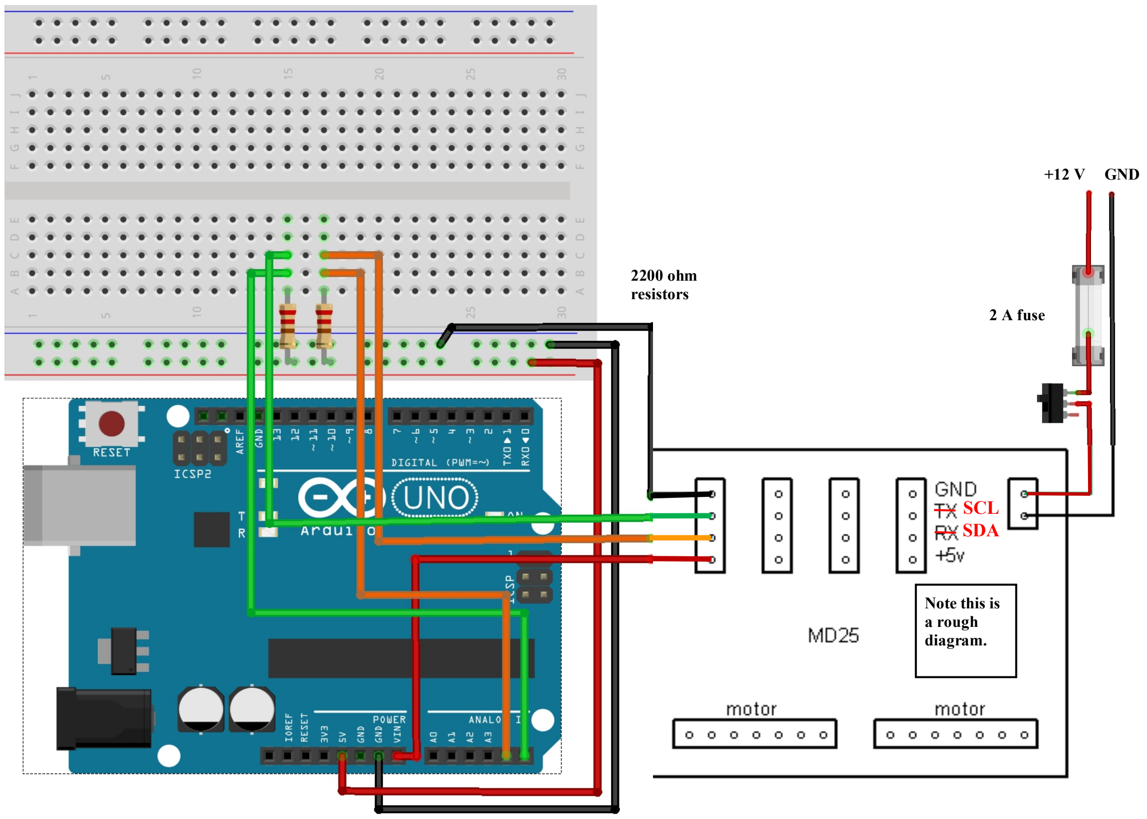
MD25 Circuit Guide

1. The MD25 is a motor driver that we will communicate with Arduino via I2C.
   Read the technical datasheets!
   http://www.robot-electronics.co.uk/htm/md25tech.htm
   http://www.robot-electronics.co.uk/htm/md25i2c.htm

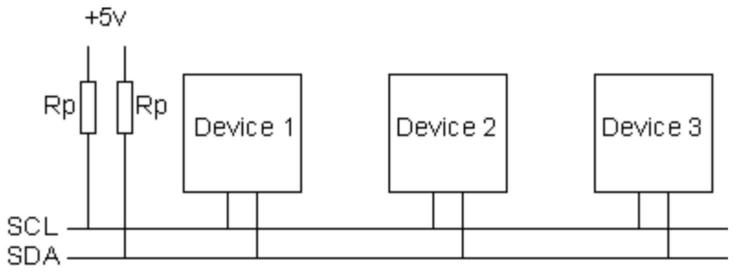
2. Circuit Diagram

Study the MD25 datasheets thoroughly. Then connect A4 to SDA, and A5 to SCL and **MAKE SURE YOU GET THE POLARITY OF THE 12 V BATTERY CORRECT. IT WILL BLOW IF YOU WIRE IT UP WRONG.**



Key things:

- The MD25 is wired up to the 12 V battery. This part of the circuit needs a fuse (2 A is ideal) and toggle switch to turn the MD25 off, both as safety precautions. MD25s cost £40.62 inc. VAT to replace, whereas fuses cost pennies. Use fuses!
- The MD25 outputs a 5 V supply and plugs into 'Vin' of the Arduino and powers it. This means the Arduino will only turn on when the MD25 is turned on.
- The MD25 and Arduino need a common ground.
- Connect A4 (SDA) to SDA on the MD25 (RX in this diagram), and A5 (SCL) to SCL on the MD25 (TX). These I2C lines need external pull-ups from a 5 V supply (the Arduino). We shall use **2200 ohm** resistors (red stripes), although 4k7 are recommended.



This is a typical I2C setup, with SCL and SDA outputted from the Arduino, and our MD25 is device 1. We don't need any other devices in our circuit.

3. Wire library - http://arduino.cc/en/reference/wire
   An Arduino library is used to communicate with I2C devices. The datasheets contain all the relevant addresses and commands in Hexadecimal (base 16).

   Example: To reset the motor encoder values to zero.

```
Wire.beginTransmission(0x58);   //begins transmission to 7bit MD25 address
Wire.write(0x10);               //accesses the command register
Wire.write(0x20);               //resets the encoder registers to zero
Wire.endTransmission();         //ends this piece of I2C communication
```

   Note: You need to quickly become familiar with using Hexadecimal numbers, as it will save confusion when trying to write your own I2C functions. Do some research!

   Eg. The command register is number 16 in base 10, which is the equivalent of 0x10 in base 16. The '0x' is just to show the number is Hexadecimal, which is the notation used in our C++ programming.

4. The example code comes from an old example that can be found here https://www.robot-electronics.co.uk/htm/arduino_examples.htm#MD25%20RD02%20Motor%20Controller. (Note that the full example does combine an MD25 with an additional LCD screen that we are not using.)

```
long encoder1(){     // Function to read and display value of encoder 1 as a long

Wire.beginTransmission(0x58);  // MD25 address
Wire.write(0x2);               // to read the position of encoder 1
Wire.endTransmission();

Wire.requestFrom(0x58, 4);     // Request 4 bytes from MD25
while(Wire.available() < 4);   // Wait for 4 bytes to arrive
long poss1 = Wire.read();      // First byte for encoder 1, HH.
poss1 <<= 8;
poss1 += Wire.read();          // Second byte for encoder 1, HL
poss1 <<= 8;
poss1 += Wire.read();          // Third byte for encoder 1, LH
poss1 <<= 8;
poss1 +=Wire.read();           //Final byte

return(poss1);
}
```

   All this is doing is taking the 4 bytes from the MD25 that make up the encoder count and organising them into a long (type of value). The '<<=' is a form of 'Bit Shift Operator' and it is combining the 4 bytes (each 8 bits) into one large 32 bit long value. This 32 bit long called 'encoder1()' is useful to us.

This 'encoder1()' value (returned from the function) is in encoder counts, and there are 360 encoder counts per one full revolution of the output shaft (360 degrees). For example you could convert it to a distance by using the circumference of your wheels.

5. Finally if you do get completely stuck, then you can e-mail me at thomas.blumensath@soton.ac.uk, but I'm not going to hand over any information easily as part of this task is to get you to figure things out yourself within your team.