CS 220 – COMPUTER ARCHITECTURE

WEEK #2: BOOLEAN LOGIC (PART II)

CANONICAL REPRESENTATION

Whodunit story: Each suspect may or may not have an alibi (*a*), a motivation to commit the crime (*m*), and a relationship to the weapon found in the scene of the crime (*w*). The police decides to focus attention only on suspects for whom the proposition Not(*a*) And (*m* Or *w*) is true.

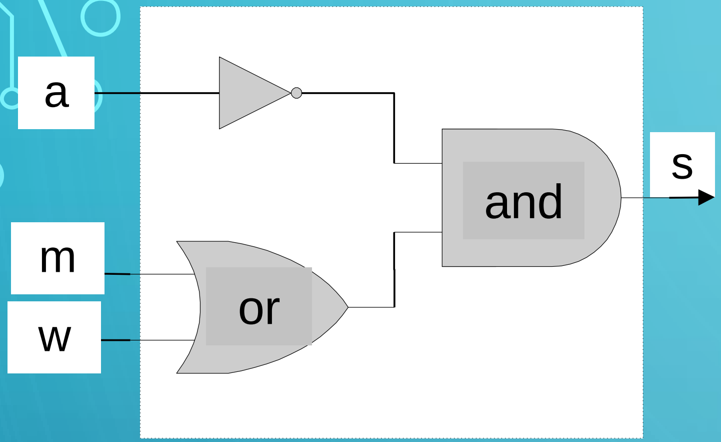
Truth table of the "suspect" function $s(a,m,w) = \bar{a} \cdot (m + w)$

a	m	w	*minterm*	suspect(a,m,w)= not(a) and (m or w)
0	0	0	$m_0 = \bar{a}\bar{m}\bar{w}$	0
0	0	1	$m_1 = \bar{a}\bar{m}w$	1
0	1	0	$m_2 = \bar{a}m\bar{w}$	1
0	1	1	$m_3 = \bar{a}mw$	1
1	0	0	$m_4 = a\bar{m}\bar{w}$	0
1	0	1	$m_5 = a\bar{m}w$	0
1	1	0	$m_6 = am\bar{w}$	0
1	1	1	$m_7 = amw$	0

Canonical form: $s(a,m,w) = \bar{a}\,\bar{m}\,w + \bar{a}\,m\,\bar{w} + \bar{a}\,m\,w$

CANONICAL REPRESENTATION

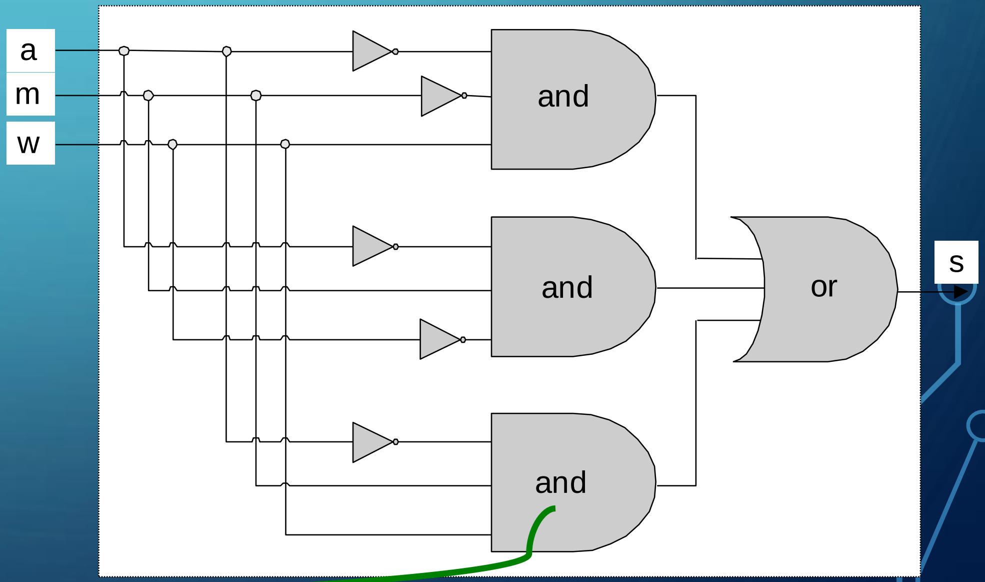
$$s(a,m,w) = \bar{a} \cdot (m + w)$$



<--Efficient Solution

May not be the most efficient solution, but Canonical Representation at least gets you one!

$$s(a,m,w) = \bar{a}\,\bar{m}\,w + \bar{a}\,m\,\bar{w} + \bar{a}\,m\,w$$



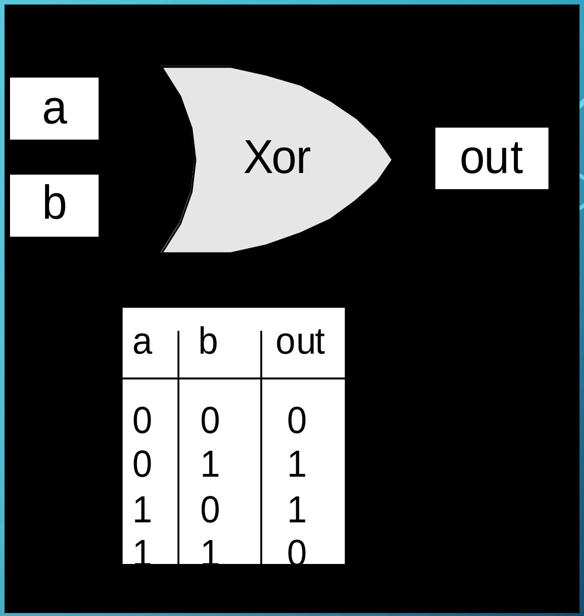
Note: This is what we would call And3Way, as it has 3 data pins

XOR EXAMPLE

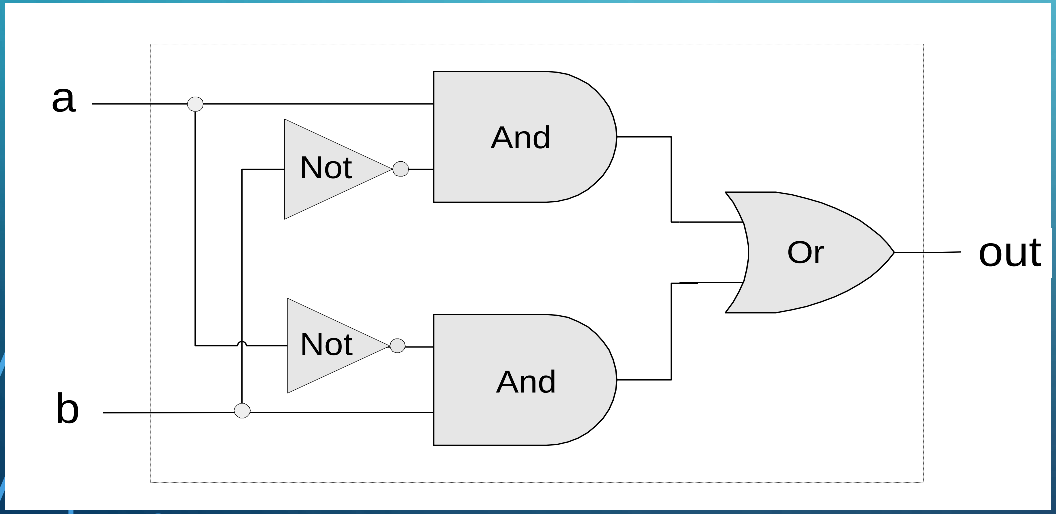
Minterms: $xor(a,b) = \bar{a}b + a\bar{b}$

a	b	out	minterms
0	0	0	-
0	1	1	$\bar{a}b$
1	0	1	$a\bar{b}$
1	1	0	-

Interface



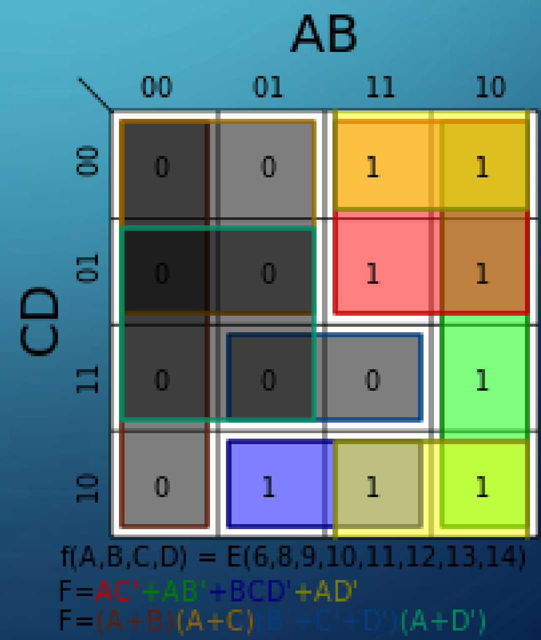
Implementation



Xor(a,b) = Or(And(a,Not(b)),And(Not(a),b)))

CANONICAL REPRESENTATION

- Solutions may be inefficient
- Can we simplify?
  - Need to know Discrete Math
  - Karnaugh Maps (K-Maps) [Pronounced: Car-no]
    - Help simplify complicated expressions
      - using AND, OR, NOT only
    - 5 steps!



KARNAUGH MAPS

- **1)** Start with your Truth Table (example: XOR)
- **2)** Draw a K-Map based on the number of input pins:
  - Each input gets a (set of) column/row
    - Each possibility gets its own column/row
  - Example: Xor has inputs a and b, each can be 0 or 1
- **3)** Fill in K-Map according to Truth Table

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

		a	
		0	1
b	0		
	1		

Fill K-Map!

		a	
		0	1
b	0	0	1
	1	1	0

KARNAUGH MAPS

- K-Map is filled, now to simplify!
- **4)** Find minimum # of groups (of 1's only) that cover all 1's
- Only groups allowed (check in this order):
  - *4x2*
  - *2x2*
  - *2x1*
  - *1x1*
- **5)** Write minterms of *groups*:

$$xor(a,b) = \overline{a}b + a\overline{b}$$

		a	
		0	1
b	0	0	1
	1	1	0

- This is the most simplified solution!
  - Not much different from what we came up with…
  - Remember, only does AND, NOT, OR*

KARNAUGH MAPS

- Boring example!
- Something more interesting:
- Write f(a,b,c,d) canonical representation:
- How many chips will it take to make?
- K-Map time!

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

KARNAUGH MAPS

- 1) Start with the truth table:
- 2) Draw the K-Map:

cd \ ab	00	01	11	10
00	1			1
01		1		
11	1	1	1	1
10	1	1	1	1

Notice the way they double up and are numbered!

Remember the order:

4x2
2x2
2x1
1x1

- 3) Fill it in
- 4) Find min # of groups to cover all 1's
  - You can do that?!
- 5) Write the simplified canonical representation

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

KARNAUGH MAPS

c d \ ab	$\bar{a}\bar{b}$	$\bar{a}b$	ab	$a\bar{b}$
$\bar{c}\bar{d}$	1			1
$\bar{c}d$		1		
cd	1	1	1	1
$c\bar{d}$	1	1	1	1

- Writing simplified canonical form:
  - Each group becomes a minterm
    - Write term if unpaired
  - OR each of the groups together
  - For 1x1 this was easy…
- Group #1 (2x1): $\bar{a}bd$
- Group #2 (2x2): $\bar{b}\bar{d}$
- Group #3 (4x2): c
- Simplified Canonical Form:

$$F(a,b,c,d) = \bar{b}\bar{d} + \bar{a}bd + c$$

How many chips for this one? How does it compare to the first one?

KARNAUGH MAPS

- **Canonical Representation:**

$$F(a,b)=\bar{a}\bar{b}\bar{c}\bar{d}+\bar{a}\bar{b}c\bar{d}+\bar{a}\bar{b}cd+\bar{a}b\bar{c}d+\bar{a}bc\bar{d}+\bar{a}bcd+a\bar{b}\bar{c}\bar{d}+a\bar{b}c\bar{d}+a\bar{b}cd+abc\bar{d}+abcd$$

  - 33 AND's
  - 10 OR's
  - 21 NOT's

  TOTAL = 64 gates

*think

- 24 AND's //why 24?
- 10 OR's
- 4 NOT's

TOTAL = gates

- **Simplified Canonical Form:**

$$F(a,b,c,d)=\bar{b}\bar{d}+\bar{a}bd+c$$

  - 3 AND's
  - 2 OR's
  - 3 NOT's

  TOTAL = 8 gates

*think = critically, creatively. This is usually difficult!

But this is how XOR can be done in 3 chips (hint: NAND)