

```
#define F_CPU 8000000UL
```

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

```
#include <util/delay.h>
```

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

```
#define MAX_TAM 20 // "1024" + "|" + "1024" + "|" + "1024" + "|" + "3000" + NULL
```

```
#define NUM_DATA 4 // 3 sensores + potenciómetro
```

```
uint16_t Data[NUM_DATA]; //Datos provenientes de los sensores
```

```
uint8_t Cadena[MAX_TAM];
```

```
uint16_t Contador = 0; // Contador de las señales de pulsos del contador
```

```
uint16_t Frecuencia = 0;
```

```
uint16_t Pot = 0;
```

```
//uint16_t RPM;
```

```
void ConfigurarADC(){
```

```
    ADMUX = (1<<REFS0); // Vref = AVCC = 5V
```

```
    ADCSRA = (1<<ADEN) | (1<<ADPS2) | (1<<ADPS1); // Fadc=125kHz
```

```
    //ADEN: Prende el ADC
```

```
    //ADIE: Activa las interrupciones de fin de conversión
```

```
    //ADPS: prescalador 16 : fadc = 125kHz
```

```
    //ADLAR: 8bits (ADCH)
```

```
    //REFS: Vref = AVCC = 5V -> 1LSB = 5mV aprox
```

```
}
```

```
uint16_t ADCRead(uint8_t canal) {
```

```

// Seleccionamos el canal modificando los registros MUX0 MUX1 y MUX2.
ADMUX = (ADMUX & 0xF8) | (canal & 0x07);

//Iniciamos la conversión
ADCSRA |= (1<<ADSC);

// Esperamos a que termine la conversión.
while(ADCSRA & (1<<ADSC));

// Regresamos el valor de la conversión.
return ADC;
}

```

```

void conf_usart(void){ // 38.4Kbps, 8bits, 1 bit de parada, sin bit de paridad
    UBRRL = 25;
    UCSRA = (1<<U2X);
    UCSRB = (1<<TXEN)|(1<<RXEN);
}

```

```

void usart_txcar(uint8_t c){
    while (!(UCSRA & (1<<UDRE)));
    UDR = c;
    //_delay_ms(10);
    //_delay_ms(1);
    //_delay_us(1);
}

```

```

void usart_txmens(uint8_t* Mensaje){
    uint8_t c,i=0;
    c = Mensaje[i];
    while (c != 0){
        if (c != 32){
            usart_txcar(c);
        }
        i++;
        c = Mensaje[i];
    }
    usart_txcar(13);
    usart_txcar(10);
}

```

```

void ConfigurarTimer1(void){ //Para PWM
    TCCR1A = (1<<WGM11) | (1<<WGM10); //PWM no invertido de 10 bits
    TCCR1B = (1<<CS10) | (1<<WGM12); //preescalador 1, PWM de 7.8k
    OCR1A = 0;
}

```

```

void ConfigurarTimer2(void){ //Para Muestreo
    TCCR2 = (1<<WGM21) | (1<<CS22) | (1<<CS21) | (1<<CS20); //preescalador 1024, Modo
    CTC
    OCR2 = 77; // Aproximadamente 100 Hz
}

```

```

void ConfigurarIntExt(void){
    MCUCR |= (1<<ISC01) | (1<<ISC00); //Interrupción en flanco de subida
}

```

```
void NumerosCadena(uint16_t Datos[], uint8_t Cadena[]){ //Convierte la data a una cadena de texto
```

```
    uint8_t i,j=MAX_TAM,digito;
```

```
    for (i=0; i<MAX_TAM; i++){
```

```
        Cadena[i]= 32; //espacios en blanco
```

```
    }
```

```
    Cadena[--j]=0;
```

```
    for (i=0; i<NUM_DATA; i++){
```

```
        if (Datos[i] == 0) {
```

```
            Cadena[--j] = 48;
```

```
        }
```

```
        while (Datos[i]) {
```

```
            digito = Datos[i] % 10; //ultimo digito
```

```
            Cadena[--j] = (digito + 48);
```

```
            Datos[i] /= 10;
```

```
        }
```

```
        if (i < (NUM_DATA - 1)){
```

```
            Cadena[--j] = ','; //delimitador
```

```
        }
```

```
    }
```

```
}
```

```

int main(void)
{
    DDRB |= (1<<PINB1); //PWM (OC1A)
    PORTB |= (1<<PINB1);

    DDRB &= ~(1<<PINB0); //Pulsador de inicio
    DDRD &= ~(1<<PIND2) | ~(1<<PIND0); //Interrupción Externa y RX
    DDRD |= (1<<PIND1); //TX
    DDRC = 0x00; //Todas entradas (ADC)
    //DDRD |= (1<<PIND3) | (1<<PIND4);

    conf_usart();
    ConfigurarADC();
    ConfigurarTimer1();
    ConfigurarTimer2();
    ConfigurarIntExt();

    while (!(PINB & (1<<PINB0)));
    _delay_ms(10);
    while (PINB & (1<<PINB0));
    _delay_ms(10);

    TCCR1A |= (1<<COM1A1); //Activa PWM
    TIMSK |= (1<<OCIE2); // OCR2 generará interrupciones a 100Hz
    GICR |= (1<<INT0); // Activa las interrupciones externas

    sei();
}

```

```

while(1){ //Continuamente espera datos disponibles para enviar
    while(TCCR0==0); //Salta cuando TIMER2 le pasa el control
    NumerosCadena(Data,Cadena);
    usart_txmens(Cadena);
    Pot = ADCRead(3);
    OCR1A = Pot;
    Frecuencia = 0;
    TCCR0 = 0;
    //PORTD &= ~(1<<PIND4);
}
}

```

```

ISR(TIMER2_COMP_vect){ //Se ejecuta a 100Hz

```

```

    //PORTD |= (1<<PIND3);

    Frecuencia = Contador;
    Data[0] = Frecuencia;
    Data[1] = ADCRead(2);
    Data[2] = ADCRead(1);
    Data[3] = ADCRead(0);

    TCCR0 = 1;
    Contador = 0;
    //PORTD &= ~(1<<PIND3);

}

```

```
ISR(INT0_vect){  
    Contador++;  
}
```