RISC V Architecture

Prof. H R Vanamala
Department of Electronics and Communication Engg.

RISC V ARCHITECTURE

UNIT 4: Arithmetic for Computers

Prof. H R Vanamala

Department of Electronics and Communication Engineering

Topics covered:

- Introduction
- Addition and Subtraction
- Multiplication
- Division
- Floating Point
- Parallelism and Computer Arithmetic: Sub word parallelism
  Real Stuff
- Streaming SIMD Extensions
- Advanced Vector Extensions in x86
- Going Faster: Sub word parallelism
- Going Faster: Matrix Multiply
- Fallacies and Pitfalls

References:

Book Type	Author & Title	Edition	Publisher	Year
Text book	David A. Patterson, John L. Hennessy, Computer Organization and Design RISC V edition The Hardware and Software Interface	Second	Elsevier	2021
Reference book	Sarah Harris, David Harris, Digital Design and Computer Architecture, RISC-V Edition	RISC V	Elsevier	2022

Unit 4: Arithmetic for Computers

Introduction:

Integers can be represented either in decimal or binary form.

- What about fractions and other real numbers?
- What happens if an operation creates a number bigger than can be represented?
- How does hardware really multiply or divide numbers?

We will study:

- Representation of real numbers, arithmetic algorithms, hardware that follows these algorithms and the implications of all this for instruction sets.
- Use this knowledge to make arithmetic-intensive programs go much faster.

Addition and Subtraction

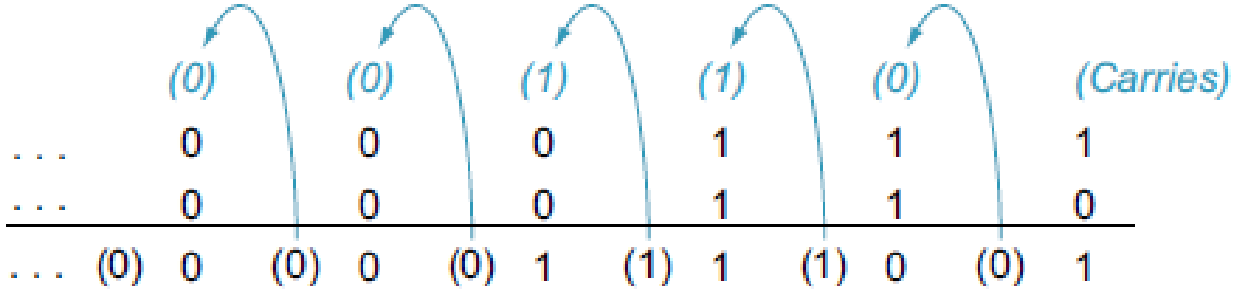
Addition: Digits are added bit by bit from right to left, with carries passed to the next digit to the left.

Subtraction: Uses addition, the appropriate operand is simply negated before being added.

Binary Addition and Subtraction

Let's try adding 6_{ten} to 7_{ten} in binary and then subtracting 6_{ten} from 7_{ten} in binary.

$$\begin{array}{rll} & 00000000\ 00000000\ 00000000\ 00000111_{two} & = 7_{ten} \\ + & 00000000\ 00000000\ 00000000\ 00000110_{two} & = 6_{ten} \\ \hline = & 00000000\ 00000000\ 00000000\ 00001101_{two} & = 13_{ten} \end{array}$$



Binary Subtraction:

Subtracting 6_{ten} from 7_{ten} can be done directly:

$$\begin{array}{rl} & 00000000\ 00000000\ 00000000\ 00000111_{two} = 7_{ten} \\ - & 00000000\ 00000000\ 00000000\ 00000110_{two} = 6_{ten} \\ \hline = & 00000000\ 00000000\ 00000000\ 00000001_{two} = 1_{ten} \end{array}$$

or via addition using the two's complement representation of −6:

$$\begin{array}{rl} & 00000000\ 00000000\ 00000000\ 00000111_{two} = 7_{ten} \\ + & 11111111\ 11111111\ 11111111\ 11111010_{two} = -6_{ten} \\ \hline = & 00000000\ 00000000\ 00000000\ 00000001_{two} = 1_{ten} \end{array}$$

Overflow Conditions :

Adding or subtracting two 32-bit numbers can yield a result that needs 33 bits t
be fully expressed.

Operation	Operand A	Operand B	Result indicating overflow
$A + B$	≥ 0	≥ 0	< 0
$A + B$	< 0	< 0	≥ 0
$A - B$	≥ 0	< 0	< 0
$A - B$	< 0	≥ 0	≥ 0

Overflow Conditions :

Overflow occurs when adding two positive numbers and the sum is negative, or vice versa. This means a carry out occurred into the sign bit.

Overflow occurs in subtraction when we subtract a negative number from a
positive number and get a negative result, or when we subtract a positive number
from a negative number and get a positive result. This means a borrow occurred from the sign bit.

Unsigned Numbers??

Unsigned integers are commonly used for memory addresses where overflows are ignored.
Compiler checks using a branch instruction.
Addition has overflowed if the sum is less than either of the addends and subtraction has overflowed if the difference is greater than the minuend.

Extension:

Saturated arithmetic:

When a calculation overflows, the result is set to the largest positive number or the most negative number.

Ex: Instructions for Multimedia applications

But..

In two's complement arithmetic, modulo calculation is used.

The speed of addition depends on how quickly the carry into the high order bits is computed.
Many fast adders like carry look ahead adders to anticipate carry bits

Worst-case scenario is a function of the $\log_2 n$
(n - no. of bits in the adder)

THANK YOU

Vanamala H R

Department of Electronics and Communication