Journal of Computer and Communications, 2023, 11, 51-63
https://www.scirp.org/journal/jcc
ISSN Online: 2327-5227
ISSN Print: 2327-5219

ARM-Based Embedded System Platform and Its Portability Research

Hao Liu

Hangzhou Wickham International School, Hangzhou, China
Email: liuhaoliu06@sina.com

How to cite this paper: Liu, H. (2023) ARM-Based Embedded System Platform and Its Portability Research. *Journal of Computer and Communications*, **11**, 51-63.
https://doi.org/10.4236/jcc.2023.1111003

Received: October 10, 2023
Accepted: November 18, 2023
Published: November 21, 2023

Abstract

Taking ARM as the hardware platform, the embedded system is built from both hardware and software aspects with the application as the center. In the hardware design, build the hardware platform scheme, design the schematic diagram as well as PCB, complete the hardware debugging, and ensure the system hardware platform function; in the software design, optimize the three-stage pipeline structure of ARM instruction system, design the instruction set, install the embedded system on the virtual machine, build the cross-toolchain, and set up the correct NFS network file system. Finish the design of the ARM-based embedded system platform, combined with the hardware requirements of the experimental platform, transplant the powerful Uboot as the Bootloader of the system, and further transplant the Linux-2.6.32 kernel to the system start the operation normally, and finally, build the root file to finish the study of its portability.

Keywords

Embedded Platforms, Portability, Operating Systems

1. Introduction

An embedded system is an emerging technology field, and in recent years, with the rapid development of computer technology, electronic technology, communication technology network technology, and other fields, an embedded system has also been widely used. PC + operating system + software development environment centered on PC is today's mainstream embedded system development mode. With the continuous expansion of embedded system application fields, the research of embedded system platforms has become a hot spot. At present, the more common embedded system platforms on the market mainly include embedded systems based on ARM (Atmel) series processors, embedded systems

based on 51 series microprocessors, embedded systems based on AVR (Area Remote Processor) series processors, and so on. Among many embedded system platforms, ARM (Advanced RISC Machines) series chips have become the most widely used microprocessors in the current market due to their low power consumption, high performance, low price, and other characteristics [1]. Embedded System (Embedded System) refers to embedded devices, which are microcomputer systems designed for specific applications. It has the advantages of small size, low cost, strong function, high reliability, strong adaptability, etc. In recent years, it has been widely used in the fields of industrial control, information home appliances, and consumer electronics. Embedded systems are mainly composed of the following parts in terms of hardware: microprocessor (MCU), memory, input/output interfaces, human-computer interfaces, and bus interfaces. From the software side, it mainly consists of the following parts: operating system, embedded real-time operating system (RTOS), and application software development tools [2].

Currently, the design and development of embedded systems are mainly based on PCs, but this approach cannot realize hardware and software co-design in the true sense. The PC-centered embedded system development mode largely improves the efficiency of embedded system design and development, but it has limitations in both hardware and software [3]. Although the PC-centered embedded system development mode improves the efficiency of embedded system design and development, its hardware resources are limited, software resources are insufficient, it is difficult to design complex algorithms, and the lack of independent intellectual property rights and other defects prevent it from being widely used in industrial control, information appliances, and other fields. Therefore, an ARM-based embedded system platform is designed and its portability is studied. The research objective of the ARM-based embedded system platform is to identify a feasible development process and methodology to achieve development efficiency and software quality assurance, and to provide technical support for practical embedded system applications. In addition, the research also focuses on the portability of the embedded development platform, aiming to realize a prototype of a source code autoporter to improve the automation of the embedded system development process. Specifically, for the research of embedded system platform, the subject regards platform technology as a set of modern system design methods, and explores its component elements and realization methods in detail. On this basis, the platform stack model of generalized electronic engineering is comprehensively constructed, and the architecture platform and programming model platform therein are taken as the core, and the infrastructure of embedded system is proposed. Meanwhile, for the portability problem of embedded systems, the research goal is to realize a prototype of source code autoporter. This goal stems from the fact that the lack of automatic porting tools between programming model platforms and architecture platforms during embedded system development is one of the most important

reasons for the inefficiency of embedded design. The automation of the embedded system development process can be greatly improved by the implementation of a source code autoporter. To summarize, the research of ARM-based embedded system platform and the solution of portability problems are of great significance to improve the efficiency of embedded system development and ensure the quality of software, which provides effective technical support for practical applications.

2. ARM-Based Embedded System Platform Design

2.1. ARM-Based Hardware Design

With the rapid development of electronic technology, computer technology, and communication technology, embedded systems are developing towards intelligence, miniaturization, networking, and multifunctionality. Against the background of the current rapid development of computer technology, embedded systems specially designed for specific application needs are becoming one of the most popular computer systems with their high performance, low cost, and flexibility. Hardware resources are a crucial factor in various embedded applications. Due to the limited hardware resources, a solution based on a microprocessor + software system development platform has been proposed to solve the contradiction that exists between embedded application requirements and hardware resources. Since the embedded application software developed with this solution is characterized by generality and good portability, it has been widely used. In this paper, we design an embedded system platform with ARM as the core processor and STM8L051F3P6 as the hardware base [4]. The platform supports 32-bit and 64-bit RISC processors; has rich I/O ports and powerful communication functions; and supports RTOS (real-time operating system). The embedded system platform has higher requirements for functional interfaces. To improve the accuracy of the system platform for calculating a large amount of data, the enhanced 16-bit microcontroller, model STM8L051F3P6, is selected, and its internal structure is shown in the following **Figure 1**.

As the system will have certain harmonics in the process of use, to ensure the accuracy of the calculation, it is necessary to take into account the effect of harmonics, thus increasing the complexity of the calculation, the introduction of this model of microcontroller can complete the heavy accumulation and calculation, which is very helpful to speed up the sampling speed.STM8L051F3P6 microcontroller with built-in fast flash memory, so it has the advantage of ultra-low power, to ensure that the low current consumption of the monitor and prolongs the battery life. This is done as a result of micro-control accomplished in different modules and CPU states working in different ways, during the processing of interrupt events, which wake up the monitor and return to the pre-interrupt state using built-in instructions. There are FLASH ROM and RAMs in the microcontroller structure in which the execution of program instructions can be accomplished to be able to speed up the calculations and reduce the power. In

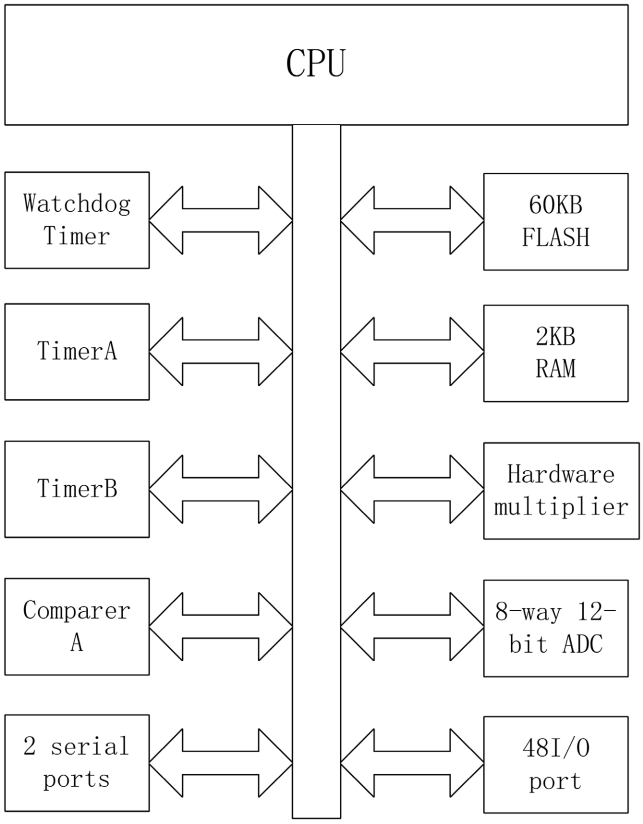

Figure 1. STM8L051F3P6 internal structure diagram.

this microcontroller, there are 5 relevant input channels for the A/D converter with a maximum allowable voltage value of 5 V and an input range of 0 - 5 V, where the resolution is 8 bits [5]. If the input voltage is in the range of 0 - 250 mV, an internal fixed gain amplifier can bring the resolution up to 11 bits, which allows the reduction of external devices. The internal control and display circuitry of the microcontroller is shown in **Figure 2** below.

The model microcontroller selected in this paper extends the scope of low-end applications in the optimization control system for hydropower photovoltaic energy storage and power supply, which can be applied in a very small package so that the control process is characterized by real-time updating, and there is also a better application of monitoring and mastering of the frequency [6]. The biggest feature in the hardware design of this paper is the ability to highly integrate external devices, which can reduce the number of external devices used in the system, thus reducing costs. On the other hand, in the optimization and control of the system, the original system of the microcontroller is used in the external oscillation circuit, the volume is large and the interference of the external environment is more obvious. To address this point, the ST7 uses a high-precision RC oscillator instead, which improves accuracy while reducing size and saving cost, and also provides more hardware and software resources. The ST7 has an internal program memory that occupies a capacity of 1.5 kB, and the RAM occupies a capacity of 128 B. The microcontroller has several multiplexed

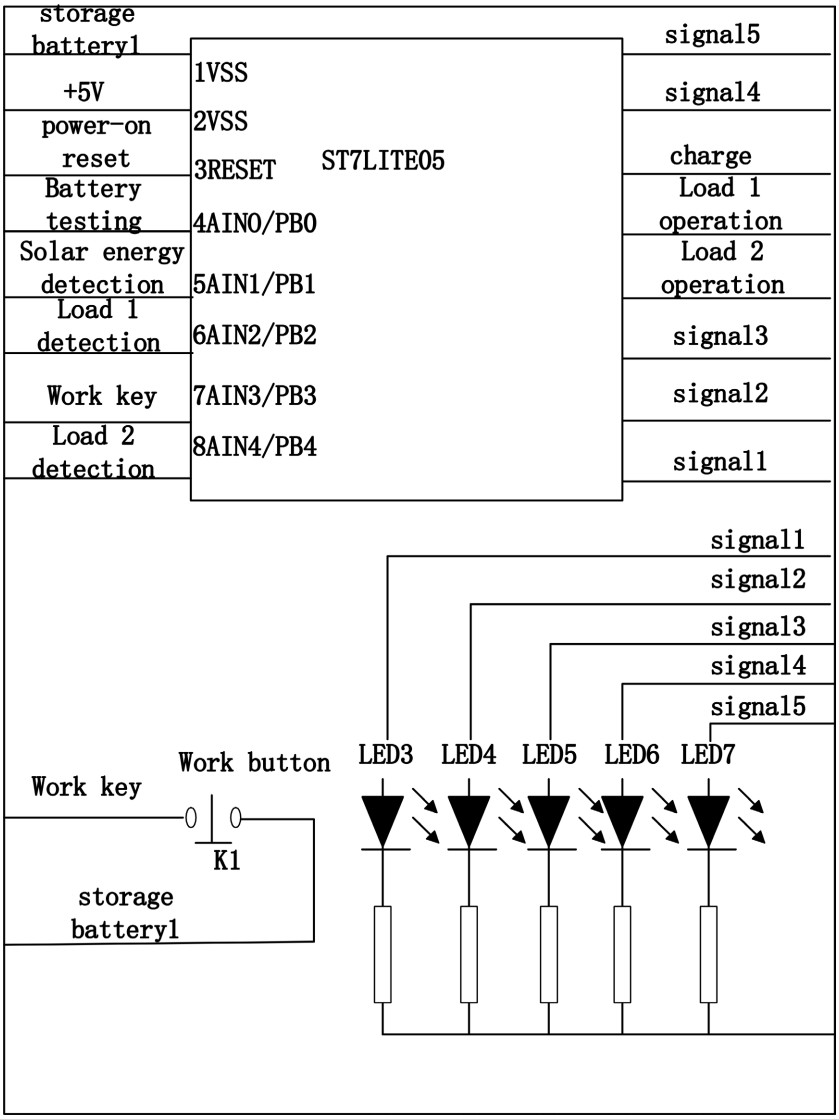

Figure 2. Circuit design around the microcontroller.

I/O pins internally, and there are also six high-current output pins. In addition, an 8-bit LIFE timer and a 12-bit auto-reload timer with output comparison and PWM are included. the LIFE timer includes a watchdog, a real-time reference, and an input trap. For this paper, the system makes full use of the resources within the microcontroller to minimize the number of external devices. This completes the hardware debugging and ensures that the system hardware platform functions [7].

2.2. Software Design

2.2.1. Optimizing the Three-Level Pipeline Structure of the ARM Instruction System

The RM instruction system is based on the Reduced Instruction Set (RISC), which increases the speed of instruction execution by improving the pipeline structure of instruction execution [8]. In embedded systems, processors under

the ARM architecture are usually used. Therefore, to achieve high efficiency and good portability of ARM architecture processors for embedded systems, it is necessary to optimize the ARM instruction system. The advantage of ARISC is that it has a simple hardware structure, does not require additional peripherals such as memories and operators, and is easy to implement complex algorithms. The disadvantage of RISC is that it occupies a large amount of hardware resources, and has a low performance. To solve the above problems, ARM uses a three-stage pipeline structure.

The kernel compilation for the three-stage pipeline structure is shown in Figure 3 below.

The first stage pipeline is used to transfer data from the address line to the control line, where the data is assigned to the corresponding operation unit and then the result is returned to the address line; the second stage pipeline is used to transfer data from the address line to the operation line, where the corresponding operation is performed; and the third stage pipeline is used to transfer data from the operation line to the address line. This three-stage pipeline structure can greatly increase the speed of instruction execution and reduce the time required for data access.

2.2.2. Designing the Instruction Set

The instruction set here refers to the instructions that are recognized by the decoding mechanism, *i.e.*, the machine instruction set, for which ARM defines the corresponding mnemonics since machine instructions are difficult to memorize. It is worth noting that pseudo-instructions and pseudo-operations are also used

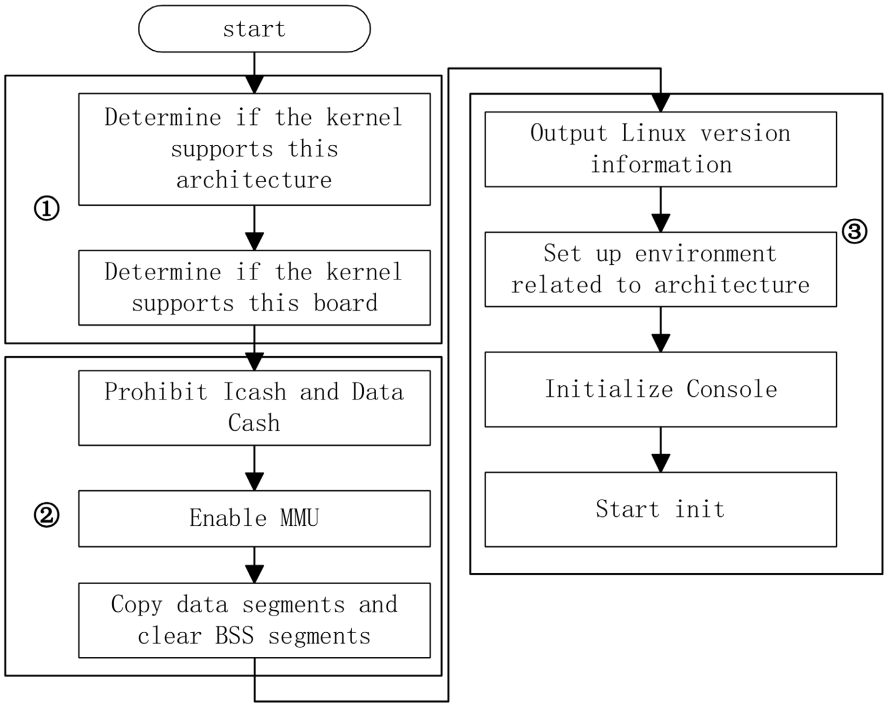

Figure 3. Optimized ARM instruction system flow structure.

in assembly-level programming, and they are actually instructions defined by the ARM assembler specification for assembly control without corresponding machine instructions and therefore do not belong to the ARM instruction system. As can be seen, the design of the instruction set includes the design of machine instructions and helpers. The design of machine instructions is mainly to divide the 32-bit machine instructions into reasonable domains, and the decoder controls the actuator according to these domains to perform what kind of operation. The main goal of mnemonic design is to realize a set of assembly statements that are easy to remember and use so that programmers can use them conveniently, so convenience and flexibility are the most important.ARM mnemonics include opcode mnemonics, conditional mnemonics, modifier mnemonics, and four types of opcode mnemonics for operand addressing and shifting, which are used in conjunction with each other to provide a very flexible and powerful assembly programming foundation. ARM7 adopts the ARMv4 instruction set, providing both ARM and THUBM instruction sets. 16-bit THUBM instructions will eventually be translated by the decoder into one-to-one 32-bit ARM instructions in real-time, so it can be said that the THUBM instruction set is a subset of the ARM instruction set. Therefore, the following will mainly discuss the ARM instruction set. According to the characteristics of the ARM processor, the S3C2440 processor in the ARM7 series is selected for this design, which has a built-in 32-bit CPU with a 16-bit RISC structure. The design requirements of the instruction set are as follows when carrying out the program design:

1) Different operations can be obtained by modifying only a few simple instructions.

2) When some new instructions need to be introduced in the program design, it is only necessary to modify the original instruction set used.

3) To make the program run more efficiently, some commonly used but infrequently used instructions can be used in the newly developed program to simplify the program design. Therefore, we designed a 32-bit RISC structure instruction set, which is the instruction set used in the S3C2440 processor in the ARM7 family, and its architecture is shown in **Figure 2**.

4) To facilitate interaction with the operating system, some simple but commonly used applications can be written on the ARM7 series processors. For example, in the application program, you can input a Chinese character and output the Chinese character information; input a string and output the character information; input a number and a letter, and so on.

5) To make the program run more efficiently, some commonly used but not so commonly used instructions in the ARM7 series processors can be simplified, e.g., inputting a number and outputting a number [9].

In program design, we can flexibly use some common but not common instructions. For example, when running some basic functions, you can use Sublime Text; when running some simple script programs, you can use Text Embedding, and so on. These are all based on simple instruction sets. However, for

complex functions or programs, since they need to call a large amount of code, they need a complex instruction set for processing. With the above design principles, the resulting mnemonic categorization table is shown in the Table 1 below:

It can be seen that ARM instructions can be broadly categorized into data processing instructions, data transfer instructions. From the table, we can see that ARM7 provides some multiplication instructions, which are done by a hardware multiplier, but still, there is no division instruction. Data processing instructions are the only instructions that can perform operations on register values. Data transfer instructions can do transfers between single registers, transfers of bytes, half-words and full words from single registers to memory,

Table 1. ARM instruction opcode mnemonic classification table.

classification	mnemonic	
Data Processing Instructions	Addition operation	ADD
		ADC
	Subtraction operation	SUB
		SBC
		RSB
		RSC
	Multiplication operation	MUL
		METAL
Data Transfer	Bitwise logical operation	AND
		ORR
		FOR
		BIG
	Compare and Test Operations by Conditional Bits	CMP
		CMN
		TST
		TEQ
	Data transfer between ordinary registers	MOV
		MVN
	Data transfer between status register and regular register	MRS
		MSR
	Data transfer between a single register and memory	LDR
		STR
		SWR
	Data transfer between multiple registers and memory	LDW
		STM

swaps between single registers and memory, and also transfers between multiple registers and memory. Because some of the fields in the machine instructions are allocated to opcodes and condition codes, only 24 are allocated to the jump offset, and the actuator calculates the 32-bit destination address by adding the jump offset to the current PC value. Jump instructions with an L suffix indicate that the entry address is saved to the link register when jumping, so they can be used for subroutine calls. When the subroutine is completed, the address value in the link register is returned to the PC. A jump instruction with an X suffix indicates that the program is switched into the THUBM code segment. Use L and X together as a suffix, and switch freely between ARM codes [10].

2.2.3. Embedded System Platforms

The core structure of ARM processor is ARM+ DSP+ FPGA, and the main features of ARM+ DSP are its higher main frequency and stronger processing capability. Therefore, we can combine ARM processor and DSP to form an embedded system platform based on ARM processor. The advantages of this scheme are: that multiple tasks can be developed at the same time, which improves the development efficiency of the embedded system platform; at the same time, it can also be seamlessly connected, making full use of the resources of each chip. Based on the above analysis, this paper designs an embedded system platform based on ARM processor. The system uses S3C2440 as the processor, which is a low-power, high-performance, low-cost, multi-functional 32-bit ARM7TDMI microcontroller designed for embedded applications, with a maximum host frequency of up to 1.33 GHz. The S3C2440 core adopts the ARM7 kernel and the ARM9 kernel, and its main function is to process and store data in real-time; it also supports Multi-tasking and interrupt processing, to provide users with a safe, reliable and stable embedded application environment.S3C2440 is a high-performance, low-power, multi-function 32-bit ARM7TDMI microcontroller introduced by ARM in April 2004, using the ARM7 core and ARM9 core.S3C2440 has three 64 KB RAM, four The S3C2440 has three 64 KB RAM, four 64 KB SRAM, 32-bit FLASH, and one 10 KB SRAM. Its built-in timer generates more than 20 timers including interrupt sources, and supports two 8-bit D/A output (DC-DC) inputs, enabling power management and signal conditioning in a variety of applications. Its core adopts ARM7TDMI structure with two kinds of processor cores, ARM7 core and ARM9 core, and can run 32-bit RISC operating system, which can complete various applications based on S3C2440 processor.

μC/OS-II operating system is an embedded operating system composed of μC/GUI graphical user interface and μC/GUI application program, which includes two parts: kernel and application program, in which the application program is mainly provided by the user, and the required functions are realized by modifying the application program provided by the user; μC/OS-II kernel embeds and optimizes μC/GUI by modifying the source code. The μC/OS-II kernel embeds and optimizes the μC/GUI by modifying the source code, so that it has better performance and more functions, and the data transmission between the

ARM processor and the μC/OS-II operating system is carried out through the SPI communication interface. Meanwhile, the embedded system platform based on ARM processor can be easily connected to PC, thus providing users with a safe, reliable, and stable embedded application environment. The development process of the system software mainly includes the following stages:

1) Boot the virtual machine. First of all, we need to burn the boot program of the system into the virtual machine and set up the relevant parameters, such as boot mode, boot mode, cross-compilation environment, and so on. Secondly, we need to set up the main program of the system, such as the structure and function of the main program. Finally, we need to boot the system into a suitable running environment.

2) System initialization. In this phase, we need to carry out the following tasks: set up the main program entry function; set up the virtual machine; initialize each interrupt service program and other programs.

3) Task initialization.

4) Mandate implementation.

5) Scheduling and control of tasks.

6) System debugging and testing. In this stage, we need to debug, analyze, and optimize the system and make necessary changes to the system based on the debugging results.

7) System updates and upgrades. In this phase, we need to work on the system such as updating and upgrading.

3. Embedded System Platform Portability Study

The embedded system platform designed in this paper adopts S3C2440 in hardware, KingView as its development environment, and embedded Linux as its operating system. The advantage of doing so is that the embedded system platform has strong portability because Linux is an open-source operating system, and its kernel source code can be downloaded for free on the Internet. Secondly, it adopts RTOS (Reliable Operating System) as its kernel, which makes the system highly portable. Furthermore, since the embedded system platform is developed based on Linux, Linux itself supports various hardware platforms and device drivers. Therefore, the system platform can be easily ported to different hardware platforms.

In addition, another feature of the embedded system platform designed in this paper is that the system software can run on ARM7. Because the S3C2440 itself is a 32-bit CPU with relatively strong computing power. It uses two 32-bit ARM7TDMI-S cores as the system core CPUs and is also equipped with one 32-bit ARM core (supporting SSE) and eight 32-bit SPI cores (supporting DDR). This allows applications running on ARM cores to run on 32-bit ARM7.

3.1. Porting the Development Environment

The software development environment of the embedded system platform de-

signed in this paper is based on the KingView development environment, which is a powerful, simple, and easy-to-use embedded Linux system with LWIP development language, and also integrates various hardware drivers and middleware, and supports hardware abstraction layer (HAL) and assembly language. Powerful development tool that provides a rich set of embedded Linux development tools and a compilation environment. It not only provides a convenient development platform for users but also a friendly interface.

The software on the embedded system platform designed in this paper is developed on KingView, so firstly, embedded Linux should be installed in KingView and then downloaded to the target board. Here, KingView will download the system software into the ARM7 kernel and then download it to the target board. Pay attention to the following points in this process:

1) Download the ARM core to the target board;
2) Download the Linux file system under S3C2440 to the target board;
3) Install the relevant drivers and middleware programs for the host.

3.2. Embedded Linux System Porting

An embedded system is hardware and software, with its characteristics of the system, it must be able to run on different hardware platforms, but also to ensure that the application program can be ported to other hardware platforms to go, for embedded systems, its development environment and general computer development environment is completely different. An embedded system development environment is developed on a specific hardware platform, this specific hardware platform usually refers to the special integrated circuit chip (ASIC) or other functional modules of the circuit. Therefore, when transplanting embedded systems, we must first understand the structure, characteristics, and software platform of this hardware platform, and then select the appropriate operating system (Linux or other). According to the characteristics of the embedded system designed in this paper and the Linux kernel source code used, a suitable Linux operating system is selected for transplantation.

3.3. Porting Device Drivers

Embedded system platforms are developed under the Linux operating system. As an open-source operating system, the source code of Linux is publicly available and its kernel source code can be downloaded for free. When porting device drivers, as long as the Linux source code used is compiled according to the requirements, it can be directly compiled into the target device driver, so that the device driver can be directly ported to the embedded system platform. The embedded system platform designed in this paper mainly adopts S3C2440 as the hardware core, and only some code modifications are needed to port the device driver. Porting the device driver for this embedded system platform mainly includes the following aspects: first, modify the hardware configuration of S3C2440. Second, the device driver ported to the kernel is compiled according to

the requirements. Third, port the relevant parts of the compiled target device driver to the embedded system platform. Finally, package the target device driver and burn it to the development board after the package is completed.

4. Concluding Remarks

ARM-based embedded system platform has a wide range of applications in embedded systems, the platform in the realization of a variety of functions at the same time, reduces the cost of the system, improves the reliability of the system, improves product quality, reduces the difficulty of development, shortens the time-to-market and so on has a very good effect. In this paper, an ARM-based embedded system platform is studied, an ARM-based embedded system platform is designed and realized, and a hardware development environment based on S3C2440 as the core and S4P is constructed on this basis.

Conflicts of Interest

The author declares no conflicts of interest regarding the publication of this paper.

References

[1] Kim, E., Kim, J., Park, J., Ko, H. and Kyung, Y. (2023) TinyML-Based Classification in an ECG Monitoring Embedded System. *Computers, Materials and Continua*, **75**, 1751-1764. https://doi.org/10.32604/cmc.2023.031663

[2] Richa, M., Prévotet, J.C., Dardaillon, M., Mroué, M. and Samhat, A.E. (2023) High-Level Power Estimation Techniques in Embedded Systems Hardware: An Overview. *The Journal of Supercomputing*, **79**, 3771-3790. https://doi.org/10.1007/s11227-022-04798-5

[3] Ansari, M., Safari, S., Yari-Karin, S., Gohari-Nazari, P., Khdr, H., Shafique, M., *et al.* (2021) Thermal-Aware Standby-Sparing Technique on Heterogeneous Real-Time Embedded Systems. *IEEE Transactions on Emerging Topics in Computing*, **10**, 1883-1897. https://doi.org/10.1109/TETC.2021.3120084

[4] Saddik, A., Latif, R., El Ouardi, A., Elhoseny, M. and Khelifi, A. (2022) Computer Development Based Embedded Systems in Precision Agriculture: Tools and Application. *Acta Agriculturae Scandinavica, Section B*, **72**, 589-611. https://doi.org/10.1080/09064710.2021.2024874

[5] Glazer, D.I., Zhao, A.H., Lacson, R., Burk, K.S., DiPiro, P.J., Kapoor, N. and Khorasani, R. (2022) Use of a PACS Embedded System for Communicating Radiologist to Technologist Learning Opportunities and Patient Callbacks. *Current Problems in Diagnostic Radiology*, **51**, 511-516. https://doi.org/10.1067/j.cpradiol.2021.09.007

[6] Liu, M., Deng, X., Lei, Z., Jiang, C. and Piao, C. (2021) Autonomous Lane Keeping System: Lane Detection, Tracking and Control on an Embedded System. *Journal of Electrical Engineering & Technology*, **16**, 569-578. https://doi.org/10.1007/s42835-020-00570-y

[7] Tarapore, D., Roozkhosh, S., Brzozowski, S. and Mancuso, R. (2021) Observing the Invisible: Live Cache Inspection for High-Performance Embedded Systems. *IEEE Transactions on Computers*, **71**, 559-572. https://doi.org/10.1109/TC.2021.3060650

[8] Eceiza, M., Flores, J.L. and Iturbe, M. (2021) Fuzzing the Internet of Things: A Re-

view on the Techniques and Challenges for Efficient Vulnerability Discovery in Embedded Systems. *IEEE Internet of Things Journal*, **8**, 10390-10411. https://doi.org/10.1109/JIOT.2021.3056179

[9] Xu, R., Sha, E.H.M., Zhuge, Q., Song, Y., Wang, H. and Shi, L. (2022). Optimizing Data Placement for Hybrid SRAM+ Racetrack Memory SPM in Embedded Systems. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, **42**, 847-859. https://doi.org/10.1109/TCAD.2022.3185548

[10] He, G., Michalek, J., Kar, S., Chen, Q., Zhang, D. and Whitacre, J.F. (2021) Utility-Scale Portable Energy Storage Systems. *Joule*, **5**, 379-392. https://doi.org/10.1016/j.joule.2020.12.005