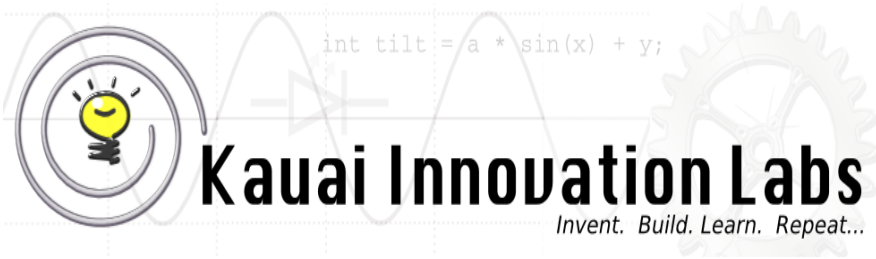
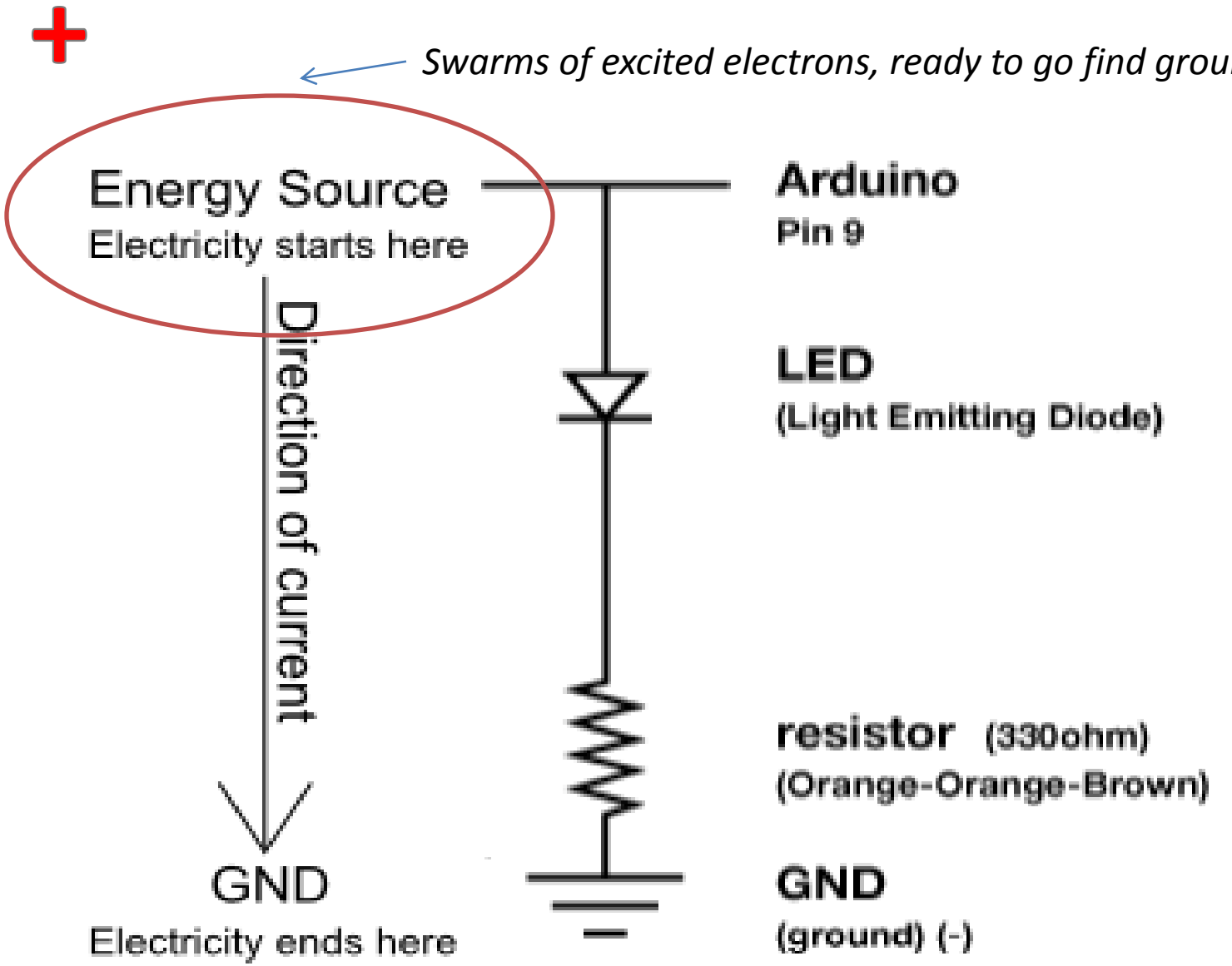
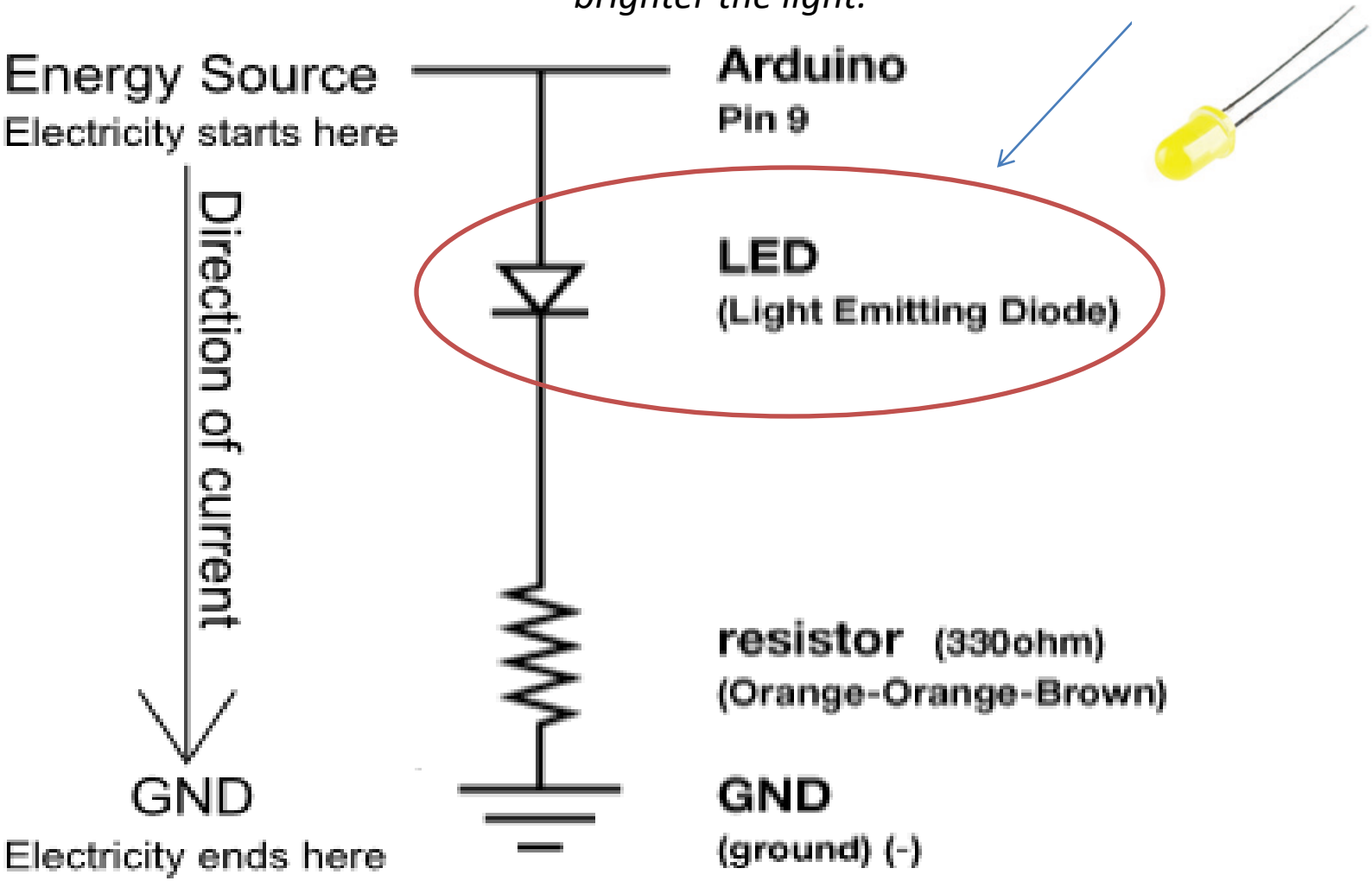
Inventing… 2

with Software and Electronics

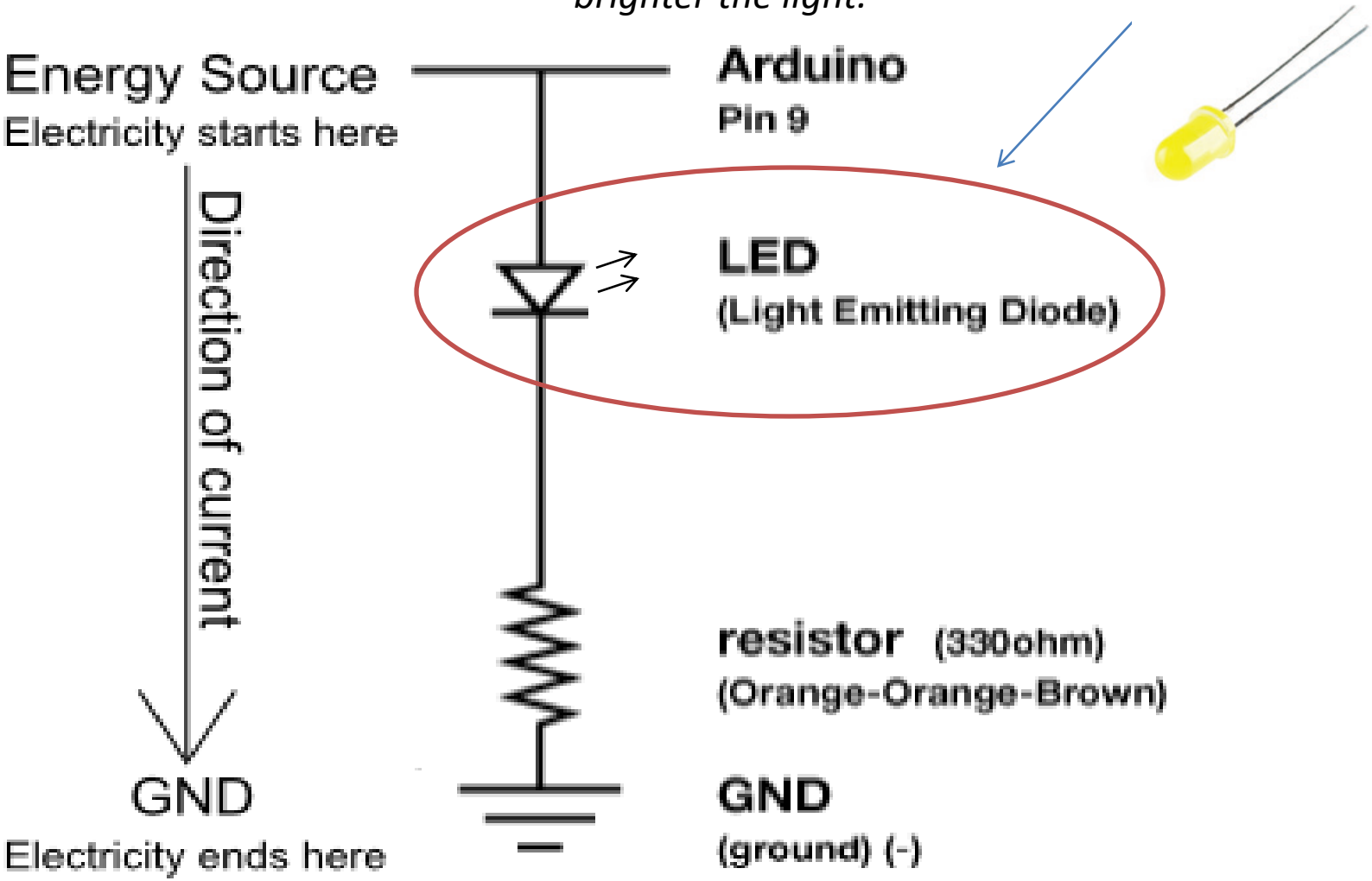


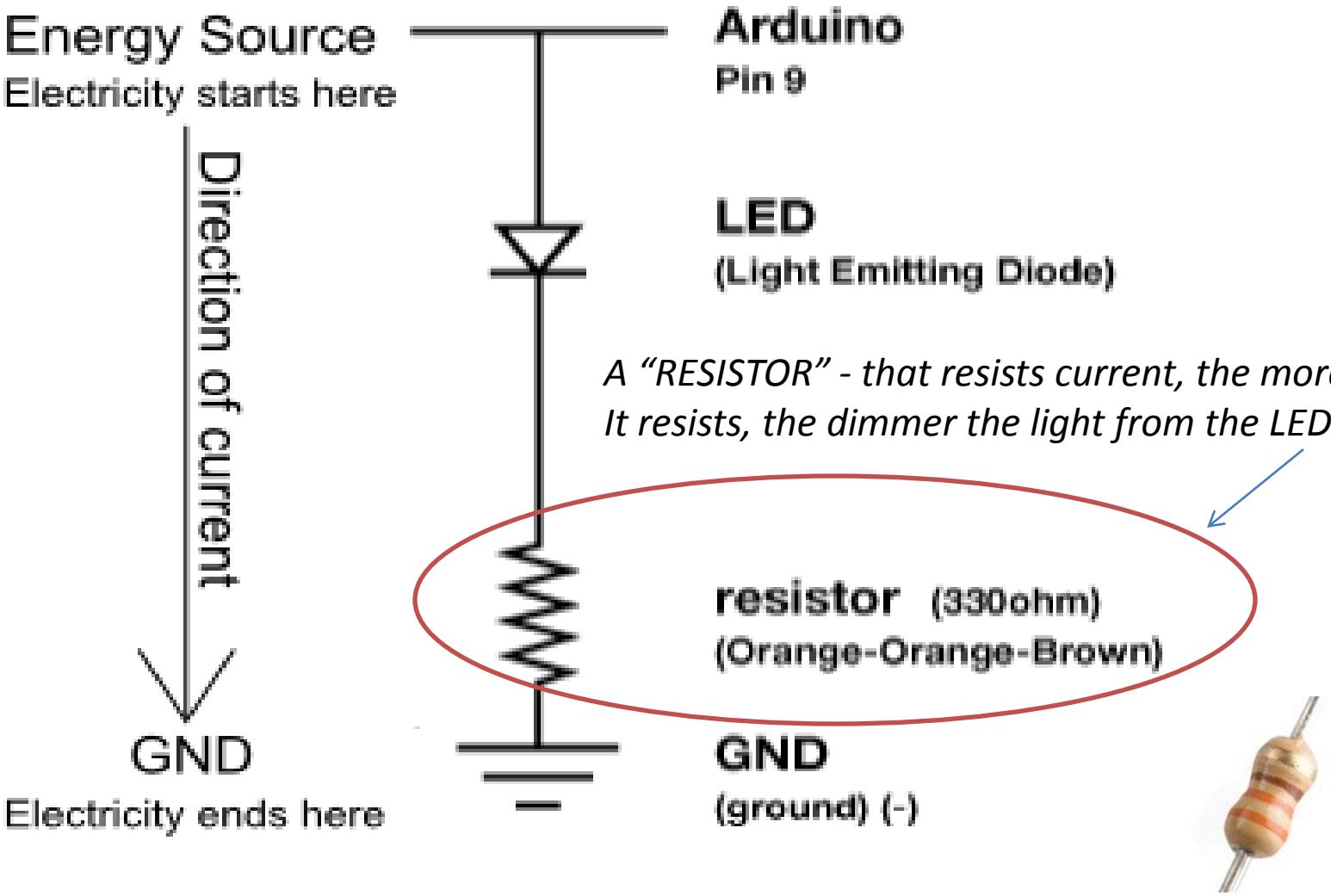

Swarms of excited electrons, ready to go find ground
Energy Source
Electricity starts here
Direction of current
GND
Electricity ends here
Arduino
Pin 9
LED
(Light Emitting Diode)
resistor (330ohm)
(Orange-Orange-Brown)
GND
(ground) (-)

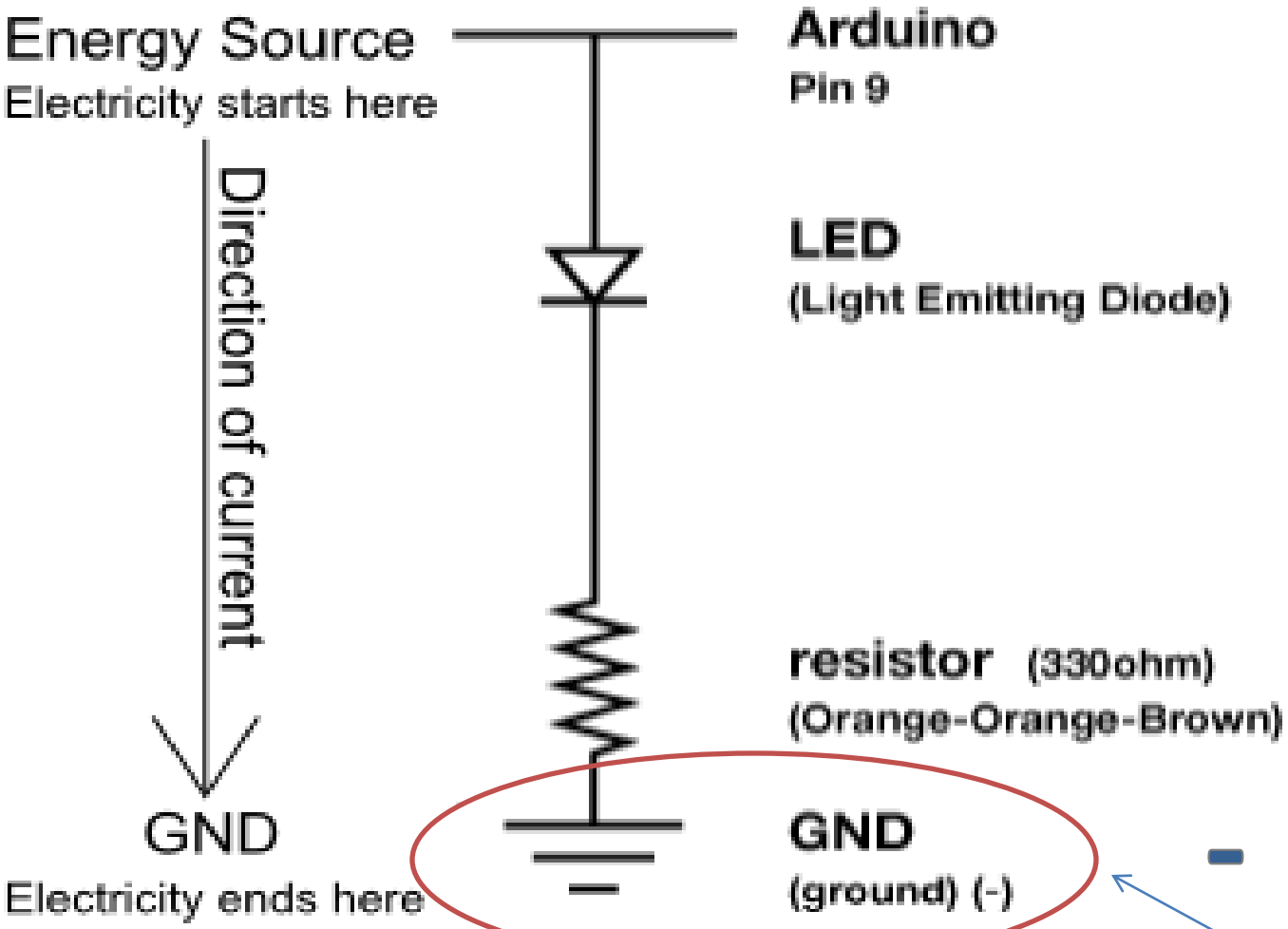
A –one-way-street (DIODE) which emits light (photons) when current passes through. The greater the current through the diode, the brighter the light.



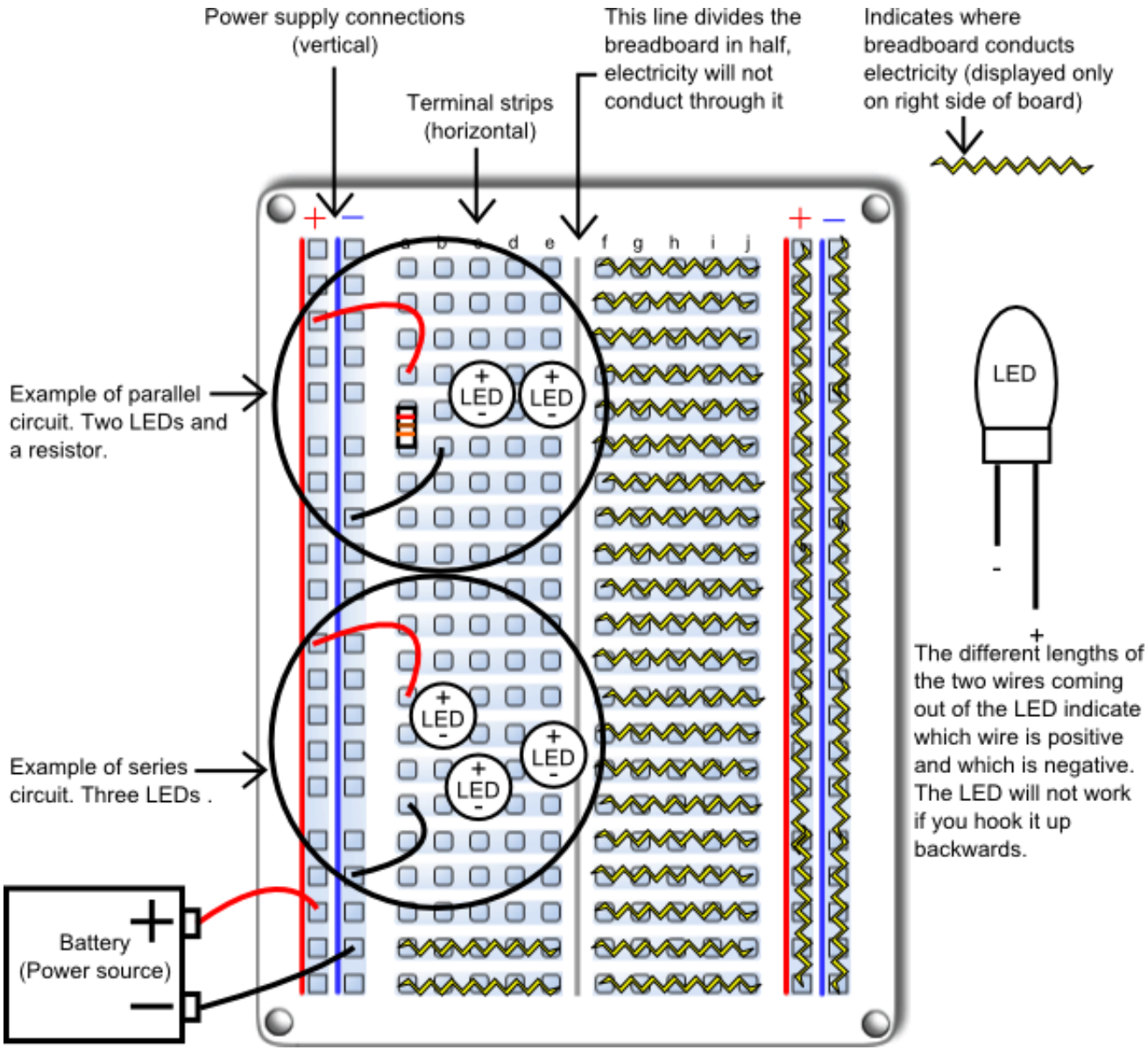
A –one-way-street (DIODE) which emits light (photons) when current passes through. The greater the current through the diode, the brighter the light.



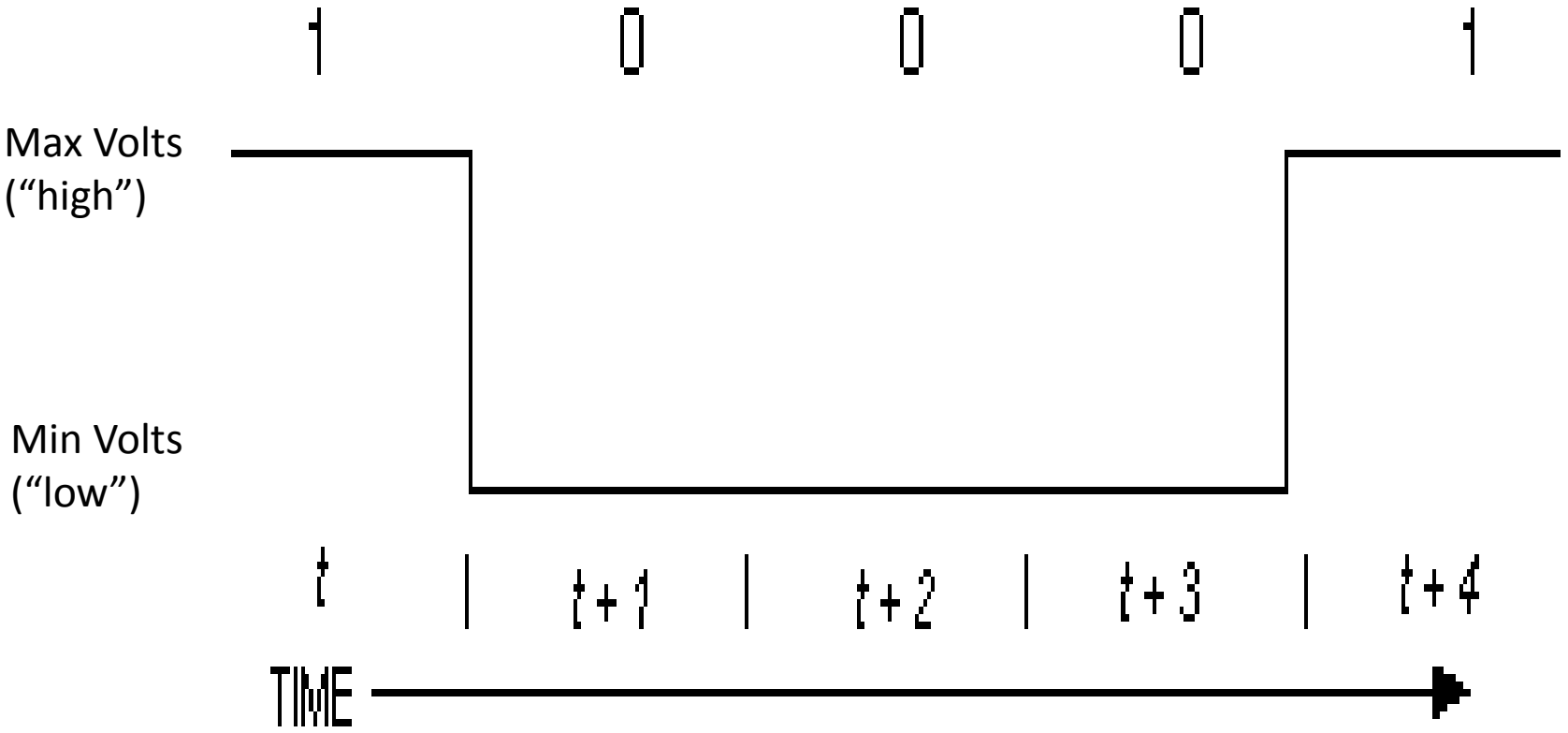

Energy Source
Electricity starts here
Direction of current
GND
Electricity ends here
Arduino
Pin 9
LED
(Light Emitting Diode)
A "RESISTOR" - that resists current, the more
It resists, the dimmer the light from the LED
resistor (330ohm)
(Orange-Orange-Brown)
GND
(ground) (-)



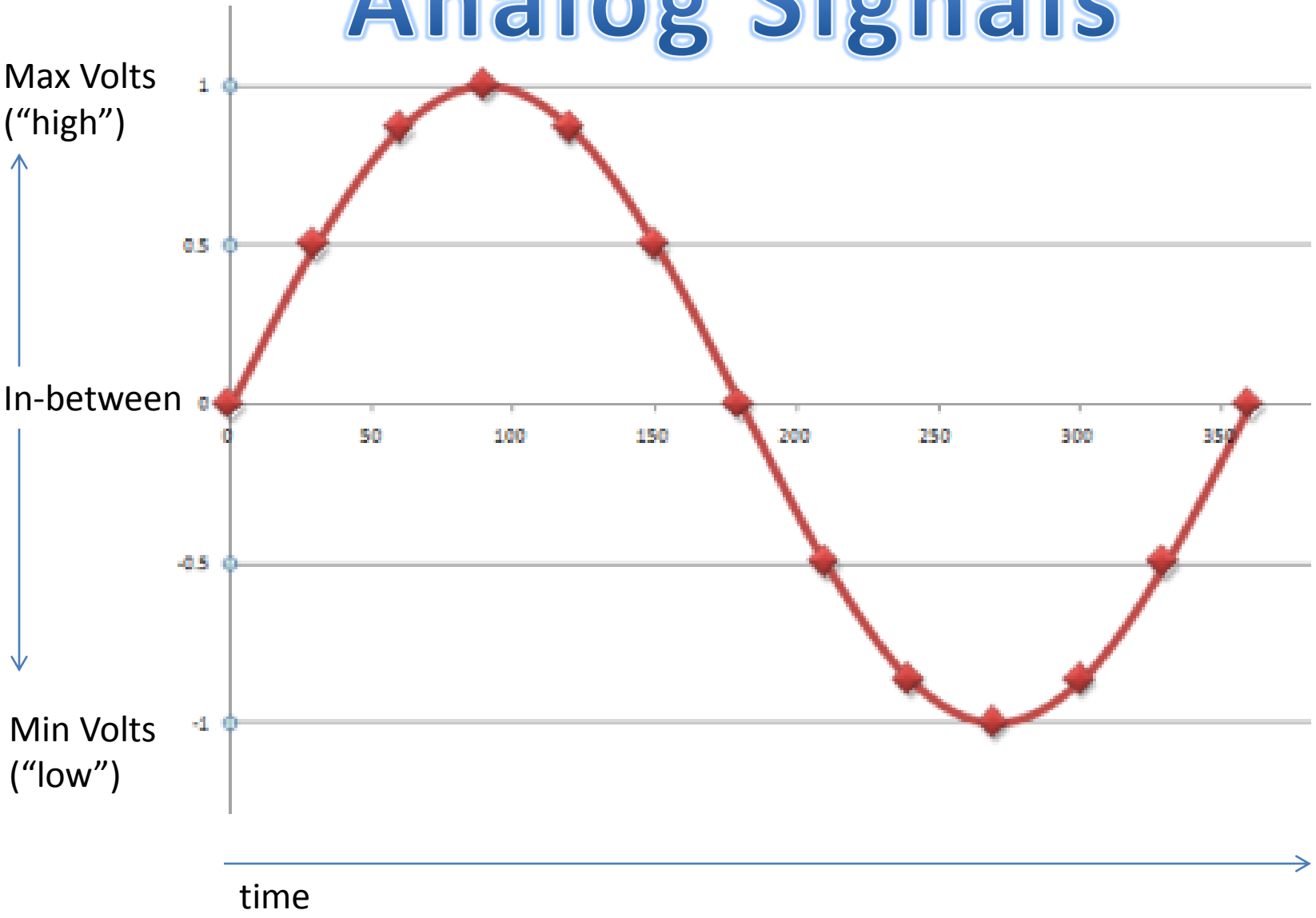
"GROUND" – nirvana for electrons weary from Travel…


Power supply connections (vertical)
Terminal strips (horizontal)
This line divides the breadboard in half, electricity will not conduct through it
Indicates where breadboard conducts electricity (displayed only on right side of board)
Example of parallel circuit. Two LEDs and a resistor.
Example of series circuit. Three LEDs .
Battery (Power source)
LED
The different lengths of the two wires coming out of the LED indicate which wire is positive and which is negative. The LED will not work if you hook it up backwards.

Digital Signals



Analog Signals

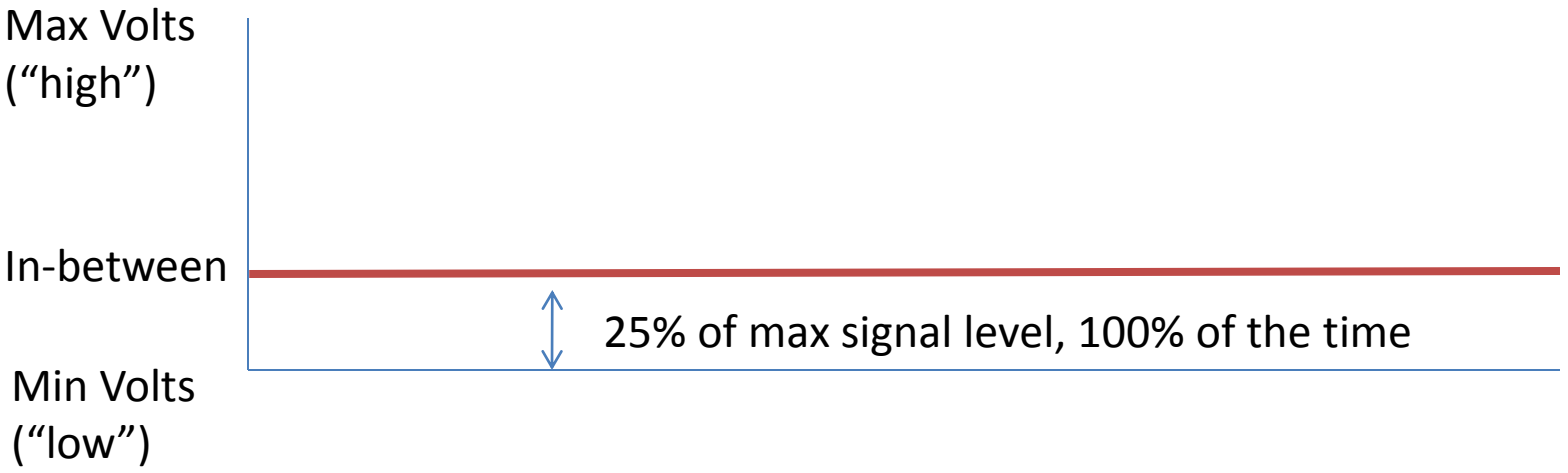


- *Arduino accepts analog and digital inputs,* ***but can only output digital signals***

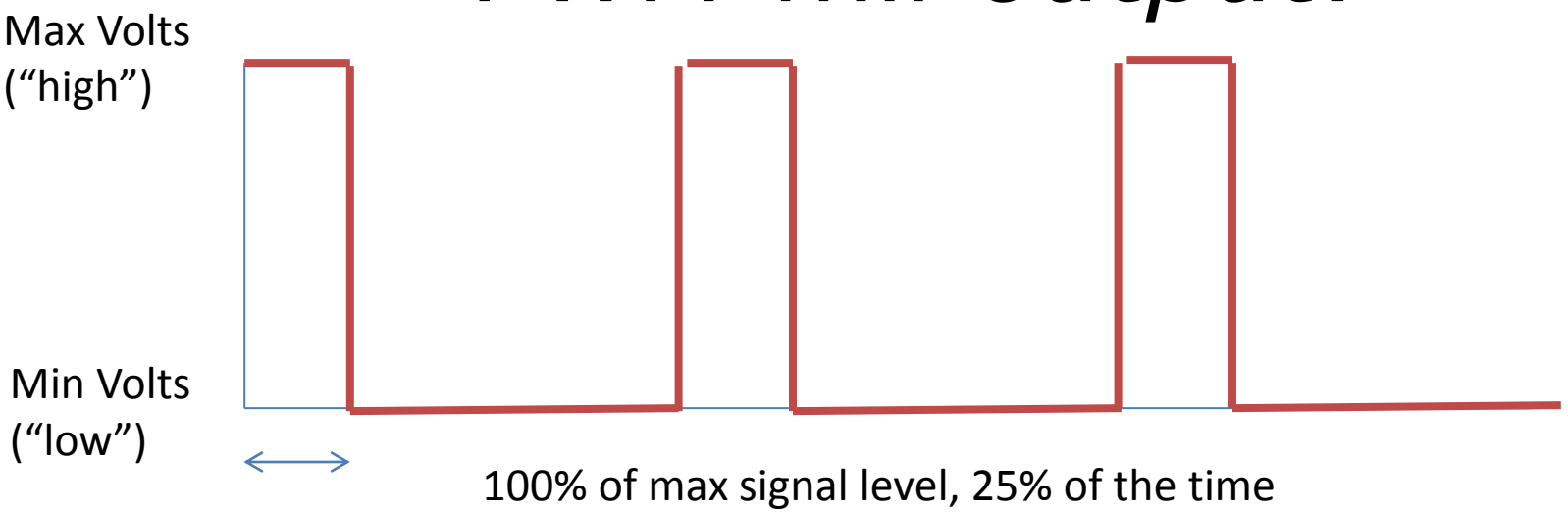
- *So how can we send vary the brightness of a LED if we can only output digital signals?*

PWM

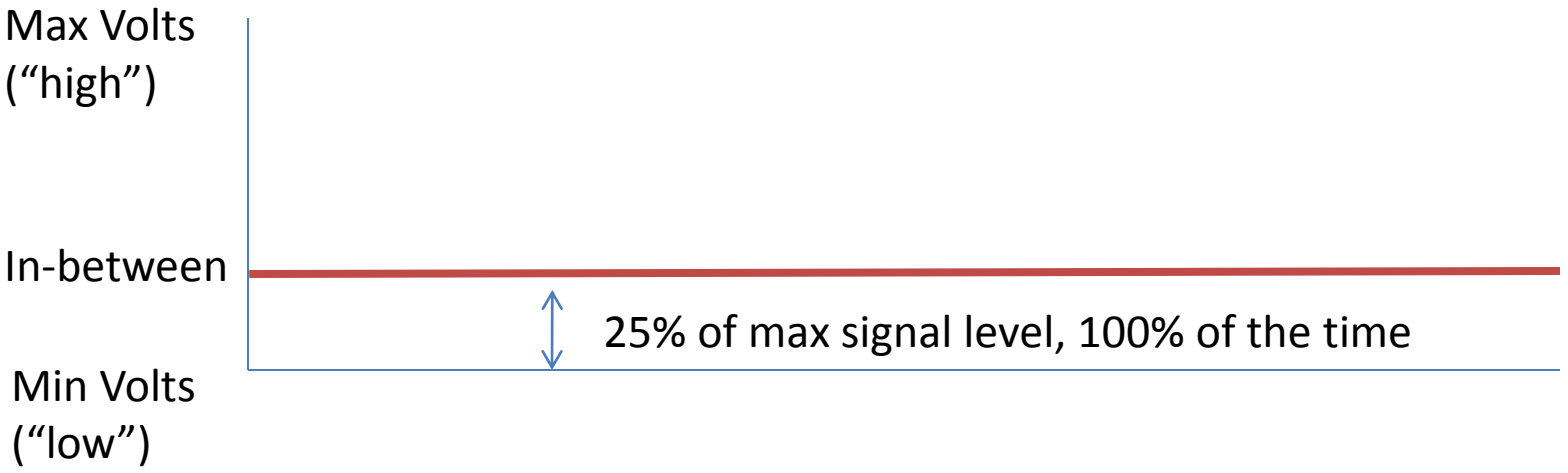
- So instead of:



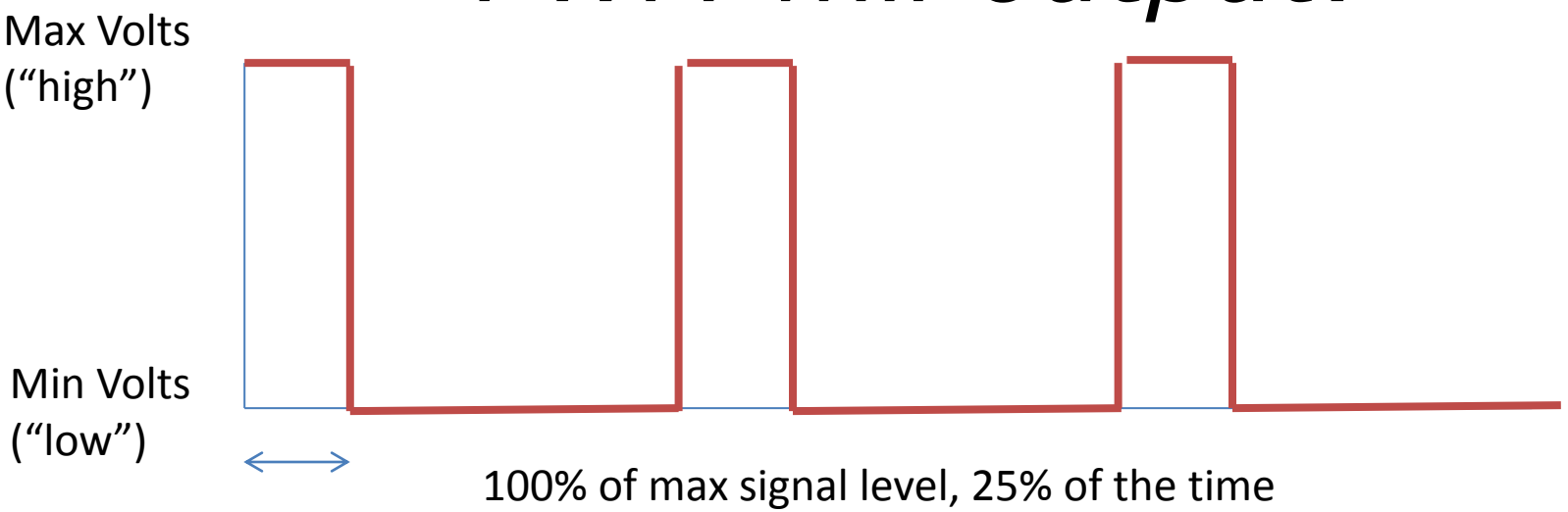
- PWM will output:



- So instead of:



- PWM will output:



THE AVERAGE SIGNAL LEVEL IS THE SAME

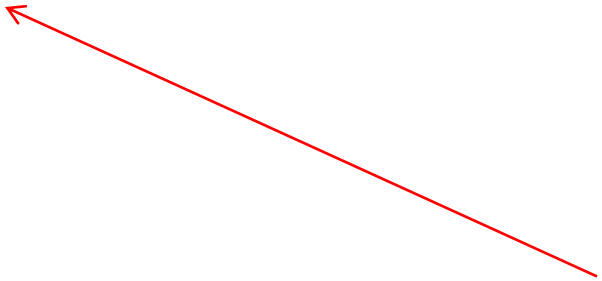
Code:

Int ledPin = 3;

variable

Code:

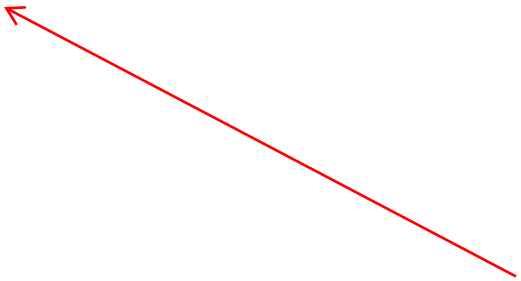
int ledPin = 3;



variable type

Code:

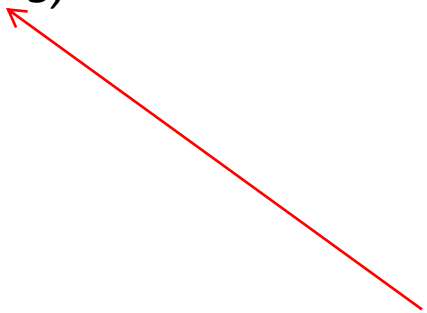
int ledPin = 3;



variable name (you get to make this up)

Code:

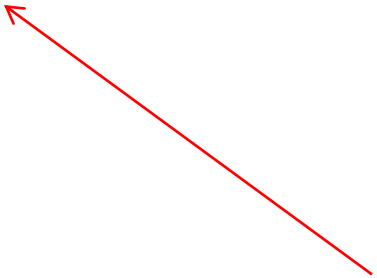
int ledPin = 3;



operator (assignment)

Code:

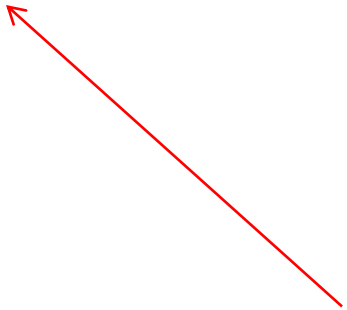
int ledPin = 3;



value

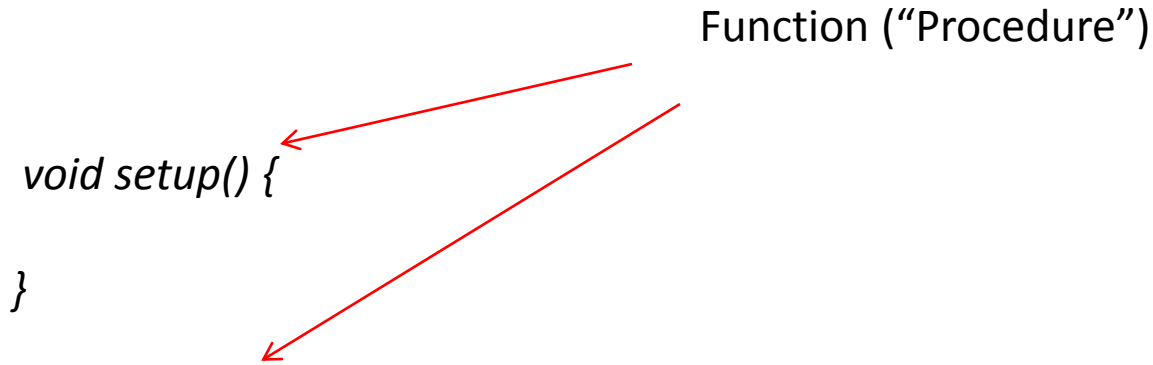
Code:

int ledPin = 3;

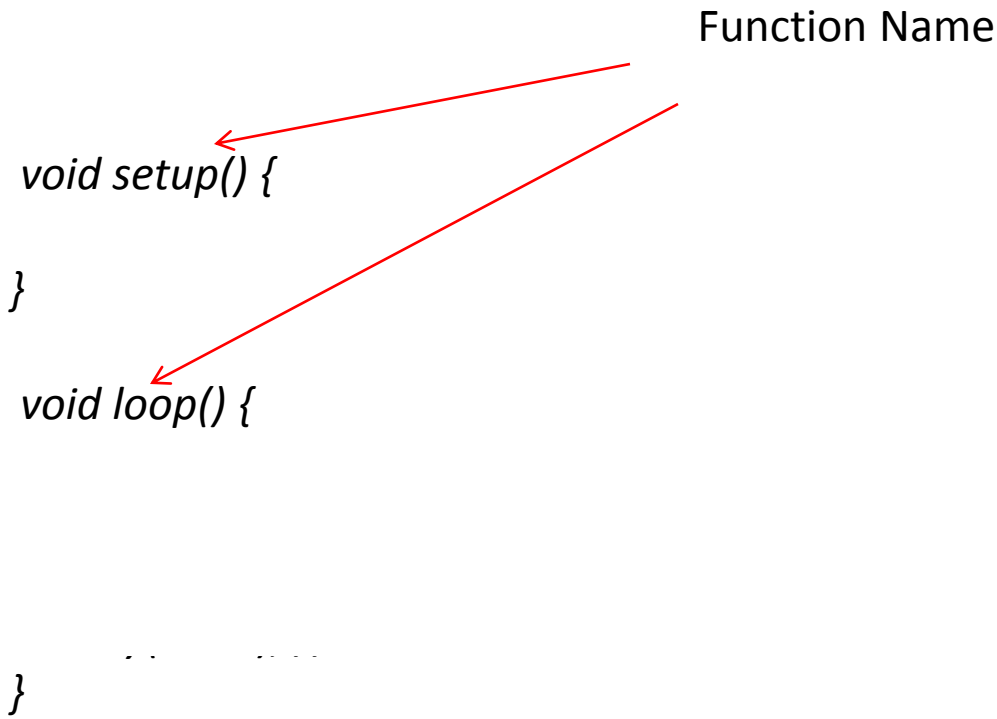


semicolon – tells the compiler (which translates code into instructions the computer understands) that the statement is complete.

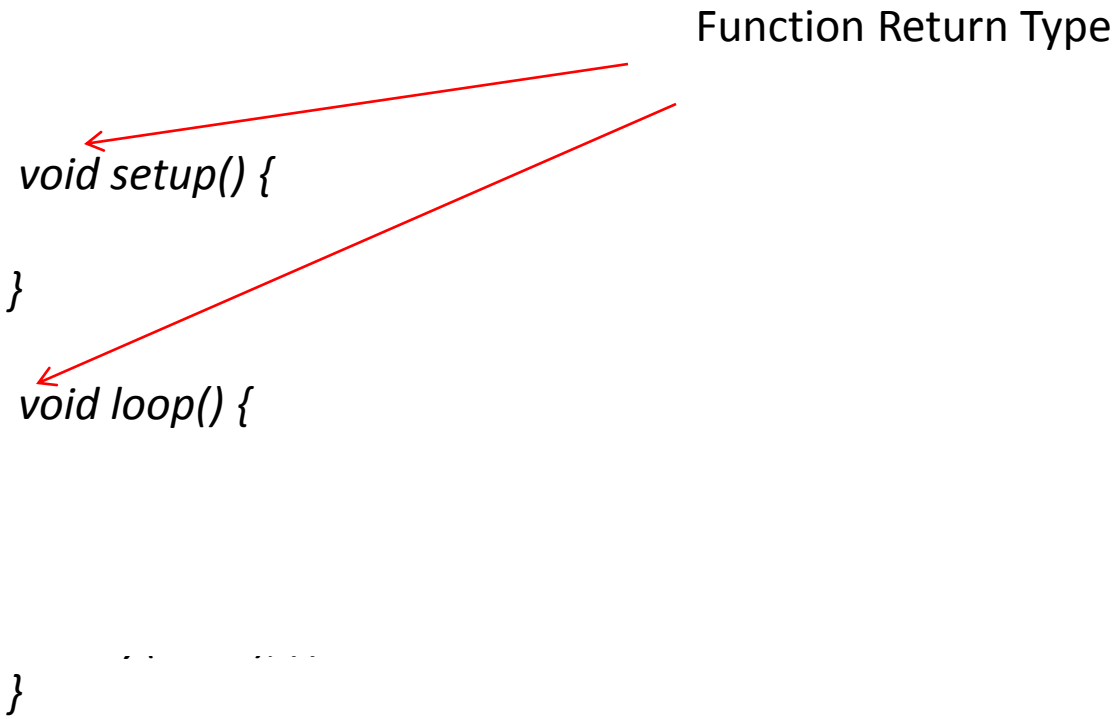
Code:



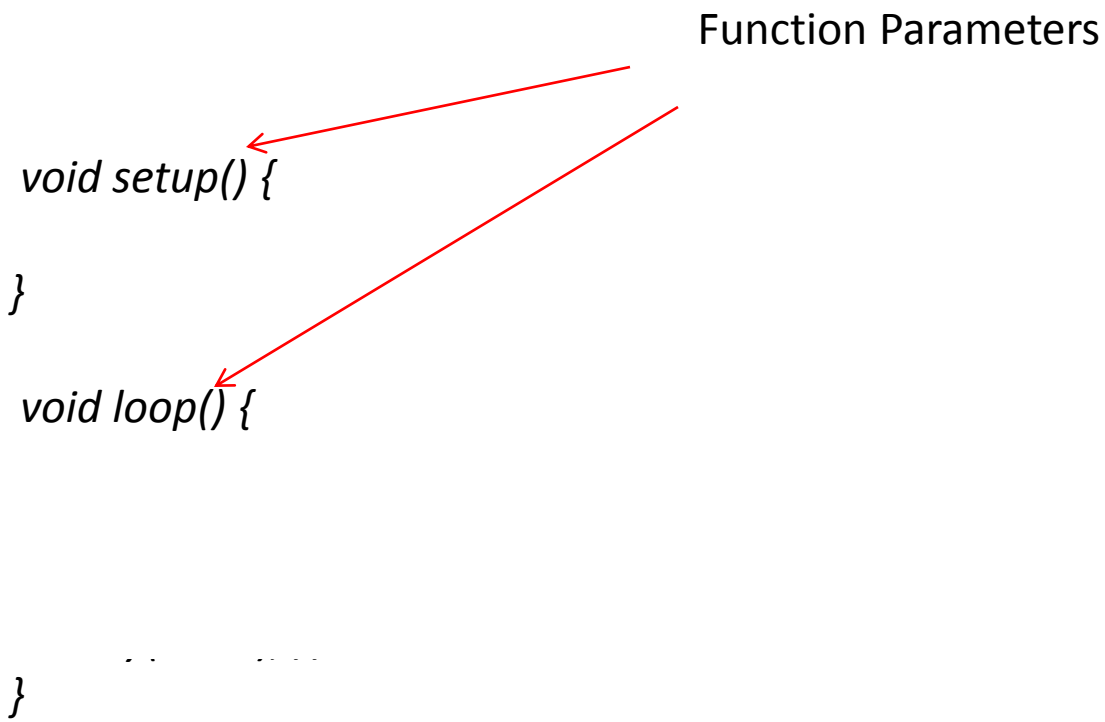
Code:



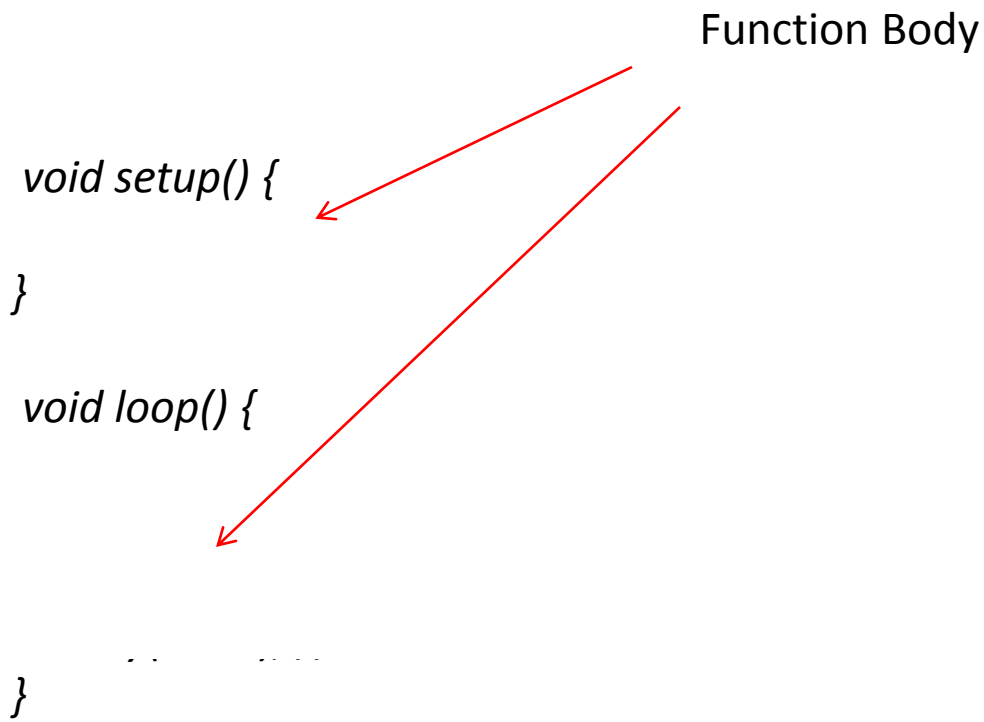
Code:



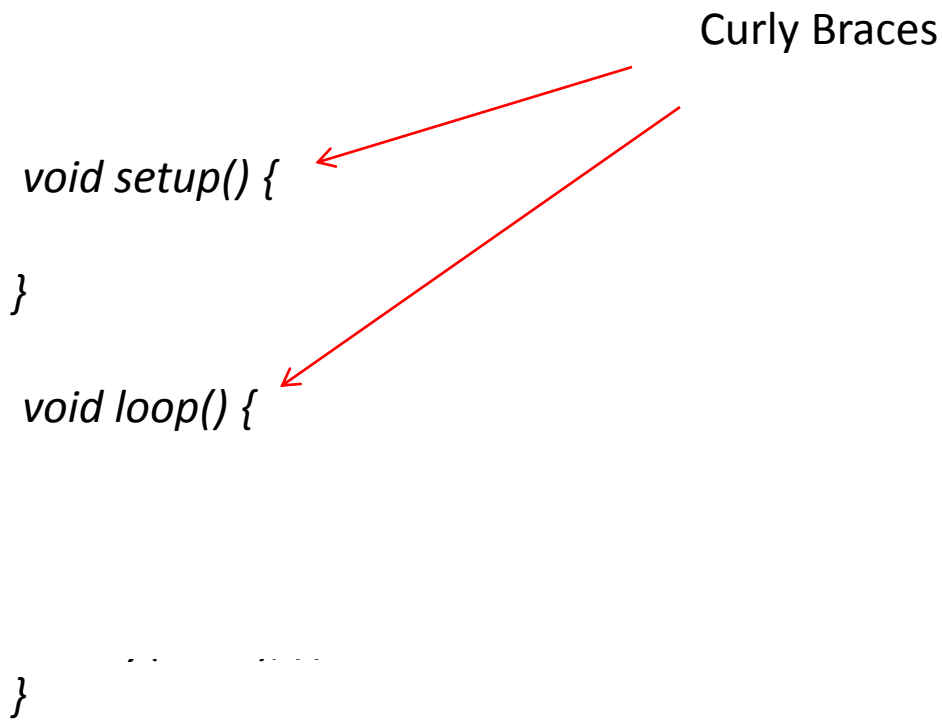
Code:



Code:

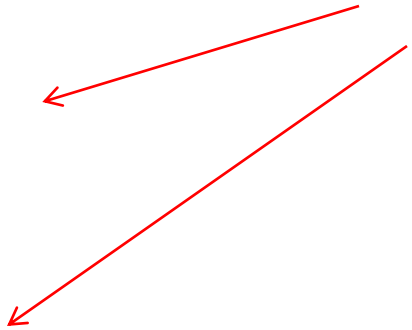


Code:



Code:

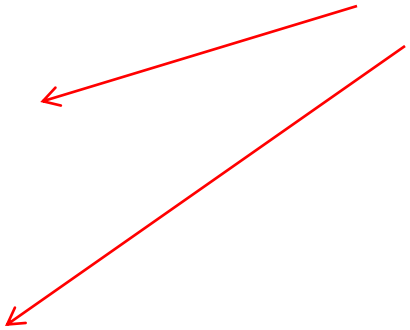
Mr. Libert is picky about Curly Braces



```
 void setup()
{

}

 void loop()
{



}
```

Code:

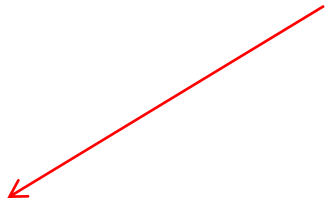
Each sketch must have a setup() and a loop() function



```
 void setup()
{

}

 void loop()
{



}
```

Code:

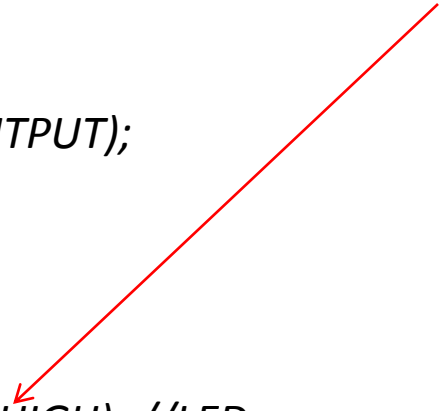
setup() is where the Arduino pins are configured.



```
 void setup()
{
   pinMode(ledPin, OUTPUT);
}

 void loop()
{




}
```

Code:

loop() is where the interesting stuff happens.



```
 void setup()
{
  pinMode(ledPin, OUTPUT);
}

 void loop()
{
  digitalWrite(ledPin, HIGH); //LED on
  delay(1000); // wait second
  digitalWrite(ledPin, LOW); //LED off
  delay(1000); // wait second
}
```

Things to remember about Analog Inputs:

- Analog Input use the Analog In pins

- To receive an Analog signal use:

analogRead(pinNumber);

- Analog Input values range from 0 to 1023 (1024 values because it uses 10 bits, 2^{10})

Things to remember about Analog Outputs:

- Analog Output uses the PWM pins

- To send a PWM (simulated Analog) signal use:

analogWrite(pinNumber, value);

- PWM Output values range from 0 to 255 (256 values because it uses 8 bits, 2^8)

Things to remember about Digital:

- Digital Input/Output uses the Digital pins, but Analog In pins can be used as Digital

- To receive a Digital signal use:

    digitalRead(pinNumber);

- To send a Digital signal use:

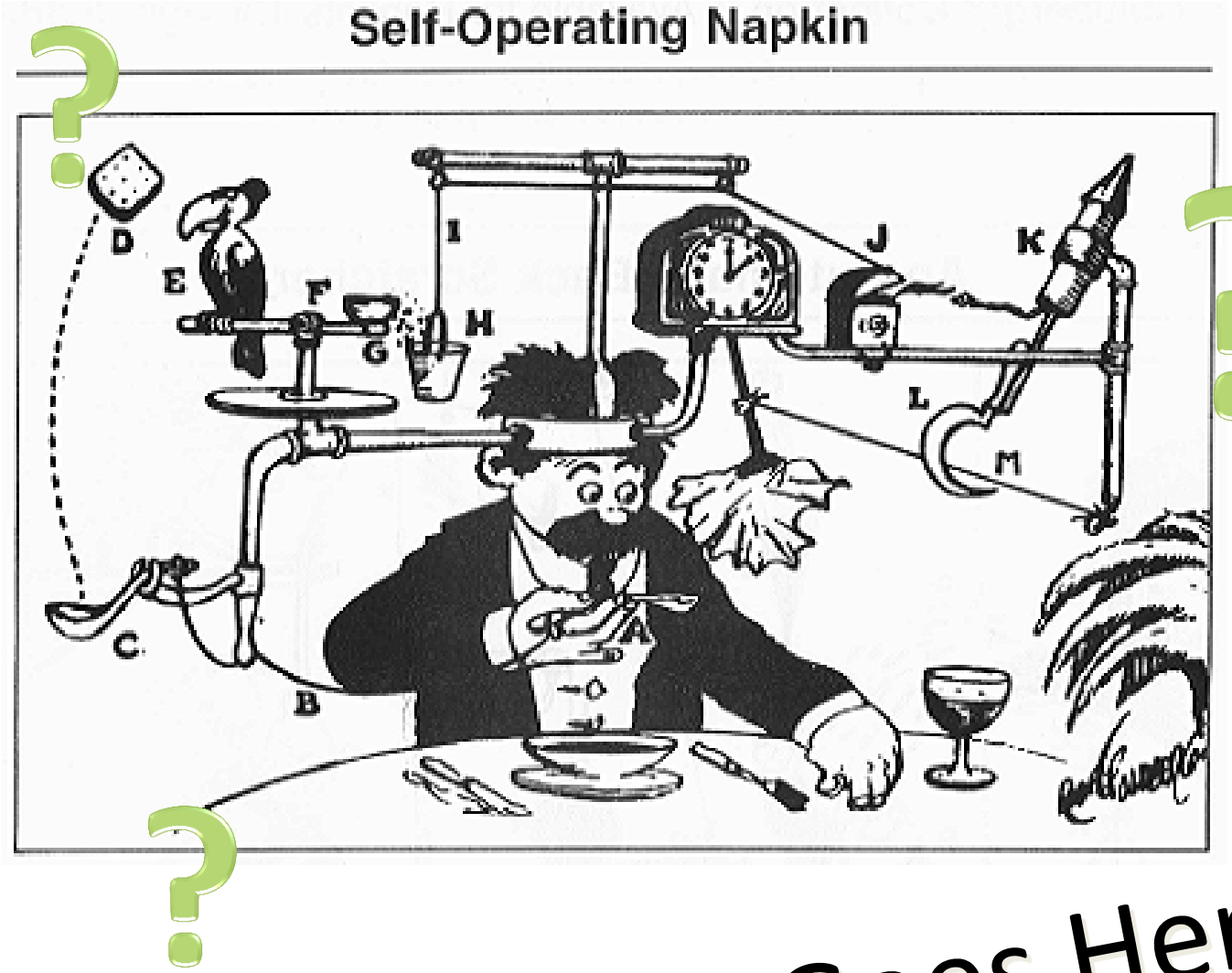
    digitalWrite(pinNumber, value);

- Digital Inputs/Outputs are either HIGH or LOW

Loops
&
Motors
&
Amps

(Oh, Boy!)

But first.....



Your Project Goes Here

Loops
&
Motors
&
Amps

(Oh, Boy!)

KEY FACT 1: Software can be written without loops

KEY FACT 2: Software can be simplified with loops

KEY FACT 2: Software can be simplified with loops

Lazy Programmer

KEY FACT 2: Software can be simplified with loops

Lazy Programmer
(K I S S)

```
Int ledPins[16] = {2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17};
void setup()
{
  pinMode(ledPins[0],OUTPUT);
  pinMode(ledPins[1],OUTPUT);
  pinMode(ledPins[2],OUTPUT);
  pinMode(ledPins[3],OUTPUT);
  pinMode(ledPins[4],OUTPUT);
  pinMode(ledPins[5],OUTPUT);
  pinMode(ledPins[6],OUTPUT);
  pinMode(ledPins[7],OUTPUT);
  pinMode(ledPins[8],OUTPUT);
  pinMode(ledPins[9],OUTPUT);
  pinMode(ledPins[10],OUTPUT);
  pinMode(ledPins[11],OUTPUT);
  pinMode(ledPins[12],OUTPUT);
  pinMode(ledPins[13],OUTPUT);
  pinMode(ledPins[14],OUTPUT);
  pinMode(ledPins[15],OUTPUT);
}
```

No Loop

```
void setup()
{
    for ( int i = 0; i < 16; i++ )
   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

For Loop

```
void setup()
{



}
```

Dear Compiler,

For Loop

```
void setup()
{

   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

Dear Compiler,

Please execute the code within the upcoming curly braces

For Loop

```
void setup()
{
   for (                          )
   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

Dear Compiler,

Please execute the code within the upcoming curly braces ***for as long it takes,***

For Loop

```
void setup()
{
   for ( int i = 0;                )
   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

Dear Compiler,

Please execute the code within the upcoming curly braces for as long it takes, ***until the "loop counter" variable, which is initially zero,***

For Loop

```
void setup()
{
   for ( int i = 0; i < 16;       )
   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

Dear Compiler,

Please execute the code within the upcoming curly braces for as long it takes, until the "loop counter" variable, which is initially zero, ***reaches it's limit.***

Yours truly,

- Programmer-for-life

For Loop

```
void setup()
{
   for ( int i = 0; i < 16;      )
   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

Dear Programmer,

*That sounds like a good idea. But **if the loop counter is not incremented, the program will "hang forever" and never end.** Are you sure that's what you want?*

Yours truly,

- Compiler

For Loop

```
void setup()
{
   for ( int i = 0; i < 16; i++ )
   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

Dear Compiler,

Wow, thanks for reviewing my code! That would have been a nasty bug! I just updated the code and now I think it should work just right.

Yours truly,

- Programmer for life

For Loop

```
void setup()
{
   for ( int i = 0; i < 16; i++ )
   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

Dear Compiler,

Wow, thanks for reviewing my code! That would have been a nasty bug! I just updated the code and now I think it should work just right.

Yours truly,

- Programmer for life

NOTE: Compiler's aren't this smart, at least not yet…

For Loop

```
void setup()
{
   for ( int i = 0; i < 16; i++ )
   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

So instead, we do Code Reviews!

Dear Compiler,

Wow, thanks for reviewing my code! That would have been a nasty bug! I just updated the code and now I think it should work just right.

Yours truly,

- Programmer for life

For Loop

```
void setup()
{
   for ( int i = 0; i < 16; i++ )
   {
      pinMode(ledPins[i],OUTPUT);
   }
}
```

So instead, we do Code Reviews!

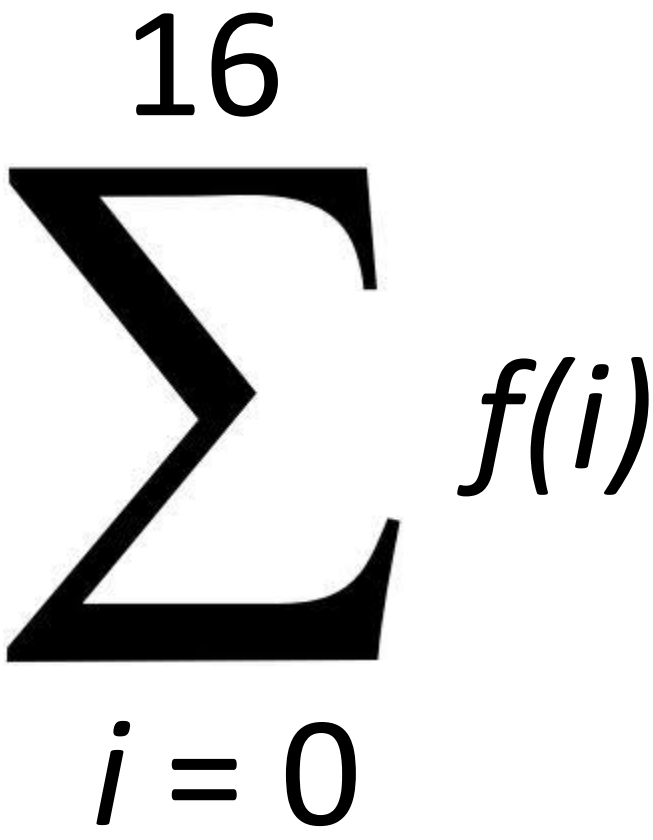
Dear Compiler,

Wow, thanks for reviewing my code! That would have been a nasty bug! I just updated the code and now I think it should work just right.

Yours truly,

- Programmer for life

For Loop


16
f(i)
i = 0

$$\sum_{i=0}^{16} f(i)$$

where f(i) is the stuff Inside the curly braces…

- But what if I don't know how many iterations there will be?

The while loop

```
void rampLightUntilButtonPushed()
{

    byte ledLevel = 0;









}
```

While Loop

```
void rampLightUntilButtonPushed()
{

   byte ledLevel = 0;
   while (            )
   {
      byte buttonState  = digitalRead(2);




      analogWrite(1,ledLevel++);
      delay(100);
   }
}
```

Dear Compiler,

Please execute the code within the upcoming curly braces for as long it takes,

While Loop

```
void rampLightUntilButtonPushed()
{
   bool done          ;
   byte ledLevel = 0;
   while ( done == false )
   {
      byte buttonState  = digitalRead(2);




      analogWrite(1,ledLevel++);
      delay(100);
   }
}
```

Dear Compiler,

Please execute the code within the upcoming curly braces for as long it takes, ***until the condition (the "done" variable),***

== *compares two variables, and evaluates to* ***true*** *if they are equal – otherwise it evalutes to* ***false****.*

While Loop

```
void rampLightUntilButtonPushed()
{
   bool done          ;
   byte ledLevel = 0;
   while ( ! done )
   {
      byte buttonState  = digitalRead(2);



      analogWrite(1,ledLevel++);
      delay(100);
   }
}
```

Dear Compiler,

Please execute the code within the upcoming curly braces for as long it takes, ***until the condition (the "done" variable),***

! means logical NOT

While Loop

```
void rampLightUntilButtonPushed()
{
   bool done = false;
   byte ledLevel = 0;
   while ( ! done )
   {
      byte buttonState  = digitalRead(2);




      analogWrite(1,ledLevel++);
      delay(100);
   }
}
```

Dear Compiler,

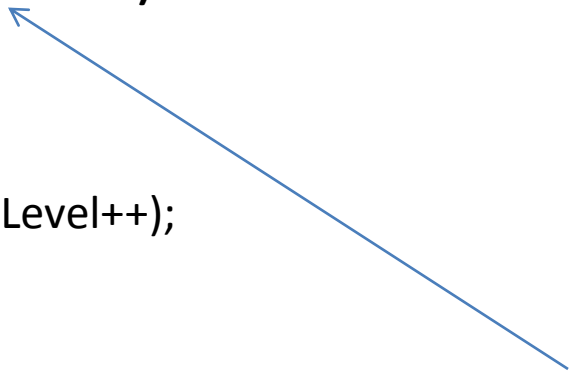
Please execute the code within the upcoming curly braces for as long it takes, until the condition (the "done" variable), ***which is initially false****,*

While Loop

```
void rampLightUntilButtonPushed()
{
   bool done = false;
   byte ledLevel = 0;
   while ( ! done )
   {
      byte buttonState  = digitalRead(2);
      if ( buttonState == HIGH )
      {
         done = true;
      }
      analogWrite(1,ledLevel++);
      delay(100);
   }
}
```

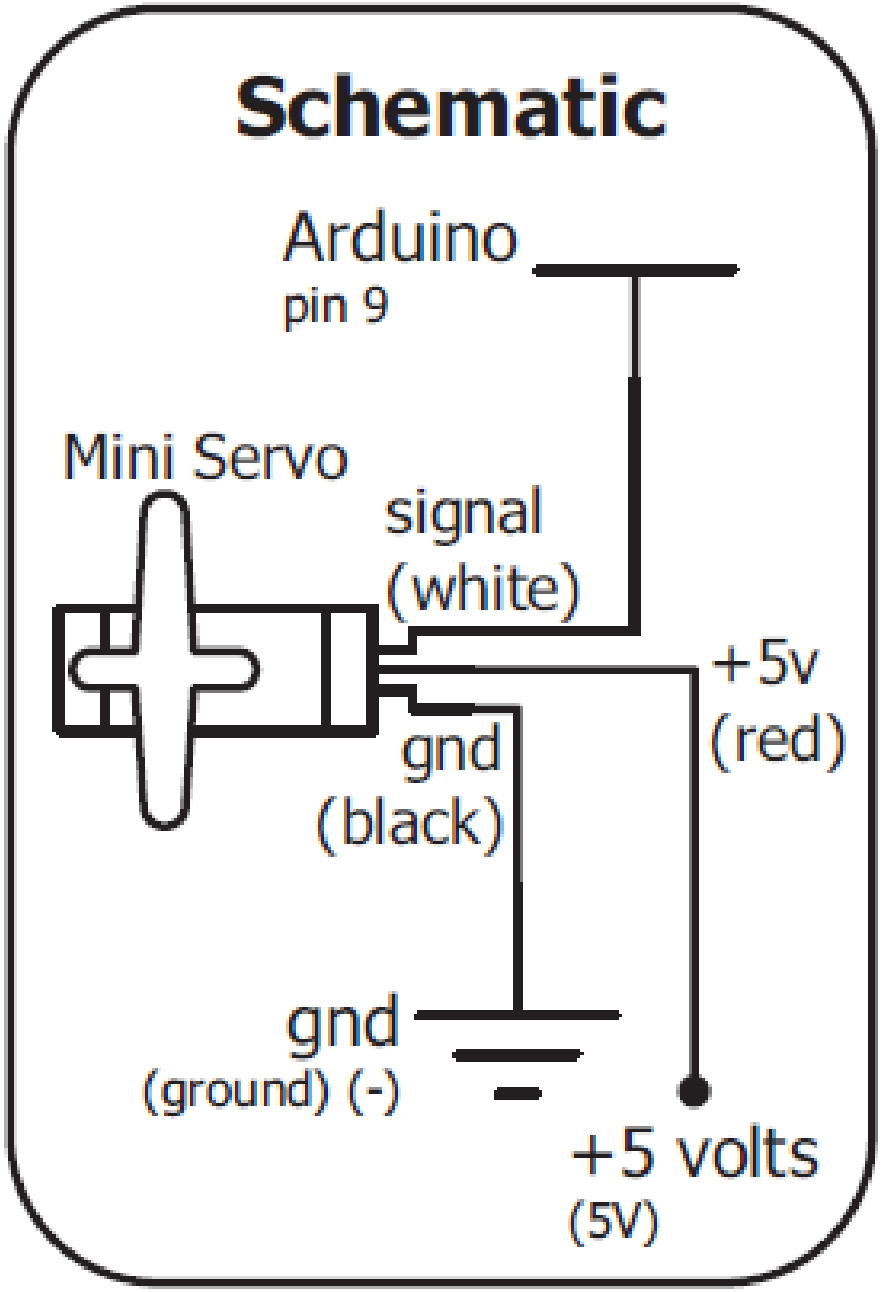
Dear Compiler,

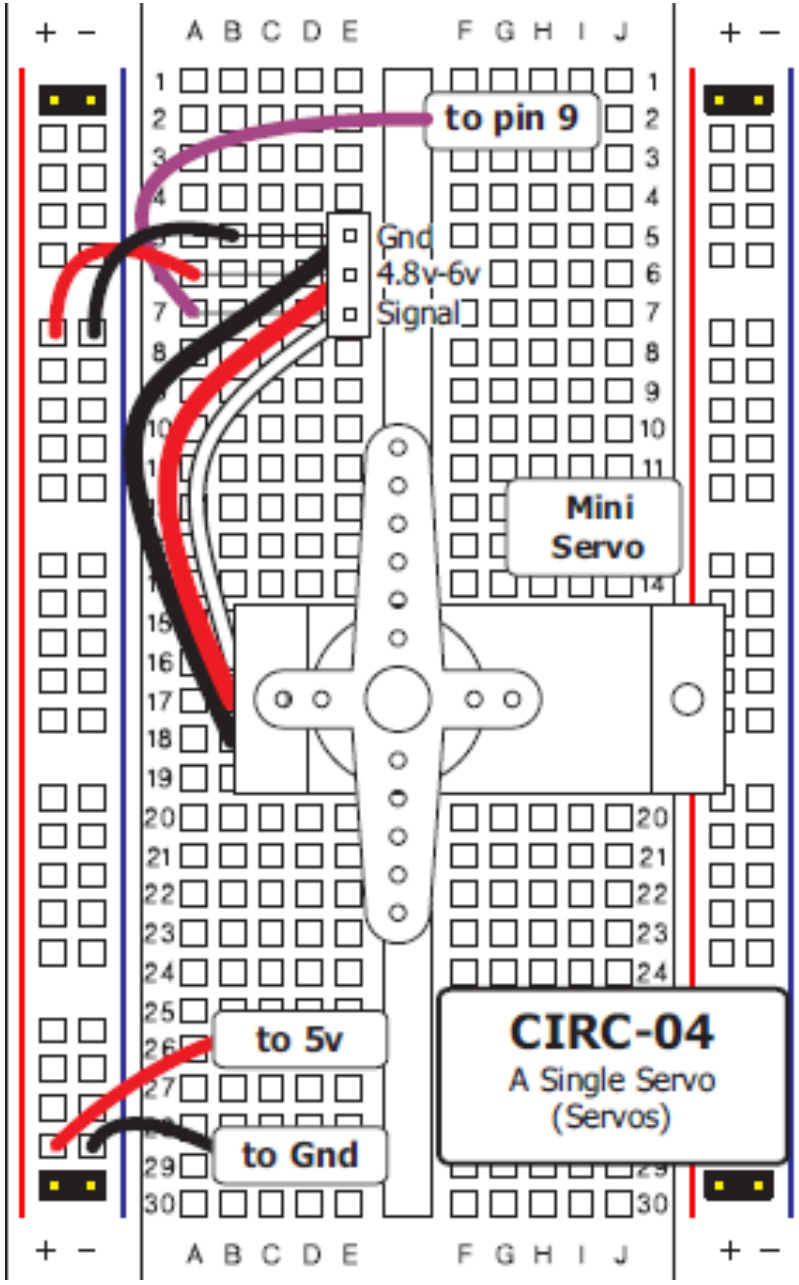
Please execute the code within the upcoming curly braces for as long it takes, until the condition (the "done" variable), which is initially false, ***becomes true****.*



== *compares two variables, and evaluates to* ***true*** *if they are equal – otherwise it evalutes to* ***false****.*

While Loop


Schematic
Arduino
pin 9
Mini Servo
signal
(white)
+5v
(red)
gnd
(black)
gnd
(ground) (-)
+5 volts
(5V)


to pin 9
Gnd
4.8v-6v
Signal
Mini
Servo
to 5v
to Gnd
CIRC-04
A Single Servo
(Servos)

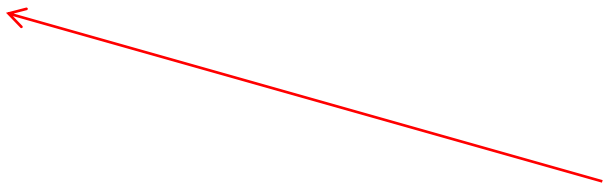
Includes another file which “declares” a class called Servo.

```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
```

Includes another file which “declares” a class called Servo. We get to use other code that somebody else wrote and tested for us!

```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
```

```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
```



Creates a Servo object so we can use it….

Servo is the type

myservo is the name

; tells the compiler the statement is complete.

```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
int pos = 0;    // variable to store the servo position
```

```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
int pos = 0;    // variable to store the servo position

void setup() {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}
```

```
#include <Servo.h>
Servo myservo;  // create servo object to control a servo
int pos = 0;    // variable to store the servo position

void setup() {
  myservo.attach(9);  // attaches the servo on pin 9 to the servo object
}

void loop() {
  for(pos = 0; pos < 180; pos += 1)  // goes from 0 degrees to 180 degrees
  {                                  // in steps of 1 degree
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
  for(pos = 180; pos>=1; pos-=1)     // goes from 180 degrees to 0 degrees
  {
    myservo.write(pos);              // tell servo to go to position in variable 'pos'
    delay(15);                       // waits 15ms for the servo to reach the position
  }
}
```

KEY FACT 1: Each Arduino pin can source 40mA of current

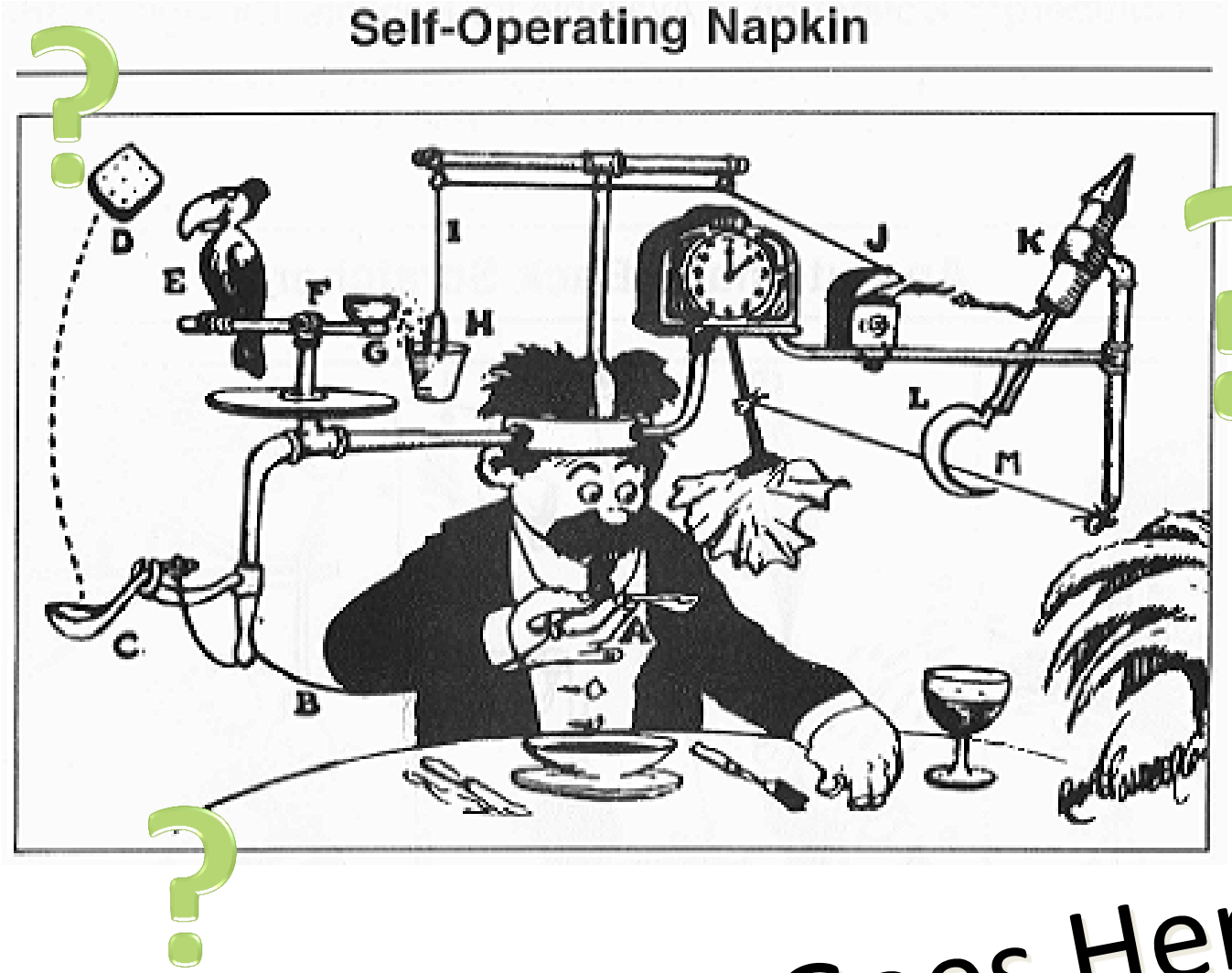
KEY FACT 2: Some actuators in our kit require more than 40mA

- *Huh?*

- So do you mean we can't use some of this cool stuff in our kit?

How can Arduino control more current than 40mA?

Why should I care?



Your Project Goes Here

How can Kate control the water pump motors w/Arduino?

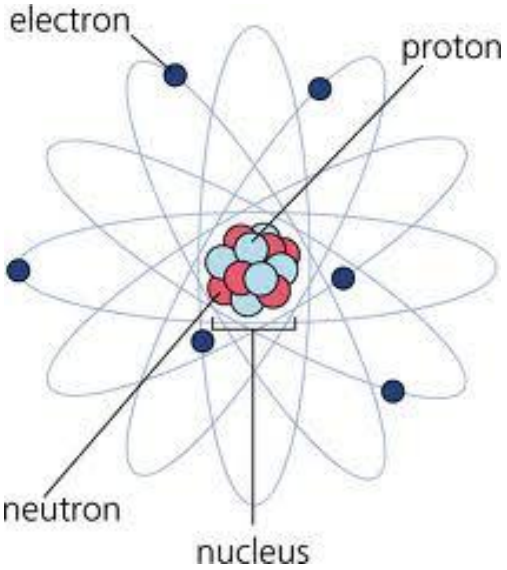
How can Charlie control the lasers in the laser harp?

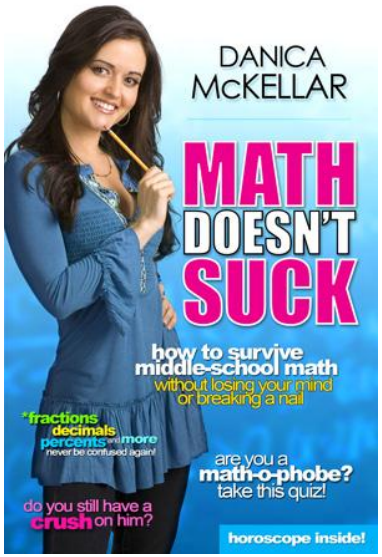
How can Jim control the valves at his brewery?

How can Ava control the motors to turn her automata?

How can Lexa control the motors to open the door lock?

What's a milli-Amp (mA)?





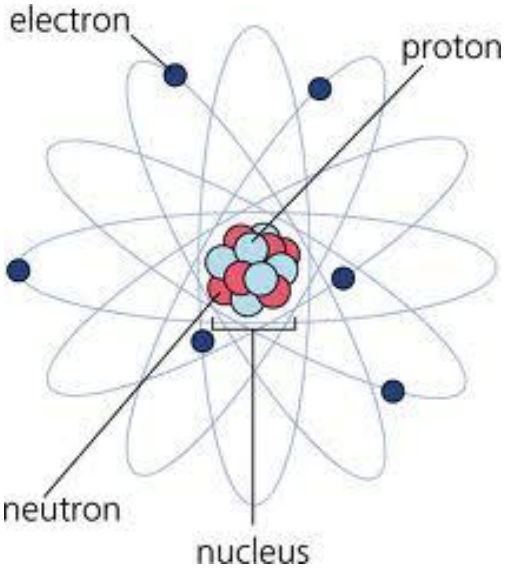
1milli-Amp (mA)

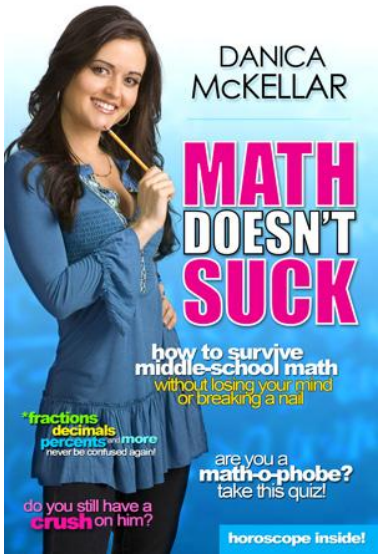
=

1 thousandth of an Amp

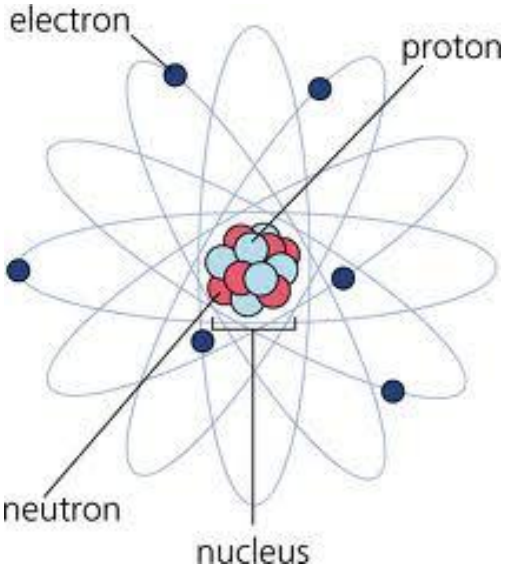
=

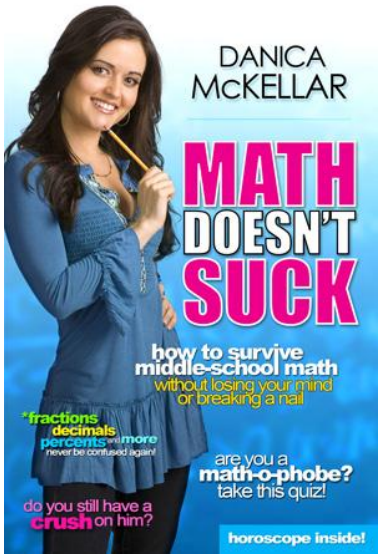
(1 Amp / 1000)





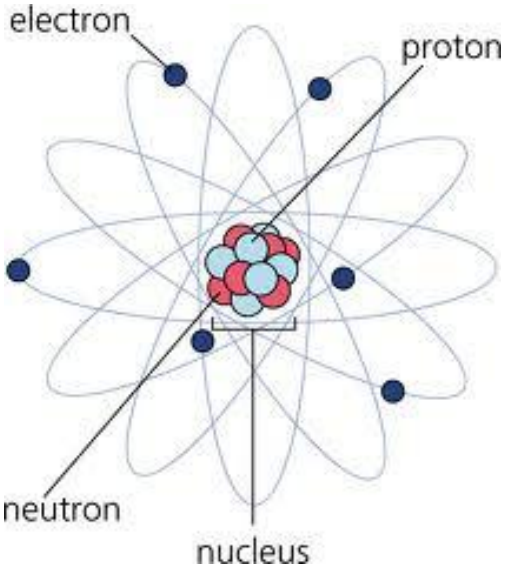
What's an Amp?

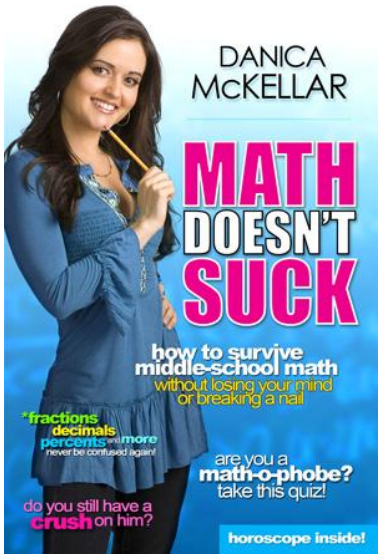

electron
proton
neutron
nucleus


DANICA
McKELLAR
MATH
DOESN'T
SUCK
how to survive
middle-school math
without losing your mind
or breaking a nail
*fractions
decimals
percents and more
never be confused again!
are you a
math-o-phobe?
take this quiz!
do you still have a
crush on him?
horoscope inside!

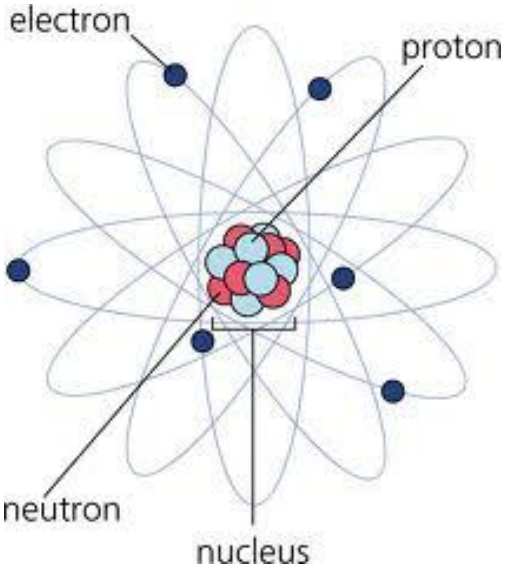
Amp = rate of electron flow

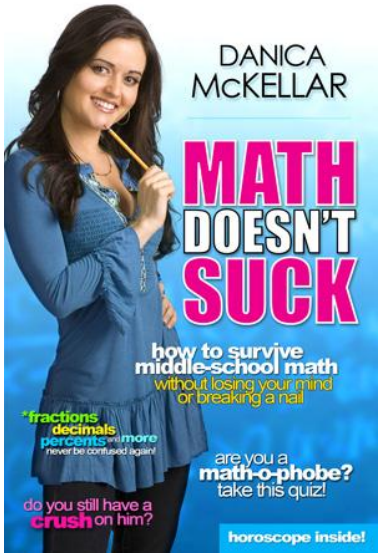
1 Amp = 6.2×10^{18} e-/second





So, in 1 second how many electrons are flowing through our 20mA LEDs?

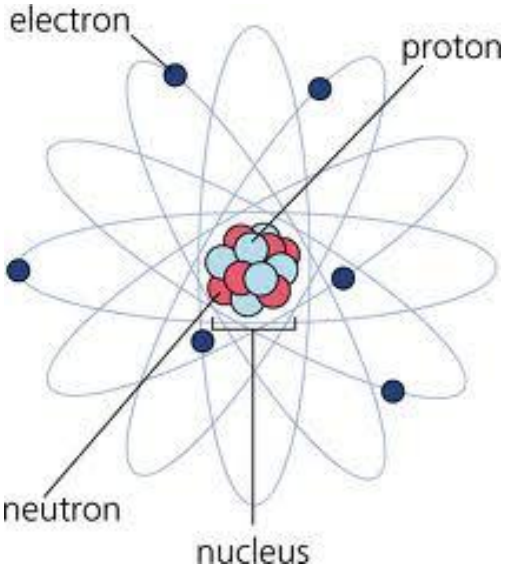


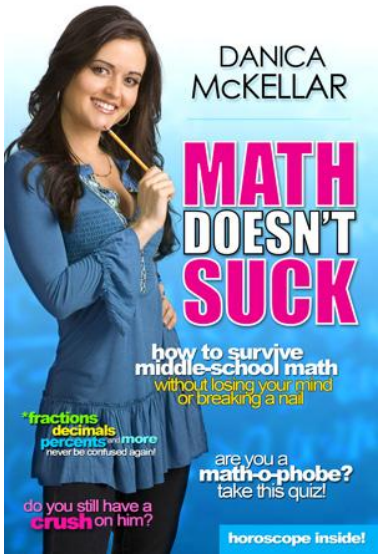


1 Amp = 6.2×10^{18} e-/second

20mA = 1Amp / 500

20mA = 1.24×10^{16} e-/second

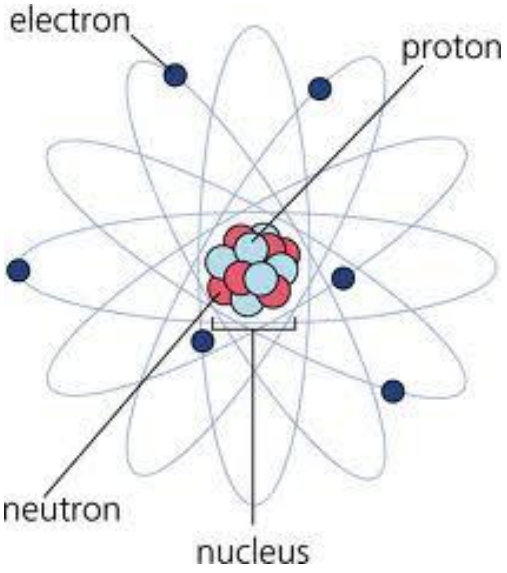


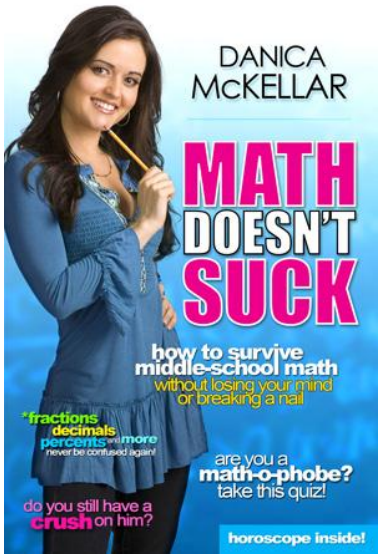


14 zeroes!

12,400,000,000,000,000

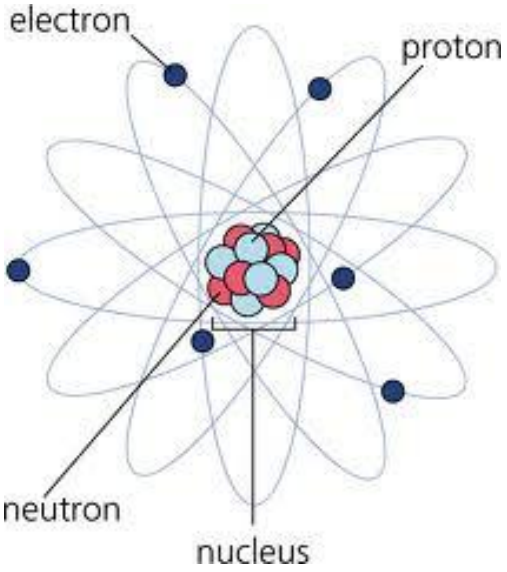
e-/second

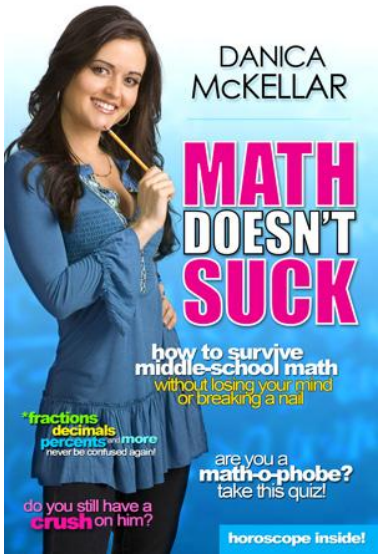




> 12 Quadrillion

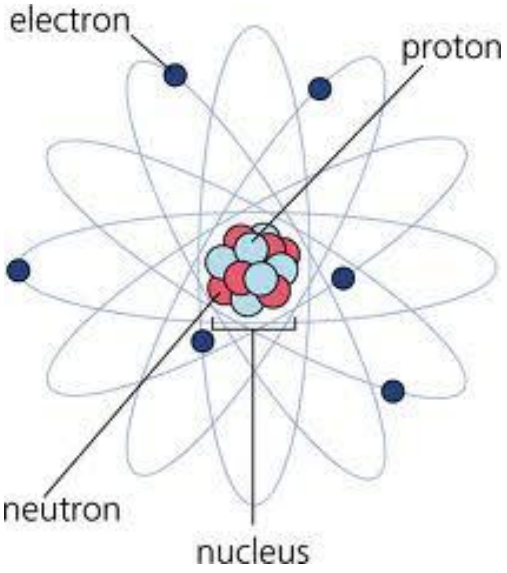
e-/second

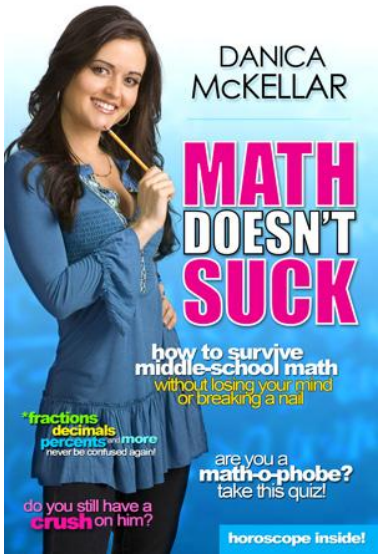




> 12 Million Billion

e-/second



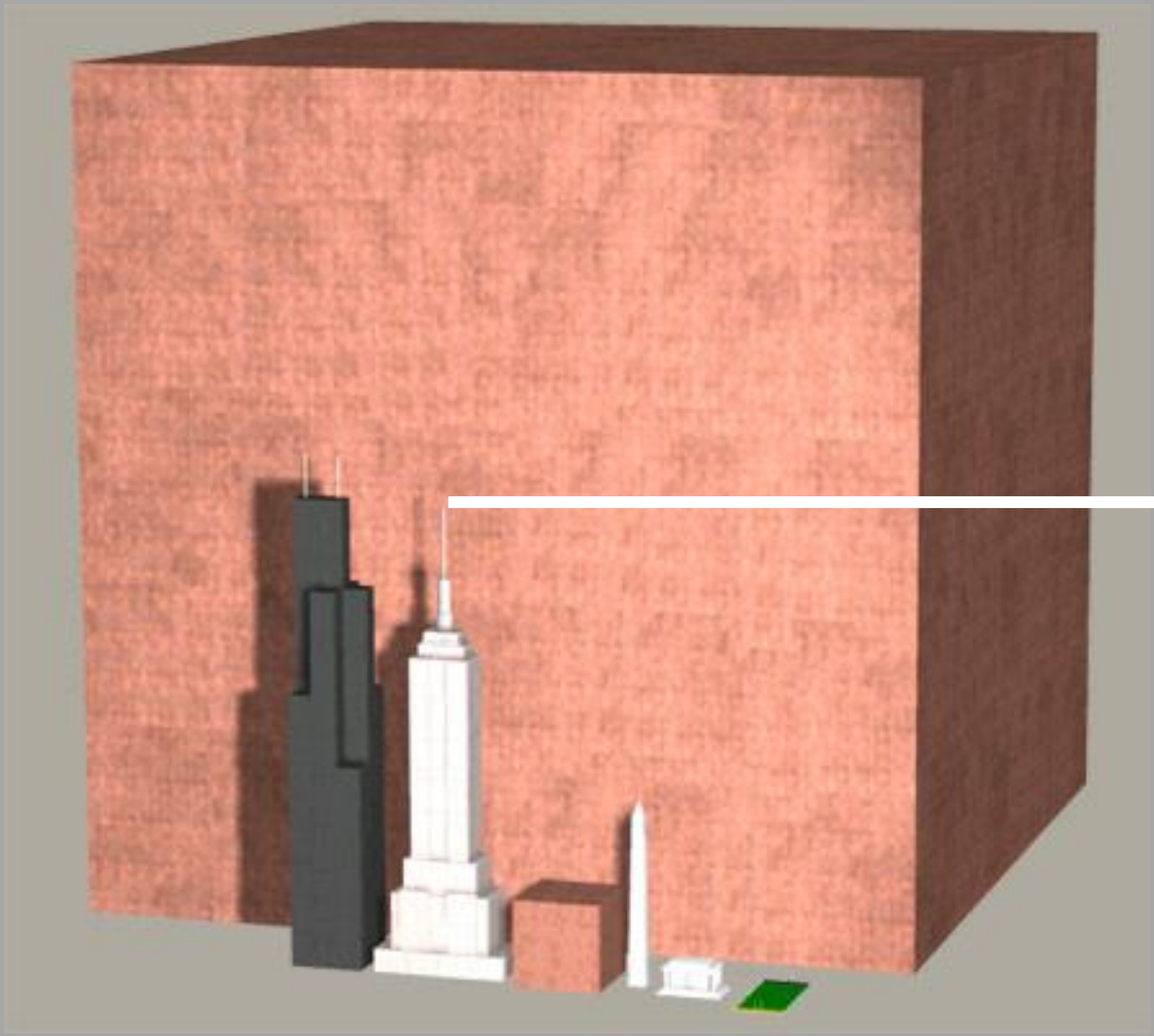


Analogy /əˈnaləjē/: *n.*

[Origin: Middle English analogie, from Old French, from Latin analogia, from Greek analogi, from analogos, *proportionate*;]

A comparison between two things, typically on the basis of their structure and for the purpose of explanation or clarification.


BILLIONS!
and
BILLIONS!



Empire State Building
102 Stories

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

ANALOGY ALERT

ANALOGY ALERT

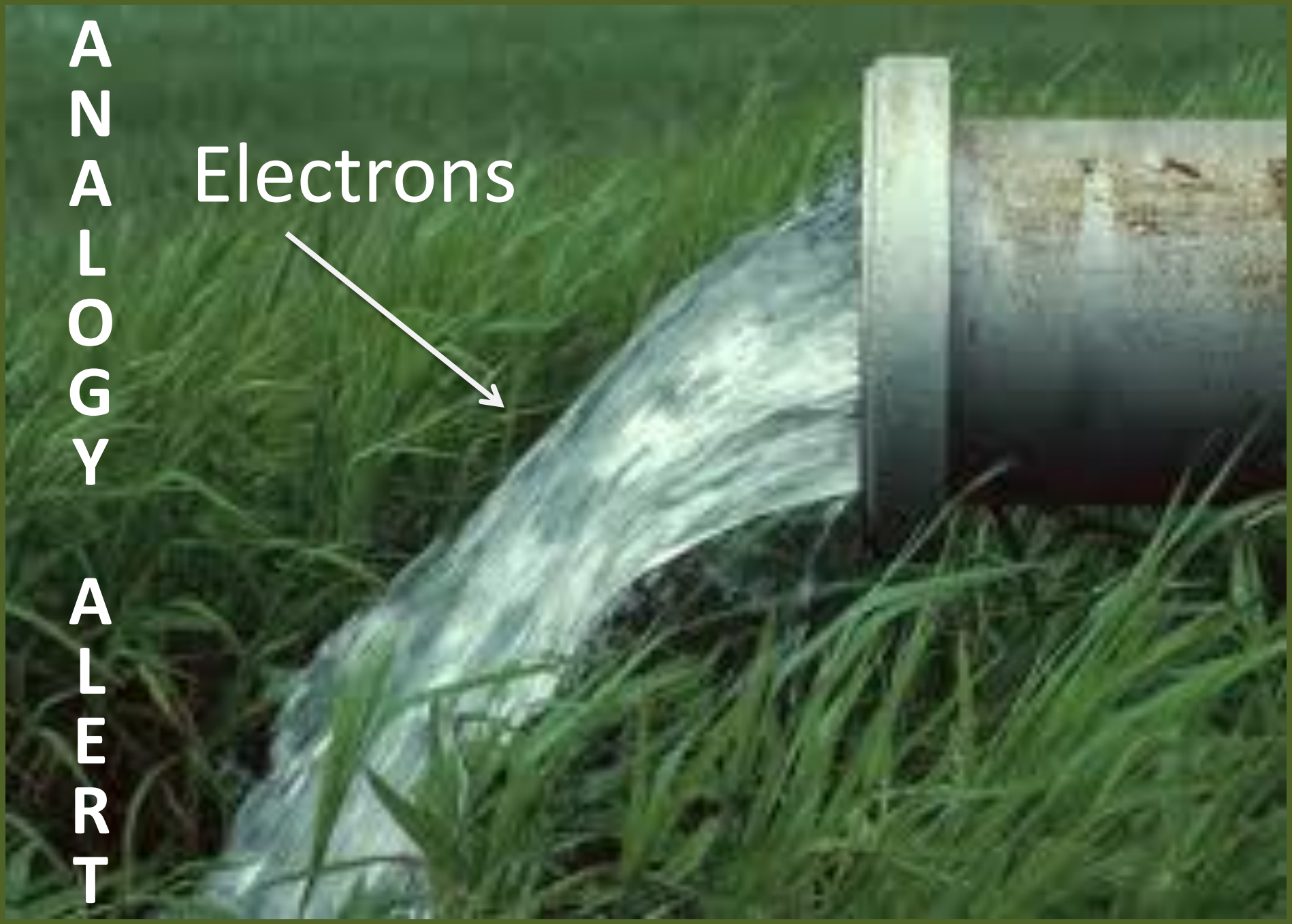
Conductor (Wire)

ANALOGY ALERT

Conductor (Wire)

ANALOGY ALERT

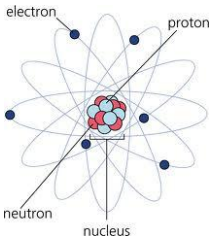
Conductor (Wire)


ANALOGY ALERT
Electrons

ANALOGY ALERT

Flow Rate (Gallons/Hour)

ANALOGY ALERT



Current (Amps)

Current Rating

ANALOGY ALERT

ANALOGY ALERT

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)

ATmega328 Datasheet

29. Electrical Characteristics

29.1 Absolute Maximum Ratings*

Operating Temperature	-55°C to +125°C
Storage Temperature	-65°C to +150°C
Voltage on any Pin except $\overline{\text{RESET}}$ with respect to Ground	-0.5V to V_{CC}+0.5V
Voltage on $\overline{\text{RESET}}$ with respect to Ground	-0.5V to +13.0V
Maximum Operating Voltage	6.0V
DC Current per I/O Pin	40.0mA
DC Current V_{CC} and GND Pins	200.0mA

*NOTICE: Stresses beyond those listed under "Absolute Maximum Ratings" may cause permanent damage to the device. This is a stress rating only and functional operation of the device at these or other conditions beyond those indicated in the operational sections of this specification is not implied. Exposure to absolute maximum rating conditions for extended periods may affect device reliability.

“The absolute max should be avoided - it is dangerous to approach it. “


EARLY GRAVE

Current Rating: 40mA

Current Rating: 40mA

Current Rating: 40mA

Current Rating: 40mA

Current Rating: 40mA

Current Rating: 40mA

Current Rating: 40mA

Current Rating: 40mA

Current Rating: 40mA

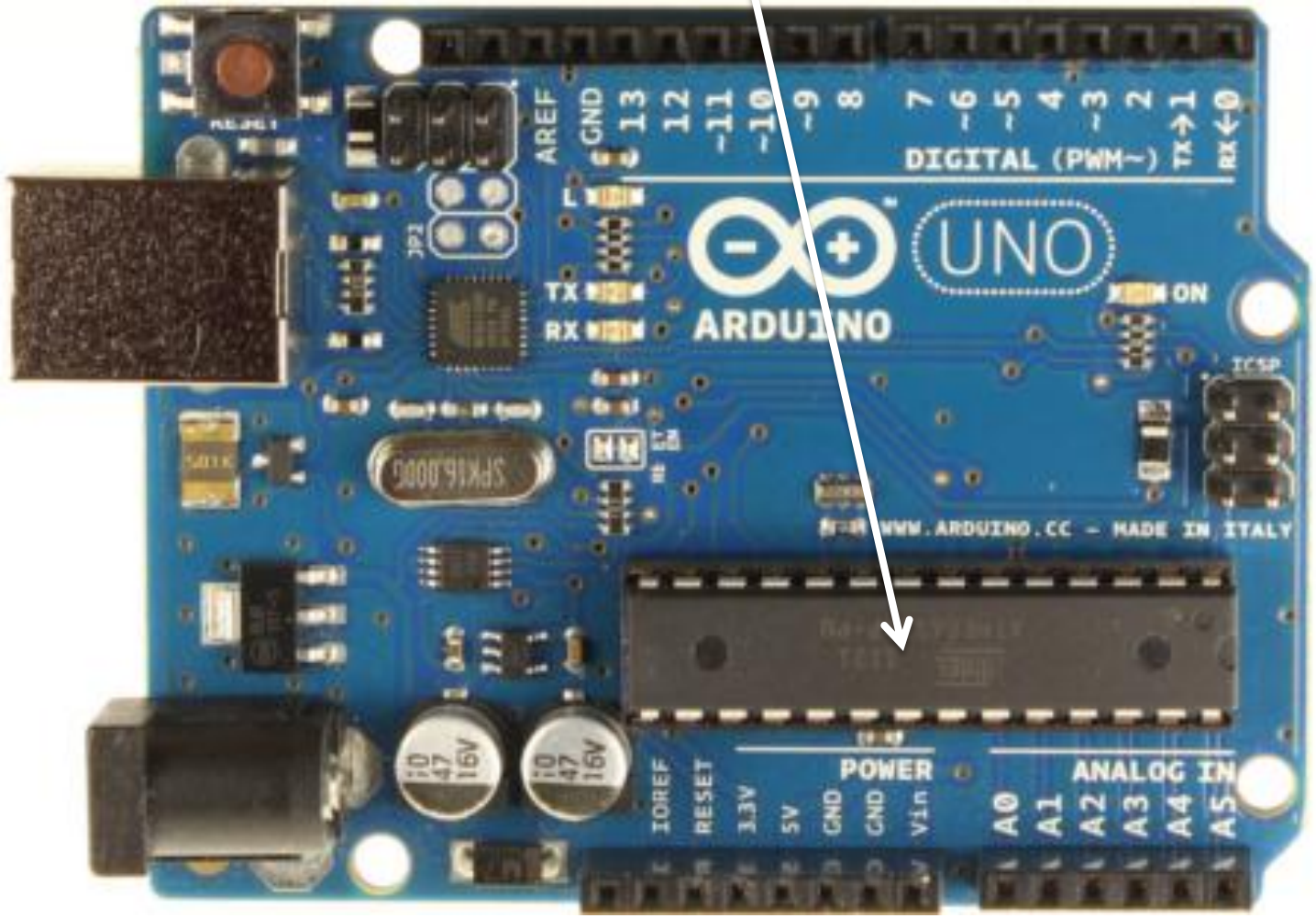
Current Rating: 40mA

Current Rating: 40mA

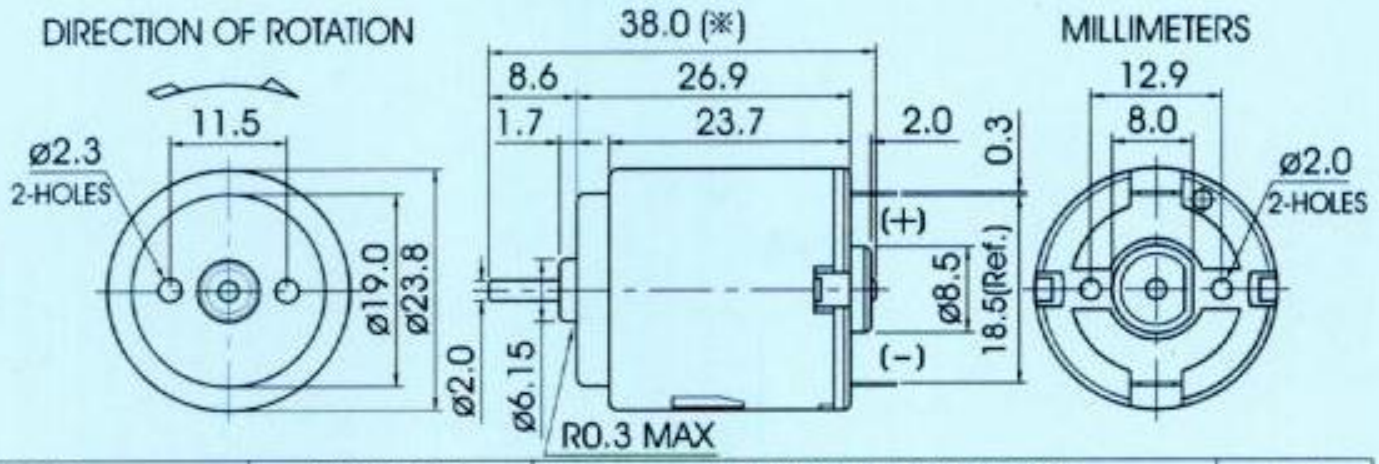
Current Rating: 40mA

Current Rating: 200mA

Current Rating: 200mA



Motor Datasheet



MODEL	VOLTAGE		NO LOAD		AT MAXIMUM EFFICIENCY					STALL TORQUE
	OPERATING RANGE	NOMINAL	SPEED	CURRENT	SPEED	CURRENT	TORQUE	OUTPUT	EFF	
			rpm	A	rpm	A	g.cm	W	%	g.cm
201-A	1.5~4.5	3.0V CONSTANT	12800	0.32	10000	1.28	20	2.05	53.4	80
201-B	1.5~4.5	3.0V CONSTANT	11800	0.30	9350	1.04	17.5	1.67	53	73
201-G	1.5~6.0	3.0V CONSTANT	6750	0.16	5050	0.46	12	0.62	45	42

Circuit Happiness

Dead

Stressed

Healthy

Underpowered

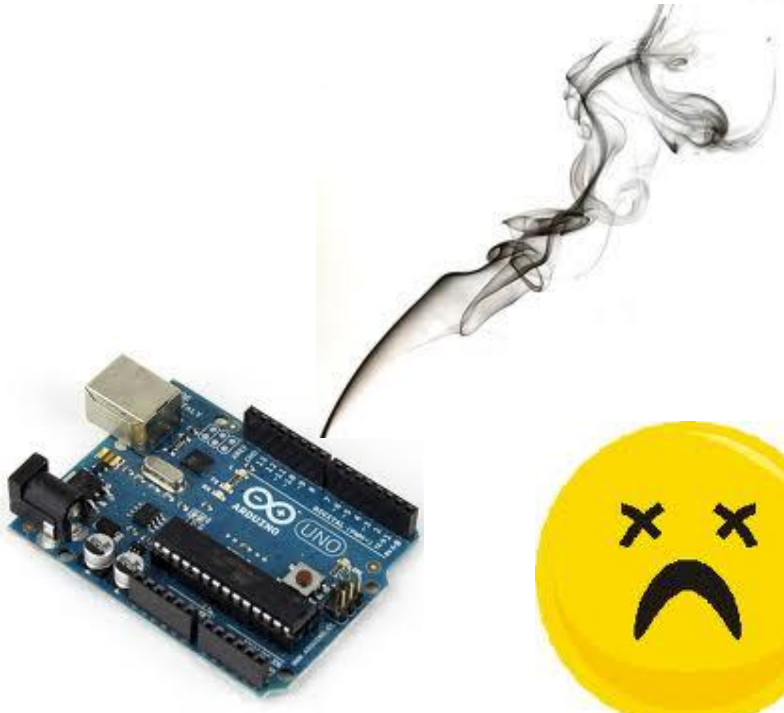
Asleep

ANALOGY ALERT

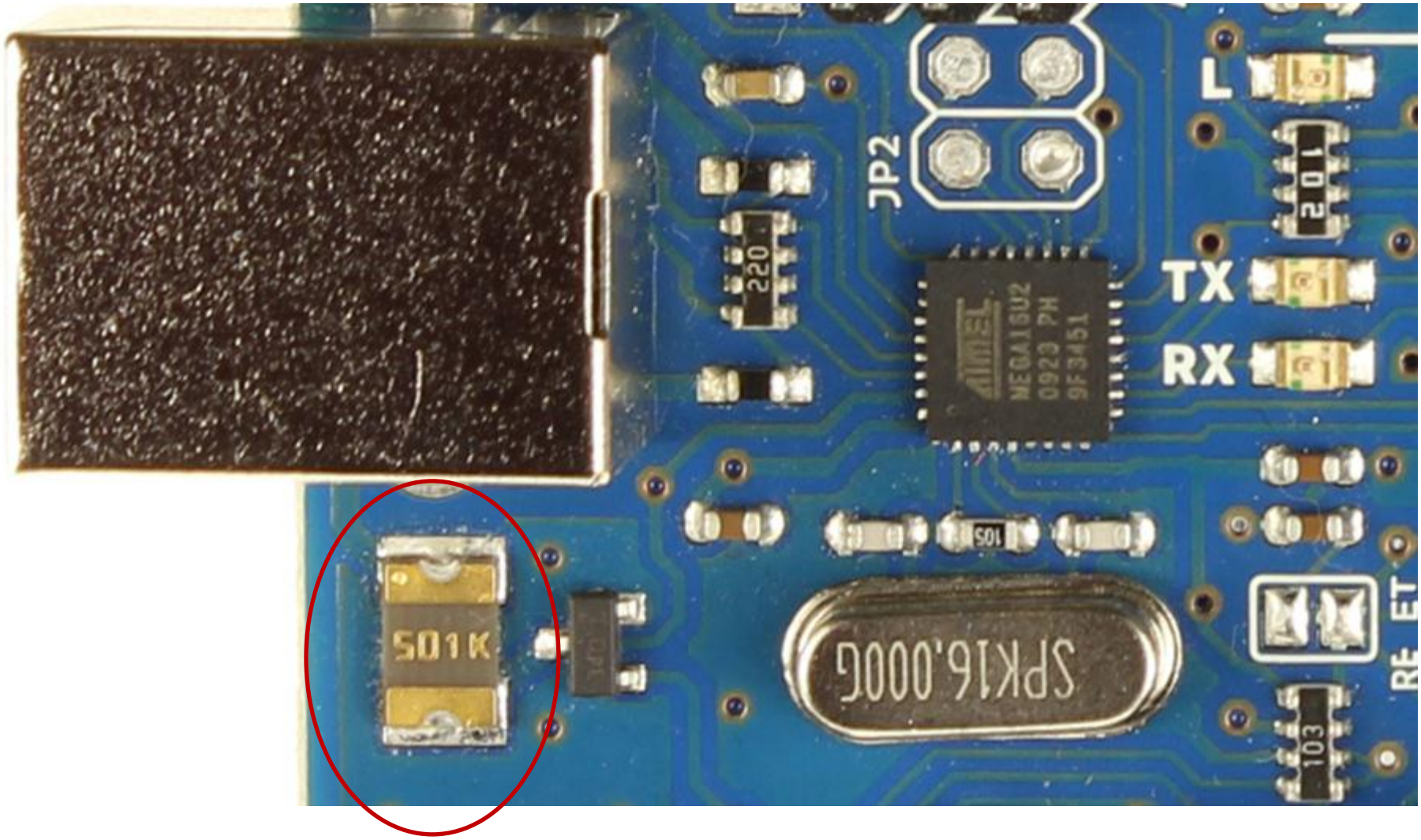
What happens if there’s not enough current?

What happens if the current rating is exceeded?


SMOKE ALARM


ARDUINO
UNO


119
119



Polyfuse (500mA)

Ways to Kill an Arduino

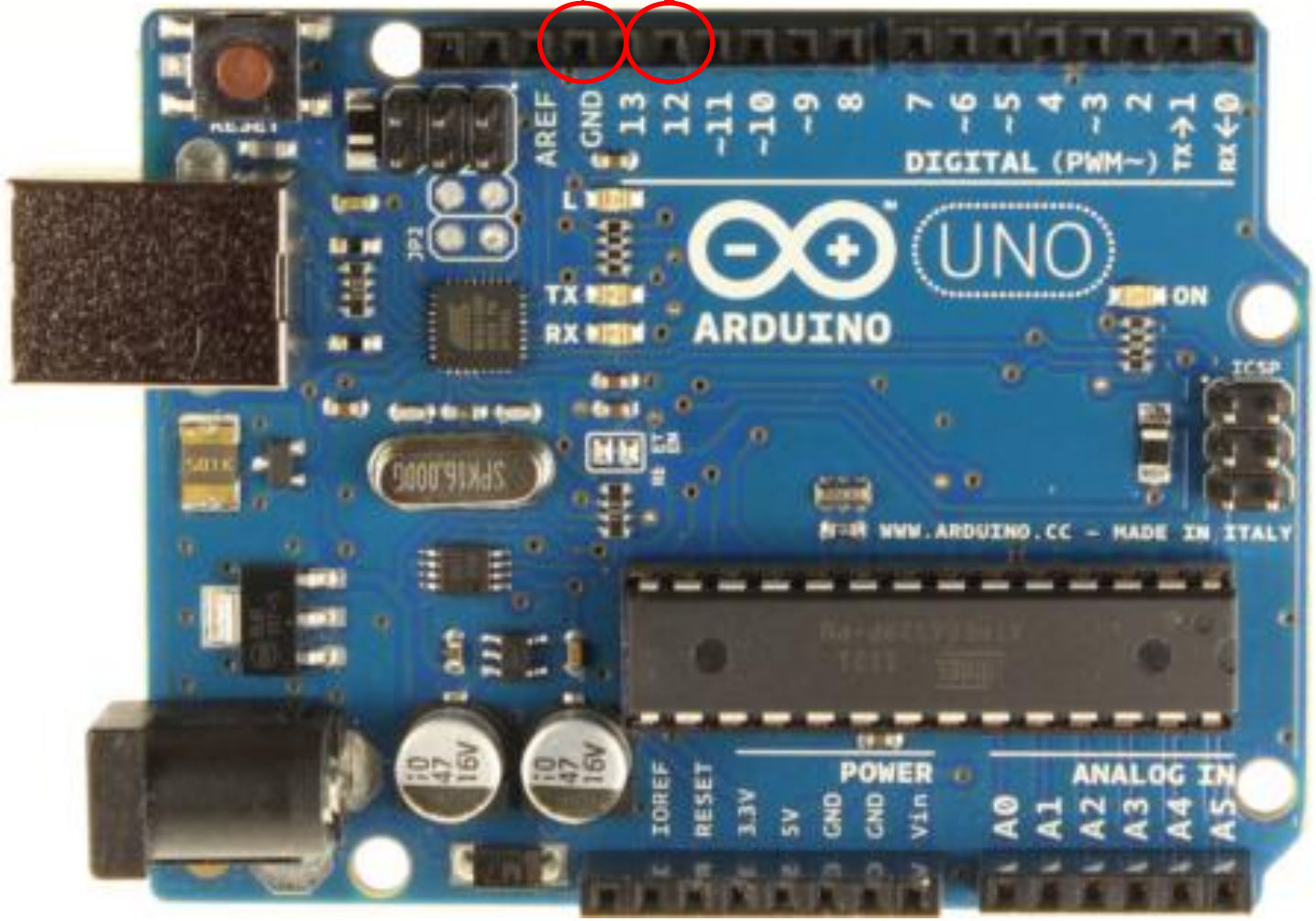
Easily Possible

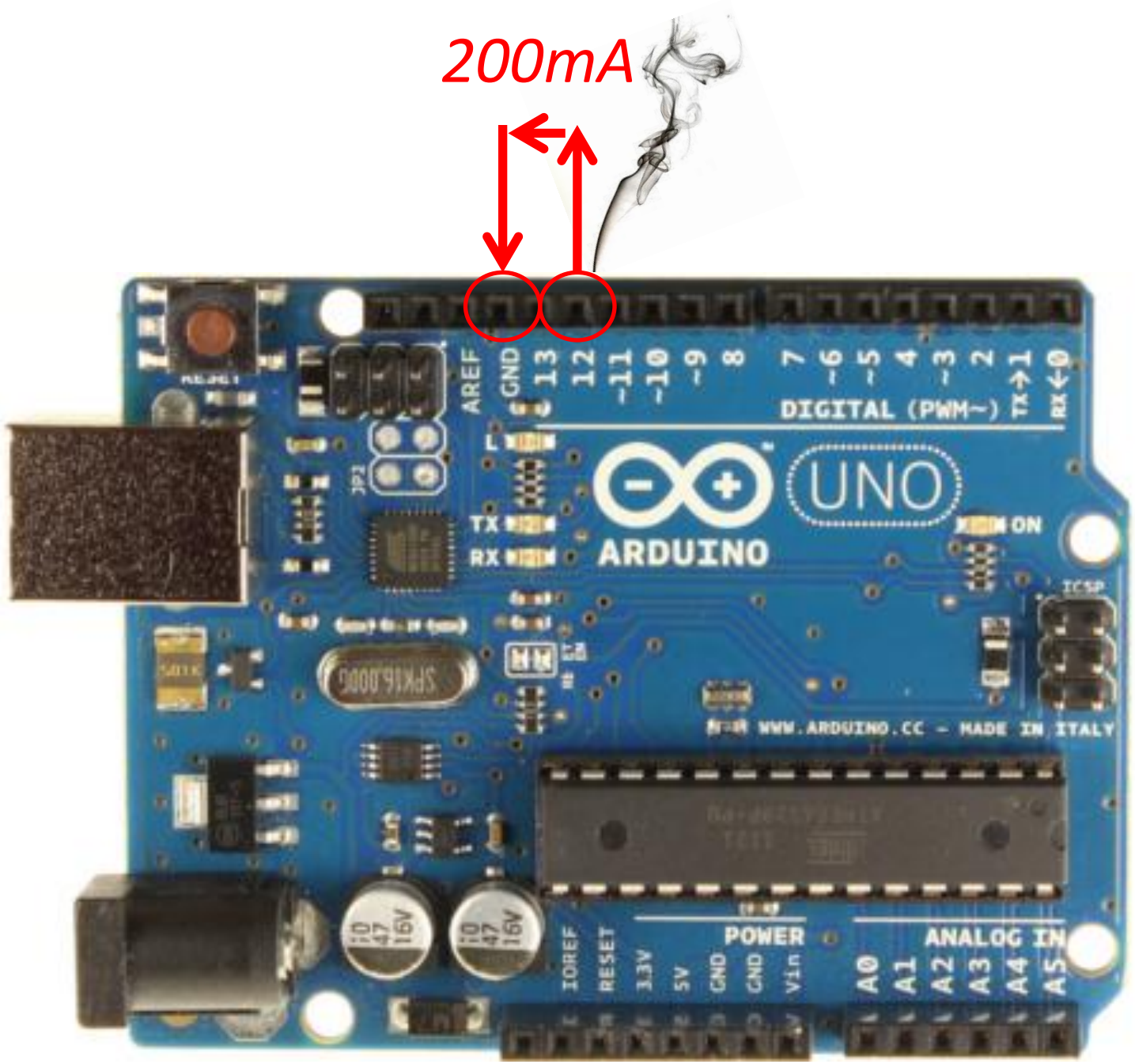
Shorting I/O Pins to Ground

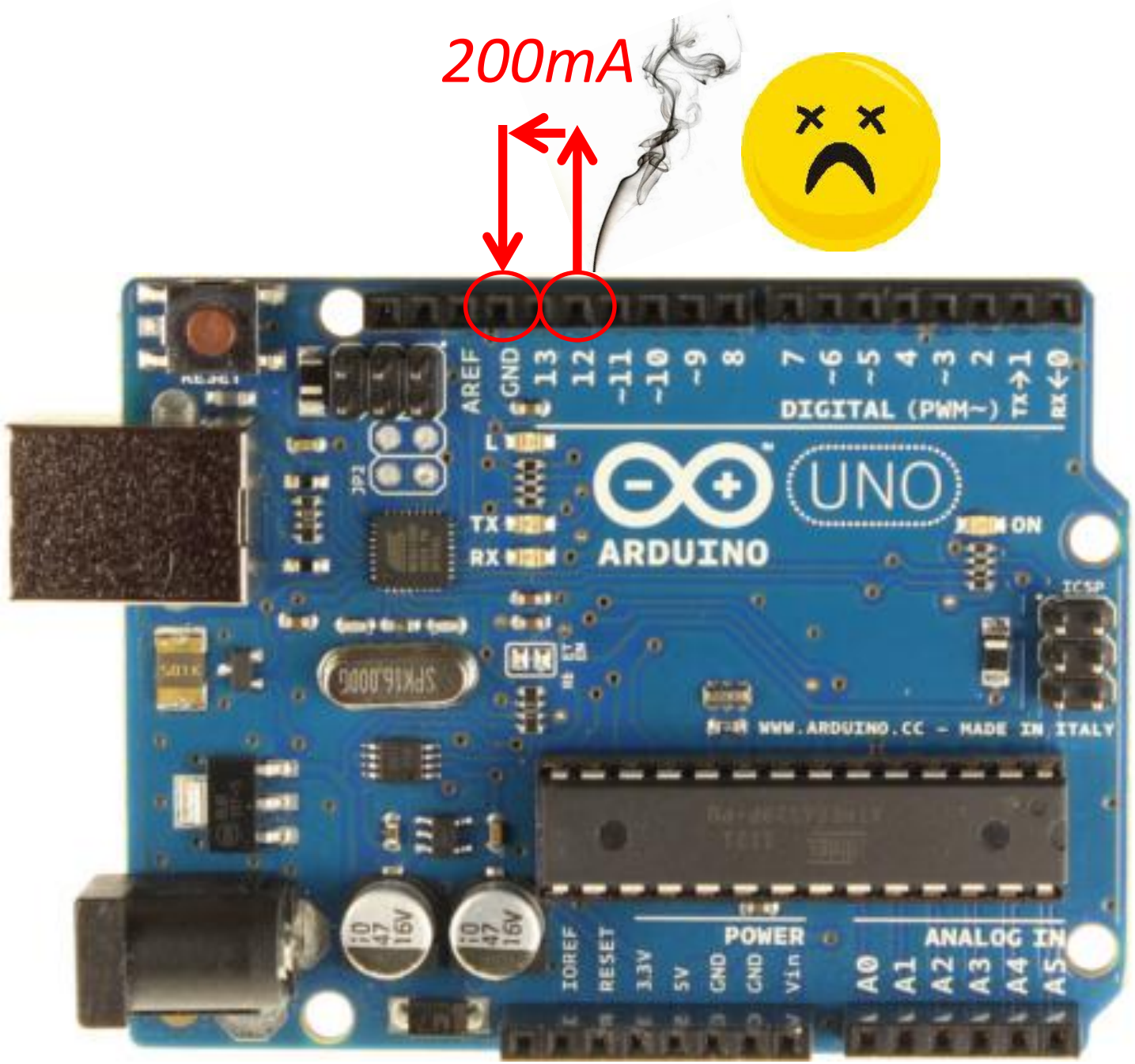
200mA

pinMode(12,OUTPUT);

digitalWrite(12,HIGH);




200mA
AREF
GND
13
12
~11
~10
~9
8
7
6
5
4
3
2
TX→1
RX←0
DIGITAL (PWM~)
UNO
ARDUINO
ON
TX
RX
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5


200mA
AREF
GND
13
12
~11
~10
~9
8
7
6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
UNO
ARDUINO
ON
TX
RX
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

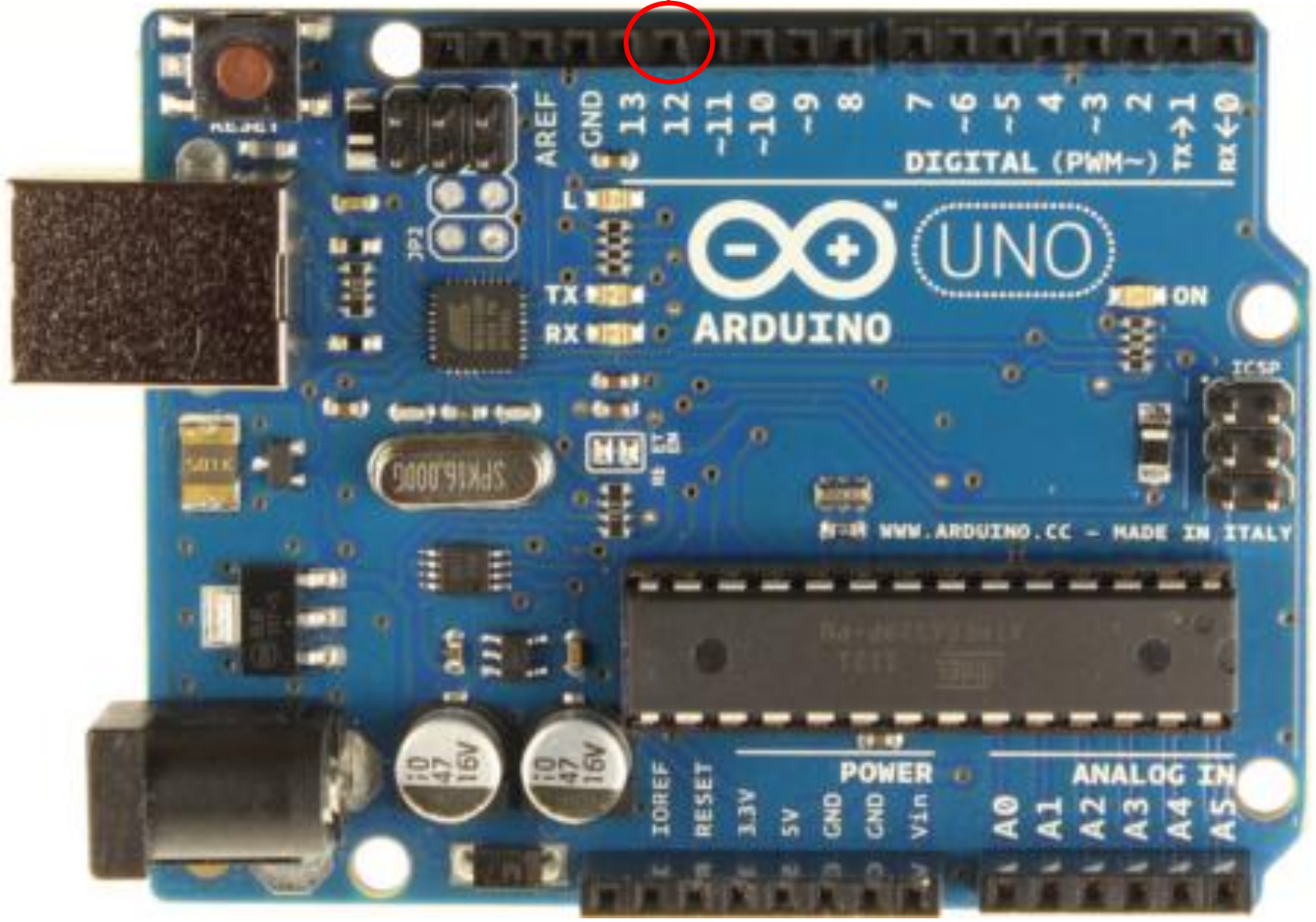
Ways to Kill an Arduino

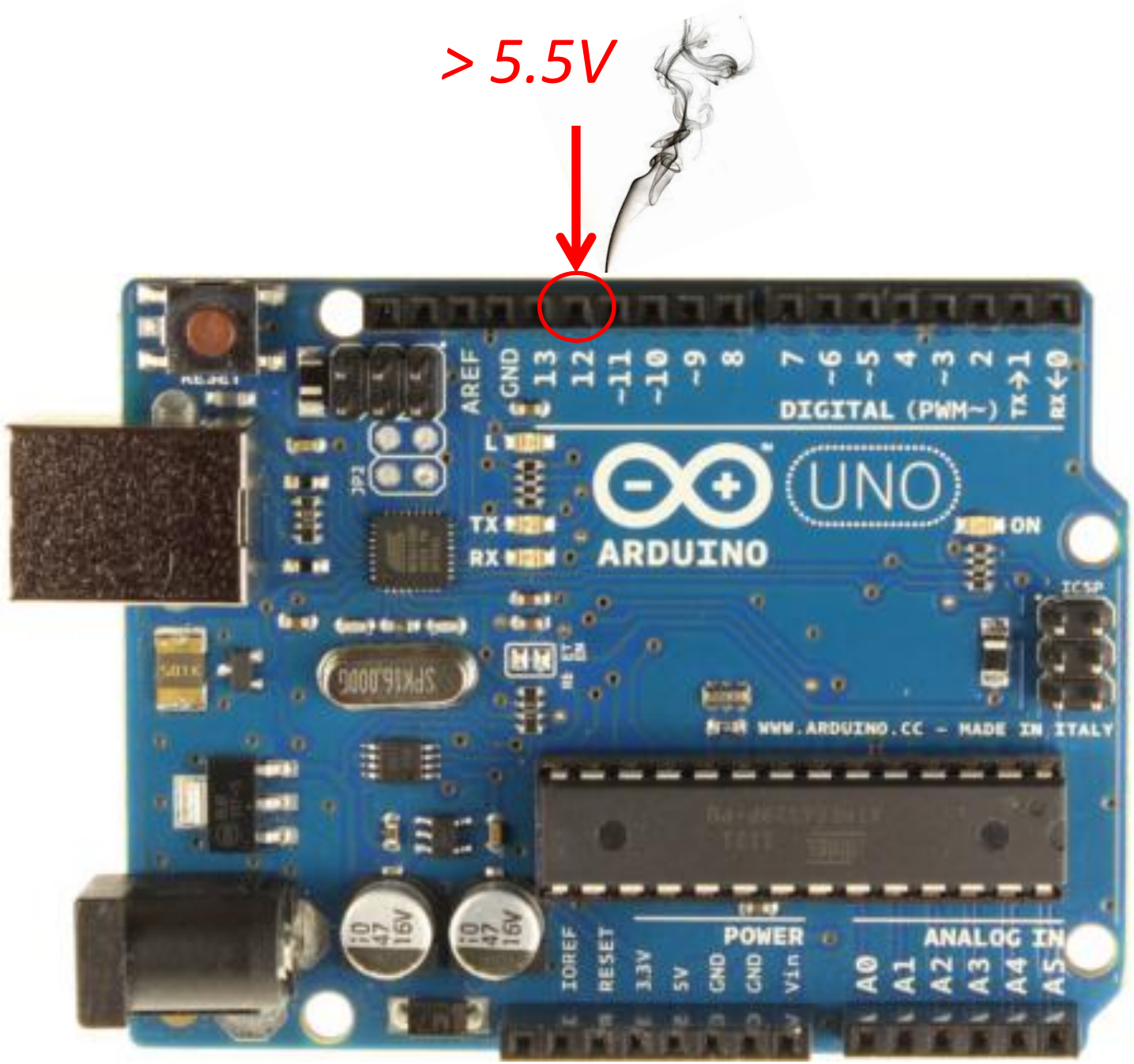
Easily Possible

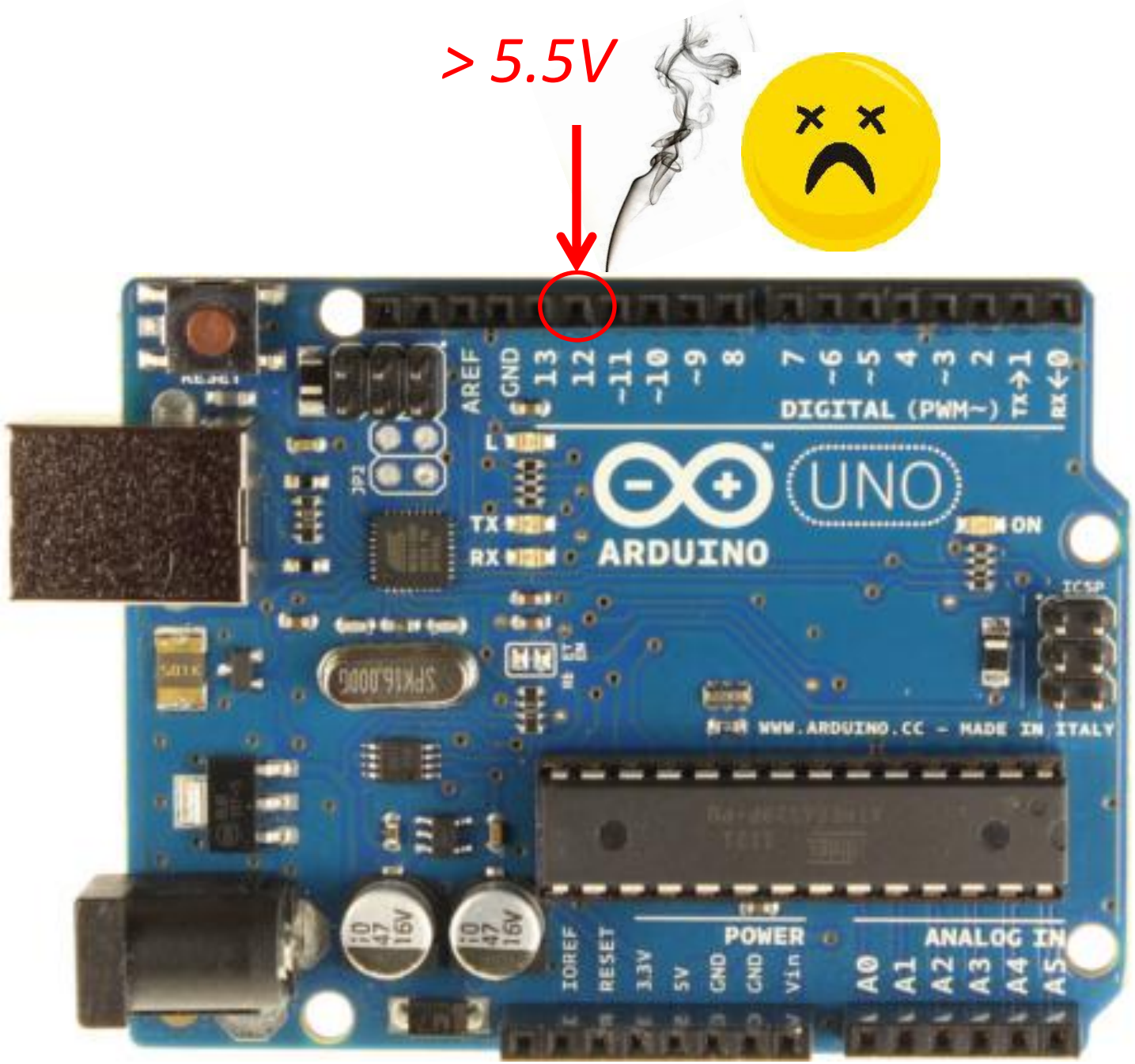
Shorting I/O Pins to Ground
Apply Overvoltage to I/O Pins

> 5.5V

pinMode(12,INPUT);




> 5.5V
DIGITAL (PWM~)
UNO
ARDUINO
ON
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN


> 5.5V
DIGITAL (PWM~)
UNO
ARDUINO
ON
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN

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

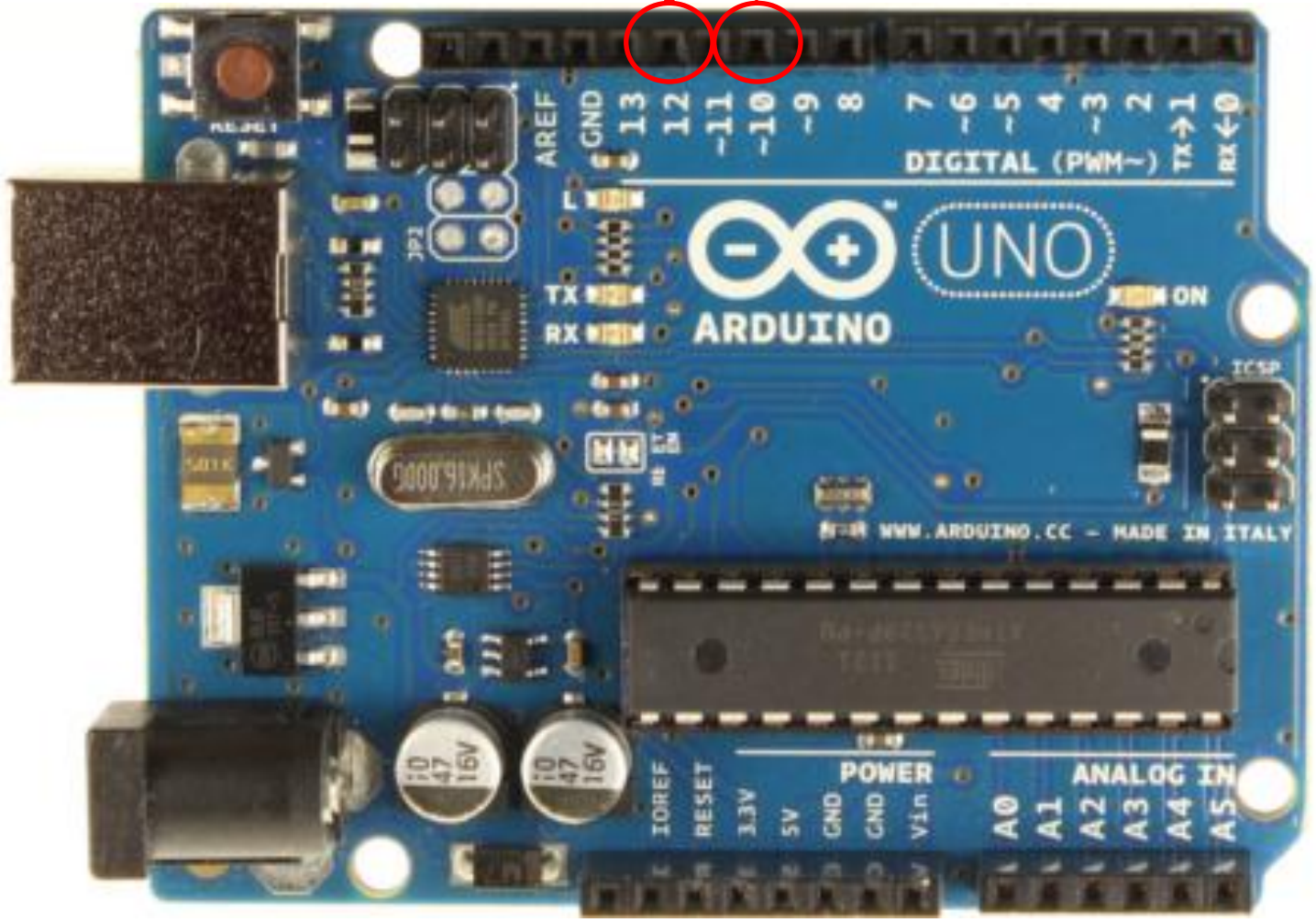
200mA

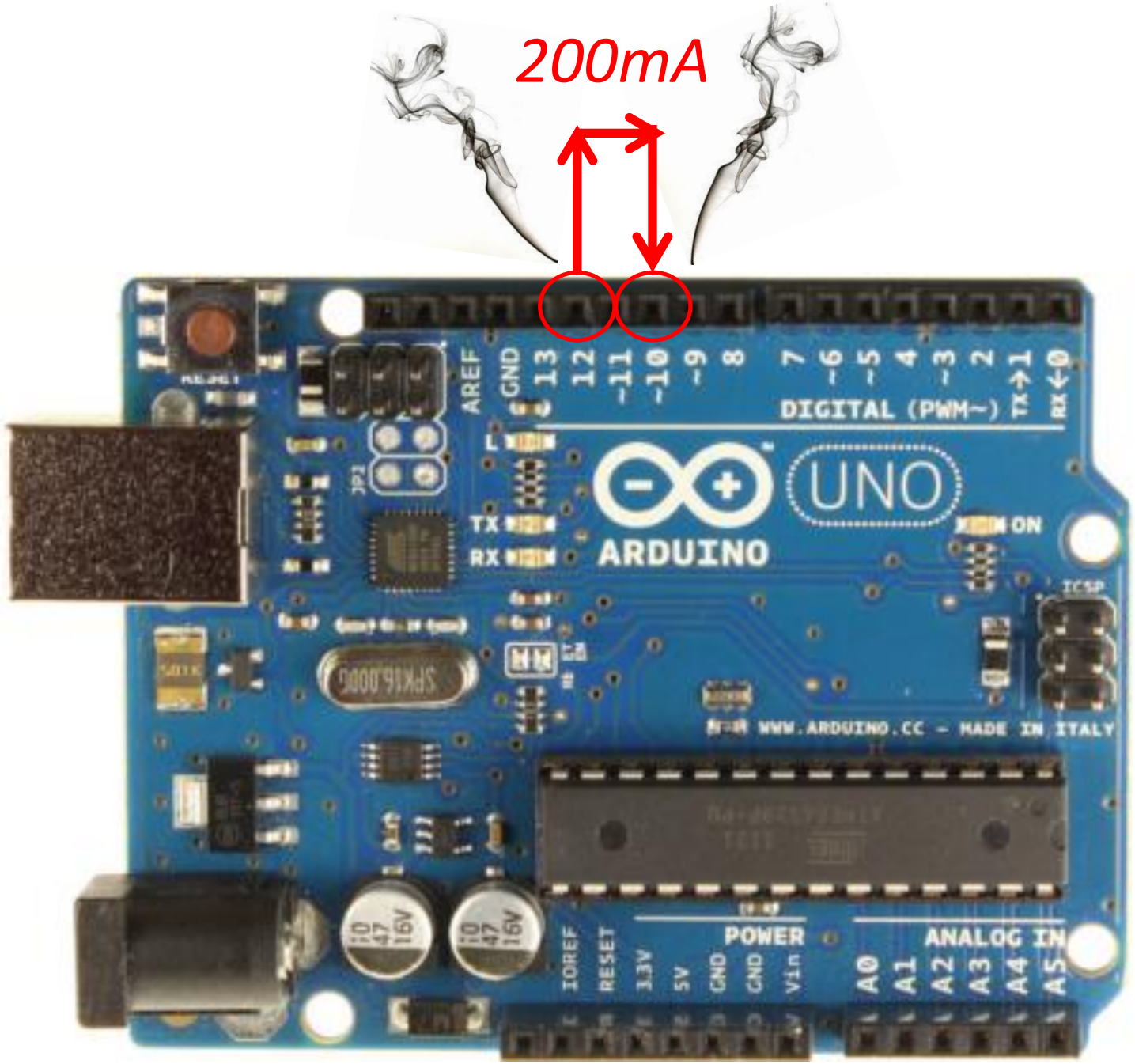
pinMode(12,OUTPUT);

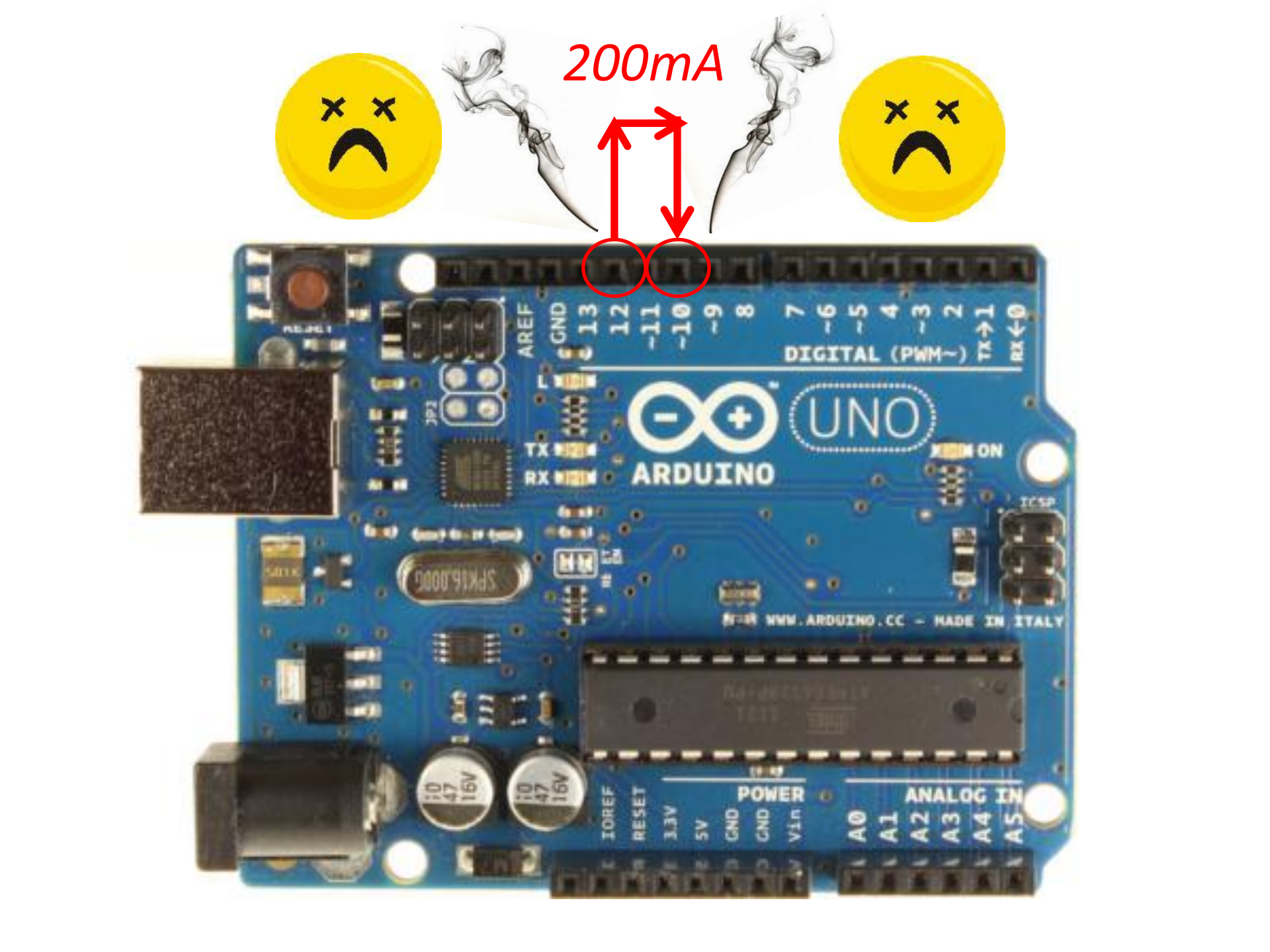
digitalWrite(12,HIGH);

pinMode(10,INPUT);

digitalWrite(10,LOW);



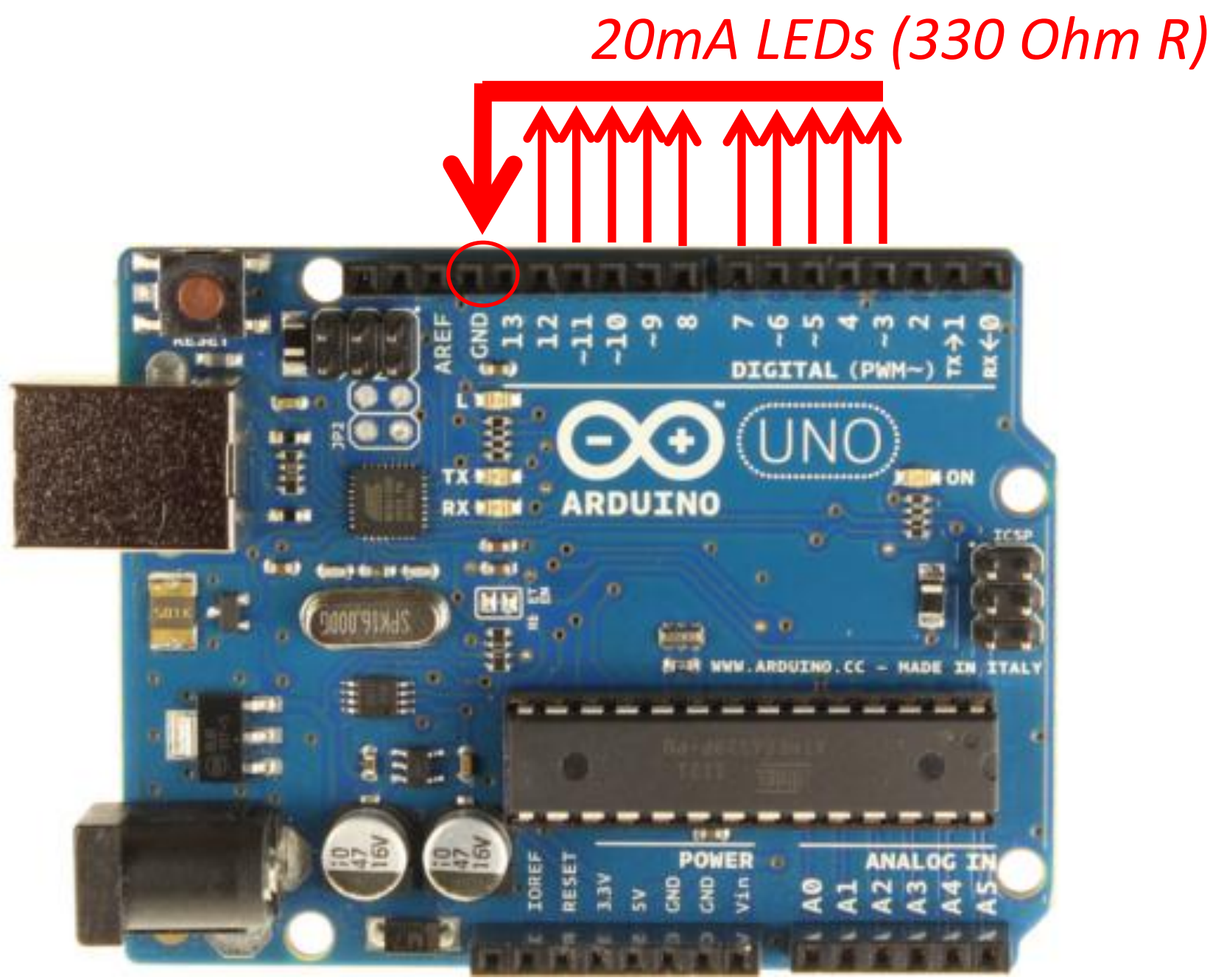

200mA
AREF
GND
13
12
~11
~10
9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
UNO
ARDUINO
ON
TX
RX
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

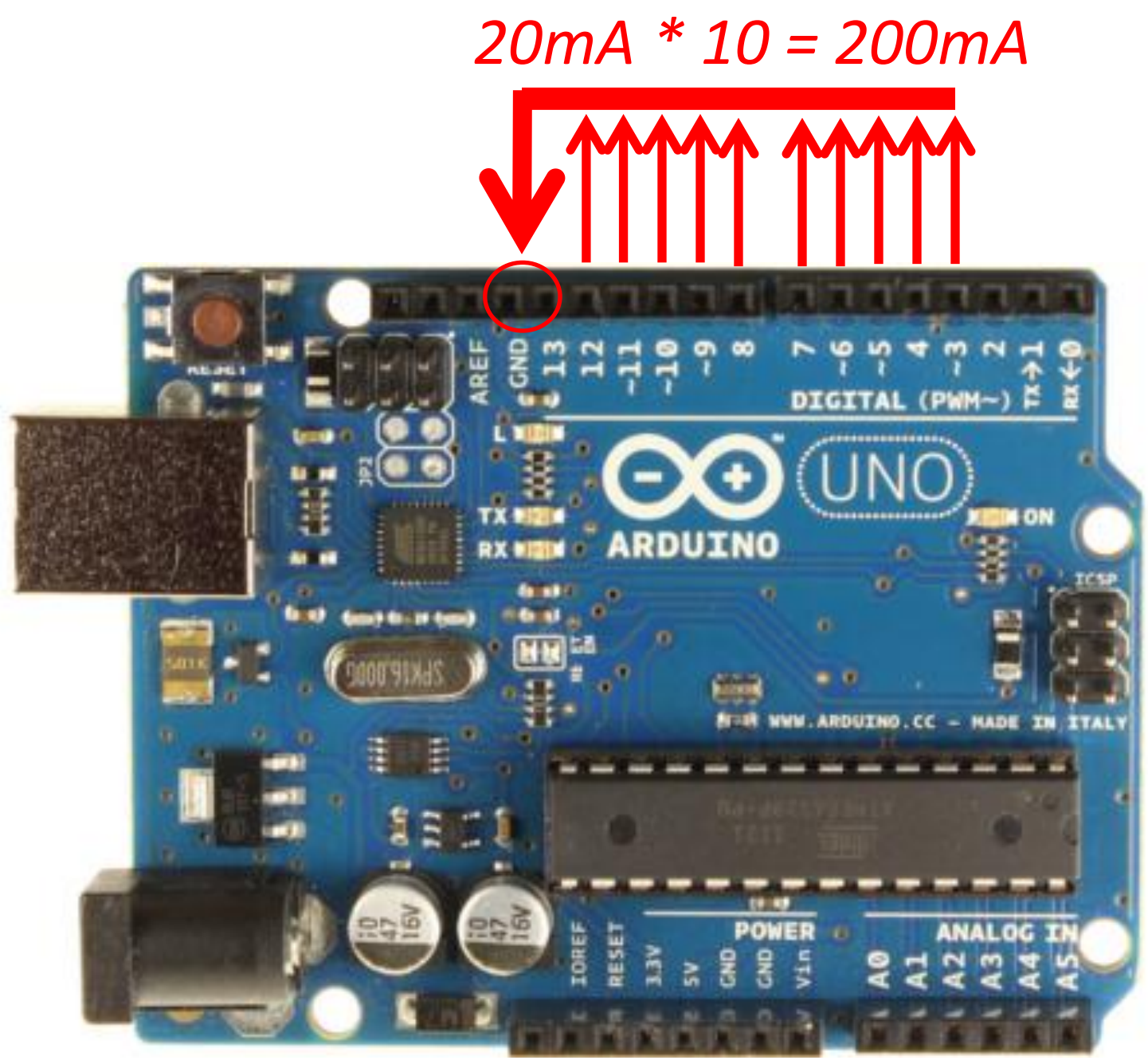

200mA
AREF
GND
13
12
~11
~10
~9
8
7
6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
UNO
ARDUINO
ON
TX
RX
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

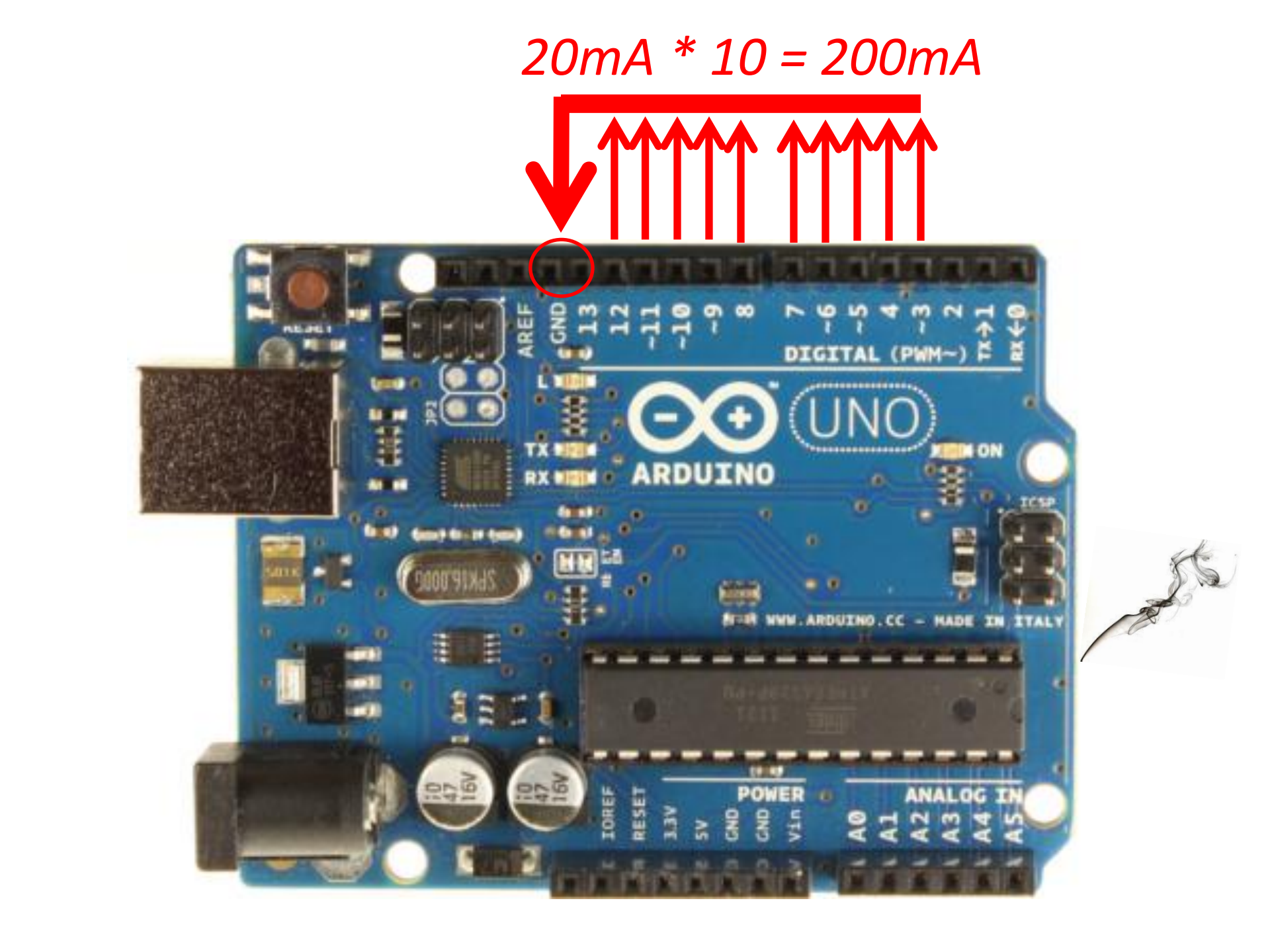
Ways to Kill an Arduino

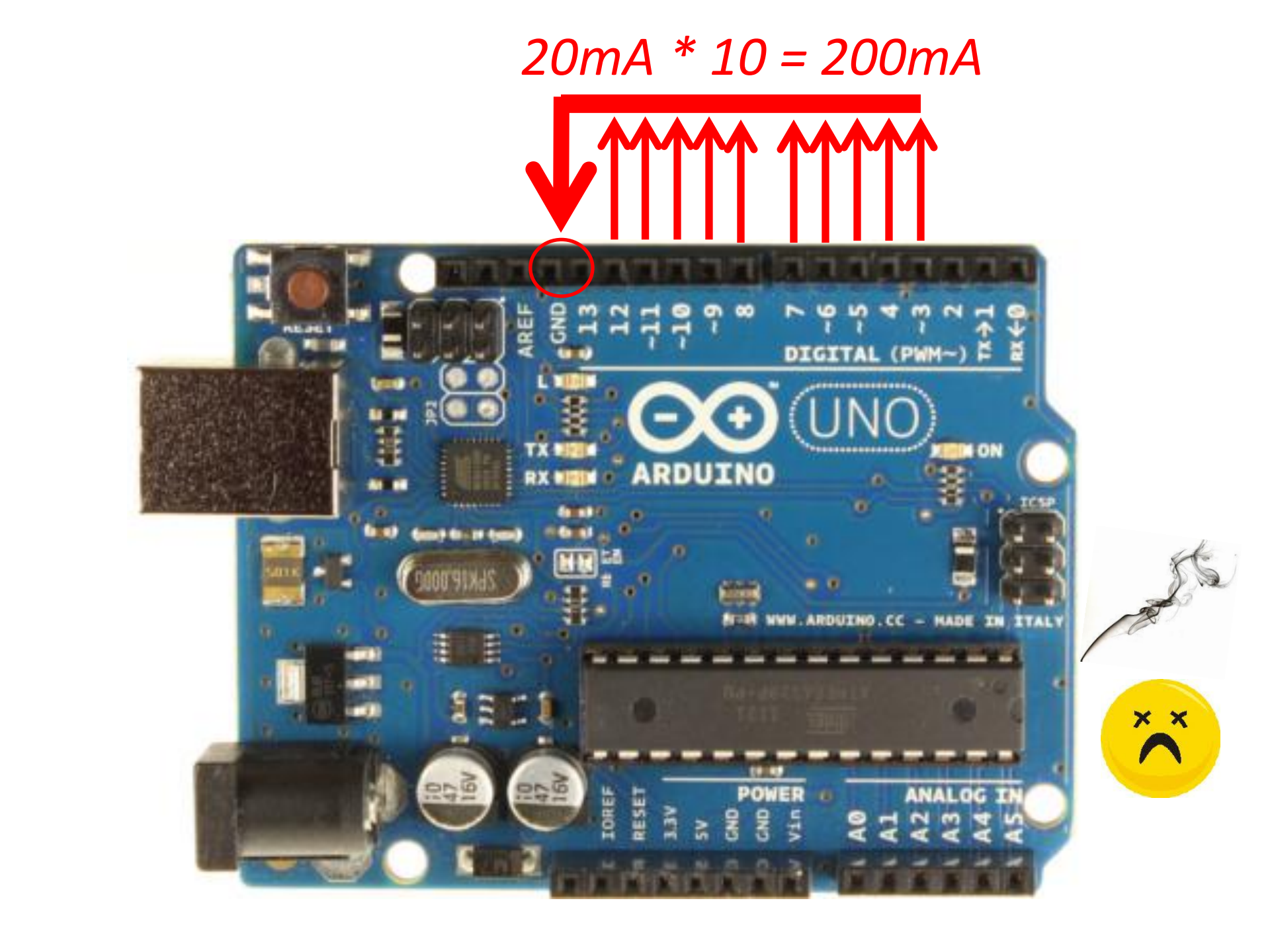
Easily Possible

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


20mA LEDs (330 Ohm R)
AREF
GND
13
12
~11
~10
~9
8
7
6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
UNO
ARDUINO
TX
RX
ON
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5


20mA * 10 = 200mA
AREF
GND
13
12
~11
~10
~9
8
7
6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
UNO
ARDUINO
ON
TX
RX
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5


20mA * 10 = 200mA
AREF
GND
13
12
~11
~10
~9
8
7
6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
UNO
ARDUINO
TX
RX
ON
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5


20mA * 10 = 200mA
DIGITAL (PWM~)
UNO
ARDUINO
ON
TX
RX
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN

Ways to Kill an Arduino

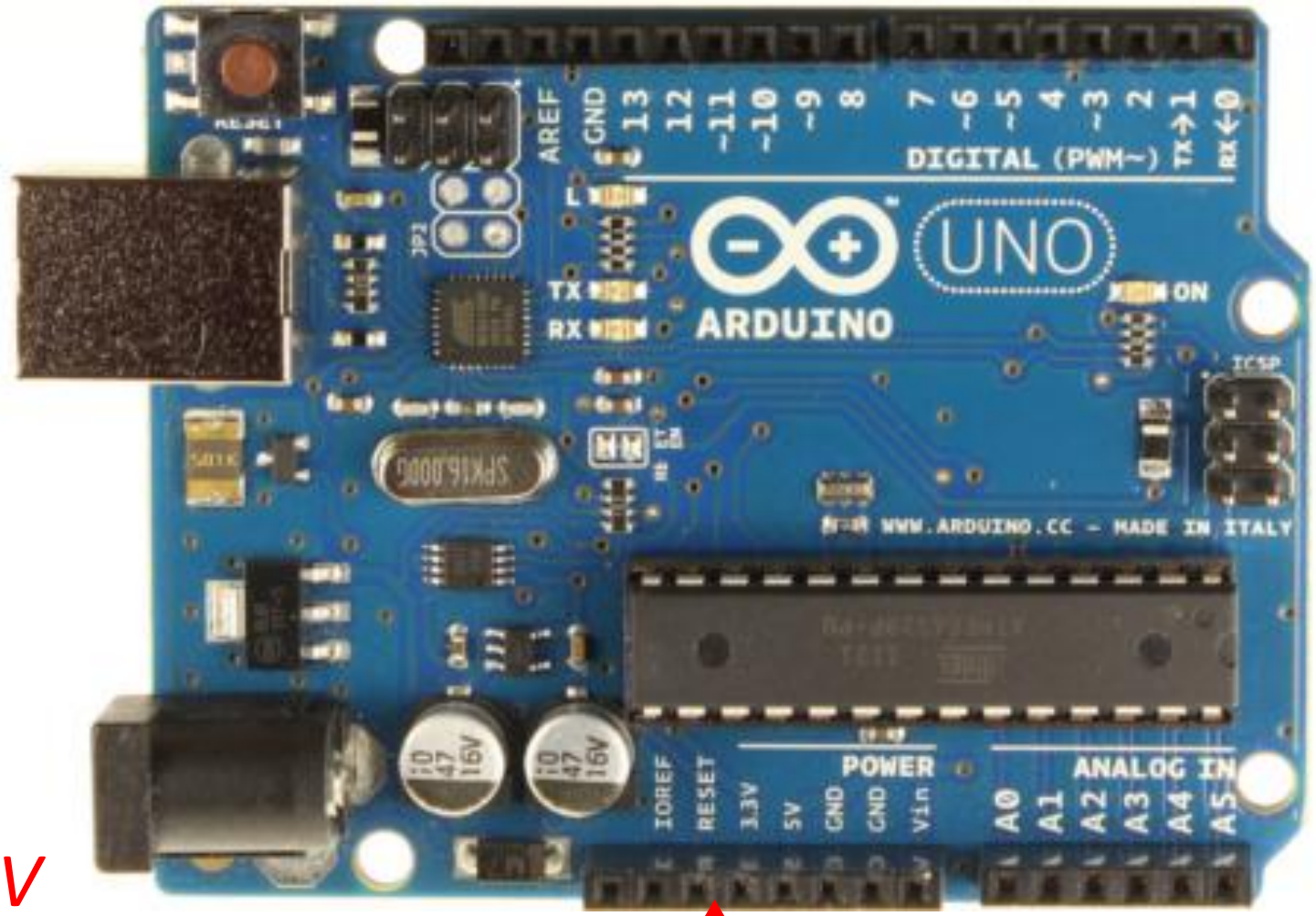
Easily Possible

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

Silly

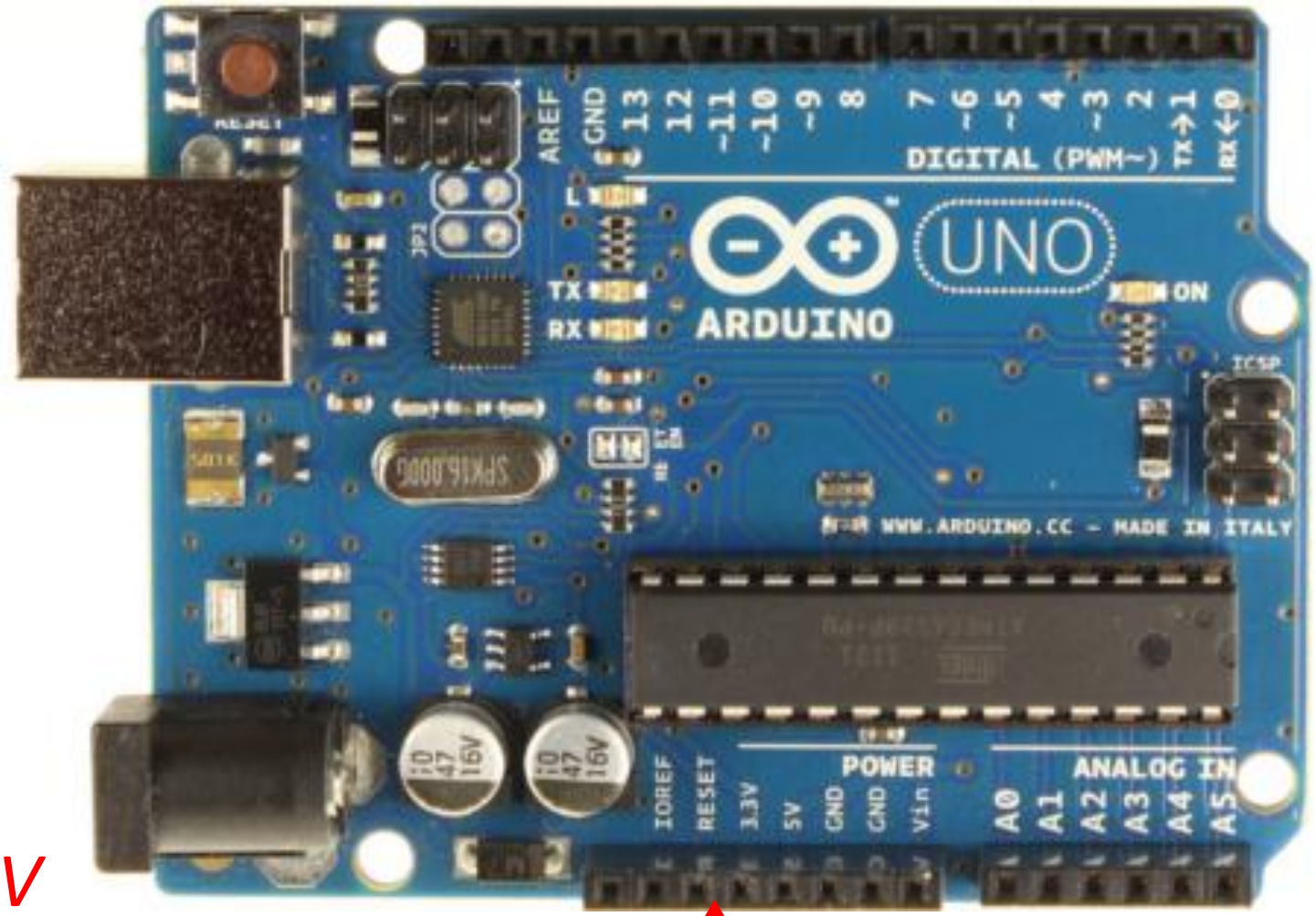
Apply > 3.3V Input to the 3.3V Output Pin

Other components connected
To 3.3V Pin



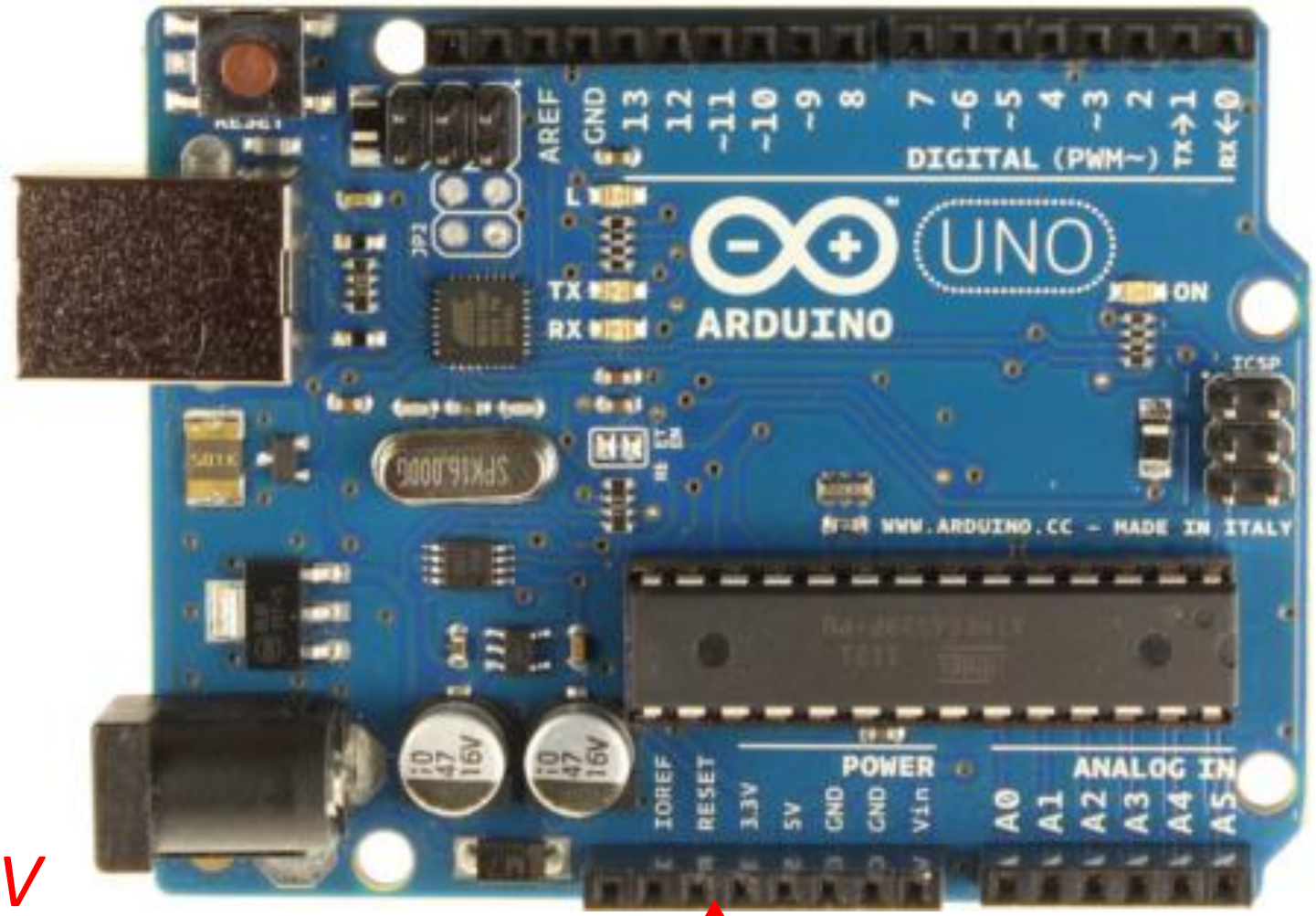
> 3.3V

Other components connected
To 3.3V Pin



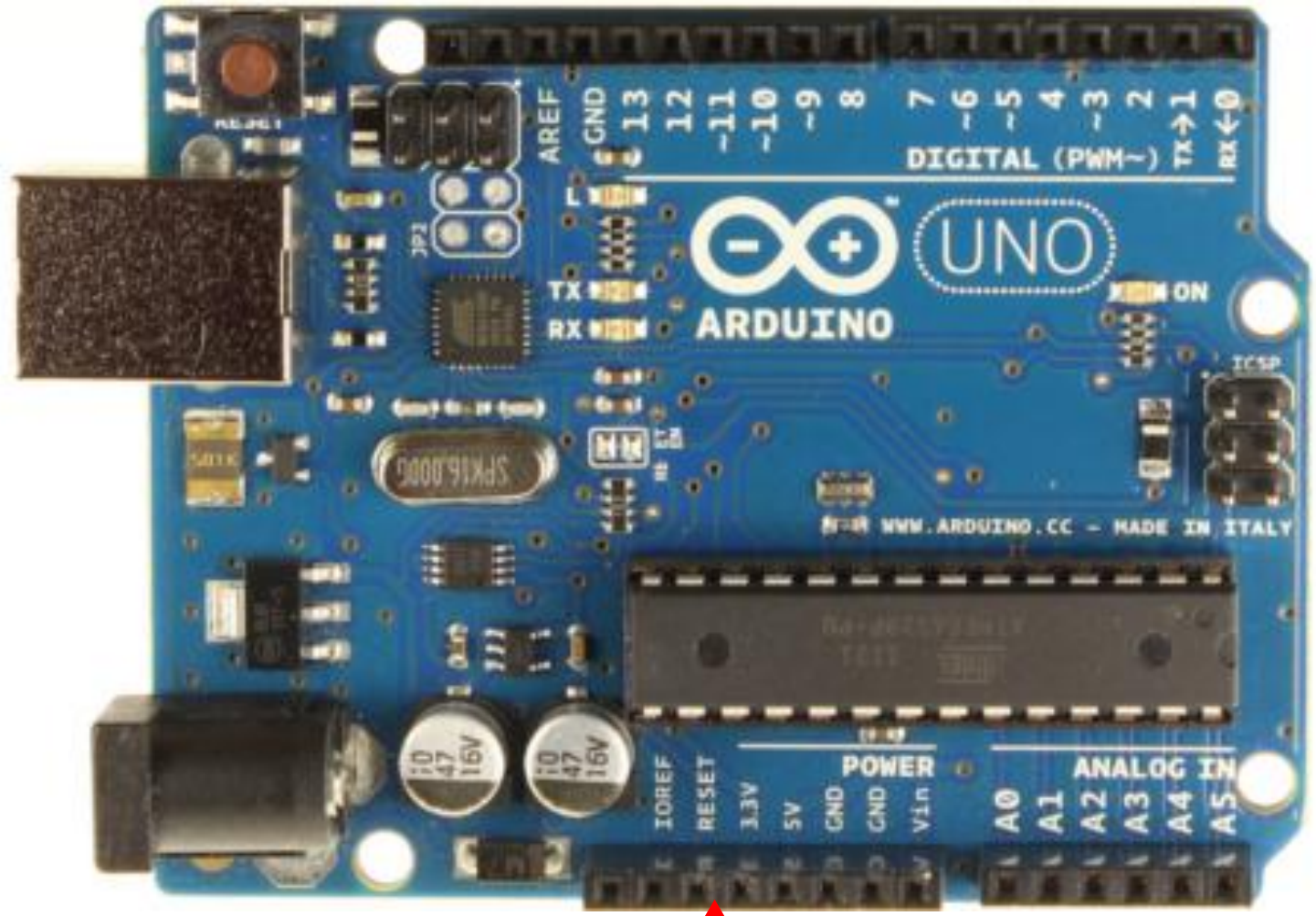
> 3.3V

Other components connected
To 3.3V Pin



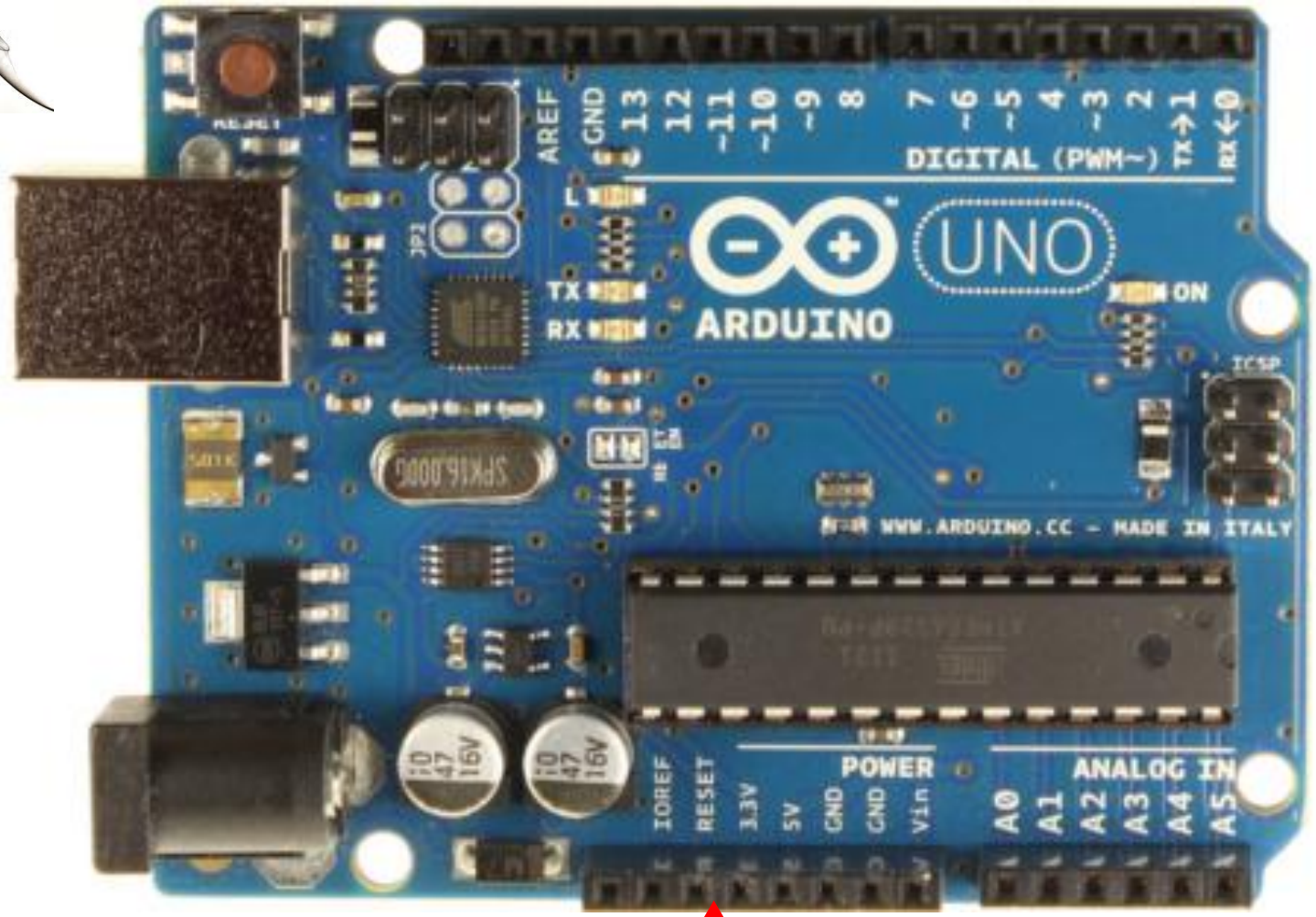
> 3.3V

Other components connected
To 3.3V Pin



> 9V

Other components connected
To 3.3V Pin

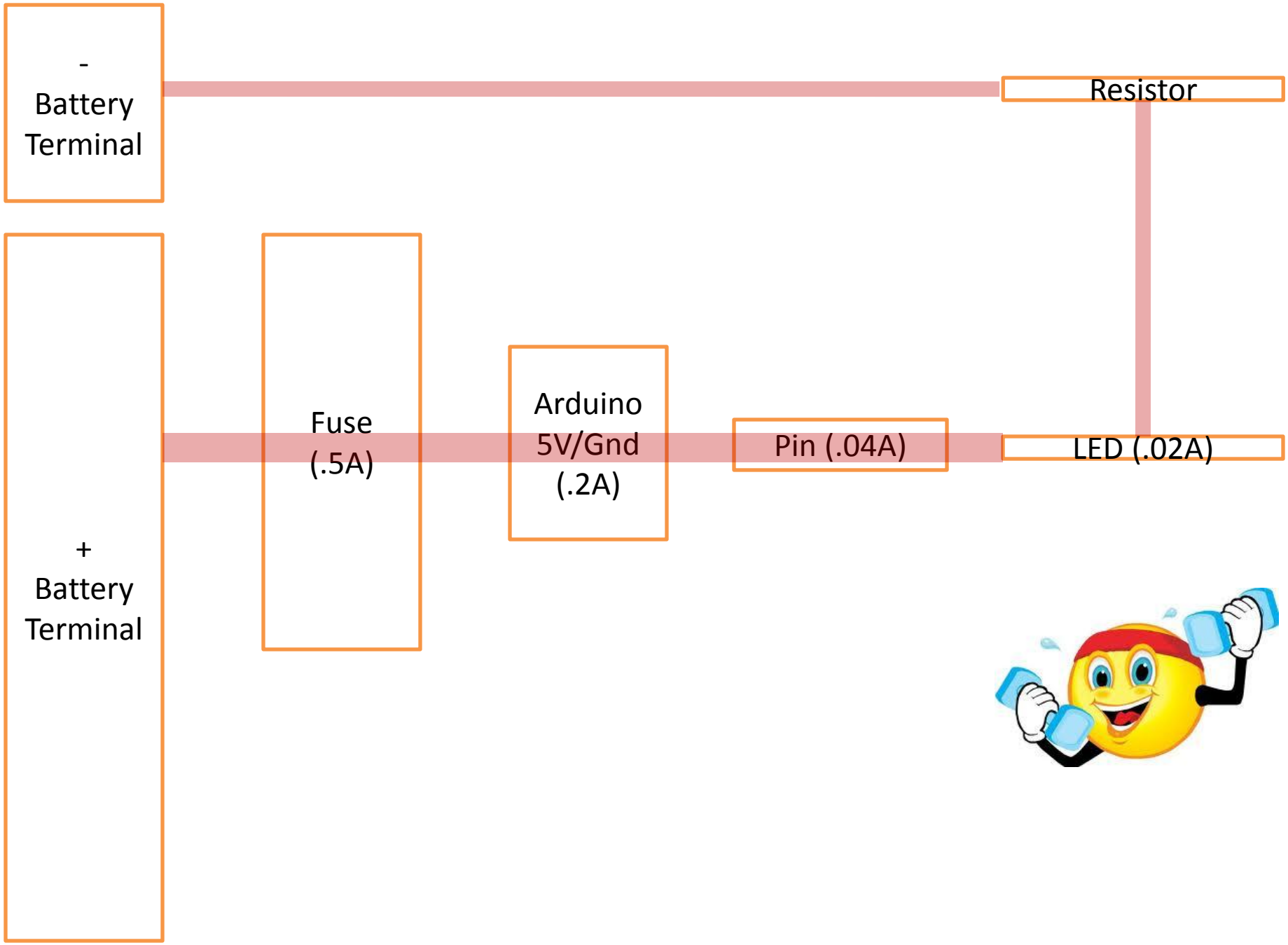


> 9V

Other components connected
To 3.3V Pin

> 9V

NO SMOKING


-
Battery
Terminal
Resistor
+
Battery
Terminal
Fuse
(.5A)
Arduino
5V/Gnd
(.2A)
Pin (.04A)
LED (.02A)

-
Battery
Terminal

+
Battery
Terminal

Fuse
(.5A)

Arduino
5V/Gnd
(.2A)

Pin (.04A)

.2A
(25Ohm)

Idle Motor(.32A)

Transistor

“the most important invention of the 20th century”

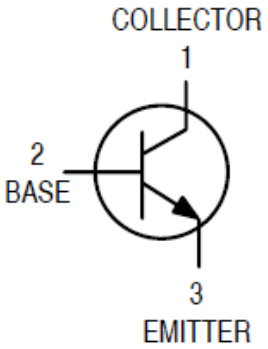
Transistor

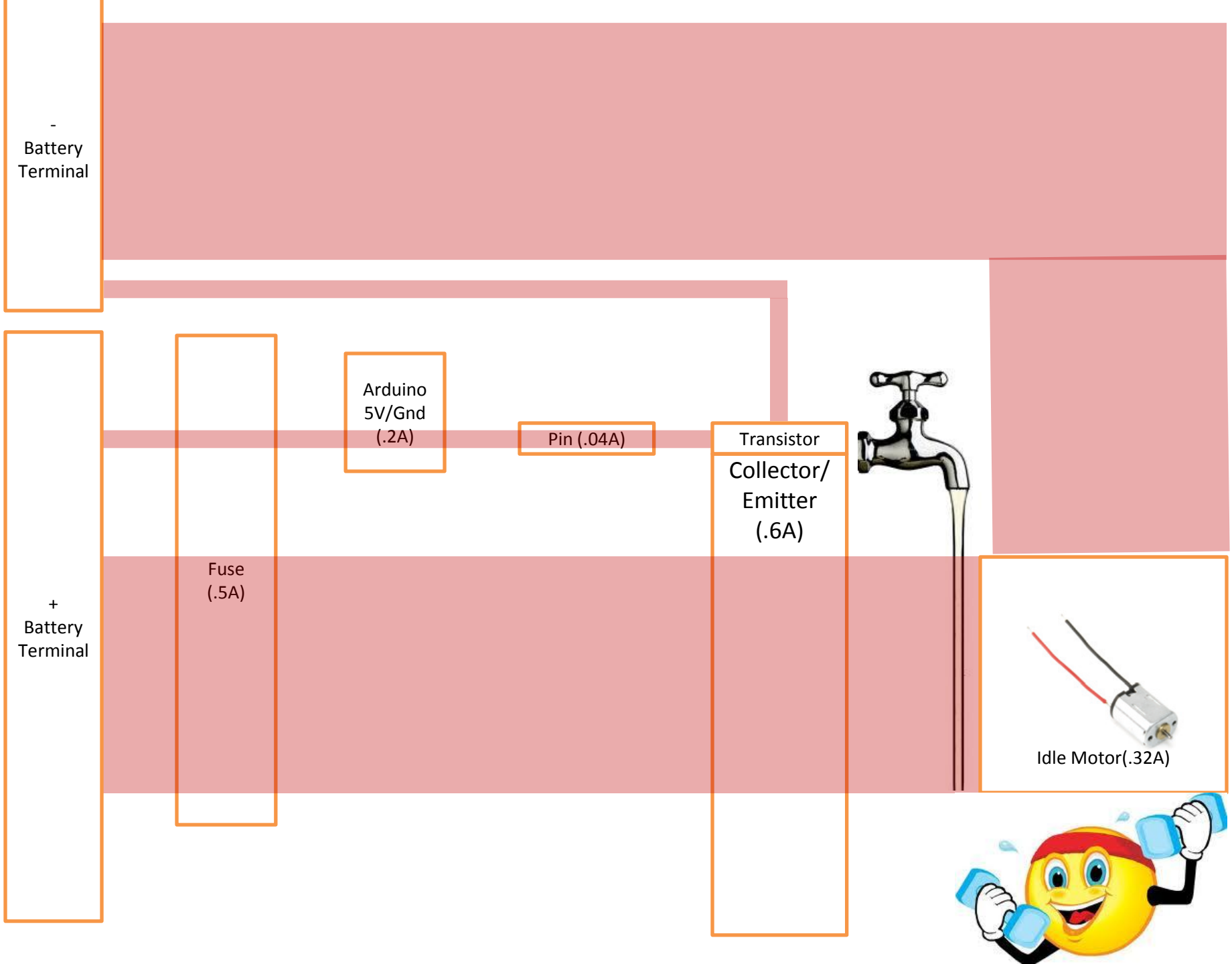
ANALOGY ALERT

Valve

MAXIMUM RATINGS (T_A = 25°C unless otherwise noted)

Characteristic	Symbol	Value	Unit
Collector–Emitter Voltage	V_{CEO}	40	Vdc
Collector–Base Voltage	V_{CBO}	75	Vdc
Emitter–Base Voltage	V_{EBO}	6.0	Vdc
Collector Current – Continuous	I_C	600	mAdc
Total Device Dissipation @ T_A = 25°C Derate above 25°C	P_D	625 5.0	mW mW/°C
Total Device Dissipation @ T_C = 25°C Derate above 25°C	P_D	1.5 12	W mW/°C
Operating and Storage Junction Temperature Range	T_J, T_{stg}	–55 to +150	°C




-
Battery
Terminal
+
Battery
Terminal
Fuse
(.5A)
Arduino
5V/Gnd
(.2A)
Pin (.04A)
Transistor
Collector/
Emitter
(.6A)
Idle Motor(.32A)

- Battery Terminal

+ Battery Terminal

Fuse (.5A)

Arduino 5V/Gnd (.2A)

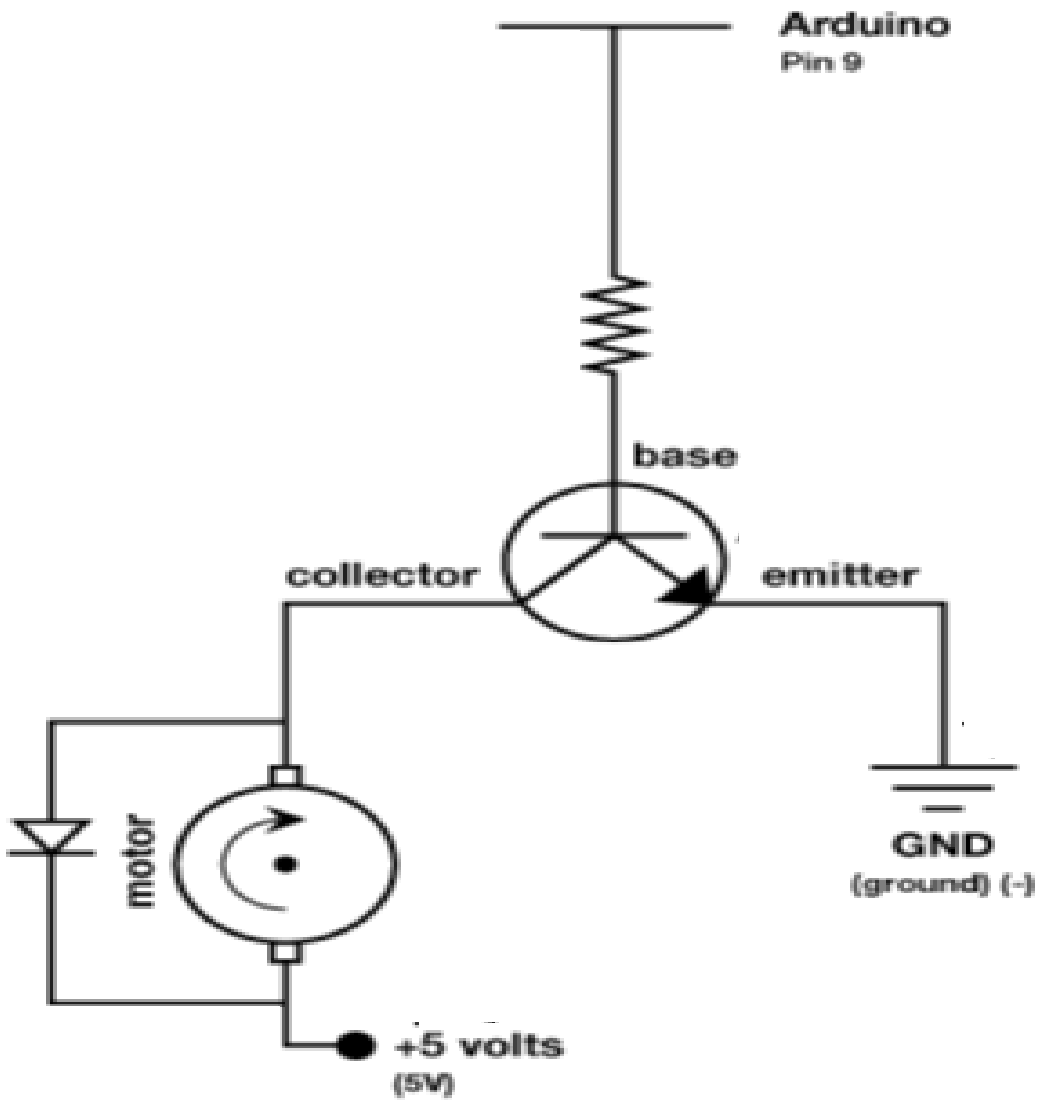
Pin (.04A)

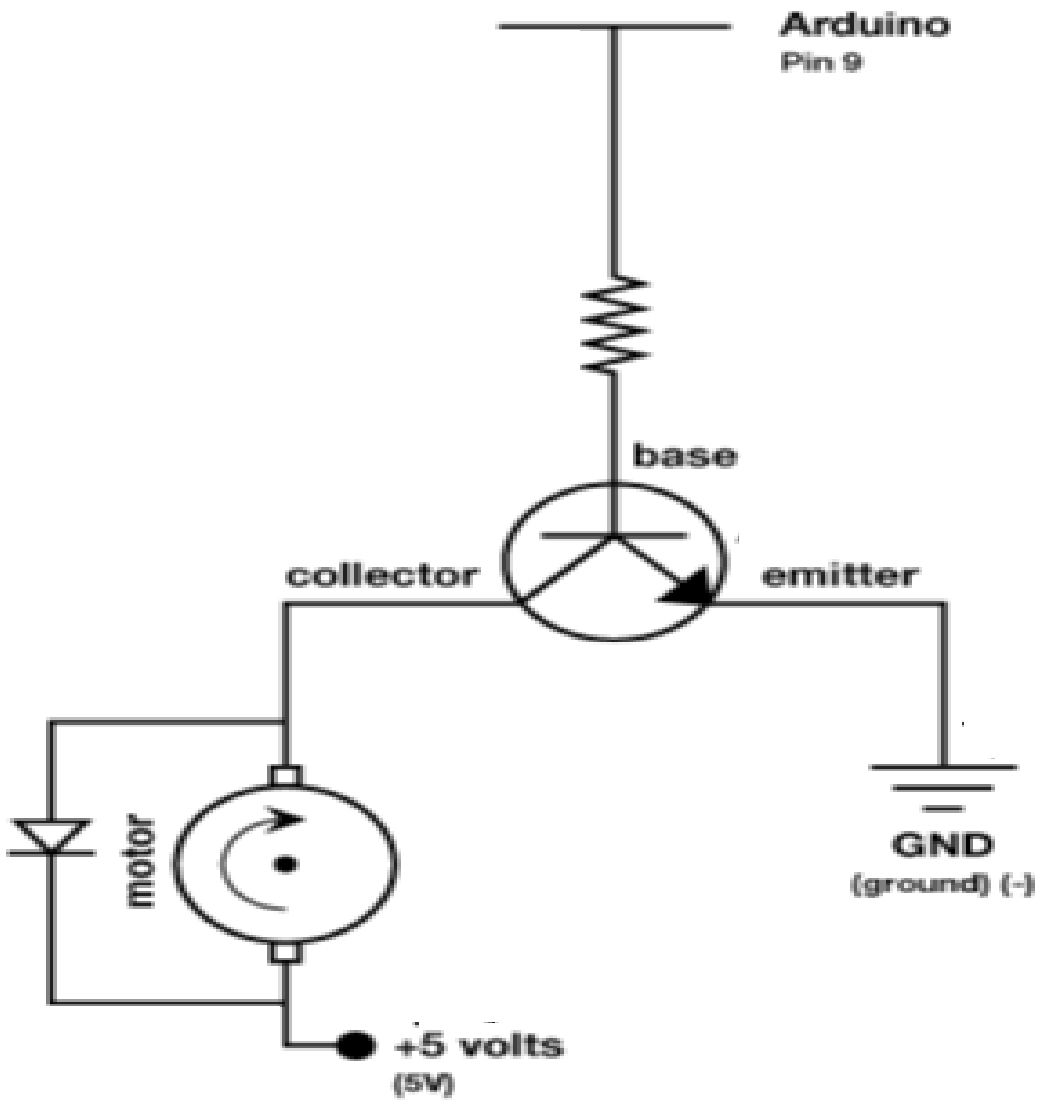
Transistor

Collector /Emitter (.6A)

Peak Efficiency Motor(1.2A)

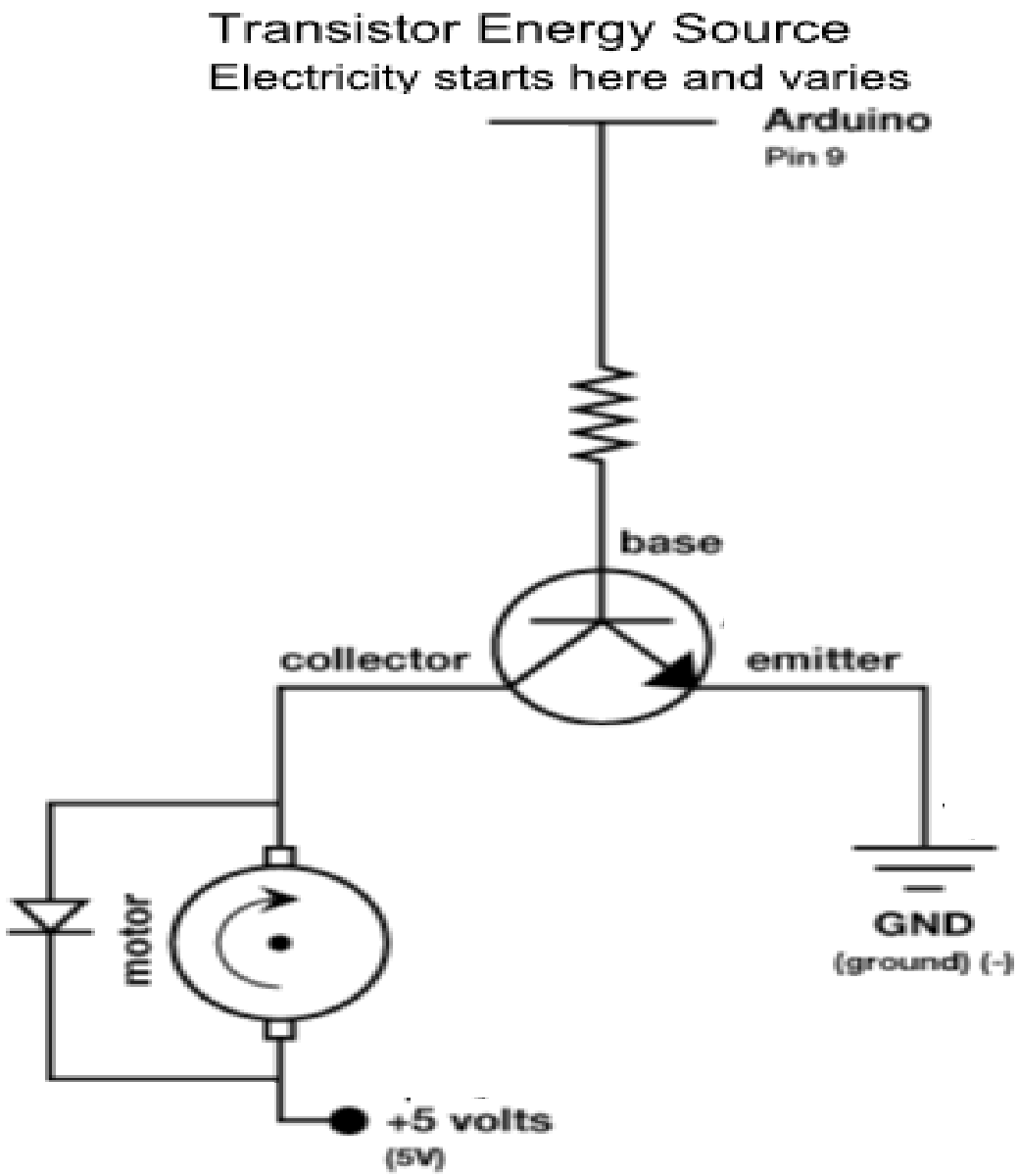
- Circuit #3

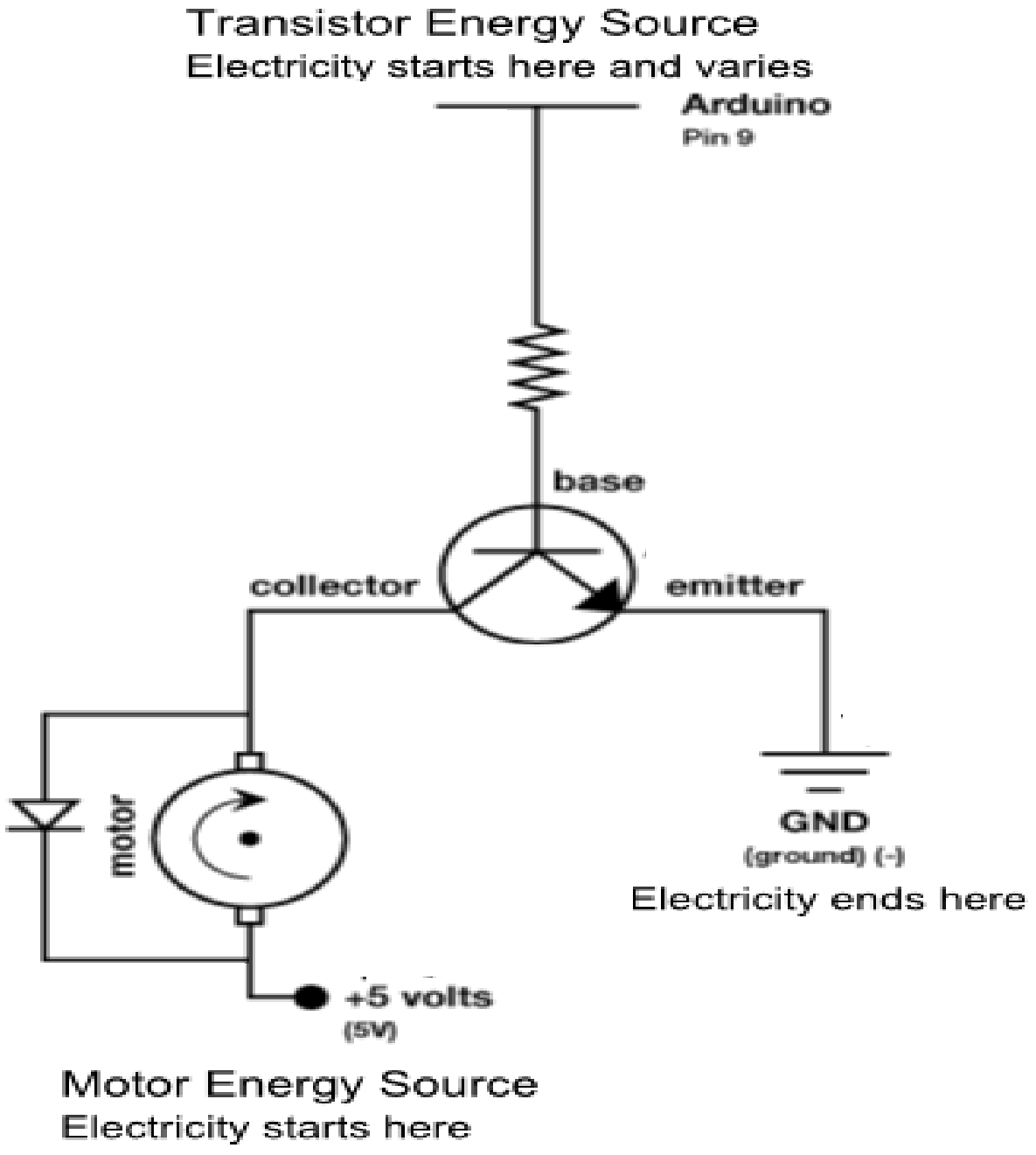

Arduino
Pin 9
base
collector
emitter
motor
GND
(ground) (-)
+5 volts
(5V)

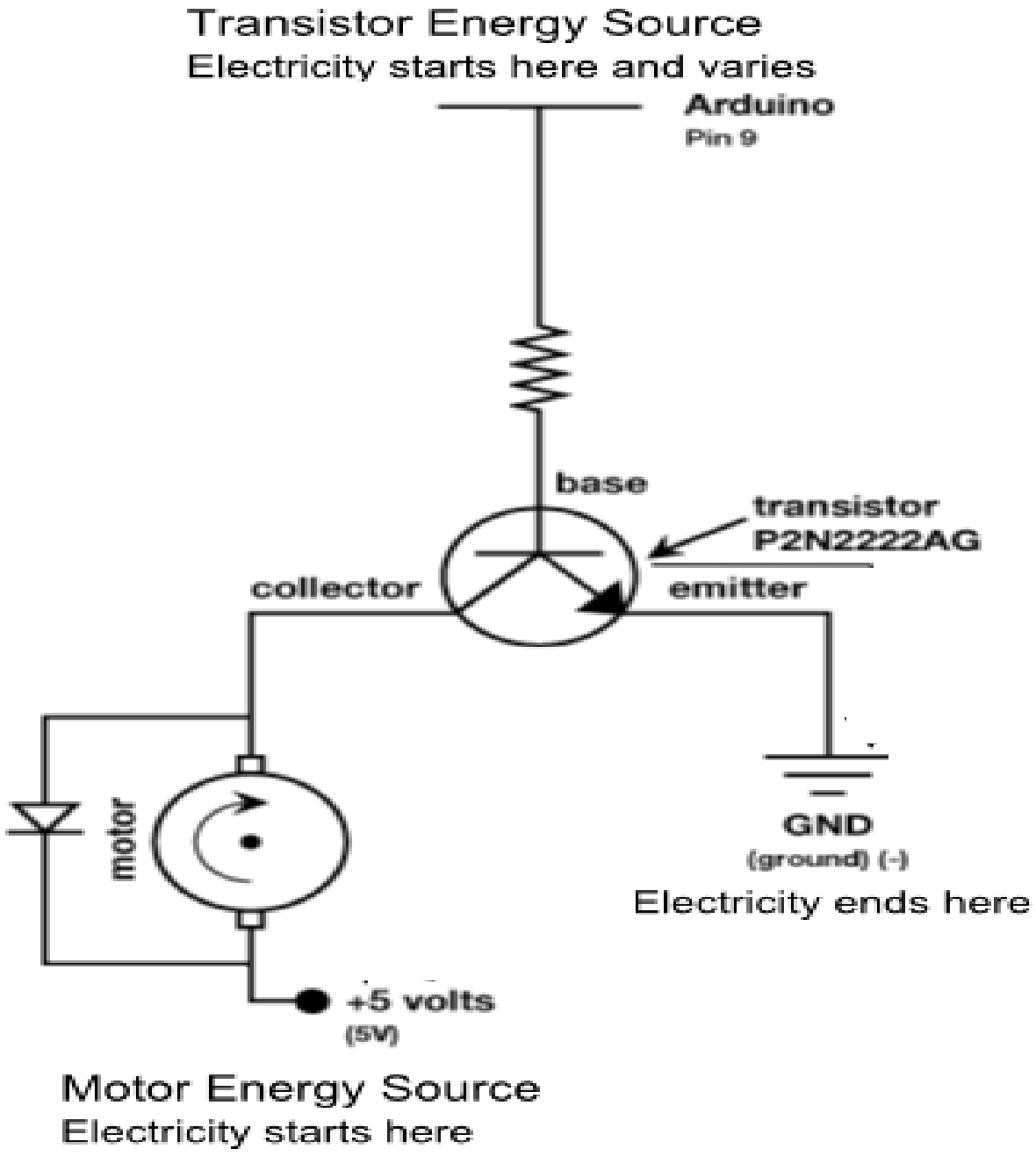


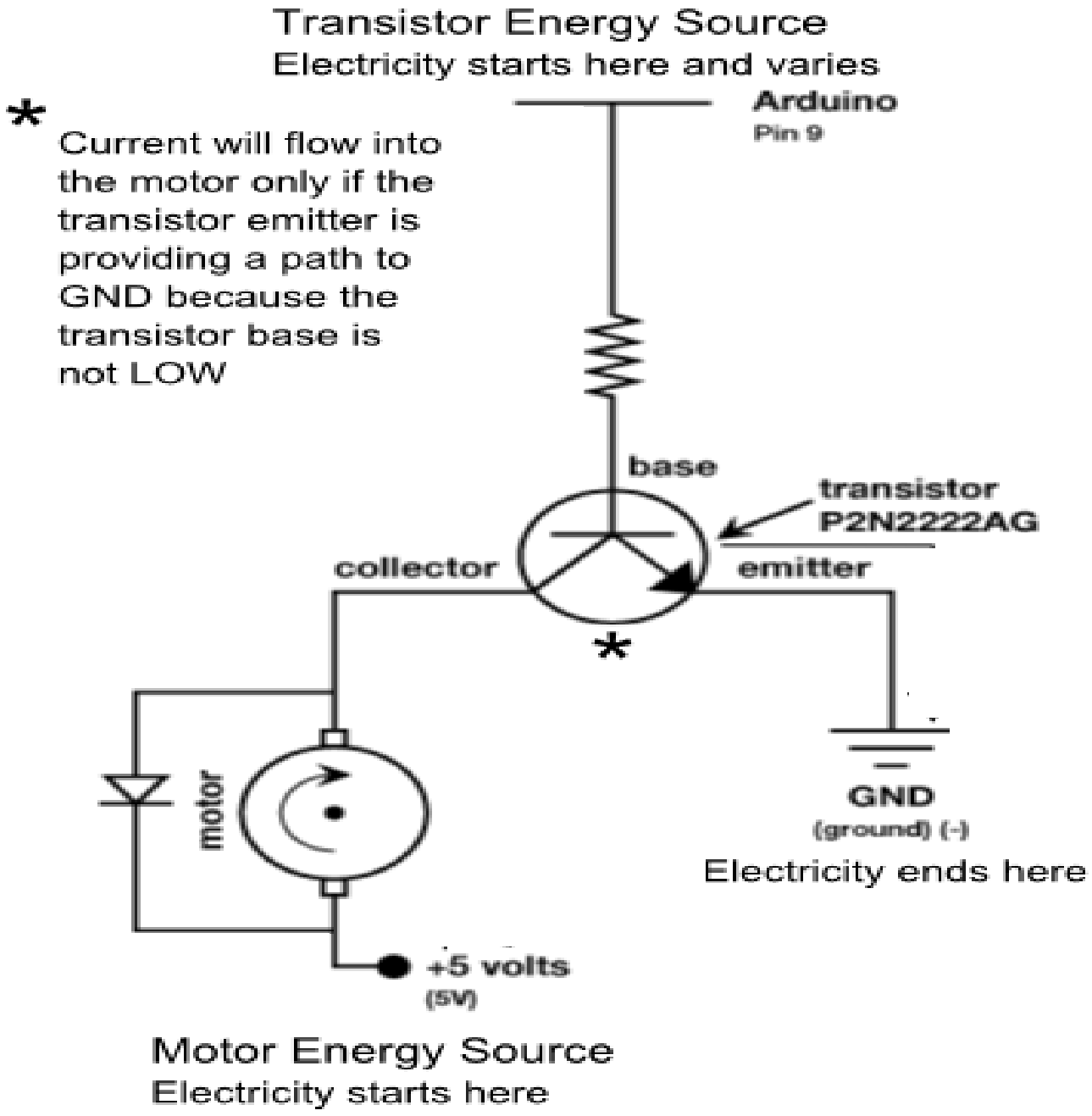
Motor Energy Source

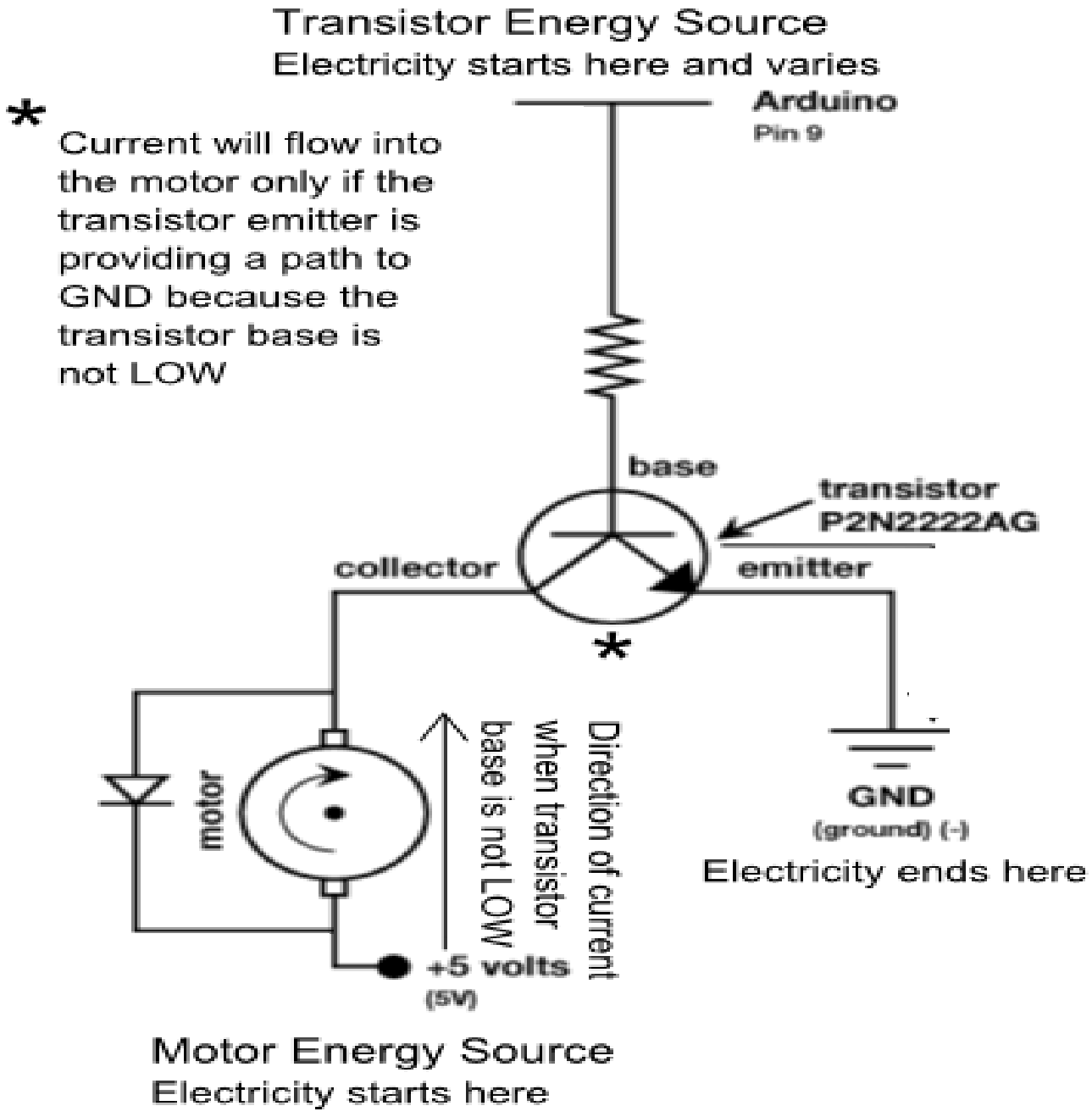
Electricity starts here

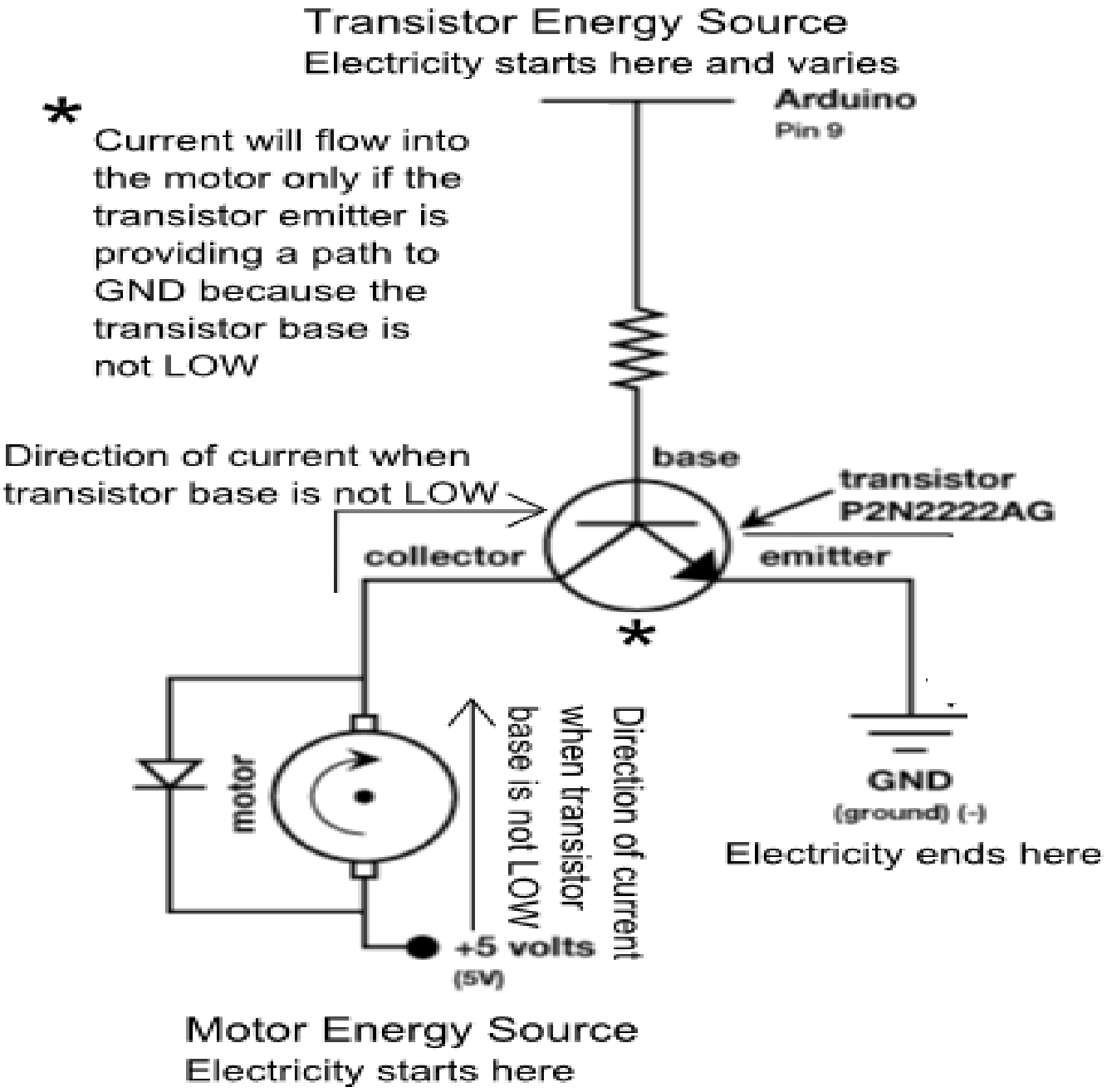

Transistor Energy Source
Electricity starts here and varies
Arduino
Pin 9
base
collector
emitter
motor
GND
(ground) (-)
+5 volts
(5V)
Motor Energy Source
Electricity starts here

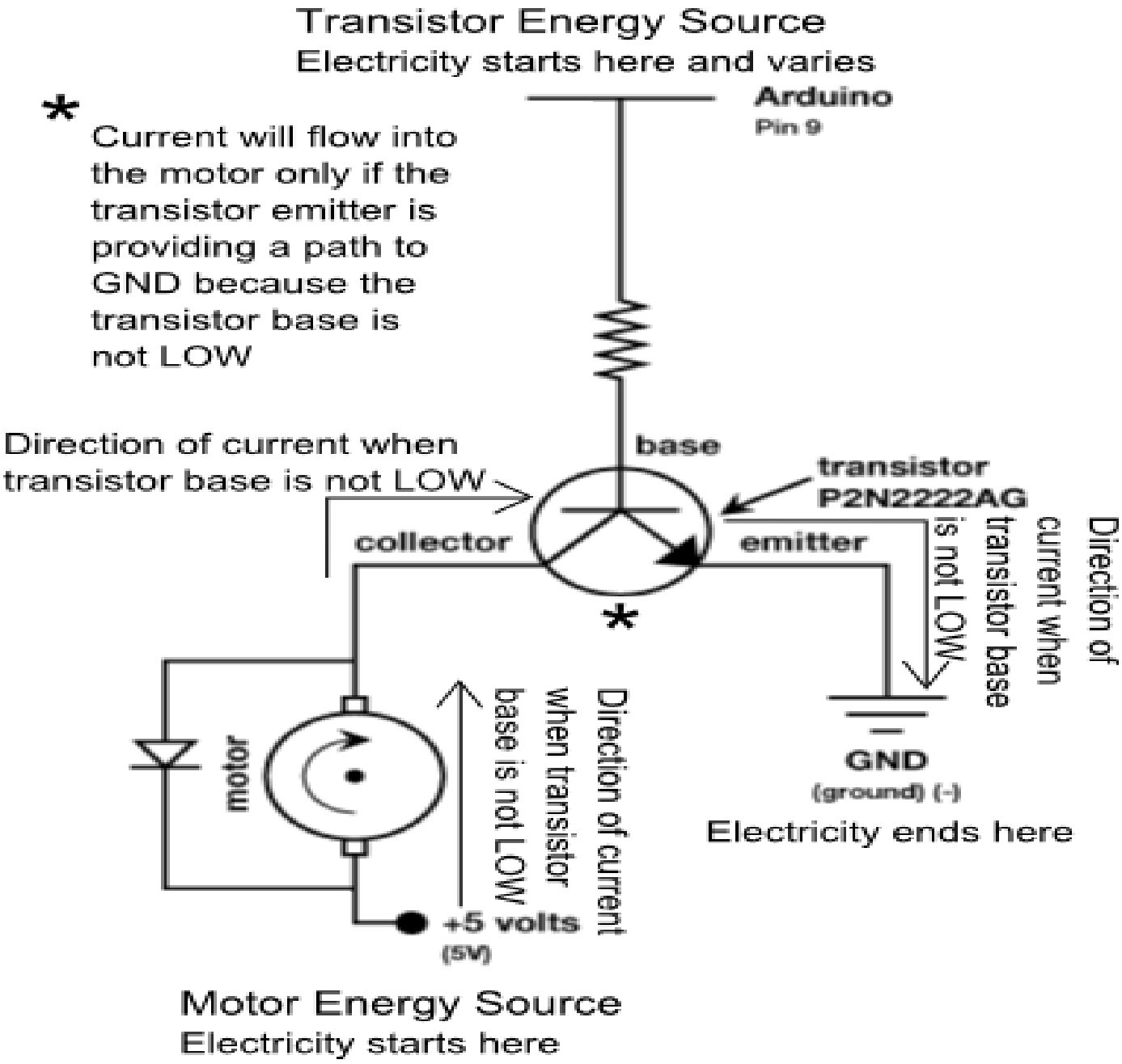

Transistor Energy Source
Electricity starts here and varies
Arduino
Pin 9
base
collector
emitter
motor
GND
(ground) (-)
Electricity ends here
+5 volts
(5V)
Motor Energy Source
Electricity starts here

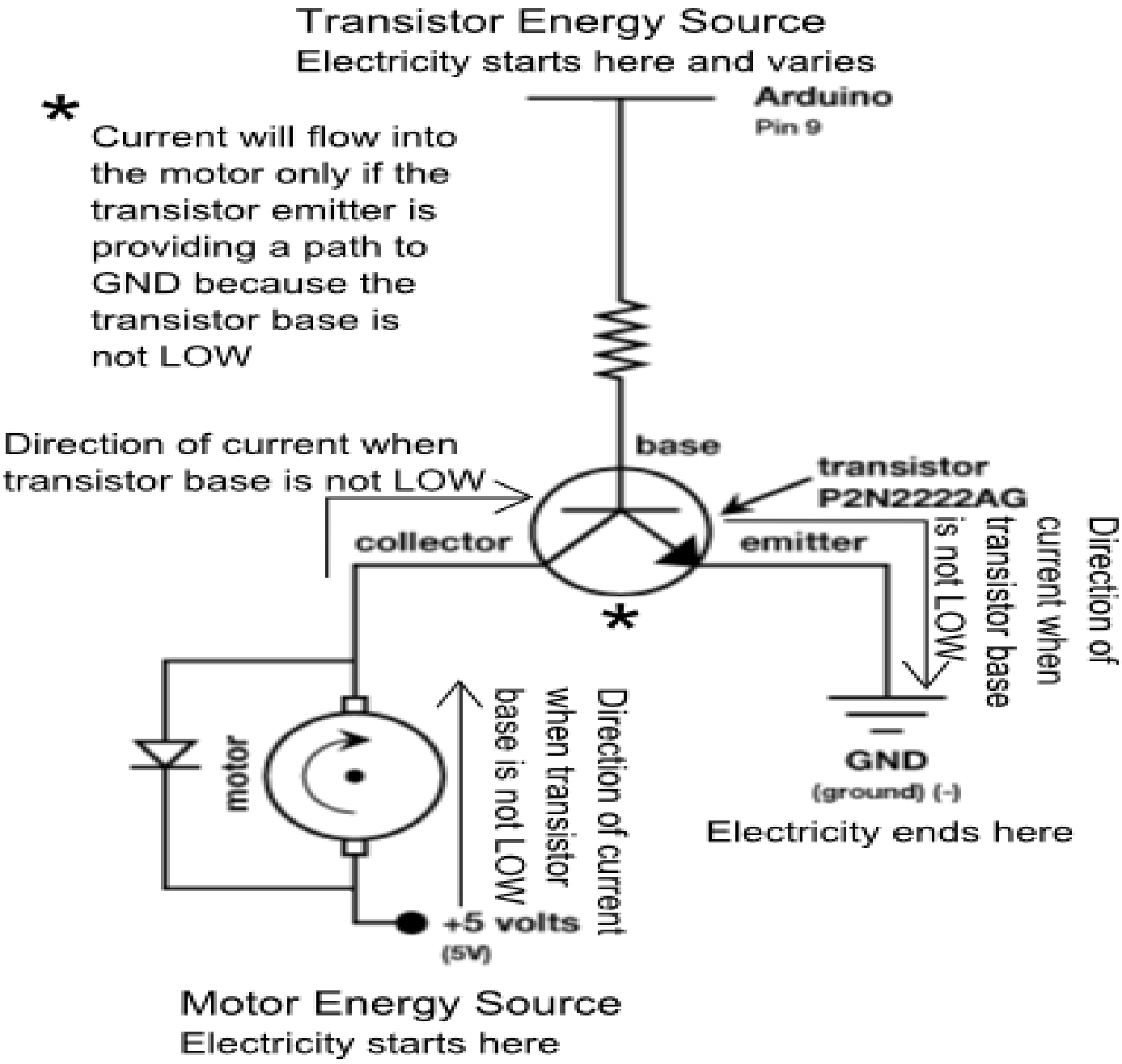

Transistor Energy Source
Electricity starts here and varies
Arduino
Pin 9
base
transistor
P2N2222AG
collector
emitter
motor
GND
(ground) (-)
Electricity ends here
+5 volts
(5V)
Motor Energy Source
Electricity starts here

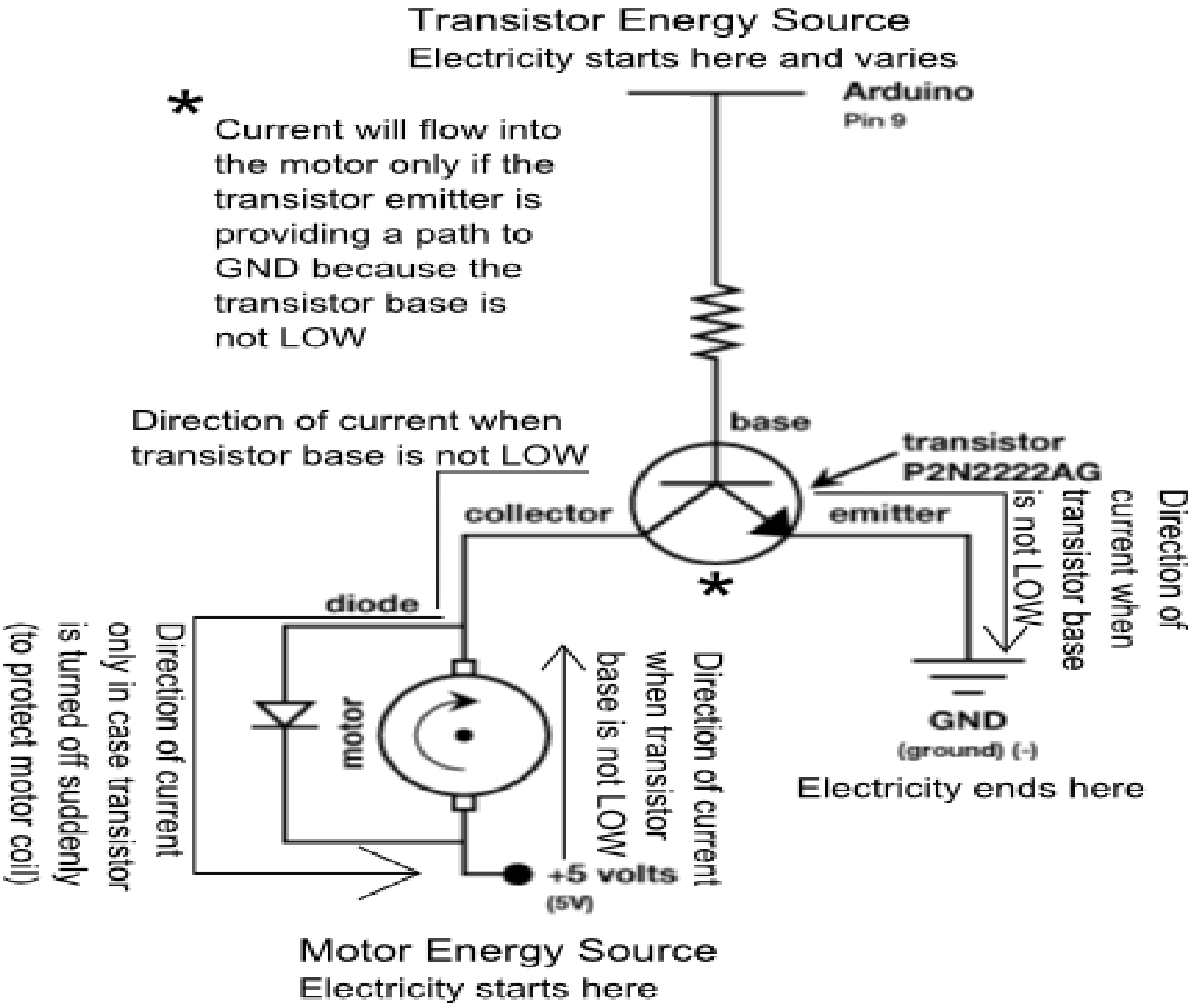

Transistor Energy Source
Electricity starts here and varies
*
Current will flow into the motor only if the transistor emitter is providing a path to GND because the transistor base is not LOW
Arduino
Pin 9
base
transistor
P2N2222AG
collector
emitter
*
motor
GND
(ground) (-)
Electricity ends here
+5 volts
(5V)
Motor Energy Source
Electricity starts here


Transistor Energy Source
Electricity starts here and varies
*
Current will flow into the motor only if the transistor emitter is providing a path to GND because the transistor base is not LOW
Arduino
Pin 9
base
transistor
P2N2222AG
collector
emitter
*
motor
Direction of current when transistor base is not LOW
+5 volts
(5V)
GND
(ground) (-)
Electricity ends here
Motor Energy Source
Electricity starts here

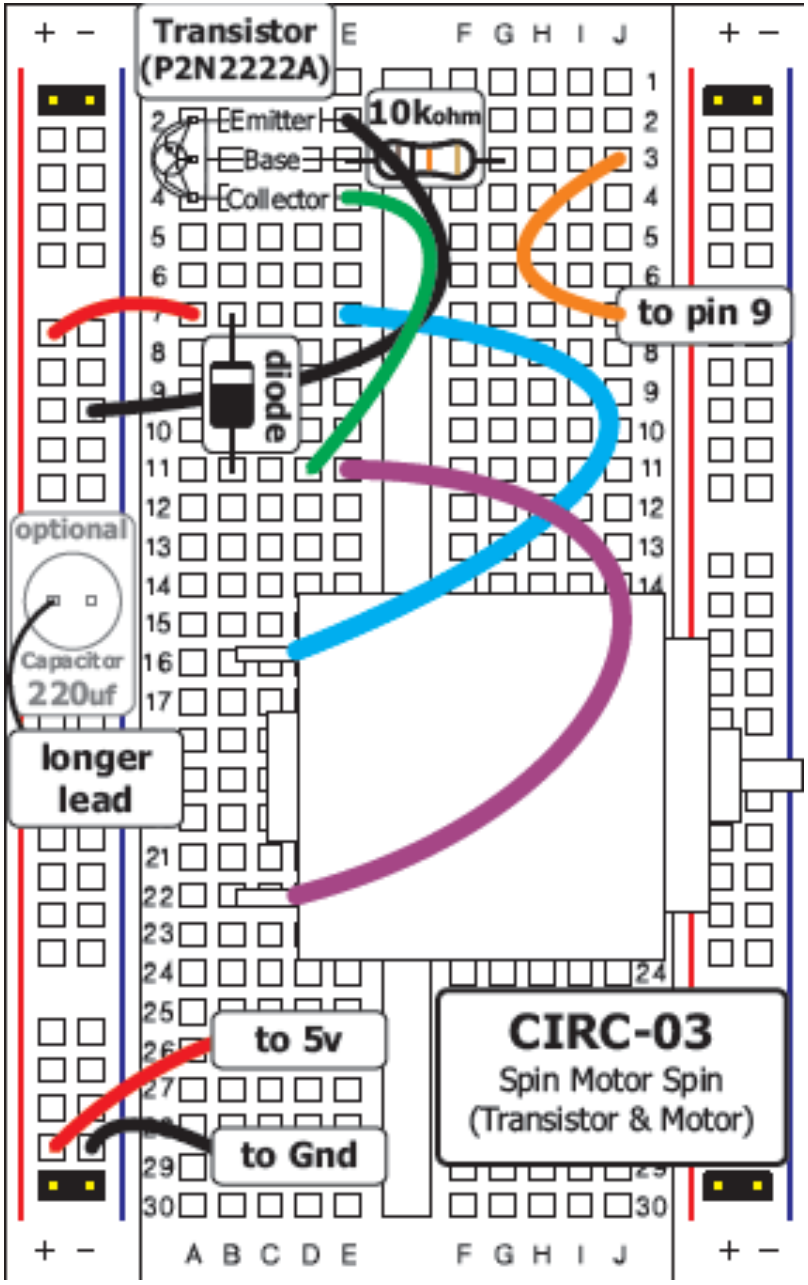

Transistor Energy Source
Electricity starts here and varies
*
Current will flow into the motor only if the transistor emitter is providing a path to GND because the transistor base is not LOW
Arduino
Pin 9
Direction of current when transistor base is not LOW
base
transistor
P2N2222AG
collector
emitter
*
motor
Direction of current when transistor base is not LOW
+5 volts
(5V)
GND
(ground) (-)
Electricity ends here
Motor Energy Source
Electricity starts here


Transistor Energy Source
Electricity starts here and varies
*
Current will flow into the motor only if the transistor emitter is providing a path to GND because the transistor base is not LOW
Arduino
Pin 9
Direction of current when transistor base is not LOW
base
transistor
P2N2222AG
collector
emitter
*
Direction of current when transistor base is not LOW
motor
Direction of current when transistor base is not LOW
+5 volts
(5V)
GND
(ground) (-)
Electricity ends here
Motor Energy Source
Electricity starts here


Transistor Energy Source
Electricity starts here and varies
*
Current will flow into the motor only if the transistor emitter is providing a path to GND because the transistor base is not LOW
Arduino
Pin 9
Direction of current when transistor base is not LOW
base
transistor
P2N2222AG
collector
emitter
*
Direction of current when transistor base is not LOW
motor
Direction of current when transistor base is not LOW
+5 volts
(5V)
GND
(ground) (-)
Electricity ends here
Motor Energy Source
Electricity starts here


Transistor Energy Source
Electricity starts here and varies
*
Current will flow into the motor only if the transistor emitter is providing a path to GND because the transistor base is not LOW
Arduino
Pin 9
base
transistor
P2N2222AG
collector
emitter
Direction of current when transistor base is not LOW
Direction of current when transistor base is not LOW
diode
motor
Direction of current only in case transistor is turned off suddenly (to protect motor coil)
Direction of current when transistor base is not LOW
+5 volts
(5V)
GND
(ground) (-)
Electricity ends here
Motor Energy Source
Electricity starts here

- The Breadboard


Transistor
(P2N2222A)
Emitter
Base
Collector
10kohm
diode
to pin 9
optional
Capacitor
220uf
longer
lead
to 5v
to Gnd
CIRC-03
Spin Motor Spin
(Transistor & Motor)

- The Sketch

Code:

```
int motorPin = 9;

void setup() {
  pinMode(motorPin, OUTPUT);
}


void loop() {
  for (int i = 0; i < 256; i++){
    analogWrite(motorPin, i);
    delay(50);
   }
}
```