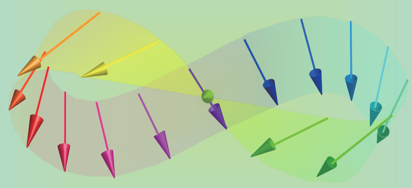
Information Storage and Spintronics 02



Atsufumi Hirohata

Department of Electronic Engineering

THE UNIVERSITY of York

14:00 Thursday, 06/October/2022 (SLB 101)

Contents of Information Storage and Spintronics

Lectures : Atsufumi Hirohata (atsufumi.hirohata@york.ac.uk, P/Z 019)

Advancement in information storages and spintronics (Weeks 2 ~ 9)

All lectures will be uploaded weekly in advance at

http://www-users.york.ac.uk/~ah566/lectures/lectures.html

14:00 ~ 15:00 Mons. (SLB 101)

14:00 ~ 15:00 Thus. (SLB 101)

I. Introduction to information storage (01 & 02)

II. Magnetic information storages (03 ~ 06)

III. Solid-state information storages (07 ~ 11)

IV. Spintronic devices (12 ~ 18)

Practicals :

Analysis on a spintronic device using VSM, EDX, MFM and MR (Weeks 3 ~ 8)

Operation, data and instruction will be uploaded weekly in advance at

Internal Wiki page &

http://www-users.york.ac.uk/~ah566/lectures/lectures.html

13:00 ~ 15:00 Weds. (P/A 016, Nanocentre and P/Z 008)

Continuous Assessment :

Assignment to be submitted via VLE (Week 10).

Quick Review over the Last Lecture

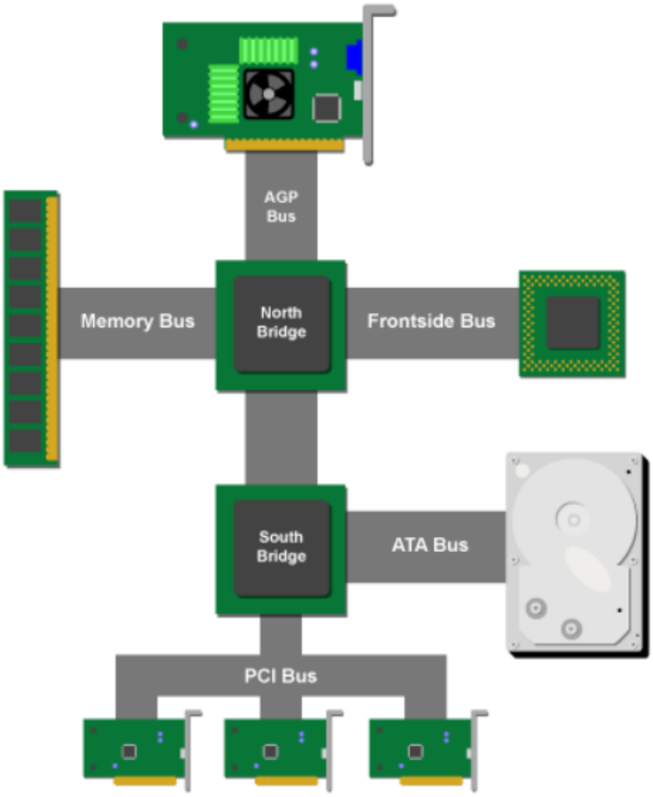
Von Neumann's model :

-
-
-
-
-

Bit / byte :

- 1 bit :
  - 2 = combinations
  - digit in binary number
- 1 byte (B) = bit

Memory access :



* http://testbench.in/introduction_to_pci_express.html;

02 Binary Data

- Binary numbers
- Conversion
- Advantages
- Logical conjunctions
- Adders
- Subtractors

Bit and Byte

Bit :

"Binary digit" is a basic data size in information storage.

1 bit : 2 = combinations ; digit in binary number

2 2 =

3 2 =

4 2 =

: : : :

Byte :

A data unit to represent one letter in Latin character set.

1 byte (B) = bit

1 kB = 1 B ×

1 MB = 1 kB ×

: :

Binary Numbers

The modern binary number system was discovered by Gottfried Leibniz in 1679 : *

Decimal notation	Binary notation
0	
1	
2	
3	
4	
5	
6	
7	
8	
:	:

* http://www.oracle.com/

Conversion to Binary Numbers 1

For example, 1192 :

2) 1192	$1192 = 2^0 \times$
2) 586…0	$1192 = 2^1 \times$
2) 293…0	$1192 = 2^2 \times$
2) 146…1	$1192 = 2^3 \times$
2) 73…0	$1192 = 2^4 \times$
2) 36…1	$1192 = 2^5 \times$
2) 18…0	$1192 = 2^6 \times$
2) 9…0	$1192 = 2^7 \times$
2) 4…1	$1192 = 2^8 \times$
2) 2…0	$1192 = 2^9 \times$
2) 1…0	$1192 = 2^{10} \times$
2) 0…1	$1192_{10} = 10010010100_2$

Conversion to Binary Numbers 2

For example, 0.1 :

0.1

$0.1 \times 2 =$

$0.2 \times 2 =$

$0.4 \times 2 =$

$0.8 \times 2 =$

$0.6 \times 2 =$

$0.2 \times 2 =$

: :

$$0.1_{10} = 0.0\underline{0011}_2$$

Why Are Binary Numbers Used ?

In order to represent a number of “1192” by ON / OFF lamps :

Binary number : 10010010100_2 (digits =   lamps)

Decimal number : 1192_{10} (4 digits × = lamps)

Similarly, Ternary number : 1122011_3 (7 digits × = lamps)

1192 = 729 + 243 + 162 + 54 + 0 + 3 + 1

$= 3^6 \times 1 + 3^5 \times 1 + 3^4 \times 2 + 3^3 \times 2 + 3^2 \times 0 + 3^1 \times 1 + 3^0 \times 1$

Quaternary number : 112220_4 (6 digits × = lamps)

1192 = 1024 + 0 + 128 + 32 + 8 + 0

$= 4^5 \times 1 + 4^4 \times 1 + 4^3 \times 2 + 4^2 \times 2 + 4^1 \times 2 + 4^0 \times 0$

Binary numbers use the   !

Mathematical Explanation

In a base-*n* positional notation, a number *x* can be described as :

$x = n^y$ (y: number of digits for a very simple case)

In order to minimise the number of devices, *i.e.*, $n \times y$,

$\ln(x) =$

Here, $\ln(x)$ can be a constant C,

$C =$

$y =$

By substituting this relationship into $n \times y$,

$n \times y =$

To find the minimum of $n / \ln(n)$,

$[n / \ln(n)]' =$

Here, $[n / \ln(n)]' = 0$ requires

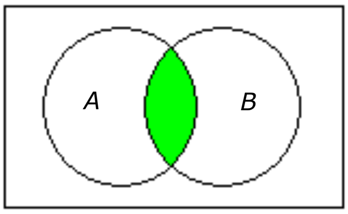
$= 0$

Therefore, $n =$ provides the minimum number of devices.

Logical Conjunctions 1

AND :

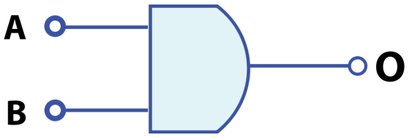
Venn diagram of $A \wedge B$



Truth table

Input		Output
A	*B*	$A \wedge B$
True (T) (1)	T (1)	
False (F) (0)	T (1)	
T (1)	F (0)	
F (0)	F (0)	

Logic circuit

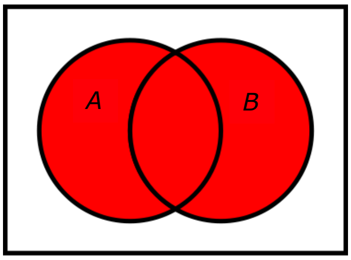


* http://www.wikipedia.org/

Logical Conjunctions 2

OR :

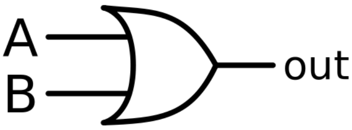
Venn diagram of $A \vee B$



Truth table

Input		Output
A	*B*	$A \vee B$
T (1)	T (1)	
F (0)	T (1)	
T (1)	F (0)	
F (0)	F (0)	

Logic circuit

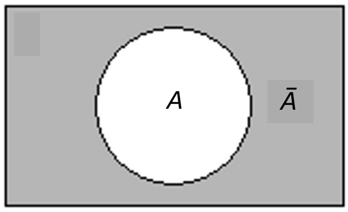


* http://www.wikipedia.org/

Logical Conjunctions 3

NOT :

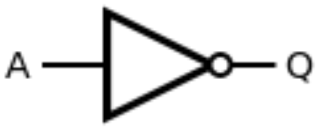
Venn diagram of $\bar{A}$



Truth table

Input	Output
A	$\bar{A}$
T (1)	
F (0)	

Logic circuit

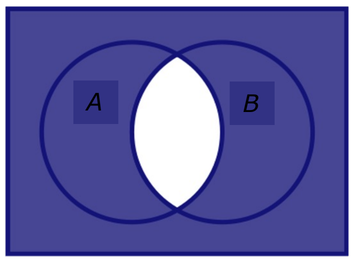


* http://www.wikipedia.org/

Additional Logical Conjunctions 1

NAND = (NOT *A*) OR (NOT *B*) = NOT (*A* AND *B*) :

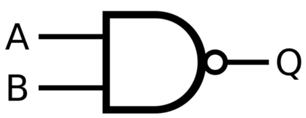
Venn diagram of $A \uparrow B$



Truth table

Input		Output
A	*B*	$A \uparrow B$
T (1)	T (1)	
F (0)	T (1)	
T (1)	F (0)	
F (0)	F (0)	

Logic circuit

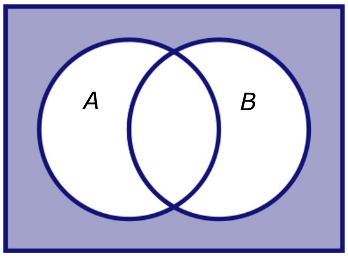


* http://www.wikipedia.org/

Additional Logical Conjunctions 2

NOR = NOT (*A* OR *B*) :

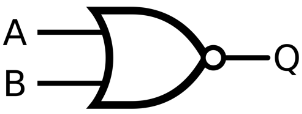
Venn diagram of *A*↓*B*



Truth table

Input		Output
A	*B*	*A*↓*B*
T (1)	T (1)	
F (0)	T (1)	
T (1)	F (0)	
F (0)	F (0)	

Logic circuit



NOR can represent all the logical conjunctions :

• NOT *A* =

• *A* AND *B* = =

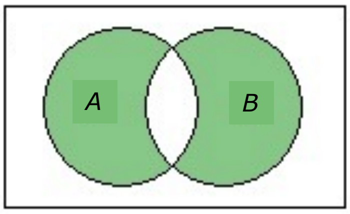
• *A* OR *B* = =

* http://www.wikipedia.org/

Additional Logical Conjunctions 3

XOR = Exclusive OR :

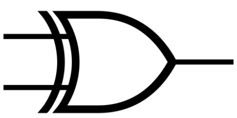
Venn diagram of *A*⊕*B*



Truth table

Input		Output
A	*B*	*A*⊕*B*
T (1)	T (1)	
F (0)	T (1)	
T (1)	F (0)	
F (0)	F (0)	

Logic circuit

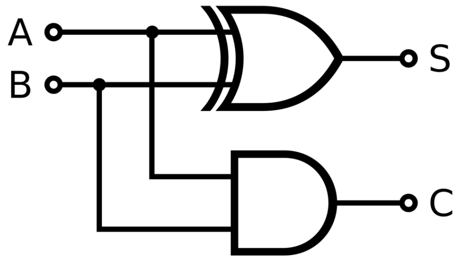


* http://www.wikipedia.org/

Half Adder

Simple adder for two single binary digits :

XOR for the sum (*S*)



AND for the carry (*C*), which represents the overflow for the next digit

Truth table

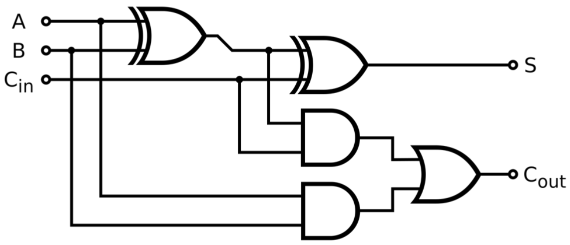
Input		Output	
A	*B*	*S*	*C*
1	1		
0	1		
1	0		
0	0		

* http://www.wikipedia.org/

Full Adder

Adder for two single binary digits as well as values carried in (C_{in}) :

2 half adders for sum (*S*)



Additional OR for the carry (C_{out}), which represents the overflow for the next digit

Truth table

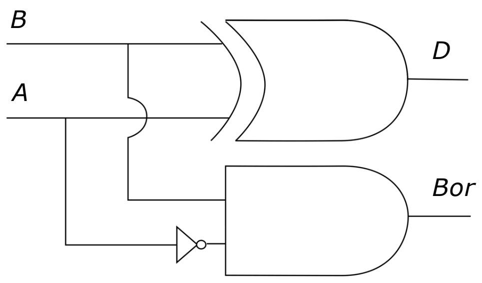
Input			Output	
A	*B*	C_{in}	*S*	C_{out}
1	1	1		
0	1	1		
1	0	1		
0	0	1		
1	1	0		
0	1	0		
1	0	0		
0	0	0		

* http://www.wikipedia.org/

Half Subtractor

Simple subtractor for two single binary digits, minuend (*A*) and subtrahend (*B*) :

XOR for the difference (*D*)



NOT and AND for the borrow (*Bor*), which is the borrow from the next digit

Truth table

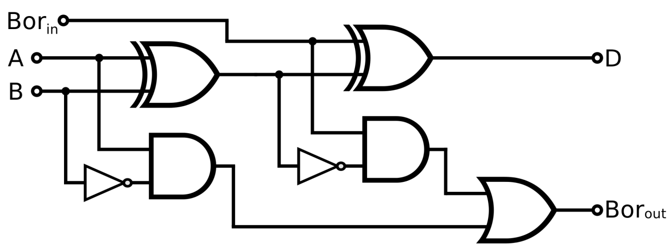
Input		Output	
A	*B*	*D*	*Bor*
1	1		
0	1		
1	0		
0	0		

* http://www.wikipedia.org/

Full Subtractor

Subtractor for two single binary digits as well as borrowed values carried in (Bor_{in}) :

2 half subtractor for difference (*D*)



Additional OR for the borrow (Bor_{out}), which is the borrow from the next digit

Truth table

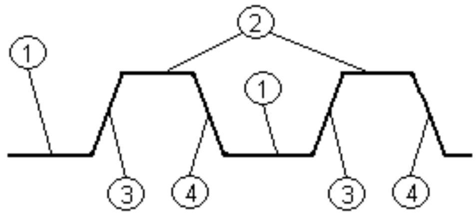
Input			Output	
A	*B*	Bor_{in}	*D*	Bor_{out}
1	1	1	1	1
0	1	1	0	1
1	0	1	0	0
0	0	1	1	1
1	1	0	0	0
0	1	0	1	1
1	0	0	1	0
0	0	0	0	0

* http://www.wikipedia.org/

Information Processing

For data processing, two distinct voltages are used to represent "1" and "0" :

Low level (1) and high level (2) voltages



Voltages used :

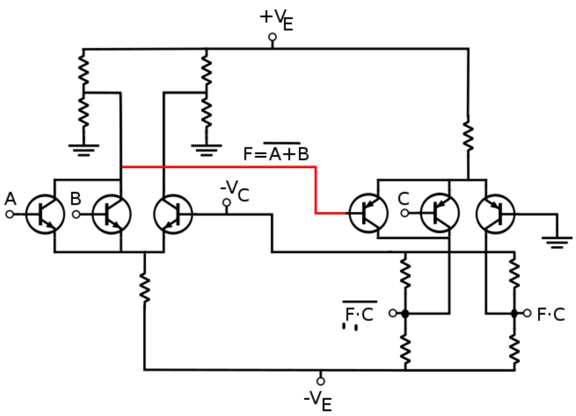
Devices	Low voltage	High voltage
Emitter-coupled logic (ECL)	- 5.2 ~ 1.175 V	0 ~ 0.75 V
Transistor-transistor logic (TTL)	0 ~ 0.8 V	2 ~ 4.75 (or 5.25) V
Complementary metal-oxide-semiconductor (CMOS)	0 ~ V_{DD} / 2	V_{DD} / 2 ~ V_{DD} (V_{DD} = 1.2, 1.8, 2.4, 3.3 V etc.)

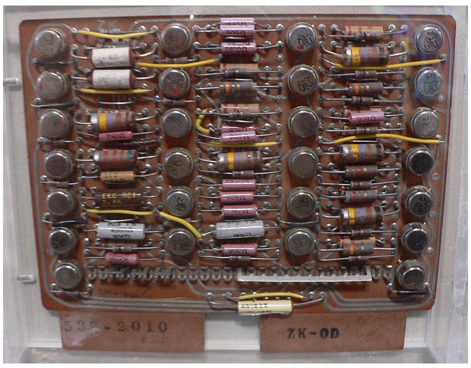
* http://www.wikipedia.org/

Emitter-Coupled Logic

In 1956, Hannon S. Yourke invented ECL at IBM : *

High-speed integrated circuit, differential amplifier, with bipolar transistors



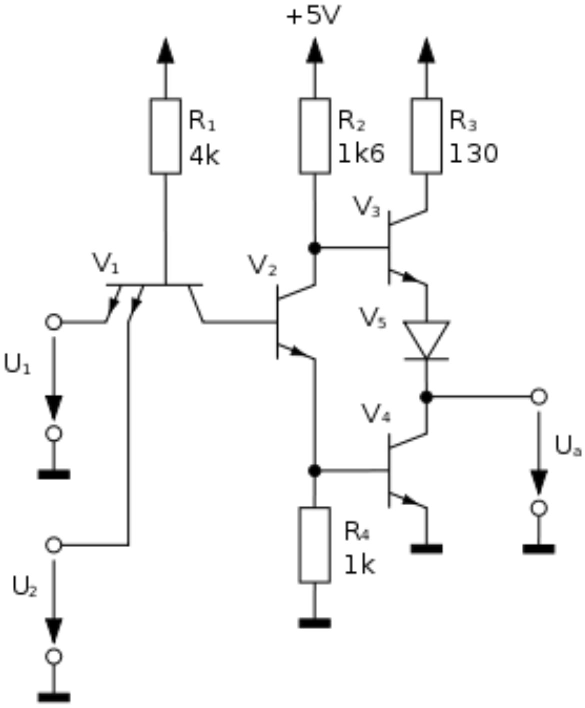


* http://www.wikipedia.org/

Transistor-Transistor Logic

In 1961, James L. Buie invented TTL at TRW : *

Integrated circuit, logic gate and amplifying functions with bipolar transistors

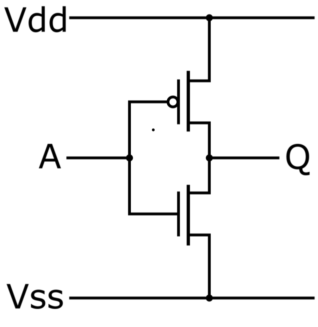


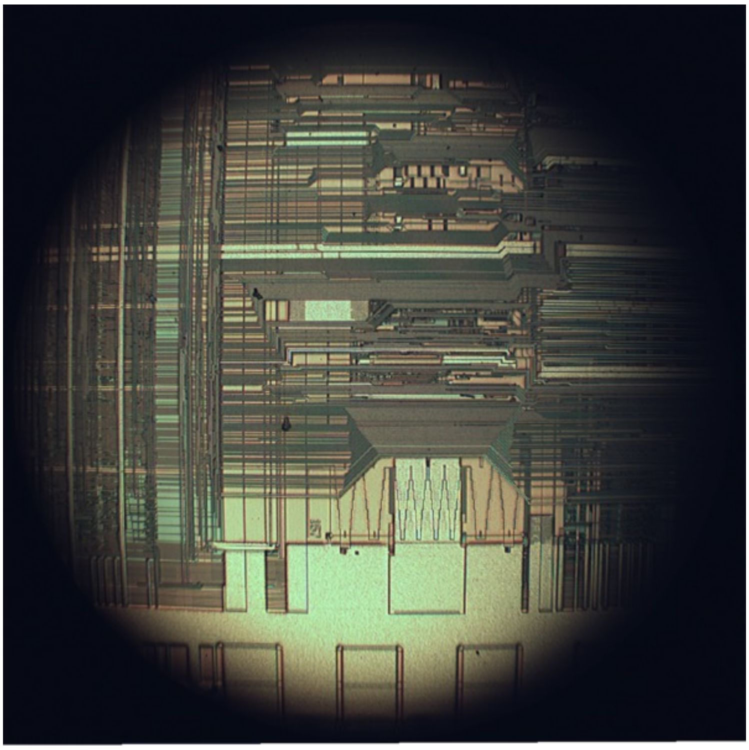
* http://www.wikipedia.org/

Complementary Metal-Oxide-Semiconductor

In 1963, Frank Wanlass patented CMOS : *

Integrated circuit with low power consumption using complementary MOSFET





* http://www.wikipedia.org/