Mission 1:

"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.

Comment:

- There is not clear definition about how to solve it. So I use a trivial way.
- Since I use Linux+gcc, in order to compile well, I have to modity the code in order to obey C standard (this course assumes everybody uses Visual Studio, which doesn’t respect C standard). But the functionality is still the same.

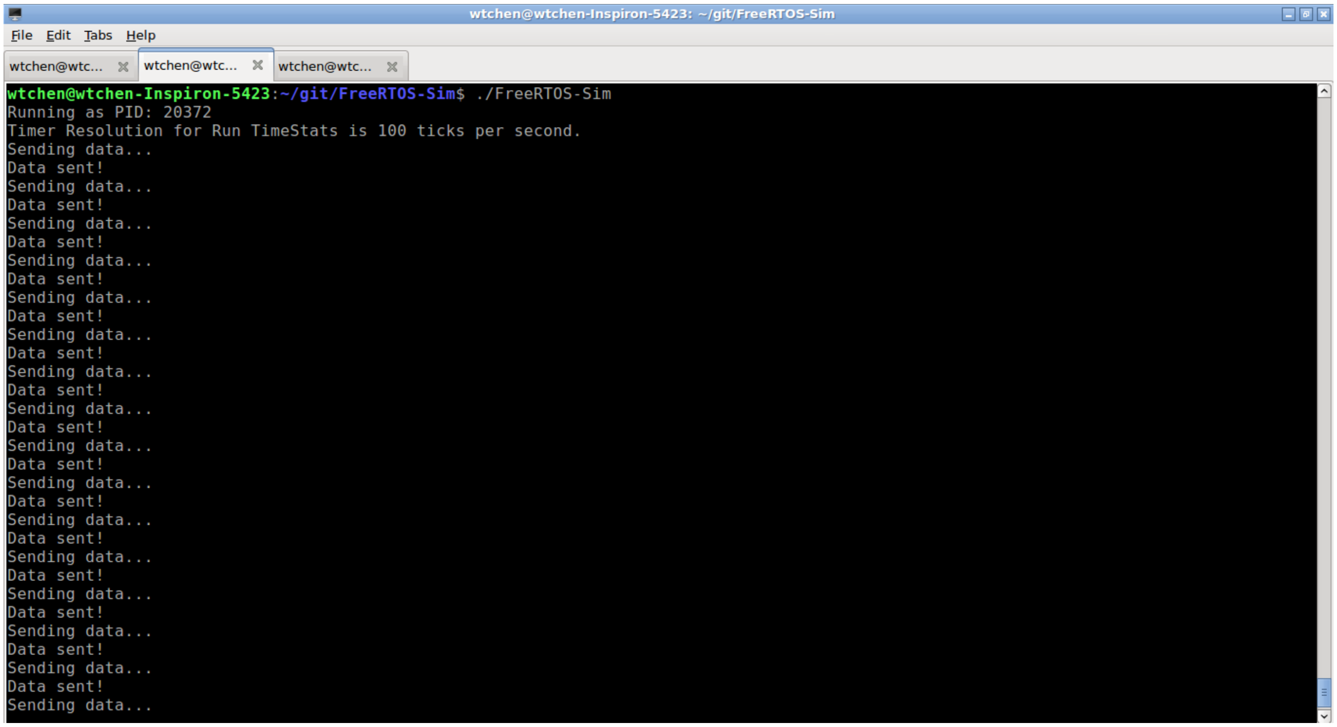
Solution:

It is quite trivial way:

- Create another task which can low the priority of matrix_task and then suspend itself. xTaskCreate((pdTASK_CODE)matrix_adjust, (char *)"Matrix_adjust", configMINIMAL_STACK_SIZE, NULL, 4, &adjust_handle);
- After vTaskStartScheduler(), since matrix_adjust has higher priority, it will run first and lower the priority of matrix_task. Since in my configuration configUSE_PREEMPTION = 1, communication_task will always preempt the resource when it’s time to run.

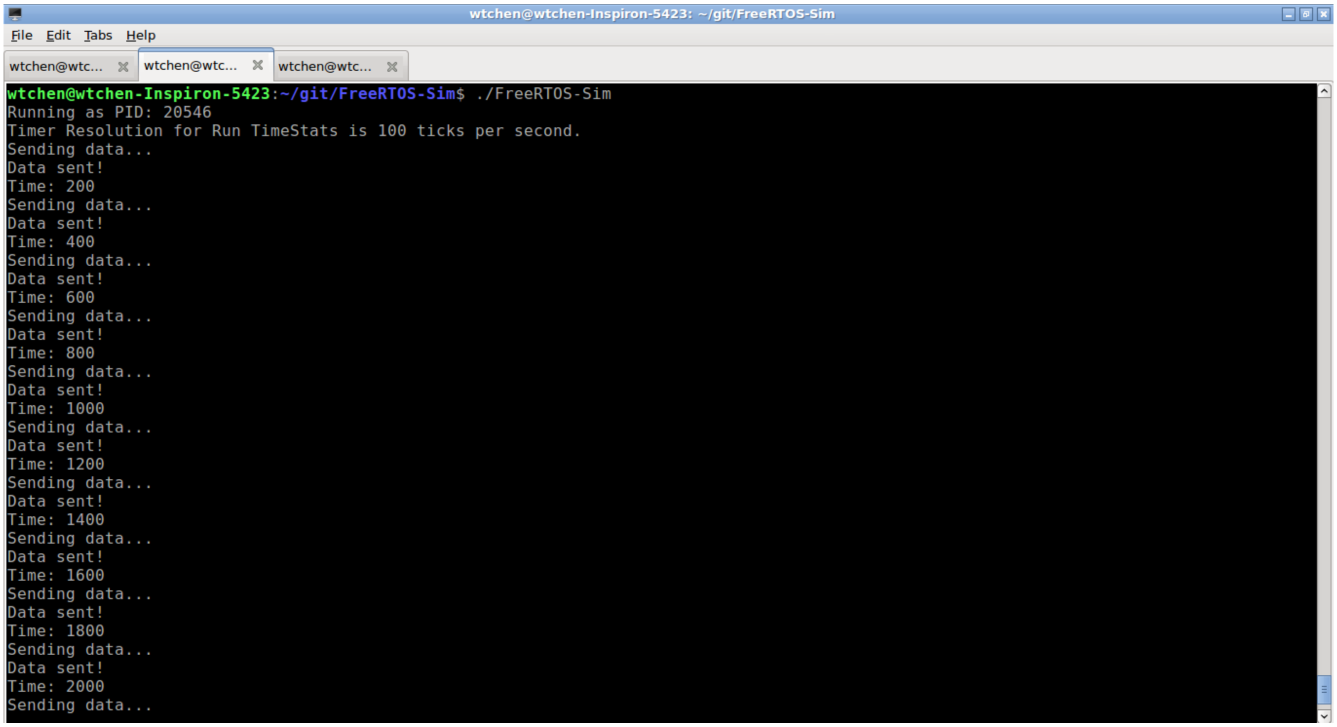
The source code main.c is in mission_1 . Please just copy it from it to the directory “Project”, and then make it.

Here is the screenshot:



```
wtchen@wtchen-Inspiron-5423:~/git/FreeRTOS-Sim$ ./FreeRTOS-Sim
Running as PID: 20372
Timer Resolution for Run TimeStats is 100 ticks per second.
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
```

In order to proof that the time is limited to 200ms, I added a printf to communication_task. Here is the screenshot:



```
wtchen@wtchen-Inspiron-5423:~/git/FreeRTOS-Sim$ ./FreeRTOS-Sim
Running as PID: 20546
Timer Resolution for Run TimeStats is 100 ticks per second.
Sending data...
Data sent!
Time: 200
Sending data...
Data sent!
Time: 400
Sending data...
Data sent!
Time: 600
Sending data...
Data sent!
Time: 800
Sending data...
Data sent!
Time: 1000
Sending data...
Data sent!
Time: 1200
Sending data...
Data sent!
Time: 1400
Sending data...
Data sent!
Time: 1600
Sending data...
Data sent!
Time: 1800
Sending data...
Data sent!
Time: 2000
Sending data...
```

You can see that time increase 200 for every loop.

Mission 2 :

Create a new task "prioritysettask" which:

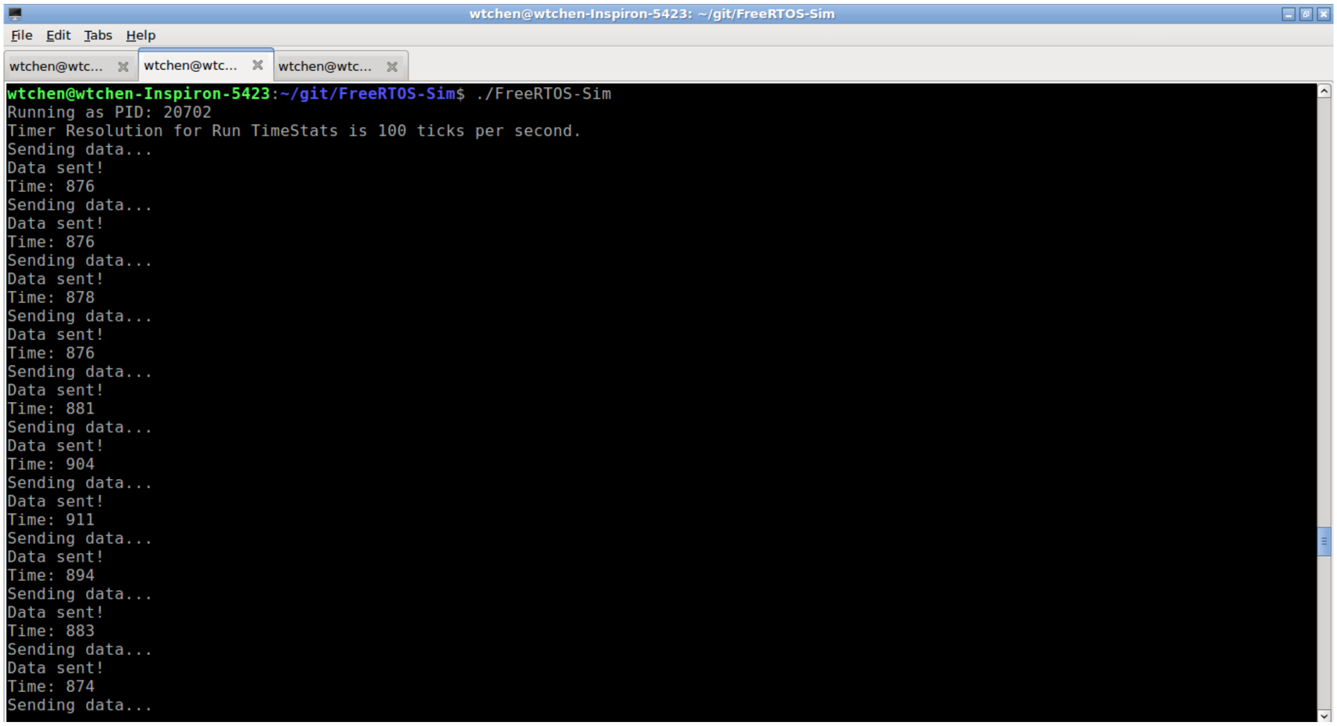
1. Sets the priority of "communicationtask" to 4 in case its execution time is more than 1000 milliseconds (Hint: look at vApplicationTickHook() to measure it)
2. Sets the priority of "communicationtask" to 2 in case its execution time is less than 200 milliseconds (Hint: look at vApplicationTickHook() to measure it)
3. Provide a screenshot of the execution and answer the following questions in a report

Comment :

- The definition of “execution time” is not clear to me. For me this task will run like this : (state: Ready) → printf + fflush (state: Running) → vTaskDelay(100) (state: Block) → (state: Ready) → printf + fflush (state: Running) → vTaskDelay(100) (state: Block) → Loop…
- Since it’s not clear for me, I assume the “execution time” is one loop of above…
- Actually we don’t need to use vApplicationTickHook() to measure it, since in this mission doesn’t forbid any modification of the functionality…

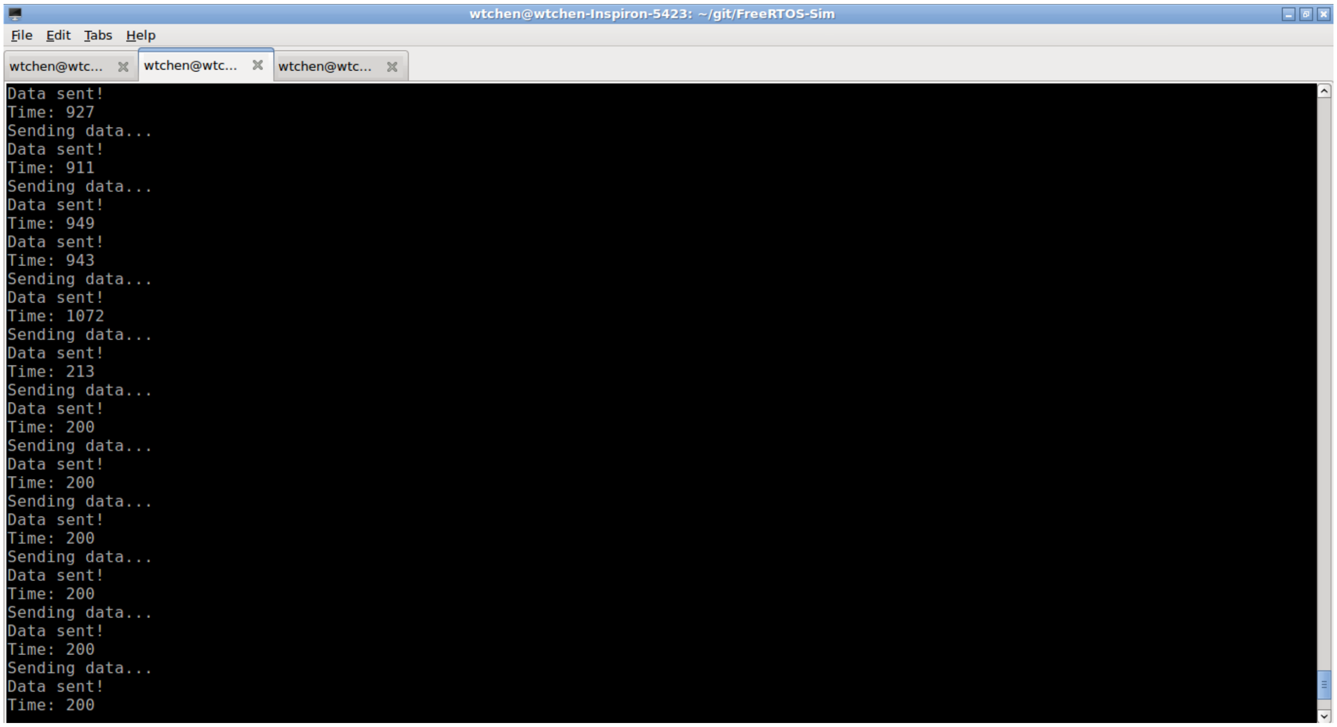
- I have modified the function used in communication_task in order to measure the “execution time”

Here is the screenshot at the beginning:



```
wtchen@wtchen-Inspiron-5423:~/git/FreeRTOS-Sim$ ./FreeRTOS-Sim
Running as PID: 20702
Timer Resolution for Run TimeStats is 100 ticks per second.
Sending data...
Data sent!
Time: 876
Sending data...
Data sent!
Time: 876
Sending data...
Data sent!
Time: 878
Sending data...
Data sent!
Time: 876
Sending data...
Data sent!
Time: 881
Sending data...
Data sent!
Time: 904
Sending data...
Data sent!
Time: 911
Sending data...
Data sent!
Time: 894
Sending data...
Data sent!
Time: 883
Sending data...
Data sent!
Time: 874
Sending data...
```

After some time, when the execution > 1000, it turns to :



```
Data sent!
Time: 927
Sending data...
Data sent!
Time: 911
Sending data...
Data sent!
Time: 949
Data sent!
Time: 943
Sending data...
Data sent!
Time: 1072
Sending data...
Data sent!
Time: 213
Sending data...
Data sent!
Time: 200
Sending data...
Data sent!
Time: 200
Sending data...
Data sent!
Time: 200
Sending data...
Data sent!
Time: 200
Sending data...
Data sent!
Time: 200
Sending data...
Data sent!
Time: 200
```

Then I have waited long time but it is never <200…

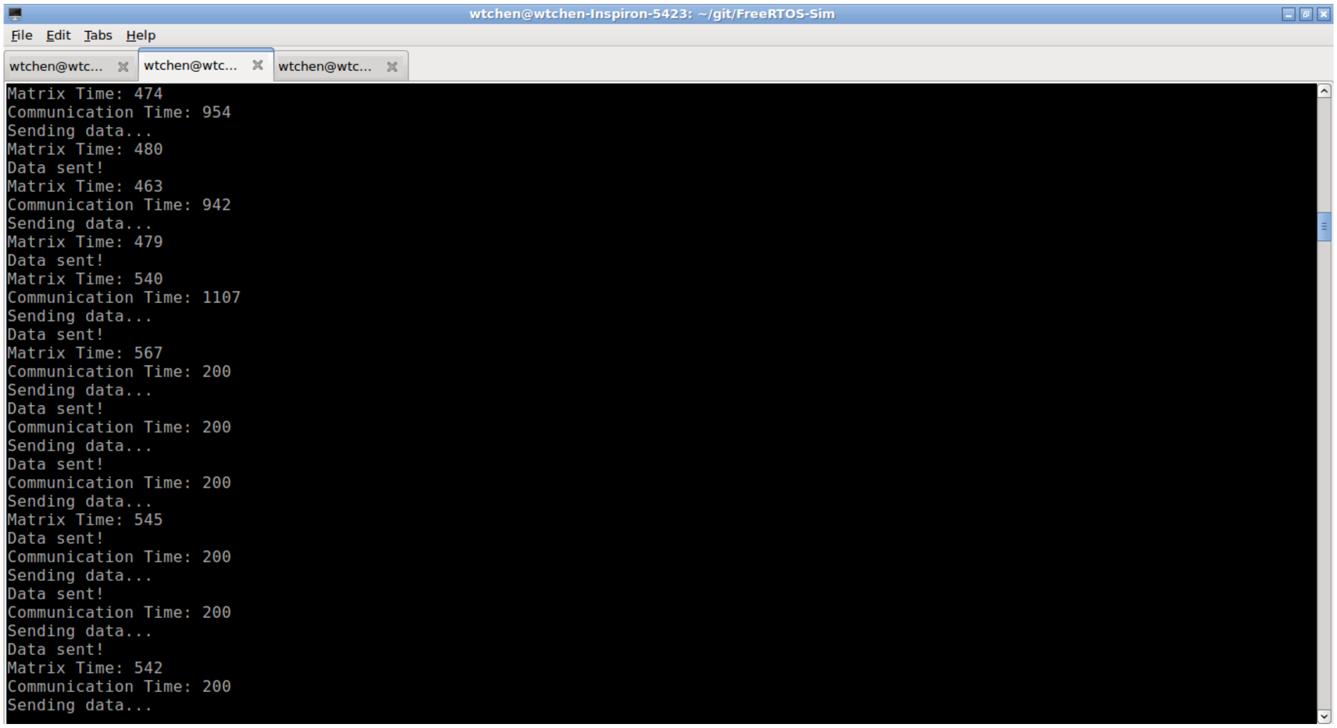
Now, answer the questions:

1. Why is "matrix_task" using most of the CPU utilization?
   Ans: Because its priority is higher (3) than comminucation_task (1). And in the config file configUSE_PREEMPTION = 1, which allows tasks with higher priority to preempt the resource of tasks with lower priority.

2. Why must the priority of "communication_task" increase in order for it to work properly?
   Ans: In order to preempt the resource when time is up for printf/fflush from other running tasks, like matrix_task.

3. What happens to the completion time of "matrix_task" when the priority of "communicationtask" is increased?
   Ans: Here I assume that the “completion time” is the time to run one loop. When communication_task has higher priority, it means matrix_task has to be interrupted when communication_task needs the resource (CPU). So matrix_task will spend more time on context switch. It means completion time will increase.

4. How many seconds is the period of "matrixtask"? (Hint: look at vApplicationTickHook() to measure it)
   Ans: Here I have implemented the function in matrix_task to show the period. Here is the transformation before and after the execution time of communication_task > 1000ms. Around “Communication time: 1107” is printed, the priorityset_task already changed the priority of communication_task, and the period of matrix_task (Matrix time) increases from ~480 to ~550.



```
Matrix Time: 474
Communication Time: 954
Sending data...
Matrix Time: 480
Data sent!
Matrix Time: 463
Communication Time: 942
Sending data...
Matrix Time: 479
Data sent!
Matrix Time: 540
Communication Time: 1107
Sending data...
Data sent!
Matrix Time: 567
Communication Time: 200
Sending data...
Data sent!
Communication Time: 200
Sending data...
Data sent!
Communication Time: 200
Sending data...
Matrix Time: 545
Data sent!
Communication Time: 200
Sending data...
Data sent!
Communication Time: 200
Sending data...
Data sent!
Matrix Time: 542
Communication Time: 200
Sending data...
```