Simple Computations

- Adding binary numbers
- Subtracting binary numbers
- Selectors
addition
1-Bit Adder

Let’s start simple: Adding two 1-Bit numbers

• Truth table

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A+B</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>10</td>
</tr>
</tbody>
</table>
## Really 2 Operations

- Truth table for "position 0" bit

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

- Truth table for carry bit

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>A+B</th>
<th>carry</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
</tr>
</tbody>
</table>

**xor**

**and**
Reminder: Basic Gates

- AND
- OR
- NOT
- NAND
- NOR
Circuits

- "Position 0" bit

\[
\begin{array}{c|c|c}
A & B & \text{OUT0} \\
\hline
0 & 0 & 0 \\
0 & 1 & 1 \\
1 & 0 & 1 \\
1 & 1 & 0 \\
\end{array}
\]

- Carry bit

\[
\begin{array}{c|c|c}
A & B & \text{OUTC} \\
\hline
0 & 0 & 0 \\
0 & 1 & 0 \\
1 & 0 & 0 \\
1 & 1 & 1 \\
\end{array}
\]
Putting them Together

A

B

OR

NAND

AND

CARRY

OUT0
N-Bit Addition

\[
\begin{align*}
11 \\
+11 \\
\hline
110
\end{align*}
\]
N-Bit Addition

\[
\begin{align*}
1 &+ 1 = 0, \text{ carry the 1}
\end{align*}
\]
N-Bit Addition

\[
\begin{array}{c}
11 \\
+11 \\
\hline \\
11 \\
\hline \\
10 \\
\end{array}
\]

1+1+1 = 1, carry the 1
N-Bit Addition

\[
\begin{array}{c}
11 \\
+11 \\
\hline
110
\end{array}
\]

copy carry bit
1-Bit Adder

Our adder cannot handle carry as input yet.
Half 1-Bit Adder

A
B

OR
NAND
AND

OUT0
CARRY

1-BIT HALF ADDER
Building a 1-Bit Full Adder

\[ \text{A} \quad \text{1-BIT HALF ADDER} \quad \text{S} \quad \text{SUM} \quad \text{B} \quad \text{CARRY} \]
Building a 1-Bit Full Adder
Building a 1-Bit Full Adder
1-Bit Full Adder

[Diagram of a 1-bit full adder with input A, input B, carry input, carry output, sum output, and OR gate]
N-Bit Full Adder

\[
\begin{array}{c}
11 \\
+ 11 \\
\hline
110
\end{array}
\]
N-Bit Full Adder

\[ \begin{array}{c}
11 \\
+ 11 \\
\hline
1 \\
\hline
0
\end{array} \]

\[
\begin{array}{c}
B0 \\
A0 \\
\hline
A \\
B \\
C1 \\
\hline
C0 \\
S \\
\hline
SUM0
\end{array}
\]
N-Bit Full Adder

11 + 11 = 11
---
10

1-BIT FULL ADDER
A B

SUM1

B1
A1

1-BIT FULL ADDER
A B

SUM0

B0
A0
N-Bit Full Adder

and so on ...
subtraction
First, a Trick

• Normally, we subtract like this:

\[
\begin{align*}
253 \\
-176 \\
\hline
11 \\
\hline
77
\end{align*}
\]
Computing the Inverse

• Now we use the inverse of the subtrahend

\[
\begin{align*}
\phantom{999} & \quad 999 \\
-176 & \quad 723 \\
\phantom{-176} & \quad 823
\end{align*}
\]
Subtraction by Addition

• This allows us to carry our subtraction by addition

\[
\begin{array}{c}
253 \\
+ 823 \\
\hline \\
1076
\end{array}
\]

• Well, with minor corrections

\[
\begin{array}{c}
1076 \\
+ 1 \\
-1000 \\
\hline \\
77
\end{array}
\]
Also Works in Binary

Original problem

253 \quad 11111101
- 176 \quad -10110000
----- \quad --------
77 \quad 01001101

Inverse of subtrahend

823 \quad 01001111

Addition

253 \quad 11111101
+ 823 \quad + 01001111
----- \quad --------
1076 \quad 101001100

Corrections

+ 1 \quad + 1
-1000 \quad -10000000
--------- \quad ---------
77 \quad 01001101
Start with N-Bit Adder
Invert Bits of Subtrahend

![Diagram of 4-bit full adder](image)

- \( A_0, A_1, A_2, A_3 \) for the augend
- \( B_0, B_1, B_2, B_3 \) for the subtrahend (bits are inverted in this context)
- \( S_0, S_1, S_2, S_3 \) for the sum
- \( C_0, C_1 \) for carry-in and carry-out
Add One

4-BIT FULL ADDER

A0 A1 A2 A3 B0 B1 B2 B3

S0 S1 S2 S3

CO CI

V

Trick: add one as carry in
Invert Overflow --- DONE
unifying
addition and subtraction
machines
Goal

- Not two machines for addition and subtraction

⇒ Combined adder and subtractor

- Input: A, B, and subtraction flag SUB

- Output
  - if SUB=0: A+B
  - if SUB=1: A-B
NOT only if SUB
• Truth table

<table>
<thead>
<tr>
<th>SUB</th>
<th>X</th>
<th>OUT</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

• Looks like XOR

[XOR symbol diagram]
Combined Machine

A0 A1 A2 A3 B0 B1 B2 B3

4-BIT FULL ADDER

CO CI

S0 S1 S2 S3

OVERFLOW SUM

SUB

SUB

SUB

OVERFLOW SUM

SUB

SUB