International Journal on Emerging Technologies (Special Issue on NCRIET-2015) **6**(2): 197-202(2015)

ISSN No. (Print) : 0975-8364
ISSN No. (Online) : 2249-3255

Friendly ARM MINI2440 & Dalvik Virtual Machine with Android

Sangamesh Gama
Assistant Professor, Computer Science and Engineering Department,
BKIT Bhalki, INDIA

(Corresponding author: Sangamesh Gama)

(Published by Research Trend, Website: www.researchtrend.net)

ABSTRACT: Android is a popular mobile-device platform developed by Google. Android is an operating system based on Linux with a Java programming interface. Android provides tools such as compiler, debugger and device emulator which is well known as Dalvik Virtual Machine (DVM). In this paper we are going to discuss on booting the ARM board, with Android OS. To boot ARM with Android cross-compilation tool is used. To list the activities, a list activity application is created in eclipse integrated development environment. With the help of the Android Development Tool, the application is transferred to ARM board. Mini2440 is a practical low-cost ARM9 development board, is currently the highest in a cost-effective learning board. It is for the Samsung S3C2440 processor and the use of professional power stable core CPU chip to chip and reset security permit system stability [1].

Keywords: Android, Dalvik virtual machine, MINI2440, Android package, Android SDK.

I. INTRODUCTION

Android is created by the Open Handset Alliance which is lead by Google. Android is a software stack for mobile devices that includes an operating system, middleware and key applications. Android operating system is based on Linux with a Java programming interface. [2]

The Android SDK (Software Development Kit) provides the tools such as a compiler, debugger and a device emulator as well as its own Java Virtual machine (Dalvik Virtual Machine - DVM). Android also provides APIs (Application Programming Interface). The SDK and APIs are necessary to begin developing applications on the Android platform using the Java programming language. Android offers new possibilities for mobile applications by offering an open development environment built on an open source Linux kernel. Hardware access is available to all applications through a series of API libraries, and application interaction, while carefully controlled, is fully supported. Third-party and native Android applications are written using the same APIs and are executed on the same run time. The run time includes core libraries and the Dalvik Virtual Machine. This paper discusses, about android being booted on the Dalvik Virtual Machine, with the help of the ADT (Android Development Tools) plug in. The ADT plug in is provided by Eclipse IDE (Integrated Development Environment).

Also, about how android is being cross-compiled for the embedded board- ARM Board. Arm board also otherwise called as MINI2440. The MINI2440 is a single board computer based on Samsung S3C2440 microprocessor.

A. Main Features

ARM Board

Few features of ARM board are as follows: [1]. Fig 1 shows the ARM board/MINI 2440.

1) CPU Processor

Samsung S3C2440A, frequency 400MHz, the highest 533MHz.

2)SDRAM Memory

-On-board 64MB SDRAM

-32-bit data bus

- SDRAM clock frequency up to 100 MHz

3)FLASH Memory

-On-board 64 MB NAND flash, Power-down non-volatile

- On-board 2 MB NOR flash, Power-down non-volatile, BIOS has been installed

4)LCD Display

- On-board integrated 4-wire resistive touch screen interface, you can directly connect

-4-wire resistive touch screen

- Support for black and white, 4 gray-scale, 16 gray-scale, 256-color, 4096-color ST

- LCD screen size from 3.5 to 12.1, 1024x768 pixels screen resolution can be achieved

- Support for black and white, 4 gray-scale, 16 gray-scale, 256-color, 64K-color, True
- Color TFT LCD screen size from 3.5 to 12.1, 1024x768 pixels screen resolution can be achieved.
- Standard configuration for the NEC 256K-color 240x320/3.5 TFT True Color LCD screen with touch screen.

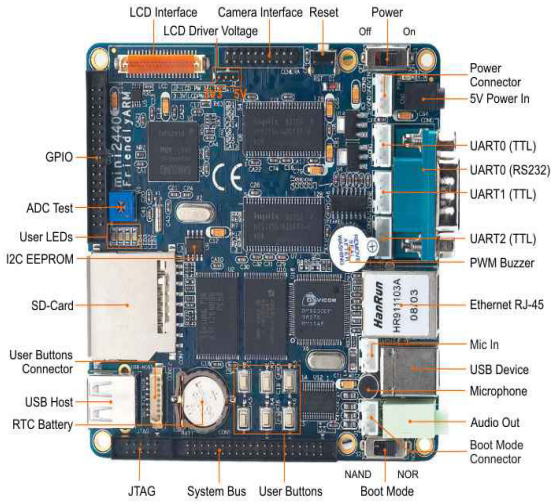

Fig. 1. Friendly ARM/MINI2440.

5) Interfaces and Resources
- 1 100 Mbps Fast Ethernet RJ-45 interface (used network chips DM9000)
- 3 Serial ports
- 1 USB host
- 1 USB slave (B-type interface)
- 1 SD card storage interface
- 1 channel stereo audio output interface, all the way microphone interface
- 1 2.0mm pitch 10-pin JTAG interface
- 4 User LEDs
- 6 User buttons (with lead blocks)
- 1 buzzer PWM control
- 1 adjustable resistor, analog-to-digital converter for A/D test
- 1 I2C-bus AT24C08 chip for I2C-bus test
- 1 2.0 mm pitch 20-pin camera interface
- On-board real-time clock battery
- Power interface (5 V), with power switch and indicator light

6) System Clock Source
- 12 MHz passive crystal

7) Real-Time Clock
- Internal real-time clock (with lithium battery back-up)

8) Expansion Interface
- 1 34-pin 2.0 mm GPIO interface
- 1 40-pin 2.0 mm system bus interface

9) Dimension
- 100 mm x 100 mm

Android

Some of the features of android [3] are as follows:

1) Handset layouts
The platform is adaptable to larger, VGA, 2D graphics library, 3D graphics library based on OpenGL ES 2.0 specifications.

2) Connectivity:
GSM/EDGE, Bluetooth, Wi-Fi, LTE, NFC and WiMAX.

3)Java support
While most Android applications are written in Java, there is no Java Virtual Machine in the platform and Java byte code is not executed.

4) Bluetooth
Supports A2DP, AVRCP, sending files (OPP), accessing the phone book (PBAP), voice dialing and sending contacts between phones.

5) Multitasking
Multitasking of applications, with unique handling of memory allocation, is available.

6) External storage
Most Android devices include microSD slot and can read microSD cards formatted with FAT32, Ext3 or Ext4 file system.

7) Messaging
Along with SMS and MMS, Android Google Cloud Messaging (GCM) is also a part of Android Push Messaging service and Android Cloud To Device Messaging (C2DM) and now enhanced version of C2DM.

8) Additional hardware support

Android can use video/still cameras, touch screens, GPS, accelerometers, gyroscopes, barometers, magnetometers, dedicated gaming controls, proximity and pressure sensors, thermometers, accelerated 2D bit blits and accelerated 3D graphics.

II. ARCHITECTURE

Android implements a complete software stack for running mobile device applications. Android is an Open Handset Alliance (OHA) project done by Google [2]. It uses Linux kernel as the kernel of the operating system. Therefore Android architecture and the performance are more similar to other Linux based operating systems.
Mainly Android architecture can be categorized as OS, middleware and applications. Fig 2 shows a complete architecture of Android operating system.

A. Applications

All mobile/portable devices will have some applications which perform basic features. Features like Contacts, SMS, and MMS will be available if the device is Phone. [3]

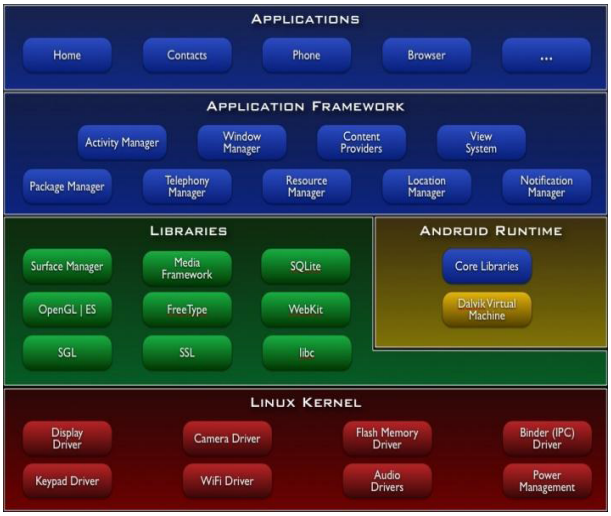

Fig. 2. Android Architecture.

The application should allow the third party applications with the help of Application Programming Interface (API). These android applications are included in the Applications layer.

B. Application Framework

Application framework gives the user interface controllers for developing software. They manage the behaviour of the running application. By providing an open development platform, Android offers developers the ability to build extremely rich and innovative applications. [3]. Developers are free to take advantage of the device hardware, access location information, run background services, set alarms, add notifications to the status bar, and much, much more.
Underlying all applications is a set of services and systems, including: Views, Content Providers, Resource Manager, Notification Manager, and Activity Manager. Each manager is takes control to manage activities. For an Example: Content Providers that enable applications to access data from other applications (such as Contacts), or to share their own data.

C. Core Libraries & Android Runtime

Middleware category includes Libraries and Android Runtime. Libraries are the techniques which add features to the operating system. [3]. Android Runtime can be included in middleware of the operating system. It has virtual machine which is similar to Java Virtual Machine (JVM) names as Dalvik Virtual Machine. This is where the developed applications are running. Because of the limited memory and battery power Android cannot directly use JVM. Therefore more reliable and mobile device targeted Dalvik Virtual Machine is used.

D. Linux Kernel

The kernel used in Android is a 2.6 series Linux kernel modified to full fill some special needs of the platform. It is based on Linux kernel for core system services such as security, memory management, process management, network stack, and driver model. [3].
The kernel is freely available and the development process is visible through the public Android source repository. There are hardware specific kernels available for OMAP.

III. DALVIK VIRTUAL MACHINE

A. Overview of Dalvik Virtual Machine (DVM)

The necessary byte code interpreter – the virtual machine – is called Dalvik. Instead of using standard byte code, Dalvik has its own byte code format which is adjusted to the needs of Android target devices. The byte code is more compact than usual Java byte code and the generated .dex files are small.

As a multitasking operating system, Android allows every application to be multithreaded and also to be spread over multiple processes. For the sake of improved stability and enhanced security each application is separated from other running applications.

Every application runs in a sandboxed environment in its own Dalvik virtual machine instance. This requires Dalvik to be small and only add little overhead.

There are two major areas to consider for minimizing the memory usage. Firstly the application itself has to be as small as possible and secondly the memory allocation of each application has to be optimized. In addition to the reduced usage of valuable memory one gains faster application load times and less needed disk storage.

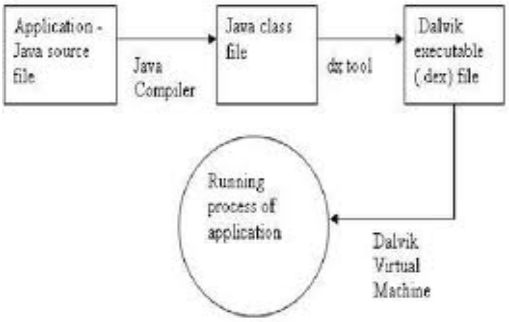

Fig. 3. Overview of APP on DVM.

Java applications for Dalvik get compiled like other Java programs with the same compilers and mostly the same tool chain. Instead of compressing and packaging the resulting class files into a .jar file, they are translated into .dex files by the *dx* tool.

These files include the Dalvik byte code of all Java classes of the application. Together with other resources like images, sound files or libraries the .dex files are packaged into .apk files. In order to save storage space, .dex files only contain unique data. If multiple class files share the same string, this string would only exist once in the .dex file and the multiple occurrences are just pointers to this one string. Fig 2 depicts the overview of how an application runs on the DVM.

The same mechanism is used for method names, constants and objects which results in smaller files with much internal "pointing".

The Dalvik byte code is designed to reduce the number of necessary memory reads and writes and increased code density compared to Java byte code. For this purpose Dalvik uses its own instruction set with a fixed instruction length of 16 bit. [4]

IV. IMPLEMENTATION

Implementing android on DVM is possible with the help of Eclipse IDE. Android offers a custom plug in for the Eclipse IDE, called Android Development Tools (ADT) that is designed to give a powerful, integrated environment in which to build Android applications. To port different versions of android on DVM, just eclipse IDE is enough, which offers versions of android, which can be deployed using the Android SDK and Android AVD manager.

SDK that is been provided by Android, comes with the package with necessary libraries and tools, that are required for creating a DVM as well as to create applications and execute the same on DVM.

A. Tools

The Android SDK includes a variety of tools that helps to develop mobile applications for the Android platform. The tools are classified into two groups: SDK tools and platform tools.

SDK tools are platform independent and are required no matter which Android platform is used to develop. Platform tools are customized to support the features of the latest Android platform.

1. SDK tools. The SDK tools are required to develop Android applications. The most important SDK tools include the Android SDK and AVD Manager.

The Emulator, A QEMU-based device-emulation tool that is used to design, debugs, and tests the applications in an actual Android run-time environment. The Dalvik Debug Monitor Server lets us debug android applications.

With the ADT plug-in that is installed in eclipse, the AVD (Android Virtual Device) Manager which provides the option of creating an AVD, with the android OS version of choice.

2. Platform-tools. The platform tools are typically updated every time a new SDK platform is installed. Of all the platform-tools—the Android Debug Bridge (adb), is a versatile tool that lets to manage the state of an emulator instance or Android-powered device. Also Android application (.apk) file can be installed on a device.

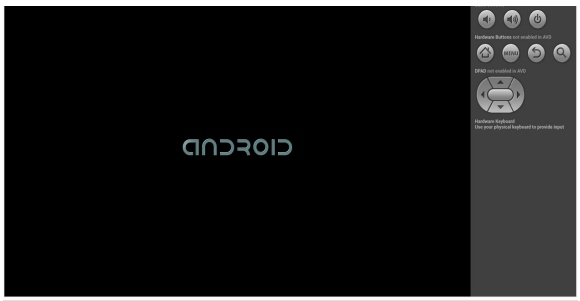

Fig. 4. DVM with Android Booting.

B. Android on ARM Board

To boot ARM Board with Android OS, Boot Files and Root File System are required. A concept called cross compilation is used to create booting files and root file system.

1) Cross Compilation

A cross compiler is a compiler capable of creating executable code for a platform other than the one on which the compiler runs. Cross compiler tools are used to generate executables for embedded system or multiple platforms.

It is used to compile for a platform upon which it is not feasible to do the compiling, like microcontrollers that don't support an operating system.

Android can be ported into mini2440 by using Linux commands. For these pre-built binaries of Bootloader, Kernel and Root File System are required, which can be downloaded for arm processor.

The cross-compiler that is required for compiling the bootloader and file system images is ABI standard *arm-linux-gcc-4.3.2* cross compiler. With the downloaded binaries, the kernel config_mini2440.[5]

make menuconfig $

This command will create the *mini2440T35_android.img*.

The next file to be cross-compiled is the ZImage, which acts as a primary bootloader, which is the kernel.

$make ZImage

Now, the required part is root file system, which can be created by using following command.

$make yaffs2

The files are ready to be ported to Friendly ARM.

To send binary images to the Friendly ARM board setup is required between the cross-compiled images created in PC and the ARM Board. Serial port connection between host PC and Board for seeing Bootloader messages, and selecting image to be transferred. USB connection between host PC and board for actual transfer of binary images.

The Friendly ARM Mini 2440 has a Switch for selecting the Flash to boot from i.e. either NAND Flash or NOR Flash. For this installation, the switch should be set at NOR Flash setting.

The mini Super Vivi bootloader runs from NOR Flash which enables transfer of the binary images.

Fig. 5. Android booted on ARM board.

When the Friendly ARM is booted, a list of commands is been displayed, that is in Super Vivi mode.

Step1: To erase the contents in NAND flash, so selecting 'x' would format the system.

Step 2: Choosing 'v' to start the download bootloader.

Step 3: Choosing 'k' so download the android kernel: zImage

Step 4: Select 'y' to download the mini2440T35_android image.

Step 5: Choosing 'b' command to start boot.

Also choosing S2 side of Nand flash reset the ARM board.

Once the board is booted up, we can see the Android blooming up.

V. APPLICATION

List Activity application is been created with the help of Android API's in the eclipse Integrated environment. This application is useful for an individual to list his/her own activities to be completed. It's a very useful application in the day to day life.

Using the Android Virtual Device(AVD) that is provided in the eclipse IDE, after installation of Android Development Plug in, this AVD can be initialized.

Fig. 6. List Activity Application.

The jar file of the application created is converted to APK file with the help of .dex tool, provided by android SDK. .apk, is the format that android can understand.

Once, the apk is available this can be pushed to the device such as friendly arm/ DVM is possible using the *adb* which is present in the platform-tools provided by the android SDK.

Adb is the android debug bridge, which acts as a bridge between the PC and the friendly arm or even the DVM. Fig. 6 shows the apk being installed on the friendly arm and fig. 7 shows how the apk is installed.

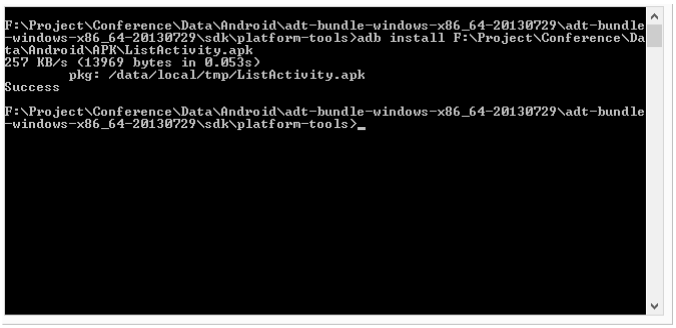

Fig. 7. APK Installation.

VI. CONCLUSION

Android OS is ported on Friendly ARM /MINI 2440 board, and list activity application is also ported with the help of adb tool. Also, Android is ported on the Dalvik Virtual Machine.

REFERENCES

[1] Friendly ARM MINI 2440 Overview Open Handset Alliance *http://www.openhandsetalliance.com/android_overview.html*

[2] www.androiddevelopers.com

[3] Analysis of the Android Architecture *Stefan Br¨ ahler*

[4] http://wiki.embeddednirvana.org/Flashing_ANDROID_on_Friendly_ARM_Mini_2440