Interesting things about XOR

- Recall that I mentioned **XOR** gates are useful in arithmetic circuits (e.g., to build adders) and other sorts of circuits.
- Sometimes, we can "discover" a **XOR** gate "buried" or "hidden" inside of a logic expression. If we can find it, it can help to reduce the implementation cost.
- Recall that **XOR** and **NXOR** operators perform the "odd" and "even" function, respectively.

Karnaugh maps (K-Maps) for **XOR** and **NXOR**

- K-Maps are not "pretty" in terms of optimization.
- Example for 4-input **XOR** and **NXOR**...

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

$$f = a \oplus b \oplus c \oplus d$$

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

$$f = \overline{a \oplus b \oplus c \oplus d}$$

- Implementing an **XOR** or **NXOR** using **AND**, **OR** and **NOT** would be a disaster.

Hunting for **XOR** and **NXOR**

- Sometimes we might have an expression which yields a K-Map that sort of looks like an **XOR**.
- Example of a 4-input function $f = f(a, b, c, d)$...

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

- We can perform Boolean algebra...

$$
\begin{aligned}
f &= a'b'c'd + a'bc'd + a'bcd + a'bcd' + abc'd + ab'c'd' + ab'cd + ab'cd' \\
&= a'b'c'd + abc'd + a'bc + ab'c + a'bd' + ab'd'
\end{aligned}
$$

- This is the minimum SOP and has a cost of 33.

Hunting for **XOR** and **NXOR**

- Try to continue the optimization...

$$
\begin{aligned}
f &= a'b'c'd + a'bc'd + a'bcd + a'bcd' + abc'd + ab'c'd' + ab'cd + ab'cd' \\
&= a'b'c'd + abc'd + a'bc + ab'c + a'bd' + ab'd' \\
&= (a'b' + ab)(c'd) + (a'b + ab')c + (a'b + ab')d' \\
&= \overline{(a \oplus b)}(c'd) + (a \oplus b)c + (a \oplus b)d' \\
&= \overline{(a \oplus b)}(c'd) + (a \oplus b)(c + d') \\
&= \overline{(a \oplus b)}(c'd) + (a \oplus b)\overline{(c'd)} \\
&= (a \oplus b) \oplus (c'd) \\
&= a \oplus b \oplus (c'd)
\end{aligned}
$$

- If we assume an 3-input **XOR** gate costs the same as any other 3-input gate, then this implementation costs 7.
- Although **XOR** are more expensive (in reality), this implementation is most certainly cheaper than the minimum SOP!

Hunting for **XOR** and **NXOR**

- Here's another example which is a bit different...

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

- We can observe that the input a is acting like a "gate" — when $a = 0$, f is an **XOR** and when $a = 1$, $f = 0$.
- In other words... $f = \overline{a}(b \oplus c \oplus d)$ which (again) is likely much cheaper than the minimum SOP implementation.
- In general, finding **XOR** operations hidden in general logic expressions can be quite difficult, but it's cool.