# Yocto 1.1 M4 Fullpass Test

# Test Report

Project: yocto

Author: admin

Printed by TestLink on 07/09/2011



## 1 Test Suite : Yocto 1.1 M4 Fullpass Test

## 1.1 Test Suite : hob

| **Test Case TC-1067: hob launch without error** | |
|---|---|
| Summary:<br><br>hob could be launched without error | |
| Steps:<br><br>1. Prepare poky build environment<br>2. launch hob with command "hob"<br>3. Check if hob is launched correctly and no error message in console | |
| Expected Results:<br><br>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-1068: package dependency check - acl** | |
|---|---|
| Summary:<br><br>acl package dependency selection should be correct | |
| Steps:<br><br>1. launch hob<br>2. select acl, 3 dependency packages should be shown in "image contents": acl, attr and ncurses<br>3. deselect acl, package acl should be removed<br>4. redo step 2,3 and the remaining packages in "image contents" should be consistent with the outputs in step 3 | |
| Expected Results:<br><br>the package list should be always consistent before/after several times package select/deselect | |
| 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-1069: package dependency check - avahi&acl** | |
|---|---|
| Summary:<br><br>avahi and acl package selection should not cause wrong package dependency | |
| Steps:<br><br>1. launch hob<br>2. select avahi, a lot of dependency packages should be shown in "image contents"<br>3. select acl, it should be shown in "image contents" also<br>4. deselect avahi, acl, attr and ncurses should not be removed from "image contents"<br>5. deselect acl, it should be removed from "image contents"<br>4. redo step 2~5 and the remaining packages in "image contents" should be consistent with the outputs in step 5 | |
| Expected Results:<br><br>Package list shown in image contents should be consistent before/after several times acl&avahi selection/deselection | |
| 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-1070: base image selection** | |
|---|---|
| Summary:<br><br>package list should be loaded for "image contents" for each selection in "base image" field | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemumips<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents" | |
| Expected Results:<br><br>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-1071: package list re-load for "base image" change** | |
|---|---|
| Summary:<br><br>package list should be re-loaded if changing image type for "base image" | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemumips<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents"<br>5. change the image type for "Base image", for example, "core-image-minimal", the list of packages should be re-loaded | |
| Expected Results:<br><br>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-1072: package list re-load for "Machine" change** | |
|---|---|
| Summary:<br><br>package list for "image contents" should be re-loaded and correct when "Machine" changing | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemuppc<br>3. select one image for "Base image", for example, "core-image-sato"<br>4. a list of packages should be loaded for "image contents"<br>5. select another machine type for "Machine", for example, beagleboard<br>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:<br><br>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-1073: package list re-load correct for "Machine" change** | |
|---|---|
| Summary:<br><br>package list re-load correct for "Machine" change | |
| Steps:<br><br>1. launch hob<br>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<br>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:<br><br>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-1074: package list reset** | |
|---|---|
| Summary:<br><br>reset button should clear package list for "image contents" | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemumips<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents"<br>5. click "reset" button, all packages should be cleared for "image contents" | |
| Expected Results:<br><br>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-1075: customized package list save as bb file(add packages)** |
|---|
| Summary: |

| |
|---|
| user could use "save" or "save as" button to save customized bb file |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemumips<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents"<br>5. select some un-selected package, for example, acpid<br>6. click "File"->"Save" or "Save As", it should save the user customized package list into a bb file<br>7. click "reset" button, and click "File"->"Open", choose the saved bb file<br>8. The user customized package list should be shown |
| Expected Results:<br><br>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-1077: cancel customized package list save action** |
|---|
| Summary:<br><br>cancel customized package list save action should not cause any error |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemux86-64<br>3. select one image for "Base image", for example, "core-image-minimal"<br>4. a list of packages should be loaded for "image contents"<br>5. select some un-selected package, for example, acpid<br>6. click "x" button, a dialog should pop up and ask user if customiszations wants be saved.<br>7. click "yes" and click "cancel" in next page<br>8. hob should exit without error log |
| Expected Results:<br><br>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-1078: No native package shown in package list** | |
|---|---|
| Summary:<br><br>There should be no native package shown in package list | |
| Steps:<br><br>1. launch hob<br>2. check if there is any -native package in "Packages" | |
| Expected Results:<br><br>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-1079: stop build during image/package building** | |
|---|---|
| Summary:<br><br>"stop build" button should be able to stop/force stop building | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemuarm<br>3. select one image for "Base image", for example, "core-image-sato"<br>4. a list of packages should be loaded for "image contents"<br>5. select some un-selected package, for example, acpid<br>6. click "bake" button to start build<br>7. in building page, click "stop build", and click "stop" or "force stop" to stop building | |
| Expected Results:<br><br>"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-1080: search package name in package list** |
|---|
| Summary: |

| |
|---|
| User could search package name from "Search packages" |
| Steps:<br><br>1. launch hob<br>2. search some package via "search packages", for example, avahi<br>3. the searched package should be shown in "packages" |
| Expected Results:<br><br>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-1081: task list re-load when base image change** |
|---|
| Summary:<br><br>task list for "package collections" should be re-loaded when base image changing |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemuppc<br>3. select one image for "Base image", for example, "core-image-sato"<br>4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections"<br>5. select another image type for "base image", for example, core-image-basic<br>6. a new list of tasks should be re-loaded |
| Expected Results:<br><br>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-1082: user could customize threads of bitbake and make** |
|---|
| Summary:<br><br>user could customize threads of bitbake and make in hob |

| |
|---|
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemux86<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections"<br>5. click Edit->Preferences, and customize number for "bitbake threads" and "make threads", for example, you could set both 1 for them<br>6. click "bake" and check 'ps' command output if there is one thread running |
| Expected Results:<br><br>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-1083: add layer for new target build** |
|---|
| Summary:<br><br>user could add layer for new target build |
| Steps:<br><br>1. launch hob<br>2. click File->Add Layer, then choose one layer, for example, you could download meta-intel.git and use sugarbay<br>3. check "Machine" list and sugarbay should be available<br>4. choose one type, for example, core-image-sato-sdk<br>5. click "bake" and check the build result |
| Expected Results:<br><br>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-1084: another build after stop build** |
|---|
| Summary: |

| |
|---|
| user could start another build after stop a build |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemuarm<br>3. select one image for "Base image", for example, "core-image-sato"<br>4. a list of packages should be loaded for "image contents"<br>5. select some un-selected package, for example, acpid<br>6. click "bake" button to start build<br>7. in building page, click "stop build", and click "stop" to stop building<br>8. back to the main UI, and select another image, then click "bake" button<br>9. wait for the build finished and it should be no error met |
| Expected Results:<br><br>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-1085: back to main UI after back stopped** |
|---|
| Summary:<br><br>click "back" button should bake to main UI after bake  stopped |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemuarm<br>3. select one image for "Base image", for example, "core-image-sato"<br>4. a list of packages should be loaded for "image contents"<br>5. select some un-selected package, for example, acpid<br>6. click "bake" button to start build<br>7. in building page, click "stop" or "force stop"<br>8. click "back" button, it should return to main UI |
| Expected Results:<br><br>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-1086: customized preference items save in local.conf** | |
|---|---|
| Summary:<br><br>user customized items should be saved in local.conf or hob.local.conf | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemumips<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents"<br>5. select some un-selected package, for example, acpid<br>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"<br>6. exit hob<br>7. check local.conf, above modifications should be set in it<br>8. re-launch hob and check "Preferences", all above modifications should be set in this page | |
| Expected Results:<br><br>user customized items should be saved in local.conf or hob.local.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-1087: bake a image without error (base image)** | |
|---|---|
| Summary:<br><br>user could use hob to build a image without error | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemumips<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents"<br>5. click "Bake" and wait for a successful build finished | |
| Expected Results:<br><br>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-1088: bake a image without error (added package)** | |
|---|---|
| Summary:<br><br>user could use hob to build a image without error | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemumips<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents"<br>5. select some un-selected package, for example, acpid<br>6. click "Bake" and wait for a successful build finished<br>7. after build finished, check if the added package built into image | |
| Expected Results:<br><br>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-1090: back to main UI after bake finished** | |
|---|---|
| Summary:<br><br>click "back" button should bake to main UI after bake finished | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemuarm<br>3. select one image for "Base image", for example, "core-image-sato"<br>4. a list of packages should be loaded for "image contents"<br>5. select some un-selected package, for example, acpid<br>6. click "bake" button to start build<br>7. in bake page, wait for build finished<br>8. click "back" button, it should return to main UI | |
| Expected Results:<br><br>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-1091: toolchain built correct with user customization** | |
|---|---|
| Summary:<br><br>toolchain generated correct with user selection | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, beagleboard<br>3. select one image for "Base image", for example, "core-image-sato"<br>4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections"<br>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<br>6. click "bake" button and it should generate toolchain as well as selected packages/images<br>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<br>8. use the toolchain to build a C program and make sure it workable in target | |
| Expected Results:<br><br>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-1092: non-GPLv3 build** | |
|---|---|
| Summary:<br><br>non-GPLv3 build should be supported for hob | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemux86<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections"<br>5. click Edit->Preferences, and select "Exclude GPLv3 packages"<br>6. click "bake" to build a non-GPLv3 image<br>7. After build is finished, check if there is any GPLv3 packages built in | |
| Expected Results:<br><br>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-1093: distribution selection for image/package build** | |
|---|---|
| Summary:<br><br>user could select different distribution for "distribution" | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemux86<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections"<br>5. click Edit->Preferences, and select different distribution for "distribution", for example, poky-lsb<br>6. click "bake" button and it should generate packages or image with selected distribution | |
| Expected Results:<br><br>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-1094: ipk package build for image/package build** | |
|---|---|
| Summary:<br><br>build image with ipk package format | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemux86<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections"<br>5. click Edit->Preferences, and select ipk for "package format"<br>6. click "bake" button and it should generate images with ipk format | |
| Expected Results:<br><br>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-1095: deb package build for image/package build** | |
|---|---|
| Summary:<br><br>build image with deb package format | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemux86<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections"<br>5. click Edit->Preferences, and select deb for "package format"<br>6. click "bake" button and it should generate images with dformat | |
| Expected Results:<br><br>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-1096: rpm package build for image/package build** | |
|---|---|
| Summary:<br><br>build image with rpm package format | |
| Steps:<br><br>1. launch hob<br>2. select one "Machine", for example, qemux86<br>3. select one image for "Base image", for example, "core-image-basic"<br>4. a list of packages should be loaded for "image contents" and you could find some tasks are select for "package collections"<br>5. click Edit->Preferences, and select rpm for "package format"<br>6. click "bake" button and it should generate images with rpm format | |
| Expected Results:<br><br>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-1097: zypper command installed and workable** | |
|---|---|
| Summary:<br><br>check if zypper is installed and can work | |
| Steps:<br><br>1. Run command "zypper", and check the output | |
| Expected Results:<br><br>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-1098: zypper help search** | |
|---|---|
| Summary:<br><br>check help option with zypper command | |
| Steps:<br><br>1. Run "zypper help search" and check the output | |
| Expected Results:<br><br>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-1099: zypper search package** | |
|---|---|
| Summary:<br><br>search package with zypper | |
| Steps:<br><br>1. Run "zypper search package_name" and check the output, for example "zypper search avahi" | |
| Expected Results:<br><br>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-1100: zypper remove package** | |
|---|---|
| Summary:<br><br>remove package with zypper | |
| Steps:<br><br>1. Run "zypper rm pakcage_name" and check the output, for example "zypper rm avahi" | |
| Expected Results:<br><br>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-1101: zypper install package** | |
|---|---|
| Summary:<br><br>install package with zypper | |
| Steps:<br><br>1. Set up a yum based repository on local server<br><br>2. Build out a package, which does not need any run-time dependency package, with local poky tree. For example, package "man"<br><br>3. In target system, run "zypper addrepo http://ip_address_of_repository zypper_test_repo"<br><br>4. Run "zypper refresh" to refresh the zypper repository cache<br><br>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:<br><br>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-1102: zypper install dependency package** | |
|---|---|
| Summary:<br><br>install dependency package with zypper | |
| Steps:<br><br>1. Set up a yum based repository on local server<br><br>2. Build out a package, which does not need any run-time dependency package, with local poky tree. For example, package "mc"<br><br>3. In target system, run "zypper addrepo http://ip_address_of_repository zypper_test_repo"<br><br>4. Run "zypper refresh" to refresh the zypper repository cache<br><br>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:<br><br>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-1103: zypper install .all packages** | |
|---|---|
| Summary:<br><br>install packages from all folder with zypper | |
| Steps:<br><br>1. Set up a yum based repository on local server<br>2. Build out a package, which belongs to all folder, for example, xcursor-transparent-theme-dbg-0.1.1-r3.all.rpm.<br>3. In target system, run "zypper addrepo http://ip_address_of_repository zypper_test_repo"<br>4. Run "zypper refresh" to refresh the zypper repository cache<br>5. Run "zypper install xcursor-transparent-theme-dbg" and check the output | |
| Expected Results:<br><br>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-1104: rpm query package** | |
|---|---|
| Summary:<br><br>make sure rootfs image is built with rpm packages | |
| Steps:<br><br>1. launch terminal<br><br>2. run command "rpm -qa", which lists all existing packages in system | |
| Expected Results:<br><br>"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-1105: rpm install package** | |
|---|---|
| Summary:<br><br>rpm format package can be installed | |
| Steps:<br><br>1. Get a RPM package(for example, avahi or powertop) from zypper repository or build one on local machine<br><br>2. Copy the package into image, run command "rpm -ivh package_name" to install the package | |
| Expected Results:<br><br>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-1106: rpm install dependency package** | |
|---|---|
| Summary:<br><br>rpm command should report dependency when installing package | |
| Steps:<br><br>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<br><br>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:<br><br>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-1107: rpm remove package** | |
|---|---|
| Summary:<br><br>rpm command can remove package in system | |
| Steps:<br><br>1. Launch terminal and run command "rpm -e package_name" to remove some package, for example, avahi | |
| Expected Results:<br><br>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-1108: check rpm install/removal log file size** | |
|---|---|
| Summary:<br><br>The case is to track log file size after rpm install/removal | |
| Steps:<br><br>1. After system is up, check the log file size after rpm/zypper install/removal<br>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.<br>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:<br><br>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-1109: boot and install from USB** | |
|---|---|
| Summary:<br><br>boot and install image from usb stick | |
| Steps:<br><br>1. plugin usb which contains live image burned<br>2. configure device BIOS to firstly boot from USB if necessary<br>3. boot the device and select some option like "Boot and Install" from boot menu<br>4. proceed through default install process<br>5. Remove USB, and reboot into new installed system. | |
| Expected Results:<br><br>1. User can choose install system from usb stick onto harddisk from boot menu or command line option<br>2. Imstalled system can boot up | |
| Test Execution Cycle Type: | Weekly |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | installation&boot |
| target: | e-menlow, blacksand, crownbay, sugarbay, jasperforest |
| image profile: | sato, sato-sdk, lsb-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1110: live boot from USB** | |
|---|---|
| Summary:<br><br>live boot from USB | |
| Steps:<br><br>boot live image from usb stick<br>1. plugin usb which contains live image burned<br>2. configure device BIOS to firstly boot from USB if necessary<br>3. boot the device and select some option like "boot from usb" from boot menu | |
| Expected Results:<br><br>1. User can choose boot from live image on usb stick from boot menu or command line option<br>2. Live image can boot up with usb stick | |
| Test Execution Cycle Type: | Weekly |
| Case Automation Type: | Manual |
| Case State: | Ready |

| | |
|---|---|
| Feature: | installation&boot |
| target: | e-menlow, blacksand, crownbay, sugarbay, jasperforest |
| image profile: | sato, sato-sdk, lsb-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1111: boot from runlevel 3** | |
|---|---|
| Summary:<br><br>Verify that system can boot from runlevel 3 | |
| Steps:<br><br>1. Boot into system and edit /etc/inittab to make sure system enter init 3 by default<br><br>########<br><br>id:3:initdefault<br><br>########<br><br>2. reboot system, and press Tab to enter "grub"<br>3. edit "kernel" line and add "psplash=false text" at the end<br>4. Press "enter" to boot system | |
| Expected Results:<br><br>system should boot to runlevel 3. | |
| Test Execution Cycle Type: | Weekly |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | installation&boot |
| target: | e-menlow, blacksand, crownbay, sugarbay, jasperforest |
| image profile: | sato, sato-sdk, lsb-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1112: boot from runlevel 5** |
|---|
| Summary:<br><br>Verify that system can boot from runlevel 5 |
| Steps:<br><br>1. Boot into system and edit /etc/inittab to make sure system enter init 5 by default<br><br>########<br><br>id:5:initdefault<br><br>########<br><br>2. reboot system, and press Tab to enter "grub" |

3. edit "kernel" line and make sure no "psplash=false text" in grub cmdline
4. Press "enter" to boot system

Note: The test is only for sato image.

Expected Results:

system should boot to runlevel 5.

| | |
|---|---|
| Test Execution Cycle Type: | Weekly |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | installation&boot |
| target: | e-menlow, blacksand, crownbay, sugarbay, jasperforest |
| image profile: | sato, sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

**Test Case TC-1113: 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-1114: 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-1115: run command make in sdk image** | |
|---|---|
| Summary:<br><br>check if command make can work in sdk image | |
| Steps:<br><br>1. Boot up sdk image<br>2. check if make is built in<br>3. run command "make" with following makefile and build the test.c file from case "gcc compile in sdk image"<br><br>test: test.o<br>gcc -o test test.o -lm<br>test.o: test.c<br>gcc -c test.c | |
| Expected Results:<br><br>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-1116: cvs project compile in sdk image** | |
|---|---|
| Summary:<br><br>cvs project could be compiled in sdk image | |
| Steps:<br><br>1. Download cvs project from http://ftp.gnu.org/non-gnu/cvs/source/feature/1.12.13/cvs-1.12.13.tar.bz2<br>2. Copy cvs tarball into sdk image<br>3. Extract the tarball and do "configure", "make" and "make install" | |
| Expected Results:<br><br>cvs project could be compiled successfully | |
| Test Execution | Weekly |

| | |
|---|---|
| Cycle Type: | |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | sdk |
| target: | 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-1117: iptables project compile in sdk image** | |
|---|---|
| Summary:<br><br>iptables project could be compiled in sdk image | |
| Steps:<br><br>1. Download iptables project from http://netfilter.org/projects/iptables/files/iptables-1.4.11.tar.bz2<br><br>2. Copy iptables tarball into sdk image<br>3. Extract the tarball and do "configure", "make" and "make install" | |
| Expected Results:<br><br>iptables could be compiled successfully | |
| Test Execution Cycle Type: | Weekly |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | sdk |
| target: | 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-1118: sudoku-savant project compile in sdk image** | |
|---|---|
| Summary:<br><br>sudoku-savant could be compiled in sdk image | |
| Steps:<br><br>1. Download sudoku-savant project from http://downloads.sourceforge.net/project/sudoku-savant/sudoku-savant/sudoku-savant-1.3/sudoku-savant-1.3.tar.bz2<br>2. Copy sudoku-savant tarball into sdk image<br>3. Extract the tarball and do "configure", "make" | |
| Expected Results:<br><br>sudoku-savant could be compiled successfully | |
| Test Execution Cycle Type: | Weekly |
| Case Automation | Manual |

| | |
|---|---|
| Type: | |
| Case State: | Ready |
| Feature: | sdk |
| target: | 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-1119: 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"<br>2. Prepare a perl program like followig test.pl<br>3. Run "perl test.pl"<br><br>########<br>$a = 9.01e+21 + 0.01 - 9.01e+21;<br>print ("the value of a is ", $a, "\n");<br><br>$a = 9.01e+21 - 9.01e+21 + 0.01;<br>print ("the value of a is ", $a, "\n");<br>######## | |
| 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-1120: shutdown system** |
|---|
| Summary: |
| verify that system can be shutdown by command |
| Steps: |
| 1. boot system<br>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-1121: reboot system** |
|---|
| Summary:<br><br>verify that system can boot by command |
| Steps:<br><br>1. boot system<br>2. launch terminal and run "reboot" |
| Expected Results:<br><br>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-1122: adjust date and time** |
|---|
| Summary:<br><br>adjust date and time |
| Steps:<br><br>1.launch terminal and run "date -R" to check current system time<br>2.adjust Date&Time by these commands:<br>For date command from coreutils, for example the sdk image use coreutils, you should use following syntax:<br>$ date -s "10:00:00 20100809"<br>$ date -R<br>$ Mon, 09 Aug 2010 10:00:00 +0000<br>For date command in busybox, for example the sato image use busybox, you should use following syntax:<br>$ date "080910002010" |

| |
|---|
| $ date -R<br>$ Mon, 09 Aug 2010 10:00:00 +0000<br>3. check date with "date -R" and the time shown on matchbox-panel |
| Expected Results:<br><br>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-1123: switch among multi applications and desktop** |
|---|
| Summary:<br><br>switch among multi applications and desktop |
| Steps:<br><br>1. launch several applications(like contacts, file manager)<br>2. launch terminal<br>3. switch among multi applications and desktop<br>4. close applications<br><br>Note: The case is for sato image only. |
| Expected Results:<br><br>1. user could switch among multi applications and desktop |

| | |
|---|---|
| Test Execution Cycle Type: | Fullpass |
| 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-1124: vncserver for target** |
|---|
| Summary:<br><br>Check if vncserver setup work in target and vnc client could connect it |
| Steps:<br><br>1. Check if x11vnc is installed in target |

| | |
|---|---|
| 2. Run command "x11vnc -display :0.0", check the ip address of the target<br>3. On a client, run command "vncviewer $ip_address_of_target:0" | |
| Expected Results:<br><br>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-1125: file manager** | |
|---|---|
| Summary:<br><br>file manager | |
| Steps:<br><br>1.launch file manager from application panel<br>2.view folder/file in file manager<br>3.copy and paste folder/file in file manager<br><br>Note: The test is only for sato image | |
| Expected Results:<br><br>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-1126: system dmesg log check** |
|---|
| Summary:<br><br>check if there is error in dmesg after system boot up |
| Steps:<br><br>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-1127: usb mount** | |
|---|---|
| Summary: | |
| verify that system can mount plugged usb automatically | |
| Steps: | |
| 1. boot system<br>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-1128: usb read files** | |
|---|---|
| Summary: | |
| verify that system can read files from usb | |
| Steps: | |
| 1. boot system<br>2. plug usb stick<br>3. view files in usb by file browser<br>4.copy some files from usb to local hardware | |
| Expected Results: | |
| 1. view/copy successfully | |
| Test Execution | Weekly |

| | |
|---|---|
| Cycle Type: | |
| 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-1129: usb umount** | |
|---|---|
| Summary:<br><br>verify that system can unmout usb automically | |
| Steps:<br><br>1. boot system<br>2. plug usb stick<br>3. view files in usb by file browser<br>4.unplug usb | |
| Expected Results:<br><br>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-1130: usb write files** | |
|---|---|
| Summary:<br><br>verify that system can write files to usb | |
| Steps:<br><br>1. boot system<br>2. plug usb stick<br>3. create files in usb<br>4.copy some files from local hardware to usb | |
| Expected Results:<br><br>1. create/copy successfully | |
| Test Execution Cycle Type: | Weekly |
| Case Automation | Manual |

| | |
|---|---|
| Type: | |
| 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-1131: file copy by scp** | |
|---|---|
| Summary:<br><br>check if file can be copied from remote machine to device by scp | |
| Steps:<br><br>1. check avahi is install and started<br>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:<br><br>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-1132: connman launch after boot** | |
|---|---|
| Summary:<br><br>After system booted, the connmand daemon should be launched | |
| Steps:<br><br>1. boot system<br>2. "ps \|grep connmand"<br>3. check if there is a thread named connmand in background | |
| Expected Results:<br><br>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 |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1133: ethernet enabled in connman** | |
|---|---|
| Summary:<br><br>After system boot, ethernet can get IP address with connman | |
| Steps:<br><br>1. boot system with network cable plugged in<br>2. "ps \|grep connmand" if connmand is started<br>3. "ifconfig" check ethernet could get IP address and ping the address from remote machine | |
| Expected Results:<br><br>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 |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1134: only one connmand in background** | |
|---|---|
| Summary:<br><br>there should be no more than one connmand in background | |
| Steps:<br><br>1. boot system<br>2. "ps \|grep connmand"<br>3. the connmand should be in background<br>4. run command "connmand"<br>5. check if the second connmand can be generated | |
| Expected Results:<br><br>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-1135: remote access by ssh** | |
|---|---|
| Summary:<br><br>check if the device can be accessed remotely by ssh | |
| Steps:<br><br>1. check avahi is install and started<br>2. get system IP and try "ssh $IP" from remote machine | |
| Expected Results:<br><br>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-1136: ethernet static ip set in connman** | |
|---|---|
| Summary:<br><br>we could set static ip for ethernet in connman | |
| Steps:<br><br>1. launch connman-properities<br><br>2. choose ethernet device and set static ip for it. For example, in our internal network, we can set as following:<br><br>ip address: 10.239.48.xxx<br><br>Broadcast: 10.239.48.255<br><br>Mask: 255.255.255.0 | |
| Expected Results:<br><br>we can set static ip for ethernet device | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | connectivity |

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

| **Test Case TC-1137: ethernet get IP in connman via DHCP** | |
|---|---|
| Summary:<br><br>ethernet device can get IP in connman via DHCP | |
| Steps:<br><br>1. Set static IP for ethernet device in connman<br>2. Check if ethernet device can work with static IP<br>3. Choose DHCP method for ethernet device<br>4. Check with ping if ethernet device get IP address via DHCP | |
| Expected Results:<br><br>Ethernet device can get dynamic IP address via DHCP in connman | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | connectivity |
| target: | e-menlow, blacksand, crownbay, sugarbay, jasperforest |
| image profile: | sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1138: connman offline mode in connman-gnome** | |
|---|---|
| Summary:<br><br>change offline mode in comman-gnome can make all connection off | |
| Steps:<br><br>1. Launch connman-properties after system booting<br><br>2. choose "offline mode" and check the connection of all network interfaces | |
| Expected Results:<br><br>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-1139: X server can start up with runlevel 5 boot** | |
|---|---|
| Summary:<br><br>check if X server can work well after system runlevel 5 booting | |
| Steps:<br><br>1. boot up system with default runlevel | |
| Expected Results:<br><br>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-1140: qt application quicky** | |
|---|---|
| Summary:<br><br>quicky is a simple note-taking application with Wiki-style syntax and behaviour | |
| Steps:<br><br>launch quicky and write something in quicky | |
| Expected Results:<br><br>http://qt-apps.org/content/show.php/Quicky?content=80325 | |
| Test Execution Cycle Type: | Weekly |
| Case Automation Type: | Manual |
| 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-1141: standby** |
|---|
| Summary: |

system can enter standby and resume from standby

Steps:

1. boot system and launch terminal; check output of "date" and launch script "continue.sh"
2. echo "mem" > /sys/power/state
3. After system go into S3 mode, move mouse or press any key to make it resume
4. Check "date" and script "continue.sh"
5. Check if application in X can work as normal

continue.sh as below:

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

i=1
while [ 0 ]
do
 echo $i
 sleep 1
 i=$((i+1))
done
##################
```

Expected Results:

screen should resume back and script can run continuously

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

**Test Case TC-1142: check CPU utilization after standby**

Summary:

check CPU utilization after standby

Steps:

1. Start up system
2. run "top" command and check if there is any process eating CPU time
3. make system into standby and resume it
4. run "top" command and check if there is any difference with the data before standby

Expected Results:

There should be no big difference before/after standby with "top"

| | |
|---|---|
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | system usage |

| | |
|---|---|
| target: | crownbay, sugarbay |
| image profile: | sato, sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1143: Test if LAN device works well after resume from suspend state** | |
|---|---|
| Summary:<br><br>Test if LAN device works well after resume from suspend state. | |
| Steps:<br><br>1. boot system and launch terminal<br>2. echo "mem" > /sys/power/state<br>3. After system go into S3 mode, move mouse or press any key to make it resume<br>4. check ping status | |
| Expected Results:<br><br>ping should always work before/after standby | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | system usage |
| target: | e-menlow, blacksand, crownbay, sugarbay, jasperforest |
| image profile: | sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1144: Test if usb hid device works well after resume from suspend state** | |
|---|---|
| Summary:<br><br>Test if usb hid device works well after resume from suspend state. | |
| Steps:<br><br>1. boot system and launch terminal<br>2. echo "mem" > /sys/power/state<br>3. After system go into S3 mode, move mouse or press any key to make it resume<br>4. check usb mouse and keyboard | |
| Expected Results:<br><br>usb mouse and keyboard should work | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | system usage |
| target: | e-menlow, blacksand, crownbay, sugarbay, jasperforest |
| image profile: | sato-sdk |
| Last Result | **Not Run** |

| | |
|---|---|
| Keywords: | None |

| **Test Case TC-1145: disk space check** | |
|---|---|
| Summary:<br><br>There should be enough disk space for QEMU rootfs | |
| Steps:<br><br>1. Launch QEMU targets(with rootfs.ext3 file)<br>2. Check the output of command df<br>3. If there is less than 5M disk space available, we assume it a failure | |
| Expected Results:<br><br>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-1146: click terminal icon on X desktop** | |
|---|---|
| Summary:<br><br>terminal icon should work without problem on X desktop | |
| Steps:<br><br>1. After system launch and X start up, click terminal icon on desktop<br>2. Check if only one terminal window launched and no other problem met | |
| Expected Results:<br><br>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-1147: Add multiple files in music player** |
|---|

| |
|---|
| Summary:<br><br>music player should be no problem when adding multiple files at same time |
| Steps:<br><br>1. Launch music player<br>2. Add multiple files(5 files) in music player at same time |
| Expected Results:<br><br>music player should be OK with this action |

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

| **Test Case TC-1148: system shutdown with UNFS** |
|---|
| Summary:<br><br>system shutdown with UNFS should work |
| Steps:<br><br>1. Use UNFS to start QEMU targets<br>2. Run shutdown in QEMU targets |
| Expected Results:<br><br>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, qemuppc, qemumips |
| image profile: | sato, sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1149: no connman-gnome icon on desktop** |
|---|
| Summary:<br><br>there should be no connman-gnome icon on desktop |
| Steps:<br><br>1. Launch sato image<br>2. There should be no connman-gnome icon on desktop, and connman-properties should be only |

| | |
|---|---|
| invoked by toolbar | |
| Expected Results:<br><br>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-1150: application contacts should work** | |
|---|---|
| Summary:<br><br>application contacts should work without problem | |
| Steps:<br><br>1. Make sure X is started up<br>2. Check if there is "contacts" icon on desktop and run it<br>3. Check if there is any error by checking the output of this action and dmesg log | |
| Expected Results:<br><br>"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-1151: x11vnc icon click for target** |
|---|
| Summary:<br><br>Check if vncserver could work in target by clicking x11vnc icon |
| Steps:<br><br>1. Check if there is a x11vnc icon in target<br>2. Click the x11vnc icon and  check the ip address of the target<br>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-1152: RTLDLIST path check for ldd command** |
|---|
| Summary:<br><br>check if the file set in RTLDLIST is valid |
| Steps:<br><br>1. After system is up, check if the RTLDLIST variable in ldd command<br>2. The file path set in RTLDLIST should be valid |
| Expected Results:<br><br>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-1153: check bash in image** |
|---|
| Summary:<br><br>check if bash exists in image |
| Steps:<br><br>1. After system is up, check if bash command exists |
| Expected Results:<br><br>bash command should exist in image |

| | |
|---|---|
| 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-1154: "Install/Remove Software" icon should work** | |
|---|---|
| Summary:<br><br>"Install/Remove Software" icon should work | |
| Steps:<br><br>1. After system is up, check if "Install/Remove Software" icon could work | |
| Expected Results:<br><br>"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-1155: gcc from ADT toolchain can build c program** |
|---|
| Summary:<br><br>gcc from ADT toolchain can build c program and run with qemu-${ARCH} command or in target image |
| Steps:<br><br>1. Install toolchain tarball and setup cross compile environment<br>2. compile following program test.c "${CC} test.c -o test -cc -lm"<br>3. run "test" with qemu-${ARCH} or run it into corresponding target image and check the output<br><br>Note: Currently, only i586_i586, x86-64_x86-64 and i586_$X(x is mips, arm and ppc) toolchain tarballs are covered in testing.<br><br>######### |

```
#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-1156: 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-1157: 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-1158: ADT toolchain could build iptables project** | |
|---|---|
| Summary:<br><br>iptables project could be compiled with ADT toolchain | |
| Steps:<br><br>1. Install toolchain tarball and setup cross compile environment<br>2. Download iptables project, http://netfilter.org/projects/iptables/files/iptables-1.4.11.tar.bz2<br>3. With the cross compile environment, run "./configure ${CONFIGURE_FLAGS}", "make", "make install DESTDIR=/opt/tmp"<br><br>Note: Currently, only i586_i586, x86-64_x86-64 and i586_$X(x is mips, arm and ppc) toolchain tarballs are covered in testing. | |
| Expected Results:<br><br>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-1159: ADT toolchain could build sudoku-savant project** | |
|---|---|
| Summary:<br><br>sudoku-savant could be compiled with ADT toolchain | |
| Steps:<br><br>1. Install toolchain tarball and setup cross compile environment<br>2. Download sudoku-savant project, http://downloads.sourceforge.net/project/sudoku-savant/sudoku-savant/sudoku-savant-1.3/sudoku-savant-1.3.tar.bz2<br>3. With the cross compile environment, run "./configure ${CONFIGURE_FLAGS}", "make", "make install DESTDIR=/opt/tmp"<br><br>Note: Currently, only i586_i586, x86-64_x86-64 and i586_$X(x is mips, arm and ppc) toolchain tarballs are covered in testing. | |
| Expected Results:<br><br>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-1160: unfs support for qemu target** | |
|---|---|
| Summary:<br><br>Check if unfs works for qemu target | |
| Steps:<br><br>1. Prepare a *rootfs.tar.bz2 image<br>2. Prepare a folder under poky directory as <rootfs-dir>, for example poky/temp<br>3. Run command "runqemu-extract-sdk *rootfs.tar.bz2 poky/temp"<br>4. Run command "runqemu nfs <kernel> <rootfs-dir>" | |
| Expected Results:<br><br>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-1161: crashme for stress** | |
|---|---|
| Summary:<br><br>Run crashme in real hardware for stress testing | |
| Steps:<br><br>1. Get crashme from http://people.delphiforums.com/gjc/crashme.html<br>2. By following the setup steps on above URL, build crashme in target.<br>3. Run crashme for 24 hours | |
| Expected Results:<br><br>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-1162: helltest for stress** | |
|---|---|
| Summary:<br><br>Run helltest for stress in target | |
| Steps:<br><br>1. helltest is stress test suite, which does compiler test for hours<br>2. We download the test suite and run it for 24 hours | |
| Expected Results:<br><br>helltest should not make target crash | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | stress |
| target: | jasperforest |
| image profile: | lsb-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1163: ltp for stress** | |
|---|---|
| Summary:<br><br>Run ltp stress in real hardware for stress testing | |
| Steps:<br><br>LTP download: http://sourceforge.net/projects/ltp/files/LTP%20Source/ltp-20101031/ltp-full-20101031.bz2/download<br>build steps: refer to http://ltp.sourceforge.net<br><br>Run steps:<br>1. Build LTP with toolchain or in sdk image<br>2. Copy LTP folder into target, for example, /opt/ltp. Modify script "testscripts/ltpstress.sh", set “lostat=1”, ”NO_NETWORK=1”<br>3. cd testscripts/ && ./ltpstress.sh<br>4. This stress case will run for 24 hours | |
| Expected Results:<br><br>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 : Power/Performance

| **Test Case TC-1164: boot time collection** | |
|---|---|
| Summary:<br><br>To collect boot time of clean installation, from grub to full desktop | |
| Steps:<br><br>1. Reboot testing device at least 3 times and do not plug anything while collecting boot time by stopwatcher:<br><br>#reboot | |
| Expected Results:<br><br>Provide average boot time and dmesg log | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | performance |
| target: | crownbay, sugarbay |
| image profile: | sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1165: memory footprint** | |
|---|---|
| Summary:<br><br>collect data of the used/free memory | |
| Steps:<br><br>With default installtion, launch terminal and type 'free' to read the used/free disk space | |
| Expected Results:<br><br>Provide 'free' output | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | core |

| | |
|---|---|
| target: | crownbay, sugarbay |
| image profile: | sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1166: powertop log** | |
|---|---|
| Summary:<br><br>collect powertop data | |
| Steps:<br><br>1. Run "powertop -d" and record output<br><br>2. Save the percentage of deepest C state(C3 or C2) | |
| Expected Results:<br><br>Provide powertop output | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | core |
| target: | crownbay, sugarbay |
| image profile: | sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1167: Idle power consumption** | |
|---|---|
| Summary:<br><br>Collect idle power consumption of target system | |
| Steps:<br><br>1. Use power meter to collect ilde power consumption of target system for 10 minutes<br><br>2. Save it and compare it with old data | |
| Expected Results:<br><br>There should be no regression between old and new ilde power data | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | performance |
| target: | crownbay, sugarbay |
| image profile: | sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| Test Case TC-1168: core build time for sato image | |
|---|---|
| Summary:<br><br>collect the core build time for sato qemux86 image | |
| Steps:<br><br>1. Perpare a system with following configuration<br>CPU: 4-core * 2-threads Intel(R) Core(TM) i7 CPU 860 @ 2.80GHz<br>Memory: 4GB<br>Harddisk: 1TB<br><br>OS: Ubuntu 10.04 x86_64<br>Kernel: 2.6.32-21<br><br>2. Download poky tree and make sure all the source packages have been downloaded<br>3. Build a qemux86 sato image and collect the time | |
| Expected Results:<br><br>There should be no regression for build time | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | performance |
| target: | qemux86_32 |
| image profile: | sato |
| Last Result | **Not Run** |
| Keywords: | None |

## 1.6 Test Suite : Graphics

| Test Case TC-1169: Graphics ABAT |
|---|
| Summary:<br><br>Yocto on SugarBay should pass Intel graphics ABAT testing |
| Steps:<br><br>1. Download ABAT test suite from internal git repository,  git clone<br>git://tinderbox.sh.intel.com/git/abat<br>2. Apply following patch to make it work on yocto environment<br>3. Run "./abat.sh" to run ABAT test<br><br>########<br>diff --git a/glxgears_check.sh b/glxgears_check.sh<br>index 17622b8..c4d3b97 100755<br>--- a/glxgears_check.sh<br>+++ b/glxgears_check.sh<br>@@ -31,7 +31,7 @@ else<br><br>    sleep 6 |

```
-    XPID=$( ps ax | awk '{print $1, $5}' | grep glxgears | awk '{print $1}')
+     XPID=$( ps | awk '{print $1, $5}' | grep glxgears | awk '{print $1}')
     if [ ! -z "$XPID" ]; then
         kill -9 $XPID >/dev/null 2>&1
         echo "glxgears can run, PASS!"
diff --git a/x_close.sh b/x_close.sh
index e287be1..3429f1a 100755
--- a/x_close.sh
+++ b/x_close.sh
@@ -22,7 +22,7 @@
 #
 function close_proc(){
 echo "kill process Xorg"
-XPID=$(  ps ax | awk '{print $1, $5}' | egrep "X$|Xorg$" | awk '{print $1}')
+XPID=$(  ps | awk '{print $1, $6}' | egrep "X$|Xorg$" | awk '{print $1}')
 if [ ! -z "$XPID" ]; then
     kill $XPID
     sleep 4
diff --git a/x_start.sh b/x_start.sh
index 9cf6eab..2305796 100755
--- a/x_start.sh
+++ b/x_start.sh
@@ -24,7 +24,7 @@
 X_ERROR=0

 #test whether X has started
-PXID=$(ps ax |awk '{print $1,$5}' |egrep "Xorg$|X$" |grep -v grep | awk '{print $1}')
+PXID=$(ps |awk '{print $1,$6}' |egrep "Xorg$|X$" |grep -v grep | awk '{print $1}')
 if [ ! -z "$PXID" ]; then
     echo "[WARNING] Xorg has started!"
     XORG_STATUS="started"
@@ -35,9 +35,11 @@ else
     #start up the x server
     echo "Start up the X server for test in display $DISPLAY....."

-    $XORG_DIR/bin/X >/dev/null 2>&1 &
+    #$XORG_DIR/bin/X >/dev/null 2>&1 &
+    #sleep 8
+    #xterm &
+    /etc/init.d/xserver-nodm start &
     sleep 8
-    xterm &
 fi
     XLOG_FILE=/var/log/Xorg.0.log
     [ -f $XORG_DIR/var/log/Xorg.0.log ] && XLOG_FILE=$XORG_DIR/var/log/Xorg.0.log
@@ -54,7 +56,7 @@ fi
         X_ERROR=1
     fi

-    XPID=$( ps ax | awk '{print $1, $5}' | egrep "X$|Xorg$" |grep -v grep| awk '{print $1}')
+    XPID=$( ps | awk '{print $1, $6}' | egrep "X$|Xorg$" |grep -v grep| awk '{print $1}')
     if [ -z "$XPID" ]; then
         echo "Start up X server FAIL!"
  echo
########
```

Expected Results:

All ABAT test should pass

| | |
|---|---|
| Test Execution Cycle Type: | Weekly |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | bsp |
| target: | e-menlow, blacksand, crownbay, sugarbay |
| image profile: | sato, sato-sdk |

| | |
|---|---|
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1170: openarena - 3D** | |
|---|---|
| Summary:<br><br>Run opernarena testing and compare the result with upstream graphics result | |
| Steps:<br><br>1. Download and build openarena through phoronix test suite. first download a new phoronix from its website, then download the game in it. The openarena we use is v0.8.5.<br>####<br>phoronix-test-suite list-tests<br>phoronix-test-suite install openarena<br>####<br>2. Run the test suite with following command<br>####<br>vblank_mode=0 openarena +exec pts +set r_mode -1 +set r_fullscreen 1 +set r_customWidth $VIDEO_WIDTH +set r_customHeight $VIDEO_HEIGHT<br>####<br><br>The VIDEO_WIDTH and VIDEO_HEIGHT set the game's resolution，you can get current resolution by command "xrandr" | |
| Expected Results:<br><br>Compare the result of Yocto with upstream graphics | |
| Test Execution Cycle Type: | Weekly |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | bsp |
| target: | sugarbay |
| image profile: | sato, sato-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1171: urbanterror - 3D** |
|---|
| Summary:<br><br>Run urbanterror and compare the result of Yocto with upstream graphics |
| Steps:<br><br>1. download and build: This game also can get through phoronix-test-suite. 2. we should set some environments as following before test:<br>###<br>OS_TYPE=Linux<br>OS_ARCH=`uname -i`<br>LOG_FILE=${LOGNOW_DIR}/${LOG_FILE}<br>###<br>3. Run urbanterror with following command<br>###<br>vblank_mode=0 ./urbanterror +timedemo 1 +set demodone 'quit' +set demoloop1 'demo pts1; set nextdemo vstr demodone' +vstr demoloop1 +set r_customwidth $VIDEO_WIDTH +set r_customheight $VIDEO_HEIGHT |

| |
|---|
| ### |
| Expected Results:<br><br>Get the FPS data of Yocto and compare it with upstream graphics |

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

| **Test Case TC-1172: x11perf - 2D** |
|---|
| Summary:<br><br>Get fps data of x11per running |
| Steps:<br><br>1. Run "x11perf -aa10text" and "x11perf -rgb10text"<br>2. Get the FPS result and compare it with upstream graphics data on Sandybridge |
| Expected Results:<br><br>There should not be big regression between Yocto and upstream linux |

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

## 1.7 Test Suite : Mulitimedia

| **Test Case TC-1173: sound on/off** |
|---|
| Summary:<br><br>check if sound can be turned on/off |
| Steps:<br><br>1. copy amixer is installed |

| | |
|---|---|
| 2. Run "amixer set Master on" to turn on audio device<br>3. Run "amixer set Master 64" to adjust to maxium volumn<br>4. Run "amixer set Speaker on" to turn on speaker<br>5. Run "amixer set Speaker 64" to adjust to maxium volumn<br>6. Run "amixer set Master off" to turn off audio device<br>7. Run "amixer set Speaker off" to turn off speaker | |
| Expected Results:<br><br>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-1174: audio play (mp3)** | |
|---|---|
| Summary:<br><br>make sure music player cannot play mp3 format file | |
| Steps:<br><br>1. copy sample mp3 file to system<br>2. launch music player and make sure it cannot play the mp3 file | |
| Expected Results:<br><br>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-1175: audio play (ogg)** |
|---|
| Summary:<br><br>check if music player can play ogg format file |
| Steps:<br><br>1. copy sample ogg file to system<br>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-1176: audio stop (ogg)** | |
|---|---|
| Summary:<br><br>check if music player can play ogg format file | |
| Steps:<br><br>1. copy sample ogg file to system<br>2. launch music player can play the ogg file<br>3. click "stop" button to stop playing<br>4. click "start" button to resume playing | |
| Expected Results:<br><br>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-1177: audio play (wav)** | |
|---|---|
| Summary:<br><br>check if music player can play wav format file | |
| Steps:<br><br>1. copy sample wav file to system<br>2. launch music player can play the wav file | |
| Expected Results:<br><br>wav file can be played 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-1178: audio stop (wav)** | |
|---|---|
| Summary:<br><br>check if music player can stop playing with wav format file | |
| Steps:<br><br>1. copy sample wav file to system<br>2. launch music player can play the wav file<br>3. click "stop" button to stop playing<br>4. click "start" button to resume playing | |
| Expected Results:<br><br>wav 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-1179: video play (mpeg)** | |
|---|---|
| Summary:<br><br>make sure video player cannot play mpeg format file | |
| Steps:<br><br>1. copy sample mpeg file to system<br>2. launch video player and make sure it cannot play the mpeg file | |
| Expected Results:<br><br>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-1180: video play (ogg)** | |
|---|---|
| Summary:<br><br>check if video player can play ogg format file | |
| Steps:<br><br>1. copy sample ogg file to system<br>2. launch video player can play the ogg file | |
| Expected Results:<br><br>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-1181: video stop (ogg)** | |
|---|---|
| Summary:<br><br>check if video player can play ogg format file | |
| Steps:<br><br>1. copy sample ogg file to system<br>2. launch video player can play the ogg file<br>3. click "stop" button to stop playing<br>4. click "start" button to resume playing | |
| Expected Results:<br><br>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 |

## 1.8 Test Suite : Compliance

| **Test Case TC-1182: LTP subset test suite** | |
|---|---|
| Summary:<br><br>LTP subset test suite | |
| Steps:<br><br>For real hardware, run following component,<br>syscalls<br>fs<br>fsx<br>dio<br>io<br>mm<br>ipc<br>sched<br>math<br>nptl<br>pty<br>admin_tools<br>timers<br>commands<br><br>For QEMU, run following component<br>syscalls<br>mm<br>ipc<br>sched<br>math<br>nptl<br>pty<br>admin_tools<br>commands<br><br>Run Instructions:<br>LTP download: http://sourceforge.net/projects/ltp/files/LTP%20Source/ltp-20110606/ltp-full-20110606.bz2/download<br><br>build steps: refer to http://ltp.sourceforge.net<br><br>Run steps:<br>1. Build LTP with toolchain or in sdk image<br>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.<br>3. Copy LTP folder into target, for example, /opt/ltp. Modify script "runltp", remove test suites not to be tested<br>4. Comment runtests/sched: hackbench, which is not suitable to run in emulators<br>5. Prepare a tmp folder under your ltp folder, for example, create a tmp folder under your ltp folder, like /opt/ltp/tmp<br>6. ./runltp -p -l result-M2-20101218.log -C result-M2-20101218.fail -d /opt/ltp/tmp &> result-M2-20101218.fulllog<br>(assume you mount your LTP disk at /opt and create your own tmp dir at /opt/ltp/tmp) | |
| Expected Results:<br><br>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 | Semi-Auto |

| | |
|---|---|
| Type: | |
| 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-1183: POSIX subset test suite** | |
|---|---|
| Summary:<br><br>Run subset test suite of POSIX test suite | |
| Steps:<br><br>POSIX test suite download: http://sourceforge.net/projects/posixtest/files/posixtest/posixtestsuite-1.5.2/posixtestsuite-1.5.2.tar.gz/download<br>build: refer to http://posixtest.sourceforge.net/<br><br>Run steps:<br>1. Get POSIX test suite as above<br>2. Start target and copy test suite into it<br>3. For qemu, option "-m 512" should be added<br>4. Make sure below is uncommented from LDFLAGS file:<br>#-D_XOPEN_SOURCE=600 –lpthread –lrt –lm<br><br>5. For gcc 4.6, you need to add "-Wno-unused-but-set-variable -Wno-address" to CFLAGS in Makefile<br>6. Run following commands under POSIX test suite<br>run_tests SIG<br>run_tests SEM<br>run_tests THR<br>run_tests TMR<br>run_tests MSG<br>run_tests TPS<br>run_tests MEM | |
| Expected Results:<br><br>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 |

| **Test Case TC-1184: LSB subset test suite** |
|---|
| Summary: |

| |
|---|
| Run LSB subset test suite in target |
| Steps:<br><br>1. Get LSB image and start the image(if it is QEMU) with option "-m 512M"<br>2. Get the LSB test suite or run script creat-lsb-image under poky source directory "scripts/creat-lsb-image"<br>3. Setup environment for lsb image in target with script LSB_Setup.sh, it could be found under poky source directory "/meta/recipes-extended/lsb/lsbsetup/LSB_Setup.sh"<br>4. Select LSB test items in LSB web interface and run them |
| Expected Results:<br><br>Check the result on wiki, https://wiki.pokylinux.org/wiki/index.php?title=LSB_result&action=edit&redlink=1. No regression failures should be met. |

| | |
|---|---|
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | core |
| target: | blacksand, mpc8315e-rdb, sugarbay |
| image profile: | lsb-sdk |
| Last Result | **Not Run** |
| Keywords: | None |

### 1.9 Test Suite : Core Build System

| **Test Case TC-1185: Init scripts** |
|---|
| Summary:<br><br>Provide an image/recipe skeleton as a canonical example. Check if can be built and run correctly |
| Steps:<br>1. Build image from poky source, check if skeleton script and skeleton-test can be built into the image<br>a. download poky source<br>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)<br>c. $ source oe-init-build-env<br>add line “<POKY_BASE>/meta-skeleton \” to conf/bblayer.conf<br>d. build the image<br>e. boot up the image, check the skeleton and skeleton-test should be |

| |
|---|
| in right place<br>/etc/init.d/skeleton<br>/usr/sbin/skeleton-test<br><br>2. Verify the basic function of skeleton. Check if skeleton script can start/stop the skeleton-test daemon. |
| Expected Results:<br><br>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-1186: Minimal image** | |
|---|---|
| Summary:<br><br>Check if the minimal image can be built and run correctly. | |
| Steps:<br><br>1. Build a minimal image from poky source by following the wiki: https://wiki.yoctoproject.org/wiki/Minimal_Image<br>2. Check the size of the image. It should take less than 5M disk space after extraction.<br>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:<br><br>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-1187: Share gcc work directories** |
|---|
| Summary:<br><br>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-1188: 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:<br><br>The ccache-native should be built successfully and be installed to the correct location.<br>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-1189: PAM support** | |
|---|---|
| Summary:<br><br>Check the Yocto should support PAM (Pluggable Authentication Module) | |
| Steps:<br><br>1. Build a sato-sdk image from poky source with PAM support by following the wiki: https://wiki.yoctoproject.org/wiki/PAM_Integration<br>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:<br><br>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-1190: kernel interactive targets** |
|---|
| Summary:<br><br>Check if yocto can support kernel interactive target build |
| Steps:<br><br>1. download yocto source tree<br>2. prepare yocto build environment<br>3. Run "bitbake linux-yocto -c menuconfig"<br>4. Check if a new bash terminal pop up and menuconfig can be triggered |
| Expected Results:<br><br>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-1191: KVM enabled with qemu** | |
|---|---|
| Summary:<br><br>qemu can be started with KVM enabled | |
| Steps:<br><br>1. build a kernel with KVM enabled<br>2. Start qemu with option "kvm" with runqemu<br>3. Check if qemu starts up and if kvm_intel is used<br>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:<br><br>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-1192: non-GPLv3 build check** |
|---|
| Summary:<br><br>Check if non-GPLv3 build could pass and it does not has any GPLv3 packages installed |
| Steps:<br><br>1. Set following sentences in local.conf to GPLv3<br>#####<br>INCOMPATIBLE_LICENSE = "GPLv3"<br>#####<br>2. Build core-image-minimal and core-image-basic<br>3. Start up target after build is finished<br>4. Run following script to check if any GPLv3 packages installed<br><br>###################<br>#!/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-1193: 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-1194: yocto build in OpenSuse 11.4** | |
|---|---|
| Summary:<br><br>Build latest yocto in x86_64 OpenSuse 11.4 | |
| Steps:<br><br>1. By following the yocto handbook, download latest yocto source<br>2. Build core-image-minimal on OpenSuse 11.4 | |
| Expected Results:<br><br>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-1195: yocto build in Ubuntu 11.04** | |
|---|---|
| Summary:<br><br>Build latest yocto in x86_64 Ubuntu 11.04 | |
| Steps:<br><br>1. By following the yocto handbook, download latest yocto source<br>2. Build core-image-minimal on Utuntu 11.04 | |
| Expected Results:<br><br>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-1196: yocto build in KVM** |
|---|
| Summary:<br><br>Build yocto in KVM should work |
| Steps:<br><br>1. Setup a VM environment with KVM enabled, for example, RHEL6 |

| | |
|---|---|
| 2. Prepare a VM for yocto build testing, for example, OpenSuse 11.3<br>3. By following the yocto handbook, download latest yocto source into the VM<br>4. Build core-image-minimal in the VM | |
| Expected Results:<br><br>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-1197: sstate work on local host** | |
|---|---|
| Summary:<br><br>Check if sstate could work with local cache | |
| Steps:<br><br>1. Follow the wiki steps to setup a sstate cache on local machine, https://wiki.yoctoproject.org/wiki/Enable_sstate_cache<br>2. Prepare another yocto source directory and set the SSTATE_DIR the cache you setup in step 1)<br>3. Run poky build, for example, "bitbake core-image-minimal". You should note following things if sstate works:<br><br>########<br>NOTE: Preparing runqueue<br>NOTE: Executing SetScene Tasks<br>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)<br>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<br>######## | |
| Expected Results:<br><br>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-1198: 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:<br><br>1. Download poky source and prepare the build environment<br>2. Set GCCVERSION and SDKGCCVERSION to 4.5.1 in meta/conf/distro/include/tcmode-default.inc<br>3. Run "bitbake -s \| grep gcc" and check the output, all gcc related options should be set to 4.5.1 |
| Expected Results:<br><br>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-1199: btrfs format image build** |
|---|
| Summary:<br><br>btrfs format image could be built out |
| Steps:<br><br>1. set IMAGE_FSTYPES = "btrfs" in local.conf<br>2. build a core-image-minimal image, the image should be btrfs format |
| Expected Results:<br><br>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-1221: btrfs format image boot up** |
|---|
| Summary:<br><br>btrfs format image could be booted up |
| Steps:<br><br>1. set IMAGE_FSTYPES = "btrfs" in local.conf<br>2. build a qemux86 core-image-minimal image and boot up it |

| | |
|---|---|
| Expected Results:<br><br>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-1200: lib64-zlib lib32-zlib build** | |
|---|---|
| Summary:<br><br>lib64-zlib lib32-zlib build should pass with multilib enabled | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build<br>3. build lib64-zlib and lib32-zlib, which should build pass without error | |
| Expected Results:<br><br>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-1201: lib32 sato image build - qemux86** | |
|---|---|
| Summary:<br><br>lib32 sato image could be built out with multilib support | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86<br>3. with rpm set for package format, build lib32 core-sato image<br>4. after build finished, start up the image and check if all app are 32-bit, kernel with 32-bit | |
| Expected Results:<br><br>lib32 sato image could be built out with multilib support | |
| Test Execution Cycle | Fullpass |

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

| **Test Case TC-1214: lib32 sato image build - qemux86-64** | |
|---|---|
| Summary:<br><br>lib32 sato image could be built out with multilib support | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86<br>3. with rpm set for package format, build lib32 core-sato image<br>4. after build finished, start up the image and check if all app are 32-bit, kernel with 64-bit | |
| Expected Results:<br><br>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-1203: lib64 sato image build - qemux86** | |
|---|---|
| Summary:<br><br>lib64 sato image should be built out with multilib support | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86<br>3. with rpm set for package format, build lib64 core-sato image<br>4. after build finished, start up the image and check if all app are 64-bit, kernel with 32-bit | |
| Expected Results:<br><br>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-1215: lib64 sato image build - qemux86-64** | |
|---|---|
| Summary:<br><br>lib64 sato image should be built out with multilib support | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86<br>3. with rpm set for package format, build lib64 core-sato image<br>4. after build finished, start up the image and check if all app are 64-bit, kernel with 64-bit | |
| Expected Results:<br><br>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-1219: lib64 sato image build - qemux86-64/ipk** | |
|---|---|
| Summary:<br><br>lib64 sato image should be built out with multilib support | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86<br>3. with ipk set for package format, build lib64 core-sato image<br>4. after build finished, start up the image and check if all app are 64-bit, kernel with 64-bit | |
| Expected Results:<br><br>lib64 sato-sdk image should be built out with multilib support | |
| 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-1220: lib64 sato image build - qemux86-64/deb** | |
|---|---|
| Summary:<br><br>lib64 sato image should be built out with multilib support | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86<br>3. with deb set for package format, build lib64 core-sato image<br>4. after build finished, start up the image and check if all app are 64-bit, kernel with 64-bit | |
| Expected Results:<br><br>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-1216: lib32 connman-gnome built for qemux86-64 - rpm** | |
|---|---|
| Summary:<br><br>build lib32 connman-gnome and include it in qemux86-64 image | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86-64<br>3. set "MULTILIB_IMAGE_INSTALL = "lib32-connman-gnome""<br>4. with rpm set for package format, build core-sato image<br>5. after build finished, start up the image and check if connman and related packages are 32-bit | |
| Expected Results:<br><br>user could build lib32 connman-gnome and include it in qemux86-64 image | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation | Manual |

| | |
|---|---|
| Type: | |
| Case State: | Ready |
| Feature: | core |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1217: lib32 connman-gnome built for qemux86-64 - ipk** | |
|---|---|
| Summary:<br><br>build lib32 connman-gnome and include it in qemux86-64 image | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86-64<br>3. set "MULTILIB_IMAGE_INSTALL = "lib32-connman-gnome""<br>4. with ipk set for package format, build core-sato image<br>5. after build finished, start up the image and check if connman and related packages are 32-bit | |
| Expected Results:<br><br>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-1218: lib32 connman-gnome built for qemux86-64 - deb** | |
|---|---|
| Summary:<br><br>build lib32 connman-gnome and include it in qemux86-64 image | |
| Steps:<br><br>1. Prepare poky build environment<br>2. by following https://wiki.pokylinux.org/wiki/Multilib, set local.conf to enable multilib build and set MACHINE to qemux86-64<br>3. set "MULTILIB_IMAGE_INSTALL = "lib32-connman-gnome""<br>4. with deb set for package format, build core-sato image<br>5. after build finished, start up the image and check if connman and related packages are 32-bit | |
| Expected Results:<br><br>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-1206: bitbake-layers show_layers** | |
|---|---|
| Summary:<br><br>show_layers could show current layers | |
| Steps:<br><br>1. prepare poky build environment<br>2. add meta-rt into bblayer.conf<br>3. run "bitbake-layers show_layers", it should show the layers defined in bblayer.conf | |
| Expected Results:<br><br>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-1207: bitbake-layers show_overlayed** | |
|---|---|
| Summary:<br><br>overlayed recipes should be shown with bitbake-layers | |
| Steps:<br><br>1. prepare poky build environment<br>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<br>3. run "bitbake-layers show_overlayed", it should report clutter is overlayed by meta-yocto | |
| Expected Results:<br><br>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-1208: bitbake-layers show_appends** | |
|---|---|
| Summary:<br><br>bitbake-layers show_appends should list bbappend files and recipe files they apply to | |
| Steps:<br><br>1. prepare poky build environment<br>2. run "bitbake-layers show_appends", it should list bbappend files and recipe files they apply to | |
| Expected Results:<br><br>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-1209: bitbake-layers flatten** | |
|---|---|
| Summary:<br><br>bitbake-layers flattens layer configuration into a separate output directory | |
| Steps:<br><br>1. prepare poky build environment<br>2. create a folder, for example, test<br>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<br>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:<br><br>bitbake-layers flattens layer configuration into a separate output directory | |
| 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-1222: x32 image build** | |
|---|---|
| Summary:<br><br>x32 image could be built out successfully | |
| Steps:<br><br>1. Prepare yocto build environment<br>2. add meta-x32 layer, http://git.yoctoproject.org/cgit/cgit.cgi/experimental/meta-x32/<br>3. Add following lines in your conf/local.conf<br>MACHINE = "qemux86-64"<br>DEFAULTTUNE = "x86-64-x32"<br>TUNE_PKGARCH = "x86-64-x32" | |
| Expected Results:<br><br>x32 image could be built out successfully | |
| Test Execution Cycle Type: | Fullpass |
| Case Automation Type: | Manual |
| Case State: | Ready |
| Feature: | core |
| Last Result | **Not Run** |
| Keywords: | None |

| **Test Case TC-1223: x32 image build boot up and check** | |
|---|---|
| Summary:<br><br>x32 image could be built out successfully and binaries/libraries are x32 in it | |
| Steps:<br><br>1. Prepare yocto build environment<br>2. add meta-x32 layer, http://git.yoctoproject.org/cgit/cgit.cgi/experimental/meta-x32/<br>3. Add following lines in your conf/local.conf<br>MACHINE = "qemux86-64"<br>DEFAULTTUNE = "x86-64-x32"<br>TUNE_PKGARCH = "x86-64-x32"<br>4. build minimal image with "bitbake core-image-minimal"<br>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:<br>$ file bin/busybox<br>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<br><br>$file usr/lib/libz.so.1.2.5<br>usr/lib/libz.so.1.2.5: ELF 32-bit LSB shared object, x86-64, version 1 (SYSV), dynamically linked, not stripped | |
| Expected Results:<br><br>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 |

### 1.10 Test Suite : BSP specific

| **Test Case TC-1210: RTC** | |
|---|---|
| Summary:<br><br>Check if RTC(Real Time Clock) can work correctly | |
| Steps:<br><br>1. Read time from RTC registers.<br><br>root@localhost:/root> hwclock -r<br><br>Sun Mar 22 04:05:47 1970 -0.001948 seconds<br><br>2. Set system current time<br><br>root@localhost:/root> date 062309452008<br><br>3. Synchronize the system current time to RTC registers<br><br>root@localhost:/root> hwclock -w<br><br>4. Read time from RTC registers<br><br>root@localhost:/root> hwclock -r<br><br>5. Reboot target and read time from RTC again. | |
| Expected Results:<br><br>Can read and set the time successful | |
| Test Execution Cycle Type: | Weekly |
| 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-1211: Watchdog | |
|---|---|
| Summary:<br><br>Check if watchdog can reset the target system | |
| Steps:<br><br>1. Check if watchdog device exist in /dev/ directory<br><br>2. Run command “echo 1 > /dev/watchdog” and wait for 60s. Then the target will reboot. | |
| Expected Results:<br><br>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-1212: SATA |
|---|
| Summary:<br><br>Test general use of SATA device on target, like mount, umount, read and write. |
| Steps:<br><br>1. Run “fdisk” command to create partition on SATA disk.<br><br>2. Mount/Umount<br><br>mke2fs /dev/sda1<br><br>mount -t ext2 /dev/sda1 /mnt/disk<br><br>umount /mnt/disk<br><br>3. Read/Write (filesystem)<br><br>touch /mnt/disk/test.txt<br><br>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-1213: 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 |