**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](#_Toc121751351)

[LINK GITHUB 3](#_Toc121751352)

[BÁO CÁO 4](#_Toc121751353)

[**1. Hiện thực** 4](#_Toc121751354)

[*1. Hàm in thời gian hiện tại (theo giây)* 4](#_Toc121751355)

[*2. Hàm callback của timer* 4](#_Toc121751356)

[*3. Hàm main* 5](#_Toc121751357)

[**2. Kết quả** 7](#_Toc121751358)

**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% |

# LINK GITHUB

https://github.com/duytran1511/Embedded\_System\_LAB

**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:
  + Cộng thêm 1 cho biến đếm số lần in của mỗi timer (timer1\_reading\_time và timer2\_reading\_time)
  + In chuỗi ký 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)

* + 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))

        // {

        //     ;

        // }

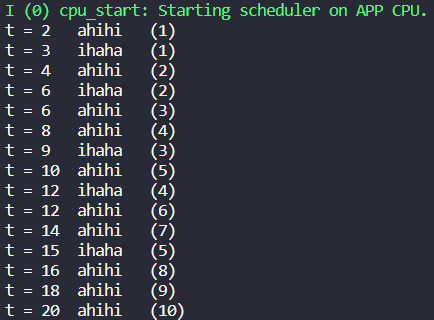
    }

}

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:
  + Tên timer: “Timer 1”
  + Period: 2000 ms
  + AutoReload: pdTRUE
  + ID = 1
  + Hàm callback: vTimerCallback
* Khởi tạo biến đếm số lần in (timer2\_reading\_time = 0) cho timer 2.
* Dùng hàm xTimerCreate() để tạo timer 2:
  + Tên timer: “Timer 2”
  + Period: 3000 ms
  + AutoReload: pdTRUE
  + ID = 2
  + Hàm callback: vTimerCallback
* Nếu 2 timer đã được tạo thành công, ta dùng hàm xTimerStart để bắt đầu 2 timer.

## **2. Kết quả**



* 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.