REV	DATA	ZMIANY
0.1	06.2019	Grzegorz Jurek, Jakub Marcinkowski

Obsługa sterowania silnikiem DC

Autorzy: Grzegorz Jurek, Jakub Marcinkowski

Elektronika

Wydział Informatyki, Elektroniki i Telekomunikacji

Akademia Górniczo-Hutnicza

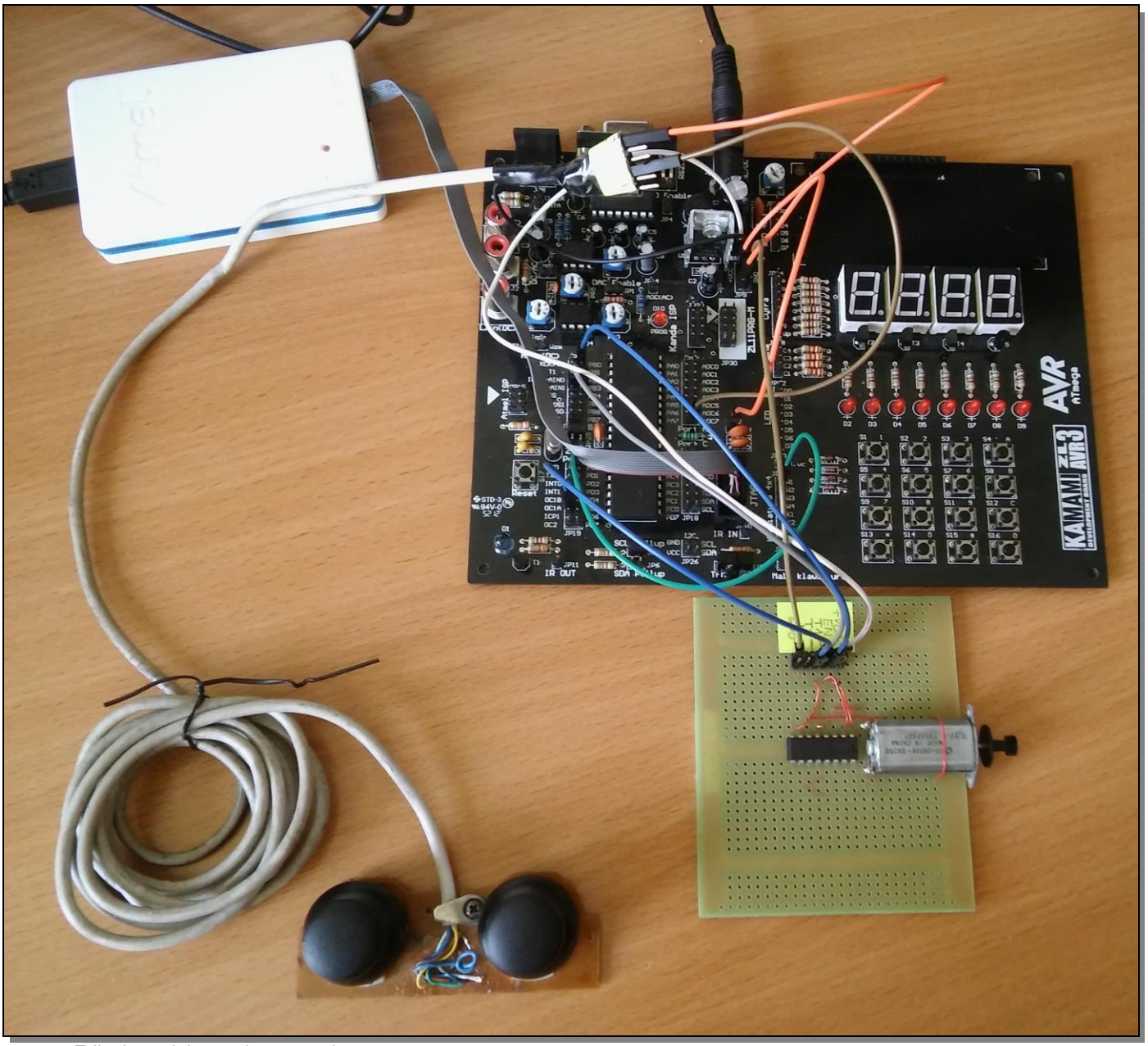
Kraków 2019

Spis treści

1. Wstęp

Silnik prądu stałego jest sterowany za pomocą joystick'a. Obroty silnika mogą być regulowane w zależności od pozycji. Obroty w lewo i prawo, joystick w pozycji neutralnej – silnik nie obraca się. Załączenie i odłączenie silnika przy pomocy przycisku.

Z joystick'a otrzymujemy napięcie z potencjometru z zakresu 0-5V, które trafia na przetwornik Analogowo Cyfrowy (ADC) do mikrokontrolera, zamieniane jest na 10-bitową postać cyfrową. W programie parametry l_pos i h_pos wyznaczają „martwy zakres" pracy joystick'a (silnik nie obraca się). Po przekroczeniu tej strefy silnik obraca się z prędkością regulowaną przez pozycję joysticka. Postać 10 bitową przesuwamy o dwa miejsca, aby pozbyć się LSB i dopasować do postaci 8-bitowej, która trafi na OCR0 lub OC1A. Użycie Pulse Width Modulation (PWM) reguluje prędkość obrotów.



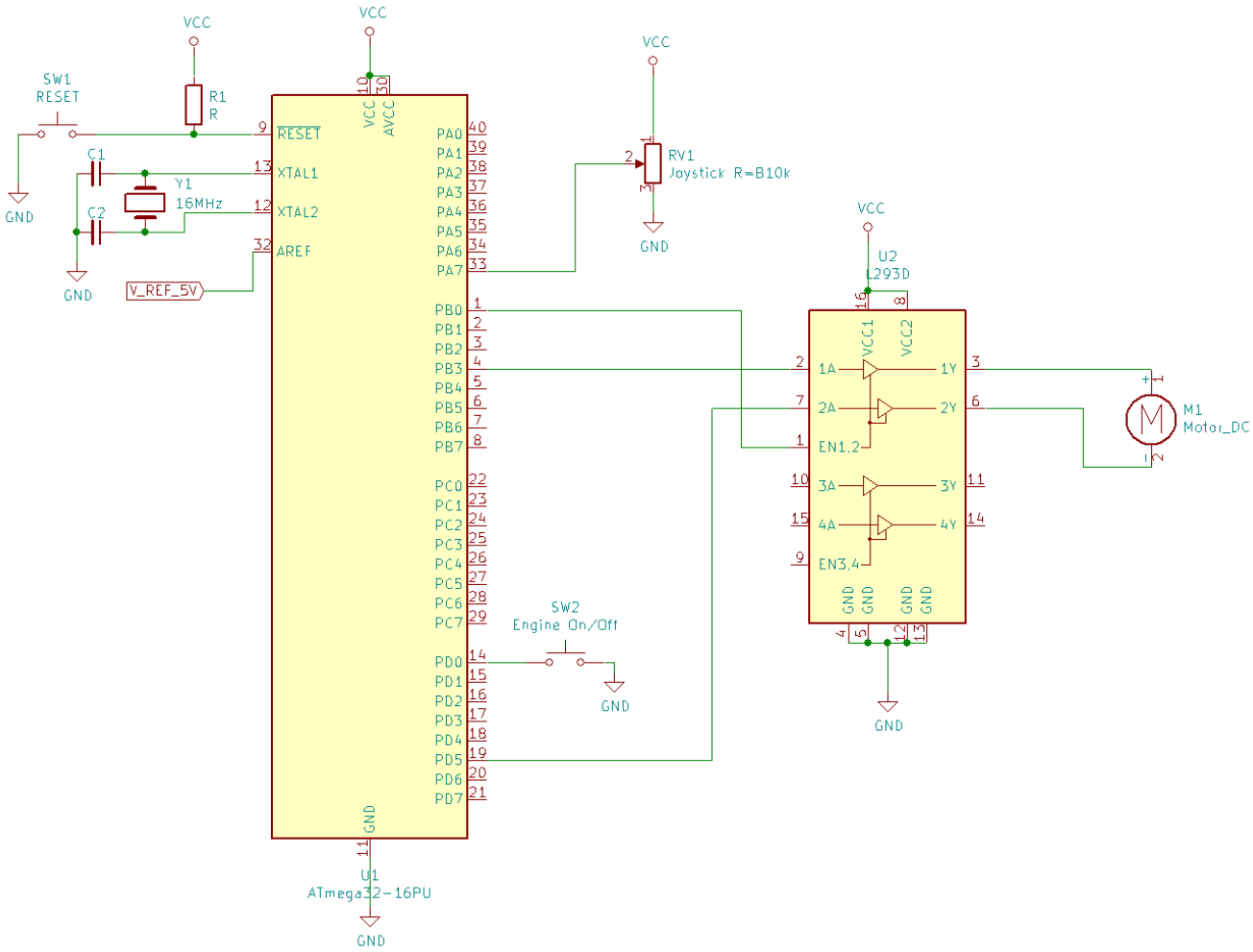
Rys 1.1 Zdjęcie projektu z elementami.

2. Wymagania sprzętowe

- Mikrokontroler - ATmega32
- Układ mostka H - L293D
- Silnik prądu stałego 9V
- Nastawnik – joystick z potencjometrem liniowym 10kΩ oraz z przyciskiem

3. Projekt techniczny

3.1 Schemat połączeń



Rys. 3.1 Schemat połączeń

3.2 Kod, implementacja PWM, ADC

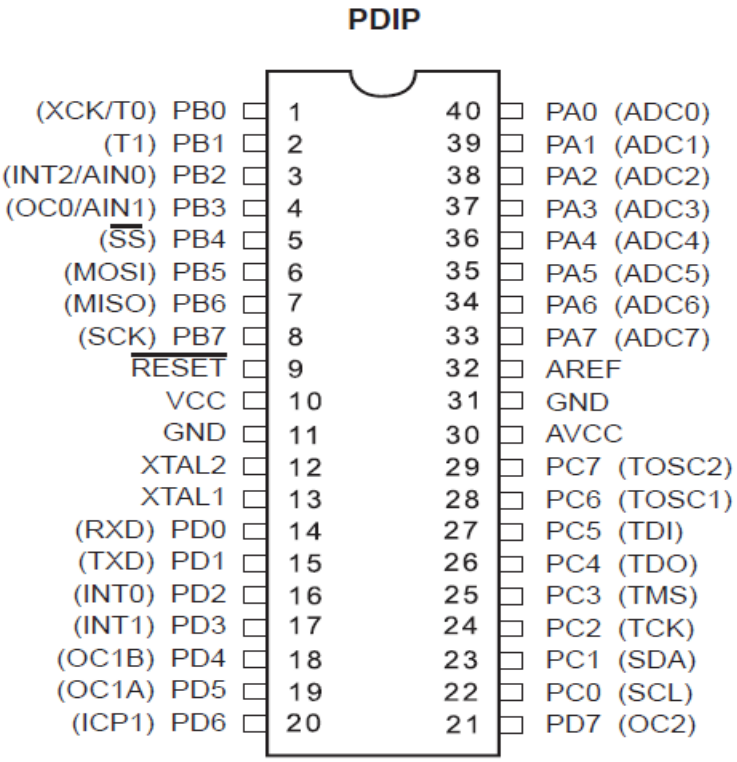
```
#define F_CPU 16000000
#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>




int main (void)
{

	//################## Konfiguracja PWM ##############################
	//Pierwszy kanal
	TCCR0 |= (1<<WGM01);// Tryb : Fast PWM 8bit
	TCCR0 |= (1<<WGM00);

	TCCR0 |= (1<<COM01)|(0<<COM00);//Clear OC0 on Compare Match
	TCCR0 |= (0<<CS02)|(1<<CS01)|(1<<CS00);// preskaler = 64
	/*
	000 No clock source (Timer/Counter stopped).
	001 clkI/O/(No prescaling )
	010 clkI/O/8 (From prescaler)
	011 clkI/O/64 (From prescaler)
	100 clkI/O/256 (From prescale)
	101 clkI/O/1024 (From pres)
	*/

	// datasheet, page 107!!!!!
	//Drugi kanal
	TCCR1A |= (1<<WGM10);// Tryb : Fast PWM 8bit
	TCCR1A |= (0<<WGM11);
	TCCR1B |= (1<<WGM12); // TCCR1B !!

	TCCR1A |= (1<<COM1A1)|(0<<COM1A0);//Clear OC1A on Compare Match
	TCCR1B |= (0<<CS12)|(1<<CS11)|(1<<CS10);// preksaler = 64


	/*
	wyjscie na OCR0 (PB3) lewo
	wyjscie na OC1A (PD5) prawo
	ENABLE aktywe H
	*/
```

```
//################## Przetwornik ADC ############################
      ADMUX |= (0<<REFS0) | (0<<REFS1); // VREF OFF from VCC // VREF JP12
      /*
      REFS1 REFS0 Voltage Reference Selection
      0 0 AREF, Internal Vref turned off
      0 1 AVCC with external capacitor at AREF pin
      1 0 Reserved
      1 1 Internal 2.56V Voltage Reference with external capacitor at AREF pin
    */

      ADMUX |= (1<<MUX2) | (1<<MUX1) | (1<<MUX0); // PA7 / ADC7 input

      ADCSRA |= (1<<ADEN); //enable ADC
      ADCSRA |= (1<<ADSC); // pojedyncza konwersja / 1-free running mode
      ADCSRA |= (1<<ADPS2) | (1<<ADPS1) | (1<<ADPS0);// dzielnik f,16MHz-> 128kHz

      ADCSRA |= (1<<ADLAR); // starsza do lewej


      //################## Konfiguracja portow ############################
      DDRB = 0xFF; // portB jako wyjscie dla PWM OC0(PB3), oraz
                   //(PB0) wyjście dla ENABLE (uklad L293D)
      DDRD = 0b11111110; // portD jako wyjscie dla PWM OC1A(PD5)
      PORTD = (1<<PIND0); //pullup

      DDRA = 0x00; //wejscie pod Przetwornik ADC (PA7)
      PORTA = 0XFF; //pull-up


//#### < L_pos, H_pos > = martwy zakres, silnik nie pracuje w tym przedziale#####
int h_pos = 600;//560;
int l_pos = 500;//540;
```

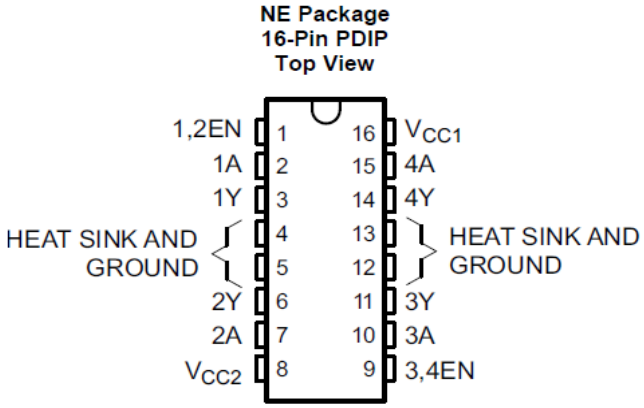
```
//################## Program ############################

      while(1)
      {
            if(~(PIND >> 0) & 0x01) //gdy przycisk PD0 == 1, zmien PB0 (on/off engine)
            {
                  _delay_ms(10);
                  if(~(PIND >> 0) & 0x01)
                  {
                        PORTB^=(1<<PINB0);
                        _delay_ms(50);
                  }
            }


            if ((ADC >= l_pos) && (ADC <= h_pos))
            {
                  //stop
                  OCR0 = 0;
                  OCR1A = 0;
            }

            else if (ADC >h_pos)
            {
                  //prawo
                  OCR0 = 0; //wylaczyc drugi kanal!
                  OCR1A = ADC >> 2;//ADC w trybie 10 bit, usuwamy LSB do postaci 8 bit
            }

            else if (ADC <l_pos)
            {
                  //lewo
                  OCR1A = 0; //wylaczyc drugi kanal!
                  OCR0 =  0xFF - (ADC >> 2);
            }

      }
}
```

4. Obudowy użytych układów

4.1 ATmega 32



4.2 L293D – mostek H



Pin Functions

PIN		TYPE	DESCRIPTION
NAME	NO.		
1,2EN	1	I	Enable driver channels 1 and 2 (active high input)
<1:4>A	2, 7, 10, 15	I	Driver inputs, noninverting
<1:4>Y	3, 6, 11, 14	O	Driver outputs
3,4EN	9	I	Enable driver channels 3 and 4 (active high input)
GROUND	4, 5, 12, 13	—	Device ground and heat sink pin. Connect to printed-circuit-board ground plane with multiple solid vias
V_{CC1}	16	—	5-V supply for internal logic translation
V_{CC2}	8	—	Power VCC for drivers 4.5 V to 36 V

5. Źródła

Programowanie w C

>> https://upel.agh.edu.pl/wiet/pluginfile.php/42089/mod_resource/content/2/AVR_programming_C.pdf

Atmega32

>> https://upel.agh.edu.pl/wiet/pluginfile.php/42092/mod_page/content/1/ATMega_ds.pdf

L293D

>> http://www.ti.com/lit/ds/symlink/l293.pdf