```c
/***************************************************************************//**
 * @file main.c
 * @brief main() function.
 ******************************************************************************
 * # License
 * <b>Copyright 2020 Silicon Laboratories Inc. www.silabs.com</b>
 ******************************************************************************
 *
 * The licensor of this software is Silicon Laboratories Inc. Your use of this
 * software is governed by the terms of Silicon Labs Master Software License
 * Agreement (MSLA) available at
 * www.silabs.com/about-us/legal/master-software-license-agreement. This
 * software is distributed to you in Source Code format and is governed by the
 * sections of the MSLA applicable to Source Code.
 *
 ******************************************************************************/
#include "sl_component_catalog.h"
#include "sl_system_init.h"
#include "app.h"
#include "EM_device.h"
#include "EM_chip.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

/***************************************************************************//**
 * Extern Includes for Lab03
 ******************************************************************************/
extern void task_A(), task_B(), task_C(), task_D(), task_E(), task_F();

int n = 0;


void SysTick_Handler(void)
{
  switch(n)
  {
    case 0:
      task_A();
      task_B();
      task_D();
      break;
    case 1:
      task_F();
      task_A();
      task_C();
      break;
    case 2:
      task_A();
      task_B();
      task_C();
```

```
 58      break;
 59    case 3:
 60      task_A();
 61      task_E();
 62      break;
 63    case 4:
 64      task_B();
 65      task_A();
 66      task_D();
 67      break;
 68    case 5:
 69      task_A();
 70      task_B();
 71      task_C();
 72      break;
 73    case 6:
 74      task_F();
 75      task_A();
 76      task_C();
 77      break;
 78    case 7:
 79      task_A();
 80      task_B();
 81      task_D();
 82      break;
 83    case 8:
 84      task_A();
 85      task_E();
 86      break;
 87    case 9:
 88      task_A();
 89      task_B();
 90      task_C();
 91      //n = 0;
 92      break;
 93  }
 94  n = (n+1)%10;
 95}
 96
 97
 98int main(void)
 99{
100  // Initialize Silicon Labs device, system, service(s) and protocol stack(s).
101  // Note that if the kernel is present, processing task(s) will be created by
102  // this call.
103  sl_system_init();
104  SystemCoreClock = 14000000; // 14 MHz for this device
105  SysTick_Config(3 * SystemCoreClock / 1000); //3ms slice
106
107
108  // Initialize the application. For example, create periodic timer(s) or
109  // task(s) if the kernel is present.
110  app_init();
111
112#if defined(SL_CATALOG_KERNEL_PRESENT)
113  // Start the kernel. Task(s) created in app_init() will start running.
114  sl_system_kernel_start();
```

```
115 #else // SL_CATALOG_KERNEL_PRESENT
116
117
118   while (1) {
119     // Do not remove this call: Silicon Labs components process action routine
120     // must be called from the super loop.
121     sl_system_process_action();
122
123     // Application process.
124     app_process_action();
125
126 #if defined(SL_CATALOG_POWER_MANAGER_PRESENT)
127     // Let the CPU go to sleep if the system allows it.
128     sl_power_manager_sleep();
129 #endif
130   }
131 #endif // SL_CATALOG_KERNEL_PRESENT
132 }
133
```