**CHUYÊN ĐỀ THIẾT KẾ HỆ THỐNG NHÚNG 1**

**BÀI THỰC HÀNH 5: LẬP TRÌNH TRUY CẬP BỘ NHỚ FLASH**

# Mục tiêu

* Giúp sinh viên tìm hiểu và tích lũy kinh nghiệm trong lập trình truy xuất và kiểm soát việc truy xuất trong vùng nhớ của Vi điều khiển.
* Giúp sinh viên làm quen với cách tổ chức và bố trí của bộ nhớ Flash của dòng vi điều khiển ARM-Cortex M4 mà cụ thể là vi điều khiển STM32F405 trong bộ KIT thực hành.

# Chuẩn bị trước

* Tìm hiểu tìm hiểu các thông tin cơ bản về KIT Open405R-C Package A, STM32F4 Development Board [1]
* Sinh viên tìm hiểu về tổ chức và bố trí bộ nhớ theo datasheet của vi điều khiển STM32F405 [2]
* Sinh viên tìm hiểu về tổ chức và bố trí bộ nhớ Flash theo tài liệu của nhà sản xuất ST cho dòng vi điều khiển STM32F40X [3]

# Nội dung và hướng dẫn thực hành

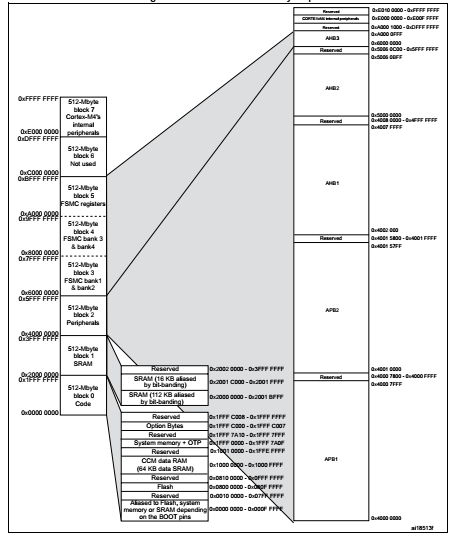
## Tổ chức bộ nhớ của dòng vi điều khiển STM32F40X

Trong mỗi vi điều khiển, nhà sản xuất thường cấp phát một vùng nhớ Flash để người dùng nạp chương trình cho vi điều khiển. Tuy nhiên, trong bài thực hành này, sinh viên sẽ làm quen với việc dùng vùng nhớ này để lưu trữ một số dữ liệu. Lợi ích quan trọng nhất của vùng nhớ Flash là nó có thể lưu trữ dữ liệu ngay cả khi vi điều khiển bị mất nguồn điện và dữ liệu sẽ vẫn được giữ an toàn trong vùng nhớ Flash.

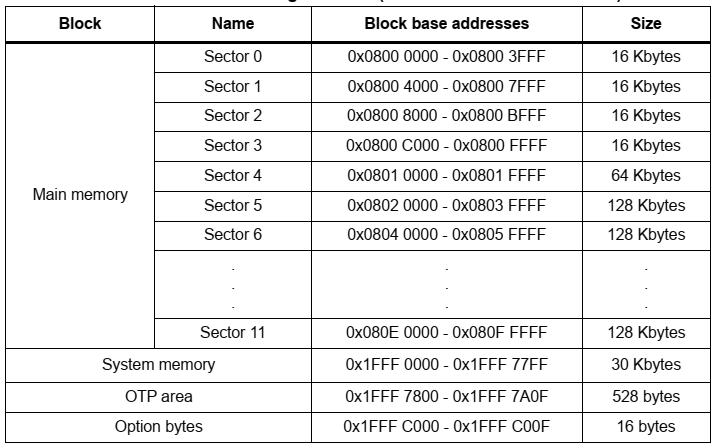
Theo như tài liệu datasheet của dòng vi điều khiển STM32F40X thì bộ nhớ được tổ chức như trong **Hình 1**. Ta có thể nhận thấy rằng, bộ nhớ Flash được bố trí ở vùng nhớ 0x08000000 – 0x080FFFFF và nó nằm trong 512-Mbyte của khối đầu tiên để chứa code. Đối với dòng vi điều khiển STM32F40X thì bộ nhớ Flash có đặc điểm sau:

* Dung lượng lên tới 1Mbyte
* Độ rộng dữ liệu 128 bits
* Có thể truy xuất theo byte, half-word, word và double word
* Có thể xóa theo phân đoạn và toàn bộ

Tổ chức của bộ nhớ Flash trong dòng vi điều khiển STM32F40X được phân chia theo các phân đoạn (sector) như **Hình 2**.



**Hình 1 Sơ đồ tổ chức bộ nhớ của dòng vi điều STM32F40X**



**Hình 2 Sơ đồ tổ chức bộ nhớ FLASH của dòng vi điều khiển STM32F40X**

## Lập trình truy xuất bộ nhớ Flash:

Để hiện thực phần truy xuất bộ nhớ Flash, sinh viên cần hiện thực 2 hàm sau:

* Flash\_Write\_Data: ghi dữ liệu xuống bộ nhớ Flash
* Flash\_Read\_Data: đọc dữ liệu từ bộ nhớ Flash

## Ghi dữ liệu xuống Flash

Hàm Flash\_Write\_Data sẽ ghi dữ liệu xuống vùng nhớ mong muốn. Đầu tiên, vi điều khiển sẽ tính toán vị trí bắt đầu của phân đoạn dựa trên thông số StartSectorAddress. Sau đó, vi điều khiển sẽ xóa vùng nhớ cần ghi và ghi xuống đó dữ liệu mới.

uint32\_t **Flash\_Write\_Data** (uint32\_t StartSectorAddress, uint32\_t \*Data, uint16\_t numberofwords)

{

static FLASH\_EraseInitTypeDef EraseInitStruct;

uint32\_t SECTORError;

int sofar=0;

*/\* Unlock the Flash to enable the flash control register access \*\*\*\*\*\*\*\*\*\*\*\*\*/*

**HAL\_FLASH\_Unlock**();

*/\* Erase the user Flash area \*/*

*/\* Get the number of sector to erase from 1st sector \*/*

uint32\_t StartSector = **GetSector**(StartSectorAddress);

uint32\_t EndSectorAddress = StartSectorAddress + numberofwords\*4;

uint32\_t EndSector = **GetSector**(EndSectorAddress);

*/\* Fill EraseInit structure\*/*

EraseInitStruct.TypeErase = FLASH\_TYPEERASE\_SECTORS;

EraseInitStruct.VoltageRange = FLASH\_VOLTAGE\_RANGE\_3;

EraseInitStruct.Sector = StartSector;

EraseInitStruct.NbSectors = (EndSector - StartSector) + 1;

*/\* Note: If an erase operation in Flash memory also concerns data in the data or instruction cache,*

*you have to make sure that these data are rewritten before they are accessed during code*

*execution. If this cannot be done safely, it is recommended to flush the caches by setting the*

*DCRST and ICRST bits in the FLASH\_CR register. \*/*

if (**HAL\_FLASHEx\_Erase**(&EraseInitStruct, &SECTORError) != HAL\_OK)

{

return **HAL\_FLASH\_GetError** ();

}

*/\* Program the user Flash area word by word*

*(area defined by FLASH\_USER\_START\_ADDR and FLASH\_USER\_END\_ADDR) \*\*\*\*\*\*\*\*\*\*\*/*

while (sofar<numberofwords)

{

if (**HAL\_FLASH\_Program**(FLASH\_TYPEPROGRAM\_WORD, StartSectorAddress, Data[sofar]) == HAL\_OK)

{

StartSectorAddress += 4; *// use StartPageAddress += 2 for half word and 8 for double word*

sofar++;

}

else

{

*/\* Error occurred while writing data in Flash memory\*/*

return **HAL\_FLASH\_GetError** ();

}

}

*/\* Lock the Flash to disable the flash control register access (recommended*

*to protect the FLASH memory against possible unwanted operation) \*\*\*\*\*\*\*\*\*/*

**HAL\_FLASH\_Lock**();

return 0;

}

## Đọc dữ liệu từ Flash

Hàm Flash\_Read\_Data sẽ đọc khối lượng dữ liệu mong muốn từ vùng nhớ được yêu cầu truy xuất và lưu nó vào một Buffer.

void **Flash\_Read\_Data** (uint32\_t StartSectorAddress, uint32\_t \*RxBuf, uint16\_t numberofwords)

{

while (1)

{

\*RxBuf = \*(\_\_IO uint32\_t \*)StartSectorAddress;

StartSectorAddress += 4;

RxBuf++;

if (!(numberofwords--)) break;

}

}

# Bài tập thực hành

Sinh viên ghi xuống vùng nhớ Flash tại vị trí bắt đầu của Sector11 100 dòng dữ liệu “Hello, We are access Flash Memory in the course CE437\r\n”. Sau đó, sinh viên kiểm tra lại dữ liệu bằng cách truy xuất toàn bộ vùng nhớ đã ghi và in toàn bộ dữ liệu ra UART.

# Báo cáo thực hành

Sinh viên viết báo cáo theo mẫu đã được cung cấp để trình bày lại các kết quả đã làm được trong bài thực hành. Giải thích các đoạn chương trình mà sinh viên đã hiện thực để giải quyết các yêu cầu của bài thực hành.

# Tài liệu tham khảo

|  |  |
| --- | --- |
| [1] | “Open405R-C Package A, STM32F4 Development Board,” Waveshare Electronics, 2022. [Trực tuyến]. Available: https://www.waveshare.com/open405r-c-package-a.htm. |
| [2] | STMicroelectronics, [Trực tuyến]. Available: https://www.st.com/resource/en/datasheet/dm00037051.pdf. [Đã truy cập 2022]. |
| [3] | “ST,” STMicroelectronics, [Trực tuyến]. Available: https://www.st.com/resource/en/reference\_manual/dm00031020-stm32f405-415-stm32f407-417-stm32f427-437-and-stm32f429-439-advanced-arm-based-32-bit-mcus-stmicroelectronics.pdf. [Đã truy cập 2022]. |
| [4] | Waveshare Electronics, [Trực tuyến]. Available: https://www.waveshare.com/w/upload/f/fe/OpenX05R-C-Schematic.pdf. [Đã truy cập 2022]. |