BÁO CÁO THỰC HÀNH BÀI 5

Môn học: **CHUYÊN ĐỀ THIẾT KẾ HỆ THỐNG NHÚNG 1**- Mã lớp: **CE437.N11**
Giảng viên hướng dẫn thực hành: Phạm Minh Quân

	Mã số sinh viên	**Họ và tên sinh viên**
Thông tin các sinh viên	19520887	Phạm Trung Quốc
	19521651	Phạm Trọng Huỳnh
	19520928	Viên Minh Tân
	19520036	Phạm Quốc Đăng
Link các tài liệu tham khảo		
Đánh giá của giảng viên:		

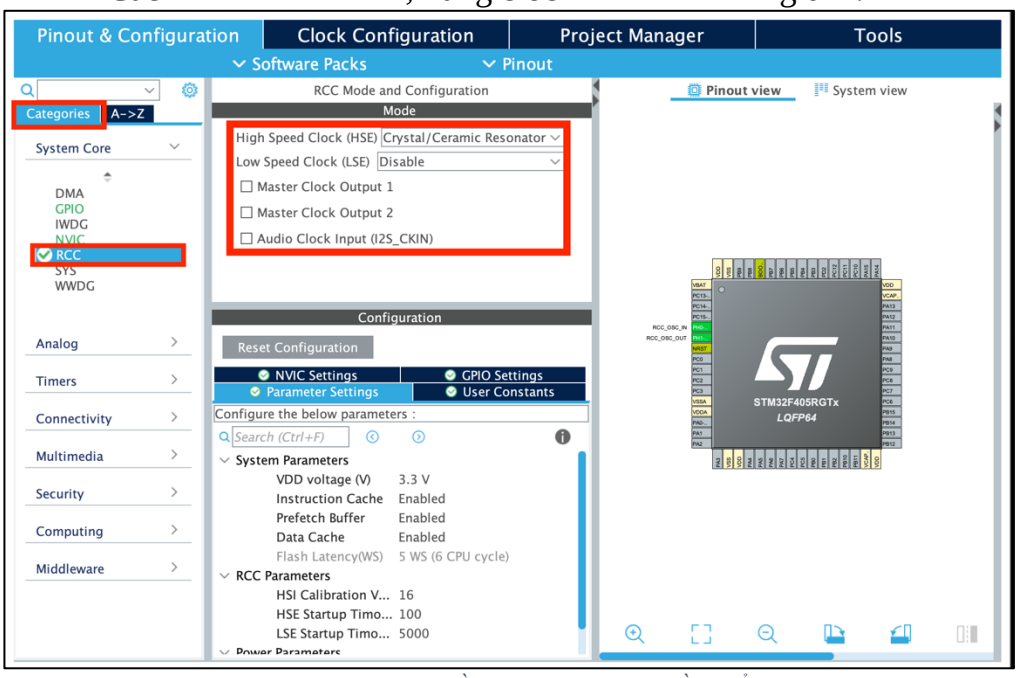
Mục lục

Yêu cầu: Lập trình truy cập bộ nhớ Flash

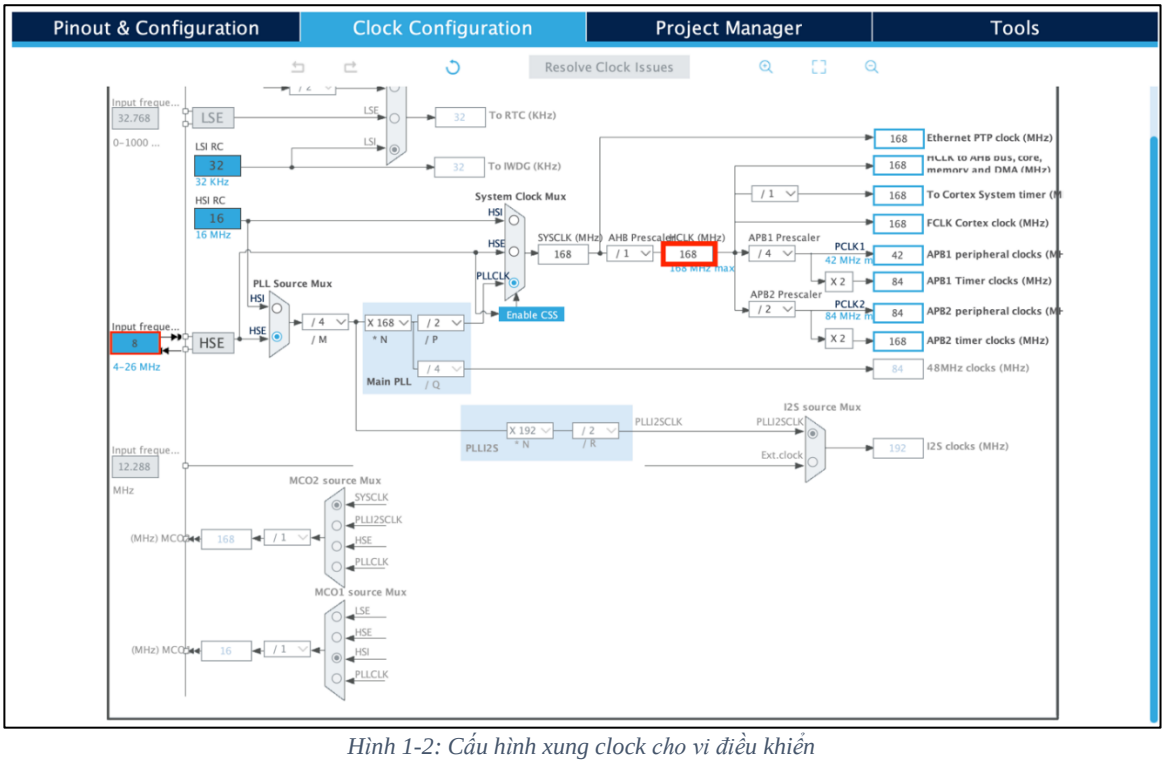
- Sinh viên ghi xuống vùng nhớ Flash tại vị trí bắt đầu của Sector11 100 dòng dữ liệu "Hello, We are access Flash Memory in the course CE437\r\n".
- Sau đó, sinh viên kiểm tra lại dữ liệu bằng cách truy xuất toàn bộ vùng nhớ đã ghi và in toàn bộ dữ liệu ra UART.

1. Thiết lập các cấu hình liên quan

- Các cấu hình ban đầu, xung clock làm như hướng dẫn.

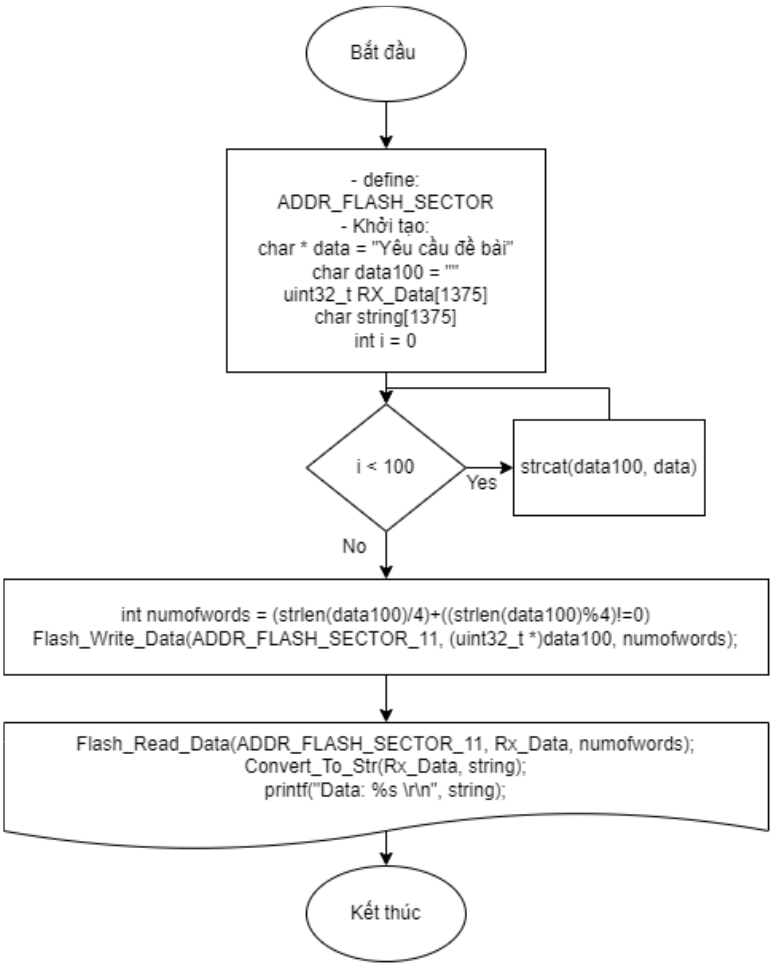


Hình 1-1: Chọn nguồn xung clock cho vi điều khiển



Hình 1-2: Cấu hình xung clock cho vi điều khiển

2. Sơ đồ khối



Hình 2-1: Sơ đồ khối của Yêu cầu lab 5

3. Mã nguồn và giải thích

```
#define ADDR_FLASH_SECTOR_0     ((uint32_t)0x08000000) /* Base @ of Sector 0,
16 Kbytes */
#define ADDR_FLASH_SECTOR_1     ((uint32_t)0x08004000) /* Base @ of Sector 1,
16 Kbytes */
#define ADDR_FLASH_SECTOR_2     ((uint32_t)0x08008000) /* Base @ of Sector 2,
16 Kbytes */
#define ADDR_FLASH_SECTOR_3     ((uint32_t)0x0800C000) /* Base @ of Sector 3,
16 Kbytes */
#define ADDR_FLASH_SECTOR_4     ((uint32_t)0x08010000) /* Base @ of Sector 4,
64 Kbytes */
#define ADDR_FLASH_SECTOR_5     ((uint32_t)0x08020000) /* Base @ of Sector 5,
128 Kbytes */
#define ADDR_FLASH_SECTOR_6     ((uint32_t)0x08040000) /* Base @ of Sector 6,
128 Kbytes */
#define ADDR_FLASH_SECTOR_7     ((uint32_t)0x08060000) /* Base @ of Sector 7,
128 Kbytes */
#define ADDR_FLASH_SECTOR_8     ((uint32_t)0x08080000) /* Base @ of Sector 8,
128 Kbytes */
#define ADDR_FLASH_SECTOR_9     ((uint32_t)0x080A0000) /* Base @ of Sector 9,
128 Kbytes */
#define ADDR_FLASH_SECTOR_10    ((uint32_t)0x080C0000) /* Base @ of Sector 10,
128 Kbytes */
```

```
#define ADDR_FLASH_SECTOR_11    ((uint32_t)0x080E0000) /* Base @ of Sector 11,
128 Kbytes */

#define FLASH_Sector_0     ((uint16_t)0x0000) /*!< Sector Number 0 */
#define FLASH_Sector_1     ((uint16_t)0x0008) /*!< Sector Number 1 */
#define FLASH_Sector_2     ((uint16_t)0x0010) /*!< Sector Number 2 */
#define FLASH_Sector_3     ((uint16_t)0x0018) /*!< Sector Number 3 */
#define FLASH_Sector_4     ((uint16_t)0x0020) /*!< Sector Number 4 */
#define FLASH_Sector_5     ((uint16_t)0x0028) /*!< Sector Number 5 */
#define FLASH_Sector_6     ((uint16_t)0x0030) /*!< Sector Number 6 */
#define FLASH_Sector_7     ((uint16_t)0x0038) /*!< Sector Number 7 */
#define FLASH_Sector_8     ((uint16_t)0x0040) /*!< Sector Number 8 */
#define FLASH_Sector_9     ((uint16_t)0x0048) /*!< Sector Number 9 */
#define FLASH_Sector_10    ((uint16_t)0x0050) /*!< Sector Number 10 */
#define FLASH_Sector_11    ((uint16_t)0x0058) /*!< Sector Number 11 */
```

Định nghĩa địa chỉ cho các Flash Sector

```
static uint32_t GetSector(uint32_t Address)
{
  uint32_t sector = 0;

  if((Address < ADDR_FLASH_SECTOR_1) && (Address >= ADDR_FLASH_SECTOR_0))
  {
    sector = FLASH_SECTOR_0;
  }
  else if((Address < ADDR_FLASH_SECTOR_2) && (Address >= ADDR_FLASH_SECTOR_1))
  {
    sector = FLASH_SECTOR_1;
  }
  else if((Address < ADDR_FLASH_SECTOR_3) && (Address >= ADDR_FLASH_SECTOR_2))
  {
    sector = FLASH_SECTOR_2;
  }
  else if((Address < ADDR_FLASH_SECTOR_4) && (Address >= ADDR_FLASH_SECTOR_3))
  {
    sector = FLASH_SECTOR_3;
  }
  else if((Address < ADDR_FLASH_SECTOR_5) && (Address >= ADDR_FLASH_SECTOR_4))
  {
    sector = FLASH_SECTOR_4;
  }
  else if((Address < ADDR_FLASH_SECTOR_6) && (Address >= ADDR_FLASH_SECTOR_5))
  {
    sector = FLASH_SECTOR_5;
  }
  else if((Address < ADDR_FLASH_SECTOR_7) && (Address >= ADDR_FLASH_SECTOR_6))
  {
    sector = FLASH_SECTOR_6;
  }
  else if((Address < ADDR_FLASH_SECTOR_8) && (Address >= ADDR_FLASH_SECTOR_7))
  {
    sector = FLASH_SECTOR_7;
  }
  else if((Address < ADDR_FLASH_SECTOR_9) && (Address >= ADDR_FLASH_SECTOR_8))
  {
    sector = FLASH_SECTOR_8;
  }
  else if((Address < ADDR_FLASH_SECTOR_10) && (Address >=
ADDR_FLASH_SECTOR_9))
  {
```

```
    sector = FLASH_SECTOR_9;
  }
  else if((Address < ADDR_FLASH_SECTOR_11) && (Address >=
ADDR_FLASH_SECTOR_10))
  {
    sector = FLASH_SECTOR_10;
  }
  else /*(Address < FLASH_END_ADDR) && (Address >= ADDR_FLASH_SECTOR_11))*/
  {
    sector = FLASH_SECTOR_11;
  }
  return sector;
}
```

Hàm GetSector trả về Flash Sector tương ứng với Address đầu vào của hàm

```
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;
}
```

Hàm Flash_Write_Data: ghi dữ liệu của biến *Data* vào flash memory tại địa chỉ *StartSectorAddress* với số lượng word là *numberofwords*.

```
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;
       }
}
```

Hàm Flash_Read_Data: đọc dữ liệu từ flash memory tại địa chỉ *StartSectorAddress* với số lượng word là *numberofwords* vào biến *RxBuf*.

```
void Convert_To_Str (uint32_t *Data, char *Buf)
{
       int numberofbytes = ((strlen((char *)Data)/4) + ((strlen((char *)Data) %
4) != 0)) *4;

       for (int i=0; i<numberofbytes; i++)
       {
              Buf[i] = Data[i/4]>>(8*(i%4));
       }
}
```

Hàm Convert_To_Str dùng chuyển dữ liệu từ uint32_t sang char

```
  HAL_Init();
  SystemClock_Config();
  MX_GPIO_Init();
  MX_USART1_UART_Init();
```

Gọi các hàm khởi tạo cần thiết cho hệ thống

```
char *data = "Hello, We are access Flash Memory in the course CE437\r\n";
char data100[5500] = "";
for (int i = 0; i < 100; i++)
{
	strcat(data100, data);
}

uint32_t Rx_Data[1375];
char string[1375];

int numofwords = (strlen(data100)/4)+((strlen(data100)%4)!=0);

Flash_Write_Data(ADDR_FLASH_SECTOR_11, (uint32_t *)data100, numofwords);
Flash_Read_Data(ADDR_FLASH_SECTOR_11, Rx_Data, numofwords);
Convert_To_Str(Rx_Data, string);
printf("Data: %s \r\n", string);

printf("Done...  \r\n");
```

Phần xử lý trong main:

- Khởi tạo data = `"Hello, We are access Flash Memory in the course CE437\r\n"`
- Dùng strcat để gán 100 dòng dữ liệu của biến data vào biến data100
- Khởi tạo biến Rx_Data dùng để lưu dữ liệu khi đọc dữ liệu từ flash memory, string dùng để lưu dữ liệu khi chuyển Rx_Data về dạng chuỗi
- Khởi tạo biến numofwords là số lượng word trong data100
- Gọi hàm Flash_Write_Data để ghi data100 vào flash memory ở ADDR_FLASH_SECTOR_11 với số lượng word được ghi là numofwords
- Gọi hàm Flash_Read_Data để đọc dữ liệu từ flash memory ở ADDR_FLASH_SECTOR_11 với số lượng word được đọc là numofword vào biến Rx_Data
- Dùng hàm Convert_To_Str để chuyển dữ liệu của biến Rx_Data về dạng chuỗi và lưu vào biến string
- Dùng printf để ghi dữa liệu vào UART

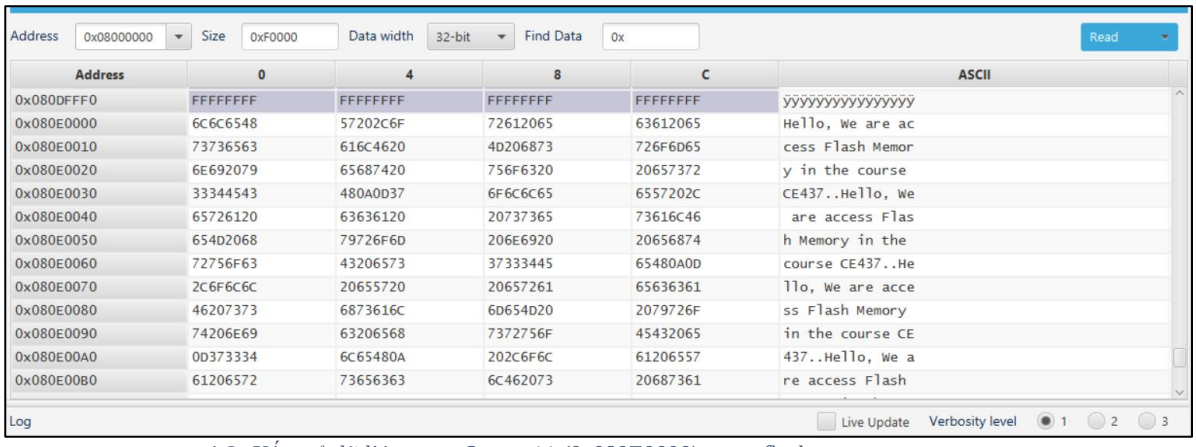
4. Kết quả

```
11:48:24.139 -> Data: Hello, We are access Flash Memory in the course CE437
11:48:24.139 -> Hello, We are access Flash Memory in the course CE437
11:48:24.139 -> Hello, We are access Flash Memory in the course CE437
11:48:24.139 -> Hello, We are access Flash Memory in the course CE437
11:48:24.172 -> Hello, We are access Flash Memory in the course CE437
11:48:24.172 -> Hello, We are access Flash Memory in the course CE437
11:48:24.172 -> Hello, We are access Flash Memory in the course CE437
11:48:24.172 -> Hello, We are access Flash Memory in the course CE437
11:48:24.172 -> Hello, We are access Flash Memory in the course CE437
11:48:24.172 -> Hello, We are access Flash Memory in the course CE437
11:48:24.205 -> Hello, We are access Flash Memory in the course CE437
11:48:24.205 -> Hello, We are access Flash Memory in the course CE437
11:48:24.205 -> Hello, We are access Flash Memory in the course CE437
11:48:24.205 -> Hello, We are access Flash Memory in the course CE437
11:48:24.205 -> Hello, We are access Flash Memory in the course CE437
11:48:24.205 -> Hello, We are access Flash Memory in the course CE437
11:48:24.238 -> Hello, We are access Flash Memory in the course CE437
11:48:24.238 -> Hello, We are access Flash Memory in the course CE437
11:48:24.238 -> Hello, We are access Flash Memory in the course CE437
11:48:24.238 -> Hello, We are access Flash Memory in the course CE437
11:48:24.238 -> Hello, We are access Flash Memory in the course CE437
11:48:24.238 -> Hello, We are access Flash Memory in the course CE437
11:48:24.238 -> Hello, We are access Flash Memory in the course CE437
11:48:24.271 -> Hello, We are access Flash Memory in the course CE437
11:48:24.271 -> Hello, We are access Flash Memory in the course CE437
11:48:24.271 -> Hello, We are access Flash Memory in the course CE437
11:48:24.271 -> Hello, We are access Flash Memory in the course CE437
11:48:24.271 -> Hello, We are access Flash Memory in the course CE437
11:48:24.271 -> Hello, We are access Flash Memory in the course CE437
11:48:24.304 -> Hello, We are access Flash Memory in the course CE437
11:48:24.304 -> Hello, We are access Flash Memory in the course CE437
11:48:24.304 -> Hello, We are access Flash Memory in the course CE437
```

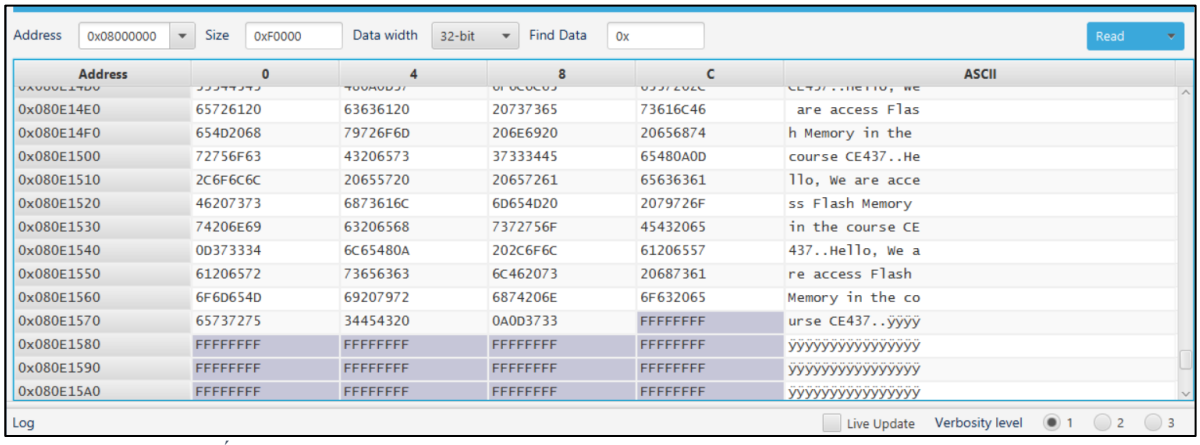
4-1: Kết quả khi bắt đầu in vào UART

```
11:48:24.999 -> Hello, We are access Flash Memory in the course CE437
11:48:24.999 -> Hello, We are access Flash Memory in the course CE437
11:48:25.032 -> Hello, We are access Flash Memory in the course CE437
11:48:25.032 -> Hello, We are access Flash Memory in the course CE437
11:48:25.032 -> Hello, We are access Flash Memory in the course CE437
11:48:25.032 -> Hello, We are access Flash Memory in the course CE437
11:48:25.032 -> Hello, We are access Flash Memory in the course CE437
11:48:25.032 -> Hello, We are access Flash Memory in the course CE437
11:48:25.032 -> Hello, We are access Flash Memory in the course CE437
11:48:25.066 -> Hello, We are access Flash Memory in the course CE437
11:48:25.066 -> Hello, We are access Flash Memory in the course CE437
11:48:25.066 -> Hello, We are access Flash Memory in the course CE437
11:48:25.066 -> Hello, We are access Flash Memory in the course CE437
11:48:25.066 -> Hello, We are access Flash Memory in the course CE437
11:48:25.066 -> Hello, We are access Flash Memory in the course CE437
11:48:25.099 -> Hello, We are access Flash Memory in the course CE437
11:48:25.099 -> Hello, We are access Flash Memory in the course CE437
11:48:25.099 -> Hello, We are access Flash Memory in the course CE437
11:48:25.099 -> Hello, We are access Flash Memory in the course CE437
11:48:25.099 -> Hello, We are access Flash Memory in the course CE437
11:48:25.099 -> Hello, We are access Flash Memory in the course CE437
11:48:25.132 -> Hello, We are access Flash Memory in the course CE437
11:48:25.132 -> Hello, We are access Flash Memory in the course CE437
11:48:25.132 -> Hello, We are access Flash Memory in the course CE437
11:48:25.132 -> Hello, We are access Flash Memory in the course CE437
11:48:25.132 -> Hello, We are access Flash Memory in the course CE437
11:48:25.132 -> Hello, We are access Flash Memory in the course CE437
11:48:25.132 -> Hello, We are access Flash Memory in the course CE437
11:48:25.165 -> Hello, We are access Flash Memory in the course CE437
11:48:25.165 -> Hello, We are access Flash Memory in the course CE437
11:48:25.165 -> Hello, We are access Flash Memory in the course CE437
11:48:25.165 -> Hello, We are access Flash Memory in the course CE437
11:48:25.165 -> Hello, We are access Flash Memory in the course CE437
11:48:25.165 -> Hello, We are access Flash Memory in the course CE437
11:48:25.198 -> Hello, We are access Flash Memory in the course CE437
11:48:25.198 -> _
11:48:25.198 -> Done...
```

4-2: In Done sau khi in hết 100 dòng vào UART



4-3: Kết quả dữ liệu trong Sector 11 (0x080E0000) trong flash memory



4-4: Kết thúc 100 dòng dữ liệu trong Sector 11 (0x080E0000) trong flash memory