# Semaphores

## Exercise 1 – Resource guard

Write program that uses a **mutex** to guard write access to debug serial port. Only one task may write to the serial port at a time.

See lpc_board_nxp_lpcxpresso_1549/inc/board_api.h for debug serial port API details (Functions starting with Board_UART).

All three buttons are grounding buttons and they require you to use pullup-more in the input pins. Pullup gives the input pins a default value (high --> 1). When button is pressed the input pin goes low (= 0).

The program must have three tasks:

1. Task to monitor SW1 (PIO 0.17). When button is pressed. Task prints "SW1 pressed" to the serial port.
2. Task to monitor SW2 (PIO 1.11). When button is pressed. Task prints "SW2 pressed" to the serial port.
3. Task to monitor SW3 (PIO 1.9). When button is pressed. Task prints "SW3 pressed" to the serial port.

## Exercise 2 – Activity indicator

Write a program that creates two tasks: one for reading characters from the serial port and the other for indicating received characters on the serial port. Use a **binary semaphore** to notify serial port activity to the indicator task. Note that a single blink sequence (200 ms) takes much longer than transmission time of one character (0.1 ms) and only one blink after last character is allowed.

### Task 1

Task reads characters from debug serial port and echoes them back to the serial port. When a character is received the task sends an indication to blinker task.

See lpc_board_nxp_lpcxpresso_1549/inc/board_api.h for debug serial port API details.

### Task 2

This task blinks the led once (100 ms on, 100 ms off) when it receives activity indication.

## Exercise 3 – Oracle

Write a program that simulates an oracle that gives obscure answers to questions that are asked from it. The program must have at least two tasks.

Use **counting semaphore** to notify Oracle that an answer is required. Both reader task and the oracle use debug serial port so you must use another semaphore to guard access to debug serial port.

Write a program that creates two tasks:

## Task 1

Task reads characters from debug serial port until a line feed character (LF, '\n') or carriage return (CR, '\r') is received or 60 characters have been received. Then the characters are sent back to the serial port preceded by "[You] ". If the received characters contains a question mark '?' anywhere inside the line then the oracle task is notified.

See lpc_board_nxp_lpcxpresso_1549/inc/board_api.h for debug serial port API details.

## Task 2

Task wait for notification and when the notification is received it prints "[Oracle] Hmmm…" and then after 3 seconds a random answer (make up a bunch of crazy answer to choose from). The answer starts with "[Oracle]" which is followed by the answer and then pauses for two seconds before starting to wait for another notification.

Note that in this case the indications must queue up so that if you send three questions back to back the oracle will send 3 answers at 5 second intervals (three seconds "thinking" time plus two seconds "sleep" after the answer is written).

Note that task 1 must be able to use serial port while oracle is "thinking" or "sleeping".

Example:

[You] WTF?

[Oracle] Hmmm…

[Oracle] You will meet a tall dark stranger.

[You] Do penguins fly?

[Oracle] Hmmm…

[Oracle] Why would you go there?

Hint: Echo the received characters back at the time of writing except for LF/CR. Then send a CR followed by "[You]" to overwrite the current line. Send a CR+ LF at the end to begin a new line.