

Introduction to Digital Signal Processing Systems

Lan-Da Van (范倫達), *Ph. D.*
Department of Computer Science
National Chiao Tung University
Taiwan, R.O.C.
Spring, 2007



ldvan@cs.nctu.edu.tw

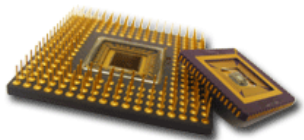


<http://www.cs.nctu.edu.tw/~ldvan/>



Outlines

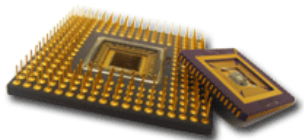
- ◆ *Introduction*
- ◆ DSP Algorithms
- ◆ DSP Applications and CMOS IC's
- ◆ Representations of DSP Algorithms
- ◆ Conclusion
- ◆ References





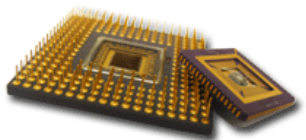
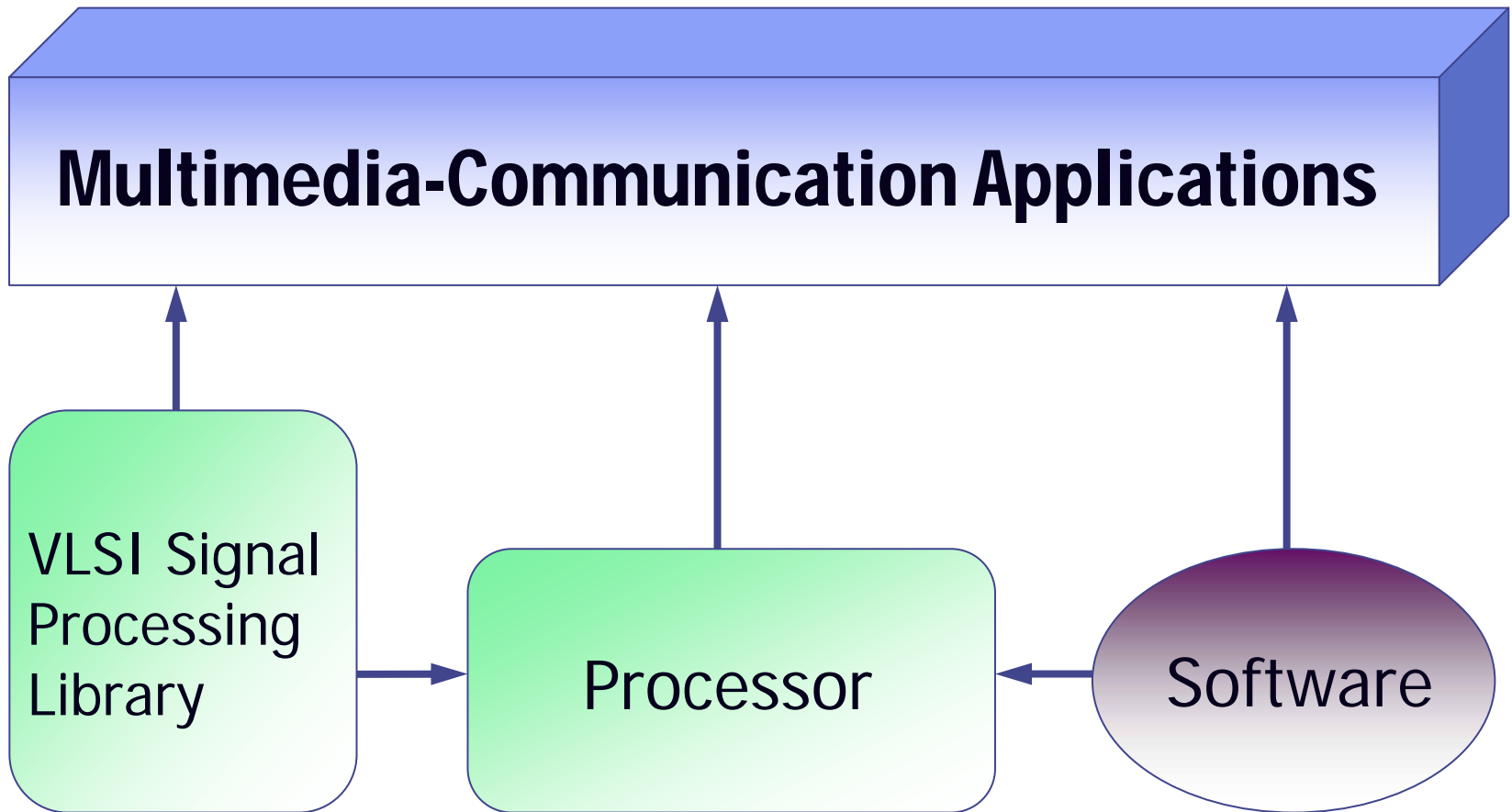
Why Use Digital Signal Processing?

- ◆ Robust to temperature and process variations
- ◆ Controlled better to accuracy
- ◆ Noise/interference tolerances
- ◆ Mathematical representation
- ◆ Programming capability





Common System Configuration





VLSI Signal Processing System Design Spectrum

◆ Computer arithmetic

- Adder
- Multiplier

◆ Digital filter

◆ Adaptive digital filter

- LMS/DLMS (Delay LMS) based
- RLS based

◆ Transform

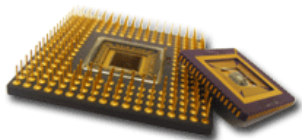
- Multiplier-accumulator based
- Recursive-filter based
- ROM-based: DA, CORDIC
- Butterfly based

◆ Processor

- General purposed processor
- DSP processor
- Reconfigurable computing processor

◆ Non-numerical operation

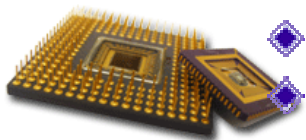
- Error control coding
 - ◆ Viterbi Decoder
 - ◆ Turbo Code
- Polynomial computation
- Dynamic programmable
- Etc..





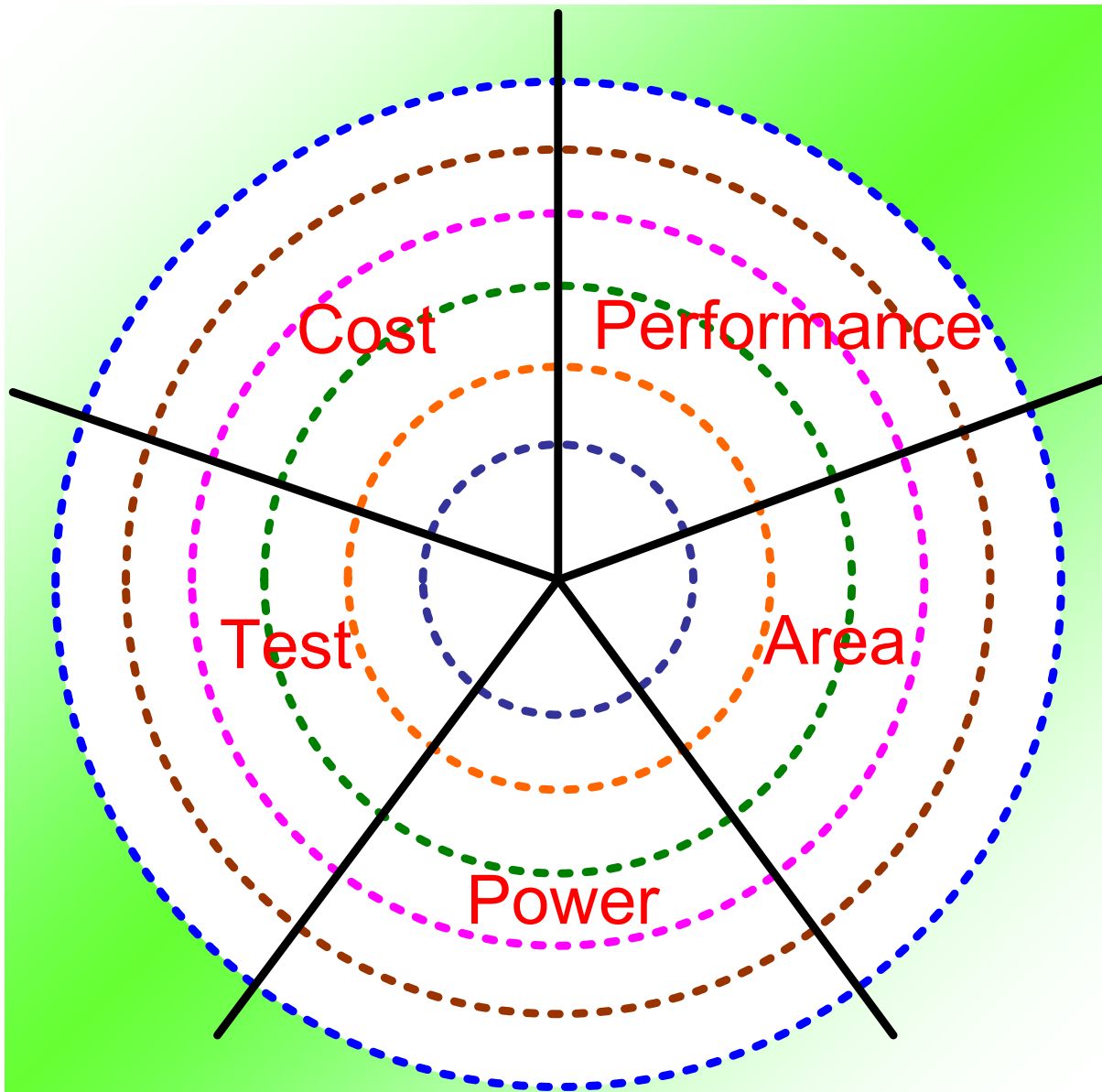
VLSI Signal Processing System Publication Area (But not limited...)

- ◆ *IEEE Journal of Solid-State Circuits*
- ◆ *IEEE Journal on Selected Areas in Communications*
- ◆ *IEEE Micro*
- ◆ *IEEE Trans. on Biomedical Engineering*
- ◆ *IEEE Trans. on Circuits and Systems I: Regular Papers*
- ◆ *IEEE Trans. on Circuits and Systems II: Express Briefs*
- ◆ *IEEE Trans. on Circuits and Systems for Video Technology*
- ◆ *IEEE Trans. on Communications*
- ◆ *IEEE Trans. on Computer-Aided Design of Integrated Circuits*
- ◆ *IEEE Trans. on Computers*
- ◆ *IEEE Trans. on Image Processing*
- ◆ *IEEE Trans. on Information Theory*
- ◆ *IEEE Trans. on Multimedia*
- ◆ *IEEE Trans. on Nanotechnology*
- ◆ *IEEE Trans. on Neural Networks*
- ◆ *IEEE Trans. on Signal Processing*
- ◆ *IEEE Trans. on VLSI Systems*
- ◆ *Proceedings of the IEEE*
- ◆ *The Journal of VLSI Signal Processing Systems*
- ◆ *Elsevier Integration - The VLSI Journal*





VLSI Signal Processing System Design Space

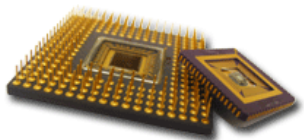


- System Level
- Algorithm Level
- Architecture Level
- Logic Level
- Circuit Level
- Process Level



Outlines

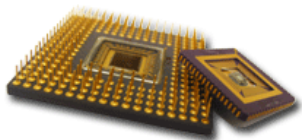
- ◆ Features:
- ◆ *DSP Algorithms*
- ◆ DSP Applications and CMOS IC's
- ◆ Representations of DSP Algorithms





DSP Algorithms

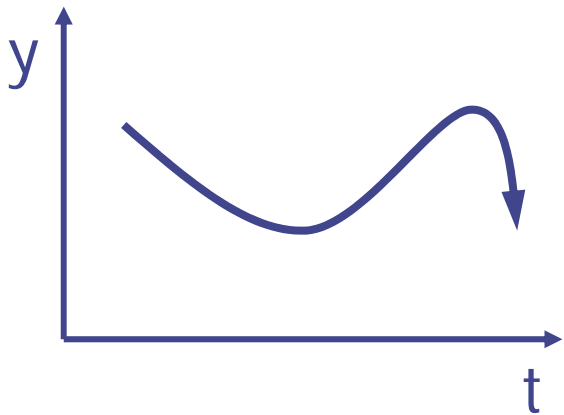
- ◆ Convolution
 - ◆ Correlation
 - ◆ Digital filters
 - ◆ Adaptive filters
 - ◆ Discrete Fourier transform
 - ◆ Source Coding Algorithms
 - Discrete cosine transform
 - Motion estimation
 - Huffman coding
 - Vector quantization
 - ◆ Decimator and expander
 - ◆ Wavelet and filter banks
 - ◆ Viterbi algorithm and dynamic programming
- Algorithm: A set of rules for solving a problem in a finite number of steps.



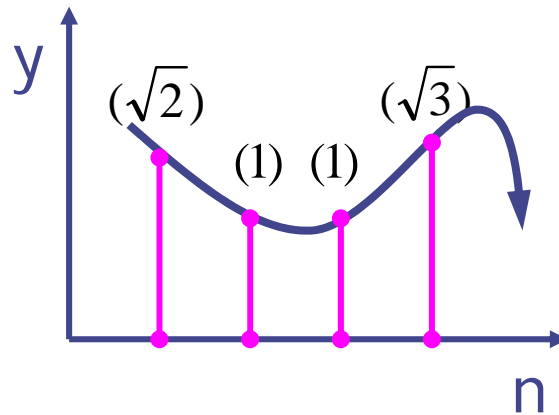


Signals

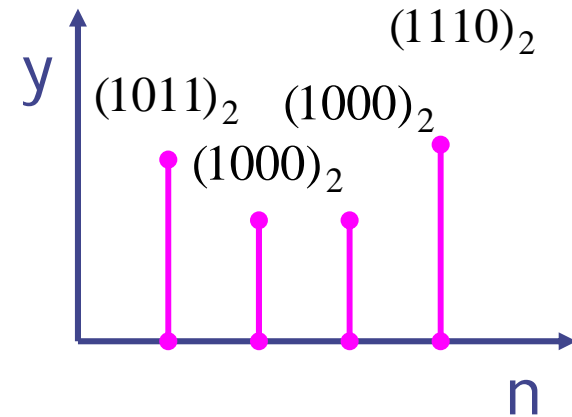
- ◆ Analog signal
 - $t \rightarrow y: y=f(t), y:\mathbb{C}, n:\mathbb{C}$
- ◆ Discrete-time signal
 - $n \rightarrow y: y=f(nT), y:\mathbb{C}, n:\mathbb{Z}$
- ◆ Digital signal
 - $n \rightarrow y: y=D\{f(nT)\}, y:\mathbb{Z}, n:\mathbb{Z}$



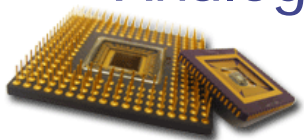
Analog Signal



Discrete-Time Signal



Digital Signal





LTI Systems

◆ Linear systems

- Assume $x_1(n) \rightarrow y_1(n)$ and $x_2(n) \rightarrow y_2(n)$, where “ \rightarrow ” denotes “lead to”. If $ax_1(n) + bx_2(n) \rightarrow ay_1(n) + by_2(n)$, then the systems is referred to as “Linear System.”
- Homogenous and additive properties

◆ Time-invariant (TI) systems

- $x(n-n_0) \rightarrow y(n-n_0)$

◆ LTI systems

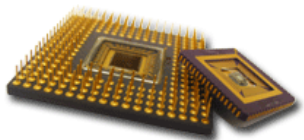
- $y(n) = h(n) * x(n)$

◆ Causal systems

- $y(n_0)$ depends only on $x(n)$, where $n \leq n_0$

◆ Stable systems

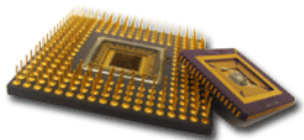
- BIBO





Sampling of Analog Signals

- ◆ Nyquist sampling theorem
 - The analog signal must be band-limited
 - Sample rate must be larger than twice the bandwidth





System-Equation Representation

- ◆ Impulse/unit sample response

$$h(n) = b_0 a_1^n u[n]$$

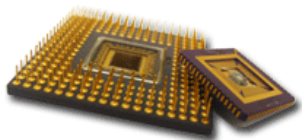
- ◆ Transfer function / frequency response

$$H(z) = \frac{Y(z)}{X(z)} = \frac{b_0}{1 - a_1 z^{-1}}$$

- ◆ Difference equations

$$y(n) = a_1 y(n-1) + b_0 x(n)$$

- ◆ State equations





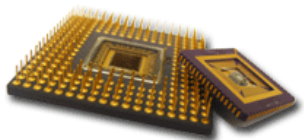
Convolution & Correlation

◆ Convolution

$$\begin{aligned}y(n) &= x(n) * h(n) = \sum_{k=-\infty}^{\infty} x(k)h(n-k) \\ &= \sum_{k=-\infty}^{\infty} h(k)x(n-k)\end{aligned}$$

◆ Correlation

$$\begin{aligned}y(n) &= \sum_{k=-\infty}^{\infty} a(k)x(n+k) \\ &= \sum_{k=-\infty}^{\infty} a(-k)x(n-k) = a(-n) * x(n)\end{aligned}$$





Linear Phase FIR Digital Filters

- ◆ Digital filters are an important class of LTI systems.
- ◆ Linear phase FIR filter

$$h(n) = h(M - n)$$

$$h(0) = h(6) = b_0$$

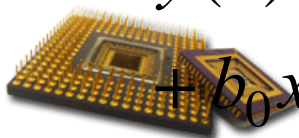
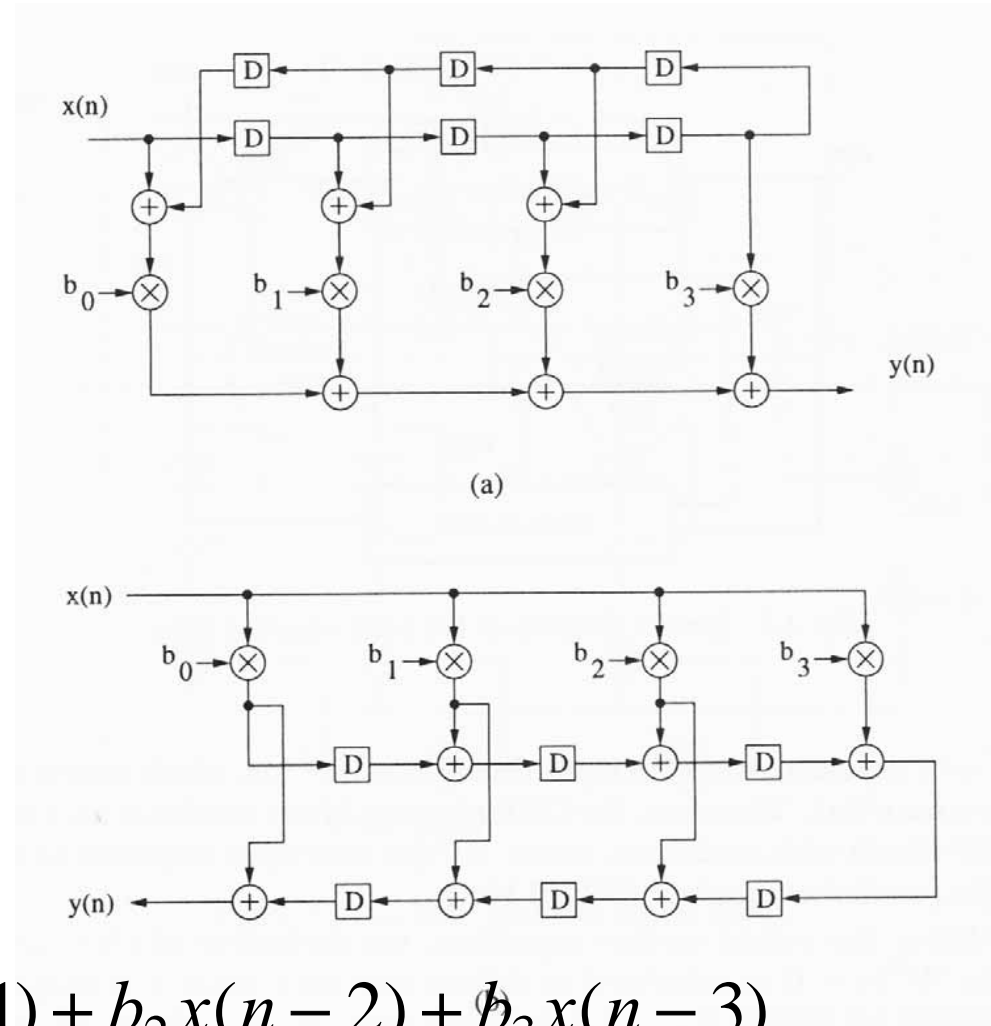
$$h(1) = h(5) = b_1$$

$$h(2) = h(4) = b_2$$

$$h(3) = b_3$$

$$y(n) = b_0x(n) + b_1x(n-1) + b_2x(n-2) + b_3x(n-3)$$

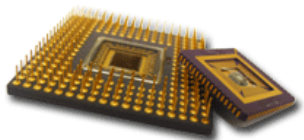
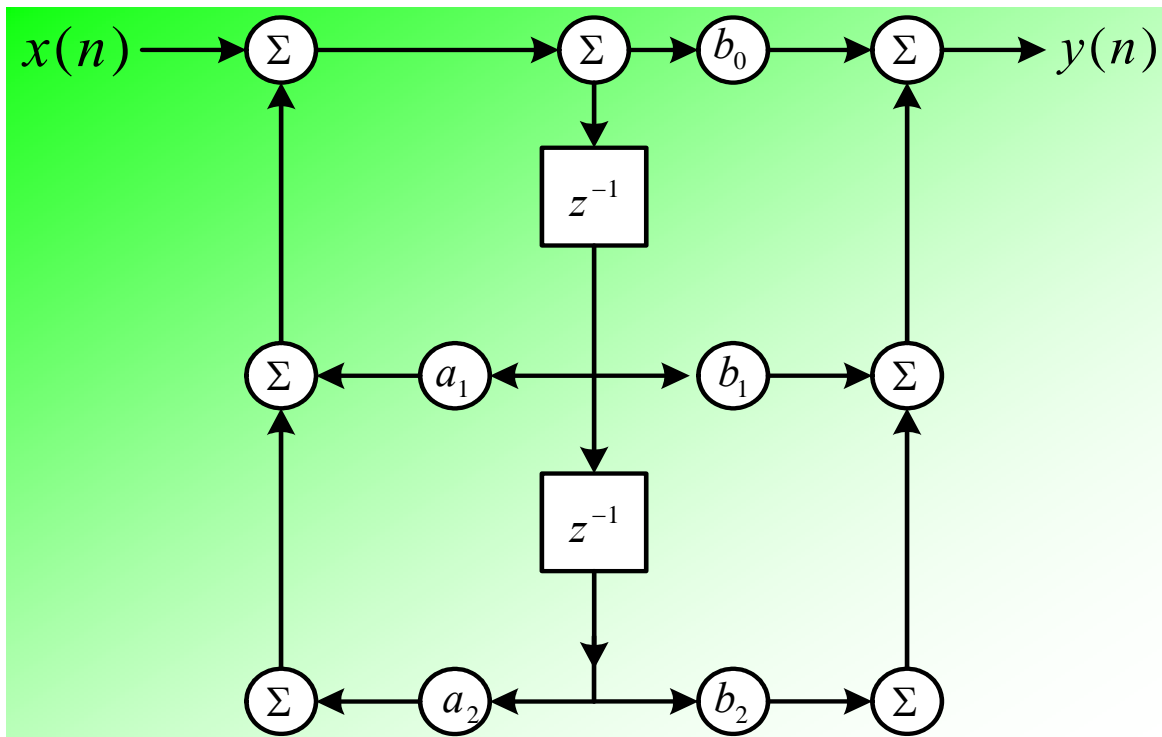
$$+ b_0x(n-6) + b_1x(n-5) + b_2x(n-4)$$





IIR Filter Structures

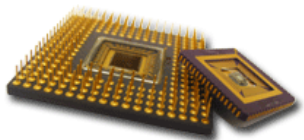
$$H(z) = \frac{b_0 + b_1z^{-1} + b_2z^{-2}}{1 - a_1z^{-1} - a_2z^{-2}}$$





Introduction to an Adaptive Algorithm

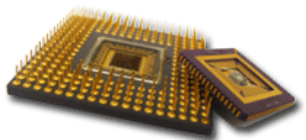
- ◆ Widely used in communication, DSP, and control system
- ◆ Deterministic gradient / least square algorithm
 - Steepest descent algorithm
 - RLS algorithm
- ◆ Stochastic gradient algorithm
 - LMS algorithm, DLMS algorithm
 - Block LMS algorithm
 - Gradient Lattice algorithm





Adaptive Applications

- ◆ Channel equalizer
- ◆ System identification
- ◆ Image enhancement
- ◆ Echo canceller
- ◆ Noise cancellation
- ◆ Predictor
- ◆ Line enhancement
- ◆ Beamformer





Notation

1. *Input Signal: $X(n)$*

2. *Desired Output: $d(n)$*

3. *Weight Vector: $W(n)$*

4. *Adaptation Factor: μ*

5. *Error: $e(n)$*

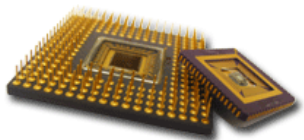
6. *Misadjustment: M_{adj}*

7. *Tap Number: N*

8. *Autocorrelation Matrix: R*

9. *Eigenvalue: λ*

10. *Diagonal Matrix : Λ*





Steepest Descent Algorithm

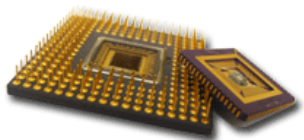
$$y(n) = W^T(n)X(n)$$

where $X(n) = [x(n) \ x(n-1) \ \dots \ x(n-N+1)]^T$

$$W(n) = [w_0(n) \ w_1(n) \ \dots \ w_{N-1}(n)]^T$$

◆ The error at the n-th time is

$$\begin{aligned} e(n) &= d(n) - y(n) \\ &= d(n) - W^T(n)X(n) \\ &= d(n) - X^T(n)W(n) \end{aligned}$$





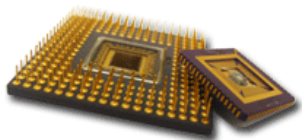
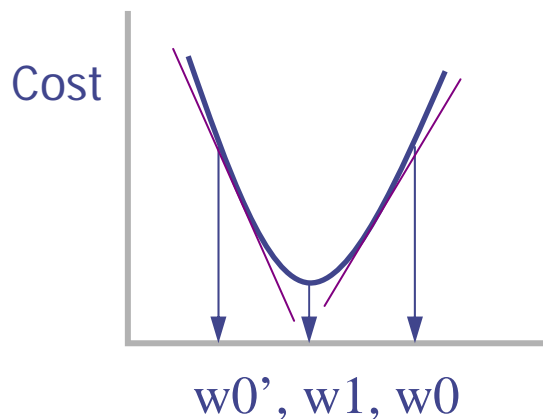
LMS Algorithm

- ◆ An efficient implementation in software of steepest descent using measured or estimated gradients
- ◆ The gradient of the square of a single error sample

$$W(n+1) = W(n) + \frac{1}{2} \mu (-\hat{\nabla} J(n))$$

$$\hat{\nabla} J(n) = -2e(n)X(n)$$

$$\Rightarrow W(n+1) = W(n) + \mu e(n)X(n)$$



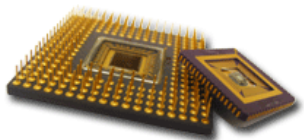


Summary of LMS Adaptive Algorithm (1960)

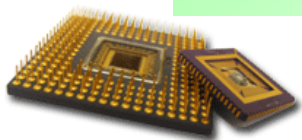
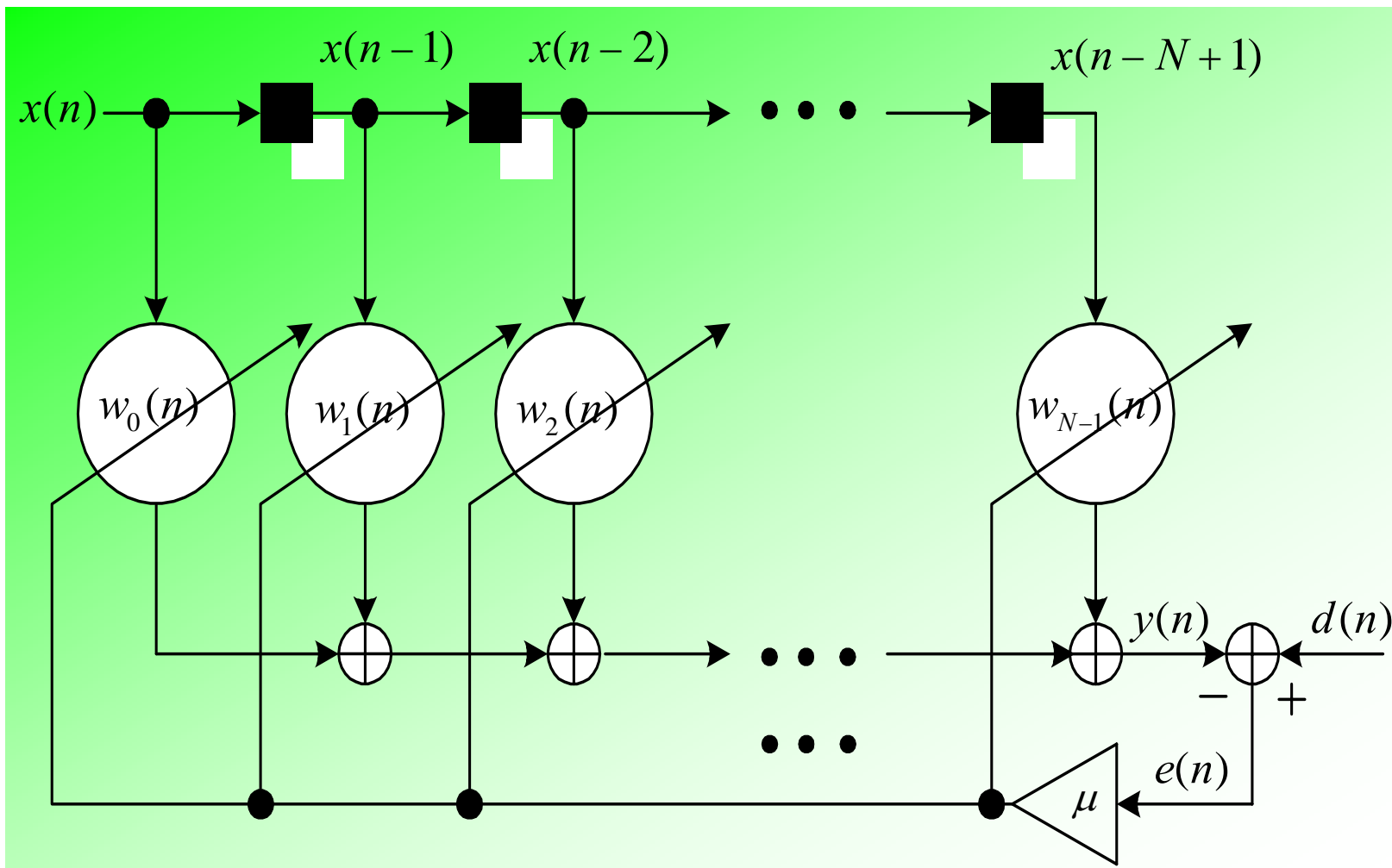
$$y(n) = \mathbf{w}^T(n)\mathbf{x}(n)$$

$$e(n) = d(n) - y(n)$$

$$\mathbf{w}(n+1) = \mathbf{w}(n) + \mu e(n)\mathbf{x}(n)$$



Block Diagram of an Adaptive FIR Filter Driven by the LMS Algorithm





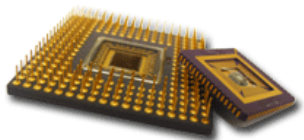
Unitary/Orthogonal Transform (1/4)

◆ Definition: (from Linear Algebra)

Let \mathbf{A} be $n \times n$ matrix that satisfies

$$\mathbf{A}\mathbf{A}^* = \mathbf{A}^*\mathbf{A} = \mathbf{I}.$$

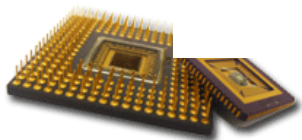
We call \mathbf{A} as an unitary matrix if \mathbf{A} has complex entries, and we call \mathbf{A} an orthogonal matrix if \mathbf{A} has real number.





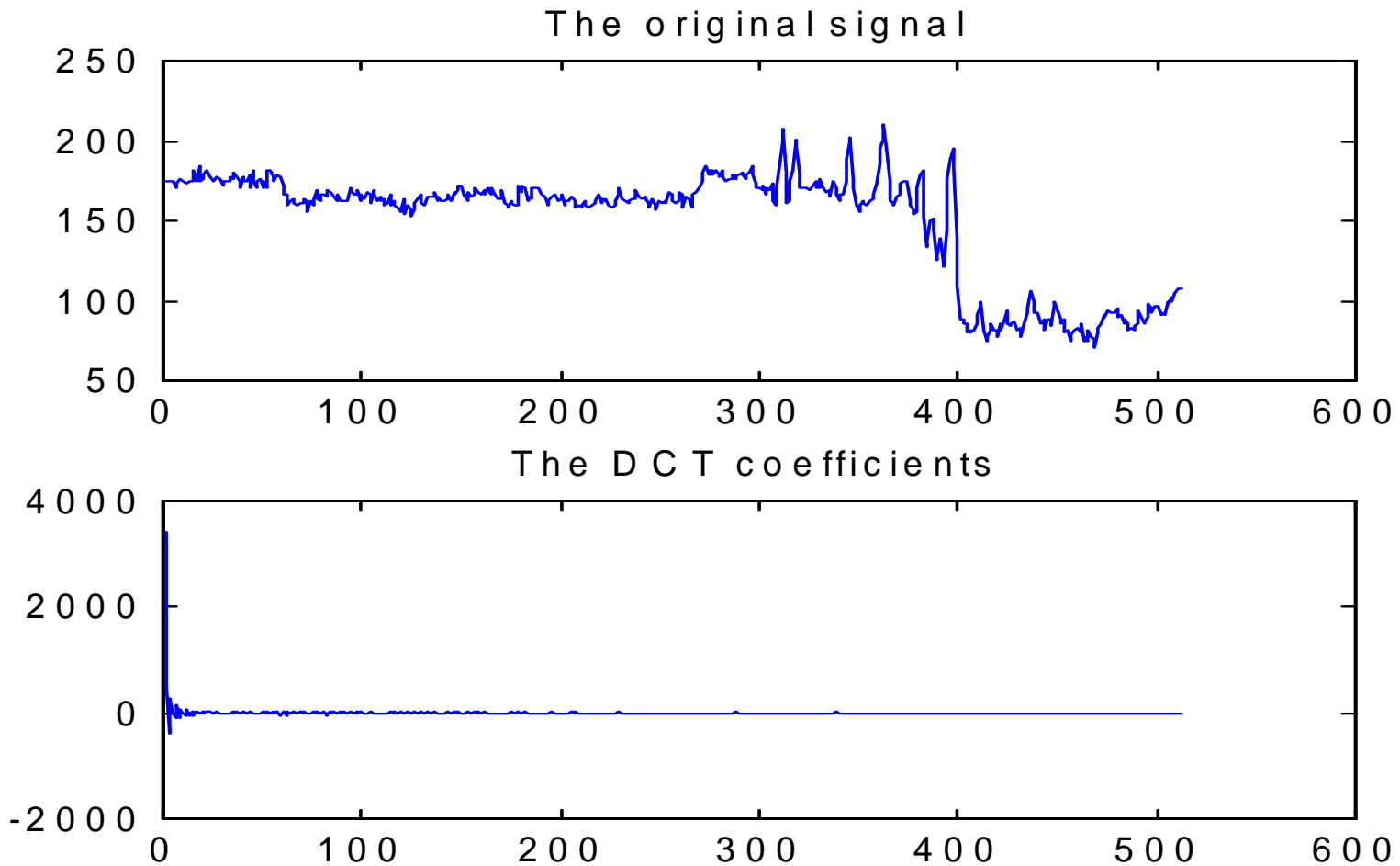
Why Orthogonal Transformation? (2/4)

- ◆ Energy conservation
- ◆ Energy compaction
 - Most unitary transforms tend to pack a large fraction of the average energy of signals into a relatively few components of the transform coefficients.
- ◆ Decorrelation
 - When signals are highly correlated, the transform coefficients tend to be uncorrected (or less correlated).
- ◆ Information preservation
 - The information carried by signals are preserved under a unitary transform.

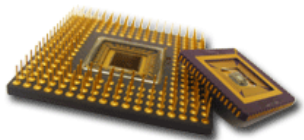




Why Orthogonal Transformation? (3/4)



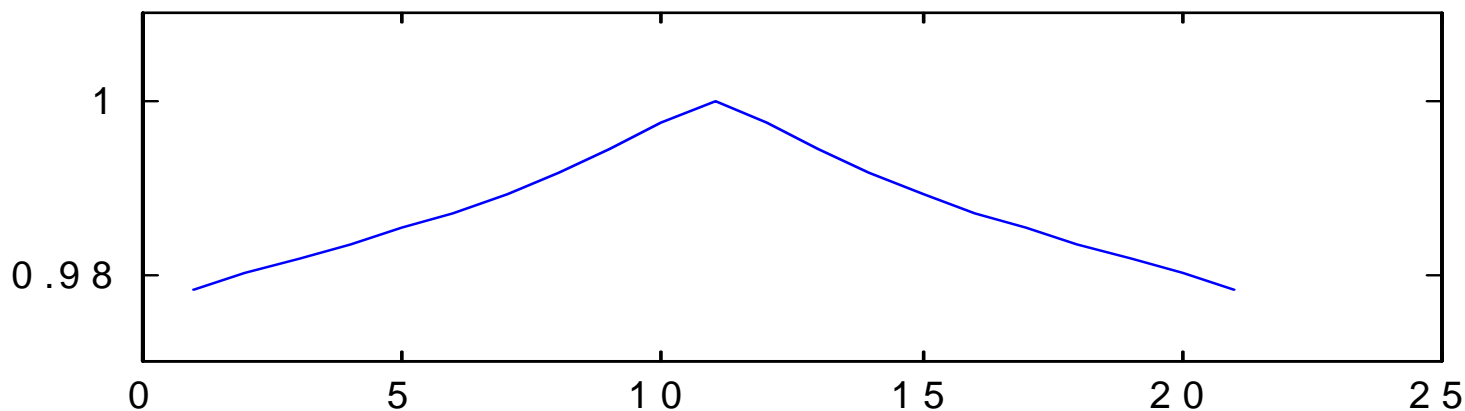
Source: Lecture of Prof. Dennis Deng



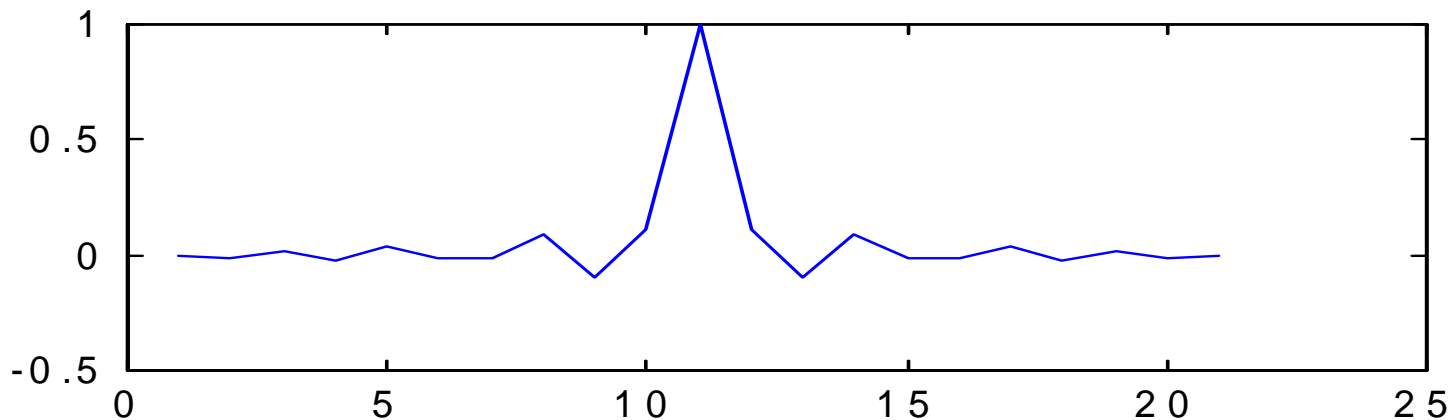


Why Orthogonal Transformation? (4/4)

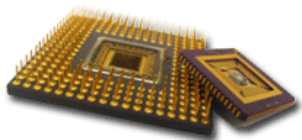
The auto correlation of original signal



The auto correlation of DCT coefficients



Source: Lecture of Prof. Dennis Deng





Discrete Fourier Transform (1/9)

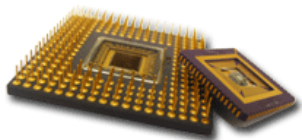
◆ DFT

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

$$W_N = e^{-j\frac{2\pi}{N}}, W_N^{nk} = e^{-j\frac{2\pi}{N}nk}$$

◆ IDFT

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

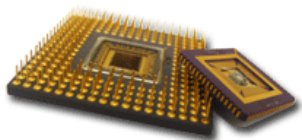




Fast Fourier Transform (2/9)

- ◆ The radix-2 algorithm is the most widely used fast algorithm to compute the DFT.
- ◆ Let us use an 8-point DFT ($N=8$) to illustrate the development of the fast algorithm

$$\begin{aligned}
 X(k) &= \sum_{n=0}^7 x(n)W_N^{kn} \\
 &= x(0) + x(2)W_N^{2k} + x(4)W_N^{4k} + x(6)W_N^{6k} \\
 &\quad + x(1)W_N^k + x(3)W_N^{3k} + x(5)W_N^{5k} + x(7)W_N^{7k} \\
 &= x(0) + x(2)W_N^{2k} + x(4)W_N^{4k} + x(6)W_N^{6k} \\
 &\quad + W_N^k (x(1) + x(3)W_N^{2k} + x(5)W_N^{4k} + x(7)W_N^{6k})
 \end{aligned}$$





Fast Fourier Transform (3/9)

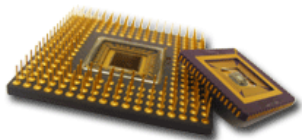
◆ Since
$$W_N^{2k} = e^{-j\frac{2\pi}{N}2k} = e^{-j\frac{2\pi}{(N/2)}k} = W_{N/2}^k$$

◆ 8-point DFT => Nearly two 4-point DFT

$$\begin{aligned} X(k) &= x(0) + x(2)W_{N/2}^k + x(4)W_{N/2}^{2k} + x(6)W_{N/2}^{3k} \\ &+ W_N^k (x(1) + x(3)W_{N/2}^k + x(5)W_{N/2}^{2k} + x(7)W_{N/2}^{3k}) \\ &= F_1(k) + W_N^k F_2(k), \quad \text{where } k = 0, 1, 2, \dots, N-1 \end{aligned}$$

where $F_1(k)$ and $F_2(k)$ represent the DFT of two sequences

$$f_1(n) = x(2n) \text{ and } f_2(n) = x(2n+1)$$





Fast Fourier Transform (4/9)

- ◆ One step further: (\Rightarrow Two 4-point DFT)

$$W_N^{k+N/2} = e^{-j\frac{2\pi}{N}(k+N/2)} = e^{-j\frac{2\pi}{N}k} e^{-j\pi} = -W_N^k \quad \text{and} \quad W_{N/2}^{k+N/2} = W_{N/2}^k$$

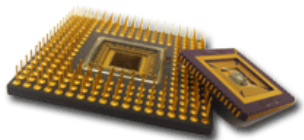
$$X(k) = F_1(k) + W_N^k F_2(k), \quad k = 0, 1, \dots, N/2 - 1$$

$$X(k + N/2) = F_1(k) - W_N^k F_2(k), \quad k = 0, 1, \dots, N/2 - 1$$

- ◆ An N-point DFT requires N^2 complex multiplications. The number of complex multiplications required by the above algorithm

$$2(N/2)^2 + N = 40 \quad (N = 8)$$

- ◆ An 8-point DFT requires 64 complex multiplications





Fast Fourier Transform (5/9)

- ◆ The 4-point DFT can be decomposed into two 2-point DFT in a similar way

$$\begin{aligned}
 F_1(k) &= x(0) + x(4)W_{N/2}^{2k} + x(2)W_{N/2}^k + x(6)W_{N/2}^{3k} \\
 &= x(0) + x(4)W_{N/4}^k + W_{N/2}^k(x(2) + x(6)W_{N/4}^k) \\
 &= V_{11}(k) + W_{N/2}^k V_{12}(k)
 \end{aligned}$$

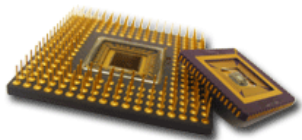
where $V_{11}(k)$ and $V_{12}(k)$ represent the DFT of two sequences

- ◆ As before,

$$v_{11}(n) = f_1(2n) \text{ and } v_{12}(n) = f_1(2n+1)$$

$$F_1(k) = V_{11}(k) + W_{N/2}^k V_{12}(k), k = 0, 1, \dots, N/4 - 1$$

$$F_1(k + N/2) = V_{11}(k) - W_{N/2}^k V_{12}(k), k = 0, 1, \dots, N/4 - 1$$



Fast Fourier Transform (6/9)

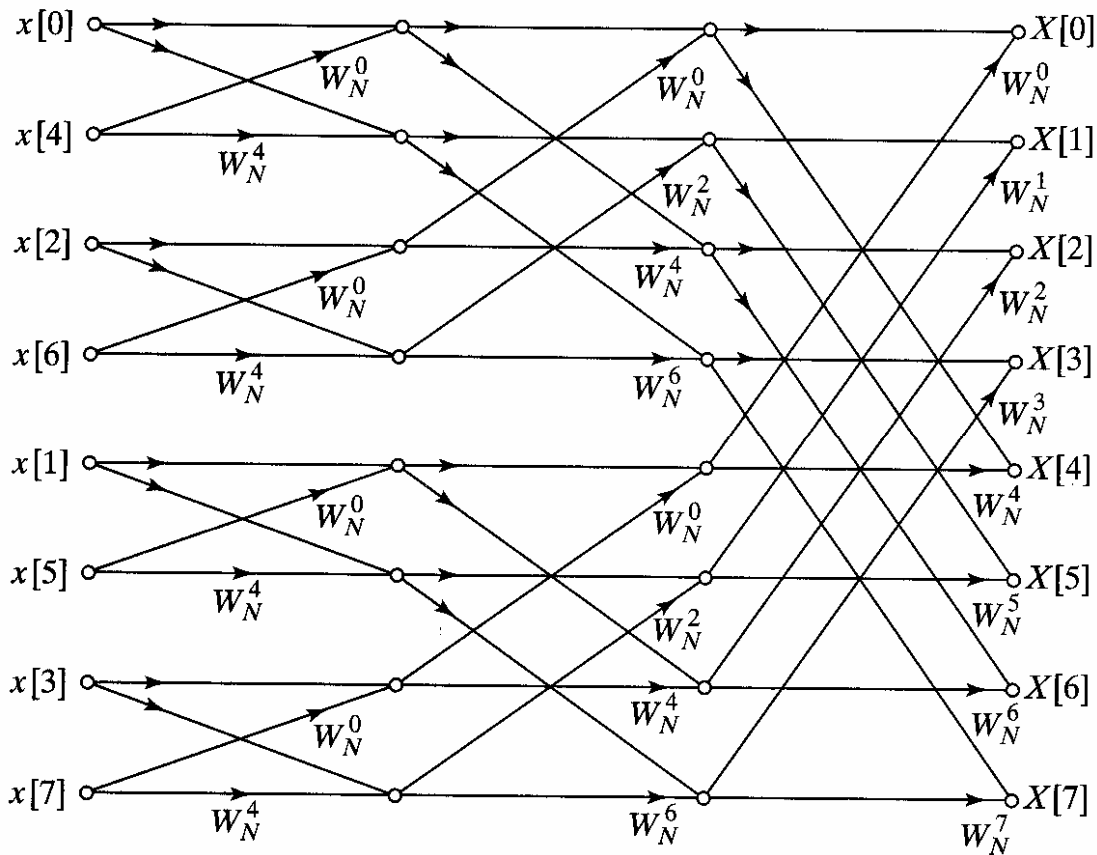
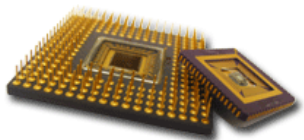


Figure 9.7. Flow graph of complete decimation-in-time decomposition of an 8-point DFT computation.





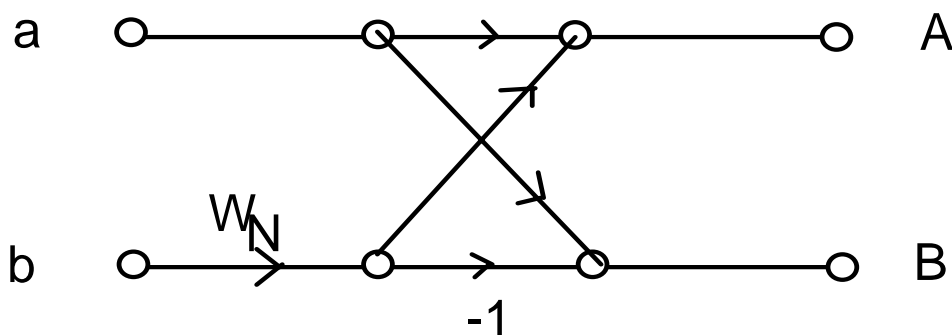
Fast Fourier Transform (7/9)

- ◆ A 2-point FFT, such as $V_{11}(k)$ and $V_{12}(k)$ involves only real addition

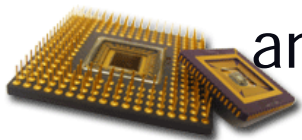
$$V_{11}(k) = x(0) + W_2^k x(4), W_2^0 = 1, W_2^1 = -1$$

$$V_{11}(0) = x(0) + W_N^0 x(4)$$

$$V_{11}(1) = x(0) - W_N^0 x(4)$$



- ◆ Each butterfly requires one complex multiplication and two complex addition

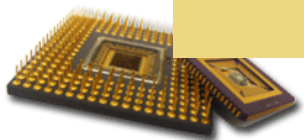




Fast Fourier Transform (8/9)

- ◆ After decimation, the sequence is in a *bit-reversed order*

<i>original order</i>		<i>decimation 1</i>		<i>decimation 2</i>	
	<i>$n_2n_1n_0$</i>		<i>$n_0n_2n_1$</i>		<i>$n_0n_1n_2$</i>
0	000	0	000	0	000
1	001	2	010	4	100
2	010	4	100	2	010
3	011	6	110	6	110
4	100	1	001	1	001
5	101	3	011	5	101
6	110	5	101	3	011
7	111	7	111	7	111



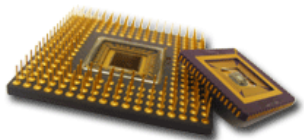


Fast Fourier Transform (9/9)

- ◆ This FFT algorithm is generally true for any data sequence of $N = 2^v$
- ◆ There are $N/2$ butterflies per stage and $\log_2 N$ stages
- ◆ The number of operations required for an FFT:
(Before simplifying)

Complex multiplication: $N \log_2 N$

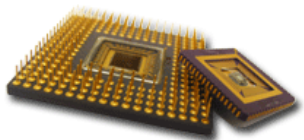
Complex addition: $N \log_2 N$





Image/Video Compression

- ◆ Where coding?
 - Source coding
 - Channel coding
- ◆ Source coding benefits
 - Lower bit rate
 - Less transmission time
 - Fewer storage data
- ◆ What kind of loss?
 - Lossless data compression
 - Lossy data compression
- ◆ Why can we do compression?
 - Coding redundancy
 - Inter-sample redundancy (Spatial redundancy)
 - Inter-frame redundancy (Temporal redundancy)





Source Coding Spectrum

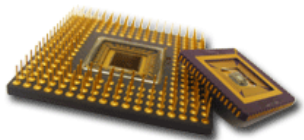
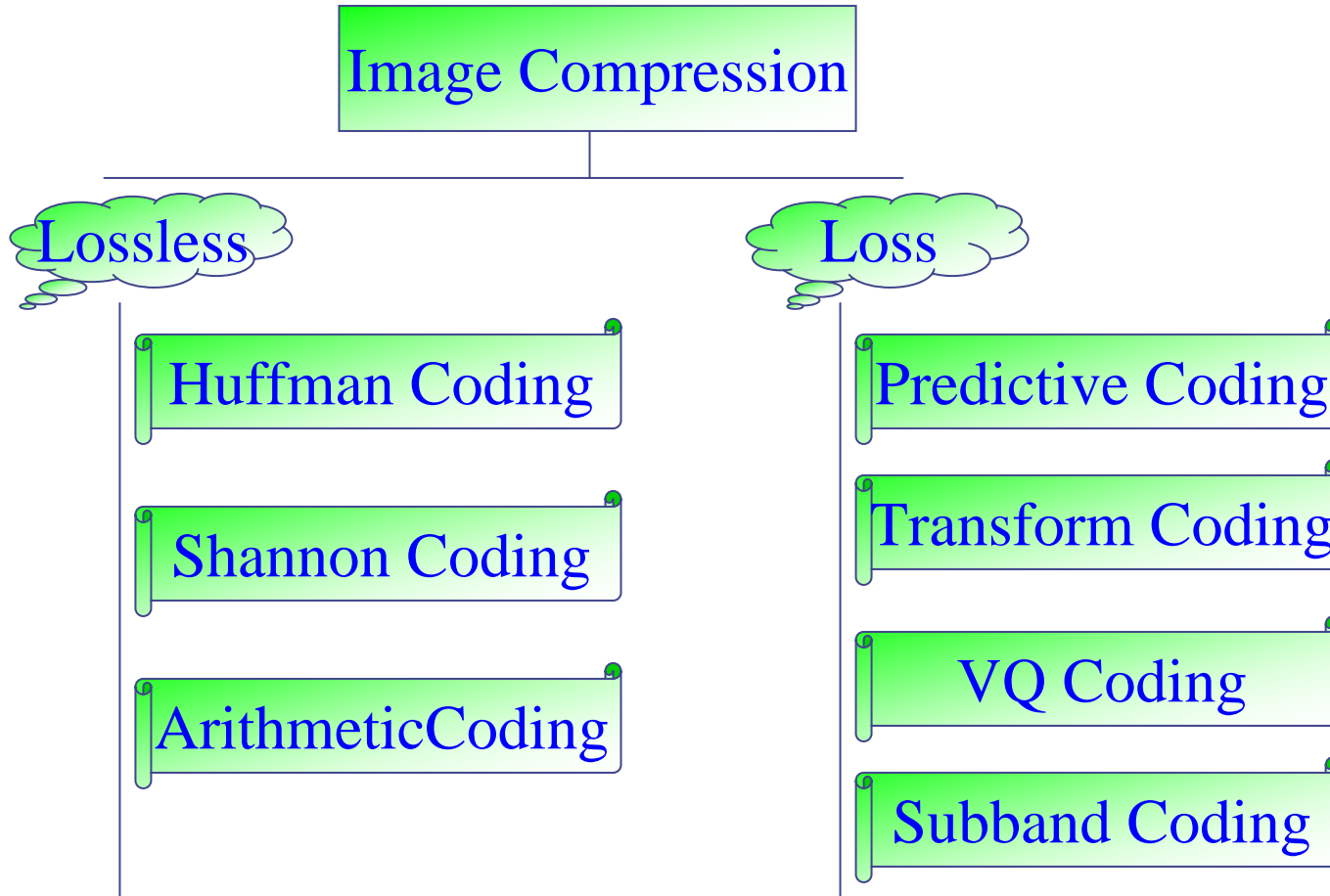




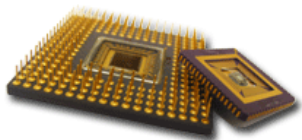
Image Measurement and Evaluation

$$\text{SNR(dB)} = 10 \log_{10} (\sigma_x^2 / \sigma_n^2)$$

$$\text{PSNR(dB)} = 10 \log_{10} (255^2 / \sigma_n^2)$$

$$\sigma_n = \sqrt{\frac{1}{N^2} \sum_{i=1}^N \sum_{j=1}^N [x(i, j) - \hat{x}(i, j)]^2}$$

where $x(i, j)$ and $\hat{x}(i, j)$ are the original image and reconstructed image values.



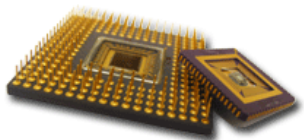


Discrete Cosine Transform (DCTII)

$$X(k) = \alpha(k) \sum_{n=0}^{N-1} x(n) \cos\left[\frac{(2n+1)k\pi}{2N}\right], \quad 0 \leq k \leq N-1$$

$$\alpha(0) = \sqrt{\frac{1}{N}}$$

$$\alpha(k) = \sqrt{\frac{2}{N}}, \quad \text{for } 1 \leq k \leq N-1$$





2-D DCT-II and IDCT-II

DCT-II

$$Z(k, l) = \frac{2}{N} \alpha(k) \alpha(l) \sum_{m=0}^{N-1} \sum_{n=0}^{N-1} x(m, n) \cos\left(\frac{(2m+1)k\pi}{2N}\right) \cos\left(\frac{(2n+1)l\pi}{2N}\right)$$

$$Z = AXA^T$$

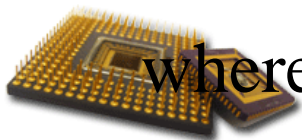
IDCT-II

$$x(m, n) = \frac{2}{N} \sum_{k=0}^{N-1} \sum_{l=0}^{N-1} \alpha(k) \alpha(l) Z(k, l) \cos\left(\frac{(2m+1)k\pi}{2N}\right) \cos\left(\frac{(2n+1)l\pi}{2N}\right)$$

$$X = A^T Z A$$

where k , l , m , and n range from 0 to $N-1$ and

where $\alpha(0) = 1/\sqrt{2}$ and $\alpha(j) = 1$ for $j \neq 0$.





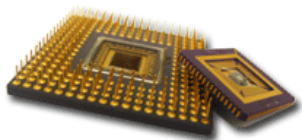
How to Decide the Coefficients?

Orthogonal Property

$$AA^T = A^T A = I$$

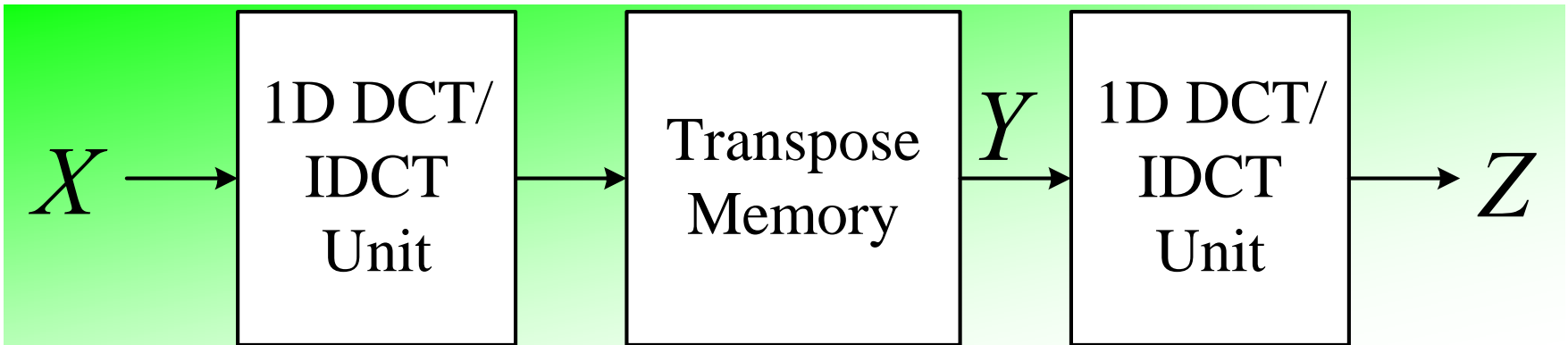
Parseval's Theorem: Energy Conservation

$$\sum_{n=0}^{N-1} |x(n)|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X(k)|^2$$

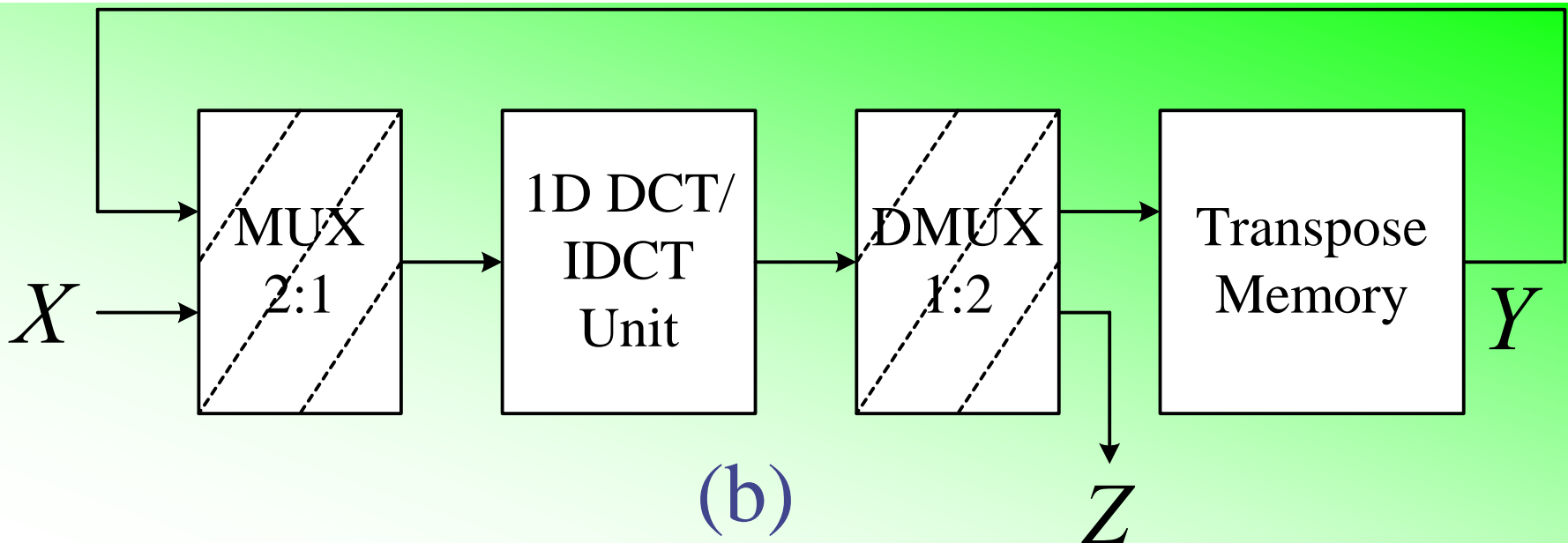




2-D DCT/IDCT Processor



(a)

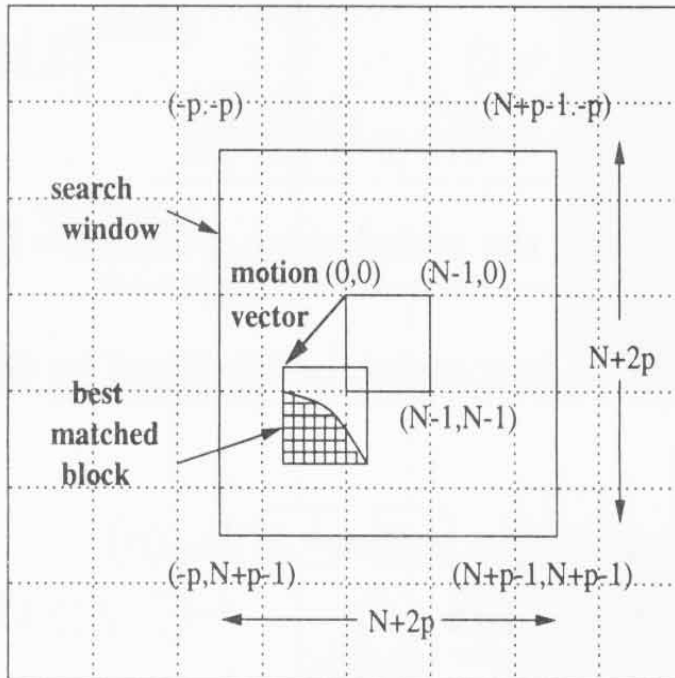


(b)

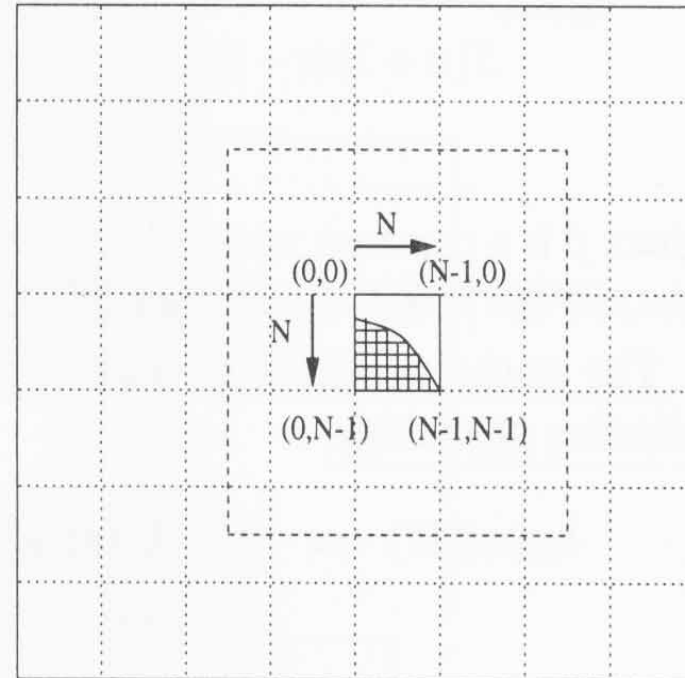


Block-Matching Algorithm

Previous frame



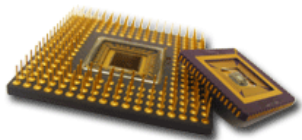
Current frame



Rule:
$$s(m,n) = \sum_{i=0}^{N-1} \sum_{j=0}^{N-1} |x(i,j) - y(i+m, j+n)| \quad \text{for } -p \leq m, n \leq p$$

$$u = \min_{(m,n)} \{s(m,n)\} \quad \text{for } -p \leq m, n \leq p$$

$$v = (m,n) | u$$





Huffman Coding (1/3)

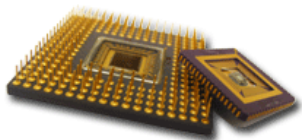
◆ Entropy

- Information Measurement
- Uncertainty Measurement
- Surprise Measurement

$$H(P) = \sum_{i=1}^q p_i \log_2(1/p_i)$$

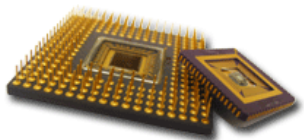
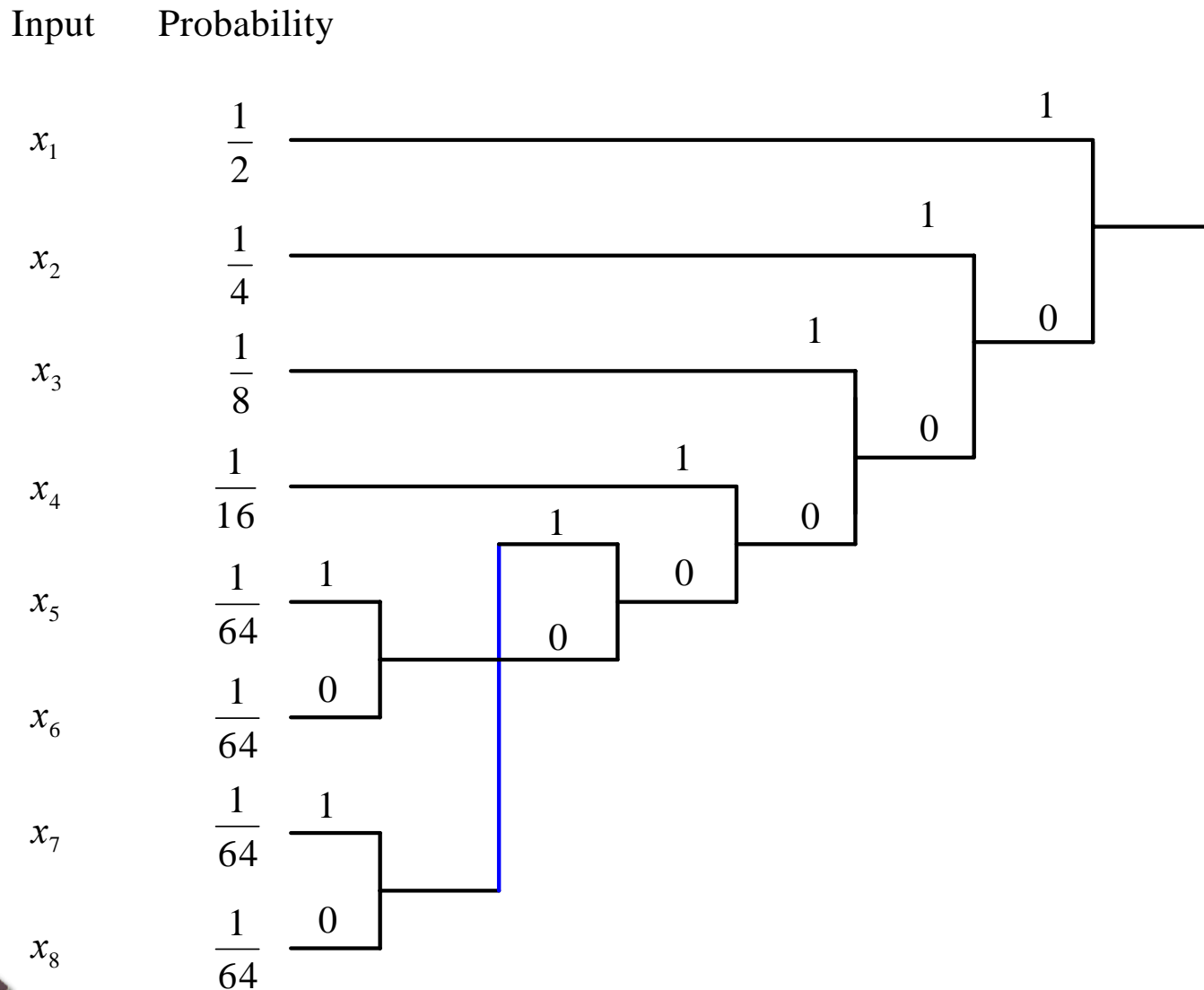
◆ Compression Ratio

$$Cr = \frac{\text{uncoding bits}}{\text{coding bits}}$$





Huffman Coding (2/3)





Huffman Coding (3/3)

$$\text{AvLen}_{\text{Natural_Code}} = 3 \text{ bit}$$

$$\text{Cr} = \frac{\text{uncoding bits}}{\text{coding bits}}$$

$$= \frac{3}{2} = 1.5$$

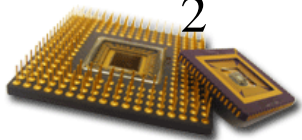
$$H(x)_{\text{Entropy}}$$

$$= \frac{1}{2} \log_2 2 + \frac{1}{4} \log_2 4 + \frac{1}{8} \log_2 8 + \frac{1}{16} \log_2 16 + \frac{1}{64} (4 \times \log_2 64) = 2 \text{ bit}$$

$$\text{AvLen}_{\text{Huffman_Code}}$$

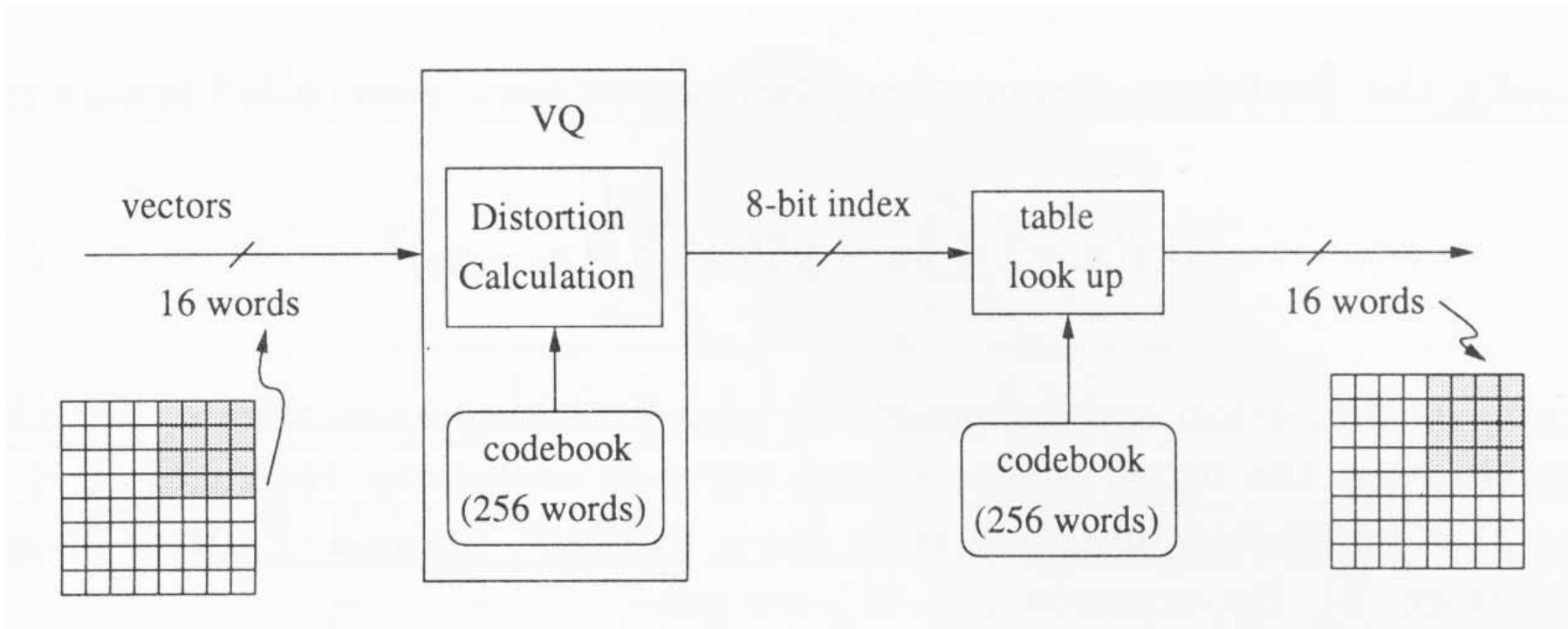
$$= \frac{1}{2} \times 1 + \frac{1}{4} \times 2 + \frac{1}{8} \times 3 + \frac{1}{16} \times 4 + \frac{1}{64} (4 \times 6) = 2 \text{ bit}$$

Data	Huffman Code	Natural Code
x_1	1	000
x_2	01	001
x_3	001	010
x_4	0001	011
x_5	000001	100
x_6	000000	101
x_7	000011	110
x_8	000010	111

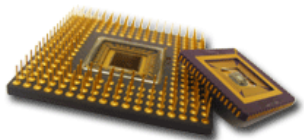




Vector Quantization



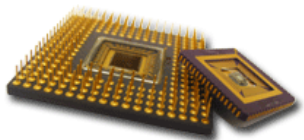
$$d(x, y) = \|x - y\|^2 = \sum_{i=0}^{k-1} (x_i - y_i)^2$$





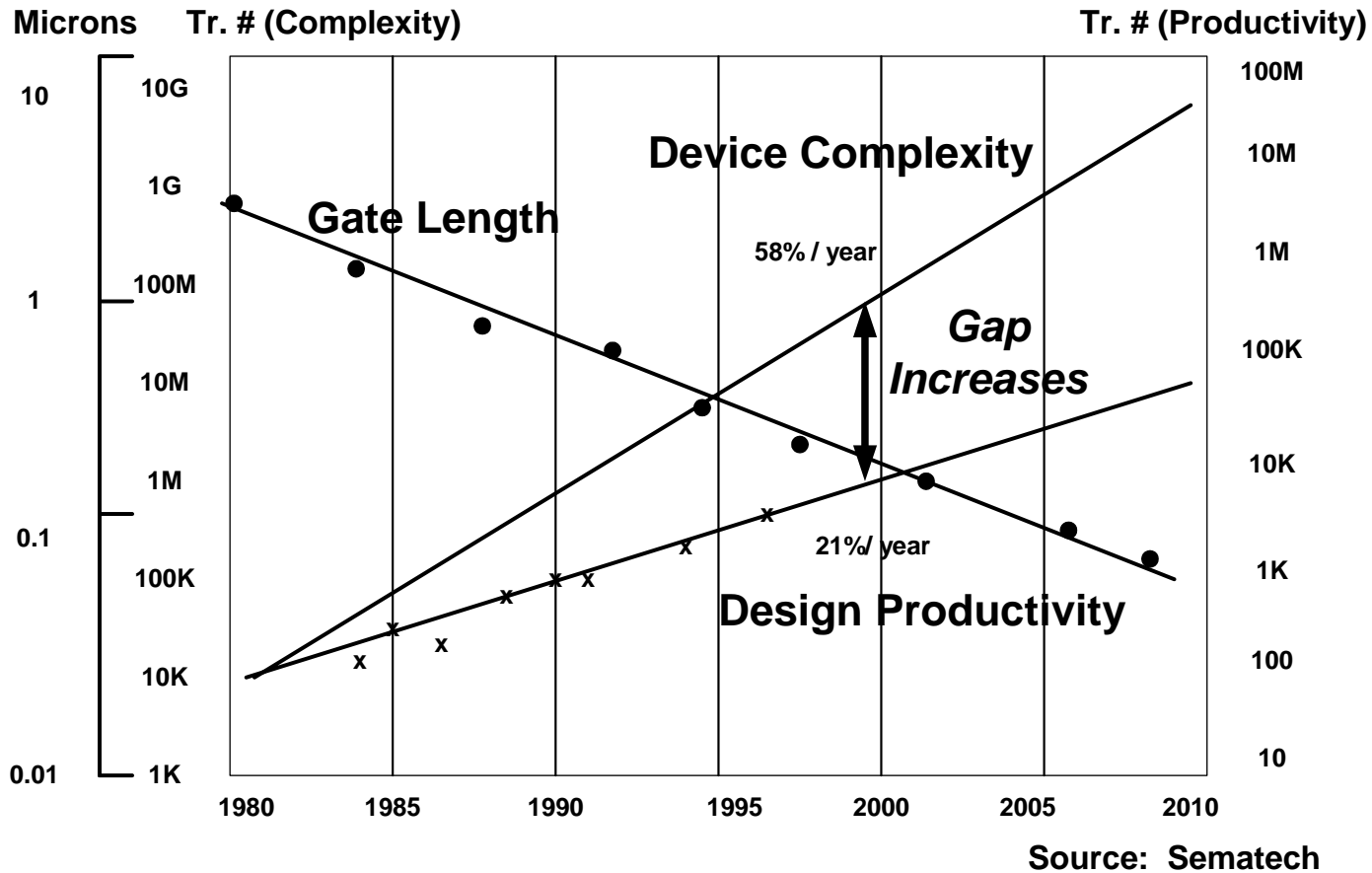
Outlines

- ◆ Features:
- ◆ DSP Algorithms
- ◆ *DSP Applications and CMOS IC's*
- ◆ Representations of DSP Algorithms





Moore's Law



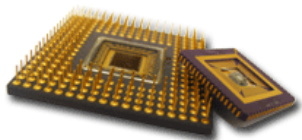
The number of transistors per chip doubles every 18 months.

* Cordon Moore: One of the founders of Intel



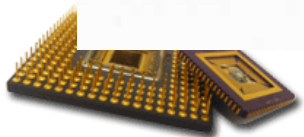
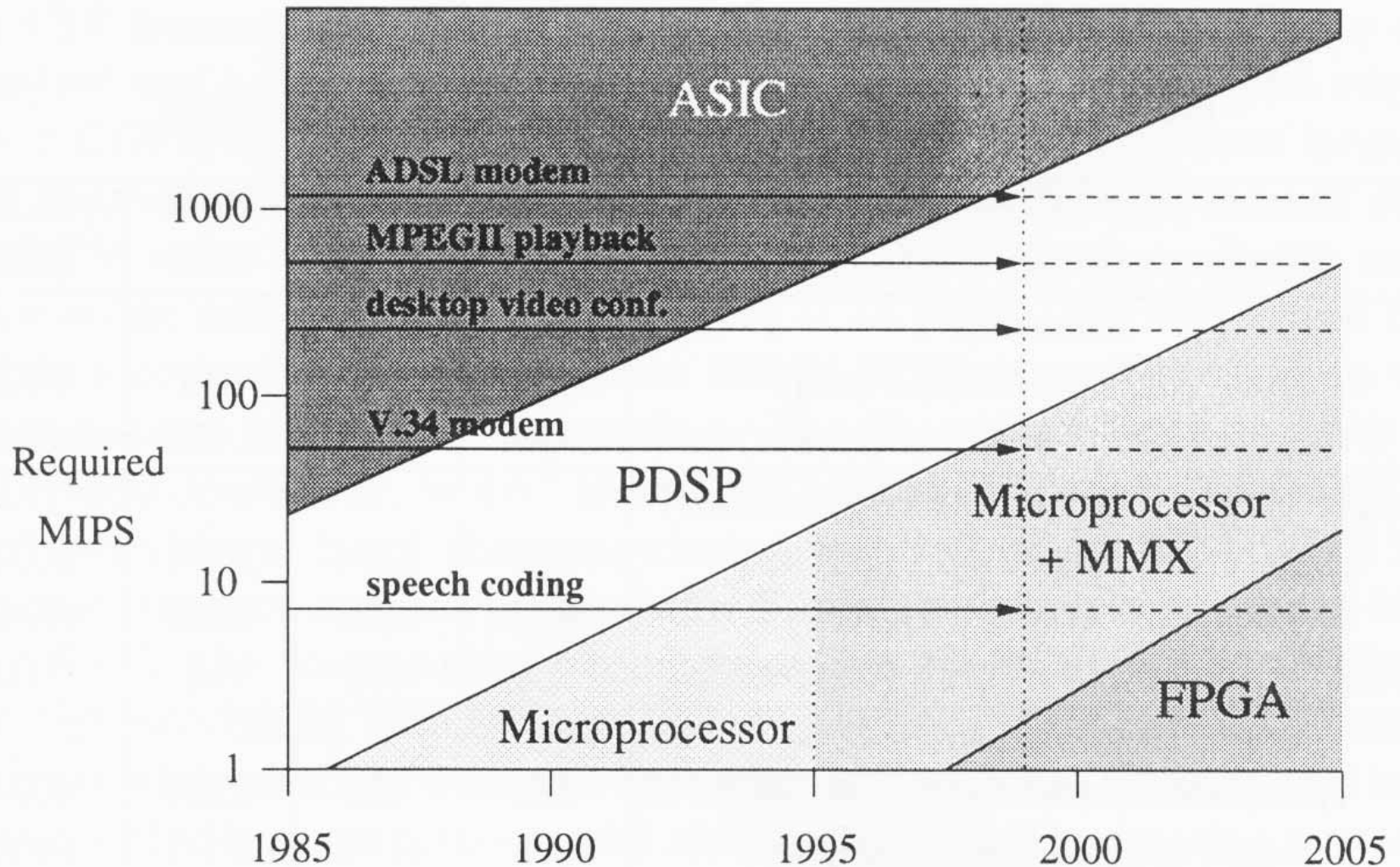
Common DSP Algorithms and Their Applications

DSP Algorithms	System Applications
Speech coding and decoding	Digital cellular phones, personal communication systems, digital cordless phones, multimedia computers, secure communications
Speech encryption and decryption	Digital cellular phones, personal communication systems, digital cordless phones, secure communications
Speech recognition	Advanced user interfaces, multimedia workstations, robotics and automotive applications, digital cellular phones, personal communication systems, digital cordless phones
Speech synthesis	Multimedia PCs, advanced user interfaces, robotics
Modem algorithms	Digital cellular phones, personal communication systems, digital cordless phones, digital audio broadcast, multimedia computers, wireless computing, navigation, data/facsimile modems, secure communications
Noise cancellation	Professional audio, advanced vehicular audio
Audio equalization	Consumer audio, professional audio, advanced vehicular audio
Image compression and decompression	Digital cameras, digital video, multimedia computers, consumer video
Beamforming	Navigation, radar/sonar, signals intelligence
Echo cancellation	Speakerphones, modems, telephone switches





Evolution of Applications



Chronological Table of Video Coding Standards



ITU-T
VCEG

H.261
(1990)

H.263
(1995/96)

H.263+
(1997/98)

H.263++
(2000)

MPEG-2
(H.262)
(1994/95)

H.264
(MPEG-4
Part 10)
(2002)

ISO/IEC
MPEG

MPEG-4 v1
(1998/99)

MPEG-4 v2
(1999/00)

MPEG-1
(1993)

MPEG-4 v3
(2001)

1990

1992

1994

1996

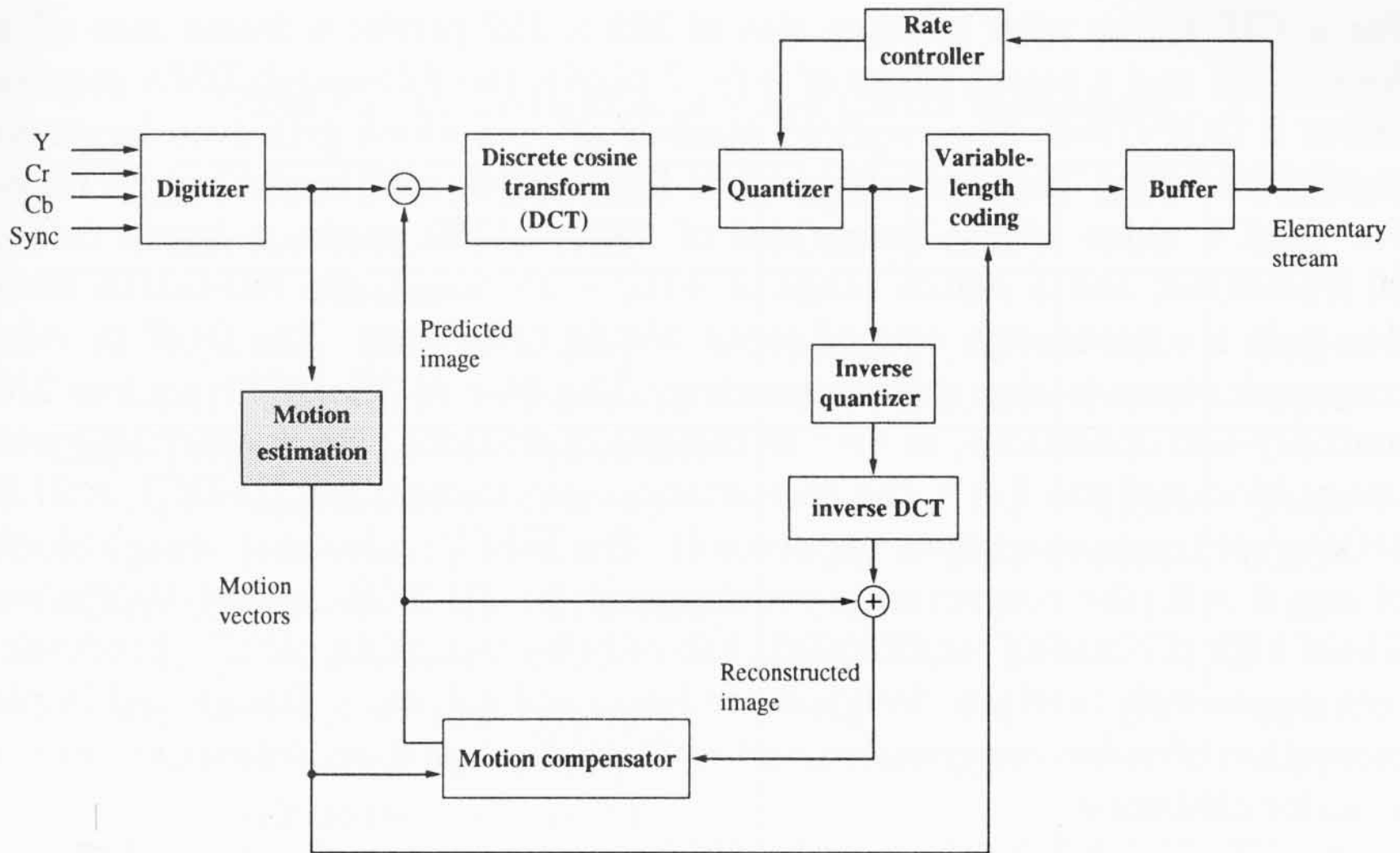
1998

2000

2002

2003

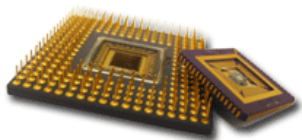
Block Diagram of MPEG-2 Encoder





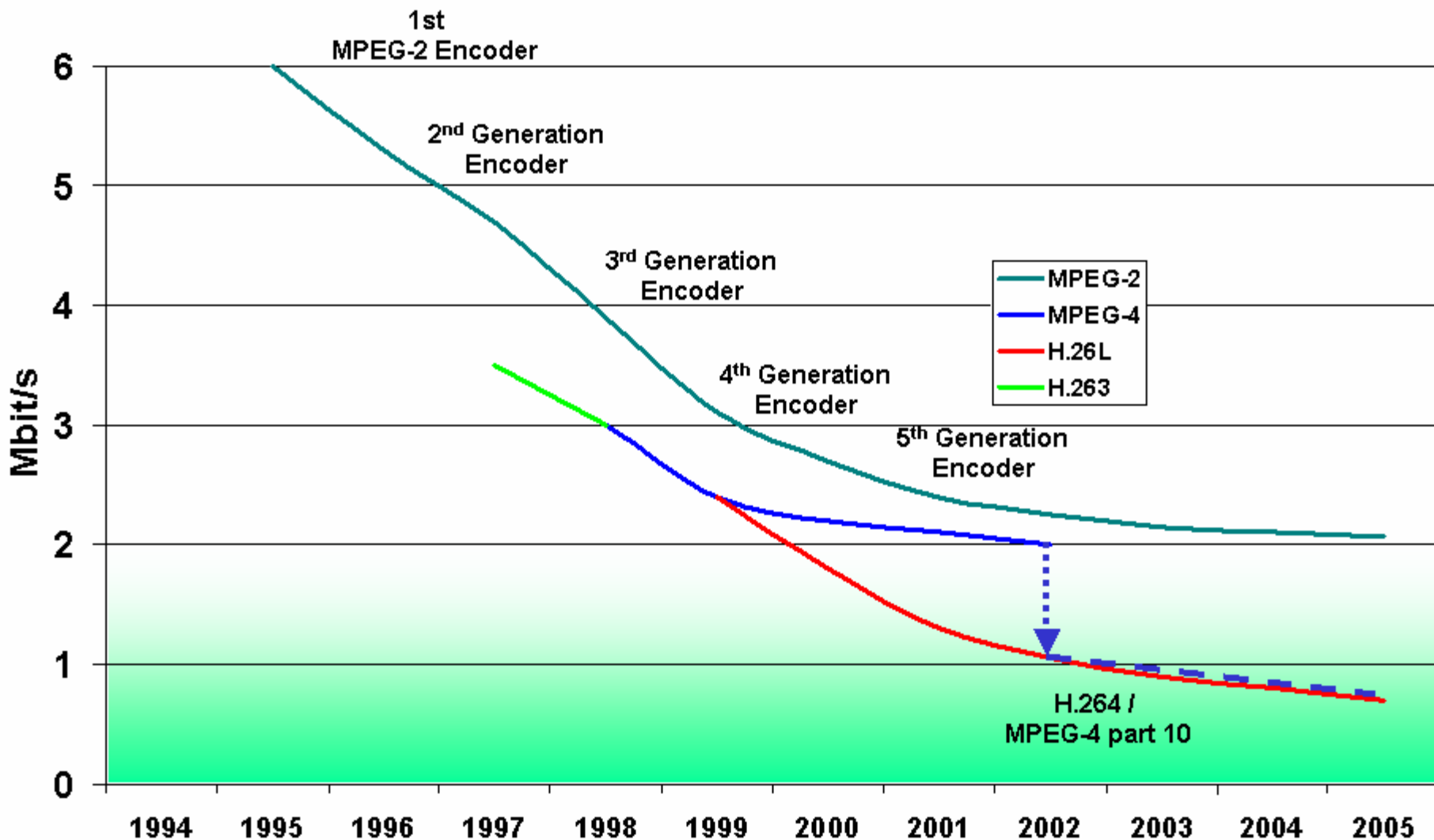
MPEG-2 / H.262: High Bit Rate, High Quality

- ◆ MPEG-2 contains 10 parts
- ◆ MPEG-2 Visual = H.262
- ◆ Not especially useful below 2 Mbps (range of use normally 2-20 Mbps)
- ◆ Applications: SDTV (2-5Mbps), DVD (6-8Mbps), HDTV (20Mbps), VOD
- ◆ Support for interlaced scan pictures
- ◆ PSNR, temporal, and spatial scalability
- ◆ “Profile” and “Level”
- ◆ 10-bit precision video sampling

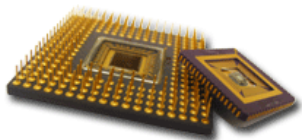
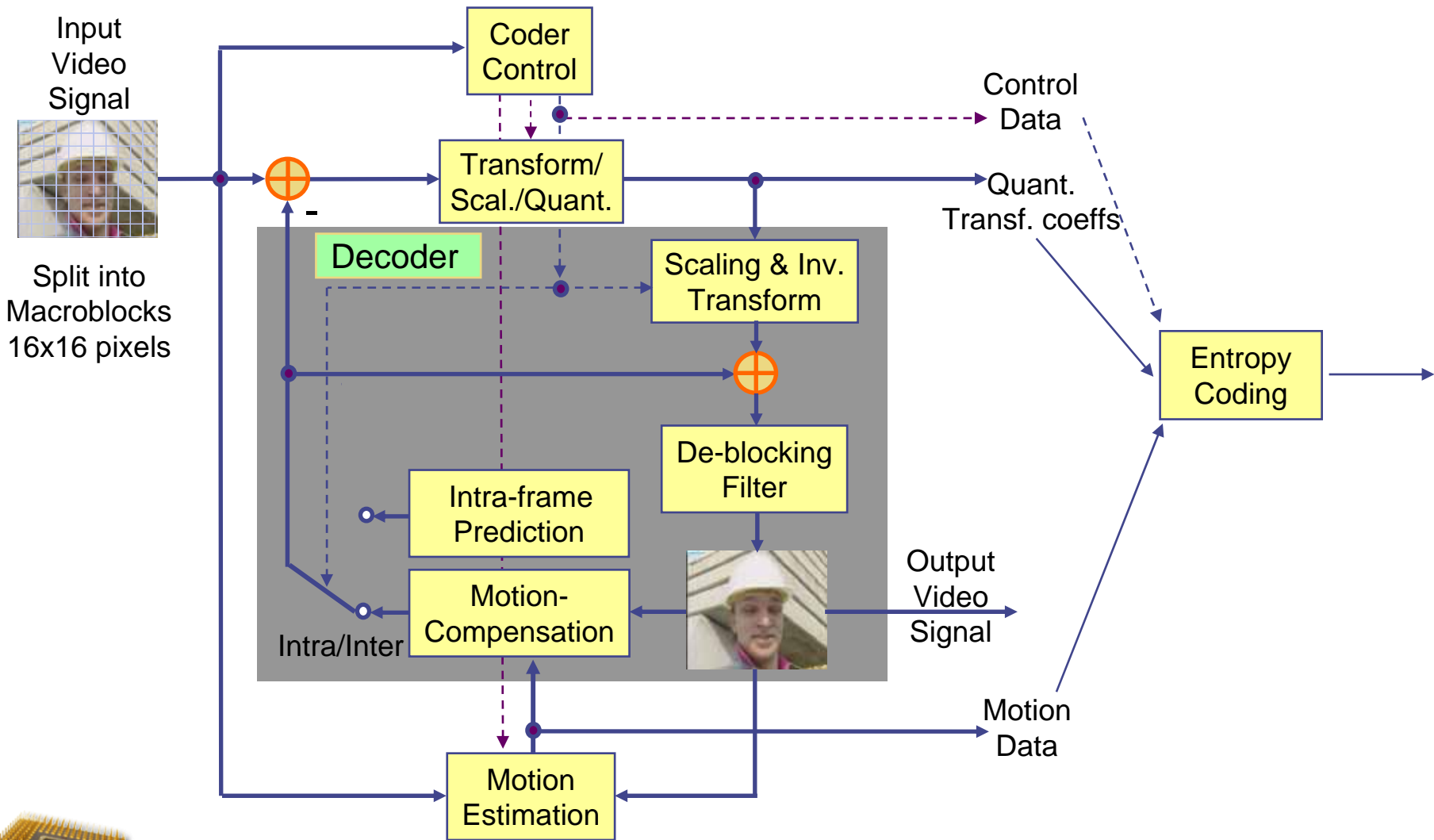




Position of H.264



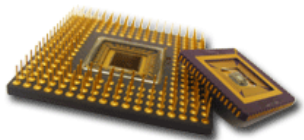
Block Diagram of H.2264/AVC Encoder





New Features of H.264

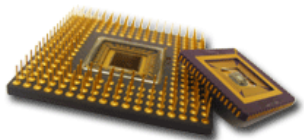
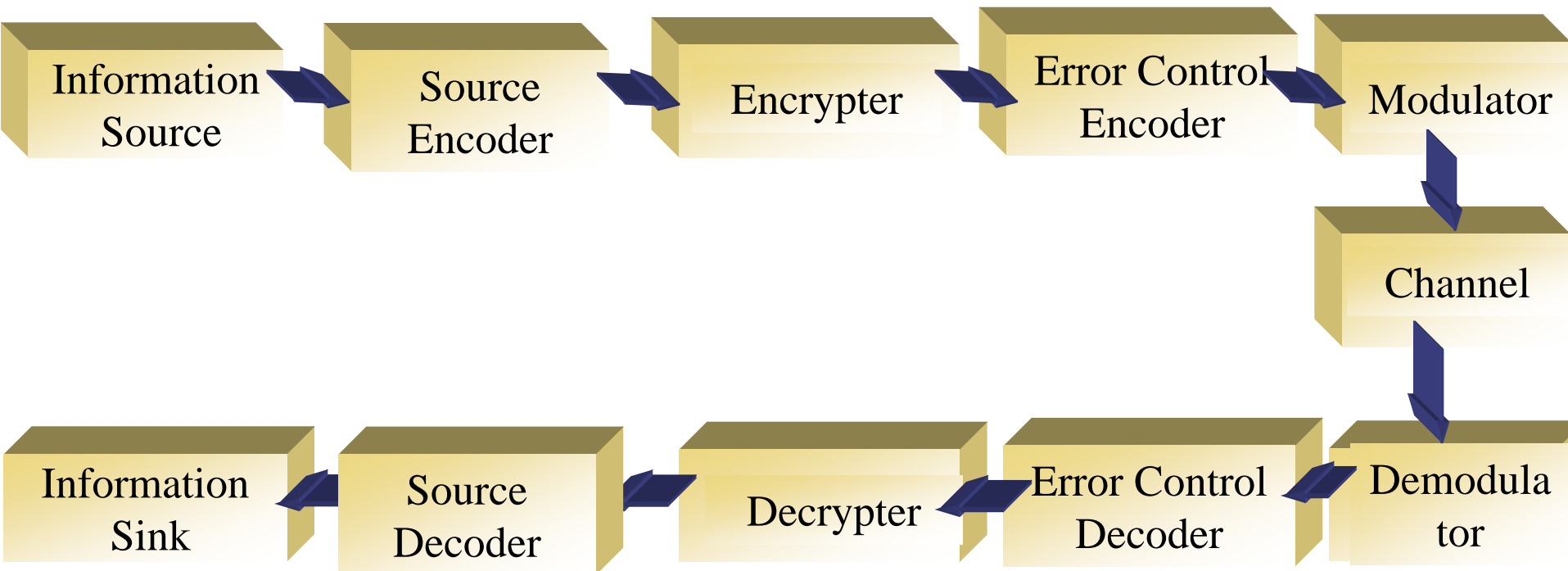
- ◆ Multi-mode, multi-reference MC
- ◆ Motion vector can point out of image border
- ◆ 1/4-, 1/8-pixel motion vector precision
- ◆ B-frame prediction weighting
- ◆ 4×4 integer transform
- ◆ Multi-mode intra-prediction
- ◆ In-loop de-blocking filter
- ◆ UVLC (Uniform Variable Length Coding)
- ◆ NAL (Network Abstraction Layer)
- ◆ SP-slices





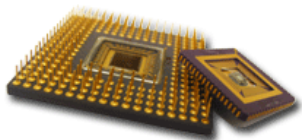
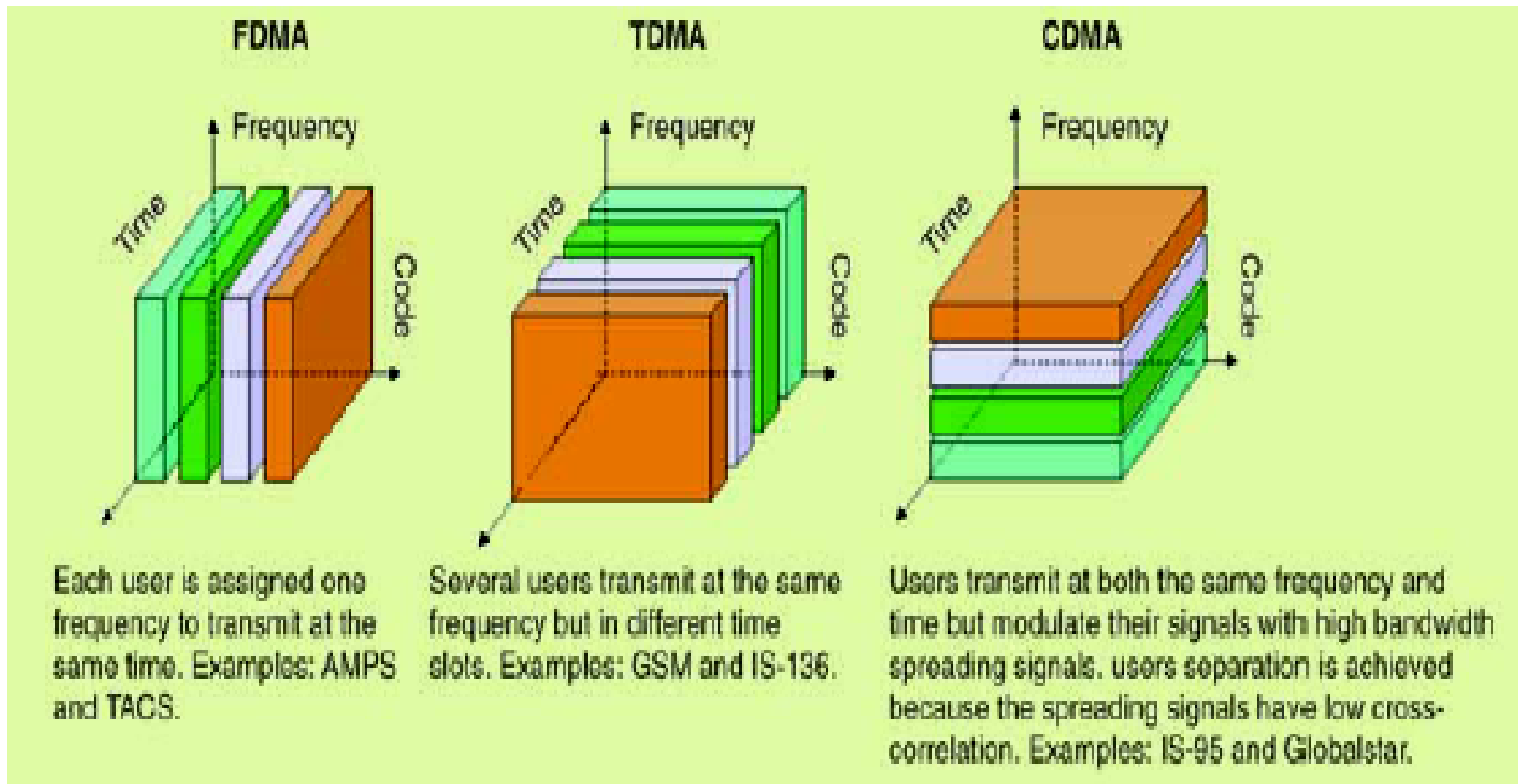
Digital Communications System

- ◆ Enabling the transmitted signal to withstand the effects of various channel impairments, such as noise, interference, and fading.





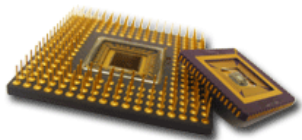
Multiple Access Techniques





Comparisons of Various Cellular Standards (1/2)

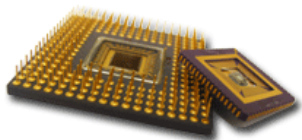
Technology	Frequency	Bandwidth	Bit Rate	Range
Bluetooth	2.4GHz	79MHz	1Mbps (Revision > 10K & 100 M)	10m
GPRS	890~915 MHz 935~960 MHz 1,900 MHz	200KHz	Typical: 80 kbps Up to 470 kbps	10km
W-CDMA	1,900~1,980 MHz 2,110~2,170 MHz	140MHz	144~384kbps Up to 2 Mbps	10km
PHS (Low Power ~10mW)	1,900~1,905 MHz 1,905~1,915 MHz	20MHz	128kbp (demo 2 MHz)	500m
802.11 a/b/g	2.4~5 GHz	83MHz	11~54Mbps (162 Mbps)	100m





Comparisons of WLAN Standards

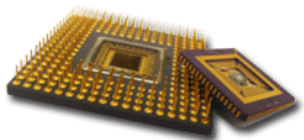
802.11x 標準				
	802.11a	802.11b	802.11g	802.11n
最快速率	54 Mbps	11 Mbps	54 Mbps	+100 Mbps
頻帶	5 GHz	2.4 GHz	2.4 GHz	2.4-5 GHz
傳輸距離	70m	100 m	100 m	+100m
IEEE 認證	1999 年	1999 年	2003 年	2006 年
Source : IEEE, 金鼎整理, 2006 年 3 月				





Outlines

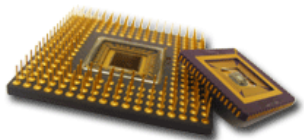
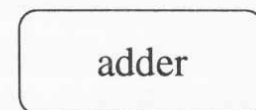
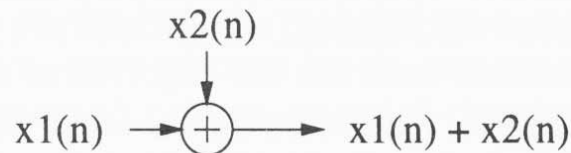
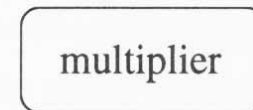
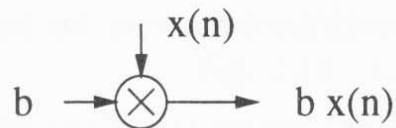
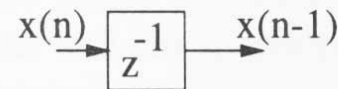
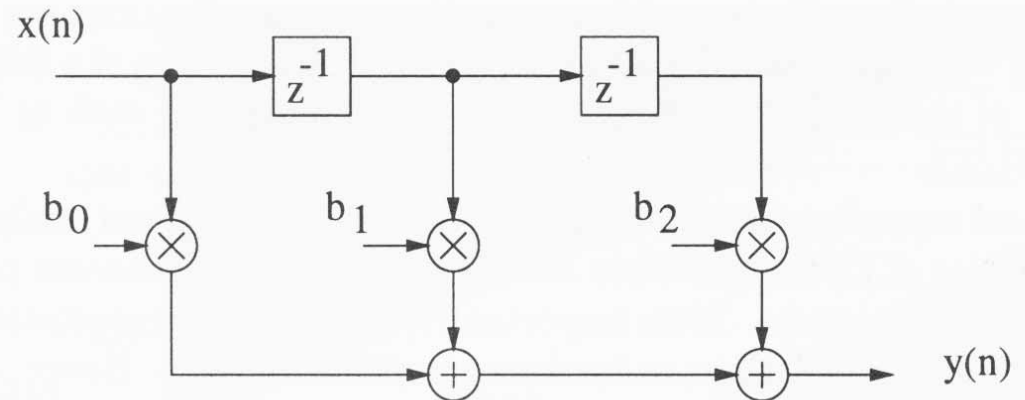
- ◆ Features:
- ◆ DSP Algorithms
- ◆ DSP Applications and CMOS IC's
- ◆ *Representations of DSP Algorithms*
 - Block Diagrams
 - Signal-Flow Graph
 - Data-Flow Graph
 - Dependence Graph





Block Diagram of a 3-Tap FIR Filter

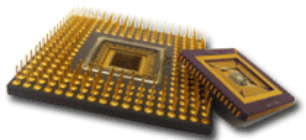
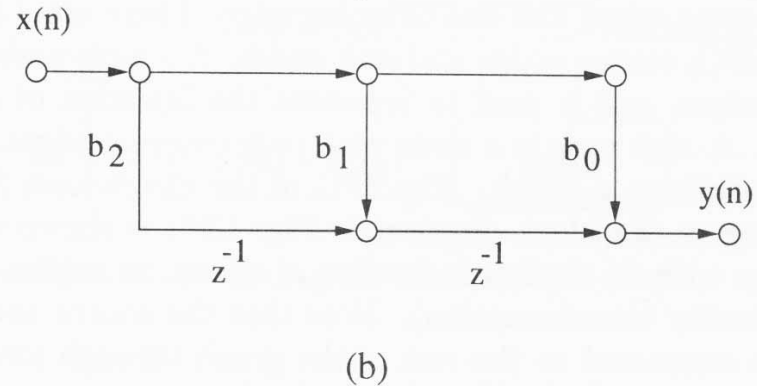
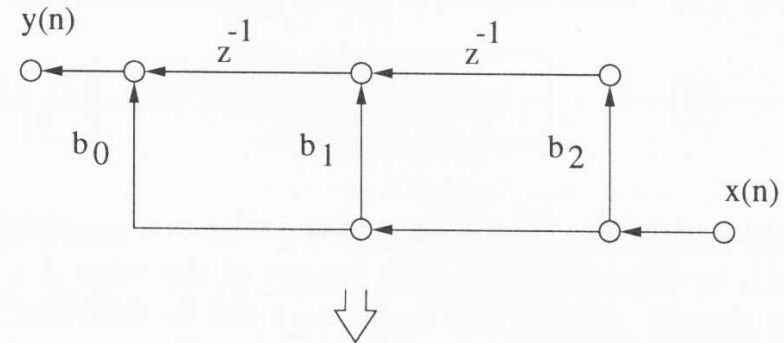
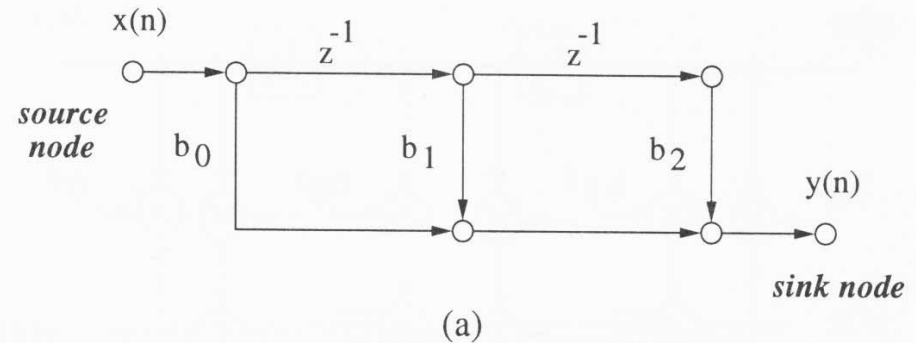
- ◆ Def: A block diagram consists of functional blocks connected with directed edges.





SFG of a 3-Tap FIR Filter

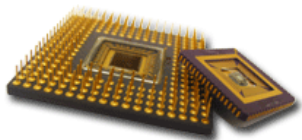
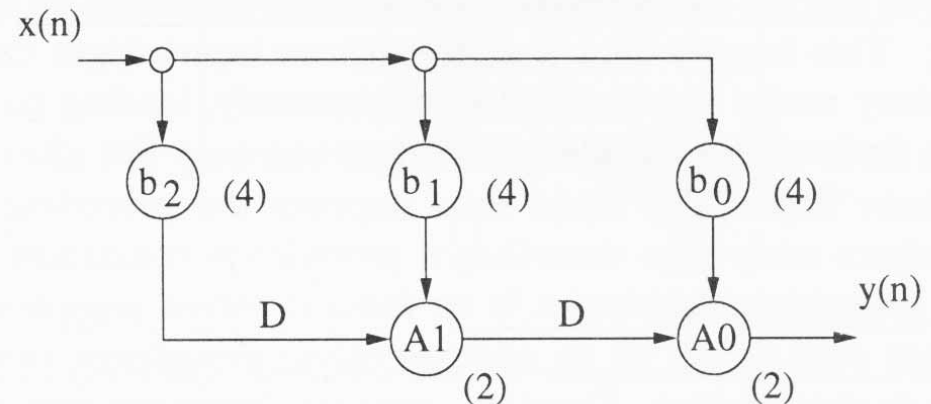
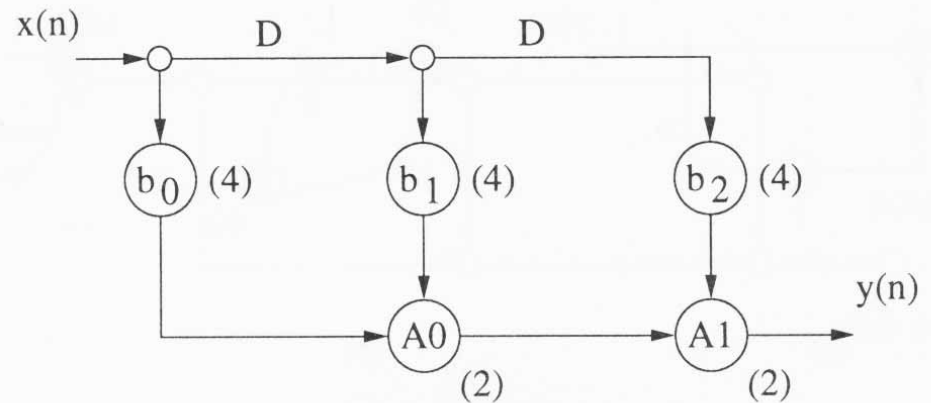
- ◆ Def: A signal flow graph (SFG) is a collection of nodes and directed edges. The nodes represent computations or tasks. In digital networks, the edges are usually restricted to constant gain multipliers or delay elements.





DFG of a 3-Tap FIR Filter

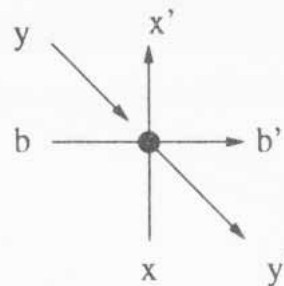
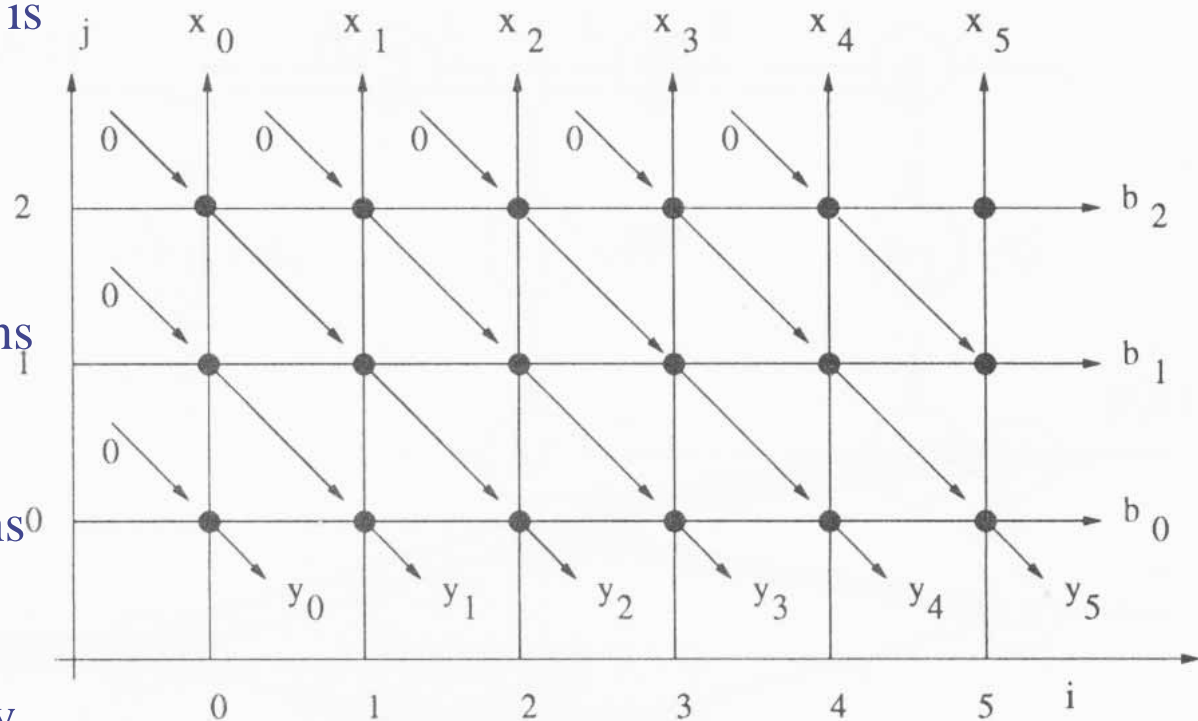
- ◆ Def: A data flow graph (DFG) is a collection of nodes and directed edges. The nodes represent computations (or functions or subtasks) and the directed edges represent data path and each edge has a nonnegative number of delays associated with it.



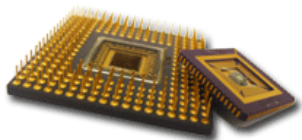


DG of a 3-Tap FIR Filter

◆ Def: A dependence graph is a direct graph that shows the dependence of the computations in an algorithm. The node in a DG represent computations and the edges represent precedence constraints among nodes. DG contains computations for all iterations in an algorithm and does not contain delay elements.



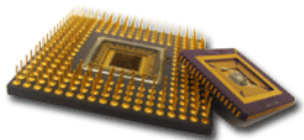
$$\begin{aligned} b' &= b \\ x' &= x \\ y' &= y + b x \end{aligned}$$





Conclusions

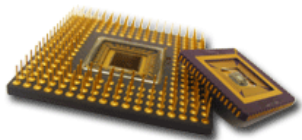
- ◆ Briefly introduced the following:
 - DSP design issue and design view
 - DSP algorithms
 - Overview of DSP applications
 - Representations of DSP algorithms





Self-Test Exercises

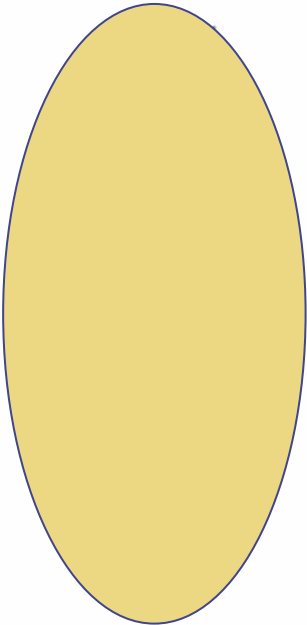
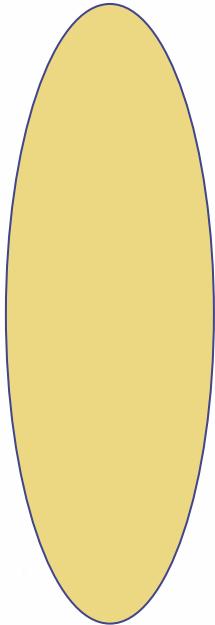
- ◆ STE1: What's difference between the convolution and digital filter?
- ◆ STE2: i) Please check the functionality of inverted form FIR filter structure using timing table. ii) Which one has higher speed between direct form and inverted form FIR filter?
- ◆ STE3: i) Derive the radix-2 algorithm to compute a 16-point FFT and draw a block diagram (using the butterfly) to illustrate its implementation. ii) Calculate the number of complex multiplication and addition operations required.
- ◆ STE4: If the length of the data sequence is not $N=2^y$ then zeros can be appended to the data sequence. How do you relate the DFT coefficients calculated with zero padding to those calculated without zero padding ? (use the Matlab *fft* function)

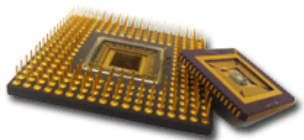




Self-Test Exercise

- ◆ STE5: According the following figure, calculate Code 2, $l_2(s_k)$, as well as the average code length using huffman code. Also compare the above average code length with that using Entropy Theorem.

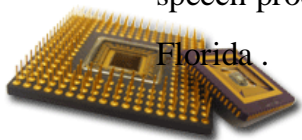
s_k	$p(s_k)$	Code 1	$l(s_k)$	Code 2	$l_2(s_k)$
0	0.19	000	3		
1	0.25	001	3		
2	0.21	010	3		
3	0.16	011	3		
4	0.08	100	3		
5	0.06	101	3		
6	0.03	110	3		
7	0.02	111	3		





References (1/2)

- [1] K. K. Parhi, *VLSI Digital Signal Processing Systems: Design and Implementation*. NY: Wiley, 1999.
- [2] P. Pirsch, *Architectures for Digital Signal Processing*. NY: Wiley, 1998.
- [3] A. V. Oppenheim and R. W. Schaffer, *Discrete-Time Signal Processing*. Englewood Cliffs, NJ: Prentice-Hall, 1989.
- [4] S. Haykin, *Adaptive Filter Theory*, 3rd ed. Englewood Cliffs, NJ: Prentice-Hall, 1996.
- [5] 連國珍,數位影像處理, 1992.
- [6] L. D. Van, S. S. Wang, and W. S. Feng, "Design of the lower-error fixed-width multiplier and its application", *IEEE Trans. Circuits Syst. II*, vol. 47, pp. 1112-1118, Oct. 2000.
- [7] L. D. Van, C. C. Yang, "Generalized low-error area-efficient fixed-width multipliers," *IEEE Trans. Circuits Syst. I*, vol. 52, pp. 1608-1619, Aug. 2005.
- [8] M. A. Song, L. D. Van, T. C. Huang, and S. Y. Kuo, "A generalized methodology for low-error and area-time efficient fixed-width Booth multipliers", *IEEE Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Accepted, 2004.
- [9] M. A. Song, L. D. Van, T. C. Huang and S. Y. Kuo, "A low-error and area-time efficient fixed-width Booth multiplier," *IEEE Int. Midwest Symp. Circuits Syst. (MWSCAS)*, Accepted, 2003.
- [10] L. D. Van and S. H. Lee, "A generalized methodology for lower-error area-efficient fixed-width multipliers," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2002, vol. 1, pp. 65-68, Phoenix, Arizona.
- [11] L. D. Van, S. S. Wang, S. Tenqchen, W. S. Feng, and B. S. Jeng, "Design of a lower error fixed-width multiplier for speech processing application," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 1999, vol. 3, pp. 130-133, Orlando Florida.





References (2/2)

- [12] L. D. Van, "A new 2-D systolic digital filter architecture without global broadcast," *IEEE Trans. VLSI Syst.*, vol. 10, pp. 477-486, Aug. 2002.
- [13] L. D. Van, C. C. Tang, S. Tenqchen, and W. S. Feng, "A new VLSI architecture without global broadcast for 2-D systolic digital filters," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2000, vol. 1, pp. 547-550, Geneva, Switzerland.
- [14] L. D. Van, S. Tenqchen, C. H. Chang, and W. S. Feng, "A new 2-D digital filter using a locally broadcast scheme and its cascade form," in *Proc. IEEE Asia Pacific Conf. on Circuits Syst. (APCCAS)*, Dec. 2000, pp. 579-582, Tianjin, China.
- [15] L. D. Van and W. S. Feng, "An efficient systolic architecture for the DLMS adaptive filter and its applications," *IEEE Trans. Circuits Syst. II*, vol. 48, pp. 359-366, April 2001.
- [16] L. D. Van, S. Tenqchen, C. H. Chang, and W. S. Feng, "A tree-systolic array of DLMS adaptive filter," in *Proc. IEEE Int. Conf. on Acoustics, Speech and Signal Processing (ICASSP)*, Mar. 1999, vol. 3, pp. 1253-1256, Phoenix, Arizona.
- [17] L. D. Van and W. S. Feng, "Efficient systolic architectures for 1-D and 2-D DLMS adaptive digital filters," in *Proc. IEEE Asia Pacific Conf. on Circuits Syst. (APCCAS)*, Dec. 2000, pp. 399-402, Tianjin, China.
- [18] L. D. Van and C. H. Chang, "Pipelined RLS adaptive architecture using relaxed Givens rotations (RGR)," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2002, vol. 1, pp. 37-40, Phoenix, Arizona.
- [19] L. D. Van and C. C. Yang, "High-speed area-efficient recursive DFT/IDFT architectures," in *Proc. IEEE Int. Symp. Circuits Syst. (ISCAS)*, May 2004, vol. 3, pp. 357-360, Vancouver, Canada.
- [20] L. D. Van, Y. C. Yu, C. M. Huang, C. T. Lin, "Low computation cycle and high speed recursive DFT/IDFT VLSI algorithm and architecture," in *Proc. IEEE Workshop on Signal Processing Systems (SiPS)*, Nov. 2005, pp. 579-584, Athens, Greece.
- [21] L. D. Van, H. F. Luo, C. M. Wu, W. S. Hu, C. M. Huang, and W. C. Tsai, "A high-performance area-aware DSP processor architecture for video codecs," in *Proc. IEEE Int. Conf. Multimedia and Expo. (ICME)*, Jun. 2004, vol. 3, pp. 1499-1502, Taipei, Taiwan.

