KEYSIGHT

J4u03 improvements for 212Gbps (179.9.4.6.2 J4u03 proposed revisions)

IEEE P802.3dj Plenary (10-14 Nov 2025) Bangkok

Associated comments (P802.3dj D2.2): 276, 358, 201, 207, 224

Author/Presenter: John Calvin (Keysight Technologies)
Principal contributor: David Gines (Keysight Technologies)

Based on draft release of IEEE P802.3dj™/D2.2

Abstract: J4u03 methodology as presently outlined in clause 179.9.4.6.2 (D2.2) exhibits sensitivity to channel loss induced jitter magnification. This proposal is a continuation of the January 9'th 2025 presentation: https://www.ieee802.org/3/dj/public/adhoc/optics/0125_OPTX/gines_3dj_optx_01a_250109.pdf which outlines a method of improved J4u as well as for JRMS. The 1.6T Taskforce embraced the JRMS change, but J4U required additional technical feasibility. This contribution address several comments against the stability of J4u03 and offers a solid foundation for an improved methodology that is leverageable to the future.

Supporters/Collaborators (Version 1.0)

TBD

Useful References:

IEEE P802.3dj: Physical layer jitter proposal to advance/close comments against present JRMS and J4u03 methodologies
Contribution: https://www.ieee802.org/3/dj/public/25_01/calvin_3dj_01b_2501.pdf

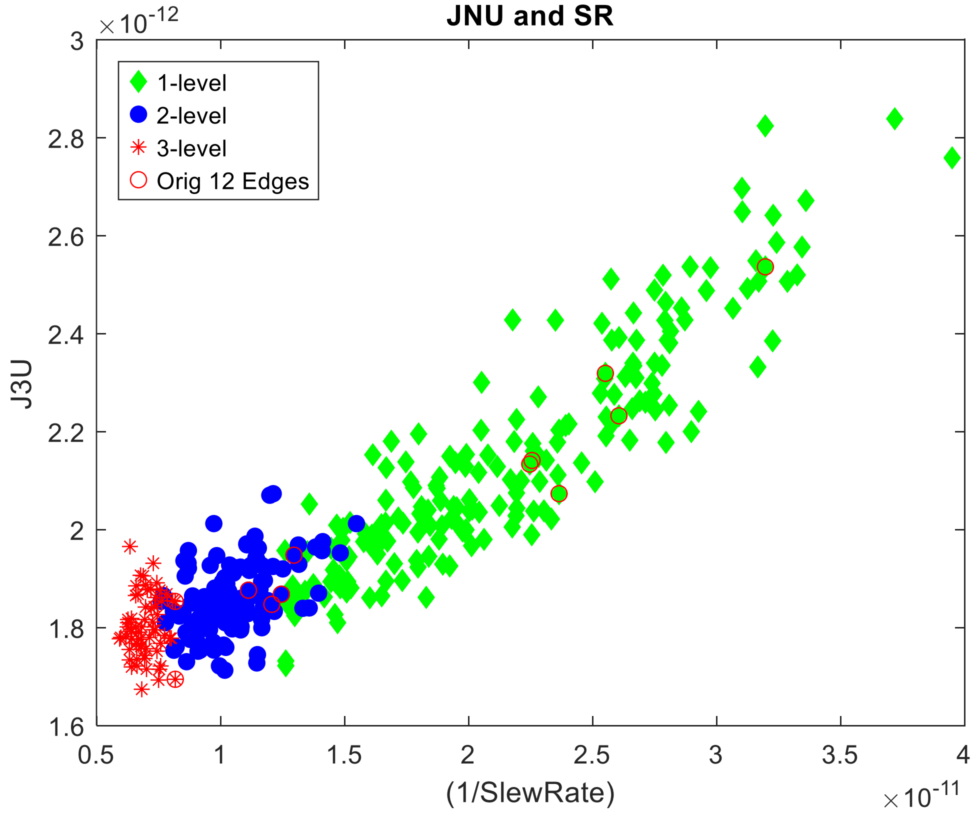
IEEE P802.3dj 01/09/2025: Gines_Phase Only Jitter
Contribution: https://www.ieee802.org/3/dj/public/adhoc/optics/0125_OPTX/gines_3dj_optx_01a_250109.pdf

IEEE P802.3dj 07/15/2024:Calvin_1.6Tbps JNu operations / high loss channels
Contribution: https://www.ieee802.org/3/dj/public/24_07/calvin_3dj_01b_2407.pdf

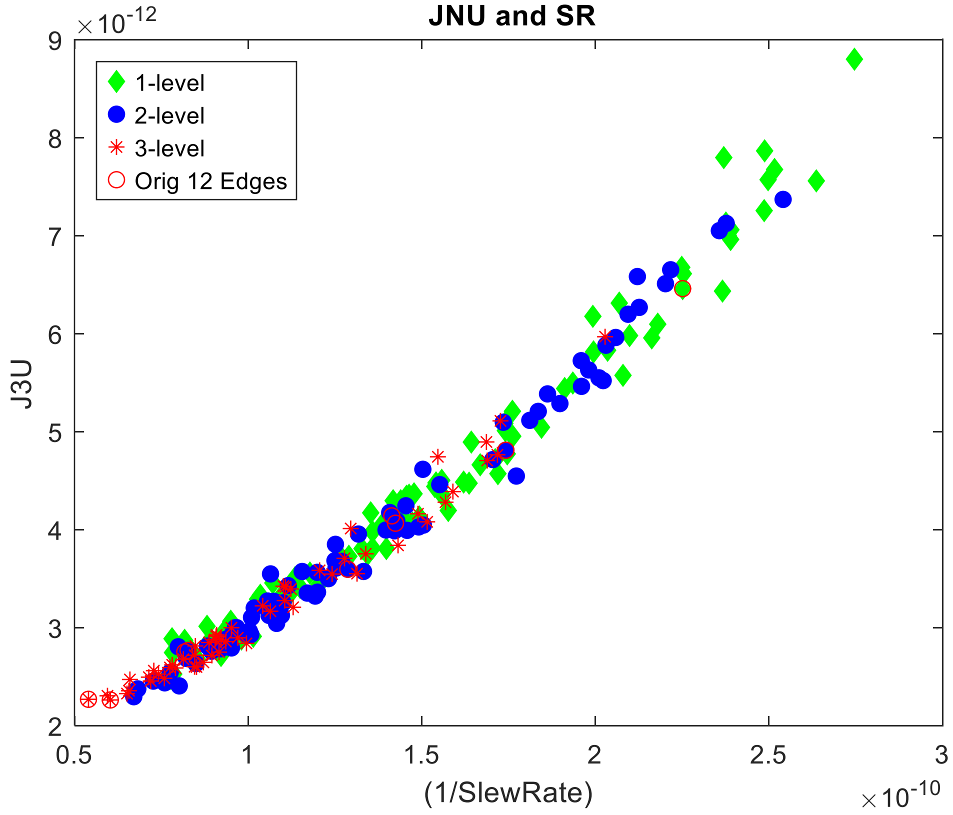
JNU

JNU is affected by slew rate

JNU is affected by slew rate and has behavior similar to Jrms. JNU goes up and becomes more variable with increased channel loss. (note the different scales in the two plots)



No Channel



31 dB Channel

Jitter Model

JNU using dual-Dirac model

The dual-dirac model seems to be the most promising approach for finding the horizontal (phase-only) measurement of JNU. The dual-dirac model is:

$$JNU = DJ_{dd} + 2 * Q * RJ_{rms}$$

The idea is to estimate horizontal components DJH_{dd} and RJH_{rms}. We can then approximate,

$$JNUH = DJH_{dd} + 2 * Q * RJH_{rms}$$

Define DJdd as the simple sum of horizontal and vertical, leading to a linear model:

$$DJ_{dd} = DJH_{dd} + DJV_{dd}$$

$$DJ_{dd} = DJH_{dd} + \left(\frac{1}{SR}\right) Ndd$$

RJrms is an RMS value, so it has the same model as Jrms:

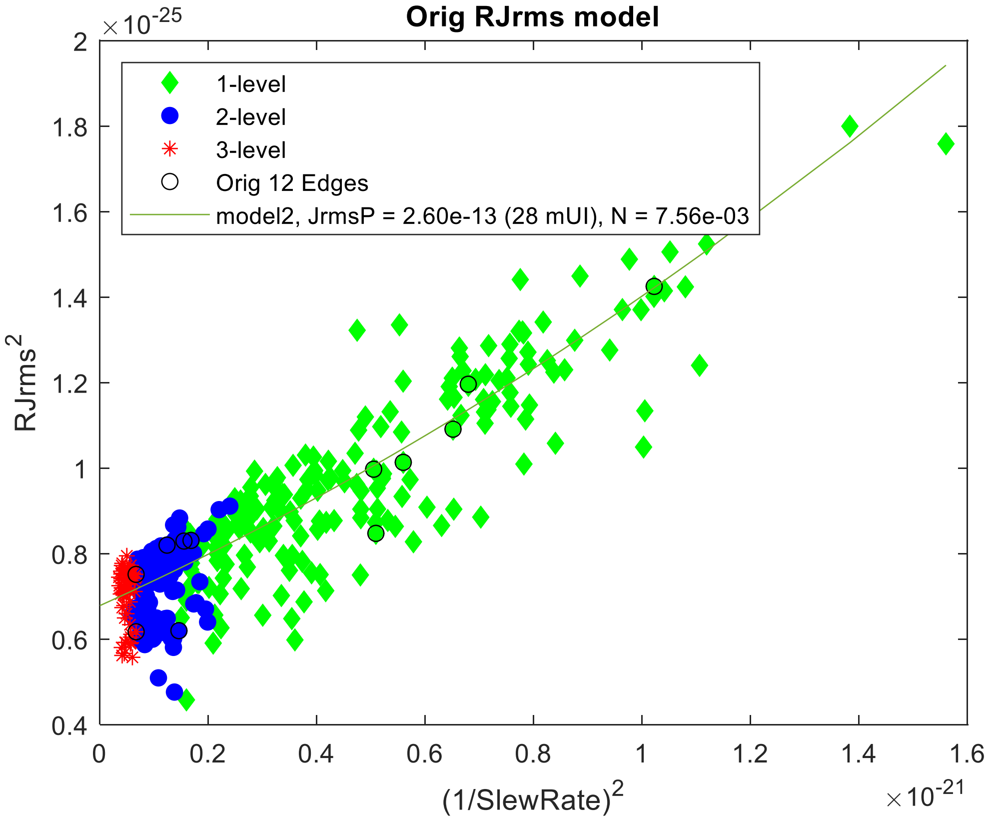
$$RJ_{rms}^2 = RJH_{rms}^2 + RJV_{rms}^2$$

$$RJ_{rms}^2 = RJH_{rms}^2 + \left(\frac{1}{SR}\right)^2 N_{rms}^2$$

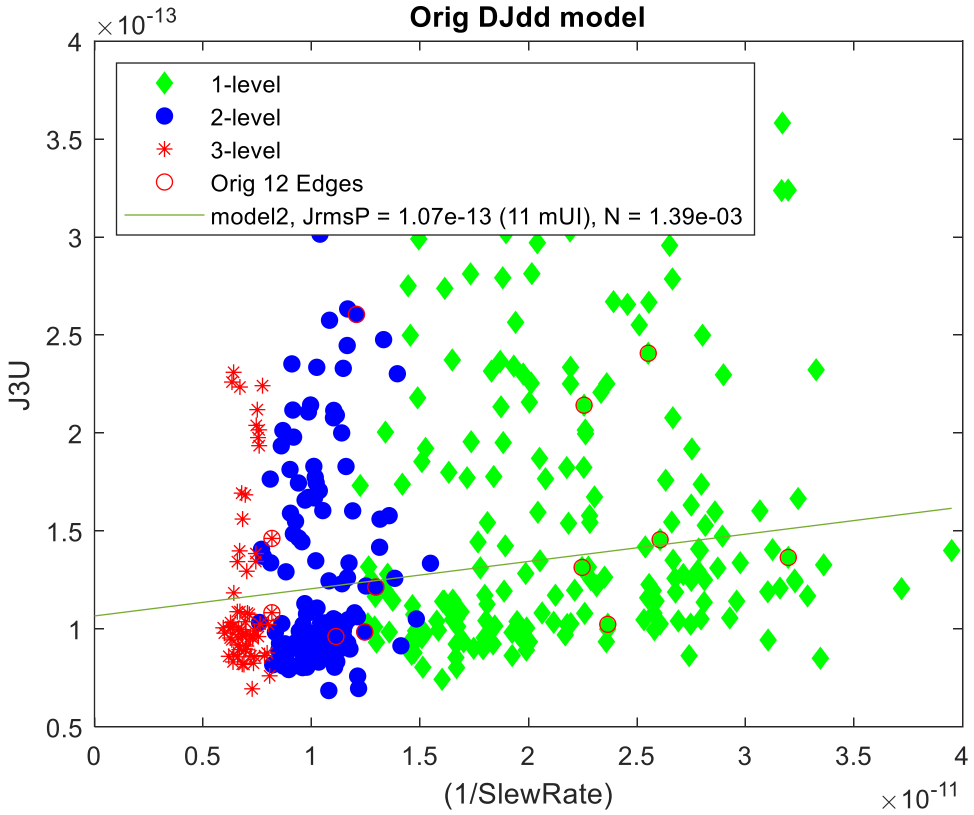
Jitter Model

Dual Dirac approach

Here is what the models for RJrms and DJdd might look like (after a 31dB Channel



Model for RJrms

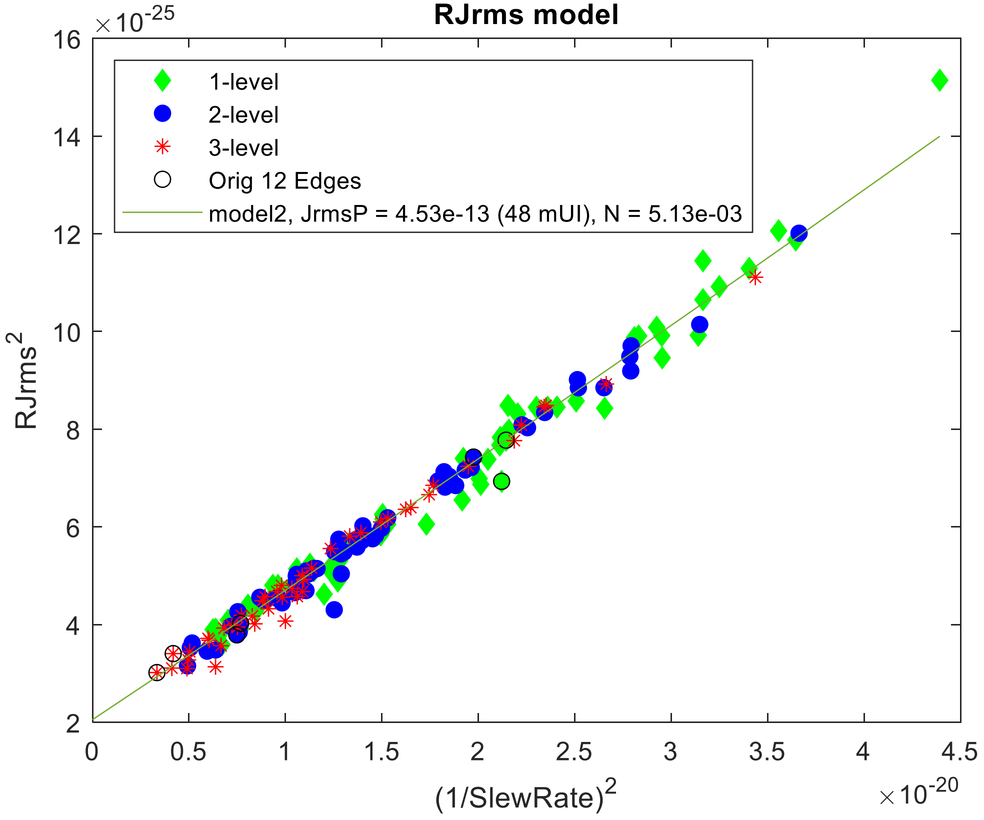


Model for DJdd

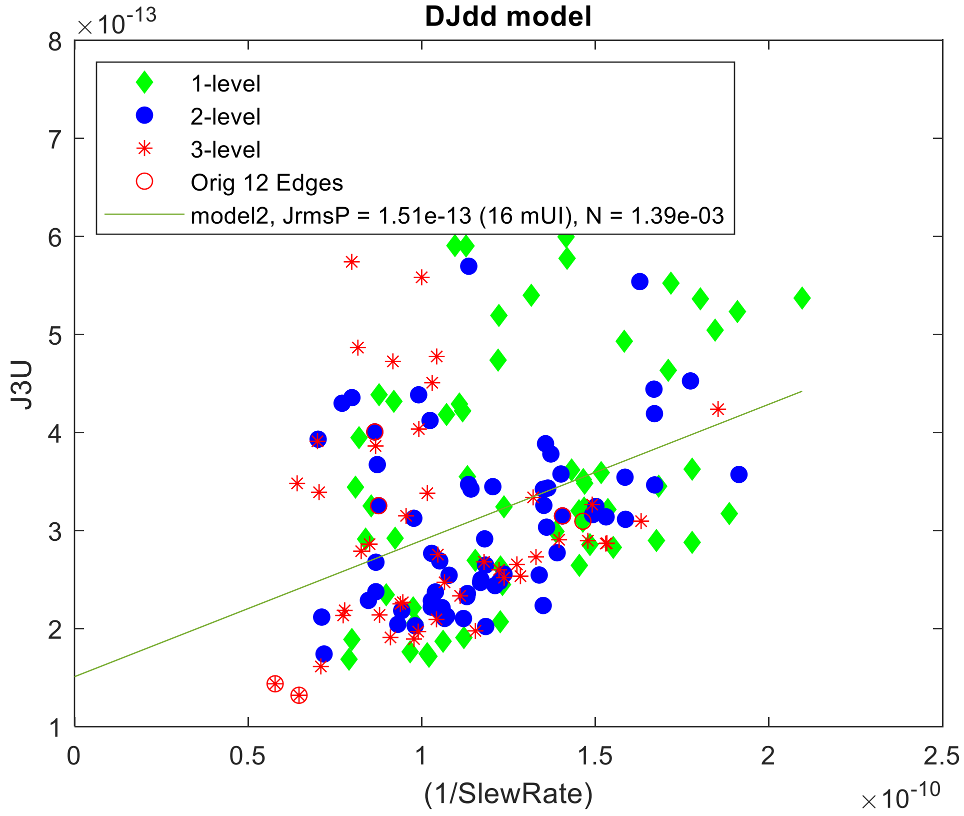
Jitter Model

Dual Dirac approach

BERT, 106.25 GBd, PRBS9Q, 50 mUI RJrms, 50 mUI PJpp, Channel 31dB ISI



Model for RJrms



Model for DJdd

Jitter Model

Some challenges

- Tailfitting must be done on each edge. Tailfitting is an additional source of variability and potential error when compared to Jrms, which is a simple standard deviation.
- DJdd is not a physical parameter, it is a model parameter, so the POJ model for DJdd is only approximate.
- RJrms works the same way as Jrms, so it is on solid theoretical ground. However, in the dual-dirac model it is multiplied by 2*Q, so any errors are also amplified by this same amount.
- These all suggest that refinements in all pre-processing (clock recovery, TIE measurements, slope estimations, etc.) will prove useful for improving the approach.

- Consequently, areas of research include:
  - Better ways of estimating DJHdd
  - Improved pre-processing
  - Approaches to make tailfitting more stable

Jitter Model

Some results

Here are some points of comparison. This table is for simulations using a mild (12dB) channel with different impairments. The true and estimated values of JNU are circled. Generally, these results were good, which is encouraging.

Input Params			True Horizontal Params				
RJrmsIn	PJ Amplitude	NrmsIn	RJrms	PJpp	Jrms	TJpp	J3U
eps	0	0	0	0	0	5	3
50	0	0	50	0	50	498	330
0	50	0	0	100	35	107	101
50	50	0	50	100	61	567	386
50	0	50	50	0	50	498	330
0	50	50	0	100	35	112	100
50	50	50	50	100	61	567	386

Orig DJdd		
RJrmsH	DJddH	J3UH
1	0	4
48	21	335
0	103	103
55	42	404
49	20	339
-4	94	65
54	46	400

Jitter Model

Some results

These results are for live data, PRBS9Q. These tables also include the JNU03 approach for comparison. The first table has no channel, and the second has a 31 dB channel. With no channel, the JNU03 results are similar to the JNUH results, and both are reasonably accurate. For the 31dB channel, the **<u>JNU03 measurements grow large, while JNUH results do not.</u>**

*mUI

No Channel

Input				JNU 03			
RJrms True	PJpp True	PJrms True	J3UH True		RJrmsH	DJddH	J3UH
0	0	0	0	51	2	2	15
30	0	0	198	188	28	11	193
0	50	35	50	87	4	35	62
0	0	0	0				
30	50	35	223	219	29	18	209

31 dB Channel

Input				JNU 03	JHNU		
RJrms True	PJpp True	PJrms True	J3UH True		RJrmsH	DJddH	J3UH
0	0	0	0	150	0	0	25
30	0	0	198	241	25	7	171
0	50	35	50	184	11	6	78
0	0	0	0				
30	50	35	223	368	27	0	174

Lab Instrument based test results

JH3U

Latest JHNU results: 106.25 GBd, PRBS9Q, BERT

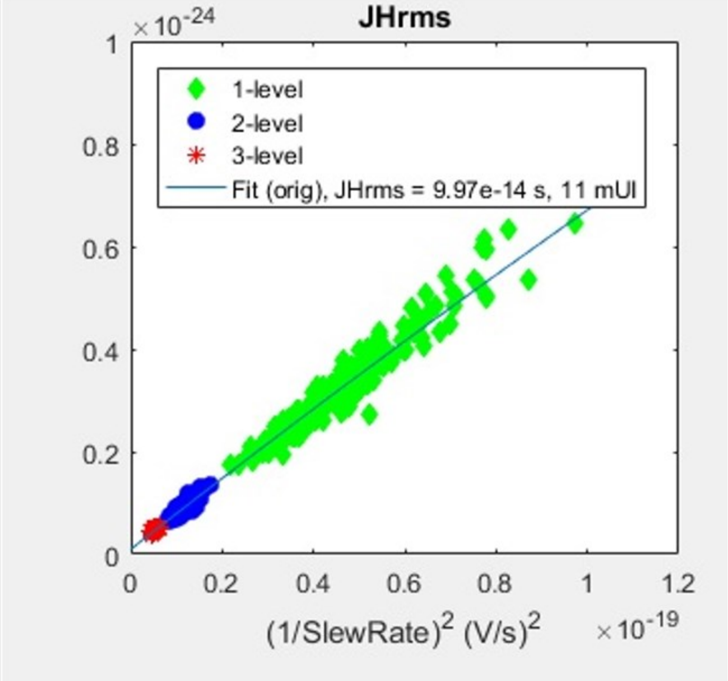
Channel	RJrms True	PJpp True	JH3U True	JH3U	JNU 03
0 dB	0	0	0	22	51
0 dB	30	0	198	192	188
0 dB	0	50	50	85	87
0 dB	30	50	223	218	219
12 dB	0	0	0	25	56
12 dB	30	0	197	193	202
12 dB	0	50	50	57	90
12 dB	30	50	222	219	232
31 dB	0	0	0	49	150
31 dB	30	0	198	190	241
31 dB	0	50	50	104	184
31 dB	30	50	223	230	368
31 dB	0	0	0	42	123
31 dB	50	0	330	317	331
31 dB	0	50	50	97	151
31 dB	50	50	347	333	394

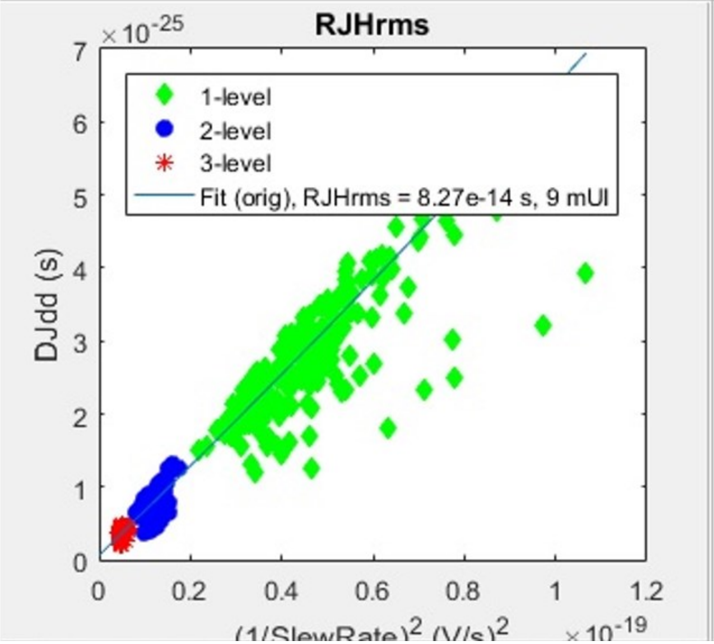
Field deployed JH4U beta-tool results.

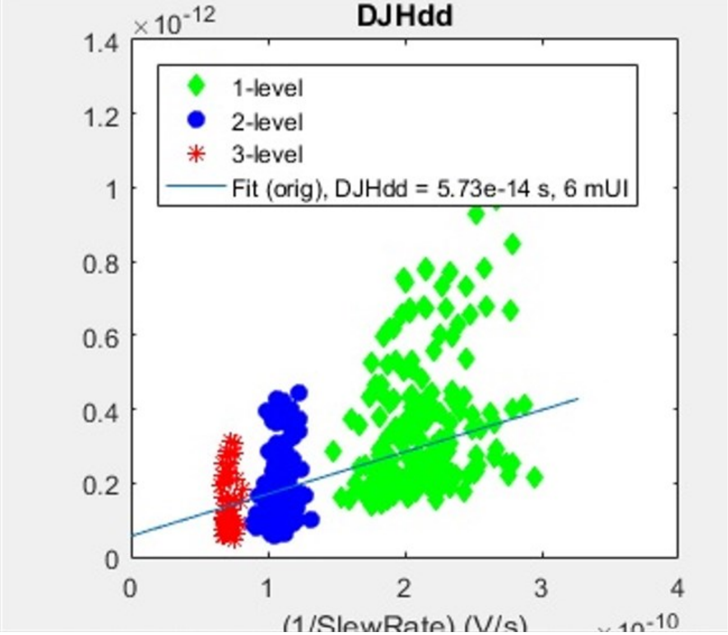
Anyone interested in access should contact: John.calvin@keysight.com

Run

	seconds	mUI
JHrms	9.9698e-14	11
RJHrms	8.2740e-14	9
DJHdd	5.7341e-14	6
JH3U	6.0186e-13	64
JH4U	7.0116e-13	74



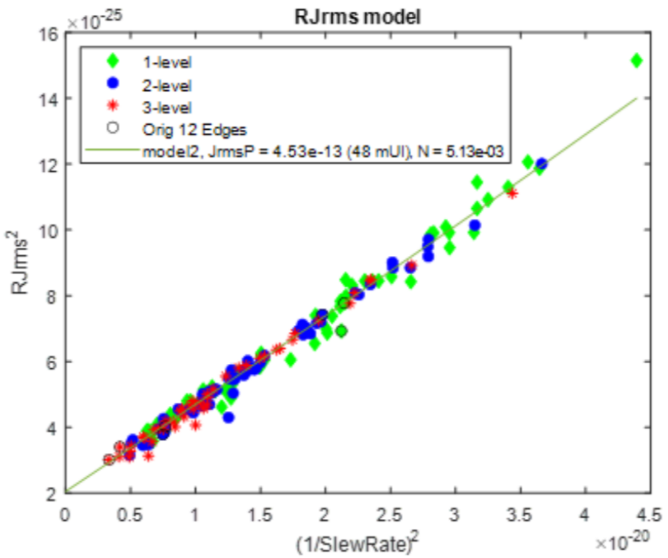


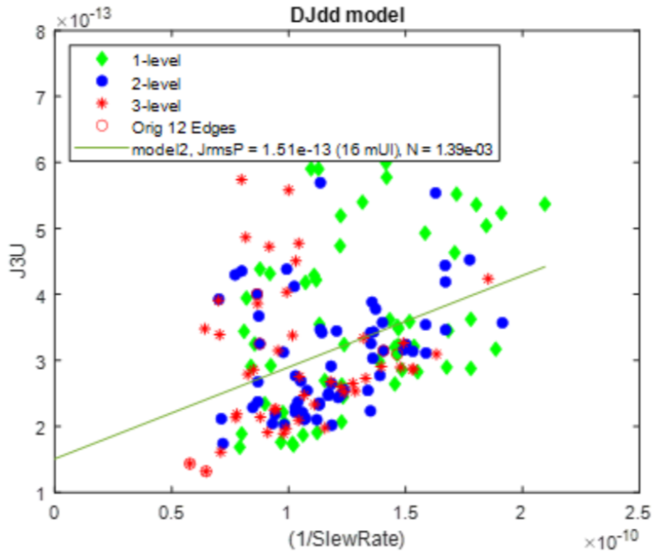


Jitter Model

Conclusions

- This contribution outlines a dual Dirac method of Total Jitter determination (J4u) for a prescribed arbitrary probability and extends the phase-only jitter concept used currently in RJrms to DJH_{DD}.





- This contribution proposes a new model for JNUH (Phase only jitter version of JNU)
- Early test results show favorable potential of the model in preserving jitter integrity after 30+dB channel losses.
- Field evaluations show real-world 1.6T hosts change failing J4U03 results to passing J4UH results as evaluated at TP1a.
- Existing dual-Dirac A_{DD} methodology found in (178.9.3.4.2 Noise calibration) can be referenced in evaluating DJH_{DD}

Proposed revision to Clause 179.9.4.6.2 J4u03

Calculation of the jitter parameter J4U uses the timing of multiple transitions in the pattern to estimate the jitter caused by clock phase noise while minimizing the effect of additive noise, using the method described in this subclause.

Select a set A of transitions from the test pattern used in the test. The size of this set should be large enough to enable calculation of J4U (as defined below) with sufficient accuracy. The set A should include multiple transition types between different PAM4 levels.

For each specific transition :

— Obtain a set Ai = {ti(1), ti(2), …} of transition times modulo the period of the pattern. The size of this set should be large enough to enable estimating JNU with sufficient accuracy.

— Estimate the random jitter RJrms_i and deterministic jitter DJdd_i from the measured dataset Ai using jitter decomposition, such as tailfit.

— Measure the slope of the signal at time μi, denoted si.

From the collections of RJrms_i and si, calculate the best-fit coefficients for the polynomial in 1 / si^2 defined by

RJrms_i^2 = a + b/si^2 + c/si^4

The free coefficient a is the extrapolation of the measured data to an infinite slope. RJHrms is defined as the square root of a.

From the collections of DJdd_i and si, calculate the best-fit coefficients for the polynomial in 1 / si defined by

DJdd_i = a + b/si

The free coefficient a is the extrapolation of the measured data to an infinite slope. DJHdd is defined as a.

Using RJHrms and DJHdd estimate the value of J4U as

J4U = DJHdd + 2*Q*RJHrms,   Where Q = 3.891.