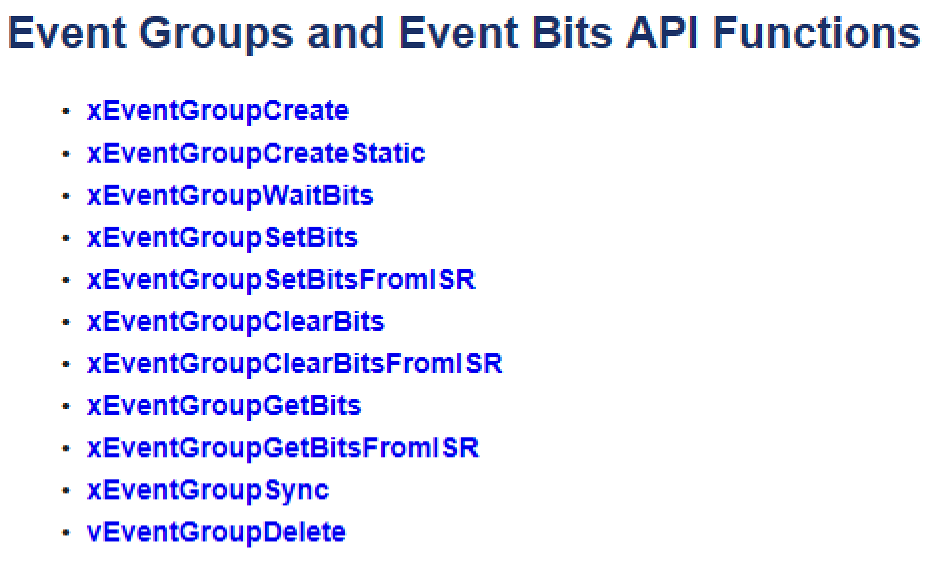
Lab Objective

To demonstrate FreeRTOS event groups using the following features: -

- Using an event group bit to synchronise 2 tasks
- Programming a task to wait for multiple event group bits



Event Groups and Event Bits API Functions

- xEventGroupCreate
- xEventGroupCreateStatic
- xEventGroupWaitBits
- xEventGroupSetBits
- xEventGroupSetBitsFromISR
- xEventGroupClearBits
- xEventGroupClearBitsFromISR
- xEventGroupGetBits
- xEventGroupGetBitsFromISR
- xEventGroupSync
- vEventGroupDelete

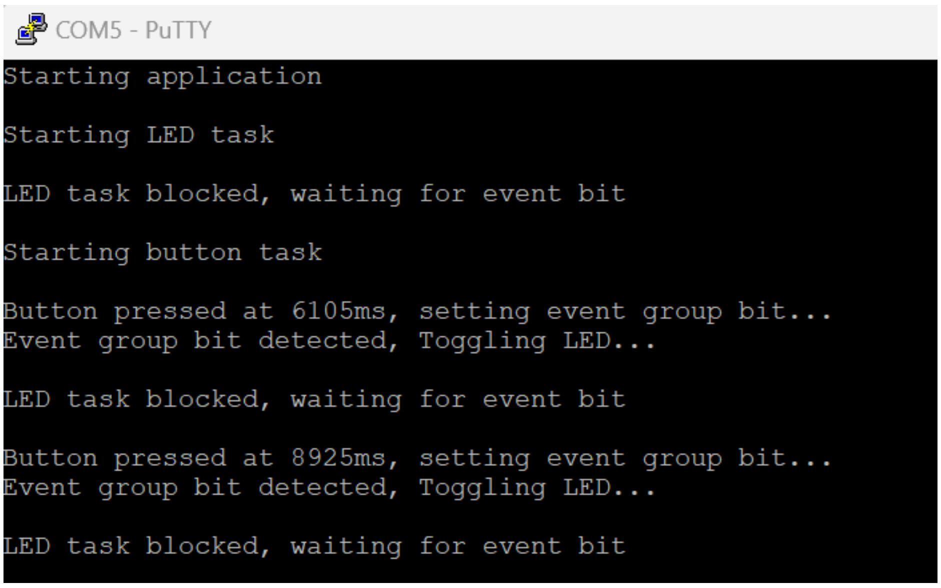
Reference: FreeRTOS Reference Manual Chapter 6

Part A: Use a single event group bit to synchronise 2 tasks

Create an event group with a single event bit. Create 2 tasks called SW2_Task and LED_Task in userApp().

The button task polls the active-low push-button switch every 100ms. The event bit is set when a switch press is detected. The time of the switch press should be printed using the xTaskGetTickCount() function.

The LED task blocks waiting for the event bit to be set. LED2 should toggle when the LED task unblocks.



```
Starting application

Starting LED task

LED task blocked, waiting for event bit

Starting button task

Button pressed at 6105ms, setting event group bit...
Event group bit detected, Toggling LED...

LED task blocked, waiting for event bit

Button pressed at 8925ms, setting event group bit...
Event group bit detected, Toggling LED...

LED task blocked, waiting for event bit
```

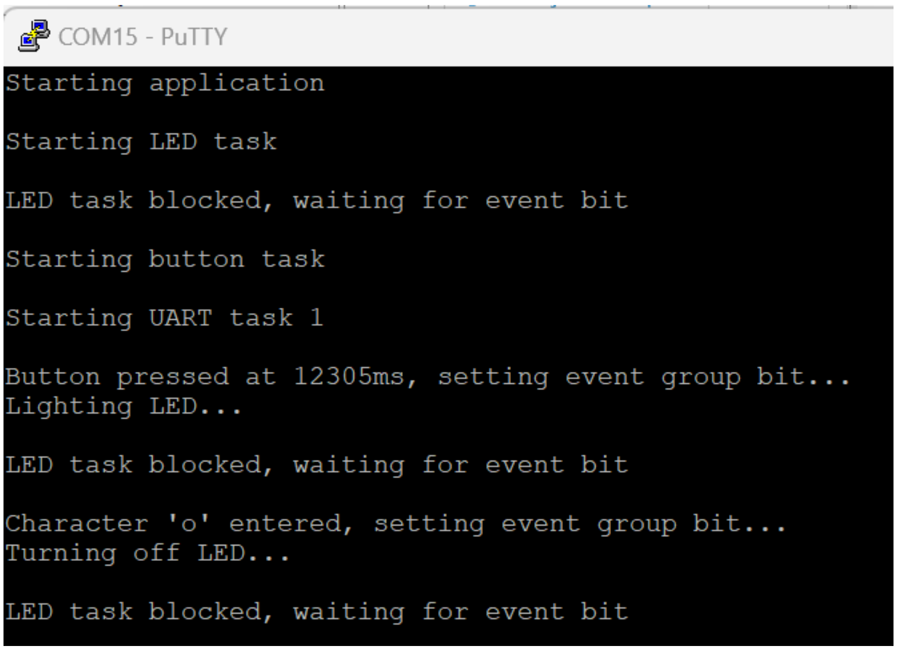
Part B: Single task waiting on event group bits from multiple tasks

Create an event group with 2 event group bits, buttonBit and offBit. Create 3 tasks called buttonTask, UARTTask and LEDTask in userApp().

The UART task polls the UART receiver for a received character. The offBit event group bit is set on receiving the character 'e'. An error message should be printed for any other received character.

The buttonTask polls the active-low push-button switch every 100ms. The buttonBit event group bit is set when a switch press is detected.

The LEDTask blocks waiting for **any** of the two event bits to be set. LED2 should light when the buttonBit is set by the buttonTask. LED2 should turn off when the offBit is set by the UARTTask.



```
Starting application

Starting LED task

LED task blocked, waiting for event bit

Starting button task

Starting UART task 1

Button pressed at 12305ms, setting event group bit...
Lighting LED...

LED task blocked, waiting for event bit

Character 'o' entered, setting event group bit...
Turning off LED...

LED task blocked, waiting for event bit
```

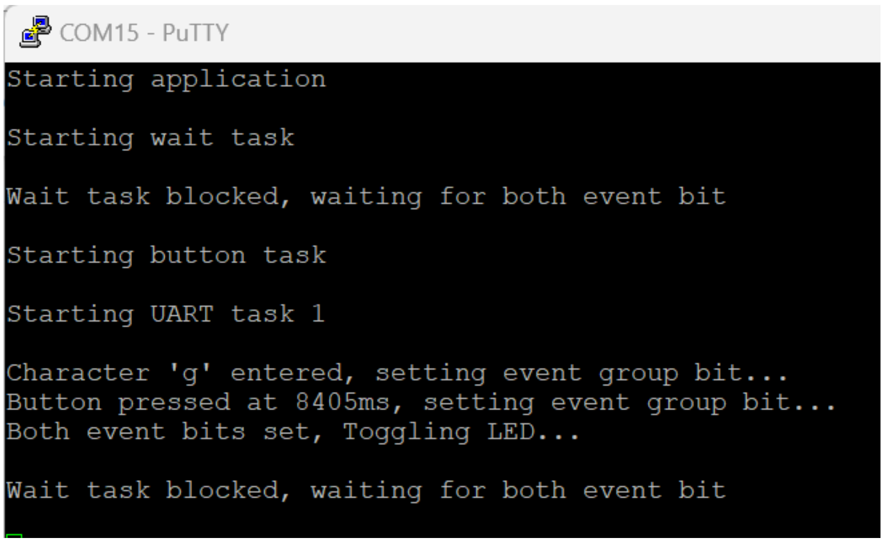
Part C: Using an Event Group to Synchronise to Multiple Events

Create an event group with 2 event group bits, buttonBit, and goBit. Create 3 tasks called buttonTask, UARTTask and waitTask in userApp().

The buttonTask polls the active-low push-button switch every 100ms. The buttonBit event group bit is set when a switch press is detected.

The UART task polls the UART receiver for a received character. The goBit event group bit is set on receiving the go character ‘g’. An error message should be printed for any other received character.

The waitTask blocks waiting for **both** event bits to be set. On unblocking, the waitTask should flash the green LED 3 times and clear the 2 event bits.



```
Starting application

Starting wait task

Wait task blocked, waiting for both event bit

Starting button task

Starting UART task 1

Character 'g' entered, setting event group bit...
Button pressed at 8405ms, setting event group bit...
Both event bits set, Toggling LED...

Wait task blocked, waiting for both event bit
```

Part D: Synchronising multiple tasks

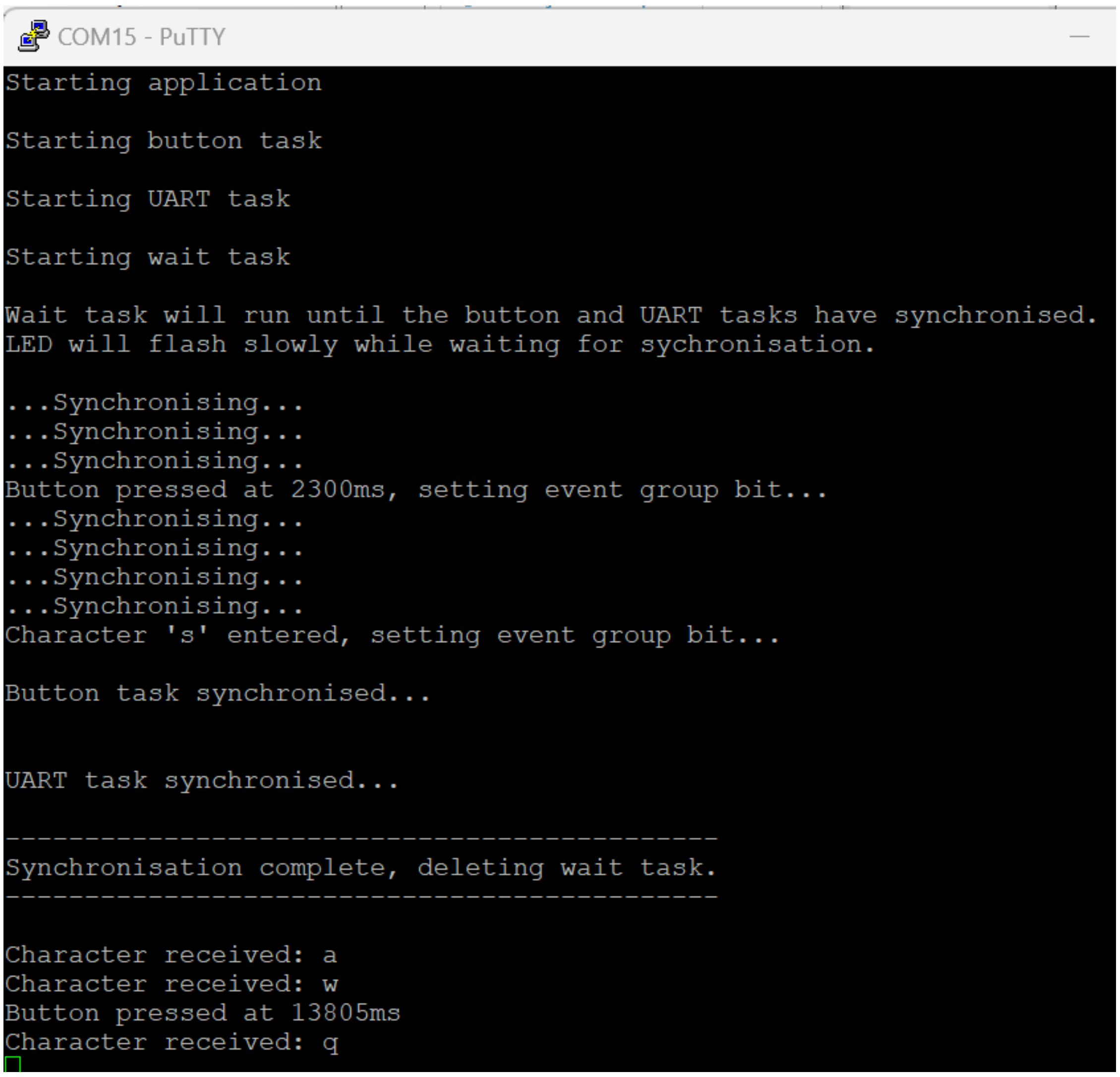
This functionality is typically used to synchronise multiple tasks (often called a task rendezvous), where each task must wait for the other tasks to reach a synchronisation point before proceeding.

Create an event group with 2 event bits, buttonBit and UARTBit. Create 3 tasks called buttonTask, UART_Task, and waitTask in userApp(). Your program should use the xEventGroupSync() function to synchronise the button and UART tasks. See the example code in the FreeRTOS API reference manual.

The UART task polls the UART receiver for a received character. The UART_Task starts the synchronisation process on receiving the character ‘s’ by setting its sync bit UARTBit and blocking while waiting for synchronisation. An error message should be printed for any other received character. After synchronisation, the task should just print any character received by the UART.

The buttonTask polls the active-low push-button switch every 100ms. On detection of a switch press, the task will set its sync bit (buttonBit) and block while waiting for synchronisation. After synchronisation, the buttonTask should print a message every time the switch is pressed.

The waitTask task should run while the other 2 tasks are waiting for synchronisation. The LED flashes slowly while the Waiting task runs, and a synchronising message should be printed. The wait task should be deleted once synchronisation has occurred.


COM15 - PuTTY

```
Starting application

Starting button task

Starting UART task

Starting wait task

Wait task will run until the button and UART tasks have synchronised.
LED will flash slowly while waiting for sychronisation.

...Synchronising...
...Synchronising...
...Synchronising...
Button pressed at 2300ms, setting event group bit...
...Synchronising...
...Synchronising...
...Synchronising...
...Synchronising...
Character 's' entered, setting event group bit...

Button task synchronised...


UART task synchronised...

---------------------------------------------
Synchronisation complete, deleting wait task.
---------------------------------------------

Character received: a
Character received: w
Button pressed at 13805ms
Character received: q
```