Lecture 12
Subtractors

Combinational Arithmetic Circuits

- Addition:
  - **Half Adder (HA).**
  - **Full Adder (FA).**
  - **Binary Adder**
  - **BCD(Decimal) Adder.**
- Subtraction:
  - **Half Subtractor.**
  - **Full Subtractor.**
- Multiplication:
  - **Binary Multipliers.**
- Comparator:
  - **Magnitude Comparator.**

Combinational Arithmetic Circuits

- Multiplexers
- Demultiplexers
- Encoders
- Decoders

Half Subtractor

- Subtracting a single-bit binary value Y from anther X (I.e. X -Y) produces a difference bit D and a borrow out bit B-out.
- This operation is called half subtraction and the circuit to realize it is called a half subtractor.

$D(X,Y) = \Sigma\,(1,2)$

$D = X'Y + XY'$

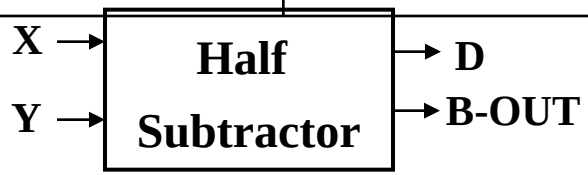
$D = X \oplus Y$

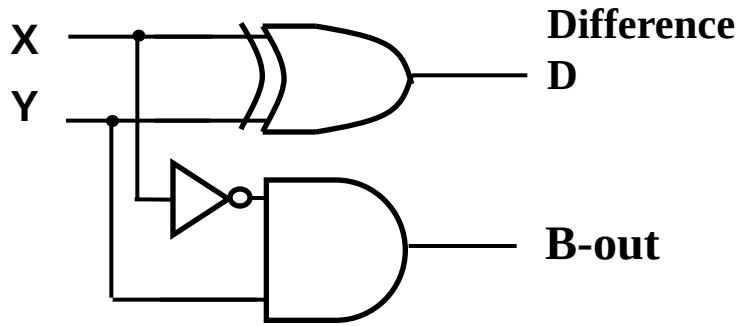
Half Subtractor Truth Table

Inputs		Outputs	
X	Y	D	B-out
0	0	0	0
0	1	1	1
1	0	1	0
1	1	0	0

$B\text{-out}(x, y, C\text{-in}) = \Sigma\,(1)$

$B\text{-out} = X'Y$





Binary Arithmetic Operations Subtraction

- Two binary numbers are subtracted by subtracting each pair of bits together with borrowing, where needed.
- Subtraction Example:

				0	0	1	1	1	1	1	0	0	Borrow
X	229			1	1	1	0	0	1	0	1		
Y -	46	-		0	0	1	0	1	1	1	0		
	183			1	0	1	1	0	1	1	1		

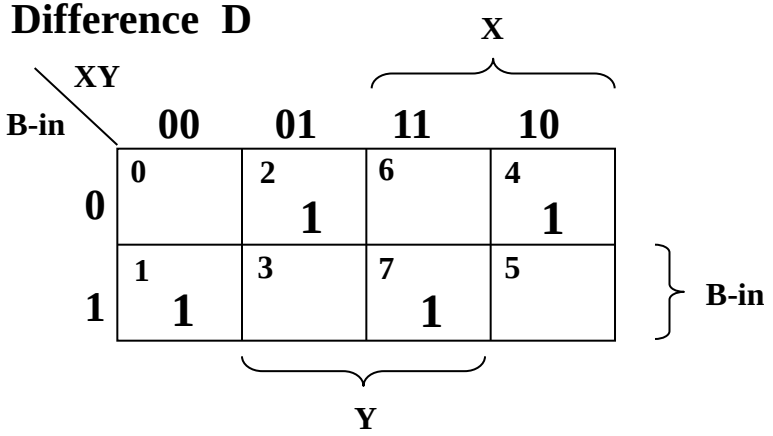
Full Subtractor

- Subtracting two single-bit binary values, Y, B-in from a single-bit value X produces a difference bit D and a borrow out B-out bit. This is called full subtraction.

Full Subtractor Truth Table

Inputs			Outputs	
X	Y	B-in	D	B-out
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	0	1
1	0	0	1	0
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

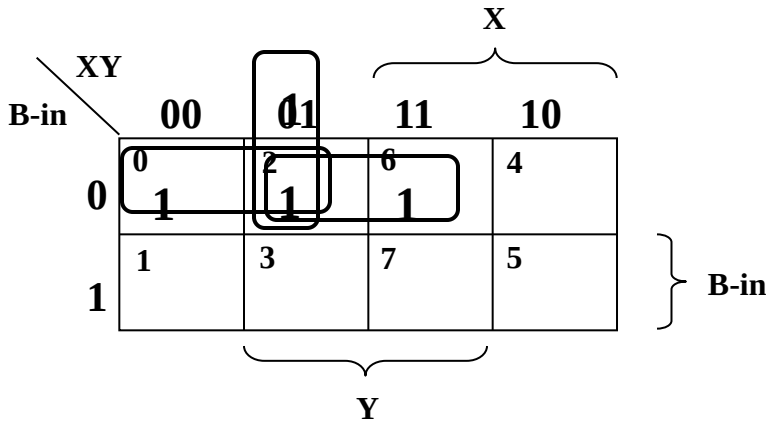
Difference D



$S = X'Y'(B\text{-in}) + XY'(B\text{-in})' + XY'(B\text{-in})' + XY(B\text{-in})$

$S = X \oplus Y \oplus (C\text{-in})$

Borrow B-out

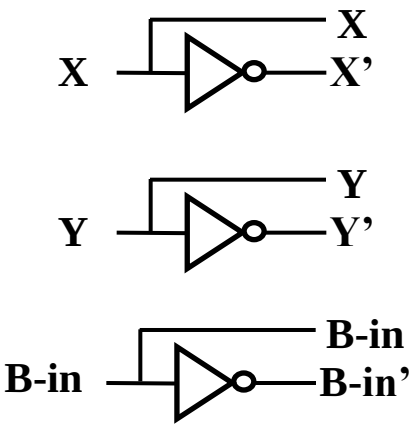


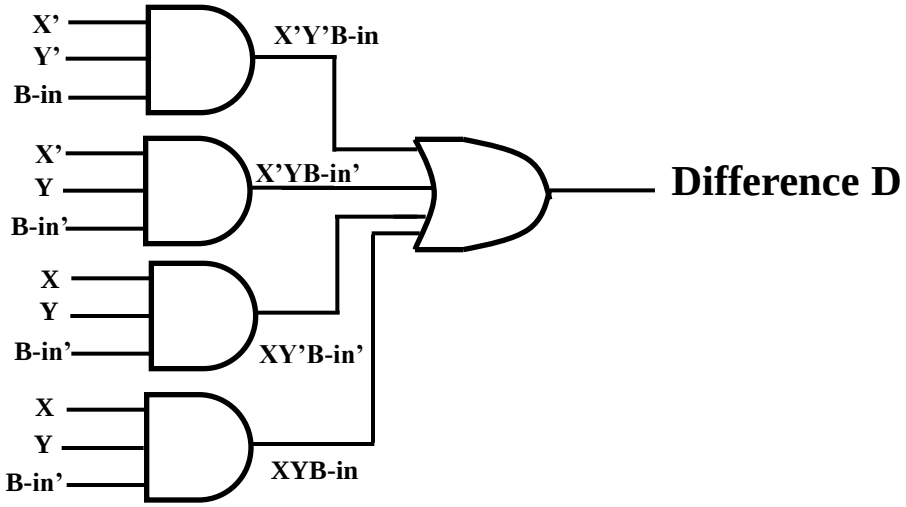
$S(X,Y, C\text{-in}) = \Sigma\ (1,2,4,7)$

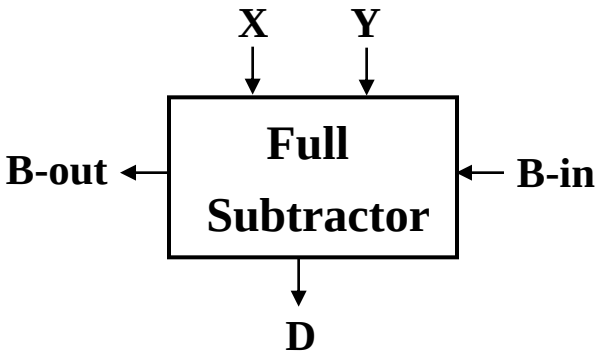
$C\text{-out}(x, y, C\text{-in}) = \Sigma\ (1,2,3,7)$

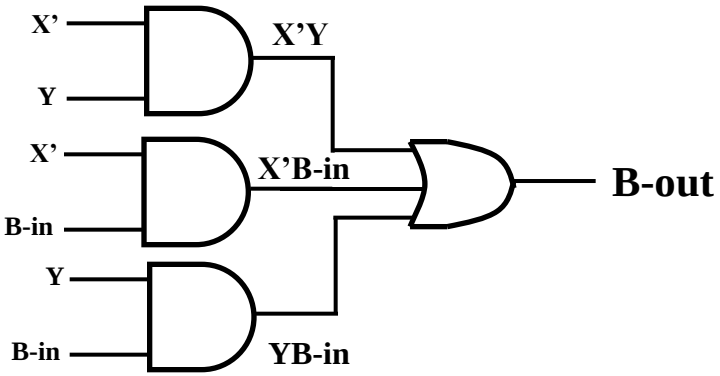
$B\text{-out} = X'Y + X'(B\text{-in}) + Y(B\text{-in})$

Full Subtractor Circuit Using AND-OR

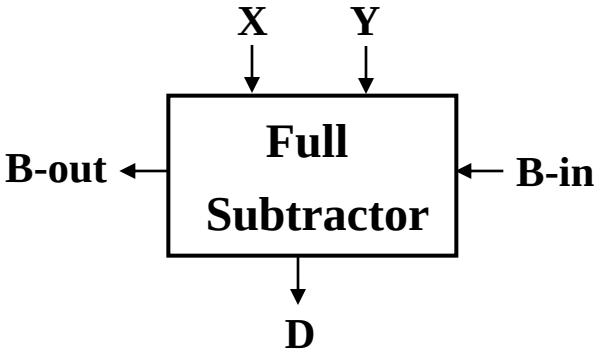


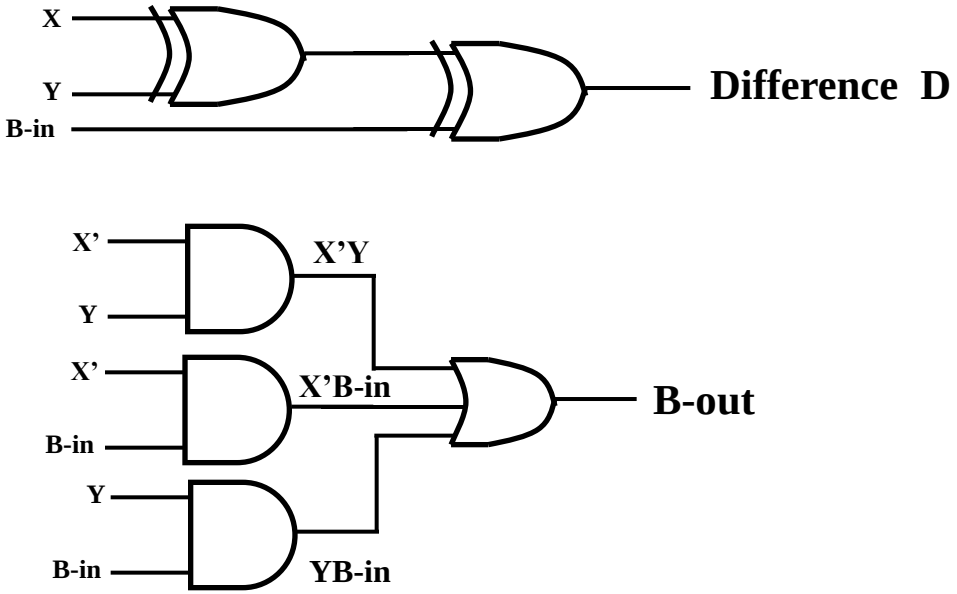






Full Subtractor Circuit Using XOR





n-bit Subtractors

An n-bit subtracor used to subtract an n-bit number Y from another n-bit number X (i.e X-Y) can be built in one of two ways:

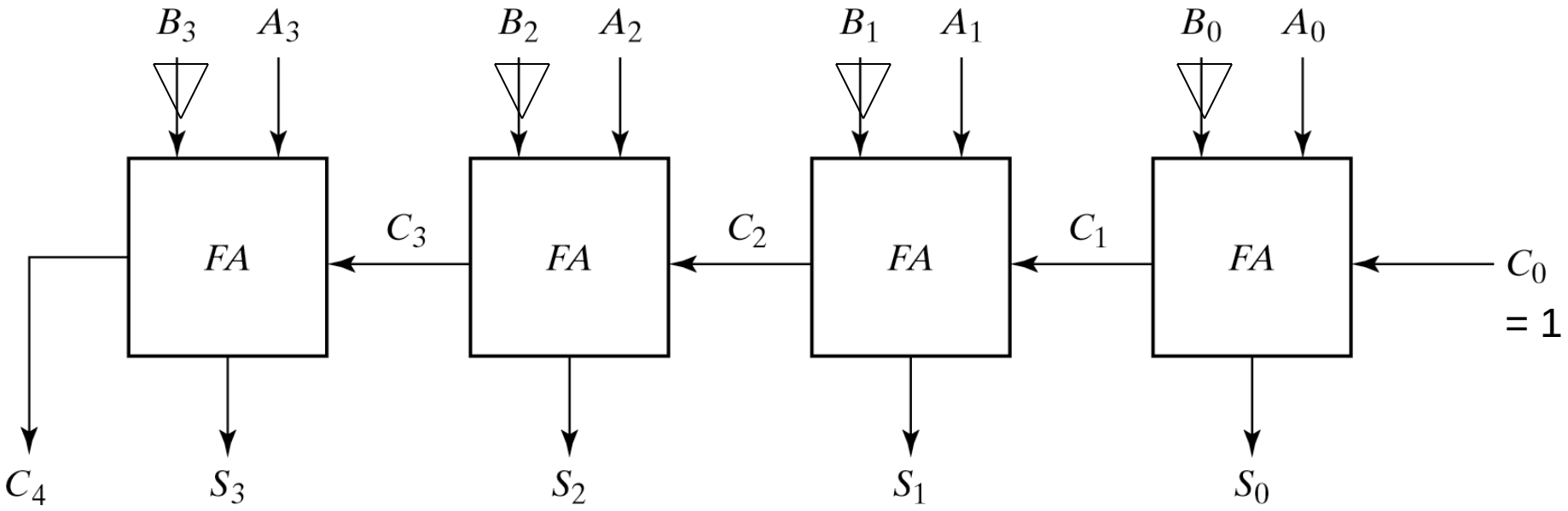
- By using n full subtractors and connecting them in series, creating a borrow ripple subtractor:
  - **Each borrow out B-out from a full subtractor at position j is connected to the borrow in B-in of the full subtracor at the higher position j+1.**

- By using an n-bit adder and n inverters:
  - **Find two's complement of Y by:**
    - Inverting all the bits of Y using the n inverters.
    - Adding 1 by setting the carry in of the least significant position to 1
  - **The original subtraction (X - Y) now becomes an addition of X to two's complement of Y using the n-bit adder.**

Binary Subtractor

- The subtrcation A – B can be done by taking the 2’s complement of B and adding it to A because A- B = A + (-B)
- It means if we use the inveters to make 1’s complement of B (connecting each Bi to an inverter) and then add 1 to the least significant bit (by setting carry C0 to 1) of binary adder, then we can make a binary subtractor.

4 bit 2's complement Subtractor



Adder Subtractor

- The addition and subtraction can be combined into one circuit with one common binary adder (see next slide).
- The mode M controls the operation. When M=0 the circuit is an adder when M=1 the circuit is subtractor. It can be don by using exclusive-OR for each Bi and M. Note that $1 \oplus x = x'$ and $0 \oplus x = x$

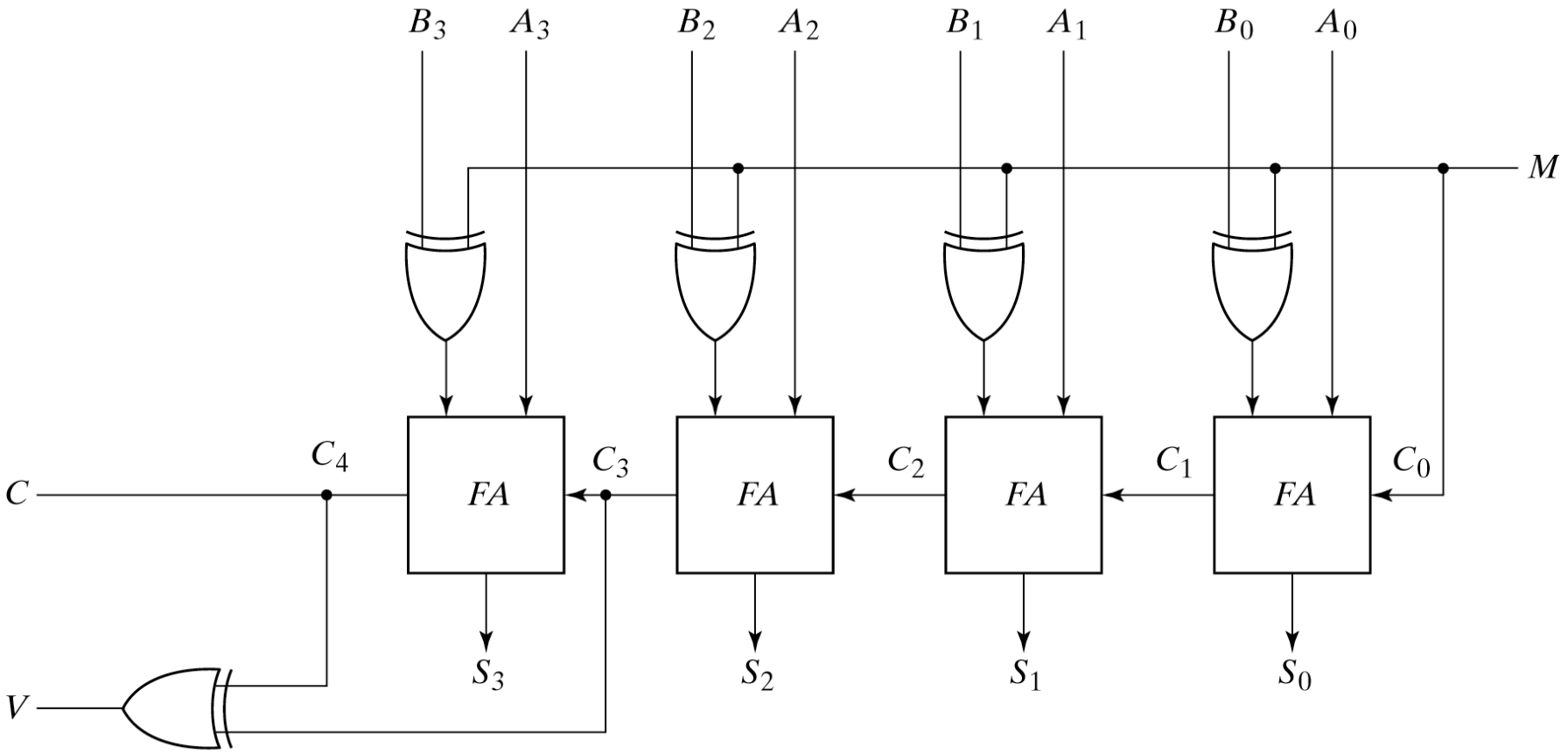
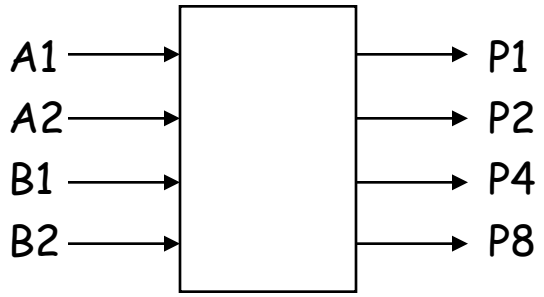


Fig. 4-13 4-Bit Adder Subtractor

Checking Overflow

- Note that in the previous slide if the numbers considered to be signed V detects overflow. V=0 means no overflow and V=1 means the result is wrong because of overflow
- Overflow can be happened when adding two numbers of the same sign (both negative or positive) and result can not be shown with the available bits. It can be detected by observing the carry into sign bit and carry out of sign bit position. If these two carries are not equal an overflow occurred. That is why these two carries are applied to exclusive-OR gate to generate V.

Design example: 2x2-bit multiplier

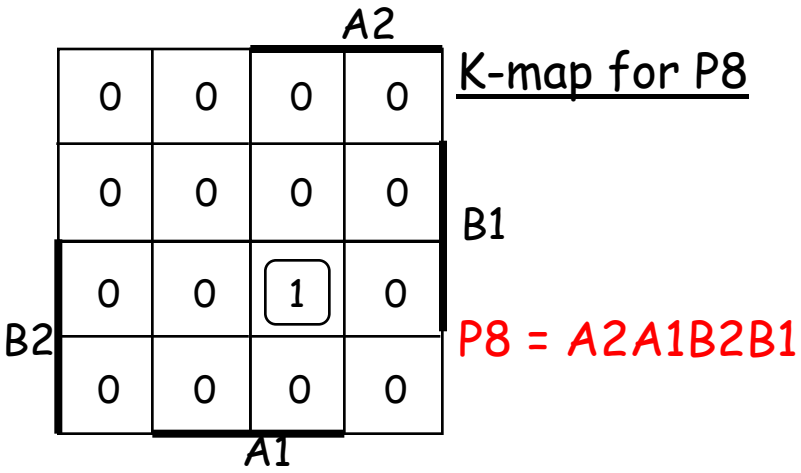


block diagram
and
truth table

A2	A1	B2	B1	P8	P4	P2	P1
0	0	0	0	0	0	0	0
		0	1	0	0	0	0
		1	0	0	0	0	0
		1	1	0	0	0	0
0	1	0	0	0	0	0	0
		0	1	0	0	0	1
		1	0	0	0	1	0
		1	1	0	0	1	1
1	0	0	0	0	0	0	0
		0	1	0	0	1	0
		1	0	0	1	0	0
		1	1	0	1	1	0
1	1	0	0	0	0	0	0
		0	1	0	0	1	1
		1	0	0	1	1	0
		1	1	1	0	0	1

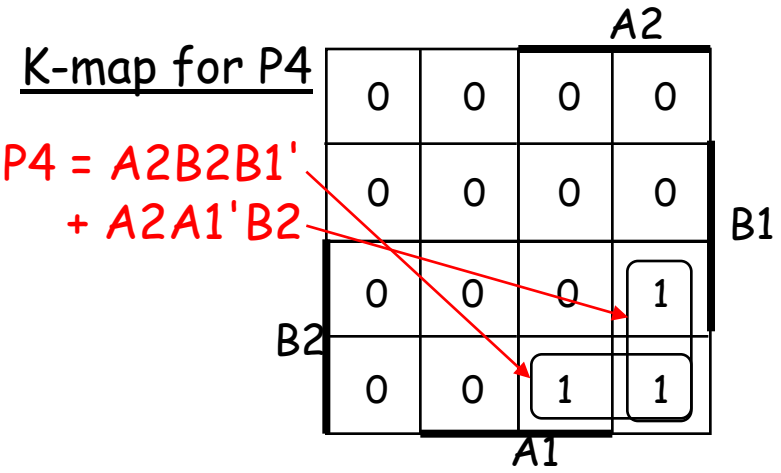
4-variable K-map
for each of the 4
output functions

Design example: 2x2-bit multiplier (cont'd)



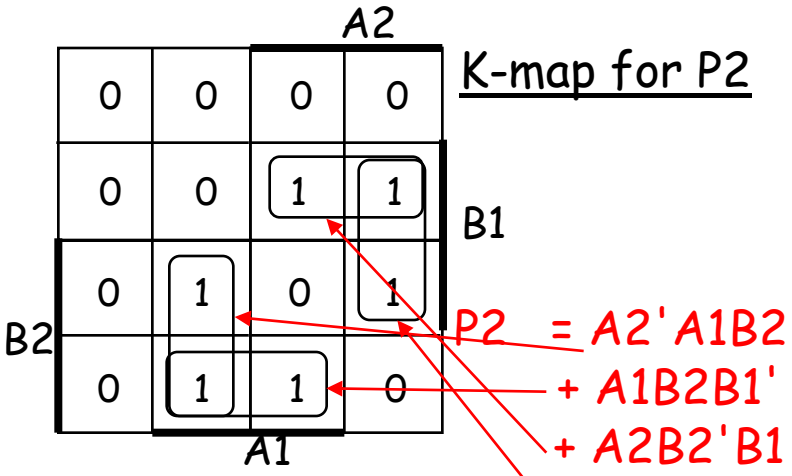
K-map for P8

P8 = A2A1B2B1



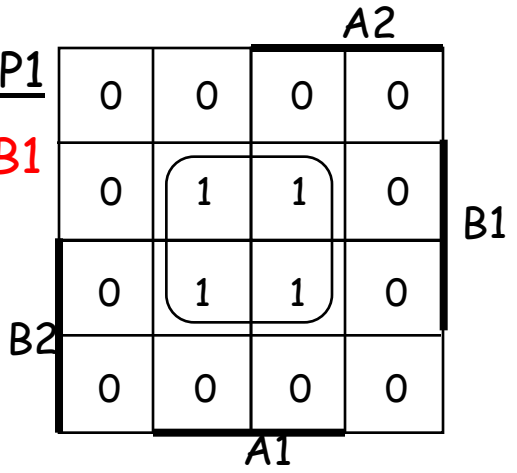
K-map for P4

P4 = A2B2B1'
+ A2A1'B2



K-map for P2

P2 = A2'A1B2
+ A1B2B1'
+ A2B2'B1
+ A2A1'B1



K-map for P1

P1 = A1B1