Research on the technology of Smart Energy Meter integrating time-sharing Metering and Billing

Xingzhi Liu¹, Quan Zhou², Qing Qin¹, Yongming Xie³ and Yi Pan³,*

¹Marketing Service Center, State Grid Chongqing Electric Power Company, Chongqing, China
²Electric Power Research Institute, State Grid Chongqing Electric Power Company, Chongqing, China
³Holley Technology Co., Ltd., Hangzhou, China

*Corresponding author: meterming@holley.cn

Abstract. As one of the most basic and most critical devices in the perception layer of the ubiquitous power Internet of Things, smart energy meters require high reliability, low cost, and low power consumption, as well as certain functional expansion capabilities. This article mainly focuses on the smart energy meter integrating the metering and billing functions with variable tariffs and flexible and expandable electricity price packages, and analyzes the requirements for MCU instruction set, kernel structure, and SOC peripherals of this type of smart energy meter when working with a lightweight microkernel real-time operating system. And put forward the key design of the main frequency and large-capacity storage, which can meet the basic functions of the smart energy meter, while meeting the needs of flexible and expandable functions.

1. Introduction
The smart energy meter has evolved from a single-function electric energy measuring instrument in the traditional sense to an intelligent terminal integrating multiple functions such as metering, freezing, communication, monitoring, charge control, and load control. In view of the experience accumulated in the recent 10 years of large-scale implementation of the smart energy meters of State Grid Corporation and China Southern Power Grid, the implementation of IR46, the integration of user-side distributed energy and micro-grid energy storage, the diversification of prices on the sales side, the charging of electric vehicles, and the operation and maintenance of complex metering systems, etc. on demand, functional requirements such as separation of legal system, legal protection, fine-grained large-capacity storage, high-speed two-way communication, modular upgrade, diversified billing and settlement, distributed energy and load control are proposed for smart energy meter.

Practical experience tells us that the more functions of the smart energy meter, the more complex the software and the more faults. According to statistics, software failures in existing smart energy meters account for 70%. Moreover, the operation and maintenance of software failures is more difficult. Therefore, while meeting the functional requirements, it is necessary to design a "layering and sub-module" scalable and robust software structure based on the idea of "high cohesion and low coupling"[1] to support the realization and expansion of more and more functions. On the other hand, the
measurement part of the legal system is separated from the management part to ensure the independence, solidification and reliability of the legal measurement part. At the same time, it also takes into account the complex functions and flexible expansibility of illegal metering part. In particular, the metering and billing functions are integrated, and the billing functions are complex, diversified and changeable. In this way, it is necessary to break the existing software and hardware framework of smart energy meters and explore a framework with forward-looking, interoperability and scalability.

This paper analyzes the technical characteristics of the smart energy meter in the competitive electricity retail market and the grid-load interaction mode, and studies the new single/three phase energy meter of metering and billing integration based on IR46 international recommendations and adapted to the competitive electricity retail market and the grid-load interaction model. The selection of the smart energy meter management core MCU and associated design, as well as the requirements and selection of the embedded operating system are carried out.

2. Overview of design scheme of smart energy meter
The smart energy meter integrating time-sharing metering and billing is modified on the basis of the independent smart energy meter of time-sharing metering and billing. The main modifications are as follows:

1) Reselect the management core MCU on the hardware, adjust the MCU IO port, FLASH plans to use the Qual SPI interface [2] and high-speed wiring. Others use the original entire metering core circuit, power supply and management core peripheral circuit.

2) The software management core is changed to a lightweight operating system as the software framework, and the application layer does not change the function to reconstruct the original front-end system code according to the operating system programming interface and application modularization, and design and develop the billing application module.

3. Selection of management core MCU for smart energy meter and associated design

3.1. Instruction set system selection
The strength of the instruction is an important indicator of the CPU. From the current mainstream structure, the instruction set can be divided into two parts: complex instruction set computer (CISC) and reduced instruction set computer (RISC). The advantages and disadvantages of the two instruction set systems are as follows:

1) CISC adopts the design of microinstruction code control unit due to the complexity of the instructions, while 90% of the RISC instructions are directly completed by the hardware, and only 10% of the instructions are completed by the software in a combined manner. Therefore, the instruction execution time is shorter in RISC, but the ROM space required by RISC is relatively large.

2) CISC needs more addressing modes, while RISC has only a few addressing modes. Therefore, when the CPU calculates the effective address of the memory, CISC takes more cycles.

3) The format of the CISC instruction varies in length, and the number of cycles during execution is not uniform. The RISC structure is just the opposite, so it is suitable to adopt the design of pipeline processing structure, which can achieve an average of one cycle to complete an instruction.

4) RISC is simpler than CISC in design. At the same time, because CISC has too many execution steps, the waiting time of idle unit circuits increases, which is not conducive to the design of parallel processing, so RISC is better than CISC in terms of performance.

By analyzing the instruction set system, a simple, high-efficiency, low-power RISC is more suitable as the main chip of the management core.

The RISC structure instruction set is divided into ARM, MIPS, PowerPC, RISC-V and other instruction structures. Among them, ARM’s Thumb instruction set has become the best instruction set structure for embedded CPU due to its small size, low power consumption, low cost, and high performance. Moreover, the price of the MCU with the 32-bit core of the Cortex-M series with Thumb instruction set is comparable to that of the traditional 8-bit microcontroller, but the performance is much
stronger than that of the 8-bit computer, so it is very suitable for the application of the embedded microcontroller.

3.2. **MCU core selection**

CPU is divided into John Von Neumann structure and Harvard structure according to the structure. The John Von Neumann structure is a structure that stores instructions and data at different addresses in the same memory, using a single instruction/data bus. The Harvard structure is a memory structure that separates instructions and data storage. Instructions and data use independent data buses. The CPU of the Harvard structure is relatively complicated, but the instruction execution speed is faster, and the power consumption is smaller under the same main frequency.

The cortex-m0/m0+ and cortex-m3 cores are widely used in the energy meter industry. The cortex-m0/m0+ is a John Von Neumann structure, in which cortex-m0 lacks a memory protection unit (MPU). The existing energy meters do not require high processing performance and do not require high frequency, so cortex-m0/m0+ is widely used. The cortex-m3 is a Harvard structure, and it is widely used in the high-end meter market in Western Europe that requires high frequency and processing performance. The difference between cortex-m4 and cortex-m3 is the addition of a floating-point FPU unit to enhance DSP performance. The demand for DSP and FPU will become more and more urgent in response to the demand for power grid quality, harmonic measurement and measurement, and future location analog sampling and digital operations. So choose the cortex-m4 solution.

3.3. **SOC chip selection**

ST firm’s cortex-m series chips have been widely used in the field of energy meters, and its cortex-m4 chips have two series: STM32F4 and STM32L4. STM32F4 has a higher main frequency, and the main frequency of STM32L4 is slightly lower than that of F4, but it has extremely low power consumption, which can meet the application scenarios of disposable battery-powered full life cycle.

At this stage, the use of the energy meter for MCU’s RAM has reached about 16K, and the use of ROM has reached about 384K. In view of the OS’s multi-task scheduling and parallel processing mechanism, the demand for RAM is proportional to the number of tasks, which is basically an order of magnitude higher than the demand for non-OS systems. In view of the overhead of OS kernel code and interface code in modular design, and the addition of new application functions, the use of ROM will be 2 to 3 times higher than that of traditional MCU.

Through the analysis of the instruction set system, MCU core, and SOC chip, ST firm’s STM32L496VT6 was finally selected as the preselected chip. It belongs to a series of chips. When the RAM, ROM or peripheral resources are not satisfied during the development process, there is a replacement chip for PIN-TO-PIN. There is no need to modify the hardware, and the embedded software can be made minor modifications or recompiled without modification.

3.4. **Associative design**

The associated design mainly includes two points: main frequency selection and mass storage design.

The selected SOC supports up to 80mHz. If the main frequency is lower than 40mHz, USB applications cannot be expanded. At the software level, using SPI FLASH as a mass storage, the bottleneck lies in the rate of the SPI port. The higher the main frequency, the higher the SPI port rate. However, when the main frequency is greater than 40mHz, the difficulty of EMI design is greatly increased, and the added protection devices to suppress EMI will also increase, and the traditional double-layer ordinary PCB manufacturing process of the energy meter cannot meet the requirements. On the other hand, when the main frequency is higher than 40mHz, the operating power consumption of the chip will increase, which not only wastes a lot of energy, but also requires the discharge capacity of the disposable battery when awakened from a power failure. When the operating power consumption exceeds 10mA after waking up, the battery voltage is pulled down, causing constant reset and causing the battery to run out of power. Once again, when the main frequency is higher than 40mHz, the SOC internal FLASH will need to insert more waiting time when reading instructions. If the main frequency
is increased from 40mHz, the performance will not increase linearly as if the main frequency is increased below 40mHz. In summary, it was decided to design the main frequency at 40mHz.

For SPI FLASH, there are three types: standard SPI, Dual SPI, and Qual SPI. Standard SPI is usually called SPI, which is a serial peripheral interface specification with 4 pin signals: clk, cs, mosi, miso; Dual SPI is only for SPI FLASH, not for all SPI peripherals. For SPI FLASH, full-duplex is not commonly used, so the usage of mosi and miso is expanded to make them work in half-duplex to double data transmission. You can send a command byte into dual mode to make mosi become SIO0 (serial io 0), mosi becomes SIO1 (serial io 1), so that 2 bits of data can be transmitted in one clock cycle, which doubles the data transmission; Qual SPI is similar to Dual SPI, but also for SPI FLASH. However, two I/O lines (SIO2, SIO3) have been added for the purpose of transmitting 4 bits within one clock. These three types correspond to 3-wire, 4-wire, and 6-wire respectively. Under the same clock, the more wires, the higher the transmission rate. In addition, the current mainstream SPI FLASH supports Qual SPI interface, so the Qual SPI technology is used to increase the speed of external FLASH.

4. Embedded operating system requirements and selection

4.1. Embedded operating system requirements
According to the functional requirements of the management core, the following requirements are put forward for the operating system:

4.1.1. Instantaneity. For smart energy meters, key data such as electrical energy, event records, demand, etc., need to be quickly stored in non-volatile memory after they are generated, in case of power loss, and rapid response to communications, to ensure the efficiency and success rate of data reading.

4.1.2. Reliability. Smart energy meters cannot be powered down all year round, and recording data is very important. The application functions are becoming more and more complex, and the probability of bugs is increasing. It requires the software to run stably and reliably for a long time without fatal data errors, loss or frequent restarts. When the application software is running, bugs can be quickly detected, isolated and restored.

4.1.3. Low power consumption. When the smart energy meter is energized for a long time, the smart energy meter with high power consumption will waste more electric energy during the whole life cycle; the internal space of the smart energy meter is limited, the outer casing is sealed, the heat dissipation is poor, and the battery power supply is required in the event of a power failure. All of them require the operating system to support low power consumption.

4.1.4. Scalability. The functions of the energy meter are becoming more and more complex, and the functional modules should be able to be expanded flexibly and easily upgraded. On the other hand, there are more and more underlying technologies that functions rely on, and it is too difficult to re-implement them. There should be a good community ecology, shared mature technologies can be used, and they can be used after quick and simple adaptation.

4.2. Introduction to Huawei LiteOS
LiteOS is currently the world’s lightest IoT operating system [3]. Its system is as light as 10KB, with zero-configuration, self-organizing, and cross-platform capabilities. It is mainly used in smart hardware in the IoT field such as smart home, wearable, Internet of Vehicles, smart meter reading, and industrial Internet, etc. Data collection and real-time control are its typical use environments. The LiteOS operating system has the characteristics of the lowest energy consumption, smallest size, and fastest response. It has launched a fully open source community, providing chips, modules and open source hardware boards, such as HiSilicon’s PLC chip HCT3911, media chip 3798M/C, IPCamera chip Hi3516A, and LTE-M chips, etc. (third-party chips can also be selected, such as STM32, etc.).
Huawei’s IoT operating system Huawei LiteOS is a lightweight operating system based on a real-time kernel developed by Huawei for the IoT field, and is one of the typical lightweight embedded operating systems at home and abroad. This article studies the lightweight microkernel real-time operating system used in the smart energy meter, which belongs to the basic kernel source code application of Huawei’s IoT operating system Huawei LiteOS, including the existing code supports common data structures such as task scheduling, memory management, interrupt mechanism, queue management, event management, IPC mechanism, time management, soft timer, and doubly linked list, etc.

4.3. Lightweight OS function design

Whether the embedded OS can be sustainable depends on whether the ecology is mature, and the ecology is mainly related to the length of time the OS has appeared and whether the OS is open source licensed. There is not much difference in the reliability of the kernel. The more open the ecosystem, the better and the more sustainable the OS. The interpretation of the main restrictions of the open source community’s license is shown in the figure 1 below.

![Figure 1. Interpretation of the main restrictions of the license.](image)

Recognizing the importance of ecology, the OS license is very loose. Huawei uses the BSD-3 license, and the code of Huawei LiteOS will use the BSD 3-Clause License. After the system modifies the code, the source can be closed.

The lightweight OS mainly includes the operating system kernel, operating system components, and software packages in the kernel architecture. Among them, the operating system kernel is a very mature technology, so there is little difference in the function of each operating system. They all include a multi-task scheduling mechanism based on priority preemption and rotation of the same priority time slice; including semaphores, mutually exclusive models, message queue, message mailbox, software timing,
interrupt management and other functions; the number of software packages is uneven, and the software packages mainly depend on the OS community and ecology. Most of the original software packages are common to each ecology, they are all source code obtained from the same open source community, and then adapted to the OS.

However, the development of various operating system components is later than the kernel, especially the emergence of IoT applications. There are big differences in kernel and application separation, application programming interface, kernel protection, application isolation, low power consumption, and application modular loading. And these component functions are just the proprietary functions necessary for a good smart energy meter operating system. Each system needs more or less component function increase or improvement. The specific technical improvements mainly include:

4.3.1. **Kernel and application separation, application programming interface.** The technology of SVC system call falling into the kernel space is adopted to realize the separation of OS and application code programming mirror. The application is programmed based on the C library and system calls (including custom system calls) interfaces. Using a virtual file system, the large-capacity FLASH storage, device drivers and functions (communication between modules) are abstracted as files, and application programming is performed according to POSIX standards.

4.3.2. **Kernel protection, application isolation.** The MPU protection technology is adopted to realize the protection of core data and task data. When the MCU is in the kernel space, the maximum privilege is enabled. When the MCU is in the user space, first close all permissions, then read the MPU whitelist entries of the task control block, load them one by one, and reload the previous entries with subsequent entries. Including: all spaces are read-only, task heap and stack are readable and writable, and task global variables are readable and writable. Each task has its own whitelist entry to achieve isolation between tasks.

4.3.3. **Application modular loading.** The method of specifying the link address and standardized module description header is adopted to realize application modularization. The RAM, ROM, EEPROM, FLASH and other resources used by the module are managed uniformly, and self-description in the module header. The module interface is described by a standardized API, which is called by the system to communicate between modules through the kernel. The kernel performs resource allocation by scanning each 4K header, and calls the module entry function to create tasks. Set up 3 interface functions, “get, set” to read and write data, “act” to operate the method, and uniformly code the interface function read and write and operation object identification, and the kernel realizes the distribution between modules.

4.3.4. **Low power consumption technology.** Use tickless technology to achieve low power consumption, calculate the shortest blocking time of all tasks when the system is in an idle state, and modify the tick to put the MCU in sleep mode, automatically wake up the MCU when the task is ready, and recalculate the tick. Sleep can be “<10uA” when powered by battery.

5. **Conclusions**

This article mainly completes the selection and associated design of a new type of smart energy meter hardware management core MCU that integrates time-sharing metering and billing to meet the needs of embedded operating systems and the addition or improvement of lightweight operating system functions. In terms of hardware, the simplified instruction set RISC is selected as the main chip of the management core, and the Thumb instruction set of ARM is adopted, combined with the cortex-m4 solution, to meet the needs of DSP and FPU for analog sampling and digital operations. In order to meet the life cycle of one-time battery power supply, ST firm’s cortex-m4 chip STM32L4 series STM32L496VGT6 is selected as the preselected chip, the main frequency is designed at 40mHz, and the Qual SPI port design is used to increase the rate of external FLASH. In terms of software, in response to functional
requirements, the management core changed to a lightweight operating system as the software framework, Huawei LiteOS was selected and the system component function design improvement requirements were proposed. The research results can support the realization of the basic functions and expandable functions of the smart energy meter, and lay a foundation for the technological development of the energy meter field, and have important construction significance.

Acknowledgment
This work is supported by the State Grid Corporation of China (5210EF18000G).

References
[1] Cheng Chunrui, Liu Wanjun. Construction of high cohesion and low coupling software architecture[J]. Computer Systems & Applications, 2009, (7):19-22.
[2] Peng Yeqiang. Design and implementation of QSPI-Flash controller for secure storage of Internet of Things[D]. Huazhong University of Science and Technology, 2019.
[3] Li Zeqiu. Research and Design of Embedded Operating System Microkernel Architecture[D]. Guilin University of Technology, 2011.