Introduction to Digital Electronics

1. How are computers made?

We started by showing the stuff that computers are made of. It all begins with common sand, which consists mostly of silicon dioxide (quartz). Using chemical methods, the sand is converted to pure silicon. Very pure silicon, 99.999 999% -- you can't get anything more pure.
Pure silicon is a funny material. It shines like a metal, but is breakable like a ceramic. It is a semiconductor. That means it is on the edge: does it conduct electricity or doesn't it? Well, we can make it do both: make it conduct, or make it stop conducting.We can switch an electrical current in silicon on or off, at will, and very, very fast. From silicon, we make fast switches!
A whole bunch of those switches together make a chip, which is put inside a plastic cover.
A bunch of chips are mounted on a printed circuit board.
A bunch of boards make an electronic box: a VCR, a TV, a radio, a computer.
Well, of course you need more stuff, like a power supply, a display, a hard drive, and a box to fit it all in. But the heart of anything electronic is those silicon switches.

2. What do computers do?

We use computers for a lot of things. Playing games, writing book reports, calculating math problems... It actually all started with math problems.

- So these boxes can calculate quite well. Very well.
- We do know that they do what they are told. You push a button, and the computer does it. It does exactly what you *tell* it to do (which is not necessarily what you *meant* it to do...). It follows instructions.
- Computers move information, for example your book report from the disk to the printer. Or a file from the Internet to your display screen, or to your own hard disk. They store information (all the book reports you have written are stored on the hard disk), and they manage it (you can find it again).

How do those silicon switches we talked about actually make all this happen?

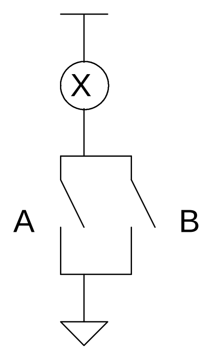
This class is going to explore just that: how we can do cool things, such as writing text, making pictures and calculating with switches. Just like computers do.
This is called switch logic, or Boolean logic, after George Boole (English mathematician, 1815-1864), who was the first to think of it -- long before electronics existed!

3. Switch logic: OR and AND

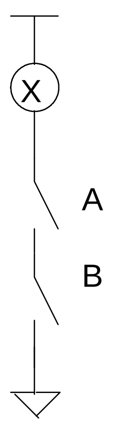
The following diagrams are the first circuits we explore: an OR circuit and an AND circuit. While the diagrams are drawn as if these are electrical circuits (a light goes on if the correct switches are closed), we actually demonstrated these with a water circuit, consisting of a reservoir on top, a catch basin at the bottom, tubes instead of wires, and valves for switches. George Boole's switch logic works for water just as it does for electricity!

For the diagrams below, imagine a teacher asking a question. If they think the answer is "Yes", then the students press a switch on their desk.
In the OR circuit, if either Annie ***or*** Bert say "Yes" (press their switch) then the light goes on. (It also works if they both press their switch).
In the AND circuit, both Annie ***and*** Bert have to agree that the answer is YES, and both have to press their switches for the light to go on.

OR -- either Annie or Bert (or both) say YES



AND -- both Annie and Bert say YES



A allows flow?	B allows flow?	Flow? (light on?)
N	N	N
N	Y	Y
Y	N	Y
Y	Y	Y

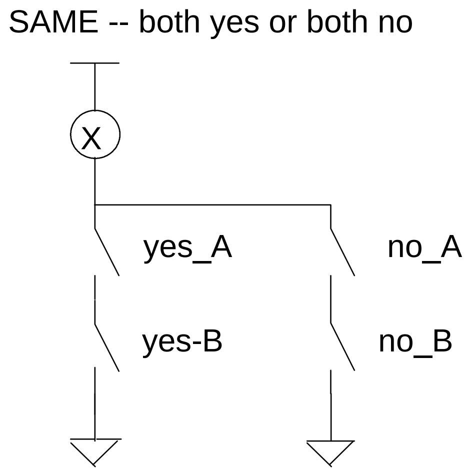
A allows flow?	B allows flow?	Flow? (light on?)
N	N	N
N	Y	N
Y	N	N
Y	Y	Y

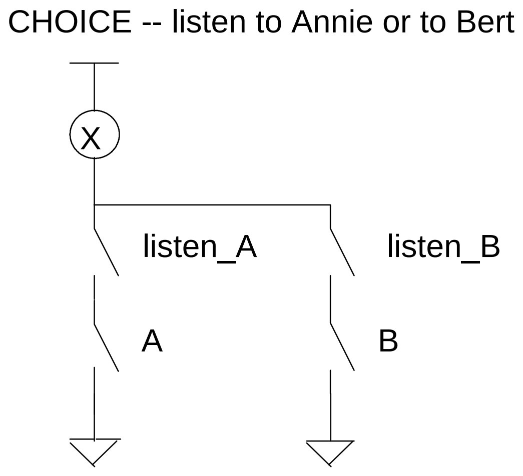
Examples of OR circuits:
- Electric car windows: either the driver **or** a passenger can open the passenger's side window.
- Interior light in a car: goes on when any door is opened (driver's side door **or** passenger's side door **or** driver's side back door **or** ...). But you have to think a little: when the car door is *opened*, an electrical switch (usually in the door jamb) is actually *closed* to allow the current to flow to light up the lamp. Can you find those light switches on your car?

Example of AND circuit:

- Again the electric car window: a master switch (at the driver's seat) has to be on **and** a local switch needs to be pressed for the window to operate.

Slightly more complicated switch logic:





The "SAME" circuit above is a combination of AND and OR. The effect is that the light will be on if Annie and Bert both say "yes", or both say "no" to a teacher's question, in other words, if they both agree. The logic is: *(yes_A* **AND** *yes_B)* **OR** *(no_A* **AND** *no_B).*
This is also known as an "equivalence" circuit.
If the yes_A and no_A switches were reversed, then this would be a "NOT-THE-SAME" circuit *(no_A* **AND** *yes_B)* **OR** *(yes_A* **AND** *no_B).* Such a circuit is also known as an "exclusive OR": The light goes on if Annie OR Bert says yes, but not both (they disagree -- just as you would expect from Annie and Bert!). An abbreviation for "exclusive or" is XOR, and it turns out that this is a very handy circuit to have around.
"Same" or "not-the-same" circuits are used wherever a computer makes comparisons.
Another example of a "SAME" circuit (though wired differently): hall lights in a house with an upstairs and downstairs switch.

The other circuit, above on the right, looks very similar. The difference is who controls the switches. In the "CHOICE" circuit, the top two switches are controlled by the teacher, who makes to choice to listen to Annie or to listen to Bert. If the teacher switches on Annie, then Annie can answer a question by pressing her switch, and Bert's answer is ignored. And the other way round.
The fancy name for such a circuit is a "multiplexer", and it is used wherever a choice is made in a logic circuit.

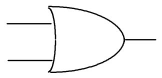
With AND and OR gates, almost any logic can be made. The missing element is a NOT.
For example, in the choice circuit above, a NOT circuit could be connected between the Listen_A and listen_B switches, so that at any time one is closed, and the other is open (thus, the teacher listens to Annie or to Bert, but cannot switch them both off).
With just AND, OR and NOT, any logic circuit can be made!

Look Ma, no fingers...
How does this work in electronics, without lots of fingers to push lots of switches?
Well, the switches are special. They are called transistors, and work with electrical voltages acting as the “fingers” pushing them open or closed.
As if, in the circuits above, the light from the lamp would push on a switch in the next circuit, and that one on the next, and the next.... Only, no lamps either. Only voltages and currents.
Exactly how all that works does not matter, though. Switch logic works as well with water valves as it does with finger-actuated switches, as it does with transistors.
(It is just that transistors are faster.... much, much faster...).

4. Play on floor with simple gates

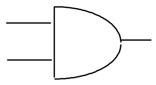
Example tasks: AND, OR, XOR (not-the-same), via truth tables.
Above, we showed tables for AND and OR circuits that indicated electrical current (or water) flow. More customary is to express the logic in 0 and 1, where 0 may stand for “no flow”, or for “low voltage” or “false” or “no” and where 1 may stand for “flow”, or for “high voltage” or “true” or “yes”, respectively. That leads to the following “truth tables”:

OR



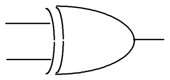
A	B	Out
0	0	0
0	1	1
1	0	1
1	1	1

AND



A	B	Out
0	0	0
0	1	0
1	0	0
1	1	1

***XOR* --not the same--**



A	B	Out
0	0	0
0	1	1
1	0	1
1	1	0

5. So you have switches. Now what?

A single switch can only represent “yes-no”, “true-false”, “1-0” (because that is the least writing...). But a bunch of switches can represent anything you want...

Numbers

Binary	***Decimal***
“000”	0
“001”	1
“010”	2
“011”	3
“100”	4
“101”	5
“110”	6
“111”	7

Text

Binary	***Character***
“0100 0000”	@
“0100 0001”	A
“0100 0010”	B
“0100 0011”	C
“0100 0100”	D
“0100 0101”	E
“0100 0110”	F
“0100 0111”	G

Colors

Binary	***Color***	
“0 0 0”	Black	⬛
“1 0 0”	Red	🟥
“0 1 0”	Green	🟩
“0 0 1”	Blue	🟦
“0 1 1”	Cyan	🟦
“1 0 1”	Magenta	🟪
“1 1 0”	Yellow	🟨
“1 1 1”	White	

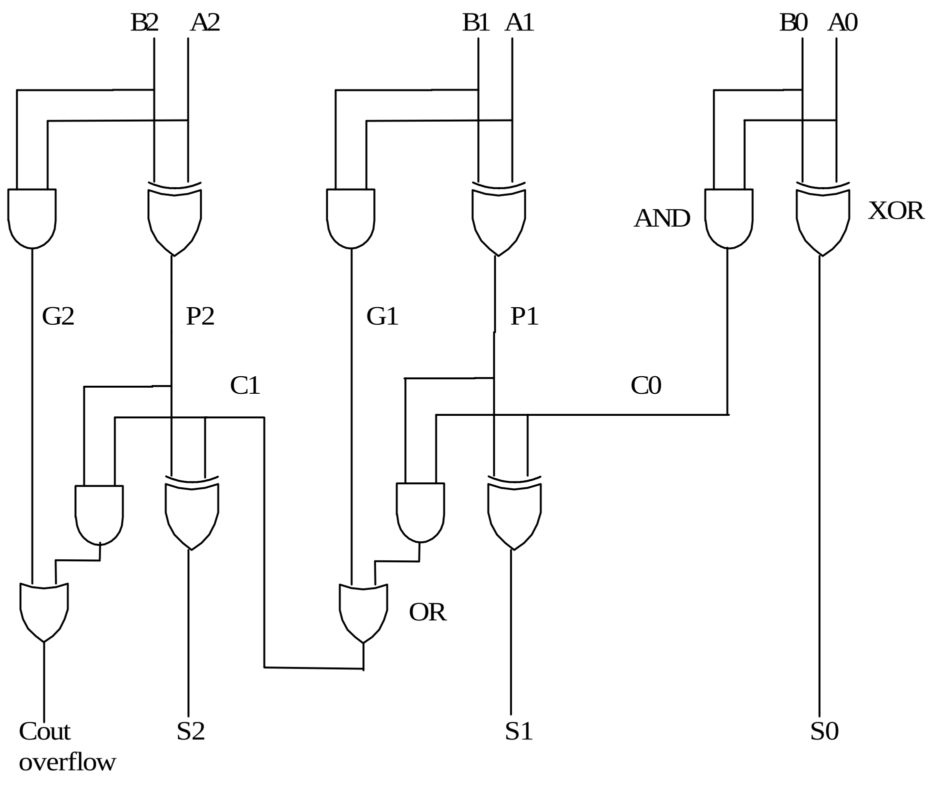
So, dependent on what you tell the computer to do with it, a bunch of 0s and 1s can be used to represent text, so that you can use it to type up a book report....

Or a colored pixel on your video game display...

Or it can represent a number, so that you can tell the computer to calculate something.

How does it calculate with only 0s and 1s? Well, that’s a neat trick...

6. Schematic of a 3-bit adder -- on the floor .



We laid out the above circuit schematic on the floor, except that each gate symbol was a box, with a student doing the appropriate operation: AND, OR or XOR (not-the-same), and 0/1 values being indicated by a red/green 0/1 marker on each wire.

This circuit represents a 3-bit adder, with which additions can be done from the binary numbers table in the previous section. Only additions with the sum less than 15 can be done.

Observations:

- See the glitch in this “ripple carry” adder propagate through: S2 and Cout first get one answer (from P2 and old C1), then new answer (P2 and new C1).
- Notice that the Carry-out/overflow is set if the sum is above 7 ! In fact, we can make sums up to 15.
- If the sum goes wrong, we have to do debugging -- real engineering!
- How fast could we do additions?
  How long would it take to do a million additions? A billion?

  With fast finger tapping, about 12 per second -> 1 million finger taps takes a day
  (86400 seconds/day x 12 taps/second = 1.03 Million taps/day)
  1 billion takes 2 years 9 months ... almost 3 years. A trillion 2740 years... A quadrillion (million*billion) takes 2 740 000 years...)

Using transistors, however,
--> a modern day desktop computer can do a billion (thousand*million) additions per second... (Every screen refresh on your video game takes a million additions or so!)
--> a supercomputer can do a trillion (million*million) per second...
-->and BlueGene, a proposed IBM supercomputer for biological calculations, can do a million*billion per second...

- Would you like to be a logic gate and change 1 to 0 or 0 to 1 all day? Would it get boring? Why? Is a logic gate intelligent? Does it require thinking? Does a logic gate understand arithmetic?
  No, logic gates just do a simple job, all day, very fast... And they don't understand what they are really doing.
  In the case of our adder-on-the-floor, only the "tester person" and the people wiggling the inputs and looking at the outputs can figure out what the meaning of those 1s and 0s actually is. The gates in the middle just did their job, but did not have a clue as to what it meant.

  You have learned to add and multiply in school... Slowly... But at least, you understand what you are doing. You know *what* to add and *when* to multiply, and that is more than the computer knows!

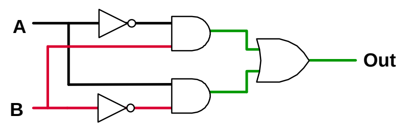
  If you are really interested in how the above circuit works, please check out the IBM Family Science Web page, http://www.watson.ibm.com/leo/fs.html , where you will find a link to the Additional Information for the Introduction to Electronics class.

7. Play time with switch boards

We had a number of “suitcases” with logic gates, that could be wired together.
Each input or output has a little light (LED = light-emitting diode) indicating whether the input is 0 (dark) or 1 (light).

- Simple logic gates : and, or, not, xor

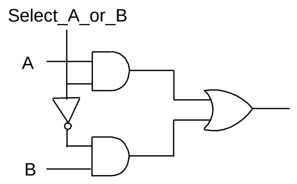
- Build an XOR from AND, OR and NOT gates:



Do you see that the logic here is:
Out = (not(A) **and** B) **or** (A **and** not(B)) ?

And that this is the same as a single XOR gate does?

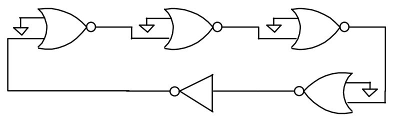
- More complex logic gates: Multiplexer (mux).



Can you build a multiplexer from two ANDs, an OR and a NOT? (Hint: see the “choice” circuit).

If Select_A_or_B = 1, the output follows A and ignores B.
If Select_A_or_B = 0, the output follows B and ignores A.

- Can you make an oscillator?
  Any odd number (of inverting circuits) in a ring will oscillate. Why?



•The unused input of the not-ORs (NORs) is grounded (tied to 0). What would happen if you put a 1 on it?
•What happens if you unplug one of the connections?
•What would happen if you have an even number of inverting circuits in the ring?

- We made a 2-bit adder: a smaller version of the 3-bit adder-on-the-floor.

- Additions we plan for later: Sensor inputs, actuator outputs
  - Build logic for a toaster:
    Heater is on when power switch is on AND the thermal sensor is cold
    Alarm is on when power switch is on AND thermal sensor is warm.

8. How do you make all this stuff?

a. What designers do:
They first think of an architecture (like: I need to add up two numbers)
They then think of a way of arranging logic gates to make an adder: like the big schematic above. Once they are satisfied that that works, the schematic is converted into a layout.
The layout corresponds to masks that are made for each layer of the chip:
the layers that define transistors and wires.

b. On the process end:
We talked above about sand being converted to silicon wafers.
The masks (corresponding to what the designer specified) are used to make patterns on the wafer, using a photographic process. That way, all transistors on a wafer are made at the same time, all wires on a given layer are made at the same time, and the chips are built up layer by layer.

c. Once the wafer is finished, the chips are tested. The good chips are diced out with a wafer saw, are put in a package and soldered onto a board.

d. The boards are the heart of the electronic system. But of course it needs a box to fit in, displays, knobs, power supply, cooling, hard drives etc.
And it needs software (programs) to run.

Every step in this list needs good scientists and engineers. People who know their math and science, who can solve problems, and think of new ways to do things, or new things to do.
And oh yeah... we need to sell it...

9. Safety issues

The suitcase-electronics that we played with is safe to touch: only 5 Volt electronics.
If you want to explore more:

- You can take electronic things apart ***if*** your parents give you permission
- But, you should NEVER take electronic things apart
  - if they have a line cord (110V is way more lethal than 5V)
  - if they have a picture tube (thousands or tens of thousands of Volts -- really lethal).
- Soldering:
  - requires a fan
  - requires safety glasses

10. Where to go from here

http://math.hws.edu/TMCM/java/labs/xLogicCircuitsLab1.html
is an on-the-web logic simulator, part of a digital logic course. It includes an applet which can be used to create and test logic circuits. Allows the user to position gates, tie elements together and turn inputs on and off. Excellent to play with if you want to explore more digital logic on your own.

Several toy stores and science/hobby stores sell electronic hobby kits.
Internet: a good place to start is: http://shopping.yahoo.com/ , search on: electronic+hobby+kits .
Note that many electronic hobby kits (metal detectors, burglar alarms, walkie-talkie, radio, etc.) are *analog* in nature, that is: effects are dependent on exact values of voltages and currents.
In contrast, in this course we focussed on switch logic, which is *digital*: all decisions are made on basis of true-false, yes-no, or 1-0, which is independent of the exact values of voltages or currents -- it only matters if a voltage is high or low.