اسم الطالب	رقم الجلوس	الفصل	رقم المقعد	رقم المجموعة على البلاكبورد
سهر عادل على يوسف	32076	2	21	6
محمد خالد محمد الاحمدى	32143	3	33	

1 System Design

- We can conclude our design by some points:
  - There are two senders have access to send messages via queue, and one receiver to read messages that successfully sent.
  - Two senders send message with timers with uniformly distribution random period that determines when sender tasks wake to send message and sleep again, similarly receiver task but with fixed period.
    We used for random period generation this expression by knowing lower and upper limits
    $$Random\ Period = Lower\ Limit + rand()\%(Upper\ Limit - Lower\ Limit + 1)$$
    as we know modulus operator will return values from 0 to Upper Limit – Lower Limit by adding Lower Limit to this range will get range we want which is [Lower Limit : Upper Limit].
  - Using binary semaphores to protect shared resource which are
    - Total no.of transmitted messages
    - Total no.of blocked messages
    - Total no.of received messages
  - As we see in figure [1] which represents implementation of sender task, its function to send message "Time is XYZ", where XYZ represent number of ticks, we used **xTaskGetTickCount** to get it, then ask to take semaphore to able to send message to queue and give it access to update number of transmitted and blocked messages.
  - As we see in figure [2] which represents implementation of receiver task, its function to read message from queue by asking to take semaphore, which give it access to update number of received messages.

```
void MySender1(void *p)
{
    char msg_to_send[30] = "Time is ";
    while (1)
    {
        str_cat_num(msg_to_send, xTaskGetTickCount());
        if(xSemaphoreTake(MySemapore_Sender,portMAX_DELAY))
        {
            Send_message_via_Queue(MyQueue, msg_Send, msg_to_send);
            msg_to_send[8] = '\0';
            xSemaphoreGive(MySemapore_Sender);
        }
        vTaskDelay(pdMS_TO_TICKS(Random_period_sender1));
    }
}
```

Figure 1: Sender Task

```
void MyReceiver(void *p)
{
    char buffer[30];
    int test = 0;
    while (1)
    {
        if(xSemaphoreTake(MySemapore_Reciever,portMAX_DELAY))
        {
            if (xQueueReceive(MyQueue, buffer, (TickType_t)1000))
            {
                printf("Data Received %s\n", buffer);
                no_received_messages++;
            }
            else
                printf("Queue is empty\n");
            xSemaphoreGive(MySemapore_Reciever);
        }
        vTaskDelay(pdMS_TO_TICKS(100));
    }
}
```

Figure 2: Receiver Task

- Following flow chart describe the sequence of the program

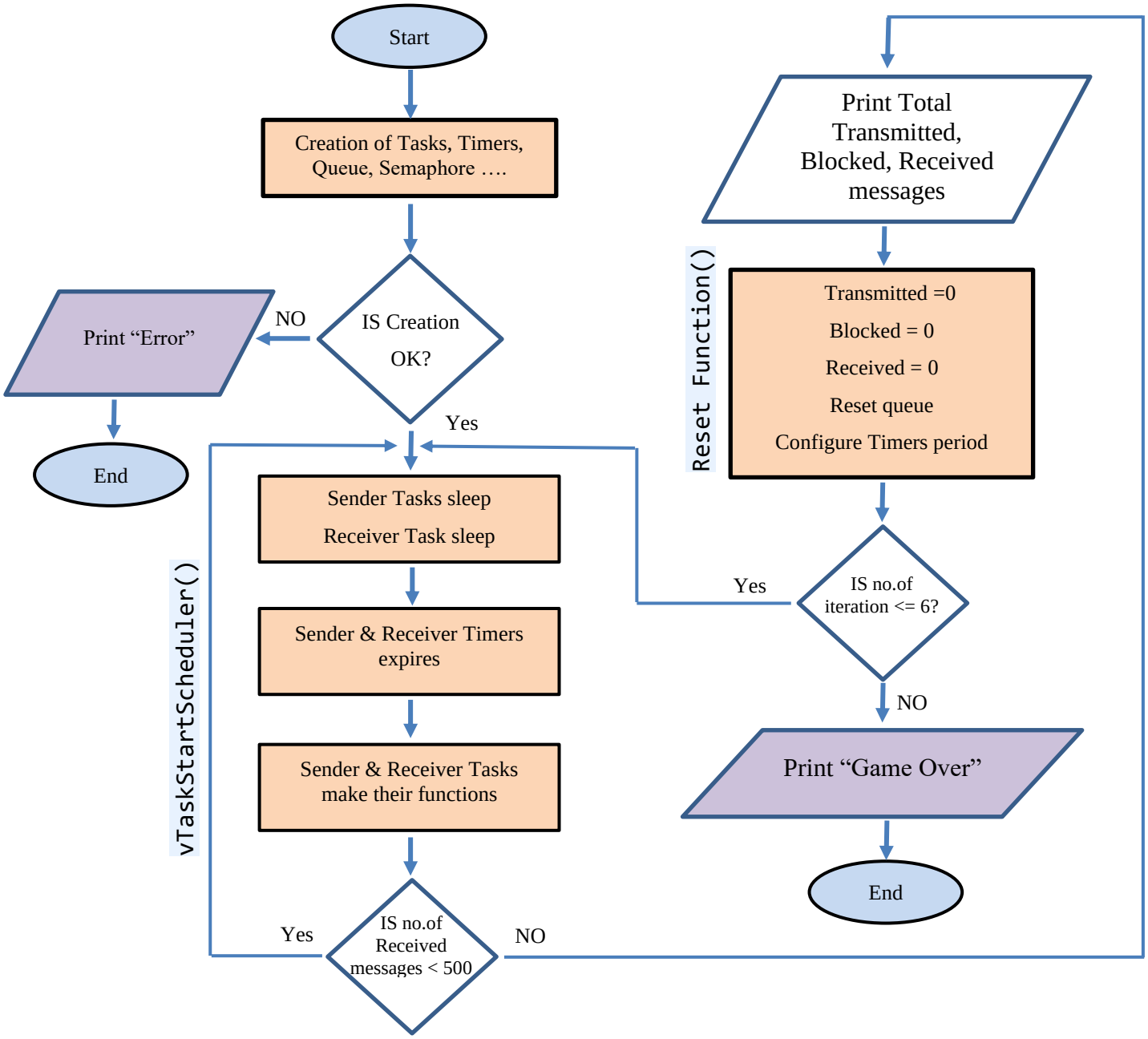


Figure 3: Flow Chart for Design

2 Results and Discussion

- After program run, we get this output which describe total number of transmitted, blocked, received messages in every range and get output as a text file to able to open it in excel for plotting this data.
- As we see in figure [4] from result as expected, when Tsender period increase, no of blocked messages decrease due to receiver able to read data via queue faster, and at specific time blocked messages becomes zero which mean all data I send is read by receiver.
- When using queue of size 20, we realize as expected in figure [5] that total number of blocked messages decrease, as queue take more time to full.

```
Range From 50 : 150
Total no.of Transmitted messages are: 501
Total no.of Blocked messages are: 498
Total no.of Received messages are: 500
------------------
Range From 80 : 200
Total no.of Transmitted messages are: 502
Total no.of Blocked messages are: 212
Total no.of Received messages are: 500
------------------
Range From 110 : 250
Total no.of Transmitted messages are: 501
Total no.of Blocked messages are: 56
Total no.of Received messages are: 500
------------------
Range From 140 : 300
Total no.of Transmitted messages are: 501
Total no.of Blocked messages are: 0
Total no.of Received messages are: 500
------------------
Range From 170 : 350
Total no.of Transmitted messages are: 500
Total no.of Blocked messages are: 0
Total no.of Received messages are: 500
------------------
Range From 200 : 400
Total no.of Transmitted messages are: 500
Total no.of Blocked messages are: 0
Total no.of Received messages are: 500
------------------
```

Figure 4: Outputs at size of queue

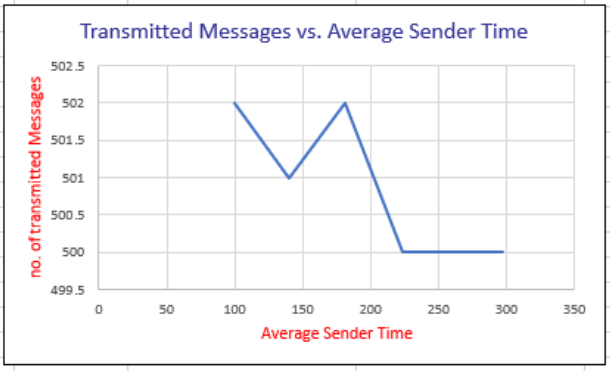
```
Range From 50 : 150
Total no.of Transmitted messages are: 520
Total no.of Blocked messages are: 468
Total no.of Received messages are: 500
------------------
Range From 80 : 200
Total no.of Transmitted messages are: 519
Total no.of Blocked messages are: 190
Total no.of Received messages are: 500
------------------
Range From 110 : 250
Total no.of Transmitted messages are: 520
Total no.of Blocked messages are: 38
Total no.of Received messages are: 500
------------------
Range From 140 : 300
Total no.of Transmitted messages are: 500
Total no.of Blocked messages are: 0
Total no.of Received messages are: 500
------------------
Range From 170 : 350
Total no.of Transmitted messages are: 500
Total no.of Blocked messages are: 0
Total no.of Received messages are: 500
------------------
Range From 200 : 400
Total no.of Transmitted messages are: 501
Total no.of Blocked messages are: 0
Total no.of Received messages are: 500
------------------
```

Figure 5: Outputs at size of queue is 20

❖ For plotting by using excel we get the following result:

Average Time	no of blocked messages	no of transmitted messages	no of recieved messages
99.68819374	499	502	500
139.8788732	210	501	500
181.284153	55	502	500
223.3863179	0	500	500
259.4641434	0	500	500
297.1556886	0	500	500

By using Queue with 2 cells



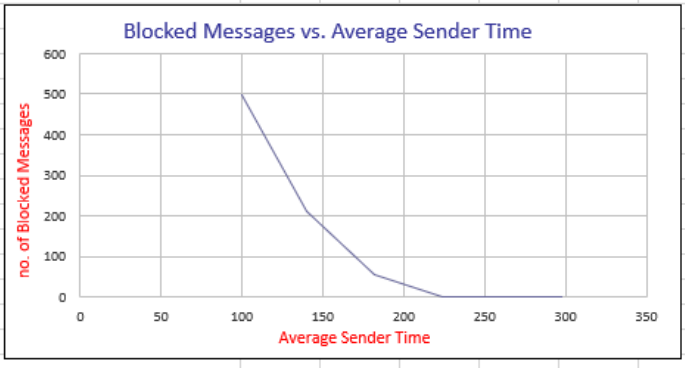
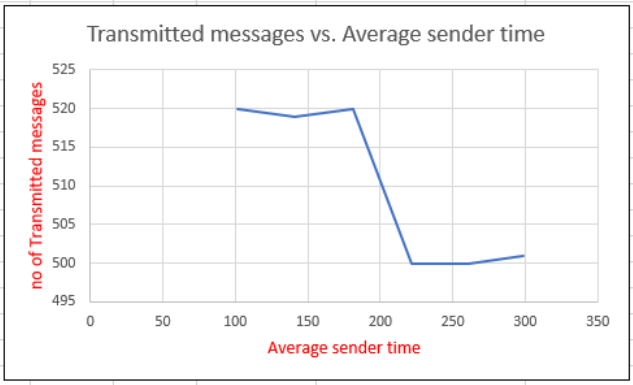


Figure 6: Plotting Sent & Blocked messages vs. Average Sender Time

Average time	no of blocked messages	no of transmitted messages	no of received messages
101.4835729	468	520	500
140.5714286	190	519	500
181.5447898	38	520	500
221.712	0	500	500
260.9356137	0	500	500
298.3767535	0	501	500

By using Queue with 20 cells



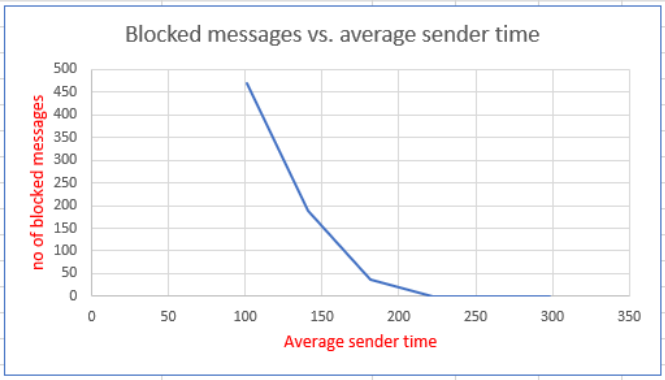


Figure 7: Plotting Sent & Blocked messages vs. Average Sender Time

- After visualizing data, difference become appear when using queue with size 20 instead of 2, as able to store more data in queue instead of blocked it.
- Note transmitted messages here in graph represent number of successfully messages stored via queue.
- The gap between sent and received messages is appeared due to difference in periodic time which each task wake to send or read data via queue which blocked messages represent this gap

3 Uploading Code

We upload all source code, output files, excel sheet in GitHub, this is repo for it.

4 References

[1] （Services，2020）FreeRTOS Documentation
[2] FreeRTOS Tutorial PDF Attached to project announcement.
[3] [IngeniørSnømann] FreeRTOSTutoria [Cited:10/2/2019].

4.1 Code Snippets

Here are code snippets (format as ProgCode)

```
We attached main.c with this document.
```