Fabrication and Development of the Embedded Linux Based on the ARM System

Ruoyu Wang
School of Electronic Engineering
Xi'an University of Posts and Telecommunications
Xi'an, China
E-mail: 2544560286@qq.com

Jiyao Fan
School of Electronic Engineering, Xi'an
Aeronautical Institute, Xi’an, 710077, China
E-mail: fjy_stu@126.com

Lulu Chen
School of Mechanical Engineering,
Xihua University,
Chengdu, China
E-mail: chenll1232022@qq.com

Xiaoheng Sun
School of Electronic Engineering
Xi'an University of Posts and Telecommunications
Xi'an, China
E-mail: xhsun_xiyou@126.com

Chenyu Zhang
School of Electronic Engineering
Xi'an University of Posts and Telecommunications
Xi’an, China
Email: zcy_xupt@163.com

Lei Tian
School of Electronic Engineering
Xi'an University of Posts and Telecommunications
Xi'an, China
E-mail: tla02@126.com

***Abstract*—With the development of the embedded system and improvement of the requirements from customers, embedded GUI which was used to communicate between users and embedded system became to be a key in researching the embedded system. This article introduced what is embedded system and what is embedded GUI. It described the porting of an embedded GUI on ARM Linux platform, including the establishment of compile environment, configuring and modifying of compile options and setting of runtime parameters. Above all, this article introduces the development of Embedded Linux GUI, describes the GUI development environment of Qt /Embedded and Qt, one procedure of the app location development based on Qt/Embedded is described in detail. This system adopts the B/S mode and uses the ARM11 development board with S3C6410 as the core. Sensors are used to obtain information, and the obtained data is judged and analyzed. When the preset conditions are not met, an alarm function is triggered. At the same time, data transmission is carried out through the Zigbee wireless module, and the receiving end analyzes and displays the received data. Each device is connected to the Zigbee module, which distinguishes it by assigning different network addresses and constantly transmits the status of the monitoring area to the receiving end. The receiving end distinguishes the monitoring status of different locations based on the different network addresses, increasing mobility.**

Keywords-Virtual System; Embedded System; GUI; Porting; Qt; Qt/Embedded

I ESTABLISHMENT OF VIRTUAL MACHINE AND INSTALLATION OF LINUX

A. The concept of virtual machines and virtual machine software

The so-called virtual computer (abbreviated as virtual machine) is actually an application software [1]. VMware, the narrow virtual machine software that will be introduced to you today, is actually just an application software. The virtual machine created by VMware is almost identical to the real computer [2-3]. The currently popular virtual machine software includes VMware and Virtual PC. This article is based on VMware Workstation to conduct various practical exercises [4-5].

DOI: 10.2478/ijanmc-2024-0018

1) Establishment of virtual machines

After clicking "New Virtual Machine". Click the "Next" button to enter the virtual machine configuration interface, where there are two options: one is the "Typical" method; The second is the "custom" method. We choose a typical approach here.

Click the "Next" button to enter the virtual machine operating system selection interface, where you can click "Linux". Select "Red Hat Linux". and so on, the user will be asked to set the size of the hard drive, which is 10GB.

2) Installing Linux on a Virtual Machine

The method to install Linux on a virtual machine is actually very simple, as follows [6-7]:

Before installation, we need to set up the optical drive of the virtual machine. At the beginning of the installation process, a welcome dialog box appears and the user presses “Enter” to continue. Red Hat will sequentially ask users what language they are using, the type of keyboard they are using, and the location where the software is installed. Select “Install” in the subsequent upgrade or system installation inquiry, and choose “Custom Installation” for the type of installation to use.

Now users need to create two partitions in the reserved hard drive space. The first partition serves as the root partition of Linux and is used to install Linux files. The second partition serves as a swap partition to supplement the user's physical memory, and it is recommended to set it to one to two times the amount of memory. Here, users will be prompted to set up some hardware, such as mouse, network, Linux Loader and Configurator.

B. Communication between virtual and board

The VMware Workstation software comes with the function of sharing host files. The specific method is to select Virtual Machine→Settings→Options in the menu, where one option is to share a folder. Then, you select the folder you want to share, click OK, and the file will appear in a directory of the virtual machine.

The video monitoring and alarm system based on wired and wireless data transmission is an embedded system developed based on the S3C6410 development board. This system is based on pure video monitoring, combined with data collected from human infrared sensors, sound sensors, light sensors, temperature and humidity sensors, to determine whether an alarm is needed. The data is transmitted through the Zigbee. wireless module in the wireless system to complete the data transmission of the monitoring system. The embedded board is in Figure 1.

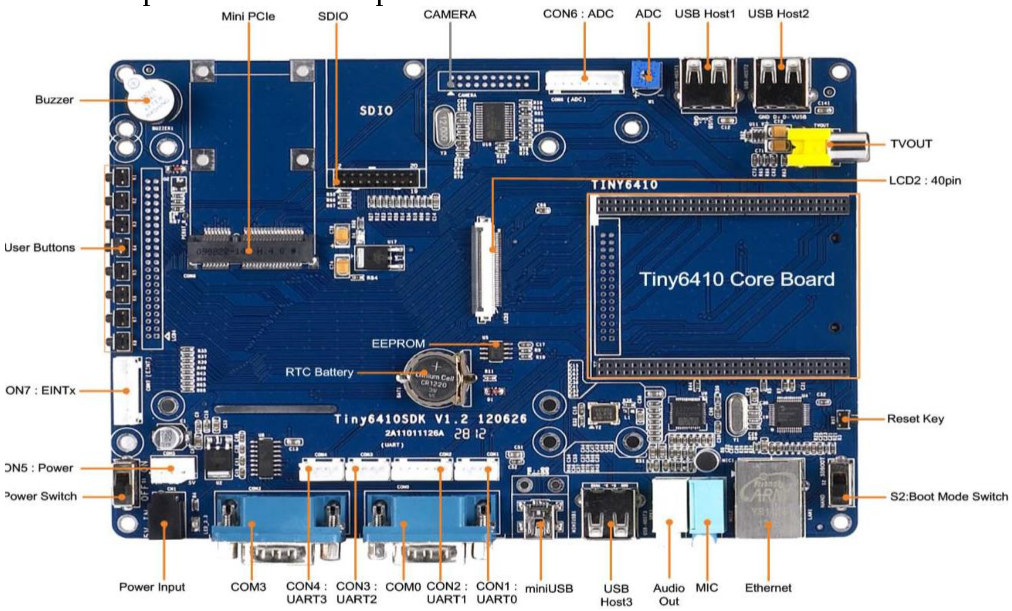


Figure 1. The Embedded Development Board

In Figure 1, the system is the core board which based on the S3C6410 ARM11 chip as the main processor. It integrates 256M of DDR RAM, 2GB of NAND Flash memory, and is powered by 5V. It can be connected to a PC through the COM1 port to access the development board. It can be connected to the Zigbee wireless module through the COM3 port for information transmission and reception, and has three USB interfaces that can be used to connect USB cameras for real-time video monitoring. The core board also comes with two rows of SDIO interface arrays, which can self-connect the required modules. This system requires the application of SDIO ports, human infrared sensor modules, sound sensor modules, temperature and humidity sensor modules, and light sensor modules. The core board comes with LED lights and PWM modules. This article uses an external USB camera and various sensors to obtain data from the monitoring site. It is wirelessly transmitted through the external Zigbee, and the alarm function is achieved through the system's integrated LED lights and PWM.

II EMBEDDED LINUX KERNEL AND FILE SYSTEM

A. Establish the Cross Compilation in S3C6410

The S3C6410 microprocessor is a low-power, highly integrated ARM920T core-based microprocessor designed by SAMSUNG for handheld devices. The development board selected in this article is YF6410II, which uses the S3C6410 microprocessor, making it suitable for developing high-performance handheld and portable smart devices or terminals.

Copy YUANFENG.tar.gz from the Linux directory on the development board's CD to the root directory. The execute script is:

```
[root@localhost YF6410] #./YFINSTALL.sh
```

After the script file is executed, the compiled development environment (arm Linux gcc-2.95.3) is successfully installed, and a tftp server is also installed. The default working directory for tftp server is/tfpboot

Select load configuration file and enter the path to the configuration file in the dialogbox. Next, select the LCD option is 3.9TFT. The touch screen is used in the design, so it is necessary to modify kernel/drivers/char/s3c6410 ts. c. *Download file system: YFLoader#load flash 0x40 0x1b00000 root.*

Turn off the power to the development board and connect the LCD. Restart the development board, and the calibration program for QT will appear on the LCD. After calibration, enter the QT interface.

B. Signal and slot principle

The signal and slot provide a mechanism for communication between objects, and the QT designer can complete it. GUI programs can respond to user actions. When a user clicks on a menu item or toolbar button, the program will execute some code. More generally, various objects need to be able to communicate with each other. Programmers must associate events with relevant code. Trolltech has invented a solution called "Signal and Slot". The signal and slot mechanism are a powerful object to object communication mechanism that can completely replace the rough callback and message mapping of traditional toolkits.

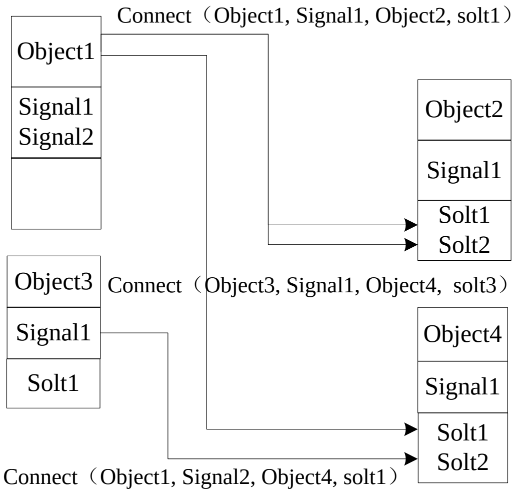

Connect（Object1, Signal1, Object2, solt1）
Object1
Signal1
Signal2
Object2
Signal1
Solt1
Solt2
Object3
Connect（Object3, Signal1, Object4, solt3）
Signal1
Solt1
Object4
Signal1
Solt1
Solt2
Connect（Object1, Signal2, Object4, solt1）

Figure 2. Flowchart of Signal to Slot Connection

In Figure 2, when using the old callback mechanism to connect some code with a button, a function pointer needs to be passed to the button. When the button is clicked, the function is called. Old toolkits cannot guarantee that the parameters given when a function is called have the correct type, making it easy to crash. Another issue with callback methods is that they tightly bind GUI elements with their functionality, making it difficult to independently develop classes.

The signal and slot mechanism of QT are different. When an event occurs, the components emit signals. For example, when a button is clicked, a "clicked" signal is sent. By creating a function (slot) and calling the connect () function to connect signals, programmers can connect signals to slots. The signal and slot mechanisms do not require classes to know each other, making it easier to develop highly reusable classes.

III PROGRAMMING UNDER LINUX SYSTEM

A. Common System Instruction Set for Linux

For beginners of Linux, the biggest headache at first may be not being able to find the software they need to use. In fact, there are also some commonly used software under Linux that have the same functions as similar software under Windows. Linux's vi and Windows's Edit are very similar, and Open Office has almost all the functions of MS Office. The comparison of command lines in Linux and DOS systems is shown in Table 1.

TABLE I. COMPARISON TABLE OF COMMONLY USED COMMANDS IN LINUX AND DOS

Command	***MS-DOS***	***Linux***	***Example of Linux***
Using an editor to edit files	edit	vi	vi thisfile.txt
Copying files	copy	cp	cp thisfile.txt
Transfer files	move	mv	mv thisfile.txt
List files	dir	ls	ls
Delete files	del	rm	rm thisfile.txt
Create directory	mkdir	mkdir	Mkdir directory
Using the specified path	Cd path	Cd path	cd directory

From the Table 1, many Linux commands typed at Shell prompts are similar to commands typed in DOS. In fact, some commands are exactly the same. Table 1 provides commonly used commands under Windows DOS prompts and equivalent commands in Linux. The appendix also provides a simple example of how to use these commands at Linux Shell prompts. Please note that these commands usually have many options, and to further learn each command, please read the relevant manual (man) page (for example, typing man ls at the shell prompt can read information about the ls command).

If the user inserts a USB drive or CD under Windows, Windows may recognize them directly, but in Linux, they must be mounted. Under Red Hat Linux, the command mount is generally used to mount devices. Mount lists all partitions of the system.

B. Common software under Linux

VI is a widely used full screen document editor in the Unix world, and it can be said that almost any Unix machine will provide this software. Qt is a cross platform C++graphical user interface library produced by TrollTech in Norway. It currently includes Qt, Framebuffer and the Qt designer. Qt supports all Unix systems, including Linux. It also supports WinNT/Win2000 and Win95/98 platforms. Qt Designer is a tool used to design and implement user interfaces that can be used on multiple platforms. It can simplify user interface design experiments.

KDevelop is a fast window development tool in the X system. KDevelop itself does not include a compiler, and it uses the GNU compiler suite to generate executable code.

C. Design and Compilation of Qt Window

In order to have a more direct understanding of Qt programming, here is an example for everyone. Run Qt Designer, click on the menu File→new, and create a new project. Select the save path and file name hello.pro, then click on the menu File→New, select C++Source File, confirm, and enter the following content. Using the Qmake for program compilation.

Name the source file hello.cpp. Then open the terminal, enter the directory where the source file is located. At this point, a hello executable file will be generated. Then run it: #/ Hello. The results are as follows in Figure 3.

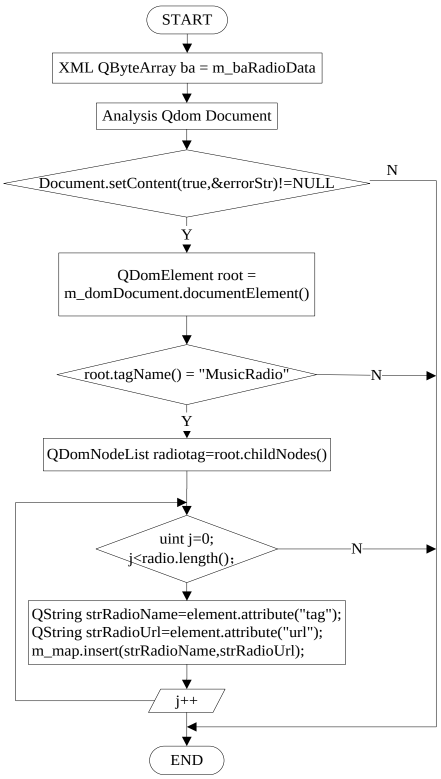

START
XML QByteArray ba = m_baRadioData
Analysis Qdom Document
Document.setContent(true,&errorStr)!=NULL
N
Y
QDomElement root =
m_domDocument.documentElement()
root.tagName() = "MusicRadio"
N
Y
QDomNodeList radiotag=root.childNodes()
uint j=0;
j<radio.length();
N
QString strRadioName=element.attribute("tag");
QString strRadioUrl=element.attribute("url");
m_map.insert(strRadioName,strRadioUrl);
j++
END

Figure 3. The flowchart of the XML file parsing

As shown in the Figure 3, in order to provide more convenient engineering management for user applications, the system parses XML files. A project includes pro, ui, ui.h and the main function. cpp. Among them, the role of engineering files is to manage all files and their relationships in the current project, while the role of form files is to manage various interface elements and their properties in the form. In addition to reading and writing to. ui files, Qt Designer also adds a form ui.h file. It is a regular C++ file, mainly used for customizing slot methods the ui.h file is the responsibility of Qt Designer to maintain consistency with the slots related to the form. Whenever a user adds, removes, or changes the connection of a slot to the form, Qt Designer will automatically update it the content of the ui.h file.

IV PORTING AND DEBUGGING OF QT GRAPHICS PROGRAM

A. Establishment of Qt/Embedded Development Environment

Due to Qt/Embedded using Framebuffer as the underlying graphics interface, in order to use Qt/Embedded [8]-[10]. It is necessary to start the Framebuffer driver program. Because by default. Linux does not configure frame buffer drivers [11]-[13]. Therefore, it is necessary to enable frame buffering under Linux. The specific methods are as follows:

Log in with root privileges to the/boot/grub directory, type the vi gruB.conf command to modify the gruB.conf file. If you restart and enter, you will see the cute little penguin in the upper left corner of the screen, which indicates that Linux has enabled the frame buffer driver [14-15].

B. The process of establishing a Qt/Embedded development environment

Qt/Embedded can create different development environments for different platforms. In this article, as the ARM system is used, it is necessary to establish an X86 development environment and ARM development platform. The specific methods are as follows. Here, the environment to run on the X86 platform. Run the following command in Linux command mode, it is show in Figure 4.

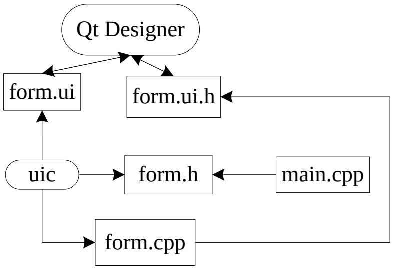


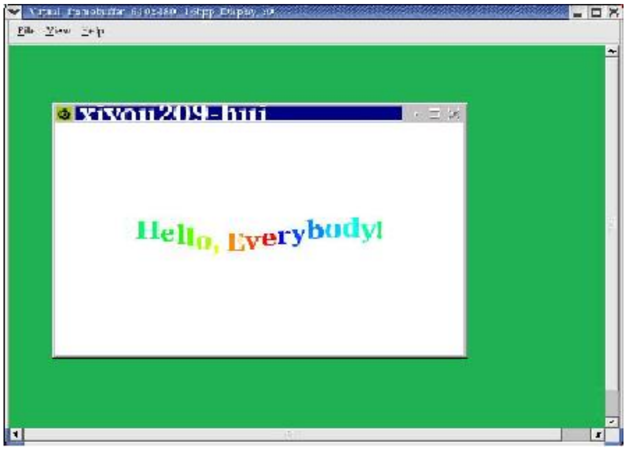
Figure 4. Working principle of all engineering files

After working the Qt files, then run the following command in Linux command mode. The specific usage of compilation options is feasible/ View the configure help and make help commands. According to the developer's own development environment, other parameters can also be added to the configure parameters, such as - *no opengl or - no xft.* The installation of QT/X11 is mainly to provide designer tools and qvfb tools to QT/E.

In fact, the method for establishing the QT/E environment of ARM is basically the same as that for establishing the environment of X86. Just because the libraries used on the ARM platform and X86 are different, it is necessary to add corresponding parameters during compilation. Here, it is necessary to:

If the program is written using an interface designed by Qt Designer, the *. ui file needs to be converted to*. h file and *. cpp file. Then write a *. pro file (used to generate Makefile files), which has a relatively fixed format.The basic format of the hui. pro file is as follows. Here, runs the application on the host. Before compiling and executing QT/E, it is necessary to first set up the TMAKE and QT/E LIB environments.

After entering the directory where QT/X11 is installed and execute the program qvfb in the bin directory. Then execute the QT/E program, the corresponding program display will appear in the window opened by the qvfb program, it is showed in Figure 5.



```
Hello, Everybody!
```

Figure 5. The execution result of hello

Compile programs for the development board. Similarly, the first step is to set up the TMAKE and QT/E LIB environments, with basic parameters identical to those on the host. There is only a difference when setting the TMAKEPATH path for tmake and the LD-LIBRARY-PATH path for QT/E. Need to convert:

C. The Embedded Development Board Control Platform

Due to the design of the remote monitoring and alarm system in this article, it is necessary to display the data obtained from modules such as USB cameras, temperature and humidity sensors, human infrared sensors, light sensors, sound sensors, etc. connected to the development board on a PC through a web page. Therefore, a web server should be established on the development board to enable it to run web programs, and the PC can access the data on the development board. The specific physical image is shown in Figure 6.

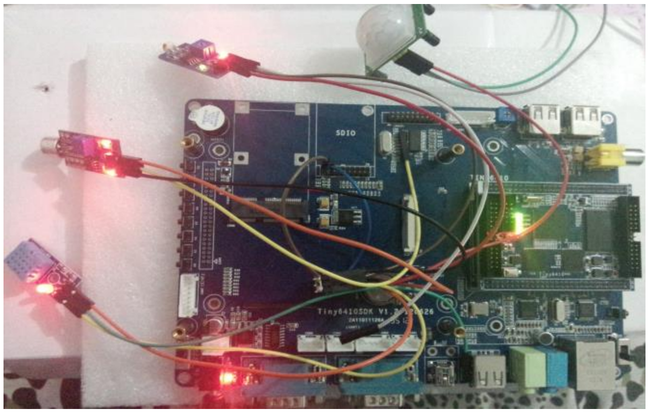


Figure 6. The specific physical chart

After the system is powered on, when the network addresses of multiple modules are in the same network segment, the modules automatically form a network, and the coordinator automatically assigns addresses to the routing. It can also be configured through module configuration software. If the module loses power, the network can be automatically repaired after power on. Firstly, the configured coordinator receives information from the serial port and automatically sends it to all routers in the network; After obtaining data from the serial port, the router will forward it to the coordinator; At this point, users can transmit to each other through specified addresses between any two routers or coordinators.

The Zigbee wireless transmission module mainly achieves transparent data transmission from serial port to Zigbee wireless through the Zigbee wireless transmission module. It is based on TI's CC2530 chip, which adds RF transceiver and runs the ZIGBEE 2007/PRO protocol stack internally, with all the characteristics of the ZIGBEE protocol. For the complex Zigbee protocol, the transparent transmission Zigbee module embeds the protocol stack into the module and, in conjunction with the UART to RS232 motherboard, converts the UART of the Zigbee module into a standard RS232 interface. It can be directly connected to the serial port of the development board to transmit and receive data through the read/write serial port.

V CONCLUSIONS

So far, our porting work has been successfully completed. Qt has successfully run on our S3C6410 platform. Due to the fact that embedded systems belong to resource limited systems, embedded GUI development must also consider the issue of system overhead, which requires careful analysis of specific applications to meet the specific needs of users. Qt/Embedded continues all the functions of Qt in desktop systems, with rich API interfaces and component.

This paper designs a monitoring system based on wired transmission, a monitoring system based on wireless transmission, and a wired wireless joint debugging system. This system is a low-cost network monitoring embedded system based on ARM11's S3C6410 as the core. This system utilizes embedded devices, such as the S3C6410 development board. The early warning system studied through this ARM development board is based on a pure alarm system. In addition, data collected from human infrared sensors, sound sensors, light sensors, temperature and humidity sensors, and images captured by USB cameras are sent through wired or Zigbee wireless modules to determine whether an alarm is needed. And porting the client to Windows by accessing the development board to access the server meets the needs of more users.

ACKNOWLEDGMENT

This work was partly supported by the research project of Xi’an University of posts and telecommunications teaching reform JGA202304 and the undergraduate innovation and entrepreneurship plan 202311664058.

REFERENCES

[1] N. Jagadeeswari and V. M. Raj, “Homogeneous Batch Memory Deduplication Using Clustering of Virtual Machines,” Computer Systems Science and Engineering, vol. 44, pp. 929-943, June 2022.

[2] L. Huibo, W. Haiqiao, L. Ping and G. Peng, “VMware-based Hardware-in-the-loop Simulation Approach for Link11 Data Link,” Binggong Xuebao/Acta Armamentarii, vol. 41, pp. 224-233, June 2020.

[3] L. Ying. “Lightweight information integration model of IOT based on VMware cluster technology,” International Journal of Information and Communication Technology, vol. 18, pp. 41-56, December 2021.

[4] M. A. A. Mohamed, “VFST: Virtual and fully software-based toolchain for PC interfacing education and research,” Computer Applications in Engineering Education, vol. 31, pp. 389-407, March 2023.

[5] A. Bhonde and S. R. Devane, “Priority-based security-aware virtual machine allocation policy,” International Journal of Information and Computer Security, vol. 23, pp. 40-56, February 2024.

[6] N. Karapetyants and D. Efanov, “A practical approach to learning Linux vulnerabilities,” Journal of Computer Virology and Hacking Techniques, vol. 19, pp. 409-418, September 2023.

[7] L. Kunli, L. Wenqing, Z. Kun and T. Bibo, “HyperPS: A Virtual-Machine Memory Protection Approach Through Hypervisor's Privilege Separation,” IEEE Transactions on Dependable and Secure Computing, vol. 20, pp. 2925-2938, July 2023.

[8] A. K. George. “Real-time performance analysis of distributed multithreaded applications in a cluster of ARM-based embedded devices,” International Journal of High-Performance Systems Architecture, vol. 11, pp. 105-116, December 2022.

[9] L. Vignati, S. Zambon and L. Turchet, “A Comparison of Real-Time Linux-Based Architectures for Embedded Musical Applications,” AES: Journal of the Audio Engineering Society, vol. 70, pp. 83-93, January 2022.

[10] W. Tao, W. Li, Y. Pengfei, R. Popli, P. Rani and R. Kumar, “Application of embedded Linux in the design of Internet of Things gateway,” Journal of Intelligent Systems, vol. 31, pp. 1014-1023, January, 2022.

[11] J. Qiaowen, M. Haoyu, L. Yan, W. Zheyu and S. Wenchang, “A Novel Integrity Measurement Architecture for Embedded Linux Systems,” Jisuanji Yanjiu yu Fazhan/Computer Research and Development, vol. 59, pp. 2362-2375, October 2022.

[12] F. Bowen, L. Su, W. Jiangdong, L. Qiran, W. Qingnan and T. Jihui, “A Novel Intelligent Garbage Classification System Based on Deep Learning and an Embedded Linux System,” IEEE Access, vol. 9, pp. 131134-131146, September 2021.

[13] Q. Jiaqing, W. Huachen and G. Fei, “A multiprocessor real-time scheduling embedded testbed based on Linux,” International Journal of Embedded Systems, vol. 14, pp. 451-464, January 2021.

[14] L. Kaizheng, Y. Ming, L. Zhen, Y. Huaiyu, Z. Yue, F. Xinwen and Z. Wei, “On Manually Reverse Engineering Communication Protocols of Linux-Based IOT Systems,” IEEE Internet of Things Journal, vol. 8, pp. 6815-6827, April 2021.

[15] D. E. Mera, R. A. R. Solis, L. Reyes, R. Armstrong, W. J. Hernandez and A. L. Guzman-Morales, “A Power and Performance Study of Compact L-Band Total Power Radiometers for UAV Remote Sensing Based in the Processing on ZYNQ and ARM Architectures,” IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing, vol. 15, pp. 1103-1113, December 2022.