ESC SV 2012 Embedded Android Workshop part 1

VM Setup

- Set up VirtualBox
- Copy all contents off USB stick to Laptop
- Extract oea-110714.vdi.bz2
- Create new VM
  - Wizard
    - OS type is Linux->Ubuntu (NOT 64bit)
    - 1GB or RAM
    - HDD = oea-110713.vdi
  - Settings
    - 2 CPUs
    - May need to reboot, go into BIOS and enable virtualization instructions
- Start new VM
- Log into VM
  - User is “Opersys” / pwd is “opersys”
- Create a shared folder between VM and Host
  - VirtualBox->Devices->Shard Folders->Add
- Mount the folder

  ```
  $ mount -t vboxsf folder-name mount-location
  ```

- Copy the content retrieved from the USB stick to the VM through the shared folder

MicroSD Programming

- Extract BeagleBone.tar.gz
- Replace the mkmmc-android.sh in BeagleBone/ with http://www.opersys.com/downloads/escsv2012/mkmmc-android.sh
- Plug the MicroSD card into the MicroSD-USB connector and into your laptop
- Share the USB device with the guest VM
  - VirtualBox->Devices->USB Devices->MicroSD card
- Program the MicroSD card

  ```
  $ sudo ./mkmmc-android.sh
  ```

BeagleBone Recognition

- Set up Ubuntu to recognize BeagleBoard
  - See steps 1 to 3 here http://www.opersys.com/blog/beaglebone-android-start
- Get minicom

  ```
  $ sudo apt-get install minicom
  ```

- Configure minicom

  ```
  $ sudo minicom -s
  ```

  - “Serial port setup”
    - /dev/beaglebone-serial
    - 115200 8N1 no hardware flow control
  - ESC
  - “Save setup as dfl”
  - “Exit from minicom”

BeagleBone Bootup

- Plug Bone through barrel connector to laptop
- Plug Bone mini USB to laptop
- Make sure Ubuntu gets control of the USB device
  - VirtualBox->Devices->USB Devices->FTDI BeagleBone
- Start minicom

  ```
  $ minicom
  ```

- You should now see the kernel start up and Android's boot sequence

Android Visualization

- Once Android is started, share it with the VM
  - VirtualBox->Devices->USB Devices->Android
- Connect to target using ADB

  ```
  $ ~/android/android-sdk-linux_x86/platform-tools/adb  shell
  ```

- Start the VNC server

  ```
  # androidvncserver &
  ```

- Tunnel VNC traffic over ADB

  ```
  $ ~/android/android-sdk-linux_x86/platform-tools/adb forward tcp:5901 tcp:5901
  ```

- Connect to target over VNC

```
$ vinagre &
```

Connect->

- Protocol: VNC
- Host: localhost:5901
- Use JPEG encoding

You should now be able to see Android UI

If you encounter too many issues with the VNC over USB

- See explanation on how to set up VNC over Ethernet: http://www.opersys.com/blog/beaglebone-android-start
- You'll need to share your Ethernet with VirtualBox. See VirtualBox's manual.

Android @Work

Explore the Android UI through VNC

Note that all events have to have a “left-click” appended for them to propagate to the target

Check the version of Android you're running

- apps->Settings->About Phone->Android version

Use “logcat” to observe the target's Android

```
$ adb logcat
```

Create a “Hello World” application in the VM and run it on the target

Have fun

Homework

Install U-Boot mkimage

```
$ sudo apt-get install uboot-mkimage
```

Make sure you're using the right “jar”

```
$ sudo update-alternatives --config jar
```

Select “2”

Extract the TI BeagleBone Sources

```
$ tar xvzf /home/opersys/Desktop/ubuntu-ubuntu-bridge/TI_Android_GingerBread_2_3_4_AM335x_Sources.tar.gz
```

Retrieve source

```
$  cd TI_Android_GingerBread_2_3_4_AM335x_Sources/
$ ./.repo/repo/repo sync --local-only
```

Build source

```
$ make TARGET_PRODUCT=beaglebone OMAPES=4.x -j4
```

This will take a while

Don't “make clean”