Lecture 11: Hardware for Arithmetic

• Today’s topics:
  ▪ Logic for common operations
  ▪ Designing an ALU
Boolean Algebra

• Equations involving two values and three primary operators:
  - **OR**: symbol $+$, $X = A + B \Rightarrow X$ is true if at least one of $A$ or $B$ is true
  - **AND**: symbol $\cdot$, $X = A \cdot B \Rightarrow X$ is true if both $A$ and $B$ are true
  - **NOT**: symbol $\overline{}$, $X = \overline{A} \Rightarrow X$ is the inverted value of $A$
Boolean Algebra Rules

- Identity law: $A + 0 = A$ ; $A \cdot 1 = A$
- Zero and One laws: $A + 1 = 1$ ; $A \cdot 0 = 0$
- Inverse laws: $A \cdot \bar{A} = 0$ ; $A + \bar{A} = 1$
- Commutative laws: $A + B = B + A$ ; $A \cdot B = B \cdot A$
- Associative laws: $A + (B + C) = (A + B) + C$
  \[ A \cdot (B \cdot C) = (A \cdot B) \cdot C \]
- Distributive laws: $A \cdot (B + C) = (A \cdot B) + (A \cdot C)$
  \[ A + (B \cdot C) = (A + B) \cdot (A + C) \]
DeMorgan’s Laws

• \( A + B = \overline{A} \cdot \overline{B} \)

• \( A \cdot B = \overline{A} + \overline{B} \)

• Confirm that these are indeed true
Pictorial Representations

AND  OR  NOT

Source: H&P textbook

What logic function is this?

Source: H&P textbook
Boolean Equation

• Consider the logic block that has an output $E$ that is true only if exactly two of the three inputs $A$, $B$, $C$ are true

Multiple correct equations:

Two must be true, but all three cannot be true:
$E = ((A \cdot B) + (B \cdot C) + (A \cdot C)) \cdot (A \cdot B \cdot C)$

Identify the three cases where it is true:
$E = (A \cdot B \cdot \overline{C}) + (A \cdot C \cdot \overline{B}) + (C \cdot B \cdot \overline{A})$
Sum of Products

- Can represent any logic block with the AND, OR, NOT operators
  - Draw the truth table
  - For each true output, represent the corresponding inputs as a product
  - The final equation is a sum of these products

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>E</th>
</tr>
</thead>
<tbody>
<tr>
<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>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<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>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

\[(A \cdot B \cdot \overline{C}) + (A \cdot C \cdot \overline{B}) + (C \cdot B \cdot \overline{A})\]

- Can also use “product of sums”
- Any equation can be implemented with an array of ANDs, followed by an array of ORs
NAND and NOR

• NAND: NOT of AND: \( A \text{ nand } B = A \cdot B \)

• NOR: NOT of OR: \( A \text{ nor } B = A + B \)

• NAND and NOR are *universal gates*, i.e., they can be used to construct any complex logical function
Common Logic Blocks – Decoder

Takes in N inputs and activates one of $2^N$ outputs

<table>
<thead>
<tr>
<th>$I_0$</th>
<th>$I_1$</th>
<th>$I_2$</th>
<th>$O_0$</th>
<th>$O_1$</th>
<th>$O_2$</th>
<th>$O_3$</th>
<th>$O_4$</th>
<th>$O_5$</th>
<th>$O_6$</th>
<th>$O_7$</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<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>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

3-to-8 Decoder

$O_{0-7}$
**Common Logic Blocks – Multiplexor**

- Multiplexor or selector: one of $N$ inputs is reflected on the output depending on the value of the $\log_2 N$ selector bits.

Source: H&P textbook

2-input mux

Source: H&P textbook
Adder Algorithm

\[
\begin{array}{cccc}
1 & 0 & 0 & 1 \\
0 & 1 & 0 & 1 \\
\hline
\text{Sum} & 1 & 1 & 1 & 0 \\
\text{Carry} & 0 & 0 & 0 & 1 \\
\end{array}
\]

Truth Table for the above operations:

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Cin</th>
<th>Sum</th>
<th>Cout</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>
Adder Algorithm

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>Cin</th>
<th>Sum</th>
<th>Cout</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>

Equations:

Sum = Cin \cdot \overline{A} \cdot \overline{B} + B \cdot \overline{Cin} \cdot \overline{A} + A \cdot \overline{Cin} \cdot \overline{B} + A \cdot B \cdot Cin

Cout = A \cdot B \cdot Cin + A \cdot B \cdot Cin + A \cdot \overline{Cin} \cdot \overline{B} + B \cdot Cin \cdot \overline{A} = A \cdot B + A \cdot Cin + B \cdot Cin
Carry Out Logic

Equations:

\[
\text{Sum} = \text{Cin} \cdot \overline{A} \cdot \overline{B} + \\
B \cdot \text{Cin} \cdot \overline{A} + \\
A \cdot \text{Cin} \cdot \overline{B} + \\
A \cdot B \cdot \text{Cin}
\]

\[
\text{Cout} = A \cdot B \cdot \text{Cin} + \\
A \cdot B \cdot \text{Cin} + \\
A \cdot \text{Cin} \cdot \overline{B} + \\
B \cdot \text{Cin} \cdot \overline{A} \\
= A \cdot B + \\
A \cdot \text{Cin} + \\
B \cdot \text{Cin}
\]

Source: H&P textbook
1-Bit ALU with Add, Or, And

- Multiplexor selects between Add, Or, And operations

Source: H&P textbook
32-bit Ripple Carry Adder

1-bit ALUs are connected “in series” with the carry-out of 1 box going into the carry-in of the next box.

Source: H&P textbook
Incorporating Subtraction

Must invert bits of B and add a 1
  • Include an inverter
  • CarryIn for the first bit is 1
  • The CarryIn signal (for the first bit) can be the same as the Binvert signal

Source: H&P textbook
Incorporating NOR and NAND

[Diagram of a circuit with labels for Ainvert, Binvert, CarryIn, Operation, CarryOut, Result, a, b, 0, 1, 2, +]

Source: H&P textbook
Incorporating slt

- Perform a – b and check the sign

- New signal (Less) that is zero for ALU boxes 1-31

- The 31st box has a unit to detect overflow and sign – the sign bit serves as the Less signal for the 0th box

Source: H&P textbook
Incorporating beq

• Perform $a - b$ and confirm that the result is all zero's

Source: H&P textbook
Control Lines

What are the values of the control lines and what operations do they correspond to?
Control Lines

What are the values of the control lines and what operations do they correspond to?

<table>
<thead>
<tr>
<th></th>
<th>Ai</th>
<th>Bn</th>
<th>Op</th>
</tr>
</thead>
<tbody>
<tr>
<td>AND</td>
<td>0</td>
<td>0</td>
<td>00</td>
</tr>
<tr>
<td>OR</td>
<td>0</td>
<td>0</td>
<td>01</td>
</tr>
<tr>
<td>Add</td>
<td>0</td>
<td>0</td>
<td>10</td>
</tr>
<tr>
<td>Sub</td>
<td>0</td>
<td>1</td>
<td>10</td>
</tr>
<tr>
<td>SLT</td>
<td>0</td>
<td>1</td>
<td>11</td>
</tr>
<tr>
<td>NOR</td>
<td>1</td>
<td>1</td>
<td>00</td>
</tr>
</tbody>
</table>

Source: H&P textbook
Title

• Bullet