

Software Based Methodologies to Extend Energy Level of Android Mobile Devices

Thanapal P

*Assistant Professor Selection Grade, School of Information Technology and Engineering,
VIT University, Vellore. Tamilnadu. India. thanapal.p@vit.ac.in*

Dr. Saleem Durai M.A.,

*Associate Professor, School of Computing Science and Engineering
VIT University, Vellore. Tamilnadu. India. masaleemdurai@vit.ac.in*

Abstract

Advanced Smartphones are profoundly construct in android platforms and the vast majority of the low end devices are additionally accompanies android operating system, on other hand the same element import huge requirements on their memory, processing, and other storage hardware devices. Android application accompanies a limitless element and operations with extreme size which devours the greater part of the energy to perform the straightforward undertaking it's because of the User Interface (UI), here by dealing with the energy effectively is a foremost in the greater part of the android cell phones. The various utilize and scope of the remote interfaces and sensors notwithstanding this there are numerous power hunger application which exploit these energy can diminish the battery life of android devices. On the other hand, the craft of lithium-ion batteries obviously shows that efficiency ought to be accomplished both at software and hardware level. In this paper we are going to implement the software for all android devices to save the battery life by flipping off the mobile data and Wi-Fi services when not being used.

Keywords: Operating Systems, Mobile Communication, Wi-Fi Services, Energy Efficiency, Mobile Devices.

Introduction

The previous decade has recognized that there is a gigantic development in the wireless handheld gadgets and internet technologies. Most generally utilized handheld gadgets are Black Berry, Nokia, iPhone, and different cell phones which accompanies android gadgets are increasing much fame with extraordinary functionalities and processing abilities which likewise underpins the third party application, modern smartphones accompanies mixture of application and number of wireless interfaces in particular, Wi-Fi, 2G, 3G and Bluetooth[3].

Android gadgets have been putting forth various radio frequencies. Google has created open source operating system called Android [7]. Android force Dalvik JVM is streamlined particularly for moderate CPU. A large portion of the handheld gadgets accompanies lithium ion batteries, which not able to withstand entire day in light of the fact that the vast majority of the applications are made heavier so it devours much battery specifically games which has high end graphics and numerous different wireless applications issues, now a

days numerous applications accompanies immense size which performs numerous operations, which expends the vast majority of the battery. Meanwhile the android operating system has some dynamic energy management strategies incorporated with the product that moderate the power [4]. However clients still grumble about the battery life. In this paper we implemented the application totally in light of java. We assessed our own key stroke so it can be easily installed in the android gadgets without checking licensing [5]. We chose the arrangement that can help the client to flip their mobile data or Wi-Fi service off with no delay, this application will naturally empower the night mode, there are numerous more feature which we will discuss later.

Evaluated Techniques

A. Sleep to save

On the off chance that the mobile task scheduler has no work to execute it is consider being an ideal state. Sleep states are for the most part squandered the battery in the handheld devices [8], so on the off chance that you are in rest mode you without a doubt get the battery diminished status. So we are all that much pondering in the status of amid sleep mode. Another mode was ideal state mode this mode regularly amid period (aside from sleep mode). Secondary thing we are anticipating interrupts. Amid both the states interrupts on may happen. Interrupts has two sorts internal and external [4]. If some an external or unwanted interrupt comes, the processor module back to the running state. The processor spares more energy if it is said to be in sleep state rather than idle or running state.

B. Power Efficient Communications

We are anticipating power efficient. We need to works in the interfaces. Interface is only some basic component share the one thing which has similar set of attributes. The wireless network interface correspondence goes about as a manufacture in mechanism to permit user to perform dynamic power management [18].

A WINC card does not expend any energy in the power off state [17]. While the ideal state expends a lot of energy creating their amid time of transmission in the same sense we are searching for active state light of the energy amount was utilized.

C. GUI Design

GUI is only Graphical User Interface which acts as intermediate between the mobile phones and the users. GUI was most fundamental part during the time spent form a pc's and laptops and even like the desire of mobile (handheld) devices for the higher-lower convey from their mobile handheld devices [22].

Handheld gadgets are littler in size its utilization to collaboration among the applications. For the most part developers are utilized regular attributes yet fundamentally developers concentrated on the human factors and the applications. It has two types

1. Display
2. User interface

System Architecture

When all is said in done the MVC architecture can be utilized for the application, keeping in mind the end goal to permit great effective improvement and decrease inordinate coupling between the rationale and the user interface [9]. In this paper we implemented a methodology for a good user interface which can't be diminishing the utilization of more battery.

Our application has a good adaptability and we had utilized java eclipse and android development package which help us to develop a good application with great basic interface and this interface is further designed by eclipse's built in layouts and the format to be specific, battery_minder, list_item_icon, mode_selection and widget all are forced as xml file. The icon has been incorporated in res folder in which numerous drawable folders are made with the goal that it will be anything but difficult to change from platform to platform, As we made an target platform as form 4.2.2, it can likewise be change its icon measure according to we made a default as 64px[20].

We had developed a widget for our application with the goal that it can be easy to toggle our application enable and disable, as same as in application icons, we designed a widget so that it can have its size which is dependent to the display of the android gadgets the smallest pixel is 16px. We additionally included android private libraries and android dependencies [10]. We had likewise actualized a time picker management as android inclination and developed a package called com.tobyw2cypher.android in which we included utils.java, utilsConstant.java and debug.java, we additionally added to one more package named as com.tobyw2cypher.liionsaver in which we recorded as Batteryminder.java, Batteryseviceinfo.java, datatoggler.java, settings.java and togglewidget.java [14]. Furthermore we added to a data switcher package which helps the installer to check which version of android operating system is being utilized to be specific, Droid switcher, Gingerbread switcher and Ice Cream Sandwich switcher. Later the eclipse will naturally create a R.java file and buildconfig.java file like:

```
/** Automatically generated file. DO NOT MODIFY */  
package com.tobyw2cypher.LiIonSaver;  
public final class BuildConfig {  
    public final static boolean DEBUG = true;  
}
```

Other icons which are being utilized as a part of the application is stored in resource/ Crunch directory under drawable directory as we discussed about before, these crunch directory which has numerous iconic directories with API version in particular hdpi-v11. What's more, the Android manifest.xml file produced naturally in which the application setup and the target android gadgets and their version is being included which can be utilized while publishing or launching the application by sending out or utilizing the android tools for generating signed or unsigned application. Additionally we included a gradle wrapper property which gives the information like:

```
#Tue Oct 08 18:04:48 SAST 2013  
distributionBase=GRADLE_USER_HOME  
distributionPath=wrapper/dists  
zipStoreBase=GRADLE_USER_HOME  
zipStorePath=wrapper/dists  
distributionUrl=http://services.gradle.org/distributions/gradle-1.8-bin.zip
```

Also, we tested our application in ant debugger and performance and validation testing and usability testing is being tested with jubula it is the eclipse based testing.

Feature Selection

At first, there are a few potential features which we had would have liked to implement, may at present consider to later on. We had implemented numerous features in regards to sparing the battery life over the mobile data and Wi-Fi which incorporates:

1. Battery Minder
2. Setting
3. Battery Service Information
4. Data Toggle
5. General Receiver
6. Main Process
7. Mode Select
8. Benchmark and Comparison

Battery Minder modules includes kernel wake lock and CPU stats and figure 1 shows the screen shot of the module:

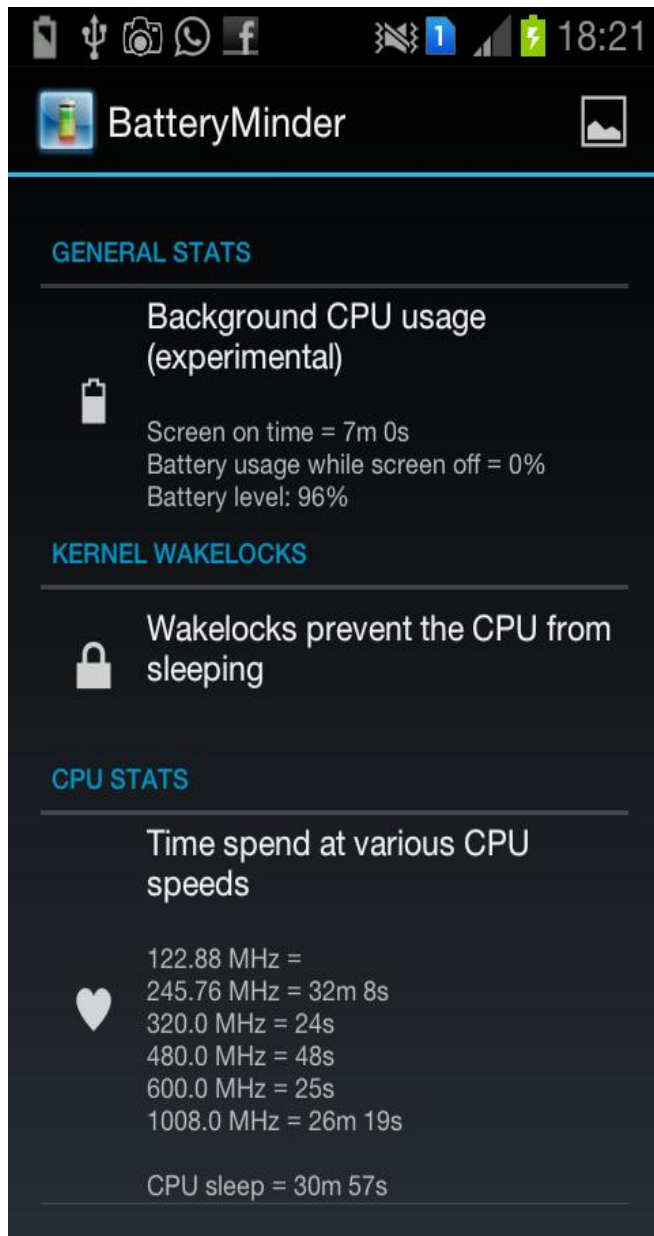


Figure 1. Battery Minder

The above module named battery minder is built with numerous packages which incorporate Byte Array Output Stream, Timer, Content Intent, Resolve Info and Android OS Bundle. Few user characterized packages developed such as Preference Manager, Log Manager and Activity base which manages widgets.

In this module we set the default value for wake lock as 60*30 by means of seconds in an exact manner, which is being reached out from battery INFO which will just show the wake lock of 30 minutes. In this module we outlined a save instance state where the CPU state is being spared to a log and displayed by means of list to the user as "Time spend at different CPU speed". The set content view is designed as a layout to display the aggregate battery status and the present time is displayed utilizing the built in function `System.currentTimeMillis() * 1000` which shows the raw real

time, in the event that it comes up short in showing the local time it utilizes an attempt work and shows by executing from uptime by part as [1]. Getting the CPU stat is as straightforward as getting the time where it just gets the string as `cslog` and displays the CPU stat.

CPU stat is built with following features:

1. CPU frequency
2. Time Format
3. CPU stat message
4. CPU Sleep
5. System Uptime
6. Wake Lock Info
7. Battery Usage
8. Error Report
9. Power Usage Info
10. Power Usage Intent.

The CPU status is being shown with RAW format by executing the implicit `time_in_state` procedures and characterizing these RAW information as Long data type by passing the data by multiplying it with 100001 and incrementing it with uptime by dividing it with 1000f (Frequency) which is formatted as Mega Hertz with time format as showed in fig 1.0 battery minder. The slip by timing is shown by getting the worth from system uptime.

"Wakelock" is an instrument of power management service in AndroidOS, which can be utilized to keep CPU awake (Partial wakelock) and keep the screen on (Full wakelock) [13]. A wake lock is basically used to secure the gadget an "awake" state, in which the CPU will be on, and the screen may or may not be on. It is unrealistic to do these tasks without a wakelock if the phone is in sleeping otherwise, as then the CPU is likewise in sleep mode. Nonetheless, if the user is utilizing the gadget for something else, and user's app is out of sight, you can do these tasks without a wakelock. Verging on everything user doing is battery intensive (sensors, Wi-Fi, wakelock) and user ought not to do it again and again so that the user don't degrade the user's battery life.

Getting the status of wakelock is minimal precarious which needs a root authorization from the third party tool example super user which gets the wakelock access. The figure 1 demonstrates the information of android which is not rooted where it doesn't show the wakelock stats. While in default this application will just show stuff that is wakelocked for over 30 minutes. In this way the wakelock is utilized here to prevent the CPU from sleeping). The time format is shown by giving back the values as micro seconds in turns as return ($(\text{microsecs} - 500) / (1000 * 1000)$).

The time displays is a String which annexes days by simple logic as $\text{seconds} / (60 * 60 * 24)$ and the utilization as $\text{hour} * 60 * 60$. By suffixing it with hours and for minutes it does likewise by barring hours and simply multiplying with 60 seconds and suffix it as minutes.

Setting modules permits the user to enable and disable the application and other time picker management tools which help us to enable and disable the wanted alternative for the gadgets for later utilize:

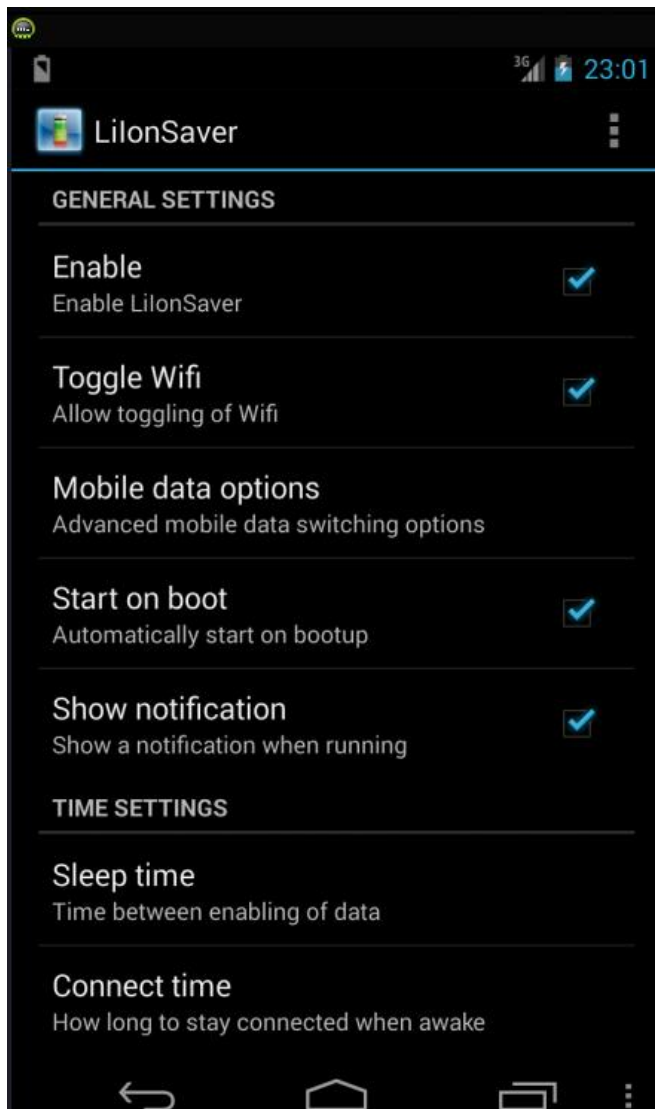


Figure 2 Setting

The Setting module is being implemented by importing the rundown of packages such as, Preference manager, Shared Preferences, Context and Toast which is shown in figure 2. Before implementing the general settings we implemented few core elements such as setting the default values to troubleshoot night mode as 'false'. Also, Night mode core settings with two main functions of it as start and end. Least awake time is situated as last with the goal that it begins at start up. The few core elements to the display properties is being set are awake time, sleep time, default awake period, default sleep period and run version.

Here it utilizes a Hash map package for deterministic emphasess HashMap is an execution of Map. Every single discretionary operations are upheld. All components are allowed as keys or values, including null. The cycle request for HashMap is non-deterministic. In the event that you need deterministic emphasis, use LinkedHashMap. This strategy is referred from the android site (<https://www.developers.android.com>).

The execution of HashMap is not synchronized. In the event that one thread of a few threads accessing an instance alters the map basically, access to the map should be synchronized. A structural modification is an operation that includes or evacuates an entry. Changes in the value of a passage are not structural changes. The Iterator made by calling the iterator technique may throw a Concurrent Modification Exception if the map is structurally changed while an iterator is utilized to repeat over the components. Just the remove method that is given by the iterator permits to evacuation of components amid iteration. It is unrealistic to ensure that this system lives up to expectations in all instances of unsynchronized simultaneous adjustment. It ought to just be utilized for investigating purposes.

To flip the gadget these are few packages has been utilized to associate with the android core elements, for example, Pending intent, Context, Intent, Shared Preferences, Remote Views and Widget Providers. At first setting the parameters for gadget with basic flip by acquiring the App gadget supplier which is not showed in the setting menu this is the reason it utilizes a remote views.

Battery Service Information which gives the information about the current condition of the battery. The module utilizes few built-in and user defined packages such as Map as we given the brief portrayal about mapping in the setting area, Timer, IBinder, Sparse Array. Here we fabricated with few elements which manages the information in the details including the previously saved data furthermore incorporate just the keep running following the last time the gadget was unplugged in the stats.

It likewise offers the user with the fundamental element as screen on type where the screen brightness is default set to diminish to spare the energy furthermore offers the user with alternatives as dim, faint, medium, light and splendid. The utilization of sparse array in this application is to gather the User Id status and display it to the user in the orderly manner by throwing it as the exceptions. This component likewise incorporates Battery Uptime where the application computes Real-time battery stats and screen on time and discharge time and level of the android gadget.

Data Toggle offers the smooth and simple to customize alarms, Notifications, Broadcasting, Wi-Fi Manager, Async Talk, Mobile Data toggles and screen service modes. Switches between auto synchronization choices in android, checks the screen services in case it was being killed. The principle usefulness is to turn the data on when the system is woken up, it skips to turn on the data if the service is in Airplane mode and it automatically disables the data services. Its functionality to run the notification always to invoke the user by updating the status of the current screen time, type and mode which is being activated. It can likewise handle the broadcasting features from the gadget and empowers either the Wi-Fi or mobile data and saves the current state by clearing the past notices consequently. It sets the flag if any synchronization ought to run while checking for network connections once it connected it starts the power sync features of android gadget. Numerous android users don't trust android gadgets to reliably notify of data connection so we introduce a manual check options with proper timeouts.

General Receiver is not as of now utilized, for this to work, there must be a long living services as Android, which would not like to processes when the screen turns off or on. General receiver gets the broadcast, check for the wake or sleep of the screen. On the off chance that screen on data is empowered, switch data off when screen goes to sleep and indenting to switch off the data with no postponement. It utilizes a timer to switch off the data furthermore it has the capacities to plan the sleep alarm and notifies the screen is on and the data is empowered. It likewise includes the popup message at whatever point the android system or gadget is booted up, it additionally notifies the user when the system is being changed or connectivity is changed with alternatives in it as begin sync with timer or sync just once. General receiver notifies when the gadgets being plugged in to AC power cable charger or USB charger and advises in the event that it's expelled from the charger.

Main Process, the thought of this module is referred from the one of extraordinary battery sparing application greenify. The fundamental functionary of this primary module is to handle the various capacities and components of the application. This primary procedure incorporates numerous utilities, for example, Data Format, Alarm Manager, Notification Manager, Pending Intents, Resolver, URI, APN switcher, Mobile Data Switcher and other built-in packages. It by and large begins with general receiver as an data state of period 24 hours.

Recently it begins the scheduling process by utilizing an android scheduler, and cancels every one of the synchronizations and let the gadget realize that the application has been begun and empowers the Li Ion saver application. In the event that the user has been incapacitated fiddling with the mobile data, restores APN type else it empowers the mobile data and switch the APN. The fundamental procedure does set up the default flags such as mode selections, alarm managers. For the Night mode there is no requirement for typical data alarms and it verifies whether the data is on. It stops the scheduling process and re-awakens the data and does likewise transform once more.

At the point when the alarm goes off, we need to broadcast Intent to our Broadcast Receiver. Here we make an Intent with an explicit class name to have our own receiver recipient (which has been published in `AndroidManifest.xml`) instantiated and called, and afterward make an Intent Sender to have the intent executed as a broadcast, this Intent Sender is configured to permit itself to be sent various times. It offers the user with numerous valuable elements as setting the sleep period, booking the wake alarm, setting up the night mode alarms which will be configured as simple data format which has been added unequivocally to the application amid the outlining it.

Mode Select the package is planned with the assistance of Android studio with open source icons it displays the user a list of modes in a list view as demonstrated in figure 3.

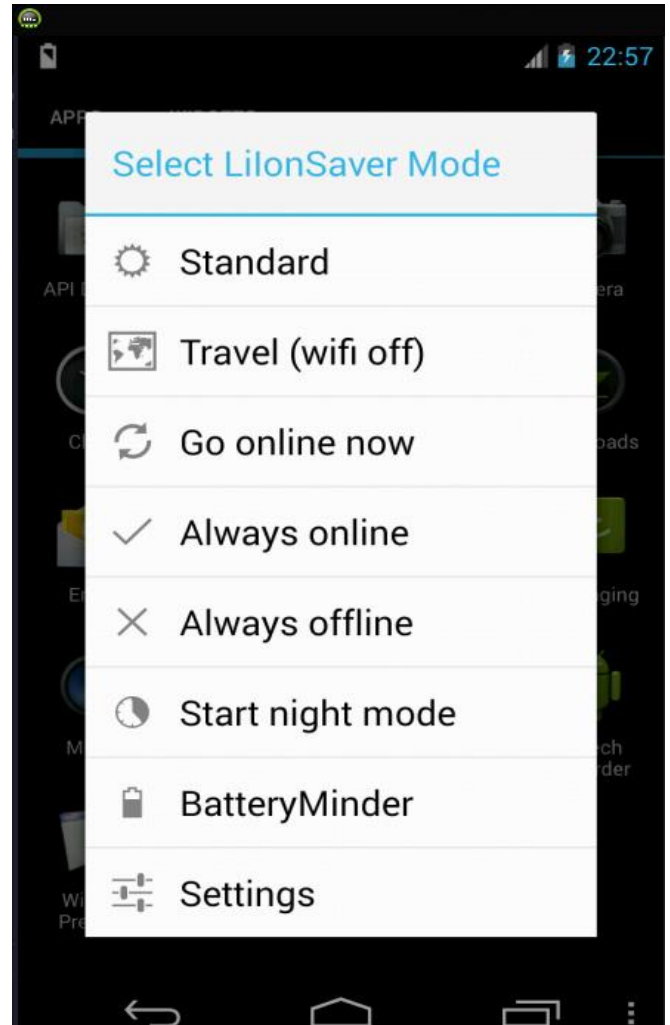


Figure 3 Mode Select

What's more, different modules are mode selection in which we can choose the modes like standard and night mode. In night mode the application will automatically toggle off the mobile data and Wi-Fi services and we likewise offered the sleek setting GUI so that the user can without much of a stretch view the present status of their battery furthermore the restart period of the smart phone.

Bench Mark and Comparison this segment is to give the better aftereffects of performance and energy effectiveness of the android gadget with and without Li Ion saver application installed in it. In Fig 8.0 it unmistakably shows the evaluated battery existence with the observed results. It shows the general normal standby time elapse by both in 100% and present condition of the battery. It additionally shows the evaluated time for getting depleted of the battery incorporates 3D, 2D Games, Online and offline videos, Internet over Wi-Fi, Internet over mobile networks, different broadcastings. It reliably changes in time when the mode is being chosen, for example, night or travel mode. This likewise incorporates battery minder as we given complete information about the utilization and the feature of battery minder utility in the past segment.

Li Ion Saver Mode Enabled













Statistics	Estimates	Use Times	Calibration	Cor
Use types		100%	78%	
	Average standby	2d4h	1d17h	
	Average use	3h43m	2h54m	
	3D Game	2h24m	1h52m	
	2D Game	4h27m	3h28m	
	Online Video	2h44m	2h07m	
	Online Audio	4h52m	3h47m	
	Local Video	3h19m	2h35m	
	Local Audio	7h05m	5h31m	
	Internet Wi-Fi	4h27m	3h28m	
	Internet Mobile	3h54m	3h02m	
	SMS	3h42m	2h53m	
	Talk	7h48m	6h04m	

Figure 4 Bench Mark for Battery life

In the comparison area we tested our android software based approach in two devices. Figure 4 and Figure 5 shows the result of this tested in Asus Zenfone.

Li Ion Saver Mode Disabled

Statistics	Estimates	Use Times	Calibration	Cor
	Global Estimate	4h23m		
All-time average:		4h13m		
Average since (un)plug:		7h37m		
Battery (dis)charge:		1h50m		
Average Times:		78%		
AC charge:		4h32m	1h00m	
Wireless charge:		n/a	n/a	
USB charge:		10h31m	2h19m	
Discharge (100->0%):		5h25m	4h13m	
Last Measured:				
Actual battery available:		78%		

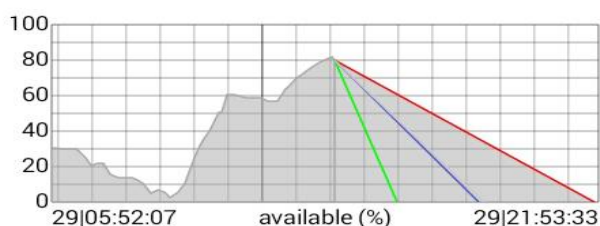



Figure 5 Battery saver mode disabled

In disabled mode the battery get deplete so early, from the figure 5 battery saver mode disabled it obviously shows the evaluated time of the energy. It includes Global estimation as some hour took after with minutes furthermore shows the average times by means of percentage.

From the figure 6 battery saver mode enabled which shows the time estimations of present and previous services or history of last battery saved states and measured the overall battery life for a certain time or days or week tops, illustration from 23:00 to 14:00 it spared few hour of battery life representing it in a graph with date and time emboss inside of a UI.

Li Ion Saver Mode Enabled

Statistics	Estimates	Use Times	Calibration	Cor
	Global Estimate	5h08m		
All-time average:		5h01m		
Average since (un)plug:		9h42m		
Battery (dis)charge:		2h38m		
Average Times:		77%		
AC charge:		4h32m	1h00m	
Wireless charge:		n/a	n/a	
USB charge:		10h31m	3h08m	
Discharge (100->0%):		5h25m	5h11m	

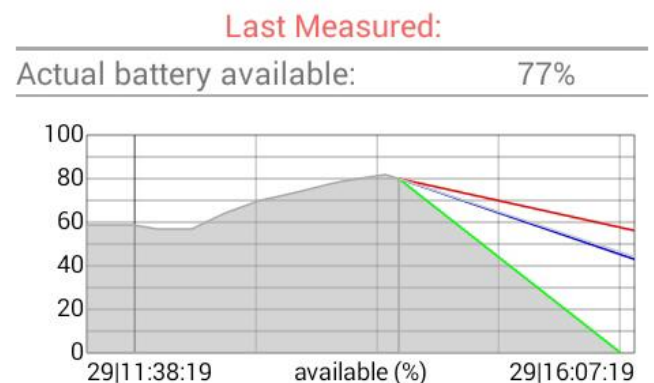


Figure 6 Battery Saver Mode Enabled

Conclusion

Final conclusion of the paper was diminished the undesirable battery utilization for the users in mobile devices (handheld devices). A considerable lot of the organizations concentrate on hardware implementation like size of the battery expanded furthermore hardware designs are complicated because of that. Really we have produced the solution in the software methodologies. In that application works based on the user time synchronizing so auto process conveys the work because

of that a few activities are working beyond that then battery life increased. We implemented power synchronization method which automatically syncs an online application in particular whatsapp, other email applications, and so forth.

References

- [1] Energy Management Techniques in Modern Mobile Handsets 1553-877X 2013 IEEE.
- [2] A Survey of Software Based Energy Saving Methodologies for Handheld Wireless Communication Devices
- [3] A. Vahdat, A. Lebeck, and C. S. Ellis, "Every joule is precious: The case for revisiting operating system design for energy efficiency,".
- [4] Energy Management Techniques in Modern Mobile Handsets Narseo Vallina-Rodriguez and Jon Crowcroft, Fellow, IEEE 2012.
- [5] S. Schroeder, "Android Overtakes BlackBerry As the Top U.S. Smartphone Platform," Mashable Tech, retrieved March 1, 2011. Available: <http://mashable.com/2011/03/08/android-smartphoneplatform-report/>.
- [6] J. O'Dell, "Android Wins in the U.S. Smartphone Wars," Mashable Tech, retrieved March 1, 2012. Available: <http://mashable.com/2011/03/03/nielsen-android-winner/>.
- [7] Android Open Source Project, retrieved March 1, 2012. Available: <http://source.android.com/>.
- [8] J. Flynn and M. Satyanarayanan, "Energy-aware adaptation for mobile Applications," in *Proceedings of the seventeenth ACM symposium on Operating systems principles*. New York, NY, USA: ACM, 1999, pp. 48–63.
- [9] Flynn, Jason and Satyanarayanan, M., "Managing battery lifetime with Energy-aware adaptation," *ACM Trans. Comput. Syst.*, vol. 22, pp. 137– 179, May 2004.
- [10] D. Ferreira, A. K. Dey, and V. Kostakos, "Understanding Human- Smartphone Concerns: A Study of Battery Life," in *Proceedings of the 9th international conference on Pervasive (Pervasive'11)*, pp. 19-33, 2011.
- [11] J. Zhang, "Linux Kernel and Android Suspend/Resume," retrieved March 1, 2012. Available: <http://kzjblog.appspot.com/2010/11/20/suspend-en.html#sec-6>
- [12] M. R. Jongerden and B. R. Haverkort, "Which Battery Model to Use?," *IET Software*, Vol. 3(6), 2009, pp. 445-457.
- [13] A. Roy, S. M. Rumble, R. Stutsman, P. Levis, D. Mazières, and N. Zeldovich, "Energy management in mobile devices with the cinder operating system," in *Proceedings of the sixth conference on Computer systems*, ser. EuroSys '11. New York, NY, USA: ACM, 2011, pp. 139–152.
- [14] N. Vallina-Rodriguez and J. Crowcroft, "ErdOS: achieving energy savings in mobile OS," in *Proceedings of the sixth international workshop on MobiArch*, ser. MobiArch '11. New York, NY, USA: ACM, 2011, pp. 37–42.
- [15] E. Oliver and P. S. Keshav, "Data Driven Smartphone Energy Level Prediction," *University of Waterloo, Technical Report CS-2010-06*, 2010.
- [16] Power Management, Android Platform Development Kit, retrieved March 1, 2012. Available: http://www.netmite.com/android/mydroid/development/pdk/docs/power_management.html
- [17] Pathak, A., Hu, Y.C., Zhang, M., Bahl, P., Wang, Y.-M.: Enabling automatic offloading of resource-intensive smartphone applications. In: ECE Technical Reports, Purdue University, TRECE- 11-13 (2011).
- [18] Priyantha, B., Lymberopoulos, D., Liu, J.: Little rock: enabling energy-efficient continuous sensing on mobile phones. *IEEE Pervasive Comput.* 10, 12–15 (2011).
- [19] Chamodrakas, I., Martakos, D.: A utility-based fuzzy TOPSIS method for energy efficient network selection in heterogeneous wireless networks. *Appl. Soft Comput.* 11, 3734–3743 (2011).
- [20] C. Shepard, A. Rahmati, C. Tossell, L. Zhong, and P. Kortum, "Live-Lab: measuring wireless networks and smartphone users in the field," *SIGMETRICS Perform. Eval. Rev.*, vol. 38, pp. 15–20, January 2011.
- [21] S.-L. Tsao and C.-H. Huang, "Review: A survey of energy efficient MAC protocols for IEEE 802.11 WLAN," *Comput. Commun.*, vol. 34, pp. 54–67, January 2011.
- [22] Carroll, A., Heiser, G.: An analysis of power consumption in a smartphone. In: *Proceedings of the 2010 USENIX Annual Technical Conference*, p. 21. Boston, MA, USA (2010).
- [23] Rozner, E., Navda, V.: NAPman: network-assisted power management for WiFi devices. In: *Proceedings of the 8th International Conference on Mobile Systems, Applications, and Services (MobiSys '10)*, pp. 91–105. San Francisco, CA, USA (2010).