Binary Addition: recall the full-adder design.
Integer Addition

Full-adder:

- Three input bits $a, b, s$
- Output: two bits $sum$ and $carry$

Logic equations and gate diagram derived from truth-tables.

What about 4-bit addition?
Integer Addition

Solution 1: write truth-table, derive logic equations, draw gate diagram.

Solution 2:

\[
\begin{array}{cccc}
1 & 1 & 0 & 0 \\
1 & 0 & 1 & 0 \\
+ & 0 & 1 & 1 \\
\hline
1 & 0 & 0 & 0 & 1 \\
\end{array}
\]

Use a number of full-adders!
Integer Addition

2's complement? Addition time for $N$ bits?
Integer Addition

Observation: all we need is the carry-out...

⇒ compute carry-out $cout$ for blocks

- input: 0 0, $cout = 0$ kill
- input: 1 1, $cout = 1$ generate
- input: 0 1 or 1 0, $cout = \text{carry-in (cin)}$ propagate

$$cout = cin \cdot P + G$$
$$G = a \cdot b$$
$$P = a + b$$

Block codes:

$$G_{01} = G_1 + G_0P_1$$
$$P_{01} = P_0P_1$$
Integer Addition

**Carry Lookahead adder:** compute block codes to speed up carry computation.
Subtraction

To calculate $a - b$, use $a + (-b)$. To calculate $-b$, flip all the bits and add 1.

⇒ build it using an adder
Given: one bit of control $c$, two $N$ bit inputs $a$ and $b$. Compute $a + b$ if $c = 0$, $a - b$ if $c = 1$.

- Carry-in to the adder is $c$
- one input: $a$
- other input: $b$ if $c = 0$, complement of $b$ if $c = 1$.

Standard element: MUX (multiplexor)
Combined Add/Subtract Unit

- Hierarchical design
- Reuse components
- Replication
**Shifter**

4-input MUX?

**Simple shifter:**

![Diagram of a 4-input MUX with control signals and data inputs]
Arithmetic Logic Unit (ALU)

Example ALU: given inputs $a$ and $b$, and an operation code, produce output.

Operation code:
- 000: AND
- 001: OR
- 010: NOR
- 011: ADD
- 111: SUB

How do we implement this ALU?
Selecting An Operation

2-bit decoder: 2 bit input, 4 bit output

- input: 00, output: 0001
- input: 01, output: 0010
- input: 10, output: 0100
- input: 11, output: 1000
Use decoder to select operation, and use combined add/subtract unit.
ALU: Multiple Bits

Chain ALU *bit slices* to get an $N$ bit ALU:

How can we use a better adder in the ALU?
Overflow Detection

Overflow = result of operation cannot be represented

Unsigned $N$-bit addition:

- Overflow = result requires more than $N$ bits
  $\Rightarrow$ carry-out of MSB is 1

Signed addition:

- Adding two positive numbers
- Adding two negative numbers

Overflow $\equiv$ carry-in to MSB $\neq$ carry-out of MSB
Comparison

When is $a < b$?

- $a < b \equiv a - b < 0$
- Subtract $b$ from $a$, check sign of result
- Sign bit is MSB

When is $a = b$?

- $a = b \equiv a - b = 0$
- Subtract $b$ from $a$, check if all bits are zero
- Use NOR gate