Reversible Computing for Beginners

Lecture 1.

Marek Perkowski

The Most Important aspect of research is Motivation

Why I have motivation to work on Reversible Logic?

Reasons to work on Reversible Logic

- *Build Intelligent Robots*

- *Save Power*

- *Save our Civilization*

  - *and U.S Supremacy*

End presentation

Honda's robot battery is only for 15 minutes!

How to build extremely large finite state machines with small power consumption??

…and this leads us to the second reason…..

Need for Dramatic Power Reduction

Save our Civilization

Quantum Computers will be reversible

Quantum Computers will solve NP-hard problems in polynomial time

If Quantum Computers will be not build, USA and the world will be in trouble

Motivation for this work:
Quantum Logic touches the future of our civilization

- We live in a very exciting time.

- US economy grows

- World economy grows

- Thanks to advances in information-processing technology.

- Usefulness of computers has been enabled primarily by semiconductor-based electronic transistors.

Moore's Law

- In 1965, Gordon Moore observed a trend of increasing performance in the first few generations of integrated-circuit technology.

- He predicted that in fact it would continue to improve at an exponential rate - with the performance per unit cost increasing by a factor of 2 every 18 months or so - for at least the next 10 years.

- The computer industry has followed his prediction throughout for 45 years..

- <u>Increased power of computers</u> created a new civilization:

  - communications, manufacturing, finance, and all manner of products and services.

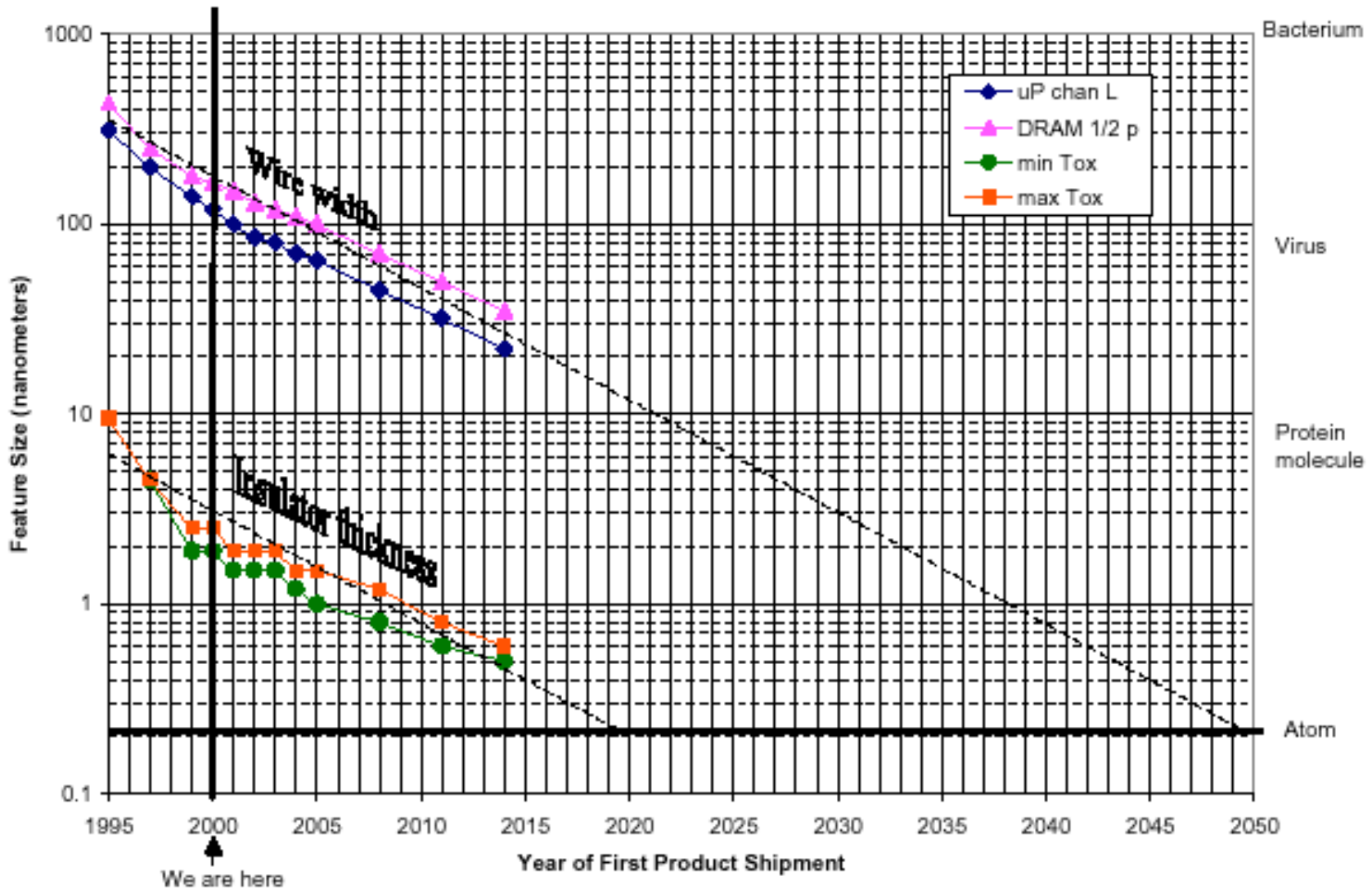 - It has affected nearly everything.

Questions.

- How long can Moore's law continue to hold?

- What are the ultimate limits, if any, to computing technology?

- How will the technology need to change in order to improve as much as possible?

- What will happen to our civilization if the Moore Law will stop to work?

Amazing thing

- A product that seems to get better in every respect as you make it smaller and smaller.

- You *can* make it smaller as you improve the fabrication process in a fairly methodical way.

- But, how long can this trend continue?

- There are a lot of fundamental physical limits that will ultimately come to bear on the shrinkage of transistors, and the improvement of computing technology in general.

- Even beyond this, after not too many decades of Moore's law, the entire transistor itself will approach the atomic scale.

ITRS Feature Size Projections

Legend:
- uP chan L
- DRAM 1/2 p
- min Tox
- max Tox

Chart labels: Wire width, Insulator thickness

Y-axis: Feature Size (nanometers) — 0.1, 1, 10, 100, 1000

X-axis: Year of First Product Shipment — 1995, 2000, 2005, 2010, 2015, 2020, 2025, 2030, 2035, 2040, 2045, 2050

Right-side reference scale: Bacterium, Virus, Protein molecule, Atom

We are here (at 2000)

Problems in Reversible/Quantum Computing

- <u>Building physically</u> Reversible Gates in quantum, optical or CMOS technologies (Fredkin, Feynman, Toffoli gates)
  - this is done by physicists and material science people,
  - requires deep understanding of quantum mechanics,
  - big costs, expensive labs.
- <u>Solving NP-hard problems</u> in polynomial time using hypothetical gates and computers (Deutsch)
  - this is done by mathematicians and computer scientists in IBM, Bell Labs, MIT and other top places.
  - Does not require understanding of physics of quantum mechanics but only its mathematics.
  - We can do it if we learn more math - Sarah Mocas?

Problems in Reversible/Quantum Computing

- <u>Building physically</u> Reversible Computers from CMOS gates (MIT, USC, Korea, De Vos).
  - this is done by EE scientists with background in CMOS design.
  - Does not require understanding of quantum mechanics,
  - we can do this.
- <u>Designing new Reversible and Quantum gates</u> . (De Vos, Kerntopf, Picton). This requires elementary knowledge of logic synthesis. Topic of this lecture. We can do this. Easy.
- <u>Designing logic synthesis theory for reversible and Quantum Logic</u>
  - this is done by mathematicians and computer scientists with no background in logic synthesis
  - Picton and other, weak results, few publications
  - This is a virgin field waiting for pioneers.
  - We should do this - see this lecture.

Landauer Principle and What is Reversible Logic Gate?

Classical gates are irreversible

- Conventional computers are built from basic building blocks, such as the AND, NAND, OR, NOR, and XOR gates.

- Such tables are logically irreversible.

- This means that, if we forget the value of the input (*A, B*), knowledge of the output *P* is not sufficient to calculate backwards and to recover the value of (*A, B*).

AB	P
0 0	0
0 1	0
1 0	0
1 1	1

AB	P
0 0	1
0 1	1
1 0	1
1 1	0

AB	P
0 0	0
0 1	1
1 0	1
1 1	1

AB	P
0 0	1
0 1	0
1 0	0
1 1	0

AB	P
0 0	0
0 1	1
1 0	1
1 1	0

(a) (b) (c) (d) (e)

AND   NAND   OR   NOR   XOR

Logical irreversibility = physical irreversibility.
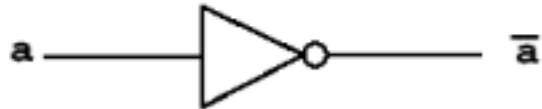
- The NOT gate is reversible
- The AND gate is irreversible
  - the AND gate erases information.
- the AND gate is physically irreversible.

Information is Physical

- Is a minimum amount of energy required per computation step?

A → [irreversible] → A XOR B

B → [irreversible]

- Rolf Landauer, 1970: Whenever we use a logically irreversible gate <u>we dissipate energy into the environment.</u>

A → [reversible] → A

B → [reversible] → A XOR B

Repeating input A on output makes this gate reversible

A	B	X	Y
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Overview of Reversible Gates

- A reversible gate is one where there is a <u>one-to-one correspondence</u> between the inputs and the outputs (i.e., if in the truth table of the gate there is a distinct output row for each distinct input row).

- In Boolean algebra, such a function is both *one-to-one* (or **injective**) and *onto* (or **surjective**).
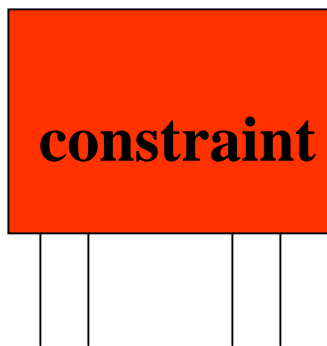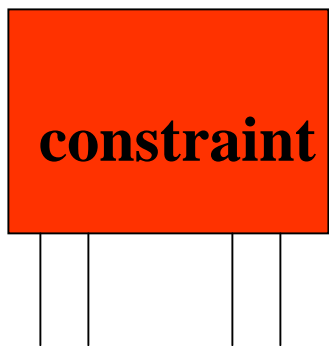
A	B	X	Y
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

Think about a gate as an input/output constraint

A	B	X	Y
0	0	0	0
0	1	0	1
1	0	1	1
1	1	1	0

A $\longrightarrow$ reversible $\longrightarrow$ A=**X**

B $\longrightarrow$ $\longrightarrow$ A XOR B=**Y**

R(input, output) = R(<A,B>,<X,Y>)
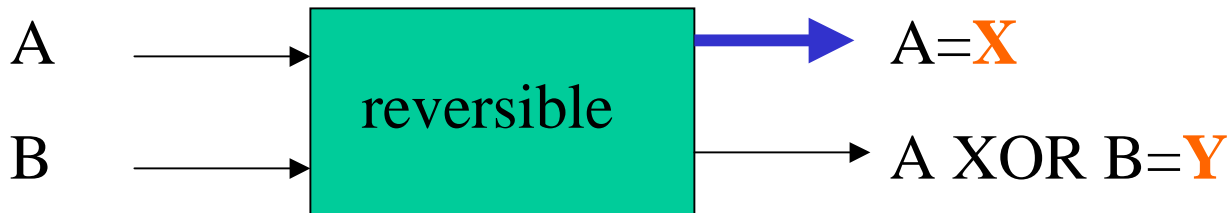
R(<A,B>,<X,Y>) = Permute(2,3)

constraint

constraint
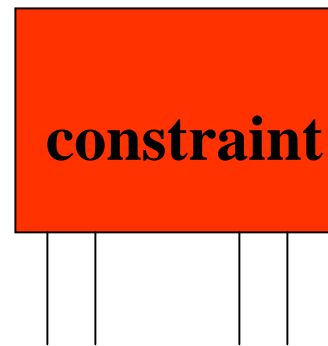
constraint

<0,0> ==> <0,0>

<1,0> <== <1,1>

<<1,0> , <1,1>> YES

<<1,1> , <0,1>> NO

Think about a gate as an input/output constraint

constraint

equation

Physical process
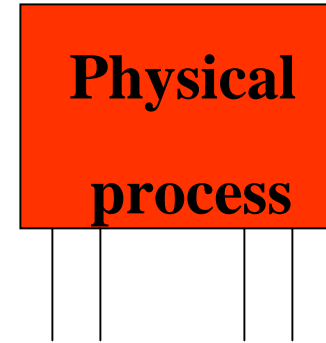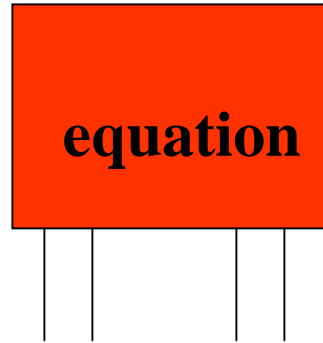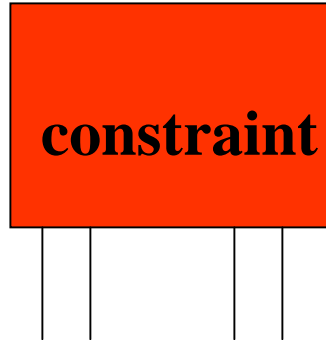
Few ways to think about reversible gates

Landauer Theorem

- Whenever we use a logically irreversible gate <u>we dissipate energy into the environment.</u>

- Reversible gate is a necessary but not sufficient condition of losing no power

- In addition, we need at least an ideal switch.
- Can ideal switch be build?

A ———→ reversible ——→ A
B ———→   ——→ A XOR B
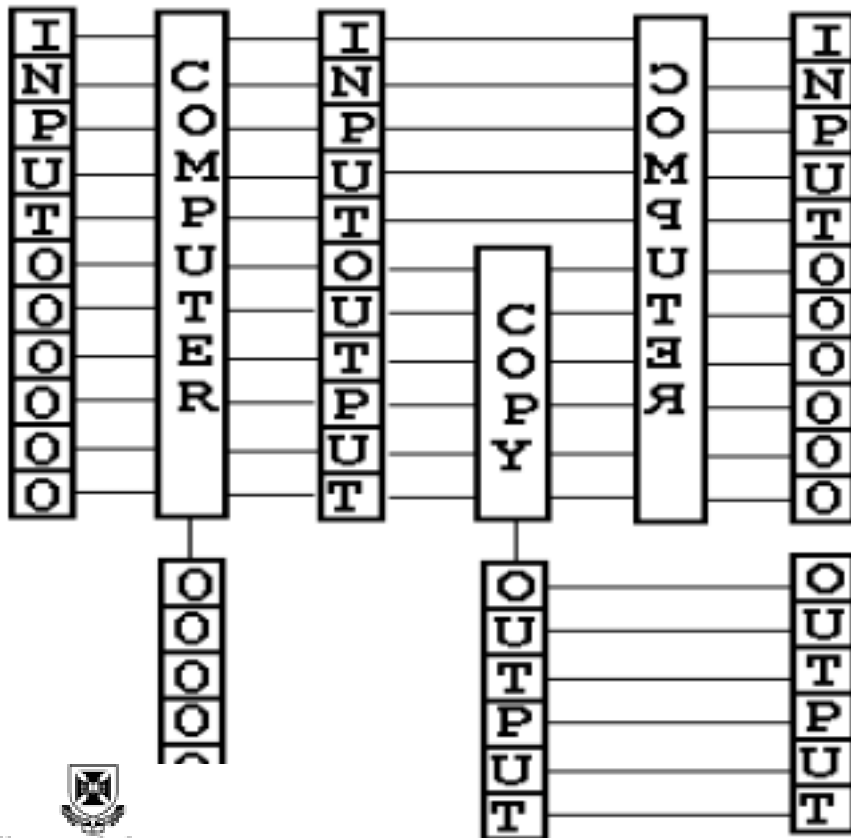
Information is Physical

- **Charles Bennett, 1973:**

- There are no **unavoidable** energy consumption requirements per step in a computer.

- Power dissipation of *reversible* computer, under ideal physical circumstances, is **zero**.

Reversible computation.

- **Charles Bennett, IBM, 1973.**
  - *Logical reversibility of computation,*
    - **IBM J. Res. Dev. 17, 525 (1973).**

- This principle applies also to combinational circuits that we build.

- But is this a best way?

- No, this is extremely wasteful

- No principles of logic synthesis should be invented

Landauer's principle

- *Landauer's principle:* **logic computations that are not reversible, necessarily generate heat:**

  - **i.e. kTlog(2), for every bit of information that is lost.**

    where k is Boltzmann's constant and T the temperature.

- For T equal room temperature, this package of heat is small, i.e. 2.9 x 10 $^{-21}$ joule, but non-negligible.

- In order to produce zero heat, a computer is only allowed to perform ***reversible computations.***

- Such a logically reversible computation can be `undone': the value of the output suffices to recover what the value of the input `has been'.

- The hardware of a reversible computer **cannot** be constructed from the conventional gates

- On the contrary, it consists ***exclusively of logically reversible*** building blocks.

First Example: Toffoli's Gate

- Tomasso Toffoli, 1980: There exists a reversible gate which could play a role of a universal gate for reversible circuits.
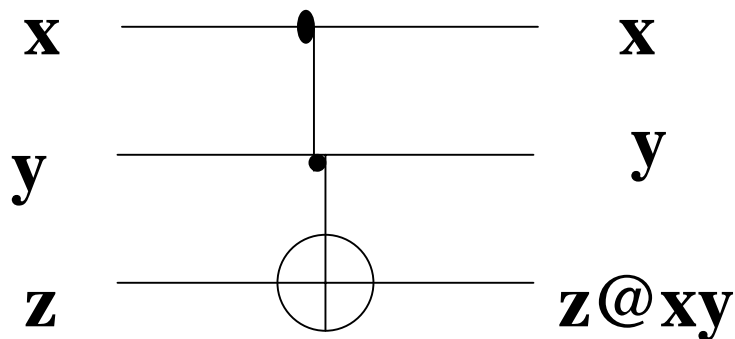
$$Q^{(3)} : (x,y,z) ==> (x,y,z@xy)$$

@ denotes EXOR

x   x

y   y

z   z@xy
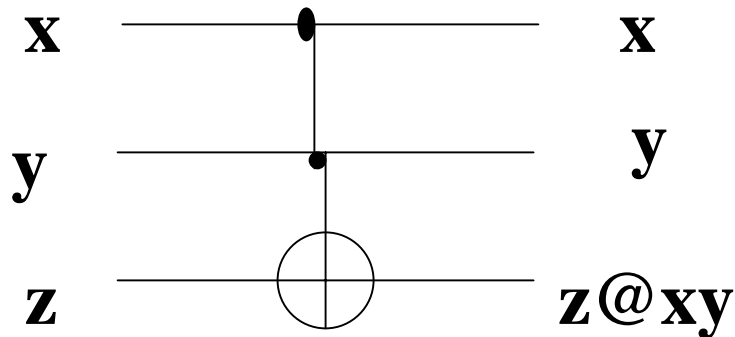
Fredkin and Toffoli created the first (3,3) universal gate

A → reversible → A

B →   → A XOR B

Toffoli Gate

Non-linear Gate: Toffoli

- **Important example of non-linear gate:**
- **3-bit Toffoli gate, (3,3) Toffoli gate**
- **Called also controlled-controlled-NOT**

x ———————— x

y ———————— y

z ———————— z@xy

We already introduced this gate earlier, now will be more detail

It flips the third bit if the first two bits are 1 and does nothing else

Like the XOR gate, it is its own inverse

- Of course, a computer built of only this type of gate, even using quantum technology, would have the power of only classical computers

But we can in addition to these gates add gates that are possible only in quantum

$$x \quad\quad\quad\quad\quad\quad x$$
$$y \quad\quad\quad\quad\quad\quad y$$
$$z \quad\quad\quad\quad\quad z@xy$$
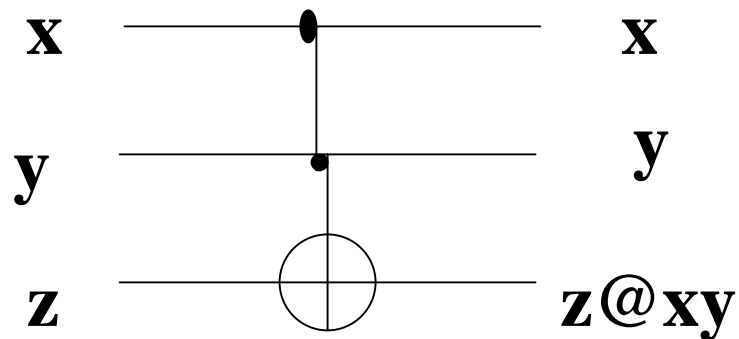
Sufficient motivation is also that these gates can be realized in many other technologies that already exist

What other Reversible Gates can exist?

Width of Reversible Gate

- The number of output bits of a reversible logic gate necessarily _equals its number of input bits._

- We will call this number the `*logic width'* of the gate.

Quantum Reversible Gates

- **In designing gates for a quantum computer, certain <u>constraints</u> must be satisfied.**

- **In particular, the *matrix of transition amplitudes* must be unitary, which implies, roughly speaking, that it <u>conserves probability</u>:**

- **The sum of the probabilities of all possible outcomes must be exactly 1.**

- **A consequence of this requirement is that any quantum computing operation must be reversible:**

Quantum Reversible Gates

- **quantum computing operation must be reversible**:
  - You must be able to take the results of an operation and put them back through the machine in the opposite direction to recover the original inputs.
- Standard gates do not obey this rule, since information is irretrievably lost when two input bits are condensed into a single output bit.

Universal Classical and Quantum Reversible Gates

- The study of reversible computing has gotten a lot of attention lately.

- This is because of quantum computing but also because it was found that reversible computation can be done also in CMOS, optically and in nano-technologies.

- It consumes very little power, and power reduction is becoming the most important design objective of computers.

  - a reversible computer can perform any computation and can do so with arbitrarily low energy consumption (Charles H. Bennett and Rolf Landauer of IBM)

Universal <u>Classical</u> and <u>Quantum</u> *Reversible Gates*

- A reversible (3,3) gate devised by Tommaso Toffoli of MIT is a "universal" classical gate:

- A computer could be built out of copies of this gate alone

- **Deutsch has shown that a similar gate is universal for quantum computers**

- Both the Toffoli and the Deutsch gates have three inputs and three outputs, but more recently two-qubit (2,2) gates have also proved universal for **quantum computations**

- <u>I believe that (2,2) gates can be constructed in MVL and I am working on this.</u>

Quantum Dots

- Practical *quantum technologies* are years or decades away.

- Few implementation schemes are already under discussion.

- The idea closest to existing electronic technology relies on

- "Quantum dots" are closest to existing electronic technologies

- We will discuss them in one of next lectures

  - They are isolated conductive regions within a semiconductor substrate.

  - Each quantum dot can hold a *single electron*, whose presence or absence represents one qubit of information. (qubit is quantum bit, it will be much more on it).

Another Quantum Circuits

- *Hypothetical polymeric molecule* in which the individual subunits could be toggled between the ground state and an excited state

- **David P. DiVincenzo** of IBM has described a mechanism by which <u>isolated nuclear spins would interact</u> -- and thereby compute -- when they are brought together by the *meshing of microscopic gears*

- *Others* will be also discussed

•Reversible Logic reversible gate invertible function

Terminologies are not yet consistent, be aware

Linear Reversible Gates

Second Example: Feynman Gate

Quantum XOR

- **Controlled NOT**
- **Quantum XOR**
- **Reversible XOR**
- **QCF**
- **Feynman gate**

Quantum XOR

- How do we build up a complicated reversible computation from elementary components -

  - what constitutes a universal gate set?

- We will see that **one-bit and two-bit reversible gates do not suffice**; we will need three-bit gates for universal reversible computation.

- Of the *four 1-bit --> 1-bit gates*, two are reversible; the trivial gate and the NOT gate.

- Of the $(2^4)^2 =$ **256 possible 2-bit--> 2-bit gates**, 4! = 24 are reversible.

Quantum **XOR**

- Of special interest is the <u>controlled-NOT</u> or **reversible XOR** gate:
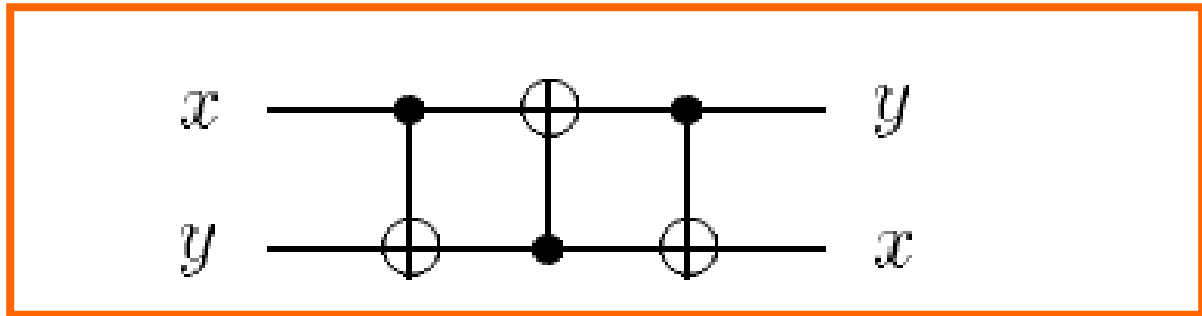
$$XOR : (x; y) \implies (x, x @ y);$$

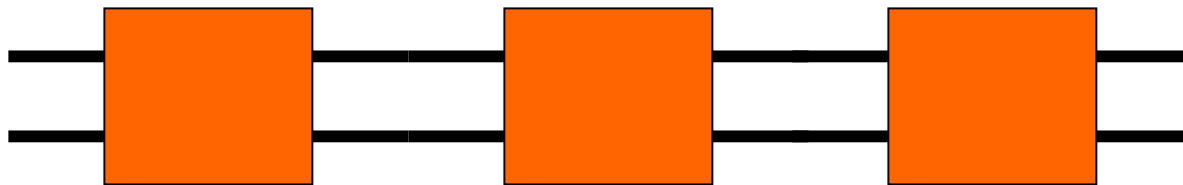by @ we denote EXOR (modulo-2 sum)
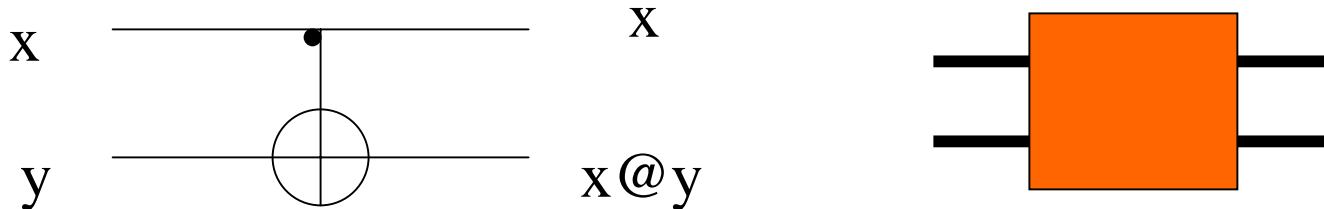
Swapping bits using XOR cascade

With the circuit

Constructed from three Quantum XORs, we can swap two bits:
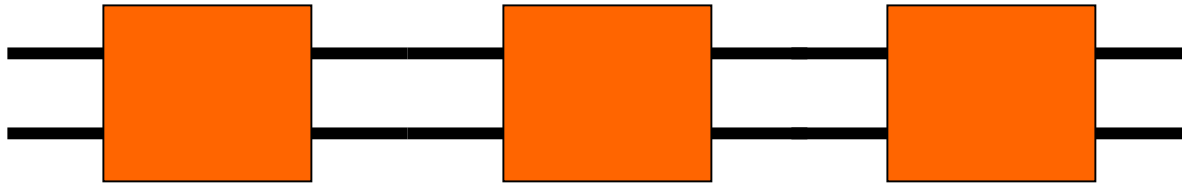
(x,y) ==> (x,x@y) ==> (y,x@y) ==> (y,x)

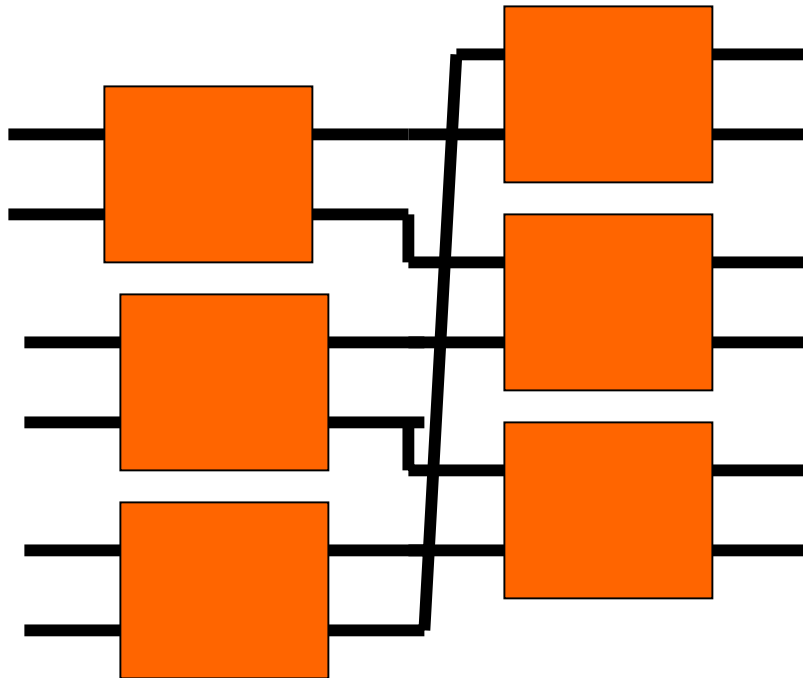*Conclusion: in quantum logic you pay for crossing wires!!!*

cascade

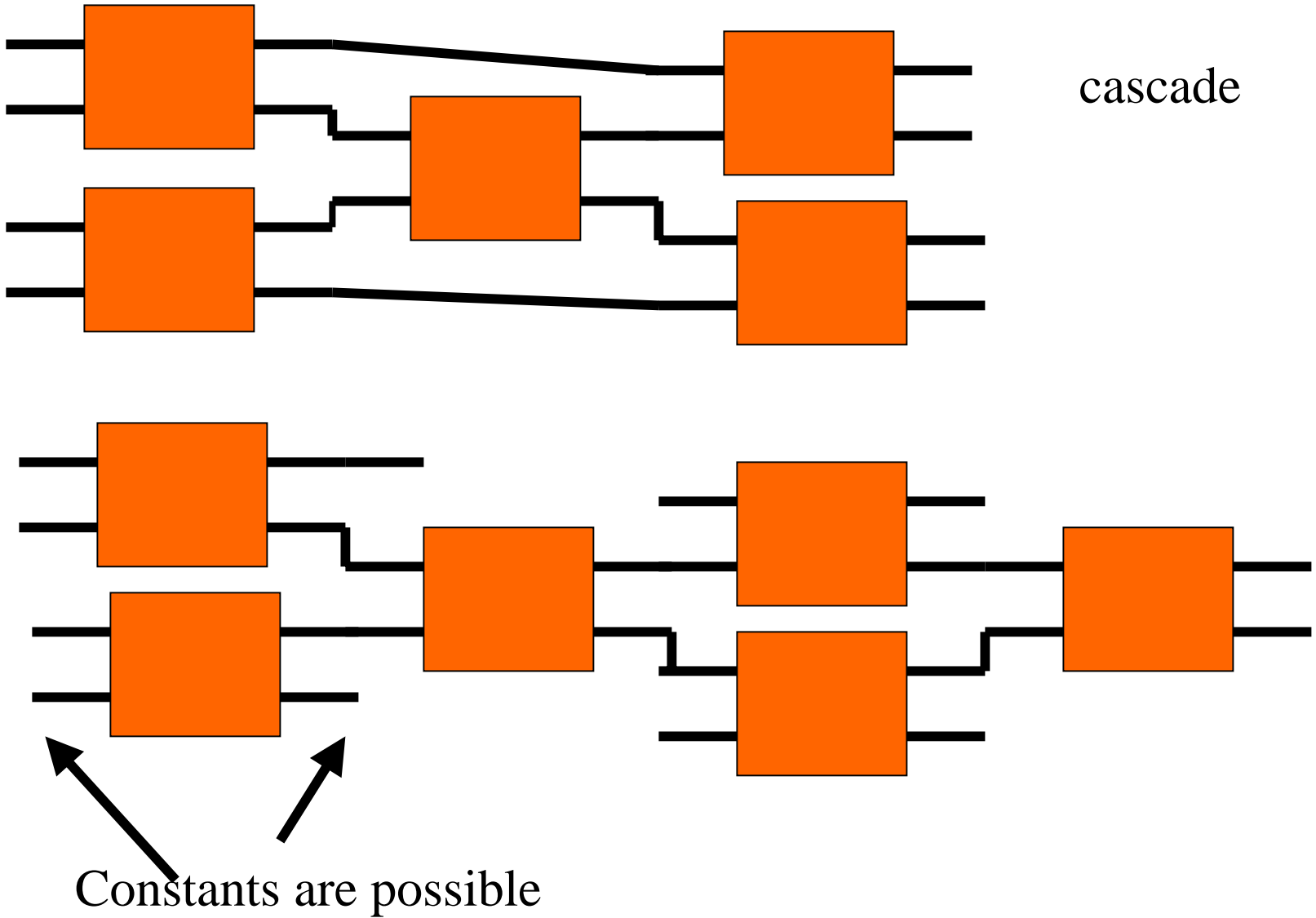**Of importance in quantum, quantum dot, but not CMOS**

Structures

cascade

Only linear circuits with linear gates

No branchings!!

Structures

cascade

Constants are possible

Rules for creating Structures

- **Do not generate the waste of outputs**

- **Create constants rather than functions on outputs**

- **Create re-usable functions (repeating inputs by branching may be useful)**

The QCF gate is not complete

- The QCF gate is not enough to build a complete quantum computer
  - like NOT gates are not enough to build a classical computer.

- Performing useful calculations requires gates that process more than one bit (or qubit in case of quantum logic) at a time.

- For example, AND gates in conventional computers.

Linear Gates

- To prove that one-bit and two-bit gates are not universal, it can be observed that they are linear.

- Composition of linear gates is always linear, and there exist of course non-linear functions (by linear we mean a circuit that satisfies superposition, as in control theory)

- Each reversible two-bit gate has an action of the form:

$$\begin{pmatrix} x \\ y \end{pmatrix} \rightarrow \begin{pmatrix} x' \\ y' \end{pmatrix} = M \begin{pmatrix} x \\ y \end{pmatrix} + \begin{pmatrix} a \\ b \end{pmatrix}$$

Linear Gates

Where the constant $\begin{pmatrix} a \\ b \end{pmatrix}$

takes one of four possible values, and the matrix M is one of the six invertible matrices

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

All additions are performed in modulo-2 (EXOR)

Combining the six choices for M with the four possible constants,

we obtain 24 distinct gates, which exhausts all the reversible

2==> 2 gates

Linear Gates

- Since the linear transformations are closed under composition, any circuit composed from reversible 2 ==> 2 and 1 ==> 1 gates will compute a linear function **x ==> Mx+a**

- **As shown by Toffoli gate (controlled-controlled-Not , Q $^{(3)}$) there are (3,3) gates that are non-linear.**

- **We will investigate such gates for n>= 3**

- **Linear gates will still remain very important because of their special properties.**

- **EXOR forests used in BIST are linear circuits, they can be modified to reversible logic.**

Gates of logic width 3

- *Fredkin* and *Toffoli*:

  - demonstrated that three inputs and three outputs is <u>necessary and sufficient</u> in order to construct a *reversible implementation* of an <u>arbitrary boolean function</u> of a finite number of logic variables.

- Thus, from the fundamental point of view, <u>**reversible logic gates with a width equal to three have a privileged position**</u>.

How many reversible gates exists?

- The truth table of a logic gate of width w consists of 2^w lines, each containing two **w-bit** numbers: the w-bit input ($A, B, C, ...$) and the w-bit output ($P, Q, R, ...$).

  - For convenience, all possible inputs, ranging from (0, 0, 0, ...) to (1, 1, 1, ...), are ordered arithmetically.

- Such a gate is reversible if-and-only-if all 2^w output numbers **form a permutation of the 2^w input numbers.**

- Hence, there exist **exactly $(2^w)!$ different reversible gates of width** w.

How many reversible (3,3) gates exists?

- In particular, there are $8! = 40{,}320$ reversible gates with *3-bit width*.

- We will investigate which of these 40,320 gates fulfil the role of **universal building block**, and which fulfil this job **more efficiently** than the others.

- In order to tackle the problem, we will successively study the reversible gates with $w = 1$, $w = 2$, **and** $w = 3$.

Single bit Reversible Gates

- There exist only four different truth tables with one bit input and one bit output.

- Two of them are logically irreversible: the **resetter** ($P = 0$) and the **setter** ($P = 1$).

- The two others are reversible: the follower ($P = A$) and the inverter ($P = \texttt{NOT } A$).

  - If, for example, we have `forgotten' the value of A, knowledge of the value of the inverter's output P suffices to recover it.

Single bit Reversible Gates

- Note that among the 1-bit reversible gates, the NOT gate is a `generator'.

  – This means we can make any reversible gate of width 1 by combining a finite number of this particular gate.

- Indeed, a follower can be fabricated by the sequence of two inverters.

- The opposite is not true: one cannot fabricate an inverter by cascading followers.
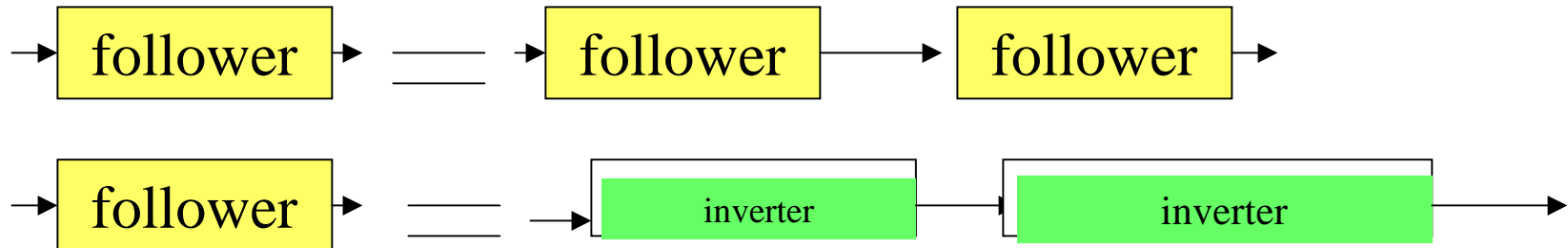
Reversible Gates with two bits

- There are $4^4 = 256$ different truth tables with two inputs (A, B) and two outputs (P, Q).

- Among them, only $4! = 24$ are reversible.

- However, some of these twenty-four truth tables fall apart into two separate 1-bit reversible tables.

  – **Table a** decomposes into one follower $Q = A$ (Table 2b) and one inverter P = NOT B (**Table c**).

AB	PQ
0 0	1 0
0 1	0 0
1 0	1 1
1 1	0 1

(a)

=

A	Q
0	0
1	1

(b)

&

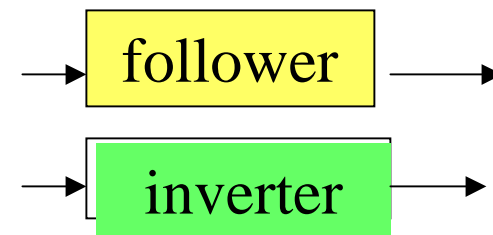| B	P
0	1
1	0

(c)

follower

inverter

Table : Falling apart of a truth table.

Calculation with two bits

- On the contrary, truth Table GATES b is an example of a 2-bit reversible table that cannot be reduced to two separate 1-bit reversible tables, and therefore is called a true two-bit reversible gate.

- Among the 24 reversible 2-bit tables, only 16 are true 2-bit tables.

A	P
0	1
1	0

(a)

AB	PQ
0 0	0 0
0 1	0 1
1 0	1 1
1 1	1 0

(b)

A B C	P Q R
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

(c)

P=A

Q=A@B

Table GATES: Feynman's truth tables: (a) NOT, (b) CONTROLLED NOT, (c) CONTROLLED CONTROLLED NOT.

CONTROLLED NOT by Feynman

- **All reversible true 2-bit gates can be fabricated from the same building block, combined with an inverter before and/or an inverter after.**

- **Indeed, Table 3b together with the inverter (Table 3a) forms a set of two building blocks with which we can synthetize an arbitrary reversible 2-bit gate.**

- **Truth Table 3b is called the CONTROLLED NOT by Feynman**

- **Its logic operation looks like this:**

  $$P = A$$

  $$Q = A \text{ XOR } B,$$

  where XOR is the abbreviation of the EXCLUSIVE OR function.

- **The gate is the reversible form of the classical (irreversible) XOR gate.**

A	P
0	1
1	0

(a)

AB	PQ
0 0	0 0
0 1	0 1
1 0	1 1
1 1	1 0

(b)

A B C	P Q R
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

(c)

Table GATES: Feynman's truth tables: (a) NOT, (b) CONTROLLED NOT, (c) CONTROLLED CONTROLLED NOT.

The synthesis of a 2-bit reversible gate:
(a) truth table, (b) three different implementations combining `NOTs` and `CONTROLLED NOT`.
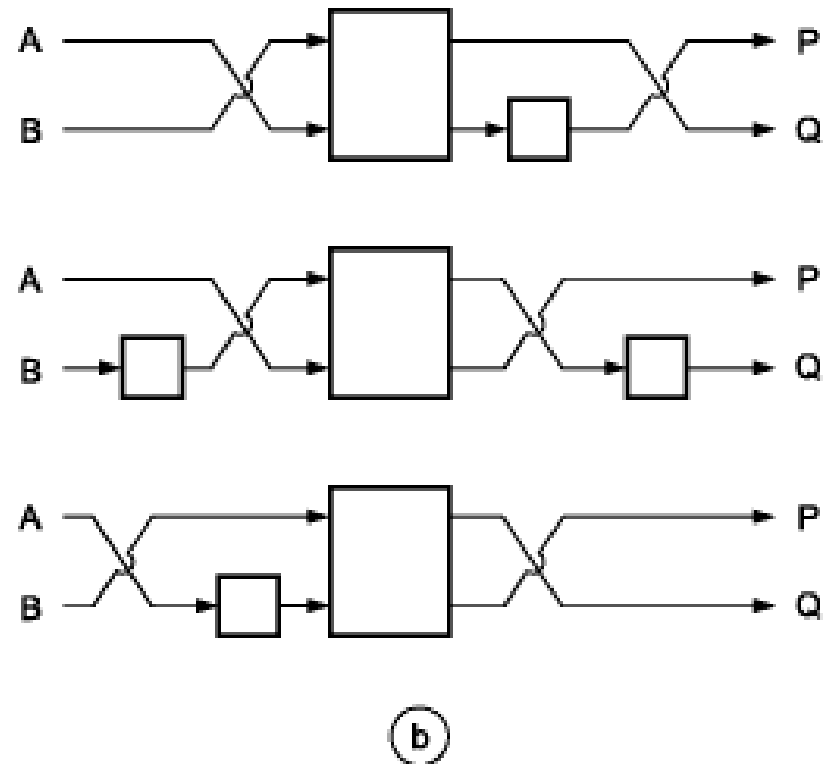
- Figure gives a representative example of a 2-bit reversible gate, realized by combining NOT and CONTROLLED NOT gates.

- Whereas output Q simply equals input B, output P can be described in three different ways:

- $P = $ NOT $(A$ XOR $B)$

- $P = A$ XOR $($NOT $B)$

- $P = ($NOT $A)$ XOR B .

AB	PQ
0 0	1 0
0 1	0 1
1 0	0 0
1 1	1 1

(a)

(b)

Controlled NOT gate

- These three boolean expressions are identical, but lead to different physical realizations.

- We note, however, that these implementations not only make use of the 1-bit NOT function and the 2-bit CONTROLLED NOT function, but also of the 2-bit exchanger, i.e. the gate that interchanges two logic variables (realizing $P = B$ as well as $Q = A$).

- This is an example of a general property: the EXCHANGER, the NOT, and the CONTROLLED NOT form a natural `generating set' for the twenty-four 2-bit reversible gates.

- More precisely, each reversible 2-bit gate can be synthesized by taking one or zero CONTROLLED NOTs and adding one or zero EXCHANGERs and one or zero NOTs to the left and to the right of it.

- Note that:

  - neither the EXCHANGER nor the NOT is a true 2-bit gate,

  - but the CONTROLLED NOT is one.

- See Table 4

AB	PQ
0 0	0 0
0 1	1 0
1 0	0 1
1 1	1 1

(a)

AB	PQ
0 0	0 1
0 1	0 0
1 0	1 1
1 1	1 0

(b)

AB	PQ
0 0	0 0
0 1	0 1
1 0	1 1
1 1	1 0

(c)

Table 4: The three generators of the 2-bit reversible gates: (a) EXCHANGER, (b) NOT, (c) CONTROLLED NOT.

Example from Literature

- Reversible half and full adders as given by De Vos (1999). We use the following gates:
  - The **CONTROLLED NOT** described by:

  P = A and Q=A XOR B

  - The **CONTROLLED CONTROLLED NOT** described by:

  P = A, Q = B, and R = (A AND B) XOR C

Implications of Reversibility: Additional Inputs

- Reversibility criteria may require us to add inputs to a given function. Take a 2-input OR function:

A	B	F
0	0	0
0	1	1
1	0	1
1	1	1

→

A	B	C	F	G	H
0	0	0	0	0	0
0	1	0	1	0	1
1	0	0	1	1	0
1	1	0	1	1	1
+	+	+	+	+	+(?)

  – Note that we have some degrees of freedom in how we fill out the columns C, G, and H

Implications of Reversibility: Additional Inputs

- Intuitively, the need for extra inputs has to do with the number of original inputs as well as the degree of balance of the output function(s) (ratio of 0s and 1s).

- It would be good to derive the mathematical basis of this property.

Implications of Reversibility: Additional Inputs

- For example, a 2-input XOR can be made reversible by simply adding an output to the original gate

A	B	F
0	0	0
0	1	1
1	0	1
1	1	0

→

A	B	F	G
0	0	0	0
0	1	1	0
1	0	1	1
1	1	0	1

Implications of Reversibility: Additional Inputs

- All n-input XOR functions can be made reversible by adding n-1 outputs.

- Linearly Independent gates such as Galois Addition or MODSUM have the same property

- Similarly, it seems that reversible versions of OR and AND gates always require us to add one input to the original list of inputs. But more must be done to create such reversible gates

- Negations of inputs or outputs simply permute the rows (so analysis for NAND gates is same as for AND gates)

Implications of Reversibility: Additional Inputs

- Conclusion:
  - We may need to add a number of inputs and outputs to a given specification of a function to make it reversible.

  - I would like to be able to characterize the conditions under which this is needed and the nature of what we're adding.

  - This problem may already have been tackled elsewhere in Mathematics (how to turn a non-invertible function into an invertible one)

Implications of Reversibility: Additional Inputs

- **Completing** a given specification for a function with additional inputs and/or outputs may constitute the first step of our quest to synthesize an arbitrary function into a reversible circuit.

Implications of Reversibility: Fan-out operation

- In regular binary logic, we cannot "combine" two wires into one.
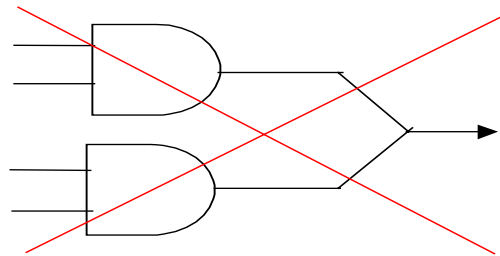
- Since we want our circuits to be reversible, any branching of a signal looked at in reverse will appear as combining signals.

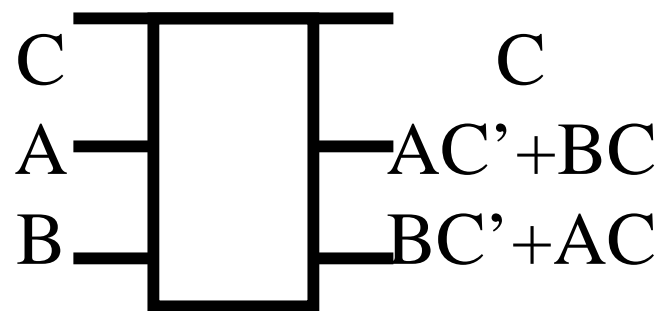# Implications of Reversibility: Fan-out operation

- Moreover, as Fredkin and Tofolli (1981) point out, duplication of signals from a physical viewpoint is far from trivial.

- We will therefore need to use gates anytime we wish to duplicate a signal.

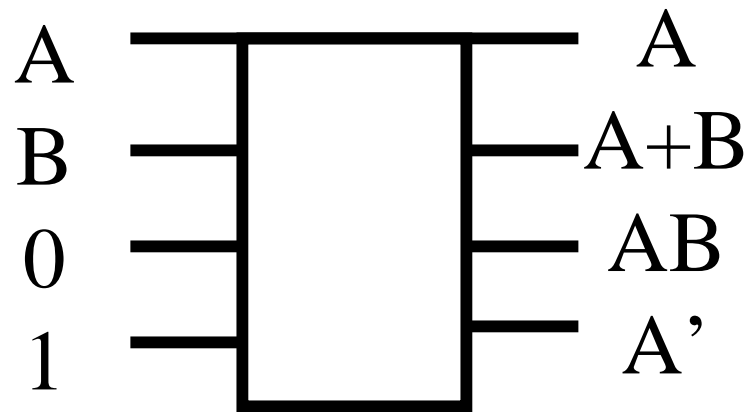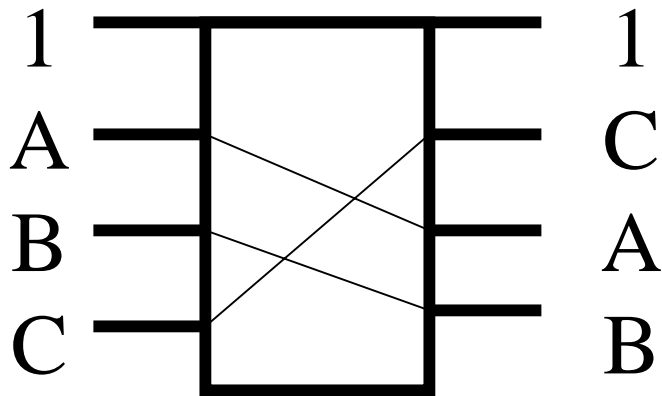- In minimizing a given function reversibly, we cannot assume free fan-out.
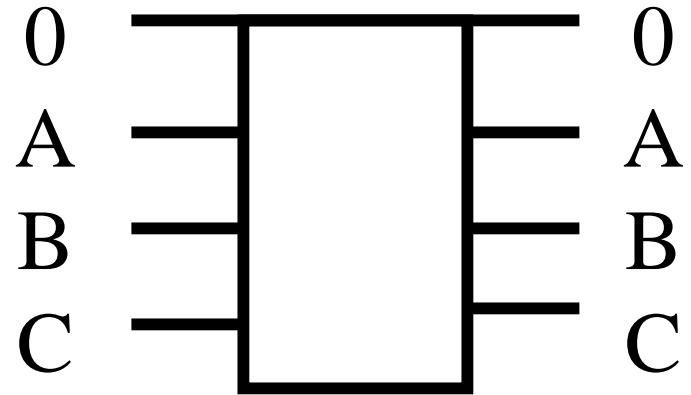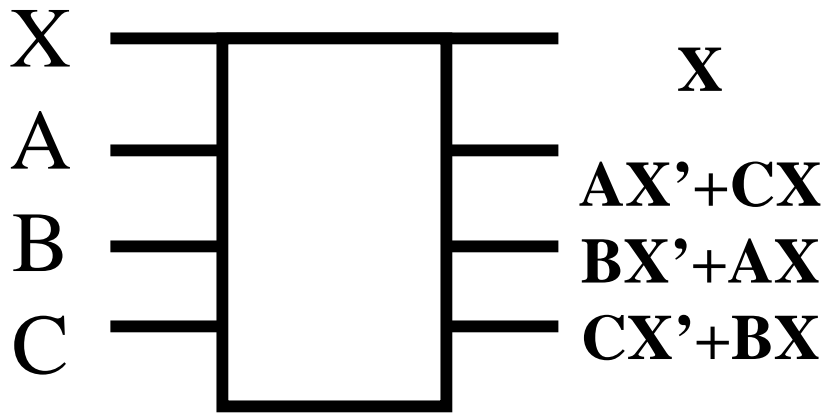
Third Example: Fredkin's Gate

Operation of the Fredkin gate

A 4-input Fredkin gate

Input	Output
X	X
A	AX'+CX
B	BX'+AX
C	CX'+BX

Input	Output
0	0
A	A
B	B
C	C

Input	Output
1	1
A	C
B	A
C	B

Input	Output
A	A
B	A+B
0	AB
1	A'
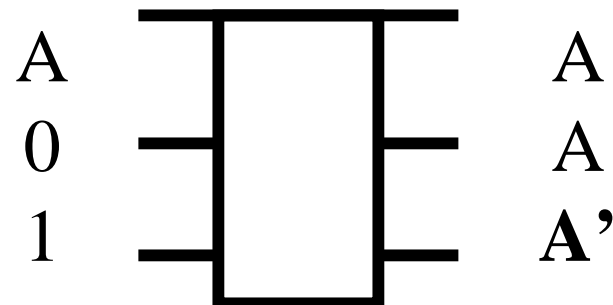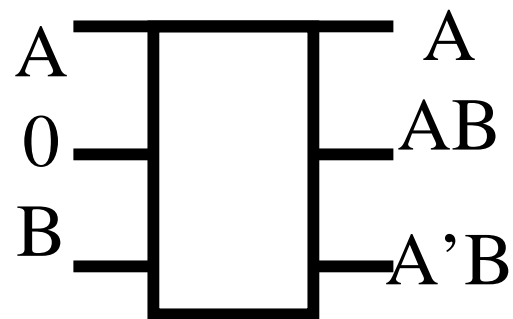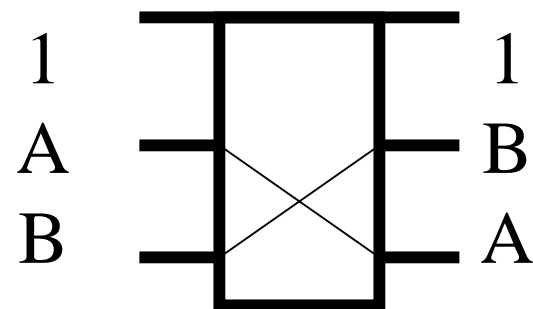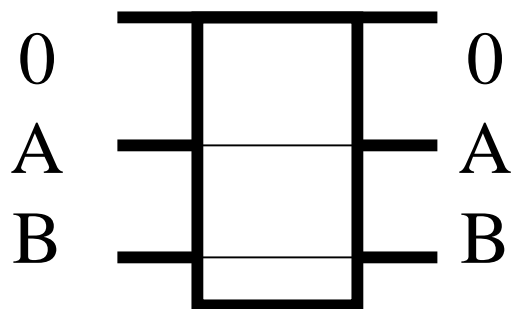
Towards Designing New Reversible Gates