Binary Multiplier

multiplication of binary numbers is performed in the same way as multiplication of decimal numbers.

1- multiplication of two 2-bit numbers :

B_1 B_0 → multiplicand (B)

A_1 A_0 → multiplier (A)

A_0B_1 A_0B_0 → partial product

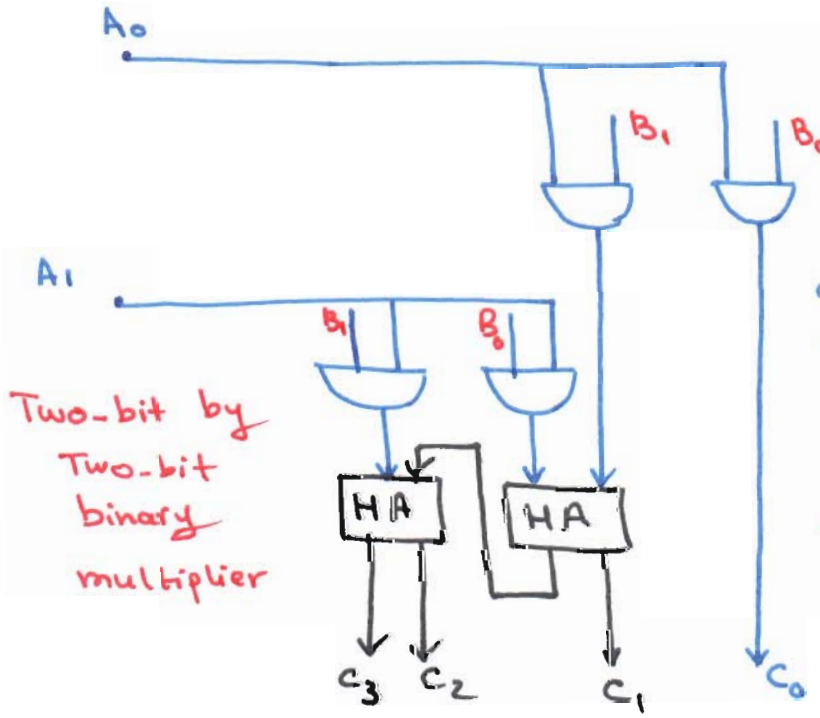
A_1B_1 A_1B_0

C_3 C_2 C_1 C_0 → final product (the sum of the partial products)

Sum of products →

Combinational Circuit.

* The multiplication of two bits such as A_0 and B_0 produces a 1 if both bits are 1; otherwise, it produces a 0, This is identical to an AND operation.
* The two partial products are added with two half-adder (HA) circuits. (if there are more than two bits, we must use full-Adder (FA).



Two-bit by Two-bit binary multiplier

2- Combinational Circuit of binary multiplier with more bits.

* For J multiplier bits and K multiplicand bits, We need :

JXK. and gates.

(J-1)K-bits adders

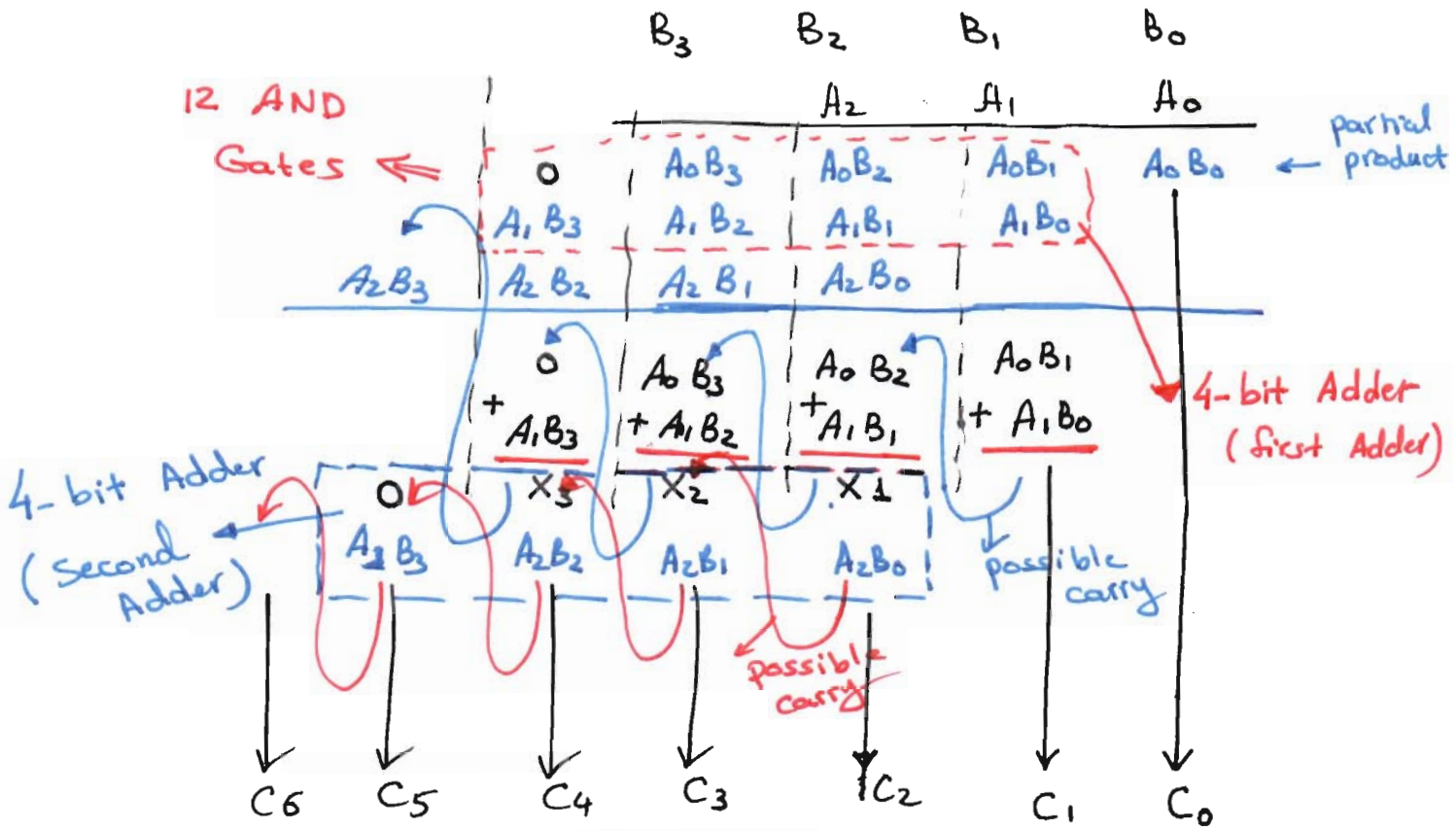
to produce a product of j+K bits.

Example: Create a logic circuit for

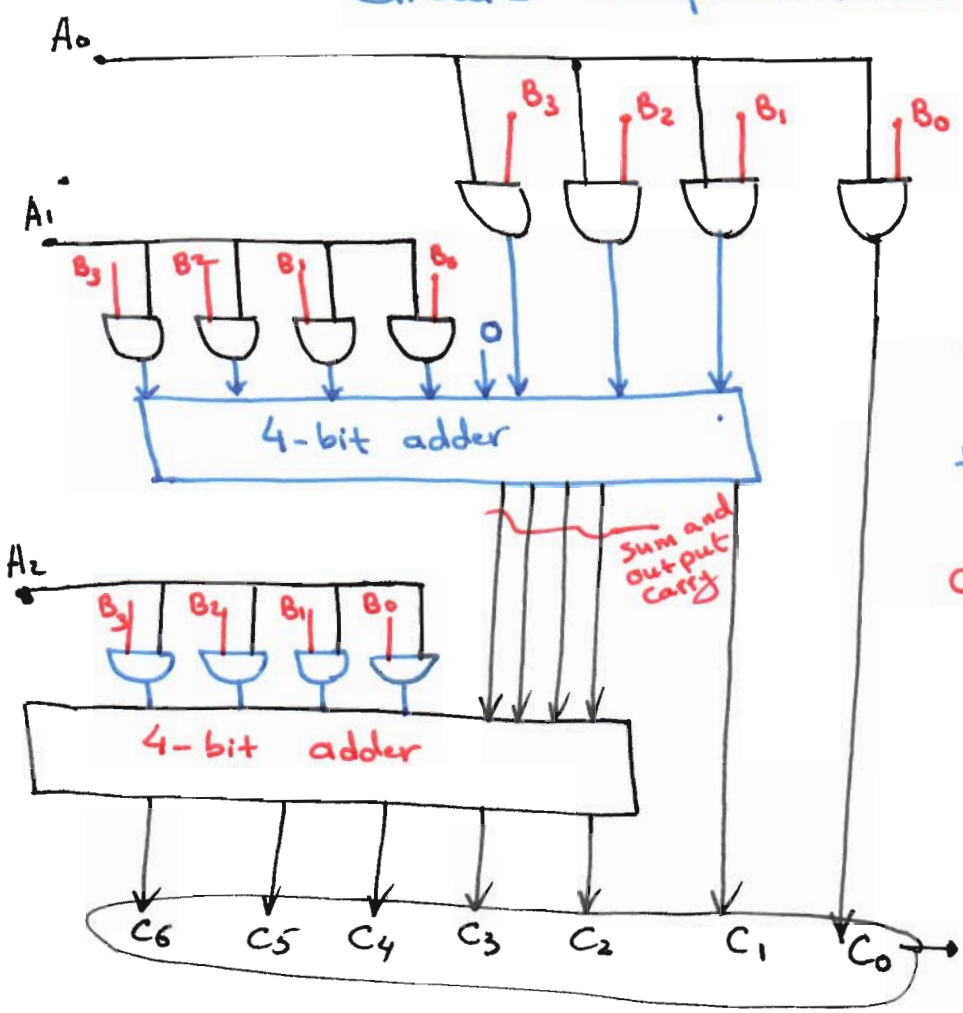
$B_3\ B_2\ B_1\ B_0$ ×

$A_2\ A_1\ A_0$

$C_6\ C_5\ C_4\ C_3\ C_2\ C_1\ C_0$ → Answer.



Circuit Implementation.



* K = 4

J = 3

So:

we need

12 AND Gates

and

2 four-bit Adders.

to produce a product of seven bits:

$C_6\ C_5\ C_4\ C_3\ C_2\ C_1\ C_0$

Decoders

information is represented in digital systems by binary codes. A binary code of n bits is capable of representing up to 2^n distinct elements of coded information.

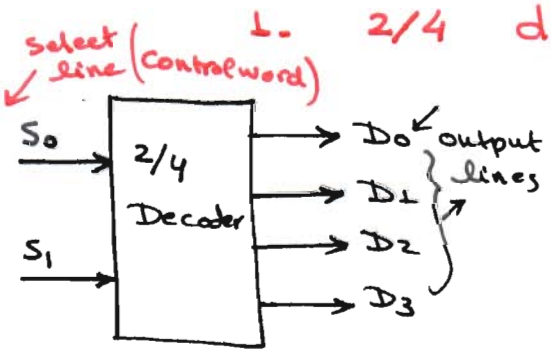
A **decoder** is a combinational circuit that covers binary information from n input lines to a maximum of 2^n unique output lines.

The decoders are called n-to-m line decoders, where $m \leq 2^n$ (for example BCD-to-seven-segment decoder, or 3-to-8 decoder). The purpose of the decoders is to generate the 2^n (or fewer) minterms of n input variables.

* Decoder is a circuit that allows us to activate an output line by specifying a control word, it is either active-high or active-Low.

- Active-high sets a particular output to logic 1. while remaining other outputs to logic 0.
- Active-Low sets a particular output to logic 0, other outputs to logic 1.

1. 2/4 decoder (Active-high).



S_0, S_1 → select lines

D_0, D_1, D_2, D_3 → output lines.

S_1	S_0	D_3	D_2	D_1	D_0
0	0	0	0	0	1
0	1	0	0	1	0
1	0	0	1	0	0
1	1	1	0	0	0

Truth table

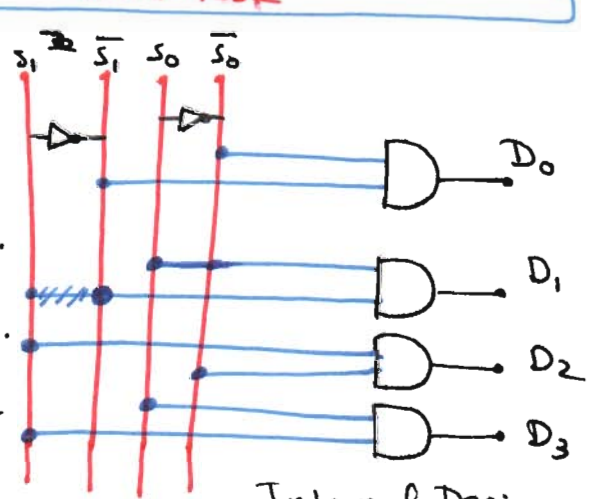
1) $S_1 = 0$, $S_0 = 0 \Rightarrow D_0 = 1$

$D_0 = \overline{S_1} \cdot \overline{S_0}$; $D_1 = D_2 = D_3 = 0$.

2) $S_1 = 1$, $S_0 = 1 \Rightarrow D_1 = 1$

$D_0 = D_2 = D_3 = 0$; $D_1 = S_1 S_0$

3) $D_2 = S_1 \cdot \overline{S_0}$ 4) $D_3 = S_1 \cdot S_0$

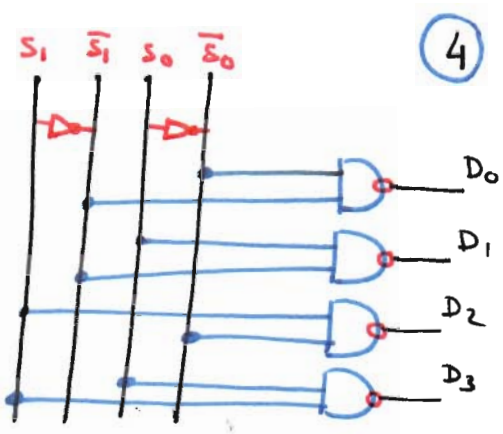


Internal Design

* from truth table:
  - two inputs are decoded to 4 outputs.
  - each output represents one of the minterms of the three input variables.
  - for each possible combination (input), there are 3 outputs that are equal to 0 and only one that is equal to 1.

- **Active-Low Decoders:**

S_1	S_0	D_3	D_2	D_1	D_0
0	0	1	1	1	0
0	1	1	1	0	1
1	0	1	0	1	1
1	1	0	1	1	1



AND becomes NAND

$D_0 = \overline{\overline{S_1} \cdot \overline{S_0}}$ → Complement

$= S_1 + S_0$

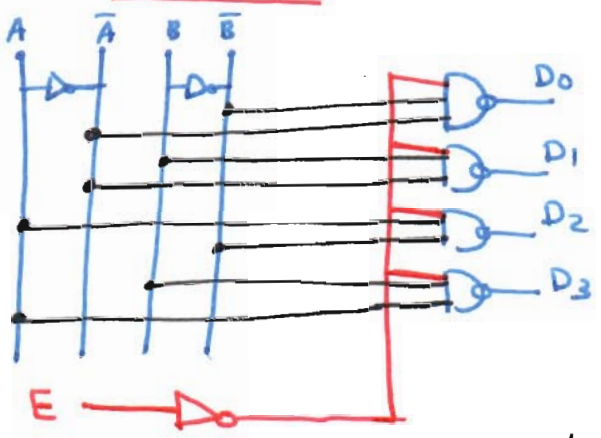
$D_1 = \overline{\overline{S_1} \cdot S_0} = S_1 + \overline{S_0}$; $D_2 = \overline{S_1} + S_0$; $D_3 = \overline{S_1} + \overline{S_0}$.

- **Decoders with enable inputs:**

- Some decoders include one or more enable inputs to control the circuit operation.

a two-to-four line decoder with an enable input constructed with NAND gate is the following:

E	Inputs A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	1



The circuit operates with complemented outputs and a complemented enable input (Active-low enable)

$E = 0$ → The decoder is enabled

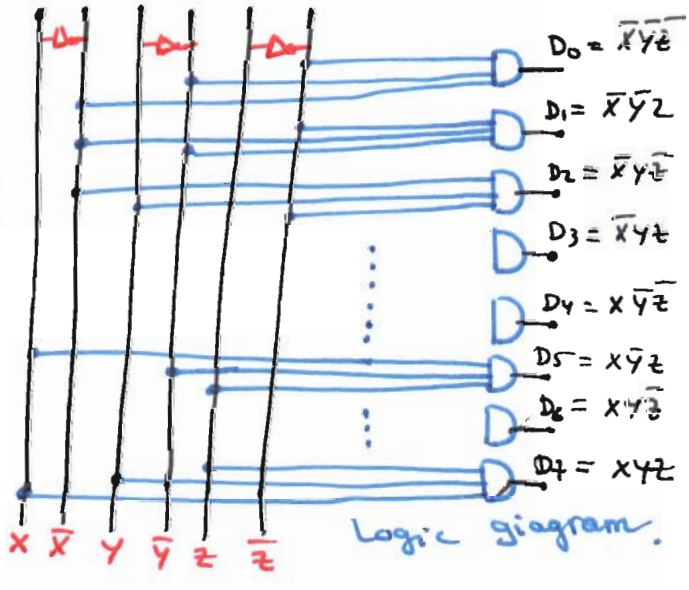
$E = 1$ → The decoder is disabled [Active-low enable].

* The enable input may be activated with 0 or with a 1 signal.

2. three-to-eight-line decoder circuit

Inputs X	Y	Z	outputs D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7
0	0	0	1	0	0	0	0	0	0	0
0	0	1	0	1	0	0	0	0	0	0
0	1	0	0	0	1	0	0	0	0	0
0	1	1	0	0	0	1	0	0	0	0
1	0	0	0	0	0	0	1	0	0	0
1	0	1	0	0	0	0	0	1	0	0
1	1	0	0	0	0	0	0	0	1	0
1	1	1	0	0	0	0	0	0	0	1

Truth table.

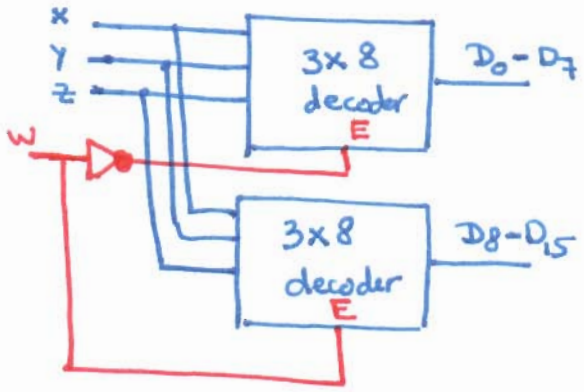


Logic diagram.

3. Larger decoder Circuit.

decoders with enable inputs can be connected together to form a larger decoder ciruit.

To design a 4-to-16 line decoder, using two 3-to-8 decoders with enable inputs, we do the following connection:



when

* $W = 0 \Rightarrow$ the top decoder is enable and the other is disable.

The bottom decoder outputs are all 0's, and the top eight outputs generate minterms 0000 to 0111.

* when $W = 1 \Rightarrow$ the enable condition is reversed, the bottom decoder outputs generate minterms 1000 to 1111.

So, to Connect more standard components, we can use the enable input.

4. Combinational Logic implementation:

Since, A decoder provides the 2^n minterms of n input, and any Boolean function can be expressed in sum-of-minterms form, A decoder that generates the minterms of the functions, together with an external OR gate that forms their logical sum, provides a hardware implementation of the function.

In this way, any combinational circuit with n inputs and m outputs can be implemented with an n-to-2^n line decoder and m OR gates.

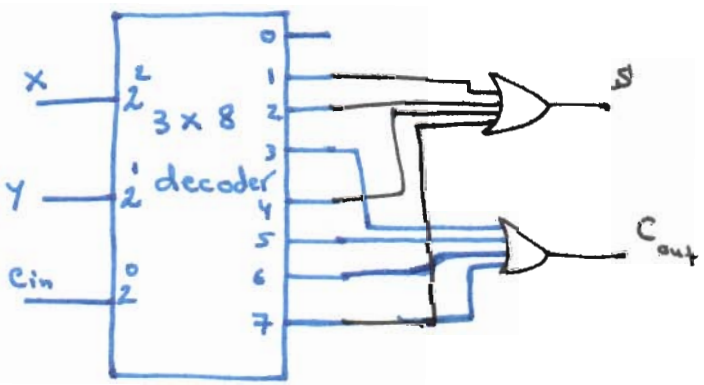
The procedure for implementing a combinational circuit by means of decoders and OR gates requires that the Boolean function for the circuit be expressed as a sum of minterms.

example: implementation a full-adder Circuit using decoders:

From the truth table of the full adder, we can obtain the functions for the sum and carry in sum-of-minterms form:

$$S(x, y, C_{in}) = \sum(1,2,4,7)$$

$$C_{out}(x, y, C_{in}) = \sum(3,5,6,7)$$



We have 3 inputs: $\Rightarrow$
So, we need a three-to-eight-line decoder.