```c
#include "sl_component_catalog.h"
#include "sl_system_init.h"
#include "app.h"
#if defined(SL_CATALOG_POWER_MANAGER_PRESENT)
#include "sl_power_manager.h"
#endif
#if defined(SL_CATALOG_KERNEL_PRESENT)
#include "sl_system_kernel.h"
#else // SL_CATALOG_KERNEL_PRESENT
#include "sl_system_process_action.h"
#endif // SL_CATALOG_KERNEL_PRESENT

#include "FreeRTOS.h"
#include "task.h"

#include "em_device.h"
#include "em_chip.h"

/*******************************************************************************//**
 * Extern Includes for Lab04
 ******************************************************************************/
extern void task_A(), task_B(), task_C(), task_D();

TickType_t startTime = pdMS_TO_TICKS(10); //handles the hazard in RTOS

void TaskA(void *params)
{
  const int period = pdMS_TO_TICKS(3); // period = 3ms
  TickType_t prevWakeTime = 0;
  (void) params; // suppress warning

  // wait for first release
  vTaskDelayUntil(&prevWakeTime, startTime);
  for (;;)
  {
    task_A(); // perform actual task
    vTaskDelayUntil(&prevWakeTime, period);
  }
}

void TaskB(void *params)
{
  const int period = pdMS_TO_TICKS(5); // period = 5ms
  TickType_t prevWakeTime = 0;
  (void) params; // suppress warning

  // wait for first release
  vTaskDelayUntil(&prevWakeTime, startTime);
  for (;;)
  {
    task_B(); // perform actual task
    vTaskDelayUntil(&prevWakeTime, period);
  }
}

void TaskC(void *params)
{
```

```
 58   const int period = pdMS_TO_TICKS(6); // period = 6ms
 59   TickType_t prevWakeTime = 0;
 60   (void) params; // suppress warning
 61
 62   // wait for first release
 63   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 {
 73   const int period = pdMS_TO_TICKS(11); // period = 11ms
 74   TickType_t prevWakeTime = 0;
 75   (void) params; // suppress warning
 76
 77   // wait for first release
 78   vTaskDelayUntil(&prevWakeTime, startTime);
 79   for (;;)
 80   {
 81     task_D(); // perform actual task
 82     vTaskDelayUntil(&prevWakeTime, period);
 83   }
 84 }
 85
 86 int main(void)
 87 {
 88   // Vendor function to work around bugs in some versions of the hardware
 89     CHIP_Init();
 90
 91     //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                 1,
 98                 NULL);
 99
100     xTaskCreate(TaskB,
101                 "TaskB",
102                 configMINIMAL_STACK_SIZE,
103                 NULL,
104                 2,
105                 NULL);
106
107     xTaskCreate(TaskC,
108                 "TaskC",
109                 configMINIMAL_STACK_SIZE,
110                 NULL,
111                 3,
112                 NULL);
113
```

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