MG2470-ISP Programming Guide

(No. ASW0407)

V1.0

REVISION HISTORY

Version	Date	Description
VER.1.0	2011.2.22	▪ First Version Release

CONTENTS

1. INTRODUCTION

MG2470, a ZigBee Single chip of RadioPulse, is operated on 2.4GHz ISM frequency band..

MG2470 has ZigBee RF Transceiver, modem, 8051 Processor, 6KB SRAM and 64KB Flash Memory.

This document explains how to download the program image to FLASH image to FLASH Memory used as code memory of MG2470.

MG2470 can distinguish Normal Mode and Programming Mode(ISP mode) by external pin. For the detailed information of the pin, refer to 'MG2470 Datasheet'.

When MG2470 is operated as ISP mode, it uses UART1 for the connection with the host based on the PC. The following shows the configuration for it.

Port	UART1(P1.0-RX, P1.1-TX)
Baud Rate	115200bps
Data Bit	8-bit
Parity	No Parity
Stop Bits	1-Stop

2. Flow Chart

The following shows the process to download the program image to FLASH Memory used as code memory of MG2470.

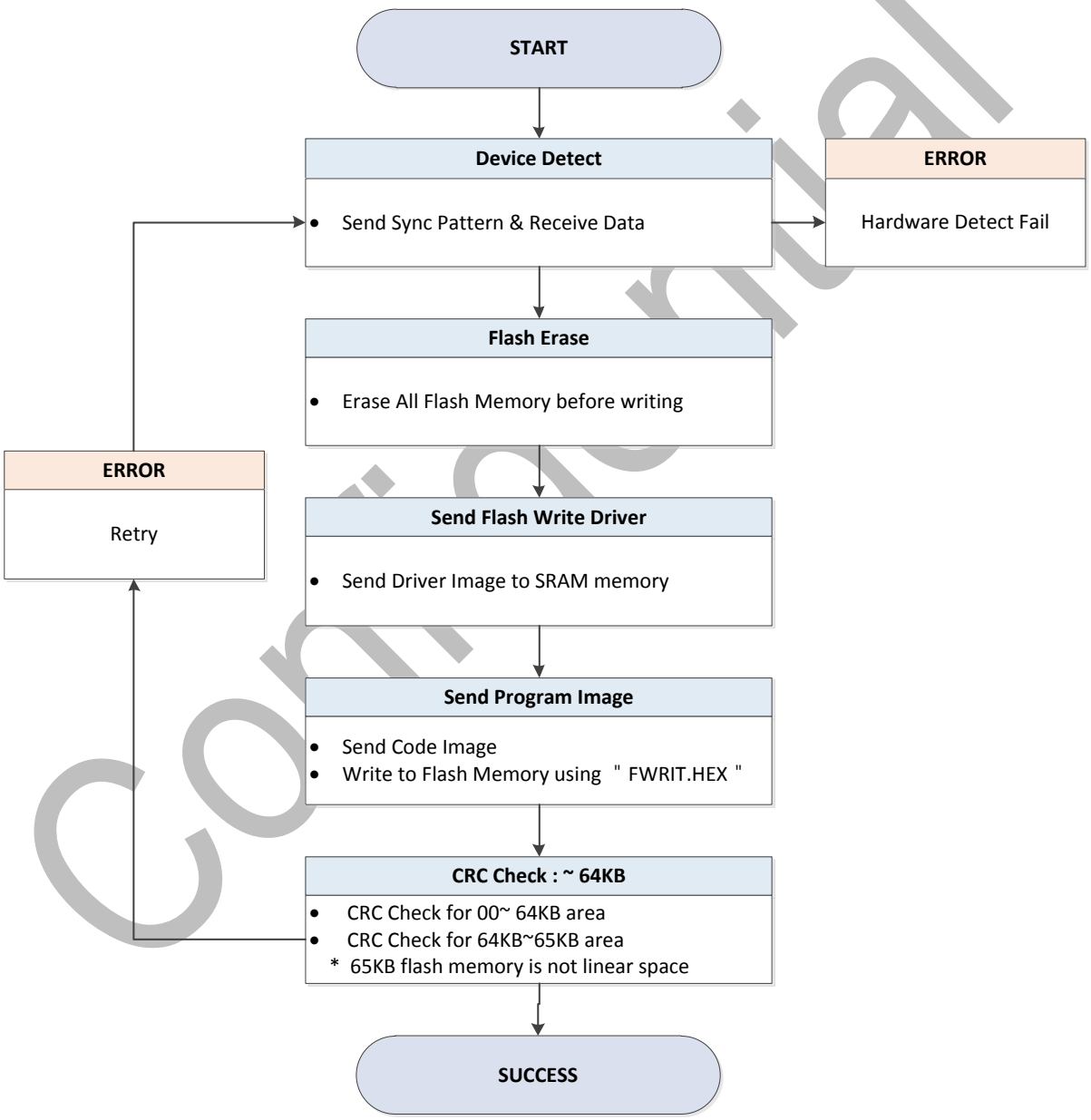


Figure 1. Process of downloading the program image

3. ISP Programming Source File

Followings are Linux and Window source files to develop ISP download program of MG2470. These source files are same to library used for “Device Programmer MD” development. A user can develop MG2470 ISP download program for own environment and purpose.

3.1. FILE Description

	Folder	File	Description	
Linux	Driver	FREAD_B0.HEX	MG2470 driver file for Flash Read	
		FREAD_B1.HEX	MG2470 driver file for Flash Read	
		FREAD_B4.HEX	MG2470 driver file for Flash Read	
		FWRIT.HEX	MG2470 driver file for Flash Write	
		mg2470_isp.c	Main	ISP Run example
		serial.h	serial interface	Correct to suitable serial interface for environment
		serial.c		
		intelhex.h	Intelhex file utility	Supports read, write, and merge functions of Intel hex file.
		intelhex.c		
		mg2470_rom_isp.h	ISP host handler for internal ROM	
		mg2470_rom_isp.c		
		mg2470_ihx_isp.h	ISP host handler for internal ROM	
		mg2470_ihx_isp.c		
		mg2470_hib_def.h	Hardware Information	Hardware Information define
Windows	Driver	FREAD_B0.HEX	MG2470 driver file for Flash Read	
		FREAD_B1.HEX	MG2470 driver file for Flash Read	
		FREAD_B4.HEX	MG2470 driver file for Flash Read	
		FWRIT.HEX	MG2470 driver file for Flash Write	
	SerialPort	SerialPort.h	Windows Serial	Serial communication file for Window.
		SerialPort.cpp		
	isp	mg2470_isp.cpp	Main	ISP Run example
		serial.h	serial interface	Correct to suitable serial interface for

		serial.c		environment.
		intelhex.h	Intelhex file utility	Supports read, write, and merge functions of Intel hex file.
		intelhex.cpp		
		mg2470_rom_isp.h	ISP host handler for internal ROM	
		mg2470_rom_isp.cpp		
		mg2470_ihx_isp.h	ISP host handler for internal ROM	
		mg2470_ihx_isp.cpp		
		mg2470_hib_def.h	Hardware Information	Hardware Information define

3.2. Major Functions

3.2.1. isp_cmd(CMD, ADDR, SIZE)

Run ISP command between ISP host and MG2470 ROM.

Syntax	Void isp_cmd(unsigned char cmd, unsigned addr, unsigned size)
Description	Executes ISP command to MG2470 ROM. Please refer to Sec 5 for detailed command.
Parameter	・ cmd: ISP command ・ addr : 16bit address (2byte) ・ size : 16bit length (2byte)
Return Value	void
File	mg2470_rom_isp.h, mg2470_rom_isp.c or mg2470_rom_isp.cpp

3.2.2. isp_sync()

Check ISP mode starting of MG2470.

Syntax	void isp_sync(void)
Description	Check whether MG2470 is in ISP mode or not. TX: 0x55, 0x55, 0x55, 0x55, 0x55, 0x55 RX: 0x55, 0x55, 0x55, 0x55, 0x55, 0x55, 0xFF
Parameter	
Return Value	void
File	mg2470_rom_isp.h, mg2470_rom_isp.c or mg2470_rom_isp.cpp

3.2.3. ihx_fsh_writ(FILE)

Download firmware code to MG2470 code data area. It means this function updates entire firmware of MG2470.

Syntax	void ihx_fsh_writ(const char *fname)
Description	Updates after Flash Mass Erase.
Parameter	· fname : Intel hex 386 file name to write in flash.
Return Value	void
File	mg2470_ihx_isp.h, mg2470_ihx_isp.c or mg2470_ihx_isp.cpp

3.2.4. ihx_fsh_read(FILE)

Read entire code data area of MG2470.

Syntax	void ihx_fsh_read(const char *fname)
Description	Read current firmware code of MG2470.
Parameter	· fname : Intel hex 386 file name to store read data in flash.
Return Value	void
File	mg2470_ihx_isp.h, mg2470_ihx_isp.c or mg2470_ihx_isp.cpp

3.2.5. ihx_hib_read(FILE, ADDR, SIZE)

Read entire hardware information area of MG2470.

Syntax	void ihx_hib_read(const char *fname, unsigned addr, unsigned size)
Description	Read Hardware Information.
Parameter	· fname : Intel hex 386 file name to store read data in flash. · addr : 0x1000 = 16bit address (2byte) · size : 0x00200= 16bit length (2byte)
Return Value	Void
File	mg2470_ihx_isp.h, mg2470_ihx_isp.c or mg2470_ihx_isp.cpp

3.3. ISP Programming Example

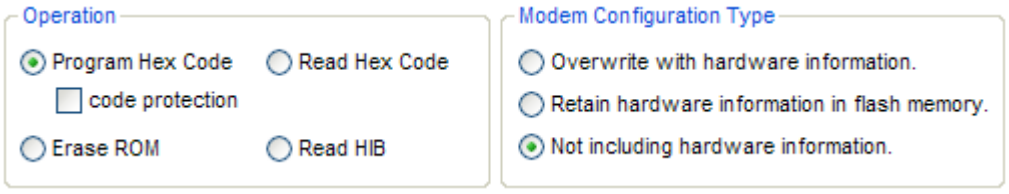
3.3.1. Download Firmware

There are three ways to download MG2470 firmware as follows;

- Not including hardware information
- Overwrite with hardware information
- Retain hardware information in flash memory

3.3.1.1. Not including hardware information

It only downloads firmware hex code. It is same to figure below from Device-Programmer MD.

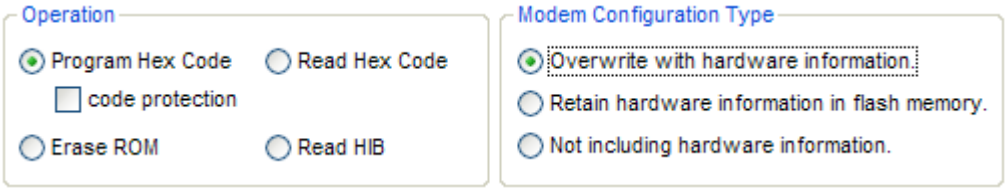


Example : "-fsh_write" option of main()

```
printf("\tCommand : fsh_write\n");
printf("\tFile Name : %s\n", argv[2]);
printf("================================================\n");

c_open_port(TTYS, 115200);
isp_sync();
printf("SYNC:SUCC\n");
fname = argv[2];
printf("Start...\n");
ihx_fsh_writ(fname);
printf("FLASH_WRITE:SUCC\n");
c_close_port();
```

3.3.1.2. Overwrite with hardware information

It downloads firmware hex code with new hardware information. It is same to figure below from Device-Programmer MD.



Example : "- hib_overwrite" option of main()

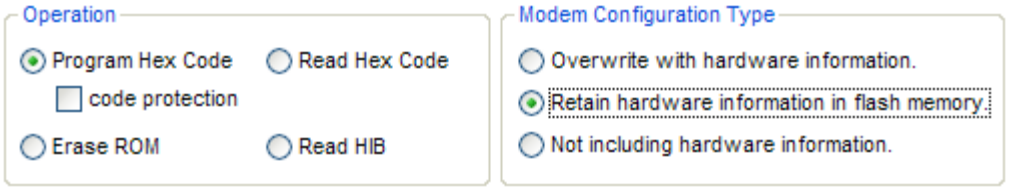
CAUTION : If the example is executed, Hardware Information area of original Intel Hex file is replaced.

```
printf("\tCommand : hib_overwrite\n");
printf("\tFile Name : %s\n", argv[2]);
printf("================================================\n");

int i=0;
int sum=0;
S_HIB HIB;
memset(&HIB, 0, sizeof(HIB));

HIB.MG2470.IEEEAddr[0] = 0x01;   HIB.MG2470.IEEEAddr[4] = 0x01;
HIB.MG2470.IEEEAddr[1] = 0x00;   HIB.MG2470.IEEEAddr[5] = 0x51;
HIB.MG2470.IEEEAddr[2] = 0x00;   HIB.MG2470.IEEEAddr[6] = 0x15;
HIB.MG2470.IEEEAddr[3] = 0x00;   HIB.MG2470.IEEEAddr[7] = 0x00;
HIB.MG2470.ChipId = CHIPID_MG245X;
HIB.MG2470.TxPower = 0;
HIB.MG2470.DataRate = DATARATE_250_K;
HIB.MG2470.StackID = STACKID_NONE;
HIB.MG2470.Ch = 0x0B;
HIB.MG2470.PanID[0] = 0x34;                HIB.MG2470.PanID[1] = 0x12;
```

```
		HIB.MG2470.NwkAddr[0] = 0x01;	HIB.MG2470.NwkAddr[1] = 0x00;
		HIB.MG2470.SecLevel;
		HIB.MG2470.PreConfig;
		HIB.MG2470.NetworkKey;
		HIB.MG2470.Rsv_0;
		HIB.MG2470.Rsv_EPID;
		HIB.MG2470.Rsv_1;

		for(i = 0; i < (MG2470_HIB_SIZE-1); i++)
		{
			sum += HIB.Value[i];
		}
		HIB.Value[MG2470_HIB_SIZE-1] = 0 - sum;

		unsigned char *codes = (unsigned char
*)malloc(MG2470_FLASH_BANK_SIZE*4+MG2470_FLASH_INFO_SIZE);
		memset(codes,0xFF,MG2470_FLASH_BANK_SIZE*4+MG2470_FLASH_INFO_SIZE);

		fname = argv[2];
		intelhex_read(fname, codes);
		memcpy(&codes[MG2470_HIB_OFFSET], HIB.Value, MG2470_HIB_SIZE);
		intelhex_writ(fname, codes);
		free(codes);

		printf("\n");
		printf("IEEE Address : 0x%02X%02X%02X%02X%02X%02X%02X%02X\n",
HIB.MG2470.IEEEAddr[7], HIB.MG2470.IEEEAddr[6],
			HIB.MG2470.IEEEAddr[5], HIB.MG2470.IEEEAddr[4],
HIB.MG2470.IEEEAddr[3], HIB.MG2470.IEEEAddr[2],
			HIB.MG2470.IEEEAddr[1], HIB.MG2470.IEEEAddr[0]);
		printf("\n");


		c_open_port(TTYS, 115200);
		isp_sync();
		printf("SYNC:SUCC\n");
		printf("Start...\n");
		ihx_fsh_writ(fname);
		printf("HIB_OVERWRITE :SUCC\n");
		c_close_port();
```

3.3.1.3. Retain hardware information in flash memory

It downloads firmware hex code with hardware information of existing Flash Memory. It is same to figure below from Device-Programmer MD.

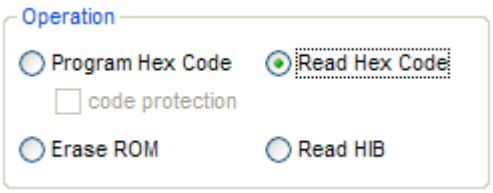


Example : "-hib_retain" option of main()
주의 : **CAUTION** : If the example is executed, Hardware Information area of original Intel Hex file is replaced.

```
        printf("\tCommand : hib_retain\n");
        printf("\tFile Name : %s\n", argv[2]);
        printf("=============================================\n");

        int i=0;
        int sum=0;
        S_HIB HIB;
        memset(&HIB, 0, sizeof(HIB));

        c_open_port(TTYS, 115200);
        char* hib_fname = "HIB.hex";
        isp_sync();
        printf("SYNC:SUCC\n");
        printf("Start HIB Read...\n");
        ihx_hib_read(hib_fname, MG2470_HIB_OFFSET, MG2470_FLASH_PAGE_SIZE);
        printf("HIB_READ:SUCC\n");

        unsigned char *codes = (unsigned char
*)malloc(MG2470_FLASH_BANK_SIZE*4+MG2470_FLASH_INFO_SIZE);
        memset(codes,0xFF,MG2470_FLASH_BANK_SIZE*4+MG2470_FLASH_INFO_SIZE);
        intelhex_read(hib_fname, codes);
        memcpy(HIB.Value, &codes[MG2470_HIB_OFFSET], MG2470_HIB_SIZE);

        memset(codes,0xFF,MG2470_FLASH_BANK_SIZE*4+MG2470_FLASH_INFO_SIZE);
        fname = argv[2];
        intelhex_read(fname, codes);
        memcpy(&codes[MG2470_HIB_OFFSET], HIB.Value, MG2470_HIB_SIZE);
        intelhex_writ(fname, codes);
        free(codes);

        printf("\n");
        printf("IEEE Address : 0x%02X%02X%02X%02X%02X%02X%02X%02X\n",
HIB.MG2470.IEEEAddr[7], HIB.MG2470.IEEEAddr[6],
                HIB.MG2470.IEEEAddr[5], HIB.MG2470.IEEEAddr[4],
HIB.MG2470.IEEEAddr[3], HIB.MG2470.IEEEAddr[2],
                HIB.MG2470.IEEEAddr[1], HIB.MG2470.IEEEAddr[0]);
        printf("\n");

        printf("Start WRITE...\n");
        ihx_fsh_writ(fname);
        printf("HIB_RETAIN :SUCC\n");
        c_close_port();
```

3.3.2. Read Hex Code

Read entire Hex code of MG2470 Flash Memory. It is same to figure below from Device-Programmer MD.



Example : "-fsh_read" option of main()

```
printf("\tCommand : fsh_read\n");
printf("\tFile Name : %s\n", argv[2]);
printf("================================================\n");

c_open_port(TTYS, 115200);
isp_sync();
printf("SYNC:SUCC\n");
fname = argv[2];
printf("Start...\n");
ihx_fsh_read(fname);
printf("FLASH_READ:SUCC\n");
c_close_port();
```

3.3.3. Initialization

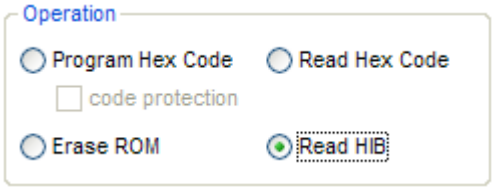
Initialize entire Flash Memory of MG2470. It is same to figure below from Device-Programmer MD.

Example : "-erase" option of main()

```
printf("\tCommand : erase\n");
printf("================================================\n");

c_open_port(TTYS, 115200);
//Erase Reference Cell
printf("Erase Reference Cell....");
isp_fsh_rers();
printf("Success\n");
//Mass erase
printf("Mass erase....");
isp_fsh_mers(0xFFFF);
printf("Success\n");
printf("ERASE:SUCC\n");
c_close_port();
```

3.3.4. Read Hardware Information

Read only hardware information from MG2470. It is same to figure below from Device-Programmer MD.



Example : “-hib_read” option of main()

```
        printf("\tCommand : hib_read\n");
        printf("\tFile Name : %s\n", argv[2]);
        printf("================================================\n");

        c_open_port(TTYS, 115200);
        fname = argv[2];
        isp_sync();
        printf("SYNC:SUCC\n");
        printf("Start...\n");
        ihx_hib_read(fname, MG2470_HIB_OFFSET, MG2470_FLASH_PAGE_SIZE);
        printf("HIB_READ:SUCC\n");
        c_close_port();

        S_HIB HIB;
        memset(&HIB, 0, sizeof(HIB));
        unsigned char *codes = (unsigned char
*)malloc(MG2470_FLASH_BANK_SIZE*4+MG2470_FLASH_INFO_SIZE);
        memset(codes,0xFF,MG2470_FLASH_BANK_SIZE*4+MG2470_FLASH_INFO_SIZE);
        intelhex_read(fname, codes);
        memcpy(HIB.Value, &codes[MG2470_HIB_OFFSET], MG2470_HIB_SIZE);
        free(codes);

        printf("\n");
        printf("IEEE Address : 0x%02X%02X%02X%02X%02X%02X%02X%02X\n",
HIB.MG2470.IEEEAddr[7], HIB.MG2470.IEEEAddr[6],
                HIB.MG2470.IEEEAddr[5], HIB.MG2470.IEEEAddr[4],
HIB.MG2470.IEEEAddr[3], HIB.MG2470.IEEEAddr[2],
                HIB.MG2470.IEEEAddr[1], HIB.MG2470.IEEEAddr[0]);
```

4. Hardware Information Base (HIB)

Hardware Information is for reading writing in specific Flash Memory after selecting basic information used in the library provided by RadioPulse.

4.1. Start Address

Starting address of Flash is 0x1000.

4.2. Total Size

The size allotted to entire Hardware Information Base is 64bytes.

4.3. Structure of Hardware Information Base

Name	Byte	Description
IEEE Address	8	64bit IEEE Address. MSB of Address is stored in MSB of IEEE_ADDR
ChipID	1	1=MG2400-F48, 2=MG2450.55, 3=MG2470
Transmit Power	1	0~25
Data Rate	1	0= 31.25 Kbps, 1= 62.50 Kbps, 2= 125 Kbps 3= 250 Kbps, 4= 500 Kbps, 5= 1.0 Mbps, 6= 1.3 Mbps, 7= 1.5 Mbps, 8= 2.0 Mbps, 9= 2.6 Mbps, 10= 3.0 Mbps
Stack Identitier	1	0x00=None, 0x01=IEEE 802.15.4 0x02=RF4CE, 0x10=Zigbee2004, 0x11=ZigBee2005, 0x12=ZigBee2006, 0x13=ZigBee2007/ZigBee Pro
RF Channel	1	Initial RF channel.(0x0B~0x1A)
Pan ID	2	16bit PanID. PanID[15:8] = Array[1] PanID[7:0] = Array[0]
Network Address	2	16bit Short(or Network) Address ShortAddr[15:8] = Array[1] ShortAddr[7:0] = Array[0]
Security Level	1	0=No Security, 1=MIC32, 2=MIC64, 3=MIC128 4=ENC, 5=ENCMIC32, 6=ENCMIC64, 7=ENCMIC128
PreConfig-Mode	1	Pre-configured Mode Value
Network Key	16	Network Key for Security MSB of Key is stored in MSB of Network Key.
Reserved_0	8	8Byte reserved buffer
Extended PanID	8	64bit Extended PanID MSB of PanID is stored in MSB of Extended PanID.
Reserved_1	8	8Byte reserved buffer
General Word-0	2	2Byte reserved buffer
General Word-1	2	2Byte reserved buffer
CSUM	1	CheckSum Value =0-(sum value of 63byte in IEEE Address~General Word-1) It means that sum value of Hardware Infromation 64 byte is set as 0.

Flash Address
IEEE_ADDR[7]: 0x1000
IEEE_ADDR[6]: 0x1001
IEEE_ADDR[5]: 0x1002

….
General_Word_1[1] : 0x103D
General_Word_1[0] : 0x103E
CSUM address : 0x103F

5. ISP COMMAND

In ISP mode, MG2470 has a ROM as Boot Loader in the chip. The ISP function is executed by this ROM code in ISP mode.

[Table 1] is a structure for ISP command transmitting.

Table 1. Definition of ISP Command Structure

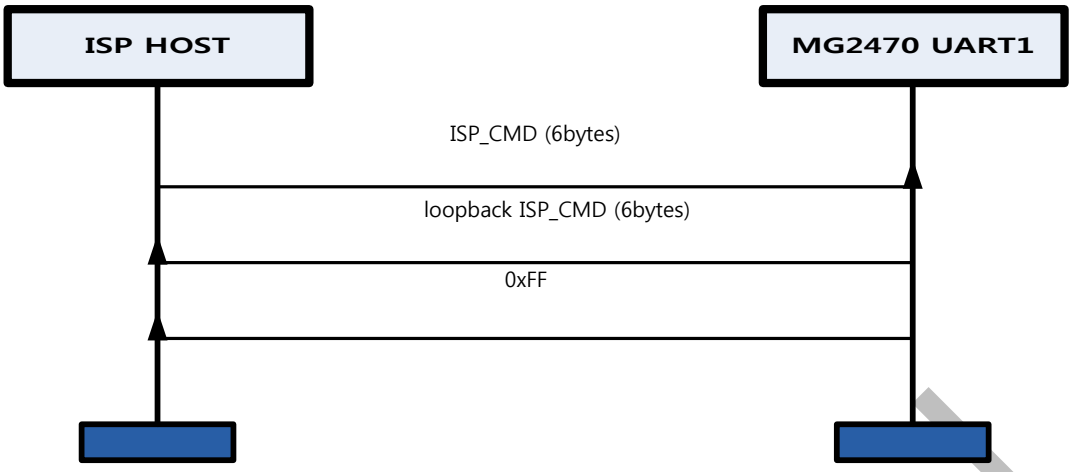
Byte : 1	1	1	1	1	1
SYNC(0x55)	CMD	ADDRH	ADDRL	LENH	LENL
		ADDR		LEN	

- SYNC : Sync Pattern (0x55)
- CMD : Command code supported by MG2470 ROM (Refer to [Table 2].)

Table 2. ISP Command List

Identifier	Description	
0x00	Flash Mass Erase	
0x01	Flash Erase Reference Cell	
0x02	Flash Page Erase	
0x03	Flash Checksum	
0x04	Flash Bank Select	
0x05	Flash Code Execution	
0x06	Flash Status	Not Used
0x07	XDATA Read	Not Used
0x08	XDATA Write	

- ADDR : 16bit address(2byte)
- LEN : 16bit length (2byte)



ISP_CMD Format Error (isp_cmd[0] != 0x55 or isp_cmd[1] > 0x09)

Figure 2. ISP Command Procedure

5.1. Flash Mass Erase - 0x00

Table 3. Definition of ISP Command Structure

Byte : 1	1	1	1	1	1
SYNC	CMD	ADDRH	ADDRL	LENH	LENL
0x55	0x00	Don't Care		Don't Care	

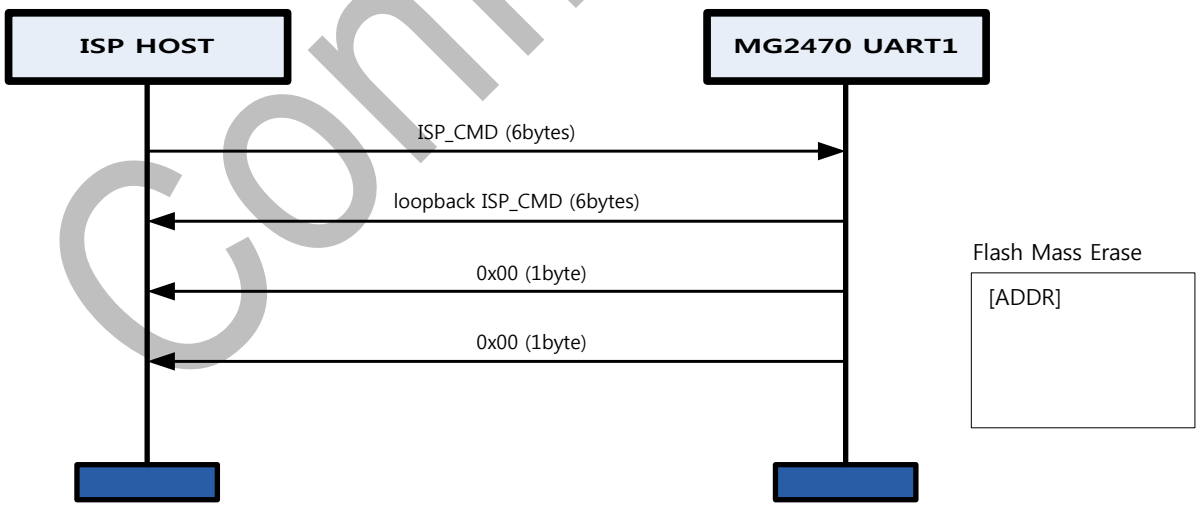


Figure 3. Flash Mass Erase Command Procedure

5.2. Flash Erase Reference Cell - 0x01

Table 4. Definition of ISP Command Structure

Byte : 1	1	1	1	1	1

SYNC	CMD	ADDRH	ADDRL	LENH	LENL
0x55	0x01	Don't Care		Don't Care	

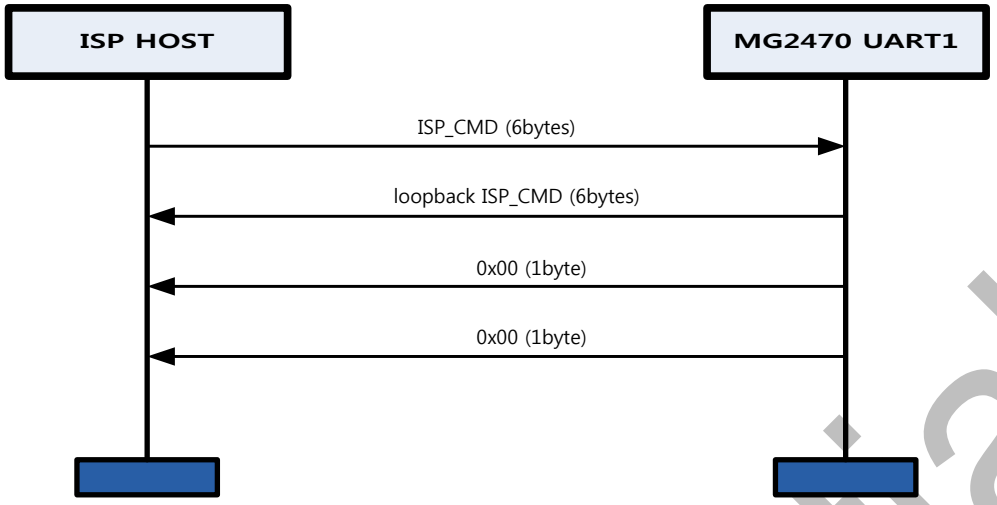


Figure 4. Flash Erase Reference Cell Command Procedure

5.3. Flash Page Erase - 0x02

Table 5. Definition of ISP Command Structure

Byte : 1	1	1	1	1	1
SYNC	CMD	ADDRH	ADDRL	LENH	LENL
0x55	0x02	16bit address(2byte)		Don't Care	

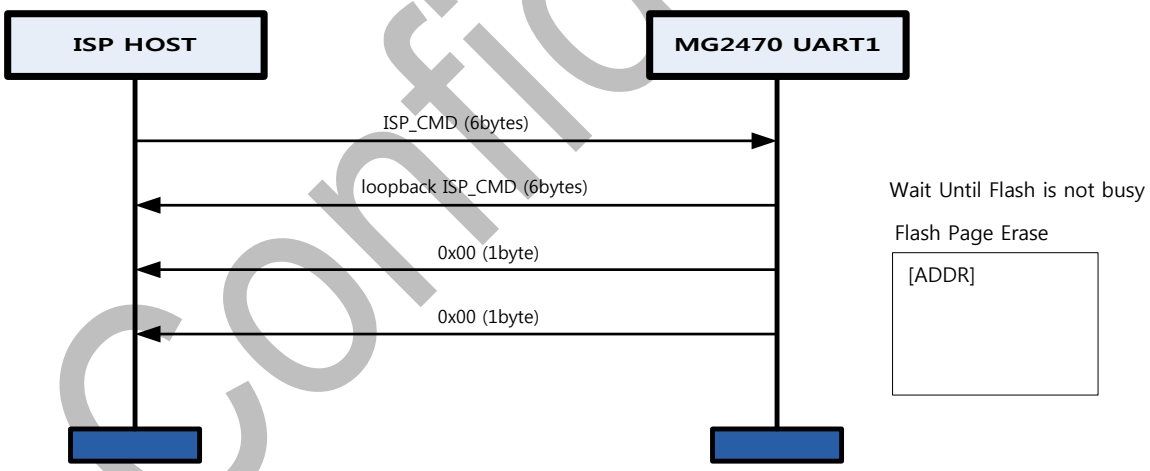


Figure 5. Flash Page Erase Command Procedure

5.4. Flash Checksum - 0x03

Table 6. Definition of ISP Command Structure

Byte : 1	1	1	1	1	1
SYNC	CMD	ADDRH	ADDRL	LENH	LENL
0x55	0x03	16bit address (2byte)		16bit length (2byte)	

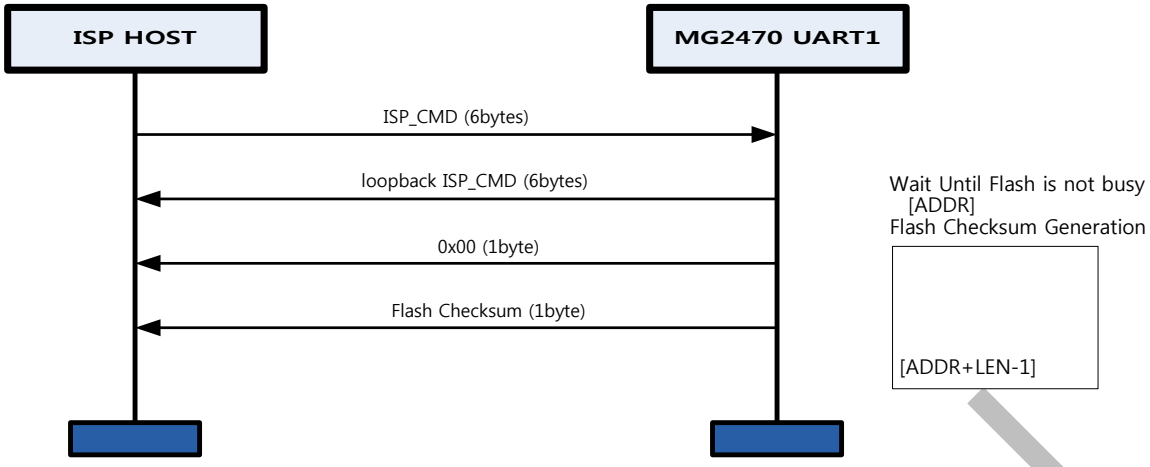


Figure 6. Flash Checksum Command Procedure

5.5. Flash Bank Select - 0x04

Table 7. Definition of ISP Command Structure

Byte : 1	1	1	1	1	1
SYNC	CMD	ADDRH	ADDRL	LENH	LENL
0x55	0x04	Don't Care		16bit length (2byte)	

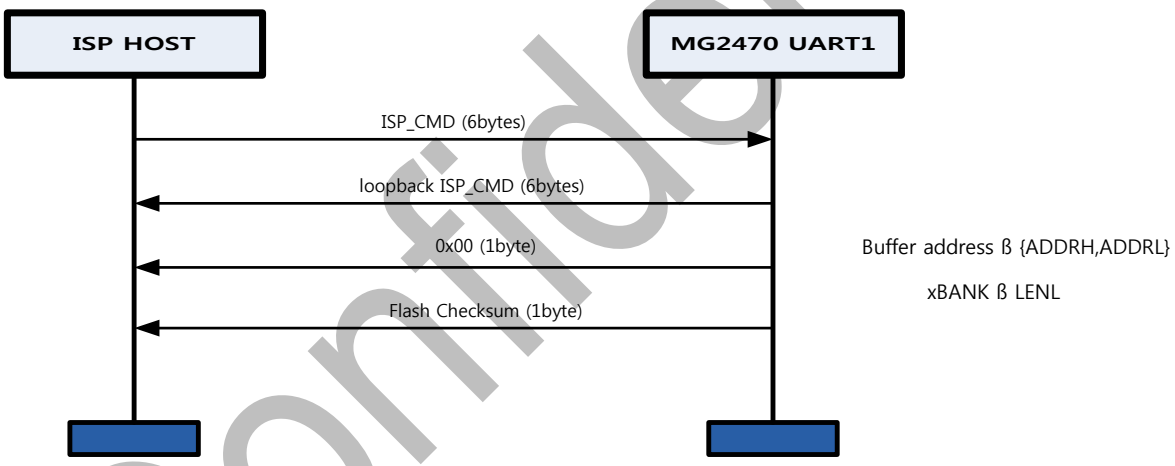


Figure 7. Flash Bank Select Command Procedure

5.6. Flash Code Execution - 0x05

Table 8. Definition of ISP Command Structure

Byte : 1	1	1	1	1	1
SYNC	CMD	ADDRH	ADDRL	LENH	LENL
0x55	0x05	16bit address (2byte)		Don't Care	

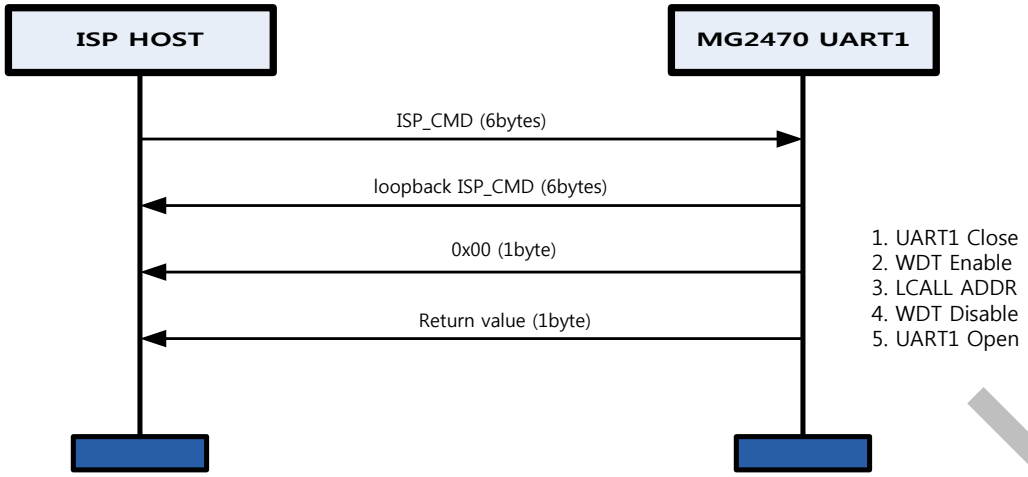


Figure 8. Flash Code Execution Command Procedure

5.7. XDATA Write - 0x08

Table 9. Definition of ISP Command Structure

Byte : 1	1	1	1	1	1
SYNC	CMD	ADDRH	ADDRL	LENH	LENL
0x55	0x08	16bit address (2byte)		16bit length (2byte)	

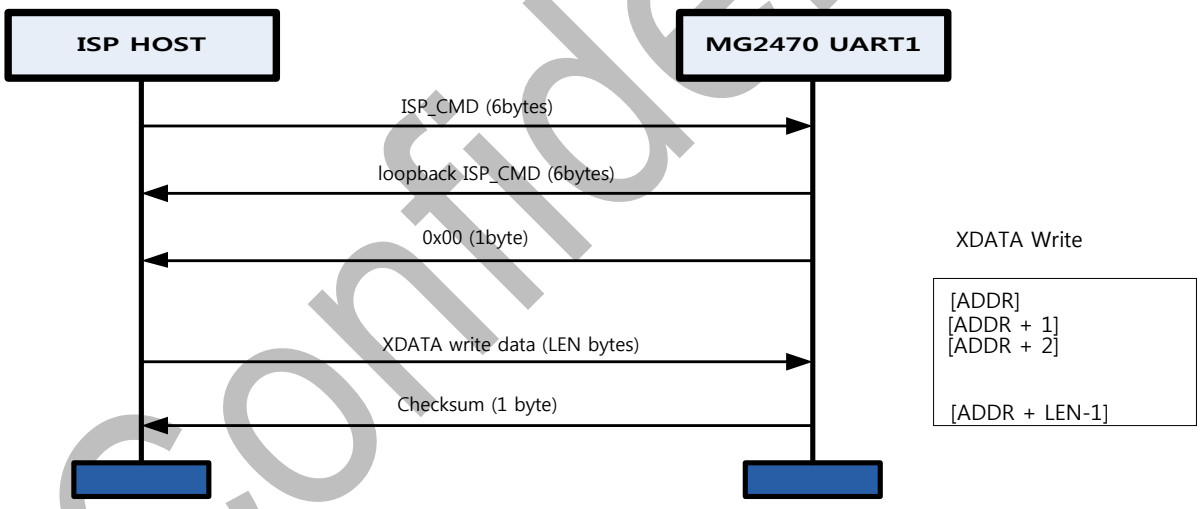


Figure 9. XData Write Command Procedure

RadioPulse Inc

3rd Fl., Hans B/D II, 111-6 Seongnae-Dong,
Gangdong-Gu, Seoul, Korea, 134-883, Korea

URL: www.radiopulse.co.kr
Tel: +82-2-478-2963~5
Fax: +82-2-478-2967

sales@radiopulse.co.kr

About RadioPulse Inc.

RadioPulse is a Being Wireless solution provider offering wireless communication & network technologies and developing next generation wireless networking technologies.

The new wireless networking solutions envisioned by RadioPulse will enable user to enjoy wireless technologies with easy interface.