Coursera's
Development of Real-Time Systems
Peer-graded Assignment: Assignment 2

Requirement

- "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.

- 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)

Task Source Code

1- Update Communication Task

```
static void communication_task()
{
    while (1) {
        printf("Communication Task \n");
        fflush(stdout);

        CommunicationTask = TRUE;
        CommunicationTaskTick = 0;

        printf("Sending data...\n");
        fflush(stdout);
        vTaskDelay(100);
        printf("Data sent!\n");
        fflush(stdout);
        vTaskDelay(100);
        fflush(stdout);

        CommunicationTask = FALSE;
        printf("Communication Task Execution Time = %i\n",CommunicationTaskTick * portTICK_PERIOD_MS);
        fflush(stdout);
    }
}
```

2- Implement Priority Task

```
static void prioritysettask()
{
    static int CommunicationTaskExecutionTime  = 0;
    CommunicationTaskExecutionTime = CommunicationTaskTick * portTICK_PERIOD_MS;
    printf("Priority Task\n");
    fflush(stdout);
    while (1)
    {
        CommunicationTaskExecutionTime  = CommunicationTaskTick * portTICK_PERIOD_MS;
        if( CommunicationTaskExecutionTime > 1000 )
            {
            /* Sets the priority of "communicationtask" to 4  */
                vTaskPrioritySet(communication_handle, HighestCommunicationPriority);
                printf("Communication Task Priority = 4\n");
                fflush(stdout);
            }
            else if( CommunicationTaskExecutionTime < 200 )
            {
                /* Sets the priority of "communicationtask" to 2 */
                vTaskPrioritySet(communication_handle, LowestCommunicationPriority);
                printf("Communication Task Priority = 2\n");
                fflush(stdout);
            }
        vTaskDelay(1000);
    }
```

3- Update vApplicationTickHook Function

```
void vApplicationTickHook( void )
{
    /* This function will be called by each tick interrupt if
    configUSE_TICK_HOOK is set to 1 in FreeRTOSConfig.h.  User code can be
    added here, but the tick hook is called from an interrupt context, so
    code must not attempt to block, and only the interrupt safe FreeRTOS API
    functions can be used (those that end in FromISR()). */
       if(CommunicationTask)
        {
            CommunicationTaskTick++;
        }
       if(MatrixTask)
        {
            MatrixTaskTick++;
        }
```

The Output of the Task

```
Priority Task
Communication Task Priority = 2
Communication Task Priority = 2
Matrix Task Execution Time = 1282 ms
Communication Task
Sending data...
Matrix Task Execution Time = 1277 ms
Data sent!
Communication Task Priority = 4
Communication Task Execution Time = 1718
Communication Task
Sending data...
Data sent!
Communication Task Execution Time = 200
Communication Task
Sending data...
Data sent!
Communication Task Execution Time = 200
Communication Task
Sending data...
Data sent!
Communication Task Execution Time = 200
Communication Task
Sending data...
Data sent!
Communication Task Execution Time = 200
Communication Task
Sending data...
Data sent!
```

- **Why is "matrixtask" using most of the CPU utilization?**
    - **Because of it's higher priority and has many resources**
- **Why must the priority of "communicationtask" increase in order for it to work properly ?**
    - As the matrix Task is Blocking it as matrix task has the highest priority.
- **What happens to the completion time of "matrixtask" when the priority of "communicationtask" is increased?**
    - Very little difference
- **How many seconds is the period of "matrixtask"? (Hint: look at vApplicationTickHook() to measure it)**
    - 1281 ms