The basics of logic design: Sequential logic

Yipeng Huang

Rutgers University

April 23, 2024

Table of contents

Announcements

Class session plan

- Tuesday, 4/23: Diving deeper: Digital logic. (CS:APP Chapter 4.2) (Recommended reading: Patterson & Hennessy, Computer organization and design, appendix on "The Basics of Logic Design." Available online via Rutgers Libraries)
- Thursday, 4/25: Survey of advanced topics in (quantum) computer architecture.
- Tuesday, 5/7: 12:00-15:00, SERC 111, closed book, closed notes, no electronic devices, no calculator final exam.

Table of contents

The set of logic gates {NOT, AND, OR} is universal

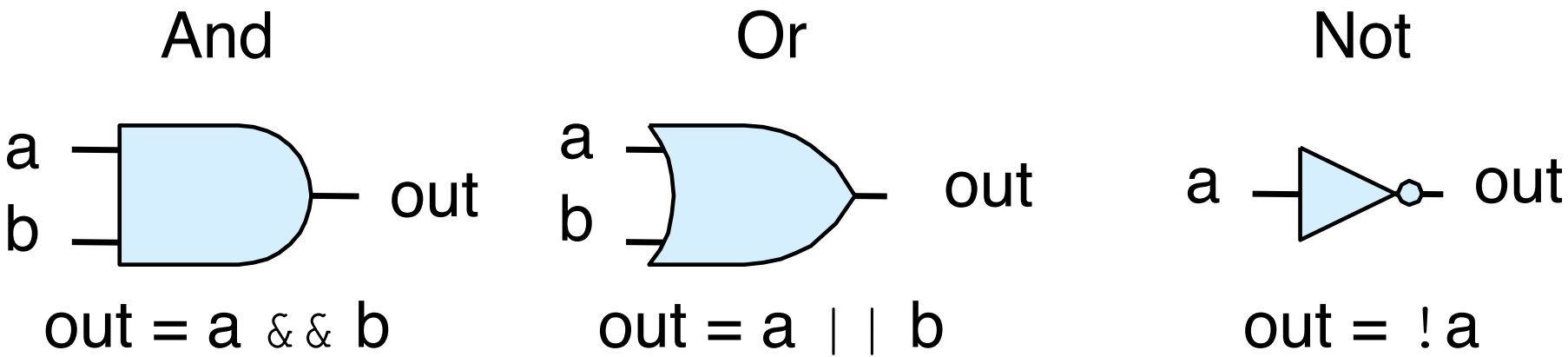


Figure: Source: CS:APP

The set of logic gates {NOT, AND, OR} is universal

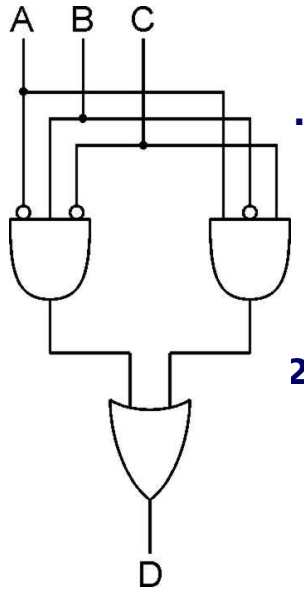
- Any truth table can be expressed as sum of products form.
- Write each row with output 1 as a product (minterm).
- Sum the products (minterm).
- Forms a disjunctive normal form (DNF).
- $D = \overline{A}B\overline{C} + A\overline{B}C$
- Always only needs NOT, AND, OR gates.
- Supplementary slides example...

Logical Completeness

Can implement ANY truth table with AND, OR, NOT.

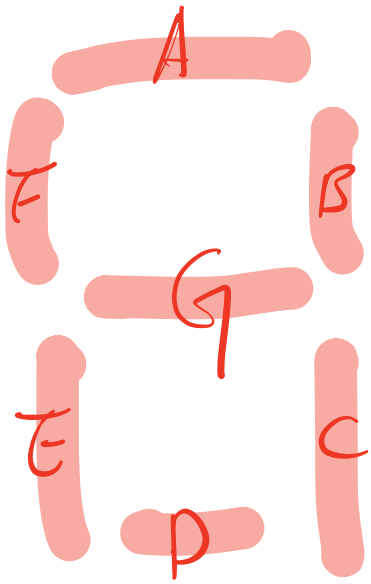
A	B	C	D
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	0
1	1	1	0

Sum of products
OR of AND clauses



. AND combinations that yield a "1" in the truth table.

2. OR the results of the AND gates.



$E = \{0, 2, 6, 8, 9\}$

input-[3:0]

input [3]

input [2]

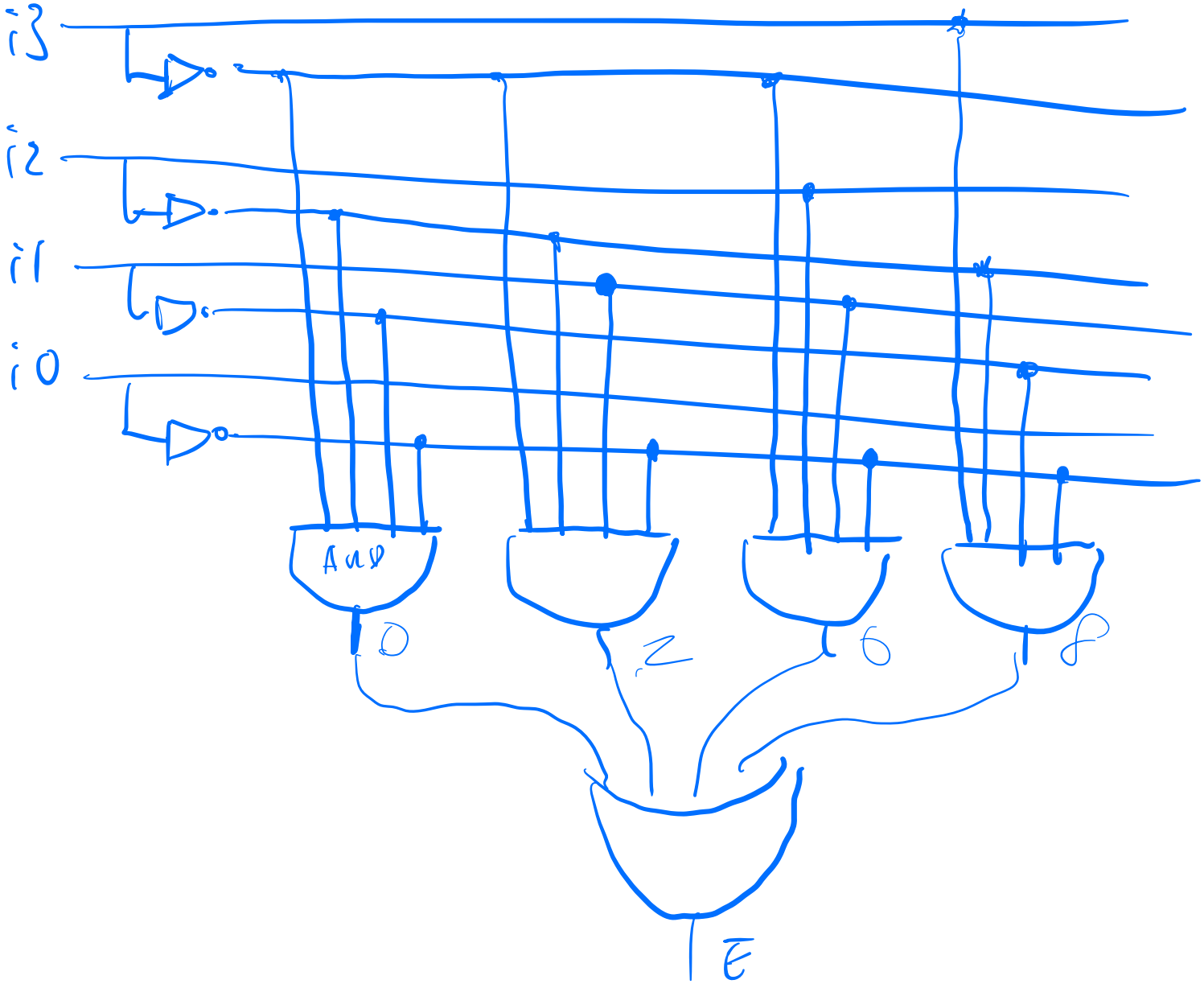
input [1]

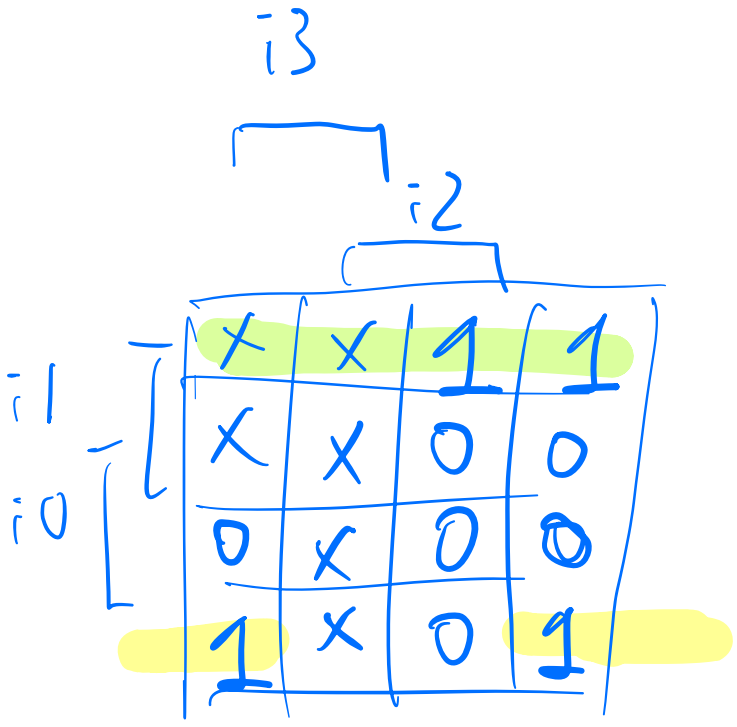
input [0]

digit	input[3]	[2]	[1]	[0]	E
0	0	0	0	0	1
1	0	0	0	1	0
2	0	0	1	0	1
3	0	0	1	1	0
4	0	1	0	0	0
5	0	1	0	1	0
6	0	1	1	0	1
7	0	1	1	1	0
8	1	0	0	0	1
9	1	0	0	1	0

$E = (\bar{3} \cdot \bar{2} \cdot \bar{1} \cdot \bar{0}) + (\bar{3} \bar{2} \bar{1} \bar{0}) + (\bar{3} 2 \bar{1} \bar{0}) + (3 \bar{2} \bar{1} \bar{0})$

sum of products, ors-of-ands


i3
i2
i1
i0
A
D
Z
6
E



$$E = \bar{i1}\,\bar{i0} + \bar{i1}\,i0\,\bar{i2}$$

$$= \bar{i0}\left(\bar{i1} + \bar{i1}\,\bar{i2}\right)$$

The set of logic gates {NOT, AND, OR} is universal

- Any truth table can be expressed as sum of products form.
- Write each row with output 1 as a product (minterm).
- Sum the products (minterm).
- Forms a disjunctive normal form (DNF).
- $D = \overline{A}B\overline{C} + A\overline{B}C$
- Always only needs NOT, AND, OR gates.
- Supplementary slides example...

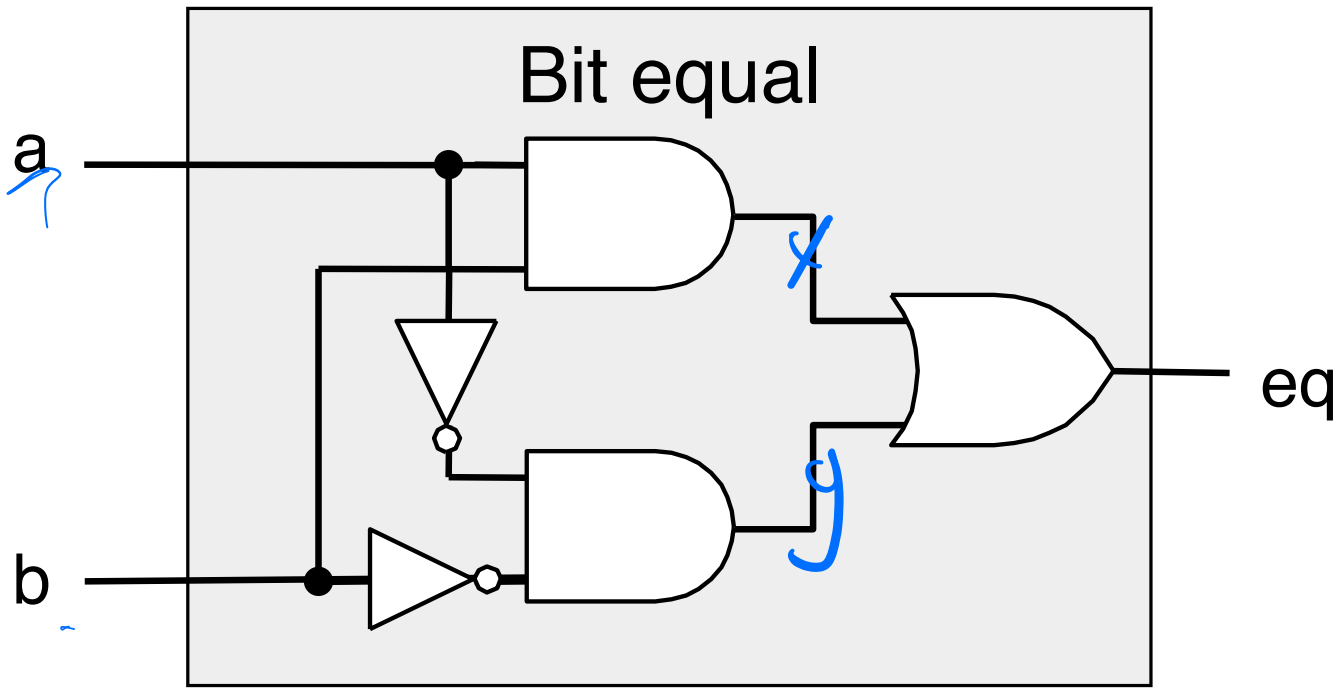
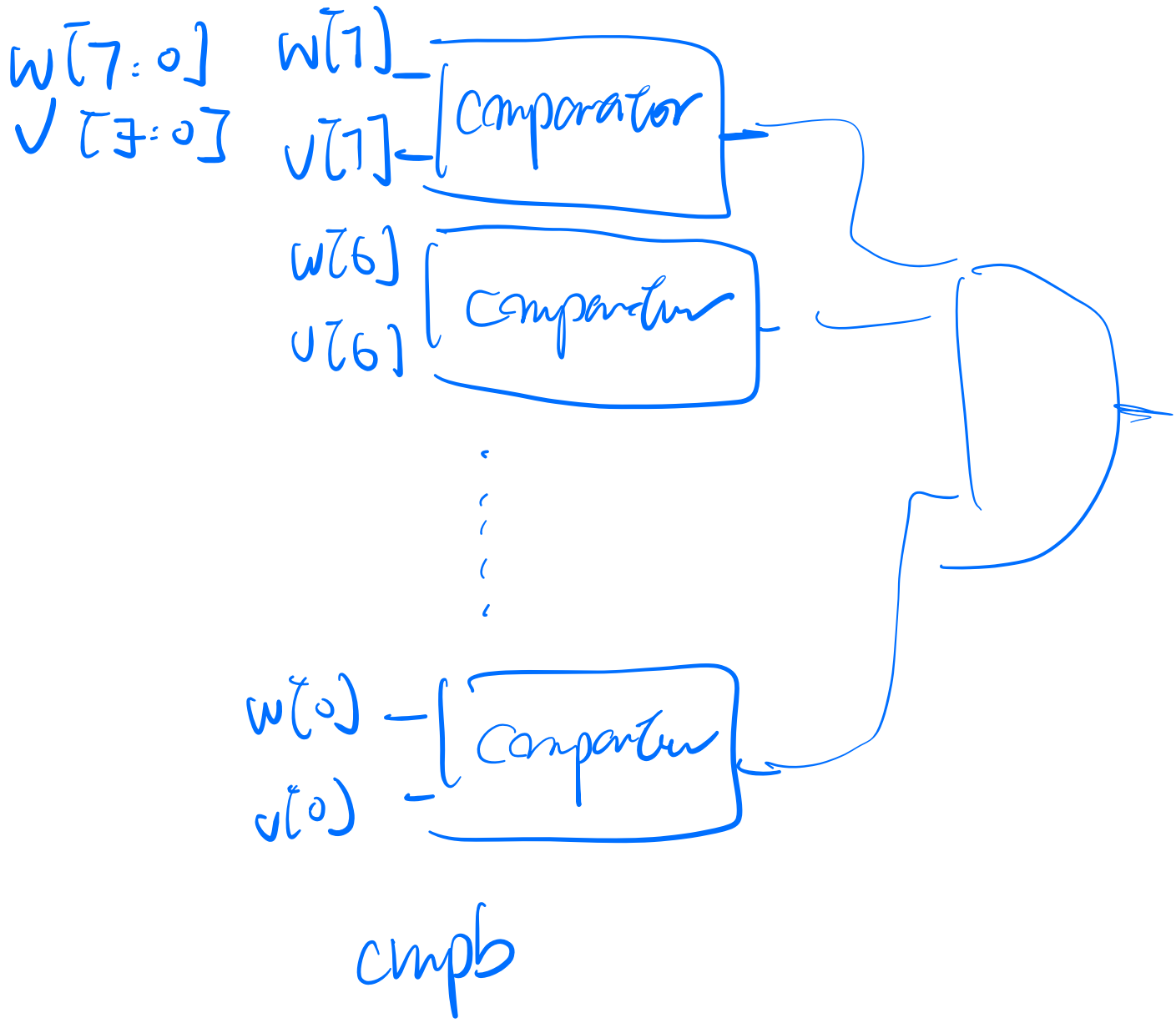


Figure: Source: CS:APP

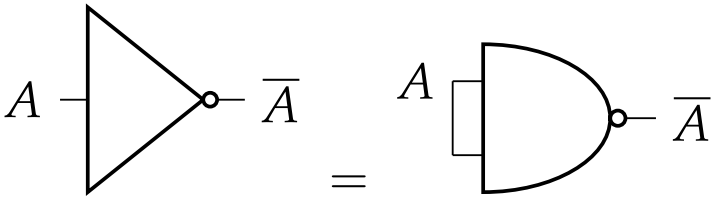
a	$\bar{a}$	b	$\bar{b}$	x	y	eq
0	1	0	1	0	1	1
0	1	1	0	0	0	0
1	0	0	1	0	0	0
1	0	1	0	1	0	1

char w char v



The NAND gate is universal

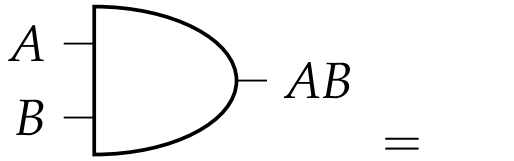
NOT gate as a single NAND gate

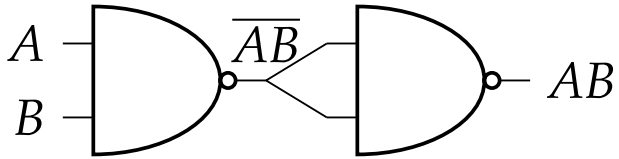


A	$\overline{A}$	AA	$\overline{AA}$
0	1	0	1
1	0	1	0

Table: $\overline{A} = \overline{AA}$

AND gate as two NAND gates





A	B	AB	$\overline{AB}$	$\overline{\overline{AB}}$
0	0	0	1	0
0	1	0	1	0
1	0	0	1	0
1	1	1	0	1

Table: $AB = \overline{\overline{AB}}$

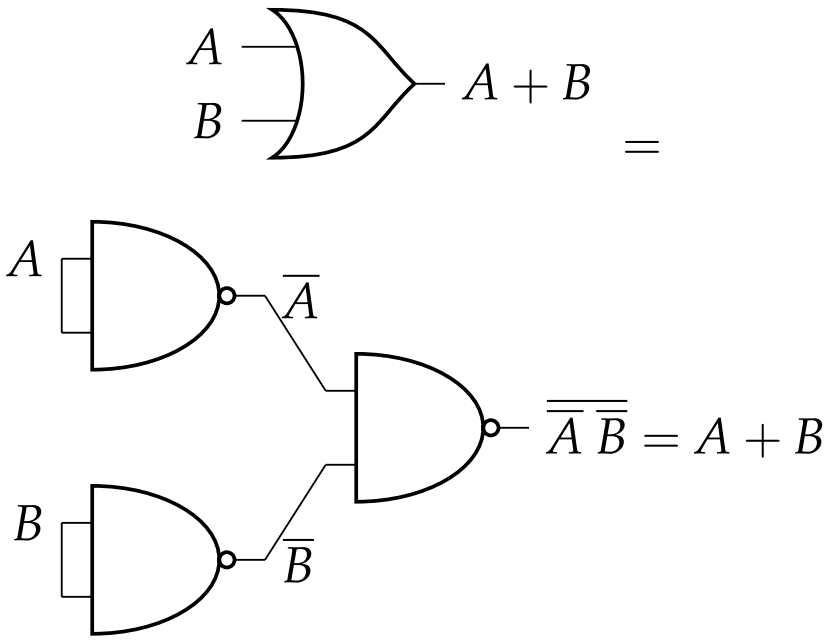
The NAND gate is universal

De Morgan's Law

A	B	$\overline{A}$	$\overline{B}$	$\overline{A}\,\overline{B}$	$A+B$	$\overline{A+B}$
0	0	1	1	1	0	1
0	1	1	0	0	1	0
1	0	0	1	0	1	0
1	1	0	0	0	1	0

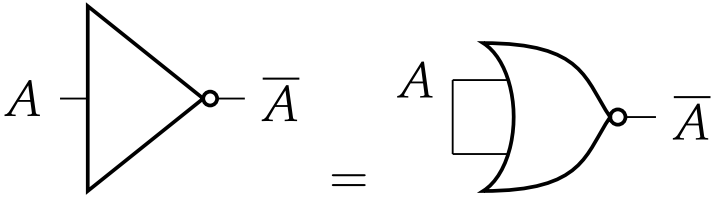
Table: $\overline{A}\,\overline{B} = \overline{A+B}$

OR gate as three NAND gates



The NOR gate is universal

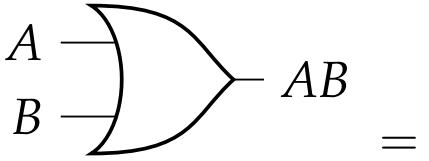
NOT gate as a single NOR gate

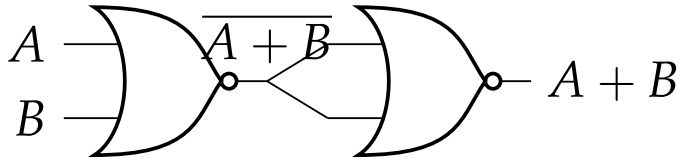


A	$\overline{A}$	$A+A$	$\overline{A+A}$
0	1	0	1
1	0	1	0

Table: $\overline{A} = \overline{A+A}$

OR gate as two NOR gates





A	B	$A+B$	$\overline{A+B}$	$\overline{\overline{A+B}}$
0	0	0	1	0
0	1	1	0	1
1	0	1	0	1
1	1	1	0	1

Table: $A+B = \overline{\overline{A+B}}$

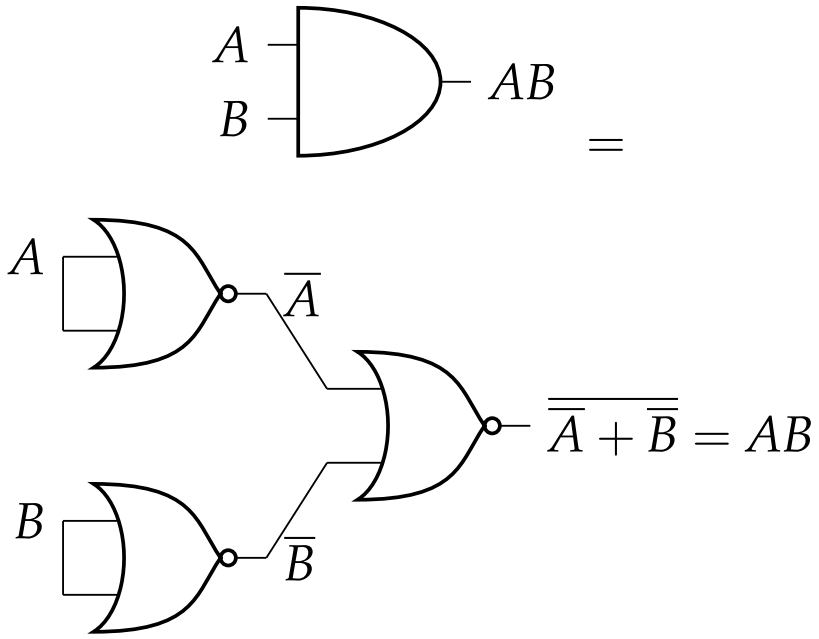
The NOR gate is universal

De Morgan's Law

A	B	$\overline{A}$	$\overline{B}$	$\overline{A}+\overline{B}$	AB	$\overline{AB}$
0	0	1	1	1	0	1
0	1	1	0	1	0	1
1	0	0	1	1	0	1
1	1	0	0	0	1	0

Table: $\overline{A}+\overline{B}=\overline{AB}$

AND gate as three NOR gates



Combinational vs. sequential logic

Combinational logic

- No internal state nor memory
- Output depends entirely on input
- Examples: NOT, AND, NAND, OR, NOR, XOR, XNOR gates, decoders, multiplexers.

Sequential logic

- Has internal state (memory)
- Output depends on the inputs and also internal state
- Examples: latches, flip-flops, Mealy and Moore machines, registers, pipelines, SRAMs.

Table of contents

Decoders

Takes n-bit input, uses it as an index to enable exactly one of 2^n outputs

Internal design of 1:2 decoder

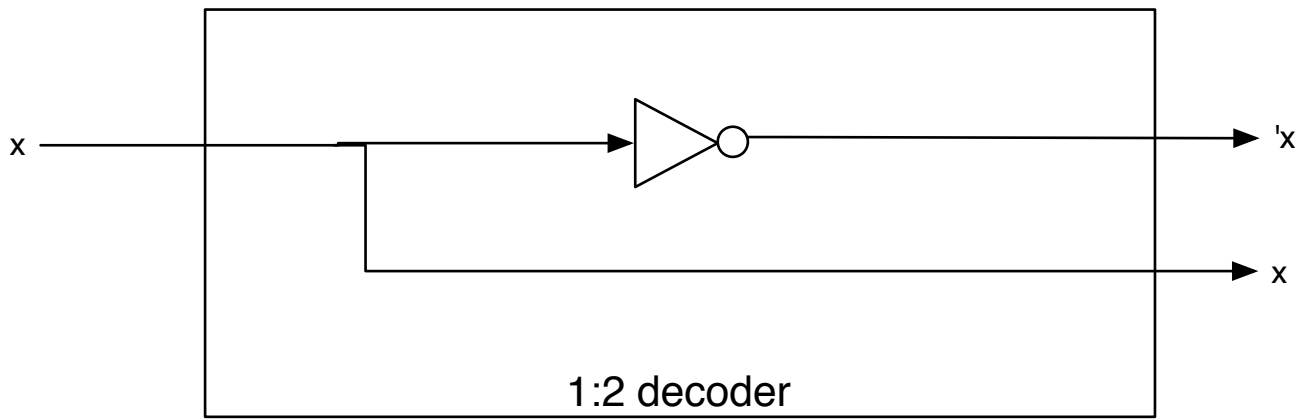


Figure: Source: Mano & Kime

Decoders

Hierarchical design of decoder (2:4 decoder)

Takes n-bit input, uses it as an index to enable exactly one of 2^n outputs

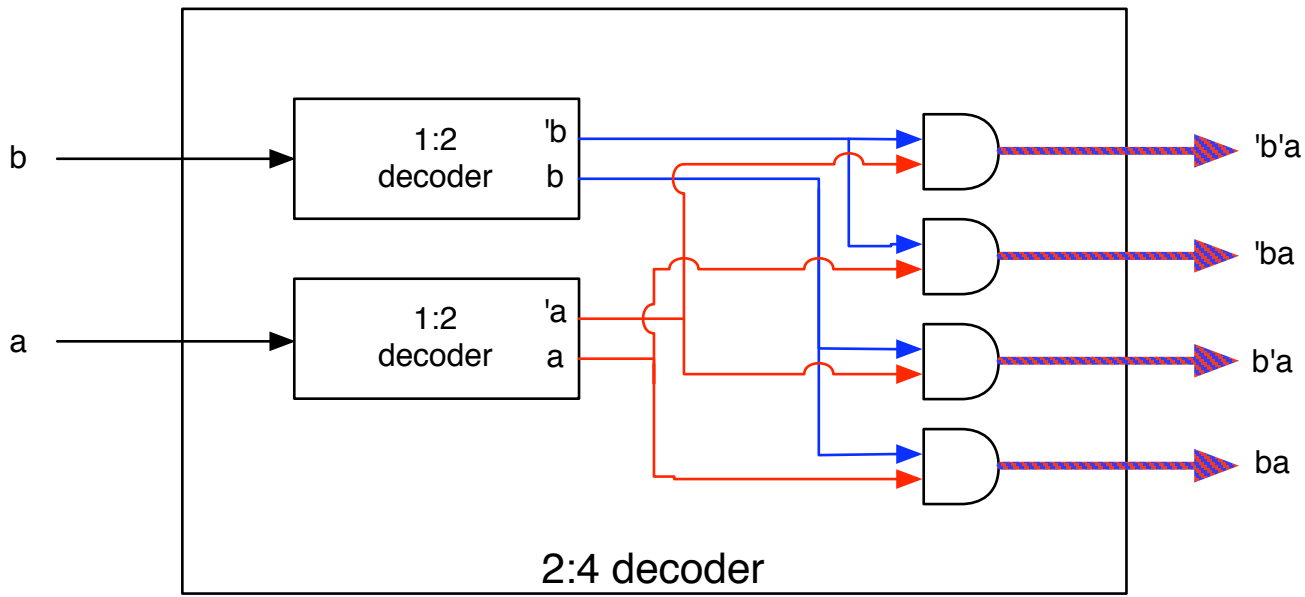


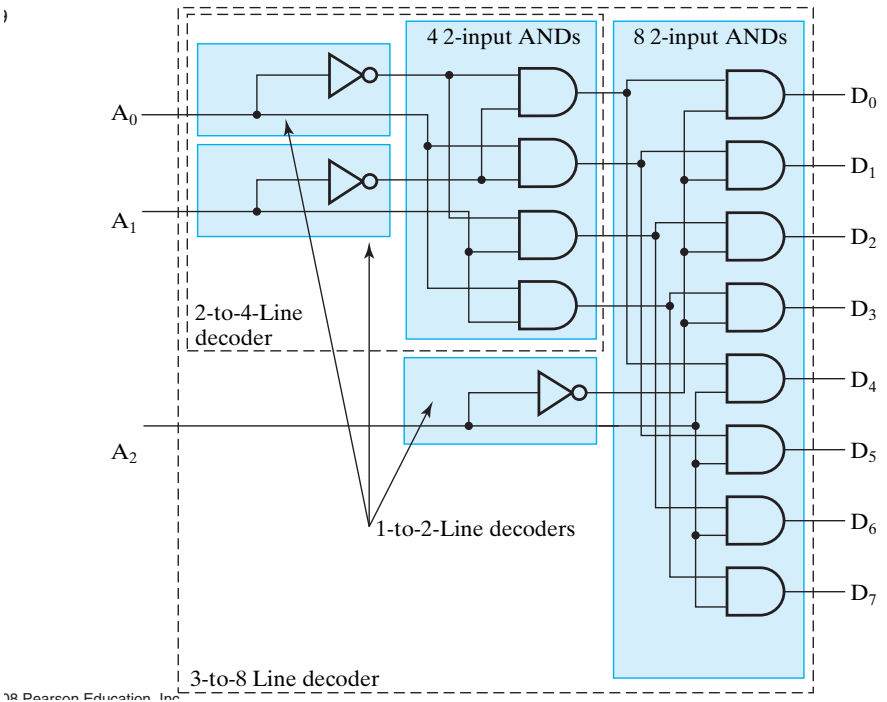
Figure: Source: Mano & Kime

Decoders

Decoder (3:8)

Takes n-bit input, uses it as an index to enable exactly one of 2^n outputs

Hierarchical design: use small decoders to build bigger decoder



Note: A_2 "selects" whether the 2-to-4 lin decoder is active in th top half (A_2=0) or the bottom (A_2=1)

Figure: Source: Mano & Kime

Multiplexers

Using n-bit selector input, select among one of 2^n choices

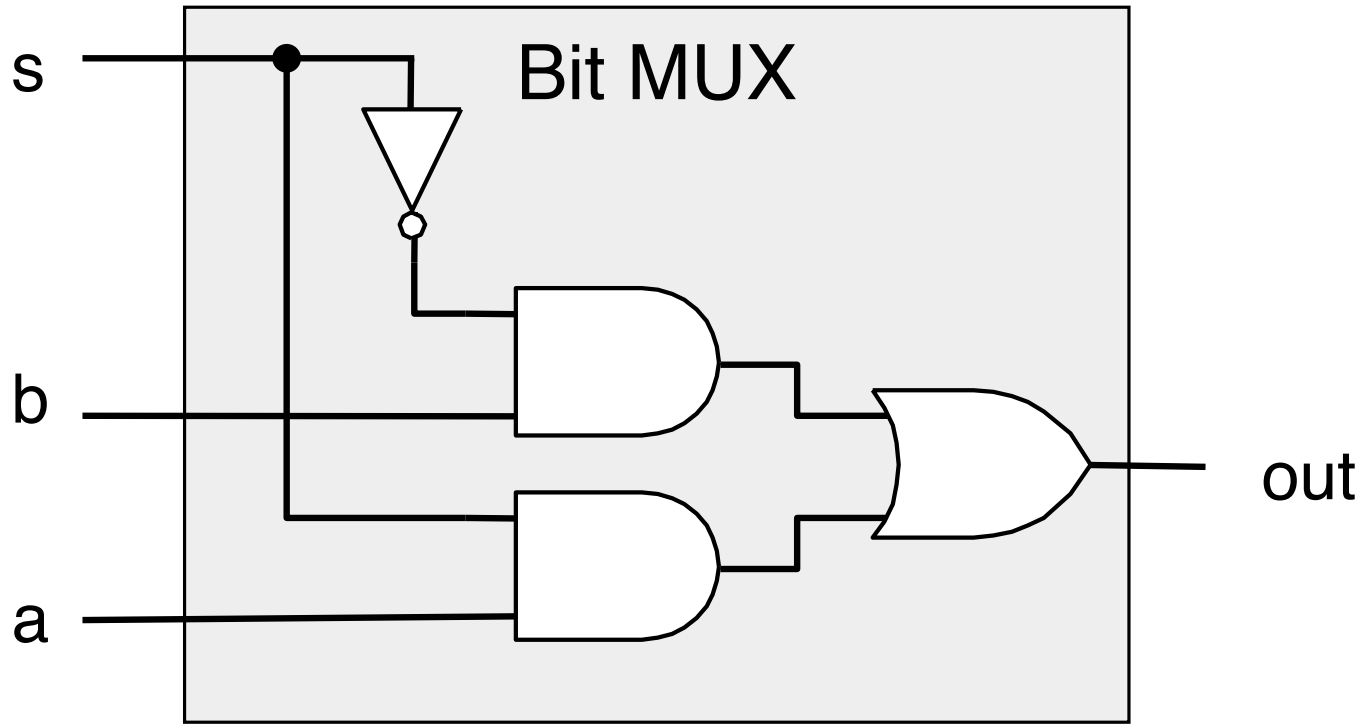


Figure: Source: CS:APP

Multiplexers

Using n-bit selector input, select among one of 2^n choices

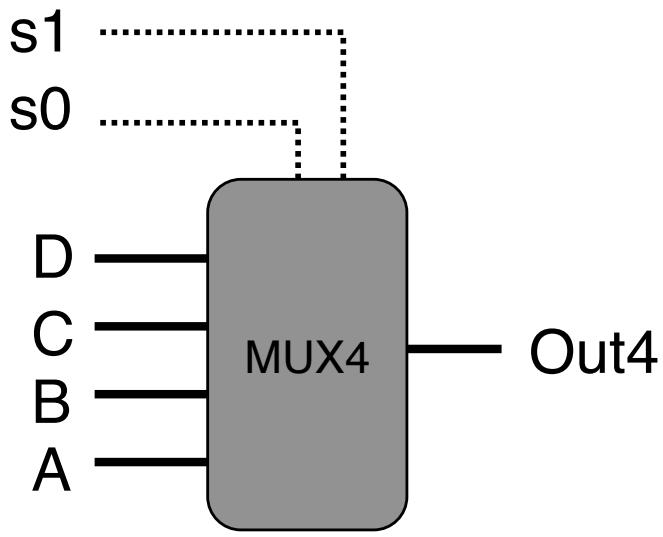


Figure: Source: CS:APP

Multiplexers

Using n-bit selector input, select among one of 2^n choices

Internal mux organization

3-26

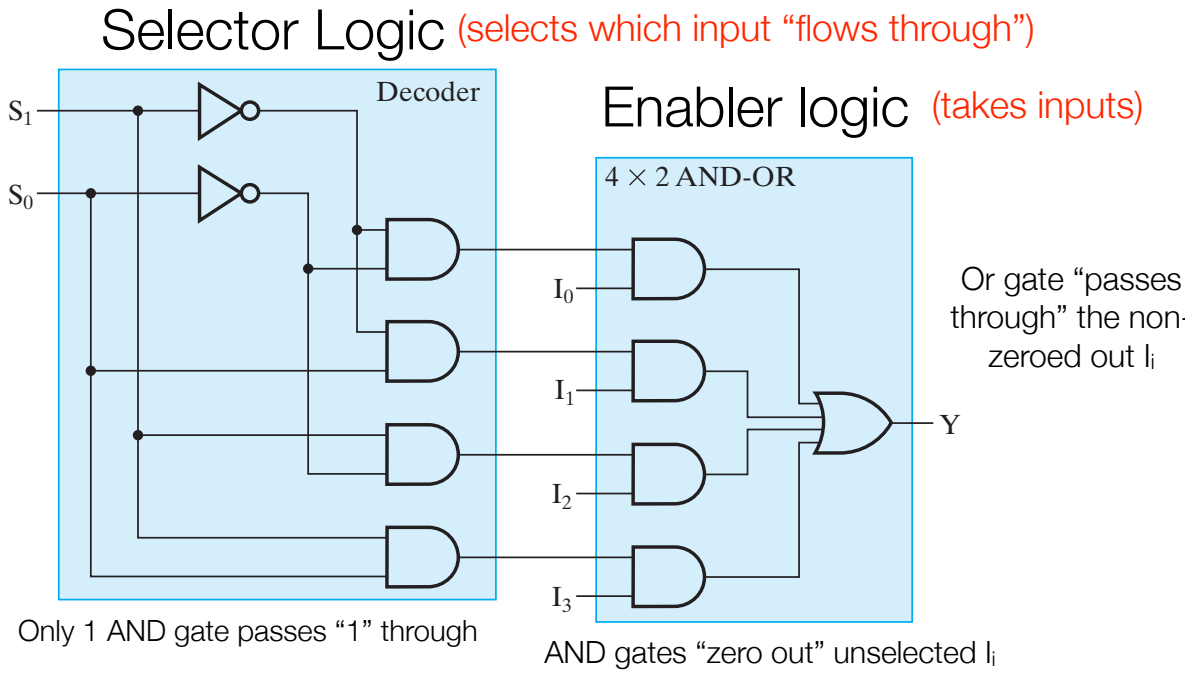
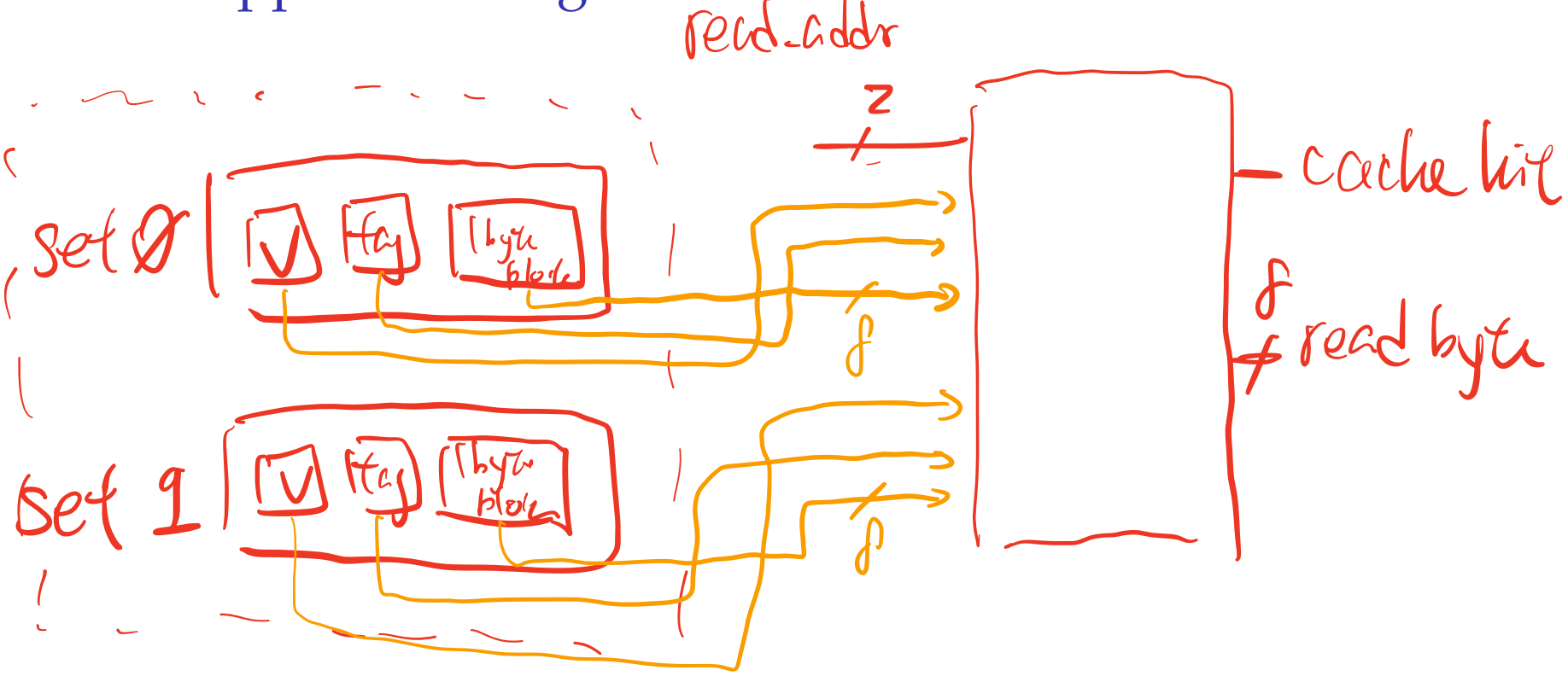


Figure: Source: Mano & Kime

Table of contents

directMapped read logic



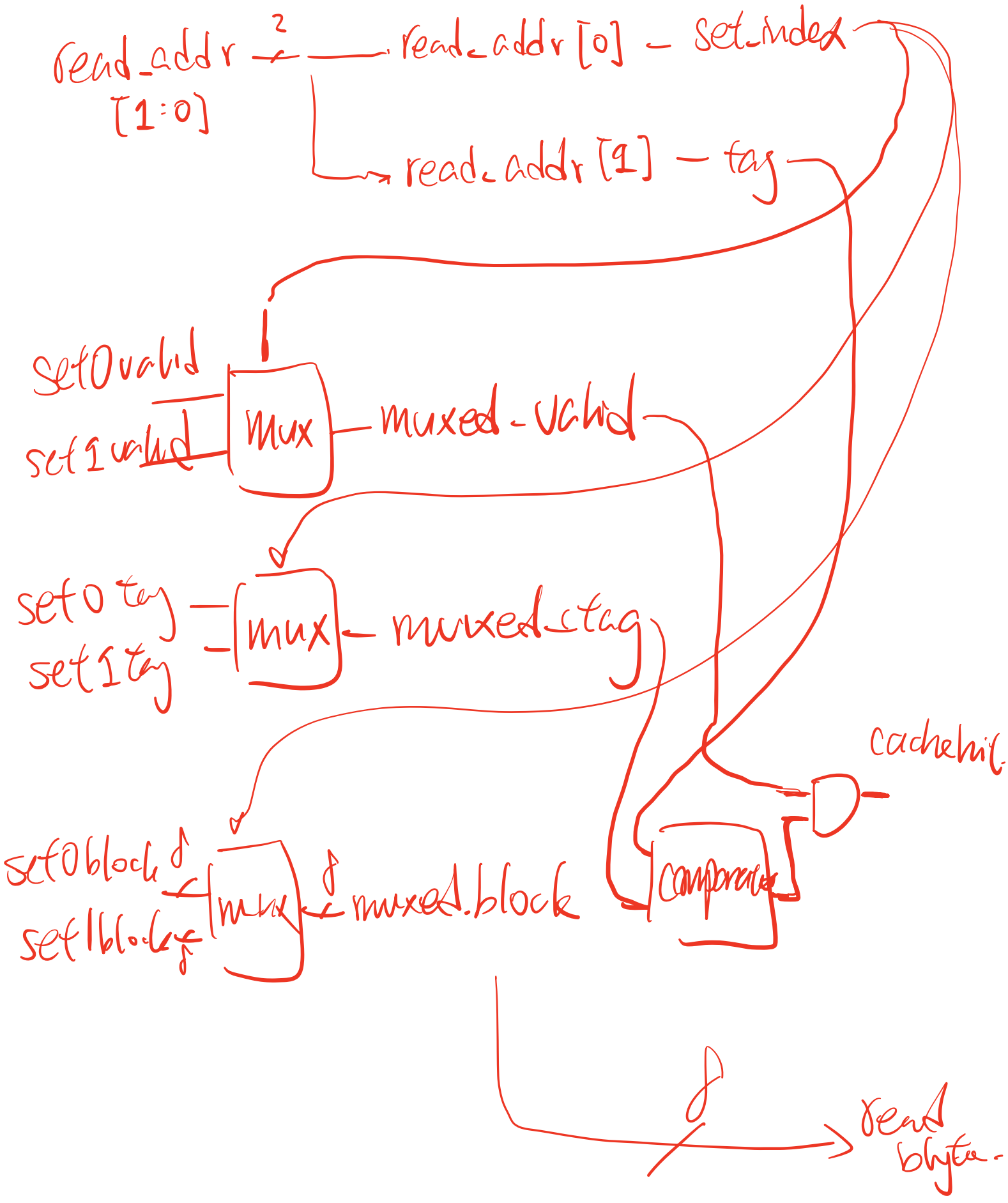

read_addr [1:0]
2
read_addr[0] - set_index
read_addr[1] - tag
set0valid
set1valid
Mux
muxed_valid
set0 tag
set1 tag
mux
muxed_tag
cachehit
set0block
set1block
mux
muxed_block
comparator
read bytes

Table of contents

Sequential logic

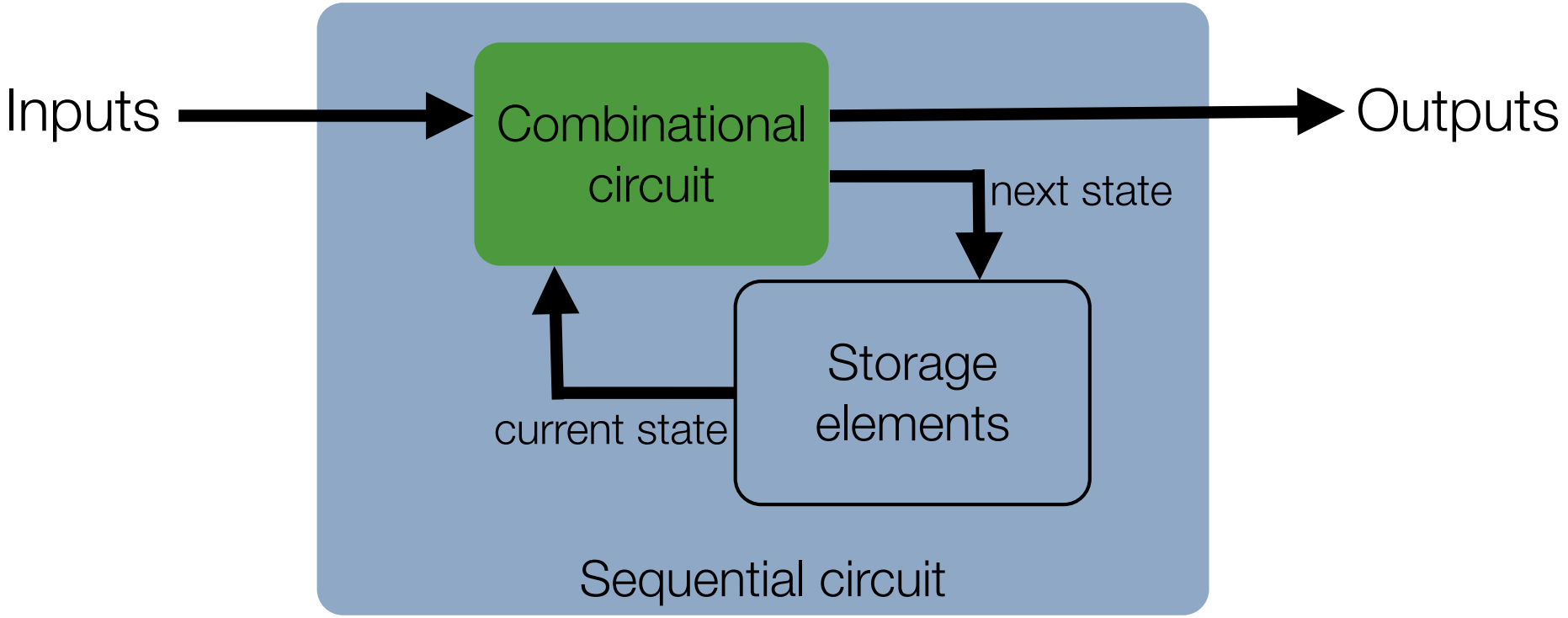


Figure: Source: Mano & Kime

The simplest sequential logic element: The set/reset (SR) latch

SR latch

- Latch constructed of cross-coupled NOR gates

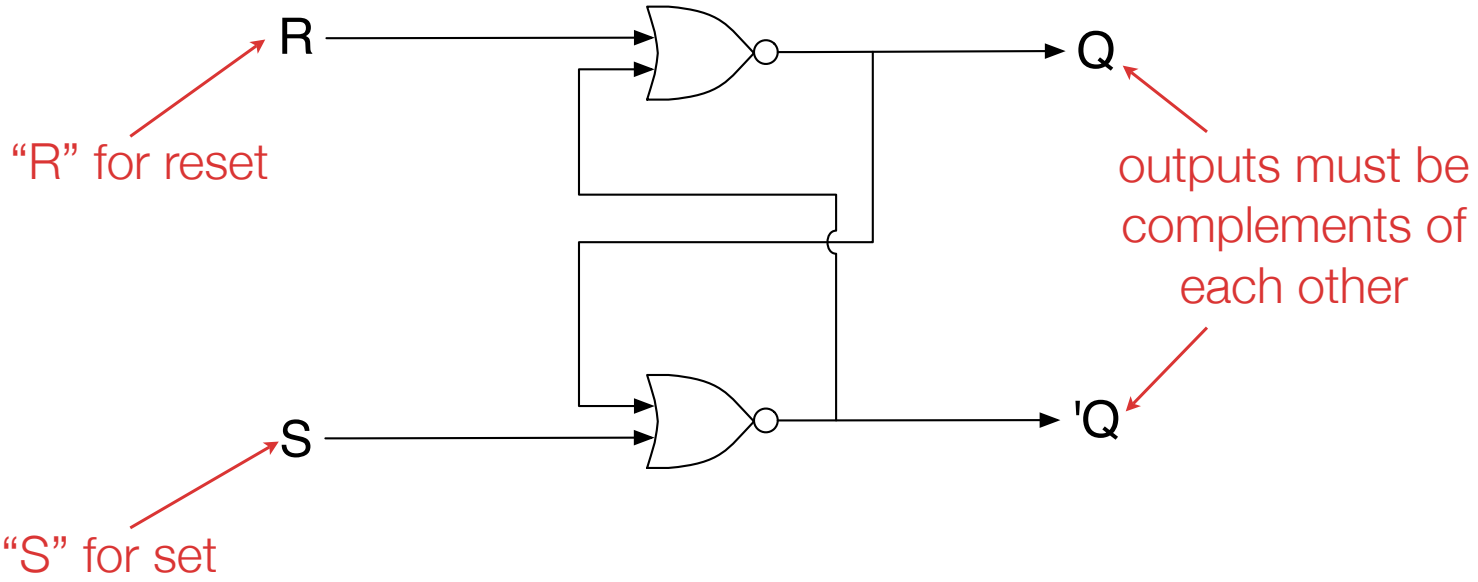
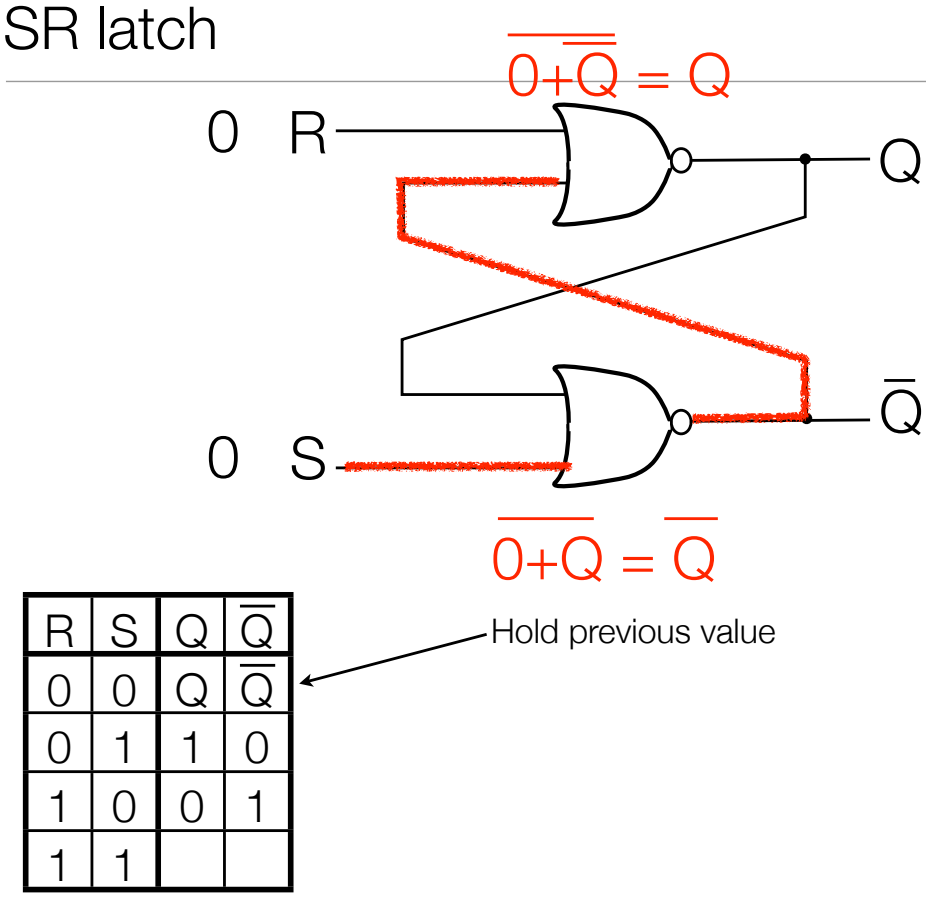


Figure: Source: Mano & Kime

The simplest sequential logic element: The set/reset (SR) latch



R	S	Q	$\overline{Q}$
0	0	Q	$\overline{Q}$
0	1	1	0
1	0	0	1
1	1		

Figure: Source: Mano & Kime

6 transistor SRAM cell

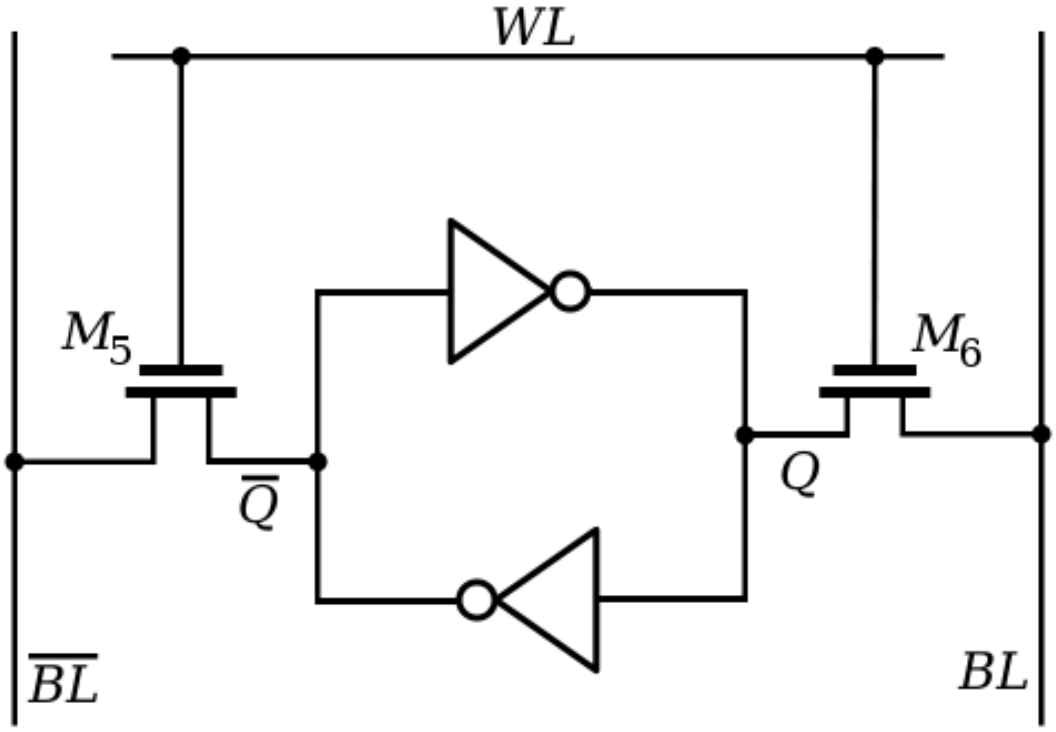
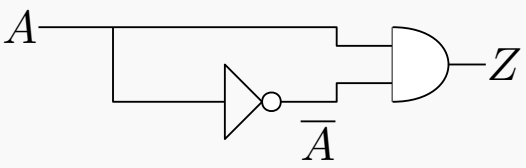


Figure: Source: Wikimedia

Asynchronous / Synchronous circuits

Timing

Circuit:



Voltages over time: