Yocto 1.1.1 Fullpass Test

Test Report

Project: yocto

Author: admin

Printed by TestLink on 22/12/2011

1 Test Suite : Yocto 1.1.1 Fullpass Test

1.1 Test Suite : hob

Test Case TC-1548: hob launch without error	
Summary: hob could be launched without error	
Steps: 1. Prepare poky build environment 2. launch hob with command "hob" 3. Check if hob is launched correctly and no error message in console	
Expected Results: hob launched correctly and no error message	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1549: base image selection	
Summary: package list should be loaded for "image contents" for each selection in "base image" field	
Steps: 1. launch hob 2. select one "Machine", for example, qemumips 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents"	
Expected Results: package list should be loaded for "image contents" for each selection in "base image" field	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1550: package list re-load for "base image" change	
Summary: package list should be re-loaded if changing image type for "base image"	
Steps: 1. launch hob 2. select one "Machine", for example, qemumips 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" 5. change the image type for "Base image", for example, "core-image-minimal", the list of packages should be re-loaded	
Expected Results: package list should be re-loaded if changing image type for "base image"	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1551: package list re-load for "Machine" change	
Summary: package list for "image contents" should be re-loaded and correct when "Machine" changing	
Steps: 1. launch hob 2. select one "Machine", for example, qemuppc 3. select one image for "Base image", for example, "core-image-sato" 4. a list of packages should be loaded for "image contents" 5. select another machine type for "Machine", for example, beagleboard 6. a new list of packages should be re-loaded for "image contents" and should not same as the outputs in step 4	
Expected Results: package list for "image contents" should be re-loaded and correct when "Machine" changing	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1552: package list re-load correct for "Machine" change	
Summary: package list re-load correct for "Machine" change	
Steps: 1. launch hob 2. check the default value of "Machine", for example, qemux86, then choose a value for "base image", for example, "core-image-sato", write down the package number for the image 3. choose another value for "Machine", for example, beagleboard and choose the same value for "base image" as for qemux86, the pakcage number for beagleboard should not same as qemux86	
Expected Results: Different machine/image should have different package list	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1553: package list reset	
Summary: reset button should clear package list for "image contents"	
Steps: 1. launch hob 2. select one "Machine", for example, qemumips 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" 5. click "reset" button, all packages should be cleared for "image contents"	
Expected Results: reset button should clear package list for "image contents"	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1554: customized package list save as bb file(add packages)
Summary:

user could use "save" or "save as" button to save customized bb file
Steps: 1. launch hob 2. select one "Machine", for example, qemumips 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" 5. select some un-selected package, for example, acpid 6. click "File"->"Save" or "Save As", it should save the user customized package list into a bb file 7. click "reset" button, and click "File"->"Open", choose the saved bb file 8. The user customized package list should be shown
Expected Results: user could use "save" or "save as" button to save customized bb file

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1555: cancel customized package list save action
Summary: cancel customized package list save action should not cause any error
Steps: 1. launch hob 2. select one "Machine", for example, qemux86-64 3. select one image for "Base image", for example, "core-image-minimal" 4. a list of packages should be loaded for "image contents" 5. select some un-selected package, for example, acpid 6. click "x" button, a dialog should pop up and ask user if customiszations wants be saved. 7. click "yes" and click "cancel" in next page 8. hob should exit without error log
Expected Results: No error log with hob exit when cancel customized package list save action

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1556: No native package shown in package list	
Summary: There should be no native package shown in package list	
Steps: 1. launch hob 2. check if there is any -native package in "Packages"	
Expected Results: There should be no native package shown in package list	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1557: stop build during image/package building	
Summary: "stop build" button should be able to stop/force stop building	
Steps: 1. launch hob 2. select one "Machine", for example, qemuarm 3. select one image for "Base image", for example, "core-image-sato" 4. a list of packages should be loaded for "image contents" 5. select some un-selected package, for example, acpid 6. click "bake" button to start build 7. in building page, click "stop build", and click "stop" or "force stop" to stop building	
Expected Results: "stop build" button should be able to stop/force stop building	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1558: search package name in package list
Summary:

User could search package name from "Search packages"	
Steps: 1. launch hob 2. search some package via "search packages", for example, avahi 3. the searched package should be shown in "packages"	
Expected Results: User could search package name from "Search packages"	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1559: task list re-load when base image change	
Summary: task list for "package collections" should be re-loaded when base image changing	
Steps: 1. launch hob 2. select one "Machine", for example, qemuppc 3. select one image for "Base image", for example, "core-image-sato" 4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections" 5. select another image type for "base image", for example, core-image-basic 6. a new list of tasks should be re-loaded	
Expected Results: task list for "package collections" should be re-loaded when base image changing	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1560: user could customize threads of bitbake and make
Summary: user could customize threads of bitbake and make in hob

Steps: 1. launch hob 2. select one "Machine", for example, qemux86 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections" 5. click Edit->Preferences, and customize number for "bitbake threads" and "make threads", for example, you could set both 1 for them 6. click "bake" and check 'ps' command output if there is one thread running	
Expected Results: user could customize threads of bitbake and make in hob	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1561: add layer for new target build	
Summary: user could add layer for new target build	
Steps: 1. launch hob 2. click File->Add Layer, then choose one layer, for example, you could download meta-intel.git and use sugarbay 3. check "Machine" list and sugarbay should be available 4. choose one type, for example, core-image-sato-sdk 5. click "bake" and check the build result	
Expected Results: user could add layer for new target build	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1562: another build after stop build
Summary:

user could start another build after stop a build
Steps: 1. launch hob 2. select one "Machine", for example, qemuarm 3. select one image for "Base image", for example, "core-image-sato" 4. a list of packages should be loaded for "image contents" 5. select some un-selected package, for example, acpid 6. click "bake" button to start build 7. in building page, click "stop build", and click "stop" to stop building 8. back to the main UI, and select another image, then click "bake" button 9. wait for the build finished and it should be no error met
Expected Results: user could start another build after stop a build

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1563: back to main UI after back stopped
Summary: click "back" button should bake to main UI after bake stopped
Steps: 1. launch hob 2. select one "Machine", for example, qemuarm 3. select one image for "Base image", for example, "core-image-sato" 4. a list of packages should be loaded for "image contents" 5. select some un-selected package, for example, acpid 6. click "bake" button to start build 7. in building page, click "stop" or "force stop" 8. click "back" button, it should return to main UI
Expected Results: click "back" button should bake to main UI after bake stopped

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1564: customized preference items save in hob.conf	
Summary: user customized items should be saved in local.conf or hob.local.conf	
Steps: 1. launch hob 2. select one "Machine", for example, qemumips 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" 5. select some un-selected package, for example, acpid 6. click "Edit"->"Preferences", change the value of all items in this page, for example, changing "poky" to "poky bleeding" for "distribution", selecting "GPLv3", "rpm" for "package format", "3", "4" for "bitbake threads" and "Make threads" and enable toolchain build, setting "x86_64" for "Toolchain host" 6. exit hob 7. check hob*.conf, above modifications should be set in it 8. re-launch hob and check "Preferences", all above modifications should be set in this page	
Expected Results: user customized items should be saved in hob*.conf	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1565: bake a image without error (base image)	
Summary: user could use hob to build a image without error	
Steps: 1. launch hob 2. select one "Machine", for example, qemumips 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" 5. click "Bake" and wait for a successful build finished	
Expected Results: user could use hob to build a image without error	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1566: bake a image without error (added package)	
Summary: user could use hob to build a image without error	
Steps: 1. launch hob 2. select one "Machine", for example, qemumips 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" 5. select some un-selected package, for example, acpid 6. click "Bake" and wait for a successful build finished 7. after build finished, check if the added package built into image	
Expected Results: user could use hob to build a image without error	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1567: back to main UI after bake finished	
Summary: click "back" button should bake to main UI after bake finished	
Steps: 1. launch hob 2. select one "Machine", for example, qemuarm 3. select one image for "Base image", for example, "core-image-sato" 4. a list of packages should be loaded for "image contents" 5. select some un-selected package, for example, acpid 6. click "bake" button to start build 7. in bake page, wait for build finished 8. click "back" button, it should return to main UI	
Expected Results: click "back" button should bake to main UI after bake finished	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1568: toolchain built correct with user customization	
Summary: toolchain generated correct with user selection	
Steps: 1. launch hob 2. select one "Machine", for example, beagleboard 3. select one image for "Base image", for example, "core-image-sato" 4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections" 5. click Edit->Preferences, and select "Build external development toolchain with image", for "toolchain host", you could pick one and choose one arch for "toolchain host", for example, x86_64 6. click "bake" button and it should generate toolchain as well as selected packages/images 7. check the generated toolchain tarball, the name should be consistent with the above selection, for example, x86_64 for host name, arm for beagleboard 8. use the toolchain to build a C program and make sure it workable in target	
Expected Results: toolchain generated correct with user selection	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1569: non-GPLv3 build	
Summary: non-GPLv3 build should be supported for hob	
Steps: 1. launch hob 2. select one "Machine", for example, qemux86 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections" 5. click Edit->Preferences, and select "Exclude GPLv3 packages" 6. click "bake" to build a non-GPLv3 image 7. After build is finished, check if there is any GPLv3 packages built in	
Expected Results: non-GPLv3 build should be supported for hob	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	

image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1570: distribution selection for image/package build	
Summary: user could select different distribution for "distribution"	
Steps: 1. launch hob 2. select one "Machine", for example, qemux86 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections" 5. click Edit->Preferences, and select different distribution for "distribution", for example, poky-lsb 6. click "bake" button and it should generate packages or image with selected distribution	
Expected Results: user could select different distribution for "distribution"	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1571: ipk package build for image/package build	
Summary: build image with ipk package format	
Steps: 1. launch hob 2. select one "Machine", for example, qemux86 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections" 5. click Edit->Preferences, and select ipk for "package format" 6. click "bake" button and it should generate images with ipk format	
Expected Results: build image with ipk package format	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob

target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1572: deb package build for image/package build	
Summary: build image with deb package format	
Steps: 1. launch hob 2. select one "Machine", for example, qemux86 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections" 5. click Edit->Preferences, and select deb for "package format" 6. click "bake" button and it should generate images with dformat	
Expected Results: build image with deb package format	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1573: rpm package build for image/package build	
Summary: build image with rpm package format	
Steps: 1. launch hob 2. select one "Machine", for example, qemux86 3. select one image for "Base image", for example, "core-image-basic" 4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections" 5. click Edit->Preferences, and select rpm for "package format" 6. click "bake" button and it should generate images with rpm format	
Expected Results: build image with rpm package format	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready

Feature:	hob
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

1.2 Test Suite : System & Core OS

Test Case TC-1574: zypper command installed and workable	
Summary: check if zypper is installed and can work	
Steps: 1. Run command "zypper", and check the output	
Expected Results: Command "zypper" print the list of available global options and commands	
Test Execution Cycle Type:	Sanity
Case Automation Type:	Auto
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1575: zypper help search	
Summary: check help option with zypper command	
Steps: 1. Run "zypper help search" and check the output	
Expected Results: The command should print help for the search command	
Test Execution Cycle Type:	Sanity
Case Automation Type:	Auto
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow,

	blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1576: zypper search package	
Summary: search package with zypper	
Steps: 1. Run "zypper search package_name" and check the output, for example "zypper search avahi"	
Expected Results: The command should search package "avahi" is installed or not	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Auto
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1577: zypper remove package	
Summary: remove package with zypper	
Steps: 1. Run "zypper rm pakcage_name" and check the output, for example "zypper rm avahi"	
Expected Results: The command should remove package "avahi"	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1578: zypper install package	
Summary: install package with zypper	
Steps: 1. Set up a yum based repository on local server 2. Build out a package, which does not need any run-time dependency package, with local poky tree. For example, package "man" 3. In target system, run "zypper addrepo http://ip_address_of_repository zypper_test_repo" 4. Run "zypper refresh" to refresh the zypper repository cache 5. Run "zypper install package_name" and check the output, for example "zypper install man" to install package, which has no run-time dependency	
Expected Results: The command should install package "man"	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1579: zypper install dependency package	
Summary: install dependency package with zypper	
Steps: 1. Set up a yum based repository on local server 2. Build out a package, which does not need any run-time dependency package, with local poky tree. For example, package "mc" 3. In target system, run "zypper addrepo http://ip_address_of_repository zypper_test_repo" 4. Run "zypper refresh" to refresh the zypper repository cache 5. Run "zypper install package_name" and check the output, for example "zypper install mc" to install package, which needs run-time dependency packages installed also, like ncurses-terminfo.	
Expected Results: The command should install package "mc" and denpendency package ncurses-terminfo.	
Test Execution Cycle Type:	Weekly
Case Automation	Manual

Type:	
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1580: zypper install .all packages	
Summary: install packages from all folder with zypper	
Steps: 1. Set up a yum based repository on local server 2. Build out a package, which belongs to all folder, for example, xcursor-transparent-theme-dbg-0.1.1-r3.all.rpm. 3. In target system, run "zypper addrepo http://ip_address_of_repository zypper_test_repo" 4. Run "zypper refresh" to refresh the zypper repository cache 5. Run "zypper install xcursor-transparent-theme-dbg" and check the output	
Expected Results: package install from all folder should be installed successfully with zypper	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1581: rpm query package	
Summary: make sure rootfs image is built with rpm packages	
Steps: 1. launch terminal 2. run command "rpm -qa", which lists all existing packages in system	
Expected Results: "rpm -qa" should print all existing packages in system	
Test Execution Cycle Type:	Sanity
Case Automation Type:	Manual

Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1582: rpm install package	
Summary: rpm format package can be installed	
Steps: 1. Get a RPM package(for example, avahi or powertop) from zypper repository or build one on local machine 2. Copy the package into image, run command "rpm -ivh package_name" to install the package	
Expected Results: RPM format package can be installed	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1583: rpm install dependency package	
Summary: rpm command should report dependency when installing package	
Steps: 1. Get a RPM package or build one on local machine, which should have run-time dependency. For example, mc RPM should depends on ncurses-terminfo 2. Run "rpm -ivh package_name" and check the output, for example "rpm -ivh mc.rpm*" should report the dependency on ncurses-terminfo	
Expected Results: rpm command should report message when some RPM installation depends on other packages	
Test Execution Cycle Type:	Weekly
Case Automation	Manual

Type:	
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1584: rpm remove package	
Summary: rpm command can remove package in system	
Steps: 1. Launch terminal and run command "rpm -e package_name" to remove some package, for example, avahi	
Expected Results: RPM package can be removed by command rpm	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1585: check rpm install/removal log file size	
Summary: The case is to track log file size after rpm install/removal	
Steps: 1. After system is up, check the log file size after rpm/zypper install/removal 2. for rpm, there will be some database files under /var/lib/rpm/, named as "__db.xxx" and there will be some log files under /var/lib/rpm/log, named as "log.xxxxxx". Each file will occupy about 10MB. 3. after several rpm/zypper install/removal, rpm will create several log files under /var/lib/rpm/log, which eat lots of system disk space.	
Expected Results: there should be some method to keep rpm log in a small size	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual

Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1590: g++ compile in sdk image

Summary:

check if g++ can compile program in sdk image

Steps:

1. Boot up sdk image
2. check if g++ is built in
3. compile following program test.c "g++ test.c -o test -lm"
4. run "test" and check the output

test.c:

```
##########
#include <stdio.h>
#include <math.h>

double
convert(long long l)
{
   return (double)l;    // or double(l)
}

int
main(int argc, char * argv[])
{
   long long l = 10;
   double f;

   f = convert(l);
   printf("convert: %lld => %f\n", l, f);

   f = 1234.67;
   printf("floorf(%f) = %f\n", f, floorf(f));
   return 0;
}
##########
```

Expected Results:

executable binary test can run without problem

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	sdk
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato-sdk, lsb-sdk
Last Result	**Not Run**

Keywords:	None

Test Case TC-1591: gcc compile in sdk image

Summary:

check if gcc can compile program in sdk image

Steps:

1. Boot up sdk image
2. check if gcc is built in
3. compile following program test.c "gcc test.c -o test -lm"
4. run "test" and check the output

test.c:
```
##########
#include <stdio.h>
#include <math.h>

double
convert(long long l)
{
   return (double)l;    // or double(l)
}

int
main(int argc, char * argv[])
{
   long long l = 10;
   double f;

   f = convert(l);
   printf("convert: %lld => %f\n", l, f);

   f = 1234.67;
   printf("floorf(%f) = %f\n", f, floorf(f));
   return 0;
}
##########
```

Expected Results:

executable binary test can run without problem

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	sdk
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1592: run command make in sdk image

Summary:

check if command make can work in sdk image	
Steps: 1. Boot up sdk image 2. check if make is built in 3. run command "make" with following makefile and build the test.c file from case "gcc compile in sdk image" test: test.o gcc -o test test.o -lm test.o: test.c gcc -c test.c	
Expected Results: make command can work without problem	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	sdk
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1596: perl program work in image	
Summary: A perl program could be executed and output correctly in image	
Steps: 1. Check if perl is installed in image and could run with "perl -v" 2. Prepare a perl program like followig test.pl 3. Run "perl test.pl" ######## $a = 9.01e+21 + 0.01 - 9.01e+21; print ("the value of a is ", $a, "\n"); $a = 9.01e+21 - 9.01e+21 + 0.01; print ("the value of a is ", $a, "\n"); ########	
Expected Results: The test.pl could run without problem	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Auto
Case State:	Ready
Feature:	system usage

target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1597: shutdown system	
Summary: verify that system can be shutdown by command	
Steps: 1. boot system 2. launch terminal and run "shutdown -h now" or "poweroff"	
Expected Results: System can be shutdown successfully	
Test Execution Cycle Type:	Sanity
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1598: reboot system	
Summary: verify that system can boot by command	
Steps: 1. boot system 2. launch terminal and run "reboot"	
Expected Results: System can reboot successfully	
Test Execution Cycle Type:	Sanity
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**

Keywords:	None

Test Case TC-1599: adjust date and time	
Summary: adjust date and time	

Steps:
 	
1.launch terminal and run "date -R" to check current system time
2.adjust Date&Time by these commands:
For date command from coreutils, for example the sdk image use coreutils, you should use following syntax:
$ date -s "10:00:00 20100809"
$ date -R
$ Mon, 09 Aug 2010 10:00:00 +0000
For date command in busybox, for example the sato image use busybox, you should use following syntax:
$ date "080910002010"
$ date -R
$ Mon, 09 Aug 2010 10:00:00 +0000
3. check date with "date -R" and the time shown on matchbox-panel	

Expected Results: System time should be adjust to what you specified	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Auto
Case State:	Ready
Feature:	system usage
target:	e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1600: switch among multi applications and desktop	
Summary: switch among multi applications and desktop	
Steps: 1. launch several applications(like contacts, file manager) 2. launch terminal 3. switch among multi applications and desktop 4. close applications Note: The case is for sato image only.	
Expected Results: 1. user could switch among multi applications and desktop	
Test Execution Cycle Type:	Fullpass
Case Automation	Manual

Type:	
Case State:	Ready
Feature:	system usage
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1601: vncserver for target	
Summary: Check if vncserver setup work in target and vnc client could connect it	
Steps: 1. Check if x11vnc is installed in target 2. Run command "x11vnc -display :0.0", check the ip address of the target 3. On a client, run command "vncviewer $ip_address_of_target:0"	
Expected Results: A virtual X desktop of target should be pop-up on the client	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemumips, e-menlow, blacksand, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1602: file manager	
Summary: file manager	
Steps: 1.launch file manager from application panel 2.view folder/file in file manager 3.copy and paste folder/file in file manager Note: The test is only for sato image	
Expected Results: 1.folder and file could be listed in file browser with different display mode	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready

Feature:	system usage
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1603: system dmesg log check	
Summary: check if there is error in dmesg after system boot up	
Steps: 1. boot system and run command "dmesg"	
Expected Results: No error message in dmesg	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1604: usb mount	
Summary: verify that system can mount plugged usb automatically	
Steps: 1. boot system 2. plug usb stick	
Expected Results: 1. system notify that usb stick is accessible	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1605: usb read files	
Summary: verify that system can read files from usb	
Steps: 1. boot system 2. plug usb stick 3. view files in usb by file browser 4.copy some files from usb to local hardware	
Expected Results: 1. view/copy successfully	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1606: usb umount	
Summary: verify that system can unmout usb automically	
Steps: 1. boot system 2. plug usb stick 3. view files in usb by file browser 4.unplug usb	
Expected Results: 1. usb direcoty in file browser automatically missed	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1607: usb write files	
Summary: verify that system can write files to usb	
Steps: 1. boot system 2. plug usb stick 3. create files in usb 4.copy some files from local hardware to usb	
Expected Results: 1. create/copy successfully	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1608: file copy by scp	
Summary: check if file can be copied from remote machine to device by scp	
Steps: 1. check avahi is install and started 2. get system IP and try "scp file $IP:/home/root" from remote machine (file >= 500M for real HW, file>=5M for QEMU)	
Expected Results: File can be copied from remote machine to device by scp	
Test Execution Cycle Type:	Sanity
Case Automation Type:	Auto
Case State:	Ready
Feature:	connectivity
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1609: connman launch after boot
Summary:

After system booted, the connmand daemon should be launched	
<u>Steps:</u> 1. boot system 2. "ps |grep connmand" 3. check if there is a thread named connmand in background	
<u>Expected Results:</u> There should be one thread named connmand in background	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	connectivity
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk
<u>Last Result</u>	**Not Run**
<u>Keywords:</u>	None

Test Case TC-1610: ethernet enabled in connman	
<u>Summary:</u> After system boot, ethernet can get IP address with connman	
<u>Steps:</u> 1. boot system with network cable plugged in 2. "ps |grep connmand" if connmand is started 3. "ifconfig" check ethernet could get IP address and ping the address from remote machine	
<u>Expected Results:</u> Ethernet interface can get IP via connman	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	connectivity
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk
<u>Last Result</u>	**Not Run**
<u>Keywords:</u>	None

Test Case TC-1611: only one connmand in background
<u>Summary:</u> there should be no more than one connmand in background
<u>Steps:</u>

1. boot system 2. "ps |grep connmand" 3. the connmand should be in background 4. run command "connmand" 5. check if the second connmand can be generated	
Expected Results: There will be only one connmand instance in background	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	connectivity
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1612: remote access by ssh	
Summary: check if the device can be accessed remotely by ssh	
Steps: 1. check avahi is install and started 2. get system IP and try "ssh $IP" from remote machine	
Expected Results: it is ok to access system by ssh from remote machine	
Test Execution Cycle Type:	Sanity
Case Automation Type:	Auto
Case State:	Ready
Feature:	connectivity
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1615: connman offline mode in connman-gnome
Summary: change offline mode in comman-gnome can make all connection off
Steps: 1. Launch connman-properties after system booting

2. choose "offline mode" and check the connection of all network interfaces	
Expected Results: All connection should be off after clicking "offline mode"	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	connectivity
target:	qemux86_32, qemux86_64, qemuarm, qemumips, e-menlow, blacksand, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1616: X server can start up with runlevel 5 boot	
Summary: check if X server can work well after system runlevel 5 booting	
Steps: 1. boot up system with default runlevel	
Expected Results: X server can start up well and desktop display has no problem	
Test Execution Cycle Type:	Sanity
Case Automation Type:	Auto
Case State:	Ready
Feature:	graphics
target:	qemux86_32, qemux86_64, qemuarm, qemumips, e-menlow, blacksand, beagleboard, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1617: qt application quicky	
Summary: quicky is a simple note-taking application with Wiki-style syntax and behaviour	
Steps: launch quicky and write something in quicky	
Expected Results: http://qt-apps.org/content/show.php/Quicky?content=80325	
Test Execution Cycle Type:	Weekly
Case Automation	Manual

Type:	
Case State:	Ready
Feature:	graphics
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay, jasperforest
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1622: disk space check	
Summary: There should be enough disk space for QEMU rootfs	
Steps: 1. Launch QEMU targets(with rootfs.ext3 file) 2. Check the output of command df 3. If there is less than 5M disk space available, we assume it a failure	
Expected Results: There should be enough disk space for QEMU targets	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1623: click terminal icon on X desktop	
Summary: terminal icon should work without problem on X desktop	
Steps: 1. After system launch and X start up, click terminal icon on desktop 2. Check if only one terminal window launched and no other problem met	
Expected Results: there should be no problem after launching terminal	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemumips, e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato, sato-sdk

Last Result	**Not Run**
Keywords:	None

Test Case TC-1625: system shutdown with UNFS	
Summary: system shutdown with UNFS should work	
Steps: 1. Use UNFS to start QEMU targets 2. Run shutdown in QEMU targets	
Expected Results: QEMU shutdown with UNFS should work	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	sdk
target:	qemux86_32, qemux86_64, qemuarm, qemumips
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1626: no connman-gnome icon on desktop	
Summary: there should be no connman-gnome icon on desktop	
Steps: 1. Launch sato image 2. There should be no connman-gnome icon on desktop, and connman-properties should be only invoked by toolbar	
Expected Results: There should be no connman-gnome icon on desktop, and connman-properties should be only invoked by toolbar	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemumips, e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1627: application contacts should work	
Summary: application contacts should work without problem	
Steps: 1. Make sure X is started up 2. Check if there is "contacts" icon on desktop and run it 3. Check if there is any error by checking the output of this action and dmesg log	
Expected Results: "contacts" launch should not cause any error	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemumips, e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1628: x11vnc icon click for target	
Summary: Check if vncserver could work in target by clicking x11vnc icon	
Steps: 1. Check if there is a x11vnc icon in target 2. Click the x11vnc icon and check the ip address of the target 3. On a client, run command "vncviewer $ip_address_of_target:0"	
Expected Results: A virtual X desktop of target should be pop-up on the client	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemumips, e-menlow, blacksand, crownbay, sugarbay
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1629: RTLDLIST path check for ldd command
Summary: check if the file set in RTLDLIST is valid

Steps: 1. After system is up, check if the RTLDLIST variable in ldd command 2. The file path set in RTLDLIST should be valid	
Expected Results: check if the file set in RTLDLIST is valid	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1630: check bash in image	
Summary: check if bash exists in image	
Steps: 1. After system is up, check if bash command exists	
Expected Results: bash command should exist in image	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips, e-menlow, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, crownbay, sugarbay, jasperforest
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1631: "Install/Remove Software" icon should work
Summary: "Install/Remove Software" icon should work
Steps: 1. After system is up, check if "Install/Remove Software" icon could work
Expected Results:

"Install/Remove Software" icon should work	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	system usage
target:	qemux86_32, qemux86_64, qemuarm, qemumips
image profile:	sato, sato-sdk
Last Result	**Not Run**
Keywords:	None

1.3 Test Suite : ADT

Test Case TC-1632: gcc from ADT toolchain can build c program

Summary:

gcc from ADT toolchain can build c program and run with qemu-${ARCH} command or in target image

Steps:

1. Install toolchain tarball and setup cross compile environment
2. compile following program test.c "${CC} test.c -o test -cc -lm"
3. run "test" with qemu-${ARCH} or run it into corresponding target image and check the output

Note: Currently, only i586_i586, x86-64_x86-64 and i586_$X(x is mips, arm and ppc) toolchain tarballs are covered in testing.

```
#########
#include <stdio.h>
#include <math.h>

double
convert(long long l)
{
   return (double)l;    // or double(l)
}

int
main(int argc, char * argv[])
{
   long long l = 10;
   double f;

   f = convert(l);
   printf("convert: %lld => %f\n", l, f);

   f = 1234.67;
   printf("floorf(%f) = %f\n", f, floorf(f));
   return 0;
}
#########
```

Expected Results: executable binary test can run without problem	
Test Execution Cycle Type:	Sanity
Case Automation Type:	Auto
Case State:	Ready
Feature:	sdk
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1633: g++ from ADT toolchain can build c program

Summary:

g++ from ADT toolchain can build c program and run with qemu-${ARCH} command or in target image

Steps:

1. Install toolchain tarball and setup cross compile environment
2. compile following program test.c "${CXX} test.c -o test -cc++ -lm"
3. run "test" with qemu-${ARCH} or run it in corresponding target image and check the output

Note: Currently, only i586_i586, x86-64_x86-64 and i586_$X(x is mips, arm and ppc) toolchain tarballs are covered in testing.

```
#########
#include <stdio.h>
#include <math.h>

double
convert(long long l)
{
   return (double)l;    // or double(l)
}

int
main(int argc, char * argv[])
{
   long long l = 10;
   double f;

   f = convert(l);
   printf("convert: %lld => %f\n", l, f);

   f = 1234.67;
   printf("floorf(%f) = %f\n", f, floorf(f));
   return 0;
}
#########
```

Expected Results:

executable binary test can run without problem

Test Execution Cycle Type:	Sanity

Case Automation Type:	Auto
Case State:	Ready
Feature:	sdk
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1634: ADT toolchain could build cvs project	
Summary: ADT toolchain could build cvs project	
Steps: 1. Install toolchain tarball and setup cross compile environment 2. Download cvs project, http://ftp.gnu.org/non-gnu/cvs/source/feature/1.12.13/cvs-1.12.13.tar.bz2 3. With the cross compile environment, run "./configure ${CONFIGURE_FLAGS}", "make", "make install DESTDIR=/opt/tmp" Note: Currently, only i586_i586, x86-64_x86-64 and i586_$X(x is mips, arm and ppc) toolchain tarballs are covered in testing.	
Expected Results: cvs project could be compiled successfully with ADT toolchain	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	sdk
target:	build_system
image profile:	lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1635: ADT toolchain could build iptables project
Summary: iptables project could be compiled with ADT toolchain
Steps: 1. Install toolchain tarball and setup cross compile environment 2. Download iptables project, http://netfilter.org/projects/iptables/files/iptables-1.4.11.tar.bz2 3. With the cross compile environment, run "./configure ${CONFIGURE_FLAGS}", "make", "make install DESTDIR=/opt/tmp" Note: Currently, only i586_i586, x86-64_x86-64 and i586_$X(x is mips, arm and ppc) toolchain tarballs are covered in testing.
Expected Results: iptables could be compiled successfully

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	sdk
target:	build_system
image profile:	lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1636: ADT toolchain could build sudoku-savant project	
Summary: sudoku-savant could be compiled with ADT toolchain	
Steps: 1. Install toolchain tarball and setup cross compile environment 2. Download sudoku-savant project, http://downloads.sourceforge.net/project/sudoku-savant/sudoku-savant/sudoku-savant-1.3/sudoku-savant-1.3.tar.bz2 3. With the cross compile environment, run "./configure ${CONFIGURE_FLAGS}", "make", "make install DESTDIR=/opt/tmp" Note: Currently, only i586_i586, x86-64_x86-64 and i586_$X(x is mips, arm and ppc) toolchain tarballs are covered in testing.	
Expected Results: sudoku-savant could be compiled successfully	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	sdk
target:	build_system
image profile:	lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1637: unfs support for qemu target	
Summary: Check if unfs works for qemu target	
Steps: 1. Prepare a *rootfs.tar.bz2 image 2. Prepare a folder under poky directory as <rootfs-dir>, for example poky/temp 3. Run command "runqemu-extract-sdk *rootfs.tar.bz2 poky/temp" 4. Run command "runqemu nfs <kernel> <rootfs-dir>"	
Expected Results: QEMU target should be started with unfs	
Test Execution Cycle Type:	Weekly

Case Automation Type:	Manual
Case State:	Ready
Feature:	sdk
target:	qemux86_32, qemux86_64, qemuarm, qemuppc, qemumips
image profile:	sato, sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

1.4 Test Suite : Stress

Test Case TC-1638: crashme for stress	
Summary: Run crashme in real hardware for stress testing	
Steps: 1. Get crashme from http://people.delphiforums.com/gjc/crashme.html 2. By following the setup steps on above URL, build crashme in target. 3. Run crashme for 24 hours	
Expected Results: target should not crash with the program	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	stress
target:	beagleboard, jasperforest
image profile:	sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1640: ltp for stress
Summary: Run ltp stress in real hardware for stress testing
Steps: LTP download: http://sourceforge.net/projects/ltp/files/LTP%20Source/ltp-20101031/ltp-full-20101031.bz2/download build steps: refer to http://ltp.sourceforge.net Run steps: 1. Build LTP with toolchain or in sdk image 2. Copy LTP folder into target, for example, /opt/ltp. Modify script "testscripts/ltpstress.sh", set “Iostat=1”, ”NO_NETWORK=1” 3. cd testscripts/ && ./ltpstress.sh

4. This stress case will run for 24 hours	
Expected Results: Check the result, target should not crash with the program.	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	stress
target:	beagleboard
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

1.5 Test Suite : Mulitimedia

Test Case TC-1650: sound on/off	
Summary: check if sound can be turned on/off	
Steps: 1. copy amixer is installed 2. Run "amixer set Master on" to turn on audio device 3. Run "amixer set Master 64" to adjust to maxium volumn 4. Run "amixer set Speaker on" to turn on speaker 5. Run "amixer set Speaker 64" to adjust to maxium volumn 6. Run "amixer set Master off" to turn off audio device 7. Run "amixer set Speaker off" to turn off speaker	
Expected Results: Above commands can run without problem	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	multi-media
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1651: audio play (mp3)
Summary: make sure music player cannot play mp3 format file

Steps: 1. copy sample mp3 file to system 2. launch music player and make sure it cannot play the mp3 file	
Expected Results: mp3 file can not be played	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	multi-media
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1652: audio play (ogg)	
Summary: check if music player can play ogg format file	
Steps: 1. copy sample ogg file to system 2. launch music player can play the ogg file	
Expected Results: ogg file can be played without problem	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	multi-media
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1653: audio stop (ogg)
Summary: check if music player can play ogg format file
Steps: 1. copy sample ogg file to system 2. launch music player can play the ogg file 3. click "stop" button to stop playing 4. click "start" button to resume playing
Expected Results:

ogg file can be start/stop without problem	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	multi-media
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1654: audio play (wav)	
Summary: check if music player can play wav format file	
Steps: 1. copy sample wav file to system 2. launch music player can play the wav file	
Expected Results: wav file can be played without problem	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	multi-media
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1655: audio stop (wav)	
Summary: check if music player can stop playing with wav format file	
Steps: 1. copy sample wav file to system 2. launch music player can play the wav file 3. click "stop" button to stop playing 4. click "start" button to resume playing	
Expected Results: wav file can be start/stop without problem	
Test Execution Cycle Type:	Weekly
Case Automation	Manual

Type:	
Case State:	Ready
Feature:	multi-media
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1656: video play (mpeg)

Summary:

make sure video player cannot play mpeg format file

Steps:

1. copy sample mpeg file to system
2. launch video player and make sure it cannot play the mpeg file

Expected Results:

mpeg file cannot be played

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	multi-media
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1657: video play (ogg)

Summary:

check if video player can play ogg format file

Steps:

1. copy sample ogg file to system
2. launch video player can play the ogg file

Expected Results:

ogg file can be played without problem

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	multi-media
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato-sdk
Last Result	**Not Run**

Keywords:	None

Test Case TC-1658: video stop (ogg)	
Summary: check if video player can play ogg format file	
Steps: 1. copy sample ogg file to system 2. launch video player can play the ogg file 3. click "stop" button to stop playing 4. click "start" button to resume playing	
Expected Results: ogg file can be start/stop without problem	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	multi-media
target:	e-menlow, blacksand, beagleboard, crownbay, sugarbay
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

1.6 Test Suite : Compliance

Test Case TC-1659: LTP subset test suite
Summary: LTP subset test suite
Steps: For real hardware, run following component, syscalls fs fsx dio io mm ipc sched math nptl pty admin_tools timers commands For QEMU, run following component

syscalls
mm
ipc
sched
math
nptl
pty
admin_tools
commands

Run Instructions:
LTP download: http://sourceforge.net/projects/ltp/files/LTP%20Source/ltp-20110606/ltp-full-20110606.bz2/download

build steps: refer to http://ltp.sourceforge.net

Run steps:
1. Build LTP with toolchain or in sdk image
2. For QEMU, create the qemu target with "-m 512", which makes some memory stress cases pass. For some issues, we could only set 128M for qemuarm and 256M for qemumips.
3. Copy LTP folder into target, for example, /opt/ltp. Modify script "runltp", remove test suites not to be tested
4. Comment runtests/sched: hackbench, which is not suitable to run in emulators
5. Comment creat08, oom01, oom02, oom03, oom04, which consume lots of memory
6. Prepare a tmp folder under your ltp folder, for example, create a tmp folder under your ltp folder, like /opt/ltp/tmp
7. ./runltp -p -l result-M2-20101218.log -C result-M2-20101218.fail -d /opt/ltp/tmp &> result-M2-20101218.fulllog

(assume you mount your LTP disk at /opt and create your own tmp dir at /opt/ltp/tmp)

Expected Results:

Check the result on wiki, https://wiki.yoctoproject.org/wiki/LTP_result, there should be no regression failure met.

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Semi-Auto
Case State:	Ready
Feature:	core
target:	qemuarm, qemuppc, qemumips, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, sugarbay
image profile:	sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1660: POSIX subset test suite

Summary:

Run subset test suite of POSIX test suite

Steps:

POSIX test suite download: http://sourceforge.net/projects/posixtest/files/posixtest/posixtestsuite-1.5.2/posixtestsuite-1.5.2.tar.gz/download
build: refer to http://posixtest.sourceforge.net/

Run steps:
1. Get POSIX test suite as above
2. Start target and copy test suite into it

3. For qemu, option "-m 512" should be added 4. Make sure below is uncommented from LDFLAGS file: #-D_XOPEN_SOURCE=600 –lpthread –lrt –lm 5. For gcc 4.6, you need to add "-Wno-unused-but-set-variable -Wno-address" to CFLAGS in Makefile 6. Run following commands under POSIX test suite run_tests SIG run_tests SEM run_tests THR run_tests TMR run_tests MSG run_tests TPS run_tests MEM	
Expected Results: Compare the test result on wiki, https://wiki.yoctoproject.org/wiki/Posix_result, there should be no more regression failures met.	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Semi-Auto
Case State:	Ready
Feature:	core
target:	qemuarm, qemuppc, qemumips, blacksand, beagleboard, mpc8315e-rdb, routerstationpro, sugarbay
image profile:	sato-sdk, lsb-sdk
Last Result	**Not Run**
Keywords:	None

1.7 Test Suite : Core Build System

Test Case TC-1662: Init scripts
Summary: Provide an image/recipe skeleton as a canonical example. Check if can be built and run correctly
Steps: 1. Build image from poky source, check if skeleton script and skeleton-test can be built into the image a. download poky source b. modify the line IMAGE_FEATURES += "apps-console-core ${SATO_IMAGE_FEATURES}" to IMAGE_FEATURES += "apps-console-core ${SATO_IMAGE_FEATURES}} service" in meta/recipes-sato/images/core-image-sato.bb (for sato image) or core-image-sato-sdk.bb (for sato-sdk image) c. $ source oe-init-build-env

add line “<POKY_BASE>/meta-skeleton \” to conf/bblayer.conf d. build the image e. boot up the image, check the skeleton and skeleton-test should be in right place /etc/init.d/skeleton /usr/sbin/skeleton-test 2. Verify the basic function of skeleton. Check if skeleton script can start/stop the skeleton-test daemon.
Expected Results: Init scripts can be built and run correctly

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1663: Minimal image
Summary: Check if the minimal image can be built and run correctly.
Steps: 1. Build a minimal image from poky source by following the wiki: https://wiki.yoctoproject.org/wiki/Minimal_Image 2. Check the size of the image. It should take less than 5M disk space after extraction. 3. Verify the basic function of the image. Run “busybox –list” to get the commands list. Check if these commands can run correctly.
Expected Results: The minimal image can be built and run correctly.

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1664: Share gcc work directories	
Summary: This feature make gcc use the shared source directory during the different building. Check if this feature can work for gcc 4.5.1 and gcc 4.6.0.	
Steps: 1. Download the poky source and set build environment. 2. For gcc 4.5.1, add 2 lines to conf/local.conf : GCCVERSION ?= "4.5.1" SDKGCCVERSION ?= "4.5.1" For gcc 4.6.1, there is no need to add these 2 lines to conf/local.conf 3. Run bitbake command as below: bitbake gcc-cross bitbake gcc-cross gcc-cross-initial gcc-cross-intermediate -c clean bitbake gcc-crosssdk bitbake gcc-runtime bitbake libgcc bitbake gcc-cross-canadian-arm (for arm arch) bitbake gcc-cross-canadian-powerpc (for ppc arch) bitbake gcc-cross-canadian-mips (for mips arch) 4. Run "bitbake core-image-minimal", "bitbake core-image-sato", "bitbake core-image-sato-sdk" to build images. Verify the basic function of the images.	
Expected Results: After step3, you can check the tmp/work-shared/gcc-4.6.0 or tmp/work-shared/gcc-4.5.1 should in the build directory. Check the time of build process and the disk space usage of tmp/work-shared/gcc-version sub-directory. The images should be built and can work correctly.	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1665: ccache as native tool
Summary: ccache - a fast C/C++ compiler cache.

Steps:
 	
1. Make sure the native ccache is not installed on local machine and compile 'less' bbfile without native ccache support.
  bitbake ccache-native -c clean
  bitbake less -c clean
  bitbake less -c compile
Check the compile log under .../tmp/work/mips-poky-linux/less-443-r0/temp/log.do_compile
 	
2. Build native tool 'ccache'
  bitbake ccache-native

Check the ccache-native installed location ..tmp/sysroots/x86_64-linux/usr/bin/ccache 3. Compile less bbfile again with native ccache support bitbake less -c clean bitbake less -c compile Check the compile with ccache log under .../tmp/work/mips-poky-linux/less-443-r0/temp/log.do_compile. The native ccache should be used when compiled.
Expected Results: The ccache-native should be built successfully and be installed to the correct location. The ccache-navive will be used when compile file.

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1666: PAM support
Summary: Check the Yocto should support PAM (Pluggable Authentication Module)
Steps: 1. Build a sato-sdk image from poky source with PAM support by following the wiki: https://wiki.yoctoproject.org/wiki/PAM_Integration 2. Refer to https://wiki.yoctoproject.org/wiki/PAM_Integration , check the commands 'dropbear', 'login', 'passwd', 'useradd', 'su' can work correctly with PAM support and verify the function of PAM.
Expected Results: The commands which have PAM support should run correctly and the function of PAM should work without problems.

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1667: kernel interactive targets
Summary: Check if yocto can support kernel interactive target build
Steps: 1. download yocto source tree

2. prepare yocto build environment 3. Run "bitbake linux-yocto -c menuconfig" 4. Check if a new bash terminal pop up and menuconfig can be triggered
Expected Results: menuconfig for kernel can be triggered with yocto build command

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1668: KVM enabled with qemu
Summary: qemu can be started with KVM enabled
Steps: 1. build a kernel with KVM enabled 2. Start qemu with option "kvm" with runqemu 3. Check if qemu starts up and if kvm_intel is used 4. If kvm_intel is not used when starting qemu, it will shows 0 in "Used by" column when you run "lsmod | grep kvm_intel"
Expected Results: KVM enabled with qemu

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1669: non-GPLv3 build check
Summary: Check if non-GPLv3 build could pass and it does not has any GPLv3 packages installed
Steps: 1. Set following sentences in local.conf to GPLv3 ##### INCOMPATIBLE_LICENSE = "GPLv3" ##### 2. Build core-image-minimal and core-image-basic

3. Start up target after build is finished 4. Run following script to check if any GPLv3 packages installed, some packages are GPLv3 exception, like libgcc1, libstdc++ and less.

```
##################
#!/bin/sh

temp=`mktemp`
rpm -qa > $temp
ret=0

for i in `cat $temp`
do
        rpm -qi $i | grep License | grep -i gplv3 > /dev/null 2>&1
        if [ $? -eq 0 ]; then
                license=`rpm -qi $i | grep License | awk -F"License:" '{print
$2}'`
                echo "package $i has inconsistent license: $license"
                ret=1
        fi
done

rm -rf $temp
exit $ret
###################
```

Expected Results: non-GPLv3 build pass and no GPLv3 packages installed in the image	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1670: yocto build in Fedora 15	
Summary: Build latest yocto in x86_64 Fedora 15 host	
Steps: 1. By following the yocto handbook, download latest yocto source 2. Build core-image-minimal on Fedora 15	
Expected Results: Yocto build should pass on Fedora 15	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system

image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1671: yocto build in OpenSuse 11.4	
Summary: Build latest yocto in x86_64 OpenSuse 11.4	
Steps: 1. By following the yocto handbook, download latest yocto source 2. Build core-image-minimal on OpenSuse 11.4	
Expected Results: Build should pass on OpenSuse 11.3	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1672: yocto build in Ubuntu 11.04	
Summary: Build latest yocto in x86_64 Ubuntu 11.04	
Steps: 1. By following the yocto handbook, download latest yocto source 2. Build core-image-minimal on Utuntu 11.04	
Expected Results: Yocto build should pass on Utuntu 10.04	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1673: yocto build in KVM	
Summary: Build yocto in KVM should work	
Steps: 1. Setup a VM environment with KVM enabled, for example, RHEL6 2. Prepare a VM for yocto build testing, for example, OpenSuse 11.3 3. By following the yocto handbook, download latest yocto source into the VM 4. Build core-image-minimal in the VM	
Expected Results: Yocto build in VM should work same as in real host	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1674: sstate work on local host	
Summary: Check if sstate could work with local cache	
Steps: 1. Follow the wiki steps to setup a sstate cache on local machine, https://wiki.yoctoproject.org/wiki/Enable_sstate_cache 2. Prepare another yocto source directory and set the SSTATE_DIR the cache you setup in step 1) 3. Run poky build, for example, "bitbake core-image-minimal". You should note following things if sstate works: ######## NOTE: Preparing runqueue NOTE: Executing SetScene Tasks NOTE: Running setscene task 118 of 155 (virtual:native:/home/lulianhao/poky-build/edwin/poky/meta/recipes-devtools/pseudo/pseudo_git.bb:do_populate_sysroot_setscene) NOTE: Running setscene task 119 of 155 (/home/lulianhao/poky-build/edwin/poky/meta/recipes-devtools/quilt/quilt-native_0.48.bb:do_populate_sysroot_setscene ########	
Expected Results: sstate should work and reduce build time	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**

Keywords:	None

Test Case TC-1675: gcc set to 4.5.1 for core build	
Summary: gcc related options should be set to 4.5.1 for 4.5.1 build	
Steps: 1. Download poky source and prepare the build environment 2. Set GCCVERSION and SDKGCCVERSION to 4.5.1 in meta/conf/distro/include/tcmode-default.inc 3. Run "bitbake -s | grep gcc" and check the output, all gcc related options should be set to 4.5.1	
Expected Results: all gcc related options should be set to 4.5.1	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1676: btrfs format image build	
Summary: btrfs format image could be built out	
Steps: 1. set IMAGE_FSTYPES = "btrfs" and KERNEL_FEATURES_append = " cfg/btrfs " in local.conf 2. build a core-image-minimal image, the image should be btrfs format	
Expected Results: btrfs format image could be built out	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1677: btrfs format image boot up
Summary:

btrfs format image could be booted up	
Steps: 1. set IMAGE_FSTYPES = "btrfs" and KERNEL_FEATURES_append = " cfg/btrfs " in local.conf 2. build a qemux86 core-image-minimal image and boot up it	
Expected Results: btrfs format image could be booted up	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1678: lib64-zlib lib32-zlib build	
Summary: lib64-zlib lib32-zlib build should pass with multilib enabled	
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build 3. build lib64-zlib and lib32-zlib, which should build pass without error	
Expected Results: lib64-zlib lib32-zlib build should pass with multilib enabled	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1679: lib32 sato image build - qemux86
Summary: lib32 sato image could be built out with multilib support
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set

MACHINE to qemux86 3. with rpm set for package format, build lib32 core-sato image 4. after build finished, start up the image and check if all app are 32-bit, kernel with 32-bit
Expected Results: lib32 sato image could be built out with multilib support

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1680: lib32 sato image build - qemux86-64
Summary: lib32 sato image could be built out with multilib support
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86 3. with rpm set for package format, build lib32 core-sato image 4. after build finished, start up the image and check if all app are 32-bit, kernel with 64-bit
Expected Results: lib32 sato image could be built out with multilib support

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1681: lib64 sato image build - qemux86
Summary: lib64 sato image should be built out with multilib support
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86 3. with rpm set for package format, build lib64 core-sato image 4. after build finished, start up the image and check if all app are 64-bit, kernel with 32-bit

Expected Results: lib64 sato-sdk image should be built out with multilib support	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1682: lib64 sato image build - qemux86-64	
Summary: lib64 sato image should be built out with multilib support	
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86 3. with rpm set for package format, build lib64 core-sato image 4. after build finished, start up the image and check if all app are 64-bit, kernel with 64-bit	
Expected Results: lib64 sato-sdk image should be built out with multilib support	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1683: lib64 sato image build - qemux86-64/ipk
Summary: lib64 sato image should be built out with multilib support
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86 3. with ipk set for package format, build lib64 core-sato image 4. after build finished, start up the image and check if all app are 64-bit, kernel with 64-bit

Expected Results: lib64 sato-sdk image should be built out with multilib support	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1684: lib64 sato image build - qemux86-64/deb	
Summary: lib64 sato image should be built out with multilib support	
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86 3. with deb set for package format, build lib64 core-sato image 4. after build finished, start up the image and check if all app are 64-bit, kernel with 64-bit	
Expected Results: lib64 sato-sdk image should be built out with multilib support	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1685: lib32 connman-gnome built for qemux86-64 - rpm
Summary: build lib32 connman-gnome and include it in qemux86-64 image
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86-64 3. set "MULTILIB_IMAGE_INSTALL = "lib32-connman-gnome"" 4. with rpm set for package format, build core-sato image 5. after build finished, start up the image and check if connman and related packages are 32-bit

Expected Results: user could build lib32 connman-gnome and include it in qemux86-64 image	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	core
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1686: lib32 connman-gnome built for qemux86-64 - ipk	
Summary: build lib32 connman-gnome and include it in qemux86-64 image	
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86-64 3. set "MULTILIB_IMAGE_INSTALL = "lib32-connman-gnome"" 4. with ipk set for package format, build core-sato image 5. after build finished, start up the image and check if connman and related packages are 32-bit	
Expected Results: user could build lib32 connman-gnome and include it in qemux86-64 image	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	core
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1687: lib32 connman-gnome built for qemux86-64 - deb
Summary: build lib32 connman-gnome and include it in qemux86-64 image
Steps: 1. Prepare poky build environment 2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86-64 3. set "MULTILIB_IMAGE_INSTALL = "lib32-connman-gnome"" 4. with deb set for package format, build core-sato image 5. after build finished, start up the image and check if connman and related packages are 32-bit

Expected Results: user could build lib32 connman-gnome and include it in qemux86-64 image	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	core
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1688: bitbake-layers show_layers	
Summary: show_layers could show current layers	
Steps: 1. prepare poky build environment 2. add meta-rt into bblayer.conf 3. run "bitbake-layers show_layers", it should show the layers defined in bblayer.conf	
Expected Results: show_layers could show current layers	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1689: bitbake-layers show_overlayed
Summary: overlayed recipes should be shown with bitbake-layers
Steps: 1. prepare poky build environment 2. copy a recipe from meta layer into meta-yocto, for example, /home/jxu49/osel/poky/meta/recipes-graphics/clutter/clutter-1.6_1.6.14.bb 3. run "bitbake-layers show_overlayed", it should report clutter is overlayed by meta-yocto
Expected Results: overlayed recipes should be shown with bitbake-layers

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1690: bitbake-layers show_appends	
Summary: bitbake-layers show_appends should list bbappend files and recipe files they apply to	
Steps: 1. prepare poky build environment 2. run "bitbake-layers show_appends", it should list bbappend files and recipe files they apply to	
Expected Results: bitbake-layers show_appends should list bbappend files and recipe files they apply to	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1691: bitbake-layers flatten	
Summary: bitbake-layers flattens layer configuration into a separate output directory	
Steps: 1. prepare poky build environment 2. create a folder, for example, test 3. run "bitbake-layers flatten test", all contents of all layers should be moved into the test folder, with any bbappends appended to corresponding recipes 4. check if bbappends take effect, for example, check if test/recipes-bsp/formfactor/formfactor_0.0.bb has the code defined in meta-yocto/recipes-bsp/formfactor/formfactor_0.0.bbappend	
Expected Results: bitbake-layers flattens layer configuration into a separate output directory	
Test Execution Cycle Type:	Fullpass
Case Automation	Manual

Type:	
Case State:	Ready
Feature:	poky
target:	build_system
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1692: x32 image build	
Summary: x32 image could be built out successfully	
Steps: 1. Prepare yocto build environment 2. add meta-x32 layer, http://git.yoctoproject.org/cgit/cgit.cgi/experimental/meta-x32/ 3. Add following lines in your conf/local.conf MACHINE = "qemux86-64" DEFAULTTUNE = "x86-64-x32"	
Expected Results: x32 image could be built out successfully	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	core
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

Test Case TC-1693: x32 image build boot up and check
Summary: x32 image could be built out successfully and binaries/libraries are x32 in it
Steps: 1. Prepare yocto build environment 2. add meta-x32 layer, http://git.yoctoproject.org/cgit/cgit.cgi/experimental/meta-x32/ 3. Add following lines in your conf/local.conf MACHINE = "qemux86-64" DEFAULTTUNE = "x86-64-x32" 4. build minimal image with "bitbake core-image-minimal" 5. Run the file command to know what type of elf binary is it. It should be 32bit x86-64 elf binary as seen here: $ file bin/busybox bin/busybox: setuid ELF 32-bit LSB executable, x86-64, version 1 (SYSV), dynamically linked (uses shared libs), for GNU/Linux 2.6.35, not stripped $file usr/lib/libz.so.1.2.5

usr/lib/libz.so.1.2.5: ELF 32-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, not stripped	
Expected Results: x32 image could be built out successfully and binaries/libraries are x32 in it	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	core
target:	
image profile:	
Last Result	**Not Run**
Keywords:	None

1.8 Test Suite : BSP specific

Test Case TC-1694: RTC	
Summary: Check if RTC(Real Time Clock) can work correctly	
Steps: 1. Read time from RTC registers. root@localhost:/root> hwclock -r Sun Mar 22 04:05:47 1970 -0.001948 seconds 2. Set system current time root@localhost:/root> date 062309452008 3. Synchronize the system current time to RTC registers root@localhost:/root> hwclock -w 4. Read time from RTC registers root@localhost:/root> hwclock -r 5. Reboot target and read time from RTC again.	
Expected Results: Can read and set the time successful	
Test Execution	Weekly

Cycle Type:	
Case Automation Type:	Manual
Case State:	Ready
Feature:	bsp
target:	beagleboard, mpc8315e-rdb
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1695: Watchdog	
Summary: Check if watchdog can reset the target system	
Steps: 1. Check if watchdog device exist in /dev/ directory 2. Run command “echo 1 > /dev/watchdog” and wait for 60s. Then the target will reboot.	
Expected Results: The watchdog device exist in /dev/ directory and can reboot the target.	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	bsp
target:	beagleboard, routerstationpro
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1696: SATA
Summary: Test general use of SATA device on target, like mount, umount, read and write.
Steps: 1. Run “fdisk” command to create partition on SATA disk. 2. Mount/Umount mke2fs /dev/sda1

mount -t ext2 /dev/sda1 /mnt/disk

umount /mnt/disk

3. Read/Write (filesystem)

touch /mnt/disk/test.txt

echo “abcd” > /mnt/disk/test.txt

cat /mnt/disk/test.txt

4. Read/Write (raw)

dd if=/dev/sda1 of=/tmp/test bs=1k count=1k

This command will read 1MB from /dev/sda1 to /tmp/test

Expected Results:

The SATA device can mount, umount, read and write

Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	bsp
target:	mpc8315e-rdb
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

Test Case TC-1697: I2C/EEPROM

Summary:

Check if target can support EEPROM

Steps:

1. Check eeprom device exist in /sys/bus/i2c/devices/

2. Run "hexdump eeprom" command

```
root@mpc8315e-rdb:/sys/bus/i2c/devices/1-0051> hexdump eeprom

0000000 9210 0b02 0211 0009 0b52 0108 0c00 3c00

0000010 6978 6930 6911 208c 7003 3c3c 00f0 8381
```

Expected Results:

Hexdump can read data from eeprom	
Test Execution Cycle Type:	Weekly
Case Automation Type:	Manual
Case State:	Ready
Feature:	bsp
target:	mpc8315e-rdb
image profile:	sato-sdk
Last Result	**Not Run**
Keywords:	None

1.9 Test Suite : NAS

Test Case TC-1698: Baryon build	
Summary: Baryon image could be built with 1.1.1 branch	
Steps: 1. Get baryon source from http://git.yoctoproject.org/cgit/cgit.cgi/meta-baryon/ and put it under poky source directory 2. Get meta-intel source and put it under poky source directory 3. Set MACHINE to n450 and set DISTRO to baryon 4. run "bitbake baryon" to build an image for n450	
Expected Results: Baryon image could be built with 1.1.1 branch	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	undecided
Last Result	**Not Run**
Keywords:	None

Test Case TC-1699: baryon image could boot up
Summary: baryon image could boot up without issue
Steps: 1. get baryon image for n450 2. burn it on n450 and boot up it
Expected Results: baryon image could boot up without issue

Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	undecided
Last Result	**Not Run**
Keywords:	None

Test Case TC-1700: webmin start up as web interface	
Summary: webmin is started by default and accessible via http port 10000	
Steps: 1. start up baryon image on n450 2. check the ip address of n450 and access its port 10000 via http	
Expected Results: webmin is started by default and accessible via http port 10000	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	undecided
Last Result	**Not Run**
Keywords:	None

Test Case TC-1701: proftpd configure via webmin	
Summary: proftpd should be configurable and workable via webmin	
Steps: 1. start up baryon image on n450 2. configure Proftpd by clicking Servers->Proftpd in webmin 3. click "Files and Directories" and expose a directory for user 4. on remote machine, connect to n450 via ftp and upload/download some files from it	
Expected Results: proftpd should be configurable and workable via webmin	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	undecided
Last Result	**Not Run**
Keywords:	None

Test Case TC-1702: samba configure via webmin	
Summary: samba should be configurable and workable via webmin	
Steps: 1. start up baryon image on n450 2. configure samba by clicking Servers->Samba Windows File in webmin 3. click "Create a new file share" and expose a directory for user 4. on remote machine, connect to n450 via samba and upload/download some files from it	
Expected Results: samba should be configurable and workable via webmin	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	undecided
Last Result	**Not Run**
Keywords:	None

Test Case TC-1703: NFS configure via webmin	
Summary: NFS should be configurable and workable via webmin	
Steps: 1. start up baryon image on n450 2. configure NFS by clicking Networking->NFS Exports in webmin 3. click "Add a new export" and expose a directory for user 4. on remote machine, connect to n450 via NFS and upload/download some files from it	
Expected Results: NFS should be configurable and workable via webmin	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	undecided
Last Result	**Not Run**
Keywords:	None

Test Case TC-1704: mediatomb configure via webmin
Summary: mediatomb should be configurable and workable via webmin
Steps: 1. start up baryon image on n450 2. configure mediatomb by clicking Others->Media Tomb in webmin 3. click the link provided by webmin and you should be redirected to media tomb web interface

4. in mediatomb, add/remove/move files and check if above modification could work	
Expected Results: mediatomb should be configurable and workable via webmin	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	undecided
Last Result	**Not Run**
Keywords:	None

Test Case TC-1705: user configuration via webmin	
Summary: user could be configured via webmin	
Steps: 1. start up baryon image on n450 2. configure user and its group by clicking System->Users and Groups in webmin 3. click "Create a new user" and add a user "test" 4. check if the new added test exists	
Expected Results: user could be configured via webmin	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual
Case State:	Ready
Feature:	undecided
Last Result	**Not Run**
Keywords:	None

Test Case TC-1706: Soft RAID configuration via webmin	
Summary: Soft RAID could be configurable and workable via webmin	
Steps: 1. start up baryon image on n450 and connect 2 extra harddisk to it 2. configure RAID group by clicking Others->Linux RAID in webmin 3. configure the extra 2 harddisk to be RAID0 4. check by fdisk if the RAID could work	
Expected Results: Soft RAID could be configurable and workable via webmin	
Test Execution Cycle Type:	Fullpass
Case Automation Type:	Manual

Case State:	Ready
Feature:	undecided
Last Result	**Not Run**
Keywords:	None

Reports and Metrics