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

# 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

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.

Text

Description automatically generated

Figure 1: Sender Task

Graphical user interface, text, application

Description automatically generated

Figure 2: Receiver Task

* Following flow chart describe the sequence of the program

Print Total Transmitted, Blocked, Received messages

Creation of Tasks, Timers, Queue, Semaphore ….

Transmitted =0

Blocked = 0

Received = 0

Reset queue

Configure Timers period

IS Creation

OK?

Print “Error”

NO

Reset Function()

Yes

Sender Tasks sleep

Receiver Task sleep

IS no.of iteration <= 6?

Yes

Sender & Receiver Timers expires

vTaskStartScheduler()

NO

Print “Game Over”

Sender & Receiver Tasks make their functions

IS no.of Received messages < 500 ?

Yes

NO

Figure 3: Flow Chart for Design

# 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.



Figure 4: Outputs at size of queue is 2

Figure 5: Outputs at size of queue is 20

Figure 6: Numbers of messages in every range with Queue size 2



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

Table

Description automatically generated

Graphical user interface, chart, line chart

Description automatically generated

Figure 7: Plotting Sent & Transmitted & Blocked messages vs. Average Sender Time with Queue size 2

Table

Description automatically generated

Figure 8: Numbers of messages in every range with Queue size 20

Graphical user interface, chart, line chart

Description automatically generated

Figure 9: Plotting Sent & Transmitted & Blocked messages vs. Average Sender Time with Queue size 20

* 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

# Uploading Code

We upload all source code, output files, excel sheet in GitHub, this is [repo](https://github.com/mohamedkhaledalahmady/FreeRTOS_Project) for it.

# References

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

## Code Snippets

Here are code snippets (format as ProgCode)

We attached main.c with this document.