jobs defined in json
- job_name – description of the job
- timeout – how long, at most, should it take
- target – which machine to run on
- actions [...]
Actions

- deploy_linaro_image
  - Parameters: hwpack, rootfs
- install_abrek
  - Parameters: tests[]
- boot_linaro_image
- test_abrek
  - Parameters: test_name
- submit_results
  - Parameters: server, stream
Example Job Submission

{
    "job_name": "foo",
    "target": "panda01",
    "timeout": 18000,
    "actions": [
        {
            "command": "deploy_linaro_image",
            "parameters": {
                "rootfs": "http://snapshots.linaro.org/11.05-daily/linaro-developer/20110208/0/images/tar/linaro-n-developer-tar-20110208-0.tar.gz",
                "hwpack": "http://snapshots.linaro.org/11.05-daily/linaro-hwpacks/panda/20110208/0/images/hwpack/hwpack_linaro-panda_20110208-0_armel_supported.tar.gz"
            }
        },
        {
            "command": "boot_linaro_image"
        },
        {
            "command": "test_abrek",
            "parameters": { "test_name": "ltp" }
        },
        {
            "command": "submit_results",
            "parameters": {
                "server": "http://dashboard.linaro.org",
                "stream": "panda01-ltp"
            }
        }
    ]
}
Test Execution

• Abrek
  • Lightweight, simple test execution framework
  • Modular, extensible
  • Interfaces with the dashboard
  • Test definition tells abrek how to:
    • Install (with dependencies)
    • Execute
    • Parse results
<table>
<thead>
<tr>
<th>Testsuites</th>
<th>Status</th>
</tr>
</thead>
<tbody>
<tr>
<td>Stream</td>
<td></td>
</tr>
<tr>
<td>LTP</td>
<td></td>
</tr>
<tr>
<td>Open Posix Test Suite</td>
<td></td>
</tr>
<tr>
<td>gmpbench</td>
<td></td>
</tr>
<tr>
<td>gtkperf</td>
<td></td>
</tr>
<tr>
<td>x11perf</td>
<td></td>
</tr>
<tr>
<td>glmemperf</td>
<td></td>
</tr>
<tr>
<td>tiobench</td>
<td></td>
</tr>
<tr>
<td>qgears</td>
<td></td>
</tr>
<tr>
<td>es2gears</td>
<td></td>
</tr>
<tr>
<td>clutter-eglx-es20</td>
<td></td>
</tr>
<tr>
<td>renderbench</td>
<td></td>
</tr>
<tr>
<td>glmark2-es2</td>
<td></td>
</tr>
<tr>
<td>Unixbench</td>
<td></td>
</tr>
<tr>
<td>GCC/Toolchain tests</td>
<td></td>
</tr>
</tbody>
</table>

Currently Supported
In Progress
Under Investigation
Master Image

- Basically, just a known-good, stripped down, Linaro image
- Needs to be able to
  - Manipulate images on the test partition
  - Talk to the network
- Also serves as a recovery partition
Deployment

- On the server:
  - Download hwpack and rootfs
  - Use linaro-media-create to create image
  - Extract tarball of boot and root fs
- On the test system:
  - Boot to master image
  - Reformat test partitions
  - wget/untar boot and root tarballs
  - Reboot
    - Directly interact with uboot to boot into test image
Test Execution

• Abrek

• Simple interface for installing/running tests, and dealing with results

• Works with launch-control (our dashboard for storing/visualizing results)

• Other tests and execution frameworks could easily be supported
  • Using abrek
  • Directly running from the dispatcher
LAVA Components

- Driver
- Scheduler
- Job Dispatcher
- Results Dashboard
Dashboard

- Django
- XML-RPC
- Well-defined, versioned, JSON bundle submissions
- Basic or OpenID authentication
- Supports text or binary attachments
- CLI interface (lc-tool)
  - Backup/Restore
  - Sync/transfer to another system
- Reporting (in progress)
  - Data source definition in XML
  - Flexible reporting
<table>
<thead>
<tr>
<th>Pathname</th>
<th>Name</th>
<th>Number of test runs</th>
<th>Number of bundles</th>
</tr>
</thead>
<tbody>
<tr>
<td>/anonymous/</td>
<td>not set</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>/anonymous/gcc/</td>
<td>gcc</td>
<td>1565</td>
<td>1565</td>
</tr>
<tr>
<td>/anonymous/panda01-ltp/</td>
<td>/panda01/ltp</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>/anonymous/panda01- posixtestsuite/</td>
<td>/panda01/posixtestsuite</td>
<td>1</td>
<td>2</td>
</tr>
<tr>
<td>/anonymous/panda01-stream/</td>
<td>/panda01/stream</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>/anonymous/zyga/</td>
<td>zyga</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

Streams are containers for bundles and are quite similar to folders or directories on your computer. (learn more)

Currently you have access to:
- All public streams

You must sign in to get more access

Launch Control is free software developed by Linaro. It is distributed under the terms of the GNU Affero General Public License version 3. You have the right to obtain source code of any server side installations of this software that you interact with.

This website should contain only valid XHTML markup, you can validate it if you wish.
Lab Hardware

• Server
• APC Switched PDUs
  • For hard resetting
• Cyclades ACS 6032
  • Console servers
• Basic networking infrastructure, cables, db9 connectors, mounting hardware, etc.
Current Systems Available

- TI PandaBoard (2)
- TI Beagle XM
- Samsung SMDK v310
- Freescale imx51
- Freescale imx53
- STE u8500 (2)
- Versatile Express (2)
- …many more on the way :)
Gratuitous Pictures
More Gratuitous Pictures
Thank You

Any questions?