Digital signal processors

Lecturer: Krébesz, Tamás

General computing applications

- Data manipulation
  - Word processing
  - Database management
  - Data movement(A->B)
  - Value testing
    - Execution time is not critical, not predictable
- Math calculations
  - Digital signal processing
  - Motion control
  - Simulations
  - Real-time signal proc.
  - Addition (C=A+B)
  - Multiplication (C=A*B)
    - Time to execute is critical and predictable

Processing units

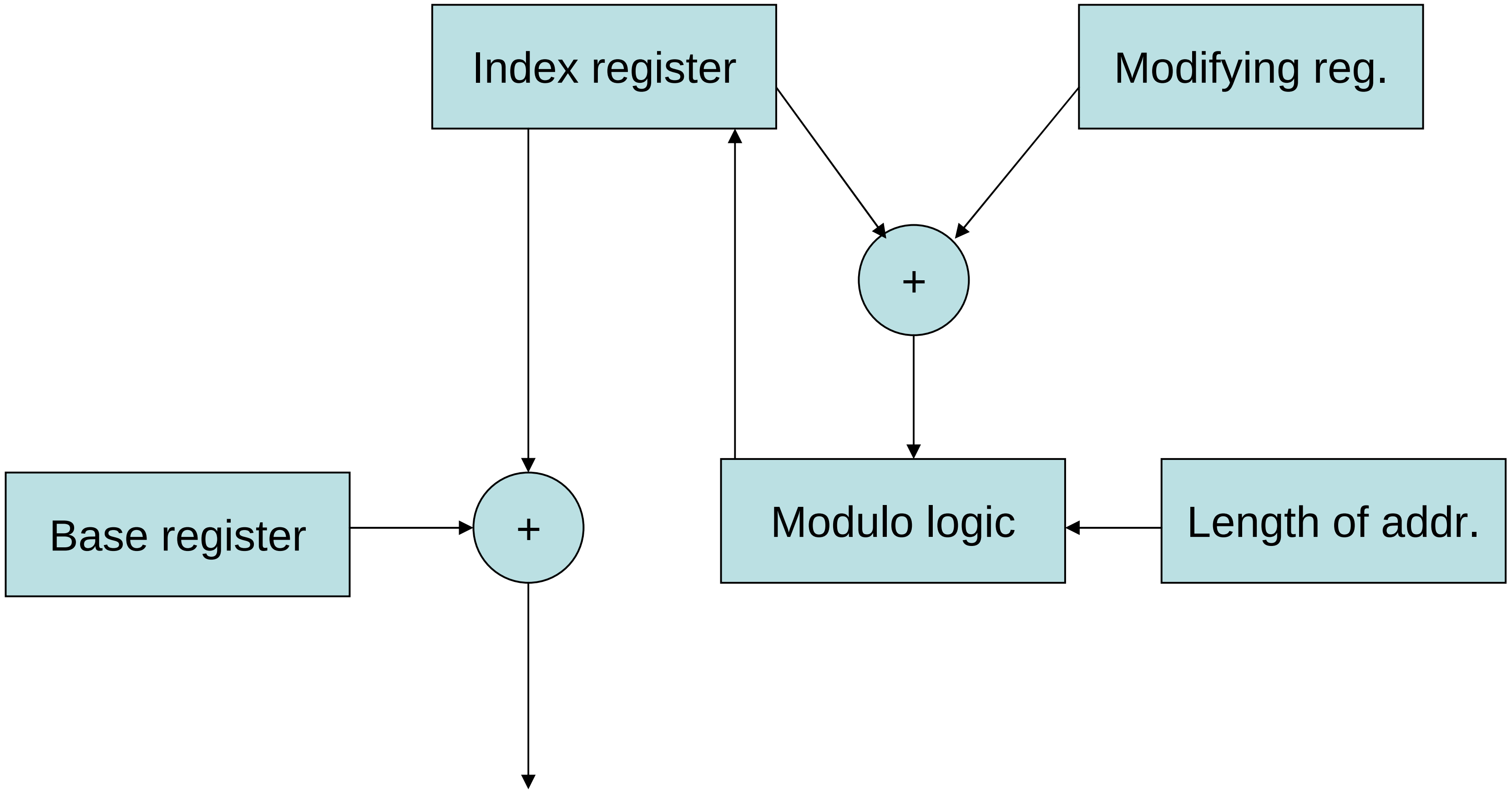
- uC
  - Best in I/O capabilities and control
  - Speed less important
  - Haeward architecture
  - RISC operations
- uP
  - Neumann architecture
  - General applications
  - CISC (RISC) operations
- DSP
  - Optimized for fast repetitive math for real-time processing
  - RISC/VLIW

DSP operations

- Application areas
  - Telecommunications
  - Measurement data acquisition and processing
  - Control engineering
  - Medical circuits
  - Audio processing
  - Data compression, coding (MP3)
  - Graphical processing

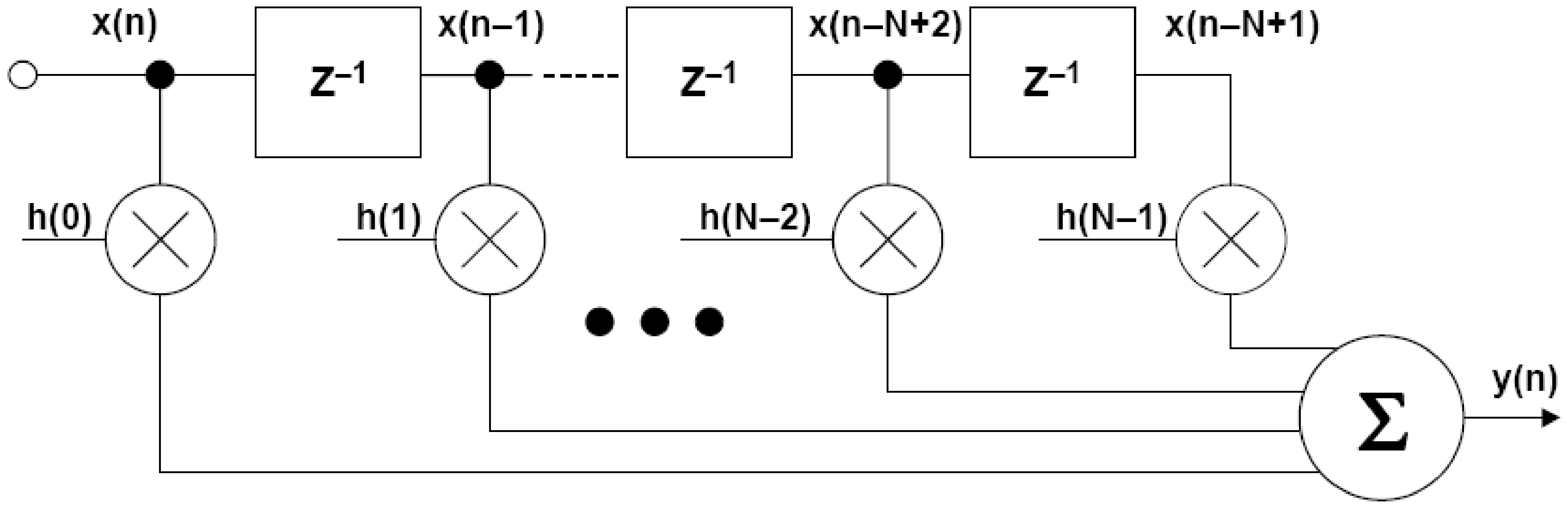
- Common DSP operations
  - Digital filtering, convolution
  - Decimation, interpolation
  - Adaptive filters
  - Transformation (DFT, FFT, DCT,…)
  - Signal generation
  - Modulation/demodulation

- Independent addressing arithmetic in DSP



Digital filtering (IIR, FIR, Multirate)

- FIR (Finite impulse response)
  - Uses the most fundamental DSP operation: sum of products->MAC multiply and accumulate
  - Precisely reproductable
  - Linear phase
  - Always stable
  - Steep cut-off



- $y(n) = h(n) * x(n) = \sum_{k=0}^{N-1} h(k)\, x(n-k)$
- $*$ = Symbol for Convolution
- Requires N multiply-accumulates for each output

– Requirements

- Fast arithmetic – fast MAC
  - – Basic requirement
  - – Requires high bandwidth
- Extended precision
  - – 16-bit word * 16-bit word = 32-bit word
  - – ADSPP21xx DSP
    - » 16-bit fix point core architecture (other may have floating point)
    - » 40-bit accumulator ->high degree of overflow protection
- Harvard architecture
  - – Dual operand fetch -> convolution, MAC

- MAC instruction
  - MR=MR+MX0*MY0
    - MR: accumulator register
    - MX0, MY0: operand registers
- HW supported loop
  - DO convolution UNTIL CE
- HW suppoerted circular buffer -> memory part
  - Buffer content circulated (filter coefficients)
  - When last buffer location reached memory pointer reset to the buffer beginning

- FIR filter algorithm

1. Obtain sample from ADC (typically interrupt driven)
2. Move sample into input signal's circular buffer
3. Update the pointer for the input signal's circular buffer
4. Zero the accumulator
5. Implement filter (control the loop through each of the coefficients)
    6. Fetch the coefficient from the coefficient's circular buffer
    7. Update the pointer for the coefficient's circular buffer
    8. Fetch the sample from the input signal's circular buffer
    9. Update the pointer for the input signal's circular buffer
    10. Multiply the coefficient by the sample
    11. Add the product to the accumulator
12. Move the filtered sample to the DAC

- Fast Fourier Transform (FFT)
  - FFT is an algorithm for efficiently calculating Discrete Fourier Transform (DFT)

Frequency Domain ←← DFT ←← Time Domain

$$X(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}} = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\left[\cos\frac{2\pi nk}{N} - j\sin\frac{2\pi nk}{N}\right]$$

$$\boxed{W_N = e^{\frac{-j2\pi}{N}}} \qquad = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\, W_N^{nk} \quad , \quad 0 \le k \le N-1$$

Time Domain ←← INVERSE DFT ←← Frequency Domain

$$x(n) = \sum_{k=0}^{N-1} X(k)\, e^{\frac{j2\pi nk}{N}} = \sum_{k=0}^{N-1} X(k)\left[\cos\frac{2\pi nk}{N} + j\sin\frac{2\pi nk}{N}\right]$$

$$= \sum_{k=0}^{N-1} X(k)\, W_N^{-nk} \quad , \quad 0 \le n \le N-1$$

– Example: 8-point DFT (Number of taps=8)

$$X(k) = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\, e^{\frac{-j2\pi nk}{N}} = \frac{1}{N}\sum_{n=0}^{N-1} x(n)\, W_N^{nk}$$

$$\boxed{W_N = e^{\frac{-j2\pi}{N}}}$$

$X(0) =$	$x(0)W_8^0$	$+ x(1)W_8^0$	$+ x(2)W_8^0$	$+ x(3)W_8^0$	$+ x(4)W_8^0$	$+ x(5)W_8^0$	$+ x(6)W_8^0$	$+ x(7)W_8^0$
$X(1) =$	$x(0)W_8^0$	$+ x(1)W_8^1$	$+ x(2)W_8^2$	$+ x(3)W_8^3$	$+ x(4)W_8^4$	$+ x(5)W_8^5$	$+ x(6)W_8^6$	$+ x(7)W_8^7$
$X(2) =$	$x(0)W_8^0$	$+ x(1)W_8^2$	$+ x(2)W_8^4$	$+ x(3)W_8^6$	$+ x(4)W_8^8$	$+ x(5)W_8^{10}$	$+ x(6)W_8^{12}$	$+ x(7)W_8^{14}$
$X(3) =$	$x(0)W_8^0$	$+ x(1)W_8^3$	$+ x(2)W_8^6$	$+ x(3)W_8^9$	$+ x(4)W_8^{12}$	$+ x(5)W_8^{15}$	$+ x(6)W_8^{18}$	$+ x(7)W_8^{21}$
$X(4) =$	$x(0)W_8^0$	$+ x(1)W_8^4$	$+ x(2)W_8^8$	$+ x(3)W_8^{12}$	$+ x(4)W_8^{16}$	$+ x(5)W_8^{20}$	$+ x(6)W_8^{24}$	$+ x(7)W_8^{28}$
$X(5) =$	$x(0)W_8^0$	$+ x(1)W_8^5$	$+ x(2)W_8^{10}$	$+ x(3)W_8^{15}$	$+ x(4)W_8^{20}$	$+ x(5)W_8^{25}$	$+ x(6)W_8^{30}$	$+ x(7)W_8^{35}$
$X(6) =$	$x(0)W_8^0$	$+ x(1)W_8^6$	$+ x(2)W_8^{12}$	$+ x(3)W_8^{18}$	$+ x(4)W_8^{24}$	$+ x(5)W_8^{30}$	$+ x(6)W_8^{36}$	$+ x(7)W_8^{42}$
$X(7) =$	$x(0)W_8^0$	$+ x(1)W_8^7$	$+ x(2)W_8^{14}$	$+ x(3)W_8^{21}$	$+ x(4)W_8^{28}$	$+ x(5)W_8^{35}$	$+ x(6)W_8^{42}$	$+ x(7)W_8^{49}$

N^2 Complex Multiplications

$\frac{1}{N}$ Scaling Factor Omitted

– Exploiting symmetry and periodicity

Symmetry: $W_N^{r+N/2} = -W_N^r$, Periodicity: $W_N^{r+N} = W_N^r$

N = 8

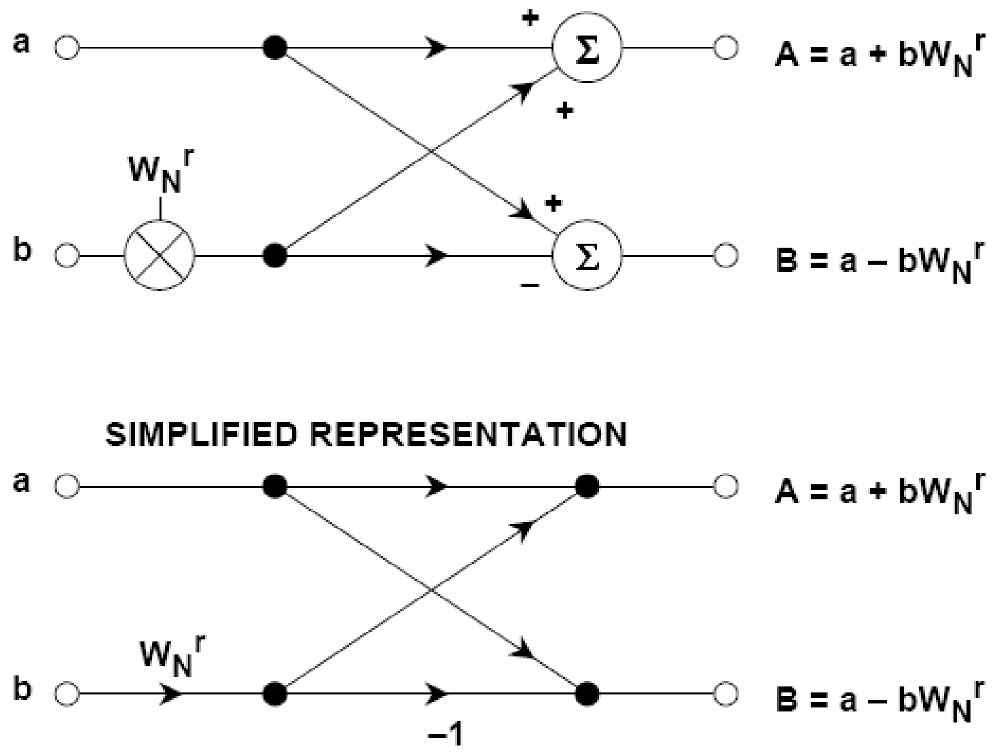
$$W_8^4 = W_8^{0+4} = -W_8^0 = -1$$
$$W_8^5 = W_8^{1+4} = -W_8^1$$
$$W_8^6 = W_8^{2+4} = -W_8^2$$
$$W_8^7 = W_8^{3+4} = -W_8^3$$
$$W_8^8 = W_8^{0+8} = +W_8^0 = +1$$
$$W_8^9 = W_8^{1+8} = +W_8^1$$
$$W_8^{10} = W_8^{2+8} = +W_8^2$$
$$W_8^{11} = W_8^{3+8} = +W_8^3$$

● ● ●

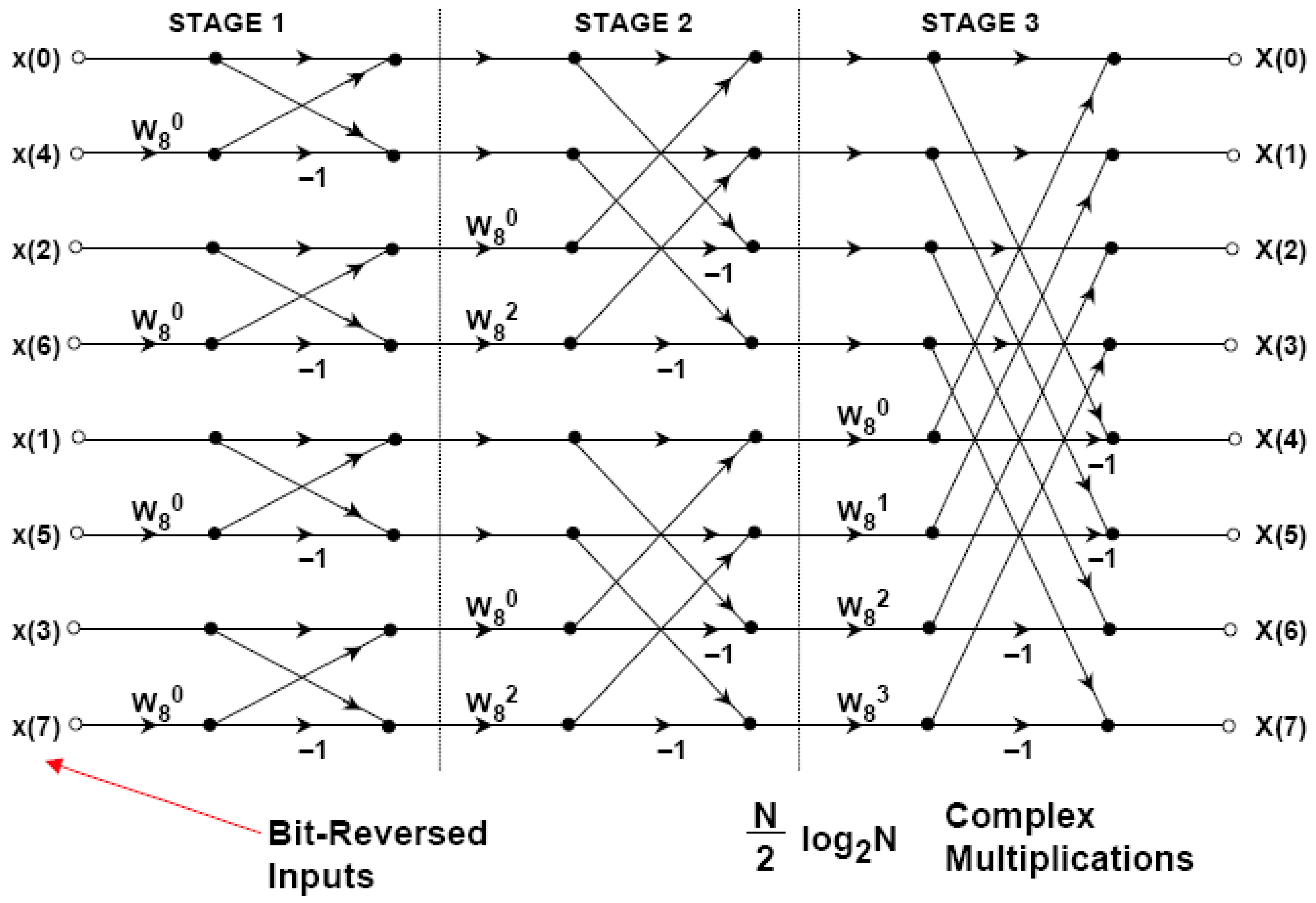
● ● ●

● ● ●

– Butterfly computation in FFT



- Example: N=8

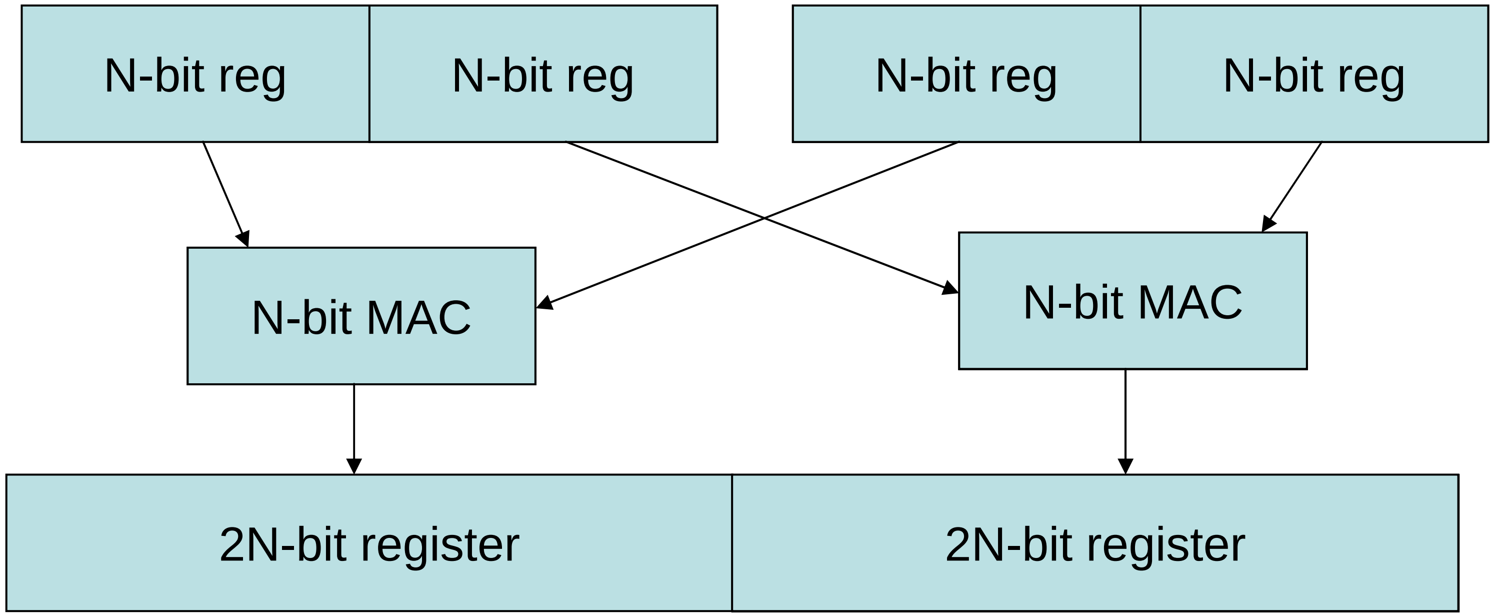

STAGE 1
STAGE 2
STAGE 3
x(0)
x(4)
x(2)
x(6)
x(1)
x(5)
x(3)
x(7)
X(0)
X(1)
X(2)
X(3)
X(4)
X(5)
X(6)
X(7)
W_8^0
W_8^1
W_8^2
W_8^3
−1
Bit-Reversed Inputs
$\frac{N}{2}\log_2 N$ Complex Multiplications

– Note: order of input is changed -> bit-reverse algorithm is used for input

New generation of DSP

- Increasing speed?
  - Higher clock frequency
    - Higher power consumption/dissipation
    - Limited by technonlogy
  - Parallel structures: more processors
  - Same function multiple blocks in one core
    - Single instruction multiple data (SIMD)
    - VLIW, superscalar processor

- SIMD architecture
  - Two identical instructions at the same time
  - Dedicated registers, ALU, bus width doubled
  - Faster operation with block of data (filtering, DFT)



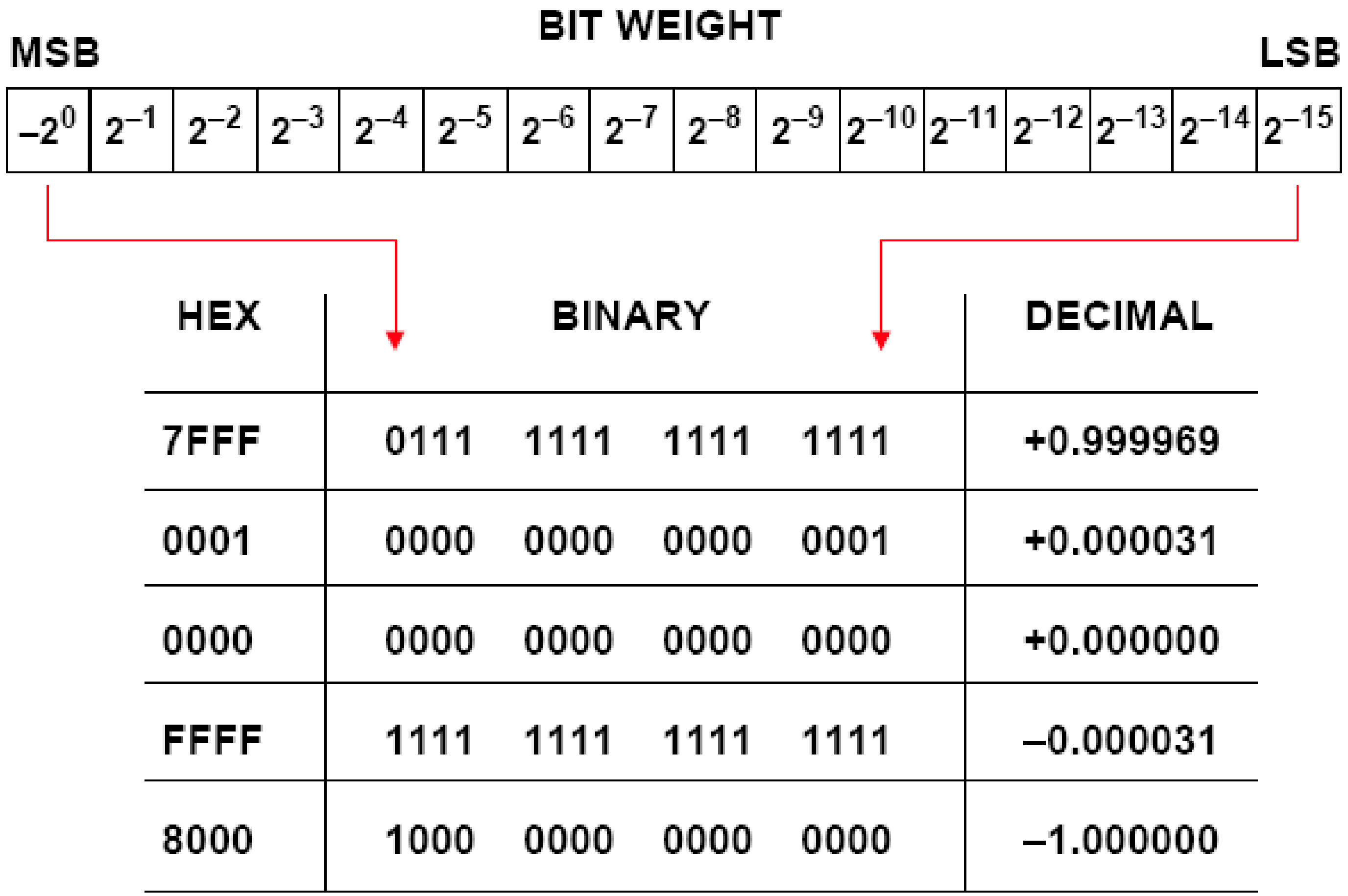
- VLIW DSP
  - One long instruction contains many short RISC-type ones
    - Compile time scheduling
      - » Long instruction built up during compilation
  - Program memory
    - 128-256-bit wide
    - 4-8 short instructions/cycle
- Superscalar architecture
  - Runtime scheduling
    - Instructions executed parallel in runtime

- Embedded DSP
  - Offer specialized peripherals to solve a problem
    - High performance ADC
    - Power measurement
    - PWM
    - Data compression
    - Encoding

Data representation

- Fixed point
  - 16-bit: 2^16=65536 possible bit patterns
  - Unsigned integer: 0 -> 65536
  - Signed integer: two's complement for negative: -32768 -> 32767
  - Unsigned fractional: 65536 levels spread uniformly between 0 and 1
  - Signed fractional 65536 levels spread equally between -1 and 1

- 16-bit fixed point arithmetic fractional 1.15 format



BIT WEIGHT

MSB															LSB
-2^{0}	2^{-1}	2^{-2}	2^{-3}	2^{-4}	2^{-5}	2^{-6}	2^{-7}	2^{-8}	2^{-9}	2^{-10}	2^{-11}	2^{-12}	2^{-13}	2^{-14}	2^{-15}

HEX	BINARY	DECIMAL
7FFF	0111 1111 1111 1111	+0.999969
0001	0000 0000 0000 0001	+0.000031
0000	0000 0000 0000 0000	+0.000000
FFFF	1111 1111 1111 1111	–0.000031
8000	1000 0000 0000 0000	–1.000000

- Single precision IEEE-754 32-bit floating point format

$$\text{NUMBER}_{10} = (-1)^S \times \underset{\text{Assumed}}{\underline{1}}.M \times 2^{(E-\overset{\text{Bias}}{127})}$$

	MSB							LSB	MSB						LSB
0/1	2^7	2^6	2^5	2^4	2^3	2^2	2^1	2^0	2^{-1}	2^{-2}	2^{-3}	2^{-4}	• • •	2^{-22}	2^{-23}
SIGN BIT = S	EXPONENT = E 8-BITS								MANTISSA = M 23-BITS						

$$\underbrace{0}_{+}\ \underbrace{00000111}_{7}\ \underbrace{1100\ldots00}_{0.75} \qquad +\underset{\text{Assumed}}{\underline{1}}.75 \times 2^{(7-\overset{\text{Bias}}{127})} = +1.316554 \times 10^{-36}$$

$$\underbrace{1}_{-}\ \underbrace{10000001}_{129}\ \underbrace{0110\ldots00}_{0.375} \qquad -\underset{\text{Assumed}}{\underline{1}}.375 \times 2^{(129-\overset{\text{Bias}}{127})} = -\ 5.500000$$

References

- Analog devices: Mixed signal and DSP design techiques