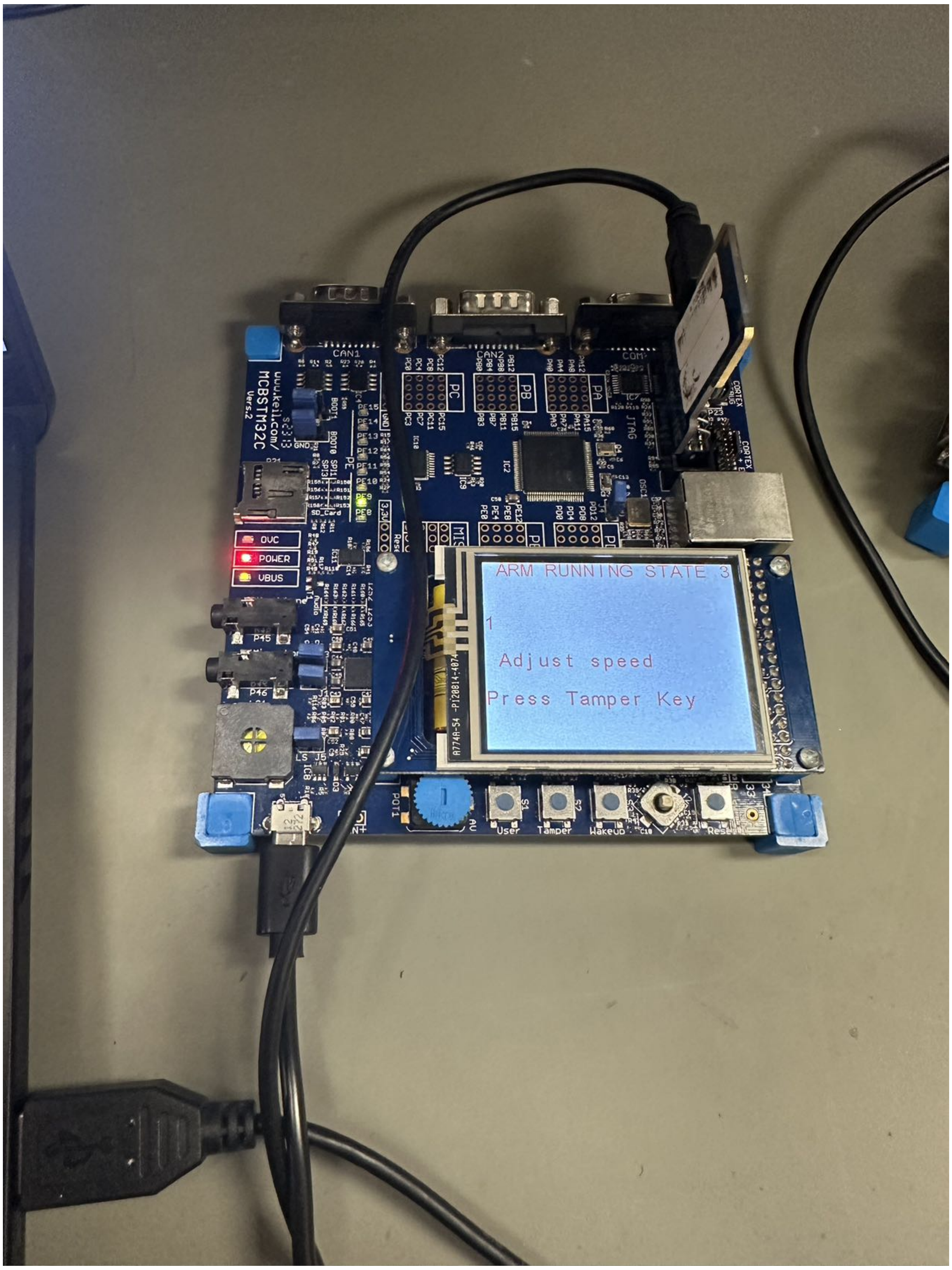
Lab 9 Yayu Mo

- Code

    -
      ```
      /******************************************************************************
      multitasking.c
      CSE/EE 5385/7385 Microprocessor Architecture and Interfacing
      ARM MCBSTM32C Finite state machine: polling & delay loops
      ******************************************************************************/

      #include "stdio.h"
      #include <stm32f10x_cl.h>
      #include "GLCD.h"
      #include <ctype.h>                    /* character functions                 */


      #define LED_NUM     8                   /* Number of user LEDs                 */
      const long led_mask[] = {1 << 8, 1 << 9, 1 << 10, 1 << 11, 1 << 12, 1 << 13, 1 << 14, 1 << 15};
      int i = 0;

      /*Converts an integer to a char*/
      char numToChar(int n){
          char value;
          switch (n){
              case 0: value = '0'; break;
              case 1: value = '1'; break;
              case 2: value = '2'; break;
              case 3: value = '3'; break;
              case 4: value = '4'; break;
              case 5: value = '5'; break;
              case 6: value = '6'; break;
              case 7: value = '7'; break;
              case 8: value = '8'; break;
              default: value = '0'; break;
          }
          return value;
      }

      void delay(int n) {
          for (i = 0; i < (n * 1000000); i++);/* Delay */
      }

      int main(void) {
          int num = -1;
          int dir = 1;
          int AD_val;
          int but = 0, but_ = -1;
          int state = 0;
          int j;
          SystemInit();
          //SysTick_Config(SystemFrequency/100-1);/* Generate interrupt every 10 ms     */

          /* Configure the GPIO for Push Buttons                                    */
          RCC->APB2ENR |= 1 << 2;             /* Enable GPIOA clock                 */
          RCC->APB2ENR |= 1 << 3;             /* Enable GPIOB clock                 */
          RCC->APB2ENR |= 1 << 4;             /* Enable GPIOC clock                 */
          GPIOA->CRL &= 0xFFFFFFF0;
          GPIOA->CRL |= 0x00000004;
          GPIOB->CRL &= 0x0FFFFFFF;
          GPIOB->CRL |= 0x40000000;
          GPIOC->CRH &= 0xFF0FFFFF;
          GPIOC->CRH |= 0x00400000;

          /* Setup GPIO for LEDs                                                    */
          RCC->APB2ENR |= 1 << 6;             /* Enable GPIOE clock                 */
          GPIOE->CRH = 0x33333333;           /* Configure the GPIO for LEDs        */

          GLCD_Init();                          /* Initialize graphical LCD display   */
          GLCD_Clear(White);                    /* Clear graphical LCD display        */

          for (;;) {                            /* Loop forever                       */
              /* Button inputs                                                      */
              but = 0;
              if (GPIOB->IDR & (1 << 7)) but |= (1 << 0);  /* Button User (S1)        */
              //if ((GPIOB->IDR & (1 <<  7)) && state == 3) cycle = 1;  /* Button User (S1)             */
              if (GPIOC->IDR & (1 << 13)) but |= (1 << 1);  /* Button Tamper (S2)       */
              if (GPIOA->IDR & (1 << 0)) but |= (1 << 2);  /* Button Wakeup (S3)       */
              but ^= 0x03;
      ```

```
        switch (state) {
            case 0:
                /********* YOUR CODE GOES HERE **********/
                /*Display state, check for push button, move to state if User pressed*/
//                GLCD_Clear(White);
                GLCD_SetBackColor(Red);
                GLCD_SetTextColor(White);
                GLCD_DisplayString(0, 0, 1, " ARM RESET STATE 0 ");
                GLCD_SetBackColor(White);
                GLCD_SetTextColor(Blue);
                GLCD_DisplayString(3, 0, 1, " Engine Idle ... ");
                GLCD_DisplayString(5, 0, 1, " Waiting to start ");
                GLCD_DisplayString(7, 0, 1, " Press user key ");

                if (!(GPIOB->IDR & (1 << 7))) {
                    state = 1;
                }
                /********** END **********/
                break;

            case 1:
                /********* YOUR CODE GOES HERE **********/
                /*Display state, add delay, move to next state*/
                GLCD_Clear(White);
                GLCD_SetBackColor(White);
                GLCD_SetTextColor(Black);
                GLCD_DisplayString(0, 0, 1, " ARM RUNNING STATE 1 ");
                GLCD_DisplayString(5, 0, 1, " Gear 1 ");
                delay(3);
                state = 2;
                /********** END **********/
                break;

            case 2:
                /********* YOUR CODE GOES HERE **********/
                GLCD_Clear(White);
                GLCD_SetTextColor(Black);
                GLCD_DisplayString(0, 0, 1, " ARM RUNNING STATE 2 ");
                GLCD_DisplayString(5, 0, 1, " Gear 2 ");
                delay(5);
                state = 3;
                /********** END **********/
                break;

            case 3:
                /********* YOUR CODE GOES HERE **********/
                /*Display state, check for push button, and move to corresponding state*/
                GLCD_SetTextColor(Red);
                GLCD_DisplayString(0, 0, 1, " ARM RUNNING STATE 3 ");
                GLCD_DisplayString(5, 0, 1, " Adjust speed ");
                GLCD_DisplayString(7,0,1,"Press Tamper Key");


                /*display User, Tamper, WakeUp, buttons state*/
                if(!(GPIOB->IDR & (1<<7))){ // User
                    /*Flash all 8 LEDs sequentially*/
                    for(;;){ // Start an infinite loop
                        if(ADC1->SR & (1<<1)) { // Check if the ADC conversion is complete
                            AD_val = ADC1->DR & 0x0FFFFFFF; // Read the ADC value
                            ADC1->CR2 |= 1 << 22; // Start a new ADC conversion
                        }

                        /* claculate the number */
                        num += dir;
                        if(num >= LED_NUM){
                            dir = -1;
                            num = LED_NUM - 1;
                        } else if(num < 0) {
                            dir = 1;
                            num = 0;
                        }

                        GPIOE->BSRR = led_mask[num]; // Turn on the LED corresponding to the current counter value 'num'
                        GLCD_DisplayChar(3,0,1, numToChar(num)); // Display the counter value on the LCD
                        for(i=0; i<((AD_val << 7) + 100000); i++); // Wait for a certain amount of time, determined by the ADC value
                        GPIOE->BSRR = led_mask[num] << 16; // Turn off the LED corresponding to the current counter value 'num'

                        if(!(GPIOC->IDR & (1<<13))){ // Tamper
                            break;
                            state = 2;
                        }else if((GPIOA->IDR & (1<<0))){ // Wakeup
                            break;
                            state = 4;
                        }
                    } // Repeat the loop



                }else if(!(GPIOC->IDR & (1<<13))){ // Tamper
                    state = 2;
                }else if((GPIOA->IDR & (1<<0))){ // Wakeup
                    state = 4;
                }
                /*********** END ***********/
                break;

            case 4:
                /********* YOUR CODE GOES HERE ***********/
                /*Display state, add delay, move back to state zero*/
                GLCD_Clear(White);
                GLCD_SetTextColor(Black);
                GLCD_DisplayString(0, 0, 1, " ARM RUNNING STATE 4 ");
                GLCD_DisplayString(5, 0, 1, " Gear 4 ");
                delay(6);
                state = 0;
                /*********** END ***********/
                break;
        }
    }
}
```

- Screenshot

- 

```
ARM RUNNING STATE 3
1
Adjust speed
Press Tamper Key
```