Freescale MQX RTOS Example Guide

lwsem example

This document explains the lwsem example, what to expect when running it and a brief introduction to the API.

The example

The lwsem example code shows how lightweight semaphores works. The code is written in a way that two different semaphores are synchronized to ensure mutual exclusion of a common memory space.

Running the example

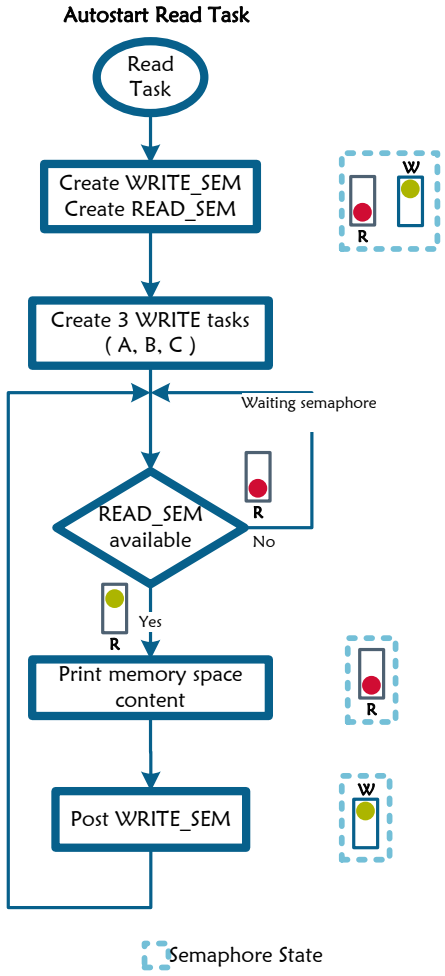
Start HyperTerminal on the PC (Start menu->Programs->Accessories->Communications). Make a connection to the serial port that is connected to the board (usually will be COM1).

Set it for 115200 baud, no parity, 8 bits and click OK.

Explaining the example

The light weight semaphores example uses four tasks:

- 1 read_task
- 3 write_task



read_task flow chart

The read_task starts by creating two lightweight semaphores (WRITE_SEM and READ_SEM) that govern the access to a data memory location (FIFO.Data). The WRITE_SEM starts enabled, while the REA_SEM is disabled.

```
Result = _lwsem_create(&fifo.READ_SEM, 0);
Result = _lwsem_create(&fifo.WRITE_SEM, 1);
```

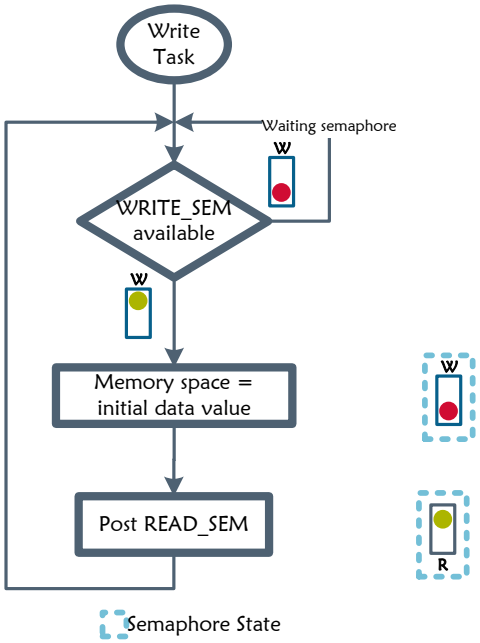
Parameters:
&fifo.READ_SEM = pointer to the lightweight semaphore to create
0 = Initial semaphore counter

Then 3 write task are created with initial_data equal to ‘A’, ‘B’, ‘C’ correspondingly.

If the READ_SEM is available (_lwsem_wait), the read_task prints on the HyperTerminal whatever the shared memory space contains.

```
Result = _lwsem_wait(&fifo.READ_SEM);
putchar('\n');
putchar(fifo.DATA);
```

Finally the WRITE_SEM is posted
_lwsem_post(&fifo.WRITE_SEM);



write_task flow chart

The write task verifies if the WRITE_SEM is available (_lwsem_wait), then writes at the shared memory space, task initial_value.

```
Result  = _lwsem_wait(&fifo.WRITE_SEM)
fifo.DATA = (uchar)initial_data;
```

Finally the READ_SEM is posted
```
_lwsem_post(&fifo.READ_SEM);
```

The following figure shows how the tasks behave in the lwsem example.