Lecture 2

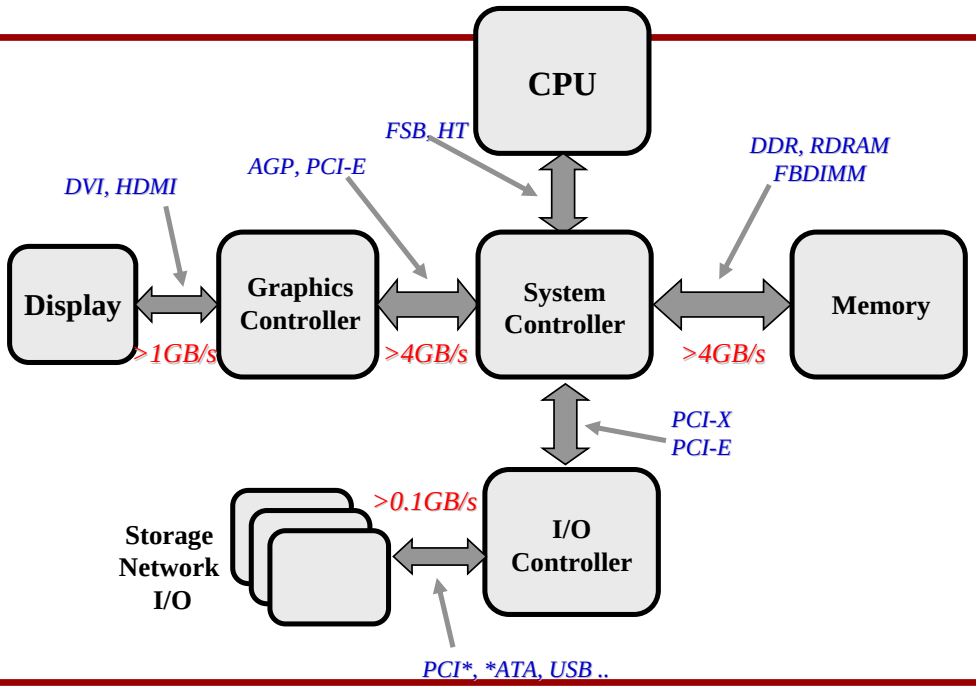
High-Speed I/O

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

Readings

- Readings
  - Techniques for High-speed Implementation of Nonlinear Cancellation, Sanjay Kasturia and Jack H. Winters
- Overview:
  - Your project will be the design of a circuit that processes the input data from a high-speed I/O. This processing is generally done in a mixed signal manner today, but your job will be to build a digitial implementation of the algorithm. This lecture will try to give you some background about why I/O rates are important, and what issues need to be resolved to achieve high performance. The next lecture will discuss the operation of the circuit you need to build.

Computers Today



Speed of Light: The Difference Between I/O and On-Chip Wires

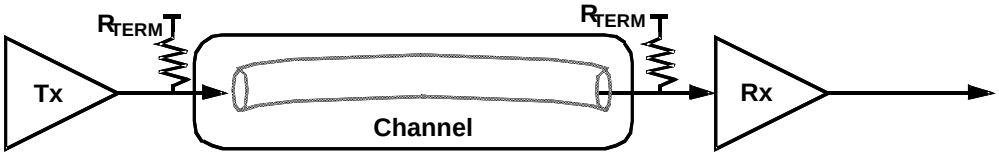
- First question:
  - Why is I/O different from on-chip wires?
    - Both send signals to each other
- Gates send data to each other all the time
  - Don't generally worry about signals, or delay
  - Model the connection between gates as a capacitor
    - Sometimes a capacitor/resistor network
- Answer:
  - On-chip, ignore the speed of light, assume "c" infinite
    - For external wires can't make that assumption
      - Wire connecting the pins is not an equipotential
  - References are different

Finite Speed of Light Ramifications

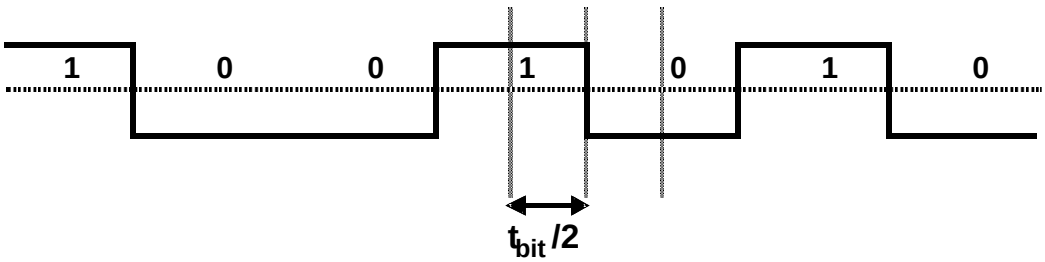
- Signals must have delay in reaching destination
  - Td = L/ν, bits arrive at a different time than when sent
  - Thus must determine 'right' time to sample them
- Wires store energy
  - Current is set by the geometry of wire (what else?)
    - Signal can't see termination resistor (causality)
    - V/I for the line is called the impedance, Z < 300 Ω
  - When signal is traveling on the wire
    - Power goes into the wire before it hits load
    - Since energy is conserved, wire must be storing energy
- Signal is ALWAYS a pair of currents

Link Issues

- Signaling: getting the bit to the receiver



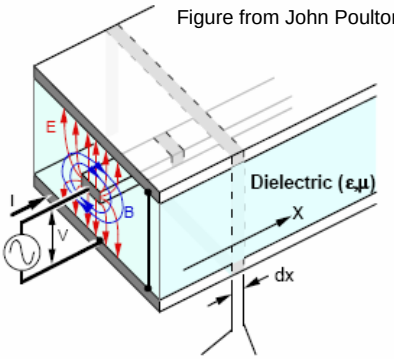
- Timing: Determining which bit is which

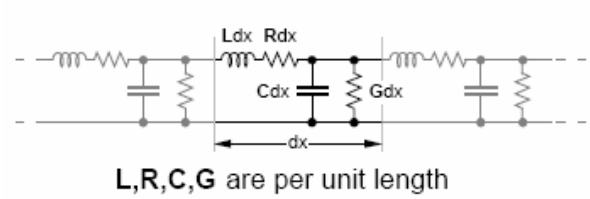


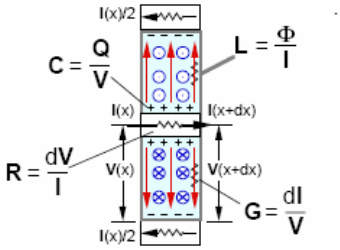
Transmission Lines

- Wire where you notice 'c' is finite
  - Current flows in one terminal
  - And flows out the other
- Energy is stored in E and B fields
  - But can model with L, C

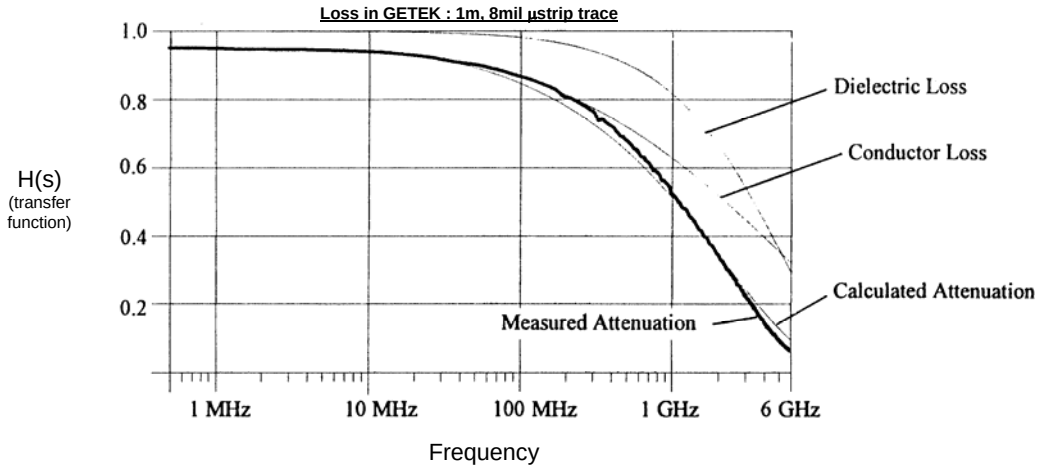
Figure from John Poulton







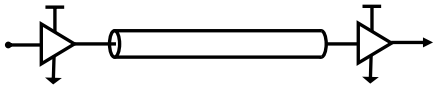
Problems : Material Loss



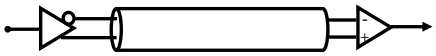
- PCB Loss : skin & dielectric loss
  - Skin Loss $\propto \sqrt{f}$
  - Dielectric loss $\propto f$: a bigger issue at higher f

Dealing With Current Return/References

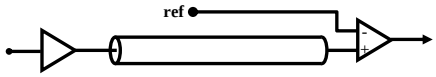
- Wire Utilization:
  - Single Ended
    shared signal return path



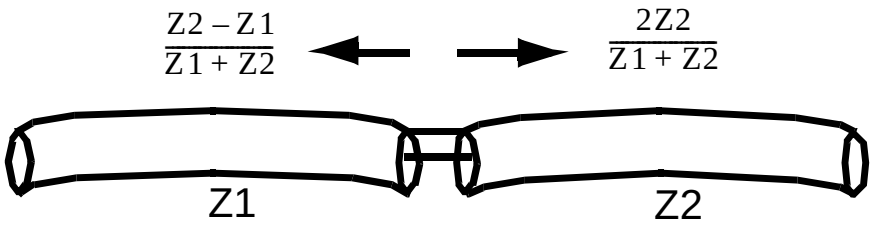
  - Differential
    explicit signal return path



  - "Pseudo" Differential



Transmission Lines

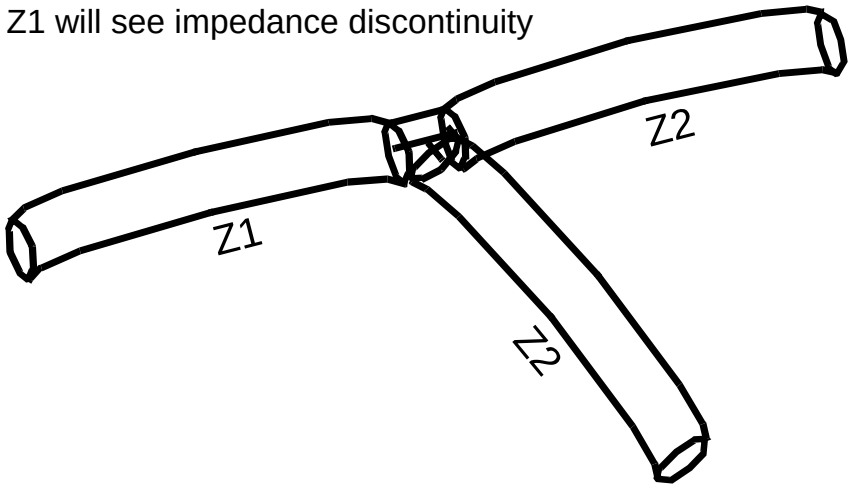


Two constraints govern behavior at any junction:

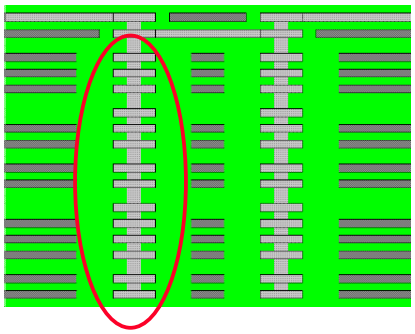
- Voltage are equal
  - They are electrically connected
- Power is conserved
  - Energy flow into junction is equal to transmitted and reflected

High-Speed Wires Are Point to Point

- Can't split a wire to go to two location
  - You will get a reflection from the junction
  - Z1 will see impedance discontinuity



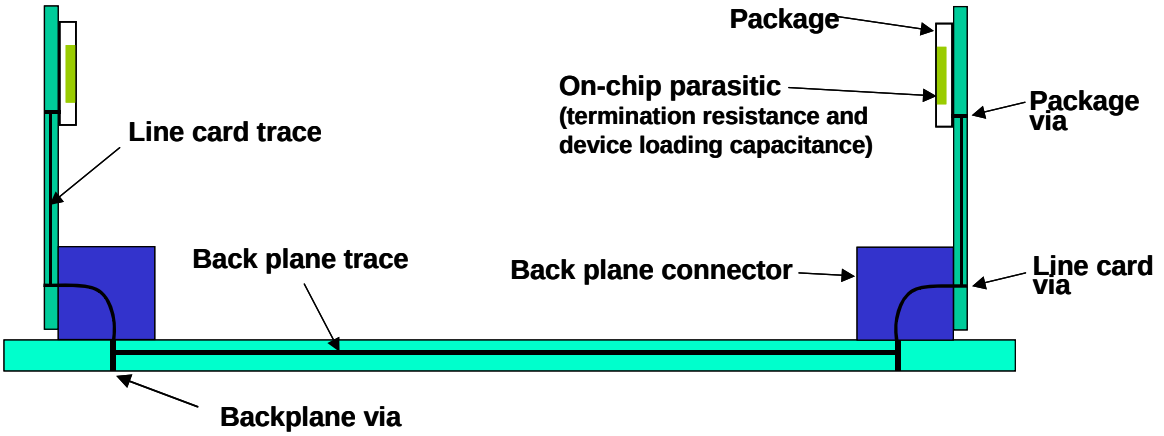
At High Speeds, Vias are Stubs



Top layer signaling
results in large via stub

- Signal energy splits at via
  - If via is short can be modeled as a cap load
  - Causes a reflection in signal
- Higher the frequency, the more sensitive you are to stubs

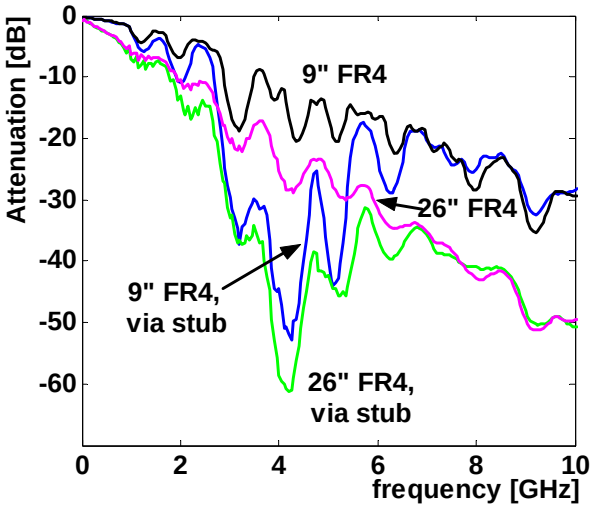
Backplane Environment



- Line attenuation
- Reflections from stubs (vias)

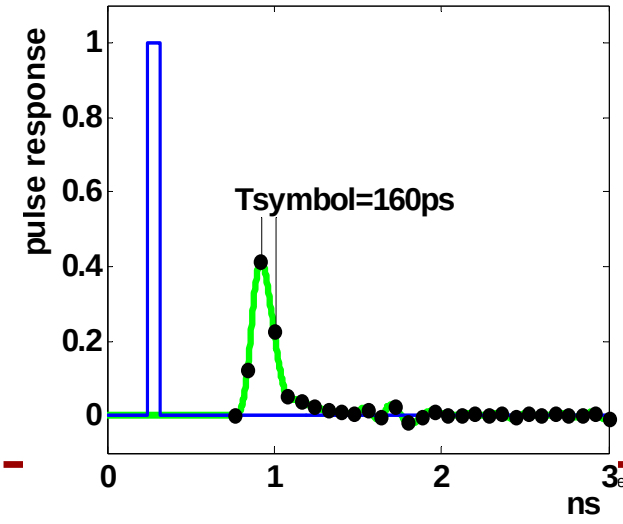
Backplane Channel

- Loss is variable
  - Same backplane
  - Different lengths
  - Different stubs
    - Top vs. Bot
- Attenuation is large
  - >30dB @ 3GHz
  - But is that bad?



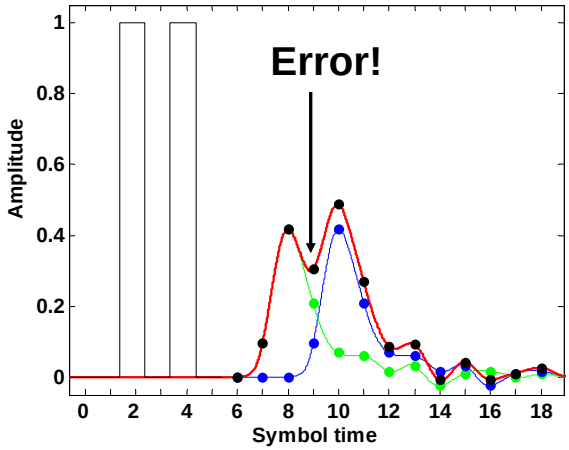
Inter-Symbol Interference (ISI)

- Channel is low pass
  - Our nice short pulse gets spread out



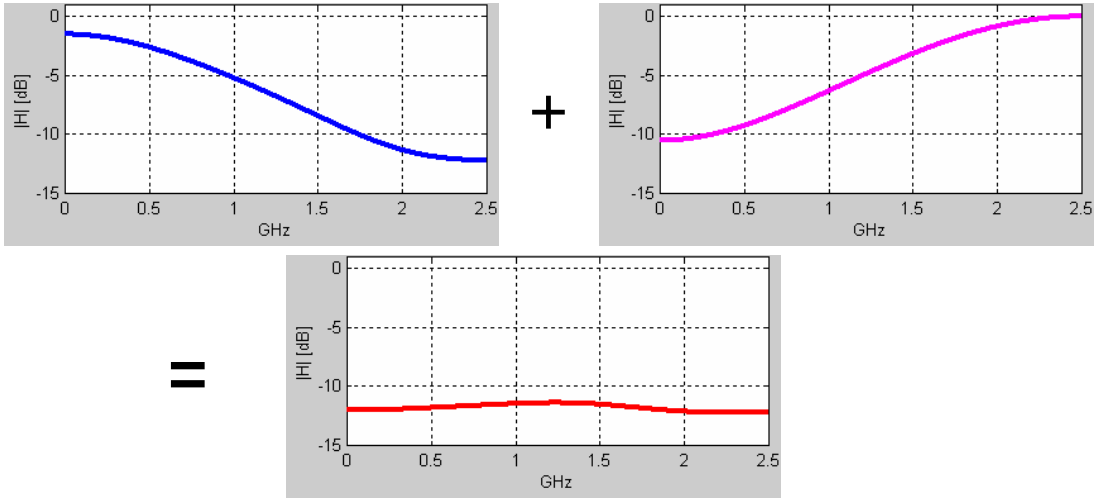
- Dispersion – short latency (skin-effect, dielectric loss)
- Reflections – long latency (impedance mismatches – connectors, via stubs, device parasitics, package)

ISI



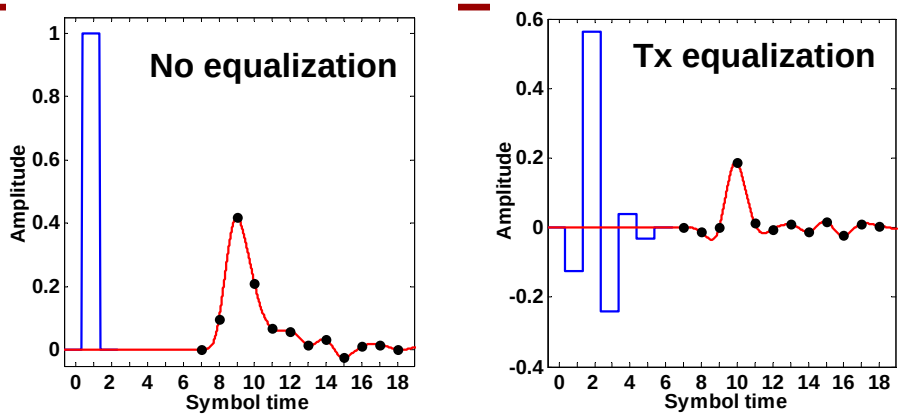
- Middle sample is corrupted by
  - 0.2 trailing ISI (from the previous symbol),
  - 0.1 leading ISI (from the next symbol) resulting in 0.3 total ISI
- As a result middle symbol is detected in error

Equalization For Loss : Goal is to Flatten Response



- Channel is band-limited
- Equalization : boost high-frequencies; or attenuate low freq

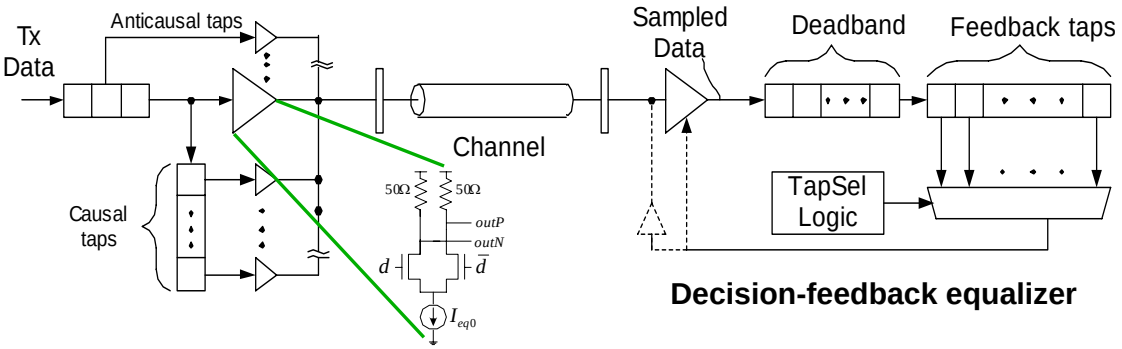
Equalization Mechanisms



- Tx equalization
  - Pre-filter the pulse with the inverse of the channel
  - Filters the low freq. to match attenuation of high freq.
- Rx feedback equalization
  - Subtract the error from the signal

Removing ISI

Linear transmit equalizer

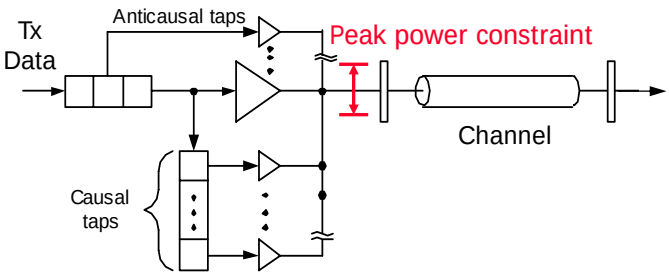


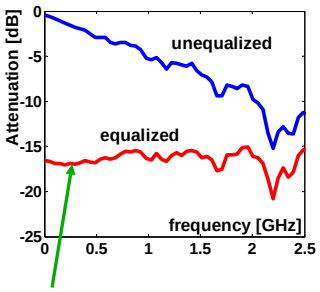
Decision-feedback equalizer

- Transmit and Receive Equalization
  - Changes signal to correct for ISI
  - Initial work was at transmitter

J. Zerbe et al, "Design, Equalization and Clock Recovery for a 2.5-10Gb/s 2-PAM/4-PAM Backplane Transceiver Cell," *IEEE Journal Solid-State Circuits*, Dec. 2003.

Transmit Equalization – Headroom Constraint



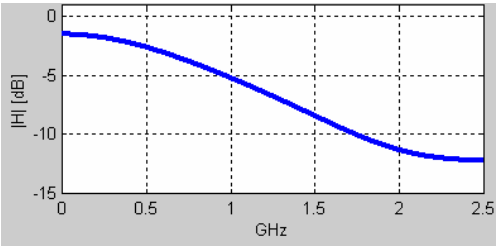


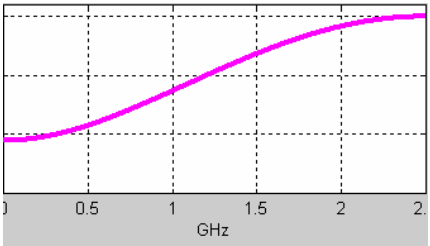
Amplitude of equalized signal depends on the channel

- Transmit DAC has limited voltage headroom
- Unknown target signal levels
  - Harder to make adaptive equalization work
- Need to tune the equalizer and receive comparator levels
  - If you have multi-level signals

Removing Interference at Receiver

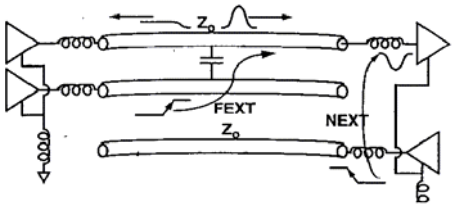
- Could also build a linear filter
  - Could have gain in the filter
  - But either it would need to be analog and have gain
  - Or need high-speed A/D
    - And real multiplication
    - Sum (ai*xi)
  - Increases channel noise too





High Frequency Channel Noise: Crosstalk

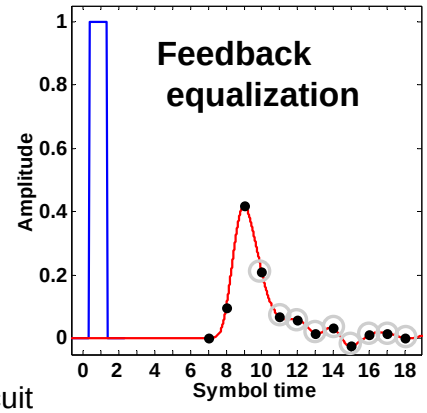
- Many sources
  - On-chip
  - Package
  - PCB traces
  - Inside connector



- Differential signaling can help
  - Minimize xtalk generation & make effects common-mode
- Both NEXT & FEXT
  - NEXT very destructive if RX and TX pairs are adjacent
    - Full swing-TX coupling into attenuated RX signal
    - Effect on SNR is multiplied by signal loss
  - Simple solution : group RX/TX pairs in connector
  - NEXT typically 3-6%, FEXT typically 1-3%

Subtract Out Residual Interference

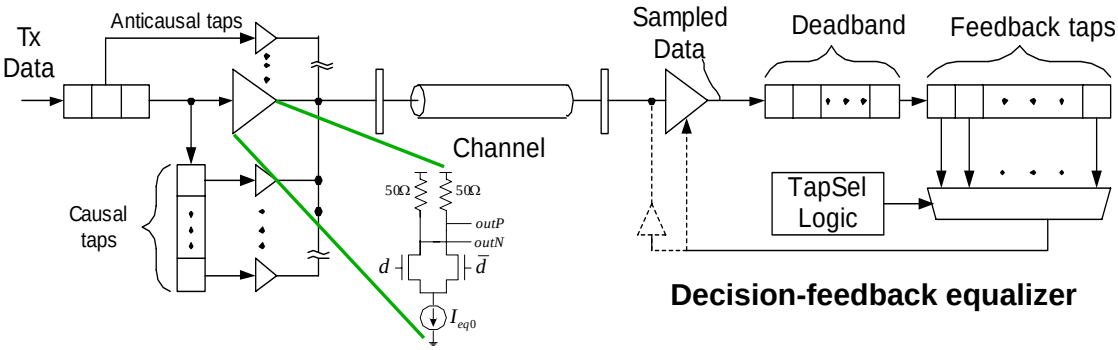
- Called Decision feedback equalization (DFE)
  - Subtracts error from input
  - No attenuation



- Problem with DFE
  - Need to know interfering bits
  - ISI must be causal
    - Problem - latency in the decision circuit
    - Receive latency + DAC settling < bit time
  - Can increase allowable time by loop unrolling
    - Receive next bit before the previous is resolved

Removing ISI

Linear transmit equalizer



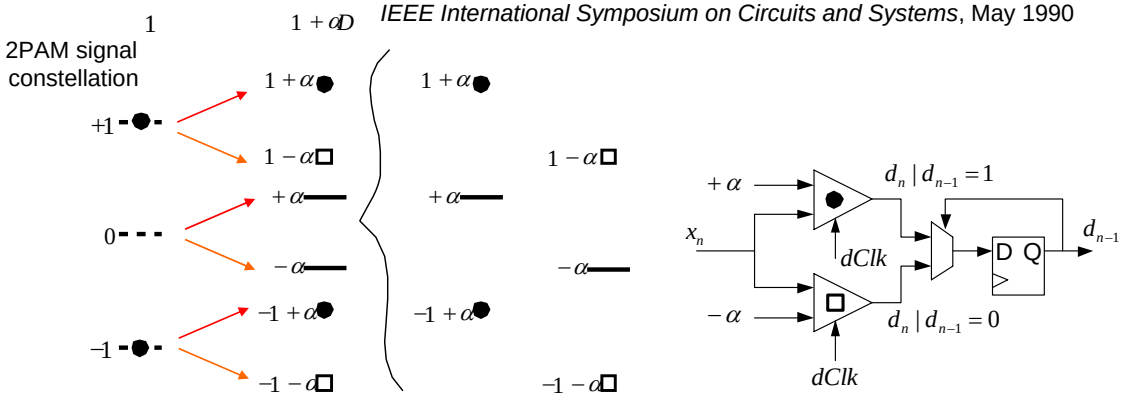
Decision-feedback equalizer

- Transmit and Receive Equalization
  - Changes signal to correct for ISI
  - Initial work was at transmitter

J. Zerbe et al, "Design, Equalization and Clock Recovery for a 2.5-10Gb/s 2-PAM/4-PAM Backplane Transceiver Cell," *IEEE Journal Solid-State Circuits*, Dec. 2003.

One Bit Loop Unrolling (for 2 level signal)

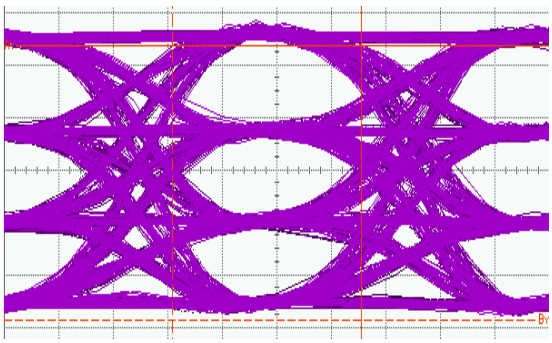
K.K. Parhi, "High-Speed architectures for algorithms with quantizer loops," *IEEE International Symposium on Circuits and Systems*, May 1990



- Instead of subtracting the error
  - Move the slicer level to include the interference
  - Slice for each possible level, since previous value unknown

More Bits/Hz

- Multi-level signaling (aka PAM)
  - Convert extra voltage margin to more bits

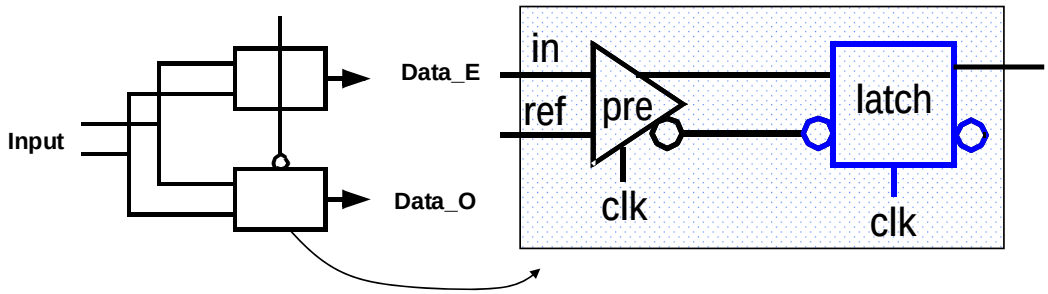


  - Works well when the noise is small
    - Need even more signal processing

Internal Speed Limitation

- Links need good quality clocks with low jitter
  - That means you want them to settle to both Vdd, and Gnd
  - If you make the clock to fast, it will not “rail”
  - And that means it will be prone to jitter
- So one limitation for links is internal clock rate
  - For power efficiency want FO on clock to be around 4
  - Need pulse width 3-4 times the slowest gate
  - Gives around 8 FO4 clock
- For higher speed bit rates
  - Need to generate multiple bits/clock
  - Use non-static CMOS clock circuits (CML & inductors)

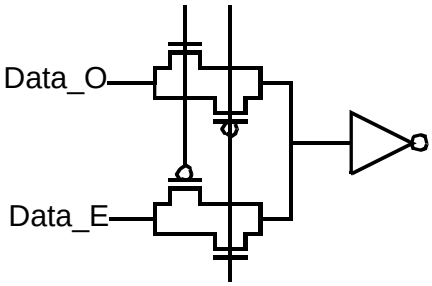
Simple Demultiplexing Receiver

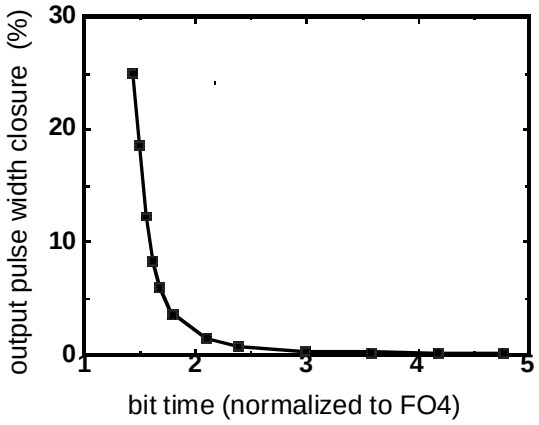


- 2-1 demux at the input
- Preconditioning stage: filter/integrate, can be clocked to avoid ISI
  - Reject CM
  - Sometimes not used
- Latch makes decision (4-FO4)

Simple Multiplexing Transmitter

- DDR: send a bit per clock edge
- Critical issues:
  - 50% duty cycle
  - Tbit > 4-FO4



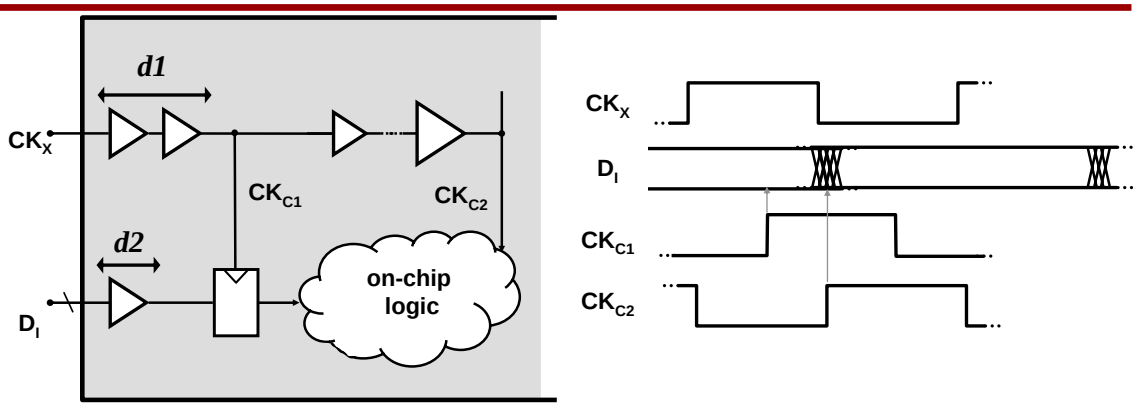


I/O Clocking Issues

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

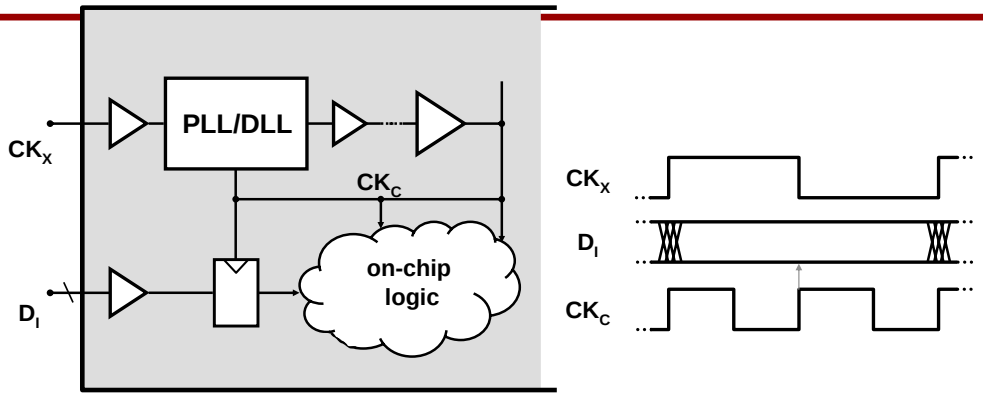
- For I/O need to worry about a number of delays
  - Clock skew between chips
  - Data delay between chips
    - Can be larger than a clock cycle (speed of light)
  - Clock skew between external clock and internal clock
    - This can be very large if not compensated
    - It is essentially the insertion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

System Clocking: Simple Synchronous Systems



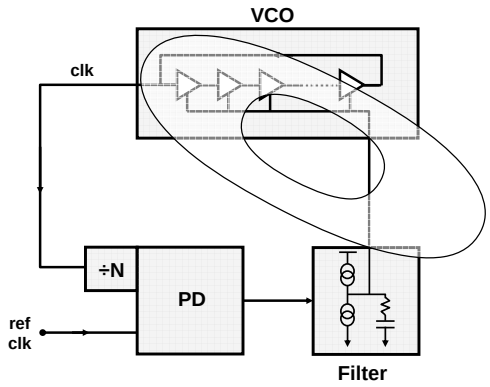
- Long bit times compared to on chip delays:
  - Rely on buffer delays to achieve adequate timing margin

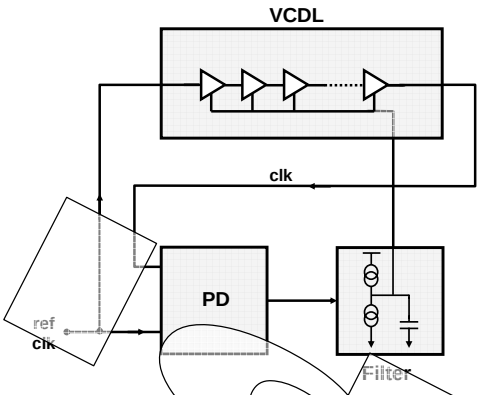
PLLs: Creating Zero Delay Buffers



- 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





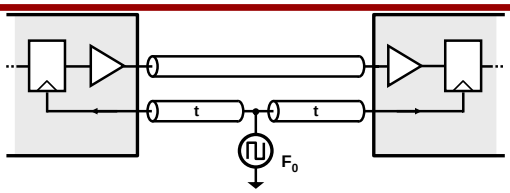
- Second/third order loop:
  - Stability is an issue
  - Frequency synthesis easy
  - Ref. Clk jitter gets filtered
  - Phase error accumulates
- First order loop:
  - Stability guaranteed
  - Frequency synthesis problematic
  - Ref. Clk jitter propagates
  - Phase error does not accumulate

After Many Years of Research

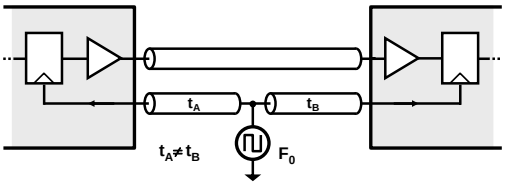
- And many papers and products
- One can mess up either a DLL or PLL
  - Each has it own strengths and weaknesses
- If designed correctly, either will work well
  - Jitter will be dominated by other sources
- Many good designs have been published
  - It is now a building block that is often reused
 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

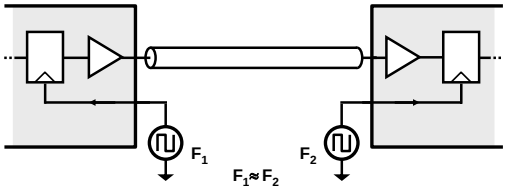
- Synchronous:
  Same frequency and phase
  - Conventional buses



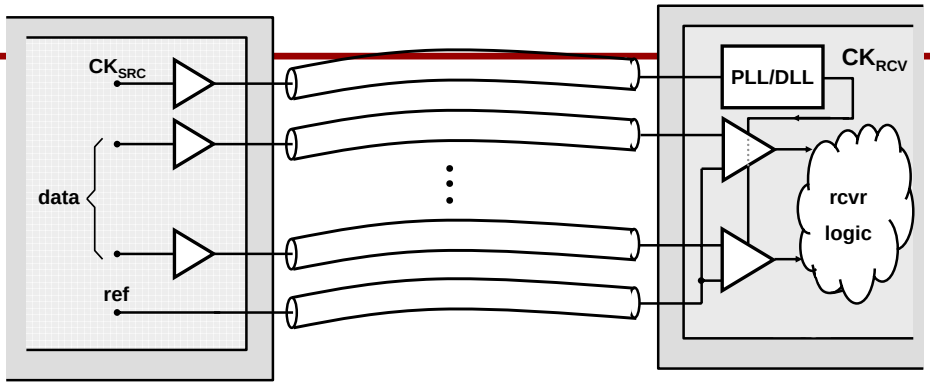
- Mesochronous
  Same frequency, unknown phase
  - Fast memories
  - Internal system interfaces
  - MAC/Packet interfaces

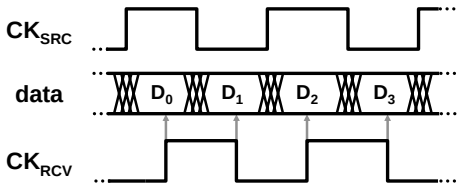


- Plesiochronous:
  Almost the same frequency
  – Mostly everything else today



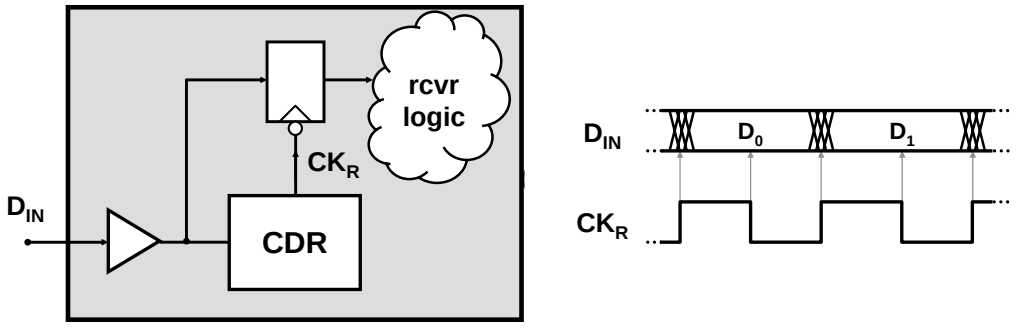
Source Synchronous Systems





– 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