BN	Section	Student ID	اسم الطالب
31	**1**	**9210184**	**احمد يحيي محمد عبد الله**
45	**4**	**9211396**	**يوسف حسام سعد الدين محمد**

1 System Design

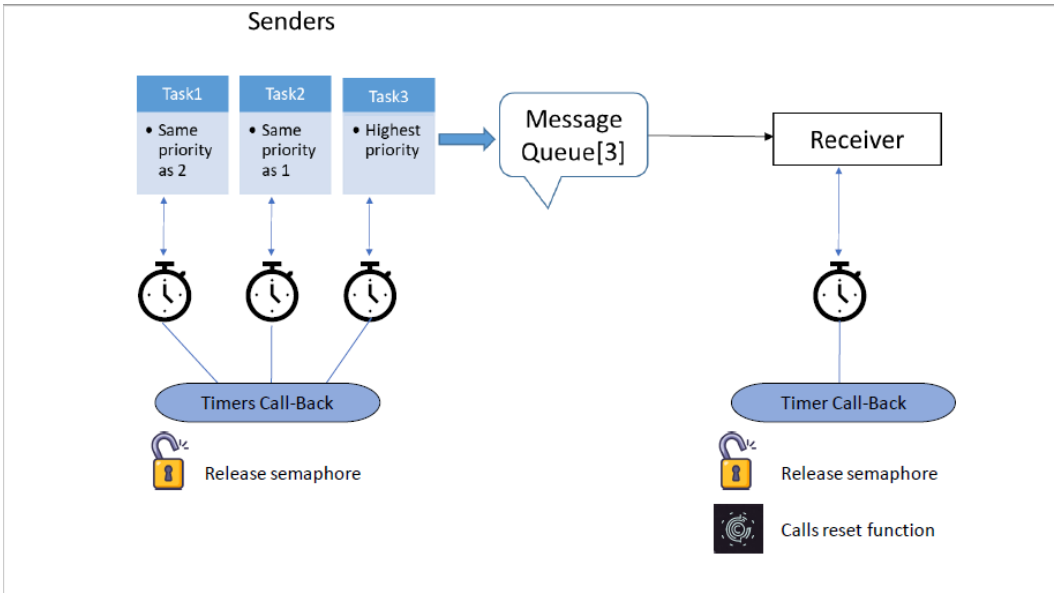


Figure 1: System Design

1.1 List the main structures

- **3-Sender tasks**
  The main function of the sender is send a message to receiver having the current time in msec. but the sequence is Sender - Queue - Receiver, so if Queue is Full the message is blocked and Blocked Message counter increase, else if there is an empty Cell in Queue the message is sent successfully and Successful Message counter increase. We cannot execute this function without calling "Sender timer callback function" for every sender.

- **Sender timer call-back function**
  The function allows the sender task to send to the queue by releasing a corresponding semaphore.

- **Receiver task**
  Receive the successful messages sent to the Queue which having the current time. When Number of received messages =1000, receiver Timer callback calls the Reset function.

- **Receiver timer call-back function**
  The function allows the receiver task to receive messages from the queue by releasing Receiver's semaphore.

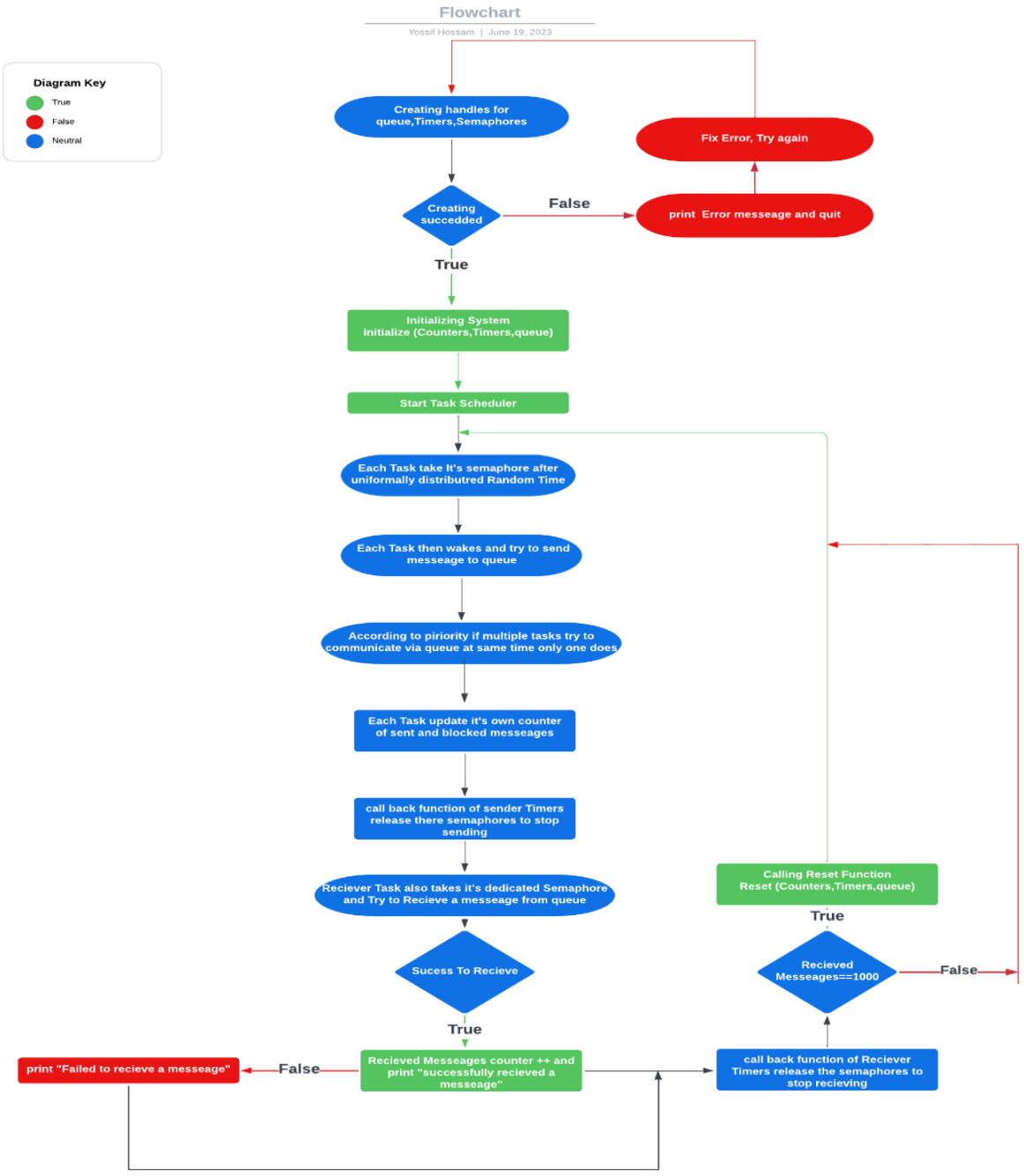
- **Queue**
  It is the link by which the sender tasks and the receiver task can communicate since each task is invisible to other. The queue size affects the number of blocked and transmitted messages.

- **Random function**
  It is used to generate uniformly distributed random numbers in every one of the six iterations between the 2 arrays: lower bound: {50, 80, 110, 140, 170, 200} & upper bound: {150, 200, 250, 300, 350, 400}.

- **Reset function**
  This function prints the number of blocked and successfully sent messages then it resets all of the counters and It also clears the communication queue.

- **Main function**
  In this function we create queue, tasks, semaphores and timers. Then we call initializing function, to initialize system and then start the task scheduler.

1.2 System Flow Chart



Figure(2) System Flow Chart

1.3 Pseudo Code Snippets

Sender Tasks

```
char SendMessaged[50];
while (1) {
  xSemaphoreTake(Semaphores , RandomTime_1);
    Status_s = xQueueSend(messageQueue, &SendMessaged, 0);
    if (Status_s == pdPASS) //check if queue is full
    {
    printf("success to send message from sender_1 \n");
    TransmittedCount_1++;
    }
    else
    {
    printf("failed to send message from sender_1 \n");
    BlockedCount_1++;
    }
    RandomSample();
    xTimerChangePeriod(xTimerHandlerSender_1,RandomTime_1,NULL );
```

This part is a pseudo code representing the sender tasks which are controlled by semaphores to allow each sender to send message to queue in random time that generated from random function. The semaphore will be available by call back functions.

Receiver Task

```
void ReceiverTask (void *s)
{ char RecievedMessaged[50];
while(1)
{ xSemaphoreTake(Semaphores, RecieverWait);
    Status_r = xQueueReceive( messageQueue,&RecievedMessaged, 0 );
     if (Status_r == pdPASS) {
          printf("success to receive message \n");
          RecieverCount++;
        }
        else
        {
          printf("failed to receive message\n"  );
        }
```

This part is a pseudo code representing the receiver task which is controlled by semaphores to allow receiver to receive message from queue in fixed time. The semaphore will be available by call back function as sender tasks.

Random Sample Function

```
void RandomSample() {
    srand(time(NULL)); //initialize the random number


    difference = UpperBound[Iterations] - LowerBound[Iterations] +1;
    RandomValue1 = (double)rand() / RAND_MAX ;
          //get a random value between 0 and 1
    Random = (int)(RandomValue1*difference) + LowerBound[Iterations];


    RandomTime_1 = pdMS_TO_TICKS ( Random );
```

Timer Call-back Function

```
void Sender_TimerCallback( TimerHandle_t xTimer  ) {   xSemaphoreGive( Semaphores); }

void RecieverTimerCallback(TimerHandle_t xTimer  ) {  xSemaphoreGive( Semaphores);
   if (RecieverCount ==1000)
        {
        inti_reset();
        }
}
```

Reset Function

```
void inti_reset(void)
{if( RecieverCount != 0 ) // called from main for the first time
    {    printf( "Total Sent Messages: %d,Total blocked Messages: %d\n",
          printf( "Sender 1 sent messages: %d, blocked messages: %d\n",
          printf( "Sender 2 sent messages: %d, blocked messages: %d\n",
          printf( "Sender 3 sent messages: %d, blocked messages: %d\n",
       Iterations++;}
  printf(  "loop_iteration ");
    //reseting counters
  TransmittedCount_total=0;
  BlockedCount_total=0;
  BlockedCount_1 =0 ;
  BlockedCount_2 =0 ;
  BlockedCount_3 =0 ;
  TransmittedCount_1 =0 ;
  TransmittedCount_2 =0 ;
  TransmittedCount_3 =0 ;
  RecieverCount =0;
  // clearing the queue
  xQueueReset(messageQueue);
}
```

2 Results and Discussion

2.1 output of console

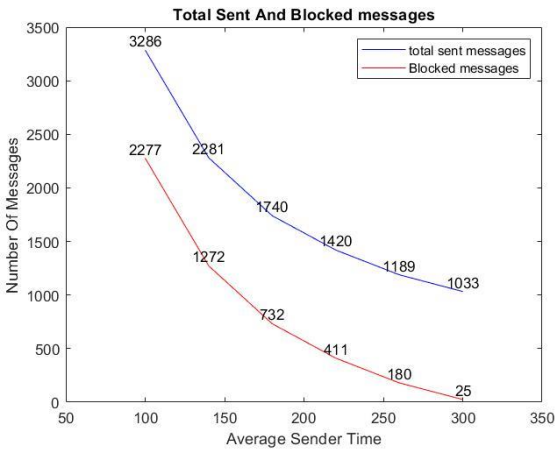
Queue of size 3

```
         Iteration number: 1
Total Sent Messages: 1002,Total blocked Messages: 2277
Sender 1 sent messages: 336, blocked messages: 754
Sender 2 sent messages: 348, blocked messages: 745
Sender 3 sent messages: 318, blocked messages: 778
         Iteration number: 2
Total Sent Messages: 1002,Total blocked Messages: 1285
Sender 1 sent messages: 354, blocked messages: 409
Sender 2 sent messages: 318, blocked messages: 442
Sender 3 sent messages: 330, blocked messages: 434
         Iteration number: 3
Total Sent Messages: 1002,Total blocked Messages: 747
Sender 1 sent messages: 339, blocked messages: 243
Sender 2 sent messages: 322, blocked messages: 259
Sender 3 sent messages: 341, blocked messages: 245
         Iteration number: 4
Total Sent Messages: 1001,Total blocked Messages: 422
Sender 1 sent messages: 326, blocked messages: 148
Sender 2 sent messages: 333, blocked messages: 142
Sender 3 sent messages: 342, blocked messages: 132
         Iteration number: 5
Total Sent Messages: 1001,Total blocked Messages: 189
Sender 1 sent messages: 337, blocked messages: 59
Sender 2 sent messages: 335, blocked messages: 60
Sender 3 sent messages: 329, blocked messages: 70
         Iteration number: 6
Total Sent Messages: 1002,Total blocked Messages: 49
Sender 1 sent messages: 336, blocked messages: 11
Sender 2 sent messages: 330, blocked messages: 22
Sender 3 sent messages: 336, blocked messages: 16
Game Over
```

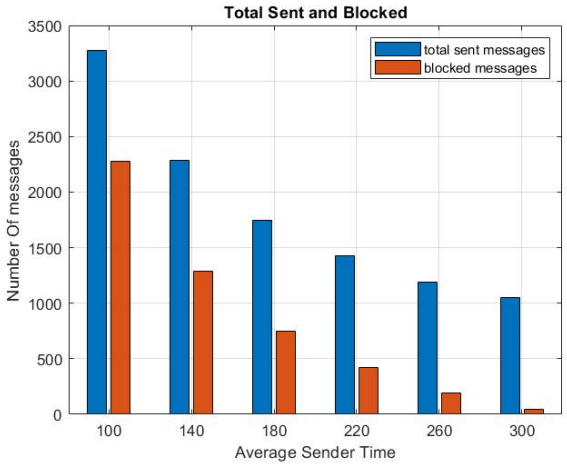
Queue of size 10

```
         Iteration number: 1
Total Sent Messages: 1009,Total blocked Messages: 2277
Sender 1 sent messages: 360, blocked messages: 729
Sender 2 sent messages: 325, blocked messages: 766
Sender 3 sent messages: 324, blocked messages: 782
         Iteration number: 2
Total Sent Messages: 1009,Total blocked Messages: 1272
Sender 1 sent messages: 336, blocked messages: 426
Sender 2 sent messages: 350, blocked messages: 412
Sender 3 sent messages: 323, blocked messages: 434
         Iteration number: 3
Total Sent Messages: 1008,Total blocked Messages: 732
Sender 1 sent messages: 341, blocked messages: 236
Sender 2 sent messages: 334, blocked messages: 247
Sender 3 sent messages: 333, blocked messages: 249
         Iteration number: 4
Total Sent Messages: 1009,Total blocked Messages: 411
Sender 1 sent messages: 328, blocked messages: 143
Sender 2 sent messages: 340, blocked messages: 135
Sender 3 sent messages: 341, blocked messages: 133
         Iteration number: 5
Total Sent Messages: 1009,Total blocked Messages: 180
Sender 1 sent messages: 337, blocked messages: 58
Sender 2 sent messages: 333, blocked messages: 63
Sender 3 sent messages: 339, blocked messages: 59
         Iteration number: 6
Total Sent Messages: 1008,Total blocked Messages: 25
Sender 1 sent messages: 336, blocked messages: 8
Sender 2 sent messages: 337, blocked messages: 7
Sender 3 sent messages: 335, blocked messages: 10
Game Over
```

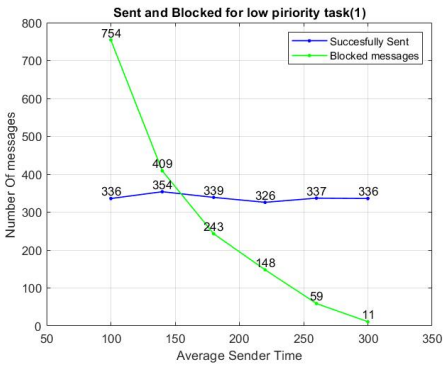
2.2 Graphs when Queue is of size 3

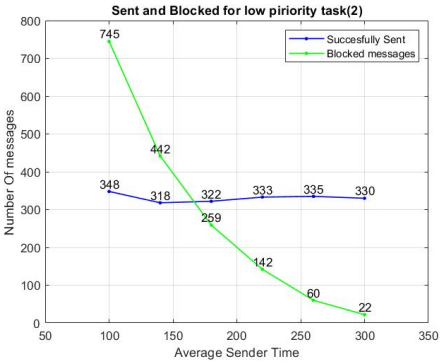


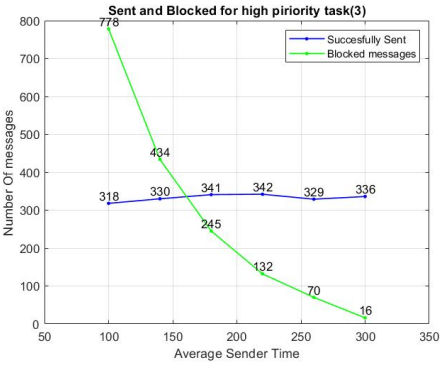
Figure(3): plot of (Total Sent Vs Blocked)



Figure(4): Sender tasks sent and blocked messages





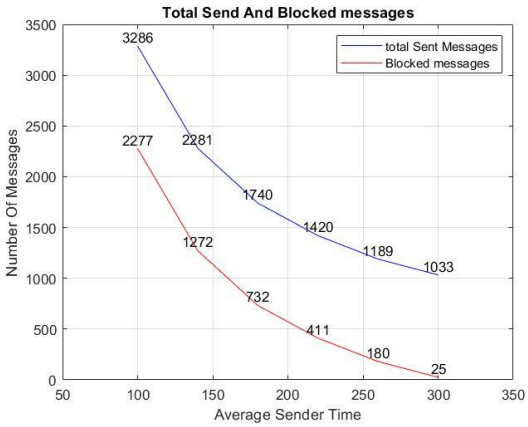


Figure(5): bar graph of (Total Sent Vs Blocked)

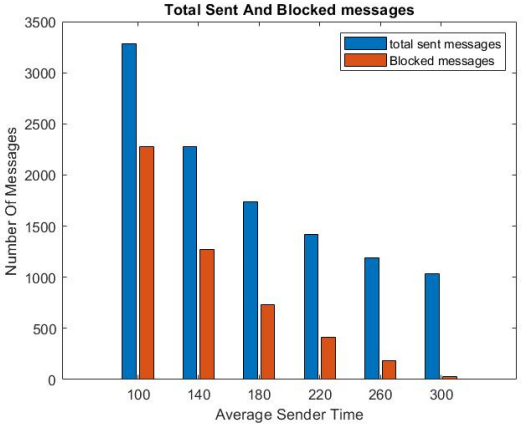
The gap between the number of sent and received messages?

As we see there is a large gap between total sent messages and blocked messages because we have one receiver which receives 1000 message only in one iteration while we have three senders that send more messages than that the receiver receives so part of them will blocked.

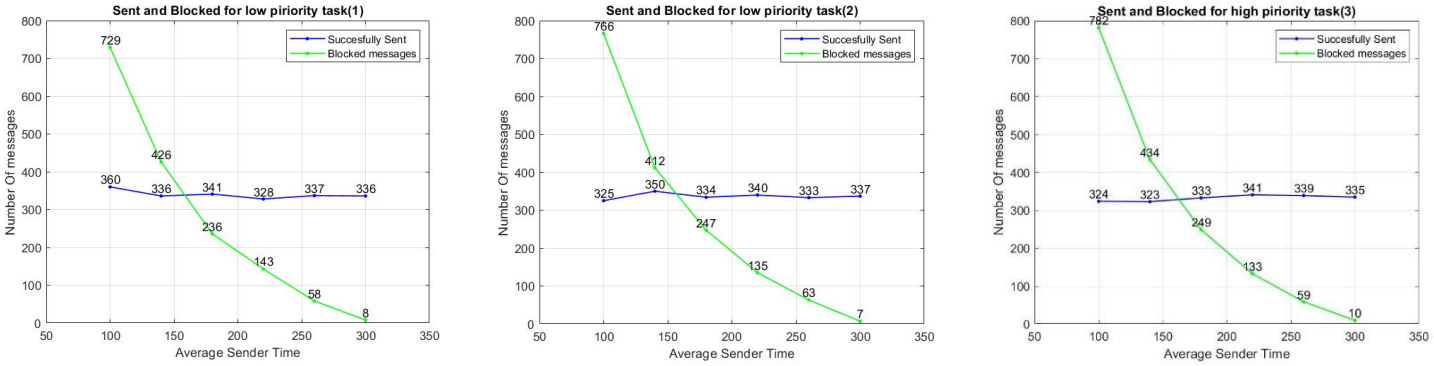
2.3 Graphs when Queue is of size 10



Figure(5): plot of (Total Sent Vs Blocked)



Figure(6): bar graph of (Total Sent Vs Blocked)



Figure(7): Sender tasks sent and Blocked messages

What happens when queue size increases from 3 to 10?

From graphs we conclude that Increasing the size of communication queue, Increases the number of successfully sent messages and Decreases the number of Blocked messages, as it can store more number of messages without getting full allowing receiver task to be able to receive them.

References

[1] “FreeRTOS API categories.” https://www.freertos.org/a00106.html (accessed June. 16, 2023).
[2] D. E. Simon, "An Embedded Software Primer," Chapter 6: Real-Time Operating Systems, Pearson Education, 1999.