Chapter 1

Bits

Information is measured in bits, just as length is measured in meters and time is measured in seconds. Of course knowing the amount of information, in bits, is not the same as knowing the information itself, what it means, or what it implies. In these notes we will not consider the content or meaning of information, just the quantity.

Different scales of length are needed in different circumstances. Sometimes we want to measure length in kilometers, sometimes in inches, and sometimes in Ångströms. Similarly, other scales for information besides bits are sometimes used; in the context of physical systems information is often measured in Joules per Kelvin.

How is information quantified? Consider a situation or experiment that could have any of several possible outcomes. Examples might be flipping a coin (2 outcomes, heads or tails) or selecting a card from a deck of playing cards (52 possible outcomes). How compactly could one person (by convention often named Alice) tell another person (Bob) the outcome of such an experiment or observation?

First consider the case of the two outcomes of flipping a coin, and let us suppose they are equally likely. If Alice wants to tell Bob the result of the coin toss, she could use several possible techniques, but they are all equivalent, in terms of the amount of information conveyed, to saying either "heads" or "tails" or to saying 0 or 1. We say that the information so conveyed is one bit.

If Alice flipped two coins, she could say which of the four possible outcomes actually happened, by saying 0 or 1 twice. Similarly, the result of an experiment with eight equally likely outcomes could be conveyed with three bits, and more generally 2^n outcomes with n bits. Thus the amount of information is the logarithm (to the base 2) of the number of equally likely outcomes.

Note that conveying information requires two phases. First is the "setup phase," in which Alice and Bob decided what they will communicate about, and exactly what each sequence of bits means. This common understanding is called the code. For example, to convey the suit of a card chosen from a deck, their code might be that 00 means clubs, 01 diamonds, 10 hearts, and 11 spades. Agreeing on the code may be done before any observations have been made, when there is not yet any information to be sent. The setup phase can also include informing the recipient that there is new information. Then, there is the "outcome phase," where actual sequences of 0 and 1 representing the outcomes are sent. These sequences are the data. Using the agreed-upon code, Alice draws the card, and tells Bob the suit by sending two bits of data. She could do so repeatedly for multiple experiments, using the same code.

After Bob knows that a card is drawn but before receiving Alice's message, he is uncertain about the suit. His uncertainty, or lack of information, can be expressed in bits. Upon hearing the result, his uncertainty is reduced by the information he receives. Bob's uncertainty rises during the setup phase and then is reduced during the outcome phase.

Author: Paul Penfield, Jr.
This document: http://www.mtl.mit.edu/Courses/6.050/2010/notes/chapter1.pdf
Version 1.6, February 1, 2010. Copyright © 2010 Massachusetts Institute of Technology

Note some important things about information, some of which are illustrated in this example:

- Information can be learned through observation, experiment, or measurement
- Information is subjective, or "observer-dependent." What Alice knows is different from what Bob knows (if information were not subjective, there would be no need to communicate it)
- A person's uncertainty can be increased upon learning that there is an observation about which information may be available, and then can be reduced by receiving that information
- Information can be lost, either through loss of the data itself, or through loss of the code
- The physical form of information is localized in space and time. As a consequence,
  - Information can be sent from one place to another
  - Information can be stored and then retrieved later

1.1 The Boolean Bit

As we have seen, information can be communicated by sequences of 0 and 1 values. By using only 0 and 1, we can deal with data from many different types of sources, and not be concerned with what the data means. We are thereby using abstract, not specific, values. This approach lets us ignore many messy details associated with specific information processing and transmission systems.

Bits are simple, having only two possible values. The mathematics used to denote and manipulate single bits is not difficult. It is known as Boolean algebra, after the mathematician George Boole (1815–1864). In some ways Boolean algebra is similar to the algebra of integers or real numbers which is taught in high school, but in other ways it is different.

Algebra is a branch of mathematics that deals with variables that have certain possible values, and with functions which, when presented with one or more variables, return a result which again has certain possible values. In the case of Boolean algebra, the possible values are 0 and 1.

First consider Boolean functions of a single variable that return a single value. There are exactly four of them. One, called the identity, simply returns its argument. Another, called not (or negation, inversion, or complement) changes 0 into 1 and vice versa. The other two simply return either 0 or 1 regardless of the argument. Here is a table showing these four functions:

x	$f(x)$			
Argument	*IDENTITY*	*NOT*	*ZERO*	*ONE*
0	0	1	0	1
1	1	0	0	1

Table 1.1: Boolean functions of a single variable

Note that Boolean algebra is simpler than algebra dealing with integers or real numbers, each of which has infinitely many functions of a single variable.

Next, consider Boolean functions with two input variables A and B and one output value C. How many are there? Each of the two arguments can take on either of two values, so there are four possible input patterns (00, 01, 10, and 11). Think of each Boolean function as a list of the four outputs (0 or 1) corresponding to each of the four input patterns. There are as many Boolean functions as there are different lists, i.e., 16 (2^4). Of these 16, two simply ignore the input, four assign the output to be either A or B or their complement, and the other ten depend on both arguments. The most often used are *AND*, *OR*, *XOR* (exclusive or), *NAND* (not and), and *NOR* (not or), shown in Table 1.2. (Similarly, because there are 8

x	$f(x)$				
Argument	*AND*	*NAND*	*OR*	*NOR*	*XOR*
00	0	1	0	1	0
01	0	1	1	0	1
10	0	1	1	0	1
11	1	0	1	0	0

Table 1.2: Five of the 16 possible Boolean functions of two variables

possible input patterns for three-input Boolean functions, there are 2^8 or 256 different Boolean functions of three variables.)

It is tempting to think of the Boolean values 0 and 1 as the integers 0 and 1. Then *AND* would correspond to multiplication and *OR* to addition, sort of. However, familiar results from ordinary algebra simply do not hold for Boolean algebra, so such analogies are dangerous. It is important to distinguish the integers 0 and 1 from the Boolean values 0 and 1; they are not the same.

There is a standard notation used in Boolean algebra. This notation is sometimes confusing, but other notations that are less confusing are awkward in practice. The *AND* function is represented the same way as multiplication, by writing two Boolean values next to each other or with a dot in between: $A\ AND\ B$ is written AB or $A \cdot B$. The *OR* function is written using the plus sign: $A + B$ means $A\ OR\ B$. Negation, or the *NOT* function, is denoted by a bar over the symbol or expression, so $NOT\ A$ is written $\overline{A}$. Finally, the exclusive-or function *XOR* is represented by a circle with a plus sign inside, $A \oplus B$.

NOT	$\overline{A}$
AND	$A \cdot B$
NAND	$\overline{A \cdot B}$
OR	$A + B$
NOR	$\overline{A + B}$
XOR	$A \oplus B$

Table 1.3: Boolean logic symbols

This standard notation, used here, is the most common, but there are other possible notations. Sometimes *AND*, *OR*, and *NOT* are represented in the form $AND(A, B)$, $OR(A, B)$, and $NOT(A)$. Sometimes infix notation is used where $A \wedge B$ denotes $A \cdot B$, $A \vee B$ denotes $A + B$, and $\sim\!A$ denotes $\overline{A}$. Boolean algebra is also useful in mathematical logic, where the notation $A \wedge B$ for $A \cdot B$, $A \vee B$ for $A + B$, and $\neg A$ for $\overline{A}$ is commonly used.

Several general properties of Boolean functions are useful. These can be proven by simply demonstrating that they hold for all possible input values. For example, a function is said to be **reversible** if, knowing the output, the input can be determined. Two of the four functions of a single variable are reversible in this sense (and in fact are self-inverse). Clearly none of the functions of two (or more) inputs can by themselves be reversible, since there are more input variables than output variables. However, some combinations of two or more such functions can be reversible if the resulting combination has the name number of outputs as inputs; for example it is easily demonstrated that the exclusive-or function $A \oplus B$ is reversible when augmented by the function that returns the first argument—that is to say, more precisely, the function of two variables that has two outputs, one $A \oplus B$ and the other A, is reversible.

For functions of two variables, there are many properties to consider. For example, a function of two variables A and B is said to be **commutative** if its value is unchanged when A and B are interchanged, i.e., if $f(A, B) = f(B, A)$. Thus the function *AND* is commutative because $A \cdot B = B \cdot A$. Some of the other 15 functions are also commutative. Some other properties of Boolean functions are illustrated in the identities in Table 1.4.

Idempotent:	$A \cdot A = A$	Absorption:	$A \cdot (A + B) = A$
	$A + A = A$		$A + (A \cdot B) = A$
Complementary:	$A \cdot \overline{A} = 0$	Associative:	$A \cdot (B \cdot C) = (A \cdot B) \cdot C$
	$A + \overline{A} = 1$		$A + (B + C) = (A + B) + C$
	$A \oplus A = 0$		$A \oplus (B \oplus C) = (A \oplus B) \oplus C$
	$A \oplus \overline{A} = 1$		
Minimum:	$A \cdot 1 = A$	Unnamed Theorem:	$A \cdot (\overline{A} + B) = A \cdot B$
	$A \cdot 0 = 0$		$A + (\overline{A} \cdot B) = A + B$
Maximum:	$A + 0 = A$	De Morgan:	$\overline{A} \cdot \overline{B} = \overline{A + B}$
	$A + 1 = 1$		$\overline{A} + \overline{B} = \overline{A \cdot B}$
Commutative:	$A \cdot B = B \cdot A$	Distributive:	$A \cdot (B + C) = (A \cdot B) + (A \cdot C)$
	$A + B = B + A$		$A + (B \cdot C) = (A + B) \cdot (A + C)$
	$A \oplus B = B \oplus A$		
	$\overline{A \cdot B} = \overline{B \cdot A}$		
	$\overline{A + B} = \overline{B + A}$		

Table 1.4: Properties of Boolean Algebra
These formulas are valid for all values of A, B, and C.

The Boolean bit has the property that it can be copied or discarded. In Boolean algebra copying is done by assigning a name to the bit and then using that name more than once. Discarding is done by simply not using the bit. Because of this property the Boolean bit is not a good model for quantum-mechanical systems. A different model, the quantum bit, is described below.

1.2 The Circuit Bit

Combinational logic circuits are a way to represent Boolean expressions graphically. Each Boolean function (*NOT*, *AND*, *XOR*, etc.) corresponds to a "combinational gate" with one or two inputs and one output, as shown in Figure 1.1. The different types of gates have different shapes. Lines are used to connect the output of one gate to one or more gate inputs, as illustrated in the circuits of Figure 1.2.

Logic circuits are widely used to model digital electronic circuits, where the gates represent parts of an integrated circuit and the lines represent the signal wiring.

The circuit bit can be copied (by connecting the output of a gate to two or more gate inputs) and

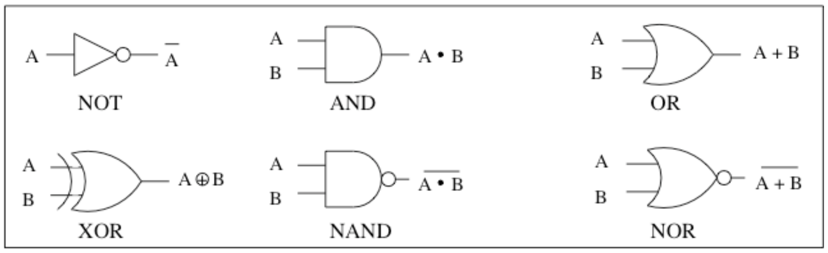

Figure 1.1: Logic gates corresponding to the Boolean functions *NOT*, *AND*, *OR*, *XOR*, *NAND*, and *NOR*

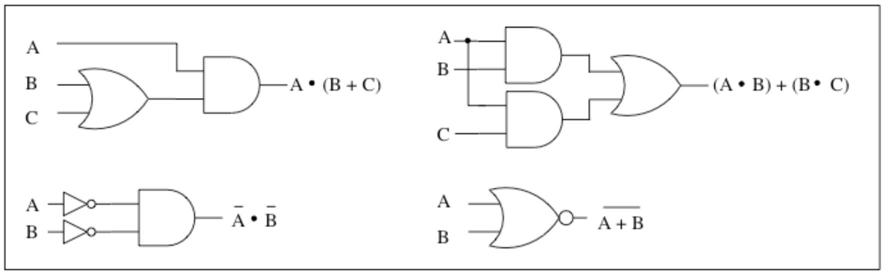

Figure 1.2: Some combinational logic circuits and the corresponding Boolean expressions

discarded (by leaving an output unconnected).

Combinational circuits have the property that the output from a gate is never fed back to the input of any gate whose output eventually feeds the input of the first gate. In other words, there are no loops in the circuit. Circuits with loops are known as **sequential logic**, and Boolean algebra is not sufficient to describe them. For example, consider the simplest circuit in Figure 1.3. The inverter (*NOT* gate) has its output connected to its input. Analysis by Boolean algebra leads to a contradiction. If the input is 1 then the output is 0 and therefore the input is 0. There is no possible state according to the rules of Boolean algebra. On the other hand, consider the circuit in Figure 1.3 with two inverters. This circuit has two possible states. The bottom circuit has two stable states if it has an even number of gates, and no stable state if it has an odd number.

A model more complicated than Boolean algebra is needed to describe the behavior of sequential logic circuits. For example, the gates or the connecting lines (or both) could be modeled with time delays. The circuit at the bottom of Figure 1.3 (for example with 13 or 15 gates) is commonly known as a ring oscillator, and is used in semiconductor process development to test the speed of circuits made using a new process.

1.3 The Control Bit

In computer programs, Boolean expressions are often used to determine the flow of control, i.e., which statements are executed. Suppose, for example, that if one variable x is negative and another y is positive, then a third variable z should be set to zero. In the language Scheme, the following statement would accomplish this

```
(if (and (< x 0) (> y 0)) (define z 0))          (1.1)
```

Other languages have their own ways of expressing the same thing.

The algebra of control bits is like Boolean algebra with an interesting difference: any part of the control expression that does not affect the result may be ignored. In the case above (assuming the arguments of `and` are evaluated left to right), if x is found to be positive then the result of the `and` operation is 0 regardless of the value of y, so there is no need to see if y is positive or even to evaluate y. As a result the program can run faster, and side effects associated with evaluating y do not happen.

1.4 The Physical Bit

If a bit is to be stored or transported, it must have a physical form. Whatever object stores the bit has two distinct states, one of which is interpreted as 0 and the other as 1. A bit is stored by putting the object in one of these states, and when the bit is needed the state of the object is observed. If the object has moved from one location to another without changing its state then communications has occurred. If the object

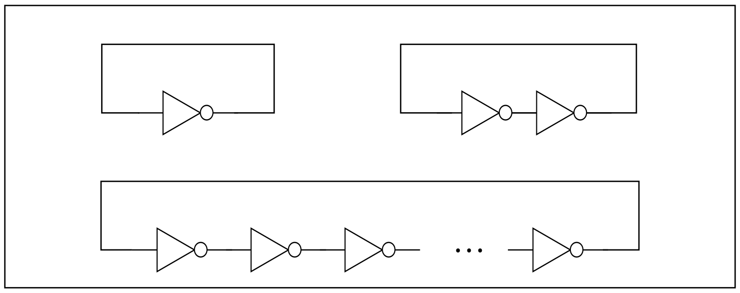

Figure 1.3: Some sequential logic circuits

has persisted over some time in its same state then it has served as a memory. If the object has had its state changed in a random way then its original value has been forgotten.

In keeping with the Engineering Imperative (make it smaller, faster, stronger, smarter, safer, cheaper), we are especially interested in physical objects that are small. The limit to how small an object can be and still store a bit of information comes from quantum mechanics. The quantum bit is a model of an object that can store a single bit but is so small that it is subject to the limitations quantum mechanics places on measurements.

1.5 The Quantum Bit

According to quantum mechanics, it is possible for a small object to have two states which can be observed. This sounds perfect for storing bits, and the result is often called a **qubit**, pronounced "cue-bit." The two values are often denoted $|0\rangle$ and $|1\rangle$ rather than 0 and 1, because this notation generalizes to what is needed to represent more than one qubit, and it avoids confusion with the real numbers 0 and 1. There are three features of quantum mechanics, **reversibility**, **superposition**, and **entanglement**, that make qubits or collections of qubits different from Boolean bits.

Reversibility: It is a property of quantum systems that if one state can lead to another by means of some transition, then the reverse transition is also possible. Thus all functions in the mathematics of qubits are reversible, and the output of a function cannot be discarded, since that would be irreversible. However, there are at least two important sources of irreversibility in quantum systems. First, if a quantum system interacts with its environment and the state of the environment is unknown, then some of the information in the system is lost. Second, the very act of measuring the state of a system is irreversible.

Superposition: Suppose a quantum mechanical object is prepared so that it has a combination, or superposition, of its two states, i.e., a state somewhere between the two states. What is it that would be observed in that case?

In a classical, non-quantum context, a measurement could determine just what that combination is. Furthermore, for greater precision a measurement could be repeated, and multiple results averaged. However, the quantum context is different. In a quantum observation, the question that is asked is whether the object is or is not in some particular state, and the answer is always either "yes" or "no," never "maybe" and never "27% yes, 73% no." Furthermore, after the observation the system ends up in the state corresponding to the answer, so further observations will not yield additional information. The result of any particular observation cannot be predicted, but the likelihood of the answer, expressed in terms of probabilities, can. This peculiar nature of quantum mechanics offers both a limitation of how much information can be carried by a single qubit, and an opportunity to design systems which take special advantage of these features.

We will illustrate quantum bits with an example. Let's take as our qubit a photon, which is the elementary particle for electromagnetic radiation, including radio, TV, and light. A photon is a good candidate for carrying information from one place to another. It is small, and travels fast.

A photon has electric and magnetic fields oscillating simultaneously. The direction of the electric field is called the direction of polarization (we will not consider circularly polarized photons here). Thus if a photon is headed in the z-direction, its electric field can be in the x-direction, in the y-direction, or in fact in any direction in the x-y plane, sometimes called the "horizontal-vertical plane."

The polarization can be used to store a bit of information. Thus Alice could prepare a photon with horizontal polarization if the bit is $|0\rangle$ and vertical polarization if the bit is $|1\rangle$. Then when Bob gets the photon, he can observe its vertical polarization (i.e., ask whether the polarization is vertical). If the answer is "yes," then he infers the bit is $|1\rangle$.

It might be thought that more than a single bit of information could be transmitted by a single photon's polarization. Why couldn't Alice send two bits, using angles of polarization different from horizontal and vertical? Why not use horizontal, vertical, half-way between them tilted right, and half-way between them tilted left? The problem is that Bob has to decide what angle to observe. He cannot, because of quantum-mechanical limitations, ask the question "what is the angle of polarization" but only "is the polarization in the direction I choose to observe." And the result of his observation can only be "yes" or "no," in other words, a single bit. And then after the observation the photon ends up either in the plane he observed (if the result was "yes") or perpendicular to it (if the result was "no").

If Bob wants to measure the angle of polarization more accurately, why couldn't he repeat his observation many times and take an average? This does not work because the very act of doing the first observation resets the angle of polarization either to the angle he observed or to the angle perpendicular to it. Thus subsequent observations will all be the same.

Or Bob might decide to make multiple copies of the photon, and then observe each of them. This approach does not work either. The only way he can make a copy of the photon is by measuring its properties and then creating a new photon with exactly those properties. All the photons he creates will be the same.

What does Bob observe if Alice had prepared the photon with an arbitrary angle? Or if the photon had its angle of polarization changed because of random interactions along the way? Or if the photon had been observed by an evil eavesdropper (typically named Eve) at some other angle and therefore been reset to that angle? In these cases, Bob always gets an answer "yes" or "no," for whatever direction of polarization he chooses to observe, and the closer the actual polarization is to that direction the more likely the answer is yes. To be specific, the probability of the answer yes is the square of the cosine of the angle between Bob's angle of observation and Alice's angle of preparation. It is not possible to predict the result of any one of Bob's observations. This inherent randomness is an unavoidable aspect of quantum mechanics.

Entanglement: Two or more qubits can be prepared together in particular ways. One property, which we will not discuss further now, is known as "entanglement." Two photons, for example, might be prepared to always have identical polarizations (either both horizontal or both vertical). Then they might travel to different places but retain their entangled polarizations. They would then be separate in their physical locations but not separate in their polarizations. If you think of them as two separate photons you might wonder how observation of the polarization of one would affect a subsequent observation of the polarization of the other, located far away.

Note that quantum systems don't *always* exhibit the peculiarities associated with superposition and entanglement. For example, photons can be prepared independently (so there is no entanglement) and the angles of polarization can be constrained to be horizontal and vertical (no superposition). In this case qubits behave like Boolean bits.

1.5.1 An Advantage of Qubits

There are things that can be done in a quantum context but not classically. Some are advantageous. Here is one example:

Consider again Alice trying to send information to Bob using polarized photons. She prepares photons that have either horizontal or vertical polarization, and tells that to Bob, during the setup phase. Now let us suppose that a saboteur Sam wants to spoil this communication by processing the photons at some point in the path between Alice and Bob. He uses a machine that simply observes the polarization at an angle he selects. If he selects 45°, every photon ends up with a polarization at that angle or perpendicular to it, regardless of its original polarization. Then Bob, making a vertical observation, will observe 0 half the time and 1 half the time, regardless of what Alice sent.

Alice learns about Sam's scheme and wants to reestablish reliable communication with Bob. What can she do?

She tells Bob (using a path that Sam does not overhear) she will send photons at 45° and 135°, so he should observe at one of those angles. Sam's machine then does not alter the photons. Of course if Sam discovers what Alice is doing, he can rotate his machine back to vertical. Or there are other measures and counter-measures that could be put into action.

This scenario relies on the quantum nature of the photons, and the fact that single photons cannot be observed by Sam except along particular angles of polarization. Thus Alice's technique for thwarting Sam is not possible with classical bits.

1.6 The Classical Bit

Because quantum observation generally alters the object being observed, a quantum bit cannot be observed a second time. On the other hand, if a bit is represented by many objects with the same properties, then after an observation enough objects can be left unchanged so that the same bit can be observed again.

In today's electronic systems, a bit of information is carried by many objects, all prepared in the same way (or at least that is a convenient way to look at it). Thus in a semiconductor memory a single bit might be represented by the presence or absence of perhaps 30,000 electrons (stored on a 5 fF capacitor charged to $1V$). Similarly, a large number of photons are used in radio communication.

Because many objects are involved, measurements of their properties are not restricted to a simple yes or no, but instead can range over a continuum of values. Thus the voltage on a semiconductor logic element might be anywhere in a range from, say, $0V$ to $1V$. The voltage might be interpreted to allow a margin of error, so that voltages between $0V$ and $0.2V$ would represent logical 0, and voltages between $0.8V$ and $1V$ a logical 1. The circuitry would not guarantee to interpret voltages between $0.2V$ and $0.8V$ properly. If the noise in a circuit is always smaller than $0.2V$, and the output of every circuit gate is either $0V$ or $1V$, then the voltages can always be interpreted as bits without error.

Circuits of this sort display what is known as "restoring logic" since small deviations in voltage from the ideal values of $0V$ and $1V$ are eliminated as the information is processed. Modern computers are robust because they use restoring logic.

A classical bit is an abstraction in which the bit can be observed without perturbing it. As a result copies of a classical bit can be made. This model works well for circuits using restoring logic.

Because all physical systems ultimately obey quantum mechanics, the classical bit is always an approximation to reality. However, even with the most modern, smallest devices available, it is an excellent one. An interesting question is whether the classical-bit approximation will continue to be useful as advances in semiconductor technology allow the size of components to be reduced. Ultimately, as we try to represent or control bits with a small number of atoms or photons, the limiting role of quantum mechanics will become important. It is difficult to predict exactly when this will happen.

1.7 Summary

There are several models of a bit, useful in different contexts. These models are not all the same. In the rest of these notes, the Boolean bit will be used most often, but sometimes the quantum bit will be needed.