Xilinx Zynq FPGA,TI DSP, MCU 기반의 프로그래밍 전문가 과정

날 짜 : 2018 . 5. 14

강사 – Innova Lee(이상훈)
gcccompil3r@gmail.com
학생 – 정한별
hanbulkr@gmail.com

UART_PWM

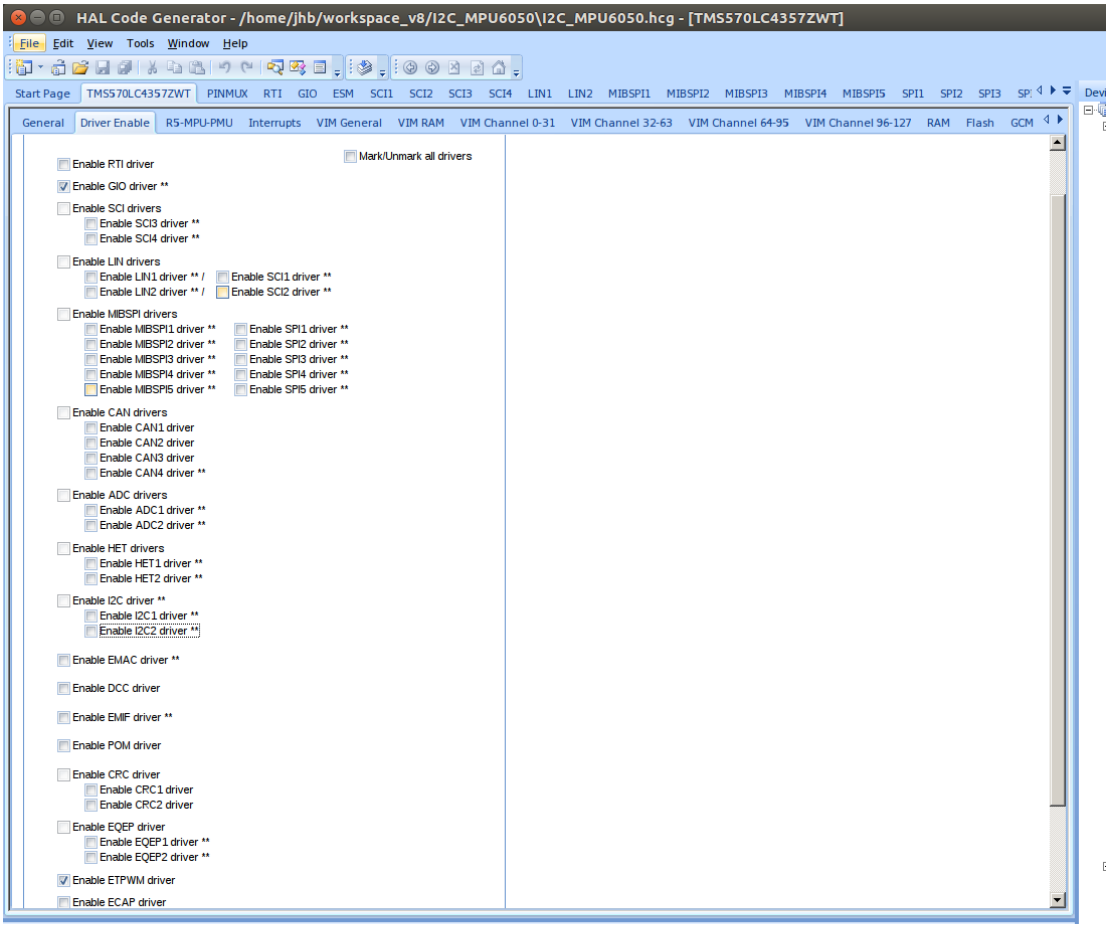
<CODE>

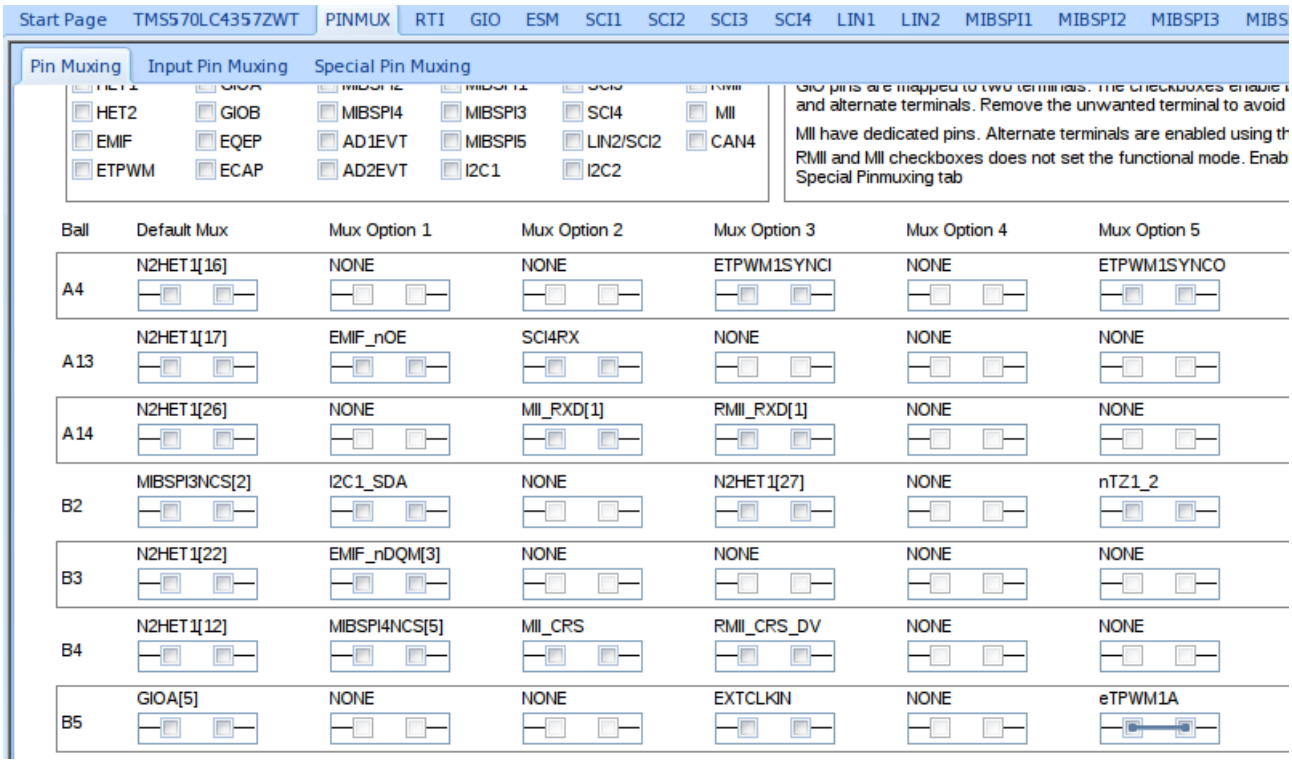
```
#include "HL_sys_common.h"
#include "HL_system.h"
#include "HL_etpwm.h"
#include "HL_sci.h"

#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);

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

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

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

    sciInit();

    sprintf(txt_buf, "SCI Configuration Success!!\n\r");
    buf_len = strlen(txt_buf);
    sciDisplayText(sciREG1, (uint8 *) txt_buf, buf_len);

    etpwmInit();

    sprintf(txt_buf, "ETPWM Configuration Success!!\n\r");
    buf_len = strlen(txt_buf);
    sciDisplayText(sciREG1, (uint8 *) txt_buf, buf_len);

    etpwmStartTBCLK();
    wait(10000);

    sprintf(txt_buf, "Please Press Key(0 ~ 5)!!\n\r");
    buf_len = strlen(txt_buf);
    sciDisplayText(sciREG1, (uint8 *) txt_buf, buf_len);

    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();
```

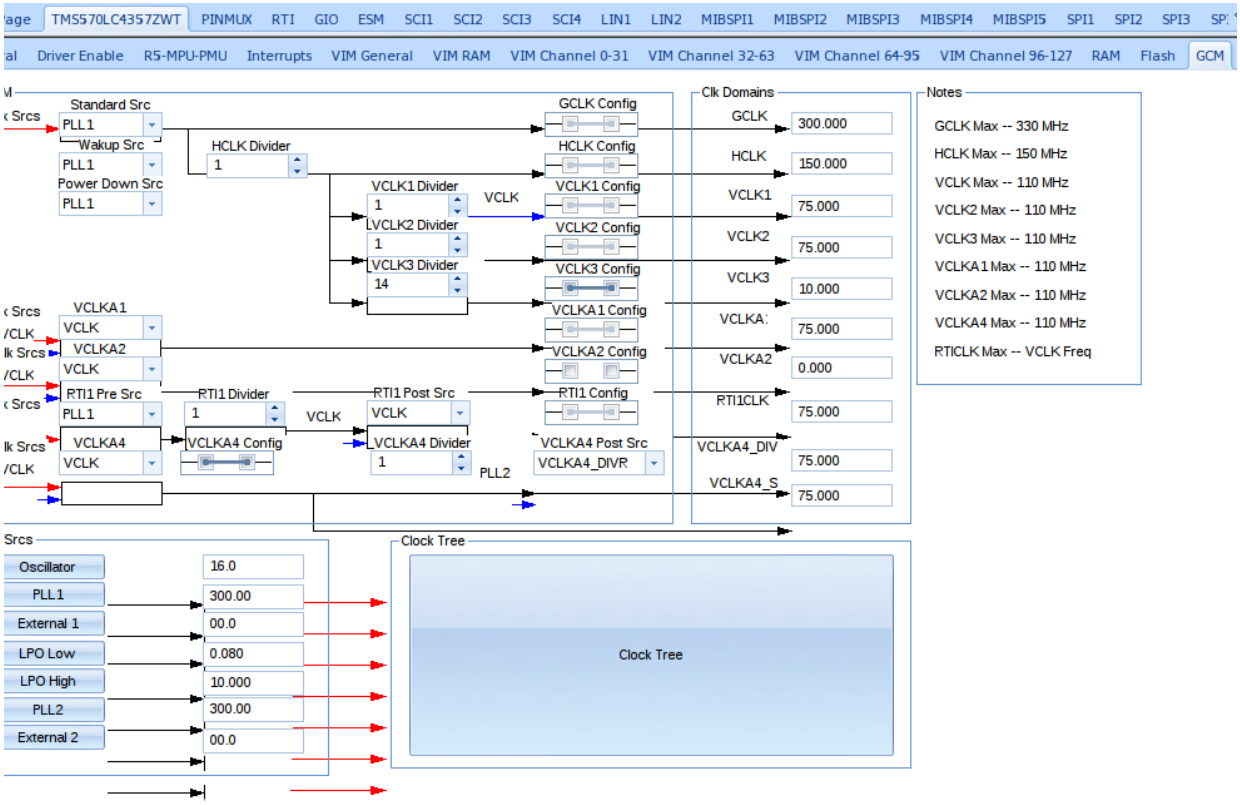
```
        sprintf(txt_buf, "PWM Duty = %d\n\r\0", value);
        buf_len = strlen(txt_buf);
        sciDisplayText(sciREG1, (uint8 *) txt_buf, buf_len);
    }

    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++);
    }
}
```

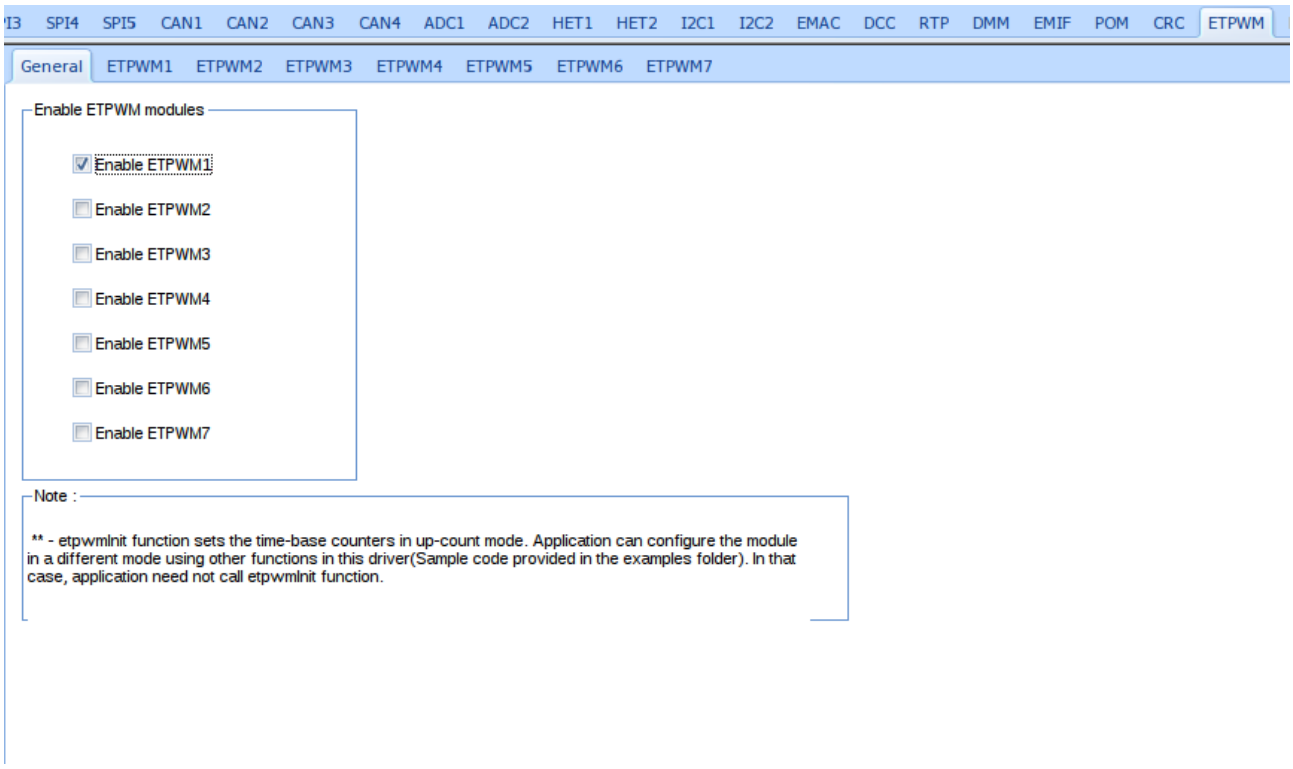
- 유아트 통신을 확인하는 프로그램이다.
- 서보모터로 0~5 까지 누르면 설정된 배열값에 의해서 움직인다.
- 6 을 누르면 오버플로우 되면서 값이 0 으로 셋팅이 된다.

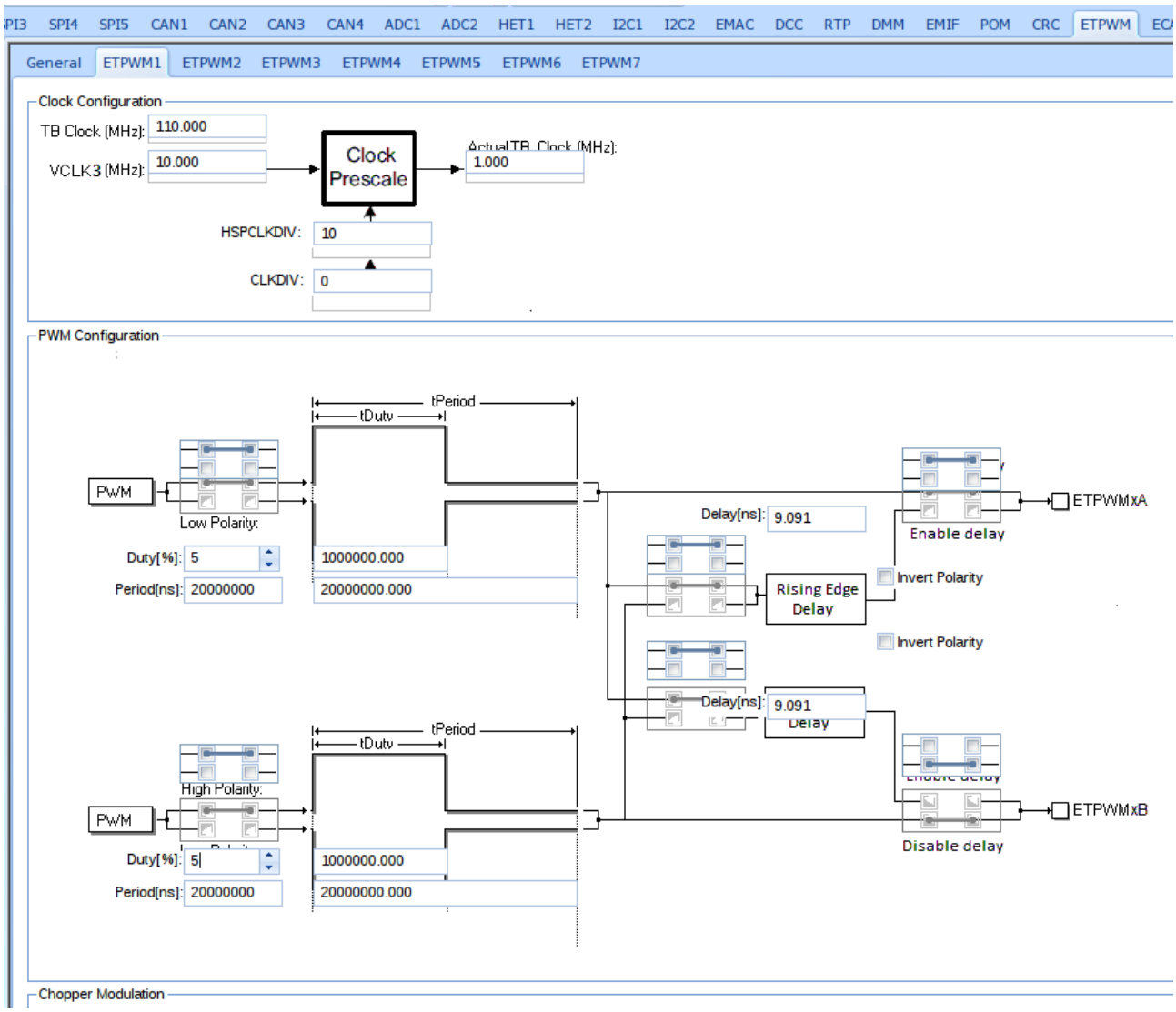
<HALCOGEN 설정>


HAL Code Generator - /home/jhb/workspace_v8/I2C_MPU6050\I2C_MPU6050.hcg - [TMS570LC4357ZWT]
File Edit View Tools Window Help
Start Page TMS570LC4357ZWT PINMUX RTI GIO ESM SCI1 SCI2 SCI3 SCI4 LIN1 LIN2 MIBSPI1 MIBSPI2 MIBSPI3 MIBSPI4 MIBSPI5 SPI1 SPI2 SPI3
General Driver Enable R5-MPU-PMU Interrupts VIM General VIM RAM VIM Channel 0-31 VIM Channel 32-63 VIM Channel 64-95 VIM Channel 96-127 RAM Flash GCM
Mark/Unmark all drivers
Enable RTI driver
Enable GIO driver **
Enable SCI drivers
Enable SCI3 driver **
Enable SCI4 driver **
Enable LIN drivers
Enable LIN1 driver ** / Enable SCI1 driver **
Enable LIN2 driver ** / Enable SCI2 driver **
Enable MIBSPI drivers
Enable MIBSPI1 driver ** Enable SPI1 driver **
Enable MIBSPI2 driver ** Enable SPI2 driver **
Enable MIBSPI3 driver ** Enable SPI3 driver **
Enable MIBSPI4 driver ** Enable SPI4 driver **
Enable MIBSPI5 driver ** Enable SPI5 driver **
Enable CAN drivers
Enable CAN1 driver
Enable CAN2 driver
Enable CAN3 driver
Enable CAN4 driver **
Enable ADC drivers
Enable ADC1 driver **
Enable ADC2 driver **
Enable HET drivers
Enable HET1 driver **
Enable HET2 driver **
Enable I2C driver **
Enable I2C1 driver **
Enable I2C2 driver **
Enable EMAC driver **
Enable DCC driver
Enable EMIF driver **
Enable POM driver
Enable CRC driver
Enable CRC1 driver
Enable CRC2 driver
Enable EQEP driver
Enable EQEP1 driver **
Enable EQEP2 driver **
Enable ETPWM driver
Enable ECAP driver


Start Page TMS570LC4357ZWT PINMUX RTI GIO ESM SCI1 SCI2 SCI3 SCI4 LIN1 LIN2 MIBSPI1 MIBSPI2 MIBSPI3
Pin Muxing Input Pin Muxing Special Pin Muxing
HET2 GIOB MIBSPI4 MIBSPI3 SCI4 MII
EMIF EQEP AD1EVT MIBSPI5 LIN2/SCI2 CAN4
ETPWM ECAP AD2EVT I2C1 I2C2
and alternate terminals. Remove the unwanted terminal to avoid
MII have dedicated pins. Alternate terminals are enabled using th
RMII and MII checkboxes does not set the functional mode. Enab
Special Pinmuxing tab
Ball Default Mux Mux Option 1 Mux Option 2 Mux Option 3 Mux Option 4 Mux Option 5
A4 N2HET1[16] NONE NONE ETPWM1SYNCI NONE ETPWM1SYNCO
A13 N2HET1[17] EMIF_nOE SCI4RX NONE NONE NONE
A14 N2HET1[26] NONE MII_RXD[1] RMII_RXD[1] NONE NONE
B2 MIBSPI3NCS[2] I2C1_SDA NONE N2HET1[27] NONE nTZ1_2
B3 N2HET1[22] EMIF_nDQM[3] NONE NONE NONE NONE
B4 N2HET1[12] MIBSPI4NCS[5] MII_CRS RMII_CRS_DV NONE NONE
B5 GIOA[5] NONE NONE EXTCLKIN NONE eTPWM1A


TMS570LC4357ZWT
PINMUX
RTI
GIO
ESM
SCI1
SCI2
SCI3
SCI4
LIN1
LIN2
MIBSPI1
MIBSPI2
MIBSPI3
MIBSPI4
MIBSPI5
SPI1
SPI2
SPI3
Driver Enable
R5-MPU-PMU
Interrupts
VIM General
VIM RAM
VIM Channel 0-31
VIM Channel 32-63
VIM Channel 64-95
VIM Channel 96-127
RAM
Flash
GCM
Standard Src
PLL1
Wakup Src
PLL1
Power Down Src
PLL1
HCLK Divider
1
GCLK Config
HCLK Config
VCLK1 Divider
1
VCLK
VCLK1 Config
VCLK2 Divider
1
VCLK2 Config
VCLK3 Divider
14
VCLK3 Config
VCLKA1
VCLK
VCLKA1 Config
VCLKA2
VCLK
VCLKA2 Config
RTI1 Pre Src
PLL1
RTI1 Divider
1
VCLK
RTI1 Post Src
VCLK
RTI1 Config
VCLKA4
VCLK
VCLKA4 Config
VCLKA4 Divider
1
VCLKA4 Post Src
VCLKA4_DIVR
PLL2
Clk Domains
GCLK
300.000
HCLK
150.000
VCLK1
75.000
VCLK2
75.000
VCLK3
10.000
VCLKA1
75.000
VCLKA2
0.000
RTI1CLK
75.000
VCLKA4_DIV
75.000
VCLKA4_S
75.000
Notes
GCLK Max -- 330 MHz
HCLK Max -- 150 MHz
VCLK Max -- 110 MHz
VCLK2 Max -- 110 MHz
VCLK3 Max -- 110 MHz
VCLKA1 Max -- 110 MHz
VCLKA2 Max -- 110 MHz
VCLKA4 Max -- 110 MHz
RTICLK Max -- VCLK Freq
Srcs
Oscillator
16.0
PLL1
300.00
External 1
00.0
LPO Low
0.080
LPO High
10.000
PLL2
300.00
External 2
00.0
Clock Tree
Clock Tree


SPI4
SPI5
CAN1
CAN2
CAN3
CAN4
ADC1
ADC2
HET1
HET2
I2C1
I2C2
EMAC
DCC
RTP
DMM
EMIF
POM
CRC
ETPWM
General
ETPWM1
ETPWM2
ETPWM3
ETPWM4
ETPWM5
ETPWM6
ETPWM7
Enable ETPWM modules
Enable ETPWM1
Enable ETPWM2
Enable ETPWM3
Enable ETPWM4
Enable ETPWM5
Enable ETPWM6
Enable ETPWM7
Note :
** - etpwmInit function sets the time-base counters in up-count mode. Application can configure the module in a different mode using other functions in this driver(Sample code provided in the examples folder). In that case, application need not call etpwmInit function.


SPI3 SPI4 SPI5 CAN1 CAN2 CAN3 CAN4 ADC1 ADC2 HET1 HET2 I2C1 I2C2 EMAC DCC RTP DMM EMIF POM CRC ETPWM
General ETPWM1 ETPWM2 ETPWM3 ETPWM4 ETPWM5 ETPWM6 ETPWM7
Clock Configuration
TB Clock (MHz): 110.000
VCLK3 (MHz): 10.000
Clock Prescale
ActualTB Clock (MHz): 1.000
HSPCLKDIV: 10
CLKDIV: 0
PWM Configuration
tPeriod
tDuty
PWM
Low Polarity:
Duty[%]: 5
Period[ns]: 20000000
1000000.000
20000000.000
Delay[ns]: 9.091
Enable delay
ETPWMxA
Rising Edge Delay
Invert Polarity
Invert Polarity
Delay[ns]: 9.091
Delay
High Polarity:
PWM
Duty[%]: 5
Period[ns]: 20000000
1000000.000
20000000.000
Disable delay
ETPWMxB
Chopper Modulation

I2C_MPU6050

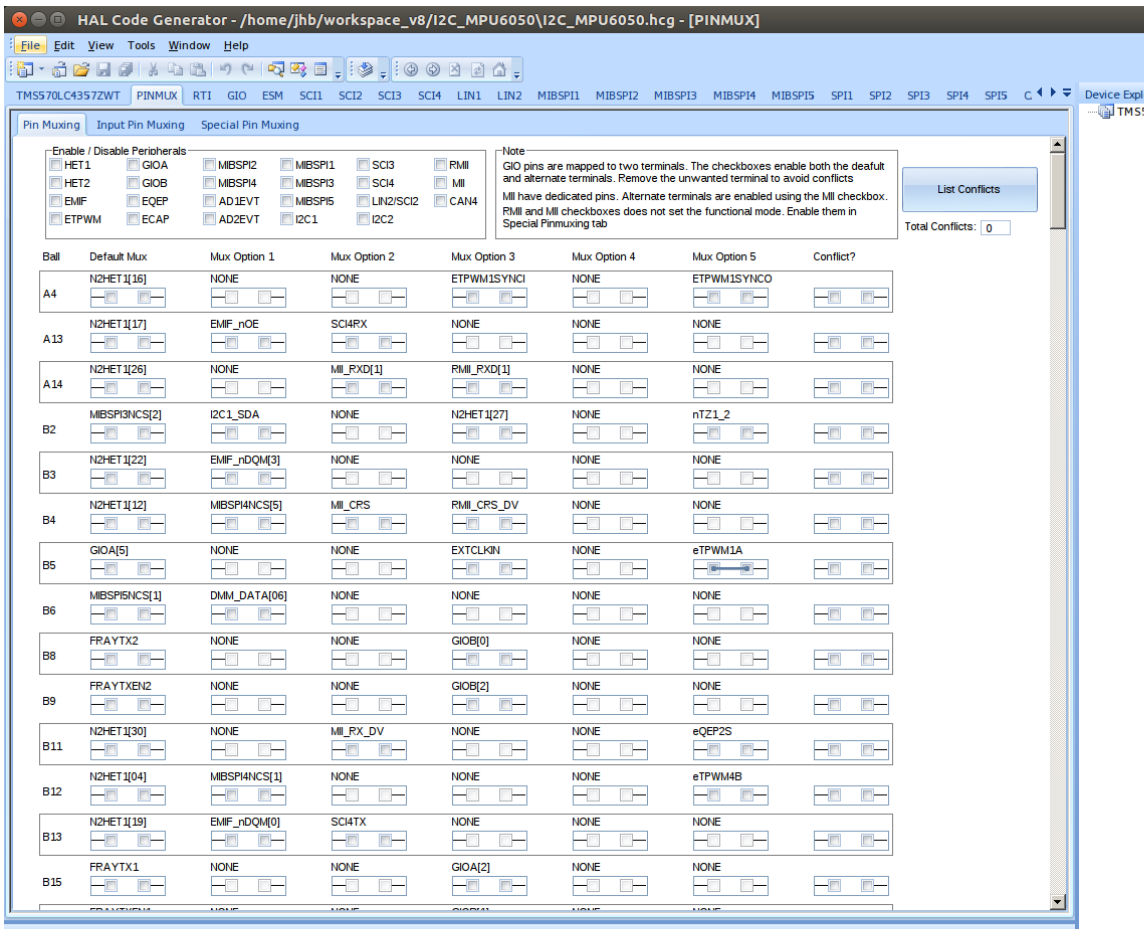
<CODE>

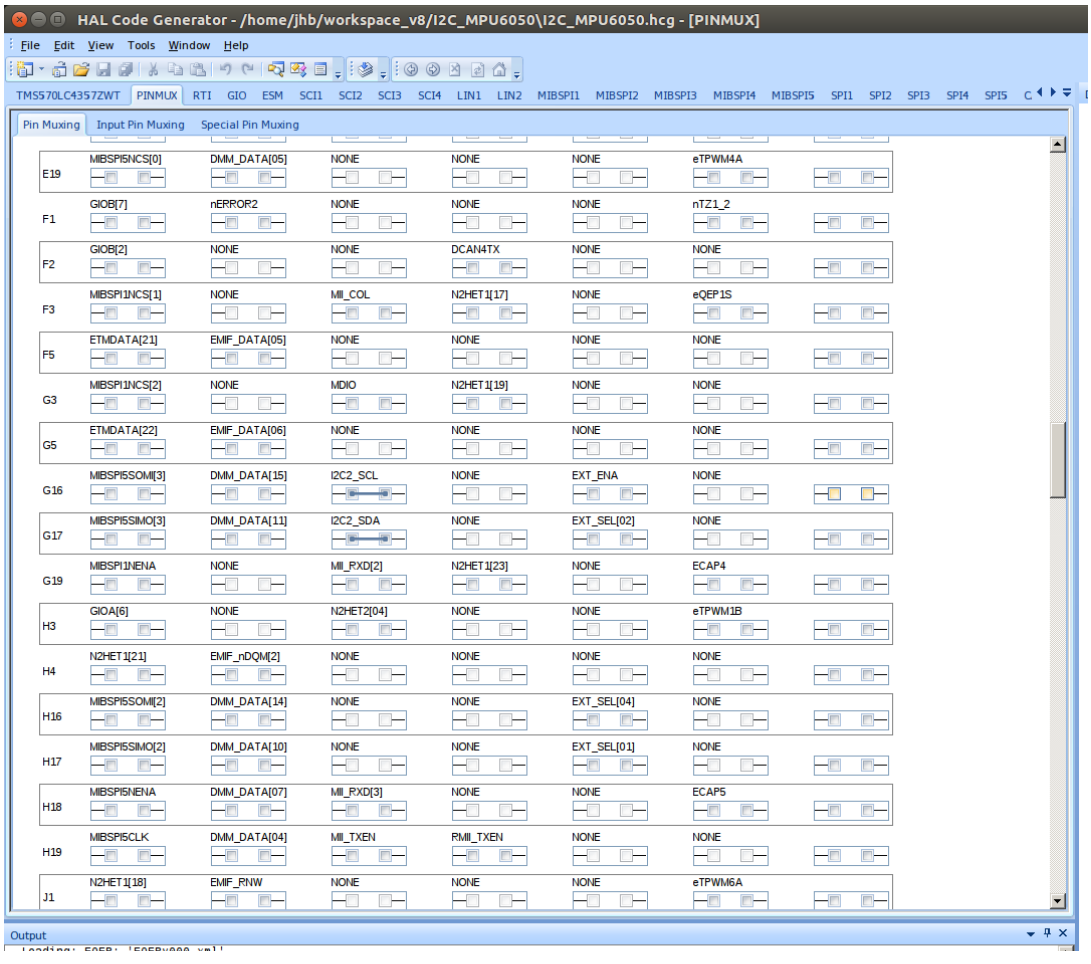
```
#include "HL_sys_common.h"
#include "HL_system.h"
#include "HL_etpwm.h"
#include "HL_sci.h"
#include "HL_gio.h"
#include "HL_i2c.h"
#include "HL_rti.h"

#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[256] = { 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();
```

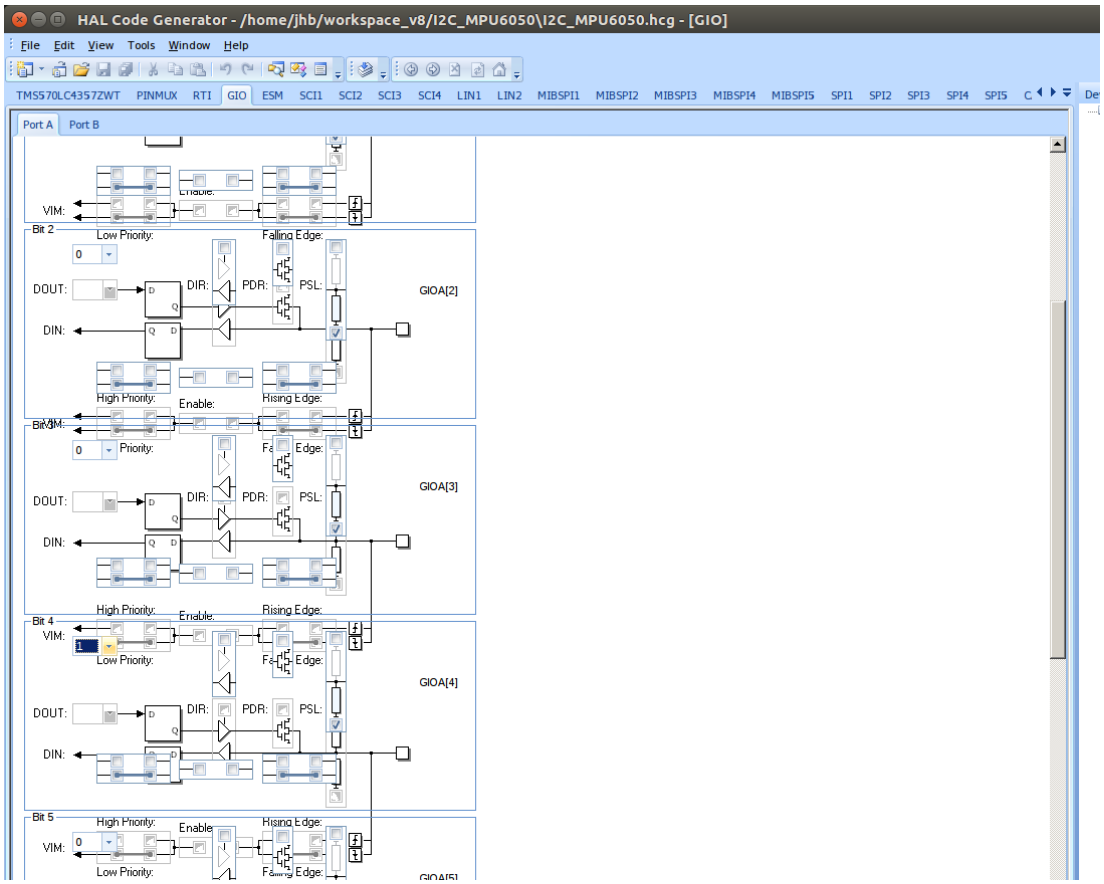
```
    disp_set("MPU6050 Accelerometer Configure Success!!\n\r\0");

    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(1000000);

    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);
```

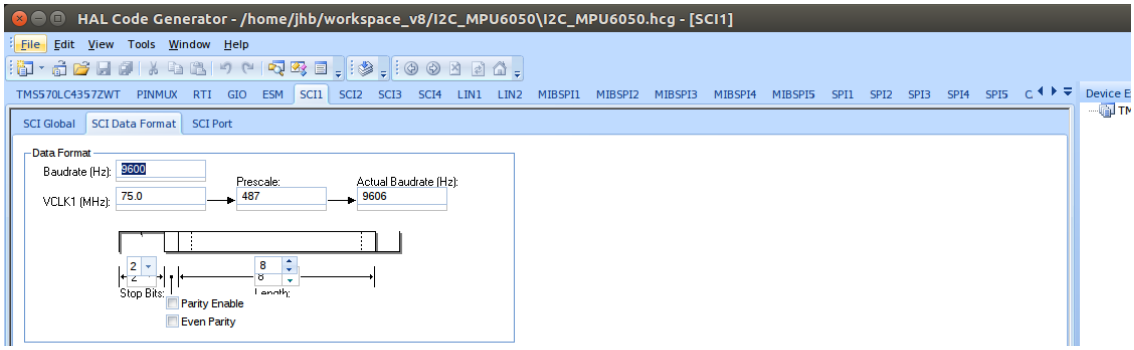
```
        sciDisplayText(sciREG1, (uint8 *)txt_buf, buf_len);
    }
#endif
    return 0;
}

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);
```

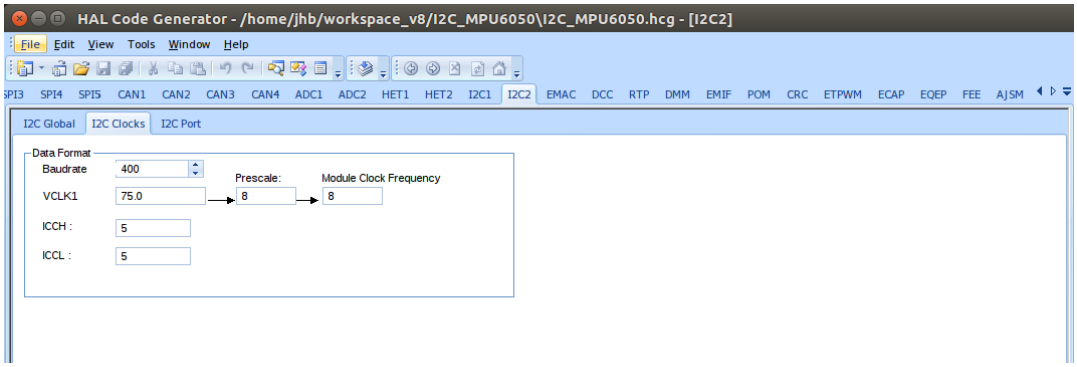
```
    i2cSetDirection(i2cREG2, I2C_TRANSMITTER);
    i2cSetCount(i2cREG2, cnt + 1);
    i2cSetMode(i2cREG2, I2C_MASTER);
    i2cSetStop(i2cREG2);
    i2cSetStart(i2cREG2);
    i2cSendByte(i2cREG2, slave_word_address);
    i2cSend(i2cREG2, cnt, data);

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

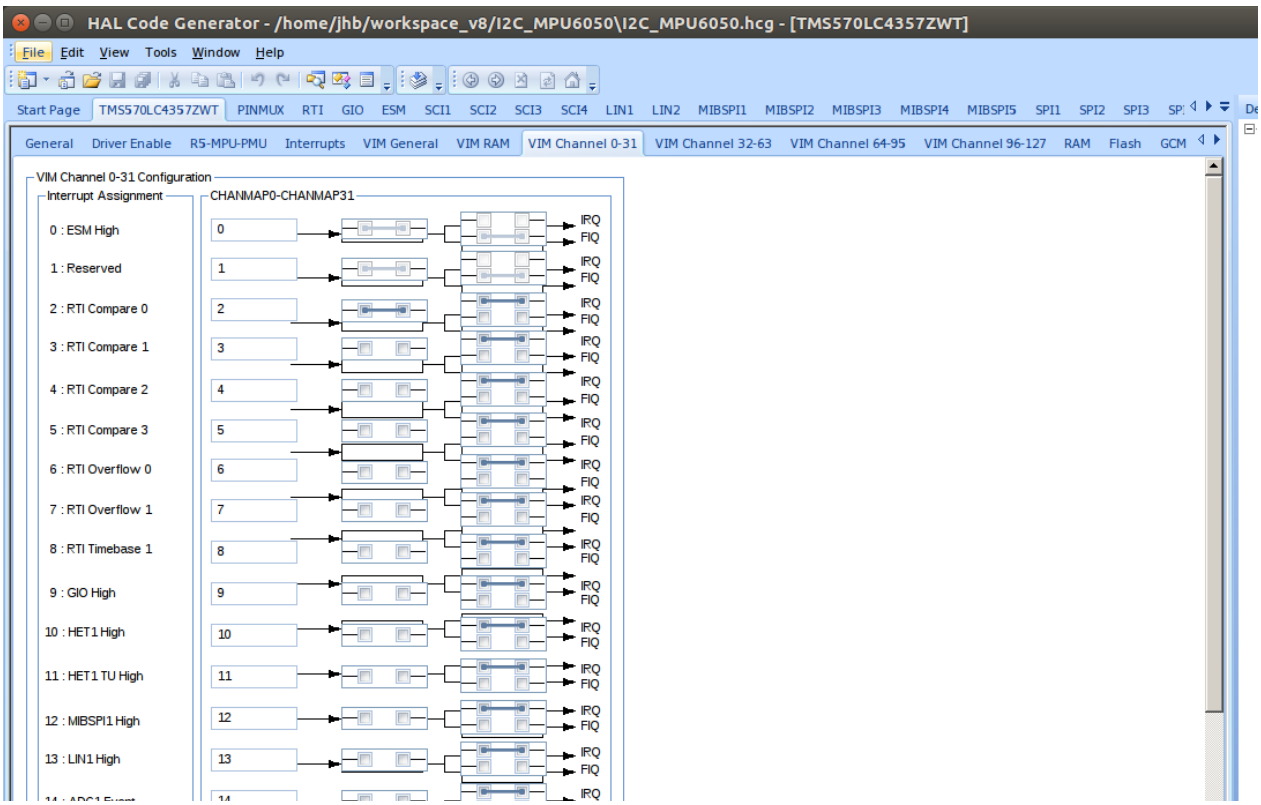

HAL Code Generator - /home/jhb/workspace_v8/I2C_MPU6050\I2C_MPU6050.hcg - [PINMUX]
File Edit View Tools Window Help
TMS570LC4357ZWT PINMUX RTI GIO ESM SCI1 SCI2 SCI3 SCI4 LIN1 LIN2 MIBSPI1 MIBSPI2 MIBSPI3 MIBSPI4 MIBSPI5 SPI1 SPI2 SPI3 SPI4 SPI5
Pin Muxing Input Pin Muxing Special Pin Muxing
Enable / Disable Perioherals
HET1 HET2 EMIF ETPWM GIOA GIOB EQEP ECAP MIBSPI2 MIBSPI4 AD1EVT AD2EVT MIBSPI1 MIBSPI3 MIBSPI5 I2C1 SCI3 SCI4 LIN2/SCI2 I2C2 RMII MII CAN4
Note
GIO pins are mapped to two terminals. The checkboxes enable both the deafult and alternate terminals. Remove the unwanted terminal to avoid conflicts
MII have dedicated pins. Alternate terminals are enabled using the MII checkbox.
RMII and MII checkboxes does not set the functional mode. Enable them in Special Pinmuxing tab
List Conflicts
Total Conflicts: 0
Ball Default Mux Mux Option 1 Mux Option 2 Mux Option 3 Mux Option 4 Mux Option 5 Conflict?
A4 N2HET1[16] NONE NONE ETPWM1SYNCI NONE ETPWM1SYNCO
A13 N2HET1[17] EMIF_nOE SCI4RX NONE NONE NONE
A14 N2HET1[26] NONE MII_RXD[1] RMII_RXD[1] NONE NONE
B2 MIBSPI3NCS[2] I2C1_SDA NONE N2HET1[27] NONE nTZ1_2
B3 N2HET1[22] EMIF_nDQM[3] NONE NONE NONE NONE
B4 N2HET1[12] MIBSPI4NCS[5] MII_CRS RMII_CRS_DV NONE NONE
B5 GIOA[5] NONE NONE EXTCLKIN NONE eTPWM1A
B6 MIBSPI5NCS[1] DMM_DATA[06] NONE NONE NONE NONE
B8 FRAYTX2 NONE NONE GIOB[0] NONE NONE
B9 FRAYTXEN2 NONE NONE GIOB[2] NONE NONE
B11 N2HET1[30] NONE MII_RX_DV NONE NONE eQEP2S
B12 N2HET1[04] MIBSPI4NCS[1] NONE NONE NONE eTPWM4B
B13 N2HET1[19] EMIF_nDQM[0] SCI4TX NONE NONE NONE
B15 FRAYTX1 NONE NONE GIOA[2] NONE NONE

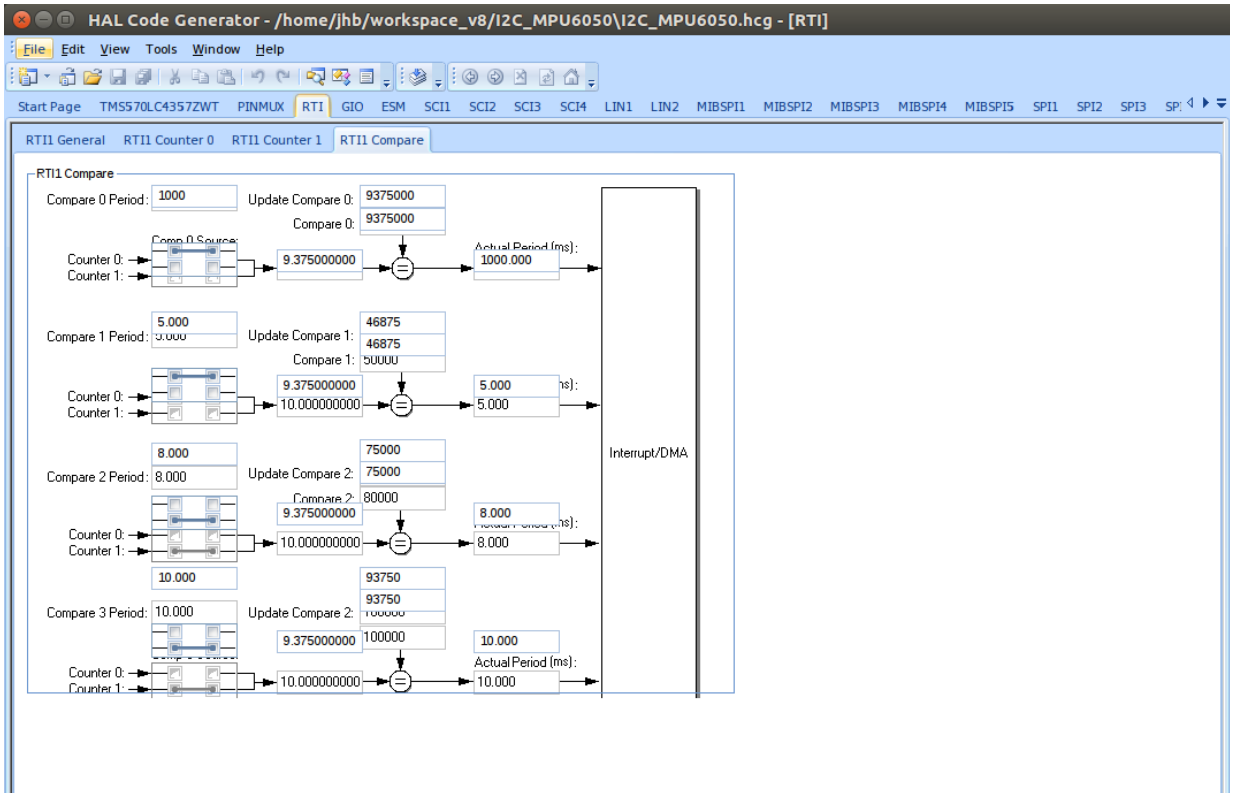

HAL Code Generator - /home/jhb/workspace_v8/I2C_MPU6050\I2C_MPU6050.hcg - [PINMUX]
File Edit View Tools Window Help
TMS570LC4357ZWT PINMUX RTI GIO ESM SCI1 SCI2 SCI3 SCI4 LIN1 LIN2 MIBSPI1 MIBSPI2 MIBSPI3 MIBSPI4 MIBSPI5 SPI1 SPI2 SPI3 SPI4 SPI5
Pin Muxing Input Pin Muxing Special Pin Muxing
E19 MIBSPI5NCS[0] DMM_DATA[05] NONE NONE NONE eTPWM4A
F1 GIOB[7] nERROR2 NONE NONE NONE nTZ1_2
F2 GIOB[2] NONE NONE DCAN4TX NONE NONE
F3 MIBSPI1NCS[1] NONE MII_COL N2HET1[17] NONE eQEP1S
F5 ETMDATA[21] EMIF_DATA[05] NONE NONE NONE NONE
G3 MIBSPI1NCS[2] NONE MDIO N2HET1[19] NONE NONE
G5 ETMDATA[22] EMIF_DATA[06] NONE NONE NONE NONE
G16 MIBSPI5SOMI[3] DMM_DATA[15] I2C2_SCL NONE EXT_ENA NONE
G17 MIBSPI5SIMO[3] DMM_DATA[11] I2C2_SDA NONE EXT_SEL[02] NONE
G19 MIBSPI1NENA NONE MII_RXD[2] N2HET1[23] NONE ECAP4
H3 GIOA[6] NONE N2HET2[04] NONE NONE eTPWM1B
H4 N2HET1[21] EMIF_nDQM[2] NONE NONE NONE NONE
H16 MIBSPI5SOMI[2] DMM_DATA[14] NONE NONE EXT_SEL[04] NONE
H17 MIBSPI5SIMO[2] DMM_DATA[10] NONE NONE EXT_SEL[01] NONE
H18 MIBSPI5NENA DMM_DATA[07] MII_RXD[3] NONE NONE ECAP5
H19 MIBSPI5CLK DMM_DATA[04] MII_TXEN RMII_TXEN NONE NONE
J1 N2HET1[18] EMIF_RNW NONE NONE NONE eTPWM6A
Output

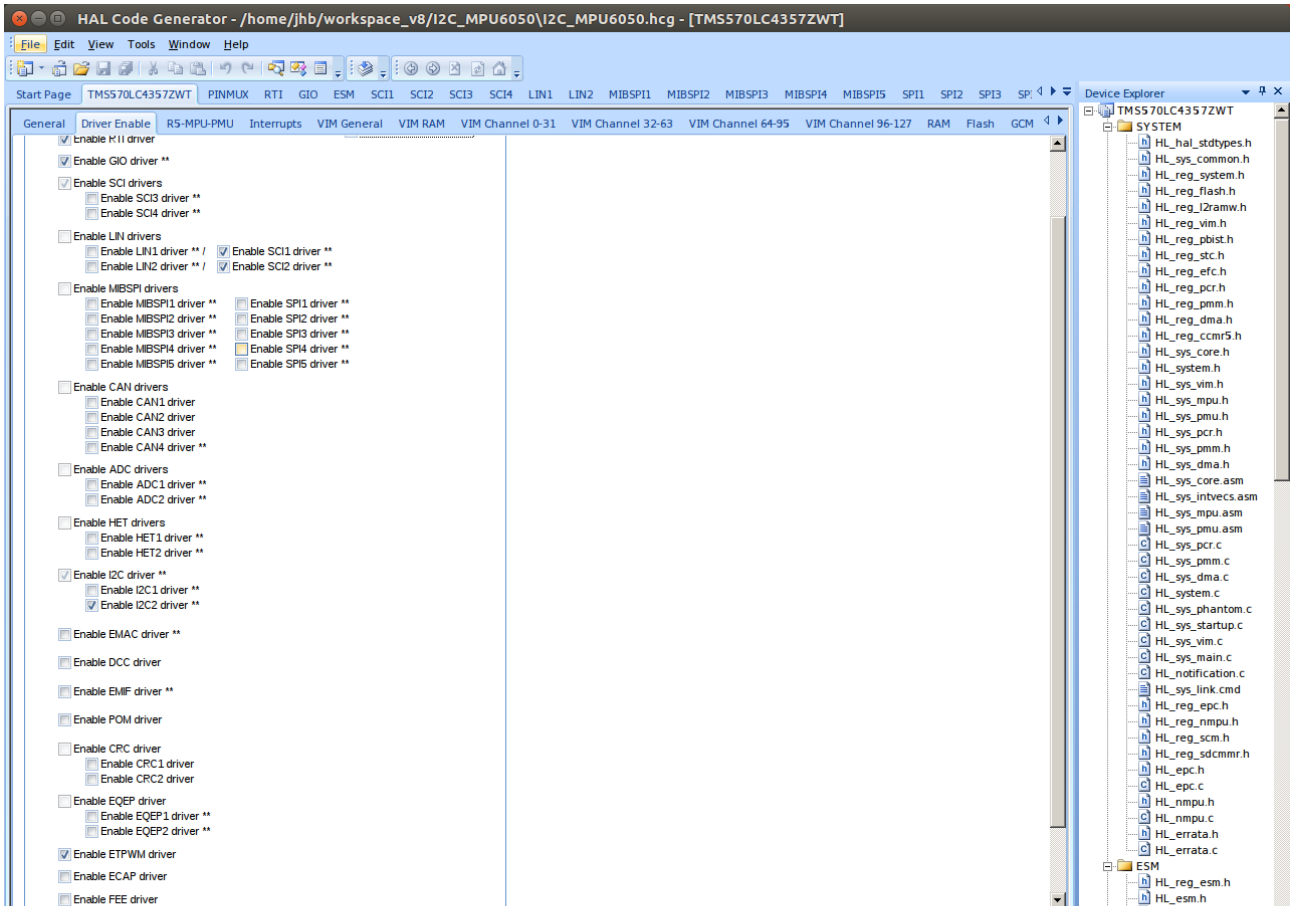

HAL Code Generator - /home/jhb/workspace_v8/I2C_MPU6050\I2C_MPU6050.hcg - [GIO]
File Edit View Tools Window Help
TMS570LC4357ZWT PINMUX RTI GIO ESM SCI1 SCI2 SCI3 SCI4 LIN1 LIN2 MIBSPI1 MIBSPI2 MIBSPI3 MIBSPI4 MIBSPI5 SPI1 SPI2 SPI3 SPI4 SPI5
Port A Port B
Bit 2
VIM:
Low Priority:
Falling Edge:
DOUT:
DIR:
PDR:
PSL:
GIOA[2]
DIN:
High Priority:
Enable:
Rising Edge:
Bit 3
GIOA[3]
Bit 4
GIOA[4]
Bit 5


HAL Code Generator - /home/jhb/workspace_v8/I2C_MPU6050\I2C_MPU6050.hcg - [SCI1]
File Edit View Tools Window Help
TMS570LC4357ZWT PINMUX RTI GIO ESM SCI1 SCI2 SCI3 SCI4 LIN1 LIN2 MIBSPI1 MIBSPI2 MIBSPI3 MIBSPI4 MIBSPI5 SPI1 SPI2 SPI3 SPI4 SPI5
SCI Global SCI Data Format SCI Port
Data Format
Baudrate (Hz): 9600
VCLK1 (MHz): 75.0
Prescale: 487
Actual Baudrate (Hz): 9606
Stop Bits:
Parity Enable
Even Parity


HAL Code Generator - /home/jhb/workspace_v8/I2C_MPU6050\I2C_MPU6050.hcg - [I2C2]
File Edit View Tools Window Help
SPI3 SPI4 SPI5 CAN1 CAN2 CAN3 CAN4 ADC1 ADC2 HET1 HET2 I2C1 I2C2 EMAC DCC RTP DMM EMIF POM CRC ETPWM ECAP EQEP FEE AJSM
I2C Global I2C Clocks I2C Port
Data Format
Baudrate 400
Prescale:
Module Clock Frequency
VCLK1 75.0
8
8
ICCH : 5
ICCL : 5


HAL Code Generator - /home/jhb/workspace_v8/I2C_MPU6050\I2C_MPU6050.hcg - [TMS570LC4357ZWT]
File Edit View Tools Window Help
Start Page TMS570LC4357ZWT PINMUX RTI GIO ESM SCI1 SCI2 SCI3 SCI4 LIN1 LIN2 MIBSPI1 MIBSPI2 MIBSPI3 MIBSPI4 MIBSPI5 SPI1 SPI2 SPI3
General Driver Enable R5-MPU-PMU Interrupts VIM General VIM RAM VIM Channel 0-31 VIM Channel 32-63 VIM Channel 64-95 VIM Channel 96-127 RAM Flash GCM
VIM Channel 0-31 Configuration
Interrupt Assignment
CHANMAP0-CHANMAP31
0 : ESM High
1 : Reserved
2 : RTI Compare 0
3 : RTI Compare 1
4 : RTI Compare 2
5 : RTI Compare 3
6 : RTI Overflow 0
7 : RTI Overflow 1
8 : RTI Timebase 1
9 : GIO High
10 : HET1 High
11 : HET1 TU High
12 : MIBSPI1 High
13 : LIN1 High
IRQ
FIQ


HAL Code Generator - /home/jhb/workspace_v8/I2C_MPU6050\I2C_MPU6050.hcg - [RTI]
File Edit View Tools Window Help
Start Page TMS570LC4357ZWT PINMUX RTI GIO ESM SCI1 SCI2 SCI3 SCI4 LIN1 LIN2 MIBSPI1 MIBSPI2 MIBSPI3 MIBSPI4 MIBSPI5 SPI1 SPI2 SPI3
RTI1 General RTI1 Counter 0 RTI1 Counter 1 RTI1 Compare
RTI1 Compare
Compare 0 Period: 1000
Update Compare 0: 9375000
Compare 0: 9375000
Counter 0:
Counter 1:
9.375000000
Actual Period (ms):
1000.000
Compare 1 Period: 5.000
Update Compare 1: 46875
Compare 1: 50000
9.375000000
10.000000000
5.000
Compare 2 Period: 8.000
Update Compare 2: 75000
Compare 2: 80000
8.000
Compare 3 Period: 10.000
Update Compare 2: 93750
100000
10.000
Interrupt/DMA


HAL Code Generator - /home/jhb/workspace_v8/I2C_MPU6050\I2C_MPU6050.hcg - [TMS570LC4357ZWT]
File Edit View Tools Window Help
Start Page TMS570LC4357ZWT PINMUX RTI GIO ESM SCI1 SCI2 SCI3 SCI4 LIN1 LIN2 MIBSPI1 MIBSPI2 MIBSPI3 MIBSPI4 MIBSPI5 SPI1 SPI2 SPI3
General Driver Enable R5-MPU-PMU Interrupts VIM General VIM RAM VIM Channel 0-31 VIM Channel 32-63 VIM Channel 64-95 VIM Channel 96-127 RAM Flash GCM
Enable RTI driver
Enable GIO driver **
Enable SCI drivers
Enable SCI3 driver **
Enable SCI4 driver **
Enable LIN drivers
Enable LIN1 driver ** /
Enable SCI1 driver **
Enable LIN2 driver ** /
Enable SCI2 driver **
Enable MIBSPI drivers
Enable MIBSPI1 driver **
Enable SPI1 driver **
Enable MIBSPI2 driver **
Enable SPI2 driver **
Enable MIBSPI3 driver **
Enable SPI3 driver **
Enable MIBSPI4 driver **
Enable SPI4 driver **
Enable MIBSPI5 driver **
Enable SPI5 driver **
Enable CAN drivers
Enable CAN1 driver
Enable CAN2 driver
Enable CAN3 driver
Enable CAN4 driver **
Enable ADC drivers
Enable ADC1 driver **
Enable ADC2 driver **
Enable HET drivers
Enable HET1 driver **
Enable HET2 driver **
Enable I2C driver **
Enable I2C1 driver **
Enable I2C2 driver **
Enable EMAC driver **
Enable DCC driver
Enable EMIF driver **
Enable POM driver
Enable CRC driver
Enable CRC1 driver
Enable CRC2 driver
Enable EQEP driver
Enable EQEP1 driver **
Enable EQEP2 driver **
Enable ETPWM driver
Enable ECAP driver
Enable FEE driver
Device Explorer
TMS570LC4357ZWT
SYSTEM
HL_hal_stdtypes.h
HL_sys_common.h
HL_reg_system.h
HL_reg_flash.h
HL_reg_l2ramw.h
HL_reg_vim.h
HL_reg_pbist.h
HL_reg_stc.h
HL_reg_efc.h
HL_reg_pcr.h
HL_reg_pmm.h
HL_reg_dma.h
HL_reg_ccmr5.h
HL_sys_core.h
HL_system.h
HL_sys_vim.h
HL_sys_mpu.h
HL_sys_pmu.h
HL_sys_pcr.h
HL_sys_pmm.h
HL_sys_dma.h
HL_sys_core.asm
HL_sys_intvecs.asm
HL_sys_mpu.asm
HL_sys_pmu.asm
HL_sys_pcr.c
HL_sys_pmm.c
HL_sys_dma.c
HL_system.c
HL_sys_phantom.c
HL_sys_startup.c
HL_sys_vim.c
HL_sys_main.c
HL_notification.c
HL_sys_link.cmd
HL_reg_epc.h
HL_reg_nmpu.h
HL_reg_scm.h
HL_reg_sdcmmr.h
HL_epc.h
HL_epc.c
HL_nmpu.h
HL_nmpu.c
HL_errata.h
HL_errata.c
ESM
HL_reg_esm.h
HL_esm.h

I2C_MPU6050_PWM 숙제

```
#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 2
uint32 duty_arr[IDX] = { 1000,2000 };

int main(void)
{
    char txt_buf[256] = { 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();
    disp_set("i2cInit Success!!\n\r\0");
    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_COMPARE2);
_enable_IRQ_interrupt_();
rtiStartCounter(rtiREG1, rtiCOUNTER_BLOCK1);

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

etpwmInit();

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

etpwmStartTBCLK();
wait(1000000);

disp_set("ETPWM startTBCLK!!\n\r\0");

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);


        if(real_acc_x>0)
        {
            //tmp = sciReceiveByte(UART);
            rx_data = 0;

            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);
        }
```

```
            else if(real_acc_x <= 0)
            {
                rx_data = 1;

                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);

            }

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

숙제는 하였으나 가속도계를 누군가 빌려가서 확인은 하지 못하였다.