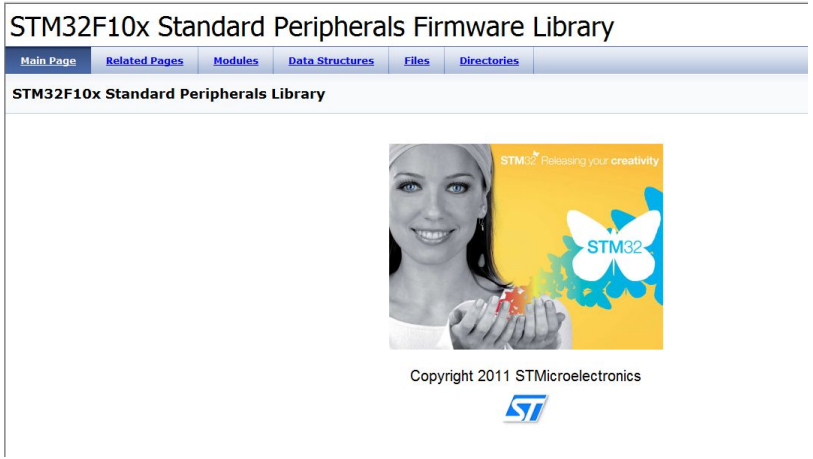
Bài 2: GPIO

Võ Thành Danh

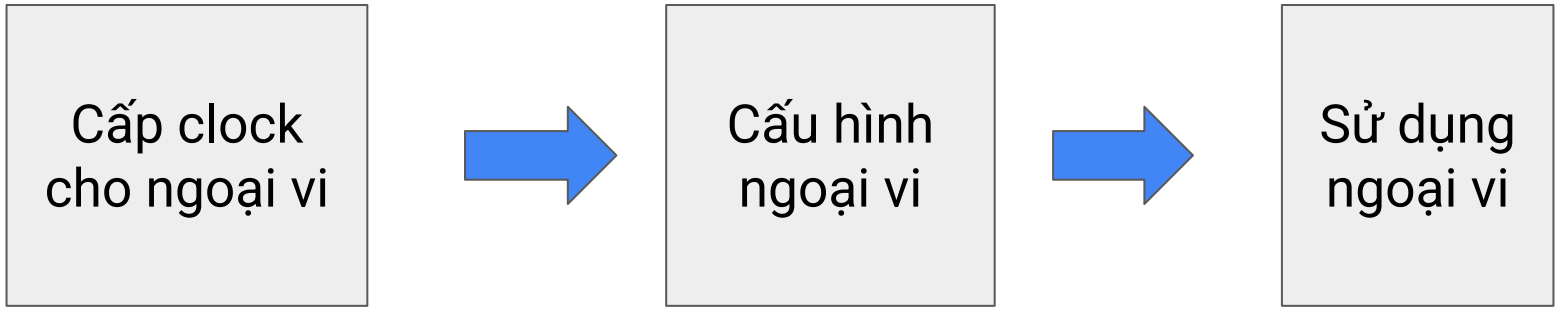
1. Thư viện STM32F10x Standard Peripherals Firmware Library

Là 1 thư viện hoàn chỉnh được phát triển cho dòng STM32. Bao gồm đầy đủ driver cho tất cả các ngoại vi tiêu chuẩn.

Thư viện này bao gồm các hàm, cấu trúc dữ liệu và macro của các tính năng thiết bị ngoại vi STM32.



2. Cấu hình và sử dụng ngoại vi (GPIO)



Thư viện SPL cung cấp các hàm và các định nghĩa giúp việc cấu hình và sử dụng ngoại vi dễ dàng và rõ ràng.

2.1 Cấp clock cho ngoại vi

Module RCC cung cấp các hàm để cấu hình xung clock.

`RCC_APB1PeriphClockCmd`

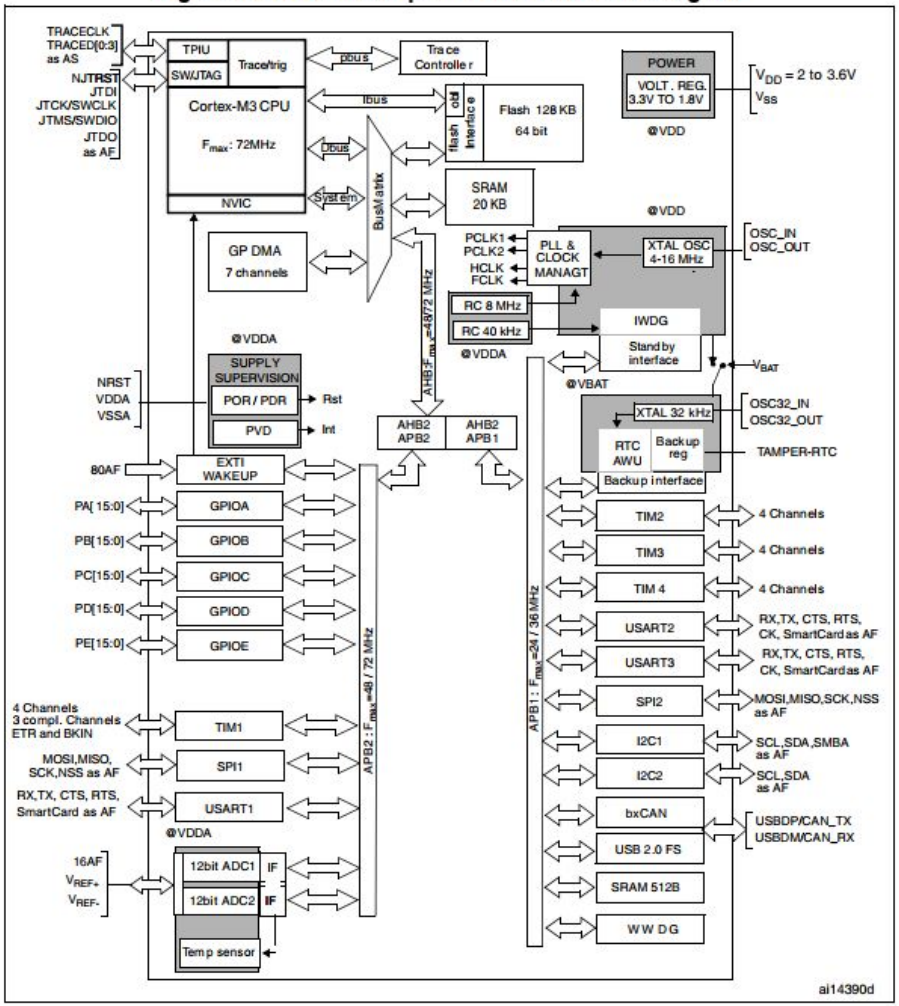
`RCC_APB2PeriphClockCmd`

`RCC_AHBPeriphClockCmd`

Hàm này nhận 2 tham số:

- Ngoại vi muốn cấp clock
- Cho phép (ENABLE) hoặc không cho phép (DISABLE) cấp clcok cho ngoại vi

Figure 1. STM32F103xx performance line block diagram



2.2 Cấu hình ngoại vi

Các tham số dùng để cấu hình cho GPIO được tổ chức trong 1 struct

```
typedef struct
{
  uint16_t GPIO_Pin;              / *!< Specifies the GPIO pins to be configured.
                                     This parameter can be any value of @ref
GPIO_pins_define */

  GPIOSpeed_TypeDef GPIO_Speed;   /*!< Specifies the speed for the selected pins.
                                     This parameter can be a value of @ref
GPIOSpeed_TypeDef */

  GPIOMode_TypeDef GPIO_Mode;     /*!< Specifies the operating mode for the
selected pins.
                                      This parameter can be a value of @ref
GPIOMode_TypeDef */
}GPIO_InitTypeDef;
```

2.2 Cấu hình ngoại vi

```
void GPIO_Config(){
    GPIO_InitTypeDef GPIO_InitStruct;

    GPIO_InitStruct.GPIO_Pin = GPIO_Pin_13;
    GPIO_InitStruct.GPIO_Mode = GPIO_Mode_Out_PP;
    GPIO_InitStruct.GPIO_Speed = GPIO_Speed_50MHz;

    GPIO_Init(GPIOC, &GPIO_InitStruct); // Lưu các cài đặt vào thanh ghi
}
```

2.3 Sử dụng ngoại vi

Các hàm thông dụng:

```
uint8_t GPIO_ReadInputDataBit(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
\\Đọc giá trị 1 chân trong GPIO được cấu hình là INPUT
uint16_t GPIO_ReadInputData(GPIO_TypeDef* GPIOx);
\\Đọc giá trị nguyên GPIO được cấu hình là INPUT
uint8_t GPIO_ReadOutputDataBit(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
\\Đọc giá trị 1 chân trong GPIO được cấu hình là OUTPUT
uint16_t GPIO_ReadOutputData(GPIO_TypeDef* GPIOx);
\\Đọc giá trị nguyên GPIO được cấu hình là OUTPUT
void GPIO_SetBits(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
\\Cho giá trị điện áp của 1 chân trong GPIO = 1
void GPIO_ResetBits(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin);
\\Cho giá trị điện áp của 1 chân trong GPIO = 0
void GPIO_WriteBit(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin, BitAction BitVal);
\\Ghi giá trị "BitVal" vào 1 chân trong GPIO
void GPIO_Write(GPIO_TypeDef* GPIOx, uint16_t PortVal);
\\Ghi giá trị "PortVal" vào nguyên GPIO
```

3.1 Blink LED PC13

```
while(1){
    GPIO_SetBits(GPIOC, GPIO_Pin_13); // Ghi 1 ra PC13
    delay(10000000);
    GPIO_ResetBits(GPIOC, GPIO_Pin_13);// Ghi 0 ra PC13
    delay(10000000);
}
```

3.2 Nháy đuổi

```
void chaseLed(uint8_t loop){
    uint16_t Ledval;
    for(int j = 0; j < loop; j++)
    {
        Ledval = 0x0010; //0b0 0001 0000
        for(int i = 0; i < 4; i++)
        {
            Ledval = Ledval << 1;
            GPIO_Write(GPIOC, Ledval);
            delay(1000000);
        }
    }
}
```

3.3 Đọc trạng thái nút nhấn

```
    // Cấu hình
    GPIO_InitTypeDef
GPIO_InitStruct;
    GPIO_InitStruct.GPIO_Pin =
GPIO_Pin_0;
    GPIO_InitStruct.GPIO_Mode =
GPIO_Mode_IPU;
    GPIO_InitStruct.GPIO_Speed =
GPIO_Speed_50MHz;

    GPIO_Init(GPIOA,
&GPIO_InitStruct);
```

```
// Điều khiển
if(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_0)
== 0){
     while(GPIO_ReadInputDataBit(GPIOA,
GPIO_Pin_0) == 0);
     if(GPIO_ReadOutputDataBit(GPIOC,
GPIO_Pin_13)){
          GPIO_ResetBits(GPIOC,
     GPIO_Pin_13);
     } else {
          GPIO_SetBits(GPIOC, GPIO_Pin_13);
     }
}
```