```
#define F_CPU 8000000UL          //Frequenz des Mikrocontroller festlegen (8MHz)

#include <avr/io.h>

#include <util/delay.h>              //delay Funktion einbinden

#include <avr/interrupt.h>       //interrupt Funktion einbinden

#define  Anzahl_Servo  2       //Anzahl der Servomotoren definieren

#define  Mitte  188              //Zählerstand für die Mittelstellung definieren. 188 = 1,5ms

uint8_t  i = 0;                  /Variable festlegen, um später zwischen den Servos zu wechseln

unsigned Vergleichswert[Anzahl_Servo];  //Array für die Vergleichswerte erstellen

uint8_t Servo_Pin[Anzahl_Servo] = { (1<<PB1), (1<<PB2) }; //Array für die Pins festlegen an dem
die PWM Leitung der Servos verbunden sind

void Servo_Einstellung()  // Initialisierung der Servos und des Timer2
{
        DDRB |= (1<<PB1) | (1<<PB2);  //Servo Pins werden als Ausgang festgelegt

        Vergleichswert[0] = Mitte;     //Der Wert für die Mittelstellung wird als erster
                                         Vergleichswert im  Array übernommen (Servo_1)
        Vergleichswert[1] = Mitte;     //Der Wert für die Mittelstellung wird als erster
                                         Vergleichswert im  Array übernommen (Servo_2)

        //Initialisierung des Timer2

        TIMSK  |= (1<<OCIE2);              //Timer/Counter2 Compare Match Interrupt aktiviert
        TCCR2  |= (1<<WGM21) | (1<<CS22);  //CTC Timer wird ausgewählt. Ein Vorteiler von
                                              64  wird festgelegt.

        OCR2 = Vergleichswert[0];     //Der Vergleichswert von dem ersten Servo wird als
                                        Obergrenze für das Register OCR2 festgelegt
        TCNT2 = 0;              //Zähler startet bei 0
}
int main(void) // Hauptprogramm
{
        //Erstellen von Variablen in den die ADC Werte gespeichert werden
        uint8_t Sensor_Wert_1 = 0;   // pin pc0
        uint8_t Sensor_Wert_2 = 0;   // pin pc1
        uint8_t Sensor_Wert_3 = 0;   // pin pc2
        uint8_t Sensor_Wert_4 = 0;   // pin pc3

        //Initialisierung des ADC-Wandler

        ADCSRA  |=  (1<<ADPS2)  |  (1<<ADPS0)  |  (1<<ADSC)  |  (1<<ADEN);     //(1<<ADPS2),
        (1<<ADPS0) = Prescaler von 32; (1<<ADSC), (1<<ADEN) = Aktivierung des ADC-Wandler
        ADMUX  |= (1<<REFS0) | (1<<ADLAR);  //(1<<REFS0) = interne Referenzspannung;
                (1<<ADLAR) = linksbündig, die 8 höchstwertigen Bits werden in ADCH abgelegt.
```

```
        Servo_Einstellung();  //Servo Einstellungen werden für das Hauptprogramm übernommen


        sei();  // globale Interruptfreigabe

        _delay_ms(500);       //Die Servomotoren haben ausreichend Zeit sich  auf ihre
                              Mittelstellung auszurichten


while(1)        //Beginn der while Schleife in der die eigentliche Arbeit gemacht wird
        {

                while (ADCSRA & (1<<ADSC)) { };   //Die Schleife dient dazu, um zu warten bis
                    die Wandlung abgeschlossen ist, da der erste wert i.d.R nicht zu gebrauchen ist

                //pin 0
                ADCSRA |= (1<<ADSC);     //Im ADCSRA wird das ADSC wieder gesetzt, weil
                                         es nach jeder Wandlung automatisch mit 0 beschrieben wird
                while(ADCSRA & (1<<ADSC) ) { };  // warten bis die Wandlung abgeschlossen
                                                    ist

                Sensor_Wert_1 = ADCH;      // Der erste ADC Wert wird in die Variable
                                           Sensor_Wert_1 geladen


                //pin 1

                ADMUX  |= (1<<MUX0);    //Die Bit Belegung wird so weit verändert, dass nun
                                          die Werte an Pin 1 übernommen werden

                ADCSRA |= (1<<ADSC);              //ADC wird gestartet
                while (ADCSRA & (1<<ADSC) ) { };  //warten bis die Wandlung abgeschlossen
                                                    ist

                Sensor_Wert_2 = ADCH;      //Der zweite ADC Wert wird in die Variable
                                           Sensor_Wert_2 geladen


                // pin 2

                ADMUX = (ADMUX &= ~(1<<MUX0));   // Die Bit Belegung wird so weit
                                         verändert, dass nun die Werte an Pin 2 übernommen werden

                ADMUX  |= (1<<MUX1)  | (1<<REFS0) | (1<<ADLAR);
                ADCSRA |= (1<<ADSC);                  //ADC wird gestartet
                while (ADCSRA & (1<<ADSC) ) { };      //warten bis die Wandlung
                                                        abgeschlossen ist
```

```
Sensor_Wert_3 = ADCH;      //Der dritte ADC Wert wird in die Variable
                             Sensor_Wert_3 geladen


// pin 3

ADMUX  |= (1<<MUX0);    // Die Bit Belegung wird wieder so weit verändert,
                           dass nun die Werte an Pin 3 übernommen werden

ADCSRA |= (1<<ADSC);              //ADC wird gestartet
while (ADCSRA & (1<<ADSC) ) { };   //warten bis die Wandlung abgeschlossen
                                     ist

Sensor_Wert_4 = ADCH;      //Der vierte ADC Wert wird in die Variable
                             Sensor_Wert_4 geladen



if  ((Sensor_Wert_1 + Sensor_Wert_2) > (Sensor_Wert_3 + Sensor_Wert_4))
//Es wird untersucht, ob die addierten Sensorwerte der rechten Hälfte größer sind als die Addition der linken Hälfte
{
        if (Vergleichswert[0] < 249)    // Es wird nun überprüft, ob der
                                    Vergleichswert noch erhöht werden kann (250 = 2ms)
        {
                Vergleichswert[0]++;  //Vergleichswert für die x-Achse wird um
                                        ein erhöht (Servo steuert nach rechts)
        }
}

if ((Sensor_Wert_1 + Sensor_Wert_2) < (Sensor_Wert_3 + Sensor_Wert_4))
//Es wird untersucht, ob die addierten Sensorwerte der rechten Hälfte kleiner sind als die Addition der linken Hälfte
{
        if (Vergleichswert[0] > 124)   //Ist der Wert kleiner, so wird überprüft, ob
                der Vergleichswert um ein verringert werden kann (125 = 1ms)
        {
                Vergleichswert[0]--;   //Vergleichswert für die x-Achse wird um ein
                                         verringert (Servo steuert nach links)
        }
}

if  ((Sensor_Wert_1 + Sensor_Wert_3) > (Sensor_Wert_2 + Sensor_Wert_4))
//Es wird untersucht, ob die addierten Sensorwerte der oberen Hälfte größer sind als die Addition der unteren Hälfte
{
        if (Vergleichswert[1] < 249)  //Ist der Wert größer, so wird überprüft, ob
                                    der Vergleichswert um ein erhöht werden kann
        {
                Vergleichswert[1]++;   //Vergleichswert für die y-Achse wird um
                                         ein erhöht (Servo steuert nach oben)
        }
```

```
                }

                if  ((Sensor_Wert_1 + Sensor_Wert_3) < (Sensor_Wert_2 + Sensor_Wert_4))
//Es wird untersucht, ob die addierten Sensorwerte der unteren Hälfte größer sind als die Addition der unteren Hälfte
                {
                        if (Vergleichswert[1] > 124)  //Ist der Wert kleiner, so wird überprüft, ob
                                                      der Vergleichswert um ein erhöht werden kann
                        {
                                Vergleichswert[1]--;  //Vergleichswert für die y-Achse wird um ein
                                                        verringert (Servo steuert nach unten)
                        }
                }
        _delay_ms(20); //Der delay dient dazu, dass die Servos zeit haben sich auszurichten
        }
}


ISR (TIMER2_COMP_vect)  //Erreicht der Zähler (TCNT2) den maximal Wert im OCR2, so
                          sprint es in die ISR
{

        if (i < 1)        //Mit der if abfrage wird gesteuert welcher Servo ausgeschaltet wird (i = 0
                           ist Servo 1 und i = 1 ist Servo 2)
        {
                PORTB &= ~Servo_Pin[i];             //Servo 1 wird ausgeschaltet
                i++;                                //ES wird auf Servo 2 umgeschaltet
        }
        else
        {
                PORTB &= ~Servo_Pin[i];             //Servo 2 wird ausgeschaltet
                i--;                                //ES wird auf Servo 1 umgeschaltet
        }

                PORTB |= Servo_Pin[i];        // Die Ausgangsleitung des nächsten Servos ist 1
                OCR2 = Vergleichswert[i];    // Der Verleichswert des aktiven Servo wird als
                                               maximal Wert festgelegt
}
```