ISSN 2395-1621

Implementation of Firmware for Android Mobile

[#1]Nikhil Dhiwar, [#2]Mahesh Dhotre, [#3]Ram Kole, [#4]Pratik Nalawade

[1]ndhiwar63@gmail.com
[2]mdhotre225@gmail.com
[3]rbkole07@gmail.com
[4]n.pratik777@gmail.com

[#1234]Department of Computer Engineering
JSPM's
Imperial College of Engineering & Research
Wagholi, Pune--412207

ABSTRACT

The wide support from large companies, especially Google, have made Android one of the most important contestants in the mobile sector. The Android platform has become one of the most popular operating system with millions of new users each year. Here various things about a Firmware have been discussed. It focusses on things like How customized firmware is better with advancement for day to day use. In this paper we present customization and installation of custom firmware for android mobiles. We describe the design, modification and extra features of Customized Firmware.

Keywords: **System Design, firmware, CPU frequency, Governor, Flashing, Tweeks.**

ARTICLE INFO

Article History

Received: 31st May 2016

Received in revised form :

1st May 2016

Accepted: 2nd June 2016

Published online :

3rd June 2016

I. INTRODUCTION

Today's most of mobile phones are Android based (on basis of Linux Operating system). The platform was originally created by Android Inc., which was then later bought by Google and released as the AOSP (Android Open Source Project) in 2007. This announcement was accompanied by the founding of the OHA (Open Handset Alliance) [4][5], Android phones accounted for 82% of globally sold smartphones in 2nd Quarter of 2014.

Linux is an open source operating system. This can be modified (customized) as per different user. The code of Linux is freely available on internet. So most of mobile manufactures like Samsung, HTC, Motorola, Micromax etc. designing android phones based on Linux. The firmware provided by default by the device manufacturer. It is the official Firmware for the device. When you purchase a phone from a carrier, it often comes packed with bloatware packages. NASCAR apps, TV apps, a Contacts app that stores your contacts on your carrier's servers instead of on your phone. These apps can clutter your system and waste disk space. Manufacturers even add their own software before the carrier gets to it, so you have two companies each adding their own bloatware to your phone before it gets to you.

So here we are going to design a customized firmware for android mobile phone having specific configuration. This firmware that we are going to design has features like Super-user access, customizable GUI, smooth performance, improved battery life.

II. RELATED WORK

Unlike other mobile operating systems such as Windows Phone or iOS, Android applications are written in Java and run in a Dalvik VM (Virtual Machine). This virtual machine is a core component, because all Android applications and the application framework are executed by it. Similar to other platforms, applications can be obtained from a dedicated place called Google Play [5].

III.EXISTING SYSTEM

The Android system architecture shown in Fig. 2.consists of five layers: Linux kernel, libraries, Android runtime, application framework and applications [4].

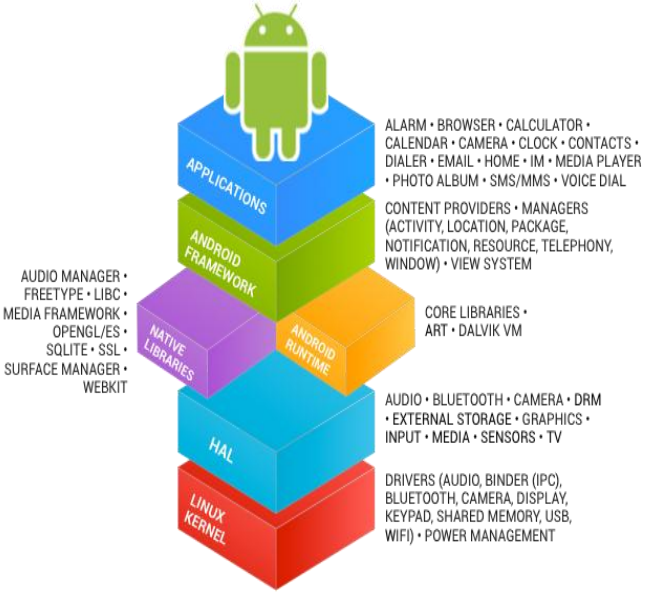

Fig. 1 Architecture of Android OS

The kernel is a 2.6 series Linux kernel device driver modified mainly for power and memory management purposes. The next level consists of native libraries written in C or C++. Due to the fact that Android was designed to run on low powered CPU (Central Processing Unit) and GPU (Graphics Processing Unit) devices with limited memory, the main libraries like libc or libm were developed to ensure low memory consumption. This layer contains also a Surface Manager responsible for screen access, a Media Framework optimized for handling audio and video codecs, e.g. MP3, MPEG-4 (Moving Picture Experts Group), H.264, a SQL (Structured Query Language) database and a native web browser engine (Web Kit).

The Android Runtime includes the Dalvik Virtual Machine and Java Core Libraries. This VM is an interpreter for byte codes that have been converted from standard Java jar files into dex files, which are more compact and efficient than class files considering the limited memory and battery power of an Android mobile device. The most important part of the Application Framework layer is the Activity Manager, responsible for controlling the life cycle of applications. Any Android application runs in its own sandboxed Dalvik VM and can consist of multiple components, e.g. services, activities, content providers, which can interact with each other within one single or many different applications on demand. Additionally to the actual Java class library, the Android SDK contains all tools necessary to build an application.

• AAPT (Android Asset Packaging Tool) – enables the developer to create, view and update zip archives in the form of apk files. Those files, containing all the resources and the program itself, can be transferred to and installed on any Android device or emulator;

• ADB (Android Debug Bridge) – sets up a connection with a physical Android device or emulator in order to transfer and install apk files on it. It also enables the developer to execute remote shell commands;

• AIDL (Android Interface Definition Language) – used to generate code enabling two processes on a single Android device to communicate with each other using IPC (Inter Process Communication);

Every Mobile manufacture have their own designed Firmware with their won skins and Pre-loaded application set (Bloatware).they provide some basic as well as unnecessary applications which consumes a lot of data as well as memory space available in mobile.

Most of manufactures have low application memory space. Today's applications are heavy which occupy high memory space, which make user's mobile slow and always popup like low space on storage etc.

The basic android firmware provided by manufactures is not easily modified. So some application and add-ons uses more data. It require data synchronization continuously which make system slow as well as effect on the Battery life. Continuous data synchronization require more power supply. So user always need continuous power supply. The battery life of mobile phone get low in just few hours or mobile phone gets heat. This may slow the performance of mobile phone.

Here end-user is not have full access as per administrator in Linux. User is not able to control access of personal dada by application which may occur sometime security problem. He/she is not going to set permission like read, write, and access as per different application. Exiting system don't have enough much modification as user requirement. The official firmware is like a closed box which is not customizable.

IV. PROPOSE SYSTEM

The important reason to design such a firmware is Customization. Many of user like to modify their own system as they want. They always need changes in system. Modification always make user get engaged in their phone.

Installation of Firmware:

A. Pre-requirement:

- Device must be charged 70%.
- Temp ClockwordMod.
- Custom Firmware Zip.

B. Installation:

- Power off device.
- Boot into Recovery.
- Wipe Data and Cache.
- Select Firmware Zip.
- After completion of flashing Reboot.
- Wait for Boot Up
- Finish

So in this firmware we are going to design and implement following features like

Super User Access:

Root access in Android/Linux is like the administration permissions in Windows – you get permission to do almost anything in your phone/tablet. Isn't it wonderful? Many of the people get rid of the default firmware to enjoy root access – you become the master of your Android – you decide what apps will be installed, which services will run in the background and much other similar permission.

Fig. 2. Super User Access

Regular Updates:

Many of the times, a new Android release come in the market but your phone company don't provide the update for your phone. Then where you stand? Your mobile vendors forget about your phone and start focusing on its new models. Then in this firmware we are providing regular updates as per official firmware via OTA.

Customizable GUI:

Here user can easily change i.e. they can customize themes (like Icons, Developer Options as well as various modes as per user).Modification is going to easy for end user.

Smooth Performance:

Underclocking can greatly affect your device's battery life. As mentioned, under clocking is only helpful if you're not a game freak and don't use your device that much. In that scenario you should probably underclock your device and restrict its CPU to a suitable low speed.

As the purpose of a CPU is to perform tasks on your Android device, higher is the CPU, faster it will perform the tasks and less will be the chances of lag. Overclocking is mostly useful on old devices having less CPU clockspeed. If you wish to play 3D/HD games or you want to run the games / apps lag free you may Overclock your device's CPU and it will work like a new device with higher clock speed.

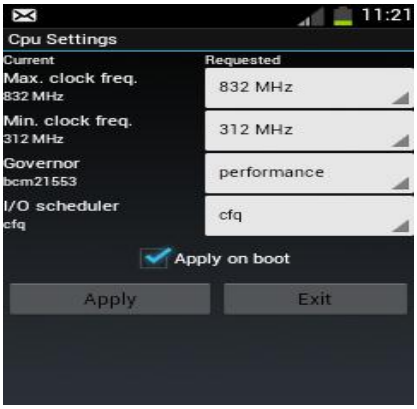

Fig. 3. CPU Frequency

Extra Toolbox:

This conclude extra features for end user like powersevr mode, Notify editor, development box, System behaviour, etc. These features are available to end user for without any complex modification.

Here we provide an extra app with above mentioned features as an touch and use way. S user can easily handle them without any problem just on one click.

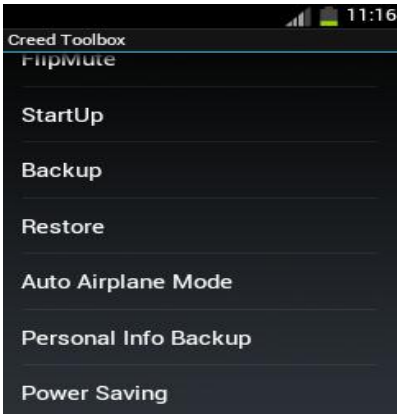

Fig. 4. Extra Tool

Advantages of proposed system:

1. Customize skins to change your entire operating system look.
2. Change quick setting menu which include add your most of used setting shortcut.
3. Easily overclock/underclock to increase performance and battery life.
4. Easily enable root access by toggling a system setting.
5. Easy touch and Use Features.
6. No complex modification as Developer.
7. End User can easily change governer as per requirement.

V. CONCLUSION

User always need Change, if change is not available then user focuses on new product rather than system. It takes a while for manufacturers to adopt a new version of Android to already released devices available on the market. Usually they do not provide a continuous support for all. We are providing a firmware which provide regular update for those mobile designed on specific configuration.

This firmware will improve the system performance as well as battery life. Customization is important part of our development which make user different experience good than default system firmware.

VI. ACKNOWLEDGEMENT

The authors wishes to thank Prof. Rashmi Sonawane (Guide), Prof.Darshika Lothe (PG–Coordinator), Prof. S.R.Todmal (HOD) and Dr. Sachin Admane (Principal) for valuable guidance and encouragement.

VII. REFRENCES

[1] Android OS: A Review Przemyslaw Gilski1, Jacek Stefanski1 1Gdansk University of Technology, Faculty of Electronics, Telecommunications and Informatics, Department of Radio Communication

Systems and Networks, ul. Gabriela Narutowicza 11/12, 80-233 Gdansk, Poland

[2] Custom ROM's in Android Meetu Gupta#1, Abhishek Bhardwaj*2, Lakshay Garg #3 A3rd Year, Guru Gobind Singh Indraprastha University Delhi, India *Assistant Professor, Department of BCA, Sirifort College of Computer Technology and Management Delhi, India

[3] A Power Saving Method with Consideration of Performance in Android Terminals Kyosuke Nagata Electrical Engineering and Electronics Kogakuin University Graduate School Tokyo, Japan cm12021@ns.kogakuin.ac.jp Saneyasu Yamaguchi, Hisato Ogawa Department of information and Communications Engineering Kogakuin University Tokyo, Japan sane@cc.kogakuin.ac.jp, c508029@ns.kogakuin.ac.jp

[4] Android Developers webpage, http://developer.android.com/index.html [access: 02.2015].

[5] Wang C., Duan W. Ma J., Wang C. (2011). The Research of Android System Architecture and Application Programming. ICCSNT.

[6] Macario G., Torchiano M., Violante M. (2009). An InVehicle Infotainment Software Architecture Based on Google Android.SIES.

[7] Kundu T.K., Paul K. (2010) Android on Mobile Devices: An Energy Perspective, CIT.

[8] Schmidt H. G., Raddatz K., Schmidt A.D., Camtepe A., Albayrak S. (2009). Google Android – A Comprehensive Introduction. TUB-DAI.

[9] Murphy M. L. (2008) The Busy Coder's Guide to Android Development. CommonsWare.

[10] Blasing, T., Batyuk, L., Schmidt, A.-D., Camtepe, S.A., Albayrak, S.: An android application sandbox system for suspicious software detection. In Malicious and unwanted software (MALWARE), 2010 5th international conference on, pp. 55–62, IEEE (2010)

