AGENDA

- Subtractors
- Multipliers
- Comparators

Subtractors

- How can we implement subtraction?
  → Subtraction is addition of complement
- How do we determine 2's complement?
  → 1's complement (flip bits) and add 1
- How can we flip bits?
  → XOR Gate
- How can we add 1?
  → input carry

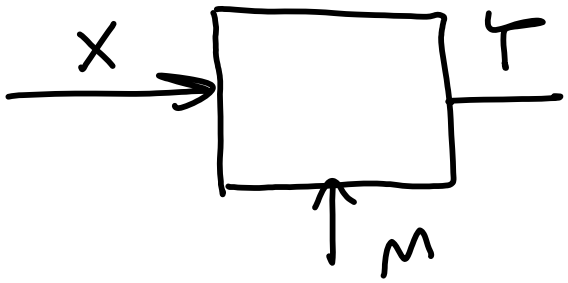
X	$\overline{X}$
0	1
1	0

$\equiv$

control

M	X	Y
0	0	0
0	1	1
1	0	1
1	1	0

$Y = M \oplus X$

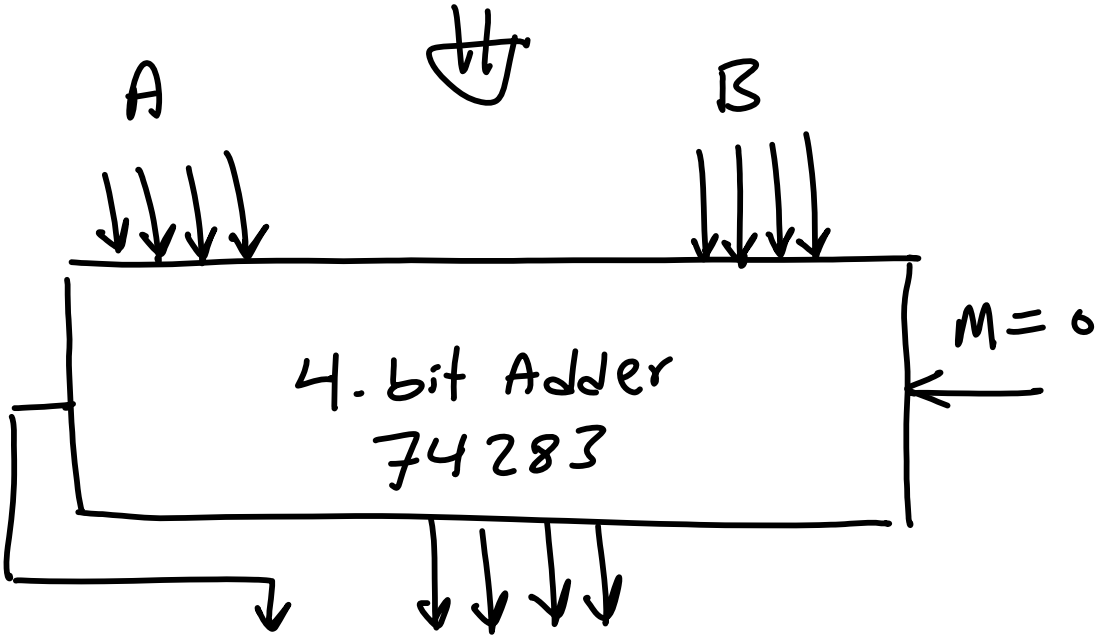


M is a control signal not data

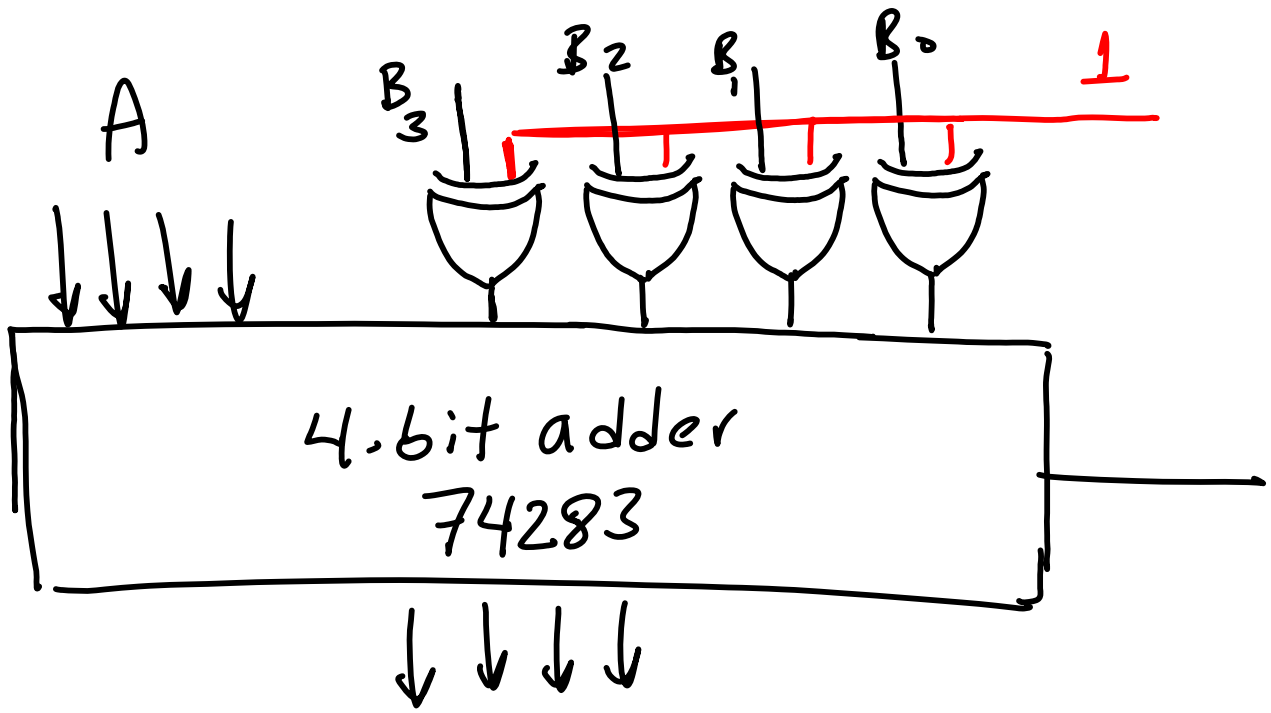
$M = 0 \Rightarrow$ addition

$M = 1 \Rightarrow$ Subtraction

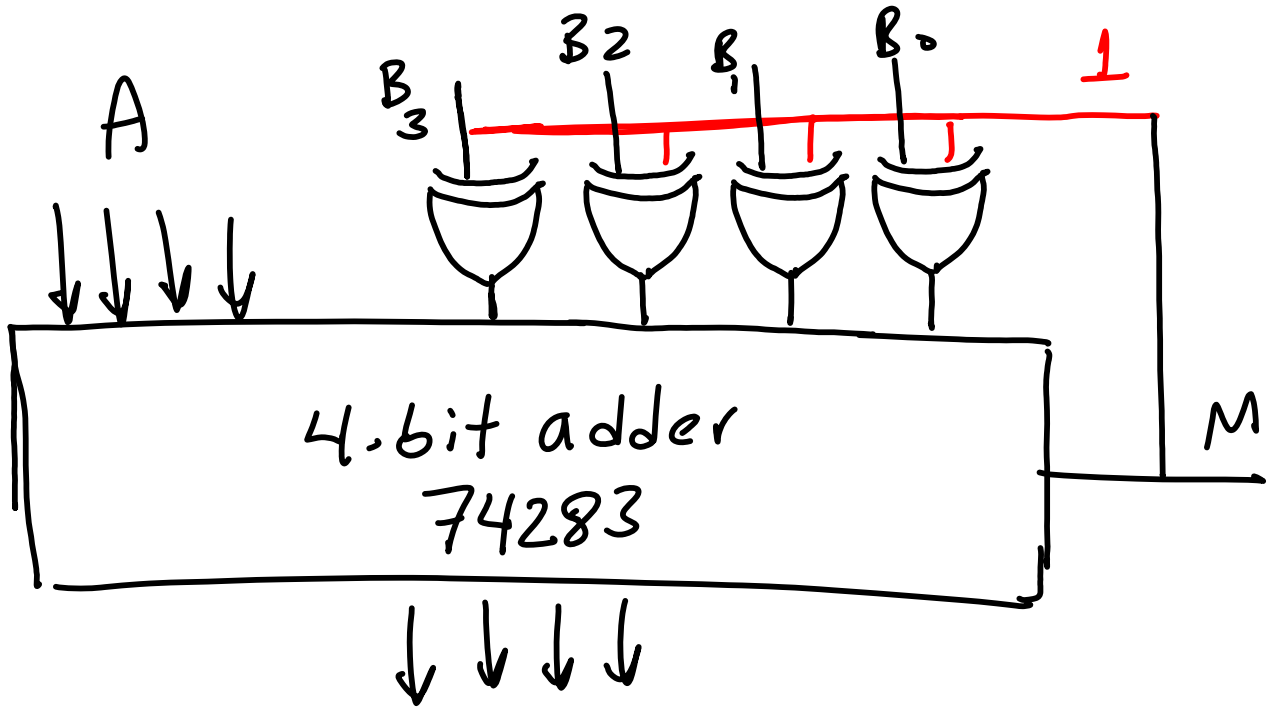
ADDER



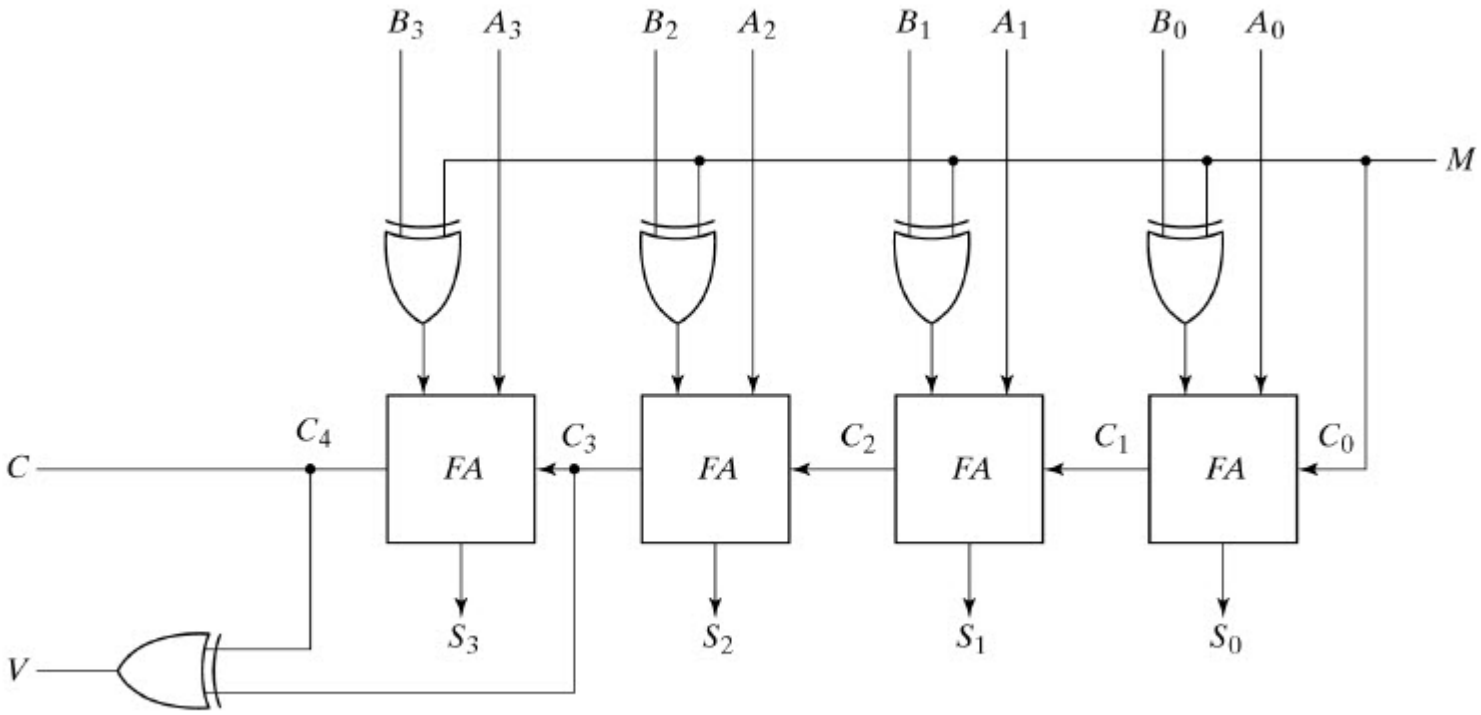
Subtractor



ADDer / Subtractor



Note it can be implemented using FA

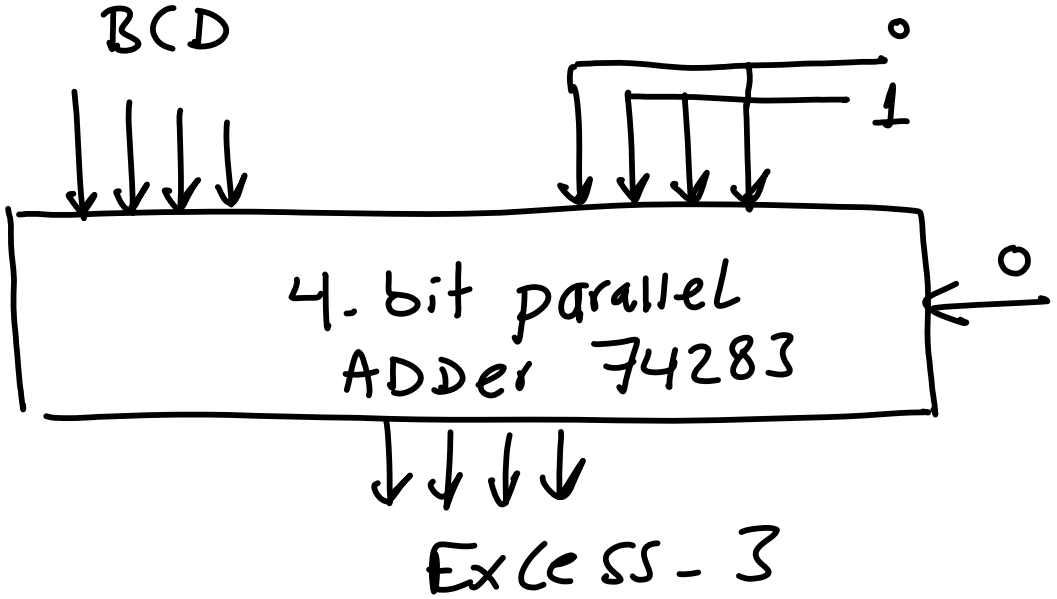


4-Bit Adder Subtractor

Ex

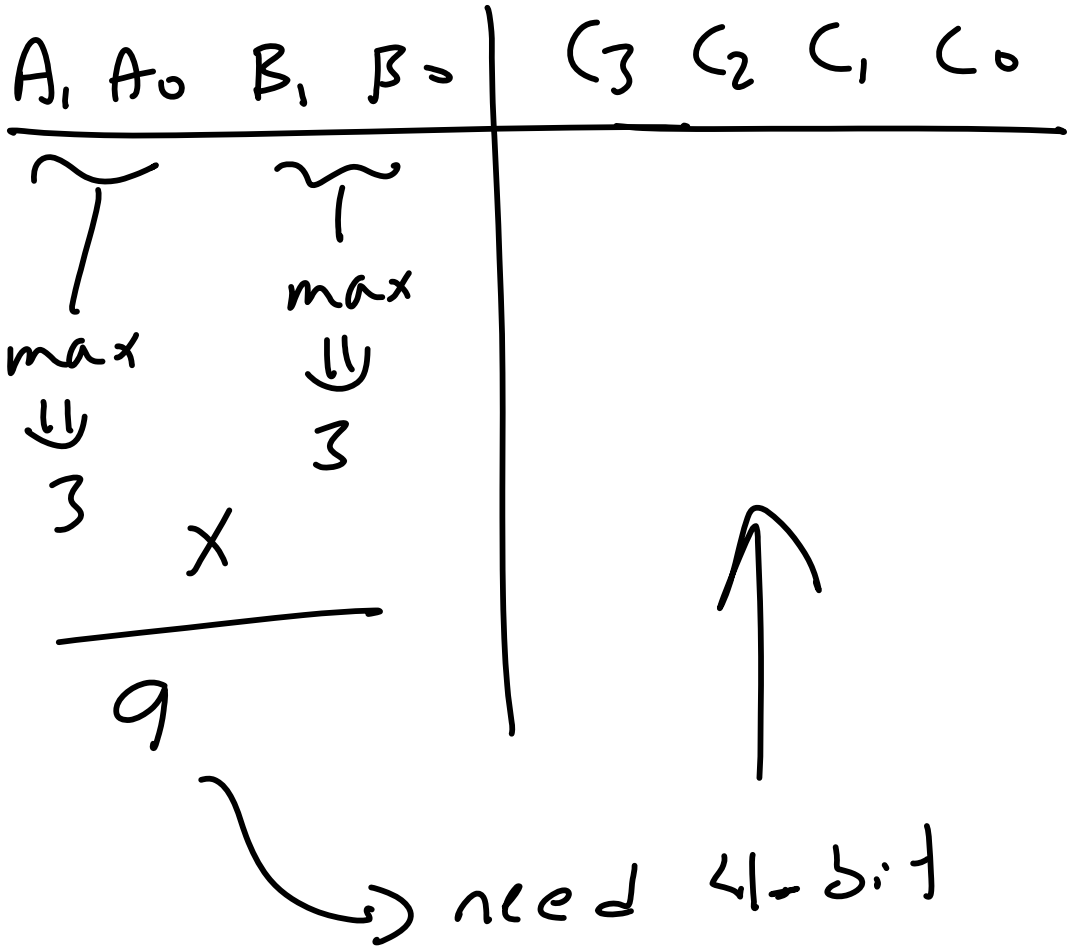
Design a logic circuit that convert from a BCD number to an excess-3 No.

Sol.



Ex

Design a circuit that produces the product of two 2-bit number



Note

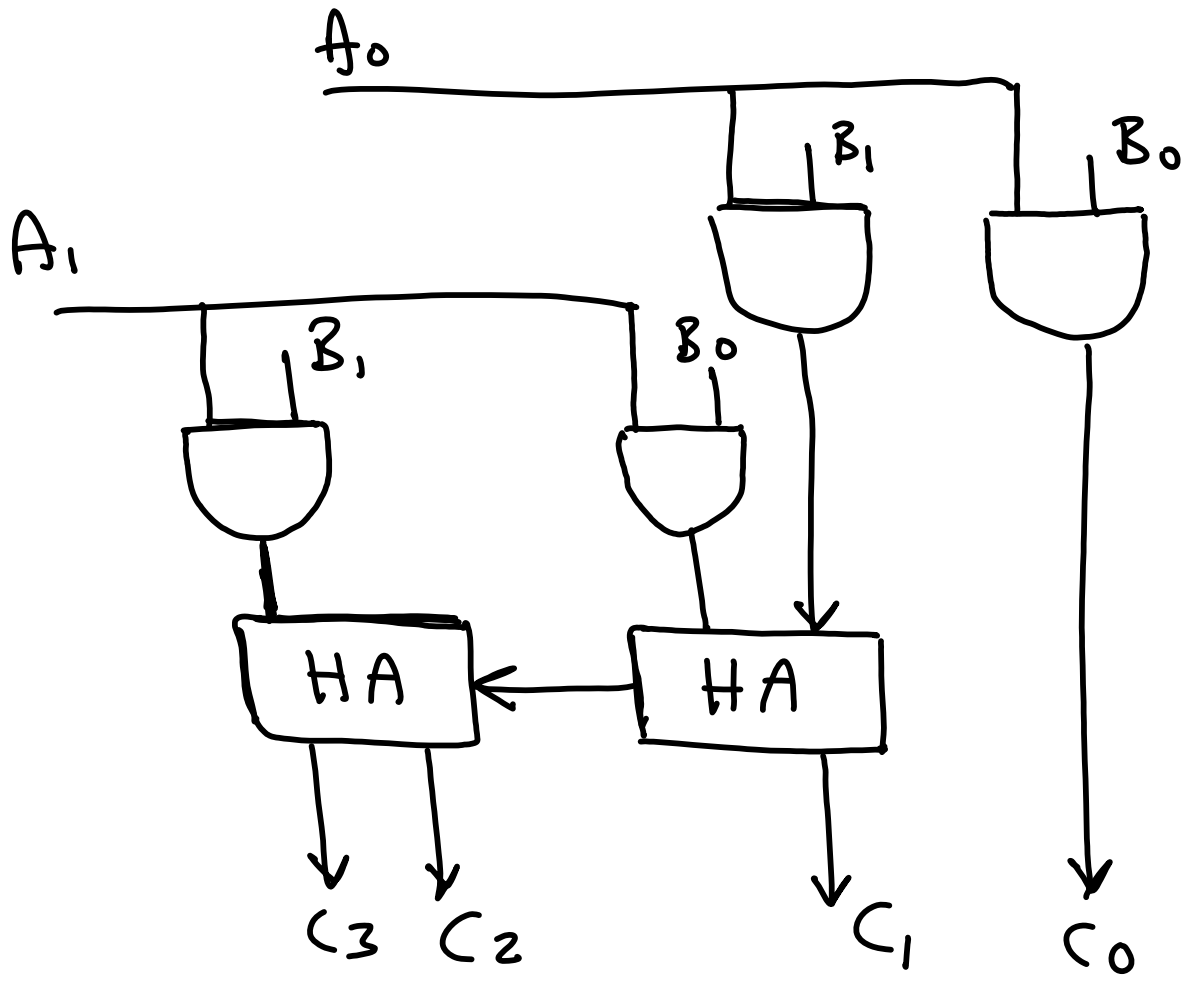
A_1 A_0

B_1 B_0

A_1B_0 A_0B_0

+

A_1B_1 A_0B_1



4-bit by 3-bit multiplier

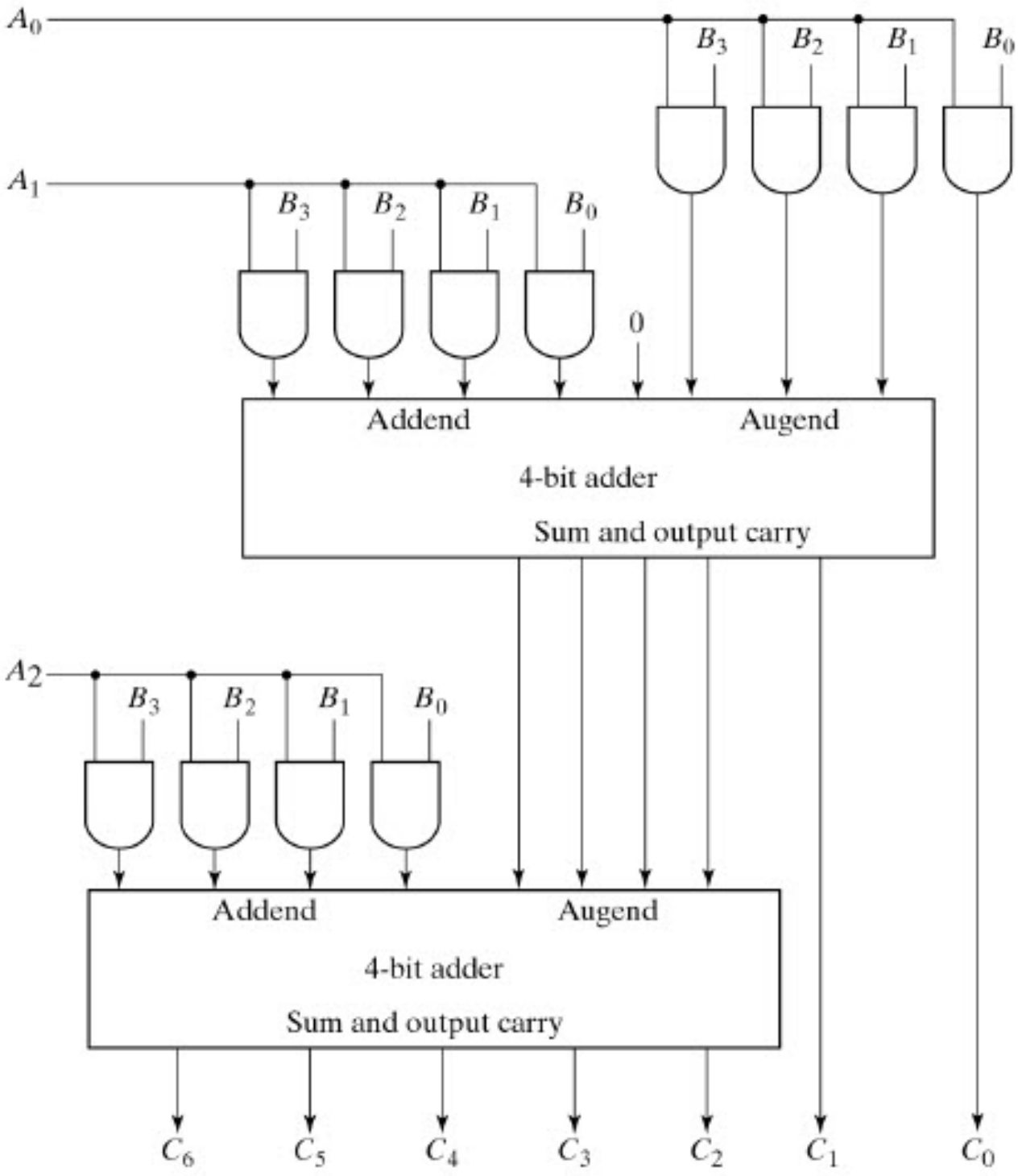
$B_3\ B_2\ B_1\ B_0$

$A_2\ A_1\ A_0$

$A_0B_3\quad A_0B_2\quad A_0B_1\quad A_0B_0$

$A_1B_3\quad A_1B_2\quad A_1B_1\quad A_1B_0$

$A_2B_3\quad A_2B_2\quad A_2B_1\quad A_2B_0$



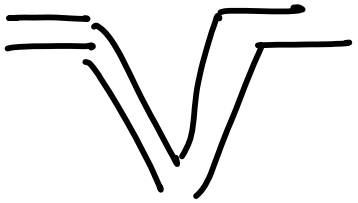
4-Bit by 3-Bit Binary Multiplier

Report

Design a 16-bit binary parallel adder using 4-bit binary adder?

Hint use 4 IC of 74283

Magnitude Comparator



Need to compare two numbers A, and B

A>B?

A=B?

A<B?

if A & B are two-bit numbers

A_1	A_0	B_1	B_0	A>B	A<B	A=B
0	0	0	0	0	0	1
0	0	0	1	0	1	0
0	0	1	0	0	1	0
0	0	1	1	0	1	0

⇓ ⇓

Continue

if A and B are n-bit numbers, then we will need 2^{2n} entries, and it would be impractical for design

We can say that $A = B$ if every digit of A equal to B

↳ $A_3 A_2 A_1 A_0 = B_3 B_2 B_1 B_0$

if and only if

$A_3 = B_3$ and $A_2 = B_2$ and $A_1 = B_1$

and $A_0 = B_0$

for one bit A_i, B_i

A_i	B_i	X_i
0	0	1
0	1	0
1	0	0
1	1	1

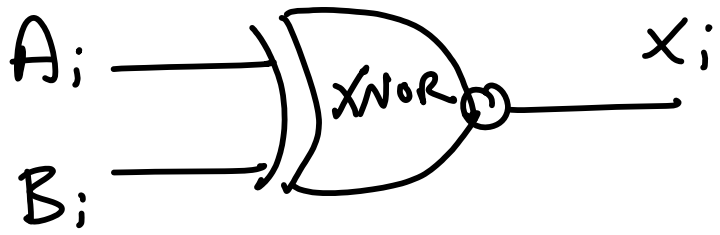
$$X_i = \overline{A_i}\,\overline{B_i} + A_i B_i$$

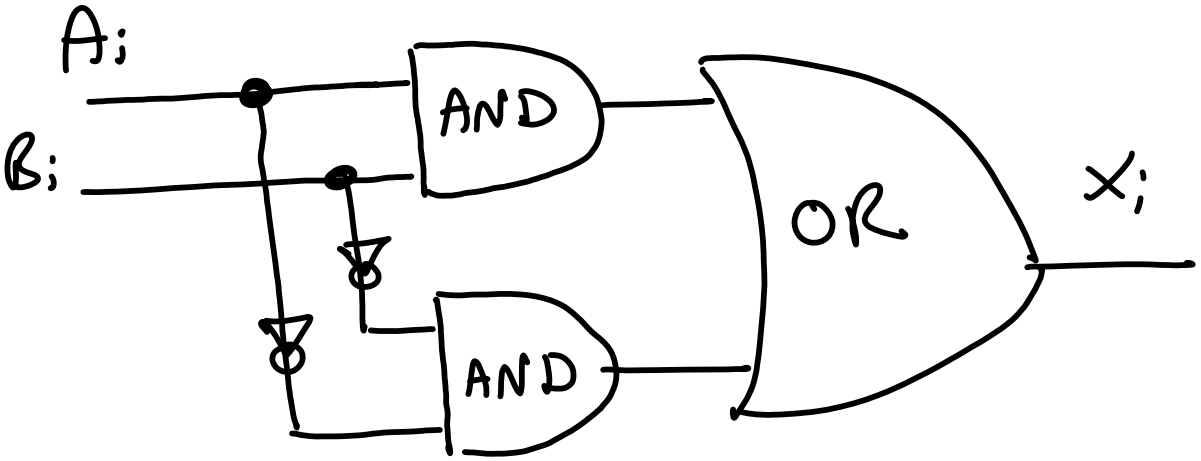
or

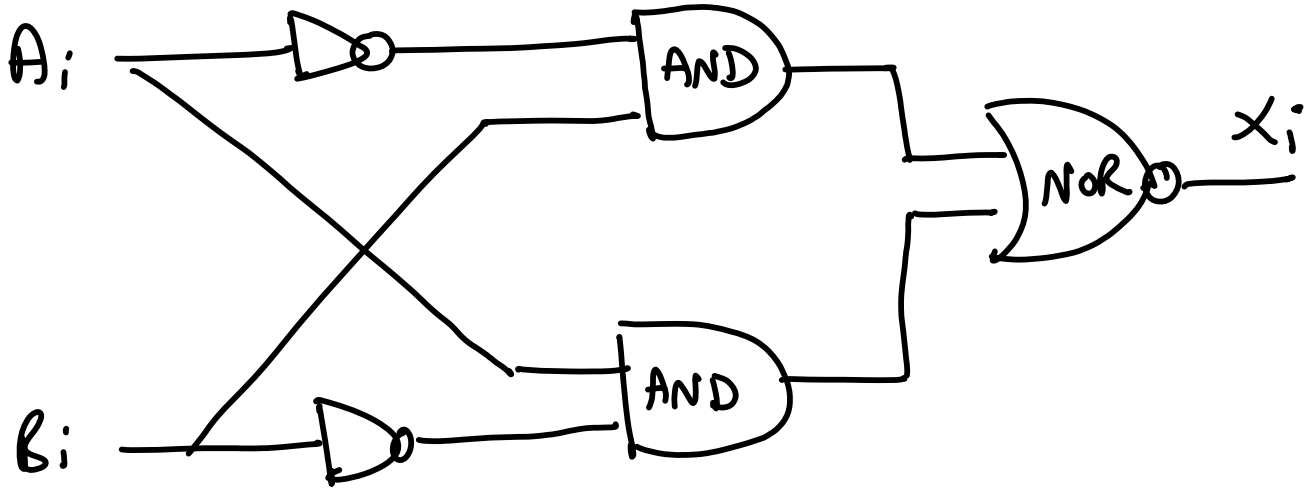
$$\overline{X_i} = \overline{A_i B_i + A_i \overline{B_i}}$$

$$X_i = \overline{A_i}\,B_i + A_i\,\overline{B_i}$$

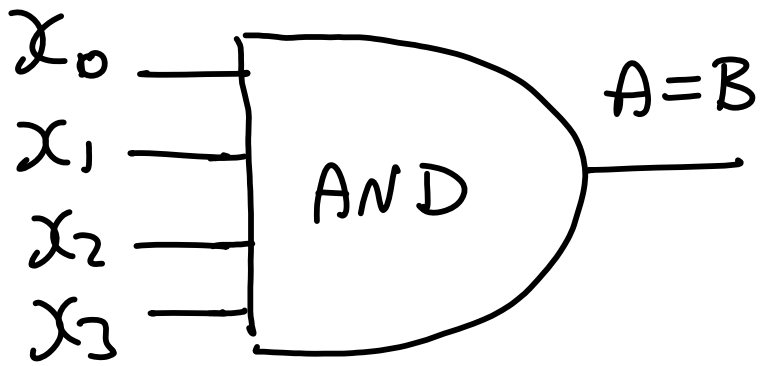
↳ XNOR


A;
B;
XNOR
X;


A;
B;
AND
AND
OR
X;


A;
B;
AND
AND
NOR
X;

So $A = B$ if



⇓

$A = B$ iff $x_3 x_2 x_1 x_0 = 1$

what about $A > B$

$A_3\ A_2\ A_1\ A_0$ $B_3 B_2 B_1 B_0$

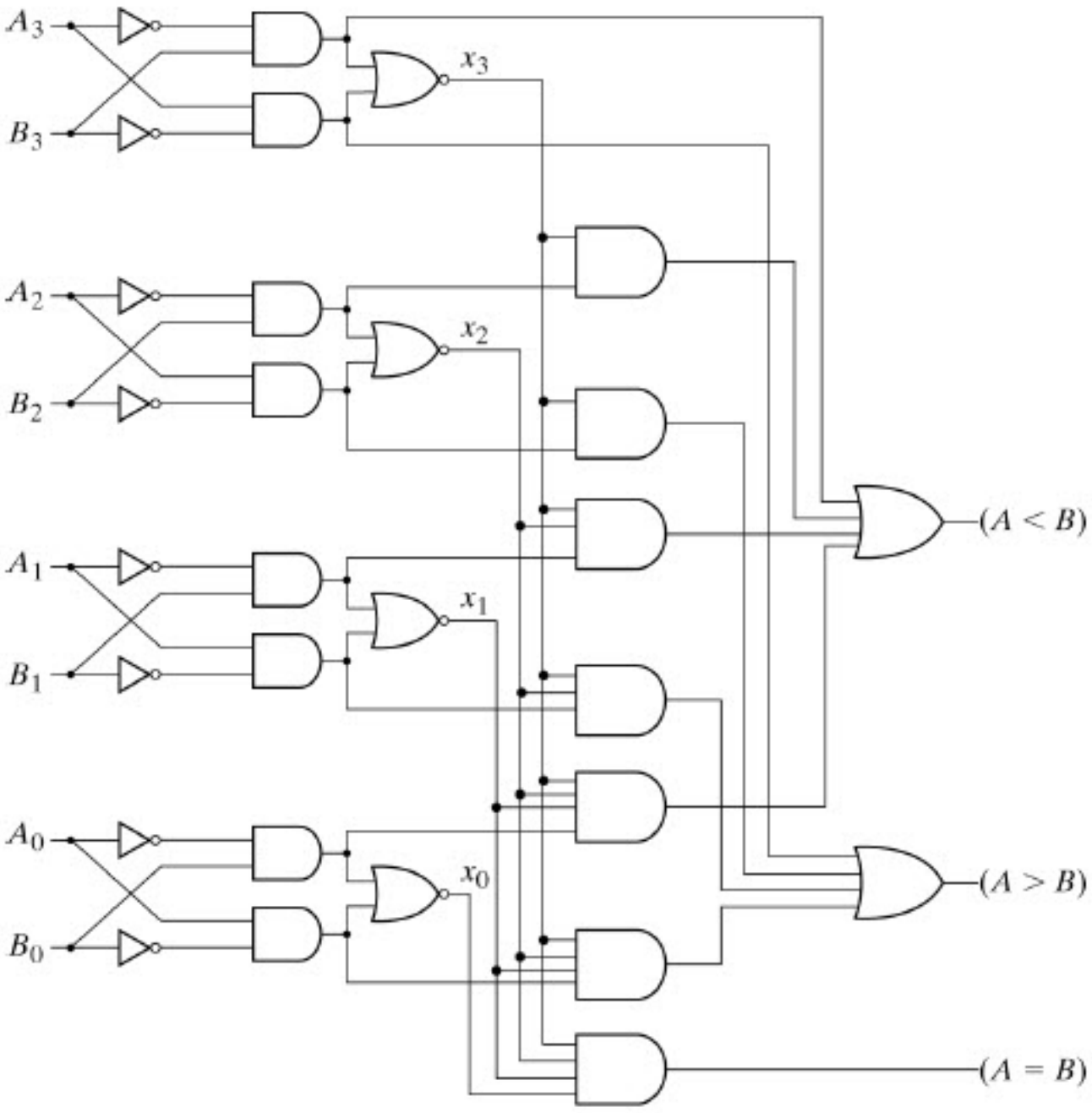
$A > B$ iff

$A_3 > B_3$ or $A_3 = B_3$ and $A_2 > B_2$

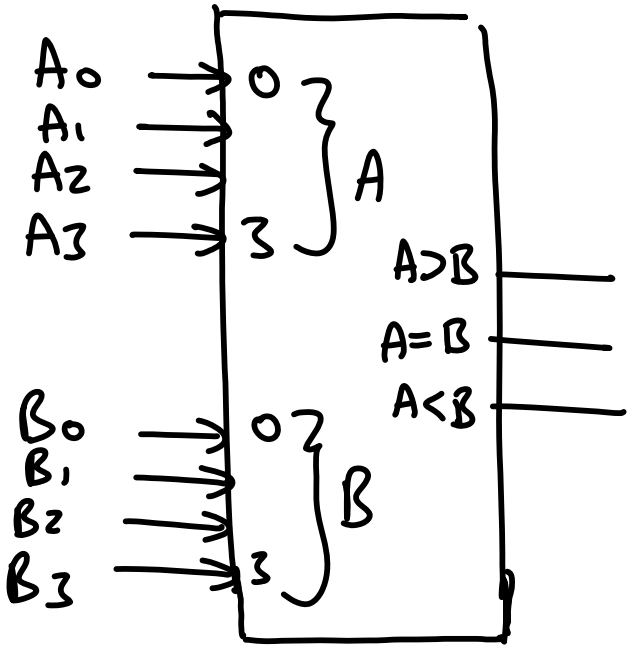
or $A_3 = B_3$ and $A_2 = B_2$ and $A_1 > B_1$

or $A_3 = B_3$ and $A_2 = B_2$ and $A_1 = B_1$ and $A_0 > B_0$

$A > B$ iff $A_3\overline{B_3} + x_3 A_2\overline{B_2} + x_3x_2 A_1\overline{B_1} + x_3x_2x_1 A_0\overline{B_0}$



4-Bit Magnitude Comparator



infoNote

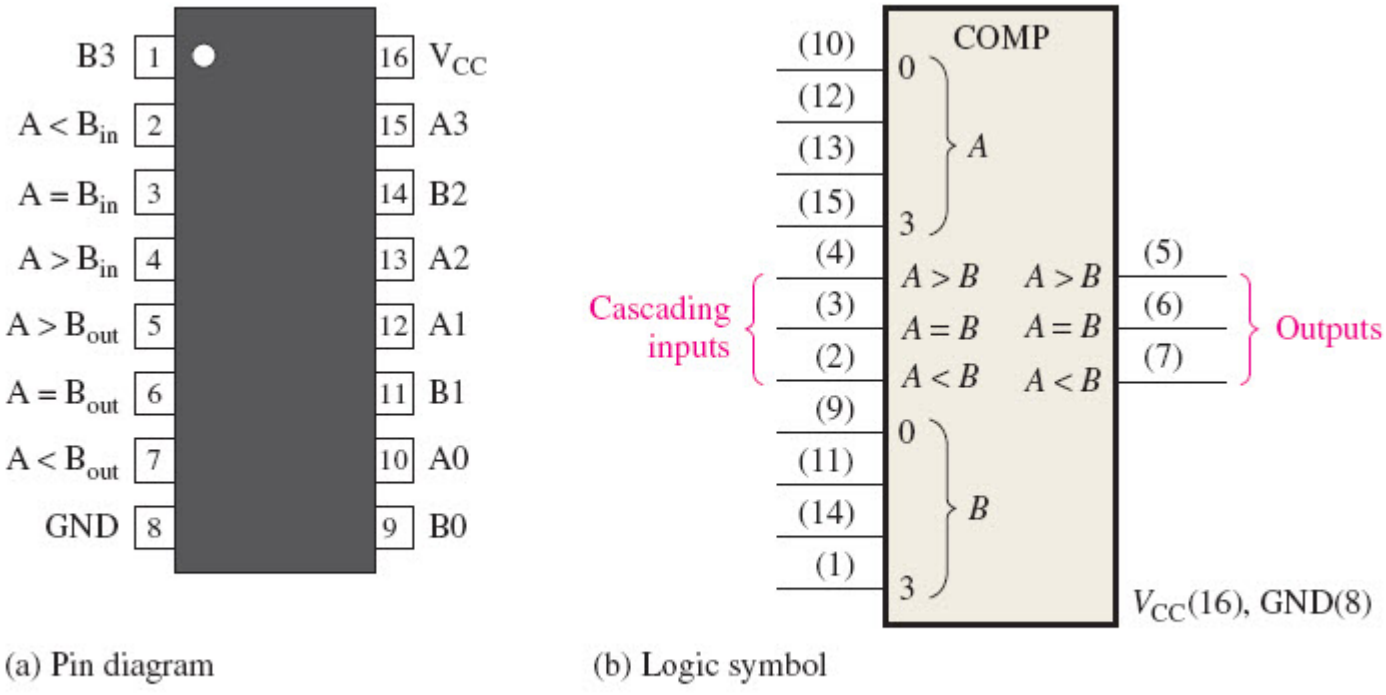
in a computer, the cache is a very fast intermediate memory between the central processing unit (CPU) and the slower main memory. the CPU requests data by sending out its address (unique location) in memory. part of this address is called a tag.

the tag address comparator compares the tag from the CPU with the tag from the

cache directory - if the two agree, the addressed data is already in the cache and is retrieved very quickly -

if the tags disagree, the data must be retrieved from the main memory at a much slower rate

74HC85 / 74LS85



(a) Pin diagram (b) Logic symbol

The 74HC85/74LS85 4-bit magnitude comparator.

Note

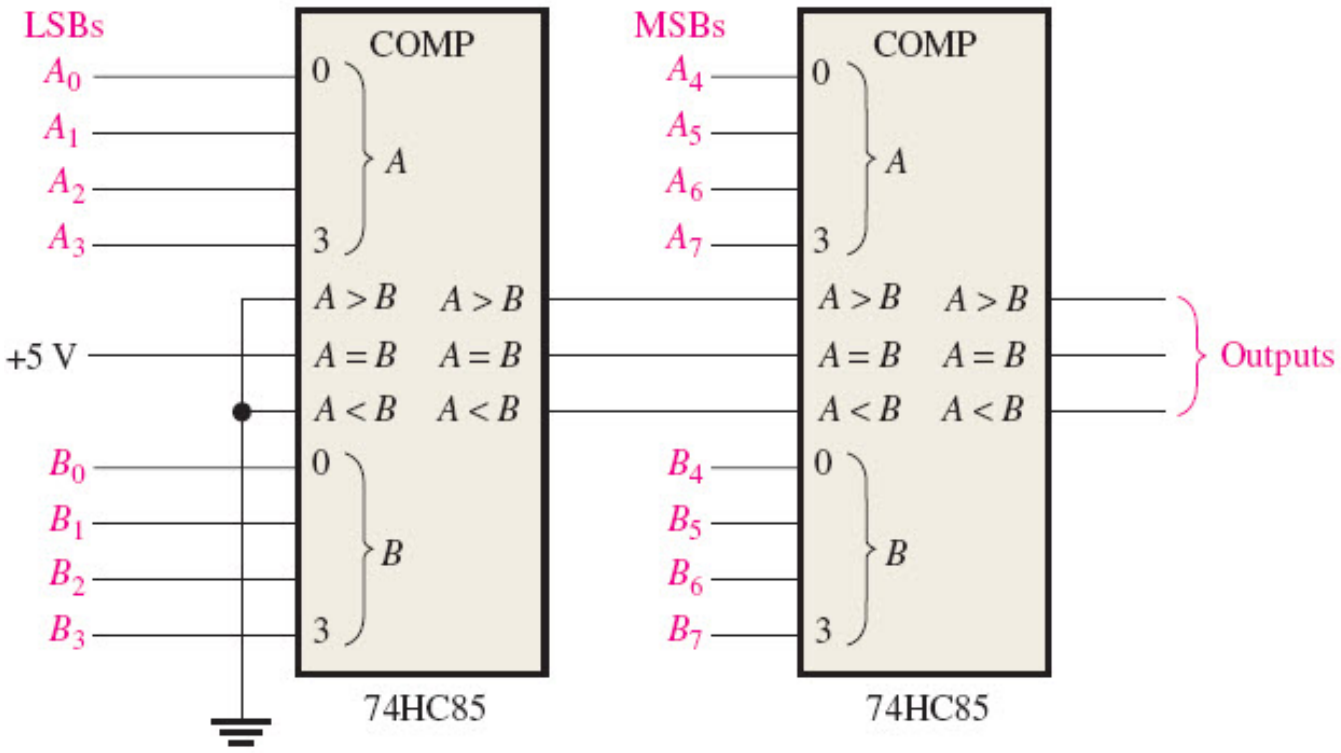
- it has three cascading inputs: $A<B$, $A=B$, $A>B$.
- these inputs allow several comparators to be cascaded for comparison of any number of bits greater than four

↓

to expand the comparator, the $A<B$, $A=B$, and $A>B$ outputs of the lower-order comparator are connected to the corresponding cascading inputs of the next higher-order comparator

Prob use 74HC85 comparators to compare the magnitudes of two 8-bit numbers — show the comparators with proper interconnections

- use MultiSIM & proteus to simulate
- Connect on testboard & 16 switches and 3 LEDs



magnitude comparator using two 74HC85s.

Problem

Expand the circuit to a 16-bit comparator

Ex ⇒

Design a logic circuit that can implement the shown truth table

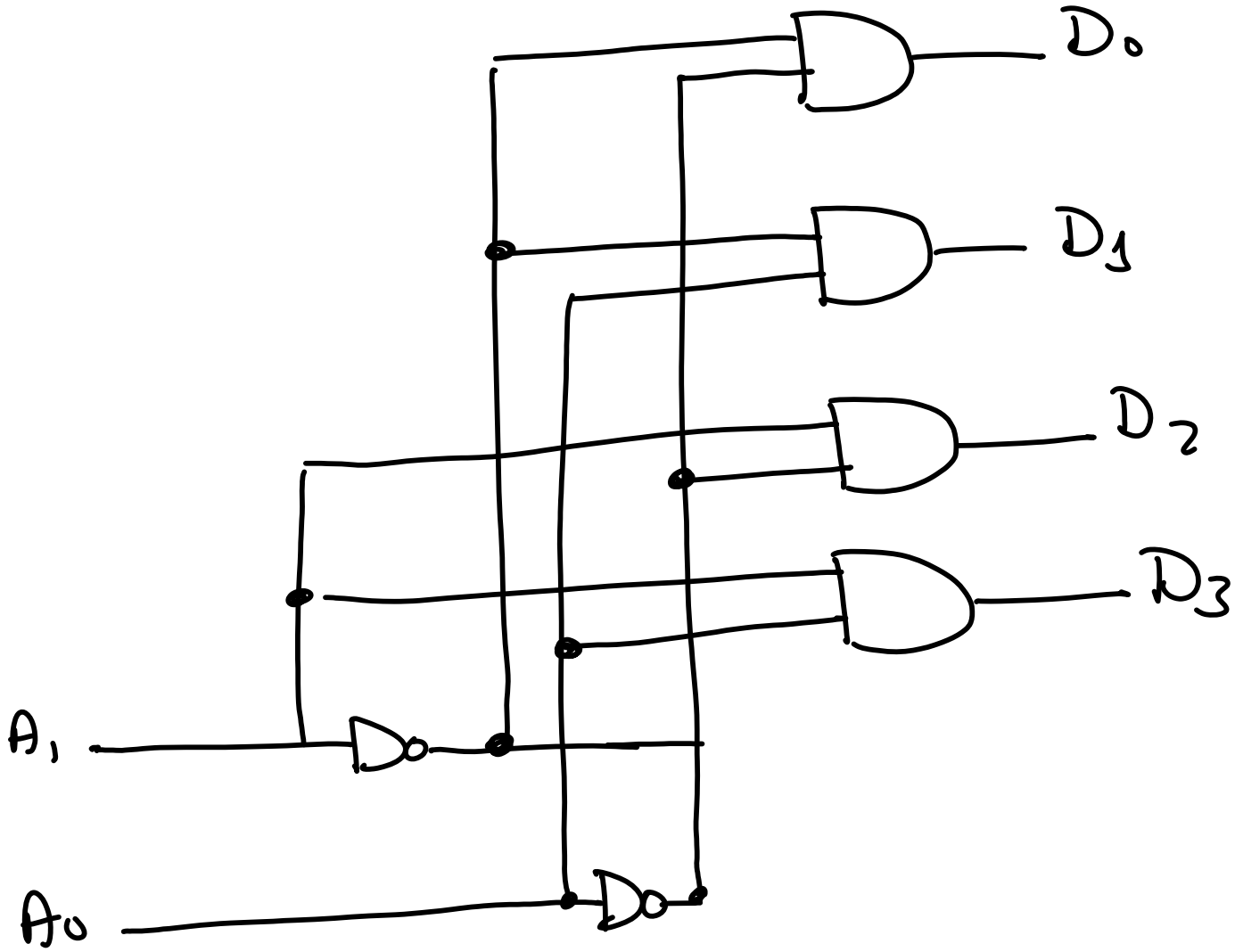
A_1	A_0	D_0	D_1	D_2	D_3
0	0	1	0	0	0
0	1	0	1	0	0
1	0	0	0	1	0
1	1	0	0	0	1

$D_0 = \overline{A_1}\,\overline{A_0}$

$D_1 = \overline{A_1}\,A_0$

$D_2 = A_1\,\overline{A_0}$

$D_3 = A_1 A_0$



Decoder 2x4