Project Part I

8-bit 4-to-1 Line Multiplexer

Specification:
This section of the project outlines the design of a 4-to-1 multiplexor which takes two 8-bit buses as inputs and produces a single 8-bit bus as output. A multiplexor (MUX) is a combinational circuit that utilizes selection inputs to choose binary information from multiple inputs and directs it to a single output (Mano and Kime, 149).

Formulation and Optimization:
The Boolean expression for a 4-to-1 line multiplexer is as follows:

\[ Y = \overline{S_1}S_0I_0 + \overline{S_1}S_0I_1 + \overline{S_1}S_0I_2 + S_1S_0I_3 \]

The gate-input cost to implement this expression is 18. Traditionally, this equation is factored to construct a slightly different implementation of the multiplexor. Although the gate-input cost of the second expression is higher at 22, this implementation facilitates expansion of the circuit to include more inputs (Mano and Kime, 151).

The factored expression is as follows:

\[ Y = (\overline{S_1}S_0)I_0 + (\overline{S_1}S_0)I_1 + (S_1\overline{S_0})I_2 + (S_1S_0)I_3 \]

The complete truth table for a 4-to-1 line multiplexer consists of 16 rows. A condensed version, given in Table 1, illustrates the possible values for selector variables \( S_1 \) and \( S_0 \) and the corresponding input variable \( I \) that is chosen to pass the data. For the selected input line, the output will be equal to the value of the input (Mano and Kime, 150).

<table>
<thead>
<tr>
<th>( S_1 )</th>
<th>( S_0 )</th>
<th>( Y )</th>
</tr>
</thead>
<tbody>
<tr>
<td>0 0</td>
<td>I_0</td>
<td></td>
</tr>
<tr>
<td>0 1</td>
<td>I_1</td>
<td></td>
</tr>
<tr>
<td>1 0</td>
<td>I_2</td>
<td></td>
</tr>
<tr>
<td>1 1</td>
<td>I_3</td>
<td></td>
</tr>
</tbody>
</table>

Table 1: 4-to-1 Line Multiplexer Condensed Truth Table

The implementation of the 4-to-1 line multiplexer is illustrated in Figure 1. The design consists of a 2-to-4 line decoder on the left side, with two single-bit selection inputs, \( S_1 \) and \( S_0 \). The four inputs are 8-bit busses \( I_0, I_1, I_2 \) and \( I_3 \). The AND-OR circuit on the right side of the diagram is broken down into sub designs, the AND portion and the OR portion. Each circuit sub design is imported hierarchically to preserve clarity and organization of the combinational circuit.
The AND gate circuit sub design is shown in Figure 2. The input 8-bit bus, B8i, is separated into single bits with the use of an expander. Each bit then passes through an AND gate along with the selector signal S. The outputs of each AND gate are then merged into a single 8-bit bus.
The OR gate circuit sub design is shown in Figure 3. Four 8-bit bus inputs are expanded into single bits. Four bits of the same position, each originating from a different 8-bit bus, pass through an OR gate and the output is then once again merged into an 8-bit bus.

**Figure 3: 8-Bit Bus OR Circuit**

**Verification:**

In order to verify the functionality of the 8-bit 4-to-1 multiplexer, it is necessary to first test the two sub components. The AND sub design is tested by setting the selector S first equal to 0 and then to 1. When the selector signal is 0, the output is always 0 as well. When the selector signal is 1, the AND circuit is enabled, and each of the 8 bits passes through an AND gate. When the 8-
bit input value is set to 0, the output is also 0 for every bit. When each of the eight bits are set to 1 (11111111 in binary, or 255 in decimal), then the output is also equal to 255 in decimal. The waveform in Figure 4 (n0 is the input, n1 is the selector and n18 is the output) illustrates this test.

Next, the OR sub design is verified by setting each bit equal to 1 in every 8-bit bus input, while the other three input busses are set to 0. This test is illustrated in the waveform in Figure 5 (n0 is B8i0, n1 is B8i1, n2 is B8i2, n3 is B8i3, and n44 is B8o).

The complete 8-bit 4-to-1 line multiplexer is tested by checking each possible input combination of the selector inputs S1 and S0, as outlined in the truth table in Table 1.

This test is illustrated in the waveform in Figure 6 (n0 is S1, n2 is S0, and n16 is Y)
8-Bit Adder

Specification:

This section of the project outlines the design of an 8-bit adder. An adder is an arithmetic combinational circuit that performs the addition operation on a given number. (Mano and Kime, 169)

Formulation and Optimization:

The 8-bit adder is constructed using eight full adders, each of which performs addition on three input bits. The Boolean expressions for the sum (S) and the carry (C) of a full adder are as follows:

\[
S = (X \oplus Y) \oplus Z \\
C = XY + Z(X \oplus Y)
\]

The complete truth table for a full adder is shown in Table 2 (Mano and Kime, 170).

<table>
<thead>
<tr>
<th>X</th>
<th>Y</th>
<th>Z</th>
<th>C</th>
<th>S</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>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

Table 2: Full Adder Truth Table

The implementation of a full adder, shown in Figure 7, consists of two single bit inputs, A and B, and a carry input C_i. The outputs are sum S and carry out C_o. As the truth table illustrates, when both input bits are 1, the sum output value is 0, while the carry output holds a 1.
The carry function of the full adder enables addition of multiple bit numbers. The 8-bit adder design (Figure 8) utilizes eight full adder sub designs connected to form a chain. The carry output from one full adder is wired to the carry input of the next adder, thus forming a ripple carry adder. Two 8-bit bus inputs, A and B, are expanded into single bits and connected to the appropriate full adder, corresponding to bit position. The $C_i$ for the full adder of the least significant bit is set to 0. The carry output of the most significant bit adder is $C_0$ (Mano and Kime, 172). Hierarchical design through the use of full adder sub components helps to maintain organization of the complete circuit diagram.
Verification:

To verify functionality of the full adder sub component it is sufficient to test only one instance of the circuit. Since the 8-bit adder is constructed using eight exact copies of the sub design, when one copy is proven to work correctly, all the rest are assumed to also be correct. The verification of the full adder functionality tests each of the eight possible input combinations, as outlined in the truth table (Table 2). The waveform for this test (n0 is A, n1 is B, n7 is C_i, n6 is S, and n3 is C_o) is shown in Figure 6.

![Figure 6: Full Adder Waveform](image-url)