Chapter 4:

Table of Contents

Decoders

Table of Contents

Modular Combinational Logic - Decoders ................................................................. 2
The generic decoder .................................................................................................. 2
  The 74139 decoder .......................................................................................... 3
The decoder specification sheet .............................................................................. 4
A decoder with active High outputs ....................................................................... 8
A Negative Logic Solution ...................................................................................... 8
The 3/8 decoder ..................................................................................................... 9
The 4/16 decoder .................................................................................................. 11
Decoder Case Study #2: Implementing a Binary Adder with a Decoder ............. 16
Modular Combinational Logic - Decoders

The Generic Decoder

A decoder is a min-term generator with each output corresponding to a single min-term. They are generally used for code conversions (binary to decimal), data routing, or equation creation. They are also referred to as "line decoders" due to the fact that the user can "activate" an output line by specifying a "control word".

The generic discrete 2/4 decoder in the figure to the right has active high inputs and outputs. Each output (or min-term) in the circuit is produced by AND'ing the specified control signals $S_1$, $S_0$, or $S_2$, which results in a unique output. It should be noted that there is absolutely NO way that more than one output can be active at the same time. The terminology 2/4 indicates that two inputs decode into four outputs.

As stated earlier, a decoder will ALWAYS have ONE and ONLY ONE ACTIVE output at a time. Normally, you will have two 2:4 Decoders on a single 14 pin chip. As is demonstrated in the circuit below, by combining the outputs with 'OR' gates you can create unique SOP equations.

It should be obvious to the student that the fewer the number of chips it takes to produce a logic expression:

- the lower real-estate on the printed circuit board,
- the lower power dissipation,
- the lower the manufacturing costs,
- fewer chips means higher reliability.
Generic Example:

In the example circuit to the right, a generic 2-4 decoder chip has one of its decoders being used to create two separate switching expressions.

Thus far, we have only discussed generic decoders with active high inputs/outputs. There are several popular decoders on the market which utilize active high and/or active low inputs and outputs. One such device is the 74139 as discussed next.

The 74139 decoder

The 74139 is an excellent example of a dual 2/4 decoder chip. It features:

- Active High Inputs
- Active Low Outputs
- Active Low Enable Input

As can be seen in the circuit to the right, the enable, \( G \), goes to one input on each of the NAND gates. There is no way that anything except a high can be seen at the outputs unless this enable input is Active, \( 0 \). Therefore, if the device is disabled, all outputs go high since they are active low outputs. If they were active high outputs, they would go low when disabled.

Note the additional inverter in the select inputs shown in the diagram:

**Question:** What does it do?

**Answer:** It improves the fan-in. To the outside world, it looks like one gate vice three gates, which is what would be seen without the inverter.
The Decoder Specification sheet

Now let's examine a few entries on the 74139 specification sheet to the right.

<table>
<thead>
<tr>
<th></th>
<th>SN74LS139A</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>MIN</td>
</tr>
<tr>
<td>Supply Voltage</td>
<td>4.75</td>
</tr>
<tr>
<td>High-level input voltage</td>
<td>2</td>
</tr>
<tr>
<td>Low-level input voltage</td>
<td>0.8</td>
</tr>
<tr>
<td>High-level output current</td>
<td>-0.4</td>
</tr>
<tr>
<td>Low-level output current</td>
<td>8</td>
</tr>
</tbody>
</table>

- Note the difference between the **maximum** supply voltage and the **recommended** supply voltage.
  - This tends to be fairly standard with TTL devices.

- The table also indicates that a
  - "HIGH" is 2v or greater  (High-Level Input Voltage)
  - "LOW" is < .8v          (Low-Level Input Voltage)

- The high and low output currents are also given. However, as long as you **don't exceed the fan-out**, you don't have to worry about these items. There is one other item which is very important.
  - The **Short Circuit Output Current** - No more than one output should be shorted or you could exceed this number.
    - Again, this should not come as a surprise that if you **SHORT** outputs together that they will at the very least be degraded.
2/4 Decoder Examples

Example 4-4a

Let’s take a look at a decoder with active low outputs:

<table>
<thead>
<tr>
<th>( I_1 )</th>
<th>( I_0 )</th>
<th>( D_3 )</th>
<th>( D_2 )</th>
<th>( D_1 )</th>
<th>( D_0 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Multisim Implementation of Example 4-4a

Ex 4-4a

Word Generator "Step mode" is selected. Start the sim and use the pause/resume button next to the on/off switch to step thru the program.
Example 4-4b

\[ f(A, B) = \overline{m_1} \cdot \overline{m_2} \text{ (NAND)} \]
\[ = m_1 + m_2 = m_1 + m_2 \]
\[ = \sum m(1, 2) \]

MultiSim implementation of 4-4b

The Word Generator is programmed as before. Note that this time the Chip Enable has been tied to Bit - 2 of the Word Generator which is always 0 in this example. Thus, it acts the same as tying the input to ground.
Example 4-4c

Note that we can achieve the same results as the last example if we use an AND gate vice a NAND gate but we need to attach to outputs 0 and 3 instead.

\[ f(A, B) = \sum m(1, 2) \]

MultiSim implementation of 4-4c

The NAND gate was replaced by an AND gate.

Ex 4-4c

Step thru program as in earlier examples. You should get a High for min-terms 1 and 2. Program was changed to include a couple of steps where the chip is disabled.
A decoder with active High outputs

Now let’s look at a decoder with Active High Outputs compared with one with Active Low Outputs. The circuit above demonstrates that we can AND the ACTIVE LOW outputs that are not in our min-term list and get the desired min-term list. Note the difference in this circuit and the one on the previous page.

The circuit below has ACTIVE HIGH outputs. Note that we can use the more obvious OR gate for the solution.

A Negative Logic Solution

ACTIVE LOW outputs sometimes give people trouble. Let’s look at the same thing now but use negative logic to help clear up the situation.

Remember that an SOP expression is a SUMMATION of individual min-terms. Also remember that when we combine positive and negative logic, we can cancel out MATCHED Bubbles. So, if the matched bubbles are canceled we can see that we really do have: $f(A, B) = \sum \begin{array}{c} m_1, m_2 \end{array}$

Unfortunately, MultiSim doesn’t have Negative logic gates. This is just a method of analyzing the circuit so that it means something.
The 3/8 decoder

Now, let's demonstrate how we can use **two 2/4 decoders to build a single 3/8 decoder**. This conversion is performed with the addition of an inverter to the circuit. As can be seen below, when one device is active, the other will be inactive. If we make the input to the Enable's the **MSB of the input control word**, we now have **three inputs decoded to eight outputs**.

Two 2:4 decoders equate to One 3:8 decoder.

Note: For this example, the enables are **ACTIVE HIGH**.

MultiSim Example of building a 3/8 decoder
Note that so far we have had active low output displays for active low outputs. The following is an example of how we can cause the display to display active high when we have active low outputs.

When the Bar graph was placed in the circuit, it was flipped horizontally so that the cathode and anode were switched. Then the anode was attached to $V_{cc}$ vice ground.

**Multisim Example: Making Active Low look like Active High**

![Multisim Diagram](image)

**A MultiSim realization of a 74138 (using Busses)**

In this example the chip is enabled until the last 3 program steps.
The 4/16 decoder

Let’s take a look at an even larger decoder. We can create a 4/16 decoder using five 2/4 decoders.

In the figure, a fifth decoder is used to select which of the four other decoders is active.

A MultiSim Realization of a 5 chip 4/16 decoder

In the example below, remember that the outputs have been inverted to appear like they are active high by reversing the LED's at the output and tying them to Vcc vice ground.
The 74154 4/16 decoder

The 74154 is an example of a popular "off-the-shelf" 4/16 decoder. It features active high inputs and active low outputs, with two active low enable inputs.

Question: While the 74154 is a very popular decoder chip, what are the advantages to using the five 2/4 decoders option instead?

Answer: The 4/16 Decoder chip is a 24 pin dip with a 0.6" center vice the 0.3" center for the 2/4 decoder 16 pin dip. If the 4/16 decoder chip was the only 24 pin chip on the PC board, the price of the completed board might be cheaper if the designer chose to use the 0.3" center devices instead.

Let's look at the same example, but this time we used mixed logic to see if it makes the resulting expressions any clearer.

Remember that in mixed logic, if you can match bubbles, the bubbles cancel out.
MultiSim Example of a 4 to 16 decoder implementation
Decoder Case Study #1: BCD to Decimal Decoder

Of course, we have already designed a BCD to Decimal Decoder out of gates with the use of K-maps and ‘Don’t Cares’ in a previous chapter. But we could use just a 74154 4:16 decoder to do the same job. Since the BCD numbers are equivalent to decimal numbers from 0-9, all we have to do is use the 74154 and ignore the outputs 10-15. This may be more expensive than using the cheaper gates but it might save money in the long run due to:

- cheaper construction cost and
- lower real estate taken up on the PC board.

And don’t forget that the output of the 74154 is negative logic so you would have to take that into account.
Multisim Implementation of a BCD to Decimal Decoder

(Note that the 7-segment display is a "common cathode" or "CK" type. In order for it to work in Multisim it SOMETIMES has to have the CK input grounded thru a 75 ohm resistor.)
Decoder Case Study #2: Implementing a Binary Adder with a Decoder

**Contemporary Approach**

The truth table for a full adder is as follows:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Ci</th>
<th>S_o</th>
<th>C_o</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

The sum $S_o$ is given by:

$$S_o = \overline{A} \cdot B \cdot C_i + \overline{A} \cdot \overline{B} \cdot C_i + A \cdot B \cdot C_i + A \cdot B \cdot C_i$$

$$S_o = A \oplus B \oplus C_i$$

The carry out $C_o$ is given by:

$$C_o = A \cdot B + A \cdot C + B \cdot C$$

$$C_o = A \cdot B + C_i \cdot (A \oplus B)$$
**Decoder Approach**

From the Full Adder table on the previous page, we can derive the following min-term lists:

$$S_o(A,B,C_i) = \sum m(1,2,4,7)$$

$$C_o(A,B,C_i) = \sum m(3,5,6,7)$$

Note that we used the negative logic NAND gates to view this representation. We used a total of 2 chips to implement this circuit while the contemporary method used 3 chips (1 XOR, 1 AND, 1 OR).

![Diagram of decoder circuit with inputs A, B, C_i and outputs S_o, C_o]