

VIETNAM NATIONAL UNIVERSITY - HCM
Ho Chi Minh City University of Technology
Faculty of Computer Science and Engineering



EMBEDDED SYSTEM (CO3053)

REPORT LAB 4

FreeRTOS Software Timer

Name	Student's ID
Nguyễn Hữu Trung Nhân	1752392

Ho Chi Minh City, June 2020 - Semester 192



Contents

1	Goal	2
2	Implementation	2
3	Source Code	3
4	Picture of my workspace	4
5	The Final Result	5

1 Goal

The objective is to create **2 software timers**. These 2 timers share **one timer callback function**

The first timer is used to print "ahihi" every 2 seconds and will stop after 10 times printing.

The second timer is used to print "ihaha" every 3 seconds and will stop after 5 times printing.

2 Implementation

After reading the lab material extremely carefully, I implemented my program based on some provided API function.

In order to create a software timer that call the callback function every 2 seconds, I did as below:

```
1 xTimerHandle timerHnd2Sec = xTimerCreate("timer2sec", pdMS_TO_TICKS(2000), pdTRUE,
    (void*)0, vTimerCallBackExpired);
2 xTimerStart(timerHnd2Sec, 0);
3
4 //pdMS_TO_TICKS(2000) means that this timer will call the callback function "
    vTimerCallBackExpired" (in my case, this is the name I choose for callback
    function)
```

In order to create a software timer that call the callback function every 3 seconds, I did as below:

```
1 timerHnd3Sec = xTimerCreate("timer3sec", pdMS_TO_TICKS(3000), pdTRUE, (void*)0,
    vTimerCallBackExpired);
2 xTimerStart(timerHnd3Sec, 0);
```

Because the requirement is **2 software timers** share "one timer callback function", I implement the callback function "vTimerCallBackExpired" as below:

```
1
2 int timesFirstTimer = 0;
3 int timesSecondTimer = 0;
4
5 static void vTimerCallBackExpired(xTimerHandle pxTimer){
6     if (pxTimer == timerHnd2Sec){
7         printf("ahihi\n");
8         timesFirstTimer = timesFirstTimer + 1;
9     }
10
11     if (timesFirstTimer == 10){
12         timesFirstTimer = 0;
13         xTimerDelete(timerHnd2Sec, 0);
14     }
15
16     if (pxTimer == timerHnd3Sec){
17         printf("ihahha\n");
18         timesSecondTimer = timesSecondTimer + 1;
19     }
20
21     if (timesSecondTimer == 5){
22         timesSecondTimer = 0;
23         xTimerDelete(timerHnd3Sec, 0);
24     }
25 }
26 }
```

Because the requirement are **the first timer will stop after 10 times printing** and **the second timer will stop after 5 times printing**, I created two variables "timesFirstTimer" and "timesSecondTimer". Whenever the first software timer calls this callback function, "timesFirstTimer" will increase 1 unit, when "timesFirstTimer" reaches 10, xTimerDelete() API function will be called to delete the first software timer. The same idea is applied for the second software timer.

3 Source Code

```
1 #include <stdio.h>
2 #include "freertos/FreeRTOS.h"
3 #include "freertos/task.h"
4 #include "esp_system.h"
5 #include "esp_spi_flash.h"
6 #include "sdkconfig.h"
7 #include "driver/gpio.h"
8 #include "freertos/timers.h"
9
10 xTimerHandle timerHnd2Sec;
11 xTimerHandle timerHnd3Sec;
12
13 int timesFirstTimer = 0;
14 int timesSecondTimer = 0;
15
16 void task1(void *p){
17     while(1){
18
19     }
20     vTaskDelete(NULL);
21 }
22
23 static void vTimerCallBackExpired(xTimerHandle pxTimer){
24     if (pxTimer == timerHnd2Sec){
25         printf("ahihi\n");
26         timesFirstTimer = timesFirstTimer + 1;
27     }
28
29     if (timesFirstTimer == 10){
30         timesFirstTimer = 0;
31         xTimerDelete(timerHnd2Sec, 0);
32     }
33
34     if (pxTimer == timerHnd3Sec){
35         printf("ihahha\n");
36         timesSecondTimer = timesSecondTimer + 1;
37     }
38
39     if (timesSecondTimer == 5){
40         timesSecondTimer = 0;
41         xTimerDelete(timerHnd3Sec, 0);
42     }
43 }
44
45
46 void app_main(void)
47 {
48     xTaskCreate(&task1, "task1", 1024*1, (void*) 0, tskIDLE_PRIORITY + 0, NULL);
49 }
```



```
50 timerHnd2Sec = xTimerCreate("timer2sec", pdMS_TO_TICKS(2000), pdTRUE, (void*)  
51 0, vTimerCallBackExpired);  
52 xTimerStart(timerHnd2Sec, 0);  
53  
54 timerHnd3Sec = xTimerCreate("timer3sec", pdMS_TO_TICKS(3000), pdTRUE, (void*)  
55 0, vTimerCallBackExpired);  
56 xTimerStart(timerHnd3Sec, 0);  
57 }
```

4 Picture of my workspace

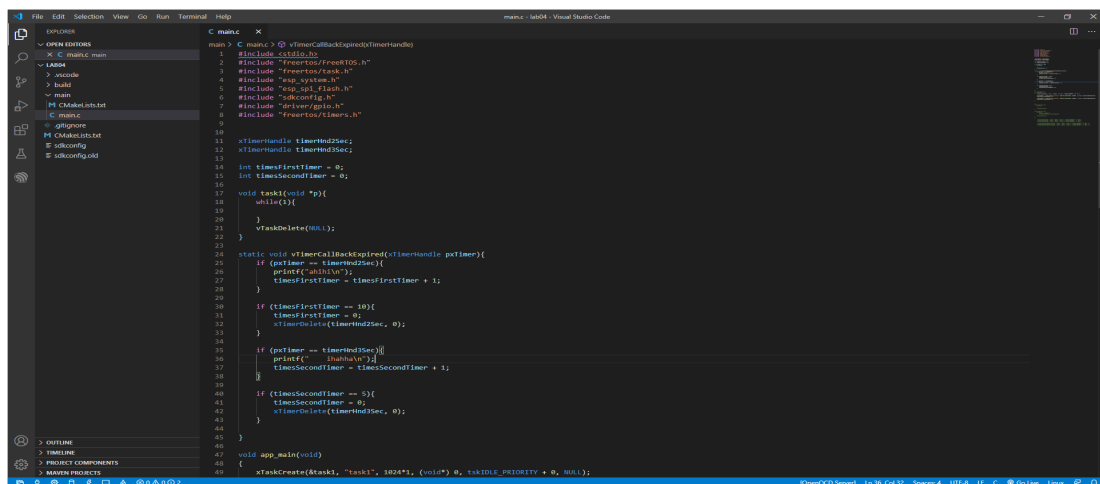


Figure 1

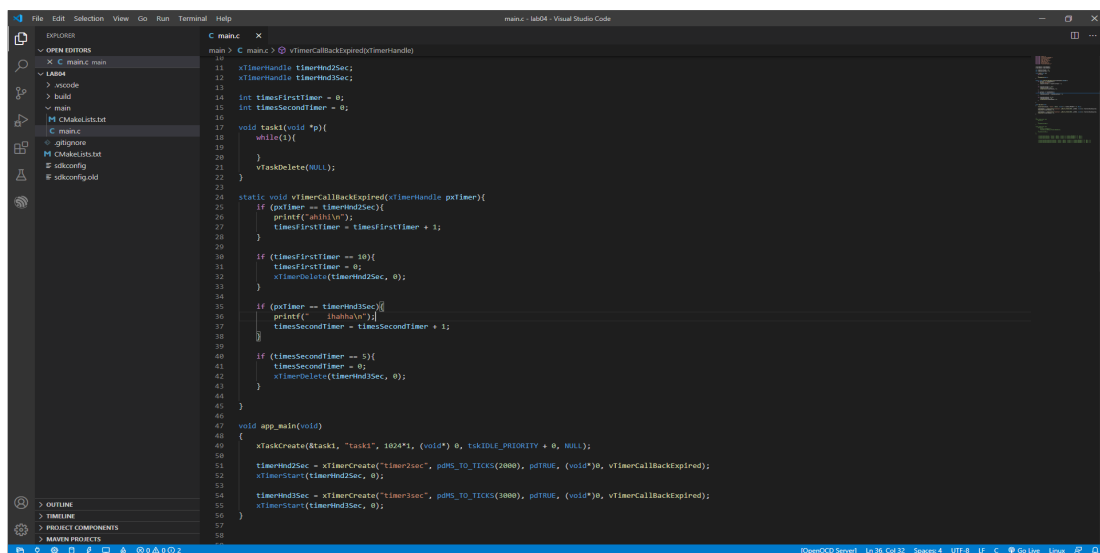


Figure 2

5 The Final Result

```
I (266) heap_init: At 3FFE4350 len 0001BCB0 (111 KiB): D/IRAM
I (272) heap_init: At 40089998 len 00016668 (89 KiB): IRAM
I (278) cpu_start: Pro cpu start user code
I (296) spi_flash: detected chip: generic
I (297) spi_flash: flash io: dio
W (297) spi_flash: Detected size(4096k) larger than the size in the b
I (307) cpu_start: Starting scheduler on PRO CPU.
I (0) cpu_start: Starting scheduler on APP CPU.
ahihi
  ihahha
ahihi
  ihahha
ahihi
ahihi
  ihahha
ahihi
  ihahha
ahihi
ahihi
  ihahha
ahihi
ahihi
ahihi
```

Figure 3