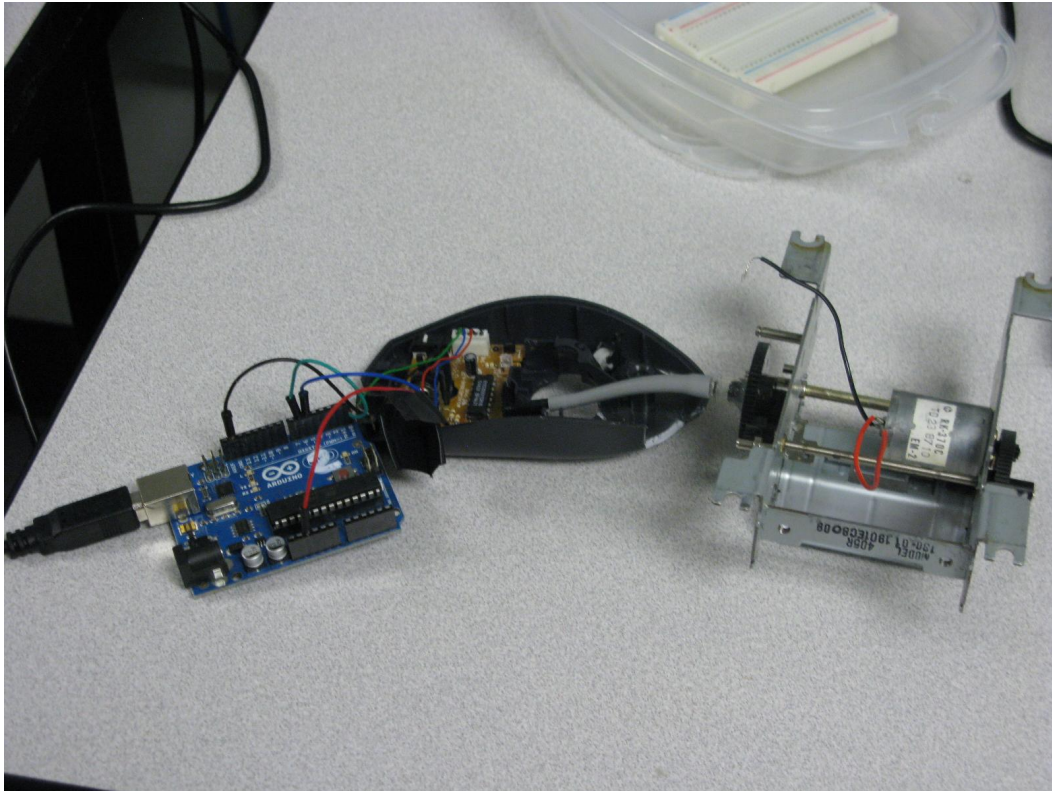


# Reading ps2 mouse output with an Arduino

Kyle P & Mike K

4 / 24 / 2012



Our goal was to create a piece of equipment that senses how much a robot drifts left to right by using an optical encoder.

Background:

Our original plan was to build an optical tachometer, but we discovered that a computer mouse has a optical encoder built into it. We then changed our plan to build a tachometer from a mouse.

Later we found a arduino java library that allowed access to the output of the mouse. When we tried to use the library an error came up because the library for the older arduino software didn't work. We then had changed Wprogram.h to Arduino.h. We then tested the

mouse with arduino and it had worked. Next, we had measured each mouse part to find out how many rpms the mouse ball and the gears could turn. We had set up the motor test unit 147 and 549 and had hand counted the rpms off of the slow gear. When we finished the slow gear, we had then hand counted the next gear. We then set up the fast gear into the tube that connects to the motor. It had calculated to be 33 rpms on the serial monitor. We then made a prediction for the slow gear, tested it and had confirmed that mu. equals 33 rpms.

## Reading the mouse

1. Found the ps2 program ([Attach:ps2.zip](#))
2. changed Wprogram.h to Arduino.h because we are using a newer arduino
3. tested mouse with arduino

mouse wire color	function	arduino input
Green	clock	6d
Blue	data	5d
Red	power	5 volts
Black	ground	ground

```
#include <ps2.h>
```

```
/*
```

```
 * an arduino sketch to interface with a ps/2 mouse.
```

```
 * Also uses serial protocol to talk back to the host
```

```
 * and report what it finds.
```

```
*/
```

```
/*
```

```
 * Pin 5 is the mouse data pin, pin 6 is the clock pin
```

```
 * Feel free to use whatever pins are convenient.
```

```
*/
```

```
PS2 mouse(6, 5);
```

```
/*
```

```
 * initialize the mouse. Reset it, and place it into remote
```

```

    * mode, so we can get the encoder data on demand.
    */
void mouse_init()
{
    mouse.write(0xff); // reset
    mouse.read(); // ack byte
    mouse.read(); // blank */
    mouse.read(); // blank */
    mouse.write(0xf0); // remote mode
    mouse.read(); // ack
    delayMicroseconds(100);
}

void setup()
{
    Serial.begin(9600);
    mouse_init();
}

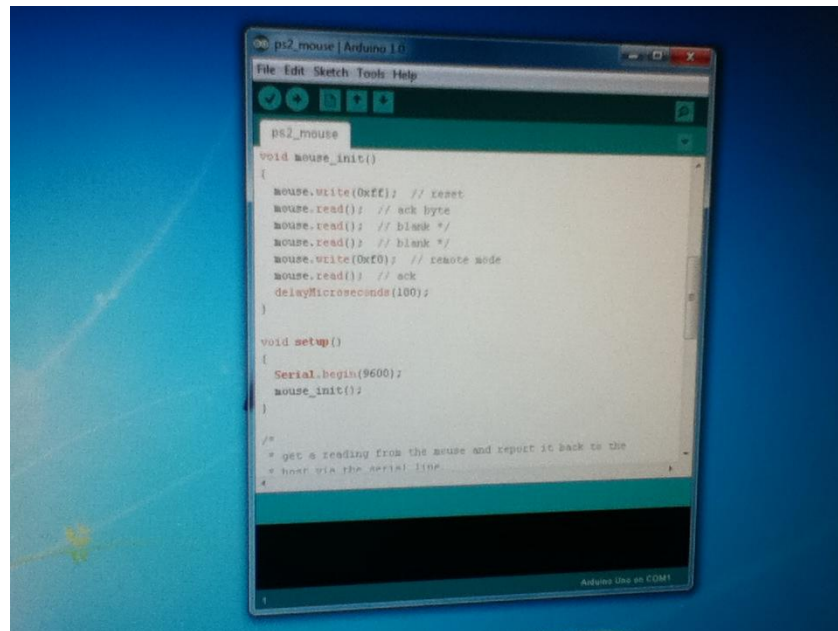
/*
 * get a reading from the mouse and report it back to the
 * host via the serial line.
 */
void loop()
{
    char mstat;
    char mx;
    char my;

    /* get a reading from the mouse */
    mouse.write(0xeb); // give me data!
    mouse.read(); // ignore ack
    mstat = mouse.read();
    mx = mouse.read();
    my = mouse.read();

    /* send the data back up */
    Serial.print(mstat, BIN);
    Serial.print("\tX=");
    Serial.print(mx, DEC);
    Serial.print("\tY=");
    Serial.print(my, DEC);
    Serial.println();
}

```

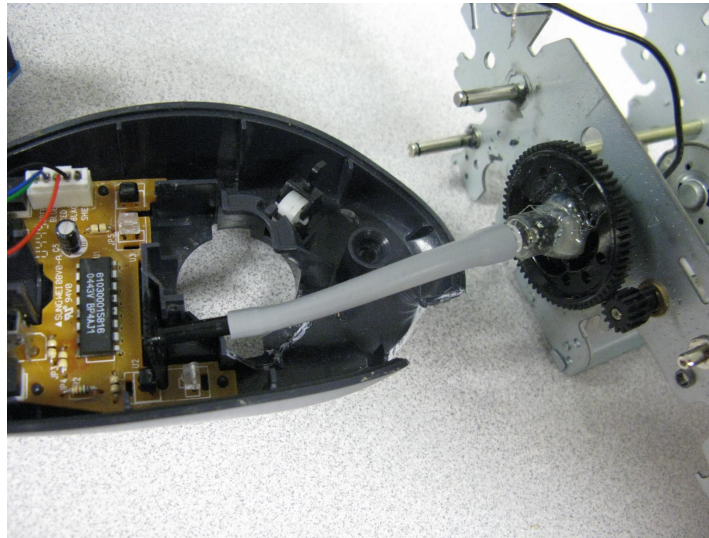
```
// delay(20); /* twiddle */
}
```



## Setting up testing

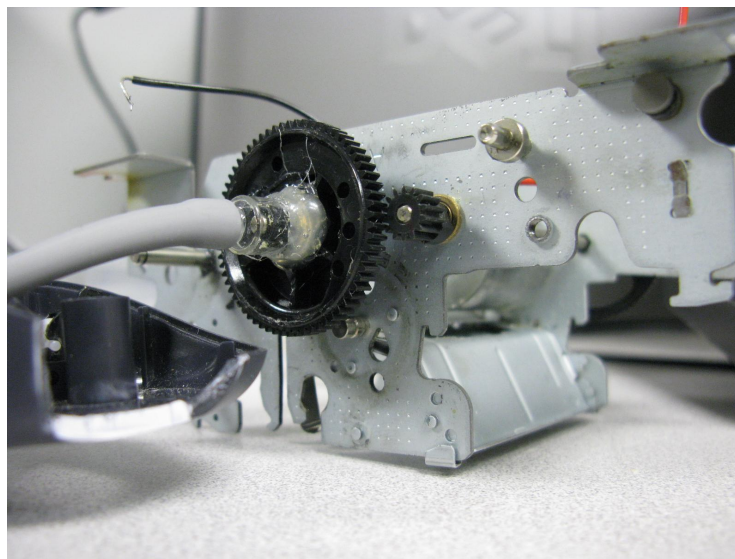
1. Measured the mouse encoder wheel drive cylinder and the ball
  - a. to know how many revolutions a certain distance rotated the ball

Item	diameter	circumference	divisions	Rpms
mouse ball	21.8 mm	68.8 mm		216.4
encoder wheel	3.6 mm	11.3mm	50 tick marks	100



### Set up a motor test unit

- we counted the teeth on the test unit 147 and 549 ( 3.7:1 )
- we hand counted rpms off of slow gear
- we calculated speed of the next gear using the 3.7:1 ratio

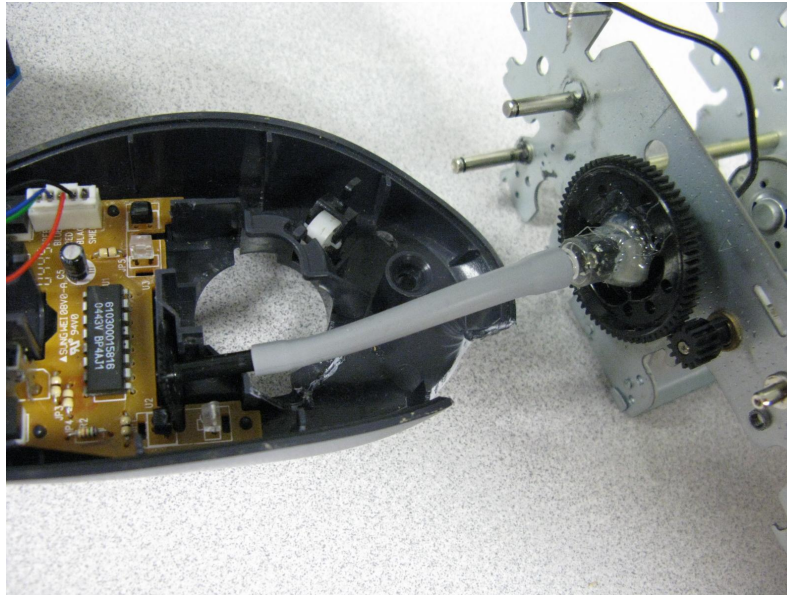


### Testing

- connected the fast gear to the mouse encoder wheel using small rubber tube



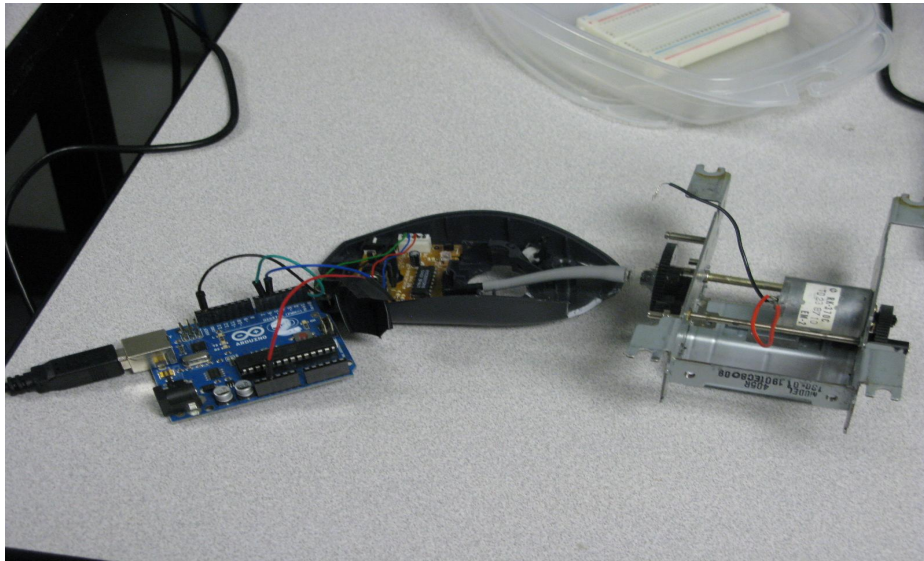
calculated 33rpm from each m.u. made prediction of speed for slow gear  
tested it and confirmed that mu equal 33 rpms



We did a things to do list to frame the project. It helped us go step by step and not forget anything we had to do in the project or get ahead of ourselves. We also came up with a things to do list so if had to go back to remember something we could. We also created a things to do list so people reading this know the steps we did.

## Sequence

1. Looked up program.
2. Downloaded the library.
3. Moved the ps2 library to the arduino folder.
4. We ran ps2 mouse library.
5. Loaded the software onto arduino.
6. Plugged in arduino based off of mouse colors.
7. Ran the code on the arduino and then run serial monitor and check if it works.  
(serial monitor is located in the tools menu of the arduino software)
8. Once confirmed the mouse is working rewrite the code for the rpm's.
9. Changed wprogram.h to arduino.h.
- 10 Loaded software onto the arduino.
- 11 Hooked up the arduino to mouse.
- 12 Figured out how to test the mouse software.
13. Tested serial monitor with the example code using analog read serial.
14. We set up the test unit for the arduino.
15. Tested all the software.



```
#include <ps2.h>

/*
 * an arduino sketch to interface with a ps/2 mouse.
 * Also uses serial protocol to talk back to the host
 * and report what it finds.
 */

/*
 * Pin 5 is the mouse data pin, pin 6 is the clock pin
 * Feel free to use whatever pins are convenient.
 */
PS2 mouse(6, 5);

/*
 * initialize the mouse. Reset it, and place it into remote
 * mode, so we can get the encoder data on demand.
 */
void mouse_init()
{
    mouse.write(0xff); // reset
    mouse.read(); // ack byte
}
```

```

    mouse.read(); // blank */
    mouse.read(); // blank */
    mouse.write(0xf0); // remote mode
    mouse.read(); // ack
    delayMicroseconds(100);
}

void setup()
{
    Serial.begin(9600);
    mouse_init();
}

/*
 * get a reading from the mouse and report it back to the
 * host via the serial line.
 */
void loop()
{
    char mstat;
    char mx;
    char my;

    /* get a reading from the mouse */
    mouse.write(0xeb); // give me data!
    mouse.read(); // ignore ack
    mstat = mouse.read();
    mx = mouse.read();
    my = mouse.read();

    /* send the data back up */
    Serial.print(mstat, BIN);
    Serial.print("\tX=");
    Serial.print(mx, DEC);
    Serial.print("\tY=");
    Serial.print(my, DEC);
    Serial.println();
    // delay(20); /* twiddle */
}

```



