

**ĐẠI HỌC QUỐC GIA
ĐẠI HỌC BÁCH KHOA TP HỒ CHÍ MINH**



– Embedded System–

LAB 5

Nhóm 2 – L02

Họ và tên	MSSV
Trần Nguyễn Minh Duy	1910095
Đặng Trung Kiên	1911437
Nguyễn Hải Long	1911517
Nguyễn Nhật Trường	1912344

Thành phố Hồ Chí Minh – 2022

MỤC LỤC

ĐÓNG GÓP CỦA CÁC THÀNH VIÊN	2
BÁO CÁO.....	3
1. Hiện thực	3
<i>1. Hàm in thời gian hiện tại (theo giây).....</i>	<i>3</i>
<i>2. Hàm callback của timer</i>	<i>3</i>
<i>3. Hàm main</i>	<i>4</i>
2. Kết quả.....	6

ĐÓNG GÓP CỦA CÁC THÀNH VIÊN

Họ và tên	MSSV	Đóng góp
Trần Nguyễn Minh Duy	1910095	100%
Đặng Trung Kiên	1911437	100%
Nguyễn Hải Long	1911517	100%
Nguyễn Nhật Trường	1912344	100%

BÁO CÁO

1. Hiện thực

1. Hàm in thời gian hiện tại (theo giây)

```
struct timeval tv_now;
static inline void print_current_time()
{
    gettimeofday(&tv_now, NULL);
    printf("t = %d\\t", tv_now.tv_sec);
}
```

- Hàm in ra thời gian hiện tại bằng hàm gettimeofday().
- Cú pháp: t = (giây hiện tại) \\t

2. Hàm callback của timer

```
uint8_t timer1_reading_time = 0;
uint8_t timer2_reading_time = 0;

static void vTimerCallback(TimerHandle_t tim )
{
    uint32_t id = (uint32_t) pvTimerGetTimerID(tim);

    if (id == 1)
    {
        timer1_reading_time++;

        print_current_time();
        printf("%s\\t(%d)\\n", TIMER_1_MESSAGE, timer1_reading_time);

        if (timer1_reading_time == TIMER_1_READING_TIME)
        {
            if (xTimerStop(tim, 0) == pdFAIL)
            {
                printf("Cound not to stop timer 1\\n");
            }
        }
    }
    else if (id == 2)
    {
        timer2_reading_time++;

        print_current_time();
        printf("%s\\t(%d)\\n", TIMER_2_MESSAGE, timer2_reading_time);

        if (timer2_reading_time == TIMER_2_READING_TIME)
```

```

    {
        if (xTimerStop(tim, 0) == pdFAIL)
        {
            printf("Cound not to stop timer 2\n");
        }
    }
}
else
{
    printf("Error: Invalid ID %d\n", id);
}
}

```

- Đầu tiên, ta dùng hàm `pvTimerGetTimerID()` để có được ID của timer đang gọi hàm Callback.
- Tiếp theo, tương ứng với ID trả về, ta sẽ thực hiện các hành động sau:
 - o Cộng thêm 1 cho biến đếm số lần in của mỗi timer (`timer1_reading_time` và `timer2_reading_time`)
 - o In chuỗi kí tự tương ứng với mỗi timer (timer 1: “ahihi”, timer2: “ihaha”) theo cú pháp:
`t = (thời gian hiện tại) \t (“ahihi” hoặc “ihaha”) \t (số lần in)`
 - o Cuối cùng, kiểm tra xem số lần in của mỗi timer có đạt tới số lần tối đa hay chưa (timer 1: 10 lần, timer2: 5 lần). Nếu đã đủ, ta dùng hàm `xTimerStop()` để dừng timer đó.

3. Hàm main

```

#define TIMER_1_PERIOD 2000
#define TIMER_2_PERIOD 3000

TimerHandle_t timer1;
TimerHandle_t timer2;

BaseType_t xTimer1Started;
BaseType_t xTimer2Started;

void app_main(void)
{
    timer1_reading_time = 0;
    timer1 = xTimerCreate(
        "Timer 1",
        pdMS_TO_TICKS(TIMER_1_PERIOD),

```

```

    pdTRUE, // Auto reload
    1,
    vTimerCallback);

timer2_reading_time = 0;
timer2 = xTimerCreate(
    "Timer 2",
    pdMS_TO_TICKS(TIMER_2_PERIOD),
    pdTRUE, // Auto reload
    2,
    vTimerCallback);

if ((timer1 != NULL) && (timer2 != NULL))
{
    xTimer1Started = xTimerStart(timer1, 0);
    xTimer2Started = xTimerStart(timer2, 0);
    // if ((xTimer1Started == pdPASS) && (xTimer2Started == pdPASS))
    // {
    //     ;
    // }
}
}

```

Ở hàm main, ta lần lượt thực hiện các hành động sau:

- Khởi tạo biến đếm số lần in (timer1_reading_time = 0) cho timer 1.
- Dùng hàm xTimerCreate() để tạo timer 1:
 - o Tên timer: "Timer 1"
 - o Period: 2000 ms
 - o AutoReload: pdTRUE
 - o ID = 1
 - o Hàm callback: vTimerCallback
- Khởi tạo biến đếm số lần in (timer2_reading_time = 0) cho timer 2.
- Dùng hàm xTimerCreate() để tạo timer 2:
 - o Tên timer: "Timer 2"
 - o Period: 3000 ms
 - o AutoReload: pdTRUE
 - o ID = 2
 - o Hàm callback: vTimerCallback
- Nếu 2 timer đã được tạo thành công, ta dùng hàm xTimerStart để bắt đầu 2 timer.

2. Kết quả

```
I (0) cpu_start: Starting scheduler on APP CPU.  
t = 2  ahihi  (1)  
t = 3  ihaha  (1)  
t = 4  ahihi  (2)  
t = 6  ihaha  (2)  
t = 6  ahihi  (3)  
t = 8  ahihi  (4)  
t = 9  ihaha  (3)  
t = 10 ahihi  (5)  
t = 12 ihaha  (4)  
t = 12 ahihi  (6)  
t = 14 ahihi  (7)  
t = 15 ihaha  (5)  
t = 16 ahihi  (8)  
t = 18 ahihi  (9)  
t = 20 ahihi  (10)
```

- Vì timer 1 sẽ in “ahihi” mỗi 2 giây và timer 1 sẽ in 10 lần, nên vào thời điểm $t = 2, 4, 6, 8, 10, 12, 14, 16, 18, 20$ chuỗi “ahihi” được in ra.
- Vì timer 2 sẽ in “ihaha” mỗi 3 giây và timer 2 sẽ in 5 lần, nên vào thời điểm $t = 3, 6, 9, 12, 15$ chuỗi “ihaha” được in ra.