Arduino

AS220 Workshop

Part III – *Multimedia Applications*

Lutz Hamel

hamel@cs.uri.edu

www.cs.uri.edu/~hamel/as220

AS220
LABS
SOUND
PIXEL GRIDS
MOVING PIXELS
FOLK ELECTRONICS

Basic Building Blocks

- The basic building blocks for Arduino interactive object(s) are:
  - Digital Input (pushbutton)
  - Analog Input (pot)
  - Digital Output (switching an LED)
  - Analog Output (PWM signal)
  - Serial/USB Communication
    - this is the one thing we haven't really explored

Arduino

Multimedia Applications

- Idea:
  - the Arduino becomes a specialized sensor/actuator for an application running on a multimedia capable computer
  - we use the USB connectivity to communicate between the Arduino and the MM computer

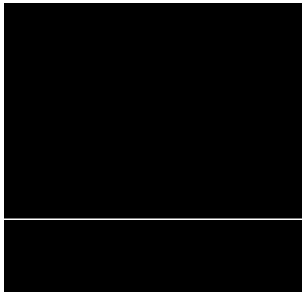
Processing

- Processing:
  - is a programming environment tailored for multimedia applications
  - programs are also called sketches and are the counterpart to Arduino sketches
  - is open-source software
    - www.processing.org

Arduino

Processing

```
// Processing Sketch - draw a moving line
// the y coordinate of our line
int y = 100;

// The statements in the setup() function
// execute once when the program begins
void setup()
{
  size(200, 200);  // Size should be the first statement
  stroke(255);     // Set line drawing color to white
  frameRate(30);   // 30 frames per second
}

// The statements in draw() are executed until the
// program is stopped. Each statement is executed in
// sequence and after the last line is read, the first
// line is executed again.
void draw()
{
  background(0);   // Set the background to black
  y = y - 1;
  if (y < 0) { y = height; }
  line(0, y, width, y);
}
```



- Sketches consist of two sections:
  - setup() - runs once
  - draw() - loops cont.
- Graphics primitives are supported
  - canvas
  - lines
  - points
  - circles, etc
- Many libraries
  - serial library!

Arduino

Processing

```
// based on an example by Tom Igoe

import processing.serial.*;

Serial myPort;  // The serial port

void setup() {
  // List all the available serial ports
  println(Serial.list());
  // I know that the first port in the serial list on my mac
  // is always my  Keyspan adaptor, so I open Serial.list()[0]
  // Open whatever port is the one you're using.
  myPort = new Serial(this, Serial.list()[0], 9600);

}

void draw() {
  if (myPort.available() > 0) {
    int inByte = myPort.read();
    println(inByte);
  }
}
```

- In this example we set up a 9600 baud serial communication on an appropriate port
- We use the draw() function that loops continuously to check for bytes on the serial line
- If we find bytes we print them

Arduino

Processing

```
// Example by Tom Igoe

import processing.serial.*;

Serial myPort;     // The serial port:
PFont myFont;      // The display font:
String inString;   // Input string from serial port:
int lf = 10;       // ASCII linefeed

void setup() {
  size(400,200);
  // Make your own font. It's fun!
  myFont = createFont("Times-Roman",32,true);
  // register our font
  textFont(myFont, 18);
  // open the serial port
  myPort = new Serial(this, “COM6”, 9600);
  // read bytes until linefeed
  myPort.bufferUntil(lf);
}

void draw() {
  background(0);
  text("received: " + inString, 10,50);
}

void serialEvent(Serial p) {
  inString = p.readString();
}
```

- In this example we set up a 9600 baud serial communication on an appropriate port
- Here we use the serialEvent() function to to read a string from the serial line
- serialEvent() is called an *event handler* and is called when a string is available on the serial line

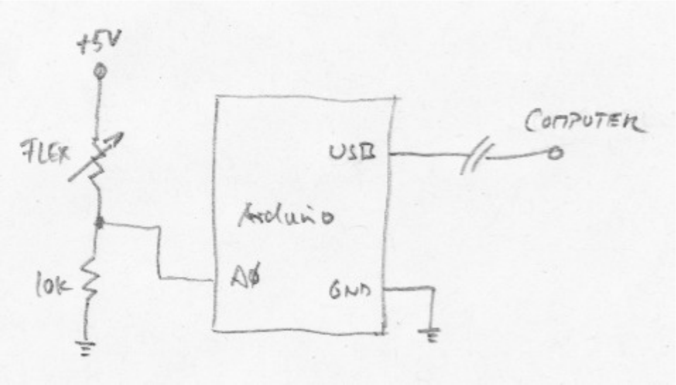
Listing the Serial Ports

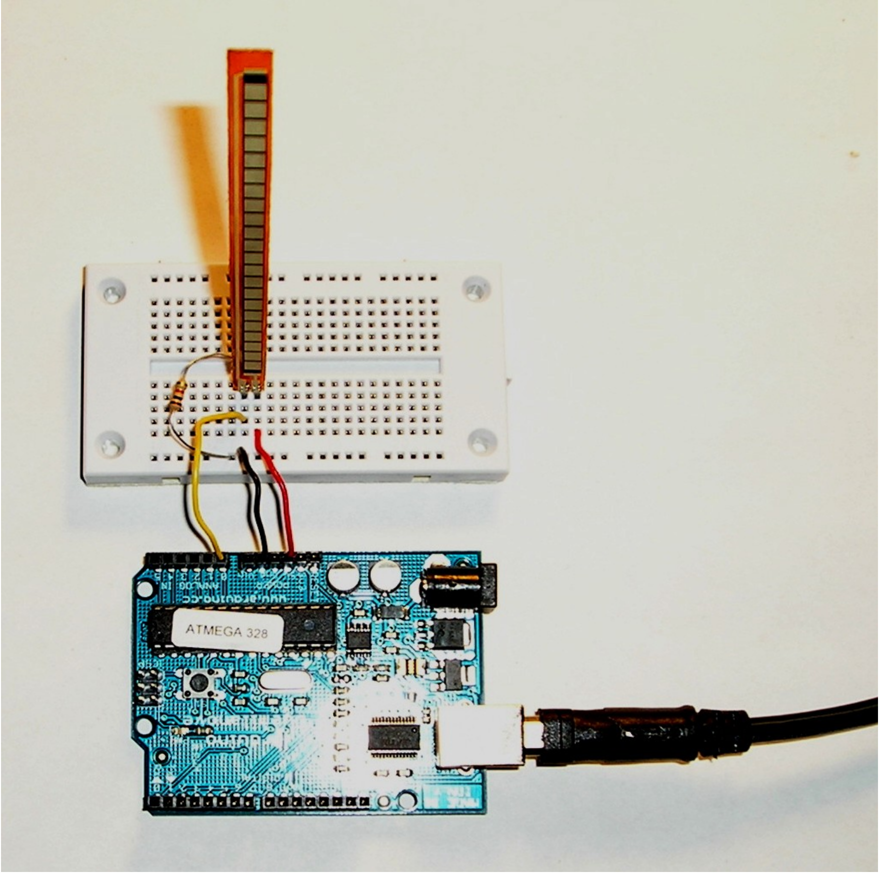
```
/*
 * A simple sketch to find out to which port
 * the arduino is connected.
 */

import processing.serial.*;

void setup()
{
  println(Serial.list());
}

void draw()
{
  exit();
}
```

Squish the Circle

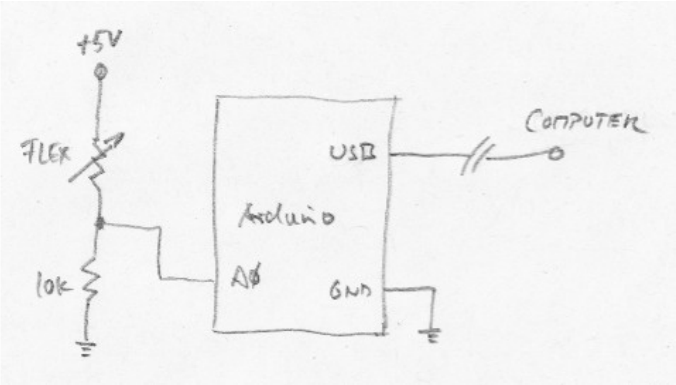
- Idea:
  - Use the flex sensor to generate an analog signal for the Arduino
  - Transmit the digitized signal from the Arduino to the computer
  - Read the value in Processing
  - Use the value to scale the y-dimension of a circle

Squish the Circle





Squish the Circle



Note: Here we do some preprocessing on the Arduino in order to reduce noise. We keep a running average of the last five measurement points.

```
/*
 * Squish the circle - Arduino Sketch
 * reads the value of a flex-sensor on analog input pin 0
 * and then writes a value scaled to between 0 and 100 to
 * the serial line.
 */

int analogPin = 0;
int val = 0;
int values[5] ={0,0,0,0,0};
int average = 0;

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

void loop() {
  val = analogRead(analogPin);
  val = map(val,200,550,0,100);
  // Shift over the existing values to make room for the new one.
  for (int i = 0; i < 4; i++){
    values[i] = values[i + 1];
  }
  // Add the received value to the end of the array.
  values[4] = val;
  // compute running average
  int sum = 0;
  for (int i = 0; i < 5; i++){
    sum += values[i];
  }
  average = sum/5;
  // write the data as a single byte
  Serial.print(average,BYTE);
  delay(100);
}
```

Arduino

Squish the Circle

```
/*
 * Squish the Circle - Processing Sketch
 * Read data from the arduino board on the
 * serial line, we expect integer values
 * between 0 and 100. We scale these values
 * to between 0 and 1 and use the scaled
 * values to distort a circle drawn on the
 * canvas.
 */

import processing.serial.*;
// Create a serial port
Serial myPort;
// the is the value we use to distort the circle
float distort = 1;

void setup() {
  // set up the canvas
  size(400, 400);
  // Set the color used to fill shapes.
  fill(255);
  // frame rate of the draw() function
  frameRate(30);
  // Draw with smooth (anti-aliased) edges
  smooth();
  // set up our serial port, on my PC the Arduino board
  // always shows up on COM6, insert the appropriate
  // portname.  The speed must match the speed set up
  // on the Arduino board.
  myPort = new Serial(this, "COM6", 9600);
}
```

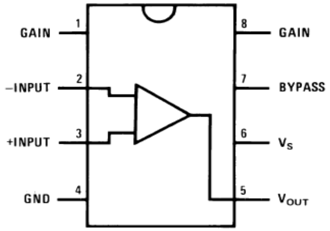
```
void draw() {
  // In the draw() function, the background color is
  // used to clear the display window at the beginning
  // of each frame
  background(204);
  // display the circle
  ellipse(200,200,300,300*distort);
}

// the serialEvent function is called every time there
// is data available on the serial line
void serialEvent(Serial p) {
  // we read integer values between 0 and 100
  // scale to between 0 and 1
  distort = p.read()/100.0;
}
```

Visualizing Sound

- Idea:
  - Use a *microphone* to capture sound
  - Digitize the analog signal
  - Send the digitized signal to the computer for visualization with Processing

- Caveat:
  - The signal from the microphone is too weak, use an *amplifier* to increase the signal for good resolution on the A/D converter

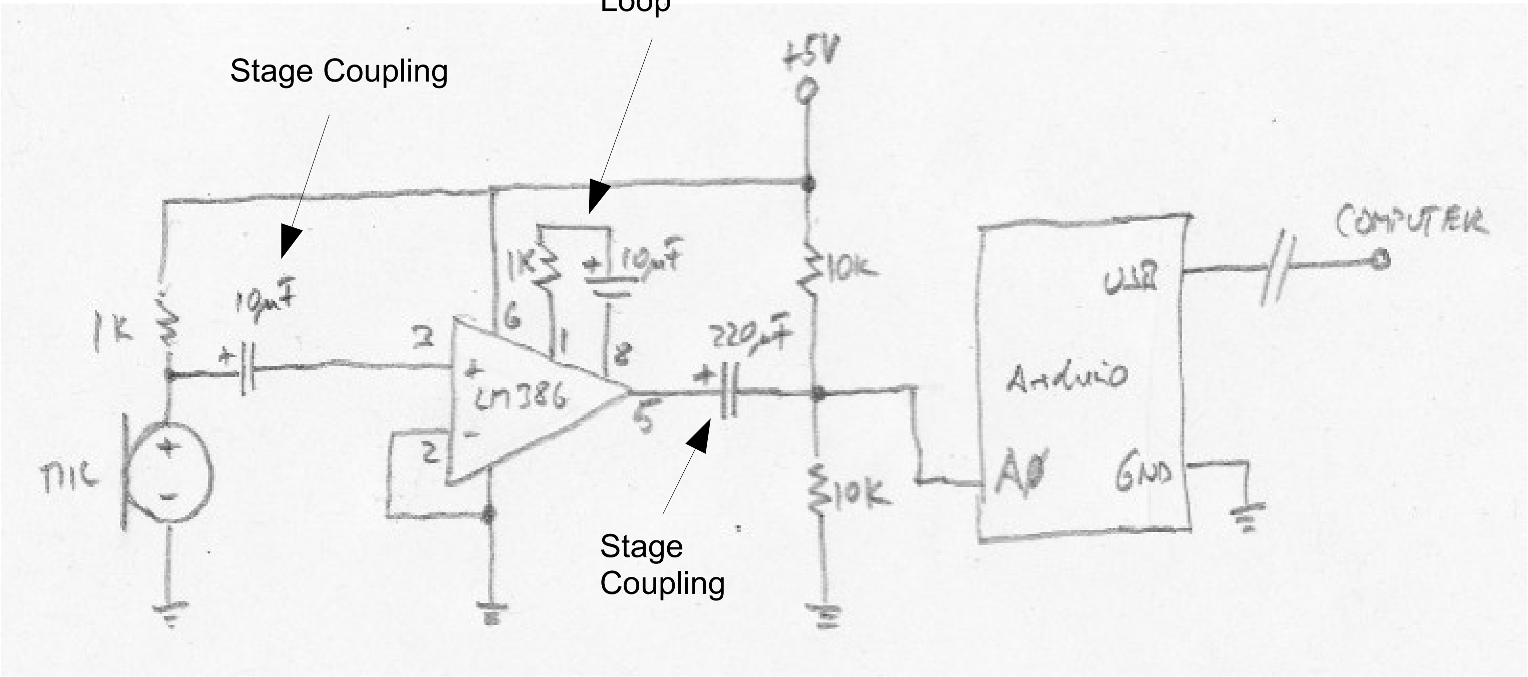
The Circuit



LM386

Amplifier Feedback Loop

Stage Coupling



Stage Coupling

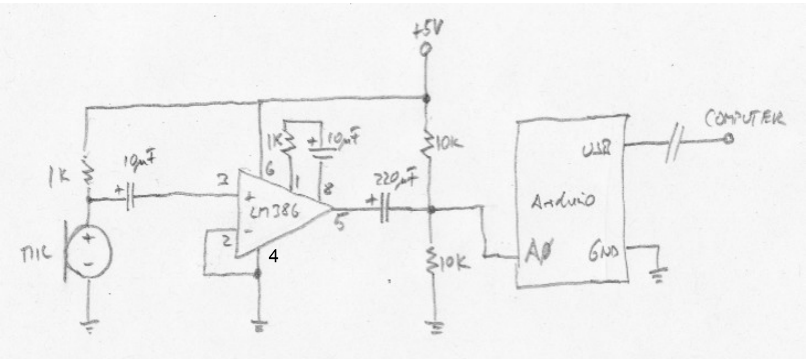
Input Signal Generation

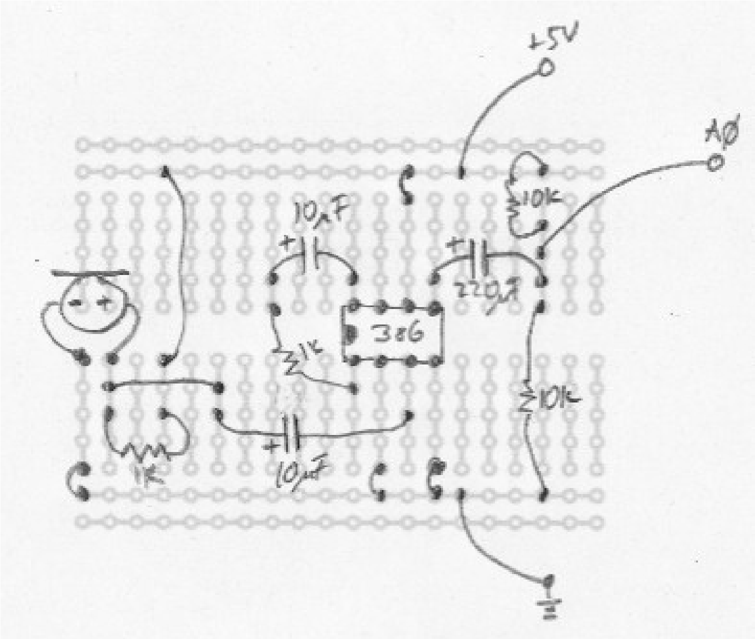
Amplification

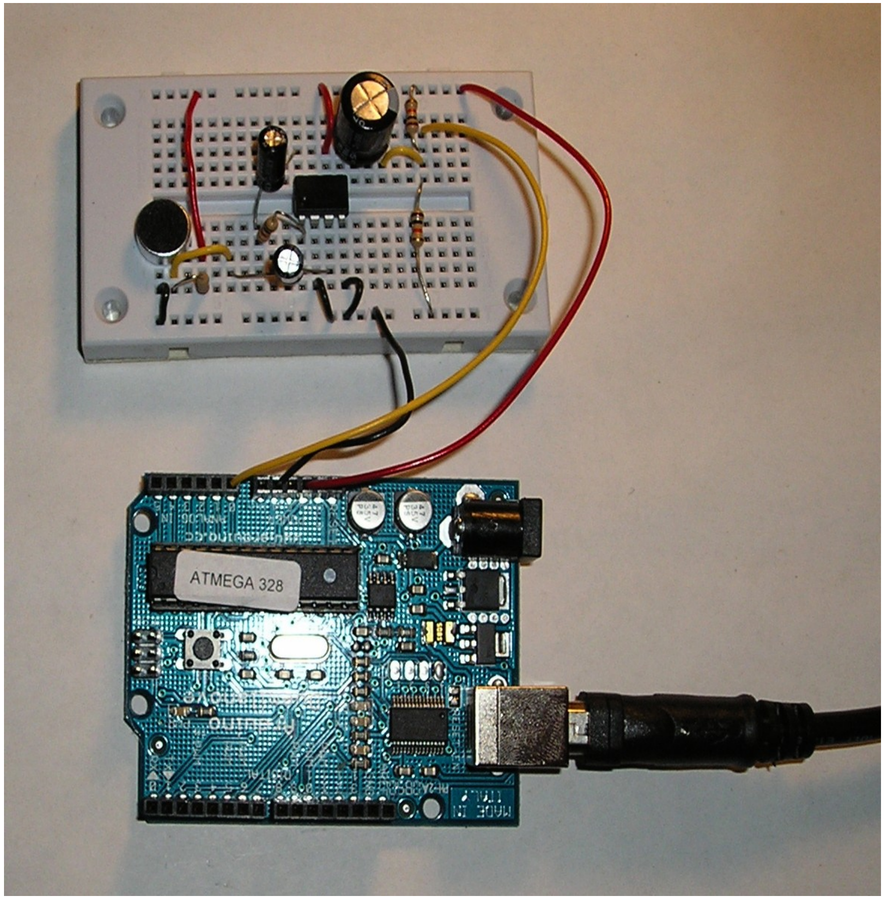
Signal Balancing (no signal at 2.5V)

Digitization

Visualizing Sound



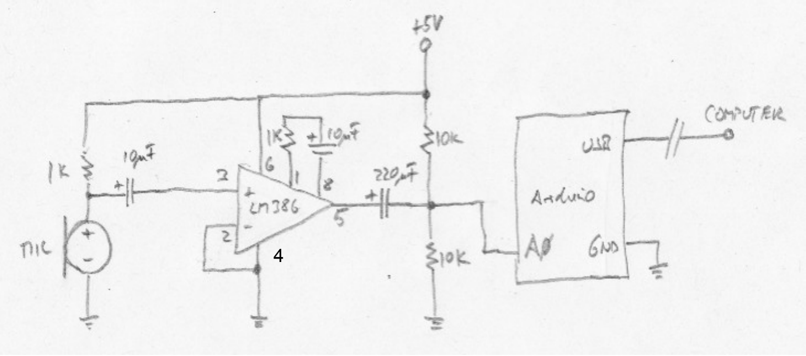




Arduino

Visualizing Sound

```
// visualizing sound - Arduino Sketch
// read input from the amplified mic
// and send the value to the multi-media
// computer for visualization

int soundinPin = 0;
int val = 0;

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

void loop() {
  val = analogRead(soundinPin);
  val = map(val,0,1023,0,255);
  Serial.print(val,BYTE);
  delay(8);
}
```



```
// Visualize sound - Processing sketch
// visualize input from the arduino board
// based on Graph by David A. Mellis

import processing.serial.*;

Serial myPort;
int inputByte = 0;
// Store the last 64 values received so we can graph them.
int[] values = new int[64];

void setup()
{
  size(512, 256);
  frameRate(120);
  myPort = new Serial(this, "COM6", 9600);
}

void draw()
{
  background(53);
  stroke(255);

  // Graph the stored values by drawing lines between them.
  for (int i = 0; i < 63; i++){
    line(i*8,255-values[i],(i+1)*8,255-values[i+1]);
    // Shift over the existing values to make room
    values[i] = values[i+1];
  }
  if (myPort.available() > 0) {
    inputByte = myPort.read();
    values[63] = inputByte;
  }
}
```

Visualizing Sound

- Problems:
  - A big problem with this application is that the display has very low bandwidth
    - On most computers you can probably not achieve a frame rate higher than 240 frames per second
    - This means we sample our sound wave every 4 msec (or sample freq=240Hz)
    - This implies that the maximum frequency that we can visualize *without distortion* is 120Hz, not very useful

Visualizing Sound

- Idea:
  - Instead of visualizing the sound wave, visualize the composition of sound in terms of frequencies
    - Fast-Fourier Transform (FFT)
  - In this case we turn our Arduino board into a DSP chip
  - However, the code for this is too complex to present here, watch for it on the arduino mailing list.

The Clapper

- Idea:
  - Leave the hardware as is but we change the software
  - If we hear a loud noise send a signal to the computer
  - On the computer the signal determines how fast a line rises on a display

Arduino

The Clapper

```
// The Clapper - Arduino Sketch
// read input from the amplified mic,
// if you hear a loud sound
// send a signal to the computer

// read the sound input on analog pin 0
int soundinPin = 0;
// the sound threshold above which we
// consider a sound to be loud
int threshold = 700;
// the value read from the soundinput
int val = 0;

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

void loop() {
  val = analogRead(soundinPin);
  if (val >= threshold) {
    Serial.print(1,BYTE);
    // debounce
    delay(10);
  }
}
```

Note: Good example of signal thresholding.

Note: Clapping produces echoes etc, we *debounce* our signal by waiting until the echoes are gone so we don't accidentally react to the echoes.

```
// The Clapper - Processing Sketch
// Every time we receive an event on the serial
// port we let the line rise faster up to a certain
// value and then we start again.

import processing.serial.*;
Serial myPort;
int inputByte = 0;
float y = 100;
int increment = 1;

void setup()
{
  size(200, 400);  // Size should be the first statement
  stroke(255);     // Set stroke color to white
  frameRate(60);
  // need to pick the right com port
  myPort = new Serial(this, "COM6", 9600);
}

void draw()
{
  background(0);   // Set the background to black
  line(0, y, width, y);
  y = y - increment;
  if (y < 0) {
    y = height;
  }
}

void serialEvent(Serial p) {
  // remove the byte from the serial port
  inputByte = p.read();
  // speed up the line
  increment = (increment + 2) %  8;
  println(increment);
}
```

Other Libraries

- Processing has many other libraries
  - Quicktime
  - Network
    - The “Getting Started...” book has a great example of this in “Talking to the Cloud”
  - Sound
  - OpenGL
    - Sophisticated rendering possible

Next Week

- Next week is our last class
  - We will look at communication
    - RS232, MIDI
    - We will build our own Arduino remote control using infrared LEDs and Phototransistors
  - Show and tell of your projects
  - General Q&A