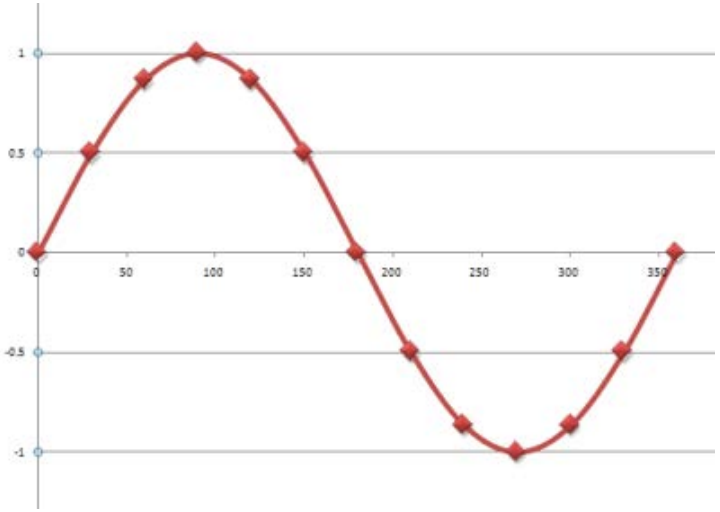
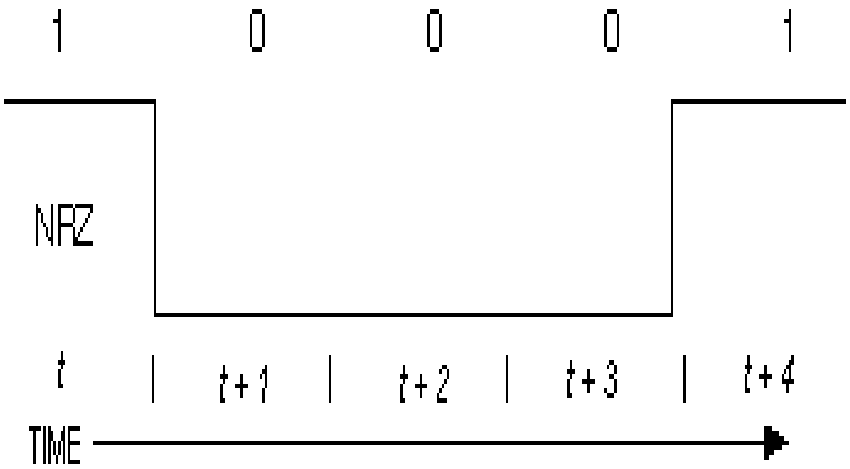
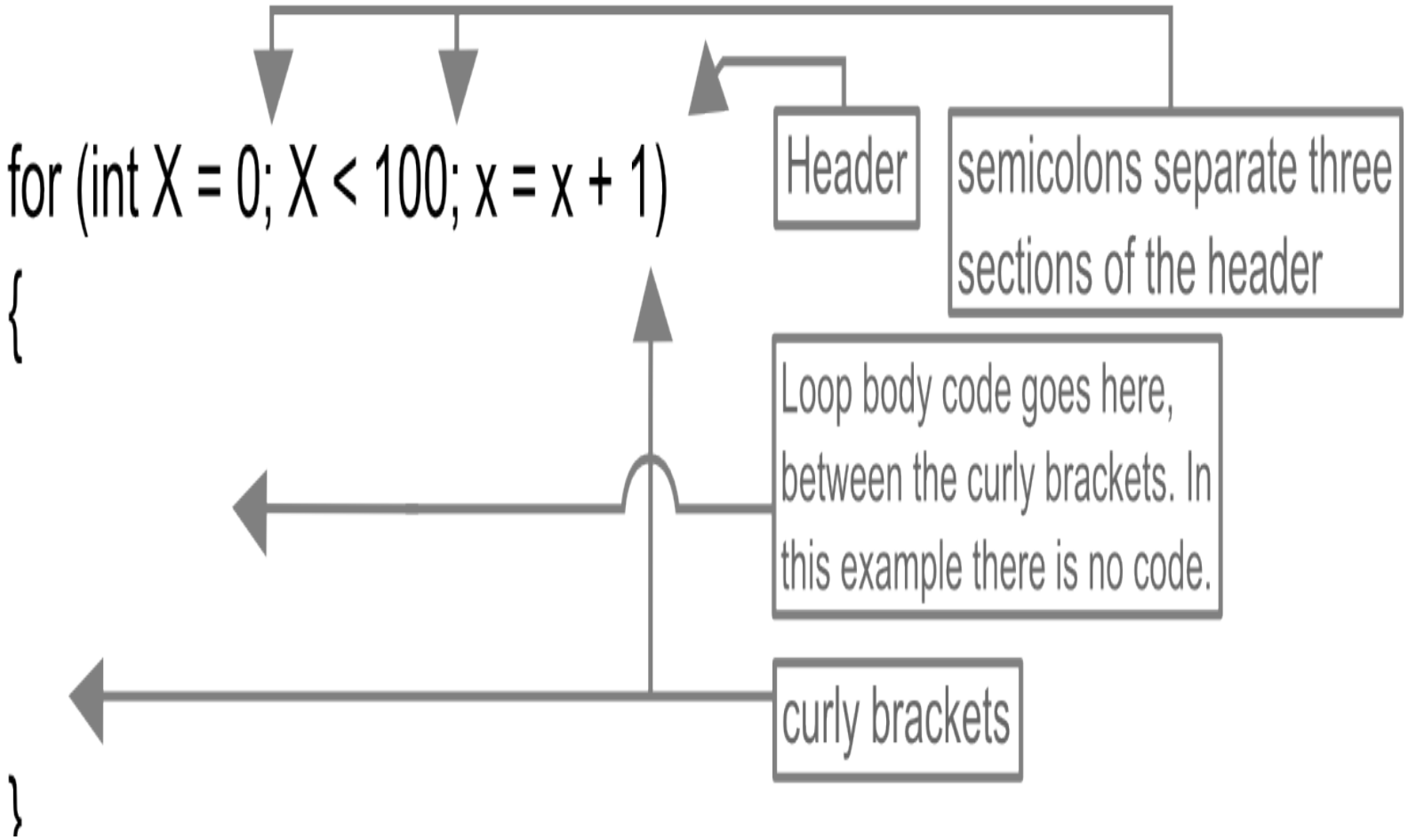
Inventing… 5

with Software and Electronics

PWM


1
0.5
0
-0.5
-1
0
50
100
150
200
250
300
350


1
0
0
0
1
NRZ
t
t+1
t+2
t+3
t+4
TIME


for (int X = 0; X < 100; x = x + 1)
{
}
Header
semicolons separate three sections of the header
Loop body code goes here, between the curly brackets. In this example there is no code.
curly brackets

For Loop

```
variableN = 0;

while (variableN < 10)      {

variableN = variableN + 1;

}
```

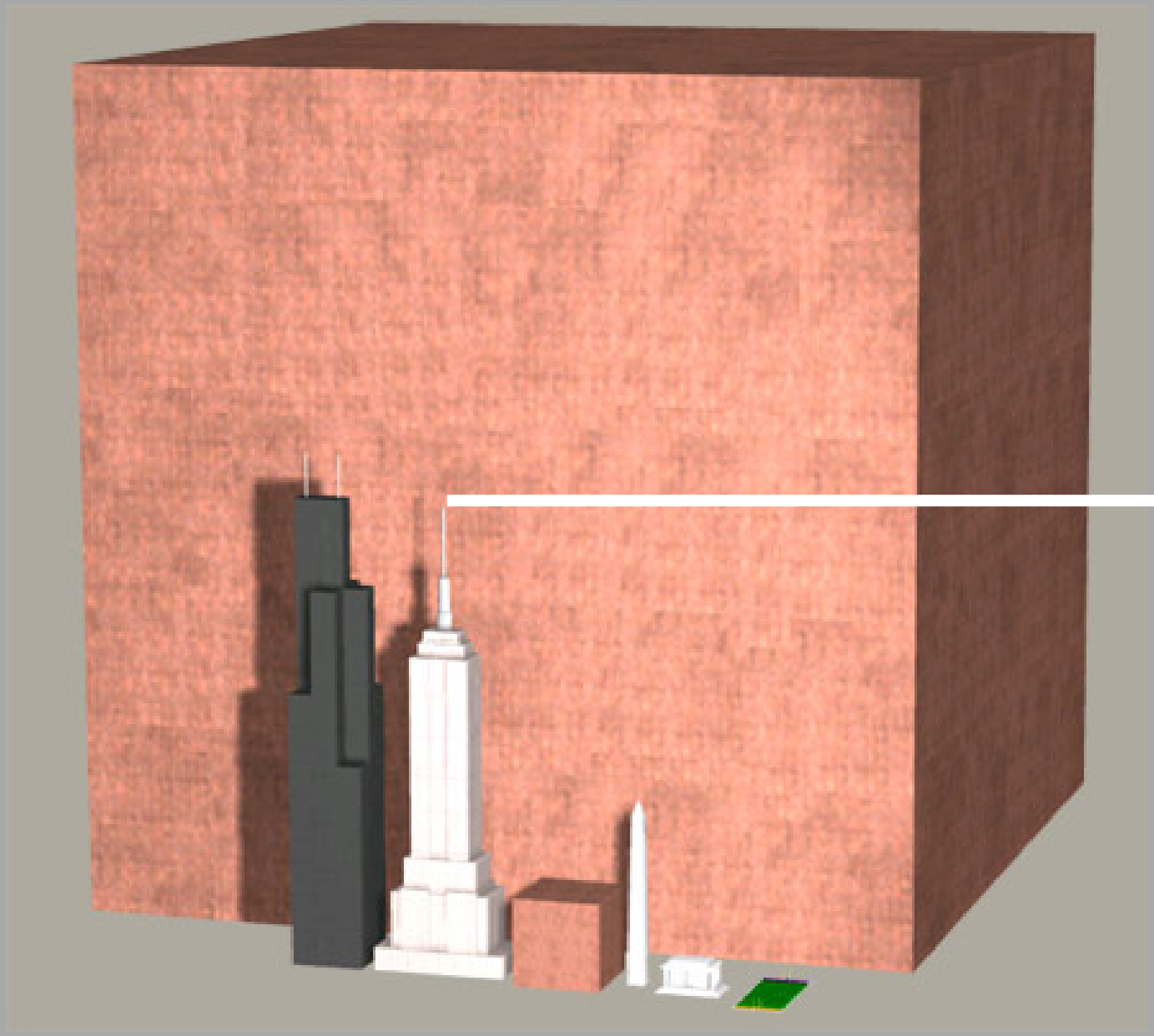
This is not a part of the while loop, it just sets variableN equal to zero before the while loop happens.

Header

Loop body code goes here, between the curly brackets. In this example it changes variableN so you're not stuck in the while loop forever.

curly brackets

The while loop

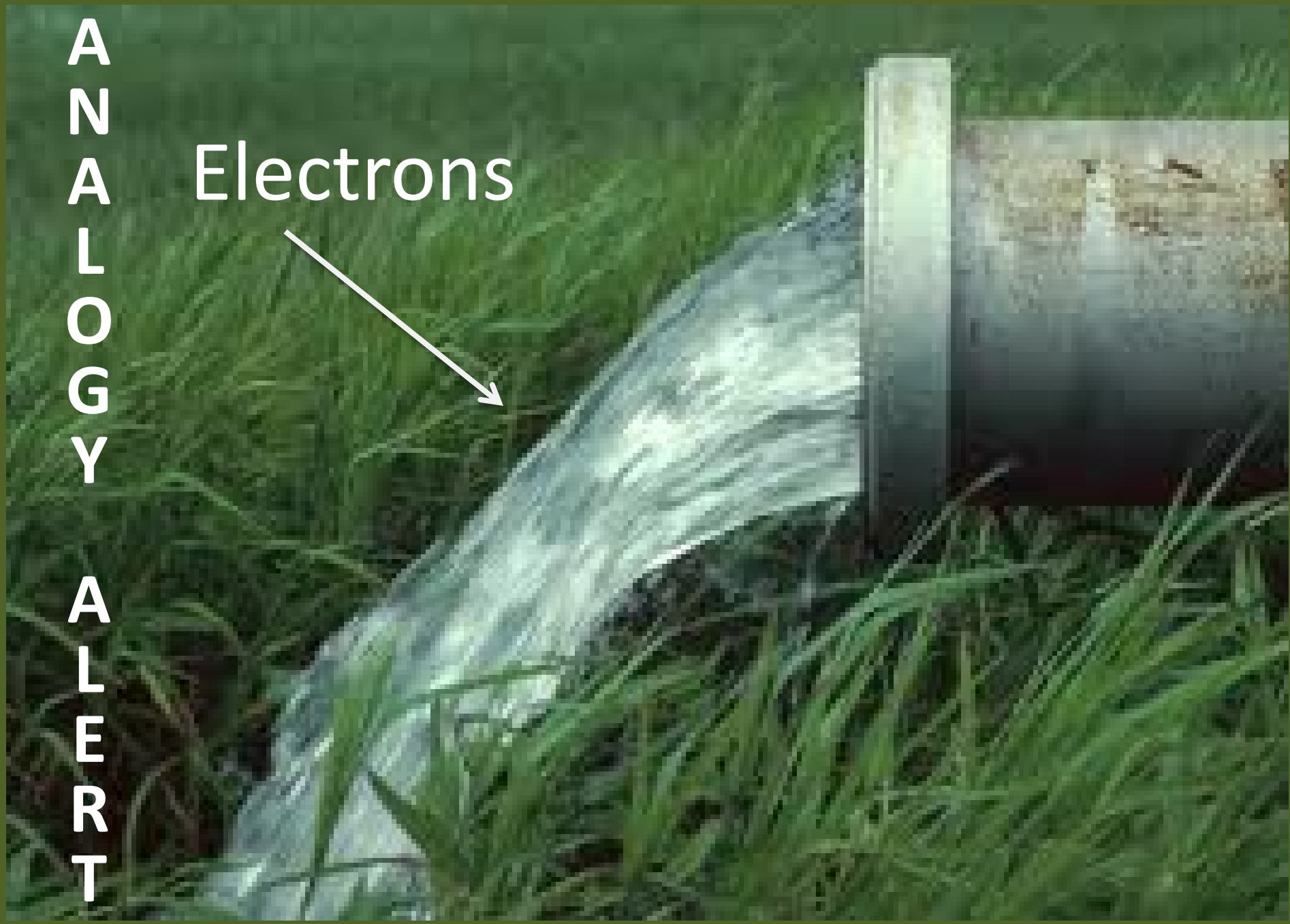


Empire State Building
102 Stories

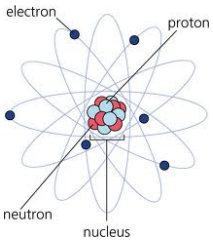
Or – a pile of pennies
986,426,768 Miles
High….

ANALOGY ALERT

Conductor (Wire)


ANALOGY ALERT
Electrons

ANALOGY ALERT



Current (Amps)

Arduino Uno Datasheet

Summary

Microcontroller	ATmega328
Operating Voltage	5V
Input Voltage (recommended)	7-12V
Input Voltage (limits)	6-20V
Digital I/O Pins	14 (of which 6 provide PWM output)
Analog Input Pins	6
DC Current per I/O Pin	40 mA
DC Current for 3.3V Pin	50 mA
Flash Memory	32 KB (ATmega328) of which 0.5 KB used by bootloader
SRAM	2 KB (ATmega328)
EEPROM	1 KB (ATmega328)

Circuit Happiness

Dead

Stressed

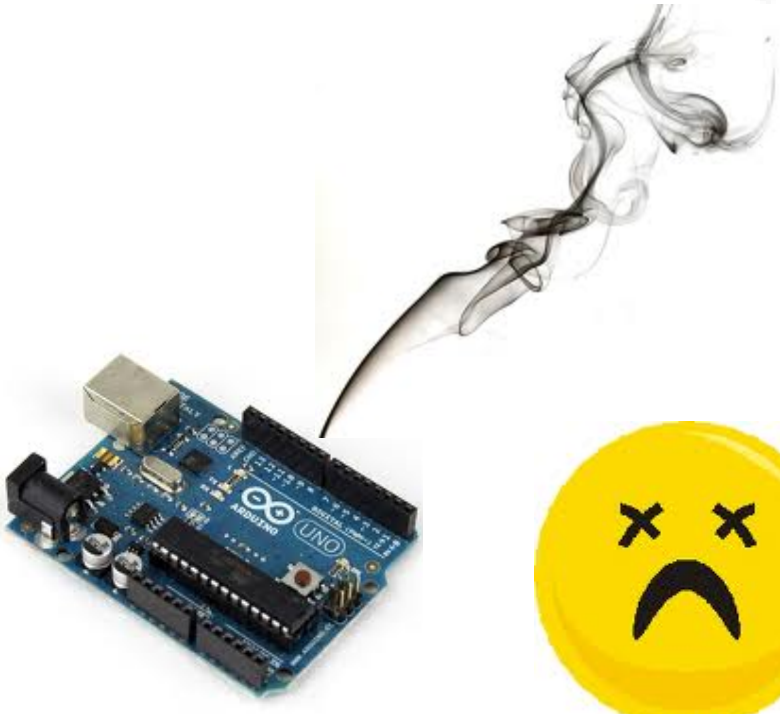
Healthy

Underpowered

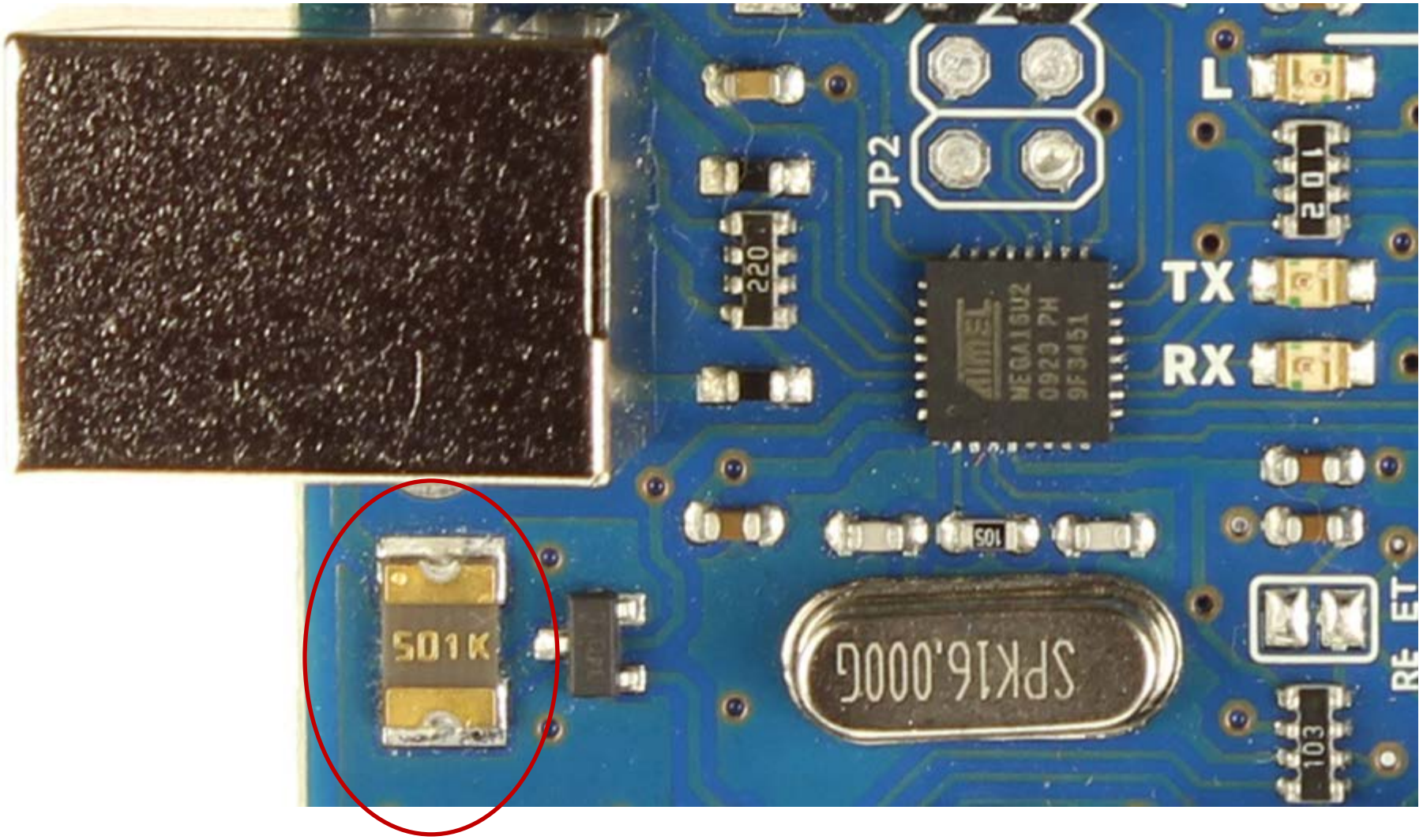
Asleep

ANALOGY ALERT


SMOKE ALARM


ARDUINO
UNO


119
119



Polyfuse (500mA)

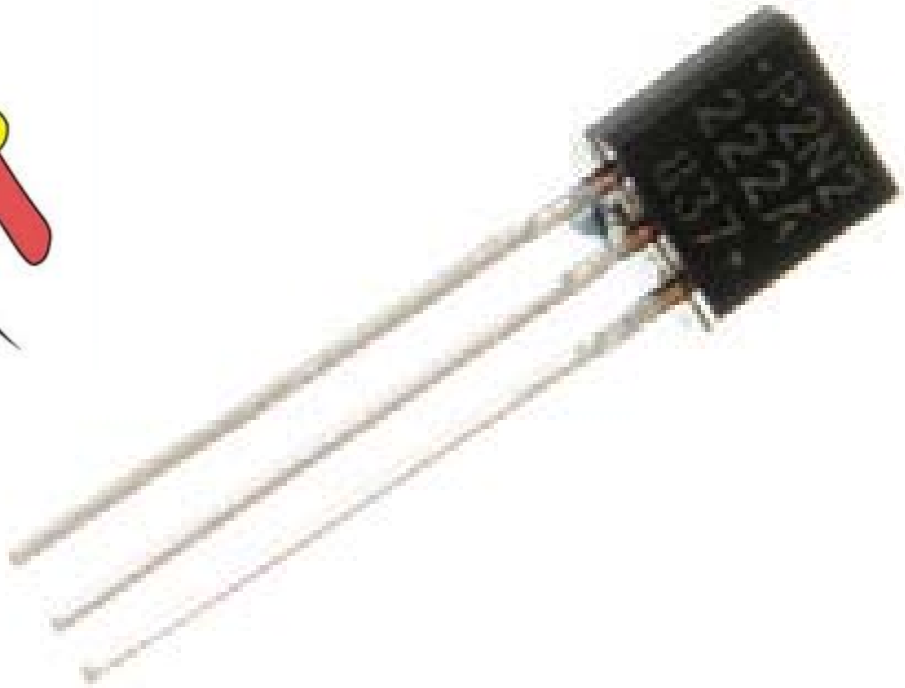
Ways to Kill an Arduino

Easily Possible

Shorting I/O Pins to Ground
Apply Overvoltage to I/O Pins
Shorting I/O Pins to Each Other
Exceed Total Microcontroller Current (200mA)

NO SMOKING

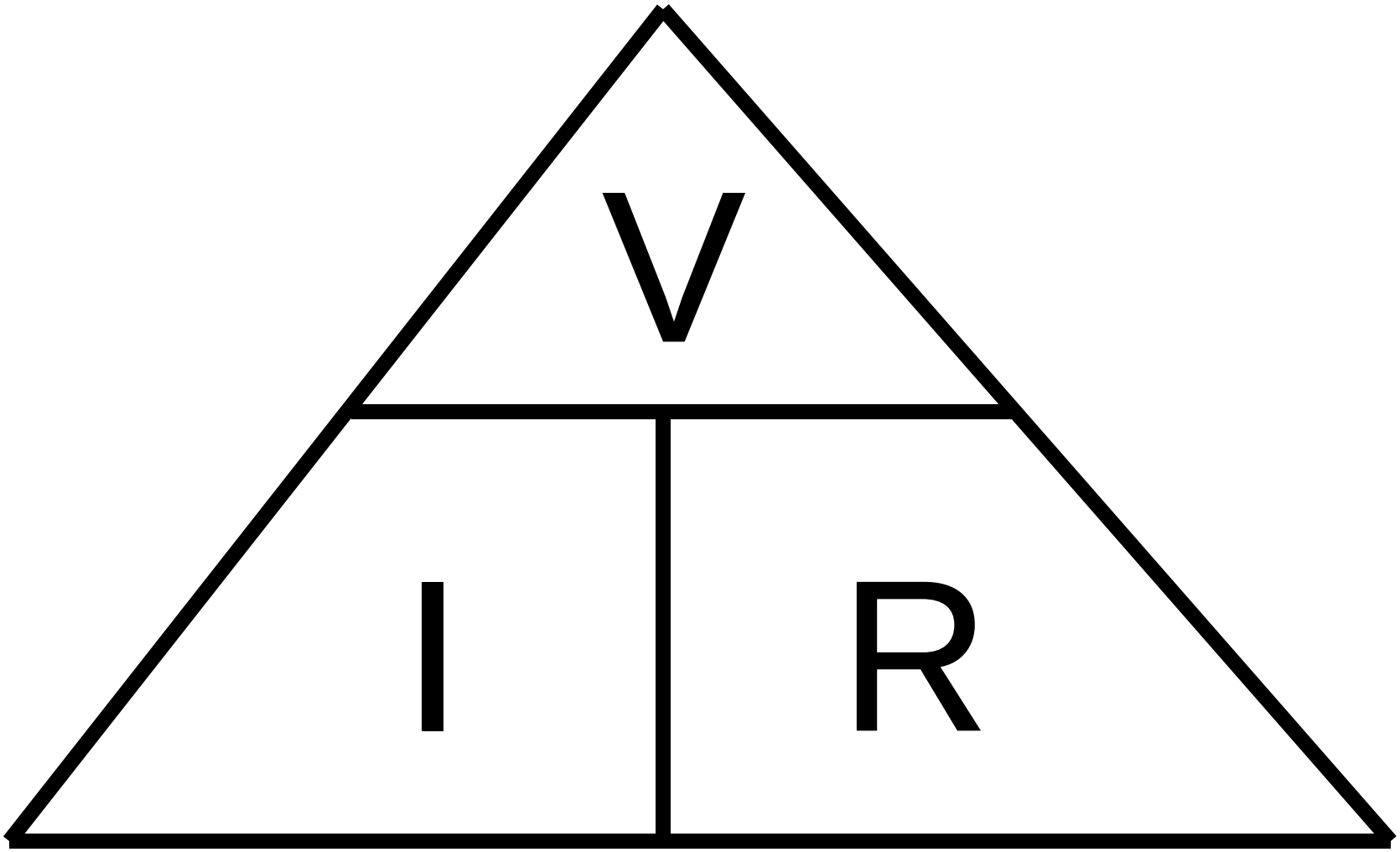
“the most important invention of the 20th century”



Transistor

ANALOGY ALERT

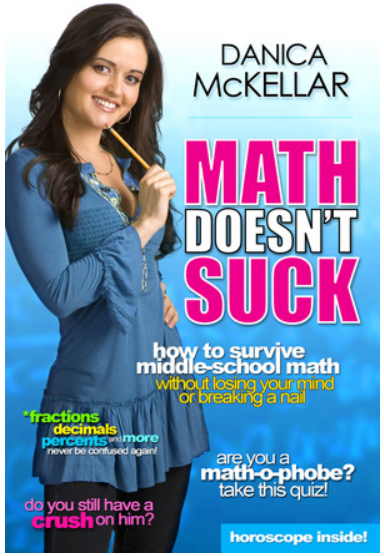
Valve

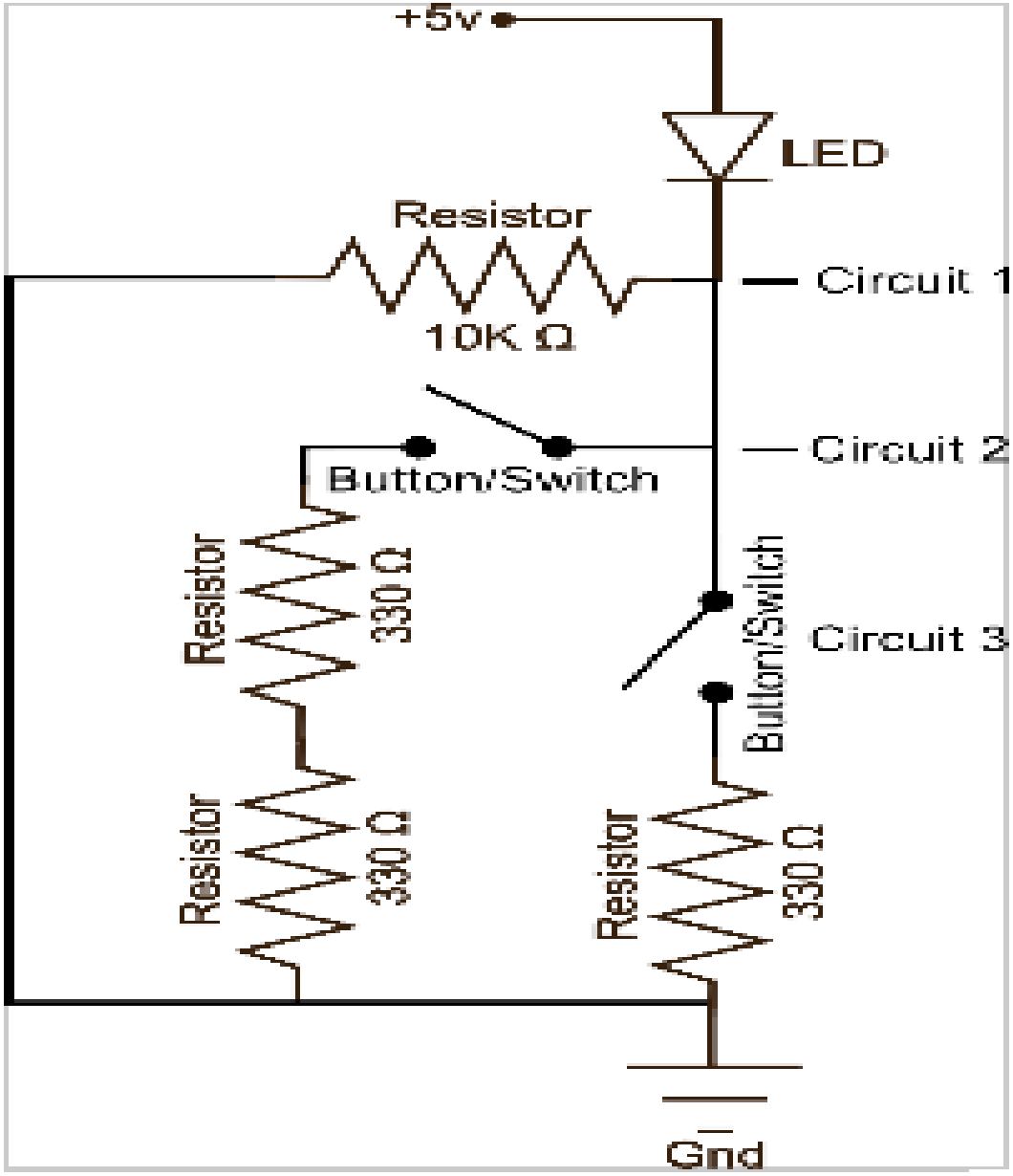

V
I
R

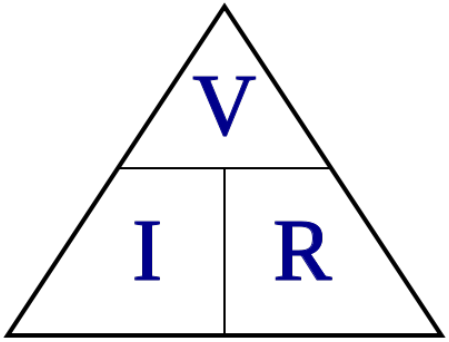
To calculate:

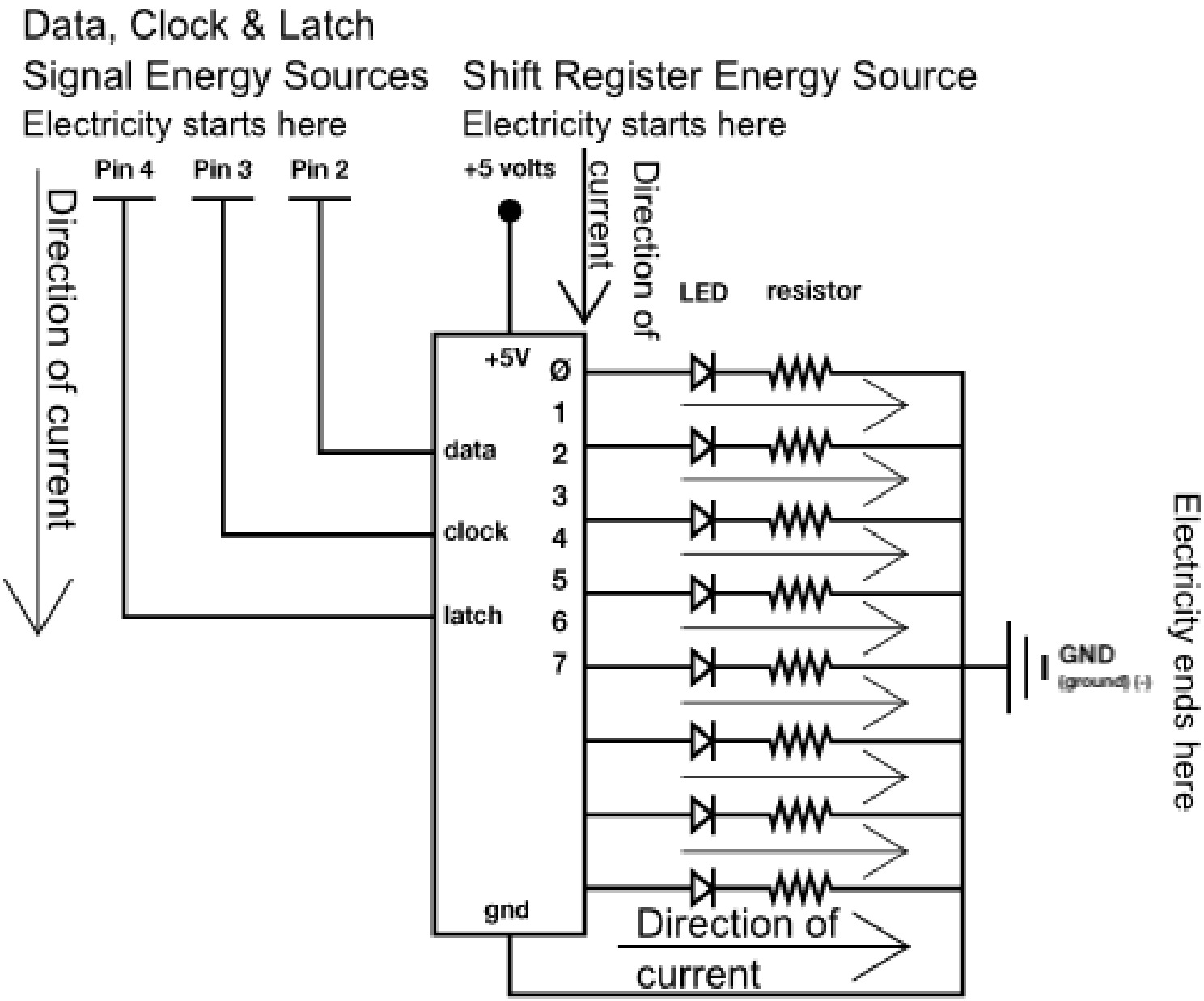
$$\frac{\text{VOLTS}}{\text{AMPS}} = \text{I} \times \text{R}$$

OHMS



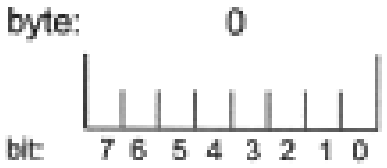

+5v
LED
Resistor
10K Ω
Circuit 1
Circuit 2
Button/Switch
Resistor
330 Ω
Circuit 3
Button/Switch
Resistor
330 Ω
Resistor
330 Ω
Gnd


V
I
R


Data, Clock & Latch
Signal Energy Sources
Electricity starts here
Shift Register Energy Source
Electricity starts here
Pin 4
Pin 3
Pin 2
+5 volts
Direction of current
Direction of current
LED
resistor
+5V
data
clock
latch
gnd
Ø
1
2
3
4
5
6
7
GND
(ground) (-)
Direction of current
Electricity ends here


0 0 1 1 0 1 0 1
byte (8-bits)

‘a’ = 0110 0001



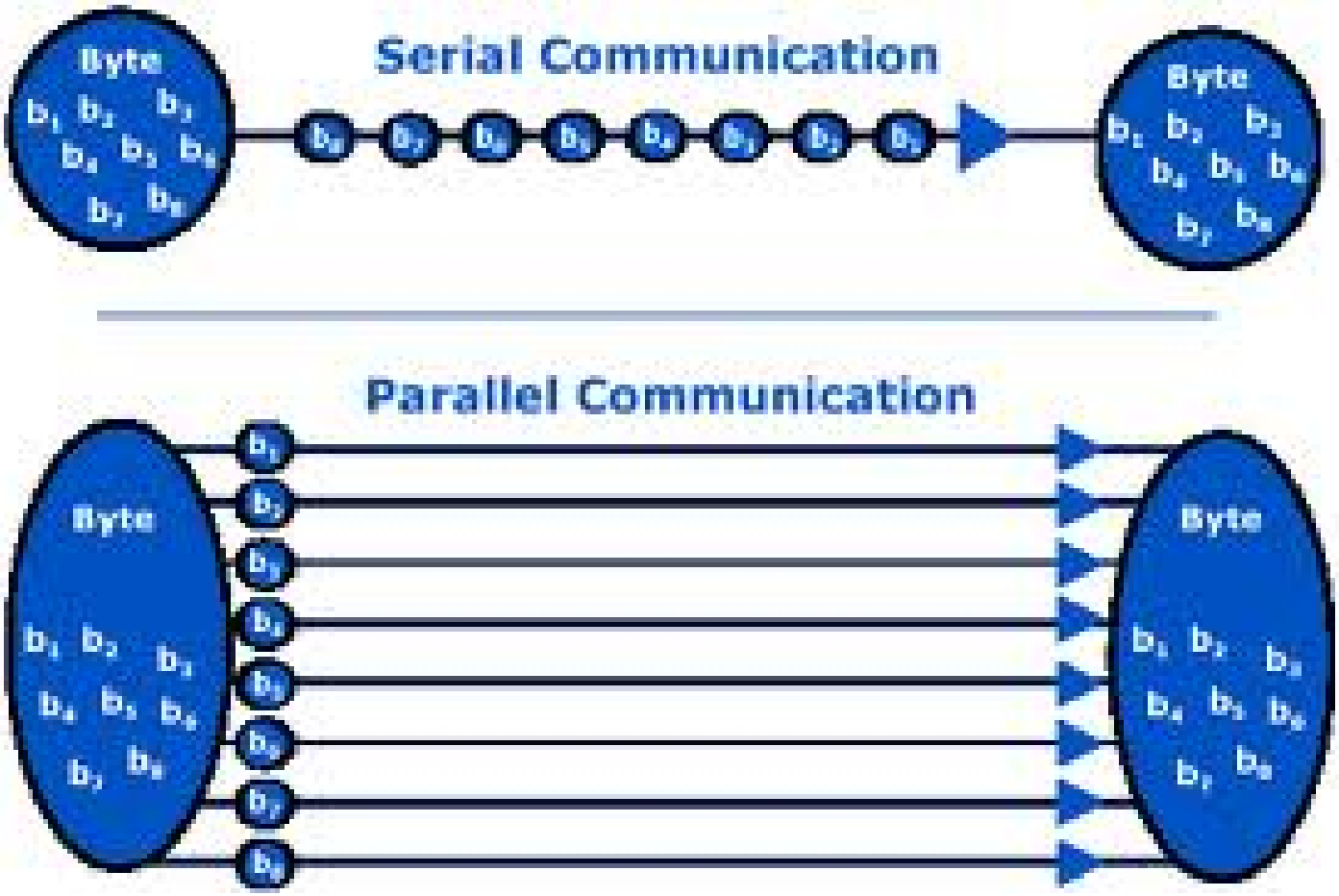
characters in ASCII (8- bits)

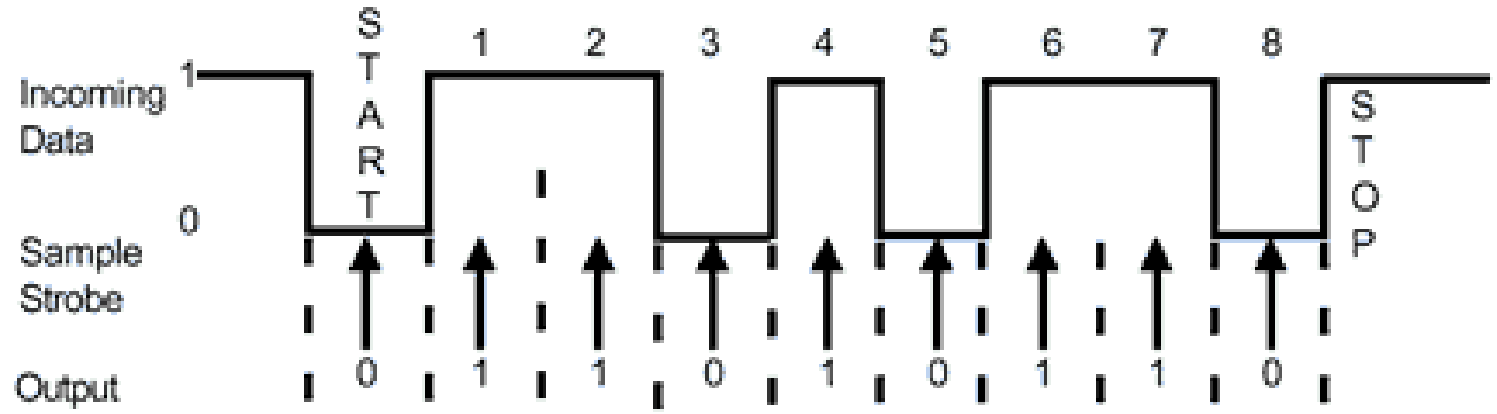
ASCII Code: Character to Binary

0	0011 0000	@	0100 0000	`	0110 0000
1	0011 0001	A	0100 0001	a	0110 0001
2	0011 0010	B	0100 0010	b	0110 0010
3	0011 0011	C	0100 0011	c	0110 0011
4	0011 0100	D	0100 0100	d	0110 0100
5	0011 0101	E	0100 0101	e	0110 0101
6	0011 0110	F	0100 0110	f	0110 0110
7	0011 0111	G	0100 0111	g	0110 0111
8	0011 1000	H	0100 1000	h	0110 1000
9	0011 1001	I	0100 1001	i	0110 1001
A	0100 0001	J	0100 1010	j	0110 1010
B	0100 0010	K	0100 1011	k	0110 1011
C	0100 0011	L	0100 1100	l	0110 1100
D	0100 0100	M	0100 1101	m	0110 1101
E	0100 0101	N	0100 1110	n	0110 1110
F	0100 0110	O	0100 1111	o	0110 1111
G	0100 0111	P	0101 0000	p	0111 0000
H	0100 1000	Q	0101 0001	q	0111 0001
I	0100 1001	R	0101 0010	r	0111 0010
J	0100 1010	S	0101 0011	s	0111 0011
K	0100 1011	T	0101 0100	t	0111 0100
L	0100 1100	U	0101 0101	u	0111 0101
M	0100 1101	V	0101 0110	v	0111 0110
N	0100 1110	W	0101 0111	w	0111 0111
		X	0101 1000	x	0111 1000
		Y	0101 1001	y	0111 1001
		Z	0101 1010	z	0111 1010
		[	0101 1011	{	0111 1011
		\	0101 1100	|	0111 1100
		]	0101 1101	}	0111 1101
		^	0101 1110	~	0111 1110
		_	0101 1111		
		`	0110 0000		
		a	0110 0001		
		b	0110 0010		
		c	0110 0011		
		d	0110 0100		
		e	0110 0101		
		f	0110 0110		
		g	0110 0111		
		h	0110 1000		
		i	0110 1001		
		j	0110 1010		
		k	0110 1011		
		l	0110 1100		
		m	0110 1101		
		n	0110 1110		
		o	0110 1111		
		p	0111 0000		
		q	0111 0001		
		r	0111 0010		
		s	0111 0011		
		t	0111 0100		
		u	0111 0101		
		v	0111 0110		
		w	0111 0111		
		x	0111 1000		
		y	0111 1001		
		z	0111 1010		
		{	0111 1011		
		|	0111 1100		
		}	0111 1101		
		~	0111 1110		
		space	0010 0000		

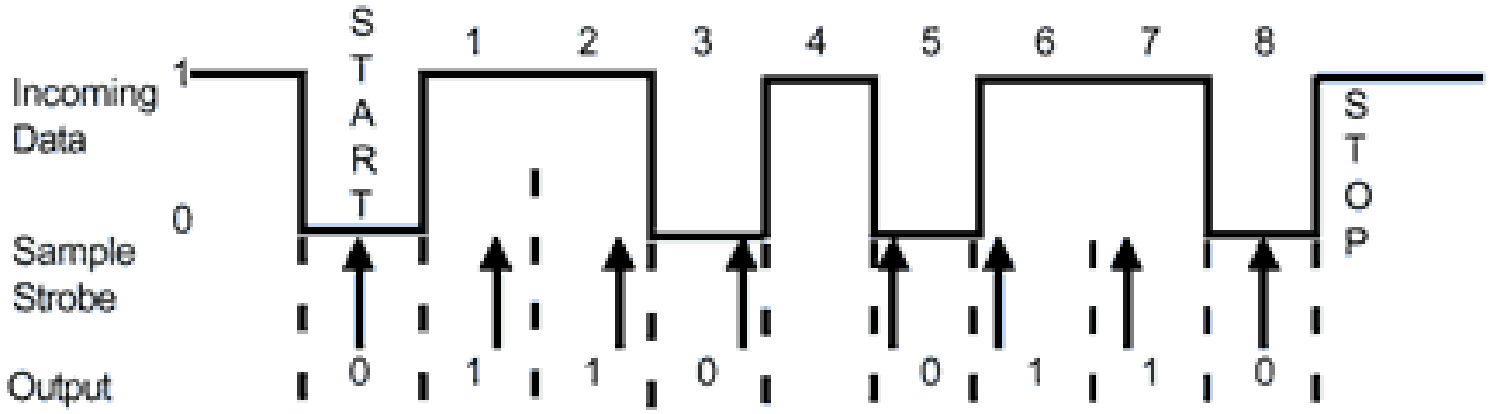
Data Types

- void
- boolean
- char
- unsigned char
- byte
- int
- unsigned int
- word
- long
- unsigned long
- float
- double


Serial Communication
Byte
Byte
Parallel Communication
Byte
Byte



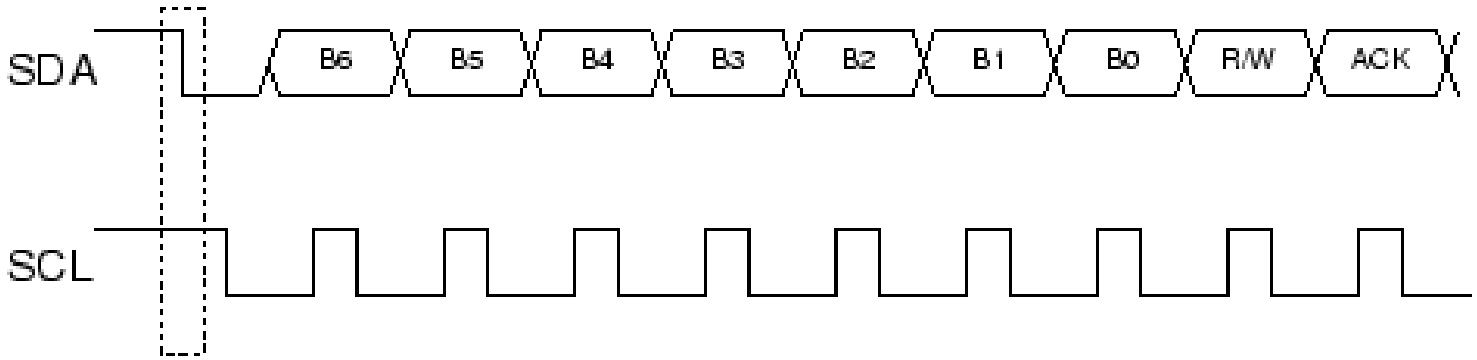
a. Best case, receiver samples at midpoint of each bit.



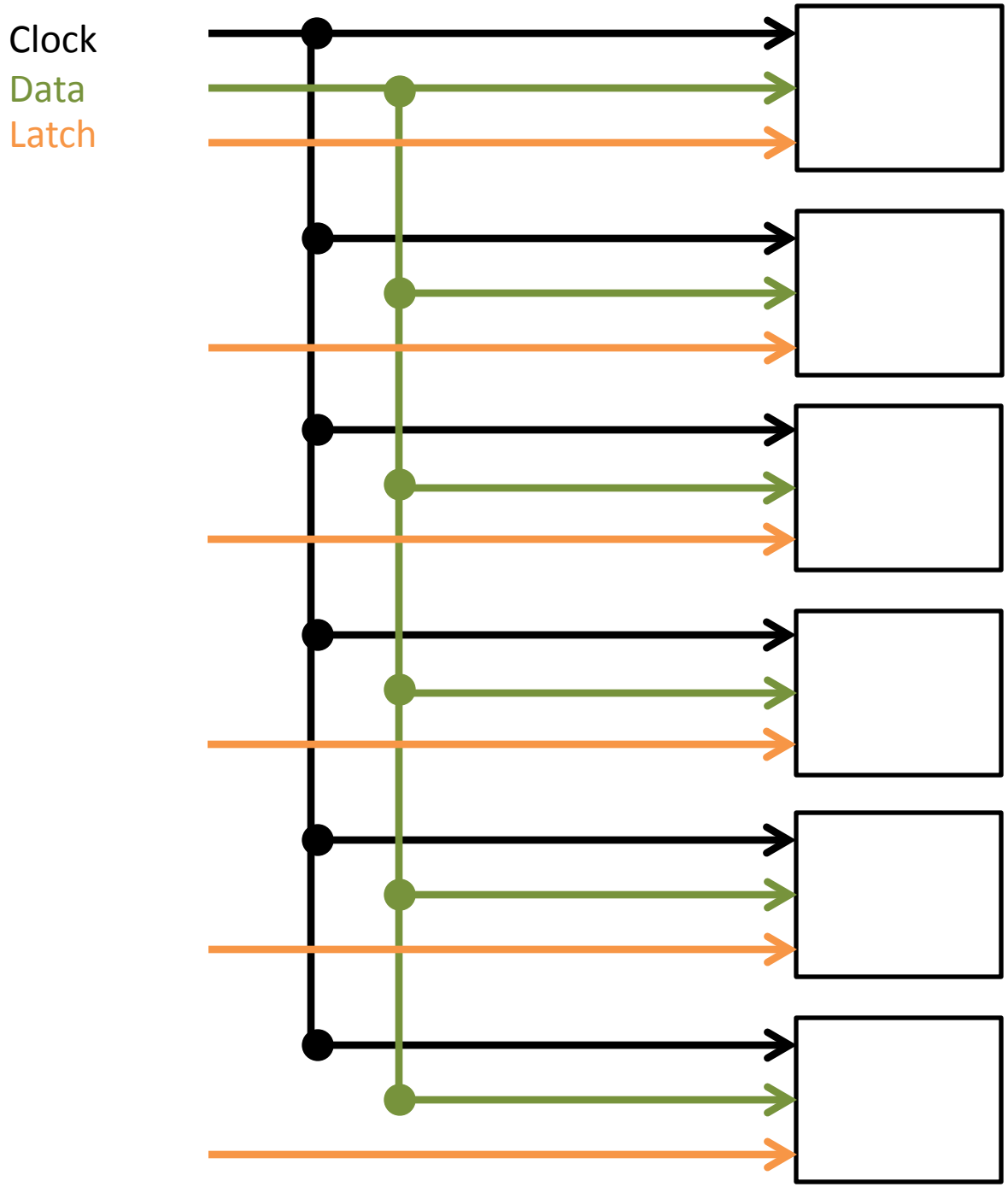
b. Receiving clock is too slow, causing bit 4 to be skipped and the data to be corrupted.

Ideal and corrupted asynchronous data sampling

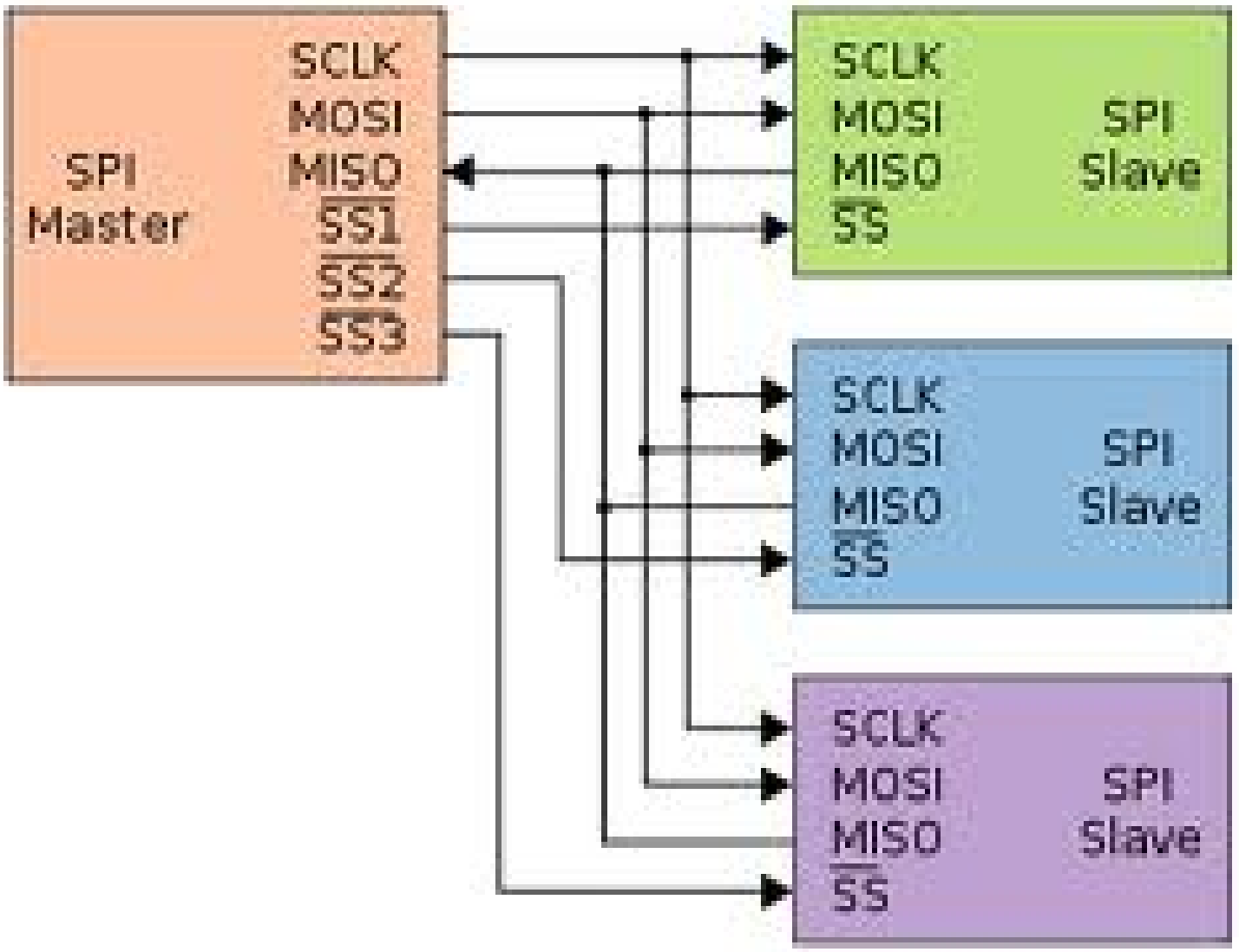
One-Wire



Two-Wire



Three-Wire

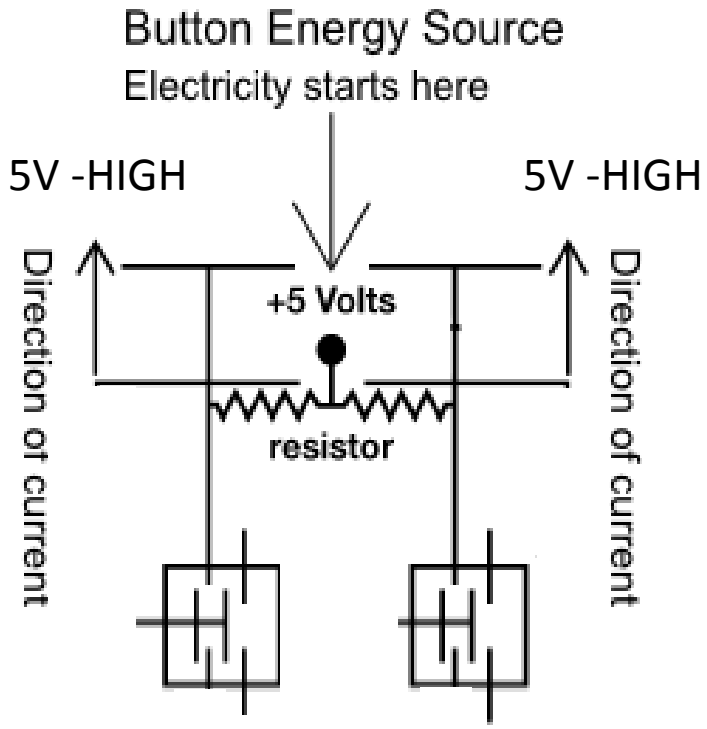

SPI
Master
SCLK
MOSI
MISO
SS1
SS2
SS3
SCLK
MOSI
MISO
SS
SPI
Slave
SCLK
MOSI
MISO
SS
SPI
Slave
SCLK
MOSI
MISO
SS
SPI
Slave

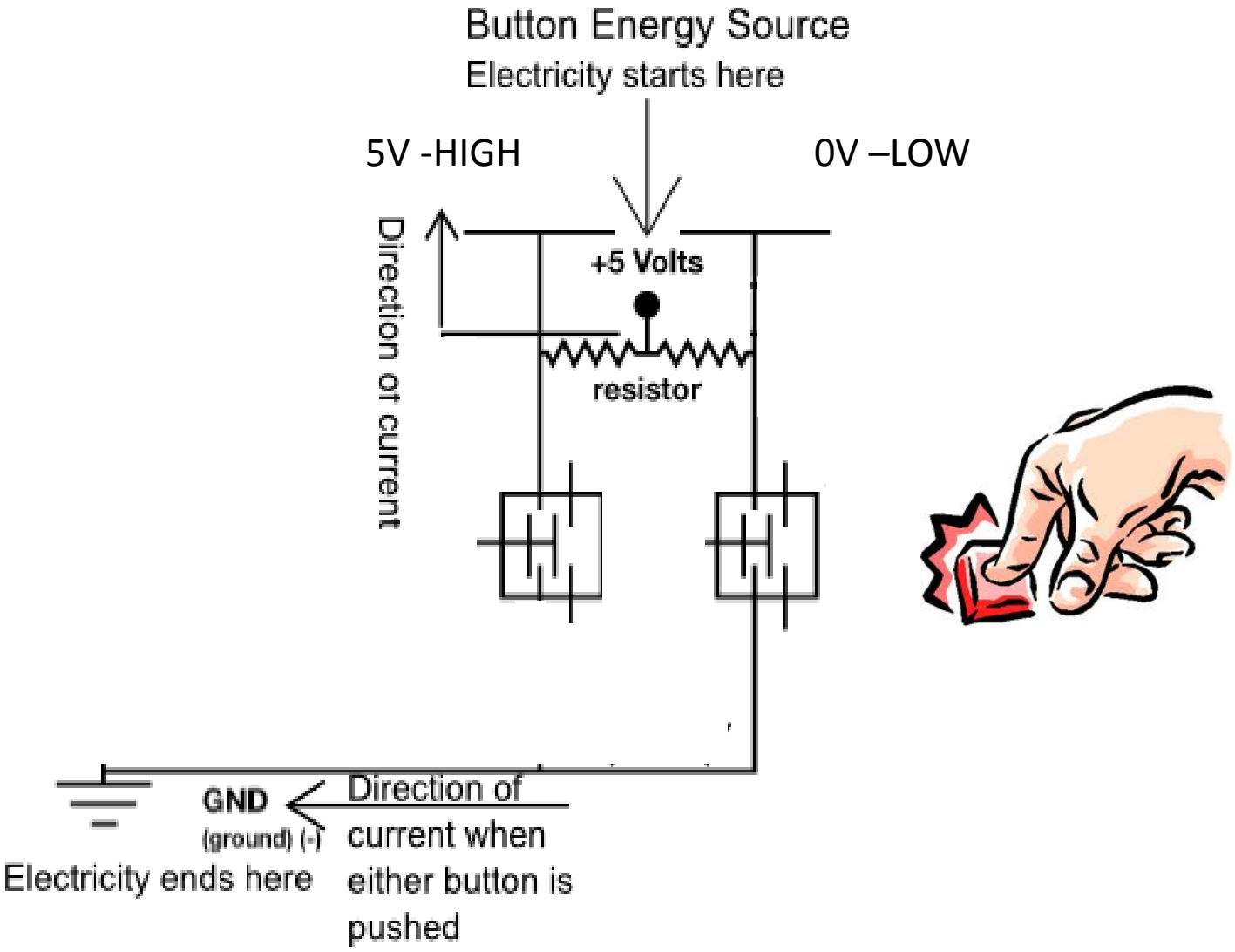
Four-Wire

```
int temperaturePin = 0;

void setup()
{
Serial.begin(9600);     //Serial comm. at a Baud Rate of 9600
}

void loop()
{
 float temp = getVoltage(temperaturePin);

 //Below is a line that compensates for an offset (see datasheet)
 temp = (temp - .5) * 100;

 Serial.println(temp);  // Send data to PC
 delay(1000);
}

float getVoltage(int pin)
{
 return (analogRead(pin) * .004882814);
}
```

So how can we make the button appear like a digital signal?

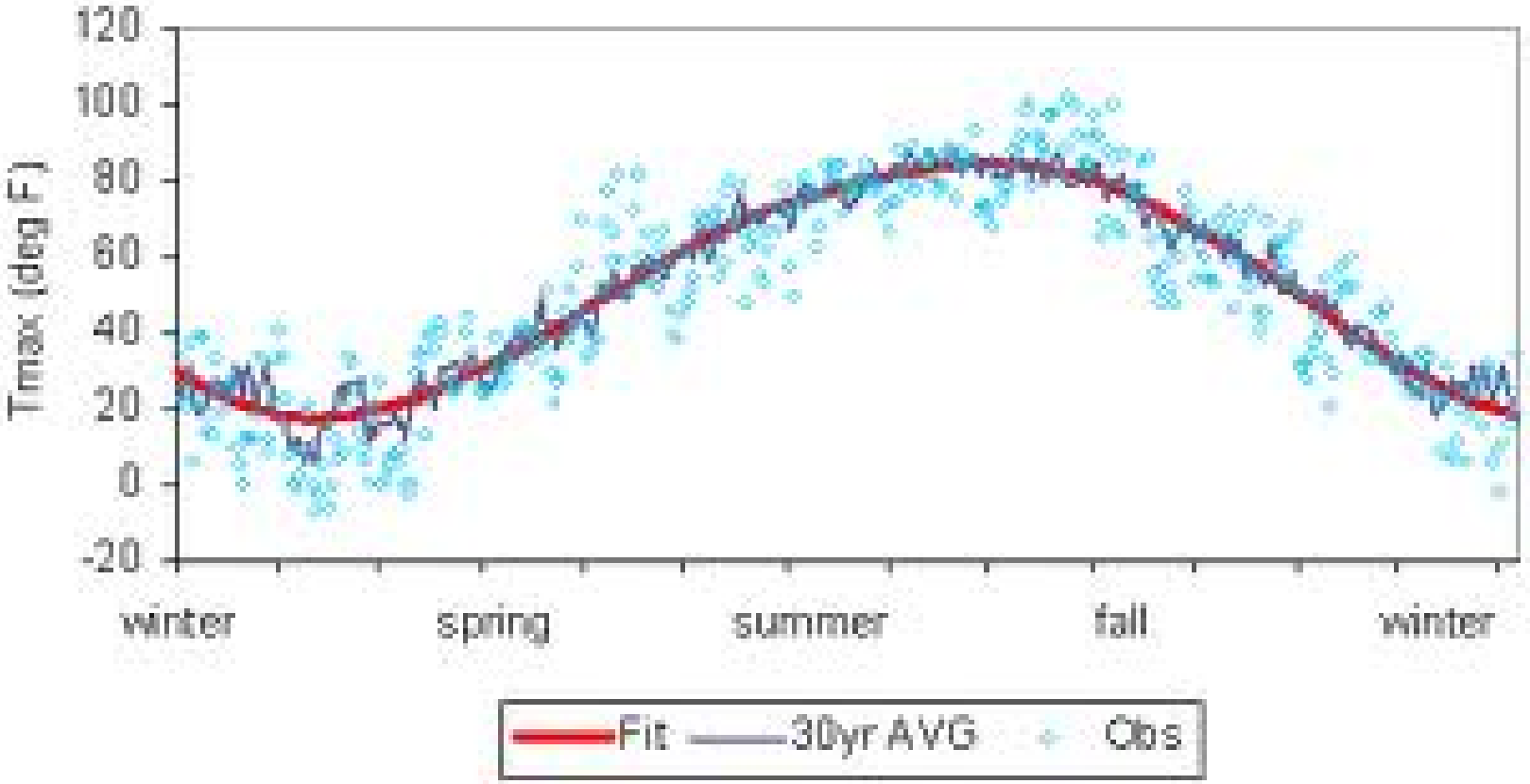
Pull-up Resistors

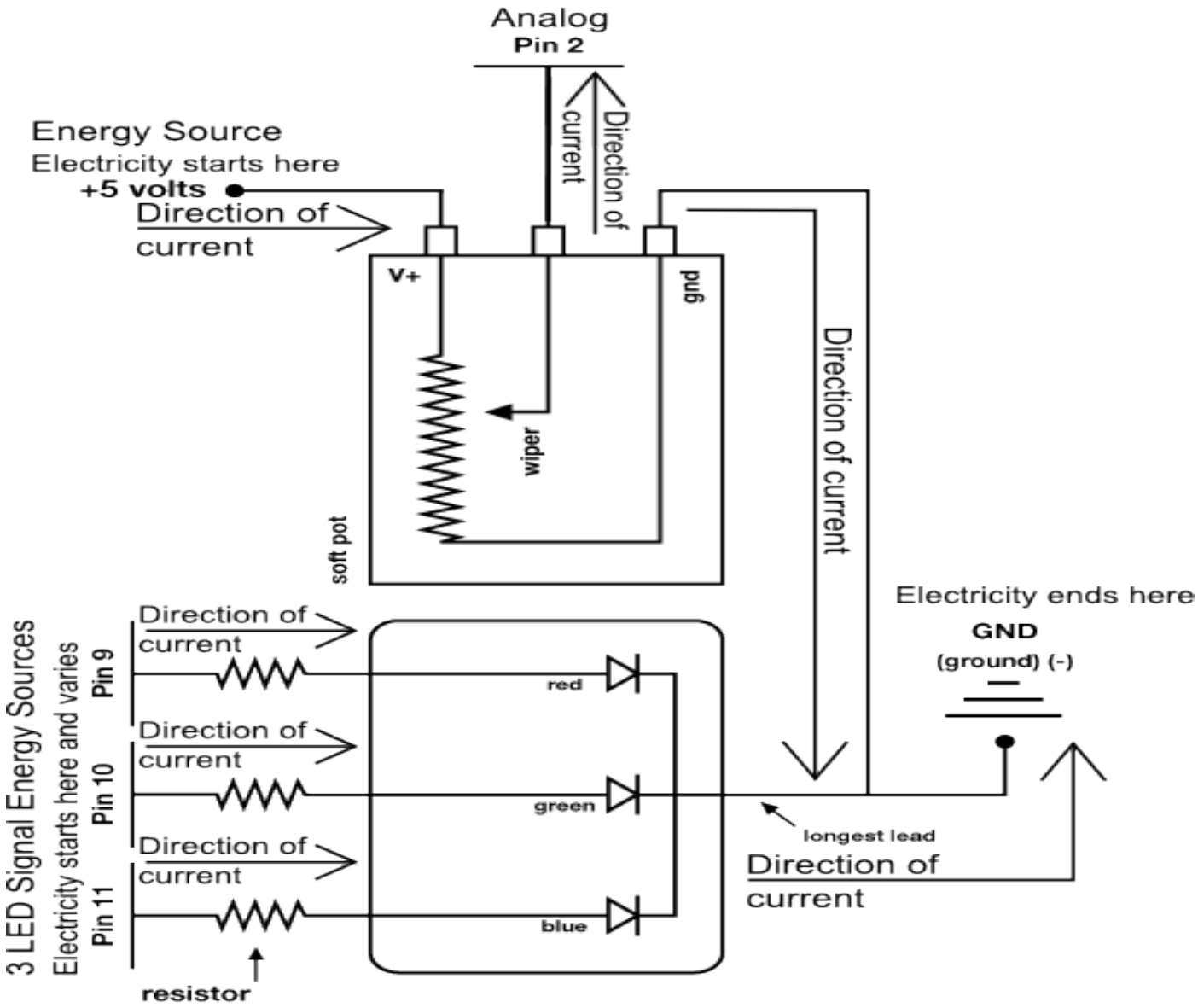

Button Energy Source
Electricity starts here
5V -HIGH
5V -HIGH
Direction of current
Direction of current
+5 Volts
resistor

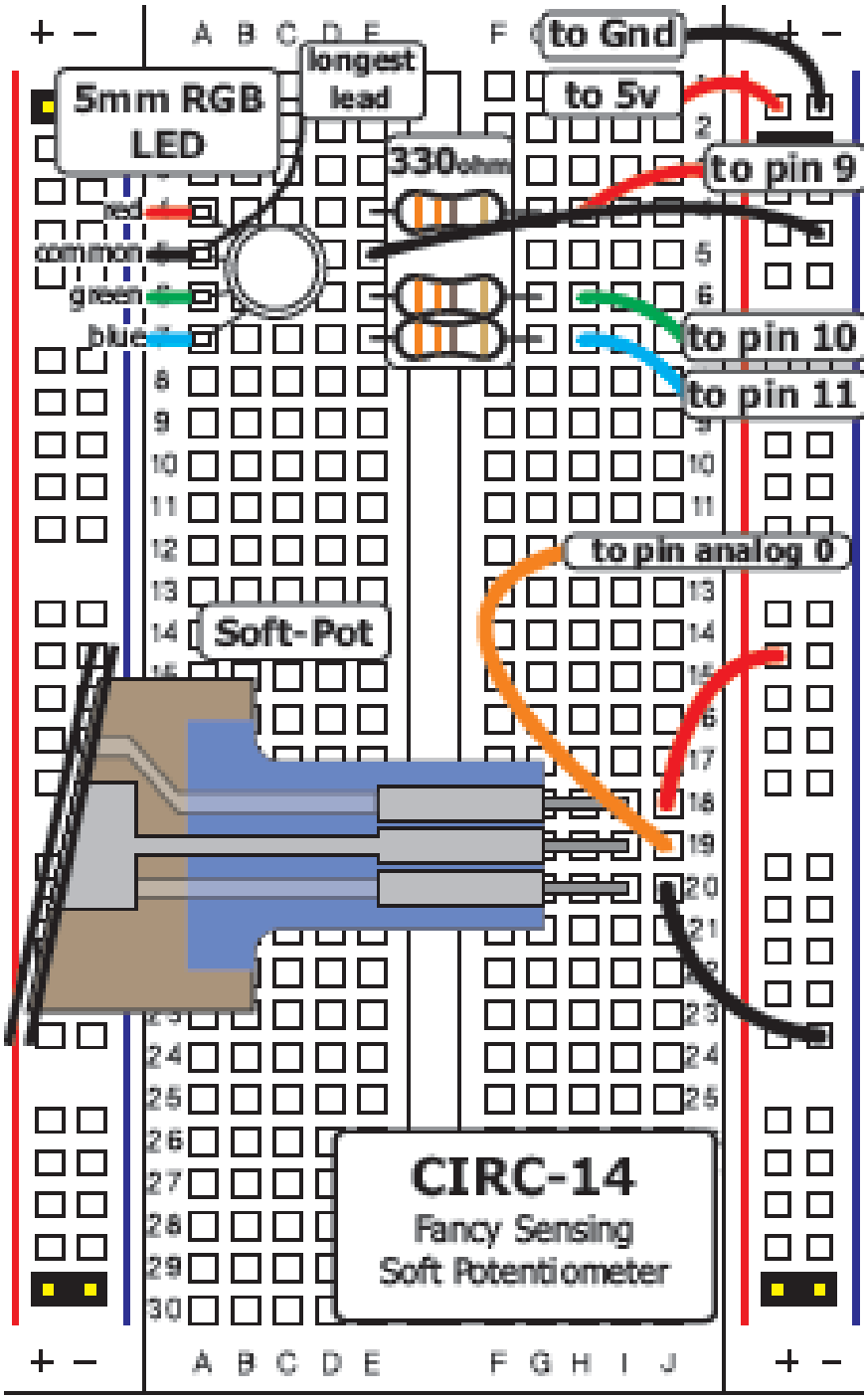

Button Energy Source
Electricity starts here
5V -HIGH
0V –LOW
Direction of current
+5 Volts
resistor
GND
(ground) (-)
Electricity ends here
Direction of current when either button is pushed

Low-Pass Filtering

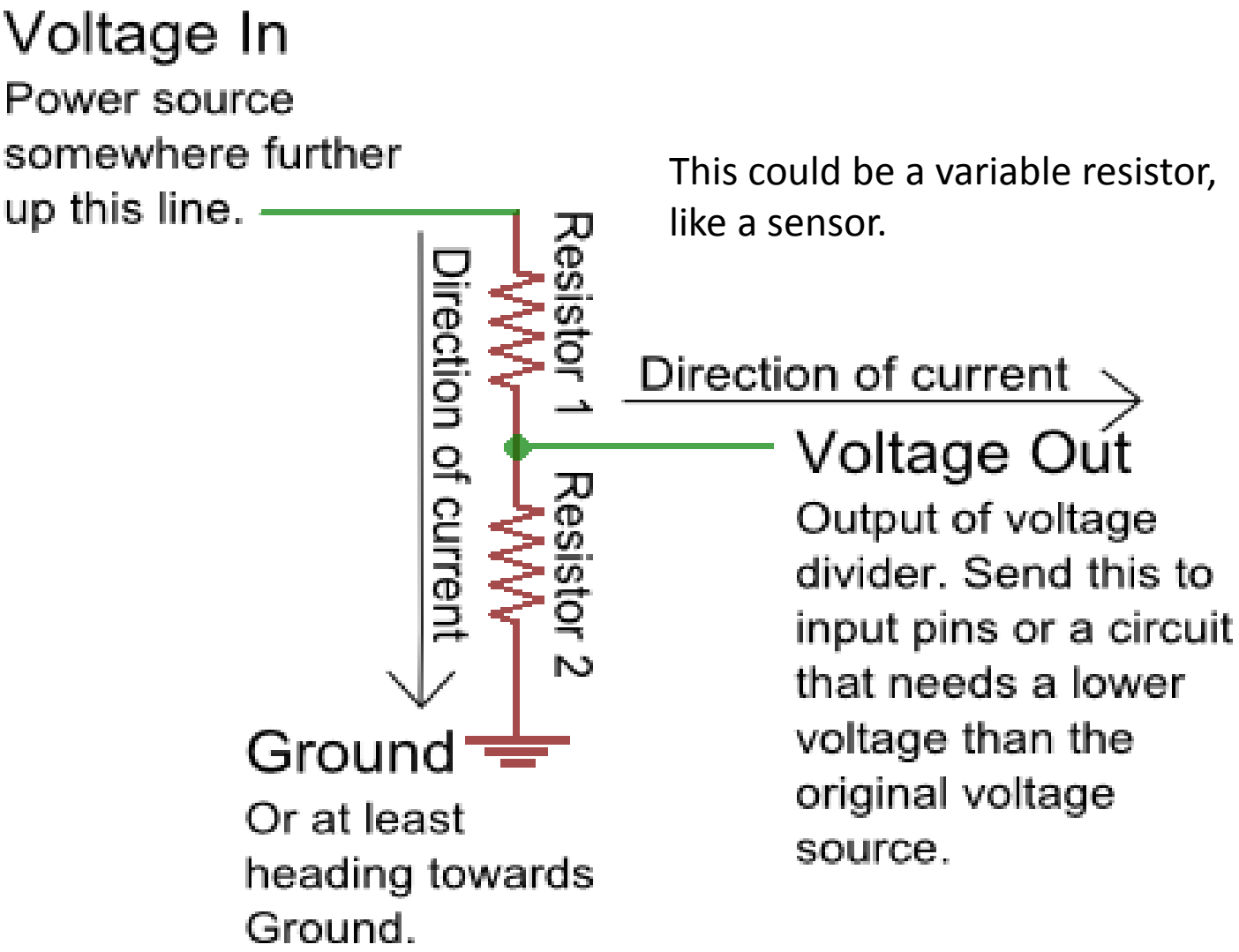
By averaging consecutive values, the rapidly-changing values are removed, revealing the underlying “trend”

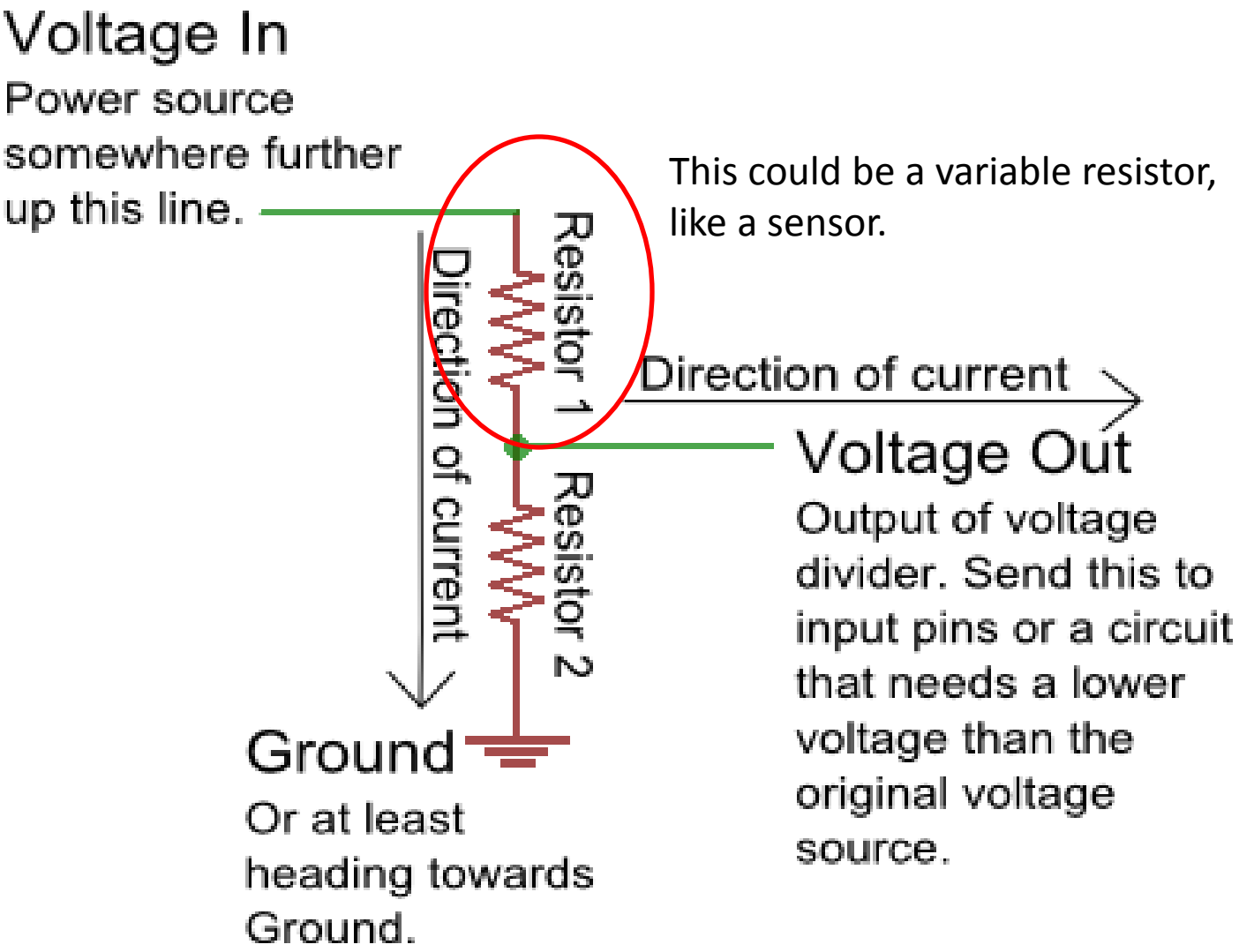

Tmax (deg F)
120
100
80
60
40
20
0
-20
winter
spring
summer
fall
winter
Fit
30yr AVG
Obs


Analog
Pin 2
current
Direction of
Energy Source
Electricity starts here
+5 volts
Direction of
current
V+
gnd
wiper
soft pot
Direction of current
Electricity ends here
GND
(ground) (-)
Direction of
current
Pin 9
Direction of
current
Pin 10
Direction of
current
Pin 11
3 LED Signal Energy Sources
Electricity starts here and varies
red
green
blue
longest lead
Direction of
current
resistor

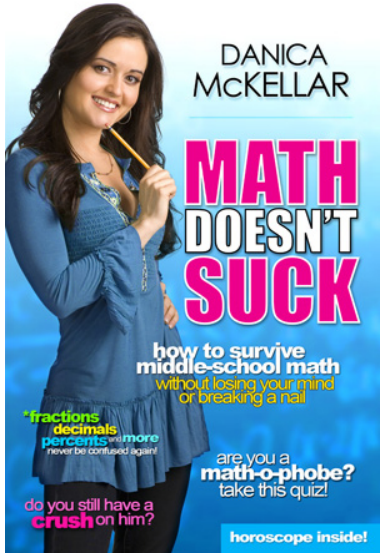

5mm RGB LED
longest lead
red
common
green
blue
330ohm
to Gnd
to 5v
to pin 9
to pin 10
to pin 11
to pin analog 0
Soft-Pot
CIRC-14
Fancy Sensing
Soft Potentiometer

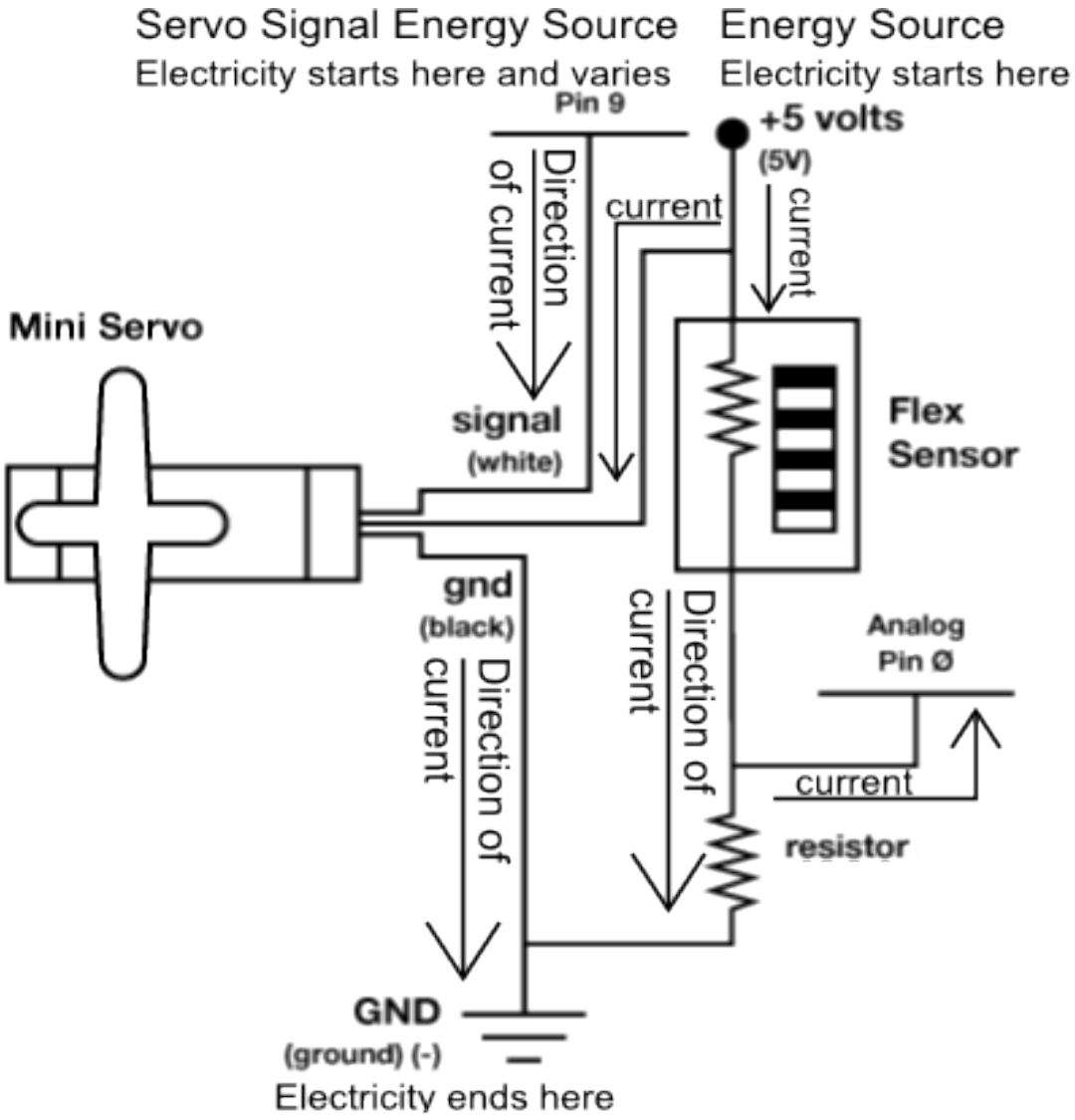
Voltage Dividers

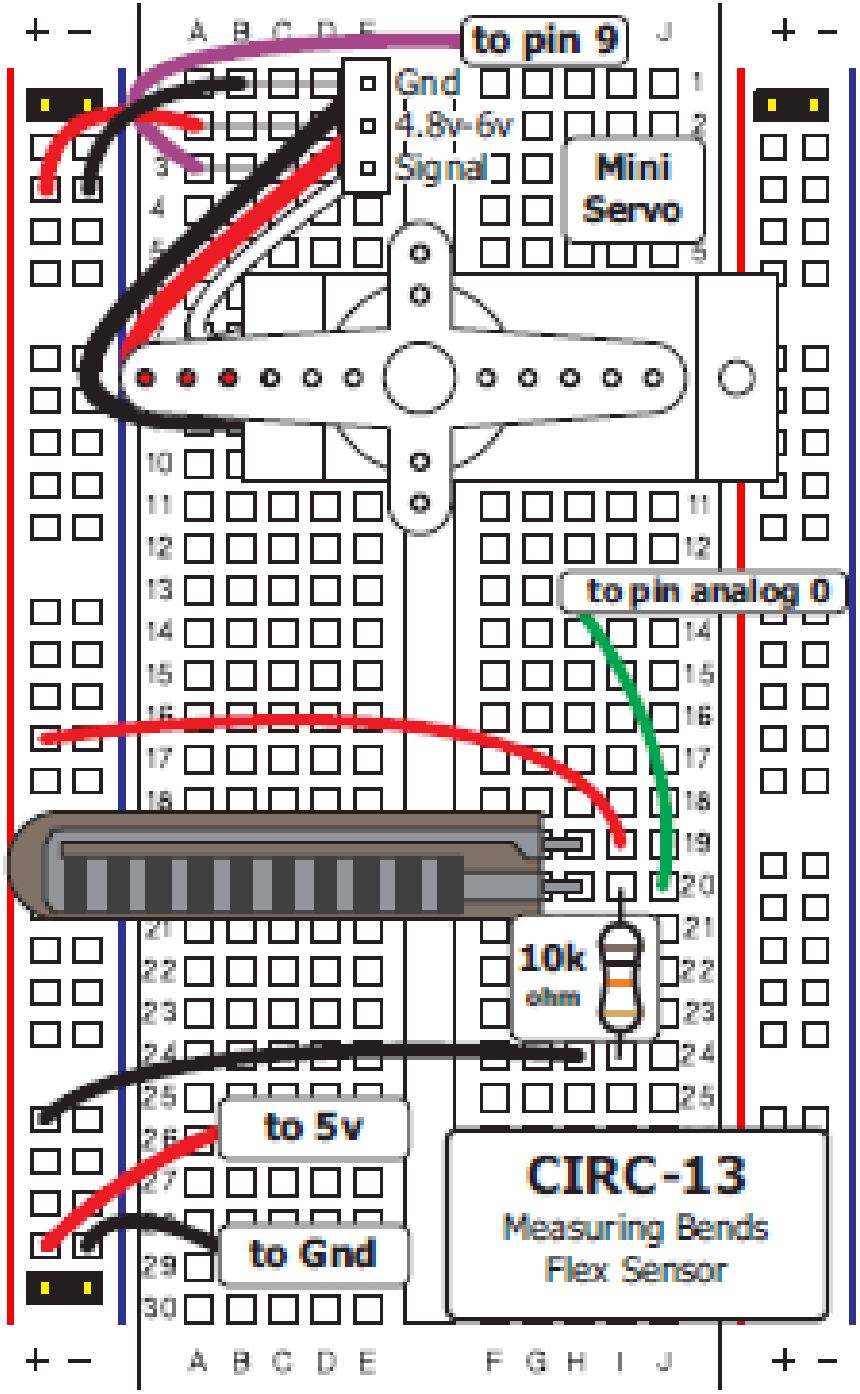

Voltage In
Power source somewhere further up this line.
This could be a variable resistor, like a sensor.
Resistor 1
Resistor 2
Direction of current
Direction of current
Voltage Out
Output of voltage divider. Send this to input pins or a circuit that needs a lower voltage than the original voltage source.
Ground
Or at least heading towards Ground.


Voltage In
Power source somewhere further up this line.
This could be a variable resistor, like a sensor.
Resistor 1
Direction of current
Direction of current
Voltage Out
Output of voltage divider. Send this to input pins or a circuit that needs a lower voltage than the original voltage source.
Resistor 2
Ground
Or at least heading towards Ground.

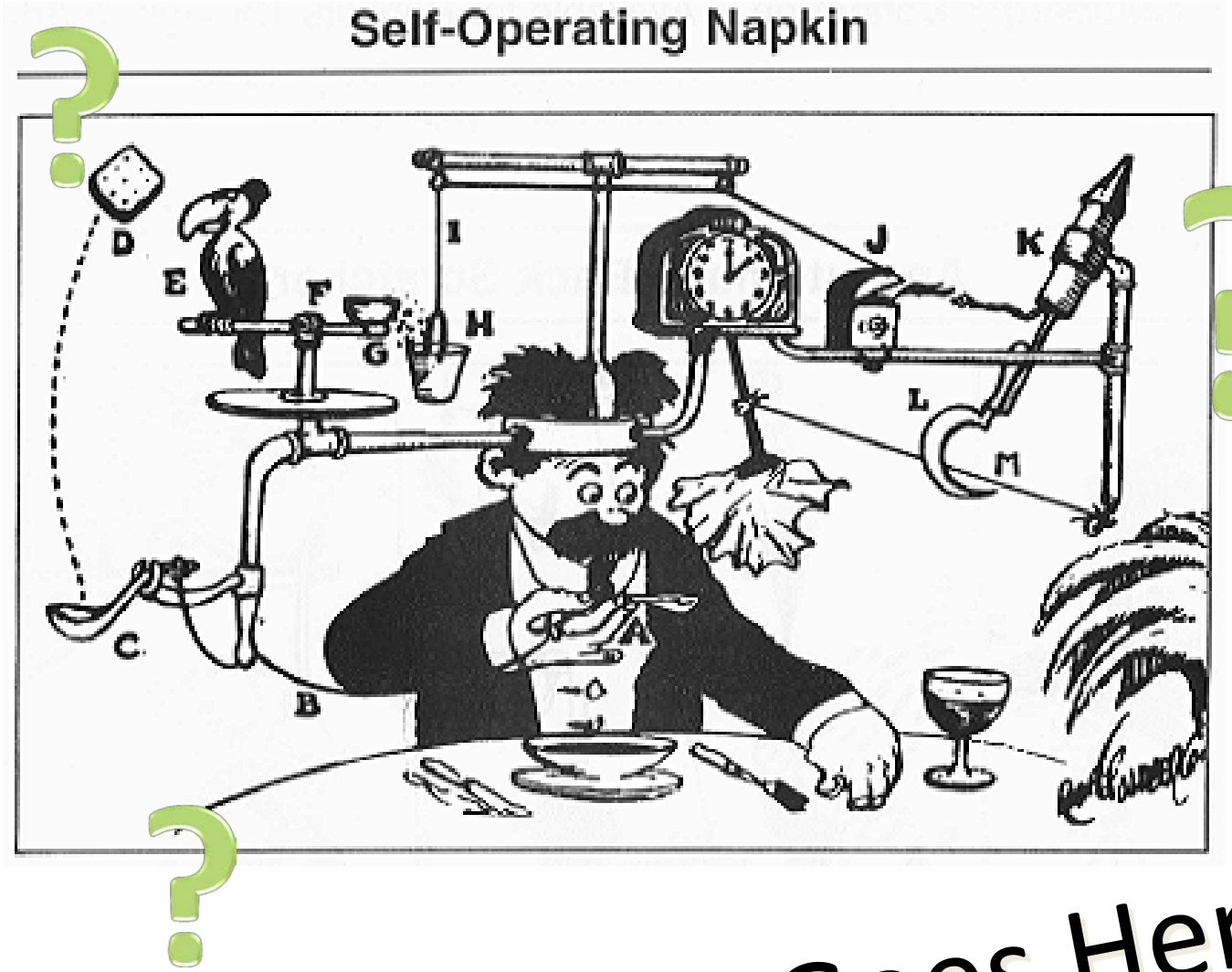
$$V_{out} = V_{in} \frac{R_2}{(R_1 + R_2)}$$




Servo Signal Energy Source
Electricity starts here and varies
Energy Source
Electricity starts here
Pin 9
+5 volts
(5V)
Direction of current
current
current
Mini Servo
signal
(white)
Flex Sensor
gnd
(black)
current
Direction of
current
Direction of
Analog Pin Ø
current
resistor
GND
(ground) (-)
Electricity ends here


to pin 9
Gnd
4.8v-6v
Signal
Mini
Servo
to pin analog 0
10k
ohm
to 5v
to Gnd
CIRC-13
Measuring Bends
Flex Sensor


OFF THE MAP



Your Project Goes Here

	LED Array	Midi	LCD Array	Motor	Button	Serial	Sonar	Audio	Sharp IR	Servos	Op-Amp
Kate		X		X			X				
Charlie		X							X?	X?	X
Jasmine	X	X									
Quinn			X		X						
Ava				X	X						
Lexa				X				X			X
Jim						X					
Braden			X		X						
Jameyia				X					X	X	

Libraries

Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from **Sketch > Import Library**.

Standard Libraries

- EEPROM - reading and writing to "permanent" storage
- Ethernet - for connecting to the internet using the Arduino Ethernet Shield
- Firmata - for communicating with applications on the computer using a standard serial protocol.
- LiquidCrystal - for controlling liquid crystal displays (LCDs)
- SD - for reading and writing SD cards
- Servo - for controlling servo motors
- SPI - for communicating with devices using the Serial Peripheral Interface (SPI) Bus
- SoftwareSerial - for serial communication on any digital pins
- Stepper - for controlling stepper motors
- Wire - Two Wire Interface (TWI/I2C) for sending and receiving data over a net of devices or sensors.

http://arduino.cc/en/Reference/Libraries

Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from **Sketch > Import Library**.

Standard Libraries

- EEPROM - reading and writing to "permanent" storage
- Ethernet - for connecting to the internet using the Arduino Ethernet Shield
- Firmata - for communicating with applications on the computer using a standard serial protocol.
- LiquidCrystal - for controlling liquid crystal displays (LCDs)
- SD - for reading and writing SD cards
- Servo - for controlling servo motors
- SPI - for communicating with devices using the Serial Peripheral Interface (SPI) Bus
- SoftwareSerial - for serial communication on any digital pins
- Stepper - for controlling stepper motors
- Wire - Two Wire Interface (TWI/I2C) for sending and receiving data over a net of devices or sensors.

http://arduino.cc/en/Reference/Libraries

Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from **Sketch > Import Library**.

Standard Libraries

- EEPROM - reading and writing to "permanent" storage
- Ethernet - for connecting to the internet using the Arduino Ethernet Shield
- Firmata - for communicating with applications on the computer using a standard serial protocol.
- LiquidCrystal - for controlling liquid crystal displays (LCDs)
- SD - for reading and writing SD cards
- Servo - for controlling servo motors
- SPI - for communicating with devices using the Serial Peripheral Interface (SPI) Bus
- SoftwareSerial - for serial communication on any digital pins
- Stepper - for controlling stepper motors
- Wire - Two Wire Interface (TWI/I2C) for sending and receiving data over a net of devices or sensors.

http://arduino.cc/en/Reference/Libraries

Libraries

Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data. To use a library in a sketch, select it from **Sketch > Import Library**.

Standard Libraries

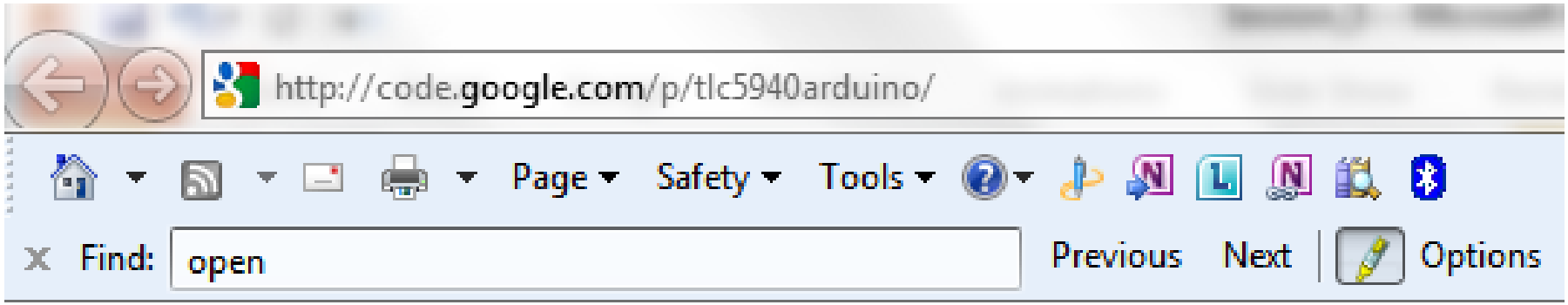
- EEPROM - reading and writing to "permanent" storage
- Ethernet - for connecting to the internet using the Arduino Ethernet Shield
- Firmata - for communicating with applications on the computer using a standard serial protocol.
- LiquidCrystal - for controlling liquid crystal displays (LCDs)
- SD - for reading and writing SD cards
- Servo - for controlling servo motors
- SPI - for communicating with devices using the Serial Peripheral Interface (SPI) Bus
- SoftwareSerial - for serial communication on any digital pins
- Stepper - for controlling stepper motors
- Wire - Two Wire Interface (TWI/I2C) for sending and receiving data over a net of devices or sensors.

http://arduino.cc/en/Reference/Libraries

But what if the standard libraries don't meet my needs?

User-created Libraries


Google
Evil
Web
Images
Groups
Arduino library TLC5940



tlc5940arduino

An Arduino Library for the TI TLC5940 16-Channel PWM Chip

Project Home | Downloads | Wiki | Issues | Source

User-created libraries go in a subdirectory of your default sketch directory:

~/Documents/Arduino/libraries/

My Documents\Arduino\libraries

*It will then appear in the **Sketch | Import Library** menu in the Arduino IDE.*

```
#include "Tlc5940.h"

void setup()
{
  Tlc.init();
}
```

```
#include "Tlc5940.h"

void setup()
{
  Tlc.init();
}
```

NOTE: #include will tell the compiler about the functions you can call from your code.

```
#include "Tlc5940.h"

void setup()
{
  Tlc.init();
}
```

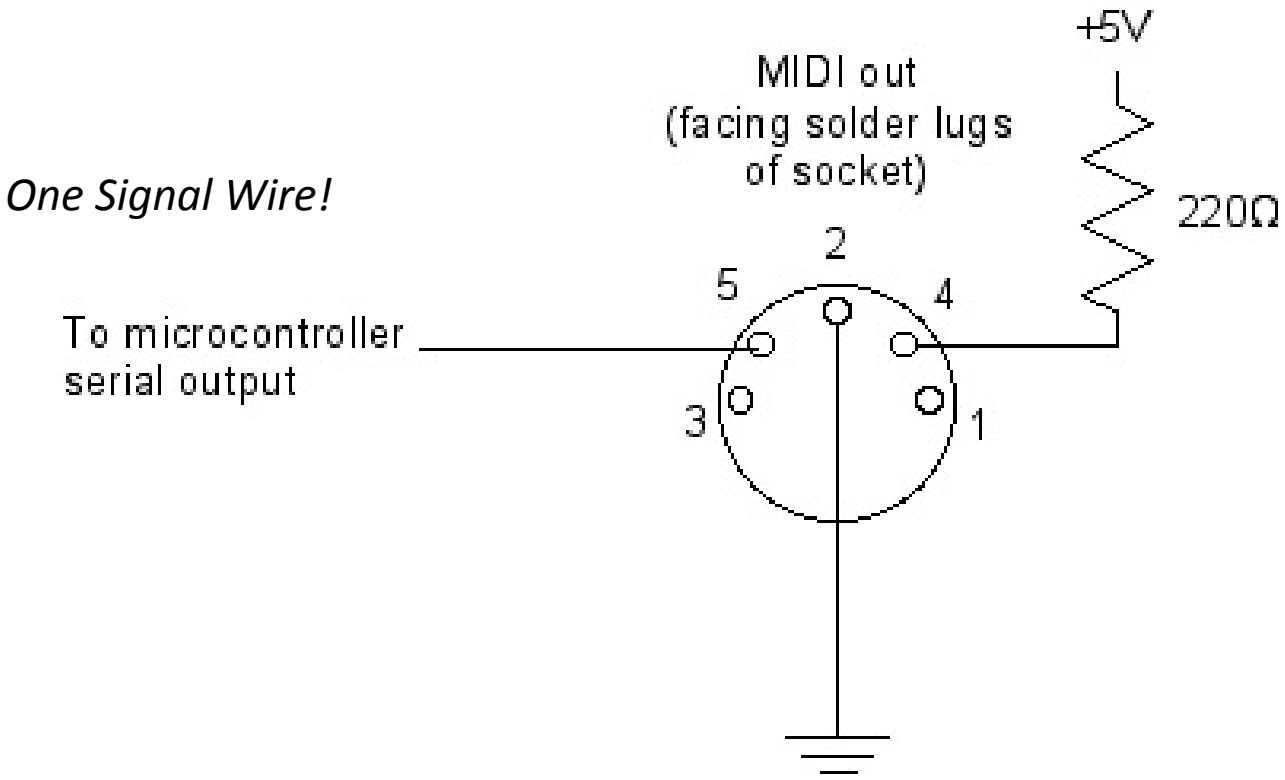
NOTE: #include will tell the compiler about the functions you can call from your code.

NOTE: This will link the library to your program, making it larger.

ColorLCDShield
PlainADC
TLC5940
PlainFFT

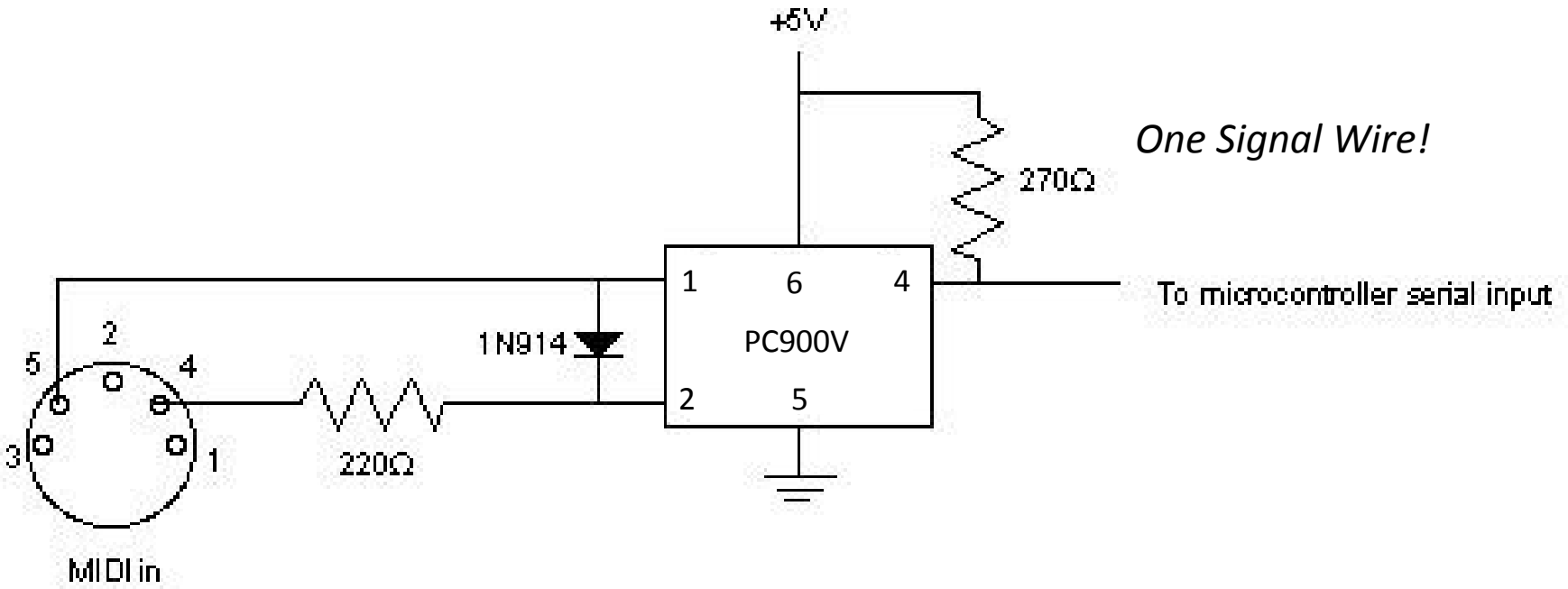
Hi Scott,

You made my day! Because of the nature of your project: when I demonstrated Arduino to my associate Bernard he had a deep tought: "With Arduino, we can build Silicon Valleys in any place around the world". Well, I guess this includes the middle of the Pacific.



Sending a MIDI message to a Synthesizer

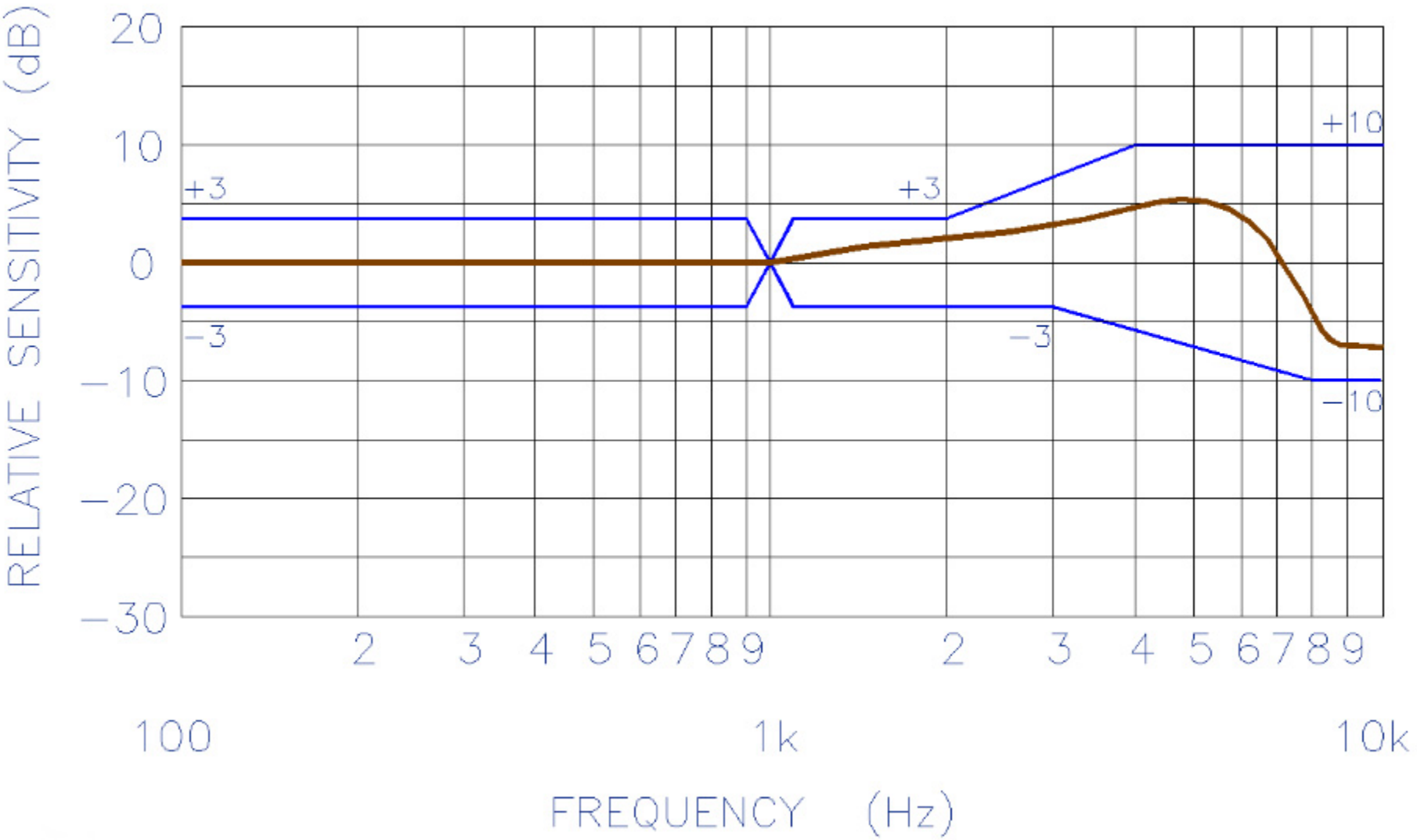
```
void setup()
{
   Serial.begin(31250);   // MIDI Serial Comm. bitrate
}

void loop()
{
  for (int note = 0x1E; note < 0x5A; note ++)  // F#-0 (0x1E) to F#-5 (0x5A):
    {
      noteOn(0x90, note, 0x45);  // channel 1 (0x90), middle velocity (0x45)
      delay(100);
      noteOn(0x90, note, 0x00); // same channel & note, silent velocity (0x00)
      delay(100);
    }
}

void noteOn(int cmd, int pitch, int velocity)  // Check to see if cmd > 127, data < 127 ?
{
  Serial.write(cmd);
  Serial.write(pitch);
  Serial.write(velocity);
}
```



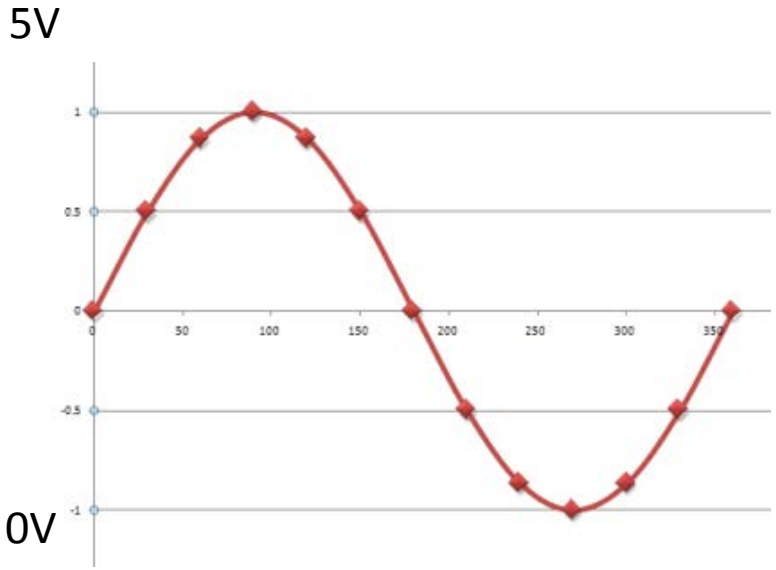
Receiving a MIDI message from a Synthesizer

```
void setup()
{
  pinMode(midiEnable,OUTPUT);
  Serial.begin(31250);   // MIDI Serial Comm. bitrate
  digitalWrite(midiEnable, HIGH); // Turn MIDI input on
}

void loop()
{
  if (Serial.available() > 0)
  {
     blink();
     delay(200);
     byte inByte = Serial.read();
     if (incomingByte== 0x90)
     {
       // note on message channel 1; followed by 2 bytes (key, and velocity)
     }
     if (incomingByte== 0x80)
     {
       // note off message channel 1; followed by 1 byte (key)
     }
}
```

Electret Microphone

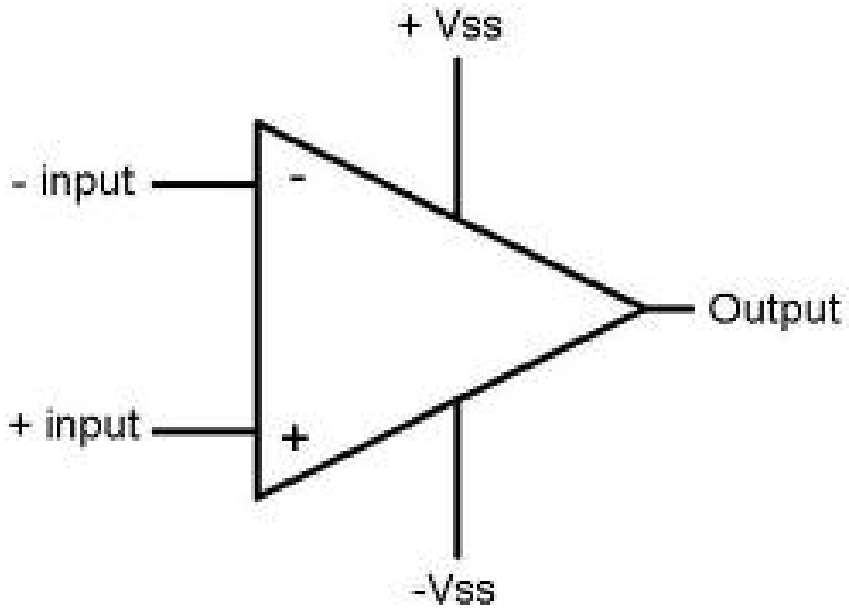

RELATIVE SENSITIVITY (dB)
20
10
0
−10
−20
−30
+3
−3
+3
−3
+10
−10
2 3 4 5 6 7 8 9
2 3 4 5 6 7 8 9
100
1k
10k
FREQUENCY (Hz)

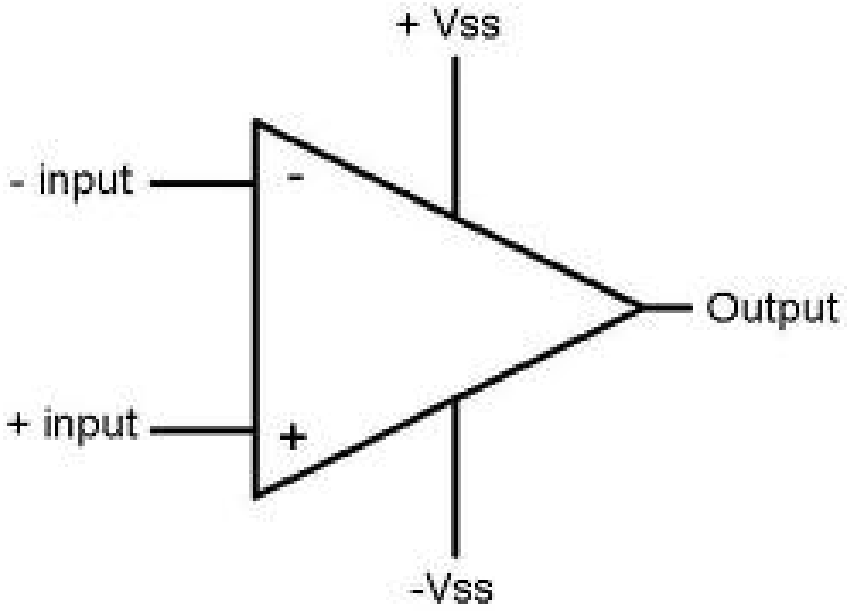
Fact: The peak output of the Electret Microphone is a few hundred uV (micro-Volts)

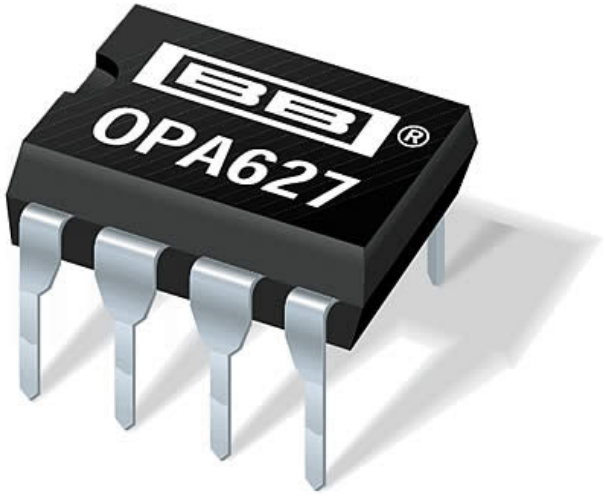


So how can the Arduino measure the signal level of the Electret Microphone?

Op-Amp


+ Vss
- input
-
Output
+ input
+
-Vss


+ Vss
- input
-
Output
+ input
+
-Vss

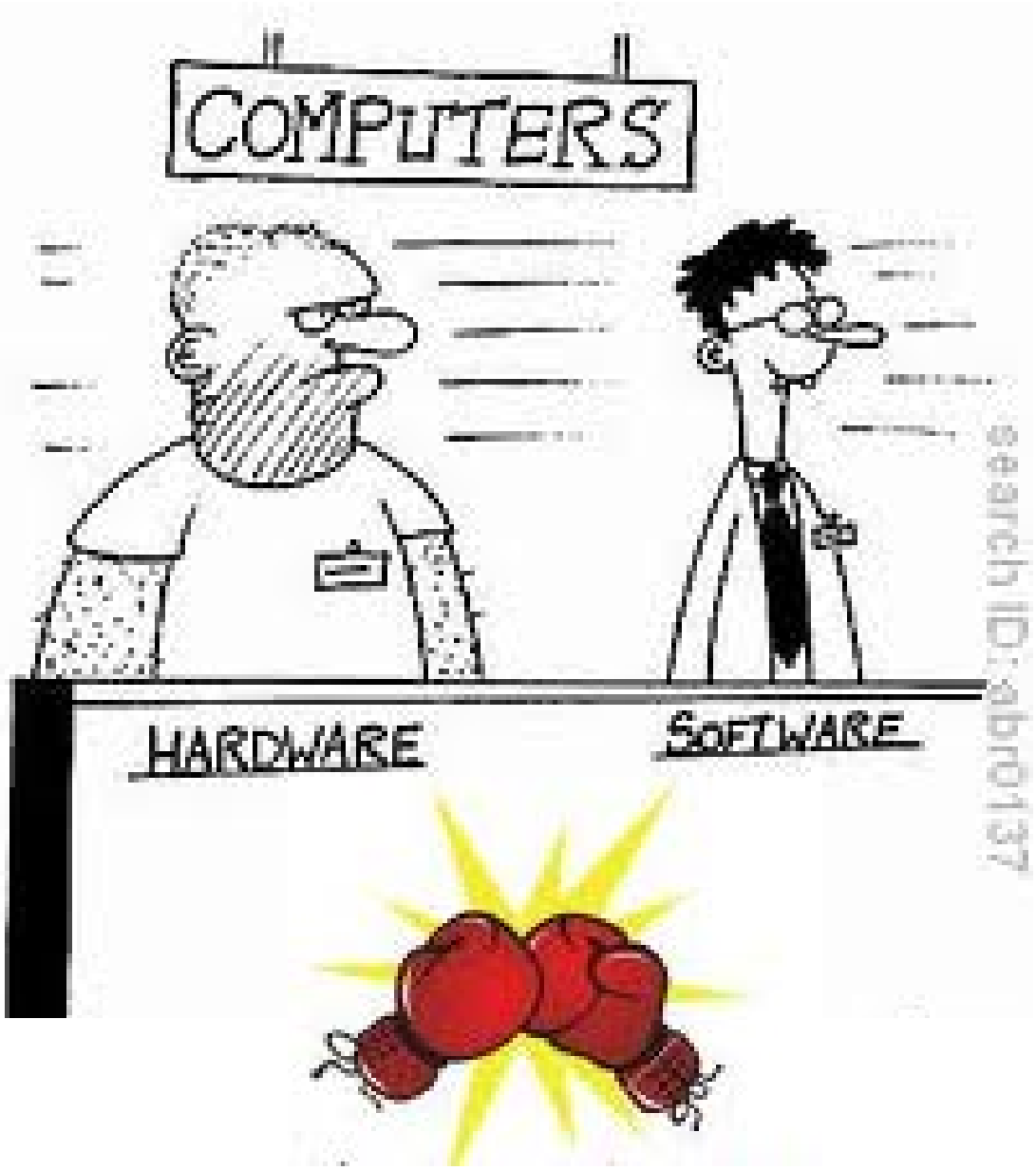

OPA627

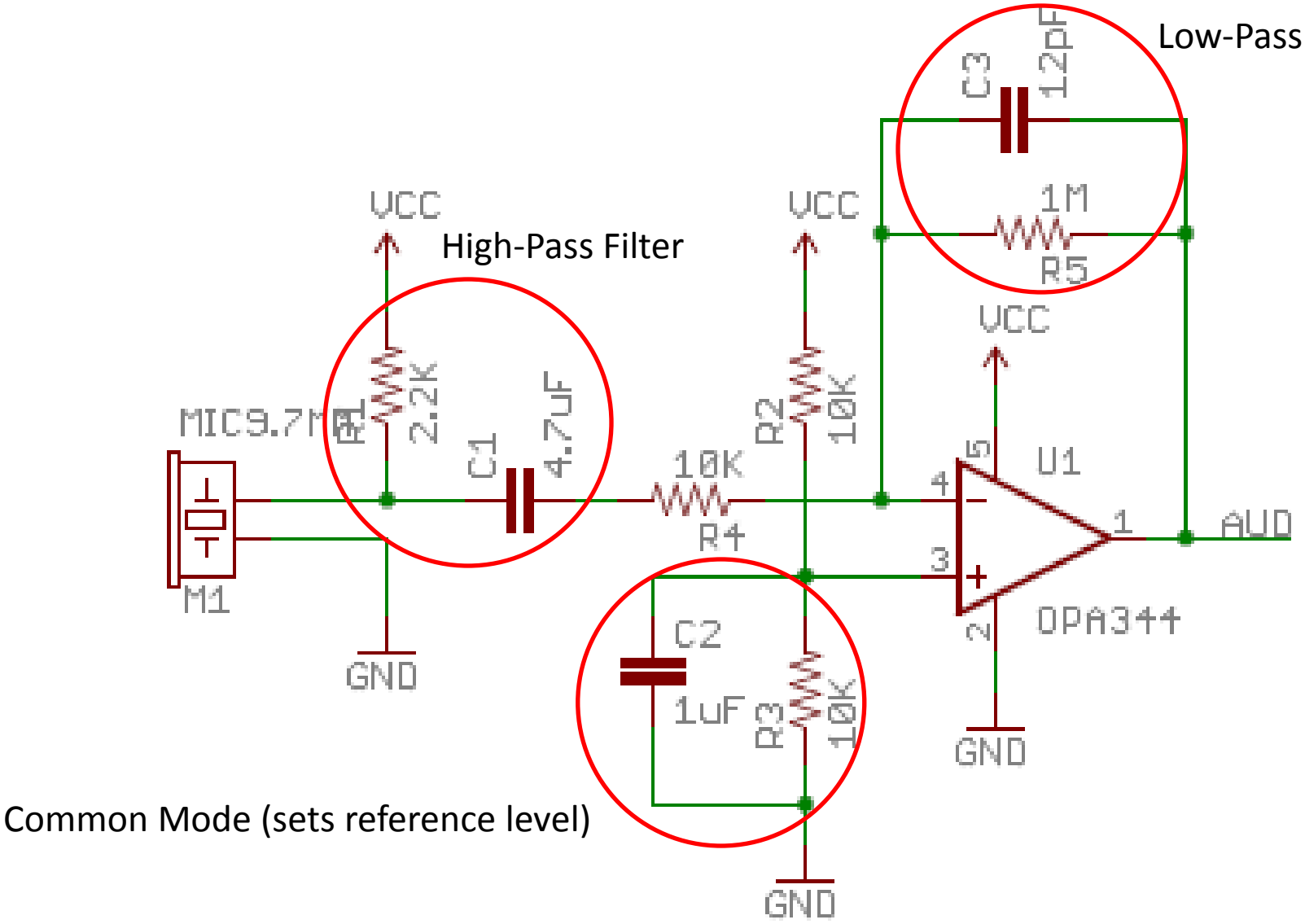
Fact: The electret microphone responds to a larger range of frequencies that we are concerned with.

5V

0V

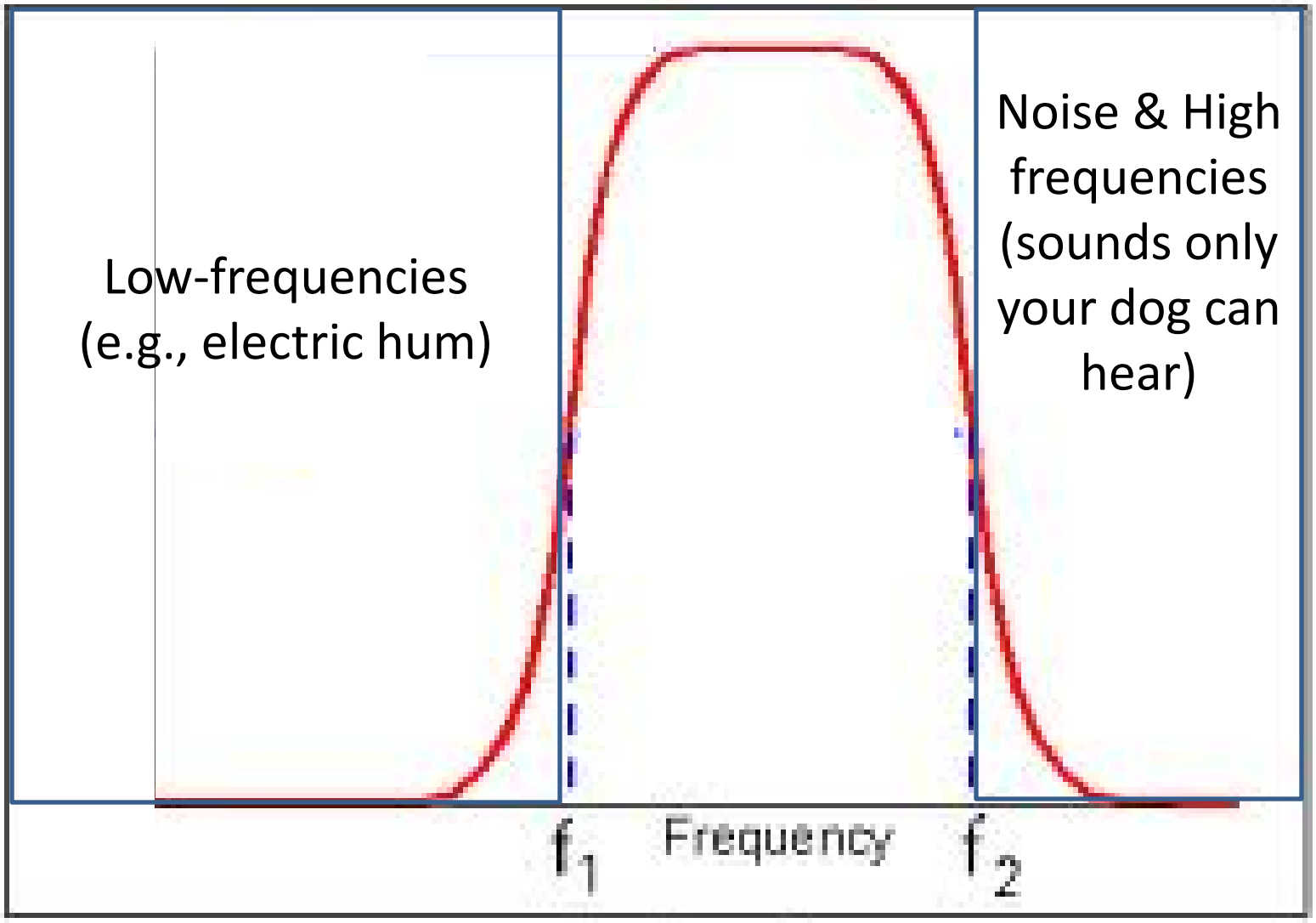
So how can these troublesome frequencies be removed?

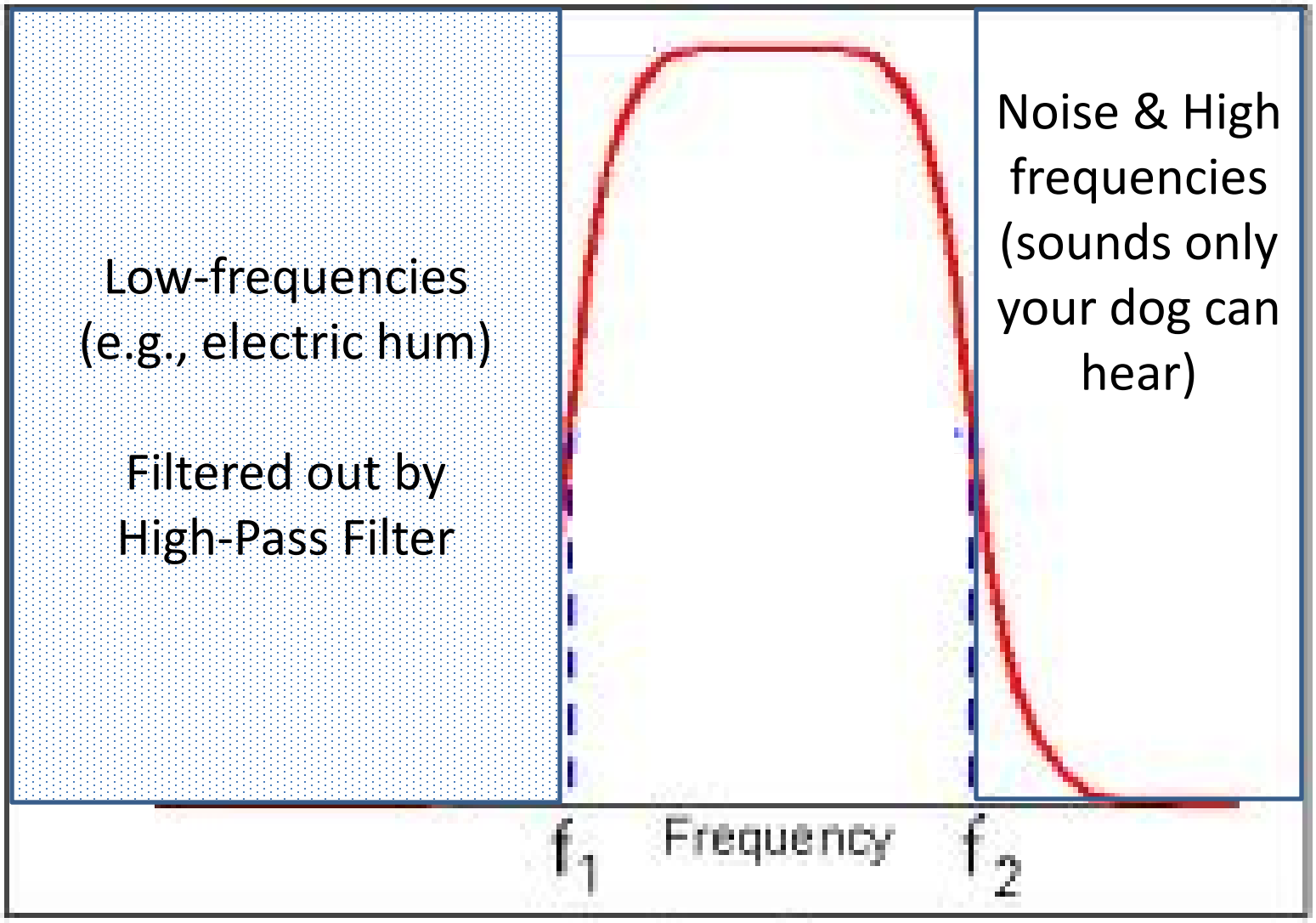

COMPUTERS
HARDWARE
SOFTWARE
search ID: abr0137

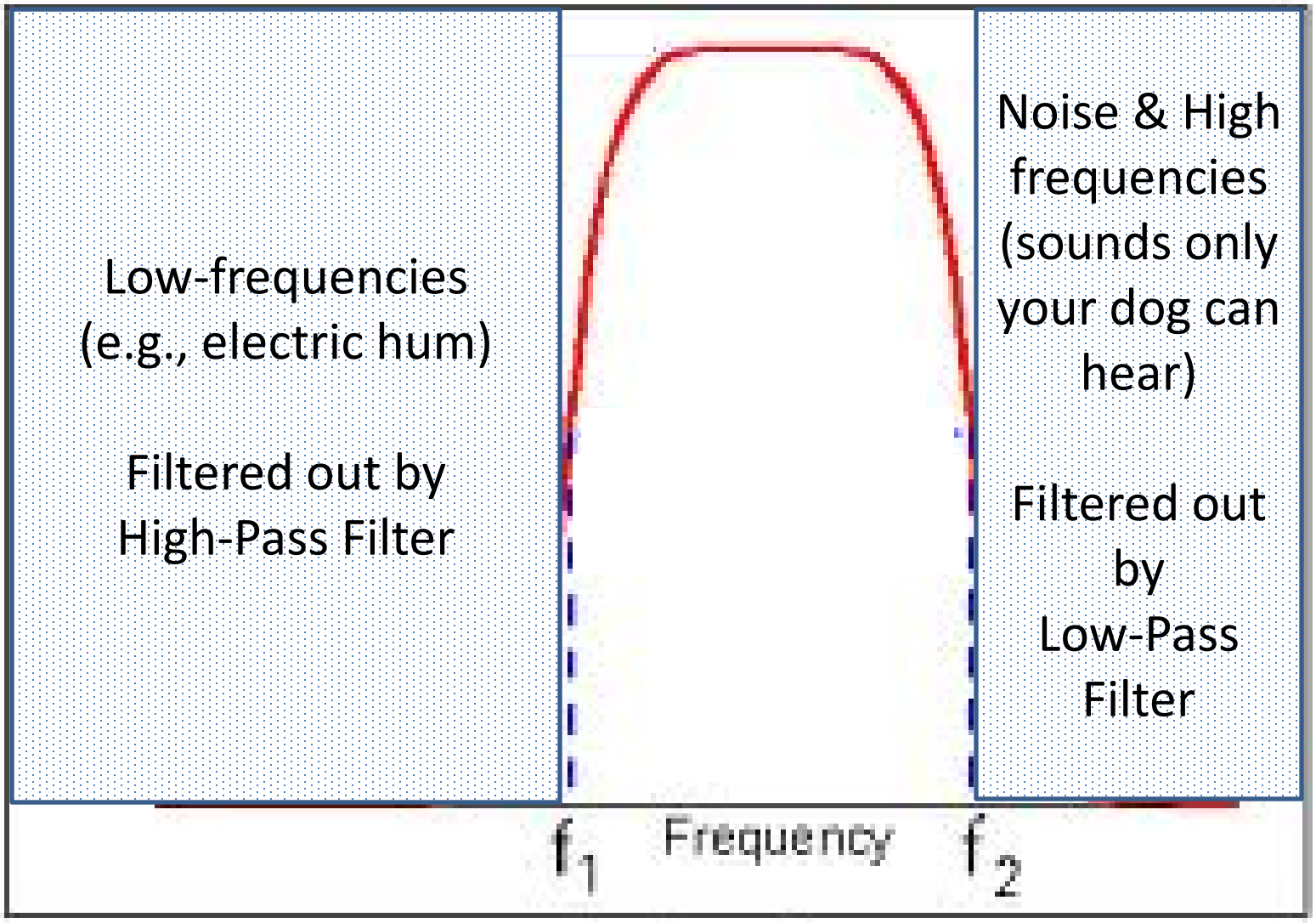


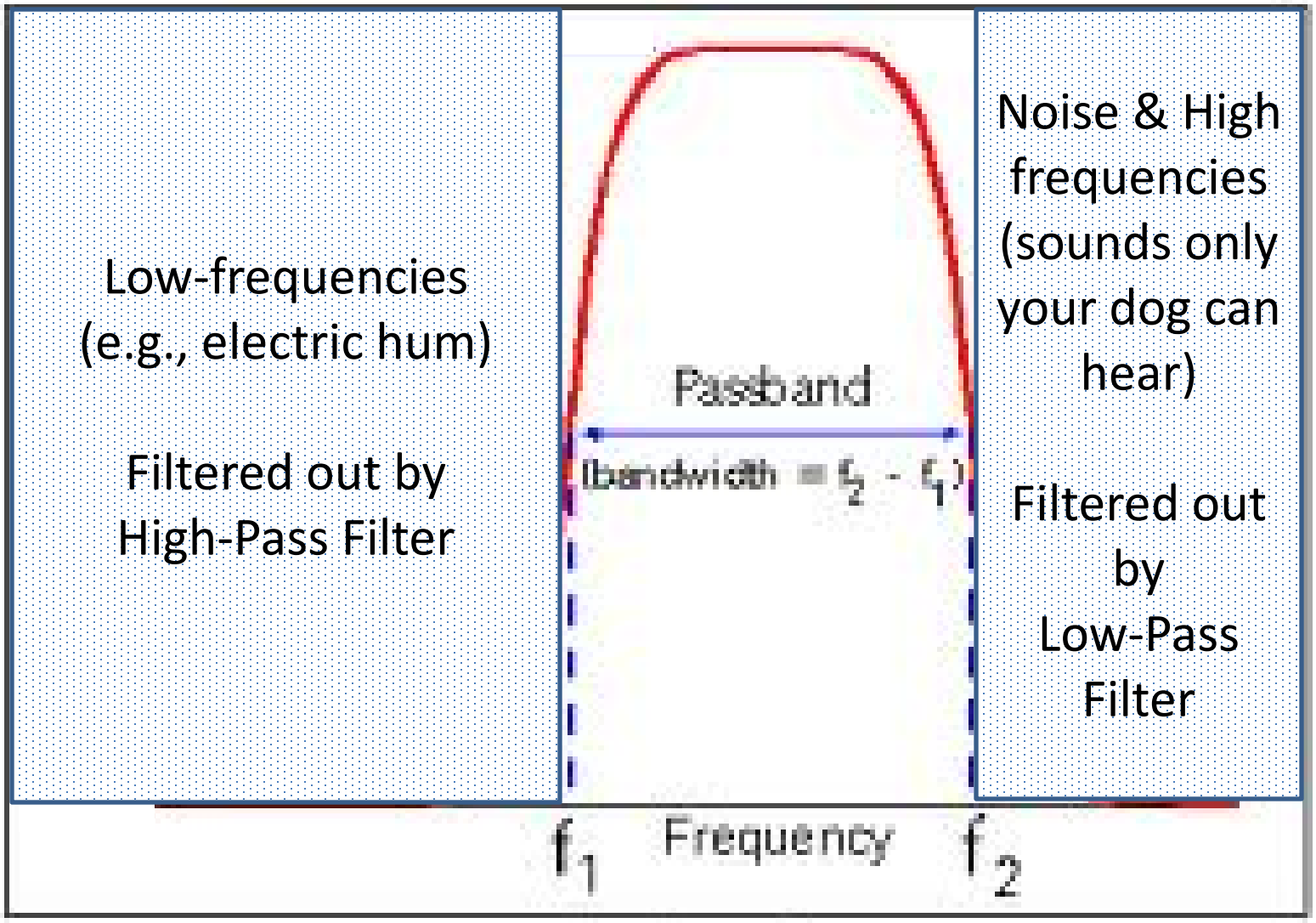
Common Mode (sets reference level)

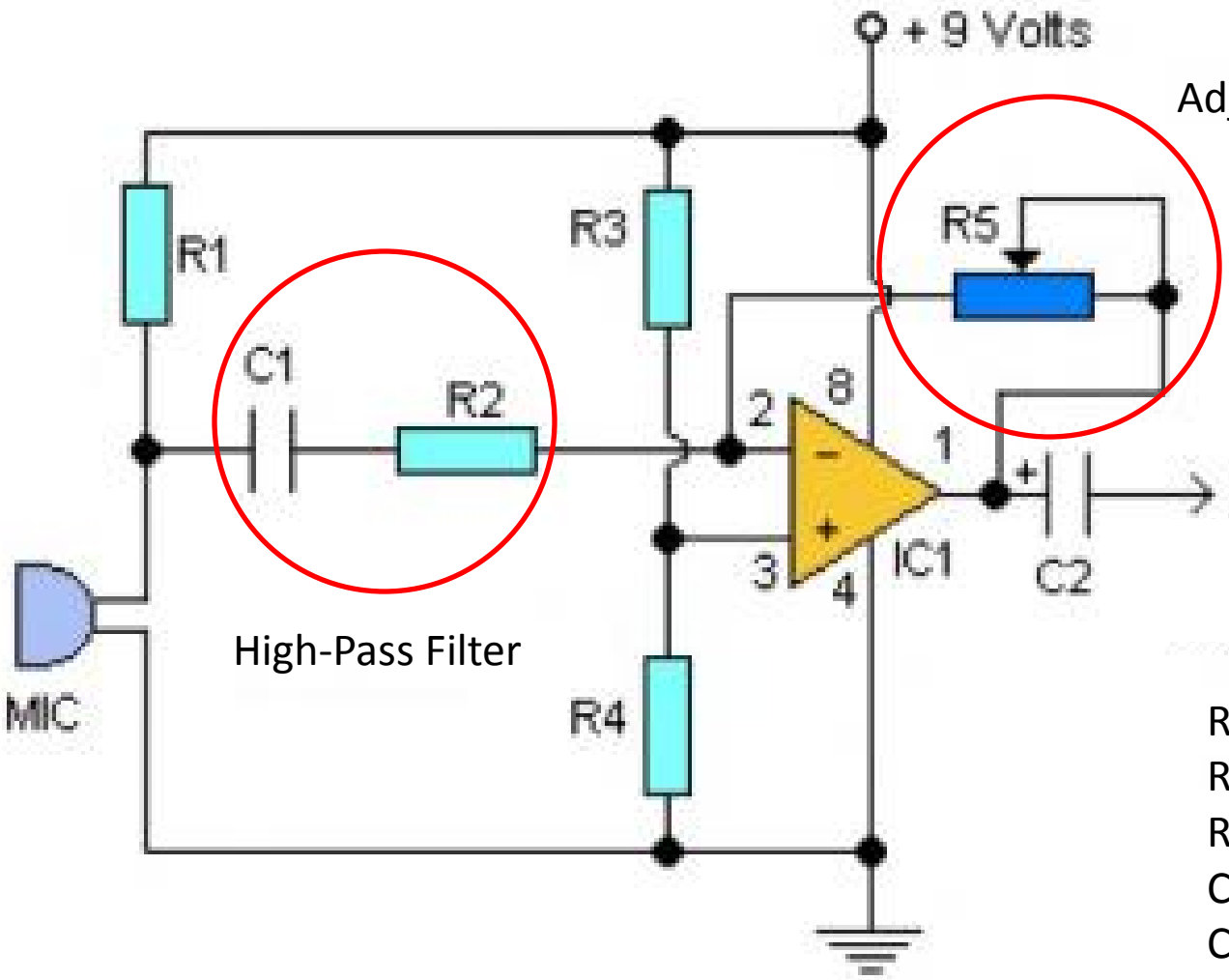
Microphone Pre-amp


Low-frequencies
(e.g., electric hum)
Noise & High frequencies (sounds only your dog can hear)
f1
Frequency
f2


Low-frequencies
(e.g., electric hum)
Filtered out by
High-Pass Filter
Noise & High
frequencies
(sounds only
your dog can
hear)
f1
Frequency
f2


Low-frequencies
(e.g., electric hum)
Filtered out by
High-Pass Filter
Noise & High
frequencies
(sounds only
your dog can
hear)
Filtered out
by
Low-Pass
Filter
f1
Frequency
f2

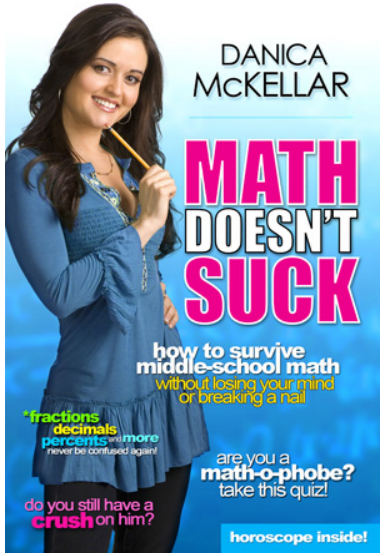

Low-frequencies
(e.g., electric hum)
Filtered out by
High-Pass Filter
Passband
(bandwidth = f2 - f1)
Noise & High frequencies (sounds only your dog can hear)
Filtered out by Low-Pass Filter
f1
Frequency
f2



R1, R3, R4 = 10K
R2 = 1K
R5 = 100K-1M Potentiometer
C1 = 0.1uF
C2 = 4.7uF/16V
IC1 = LM358 dual op-amplifier
Mic = Electret Microphone

Microphone Pre-amp

$$f_c = \frac{1}{2\pi RC}$$



Cut-off Frequency

where resistance in ohms and capacitance in farads yields the frequency in Hz.

Fc=1000 (1K), C = 0.0000001 (0.1uF) = 159Hz

Fc=2200 (2.2K), C = 0.0000047 (4.7uF) = 15.4Hz

Fc=1000000 (1M), C = 0.000000000012 (12 pF) = 13269Hz