International Journal of Intelligent Information and Management Science
ISSN: 2307-0692, Volume 8, Issue 1, February, 2019

Development and Application of Embedde d System Driver based on BREW Platform

Hao Liang, Hua Wu
Department of Information Science and Electrical Engineering, Shandong Jiaotong University, Ji'nan, 250357, China

Abstract: An important application field of embedded system is the development of system driver, which promotes the development of telecommunication industry by taking mobile phone as the entity. The characteristics of the mobile phone is limited capacity, embedded devices in a kind of popular demand is higher, the market will emphasize the capacity of the mobile phone, durability and convenience, the characteristics of the embedded application software will be at a certain time to access the bottom part of the firmware, also to different degree of control of the hardware, it's potential to improve the driver's request, how to solve the problem of hardware resource environment this is particularly important. This paper focuses on the exploration and discussion of this part, and deals with a series of problems existing in BREW platform's embedded system, making the platform have a qualitative leap in terms of speed, efficiency and quality, mainly focusing on the two important issues of graphics and image processing and memory management. Many BREW applications are related to the drawing and processing of images. However, the processing speed of ARM chips is too slow, especially in the field of games, which requires a lot of speed. The speed of screen drawing affects the user's experience. In this paper, BREW is introduced, how to optimize the image resources on the BREW platform, BREW memory management research and optimization, combined with the actual situation analysis, so as to seek the optimal solution.

Keywords: BREW; Embedded; Driver program; Graphic image processing; Memory management

1. Introduction

At present, with the ever-changing changes of communication equipment and the ever-expanding software requirements of system drivers, higher requirements have been put forward for software development. First, the service functions are more complete than before to meet the needs of different groups of users. Second, the software has reusability, the high efficiency of software development; The software has universality, using interface programming, easy application in different companies, models, manufacturers of communication equipment; Third, the software has the characteristic of tailoring, which makes the software customizable and personalized. Fourth, the software can be expanded, easy to upgrade software; Fifth, software should be easy to maintain.

2. Research Background and Significance

In view of the above situation, a common platform for communication applications has emerged in the market one after another [1]. In which SUN microsystem's J2ME platform, as well as the BREW platform of qualcomm, is the main representative. The former programming language is Java, but because Java bytecode itself exists in the mobile terminal running slowly, and the integration of J2ME is more difficult, so gradually eliminated in the development process, but qualcomm's BREW platform in the United States is slightly dominant in this respect. BREW's programming language, C++, is compiled into binary code before installation and is fast. Moreover, BREW and mobile terminals are simple and practical. The advent of BREW has made it easier to use drivers on different types of handheld devices.

3. Introduction of the BREW

BREW is a common interface platform provided by qualcomm for driver execution and development. It is convenient to use and can better serve users' software download. BREW provides an application execution environment (AEE) with a number of advantages, such as high efficiency, low cost, and scalability. Focus on the development of convenient and fast implant into any handheld device, and simple application driver. Traditionally, people have used high-speed ASIC (special-purpose integrated circuit) technology to implement new application functions, both high-end and low-end devices, using more functions for seamless integration. Today, BREW offers a BREW environment that functions like a regular piece of software, allowing users to download a specific type of software for use on their mobile devices. It also offers a range of business functions via a BREW - owned interface. At present, BREW has been widely used in navigation and positioning, bluetooth, and telephone services.

BREW is based on a standard on wireless devices and is an open platform for application execution. BREW features: the first is the small size, compared with other drivers, the use of BREW to many times smaller; Secondly, it is fast, because it USES C++ as a programming language, using unique chip system software, making the integration more simple and fast; It also has the advantage of being open. In addition to C++, BREW also supports other languages for a variety of environments. Because it can be found on smartphones running mobile operating systems such as Palm, applications written for these systems can be downloaded wirelessly using the NREW distribution system (BDS) and, like BREW, commercialized. Similarly, it also has the characteristics of extensibility. BREW extension is convenient and can realize additional functions more quickly. Finally, it has high benefits by reducing development costs and thus reducing equipment interview time.

4. BREW Driver Design in the Image Processing Technology

In the design and production process of large-scale games, most of the time is spent on the drawing operation of the screen, which affects the user experience and greatly tests the speed of the game, accounting for 80% of the running time of the game. Next, the BREW graphical user interface is introduced. In combination with the practical situation, a certain research is conducted on the processing of resources and the drawing speed of images [2].

4.1. BREW graphical user interface

Today, the graphical user interface (GUI) is an important part of almost all interactive application drivers. Graphical user interfaces (guis) are important, especially for micro devices. The limited space on the phone means that a simple, usable interface is important. BREW provides a very comprehensive toolkit with all the basic controls needed. There are three main interfaces: IDisplay, IGraphics, IImage, and IBitmap and IDIB. About IDisplay and IGraphics interfaces, the former is mainly applied to the drawing operation of the screen, while the latter is the supplement of the former function, mainly making the picture drawing more realistic and combining with the 3D effect. If you're just doing a simple drawing operation, just use the former. The IDisplay interface is used to draw partial patterns, text, bitmaps and some simple geometric shapes. Since all applications currently use a simple IDisplay interface, the system automatically creates an IDisplay object after the application is created, which makes the application more convenient and more practical. The IGraphics interface is much richer, mainly in view switching and clipping. This interface can support dual buffering technology in the future, which means that the program can perform drawing operations directly on the double buffering of the current screen.

However, the drawing of graphics is far from enough to meet the speed requirements in real situations. The IImage interface solves this problem. This interface is mainly used to display images and draw images, which have a lot of details on the processing. Specifically, each part of the IImage interface is associated with a bitmap stored in a resource file created by the BREW resource editor. BREW provides interface environment support for image formats in the default environment: PNG, Windows BMP and BCI. With respect to the IBitmap and IDIB interfaces, it is possible to manipulate bitmaps and convert them to device specific bitmaps when loading images that are device independent. Only in this way can the image processing be faster.

4.2. BREW image resource optimization processing

At present, both BMP and PNG formats can support BREW. There are many ways to draw pictures with BMP, but only IIMGE_Draw can be used to draw pictures with PNG. BMP is a native image format, which occupies a large volume, while PNG is a lossless compression format for images, with a small volume. In addition, because the internal image rendering is BMP, a certain format conversion is required, which makes the speed of PNG slower than that of BMP. In addition, all SDKS above BREW2.0 have IUnzipAStream interface. We can use this interface to compress the file into zip format and use IUnizipAStream to decompress it in the program. There are three resource optimization methods: PNG image conversion to BMP, BMP image inversion, BMP image compression and decompression.

The first method is to convert the PNG images into BMP format by using the IIMAGE_Draw interface function to draw the PNG images, complete the image format conversion, and finally set the pointer of the drawn display buffer to NULL, so that the screen will return to the default memory address of the system, and continue to draw things on the screen. The second, BMP format image inversion, using BREW to provide IDIB interface control BMP format image internal pixels. IDIB inherits all the member functions of this interface, so it can be cast to an IBitmap type, including Shared data members, by type conversion. About the compression and decompression of BMP images, there are mainly GZIP compression files in BMP format, and the compressed file is changed into the suffix of. BMP, which is added to the resource as images to complete the task.

5. BREW Driver Design in Memory Management Techniques

In large capacity such as computer or server code, we don't need to pay attention to internal details such as CPU usage, but relatively speaking, this kind of resources are

relatively limited in most of the BREW equipment, so be sure to learn more about the BREW of memory management technology, and adopt some methods and techniques to reduce the unit of time for the consumption of memory [3].

The BREW program's size is limited by the available free file system size and the available register size. When BREW is executed, the resource is first loaded and stored in Heap RAM. The remaining space is used to perform other operations, such as memory allocation, resource loading, and control creation. The amount of Heap RAM is determined by the individual (cell phone model and configuration), and there are certain algorithms for space management, such as program segmentation, movement, and fragmentation integration.

At present, because the storage space of handheld devices is not large, so in the device driver to take certain protective measures, how to reasonable BREW memory allocation is particularly important.

First of all, we should use variables skillfully, try to use short word length and use bit marks. Second, we should pay attention to memory alignment. If structure A occupies 16Byte of memory, while structure B occupies 20Byte of memory. Also, reduce dynamic memory allocation. In the operating system, the allocation of memory corresponds to different algorithms, so it takes a long time to find the memory block and allocate it to the user. Therefore, we need to reduce the amount of memory allocated as much as possible. The solution is to use array pre-allocation instead of dynamic allocation of memory, so that the location can be clearly visible. In addition, you can also use the buffer pool method, the specific method is in the driver before the execution of the memory pre-allocation, and then with the pointer to track the location. You can also add some debugging information, the advantage is in the debugging process can also save time. Again, in memory allocation, it is important to check for NULL memory states and unnecessary allocations. At the same time, do not force allocation of unreasonable chunks, reduce the memory fragmentation. Since BREW's memory is extremely valuable and memory allocation is a slow process, any random memory movement or change will result in the generation of internal fragmentation. For memory management, there is a more reasonable method is to load and unload resources, the use of the interface should be immediately released, so that it can be in the appropriate circumstances to free up unwanted memory.

6. BREW Performance Optimization

As for BREW performance optimization, it is mainly reflected in the performance of calculation, memory occupation, program startup time, and the size of the program executable file. The main problems solved are still CPU resources and memory space [4].

First, we can estimate the CPU resources and time consumed by each piece of code in advance. In this case, the BREW helper function GET_TIMEMS() is used for the estimation, and this function can obtain the current time of the system in milliseconds, so as to properly select the memory space for allocation. Secondly, in the development of a driver, must be clear about the performance requirements of the program, but at the same time to take into account the requirements of other programs, choose a compromise solution, determine the optimal driver development strategy. Third, to understand what the current gap is, the gap between the current and the expected to make some estimates. Then, you have to analyze the performance bottlenecks of the finder, usually by writing your own review code. Finally, take appropriate steps to optimize the process, and then repeat the process until the required metrics are met.

In short, the development of the embedded system driver based on the BREW platform occupies an important position in the handheld devices, and it is difficult to achieve the advantages of other platforms. The research on BREW will continue, which requires developers to master more technologies under the conditions of BREW's principles and SDK components, so as to make excellent embedded application driver based on the BREW platform.

References

[1] Gao Li, Shi hao, Zhang hui. Establishment and implementation of platform stability control system based on embedded technology. Electronic fabrication. 2019(01), 48-49.

[2] Luo Zhijuan. Research and application of embedded system driver. Journal of changsha aviation vocational and technical college. 2012, 12(04), 62-66.

[3] Jia Xiaojing, Li yiyou, Jia lixin. Development of embedded system network driver. Journal of henan institute of education (natural science edition). 2009, 18(03), 28-31.

[4] Fang Haoshuai, Zhu jie, Li nan. Teaching discussion on embedded system device driver. Journal of north China university of aeronautics and astronautics. 2016, 26(04), 20-22.