Different Levels of Energy and Power Management Systems in Embedded Devices

Min, Xiaoyu
Aalto University, School of Science

Abstract—The number of embedded devices enjoys a rapid increase in recent years. The performance of the devices increased much, while the development of power supplies, especially batteries, reaches a bottleneck. Therefore, the energy and power management (EPM) systems become a more important part in the systems. EPM systems could affect the whole system from application level, to operating systems level and hardware level.

I. INTRODUCTION

The concept of Internet of Things (IoT) has evolved in the past decade due to a convergence of multiple technologies. In the process of the evolution, sensors play an important part, since they are the 'eyes' of the various mobile devices in the IoT. Due to the bottleneck reached for the batteries or other power supplies, energy and power management (EPM) systems have become an important part in the embedded devices that require strict power control on the systems.

Most of the nodes or devices in the IoT network have limited hardware resources, e.g. a tiny micro-control chip with only a few sensors attached to the chip. To make the small system operate more energy-efficient, the energy and power management is required to strictly manage the power consumption of each part of the hardware, either to turn some part of the device into sleep mode when not in use, better schedule the running state of each part of the system, or even adjust the currents running on the circuits, so that the whole system could operate normally with less power consumed. [1]

This paper will try to go through the energy and power management systems in different levels in common embedded systems, and try to summarize the advantages and disadvantages of the techniques in these EPM system.

In section II, application-level EPM is introduced, and the paper would take one of the most famous embedded software platform, Android as an example, to show the impact of EPM on application level. In section III, EPM systems in operating system level and hardware level are introduced, including the ACPI, Linux EPM and a brief introduction to hardware EPMs. Summary and comparison of the different level and the future scope will be included in the conclusion section.

II. APPLICATION-LEVEL EPM: TAKE ANDROID-BASED APPLICATIONS AS AN EXAMPLE

The design principles of applications running on the embedded devices could of course have a grate impact on the power consumption of the device [2]. But the impact is strict relied on how much permissions are given by the operating systems [3]. If the application is not granted much permissions accessing the EPM systems of the OS, it might be running under grate restriction and therefore have less impact on the whole device [4]. The open-source operating system Android delivered by Google is popular in many different platforms and provides several EPM APIs to the applications.

Android applications try to affect the system EPM via the Android Wake lock mechanism. Android implements an application framework on the top of the kernel called Android Power Management Applications Framework. The Android PM Framework is written in Java which connects to Android power driver through JNI. Through the framework, user space applications can use the class 'PowerManager' to control the power state of the device. [5]

Figure 1 shows the process of how an application could apply and control the power state of the device. It not only shows the effect of application level, but also indicates that management of power system is a vertical process through the application front end to the hardware back end.
III. OS-LEVEL EPM

As stated in the previous section, Android OS offers a kind of Power Management APIs to the applications, so that the applications could have access to control of the state of the device. However, the control process is done by the OS through the communications between the Linux kernel and low-level hardware [6]. Similarly, the operating systems running on embedded devices could have done a lot to control the power state or schedule of resources. Figure 2 shows the power management relationship between the Windows Phone OS and the low-level hardware. The major connection of EPM control are the Advanced Configuration and Power Interface (ACPI) functionality.

Fig. 2. Windows Phone Power Management System [7]

ACPI is an open standard for device configuration and power management by the operating systems, which replaced the previous standard Advanced Power Management (APM). It takes the EPM control to the operating systems from the previous BIOS-central system. Several power states are defined by ACPI for the whole system, and some running states for the CPUs are optional defined as an extension [8]. It enables the operating system to choose which state to step in, so that the system could be turned into different power states, for example low power state or sleep state, and better save and manage the power [9].

Compared to the more generic ACPI standard, Linux EMP provides more freedom to the developers and architecture designers, since the kernel could be re-configured in modules, according to different specific requirements [10]. The Linux EPM is a large sub-system in the Linux kernels, including Power Supply, Charger, Clock, Frequency, Voltage, Suspend/Resume and other aspects. Figure 3 indicates a brief connections between these Linux EPM modules.

Fig. 3. Linux Energy & Power Management System [8]

IV. HARDWARE LEVEL EPM

Besides the OS-controlled EPM, some hardware of the embedded devices also have their hardware EPM on the chip – the Power Management Integrated Circuit (power management ICs or PMICs). These PMICs could also be a system block in a system-on-chip system are used for managing power requirements of the host system. PMICs normally exist in battery-operated devices such as mobile phones and portable media players, and they are often integrated with the micro-controller chip. Therefore, the product manufacturers would often buy a complete PMIC solution with the micro-controllers from the chip manufactures, since it is fast and cheap, and does not require much research on the hardware level, unless they have some specific requirements. [11]

A PMIC may consist of different sub-modules, for example, they may have Battery Management module, Voltage regulations, DC to DC conversion (dynamic voltage scaling), Power sequencing module, and other Miscellaneous functions modules. Most of the design of PMIC are patterned or protected by the chip manufacturers, and it takes a bunch of resources and funds to do the PMIC design and implementation. Thus, only part of the specifications of the PMIC could be found on the Internet.

V. CONCLUSION

The article walks through the EPM systems in different levels in the embedded devices. There is no doubt that each level could effect the state of the whole system. Both software and hardware could have an impact on the EPM system. The applications running on the embedded devices could have access to the low-level EPM systems if EPM APIs are provided by the operating system. The operating system is the major controller of the EPM system of the whole device. Low-level hardware EPM system could somehow be running on their own without support from the operating systems, dynamically adjusting the status of the circuits.
Currently, most of the EPM controls are done by the operating systems for most of the embedded devices. The applications designed should be optimized for each OS, since different operating systems employ different EPM policies. The hardware level EPM systems on the other hand, are not exposed to the developers at all. These different levels of EPM on the embedded devices are closely related with each other. If all these levels could be designed and configured properly and coherently, the power consumption of a embedded device might be able to reach an ideal level.

REFERENCES