Lecture 3

FIR Design and Decision Feedback Equalization

Mark Horowitz
Computer Systems Laboratory
Stanford University
horowitz@stanford.edu

Readings

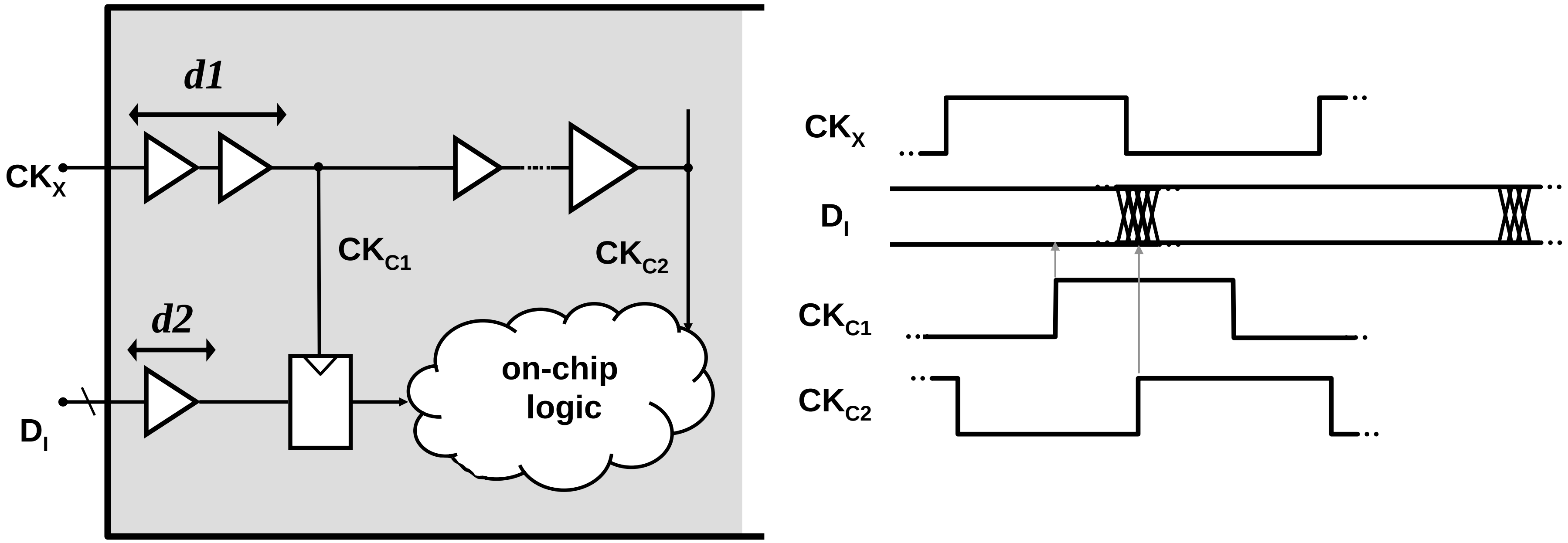
- Readings (for next lecture on adders)
  - Chandrakasan Chapter 10.1-10.2.10
  - Harris Taxonomy of adders (either paper on web or WH 10.2 to 10.2.2.9
- Overview:
  - Finish up some timing issues from high-speed links
  - Your project will be the design of a decision feedback equalizer, but most of the hardware will be the same as a normal FIR filter. So the lecture will start talking about FIR filter design, and then will go into the added issues with building a DFE. WARNING: I am not an expert in this area, so there might be better ideas out there (and some bugs in these notes)
  - The FIR notes are from Bora Nikolic at UCB.

I/O Clocking Issues

- Remember the clocking issues:
  - Long path constraint (setup time)
  - Short path constraint (hold time)
  - Need to worry about them for I/O as well

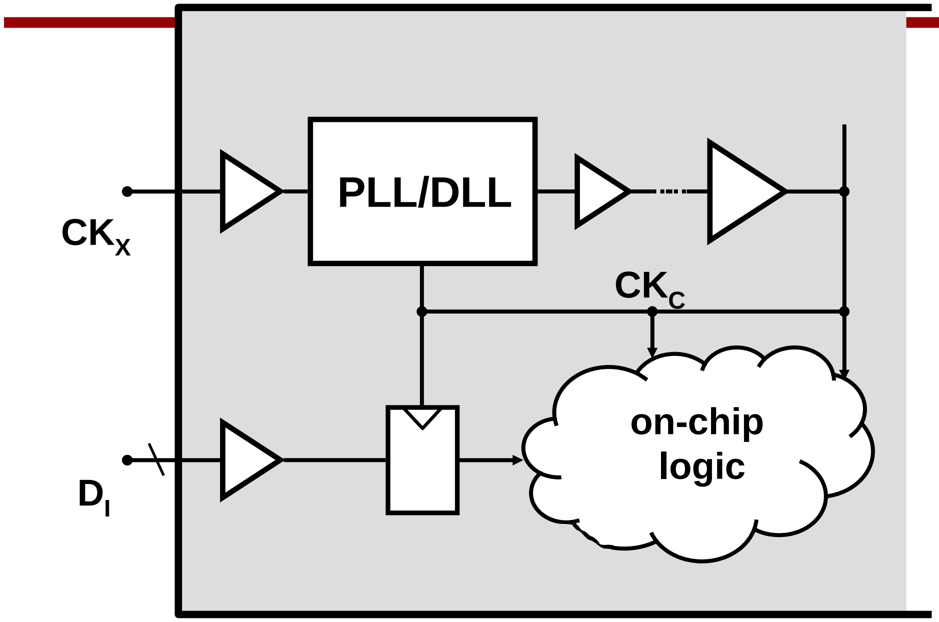
- For I/O need to worry about a number of delays
  - Clock skew between chips
  - Data delay between chips
    - Can be larger than a clock cycle (speed of light)
  - Clock skew between external clock and internal clock
    - This can be very large if not compensated
    - It is essentially the insertion delay of the clock tree

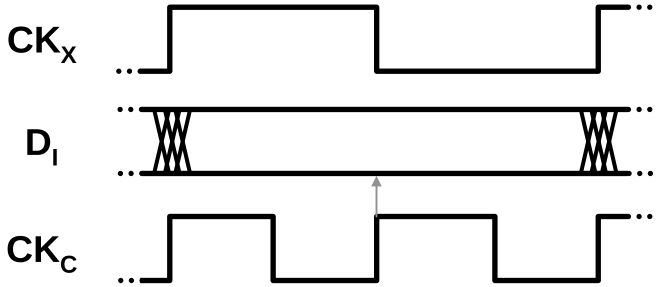
System Clocking: Simple Synchronous Systems



- Long bit times compared to on chip delays:
  - Rely on buffer delays to achieve adequate timing margin

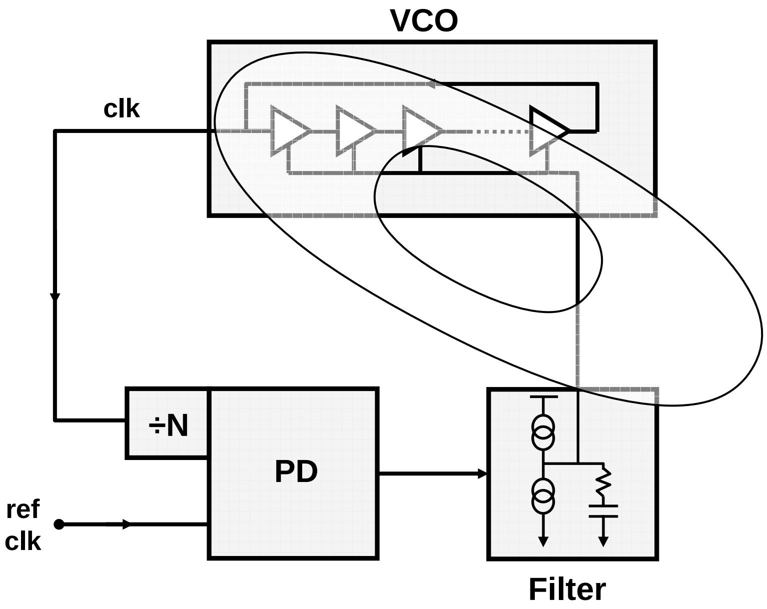
PLLs: Creating Zero Delay Buffers



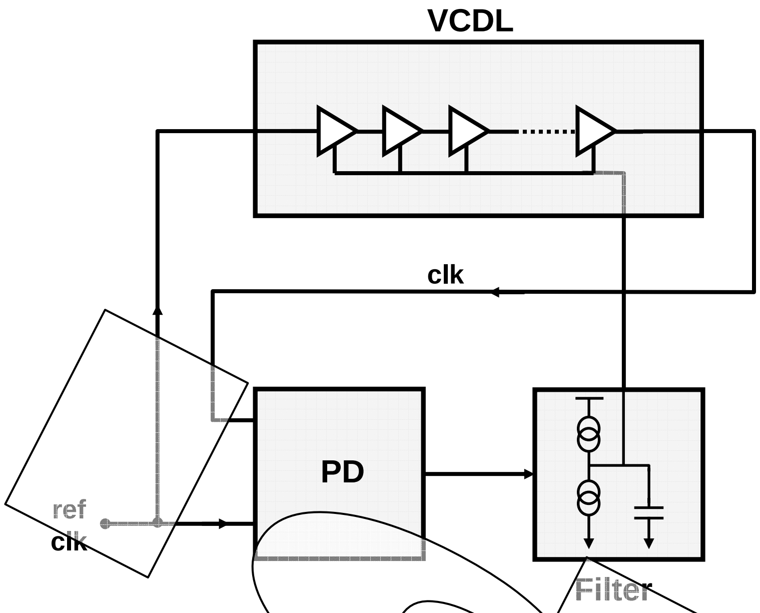


- On-chip clock might be a multiple of system clock:
  - – Synthesize on-chip clock frequency
- On-chip buffer delays do not match
  - – Cancel clock buffer delay

Used to Argue About PLLs vs DLLs



- Second/third order loop:
  - Stability is an issue
  - Frequency synthesis easy
  - Ref. Clk jitter gets filtered
  - Phase error accumulates



- First order loop:
  - Stability guaranteed
  - Frequency synthesis problematic
  - Ref. Clk jitter propagates
  - Phase error does not accumulate

After Many Years of Research

- And many papers and products
- One can mess up either a DLL or PLL
  - Each has it own strengths and weaknesses
- If designed correctly, either will work well
  - Jitter will be dominated by other sources
- Many good designs have been published
  - It is now a building block that is often reused
  - We all have our favorites, mine is the dual-loop design

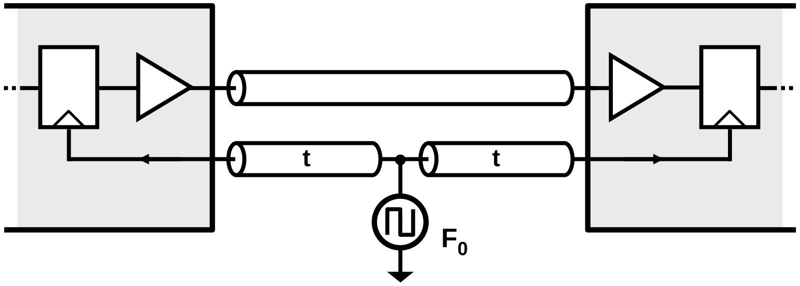
- And yes, people use ring oscillators
  - Still an open question about how much LC helps (in system)

Clocking Structures

- Synchronous:

  Same frequency and phase

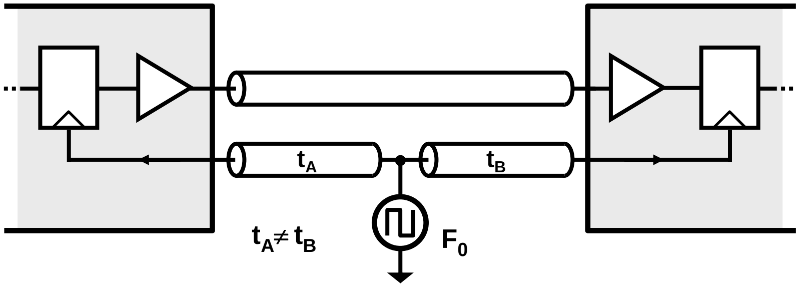
  - Conventional buses



- Mesochronous

  Same frequency, unknown phase

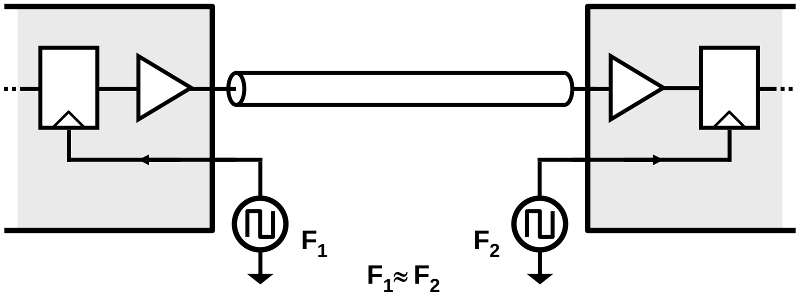
  - Fast memories
  - Internal system interfaces
  - MAC/Packet interfaces



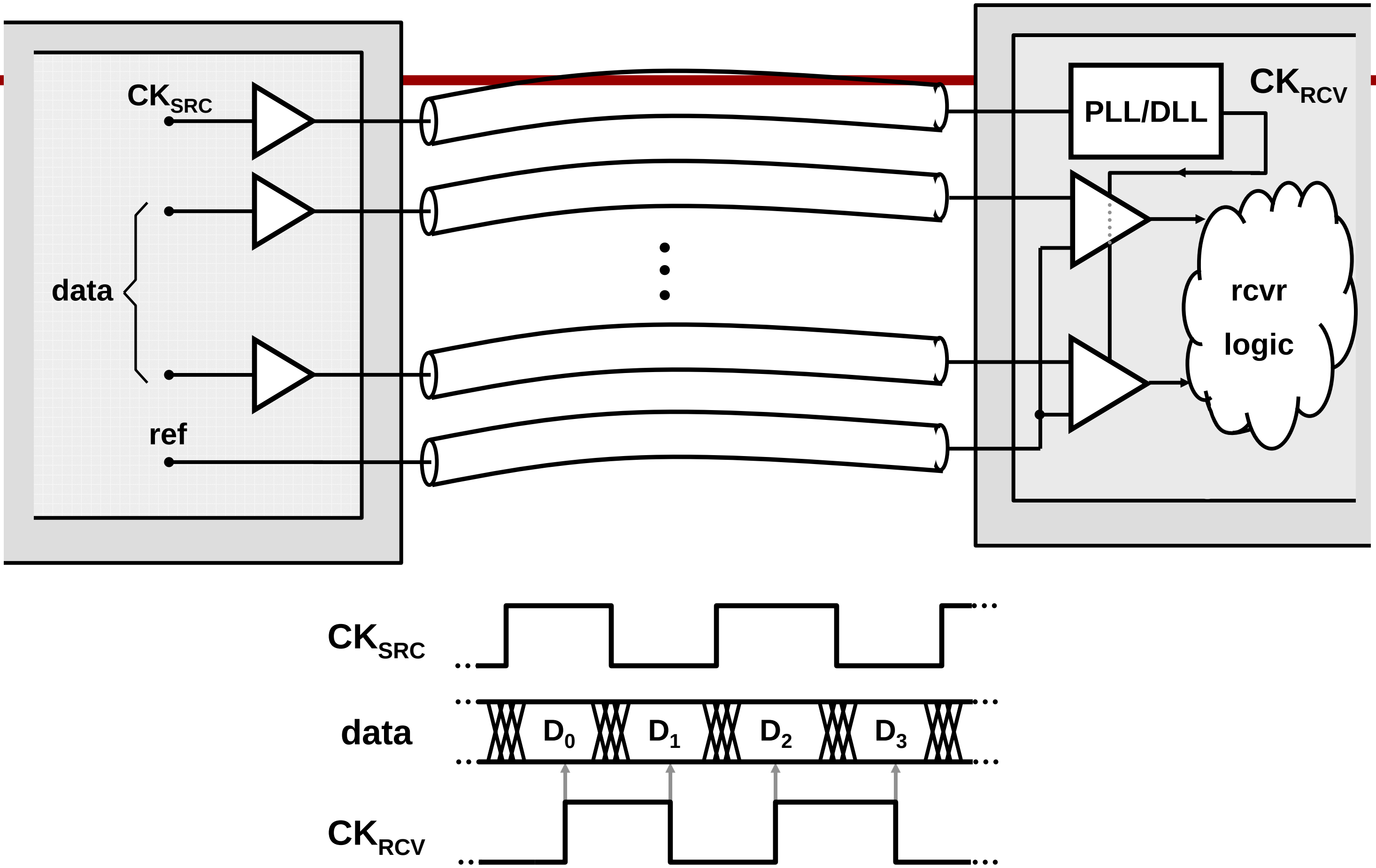
- Plesiochronous:

  Almost the same frequency

  – Mostly everything else today



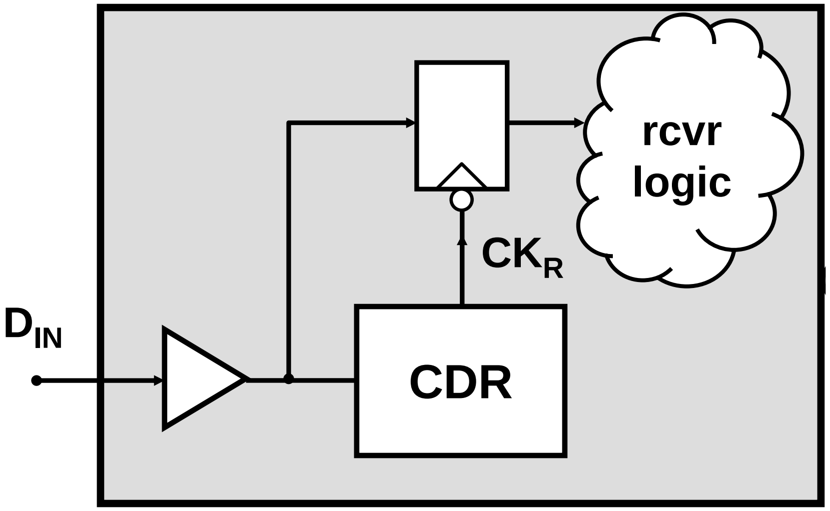
Source Synchronous Systems

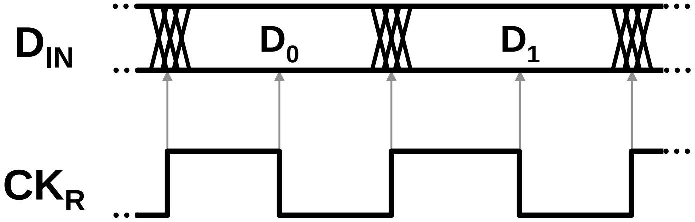


– Position on-chip sampling clock at the optimal point

i.e. maximize “timing” margin

Serial Link Circuit





- Recover incoming data fundamental frequency
- Position sampling clock at the "optimal" point

Finite Impulse Response Filters

- In DSP filters are done in the discrete time domain
  - Instead of x(t), x_n

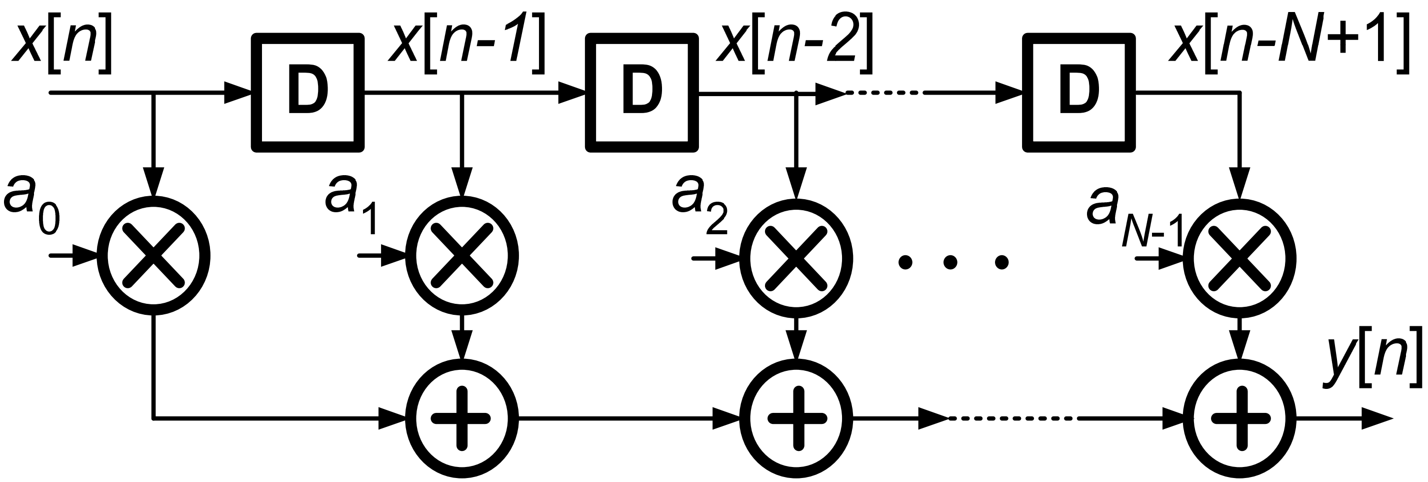
- Filter is formed by convolution of input with filter h(t)
  - Output at every point is the sum:

$$y[n] = a_0 x[n] + a_1 x[n-1] + a_2 x[n-2] + \ldots + a_N x[n-N+1]$$

- This is generally called an FIR filter
  - Finite impulse response filter (output depends only on input)
  - IIR filters have output depend on prior output
    - Infinite impulse response (like RC circuits)

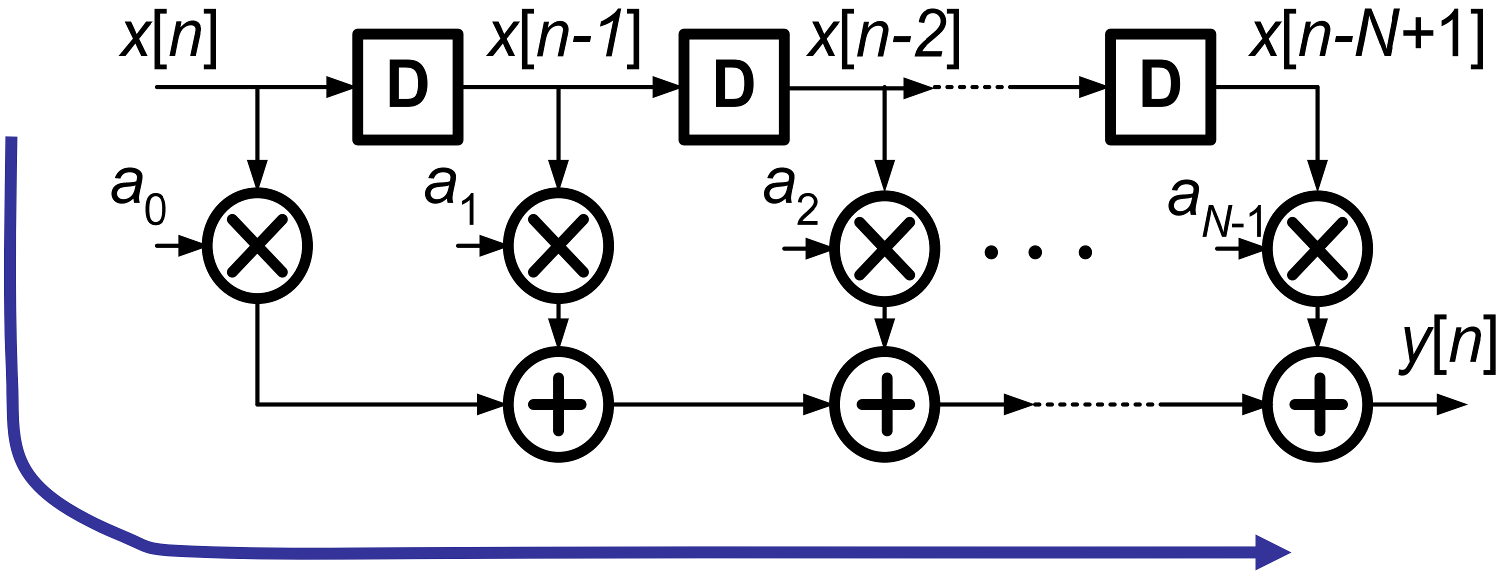
Transversal Filter

$$y[n] = a_0 x[n] + a_1 x[n-1] + a_2 x[n-2] + \ldots + a_N x[n-N+1]$$



Critical Path

- Digital FIR



$$T = T_{mult} + (N\text{-}1)T_{add}$$

One Point To Keep In Mind

- We are working with small signal values
  - For binary (2 PAM) x is in {0,1}
  - For 4PAM x is in {0,1,2,3}

- So multiplication is generally not an issue
  - For 2 PAM it is trivial
  - For 4 PAM one shift and add

- The problem is the adds
  - While x is one or two bits, the “a” are larger
    - Generally larger then input precision
    - Since you need to add many of them up and have small quantization errors.

Pipelining

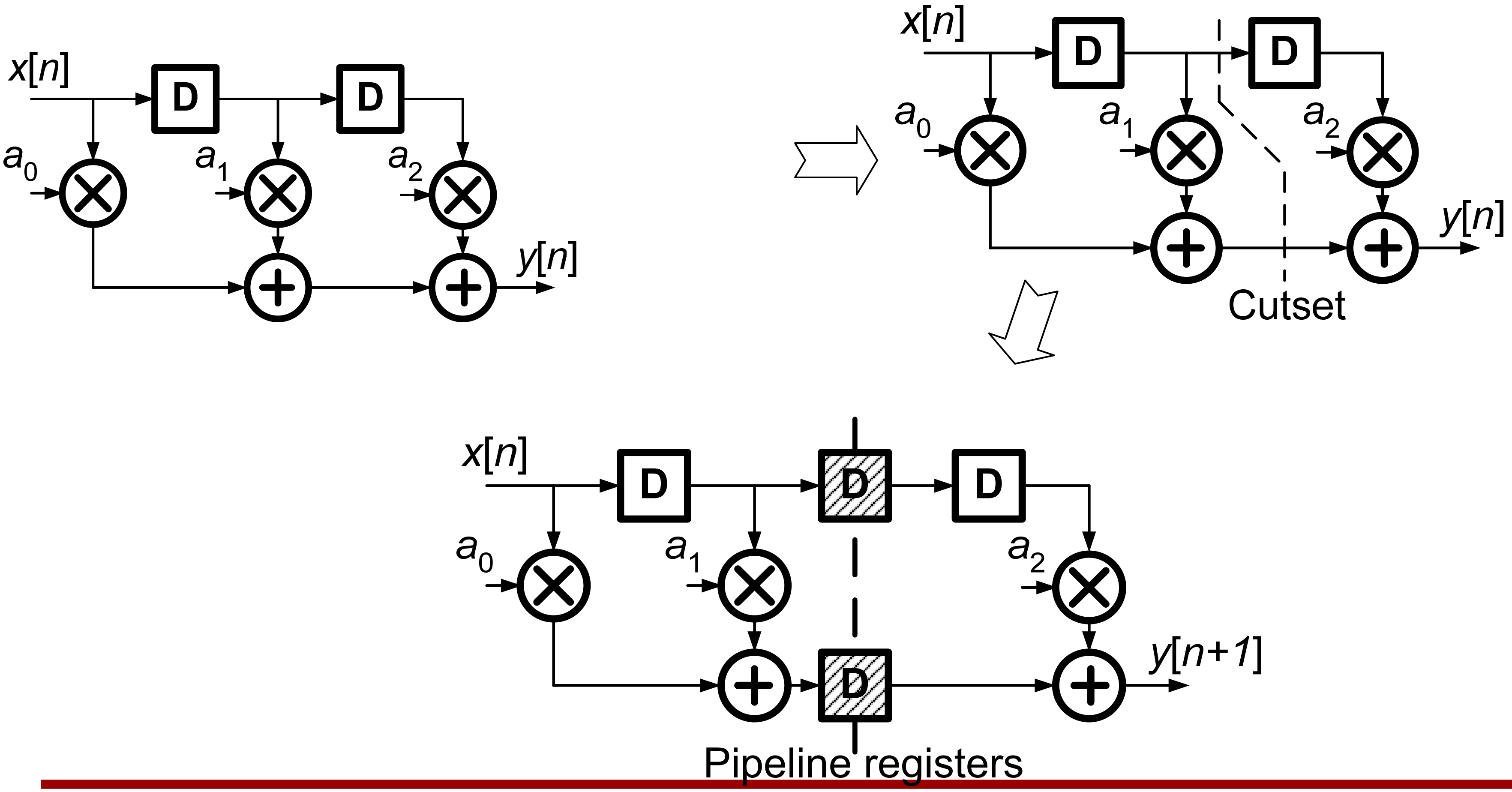
- Pipelining can be used to increase throughput
  - True for digital and mixed signal inplementations

- Pipelining: Adding same number of delay elements
  - In each forward cutset (in the data-flow graph)
    - From the input to the output
- Cutset: set of edges that if removed, graph becomes disjoint
  - Forward cutset: cutset from input to output over all edges

- Plus - Increases frequency
- Minus - Increases latency and register overhead (power, area)

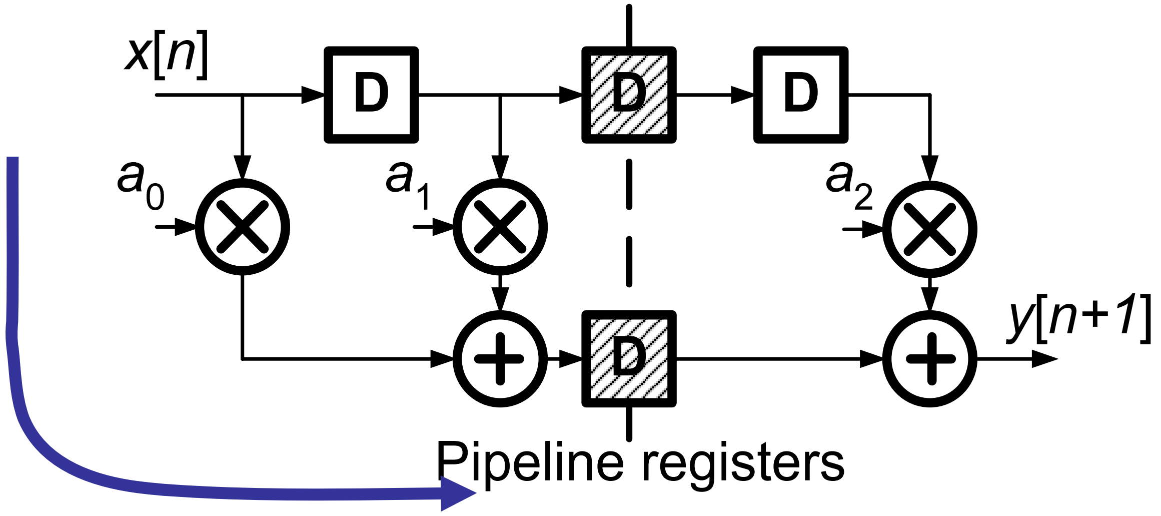
Pipelining

- 3-tap FIR



Pipelined Direct FIR

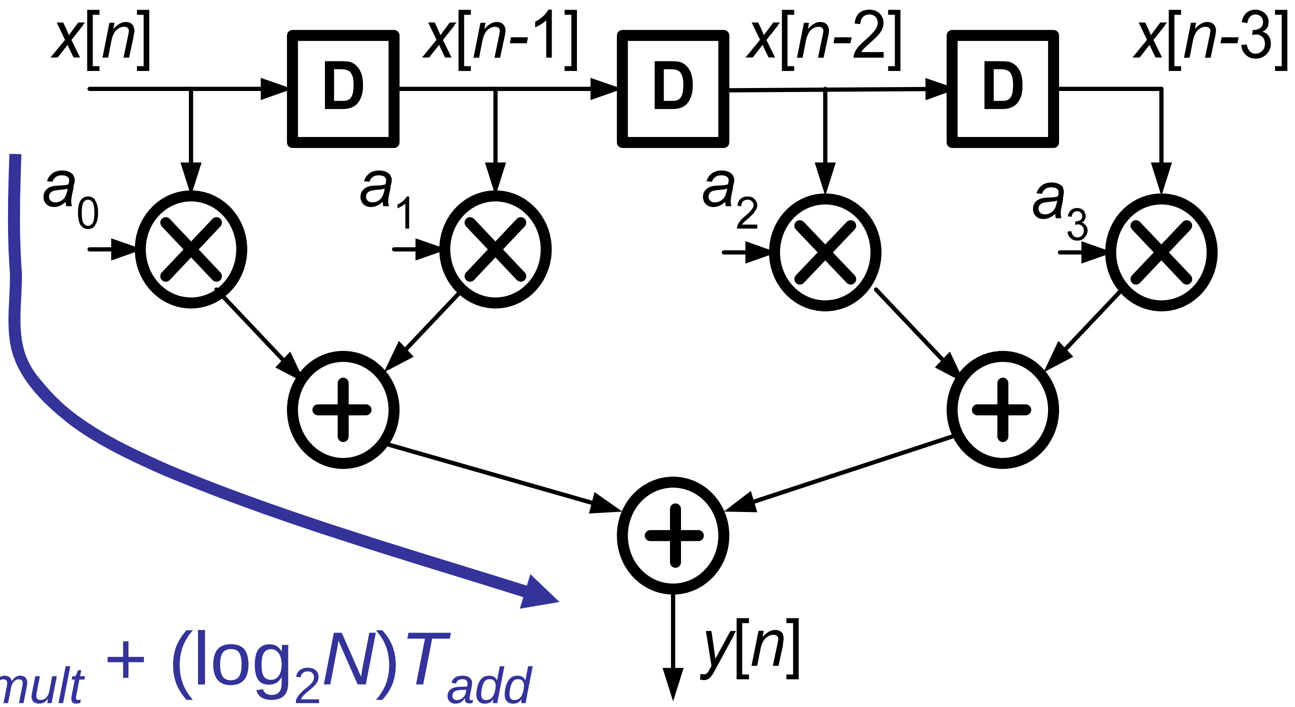
- Critical path



$$T = T_{mult} + T_{add}$$

Multi-Operand Addition

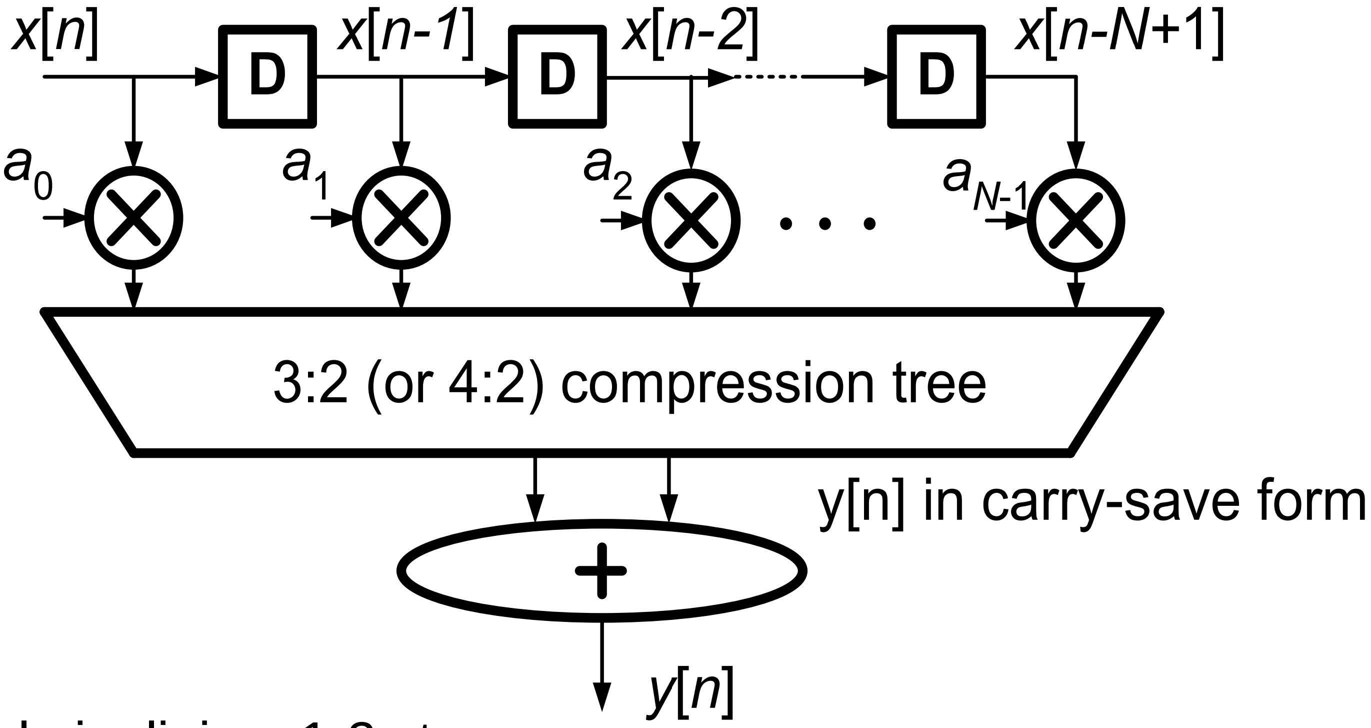
- Adders form a tree



$T = T_{mult} + (\log_2 N) T_{add}$

Multi-Operand Addition

- Using 3:2 or 4:2 compression
  - This is the same as a multiplier tree (in two lectures)



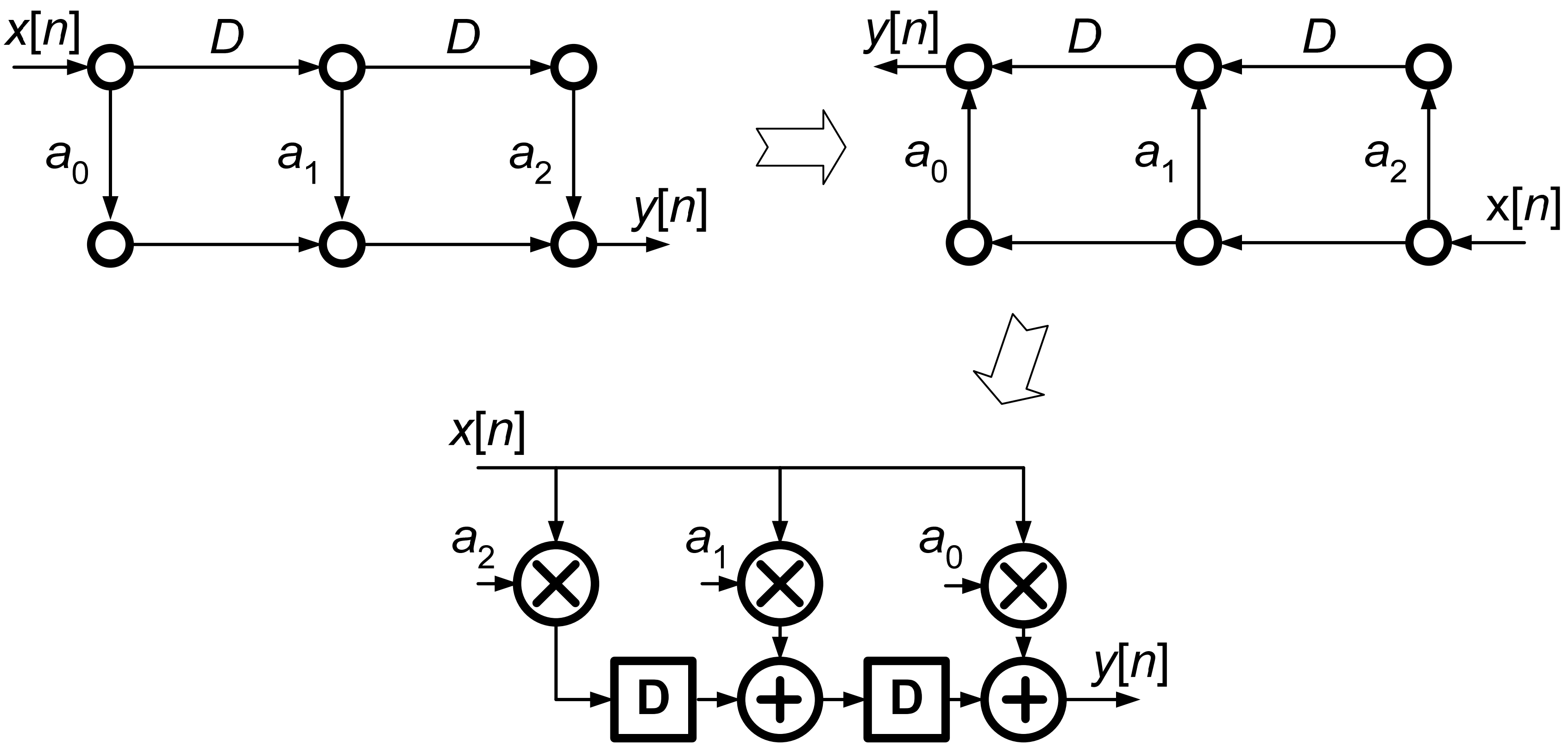
- Optional pipelining, 1-2 stages

Transposing FIR

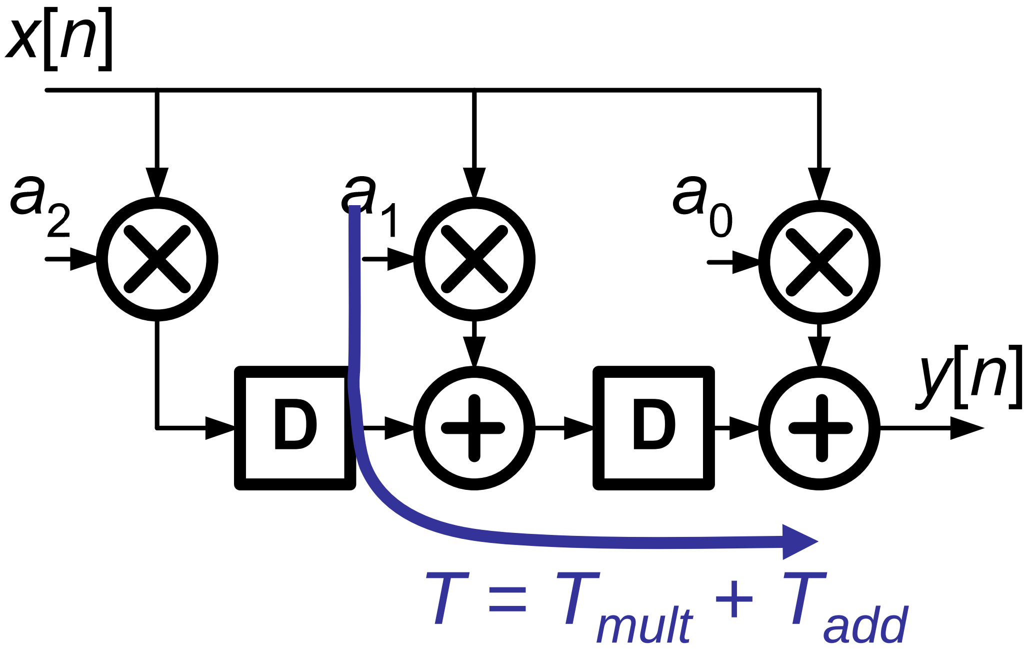
- Transposition:
  - Reversing the direction of all the edges
    - In a signal-flow graph,
  - Interchanging the input and output ports
  - Functionality unchanged

Transposed FIR

- Represent as a signal-flow graph



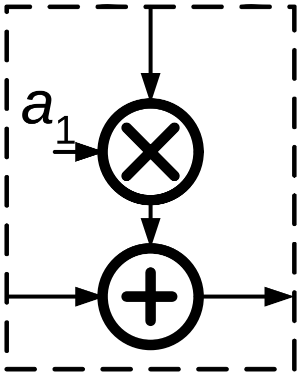
Transposed FIR



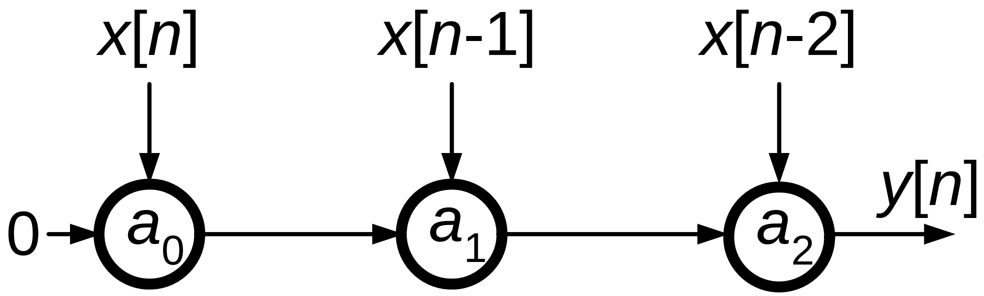
- Critical path shortened
- Input loading increased

Parallel FIR

- Feed-forward algorithms are easy to parallelize
- Processing element representation of a transversal filter



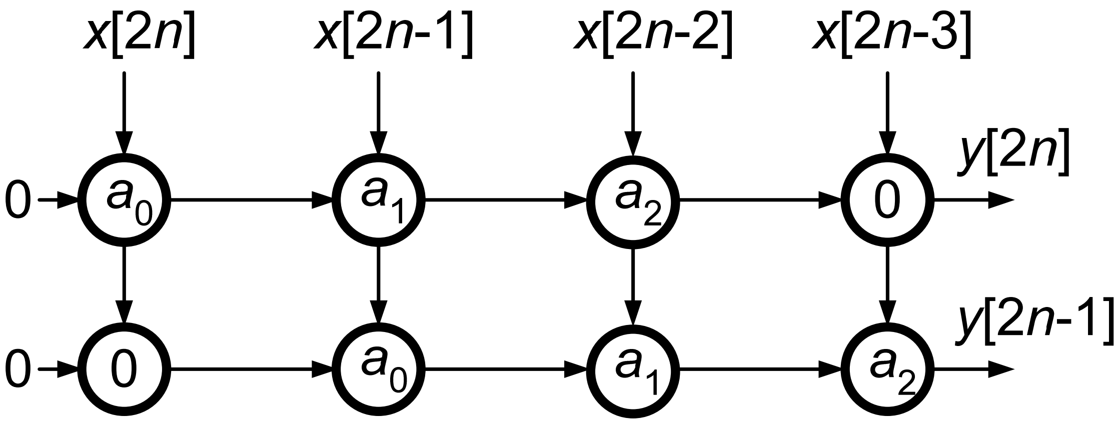
Processing element



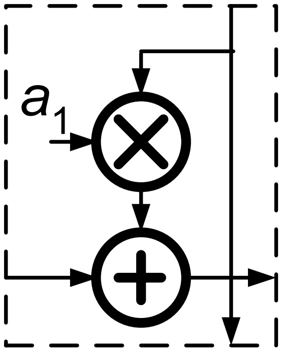
Transversal filter

Parallel FIR

- Two parallel paths
- Two cycles to complete operation
- Can be extended to more



Two parallel path FIR



Processing element

Table Lookup

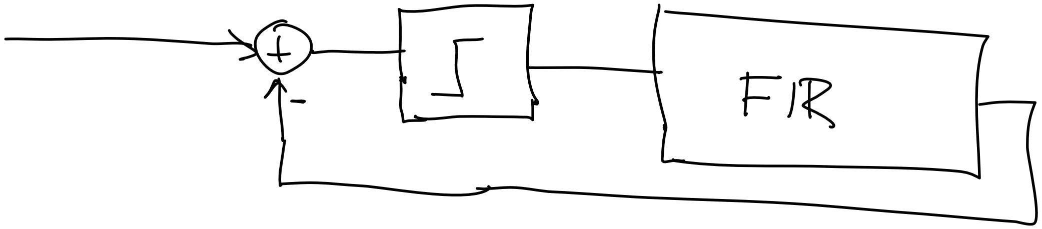
- If the input data is only one or two bits
  - There are not that many input combinations

- Rather than adding the numbers together
  - Add them before hand, and just store the results in a SRAM
  - Address of SRAM is just sequence of inputs to filter
    - x_n x_{n-1} x_{n-2} x_{n-3} x_{n-4}
  - Values in memory

00000	0	00001	a4
00010	a3	00011	a3+a4

- Replaces adds and multipliers by memory
  - But it grows exponentially with number of bits needed

Decision Feedback Equalization

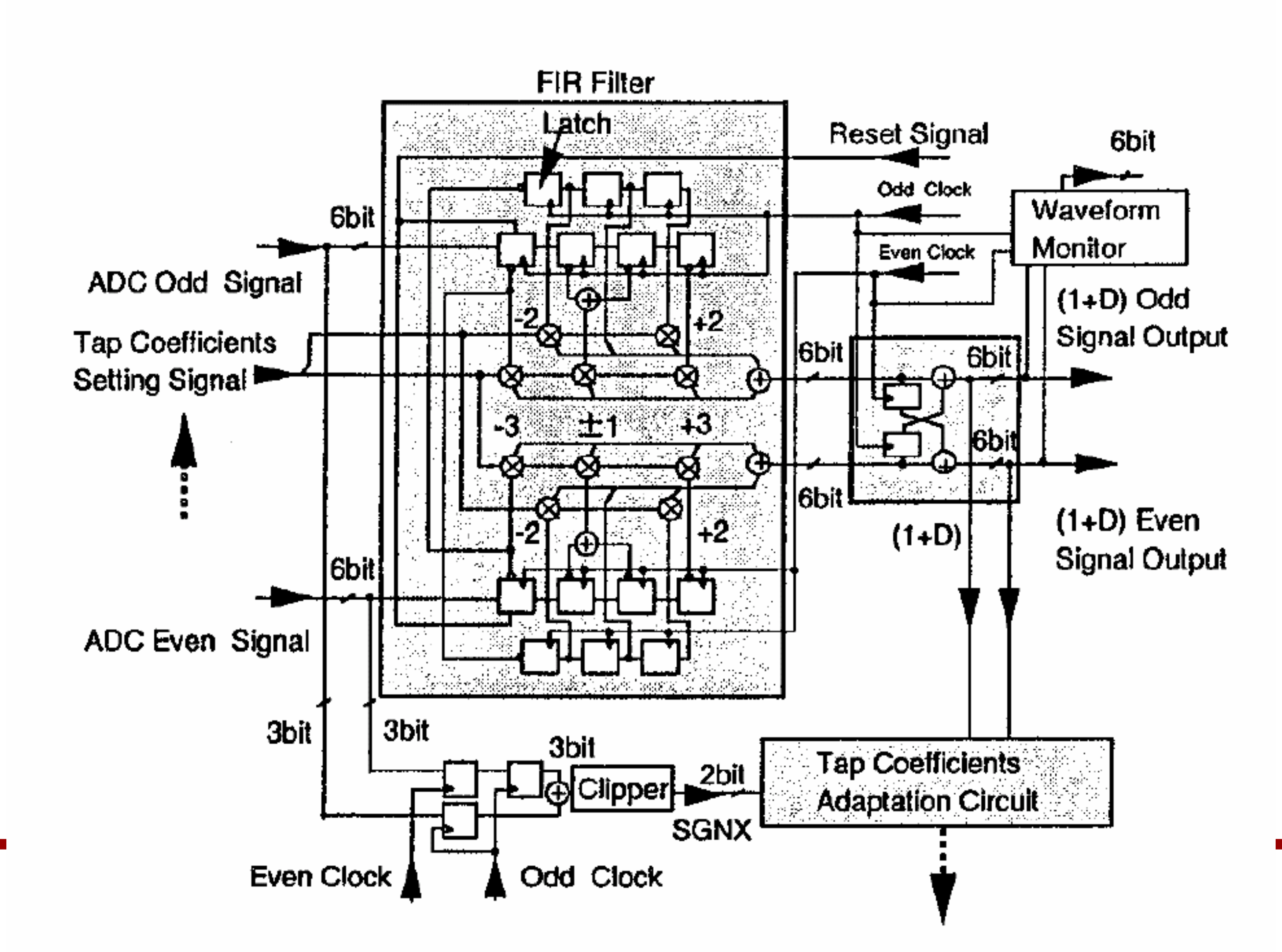


- The main problem with DFE
  - You need the output of the FIR filter NOW
    - Need it to generate the next bit

- Latency in the FIR filter is a problem

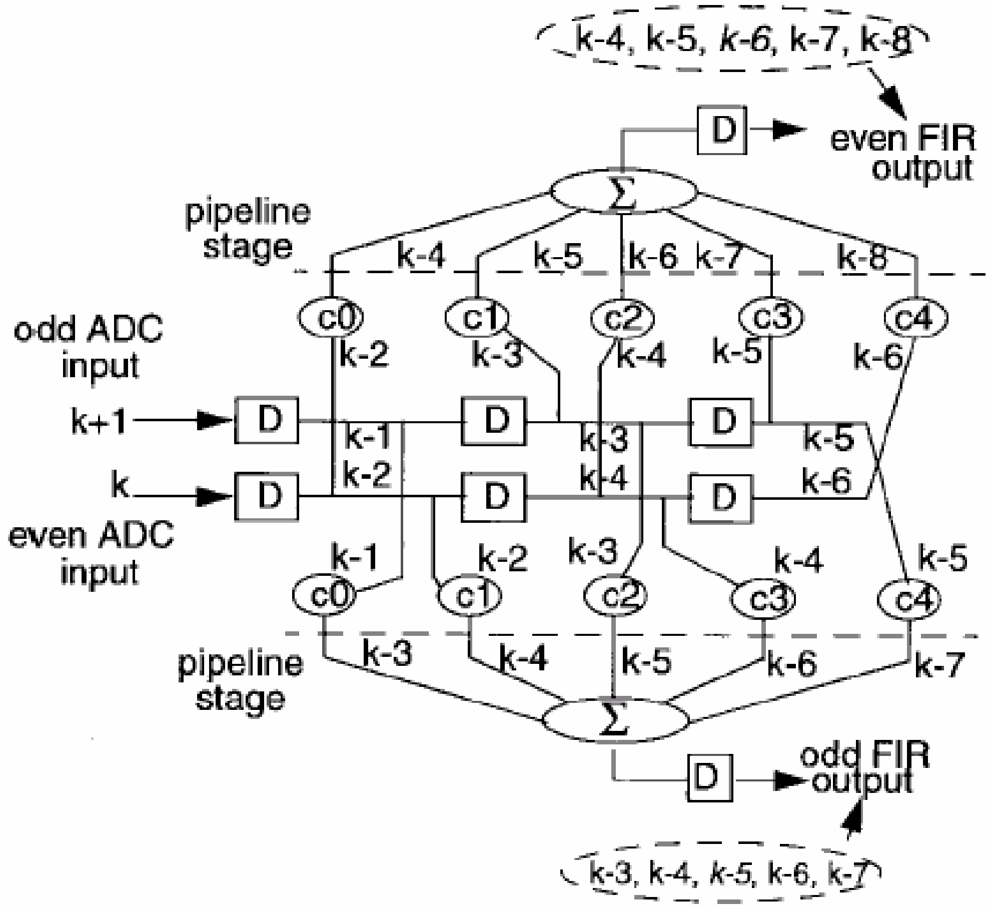
Practical Digital Equalizers

- Mita, ISSCC’96, two parallel paths
- 150Mb/s 0.7µm BiCMOS



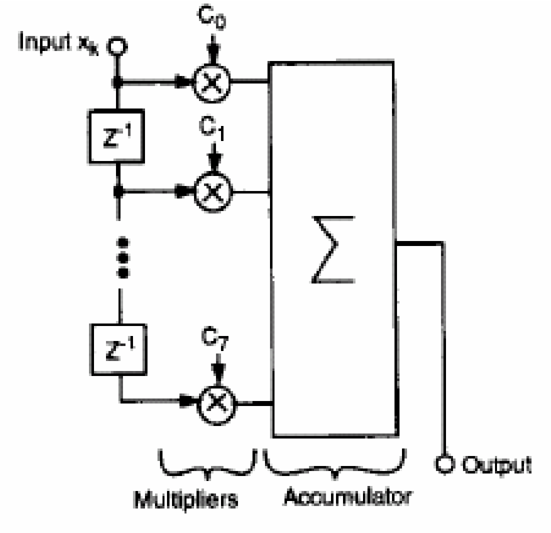
Practical Digital Equalizers

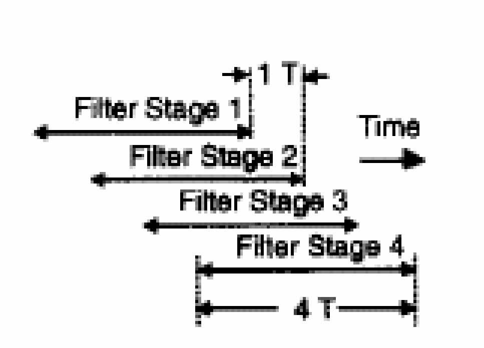
- Moloney, JSSC 7/98, 2 parallel paths, 3:2 Wallace
- 150Mb/s 0.7µm BiCMOS

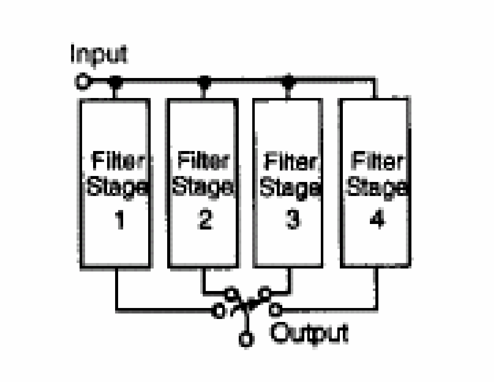


Practical Digital Equalizers

- Wong, Rudell, Uehara, Gray JSSC 3/95,
  4 parallel paths
- 50Mb/s, 1.2µm CMOS

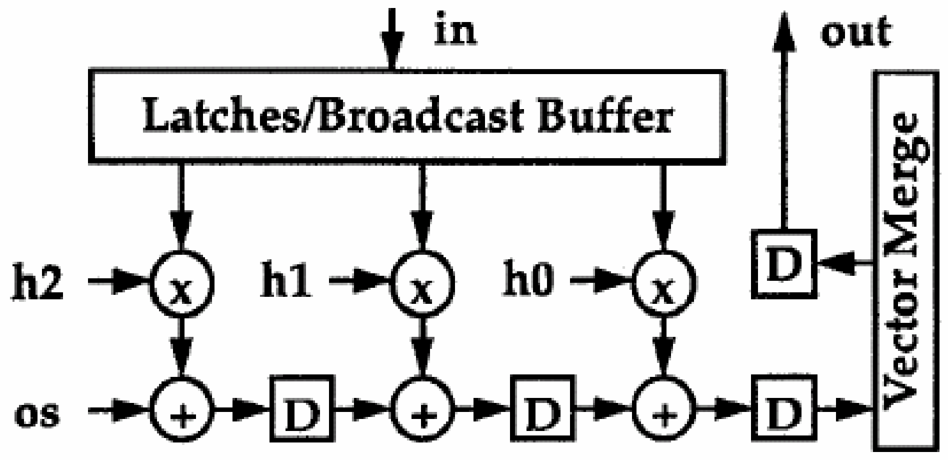






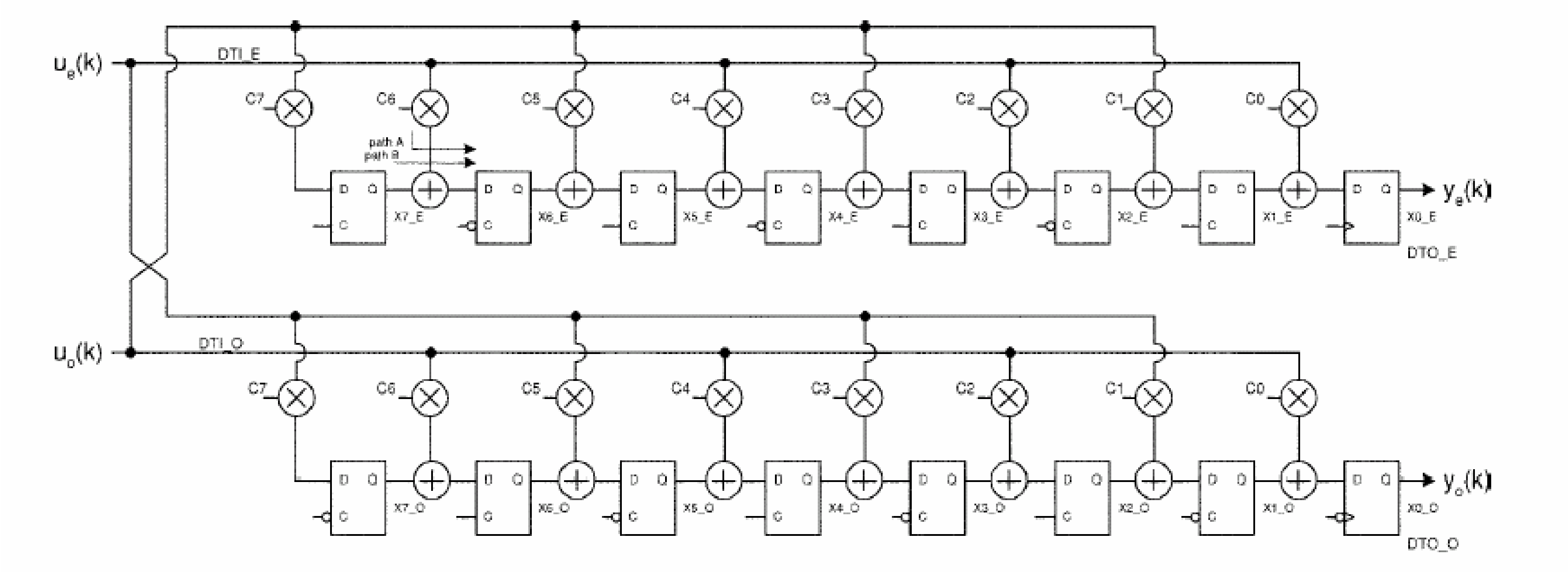
Practical Digital Equalizers

- Thon, ISSCC’95
- Transposed filter, 240Mb/s 0.8µm 3.7V CMOS, 150mW
- Semi-static coefficients, Booth-encoded



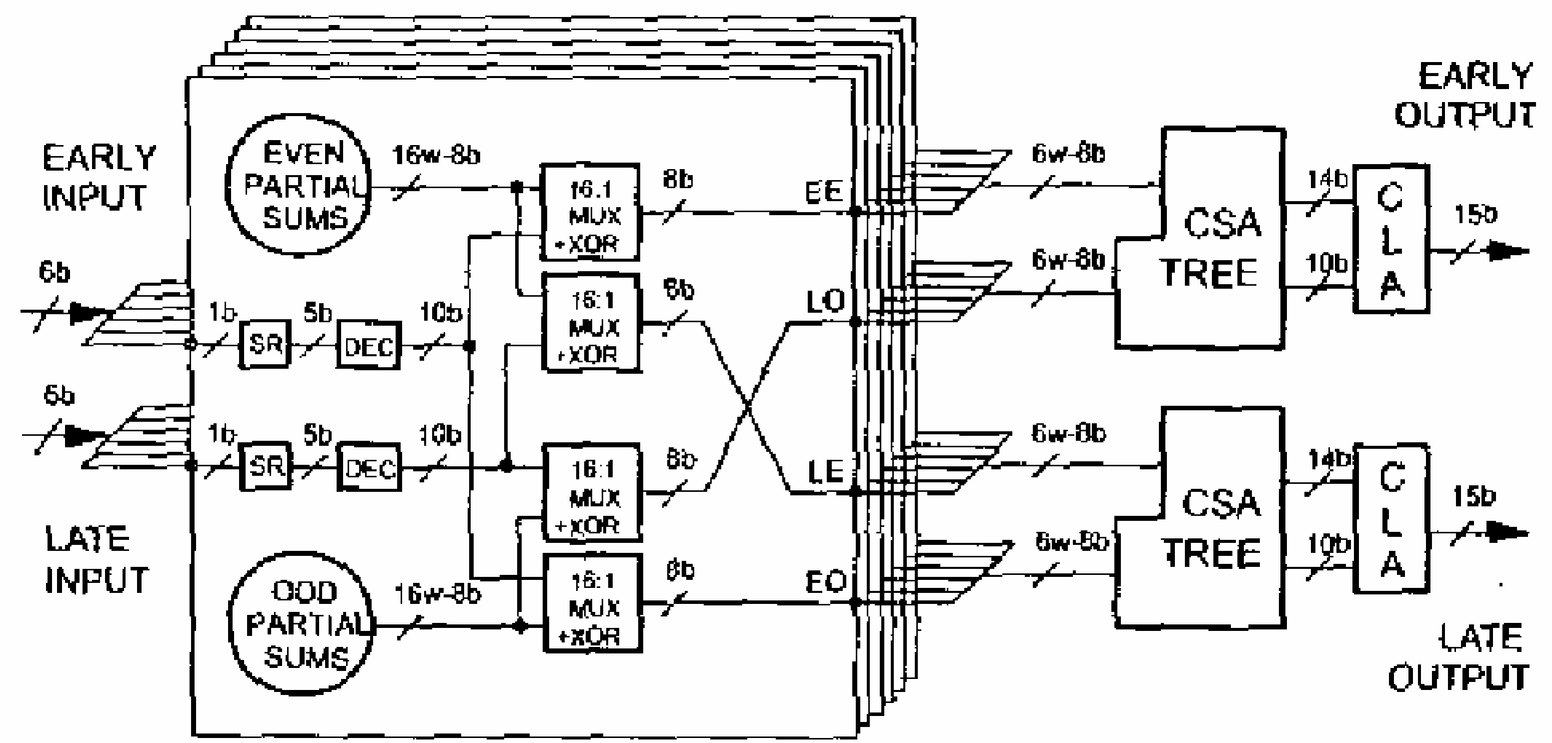
Practical Digital Equalizers

- Staszewski, JSSC 8/00
- 2 parallel transposed paths, Booth encoded data
- 550Mb/s 0.21µm CMOS, 36mW



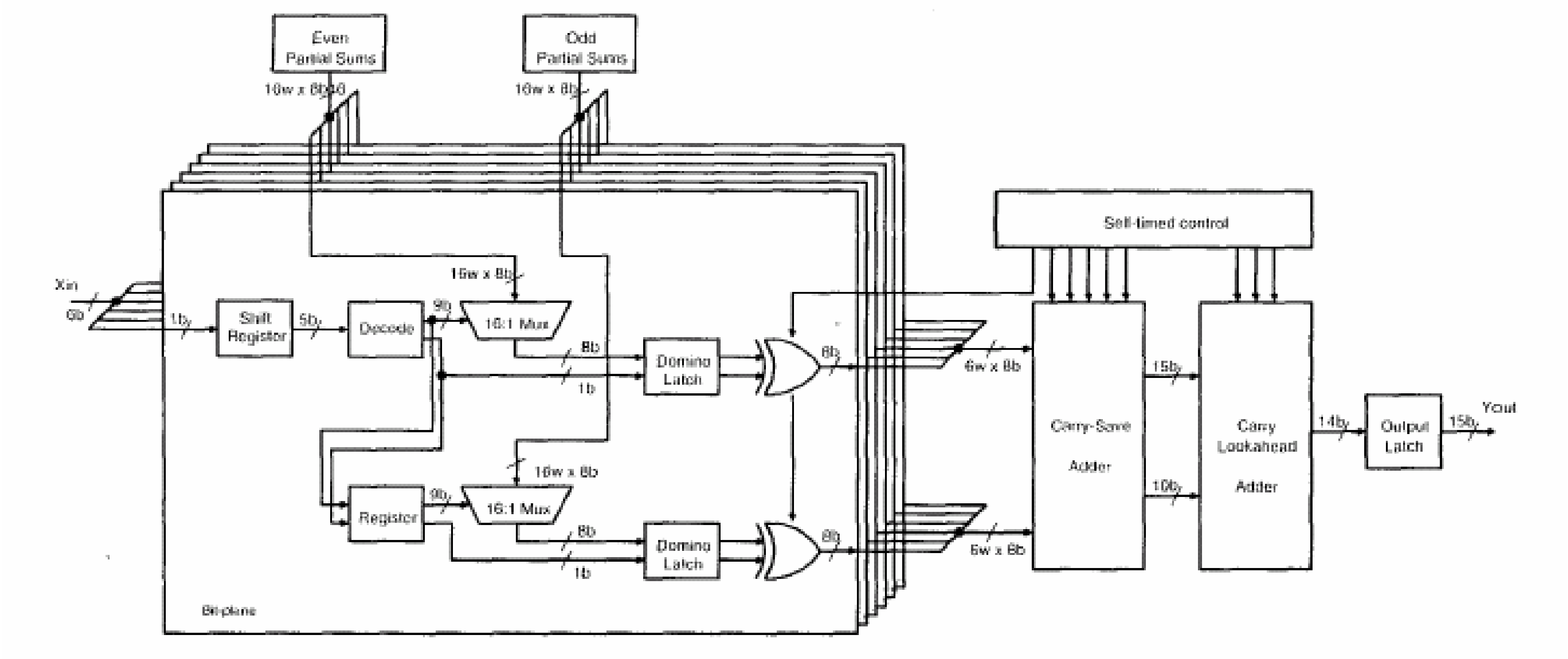
Practical Digital Equalizers

- Rylov, ISSCC'01
- 2.3Gb/s, 1.2W, 0.18µm domino CMOS
- Distributed arithmetic



Practical Digital Equalizers

- Tierno, ISSCC'02
- 1.3Gb/s, 450mW, 0.18µm 2.1V domino CMOS



TI DFE Design ISSCC 07

- Uses Memory lookup
  - Runs at 12Gs/s
  - Binary

- Check it out …

References from Bora Nikolic

- R. Jain, P.T. Yang, T. Yoshino, "FIRGEN: a computer-aided design system for high performance FIR filter integrated circuits," IEEE Transactions on Signal Processing, vol.39, no.7, pp.1655-1668, July 1991.
- R.A. Hawley, B.C. Wong, T.-J. Lin, J. Laskowski, H. Samueli, "Design techniques for silicon compiler implementations of high-speed FIR digital filters," IEEE Journal of Solid-State Circuits, vol.31, no.5, pp.656-667, May 1996.
- W.L. Abbott, et al, "A digital chip with adaptive equalizer for PRML detection in hard-disk drives" IEEE International Solid-State Circuits Conference, Digest of Technical Papers, ISSCC'94, San Francisco, CA, Feb. 16-18, 1994, pp. 284 -285.
- D.J. Pearson, et al, "Digital FIR filters for high speed PRML disk read channels," IEEE Journal of Solid-State Circuits, vol.30, no.12, pp.1517-1523, May 1995.
- S. Mita, et al, "A 150 Mb/s PRML chip for magnetic disk drives," IEEE International Solid-State Circuits Conference, Digest of Technical Papers, ISSCC'96, San Francisco, CA, Feb. 8-10, 1996, pp. 62 -63, 418.
- D. Moloney, J. O'Brien, E. O'Rourke, F. Brianti, "Low-power 200-Msps, area-efficient, five-tap programmable FIR filter," IEEE Journal of Solid-State Circuits, vol.33, no.7, pp.1134-1138, July 1998.
- C.S.H. Wong, J.C. Rudell, G.T. Uehara, P.R. Gray, "A 50 MHz eight-tap adaptive equalizer for partial-response channels," IEEE Journal of Solid-State Circuits, vol.30, no.3, pp.228-234, March 1995.
- L.E. Thon, P. Sutardja, F.-S. Lai, G. Coleman, "A 240 MHz 8-tap programmable FIR filter for disk-drive read channes," 1995 IEEE International Solid-State Circuits Conference, Digest of Technical Papers, ISSCC '95, pp.82-3, 343, San Francisco, CA, 15-17 Feb. 1995.
- R. B. Staszewski, K. Muhammad, P. Balsara, "A 550-MSample/s 8-Tap FIR Digital Filter for Magnetic Recording Read Channels," IEEE Journal of Solid-State Circuits, vol. 35, no. 8, pp. 1205-1210, August 2000.
- S. Rylov, et al, "A 2.3 GSample/s 10-tap digital FIR filter for magnetic recording read channels," IEEE International Solid-State Circuits Conference, Digest of Technical Papers, ISSCC'01, San Francisco, CA, Feb. 5-7, 2001, pp. 190-191.
- J. Tierno, et at, "A 1.3 GSample/s 10-tap full-rate variable-latency self-timed FIR filter with clocked interfaces," IEEE International Solid-State Circuits Conference, Digest of Technical Papers, ISSCC'02, San Francisco, CA, Feb. 3-7, 2002, pp. 60 -61, 444.