ANIMESH DATTA

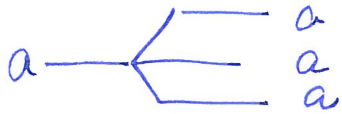
1. Classical Computational Models

(warwick.ac.uk/qinfo)

1. Turing Machine (coming later).
2. Circuit Model : bits, wires, gates.

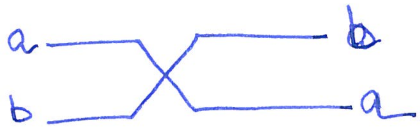
SINGLE BIT GATES:

1. IDENTITY: $a \longrightarrow a$
2. NOT $\quad a - \boxed{\text{NOT}} - \bar{a} = a \oplus 1$
3. FANOUT (COPY)

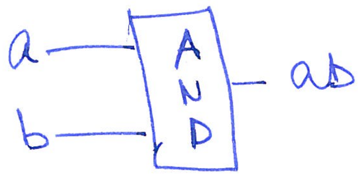


TWO BIT GATES:

1. SWAP

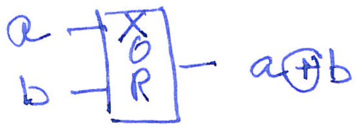


2. AND



a	b	ab
0	1	0
0	1	0
1	0	0
1	1	1

3. XOR

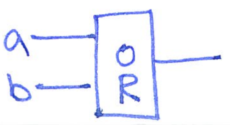


a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

4. NAND – not AND – $1 \oplus ab$

5. NOR – not OR – $1 \oplus a \oplus b \oplus ab$

6. OR



$a \oplus b \oplus ab = 1 \oplus (1 \oplus a)(1 \oplus b)$

a	b	a OR b
0	0	0
0	1	1
1	0	1
1	1	1

That is already quite a lot of gates, so what can we do with these things.

<u>Question:</u> What <u>do</u> we want to do?

<u>Answer:</u> Evaluate functions $f: \{0,1\}^N \to \{0,1\}^M$

The simplest case is $M=1$. Such Boolean functions are called <u>DECISION</u> problems.

In fact, there are 2^{2^N} Boolean functions on N bits.
Single- and two-bit gates are $N=1$ & $N=2$ Boolean functions

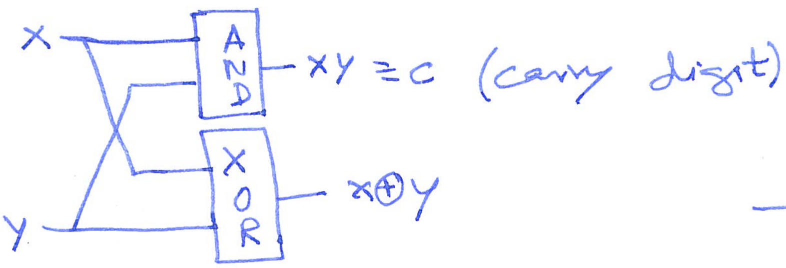
Most (or many) functions (or problems) can be cast as decision problems. <u>Example:</u> Factoring

$$f(x,y) = \begin{cases} 1 & \text{if } x \text{ has a factor } y \\ 0 & \text{o.w.} \end{cases}$$

<u>EXAMPLE:</u> Add $x = x_2 x_1 x_0$ and $y = y_2 y_1 y_0$

Needs a half-adder (HA) and a full-adder (FA).

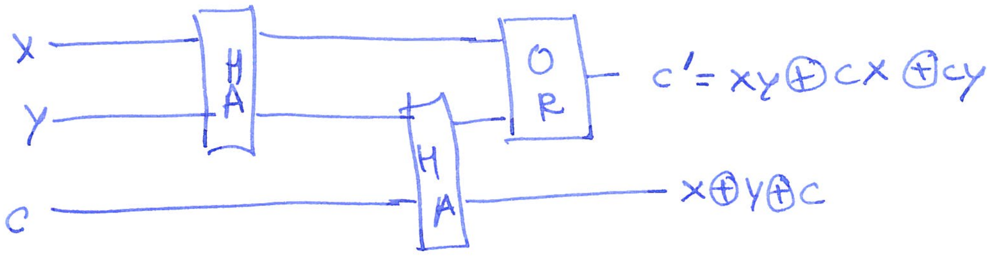
<u>HALF ADDER:</u>



X	Y	C	X⊕Y
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

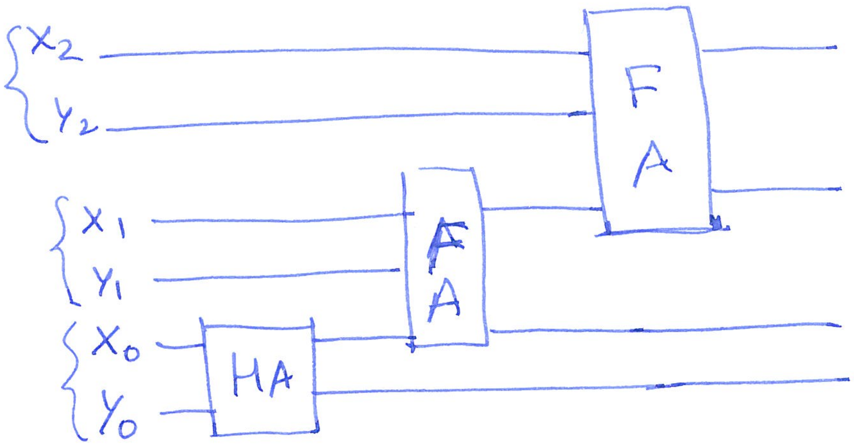
Binary version of the add & carry method ~~the~~ taught to children.

FULL ADDER:



x	y	c	c'	$x \oplus y \oplus c$
0	0	0	0	0
0	1	0	0	1
1	0	0	0	1
1	1	0	1	0
0	0	1	0	1
0	1	1	1	0
1	0	1	1	0
1	1	1	1	1

Putting it all together:



Computing any Boolean function:

The computation proceeds by mathematical induction.

Assume there is a circuit for any Boolean function on N bits.

Say, f is an $(N+1)$-bit function.

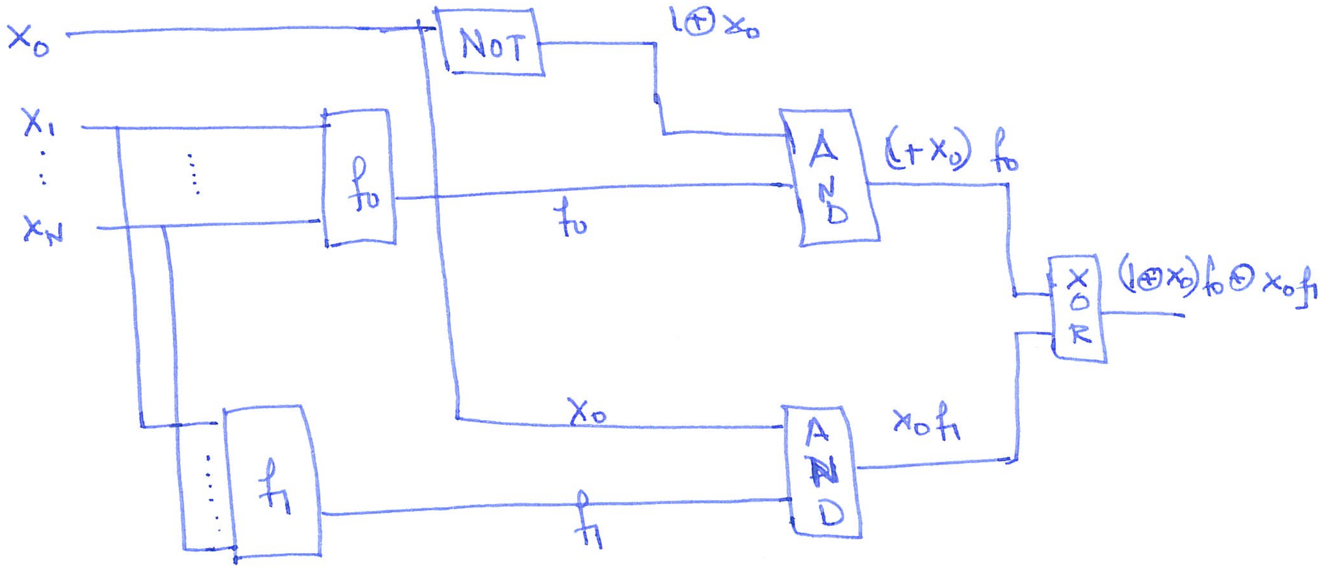
then define N-bit functions f_0 and f_1 as

$f_0(x_1, \dots x_N) \equiv f(0, x_1, \dots, x_N)$

$f_1(x_1, \dots x_N) \equiv f(1, x_1, \dots x_N)$ that is,

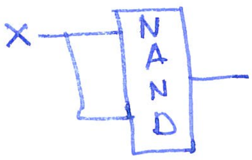
$f_{x_0}(x_1, \dots, x_N) = f(x_0, x_1, \dots, x_N)$ OR $(1 \oplus x_0) f_0(x_1, \dots, x_N) \oplus x_0 f_1(x_1, \dots x_N)$.

Thus:



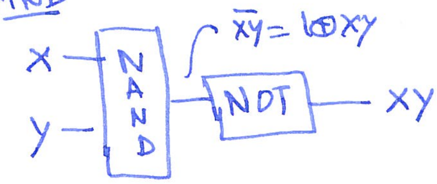
The induction thus needs AND, NOT, XOR & FANOUT.

Furthermore, NOT

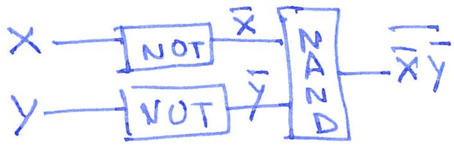


$1 \oplus x^2 = 1 \oplus x = \bar{x}$

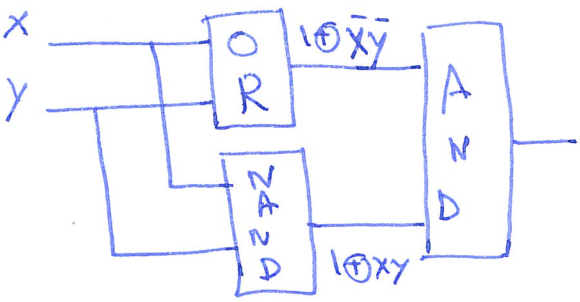
AND



OR



XOR



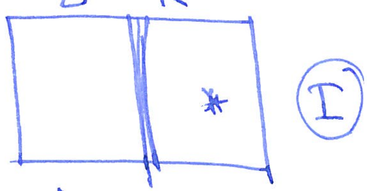
$$(1 \oplus \bar{x}\bar{y})(1 \oplus xy) = 1 \oplus \bar{x}\bar{y} + xy + 0$$
$$= 1 \oplus (1 \oplus x)(1 \oplus y) + xy$$
$$= x \oplus y.$$

NAND is a <u>universal gate</u> with ancilla bits & FANOUT.

The NAND gate is also irreversible. This is because it has 2 input bits and one output bit; and it is not possible to recover the inputs by knowing the output bits.

This leads to energy dissipation: Landauer's principle.

Here is an illustration: A molecule of gas can be put in a box with a partition. We can put the gas in the left or Right. Say on the R.



Now, erasure means putting it in one of the sides irrespective of where it started. This can be done by removing the partition and then compressing the partition to one side.

This reduces the entropy by

$$\Delta S = k \ln 2.$$

At isothermal temperature T, $\Delta W = kT \ln 2$, which has to be provided. Thus, erasure of information leads to an energy bill. At room temperature $(20°C)$, this is about

$$\Delta W \approx 2.75 \text{ zJ} \sim 10^{-24} \text{ J}.$$

Modern computations are at about 10^{6} times this.

~~Hardware~~

Reversible classical computation:

Any irreversible function $f: \{0,1\}^N \longrightarrow \{0,1\}^M$ can be embedded in a function $\tilde{f}: \{0,1\}^{N+M} \longrightarrow \{0,1\}^{N+M}$ such that

$$\tilde{f}(\underset{\underset{N}{\uparrow}}{x}, \underset{\underset{M}{\uparrow}}{0^M}) = (x, f(x)).$$

$\tilde{f}$ can be extended to a 1-1 function, e.g:

$$\boxed{\tilde{f}(x,y) = (x, y \oplus f(x))}$$

This is not a unique extension, but we will use it as the reversible function.

Question: Does this work for reversible gates as well?

Number of N-bit gates: (N inputs N outputs) : $\left(2^N\right)^{2^N}$: (N_i)

" " " reversible gates : $(2^N)!$: (N_r)

	N=1	N=2
N_i :	4	256
N_r :	2	24

Let us study this in some detail for N=2 bits. A general 2 bit gate is:

$$\begin{pmatrix} x \\ y \end{pmatrix} \longrightarrow \begin{pmatrix} x' \\ y' \end{pmatrix} = \begin{pmatrix} a \oplus M_{11}x \oplus M_{12}y \oplus cxy \\ b \oplus M_{21}x \oplus M_{22}y \oplus dxy \end{pmatrix}$$

$$= \begin{pmatrix} a \\ b \end{pmatrix} \oplus M \begin{pmatrix} x \\ y \end{pmatrix} \oplus \begin{pmatrix} c \\ d \end{pmatrix} xy$$

As one can see, the irreversibility resides in the nonlinear term. Thus, 2-bit reversible gates are all linear, with invertible M.

$$\begin{pmatrix} x \\ y \end{pmatrix} \to M \begin{pmatrix} x \\ y \end{pmatrix} \oplus \begin{pmatrix} a \\ b \end{pmatrix}$$

$$M : \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 0 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}, \begin{pmatrix} 1 & 1 \\ 1 & 0 \end{pmatrix}$$

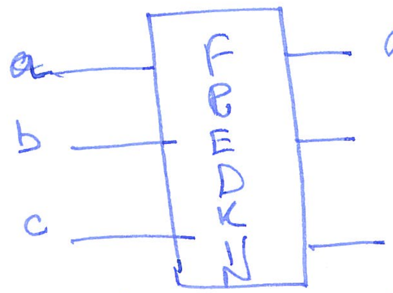
$$\begin{pmatrix} a \\ b \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} 0 \\ 1 \end{pmatrix}, \begin{pmatrix} 1 \\ 0 \end{pmatrix}, \begin{pmatrix} 1 \\ 1 \end{pmatrix}$$

Thus, $6 \times 4 = 24$ possibilities.

EXERCISE: Show that no 2-bit gate can lead to a universal set.

$\therefore$ We need 3 bit reversible gates for a universal set.

FREDKIN GATE:



$a' = \bar{c}a \oplus cb$

$b' = ca \oplus \bar{c}b$

nonlinearities

$c' = c$ (control bit)

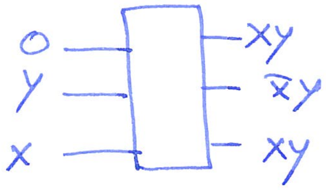
a b c	a' b' c'
0 0 0	0 0 0
0 0 1	0 0 1
0 1 0	0 1 0
0 1 1	1 0 1
1 0 0	1 0 0
1 0 1	0 1 1
1 1 0	1 1 0
1 1 1	1 1 1

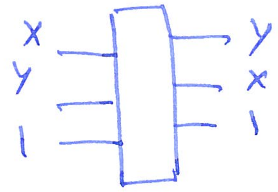
FREDKIN is its own inverse.

When $c=0$: $(a,b) \rightarrow (a,b)$, ~~[illegible]~~ $c=1$, $(a,b) = (b,a)$.

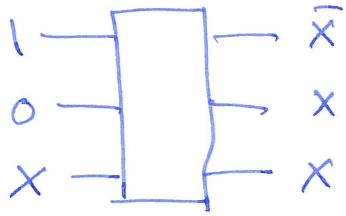
(controlled SWAP)

AND





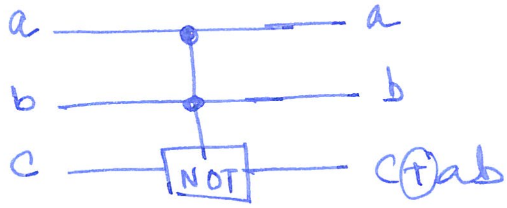
NOT & FANOUT



NOT & AND give NAND.

$\therefore$ FREDKIN is a universal and reversible gate.

TOFFOLI GATE: (Controlled controlled NOT)

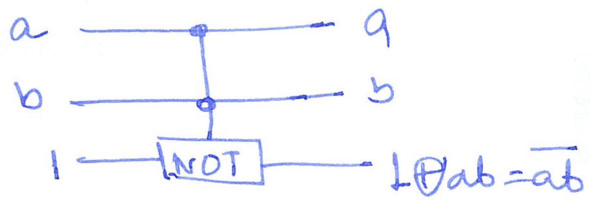


$\leftarrow$ nonlinearity

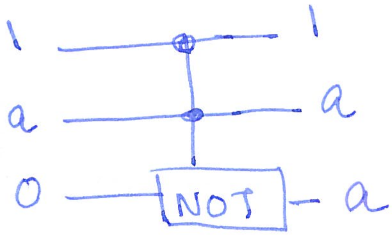
a	b	c	a	b	$c \oplus ab$
0	0	0	0	0	0
0	0	1	0	0	1
0	1	0	0	1	0
0	1	1	0	1	1
1	0	0	1	0	0
1	0	1	1	0	1
1	1	0	1	1	1
1	1	1	1	1	0

TOFFOLI is its own inverse.

NAND : $(c=1)$



FANOUT.



TOFFOLI is a universal and reversible gate.

Computing a function reversibly:

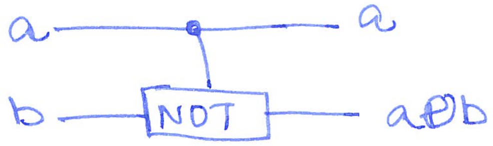
$$\underset{\substack{\uparrow \\ N}}{X}, \underset{\substack{\uparrow \\ M}}{0}, \underset{\substack{\downarrow \\ \text{whatever}}}{0}, \underset{\substack{\uparrow \\ M}}{Y} \qquad (X, 0, 0, Y) \xrightarrow[\text{implementation of } f]{\text{Reversible gate}} (X, f(X), \underset{\substack{\downarrow \\ \text{garbage bits, however many needed}}}{g(X)}, Y)$$

$$\xrightarrow[\text{Reversibly using CNOTS}]{\text{Copy } f(x) \text{ into last M bits}} (X, f(X), g(X), Y \oplus f(X))$$

$$\xrightarrow[\text{reversibly}]{\text{Uncompute } f} (X, 0, 0, Y \oplus f(X))$$

EXERCISE:

1. Show that the CNOT Gate is reversible.



2. Obtain the SWAP gate using only CNOT.

3. What is the least such ~~mb~~ number?