Hindawi
Scientific Programming
Volume 2022, Article ID 1880811, 12 pages
https://doi.org/10.1155/2022/1880811

Research Article

Discussion on Application of Embedded Operating System in Dual-Core Smart Electric Meter

Xuesong Zhao,[1] **Bensong Zhang**,[2] **Shihong Yin**,[1] **Qianxian Xie**,[1] **Leping Zhang**,[2] **and Haowen Wu**[3]

[1]*Metrology Management Institute, Shenzhen Power Supply Bureau Co., Ltd., Shenzhen, Guangdong 518001, China*
[2]*Metrology Center, CSG Digital Grid Research Institute Co., Ltd., Guangzhou, Guangdong 510700, China*
[3]*Communication and Internet of Things Division, Digital Grid Branch of CSG Digital Grid Research Institute Co., Ltd., Guangzhou, Guangdong 510700, China*

Correspondence should be addressed to Xuesong Zhao; xuesongzhao2021@163.com

Received 17 November 2021; Revised 1 December 2021; Accepted 7 December 2021; Published 18 January 2022

Academic Editor: Le Sun

The new generation of smart meters need to realize the requirements of flexible expansion of advanced applications and transplantation of application software across hardware platforms. The software platform composed of operating system and middle layer service components is conducive to reducing the cost of software and hardware development and management. This paper first introduces the research and development background of the new generation smart meter, then introduces its overall software and hardware architecture and management unit hardware development platform, then introduces the construction of development environment and development tools and the basic functions based on the software platform, and finally introduces the development practice of application software. The large-scale promotion of the new generation of smart meters has become a development trend. This paper has a good reference for the application software development based on the new generation of smart meter software platform and is conducive to promoting the large-scale application of the new generation of smart meters.

1. Introduction

With the development of social economy, the demand for electric power in various fields is increasing rapidly. Meanwhile, the demand for power supply reliability, security, power quality, and power service is becoming higher and higher. For power supply enterprises, to give full play to the potential of power equipment, to provide low-cost, high-quality, safe, reliable electricity and high-quality comprehensive services, is an effective means to improve the competitiveness of enterprises and promote economic benefits. In order to achieve a win-win situation between power users and power supply enterprises, it is necessary to mobilize the subjective initiative of tens of millions of power users to use electricity reasonably and save electricity. At the same time, it is also necessary for power enterprises to continuously optimize the structure and realize the rational allocation of electric energy. In order to achieve a win-win situation, many countries and organizations have proposed the construction of flexible and standardized, clean and friendly, safe and economic smart power grid, and regard the construction of smart power grid as the primary direction of future power grid construction. As the terminal of smart grid extending to users, smart electricity meter is the key link between users and power enterprises. In order to build a sound and mature smart grid, the current management mode of electric energy distribution and use is changing from extensive to intensive, just as the current implementation of stepped electricity price, peak-valley electricity price, distributed generation, and other measures are the embodiment of this change. Intelligent electric energy is an important carrier of technological innovation, so every change of electricity management mode is promoting the renewal of electricity meters. In order to adapt to this trend

and achieve long-term utilization of resources, it is urgent to launch a universal smart electricity meter that can realize power management mode transformation without replacing hardware but only updating software.

With the rapid implementation of China's smart power grid and digital power grid construction, as an important part of electricity consumption, the number of smart meters keeps growing rapidly. By the end of 2019, the number of smart meters in China has exceeded 600 million.

Under the situation of increasing pressure on global resources and environment, how to allocate energy resources scientifically and rationally, achieve maximum utilization of energy resources, and at the same time ensure the healthy development of energy resources is a difficult problem faced by all countries. In order to promote the implementation of IR46 standard in China [1], China's JJF1245 series technical specifications will be officially released and implemented. This series of technical specifications clearly divide the legal and nonlegal measurement functions of the new generation smart meters and also define the basic technical framework for the software and hardware design of the new generation smart meters, as shown in Table 1 [2].

The concept of smart electricity meter was put forward as early as the end of the twentieth century, but the characteristics of the early smart electricity meter are mainly reflected in the basic functions such as remote centralized meter reading and digitalization. At present, the proposal of smart power grid and AMI model has promoted the function and characteristics of smart electricity meter to an unprecedented height. At present, smart electricity meter has developed from a simple electric energy metering and billing device to an intelligent device integrating measurement, execution, communication, fault protection, and response. As the core component in the process of smart grid construction, it will have broad application prospects. Therefore, relevant enterprises and research institutions at home and abroad have attached great importance to the research and development of multifunctional smart meters. Many smart chip suppliers have also launched a variety of smart meter solutions, and a large number of smart-meter-related literature has also emerged. Some power supply and production enterprises also jointly issued a lot of smart-meter-related standards, and a variety of smart meter products appear in the market. At present, smart electricity meters at home and abroad can be divided into multifunction meters, compound rate meters, prepaid meters, carrier meters, network meters, etc., and can also be combined according to user requirements.

2. New Generation Smart Meter

The new generation smart meter needs to meet the requirements of IR46 international standard and JJF1245 technical specification. Therefore, this paper puts forward the software and hardware architecture design scheme of the new generation smart meter as shown in Figure 1.

It is mainly to meet the needs of two aspects: one is to achieve the separation of legal measurement function (measurement core software) and nonlegal function (management core software) [3] and, second, to support independent apps to flexibly meet differentiated functional needs such as load identification, orderly charging, and contract power purchase, and support the construction of smart electricity meter application market ecology. The embedded operating system is introduced into the management unit, which forms the software platform of the new generation smart meter together with the middle-tier components. The application software is completely decoupled from the hardware platform, thus achieving the purpose of simplifying application development and adapting to iterative upgrade of applications [4].

3. Hardware Design of New Generation Smart Meter Management Unit

The hardware architecture of the new generation intelligent watt-hour meter as shown in Figure 2 is mainly divided into three parts: communication unit, management unit, and metering unit [3, 5]. The following figure shows the hardware schematic diagram and prototype of the management unit, and the main key components are the management MCU chip. The MCU chip of the management unit is a STMicroelectronics product with Cortex-M4 core, with the highest frequency of 100 MHz. On-chip RAM capacity is 320K, and on-chip FLASH capacity is 1M. Five UART ports, two SPI ports, and one 7816 port can support up to 24 GPIO, and the chip package is LQFP64.

The management unit nonvolatile memory is divided into EEPROM, on-chip flash memory, and off-chip flash memory. EEPROM is accessed through the device file, and the application reads and writes data according to the address in the device. The on-chip flash memory is managed by the read-only file system flashfs. Off-chip flash memory is divided into two parts: one is directly read and written by applications using bare sector devices, and the other is managed by power-off secure file system littlefs [6–8].

To meet the needs of users and power supply enterprises, in addition to the original metering and data display functions, the new generation of smart meters also needs to have two-way communication, remote intelligent indoor tripping protection, and good human-computer interaction functions. Higher design requirements for the smart meter management unit are proposed for scalability, intelligence, and adaptability [9–12]. On-chip FLASH storage files include bootloader boot program; the starting address is the running address after power-on; Rtos image, started by bootloader; Start parameter, rtos starts service components and basic applications according to this file, and allocates resources and permissions for them; Service component and APPlication app in middle layer [13–15]. Off-chip LFLASH storage files include application backup, application to be upgraded, power freeze, and event data. Off-chip EEPROM mainly stores application data. And it also supports a number of LCD module interfaces that have these peripheral hardware resources to meet the design requirements of the terminal [12, 14].

Table 1: Function division of legal software and nonlegal software of new generation smart meter.

Functional item	Legal software part	Nonlegal software part
1	Measurement of electrical parameters (such as voltage, current, active power, reactive power, and frequency)	Calculation, storage, and freezing of electric energy rate (including setting of rate and time period)
2	Measurement of electrical energy (e.g., active energy and reactive energy)	Calculation and storage of maximum demand (including the maximum demand of each rate)
3	Or clock maintenance	Display mode and display of nonlegal related content
4	Protection of proprietary parameters of equipment	In addition to the legal system related functions of the event record
5	Freezing of electrical parameters and energy data	Implementation of external data communication interface (such as infrared, RS485, carrier, wireless, etc.)
6	Event records of clearing, upgrading, timing, and modifying equipment-specific parameters	Control function (such as control of load switch)
7	Software interface between legal related software and non-legal related software	Alarm function
8	Methods to ensure the safety of functions related to legal system	Cost control function
9	Identification of legal-related software	Other functions not mentioned
10	Display of legal related contents	—

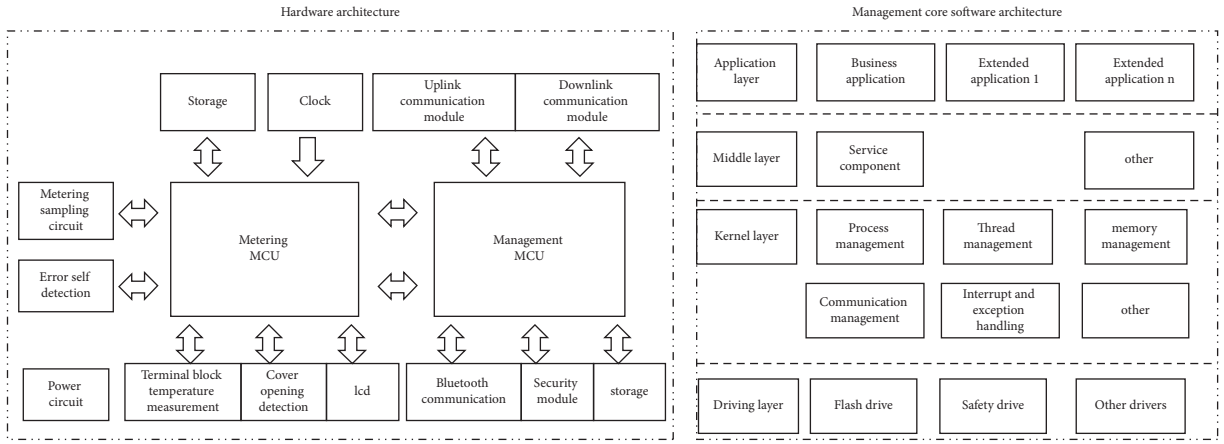

Figure 1: Typical application scheme of new generation smart meter.

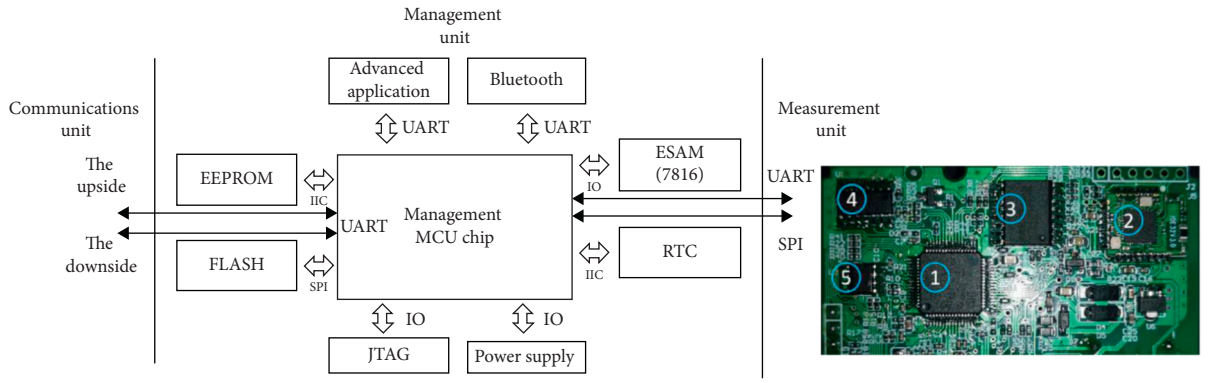

Figure 2: Hardware schematic diagram and prototype of management unit.

4. New Generation Smart Meter Management Unit Software Platform

The software of the new generation smart meter management unit is hierarchically designed as kernel layer, middle layer, and application layer, as shown in Figure 3. The application layer is based on the software platform built by the core layer and the middle layer to realize the specific business functions of the smart meter, including a basic application and multiple extended applications.

The kernel layer mainly includes the operating system kernel, device drivers, and many specific functions built by

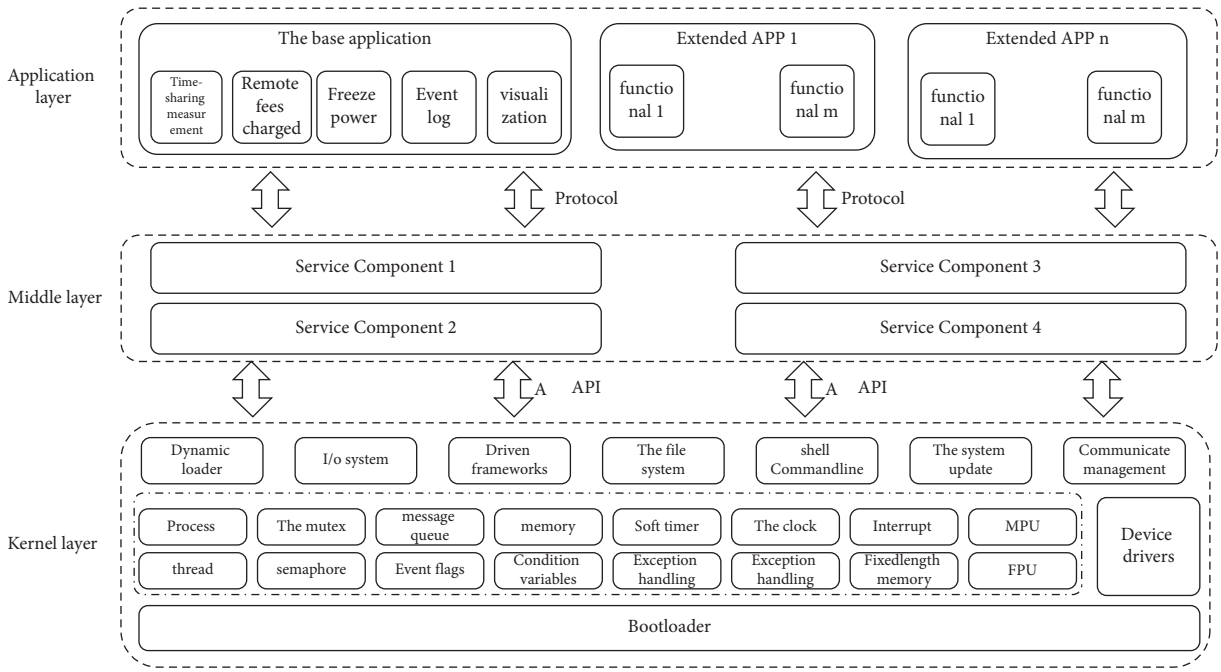

Figure 3: Snap-in software platform.

the operating system. These specific functions are customized according to the requirements of smart meters to realize task scheduling and resource allocation and have the characteristics of standardized programming interfaces, extensible function modules, and portable operating system kernel [8–10, 16]. The middle layer realizes the decoupling between the operating system and the electricity meter service and improves the stability and universality of the operating system. The middle layer runs on the system kernel and provides common services and component support for the application layer. This scheme mainly includes virtual bus service component, communication management service component, measurement management service component, and platform management service component as shown in Figure 3.

5. Development Environment Building and Development Tools

Software is the soul of a system; if a system is without software, then its hardware circuit will lose the value of existence. Only by combining the two can the system function be realized. In the actual system, if a software system with perfect functions and excellent performance is configured for the hardware circuit, it can not only make up for some defects of the hardware circuit, but also greatly reduce the design cost [17]. More importantly, only in this way can the hardware give full play to its functions [18].

This system design chooses the embedded system with the operating system because the embedded system with the operating system is better than the embedded system without the operating system in real time, tailoring open and scalability, and strong stability. At the same time, compared with the embedded system without operating system, the embedded system with operating system is also portable and deterministic.

To develop application software based on the new generation smart meter software platform, the development environment should be set up first, which mainly includes integrated development environment software, new generation smart meter prototype (including management unit), Jlink simulator, and upper computer debugging software as shown in Figure 4, realizing the adaptation of operating system to hardware platform, complete the development of hardware interface and peripheral driver, and realizing the transplantation and testing of software system. Finally, the application software is developed based on the API interface provided by the operating system and the intermediate layer protocol data structure.

The integrated development environment software (IDE) used in this paper is IoT Studio, and its main functions are RTOS project management and program debugging as shown in Figure 5. IoT Studio can create RTOS Base project, RTOS Bsp project, RTOS App project, RTOS App Static Lib project, RTOS Kernel Static Lib project, and RTOS Unit Test project. Support downloading and debugging BSP project and App project with one click of J-Link. IoT Studio mainly integrates the following development tools as shown in Figure 5.

(1) Compiler tool chain optimized for platform
(2) Intelligent code editor
(3) Code coverage analysis tool
(4) Code static analysis tool
(5) Support debugging tools through emulators such as J-Link

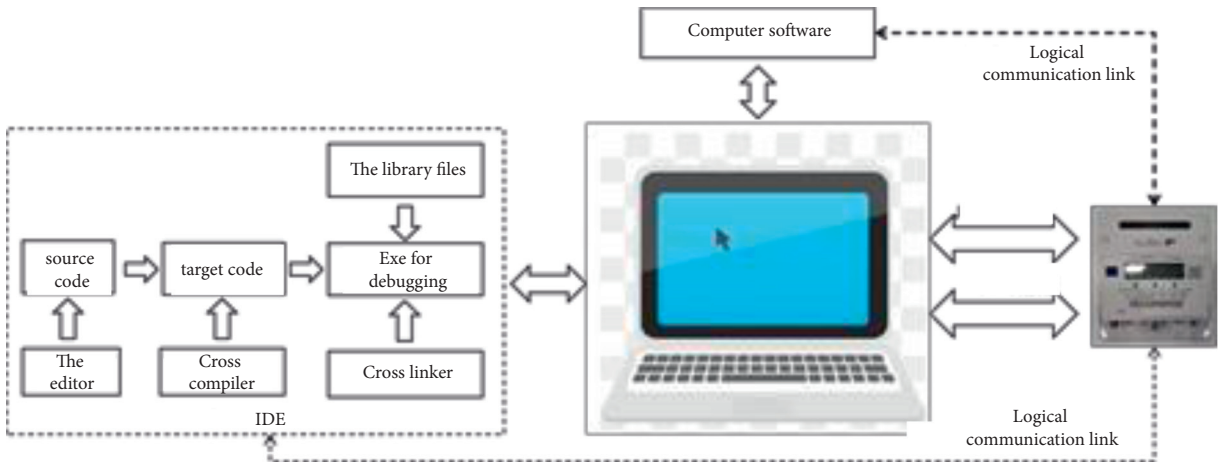

Figure 4: Embedded application software cross development.

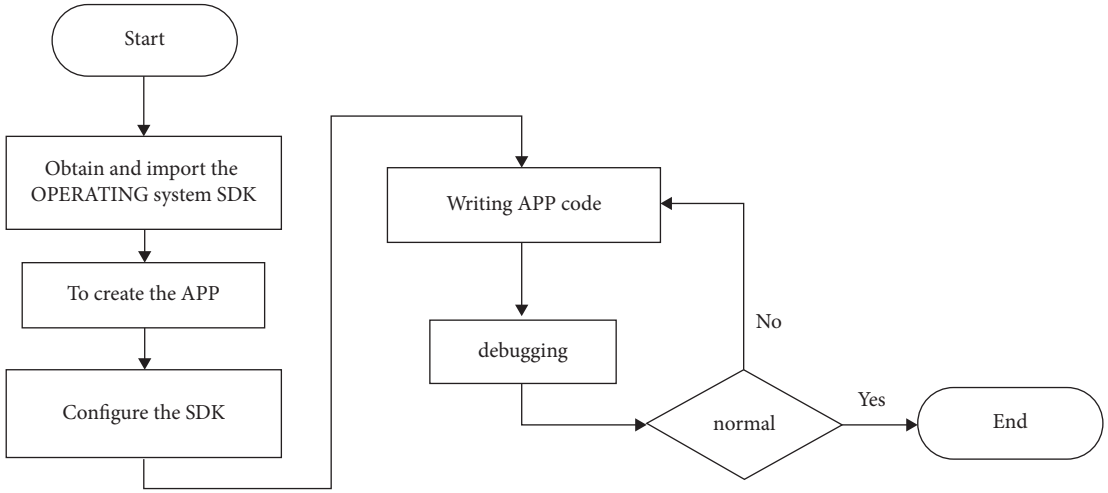

Figure 5: IDE usage flow.

RTOS Image Packager is an image of RTOS operating system and an application packaging tool. The tool specially provides the factory firmware generation function, which can package the BootLoader image, operating system image, startup parameters, and application image into a factory firmware. In the mass production stage, the packaged factory firmware only needs to be burned into Flash as shown in Figure 6.

RTOS AutoTester is an automatic testing tool for RTOS operating system. The tool integrates the functions of firmware writing, Shell terminal, and modem protocol file sending and has the function of C language automatic test script execution as shown in Figure 7.

6. Basic Functions of Software Platform

The platform provides rich functions for application software development. The core functions are process management, thread management, interprocess communication, thread synchronization and communication, clock and soft timer management, interrupt/exception management, multiprocess security isolation, dynamic loading, and low-power consumption management [19]. The extended functions are mainly memory management, system update, file system, IO system and drive framework, etc., and can be extended according to actual requirements. Among them, kernel layer and kernel component layer work in CPU privilege mode, while middle layer components and application layer work in CPU user mode. The following mainly introduces the common functions and API interfaces of application software development.

6.1. Process Management. It is the smallest unit for operating system to allocate resources. Common API interfaces are ms_process_self () to get the current process ID, ms_process_kill () to kill the specified process, ms_process_exit () to voluntarily exit, and ms_process_find () to find the process.

Process management mainly has the following functions:

(1) The number of processes is configurable

(2) Process memory resources can be configured (minimum 4K, 2 to the nth power kb)

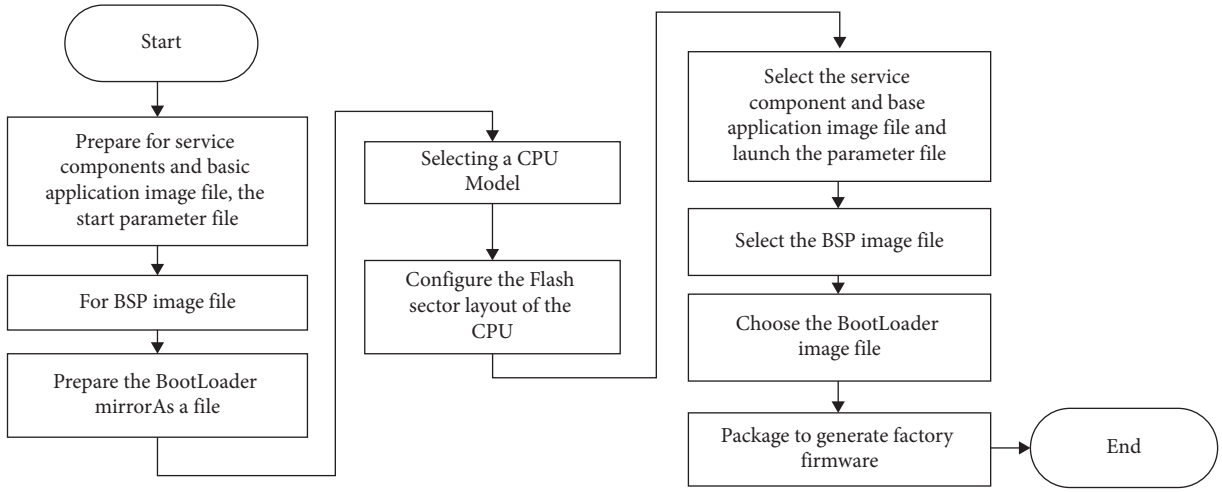

Figure 6: Use process of software packaging tools.

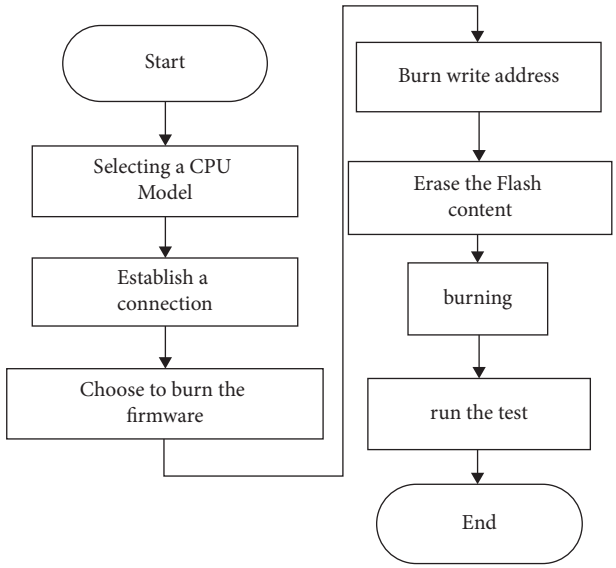

Figure 7: Use process of test tools.

(3) Process kernel resources can be configured (maximum number of threads, number of kernel objects, and number of open files)

(4) Process permissions can be configured (EEPROM access space, privileged system call)

(5) Process address spaces are isolated from each other

6.2. Thread Management. Thread is the basic unit of task scheduling in operating system. Commonly used API interfaces are as follows: ms_thread_create () creates a ready thread, ms_thread_init () creates a suspended thread, ms_thread_self () gets the ID of the current thread, ms_thread_kill () kills a thread, ms_thread_exit () the current thread voluntarily exits, the ms_thread_suspend () suspends a thread, the ms_thread_resume () resumes a thread, and the ms_thread_yield () current thread gives up the CPU usage right. Thread management mainly has the following functions:

(1) Support 64 priorities

(2) Support high priority preemption and time slice rotation scheduling with the same priority

(3) Decompose tasks and simplify execution logic

6.3. IO System. IO system as shown in Figure 8, also known as I/O system, isolates the direct access of applications to hardware devices and unifies various hardware access interfaces. The replacement of hardware devices no longer needs to modify the application code, but only needs to add the corresponding hardware drivers in the operating system. Commonly used API interfaces are ms_io_creat () file, ms_io_open () file, ms_io_close () file, ms_io_read () file, ms_io_write () file, ms_io_ioctl() IO control, ms. IO system mainly has the following functions:

(1) All devices can be accessed as files

(2) File operation interface supports POSIX flag

(3) C library file interface is supported

(4) Provide a more resource-efficient system interface

6.4. IPC. The interprocess communication mechanism as shown in Figure 9 involved in this paper mainly includes named pipe and shared memory. Common API interfaces are as follows: ms_pipe_dev_create () creates a pipeline device, ms_shm_dev_create () creates a shared memory device (kernel space), ms_io_open () opens a pipeline or shared memory device, ms_io_read () reads pipeline data, and ms_io_write () writes.

6.5. Service Data Routing Distribution. The virtual bus service component is the data flow center, and each service and APP need to distribute data through the virtual bus, so as to realize the communication and data interaction between each service and app. The main functions are as follows: safe and efficient well-known read/write pipeline is adopted for interprocess communication, data routing and forwarding of interprocess protocol, support for

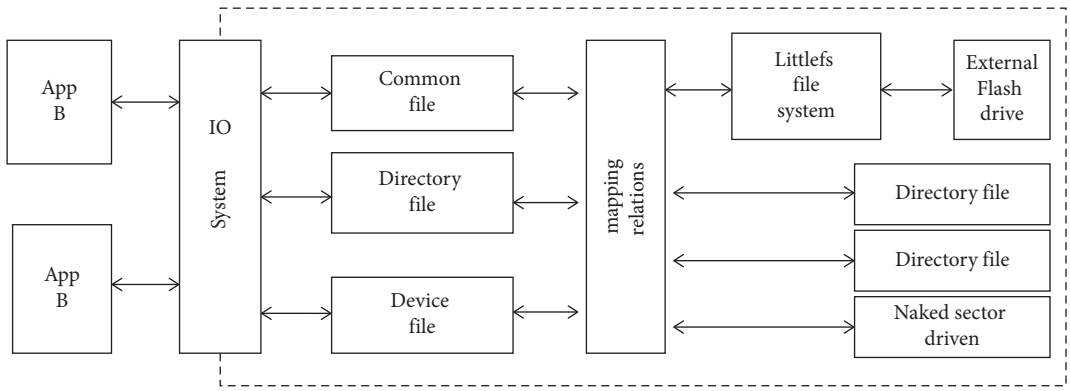

Figure 8: IO system.

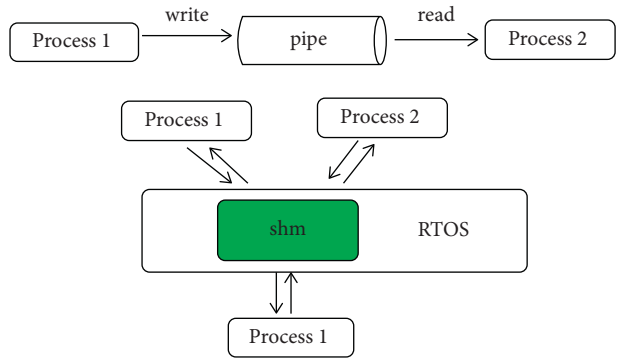

Figure 9: Pipeline and shared memory communication.

registration of DL/T645 extended protocol data identifier, support for pipeline registration of extended APP, and support for combination data reading. The format of protocol data frame is shown in Table 2.

The source address and destination address coding rules are shown in Table 3.

Description of interface between virtual bus component and each service component and application is shown in Table 4.

7. Application Software Development Practice

This paper only introduces the basic application APP in detail as a concrete practice, which needs to meet the Technical Requirements of the New Generation Single-phase Smart Energy Meter in South Power Grid, including basic functions, extended functions, dual-core interactive functions, and data security functions.

The architecture of the basic application app as shown in Figure 10 is divided into six modules, which are the basic application app virtual bus module, time-sharing measurement module, display module, meter reading freezing module, event recording module, and control module. The APPlication APP virtual bus module realizes the internal data interaction of each functional module of the basic application app and realizes the interaction between the basic application app and external data; the time-sharing metering module realizes time-sharing rate electricity metering, ladder electricity metering, and rate electricity synchronization after the management core is plugged and unplugged. The meter reading freezing module realizes five types of electricity data freezing, completes multiple backups, and realizes supplementary freezing. The display module realizes display functions such as tracking display and organizes display data for use by the metering core. The event recording module records various events according to the configuration and stores them in EEPROM. The control module realizes the functions of remote charge control, opening and closing, self-protection, and so on. The following mainly introduces the main workflow of the basic application APP virtual bus module, freezing module, and display module.

7.1. Virtual Bus Module. As shown in Figure 11, the data from outside is distributed to the other five modules through the basic application app virtual bus module.

(1) Create a thread to monitor, and receive and analyze the data distributed by virtual bus. In order to ensure the real-time reception, select function is used to monitor and avoid polling and sleep waiting.

(2) The received data is judged and distributed to specific modules through control words and data identification, and the API interface of

Table 2: Protocol data frame format.

name	Meaning	Member
agrform_t	Communication frame header structure between 4 services and basic application app	FristSepa: identifier, fixed as 0×68 CAID: source address, address of initiator Said: destination address, response module address encryption: encryption method MetersAddr: type of protocol communication address AgreType: protocol type ComType: command type overRbye1: and verify that all save fields are reserved to use the length of DataLen: uplink/downlink 645 protocol frame
AgrDataBuf	Frame content data	Complete 645 frames of data of uplink/downlink 645 protocol (from the starter 68 to the terminator 16)
pipe_t	Pipe type	The trend of data, such as V_BUS_TO_COM_SRV, indicates the pipeline from virtual bus to communication management service, virtual bus receiving pipeline, and communication management service sending pipeline

Table 3: Source address and destination address coding rules of protocol data frames.

bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Process number					Every thread number in APP, 0–7		
0: initial value					0: main thread		
1: basic APP					Basic APP		
2: communication service					0: general basic application app		
3: metering management service					1: time-sharing measurement		
4: platform management service					2: demand measurement		
5: extended app1					3: freezing		
6: extended app2					4: display		
7: extended app3				Reserve	5: cost control		
8: expand app4					6: event		
9: expand app5					Communication APP		
10: expand app6					1: uplink carrier		
11: extend app7					2: downlink Bluetooth		
12: extend app8					3: load identification		
13: extend app9					4: other communications (continuous coding)		
14: extend app10							
15: virtual bus service					Other processes		

corresponding modules is called. There are three API interfaces: get, set, and action.

(3) When each module actively requests data or actively operates a task, it judges whether it is internal module data. If it is internal module data, it directly calls the get, set, and action functions of the basic application app, calls the API interface of the corresponding module, and replies to the data.

(4) If each module actively requests external data, it needs to call the get, set, and action functions encapsulated by the basic application app, write the external pipeline of the basic application app, and obtain data through the virtual bus. After the virtual bus obtains the data, it returns the data to the basic application app through the pipeline, and the basic application app sends it back to each module through the message queue. Set, get, and action functions encapsulated by the basic application app package the message queue internally, which is used for the interaction between the basic application app general thread and each module thread as shown in Figure 11.

7.2. Freeze Module. According to the interaction with other modules of basic application APP, the freezing module can be divided into two parts: the other module reads the data of freezing module and the freezing module obtains the data of other modules. The running process of frozen thread is shown in Figure 12.

Table 4: Interface description between virtual bus components and various service components and applications.

Interface function	Description	Input	Operate	Output
Pipeline function	The virtual bus automatically creates a pipeline with other apps and services.	Pipeline no.	Judging whether the pipeline exists or not	Do all communication channels exist?
		Pipeline communication frame (CA and SA are consistent, single frame, multiframe, small data frame, and large data frame)	Send the pipeline communication frame to the virtual bus service through the corresponding pipeline	The virtual bus is distributed through the corresponding pipeline data
Virtual bus data distribution function	The virtual bus performs correct distribution processing according to the data identification and control code in the data frame	The data identification of metering service management constitutes the pipeline communication frame	Send the pipeline communication frame to the virtual bus through the corresponding pipeline	Whether the virtual bus distributed data is distributed through the pipeline of metering management service, comparing the integrity and legitimacy of distributed data
		The data identifier managed by the platform management service constitutes a pipeline communication frame		Whether the virtual bus distributed data is distributed through the pipeline of platform management service, comparing the integrity and legitimacy of distributed data
		The data identifier of the management service management constitutes a communication frame		Whether the virtual bus distributed data is distributed through the pipeline of communication management service, comparing the integrity and legitimacy of distributed data
		The data identifier managed by the basic app is used to form a pipeline communication frame		Whether the virtual bus distributed data is distributed through the pipeline of basic application app, comparing the integrity and legitimacy of distributed data
		Compose pipeline communication frame through transparent transmission control code		Whether the virtual bus is forwarded to the corresponding extended app pipeline according to the object code SA of the extended app, and the integrity and legitimacy of the distributed data are compared
The virtual bus supports the extended app registration function	According to the extended app registration information, the virtual bus creates a corresponding pipeline and stores the extended app registration information	The extended app registration information constitutes a pipeline frame, and the extended app registration information is read to constitute a pipeline frame	Send the pipeline frame to the virtual bus through the extended app reservation registration pipeline	The virtual bus establishes the corresponding extended app communication pipeline according to the registration information and returns the stored extended app registration information

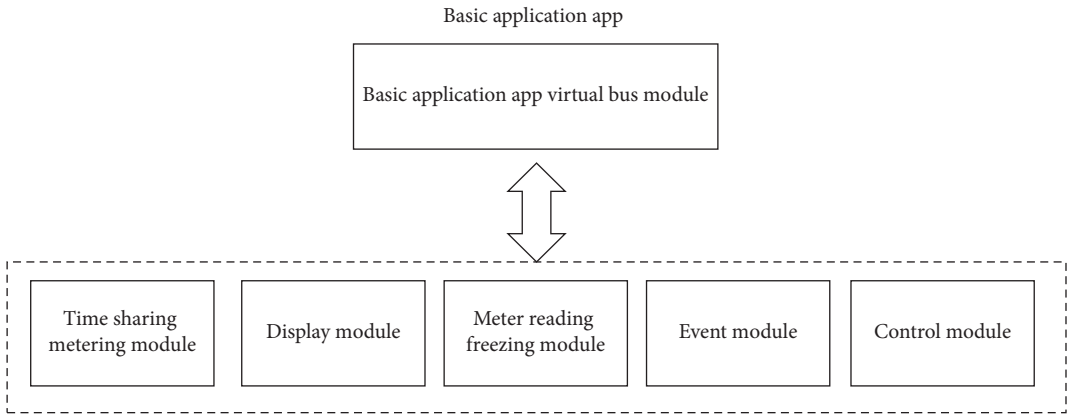

Figure 10: Software architecture of metering basic application APP.

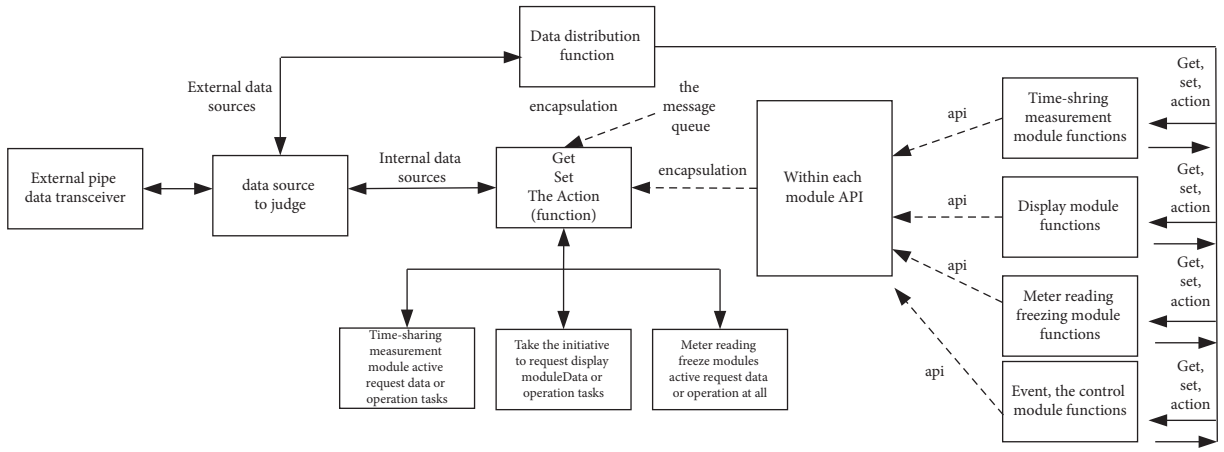

Figure 11: Virtual bus module data distribution process.

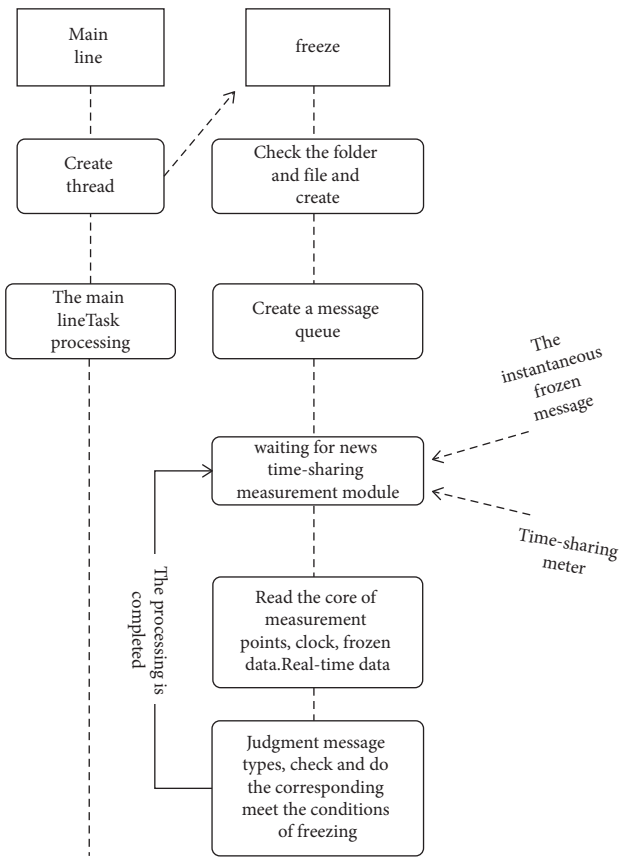

Figure 12: Freezing thread running process.

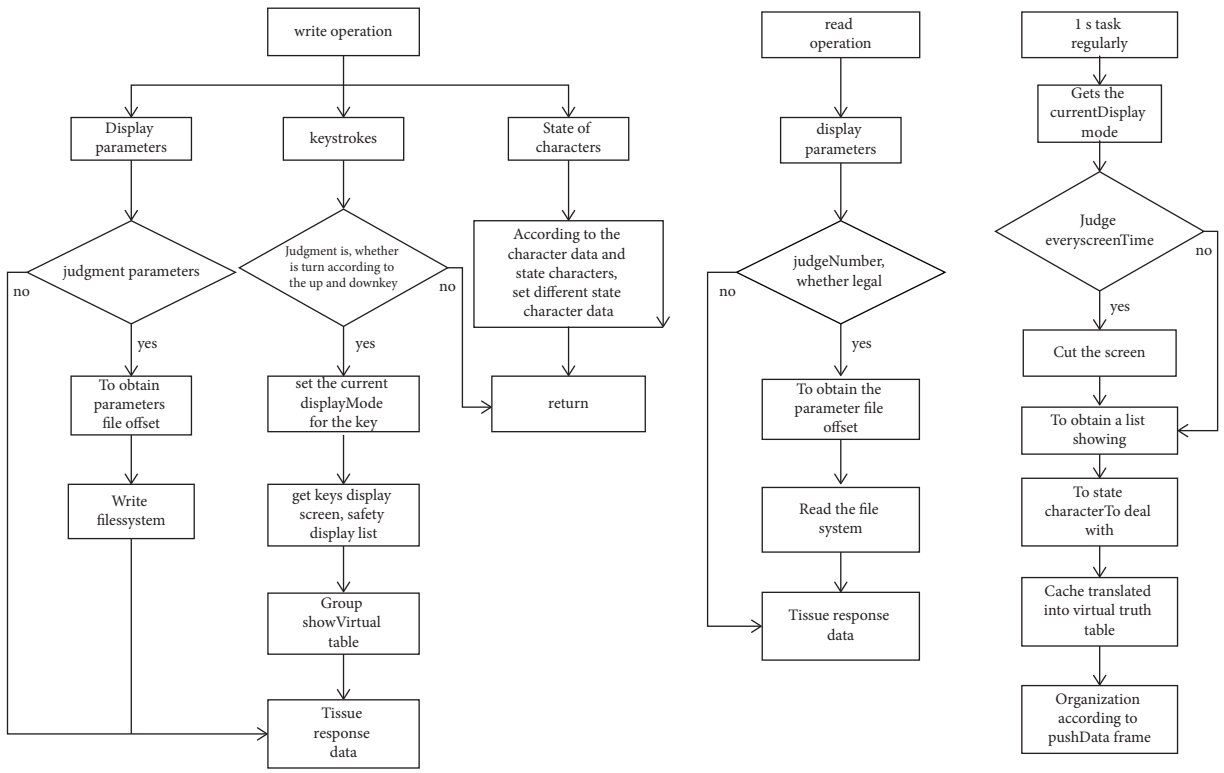

Figure 13: Display thread write operation, read operation, and display processing flow.

7.3. Display Module. The display module mainly includes three parts: writing operation of the display module, reading operation of display parameters, and 1s timing display data processing. The running process of the display thread is shown in Figure 13.

8. Conclusion

Based on the new generation smart meter software and hardware platform, this paper builds an application software development environment by using IDE and related tools and puts forward the concrete practice of business application software development, which has good reference significance for other application software development based on the new generation smart meter software and hardware platform. The simple development and quick iteration of the application software are conducive to promoting the large-scale application of the new generation of smart meters and have laid a solid technical foundation for meeting the differentiated business needs of the new generation of smart meters within their 15-year service life and building a grand application ecological market in the future [11–19].

Data Availability

The labeled dataset used to support the findings of this study is available from the corresponding author upon request.

Conflicts of Interest

The authors declare no conflicts of interest.

Acknowledgments

This study was supported by the Research Project of China Southern Power Grid Co., Ltd. (JY-O-JL-20-002)

References

[1] M. Liu, L. Zhou, C. Miao, W. Wu, and J. Xiang, "Discussion on the development of a new generation of intelligent electric energy meters," *Electrical Measurement and Instrument*, vol. 54, no. 18, pp. 94–100, 2017.

[2] Jjf1245-2019, *Type Evaluation Outline of Mounted AC Electric Energy Meter*, State Administration of Market Supervision, Beijing, China, 2020.

[3] Z. Zhou, D. Zhao, H. Zhang, J. Du, and W. Wang, "Research on key technologies of dual-core watt-hour meter master control based on IR46 standard," *Application of Electronic Technology*, vol. 43, no. 10, pp. 7–11+19, 2017.

[4] J. Li, S. Bai, and Z. Wang, "Smart meter based on MQX operating system," *China Science and Technology Information*, vol. 01, pp. 50-51, 2016.

[5] X. Ou, Y. Zeng, X. Ran, X. Wang, and X. Ou, "Research on software reliability design and test of IR46 intelligent electric energy meter," *Electrical Measurement and Instrument*, vol. 56, no. 19, pp. 147–152, 2019.

[6] J. P. Sun, *Research and Implementation of Embedded File System Based on Flash Memory*, xidian university, Xi 'an, China, 2008.

[7] T. Ling, Y. Xu, S. Shen, and R. L. Zhang, "A design method of embedded file system with power-off security," *Application of Single Chip Microcomputer and Embedded System*, vol. 5, pp. 33–36, 2019.

[8] Gj/b7718-2012, *Technical Requirements of Military Embedded Operating System*, General Armament Department of Chinese People's Liberation Army, Beijing, China, 2013.

[9] J. Cheng, J. Lou, Y. Cao, and M. Xu, "A general test software architecture design for avionics products," *Measurement and Control Technology*, vol. 39, no. 4, pp. 33–37, 2020.

[10] X. Zhang, *Research and Implementation of Key Technologies of Embedded System of Internet of Things Based on Mbed*, Shenyang Institute of Computing Technology, Chinese Academy of Sciences, Shenyang, China, 2018.

[11] L. Gao, Z. Wang, J. Zhou, and C. Zhang, "Design of smart home system based on ZigBee technology and R&D for application," *Energy and Power Engineering*, vol. 08, no. 1, pp. 13–22, 2016.

[12] F. L. Huang and S. Y. Tseng, "Predictable smart home system integrated with heterogeneous network and cloud computing," in *Proceedings of the 2016 International Conference on Machine Learning and Cybernetics (ICMLC)*, July 2016.

[13] Y. H. Lin, "A cloud analytics-based electrical energy management architecture empowered by edge analytics AIduino with push notifications for demand-side management," in *Proceedings of the 2019 IEEE 2nd International Conference on Power and Energy Applications (ICPEA)*, Singapore, April 2019.

[14] A. Muti and L. Fatzinger, "System And method for scheduling posts on a web site," 2013.

[15] M. H. Yaghmaee Moghaddam and A. Leon-Garcia, "A fog-based internet of energy architecture for transactive energy management systems," *IEEE Internet of Things Journal*, vol. 5, no. 2, pp. 1055–1069, 2018.

[16] H. H. Chang, C. L. Lin, and L. S. Weng, "Application of artificial intelligence and non-intrusive energy-managing system to economic dispatch strategy for cogeneration system and utility," in *Proceedings of the International Conference on Computer Supported Cooperative Work in Design*, May 2009.

[17] L. Yu-Hsiu and H. Yu-Chen, "Residential consumer-centric demand-side management based on energy disaggregation-piloting constrained swarm intelligence: towards edge computing," *Sensors*, vol. 18, no. 5, p. 1365, 2018.

[18] H. Hafiz, J. Nadeem, I. Sohail, U. H. Qadeer, A. Khursheed, and A. Musaed, "An efficient demand side management system with a new optimized home energy management controller in smart grid," *Energies*, vol. 11, no. 1, p. 190, 2018.

[19] T. Ming, J. Zuo, Z. Liu, A. Castiglione, and F. Palmieri, "Multi-layer cloud architectural model and ontology-based security service framework for IoT-based smart homes," *Future Generation Computer Systems*, vol. 78, pp. 1040–1051, 2016.