# 임베디드시스템 설계 및 실험
# 텀프로젝트 보고서

001분반 2조
201924586 조주영
201824170 정다현
201824441 김승혁
202155626 홍진욱

1. 주제
   A. 공부시간 측정 스탠드

2. 목적
   A. 수업시간에 배운 보드의 기능과 여러 센서들을 사용해 하드웨어를 개발한다.
   B. 블루투스를 이용해 휴대폰과 통신할 수 있는 하드웨어를 개발한다.
   C. 사람 또는 주변 환경을 인지하여 불빛을 자동으로 조절하고, 스탠드를 사용하는 학생들에게 필요한 여러가지 기능을 제공하는 하드웨어를 개발한다.

3. 내용 및 코드
   A. 기본적인 설정 코드

```
void sendDataUART1(uint16_t data);
void RCC_Configure(void);
void GPIO_Configure(void);
void USART1_Init(void);
void USART2_Init(void);
void NVIC_Configure(void);
void EXTI_Configure(void);
void sendPhone(char* buf);
void delay_2();
void lightOff();
void lightOn();
void sendDataUSART2(uint16_t data);
void delay();
```

i. RCC_Configure

```
void RCC_Configure(void)
{
  //port
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOA,ENABLE);
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB,ENABLE);
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOC,ENABLE);
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOD,ENABLE);
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOE,ENABLE);

  //ADC1
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_ADC1,ENABLE);
  RCC_AHBPeriphClockCmd(RCC_AHBPeriph_DMA1,ENABLE);

  //bluetooth
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_USART1,ENABLE);
  RCC_APB1PeriphClockCmd(RCC_APB1Periph_USART2,ENABLE);
  RCC_APB2PeriphClockCmd(RCC_APB2Periph_AFIO,ENABLE);

  //timer
  RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2,ENABLE);
  RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM3,ENABLE);
  RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM4,ENABLE);
  RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM5,ENABLE);
}
```

GPIO 포트, ADC1, DMA,USART1,2 타이머 동작을 위한 RCC configure 수행

ii. GPIO_Configure

```
void GPIO_Configure(void)
{
  GPIO_InitTypeDef GPIO_InitStructure;

  //input
  //적외선 센서 PA0 - channel 0
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPD;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOA, &GPIO_InitStructure);

  //조도 센서 PA1 - channel 1
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_1;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AIN;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOA, &GPIO_InitStructure);

  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_8;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP; //output-triger
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOB, &GPIO_InitStructure);

  //터치 센서 PB1 - channel 9
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_1;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AIN;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOB, &GPIO_InitStructure);

  //버튼 1,3
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_4 | GPIO_Pin_13;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
  GPIO_Init(GPIOC, &GPIO_InitStructure);

  //버튼 2
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10 ;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
  GPIO_Init(GPIOB, &GPIO_InitStructure);

  //output 밝기 조절 안되는 핀,모드
  //led
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_7; //green
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOD, &GPIO_InitStructure);

  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_8; //red
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOD, &GPIO_InitStructure);

  GPIO_InitStructure.GPIO_Pin =GPIO_Pin_9; //blue
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOD, &GPIO_InitStructure);
```

```
  //UART1
  //TX
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_9;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOA, &GPIO_InitStructure);

  //RX
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_10;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
  GPIO_Init(GPIOA, &GPIO_InitStructure);

  //UART2
  //TX
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
  GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
  GPIO_Init(GPIOA, &GPIO_InitStructure);

  //RX
  GPIO_InitStructure.GPIO_Pin = GPIO_Pin_3;
  GPIO_InitStructure.GPIO_Mode = GPIO_Mode_IPU;
  GPIO_Init(GPIOA, &GPIO_InitStructure);
}
```

GPIO 설정하는 코드.

상세 내용은 주석 참고

iii. USART1_init

```
void USART1_Init(void)
{
    USART_InitTypeDef USART1_InitStructure;

    USART_Cmd(USART1, ENABLE);


    USART1_InitStructure.USART_BaudRate=9600;
    USART1_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART1_InitStructure.USART_StopBits = USART_StopBits_1;
    USART1_InitStructure.USART_HardwareFlowControl =
USART_HardwareFlowControl_None;
    USART1_InitStructure.USART_Parity = USART_Parity_No;
    USART1_InitStructure.USART_Mode= USART_Mode_Rx|USART_Mode_Tx;
    USART_Init(USART1, &USART1_InitStructure);

    USART_ITConfig(USART1, USART_IT_RXNE, ENABLE);
}
```

USART1 사용을 위한 init 코드

iv. USART2_init

```
void USART2_Init(void)
{
    USART_InitTypeDef USART2_InitStructure;

    // Enable the USART2 peripheral
    USART_Cmd(USART2, ENABLE);
```

```
        USART2_InitStructure.USART_BaudRate=9600;
    USART2_InitStructure.USART_WordLength = USART_WordLength_8b;
    USART2_InitStructure.USART_StopBits = USART_StopBits_1;
    USART2_InitStructure.USART_HardwareFlowControl =
USART_HardwareFlowControl_None;
    USART2_InitStructure.USART_Parity = USART_Parity_No;
    USART2_InitStructure.USART_Mode= USART_Mode_Rx|USART_Mode_Tx;
    USART_Init(USART2, &USART2_InitStructure);

    USART_ITConfig(USART2, USART_IT_RXNE, ENABLE);
}
```

이하 내용은 첨부한 코드 참고.

v. NVIC_Configure

vi. EXTI_Configure

vii. sendPhone

viii. delay_2

ix. lightOff

x. sendDataUART1

B. 적외선 센서를 이용하여 사용자를 인식한다.

```
while(1){
  if(GPIO_ReadInputDataBit(GPIOA, GPIO_Pin_0) == 1) {
    break;
  }
}
```

i. PA0의 출력 값이 1인경우 while문을 탈출한다.

C. 터치 센서를 이용하여 사용자를 인식하여 자동으로 led를 점등한다.

```
while(1){
    if(ADC_Value[2] > 2000) {
        LCD_ShowNum(100,70, 1,4,BLACK, WHITE);

        lightOn();
        delay();
        break;
    }
}

 LCD_Clear(WHITE);
lightOff();
```

i. ADC_Value[2]는 터치센서의 ADC 값

ii. 2000을 넘어가면 LCD에 1출력 후, LED를 하얀색으로 점등한 뒤 잠시 뒤에 꺼지고, LCD를 초기화한 뒤 LED를 소등한다.

D. 보드의 버튼을 이용하여 공부모드를 설정한다.

```
while(1) {
  if(!empty_check) {
    empty_timer =0;
  }
  if(show1_flag){
```

```
    show1();
  } else if(show3_flag){
    show3();
  } else if(show2_flag){
    show2();
  }
```

위 코드는 항시 반복하게 된다.

i. 집중모드

1. 공부타이머 작동
2. LED 초록색 점등

```
void EXTI4_IRQHandler(void) {
  if(EXTI_GetITStatus(EXTI_Line4) != RESET) {
    if((GPIO_ReadInputDataBit(GPIOC,GPIO_Pin_4) == Bit_RESET)) {
        LCD_Clear(WHITE);
       green_on = true;
       red_on = false;
       blue_on =false;
       show2_flag = false;
       show3_flag = false;
       show1_flag = true;
       studytimer_on =true;
    }
    EXTI_ClearITPendingBit(EXTI_Line4);
  }
}
```

인터럽트 핸들러를 활용하여 버튼1을 눌렀을 때, 각 flag들을 설정해준다.

```
void TIM2_IRQHandler(void) { // 타이머 intterrupt handler (1s):공부시간측정,
알람, 졸음방지, 자리비움


  if(TIM_GetITStatus(TIM2, TIM_IT_Update) != RESET){
    if(studytimer_on == true) {
      study_timer++;
    }

    //. 자리비움 감지
    if(empty_check){
      empty_timer++;
    }
    if(empty_timer >= 5){
       LCD_Clear(WHITE);
       blue_on= true;
       green_on = false;
       red_on =false;
       show2_flag = false;
       show1_flag = false;
       show3_flag = true;
       studytimer_on = false;
       empty_timer=0;
    }


    TIM_ClearITPendingBit(TIM2,TIM_IT_Update);
  }
}
```

또한 타이머 인터럽트를 활용하여 studytimer를 증가시킨다.

ii. 휴식모드

1. 공부타이머 일시정지
2. LED 파란색 점등

```
if(EXTI_GetITStatus(EXTI_Line13) != RESET) {
  if((GPIO_ReadInputDataBit(GPIOC,GPIO_Pin_13) == Bit_RESET)) {
      LCD_Clear(WHITE);
     blue_on= true;
     green_on = false;
     red_on =false;
     show2_flag = false;
     show1_flag = false;
     show3_flag = true;
     studytimer_on = false;
  }
  EXTI_ClearITPendingBit(EXTI_Line13);
}
```

EXTI15_10_IRQHandler 함수안에 위코드를 작성해서 버튼 3을 눌렀을 때, 각종 플래그를 설정해준다.

따라서 studytimer는 일시정지가 된다. 또한, main의 while(1)에서 show3()을 수행한다.

iii. 휴식모드 > 초기모드

1. 공부타이머 출력
2. LED 빨간색 점등

```
uint16_t tmp = study_timer;

study_timer_hour=tmp/3600;
tmp=tmp%3600;
study_timer_min=tmp/60;
tmp=tmp%60;
study_timer_sec=tmp;

char str[30];

sprintf(str, "study time : %uh %um %us \r\n", study_timer_hour,
study_timer_min, study_timer_sec);

if(EXTI_GetITStatus(EXTI_Line10) != RESET) {
   if((GPIO_ReadInputDataBit(GPIOB,GPIO_Pin_10) == Bit_RESET)) {
     if(blue_on){
         LCD_Clear(WHITE);
       red_on = true;
       green_on = false;
       blue_on =false;
       show2_flag = true;
       show3_flag = false;
       show1_flag = false;
       for(int i=0; i<30; ++i){
         sendDataUART1(str[i]);
       }
       sendPhone(str);
       study_timer = 0;
     }
```

```
    }
    EXTI_ClearITPendingBit(EXTI_Line10);
  }
```

EXTI15_10_IRQHandler 함수 안에 위 코드를 작성하여 공부 시간을 출력하도록 한다

If(blue_on) 을 사용해 휴식 모드일때에만 동작하게 한다.

공부시간을 usart1, 2 를 활용해 putty 혹은 블루투스 연결된 기기로 출력한다.

타이머를 초기화한다.

E. 보드의 LCD에 사용자가 설정한 타이머가 나타난다.

```
void show1(){

  GPIO_ResetBits(GPIOD,GPIO_Pin_7);
  GPIO_SetBits(GPIOD,GPIO_Pin_8);
  GPIO_SetBits(GPIOD,GPIO_Pin_9);

  int a = GPIO_ReadInputDataBit(GPIOA,GPIO_Pin_0);
  if(tmp_empty == 0){
     if(a == 1){
       empty_check = false;
       tmp_empty = a;
     }
     if(a==0){
       empty_check = true;
     }
    }
    if(tmp_empty == 1){

     if(a == 0 ){
       empty_check = false;
       tmp_empty = a;
     }
     if(a==1){
       empty_check = true;
     }
    }

  LCD_ShowNum(80, 60, empty_timer, 4, BLACK, WHITE);



  show_timer();
}

void show2(){
  GPIO_ResetBits(GPIOD,GPIO_Pin_8);
  GPIO_SetBits(GPIOD,GPIO_Pin_7);
  GPIO_SetBits(GPIOD,GPIO_Pin_9);
    show_timer();



  // 공부시간 출력
```

```
  // 휴대폰으로 공부시간 전송
  // 초기상태로 돌아감
}
void show3(){
  empty_check=false;
  GPIO_SetBits(GPIOD,GPIO_Pin_7);
  GPIO_SetBits(GPIOD,GPIO_Pin_8);
  GPIO_ResetBits(GPIOD,GPIO_Pin_9);


  LCD_ShowNum(80, 60, empty_timer, 4, BLACK, WHITE);

  show_timer();
}
```

각 모드에 대한 코드이다.

집중 모드 : show1

휴식 모드 : show3

종료 : show2 ( show3인 경우에서 버튼2를 눌렀을 때)

4. 사용한 센서

4. 사용센서

1) 모션 인식 – 적외선 센서: 인체감지센서모듈 HC-SR501 [SZH-EK052]

(링크: https://www.devicemart.co.kr/goods/view?no=1287086)

2) 모션 인식 – 터치 센서: TTP223B 아두이노 터치 센서 모듈 [SZH-SSBH-028]

(링크: https://www.devicemart.co.kr/goods/view?no=1327426)


3) 화면 출력 – 실습시간에 사용했던 TFT-LCD

4) LED 전구: DG-53N RGB 262C-A9001 (10개)

(링크: https://www.devicemart.co.kr/goods/view?no=6224)

5. 기본 시나리오

A. 스탠드 사용자 인지

i. 적외선 센서를 이용해 사용자를 감지하고 터치 센서에 입력이 주어지면 빨간색 LED가 깜빡이고 LCD 타이머가 화면에 실행된다.

B. 보드의 버튼1번(공부 모드, 시작·재개)과 3번(휴식 모드, 일시 정지)을 누르며 사용자 상황에 맞는 (공부모드일 때 초록색, 휴식모드일 때 파란색) 스탠드 led 색상을 조정한다. 집중 모드일 때는 카운트가 시작되고 숫자가 계속 올라가지만 휴식 모드가 되었을 때는 일시 정지한다.
만약 집중 모드가 켜져있고 5초동안 자리비움이 감지될 경우(적외선 센서로 감지) 자동으로 휴식모드로 들어간다.

C. 일시 정지 상태에서 보드의 버튼 2번을 누르면 종료를 수행하며 LCD에 숫자가 0으로 초기화되고 블루투스로 총 공부한 시간을 출력하고 빨간색 LED를 점등한다.