```
1#include "sl_component_catalog.h"
 2#include "sl system init.h"
 3#include "app.h"
4#if defined(SL_CATALOG_POWER_MANAGER_PRESENT)
5#include "sl_power_manager.h"
6#endif
7#if defined(SL CATALOG KERNEL PRESENT)
8#include "sl_system_kernel.h"
9#else // SL CATALOG KERNEL PRESENT
10#include "sl_system_process_action.h"
11#endif // SL_CATALOG_KERNEL_PRESENT
13 #include "FreeRTOS.h"
14#include "task.h"
16 #include "em_device.h"
17 #include "em chip.h"
20 * Extern Includes for Lab04
                          22 extern void task_A(), task_B(), task_C(), task_D();
24TickType_t startTime = pdMS_TO_TICKS(10); //handles the hazard in RTOS
25
26 void TaskA(void *params)
27 {
28 const int period = pdMS_TO_TICKS(3); // period = 3ms
   TickType_t prevWakeTime = 0;
30
   (void) params; // suppress warning
31
32 // wait for first release
33 vTaskDelayUntil(&prevWakeTime, startTime);
34
   for (;;)
35
36
     task A(); // perform actual task
37
     vTaskDelayUntil(&prevWakeTime, period);
38
39 }
40
41 void TaskB(void *params)
   const int period = pdMS_TO_TICKS(5); // period = 5ms
43
44
   TickType t prevWakeTime = 0;
45
   (void) params; // suppress warning
46
47
   // wait for first release
48
   vTaskDelayUntil(&prevWakeTime, startTime);
49
   for (;;)
50
   {
51
     task B(); // perform actual task
52
     vTaskDelayUntil(&prevWakeTime, period);
53
   }
54 }
55
56 void TaskC(void *params)
57 {
```

```
58
     const int period = pdMS_TO_TICKS(6); // period = 6ms
     TickType t prevWakeTime = 0;
     (void) params; // suppress warning
 61
    // wait for first release
    vTaskDelayUntil(&prevWakeTime, startTime);
 64
    for (;;)
 65
 66
       task C(); // perform actual task
 67
       vTaskDelayUntil(&prevWakeTime, period);
 68
 69 }
 70
71 void TaskD(void *params)
 72 {
    const int period = pdMS_TO_TICKS(11); // period = 11ms
 73
     TickType t prevWakeTime = 0;
 75
     (void) params; // suppress warning
 76
 77
     // wait for first release
 78
    vTaskDelayUntil(&prevWakeTime, startTime);
 79
     for (;;)
 80
       task_D(); // perform actual task
 81
 82
       vTaskDelayUntil(&prevWakeTime, period);
 83
     }
 84 }
 85
 86 int main(void)
 87 {
    // Vendor function to work around bugs in some versions of the hardware
 89
       CHIP_Init();
 90
       //use xTaskCreate(function name, "string name", configMINIMAL_STACK_SIZE, NULL, PRIORITY,
  NULL);
 92
 93
       xTaskCreate(TaskA,
 94
                    "TaskA",
 95
                    configMINIMAL_STACK_SIZE,
 96
                   NULL,
 97
 98
                   NULL);
99
100
       xTaskCreate(TaskB,
101
                    "TaskB",
102
                    configMINIMAL STACK SIZE,
103
                   NULL,
104
                    2,
105
                   NULL);
106
       xTaskCreate(TaskC,
107
108
                    "TaskC"
                    configMINIMAL_STACK_SIZE,
109
110
                   NULL,
111
112
                   NULL);
113
```

```
main.c
      xTaskCreate(TaskD,
114
                   "TaskD",
115
                  configMINIMAL_STACK_SIZE,
116
                   NULL,
117
118
                   4,
119
                   NULL);
120
     vTaskStartScheduler();
121
122
123
124 while (1) {
125
126 }
127 }
128
```

Tuesday, October 17, 2023, 10:49 AM