

0110

Arithmetic

ENGR 3410 - Computer Architecture
Fall 2010

Computer Arithmetic

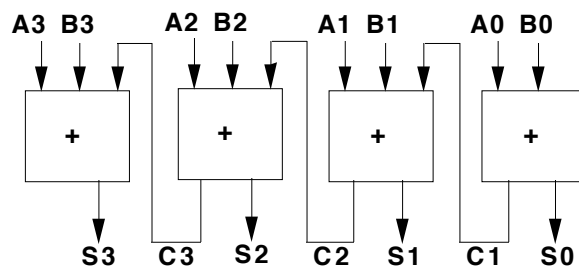
- Readings: Chapter 3
- Review binary numbers, 2's complement
- Develop Arithmetic Logic Units (ALUs) to perform CPU functions.
- Introduce multiplication, division, floating point.

Full Adder

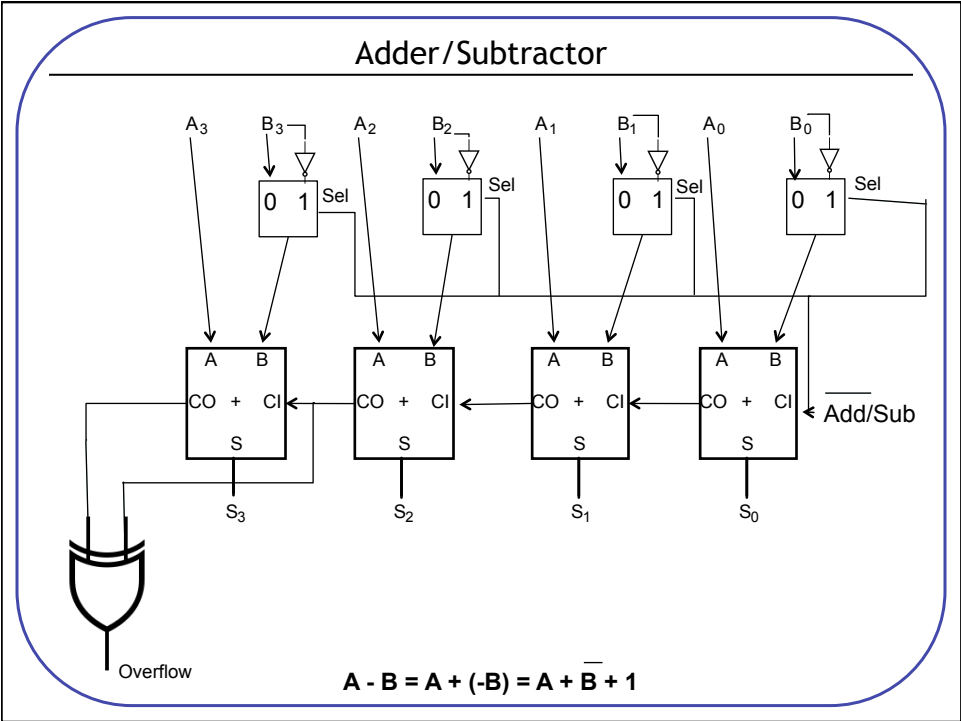
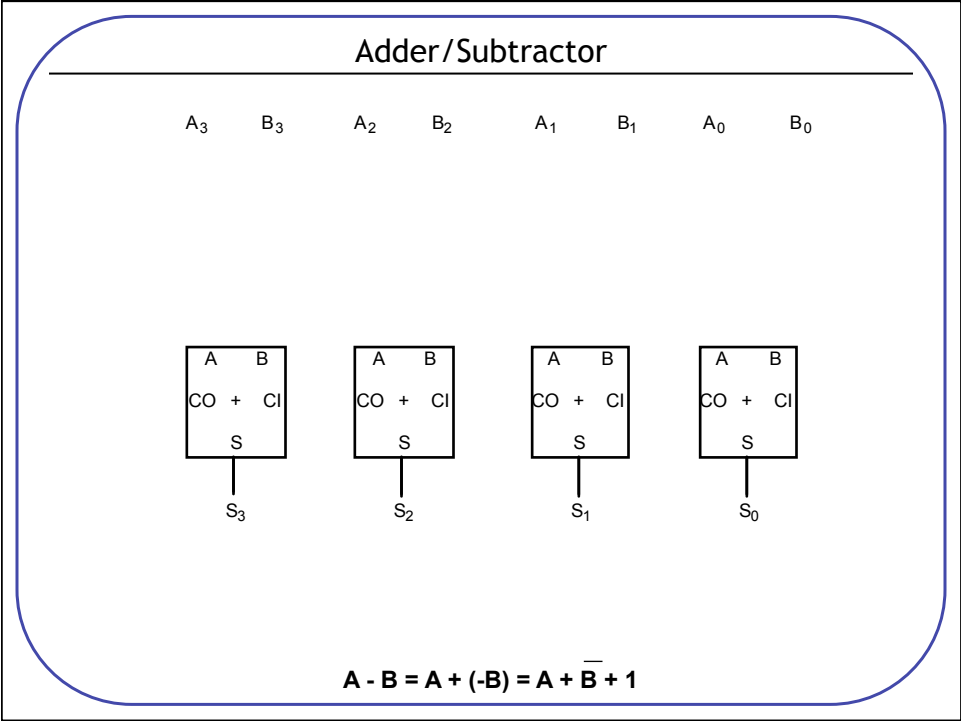
A	B	CI	CO	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

Multi-Bit Addition

Cascaded Multi-bit Adder



usually interested in adding more than two bits
this motivates the need for the full adder



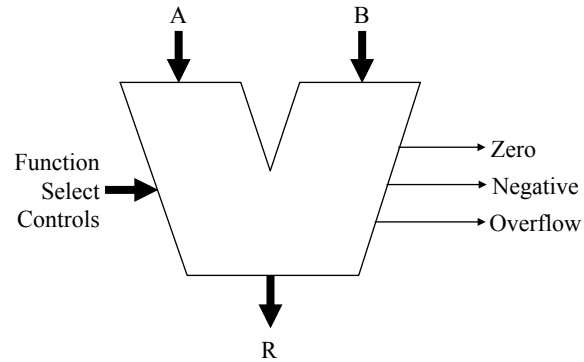
ALU: Arithmetic Logic Unit

- Computes arithmetic & logic functions based on controls

Add, subtract

XOR, AND, NAND, OR, NOR

==, <, overflow, ...

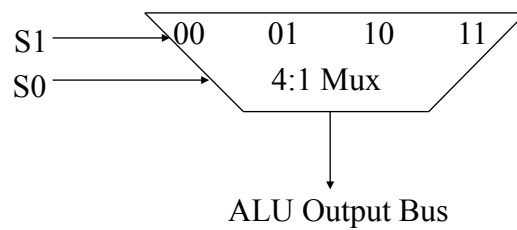


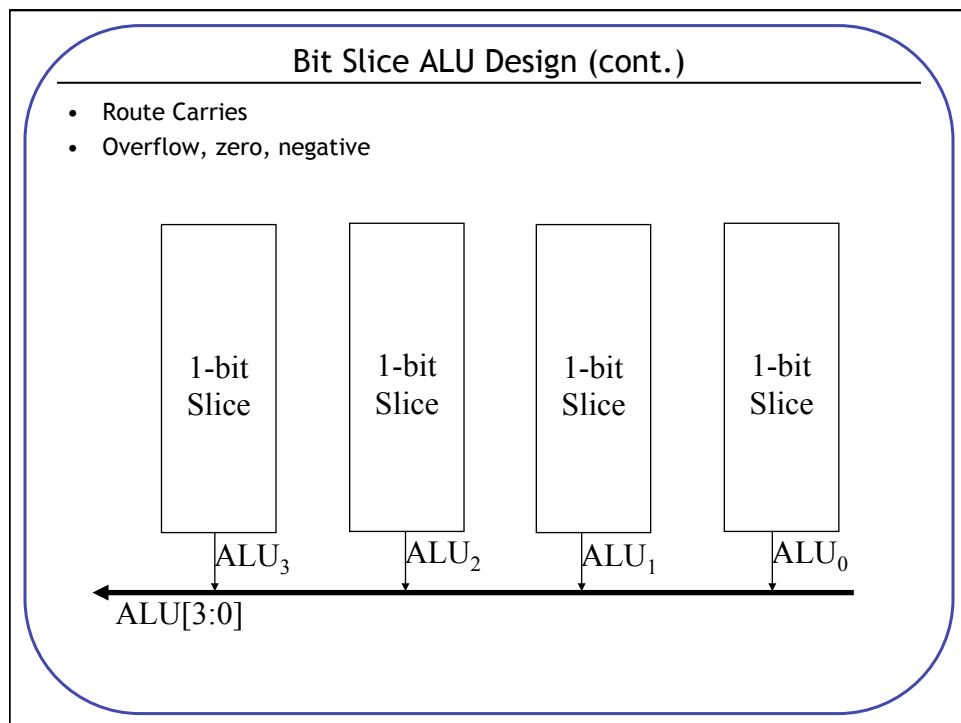
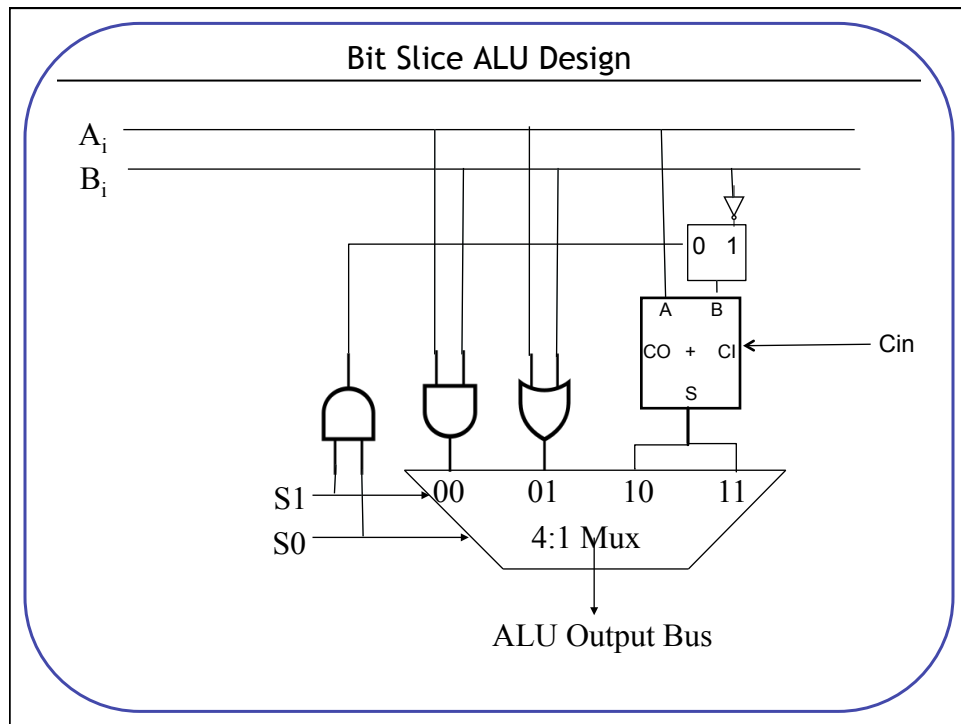
Bit Slice ALU Design

- Add, Subtract, AND, OR

A_i _____

B_i _____





SLT

- Set less than: if $(A < B)$ then $R = 1$, else $R = 0$
 - How do we know if $(A < B)$
 - Interaction w/overflow

Shifter

- Support shift operations: $(A \ll 01101)$
- Optional shift by one: $(A \ll b_0)$
- Optional shift by two: $(A \ll b_1)$

Shifter (cont.)

Multiplication

- Example

Multiplicand:	0	1	1	0	6
Multiplier:	0	1	0	1	5

4 partial products

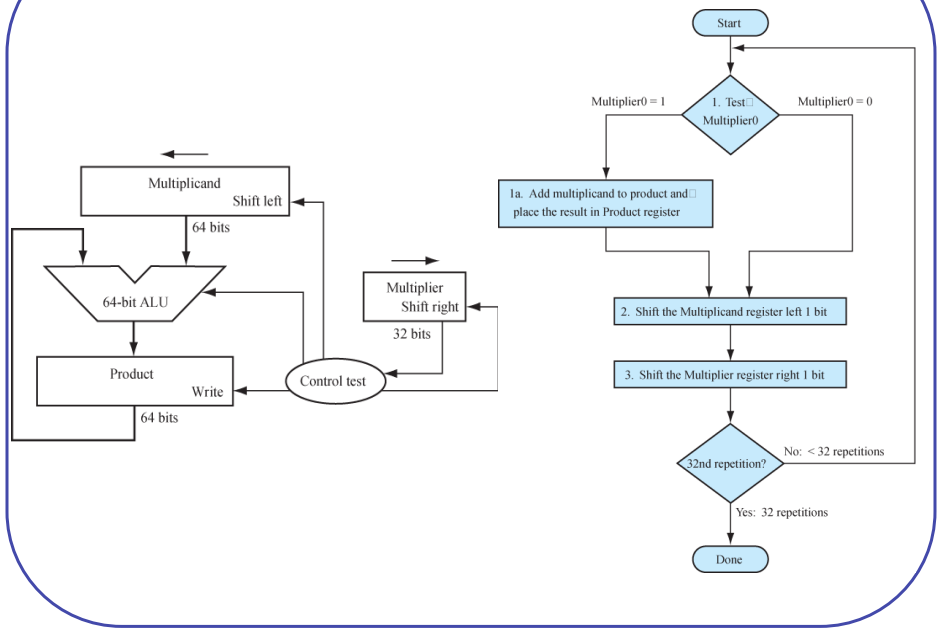
30

Repeat n times:

Compute partial product; shift; add

NOTE: Each bit of partial products is just an AND operation

Sequential Multipliers



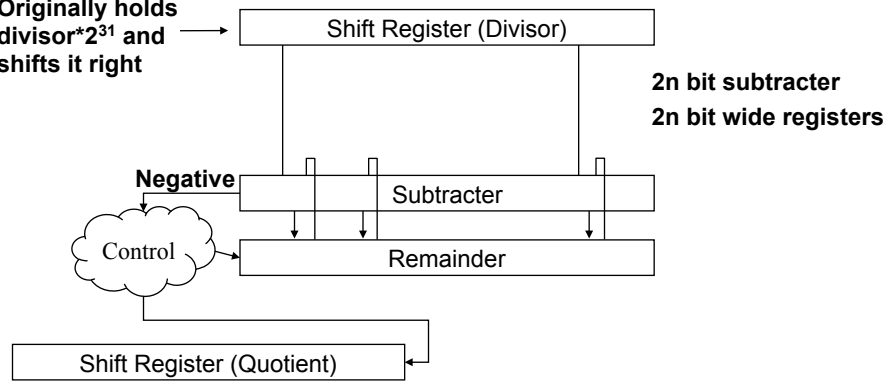
Division

- Example

Divisor: 0010 (2) **0 1 1 0 (6) Dividend**

Sequential Dividers

Originally holds
divisor* 2^{31} and
shifts it right



Remainder

Alternatively:
Shift remainder register to left
Use only n-bit subtractor

Floating Point

Want to represent numbers outside $2^{31}-1 \dots -2^{31}$

Ideal: -Scientific Notation

(+/-)Significand*Base^{Exponent} $5.439 * 10^{12}$ $1.010010 * 2^{100101}$

Multiplication:

$(5.1 * 10^{12}) * (-2.0 * 10^{-3})$

Sign:

Exponent:

Significand:

Floating Point Addition

Addition: $(5.38 * 10^5) + (4.99 * 10^5)$

$(9.99 * 10^4) + (-1.0 * 10^5)$

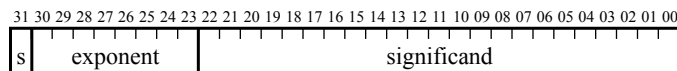
Sign:

Exponent:

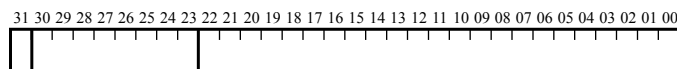
Significand:

Floating Point Representation

- Floating Point (Float) = $(-1)^s * (1.\text{significand}) * 2^{(\text{exponent}-127)}$

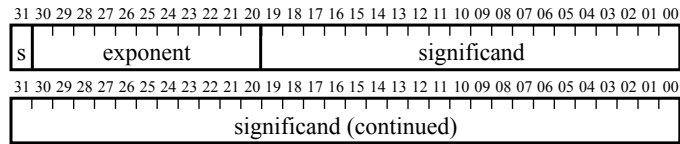


- 0.75 in Floating Point?



Double Precision Representation

- Double Precision (double) = $(-1)^s * (1.\text{significand}) * 2^{(\text{exponent}-1023)}$



- 0.75 in Double Precision?

