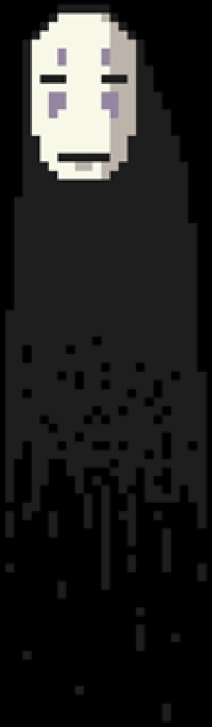
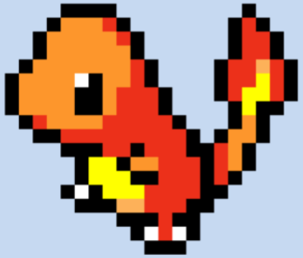
Xilinx
Zynq FPGA
TI DSP MCU 기반의
프로그래밍 및 회로 설계 전문가

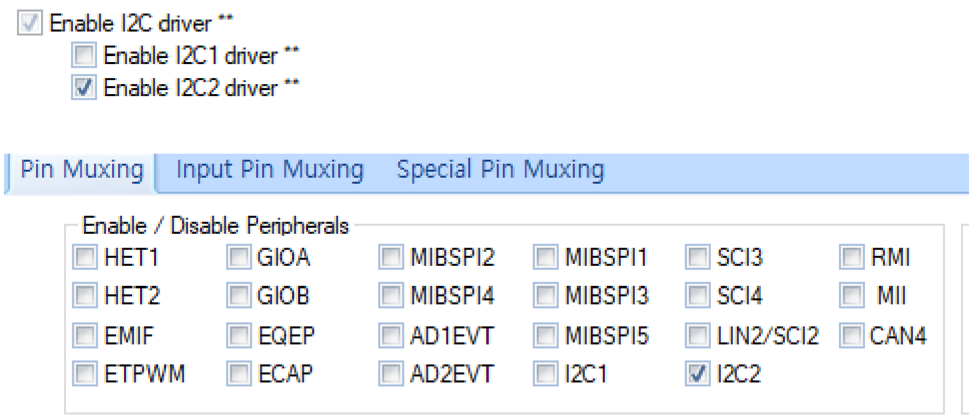


강사 이상훈
gcccompil3r@gmail.com

학생 김민호
minking12@naver.com



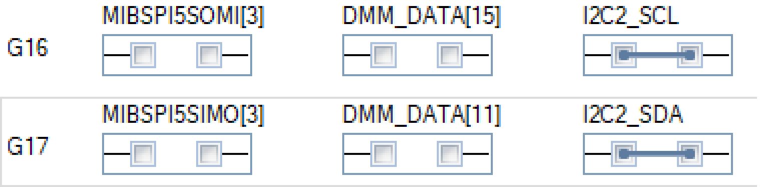
I2C_LCD basic setting



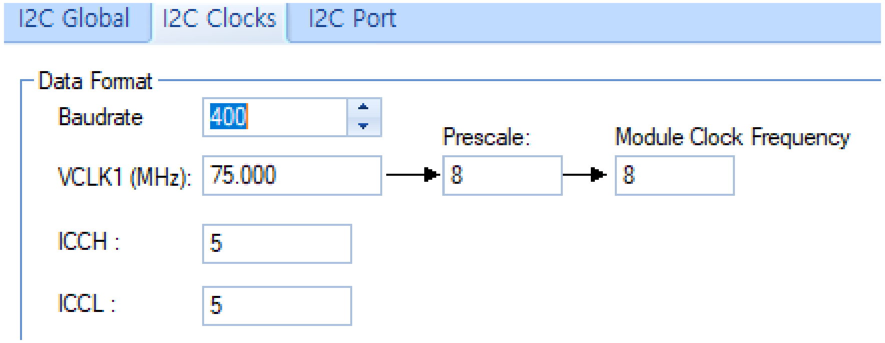
☑ Enable I2C driver **
- □ Enable I2C1 driver **
- ☑ Enable I2C2 driver **

Enable / Disable Peripherals

□ HET1	□ GIOA	□ MIBSPI2	□ MIBSPI1	□ SCI3	□ RMI
□ HET2	□ GIOB	□ MIBSPI4	□ MIBSPI3	□ SCI4	□ MII
□ EMIF	□ EQEP	□ AD1EVT	□ MIBSPI5	□ LIN2/SCI2	□ CAN4
□ ETPWM	□ ECAP	□ AD2EVT	□ I2C1	☑ I2C2	



G16	MIBSPI5SOMI[3]	DMM_DATA[15]	I2C2_SCL ●
G17	MIBSPI5SIMO[3]	DMM_DATA[11]	I2C2_SDA ●



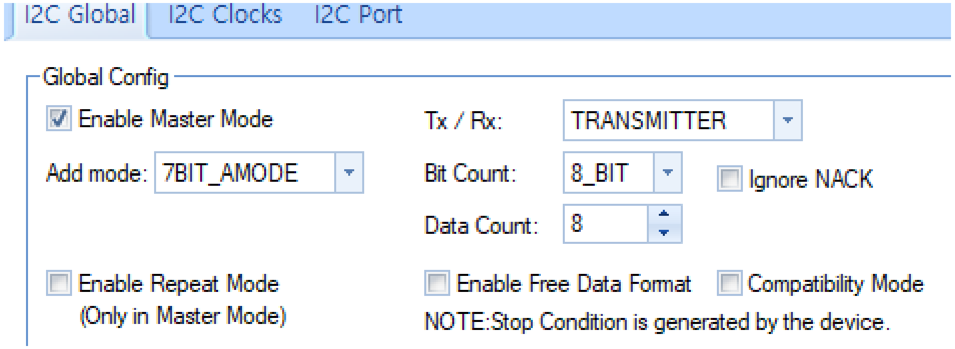
Data Format

Baudrate: 400

VCLK1 (MHz): 75.000 → Prescale: 8 → Module Clock Frequency: 8

ICCH : 5

ICCL : 5



Global Config

☑ Enable Master Mode

Add mode: 7BIT_AMODE

Tx / Rx: TRANSMITTER

Bit Count: 8_BIT

□ Ignore NACK

Data Count: 8

□ Enable Repeat Mode (Only in Master Mode)

□ Enable Free Data Format

□ Compatibility Mode

NOTE:Stop Condition is generated by the device.

I2C_LCD basic code

```
#include <HL_i2c.h>
#include <HL_reg_i2c.h>
#include <stdbool.h>
#include <stdint.h>
#include <stdlib.h>
#define LCD_ADDRESS 0x27
void lcd_Backlight();
void lcd_noBacklight();
void lcd_sned_string(char *str);
void lcd_init(void);
void lcd_send_cmd(char cmd);
void lcd_send_data(char data);
void lcd_set_cursor(int row, char col);
void lcd_clear();
unsigned char LCD_BACKLIGHT = 0X0;
void lcd_send_string(char *str)
{
    while (*str)
        lcd_send_data(*str++);
}
int main(void)
{
    volatile int i;
    for (i = 0; i < 10000000; i++)
        ;
    i2cInit();
    for (i = 0; i < 10000000; i++)
        ;
    lcd_init();
    lcd_Backlight();
 while (1)
    {
        int abc = 1000;
        char str[10];
lcd_set_cursor(0, 0);
        lcd_send_string("Velocity : ");
lcd_set_cursor(1, 0);
        sprintf(str, "%d", abc);
        lcd_send_string(str);
        for (i = 0; i < 80000000; i++)
            ;
        lcd_clear();
    }
}
```

```
void lcd_send_cmd(char cmd)
{
    volatile unsigned int cnt = 4;
    unsigned char data_u, data_l;
    uint8_t data_t[4];
    data_u = (cmd & 0xf0);
    data_l = ((cmd << 4) & 0xf0);
    data_t[0] = data_u | (0x04 + LCD_BACKLIGHT);  //en=1, rs=0
    data_t[1] = data_u | (0x00 + LCD_BACKLIGHT);  //en=0, rs=0
    data_t[2] = data_l | (0x04 + LCD_BACKLIGHT);  //en=1, rs=0
    data_t[3] = data_l | (0x00 + LCD_BACKLIGHT);  //en=0, rs=0
    i2cSetSlaveAdd(i2cREG2, LCD_ADDRESS);
    i2cSetDirection(i2cREG2, I2C_TRANSMITTER);
    i2cSetCount(i2cREG2, cnt + 1);
    i2cSetMode(i2cREG2, I2C_MASTER);
    i2cSetStop(i2cREG2);
    i2cSetStart(i2cREG2);
    i2cSendByte(i2cREG2, LCD_ADDRESS);
    i2cSend(i2cREG2, cnt, data_t);
    while (i2cIsBusBusy(i2cREG2) == true)
        ;
    while (i2cIsStopDetected(i2cREG2) == 0)
        ;
    i2cClearSCD(i2cREG2);
    for (cnt = 0; cnt < 1000000; cnt++)
        ;
}
```

I2C_LCD basic code

```
void lcd_send_data(char data)
{
    volatile unsigned int cnt = 4;
    char data_u, data_l;
    uint8_t data_t[4];

    data_u = (data & 0xf0);
    data_l = ((data << 4) & 0xf0);
    data_t[0] = data_u | (0x05 + LCD_BACKLIGHT);  //en=1, rs=0
    data_t[1] = data_u | (0x01 + LCD_BACKLIGHT);  //en=0, rs=0
    data_t[2] = data_l | (0x05 + LCD_BACKLIGHT);  //en=1, rs=0
    data_t[3] = data_l | (0x01 + LCD_BACKLIGHT);  //en=0, rs=0

    i2cSetSlaveAdd(i2cREG2, LCD_ADDRESS);
    i2cSetDirection(i2cREG2, I2C_TRANSMITTER);
    i2cSetCount(i2cREG2, cnt + 1);
    i2cSetMode(i2cREG2, I2C_MASTER);
    i2cSetStop(i2cREG2);
    i2cSetStart(i2cREG2);
    i2cSendByte(i2cREG2, LCD_ADDRESS);
    i2cSend(i2cREG2, cnt, data_t);

    while (i2cIsBusBusy(i2cREG2) == true)
        ;
    while (i2cIsStopDetected(i2cREG2) == 0)
        ;
    i2cClearSCD(i2cREG2);

    for (cnt = 0; cnt < 1000000; cnt++)
        ;
}
```

```
void lcd_init(void)
{
    lcd_send_cmd(0x02);
    lcd_send_cmd(0x28);
    lcd_send_cmd(0x0c);
    lcd_send_cmd(0x80);
}

void lcd_set_cursor(int row, char col)
{
    if (row == 0)
        lcd_send_cmd(0x80 + col);
    else if (row == 1)
        lcd_send_cmd(0xc0 + col);

}

void lcd_Backlight()
{

    LCD_BACKLIGHT = 0x08;
}

void lcd_noBacklight()
{
    LCD_BACKLIGHT = 0x00;

}

void lcd_clear()
{
    lcd_send_cmd(0x01);
}
```

I2C_LCD basic