Changes for the Better

for a greener tomorrow

Unveil How to Customize LTSI Test For Your Platform

Kengo IBE, Mitsubishi Electric

5th, October 2015

ELCE 2015 @Dublin

Who am I ?

- Kengo Ibe
- Embedded Linux Developer
  at the Mitsubishi Electric
  Information Technology R&D Center

- Loaned to Linux Foundation Since April 2015

- Joined Linux Foundation Collaborative Projects
  - LTSI : Long Term Support Initiative
  - AGL : Automotive Grade Linux

Outline

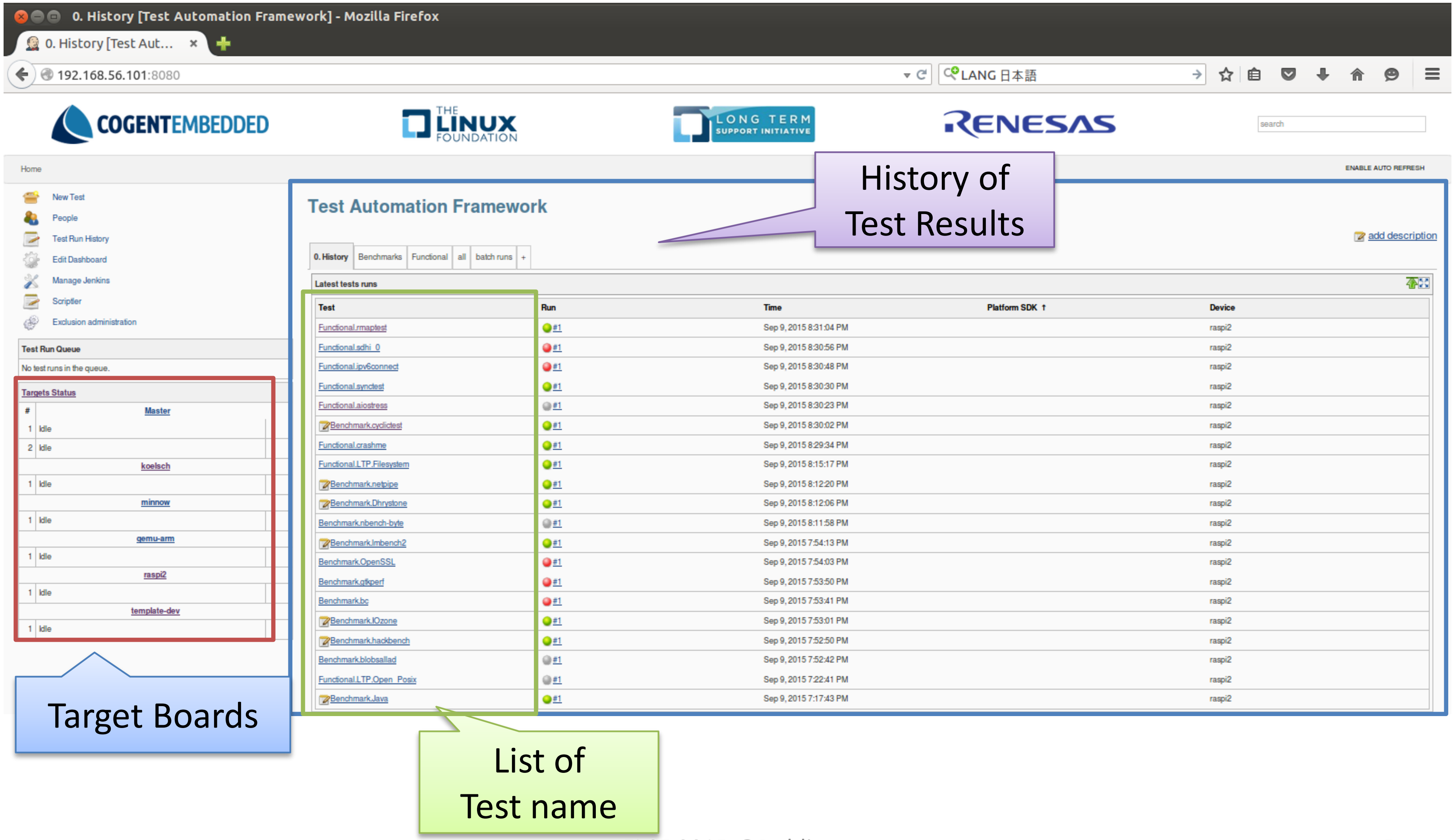
- What is the LTSI Project?
  - LTSI Test Environment
- How to Customize ?
  - Add New Board (Raspberry Pi2)
  - Add New Test Suite (LTP : Linux Test Project)
- Run LTP on Raspberry Pi2
- Summary & Future Works

What is LTSI Test Project?

- LTSI Project :
  - The project creates and maintains Linux Kernel which is expected to be stable in quality for the typical lifetime of a consumer electronics product, typically 2-3 years.
  - LTSI-4.1 Developing now
    - **Close Merge Window:** End of October
- LTSI Test Project
  - The project creates the LTSI Test Environment .
  - The LTSI Test Environment is Jenkins based automation test framework.
  - Including many test suites and kinds of target boards
    - 28 benchmarks and 33 functional test programs are already integrated
    - Minnow board(x86), koelsch(arm), quem-arm(QEMU) are already integrated
  - I hope to further increase the kind of target board, test suite.
  - I’m happy that many people will join this project.

LTSI Test Environment(Overview)

- Top of Web Interface LTSI Test Environment



Test Run Queue

No test runs in the queue.

Targets Status

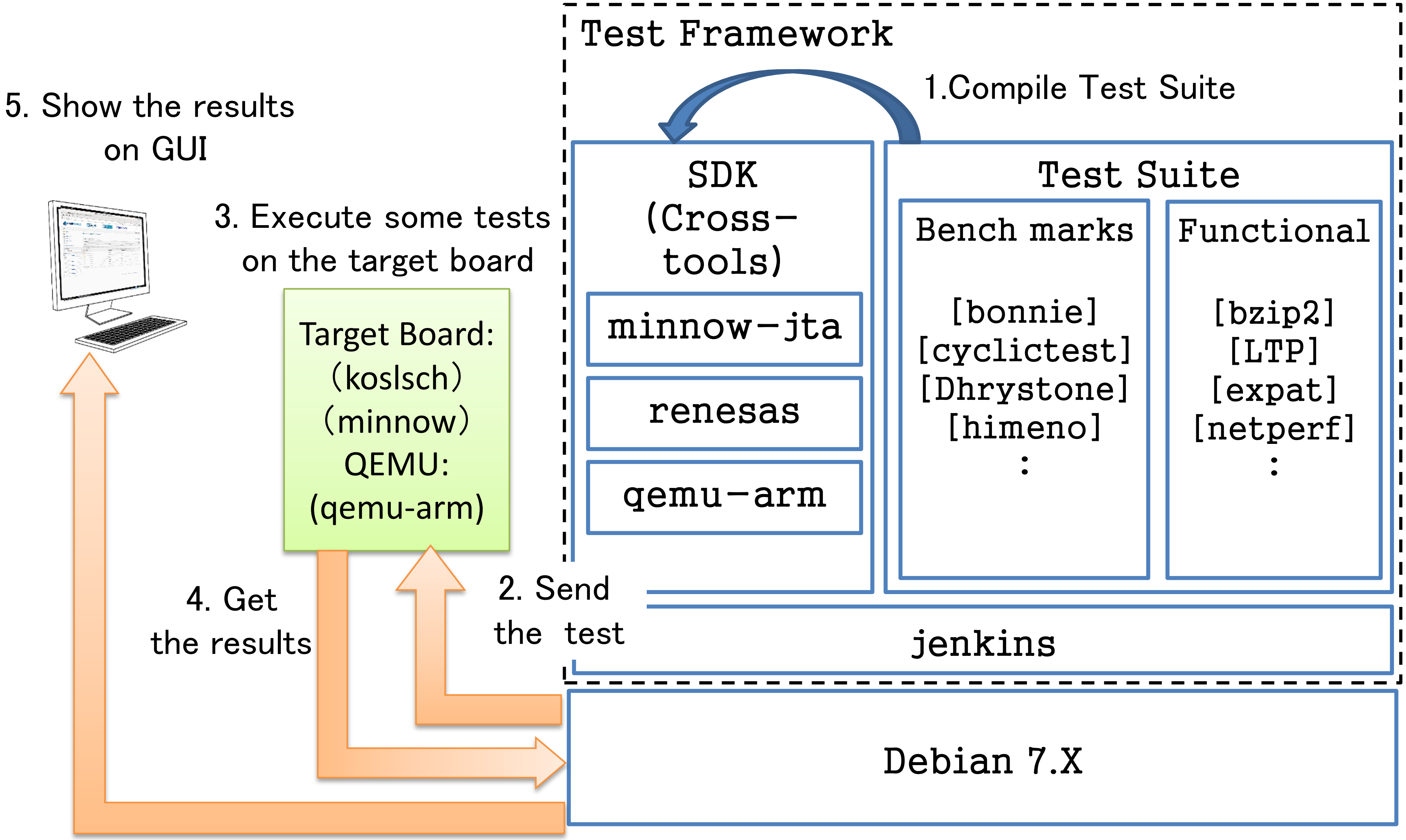
#	Master
1	Idle
2	Idle
	koelsch
1	Idle
	minnow
1	Idle
	gemu-arm
1	Idle
	raspi2
1	Idle
	template-dev
1	Idle

Test Automation Framework

Latest tests runs

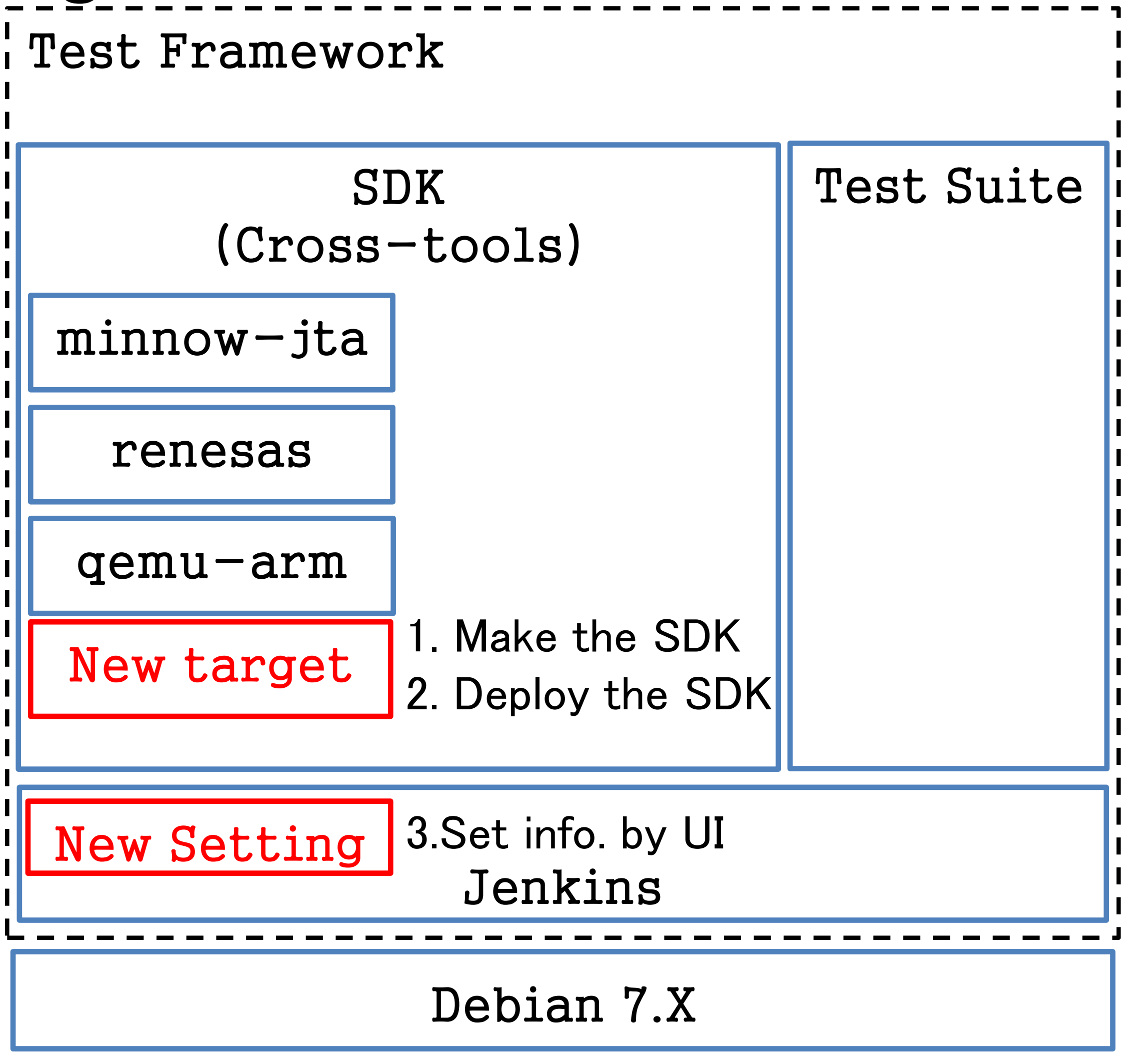
Test	Run	Time	Platform SDK ↑	Device
Functional.rmaptest	🟢 #1	Sep 9, 2015 8:31:04 PM		raspi2
Functional.sdhi_0	🔴 #1	Sep 9, 2015 8:30:56 PM		raspi2
Functional.ipv6connect	🔴 #1	Sep 9, 2015 8:30:48 PM		raspi2
Functional.synctest	🟢 #1	Sep 9, 2015 8:30:30 PM		raspi2
Functional.aiostress	● #1	Sep 9, 2015 8:30:23 PM		raspi2
Benchmark.cyclictest	🟢 #1	Sep 9, 2015 8:30:02 PM		raspi2
Functional.crashme	🟢 #1	Sep 9, 2015 8:29:34 PM		raspi2
Functional.LTP.Filesystem	🟢 #1	Sep 9, 2015 8:15:17 PM		raspi2
Benchmark.netpipe	🟢 #1	Sep 9, 2015 8:12:20 PM		raspi2
Benchmark.Dhrystone	🟢 #1	Sep 9, 2015 8:12:06 PM		raspi2
Benchmark.nbench-byte	● #1	Sep 9, 2015 8:11:58 PM		raspi2
Benchmark.lmbench2	🟢 #1	Sep 9, 2015 7:54:13 PM		raspi2
Benchmark.OpenSSL	🔴 #1	Sep 9, 2015 7:54:03 PM		raspi2
Benchmark.gtkperf	🔴 #1	Sep 9, 2015 7:53:50 PM		raspi2
Benchmark.bc	🔴 #1	Sep 9, 2015 7:53:41 PM		raspi2
Benchmark.IOzone	🟢 #1	Sep 9, 2015 7:53:01 PM		raspi2
Benchmark.hackbench	🟢 #1	Sep 9, 2015 7:52:50 PM		raspi2
Benchmark.blobsallad	● #1	Sep 9, 2015 7:52:42 PM		raspi2
Functional.LTP.Open_Posix	● #1	Sep 9, 2015 7:22:41 PM		raspi2
Benchmark.Java	🟢 #1	Sep 9, 2015 7:17:43 PM		raspi2

LTSI Test Environment(Flow)



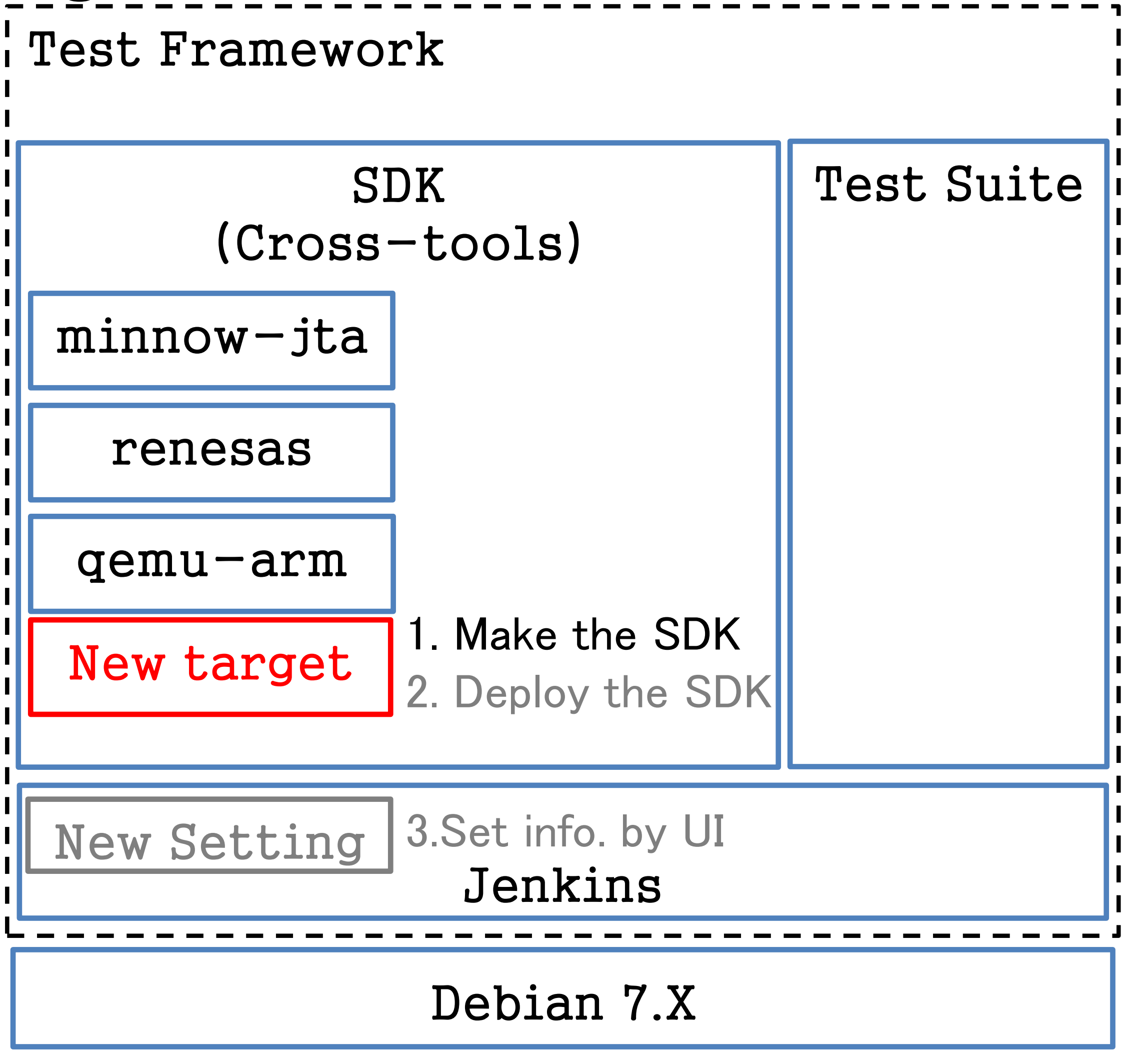
How to Customize ?(New Target)

- 3 step to add New Target
  - Make the SDK for the target
    - Using yocto project
  - Deploy the SDK into Test Framework
  - Set target Information by GUI



How to Customize ?(Raspberry pi 2)

- 3 step to add New Target
  - Make the SDK for the target
    - Using yocto project
  - Deploy the SDK into Test Framework
  - Set target Information by GUI



How to Customize ?(Raspberry pi 2)

- Make the SDK
  - Getting poky from Yocto project

```
$ git clone git://git.yoctoproject.org/poky.git
```

  - Getting meta-raspi and meta-jta
    - meta-raspi : For making a OS image and SDK for Raspberry pi2

```
$ git clone git://git.yoctoproject.org/meta-raspberrypi
```

    - meta-jta: For adding Headers and Libs for the Test Suite

```
$ git clone https://bitbucket.org/cogentembedded/meta-jta.git
```

How to Customize ?(Raspberry pi 2)

- Make the SDK (Cont'd)
  - _Configure the environment to build
    - Execute "**oe-init-build-env" script in Poky Directory**

```
poky$ source oe-init-build-env build-raspi2
```

    - Then cleated directory "**build-raspi2**"

```
build-raspi2$tree
 └ conf
        ├── bblayers.conf
        └── local.conf
```

  - "build-raspi2" includes a conf directory
  - There are "bblayers.conf" and "local.conf" in the conf directory

How to Customize ?(Raspberry pi 2)

- Make the SDK (Cont'd)
  - Setting to build(Cont'd)
    - Configure bblayers.conf for meta-raspberrypi & meta-jta

```
BBLAYERS ?= " ¥
  /home/melco/sdk/yocto/poky/meta ¥
  /home/melco/sdk/yocto/poky/meta-yocto ¥
  /home/melco/sdk/yocto/poky/meta-yocto-bsp ¥
  /home/melco/sdk/yocto/poky/meta-raspberrypi ¥
  /home/melco/sdk/yocto/poky/meta-jta ¥
```

Adding the path of **"meta-raspberrypi"** & **"meta-jta"**

- Configure local.conf for meta-raspi & meta-jta

```
#MACHINE ?= "genericx86-64"
#MACHINE ?= "mpc8315e-rdb"
#MACHINE ?= "edgerouter"
MACHINE ?= "raspberrypi2"
GPU_MEM = "16"
```

Setting **MACHINE** & **GPU Memory size** for raspi2

How to Customize ?(Raspberry pi 2)

- Make the SDK (Cont'd)
  - Build SDK

Execute "**bitbake meta-toolchain**" command in the build-raspi2 Directory

```
melco@debian-7:~/sdk/yocto/poky/build-raspi2$ bitbake meta-toolchain
Parsing recipes: 100%
|##########################################################################################
Parsing of 912 .bb files complete (0 cached, 912 parsed). 1341 targets, 61 skipped, 0 masked, 0
errors.
NOTE: Resolving any missing task queue dependencies

Build Configuration:
BB_VERSION        = "1.27.1"
BUILD_SYS         = "x86_64-linux"
NATIVELSBSTRING   = "Debian-7.8"
TARGET_SYS        = "arm-poky-linux-gnueabi"
MACHINE           = "raspberrypi2"
DISTRO            = "poky"
DISTRO_VERSION    = "1.8+snapshot-20150908"
TUNE_FEATURES     = "arm armv7a vfp thumb neon callconvention-hard vfpv4 cortexa7"
TARGET_FPU        = "vfp-vfpv4-neon"
meta
meta-yocto
meta-yocto-bsp    = "master:c1df471feacaf2590216aa476ce242908dac38cf"
meta-raspberrypi  = "master:17dad9328b100beda1cf870c9075e509b5cbfa90"
meta-jta          = "master:86387705bfe2ae9495bd661f0c4c7cead8fe06de"
```

To be able to verify **"MACHINE"** For raspi2

To be able to verify that "**bblayers.conf**" works

How to Customize ?(Raspberry pi 2)

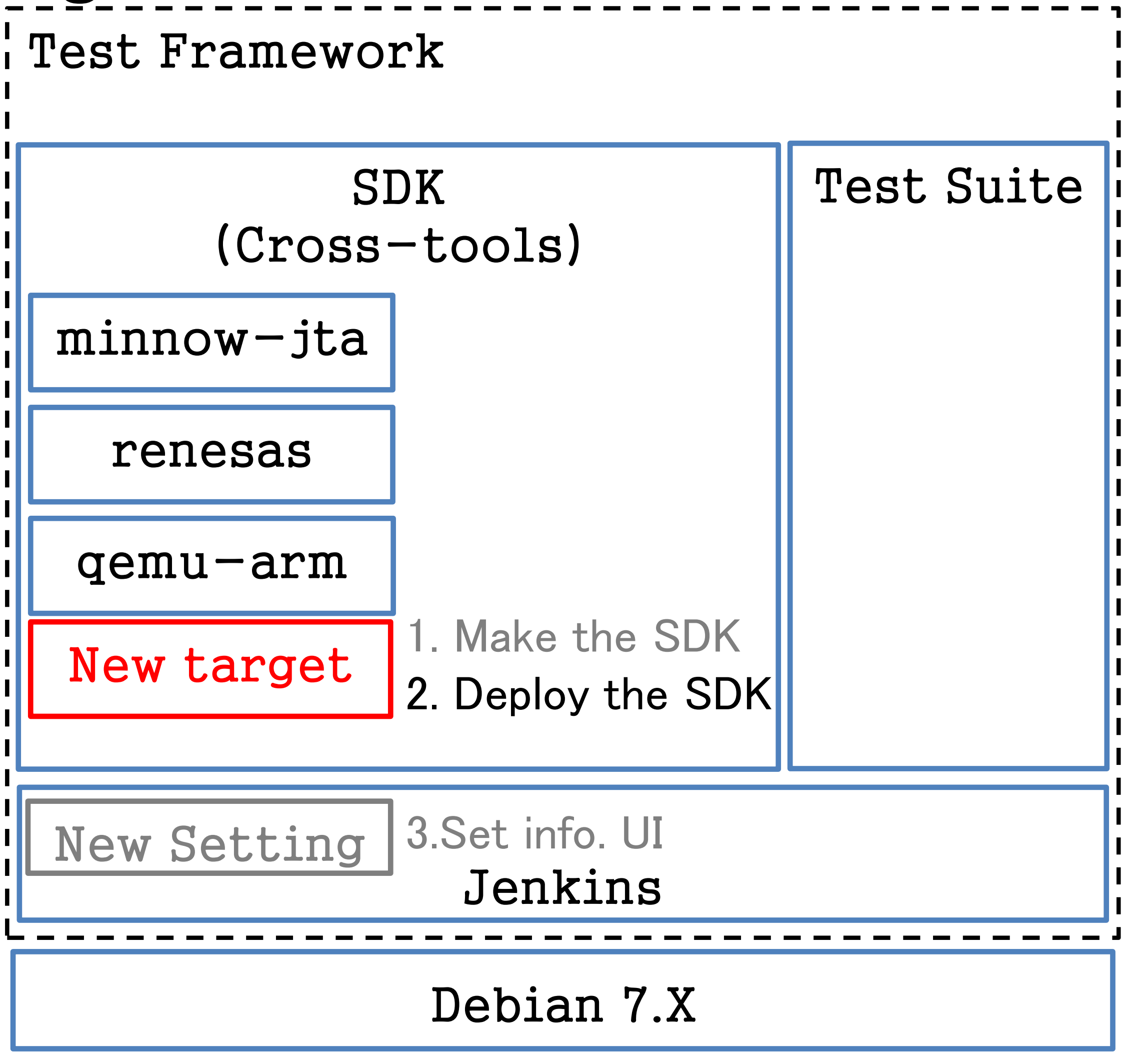
- Make the SDK (Cont’d)
  - Build SDK (Cont’d)
    - When building SDK finished,
      SDK install script is created at <Build Dir>/tmp/deploy/sdk/

```
melco@debian-7:~/sdk/yocto/poky/build-raspi2$ ls -al tmp/deploy/sdk/
合計 206104
drwxr-xr-x 2 melco melco      4096  9月  8 19:04 .
drwxr-xr-x 5 melco melco      4096  9月  8 14:45 ..
-rw------- 1 melco melco      9331  9月  8 19:04 poky-glibc-x86 [illegible]eta-toolchain-cortexa7hf-vfp-vfpv4-neon-toolchain-1.8+snapshot.host.manifest
-rwxr-xr-x 1 melco melco 103547364  9月  8 19:04 poky-glibc-x86_64-meta-toolchain-cortexa7hf-vfp-vfpv4-neon-toolchain-1.8+snapshot.sh
-rw------- 1 melco melco      1866  9月  8 19:03 poky-glibc-x86_64-meta-toolchain-cortexa7hf-vfp-vfpv4-neon-toolchain-1.8+snapshot.target.manifest
```

This file is the SDK install script.

How to Customize ?(Raspberry pi 2)

- 3 step to add New Target
  - Make the SDK for the target
    - Using yocto project
  - Deploy the SDK into Test Framework
  - Set target Information by GUI



How to Customize ?(Raspberry pi 2)

- Deploy the SDK into Test Framework
  - We can Deploy the SDK anywhere
    - This is the default Directory **/home/jenkins/tools/.** Minnow, qemu-arm and renesas-arm SDK are already in the directory.

```
melco@debian-7:~/sdk/yocto/poky/build-raspi2/tmp/deploy/sdk$ ./poky-glibc-x86_64-meta-toolchain-cortexa7hf-vfp-vfpv4-neon-toolchain-1.8+snapshot.sh -y -d /home/jenkins/tools/raspi2
Poky (Yocto Project Reference Distro) SDK installer version 1.8+snapshot
==================================================
The directory "/home/jenkins/tools/raspi2" already contains a SDK for this arc
If you continue, existing files will be overwritten! Proceed[y/N]? Y
[sudo] password for melco:
Extracting SDK...done
Setting it up...done
SDK has been successfully set up and is ready to be used.
Each time you wish to use the SDK in a new shell session, you need to source the environment
setup script e.g.
```

Selecting installing directory and run SDK install script.

How to Customize ?(Raspberry pi 2)

- Deploy the SDK into Test Framework(conf.)
  - Setting the Test Framework for the SDK
    - Adding raspi2 configuration on /home/Jenkins/scripts/tools.sh
    - The Test Framework already includes here minnow, qemu-arm and renesas-arm configurations.

```
elif [ "${PLATFORM}" = "raspi2" ];
then
      SDKROOT=$JTA_ENGINE_PATH/tools/raspi2/sysroots/cortexa7hf-vfp-vfpv4-neon-poky-linux-gnueabi
      # environment script changes PATH in the way that python uses libs from sysroot which is
        not what we want, so save it and use later
      ORIG_PATH=$PATH
      PREFIX=arm-poky-linux-gnueabi
      source $JTA_ENGINE_PATH/tools/raspi2/environment-setup-cortexa7hf-vfp-vfpv4-neon-poky-linux-gnueabi

      HOST=arm-poky-linux-gnueabi

      unset PYTHONHOME
      env -u PYTHONHOME
```

Setting raspi2

“SDKROOT” is the path of the sysroot that there is in deploying the SDK Directory.

Setting “**PREFIX**” for cross compile

Setting this path written the file of environment variable. This file is in the directory deploying the SDK. .

Set “**HOST**” for cross compile like “**PREFIX**”

How to Customize ?(Raspberry pi 2)

- Deploy the SDK into Test Framework(conf.)
  - Set of Test Framework for the Target(raspi2)
    - Adding raspi2 target board configuration on /home/jenkins/overlays/boards/<targetname>.board
    - A Sample target board configuration file is template-dev.board
    - When you add a new board,
      you should use template-dev.board

How to Customize ?(Raspberry pi 2)

- Deploy the SDK into Test Framework(conf.)

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

IPADDR="set_ip_here"
LOGIN="root"
JTA_HOME="/home/a"
PASSWORD=""
PLATFORM="set platform here (see tools.sh)"
TRANSPORT="ssh"
ARCHITECTURE="set_ia32_or_arm_here"
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"
```

Setting IP address of a target

Login user name

Directory to run some test

LOGIN user password

Setting Platform name
elif ["${PLATFORM}" = "raspi2"];

Setting Architecture name

Setting a device Information of the target board like SATA, USB and MMC etc.

Setting configuration For each test like LTP and EXPAT etc.

How to Customize ?(Raspberry pi 2)

- Deploy the SDK into Test Framework(conf.)
  - For example , <target name>.board for Raspi2

```
inherit "base-board"
include "base-params"
IPADDR="192.168.1.42"
LOGIN="root"
JTA_HOME="/home/a"
PASSWORD="pi"
PLATFORM="raspi2"
TRANSPORT="ssh"
ARCHITECTURE="arm“

MMC_DEV="/dev/mmcblk0p1"
MMC_MP="/mnt/mmc"
LTP_SYSCALL_COUNT_TPASS="4071"
LTP_SYSCALL_COUNT_TINFO="2776"
LTP_SYSCALL_COUNT_TCONF="140"
LTP_SYSCALL_COUNT_TFAIL="4"
LTP_SYSCALL_COUNT_TBROK="2764"
```

Setting IP address of a target

Login user name

Directory to run some test

LOGIN user password

Setting Platform name
elif ["${PLATFORM}" = "raspi2"];

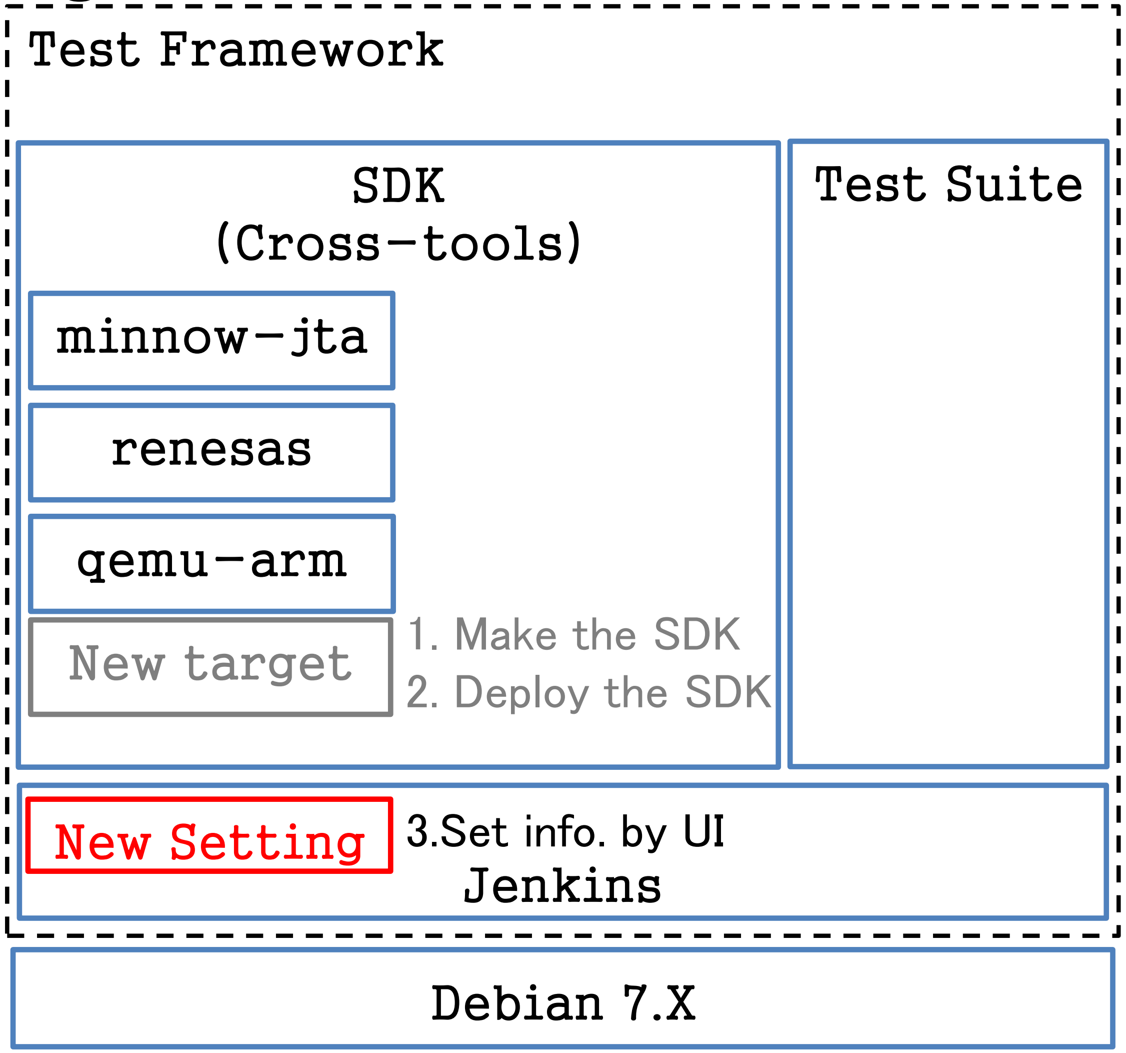
Setting Architecture name

Setting a MCC Information of Raspberry Pi2

Setting the configuration for LTP

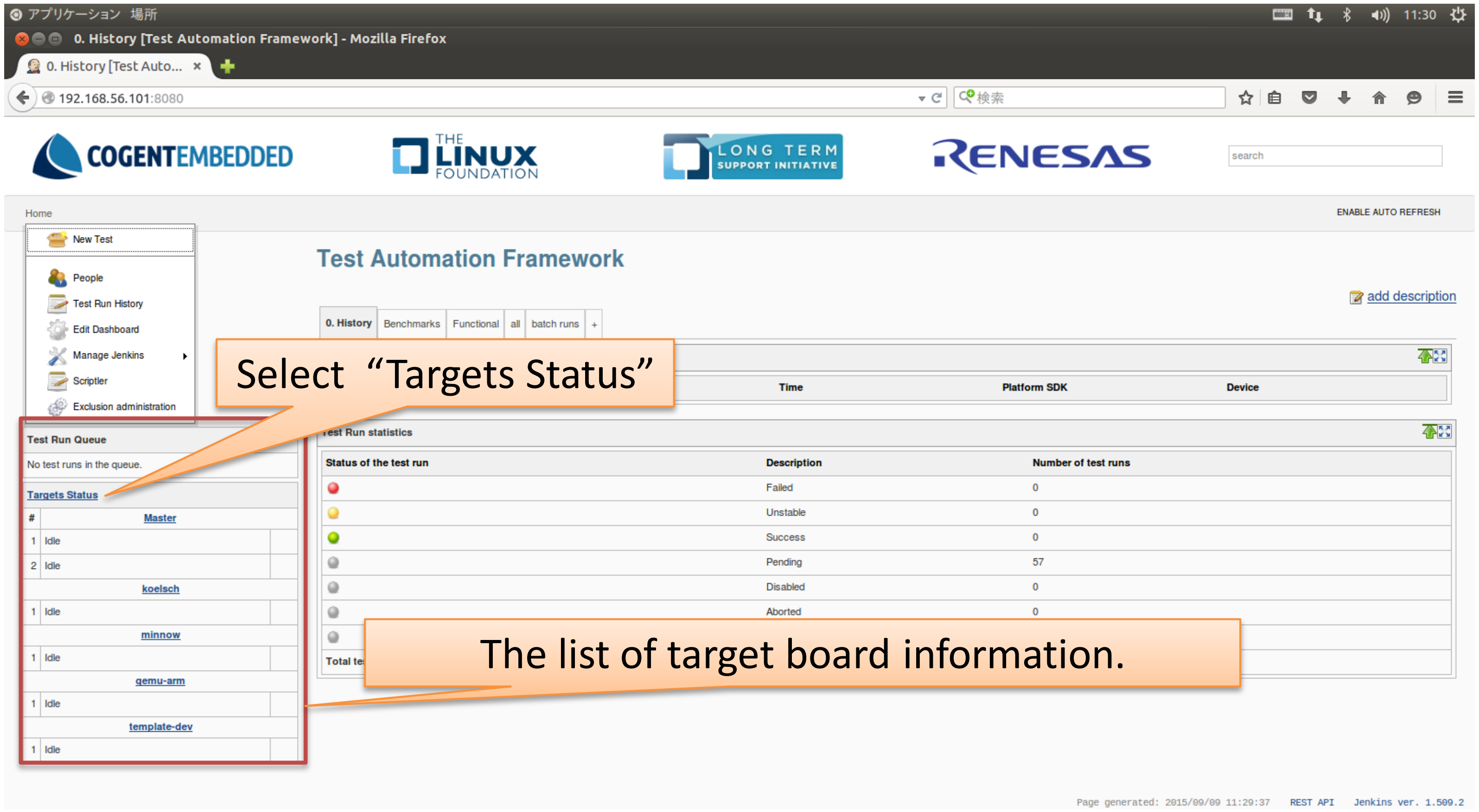
How to Customize ?(Raspberry pi 2)

- 3 step to add New Target
  - Make the SDK for target
    - Using yocto project
  - Deploy the SDK into Test Framework
  - Set target information by GUI



How to Customize ?(Raspberry pi 2)

- Set target Information by UI
  - Select “Targets Status” on top screen of Test Framework



0. History [Test Automation Framework] - Mozilla Firefox

192.168.56.101:8080

Home

ENABLE AUTO REFRESH

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

Test Automation Framework

add description

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

Select “Targets Status”

Time	Platform SDK	Device

Test Run Queue

No test runs in the queue.

Targets Status

#	Master
1	Idle
2	Idle
	koelsch
1	Idle
	minnow
1	Idle
	qemu-arm
1	Idle
	template-dev
1	Idle

Test Run statistics

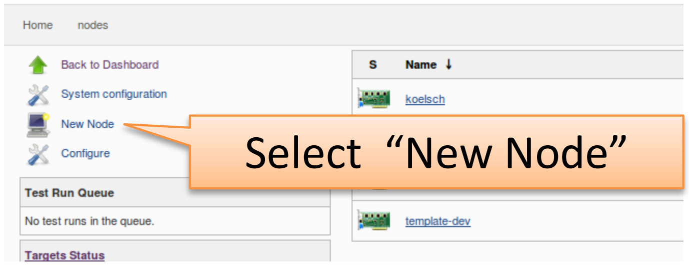
Status of the test run	Description	Number of test runs
●	Failed	0
●	Unstable	0
●	Success	0
●	Pending	57
●	Disabled	0
●	Aborted	0
●		
Total te		

The list of target board information.

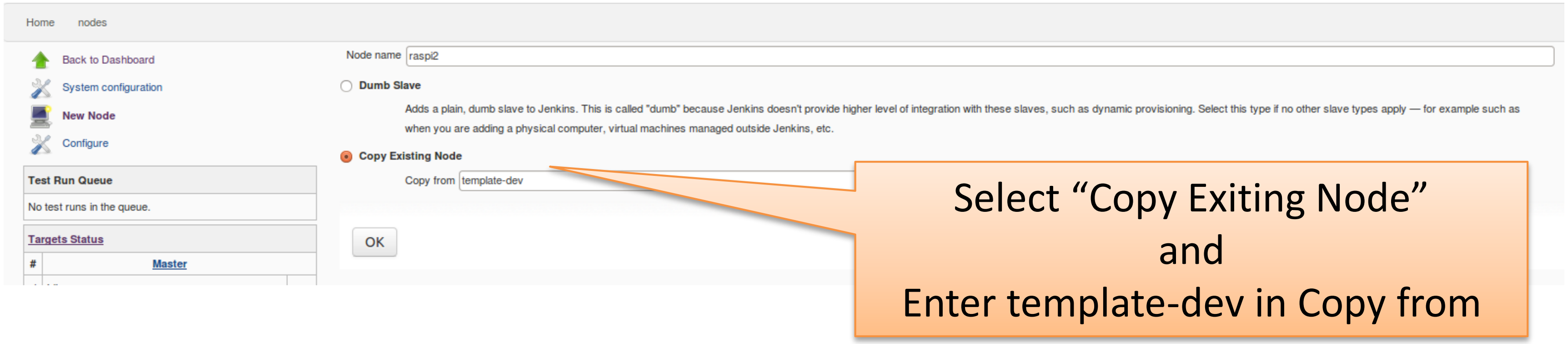
Page generated: 2015/09/09 11:29:37 REST API Jenkins ver. 1.509.2

How to Customize ?(Raspberry pi 2)

- Set target Information by UI(conf.)
  - Select “New Node”



  - Then, you can see a configuration form



Node name: raspi2

( ) Dumb Slave

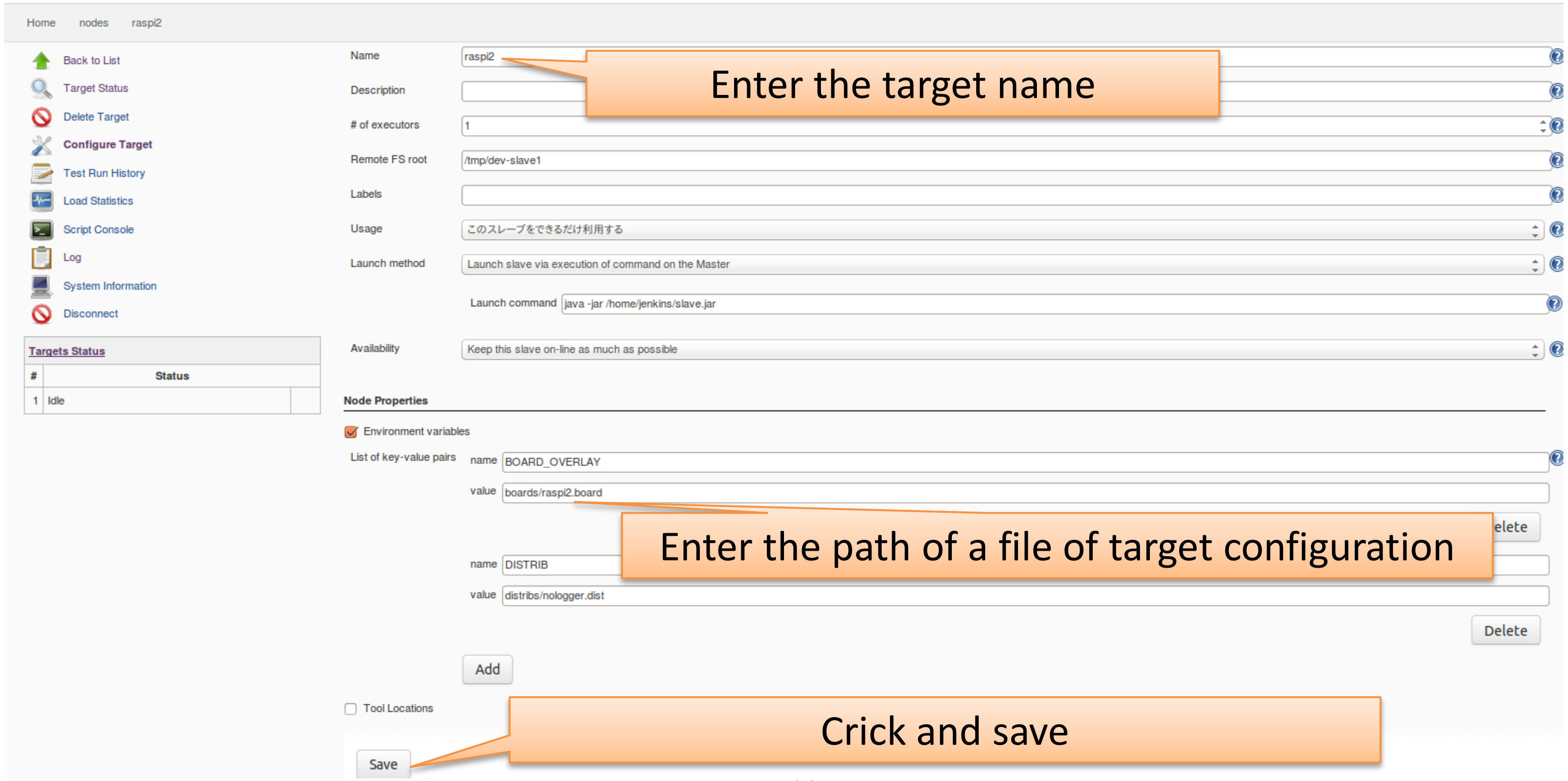
Adds a plain, dumb slave to Jenkins. This is called "dumb" because Jenkins doesn't provide higher level of integration with these slaves, such as dynamic provisioning. Select this type if no other slave types apply — for example such as when you are adding a physical computer, virtual machines managed outside Jenkins, etc.

(●) Copy Existing Node

Copy from: template-dev

How to Customize ?(Raspberry pi 2)

- Set target Information by UI(conf.)
  - You enter just 2 forms
    as "Name" and "List of Key-values pairs"



Home nodes raspi2

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

Targets Status

#	Status
1	Idle

Name	raspi2
Description	
# of executors	1
Remote FS root	/tmp/dev-slave1
Labels	
Usage	このスレーブをできるだけ利用する
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

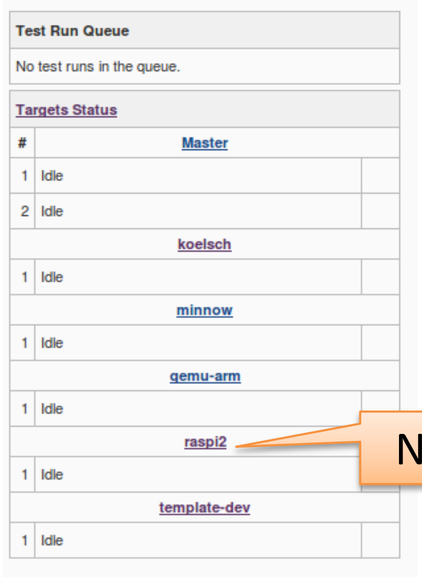
List of key-value pairs

name	BOARD_OVERLAY
value	boards/raspi2.board
name	DISTRIB
value	distribs/nologger.dist

- [ ] Tool Locations

How to Customize ?(Raspberry pi 2)

- Set target Information by UI(conf.)
  - You cat see a target list
    that New target board was added



Test Run Queue		
No test runs in the queue.		

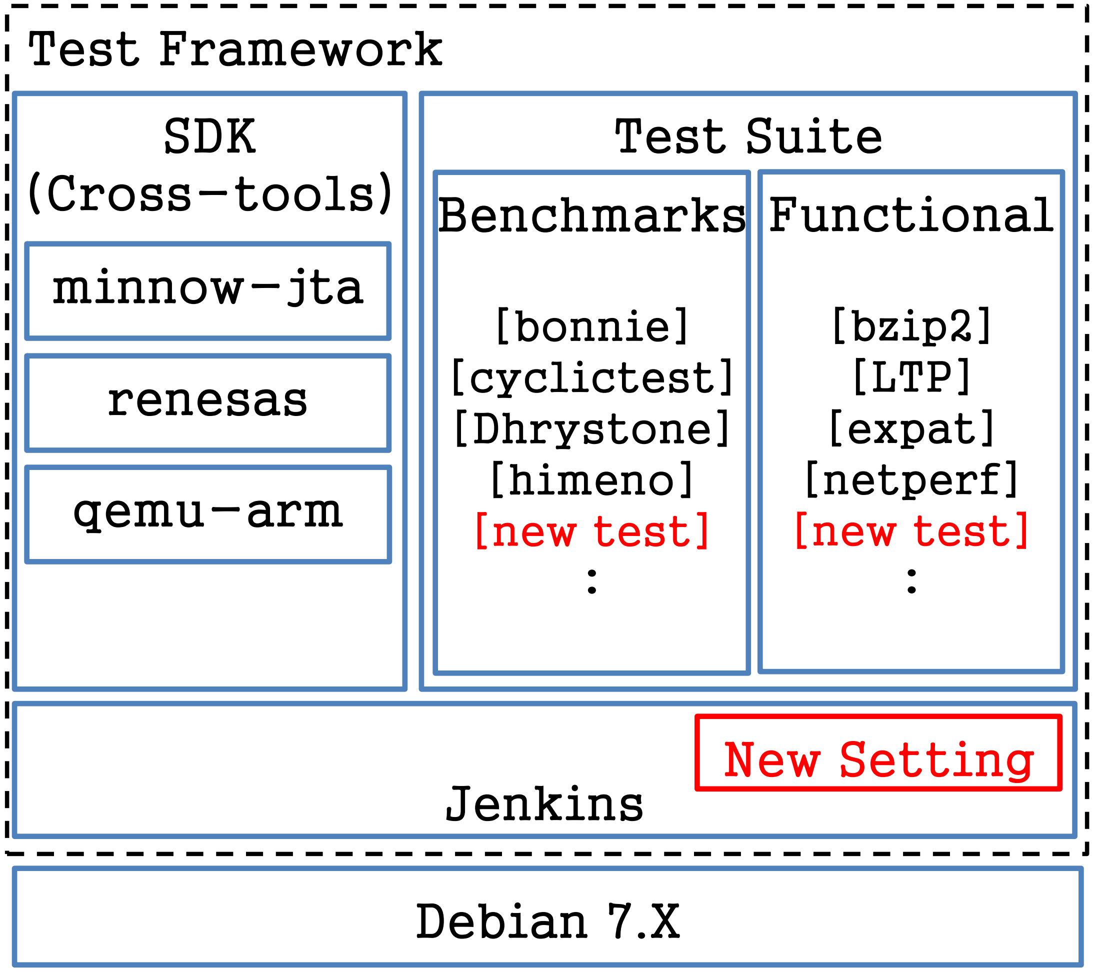
Targets Status		
#	Master	
1	Idle	
2	Idle	
	koelsch	
1	Idle	
	minnow	
1	Idle	
	qemu-arm	
1	Idle	
	raspi2	
1	Idle	
	template-dev	
1	Idle	

New target name is raspi2

Finish adding new target as Raspberry pi 2!!!

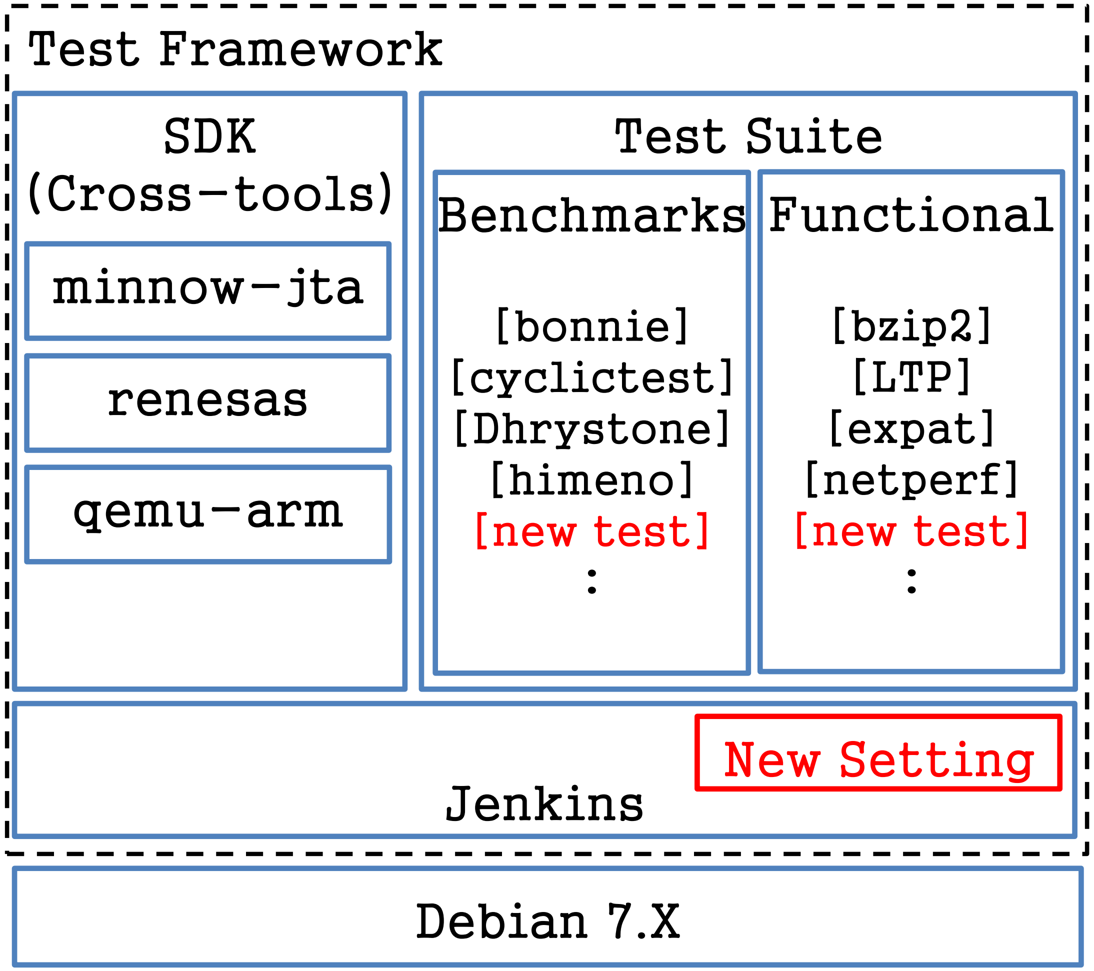
How to Customize ?(New Test Suite)

- 3 step to add New Test Suite
  - Create a script for running a new test suite
  - Deploy the script and a test suite tarball
  - Set the test suite information by GUI



How to Customize ?(New Test Suite)

- 3 step to add New Test Suite
  - Create a script for running a new test suite
  - Deploy the script and a test suite tarball
  - Set the test suite information by GUI



How to Customize ?(Linux Test Project)

- Create the script named “ltp-all.sh”（1）

```
tarball=ltp-full-20150420.tar.bz2

function test_build {
    make autotools
    ./configure CC="${CC}" AR="${AR}" RANLIB="${RANLIB}" LDFLAGS="$LDFLAGS" --
without-perl --without-python --target=$PREFIX --host=$PREFIX --
prefix=`pwd`/target_bin --build=`uname -m`-unknown-linux-gnu
    make CC="${CC}“
    make install
}

function test_deploy {
    put -r target_bin /tmp/jta.$TESTDIR/
}

function test_run {
    safe_cmd “cd /tmp/jta.$TESTDIR/target_bin; ./runltp –f syscalls |
    tee $JTA_HOME/jta.$TESTDIR/$TESTDIR.log”
}
```

To describe tarball name of the adding test suite

To describe procedure of creating test module using cross compile.

To describe procedure of deploying the test module to the target.

To describe commands to execute the test module on target.

In this case, to show running LTP command and collecting the result log.

How to Customize ?(Linux Test Project)

- Create the script named “ltp-all.sh” (conf.)

```
function test_processing {
## To judge test result
   assert_define LTP_SYSCALL_COUNT_TPASS
   assert_define LTP_SYSCALL_COUNT_TINFO
   assert_define LTP_SYSCALL_COUNT_TCONF
   assert_define LTP_SYSCALL_COUNT_TFAIL
   assert_define LTP_SYSCALL_COUNT_TBROK

   TPASS_CRIT="TPASS  :"
   TINFO_CRIT="TINFO  :"
   TCONF_CRIT="TCONF  :"
   TFAIL_CRIT="TFAIL  :"
   TBROK_CRIT="TBROK  :"

   log_compare “$TESTDIR” $LTP_SYSCALL_COUNT_TPASS "${TPASS_CRIT}" "TPASS"
   log_compare "$TESTDIR" $LTP_SYSCALL_COUNT_TINFO "${TINFO_CRIT}" "TINFO"
   log_compare "$TESTDIR" $LTP_SYSCALL_COUNT_TCONF "${TCONF_CRIT}" "TCONF"
   log_compare "$TESTDIR" $LTP_SYSCALL_COUNT_TFAIL "${TFAIL_CRIT}" "TFAIL"
   log_compare "$TESTDIR" $LTP_SYSCALL_COUNT_TBROK "${TBROK_CRIT}" "TBROK"

   echo "test_processing done“
}
. $JTA_ENGINE_PATH/scripts/functional.sh
```

To describe judgment and analysis process of test results

Verify definitions

Define Keywords to search in the log

Compare definitions and result log

Define on “<target name>.board”

ltp-all.sh is inherited functional.sh
The above functions are called by it.

How to Customize ?(Linux Test Project)

- ltp-all.sh is inherited functional.sh.
  - "functional.sh" is defined on LTSI test by default.

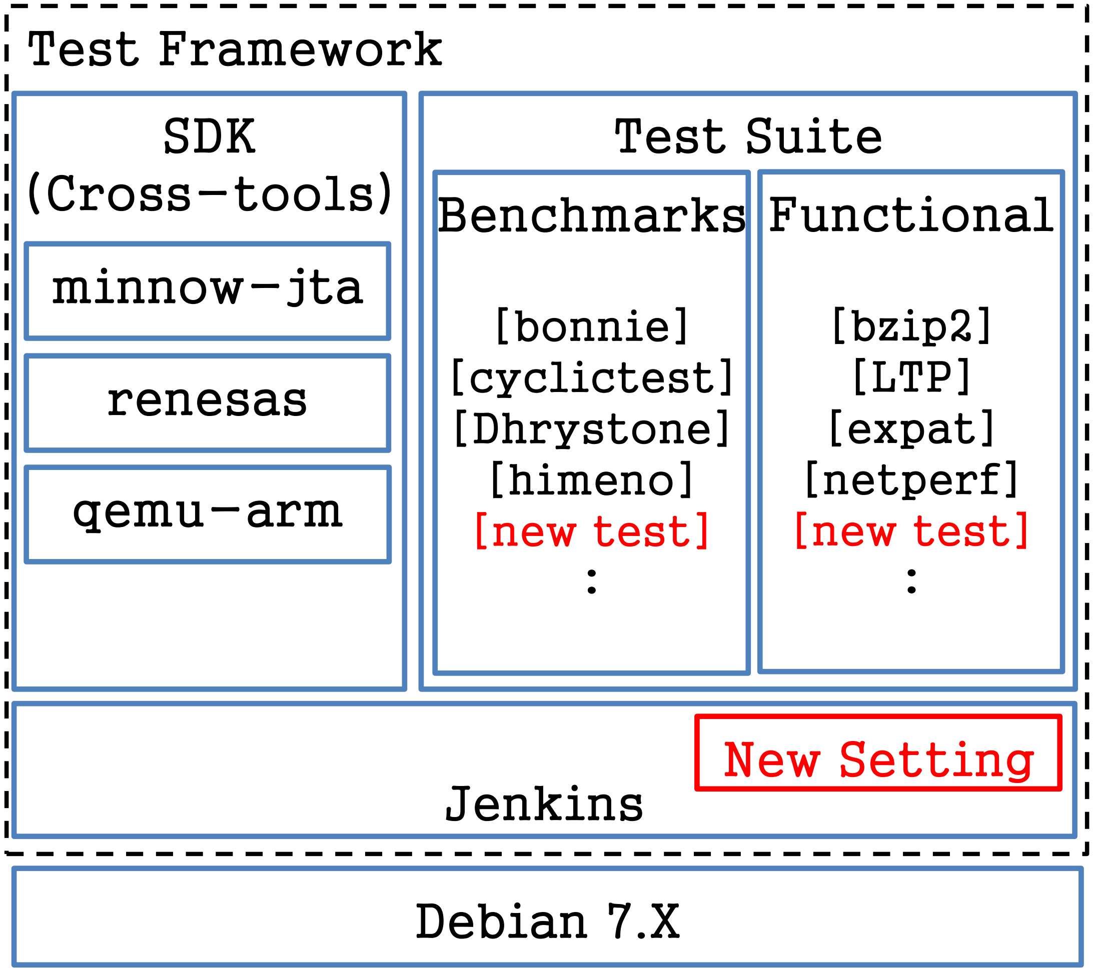
```
source $JTA_ENGINE_PATH/scripts/overlays.sh
set_overlay_vars

source $JTA_ENGINE_PATH/scripts/reports.sh
source $JTA_ENGINE_PATH/scripts/functions.sh

pre_test $TESTDIR

if $Rebuild; then
    build
fi

deploy

test_run

get_testlog $TESTDIR

test_processing
```

To include common scripts and execute overlay using Test plan and spec files. Test plan and Spec files provide the very flexibility in configuring tests to be run on different boards and scenarios in the Test Framework.

Standard sequence for running test script on the Test Framework.
- "Pre_test" is checking precondition.
- "Build" is executing test_build function.
- "Deploy" is executing test_deploy function.
- "Get_testlog" is getting the executing log.
- "test_run" and "test_processing" are difined on "ltp-all.sh".

How to Customize ?(New Test Suite)

- 3 step to add New Test Suite
  - Create a script for running a new test suite
  - Deploy the script and a test suite tarball
  - Set the test suite information by GUI



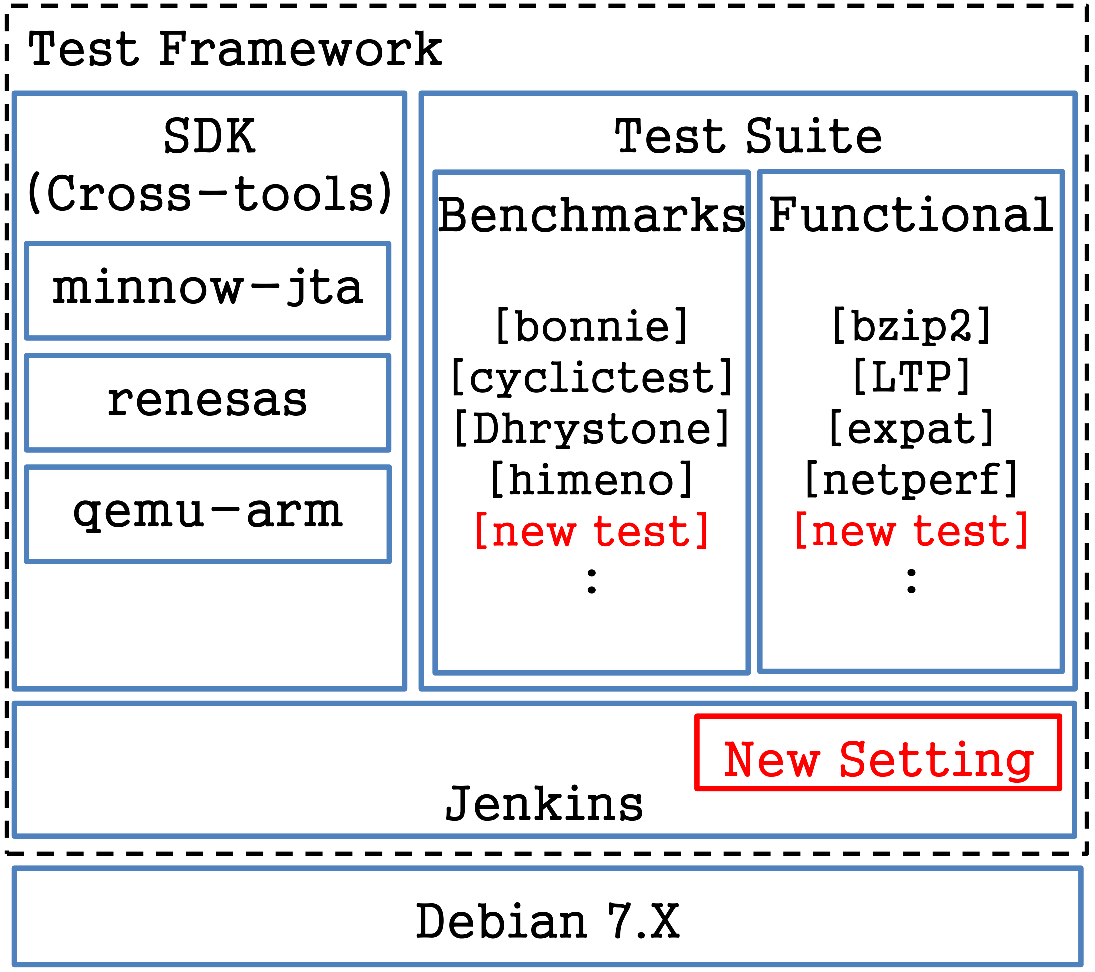
How to Customize ?(Linux Test Project)

- Deploy the script and test suite tarball
  - To create work directory "Functional.LTP.all" under /home/jenkins/tests/.
    - Arbitrary directory name can be used but the above is standard.
  - To obtain tarball of LTP from the below site
    - https://github.com/linux-test-project/ltp/releases/tag/20150420
  - To put the created script and tarball under Functional.LTP.all.

```
melco@debian-7:/home/jenkins/tests/ Functional.LTP.all$ ls
ltp-all.sh ltp-full-20150420.tar.bz2
```

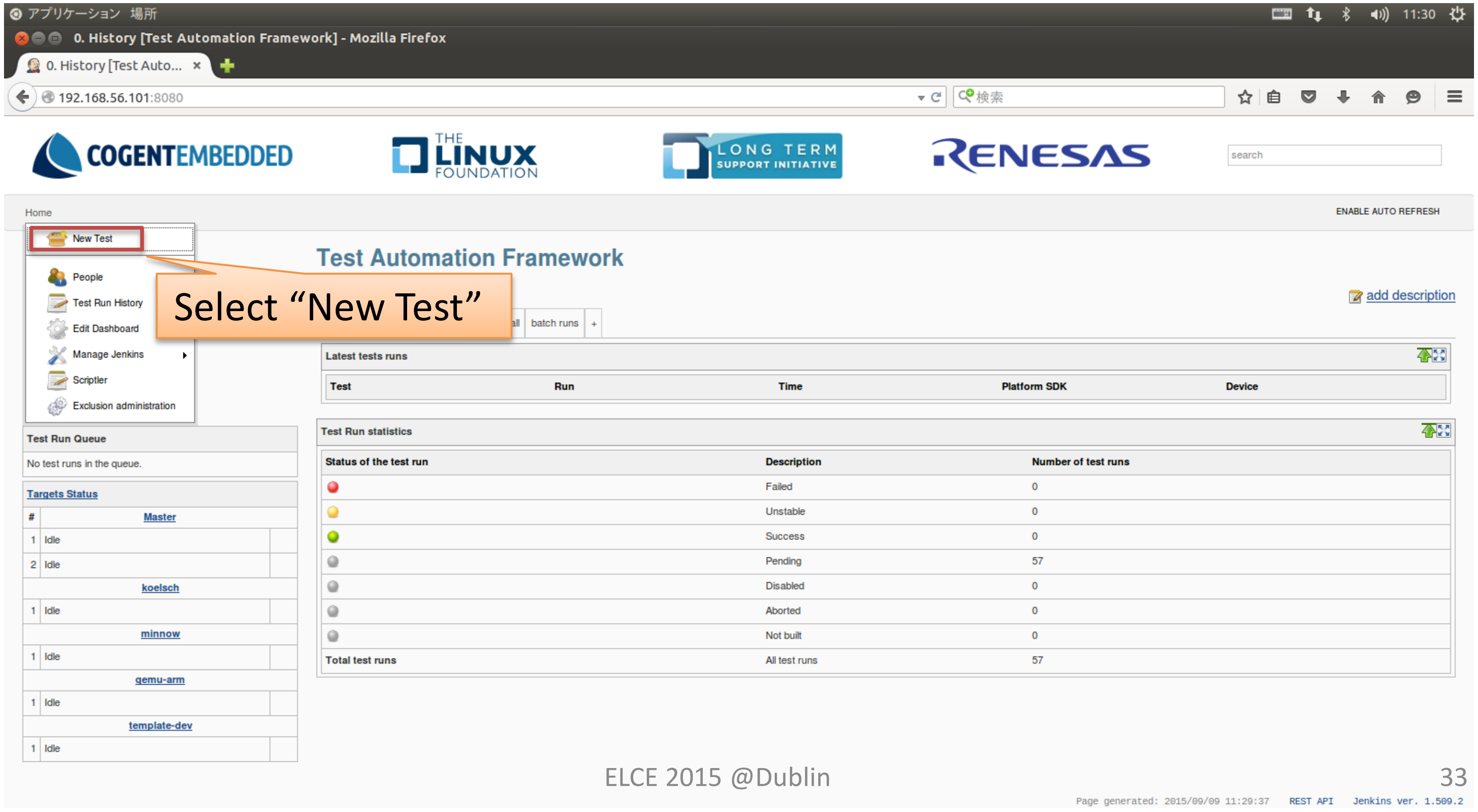
How to Customize ?(New Test Suite)

- 3 step to add New Test Suite
  - Create a script for running a new test suite
  - Deploy the script and a test suite tarball
  - Set the test suite information by GUI



How to Customize ?(Linux Test Project)

- Set test suite Information by GUI
  - To select "New Test" on the left side of screen of Test Framework



Test Automation Framework

Latest tests runs

Test	Run	Time	Platform SDK	Device

Test Run statistics

Status of the test run	Description	Number of test runs
🔴	Failed	0
🟡	Unstable	0
🟢	Success	0
●	Pending	57
●	Disabled	0
●	Aborted	0
●	Not built	0
Total test runs	All test runs	57

Test Run Queue

No test runs in the queue.

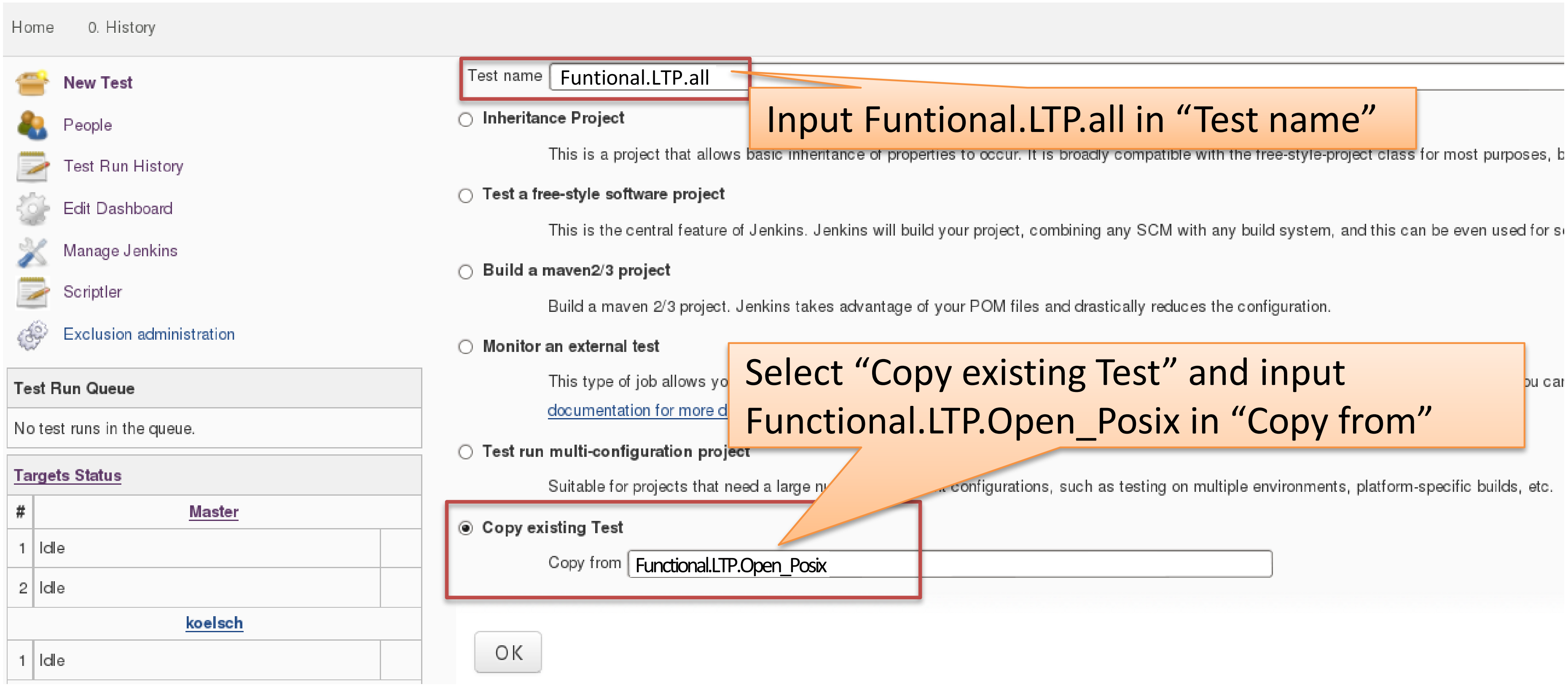
Targets Status

#	Master	
1	Idle	
2	Idle	
	koelsch	
1	Idle	
	minnow	
1	Idle	
	qemu-arm	
1	Idle	
	template-dev	
1	Idle	

Page generated: 2015/09/09 11:29:37 REST API Jenkins ver. 1.509.2

How to Customize ?(Linux Test Project)

- Set test suite Information by GUI
  - To input Test name
  - To chose “Copy existing Test” and Copy from



Test name: Funtional.LTP.all

Input Funtional.LTP.all in “Test name”

( ) Inheritance Project
This is a project that allows basic inheritance of properties to occur. It is broadly compatible with the free-style-project class for most purposes, b

( ) Test a free-style software project
This is the central feature of Jenkins. Jenkins will build your project, combining any SCM with any build system, and this can be even used for s

( ) Build a maven2/3 project
Build a maven 2/3 project. Jenkins takes advantage of your POM files and drastically reduces the configuration.

( ) Monitor an external test
This type of job allows yo ... ou car
documentation for more d

( ) Test run multi-configuration project
Suitable for projects that need a large n ... configurations, such as testing on multiple environments, platform-specific builds, etc.

(●) Copy existing Test
Copy from: Functional.LTP.Open_Posix

Select “Copy existing Test” and input Functional.LTP.Open_Posix in “Copy from”

Test Run Queue

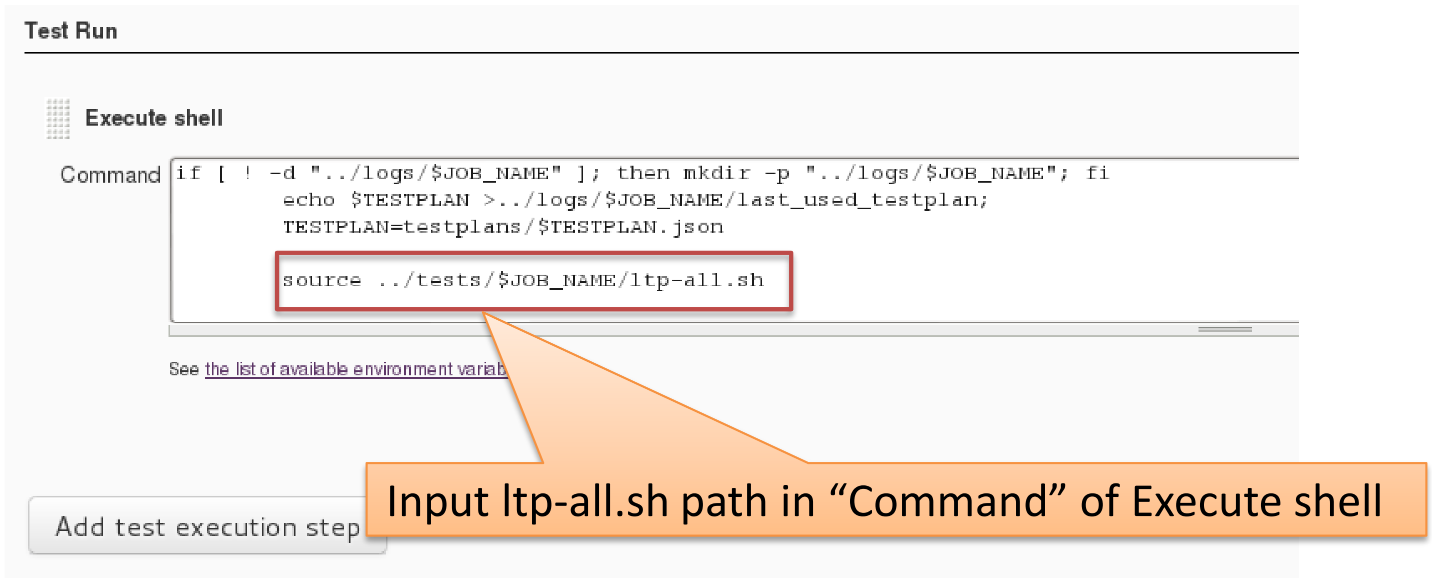
No test runs in the queue.

Targets Status

#	Master
1	Idle
2	Idle
	koelsch
1	Idle

How to Customize ?(Linux Test Project)

- Set test suite Information by GUI
  - To input the created script path in “Command” field of “Execute shell” of “Test Run”



Test Run

Execute shell

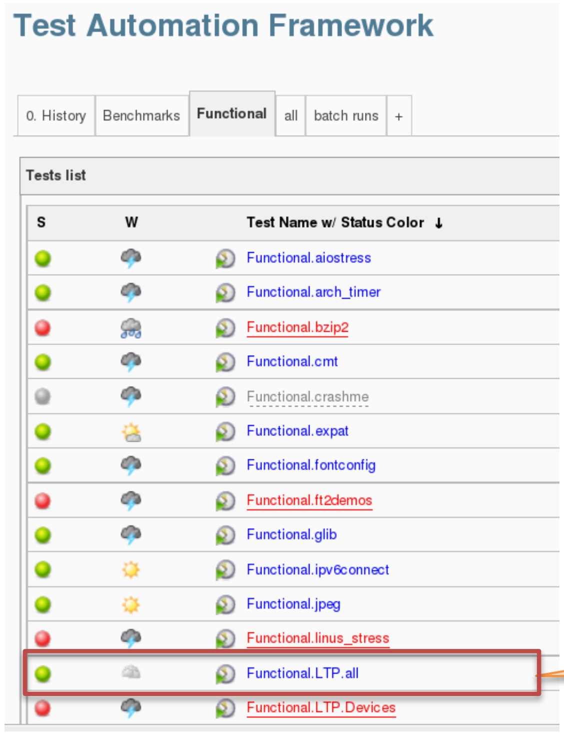
Command

```
if [ ! -d "../logs/$JOB_NAME" ]; then mkdir -p "../logs/$JOB_NAME"; fi
        echo $TESTPLAN >../logs/$JOB_NAME/last_used_testplan;
        TESTPLAN=testplans/$TESTPLAN.json

        source ../tests/$JOB_NAME/ltp-all.sh
```

See the list of available environment variab

How to Customize ?(Linux Test Project)

- Set test suite Information by GUI
  - You can see new test suite name in Functional Tab



Test Automation Framework

Tests list

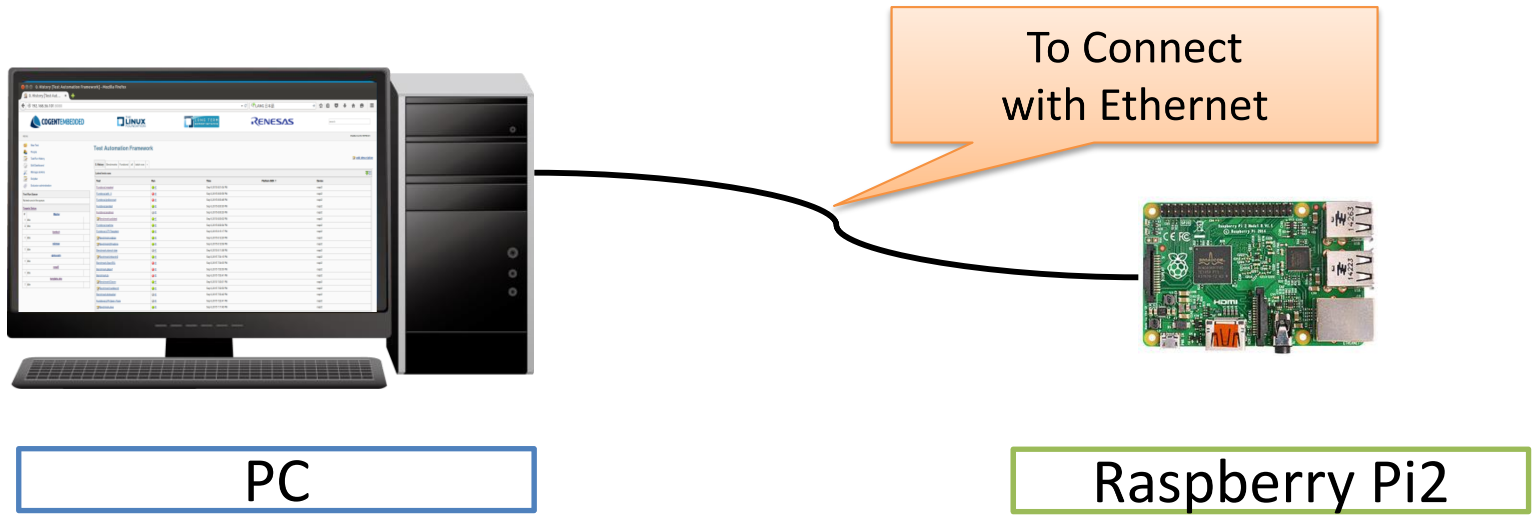
S	W	Test Name w/ Status Color ↓
🟢		Functional.aiostress
🟢		Functional.arch_timer
🔴		Functional.bzip2
🟢		Functional.cmt
●		Functional.crashme
🟢		Functional.expat
🟢		Functional.fontconfig
🔴		Functional.ft2demos
🟢		Functional.glib
🟢		Functional.ipv6connect
🟢		Functional.jpeg
🔴		Functional.linus_stress
🟢		Functional.LTP.all
🔴		Functional.LTP.Devices

New test suite

Finish adding new test suite as Linux Test Project !!!

Run new LTP on Raspberry Pi2

- Test Environment

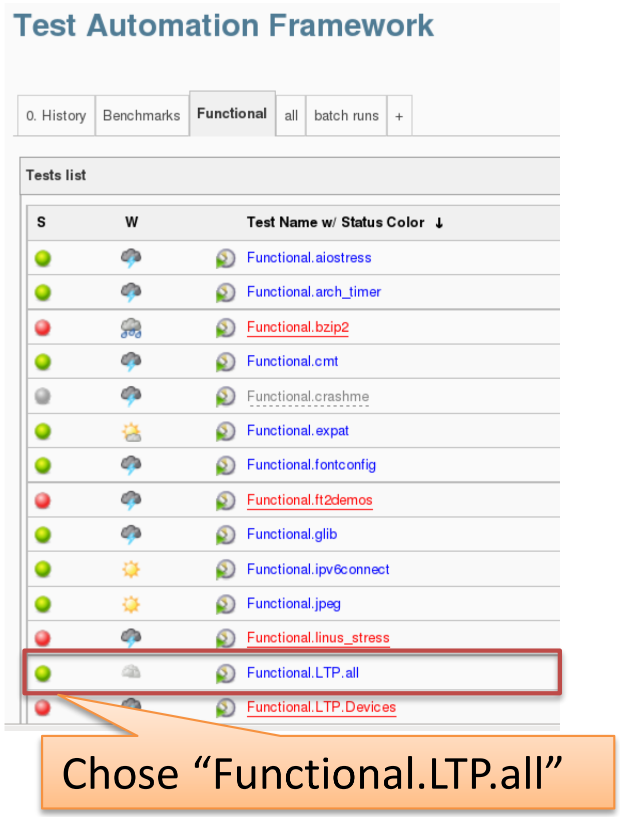


- Running Test Framework on Debian7.8
- Show the test result by web browser
- Adding the SDK Raspberry pi2
- Adding the LTP-20150420

- Running poky 1.8 from Yocto project
- Kernel version: 3.18

Run new LTP on Raspberry Pi2 (Cont'd)

- To select LTP-20150420
  - To chose Funcfional.LTP.all
  - To chose "Run Test Now"



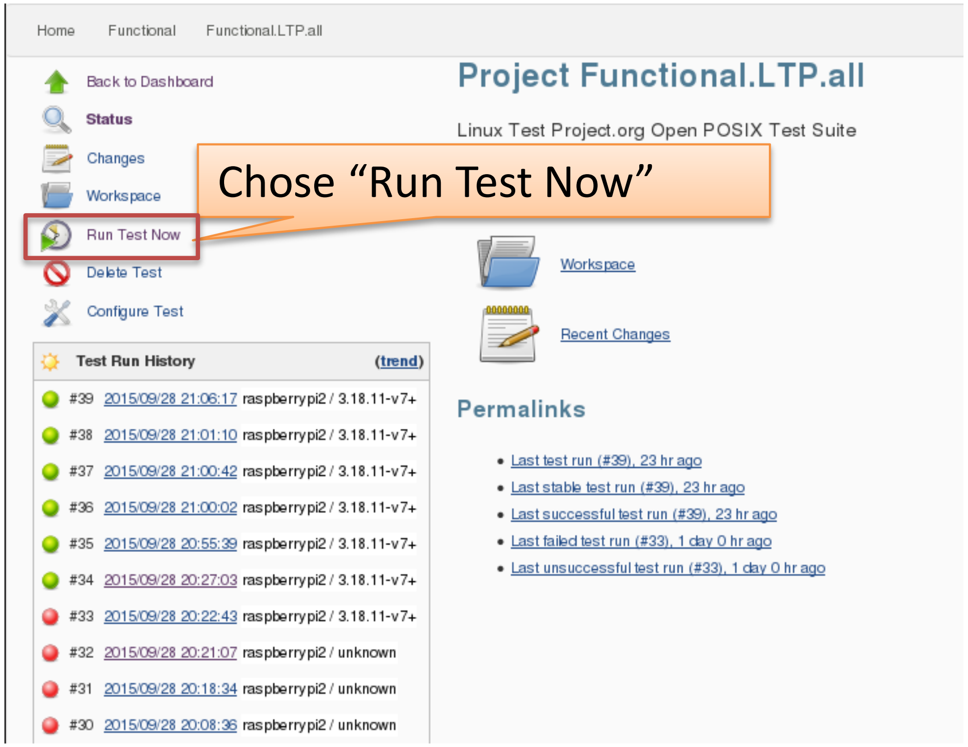
Test Automation Framework

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

Tests list

S	W	Test Name w/ Status Color ↓
🟢		Functional.aiostress
🟢		Functional.arch_timer
🔴		Functional.bzip2
🟢		Functional.cmt
⚪		Functional.crashme
🟢		Functional.expat
🟢		Functional.fontconfig
🔴		Functional.ft2demos
🟢		Functional.glib
🟢		Functional.ipv6connect
🟢		Functional.jpeg
🔴		Functional.linus_stress
🟢		Functional.LTP.all
🔴		Functional.LTP.Devices

Chose "Functional.LTP.all"



Home | Functional | Functional.LTP.all

- Back to Dashboard
- Status
- Changes
- Workspace
- Run Test Now
- Delete Test
- Configure Test

Chose "Run Test Now"

Project Functional.LTP.all

Linux Test Project.org Open POSIX Test Suite

Workspace

Recent Changes

Test Run History (trend)

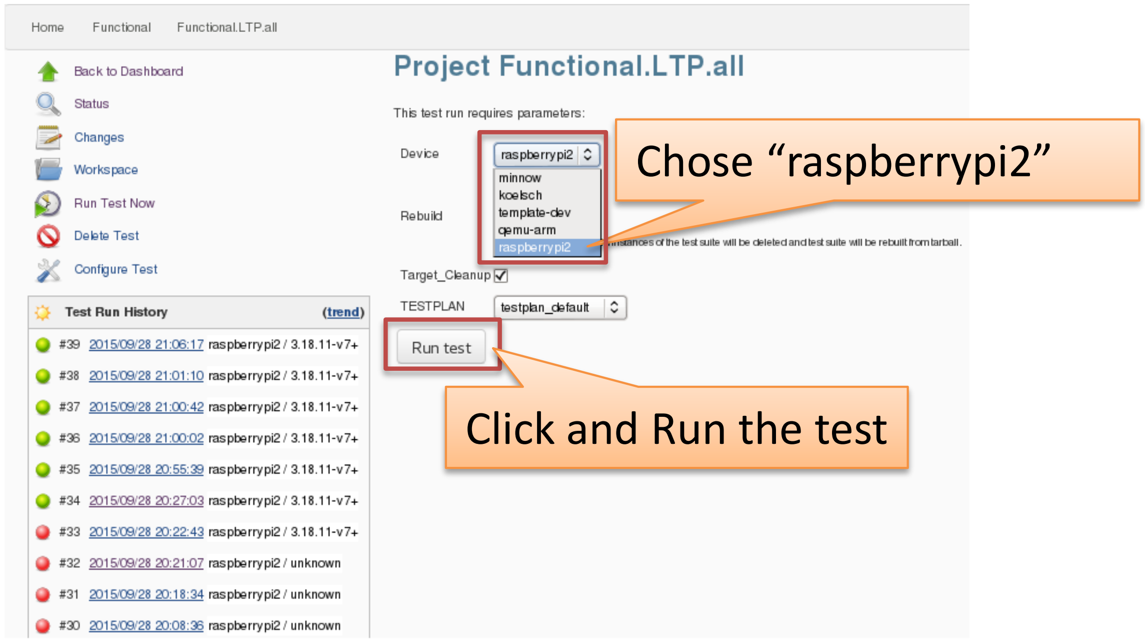
	#	Date	Target
🟢	#39	2015/09/28 21:06:17	raspberrypi2 / 3.18.11-v7+
🟢	#38	2015/09/28 21:01:10	raspberrypi2 / 3.18.11-v7+
🟢	#37	2015/09/28 21:00:42	raspberrypi2 / 3.18.11-v7+
🟢	#36	2015/09/28 21:00:02	raspberrypi2 / 3.18.11-v7+
🟢	#35	2015/09/28 20:55:39	raspberrypi2 / 3.18.11-v7+
🟢	#34	2015/09/28 20:27:03	raspberrypi2 / 3.18.11-v7+
🔴	#33	2015/09/28 20:22:43	raspberrypi2 / 3.18.11-v7+
🔴	#32	2015/09/28 20:21:07	raspberrypi2 / unknown
🔴	#31	2015/09/28 20:18:34	raspberrypi2 / unknown
🔴	#30	2015/09/28 20:08:36	raspberrypi2 / unknown

Permalinks

- Last test run (#39), 23 hr ago
- Last stable test run (#39), 23 hr ago
- Last successful test run (#39), 23 hr ago
- Last failed test run (#33), 1 day 0 hr ago
- Last unsuccessful test run (#33), 1 day 0 hr ago

Run new LTP on Raspberry Pi2 (Cont'd)

- To Run LTP-20150420



Project Functional.LTP.all

This test run requires parameters:

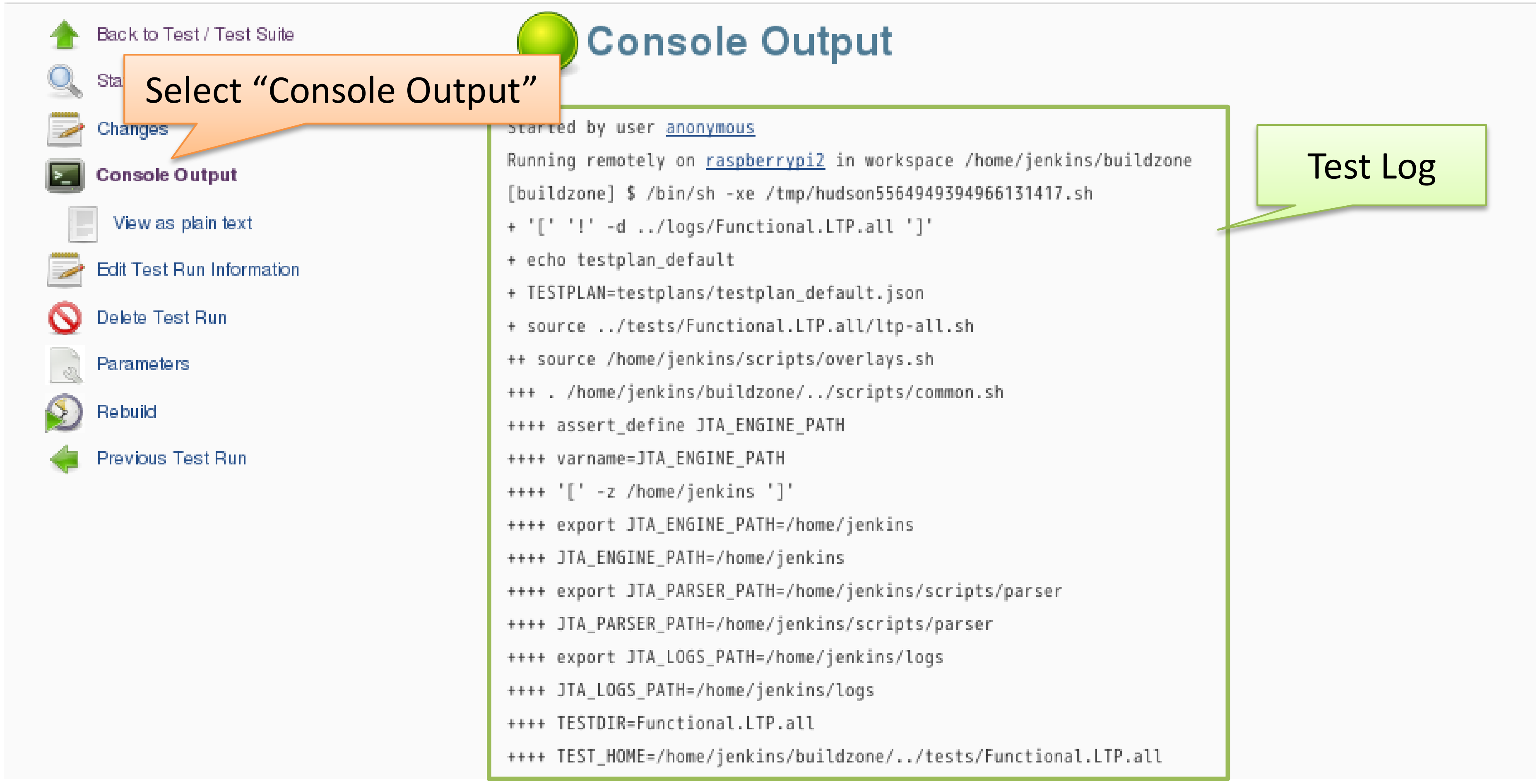
Device	raspberrypi2 (minnow, koelsch, template-dev, qemu-arm, raspberrypi2)
Rebuild	…instances of the test suite will be deleted and test suite will be rebuilt from tarball.
Target_Cleanup	✓
TESTPLAN	testplan_default

Test Run History (trend)

🟢	#39	2015/09/28 21:06:17	raspberrypi2 / 3.18.11-v7+
🟢	#38	2015/09/28 21:01:10	raspberrypi2 / 3.18.11-v7+
🟢	#37	2015/09/28 21:00:42	raspberrypi2 / 3.18.11-v7+
🟢	#36	2015/09/28 21:00:02	raspberrypi2 / 3.18.11-v7+
🟢	#35	2015/09/28 20:55:39	raspberrypi2 / 3.18.11-v7+
🟢	#34	2015/09/28 20:27:03	raspberrypi2 / 3.18.11-v7+
🔴	#33	2015/09/28 20:22:43	raspberrypi2 / 3.18.11-v7+
🔴	#32	2015/09/28 20:21:07	raspberrypi2 / unknown
🔴	#31	2015/09/28 20:18:34	raspberrypi2 / unknown
🔴	#30	2015/09/28 20:08:36	raspberrypi2 / unknown

Run new LTP on Raspberry Pi2 (Cont'd)

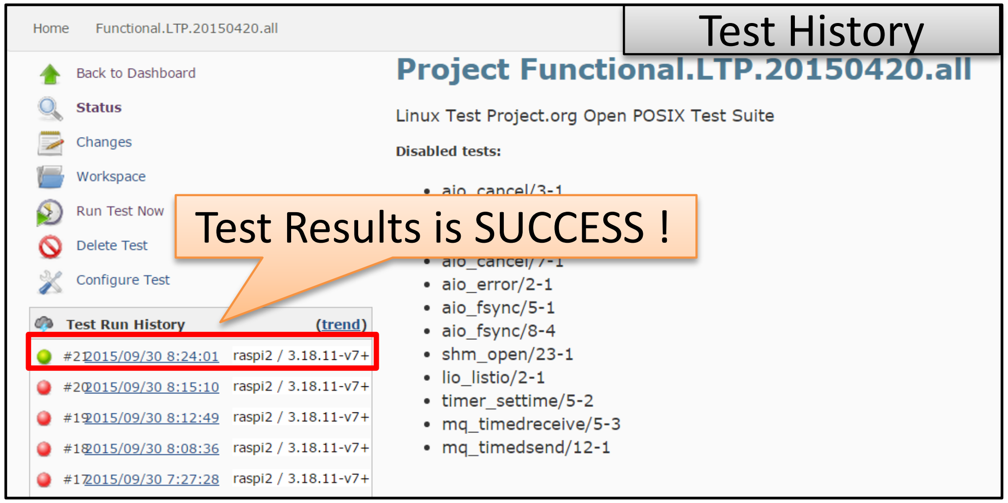
- We can Show the log with Console output at run time



```
Started by user anonymous
Running remotely on raspberrypi2 in workspace /home/jenkins/buildzone
[buildzone] $ /bin/sh -xe /tmp/hudson5564949394966131417.sh
+ '[' '!' -d ../logs/Functional.LTP.all ']'
+ echo testplan_default
+ TESTPLAN=testplans/testplan_default.json
+ source ../tests/Functional.LTP.all/ltp-all.sh
++ source /home/jenkins/scripts/overlays.sh
+++ . /home/jenkins/buildzone/../scripts/common.sh
++++ assert_define JTA_ENGINE_PATH
++++ varname=JTA_ENGINE_PATH
++++ '[' -z /home/jenkins ']'
++++ export JTA_ENGINE_PATH=/home/jenkins
++++ JTA_ENGINE_PATH=/home/jenkins
++++ export JTA_PARSER_PATH=/home/jenkins/scripts/parser
++++ JTA_PARSER_PATH=/home/jenkins/scripts/parser
++++ export JTA_LOGS_PATH=/home/jenkins/logs
++++ JTA_LOGS_PATH=/home/jenkins/logs
++++ TESTDIR=Functional.LTP.all
++++ TEST_HOME=/home/jenkins/buildzone/../tests/Functional.LTP.all
```

Run new LTP on Raspberry Pi2 (Cont'd)

- To Show Test Results



Home   Functional.LTP.20150420.all

- Back to Dashboard
- Status
- Changes
- Workspace
- Run Test Now
- Delete Test
- Configure Test

Project Functional.LTP.20150420.all

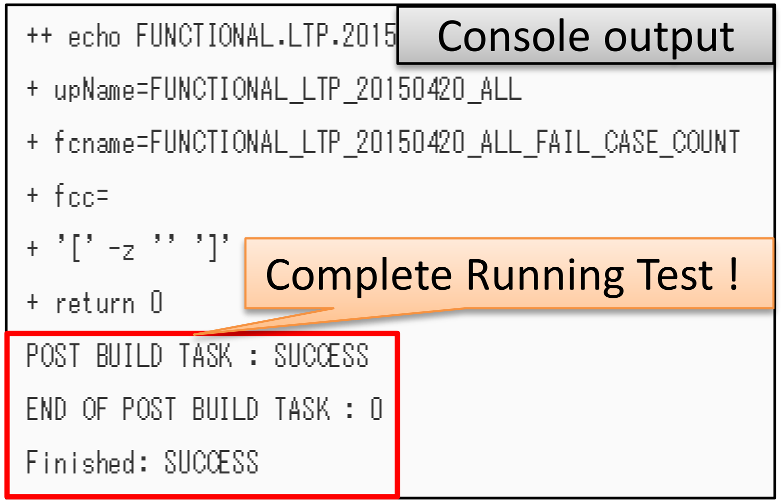
Linux Test Project.org Open POSIX Test Suite

Disabled tests:

- aio_cancel/3-1
- aio_cancel/7-1
- aio_error/2-1
- aio_fsync/5-1
- aio_fsync/8-4
- shm_open/23-1
- lio_listio/2-1
- timer_settime/5-2
- mq_timedreceive/5-3
- mq_timedsend/12-1

Test Run History (trend)

● #21	2015/09/30 8:24:01	raspi2 / 3.18.11-v7+
● #20	2015/09/30 8:15:10	raspi2 / 3.18.11-v7+
● #19	2015/09/30 8:12:49	raspi2 / 3.18.11-v7+
● #18	2015/09/30 8:08:36	raspi2 / 3.18.11-v7+
● #17	2015/09/30 7:27:28	raspi2 / 3.18.11-v7+



```
++ echo FUNCTIONAL.LTP.2015
+ upName=FUNCTIONAL_LTP_20150420_ALL
+ fcname=FUNCTIONAL_LTP_20150420_ALL_FAIL_CASE_COUNT
+ fcc=
+ '[' -z '' ']'
+ return 0
POST BUILD TASK : SUCCESS
END OF POST BUILD TASK : 0
Finished: SUCCESS
```

Case	Number
TPASS	4071
TINFO	2776
TCONF	140
TFAIL	4
TBROK	2764

- TPASS - Indicates that the test case had the expected result and passed
- TINFO - Specifies useful information about the status of the test that does not affect the result and does not indicate a problem.
- TCONF - Indicates that the test case was not written to run on the current hardware or software configuration such as machine type, or, kernel version.
- TFAIL - Indicates that the test case had an unexpected result and failed.
- TBROK - Indicates that the remaining test cases are broken and will not execute correctly, because some precondition not met, such as a resource not being available.

Summary & Future Works

- Summary
  - LTSI Test Framework has already had many kinds of target boards and Test suites.
  - We showed How to Customize.
    - Add a new target board as Raspberry pi 2
    - Add a new test suite as LTP-20150420
  - We showed the result of running LTP on Raspberry pi2

- Future Works
  - We try to add Kselftest
    - Kselftest is a quick method of running tests for the Linux kernel.
  - We think about making a SDK without yocto
    - We would like to use LTSI Test Framework for some product without yocto.
  - We think about how to analysis and judge test results.

Reference

- LTSI project :
  - http://ltsi.linuxfoundation.org/
- LTSI Test project:
  - http://ltsi.linuxfoundation.org/ltsi-test-project
  - Test Framework:
    - https://bitbucket.org/cogentembedded/jta-public.git