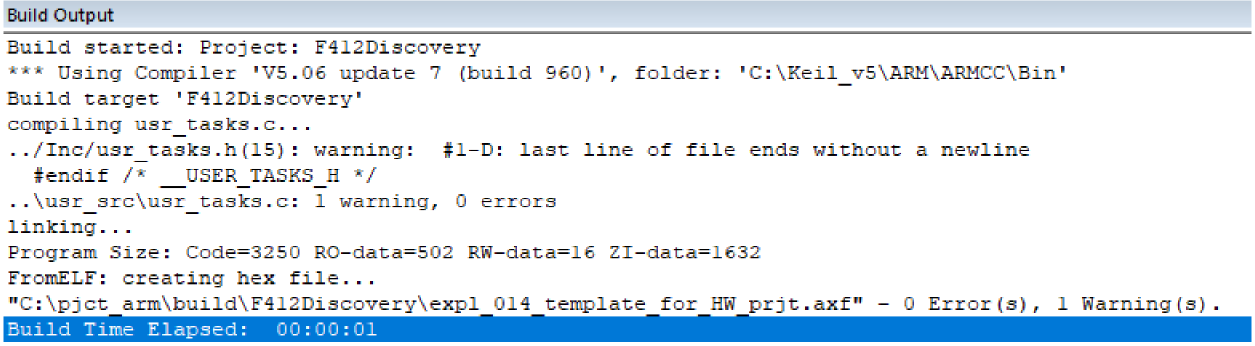
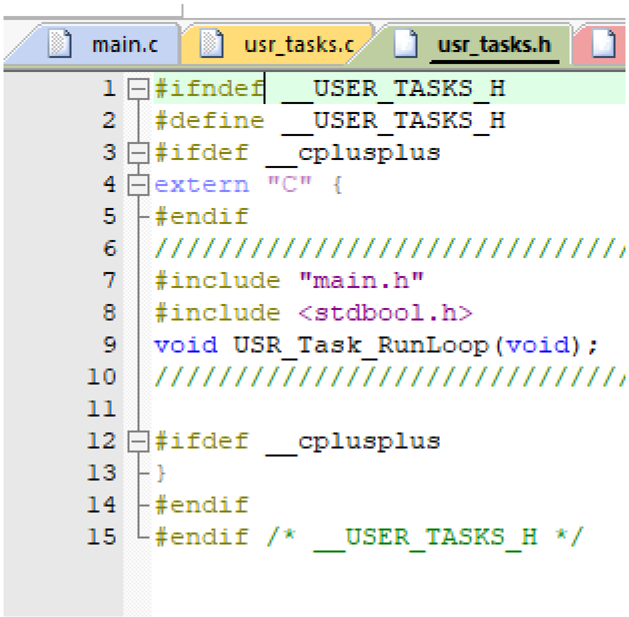
Worksheet #5 Task Screenshots & Pictures Jeremiah Webb

Task 1:



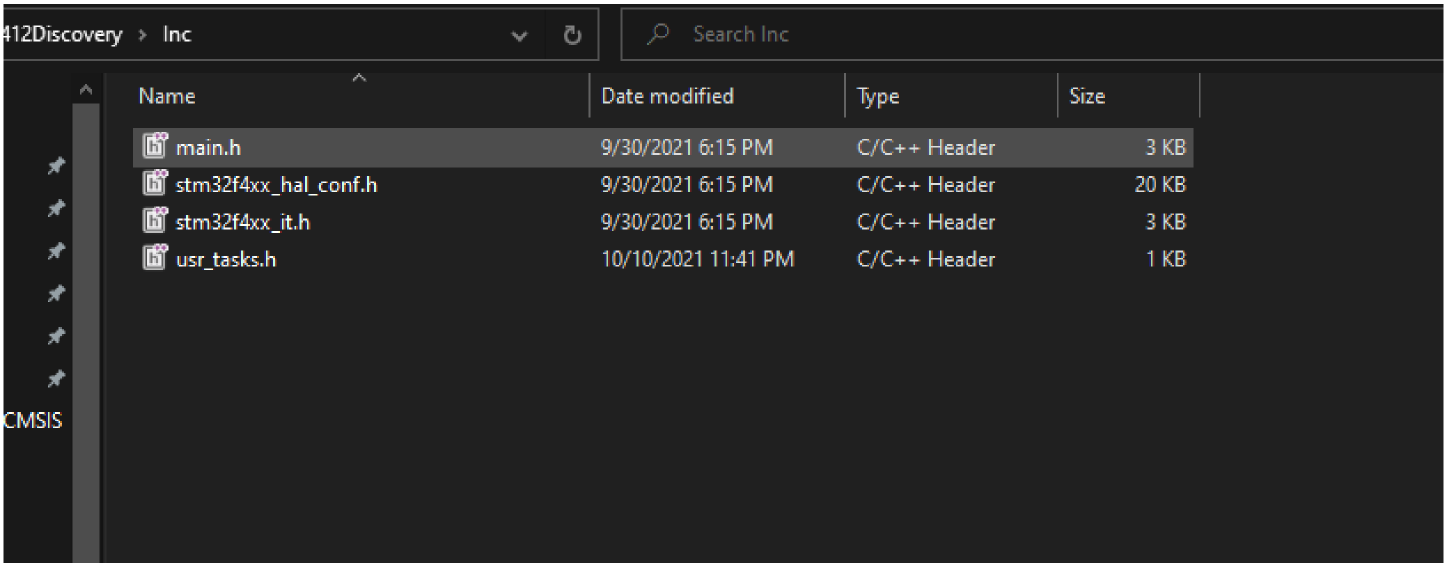
```
Build started: Project: F412Discovery
*** Using Compiler 'V5.06 update 7 (build 960)', folder: 'C:\Keil_v5\ARM\ARMCC\Bin'
Build target 'F412Discovery'
compiling usr_tasks.c...
../Inc/usr_tasks.h(15): warning:  #1-D: last line of file ends without a newline
  #endif /* __USER_TASKS_H */
..\usr_src\usr_tasks.c: 1 warning, 0 errors
linking...
Program Size: Code=3250 RO-data=502 RW-data=16 ZI-data=1632
FromELF: creating hex file...
"C:\pjct_arm\build\F412Discovery\expl_014_template_for_HW_prjt.axf" - 0 Error(s), 1 Warning(s).
Build Time Elapsed:  00:00:01
```

Task 2:

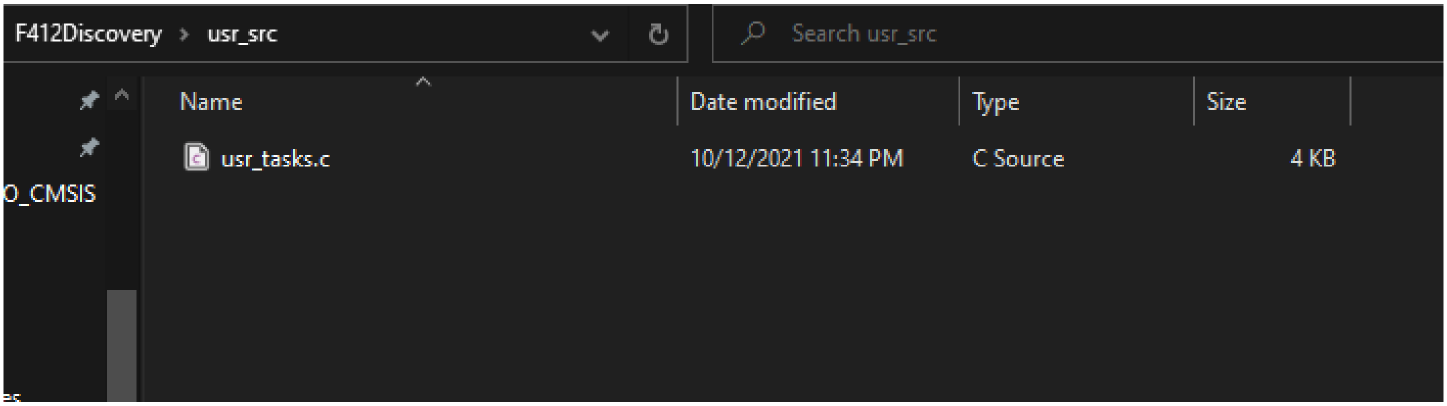


```
#ifndef __USER_TASKS_H
#define __USER_TASKS_H
#ifdef __cplusplus
extern "C" {
#endif
//////////////////////////////
#include "main.h"
#include <stdbool.h>
void USR_Task_RunLoop(void);
//////////////////////////////

#ifdef __cplusplus
}
#endif
#endif /* __USER_TASKS_H */
```

For some reason, when I put usr_tasks.h into the src folder, the program didn't work, so I put it in the inc folder.

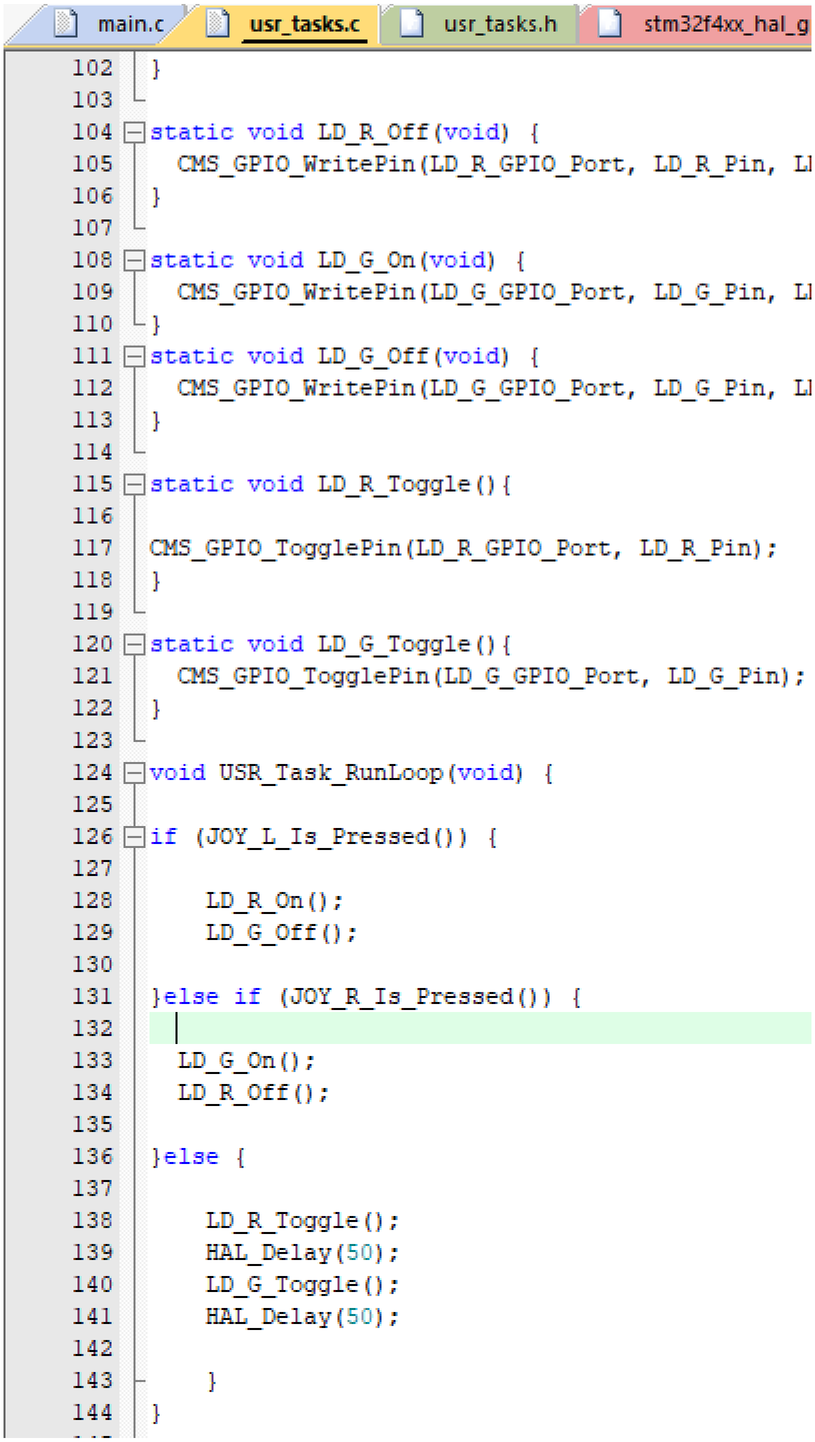


| Name | Date modified | Type | Size |
|---|---|---|---|
| main.h | 9/30/2021 6:15 PM | C/C++ Header | 3 KB |
| stm32f4xx_hal_conf.h | 9/30/2021 6:15 PM | C/C++ Header | 20 KB |
| stm32f4xx_it.h | 9/30/2021 6:15 PM | C/C++ Header | 3 KB |
| usr_tasks.h | 10/10/2021 11:41 PM | C/C++ Header | 1 KB |

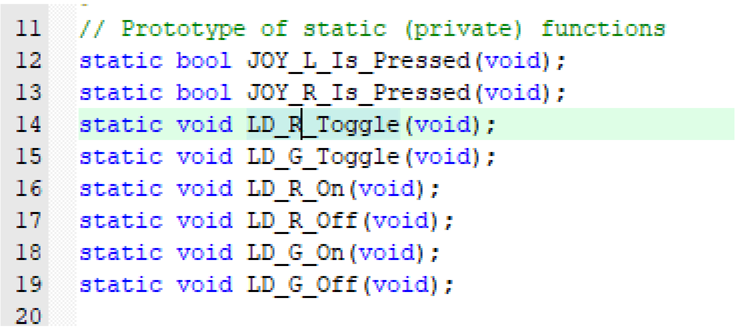


| Name | Date modified | Type | Size |
|---|---|---|---|
| usr_tasks.c | 10/12/2021 11:34 PM | C Source | 4 KB |

Task 3.1:

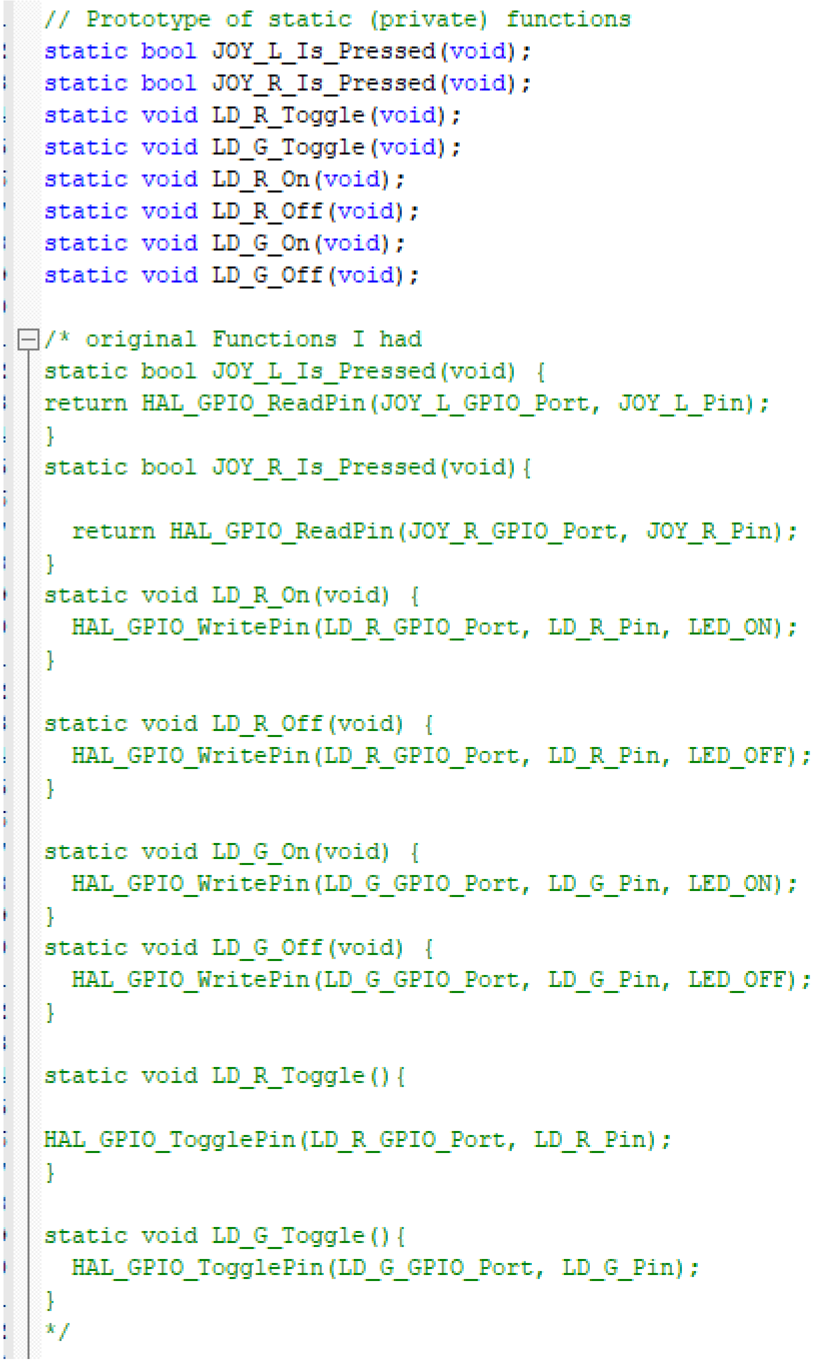


```
}

static void LD_R_Off(void) {
  CMS_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, LI
}

static void LD_G_On(void) {
  CMS_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, LI
}
static void LD_G_Off(void) {
  CMS_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, LI
}

static void LD_R_Toggle(){

CMS_GPIO_TogglePin(LD_R_GPIO_Port, LD_R_Pin);
}

static void LD_G_Toggle(){
  CMS_GPIO_TogglePin(LD_G_GPIO_Port, LD_G_Pin);
}

void USR_Task_RunLoop(void) {

if (JOY_L_Is_Pressed()) {

    LD_R_On();
    LD_G_Off();

}else if (JOY_R_Is_Pressed()) {

  LD_G_On();
  LD_R_Off();

}else {

    LD_R_Toggle();
    HAL_Delay(50);
    LD_G_Toggle();
    HAL_Delay(50);

    }
}
```



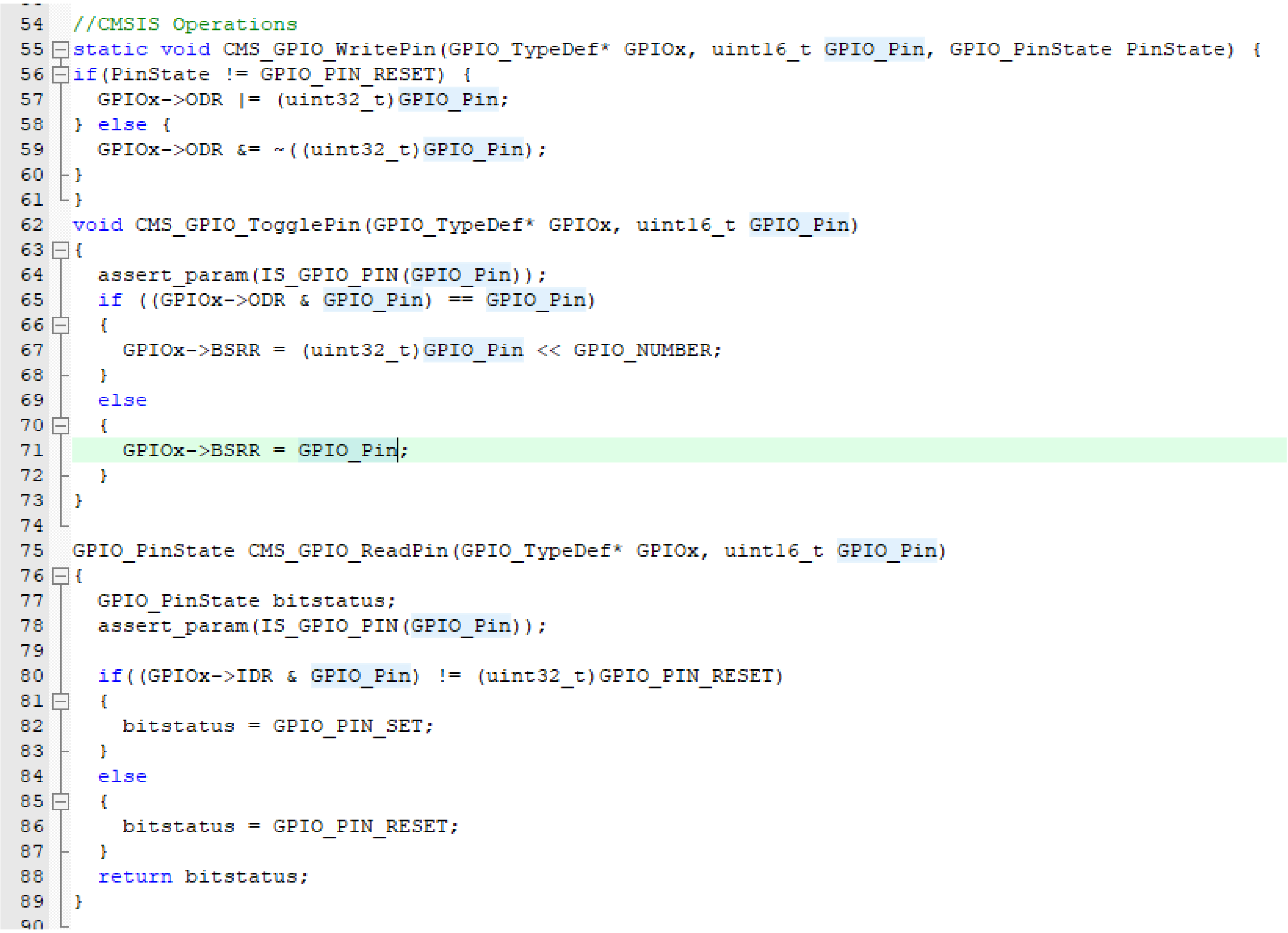
```
// Prototype of static (private) functions
static bool JOY_L_Is_Pressed(void);
static bool JOY_R_Is_Pressed(void);
static void LD_R_Toggle(void);
static void LD_G_Toggle(void);
static void LD_R_On(void);
static void LD_R_Off(void);
static void LD_G_On(void);
static void LD_G_Off(void);

```

Task 3.2

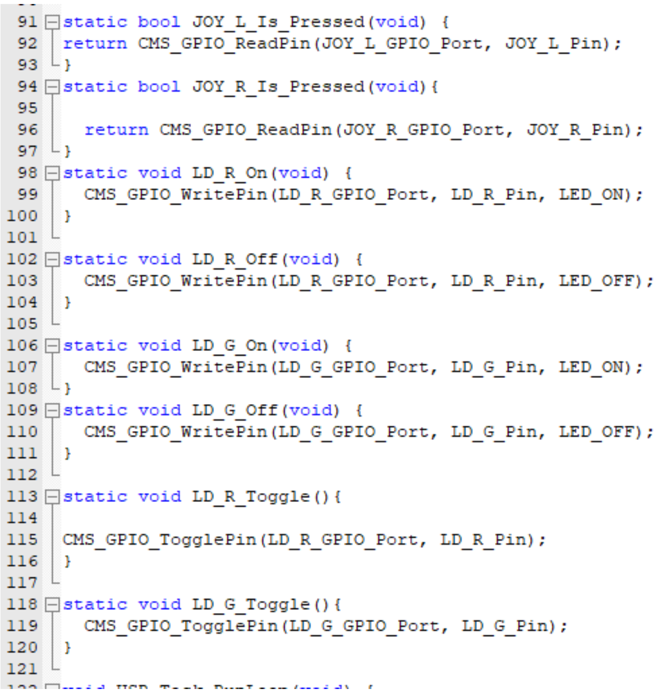


```
// Prototype of static (private) functions
static bool JOY_L_Is_Pressed(void);
static bool JOY_R_Is_Pressed(void);
static void LD_R_Toggle(void);
static void LD_G_Toggle(void);
static void LD_R_On(void);
static void LD_R_Off(void);
static void LD_G_On(void);
static void LD_G_Off(void);

/* original Functions I had
static bool JOY_L_Is_Pressed(void) {
return HAL_GPIO_ReadPin(JOY_L_GPIO_Port, JOY_L_Pin);
}
static bool JOY_R_Is_Pressed(void){

  return HAL_GPIO_ReadPin(JOY_R_GPIO_Port, JOY_R_Pin);
}
static void LD_R_On(void) {
  HAL_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, LED_ON);
}

static void LD_R_Off(void) {
  HAL_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, LED_OFF);
}

static void LD_G_On(void) {
  HAL_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, LED_ON);
}
static void LD_G_Off(void) {
  HAL_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, LED_OFF);
}

static void LD_R_Toggle(){

HAL_GPIO_TogglePin(LD_R_GPIO_Port, LD_R_Pin);
}

static void LD_G_Toggle(){
  HAL_GPIO_TogglePin(LD_G_GPIO_Port, LD_G_Pin);
}
*/
```

# Task 3.3

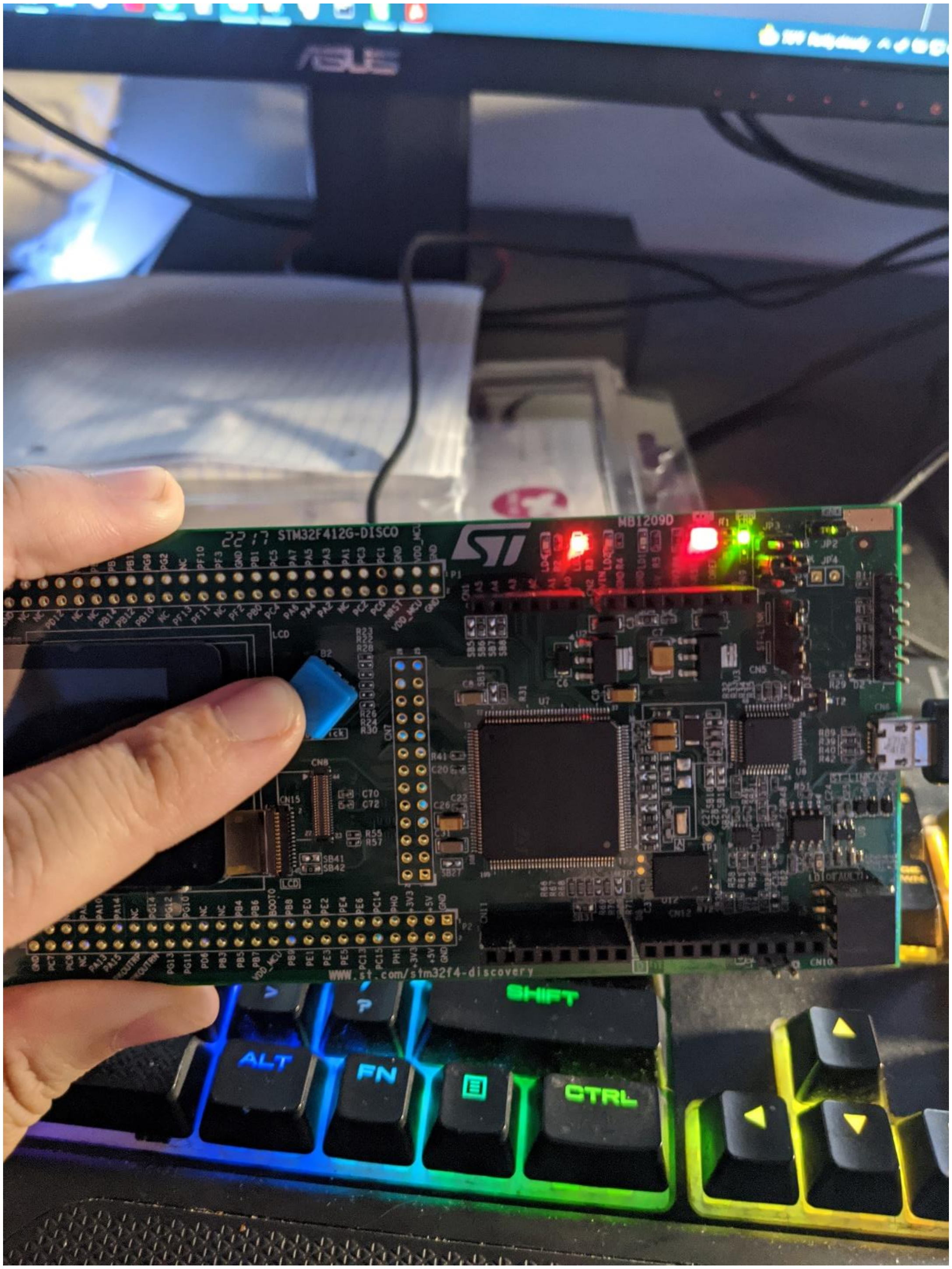


```
//CMSIS Operations
static void CMS_GPIO_WritePin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin, GPIO_PinState PinState) {
if(PinState != GPIO_PIN_RESET) {
  GPIOx->ODR |= (uint32_t)GPIO_Pin;
} else {
  GPIOx->ODR &= ~((uint32_t)GPIO_Pin);
}
}
void CMS_GPIO_TogglePin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin)
{
  assert_param(IS_GPIO_PIN(GPIO_Pin));
  if ((GPIOx->ODR & GPIO_Pin) == GPIO_Pin)
  {
    GPIOx->BSRR = (uint32_t)GPIO_Pin << GPIO_NUMBER;
  }
  else
  {
    GPIOx->BSRR = GPIO_Pin;
  }
}

GPIO_PinState CMS_GPIO_ReadPin(GPIO_TypeDef* GPIOx, uint16_t GPIO_Pin)
{
  GPIO_PinState bitstatus;
  assert_param(IS_GPIO_PIN(GPIO_Pin));

  if((GPIOx->IDR & GPIO_Pin) != (uint32_t)GPIO_PIN_RESET)
  {
    bitstatus = GPIO_PIN_SET;
  }
  else
  {
    bitstatus = GPIO_PIN_RESET;
  }
  return bitstatus;
}

```

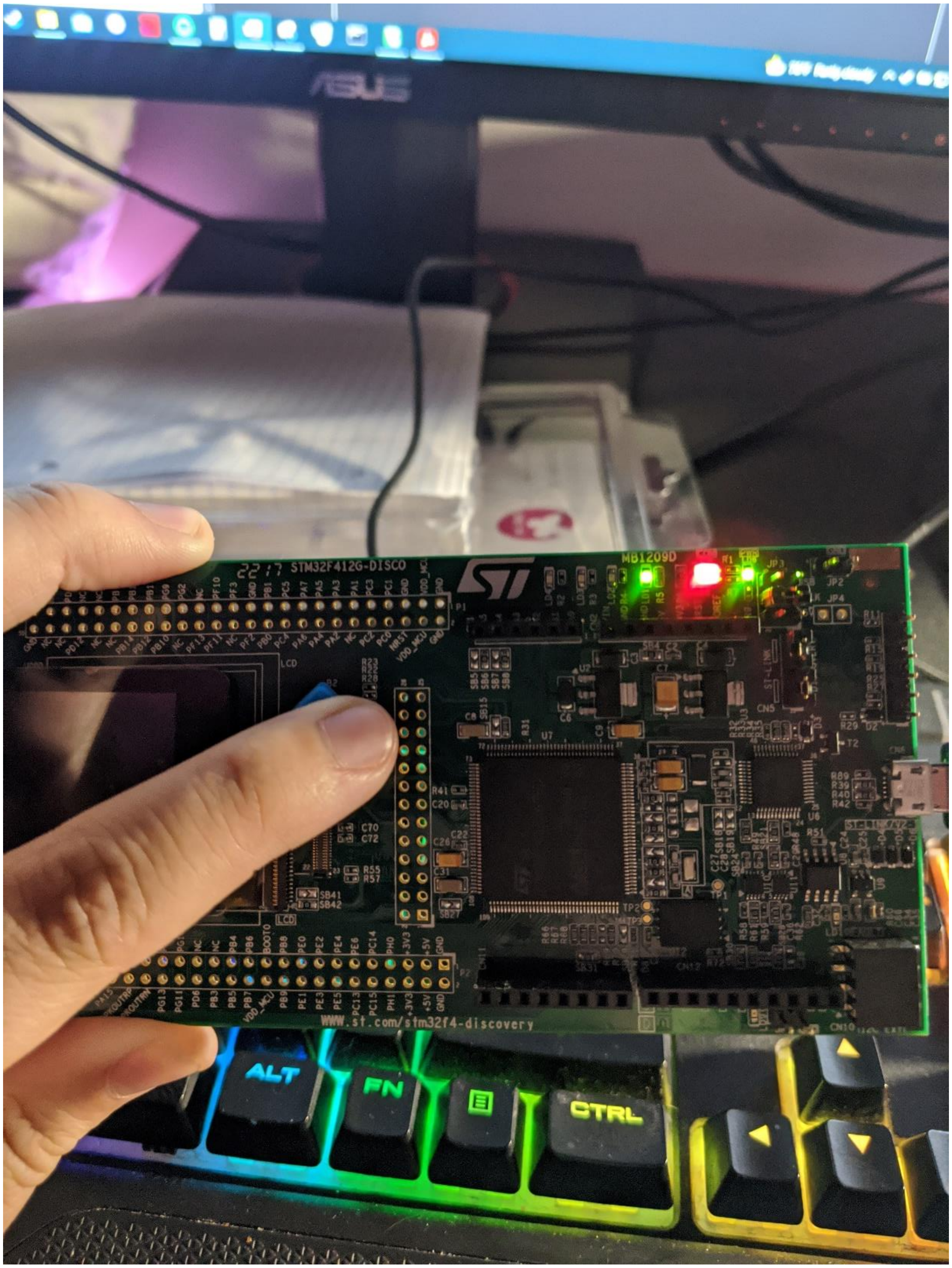


```
static bool JOY_L_Is_Pressed(void) {
return CMS_GPIO_ReadPin(JOY_L_GPIO_Port, JOY_L_Pin);
}
static bool JOY_R_Is_Pressed(void){

  return CMS_GPIO_ReadPin(JOY_R_GPIO_Port, JOY_R_Pin);
}
static void LD_R_On(void) {
  CMS_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, LED_ON);
}

static void LD_R_Off(void) {
  CMS_GPIO_WritePin(LD_R_GPIO_Port, LD_R_Pin, LED_OFF);
}

static void LD_G_On(void) {
  CMS_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, LED_ON);
}
static void LD_G_Off(void) {
  CMS_GPIO_WritePin(LD_G_GPIO_Port, LD_G_Pin, LED_OFF);
}

static void LD_R_Toggle(){

CMS_GPIO_TogglePin(LD_R_GPIO_Port, LD_R_Pin);
}

static void LD_G_Toggle(){
  CMS_GPIO_TogglePin(LD_G_GPIO_Port, LD_G_Pin);
}

void USR_Task_DualLoop(void) {
```

Photos Showing Functionality:



Pressing Left, Turn on Red, turn off Green.



Pressing Right, Green is on, red is off.