Lab Assignment: FreeRTOS Tasks

Objective

To give you experience in:

1. Loading firmware onto the SJOne board
2. Observing the RTOS round-robin scheduler in effect
3. Provide hands-on experience with the UART character output timing

Assignment

Part 0. Fill Out the Code

Use the following code template:

1. Reference the code comments for what you need to do.
2. Fill out the x**TaskCreate** method parameters.
   - See the FreeRTOS+Tasks document or checkout the FreeRTOS xTaskCreate API website.
3. Do not use **printf**, instead use **uart0_puts()** function from **#include "uart0_min.h"**
   - This is deliberate because uart0_puts() is not interrupt driven, and uses a polled version of the char output driver

> FreeRTOSConfig.h defines the total priority levels, but for the template project it is likely to be 4

```
#include "FreeRTOS.h"
#include "task.h"
#include "uart0_min.h"
void vTaskOneCode(void *p)
```

```
{
    while(1)
    {
    uart0_puts("aaaaaaaaaaaaaaaaaaaa");
    vTaskDelay(100); // This sleeps the task for 100ms (because 1 RTOS tick = 1 millisecond)
    }
}
// Create another task and run this code in a while(1) loop
void vTaskTwoCode(void *p)
{
    while(1)
    {
    uart0_puts("bbbbbbbbbbbbbbbbbbbb");
    vTaskDelay(100);
    }
}
// You can comment out the sample code of lpc1758_freertos project and run this code instead
int main(int argc, char const *argv[])
{
    /// This "stack" memory is enough for each task to run properly (512 * 32-bit) = 2Kbytes stack
    const uint32_t STACK_SIZE_WORDS = 512;

    /**
     * Observe and explain the following scenarios:
     *
     * 1) Same Priority: A = 1, B = 1
     * 2) Different Priority: A = 2, B = 1
     * 3) Different Priority: A = 1, B = 2
     *
     * Turn in screen shots of what you observed
     * as well as an explanation of what you observed
     */
    xTaskCreate(vTaskOneCode, /* Fill in the rest parameters for this task */ );
    xTaskCreate(vTaskTwoCode, /* Fill in the rest parameters for this task */ );
    /* Start Scheduler - This will not return, and your tasks will start to run their while(1) loop */
    vTaskStartScheduler();
```

```
    return 0;}
```

Code Block 1: Main.cpp

Part 1: Make observations of the output

- How come 4(or 3 sometimes) characters are printed from each task? Why not 2 or 5, or 6?
- Alter the priority of one of the tasks, and note down the observations. Note down WHAT you see and WHY.

Hints

- The printf data appears over 38400bps UART, and it takes 10-bits to output a single 8-bit char, and therefore you can send as many as 3840 chars per second.
- The SJ-One board RTOS runs at 1Khz meaning that 1 RTOS tick is 1 millisecond (for the round-robin scheduler)
- You have to relate the speed of the RTOS round-robin scheduler with the speed of the UART to answer the questions above

Part 2. Change the priority levels

Now that you have the code running with identical priority levels, try the following:

1. Change the priority of the two tasks.
    - A=1, B=1
    - A=2, B=1
    - A=1, B=2

2. Take a screenshot of what you see from the console.
3. Write an explanation of why you think the output came out the way it did using your knowledge about RTOS.

What to turn in (to Canvas as PDF):

1. Relevant code
2. Your observation and explanation
3. Snapshot of the output for all scenarios.

Revision #15
Created 1 year ago by Admin
Updated 6 days ago by sree harsha