**BÀI TẬP VỀ NHÀ TUẦN 19**

**Nhóm 01 - A01:**

|  |  |
| --- | --- |
| **Họ và tên** | **MSSV** |
| Lê Huy | 1812361 |
| Hoàng Đình Toản | 1814379 |
| Nguyễn Thanh Trung | 1814514 |
| Phạm Phương Thanh | 1713112 |

**Đề bài:** Viết 1 chương trình làm công việc sau:

Tạo ra 3 task chạy song song:

* Task1: chớp led đỏ với chu kỳ 1s
* Task 2: Nếu nhấn SW1, bật led Green, nếu nhả, tắt led Green
* Task 3: Nếu nhấn SW2, bật led Blue, nhả ra vẫn bât. Nhấn lần nữa, tắt led Blue, nhả ra vẫn tắt.

Chương trình sử dụng FreeRTOS

**\* Giải thích một số module trong chương trình:**

***1. led.c trong folder driver:*** gồm hai hàm chính là ledInit và ledControl

**- ledInit** có chức năng khởi động GPIO\_PORTF và cài đặt 3 chân PIN1, PIN2, PIN3 là output.  
**- ledControl** có hai parametter là led có kiểu ledNumber và state có kiểu ledState, dùng để điều khiển các LED RED, GREEN, BLUE bật hoặc tắt.

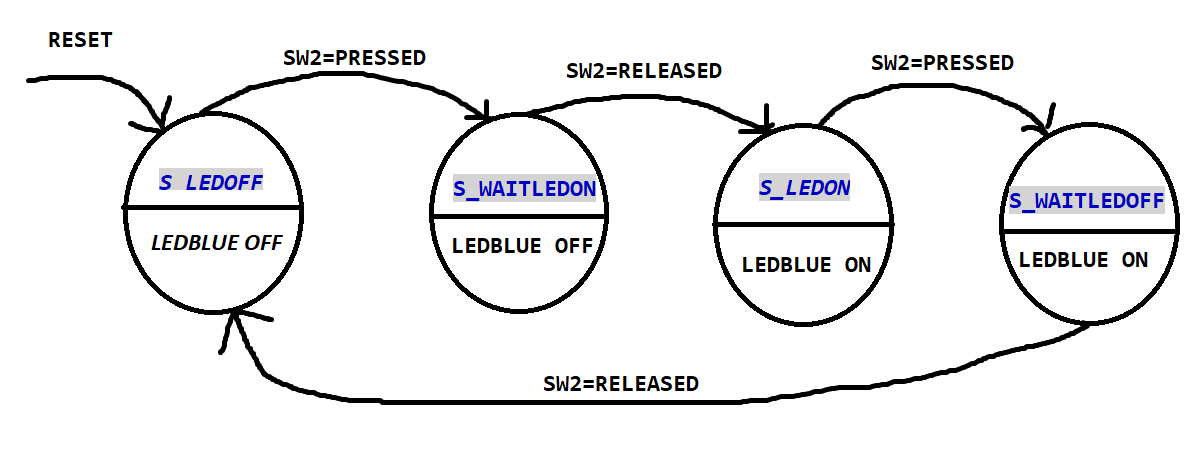
+ led có thể nhận 1 giá trị là *LEDRED, REDBLUE* và *LEDGREEN* tương ứng với địa chỉ 3 pin của portF để điều khiển LED.

+ state có hai giá trị là *ON, OFF*. Nếu state = *ON* thì cho LED sáng, nếu state = *OFF* thì cho LED tắt.

***2. switches.c trong folder driver:*** gồm hai hàm chính là switchInit và switchState

**- switchInit** có chức năng khởi động GPIO\_PORTF và cài đặt 3 chân PIN0, PIN4 là input.  
**- switchState** có một parametter là switchNumber có kiểu sw\_t, dùng để đọc trạng thái của SW0 hoặc SW1 là *PRESSED* hoặc *RELEASED.* NếuswitchNumber = 1 đọc trạng thái của SW1, switchNumber = 2 đọc trạng thái của SW2.

***3. ledBlueStateMachine.c trong folder application:*** gồm hai chương trình chính là InitMachine và ledBlueStateMachineUpdatedùng để cập nhật trạng thái của LEDBLUE dựa vào trạng thái của SW2 theo sơ đồ trạng thái dưới đây:



**\* Giải thích hàm main.c:**

- Khởi động hệ thống với clock = 50MHz.

**SysCtlClockSet**(SYSCTL\_SYSDIV\_4 | SYSCTL\_USE\_PLL |

SYSCTL\_XTAL\_16MHZ | SYSCTL\_OSC\_MAIN);

- Bật LED, Switch và setup trạng thái reset của ledbluemachine.

ledInit();

switchInit();

InitMachine();

- Tạo ra 3 Task là LEDRedTask với mức ưu tiên 1, LEDGreenTask với mức ưu tiên 3 và LEDBlueTask với mức ưu tiên 2.

xTaskCreate( LEDRedTask, "LEDTask 1",

configMINIMAL\_STACK\_SIZE, NULL,1, NULL );

xTaskCreate( LEDGreenTask, "LEDTask 2",

configMINIMAL\_STACK\_SIZE, NULL,3, NULL );

xTaskCreate( LEDBlueTask, "LEDTask 3",

configMINIMAL\_STACK\_SIZE, NULL, 2, NULL );

- Khởi động lịch cho 3 task:

vTaskStartScheduler();

***- Hàm LEDRedTask:***

**static** **void** **LEDRedTask**(**void** \*pvParameters)

{

**while** (1)

{

ledControl(*LEDRED*,*ON*); //Turn on LED

vTaskDelay(500/portTICK\_RATE\_MS);

ledControl(*LEDRED*,*OFF*); //Turn off LED

vTaskDelay(500/portTICK\_RATE\_MS);

}

}

vTaskDelay(500/portTICK\_RATE\_MS) có tác dụng đưa LEDRedTask vào trạng thái Blocked trong 500ms, sau 500ms LEDRedTask sẽ ở trạng thái Ready và sẽ chuyển qua trạng thái Running để đảo trạng thái LED RED hiện tại.

***- Hàm LEDBlueTask:***

**static** **void** **LEDBlueTask**(**void** \*pvParameters)

{

**while** (1)

{ledBlueStateMachineUpdate();

vTaskDelay(100);

}

}

LedBlueTask sẽ cập nhật liên tục trạng thái LED BLUE thông qua hàm ledBlueStateMachineUpdate(), sau đó sẽ vào trạng thái Block trong 100 x TicksToDelay lần Systick. Hết thời gian TaskDelay, LedBlueTask sẽ vào trang thái ready.

***- Hàm LEDGreenTask:***

**static** **void** **LEDGreenTask**(**void** \*pvParameters)

{

**while** (1)

{**if**(switchState(1) == *PRESSED*) ledControl(*LEDGREEN*,*ON*);

vTaskDelay(100);

**if**(switchState(1) == *RELEASED*) ledControl(*LEDGREEN*,*OFF*);

vTaskDelay(100);

}

}

LehGreenTask sẽ cập nhật trạng thái SW1 sau 100 x TicksToDelay lần Systick, nếu SW1 = *PRESSED* thì bật LED GREEN, nếu SW1 = *RELEASED* thì tắt LED GREEN.