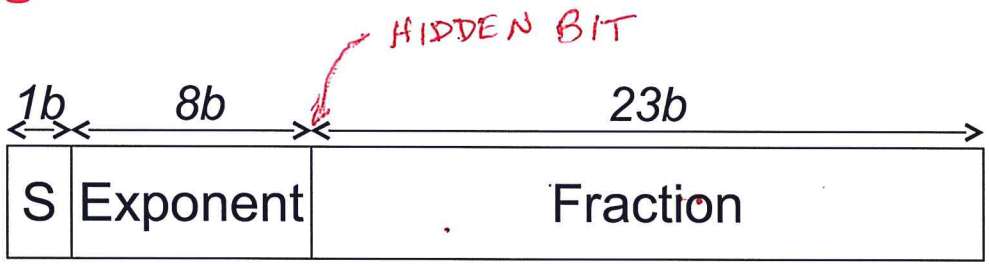
Digital Logic Structures – Chapter 3

Sept. 12, 2018

- Review of floating point
  - Examples
- ASCII – repesenting characters
- Logic operations
- Transistors – building blocks of computers
- Logic Gates
  - Building gates from transistors
  - Transforming gate types – deMorgan's law
- Commonly used logic blocks

IEEE 754 Floating Point Standard

Single Precision



$$N = (-1)^S \times 1.\text{fraction} \times 2^{\text{exponent}-127}, \quad 1 \le \text{exponent} \le 254$$

$$N = (-1)^S \times 0.\text{fraction} \times 2^{-126}, \quad \text{exponent} = 0$$

(0.375) 0.011 → NORMALIZE: 1.1×2^{-2}

$\times 2^0$

↑ HIDDEN

EXAMPLE

1 0.5

{1|01111110|1000|0000|0000|0000|0000|0000|0000}

SIGN = −1 → = 126

NUMBER = (126 − 127)

$$-1.5 \times 2^{-1}$$

$$= -0.75$$

$$-1.1_2 \times 2^{-1}$$

$$= -0.11$$

1/2 1/4

IEEE-754 Floating-Point Conversion

From Decimal Floating-Point
To 32-bit and 64-bit Hexadecimal Representations
Along with Their Binary Equivalents

Enter a decimal floating-point number here,
then click either the **Rounded** or the **Not Rounded** button.

Decimal Floating-Point: 2.325 Clear

Rounded Not Rounded

Rounding from floating-point to 32-bit representation uses the IEEE-754 round-to-nearest-value mode.

Results:

Decimal Value Entered: 2.325

Single precision (32 bits):

Binary: *Status:* normal

Bit 31 Sign Bit 0 0: + 1: -	Bits 30 - 23 Exponent Field 10000000 Decimal value of exponent field and exponent 128 - 127 = 1	Bits 22 - 0 Significand 1 .00101001100110011001101 Decimal value of the significand 1.1625000

Hexadecimal: 4014CCCD *Decimal:* 2.3250000

Double precision (64 bits):

Binary: *Status:* normal

Bit 63 Sign Bit 0 0: + 1: -	Bits 62 - 52 Exponent Field 10000000000 Decimal value of exponent field and exponent 1024 - 1023 = 1	Bits 51 - 0 Significand 1 .0010100110011001100110011001100110011001100110011001101 Decimal value of the significand 1.1625000000000000

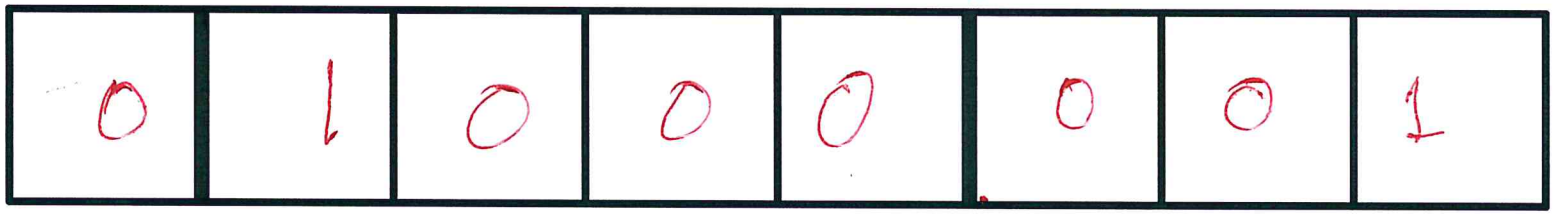
Hexadecimal: 400299999999999A *Decimal:* 2.3250000000000000

[Convert IEEE-754 32-bit Hexadecimal Representations to Decimal Floating-Point Numbers.]
[Convert IEEE-754 64-bit Hexadecimal Representations to Decimal Floating-Point Numbers.]
[Reference Material on the IEEE-754 Standard.]
[Dr. Vickery's Home Page.]

February 1998
This page was created by a Queens College undergraduate, Quanfei Wen, a member of PBK and UPE.

September 1998

Example of Floating Point



= 1 + 7 → BIAS

HIDDEN 1

REPRESENT +2.3

→ 10.0 0 0 1

Sign: 0 for positive
1 for negative

1.001 001 x 2^1

??

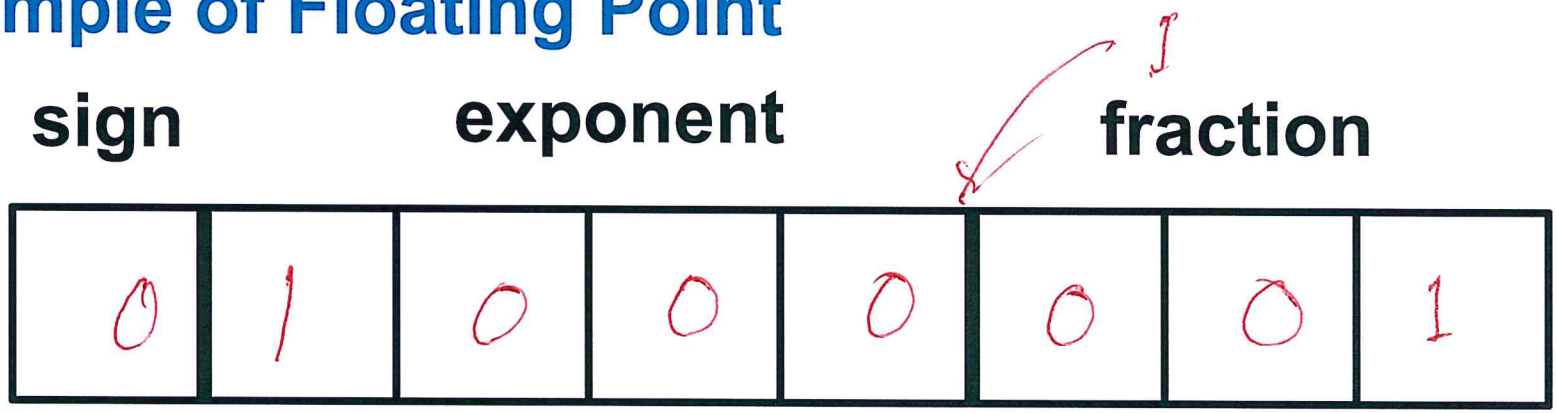
Exponent: bias or excess 7

Fraction: hidden "1"

X2

0.	3
0.	6
1.	2
0.	4
0.	8
1.	6

Example of Floating Point



= ?

EXP. = 2^1

FRACTION = 1.001

NUMBER = 10.01 ($\times 2^0$)

= 2.25

Example of Floating Point Addition

	sign	exponent			fraction		
X	0	1	0	0	1	1	0
Y	0	0	1	1	1	0	0

$X = 1.110 \times 2^{2}$ ← (9-7)

$Y = 1.000 \times 2^{0}$

$X = 1.110 \times 2^{2}$

$Y = 0.010 \times 2^{2}$

$10.000 \times 2^{2} \Rightarrow 1.000 \times 2^{3}$

$\Rightarrow 0\ 1010\ 000$

NOT ON EXAM

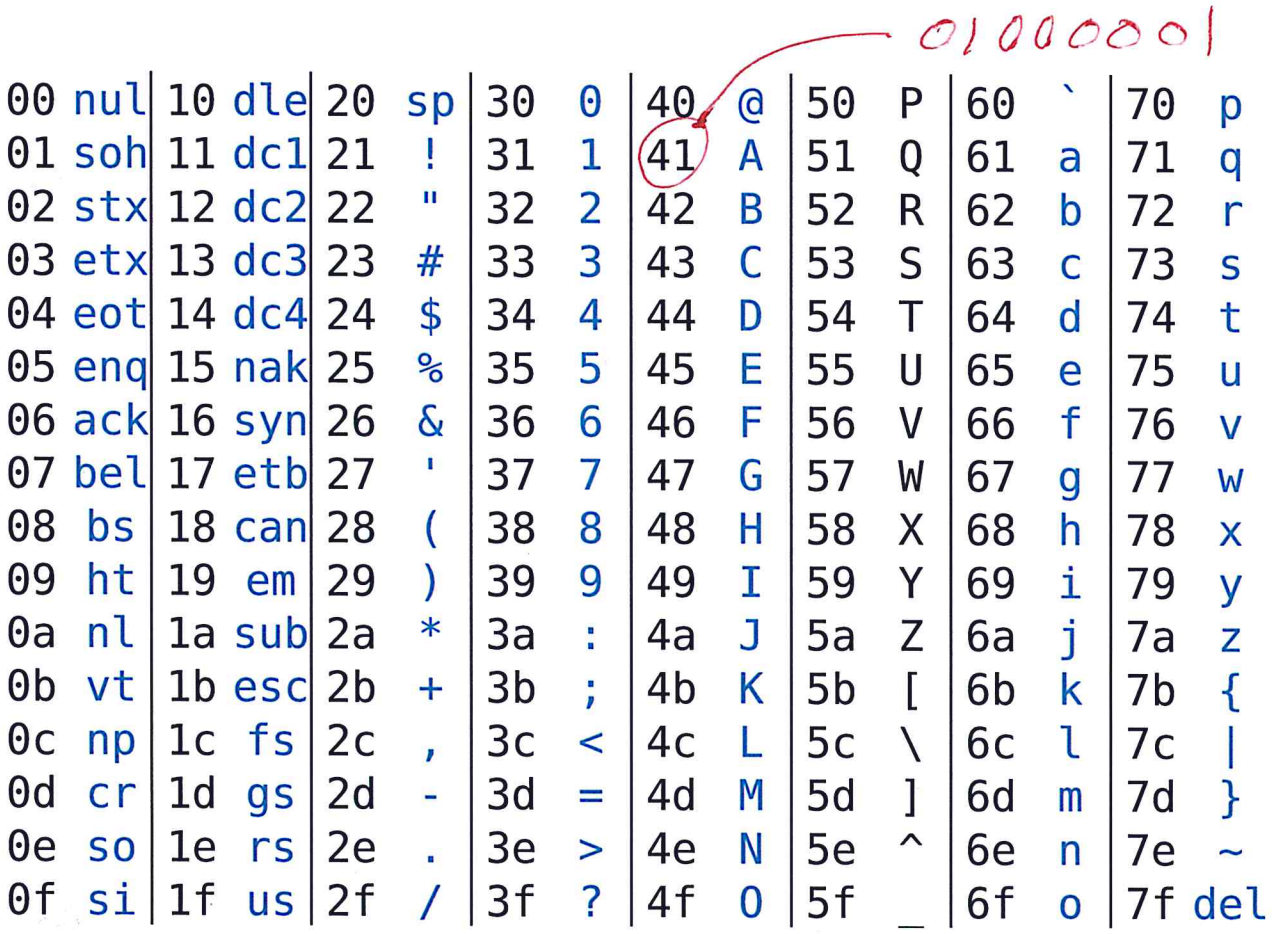
Special Floating Point Numbers

Zero	0 00000000 00000000000000000000000
Negative Zero	1 00000000 00000000000000000000000
Infinity	0 11111111 00000000000000000000000
Negative Infinity	1 11111111 00000000000000000000000
Not a Number (NaN)	0 11111111 00001000000000100001000

Text: ASCII Characters

ASCII: Maps 128 characters to 7-bit code.

- both printable and non-printable (ESC, DEL, …) characters



Interesting Properties of ASCII Code

What is relationship between a decimal digit ('0', '1', …) and its ASCII code?

What is the difference between an upper-case letter ('A', 'B', …) and its lower-case equivalent ('a', 'b', …)?

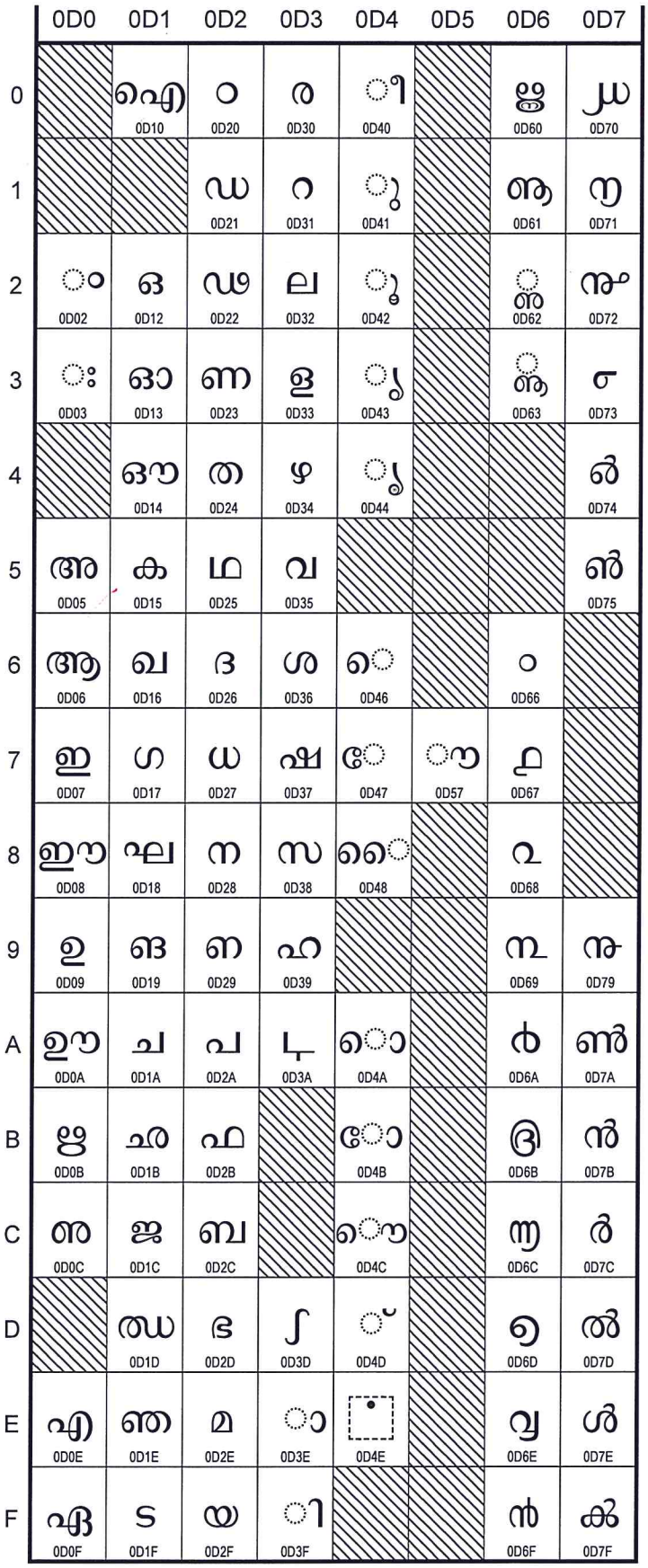
Given two ASCII characters, how do we tell which comes first in alphabetical order?

Are 128 characters enough?
(http://www.unicode.org/)

No new operations -- integer arithmetic and logic.

HEX	C			J	K	V
4E00 — 1.0	G0-523B	HB1-A440	T1-4421	J0-306C	K0-6C69	V1-4A21
4E01 — 1.1	 G0-3621	 HB1-A442	 T1-4423	 J0-437A	 K0-6F4B	V1-4A22
4E02 — 1.1	 G5-3021		 T4-2126	 J1-3021		
4E03 — 1.1	 G0-465F	 HB1-A443	 T1-4424	 J0-3C37	K0-7652	 V1-4A23
4E04 — 1.1	 GE-2121	 H-9EB3	 T3-2126	 J1-3022		
4E05 — 1.1	 GE-2122		 T3-2125	 J1-3023		
4E06 — 1.1	 GK-6837				K2-2121	
4E07 — 1.2	 G0-4D72	HB2-C945	T2-2126	J0-4B7C	 K0-5832	V1-4A24
4E08 — 1.2	 G0-5549	 HB1-A456	 T1-4437	 J0-3E66	K0-6D5B	V1-4A25
4E09 — 1.2	G0-487D	HB1-A454	T1-4435	J0-3B30	K0-5F32	V1-4A26
4E0A — 1.2	G0-494F	HB1-A457	T1-4438	J0-3E65	K0-5F3E	V1-4A27
4E0B — 1.2	G0-4F42	HB1-A455	T1-4436	J0-323C	K0-793B	V1-4A28
4E0C — 1.2	G0-5822	HB2-C946	T2-2127	J1-3024	K2-2122	
4E0D — 1.3	G0-323B	HB1-A4A3	T1-4462	J0-4954	K0-5C74	V1-4A29
4E0E — 1.3	G0-536B	HB2-C94F	T2-212F	J0-4D3F	K2-2123	V1-4A2A
4E0F — 1.3	G3-3021	HB2-C94D	T2-212D	J4-2123	K2-2124	
4E10 — 1.3	G0-5824	HB1-A4A2	T1-4461	J0-5022	K2-2125	V1-4A2B
4E11 — 1.3	G0-3373	HB1-A4A1	T1-4460	J0-312F	K0-7564	V1-4A2C
4E12 — 1.3	GE-2123		T4-2139	J1-3025		
4E13 — 1.3	G0-5728					

HEX	C			J	K	V
4E14 — 1.4	G0-4752	HB1-A542	T1-4562	J0-336E	K0-7326	V1-4A2D
4E15 — 1.4	G0-5827	HB1-A541	T1-4561	J0-5023	K0-5D60	V1-4A2E
4E16 — 1.4	G0-4A40	HB1-A540	T1-4560	J0-4024	K0-6126	V1-4A2F
4E17 — 1.3	GE-2124		T4-2155	J0-5242		
4E18 — 1.4	G0-4770	HB1-A543	T1-4563	J0-3556	K0-4E78	V1-4A30
4E19 — 1.4	G0-317B	HB1-A4FE	T1-455F	J0-4A3A	K0-5C30	V1-4A31
4E1A — 1.4	G0-5235	H-9EB2				
4E1B — 1.4	G0-3454					
4E1C — 1.4	G0-362B	H-9DD6				
4E1D — 1.4	G0-4B3F					
4E1E — 1.5	G0-5829	HB1-A5E0	T1-4722	J0-3E67	K0-632A	V1-4A32
4E1F — 1.5	GE-2125	HB1-A5E1	T1-4723	J1-3026	K1-6D4A	
4E20 — 1.5	G5-3023		T3-2262			
4E21 — 1.5	GE-2126	H-994F	T3-2261	J0-4E3E	K2-2126	
4E22 — 1.5	G0-362A		T3-2263			V1-4A33
4E23 — 1.6	GE-2127		T4-2335	J1-3027		
4E24 — 1.6	G0-413D	H-89CE	T3-243F	J1-3028	K2-2127	
4E25 — 1.6	G0-514F					
4E26 — 1.7	G8-2C76	HB1-A8C3	T1-4B64	J0-4A42	K1-6172	
4E27 — 1.7	G0-4925					



Other Data Types

Text strings

- sequence of characters, terminated with NULL (0)
- typically, no hardware support

Image

- array of pixels
  - monochrome: one bit (1/0 = black/white)
  - color: red, green, blue (RGB) components (e.g., 8 bits each)
  - other properties: transparency
- hardware support:
  - typically none, in general-purpose processors
  - MMX -- multiple 8-bit operations on 32-bit word

Sound

- sequence of fixed-point numbers

LC-3 Data Types

Some data types are supported directly by the instruction set architecture.

For LC-3, there is only one hardware-supported data type:

- **16-bit 2's complement signed integer**
- **Operations: ADD, AND, NOT**

Other data types are supported by interpreting 16-bit values as logical, text, fixed-point, etc., in the software that we write.

Logical Operations

Operations on logical TRUE or FALSE

- two states -- takes one bit to represent: TRUE=1, FALSE=0

A	B	A AND B
0	0	0
0	1	0
1	0	0
1	1	1

A	B	A OR B
0	0	0
0	1	1
1	0	1
1	1	1

A	NOT A
0	1
1	0

A	B	A XOR B
0	0	0
0	1	1
1	0	1
1	1	0

← EXCLUSIVE-OR

View *n*-bit number as a collection of *n* logical values

- operation applied to each bit independently

Examples of Logical Operations

AND **USEFUL AS "MASKS"**

- useful for clearing bits
  - AND with zero = 0
  - AND with one = no change

```
  11000101
AND 00001111
  00000101
```

OR

- useful for setting bits
  - OR with zero = no change
  - OR with one = 1

```
  11000101
OR  00001111
  11001111
```

NOT

- unary operation -- one argument
- flips every bit

```
NOT 11000101
  00111010
```

Transistor: Building Block of Computers

Microprocessors contain millions of transistors

- Intel Pentium 4 (2000): **48 Million**
- IBM/Apple PowerPC G5 (2003): **58 Million**
- Intel i7, Sandy Bridge E (2011): **2.27 Billion**
- Apple A8X (Arm64) (2014) **3 Billion**
- Intel 18-core Xeon Haswell (2014) **5.56 Billion**
- AMD 32-core Epyc (2017) **19.2 Billion**

Logically, each transistor acts as a switch

Combined to implement logic functions

- **AND, OR, NOT**

Combined to build higher-level structures

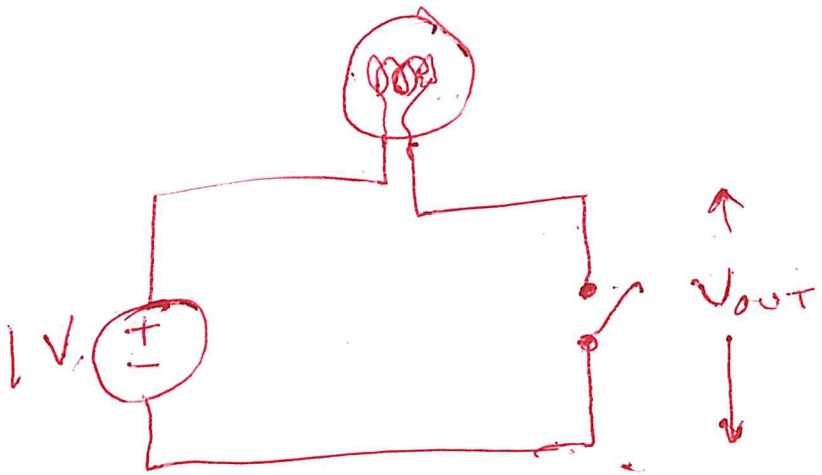
- **Adder, multiplexer, decoder, register, …**

Combined to build processor

- **LC-3**

Simple Switch Circuit

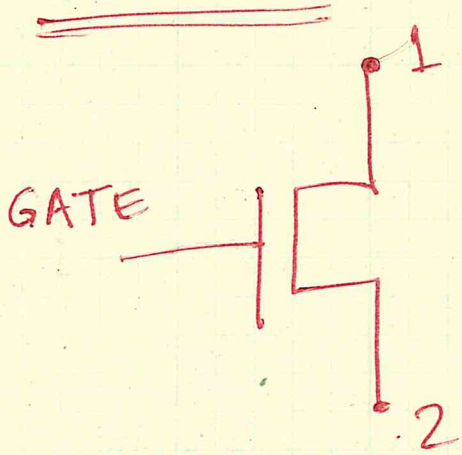
(TRANSISTOR)

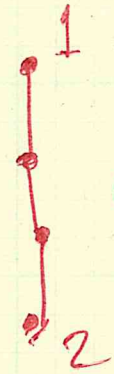


TWO STATES: {ON, OFF}

"CMOS" TECHNOLOGY

nMOS

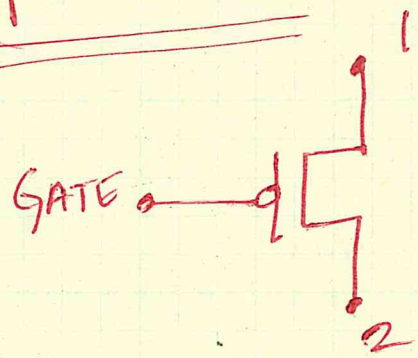




GATE = ON (1V)
SWITCH CLOSED

GATE = OFF (0V)
SWITCH OPEN

pMOS



GATE = 0: SWITCH CLOSED
1: SWITCH OPEN

CMOS Circuit

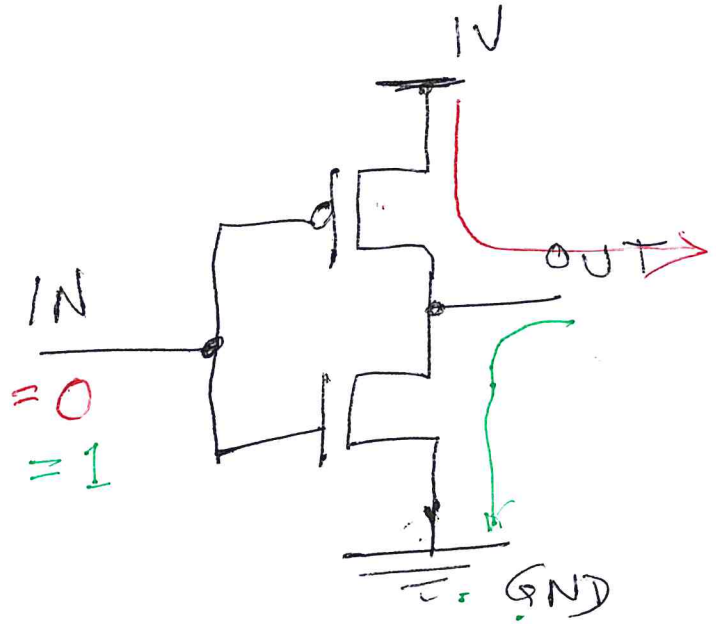
Complementary MOS

Uses both n-type and p-type MOS transistors

- **p-type**
  - **Attached to + voltage**
  - **Pulls output voltage UP when input is zero**
- **n-type**
  - **Attached to GND**
  - **Pulls output voltage DOWN when input is one**

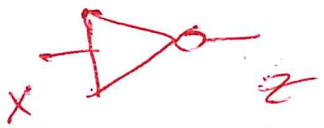
For all inputs, make sure that output is either connected to GND or to +, but not both!

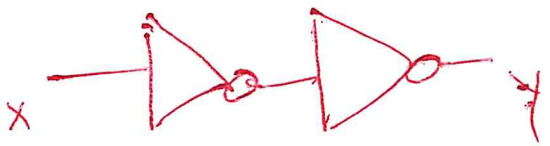
Inverter (NOT Gate)



IN	OUT
0	1
1	0

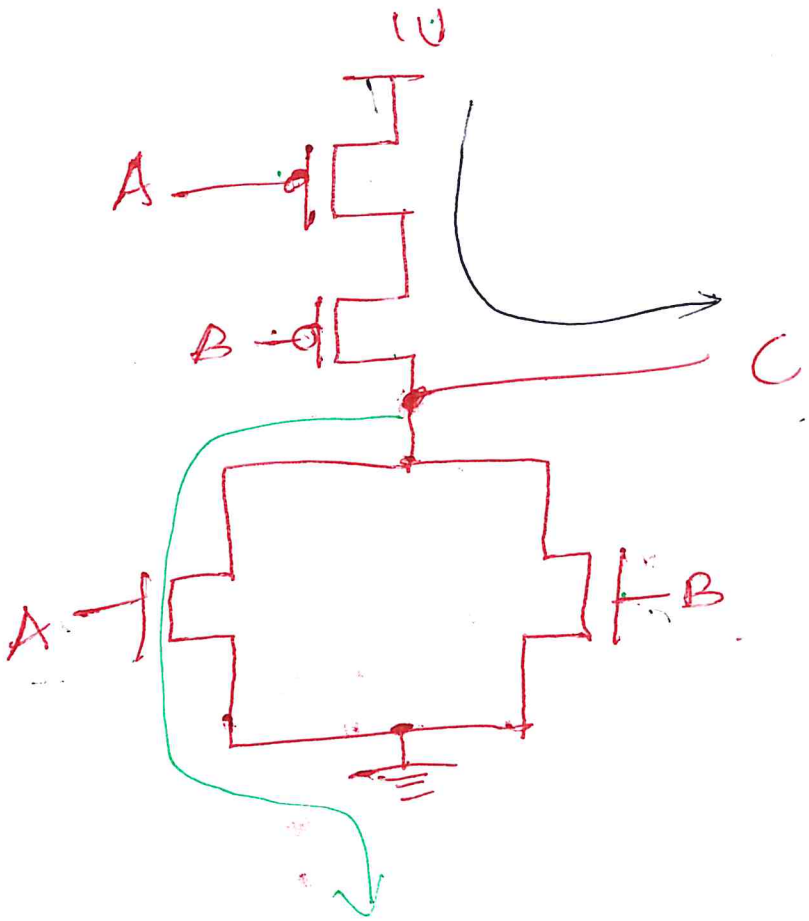
SYMBOL:

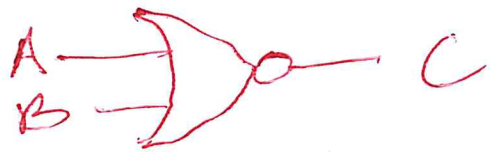




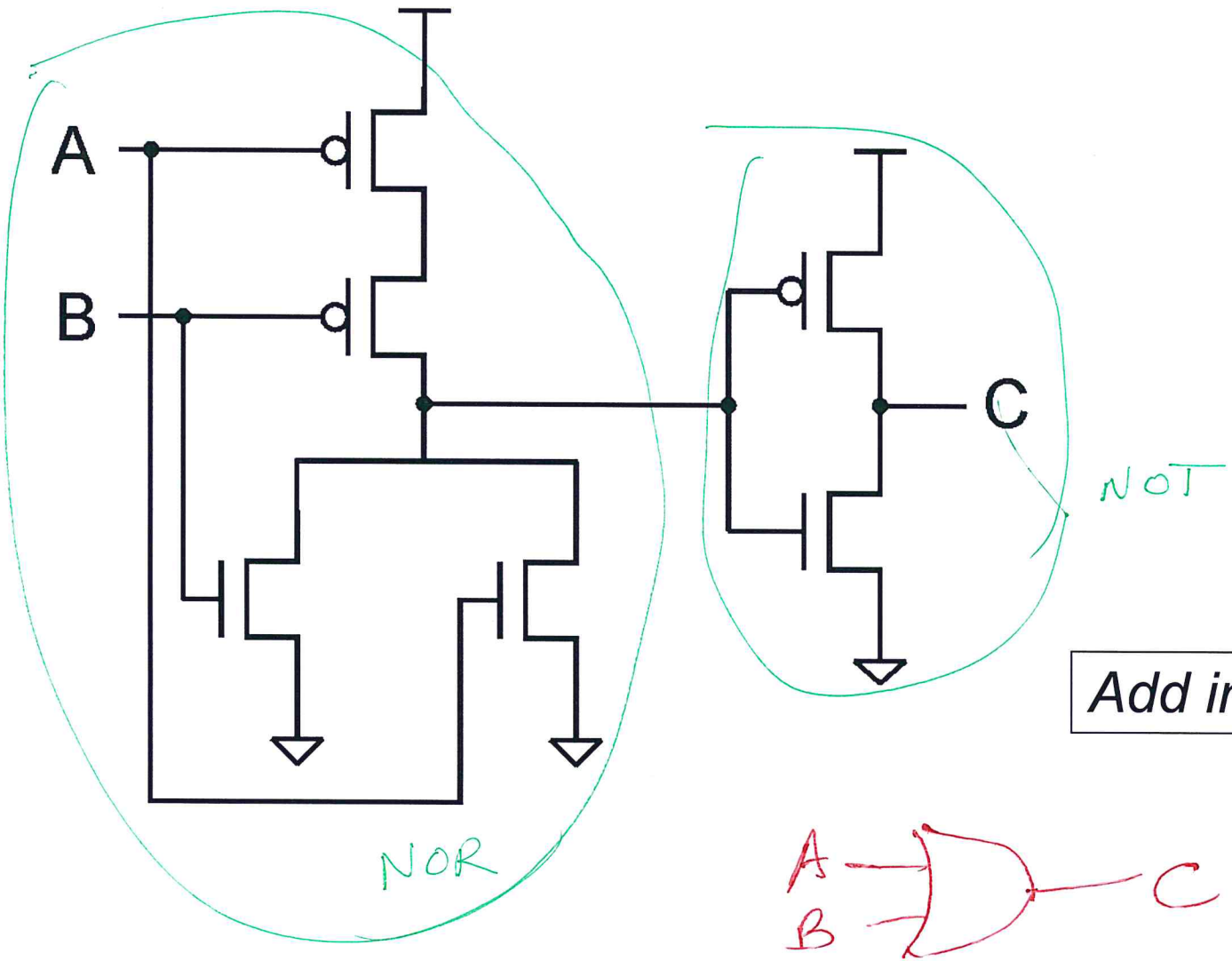
NOR Gate =(NOTOR)

A	B	C
0	0	1
0	1	0
1	0	0
1	1	0





OR Gate



A	B	C
0	0	0
0	1	1
1	0	1
1	1	1

Add inverter to NOR.