

TI DSP, MCU 및 Xilinx Zynq FPGA

프로그래밍 전문가 과정

2018-05-14 (53회차)

강사 - Innova Lee(이상훈)

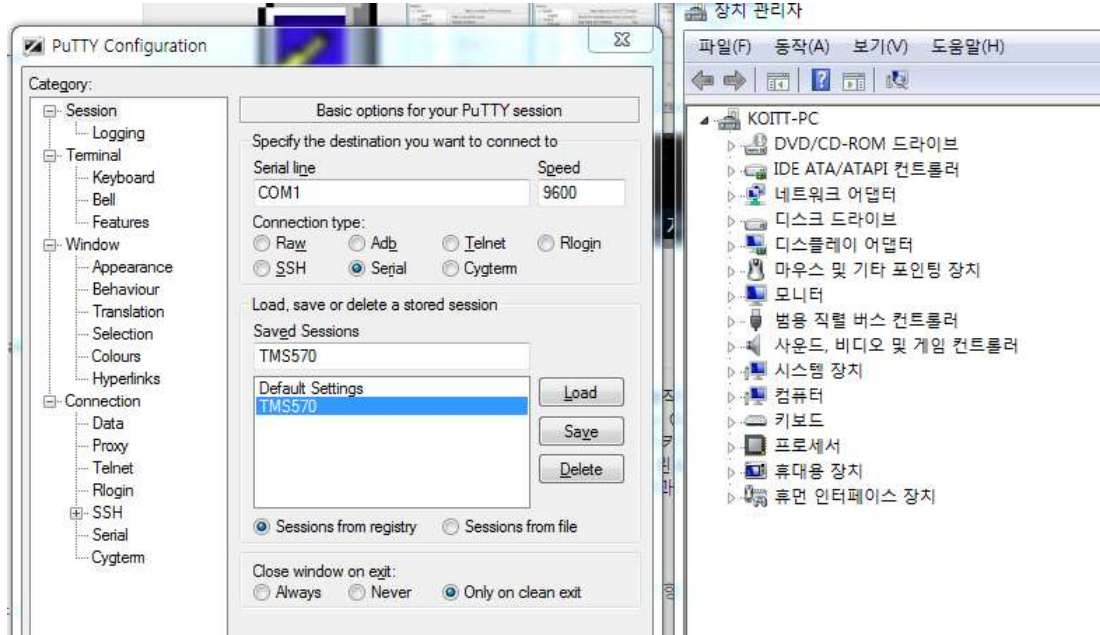
gcccompil3r@gmail.com

학생 - 정유경

ucong@naver.com

1. UART PWM

PuTTY를 설정하고, 장치관리자에서 MCU가 COM몇번에 연결되어 있는지 확인한다



DRIVER 설정: GIO, SCI, ETPWM

PINMUX 설정: eTPWMA1

B5	GIOA[5]	NONE	NONE	EXTCLKIN	NONE	eTPWM1A
B6	MIBSPI5NCS[1]	DMM_DATA[06]	NONE	NONE	NONE	NONE

ETPWM 모듈 활성화: ETPWM1설정

PINMUX	RTI	GIO	ESM	SCI1	SCI2	SCI3	SCI4	LIN1	LIN2	MIBSPI1	MIBSPI2	MIBSPI3	MIBSPI4	MIBSPI5	SPI1	SPI2
General ETPWM1 ETPWM2 ETPWM3 ETPWM4 ETPWM5 ETPWM6 ETPWM7																
Enable ETPWM modules <input checked="" type="checkbox"/> Enable ETPWM1																

Clock Configuration 설정

PINMUX	RTI	GIO	ESM	SCI1	SCI2	SCI3	SCI4	LIN1	LIN2	MIBSPI1	MIBSPI2	MIBSPI3	MIBSPI4	MIBSPI5	SPI1	SPI2
General ETPWM1 ETPWM2 ETPWM3 ETPWM4 ETPWM5 ETPWM6 ETPWM7																
Clock Configuration TB Clock (MHz): 10.000 VCLK3 (MHz): 10.000 HSPCLKDIV: 10 CLKDIV: 0 Actual TB Clock (MHz): 1.000																
PWM Configuration <div> <div> PWM High Polarity Low Polarity </div> <div> Duty(%): 50 Period(ns): 20000000 </div> <div> iDuty: 10000000.000 iPeriod: 20000000.000 </div> </div> <div> <div> PWM High Polarity Low Polarity </div> <div> Duty(%): 50 Period(ns): 20000000 </div> <div> iDuty: 10000000.000 iPeriod: 20000000.000 </div> </div> <div> Delay(ns): 1000.000 Rising Edge Delay Invert Polarity Delay(ns): 1000.000 Falling Edge Delay Invert Polarity Enable delay Disable delay ETPWMxA ETPWMxB </div>																

코드작성

```
#include "HL_sys_common.h"
#include "HL_system.h"
#include "HL_sci.h"
#include "HL_sys_common.h"

#define TSIZE1      6
uint8 TEXT1[TSIZE1] = {'H','E', 'L', 'L','O',' '};
#define TSIZE2      12
uint8 TEXT2[TSIZE2] = {'T','I',' ','H','E','R','C','U','L','E','S',' '};
#define TSIZE3      12
uint8 TEXT3[TSIZE3] = {'S','A','F','E','T','Y',' ','M','C','U','\n','\r'};

void sciDisplayText(sciBASE_t *sci, uint8 *text, uint32 length);
void wait(uint32 time);

#define UART      sciREG1

uint8  emacAddress[6U] =      {0xFFU, 0xFFU, 0xFFU, 0xFFU, 0xFFU, 0xFFU};
uint32 emacPhyAddress =      1U;

int main(void)
{
    sciInit();

    for(;;)
    {
        sciDisplayText(UART, &TEXT1[0], TSIZE1);
        sciDisplayText(UART, &TEXT2[0], TSIZE2);
        sciDisplayText(UART, &TEXT3[0], TSIZE3);
        wait(200);
    }

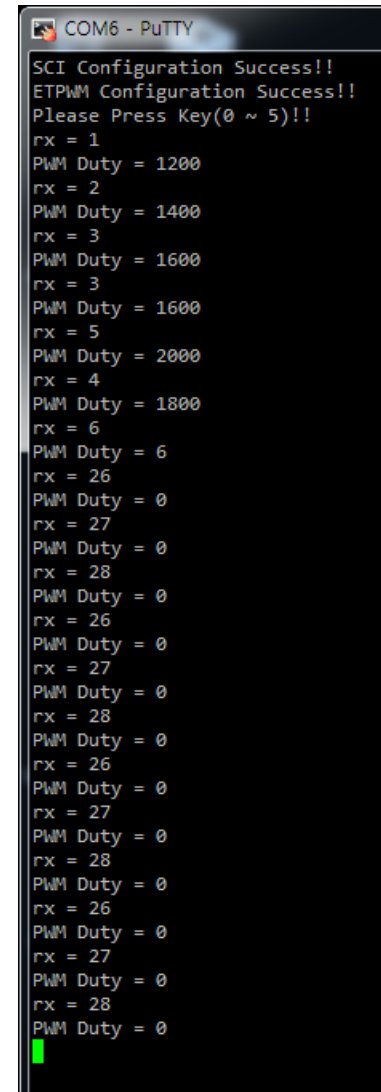
    return 0;
}
```

```

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

void wait(uint32 time)
{
    int i;
    for(i=0; i<time;i++)
        ;
}

```



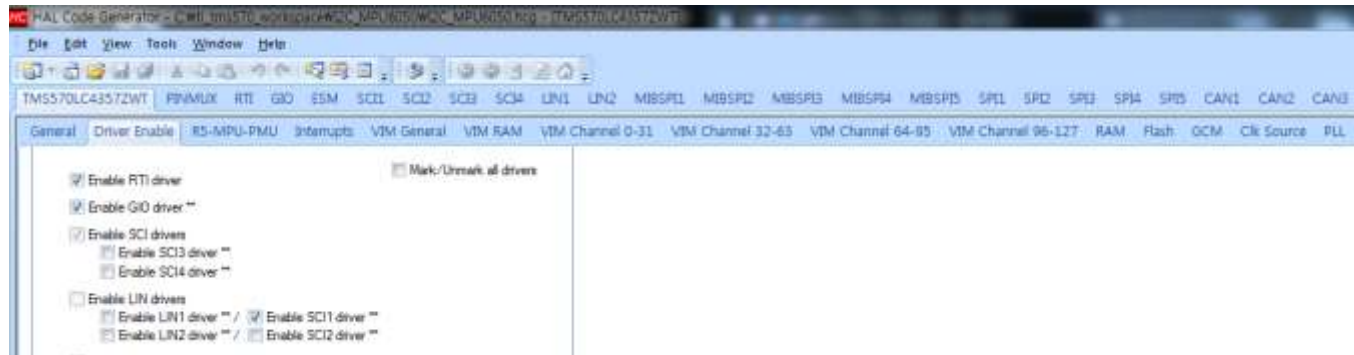
```

COM6 - PuTTY
SCI Configuration Success!!
ETPWM Configuration Success!!
Please Press Key(0 ~ 5)!!
rx = 1
PWM Duty = 1200
rx = 2
PWM Duty = 1400
rx = 3
PWM Duty = 1600
rx = 3
PWM Duty = 1600
rx = 5
PWM Duty = 2000
rx = 4
PWM Duty = 1800
rx = 6
PWM Duty = 6
rx = 26
PWM Duty = 0
rx = 27
PWM Duty = 0
rx = 28
PWM Duty = 0
rx = 26
PWM Duty = 0
rx = 27
PWM Duty = 0
rx = 28
PWM Duty = 0
rx = 26
PWM Duty = 0
rx = 27
PWM Duty = 0
rx = 28
PWM Duty = 0
rx = 26
PWM Duty = 0
rx = 27
PWM Duty = 0
rx = 28
PWM Duty = 0

```

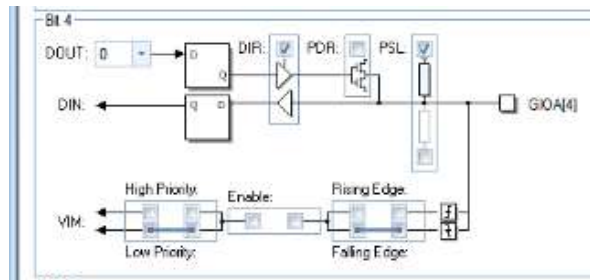
2. I2C_MPU6050

Driver Enable 설정: RTI, GIO, SCI1(LIN Driver), I2C2, ETPWM

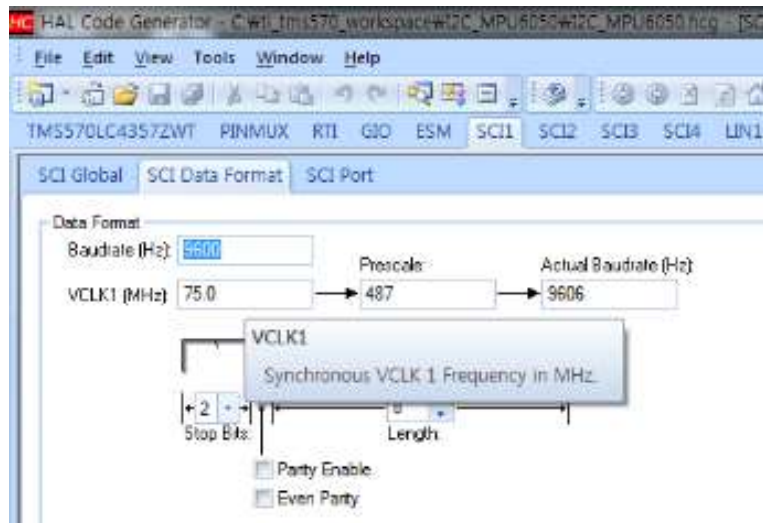


PINMUX 설정: eTPWM1A, I2C2_SCL, I2C2_SDA

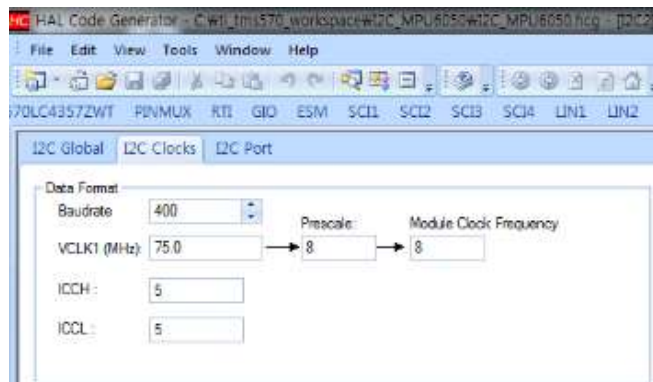
GIO > PORTA > Bit4 핀 설정



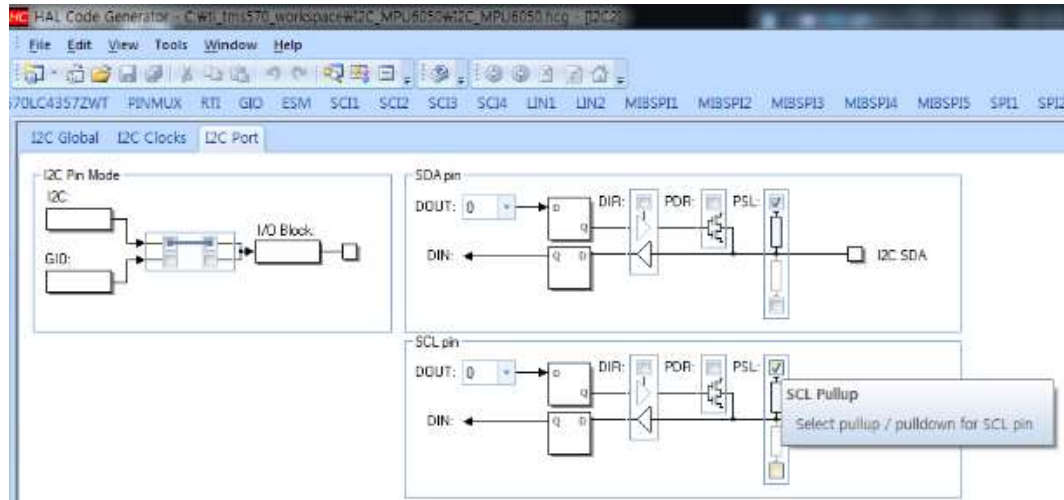
Baud Rate 설정



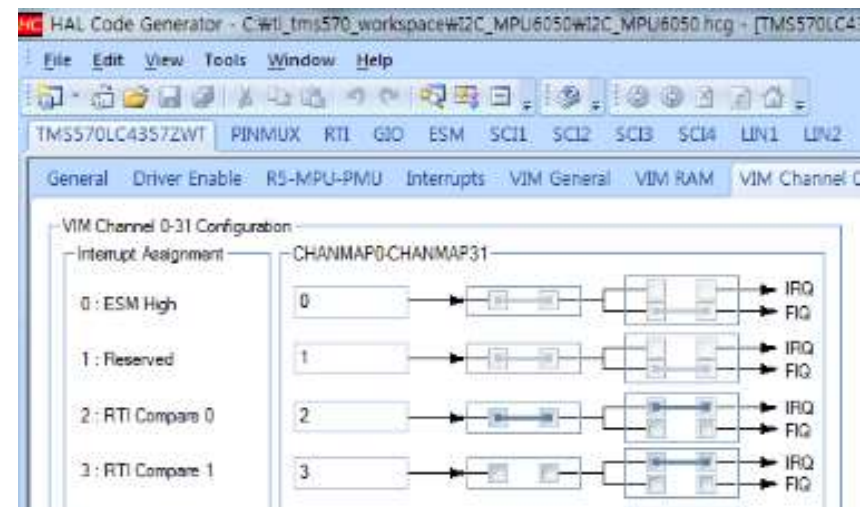
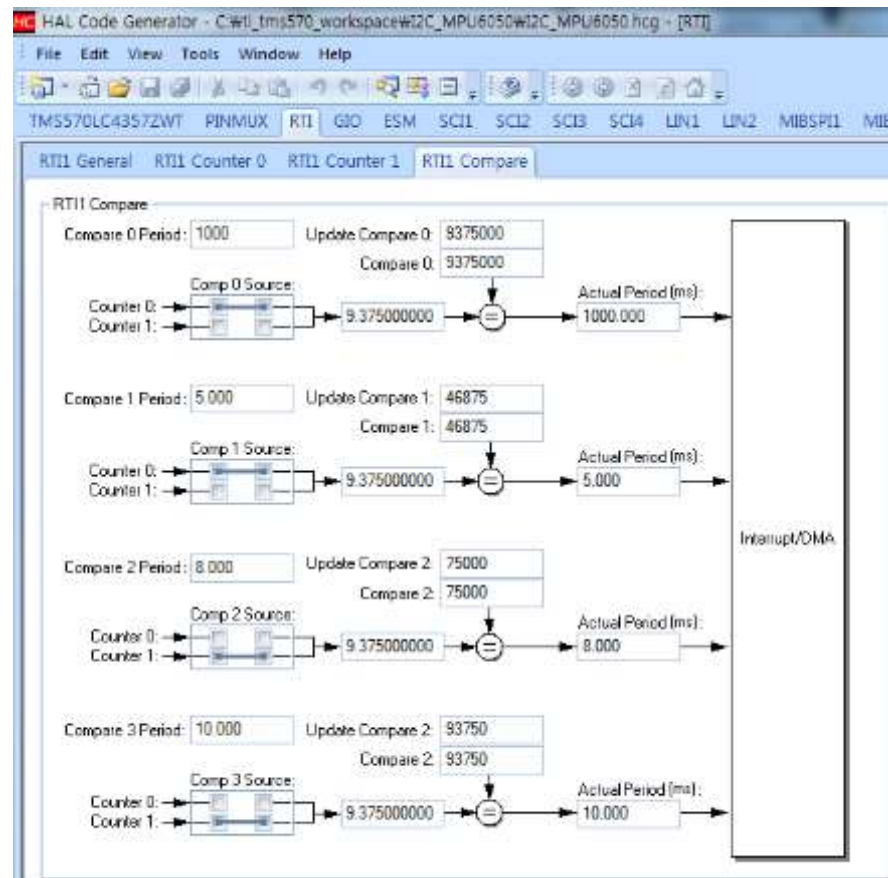
I2C Clocks 설정



SCL핀 풀업 설정(오픈드레인이므로 풀업해주지 않으면 플로팅 상태가 된다)



RTI 설정



코드작성

```
#include "HL_sys_common.h"
#include "HL_system.h"
#include "HL_etpwm.h"
#include "HL_sci.h"
#include "HL_gio.h"
#include "HL_i2c.h"
#include "HL_rti.h"

#include <string.h>
#include <stdio.h>

#define UART          sciREG1
#define MPU6050_ADDR  0x68

void sciDisplayText(sciBASE_t *sci, uint8 *text, uint32 len);
void pwmSet(void);
void wait(uint32 delay);
void MPU6050_enable(void);
void MPU6050_acc_config(void);
void disp_set(char *);

uint32 rx_data = 0;
uint32 tmp = 0;
uint32 value = 0;

volatile char g_acc_xyz[6];
volatile int g_acc_flag;

#define IDX      6
uint32 duty_arr[IDX] = {1000, 1200, 1400, 1600, 1800, 2000};

int main(void)
{
    char txt_buf[64] = {0};
    unsigned int buf_len;
```

```

volatile int i;
signed short acc_x, acc_y, acc_z;
double real_acc_x, real_acc_y, real_acc_z;

sciInit();

disp_set("SCI Configuration Success!!\n\r\0");
gioInit();
disp_set("GIO Init Success!!\n\r\0");
i2cInit();
wait(10000000);

MPU6050_enable();
disp_set("MPU6050 Enable Success!!\n\r\0");

MPU6050_acc_config();
disp_set("MPU6050 Accelerometer Configure Success!!\n\r\0");

rtiInit();
rtiEnableNotification(rtiREG1, rtiNOTIFICATION_COMPARE0);

_enable_IRQ_interrupt();
rtiStartCounter(rtiREG1, rtiCOUNTER_BLOCK0);
disp_set("RTI Init Success!!\n\r\0");

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

etpwmStartTBCLK();
wait(10000000);

for(;;)
{
    if(g_acc_flag)
    {
        acc_x = acc_y = acc_z = 0;
        real_acc_x = real_acc_y = real_acc_z = 0.0;
    }
}

```

```

    acc_x = g_acc_xyz[0];
    acc_x = acc_x << 8;
    acc_x |= g_acc_xyz[1];
    real_acc_x = ((double)acc_x) / 2048.0;

    acc_y = g_acc_xyz[2];
    acc_y = acc_y << 8;
    acc_y |= g_acc_xyz[3];
    real_acc_y = ((double)acc_y) / 2048.0;

    acc_z = g_acc_xyz[4];
    acc_z = acc_z << 8;
    acc_z |= g_acc_xyz[5];
    real_acc_z = ((double)acc_z) / 2048.0;

    sprintf(txt_buf, "acc_x = %2.5lf\tacc_y = %2.5lf\tacc_z = %2.5lf\n\r\0", real_acc_x, real_acc_y, real_acc_z);

    buf_len = strlen(txt_buf);
    sciDisplayText(sciREG1, (uint8 *)txt_buf, buf_len);
    g_acc_flag = 0;
}

}

```

```

#if 0
for(;;)
{
    tmp = sciReceiveByte(UART);
    rx_data = tmp - 48;

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

    pwmSet();

    sprintf(txt_buf, "PWM Duty = %d\n\r\0", value);
}

```

```

        buf_len = strlen(txt_buf);
        sciDisplayText(sciREG1, (uint8 *)txt_buf, buf_len);
    }
#endif
    return 0;
}

void pwmSet(void)
{
    value = duty_arr[rx_data];
    etpwmSetCmpA(etpwmREG1, value);
    wait(10000);
}

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 MPU6050_enable(void)
{
    volatile unsigned int cnt = 2;
    unsigned char data[2] = {0x00U, 0x00U};
    unsigned char slave_word_address = 0x6bU;

    i2cSetSlaveAdd(i2cREG2, MPU6050_ADDR);

```

```

i2cSetDirection(i2cREG2, I2C_TRANSMITTER);
i2cSetCount(i2cREG2, cnt + 1);
i2cSetMode(i2cREG2, I2C_MASTER);
i2cSetStop(i2cREG2);
i2cSetStart(i2cREG2);
i2cSendByte(i2cREG2, slave_word_address);
i2cSend(i2cREG2, cnt, data);

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

wait(1000000);
}

```

```

void MPU6050_acc_config(void)
{
    volatile unsigned int cnt = 1;
    unsigned char data[1] = {0x18U};
    unsigned char slave_word_address = 0x1cU;

    i2cSetSlaveAdd(i2cREG2, MPU6050_ADDR);
    i2cSetDirection(i2cREG2, I2C_TRANSMITTER);
    i2cSetCount(i2cREG2, cnt + 1);
    i2cSetMode(i2cREG2, I2C_MASTER);
    i2cSetStop(i2cREG2);
    i2cSetStart(i2cREG2);
    i2cSendByte(i2cREG2, slave_word_address);
    i2cSend(i2cREG2, cnt, data);

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

    i2cClearSCD(i2cREG2);
}

```

```

    wait(1000000);
}

void rtiNotification(rtiBASE_t *rtiREG, uint32 notification)
{
    unsigned char slave_word_address = 0x3B;

    i2cSetSlaveAdd(i2cREG2, MPU6050_ADDR);
    i2cSetDirection(i2cREG2, I2C_TRANSMITTER);
    i2cSetCount(i2cREG2, 1);
    i2cSetMode(i2cREG2, I2C_MASTER);
    i2cSetStop(i2cREG2);
    i2cSetStart(i2cREG2);
    i2cSendByte(i2cREG2, slave_word_address);

    while(i2cIsBusBusy(i2cREG2) == true)
        ;
    while(i2cIsStopDetected(i2cREG2) == 0)
        ;
    i2cClearSCD(i2cREG2);
    i2cSetDirection(i2cREG2, I2C_RECEIVER);
    i2cSetCount(i2cREG2, 6);
    i2cSetMode(i2cREG2, I2C_MASTER);
    i2cSetStart(i2cREG2);

    i2cReceive(i2cREG2, 6, (unsigned char *)g_acc_xyz);
    i2cSetStop(i2cREG2);

    while(i2cIsBusBusy(i2cREG2) == true)
        ;
    while(i2cIsStopDetected(i2cREG2) == 0)
        ;
    i2cClearSCD(i2cREG2);
    g_acc_flag = 1;
}

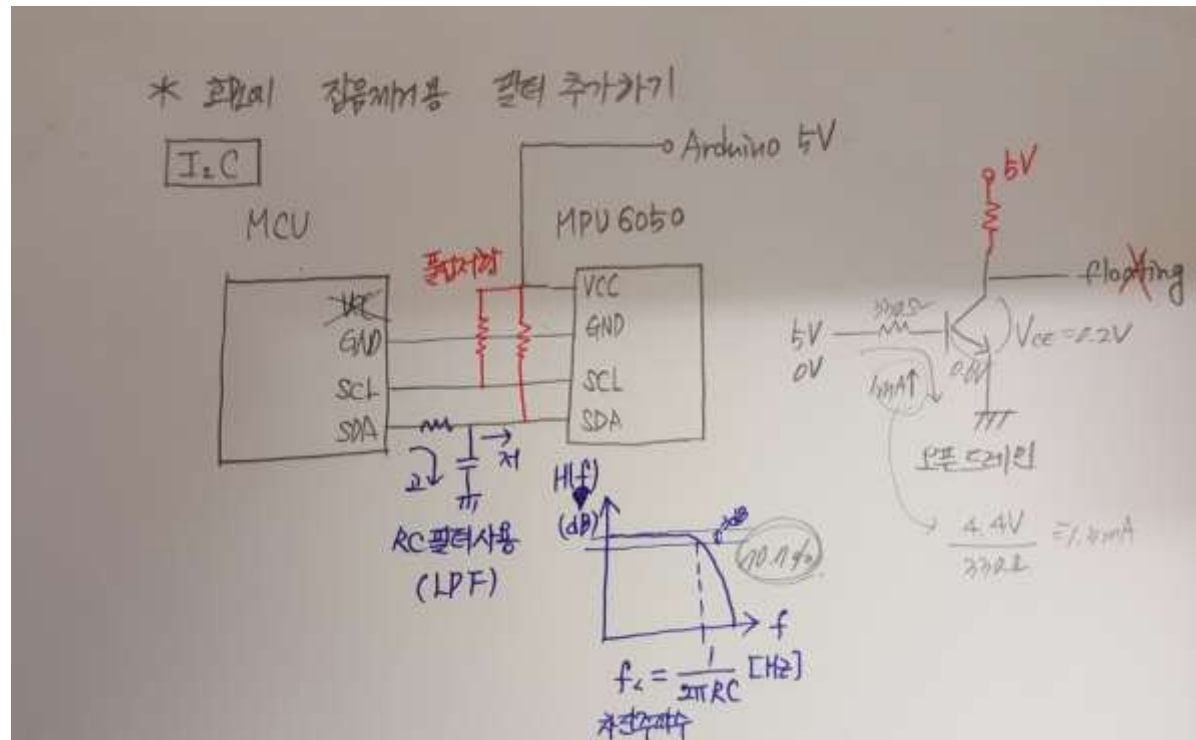
void disp_set(char *str)
{

```

```

char txt_buf[64] = {0};
unsigned int buf_len;
sprintf(txt_buf, str);
buf_len = strlen(txt_buf);
sciDisplayText(sciREG1, (uint8 *)txt_buf, buf_len);
wait(100000);
}

```



시리얼 통신 프로토콜

- I2C(Inter-Integrated Circuit)
- SPI(Serial Peripheral Interface)
- UART(Universal Asynchronous Receive-Transmit)

동기통신, 비동기통신

송신자와 수신자는 서로 동기를 맞추어야 한다. 수신자가 송신자와 상호 동기를 맞추기 위한 한가지 방법은 동기를 위한 클럭 신호를 추가로 사용하는 것이다. 이런 용도의 클럭 신호 사용 여부에 따라서 직렬 통신을 크게 동기 (synchronous) 통신과 비동기 (asynchronous) 통신으로 구분할 수 있다.

동기통신

동기 통신은 데이터 전송을 위한 선 외에 송신자와 수신자의 동기를 위한 별도의 클럭 신호를 사용하는 통신 방법이다. 동기 통신에서 송신자와 수신자는 서로 공유하는 클럭 신호에 맞추어 데이터를 주고 받는다.

SPI와 I2C 프로토콜은 임베디드 시스템에서 가장 많이 사용되고 있는 동기 통신 규격이다
주로 **MCU-IC** 간의 통신에 사용된다.

비동기통신

비동기 통신은 동기를 맞추기 위한 별도의 클럭 신호 없이 데이터를 주고 받는 방법이다. 클럭 신호가 없기 때문에 송신자는 한 바이트의 데이터를 전송하기 직전에 통신의 시작을 알리는 시작 비트 (start bit)를 먼저 보내 데이터의 전송이 시작된다는 것을 수신자에게 알린다.

시작 비트를 보낸 다음부터 데이터를 한 비트씩 보내고 마지막에는 통신의 끝을 알리는 정지비트 (stop bit)를 보내 통신이 끝났음을 알린다. 시작 비트는 논리 '0'이며 정지 비트는 논리 '1'이다.

(정지 비트가 논리 '1'이므로 데이터 전송이 없는 상태에서 통신 선로는 논리 '1'을 유지하고 있다)

이 경우에는 서로 정확한 통신 속도를 약속하고 있어야 한다. 통신의 시작을 뜻하는 시작 비트를 받으면 수신자는 이 때 부터 미리 약속한 통신 속도에 맞추어 데이터를 한 비트씩 읽어 들이기 때문이다.

통신속도

통신 속도는 보통 Baud라는 단위로 표현하는데 Baud Rate는 초당 얼마나 많은 심볼(Symbol, 의미 있는 데이터 묶음)을 전송할 수 있는가를 나타내는 말이다. 만약 하나의 심볼이 2비트로 구성되어 있는 경우라면 보는 BPS의 반이 된다. (즉, 1보 동안 2비트가 전송된다. 신호를 1비트로 여긴다면 BPS와 같은 단위가 된다) 심볼이란 말은 의미 있는 데이터 비트의 묶음이다. 일반적으로 시리얼 통신에서는 데이터 비트가 8-bit를 사용하므로 이를 하나의 심볼이라고 이야기할 수 있다. 1개의 symbol (또는 character)는 8개의 bit 정보와 같다.

예를 들어, 2400 baud rate를 갖는다는 말은 1초에 2400개의 의미 있는 심볼 즉 캐릭터를 송신할 수 있다는 이야기이다. 1초에 2400개의 ASCII 코드를 전송할 수 있다는 말로 이해할 수 있다.

UART는 비동기 통신 프로토콜의 대표적인 예다.

주로 **MCU-PC**, MCU-MCU 간 통신에 사용된다.

시리얼 통신 참고 사이트

<http://egloos.zum.com/circle5940/v/45333>

<https://eslectures.blog.me/80137860924>

LPF 참고 사이트

<https://m.blog.naver.com/PostView.nhn?blogId=specialist0&logNo=220831227390&proxyReferer=https%3A%2F%2Fwww.google.co.kr%2F>