Lab 8

- Code
  - ```
    /*****************************************************************************
    multitasking.c
    CSE/EE 5385/7385 Microprocessor Architecture and Interfacing
    Multitasking using Keil RTX Real-Time Operating System (RTX-RTOS)
    *****************************************************************************/
    #include <RTL.h>
    #include <stm32f10x_cl.h>
    #include "GLCD.h"

    OS_TID t_taskA;                         /* assigned task id of task: task_a */
    OS_TID t_taskB;                         /* assigned task id of task: task_b */
    OS_TID t_taskC;                         /* assigned task id of task: task_c */
    OS_MUT mut_GLCD;                        /* Mutex to control GLCD access     */

    __task void taskA(void);                                /*Task A*/
    __task void taskB(void);                                /*Task B*/
    __task void taskC(void);                                /*Task C*/
    __task void init(void);

    int joystick = 100;                                     /*Joystick offset*/
    int stick;
    int i;

    /*----------------------------------------------------------------------------
     *        Main: Initialize and start RTX Kernel
     *----------------------------------------------------------------------------*/
    int main(void) {
        SystemInit();                               /* initialize clocks            */

        /* 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;
        /* End Configure the GPIO for Push Buttons                                  */

        /* Configure the GPIO for Joystick  */
        RCC->APB2ENR |= 1 << 5;                     /* Enable GPIOD clock           */
        GPIOD->CRH   &= 0x00000FFF;
        GPIOD->CRH   |= 0x44444000;

        /*End Configure the GPIO for Joystick  */


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

        GLCD_Init();                                /* Initialize the GLCD          */
        GLCD_Clear(White);                          /* Clear the GLCD               */

        os_sys_init(init);                          /* Initialize RTX and start init */
    }

    /*----------------------------------------------------------------------------
     *        Task 'init': Initialize
     *----------------------------------------------------------------------------*/
    __task void init(void) {
        os_mut_init(mut_GLCD);

        /********* YOUR CODE GOES HERE *********                */
        /* Create task A,  Create task B, Create task C*/
        t_taskA = os_tsk_create(taskA, 1);
        t_taskB = os_tsk_create(taskB, 1);
        t_taskC = os_tsk_create(taskC, 1);
        /* Add Delay                                                                */
        os_dly_wait(50);
        /* send signal event flag 0x0001 to taskA           */
        os_evt_set(0x0001, t_taskA);
    ```
    ```

```
    /* Add Delay                                                          */
    os_dly_wait(50);
    /* send signal event flag 0x0001 to taskB          */
    os_evt_set(0x0001, t_taskB);
    /* Add Delay                                                          */
    os_dly_wait(50);
    /* send signal event flag 0x0001 to taskC          */
    os_evt_set(0x0001, t_taskC);
/************** END ********************              */

    os_tsk_delete_self();
}

/*--------------------------------------------------------------------------
 *        Task A: Display sequence of strings
 *-------------------------------------------------------------------------*/
__task void taskA(void) {
    /********* YOUR CODE GOES HERE **********/
    /* wait for an event flag 0x0001       */
    os_evt_wait_and(0x0001,0xffff);
    /*************** END *****************/
    for (;;) {                    /* endless loop */
        /********* YOUR CODE GOES HERE **********/
        /*Acquire and lock the LCD Mutex            */
//        os_mut_init(mut_GLCD);
        os_mut_wait(mut_GLCD, 0xffff);
        /*Set background color, set text color*/
        GLCD_SetBackColor(Blue);
        GLCD_SetTextColor(White);
        /*Display "Task A started"                  */
        GLCD_DisplayString(0,0,1, "                    ");
        GLCD_SetBackColor(Blue);
        GLCD_SetTextColor(White);
        GLCD_DisplayString(1,0,1, " Task A started     ");
        os_mut_release(mut_GLCD);
        os_dly_wait(50);

        os_mut_wait(mut_GLCD, 0xffff);
        GLCD_SetBackColor(Blue);
        GLCD_SetTextColor(White);
        /*display first name, display last name, display student ID on LCD sequentially*/
        GLCD_DisplayString(2,0,1, " Yayu               ");
        os_mut_release(mut_GLCD);
        os_dly_wait(50);

        os_mut_wait(mut_GLCD, 0xffff);
        GLCD_SetBackColor(Blue);
        GLCD_SetTextColor(White);
        GLCD_DisplayString(2,0,1, " Mo                 ");
        os_mut_release(mut_GLCD);
        os_dly_wait(50);

        os_mut_wait(mut_GLCD, 0xffff);
        GLCD_SetBackColor(Blue);
        GLCD_SetTextColor(White);
        GLCD_DisplayString(2,0,1, " 49336397           ");
        os_mut_release(mut_GLCD);
        os_dly_wait(50);


        /*Release the mutex*/
        os_mut_release(mut_GLCD);
        /* Add Delay          */
        os_dly_wait(50);
        /********** END **********/
    }
}

/*--------------------------------------------------------------------------
 *        Task B: Bargraph control
 *-------------------------------------------------------------------------*/
__task void taskB(void) {
    /********* YOUR CODE GOES HERE **********/
    /* wait for an event flag 0x0001       */
    os_evt_wait_and(0x0001,0xffff);

    /*************** END *****************/
    for (;;) {           /* endless loop */
        /********* YOUR CODE GOES HERE **********/
        /*Acquire and lock the LCD Mutex               */
```

```
//        os_mut_init(mut_GLCD);
        os_mut_wait(mut_GLCD, 0xffff);
        /*Set background color, set text color*/

        GLCD_SetBackColor(Black);
        GLCD_SetTextColor(Red);
        /*Display "Task B started"                    */
        GLCD_DisplayString(3,0,1, "                    ");
        GLCD_SetBackColor(Black);
        GLCD_SetTextColor(Red);
        GLCD_DisplayString(4,0,1, " Task B started     ");
        GLCD_SetBackColor(Black);
        GLCD_SetTextColor(Red);
        GLCD_DisplayString(5,0,1, "                    ");
        GLCD_SetBackColor(Black);
        GLCD_SetTextColor(Red);
        GLCD_DisplayString(6,0,1, "                    ");
        os_mut_release(mut_GLCD);

        /*Adjust offset according to joystick position  */
        stick = ~ GPIOD -> IDR;
        stick &= 0xf000;

        os_mut_wait(mut_GLCD, 0xffff);
        GLCD_SetBackColor(Black);
        GLCD_SetTextColor(Red);
        GLCD_Bargraph(0,120,joystick,40,1024);/*Display default bar graph*/
        os_mut_release(mut_GLCD);
        if(stick & (1 << 15)){
            joystick -= 40;
            os_mut_wait(mut_GLCD, 0xffff);
            GLCD_SetBackColor(Black);
            GLCD_SetTextColor(Red);
            GLCD_Bargraph(0,122,joystick,40,1024);/*Bar move to right when we move joystick to
right*/
            os_mut_release(mut_GLCD);
            os_dly_wait(50);
        }
        else if(stick & (1 << 13)){
            joystick += 40;
            os_mut_wait(mut_GLCD, 0xffff);
            GLCD_SetBackColor(Black);
            GLCD_SetTextColor(Red);
            GLCD_Bargraph(0,122,joystick,40,1024);/*Bar move to left when we move joystick to left*/
            os_mut_release(mut_GLCD);
            os_dly_wait(50);
        }

        /*Release the mutex*/
        os_mut_release(mut_GLCD);
        /* Add Delay          */
        os_dly_wait(50);

        /************** END ******************/
    }
}

/*---------------------------------------------------------------------------
 *        Task C: Push buttons state
 *---------------------------------------------------------------------------*/
__task void taskC(void) {
    /********* YOUR CODE GOES HERE **********/
    /* wait for an event flag 0x0001       */
    os_evt_wait_and(0x0001,0xffff);
    /************** END ******************/
    for (;;) {            /* endless loop*/
        /********* YOUR CODE GOES HERE **********/
        /*Acquire and lock the LCD Mutex             */
//        os_mut_init(mut_GLCD);
        os_mut_wait(mut_GLCD, 0xffff);
        /*Set background color, set text color*/

        /*Display "Task C started"          */
//        GLCD_DisplayString(5,0,1,"Task C started");
        GLCD_SetBackColor(Orange);
        GLCD_SetTextColor(White);
        GLCD_DisplayString(7,0,1, "                    ");
        GLCD_SetBackColor(Orange);
        GLCD_SetTextColor(White);
        GLCD_DisplayString(8,0,1, " Task C started     ");
```

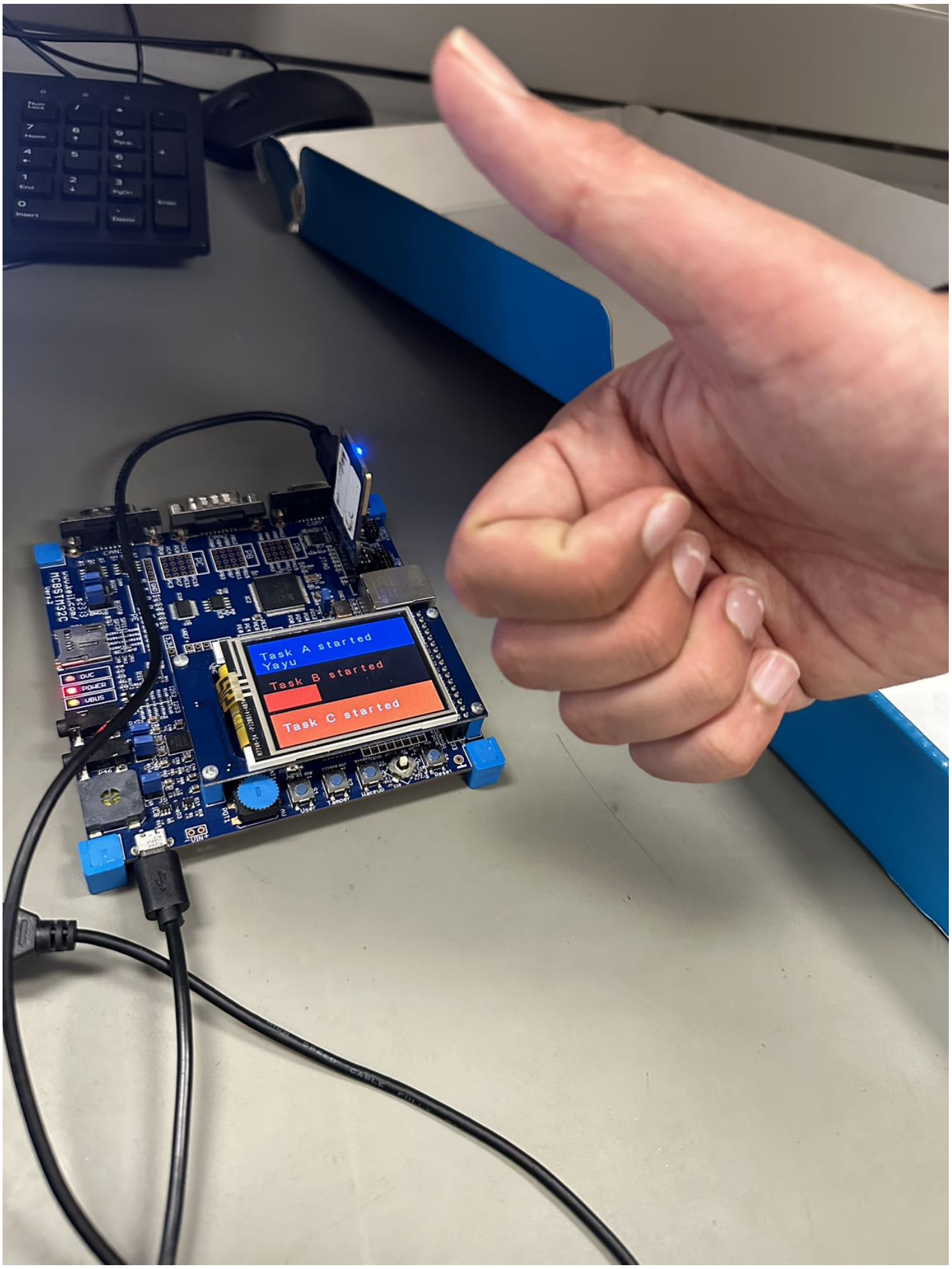
```
        GLCD_SetBackColor(Orange);
        GLCD_SetTextColor(White);
        GLCD_DisplayString(9,0,1, "                    ");
        os_mut_release(mut_GLCD);

        /*display User, Tamper, WakeUp, buttons state*/
        if(!(GPIOB->IDR & (1<<7))){
            os_mut_wait(mut_GLCD, 0xffff);
            GLCD_SetBackColor(Orange);
            GLCD_SetTextColor(White);
            GLCD_DisplayString(9,0,1, " User Pressed       ");
            os_mut_release(mut_GLCD);
            os_dly_wait(50);
        }else if(!(GPIOC->IDR & (1<<13))){
            os_mut_wait(mut_GLCD, 0xffff);
            GLCD_SetBackColor(Orange);
            GLCD_SetTextColor(White);
            GLCD_DisplayString(9,0,1, " Tamper Pressed     ");
            os_dly_wait(50);
            os_mut_release(mut_GLCD);
        }else if((GPIOA->IDR & (1<<0))){
            os_mut_wait(mut_GLCD, 0xffff);
            GLCD_SetBackColor(Orange);
            GLCD_SetTextColor(White);
            GLCD_DisplayString(9,0,1, " Wakeup Pressed     ");
            os_mut_release(mut_GLCD);
            os_dly_wait(50);
        }
        /*Release the mutex*/
        os_mut_release(mut_GLCD);

        os_dly_wait(50);

        /********** END **********/
    }
}
/*---------------------------------------------------------------------------
 * end of file
 *---------------------------------------------------------------------------*/
```

- Screenshot

- 
Task A started
Yayu
Task B started
Task C started