# 실시간성 보장 및 가독성 을 위한 Source 작성

# ❑ 목차

# ❖ 실시간성 보장 및 가독성을 위한 Soure 작성

## ◆ 약속된 Source 구조를 갖자!!!

- 자신만의 Source 구조를 갖자!
- 약속된 위치에 약속된 구문을!!
- 선언과 함께 초기값!!
- 주석을 습관화!!

- volatile ?

```
////////////////////////// Global Var ////////////////////
int a = 0;

volatile int b = 0;
```

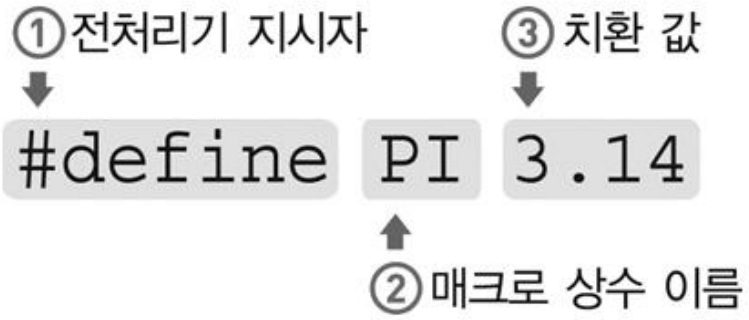
- m_temp / a_temp / p_temp ?
  - m_temp : Global Variable
  - a_temp : Local Variable
  - p_temp : Function Input Variable

```
////////////////////////////// Include ////////////////////////////
#include <avr/io.h>
#include <avr/interrupt.h>

#include "MCU_Init.h"
////////////////////////////// Define ////////////////////////////
#define     deg2rad     0.0174533
#define     rad2deg     57.2958

#define     sbi(PORTX,BitX) PORTX|= (1<<BitX)   //비트 set   명령 정의
#define     cbi(PORTX,BitX) PORTX&=~(1<<BitX)   //비트 clear 명령 정의
////////////////////////////// Function ////////////////////////////
int sum(int p_temp1, int p_temp2);
////////////////////////////// Global Var ////////////////////////////
MCU_Init mcu_init;

            int m_temp1 = 0;
volatile    int m_temp2 = 0;
////////////////////////////// Interrupt ////////////////////////////
SIGNAL(TIMER2_OVF_vect) // 10ms
{
    // Control Period Set
    TCNT2 = 0;
}

////////////////////////////// Main ////////////////////////////
int main(void)
{
    ////////////////////////////// Local Var ////////////////////
    int a_temp1 = 0;
    ////////////////////////////// Reg Init ////////////////////
    mcu_init.Init_DDR   ();
    mcu_init.Init_Timer2();
    mcu_init.Init_UART  ();
    mcu_init.Init_ADC   ();
    ////////////////////////////// Sys Init ////////////////////
    while(1)
    {
        //TODO:: Please write your application code
        //가능한 While문 내부는 적게
    }
}

////////////////////////////// Function ////////////////////////////
int sum(int p_temp1, int p_temp2)
{
    int ret = p_temp1 + p_temp2;;
    return ret;
}
```

# ❖ 실시간성 보장 및 가독성을 위한 Soure 작성

## ◆ Macro 상수 및 함수를 활용하자!!!

### • Macro 상수




```
// Input
DDRA     = 0x00; // ADC
cbi(DDRD,DDD0); // UART_RXD
// Output
DDRC     = 0xFF; // Test LED
sbi(DDRD,DDD7); // Control Period
sbi(DDRD,DDD1); // UART_TXD
```

```
#define deg2rad 0.0174533
#define rad2deg 57.2958
```

### • Macro 함수

```
#define sbi(PORTX,BitX) PORTX |= (1<<BitX)          // Set Bit
#define cbi(PORTX,BitX) PORTX &= ~(1<<BitX)         // Clear Bit

#define MIN(a,b)        ( a < b ) ? ( a ) : ( b ) ) // 비교매크로(작은값검출)
#define MAX(a,b)        ( ( a > b ) ? ( a ) : ( b ) ) // 비교매크로(큰값검출)
```

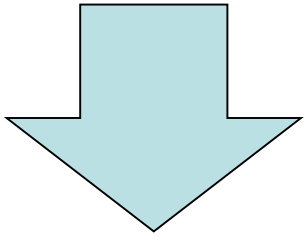
# ❖ 실시간성 보장 및 가독성을 위한 Soure 작성

◆ Init 구문의 가독성을 높이자!!!

- Shift 연산자를 활용하자!!!

```
UCSRB = 0x08;
UCSRC = 0x06;
UBRRH = 0;
UBRRL  = 8;
```

?



```
UCSRB     = (0<<RXCIE)|(0<<TXCIE)|(0<<UDRIE)|(0<<RXEN) |(1<<TXEN) |(0<<UCSZ2)|(0<<RXB8) |(0<<TXB8);
UCSRC     = (0<<URSEL)|(0<<UMSEL)|(0<<UPM1) |(0<<UPM0) |(0<<USBS) |(1<<UCSZ1)|(1<<UCSZ0)|(0<<UCPOL);
UBRRH     = 0;
UBRRL     = 8;
```

# ❖ 실시간성 보장 및 가독성을 위한 Soure 작성

## ◆ Init 구문의 가독성을 높이자!!!

- 함수를 활용하자!!!

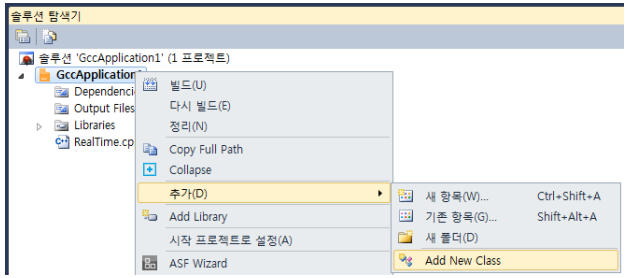
```
////////////////////// main ////////////////////////
int main(void)
{
    ////////////////////// Local Variable ////////////
    int e = 0;
    ////////////////////// Reg Init /////////////////
    // Io Init
    //Key Pad
    sbi(DDRF,1);    //row : 1
    sbi(DDRF,2);    //row : 2
    sbi(DDRF,3);    //row : 3
    sbi(DDRB,4);    //row : 4
    cbi(DDRE,5);    //col : 1
    cbi(DDRE,6);    //col : 2
    cbi(DDRE,7);    //col : 3
    sbi(PORTE,5);   //col : 1
    sbi(PORTE,6);   //col : 2
    sbi(PORTE,7);   //col : 3

    // External Int
    EICRB   = (1<<ISC71) |(0<<ISC70) |(1<<ISC61) |(0<<ISC60) |(1<<ISC51) |(0<<ISC50) |(0<<ISC41) |(0<<ISC40);
    EIMSK   = (1<<INT7 ) |(1<<INT6 ) |(1<<INT5 ) |(0<<INT4 ) |(0<<INT3 ) |(0<<INT2 ) |(0<<INT1 ) |(0<<INT0 );
    // Uart
    UCSR0B  = (0<<RXCIE0)|(0<<TXCIE0)|(0<<UDRIE0)|(1<<RXEN0) |(1<<TXEN0) |(0<<UCSZ02)|(0<<RXB80) |(0<<TXB80);
    UCSR0C  =             (0<<UMSEL0)|(0<<UPM01) |(0<<UPM00) |(0<<USBS0) |(1<<UCSZ01)|(1<<UCSZ00)|(0<<UCPOL0);
    UBRR0H  = 0;
    UBRR0L  = 8;//  115200BPS
    // Timer
    TCCR1A  = (0<<COM1A1)|(0<<COM1A0)|(0<<COM1B1)|(0<<COM1B0)|(0<<COM1C1)|(0<<COM1C0)|(1<<WGM11) |(0<<WGM10);
    TCCR1B  = (0<<ICNC1) |(0<<ICES1) |           (1<<WGM13) |(1<<WGM12) |(0<<CS12)  |(1<<CS11)  |(0<<CS10) ;
    ICR1    = 20000;

    TCCR3A  = (1<<COM3A1)|(0<<COM3A0)|(0<<COM3B1)|(0<<COM3B0)|(0<<COM3C1)|(0<<COM3C0)|(1<<WGM31) |(0<<WGM30);
    TCCR3B  = (0<<ICNC3) |(0<<ICES3) |           (1<<WGM33) |(1<<WGM32) |(0<<CS32)  |(1<<CS31)  |(0<<CS30) ;
    ICR3    = 40000;

    TIMSK   = (0<<OCIE2) |(0<<TOIE2) |(0<<TICIE1)|(0<<OCIE1A)|(0<<OCIE1B)|(1<<TOIE1) |(0<<OCIE0) |(0<<TOIE0);
    ETIMSK  = (0<<TICIE3)|(0<<OCIE3A)|(0<<OCIE3B)|(0<<TOIE3) |(0<<OCIE3C)|(0<<OCIE1C);
    // ADC
    ADCSRA  = (1<<ADEN) |(0<<ADSC) |(0<<ADFR)   |(0<<ADIF)|(0<<ADIE)|(1<<ADPS2) |(1<<ADPS1) |(1<<ADPS0);
    ADMUX   = (0<<REFS1)|(1<<REFS0)|(0<<ADLAR)  |(0<<MUX4)|(0<<MUX3)|(0<<MUX2)  |(0<<MUX1)  |(0<<MUX0);
    // Global Intterupt Enable
    SREG    |= 0x80;
    ////////////////////// Sys Init /////////////////
    PORTC   = 0x00;

    while(1)
    {
        // 가능한 While문 내부는 적게
    }
    return;
}
```

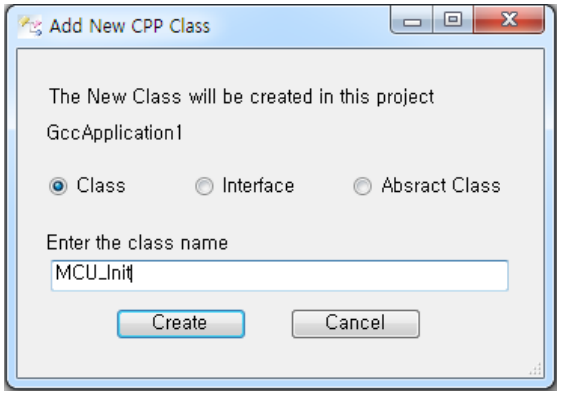
```
void UART_init(void)     // 8bit, non_parity, 9600bps
{
    UCSRB  = (0<<RXCIE)|(0<<TXCIE)|(0<<UDRIE)|(0<<RXEN) |(1<<TXEN) |(0<<UCSZ2)|(0<<RXB8) |(0<<TXB8);
    UCSRC  = (0<<URSEL)|(0<<UMSEL)|(0<<UPM1) |(0<<UPM0) |(0<<USBS) |(1<<UCSZ1)|(1<<UCSZ0)|(0<<UCPOL);
    UBRRH  = 0;
    UBRRL  = 103;
}
```

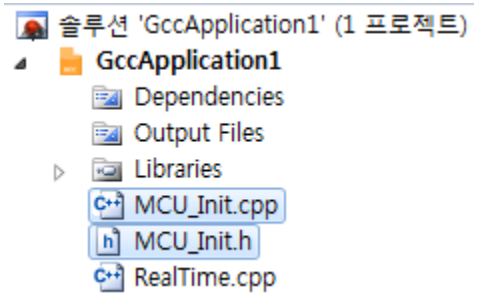
# ❖ 실시간성 보장 및 가독성을 위한 Soure 작성

## ◆ Class를 활용하자!!!

- Class를 생성 하자




- **MCU_Init.CPP**

```
#include "MCU_Init.h"

// default constructor
MCU_Init::MCU_Init()
{
} //MCU_Init

// default destructor
MCU_Init::~MCU_Init()
{
} //~MCU_Init
```

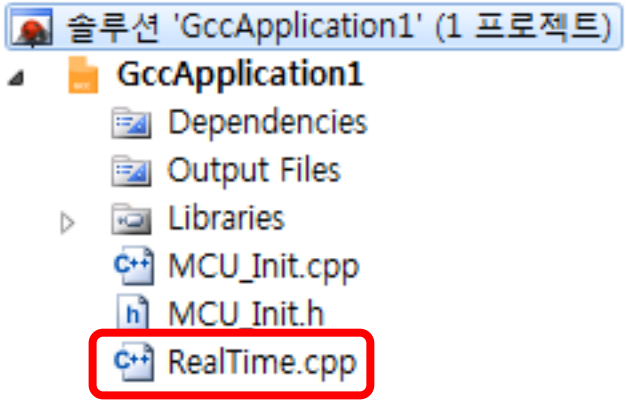
- **MCU_Init.h**

```
#ifndef __MCU_INIT_H__
#define __MCU_INIT_H__


class MCU_Init
{
//variables
public:
protected:
private:

//functions
public:
    MCU_Init();
    ~MCU_Init();
protected:
private:
    MCU_Init( const MCU_Init &c );
    MCU_Init& operator=( const MCU_Init &c );

}; //MCU_Init

#endif //__MCU_INIT_H__
```

- public : 모든 팩키지에서 이 클래스를 참조할수 있다.
- private : 자신을 포함한 클래스에서만 참조가능
- protected : 자신을 포함하는 클래스에서 상속받은 클래스에서도 참조가능

# ❖ 실시간성 보장 및 가독성을 위한 Soure 작성

## ◆ Class를 활용하자!!!



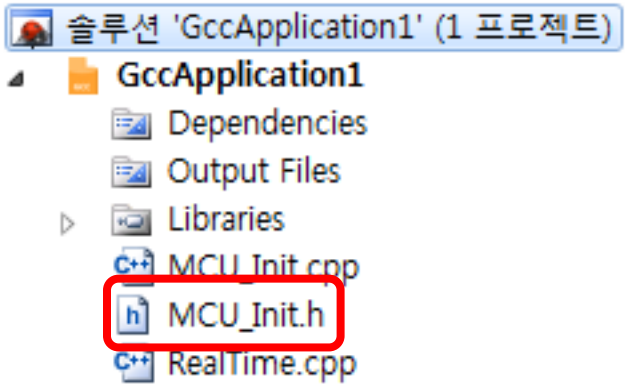

```
//////////////////////////// Include ////////////////////////////
#include <avr/io.h>
#include <avr/interrupt.h>

#include "MCU_Init.h"
//////////////////////////// Define ////////////////////////////
#define     deg2rad     0.0174533
#define     rad2deg     57.2958

#define     sbi(PORTX,BitX) PORTX|= (1<<BitX)   //비트 set  명령 정의
#define     cbi(PORTX,BitX) PORTX&=~(1<<BitX)   //비트 clear 명령 정의
//////////////////////////// Function ////////////////////////////

//////////////////////////// Global Var ////////////////////////////
MCU_Init mcu_init;
//////////////////////////// Interrupt ////////////////////////////
SIGNAL(TIMER2_OVF_vect) // 10ms
{
    // Local Variable
    unsigned int a_ADC_data[4]  = {0,};
    // ADC Read
    for (int i=0;i<4;i++)
        a_ADC_data[i] = mcu_init.ADC_Read(i);
    // Uart Send
    mcu_init.Uart_num(a_ADC_data[0]);   mcu_init.Uart_Putch('₩t');
    mcu_init.Uart_num(a_ADC_data[1]);   mcu_init.Uart_Putch('₩t');
    mcu_init.Uart_num(a_ADC_data[2]);   mcu_init.Uart_Putch('₩t');
    mcu_init.Uart_num(a_ADC_data[3]);   mcu_init.Uart_Putch('₩t');
    mcu_init.Uart_Putch('₩n');
    mcu_init.Uart_Putch('₩r');
    // Control Period Set
    TCNT2 = 156;
}

//////////////////////////// Main ////////////////////////////
int main(void)
{
    //////////////////////////// Local Var ////////////////////

    //////////////////////////// Reg Init ////////////////////
    mcu_init.Init_DDR   ();
    mcu_init.Init_Timer2();
    mcu_init.Init_TIMSK ();
    mcu_init.Init_UART  ();
    mcu_init.Init_ADC   ();
    sei();
    //////////////////////////// Sys Init ////////////////////
    PORTC = 0xFF;
    //////////////////////////// While  ////////////////////
    while(1)
    {
        //TODO:: Please write your application code
        //가능한 While문 내부는 적게
    }
}

//////////////////////////// Function ////////////////////////////
```

Class Include

Class 생성

# ❖ 실시간성 보장 및 가독성을 위한 Soure 작성

## ◆ Class를 활용하자!!!




```
#include <avr/io.h>

#ifndef __MCU_INIT_H__
#define __MCU_INIT_H__


class MCU_Init
{
//variables
public:
protected:
private:

//functions
public:
    MCU_Init();
    ~MCU_Init();
    //  IO
    void            Init_DDR        (void);
    //  EXT Int
    void            Init_ExtInt     (void);
    //  Timer
    void            Init_Timer0     (void);
    void            Init_Timer1     (void);
    void            Init_Timer2     (void);
    void            Init_TIMSK      (void);
    //  UART
    void            Init_UART       (void);
    void            Uart_Putch      (unsigned char  data);
    unsigned char   Uart_Getch      (void);
    void            Uart_num        (int            data);
    //  ADC
    void            Init_ADC        (void);
    unsigned int    ADC_Read        (unsigned char input);
protected:
private:
    MCU_Init( const MCU_Init &c );
    MCU_Init& operator=( const MCU_Init &c );

}; //MCU_Init

#endif //__MCU_INIT_H__
```

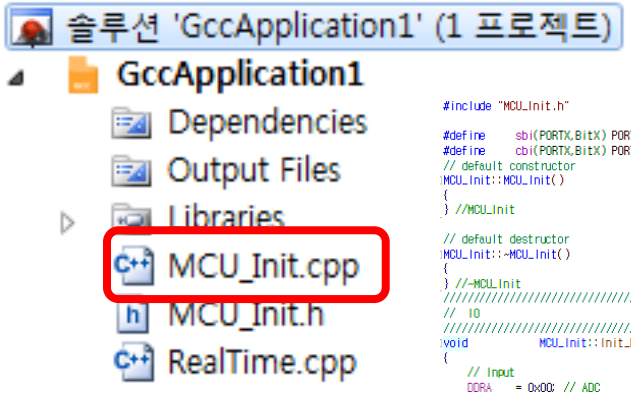
생성자

소멸자

함수 선언

# ❖ 실시간성 보장 및 가독성을 위한 Soure 작성

## ◆ Class를 활용하자!!!




```
#include "MCU_Init.h"

#define     sbi(PORTX,BitX) PORTX|= (1<<BitX)   //비트 set   명령 정의
#define     cbi(PORTX,BitX) PORTX&=~(1<<BitX)   //비트 clear 명령 정의
// default constructor
MCU_Init::MCU_Init()
{
} //MCU_Init

// default destructor
MCU_Init::~MCU_Init()
{
} //~MCU_Init
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//  IO
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void            MCU_Init::Init_DDR          (void)
{
    // Input
    DDRA    = 0x00; // ADC
    cbi(DDRD,DDD0); // UART_RXD
    // Output
    DDRC    = 0xFF; // Test LED
    sbi(DDRD,DDD7); // Control Period
    sbi(DDRD,DDD1); // UART_TXD
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//  EXT Int
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void            MCU_Init::Init_ExtInt       (void)
{
    MCUCR   = (1<<ISC01)|(0<<ISC00); // INT0를 처음엔 falling edge에서 발생하도록 설정
    GICR    = (0<<INT1)|(1<<INT0);  // INT0를 enable
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//  Timer
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void            MCU_Init::Init_Timer0       (void)
{
    TCCR0   = (1<<WGM00)|(0<<COM01)|(0<<COM00)|(1<<WGM01)|(0<<CS02)|(1<<CS01)|(0<<CS00); // fast pwm, 8분주
    TCNT0   = 0;
    OCR0    = 0;
}
void            MCU_Init::Init_Timer1       (void)
{
    TCCR1A  = (0<<COM1A1)|(0<<COM1A0)|(0<<COM1B1)|(0<<COM1B0)|(0<<FOC1A)|(0<<FOC1B)|(1<<WGM11)|(0<<WGM10);
    TCCR1B  = (0<<ICNC1) |(0<<ICES1) |(1<<WGM13) |(1<<WGM12) |(0<<CS12) |(1<<CS11) |(1<<CS10); // 64분주 타이머
    TCNT1   = 0;                    // TCNT1이 0부터 증가해서 ICR1 값 만나면 Overflow Interrupt
    ICR1    = 2500;                 // for 20ms FCPU가 8MHz 분주비 64 => 2500*64/8000000 = 0.02
}
void            MCU_Init::Init_Timer2       (void)
{
    TCCR2   = (1<<WGM20)|(0<<COM21)|(0<<COM20)|(1<<WGM21)|(1<<CS22)|(0<<CS21)|(1<<CS20); // Fast pwm, 1024분주
    TCNT2   = 156;
    OCR2    = 0;
}
void            MCU_Init::Init_TIMSK        (void)
{
    TIMSK  |= (1<<TOIE2);           // Timer Interrupt 1 Enable
}
```

```
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//  UART
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void            MCU_Init::Init_UART         (void)
{
    // UCSRA = 0x00;
    UCSRB   = (1<<RXEN)|(1<<TXEN);  // RX, TX Enable
    UCSRC   = 0x06;                 // Stop bit 1, Data size 8

    UBRRH   = 0x00;
    UBRRL   = 8;                    // 115200bps
}
void            MCU_Init::Uart_Putch        (unsigned char  data)
{
   while( !( UCSRA & (1<<UDRE)) );  // UDR empty flag

   UDR = data;
}
unsigned char   MCU_Init::Uart_Getch        (void)
{
   while ( !(UCSRA & (1<<RXC)) );   // RX complete flage

   return UDR;
}
void            MCU_Init::Uart_num          (int            data)
{
    Uart_Putch((data/1000)+48);
    Uart_Putch((data%1000)/100 +48);
    Uart_Putch((data%100) /10+48);
    Uart_Putch((data%10) +48);
}
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
//  ADC
////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////////
void            MCU_Init::Init_ADC          (void)
{
    ADMUX   = (0<<REFS1)|(1<<REFS0);    // AVCC 전압 reference
    ADCSRA  = (1<<ADEN) |(7<<ADPS0);    // ADC enable, 128 prescaler
}
unsigned int    MCU_Init::ADC_Read          (unsigned char input)
{
    ADMUX   = (input|(0<<REFS1)|(1<<REFS0));
    ADCSRA |= (1<<ADSC);
    while(!(ADCSRA & (1<<ADIF)));
    return ADC;
}
```