Slide 1

CS 240 Spring 2020
Foundations of Computer Systems
Ben Wood

WELLESLEY

Combinational Logic

Karnaugh maps
Building blocks: encoders, decoders, multiplexers

Abstraction!

Slide 2

But first…

Recall: *sum of products*

logical sum (OR)

of products (AND)

of inputs or their complements (NOT).

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

Construct with:
- 1 code detector per 1-valued output row
- 1 large OR of all code detector outputs

Is it minimal?

Slide 3

Gray Codes = reflected binary codes

Alternate binary encoding
designed for electromechanical switches and counting.

$$00 \quad 01 \mid 11 \quad 10$$
$$0 \quad\ 1 \mid 2 \quad\ 3$$

$$000 \quad 001 \quad 011 \quad 010 \mid 110 \quad 111 \quad 101 \quad 100$$
$$0 \quad\ 1 \quad\ 2 \quad\ 3 \mid 4 \quad\ 5 \quad\ 6 \quad\ 7$$

How many bits change when incrementing?

Slide 4

Karnaugh Maps: find (minimal) sums of products

ex

A	B	C	D	F(A, B, C, D)
0	0	0	0	0
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	0
0	1	1	0	1
0	1	1	1	0
1	0	0	0	1
1	0	0	1	1
1	0	1	0	1
1	0	1	1	1
1	1	0	0	1
1	1	0	1	1
1	1	1	0	1
1	1	1	1	0

gray code order →

| CD	00	01	11	10
00	0	0	0	0
01	0	0	0	1
11	1	1	0	1
10	1	1	1	1

AB

1. Cover exactly the 1s by drawing a (minimum) number of maximally sized rectangles whose dimensions (in cells) are powers of 2. (They may overlap or wrap around!)
2. For each rectangle, make a *product* of the inputs (or complements) that are 1 for all cells in the rectangle. (*minterms*)
3. Take the *sum* of these products.

Voting again with Karnaugh Maps

ex

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

Goal for next 2 weeks: **Simple Processor**

Instruction Fetch and Decode → Registers → ALU → Memory
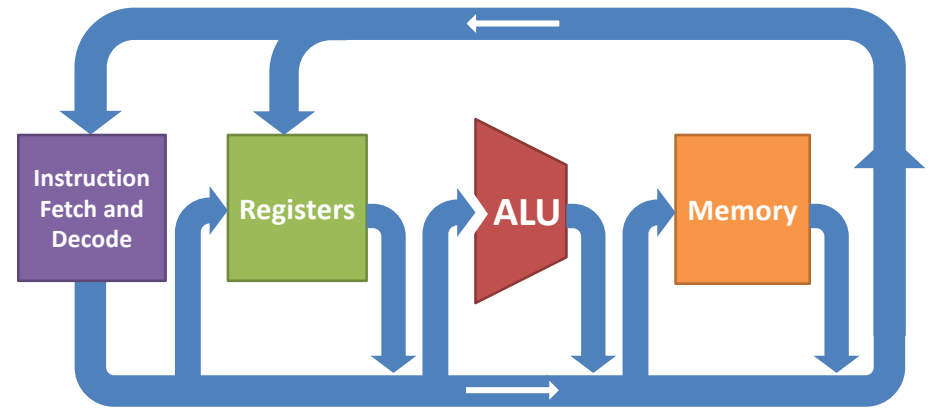
Toolbox: Building Blocks

Abstraction!

Microarchitecture

Processor datapath

Instruction Decoder
Arithmetic Logic Unit

Memory

Digital Logic

Adders
Multiplexers
Demultiplexers
Encoders
Decoders

Registers
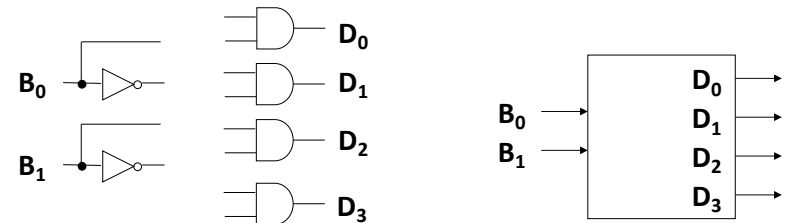
Flip-Flops
Latches

Gates

Devices (transistors, etc.)

Decoders

ex

Decodes input number, asserts corresponding output.
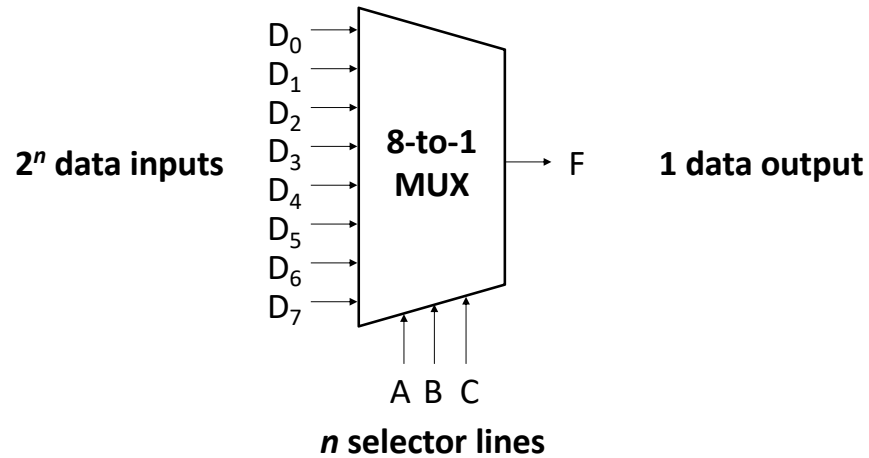
n-bit input (an unsigned number)
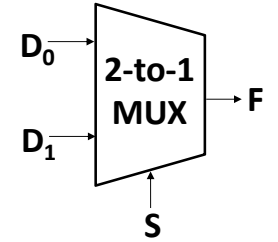
2^n outputs

Built with code detectors.

B_0 — D_0, D_1, D_2, D_3

B_1

B_0 → D_0, D_1, D_2, D_3
B_1

Multiplexers

Select one of several inputs as output.

2^n data inputs

8-to-1 MUX

F

1 data output

D_0
D_1
D_2
D_3
D_4
D_5
D_6
D_7

A B C

n selector lines

Build a 2-to-1 MUX from gates

ex

If S=0, then F=D_0.
If S=1, then F=D_1.

D_0

2-to-1 MUX

D_1

F

S

1. Construct the truth table.

2. Build the circuit.

8-to-1 MUX

D_0
D_1
D_2
D_3
D_4
D_5
D_6
D_7

F

A $\bar{A}$ B $\bar{B}$ C $\bar{C}$

A B C
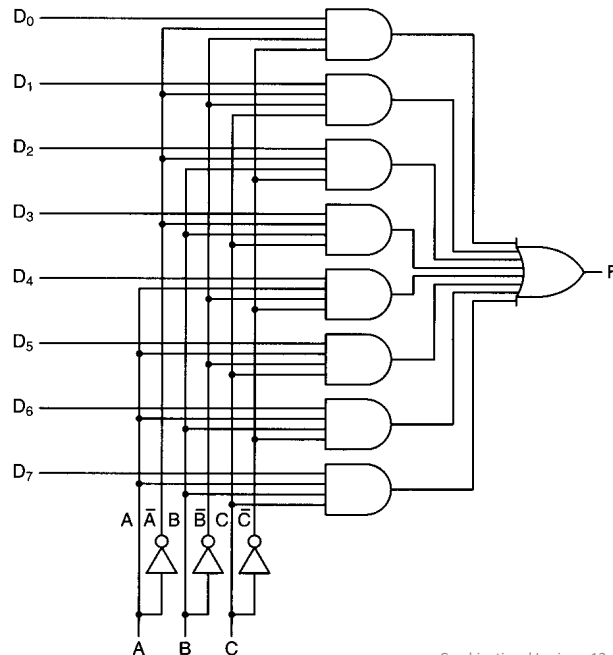
Costume idea: MUX OX

MUX + voltage source = truth table

A	B	C	M
0	0	0	**0**
0	0	1	**0**
0	1	0	**0**
0	1	1	**1**
1	0	0	**0**
1	0	1	**1**
1	1	0	**1**
1	1	1	**1**

0
1
2
3 **8-to-1 MUX**
4
5
6
7

M

A B C
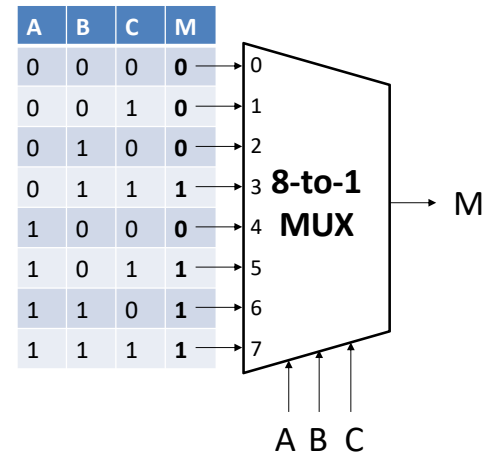
Buses and Logic Arrays

A bus is a collection of data lines
treated as a single logical signal.

= fixed-width value

Array of logic elements applies
same operation to each bit in a bus.

= bitwise operator