

main.c

```
15
16 #include <stdint.h>
17 #include <stdbool.h>
18 #include "main.h"
19 #include "drivers/pinout.h"
20 #include "utils/uartstdio.h"
21
22
23 // TivaWare includes
24 #include "driverlib/sysctl.h"
25 #include "driverlib/debug.h"
26 #include "driverlib/rom_map.h"
27 #include "driverlib/rom.h"
28 #include "driverlib/timer.h"
29 #include "driverlib/inc/hw_memmap.h"
30 #include "driverlib/inc/hw_ints.h"
31
32 // FreeRTOS includes
33 #include "FreeRTOSConfig.h"
34 #include "FreeRTOS.h"
35 #include <timers.h>
36 #include <semphr.h>
37 #include "task.h"
38 #include "queue.h"
39
40
41 #define FIB_LIMIT_FOR_32_BIT 47
42 #define TIME_TO_RUN 200 //ms
43
44 unsigned long int ulPeriod;
45 unsigned int Hz = 1; // frequency in Hz
46
47 SemaphoreHandle_t task1SyncSemaphore, task2SyncSemaphore;
48 TickType_t startTickCount;
49
50
51 void fiboncacci(int ms){
52     TickType_t xStartTick = xTaskGetTickCount();
53     TickType_t xCurrentTick = xTaskGetTickCount();
54     uint32_t fib = 1, fib_a = 1, fib_b = 1;
55     uint32_t i;
56     while((xCurrentTick - xStartTick) < pdMS_TO_TICKS(ms)){
57         for (i = 0; i < FIB_LIMIT_FOR_32_BIT; i++){
58             fib_a = fib_b;
59             fib_b = fib;
60             fib = fib_a + fib_b;
61         }
62         xCurrentTick = xTaskGetTickCount();
63     }
64
65
66 }
67
```

```
68
69 // Process 1
70 void xTask1(void * pvParameters)
71 {
72     TickType_t xLastWakeTime;
73     xLastWakeTime = xTaskGetTickCount();
74
75     while((xLastWakeTime - startTimeTick) < TIME_TO_RUN){
76         if (xSemaphoreTake(task1SyncSemaphore, portMAX_DELAY) == pdTRUE)
77         {
78             TickType_t xCurrentTick = xTaskGetTickCount();
79             fiboncacci(10);
80             TickType_t xFibTime = xTaskGetTickCount();
81             UARTprintf("Task 1) Current time after execution: %d time to execute Fib:
%d \n", xCurrentTick, (xFibTime - xCurrentTick));
82             xLastWakeTime = xCurrentTick;
83             xSemaphoreGive(task2SyncSemaphore);
84         }
85     }
86 }
87
88
89 // Process 2
90 void xTask2(void *pvParameters)
91 {
92     TickType_t xLastWakeTime;
93     xLastWakeTime = xTaskGetTickCount();
94
95     while ((xLastWakeTime - startTimeTick) < TIME_TO_RUN)
96     {
97
98
99         if (xSemaphoreTake(task2SyncSemaphore, portMAX_DELAY) == pdTRUE)
100         {
101             TickType_t xCurrentTick = xTaskGetTickCount();
102             fiboncacci(40);
103             TickType_t xFibTime = xTaskGetTickCount();
104             UARTprintf("Task 1) Current time after execution: %d time to execute Fib:
%d \n", xCurrentTick, (xFibTime - xCurrentTick));
105             xLastWakeTime = xCurrentTick;
106             xSemaphoreGive(task1SyncSemaphore);
107         }
108     }
109 }
110
111
112 // Main function
113 int main(void)
114 {
115     // Initialize system clock to 120 MHz
116     uint32_t output_clock_rate_hz;
117     output_clock_rate_hz = ROM_SysCtlClockFreqSet(
118         (SYSCTL_XTAL_25MHZ | SYSCTL_OSC_MAIN |
119          SYSCTL_USE_PLL | SYSCTL_CFG_VCO_480),
120         SYSTEM_CLOCK);
121     ASSERT(output_clock_rate_hz == SYSTEM_CLOCK);
122 }
```

```
123
124 // Initialize the GPIO pins for the Launchpad
125 PinoutSet(false, false);
126 UARTStdioConfig(0, 230400, SYSTEM_CLOCK);
127
128
129
130 task1SyncSemaphore = xSemaphoreCreateBinary();
131 task2SyncSemaphore = xSemaphoreCreateBinary();
132
133
134 xTaskCreate(xTask1, "Task1", configMINIMAL_STACK_SIZE, NULL, 1, NULL);
135 xTaskCreate(xTask2, "Task2", configMINIMAL_STACK_SIZE, NULL, 1, NULL);
136
137 xSemaphoreGive(task1SyncSemaphore);
138 startTimeTick = xTaskGetTickCount();
139
140 vTaskStartScheduler();
141
142 return (0);
143 }
144
145
146 /* ASSERT() Error function
147 *
148 * failed ASSERTS() from driverlib/debug.h are executed in this function
149 */
150 void __error__(char *pcFilename, uint32_t ui32Line)
151 {
152     // Place a breakpoint here to capture errors until logging routine is finished
153     while (1)
154     {
155     }
156 }
```