Algorithm-Architecture Co-Design for DSP Applications

Pramod Meher

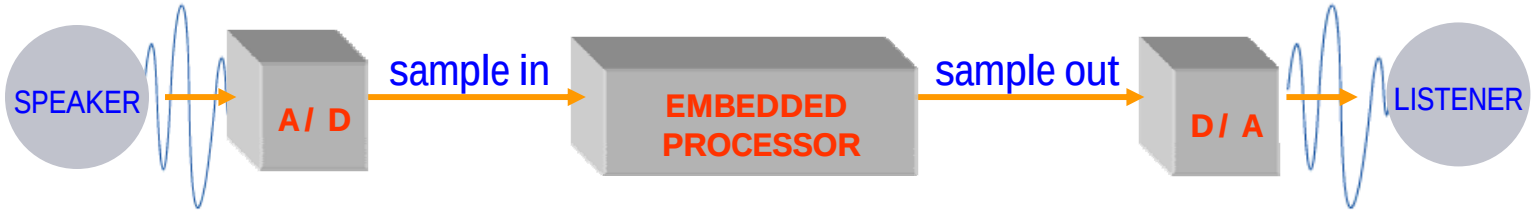
Institute for Infocomm Research
Singapore

outline

- scaling trends and design issues
- need of algorithm-architecture co-design
- design for catching up the speed
- design for dealing with power
- design for interconnect minimization
- design for matching computation with I/O
- design for demand-driven solution
- conclusions

scaling trends and design issues

real-time processing requirement

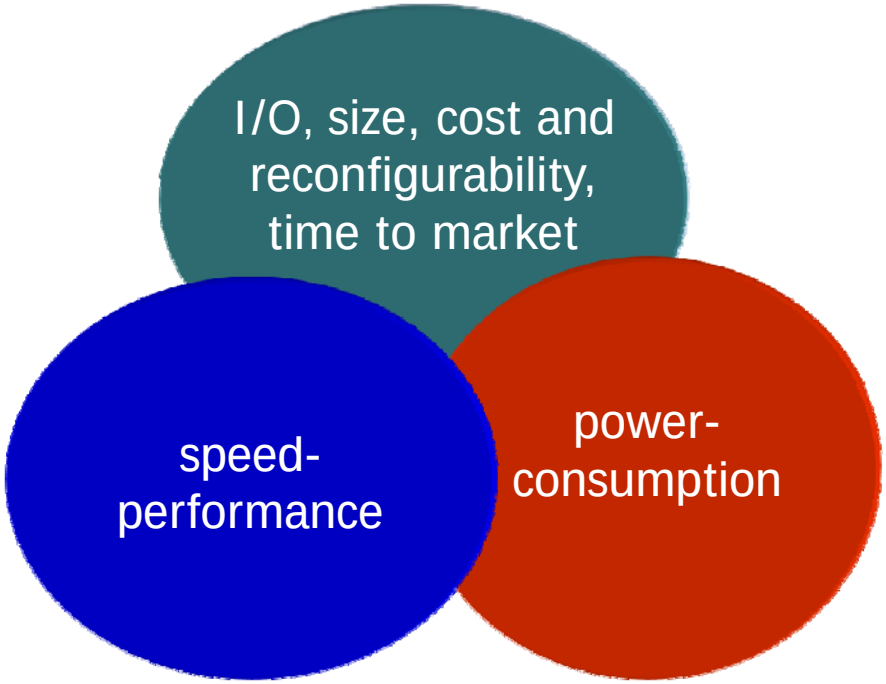


- The system must have enough speed performance to process the samples as fast as they arrive at. Slower processing will degrade the quality of reception.
- Computational demand per second = $N\,S$

  S= **no. of samples arriving at the processor/ second**

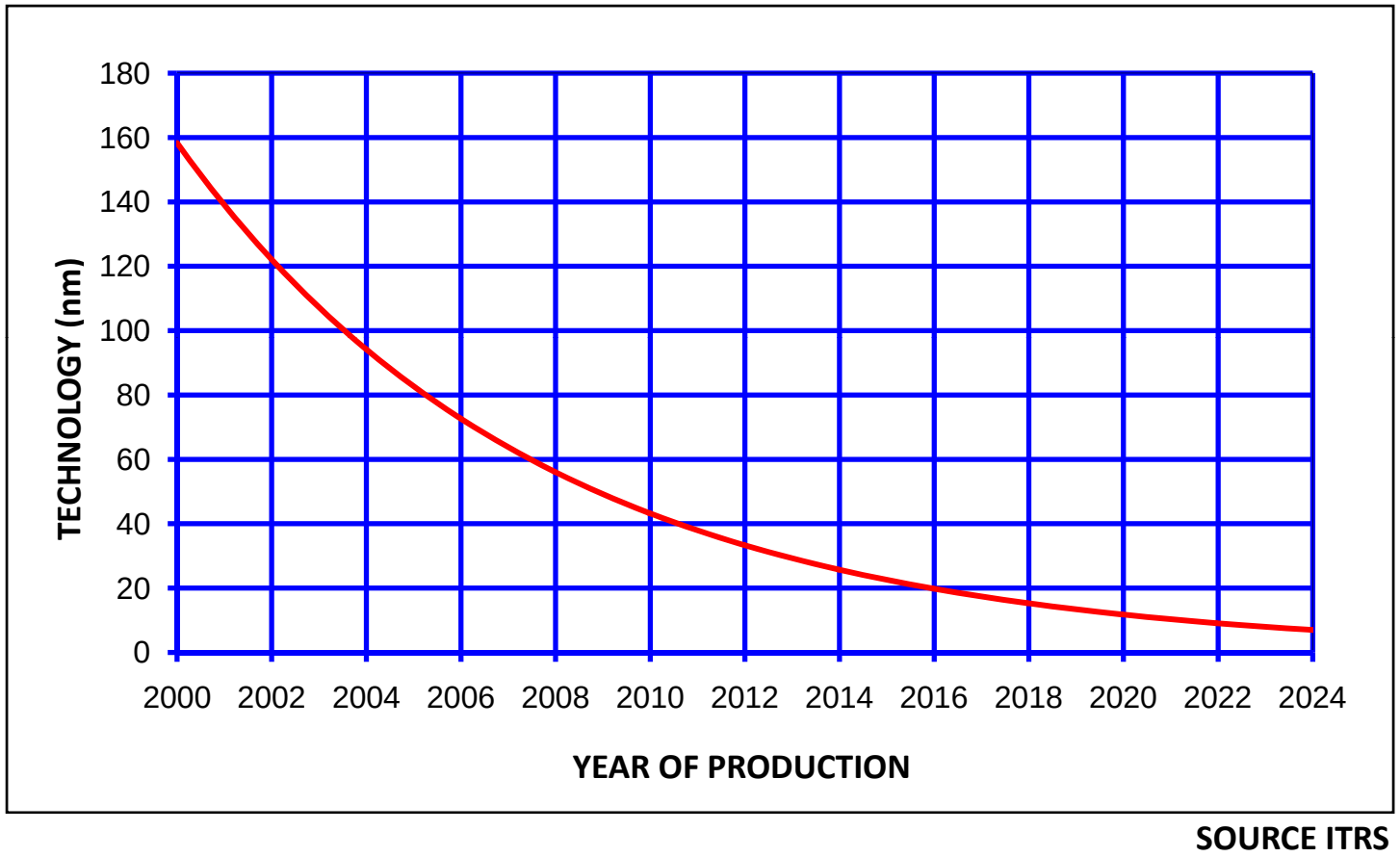
  N= **computation per sample required by the algorithm**
- Audio and telecommunication systems have sampling rate 10^4 to 10^6 samples/sec, and video applications need 10^7 to 10^8 samples/second. MPEG motion estimation requires several GOPs/sec.

design constraints and trade-offs



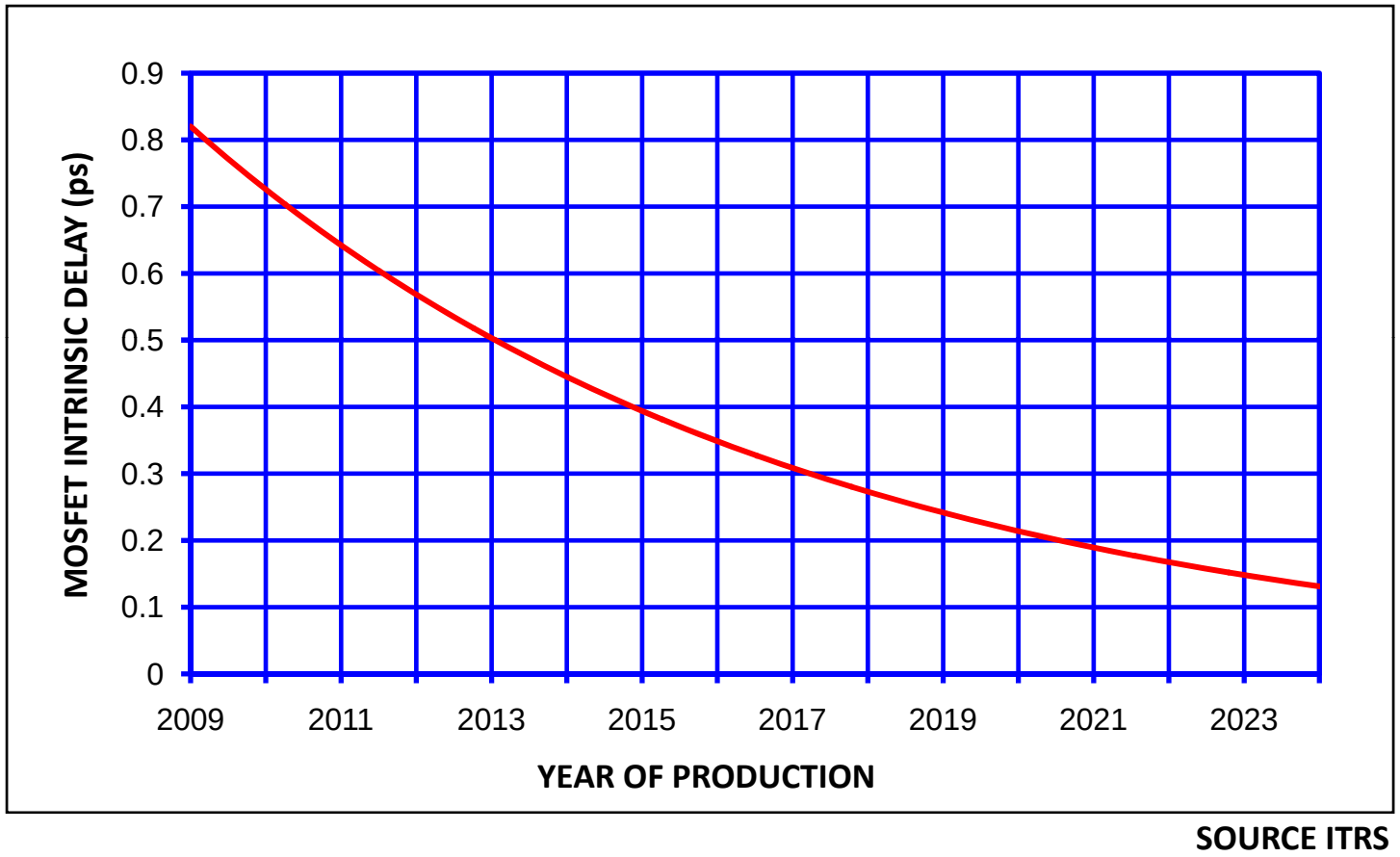
- mutually conflicting, improvement of one worsens some others
- for the best possible solution, one has to evaluate the relative importance of the constraints for the given application
- need to design the algorithms and architectures accordingly

technology scaling trend



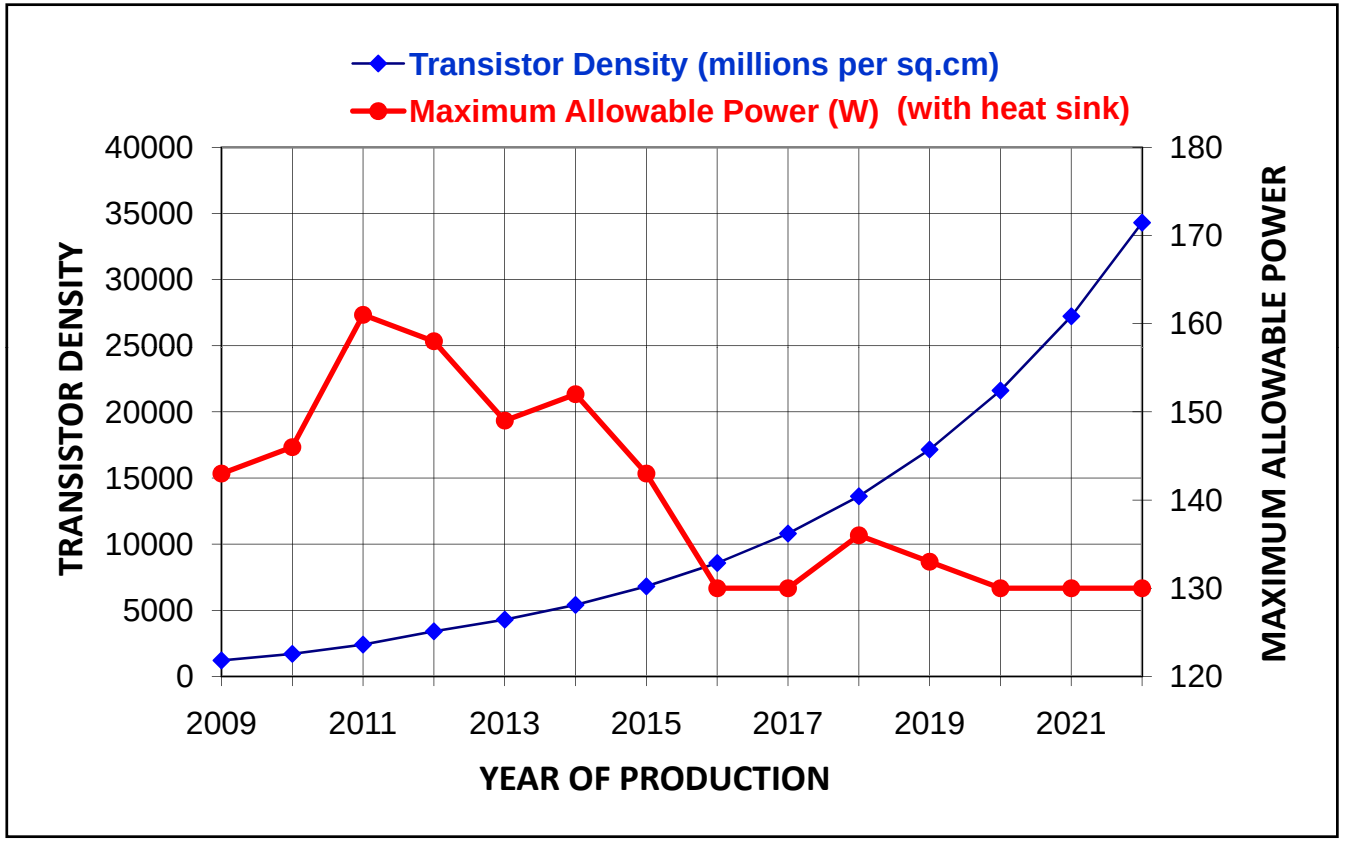
SOURCE ITRS

MOSFET intrinsic delay



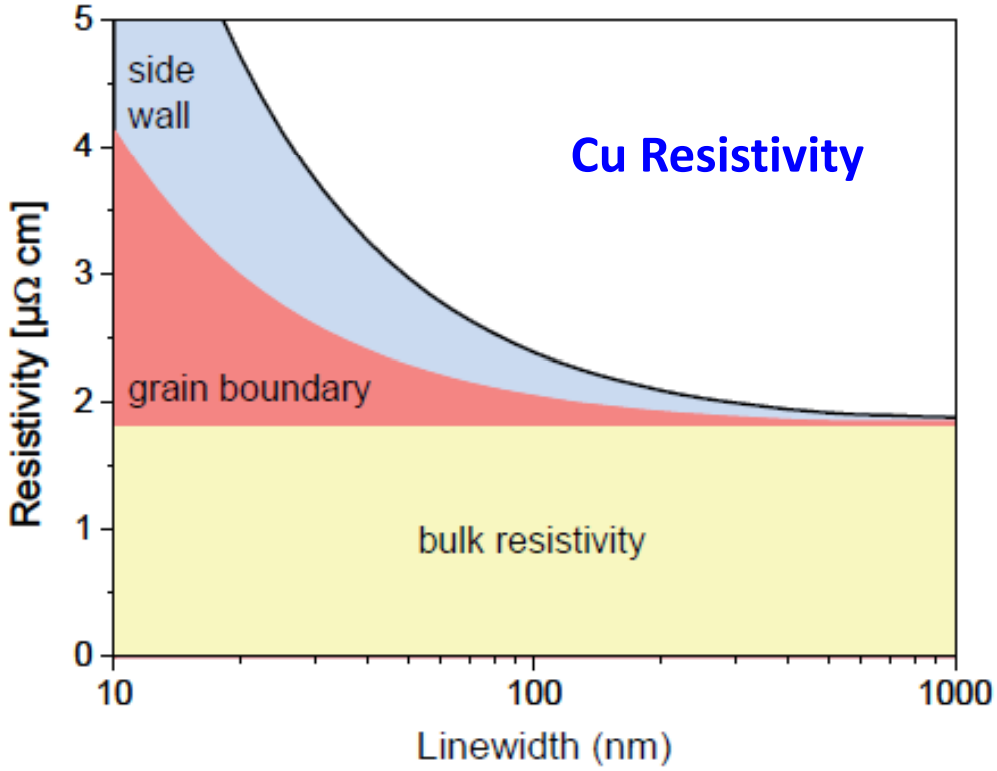
SOURCE ITRS

maximum power dissipation



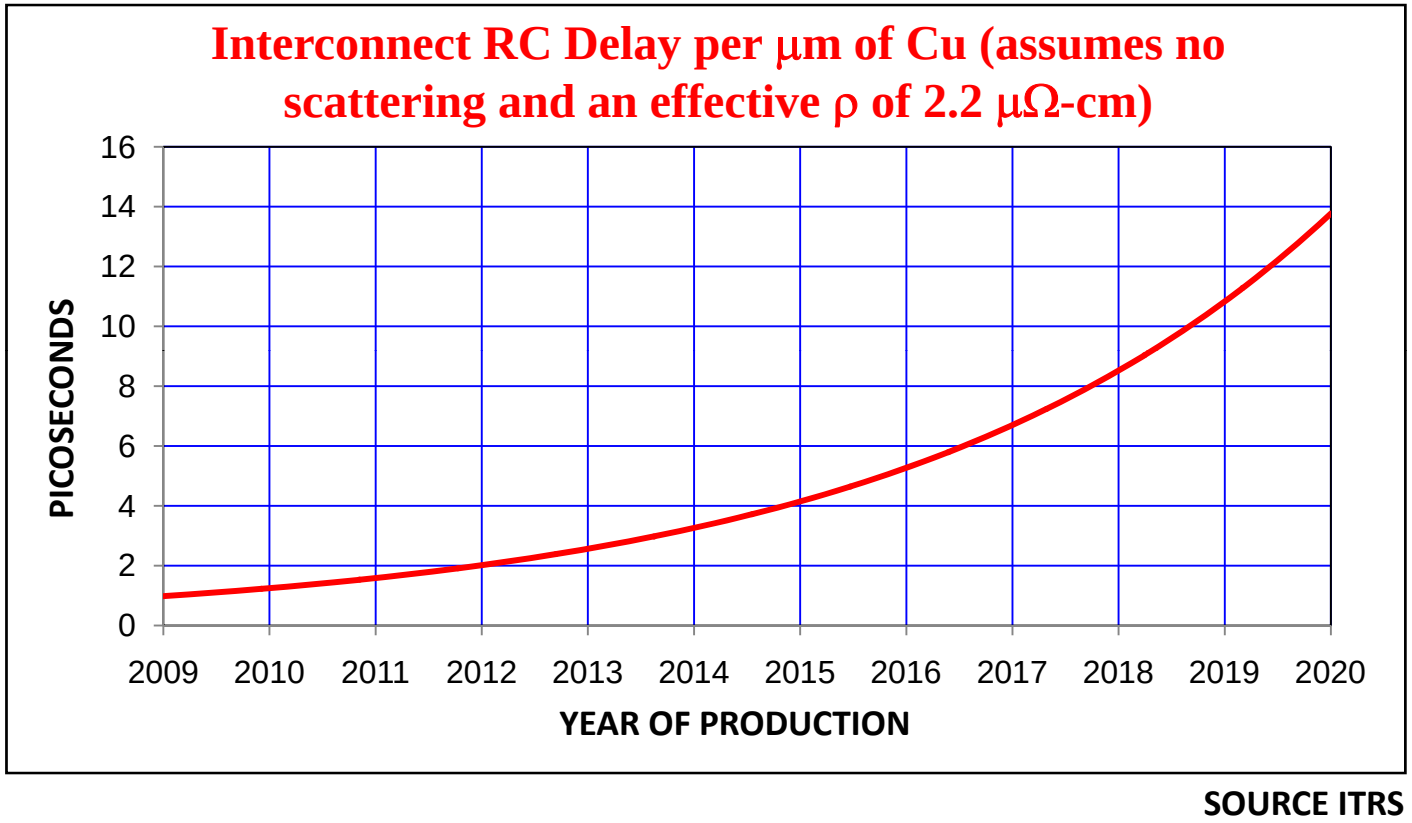
SOURCE ITRS

resistivity scaling



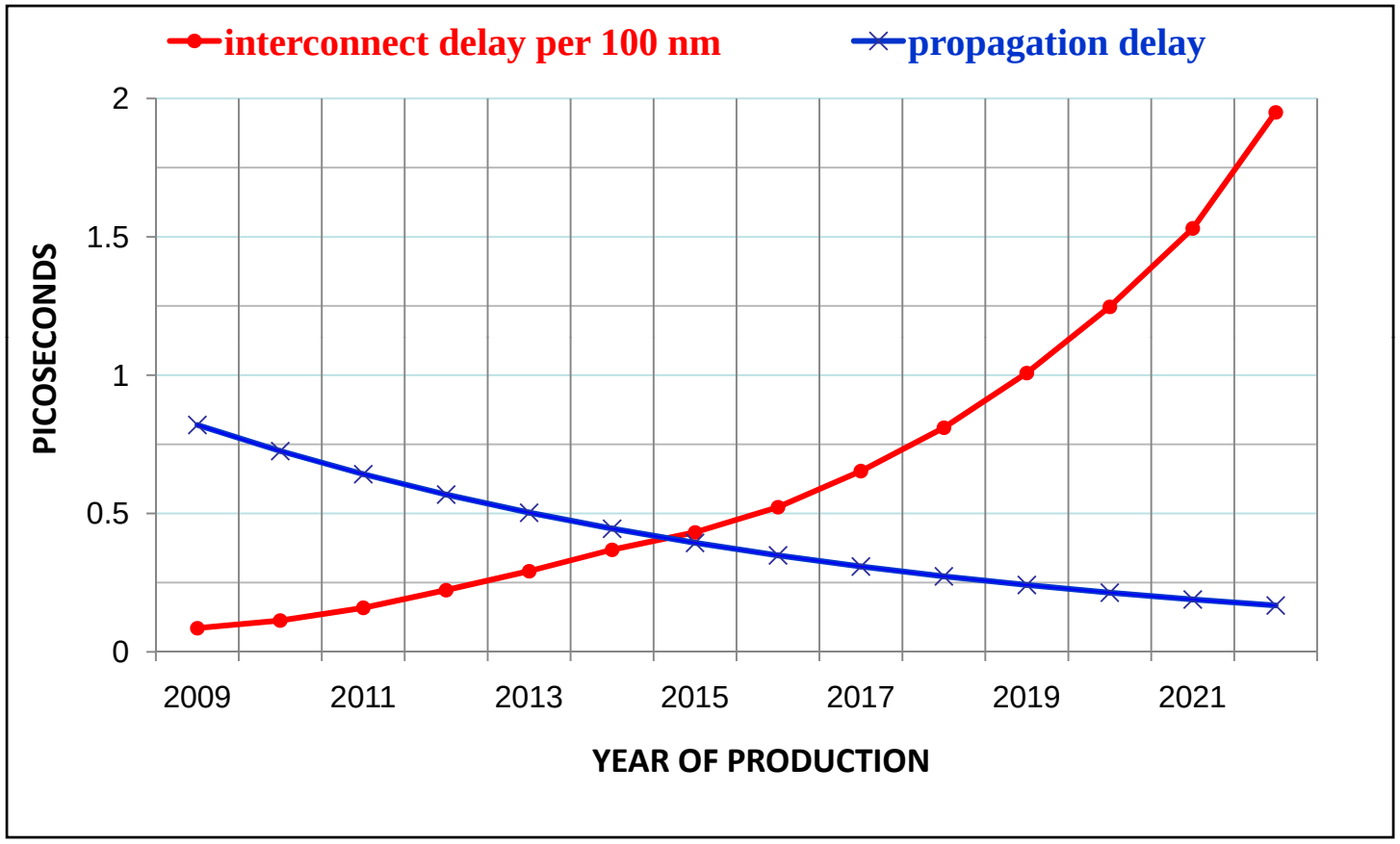
SOURCE: ITRS

interconnect delay

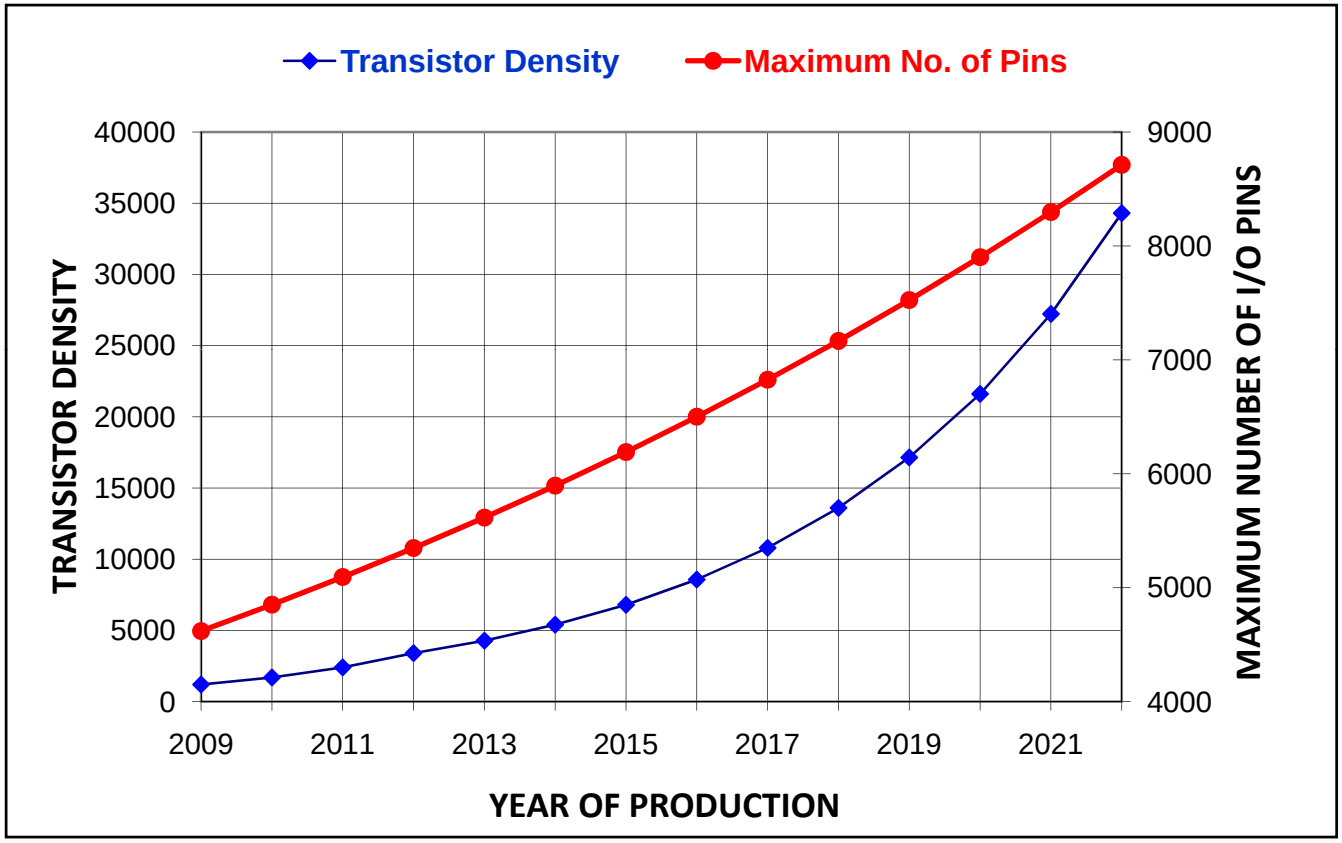


The *RC* delay increases quadratically with technology, reaching above **12ns** for a 1 mm Cu wire by 2020.

interconnect delay vs propagation delay



I/O limitations



SOURCE ITRS

analysis of scaling trend

- increasing transistor density, bigger chips, and more logic functionalities in a chip
- faster transistors and lower critical path
- limitation on maximum allowable power
- very high interconnect delay could be a show-stopper and interconnect power could be very high
- maximum number of I/O pins not even doubled in coming ten years while transistor density increased over thirty times during the same period

need of algorithm- architecture co-design

core operations in DSP

- Filtering: Finite Impulse Response (FIR), Infinite Impulse Response (IIR) filtering, and Adaptive filtering
- Matrix operations: convolution, correlation and matrix multiplication, inner-product computation
- Discrete transforms: DFT, DCT, DST, DHT, CZT, and DWT etc..

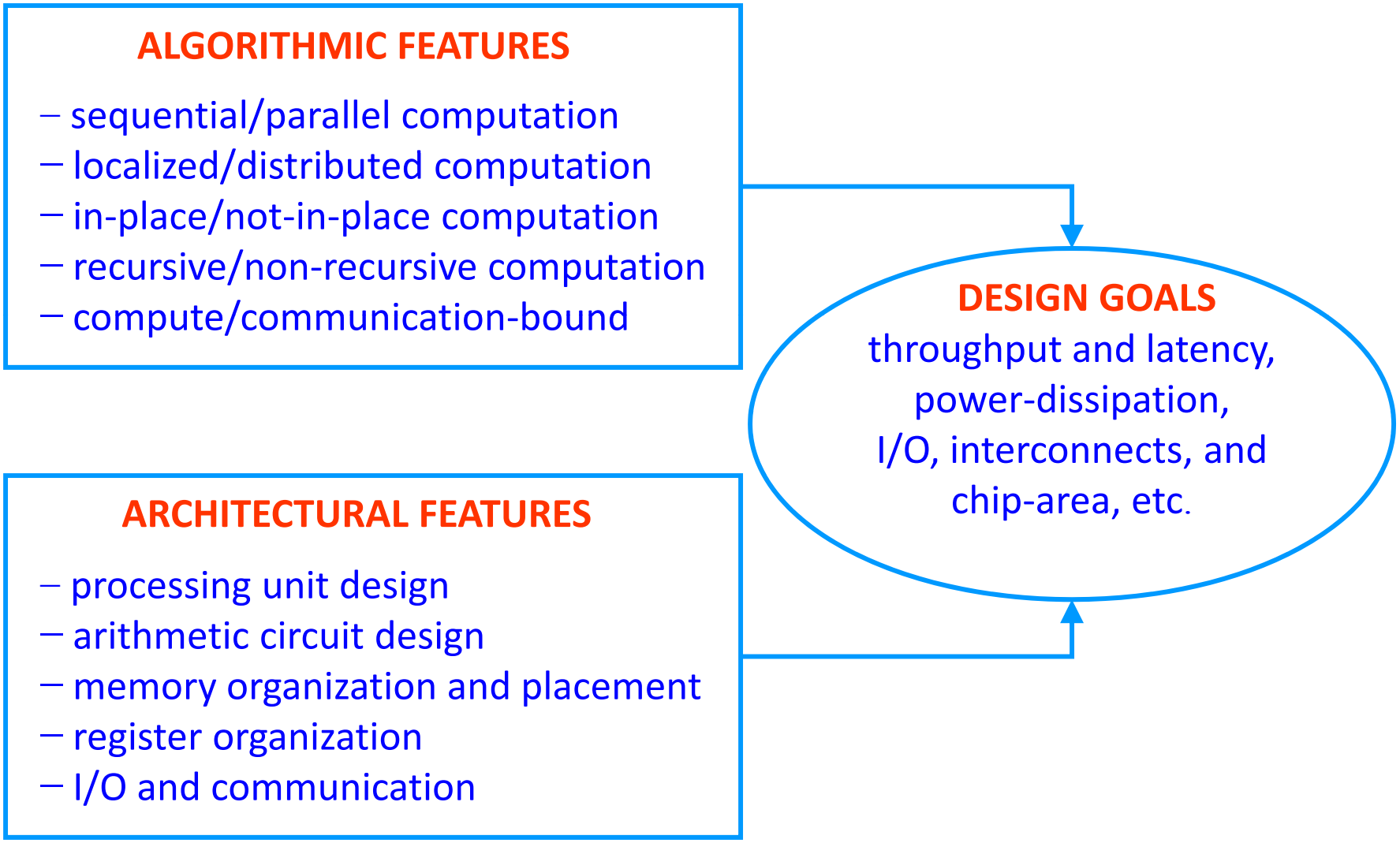
typical behavior of DSP operations

- computation-intensive
- repetitive multiply-accumulate computations
- trigonometric symmetries
- computational symmetries and redundancies
- multiplication with constant coefficients
- inner-product with a constant vector
- several implementation platforms
- dedicated architectures are often required

key design considerations

- processing time : Fast enough to process the input/ sample values as fast as they arrive at. Throughput and latency should match the requirement of real-time processing.
- power consumption: Key design issue, more demanding in portable and embedded systems.
- interconnects: Should be minimized for speed as well as power
- I/O : Number of I/Os should be minimized to reduce size and cost.
- area: It could be traded for speed or power provided that the cost allows.

scope of algorithm-architecture co-design



objectives of co-design

- design for catching with the speed
- design for dealing with power
- design for interconnect minimization
- design for matching computation with I/O
- design for demand-driven solution

catching up the speed

performance metrics: key design techniques

performance metrics

- computational bandwidth and throughput rate
- absolute and relative latencies
- execution time

design techniques

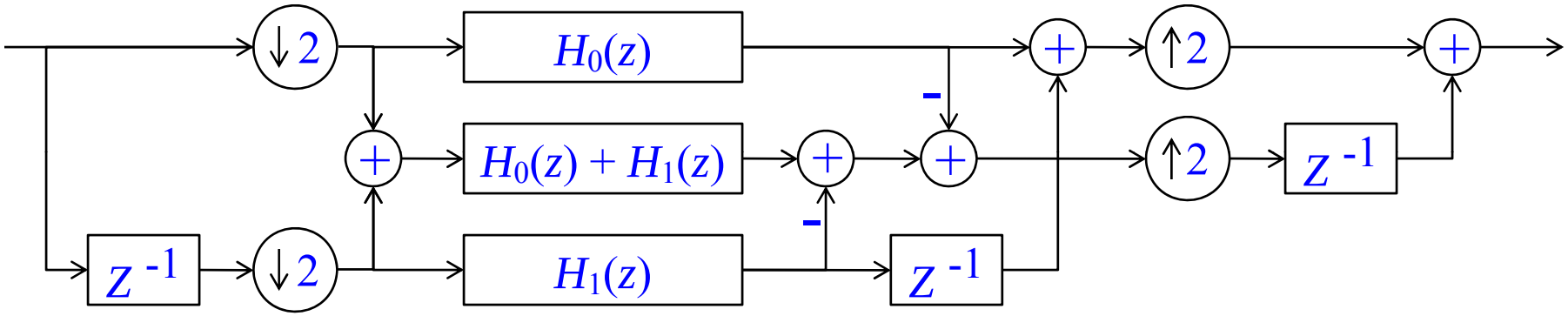
- minimization of computation and mapping computation to hardware architecture
- co-design for parallel processing and pipelining
- combination of parallel processing and pipelining

minimization of computations

- use of trigonometric symmetry: the famous Cooley–Tukey FFT algorithm and several other algorithms for discrete Fourier and other discrete transforms [Cooley 1995].
- polynomial operations and interpolation and minimization of redundant operations: Toom-Cook method and Agarwal-Cooley algorithm of digital convolution [Agarwal 1977].

design for parallel processing

multirate FIR filtering approach: Toom-Cook algorithm [Cook 1966] :



(Filter structure based on a two-point convolution algorithm.
H_0 and H_1 : the half-length filters of even and odd coefficient of filter H.)

block filter [Clark 1981] and transform domain adaptive filter [Narayan 1983] : to process a block of input in a cycle.

decomposition of computation [DA 2006, Conv 2008]: derives independent tasks and maps to parallel architecture

block formulation: parallel processing

Consider an FIR filter of order $N = 3$:

$$y(n) = a \cdot x(n) + b \cdot x(n-1) + c \cdot x(n-2)$$

This can be expressed in a block form [Lin 1996]

$$y(3k) = a \cdot x(3k) + b \cdot x(3k-1) + c \cdot x(3k-2)$$
$$y(3k+1) = a \cdot x(3k+1) + b \cdot x(3k) + c \cdot x(3k-1)$$
$$y(3k+2) = a \cdot x(3k+2) + b \cdot x(3k+1) + c \cdot x(3k)$$

Here the block size $L = 3$. Could similarly be done for longer block sizes.
The outputs are available are k-th cycle

design for pipelining

recursive algorithm formulation: mapping to systolic and systolic-like architecture for pipelining and reuse of data [DHT 1993] .

conversion to cyclic convolution form: DFT and other transforms could be converted to cyclic convolution form, and cyclic convolution could be mapped to pipelined structures [DFT 2006, DCT 2007, DST 2007].

cut-set retiming: graphical formulation to transform algorithm into pipeline form

latency, register complexity, and communication bottleneck on one hand and granularity of pipelining on the other hand.

convolutional representation of transforms

N-point sinusoidal transforms *e.g*, DFT/DCT/ DHT are given by [DFT 2006, DXT 2006, DHT 2007, DST 2007]

$$X(k) = \sum_{n=0}^{N-1} C(k,n) \cdot x(n), \text{ for } k = 0, 1, .., N-1$$

where the transform kernel is defined as

$$C(k,l) = \begin{cases} \cos(2\pi kn/N) - j\sin(2\pi kn/N), & \text{for DFT} \\ \cos(2\pi kn/N) + \sin(2\pi kn/N), & \text{for DHT} \\ \cos(\pi k(2n+1)/2N), & \text{for DCT.} \end{cases}$$

for prime values of N, the N x N kernel matrix is transformed to an $(N\text{-}1)$-point cyclic convolution.

conversion to cyclic convolution: example

N-point DFT:

$$X(0) = \sum_{n=0}^{N-1} x(n), \quad \text{and}$$

$$X(k) = x(0) + T(k), \quad \text{for } k = 1, 2, \cdots, N-1$$

$$T(k) = \sum_{n=1}^{N-1} x(n) \cdot e^{-j2\pi kn/N}.$$

For $N = 5$

4- point cyclic convolution

$$\begin{bmatrix} T(1) \\ T(2) \\ T(4) \\ T(3) \end{bmatrix} = \begin{bmatrix} h(1) & h(3) & h(4) & h(2) \\ h(2) & h(1) & h(3) & h(4) \\ h(4) & h(2) & h(1) & h(3) \\ h(3) & h(4) & h(2) & h(1) \end{bmatrix} \begin{bmatrix} x(1) \\ x(3) \\ x(4) \\ x(2) \end{bmatrix}$$

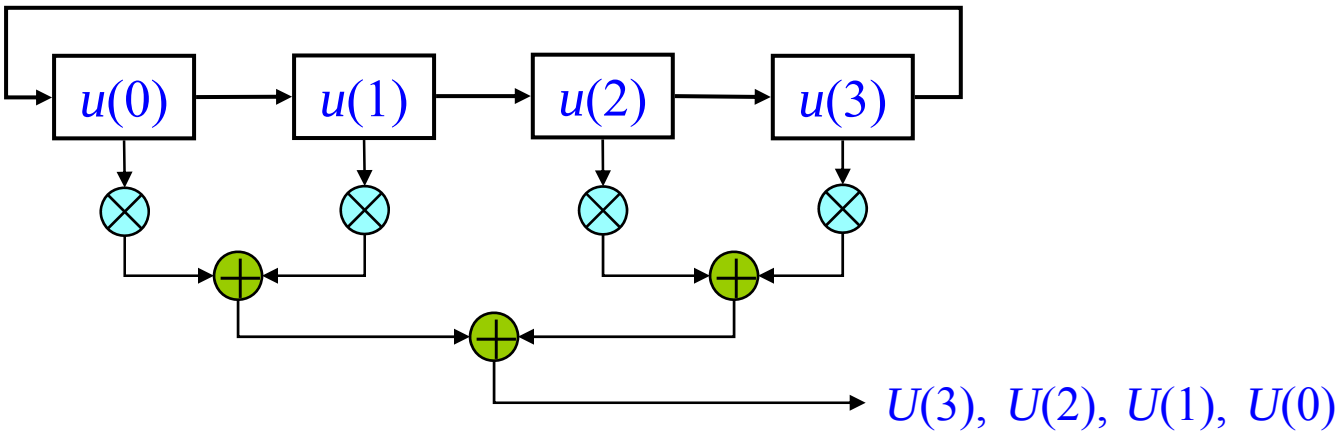
$$h(k) = e^{-j2\pi k/5} \quad \text{for } k = 1, 2, 3, 4$$

can be used for many other transforms and for other prime lengths N.

pipelined convolution: design example

implementation of 4-point cyclic convolution [DHT 2007]

$$\begin{bmatrix} U(0) \\ U(1) \\ U(2) \\ U(3) \end{bmatrix} = \begin{bmatrix} C(0) & C(3) & C(2) & C(1) \\ C(1) & C(0) & C(3) & C(2) \\ C(2) & C(1) & C(0) & C(3) \\ C(3) & C(2) & C(1) & C(0) \end{bmatrix} \begin{bmatrix} u(0) \\ u(1) \\ u(2) \\ u(3) \end{bmatrix}$$



dealing with power

components of power dissipation

CMOS power dissipation

1. Dynamic power dissipation : $P_D = \alpha . f . C . V_{dd}^2$

   α: average switching activity, f : operating frequency, C: total load capacitance of CMOS circuit, V_{dd} : supply voltage

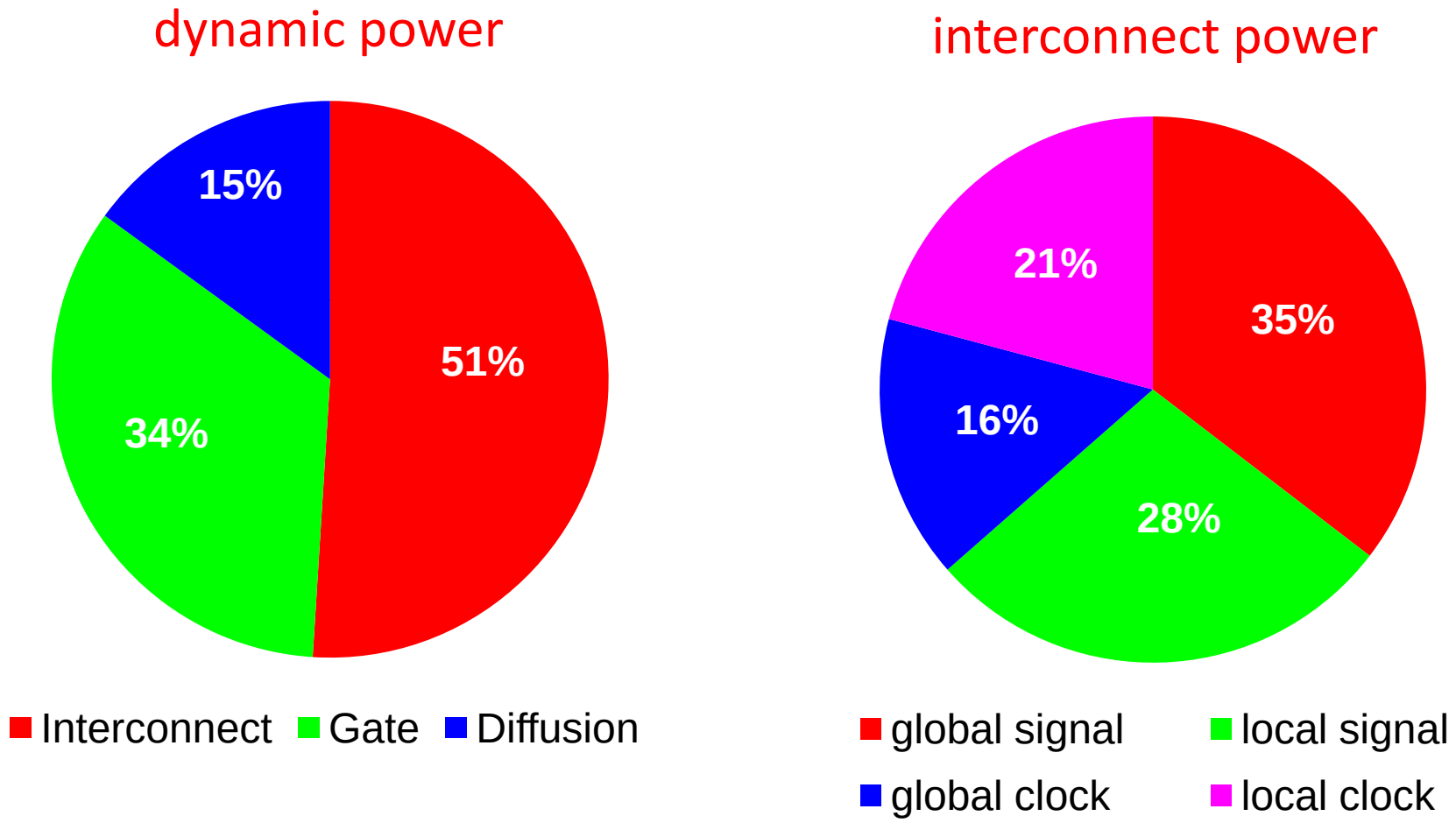
2. Leakage power: $P_{LEAK} = V_{dd} . I_{LEAK}$, I_{LEAK}: leakage current

3. Short-circuit power dissipation

Power dissipation across the conductors

1. power dissipation across clock network
2. power dissipation in interconnects

components of dynamic power dissipation



voltage scaling for dynamic power reduction

dynamic power: $P_D \propto f.C.V_{DD}^2$ could be reduced by reducing V_{DD}

propagation delay: $T_{PD} \propto \frac{C_{CHARGE} V_{DD}}{k(V_{DD} - V_{TH})^2}$

C_{CHARGE} : charging capacitance along the critical path.
V_{TH} : threshold voltage,

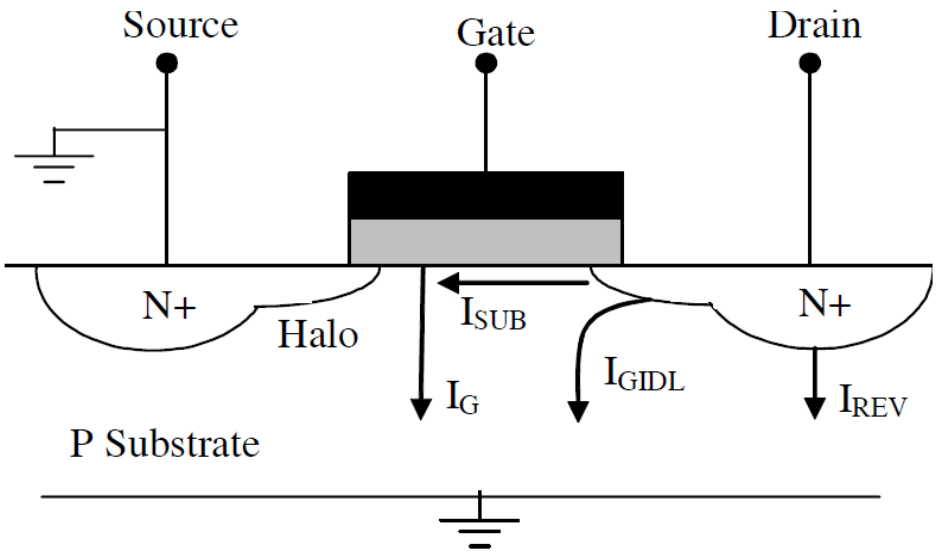
- Reduction of supply voltage leads to increase in propagation delay and hence the decrease in maximum usable frequency. But, throughput could still be maintained if we use parallel architecture.

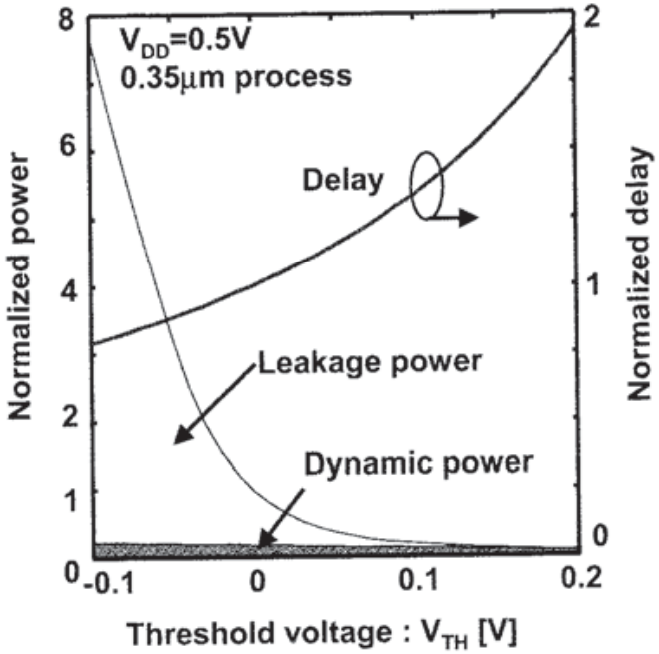
- Propagation delay could still be maintained in spite of reducing the supply voltage if C_{CHARGE} is reduced proportionately by pipelining [Parhi 1999]

leakage powers

standby leakage currents : when the circuit is in idle mode i.e., no computation takes place

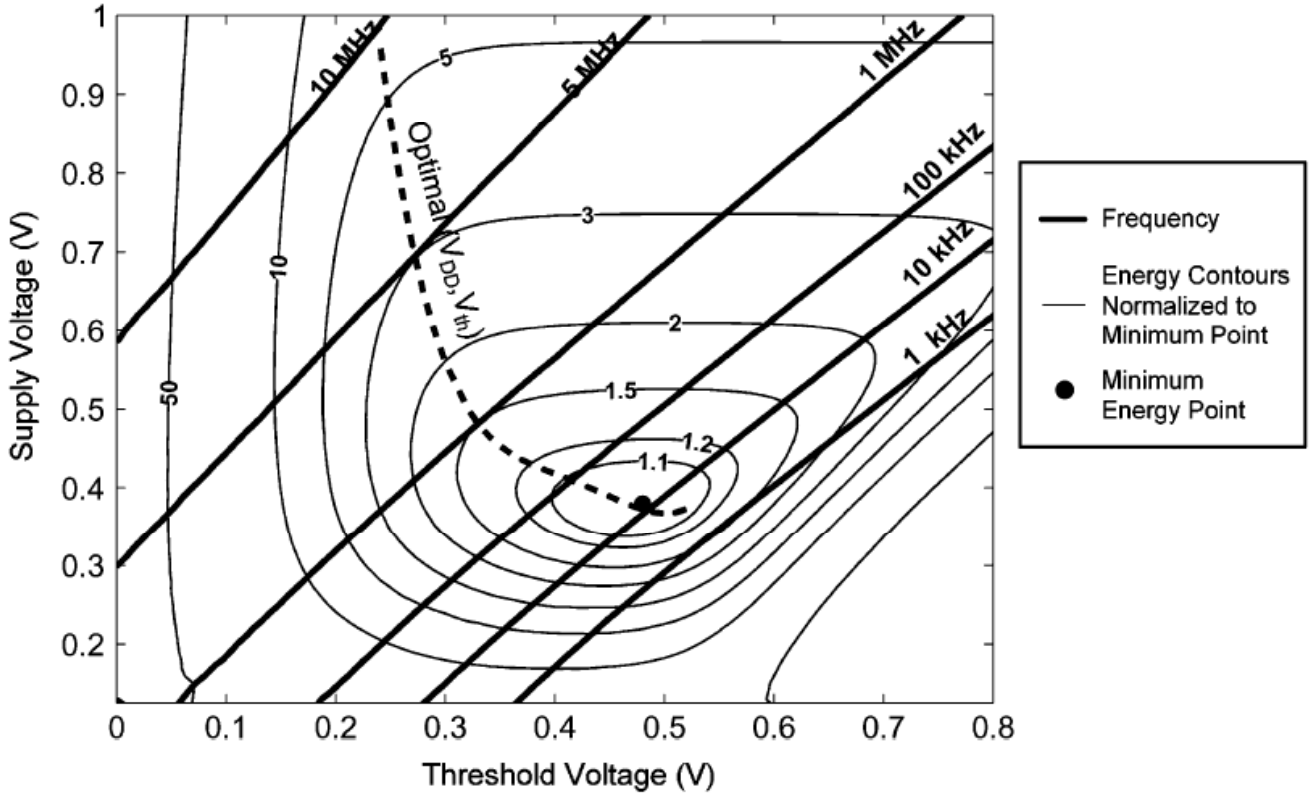
active leakage currents: when the circuit is in use





[Nose 2002]

total power minimization: design example



[Wang 2005]

interconnect minimization

recursive formulation

DFT of an N-point sequence $\{x(n)\}$ is given by

$$X(k) = \sum_{n=0}^{N-1} x(n) \cdot e^{-j2\pi kn/N}, \quad \text{for } k = 0, 1, .., N-1$$

It can be represented in a recursive form [DHT 1993, DFT 2D 2005]

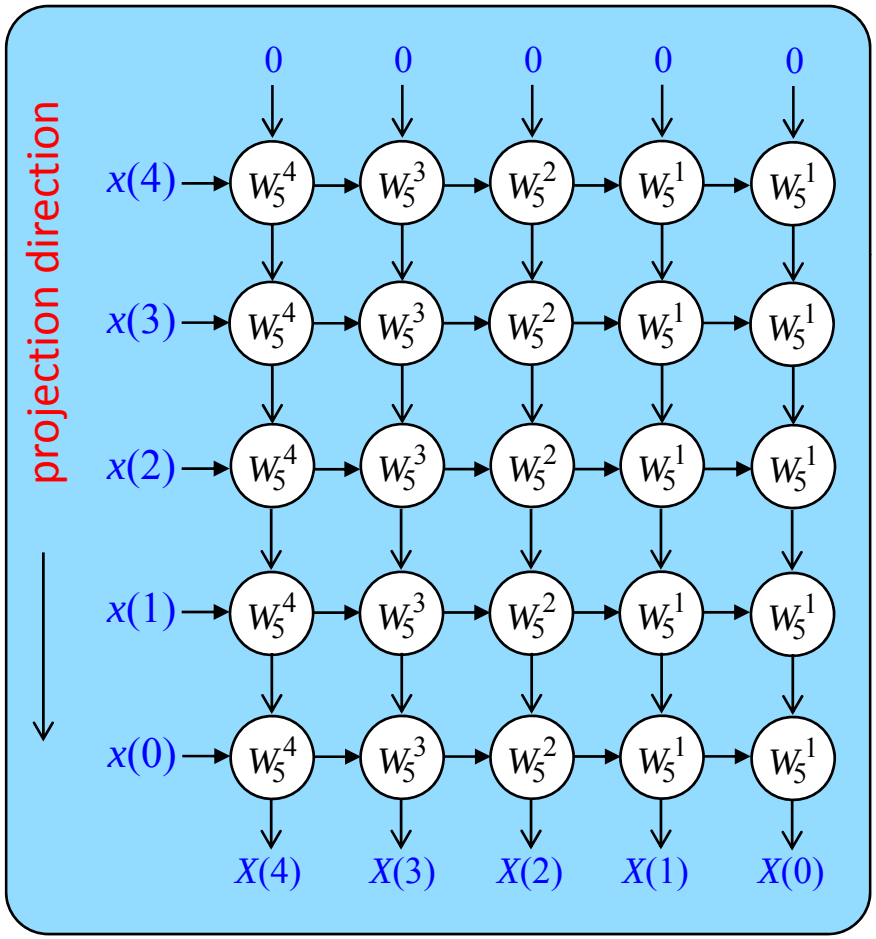
$$X(k) - \Big(\cdot \cdot \Big(\Big(x(N-1) W_N^k + x(N-1) \Big) W_N^k + x(N-2) \Big) W_N^k \cdot \cdot \\ \cdot \cdot + x(1) \Big) W_N^k + x(0)$$

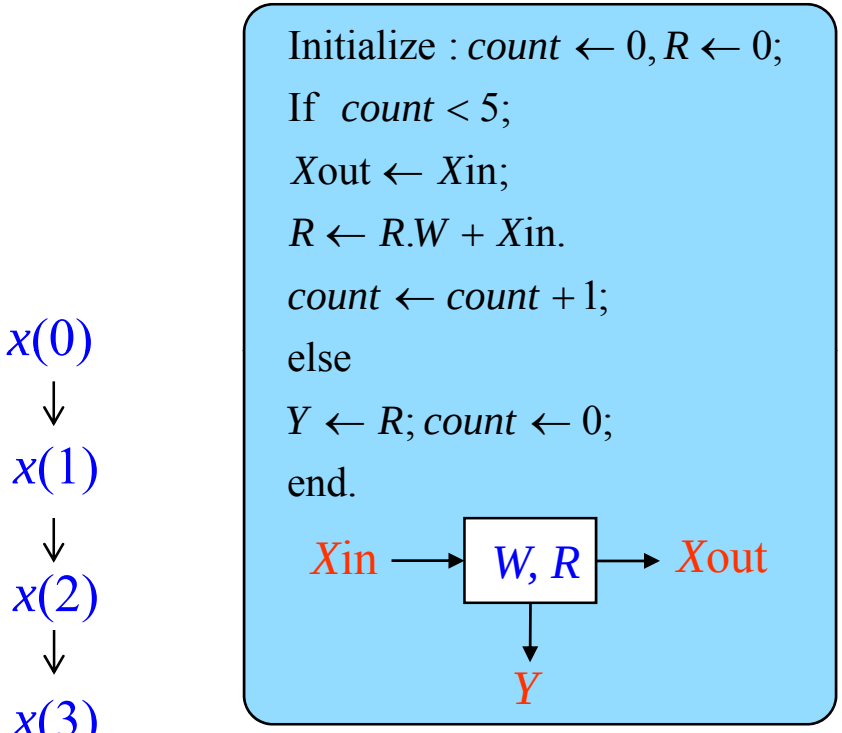
$$\text{where } W_N^k = e^{-j2\pi k/N}$$

Other sinusoidal transforms could also be expressed in this form.

recursive formulation: design example

Systolic design of DFT for N=5





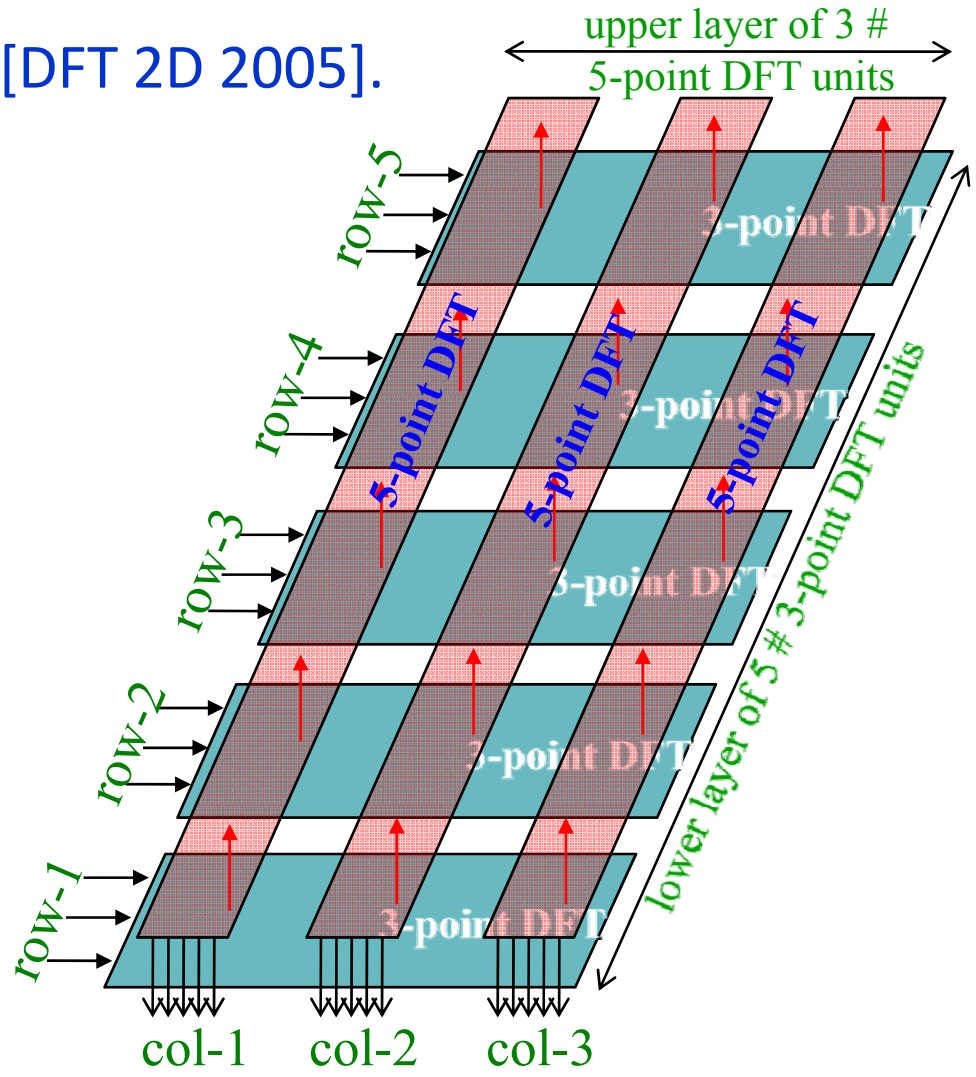
3-D designs: interconnect minimization

- Decompose the computation to multiple stages [DHT 2D 1993, DCT 1996].
- Map the computation of each part to a different layer
- The output of one layer is used as input of one of the adjacent layers
- The inner layers should preferably have less computation to perform to reduce power dissipation in in inner layers
- 3-D architectures may help to reduce the buffer space for intermediate data.

3-D architecture: design examples

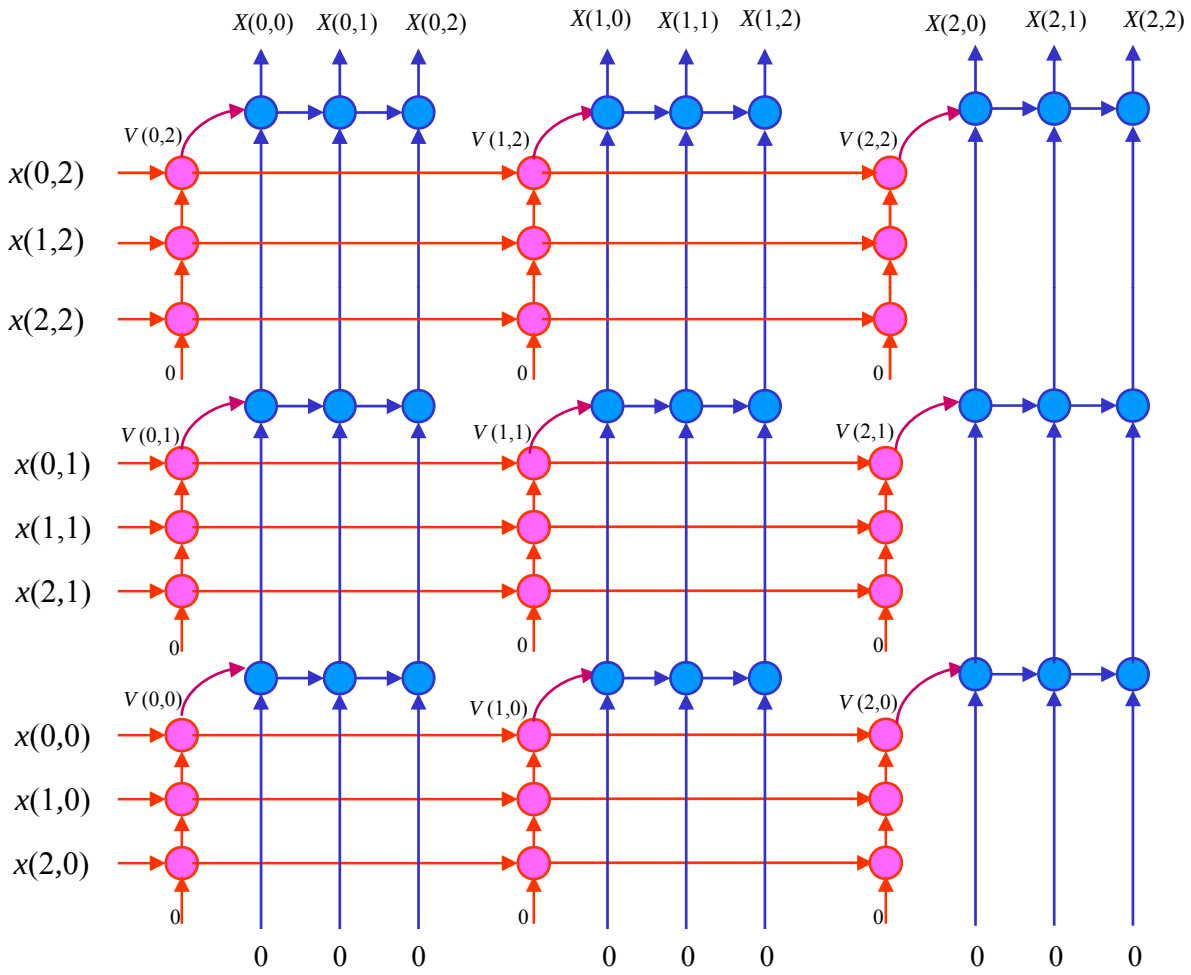
Computation of 15-point DFT [DFT 2D 2005].

- Use the prime-factor decomposition to compute the 15-point DFT in 2 stages.
- Map the input into a 5 x 3 array $[x(i, j)]$.
- In stage-1 compute 5 number of 3-point DFT of the rows of $[x(i, j)]$ to obtain a 5 x 3 intermediate result $[y(i, j)]$.
- In stage-2 compute 3 number of 5-point DFT of columns of $[y(i, j)]$ to obtain the DFT.



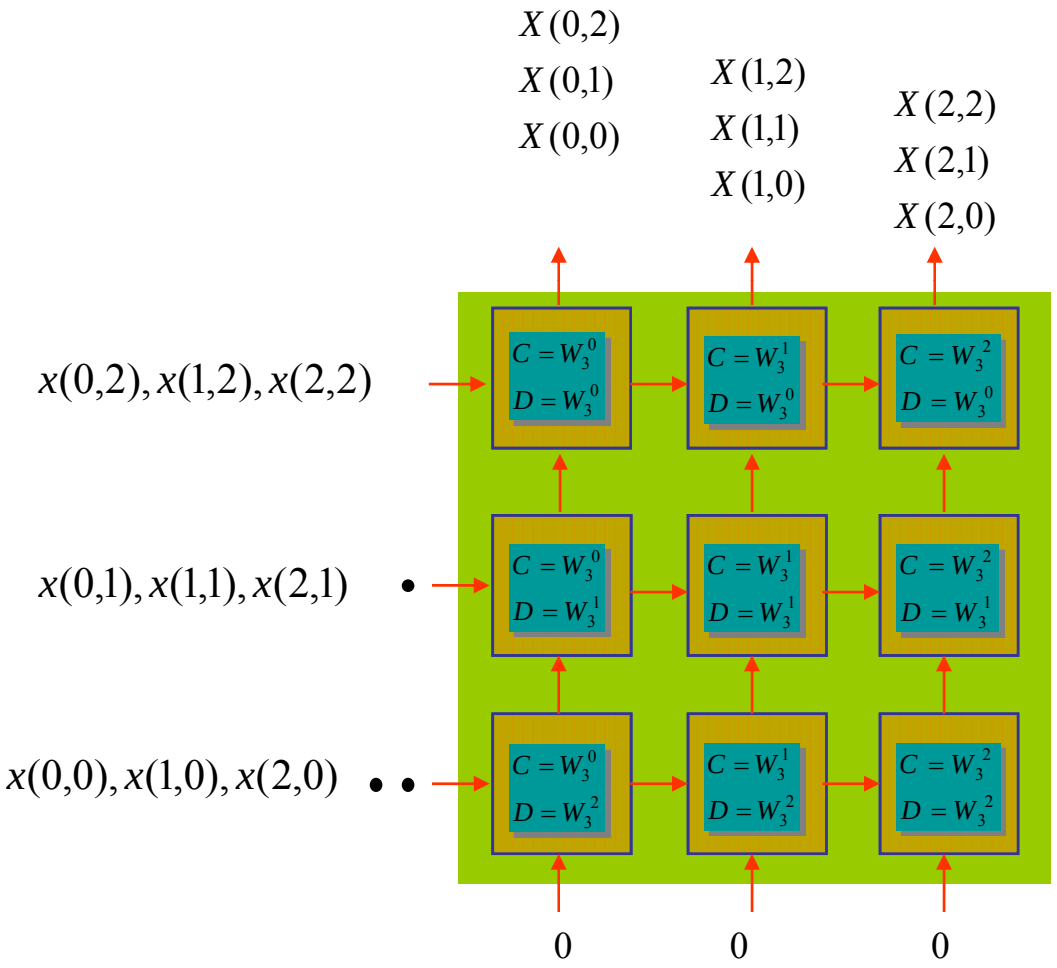
3-D architecture: design examples

Computation of 2-D DFT: Example 3x3 DFT



3-D architecture: design examples

Computation of 2-D DFT: Example 3x3 DFT



$(k+1)$th PE of $(i+1)$th row is *initialized* as:

$C \leftarrow W_N^k;\ D \leftarrow W_N^i;\ Z_1 \leftarrow 0;$

During every cycle period each PE performs:

$X_{out} \leftarrow X_{in};$

$Z_1 \leftarrow X_{in} + C.Z_1;$

$Y_{out} \leftarrow Z_2 + Y_{in};$

$Z_2 \leftarrow D.Z_2$

$COUNT \leftarrow COUNT + 1;$

If $COUNT = 3$ *Then*

$Z_2 \leftarrow Z_1;$

$Z_1 \leftarrow 0;$

Endif

matching computation with I/O

decomposition of computation

Matrix vector product $\mathbf{Y} = \mathbf{CX}$: Let $\mathbf{C}$ be of size 4 x 4. $\mathbf{Y}$ and $\mathbf{X}$ be of size 3 x 1. [DA 2006, FIR 2008]

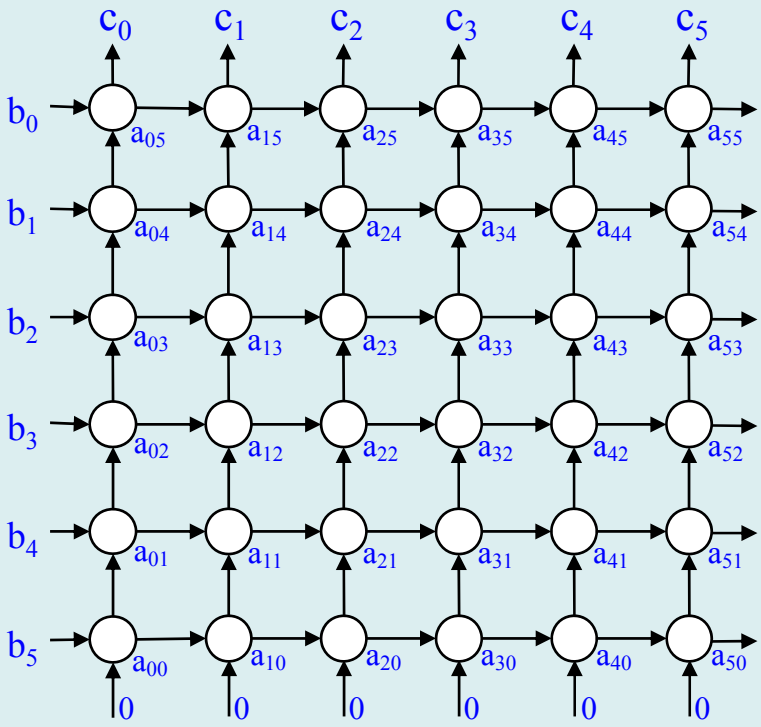
$$\begin{bmatrix} Y(0) \\ Y(1) \\ Y(2) \\ Y(3) \end{bmatrix} = \begin{bmatrix} C(0,0) & C(0,1) & C(0,2) & C(0,3) \\ C(1,0) & C(1,1) & C(1,2) & C(1,3) \\ C(2,0) & C(2,1) & C(2,2) & C(2,3) \\ C(3,0) & C(3,1) & C(3,2) & C(3,3) \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \\ x(2) \\ x(3) \end{bmatrix} \Rightarrow \begin{aligned} Y(0) &= Y1(0) + Y2(0), \\ Y(1) &= Y1(1) + Y2(1), \\ Y(2) &= Y1(2) + Y2(2), \\ Y(3) &= Y1(3) + Y2(3). \end{aligned}$$

$$\begin{bmatrix} Y1(0) \\ Y1(1) \end{bmatrix} = \begin{bmatrix} C(0,0) & C(0,1) \\ C(1,0) & C(1,1) \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \end{bmatrix} \qquad \begin{bmatrix} Y2(0) \\ Y2(1) \end{bmatrix} = \begin{bmatrix} C(0,2) & C(0,3) \\ C(1,2) & C(1,3) \end{bmatrix} \begin{bmatrix} x(2) \\ x(3) \end{bmatrix}$$

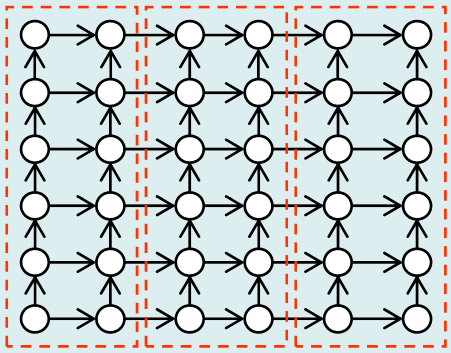
$$\begin{bmatrix} Y1(2) \\ Y1(3) \end{bmatrix} = \begin{bmatrix} C(2,0) & C(2,1) \\ C(3,0) & C(3,1) \end{bmatrix} \begin{bmatrix} x(0) \\ x(1) \end{bmatrix} \qquad \begin{bmatrix} Y2(2) \\ Y2(3) \end{bmatrix} = \begin{bmatrix} C(2,2) & C(2,3) \\ C(3,2) & C(3,3) \end{bmatrix} \begin{bmatrix} x(2) \\ x(3) \end{bmatrix}$$

decomposition by graph partitioning

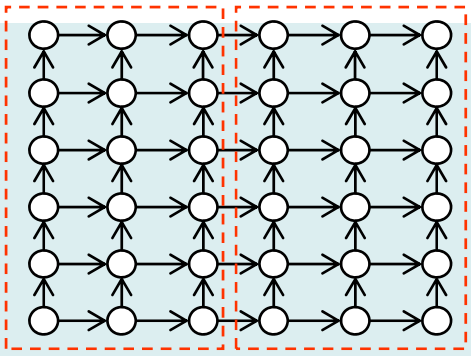
Computation of larger size problem by small array and less I/Os



DG for vector-matrix multiplication **c=Ab**.
A is a 6 x 6 matrix and **b** is 6-point column vector.



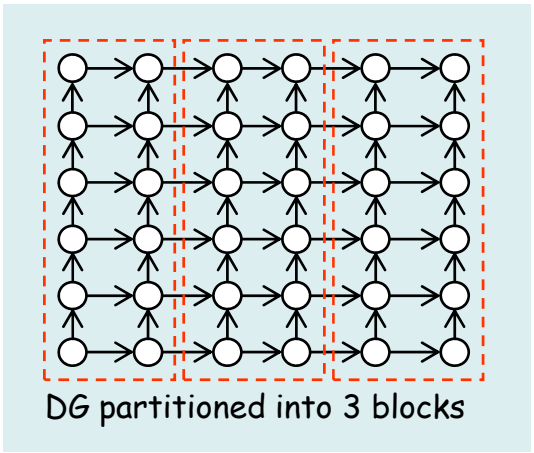
DG partitioned into 3 blocks

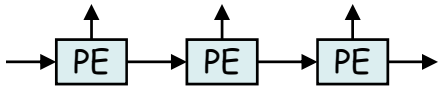


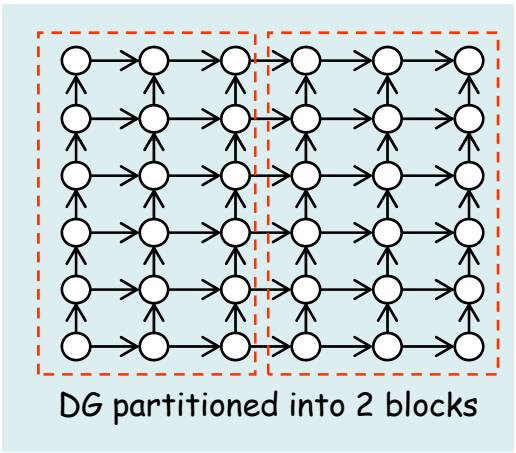
DG partitioned into 2 blocks

graph partitioning: example

Two methods: (*i*) Locally Sequential Globally Parallel (LSGP) (*ii*) Locally Parallel Globally Sequential (LPGS) methods







LSGP: One block of nodes are mapped to one PE. 3 PEs of the array are used for three blocks of the DG. All the blocks are processed concurrently.

LPGS: Each block of nodes are mapped to the whole array. 3 columns of node in a block are mapped to 3PEs of the array. Blocks are processed sequentially.

design for demand-driven solution

iterative circular optimization

- system level design, optimization, and trade-off
- algorithm design for minimization of interconnect and matching with I/O, and modify for speed and power.
- mapping algorithm to architecture: exploring 3-D design, local communication, parallel, pipeline, or a combination
- mapping architectures to circuits: selection of arithmetic circuits [FIR 2010]
- circular iterative approach: top-down to bottom-up followed by bottom-up by demand-driven trade-off [DWT 2010]

system-level design and optimization

- HW/SW partitioning
- synchronous /asynchronous partitioning: globally asynchronous and locally synchronous design
- data representation: Selection of bit-width, sign-magnitude or 2's complement representation, and data encoding
- power management schemes: dynamic voltage scaling, adaptive clocking, and clock gating

conclusions

conclusions

- Over the years, the application space of DSP has become wide and diverse.
- The complexity of DSP algorithms also has grown high.
- Dedicated architectures are needed for real-time implementation of DSP functionalities.
- Algorithms for dedicated architectures are tailored to meet a target set of design metrics and trade-offs.
- Typical behaviors of the DSP algorithms are utilized to perform reformulation of algorithm for mapping that to a hardware architecture.

conclusions

- We reviewed the scaling trends to explore the key design challenges
- Algorithms and architectures need to be designed to have just enough speed
- Surplus speed should be traded for power
- Algorithm and architecture should minimizes the interconnect length and match computation with I/O bandwidth
- Co-design should ultimately satisfy the constraints and demands

references

[Agarwal 1977] R. Agarwal and J. W. Cooley, New algorithms for digital convolution, IEEE Trans ASSP, Oct 1977, pp. 392-410.

[Clark 1981] G. Clark, S. Mitra, and S. Parker, Block implementation of adaptive digital filters, Processing, IEEE Trans ASSP, June 1981, pp.744-752.

[Conv 2008] P. K. Meher, 'Parallel and Pipelined Architectures for Cyclic Convolution by Block Circulant Formulation using Low-Complexity Short-Length Algorithms,' IEEE Trans. on Circuits & Systems for Video Technology, pp. 1422-1431, October 2008.

[Cooley 1995] J. W. Cooley and J. W. Tukey, An Algorithm for the Machine Calculation of Complex Fourier Series, Mathematics of Computation, vol. 19, no. 90, April 1965, pp. 297-301.

[Cook 1966] S. A. Cook., On the minimum computation time of functions. Harvard Thesis, 1966.

[**DA 2006**] P. K. Meher, 'Hardware-Efficient Systolization of DA-based Calculation of Finite Digital Convolution,' IEEE Trans Circuits & Systems-II, pp.707-711, Aug 2006.

references

[DCT 1996] S. S. Nayak and P. K. Meher, 'A 3-Dimensional Systolic Architecture for Parallel VLSI Implementation of the Discrete Cosine Transform', IEE Proceeding ~ Circuits, Devices and Systems, vol.143, no.5, pp.255-258, Oct 1996.

[DCT 2006] P. K. Meher, 'Systolic Designs for DCT using a Low-Complexity Concurrent Convolutional Formulation,' IEEE Trans Circuits & Systems for Video Technology, September 2006, pp.1041-1050.

[DFT 2D 2005] P. K. Meher, 'Design of a Fully-Pipelined Systolic Array for Flexible Transposition-Free VLSI of 2-D DFT', IEEE Trans Circuits and Systems-II, Feb 2005.

[DFT 2006] P. K. Meher, 'Efficient Systolic Implementation of DFT using a Low-Complexity Convolution-like Formulation,' IEEE Trans Circuits & Systems-II, vol.53, no.8, pp.702-706, August 2006.

[DHT 1993] P. K. Meher, J. K. Satapathy and G. Panda, 'Efficient Systolic Solution for a New Prime-Factor Discrete Hartley Transform Algorithm', IEE Proceedings ~ Circuits, Devices and Systems, vol. 140, no.2, pp. 135-139, April 1993.

[DHT 2D 1993] P. K. Meher and G. Panda, 'Novel Recursive Algorithm and Highly Compact Semi-Systolic Architecture for High-Throughput Computation of 2-D DHT', IEE Electronics Letters, pp. 883-885, May 1993.

references

[DHT 2007] P. K. Meher, J. C. Patra and M. N. S. Swamy, 'High-Throughput Memory-Based Architecture for DHT Using a New Convolutional Formulation,' IEEE Trans. on Circuits & Systems-II, vol.54, no.7, pp.606-610, July 2007.

[DST 2007] P. K. Meher and M. N. S. Swamy, 'New Systolic Algorithm and Array Architecture for Prime-Length Discrete Sine Transform,' IEEE Trans Circuits & Systems-II, vol.54, no.2, pp.262-266, March 2007.

[DXT 2006] P. K. Meher, 'Unified Systolic-Like Architecture for DCT and DST Using Distributed Arithmetic,' IEEE Trans Circuits & Systems-I, vol.53, no.12, pp.2656-2663, December 2006.

[DWT 2010] B. K. Mohanty and P. K. Meher, 'Parallel and Pipeline Architectures for High-Throughput Computation of Multilevel 3-D DWT,'IEEE Trans. on Circuits & Systems for Video Technology, pp.1200-1209, September 2010.

[FIR 2008] P. K. Meher, S. Chandrasekaran, and A. Amira, 'FPGA Realization of FIR Filters by Efficient and Flexible Systolization Using Distributed Arithmetic,' IEEE Trans Signal Processing, pp. 3009-3017, July 2008.

references

[FIR 2010] P. K. Meher, 'New Approach to Look-up-Table Design and Memory-Based Realization of FIR Digital Filter,' IEEE Trans Circuits & Systems-I, pp.592-603, March 2010.

[Lin 1996] Ing-Song Lin and S. K. Mitra, 'Overlapped block digital filtering,' Circuits and Systems II August 1996, pp. 586 – 596.

[Narayan 1983] S.S. Narayan, A. Peterson, M. Narasimha, Transform domain LMS algorithm , IEEE Trans ASSP, 1983 , pp. 609 – 615.

[Nose 2002] K.Nose, M.Hirabayashi, H.Kawaguchi, S.Lee, and T.Sakurai, VTH-hopping scheme to reduce subthreshold leakage for low-power processors, IEEE Journal of Solid-State Circuits, 2002, pp. 413 – 419.

[Parhi 1999] K. K. Parhi, VLSI Digital Signal Processing Systems: Design and Implementation. New York: John Wiley & Sons, Inc, 1999.

[Wang 2005] A. Wang, A. Chandrakasan, A 180-mV subthreshold FFT processor using a minimum energy design methodology, IEEE Journal of Solid-State Circuits, 2005, pp. 310-319

Thank You !