```
/**-----------------------------------------------------------------------------

   \file main.cpp

--                                                                            --
--              ECEN 5803 Mastering Embedded System Architecture              --
--                  Project 1 Module 3                                        --
--                Microcontroller Firmware                                    --
--                      main.cpp                                             --
--                                                                            --
-------------------------------------------------------------------------------
--
--  Designed for:  University of Colorado at Boulder
--
--
--  Designed by:  Tim Scherr
--  Revised by:  David Pasley & Ismail Yesildirek
--
-- Version: 2.0.1
-- Date of current revision:  2018-10-03
-- Target Microcontroller: Freescale MKL25ZVMT4
-- Tools used:  ARM mbed compiler
--              ARM mbed SDK
--              Keil uVision MDK v.5
--              Freescale FRDM-KL25Z Freedom Board
--
--
-- Functional Description:  Main code file generated by mbed, and then
--                           modified to implement a super loop bare metal OS.
--
--      Copyright (c) 2015, 2016 Tim Scherr  All rights reserved.
--
*/

#define MAIN
#include "shared.h"
#include "MKL25Z4.H"
#undef MAIN

extern volatile uint16_t SwTimerIsrCounter; //!< ISR counter

Ticker tick;  //!<  Creates a timer interrupt using mbed methods


/*****************************************************************************/
// MAIN function
/*****************************************************************************/
/// @brief Main function
/// The main function contains the setup and the main loop.
/*****************************************************************************/
int main()
{
/****************      ECEN 5803 add code as indicated   ***************/
    tick.attach(&timer0, 0.0001); //!< setup ticker to call flip every 100 microseconds
    uint32_t  count = 0;   //<! loop counter

    // initialize serial buffer pointers
   rx_in_ptr =  rx_buf; //!< pointer to the receive in data
   rx_out_ptr = rx_buf; //!< pointer to the receive out data
   tx_in_ptr =  tx_buf; //!< pointer to the transmit in data
   tx_out_ptr = tx_buf; //!< pointer to the transmit out


    /****************      ECEN 5803 add code as indicated   ***************/
   /* send a message to the terminal  */
   UART_direct_msg_put("\r\nSystem Reset\r\nCode ver. ");
   UART_direct_msg_put( CODE_VERSION );
   UART_direct_msg_put("\r\n");
   UART_direct_msg_put( COPYRIGHT );
   UART_direct_msg_put("\r\n");

```

```
72      set_display_mode();
73
74       while(1)       /// Cyclical Executive Loop
75       {
76           count++;               // counts the number of times through the loop
77
78      /****************      ECEN 5803 add code as indicated   ***************/
79           serial();            // Polls the serial port
80           chk_UART_msg();      // checks for a serial port message received
81           monitor();           // Sends serial port output messages depending
82                                //     on commands received and display mode
83       }
84   }
85
```