2021년 2학기
임베디드 시스템설계 및 실험
11주차 실험보고서

201712159 조현우
201724566 전승윤
201724611 최호진
201924650 박지호

목차

1. 실험 목적

Timer 이해 및 실습

2. 실험 과정

1) TFT LCD에 team 이름, LED 토글 ON/OFF 상태, LED ON/OFF 버튼 생성
2) LED ON 버튼 터치 시 TIM2 interrupt를 활용하여 LED 2개 제어
    a. 1초마다 LED1 Toggle
    b. 1초마다 LED2 Toggle
3) LED OFF 버튼 터치 시 LED Toggle 동작 해제
4) 서보 모터 제어

3. 실험 세부 내용

1) TFT LCD에 team 이름, LED 토글 ON/OFF 상태, LED ON/OFF 버튼 생성

```
char msg1[] = "THU_Team03";
LCD_ShowString(0, 0, msg1, color[2], color[0] );

char msgOn[] = "ON ";
char msgOff[] = "OFF";

char msgBtn[] = "BTN";
LCD_DrawRectangle(16, 64, 64, 112);
LCD_ShowString(32, 76, msgBtn, color[3], color[0] );

uint16_t x, y;

while (1) {
  if(isOnOff) {
    LCD_ShowString(0, 32, msgOn, color[3], color[0] );
  }
  else {
    LCD_ShowString(0, 32, msgOff, color[3], color[0] );
  }

  Touch_GetXY(&x, &y, 1);
  Convert_Pos(x, y, &x, &y);
  if(x >= 16 && x <= 64 && y >= 64 && y <= 112) {
    isOnOff = !isOnOff;
    led2 = 0;
  }
}
```

그림 1 - Team, Button 출력

그림 1은 LCD에 team이름, LED의 상태 및 버튼을 생성한 코드이다. LCD_DrawRectangler()와 LCD_ShowString()으로 보여진 버튼을 지정된 x, y안 공간을 touch할때마다 int isOnOff를 0, 1로 바꾸어 OFF, ON으로 바꾸어 준다.

2) LED ON 버튼 터치 시 TIM2 interrupt를 활용하여 LED 2개 제어

```
/* LED 1, 2 */
/* PD2, 3     */
RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOD, ENABLE);


/* TIM2_CH2 */
/* PB0 */
RCC_APB2PeriphClockCmd(RCC_APB2Periph_GPIOB, ENABLE);
RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM2, ENABLE);
RCC_APB1PeriphClockCmd(RCC_APB1Periph_TIM3, ENABLE);

/* Alternate Function IO clock enable */
RCC_APB2PeriphClockCmd(RCC_APB2Periph_AFIO, ENABLE);
```

```
/* LED 1, 2 */
/* PD2, 3     */
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_2 | GPIO_Pin_3;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_Out_PP;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_Init(GPIOD, &GPIO_InitStructure);

/* TIM3 */
/* PB0 */
GPIO_InitStructure.GPIO_Pin = GPIO_Pin_0;
GPIO_InitStructure.GPIO_Mode = GPIO_Mode_AF_PP;
GPIO_InitStructure.GPIO_Speed = GPIO_Speed_50MHz;
GPIO_Init(GPIOB, &GPIO_InitStructure);
```

그림 2 - RCC, GPIO configure

그림 2는 RCC Enable, GPIO 설정 코드이다. RCC_Configure()에서는 LED를 사용해야할 Port D와 Timer를 사용해야할 Port B를 Enable해준다. GPIO_Configure()에서는 사용할 Port의 Pin번호와 GPIO_Mode를 부여해 GPIO_Init()의 인자로 넣어 원하는 설정을 한다. TIM2는 interrupt를 통해 LED를 제어하며 TIM3는 서브모터를 제어하기위한 용도이다.

```
NVIC_InitTypeDef NVIC_InitStructure;

// TODO: fill the arg you want
NVIC_PriorityGroupConfig(NVIC_PriorityGroup_0);

// TIM2_IRQn
// 'NVIC_EnableIRQ' is only required for USART setting
NVIC_EnableIRQ(TIM2_IRQn);
NVIC_InitStructure.NVIC_IRQChannel = TIM2_IRQn;
NVIC_InitStructure.NVIC_IRQChannelPreemptionPriority = 0; // TODO
NVIC_InitStructure.NVIC_IRQChannelSubPriority = 0; // TODO
NVIC_InitStructure.NVIC_IRQChannelCmd = ENABLE;
NVIC_Init(&NVIC_InitStructure);
```

그림 3 - NVIC 인터럽트 setting

그림 3은 NVIC를 이용한 TIM2 인터럽트 세팅이다. 동작이 동시에 일어나지 않으므로 모두 같은 PreemptionPriority와 Subpriority를 설정해 주었다.

```
TIM_TimeBaseStructure.TIM_Period = 10000;
TIM_TimeBaseStructure.TIM_Prescaler = 7200;
TIM_TimeBaseStructure.TIM_ClockDivision = 0;
TIM_TimeBaseStructure.TIM_CounterMode = TIM_CounterMode_Down;

TIM_TimeBaseInit(TIM2, &TIM_TimeBaseStructure);
TIM_ITConfig(TIM2, TIM_IT_Update, ENABLE);
TIM_ARRPreloadConfig(TIM2, ENABLE);
TIM_Cmd(TIM2, ENABLE);
```

$$\frac{1}{f_{clk}} \times prescaler \times period$$

$$\frac{1}{72Mhz} \times 7200 \times 10000 = 1[s]$$

그림 4 – TIM2 Alternate function사용

그림 4는 TIM2를 사용하기위한 Alternate function을 설정한 코드이다. 1초 기준으로 설정하기 위해 Period에는 10000, Prescaler에는 7200을 넣어주었다.

```
if(TIM_GetITStatus(TIM2, TIM_IT_Update) != RESET ) {
  if(isOnOff) {
    led2++;
    GPIO_SetBits(GPIOD, GPIO_Pin_2);
    if(led2 == 5) {
      led2 = 0;
      GPIO_SetBits(GPIOD, GPIO_Pin_3);
    }
    delay();
    GPIO_ResetBits(GPIOD, GPIO_Pin_2);
    if(led2 == 0) GPIO_ResetBits(GPIOD, GPIO_Pin_3);
  }
```

그림 5 – TIM2 interrupt Handler

TIM2의 interrupt handler인 TIM2_IRQHandler()함수이다. 앞서 언급한 isOnOff가 1일 경우 int led2 = 0을 하나씩 올려준다. LED 1번인 pin_2를 set해주어 LED를 켜준다. pin_2가 동작하는 동안 led2를 5까지 증가시킨다. 5에 다다르면 LED 2번인 pin_3을 set해주어 LED를 켜준다. delay()함수를 통해 0.5초동안 멈춰주고 pin_2를 reset해준다. led2 가 5일때는 pin_3도 reset해준다.

3) LED OFF 버튼 터치 시 LED Toggle 동작 해제

그림 1에서 isOnOff가 "1"일 때는 ON으로 표시되며 앞서 설명한 TIM2_IRQHandler()를 동작한다. Button을 다시 터치하면 isOnOff가 "0"이 되며 OFF로 표시되며 TIM2_IRQHandler()의 동작을 해제한다.

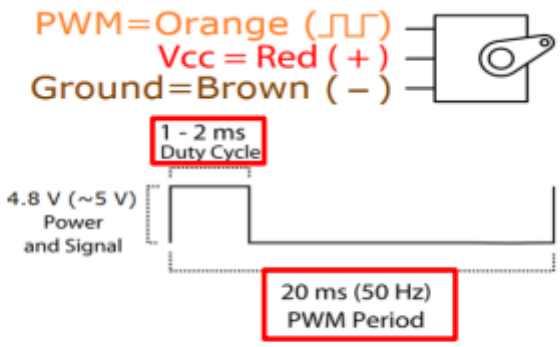
4) 서보 모터 제어

```
TIM_TimeBaseStructure.TIM_Period = 20000 - 1;
TIM_TimeBaseStructure.TIM_Prescaler = (uint16_t) (SystemCoreClock / 1000000) - 1;
TIM_TimeBaseInit(TIM3, &TIM_TimeBaseStructure);

TIM_OCInitStructure.TIM_OCMode = TIM_OCMode_PWM1;
TIM_OCInitStructure.TIM_OCPolarity = TIM_OCPolarity_High;
TIM_OCInitStructure.TIM_OutputState = TIM_OutputState_Enable;
TIM_OCInitStructure.TIM_Pulse = 1500; // us
TIM_OC3Init(TIM3, &TIM_OCInitStructure);

TIM_OC3PreloadConfig(TIM3, TIM_OCPreload_Disable);
TIM_ARRPreloadConfig(TIM3, ENABLE);
TIM_Cmd(TIM3, ENABLE);
```

그림 6 – TIM3 Alternate function사용

$$f_{clk} * \frac{1}{prescaler} * \frac{1}{period} = \text{주파수}[Hz]$$

그림 7 – General-purpose timers 주파수 식



Position "0" (1.5 ms pulse) is middle, "90" (~2 ms pulse) is all the way to the right, "-90" (~1 ms pulse) is all the way to the left.

그림 8 - PWM(Pulse Width Modulation)

그림 6에 서 TIM3의 period와 prescaler값을 설정해주어야 한다. 이를 구하기 위해 그림 7에서의 주파수 식을 활용했으며 서브모터의 주파수인 50Hz와 PWM period인 20ms(20000)을 활용해 prescaler값을 SystemCoreClock에 1000000을 나눈 값으로 설정해주었다. TIM3에 pluse값은 그림 8에서 중간인 "0"을 기준으로 1.5ms(1500)으로 설정해주었다.

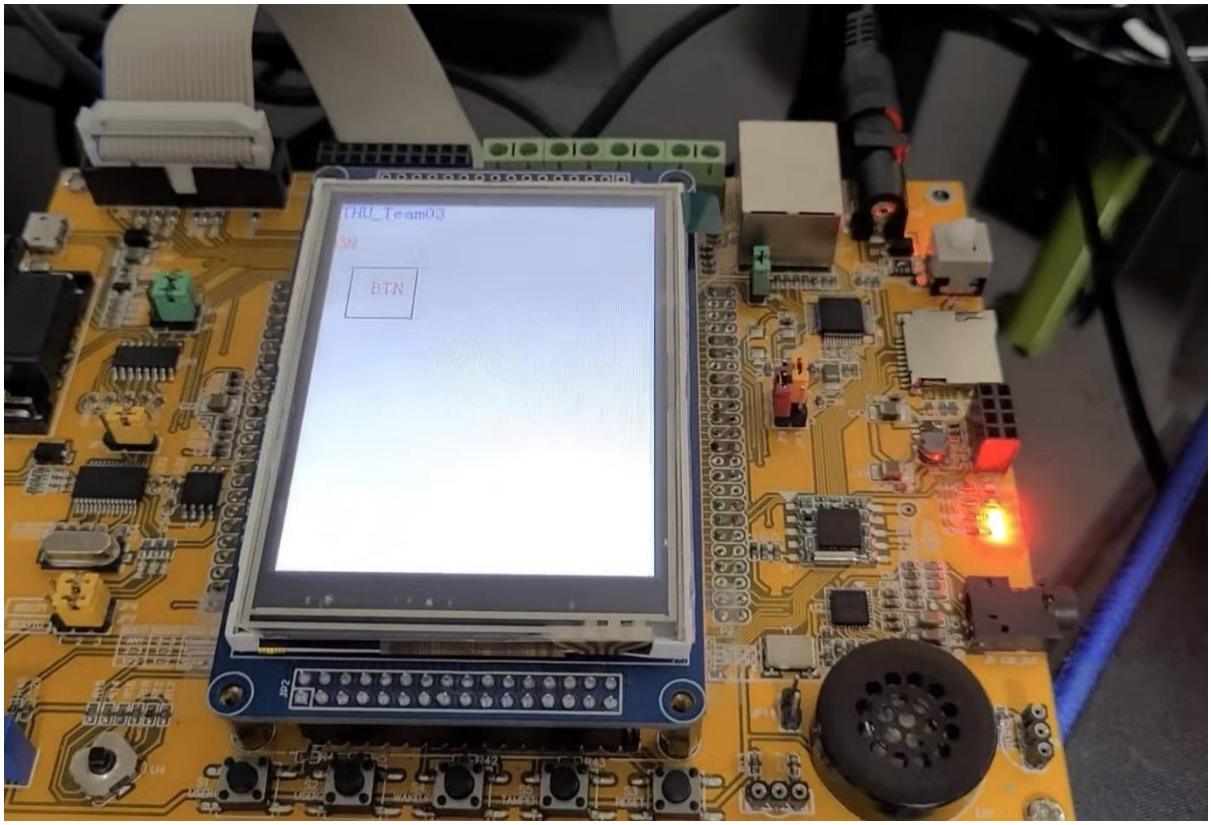
```
void change_pulse(uint16_t pulse){
  TIM_OCInitTypeDef TIM_OCInitStructure;
  TIM_OCInitStructure.TIM_OCMode = TIM_OCMode_PWM1;
  TIM_OCInitStructure.TIM_OCPolarity = TIM_OCPolarity_High;
  TIM_OCInitStructure.TIM_OutputState = TIM_OutputState_Enable;
  TIM_OCInitStructure.TIM_Pulse = pulse; // us
  TIM_OC3Init(TIM3, &TIM_OCInitStructure);
}
```

```
change_pulse(sub[cnt++]);
if (cnt == 2)
  cnt = 0;
```

그림 9 - chage_pulse()코드와 - TIM2_IRQHandler()내 chage_pulse코드

그림 9은 pluse를 바꿔주는 - chage_pulse()코드와 TIM2_IRQHandler()내 chage_pulse()사용한 코드이다. 전역변수로 uint16_t sub[2] = {1000, 2000}로 해주어 PWM의 duty cycle로 설정해주었다. TIM2_IRQHandler()에서 delay()동작을 수행할 때마다 change_pluse()에서 pluse값을 바꾸어 준다. 즉, 1초마다 0°에서 90°씩 반복하여 동작한다.

4. 실험 결과



BTN을 눌러 ON상태를 만들어주었고 Timer를 통해 led2가 5가 되어 LED 1,2가 모두 켜져있는 장면이다. 이번 실험을 통해서 Timer 이해 및 실습 방법에 대해 익힐 수 있었다. 서브모터와 전체적인 동작과정은 참조되어 있는 영상을 시청하길 바란다.