Mục lục

Bài TH # 2:
Dùng ngắt của SysTick Timer để đảo trạng thái của LEDs sau đúng 1 giây.

LÝ THUYẾT

NGẮT

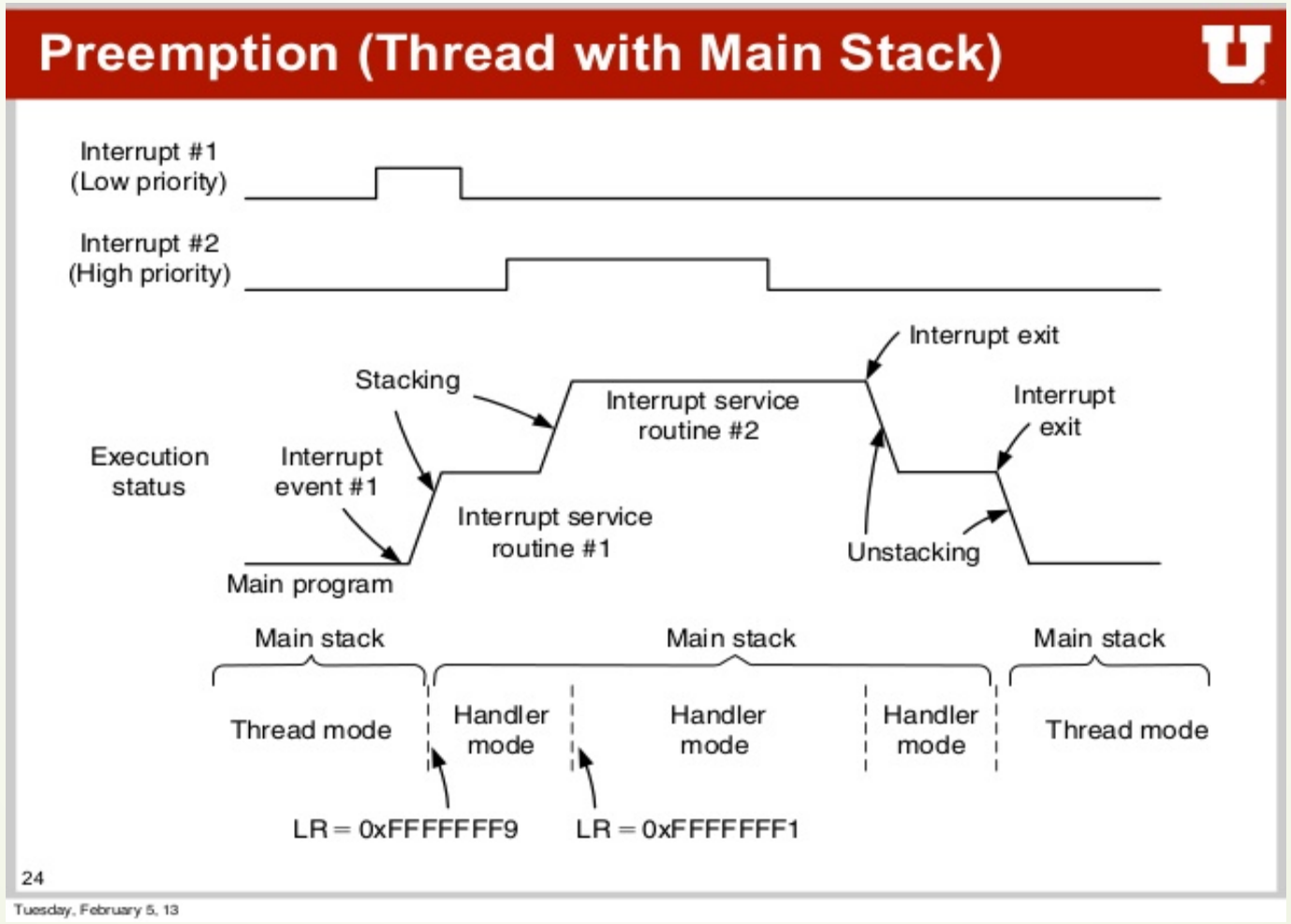
Giới Thiệu Về Ngắt (Interrupt)

- Quá trình dừng chương trình chính đang chạy để ưu tiên thực hiện một chương trình khác
- Trong Cortex-M3, ngắt được quản lý bởi bộ điều khiển vectơ ngắt lồng nhau (NVIC- Nested Vectored Interrupt Controller)
  - Tối đa 256 ngắt, mỗi ngắt có số hiệu đi kèm
  - 16 ngắt hệ thống: định nghĩa bởi ARM
  - 240 ngắt ngoại vi: định nghĩa bởi nhà sản xuất chip

LÝ THUYẾT

NGẮT

- Ngắt được phục vụ dựa trên mức độ ưu tiên.



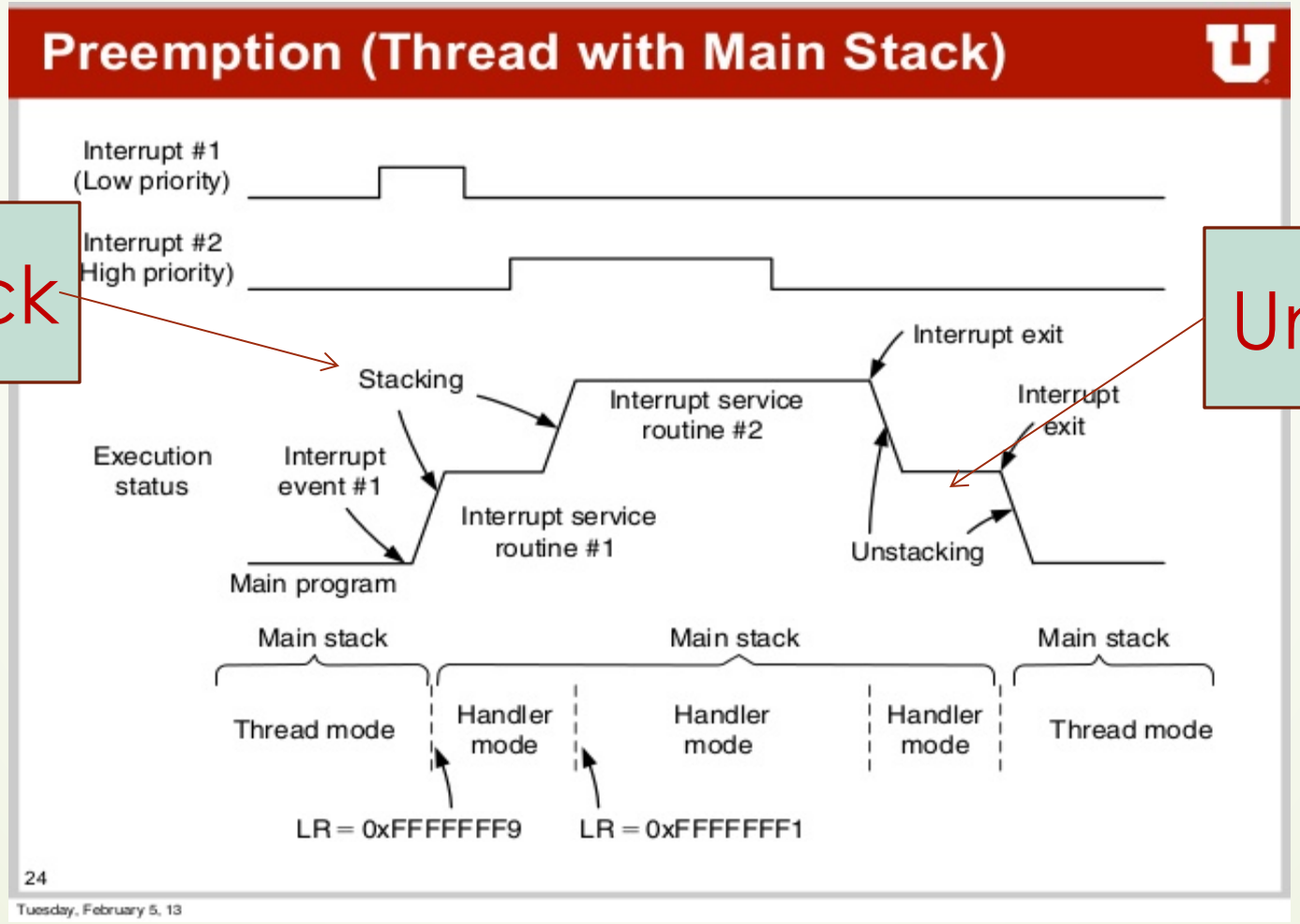
LÝ THUYẾT

NGẮT

- Khi đang thực hiện một ngắt, vi điều khiển Cortex-M3 thực hiện tự động stacking và unstacking.

Stack

Unstack



LÝ THUYẾT

NGẮT

Bộ điều khiển vectơ ngắt lồng nhau (NVIC - Nested Vectored Interrupt Controller)

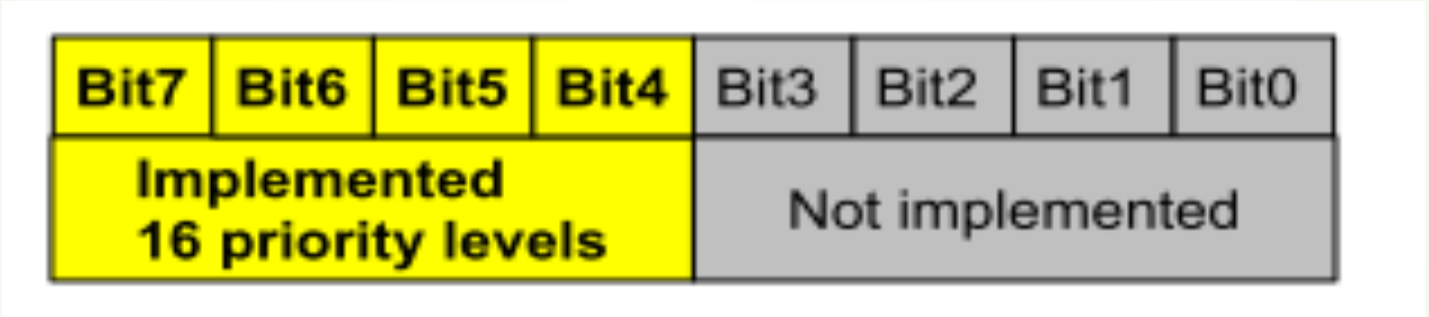
3 chức năng chính

- Kích hoạt và vô hiệu hóa ngắt
- Thiết lập các mức ưu tiên chính và phụ của một ngắt
- Thiết lập và xóa các bit xử lý của một ngắt cụ thể

LÝ THUYẾT

NGẮT

Mức ưu tiên ngắt (Interrupt Priority)



Byte thiết lập mức ưu tiên ngắt

SCB->SHP[(IRQn & 0xF)-4]=(priority<<4) & 0xFF;

SHP(System Handler Priority register) : thanh ghi ưu tiên xử lý hệ thống, SHP được định nghĩa như một mảng byte.
Trong Cortex-M3 có 3 thanh ghi SHP.

LÝ THUYẾT

NGẮT

SYSTICK

SysTick là gì?

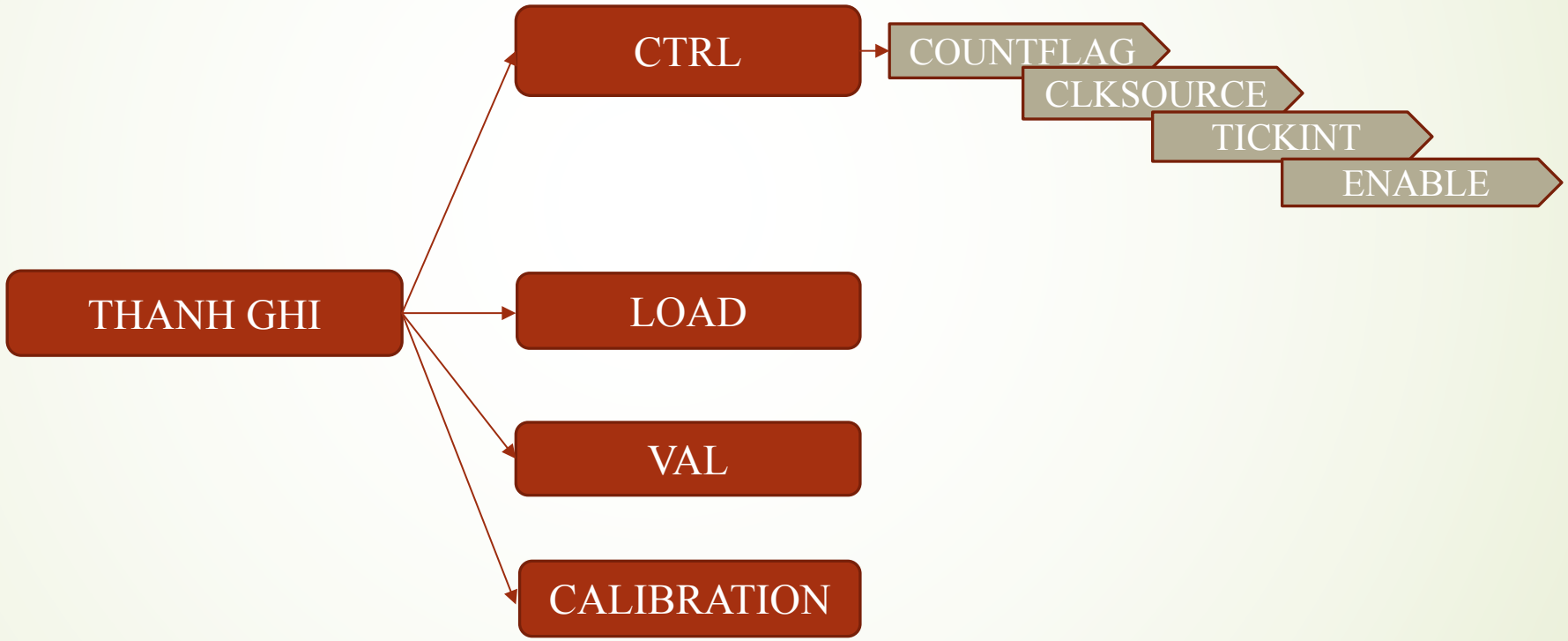
- Là bộ đếm (counter) xuống 24 bit.
  - Mỗi xung nhịp của SysTick thì đếm được 1 xung
- Đếm từ giá trị nạp lại về 0.
- Giá trị đếm tự động nạp lại.
- Có thể tạo ra ngắt khi bộ đếm đến về 0.

LÝ THUYẾT

NGẮT

SYSTICK

Các thanh ghi của SysTick



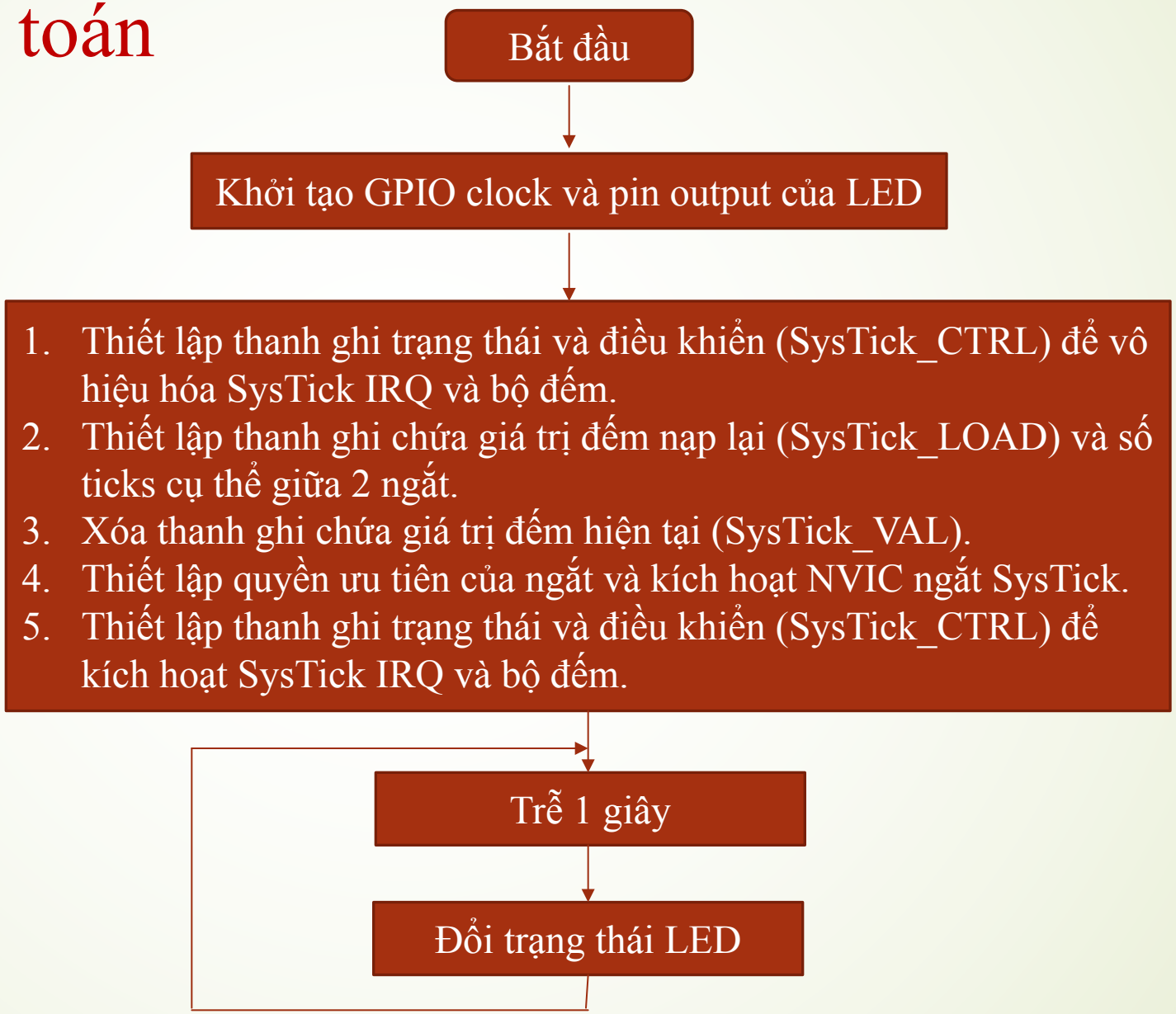
LÝ THUYẾT

NGẮT

SYSTICK

THUẬT TOÁN

Thuật toán



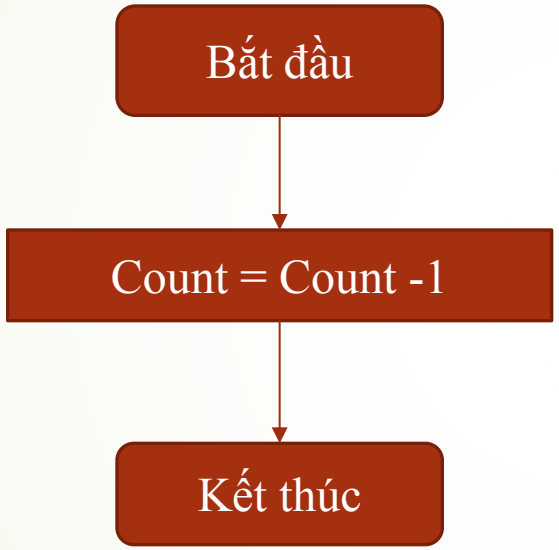
LÝ THUYẾT

NGẮT

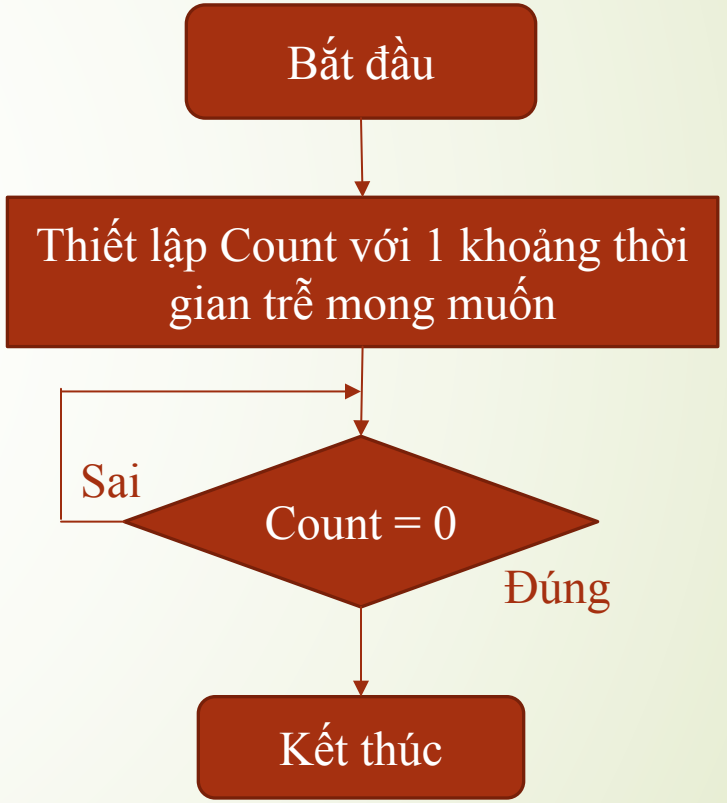
SYSTICK

THUẬT TOÁN

CT con xử lý ngắt



Hàm delay



CODE

GPIO_CLOCK_ENABLE()

Cấp xung cho cổng B

```
void GPIO_CLOCK_ENABLE()
	{
		//enable the clock to GPIO port B
		RCC->AHBENR |=0x00000002;
	}
```

CODE

GPIO_Pin_Init()

Thiết lập thông số của 2 LED

```
void GPIO_Pin_Init()
    {
        //Set pin 6,7 mode as digital input(00)
        GPIOB->MODER &=~(0x0F<<(2*6));// Clear bit 15 14 13 12
        GPIOB->MODER |=0x05<<(2*6);       //Set pin 6,7 as digital output

        GPIOB->OTYPER &= ~(3<<6);         //set output type of pin 6,7  as push-pull

        GPIOB->OSPEEDR &=~(0xF<<(2*6)); // Speed mask
        GPIOB->OSPEEDR |= 0x5<<(2*6);      // set I/0 output speed  value as 2 Mhz


        GPIOB->PUPDR &= ~(0x0F<<(2*6));  // no pull-up, no pull-down
    }
```

CODE

reverseLed()

Đảo trạng thái LED

```
void reverseLed()
{
	GPIOB->ODR ^=0x3<<6;;   //dao trang thai PB6 + PB7
}
```

CODE

SysTick_init()

Khởi tạo và thiết lập thông số của ngắt

```
void SysTick_init(uint32_t ticks)
	{
		SysTick->CTRL = 0;		//disable SysTick IRQ and Systick counter
		SysTick->LOAD = ticks-1; //Set reload register
		NVIC_SetPriority (SysTick_IRQn, (1<<__NVIC_PRIO_BITS)-1);	//Set priority
		SysTick->VAL =0;		//reset the SysTick counter value
		//Select processor clock
		//1= processor clock; 0=external clock
		SysTick->CTRL |=SysTick_CTRL_CLKSOURCE;
		SysTick->CTRL |= SysTick_CTRL_ENABLE; //Enable SysTick IRQ and SysTick timer
		SysTick->CTRL |= SysTick_CTRL_TICKINT;
	}
```

CODE

SysTick_Handler()

DELAY()

Hàm gọi tự động khi ngắt

```
void SysTick_Handler(void)
{
	if(TimingDelay!=0)
		TimingDelay--;
}
```

Hàm delay

```
void DELAY(uint32_t time)
{
	TimingDelay = time;
	while(TimingDelay!=0);
}
```

CODE

main

Hàm main

```
int main(void){

        GPIO_CLOCK_ENABLE();
        GPIO_Pin_Init();
        SysTick_init(2000);
        GPIOB->ODR |= 1<<6;     // led PB6 sang + PB7 tat

while(1) {
                {
                        DELAY(1000);
                        reverseLed();
                }
}
```