Academic Journal of Science and Technology
ISSN: 2771-3032 | Vol. 13, No. 2, 2024

A Simple Solution for Online Upgrade of User Programs Based on SPI Bus Protocol

Ji Liu[1, a], Shaohui Yan[1, *]

College of Physics and Electronic Engineering, Northwest Normal University, Lanzhou 730070, China
[a]liuji1206@163.com, [*]Corresponding author: mortals_ysh@163.com

Abstract: Traditional FPGA program upgrades typically require JTAG in conjunction with specialized EDA tools for downloading and updating programs. However, in practical applications, FPGAs are usually packaged in devices and cannot be disassembled, which brings many inconveniences to online program upgrades. To this end, this article introduces a simple solution for online upgrade of user programs based on SPI bus protocol. The upper layer user transfers the BIN file of the project to FPGA via UART protocol to start configuring SPI Flash, and completes the firmware upgrade method through MultiBoot dual mirroring technology. This method has low maintenance costs, stable updates, and can also remotely update system programs.

Keywords: SPI; UART; FPGA chip; Flash chip; MultiBoot dual mirror.

1. Introduction

In some practical applications, it is necessary to perform online maintenance and upgrade of FPGA programs in SPI Flash. Currently, the conventional online upgrade method is to disassemble the packaged device and update the FPGA program using a downloader through the JTAG port. This method requires professional personnel to use professional EDA tools, which is time-consuming, labor-intensive, and increases maintenance costs[1].To solve this problem, a simple online upgrade scheme for user programs based on SPI bus protocol was proposed. RTL code was written using Verilog HDL and successfully applied through Modelsim testing, simulation, and on-board verification, reducing the burden on staff and making FPGA maintenance simpler and more convenient.

2. System Scheme Design

2.1. Systems Design

This article takes Xilinx's Spartan-6 series xc6slx9-2tqg144 chip as an example to upgrade MultiBoot images online using ICAP (The Internal Configuration Access Port).Xc6slx9-2tqg144 is manufactured using 45nm low-power copper processing technology, which is simple and easy to use, achieving the maximum balance between power consumption cost and performance, and has a wide range of applications[2].The configuration chip used for the Flash chip is STMicroelectronics' M25p16, with a maximum clock frequency of 50MHz and a storage space of 16Mbit. Each address stores 1 byte of data, with a total storage depth of 2M. The address is divided into 32 sectors, each sector contains 256 pages, and each page contains 256 bytes.

This system consists of a serial port, FPGA, and SPI Flash. The system diagram is shown in Figure 1. FPGA serves as the core of the board, and the PC interacts with FPGA through a serial port[6][7].The upper computer software sends data with frame headers through the transmission assistant, and the Flash controller receives and parses the data to make decisions on Flash memory read, write, and erase operations.FPGA startup, if the program in region 2 is not upgraded within 30 seconds, ICAP will jump to region 2 to execute the original program. On the contrary, the FPGA receives the BIN file of the transmission project and configures the Flash to complete the firmware upgrade operation[7]. The ICAP online upgrade principle of Spartan-6 series can also be extended to A7 and K7 series.

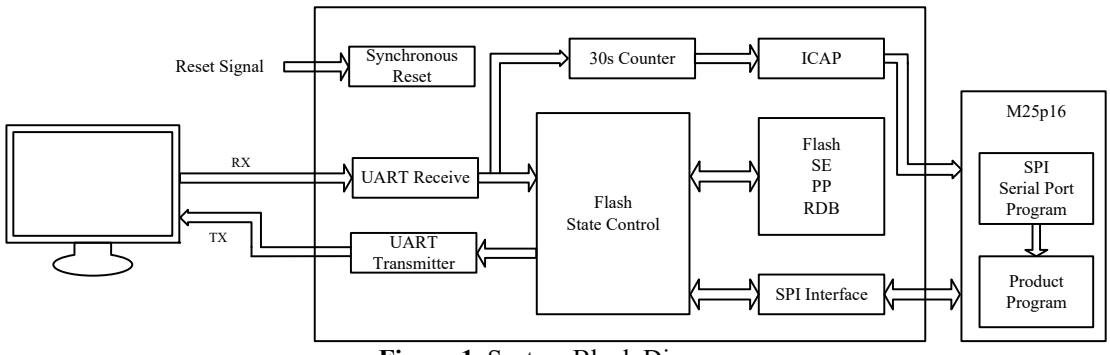

Figure 1. System Block Diagram

2.2. MultiBoot Image Design

MultiBoot is divided into two parts: Golden Image and MultiBoot Image[3][4]. Divide the M25p16 with 16Mbit of memory into two regions: Region 1 is a G image with a starting address of 0x0000000, mainly used for upgrading Flash; Region 2 is the M image with a starting address of 0x01000000. It is the product program of FPGA and can be

updated and upgraded at any time according to the needs of the M image. The division of M25p16 is shown in Figure 2.

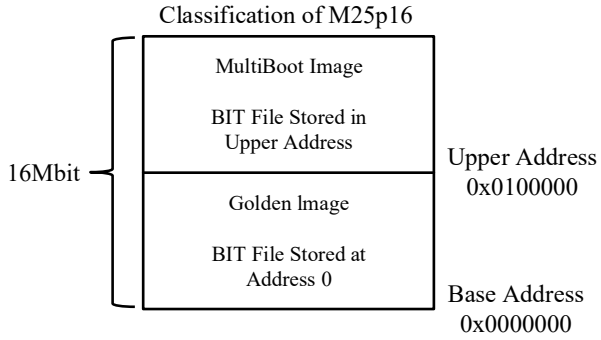

Figure 2. System Block Diagram

The development board automatically loads the G image when powered on, and the FPGA program is configured to start timing. If no erase command is detected from the serial port within 30 seconds, ICAP will be started to jump to the original M image. Otherwise, the M image will be updated through the serial port. If upgrading again, the board must be restarted and powered on within 30 seconds to perform the corresponding operation. The simplified flowchart for loading MultiBoot images is shown in Figure 3.

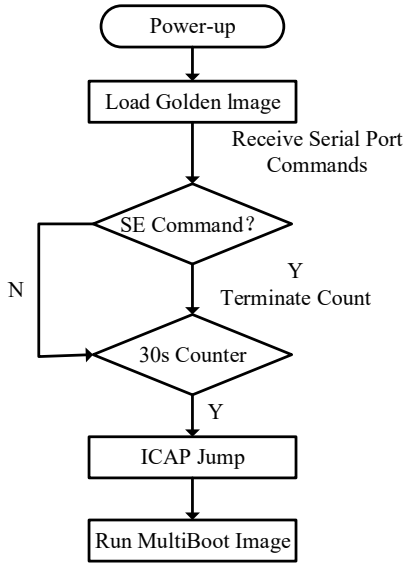

Figure 3. Simplified flowchart for loading MultiBoot images

3. FPGA Modular Design

3.1. Serial Port Sending and Receiving

Universal Asynchronous Receiver/Transmitter (UART) is an asynchronous transmitter that converts parallel data into serial data for transmission during data transmission, and converts received serial data into parallel data during data reception, enabling full duplex transmission and reception[5].This serial port reception and transmission adopts the RS-232 standard design, configured as 8N1. The baud rate used for data reception and transmission is 115200Bd, and the sampling clock clk_flag is generated through counting. The timing diagram for sending a byte is shown in Figure 4. If the Flash status is read, it will be sent to the PC through the serial port.

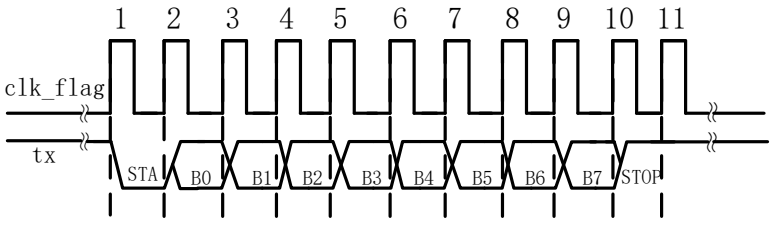

Figure 4. Sending a byte timing diagram

Within 30 seconds of power on, the board can receive PC commands and BIN files through the serial port to achieve firmware upgrade. To ensure that each data bit collected is relatively stable, data is collected at intermediate times. The timing diagram of receiving one byte of data is shown in Figure 5.

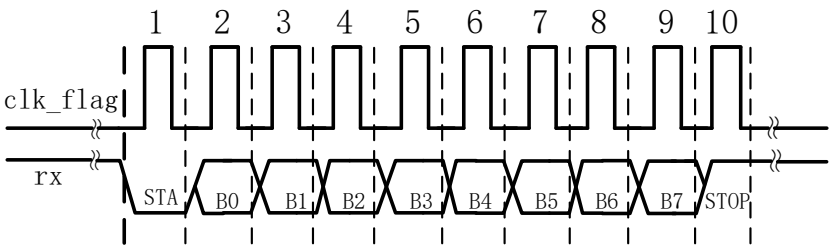

Figure 5. Receiving a byte timing diagram

3.2. SPI Flash Read-Write Erase Controller

Flash sector erase (SE) requires a write command to enable it before setting up, and then sector erase is performed.First, set Cs_n low, and then generate a 12.5MHz clock sck. Flash latches the data at the center position of each bit of sdi based on the rising edge of sck. After the data latch ends, Cs_n is pulled high for at least 100ns, and then Cs_n is pulled low again to prepare for sending the erase instruction SE and 3-byte address. The erase command state machine is shown in Figure 6.

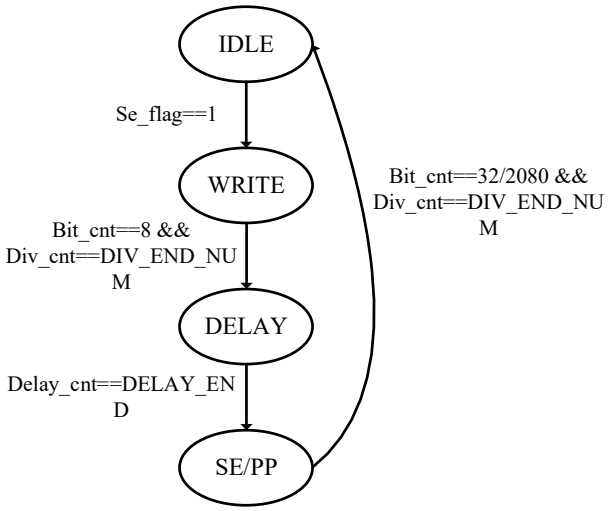

Figure 6. Erasing/Page Program State Machine

Flash page program (PP), like erase operation, requires a write instruction operation before executing the PP instruction. The Flash timing diagram is shown in Figure 7.Given the starting address of the write page, cache one page of Flash data into FIFO. After caching, start the page write operation based on the write Flash flag. For multi page program, additional control cache and write flag implementation are required. In addition, the control and erase control state machines are similar, as shown in Figure 6. Here, the Bit_cnt value is the PP instruction+address+data+Cs-n redundant period, totaling 2081 bit periods.

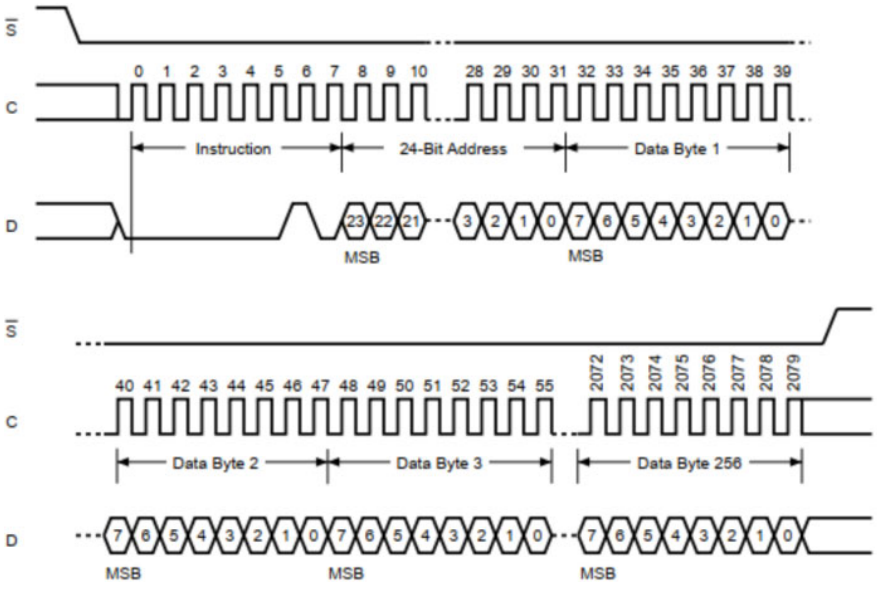

Figure 7. Flash Writing Sequence Diagram

Flash read data byte (RDB), the read instruction can read any byte of data in the Flash address space, without being

limited by sectors or pages, and can continuously read all data. After giving a read instruction and a 24 bit starting address, the data will be output through the serial data line sdo until Cs_n pulls up to interrupt the reading of the data. The timing diagram for reading data is shown in Figure 8.

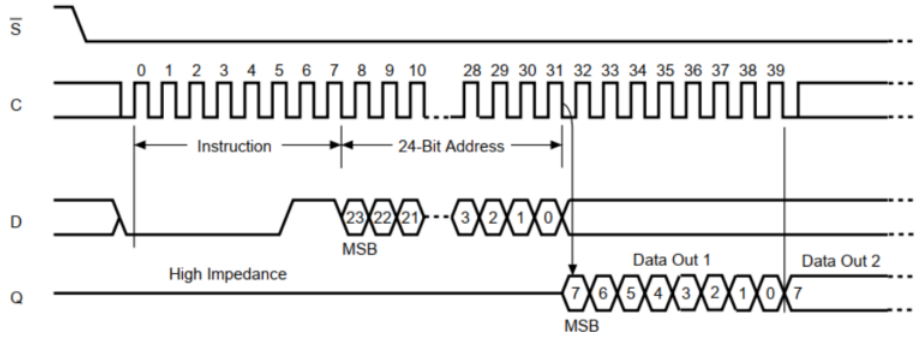

Figure 8. Flash read timing diagram

The upper computer needs to develop three types of data packets for upgrading firmware programs online through serial ports, namely erase command packets, write command packets, and read command packets. Erase command packet, packet header 0xee+address; Page write command packet, header 0xcc+address+write data byte; Read command packet, packet header 0xdd+address+read length, and read data return packet. By controlling the state jump through protocol packets, the Flash state control is shown in Figure 9.

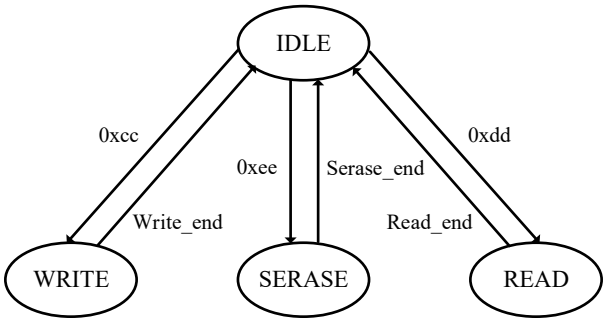

Figure 9. Flash state control

3.3. ICAP Design

By using ICAP to implement the jump stored in two Flash programs, and specifying the corresponding jump address according to the specific operation instructions of ICAP, FPGA can jump to the corresponding storage space address to burn the BIN file for configuring Flash. When reading the BIN file fails or the jump fails, it will return to the base address to continue loading, as shown in Figure 2.To facilitate the control of operational instructions, an ICAP instruction jump state machine was designed as shown in Figure 10. Among them, Pi_flag is the enable signal generated after counting for 30 seconds.

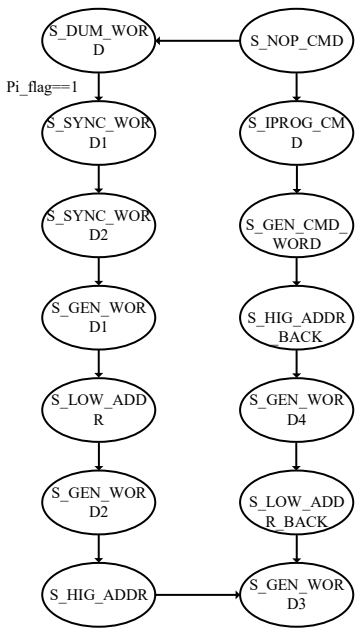

Figure 10. ICAP instruction jump state machine

4. Simulation and Verification of Experimental Results

Implement SPI Flash configuration and firmware upgrade in ISE14.7 programming environment, conduct simulation testing and board validation using ModelSim SE-64 10.5.The clock frequency of the board is 50MHz, and the clock frequency of the SPI clock line sck is set to 12.5MHz. Due to space limitations, this article only analyzes the simulated waveform diagrams of Flash erase, page write, and read, and conducts online upgrade testing of the system using a serial port.

4.1. Flash Erase Simulation and Analysis

The Flash erase simulation waveform is shown in Figure 11.

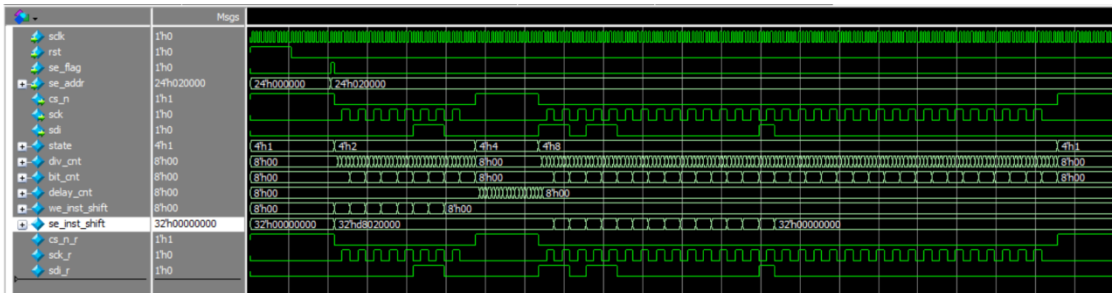

Figure 11. Flash erase simulation waveform

First, send a write enable command, and then perform sector erase. The write enable instruction is 8'h06, and Flash will latch the data of sdi based on the rising edge of sck. The rising edge of sck corresponds to the center position of each data in sdi. To ensure that the write enable is stored by Flash, Cs_n is pulled up and held for 100ns. Afterwards, lower Cs_n again and send the erase instruction 8'hD8 along with a 3-byte address. Finally, to ensure that the sector is erased, raise Cs_n again and hold it high for at least 3 seconds.

4.2. Flash Page Program Simulation and Analysis

The simulation waveform of Flash page program is shown in Figure 12.

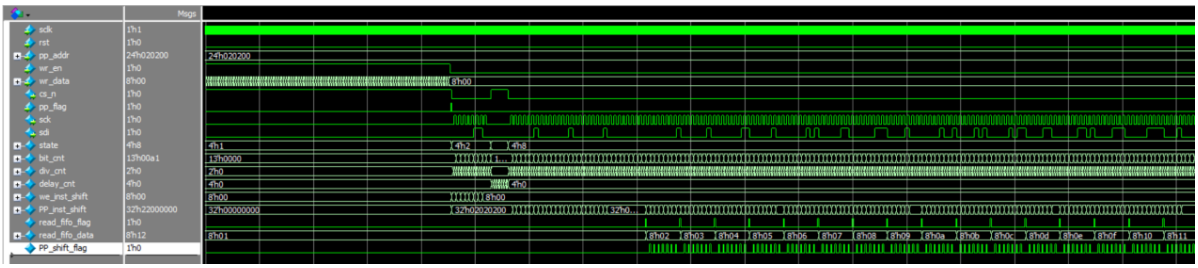

Figure 12. Flash page simulation waveform

Receive BIN file data wr_data sent by the user through UART serial protocol and cache it in FIFO. When the write enable is raised, perform page write function. The page programming instruction is 8'h02. Like the erase instruction, a write enable must be sent before each page write, and the write enable will automatically reset after the page write is completed.When reading data from FIFO, read 8-bit data one beat in advance and assign the highest bit data to the register. Then, the low bit data is sequentially shifted to the highest bit according to the pp_sthift_flag flag and output to the SDI data line.

4.3. Flash Reading Simulation and Analysis

The Flash read simulation waveform is shown in Figure 13.

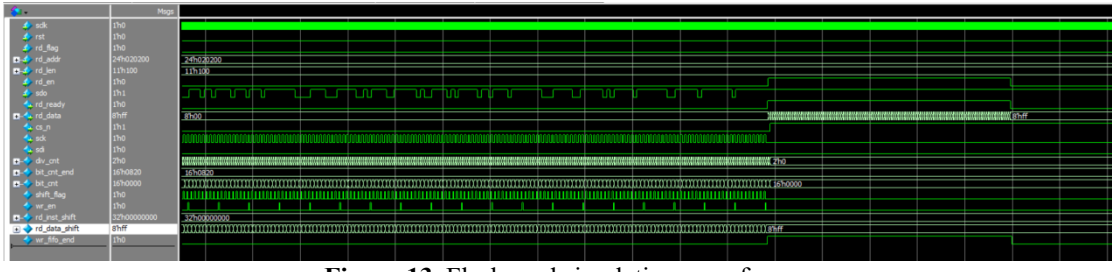

Figure 13. Flash read simulation waveform

Reading Flash timing does not require prior commands, only the design of a frequency divider counter and a bit counter is needed. The final count value of the counter is derived based on the read length, and the data output by SDO is shifted and latched to achieve serial conversion and then written into the FIFO buffer. In the above figure, data is uploaded to FPGA through SDO using Flash, and the stored data can be retrieved from FIFO read enable. It can be uploaded to the user end through UART serial port.

4.4. System Upgrade Testing

The address of region 1 starts from 0x0000000, and the FPGA automatically loads the program in region 1 when powered on. The program in region 1 can read, write, and erase Flash to complete the online upgrade of the system. Area 2 addresses start from 0x01000000 and store the ticker program. To verify the online upgrade function of the system, the program was tested using the serial assistant software installed on the upper computer. The appropriate baud rate was set to transmit the BIN file, image address, and instructions of the breathing light to the board within 30 seconds. After successful transmission, the scrolling light changed to a breathing light and instructions were sent to read the current BIN file. After multiple tests, the program runs normally.

5. Conclusion

This article focuses on practical applications, and through simulation testing of various functional modules of FPGA and on-board verification of the program, the system's online update and maintenance experiments have achieved the expected results. The system uses UART protocol, FPGA chip, SPI Flash chip, and MultiBoot dual mirror technology to complete the online upgrade function of user programs based on SPI bus protocol.This feature allows the system to return the base address to continue loading when reading BIN files fails, ensuring the stability of system firmware upgrades. This method has a simple user interface, high portability, low maintenance cost, and high stability. It is suitable for updating encapsulated FPGAs in engineering, and firmware online upgrades can be completed as long as there is a serial port.

References

[1] SunYonglai.Design and Implementation of Serial Communication Interface Based on FPGA Control [D]. University of Chinese Academy of Sciences (College of Engineering Management and Information Technology), 2014.

[2] Guan Shanshan, Zhou Jiemin.Design and Verification of SPI Flash Controller Based on Xilinx FPGA[J].Electronic device,2012,35(02):216-220.

[3] Wang Feng, LvTianzhi, Yang Mingyang.A method for online updating FPGA programs in SPI Flash based on PCIe bus[J].Electronic production,2021,(09):56-59+65.DOI:10.16589/j.cnki.cn11-3571/tn.2021.09.018.

[4] Chang Liang, Bi Jinchao, Jiang Jiaqi.The method of upgrading FPGA using UART [J]. Journal of Jilin University (Information Science Edition), 2020, 38(03): 286-290. DOI:10.19292/j.cnki.jdxxp.2020.03.008.

[5] Gupta, A. (2019). UART Communication. In: The IoT Hacker's Handbook. Apress, Berkeley, CA. https://doi.org/10.1007/978-1-4842-4300-8_4

[6] HAN Liujun, YAO Yao, HAO Guofeng, ZHAO Can. Implementation of FPGA Multi-Configuration Method Based on CPLD[J]. Electronics & Packaging, 2022, 22(5): 050303 .

[7] P. S. Mutha and Y. M. Vaidya, "FPGA reconfiguration using UART and SPI flash," 2017 International Conference on Trends in Electronics and Informatics (ICEI), Tirunelveli, India, 2017, pp. 59-63, doi: 10.1109/ICOEI.2017.8300765.