Lab Session - 9

Analysis of program with Queue

Name : Darvesh Naresh Patil
PRN : 2019BTEEN00035
Batch : EN-1
Sub. : RTOS

1) What is queue ?.

⇒ A queue is a facility provided by RTOS for communication between the 2 tasks.

2) Why do we need it ?

⇒ Mailbox facility fails in situations where messages arrive in cluster manner. As mailbox can hold only one message, other messages will be lost.

3) Steps to use queue

⇒ (i) Create a pointer to queue
(ii) Decide the size of array queue
(iii) Create an array for queue
(iv) Create char array for storing actual messages
(v) Create Queue in main using OSQCreate() Function
(vi) Sender task will post the messages to the queue using OSQPost() function
(vii) Receiver task will receive the messages using the OSQPend() function

Note : Receiver rate must be less than equal to the sending rate, else messages will be lost.

Queue Normal Operations

Code:

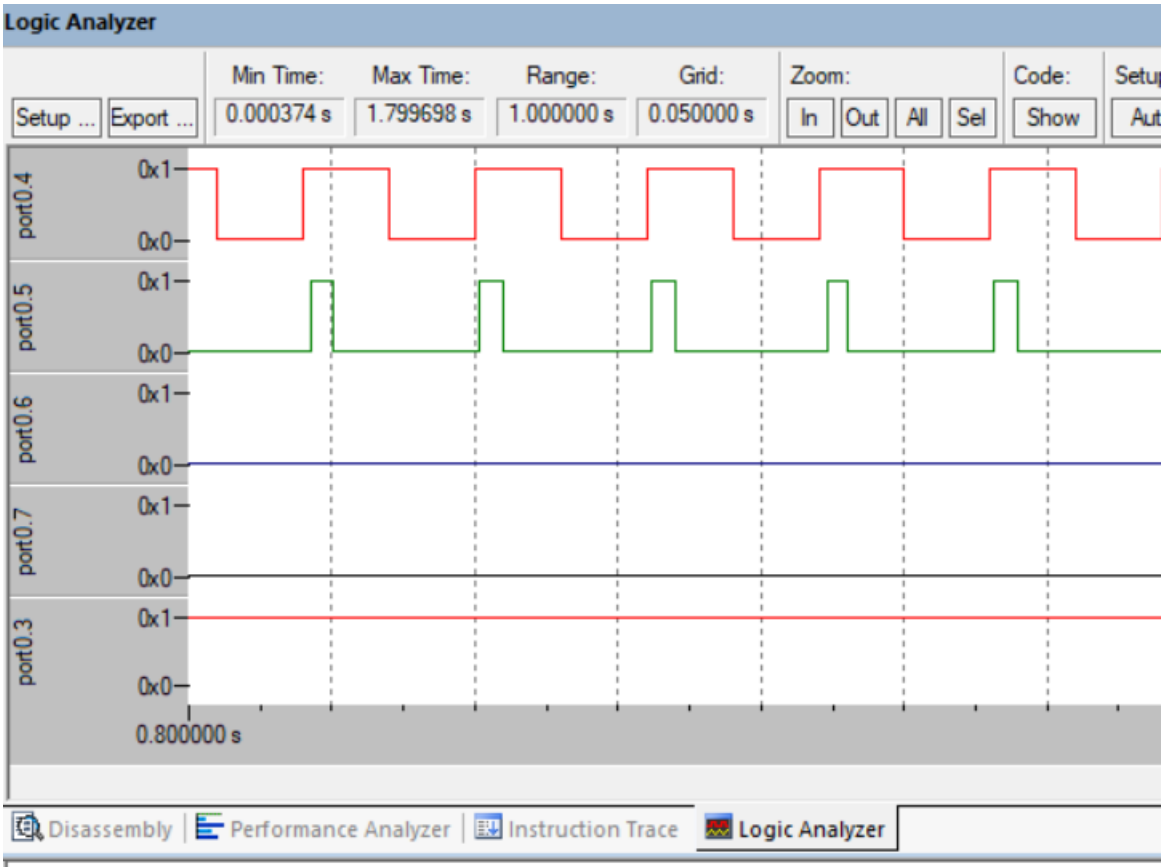
```
#include "config.h"
#include "stdlib.h"
#include <stdio.h>

#defineTaskStkLengh 64                          //Define the Task0 stack length

OS_STK        TaskStk0 [TaskStkLengh];          //Define the Task stack
OS_STK        TaskStk1 [TaskStkLengh];          //Define the Task stack
OS_STK        TaskStk2 [TaskStkLengh];          //Define the Task stack
OS_STK        TaskStk3 [TaskStkLengh];          //Define the Task stack

void     Task0(void *pdata);
void     Task1(void *pdata);
void     Task2(void *pdata);
void     Task3(void *pdata);


OS_EVENT* MsgQueue;

#define SIZE_OF_Q 150
// This is the actual queue
void* MessageStorage[SIZE_OF_Q];

uint8 err;

// Array for storing the message
char msg_1[25];
/*********************
          main()
*********************/
int main (void)
```

```
{
        LED_init();
        UART0_Init();

        TargetInit();
        OSInit ();

        // Create a Queue, Attach it to message storage are
        MsgQueue = OSQCreate(MessageStorage,SIZE_OF_Q);

        OSTaskCreate (Task0,(void *)0, &TaskStk0[TaskStkLengh - 1], 6);
        OSTaskCreate (Task1,(void *)0, &TaskStk1[TaskStkLengh - 1], 8);

        OSStart();
        return 0;

}
/*********************
**                      Task0
*********************/

void Task0      (void *pdata)
{

        pdata = pdata;                                          /* Dummy data */

        while(1)
        {


                sprintf(msg_1,"Hello\n");
                OSQPost(MsgQueue, msg_1);
                LED_on(0);
                OSTimeDly(3);
```

```
		LED_off(0);
		OSTimeDly(3);

		sprintf(msg_1,"Durvesh\n");
		OSQPost(MsgQueue, msg_1);
		LED_on(0);
		OSTimeDly(3);
		LED_off(0);
		OSTimeDly(3);

		sprintf(msg_1,"Good afternoon\n");
		OSQPost(MsgQueue, msg_1);
		LED_on(0);
		OSTimeDly(3);
		LED_off(0);
		OSTimeDly(3);

	}
}

void Task1	(void *pdata)
{
	char* ptr_c;
	pdata = pdata;					/* Dummy data */

	while(1)
	{
		// wait for message from queue
		ptr_c = OSQPend(MsgQueue,0, &err);
		UART0_SendData(msg_1);

		LED_on(1);
		OSTimeDly(1);
		LED_off(1);
```

```
            OSTimeDly(1);


        }
}
```

Observations:




```
Hello
Durvesh
Good afternoon
Hello
Durvesh
Good afternoon
Hello
Durvesh
Good afternoon
Hello
Durvesh
Good afternoon
Hello
Durvesh
Good afternoon
```

Receiving rate is lower than sending rate

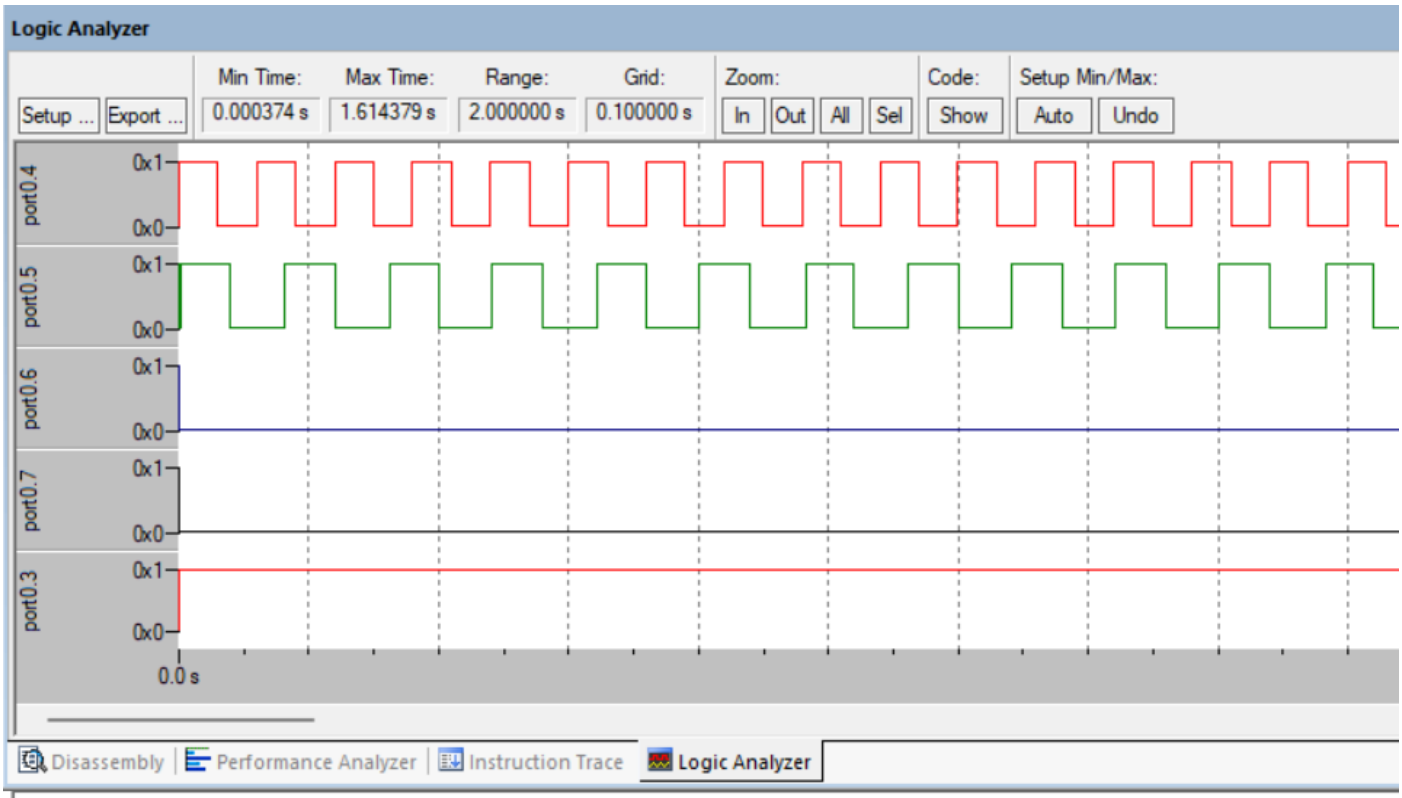
Code:

```
#include "config.h"
#include "stdlib.h"
#include <stdio.h>

#defineTaskStkLengh 64                    //Define the Task0 stack length

OS_STK        TaskStk0 [TaskStkLengh];    //Define the Task stack
OS_STK        TaskStk1 [TaskStkLengh];    //Define the Task stack
OS_STK        TaskStk2 [TaskStkLengh];    //Define the Task stack
OS_STK        TaskStk3 [TaskStkLengh];    //Define the Task stack

void     Task0(void *pdata);
void     Task1(void *pdata);
void     Task2(void *pdata);
void     Task3(void *pdata);


OS_EVENT* MsgQueue;

#define SIZE_OF_Q 150
// This is the actual queue
void* MessageStorage[SIZE_OF_Q];

uint8 err;

// Array for storing the message
char msg_1[25];
int main (void)
{
        LED_init();
        UART0_Init();
```

```
	TargetInit();
	OSInit ();

	// Create a Queue, Attach it to message storage are
	MsgQueue = OSQCreate(MessageStorage,SIZE_OF_Q);

	OSTaskCreate (Task0,(void *)0, &TaskStk0[TaskStkLengh - 1], 6);
	OSTaskCreate (Task1,(void *)0, &TaskStk1[TaskStkLengh - 1], 8);

	OSStart();
	return 0;

}
void Task0	(void *pdata)
{

	pdata = pdata;					/* Dummy data */

	while(1)
	{


		sprintf(msg_1,"Hello\n");
		OSQPost(MsgQueue, msg_1);
		LED_on(0);
		OSTimeDly(3);
		LED_off(0);
		OSTimeDly(3);

		sprintf(msg_1,"Durvesh\n");
		OSQPost(MsgQueue, msg_1);
		LED_on(0);
		OSTimeDly(3);
```

```
                LED_off(0);
                OSTimeDly(3);

                sprintf(msg_1,"Good afternoon\n");
                OSQPost(MsgQueue, msg_1);
                LED_on(0);
                OSTimeDly(3);
                LED_off(0);
                OSTimeDly(3);

        }
}

void Task1      (void *pdata)
{
        char* ptr_c;
        pdata = pdata;                                  /* Dummy data */

        while(1)
        {
                // wait for message from queue
                ptr_c = OSQPend(MsgQueue,0, &err);
                UART0_SendData(msg_1);

                LED_on(1);
                OSTimeDly(4);
                LED_off(1);
                OSTimeDly(4);

        }
}
```

Obsevations:




```
Hello
Durvesh
Good afternoon
Durvesh
Good afternoon
Hello
Good afternoon
Hello
Durvesh
Hello
Durvesh
Good afternoon
Durvesh
Good afternoon
Hello
```

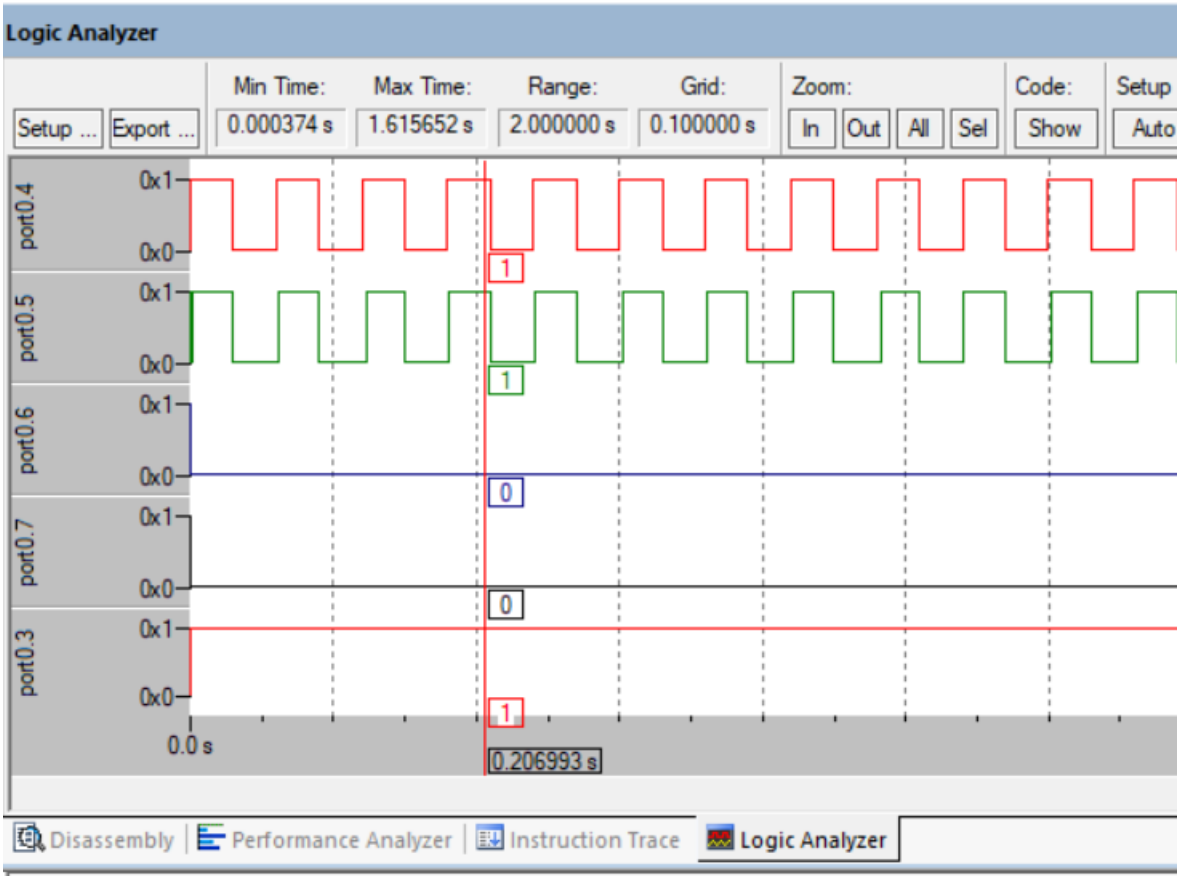
Comments:

Messages will be lost.

To avoid the above problem, we can:

1) Increase the queue size
2) Increase the receiving rate

Output after increasing the receiving rate:




```
Hello
Durvesh
Good afternoon
Hello
Durvesh
Good afternoon
Hello
Durvesh
Good afternoon
Hello
Durvesh
Good afternoon
Hello
Durvesh
Good afternoon
```

Conclusion :

(i) Like mailbox, queue is used to send data from one task to another task

(ii) Unlike mailbox, queue is able to send the clustered messages

(iii) We need to set proper queue size

(iv). If receiving delay is more than the sending ~~rate~~ delay then messages will be lost

i.e. Receiving rate must be greater than that of sending rate