

Assignment 3

Part 1 :

Q1: When you execute the project, What do you get? What is the reason for that?

When I clicked the play button, the system prompted an empty terminal. The reason is that the C code's logic always starts from the main function in main.c file, but there is no print code inside the main function, as shown in the picture below.



```
int main( void )
{
    /* This demo uses heap_5.c, so start by defining some heap regions.  he
    is only used for test and example reasons.  Heap_4 is more appropriate.
    http://www.freertos.org/a00111.html for an explanation. */
    prvInitialiseHeap();

    /* Initialise the trace recorder.  Use of the trace recorder is optional
    See http://www.FreeRTOS.org/trace for more information. */
    vTraceEnable( TRC_START );

    vTaskStartScheduler();
    for (;;)
    {
        return 0;
    }
}

/*-----*/
```

Q2: Figure out what is the meaning of the parameters passed to the xTaskCreate function

From task.h, we can find the definition of xTaskCreate, the function is to create a new task and put the task to run.

```
xTaskCreate(HelloTask, "HelloTask", 100, NULL, 1, &HT);
```

```
BaseType_t xTaskCreate( TaskFunction_t pxTaskCode,  
                        const char * const pcName, /*lint !e971 Unqualified char types are  
allowed for strings and single characters only. */  
                        const configSTACK_DEPTH_TYPE usStackDepth,  
                        void * const pvParameters,  
                        UBaseType_t uxPriority,  
                        TaskHandle_t * const pxCreatedTask ) PRIVILEGED_FUNCTION;
```

Parameter	Description
pxTaskCode	The name of the function which going to implement in the task list. In this project, we called function: HelloTask()
pcName	A name to represent the task function is useful because it can be used to get task handle.
usStackDepth	A number to define the size of task`s stack, the stack size = usStackDepth * system word length.
pvParameters	Passing the parameter to implement the task. In this example, HelloTask() does not need any parameter, so it can be NULL.
uxPriority	Setting the priority of the task, task will run first if it has high priority.
pxCreatedTask	This parameter gets the handle and passes to the task, which can be NULL.

Part 2

Q3: A screenshot of the execution in a report



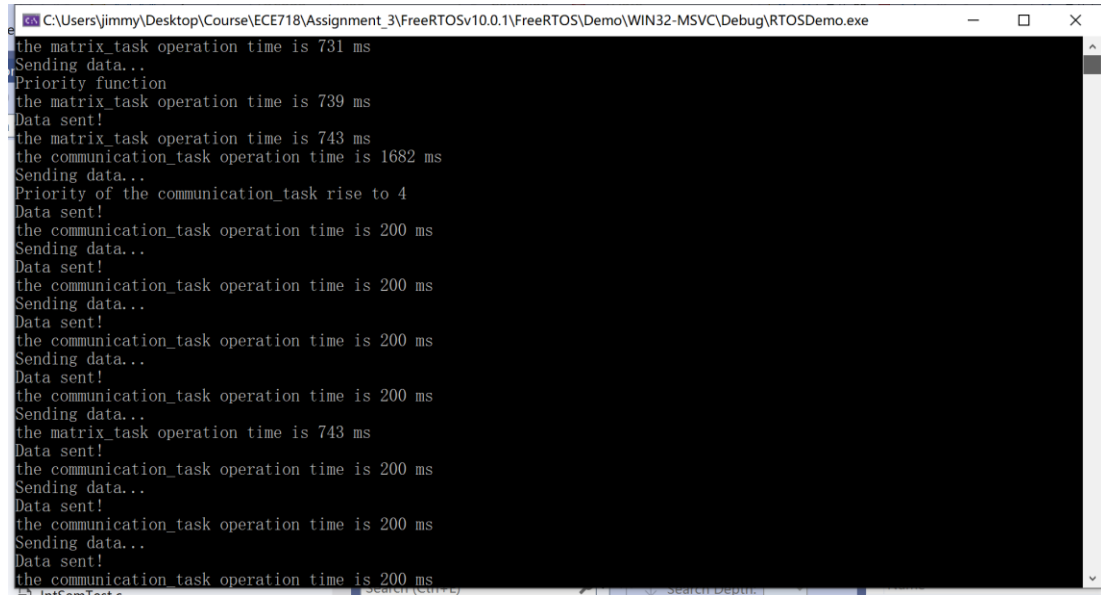
```
C:\Users\jimmy\Desktop\Course\ECE718\Assignment_3\FreeRTOSv10.0.1\FreeRTOS\Demo\WIN32-MSVC\Debug\RTOSDemo.exe
This is task 1
This is task 2
This is task 1
This is task 1
This is task 1
This is task 1
This is task 1
This is task 1
This is task 2
This is task 1
This is task 1
This is task 1
This is task 1
This is task 1
This is task 1
This is task 2
This is task 2
This is task 1
This is task 1
This is task 1
This is task 1
This is task 1
This is task 2
This is task 1
This is task 1
This is task 1
This is task 1
This is task 2
This is task 1
This is task 1
This is task 1
This is task 1
This is task 2
This is task 1
This is task 1
This is task 1
```

Part 3

Q4) “communicationtask” must send a simulated data packet every 200ms but is often blocked by matrixtask, fix this problem without changing the functionality in the tasks

The communicationtask is often blocked by matrixtask because the priority of communicationtask is lower, so the computing power is used to calculate matrixtask at first. To solve this problem, we can reverse the priority of the two tasks.

Q5) Provide a screenshot of the execution and answer the following questions in a report:



```
C:\Users\jimmy\Desktop\Course\ECE718\Assignment_3\FreeRTOSv10.0.1\FreeRTOS\Demo\WIN32-MSVC\Debug\RTOSDemo.exe
the matrix_task operation time is 731 ms
Sending data...
Priority function
the matrix_task operation time is 739 ms
Data sent!
the matrix_task operation time is 743 ms
the communication_task operation time is 1682 ms
Sending data...
Priority of the communication_task rise to 4
Data sent!
the communication_task operation time is 200 ms
Sending data...
Data sent!
the communication_task operation time is 200 ms
Sending data...
Data sent!
the communication_task operation time is 200 ms
Sending data...
Data sent!
the communication_task operation time is 200 ms
Sending data...
Data sent!
the communication_task operation time is 200 ms
Sending data...
Data sent!
the communication_task operation time is 200 ms
Sending data...
Data sent!
the communication task operation time is 200 ms
```

- Why is "matrixtask" using most of the CPU utilization?

There has two reasons:

1. There are too many for loops inside the function. It takes a long time for computation, especially: `for (simulationdelay = 0; simulationdelay < 1000000000; simulationdelay++)`.
2. The priority of "matrixtask" is 3, it is higher than other task.

- Why must the priority of "communicationtask" increase in order for it to work properly

Because "communicationtask" has low priority, "matrixtask" costs most computing power. Therefore, we must increase "communicationtask" priority to avoid being blocked.

- What happens to the completion time of "matrixtask" when the priority of "communicationtask" is increased?

In my opinion, the running completion time of "matrixtask" will grow. However, the output result shows that the completion time of "matrixtask" does not change. Therefore, I think "communication" has a short running time and costs less

computing power. Even if the priority of "communication" is increased, the remaining computing power can still ensure that "matrixtask" is completed on time.

- How many seconds is the period of "matrixtask"? (Hint: look at `vApplicationTickHook()` to measure it)

As seen in the screenshot, the period of aperiodic is about 740ms.

Part 4

Q6. The following questions should be solved with programming and the questions should be answered in a report:

- Is the system fast enough to handle all aperiodic tasks? Why?

No, the priority of aperiodic tasks is 2, but the priority of matrix is 3. The system blocked the aperiodic tasks to run the matrix.

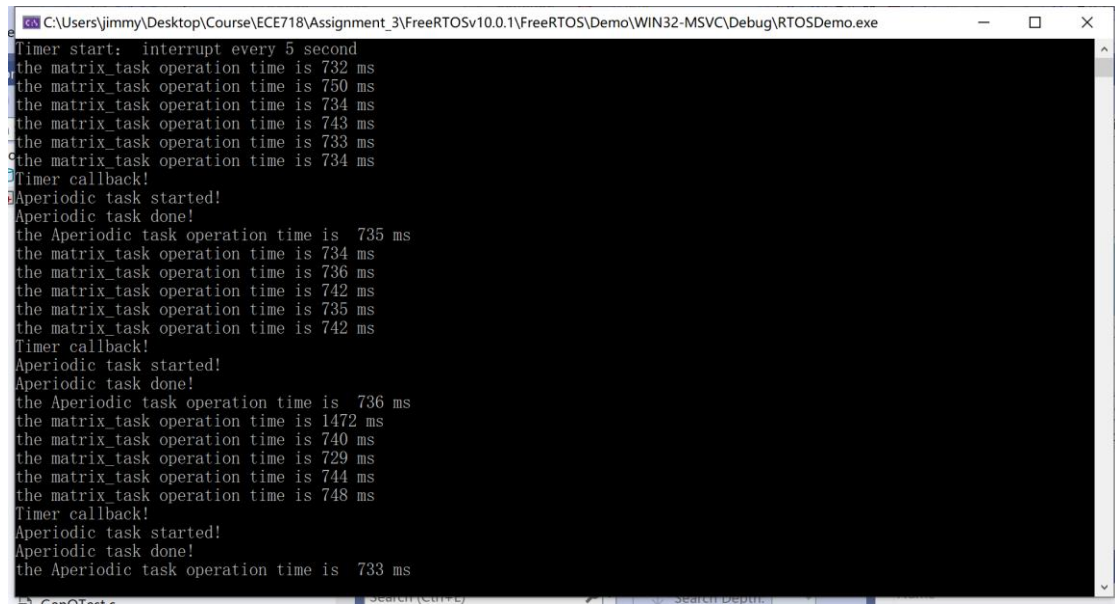
- If not, solve this problem without alter the functionality of any task

Change the priority of aperiodic tasks. The number should be larger than 3.

- What is the response time of the aperiodic task?

As seen in the screenshot, the period of aperiodic is about 750ms.

- Provide a screenshot of the running system



```
C:\Users\jimmy\Desktop\Course\ECE718\Assignment_3\FreeRTOSv10.0.1\FreeRTOS\Demo\WIN32-MSVC\Debug\RTOSDemo.exe
Timer start: interrupt every 5 second
the matrix_task operation time is 732 ms
the matrix_task operation time is 750 ms
the matrix_task operation time is 734 ms
the matrix_task operation time is 743 ms
the matrix_task operation time is 733 ms
the matrix_task operation time is 734 ms
Timer callback!
Aperiodic task started!
Aperiodic task done!
the Aperiodic task operation time is 735 ms
the matrix_task operation time is 734 ms
the matrix_task operation time is 736 ms
the matrix_task operation time is 742 ms
the matrix_task operation time is 735 ms
the matrix_task operation time is 742 ms
Timer callback!
Aperiodic task started!
Aperiodic task done!
the Aperiodic task operation time is 736 ms
the matrix_task operation time is 1472 ms
the matrix_task operation time is 740 ms
the matrix_task operation time is 729 ms
the matrix_task operation time is 744 ms
the matrix_task operation time is 748 ms
Timer callback!
Aperiodic task started!
Aperiodic task done!
the Aperiodic task operation time is 733 ms
```