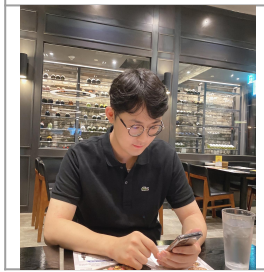
자동주차 지원을 위한 초음파 센서 기반 충돌 방지 경보 시스템

Ultrasonic Sensor-based Collision Avoidance Warning System for Automatic Parking Assistance

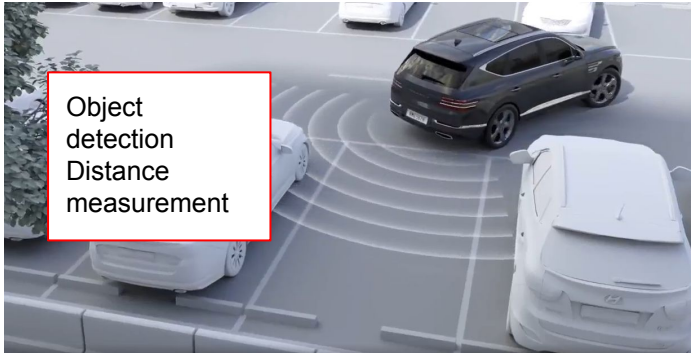
Group 1

Kim Donghwan	Kang Jinseoung	Jang Hyunyoung	Ryu Jungmin
			

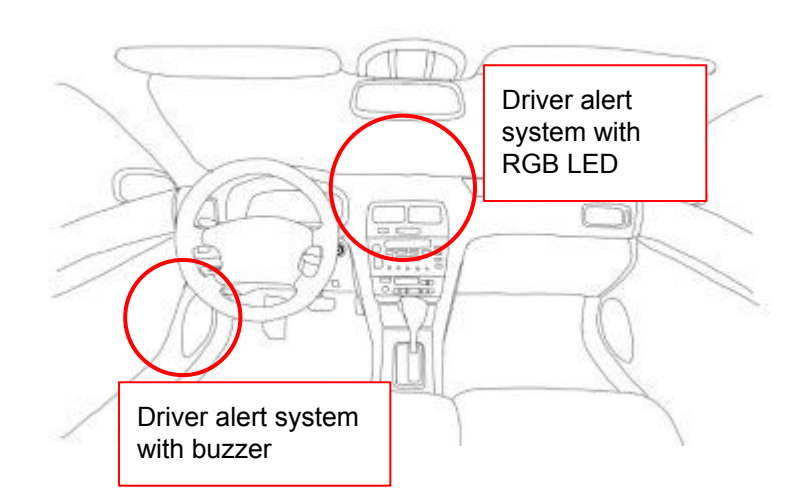
Contents

1 Introduction

- Illuminates the hazard lights to inform the surrounding area that they are in automatic parking (D12, D13 LED)
- Measure the distance from the surrounding objects using an ultrasonic sensor
- Operate buzzer and RGB LED for driver alert system
- Adjust the frequency of the buzzer and the color and frequency of the RGB LED according to the distance from the object
- Change the buzzer sound based on the drivers' preference (using potentiometer)
- Items used : GPIO, LED, Potentiometer, ADC, PWM, Buzzer, Ultrasonic Sensor



Hazard lights for notifying automatic parking (D12, D13 LED)



2 Function

1. **D12, D13 LED** blinking : Auto parking mode is activated

2. Using **ultrasonic sensor**, get distance between wall and board.

   According to the distance, give a warning through **RGB LED** and **Buzzer**.

   i. Distance >= 50cm

      1. **RGB LED** on : Blue

   ii. 50cm >= Distance >= 30cm

      1. **RGB LED** on : Green, LED freq. : 0.5Hz

      2. **Buzzer** on, freq. : 0.5Hz

   iii. 30cm >= Distance >= 15cm

      1. **RGB LED** on : Orange, LED freq. : 1Hz

      2. **Buzzer** on, freq. : 1Hz

   iv. Distance <= 15cm

      1. **RGB LED** on : Red, LED freq. : 2Hz

      2. **Buzzer** on, freq. : 2Hz

2 Function

3. Change **buzzer** key using **potentiometer** and **ADC.**

   i. **ADC** >= 3096

      1. **Buzzer** key : D

   ii. 3096 >= **ADC** >= 2048

      1. **Buzzer** key : F

   iii. 2048 >= **ADC** >= 1024

      1. **Buzzer** key : A

   iv. 1024 < **ADC**

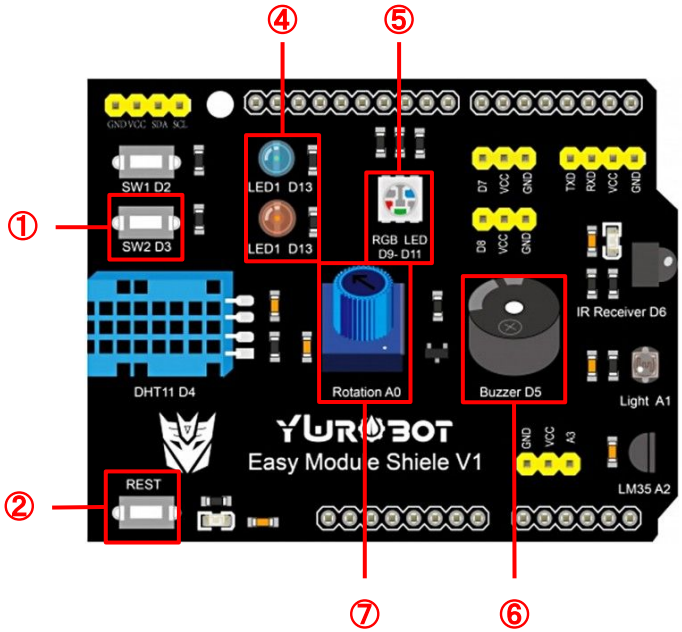
      1. **Buzzer** key : C

4. **Turn on and off** the system by a **button press**.

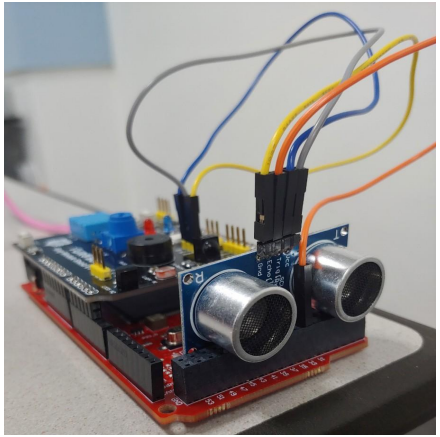
   i. First press SW3 : System on

   ii. Second press SW3 : System off with auditory feedback

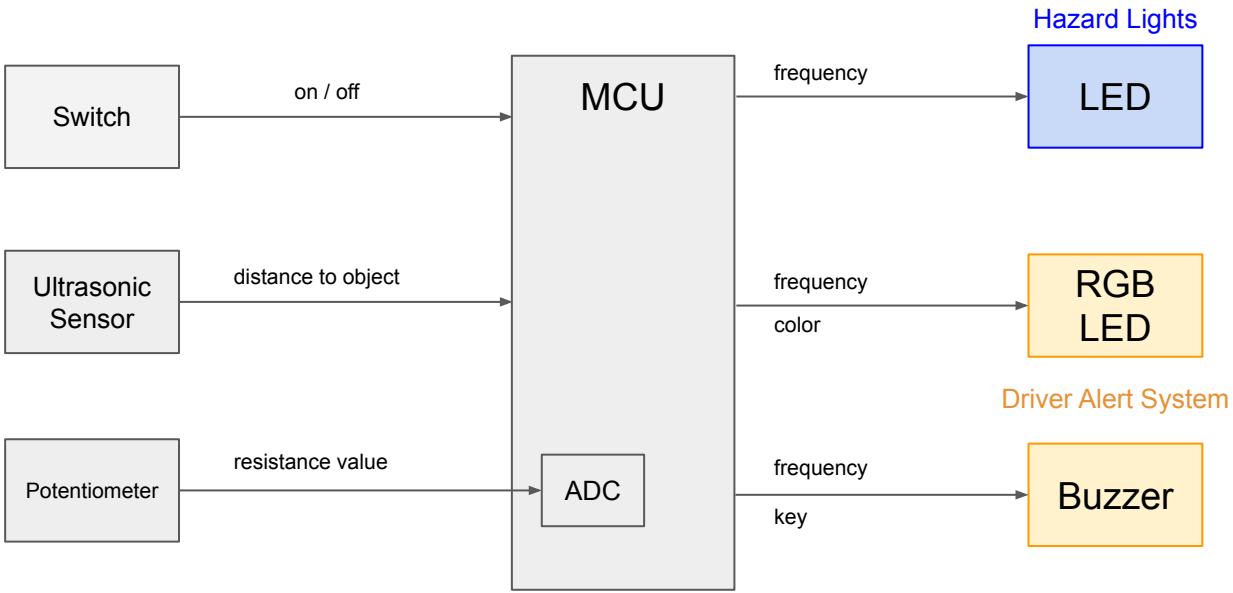
3 System Architecture



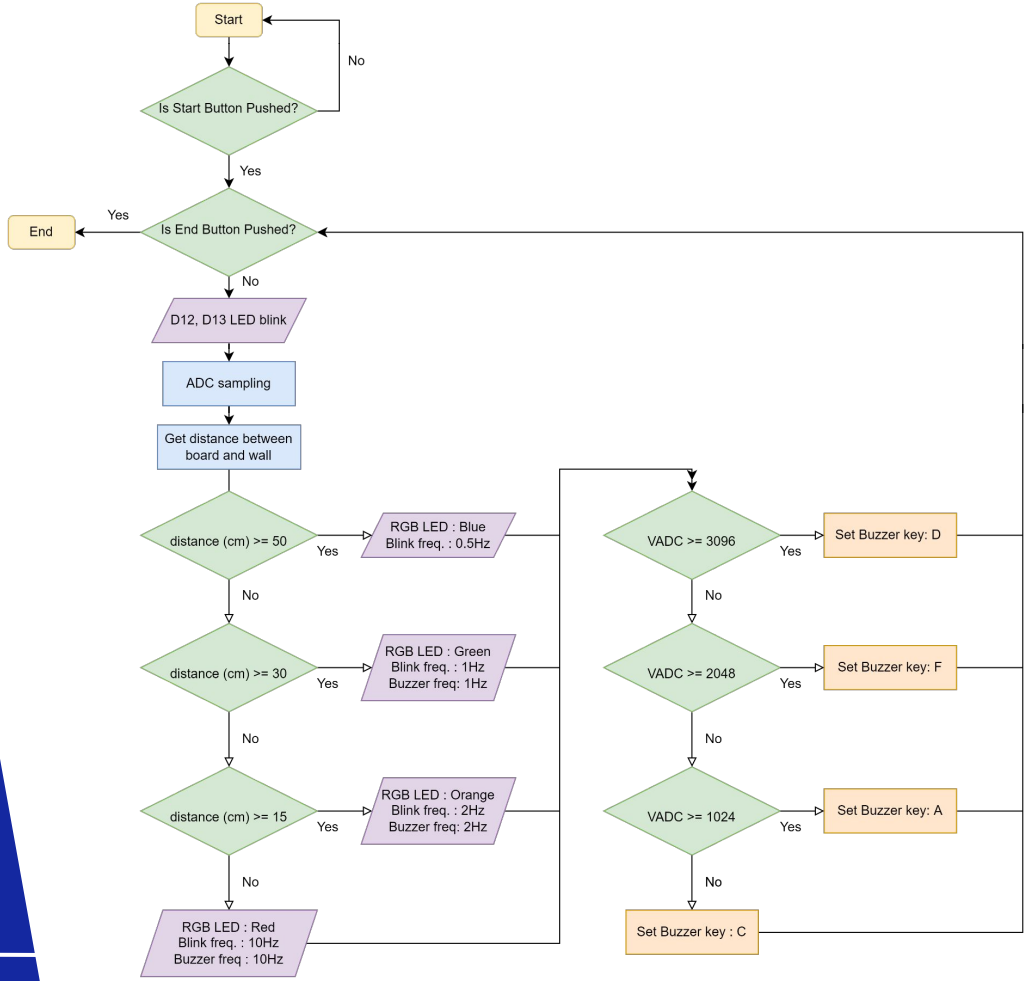
① On / Off switch
② Reset
③ Ultrasonic Sensor
④ Hazard Lights
⑤ Driver Alert RGB LED
⑥ Driver Alert Buzzer
⑦ Buzzer Key Change



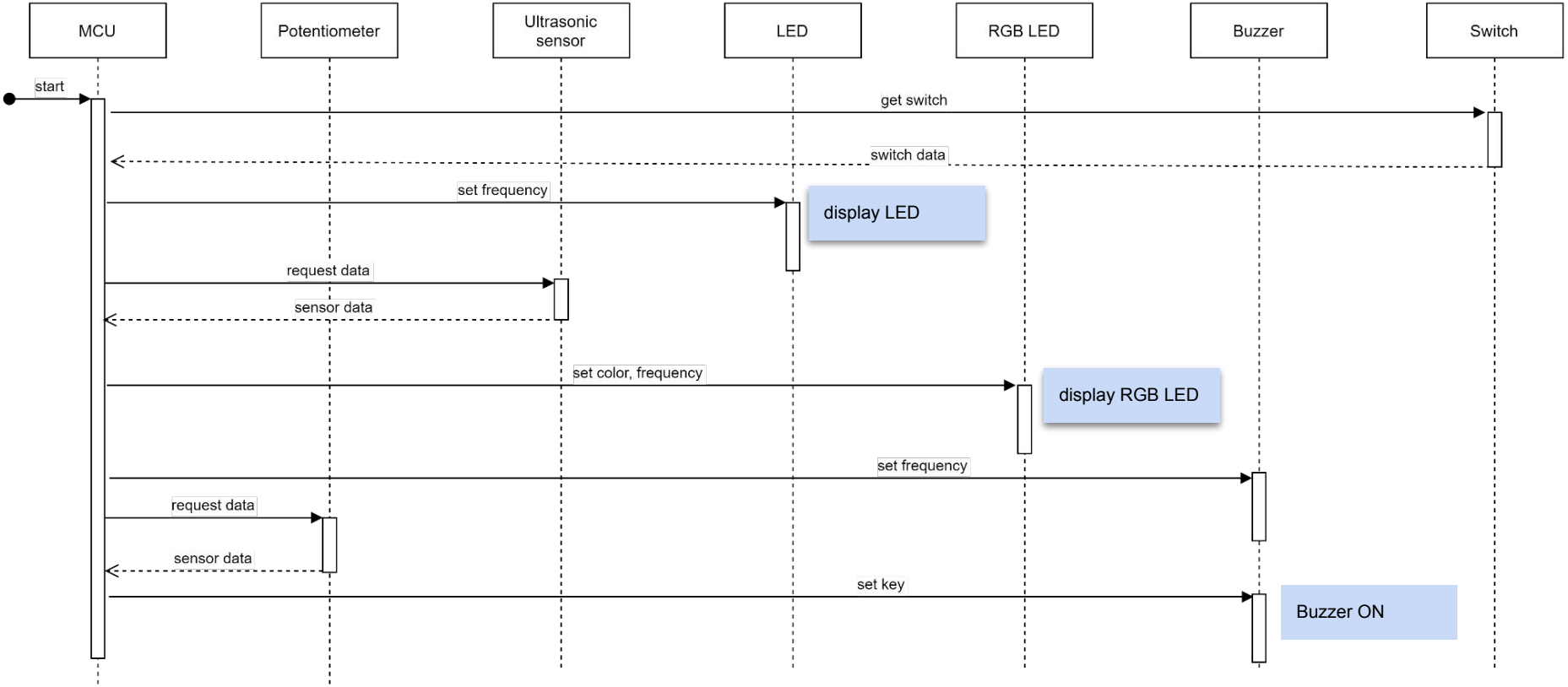
3 System Architecture



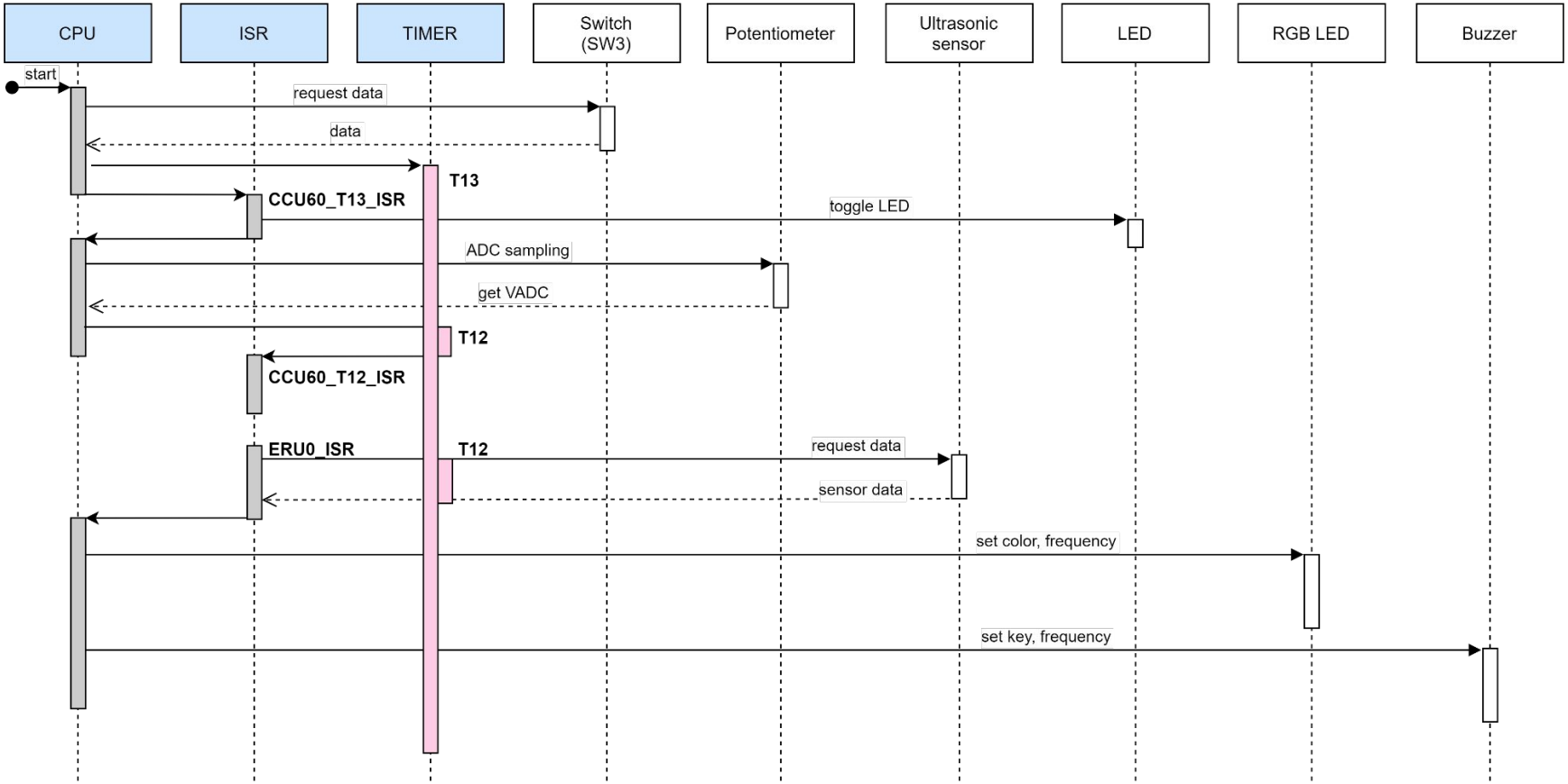
4 Flow Chart


Start
No
Is Start Button Pushed?
Yes
Yes
End
Is End Button Pushed?
No
D12, D13 LED blink
ADC sampling
Get distance between board and wall
distance (cm) >= 50
Yes
RGB LED : Blue
Blink freq. : 0.5Hz
No
distance (cm) >= 30
Yes
RGB LED : Green
Blink freq. : 1Hz
Buzzer freq: 1Hz
No
distance (cm) >= 15
Yes
RGB LED : Orange
Blink freq. : 2Hz
Buzzer freq: 2Hz
No
RGB LED : Red
Blink freq. : 10Hz
Buzzer freq : 10Hz
VADC >= 3096
Yes
Set Buzzer key: D
No
VADC >= 2048
Yes
Set Buzzer key: F
No
VADC >= 1024
Yes
Set Buzzer key: A
No
Set Buzzer key : C

5 Sequence Diagram *(simplified)*



5 Sequence Diagram



Main

5 Code Review - Detect button push & Initialization

```c
int core0_main(void)
{
    IfxCpu_enableInterrupts();

    /* !!WATCHDOG0 AND SAFETY WATCHDOG ARE DISABLED HERE!!
     * Enable the watchdogs and service them periodically if it is required
     */
    IfxScuWdt_disableCpuWatchdog(IfxScuWdt_getCpuWatchdogPassword());
    IfxScuWdt_disableSafetyWatchdog(IfxScuWdt_getSafetyWatchdogPassword());

    /* Wait for CPU sync event */
    IfxCpu_emitEvent(&g_cpuSyncEvent);
    IfxCpu_waitEvent(&g_cpuSyncEvent, 1);

    initButton();
    initRGBLED();
    initLED();

    while(1)    // first while to detect GPIO input
    {
        int pre = P02_IN.U & (0x1 << P1_BIT_LSB_IDX);
        for(uint32 j = 0; j < 100; j++);
        int cur = P02_IN.U & (0x1 << P1_BIT_LSB_IDX);

        if(pre ^ cur)   // sw3 button pushed
        {
            glb_start_flag = 1;
            break;
        }
    }
    for (unsigned int i = 0; i < 20000000; i++);    // delay

    initERU();
    initCCU60();    // T12 : Trigger 펄스 신호 10us 길이 측정, T13 : LED
    initCCU61();    // T12 : Echo 신호 HIGH 길이 측정
    initVADC();
    initGTM();
    initBuzzer();
    initUSonic();
```

Wait until button pushed

Break while loop and start program

<CCU60>

- T12 : timer for measuring 10us trigger pulse in ultrasonic sensor
- T13 : timer LED blink

<CCU61>

- T12 : timer for measuring echo signal in ultrasonic sensor

5 Code Review - Program start

```
while(1)
{
    // Detect program off button pushed
    if((P02_IN.U & (0x1 << P1_BIT_LSB_IDX)) == 0)
    {
        if( glb_start_flag == 1 ) break;
    }


    // get ADC result
    unsigned int adcResult;
    VADC_startConversion();
    adcResult = VADC_readResult();


    // usonic sensor
    usonicTrigger();
    while (range_valid_flag == 0)
    {
        if((P02_IN.U & (0x1 << P1_BIT_LSB_IDX)) == 0)
        {
            if( glb_start_flag == 1 ) break;
        }
    }
```

- Check if button is pushed again
- If button pushed, program off

- If range is not valid (range_valid_flag == 0), infinite loop and can't turn off the program (can't detect button push)

- To solve the problem, keep detecting if the button is pushed

5 Code Review - Ultrasonic sensor

```c
// usonic sensor
usonicTrigger();
while (range_valid_flag == 0)
{
    if((P02_IN.U & (0x1 << P1_BIT_LSB_IDX)) == 0)
    {
        if( glb_start_flag == 1 ) break;
    }
}

int delay_idx = 0;
unsigned int delay[3] = {10000000, 5000000, 1000000};   // delay for RGB LED blink & buzzer

if (range >= 50) // blue
{
    P02_OUT.U &= ~(0x1 << P7_BIT_LSB_IDX);
    P10_OUT.U &= ~(0x1 << P5_BIT_LSB_IDX);
    P10_OUT.U |= (0x1 << P3_BIT_LSB_IDX);

    for (unsigned int i = 0; i < 20000000; i++);

    continue;     // don't blink when distance >= 50cm
}

else if (range >= 30) // green
{

    P02_OUT.U &= ~(0x1 << P7_BIT_LSB_IDX);
    P10_OUT.U |= (0x1 << P5_BIT_LSB_IDX);
    P10_OUT.U &= ~(0x1 << P3_BIT_LSB_IDX);

    delay_idx = 0;
}
```

```c
else if (range >= 15) // orange
{

    P02_OUT.U |= (0x1 << P7_BIT_LSB_IDX);
    P10_OUT.U |= (0x1 << P5_BIT_LSB_IDX);
    P10_OUT.U &= ~(0x1 << P3_BIT_LSB_IDX);

    delay_idx = 1;
}

else // red
{
    P02_OUT.U |= 0x1 << P7_BIT_LSB_IDX;
    P10_OUT.U &= ~(0x1 << P5_BIT_LSB_IDX);
    P10_OUT.U &= ~(0x1 << P3_BIT_LSB_I  DX);

    delay_idx = 2;
}

for (unsigned int i = 0; i < delay[delay_idx]; i++);    // RGB LED delay

// rgb led off
P02_OUT.U &= ~(0x1 << P7_BIT_LSB_IDX);
P10_OUT.U &= ~(0x1 << P5_BIT_LSB_IDX);
P10_OUT.U &= ~(0x1 << P3_BIT_LSB_IDX);
```

RGB LED color & blinking frequency changes depending on the distance

5 Code Review - Buzzer

```
// get ADC result
unsigned int adcResult;
VADC_startConversion();
adcResult = VADC_readResult();
```

get ADC result

```
// run buzzer
for (unsigned int i = 0; i < 1000000; i++);

int idx_duty = 2  * (3 - adcResult / 1024) + 1;          // set buzzer delay idx
if (idx_duty < 1) idx_duty = 0;

GTM_TOM0_CH11_SR0.B.SR0 = 6250000 / duty[idx_duty];      // set buzzer key
GTM_TOM0_CH11_SR1.B.SR1 = 3125000 / duty[idx_duty];

for (unsigned int i = 0; i < delay[delay_idx]; i++);     // buzzer delay

// buzzer off
GTM_TOM0_CH11_SR0.B.SR0 = 0;
GTM_TOM0_CH11_SR1.B.SR1 = 0;
```

set buzzer key using ADC result

buzzer delay differs by distance

5 Code Review - System off

```
void systemoff(void)
{
    // LED off
    P10_OUT.U &= ~(0x1 << P2_BIT_LSB_IDX);  // toggle P10.1 (LED D12 RED)
    P10_OUT.U &= ~(0x1 << P1_BIT_LSB_IDX);  // toggle P10.2 (LED D13 BLUE)

    // RGBLED off
    P02_OUT.U &= ~(0x1 << P7_BIT_LSB_IDX);
    P10_OUT.U &= ~(0x1 << P5_BIT_LSB_IDX);
    P10_OUT.U &= ~(0x1 << P3_BIT_LSB_IDX);

    // Buzzer - program off sound
    // from 3 octave C ~ 4 octave C {C, D, E, F, G, A, B, C}
    unsigned int duty[8] = { 130, 146, 164, 174, 195, 220, 246, 262 };

    GTM_TOM0_CH11_SR0.B.SR0 = 6250000 / duty[7];
    GTM_TOM0_CH11_SR1.B.SR1 = 3125000 / duty[7];

    for(unsigned int i = 0; i < 20000000; i++);

    GTM_TOM0_CH11_SR0.B.SR0 = 6250000 / duty[0];
    GTM_TOM0_CH11_SR1.B.SR1 = 3125000 / duty[0];

    for(unsigned int i = 0; i < 20000000; i++);

    // buzzer off
    GTM_TOM0_CH11_SR0.B.SR0 = 0;
    GTM_TOM0_CH11_SR1.B.SR1 = 0;

}
```

When end button pushed
-> system off

- LED off
- RGB LED off
- Buzzer sound play
- Buzzer off

ISR

5 Code Review - ISR for ultrasonic sensor

```c
__interrupt(0x0A) __vector_table(0)
void ERU0_ISR(void)
{
    if( (P00_IN.U & (0x1 << P4_BIT_LSB_IDX)) != 0 )     // rising edge of echo
    {
        //                          ___________
        //      echo ________|
        //                          ^
        CCU61_TCTR4.U = 0x1 << T12RS_BIT_LSB_IDX;       // start CCU61 T12 counter
    }
    else                                                // falling edge of echo
    {
        //          ________
        //      echo          |___________
        //                    ^
        CCU61_TCTR4.B.T12RR = 0x1;      // stop CCU61 T12 counter

        // (1 / t_freq) * counter * 1000000 / 58 = centimeter
        range = ((CCU61_T12.B.T12CV * 1000000) / 48828) / 58;
        range_valid_flag = 1;

        CCU61_TCTR4.B.T12RES = 0x1;     // reset CCU61 T12 counter
    }
}
```

ISR for echo

- ERU interrupt occurs when GPIO input LOW -> HIGH
- CCU61 T12 counter : get ultrasonic round trip time
- calculate distance using time

```c
void usonicTrigger(void)
{
    // start of 10us Trigger Pulse
    // GPIO P02.0 --> HIGH
    P02_OUT.U |= 0x1 << P6_BIT_LSB_IDX;
    range_valid_flag = 0;
    CCU60_TCTR4.U = 0x1 << T12RS_BIT_LSB_IDX;       // T12 start counting

}
```

```c
__interrupt(0x0B) __vector_table(0)
void CCU60_T12_ISR(void)
{
    // end of 10us Trig
    // GPIO P02.6 --> LOW
    P02_OUT.U &= ~(0x1 << P6_BIT_LSB_IDX);
}
```

ISR for trigger pulse

- After 10us, period match interrupt occurs
- Set GPIO OUT to low

5 Code Review - ISR for D12, D13 LED

```
__interrupt(0x0C) __vector_table(0)
void CCU60_T13_ISR(void)
{
    P10_OUT.U ^= 0x1 << P2_BIT_LSB_IDX;  // toggle P10.1 (LED D12 RED)
    P10_OUT.U ^= 0x1 << P1_BIT_LSB_IDX;  // toggle P10.2 (LED D13 BLUE)
}
```

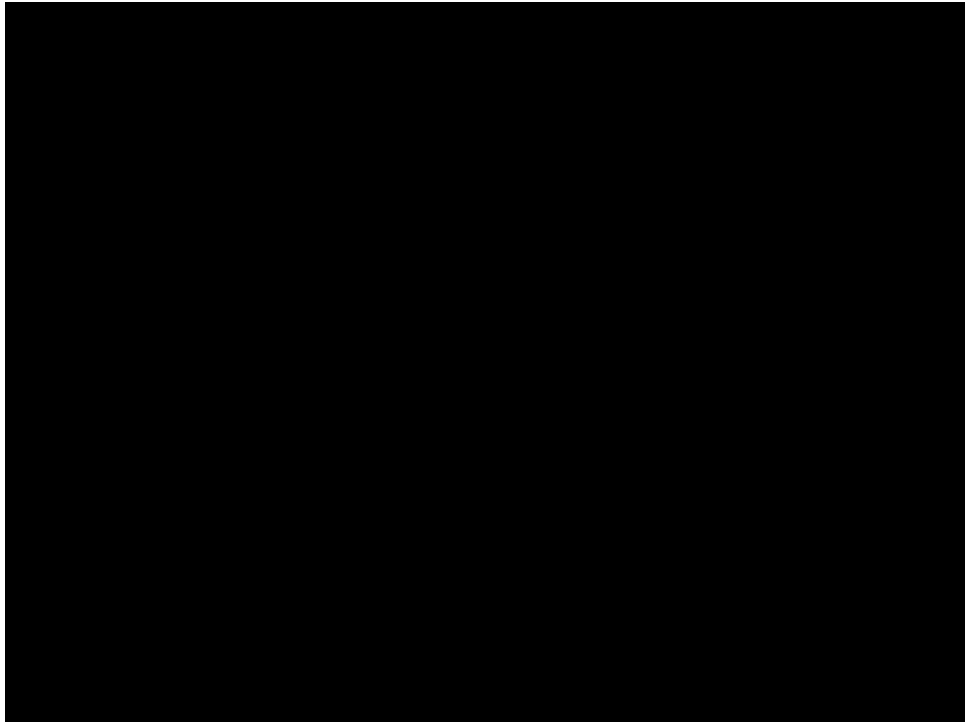
- CCU60 T13 : timer for D12, D13 LED
- LED toggles in a certain frequency through the whole program

T13 configurations in initCCU60()

```
// T13 configurations for led blink
CCU60_TCTR0.B.T13CLK = 0x2;     // f_CCU6 = 50 MHz, prescaler = 1024
CCU60_TCTR0.B.T13PRE = 0x1;     // prescaler enable
CCU60_T13PR.B.T13PV = 36621 - 1;
CCU60_TCTR4.B.T13STR = 0x1;
CCU60_T13.B.T13CV = 0x0;
CCU60_IEN.B.ENT13PM = 0x1;
CCU60_INP.B.INPT13 = 0x1;       // SR1
SRC_CCU6_CCU60_SR1.B.SRPN = 0x0C;
SRC_CCU6_CCU60_SR1.B.TOS = 0x0;
SRC_CCU6_CCU60_SR1.B.SRE = 0x1;
CCU60_TCTR4.B.T13RS = 0x1;
```

5 Demo

임베디드_sw개발_팀프로젝트_1조 - YouTube



5 Roles

	Design		Coding		Simulation		Operation Test		Documentation	
Kang	Architecture	10%	Buzzer	10%	Buzzer	25%	Buzzer	40%	Video editing	40%
Kim	Sensor	40%	Ultra sonic	20%	Ultra sonic	25%	Ultra sonic	30%	Final PT	10%
Ryu	processing unit	30%	LED	40%	LED	25%	LED	10%	presentation material	20%
Jang	H/W	20%	ADC	30%	ADC	25%	ADC	20%	presentation material	30%

5 Milestone

Office Move: By Hour								
	2	4	6	8	10	12	14	16
Kang	Floor plan & Instructions	Set frequency w/ timer		Buzzer control & Testing		Documentation & Video editing		
Kim	Floor plan & Instructions	Data collection from ultrasonic sensor & Testing			Merging & Simulating		Documentation & Video	Final
Ryu	Floor plan & Instructions	Set frequency w/ timer		Set LED color w/ PWM	LED & RGB Testing		Documentation & Video	
Jang	Floor plan & Instructions	ADC sampling from potentiometer			Set buzzer key & Testing		Documentation & Video	

THANK YOU