3D Printer Position Control

Nathan Hanuscin

Goal:

- Build mount to support rail system
- Use USART to take in user input on where to position item
- Use stepper motor to move item to desired position

Deliverables:

This project is intended to assist a user that is using 3D printing by allowing them to pick 3 positions for an item to be scanned. By using USART, the users input can be taken in and then from there a stepper motor will move the item to the specified positions.

I. Literature survey

3D printing has become increasing popular not just among hobbyists, but also in businesses and manufacturing in the recent years. Companies can now prototype and test different concepts more efficiently due to 3D printing. According to Forbes Contributor, Louis Columbus, 3D printing is used 34% of the time in proof of concept, 23% of the time for prototyping, 22% of the time in production, and only 5% for hobbyists. The demand in manufacturing and business is the majority of the market in 3D printing and these numbers are project to increase in the near future.

II. Components

A. *FTDI Breakout Board*

The FTDI breakout board uses a USB to serial connection to transmit and receive data. This board operates and 5V but can be modified to work at 3.3V. When data is being transmitted and received, there are on board LEDs that verify that data is being transferred serially. There are also 2 other pins on the FTDI, DTR and CTS, but these are more useful when using an Arduino.

B. *Unipolar Stepper Motor*

Stepper motors are used when speed isn't important but having a high torque is. The stepper motor works on similar principles as aDC motor. However, the stepper motor has a rotor in the middle, which is a magnet, and around that is the stator, which is made up of various electromagnets. By applying voltage only through one of electromagnets will cause the rotor to either stay at its position or rotate to the next. Also, a voltage and be applied to two adjacent magnets causing the rotor to go in-between them. This is known as half-stepping. Half-stepping was used in this project to rotate the middle rail.

C. L298N Motor Driver

The L298N takes in 4 inputs from the user's microcontroller, and outputs them with enough current and voltage to drive a motor since the ATmega328 is not capable to drive that amount of current. The L298N operates anywhere from 5-35V and can output 5V to drive other devices if necessary. Once the inputs are taken in they are amplified to a proper current and outputted to the motor to operate it. The user controls which inputs are being sent in via the ATmega.

III. SCHEMATICS

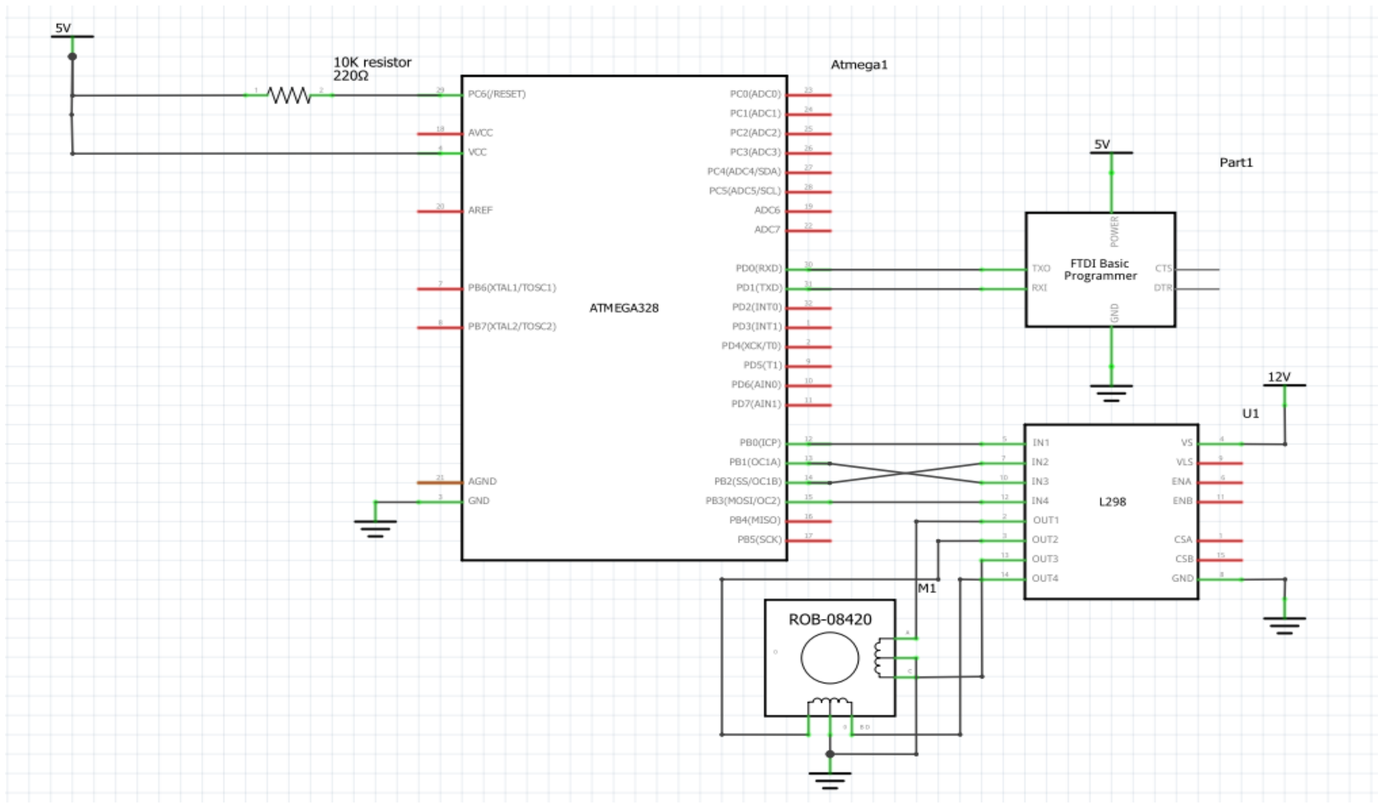

Figure 1: Schematic

IV. Implementation

- FTDI Breakout Board interfaced via USART sends information to the user on how to properly input the 3 desired positions. The user then sends the 3 positions back via USART and then the ATmega328 takes over.
    - To insure the correct values were inputted, the ATmega328 sends the inputted values back while the user is inputting them. This allows to user to see what numbers they pressed.
- ATmega328 was used to link the users positions and the L298N motor driver.
    - The ATmega328 converts the position values to integer values then calculates how long the stepper motor must operate to get to get position.
    - To determine how long the motor should rotate for, a for loop was used with a different number of iterations until the item moved approximately 1 centimeter.
- L298N motor driver supplies enough current to operate the stepper motor.
    - The L298N will drive the stepper motor until the ATmega328 stops sending signals to it.

V. Snapshots and Links

See power point for snapshots
Project Demonstration: https://www.youtube.com/watch?v=Yk-LB1Zu71k
Playlist: https://www.youtube.com/watch?v=Q2s7c8BWkkk&list=PLe1_lU5Cl2Kma14C87oWpewJ-OC9wxz4A
Project Presentation: https://www.youtube.com/watch?v=hWmUAiiog0U

VI. Code

```
#include <avr/io.h>
#define F_CPU 8000000UL
#include <util/delay.h>
#include <stdio.h>
#include <stdint.h>

#define UBRR_9600 51 // for 8Mhz with .2% error

void USART_init( unsigned int ubrr )
{
  UBRR0H = (unsigned char)(ubrr>>8);     //set baud rate
  UBRR0L = (unsigned char)ubrr;
  UCSR0B = (1 << TXEN0) | (1 <<RXEN0);     // Enable receiver, transmitter
  UCSR0C = (1 << UCSZ00) | (1 << UCSZ01);  //asynchronous 8-bit data 1 stop bit
}

void USART_tx_string( char *data )
{
  while ((*data != '\0'))
  {
    while (!(UCSR0A & (1 <<UDRE0))); //wait for the transmit buffer to empty
    UDR0 = *data;          //put the data into the empty buffer, which sends the data
    _delay_ms(50);         // wait a bit
    data++;
  }
}

void USART_Transmit( unsigned char data )
```

```
{

  while ( !( UCSR0A & (1<<UDRE0)) )
  {
    /* Wait for empty transmit buffer */
  }
  /* Put data into buffer, sends the data */
  UDR0 = data;
}

unsigned char USART_Receive( void )
{

  while ( !(UCSR0A & (1<<RXC0)) )
  {
    /* Wait for data to be received */
  }

  /* Get and return received data from buffer */
  return UDR0;
}

unsigned int get_number(int ones, int tens)
{
  unsigned char data;
  data = USART_Receive();        //get the tens digits
  USART_Transmit(data);        //transmit it back
  tens = data - 0x30;          //convert to a digit
  tens = tens *10;           //convert to tens place
  data = USART_Receive();        //get ones digit
  USART_Transmit(data);        //transmit it back
  ones = data - 0x30;          //convert to ones place
  return tens + ones;          //return position
}
//this function takes in the number to iterations to move the object forward to the desired
position
void go_foward(int iter)
{
  int i;
    for (i=0;i<iter;i++)
    {
      PORTB = 0x09;
      _delay_ms(75);
      PORTB = 0x0C;
      _delay_ms(75);
      PORTB = 0x06;
      _delay_ms(75);
      PORTB = 0x03;
      _delay_ms(75);
    }

}
//this function takes in the number of iterations to move the object back to the starting position
void go_back(int iter)
{
  int i;
  for (i=0;i<iter;i++)
  {
    PORTB = 0x03;
    _delay_ms(75);
    PORTB = 0x06;
    _delay_ms(75);
    PORTB = 0x0C;
    _delay_ms(75);
    PORTB = 0x09;
    _delay_ms(75);

  }

}

int main(void)
{ int ones_place, tens_place;
  unsigned int pos1, pos2, pos3;
  int iterations;
  unsigned char digit;
  USART_init(UBRR_9600);              //Initialize the USART (RS232 interface)

  USART_tx_string("Please enter 3 positions in centimeters with one space between each 
position\r\n");
  USART_tx_string("For single digit positions please put a zero in the tens place... \r\n");
  USART_tx_string("I.E. for 9 cm put 09\r\n");
  USART_tx_string("Example input: 01 10 25\r\n");
  USART_tx_string("Max value is 35cm\r\n");

  DDRB = 0xFF;                //set port B as an output
  ones_place = tens_place = 0;

  pos1 = get_number(ones_place, tens_place);

  digit = USART_Receive();         //get space
  USART_Transmit(digit);

  ones_place = tens_place = 0;
  pos2 = get_number(ones_place, tens_place);

  digit = USART_Receive();         //get space
  USART_Transmit(digit);

  ones_place = tens_place = 0;
  pos3 = get_number(ones_place, tens_place);

  if (pos3 > 35)
    pos3 = 35;              //if position 3 is out of range change it to max value

  iterations = pos1 * 80;        //80 iterations = 1cm, so multiply position value by 80
  go_foward(iterations);         //move object to position 1
  _delay_ms(5000);           //wait a bit

  iterations = (pos2 - pos1) * 80;   //get difference first two positions then multiply by 80
  go_foward(iterations);         //move object to position 2
  _delay_ms(5000);           //wait a bit

  iterations = (pos3 - pos2) * 80;   //get difference second and third position then multiply by 80
  go_foward(iterations);         //move object to final position
  _delay_ms(5000);           //wait a bit

  iterations = pos3 * 80;        //get total iterations made
  go_back(iterations);         //to return to starting position

  return 0;
}
```

Acknowledgment

Thanks to my dad for helping me construct the rail system.

References

[1] 3D printing stats: https://www.forbes.com/sites/louiscolumbus/2017/05/23/the-state-of-3d-printing-2017/
[2] FTDI breakout board: https://www.sparkfun.com/products/9716
[3] L298N: https://www.bananarobotics.com/shop/How-to-use-the-L298N-Dual-H-Bridge-Motor-Driver
[4] Stepper motor + L298N: http://www.instructables.com/id/Control-DC-and-stepper-motors-with-L298N-Dual-Moto/