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. 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 agree on what they will communicate about, and exactly what each sequence of bits means. This common understanding is called the code. Then, there is the "outcome" phase, where actual sequences of 0 and 1 representing the outcomes are sent. These sequences are the data.

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.

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

Think of each Boolean function of two variables as a string of Boolean values 0 and 1 of length equal to the number of possible input patterns, i.e., 4. There are exactly 16 (2^4) different ways of composing such strings, and hence exactly 16 different Boolean functions of two variables. 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, NAND, and NOR. In a similar way, because there are 8 possible input patterns for three-input Boolean functions, there are 2^8 or 256 different Boolean functions of three variables.

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

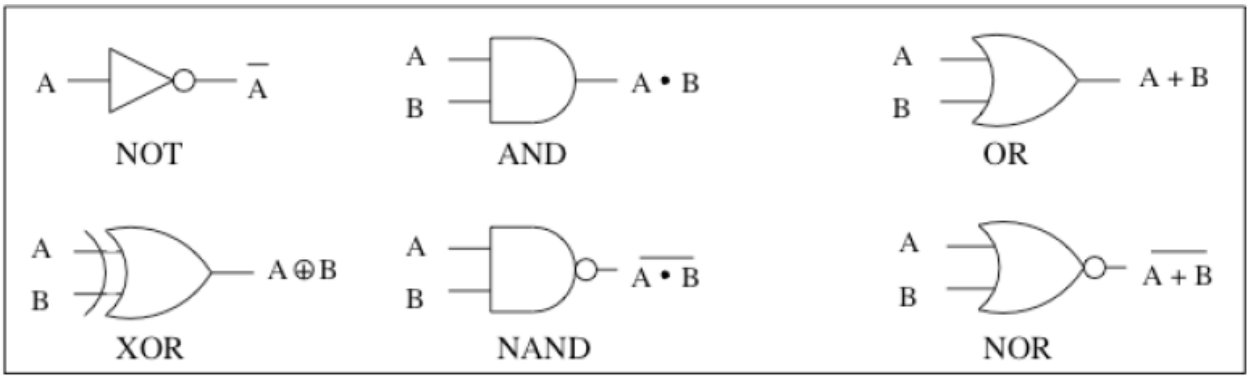
such functions can be reversible if the resulting combination has the same number of outputs as inputs; for example it is easily demonstrated that the exclusive-or function A ⊕ B is reversible when augmented by the function that returns the first argument—that is to say, more precisely, the function of two variables that has two outputs, one A ⊕ 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.B = B.A. Some of the other 15functions are also commutative. Some other properties of Boolean functions are illustrated in the identities in Table

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

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 the Figure. The different types of gates have different shapes.

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 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 measured. If the object has moved from one location to another without changing its state then communications has occurred. If the object 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 Classical Bit

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 is represented by the presence or absence of perhaps 60,000 electrons (stored on a 10 fF capacitor charged to 1V). Similarly, a large number of photons are used in radio communication.

Because many objects are involved, measurements on them 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 logical1.

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.

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.