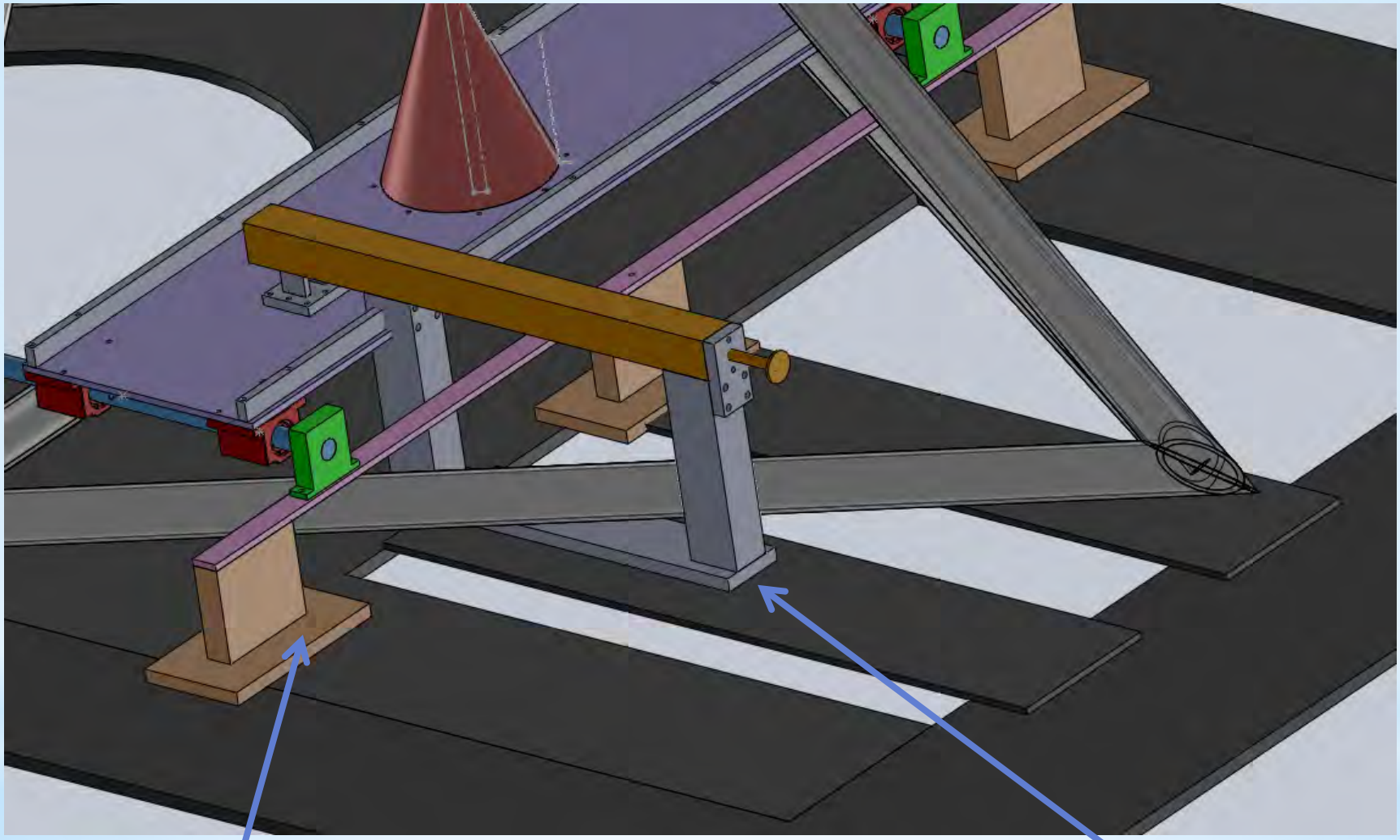
Load Mover Design

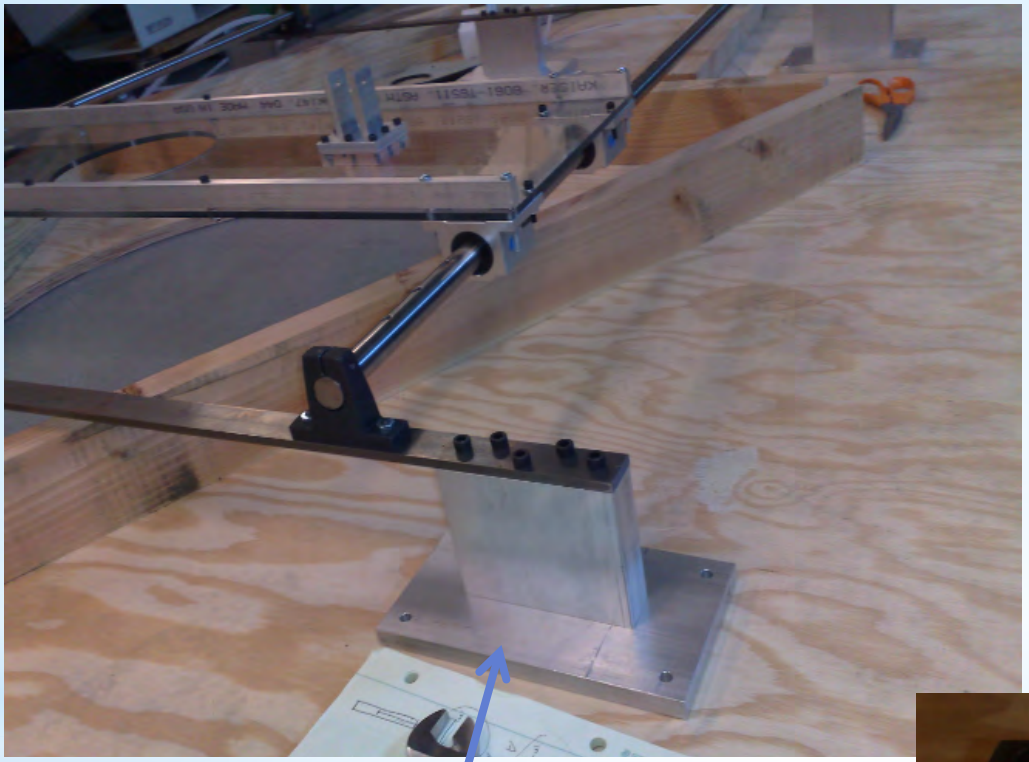
David Liu and David Kooi

August 2011

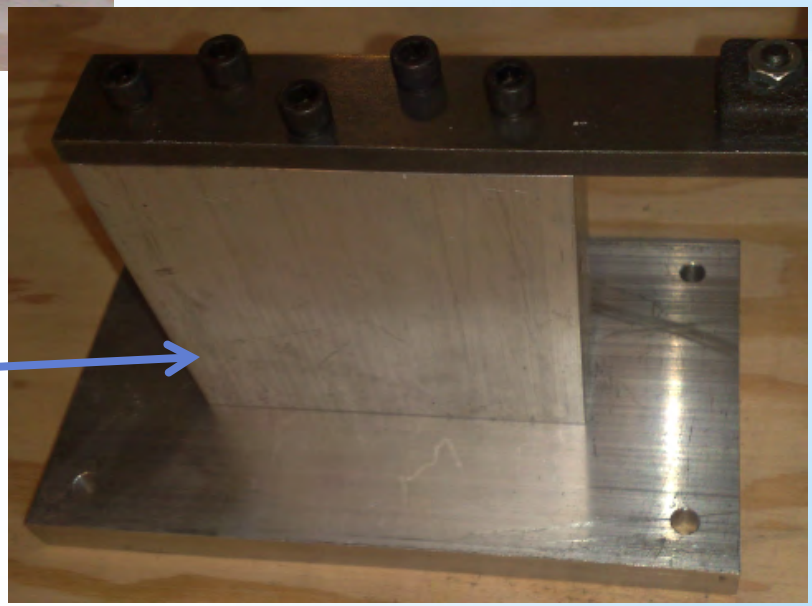


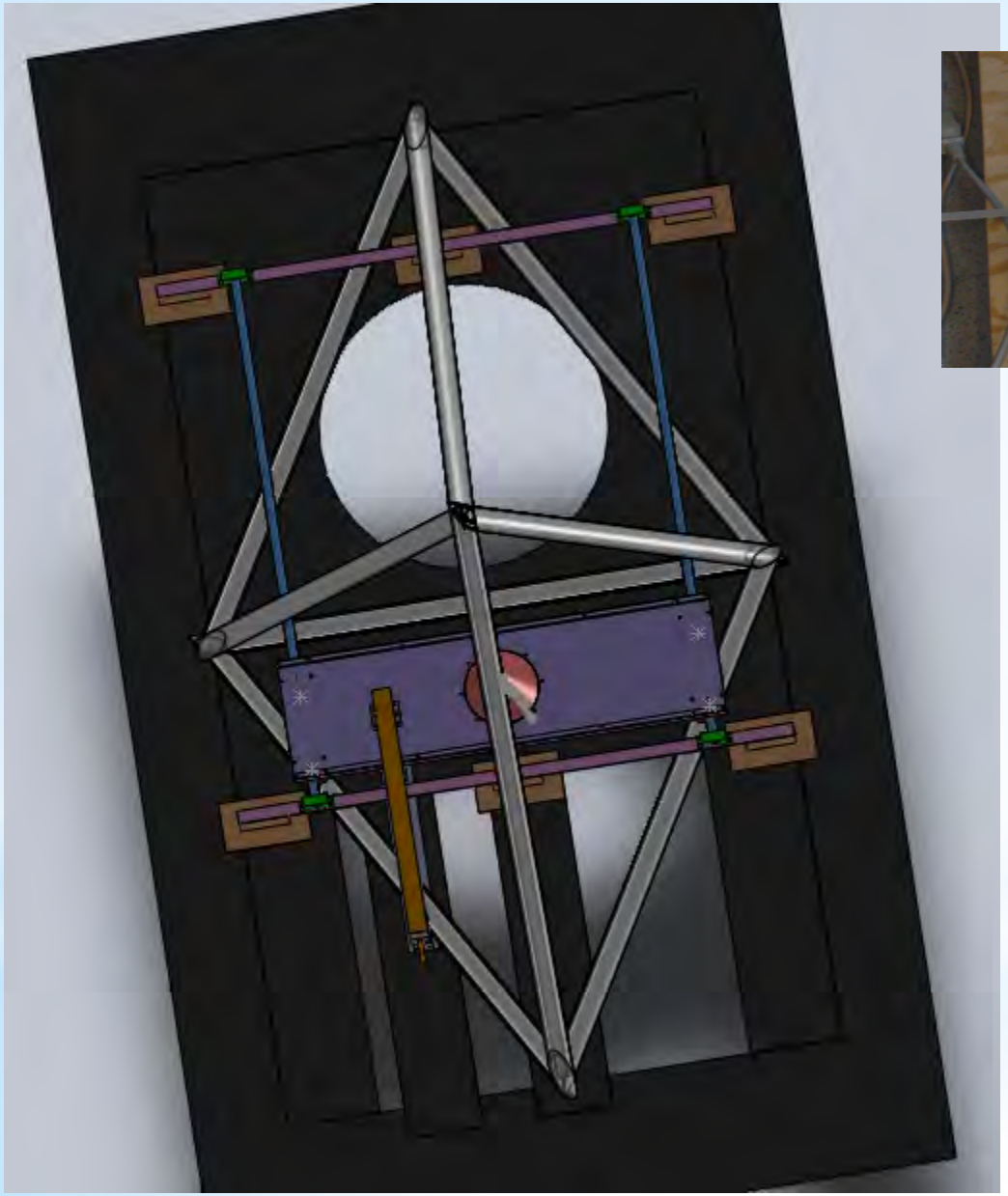
6 support pieces fixed to CSO hex plate

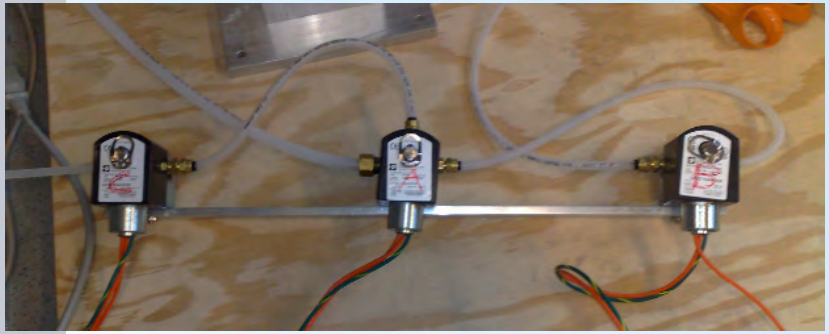
Another ¼ inch thick steel plate welded over rectangular hole to mount actuator



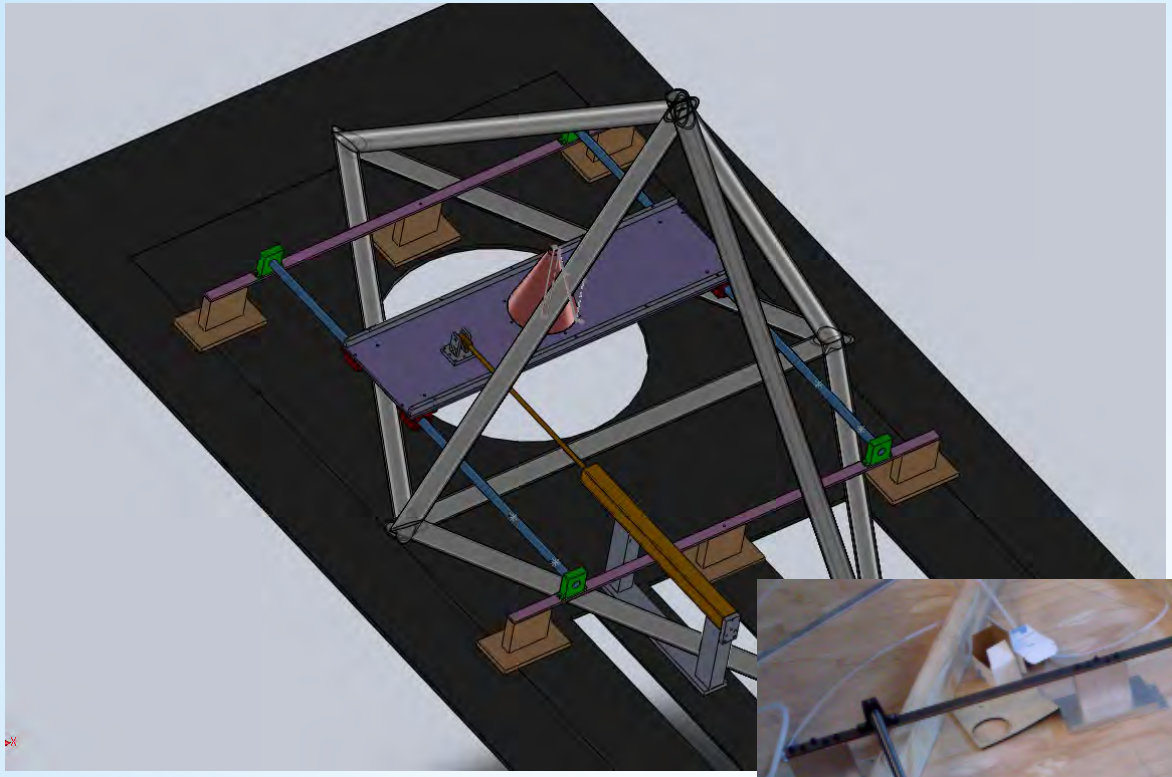
6 support pieces fixed to CSO hex plate by drilling and tapping holes

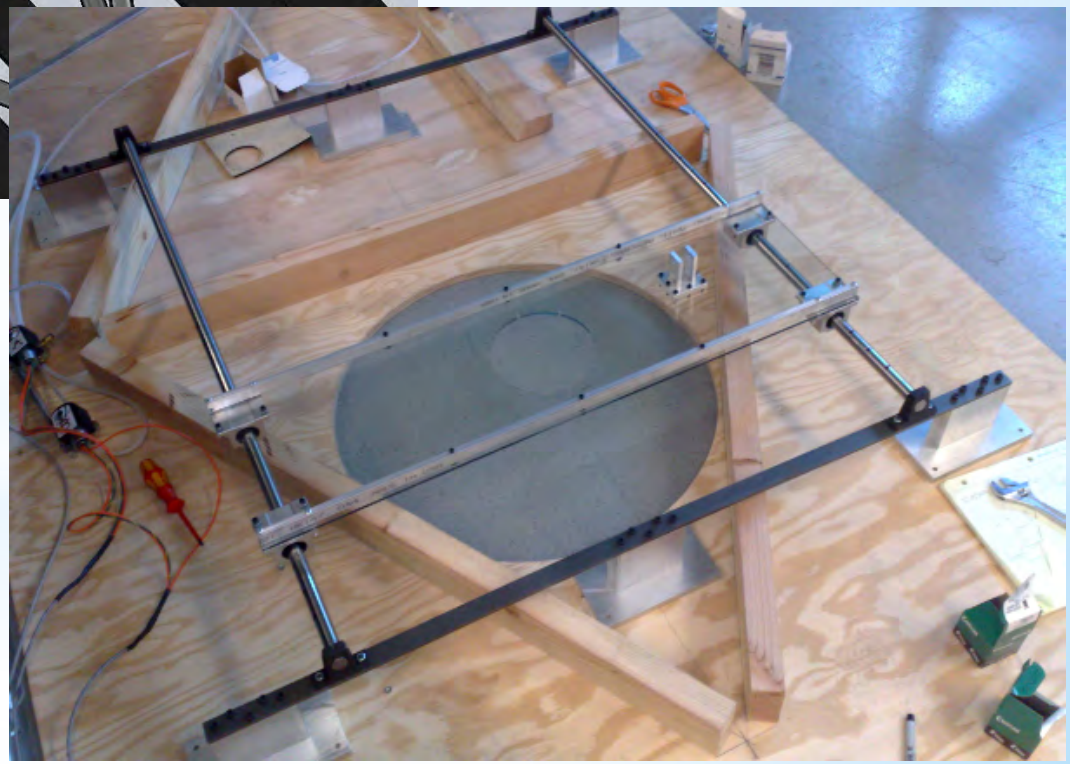




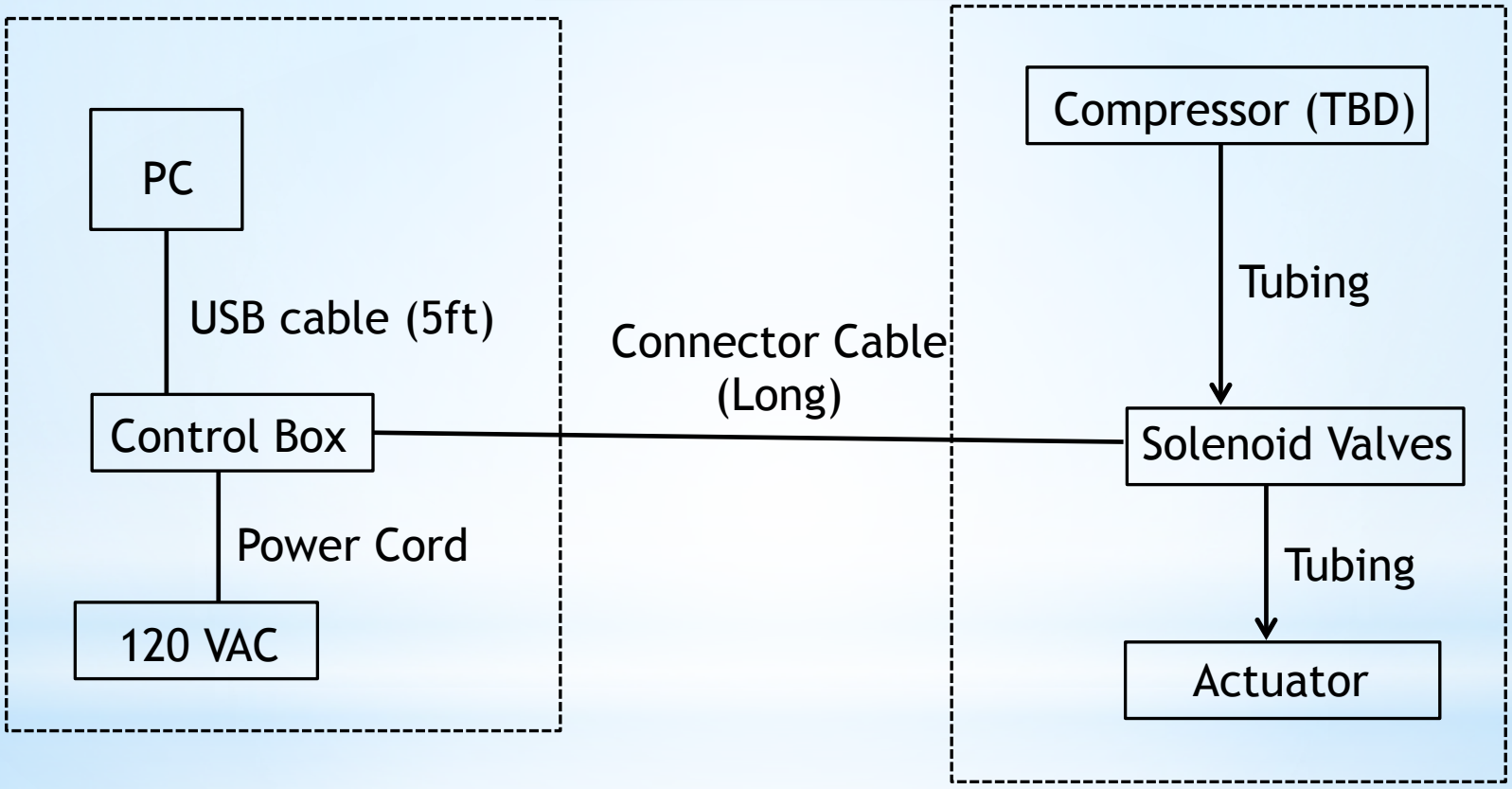


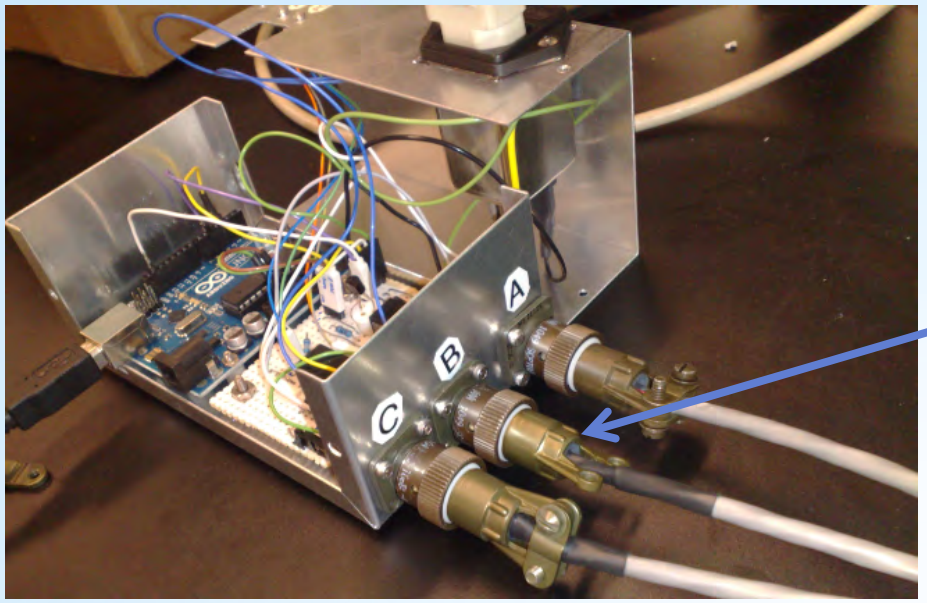
Where to place:
Compressor
Solenoid Valve Bar



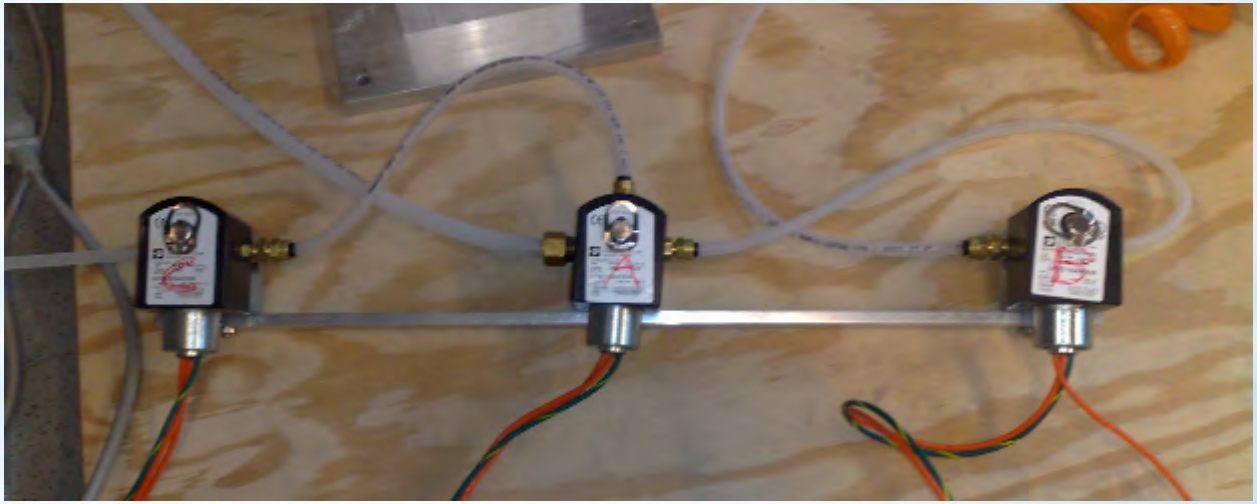


*Parts Layout





Connectors can be spliced into longer cable



Solenoid valves mounted on aluminum bar. Can be mounted wherever convenient At the CSO.

*Compressor

*Flow rate about 1.1 CFM

*Capacity?

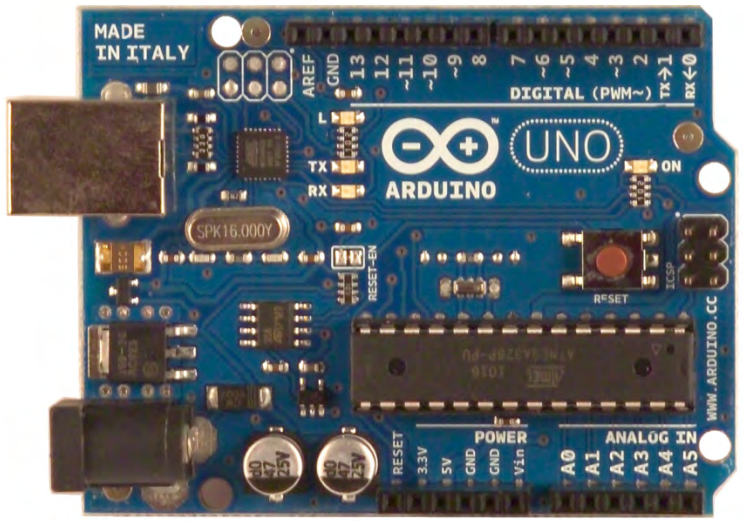
9965K62

- Air flow rate: 1.7 CFM
- Max PSI: 150
- Hp: 4/5
- Voltage at 60 Hz: 120 VAC
- Full Load Amps: 10
- Tank Capacity (gal): 6
- Fits into 20x20x20 inch cube

*Control Box

- Components
  - **Arduino Uno** microcontroller
  - Two SPDT relays to control the valves
    - **OJE-SH-105DM**
    - Or RadioShack: http://www.radioshack.com/product/index.jsp?productId=2062480
- Reliability
  - Two spare relays and one spare microcontroller.
  - One spare solenoid valve
  - Solenoid valves get warm?

Relay
Nominal Coil Voltage: 5VDC
Nominal Coil Current: 89.3 mA
Coil Resistance: 56 ohms
Rated 1A at 120 VAC



*Programming

- Microcontroller program
  - Switches relays to control valves
  - Valves are always in one of the following 4 states:
    - A extend
    - B retract
    - C exhaust
    - D exhaust
- Software
  - Arduino can interface with other software
  - Other software writes A, B, C or D to Arduino's communication port to control valves

Microcontroller Program	Other Software
Responds to the characters A, B, C, or D → switches relays	Open and write to Arduino's communication port. Writes A, B, C, or D.

Arduino Sketch

```
/*
  David Liu and David Kooi
  July 2011
*/
#define Relay 3
#define Relay2 5

void setup()
{
  pinMode(Relay, OUTPUT);
  pinMode(Relay2,OUTPUT);
  Serial.begin(9600); // open serial
  Serial.println("A open B close C pause");
}
```

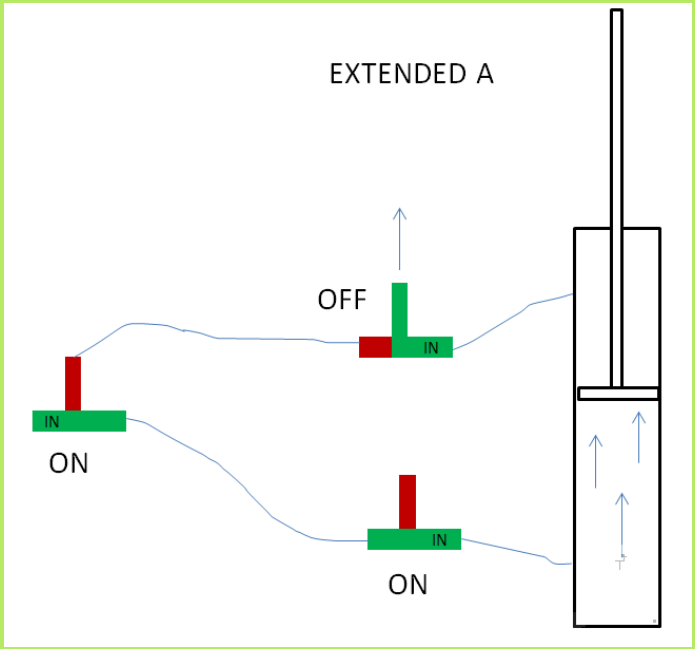
```
void loop()
{
 String state="";
 while (Serial.available() > 0)
 {
  char x=Serial.read();
  state+=x;
 }
 if(state.equals("A")) //extend
 {
   Serial.println("A is the state");
   digitalWrite(Relay, HIGH);
   digitalWrite(Relay2, HIGH);
 }
 if(state.equals("B")) //retract
 {
   Serial.println("B is the state");
   digitalWrite(Relay,LOW);
   digitalWrite(Relay2,LOW);
 }
 if(state.equals("C")) //stop air flow when pushing in
 {
   Serial.println("C is the state");
   digitalWrite(Relay,HIGH);
   digitalWrite(Relay2,LOW);
 }
 if(state.equals("D")){ //stops air flow when pushing out
   Serial.println("D is the state");
   digitalWrite(Relay,LOW);
   digitalWrite(Relay2,HIGH);
 }

}
```

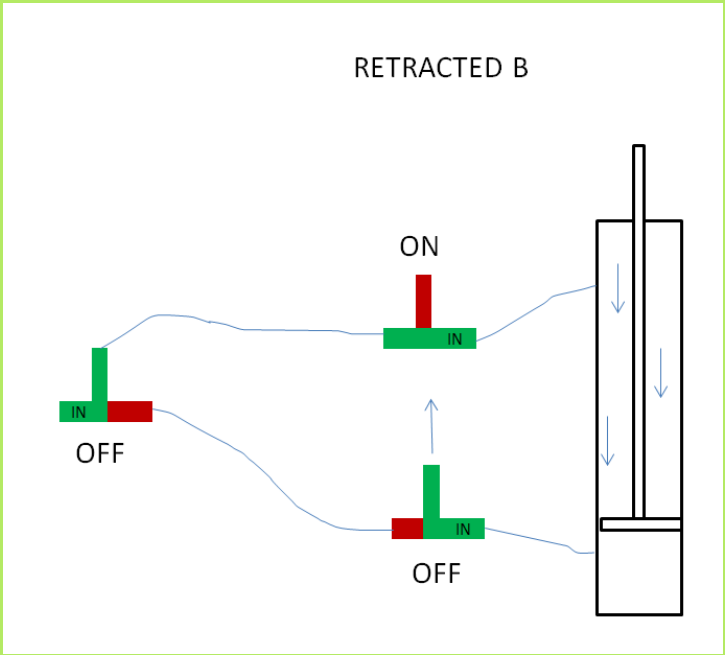
Input: A, B, C, or D

Output: switches relays to their Correct configuration

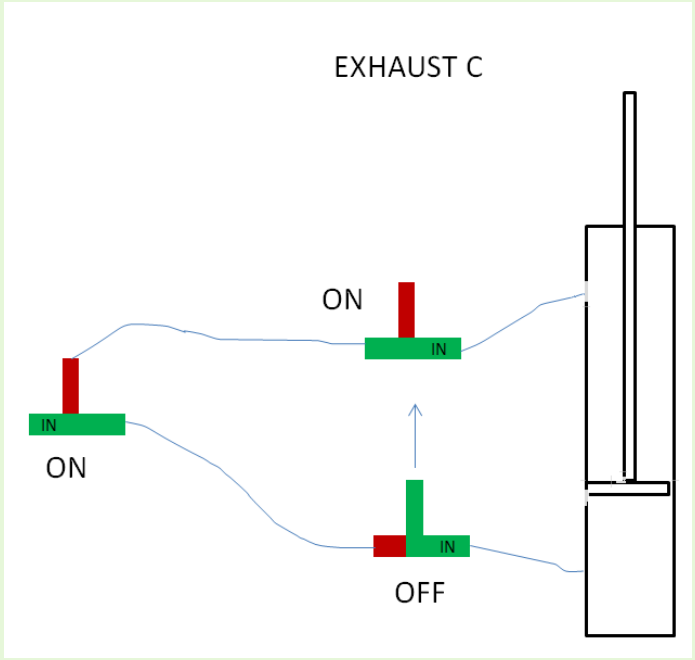
A


EXTENDED A
OFF
IN
IN
ON
IN
ON

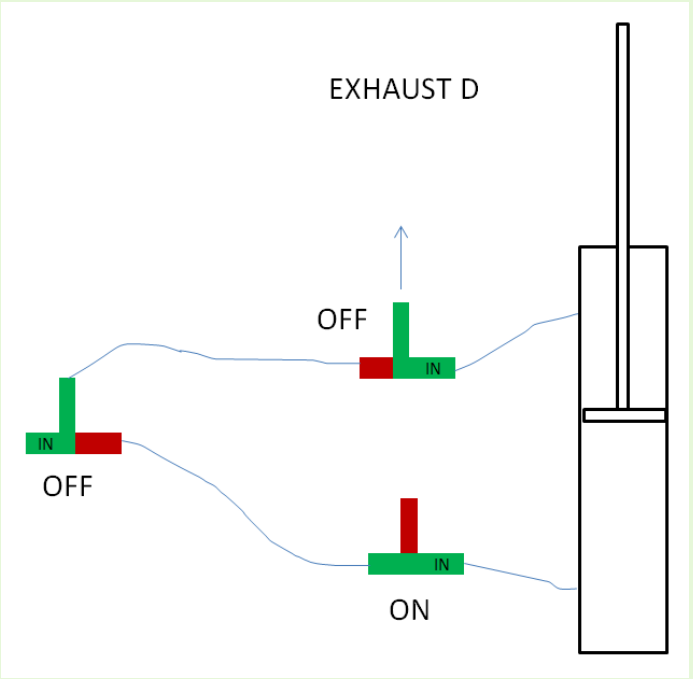
B


RETRACTED B
ON
IN
IN
OFF
IN
OFF

C

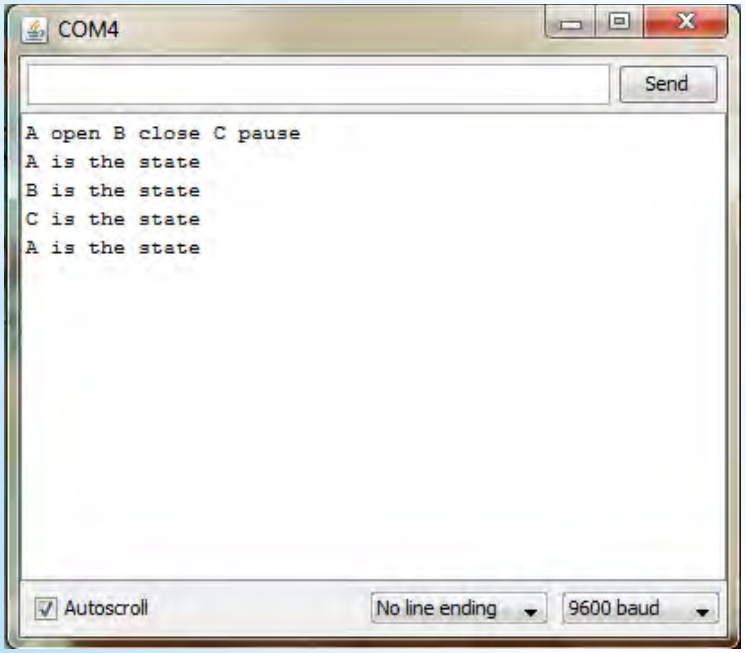

EXHAUST C
ON
IN
IN
ON
IN
OFF

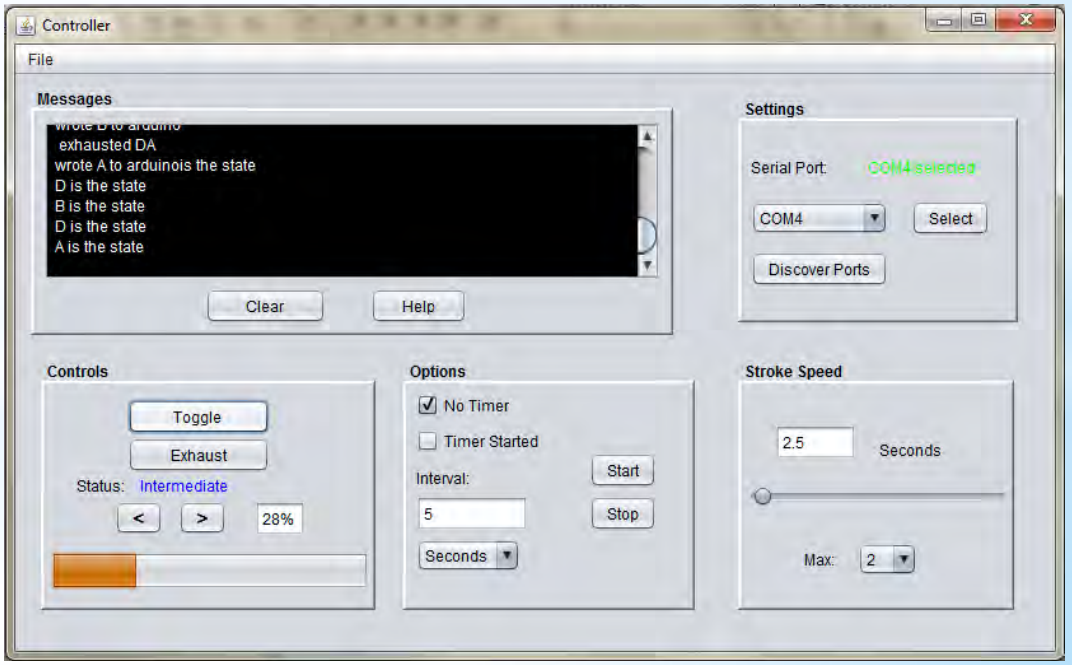
D


EXHAUST D
OFF
IN
IN
OFF
IN
ON

Using Control Box

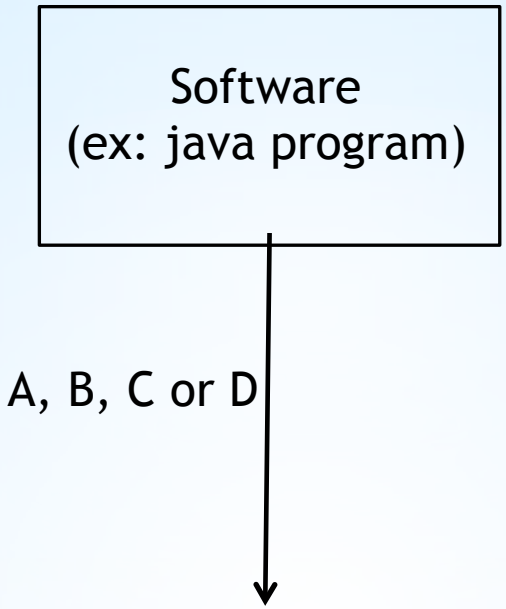
- Arduino IDE has built in serial monitor
- It is possible for the Arduino to interface with other software



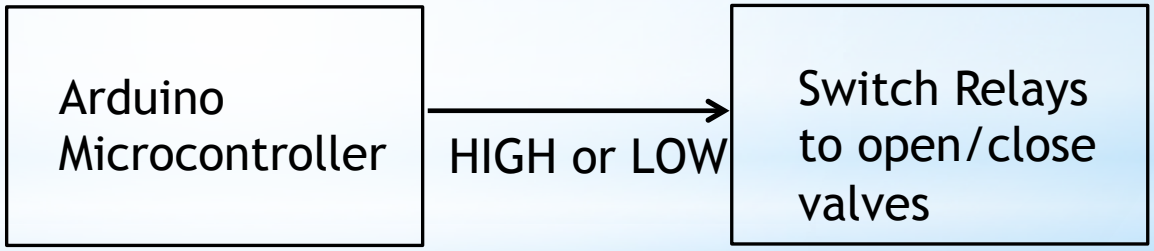


* Arduino and other software

* Arduino **input** is array of bytes representing the string “A”, “B”, “C” or “D.”
* Software needs to be able to open and write to Arduino’s communication port
  * **Output** will be the one of the characters A, B, C or D.



Software
Input: Arduino communication port(used CommPortIdentifier class in Java)
-determine communications ports available
-open communication port
Output: A, B, C, or D



Arduino
Input: A, B, C, or D
Output: writes HIGH or LOW to relays

* Arduino and Java

Setting up communication port

```
CommPortIdentifier portId = CommPortIdentifier.getPortIdentifier(com);
SerialPort port = (SerialPort)portId.open("", 4000);
input = port.getInputStream();
output = port.getOutputStream();

// add event listeners
port.notifyOnDataAvailable(true);

port.setSerialPortParams(9600,
SerialPort.DATABITS_8,
SerialPort.STOPBITS_1,
SerialPort.PARITY_NONE);
```

Communicating with Arduino

```
try{
    if(!message.equals("")){
        output.write(message.getBytes());
    }
}
catch(Exception e){
    textArea.setText(textArea.getText()+"\n"+e);
}
```