



CE Workgroup

Introduction to the **Fuego** Test System

Tim Bird

Architecture Group Chair

LF CE Workgroup



CE Workgroup

Introduction to the **Fuego** Test System

Tim Bird

Architecture Group Chair

LF CE Workgroup

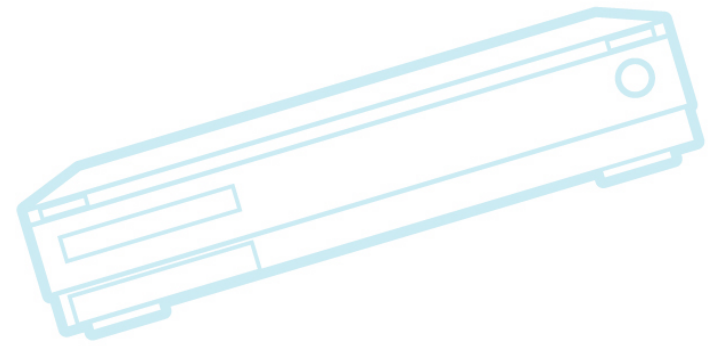
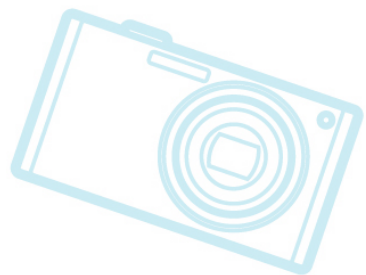
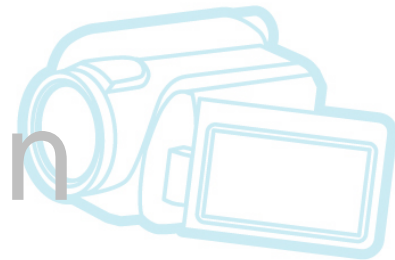
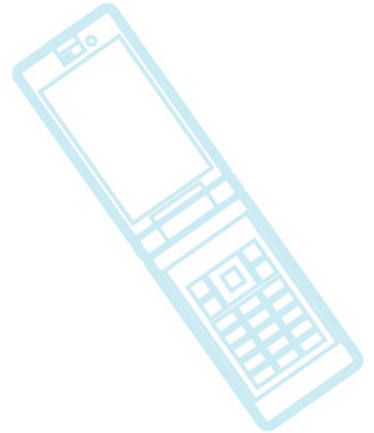




CE Workgroup

Outline

Introduction
Architecture
Customization
Vision

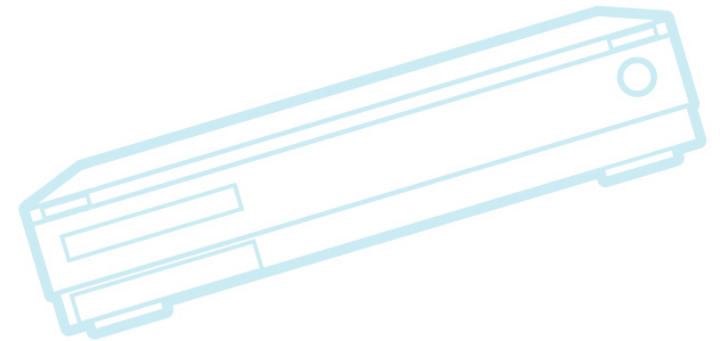
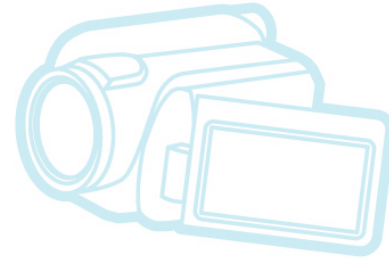
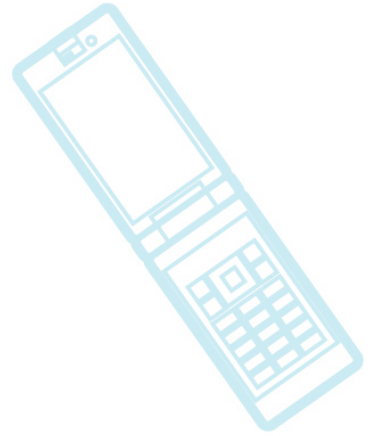




CE Workgroup

Introduction

Fuego = Jenkins +





CE Workgroup

Introduction

Fuego = Jenkins +
abstraction scripts +



CE Workgroup

Introduction

Fuego = Jenkins +
abstraction scripts +
pre-packed tests



CE Workgroup

Introduction

Fuego = (Jenkins +
abstraction scripts +
pre-packed tests)
inside a container



Jenkins

- Is a Continuous Integration system
- Handles all of that “continuous integration-y” type stuff
 - Launches test jobs based on various triggers
 - Shows test results
 - Has an ecosystem of plugins for all kinds of extended functionality
 - E-mail notifications
 - Plotting of results
 - Integration with different source code management systems
- Is too big a system to describe in detail here



CE Workgroup

Jenkins

- Base interface:

Test history and test selection dashboard

The screenshot shows the Jenkins Test Automation Framework interface in a Mozilla Firefox browser. The page title is "0. History [Test Automation Framework] - Mozilla Firefox". The address bar shows "localhost:8080/fuego/". The page features logos for COGEMBEDDED, THE LINUX FOUNDATION, LONG TERM SUPPORT INITIATIVE, and RENESAS. The main content area is titled "Test Automation Framework" and includes a navigation menu on the left with options like "New Test", "People", "Test Run History", "Edit Dashboard", "Documentation", "Manage Jenkins", "Scriptler", and "Exclusion administration". The main content area has tabs for "0. History", "Benchmarks", "Functional", "all", "batch runs", and "+". Below the tabs is a table titled "Latest tests runs" with columns for "Test", "Run", "Time", "Platform SDK", and "Device". The table lists several test runs with their respective statuses and times. Below the table is a "Test Run statistics" section with a table titled "Status of the test run" showing the number of test runs for each status: Failed (1), Unstable (0), Success (5), Pending (54), Disabled (0), Aborted (0), and Not built (0).

Test	Run	Time	Platform SDK	Device
Benchmark.dbench	#3	Apr 3, 2016 4:21:28 PM	bbb-poky-sdk	bbb-poky-sdk
Benchmark.dbench	#2	Apr 3, 2016 4:07:22 PM	bbb-poky-sdk	bbb-poky-sdk
Benchmark.Dhrystone	#1	Apr 3, 2016 4:07:08 PM	bbb-poky-sdk	bbb-poky-sdk
Functional.bzip2	#1	Apr 3, 2016 4:06:56 PM	bbb-poky-sdk	bbb-poky-sdk
Functional.posixtestsuite	#1	Apr 3, 2016 3:59:44 PM	bbb-poky-sdk	bbb-poky-sdk
Benchmark.dbench	#1	Apr 3, 2016 3:58:22 PM	bbb-poky-sdk	bbb-poky-sdk
Functional.bc	#1	Apr 3, 2016 3:57:35 PM	bbb	bbb

Status of the test run	Description	Number of test runs
Failed	Failed	1
Unstable	Unstable	0
Success	Success	5
Pending	Pending	54
Disabled	Disabled	0
Aborted	Aborted	0
Not built	Not built	0

- Fuego includes customizations to Jenkins to support host/target configurations
- Pre-install plugins for plotting and other stuff



search

- New Test
- People
- Test Run History
- Edit Dashboard
- Documentation
- Manage Jenkins
- Scriptler
- Exclusion administration

Test Automation Framework

[add description](#)

- 0. History**
- Benchmarks
- Functional
- all
- batch runs
- +

Latest tests runs

Test	Run	Time	Platform SDK	Device
Benchmark.dbench	#3	Apr 3, 2016 4:21:28 PM		bbb-poky-sdk
Benchmark.dbench	#2	Apr 3, 2016 4:07:22 PM		bbb-poky-sdk
Benchmark.Dhrystone	#1	Apr 3, 2016 4:07:08 PM		bbb-poky-sdk
Functional.bzip2	#1	Apr 3, 2016 4:06:56 PM		bbb-poky-sdk
Functional.posixtestsuite	#1	Apr 3, 2016 3:59:44 PM		bbb-poky-sdk
Benchmark.dbench	#1	Apr 3, 2016 3:58:22 PM		bbb-poky-sdk
Functional.bc	#1	Apr 3, 2016 3:57:35 PM		bbb

Test Run statistics

Status of the test run	Description	Number of test runs
	Failed	1
	Unstable	0
	Success	5
	Pending	54
	Disabled	0
	Aborted	0
	Not built	0

Test Run Queue
No test runs in the queue.

Targets Status

#	Master
1	Idle
2	Idle
bbb	
1	Idle
bbb-poky-sdk	
1	Executing Benchmark.dbench #4
lager	
1	Idle
lager2	
1	Idle



Abstraction scripts

- User defines a few variables in shell scripts, to allow system to interact with target boards
- Fuego provides shell functions for command and control of target:
 - Put/get files, execute commands, collect logs, etc.
- Fuego generates a full test script at runtime, based on board configuration, toolchain variables, and test variables
 - This allows all aspects of tests to be abstracted
 - This is a bigger deal than it sounds like



Overlay generation

- Four areas of overlaid functions and variables
 - Functions to interact with target
 - Board definitions
 - Toolchain variables
 - Test parameters
- Indirection for test program parameters
- Tests have a simple shell program wrapper
- This wrapper is expanded using an overlay generator at runtime, into a full script to execute the test and collect results



Overlay processing

Base script

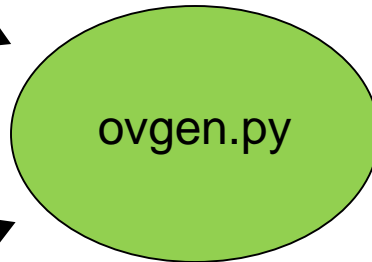
```
test-script.sh  
test_build()  
test_deploy()  
test_run()
```

Fuego functions

```
functional.sh  
functions.sh  
common.sh  
overlays.sh  
reports.sh  
etc.
```

Extended script

```
<target>_prolog.sh
```



```
<board>.conf
```

```
tools.sh
```

```
testplan
```

```
test specs
```



Test parameter abstraction

- Being able to write tests that run in multiple configurations is important
- Fuego abstracts target access methods
- Fuego also abstracts:
 - Platform for software builds
 - Filesystem device
 - Filesystem mount points
- User can easily add new items to be abstracted
- Test plan system allows a single test to be run in multiple configurations



Pre-packaged tests

- Comes with over 50 tests, already integrated
 - aim7, blobsalad, bonnie, cyclitest, dbench, dhrystone, ebizzy, ffsb, fio, GLMark, gtkperf, hackbench, himeno, Interbench, IOzone, iperf, Java, linpack, lmbench2, nbench, netperf, netpipe, OpenSSL, reboot, signaltest, Stream, tiobench, whetstone, x11perf, aiostress, arch_timer, bzip2, cmt, crashme, expat, fontconfig, glib, ipv6connect, jpeg, libpng, linus_stress, LTP, netperf, posixtestsuite, rmaptest, scifab, scrashme, sdhi_o, stress, synctest, zlib
- Includes functional, benchmark and stress tests



CE Workgroup

Test building

- Tests are built from source
- You can use your own toolchain (/sdk)
 - Or use a pre-installed generic arm toolchain
- There's an Open Embedded meta-layer available, to help you build your own SDK in YP/OE
 - Generated SDK will have libraries and headers needed for building all tests



CE Workgroup

Inside a container

- Fuego builds a docker container
- This avoids a lot of install issues
 - Fuego can run on any Linux distro
- Builds of the test programs are 100% reproducible



CE Workgroup

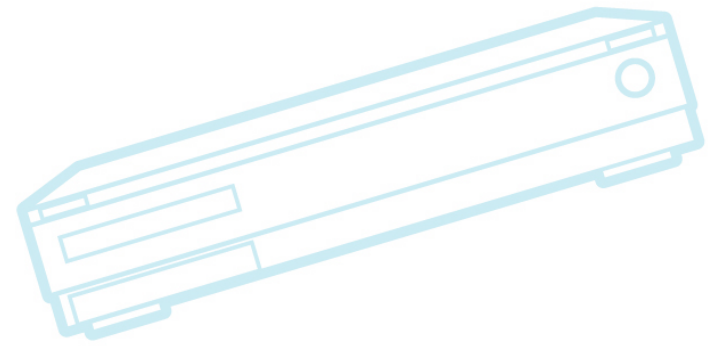
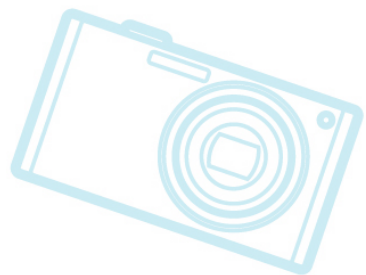
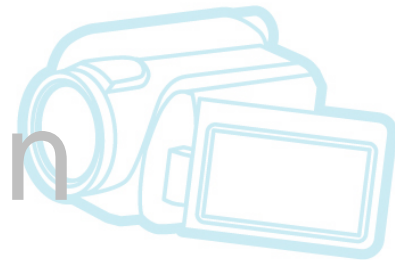
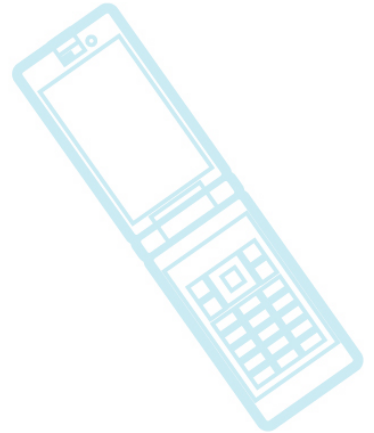
Outline

Introduction

Architecture

Customization

Vision





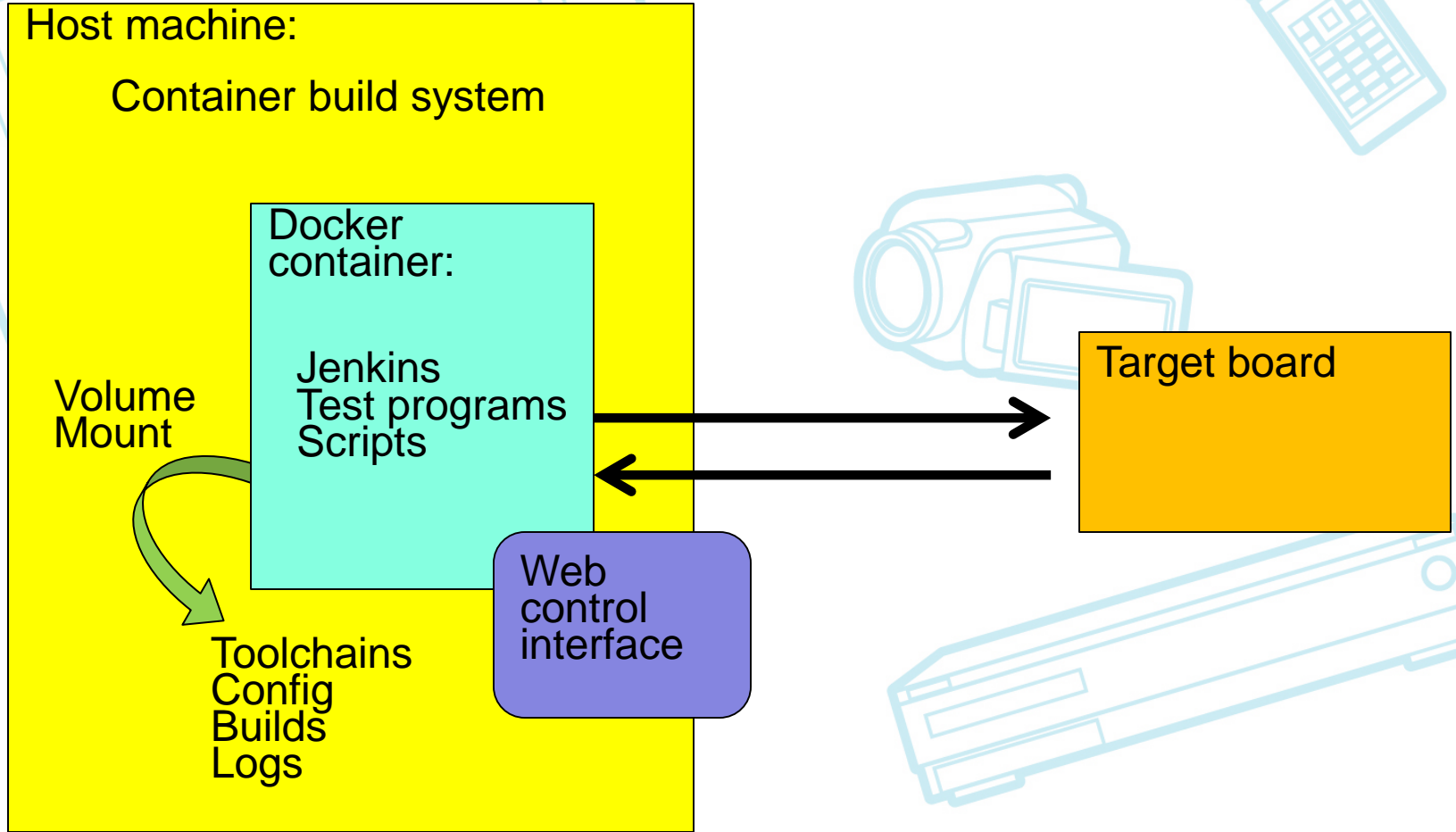
CE Workgroup

Architecture

- 2 major parts used for configuration:
 - Jenkins front-end
 - Script back-end
- Back-end is (mostly) shell-script based
 - Main interface between Jenkins and test programs is a single shell script
 - Shell is lowest common denominator language
- Very small files (glue layer) required for:
 - Log parsing
 - Results plotting



Architecture Diagram





How deployed

- Comes as 2 git repositories:
 - ‘fuego’ repository - Stuff outside the container
 - Container build system
 - Including some Jenkins plugins
 - Default config and boards
 - Host scripts for controlling the container
 - Documentation
 - ‘fuego-core’ repository - Stuff inside the container
 - Script and overlay engine
 - Pre-packaged tests
 - More jenkins extensions
- Fuego-core is downloaded for you during the container image build



CE Workgroup

Getting it and using it

- `git clone https://bitbucket.org/cogentembedded/fuego.git`
- `cd fuego ; ./install.sh`
(wait a bit)
- `fuego-host-scripts/docker-create-container.sh`
- `fuego-host-scripts/docker-start-container.sh`
- `firefox http://localhost:8080/fuego`

- Optionally, to get additional shell prompts inside the container:
 - `docker exec -i -t <container_id> bash`
 - `sshd <user>@localhost -p 2222`
 - Requires that you create a user account inside the container



Home

ENABLE AUTO REFRESH

- New Test
- People
- Test Run History
- Edit Dashboard
- Documentation
- Manage Jenkins
- Scriptler
- Exclusion administration

Test Automation Framework

[add description](#)

- 0. History
- Benchmarks
- Functional
- all
- batch runs
- +

Latest tests runs

Test	Run	Time	Platform SDK	Device
Benchmark.dbench	#3	Apr 3, 2016 4:21:28 PM		bbb-poky-sdk
Benchmark.dbench	#2	Apr 3, 2016 4:07:22 PM		bbb-poky-sdk
Benchmark.Dhrystone	#1	Apr 3, 2016 4:07:08 PM		bbb-poky-sdk
Functional.bzip2	#1	Apr 3, 2016 4:06:56 PM		bbb-poky-sdk
Functional.posixtestsuite	#1	Apr 3, 2016 3:59:44 PM		bbb-poky-sdk
Benchmark.dbench	#1	Apr 3, 2016 3:58:22 PM		bbb-poky-sdk
Functional.bc	#1	Apr 3, 2016 3:57:35 PM		bbb

Test Run Queue

No test runs in the queue.

Targets Status

#	Master
1	Idle
2	Idle
bbb	
1	Idle
bbb-poky-sdk	
1	Executing Benchmark.dbench #4
lager	
1	Idle
lager2	
1	Idle

Test Run statistics

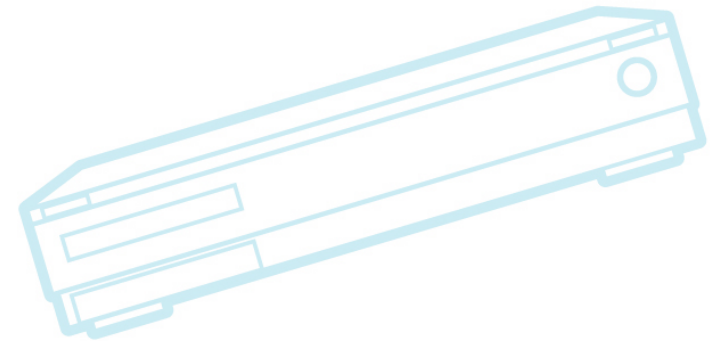
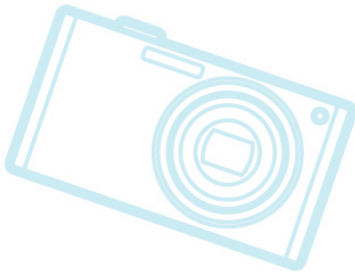
Status of the test run	Description	Number of test runs
	Failed	1
	Unstable	0
	Success	5
	Pending	54
	Disabled	0
	Aborted	0
	Not built	0



CE Workgroup

Running a test (manually)

- Select a test
- Select the target
- Select the testplan
- Push “Run the test”





CE Workgroup

Fuego tests page

Benchmarks [Test Automation Framework] - Mozilla Firefox

localhost:8080/fuego/view/Benchmarks/

COGENT EMBEDDED THE LINUX FOUNDATION LONG TERM SUPPORT INITIATIVE RENESAS

Home Benchmarks ENABLE AUTO REFRESH

New Test People Test Run History Edit Dashboard Delete Dashboard Documentation Manage Jenkins Scriptler Exclusion administration

Test Run Queue
No test runs in the queue.

Targets Status

#	Master
1	Idle
2	Idle
bbb	
1	Idle
bbb-poky-sdk	
1	Idle
lager	
1	Idle
lager2	
1	Idle

Test Automation Framework

0. History **Benchmarks** Functional all batch runs +

[add description](#)

Tests list

S	W	Test Name w/ Status Color ↓	Test Priority	Last Duration	Device
●		Benchmark.aim7	230	N/A	N/A
●		Benchmark.blobsallad	230	N/A	N/A
●		Benchmark.bonnie	150	N/A	N/A
●		Benchmark.cyclictest	150	N/A	N/A
●	☀	Benchmark.dbench	200	1 min 17 sec	bbb-poky-sdk
●	☀	Benchmark.Dhrystone	230	13 sec	bbb-poky-sdk
●		Benchmark.ebizzy	150	N/A	N/A
●		Benchmark.ffsb	230	N/A	N/A
●		Benchmark.fio	140	N/A	N/A
●		Benchmark.GLMark	160	N/A	N/A
●		Benchmark.gtkperf	240	N/A	N/A

localhost:8080/fuego/view/Benchmarks/#



CE Workg

Individual test page

Functional.expat [Test Automation Framework] - Mozilla Firefox

Functional.expat [T... x +

localhost:8080/fuego/view/Functional/job/Functional.expat/

COGENT EMBEDDED THE LINUX FOUNDATION LONG TERM SUPPORT INITIATIVE RENESAS

Home Functional Functional.expat ENABLE AUTO REFRESH

Project Functional.expat

Expat built-in test suit

[edit description](#)
Disable Test

[Workspace](#)

[Recent Changes](#)

Test Run History [\(trend\)](#)

#1	Apr 3, 2016 4:25:21 PM
----	--

[RSS for all](#) [RSS for failures](#)

Page generated: Apr 3, 2016 4:25:21 PM. REST API Jenkins ver. 1.509.2



CE Workgroup

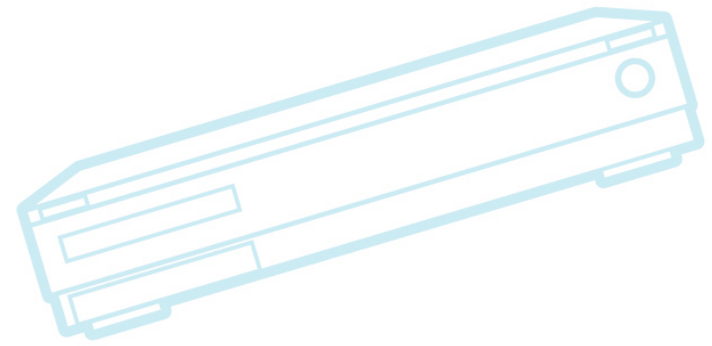
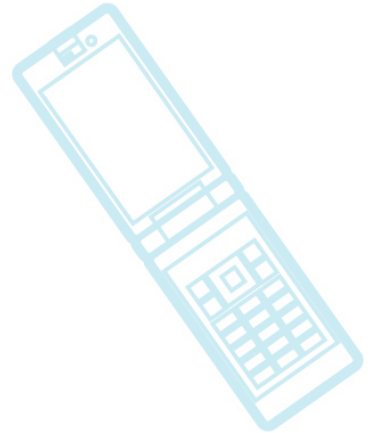
Outline

Introduction

Architecture

Customization

Vision

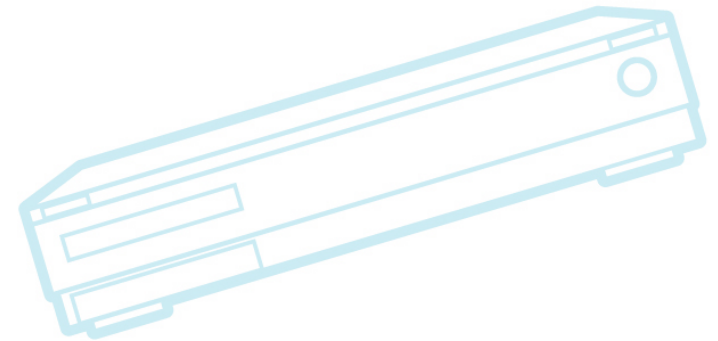
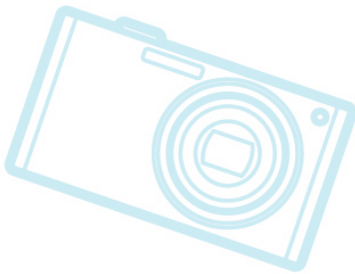
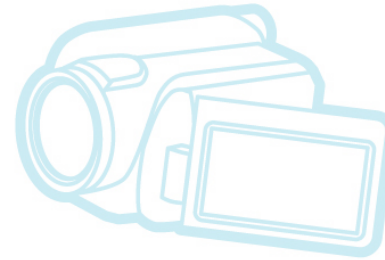
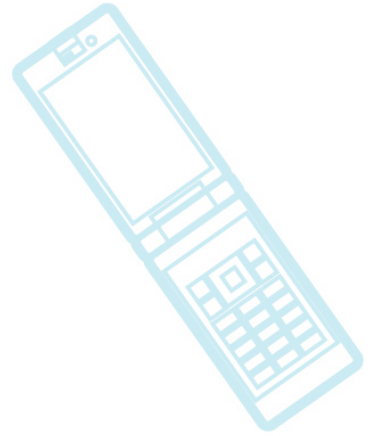




CE Workgroup

Customization

- Add a board configuration
- Add a toolchain
- Add a test

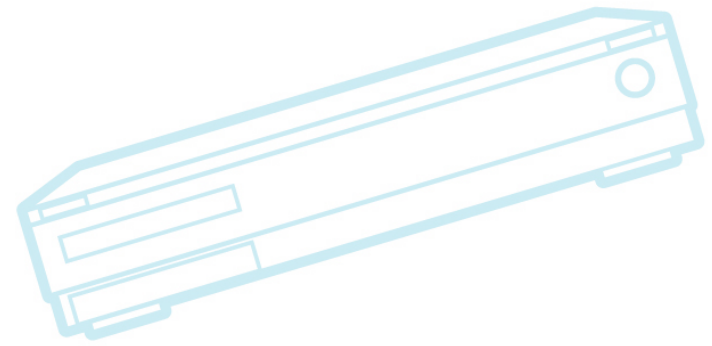
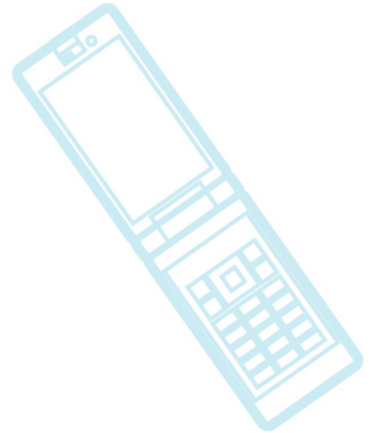




CE Workgroup

Add a board

- Overview:
 - Add a board file
 - Add the new target in the Jenkins interface





The board file

- Board file is a shell script with some variable that describe the board
- Create file in userdata/conf/boards, with filename “<target-name>.board”
 - There are examples there already
- Define IP address, ssh port, file system info (device, partitions, etc.)
- PLATFORM - indicates which SDK to use for building test programs



Board file sample (qemu-arm)

```
inherit "base-board"
include "base-params"

IPADDR="172.17.0.1"
SSH_PORT=5555
LOGIN="root"
FUEGO_HOME="/home/a"
PASSWORD="adm"
PLATFORM="qemu-armv7hf"
TRANSPORT="ssh"
ARCHITECTURE="arm"

SATA_DEV="/dev/sdb1"
SATA_MP="/mnt/sata"

USB_DEV="/dev/sda1"
USB_MP="/mnt/usb"

MMC_DEV="/dev/mmcblk0p2"
MMC_MP="/mnt/mmc"

LTP_OPEN_POSIX_SUBTEST_COUNT_POS="1319"
LTP_OPEN_POSIX_SUBTEST_COUNT_NEG="169"

EXPAT_SUBTEST_COUNT_POS="1769"
EXPAT_SUBTEST_COUNT_NEG="41"
```



Add the target in Jenkins

- Go to Target Status in main screen
- Select “New Node”
 - Enter name, and copy from “template-dev”
- Reference the board file
 - Set Environment Variable **BOARD_OVERLAY** to “boards/<target-name>.board”



Interface for adding a board

raspberrypi Configuration [Test Automation Framework] - Mozilla Firefox

localhost:8080/fuego/computer/raspberrypi/configure

COGENT EMBEDDED THE LINUX FOUNDATION LONG TERM SUPPORT INITIATIVE RENESAS

Home nodes raspberrypi

- Back to List
- Target Status
- Delete Target
- Configure Target**
- Test Run History
- Load Statistics
- Script Console
- Log
- System Information

Targets Status

#	Status

Name: blueberry-pi

Description:

of executors: 1

Remote FS root: /tmp/dev-slave1

Labels:

Usage: Utilize this target as much as possible

Launch method: Launch slave via execution of command on the Master

Launch command: java -jar /home/jenkins/slave.jar

Availability: Keep this slave on-line as much as possible

Node Properties

Environment variables

List of key-value pairs

name	BOARD_OVERLAY
value	boards/blue-pi.board
<input type="button" value="Delete"/>	
name	DISTRIB
value	distribsnologger.dist
<input type="button" value="Delete"/>	



CE Workgroup

Adding a toolchain

- Generic qemu ARM toolchain is pre-installed
- To install your own (overview):
 - Obtain or build your SDK
 - Install it inside the container in `/userdata/toolchains`
 - Modify `/userdata/conf/tools.sh` to reference it



Get SDK into the container

- To build the SDK in Yocto Project:
 - Inside your yocto build directory:
 - `bitbake <image-name> -c do_populate_sdk`
 - `docker ps` (*note the container id*)
 - `docker cp tmp/deploy/sdk/poky-*.sh <container-id>:/tmp`
- Install the SDK in the container:
 - At the shell inside the container:
 - `/tmp/poky-....sh`
 - (specify an installation path under `/userdata/toolchains`, like: `/userdata/toolchains/poky/2.0.1`)



Tell Fuego about SDK

- Add an entry to `/userdata/conf/tools.sh` for this toolchain
- Determine a platform name
- Add a new section to the `tools.sh`
 - Declare variables used by the toolchain in `userdata/conf/tools.sh` file
 - e.g. `PREFIX`, `ARCH`, `CC`, `AS`, `LD`, etc.
 - Can use a Yocto Project `environment_setup` script, and wrapper a few things
 - In this case, set `SDKROOT` variable
 - See `tools.sh` for examples
- Set `PLATFORM` environment variable in board file



Adding a test - overview

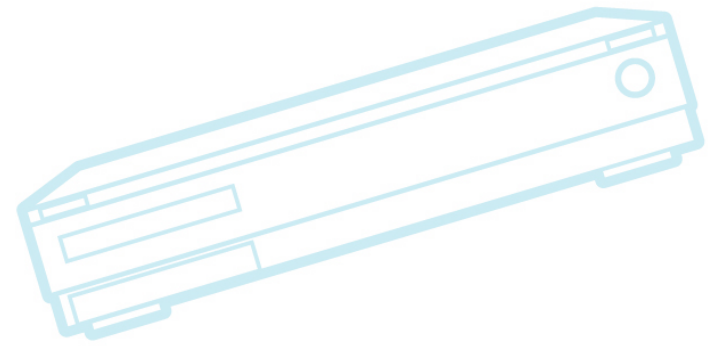
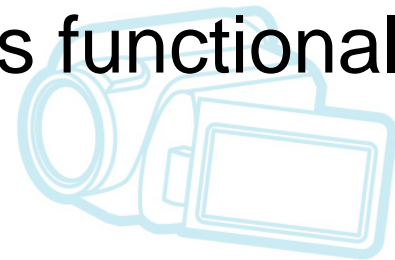
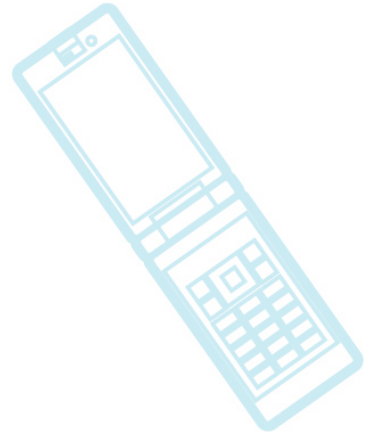
- A Fuego test consists of:
 - Actual test program (the thing that runs on the target)
 - Shipped as source
 - Test shell script
 - Results parser script (for benchmarks)
 - Results evaluator expression (for benchmarks)
 - Jenkins test declaration
- Test can be Functional or Benchmark



CE Workgroup

Functional tests

- Detects regressions
- Result is pass/fail
- Stress tests are defined as functional tests

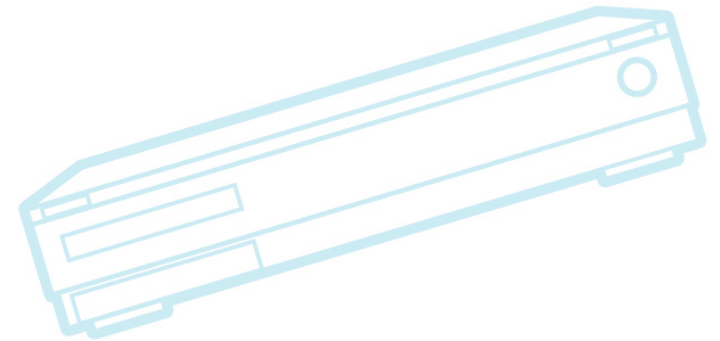
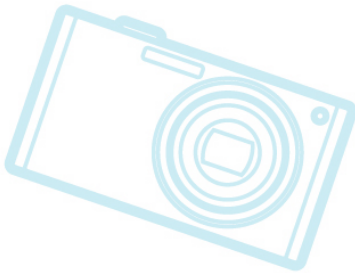
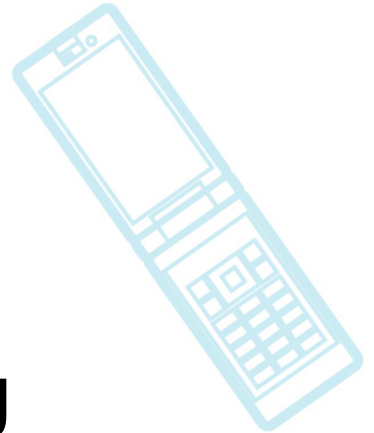




CE Workgroup

Benchmark tests

- Integrated plotting
- Parser to obtain value from test log
- Specification for data name and threshold for pass/fail





CE Workgroup

Test program

- Usually a pre-existing, compiled test program
- Source and patches are shipped in fuego-core repository
- Is cross-compiled by fuego for each target



Test script

- Shell script describes how to:
 - Build the test program (if applicable)
 - Deploy the test to the target
 - Execute the test on target, and collect results
 - Test for success or failure, by examining the log
- Specifically define the following functions:
 - test_build, test_deploy, test_run, test_processing
- Include a fuego engine script
- Script calls fuego functions to perform operations with the target



Fuego functions

- Fuego functions available in test scripts:
 - put/get – transfer files to/from target
 - cmd – execute command on target
 - report – execute command, and put results in log
 - log_compare – check log for a pattern, to check for pass or fail
 - hd_test_mount_prepare – mount a filesystem for a test
 - hd_test_clean_umount – unmount a filesystem after a test
- There are more
 - See examples in other scripts



Shell script example

CE

```
tarball=synctest.tar.gz

function test_build {
    make && touch test_suite_ready || build_error "error while building test"
}

function test_deploy {
    put synctest $FUEGO_HOME/fuego.$TESTDIR/
}

function test_run {
    assert_define FUNCTIONAL_SYNCTEST_MOUNT_BLOCKDEV
    assert_define FUNCTIONAL_SYNCTEST_MOUNT_POINT
    assert_define FUNCTIONAL_SYNCTEST_LEN
    assert_define FUNCTIONAL_SYNCTEST_LOOP

    hd_test_mount_prepare $FUNCTIONAL_SYNCTEST_MOUNT_BLOCKDEV \
        $FUNCTIONAL_SYNCTEST_MOUNT_POINT
    report "cd $FUNCTIONAL_SYNCTEST_MOUNT_POINT/fuego.\
        $TESTDIR; $FUEGO_HOME/fuego.$TESTDIR/synctest \
        $FUNCTIONAL_SYNCTEST_LEN \
        $FUNCTIONAL_SYNCTEST_LOOP"

    hd_test_clean_umount $FUNCTIONAL_SYNCTEST_MOUNT_BLOCKDEV \
        $FUNCTIONAL_SYNCTEST_MOUNT_POINT
}

function test_processing {
    log_compare "$TESTDIR" "1" "PASS : sync interrupted" "p"
}

. $FUEGO_SCRIPTS_PATH/functional.sh
```



Benchmark extras

- Extra files for plotting benchmark data
 - Parsing the test results (parser.py)
 - Extracts data from the log, using a regular expression, and formats it into a python map
 - Specifying a benchmark threshold for pass/fail
 - put an expression in reference.log file
- Modify userdata/logs/tests.info
 - Add a line describing the test and the results to plot
 - Use the name(s) emitted by parser.py



CE Workgroup

Plot example

Benchmark.dbench [Test Automation Framework] - Mozilla Firefox

localhost:8080/fuego/job/Benchmark.dbench/

COGENTEMBEDDED THE LINUX FOUNDATION LONG TERM SUPPORT INITIATIVE RENESAS

Home Benchmark.dbench ENABLE AUTO REFRESH

Back to Dashboard

Status

Changes

Workspace

Run Test Now

Delete Test

Configure Test

Documentation

Graph

Test Run History (trend)

- #15 Apr 3, 2016 6:20:48 PM bbb-poky-sdk / 3.8.13-bone50
- #14 Apr 3, 2016 6:16:19 PM bbb-poky-sdk / 3.8.13-bone50
- #13 Apr 3, 2016 6:14:48 PM bbb-poky-sdk / 3.8.13-bone50
- #12 Apr 3, 2016 6:13:06 PM bbb-poky-sdk / 3.8.13-bone50
- #11 Apr 3, 2016 5:59:45 PM bbb-poky-sdk / 3.8.13-bone50
- #10 Apr 3, 2016 4:52:47 PM bbb-poky-sdk / 3.8.13-bone50
- #9 Apr 3, 2016 4:51:21 PM bbb-poky-sdk / 3.8.13-bone50
- #8 Apr 3, 2016 4:50:56 PM bbb / 3.8.13-bone50
- #7 Apr 3, 2016 4:50:08 PM bbb-poky-sdk / 3.8.13-bone50
- #6 Apr 3, 2016 4:32:23 PM bbb-poky-sdk / 3.8.13-bone50
- #5 Apr 3, 2016 4:31:11 PM bbb-poky-sdk / 3.8.13-bone50

Project Benchmark.dbench

Dbench benchmark

[edit description](#)

Disable Test

dbench / Throughput

Legend:

- bbb-Throughput.Throughput
- bbb-poky-sdk-Throughput.Throughput
- bbb-Throughput.Th
- bbb-poky-sdk-Throughput.Th

All devices: All firmware:

- bbb
- 3.8.13-bone50
- bbb-poky-sdk

[Workspace](#)

[Recent Changes](#)

Permalinks

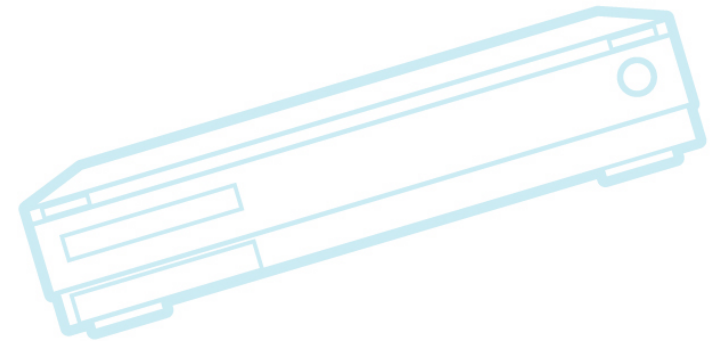
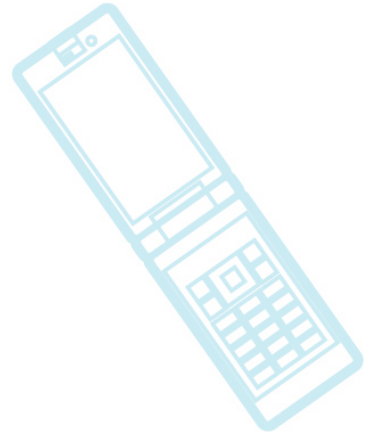
- Last test run (#15), 4 min 52 sec ago
- Last stable test run (#13), 10 min ago
- Last successful test run (#13), 10 min ago
- Last failed test run (#15), 4 min 52 sec ago



CE Workgroup

Outline

Introduction
Architecture
Customization
Vision





CE Workgroup

Vision

- Allow quick and easy setup
- Support a wide variety of configurations and build systems
 - (Yocto Project/OE, Buildroot, etc.)
- Support a wide variety of target types:
 - serial, ssh, adb, ttc
- Send data to centralized repository
- Make it possible to join a decentralized test network
 - Help solve the “developer can’t test on different hardware” problem



CE Workgroup

Next Steps

- De-clutter the Jenkins front end
- Improve documentation
- Handle USB connections
 - For ADB-based targets
 - For Sony debug board





CE Workgroup

Next Steps (cont.)

- More tests
 - kselftest
 - kernelci ??
 - Look for a vertical to build out the test suite
- Send results to a centralized repository



CE Workgroup

Resources

- <http://elinux.org/Fuego>
- <http://bird.org/fuego/FrontPage>
- Dedicated mail list (to come)
 - Using `LTSl-dev@lists.linuxfoundation.org` for now



CE Workgroup

Why “Fuego”?

- Former name was JTA (Jenkins-based Test Automation)
 - Not a very good name
- Fuego = Tierra del Fuego - one of the places on earth where penguins live
- Fuego = Fire – often associated with trials and purifying
- Fuego – it sounds neat



CE Workgroup

Fuego

It's hot!





CE Workgroup

Come play with Fuego!