Circuits & Numbers

Kevin Walsh
CS 3410, Spring 2010
Computer Science
Cornell University

See: P&H Chapter 2.4, 2.5, 3.2, C.5

Office Hours

HW1

CSUGLab

How to implement a desired function?

a	b	c	out	minterm
0	**0**	**0**	**0**	$\bar{a}\,\bar{b}\,\bar{c}$
0	**0**	**1**	**1**	$\bar{a}\,\bar{b}\,c$
0	**1**	**0**	**0**	$\bar{a}\,b\,\bar{c}$
0	**1**	**1**	**1**	$\bar{a}\,b\,c$
1	**0**	**0**	**0**	$a\,\bar{b}\,\bar{c}$
1	**0**	**1**	**1**	$a\,\bar{b}\,c$
1	**1**	**0**	**0**	$a\,b\,\bar{c}$
1	**1**	**1**	**0**	$a\,b\,c$

sum of products:

- OR of all minterms where out=1

corollary: *any* combinational circuit *can be* implemented in two levels of logic (ignoring inverters)

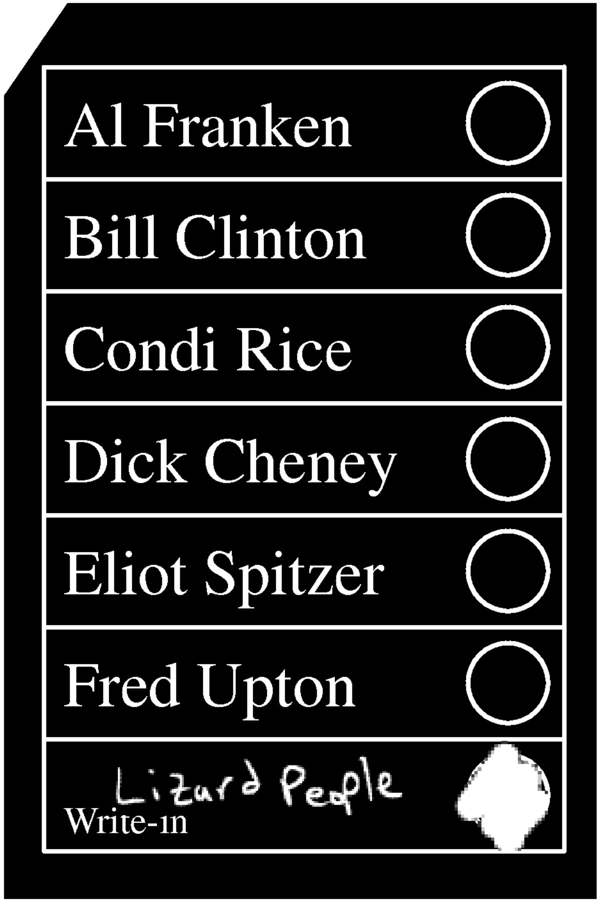
How does one find the most efficient equation?

- Manipulate algebraically until…?
- Use Karnaugh maps (optimize visually)
- Use a software optimizer

For large circuits

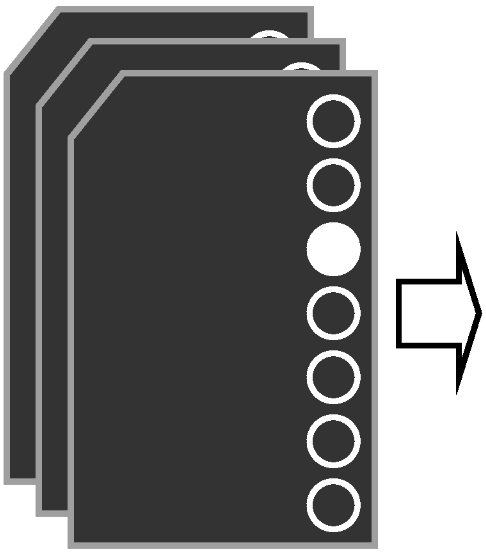
- Decomposition & reuse of building blocks

- Voting Machine!
  - optical scan (thanks FL)



- Assume:
  - vote is recorded on paper by filling a circle
  - fixed number of choices
  - don’t worry about “invalids”

5 Essential Components?



Ballots

The 3410 optical scan vote ~~counter~~ reader machine

- Input: paper with at exactly one mark
- Datapath: process current ballot
- Output: a number the supervisor can record
- Memory & control: none for now

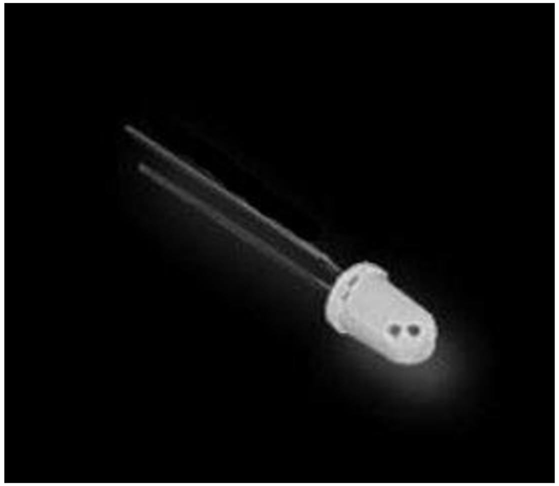
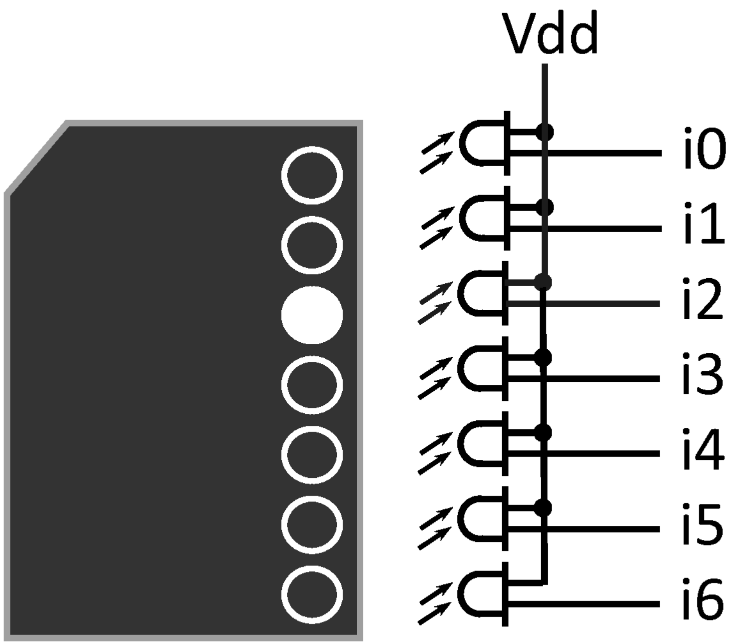


Photo-sensitive transistor

- photons replenish gate depletion region
- can distinguish dark and light spots on paper

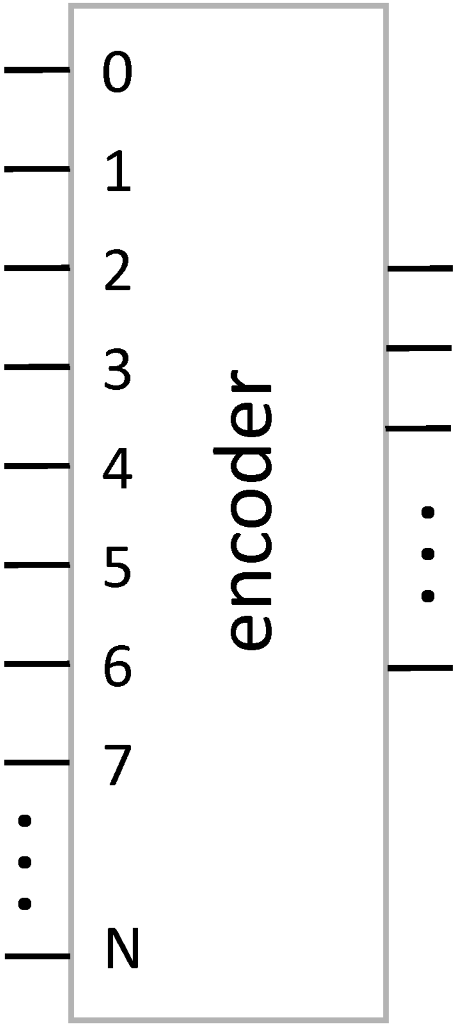


- Use array of N sensors for voting machine input

7-Segment LED

- photons emitted when electrons fall into holes


detect
N
8



N might be large

- Routing wires is expensive

More efficient encoding?

6 3 7

$$\underline{\quad 6 \quad}\ \underline{\quad 3 \quad}\ \underline{\quad 7 \quad}$$

$$10^2\ 10^1\ 10^0$$

- Base 10 - Decimal
- Just as easily use other bases
  - Base 2 - Binary
  - Base 8 - Octal
  - Base 16 - Hexadecimal

Counting

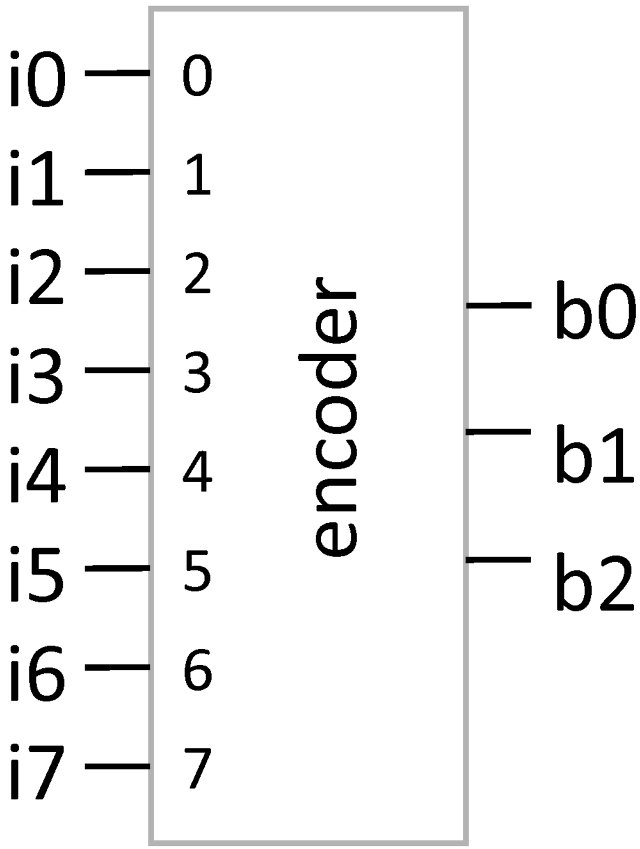
- Base conversion via repetitive division
  - Divide by base, write remainder, move left with quotient

- Base conversion via repetitive division
  - Divide by base, write remainder, move left with quotient

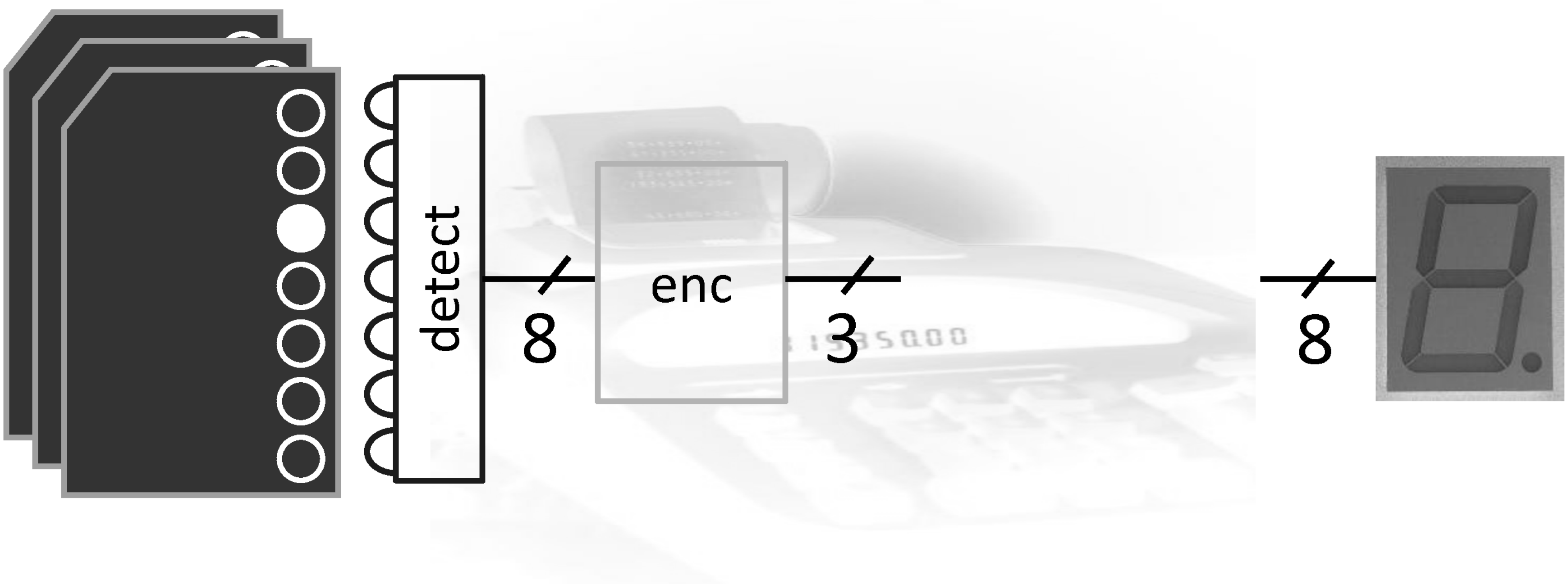
- Base conversion via repetitive division
  - Divide by base, write remainder, move left with quotient

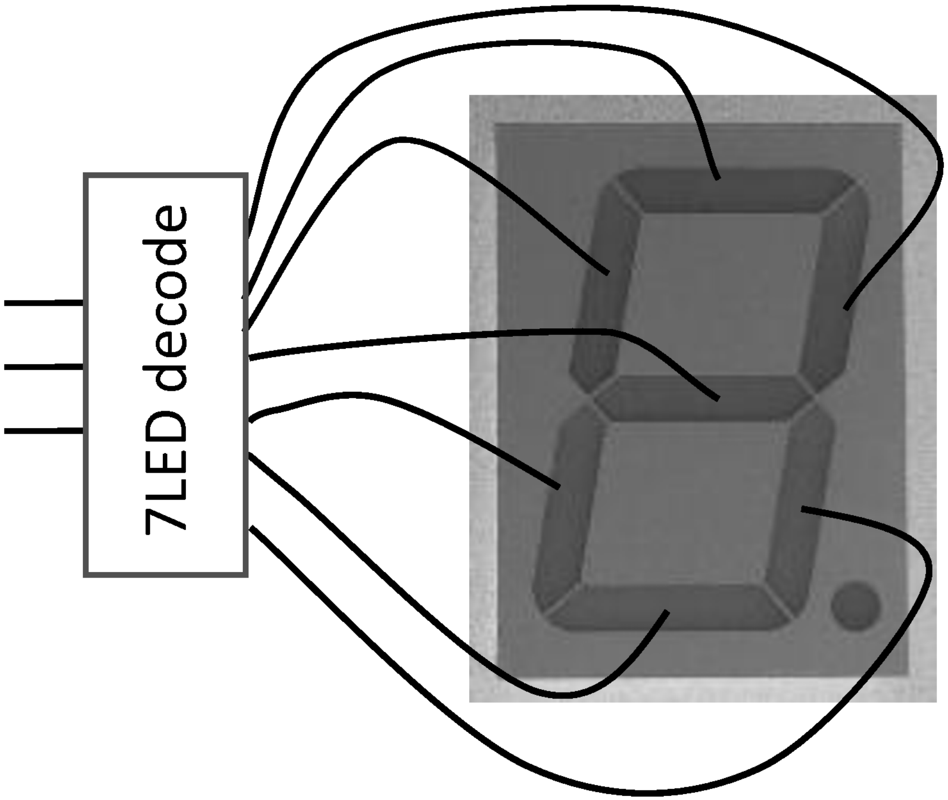
Implementation . . .

– assume 8 choices, exactly one mark detected



3-bit encoder
(8-to-3)


detect
8
enc
3
8



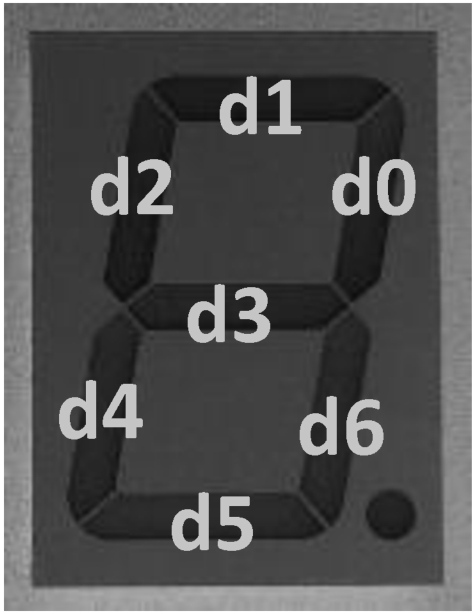
3 inputs

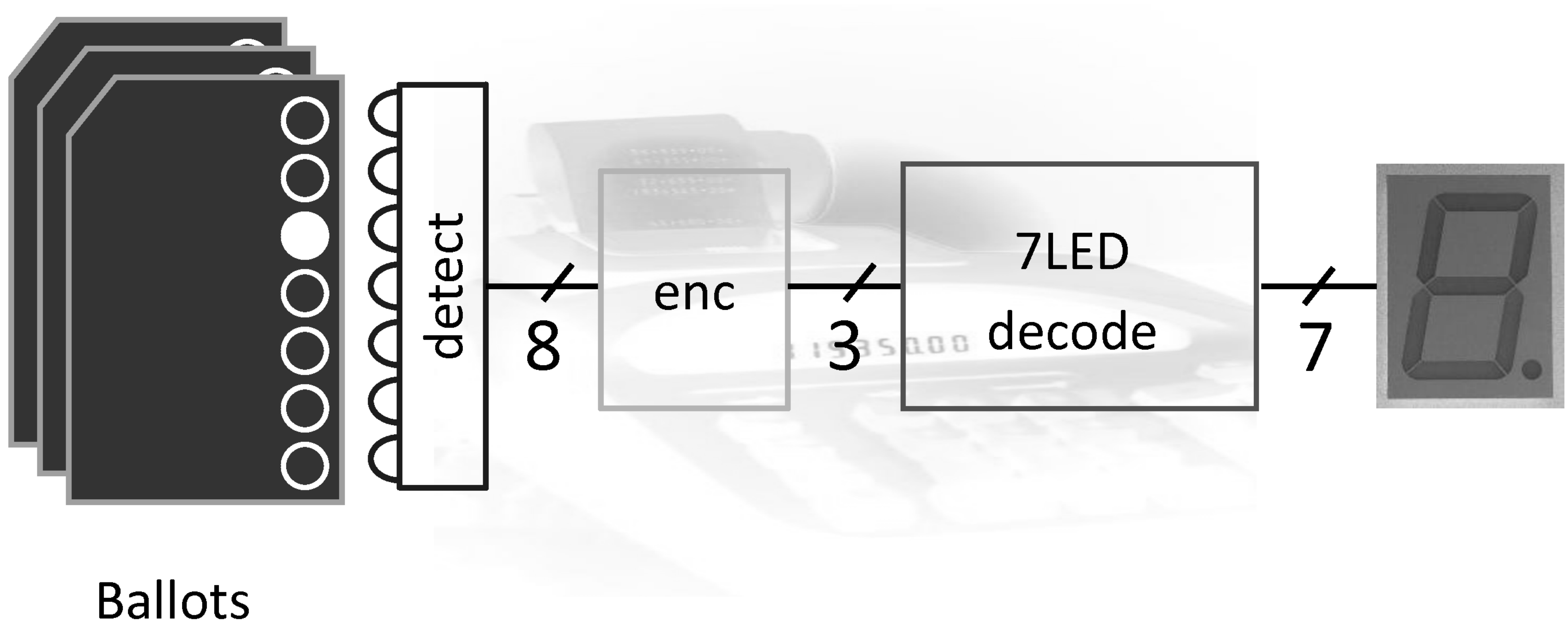
- encode 0 – 7 in binary

7 outputs

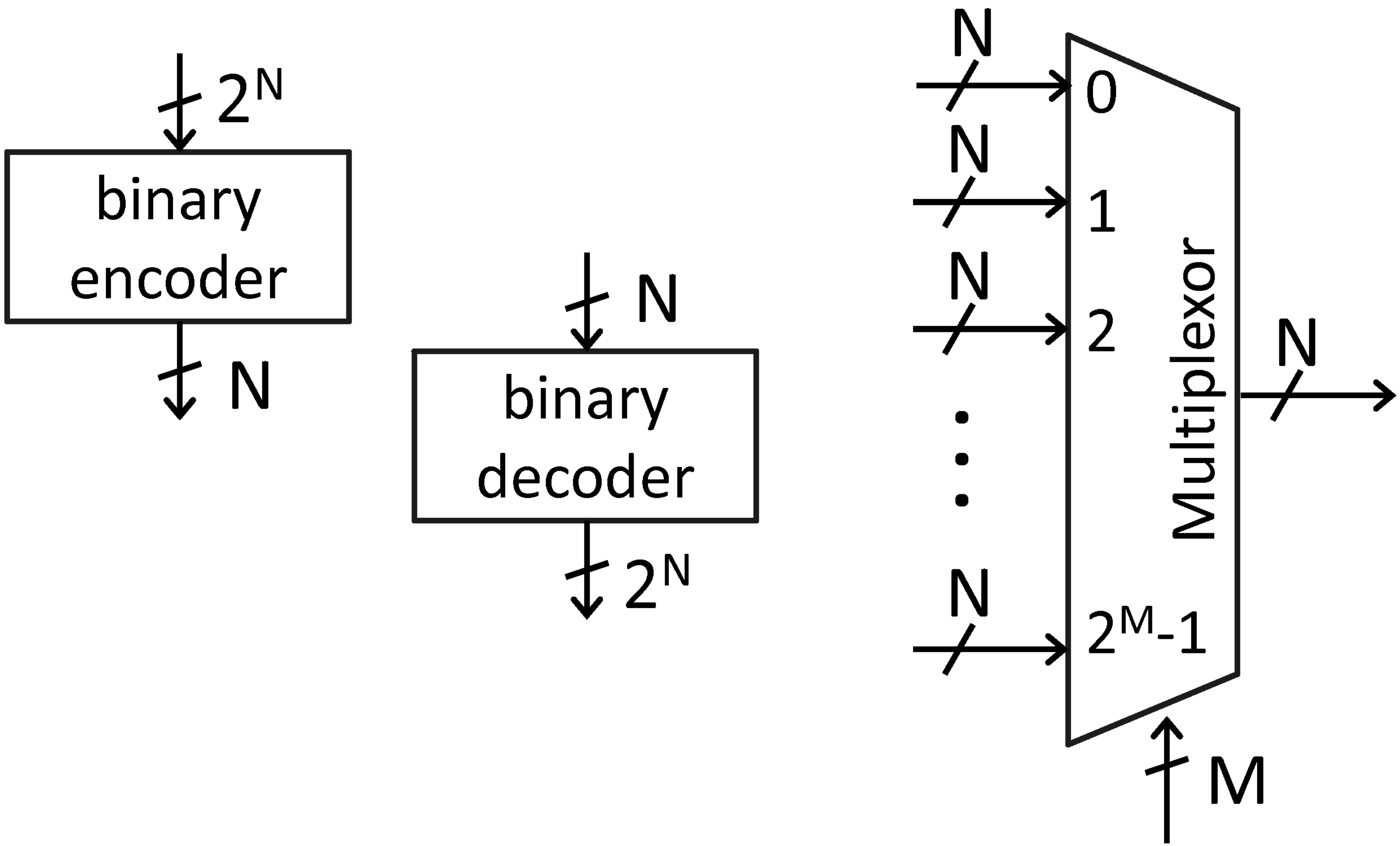
- one for each LED

b2	b1	b0	d6	d5	d4	d3	d2	d1	d0
0	0	0	1	1	1	0	1	1	1
0	0	1	1	0	0	0	0	0	1
0	1	0	0	1	1	1	0	1	1
0	1	1	1	1	0	1	0	1	1
1	0	0	1	0	0	1	1	0	1
1	0	1	1	1	0	1	1	1	0
1	1	0	1	1	1	1	1	1	0
1	1	1	1	0	0	0	0	1	1





The 3410 optical scan
vote ~~counter~~ reader
machine

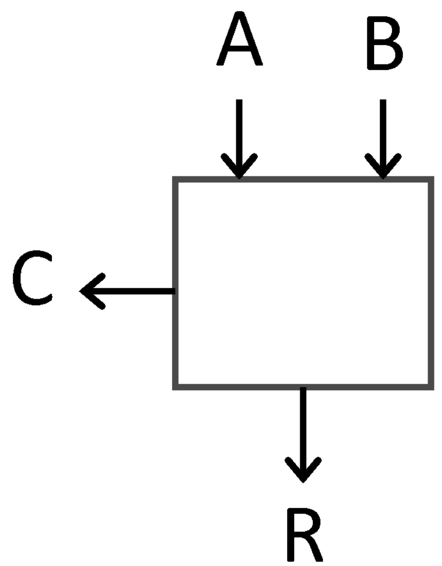

2^N
binary encoder
N
N
binary decoder
2^N
N
0
N
1
N
2
N
2^M-1
Multiplexor
N
M

$$
\begin{array}{r}
183 \\
+\ 254 \\
\hline
\end{array}
$$

Addition works the same way regardless of base

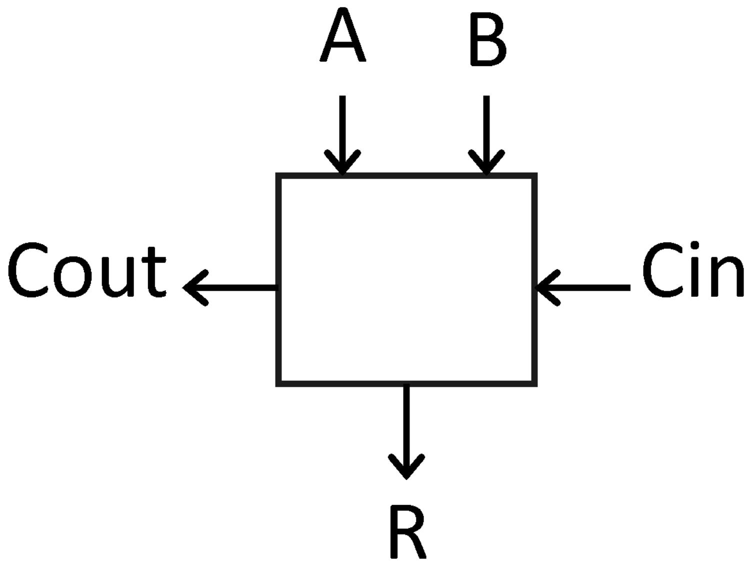
- Add the digits in each position
- Propagate the carry

$$
\begin{array}{r}
001110 \\
+\ 011100 \\
\hline
\end{array}
$$



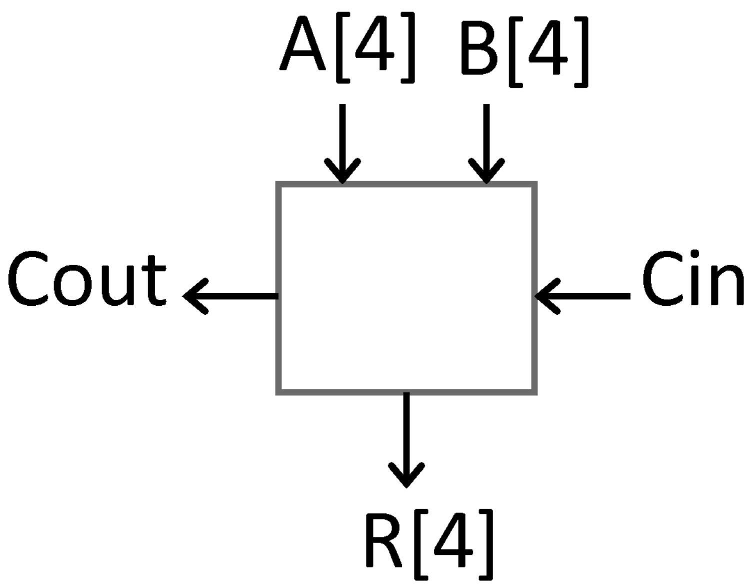
Half Adder

- Adds two 1-bit numbers
- Computes 1-bit result and 1-bit carry



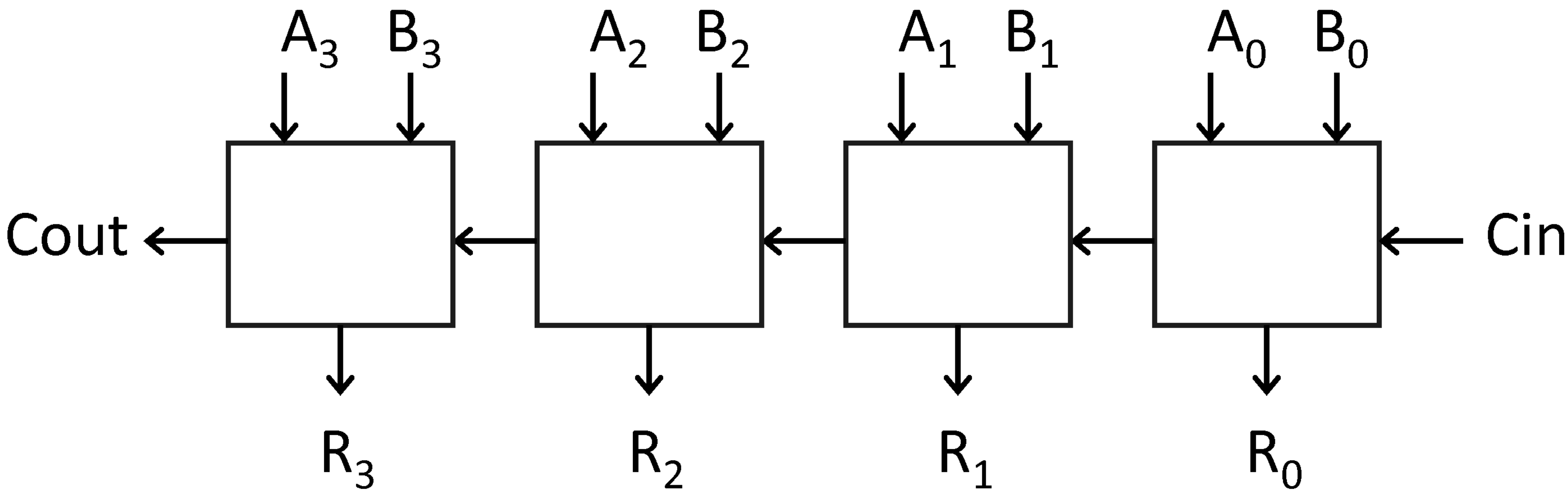
Full Adder

- Adds three 1-bit numbers
- Computes 1-bit result and 1-bit carry
- Can be cascaded



4-Bit Full Adder

- Adds two 4-bit numbers and carry in
- Computes 4-bit result and carry out
- Can be cascaded



A_3
B_3
A_2
B_2
A_1
B_1
A_0
B_0
Cout
Cin
R_3
R_2
R_1
R_0

We can now implement any combinational (combinatorial) logic circuit

- Decompose large circuit into manageable blocks
  - Encoders, Decoders, Multiplexors, Adders, ...
- Design each block
  - Binary encoded numbers for compactness
- Can implement circuits using NAND or NOR gates
- Can implement gates using use P- and N-transistors