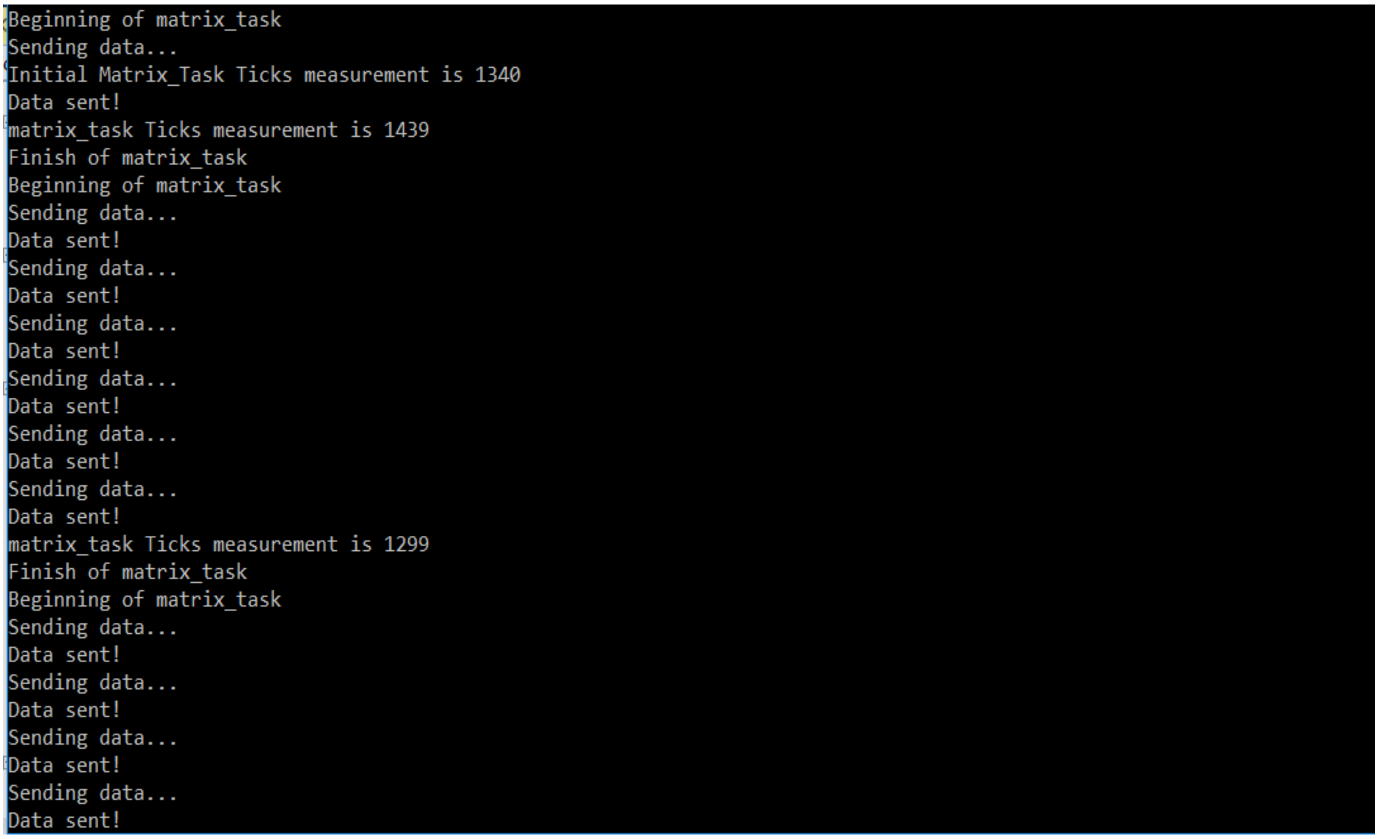
Development of Real-Time Systems – Assignment 2

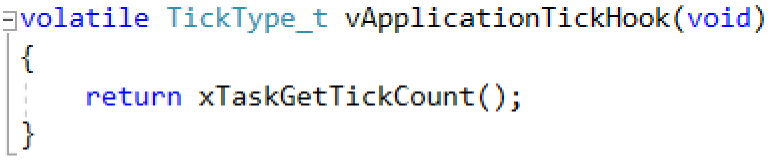
Screenshot of the Assignment-2 execution:



```
Beginning of matrix_task
Sending data...
Initial Matrix_Task Ticks measurement is 1340
Data sent!
matrix_task Ticks measurement is 1439
Finish of matrix_task
Beginning of matrix_task
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
matrix_task Ticks measurement is 1299
Finish of matrix_task
Beginning of matrix_task
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
Sending data...
Data sent!
```

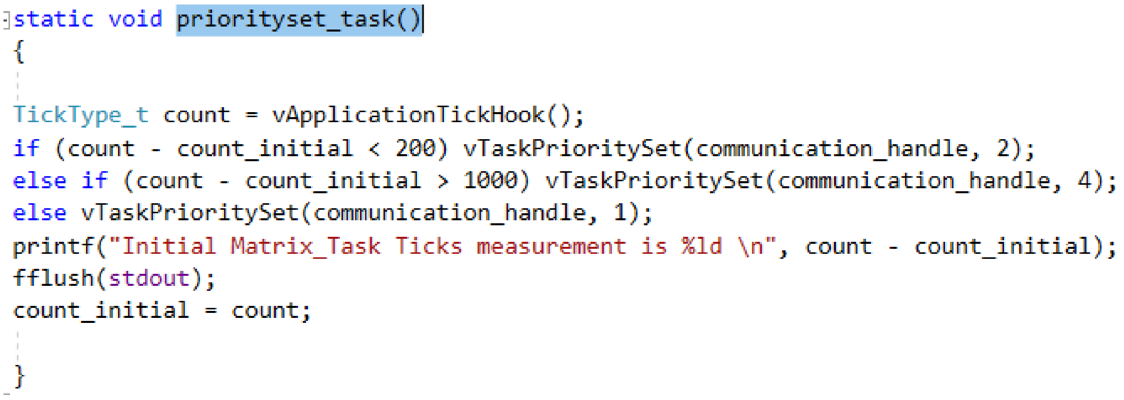
The above is the output of the tasks execution where the focus is on optimising the execution of tasks setting their priorities based on the execution period of the high intensity task.

The output informs that right when the scheduler task runs it looks for the highest priority task. The highest priority task, Matrix task which is indicated by 1st statement "**Beginning of matrix_task**". Meanwhile, the data is sent from the other created task which communication task which is indicated by 2nd statement "**Sending data**". And then the period of Matrix_task is measured by the below subroutine.



```
volatile TickType_t vApplicationTickHook(void)
{
    return xTaskGetTickCount();
}
```

The resultant measurement value of matrix_task as we could see in the screenshot is 1340 ms. Then using `priorityset_task()` the priority of the communication task is changed accordingly as in the task to get executed every 200ms.



```
static void priorityset_task()
{

TickType_t count = vApplicationTickHook();
if (count - count_initial < 200) vTaskPrioritySet(communication_handle, 2);
else if (count - count_initial > 1000) vTaskPrioritySet(communication_handle, 4);
else vTaskPrioritySet(communication_handle, 1);
printf("Initial Matrix_Task Ticks measurement is %ld \n", count - count_initial);
fflush(stdout);
count_initial = count;

}
```

Following to which the communication_task is executed every 200ms, in co-ordination with the matrix_task i.e in screenshot as it be could be seen, it keeps flooding with "**Data sent!, Sending data…** ".

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

A : As the task involves many for loops and most of the time is consumed for the matrix operations, the CPU is utilized for a quite longer time relative to any other task which do not contain the said parameters.

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

A : Otherwise, it may miss its deadlines at some point of execution which is where in principle is a catastrophic effect in a real time environment prolonging the task execution could be a major impact.

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

A : Completion time gets prolonged.

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

A : On an average (approximation) 1439 milliseconds.