CS 316: Logic

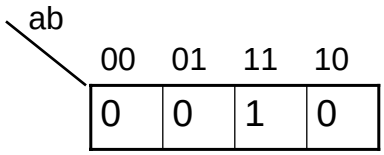
Kavita Bala
Fall 2007
Computer Science
Cornell University

Karnaugh maps

- Encoding of the truth table where adjacent cells differ in only one bit

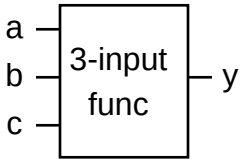
a	b	out
0	0	0
0	1	0
1	0	0
1	1	1

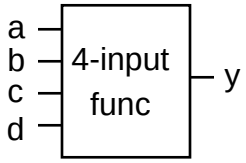
truth table for AND

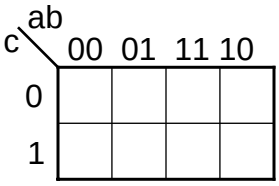


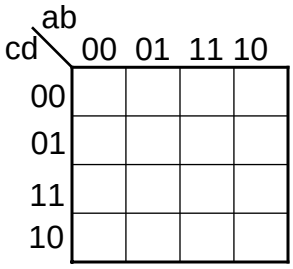
Corresponding Karnaugh map

Bigger Karnaugh Maps



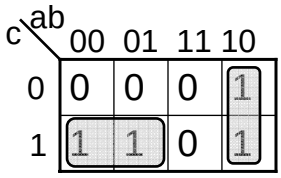






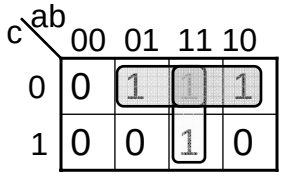
Minimization with Karnaugh maps (2)

a	b	c	out
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	0

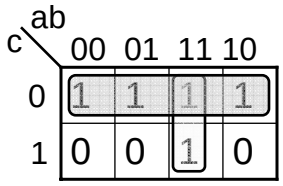


- Karnaugh map minimization
  - Cover all 1's
  - Group adjacent blocks of 2^n 1's that yield a rectangular shape
  - Encode the common features of the rectangle
    - out = $a\bar{b} + \bar{a}c$

Karnaugh Minimization Tricks (1)

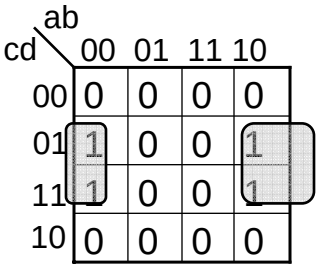


- Minterms can overlap
  - out = $b\bar{c} + a\bar{c} + ab$

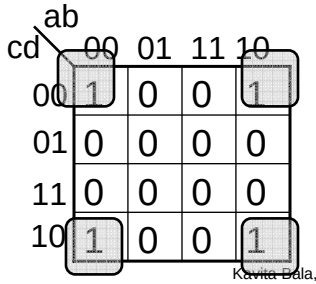


- Minterms can span 2, 4, 8 or more cells
  - out = $\bar{c} + ab$

Karnaugh Minimization Tricks (2)

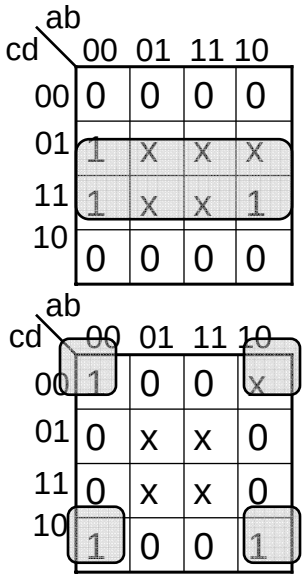


- The map wraps around
  - out = $\bar{b}\bar{d}$



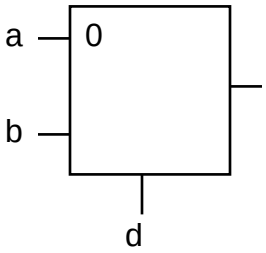
  - out = $\bar{b}\bar{d}$

Karnaugh Minimization Tricks (3)



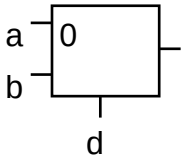
- "Don't care" values can be interpreted individually in whatever way is convenient
  - assume all x's = 1
  - out = d
  - assume middle x's = 0
  - assume 4th column x = 1
  - out = $\bar{b}\bar{d}$

Multiplexer



- A multiplexer selects between multiple inputs
  - out = a, if d = 0
  - out = b, if d = 1
- Build truth table
- Build Karnaugh map
- Derive logic diagram

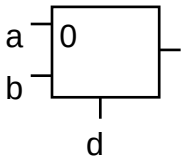
Multiplexer Implementation



- Build a truth table

a	b	d	out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Multiplexer Implementation

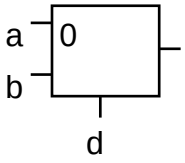


- Build the Karnaugh map

d \ ab	00	01	11	10
0	0	0	1	1
1	0	1	1	0

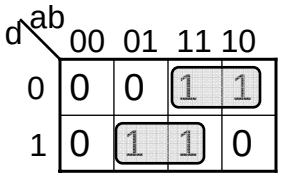
a	b	d	out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

Multiplexer Implementation



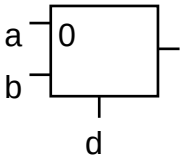
a	b	d	out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

- Derive minimal logic equation



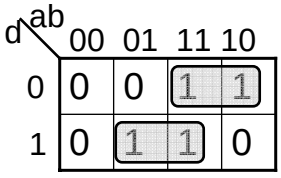
– out = $a\bar{d} + bd$

Multiplexer Implementation

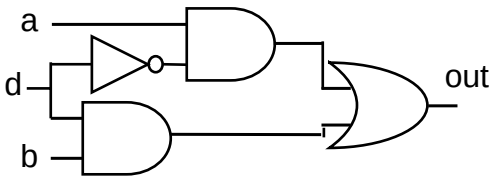


a	b	d	out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	1
1	0	0	1
1	0	1	0
1	1	0	1
1	1	1	1

- Draw the circuit



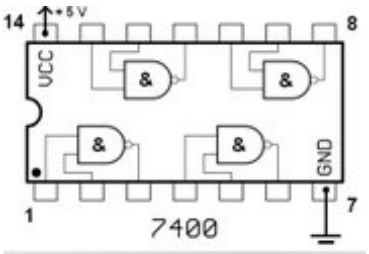
– out = $a\bar{d} + bd$

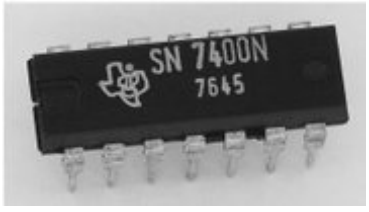


Summary

- We can now implement any logic circuit
  - Use P- and N-transistors to implement NAND or NOR gates
  - Use NAND or NOR gates to implement the logic circuit
  - Efficiency: use Karnaugh maps to find the minimal terms required

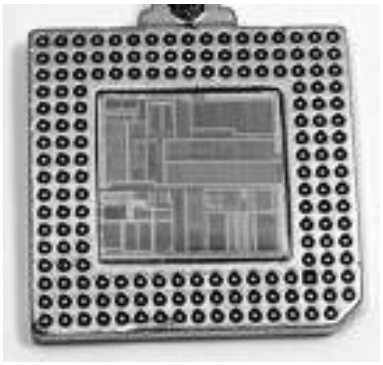
Logic Gates





- One can buy gates separately
  - ex. 74xxx series of integrated circuits
  - cost ~$1 per chip, mostly for packaging and testing
- Cumbersome, but possible to build devices using gates put together manually

Integrated Circuits



- Or one can manufacture a complete design using a custom mask
- Intel Pentium has approximately 125 million transistors

Voting machine

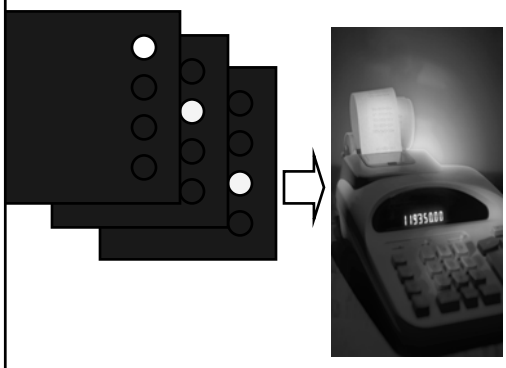
- Build something interesting
- A voting machine
  - Elections are coming up!
- Assume:
  - A vote is recorded on a piece of paper,
  - by punching out a hole
  - there are at most 7 choices
  - we will not worry about “hanging chads” or “invalids”

Voting machine

- For now, let's just display the numerical identifier to the ballot supervisor
  - we won't do counting yet, just decoding
  - we can use four photo-sensitive transistors to find out which hole is punched out

- A photo-sensitive transistor detects the presence of light
- Photo-sensitive material triggers the gate

Ballot Reading

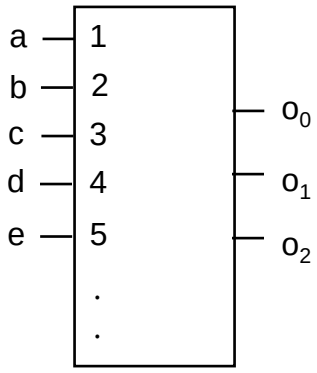


- Input: paper with a hole in it
- Out: number the ballot supervisor can record

Ballots

The 316 vote recording machine

Encoders



A 3-bit encoder
(7-to-3)
(5 inputs shown)

- N sensors in a row
- Want to distinguish which of the N sensors has fired
- Want to represent the firing sensor number in compact form
  - N might be large
  - Only one wire is on at any time
  - Silly to route N wires everywhere, better to encode in log N wires

Number Representations

$$37$$

$$10^1\ 10^0$$

- Decimal numbers are written in base 10
  - $3 \times 10^1 + 7 \times 10^0 = 37$
- Just as easily use other bases
  - Base 2 - “Binary”
  - Base 8 - “Octal”
  - Base 16 – “Hexadecimal”

Number Representations

37

10^1 10^0

- Base conversion via repetitive division
  - Divide by base, write remainder, move left with quotient
  - Sanity check with 37 and 10

Binary Representation

- 37 = 32 + 4 + 1

0100101

2^6 2^5 2^4 2^3 2^2 2^1 2^0

64 32 16 8 4 2 1

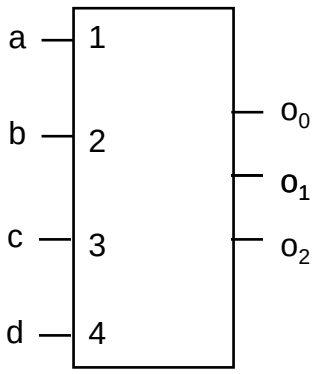
Hexadecimal Representation

$$\underline{2}\ \underline{5}$$

$16^1\ 16^0$

- 37 decimal = $(25)_{16}$
- Convention
  - Base 16 is written with a leading 0x
  - 37 = 0x25
- Need extra digits!
  - 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, A, B, C, D, E, F
- Binary to hexadecimal is easy
  - Divide into groups of 4, translate groupwise into hex digits

Encoder Truth Table

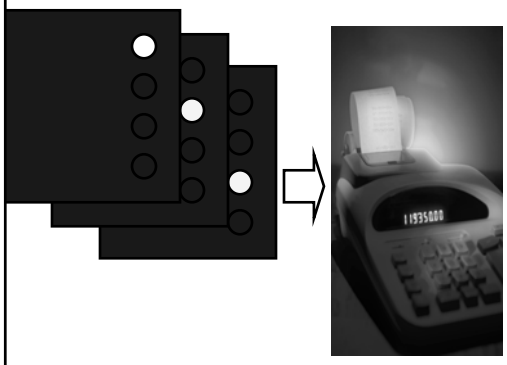


A 3-bit encoder with 4 inputs for simplicity

a	b	c	d		o2	o1	o0
0	0	0	0		0	0	0
1	0	0	0		0	0	1
0	1	0	0		0	1	0
0	0	1	0		0	1	1
0	0	0	1		1	0	0

- $o2 = \bar{a}\bar{b}\bar{c}d$
- $o1 = \bar{a}\bar{b}\bar{c}d + \bar{a}\bar{b}c\bar{d}$
- $o0 = \bar{a}\bar{b}\bar{c}d + \bar{a}\bar{b}c\bar{d}$

Ballot Reading

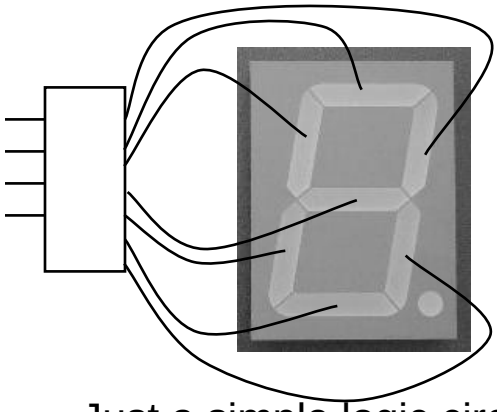


Ballots

The 316 voting machine

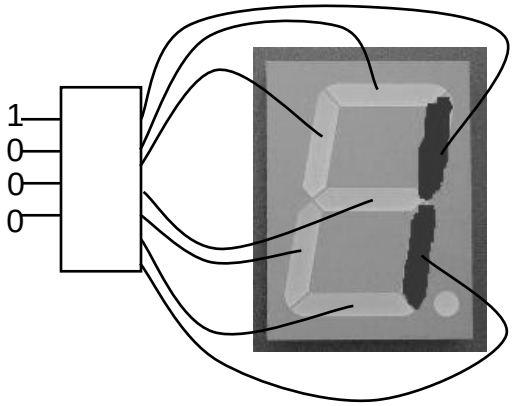
- Ok, we built first half of the machine
- Need to display the result

7-Segment LED Decoder



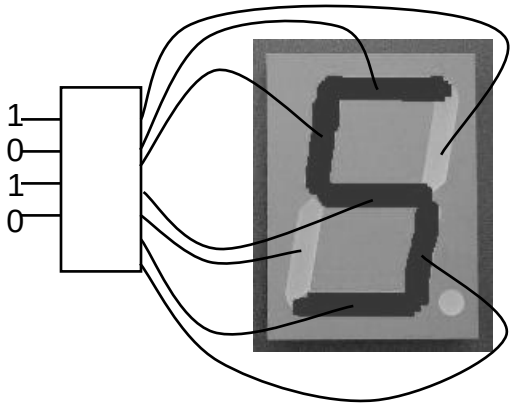
- 4 inputs encoded in binary
- 7 outputs, each driving an independent, rectangular LED
- Can display numbers
- Just a simple logic circuit
- Write the truth table

7-Segment LED Decoder



- 4 inputs encoded in binary
- 8 outputs, each driving an independent, rectangular LED
- Can display numbers

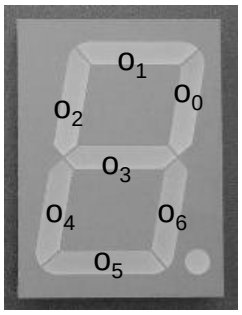
7-Segment LED Decoder



- 4 inputs encoded in binary
- 8 outputs, each driving an independent, rectangular LED
- Can display numbers

7-Segment Decoder Truth Table

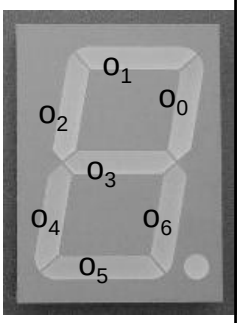
i_3	i_2	i_1	i_0		o_0	o_1	o_2	o_3	o_4	o_5	o_6
0	0	0	0		1	1	1	0	1	1	1
0	0	0	1		1	0	0	0	0	0	1
0	0	1	0		1	1	0	1	1	1	0
0	0	1	1		1	1	0	1	0	1	1
0	1	0	0		1	0	1	1	0	0	1
0	1	0	1		0	1	1	1	0	1	1
0	1	1	0		0	1	1	1	1	1	1
0	1	1	1		1	1	0	0	0	0	0
1	0	0	0		1	1	1	1	1	1	1
1	0	0	1		1	1	1	1	0	1	1
1	0	1	0	A	1	1	1	1	1	0	1
1	0	1	1	B	0	0	1	1	1	1	1
				C							



Exercise: find the error(s) in this truth table

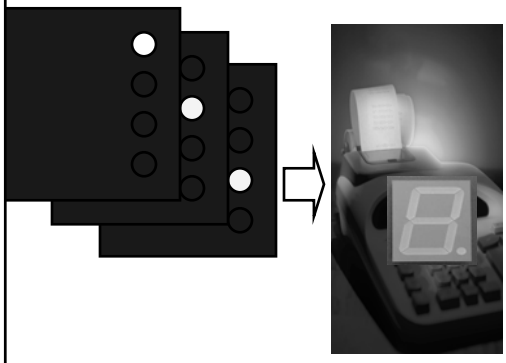
7-Segment Decoder Truth Table

i_3	i_2	i_1	i_0		o_0	o_1	o_2	o_3	o_4	o_5	o_6
0	0	0	0		1	1	1	0	1	1	1
0	0	0	1		1	0	0	0	0	0	1
0	0	1	0		1	1	0	1	1	1	0
0	0	1	1		1	1	0	1	0	1	1
0	1	0	0		1	0	1	1	0	0	1
0	1	0	1		0	1	1	1	0	1	1
0	1	1	0		0	1	1	1	1	1	1
0	1	1	1		1	1	0	0	0	0	1
1	0	0	0		1	1	1	1	1	1	1
1	0	0	1		1	1	1	1	0	1	1
1	0	1	0	a	1	1	1	1	1	0	1
1	0	1	1	b	0	0	1	1	1	1	1
1	1	0	0	c	0	1	1	0	1	1	0
1	1	0	1	d	1	0	0	1	1	1	1
1	1	1	0	e	0	1	1	1	1	1	0
1	1	1	1	f	0	1	1	1	1	0	0



Ballot Reading

- Done!



Ballots

The 316 voting machine