MIAT_STM32
內部Flash存取控制實驗

浯陽科技有限公司

WU-YANG

Technology Co., Ltd.

Declared Version

Training Only	
Declare	
Document Number	
Document Version	***1.00***
Release Date	
Document Title	MIAT_STM32 內部FLASH存取控制實驗
Exercise Time	■
Platform	■ ***MIAT_STM32.V2*** ■ ***MIAT IOB.V1***
Peripheral	■
Author	■ ***WU-YANG Technology Co., Ltd.***

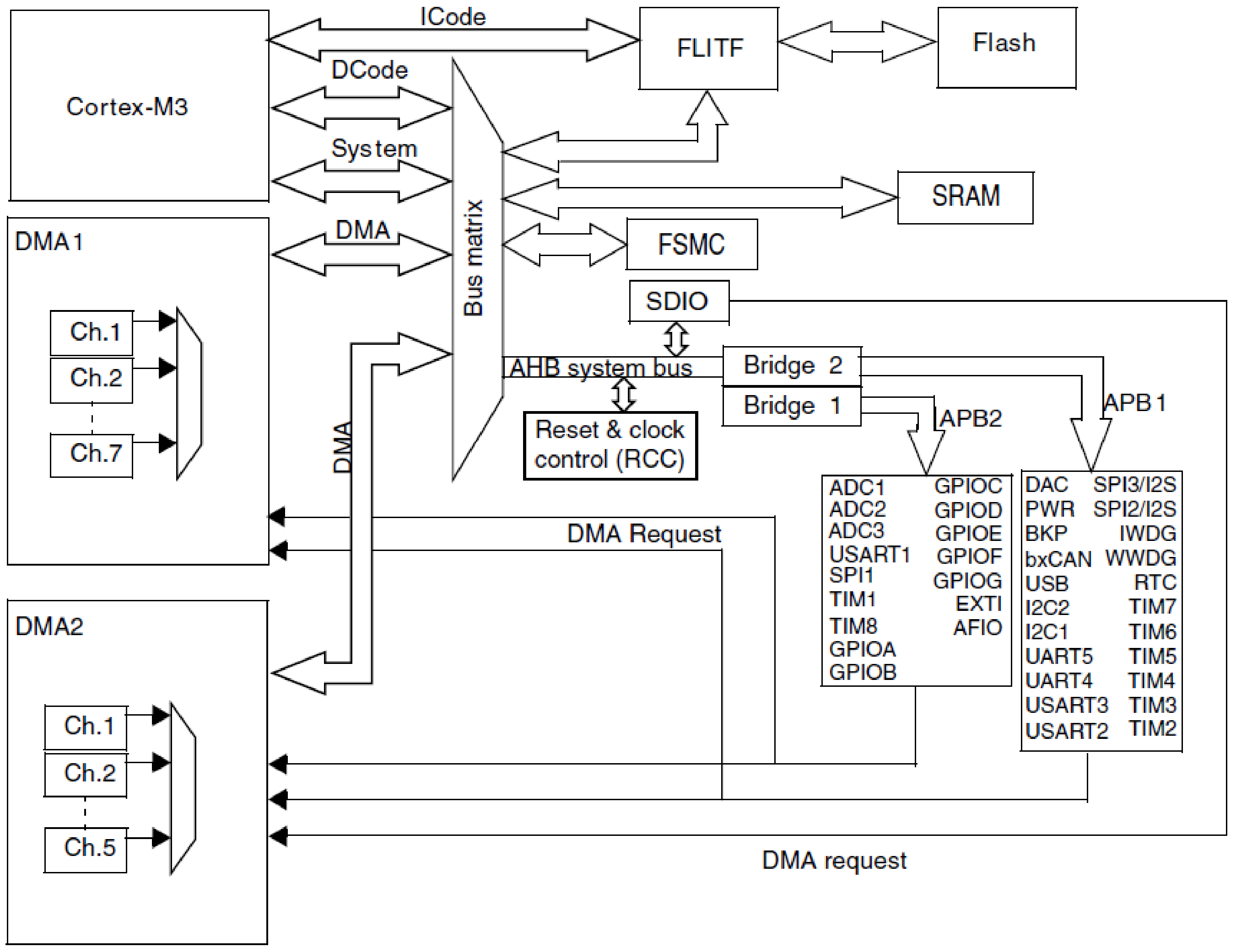
實驗目的(一)

- 使用MIAT_STM32實驗板透過Flash memory interface (FLITF)控制內部Flash進行存取控制實驗，並利用LED確認存取是否正常。

實驗原理

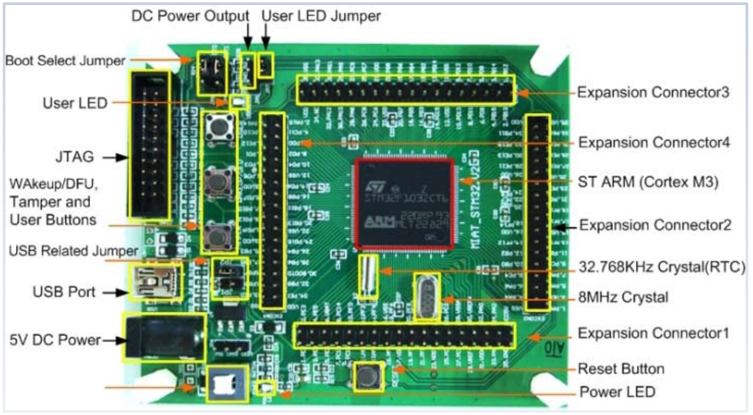
- System architecture
- Embedded Flash
  - Features
  - Memory map
  - RVMDK環境設定
  - FLITF
  - Flash module organization
- Flash library function
- Development Flow
- ARM Configure

System architecture

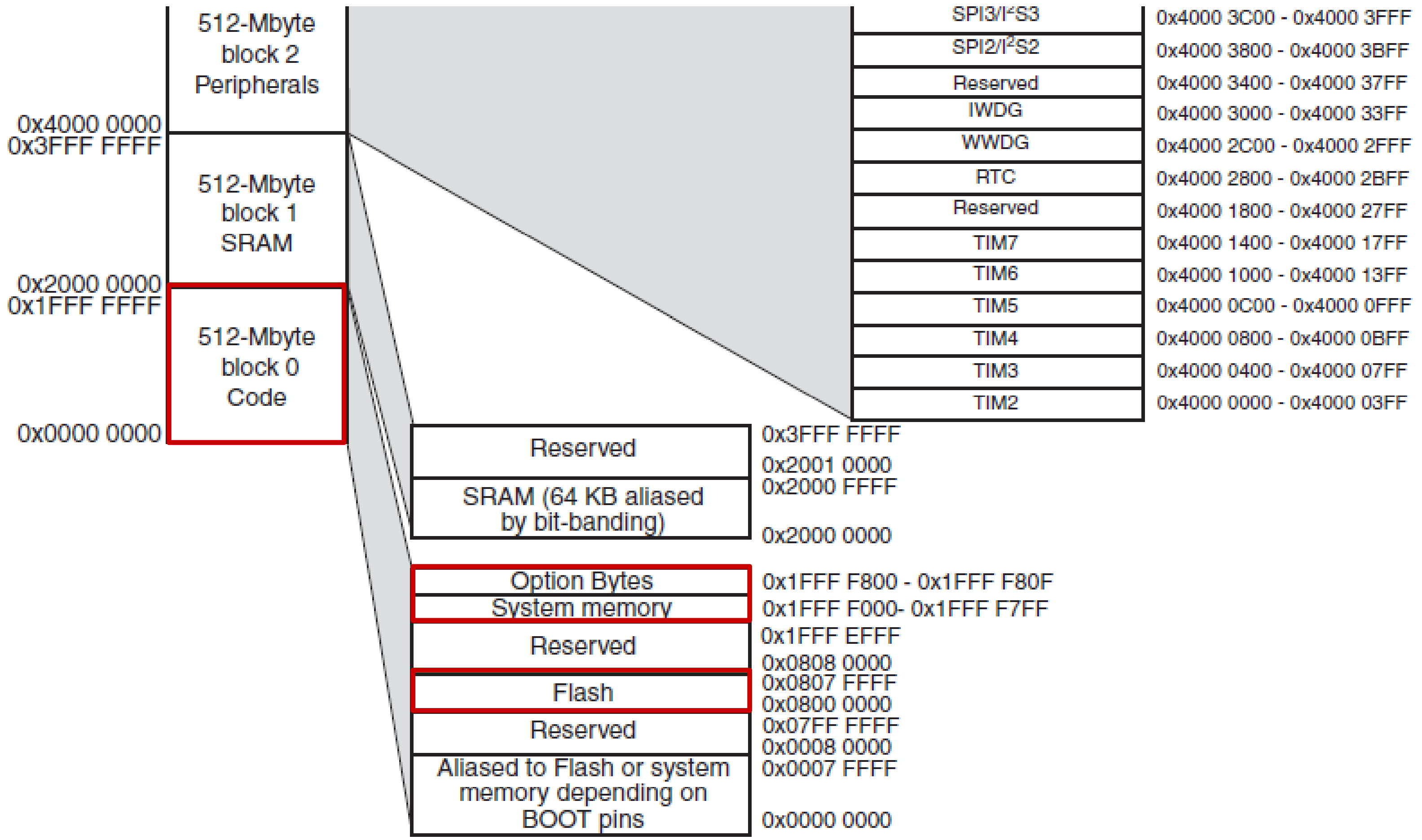


Embedded Flash

- Features
- up to 256 Kbytes
- Memory organization: the Flash memory is organized as a main block and an information block:
  - Main memory block of size:
    - up to 32 Kb $\times$ 64 bits divided into 128 pages of 2 Kbytes
  - Information block of size:
    - 258 $\times$ 64 bits



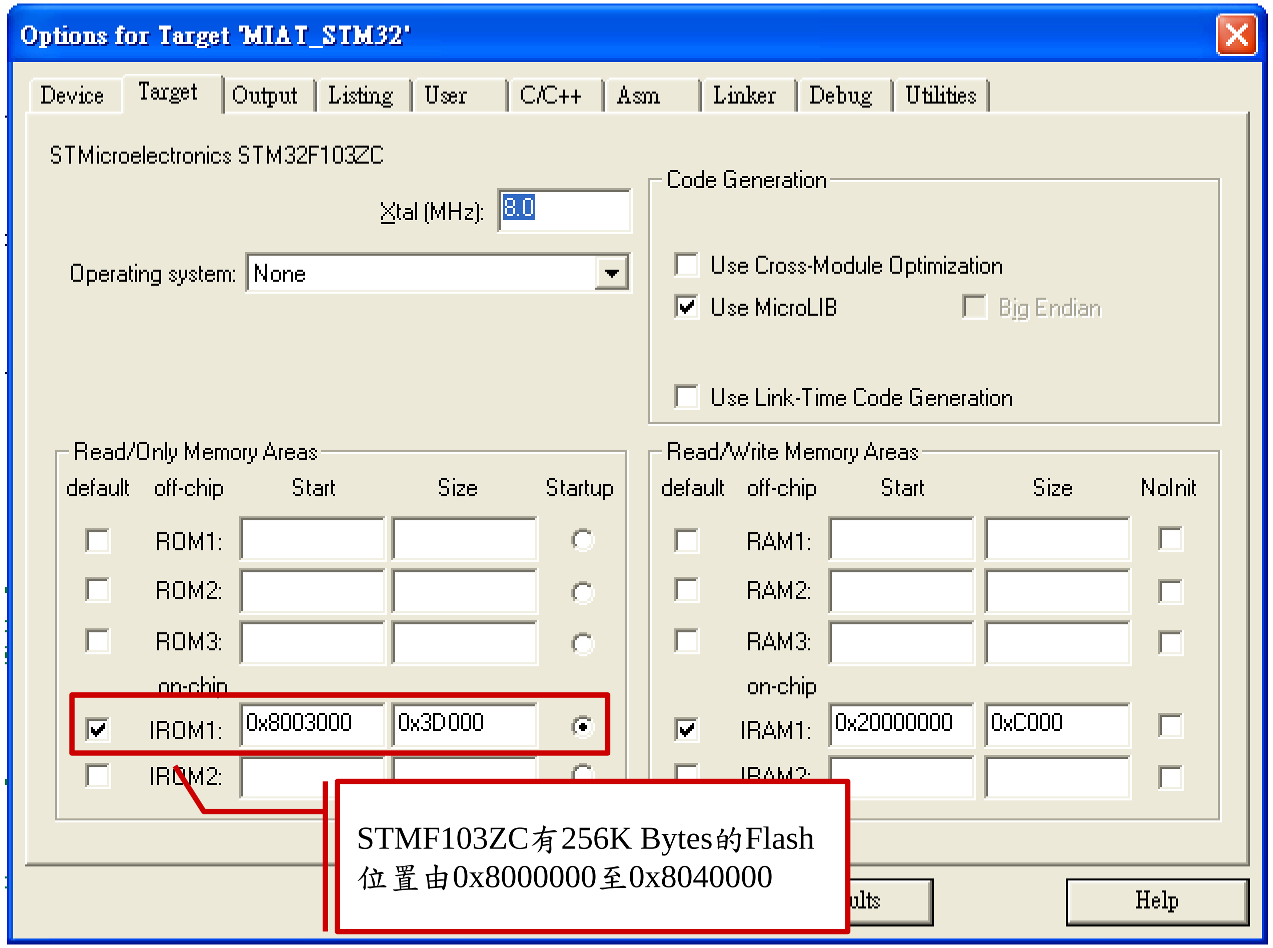
Embedded Flash Memory map


512-Mbyte block 2 Peripherals
0x4000 0000
0x3FFF FFFF
512-Mbyte block 1 SRAM
0x2000 0000
0x1FFF FFFF
512-Mbyte block 0 Code
0x0000 0000
SPI3/I²S3 0x4000 3C00 - 0x4000 3FFF
SPI2/I²S2 0x4000 3800 - 0x4000 3BFF
Reserved 0x4000 3400 - 0x4000 37FF
IWDG 0x4000 3000 - 0x4000 33FF
WWDG 0x4000 2C00 - 0x4000 2FFF
RTC 0x4000 2800 - 0x4000 2BFF
Reserved 0x4000 1800 - 0x4000 27FF
TIM7 0x4000 1400 - 0x4000 17FF
TIM6 0x4000 1000 - 0x4000 13FF
TIM5 0x4000 0C00 - 0x4000 0FFF
TIM4 0x4000 0800 - 0x4000 0BFF
TIM3 0x4000 0400 - 0x4000 07FF
TIM2 0x4000 0000 - 0x4000 03FF
Reserved
0x3FFF FFFF
0x2001 0000
SRAM (64 KB aliased by bit-banding)
0x2000 FFFF
0x2000 0000
Option Bytes 0x1FFF F800 - 0x1FFF F80F
System memory 0x1FFF F000- 0x1FFF F7FF
Reserved
0x1FFF EFFF
0x0808 0000
Flash
0x0807 FFFF
0x0800 0000
Reserved
0x07FF FFFF
0x0008 0000
Aliased to Flash or system memory depending on BOOT pins
0x0007 FFFF
0x0000 0000

Flash Memory

Block	Name	Base addresses	Size (bytes)
Main memory	Page 0	0x0800 0000 - 0x0800 07FF	2 Kbytes
	Page 1	0x0800 0800 - 0x0800 0FFF	2 Kbytes
	Page 2	0x0800 1000 - 0x0800 17FF	2 Kbytes
	Page 3	0x0800 1800 - 0x0800 1FFF	2 Kbytes
	⋮	⋮	⋮
	Page 255	0x0807 F800 - 0x0807 FFFF	2 Kbytes
Information block	System memory	0x1FFF F000 - 0x1FFF F7FF	2 Kbytes
	Option Bytes	0x1FFF F800 - 0x1FFF F80F	16

RVMDK環境設定



STMicroelectronics STM32F103ZC

Xtal (MHz): 8.0

Operating system: None

Code Generation

- [ ] Use Cross-Module Optimization
- [✓] Use MicroLIB
- [ ] Big Endian
- [ ] Use Link-Time Code Generation

Read/Only Memory Areas

default	off-chip	Start	Size	Startup
☐	ROM1:			○
☐	ROM2:			○
☐	ROM3:			○
	on-chip			
✓	IROM1:	0x8003000	0x3D000	●
☐	IROM2:			○

Read/Write Memory Areas

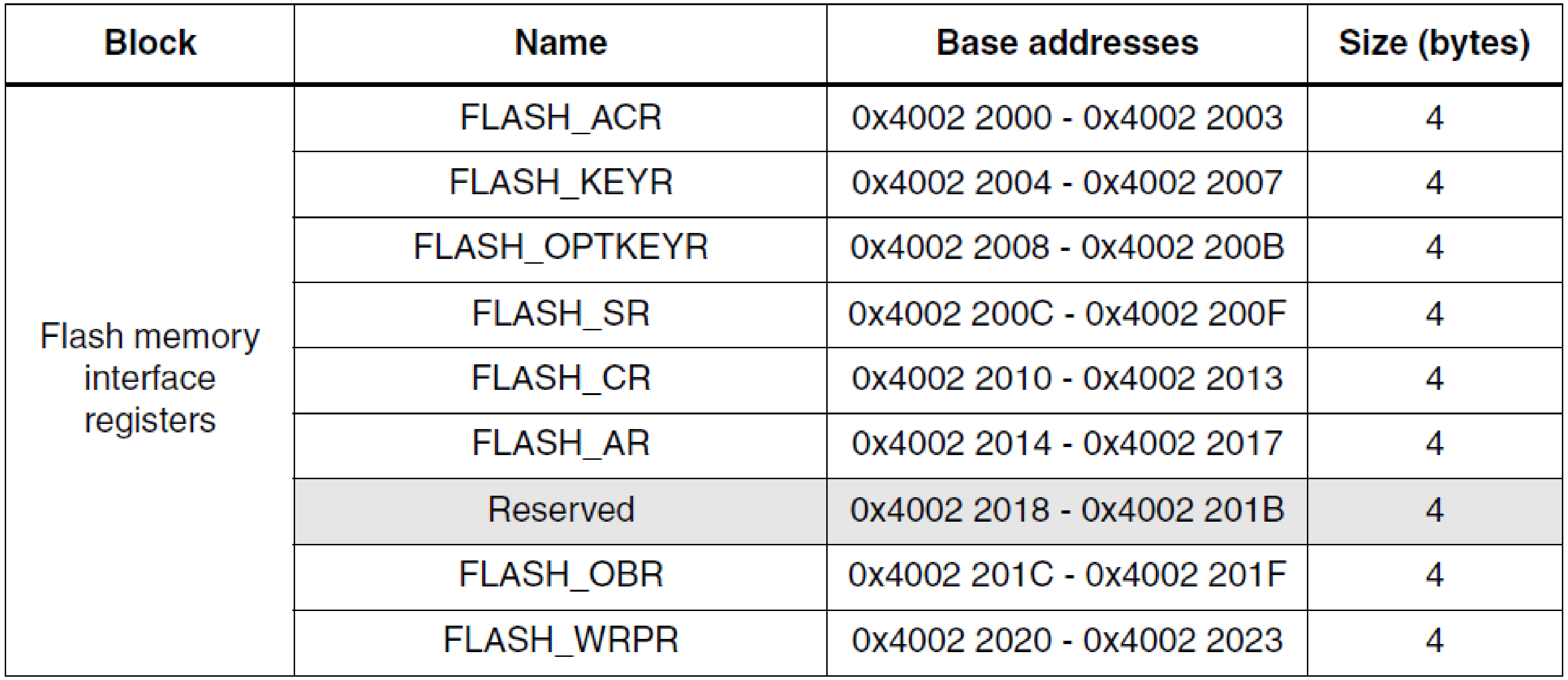
default	off-chip	Start	Size	NoInit
☐	RAM1:			☐
☐	RAM2:			☐
☐	RAM3:			☐
	on-chip			
✓	IRAM1:	0x20000000	0xC000	☐
☐	IRAM2:			☐

STMF103ZC有256K Bytes的Flash
位置由0x8000000至0x8040000

FLITF

- Features
  - Read interface with prefetch buffer (2x64-bit words)
  - Option byte Loader
  - Flash Program / Erase operation
  - Read / Write protection

Flash memory interface



Block	Name	Base addresses	Size (bytes)
Flash memory interface registers	FLASH_ACR	0x4002 2000 - 0x4002 2003	4
	FLASH_KEYR	0x4002 2004 - 0x4002 2007	4
	FLASH_OPTKEYR	0x4002 2008 - 0x4002 200B	4
	FLASH_SR	0x4002 200C - 0x4002 200F	4
	FLASH_CR	0x4002 2010 - 0x4002 2013	4
	FLASH_AR	0x4002 2014 - 0x4002 2017	4
	Reserved	0x4002 2018 - 0x4002 201B	4
	FLASH_OBR	0x4002 201C - 0x4002 201F	4
	FLASH_WRPR	0x4002 2020 - 0x4002 2023	4

Flash memory interface register

_Flash status register (FLASH_SR)_

31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved										EOP	WRPRT ERR	Res.	PG ERR	Res.	BSY
										rw	rw		rw		r

Bits 31:6 Reserved, must be kept cleared.
Bit 5 **EOP: End of operation**
Set by hardware when a Flash operation (programming / erase) is completed. Reset by writing a 1
Note: EOP is asserted at the end of each successful program or erase operation
Bit 4 **WRPRTERR: Write protection error**
Set by hardware when programming a write-protected address of the Flash memory.
Reset by writing 1.
Bit 3 Reserved, must be kept cleared.
Bit 2 **PGERR: Programming error**
Set by hardware when an address to be programmed contains a value different from '0xFFFF' before programming.
Reset by writing 1.
Note: The STRT bit in the FLASH_CR register should be reset before starting a programming operation.
Bit 1 Reserved, must be kept cleared
Bit 0 **BSY: Busy**
This indicates that a Flash operation is in progress. This is set on the beginning of a Flash operation and reset when the operation finishes or when an error occurs.

Flash memory interface register

Flash control register (FLASH_CR)

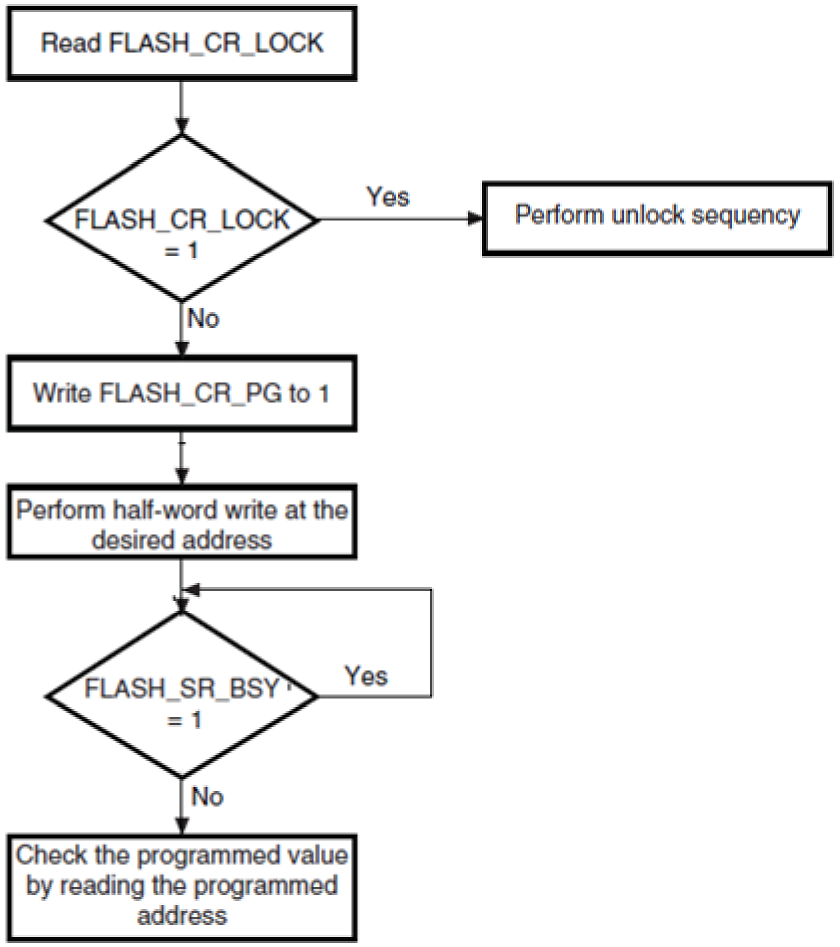
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Reserved															

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
Reserved			EOPIE	Res.	ERRIE	OPTWRE	Res.	LOCK	STRT	OPTER	OPTPG	Res.	MER	PER	PG
			rw		rw	rw		rw	rw	rw	rw		rw	rw	rw

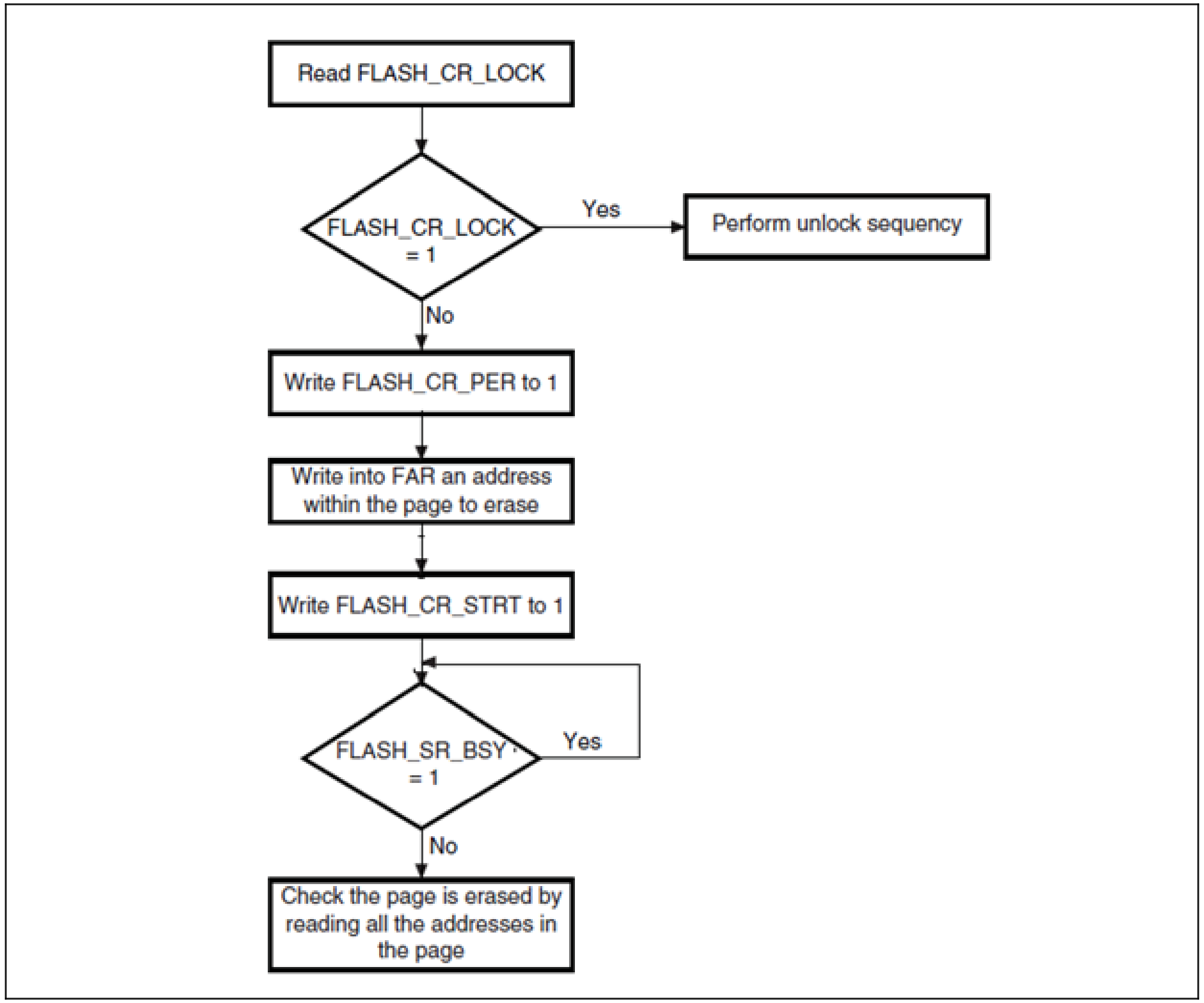
Bit 9 **OPTWRE: Option bytes write enable**
When set, the option bytes can be programmed. This bit is set on writing the correct key sequence to the FLASH_OPTKEYR register.
This bit can be reset by software
Bit 8 Reserved, must be kept cleared.
Bit 7 **LOCK: Lock**
Write to 1 only. When it is set, it indicates that the FPEC and FLASH_CR are locked. This bit is reset by hardware after detecting the unlock sequence.
In the event of unsuccessful unlock operation, this bit remains set until the next reset.

Bit 6 **STRT: Start**
This bit triggers an ERASE operation when set. This bit is set only by software and reset when the BSY bit is reset.
Bit 5 **OPTER: Option byte erase**
Option byte erase chosen.
Bit 4 **OPTPG: Option byte programming**
Option byte programming chosen.
Bit 3 Reserved, must be kept cleared.
Bit 2 **MER: Mass erase**
Erase of all user pages chosen.
Bit 1 **PER: Page erase**
Page Erase chosen.
Bit 0 **PG: Programming**
Flash programming chosen.

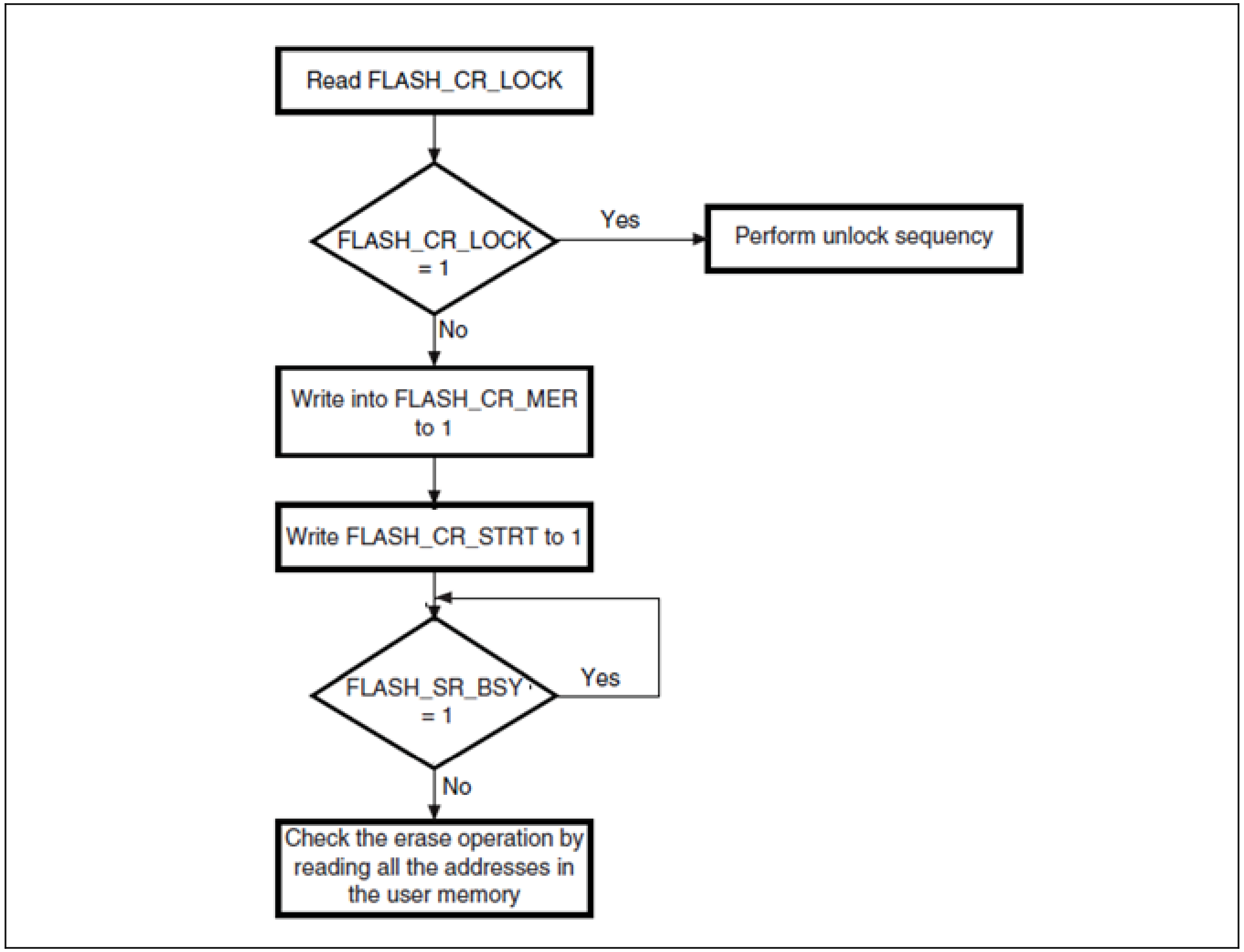
Main Flash memory programming


Read FLASH_CR_LOCK
FLASH_CR_LOCK = 1
Yes
Perform unlock sequency
No
Write FLASH_CR_PG to 1
Perform half-word write at the desired address
FLASH_SR_BSY = 1
Yes
No
Check the programmed value by reading the programmed address

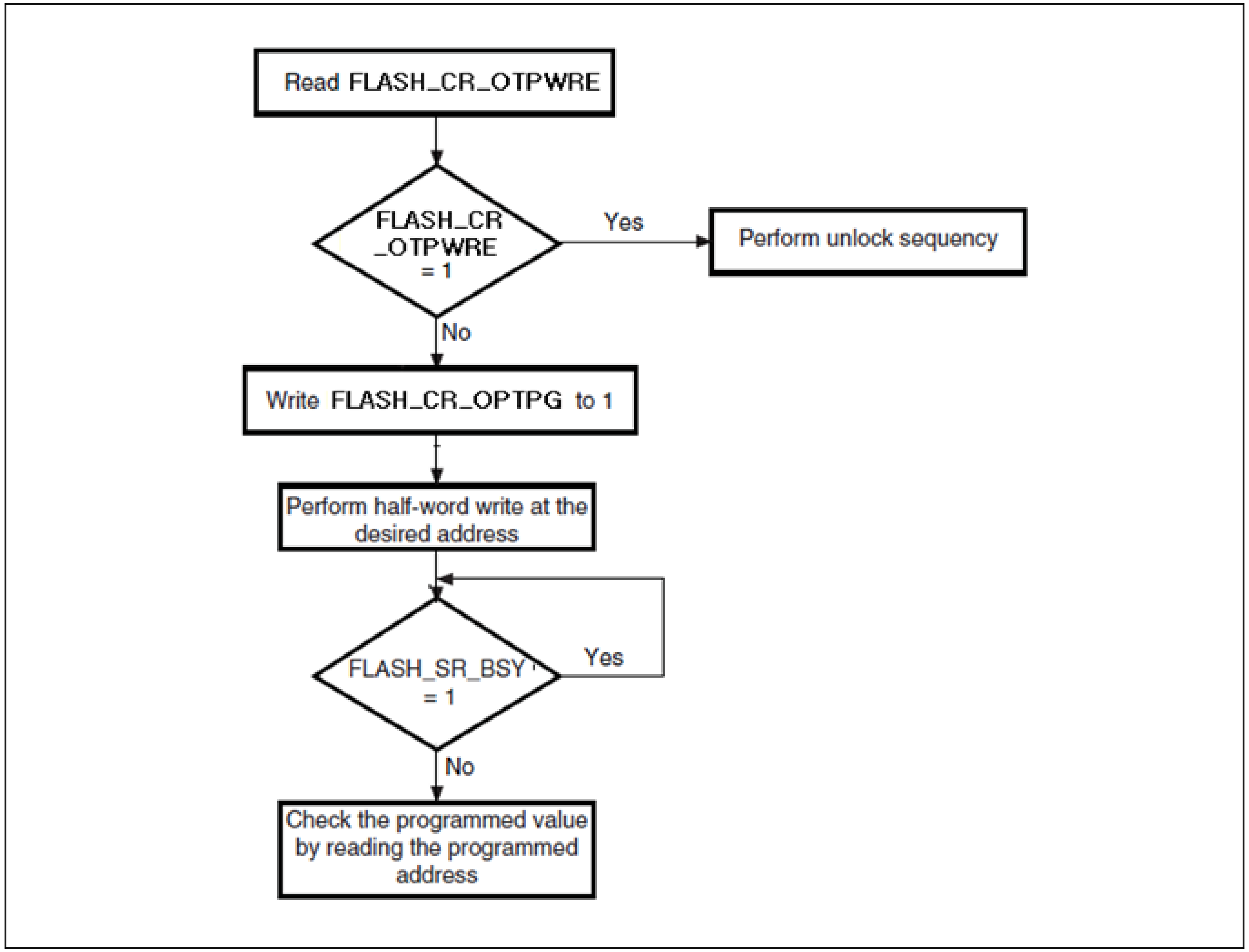
Flash memory Page Erase



Flash memory Mass Erase

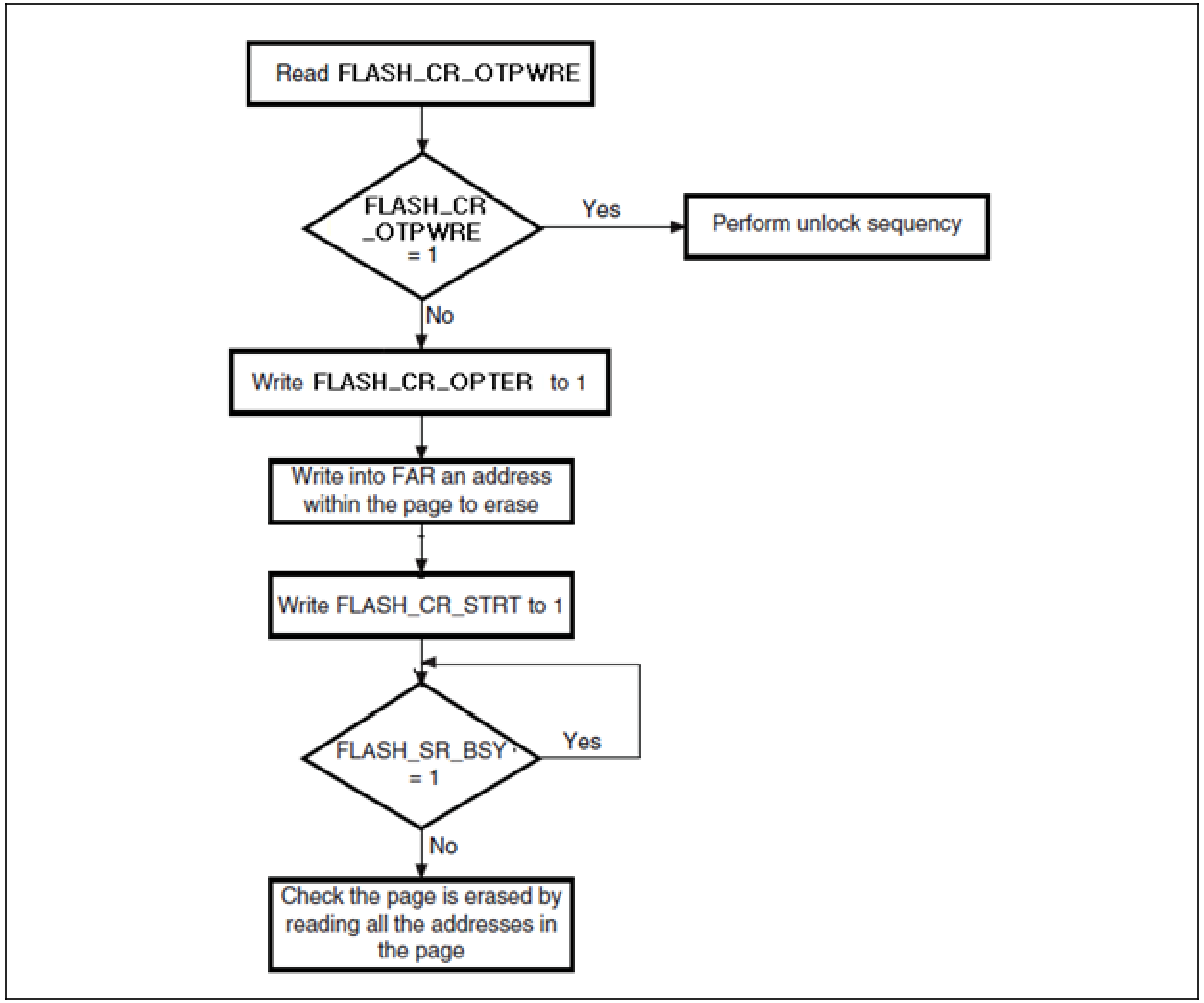


Option byte programming



Read FLASH_CR_OTPWRE
FLASH_CR _OTPWRE = 1
Yes
Perform unlock sequency
No
Write FLASH_CR_OPTPG to 1
Perform half-word write at the desired address
FLASH_SR_BSY = 1
Yes
No
Check the programmed value by reading the programmed address

Option byte Erase


Read FLASH_CR_OTPWRE
FLASH_CR _OTPWRE = 1
Yes
Perform unlock sequency
No
Write FLASH_CR_OPTER to 1
Write into FAR an address within the page to erase
Write FLASH_CR_STRT to 1
FLASH_SR_BSY = 1
Yes
No
Check the page is erased by reading all the addresses in the page

FLASH library function

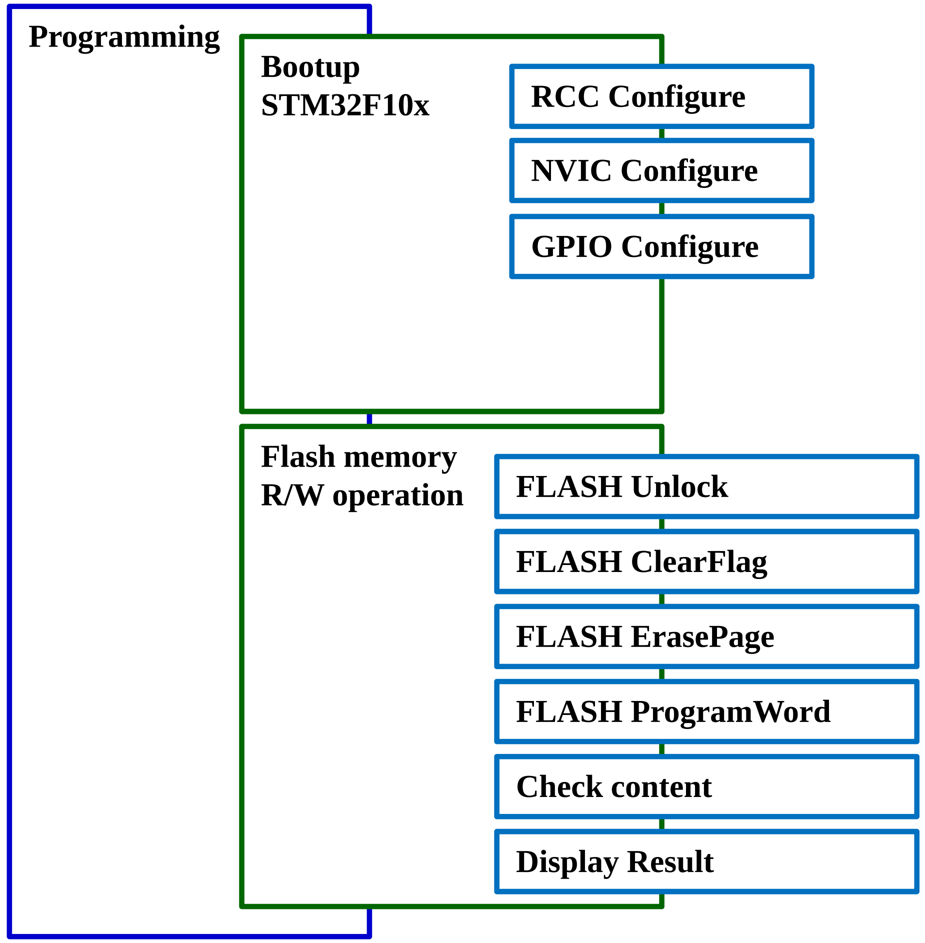
Function name	Description
FLASH_SetLatency	Sets the code latency value.
FLASH_HalfCycleAccessCmd	Enables or disables the Half cycle FLASH access.
FLASH_PrefetchBufferCmd	Enables or disables the Prefetch Buffer.
FLASH_Unlock	Unlocks the FLASH Program Erase Controller.
FLASH_Lock	Locks the Flash Program Erase Controller.
FLASH_ErasePage	Erases a specified FLASH page.
FLASH_EraseAllPages	Erases all FLASH pages.
FLASH_EraseOptionBytes	Erases the FLASH option bytes.
FLASH_ProgramWord	Programs a word at a specified address.
FLASH_ProgramHalfWord	Programs a half word at a specified address.
FLASH_ProgramOptionByteData	Programs a half word at a specified Option Byte Data address.
FLASH_EnableWriteProtection	Write protects the desired pages

FLASH library function

Function name	Description
FLASH_ReadOutProtection	Enables or disables the read out protection.
FLASH_UserOptionByteConfig	Programs the FLASH User Option Byte: IWDG_SW / RST_STOP / RST_STDBY.
FLASH_GetUserOptionByte	Returns the FLASH User Option Bytes values.
FLASH_GetWriteProtectionOptionByte	Returns the FLASH Write Protection Option Bytes Register value.
FLASH_GetReadOutProtectionStatus	Checks whether the FLASH Read Out Protection Status is set or not.
FLASH_GetPrefetchBufferStatus	Checks whether the FLASH Prefetch Buffer status is set or not.
FLASH_ITConfig	Enables or disables the specified FLASH interrupts.
FLASH_GetFlagStatus	Checks whether the specified FLASH flag is set or not.
FLASH_ClearFlag	Clears the FLASH pending flags.
FLASH_GetStatus	Returns the FLASH Status.
FLASH_WaitForLastOperation	Waits for a Flash operation to complete or a TIMEOUT to occur.

Development Flow

Embedded Software Side



Bootup STM32F10x

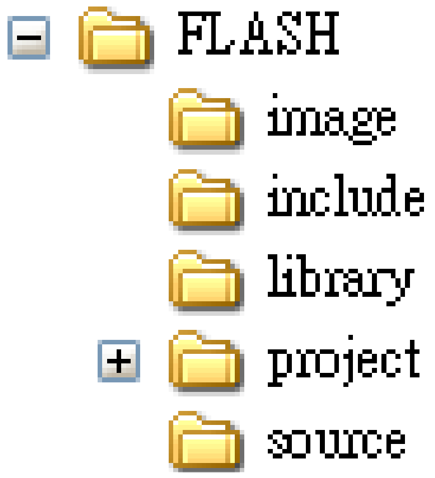
```
int main(void)
{
#ifdef DEBUG
  debug();
#endif

  FLASHStatus = FLASH_COMPLETE;
  MemoryProgramStatus = PASSED;
  Data = 0x15041975;

/* RCC Configuration */
  RCC_Configuration();

  /* NVIC Configuration */
  NVIC_Configuration();

  /* GPIO Configuration */
  GPIO_Configuration();


……………………………………………………..
```

實驗步驟

- 範例目錄架構
- 範例說明
- 預設定義說明
- 燒錄MIAT_STM32

範例目錄架構

- 範例目錄
  - 測試映像檔
  - 含括檔
  - 函式庫
  - 專案檔
  - 原始碼



範例說明

Flash FwLib Functions List

Function name	Description
FLASH_Unlock	Unlocks the FLASH Program Erase Controller.
FLASH_ClearFlag	Clears the FLASH pending flags.
FLASH_ErasePage	Erases a specified FLASH page.
FLASH_ProgramWord	Programs a word at a specified address.

```
 /* Unlock the Flash Program Erase controller */
 FLASH_Unlock();

 /* Define the number of page to be erased */
 NbrOfPage = (EndAddr - StartAddr) / FLASH_PAGE_SIZE;

 /* Clear All pending flags */
 FLASH_ClearFlag(FLASH_FLAG_BSY | FLASH_FLAG_EOP | FLASH_FLAG_PGERR |
FLASH_FLAG_WRPRTERR);

 /* Erase the FLASH pages */
 for(EraseCounter = 0; (EraseCounter < NbrOfPage) && (FLASHStatus ==
FLASH_COMPLETE); EraseCounter++)
 {
   FLASHStatus = FLASH_ErasePage(StartAddr + (FLASH_PAGE_SIZE * EraseCounter));
 }

 /*  FLASH Word program of data 0x15041979 at addresses defined by StartAddr and
EndAddr*/
 Address = StartAddr;

 while((Address < EndAddr) && (FLASHStatus == FLASH_COMPLETE))
 {
   FLASHStatus = FLASH_ProgramWord(Address, Data);
   Address = Address + 4;
 }
```

範例說明

Embedded Software Side

Flash memory
R/W operation

Check content

Display Result

Display Result

```
/* Check the corectness of written data */
 Address = StartAddr;

 while((Address < EndAddr) && (MemoryProgramStatus != FAILED))
 {
   if((*(vu32*) Address) != Data)
   {
     MemoryProgramStatus = FAILED;
   }
   Address += 4;
 }

 while (1)
 {
   if (MemoryProgramStatus == PASSED)
   {  /* OK Turn on USERLED */
     GPIO_SetBits(GPIOF, GPIO_Pin_11);
   }
   else
   { /* KO Turn off USERLED */
     GPIO_ResetBits(GPIOF, GPIO_Pin_11);
     /* Insert delay */
     Delay(0xAFFFF);
     /* Turn on USERLED */
     GPIO_SetBits(GPIOF, GPIO_Pin_11);
     /* Insert delay */
     Delay(0xAFFFF);
   }
 }
```

如果寫入與讀取Flash內容相同，
Flash記憶體使用正常，
USERLED紅燈恆亮

如果寫入與讀取Flash內容不同，
Flash記憶體使用異常，
USERLED紅燈閃爍

預設定義說明

- #define StartAddr ((u32)0x08008000)
  - 定義Flash使用起始點
  - 使用起始點必需大於0x8003000 + Code Size
    - 附註: 0x8000000~0x8003000為DFU程式區塊，使用此區塊將造成無法燒錄程式
    - Code Size可由產生的HEX檔得知
- #define EndAddr   ((u32)0x0800C000)
  - 定義Flash使用結束點
  - 使用起始點必需小於0x8040000
- Data = 0x15041975;
  - 寫入資料
  - 32Bit
- #define FLASH_PAGE_SIZE   ((u16)0x800)
  - 每一個Page有2KByte

Intel HEX Format

:	**//**	**aaaa**	**tt**	**dd**	**cc**

field	Description
:	the colon that starts every Intel HEX record.
//	the record-length field that represents the number of data bytes (**dd**) in the record.
aaaa	the address field that represents the starting address for subsequent data in the record.
tt	the field that represents the HEX record type, which may be one of the following: **00** - data record **01** - end-of-file record **02** - extended segment address record **04** - extended linear address record
dd	a data field that represents one byte of data. A record may have multiple data bytes. The number of data bytes in the record must match the number specified by the // field.
cc	the checksum field that represents the checksum of the record. The checksum is calculated by summing the values of all hexadecimal digit pairs in the record modulo 256 and taking the two's complement.

HEX Example

```
:020000040800F2
:1030000018140020453100008D9340008CD340008D8
………
:103820004237000800000000000000000000000017
:0C383000000000000000000000000000008C
:040000050800031318D
:00000001FF
```

- 資料填入起始位置為0x08003000
- 每一行有0x10(16)個Bytes
- 最後一筆資料為 :040000050800031318D
- 資料結束點為0x08033830
- (0x08003830 -0x08000000)/0x800 = 7
- 需以0x800為單位，所以使用的位置起點為0x08004000

燒錄MIAT_STM32

- Rebuilder all target files產生HEX
- DFU File Manager轉換HEX產生DFU
- DfuSe Demonstration燒錄DFU
- Leave DFU mode

內部Flash存取控制實驗

實驗一

WU-YANG

Technology Co., Ltd.

實驗一練習

- 注意:
  - 請使用預設0x08008000之後的位置，避免覆蓋DFU與使用者程式碼區塊
- 練習:
  - 修改存取位置測試是否正常
  - 修改寫入資料測試是否正常
  - 修改寫入資料後取消FLASH_Unlock測試是否正常
  - 修改寫入資料後取消FLASH_ErasePage測試是否正常

實驗目的(二)

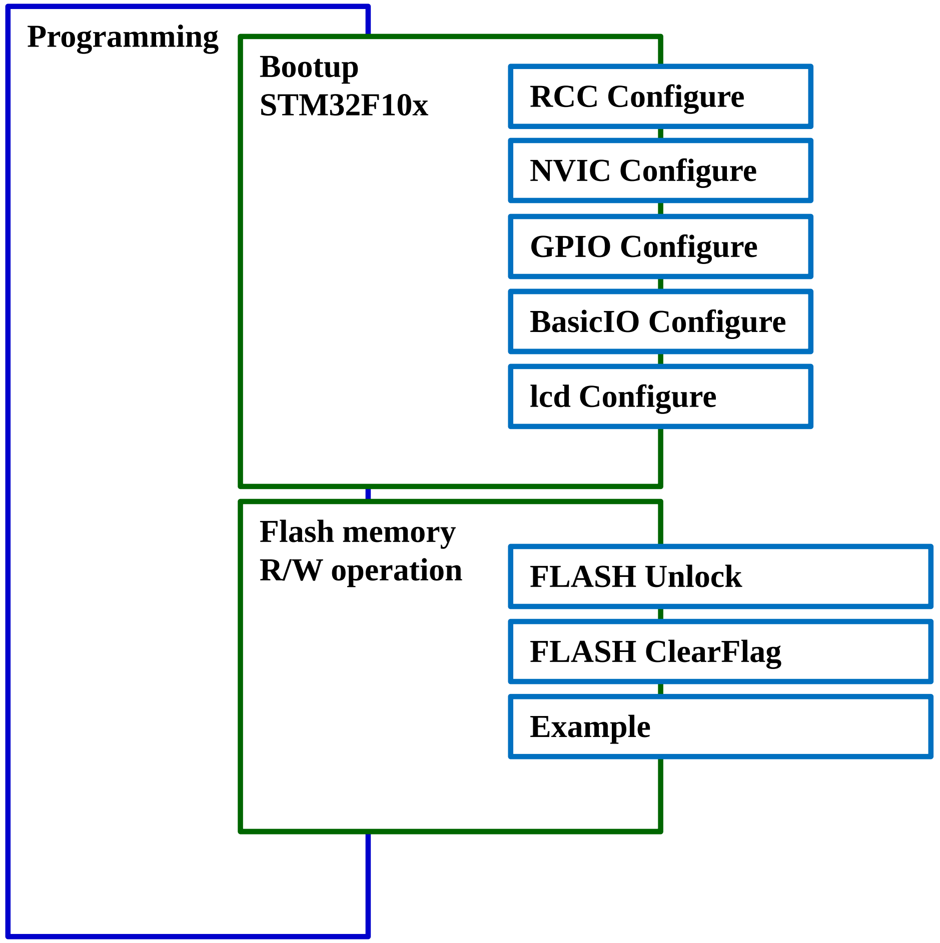
- 使用MIAT_STM32實驗板透過Flash memory interface (FLITF)控制內部Flash進行存取控制實驗，並利用SW、KEY決定資料寫入與讀取，LCD顯示狀態。

實驗原理

- System requirement
  - Embedded Flash
  - LCD
  - KEY
  - SW
- Development Flow
- ARM Configure

Development Flow

Embedded Software Side



Bootup STM32F10x

```
int main(void)
{
#ifdef DEBUG
  debug();
#endif

  FLASHStatus = FLASH_COMPLETE;

/* RCC Configuration */
  RCC_Configuration();

  /* NVIC Configuration */
  NVIC_Configuration();

  /* GPIO Configuration */
  GPIO_Configuration();

  Init_BasicIO();

  lcd_init();                          // LCD Initialization

…………………………………………………………………..
```

硬體電路配置

Mapping Table

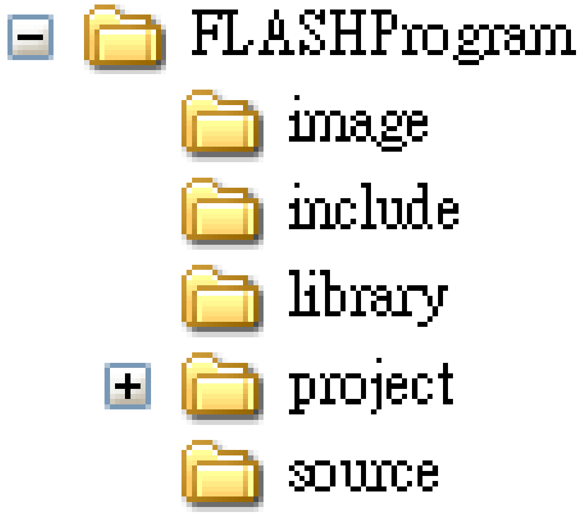
Num.	MIAT_STM32V2	MIAT_IOBV1	Num.	MIAT_STM32V2	MIAT_IOBV1
1	PC8 (3.26)	SW1	10	PE6 (1.5)	LCD_EN
2	PC9 (3.27)	SW2	11	PF6 (1.18)	LCD_R/W
3	PC10 (4.3)	SW3	12	PF7 (1.19)	LCD_RS
4	PC11 (4.4)	SW4	13	PF8(1.20)	LCD_D4
5	PB5 (4.27)	KEY1	14	PF9 (1.21)	LCD_D5
6	PB6 (4.28)	KEY2	15	PF10 (1.22)	LCD_D6
7	PB7 (4.29)	KEY3	16	PF11 (2.13)	LCD_D7
8	PB8 (4.31)	KEY4	17	VDD (2.36)	VCC3.3V
9	VCC5V (1.36)	VCC5V	18	GND (1.35)	GND

實驗步驟

- 範例目錄架構
- 範例說明
- 預設定義說明

範例目錄架構

- 範例目錄
  - 測試映像檔
  - 含括檔
  - 函式庫
  - 專案檔
  - 原始碼



範例說明

Embedded Software Side

Flash memory R/W operation

- **FLASH Unlock**
- **FLASH ClearFlag**
- **Example**

FLASH memory R/W operation

```
lcd_clear();
lcd_print ("MIAT_STM32 DEMO ");

/* Unlock the Flash Program Erase controller */
FLASH_Unlock();


/* Clear All pending flags */
FLASH_ClearFlag(FLASH_FLAG_BSY | FLASH_FLAG_EOP | FLASH_FLAG_PGERR |
FLASH_FLAG_WRPRTERR);

while(1)
{

  KEY_Buffer=Key_Scan();
  set_cursor (0, 1);
  lcd_print ("SW Value = 0x");
  lcd_putchar(((SW&0x7)+0x30));
  if(KEY_Buffer==1)
  {
    Address=(StartAddr|(SW<<12));
    set_cursor (0, 0);
    lcd_print ("Set Addr = 0x");
    lcd_putchar(((SW&0x7)+0x30));
    lcd_print ("  ");
  }
```

掃描KEY是否按下與顯示SW數值，LCD Line2顯示
SW Value = 0x?(SW)

如果KEY1按下，記錄SW數值至Address ，LCD Line1顯示
Set Adde = 0x?(Address)

範例說明

Embedded Software Side

SRAM memory R/W operation

Example

FLASH memory R/W operation

```
else if(KEY_Buffer==2)
   {
     /* Erase the FLASH pages */
     FLASHStatus = FLASH_ErasePage(Address);

     /*  FLASH Word program of data at addresses defined by SW*/
     FLASHStatus = FLASH_ProgramWord(Address, (SW&0x7));

     set_cursor (0, 0);
     lcd_print ("Write 0x");
     lcd_putchar(((SW&0x7)+0x30));
     lcd_print (" at 0x");
     lcd_putchar((((Address>>12)&0x7)+0x30));
   }
   else if(KEY_Buffer==3)
   {
     Data=(u16 *)Address;
     set_cursor (0, 0);
     lcd_print ("Addr 0x");
     lcd_putchar((((Address>>12)&0x7)+0x30));
     lcd_print (" = 0x");
     lcd_putchar(((*Data&0x7)+0x30));
     lcd_print ("  ");
   }

 }
```

如果KEY2按下， 清除Address所處區塊並寫入SW數值 ，LCD Line1顯示
Write 0x?(SW) at 0x?(Adddress)

如果KEY3按下， 讀取Address紀錄數值 ，LCD Line1顯示
Addr 0x?(Adddress) = 0x?

預設定義說明

- #define StartAddr ((u32)0x08020000)
  - 定義Flash使用起始點
  - 使用起始點必需大於0x8003000 + Code Size
    - 附註: 0x8000000~0x8003000為DFU程式區塊，使用此區塊將造成無法燒錄程式
    - Code Size可由產生的HEX檔得知
- u32 Address = 0x08020000;
  - 定義Address初始值
  - 初始值必需與Flash使用起始點相同
- Address=(StartAddr|(SW<<12));
  - Example: StartAddr = 0x08020000 SW = 1 Address = 0x08021000
    StartAddr = 0x08020000 SW = 7 Address = 0x08027000
- StartAddr|(7<<12)必需小於0x8040000

使用者界面與Flash存取控制實驗

實驗二

WU-YANG

Technology Co., Ltd.

實驗二練習

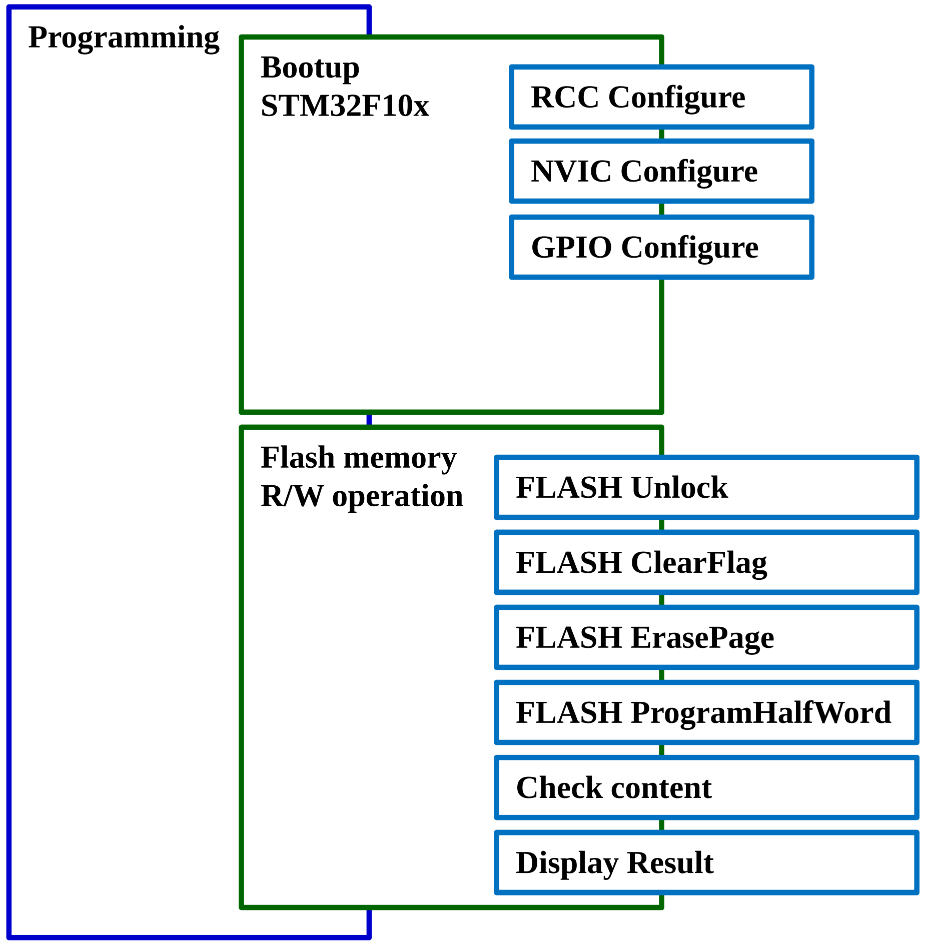
- 注意:
  - 請使用預設0x08020000之後的位置，避免覆蓋DFU與使用者程式碼區塊
- 練習:
  - 利用SW與KEY測試Flash寫入與讀取是否正常
  - Flash寫入後電源關閉，再開啟後寫入資料是否仍存在
  - 修改存取位置測試是否正常

實驗目的(三)

- 使用MIAT_STM32實驗板透過Flash memory interface (FLITF)控制內部Flash進行存取控制實驗，並藉由填入Option Bytes 設定寫入鎖定，保護Page鎖定不被覆蓋。

Development Flow

Embedded Software Side



Bootup STM32F10x

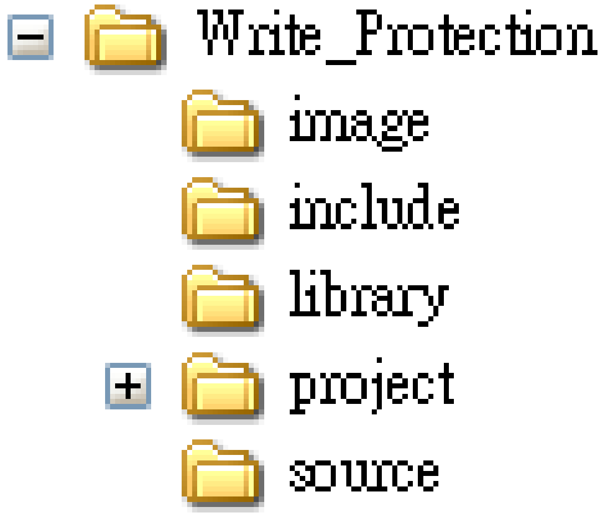
```
int main(void)
{
#ifdef DEBUG
  debug();
#endif

  FLASHStatus = FLASH_COMPLETE;
  MemoryProgramStatus = PASSED;
  Data = 0x1753;

/* RCC Configuration */
  RCC_Configuration();

  /* NVIC Configuration */
  NVIC_Configuration();

  /* GPIO Configuration */
  GPIO_Configuration();

…………………………………………………………
```

實驗步驟

- 範例目錄架構
- 範例說明
- 預設定義說明

範例目錄架構

- 範例目錄
  - 測試映像檔
  - 含括檔
  - 函式庫
  - 專案檔
  - 原始碼



```
⊟ Write_Protection
    image
    include
    library
  ⊞ project
    source
```

範例說明

Flash FwLib Functions List

Function name	Description
FLASH_Unlock	Unlocks the FLASH Program Erase Controller.
FLASH_ClearFlag	Clears the FLASH pending flags.
FLASH_EraseOptionBytes	Erases the FLASH option bytes.
FLASH_EnableWriteProtection	Write protects the desired pages
NVIC_GenerateSystemReset	Generate a system reset.

```
  /* Unlock the Flash Program Erase controller */
  FLASH_Unlock();

  /* Define the number of page to be erased */
  NbrOfPage = (EndAddr - StartAddr) / FLASH_PAGE_SIZE;

  FLASH_ClearFlag(FLASH_FLAG_BSY | FLASH_FLAG_EOP|FLASH_FLAG_PGERR
|FLASH_FLAG_WRPRTERR);

  /* Get pages write protection status */
  WRPR_Value = FLASH_GetWriteProtectionOptionByte();
  ProtectedPages = WRPR_Value & 0x000000C0;

#ifdef WriteProtection_Disable
  if (ProtectedPages == 0x00)
  {/* Pages are write protected */
    /* Disable the write protection */
    FLASHStatus = FLASH_EraseOptionBytes();
    /* Generate System Reset to load the new option byte values */
    NVIC_GenerateSystemReset();
  }
#else
 #ifdef WriteProtection_Enable
  if (ProtectedPages != 0x00)
  {/* Pages not write protected */
    FLASHStatus = FLASH_EraseOptionBytes();
    /* Enable the pages write protection */
    FLASHStatus = FLASH_EnableWriteProtection(FLASH_WRProt_Pages12to13
|FLASH_WRProt_Pages14to15);
    /* Generate System Reset to load the new option byte values */
    NVIC_GenerateSystemReset();
  }
 #endif
#endif
```

範例說明

Flash FwLib Functions List

Function name	Description
FLASH_ErasePage	Erases a specified FLASH page.
FLASH_ProgramHalfWord	Programs a half word at a specified address.

```
 if (ProtectedPages != 0x00)
  {
    /* Clear All pending flags */
    FLASH_ClearFlag(FLASH_FLAG_BSY |
FLASH_FLAG_EOP|FLASH_FLAG_PGERR |FLASH_FLAG_WRPRTERR);

    /* erase the FLASH pages */
    for(EraseCounter = 0; (EraseCounter < NbrOfPage) && (FLASHStatus ==
FLASH_COMPLETE); EraseCounter++)
    {
      FLASHStatus = FLASH_ErasePage(StartAddr + (FLASH_PAGE_SIZE *
EraseCounter));
    }

    /* FLASH Half Word program of data 0x1753 at addresses defined by  StartAddr
and EndAddr */
    Address = StartAddr;

    while((Address < EndAddr) && (FLASHStatus == FLASH_COMPLETE))
    {
      FLASHStatus = FLASH_ProgramHalfWord(Address, Data);
      Address = Address + 2;
    }
```

範例說明

Embedded Software Side

Flash memory
R/W operation

Check content

Display Result

Display Result

```
/* Check the corectness of written data */
  Address = StartAddr;

  while((Address < EndAddr) && (MemoryProgramStatus != FAILED))
  {
    if((*(vu16*) Address) != Data)
    {
      MemoryProgramStatus = FAILED;
    }
    Address += 2;
  }

 while (1)
 {
  if (MemoryProgramStatus == PASSED)
  {  /* OK Turn on USERLED */
    GPIO_SetBits(GPIOF, GPIO_Pin_11);
  }
  else
  { /* KO Turn off USERLED */
    GPIO_ResetBits(GPIOF, GPIO_Pin_11);
    /* Insert delay */
    Delay(0xAFFFF);
    /* Turn on USERLED */
    GPIO_SetBits(GPIOF, GPIO_Pin_11);
    /* Insert delay */
    Delay(0xAFFFF);
  }
 }
```

如果寫入與讀取Flash內容相同，
Flash記憶體可以寫入，
USERLED紅燈恆亮

如果寫入與讀取Flash內容不同，
Flash記憶體不能寫入，
USERLED紅燈閃爍

預設定義說明

- #define StartAddr ((u32) 0x08006000)
  - 定義Flash使用起始點
  - 使用起始點必需大於0x8003000 + Code Size
    - 附註: 0x8000000~0x8003000為DFU程式區塊，使用此區塊將造成無法燒錄程式
    - Code Size可由產生的HEX檔得知
- #define EndAddr   ((u32) 0x08008000)
  - 定義Flash使用結束點
  - 使用起始點必需小於0x8040000
- Data = 0x1753;
  - 寫入資料
  - 16Bit
- #define FLASH_PAGE_SIZE   ((u16)0x800)
  - 每一個Page有2KByte

預設定義說明

- #define WriteProtection_Enable
  - Uncomment this line to Enable Write Protection
- #define WriteProtection_Disable
  - Uncomment this line to Disable Write Protection
- FLASH_EnableWriteProtection
  - 設定影響2個Page
    - FLASH_WRProt_Pages0to1
    - FLASH_WRProt_Pages1to2 ……
    - FLASH_WRProt_Pages60to61
  - 設定影響多個Page
    - FLASH_WRProt_Pages62to255

FLASH_EnableWriteProtection

- Example:

  位置0x08006000 ~ 0x08007000
  每一個Page有0x800Byte
  StartPage = (0x08006000-0x08000000)/0x800 = 12
  EndPage = ((0x08007000-0x08000000)/0x800)-1 = 13
  執行FLASH_EnableWriteProtection(FLASH_WRProt_Pages12to13)
  將影響Pages12、 Pages13

內部Flash寫入鎖定存取控制實驗

實驗三

WU-YANG
Technology Co., Ltd.

實驗三練習

- 注意:
  - 請使用預設0x08006000之後的位置，避免覆蓋DFU與使用者程式碼區塊
  - 更改寫入位置測試寫入鎖定時需同時更改FLASH_EnableWriteProtection鎖定之Page
- 練習:
  - 打開#define WriteProtection_Enable測試是否可寫入
  - 打開#define WriteProtection_Disable測試是否可寫入
  - 修改存取位置測試是否正常
  - 修改寫入資料測試是否正常

Q & A