```
/***************************************************************************//**
 * @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();
      break;
    case 3:
```

```c
      task_A();
      task_E();
      break;
    case 4:
      task_B();
      task_A();
      task_D();
      break;
    case 5:
      task_A();
      task_B();
      task_C();
      break;
    case 6:
      task_F();
      task_A();
      task_C();
      break;
    case 7:
      task_A();
      task_B();
      task_D();
      break;
    case 8:
      task_A();
      task_E();
      break;
    case 9:
      task_A();
      task_B();
      task_C();
      //n = 0;
      break;
  }
  n = (n+1)%10;
}


int main(void)
{
  // Initialize Silicon Labs device, system, service(s) and protocol stack(s).
  // Note that if the kernel is present, processing task(s) will be created by
  // this call.
  sl_system_init();
  SystemCoreClock = 14000000; // 14 MHz for this device
  SysTick_Config(3 * SystemCoreClock / 1000); //3ms slice


  // Initialize the application. For example, create periodic timer(s) or
  // task(s) if the kernel is present.
  app_init();

#if defined(SL_CATALOG_KERNEL_PRESENT)
  // Start the kernel. Task(s) created in app_init() will start running.
  sl_system_kernel_start();
#else // SL_CATALOG_KERNEL_PRESENT

```

```c

  while (1) {
    // Do not remove this call: Silicon Labs components process action routine
    // must be called from the super loop.
    sl_system_process_action();

    // Application process.
    app_process_action();

#if defined(SL_CATALOG_POWER_MANAGER_PRESENT)
    // Let the CPU go to sleep if the system allows it.
    sl_power_manager_sleep();
#endif
  }
#endif // SL_CATALOG_KERNEL_PRESENT
}

```