Communication And Electronics Department

- -

Embedded Systems (‘’ ELC2080 ‘’)

**RTOS Communicating Tasks Project**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Name | Section | B.N. | Group |
| #1 | **مريم طاهر عبدالعزيز محمد** | **4** | **09** | **125** |
| #2 | **هايدي احمد حداد محمد اسماعيل** | **4** | **32** | **125** |

**Submitted by :**

# System Design

**1.1 List of main structures and operations:**

**1.1.1 Sender task 1 & Sender taks2 :**

They send a message containing current time in system ticks to the receiver. Each sender task sleeps for a RANDOM period of time Tsender ,and it wakes up it sends a message to the queue. The message is either blocked or sent depending on the queue cells. If the queue is full the message is blocked and if there is at least an empty cell in the queue, the message is sent successfully. This task can only be executed when it takes the sender semaphore which is released through the call back function of the sender timer.

**1.1.2 Receiver task:**

It receives a message containing current time in system ticks from the sender. It checks whether the queue is empty or not, so if the queue is empty the receiver tasks sleeps and if the queue is not empty the task receives the message at the front of the queue and print it on the screen. When the received messages =500 the Reset function is called. This task can only be executed when it takes the receiver semaphore which is released through the call back function of the receiver timer.

**1.1.3 Sender Timer:**

It is a periodic timer whose call back function is to release the receiver semaphore which enables the receiver task to take the semaphore and do its job. The period of this timer changes every time we call the Reset function. We implement a function called Random controlling the sender timer period Tsender to the next values in two arrays specifying the lower and upper bound values of the uniformly distributed timer period. The first array holds the values {50, 80, 110, 140, 170, 200} and the second holds the values {150, 200, 250, 300, 350, 400} expressing in msec the timer lower and upper bounds for a uniform distribution. When the system starts initially it starts with the values 50 and 150 , we use the function “ srand(time(NULL)) “ and using the formula “Tsender=((rand()%( upper -lower +1))+lower) ; “.

**1.1.4 Receiver Timer:**

It is a periodic timer of a certain period (100mSec) whose call back function is to release the receiver semaphore which enables the receiver task to take the semaphore and do its job.

**1.1.5 Sender semaphore 1&2:**

The sender task is to take this semaphore to perform its task and give it after doing the instructions .

**1.1.6 Receiver semaphore:**

The receiver task is to take this semaphore to perform its task and give it after doing the instructions .

**1.1.7 The Queue:**

There is only one queue to which the sender task sends messages and from which the receiver task receives messages. The size of this queue affects the number of blocked and successfully sent messages.

**1.1.8 Reset function:**

This function prints the number of blocked and successfully sent messages then it resets them. It also resets the queue and makes it empty. It changes the period of the sender timer every time we call it through the uniformly distributed timer period using the Random function . And when all values in the array are used, it deletes the timers and prints a message “Game Over”.

**1.1.9 Main function:**

In this function we call the Reset function for the first time. We also create semaphores, timers, tasks and the queue. Then we call the scheduler to start executing.

**1.2 Design of structures and Flow of the operations:**

Is queue fill ?

Yes

Have semaphore ?

No

No

Sent at the

tail

Receive at

the front

Yes ( Blocked )

No

Yes

No

Is queue empty ?

Have semaphore ?

Yes

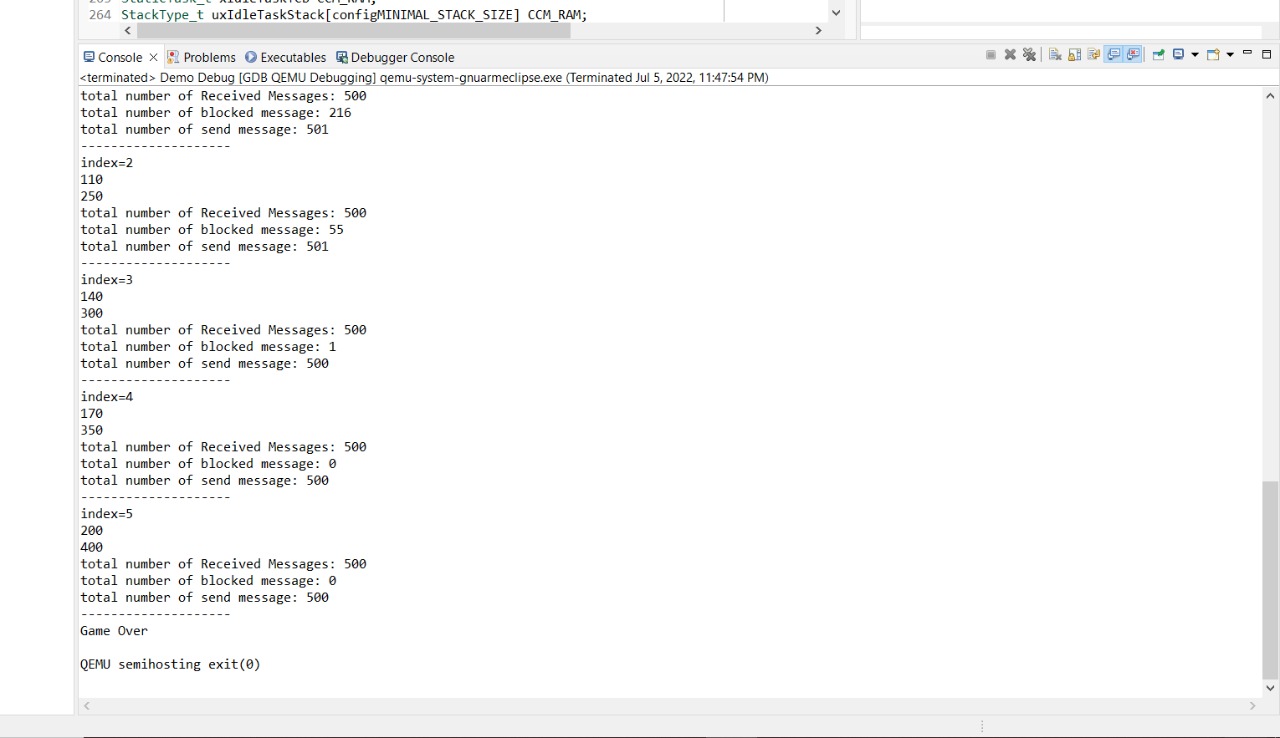
\*\*And remain in this loop for 500 messages with random period for\*\*

each send then Reset

# Results and Discussion

|  |  |  |
| --- | --- | --- |
| Lower bound | Upper bound | Avg |
| 50 | 150 | 100 |
| 80 | 200 | 140 |
| 110 | 250 | 180 |
| 140 | 300 | 220 |
| 170 | 350 | 260 |
| 200 | 400 | 300 |

**2.1 Results in case that size of the queue=2:**



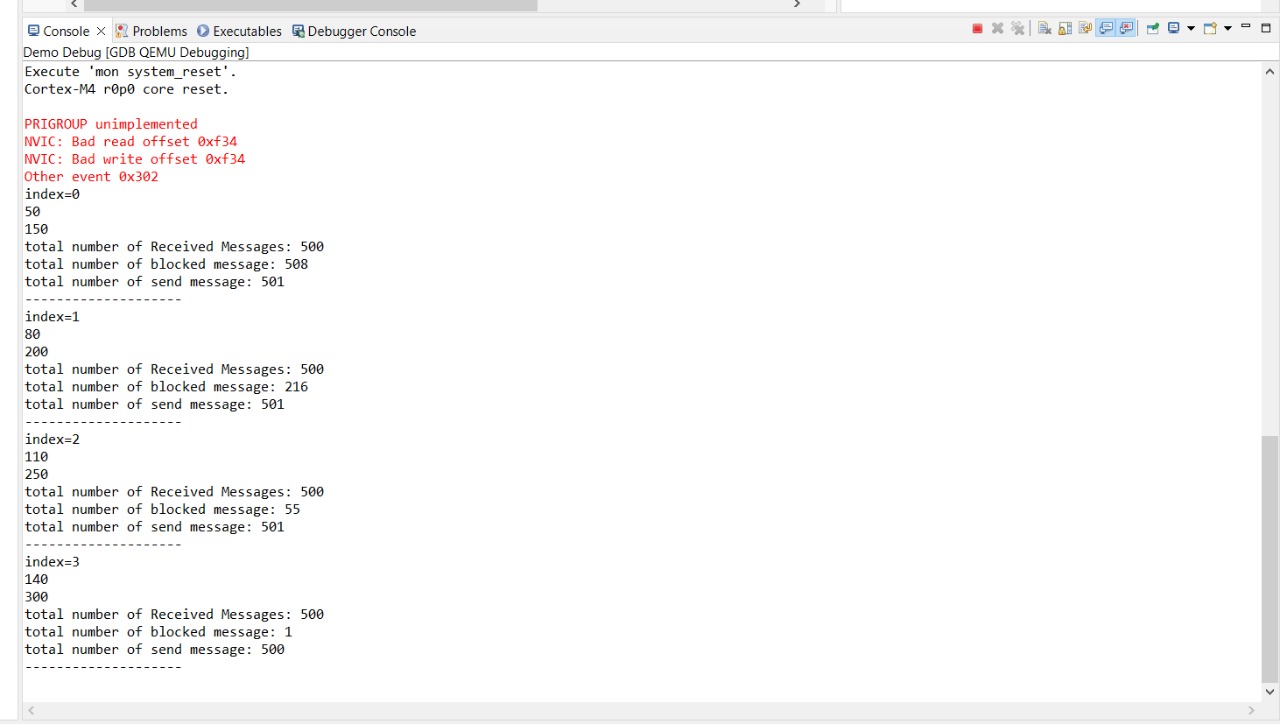
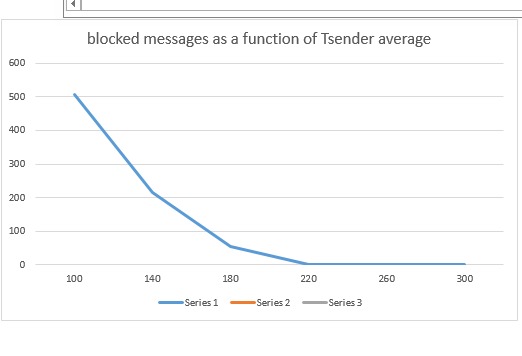


Table1: Number of messages at queue of size=2 -The

-The number of blocked messages as function of Tsender: -The number of successfully sent messages as function of Tsender



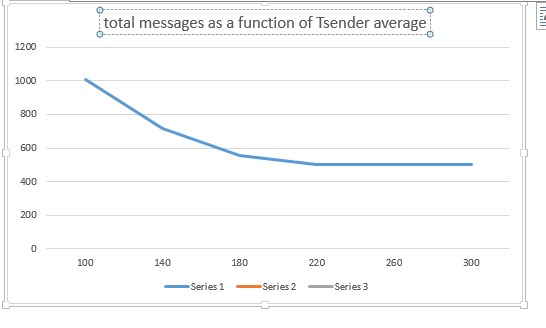


Figure1: blocked messages as function of Tsinder Figure2: totalsent messages as function of Tsinder

**2.2 Results in case that size of the queue=20:**

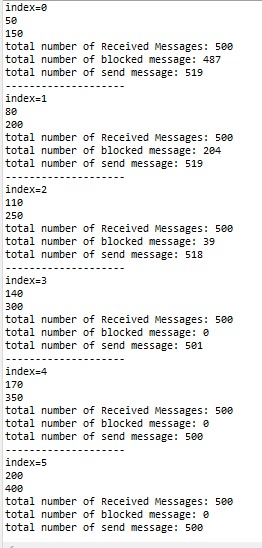
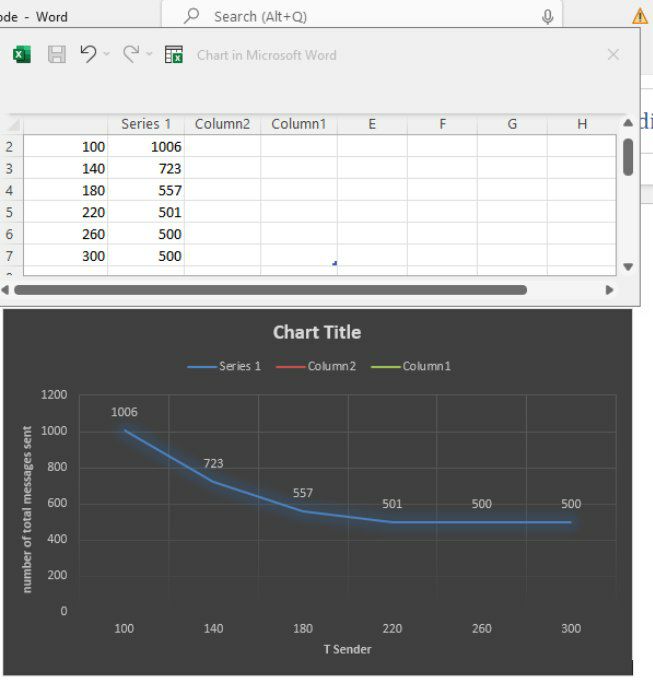


Table2: Number of messages at queue of size=20



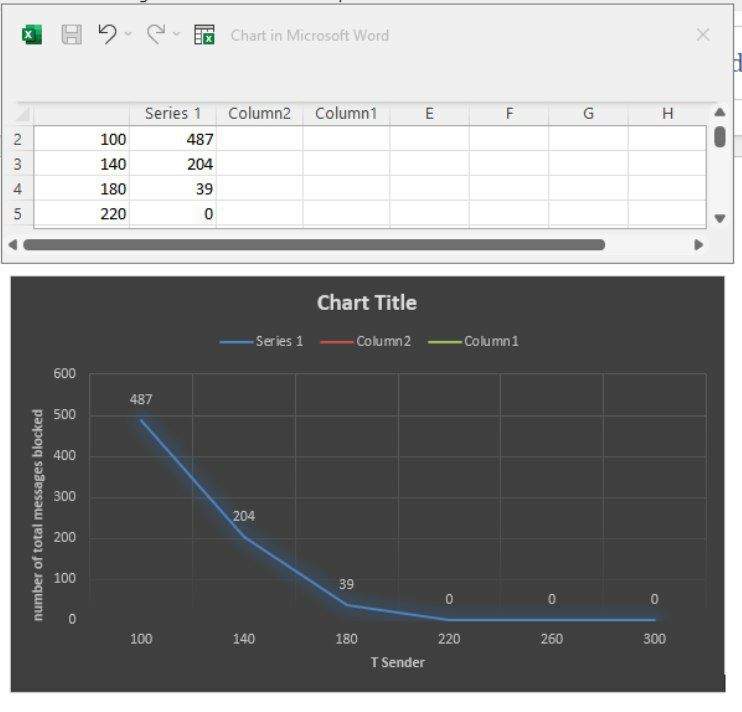


Figure3: blocked messages as function of Tsender Figure4: total sent messages as function of Tsender

# References

[1] Reference Manual for FreeRTOS version 9.0.0 issue 2. Real Time Engineers Ltd. 2016.