# Assignment 4 – Task Synchronization with uCOS

EMBSYS 105 Winter 2020

## Due: 2/2/2020 11:59 PM

This assignment exercises most task synchronization services provided by uCOS, namely semaphores, mailboxes, queues, and event flags. Services such as these are commonly provided by kernels and it will be useful to become familiar with them in preparation for developing event driven applications.

1. Download and unzip the taskSync project contained in the zip file: taskSync.zip
2. Open the taskSync.eww workspace in the EWARM IDE.
3. Make sure the taskSync project builds. There will be a dozen or so warnings which will go away after you add missing code.
4. Launch TeraTerm
5. Build and upload the project and start it running.
6. It will not run very far until you substitute your uCOS port code from Assignment 3 into os_cpu_a.asm.
7. After adding your port code, run the app and you should see a jumble of status messages in the UART from the various tasks.
8. Your job is to add missing code to sort out the status messages so they don't interrupt each other and ensure that any reported errors are fixed.

The application consists of 7 tasks not including the uCOS startup task:

### Mailbox related tasks:

1. TaskMBTx is a task that uses mailboxes to transmit messages to two other tasks, TaskMBRxA and TaskMBRxB.
2. TaskMBRxA is a task that receives messages from TaskMBTx.
3. TaskMBRxB is another task that receives messages from TaskMBTx.

### Queue related tasks:

4. TaskQTxA is a task that uses a message queue to transmit messages to TaskQRx.
5. TaskQTxB, like TaskQTxA, also uses the message queue to transmit message to TaskQRx.
6. TaskQRx is a task that receives messages from TaskQTxA and TaskQTxB using a message queue.

### Event flag related task:

7. TaskRxFlags is a task that helps synchronize the two mailbox related receiver tasks, TaskMBRxA and TaskMBRxB so that they wait for each other to finish receiving one message before either one is permitted to receive the next message.

### What to Code:

Do a global search in the project for the string "TODO". This will find all the places you need to add code. Here is the suggested order for completing the assignment:

1. Follow the “TODO semPrint” instructions in the code to use a binary semaphore, i.e. the initial counter value is 1, to protect critical code so delays and interrupts don’t mix up the output from different tasks.
2. Follow the “TODO Mailbox” instructions in the code to send messages from TaskMBTx to TaskMBRxA and TaskMBRxB using uCOS mailboxes.
3. Follow the “TODO Queue” instructions in the code to send messages from TaskQTxA and TaskQTxB to TaskQRx using a uCOS queue.
4. Follow the “TODO EventFlags” instructions in the code to synchronize TaskMBRxA and TaskMBRxB using uCOS event flags so that neither one will attempt to receive their next message until both have received their current message.
5. You should get a “Done!” message from each task and all “expected” and “actual” results should match.

**Hint**

Comment out calls to OSTaskCreate() for tasks you are not currently focusing on.

**What to submit**

- Clean your resulting project (remove the debug folder)
- Zip your project in a file named taskSync_<YourUWNetId>.zip and submit it by the due date.