EE 2720

Handout #21

$\boxed{\text{EE 2720}}$

• Propagation delay:

Propagation delay through a logic circuit is the time it takes for a change at the inputs to produce a change at the outputs.

Note: Propagation delays when outputs change from LOW to HIGH may differ from when they change from HIGH to LOW.

Example 1: Consider the logic circuit of fig. 1 below. If the propagation delay through a inverter is $t_1 = 1$ nsec, the propagation delay through a 2-input AND gate is $t_2 = 2$ nsec, the propagation delay through a 3-input AND gate is $t_3 = 3$ nsec, the propagation delay through a 2-input OR gate is $t_4 = 2$ nsec and the propagation delay through a 3-input OR gate is $t_5 = 3$ nsec, what is the worst case propagation delay through the circuit of fig. 1?

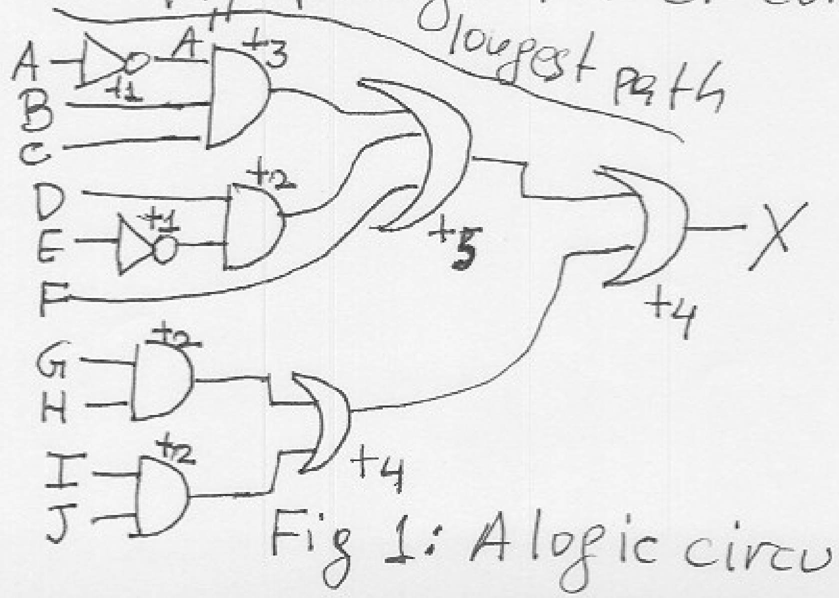

Fig 1: A logic circuit

Answer: The worst case propagation delay through the circuit of fig.1 is t where t is $t = t_1 + t_3 + t_5 + t_4$ = delay through longest path $= (1+3+3+2)$ nsec = 9 nsec.

Note: In the above, nsec means nanosecond $= 10^{-9}$ second.

• Timing diagrams:

Timing diagrams illustrate the logical behavior of the signals in a digital circuit as a function of time.

Example 2: Consider the 2-input AND gate shown in figure 2

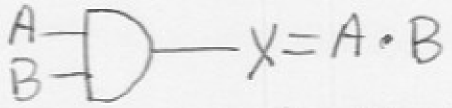

Fig. 2: A 2-input AND gate.

Suppose that the propagation delay though the above 2-input AND gate is $t = 2$ nsec. Given the timing diagrams of the inputs A and B provided in figures 3 and 4 on the next page, provide a timing diagram for the output X of the AND gate.

↳ Answer on next page →

Answer

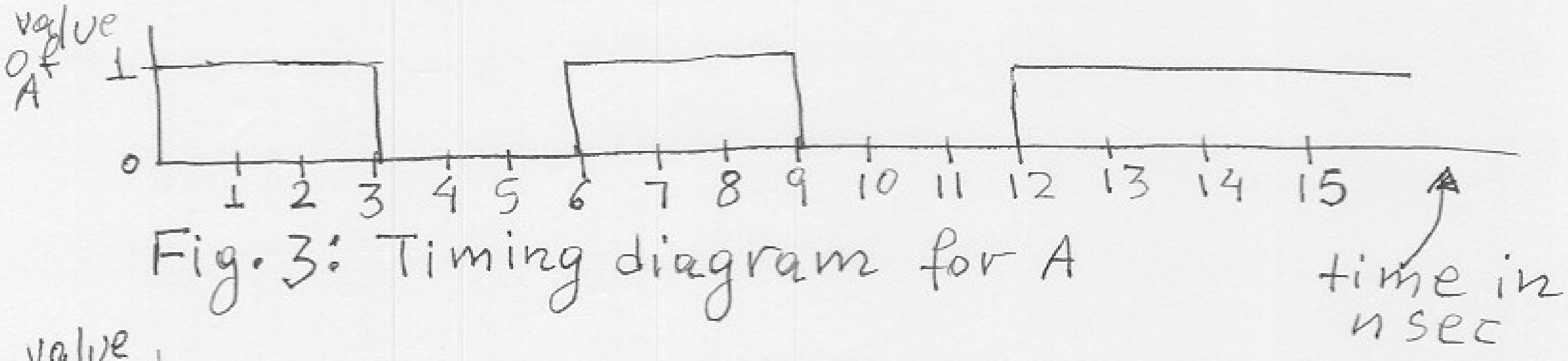

Fig. 3: Timing diagram for A

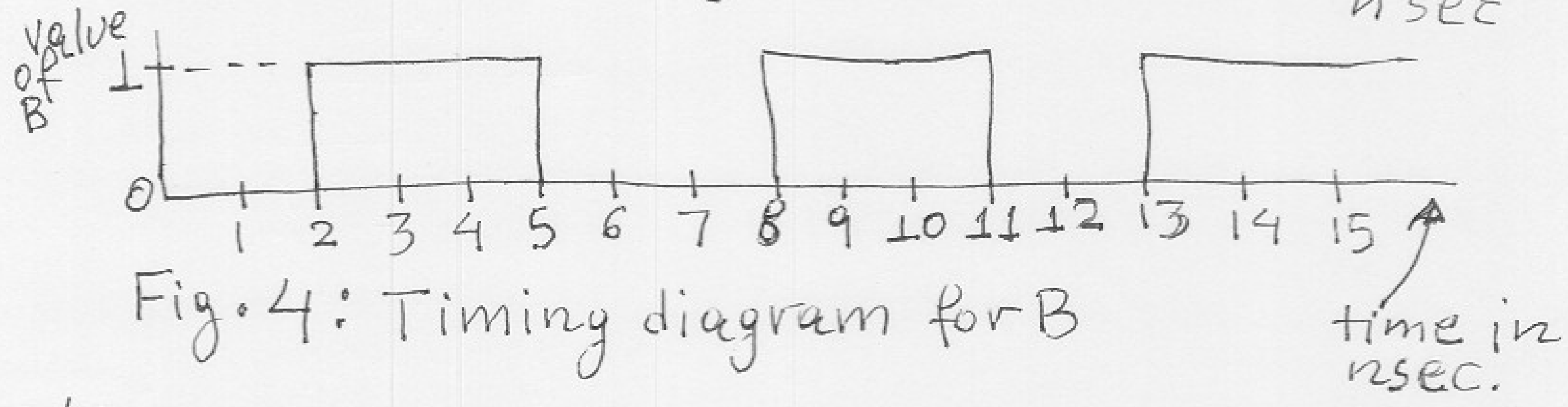

Fig. 4: Timing diagram for B

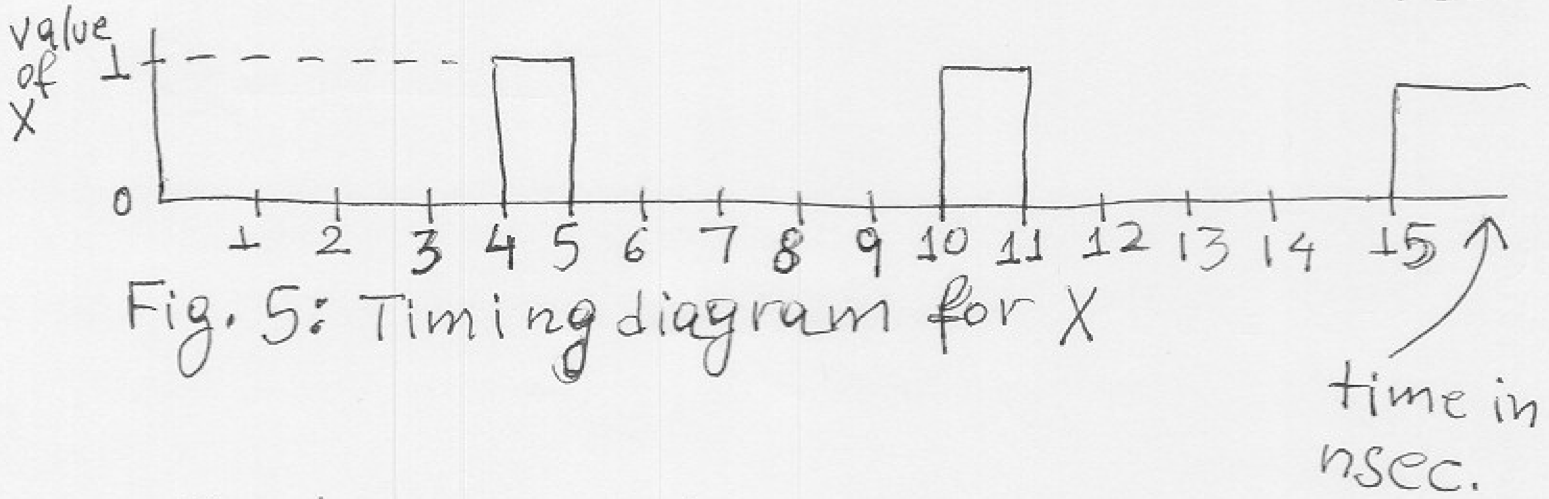

Fig. 5: Timing diagram for X

- Combinational Programmable Logic Devices (PLDs):

Historically, the first Programmable Logic Devices (PLDs) were the programmable logic arrays ([redacted] PLAs).

A PLA is a combinational,

two-level AND-OR device that can be programmed to realize any sum-of-products logic expression, subject to the size limitations of the device. Limitations are:

- The number of inputs (n).
- The number of outputs (m).
- The number of product terms (p).

We can describe such a device as "an $n \times m$ PLA with p product terms". In general, p is much less than the number of n-variable minterms which is 2^n. Thus, a PLA cannot perform arbitrary n-input, m-output logic functions. Its usefulness is limited to functions that can be expressed in sum-of-products form using p or fewer product terms. An $n \times m$ PLA with p product terms contains p $2n$-input AND gates and m p-input OR gates. All AND gates (the AND array) and all OR gates (the OR array) are programmable; (not fixed). Figure 6 on next page shows a 3×3 PLA with 6 product terms before programming it.

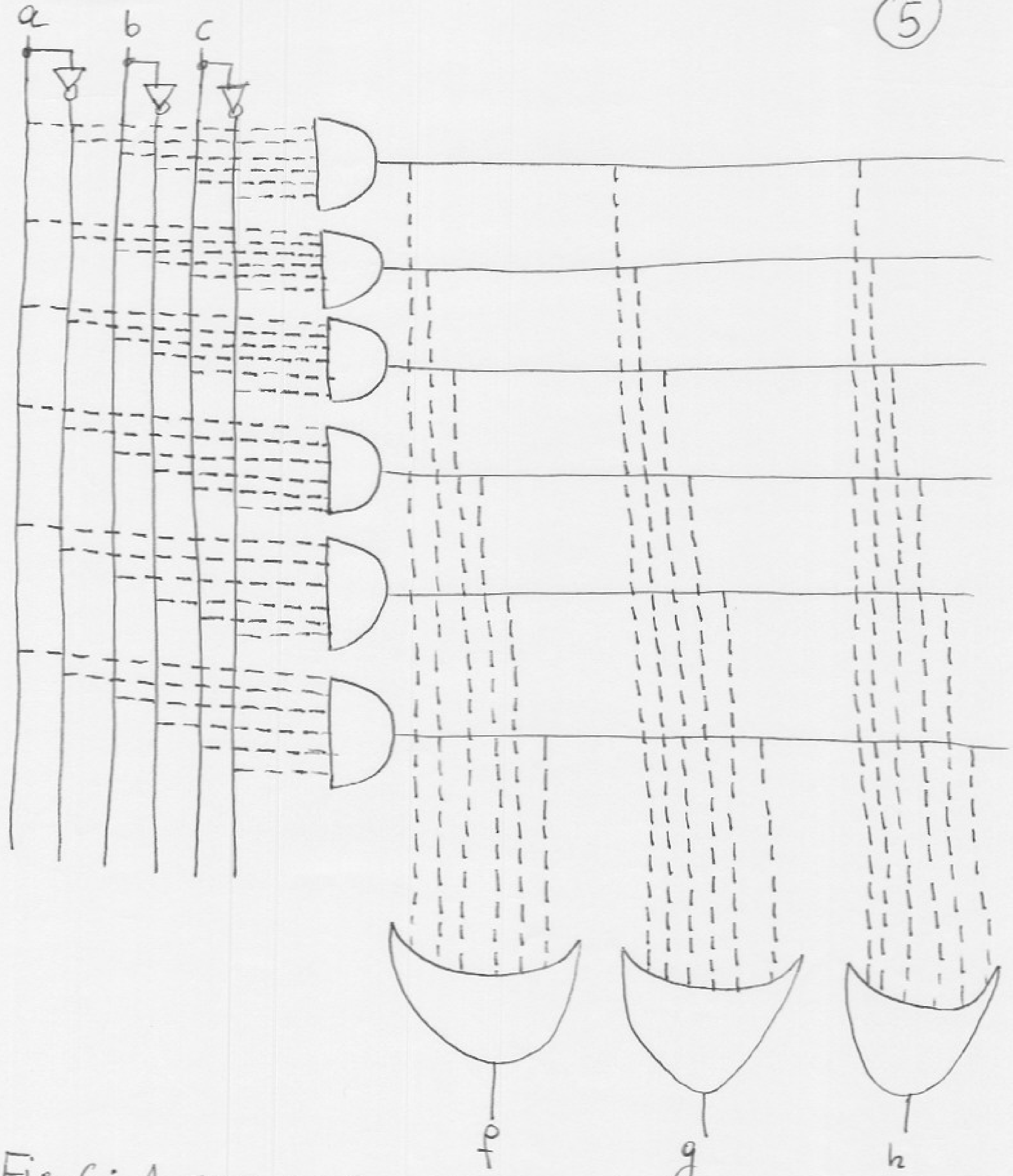

Fig.6: A 3×3 PLA with 6 product terms before programming it. The inputs to the PLA are a, b, c. The outputs are f, g, h. The inverters are internal.

In figure 6, the dashed lines indicate possible connections, not made yet.
Fig. 7 shows the PLA of fig. 6 programmed to implement f, g, h shown below:

$$f = a' \cdot b' + a \cdot b \cdot c$$
$$g = a' \cdot b' \cdot c' + a \cdot b + b \cdot c$$
$$h = a' \cdot b' + c.$$

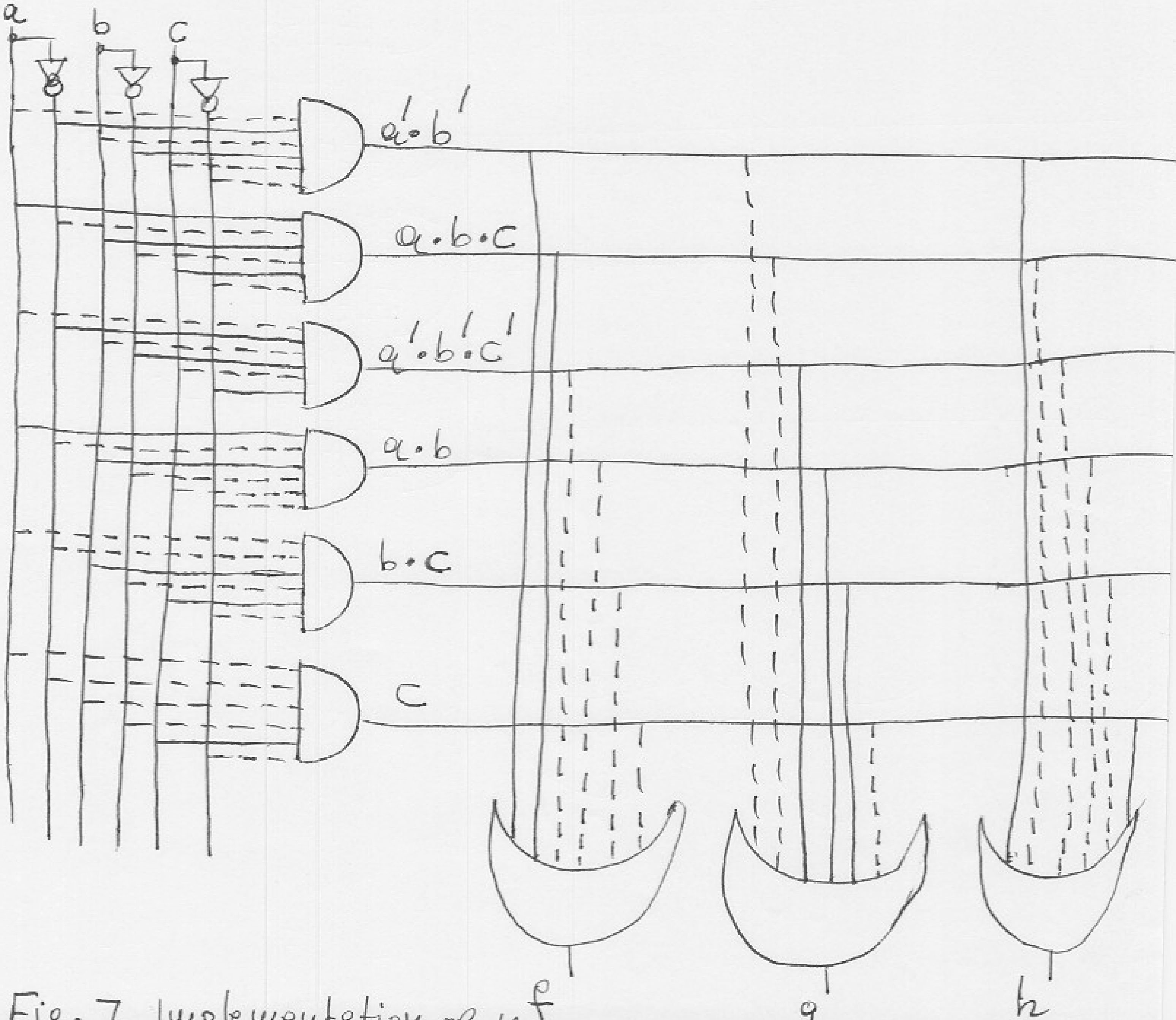

Fig. 7. Implementation of three specific functions f, g, h using the PLA of fig. 6: (it is a 3X3 PLA with 6 product terms). Solid lines indicate connections.

Note: The version of the diagram like the one of fig. 7 is rather cumbersome, particularly as the number of inputs and number of gates increase. Thus, rather than showing all the wires, only a single input line is usually shown for each gate, with $*$s or dots shown at the intersection where a ~~a~~ connection is made. This is shown in figure 8 below:

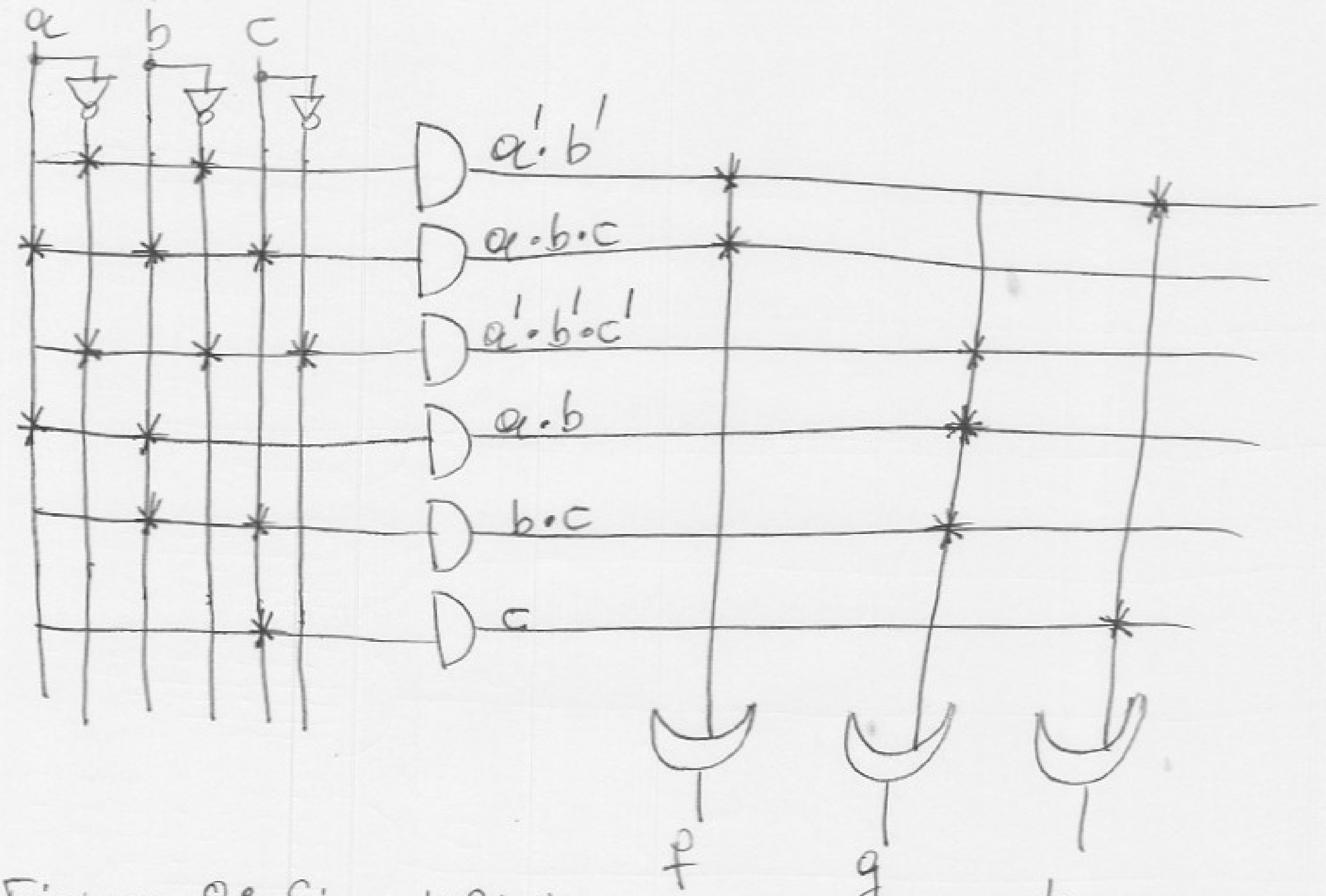

Figure 8: Simplified version of the figure 7 of the previous page.

Note: To further simplify the above figure, you can omit the AND and the OR gates and just show $*$s or dots, for the connections.

• Programmable Array Logic Devices (PALs):

A special case of a PLA, and today's most commonly used type of PLDs, is the programmable array logic (PAL) device. Unlike a PLA, in which both the AND and the OR arrays are programmable, a PAL device has a fixed OR array. The first PAL devices were introduced in the late 1970s. The PALs are easier to program and less expensive than PLAs. On the other hand, since the OR array is fixed (not programmable), it is less flexible than the PLA.

Figure 9 on the next page (it does not fit here) shows a PAL implementing the logic functions W, X, Y, where W, X, Y are:

$$W = A \cdot B' \cdot C' + C \cdot D$$

$$X = A' \cdot B \cdot C' + A' \cdot C \cdot D + A \cdot C \cdot D' + B \cdot C \cdot D$$

$$Y = A' \cdot C' \cdot D + A \cdot C \cdot D + A' \cdot B \cdot D.$$

The PAL of fig. 9 has four inputs, namely A, B, C, D, three outputs, namely W, X, Y and 12 product terms; (12 AND gates).

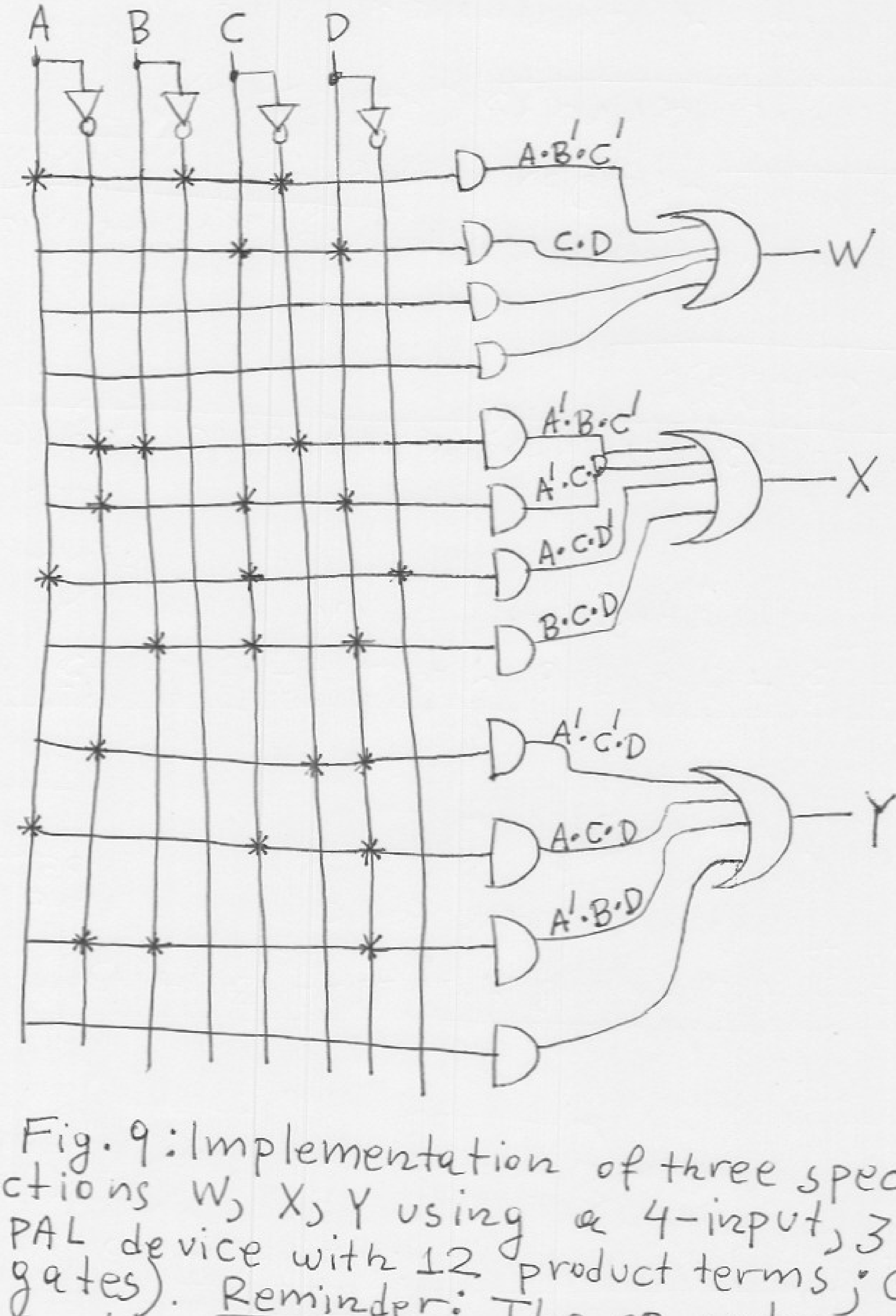

Fig. 9: Implementation of three specific functions W, X, Y using a 4-input, 3-output PAL device with 12 product terms; (12 AND gates). Reminder: The OR gates are fixed (not programmable) for a PAL. The AND gates are programmable.

• Tristate elements:

A tristate element is like a programmable switch, that can sometimes act as open (open circuit), and sometimes as closed (short circuit). The logic symbol for a tristate element is shown in figure 10, while its truth table is shown in table 1. The tristate element has two inputs namely IN (which stands for data input) and EN (which stands for enable or control input) and one output named OUT (which stands for data output)

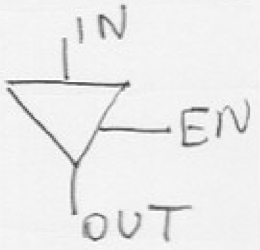

Fig.10: Logic symbol for a tristate element.

EN	IN	OUT
0	X	OPEN SWITCH; OUT and IN are isolated
1	0	0
1	1	1

Table 1: Truth table for a tristate element.

Representing the tristate element as a switch (open/closed), we have the fig.11 of the next page.

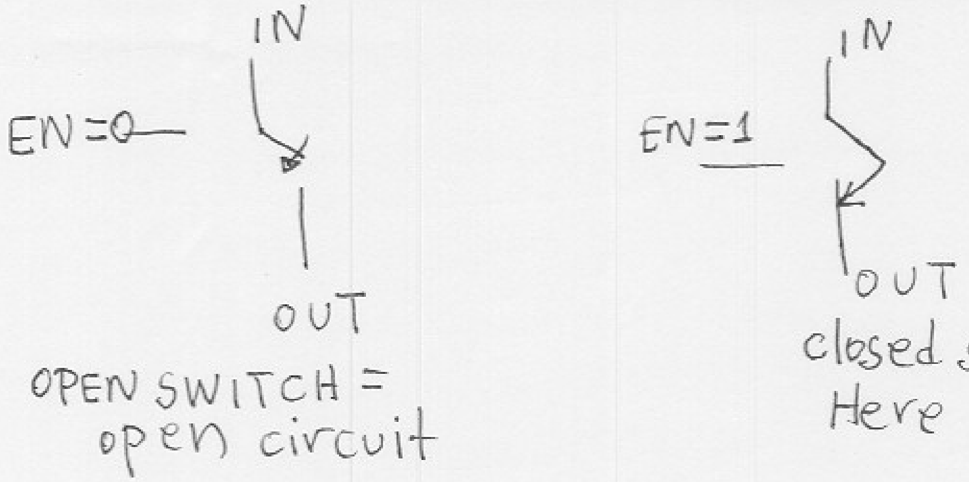

Figure 11: The tristate element modeled as a switch.

Note: When EN=0, (open switch), we say that the tristate element is found in the high impendance state or Z state.

So now, the truth table for the tristate element becomes the one shown in table 2.

EN	IN	OUT
0	X	Z or high impedance state
1	0	0
1	1	1

Table 2: Truth table for a tristate element showing the Z or high impedance state.

- Buses:

A one-bit bus is a singe bit line that is used to connect many one-bit source informations. One such arrangement is shown in figure 12 below

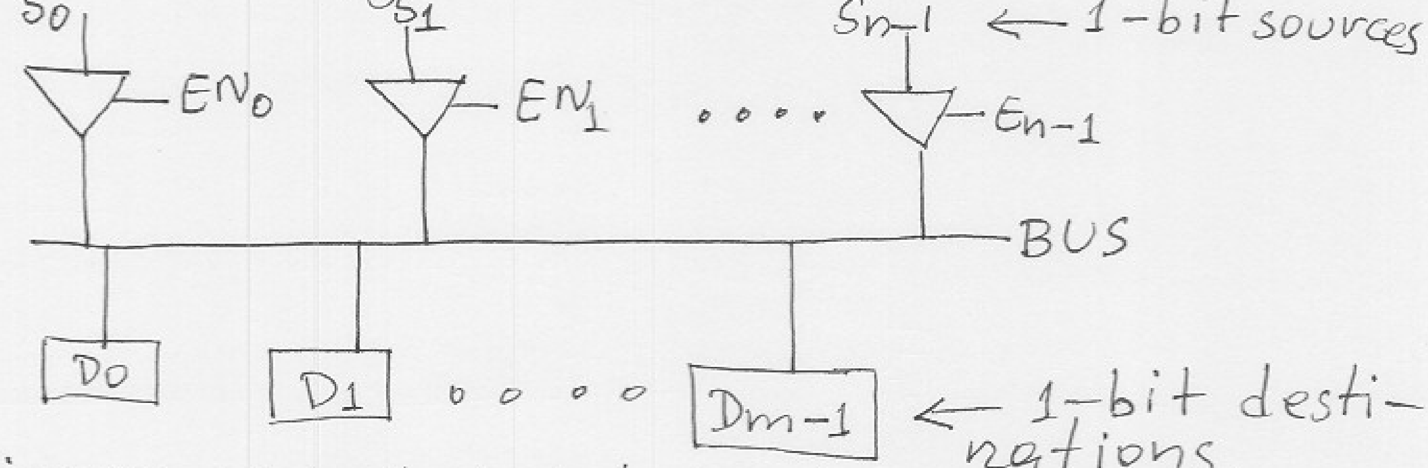

Figure 12: A tristate bus.

Important note: only one source from S_0, S_1, ..., S_{n-1} must be connected to the BUS at a time. In other words, only one of EN_0, EN_1, ..., EN_{n-1} must be 1 at a time and all others 0's.

Note: By having b copies of the arrangement of fig. 12 operating in parallel, we can transmit any b-bit information from any one of n sources to any one of m destinations. I guess I don't have to provide a figure for this.

Another way of implementing buses is by using an AND-OR implementation as shown in figure 13 below:

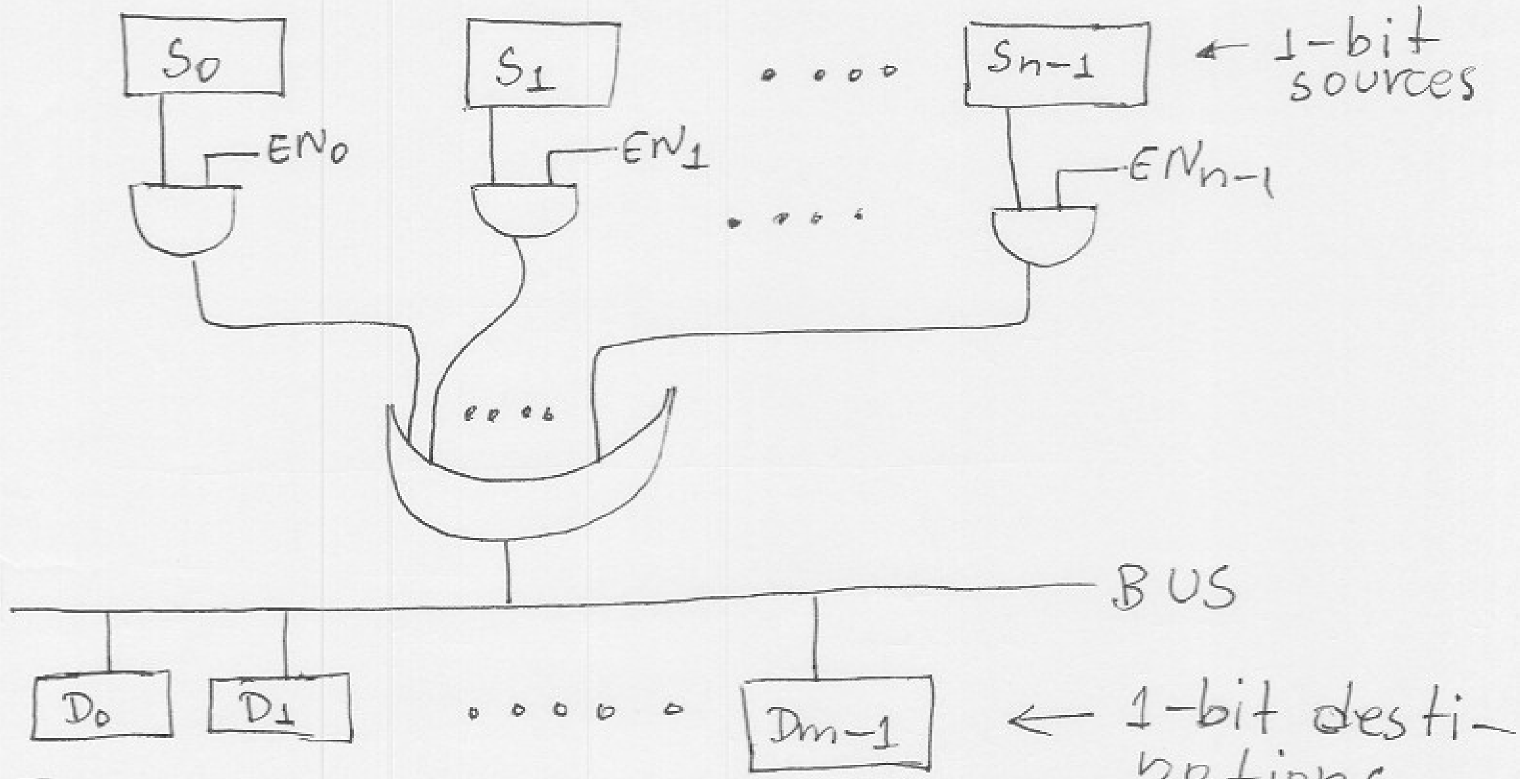

Figure 13: AND-OR implementation of a BUS.

Again, here, it must be that <u>only one</u> of $EN_0, EN_1, \ldots, EN_{n-1}$ must be 1 at a time and all others 0s, so that <u>only one</u> of $S_0, S_1, \ldots, S_{n-1}$ gets connected to the BUS.

<u>Note:</u> This is the end of EE 2720 materials. Enjoy your break!!