

OPEN VIRTUALIZATION BUILD AND BOOT GUIDE

Version 2.1



Contents

1	Introduction.....	3
2	Steps to Build Open Virtualization SDK:	3
2.1	Prerequisites.....	3
2.2	Steps to build the linux kernel:	3
2.3	Steps to build the secure kernel :	4
2.4	Steps for installing openssl package:	5
2.5	Steps to follow to resolve the toolchain error in Ubuntu 12.x	6
3	Steps to boot Open Virtualization in Fastmodel.....	7
3.1	Install the Fastmodel.....	7
3.2	Starting the Fastmodel	8
3.3	Load project and debugging.....	9
3.4	Booting the image and debugging using Fastmodel.....	12
3.5	Testing the Open-virtualization Stack.....	16

1 Introduction

The TEE demo showcases features of SierraTEE like Global Platform Communication APIs, ability to run normal world Linux guest etc. using Cortex A15 Fast Models Platform. This guide provides detailed step by step installation instructions.

2 Steps to Build Open Virtualization SDK:

2.1 Prerequisites

- Linux-2.6.38.7.tar.bz2
- Codesourcery toolchain
- Newlib toolchain is available as toolchain.tar.bz2 arm-sw-newlib

2.2 Steps to build the linux kernel:

- i. Download linux-2.6.38.7 from linux kernel source.
- ii. Copy the downloaded file to trustzone/otz_linux as linux-2.6.38.7.tar.bz2
- iii. Copy the image
`#<FastModelInstalldir>/FastModelsPortfolio_8.3/images/RTSM_EB_Linux/files
systems/armv5t_min_EB_V6_V7.image` to trustzone/otz_linux
- iv. Export the CROSS_COMPILE variable to point to the codesourcery toolchain.
`#export CROSS_COMPILE=<toolchain installed directory>/bin/arm-none- linux-gnueabi-`
- v. Run make command from otz_linux path.
`#make`

2.3 Steps to build the secure kernel :

- i. Export the CROSS_COMPILE_NEWLIB variable to point to the supplied toolchain in case of building crypto application. In case of building *crypto application*,

#export CROSS_COMPILE_NEWLIB=<supplied toolchain installed directory>/bin/arm-none-eabi-

Otherwise,

#export CROSS_COMPILE_NEWLIB=<toolchain installed directory>/bin/arm-none-linux-gnueabi

- ii. Run make command from SDK path.
- iii. Binaries and library files are available in ***tzone_sdk/bin*** and
 - a. ***tzone_sdk/lib*** respectively.
- iv. Linux trustzone api client applications (***otzapp.elf***)(***otz_tee_app.elf***), Linux trustzone client driver (***otz_client.ko***) and trustzone api shared library (***libtzapi.so***) are copied to the root file system.

2.4 Steps for installing openssl package:

1. Experimental support for openssl has been added. Since this is experimental, a separate executable file **otzone-crypto.elf** would be created.

To enable openssl support, the following changes need to be made.

- (i) Download openssl-1.0.1c from

<http://www.openssl.org/source/openssl-1.0.1c.tar.gz> and place it in */trustzone/package/storage* folder.

- (ii) Set ENABLE_LIBCRYPT to "y" CRYPTO_BUILD to "y" in ***tzone_sdk/Makefile***.

- (iii) Set the value of the variable CONFIG_CRYPT to "y" in the file *config.package*, found under the directory *package*.

- (iv) The variable CROSS_COMPILE_NEWLIB should be made to point to the appropriate path.

#export CROSS_COMPILE_NEWLIB=<supplied toolchain installed directory>/bin/arm-none-eabi-

2. Load the image **otzone-crypto.elf**, which would be available in the bin directory.

2. The app can be tested by launching *otzapp.elf* as discussed below.

#otzapp.elf

2.5 Steps to follow to resolve the toolchain error in Ubuntu 12.x

While building the secure kernel with crypto enabled in ubuntu 12.04, the tool chain might give some errors due to the missing of the required libraries. Run the following in order to fix this issue.

```
apt-get install libmpc2:i386
apt-get install libmpfr4:i386
ln -s /usr/lib/i386-linux-gnu/libmpfr.so.4 /usr/lib/i386-linux- gnu/libmpfr.so.1
apt-get install libgmp10:i386
ln -s /usr/lib/i386-linux-gnu/libgmp.so.10 /usr/lib/i386-linux- gnu/libgmp.so.3
apt-get install libelf1:i386
ln -s /usr/lib/i386-linux-gnu/libelf.so.1 /usr/lib/i386-linux- gnu/libelf.so.0
```

Note: Supported toolchain version is arm-2010q1.

3 Steps to boot Open Virtualization in Fastmodel

3.1 Install the Fastmodel

An evaluation version is available from the following link. A registration with ARM is necessary to be able to proceed with the download of the software.

<http://www.arm.com/products/tools/models/fast-models.php>. Click the Download now button and proceed to find the Fast Models Evaluation Linux. Download the latest Fast model version 8.3. When downloading the software and license file for a 45 day evaluation. Unpack the software and read the installation instructions in the Installation_Guide.txt.

Before proceeding to the fast model installation, install the standard development tools by executing the follow command

```
#sudo apt-get install ia32-libs gcc-multilib
```

```
# sudo apt-get install build-essential xutils xutils-dev
```

Now run ./setup.bin and choose an installation directory and the location of the license file and edit the current user's .bashrc file and add a line

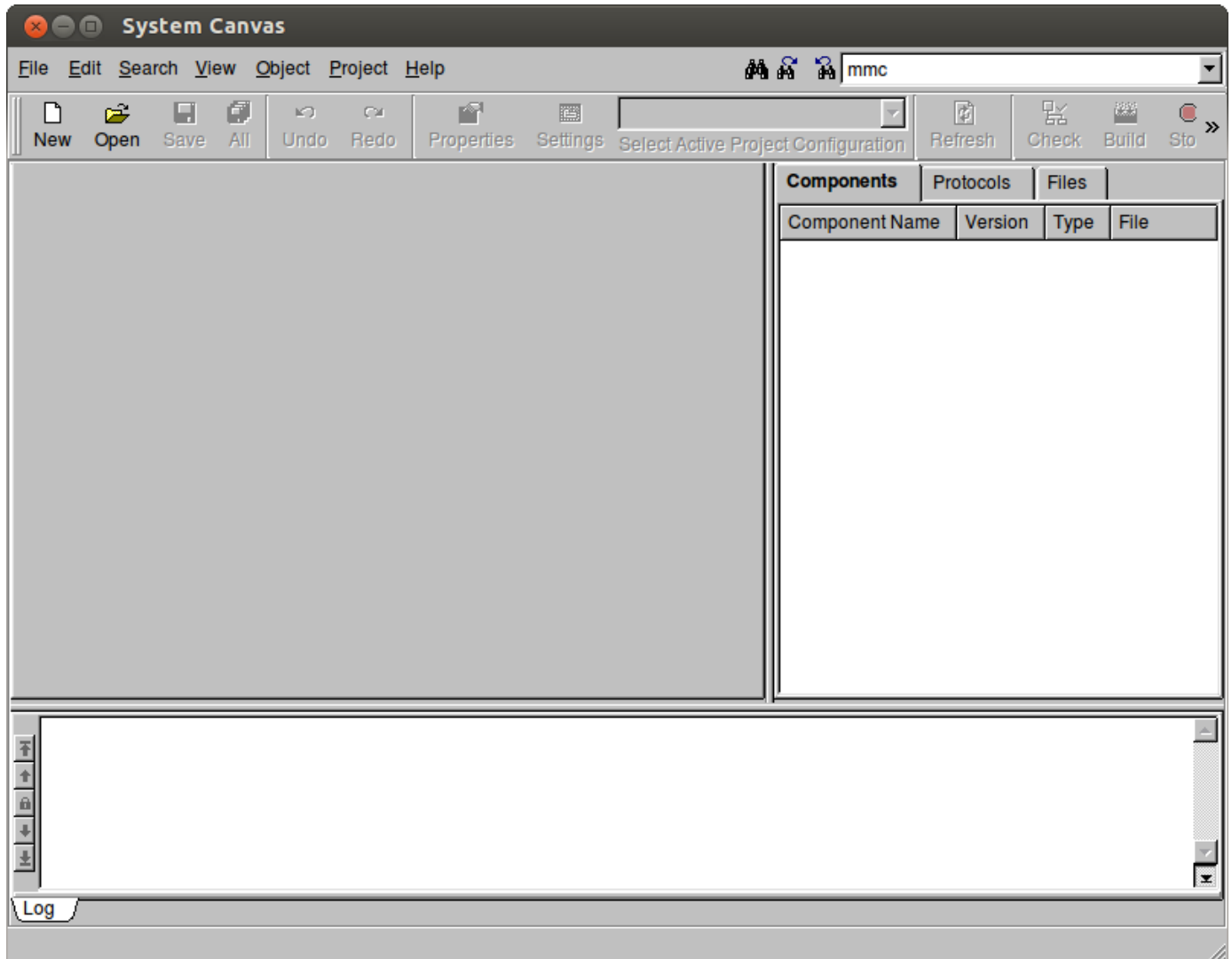
```
#source <FastModelInstalled-Dir>/FastModelTools_8.3/source_all.sh
```

3.2 Starting the Fastmodel

Run the following commands in order to launch the fast model.

```
#sgcanvas
```

The application gets started and the following screen appears.

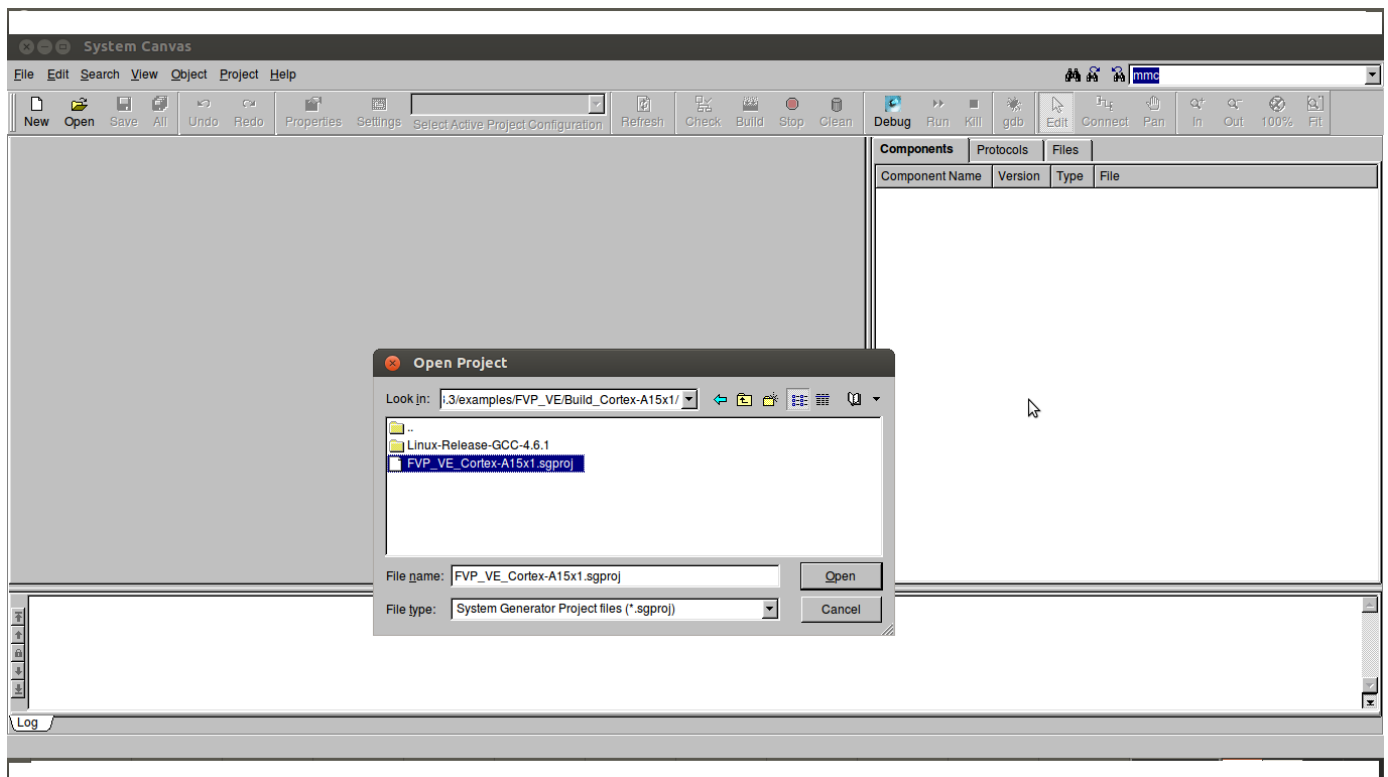


3.3 Load project and debugging

Load the required project by selecting File -> Load Project

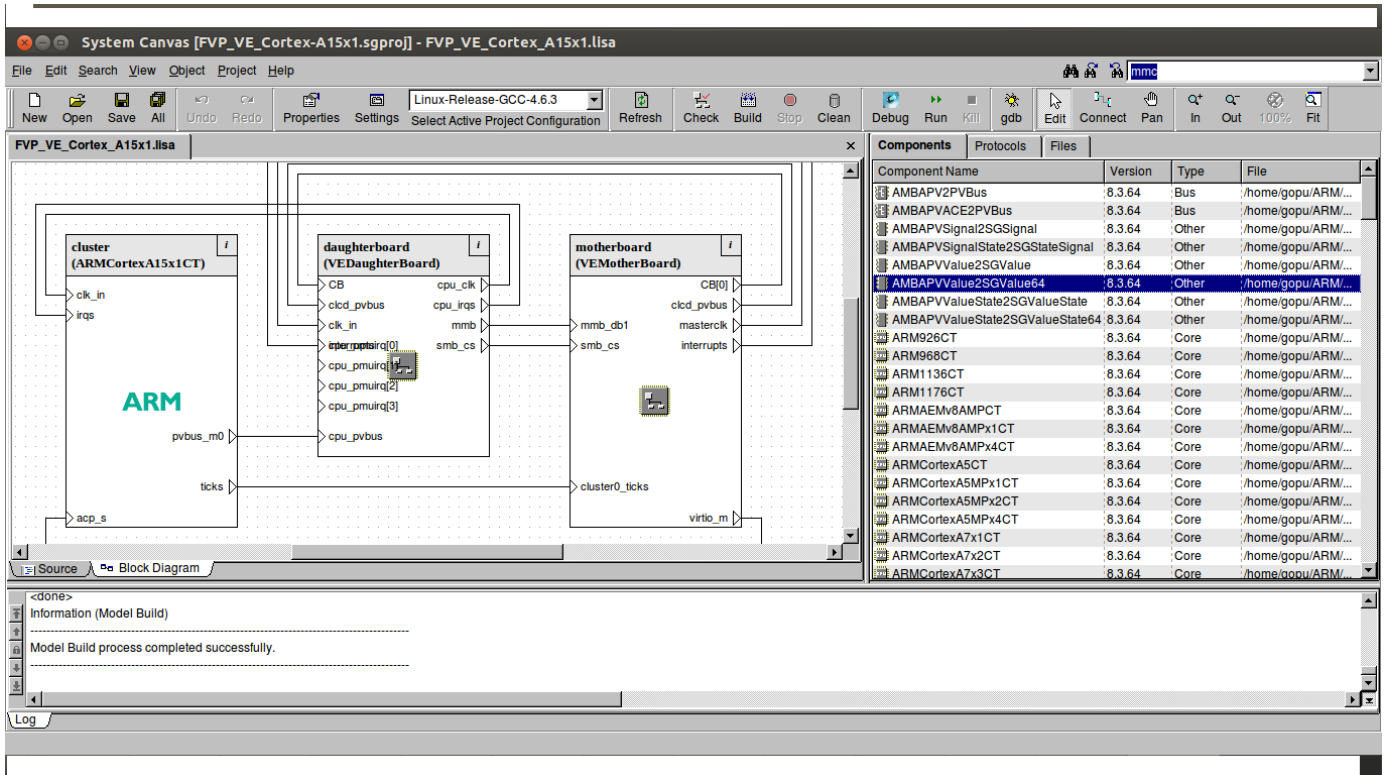
For example, Project for Real view Eval board with cortex A15 is available in the following location.

*<FastModel install dir>/FastModelsPortfolio_8.3/examples/FVP_VE/Build_Cortex-A15x1/
FVP_VE_Cortex-A15x1.sgproj*



Example projects are available in the Fastmodel installed directory.

The following screen appears on loading the project.



On trying to build (clicking Build option) the project with GCC 4.7.2, few .so files are not created which results in the Model Debugger to throw an error similar to the following.

Cannot load model library '<Fastmodel-install

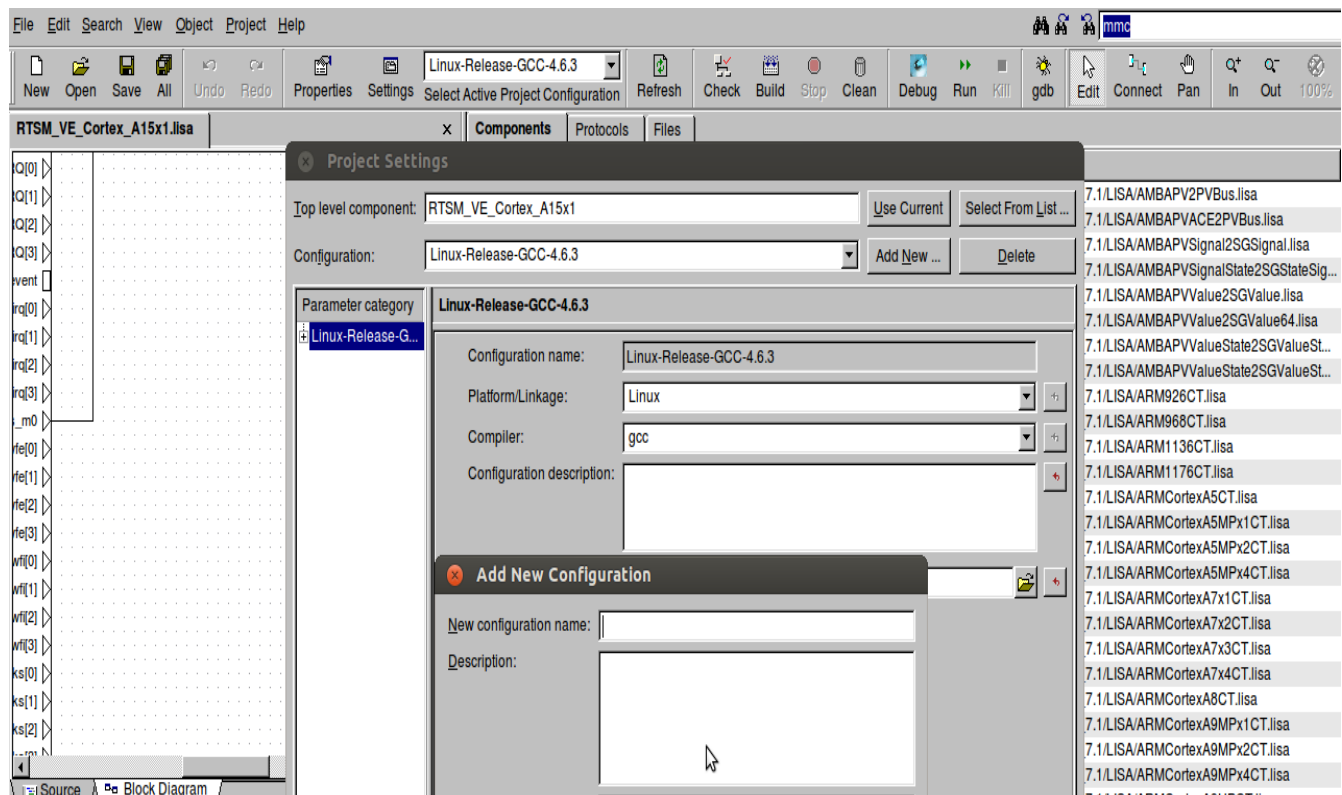
dir>/FastModelsPortfolio_8.3/examples/FVP_EB/Build_Cortex-A8/Linux-Release-GCC-4.7/cadi_system Linux-Release-GCC-4.7.so':

Error while loading lib:

<Fastmodel-install dir>/FastModelsPortfolio_8.3/examples/FVP_EB/Build_Cortex-A8/Linux-Release-GCC-4.7/cadi_system Linux-Release-GCC-4.7.so: undefined symbol: _ZN2sg13ConnectorBase5emptyEv

This error can be avoided by using GCC 4.6.3 to carry out the build. The GCC version for building the project can be changed by selecting the settings option and then by adding the new configuration

The GCC version for building the project can be changed by selecting the Settings option and then by adding a new configuration.



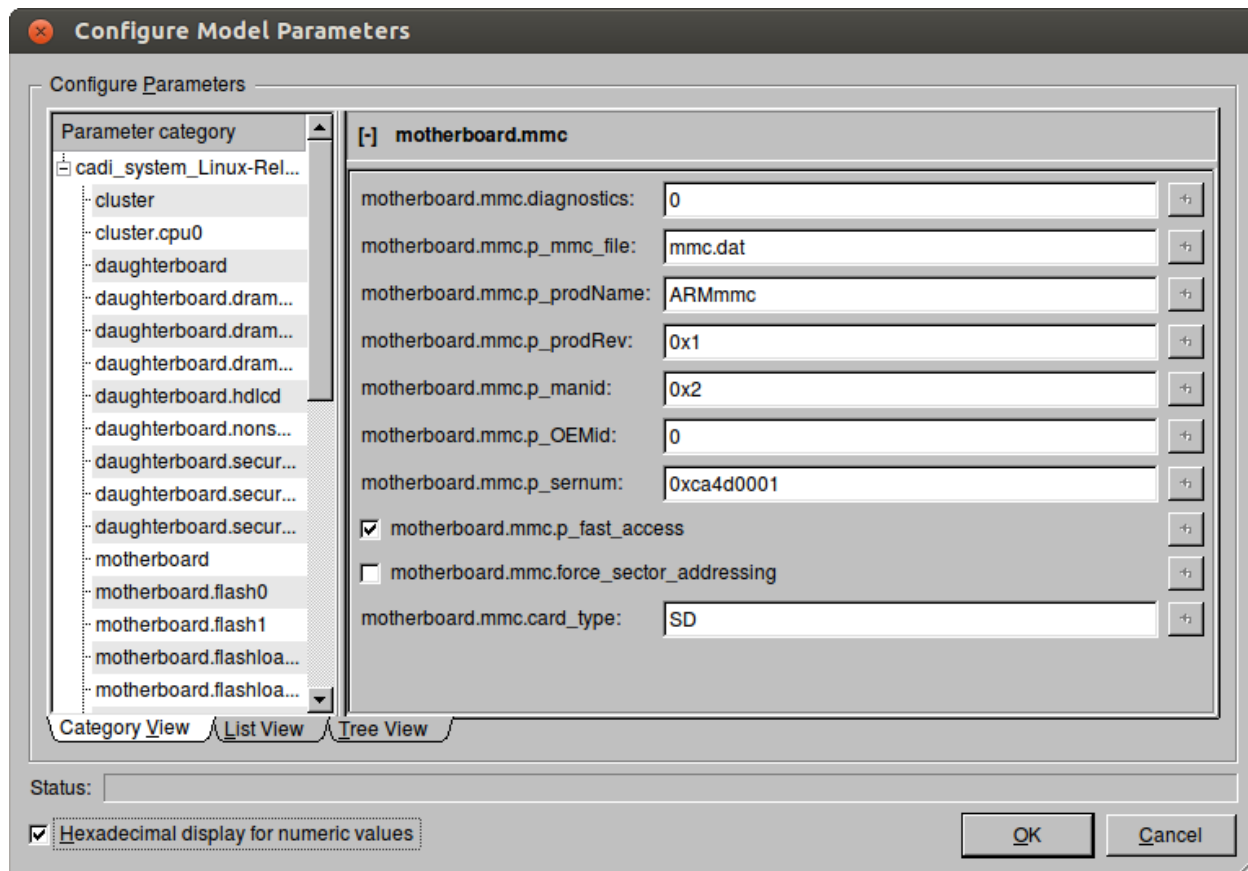
3.4 Booting the image and debugging using Fastmodel

From the configure model parameters change the motherboard.mmc to point to the following root system image

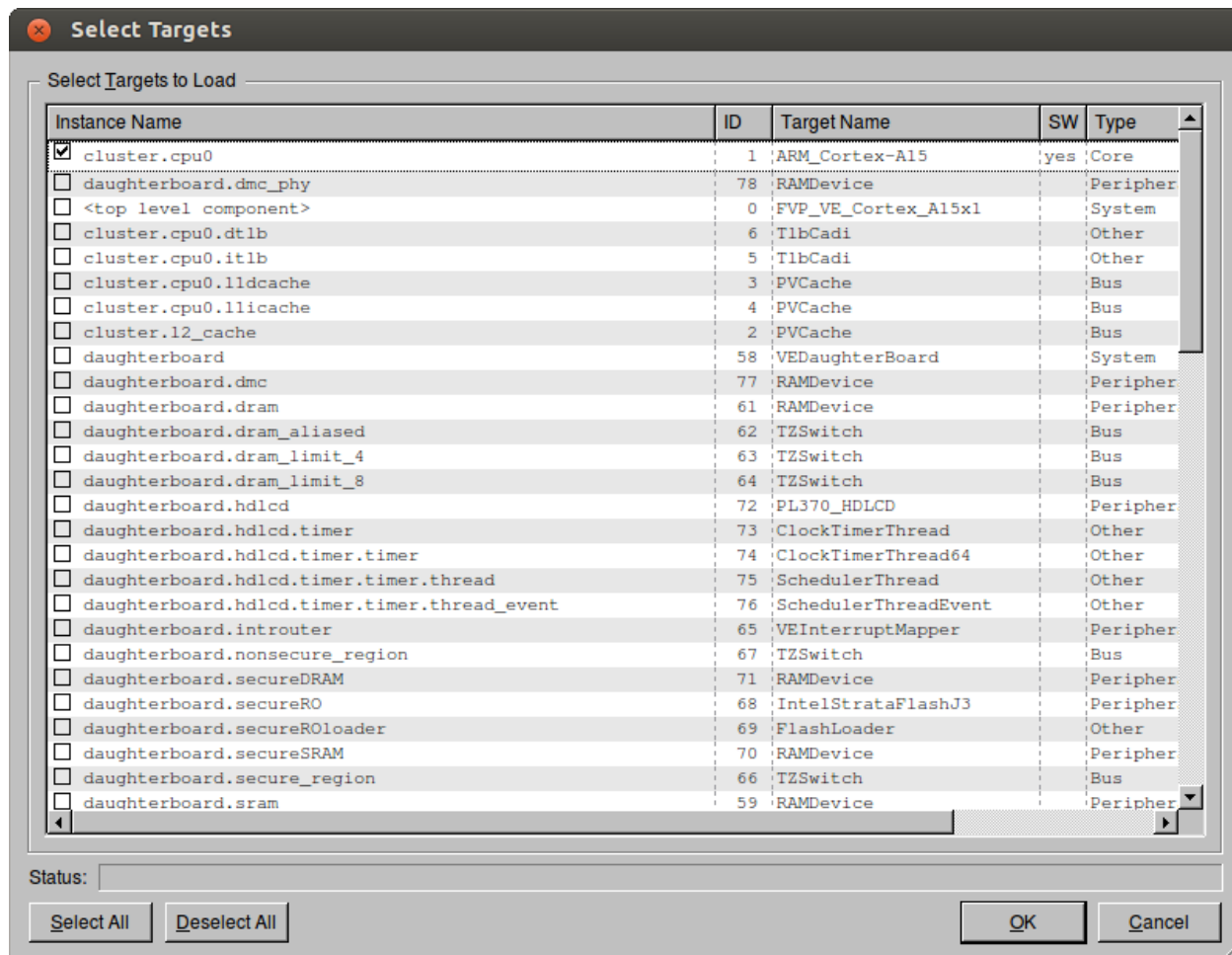
<Fastmodels_Install_directory>/Fastmodels_portfolio_7.1/images/RTSM_EB_LINUX/filesystems/armv5t_min_EB_V6_V7.image.

On clicking Debug, a Debug Simulation window appears which allows the required application to be loaded.

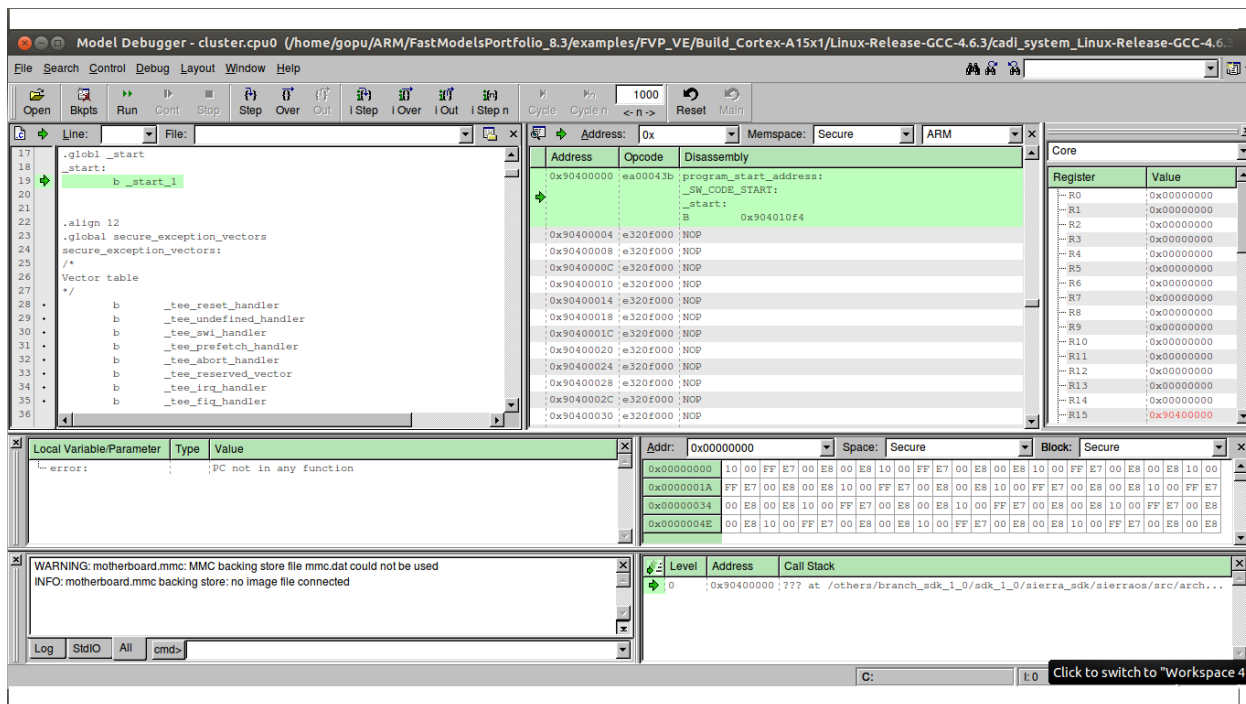
1. if crypto is not enabled choose otzone.elf
 2. if crypto is enabled choose otzone-crypto.elf
- and click ok.



Next the Select Targets window will appear which is already configured.



After all the configuration has been done and proceeding forward by clicking ok, the Model Debugger window will be appearing in the screen. In this window, the run option would make the system to boot, reset will make the system to boot from the starting point.



Login as root.

```

FVP terminal_0
[ 0.678331] ThumbEE CPU extension supported.
[ 0.678388] Registering SWP/SWPB emulation handler
[ 0.680628] drivers/rtnetlink: unable to open rtc device (rtc0)
[ 0.681261] ALSA device list:
[ 0.681306] #0: Dummy
[ 0.681682] RRD03K: getp_image found at block 0
[ 2.931020] EXT2-fs (ram0): warning: mounting unchecked fs, running e2fsck is
recommended
[ 2.931144] VFS: Mounted root (ext2 filesystem) on device 1:0.
[ 2.933107] Freeing init memory: 196K
init started: BusyBox v1.14.3 (2009-11-12 11:03:55 GMT)
starting pid 517, tty '': '/etc/rc.d/rc.local'
warning: can't open /etc/mtab: No such file or directory
/etc/rc.d/rc.local: line 53: /usr/sbin/telnetd: not found
S: devpts
S: udev
[ 3.153125] udevd (533): /proc/533/oom_adj is deprecated, please use /proc/53
3/oom_score_adj instead.
S: sshd
S: dbus id
Thu Jan  1 00:00:04 UTC 1970
S: hald
S: Xorg
R: Xorg
S: dhcdd
Found no /etc/resolv.conf you need one for e.g. browser to resolve URLs
S: ohad
/proc/asound/cards file present
No ARH AC'97 Interface found
/etc/rc.d/platform-intf-init: line 63: head: not found
starting pid 1140, tty '': '/sbin/getty -L ttyS0 38400 vt100'
No network interface 'eth0' found
/etc/rc.d/platform-intf-init: line 63: head: not found
/etc/rc.d/platform-intf-init: line 63: dhclient: not found
No network interface 'usb0' found
lo      Link encap:Local Loopback
        inet addr:169.254.0.100  Mask:255.255.0.0
        UP LOOPBACK RUNNING  MTU:65536  Metric:1
        RX packets:0 errors:0 dropped:0 overruns:0 frame:0
        TX packets:0 errors:0 dropped:0 overruns:0 carrier:0
        collisions:0 txqueuelen:0
        RX bytes:0 (0.0 B)  TX bytes:0 (0.0 B)

REL login: root
login[1140]: root login on 'ttyS0'

BusyBox v1.14.3 (2009-11-12 11:03:55 GMT) built-in shell (ash)
Enter 'help' for a list of built-in commands.

#

```

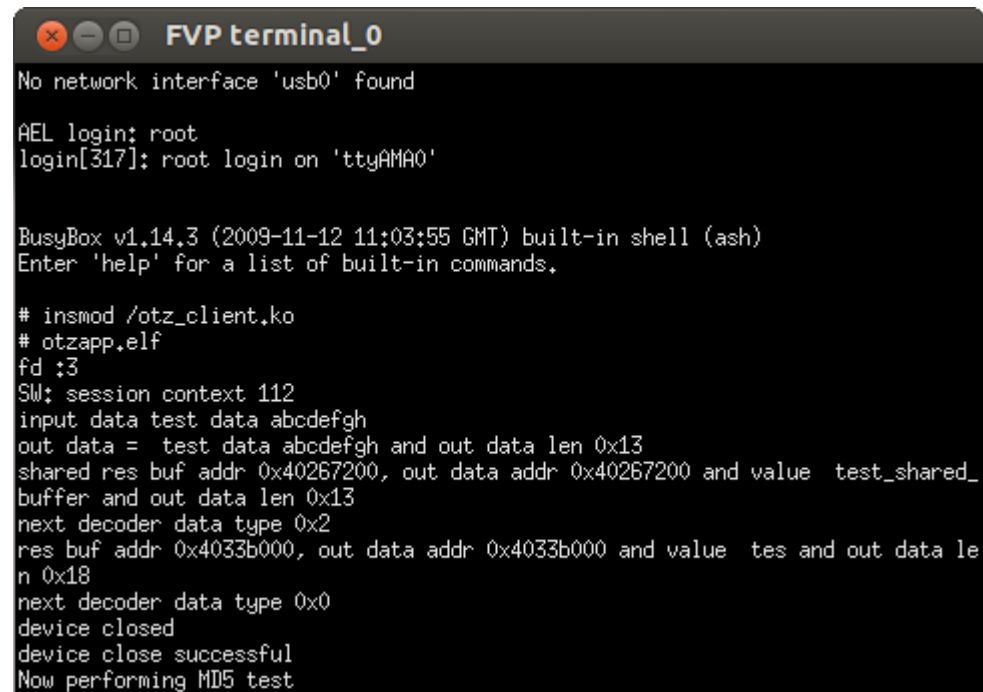
3.5 Testing the Open-virtualization Stack

1. Load the Open-Trust-Zone kernel driver by executing the following instruction.

```
#insmod /otz_client.ko
```

2. Launch the test application.

```
#otzapp.elf
```



```
FVP terminal_0
No network interface 'usb0' found

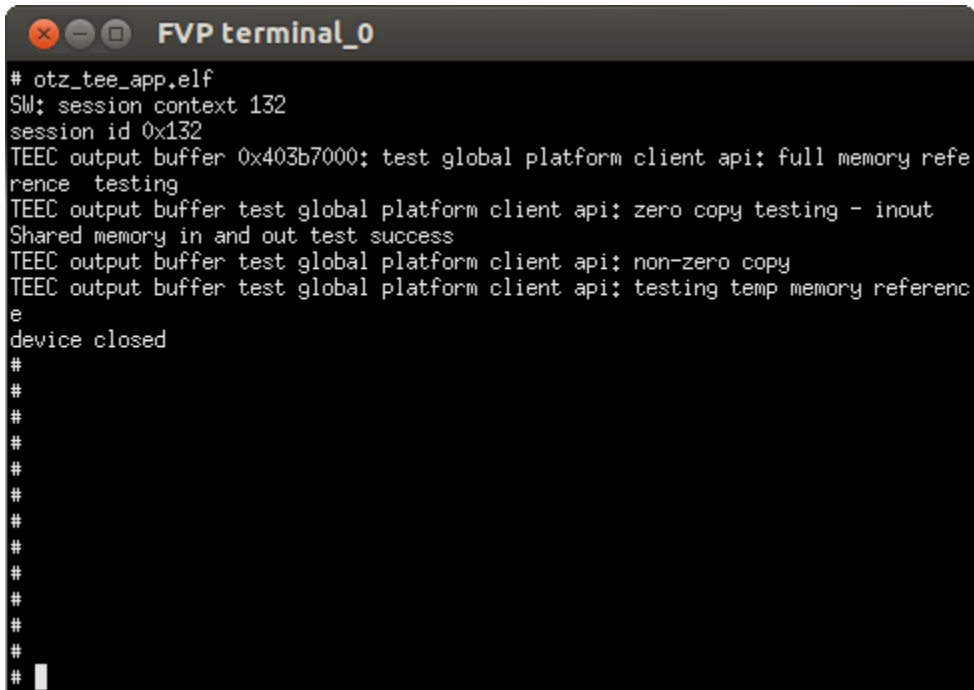
AEL login: root
login[317]: root login on 'ttyAMA0'

BusyBox v1.14.3 (2009-11-12 11:03:55 GMT) built-in shell (ash)
Enter 'help' for a list of built-in commands.

# insmod /otz_client.ko
# otzapp.elf
fd :3
SW: session context 112
input data test data abcdefgh
out data = test data abcdefgh and out data len 0x13
shared res buf addr 0x40267200, out data addr 0x40267200 and value test_shared_
buffer and out data len 0x13
next decoder data type 0x2
res buf addr 0x4033b000, out data addr 0x4033b000 and value tes and out data le
n 0x18
next decoder data type 0x0
device closed
device close successful
Now performing MD5 test
```


3. TEE compliant app can be tested by launching the following application:

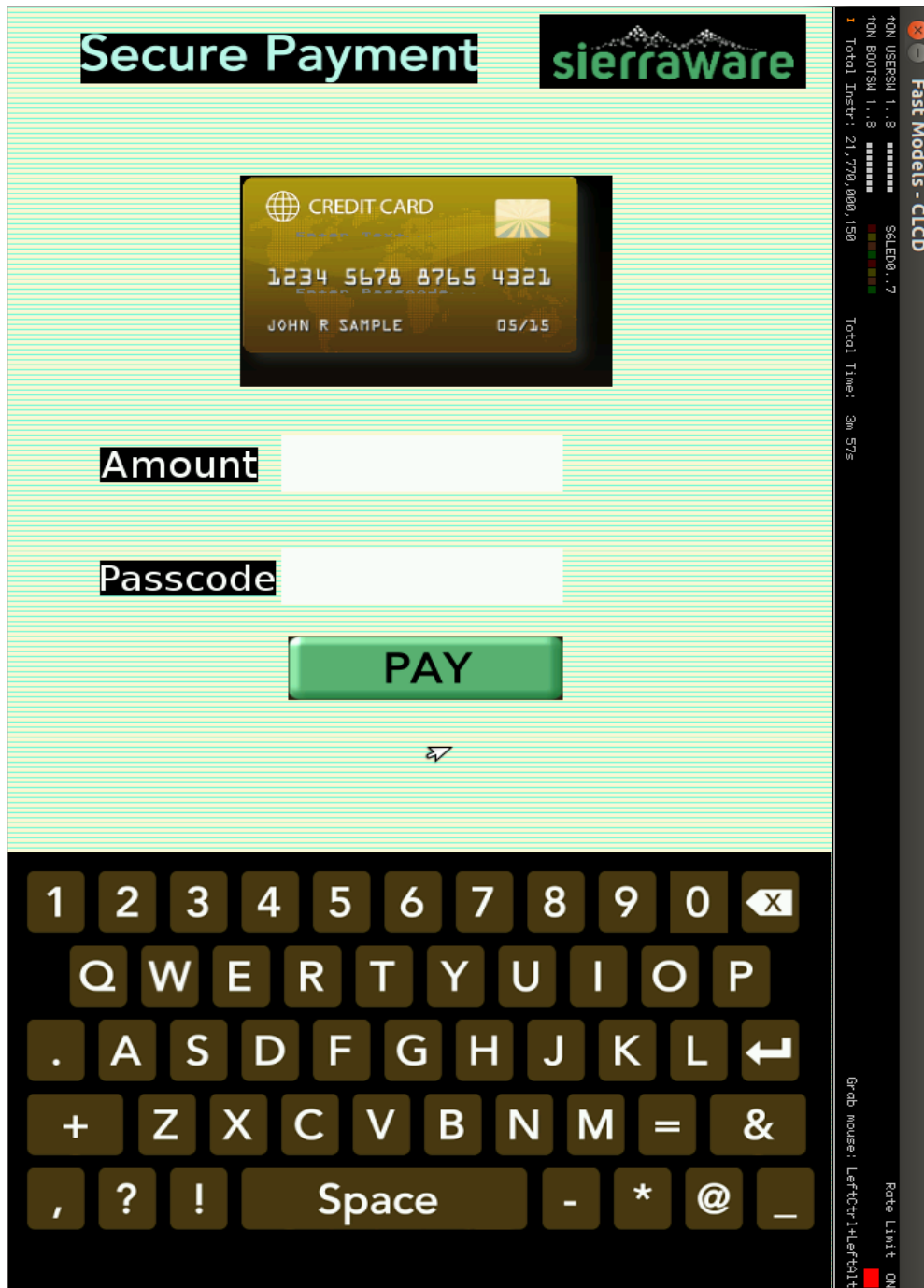
`#otz_tee_app.elf`



```
FVP terminal_0
# otz_tee_app.elf
SW: session context 132
session id 0x132
TEEC output buffer 0x403b7000: test global platform client api: full memory reference testing
TEEC output buffer test global platform client api: zero copy testing - inout
Shared memory in and out test success
TEEC output buffer test global platform client api: non-zero copy
TEEC output buffer test global platform client api: testing temp memory reference
device closed
#
#
#
#
#
#
#
#
#
#
#
#
```

4. Virtual keyboard can be tested by launching the following application:

#otz_virtual_keyboard.elf



- Media Player can be tested by launching the following application:
#otz_play_media.elf

