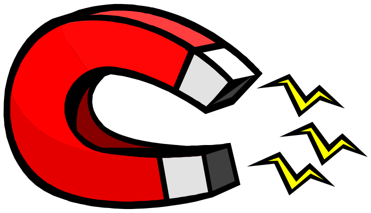
NATURE Sunday Academy 2010-2011

"Fun with Physics and Computers"



Anne Denton and Alan Denton

North Dakota State University

Contact Information:

Dr. Anne Denton
Department of Computer Science
North Dakota State University
Fargo, ND 58108-6050
phone: 701-231-6748
e-mail: anne.denton@ndsu.edu

Dr. Alan Denton
Department of Physics
North Dakota State University
Fargo, ND 58108-6050
phone: 701-231-7036
e-mail: alan.denton@ndsu.edu

Outline of Activities

Three hands-on activities reveal the deep relationship between physics and computers:

1) Using readily available equipment, we'll explore fundamental connections between electricity and magnetism.

By holding a compass close to a current-carrying wire, we'll discover that an electric current generates a magnetic force. By inserting a magnet into a coil of wire, we'll discover that a moving magnet induces an electric current in a circuit. Using a battery, magnet, metallic screw, and wire, we'll assemble a simple electric motor.

2) Using a Radio Shack Electronic Sensors Lab, we'll build simple logic gates on a bread board and learn how computers store information and perform logical operations.

3) Using free "Game Maker" software, we'll explore binary numbers and learn about information storage and computer programming concepts.

Objectives of the Activities

- Illustrate that physics is the foundation of computer technologies
- Demonstrate basic connections between electricity and magnetism
- Enhance and promote physics instruction at tribal high schools
- Discuss how magnetic materials store digital (vs. analog) data
- Learn how computers use binary numbers to do logical operations
- Explore programming concepts using free Game Maker software

Activity 1: Electricity & Magnetism

Using a compass, paper, and pencil, we'll map out the direction of the magnetic force around a magnet and around a current-carrying wire. Here's how:

1) First lay a bar magnet on a sheet of paper. In what direction does the needle point very far from the magnet?

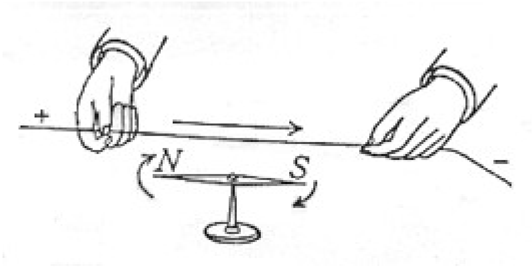
Now move the compass around the magnet and observe how the needle is deflected. At several points, draw an arrow on the paper in the direction the needle is pointing.

What do the arrows represent?

In what direction do the arrows point near the poles (ends) of the magnet?

From Electricity to Magnetism

In 1820, Hans Christian Oersted, a Danish physicist and chemist, discovered that an electric current induces magnetism. During a lecture, he noticed that the needle of a magnetic compass was deflected when current from a battery was switched on. This observation demonstrated the first connection between electricity and magnetism.



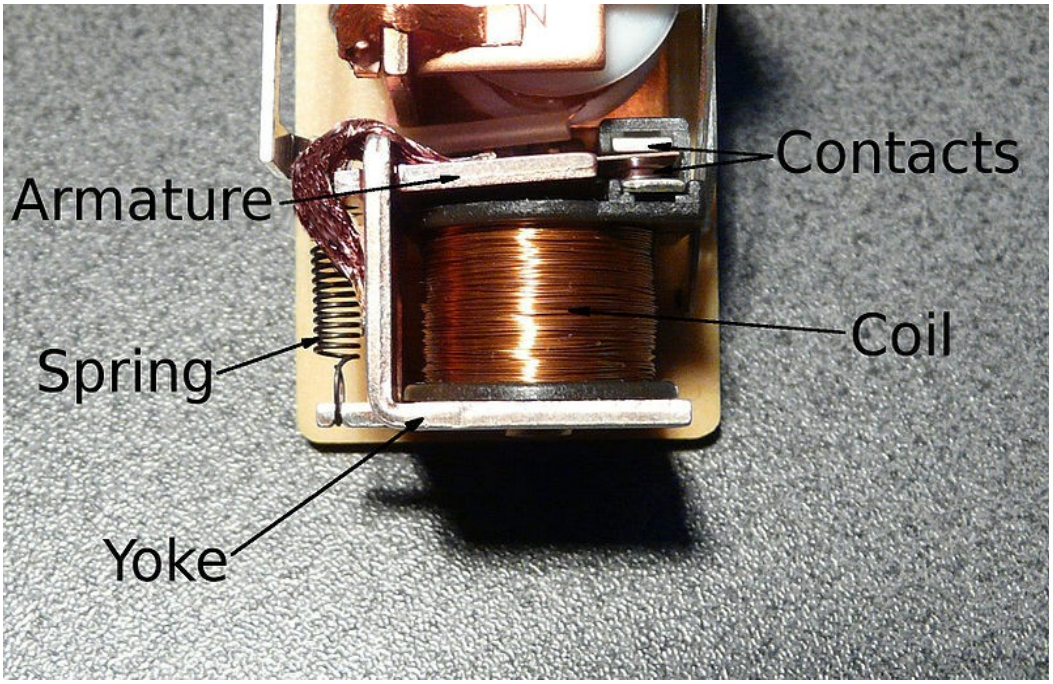
2) Remove the magnet. Now take a wire with alligator clips on either end. Stretch out a short, straight segment of the wire (pointing North) and tape it to a sheet of paper, leaving lots of slack at the ends. Touch one terminal of the battery with one end of the wire. Place the compass near the wire. Briefly touch the other terminal of the battery with the other end of the wire and draw the direction of the needle on the paper.

In what direction do the arrows point near the wire?

3) Finally, take two wires with alligator clips on the ends. Attach one end of each wire to each terminal (post) of a coil of copper wire. Bring the compass near one end of the coil. Create a circuit by touching each terminal of a battery with the free end of each wire. What do you observe and conclude?

So Mr. Oersted was onto something big here: electricity can generate magnetism!

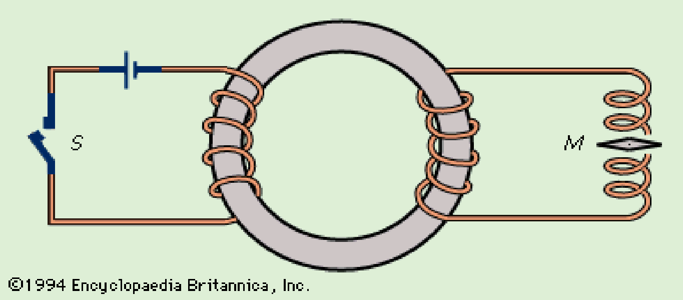
An early application that was important for switches in the first long-distance telegraphs, telephone exchanges, and digital computers is the **electromagnetic relay**:



An electric current in a wire coil moves a magnetic armature that closes a switch.

From Magnetism to Electricity

In 1831, Michael Faraday, a British physicist and chemist, discovered that moving a magnet through a coil of wire caused an electric current to flow in the wire. He also observed that a changing current in one coil induces a current in another nearby coil. These observations showed that a changing magnetic field generates an electric field!



©1994 Encyclopaedia Britannica, Inc.

A galvanometer is a sensitive instrument for measuring tiny electric currents in a circuit.

1) Using wires with alligator clips on the ends, connect each end of a coil of copper wire to one terminal of the galvanometer. You've just wired a simple circuit. Now watch this:

2) Keeping a close eye on the needle of the galvanometer, insert a magnet into the coil of copper wire. As you insert the magnet, what do you observe?

3) Now leave the magnet at rest in the coil. What does the galvanometer needle read?

4) Now remove the magnet from the coil, again watching the needle of the galvanometer. Does the needle move in the same direction or in the opposite direction?

So Mr. Faraday was right: magnetism can generate electricity! Hmm, could be useful.

An extremely important application of this principle is the electric generator.

Demonstrations of Magnetic Induction

Magnet and copper ring: Feel the force!

Magnet and wire coil connected to LEDs: See the light!

Building a Simple Electric Motor

With a AA battery, disk magnet, steel screw, and wire, we can make an electric motor:

1) Stick the disk-shaped magnet to the head of the screw and hang the tip of the screw from the positive terminal (bumpy end) of the battery.

Why doesn't the screw fall off?

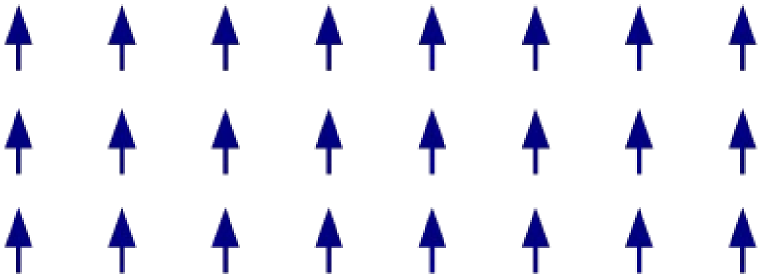
Why must the screw be made of steel?

2) Tape one end of the copper wire to the negative terminal (flat end) of the battery and touch the edge of the magnet with the other end of the wire.

What do you observe?

Ferromagnetic Materials

In ferromagnetic materials (Fe, Ni, Cr), the smallest magnets (atoms) try to be like their neighbors, so they support each other. They all "agree" on one direction and then stay that way. If we bring a strong magnet near, the tiny magnets turn to toward the strong magnet and will remain pointing in that direction. That's how we can store information.

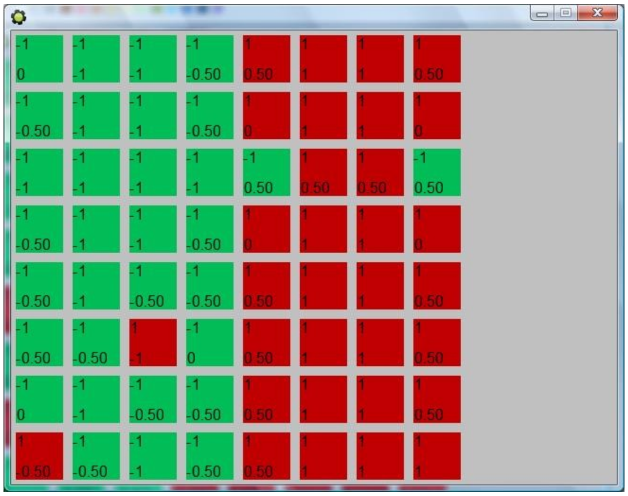


Not all materials can hold their magnetization. Diamagnetic and paramagnetic materials don't keep their orientation. Ferromagnetic materials can also lose their ability to store information when they are heated. They then become paramagnetic. Such a transition between different states of matter is called a phase transition. Have you heard the term "phase transition" before?

We can also simulate magnets with a computer (something we ask students to do).

As an illustration, let's look at a Game Maker simulation of the Ising model of a magnet.

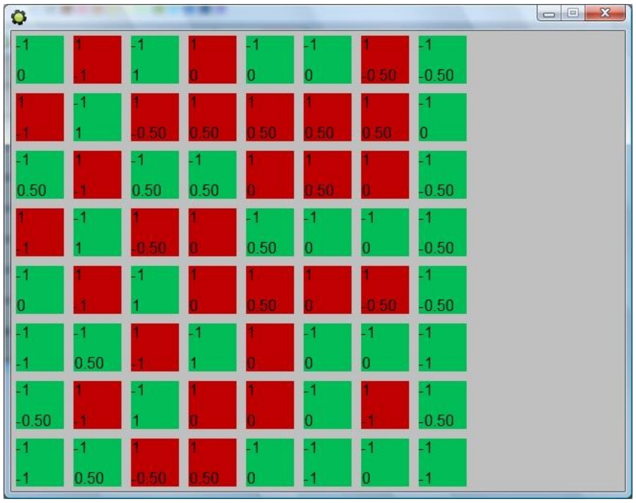
Below, green squares are magnets pointing up; red squares are magnets pointing down:



🟩 -1 0	🟩 -1 -1	🟩 -1 -1	🟩 -1 -0.50	🟥 1 0.50	🟥 1 1	🟥 1 1	🟥 1 0.50
🟩 -1 -0.50	🟩 -1 -1	🟩 -1 -1	🟩 -1 -0.50	🟥 1 0	🟥 1 1	🟥 1 1	🟥 1 0
🟩 -1 -1	🟩 -1 -1	🟩 -1 -1	🟩 -1 -1	🟩 -1 0.50	🟥 1 0.50	🟥 1 0.50	🟩 -1 0.50
🟩 -1 -0.50	🟩 -1 -1	🟩 -1 -1	🟩 -1 -0.50	🟥 1 0	🟥 1 1	🟥 1 1	🟥 1 0
🟩 -1 -0.50	🟩 -1 -1	🟩 -1 -0.50	🟩 -1 -0.50	🟥 1 0.50	🟥 1 1	🟥 1 1	🟥 1 0.50
🟩 -1 -0.50	🟩 -1 -0.50	🟥 1 -1	🟩 -1 0	🟥 1 0.50	🟥 1 1	🟥 1 1	🟥 1 0.50
🟩 -1 0	🟩 -1 -1	🟩 -1 -0.50	🟩 -1 -0.50	🟥 1 0.50	🟥 1 1	🟥 1 1	🟥 1 0.50
🟥 1 -0.50	🟩 -1 -0.50	🟩 -1 -1	🟩 -1 -0.50	🟥 1 0.50	🟥 1 1	🟥 1 1	🟥 1 1

Depending on interactions with its neighbors and external fields, each magnet may flip. At low temperatures (e.g., room temperature), the magnets tend to remain as they are.

At high temperatures, the magnets flip so easily that they don't hold their direction:



🟩 -1 0	🟥 1 -1	🟩 -1 1	🟥 1 0	🟩 -1 0	🟩 -1 0	🟥 1 -0.50	🟩 -1 -0.50
🟥 1 -1	🟩 -1 1	🟥 1 -0.50	🟥 1 0.50	🟥 1 0.50	🟥 1 0.50	🟥 1 0.50	🟩 -1 0
🟩 -1 0.50	🟥 1 -1	🟩 -1 0.50	🟩 -1 0.50	🟥 1 0	🟥 1 0.50	🟥 1 0	🟩 -1 -0.50
🟥 1 -1	🟩 -1 1	🟥 1 -0.50	🟥 1 0	🟩 -1 0.50	🟩 -1 0	🟩 -1 0	🟩 -1 -0.50
🟩 -1 0	🟥 1 -1	🟩 -1 1	🟥 1 0	🟥 1 0.50	🟥 1 0	🟥 1 -0.50	🟩 -1 -0.50
🟩 -1 -1	🟩 -1 0.50	🟥 1 -1	🟩 -1 1	🟥 1 0	🟩 -1 0	🟩 -1 0	🟩 -1 -1
🟩 -1 -0.50	🟥 1 -1	🟩 -1 1	🟥 1 0	🟥 1 0	🟩 -1 0	🟥 1 -1	🟩 -1 -0.50
🟩 -1 -1	🟩 -1 0.50	🟥 1 -0.50	🟥 1 0.50	🟩 -1 0	🟩 -1 -1	🟩 -1 0	🟩 -1 -1

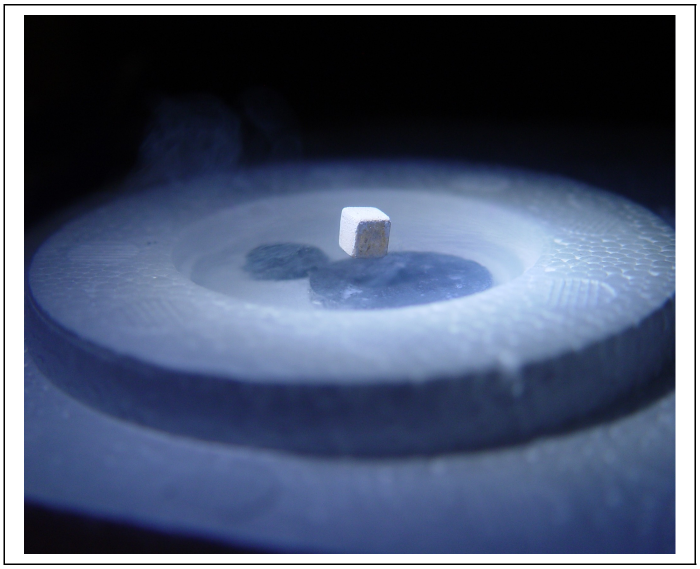
So you do not want to put your computer hard drive into the oven!

Superconductors

A superconductor is a remarkable material that conducts electricity with *no resistance*. The amazing property of superconductivity was discovered almost 100 years ago (1911) by a Dutch physicist, Heike Kamerling Onnes. Soon after first liquefying helium (1908), Kamerlingh Onnes discovered that many pure metals (e.g., mercury, tin, lead) become superconductors at extremely low temperatures, near the boiling point of liquid helium. The race was on to find materials that become superconducting at higher temperatures!

About 25 years ago, two German physicists, Johannes G. Bednorz and K. Alex Müller, discovered ceramic compounds that become superconductors at higher temperatures. Soon afterward, other researchers discovered related materials that are superconductors above the boiling point of liquid nitrogen (much cheaper than liquid helium!).

Demonstration: Superconductivity and magnetic levitation



A ceramic (YBCO) disk, cooled with liquid nitrogen, levitates a tiny (but strong) magnet.

Powerful superconducting magnets have important applications in magnetic resonance imaging (MRI), high-energy particle accelerators, and high-speed "Maglev" trains.

JR-Maglev train at Yamanashi, Japan test track in 2005 (581 km/h)

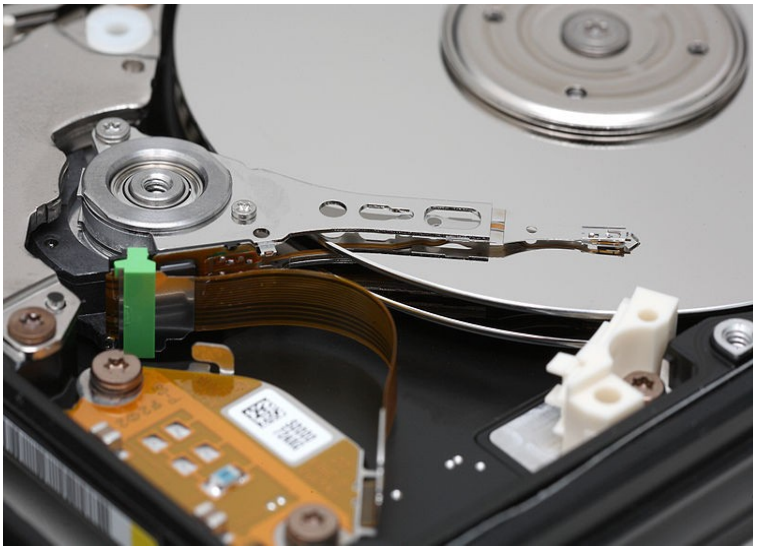
Activity 2: Building Logic Circuits

Magnets Store Information in Computers

Magnets are important for storing information (as binary numbers) in a computer. Magnetization can point either up or down and some materials will stay magnetized. Can you think of materials that can be permanently magnetized?

Information (bits) can be stored in the direction of the tiny magnets (up = 0, down = 1).

Modern computers use magnetism in hard drives: rotating magnetic disks on which many tiny points represent individual bits. The data are stored in magnets pointing up or down.

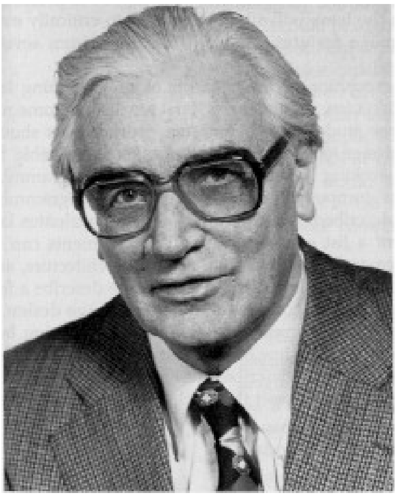


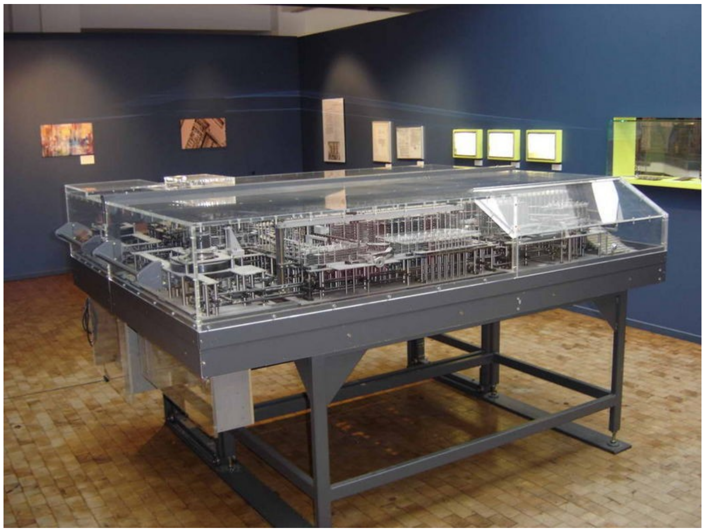
We won't get to simulating magnets today, but we will use the computer to practice counting with binary numbers some more (later, in Activity 3).

What is a computer? Can we build a computer now?

A computer uses bits (information) and processes them into other bits. The processing is nothing but switching and the instructions for doing the switching are themselves bits.

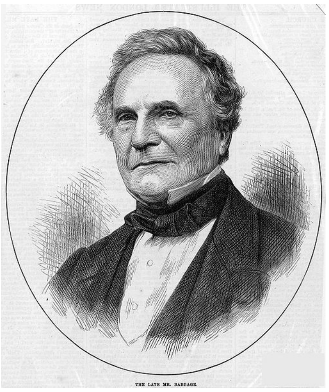
To do calculations we need switches. Can we build switches with magnets? Yes, with relays. The first programmable digital computer was built (1938-1941) by the German engineer, Konrad Zuse, using electromagnetic relays to store and process information.

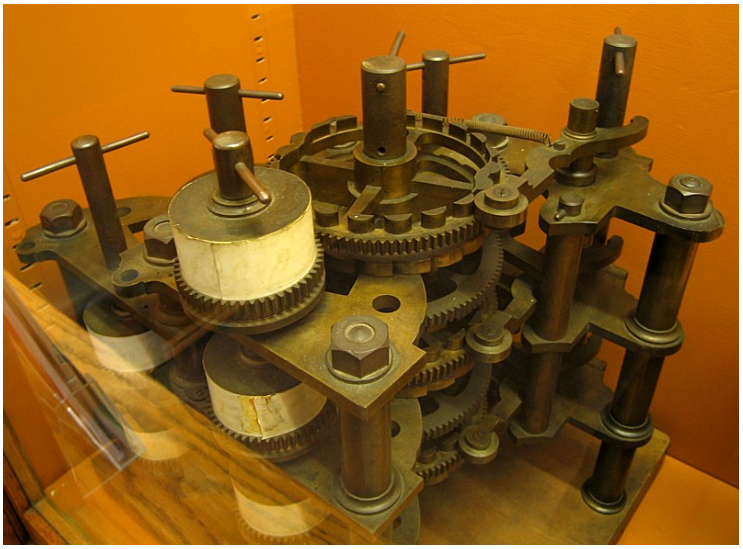




Konrad Zuse (1910-1995) and one of his first programmable digital computers.

The first programmable computer, invented in the 1820's by the English mathematician and engineer, Charles Babbage, was entirely mechanical!





Charles Babbage and his Difference Engine

Modern computers use other types of switches based on semiconductor materials. The physics of semiconductors is taught at NDSU in advanced courses (Solid State Physics). Some modern storage devices don't use magnets. CDs and DVDs use laser light and foil.

Still, everything is 0s and 1s. Who would have guessed that 150 years ago, when Charles Babbage first proposed a mechanical computer? Nobody believed his idea would work. Using electromagnetism people could make "computers" (analog computers) that could do much more complicated things: Using the properties of coils (and capacitors) they could calculate the slope of a curve => calculus. Ultimately the simple and flexible solution, using only 0s and 1s and magnets pointing up or down, succeeded.

Suppose that we have many little magnets that point either up or down:

N	N	S	S	S	S	N	N	N	N	S	S	N	S	N	N
S	S	N	N	N	N	S	S	S	S	N	N	S	N	S	S

How could we use the orientations of these magnets to store information? How could we store a number with a ferromagnet? Let's assume that we only want to distinguish between 0 and 1. How could we do that?

Define one number to have N up and one to have N down (the choice is a convention).

Let's assume that we define $\begin{array}{|c|}\hline \text{N}\\ \text{S}\\ \hline\end{array}$ to mean 0 and $\begin{array}{|c|}\hline \text{S}\\ \text{N}\\ \hline\end{array}$ to mean 1.

Now we have a way of storing 0 and 1, but how about 2?
To answer that question, think about familiar decimal numbers:
We can represent 9 as a digit, but how do we represent the next number?

Introduce a second digit that has a "place value" of 10.
For example, the number 14 is really 1*10 + 4*1.

What does the number 23 stand for?

Let's get back to the problem of storing 2 in a computer. If each digit can represent only two values, how many digits do you need to represent 0, 1, and 2?

The number 2 is represented as 10.

We call such numbers <u>binary</u> numbers. In binary numbers, each digit can have only two values, 0 or 1.

By the way, **<u>bi</u>**nary digi**<u>ts</u>** are also abbreviated as **<u>bits</u>**.

Joke: “There are only 10 types of people – those who understand binary numbers and those who don't.”

What does the binary number 11 stand for?

So now we can count to 3:

Binary Number	Decimal Number
00	0
01	1
10	2
11	3

How do we represent 4?

Think again about decimal numbers: What do you do when you want to represent numbers greater than 99? We need yet another digit.

That allows us to keep counting:

Binary Number	Decimal Number
100	4
101	5
110	6
111	7

So far we used 3 bits. How many different numbers were we able to represent (0,…,7)?

How many with 5 bits?

Keep going to get to 8 bits: $2^8 = 2*2*2*2*2*2*2*2 = 256$

8 bits is also called 1 byte.
1KB is 2^{10} = 1024 bytes, or about 1000
1MB is 2^{20} (about 1 million) bytes
1GB is about 1 billion bytes

Can you do calculations with such numbers? How do you add two binary digits (bits)?

Let's do some additions:

0 + 0 =
0 + 1 =
1 + 0 =
1 + 1 =

How can we build a computer that can do this calculation?

Since we only have to worry about 0 and 1 we can use simple on / off switches.

First we think of these as mechanical switches. In order to build a computer, we would have to be able to use electric current to do the switching. Can we use electric current to create a magnetic field?

A switch that uses an electromagnet to do the switching is called a relay. Relays are still used in simple switches like door bells. But they are quite big, so they are no longer used in computers. We will use a relay in the electronics lab.

So how do relays help us understand a computer: With that magnetic field we can turn a switch on and off (remotely), and that switch results in a bit that can do more switching. That's just what a digital computer does: a lot of switching done on a lot of bits, and the switching is itself encoded as bits!

Let's see if we can get an idea of how that works, by looking at the addition of 2 bits. What do we need: We have two input bits and two output bits:

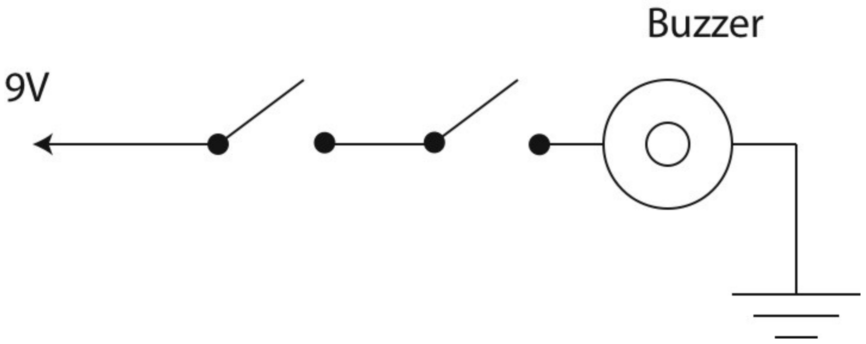
0 + 0 = 00
0 + 1 = 01
1 + 0 = 01
1 + 1 = 10
How can we produce the first output bit based on the input bits using switches?

On a bread board (Radio Shack Electronic Sensors Lab), we will build electric circuits with a battery, switches, and a buzzer.

Note: these circuits take input (current from the battery) and perform logical operations (decisions) that determine the output.

Here are some examples of circuits:

Series Circuit:



What do we expect for a Series Circuit?

Let's write out the different alternatives:

First Switch	Second Switch	Buzzer
off	off	
off	on	

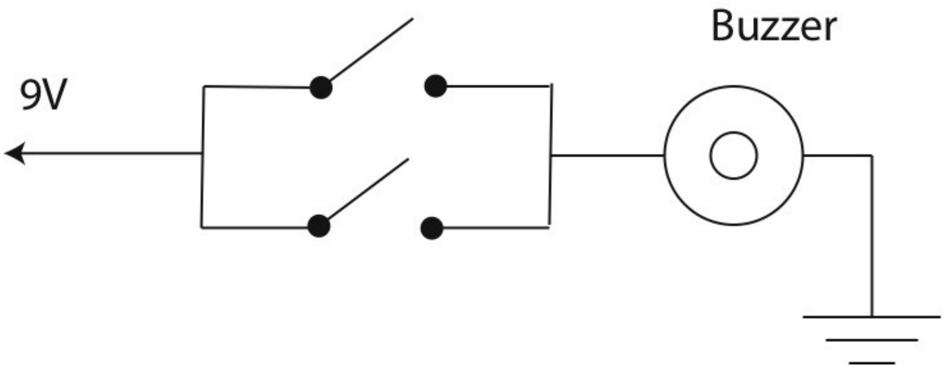
This logic operation is called "AND" (a type of logic gate). Why?

By the way, switches that are on or off can be written as 0 and 1, just like the magnets that were up or down. Also, we can consider the switches to be inputs and the buzzer to be an output. So let's see how the input-output table looks now:

First Input	Second Input	Output
0	0	0
0	1	0
1	0	0
1	1	1

Another interesting and useful circuit is the **Parallel Circuit**.

Parallel Circuit:



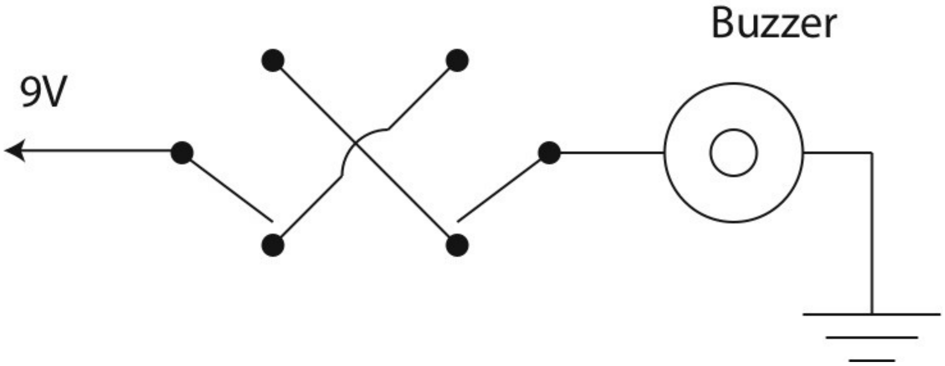
What do we expect for a Parallel Circuit?

First Input	Second Input	Output
0	0	
0	1	
1	0	
1	1	

This logic operation is called "OR" (another type of logic gate). Why?

Finally, we can build an **Alternating Circuit**: This circuit is like a lamp with switches at two sides of the room: You can turn on the lamp with either switch, but if you use both switches the lamp is off again.

Alternating Circuit:



What does the input-output table look like?

First Input	Second Input	Output
0	0	
0	1	
1	0	
1	1	

This logic operation is called "exclusive OR" or "XOR". Why is this name appropriate?

When parents ask a child, "Would you like candy or ice cream?", which of the logical possibilities do they usually mean (OR or XOR)? Hint: Do they consider "candy and ice cream" to be a reasonable answer?

Do any of these circuits help us with adding two bits? Let’s remember what we need:

First Input	Second Input	First Output (place value 2)	Second Output (place value 1)
0	0		
0	1		
1	0		
1	1		

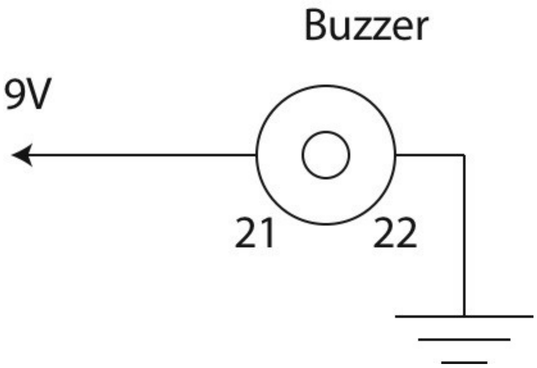
Which of the circuits gets us the First Output (i.e., leftmost bit, with place value 2)?

Which of the circuits gets us the Second Output (i.e., rightmost bit, with place value 1)?

Now let's use the Electronics Lab to make logic gates, which perform logical operations.

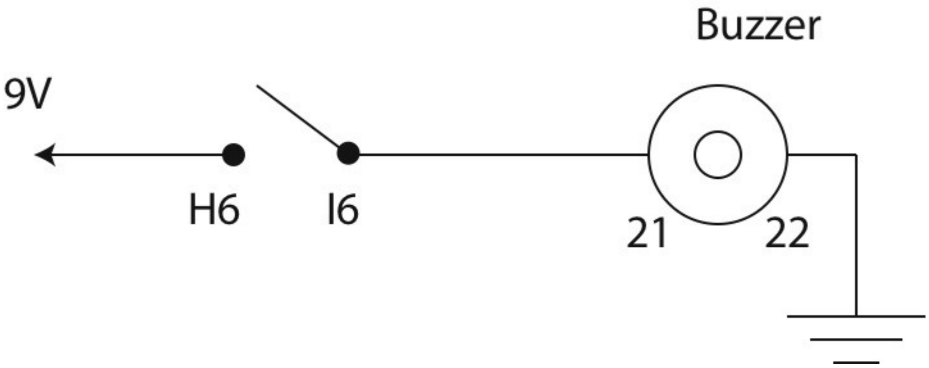
We'll use switches and a relay to build Series, Parallel, and Alternating Circuits.

First of all, we simply test the buzzer:



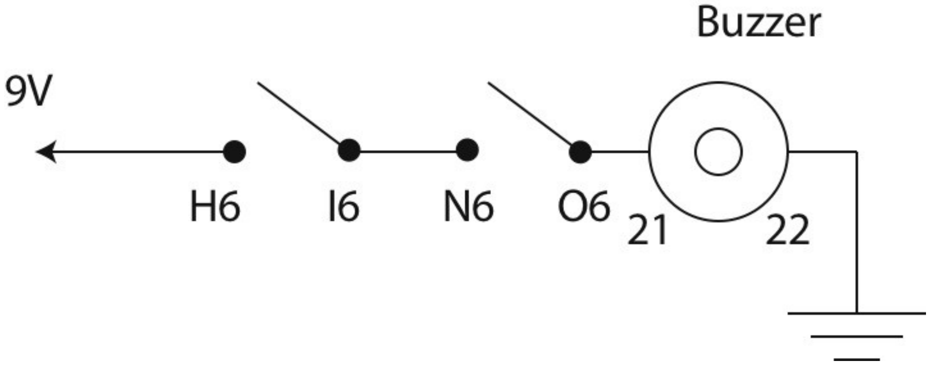
a. Push the power switch to OFF.
b. Connect spring 21 to + voltage (blue wire).
c. Connect spring 22 to ground (blue wire).
d. Push the power switch to ON.

Next, we use one of the manual switches to control the buzzer:



a. Push the power switch to OFF.
b. Insert a switch into H10, I10, J10, and push the switch down (towards you).
c. Connect + voltage to H6 (white wire).
d. Connect spring 21 to I6 (blue wire).
e. Connect spring 22 to ground (blue wire).
f. Push the power switch to ON.
g. Push the switch up to turn on the buzzer.

Now we combine two manual switches into a **Series Circuit**:

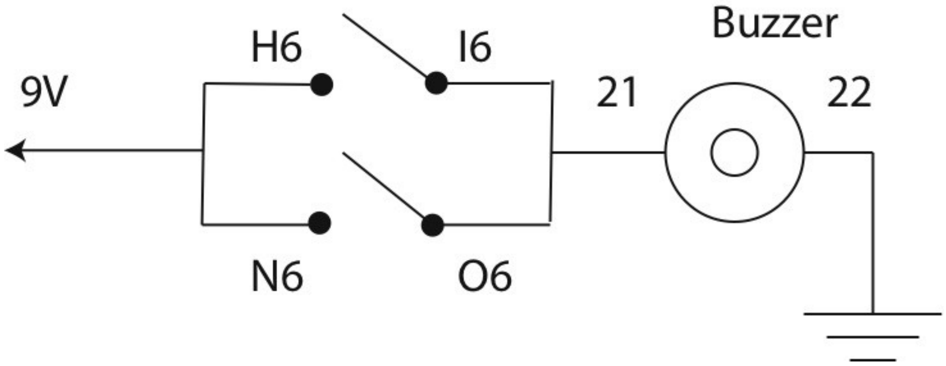


a. Push the power switch to OFF.
b. Insert a switch into H10, I10, J10.
c. Insert a second switch into N10, O10, P10.
d. Connect + voltage to H6 (white wire).
e. Connect I6 to N6 (white wire).
f. Connect spring 21 to O6 (blue wire).
g. Connect spring 22 to ground (blue wire).
h. Push the power switch to ON.

For which of the four combinations of switch positions does the buzzer sound?

What does the "series" circuit represent? AND, OR, or XOR?

With a bit of rewiring, we can build a **Parallel Circuit**:



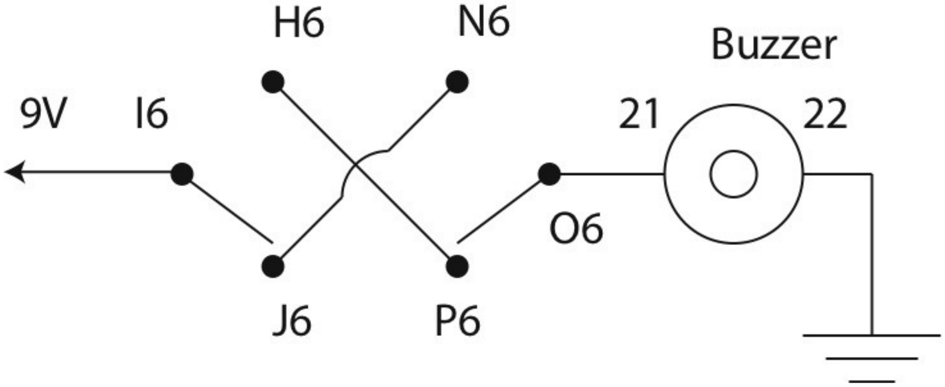
a. Push the power switch to OFF.
b. Insert a switch into H10, I10, J10.
c. Insert a second switch into N10, O10, P10.
d. Connect + voltage to H6 and N6 (white wires).
e. Connect I6 to spring 21 (blue wire).
f. Connect O6 to spring 21 (blue wire).
g. Connect spring 22 to ground (blue wire).
h. Push the power switch to ON.

For which of the four combinations of switch positions does the buzzer sound?

What does the "parallel" circuit represent? AND, OR, or XOR?

Does this help us for adding binary numbers?

With a bit more rewiring, we can build an **Alternating Circuit**:



a. Push the power switch to OFF.
b. Insert a switch into H10, I10, J10.
c. Insert a second switch into N10, O10, P10.
d. Connect + voltage to I6 (red wire).
e. Connect H6 to P6 (white wire).
f. Connect J6 to N6 (white wire).
g. Connect O6 to spring 21 (blue wire).
h. Connect spring 22 to ground (blue wire).
i. Push the power switch to ON.

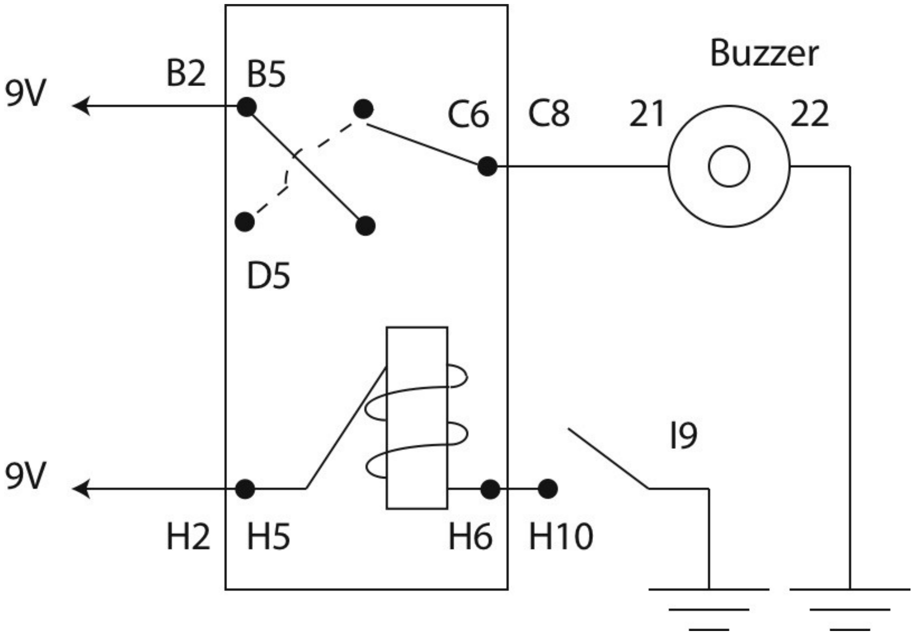
For which of the four combinations of switch positions does the buzzer sound?

What does the "alternating" circuit represent? AND, OR, or XOR? □

Does this help us for adding binary numbers?

Now we know how to get the two bits that we need for adding binary numbers! (It gets a bit more complicated when the binary numbers have more than one bit, but not much. All that has to change is that we have to treat the leftmost bit as a carry bit and add it to the other two bits.)

Finally, let's play with a **relay**. An electromagnetic relay uses electricity and magnetism to move a switch (often remotely)! You can build a whole computer based on relays!



a. Push the power switch to OFF.
b. Insert a switch into H10, I10, J10.
c. Hold a relay so that the short side with two pins is toward the bottom of the bread board. Insert the two pins on the bottom into H5 and H6. This allows the top three pins to fit into B5, D5, and C6.
d. Connect + voltage to B2 (white wire).
e. Connect + voltage to H2 (red wire).
f. Connect I9 to ground (red wire).
g. Connect spring 21 to C8 (blue wire).
h. Connect spring 22 to ground (blue wire).
i. Push the power switch to ON.

Note: When you move the manual switch, you can hear the relay switch! If you want to hear the relay switch more clearly, you can try disconnecting the buzzer.

Summary

We now know how computers store information as binary numbers and how to do the basic operations necessary for working with binary numbers in a computer. To do these operations, we used switches, including a relay – a switch that is flipped by a magnet.

Questions

What are the numbers 1 through 5 in the binary system?

Can you only represent numbers in a computer or also letters?

What logic operation does a serial circuit produce?

What logic operation does a parallel circuit produce?

What logic operation does an alternating circuit produce?

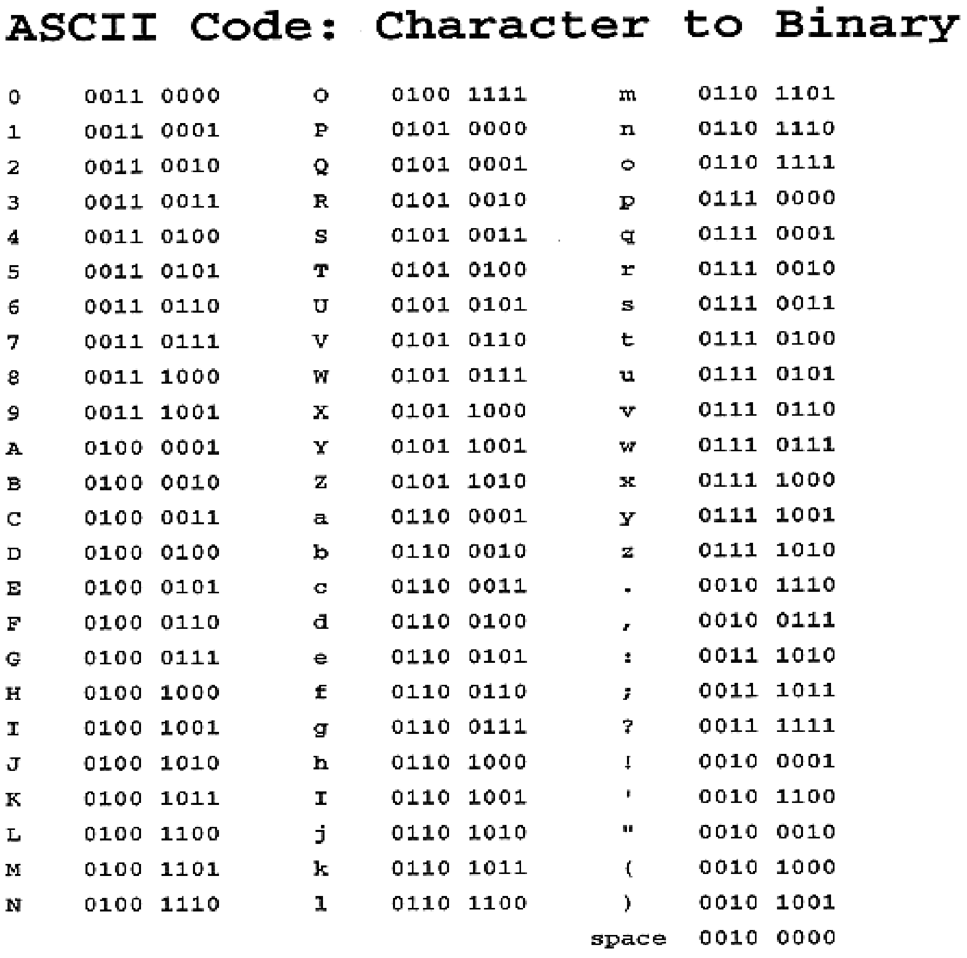
What logic operation gives the rightmost bit when adding two bits?

What logic operation gives the leftmost bit when adding two bits?

Activity 3: Computer Concepts

From Binary Numbers to Computer Games

Let's now look at a computer in action. What can be done with a computer? We have seen how to represent numbers, but can we represent letters? We sure can, but the way to encode them is somewhat arbitrary. Below is the most common encoding for letters:



ASCII Code: Character to Binary

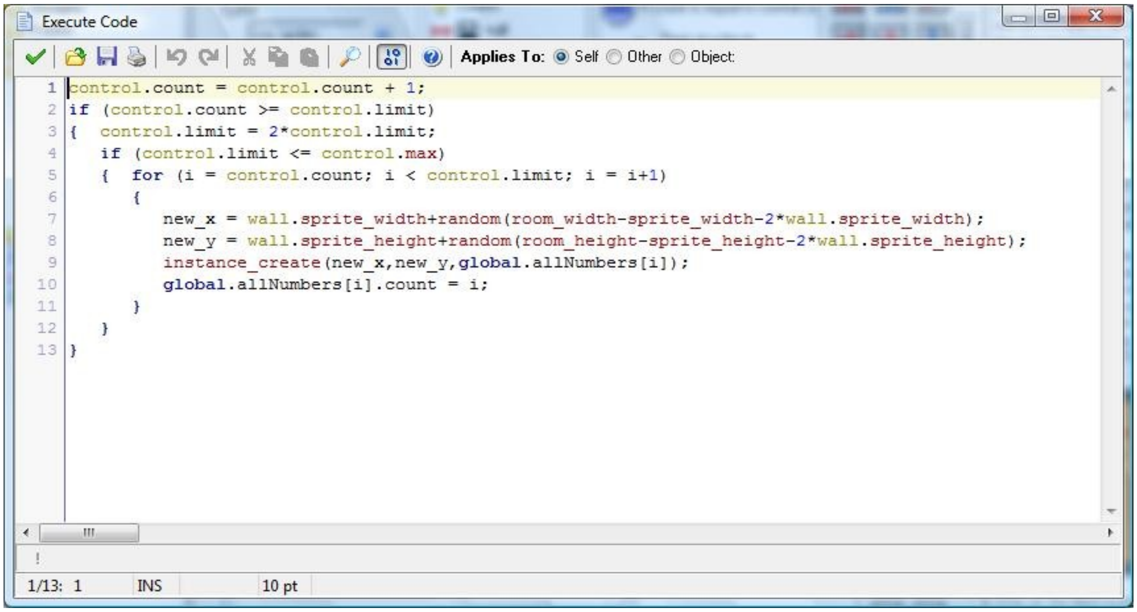
0	0011 0000	O	0100 1111	m	0110 1101
1	0011 0001	P	0101 0000	n	0110 1110
2	0011 0010	Q	0101 0001	o	0110 1111
3	0011 0011	R	0101 0010	p	0111 0000
4	0011 0100	S	0101 0011	q	0111 0001
5	0011 0101	T	0101 0100	r	0111 0010
6	0011 0110	U	0101 0101	s	0111 0011
7	0011 0111	V	0101 0110	t	0111 0100
8	0011 1000	W	0101 0111	u	0111 0101
9	0011 1001	X	0101 1000	v	0111 0110
A	0100 0001	Y	0101 1001	w	0111 0111
B	0100 0010	Z	0101 1010	x	0111 1000
C	0100 0011	a	0110 0001	y	0111 1001
D	0100 0100	b	0110 0010	z	0111 1010
E	0100 0101	c	0110 0011	.	0010 1110
F	0100 0110	d	0110 0100	,	0010 0111
G	0100 0111	e	0110 0101	:	0011 1010
H	0100 1000	f	0110 0110	;	0011 1011
I	0100 1001	g	0110 0111	?	0011 1111
J	0100 1010	h	0110 1000	!	0010 0001
K	0100 1011	I	0110 1001	'	0010 1100
L	0100 1100	j	0110 1010	"	0010 0010
M	0100 1101	k	0110 1011	(	0010 1000
N	0100 1110	l	0110 1100	)	0010 1001
				space	0010 0000

For the first computers, people had to write very detailed instructions. The instructions themselves were also written as a code, called machine code or machine language. In the figure below, left, machine code is written as numbers (hexadecimal numbers, where 4 bits are combined into numbers, 0, 1, 2, … 8, 9, A, B, C, D, E, F instead of 0000, 0001, … 1000, 1001, 1010, 1011, 1100, 1101, 1110, 1111). In the middle of the figure, the numbers are abbreviated by words that are supposed to be understandable, such as ADD or STOP. On the right side there is some explanation of what is done.

```
00000400                    ORG      $400     ;Start of the program

00000400 303900000500       MOVE.W   X,D0     ;Put the value of X in D0
00000406 C0C0               MULU     D0,D0    ;Calculate X2
00000408 323900000502       MOVE.W   Y,D1     ;Put the value of Y in D1
0000040E C2C1               MULU     D1,D1    ;Calculate Y2
00000410 D280               ADD.L    D0,D1    ;Add X2 to Y2 and put result in D1
00000412 343900000500       MOVE.W   X,D2     ;Put the value of X in D2
00000418 947900000502       SUB.W    Y,D2     ;Subtract Y from D2 to get X - Y
0000041E 82C2               DIVU     D2,D1    ;Divide D1 by D2 to get (X2 + Y2)/(X - Y)
00000420 33C100000504       MOVE.W   D1,Z     ;Put the result now in D1 into Z
00000426 4E722700           STOP     #$2700

00000500                    ORG      $500     ;Put the data here

00000500 0032           X:  DC.W     50       ;Initial dummy value for X
00000502 000C           Y:  DC.W     12       ;Initial dummy value for Y
00000504 00000002       Z:  DS.W     1        ;Reserve space for the result

         00000400           END      $400     ;End of program and address of entry
```

Machine and assembly languages (http://www.alanclements.co.uk/BCSweb/assembly.htm)

People soon tired of writing such difficult to understand programs, so they developed languages that focus on what we want to do, rather than on what the computer can do. We call these "high-level" languages. Below you see what such code can look like:



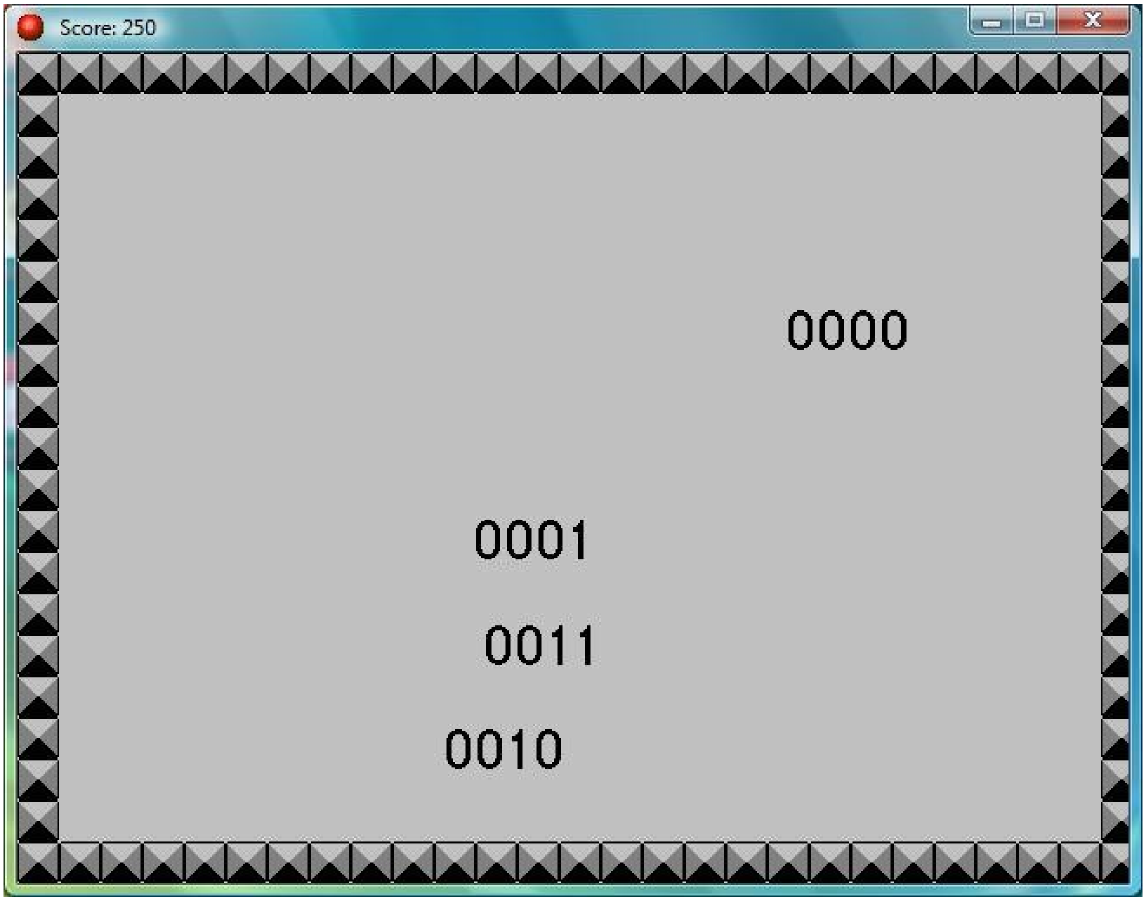
Applies To: ● Self ○ Other ○ Object:

```
control.count = control.count + 1;
if (control.count >= control.limit)
{   control.limit = 2*control.limit;
    if (control.limit <= control.max)
    {   for (i = control.count; i < control.limit; i = i+1)
        {
            new_x = wall.sprite_width+random(room_width-sprite_width-2*wall.sprite_width);
            new_y = wall.sprite_height+random(room_height-sprite_height-2*wall.sprite_height);
            instance_create(new_x,new_y,global.allNumbers[i]);
            global.allNumbers[i].count = i;
        }
    }
}
```

The magnet simulation and the counting game (below) were written in such a high-level language, which is called Game Maker. This language was designed to develop small games. The language can be programmed using pictures and we will try that out soon. You can also write more complicated programs in Game Maker. In fact, the code in the previous figure was taken from the counting game (but don't worry, you won't have to write such code in this activity).

You probably have the Game Maker software installed on your computer. If not, point your browser to www.yoyogames.com and follow the links to install Game Maker 8.

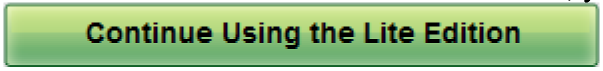
You should now open the game "BinaryCounting". The objective of the game is to click on the numbers in the right order, starting with 0000, going on to 0001, and so on. When you are done with one set of numbers, the next set will appear, until you get to 1111.



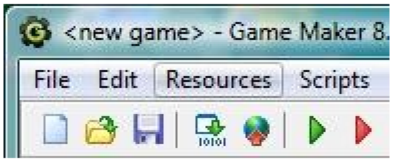
Programming the Computer

Now let's program a little game ourselves. The game we program will be like the binary counting game, but with only one ball. When the ball is clicked (if we're fast enough!), it will jump to some other place but will keep moving. OK, let's get started …

When the Game Maker software is installed, you can start it by choosing



From the **Resources** menu (shown below), choose **Create Sprite:**

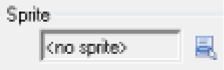


Change **Name** to spr_ball.
Click **Load Sprite.**
Select **Bouncing Balls** and select any one you like.
Click **Open** and then **OK.**

From the **Resources** menu, choose **Create Sprite.**
Change **Name** to spr_wall.
Click **Load Sprite** (you may then have to click the green arrow up to get back out of the Bouncing Balls directory).
Select **Maze Platform**, scroll down to sprites starting with "w" and pick any wall.
Click **OK.**

From the **Resources** menu, choose **Create Sound.**
Change **Name** to sound_success.
Click **Load Sound .**
Select any sound you wish.
Click **OK.**

From the **Resources** menu, choose **Create Object.**
Change **Name** to obj_wall.

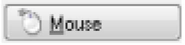
Click on the menu icon at the end of the sprite field 

and select the sprite spr_wall.
Select the solid property.
Click **OK.**

From the **Resources** menu, choose **Create Object.**
Change **Name** to obj_ball.
Click on the menu icon at the end of the sprite field and select the sprite spr_ball.
Press **Add Event** button.
Click on the **Create**

button.
Drag **Move Fixed** into list of actions for this event.
Select all 8 directions (not the middle one); a direction will be chosen randomly.
Set **Speed** to 3.
Click **OK** to indicate we're done with this action (don't yet press **OK** for object).

Press **Add Event** button again.
Click on the **Collision**

button and select obj_wall.
Drag a **Bounce** action into list of actions for this event.
(Default values are fine for this action)
Click **OK** to indicate we are done with the **Bounce** action.
Press **Add Event** button again.
Click on the **Mouse** 

button and select **Left Pressed.**

From the tab **main1** main1, drag a **Sound** action into list of actions for this event and select sound_success. Click **OK** to indicate we're done with this action.

From the tab **move** move, drag a **Jump to Random** action into list of actions for this event (leave default parameters). Click **OK** to indicate we're done with this action.

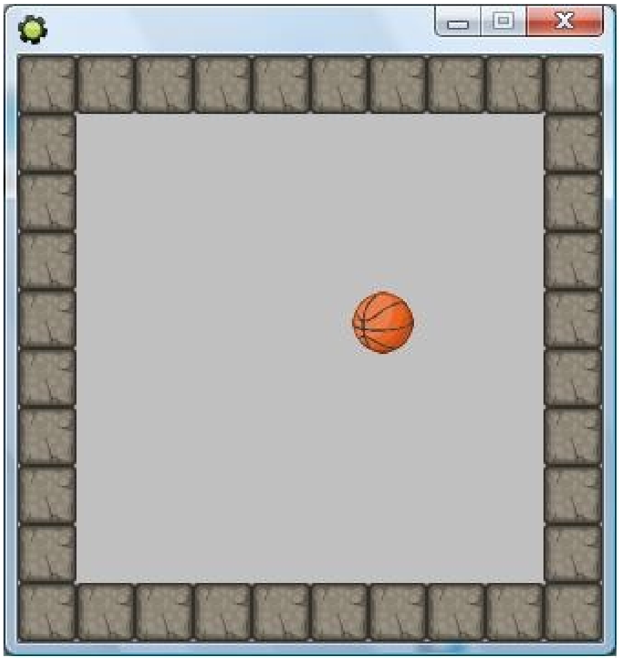
Also from tab move, drag a **Move Fixed** action and do the same things as before: Select all 8 directions and set speed to 3.
Click **OK** to indicate we are done with the action.

Click **OK** to indicate we are done with the object ball.

From the **Resources** menu, choose **Create Room.**
Top left (objects | settings | backgrounds) click the **Settings** tab; set **Width** and **Height** both to 320.
Also at the top Snap X: 16 | Snap Y: 16 set **SnapX** and **SnapY** both to 32.
Click on the **Objects** tab; the ball should be selected.
Click anywhere in the gray area to place one ball on the field.
Click on the menu next where you see the text "obj_ball" and select "obj_wall".
Click on all the border fields to place wall pieces there (if you hold down shift you can drag the mouse to place the wall pieces). Right click to erase.
Click **OK**

Save the game by clicking **File** and then **Save** in the top left corner.
Run the game by clicking the green triangle in the top middle.

When you are done, your program should look something like the figure below, although the ball and wall will probably look different.



Have fun trying to catch the ball by clicking on it with the mouse! You should hear the sound you selected whenever you catch it. To stop the game, push the esc (escape) key.

Note that this was a simplified version of the Tutorial "First Game", which is available from the Yoyo Games website. If you would like to continue with Game Maker, you may want to look at that "First Game" tutorial.

Questions

Which parts of a modern computer use magnets? Hard disks and/or switches?

Can letters of the alphabet be represented in a computer?

What happens to magnetic storage when it is heated too much?

What do we call it when a magnet goes from being ferromagnetic to being paramagnetic?

Summary and Wrap-Up

Magnetism is a fascinating phenomenon with many important applications.

Ferromagnetic materials can store information.

One can even use magnets to switch electricity, which allows us to process information.

A computer stores information as bits (binary digits) and processes them into other bits. Processing is nothing but switching and the instructions for switching are themselves bits.

Bits can represent numbers or encode letters and other characters.

We can add (and subtract) bits by using electronic circuits with logic gates.

Although computers only use very simple concepts – bits and switches – we can program them to do some amazing things.