# **Experiment** **3**

# **Combinational Logic Design with MSI**

# **Decoders**

**Objective**

* To practice with combining small decoders to construct a bigger one.
* To practice the easiness in designing and implementing different output functions with MSI components like decoder.
* Compare that easy way of design to those time and effort consuming ways we used at experiment #1 & #2 ( with SSI components: gates ).

**Basic Concepts**

The purpose of Boolean function simplification is to obtain an algebraic expression that, when implemented, results in a low-cost circuit. Numerous MSI devices are available commercially that perform specific digital functions commonly employed in the design of digital systems. If an MSI device cannot be found to produce exactly the function needed, a resourceful designer should be able to formulate a method so as to incorporate an MSI device in his circuit.

The selection of MSI components rather than SSI gates is extremely important since, in most cases, it will result in a considerable reduction of IC packages and interconnecting wires.

**Decoders**

A decoder is any circuit that detects the presence of unique input state or condition and yields only one active output for that condition. The decoder converts the binary information present on "n" input lines to a maximum of 2n unique output lines.

For each possible input condition, one and only one output signal will be active. An n-to-2n decoder can also be considered as a "min-term generator".

In general, an n-to-m line decoder has "n" inputs and "m" outputs. n and m should satisfy m ≤ 2n. The block diagram for a 3-to-8 line decoder and the corresponding function table are shown in Figure 1 (next page).

Function table of a 3-to-8 line decoder

|  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| INPUTS | | | OUTPUTS | | | | | | | |
| *A* | *B* | *C* | *Y0* | *Y1* | *Y2* | *Y3* | *Y4* | *Y5* | *Y6* | *Y7* |
| 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 |

**3X8****DECODER**

*A*

*B*

*C*

*Y0*

*Y1*

*Y2*

*Y3*

*Y4*

*Y5*

*Y6*

*Y7*

Figure 1: Block diagram and function table of a 3-to-8 line decoder

The 3-to-8 decoder is available as a standard TTL chip (74138). The pin diagram and the function table of 74138 chips are shown in Figure 2 below. The outputs of IC 74LS138 are active low. G1, G2A’, and G2B’ are the enable inputs, and for proper operation G1 = 1, and G2A’ = G2B’ = 0.

Function Table of 74138

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| ENABLE  INPUTS | | | SELECT  INPUTS | | | OUTPUTS | | | | | | | |
| G1 | G2A’ | G2B’ | A | B | C | *Y0* | *Y1* | *Y2* | *Y3* | *Y4* | *Y5* | *Y6* | *Y7* |
| X | H | H | X | X | X | H | H | H | H | H | H | H | H |
| L | X | X | X | X | X | H | H | H | H | H | H | H | H |
| H | L | L | L | L | L | L | H | H | H | H | H | H | H |
| H | L | L | L | L | H | H | L | H | H | H | H | H | H |
| H | L | L | L | H | L | H | H | L | H | H | H | H | H |
| H | L | L | L | H | H | H | H | H | L | H | H | H | H |
| H | L | L | H | L | L | H | H | H | H | L | H | H | H |
| H | L | L | H | L | H | H | H | H | H | H | L | H | H |
| H | L | L | H | H | L | H | H | H | H | H | H | L | H |
| H | L | L | H | H | H | H | H | H | H | H | H | H | L |

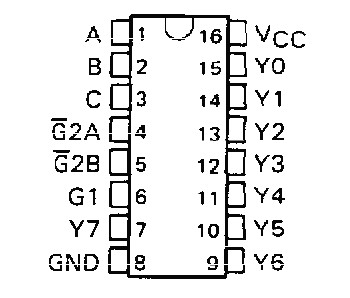
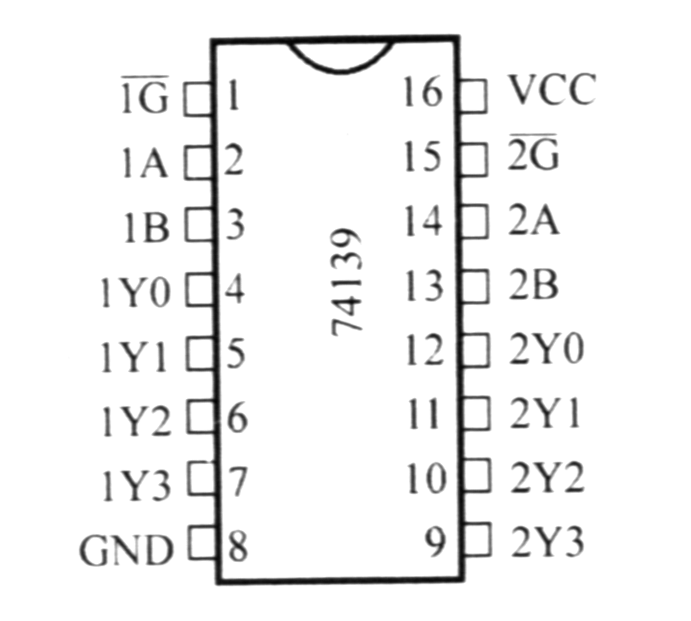


Figure 2: Pin diagram and function table of a 74LS138

The 2-to-4 decoder is also available as a standard TTL chip (74139). The pin diagram and the function table of 74139 chips are shown in Figure 3. The outputs of 74LS139 are active low. G1’ or G2’ are the enable input for each of the corresponding decoder (1 or 2) which are active low.



Function Table of 74139

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| G’ | A | B | *Y0* | *Y1* | *Y2* | *Y3* |
| H | X | X | H | H | H | H |
| L | L | L | L | H | H | H |
| L | L | H | H | L | H | H |
| L | H | L | H | H | L | H |
| L | H | H | H | H | H | L |

Figure 3: Pin diagram and function table of a 74LS13

Any Boolean function can be expressed in sum of min-terms form. Therefore, a decoder can be used to generate the min-terms. In addition an external OR gate can form the sum of the desired min-terms and implement the desired function. Since the outputs of the 74138 or the 74139 are active low, you need to replace the OR gate with a **NAND** gate.

**Procedure**

**Part 1:**

You are requested to construct and simulate using EWB a 5 x 32 decoder using one 74LS139 and four 74LS138 chips.

**Part 2:**

# You are requested to construct, simulate using EWB and implement on lab board a 4 x 16 decoder using five of 2 X 4 decoders in IC 74LS139, then use your designed 4 X 16 decoder and any number of IC 7400 to implement the following function:

F ( W, X, Y, Z ) = ∑ ( 0, 2, 8, 10)

**Part 3:**

# You are requested to construct, simulate using EWB and implement on lab board a 4 x 16 decoder using two of the 3x8 decoders in IC 74LS138 and any other IC’s, then use your designed 4 X 16 decoder and any number of IC 7400 to implement the following function:

F ( W, X, Y, Z ) = ∑ ( 0, 2, 8, 10 )

Then compare your design of this step to that of the previous step.

**Part 4:**

1. Design and implement a combinational logic circuit to multiply two unsigned two-bit integers together: *a*[1:0] and *b*[1:0] to generate the product: *P*[3:0].

Use any of the designed above 4 x 16 decoder to implement two unsigned the two-bit integers multiplication as shown in table 1 below.

A seven segment indicator of your lab board should be used to show one of the two 2 bit digits and a 3rd one for the product.

Table 1: Two unsigned two-bit integer multiplication

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **1st Number** | | **2nd Number** | | **Product** | | | |
| *a1* | *a0* | *b1* | *b0* | *P3* | *P2* | *P1* | *P0* |
| 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 |
| 0 | 0 | 1 | 1 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 0 | 1 | 0 | 1 | 0 | 0 | 0 | 1 |
| 0 | 1 | 1 | 0 | 0 | 0 | 1 | 0 |
| 0 | 1 | 1 | 1 | 0 | 0 | 1 | 1 |
| 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 0 | 0 | 1 | 0 | 0 | 1 | 0 |
| 1 | 0 | 1 | 0 | 0 | 1 | 0 | 0 |
| 1 | 0 | 1 | 1 | 0 | 1 | 1 | 0 |
| 1 | 1 | 0 | 0 | 0 | 0 | 0 | 0 |
| 1 | 1 | 0 | 1 | 0 | 0 | 1 | 1 |
| 1 | 1 | 1 | 0 | 0 | 1 | 1 | 0 |
| 1 | 1 | 1 | 1 | 1 | 0 | 0 | 1 |

1. Design and implement the comparator combinational logic circuit for the two 2-bit unsigned numbers we designed before in experiment #2 using any of the 4 X 16 decoder we designed in this experiment then compare your design with that uses NAND and NOR gates in experiment #2 only.

Note that you have to show every 2 bit digit on a numeric display and the comparison result on one of the board leds as in experiment #2.