main.c

```c

#include <stdint.h>
#include <stdbool.h>
#include "main.h"
#include "drivers/pinout.h"
#include "utils/uartstdio.h"


// TivaWare includes
#include "driverlib/sysctl.h"
#include "driverlib/debug.h"
#include "driverlib/rom_map.h"
#include "driverlib/rom.h"
#include "driverlib/timer.h"
#include "driverlib/inc/hw_memmap.h"
#include "driverlib/inc/hw_ints.h"

// FreeRTOS includes
#include "FreeRTOSConfig.h"
#include "FreeRTOS.h"
#include <timers.h>
#include <semphr.h>
#include "task.h"
#include "queue.h"


#define FIB_LIMIT_FOR_32_BIT 47
#define TIME_TO_RUN 200 //ms

unsigned long int ulPeriod;
unsigned int Hz = 1;   // frequency in Hz

SemaphoreHandle_t task1SyncSemaphore, task2SyncSemaphore;
TickType_t startTimeTick;


void fiboncacci(int ms){
    TickType_t xStartTick = xTaskGetTickCount();
    TickType_t xCurrentTick = xTaskGetTickCount();
    uint32_t fib = 1, fib_a = 1, fib_b = 1;
    uint32_t i;
    while((xCurrentTick - xStartTick) < pdMS_TO_TICKS(ms)){
        for (i = 0; i < FIB_LIMIT_FOR_32_BIT; i++){
            fib_a = fib_b;
            fib_b = fib;
            fib = fib_a + fib_b;
        }
        xCurrentTick = xTaskGetTickCount();
    }


}

```

```c

// Process 1
void xTask1(void * pvParameters)
{
    TickType_t xLastWakeTime;
    xLastWakeTime = xTaskGetTickCount();

    while((xLastWakeTime - startTimeTick) < TIME_TO_RUN){
        if (xSemaphoreTake(task1SyncSemaphore, portMAX_DELAY) == pdTRUE)
        {
            TickType_t xCurrentTick = xTaskGetTickCount();
            fiboncacci(10);
            TickType_t xFibTime = xTaskGetTickCount();
            UARTprintf("Task 1) Current time after execution: %d time to execute Fib: %d \n", xCurrentTick, (xFibTime - xCurrentTick));
            xLastWakeTime = xCurrentTick;
            xSemaphoreGive(task2SyncSemaphore);
        }
    }
}


// Process 2
void xTask2(void *pvParameters)
{
    TickType_t xLastWakeTime;
    xLastWakeTime = xTaskGetTickCount();

    while ((xLastWakeTime - startTimeTick) < TIME_TO_RUN)
    {


        if (xSemaphoreTake(task2SyncSemaphore, portMAX_DELAY) == pdTRUE)
        {
            TickType_t xCurrentTick = xTaskGetTickCount();
            fiboncacci(40);
            TickType_t xFibTime = xTaskGetTickCount();
            UARTprintf("Task 1) Current time after execution: %d time to execute Fib: %d \n", xCurrentTick, (xFibTime - xCurrentTick));
            xLastWakeTime = xCurrentTick;
            xSemaphoreGive(task1SyncSemaphore);
        }
    }
}


// Main function
int main(void)
{
    // Initialize system clock to 120 MHz
    uint32_t output_clock_rate_hz;
    output_clock_rate_hz = ROM_SysCtlClockFreqSet(
                               (SYSCTL_XTAL_25MHZ | SYSCTL_OSC_MAIN |
                                SYSCTL_USE_PLL | SYSCTL_CFG_VCO_480),
                               SYSTEM_CLOCK);
    ASSERT(output_clock_rate_hz == SYSTEM_CLOCK);


    // Initialize the GPIO pins for the Launchpad
    PinoutSet(false, false);
    UARTStdioConfig(0, 230400, SYSTEM_CLOCK);



    task1SyncSemaphore = xSemaphoreCreateBinary();
    task2SyncSemaphore = xSemaphoreCreateBinary();


    xTaskCreate(xTask1, "Task1", configMINIMAL_STACK_SIZE, NULL, 1, NULL);
    xTaskCreate(xTask2, "Task2", configMINIMAL_STACK_SIZE, NULL, 1, NULL);

    xSemaphoreGive(task1SyncSemaphore);
    startTimeTick = xTaskGetTickCount();

    vTaskStartScheduler();

    return (0);
}


/*  ASSERT() Error function
 *
 *  failed ASSERTS() from driverlib/debug.h are executed in this function
 */
void __error__(char *pcFilename, uint32_t ui32Line)
{
    // Place a breakpoint here to capture errors until logging routine is finished
    while (1)
    {
    }
}
```