Lab 9: Limit Switch Implementation

EEE 174

Anastasiya Herasimava

Thomas Bock

Daniel Komac

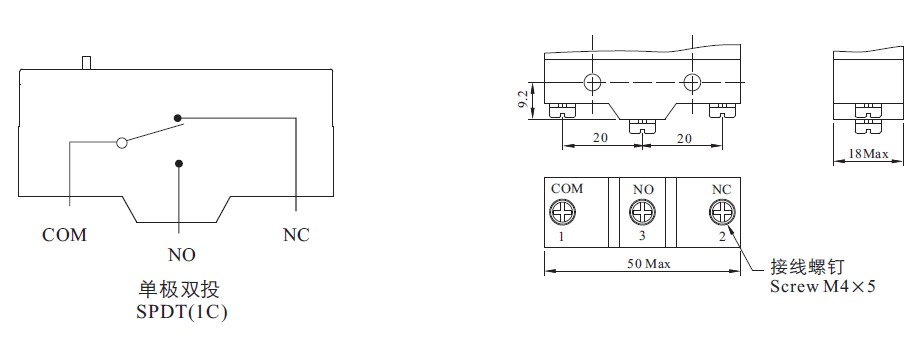
Slavic Chiley

CSUS

December 8, 2015

Introduction

We decided to go with a Robotic Arm as our final project. Limit Switches were implemented in this project as a control element. Their function was to limit the radius of motion of the step motors. As known, step motors unlike servo motors are not limited to a certain range of motion and can do a full rotation. Rotation of the step motor more than 180 degree could damage the wiring; therefore, two limit switches were installed on the base 180 degrees apart for protection purposes setting safety limit of motion to 180 degrees. Limit switches are operated by the movement of an arm part. When the object hits the plunger, Arduino receive a signal and restricts motion of the related step motor in that direction.   
The limit switch has 3 contacts via COM, NO, NC. When plunger is not pressed then COM and NC are connected and NO disconnected. When plunger is pressed COM and NO connects and NC disconnects. NO is the normally open contact, NC is the normally closed contact and COM is the common. Below you can see a pin out of the limit switch we used for our project as well as the actual picture of the switch.

Procedure

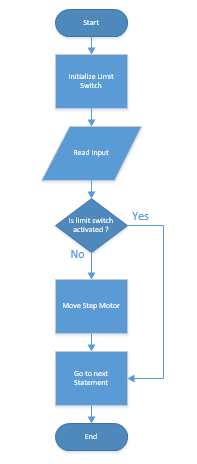
First step was testing the limit switches. We hooked up the limit switch to the Arduino and wrote a simple code to make sure that it responds to hitting the plunge. Once we verified that the switches are in the working condition we started thinking of the appropriate places to install the switches at and ways to attach them on the surface of the robotic arm. We decided to the place the switches 180 degrees apart on the base of the robotic arm thus limiting the motion of one of the step motors to 180 degrees. No better way other than gluing the switches to the structure was found. In order to hit the plunger of the limit switches a small detail in front of the structure was place. You can the switch attached to the base of the robotic arm below.



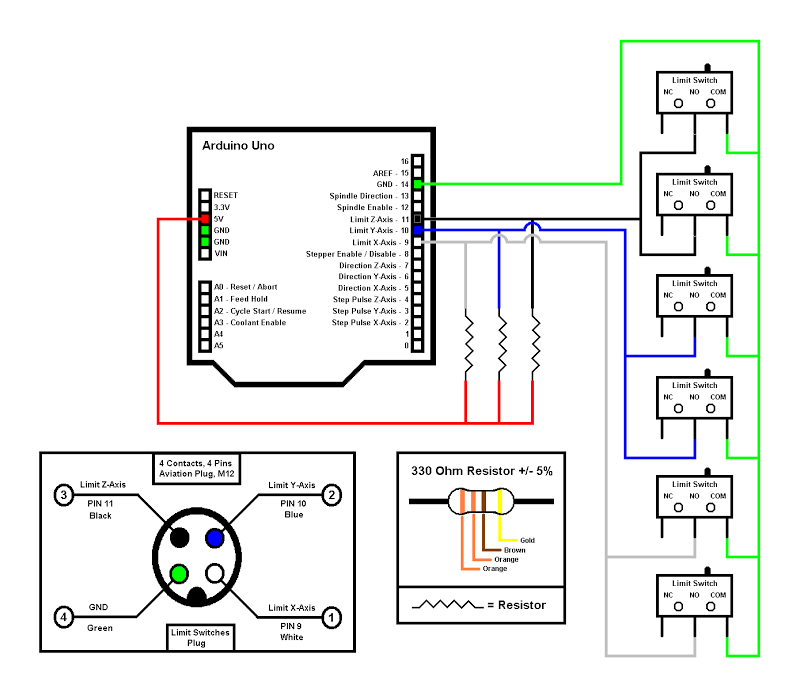
Next step was writing a program that will stop the corresponding step motor when plunge is hit. This part of the coding had to be done after the code for the joysticks were complete. As you can see from the code below, there is an if-statement inside the if-statement. The if-statement inside is checking whether that limit switch is activated. If the switch is activated, it simply prevents execution of the else-statement and thus preventing the motor from moving clockwise or counterclockwise. If the plunge on the switch was not hit, the else-statement is being executed and step motor moves in desired direction.

|  |
| --- |
| if (analogRead(JOY\_3\_PIN) > 1012) { // If joystick is moved Left  if (!digitalRead(Y\_MAX\_PIN)) { // check if limit switch is activated  }  else  { // if limit switch is not activated, move motor clockwise  stepper1.move(30);  stepper1.run();  }  }    if (analogRead(JOY\_3\_PIN) < 212) { // If joystick is moved right  stepper1.move(-30);  stepper1.run();  }  }  if (analogRead(JOY\_2\_PIN) > 1012) { // If joystick is moved Left  stepper3.move(30);  stepper3.run();  }  }    if (analogRead(JOY\_2\_PIN) < 212) { // If joystick is moved right    if (!digitalRead(Y\_MIN\_PIN)) { // check if limit switch is activated  }  else { // if limit switch is not activated, move motor counter clockwise  stepper3.move(-30);  stepper3.run();  }  }  if (analogRead(JOY\_1\_PIN) > 1012) { // If joystick is moved Left  if (!digitalRead(Y\_MAX\_PIN)) { // check if limit switch is activated  // delay(5000);  }  else { // if limit switch is not activated, move motor clockwise  stepper2.move(30);  stepper2.run();  }  }    if (analogRead(JOY\_1\_PIN) < 212) { // If joystick is moved right    if (!digitalRead(Y\_MIN\_PIN)) { // check if limit switch is activated  // delay(5000);  }  else { // if limit switch is not activated, move motor counter clockwise  stepper2.move(-30);  stepper2.run();  }  } |

Next you can also see the flowchart showing the logic behind limit switch portion of the script.



The part that we had the most issues with was hooking up the limit switches to the Arduino Mega and making them send a proper continuous signal. Initially we didn’t account for high impedance of the Arduino pins when used as input and therefore didn’t utilize a pull up resistor in our configuration. We were getting a mixture of on and off input when the plunge was hit thus making our arm not moving properly. We did our research and added a 330 resistor from the NO node on the switch to the 5V output on the Arduino. 330 Ohms resistor didn’t work well for us. By trial and error we selected 20k Ohms resistor that helped to resolve the issues. You can see the connection configuration in the diagram below.



After resolving the issues with inconsistent signal, we were able to make the switches function properly and tested its operation when using in combination with joystick.

Conclusion

This project was a great learning experience. We were able to implement the knowledge acquired earlier this semester as well as learn a lot from solving the problems that came up during the work on the project. We were able to implement the limit switches, connected the switches to the Arduino and made them send a proper signal. We also were able to write the code responsible for restricting the step motor in the direction of limit switch as the switch is activated. We can consider this project a success. This project gave a great opportunity to implement our knowledge as well as practice our team work skills and project planning skills.