TI DSP, MCU 및 Xilinx Zynq FPGA
프로그래밍 전문가 과정

강사 – Innova Lee(이상훈)

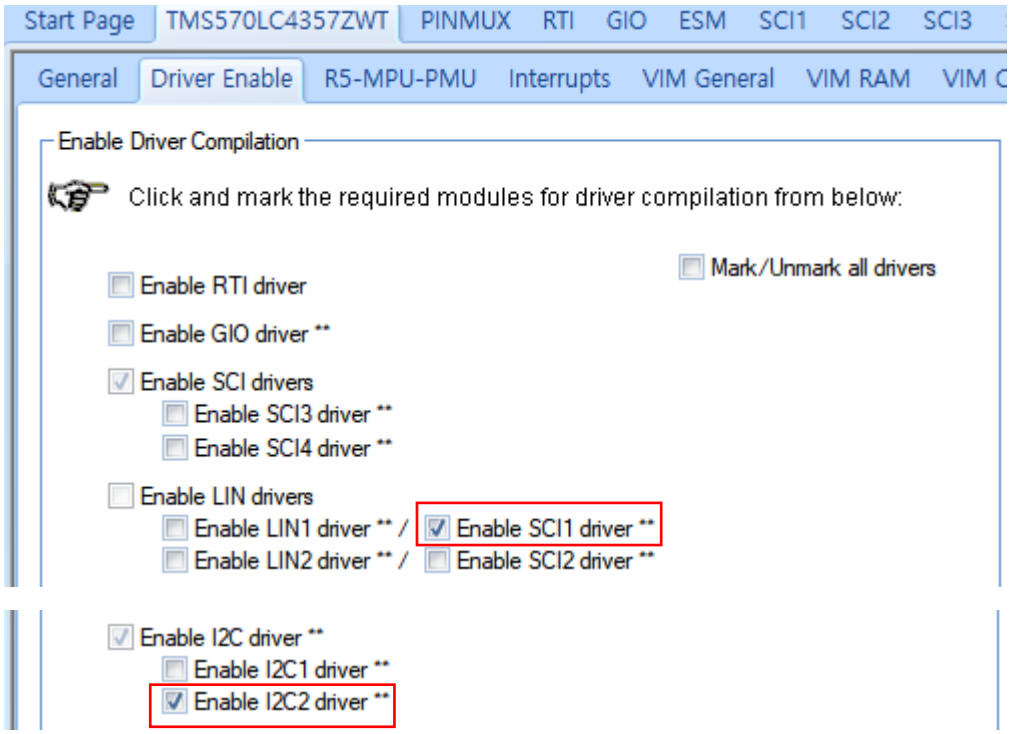
gcccompil3r@gmail.com

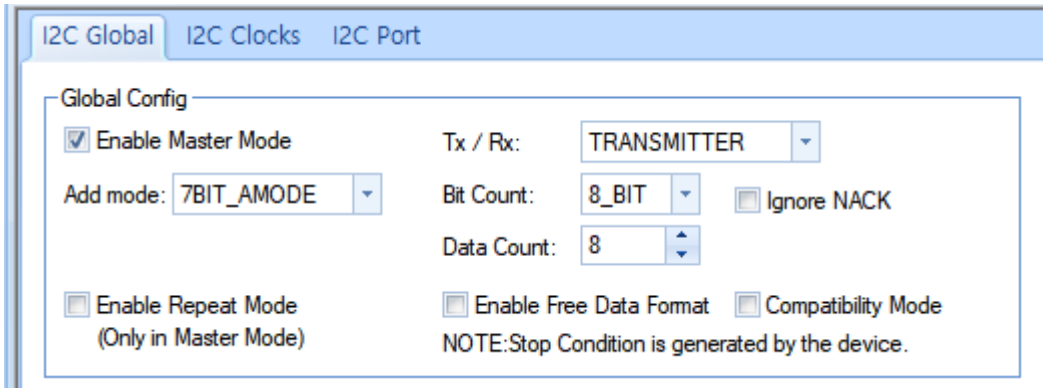
학생 – 문한나

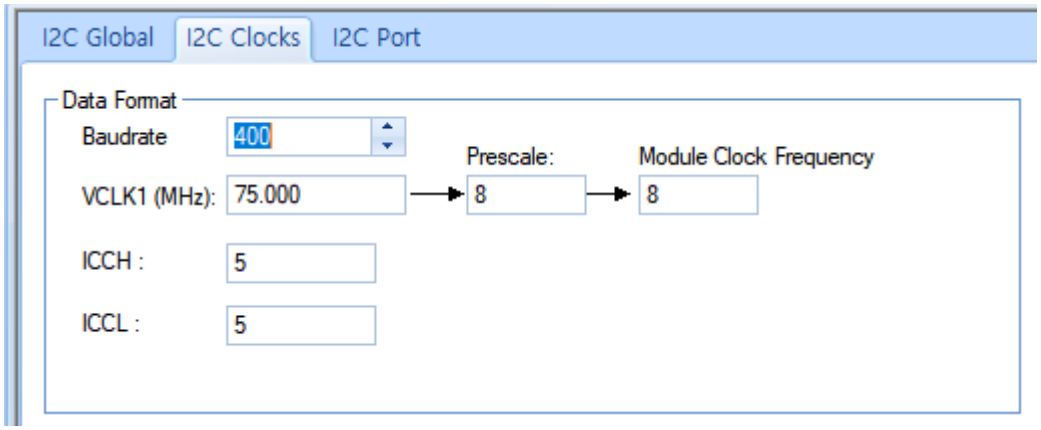
mhn97@naver.com

I2C_Lidar

HCG 설정






I2C Global
I2C Clocks
I2C Port
Data Format
Baudrate
400
Prescale:
Module Clock Frequency
VCLK1 (MHz):
75.000
8
8
ICCH :
5
ICCL :
5

CCS 코드

```
#include <HL_hal_stdtypes.h>
#include <HL_i2c.h>
#include <HL_reg_sci.h>
#include <HL_sci.h>
#include <HL_sys_core.h>
#include <stdbool.h>
#include <stdio.h>
#include <string.h>

#define UART sciREG1
#define LIDAR_SLAVE_ADDR 0x62

#define ACQ_COMMAND 0x00
#define STATUS 0x01
#define SIG_COUNT_VAL 0x02
#define ACQ_CONFIG_REG 0x04
#define THRESHOLD_BYPASS 0x1C
#define READ_FROM 0x8f

#define FULL_DELAY_HIGH 0x0f

uint8 receives[2];
volatile int g_acc_flag;

uint8 bias_cnt = 0;

void sciDisplayText(sciBASE_t *sci, uint8 *text, uint32 len);
void pwmSet(void);
void wait(uint32 delay);
void Lidar_without_bias(void);
void Lidar_bias(void);
void Get_Data(void);

void Lidar_enable(void);

void wait(uint32 delay)
{
    int i;

    for (i = 0; i < delay; i++)
        ;
}

void sciDisplayText(sciBASE_t *sci, uint8 *text, uint32 len)
{
    while (len--)
    {
        while ((UART->FLR & 0x4) == 4)
            ;
        sciSendByte(UART, *text++);
    }
}

void disp_set(char *str)
{
    char txt_buf[256] = { 0 };
    unsigned int buf_len;
    sprintf(txt_buf, str);
    buf_len = strlen(txt_buf);
    sciDisplayText(sciREG1, (uint8 *) txt_buf, buf_len);
    wait(100000);
}
```

```c
int main(void)
{
    char txt_buf[256] = { 0 };
    unsigned int buf_len;
    volatile int i ;
    int cnt = 1;

    uint16 ave[2] = { 0 };

    sciInit();

    disp_set("SCI Configuration Success!!\n\r\0");

    i2cInit();
    wait(10000000);

    disp_set("I2C Init Success!!\n\r\0");

    Lidar_enable();

    disp_set("Lidar Enable Success!!\n\r\0");

    wait(1000000);

    for (;;)
    {

        Get_Data();

        if (g_acc_flag) //1이 되었을 때 동작
        {
            uint16 tmp;
            tmp = receives[0] << 8; //8비트 두개를 합쳐서 2바이트로 저장
            tmp |= receives[1];

                if (cnt % 3 == 0)
                {
                    tmp = (ave[0] + ave[1]) / 2; //FIR필터

                    sprintf(txt_buf, "Distance = %d\n\r\0", tmp);
                    buf_len = strlen(txt_buf);
                    sciDisplayText(sciREG1, (uint8 *) txt_buf, buf_len);

                    i = 0;
                    cnt++;

                    g_acc_flag = 0;

                }
                else
                {

                    ave[i] = tmp; //필터를 위해 값을 저장

                    i++;
                    cnt++;

                    g_acc_flag = 0;

                }

            }
```

```
        Lidar_without_bias(); //99번은 ACQ_COMMAND에 0x03 세팅
        bias_cnt++;
        if(bias_cnt == 100){

            Lidar_bias(); //100번째에 ACQ_COMMAND에 0x04 세팅

            bias_cnt = 0;
        }
    }
    return 0;
}

void Lidar_enable(void)
{
    uint8 tmp[4] = { 0x80, 0x08, 0x00, 0x04 };

    volatile unsigned int cnt = 7;
    i2cSetSlaveAdd(i2cREG2, LIDAR_SLAVE_ADDR); //LIDAR_SLAVE_ADDR 0x62
    i2cSetDirection(i2cREG2, I2C_TRANSMITTER); //전송모드
    /* 마스터 모드에서만 사용가능 전송값*/
    i2cSetCount(i2cREG2, cnt + 1);
    i2cSetMode(i2cREG2, I2C_MASTER); //마스터모드
    i2cSetStop(i2cREG2); //안정성을 위해 스탑부터
    i2cSetStart(i2cREG2); //시작
    i2cSendByte(i2cREG2, SIG_COUNT_VAL);
    i2cSend(i2cREG2, 1, &tmp[0]); //0x80

    i2cSendByte(i2cREG2, ACQ_CONFIG_REG);
    i2cSend(i2cREG2, 1, &tmp[1]); //0x08

    i2cSendByte(i2cREG2, THRESHOLD_BYPASS);
    i2cSend(i2cREG2, 1, &tmp[2]); //0x00

    i2cSendByte(i2cREG2, ACQ_COMMAND);
    i2cSend(i2cREG2, 1, &tmp[3]); //0x04

    disp_set("Lidar tmp 1 Enable Success!!\n\r\0");

    while (i2cIsBusBusy(i2cREG2) == true)
        ; //데이터가 오고 있는지
    while (i2cIsStopDetected(i2cREG2) == 0)
        ; //데이터가 다 왔는지
    i2cClearSCD(i2cREG2); //초기화

    wait(100000);

}

void Get_Data()
{

    i2cSetSlaveAdd(i2cREG2, LIDAR_SLAVE_ADDR); //LIDAR_SLAVE_ADDR 0x62
    i2cSetDirection(i2cREG2, I2C_TRANSMITTER); //전송
    i2cSetCount(i2cREG2, 1);
    i2cSetMode(i2cREG2, I2C_MASTER); //마스터모드
    i2cSetStop(i2cREG2); //스탑부터(안정성)
    i2cSetStart(i2cREG2); //시작
    i2cSendByte(i2cREG2, READ_FROM); //0x8f
```

```
    while (i2cIsBusBusy(i2cREG2) == true)
        //데이터가 오고 있는지
        ;
    while (i2cIsStopDetected(i2cREG2) == 0)
        //데이터가 다 왔는지
        ;
    i2cClearSCD(i2cREG2); //초기화

    i2cSetDirection(i2cREG2, I2C_RECEIVER); //데이터 받음
    i2cSetCount(i2cREG2, 2);
    i2cSetMode(i2cREG2, I2C_MASTER); //마스터모드
    i2cSetStart(i2cREG2); //시작
    i2cReceive(i2cREG2, 2, (unsigned char *) receives);
    i2cSetStop(i2cREG2); //끝

    while (i2cIsBusBusy(i2cREG2) == true)
//데이터가 오고 있는지
        ;
    while (i2cIsStopDetected(i2cREG2) == 0)
//데이터가 다 왔는지
        ;
    i2cClearSCD(i2cREG2); //초기화

    g_acc_flag = 1; //다시 값을 받기 위하여
}

void Lidar_bias(void)
{
    volatile unsigned int cnt = 1;
    unsigned char data[1] = { 0x04U };
    i2cSetSlaveAdd(i2cREG2, LIDAR_SLAVE_ADDR);
    i2cSetDirection(i2cREG2, I2C_TRANSMITTER);
    i2cSetCount(i2cREG2, cnt + 1);
    i2cSetMode(i2cREG2, I2C_MASTER);
    i2cSetStop(i2cREG2);
    i2cSetStart(i2cREG2);
    i2cSendByte(i2cREG2, ACQ_COMMAND);
    i2cSend(i2cREG2, cnt, data);
    i2cSetStop(i2cREG2);

    while (i2cIsBusBusy(i2cREG2) == true)
        ;
    while (i2cIsStopDetected(i2cREG2) == 0)
        ;
    i2cClearSCD(i2cREG2);
    wait(1000000);
}

void Lidar_without_bias(void)
{
    volatile unsigned int cnt = 1;
    unsigned char data[1] = { 0x03U };
    i2cSetSlaveAdd(i2cREG2, LIDAR_SLAVE_ADDR);
    i2cSetDirection(i2cREG2, I2C_TRANSMITTER);
    i2cSetCount(i2cREG2, cnt + 1);
    i2cSetMode(i2cREG2, I2C_MASTER);
    i2cSetStop(i2cREG2);
    i2cSetStart(i2cREG2);
```

```
    i2cSendByte(i2cREG2, ACQ_COMMAND);
    i2cSend(i2cREG2, cnt, data);
    i2cSetStop(i2cREG2);

    while (i2cIsBusBusy(i2cREG2) == true)
        ;
    while (i2cIsStopDetected(i2cREG2) == 0)
        ;
    i2cClearSCD(i2cREG2);
    wait(1000000);
}
```

추가설명

Lidar 기본정보

I2C Protocol Information

This device has a 2-wire, I2C-compatible serial interface (refer to I2C-Bus Specification, Version 2.1, January 2000, available from Philips Semiconductor). It can be connected to an I2C bus as a slave device, under the control of an I2C master device. It supports standard 400 kHz data transfer mode. Support is not provided for 10-bit addressing.

The Sensor module has a 7-bit slave address with a default value of 0x62 in hexadecimal notation. The effective 8 bit I2C address is: 0xC4 write, 0xC5 read. The device will not presently respond to a general call.

LIDAR_SLAVE_ADDR : 0x62 (7bit mode)

Data Baudrate : 400kHz

데이터를 얻는 방법

The simplest method of obtaining measurement results from the I2C interface is as follows:

1. Write 0x04 to register 0x00.
2. Read register 0x01. Repeat until bit 0 (LSB) goes low.
3. Read two bytes from 0x8f (High byte 0x0f then low byte 0x10) to obtain the 16-bit measured distance in centimeters.

1. 0x00레지스터인 ACQ_COMMAND에 0x04를 세팅한다.
2. 0x8f레지스터에서 데이터를 얻어온다. (2바이트)

주의사항

Receiver Bias Correction

Address	Name	Description	Initial Value
0x00	ACQ_COMMAND	Device command	--

- Write 0x00: Reset device, all registers return to default values
- Write 0x03: Take distance measurement without receiver bias correction
- Write 0x04: Take distance measurement with receiver bias correction

Faster distance measurements can be performed by omitting the receiver bias correction routine. Measurement accuracy and sensitivity are adversely affected if conditions change (e.g. target distance, device temperature, and optical noise). To achieve good performance at high measurement rates receiver bias correction must be performed periodically. The recommended method is to do so at the beginning of every 100 sequential measurement commands.

99번까지는 ACQ_COMMAND에 0x03 세팅하고, 100번째에 ACQ_COMMAND에 0x04를 세팅하는 것을 권장