How to Program
Code Flash Memory in W7100A

Version 1.2

© 2012 WIZnet Co.,Ltd. All Rights Reserved.
For more information, visit our website at http://www.wiznet.co.kr

Table of Contents

© Copyright 2012 WIZnet Co.,Ltd. All rights reserved.

1 Introduction

The internal memory of W7100A is composed of the Code memory and Data memory. The Code memory is composed of Boot ROM and Code Flash. The address of Boot ROM ranges from 0x0000 to 0x07FF and the Boot ROM has ISP function and other codes necessary for W7100A built in. The address of Code Flash ranges from 0x0000 to 0xFFFF and shares the address range with Boot ROM at a different memory space to save user’s application codes. Please refer to ‘W7100A Datasheet; 2. Memory’ for more details of W7100A’s memory composition.

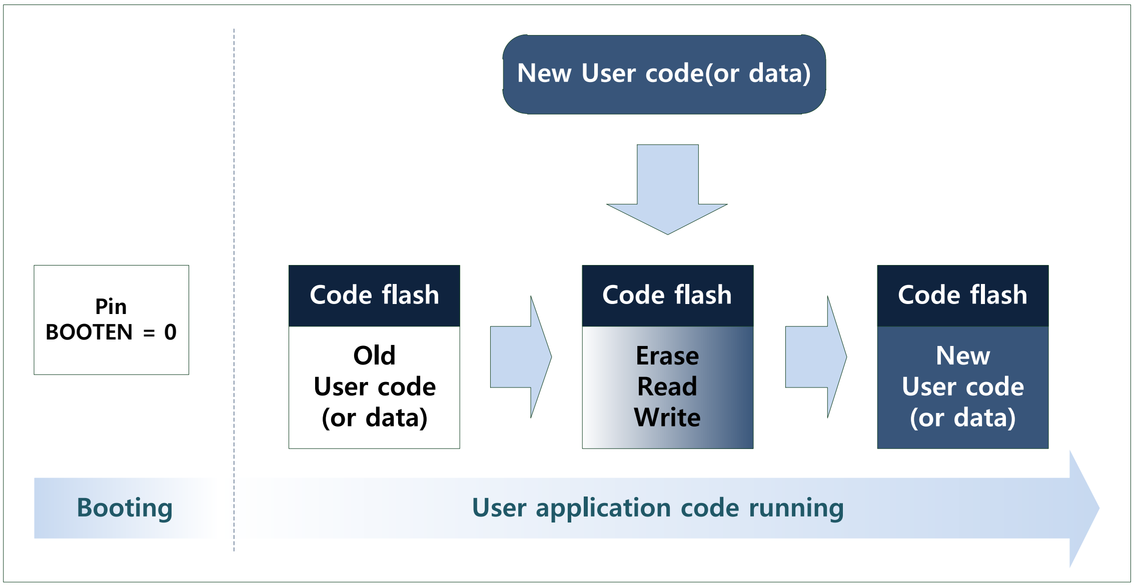


Figure 1.1 Code Flash update process

The modification or update of Code flash 64byte is possible during the operation of the user’s application code as shown in Figure 1.1. This feature allows users to update the firmware while the program is running. This document will illustrate how to read, write, and erase code flash.

The functions for controlling the Code flash are as shown below and will be explained in chapter 2.

Function	Function name	Description
Read	code_read_byte(addr)	Read 1byte by given address
Write	code_write_byte(addr, data)	Write 1byte by given address and data
Sector erase	code_sector_erase(addr)	Erase 256byte block by given address
Chip erase	code_chip_erase( )	Erase whole code flash memory

Table 1.1 Code flash access functions

Note:

Please be aware that the ISP of the WizISP program for writing and erasing W7100A’s memory data is different from the ISP that will be used in this document. Please refer to ‘WizISP and Program Guide’ for more details on WizISP.

© Copyright 2012 WIZnet Co.,Ltd. All rights reserved.

2 Code Flash Programming

2.1 Programming Process

The process for Code flash programming is as shown below.

If the user follows the steps shown below, Byte read, Byte write, or other ISPID commands will operate and the modification of code flash data is possible while the user code is in operation.

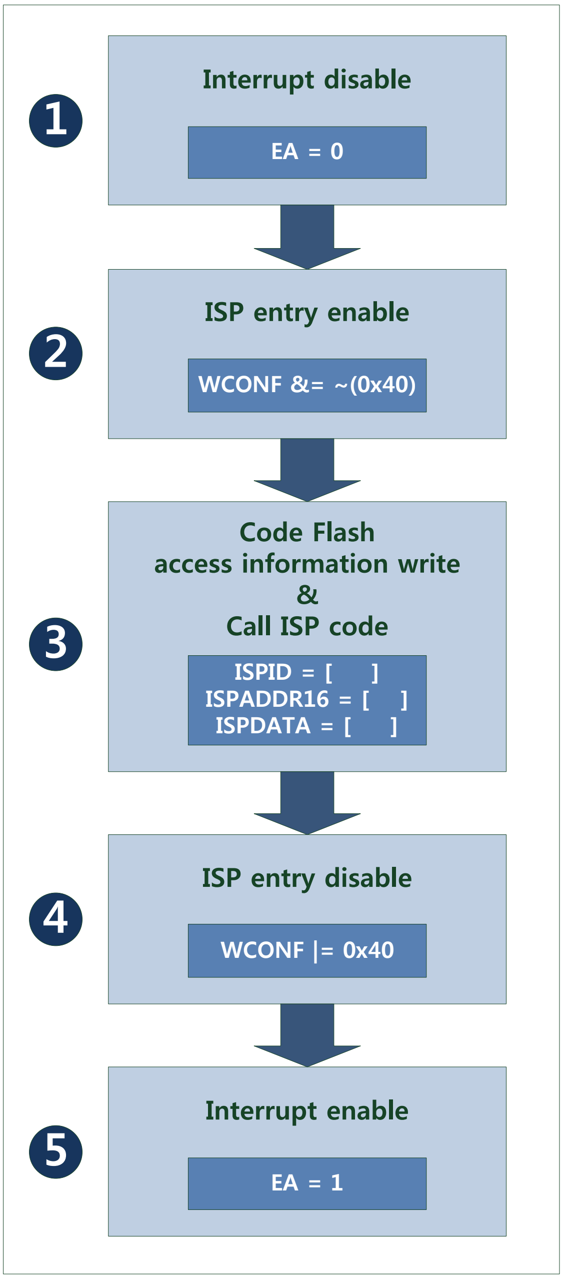


Figure 2.1 Code flash access process

1. When ISP function is in use, there must be no interrupts. Disable all interrupts by setting EA = 0.

2. Set the ISPEN bit of the WCONF register to ‘0’ in order to activate the Boot ROM and ISP ENTRY.

3. Write the ISPADDR and ISPDATA corresponding to the defined ISPID. For example, if the user wishes to write 0xFF to the memory address 0x4000, follow the box below.

```
ISPID = ISP_BPROG; ISPADDR = 0x4000; ISPDATA = 0xFF
```

Next step is calling the ISP code. In order to call the ISP code, user needs to jump to where the ISP code is defined. The ISP code is defined at 0x0003; this can be different depending on the complier. This document is based on the Keil μvision V4.10 compiler.

1. Add #pragma userclass(CODE = ISP) to the file with the main code. Add #pragma userclass(CODE = PRO) to the temporary file and include it to the Keil μvision project.

2. Add CODE_ISP(C:0x0800), CODE_PRO(C:0x0003) to the user classes, which is at the Keil μvision / [Options for target] MENU (Alt+F7) / [LX51 Locate] TAB. The address shows the location of the code.

3. Create a function at the temporary file which added #pragma userclass(CODE = PRO). If the function is called, ISP code call will occur and code flash access will operate accordingly to the values of ISPID, ISPADDR16, and ISPDATA. The ISP ENTRY automatically returns to the location where the ISP function was called when the operation is finished.

4. Set the ISPEN bit of the WCONF register to ‘1.’ Boot ROM and ISP ENTRY will deactivate.
5. Enable all interrupts by setting EA = 1.

Example functions for code flash access are shown in sections 2.2 ~ 2.5.

© Copyright 2012 WIZnet Co.,Ltd. All rights reserved.

2.2 Read

Code flash read is defined as ISP_BREAD(0x00) in ISPID command. The Read command reads 1byte of the designated address and restores it to ISPDATA. Therefore, ISPDATA is not written. The function for Read command is as shown below.

Code Example - Read function

```
unsigned char code_read_byte(unsigned short isp_addr)
{
        unsigned char ret;
        EA = 0;                         // Disable EA
        WCONF &= ~(0x40);               // Enable ISP Entry

        ISPID = ISP_BREAD;
        ISPADDR16 = isp_addr;
        flash_for_isp();                // Call ISP Entry
        ret = ISPDATA;

        WCONF |= 0x40;                  // Disable ISP Entry
        EA =1;                          // Enable EA

        return ret;
}
```

2.3 Write

Code flash write is defined as ISP_BPROG(0xA0) in ISPID command. The Write command writes 1byte of data at the designated address. The function for Write command is as shown below.

Code Example - Write function

```
unsigned char code_write_byte(unsigned short isp_addr, unsigned char isp_data)
{
        EA = 0;                         // Disable EA
        WCONF &= ~(0x40);               // Enable ISP Entry

        ISPID = ISP_BPROG;
        ISPADDR16 = isp_addr;
        ISPDATA = isp_data;
        flash_for_isp();                // Call ISP Entry

        WCONF |= 0x40;                  // Disable ISP Entry
        EA =1;                          // Enable EA

        return ISPDATA;
}
```

2.4 Sector Erase

Code flash sector erase is defined as ISP_SERASE(0x30) in ISPID command. ‘Sector erase’ erases a 256byte size of block starting from the designated memory address. There is a slight difference in Erase command compared to Read or Write commands. There are two steps; first step is prewrite, set every cell in the flash memory to the identical electrical charge level. The second step is designating the address and erasing a 256byte size of block starting from the designated point. User should be aware that ISP entry call should be operated in each step.

Code Example - Sector erase function

```
unsigned char code_sector_erase(unsigned short isp_addr)
{
        EA = 0;                         // Disable EA
        WCONF &= ~(0x40);               // Enable ISP Entry

        ISPID = ISP_ERASE;              // Pre erase
        flash_for_isp();                // Call ISP Entry

        ISPID = ISP_SERASE;             // Sector erase
        ISPADDR16 = isp_addr;           // First delete address (sector)
        ISPDATA = 0;
        flash_for_isp();                // Call ISP Entry

        WCONF |= 0x40;                  // Disable ISP entry
        EA = 1;                         // Enable EA

        return 0;
}
```

© Copyright 2012 WIZnet Co.,Ltd. All rights reserved.

2.5 Chip Erase

Code flash chip erase is defined as ISP_MERASE(0x10) in ISPID command. Once chip erase is operated, the entire code flash is erased. The steps for chip erase are identical to the steps for sector erase. We have to execute the ISP_ERASE command first, then sequentially execute the ISP_MERASE command.

Both commands use ISP entry call and have no input or return value.

Warning

Please be aware that if chip erase is commanded while the user code is operating, all operating data can be erased.

Code Example - Chip erase function

```
void code_chip_erase(void)
{
        EA = 0;                         // Disable EA
        WCONF &= ~(0x40);               // Enable ISP Entry

        ISPID = ISP_ERASE;              // Pre erase
        flash_for_isp();                // Call ISP entry

        ISPID = ISP_MERASE;             // Whole code flash memory data erase
        flash_for_isp();                // Call ISP entry

        WCONF |= 0x40;                  // Disable ISP entry
        EA = 1;                         // Enable EA
}
```

© Copyright 2012 WIZnet Co.,Ltd. All rights reserved.

3 Example Demonstration

This section shows the example codes for Read, Write, and Erase functions. The operation of functions can be checked below and operates in the order shown Figure 3.1 below.

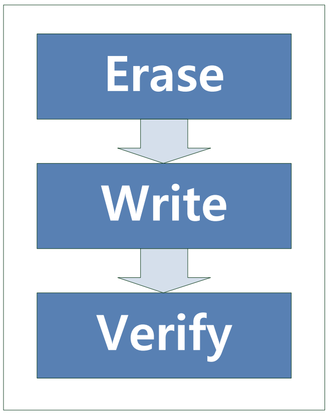


Figure 3.1 Example Demo Simple Flow

1. Erase – Erase 256byte starting from base address.
2. Write – Write 0 ~ FF in 256byte starting from base address.
3. Verify – Operate Read to read the written value and compare with the actual used value.

Note:

The program codes in this document are pseudo codes and are only a part of the entire codes.

The example codes are based on KEIL μvision V4.10 compiler.

Code Example - Read, Write, Erase test application

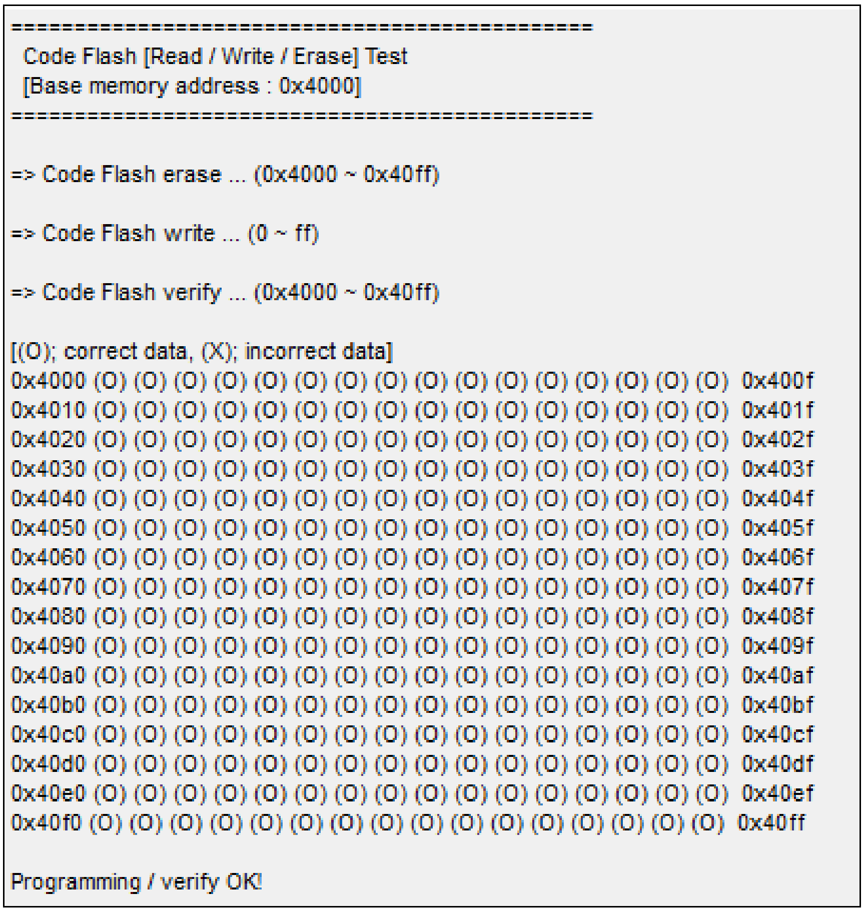
```
unsigned char wd[256]; // Save written data
{
        base_addr = 0x4000; // Set base address

        // Code flash erase
        addr = base_addr;
        code_sector_erase(addr); // erase 256 byte(sector) data

        // Code flash write
        addr = base_addr;
        for(i=0; i<=0xff; i++)
        {
```

```
                        wd[i] = code_write_byte(addr, j); // write 1 byte data
                        addr += 1; j += 1;
            }
            // Code flash verify
            // if written data and read data are same, the code flash access process success.
            addr = base_addr;
            for(i=0; i<=0xff; i++)
            {
                        if (code_read_byte(addr) == wd[i]) // read 1 byte data and compare written data
                                // Compare correct, verify success
                        else
                                // Compare incorrect, verify failed
            addr +=1;
            }
}
```

The result of the example code command is as shown below.



```
================================================
 Code Flash [Read / Write / Erase] Test
 [Base memory address : 0x4000]
================================================

=> Code Flash erase ... (0x4000 ~ 0x40ff)

=> Code Flash write ... (0 ~ ff)

=> Code Flash verify ... (0x4000 ~ 0x40ff)

[(O); correct data, (X); incorrect data]
0x4000 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x400f
0x4010 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x401f
0x4020 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x402f
0x4030 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x403f
0x4040 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x404f
0x4050 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x405f
0x4060 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x406f
0x4070 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x407f
0x4080 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x408f
0x4090 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x409f
0x40a0 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x40af
0x40b0 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x40bf
0x40c0 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x40cf
0x40d0 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x40df
0x40e0 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x40ef
0x40f0 (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O) (O)  0x40ff

Programming / verify OK!
```

Figure 3.2 Application Example results

Document History Information

Version	Date	Descriptions
Ver. 1.0	Nov, 2011	Release with W7100A launching
Ver. 1.1	Jun, 2012	Fixed some awkward expressions of English documents

Copyright Notice

Copyright 2012 WIZnet Co.,Ltd. All Rights Reserved.

Technical Support: support@wiznet.co.kr
Sales & Distribution: sales@wiznet.co.kr

For more information, visit our website at http://www.wiznet.co.kr