**Abstract**

An embedded RFID (Radio Frequency Identification) middleware based on ARM Linux operating system and S3C2410 hardware platform was designed and implemented. The system architecture and platform of software and hardware of this middleware were introduced. The key technology applied in some device management and tag data processing and the application of SQLite were specialized. Since its rich standard interfaces offered by the hardware platform, the middleware can connect with different kind of readers from different manufacturers without considering the diversity and complexity of RFID bottom layer hardware. So, it can supply upper layer application of system with a powerful, uniform platform, and lay a broader, richer foundation for the application of RFID.

© 2012 Published by Elsevier B.V. Selection and/or peer review under responsibility of ICMPBE International Committee. 
Open access under [CC BY-NC-ND license](https://creativecommons.org/licenses/by-nc-nd/).

Keywords: Embedded RFID Middleware; Equipment Management; ARM; Linux; SQLite

**1. Introduction**

RFID is a kind of software between RFID middleware reader hardware and software application, the bond to connect the RFID tag reader with application program, the core of RFID application system, the application can access the RFID tag data through the general application program interface (API) provided, and whether the application or front reader change, which will not affect the other end, and cut out the more complexity problem for the maintenance of the connection. At present, there are some RFID middleware products in the market, such as Microsoft, IBM, BEA, SUN and Sybase, these products are powerful, but which run on PC or the server software, management solutions are only focus on a small amount of common and fixed RFID equipment, the cost is (also) high. And in actual use, many applications need to run on different platforms in the environment of networks, for example a large
embedded hardware device ACTS will be used as the hardware platform, and software platforms in addition to Windows outside, still include Unix, Linux etc.; Some need to have the access conditions of supporting multiple network and various methods of communication; and some reader equipment works at the same time, which has to need reader equipment cooperative work for management, control and optimization of work and so on. So, the embedded RFID middleware emerge as The Times require that is suitable for various kinds of RFID reader equipment, which can integrated the commonly used RFID tag reader equipment and wireless communication equipment into the small scale, high level of integration with the platform, and can provide a unified power support and types of rich standard interface, and also support a variety of terminal and peripherals cooperative work. Embedded RFID middleware will play an important role in large scale application of RFID, based on the hardware high integration, which can quickly realized(realize) the advantage of software integration, hardware manufacturers can quickly upgrade series of product, which are effective actions to meet the enterprise marketing development and business scope requirement. This article introduced the method of the design and implementation of embedded RFID middleware and the key technologies been used.

2. Embedded RFID middleware and its key points of design

Embedded middleware is a kind of software located between the embedded system software (embedded operating system, embedded database etc) and application software, its function is to use the basic function and service provided by embedded operating system, and provide operation development environment for the application of the upper system. Compared with the traditional embedded middleware, the biggest characteristic is due to the limit of the embedded environment resources, which cause the middleware functions and embedded system structure to be more compact and high real time. Therefore, the development process of embedded RFID middleware is much different from that of RFID middleware development process, not only should you consider the limit of hardware and software properties, but also must make full use of the existing resources when designing system, and make the middleware to achieve the best performance, specific design points are listed as follows:

- Problem of power consumption

Embedded middleware is the software module connecting closely with the hardware, optimizing the embedded middleware power consumption is the important aspects which need to be considered for the design of embedded middleware. There are many papers which have analyzed how to optimize the power consumption of embedded system software: some analyzes how to optimize each parts of the operating system to reduce power consumption; in some articles, power control method is discussed from the compiler optimization and dynamic power control aspects etc; Some articles introduce how to reduce power consumption from memory management aspects, and some papers put forward a kind of model to analyze and calculate the power consumption of system, and illustrate several programming skills to reduce power consumption.

- Problem of real-time

Real-time problem refers to whether the response time of the outside news for the system can meet the requirements. From the point of software, designing(to design) real-time systems need to gain the
support from the real-time operating system, reliable scheduling algorithm and other agreements. So, it need(needs) us to consider the support for the real-time during planning for the design of the embedded RFID middleware, the corresponding control parameters may need to be added in interface definition, all kinds of operations support the multitasking etc.

- Problem of limited resources
  
  Cost control is bound to lead to the optimization use of resources. From the point of software, first, the use of resources is to have a complete control of resources, and second, design optimization algorithm and control the use of resources strictly.

3. Frame of Embedded RFID middleware and its platform design

3.1. Frame of Embedded RFID middleware

At present, popular operating system in the field of embedded system representatives are mainly Windows CE, ArmLinux and VxWorks, since these difference of operating system existing in kernel, so, the structure of the system framework is general the key problem of research and development of embedded RFID middleware. According to the Application requirements of RFID middleware in practical application, including reader management, data processing, the transmission, filter and storage and providing application program interface for upper. This paper designs the structure diagram of the RFID middleware embedded as figure 1 shows. The general system framework was constructed to seek a public kernel subset, and undertake unity encapsulation, which can provide the transparent system access for RFID middleware, and solve the abstract problem of operating system.

![Fig.1 The architecture of Embedded RFID middleware](image-url)
3.2. Design of hardware platform

Because the embedded system software and hardware can be cut, so after embedded system hardware is determined, which determine directly the compile and work efficiency of the software. The hardware platform design of embedded RFID middleware as figure 2 shows. The development board is composed of the core board and main board, the processor of which we choose is ARM920 kernel S3C2410 T made designed by Samsung, the frequency of which can reach 203MHz, the mainboard has expanded the RS232 serial interface, the etheric interface wireless networks interface, USB port interface etc. The ARM processor is one of the most widely used processor chips, low power consumption, low cost and high performance characteristics make its competitiveness increase markedly in the consumer electronic product. S3C2410 microprocessor is a versatile general chip, which integrate the microprocessor with common peripheral components inside, can be used in various fields, especially the application of handheld device, and make the cost low and cost-effective.

We need to pay attention to the following two points during the Hardware design:

- The plan of memory unit. According to the requirements of practical application, guide code, the kernel and image of file system should be placed in a read storage area of FLASH memory, which reduces the FLASH operation cost and extends the use time of FLASH.
- S3C2410 initialization. During the development of embedded system, how to make system start rapidly and stably is an important question. Because the embedded system resources are limited, and the program is usually located in the ROM to run, but in real application, in order to improve the system real time and speed up the execution rate of the code, the program would often be moved to RAM after it starts, the RAM access speed is faster than ROM, so that it can greatly improve the performance of the system. The task of program start include: hardware initialization, storage system configuration and establishing level 2 interrupt vector table. The system hardware initialization needs to write the instructions in Reset Exception Vector with the address of 0x0; What the storage system configuration does is to copy the system code solidified in ROM or Flash to the RAM, which need to be implemented through writing the program; And the Level 2 interrupt vector table formed by means of removing the interrupt vector table which lies in memory 0x0 to RAM via writing program and the sufficiently using of the function of MMU, and also need to make full use of the function of the MMU.
3.3. Design of software platform

Linux kernel have some characteristics such as stability, powerful function, open source, strong scalability of platform, which is in accordance with POSIX standards, and have a strong network function. These characteristics make it gain a rapid development in the embedded field in recent years, permeates widely into other market such as information home appliances, network equipment and holding terminal etc. Therefore, the RFID middleware software embedded design choose the Linux operating system as software development platform. The way commissioning enviroment choose is the cross-compilation, embedded Linux Kernel used the Kernel 2.4.21 version, and the root file systems use ramdisk.

3.3.1. Custom and transplantation of Linux kernel

The embedded system often face specific hardware platform to accomplish a specific function, the function which is never be used should not be transplanted into embedded system that occupy valuable resources. Therefore, it is necessary to cut Linux kernel, as shown in figure 3 is work flow chart of the development of the embedded middleware Bootloader. Bootloader download kernel image zImage to temporary address 0 x0c100000 of the RAM and start zImage, zImage will compress by themselves, be compressed to the address 0 x0c008000 and run.

3.3.2. The establishment of Cross-compilation environment

This design adopt arm-Linux-GCC as the crossover compiler, the main source files are listed below: the binutils-2.14. Tar. Gz, GCC says Andrea giancoli-2.95.3. Tar. Gz, GCC-g++ 2.95.3. Tar. Gz, glibc-2.2.4. Tar. Gz, glibc-linuxthreads-2.2.4. Tar. Gz, Linux-2.4.21. Tar. Gz and patch-2.4.21-rmk1. Gz.

4. The key technology in the design of RFID middleware embedded

4.1. The key technique in the management of equipment

4.4.1. Equipment management summary of RFID middleware

Monitoring and controlling RFID software and hardware equipment running status is the main function of RFID middleware equipment management, which can provide the services with the effective, reliable, safe and economic status, and finally provide a unified interface to the reader program to control compatibility of many types of reader. The equipment of typical RFID system is main reader, therefore, RFID equipment management refers to the reader the management, middleware management in the RFID is at the bottom connection layers.

This layer shield the diversity and complexity of the RFID hardware equipment and provide the interface of the equipment management and monitoring to the upper of RFID middleware, is an important part of RFID middleware, which has the listed features:

- Independent of the architecture. The equipment layer of RFID middleware is independent, and lies between the RFID reader and back applications, and can get connection with RFID reader and more back application, in order to reduce the complexity of the framework and maintenance.
- Include data collection, filtering, integration and transfer and other characteristics.
- Support the EPC global standards for

```
Set up program entrance pointer
```

```
Initialize the stack and registers of each mode of CPU
```

```
Initialize all kinds of set of inside and outside used in system
```

```
Initial goal development board
```

```
Guide operating system
```

Fig. 3 Flow chart of Bootloader

The agreement on RFID device management related reader drafted by EPC global and relevant organizations mainly include front-end agreement and back-end agreement, the middleware mainly discuss the back-end agreement which is the agreement link the reader with host. The reader equipment is divided into three layers according to the agreement, as it shows in figure 4. the Read Layer is the core of the back-end agreement and it specifies the structure, format of information exchanged reader sensor between hosts, the operation of reader and other means of the reader; Messageing Layer points out how to format and branch the information that form on the Layer, and points how to transport in the specified network; Transport Layer make the system adapt to the current network equipment, and be able to execute communication. In addition, MTB is binding of transmission format defined by a kind of news layer and transmission mode defined by the transport layer. Usually, news layer and transport layer is a pair structure, the agreement between reader and host contain multiple optional news layer and transport layer, the different MTB own different function and different security services etc.

```
Read Layer
```

```
Messageing Layer
```

```
Transport Layer
```

```
Each Pair is
called a
Messageing / Transport
Binding (M
```

Fig. 4 Protocol hierarchy between reader and host

4.1.2. The key technology of the realization of equipment management

Realization method is the key to complete the function of RFID middleware embedded equipment management. The method to accomplishing embedded RFID middleware is to determine the object of
equipment management, the content of equipment management and the functions that equipment management plan to complete, and then realize it. According to the object of management reader defined by the reader agreement and reader management agreement released by EPC global, the objects of reader equipment management contain Read Device, Source, Command Channel, Read Point, IO Port, Alarm Channel, Notification Channel, Alarm Control and Alarm; the main content of the embedded RFID middleware equipment management is the reader the configuration, detection and control; The main complete function are: (1) reader products that support the manufacturers of kinds of readers; (2) support a variety of reader communication interface; (3) support dynamic configuration, different reader possess consistent configuration interface; (4) support f the detection of reader state and report to upper application. The next is to draw their UML relationship chart According to the object of the reader equipment management. And then, according to the management of the object and its UML relationship chart, combining the characteristics of the embedded system, each operation object will be abstracted to terminal, the data conveyed among the terminal will be abstracted to streaming, the framework of reader management have been constructed.

Due to the fact that the object of reader management connect with each other through the reader communication port and host communication port, in addition, host system will not distinguish the communication port, ordinary file or database from each other, all of these data or the data receiver are considered as a file to deal with, therefore, based on the virtual file system provided by operating system, all the readers management operation boil down to the documents I/O. The data transfer between any two terminals or multiple terminals, which can come to true through creating flow object dynamically.

4.2.label data processing

RFID tags data read by the reader in RFID middleware execute data management, coding management and filtering management, and then are stored in local database. Code management is set coding standard, which can be configured to support coding standard of different data, if multiple tags are in the same scope of a reader, which calls for the label conflict processing. According to the characteristics of the embedded system, some functions are used to receive and sent label data under the workbench of Linux:

- unsigned short CalculateBlockCRC16 (int len, unsigned char * pBuf); take calculation and validation of CRC for label data.
- int AppendCRC (void * pBuf, unsigned short CRC); added the CRC value to spot designated by request packet.
- int UartSend (void * pBuf, int uart_num); sent the CRC value and command word to designated port.
- int UartRead (unsigned char * pBuf, int uart_num); to read the returning packets in the specified port to.
- int CRCcheck (unsigned char * pBuf, unsigned short CRC); This function get the CRC and the receiving packet compared to confirm whether it is correct.
RFID data filtering is one of the core functions of RFID middleware, a huge amount of original label data are collected from reader, but the data owning real meaning to the users is not much, if the redundant data are not filtered out, which can bring the burden including the three aspects: 1. it is the burden of network bandwidth, because it is needed to transfer a lot of tag data; 2. it is the burden of data processor, because the processor is needed to deal with a lot of tag data; 3. it is the burden of data storage, because a large of extra label data need to be stored in database. Therefore, filtering redundant data is the primary task of the RFID middleware, embedded RFID middleware designs specialized data processing and filtering modules to finish the task. This module is divided into two sub-modules in logic: logic mapping Engine and RFID data filtering modules. The process of data processing and data filtration are following: First, to map the original label read by all the reader into the each logical reader via the logic mapping engine (namely, based on the reader ID and antenna ID attribute of the original label data, to determine which logic reader data the label belong to), and then filter the original label data that belongs to each logical reader, the label data filtered will be act as the output of logic reader.

4.3. Application of embedded database SQLite

In order to realize the distrbuting and temporary storage of data, embedded RFID middleware design needs to use the database, the type of database is the key to design. This design chooses SQLite database with small volume, powerful function, open source, which is very suitable for the development of embedded. This database provides about 80 C language API interface functions, which makes the operation of the database very convenient, but we just need the following three core functions to achieve the operation of database connection and processing inquiry, the three core functions are:

```c
int sqlite3_open(const char* db,int mode,char** errmsg);
int sqlite3_close(sqlite* db);
int sqlite3_exec(sqlite* db,char* sql,int(*callback)(void*,int,char**,char**),void* parg,char** errmsg);
```

the first two functions are used to open and close the database, the third function sqlite3_exec() is used to deal with SQL query, the second parameter of the function is used to deal with one or more SQL statements, if it is query (SELECT) statement, each record of the results must call the third parameter of the Callback function, the fourth parameter is the first parameter Pointers of Callback function; If not the query statement, then the third, fourth parameters is NULL. After the execution of all SQL have completed which returns 0, or it return an error code, but the detailed false information can be seen through the fifth parameter value.

How to realize the equipment management and data management is also the key, the RFID middleware embedded design have used the open source XML parser expat, and analyzed XML tag data and Sqlite3 API through the use of expat library, which achieve the coupling of equipment management

* `int ProcessTag (unsigned char * pBuf, unsigned char Tags []);` Deal with the frame, make a decoding processing of receiving data or a encoding processing of sending data, and temporarily store the receiving data into an array of data, and prepare for the next step of database access.
and data management through writing data into database. Expat is a kind of XML parser based on events, the event contains tag beginning and tag ending, when these events happen, the registered callback function designated will be called to handle the events, so the writing of the callback function become the key to the analysis process, as shown in figure 5 is the XML format data analytical flow chart based on expat. In the equation, p stands for the return value of XML_ParserCreate () function; the function definition of Start, end and cdata is set by expat, the function body are written by users themselves to achieve their applications. When expat parser meet the start sign of an XML element it will execute start function, if meet the end sign of an XML element it will execute its end function.

![Flow chart of parsing XML data](image)

**5. Conclusions**

The embedded reader is the trend of reader development, so the development of embedded RFID middleware is naturally becoming the key content of development. the design and implementation of the embedded RFID middleware is attempt with much innovation, which realize the basic interface between the different readers and the upper layer application system, and the study provides a certain reference significance for the development of embedded software, but the kinds of readers which support the development is still not wide enough, and the load and power of system still needs to be further improved.

**Acknowledgement**

This research have gained the support from civil aviation university of china special fund of the central university basic scientific research business expenses (ZXH2010D016) and civil aviation university of china fund project (08CAUC-E09).
References


