Function Simplification

- Goal: find an equivalent function with the minimum number of literals.

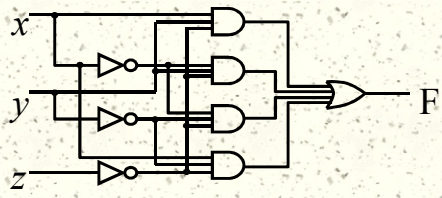
  Example: F(x,y,z) = xyz′+x′yz′+x′y′z′+xy′z′

  = yz′ + y′z′ = z′

- Why simplification?

  less hardware,

  same functionality



- How to simplify functions?

Design Example: An Encoder

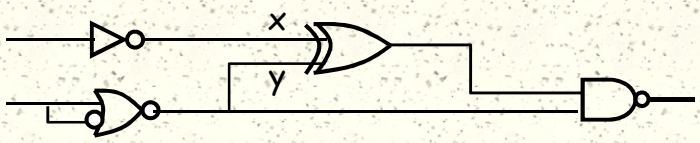
- A 3-input encoder that assigns a 2-bit code to each of the three different input combinations.
- Don't care conditions.
- Solutions:
  - a = x′yz′+xy′z′=(x⊕y)z′

    b = x′y′z + xy′z′=(x⊕z)y′
  - a = z′, b = y′
  - and a lot more

x	y	z	a	b
0	0	1	0	1
0	1	0	1	0
1	0	0	1	1

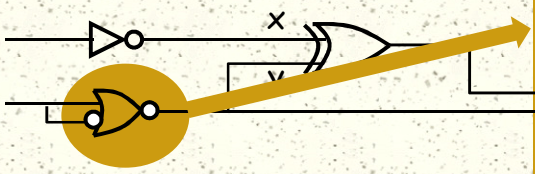
Where Are the Don't Cares: SDC

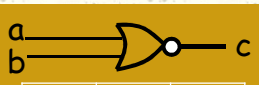
- **Satisfiability don't cares (SDC)** of a subcircuit/subsystem consist of all input patterns that will never occur.



Where Are the Don't Cares: SDC

- **Satisfiability don't cares (SDC)** of a subcircuit/subsystem consist of all input patterns that will never occur.
- Example: from definition, signal y cannot be 1.

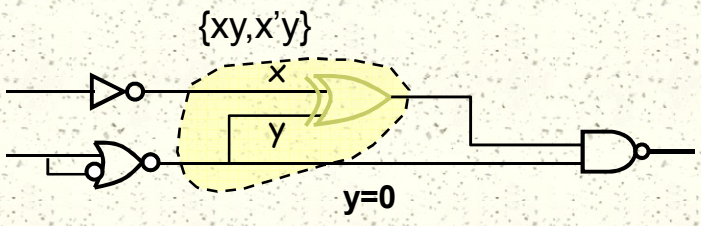




a	b	c
0	0	1
0	1	0
1	0	0
1	1	0

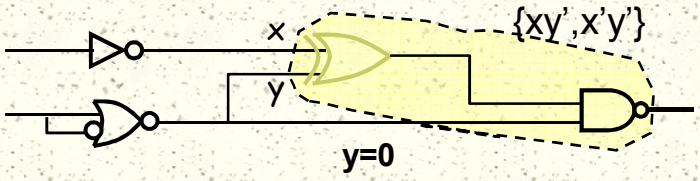
Where Are the Don't Cares: SDC

- **Satisfiability don't cares (SDC)** of a subcircuit/subsystem consist of all input patterns that will never occur.
- Example: for XOR, y input cannot be 1, so {x=1,y=1} and {x=0,y=1} will be SDCs.



Where Are the Don't Cares: ODC

- **Observability don't cares (ODC)** of a subcircuit/subsystem are the input patterns that represent situations when an output is not observed.
- Example: when y=0, output of the XOR cannot be observed. {x=1,y=0}, {x=0,y=0}



How Can SDCs and ODCs Help?

- SDCs: {x=1,y=1}, {x=0,y=1}
- ODCs: {x=1,y=0}, {x=0,y=0}
- All the four input combinations are don't care, so the XOR gate can be removed!