TI DSP, MCU 및 Xilinx Zynq FPGA 프로그래밍 전문가 과정

- MPU6050 칼만필터 내장형 (UART BASE) -

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – GJ (박현우)
uc820@naver.com

목차

MPU6050 + STM32 (칼만필터 내장형)

1. DATASHEET

Precision six-axis inertial navigation module specification

1 product description

This six-axis module uses advanced digital filtering technology (Kalman filter), can effectively reduce the measurement noise, improve the measurement accuracy.

The module integrates motion engine DMP, and obtain quaternion to get the current attitude. Attitude measurement accuracy of 0.01 degrees, high stability, performance and even better than some of the professional inclinometer!

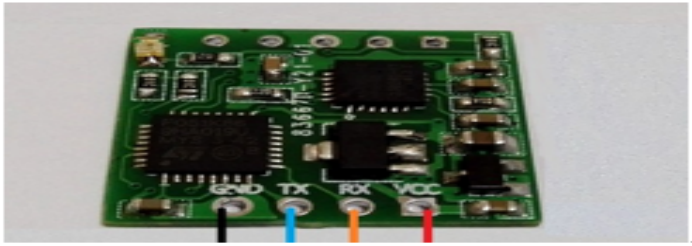
This module uses high-precision six-axis gyro accelerometer MPU6050, read the measurement data MPU6050 by a processor and output via the serial port, eliminating the need for users to develop their own MPU6050 complex I2C protocol, and careful PCB layout and processes ensure MPU6050 Receive external interference minimum, the highest precision measurement.

2 Performance parameters

1. Voltage: 3V ~ 6V.
2. Current: <10mA.
3. Volume: 17.8mm X 17.8mm Weight: 1.1g.
4. Pad pitch: up and down 100mil (2.54mm), left and right 600mil (15.24mm).
5. Measuring dimensions: Acceleration: 3-dimensional angular velocity: 3-dimensional attitude angle: 3-dimensional.
6. Range: Acceleration: ± 16g, angular velocity: ± 2000 ° / s.
7. Resolution: Acceleration: 6.1e-5g, the angular velocity: 7.6e-3 ° / s.
8. Stability: acceleration: 0.01g, angular velocity 0.05 ° / s.
9. Attitude Stability Measurement: 0.01 °.
10. Data output frequency 100Hz (baud rate 115200) / 20Hz (baud rate 9600). 11, data interface: Serial (TTL level), I2C (directly connected MPU6050, no attitude output).

10, baud rate 115200kps / 9600kps.

3 Pin Description:



1. 허용 전압과 전류 확인
2. 각가속도와 각속도 범위 확인
3. 오차범위 확인
4. 통신속도 확인

1. DATASHEET

5.1 Host computer to the module

Instruction content	Features	Remarks
Lt; / RTI & gt;	Angle initialization	The Z-axis angle to zero

0x61	Using the serial port, disable I2C	Power-down save, it is recommended to use the host computer to modify
0x62	Disable the serial port, using the I2C interface	Power-down save, it is recommended to use the host computer to modify
0x63 ('c')	115200 baud rate, frame rate100Hz	Power-down save, it is recommended to use the host computer to modify
0x64 ('d')	9600 baud rate, frame rate20Hz	Power-down save, it is recommended to use the host computer to modify

Description:

1. module after power required to remain stationary inside the module MCU will perform automatic calibration module when stationary (elimination of gyro drift), after the calibration angle Z axis will be re-initialized to 0, Z axis angle output is 0 , Can be regarded as the signal that the automatic calibration completes.

2 The factory default settings using the serial port baud rate 115200, frame rate 100Hz. Configuration can be configured through the host computer software, because all configuration is saved, so only need to configure once.

5.2 module first bit machine:

Module sends first bit machine each frame of data is divided into three packets, each packet acceleration, angular velocity and angle bag package, 3.

Packet sequence output. 115200 baud rate when every 10ms output a data baud rate of 9600 when every 50ms output a data.

1. Stm32에 0x61또는 0x62를 보내서 어떤 모드를 쓸지를 선택.

2. 시리얼 창에 0x63 또는 0x64로 통신 속도도 설정할 수 있음.

3. 공장에서 기본 셋팅으로 통신속도를 115200으로 설정되어 나옴.

1. DATASHEET

The data number	Data content	meaning
Lt; / RTI & gt;	0x55 ('U')	Baotou
1	0x51 ('Q')	Identify this package is the acceleration package
2	AxL	X-axis acceleration low byte
3	AxH	X-axis acceleration high byte
4	AyL	y-axis acceleration low byte
5	AyH	y-axis acceleration high byte
6	AzL	z-axis acceleration low byte
7	AzH	z-axis acceleration high byte
8	TL	Temperature low byte
9	TH	Temperature high byte
10	Sum	Checksum

Acceleration calculation formula:

a_x=((AxH<<8)|AxL)/32768*16g (g is the gravitational acceleration, preferably 9.8m/s^2)

a_y=((AyH<<8)|AyL)/32768*16g (g of acceleration of gravity, it is desirable 9.8m/s^2)

a_z=((AzH<<8)|AzL)/32768*16g (g is the gravitational acceleration, the desirability of 9.8m/s^2) temperature calculation formula:

T=((TH<<8)|TL) /340+36.53 ℃

Checksum:

Sum = 0x55 + 0x51 + AxH + AxL + AyH + AyL + AzH + AzL + TH + TL

1. 시리얼 창에서 0x51 이후에 나오는 값이 각가속도이다.

2. 주의할 점!!
   Low , High 순으로 나옴.

1. DATASHEET

5.2.2 Angular velocity output:

The data number	Data content	meaning
Lt; / RTI & gt;	0x55 ('U')	Baotou
1	0x52 ('R')	Identifies this package as angular velocity packets
2	WxL	X-axis angular velocity low byte
3	WxH	X-axis acceleration high byte
4	WyL	y-axis acceleration low byte
5	WyH	y-axis acceleration high byte
6	WzL	z-axis acceleration low byte
7	WzH	z-axis acceleration high byte
8	TL	Temperature low byte
9	TH	Temperature high byte
10	Sum	Checksum

Angular velocity calculation formula:

w_x=((wxH<<8)|wxL)/32768*2000(°/s)

w_x=((wyH<<8)|wyL)/32768*2000(°/s)

w_x=((wzH<<8)|wzL)/32768*2000(°/s)

temperature calculation formula:

T=((TH<<8)|TL) /340+36.53 ℃

Checksum:

Sum = 0x55 + 0x52 + wxH + wxL + wyH + wyL + wzH + wzL + TH + TL

1. 시리얼 창에서 0x52 이후에 나오는 값이 각속도이다.

2. 주의할 점!!
   Low , High 순으로 나옴.

1. DATASHEET

5.2.3 Angle Output:

The data number	Data content	meaning
Lt; / RTI & gt;	0x55 ('U')	Baotou
1	0x53 ('S')	Identifies this package as an angle package
2	RollL	X axis angle low byte
3	RollH	X axis angle high byte
4	PitchL	y-axis angle of the low byte
5	PitchH	y-axis angle high byte
6	YawL	z axis angle low byte
7	YawH	z axis angle high byte
8	TL	Temperature low byte
9	TH	Temperature high byte
10	Sum	Checksum

Angular velocity calculation formula:

Roll angle (x-axis) Roll = ((RollH << 8) | RollL) / 32768 * 180 (°)

pitch angle (y-axis) Pitch = ((PitchH << 8) | PitchL) / 32768 * 180 (°)

yaw angle (z-axis) yaw = ((YawH << 8) | YawL) / 32768 * 180

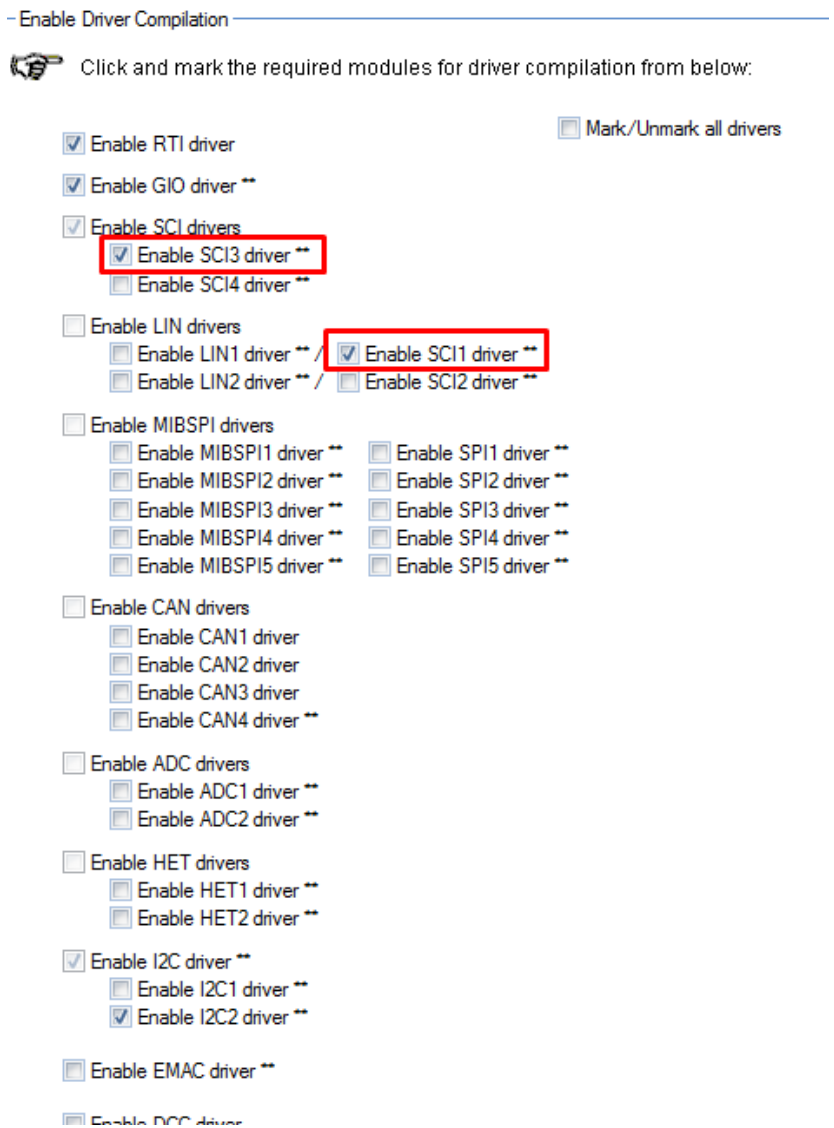
(°) temperature is calculated:

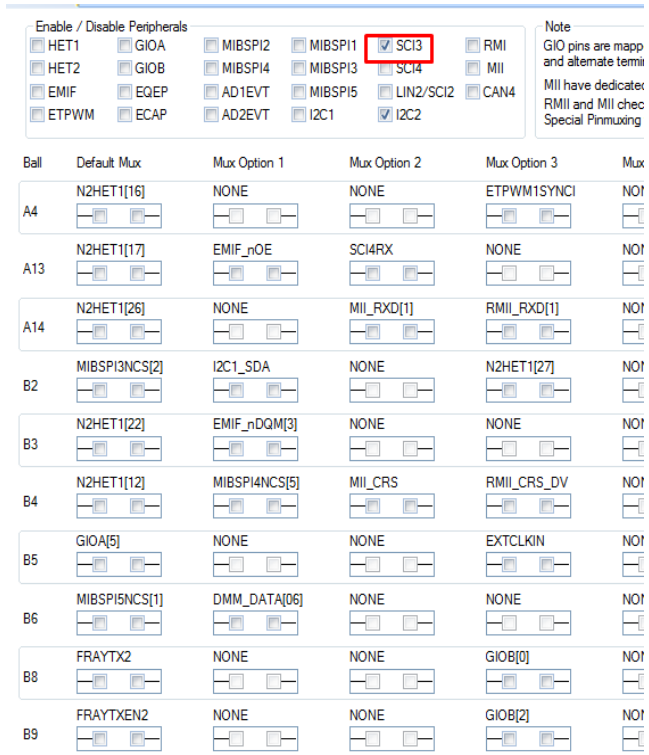
T = ((TH << 8) | TL) /340+36.53 ℃

1. 시리얼 창에서 0x53 이후에 나오는 값이 각도이다.

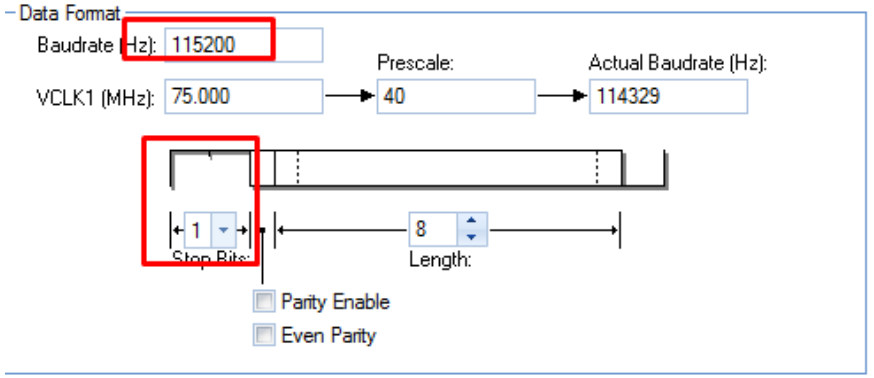
2. 주의할 점!!
   Low , High 순으로 나옴.

2. HALCOGEN 설정





Level: TTL-level (non-RS232 level, when the module is connected to the wrong RS232 level could cause damage to the modules) Baud rate: 115200/9600, 1 stop bit, parity bit 0...



3. CODE

```c
#include "HL_sys_common.h"
#include "HL_etpwm.h"
#include "HL_system.h"
#include "HL_sci.h"
#include "HL_rti.h"

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

#define UART sciREG1

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

uint32 rx_data =0;
uint32 tmp =0;
uint32 value =0;
uint32 duty_arr[10] = {1000, 1020, 1040, 1060, 1080, 1100, 1150, 1200, 1400,2000

int acc_l_x, acc_l_y, acc_l_z;
int acc_h_x, acc_h_y, acc_h_z;

double real_acc_x, real_acc_y, real_acc_z;

int gyro_x, gyro_y, gyro_z;
double real_gyro_x, real_gyro_y, real_gyro_z;

int l_roll, l_pitch, l_yaw;
int h_roll, h_pitch, h_yaw;
double real_roll, real_pitch, real_yaw;
```

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

    sciInit();
    disp_set("sci Init Success!\n\r");
    etpwmInit();
    disp_set("pwm Init Success!\n\r");

    rtiInit();
    rtiEnableNotification(rtiREG1, rtiNOTIFICATION_COMPARE0);
    _enable_IRQ_interrupt_();
    rtiStartCounter(rtiREG1, rtiCOUNTER_BLOCK0);

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

    sciSendByte(sciREG3, 0x61); // using the serial port , disable I2C

    etpwmStartTBCLK();
    wait(10000);

    for (;;)
    {

        if (sciReceiveByte(sciREG3) == 0x53)
        {
            /*roll*/
            l_roll = sciReceiveByte(sciREG3);
            h_roll = sciReceiveByte(sciREG3);
            h_roll <<= 8;
            h_roll |= l_roll;
            real_roll = ((double) h_roll) / 32768.0 * 180;
            /*pitch*/
            l_pitch = sciReceiveByte(sciREG3);
            h_pitch = sciReceiveByte(sciREG3);
            h_pitch <<= 8;
            h_pitch |= l_pitch;
            real_pitch = ((double) h_pitch) / 32768.0 * 180;
            /*yaw*/
            l_yaw = sciReceiveByte(sciREG3);
            h_yaw = sciReceiveByte(sciREG3);
            h_yaw <<= 8;
            h_yaw |= l_yaw;
            real_yaw = ((double) h_yaw) / 32768.0 * 180;

            sprintf(txt_buf, "roll = %lf \t pitch = %lf \t yaw = %lf \n\r\0",
                    real_roll, real_pitch, real_yaw);
```

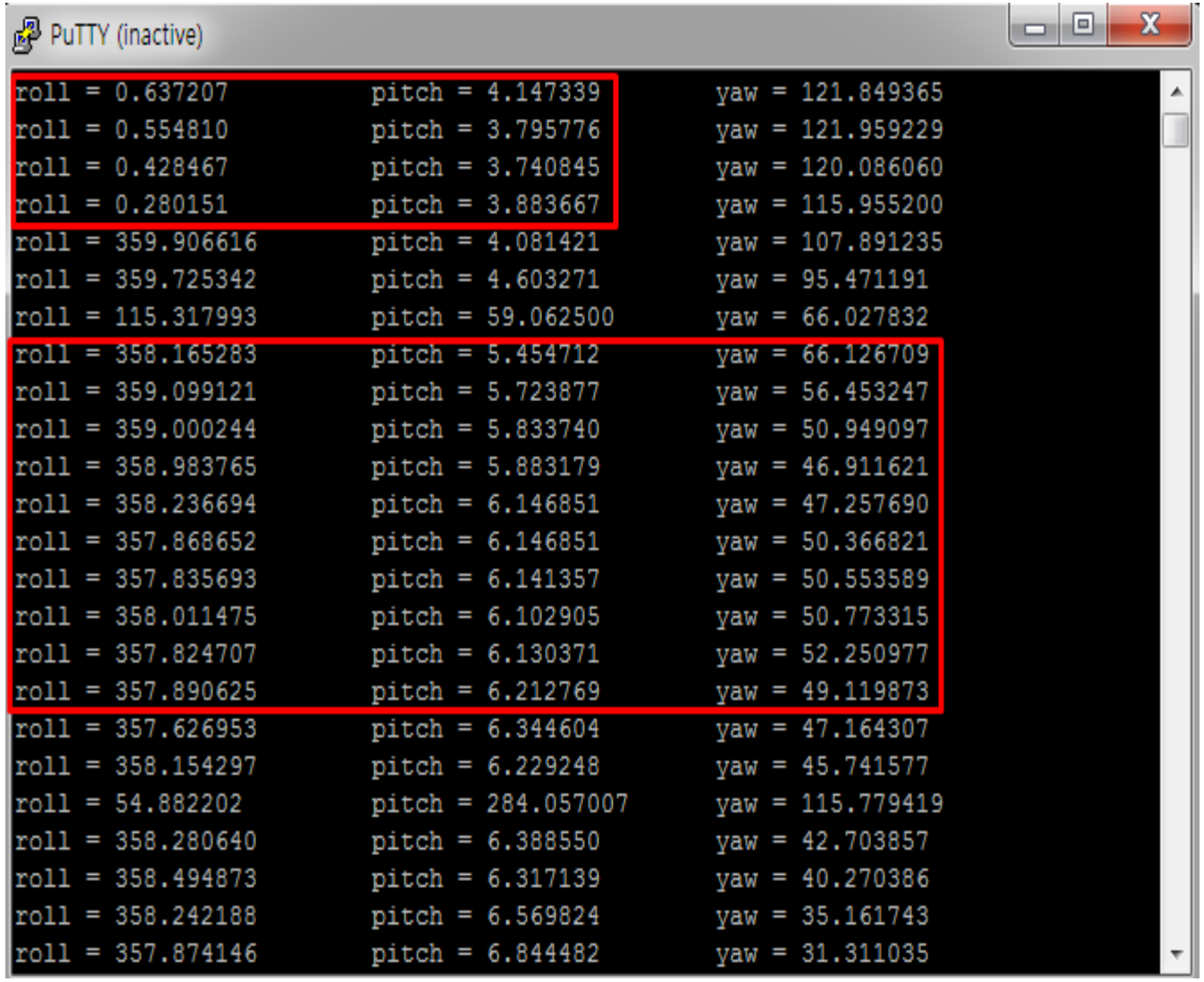
```c
void pwmSet(){
    value = duty_arr[rx_data];
    etpwmSetCmpA(etpwmREG1, value);
    wait(10000);
}

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 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);
}
```

4. SIMULATION



1. roll 과 pitch 값이 잘 나오는 것을 확인 할 수 있다.

2. 반대 방향일 때는 360에서 역으로 빼지는 것을 확인이 된다.