Arithmetics

Arithmetics are addition, subtraction, multiplication and fraction.

- Integers are represented as binary vectors.
  - Suppose each word consists of 32 bits label 0..31
  - 31 is the **MSB(**most significant bit)
  - 0 is the **LSB(**least …..)
  - Value of the binary vector interpreted as unsigned integer is

    v(B) = $\sum_{n=0}^{n=N-1} b_{n-1}2^{n-1}$

Number representation

- We need to represent both positive and negative integers.
- Three schemes are available for representing both positive and negative integers:
  - Sign and magnitude.
  - 1's complement
  - 2's complement
- All schemes use the **MSB** to carry the sign information
  - if **MSB** = 0, bit vector represents a positive integer.
  - if **MSB** = 1, bit vector represents a negative integer.
- Sign and Magnitude
  - Lower N-1 bits represent the magnitude of the integer
  - **MSB** is set to 0 or 1 to indicate positive or negative.
- 1's complement
  - construct the corresponding positive integer (MSB = 0)
  - Bitwise complement this integer
- 2's complement
  - construct the 1's complement negative integer
  - add 1 to this.

Unsigned: $0 < V(b) < 2^{N-1}$

Sign and magnitude: $-2^{N-1} - 1 < V(b) < 2^{N-1} - 1$

0 has both positive and negative representation

One's complement:: $-2^{N-1} - 1 < V(b) < 2^{N-1} - 1$

0 has both positive and negative representation

Two's complement:: $-2^{N-1} < V(b) < 2^{N-1} - 1$

0 has a single representation, easier to add/subtract.

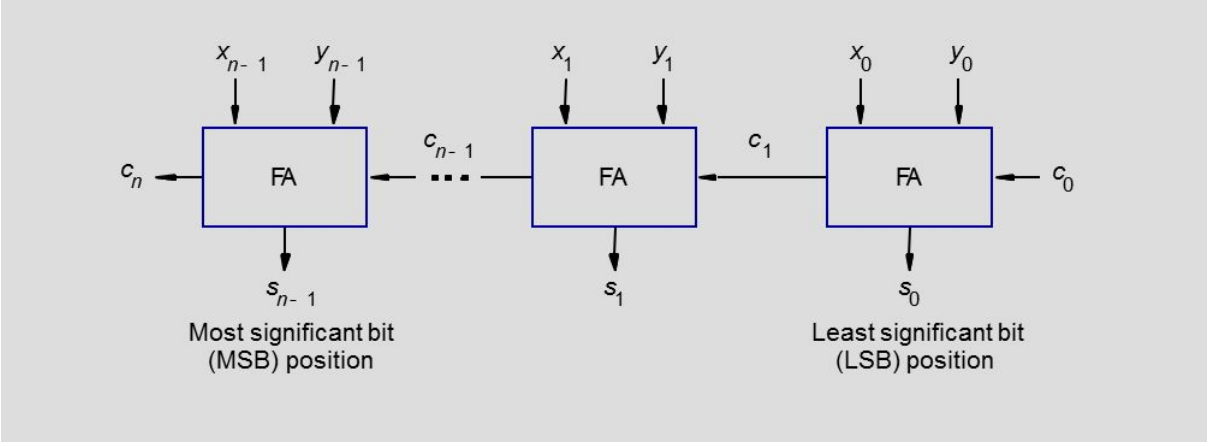
- Addition and subtraction is easiest executed using **2's complement** for signed numbers.
- Rules using 2's complement for addition/subtraction
    - Add their n-bit representations
    - Ignore the carry out from MSB position
    - Sum is the algebraically correct value in the 2's complement representation as long as the answer is in the range -2^n-1 -> +2^n-1 -1
- To subtract two numbers X and Y(x-Y)
    - form 2's complement of Y
    - add it to X using rule 1

Overflow in integer arithmetic

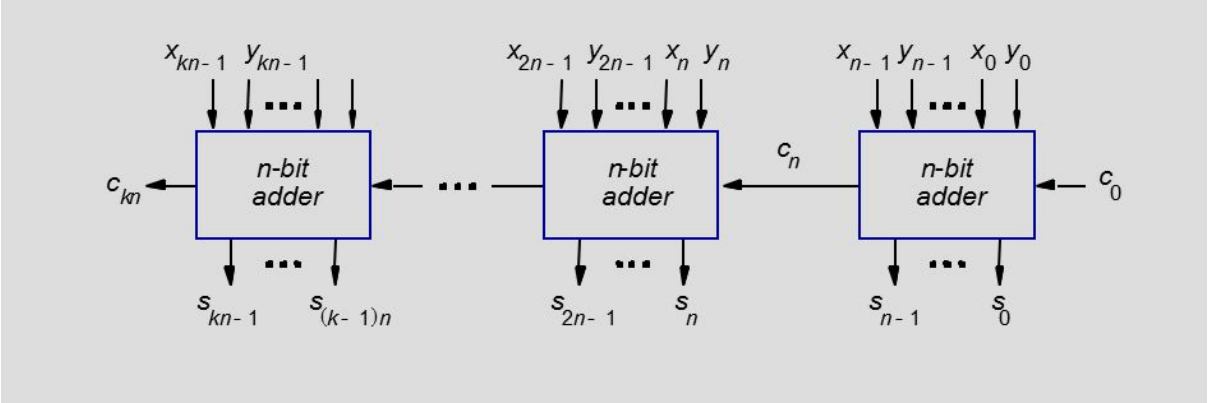
- When the result of an arithmetic operation is outside the representable range and **arithmetic overflow** has occurred.
    - Range is -2^n-1 -> +2^n-1 -1 for n bit vector
- When adding usigned numbers, carry-out from the **MSB** position serves as the **overflow indicator.**
- When **adding signed numbers** this does not work.
- **Overflow** occurs when both the numbers have the same sign.
    - Addition of numbers with different signs cannot cause an **overflow.**
- **Carry-out** signal from the **MSB** (sign-bit) position is not a sufficient indicator of overflow when adding signed numbers.
- Detect overflow when adding **X & Y.**
    - If the signs of **X & Y** are the same but the **sign** of the result differ a **overflow has occurred.**

N-Bit adder

C is the carry , s is the sum X and Y are the operands.
(n-bit ripple carry adder)



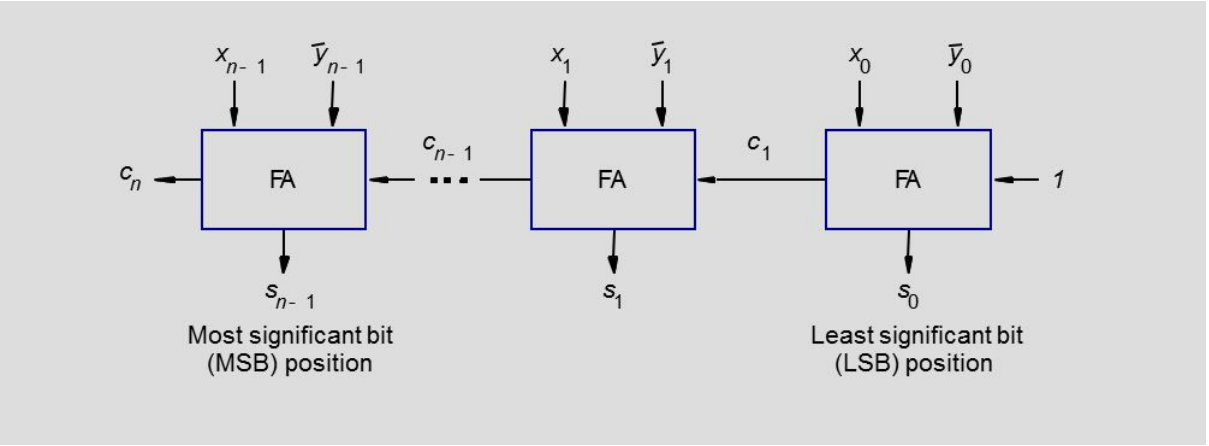
KN-bit adder



A cascading of blocks.

N-bit Subtractor.

- **X-Y** is equivalent to adding 2's complement of **Y to X.**
- **2's complement** is equivalent to **1's complement +1.**
- X-Y = X + (2's complement of Y) +1

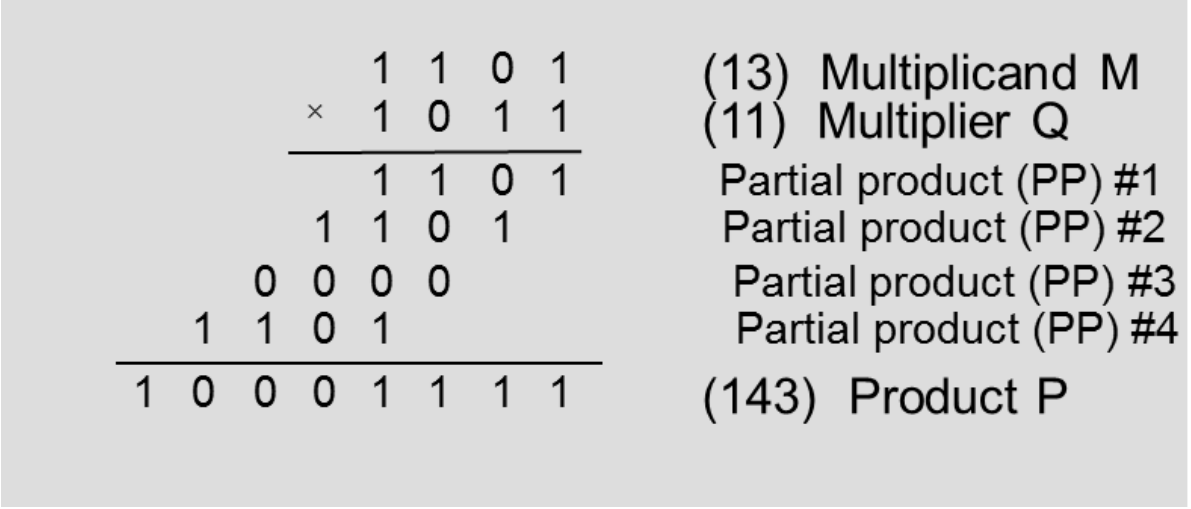


- In a normal adder the c0 is equal to 0 but in the case of subtractor the c0 is 1.

Overflow

- Overflow can only happen when two operands has the same sign
- If the result sign differ from the operands overflow occurred.

Multiplication of unsigned numbers.



Fraction slide 22 - 28 LeactureArithmetic förstår inte det så vet inte vad jag ksa skriva.