Project Number: LED102897

Project Description:

A jaunty fedora ringed with high output LEDs that blink red in alternating patterns using an Arduino hidden within the hat. Designed to celebrate any holiday or party. I have worn it to a few and it is definitely an eye-grabber.

The "brains" of the hat is the Arduino microcontroller. For me the Arduino was great because it had a lot of documentation and I had worked with it before. It also required very few external components to get working so it was easy to fit inside of the hat.

My main reason for building this hat was to experiment with a multiplexing scheme that I had been reading about, charlieplexing. I had done traditional multiplexing before but charlieplexing had this appeal to me because it required so few pins. I wanted to see how well this scheme worked and especially experiment with using it to display different patterns because of the low duty cycles it had to use because it turned each LED on one by one.

Charlieplexing makes use of three state logic and the fact that there is a voltage drop across LEDs. It requires that you turn some pin high and another low and the rest go into a high impedance state to appear unconnected to the circuit. If the LEDs are arranged in the correct configuration, it allows you to individually address any LED. This allows you to use a minimum amount of pins. For traditional multiplexing schemes, you must use "2n" pins to drive "n^2" LEDs. This is good for most applications and sure beats wiring each LED individually, but, if you want to drive the most LEDs with the fewest amount of pins charlieplexing is the better choice. In a charlieplexed display you can drive "n(n-1)" LEDs with "n" pins. Because of this, charlieplexing is much more efficient the more LEDs you use. However, it does have its limitations. Because it only turns on one LED at a time, it has a very low duty cycle making driving many LEDs impractical if you are trying to make a design that relies on persistence of vision.

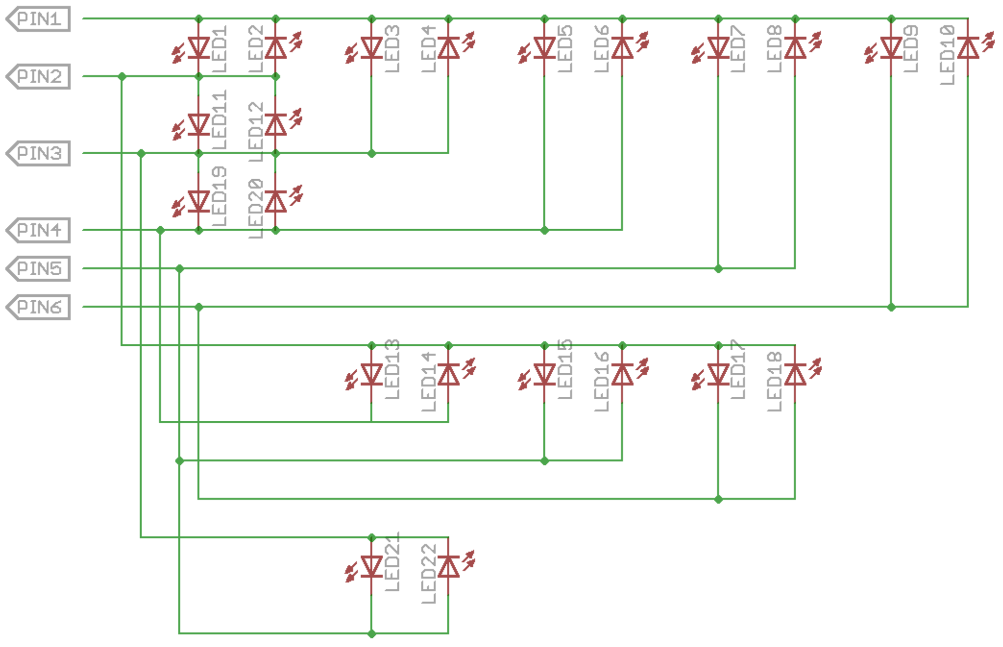
In this hat there are 22 LEDs around it. This meant that I had to use at least 6 pins. This also means that I could have driven up to 30 LEDs without much difficulty but it looked too crowded on the hat. Each LED is a high output 4 pinned red LED and is wire wrapped into place. I chose to wire wrap the LEDs in instead of soldering because I was afraid of burning the hat. To hold the Arduino, I made a printed circuit board that I designed and milled it at my school. I then soldered on the Arduino and the necessary components and attached the appropriate pins to the LEDs.

To power the hat I initially used 9V batteries, but, after I ran through a few I decided to go to the cheaper AA batteries which also have the added benefit of lasting longer. The batteries go into a voltage regulator which drops the voltage to 5V for the Arduino.

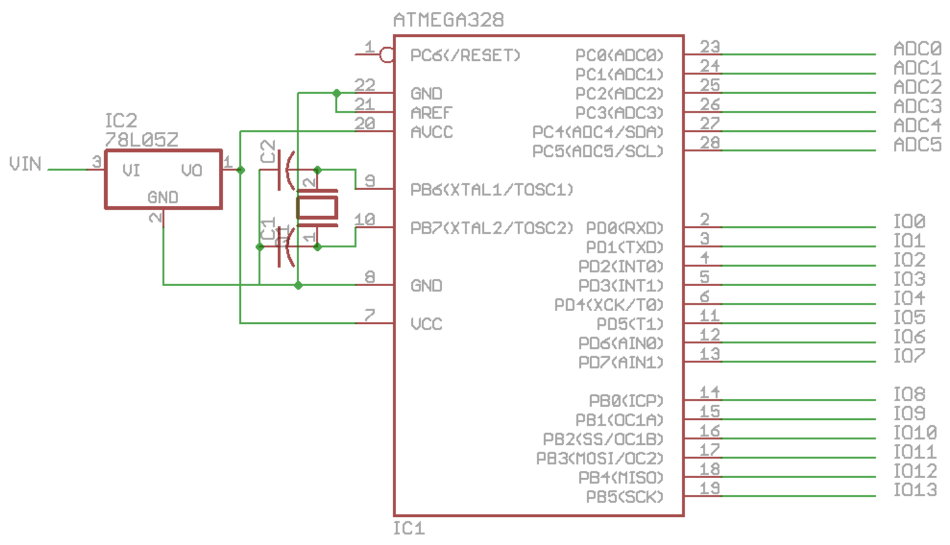
To program the Arduino I use another actual Duemilanove board which has the FTDI chip that allows communication from a USB port and program the hat Arduino by hooking the correct pins together and then uploading the sketch.

As of now the hat runs through a list of different patterns and just repeats. These patterns are: (1) a lone LED circling the hat, (2) then it speeds up, (3) all the LEDs appear to turn on and flash, (4) a snake circles the hat, (5) a download progress bar, and (6) every other LED turns on and then it appears to spin. My next step for this project is to add sensors to dictate when to switch designs.

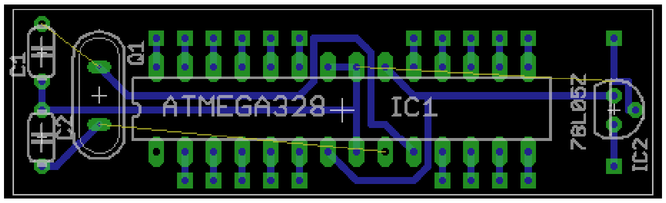
Schematic:
LED Charlieplexed Matrix:



Arduino Board Schematic:



Arduino Board Layout:



Source Code:

```
//Edward Danyliw
//Project #LED102897
//1/3/11
//Holidy LED Hat

int pin[6] = {2,3,4,5,6,7}; //the output pins used for
the charlieplexed display

void setup()  {
}

//in the main loop the hat just cycles through its
various designs and the repeats when finished
void loop()  {
  for(int y=0; y<5; y++)  { //does the circle design 5
times
    circle(250); //calls the circle method. 250ms
between each transition
  }
  speedCircle(); //same as circle but speed up each
time
  flash(5); //turns on all the LEDs at once and flashes
5 times
  snake(4,3,100); //makes a snake of length 4, circles
around 3 times and 100ms between each move
  orbit(3,100); //calls the orbit method which iterates
3 times with 100ms in between each move
  bar(250); //does the loading bar where it takes
250ms for each segment to load
  alternate(20,250); //creates a pattern where every
other LED is on and then seems to spin. 20 frames
and 250 ms on each frame
}

//methods to call pattern methods
//This method turns on the LED's one at a time
rotating around the hat. It takes an argument to
decide how long to keep each LED on
void circle(int del)  {
  for(int x=0; x<22; x++)  {
    led(x);
    delay(del);
  }
}
//This method is the same as the circle method above
except that it repeats 20 times getting faster each
iteration
void speedCircle()  {
  for(int i = 0; i<20; i++)  {
    ledOff();
    circle(200-i*10);
    if(i==19)  {
      allOn(500);
    }
  }
}
//This method turns all of the LEDs on for a certain
amount of time
void flash(int time)  {
  for(int i=0; i<time; i++)  {
    allOn(500);
    ledOff();
    delay(500);
  }
}
//turns on every other LED and then rotates to make a
spinning pattern. Frames decides how many times to
do it and time is how long on each frame
void alternate(int frames, int time)  {
   for(int i=0; i<frames; i++)  {
    halfOn(frames%2,time);
  }
}
//Snake: Makes a snake of a certain length that circles
the hat at a certain speed
void snake(int length, int circles, int time)  {
   int start=0; //defines start
   for(int i=0; i<circles; i++)  { //Repeats for how
many circles it will do
    start=0; //Set so snake starts at the 0 position
    for(int j=0; j<22; j++)  { //Increments starting
posistion until rounds the hat
      for(int d=0; d< (time/length); d++)  { //Repeats
display of snake so seems to be stationary for a time
        for(int p=start; p<(length+start); p++)  { //Draws
the snake
          int n = p;
          if(p>=22)  { //checks if the end of the snake
should wrap back to the beginning
            n=p-22; //If so, wrap
          }
      led(n);
      delay(1);
     }
    }
   start++; //Increment start position
  }
 }
}
//Loading Bar: Imitates a download progress bar.
Time says how long for each segment to load
void bar(int time)  {
 for(int i=0; i<22; i++)  {  //goes until all LEDs are
on
  for(int j=0; j< (time/(i+1)); j++)  { //holds LEDs on
for specified time
   for(int k=0; k<=i; k++)  {  //turn on correct LEDs
    led(k);
    delay(1);
   }
  }
 }
}
//Orbit: Two LEDs appear to orbit the hat in opposite
directions. Num defines number of orbits and time
defines how long the LED is on before incrementing
void orbit(int num,int time)  {
 for(int n=0; n<num; n++)  {  //orbits n times
  for(int i=0; i<22; i++)  {
   for(int j=0; j< (time/2); j++)  {
    led(i);
    delay(1);
    led(22-i);
    delay(1);
   }
  }
 }
}

//control methods
//This method turns on half of the LEDs for a certain
amount of time
void halfOn(int choice, int time)  {
 switch(choice)  {
  case 0:
  for(int i=0; i<time/11; i++)  {
   for(int j=0; j<22; j=j+2)  {
    led(j);
    delay(1);
   }
  }
  case 1:
  for(int i=0; i<time/11; i++)  {
   for(int j=1; j<22; j=j+2)  {
    led(j);
    delay(1);
   }
  }
 }
}
//Turns on all the LEDs
void allOn(int length)  {
 for(int i=0; i<length; i++)  {
  circle(0); //Calls the circle method with no delay in
between each LED
 }
}
//Turns off the LEDs by setting the outputs to High
Impedance
void ledOff()  {
 led(100); //Calls a high number out of the range of
acceptable values which puts all pins into a High
Impedance state
}
//Turns on the specified LED
void led(int loc)  {
 switch(loc)  {
  case 0:
   //In this block of code I select which pins are
going to be used and set those to outputs. The others
are set to inputs so they are High Impedance.
   pinMode(pin[0],OUTPUT);
   pinMode(pin[1],OUTPUT);
   pinMode(pin[2],INPUT);
   pinMode(pin[3],INPUT);
   pinMode(pin[4],INPUT);
   pinMode(pin[5],INPUT);
   //In this block the states of the pins are selected. It
is important that only one is set to High because even
though the others may be inputs,
   //if set to High, they will enable an internal pull
up resistor possible of inadvertantly turning on an
LED
   digitalWrite(pin[0],HIGH);
   digitalWrite(pin[1],LOW);
   digitalWrite(pin[2],LOW);
   digitalWrite(pin[3],LOW);
   digitalWrite(pin[4],LOW);
   digitalWrite(pin[5],LOW);
  break; //exits the switch loop

case 1:
  pinMode(pin[0],OUTPUT);
  pinMode(pin[1],OUTPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],HIGH);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 2:
  pinMode(pin[0],OUTPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],OUTPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],HIGH);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 3:
  pinMode(pin[0],OUTPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],OUTPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],HIGH);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 4:
 pinMode(pin[0],OUTPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],OUTPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],HIGH);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

 case 5:
  pinMode(pin[0],OUTPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],OUTPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],HIGH);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

 case 6:
  pinMode(pin[0],OUTPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],OUTPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],HIGH);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;
```

```
case 7:
  pinMode(pin[0],OUTPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],OUTPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],HIGH);
  digitalWrite(pin[5],LOW);
  break;

case 8:
  pinMode(pin[0],OUTPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],OUTPUT);

  digitalWrite(pin[0],HIGH);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 9:
  pinMode(pin[0],OUTPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],OUTPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],HIGH);
  break;

case 10:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],OUTPUT);
  pinMode(pin[2],OUTPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],HIGH);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 11:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],OUTPUT);
  pinMode(pin[2],OUTPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],HIGH);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 12:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],OUTPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],OUTPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],HIGH);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 13:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],OUTPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],OUTPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],HIGH);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 14:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],OUTPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],OUTPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],HIGH);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 15:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],OUTPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],OUTPUT);
  pinMode(pin[5],INPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],HIGH);
  digitalWrite(pin[5],LOW);
  break;

case 16:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],OUTPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],OUTPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],HIGH);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 17:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],OUTPUT);
  pinMode(pin[2],INPUT);
  pinMode(pin[3],INPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],OUTPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],LOW);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],HIGH);
  break;

case 18:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],OUTPUT);
  pinMode(pin[3],OUTPUT);
  pinMode(pin[4],INPUT);
  pinMode(pin[5],OUTPUT);

  digitalWrite(pin[0],LOW);
  digitalWrite(pin[1],LOW);
  digitalWrite(pin[2],HIGH);
  digitalWrite(pin[3],LOW);
  digitalWrite(pin[4],LOW);
  digitalWrite(pin[5],LOW);
  break;

case 19:
  pinMode(pin[0],INPUT);
  pinMode(pin[1],INPUT);
  pinMode(pin[2],OUTPUT);
```

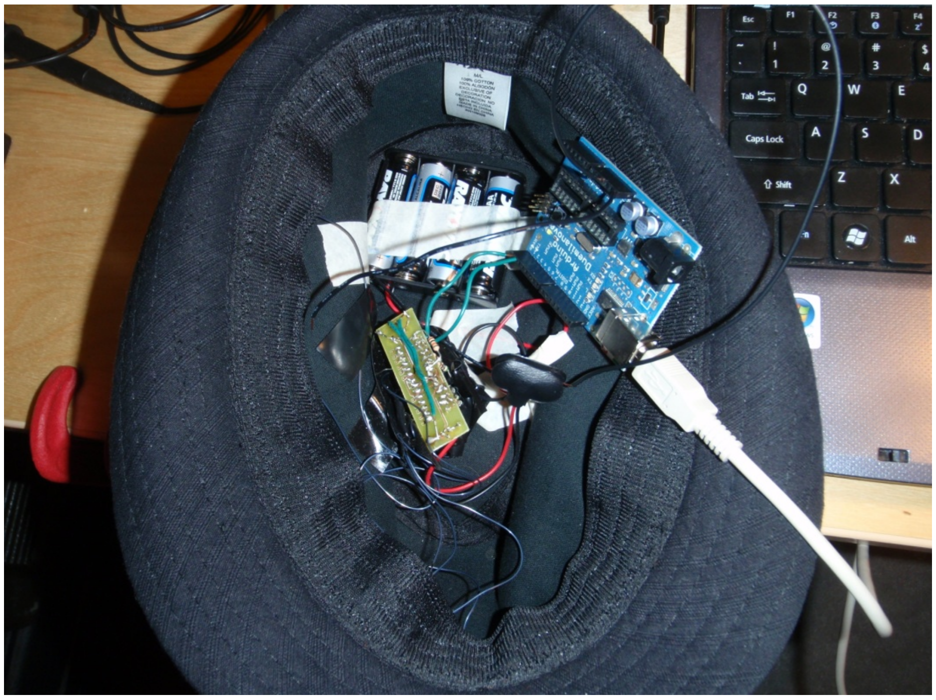
```
      pinMode(pin[3],OUTPUT);
      pinMode(pin[4],INPUT);
      pinMode(pin[5],INPUT);

      digitalWrite(pin[0],LOW);
      digitalWrite(pin[1],LOW);
      digitalWrite(pin[2],LOW);
      digitalWrite(pin[3],HIGH);
      digitalWrite(pin[4],LOW);
      digitalWrite(pin[5],LOW);
      break;

    case 20:
      pinMode(pin[0],INPUT);
      pinMode(pin[1],INPUT);
      pinMode(pin[2],OUTPUT);
      pinMode(pin[3],INPUT);
      pinMode(pin[4],OUTPUT);
      pinMode(pin[5],INPUT);

      digitalWrite(pin[0],LOW);
      digitalWrite(pin[1],LOW);
      digitalWrite(pin[2],HIGH);
      digitalWrite(pin[3],LOW);
      digitalWrite(pin[4],LOW);
      digitalWrite(pin[5],LOW);
      break;

    case 21:
      pinMode(pin[0],INPUT);
      pinMode(pin[1],INPUT);
      pinMode(pin[2],OUTPUT);
      pinMode(pin[3],INPUT);
      pinMode(pin[4],OUTPUT);
      pinMode(pin[5],INPUT);

      digitalWrite(pin[0],LOW);
      digitalWrite(pin[1],LOW);
      digitalWrite(pin[2],LOW);
      digitalWrite(pin[3],LOW);
      digitalWrite(pin[4],HIGH);
      digitalWrite(pin[5],LOW);
      break;

    default: //If a value not covered by the cases 
provided, it will put all pins into a High Impedance 
state
      pinMode(pin[0],INPUT);
      pinMode(pin[1],INPUT);
      pinMode(pin[2],INPUT);
      pinMode(pin[3],INPUT);
      pinMode(pin[4],INPUT);
      pinMode(pin[5],INPUT);

      digitalWrite(pin[0],LOW);
      digitalWrite(pin[1],LOW);
      digitalWrite(pin[2],LOW);
      digitalWrite(pin[3],LOW);
      digitalWrite(pin[4],LOW);
      digitalWrite(pin[5],LOW);
      break;
  }
}
```

Bill of Materials:

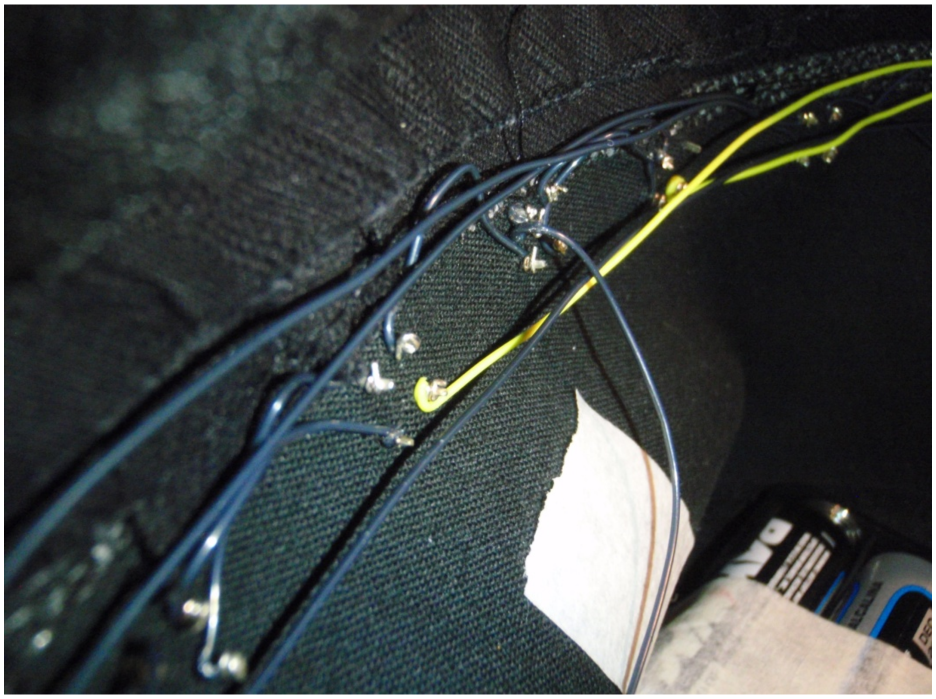
1- ATMEGA328
2- 22nF Ceramic Capacitors
1- 16 MHz Crystal
1- Printed Circuit Board
22- High Output Red LEDs (Jameco #**1952222**)
2- 9V Battery Clip
1- 4 x AA Battery Holder

Pictures:

Programming the hat:



The wire wrapped leads of the LEDs:



The hat as seen from the outside: