

Study of DDR Asymmetric Rt/Ft in Existing IBIS-AMI Flow

DesignCon IBIS Summit
Santa Clara, USA
February 1st, 2018

Wei-hsing "Michael" Huang, SPISim
Wei-hsing.Huang@spisim.com
Wei-kai Shih, SPISim
Wei-kai.Shih@spisim.com



Agenda:

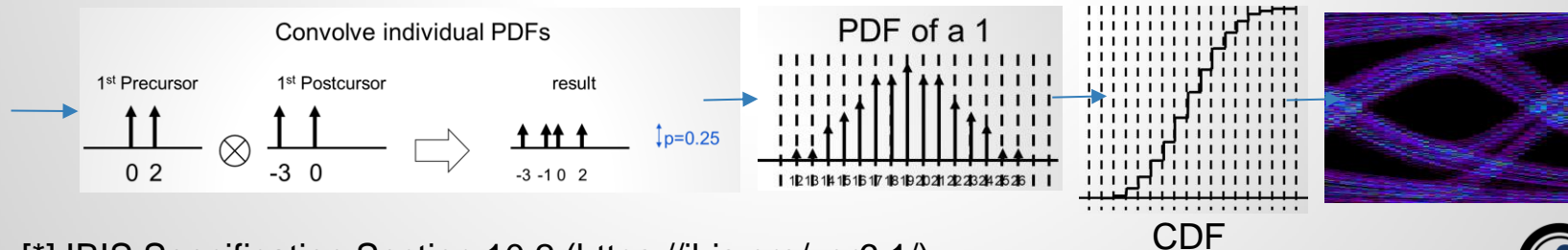
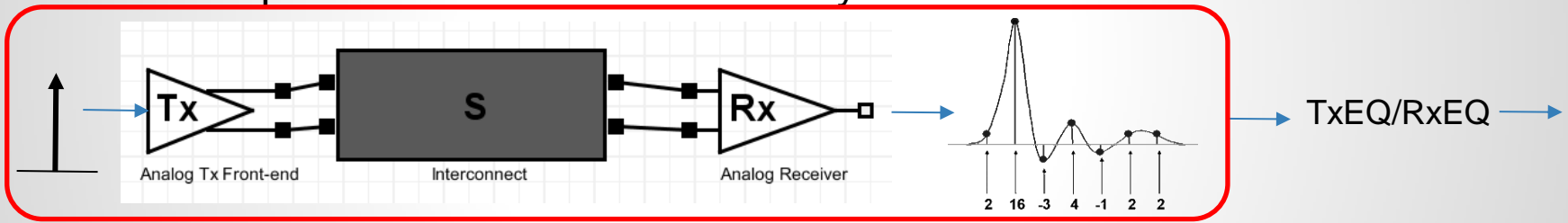
- Motivation
- Background
- Asymmetric Rt/Ft
- AMI_Init
- AMI_GetWave
- Summary
- Q & A

Motivation

- Existing AMI reference flow is for differential:
 - e.g. SERDES, centered around $V = 0.0$
 - Symmetric rise-time (Rt) /fall-time (Ft) mostly
- Can it be used for single-ended?
 - e.g. DQ or DDR, centered around $V = V_{Ref}$
 - Asymmetric Rt/Ft
- Assumption: no major flow change
 - EDA Companies have different opinions
 - No reference flow change is being proposed so far...

Background:

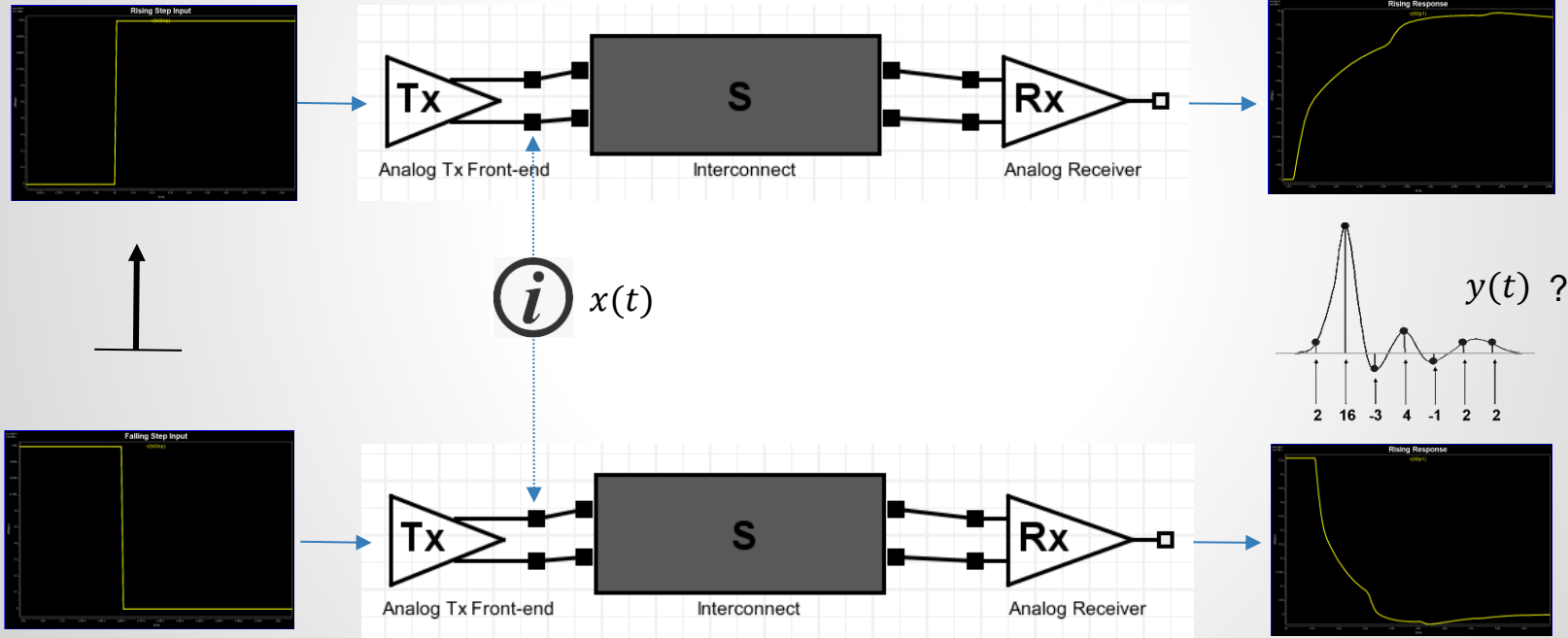
- Statistical AMI flow: [*]
 - “Calibration” is performed before statistical/AMI computation
 - Impulse Response for analog + channel (Linear Time Invariant, LTI)
 - Samples \rightarrow PDF \rightarrow CDF \rightarrow BER/Eye



[*] IBIS Specification Section 10.2 (<https://ibis.org/ver6.1/>)

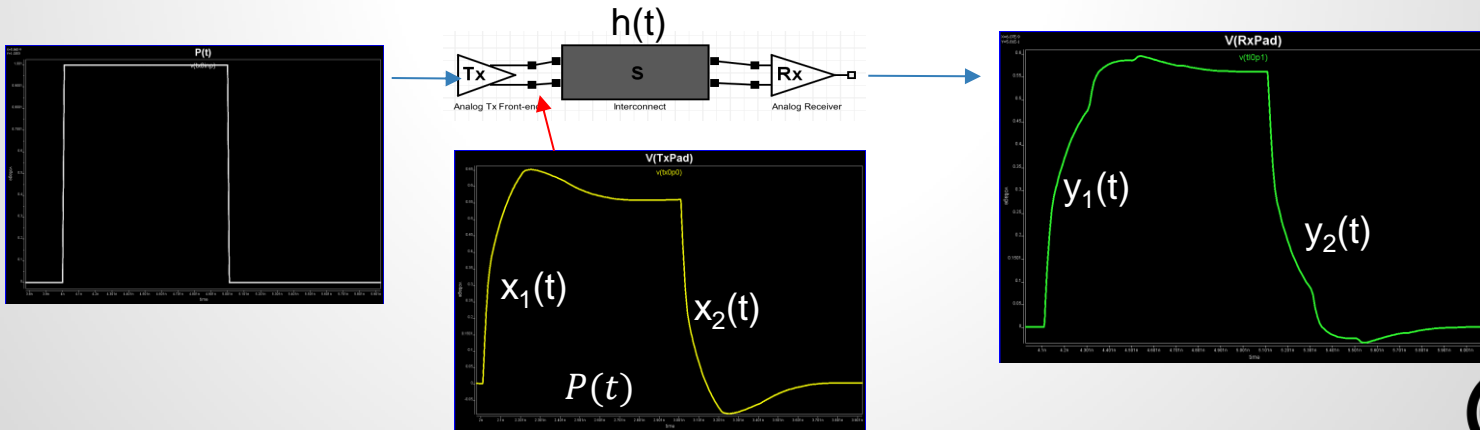
Asymmetric Rt/Ft to Impulse:

- Impulse resp. $\delta(t)$ is usually calculated from $dU(t)/dt$:



Asymmetric Rt/Ft to Impulse:

- Linear transform between Rt/Ft:
 - Rise: $y_1(t) = x_1(t) * h(t)$ Fall: $y_2(t) = x_2(t) * h(t)$
 - Fall: $x_2(t) = x_1(t) * Xform(t) \Rightarrow y_2(t) = y_1(t) * Xform(t)$
 - Simulator knows x_1 & x_2 , thus $Xform(t)$. It can then reconstruct either y_1 or y_2 from y_2 or y_1 with $Xform(t)$ used in AMI_Init. (x_1, x_2 does NOT include EQ)
 - DC info disappeared during differentiation (to get impulse response). **Has gap!**
A new parameter has been proposed to address this gap.



Example:

- Matlab/Octave pseudo-code:

```
% Generate rise and fall ramp of different slew rates
clc;
clear;
time1 = (-1:1:5)';
ustp1 = time1>=0;
xstp = time1.*ustp1;

time = (-1:1:2)';
ustp = time>=0;
ystp = time.*ustp;

m1en = 10;
rstp = ones(m1en, 1);
rstp(1:size(xstp,1), 1) = xstp / 5;

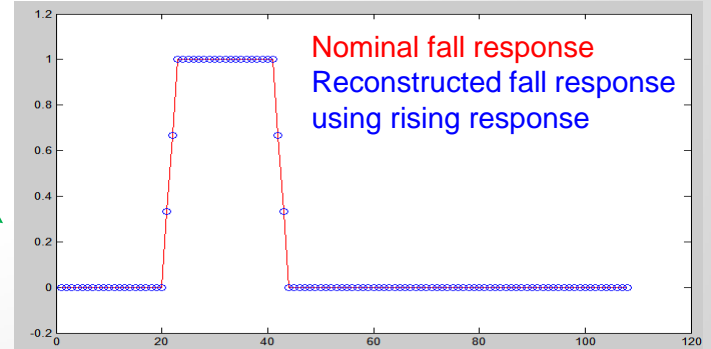
fstp = ones(m1en, 1);
fstp(1:size(ystp,1), 1) = ystp / 3;

% Convert to impulse
rimp = diff(rstp);
fimp = diff(fstp);

% Nominal rise and fall pulse response
pulse=zeros(100,1);
pulse(20:40,1)=1;
rpuls=conv(rimp, pulse);
fpuls=conv(fimp, pulse);

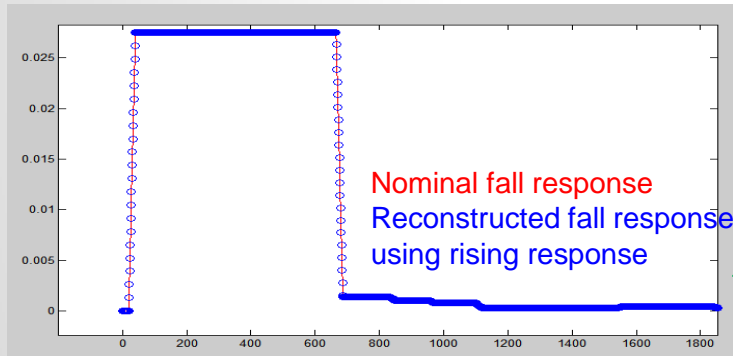
% Reconstruct fall pulse using XForm
plen =size(rpuls, 1);
xpuls=real(ifft(fft(fimp, plen) ./ fft(rimp, plen) .* fft(rpuls)));

% Plot them together
time=[1:plen];
plot(time, fpuls, 'r-', time, xpuls, 'bo');
```

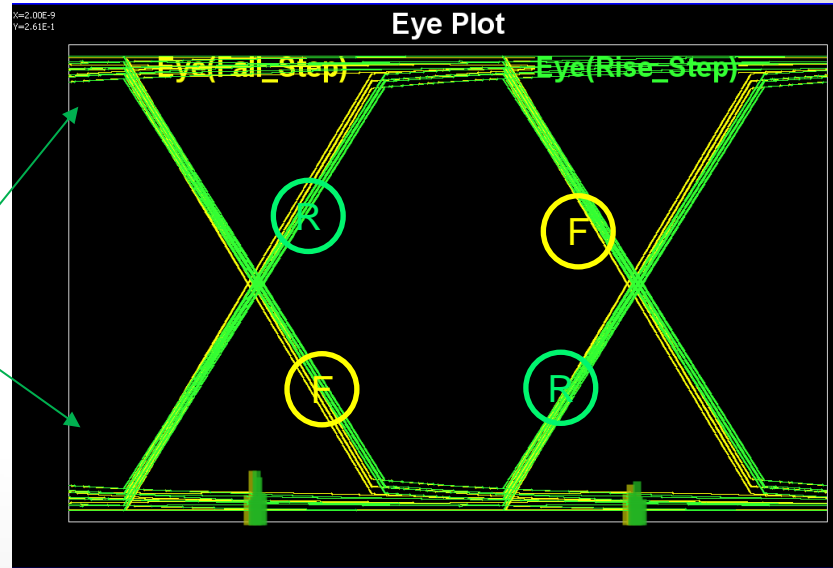


Asymmetric Rt/Ft to Eye:

- Construct different eyes portions using eyes generated by rise response and fall response (different slew rate)
 - Eye will be asymmetric as well.



Real case: (IBIS + Lossy Channel)



ISI Eye Construction with a Tree Structure

3	2	1	Cursor (0)	-1	
0	0	0	1	0	
1					
0	1	1			
1					
0	0			1	1
1					
0	1	1			
1					

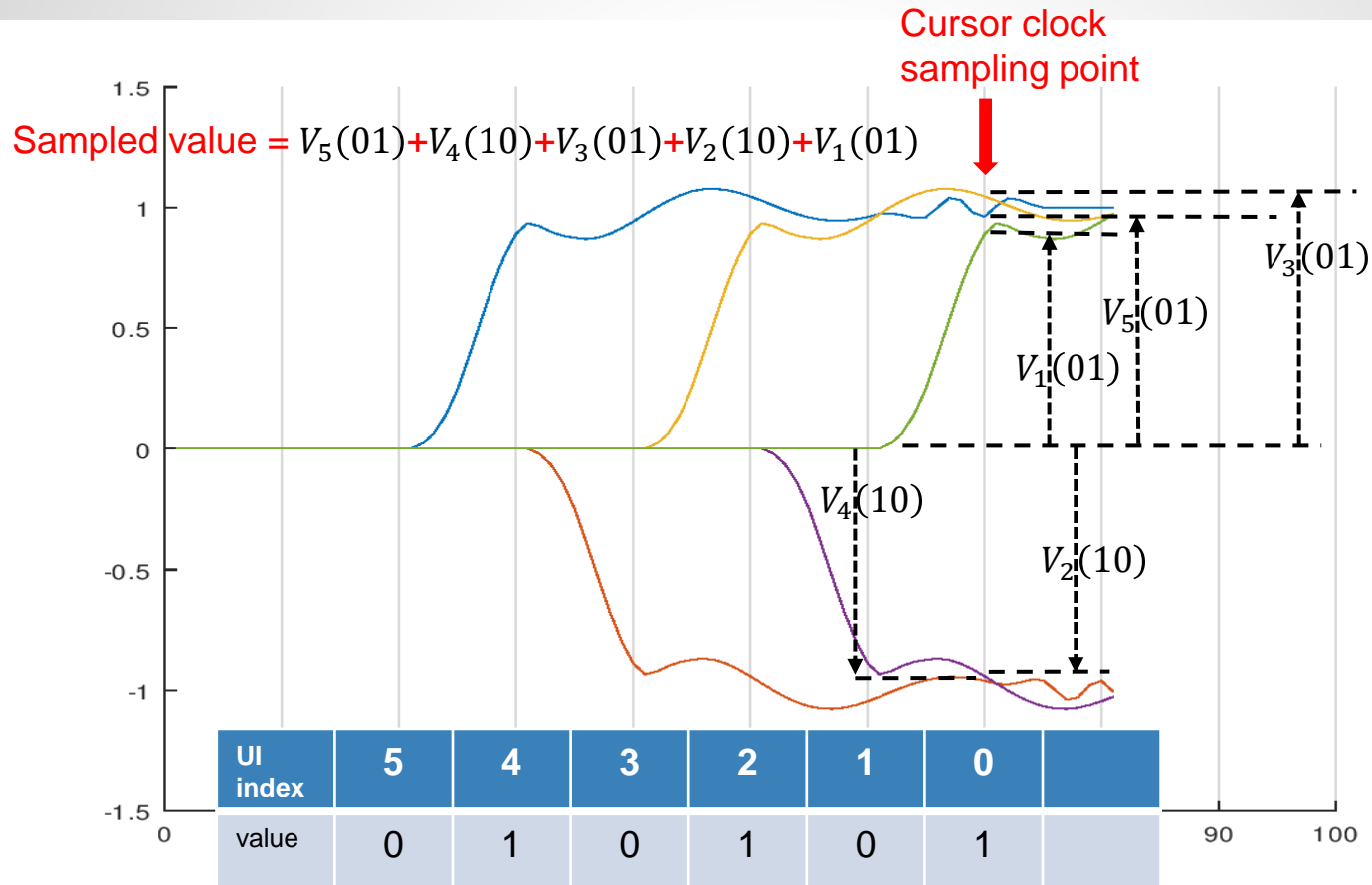
Let $V_n(ab)$ be the contribution of ISI from the n th pre-cursor edge when the n th pre-cursor= a and $(n-1)$ th pre-cursor= b , i.e., the n th pre-cursor edge is an $a \rightarrow b$ transition

When 2nd pre-cursor logic value = 0, cursor logic value = 1, all possible values for the accumulated ISI from 2nd and 1st pre-cursors can be put into a row vector : $[V_2(00) + V_1(01), V_2(01) + V_1(11)]$. There are two elements in the vector due to two possible values of the 1st pre-cursor

Extending to the 3rd pre-cursor: When 3rd pre-cursor = 0, there are 4 possible accumulated ISI values

$[V_3(00) + V_2(00) + V_1(01), V_3(00) + V_2(01) + V_1(11)]$ and $[V_3(01) + V_2(10) + V_1(01), V_3(01) + V_2(11) + V_1(11)]$

Example with Alternating Bit Pattern



Recursive Algorithm for ISI Eye Construction

n	n-1	1 ... n-2	Cursor (0)		
0	0	XXXXXXXXXX	1		
1					
0	1				
1					

$W_n(ab)$: row vector consisting all possible values of the accumulated ISI from the n th pre-cursor to cursor when logic value of the n th pre-cursor is a and logic value at cursor is b

$$W_1(01) = [V_1(01)]$$

$$W_1(11) = [V_1(11)]$$

$$W_2(01) = [V_2(00) + V_1(01), V_2(01) + V_1(11)]$$

$$W_2(11) = [V_2(10) + V_1(01), V_2(11) + V_1(11)]$$

... ..

$$W_n(01) = [V_n(00) + W_{n-1}(01), V_n(01) + W_{n-1}(11)]$$

$$W_n(11) = [V_n(10) + W_{n-1}(01), V_n(01) + W_{n-1}(11)]$$



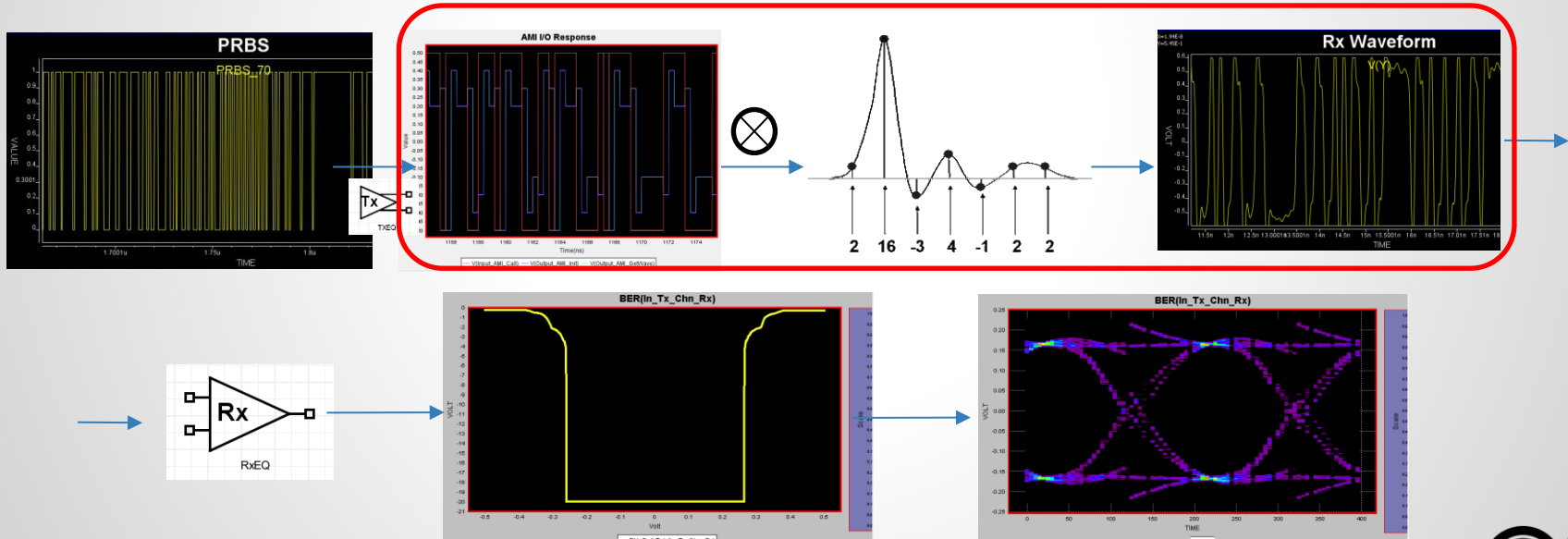
PDF Computation for ISI Eye

Waveform value	PDF of the waveform value	Notes
$V_n(ab)$	$P_{V_n(ab)}(V) = \delta(V - V_n(ab))$	
$W_1(01)$	$P_{W_1(01)} = P_{V_1(01)} \quad P_{W_1(11)} = P_{V_1(11)}$	
$W_n(01)$	$P_{W_n(01)} = \frac{1}{2} P_{W_{n-1}(01)} \otimes P_{V_n(00)}(V) + \frac{1}{2} P_{W_{n-1}(11)} \otimes P_{V_n(01)}$	
$W_n(11)$	$P_{W_n(11)} = \frac{1}{2} P_{W_{n-1}(01)} \otimes P_{V_n(10)}(V) + \frac{1}{2} P_{W_{n-1}(11)} \otimes P_{V_n(11)}$	

This is a Dirac delta when there is no jitter (ISI takes discrete value without jitter)
 With jitter the Dirac delta will spread out into a continuous distribution. But the recursive relation remains same.

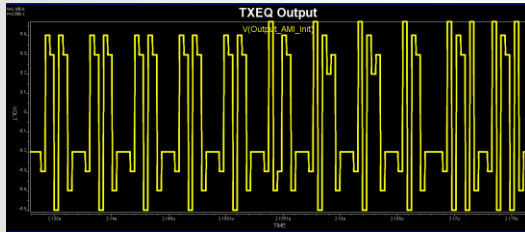
Background:

- Time-domain AMI flow:
 - Analog + channel's responses to one block of bit-sequence
 - Convolve with Tx's AMI_GetWave output then send to Rx's AMI

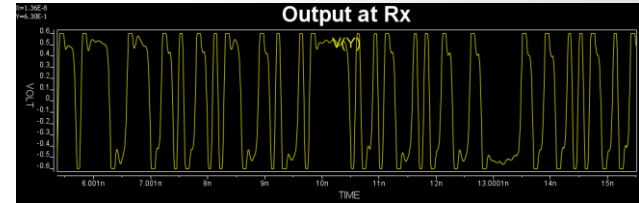


Asymmetric Rt/Ft to GetWave:

- Result will be OK if:
 - Bit-sequence waveform at Rx is simulated result from bit-sequence of output at TxEQ
 - This may not be the case mostly as it is not “standard” and will take longer to run.

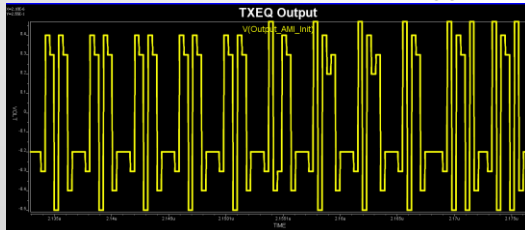


Simulated →

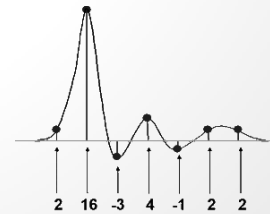


- Result will have errors if:

- Final waveform at Rx is from one bit simulated impulse response convolved with bit-sequence from TxEQ
- This is what reference flow has suggested.

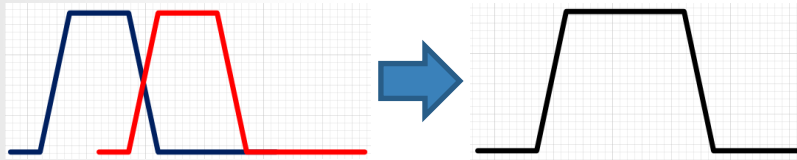


Convolve with →

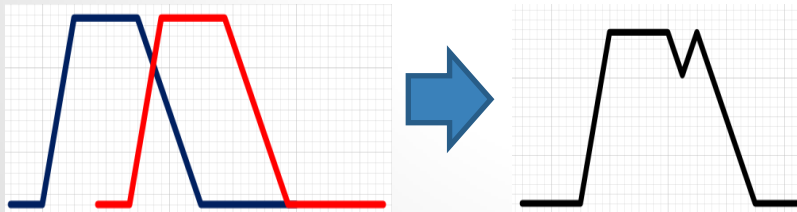


Asymmetric Rt/Ft to GetWave:

- Problem: Which impulse to use? From Rise or Fall?
- Problem: Using a full $P(t)$ will still result in errors:
 - Bit 011 using convolution with symmetric Rt/Ft



- Glitch will happen for asymmetric Rt/Ft



- Need to use Rise/Fall step separately then superimpose.

Asymmetric Rt/Ft to GetWave:

- Matlab/Octave pseudo-code:

```
% Generate one-bit pulse of different Rt/Ft
clc;
clear;
time = (0:1:2)';
ustp = time>=0;
xstp = time.*ustp;

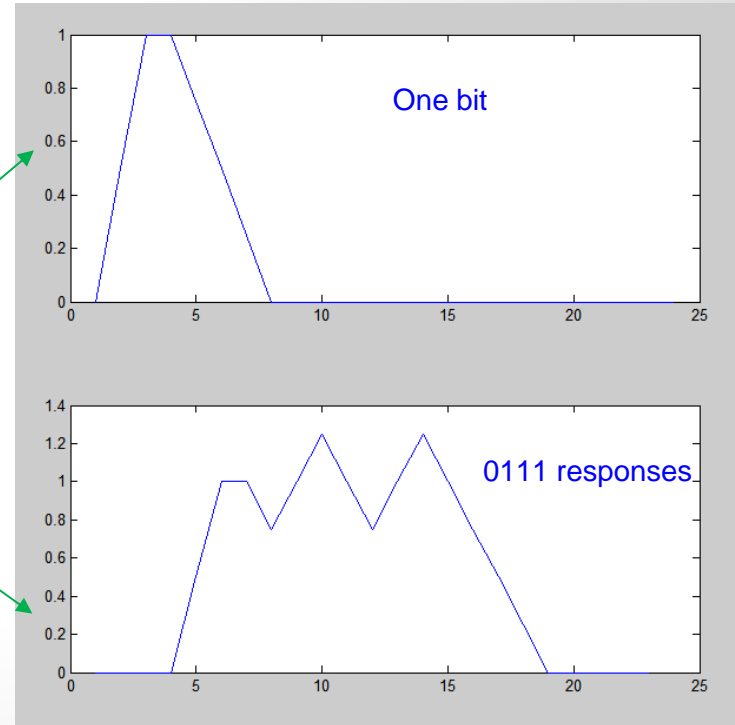
time1 = (0:1:4)';
ustp1 = time1>=0;
ystp = time1.*ustp1;

xlen = size(xstp, 1);
ylen = size(ystp, 1);
mlen = xlen + ylen;
bit1 = ones(mlen, 1);
bit1(1:xlen, 1) = xstp / 2;
bit1(xlen + 1:xlen + ylen, 1) = 1 - ystp / 4;

% Bit sequence 0111
ui = size(bit1, 1) / 2;
blen = 4 * ui;
bseq = zeros(blen, 1);
bseq(1 * ui) = 1;
bseq(2 * ui) = 1;
bseq(3 * ui) = 1;

% Form responses using convolution
resp = conv(bit1, bseq);

% Plot them together
subplot(2,1,1);
plot(padarray(bit1, blen, 'post'));
subplot(2,1,2);
plot(resp);
```



Summary:

- Existing IBIS-AMI flow:
 - Can be used for driver with asymmetric Rt/Ft.
 - Asymmetric effects can be handled within “smarter” EDA tools.
 - Assuming AMI model does not behave differently to rise/fall responses.
- Statistical flow:
 - Linear transform between rise/fall can be applied to model’s response.
 - Use rise and fall response to construct eye.
 - Tree/sequence based superposition will eliminate these glitches.
- Time-domain flow:
 - Convolution using one bit pulse will have errors.
 - Using step response based superposition may avoid such errors.





EDA Expertise in Signal, Power Integrity & Simulation

