

Modeling and Analysis of High-Speed Links

Vladimir Stojanovic^{1,2}, Mark Horowitz¹

¹ Stanford University

² Rambus Inc.

High-Speed In-k Research

- Used to focus on making chip fast
- Required precision timing—PLL
- Require HS transmitter, receiver circuitry
- Many papers on these topics

Low-Jitter Process-Independent DLL and PLL Based on Self-Biased Techniques

John G. Mauze

Abstract—Delay-locked loop (DLL) and phase-locked loop (PLL) designs based upon self-biased techniques are presented. The DLL and PLL designs achieve process technology independence, fixed damping factor, fixed bandwidth to operating frequency ratio, broad frequency range, input phase offset cancellation, and, most importantly, low input capacitance. Both the damping factor and the bandwidth to operating frequency ratio are determined completely by a ratio of capacitors. Self-biasing avoids the accuracy loss external loading, which can require special hardware bias circuitry, by generating all of the internal bias voltage and current from one another so that the bias levels are completely determined by the operating conditions. Fabricated in a 5.5- μ m 1.8V CMOS gate array process, the PLL achieves an operating frequency range of 6000 MHz to 200 MHz and input tracking jitter of 100 ps at 200 MHz with 100-mV of loop frequency spread over 1000 MHz.

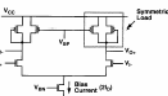


Fig. 1. Differential buffer stage with current sources.

I. INTRODUCTION
DELAY-LOCKED LOOP (DLL) and phase-locked loop (PLL) are often used in the IO interfaces of digital microprocessors in order to take clock distribution delay and to improve overall system timing. In these applications, DLLs and PLLs must closely track the input clock. However, the rising demand for high-speed IO has created an increasingly more demanding environment in which DLLs and PLLs must function. This more typically in the form of supply and voltage noise, tends to cause the output clocks of DLLs and PLLs to jitter from their ideal timing. With increasing voltage jitter in the downstream portion of the output clock, the design of low jitter DLLs and PLLs has become very challenging.
Achieving low jitter in PLL and DLL designs can be difficult due to a number of design tradeoffs. Consider a typical PLL which is based on a voltage-controlled oscillator (VCO). The amount of input tracking jitter produced as a result of supply and substrate noise is directly related to how quickly the PLL can correct the output frequency. To reduce the jitter, the loop bandwidth should be set as high as possible. Unfortunately, the loop bandwidth is affected by many process technology factors and is compromised by well before the lowest operating frequency is reached [1]. Those constraints can cause the PLL to have a narrow operating frequency range and poor jitter performance. Although typical PLLs are based on a delay line and thus require from a control perspective, a can have a broad delay range which trades to a certain extent that of the PLL.

This paper describes both a DLL and PLL design based upon self-biased techniques [2]. Self-biasing has numerous advantages:

- all of the process technology and environmental variability that affects PLL and DLL designs. Self-biasing can provide a bandwidth that tracks the operating frequency. This tracking bandwidth can in turn provide a very broad frequency range, extended supply and substrate noise induced jitter with a high input tracking bandwidth, and, in general, very robust designs. Other benefits include a fixed damping factor for PLLs, and input phase offset cancellation. Both the damping factor and the bandwidth to operating frequency ratio are determined completely by a ratio of capacitors giving effective process technology independence. The key to behind self-biasing is that it allows circuits to choose the operating bias levels in which they function best. By referencing all bias voltages and currents to other generated bias voltages and currents, the operating bias levels are essentially established by the operating frequency. The need for external biasing, which can require special hardware bias circuitry, is completely avoided.

This paper will begin by reviewing a differential buffer stage design that provides high supply and substrate noise rejection and allows the possibility of self-biasing. The loop architecture for self-biased DLL and PLL designs will be presented in Section II and Section IV, respectively. A transfer function for self-biased DLL and PLL designs will be presented in Section III and Section V, respectively. An important results compare and contrast to single-ended counterparts. The paper will also present some experimental results demonstrating the performance of the DLL and PLL designs.

II. DIFFERENTIAL BUFFER STAGE

In order to achieve low jitter operation, DLL and PLL designs require buffer stage designs with low supply and substrate noise sensitivity. The voltage controlled delay line (VCDL) and the VCO used in the DLL and PLL designs are

A 700-Mb/s/pin CMOS Signaling Interface Using Current Integrating Receivers

Shekhar Sidiroppan, Student Member, IEEE, and Mark Horowitz, Senior Member, IEEE

Abstract—A high speed CMOS signaling interface for applications in multiprocessor emerging hardware has been developed. The interface utilizes IV push-pull drivers, a delay line phase-locked loop (PLL), and sampling of the data on both edges of the clock. In order to maximize the noise immunity of the receiver, a current-integrating input pre-amplifier is used to reduce the incoming data. Clarity achieved is a 7.6-mV CMOS including average transfer rate of 700 Mbit/s using 700 pA 3.3-V supply with a bit error rate of less than 10⁻⁶.

Index Terms—Data communications, delay line phase-locked loop, digital communications, latency compensation, integrated circuit, multiprocessor, multipoint distribution, integrating receiver, mixed-signal integrated circuit.

1. INTRODUCTION

SCALING of semiconductor technology and advances in circuit design led to a rapid increase in the speed of digital and memory ICs. This created a demand for higher pin bandwidths on CMOS chips, which in turn led to the development of push-pull high-speed interfaces with driver system applications [1]–[5]. Although the developed interfaces utilize different transmission media and have diverse network configurations and line voltage swings, they all share a common characteristic which leads to increased noise sensitivity: the increasing data is amplified only once per bit period. This paper describes an interface design which increases this amplification by utilizing current integrating receivers to effectively double high frequency noise and increase noise immunity [7], [8]. Section II describes this interface architecture and the current integrating receiver. Section III introduces the concept of current integrating input pre-amplifier and describes in detail the implementation used in this design. Section IV describes the clocking circuit design including the delay line phase-locked loop (PLL) and the associated clock duty cycle adjusting circuit. Section V discusses the prototype measurement results and concluding remarks follow in Section VI.

II. SYSTEM ARCHITECTURE AND SIGNALING INTERFACE

In our system a bus-based signaling scheme was used along with one clock wire. The data is transmitted in pairs with the clock as depicted in Fig. 1. One unit of parallel information is transmitted every half period of the clock. Given that the transmission line lengths of the clock and data wires are approximately equal, the delay time is the same for both.

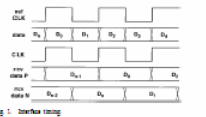


Fig. 1. Interface timing.

Equivalently equalized, the edges of the reference clock coincide with potential data transitions, so the receiver can use that timing information to position its sampling clock. Bus-based interfaces [1] usually have to synchronize their transmissions to an existing bus clock, thus requiring a transmitter DLL or PLL. In the point-to-point system, the reference clock is generated by the transmitter itself (by transmitting an alternating current data stream) thus eliminating the need for a transmitter clock alignment circuit. On the receiver side, the reference clock has to be amplified to full CMOS levels and then filtered up to drive the receiver of the incoming package data. A conventional receiver would position the sampling clock edges in the middle of the incoming data eye and use the sampling clock to sample the data once per half-clock period. In contrast, this design employs a delay-locked loop (DLL) to position the sampling clock in phase with the incoming data and average the data during the clock phase by employing current integration. At the end of the half-clock period, the receiver determines whether the incoming data was merely high or low based on the averaged value. This integration of the incoming data makes the receiver of the signal insensitive to high frequency supply or clock noise and improves the interface performance. It should be noted that using an integrating receiver is effective when the predominant limitation is noise as is the case with moderate length [CML] system interconnections. When the transmission is limited by the interconnect bandwidth, some form of channel equalization can be employed. For example, current integration is transmitted every half period of the clock. Given that the transmission line lengths of the clock and data wires are approximately equal, the delay time is the same for both.

Fig. 2 shows the block diagram of the interface. In order to minimize power consumption and simultaneously reduce noise, the transmitter uses a simple push-pull source termi-

A 0.5- μ m CMOS 4.0-Gbit/s Serial Link Transceiver with Data Recovery Using Oversampling

Chih-Kong Kim, Yung, Student Member, IEEE, Ramon Farías-Rodríguez, Student Member, IEEE, and Mark A. Horowitz, Senior Member, IEEE

Abstract—A 4-Gbit/s serial link transceiver is fabricated in a CMOS 1.8-Vm1.8V CMOS process. To achieve the high data rate without speed criticality in chip, the data are multiplexed using transmitter and immediately demultiplexed when received. This paradigm is achieved by using multiple phase output from a PLL using the phase output to determine the bit time. Using an 8:1 multiplexer (MUX) a CMOS, with a 20-MHz VCO ranging at 800 MHz. The internal logic operates at 200 MHz. For robust clock recovery, the signal is sampled at 5:1 bit rate and uses a digital phase-locked loop to recover the data. The digital phase locking is achieved by adjusting the clock rate to allow high-frequency bandwidth. With a 3.3-V supply, the chip has a measured bit error rate (BER) of 10⁻⁶.

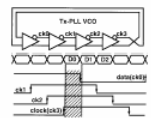


Fig. 1. System architecture.

The increasing demand for data bandwidth in networking has driven the development of high-speed and low-cost serial link technologies. Applications such as compression or compression-peripheral interconnection are requiring gigabit-per-second rates either over short distances or over longer distances in fiber. CMOS technology is used increasingly over CML or bipolar technologies because of the development toward faster and lower devices. In 0.18- μ m CMOS technology, the reduction in the number of equal or even that of the standard 0.5- μ m CMOS process. While other technologies are limited in the number of transistors due to yield or power, CMOS technology allows implementation of complex digital logic enabling more integration of the backend processing, lowering the cost. Recent development has shown CMOS capability to achieve 4.0-Gbit/s rates [1], [2], [3], [4], [5], [6]. This work pushes the state-of-the-art in the technology by achieving a 4.0-Gbit/s serial link transceiver and explore the issues involved.

II. ARCHITECTURE

A 0.5- μ m CMOS technology is used that is able to directly generate and receive a 4-Gbit/s stream (since the maximum ring oscillator frequency is <2 GHz). Instead, we use a PLL to reduce the performance requirements of each circuit. The transmitter generates the bit stream by an 8:1 multiplexer that multiplexes current pulses directly into the output channel (Fig. 1). The receiver (Fig. 2) performs a 1:8 demultiplexing by sampling with a bank of input amplifiers. Similar to the transmitter, each sample is triggered by individual clock phases. Furthermore, clock/data recovery is achieved by a 3:1 oversampling of each bit. Thus, the receiver requires a total of 24 clock phases to support both the oversampling and the 1:8 demultiplexing. Various techniques exist for generating multiple clock phases [7], [8]. The receiver side uses a voltage ring oscillator (VCO) [9], [10] followed by phase interpolators to generate intermediate phases [6][11] between the ring oscillator edges (0[11] 0[11]). Similar to the [1], [2], [3], [4], [5], [6], [7] clock control the transmitter multiplexing. A timing recovery circuit extracts the clock from the multiple samples per bit by finding the position of the data transitions. Once the transitions are determined, a decision logic selects the sample furthest from data decision (edge timing) as the recovered data byte. This approach is similar to

the entire transceiver chip is presented in Section V. Finally, some conclusions are drawn from these results in Section VI.

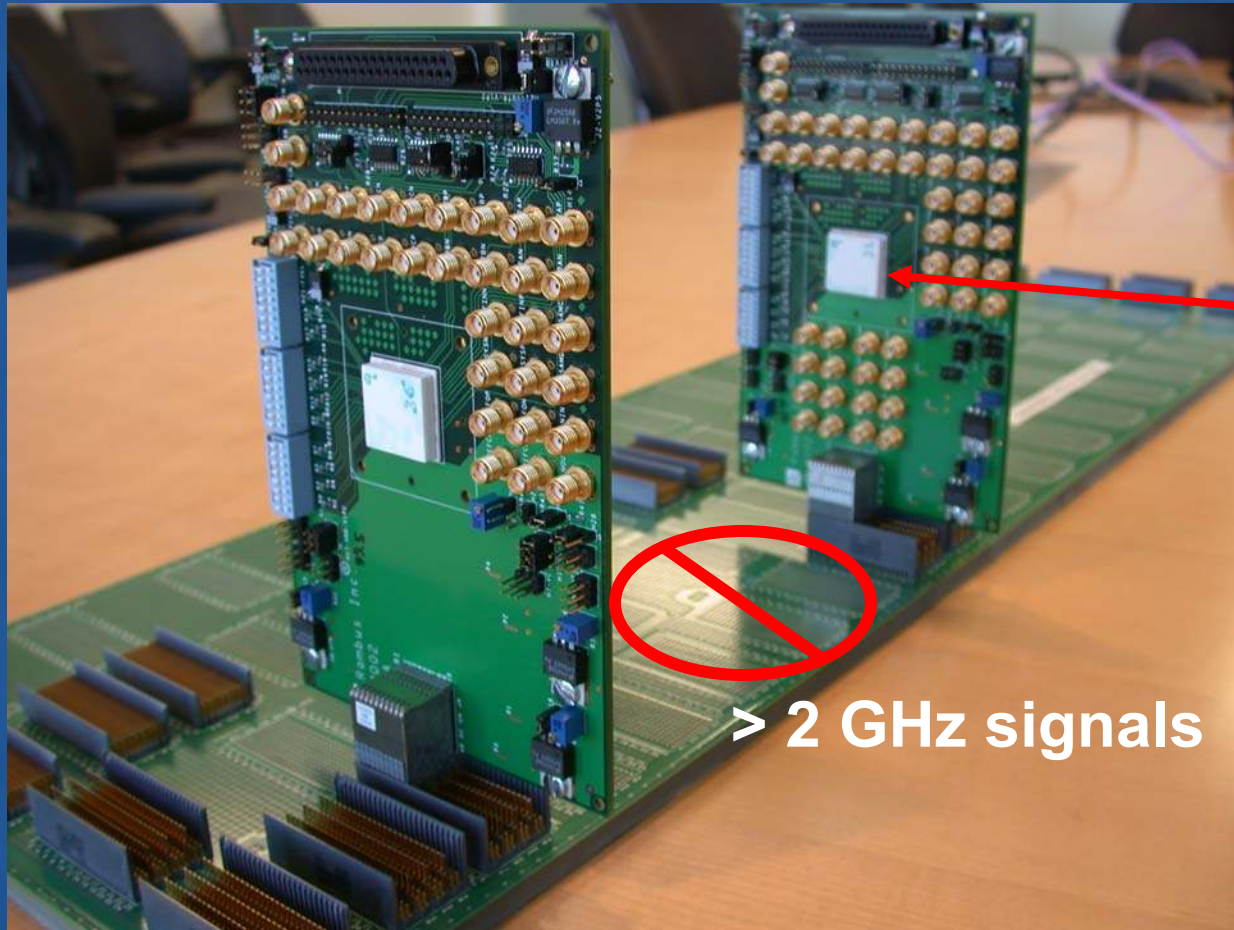
Manuscript received October 24, 2000; revised December 1, 2000. This work was supported by the Intel Corporation, Santa Clara, CA. Author's address: Intel Corporation, Santa Clara, CA 95051-2099, USA. E-mail: chihkong@intel.com.

Manuscript received October 24, 2000; revised December 1, 2000. This work was supported by the Intel Corporation, Santa Clara, CA. Author's address: Intel Corporation, Santa Clara, CA 95051-2099, USA. E-mail: chihkong@intel.com.

Manuscript received August 31, 2000; revised December 1, 2000. This work was supported by the Intel Corporation, Santa Clara, CA. Author's address: Intel Corporation, Santa Clara, CA 95051, USA. E-mail: shekhar@intel.com.

Manuscript received October 24, 2000; revised December 1, 2000. This work was supported by the Intel Corporation, Santa Clara, CA. Author's address: Intel Corporation, Santa Clara, CA 95051-2099, USA. E-mail: chihkong@intel.com.

Present Problem:



High speed
link chip

> 2 GHz signals

- Now, the bandwidth limit is in wires

New Link Research:

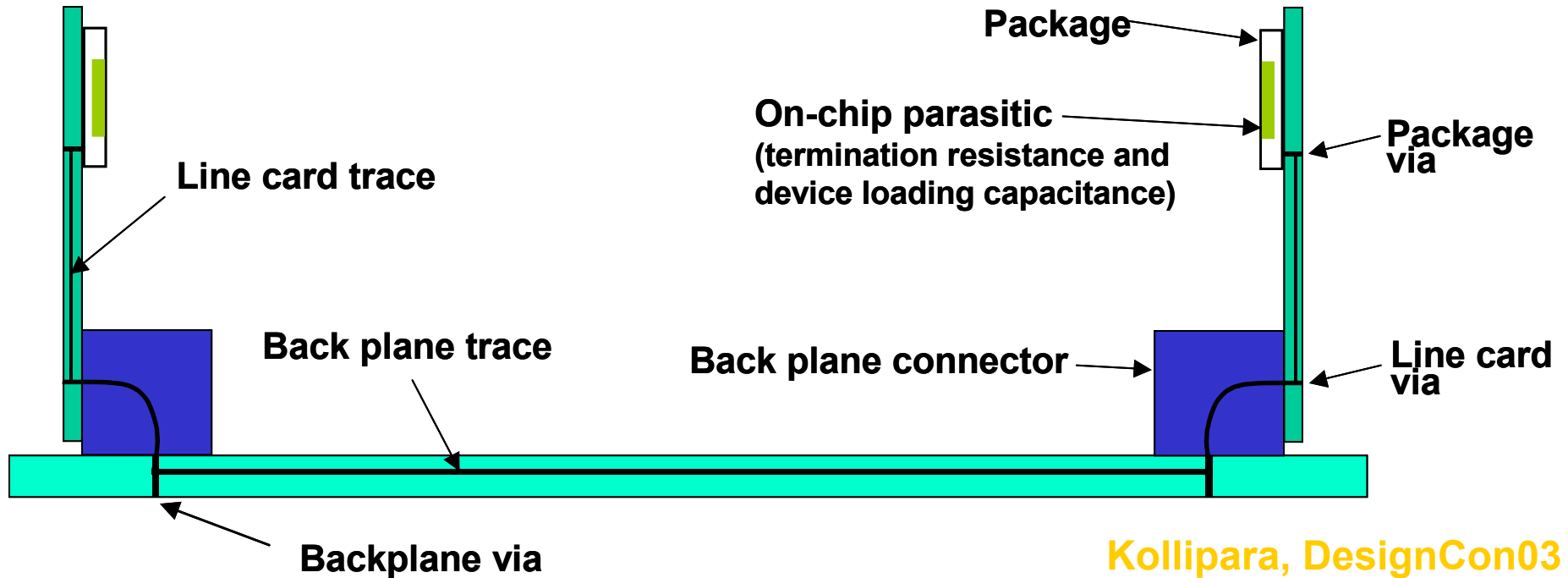
Dealing with bandwidth limited channels

- This is an old research area
 - Textbooks on digital communications
 - Think modems, DSL
- But can't directly apply their solutions
 - Standard approach requires high-speed A/Ds and digital signal processing
 - 20Gs/s A/Ds are expensive
- (Un)fortunately need to rethink issues

Outline Of This Talk

- Create a framework to evaluate trade-offs
 - For practical Gs/s digital communication systems
- Channel
 - How is the signal degraded?
- Noise (voltage and timing)
 - How large must the received signal be?
- Communication techniques
 - How much of the noise can be reduced
 - While maintaining a reasonable cost

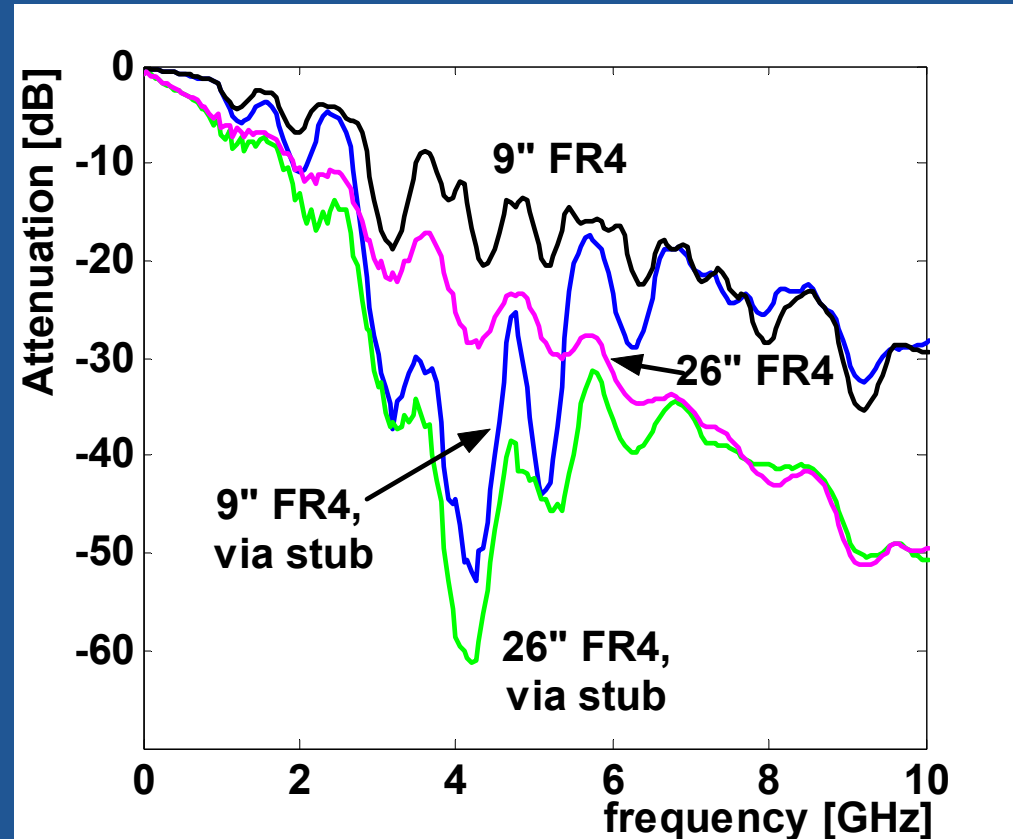
Backplane Environment



- 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?
- Required signal amplitude set by noise

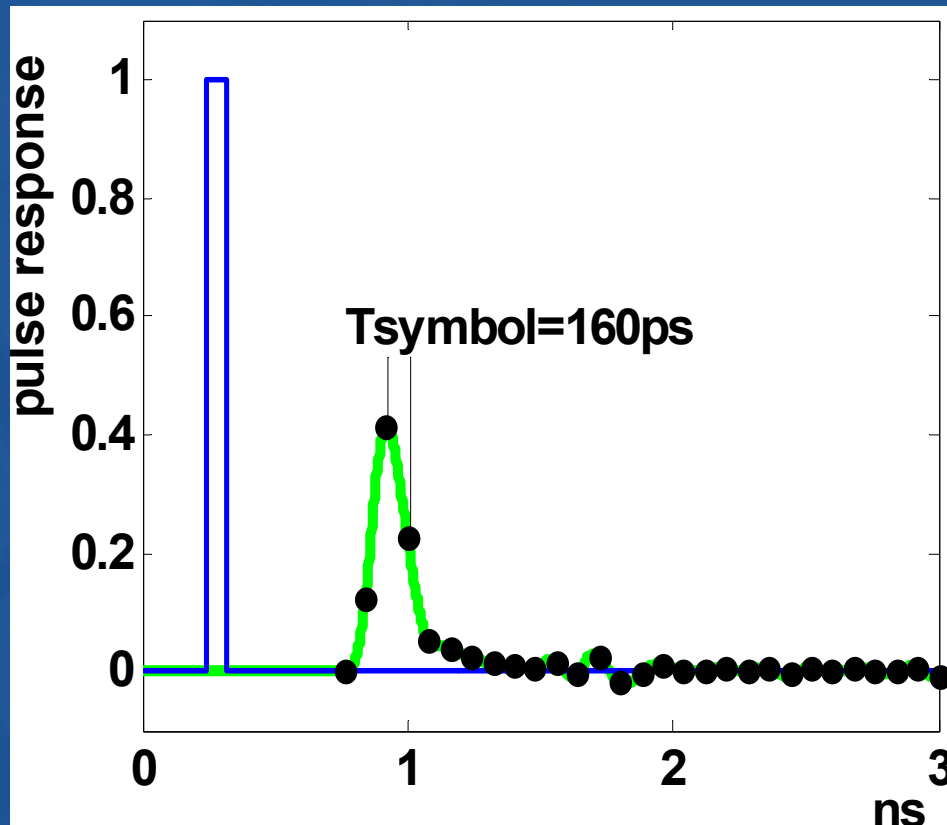


What We Will Call Noise

- Deterministic errors
 - Things we could in theory correct but don't
- Random noise
 - Have no choice
- Noise comes in two dimensions
 - Voltage
 - Timing
 - Will convert to an effective voltage noise

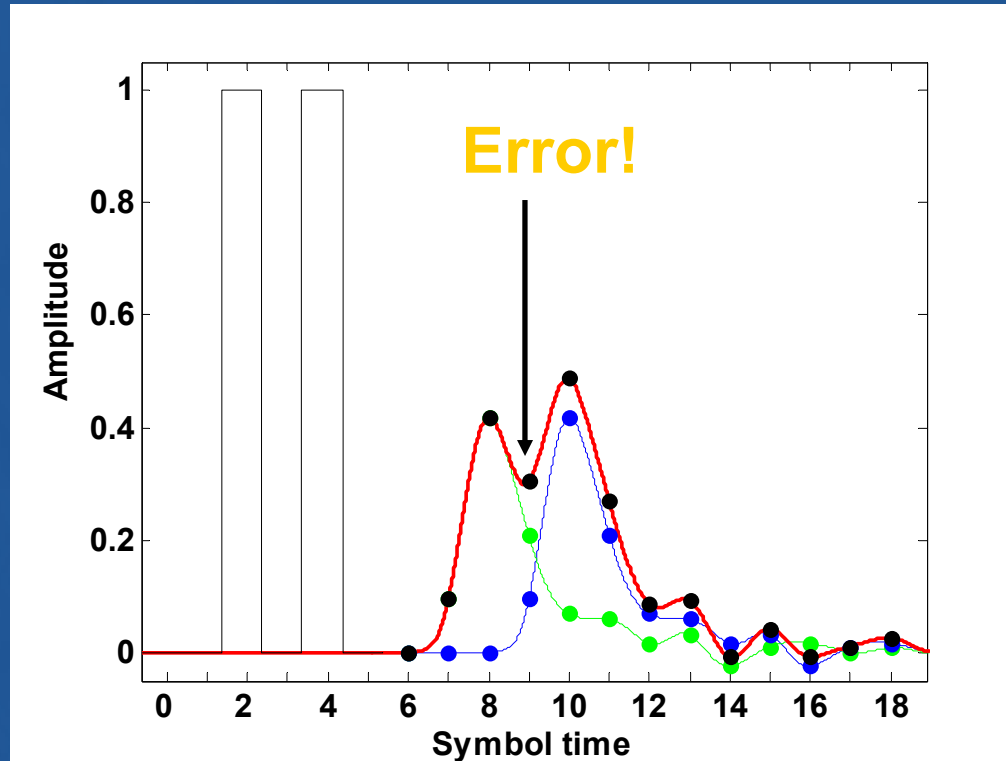
Inter-symbol Interference (ISI)

- 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

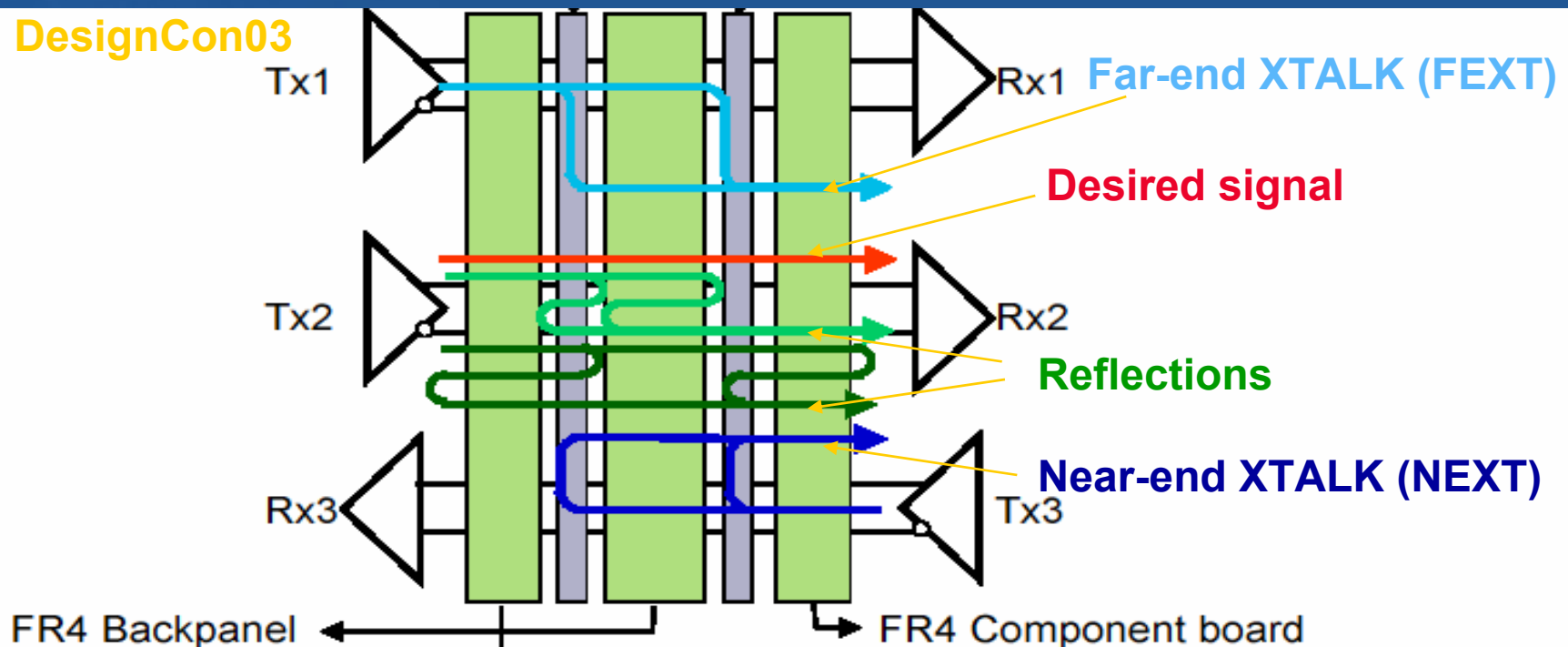


- Middle sample is corrupted by 0.2 trailing ISI (from the previous symbol), and 0.1 leading ISI (from the next symbol) resulting in 0.3 total ISI
- As a result middle symbol is detected in error

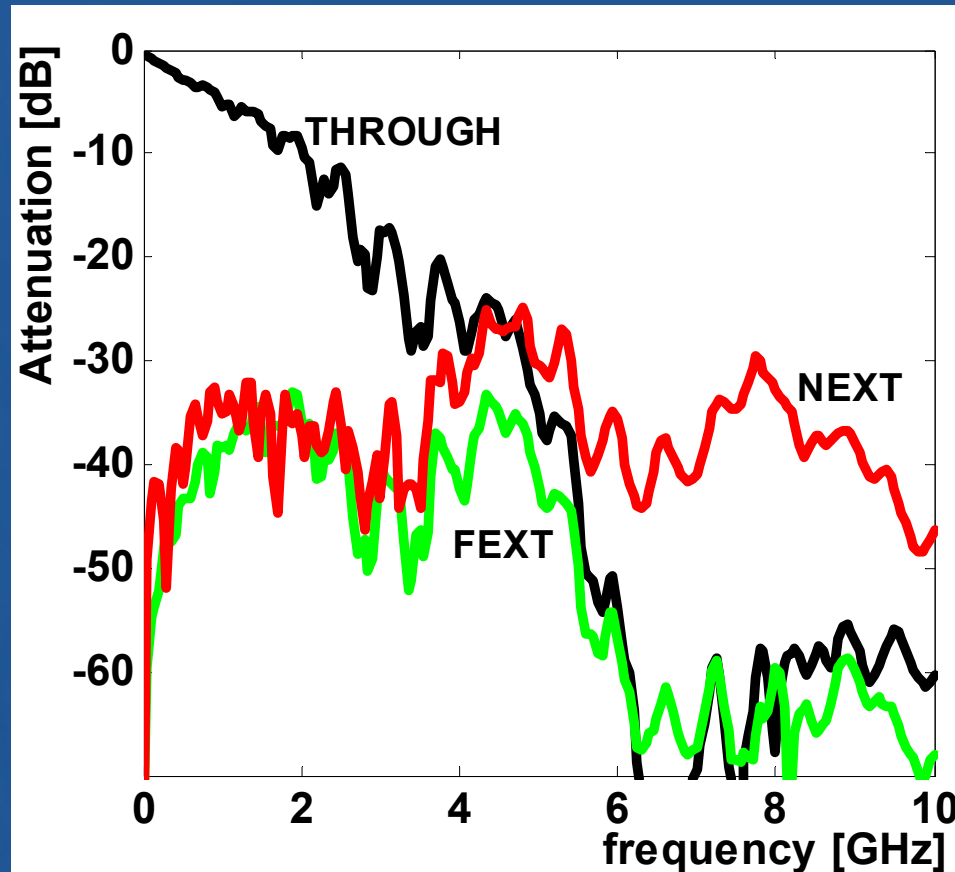
Crosstalk

- Don't just receive the signal you want
 - Get versions of signals "close" to you
 - Vertical connections have worse coupling
 - "Close" in these vertical connection regions

Sercu, DesignCon03



Frequency View of Crosstalk



- For this example:
 - > 4GHz, noise is as large as the signal

Random Voltage Noise

- Thermal noise
 - Resistor and Device noise
- Quantization
- Estimation error
- Supply noise
- Receiver offset

Modeling “Noise” Sources

Generally use one of two methods:

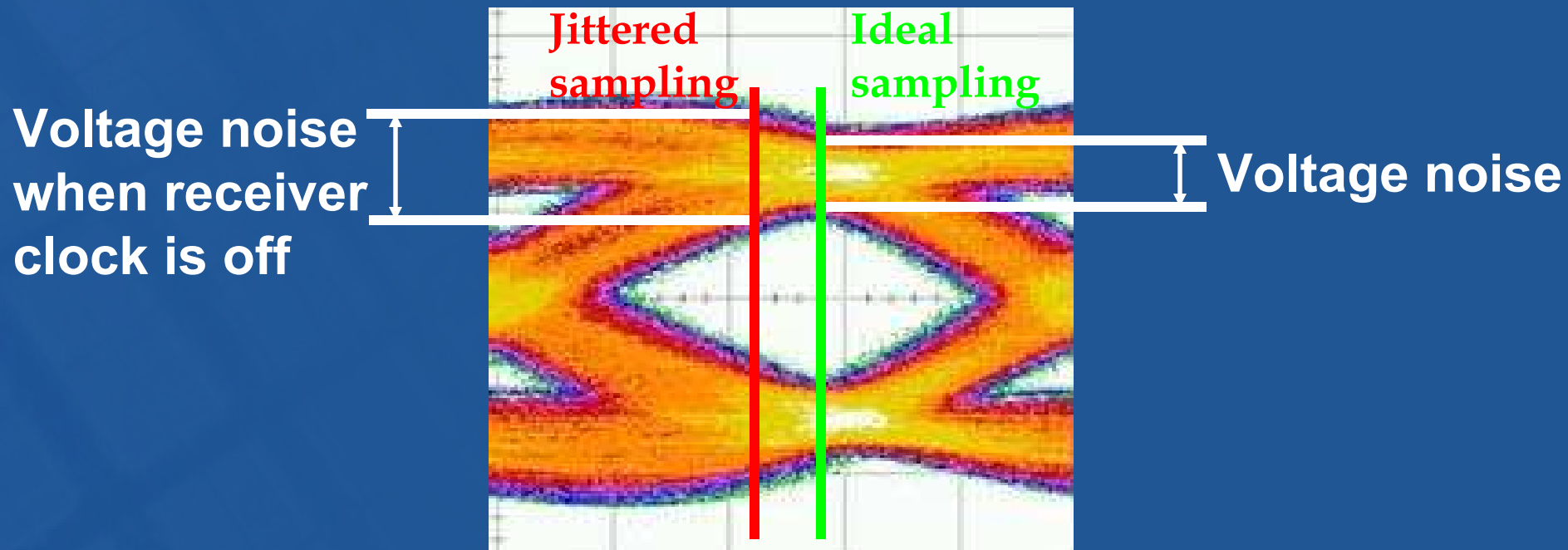
- Worst case analysis
 - Used for deterministic “noise” like ISI
 - Find worst case and subtract from signal
 - Then apply Gaussian Noise to result
- Assume Gaussian Distribution
 - Rely on Central-Limit Theorem
 - Most noise looks Gaussian, right?

Accuracy Issues

- Worst case analysis
 - Can be too pessimistic
 - If probability of worst case very small
- Gaussian distributions
 - Works well near mean
 - Often way off at tails
 - E.g. ISI distribution is bounded
- We will use direct noise statistics

Effect of Timing Noise

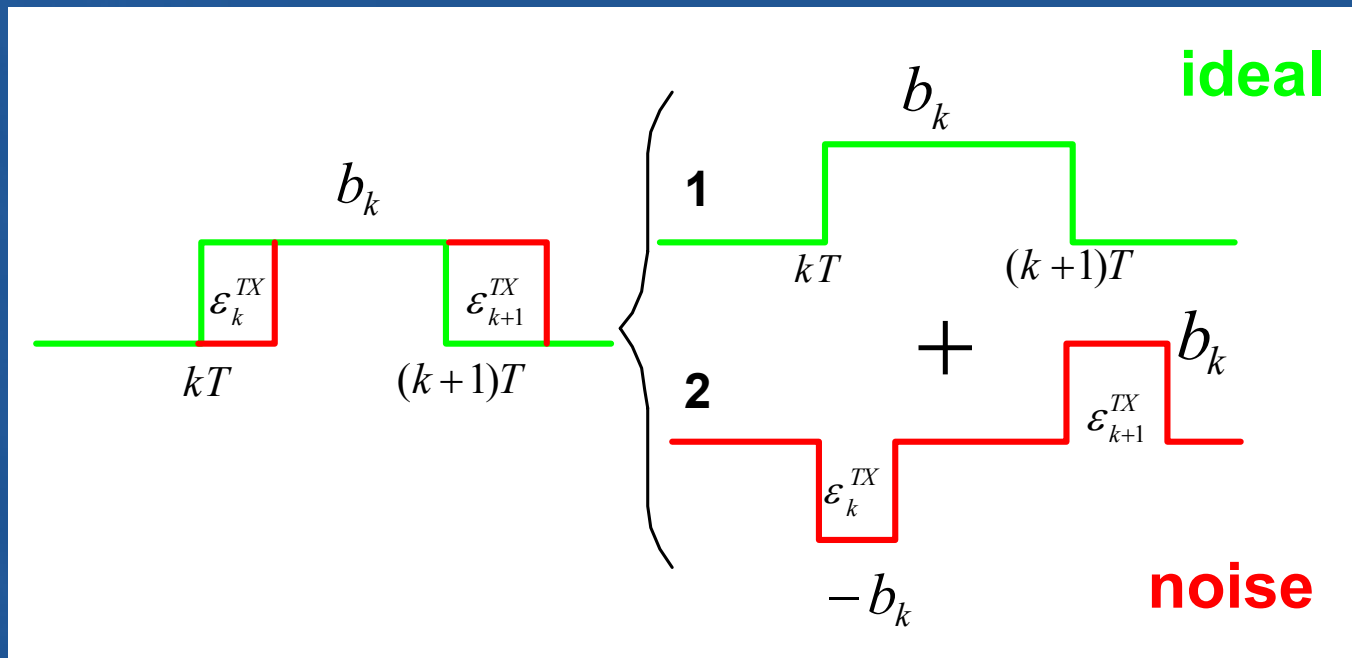
- Need to map from time to voltage



- The effect is going to depend on the size of the jitter, the input sequence, and the channel

Effect of Transmitter Jitter

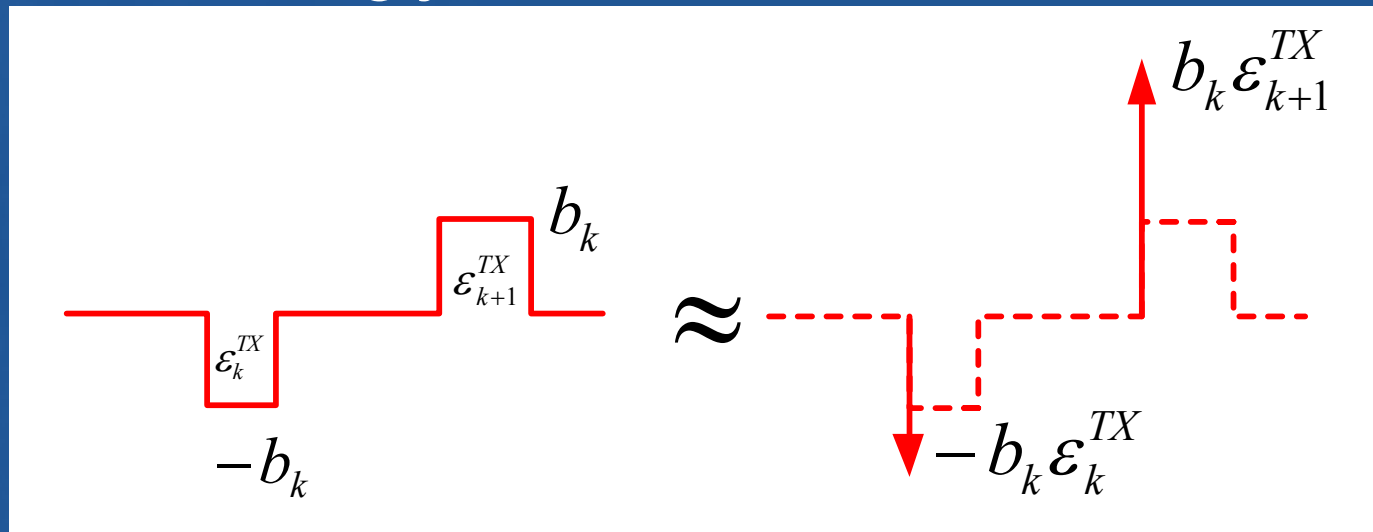
Jittered pulse decomposition



- Decompose output into ideal and noise
- Noise are pulses at front and end of symbol
 - Width of pulse is equal to jitter

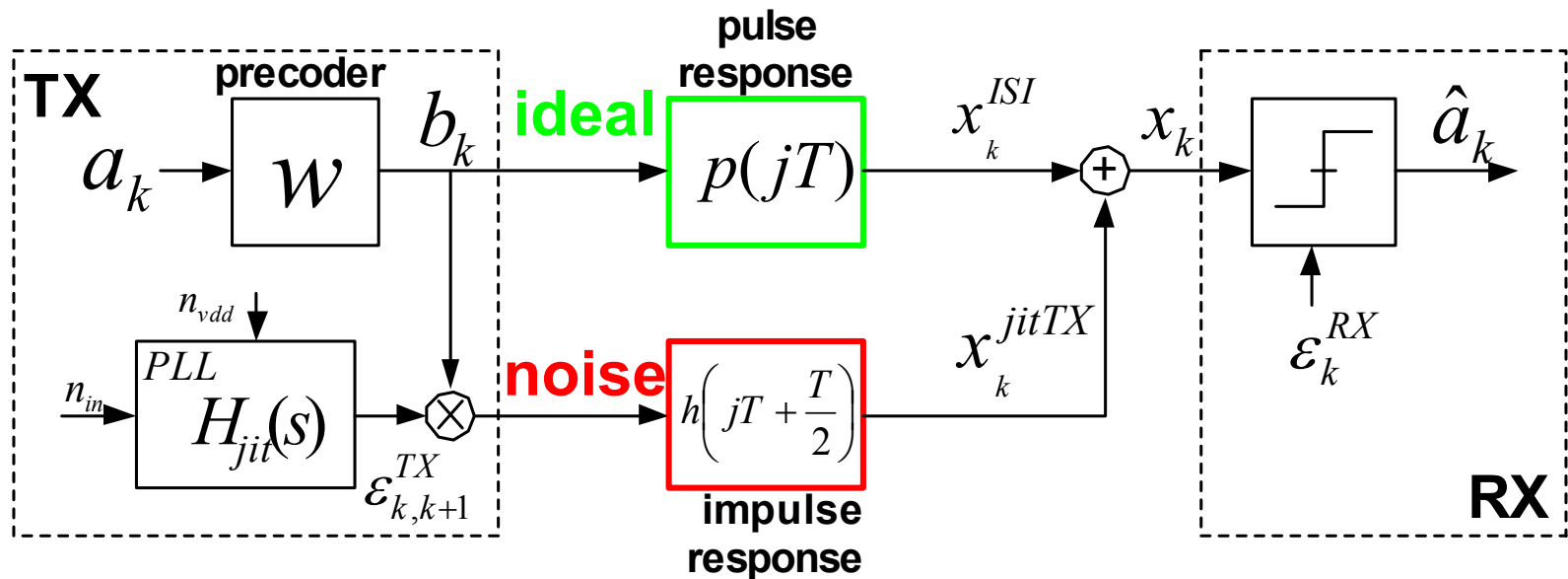
Transmitter Jitter Noise

- Approximate the noise pulses with deltas
 - Assuming jitter is small



- Channel output
 - Output with no jitter
 - Response to the noise deltas

Jitter Propagation Model



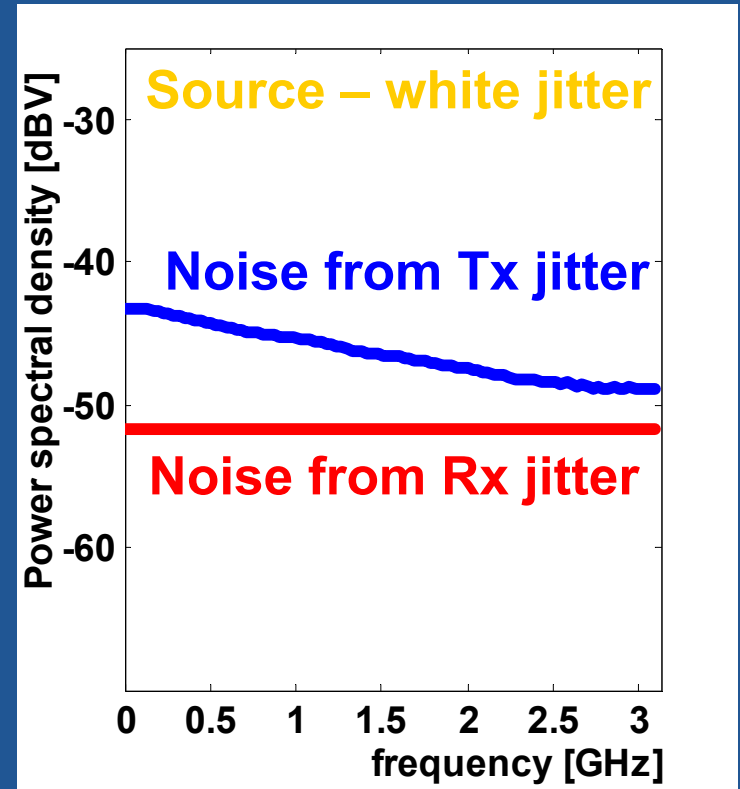
- Channel bandwidth matters
 - If $h(T/2)$ is small, the noise is small
 - $h(nT+1/2)$ not small, many pulses add

Jitter Effect On Voltage Noise

- Transmitter jitter
 - High frequency (cycle-cycle) jitter is bad
 - Changes the energy (area) of the symbol
 - No correlation of noise sources that sum
 - Low frequency jitter is less bad
 - Effectively shifts waveform
 - Correlated noise give partial cancellation
- Receive jitter
 - Modeled by shift of transmit sequence
 - Same as low frequency transmitter jitter

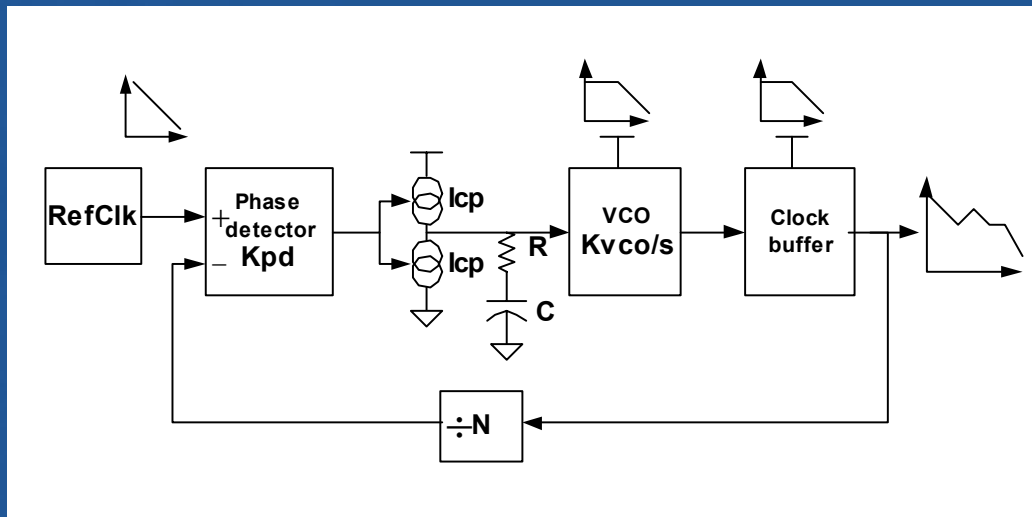
Voltage Noise From Jitter

- White jitter
 - Noise from Tx much larger than from Rx jitter
 - From Rx jitter, noise is white
 - From Tx jitter, filtered by the channel
- Y-axis is noise σ (in Volts)
 - If the noise was white
 - $\sigma = 10\text{mV} \Rightarrow -40\text{dBV}$
- Bandwidth of the jitter is critical
 - It sets the **magnitude** of the noise created



Jitter Source From PLL Clocks

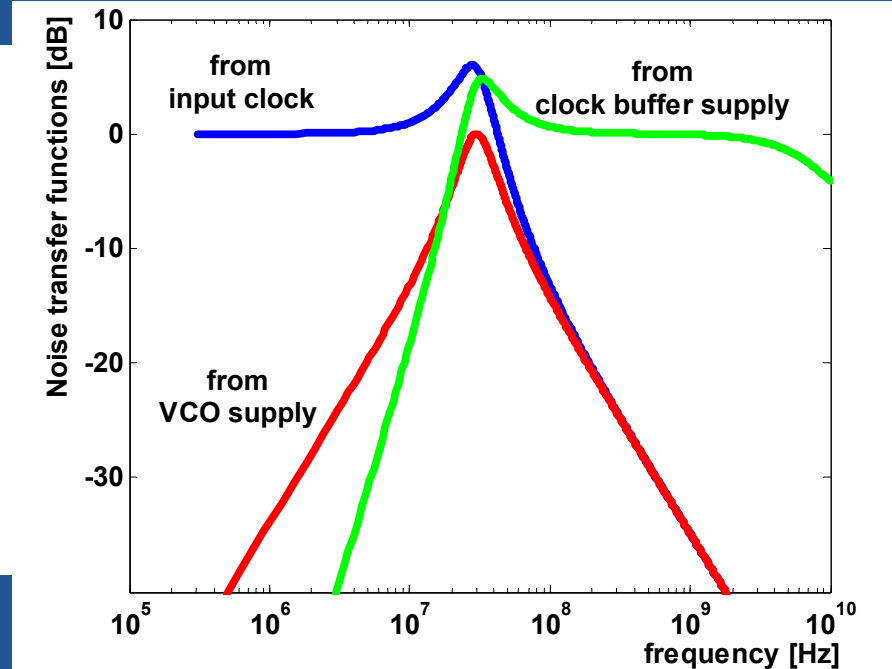
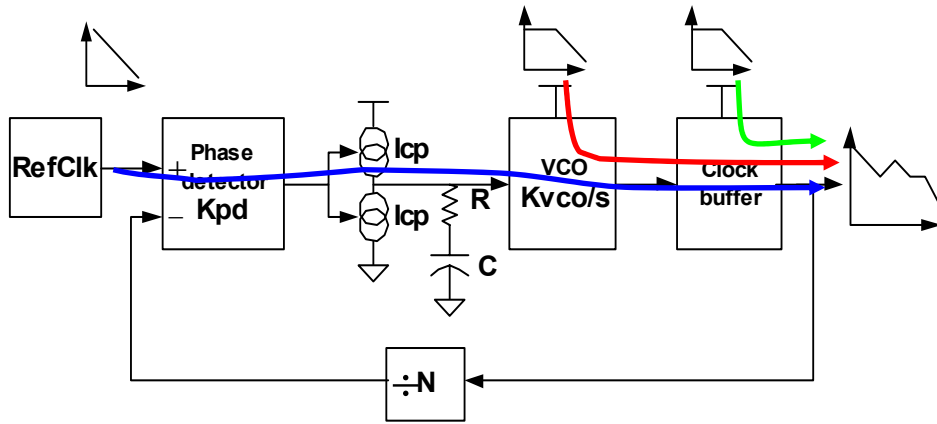
- Noise sources
 - Reference clock phase noise
 - VCO supply noise
 - Clock buffer supply noise



Mansuri '02

- Stationary phase-space model

Noise Transfer Functions

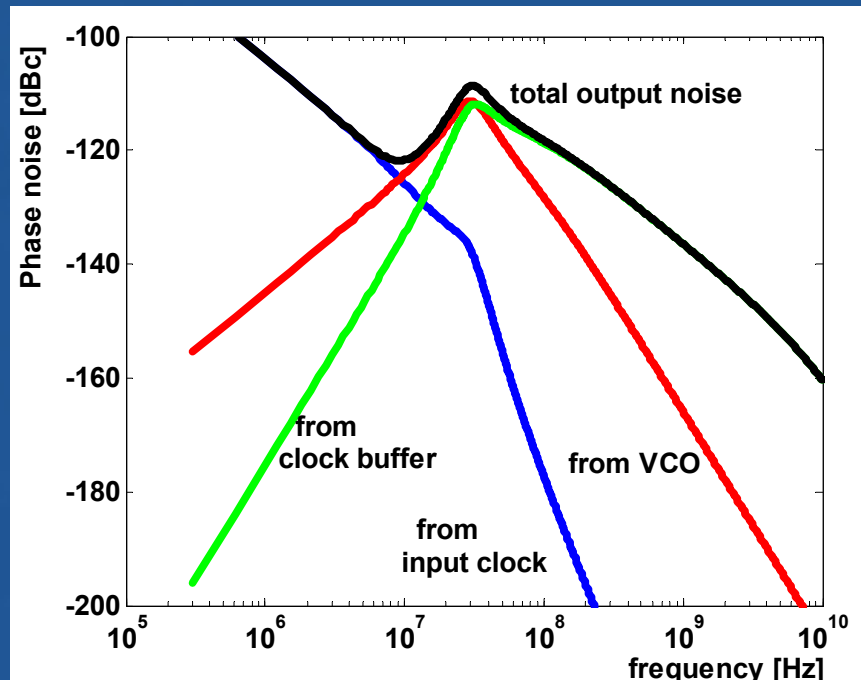


- Low-pass from reference (input clock)
- Band-pass from VCO supply
- High-pass from clock buffer supply

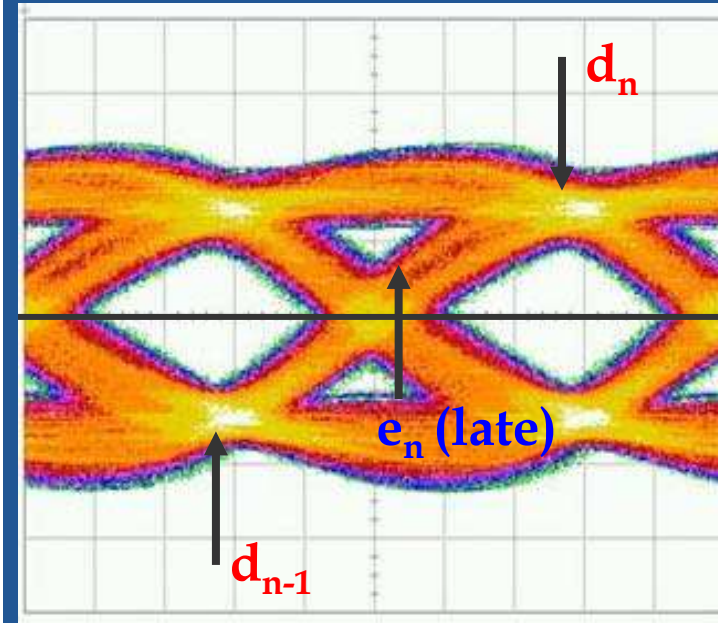
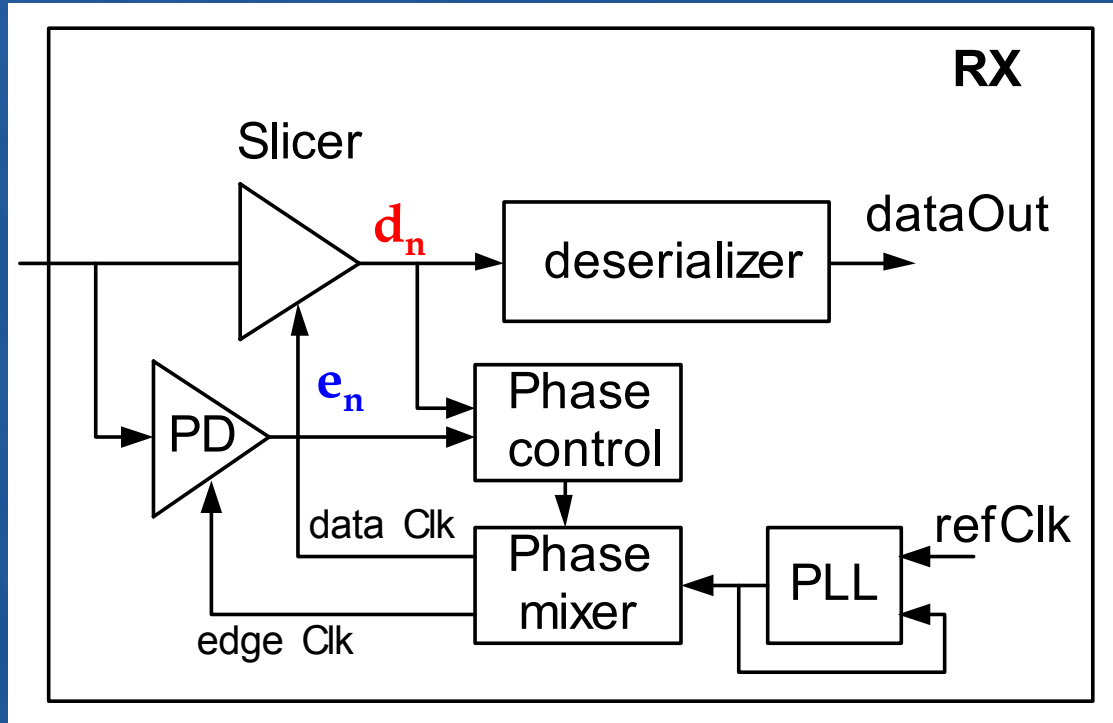
Jitter Spectrum

- Set by reference clock $f < \text{loop bandwidth}$
- Set by supply noise $f \geq \text{loop bandwidth}$

Spectrum using 100MHz BW power supply noise



2x Oversampled Bang-Bang CDR

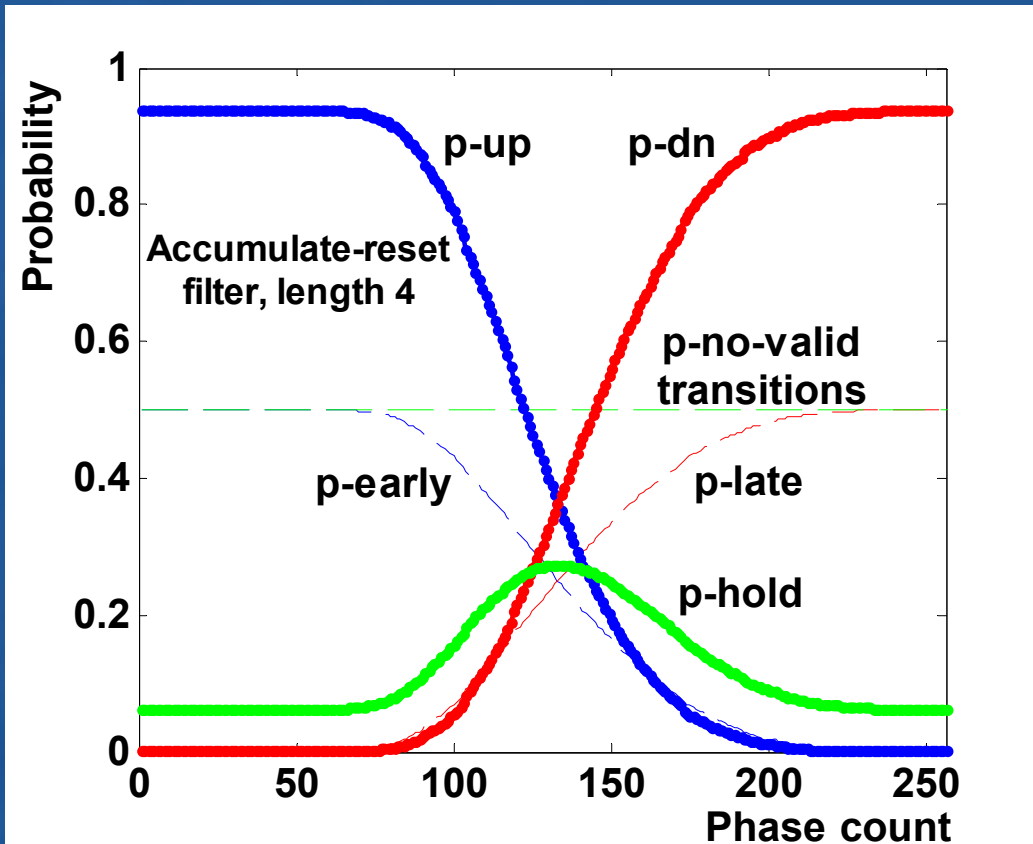


- Generate **early/late** from d_n, d_{n-1}, e_n
 - Simple 1st order loop, cancels receiver setup time
- Now need jitter on data Clk, not PLL output

Data Clk Noise

- Model phase selector and PLL
 - Base linear PLL jitter
 - Add non-linear phase selector noise from CDR
- Model the CDR loop as a state machine
 - The current phase position is the state
 - State transitions are caused by early/late
 - Jitter on input data and PLL means
 - Possible to be late and get early PD result
 - Often filter early/late to generate up/down

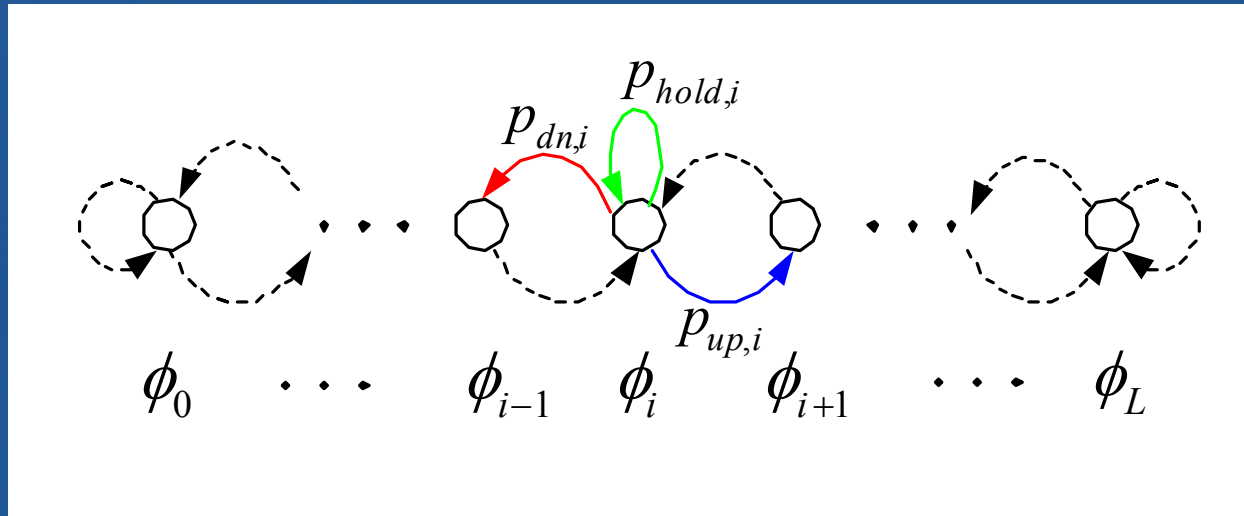
Transition Probabilities



- Example system:
 - CDR loop
 - Residual ISI
 - At edge -30dBV
 - Desired phase
 - State = 133

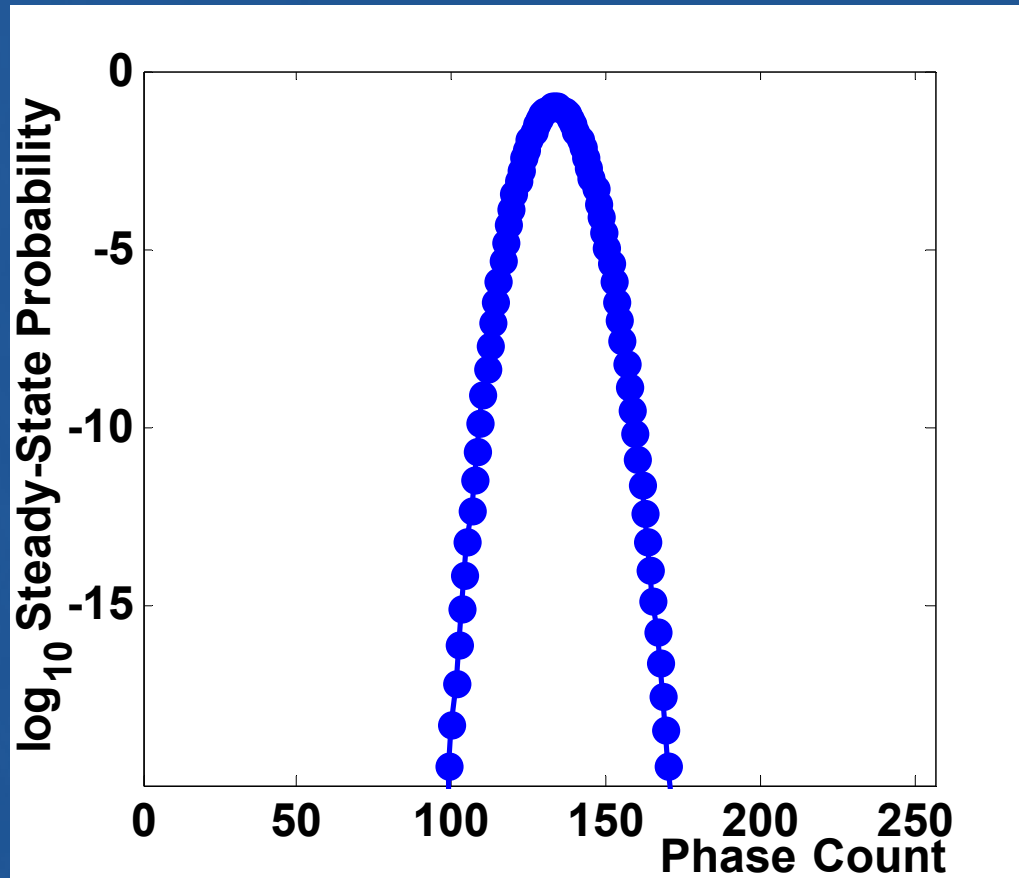
- On average move to correct position
 - But probability of wrong movement is not small
 - Need to find probability of at each phase location

Bang-Bang CDR Statistical Model



- Need steady state probabilities of the states
 - Have the transition probabilities
- Iteratively apply transition probabilities
 - Results will converge to a steady-state

Bang-Bang CDR Model



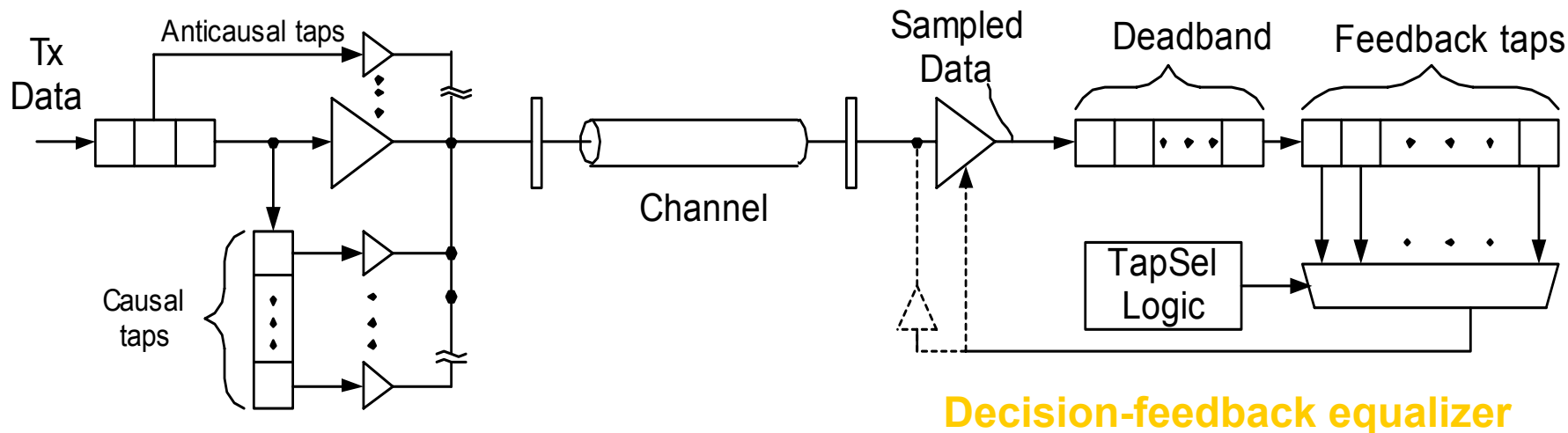
- Gives the probability distribution of phase
 - Which is the CDR jitter distribution

Noise Summary

- Many important sources of noise
 - ISI, crosstalk, quantization, estimation, etc.
- Largest noise comes from ISI
 - By factor of 10x
- Timing is noisy too
 - High frequency transmitter jitter is bad
 - CDR jitter needs to be considered
 - Especially if the data input is noisy
- How much noise can we eliminate?

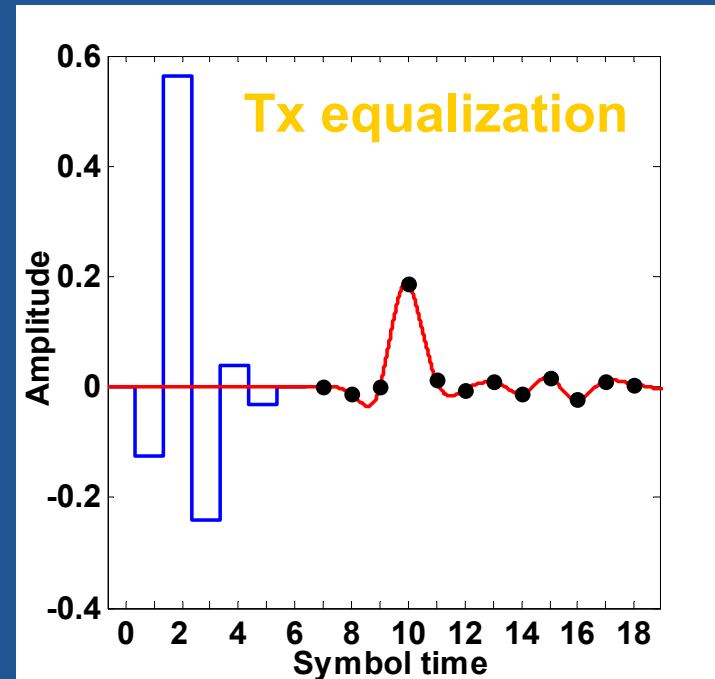
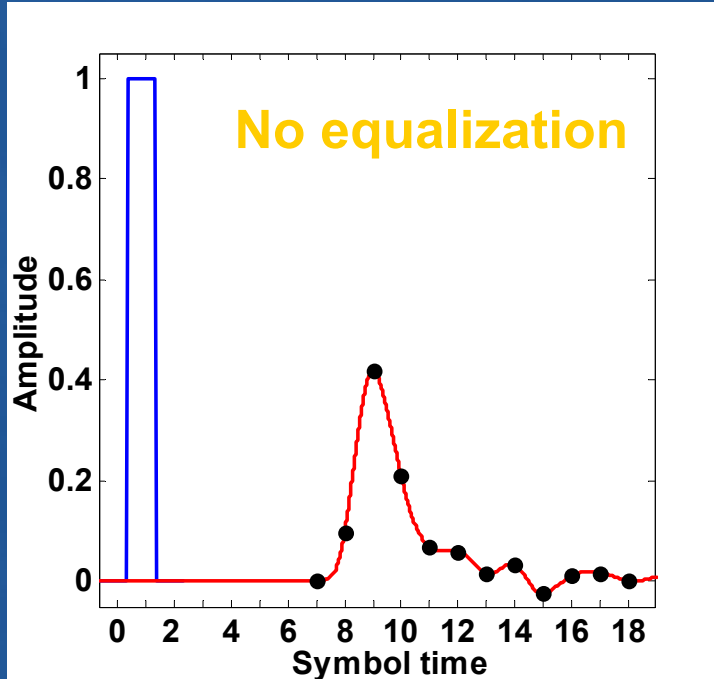
Removing ISI

Linear transmit equalizer



- Transmit and Receive Equalization
 - Changes signal to correct for ISI
 - Often easier to work at transmitter
 - DACs easier than ADCs

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

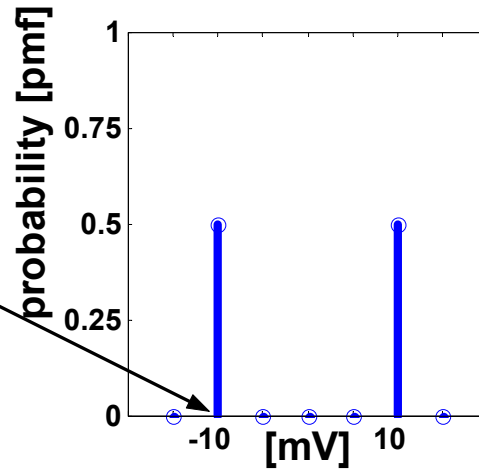
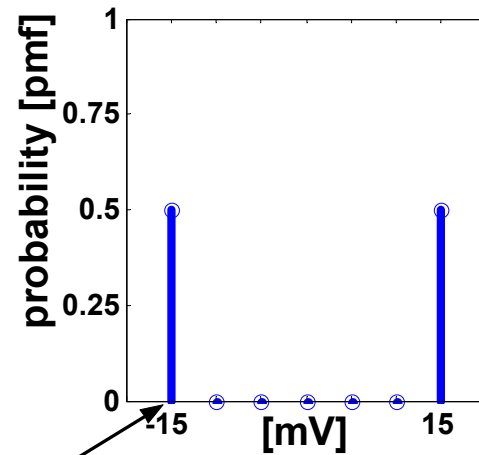
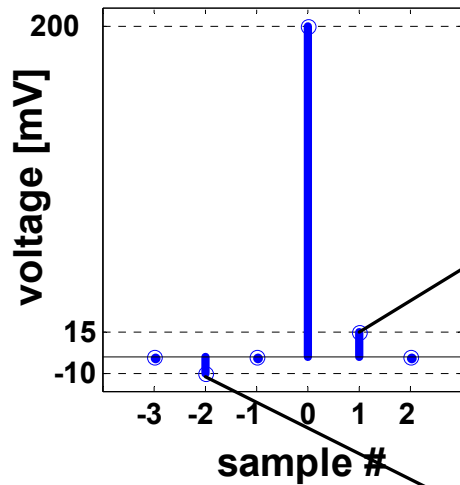
Residual Error

- Cannot correct all the ISI
 - Equalizers are finite length
 - EQ coefficients quantized
 - Channel estimate error
- The error affects both voltage and timing
- Need to find the distribution of this error

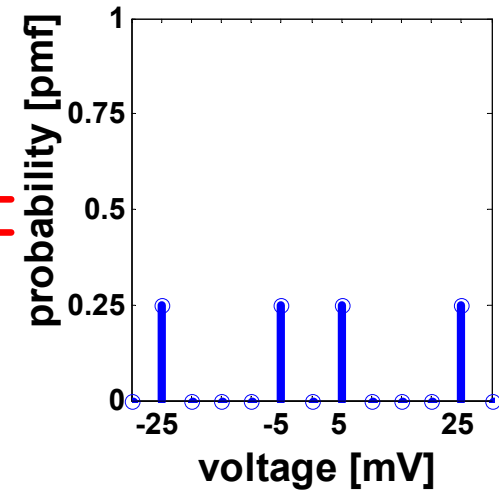
Generating ISI Distributions

Convolution method

Tx Equalized pulse response

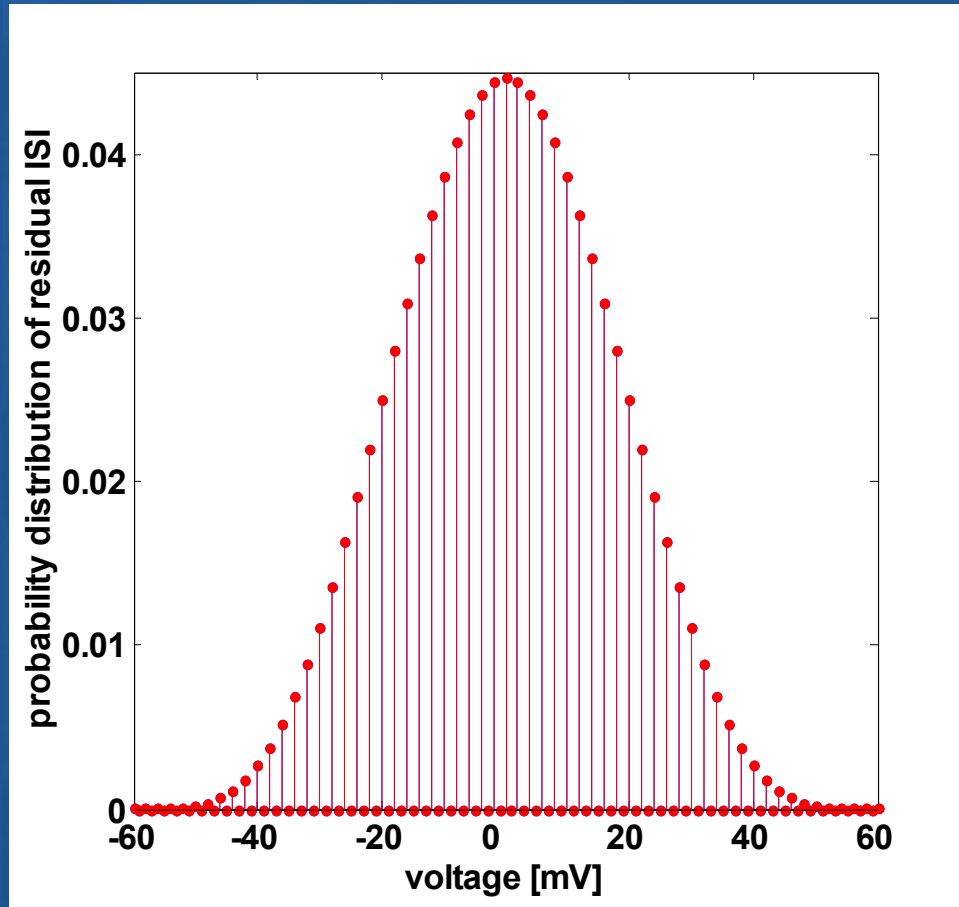


ISI distribution

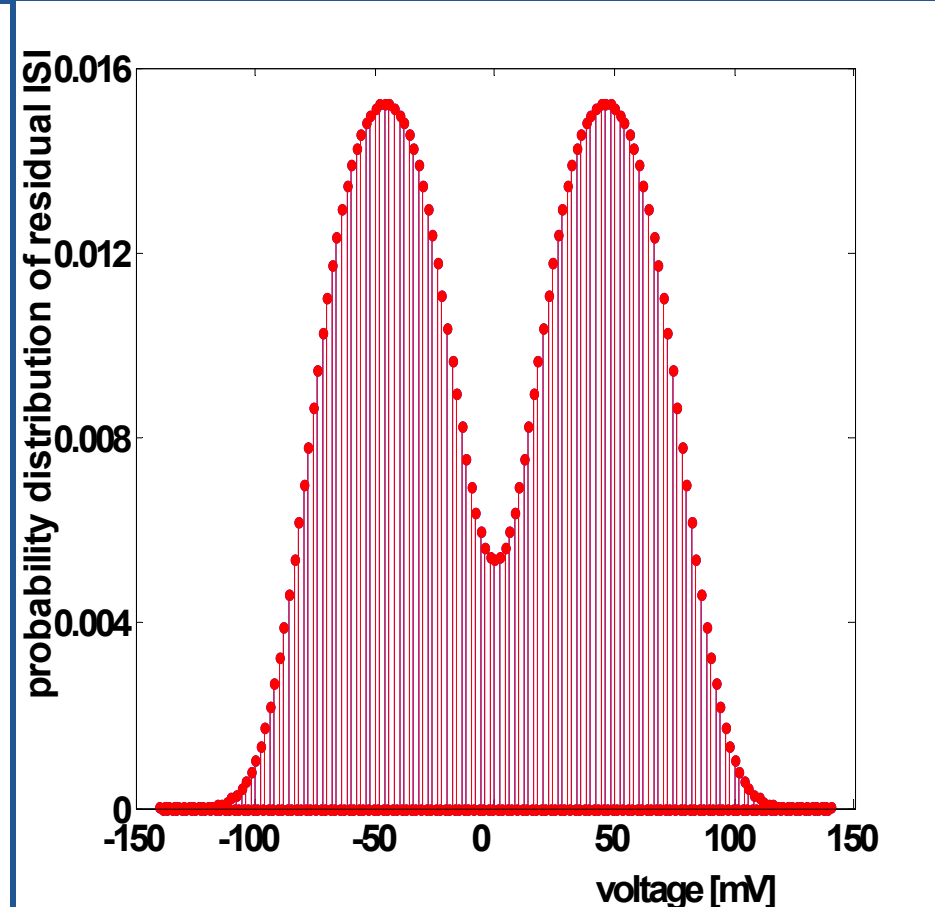


Estimated Residual Error

5 Tap Transmitter Equalizer



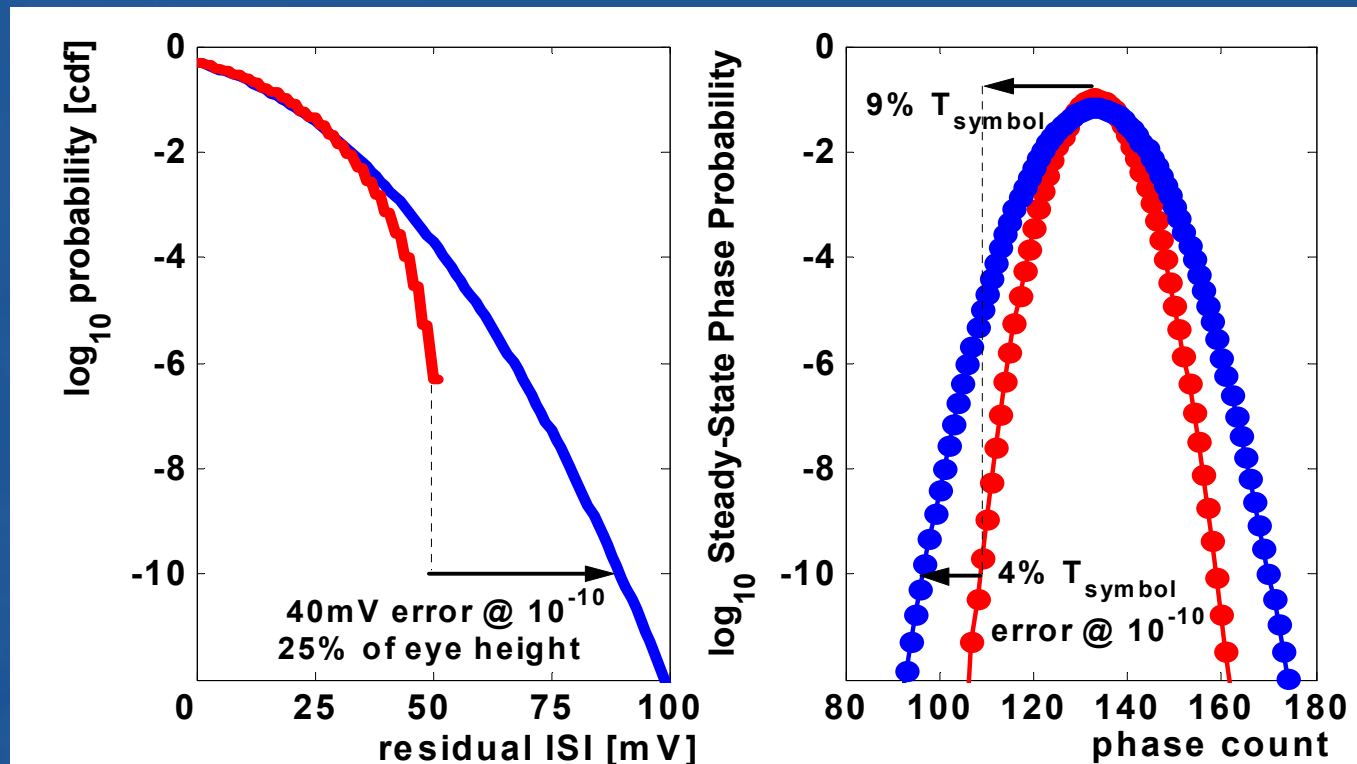
Data sample distribution



Edge sample distribution

Comparison w/ Gaussian Model

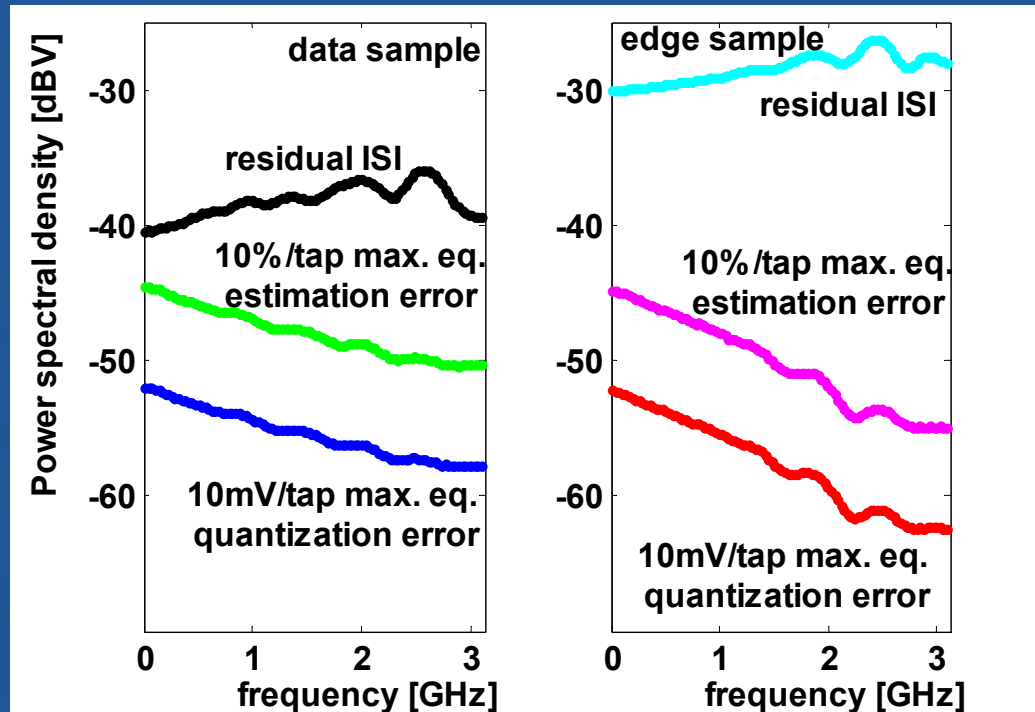
Cumulative ISI distribution



Impact on CDR phase

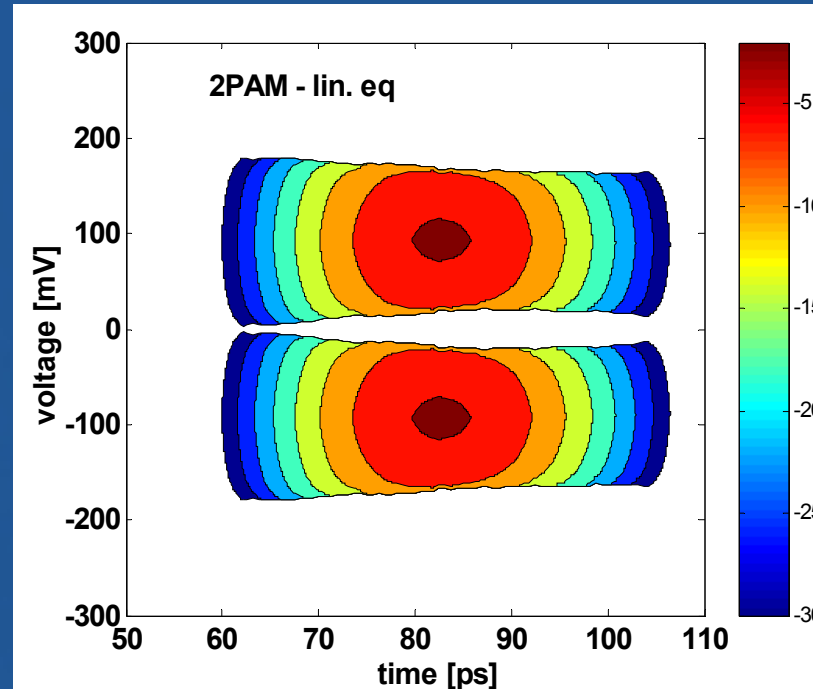
- Gaussian model only good down to 10^{-3} probability
- Way pessimistic for much lower probabilities

Equalizer Related Error Sources



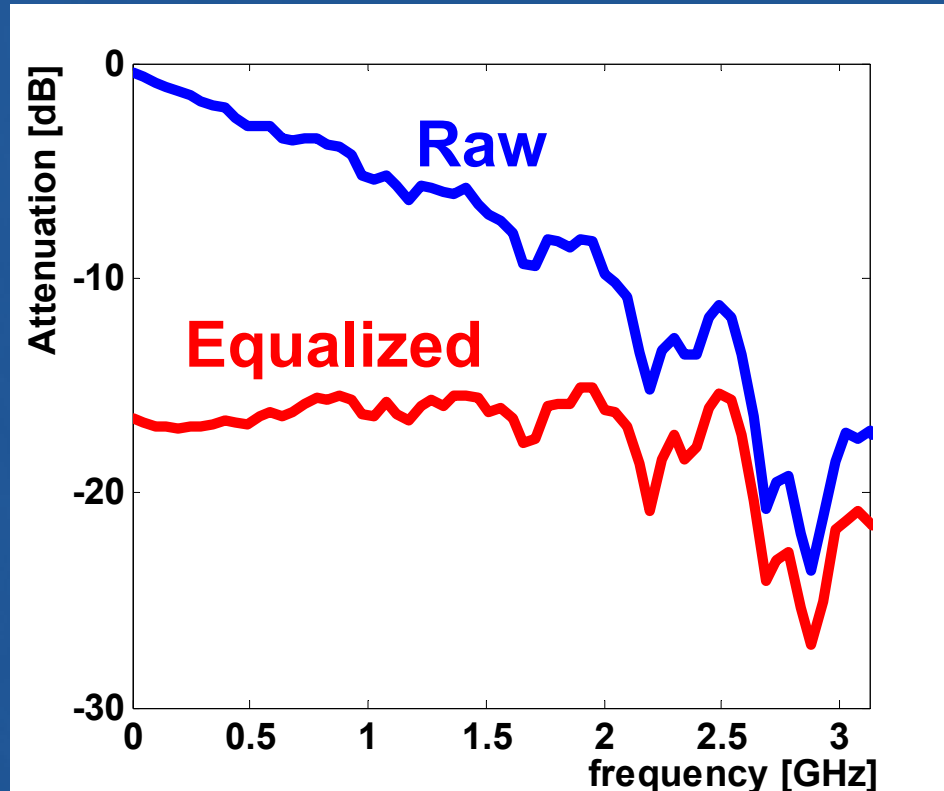
- Residual ISI is the biggest source of error
- Quantization error and equalizer estimation
 - Are significant for reasonable assumptions about accuracy

ISI and CDR Phase Distributions



- In ideal world, there would be only two dots
 - This plot shows how these dots spread out
- Vertical slice – ISI distribution per time offset
- Horizontal weight – CDR phase distribution

Tx Equalization

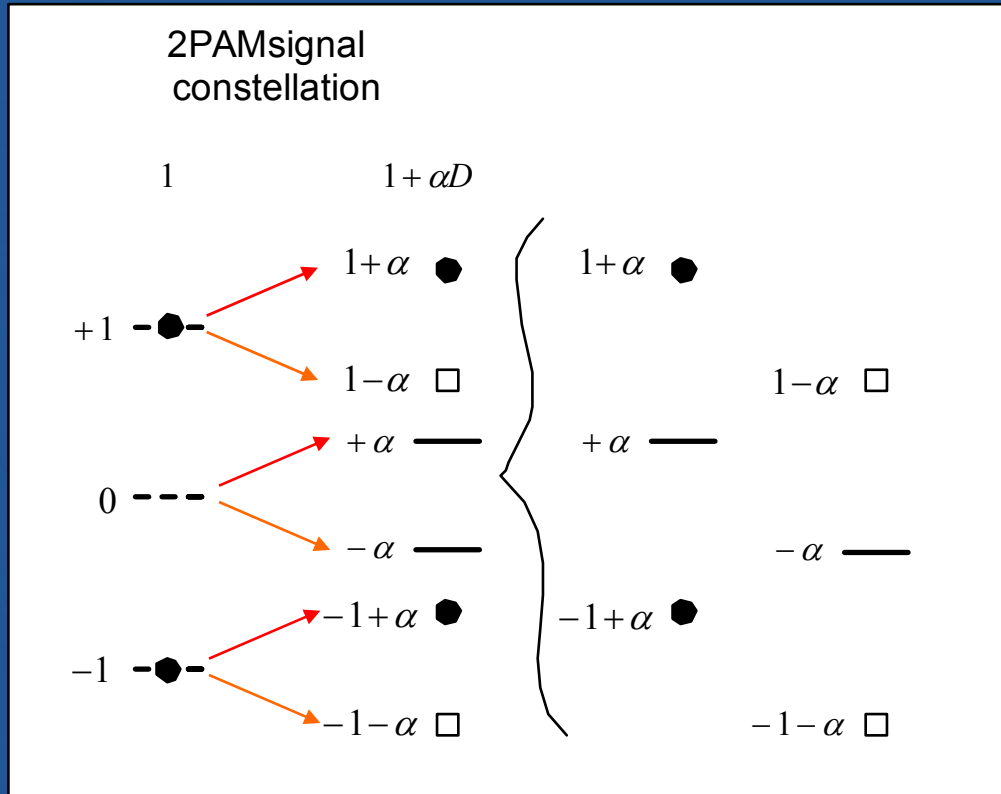


- Transmit equalization attenuates low frequencies
 - Output swing is constrained (peak power constraint)
 - Reduces ISI, but also decreases SNR (decreases signal)

Receive Equalization

- Feedback equalization (DFE)
 - Subtracts error from input
 - No attenuation
- Problem with DFE
 - Need to know values of interfering bits
 - ISI must be causal
 - And latency in the decision circuit is a problem
 - Receive latency + DAC settling < bit time
 - Can increase allowable time by loop unrolling
 - Receive next bit before the previous is resolved

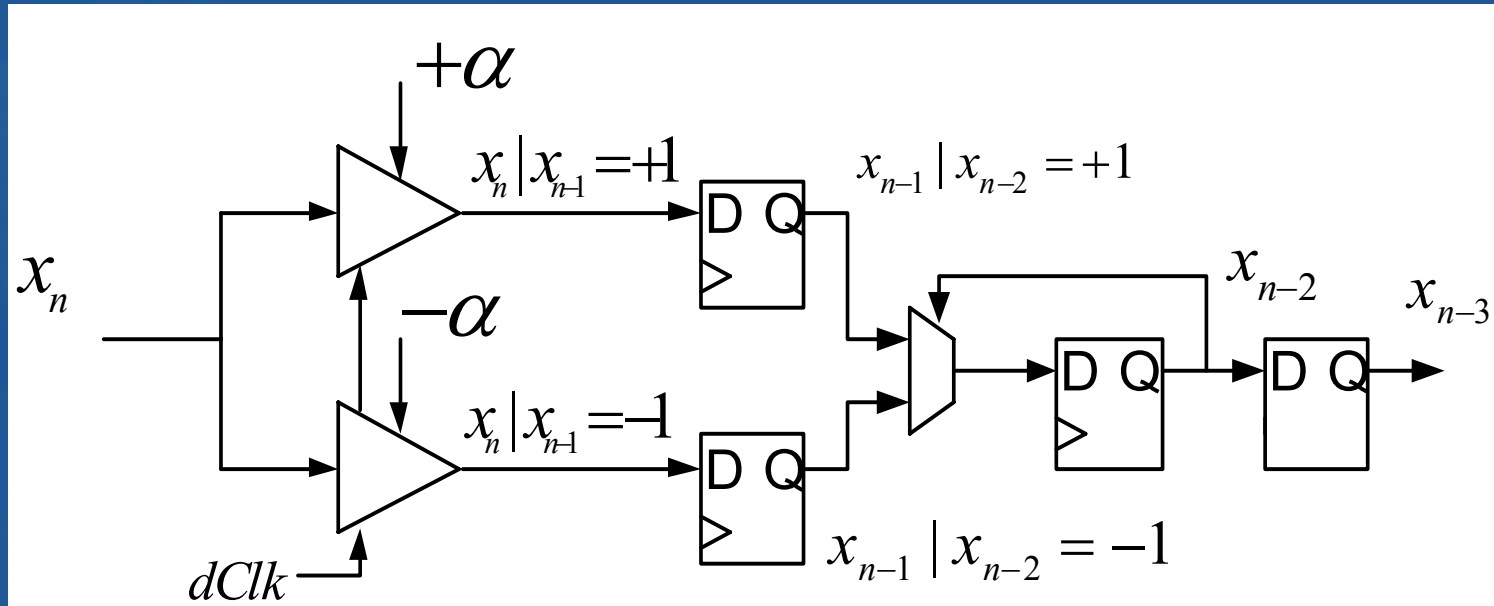
1 Bit Loop Unrolling



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

Loop Unrolling Implementation

Parhi '90 and Kasturia '91



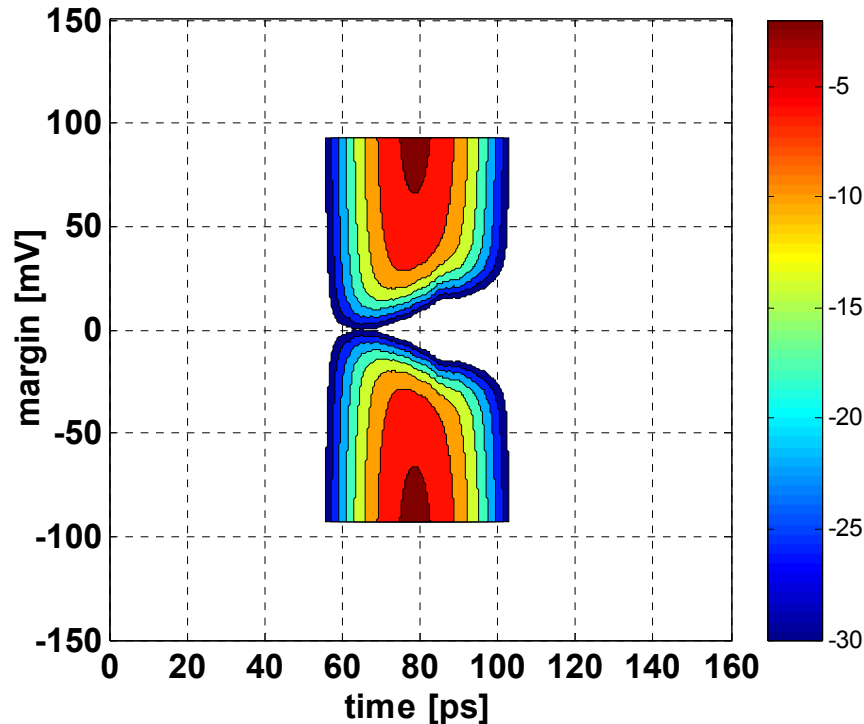
- Offset slicer levels by $\pm \alpha$
 - Previous symbol selects correct value
- M^L receivers for L taps, M level signal
 - Each receiver with M-1 comparators

Putting It All Together

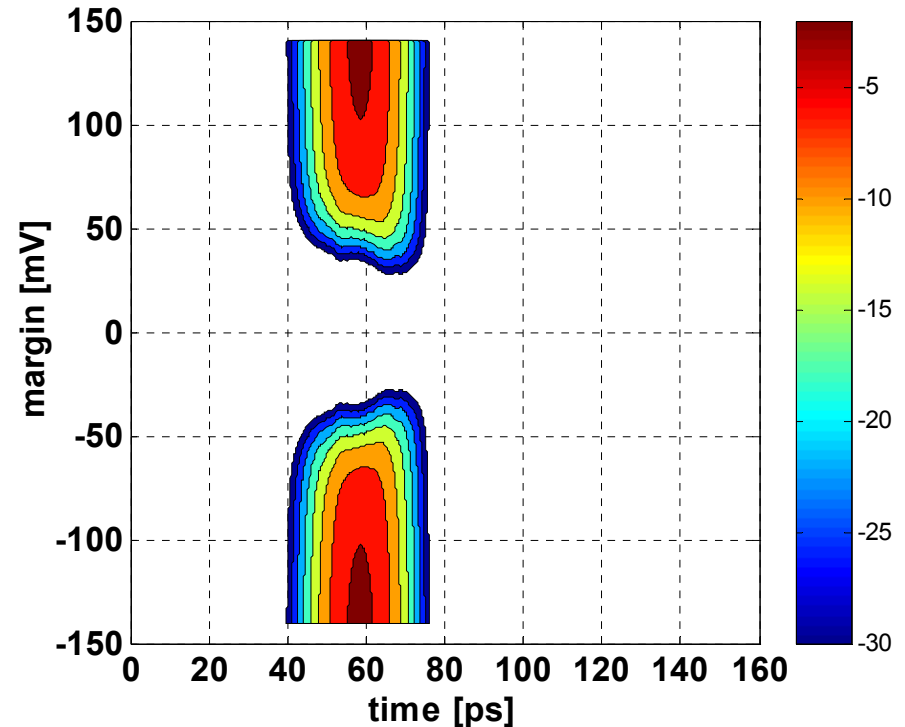
- To compare different designs
 - Compare the voltage margin at given BER
- Need to include all noise sources
 - Accurate ISI distribution
 - Transmit and receive jitter
 - CDR jitter
 - EQ quantization noise
 - Receiver offset

BER Contours

5 tap Tx Eq



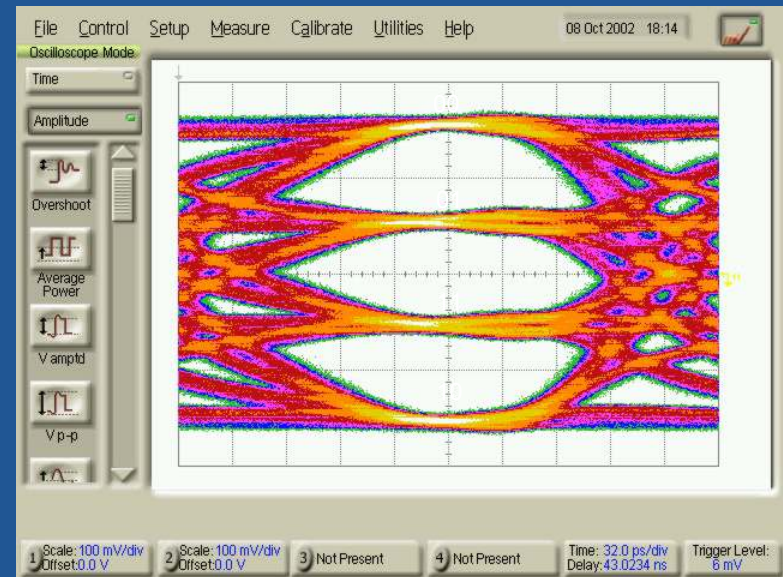
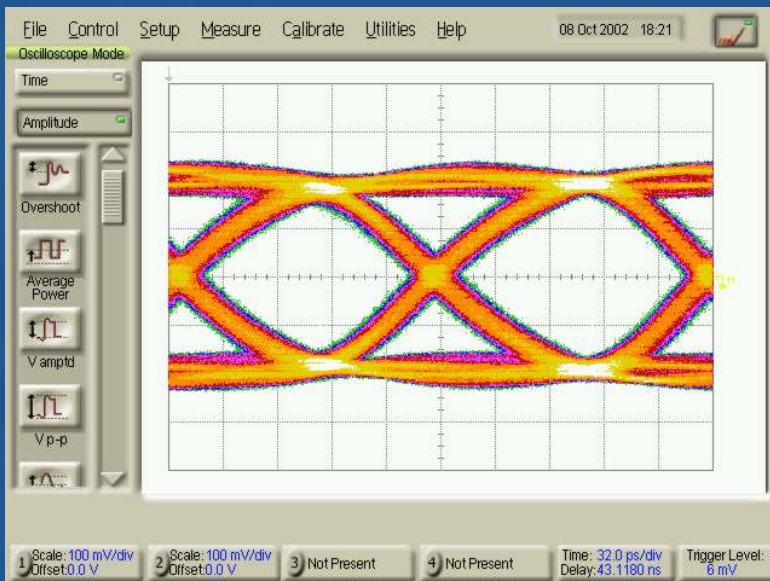
5 tap Tx Eq + 1 tap DFE



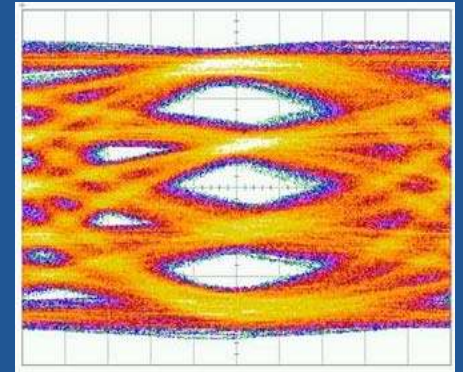
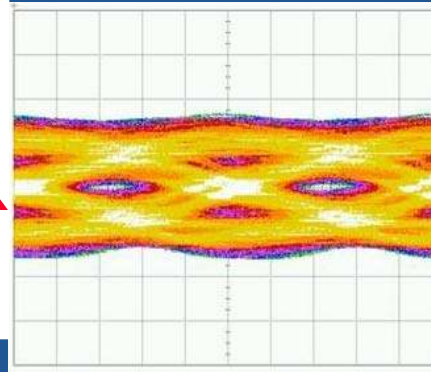
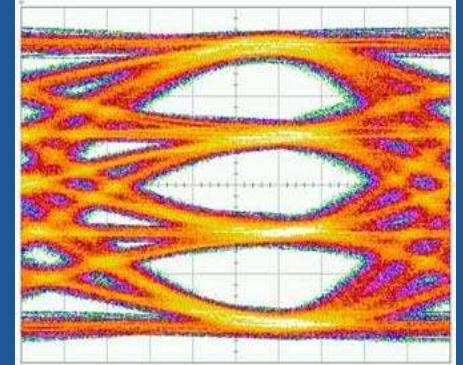
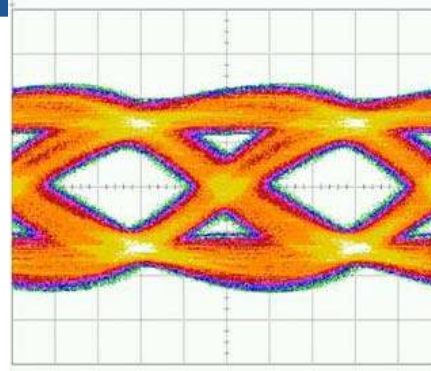
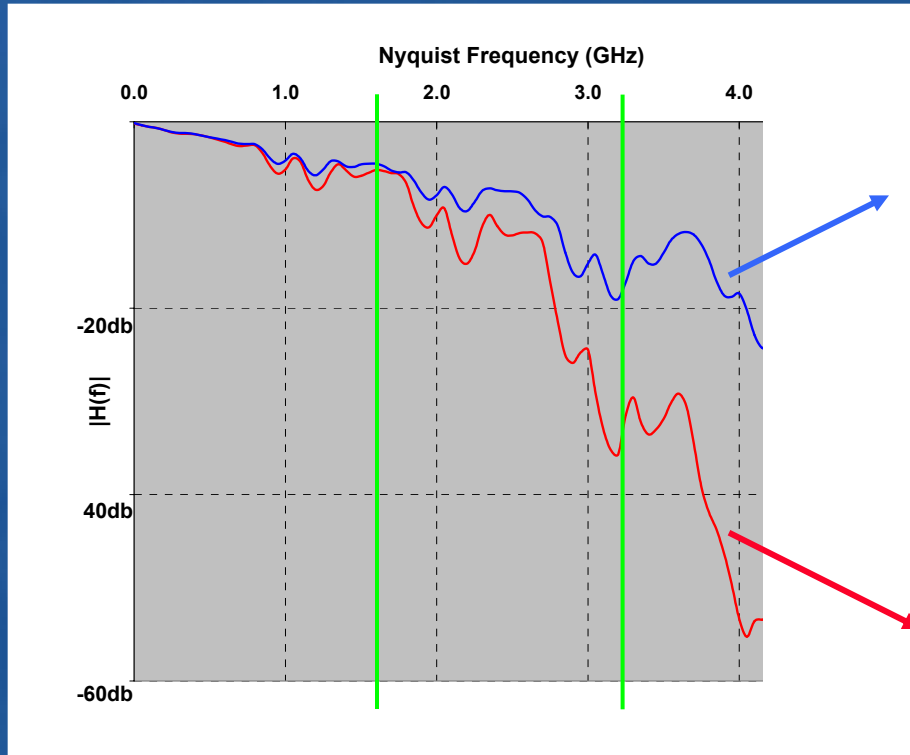
- Voltage margin
 - Min. distance between the receiver threshold and contours with same BER

Pulse Amplitude Modulation

- Binary (NRZ)
 - 1 bit / symbol
 - Symbol rate = bit rate
- 4-PAM
 - 2 bits / symbol
 - Symbol rate = bit rate/2



When Does 4-PAM Make Sense?

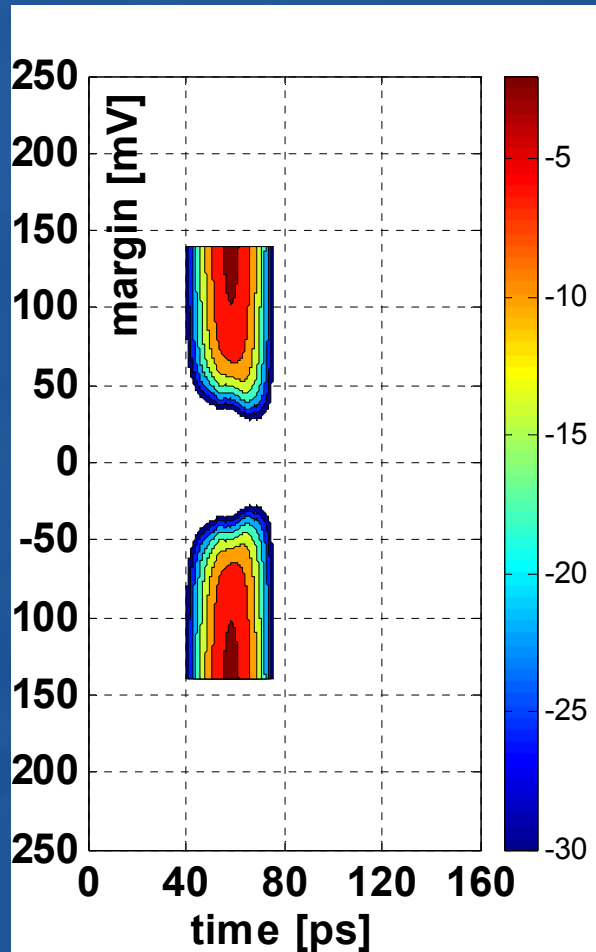


- First order : slope of S21
 - 3 eyes : 1 eye = 10db
 - loss > 10db/octave : 4-PAM should be considered

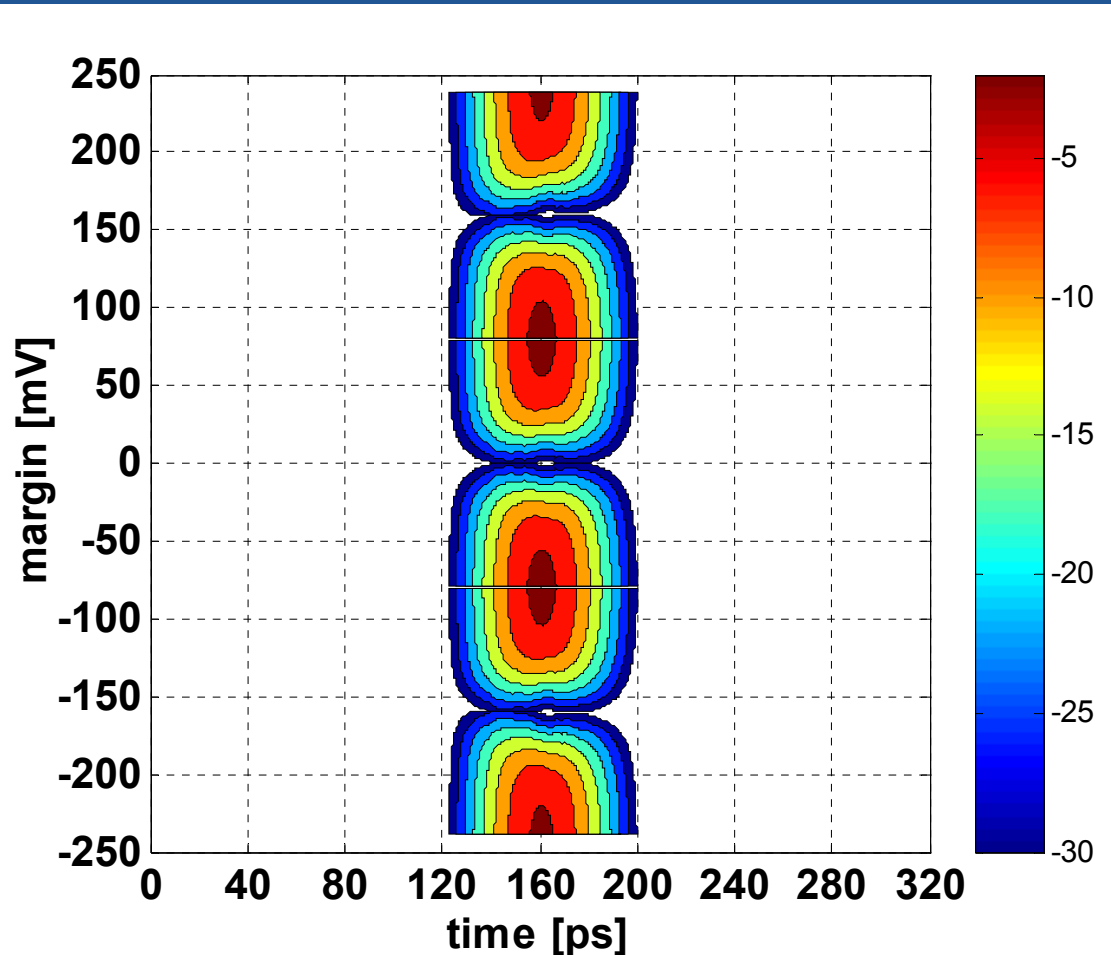
Zerbe et al '03

BER Contours

PAM2 DFE



PAM4 linear equalization



Voltage Margins [mV] at BER=10⁻¹²

Eq/Mod type vs. BP length	3"	10"	20"
2PAM	32	17	19
2PAM w. DFE	79	49	44
4PAM	10	37	31

- Longer backplane channels – more ISI
- PAM2 with DFE effectively combats ISI
- PAM4 makes better use of available bandwidth
 - Less ISI

Conclusions

- Backplane links limited by the channel
- ISI is large
 - Can't completely compensate
 - (At least not with reasonable area/power)
 - Residual ISI also increases CDR jitter
- Generally have low BER requirements
 - Accurate noise statistic important
 - Many of large noise source are bounded
- Power constrained transmitter
 - 4 PAM and simple DFE are attractive solutions