Ian Sbar

4-24-2024

Professor Muhn

RBT211

Final AVR Project

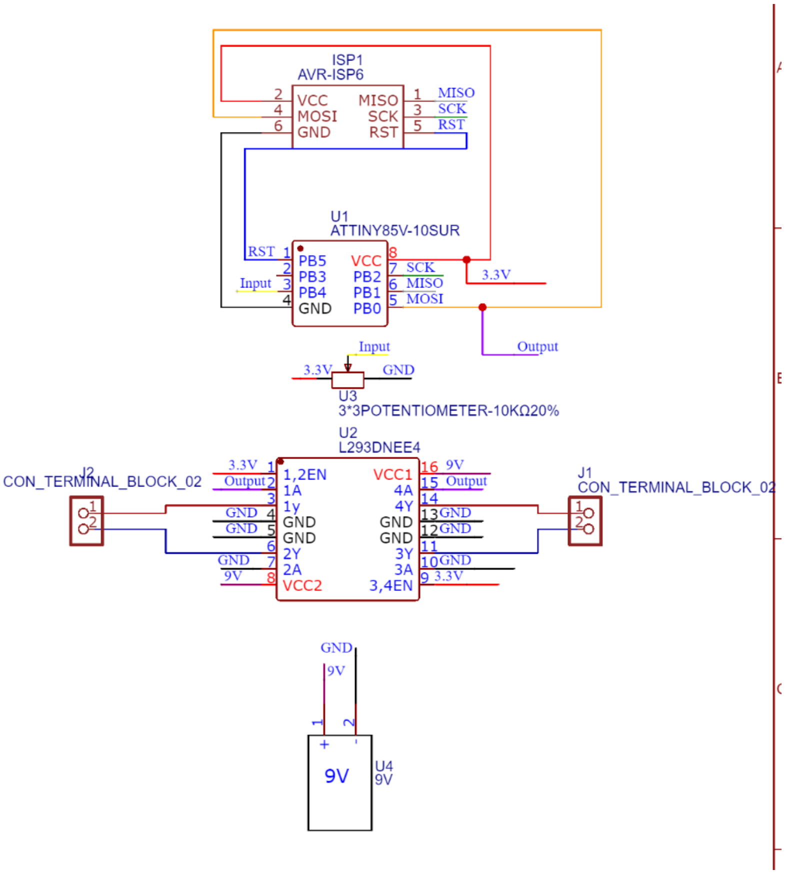
Introduction

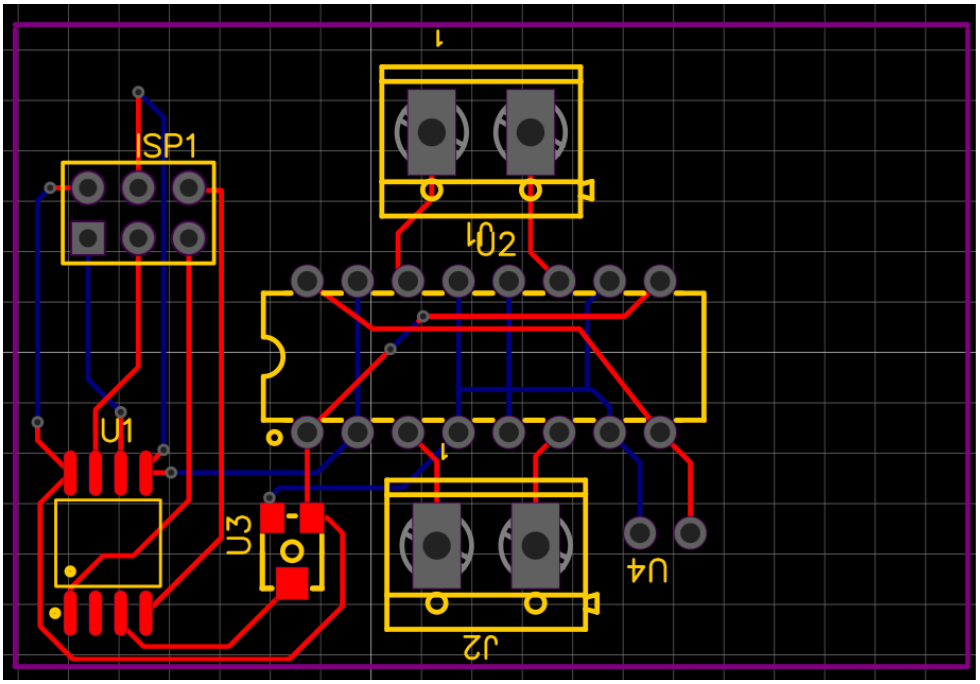
The idea for this project came from a weird problem I have when sleeping. Sometimes when I try to sleep, my feet begin to feel hot and I don't know why. This is a problem even when the room is naturally cold, and It makes me wish I had some sort of fan at the end of my bed to keep them cool.

That's when I decided to work on exactly that for this project… well, more like a very rough prototype of it. I used the code we did for the DC motor controller assignment, which originally just had an AtTiny85 connected to an L293D H-bridge, which had 1 motor connected to it. This project takes that code and adds ADC functionality with the potentiometer. The project also has the H-bridge wired up to support two DC motors.

Schematic

Create a schematic using your choice of schematic captures tool.





Bill of Materials (BOM).

- ISP programmer
- AtTiny85
- Potentiometer
- L293D (H-Bridge)
- 2 DC motors
- 9V battery

Describe what pins and signals are being used

Pin 3 on the AtTiny85 takes in the analog input from the potentiometer and converts it to a digital input that is sent through pin 5. This signal from the microcontroller then goes to pins 2 and 15 of the H-bridge to help make both motors spin.

Code

```
/*
 * RBT_Final.c
 *
 * Created: 4/21/2024 7:11:43 PM
 * Author : iansb
 */

#ifndef F_CPU
#define F_CPU 1000000UL
#endif

#include <avr/io.h>
#include <util/delay.h>
#include <avr/interrupt.h>
#include <stdio.h>

// int mode = 0;
// //This funtion uses a linear equation to calculate the PWM needed for motor speed
between 0% and 100%
// uint8_t Motor_linear(uint8_t speed ){
//     uint8_t pwm;
//
//     pwm = ((9/5)*speed)+75;
//     return pwm;
// }

int main(void)
{
       ADMUX|= (0 << REFS2) | (0 << REFS1 ) | (0 << REFS0 ) | (1 << ADLAR) | (0 << MUX3 )
| (0 << MUX2 ) | (1 << MUX1 ) | (0 << MUX0 ); //REFS pins are voltage reference selection
bits. seting (REFS1, REFS0) as (0,0)
       // ADLAR- left adjust bit
       // MUX pins select which ADC pin to use. (0010) means we are suing ADC2, which is
PB4 (pin 3). this pin is connected to the potentiometer

       ADCSRA |= (1 << ADEN ) | (0 << ADSC ) |(0 << ADATE) |(0 << ADIF ) |(0 << ADIE )
|(1 << ADPS2 ) |(1 << ADPS1 ) |(0 << ADPS0 ); // ADEN: ADC enable
       // ADSC: starts conversions in setup code if enabled. look at page 136 on
datasheet for more info
```

```
       //    ADATE: Auto trigger enable. trigger source selected in ADCSRB
       //    ADIF: intterupt flag. makes it run in an interupt block
       //  ADPS bits: Prescaler select bits. determines the prescaler division factor
between system clock frequecy and input. minimum divion is 2 (0,0,0)

       ADCSRB |= (0 << ADTS2 ) | ( 0 << ADTS1) | (0 << ADTS0 ); //ADTS = auto triger
source: (0,0,0) puts it in free running mode



       TCCR0A |= (1 << COM0A1) | (0 << COM0A0) |  (1 << WGM01) | (1 << WGM00);  //WGM02,
WGM01, WGM00 are set to (0,1,1) to enable fast PWM mode
                                                                 // The COM ports (1,0)



       TCCR0B |=(0 << CS02) | (0 << CS01) | (1 << CS00) | (0 << WGM02) ;           //
CS02, CS01, CS00vare set to (0,0,1), which sets no prescaler.

       DDRB |= (1 << PB0); // sets PB0 (pin 5) as output

       sei(); //globla interrupt enable


    /* Replace with your application code */
       mode = 0;


    while (1)
    {
              // sets the OCR0A pin to out put 100%, 75%, 50%, 25%, and 0% power in 5
second intervals using the Motor_linear eqation
              //OCR0A = 255;

              int Analog_value_H = ADCH; // reads ADC the 8-bit high (most sig bits)
value  into this  integer. if ADLAR = 0, there is only two that can be read (0b000000xx)


              //ADCH
              // 8 least significant bits 2^8. between 0-255



              ADCSRA |= (1 << ADSC);         // start ADC measurement
              while (ADCSRA & (1 << ADSC) ){ // wait till conversion complete

                     //applies the ADCH value to the OCR0A pin.
                     OCR0A = ADCH;
              }
    }
}
```

Challenges

The only challenge I had with this project was getting the motors to spin in the first place. When I started this project, I set it up the exact same what I did on the Motor control assignment, so seeing the motors refuse to work confused me. Luckily the solution was simple: Just get a new 9V battery. Apparently the old one I was using was drained, which explains why the motors were not spinning before. After that, everything else was easy.