Research of a Data Protection Method based on Chip Emulator

Jun Liu[1, a], Liang Liu[1, b], Ting Chong[1, c], Shengxia Dou[2, d], Xige Zhang[1, e]

[1]State Grid Key Laboratory of Power Industrial Chip Design and Analysis Technology, Beijing Smart-Chip Microelectronics Technology Co., Ltd. Beijing, China

[2]State Grid Ningxia Electric Power Technical Research Institute. Ningxia, China

[a]liujun5@sgitg.sgcc.com.cn, [b]liuliang2@sgitg.sgcc.com.cn, [c]chongting@sgitg.sgcc.com.cn, [d]doushengxia@dky.nx.sgcc.com.cn, [e]zhangxige@sgitg.sgcc.com.cn

Keywords: emulator, data protection, access right configuration, data download, debug.

Abstract: The chip emulator is implemented by emulating the chip function and can be used for debugging of embedded software programs. On real chips, especially those used in the security field, for security reasons, in order to prevent illegal reading or tampering, it need to protect programs, configuration information and other data which are sensitive. Similarly, when using the chip emulator for program debugging, in particular, the program data and related configuration information that is provided to the user for security function calculation still needs to consider its security and security protection measures. This paper provides a data protection measure, which mainly includes data access right configuration function, data secure download function. Through the configuration of reading, writing and debugging right to protected data, and encrypt download of data, the program can be safely debugged to achieve data protection.

1. Introduction

With the rapid development of the Internet of Things, as one of the core devices connecting all kinds of things, the chip used in security field such as right control and identity recognition are diversified and complicated, resulting in higher scale and complexity of chip-based embedded programs. How to develop high-quality, high-security embedded programs in a short period of time becomes a problem that needs to be focused on [1].

As an embedded program debugging tool, the chip emulator generally uses FPGA to emulate chip hardware logic [2], and provide data download and debug interface. By downloading the embedded program to the emulator and using the debugging function to track the execution of the program, the developer can accurately and efficiently locate the problem in the program. To a certain extent, the debugging operation directly affects the quality and development cycle of the program [3].

In order to reduce the difficulty of program development caused by the scale and complexity of embedded programs, embedded program development generally adopts a hierarchical design and collaborative development mode. The hierarchical design is generally divided into driver layer, system layer, and application layer from the bottom to up according to the function of the program. Collaborative development is generally carried out by different vendors or organizations responsible for different layer of software development, and low-level programs provide functional interfaces for upper-level program calls. For the unified description, the program that provides the function interface is referred to as the API program, and the program that uses the API is called the user program. This development mode does not require the program developer to be familiar with all layer of the development details, only need to pay attention to the program development at a layer, which greatly reduces the technical investment of the manufacturer for embedded program development and accelerates the product development cycle.

However, there are some problems when debugging with the chip emulator. When debugging the user program, the API program needs to be downloaded to the emulator, and the API program provider does not wish to provide source code for intellectual property protection. If provided as a

Hex file [4], there is still the risk of viewing assembly code and disassembly. At present, there is a method to provide a corresponding debug version of the Hex file, the interface of the debug version program is the same as the real version, but the key operations such as algorithm and authentication are modified. Although the user program can complete the normal call of the interface, but the execution result of the program debugging does not match the real expectation, the practical of the debugging is greatly affected, and there is a risk that the program problem cannot be discovered or misreported in time. There is also a common method, in the real chip production, the API program is pre-masked into the chip, the user program is downloaded to the chip through a specific download channel, and then the program is tested by directly operating the chip interface. Although it is a easy way to find problems in program, but it is not easy to locate the problem, it is generally used in the stage where the program's function is relatively stable, and it is not applicable to the early stage of program development.

In order to balance the efficiency of debugging and the protection of intellectual property rights, this paper proposes a data protection method based on the data access right configuration mechanism and data download protection mechanism is provided by the chip emulator to realize the secure downloading and debugging of the API program in the emulator. This solution can better protect sensitive programs and can be flexibly configured by providing a security solution for software collaborative development and debugging.

The first part of this paper introduces the architecture of the chip emulator system with data protection function. The second part explains the implementation of data access right configuration. The third part mainly describes the data download protection process and check mechanism. The fourth part summarizes the full text.

2. Chip Emulator System Architecture with Data Protection

The general chip emulator consists of two parts: the emulator hardware and the emulator software. The emulator hardware mainly includes the chip emulation logic, the debug protocol conversion module and other interfaces [5]. The emulator software mainly includes the embedded software development environment (referred to as IDE for short), and debugging driver interface (referred to as DLL for short).

In addition to the general function of emulator, the emulator hardware includes a data access right control module, the emulator software includes a data encryption module and a data access right configuration module. The system architecture diagram is as follows:

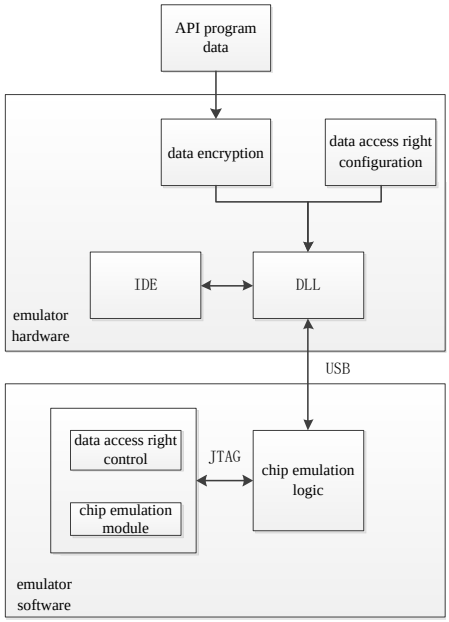

Fig 1. Chip Emulator system architecture diagram

The API program data mainly refers to the data used by the user program for debugging, and exists in the form of a Hex file. The API program data is processed by the data encryption module and passed to the DLL in cipher text. After the API program data is downloaded to the emulator hardware, the access right is configured through the data access right configuration module, and the generated configuration information is transmitted to the DLL in the form of cipher text.

When using the IDE for user program debugging, the IDE first calls the DLL to parse the access right configuration information, cipher text data and determines the legality, then sends them to the debug protocol conversion module through the USB interface [6]. The debug protocol conversion module writes the security data and the information of configuring the access right to the chip emulation module through the JTAG channel [7].

After completing of writing the API program data and the information of configuring the data access right, download the user program to the chip emulation module to start the normal debugging process.

For security reasons, the implementation mechanisms of data encryption and data access right configuration are strictly confidential to user program developers. The API program developer generates the encrypted data and data access right configuration information to the user program developer.

3. Data Access Right Configuration Implementation Mechanism

The configuration tool mainly generates the content of the data access right configuration in the form of a configuration file. The configuration file is a cipher text file, which is parsed by the DLL to obtain configuration information. The configuration file may include one or more configuration records, and each configuration record contains an access right configuration information corresponding to a range of data. If multiple ranges of program data need to be protected, multiple configuration records may be generated. The plain text of the configuration file is as follows:

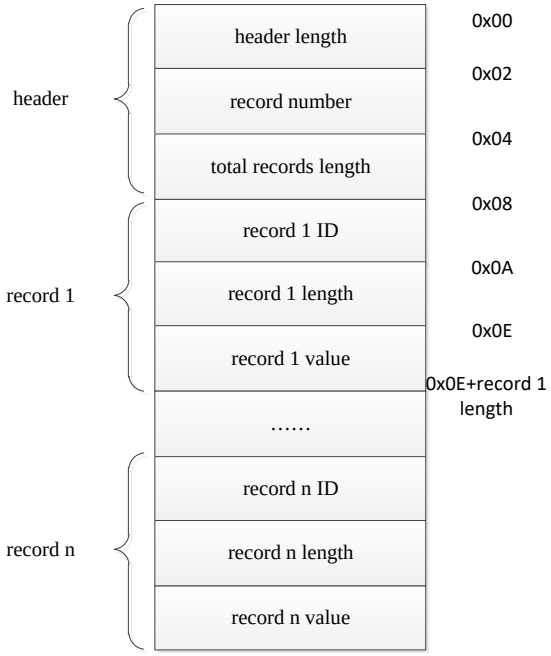

Fig 2. Plain configuration file structure

The format of each configuration record is as follows:

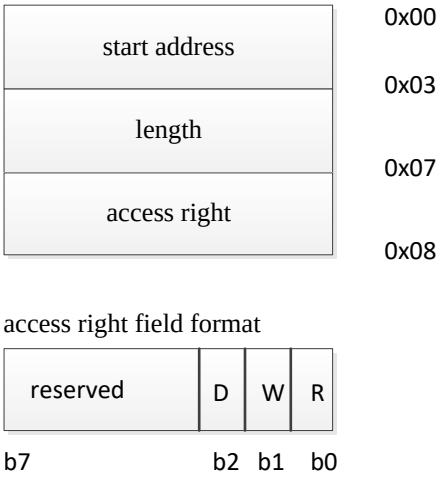

Fig 3. Configuration record structure

Each configuration record consists of 9 bytes. Each of the start address and length field occupies 4 bytes, and the access right field consists of 1 byte. The lower 3 bits of the access right field indicate reading right, writing right, and debugging right. The upper 5 bits are reserved, the corresponding bit which sets to 1 indicates that the corresponding access right is supported, which sets to 0 indicates that the corresponding access right is not supported.

The configuration file is used to generate access right configuration information. The configuration information can be flexibly modified and multiple records can be configured at the same time to meet different configuration requirements. The configuration file is stored in cipher text, which hides the configuration details and prevents illegal tampering. It becomes an important part of the data protection function.

4. Data Download Protection Implementation Mechanism

As the security data, the API program data needs to be protected before being downloaded to the emulator hardware. This paper uses the encryption of the API program data and provides it to the user in the form of cipher text.

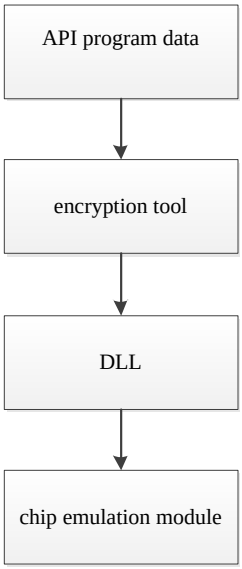

Fig 4. Data download process

The encryption tool mainly encrypts the plaintext API program data, then generates ciphertext data, and is parsed and downloaded to the chip emulation module in the emulator hardware by DLL.

If there are multiple ranges of API program data to be protected, multiple encrypted program data can be generated, and the DLL performs parsing and then performs the download operation.

When the DLL parses the encrypted data, it needs to judge the legality. The following describes the legality judgment process of an API program:

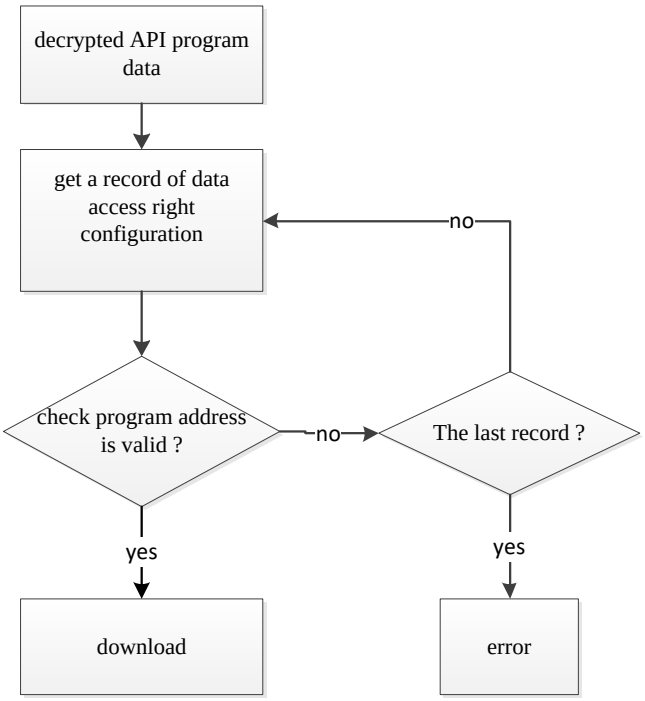

Fig 5. Data legality judgment

First, reading the decrypted API program data, the start and end address of the program data is obtained, and then a right configuration record is read to determine whether the address range of the API program data is within the address range of the configuration record, and if so, download the data; otherwise, determine whether the current right configuration record is the last record. If yes, report an error. Otherwise, read the next right configuration record and continue the judgment process. For the existence of multiple ranges of API program data to be downloaded, the data legality judgment process is performed separately for each range of API program data.

5. Summary

This paper provides a method for protecting data on a chip emulator, which is simple and easy to implement. By protecting the data storage, downloading and debugging of the API program data, the intellectual property of the API program data can be protected, and the debugging is more practical. It is very helpful to improve the efficiency of user program debugging using the chip emulator.

Acknowledgments

This research was financially supported by the science and technology project of State Grid Corporation of China. The project name is, Research on Key Technology of Low Power Consumption of Embedded CPU Core. The project No. is 546816190004.

References

[1] Wei Ma. “Discussion on the Application of Embedded IP Protocol in Internet of Things”. 2012 International Conference on New Energy, Biological Engineering and Food Security (NEBEFS 2012). Hong Kong, China, 2012, pp.81-85.

[2] Sergey Podryadchikov, Vadim Putrolaynen, Maksim Belyaev, Mikhail Chuvstvin, Igor Tabachnik. “FPGA-based testing system of NAND-memory multi-chip modules”. Microelectronics Journal, 2019, pp.73-76.

[3] Michael Spieker, Arne Noyer, Padma Iyenghar, Gert Bikker, Juergen Wuebbelmann, et al. “Model based debugging and testing of embedded systems without affecting the runtime behaviour”. Proceedings of 2012 IEEE 17th International Conference on Emerging Technologies & Factory Automation. Krakow, Poland, 2012.

[4] Wikipedia. Intel HEX [EB/OL]. (2019-03-02). https://en.wikipedia.org/wiki/Intel_HEX.

[5] Ken Bryan F. Fabay, John Cris F. Jardin, Kervin John C. Jocson, Bernard Raymond D. Pelayo, Anastacia B. Alvarez, et al. “A test port for interfacing and debugging ARM9 processors implemented in FPGA”. TENCON 2012 IEEE Region 10 Conference. Cebu, Philippines, 2012.

[6] Xuhui Chen, Dengyi Zhang, Hongyun Yang. “Design and Implementation of a Single-Chip ARM-Based USB Interface JTAG Emulator”. 2008 Fifth IEEE International Symposium on Embedded Computing, Beijing, China, 2008.

[7] IEEE.1149.1-2013-IEEE Standard for Test Access Port and Boundary-Scan Architecture [EB/OL]. (2013-05-13). https://standards.ieee.org/standard/1149_1-2013.html.