SFO15-506:
Capabilities of Workload Automation
Test Harness

Lisa Nguyen, Linaro
Introduction

- Created by ARM
- Run benchmarks repeatably
- Supports Android, Linux, and ChromeOS
- Written in python
Setup example

- Host (laptop) running Ubuntu 14.04 LTS
- Pandaboard as target (running either Android or Ubuntu)
- Networking required!
- Host has python 2.7, build-essential, git, and OpenSSH packages installed
Setup commands

On Ubuntu:

- `sudo apt-get install python-pip`
- `sudo apt-get install python-serial`

Install packages via pip:

- `sudo pip install --upgrade pip`
- `sudo pip install --upgrade setuptools`
- `sudo pip install pexpect`
- `sudo pip install pyserial`
- `sudo pip install docutils`
- `sudo pip install python-dateutil`
- `sudo pip install wlauto`
Demo

Run WA remotely on a board in Kevin Hilman’s lab and show some nifty commands!
WA Execution Flow

Legend:
- core framework
- workload
- instrumentation
- result processors
- signal dispatch

**Instrumentation**
- instrumentation initialization
- instrumentation iteration setup
- start measurement
- stop measurement
- update iteration results
- update run results

**Workload Execution**
- run started
- device initialized
- job queue initialized
- job started
- workload setup
- workload run
- workload results update
- workload teardown
- job completed
- finalize run
- run ended

**Result Processing and Export**
- process iteration results
- process run results
Sample agenda

- written in YAML
- global config settings for workloads
- lists workloads to execute in order
- Download a sample agenda at http://people.linaro.org/~lisa.nguyen/agenda.yaml
Linaro’s use of WA

- Collect power measurements
  - Pre-instrumented boards
- Run set of benchmarks on various boards
Limitations

- One-to-one. One WA instance per board.
- Only supported in python 2.7
- Assumes that target IP is static
- Limited reporting capabilities
- SSH is still required for running WA on Linux
Next steps

● Investigate feasibility of using reports to declare PASS/FAIL
  ○ Every board has different thresholds for performance/power
  ○ Intel’s LKP handles this well

● Work with kernelci.org to add another dimension to automated testing
Questions and comments?
Backup Slides
Build and install from source

To build and install from source:

- mkdir ~/projects directory as an example
- cd ~/projects directory
- git clone https://github.com/ARM-software/workload-automation.git
- cd workload-automation directory
- python setup.py sdist
- sudo pip install dist/wlauto-*.tar.gz (where * is current version number)
Create a new workload

- `wa create workload <name>`
- check for new workload template under `~/.workload_automation/workloads/<name>`
- fill in template with code

Similar process to creating a new device, instrument, or results processor.
Sample config

device_config = dict(
    # The name used by adb to identify the device. Use "adb devices" in bash to list
    # the devices currently seen by adb.
    #adb_name='10.109.173.2:5555',

    # The directory on the device that WA will use to push files to
    #working_directory='/sdcard/wa-working',

    # This specifies the device's CPU cores. The order must match how they
    # appear in cpufreq. The example below is for TC2.
    # core_names = ['a7', 'a7', 'a7', 'a15', 'a15']

    # Specifies cluster mapping for the device's cores.
    # core_clusters = [0, 0, 0, 1, 1]

    host = '192.168.1.165',
    username = 'root',
    keyfile = '/home/lisatn/.ssh/id_rsa_host'
)

- Device information
- SSH configuration
- Enable/disable instruments
- Enable/disable reporting options