60802 Time Synchronisation – Monte Carlo Analysis:
100-hop Model, "Linear" Clock Drift[1], NRR Accumulation[2]
Overview & Details, Including Equations – v2

David McCall (Intel)

1 – The model includes various options for modelling Clock Drift distribution, but assumes Clock Drift can be considered linear over the short periods of interest.

2 – The model implements calculating Rate Ratio via an accumulation of Neighbor Rate Ratio (NRR) vs. calculating it directly via Sync messages.

References – 1

1. G. Garner, “Initial Simulation Results for Time Error Accumulation in an IEC/IEEE 60802 Network”, IEC/IEEE 60802 contribution, 16 March 2020
2. G. Garner, “Further Simulation Results for Time Error Performance for Transport over an IEC/IEEE 60802 Network”, IEC/IEEE 60802 contribution, 13 July 2020
3. G. Garner, “New Simulation Results for Time Error Performance for Transport over an IEC/IEEE 60802 Network Based on Updated Assumptions”, IEC/IEEE 60802 contribution, 5 October 2020
4. G. Garner, “Further Simulation Results for Dynamic Time Error Performance for Transport over an IEC/IEEE 60802 Network Based on Updated Assumptions”, IEC/IEEE 60802 contribution, 14 December 2020
5. G. Garner, “New Simulation Results for dTE for an IEC/IEEE 60802, Based on New Frequency Stability Model”, IEC/IEEE 60802 contribution, 16 June 2021
6. G. Garner, “New Simulation Results for dTE for an IEC/IEEE 60802 Network, with Variable Inter-message Intervals”, IEC/IEEE 60802 contribution, Rev 2, 01 July 2021

References – 2

7. D. McCall, K. Stanton, G Schlechter, G Woods, T Weingartner, “The 60802 challenge of meeting time accuracy goals across long daisy-chains using 802.1AS™-2020, An analysis and a proposed path forward”, IEC/IEEE 60802 contribution, September 2021
8. D. McCall, K. Stanton, “60802 Dynamic Time Sync Error – Error Model & Monte Carlo Method Analysis”, IEC/IEEE 60802 contribution, November 2021
9. D. McCall, K Stanton, “60802 Dynamic Time Sync Error – NRR Medians, Algorithms & Analysis Validation”, IEC/IEEE 60802 contribution, January 2022
10. D. McCall, K. Stanton, “60802 Dynamic Time Sync Error – Error Model & Monte Carlo Method Analysis”, IEC/IEEE 60802 contribution, March 2022

Background

- IEC/IEEE 60802 has a stated requirement of 1us time accuracy over 64 hops (i.e. 65 devices) with a goal of 100 hops (i.e. 101 devices).
- Prior to the development of the Monte Carlo Analysis, simulations of different configurations and parameters were carried out via Time Series Simulation. See [1], [2], [3], [4] and [5]. Typically...
  - 1 replication simulates 3,100 seconds.
  - To generate statistically significant results, 300 replications are run.
  - This takes 1 to 2 weeks, depending on various parameters.
- The Monte Carlo Analysis presented in these slides was developed over several months in order to provide faster iterations, albeit at the cost of some accuracy.
  - To generate the equivalent number of Sync message simulations as 300 replications of the Time Series Simulation (7,440,000) takes 10-16 minutes
  - The analysis also enables deep insights into the source of errors and how they accumulate.

Content

- History & Current Status
- Overview & Assumptions
- Timestamp Errors
- Clock Drift Errors
- Error Contributions & Accumulation
- Main Equations
- Tracking Error Contributions & Graphical Representations
- Algorithmic Improvements & Corrections

History & Current Status

- The development history of the Monte Carlo Analysis is mostly covered in a series of contributions to IEC/IEEE 60802. See [6], [7], [8] and [9].
- Over the course of development, modelling of additional errors was added to the original model as well as options to model Clock Drift distributions based on different temperature time-series temperature ramps.
- This contribution describes the current operation of the model and the Excel workbook used for post-processing some results.
- It is still intended to open-source the R Studio script which implements the model, although the date is TBD.

Overview & Assumptions

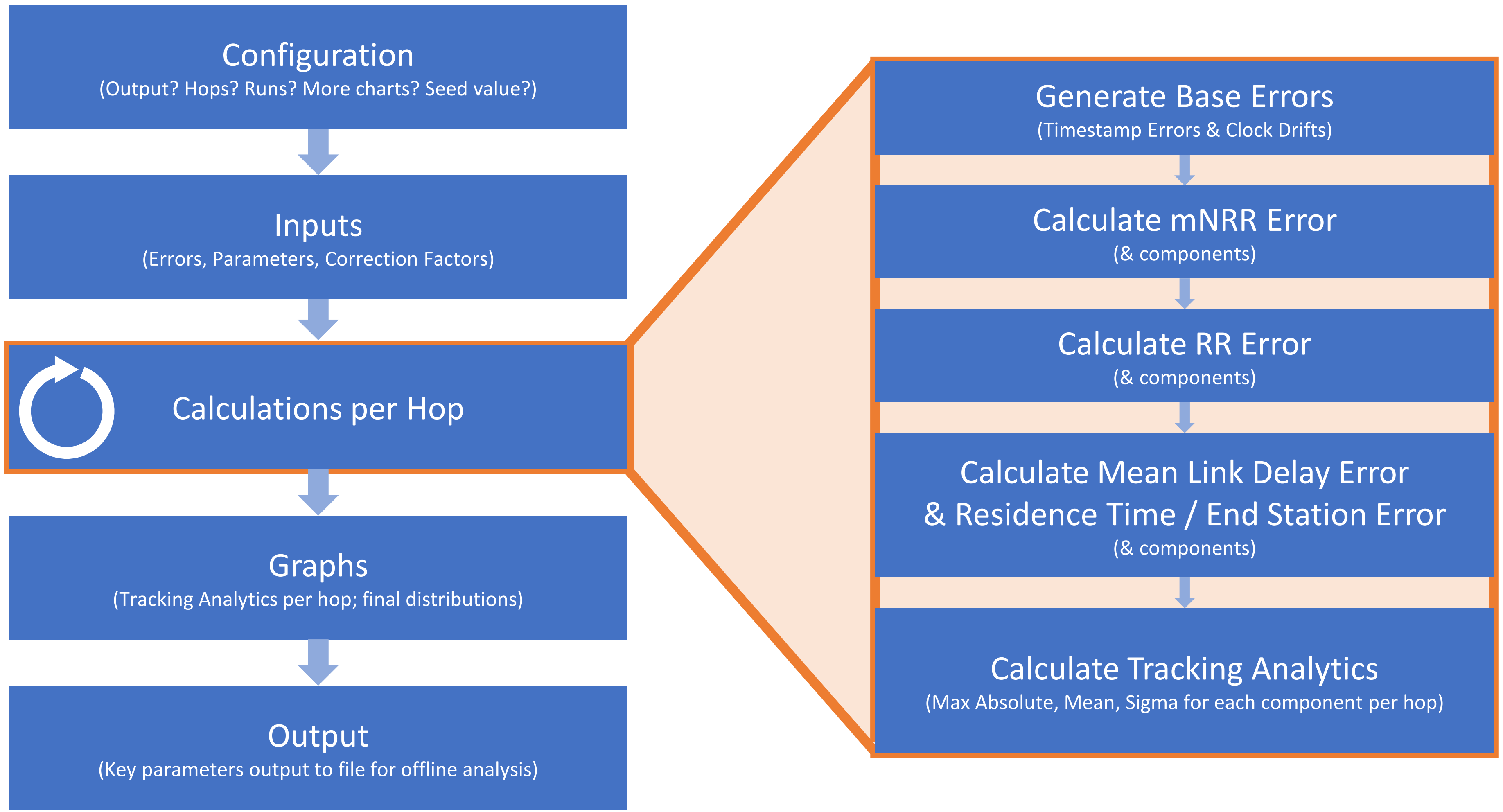
Overview – 1

- Implemented in R Studio (IDE for R)
- Models individual “runs”: single Sync message passing down a chain of nodes with all significant, associated errors.
- Script generates results for Hop 1 for all runs (typically 10,000 to several million)...then Hop 2...then Hop 3...
- Calculations are the same for every hop, with two exceptions:
  - Hop 1: first node is GM
  - Last Hop: no Residence Time; instead, there is End Station Error
- Script tracks error contributions from different sources
  - Full results (values for every error and contributing factor) for each node are calculated, but are not saved, which reduces memory footprint. Maximum absolute, mean and standard deviation values for every contributing error at every node across all runs are saved, e.g. the maximum absolute error across all runs of the contribution to Residence Time Error due to the timestamp component of Neighbor Rate Ratio via Rate Ratio at node 15 is saved.
  - Full results for final node are saved.

Overview – 2

- The script only models errors and values necessary to derive errors. Example: it does not model the Correction Field, only errors associated with it; it only models the period between pDelayResp messages to calculate the error due to Clock Drift during that interval.
- The script only models errors associated with the messaging protocol. It does not model Clock Source, Clock Target, Clock Master or Clock Slave. Neither does it model any filtering of the messaging information, i.e. it's results are most directly comparable to the "unfiltered" results from the Time Series simulation.
  - The script can be thought of, for each run, as focussed on modelling the error in the Correction Field when it arrives at the last node (usually in a Follow-up message after a Sync message) which is, at the time, the best estimate the node has of GM time. It then adds additional errors due to Rate Ratio and Clock Drift as the last node tries to track GM time prior to the arrival of the next Sync message (and Follow-up message).

RStudio Script Summary



Assumptions – 1

- The model assumes it is sufficient to account for only the major error contributors and only in enough detail to draw useful conclusions.
  - It doesn't model errors that would be swamped other larger errors in all realistic scenarios. Example: the effect of Rate Ratio error on Mean Link Delay Error, see [6].
  - It doesn't model the detail of ambient temperature on a physical system. Instead models a simple temperature ramp on a crystal oscillator (XO); the latter modelled as a cubic equation approximating the relationship between temperature and frequency offset. (This is the same as the Time Series Simulations.)
    - Note: a simpler model which generates clock drift based on a uniform probability between two values, i.e. no temperature modelling, is also available.

Assumptions – 2

- The model assumes that XO Clock Drift can be treated as linear within a single run, i.e. that for each clock a drift rate can be generated once and used for all calculations for that run.
  - This is a major simplification; one that places limits on the model's ability to model algorithms that attempt to correct for errors due to Clock Drift.
- The model assumes that some errors are uncorrelated due to the amount of time passing between their generation.
  - Example: timestamp granularity for pDelayReq and pDelayResp messages. See slide ZZZ.
- The model assumes that small (<20) ppm values can be added instead of carrying out a more accurate multiplication calculation
  - Example: the model assumes that 3ppm + 6ppm = 9ppm. The accurate value is 9.00018ppm.
  - This simplification saves processing time. It is assumes that the resulting inaccuracy is small compared with other errors that are being modelled.
  - 802.1AS makes the same assumption when calculating RR via accumulated NRR values

Assumptions – 3

- The model assumes that modelling errors generated between processing of Sync messages at **only** the last node in the chain is sufficient.
  - At all other nodes, only errors related to processing Sync messages are modelled.
  - The assumption is that although, for an individual run, the worst case DTE may not occur at the final node, the overall probability distribution of DTE will be worst at the final node.
- The model does not account for errors due to path delay asymmetry on the assumption that they are a) small relative to other errors and b) will tend to balance out over a long chain of hops.

Assumptions – 4

- The model assumes that there is no effective difference, as far as the errors ultimately being modelled is concerned, between 1-step and 2-step Sync messaging. For simplicity it therefore does not model behaviour related to Follow-up messaging.
    - The errors ultimately being modelled are at the End Station at the end of the chain, just prior to the arrival of the next Sync message.

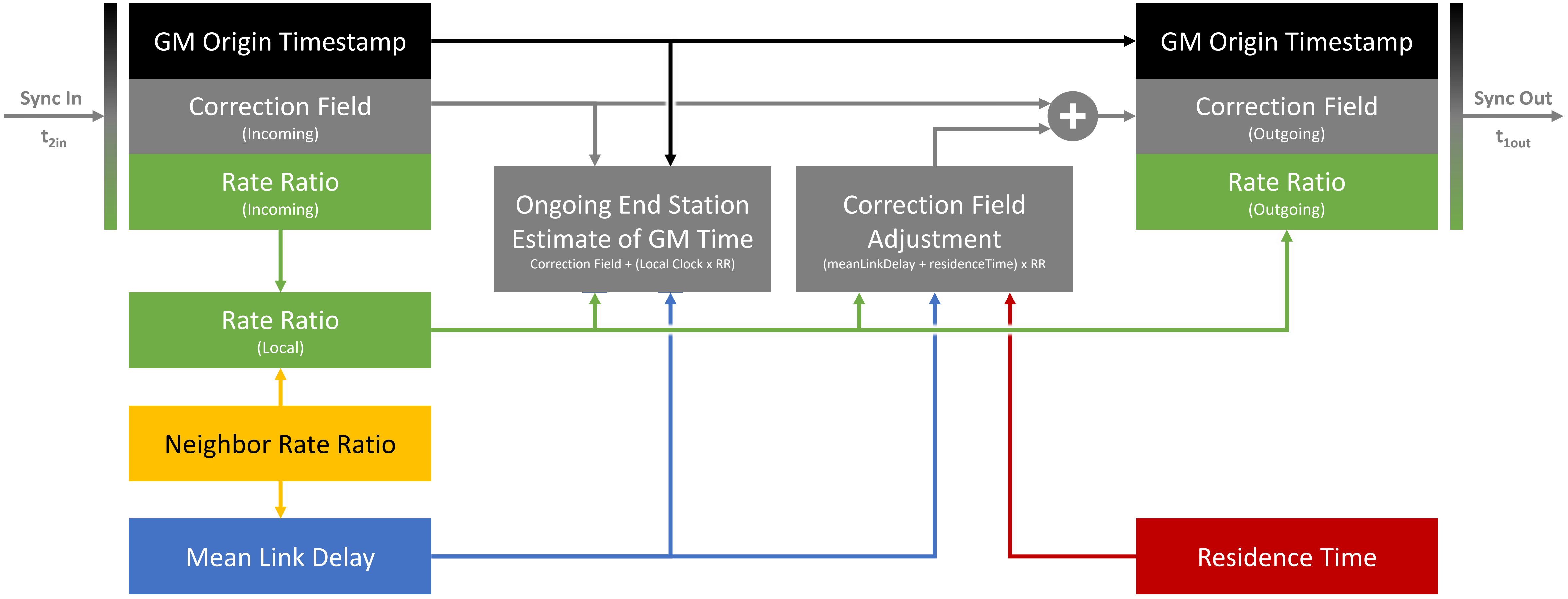
Parameters

Parameter	Default	Unit	Notes
pDelayInterval	1,000	ms	Limited to 1s x 2^n. Typical values are: 1,000ms; 500ms; 250ms; 125ms; 62.5ms; 31.25ms
syncInterval	125	ms	Limited to 1s x 2^n. Typical values are: 1,000ms; 500ms; 250ms; 125ms; 62.5ms; 31.25ms
pDelayTurnaround	10	ms	
residenceTime	10	ms	

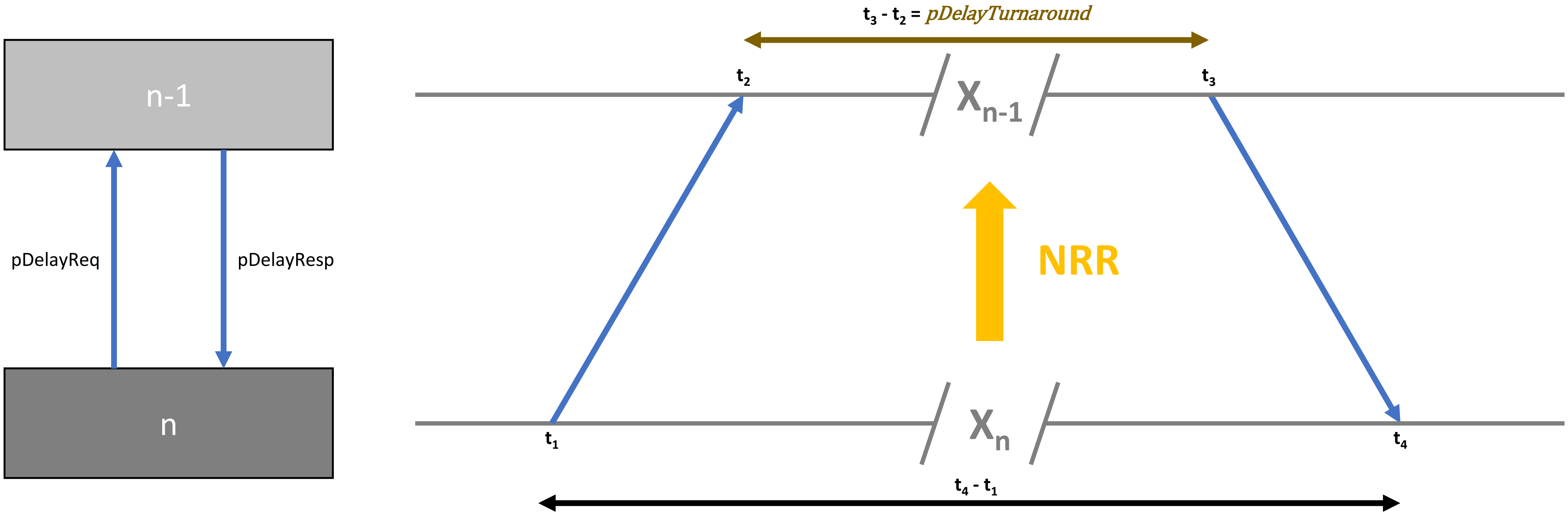
All equations can be traced back to constants, *Parameters*, *Timestamp Errors*, errors due to *Clock Drift*, or *Correction Parameters*

Error Contributors & Error Accumulation

Time Sync – Elements & Relationships



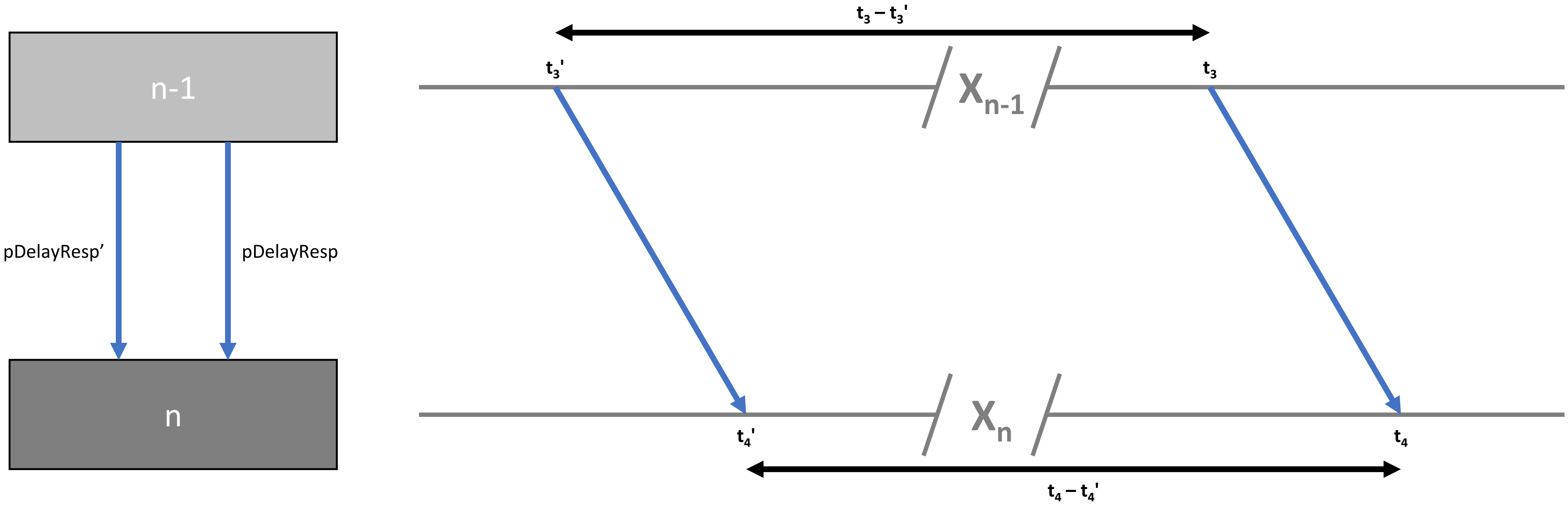
meanLinkDelay



$$meanLinkDelay = \left(\frac{(t_4 - t_1) - \frac{(t_3 - t_2)}{NRR}}{2} \right)$$

ns

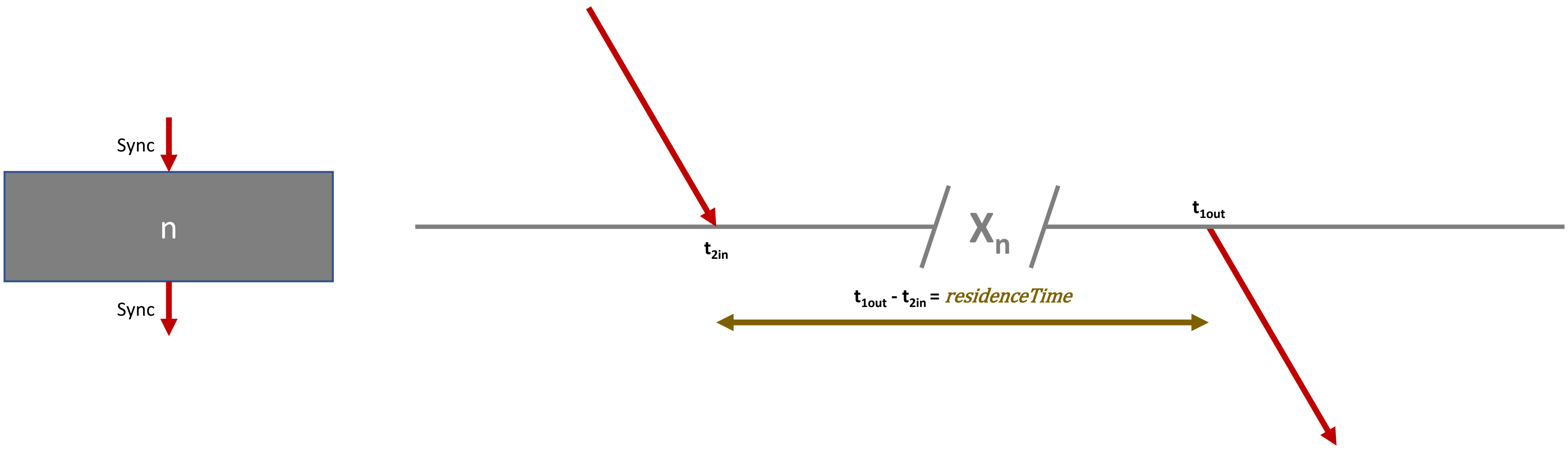
Measured Neighbor Rate Ratio (mNRR)



$$mNRR = \left(\frac{t_3 - t_3{}'}{t_4 - t_4{}'}\right)$$

ppm

Residence Time



$$residenceTime = (t_{1out} - t_{2in})$$

ns

Rate Ratio & Correction Field

- Rate Ratio (RR) is calculated via accumulated Neighbor Rate Ratios. At each node the local Rate Ratio (at Node n) is used to estimate the GM clock and passed on to the next node via an outgoing Sync message. It is calculated as follows...

$$\boldsymbol{RR}(n) = \boldsymbol{RR}(n-1) + \boldsymbol{mNRR}$$

ppm

- The outgoing correction field is calculated as follows...

$$correctionField(n) = correctionField(n-1) + \boldsymbol{RR}(\boldsymbol{meanLinkDelay} + \boldsymbol{residenceTime})$$

ns

  - The sum of meanLinkDelay (between the current and upstream node) and residenceTime (at the current node) gives the interval between reception of the incoming Sync message and transmission of the outgoing Sync message
  - Multiplication by RR translates this from Local Clock to Working Clock

Sources of Errors

- There are two types of error sources in the model...
  - Timestamp Errors; inaccuracies in measuring when messages are received or transmitted. There are two types.
    - Timestamp Granularity Error (TSGE) related to the measurement resolution.
    - Dynamic Timestamp Error (DTSE) related to accuracies inherent in the implementation, excluding TSGE.
  - Errors due to Clock Drift. If all frequency offsets were stable, there would be no errors due to Clock Drift...but they are not, and the events being modelled take place over a period of time. Thus errors occur due to the difference between...
    - Time when a measurement is effectively taken
    - Time when a measurement is used

    The model therefore includes Clock Drift and various relevant intervals, all modelled according to probability distributions.

Timestamp Error Parameters

- The model includes separate parameters for timestamp errors on transmitted (TX) and received messages (RX)

Error	Default	Unit
$TSGE_{TX}$	±4	ns
$TSGE_{RX}$	±4	ns
$DTSE_{TX}$	±4	ns
$DTSE_{RX}$	±4	ns

Timestamp Error Equations

- Both TSGE and DTSE are modelled via uniform distributions between a maximum and a minimum.
- Timestamp Granularity always results in a timestamp after the event occured...

$$\boldsymbol{Error_{TGSE}} = \sim U(0, +TSG)$$

...(where TSG is Timestamp Granularity) however, because the consequent errors are always in interval measurements which involve two events and two timestamps, modelling it as an error between ±TSG/2 is equivalent. In the R Studio script the parameter TGSE represents TSG/2...

$$\boldsymbol{Error_{TSGETX}} = \sim U\left(-\frac{TSG}{2}, +\frac{TSG}{2}\right) = \sim U(-\boldsymbol{TSGE_{TX}}, +\boldsymbol{TSGE_{TX}}) \qquad \boldsymbol{Error_{TSGERX}} = \sim U(-\boldsymbol{TSGE_{RX}}, +\boldsymbol{TSGE_{RX}})$$

- DTSE magnitude and probability distribution is implementation dependant, but implementations that deliver a uniform probability between a minimum and maximum, equally spread either side of zero, are common and a worst case.
    - Triangular or normal distributions will have fewer extreme errors.

$$\boldsymbol{Error_{DTSETX}} = \sim U(-\boldsymbol{DTSE_{TX}}, +\boldsymbol{DTSE_{TX}}) \qquad \boldsymbol{Error_{DTSERX}} = \sim U(-\boldsymbol{DTSE_{RX}}, +\boldsymbol{DTSE_{RX}})$$

Clock Drift Error Modelling

- Clock Drift Error is modelled as a combination of Clock Drift and the passage of time during relevant intervals.
- Clock Drift is modelled as a single value for each clock for each run. The script allows a choice of two classes of model for Clock Drift
  - Uniform distribution between a maximum and minimum (ppm/s)
  - Distribution based on a temperature cycle and a theoretical crystal oscillator (XO).
- For the second class, the value is generated in three steps...
  - A time (t): uniform random distribution between 0 and a maximum representing the period of a defined temperature cycle.
  - A temperature cycle that ramps temperature up and down between a minimum and maximum in a defined manner. This is used to translate the time (t) into a temperature value for a theoretical crystal oscillator (tempXO).
  - A frequency offset curve for a theoretical XO, based on measured data from a selection of representative XOs, modelled as a cubic equation. The first derivative of the cubic equation allows translation from the from time (t) and XO temperature (tempXO) to Clock Drift.
- There are also probability models for each of the relevant intervals

Clock Drift Error – Uniform Probability Clock Drift

- Model includes separate parameters for minimum and maximum clock drift for both GM and non-GM clocks.
- It also includes parameters for the fraction of GM and non-GM clocks that will experience drift.
  - This is done to emulate the behaviour of a temperature cycle that holds steady – usually at the maximum or minimum value – for a period of time. See section on Clock Drift probabilities based on a temperature cycle for more details.
  - The R Studio Script generates a value for Clock Drift and as well as a Yes/No value of 1 or 0 and multiplies them together.

Clock Drift Error – Uniform Probability Clock Drift Parameters

Error	Default	Unit
$clockDrift_{GMmax}$	+1.5	ppm/s
$clockDrift_{GMmin}$	-1.5	ppm/s
$clockDriftFraction_{GM}$	0.8	-
$clockDrift_{max}$	+1.5	ppm/s
$clockDrift_{min}$	-1.5	ppm/s
$clockDriftFraction$	0.8	-

Clock Drift Error – Uniform Probability Clock Drift Equations

$$Error_{clockDriftGM} = \sim U(\boldsymbol{clockDrift_{minGM}}, \boldsymbol{clockDrift_{maxGM}}) \times \sim B(1, \boldsymbol{clockDriftFraction_{GM}})$$

$$Error_{clockDrift} = \sim U(\boldsymbol{clockDrift_{min}}, \boldsymbol{clockDrift_{max}}) \times \sim B(1, \boldsymbol{clockDriftFraction})$$

- The fuction in R to generate random values according to a binomial probability distribution has three input parameters
  - n: number of values to generate; not shows above; equal to the number of runs
  - N; number of "trials" e.g. flips of the coin; in this case "1"
  - p; probability of success; in this case the probability of a clock instance experiencing drift, represented by a "1" (vs. "0")

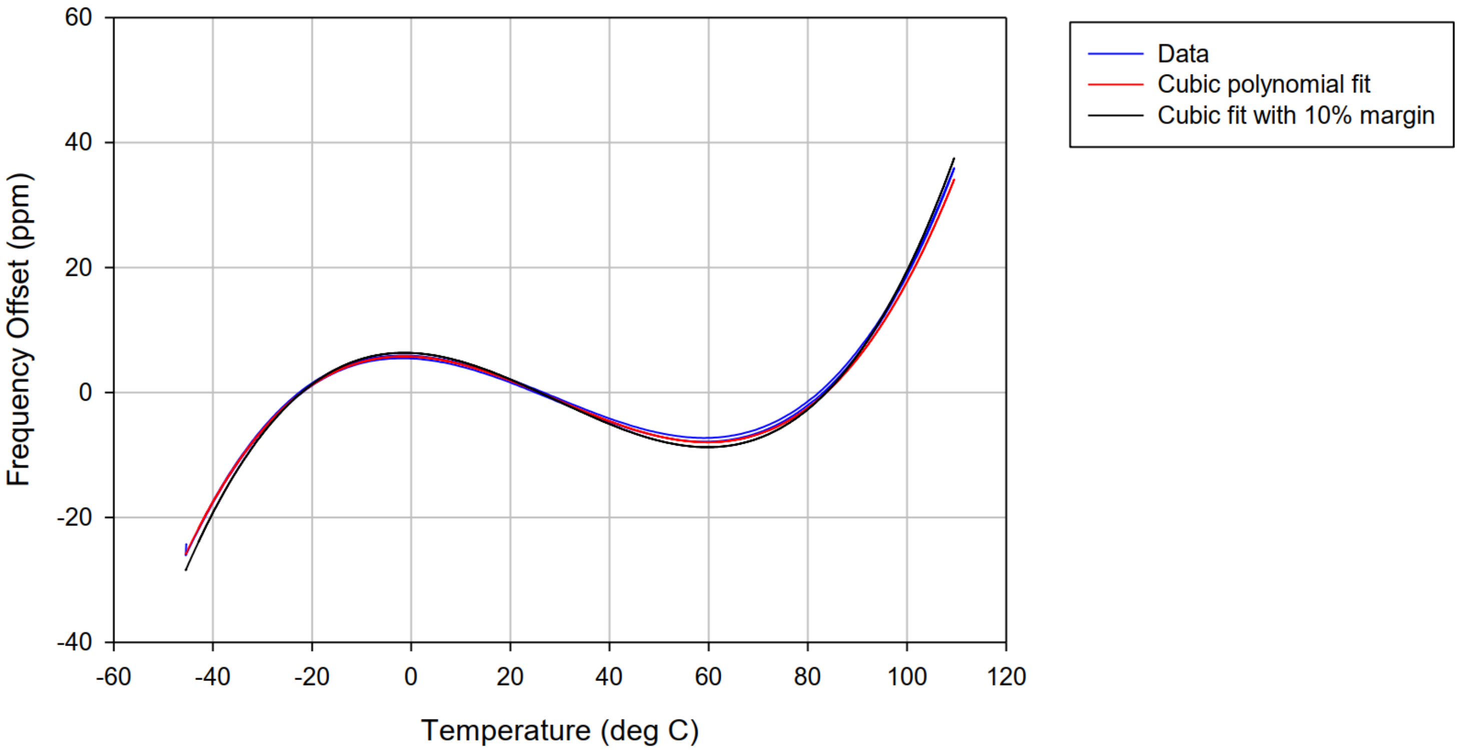
Clock Drift Error –
Clock Drift from Temp Cycle Modelling

- The temperature variation model has four sections over a complete cycle
  - Ramp from minimum to maximum temperature (Section A)
  - Hold at maximum temperature (Section B)
  - Ramp from maximum to minimum temperature (Section C)
  - Hold at minimum temperature (Section D)
- The R Studio script support three types of temperature ramp model
  - Linear
  - Sinusoidal
  - Half-sinusoidal
- The ramp is defined by temperature rate of change for linear ramp; duration of ramp for sinusoidal and half-sinusoidal. The hold period at maximum and minimum temperature is the same.
- The model for the XO's frequency offset is the same for all types of temperature ramp

Clock Drift Error – Offset Frequency Curve

Cubic Constants	**Value**
a	0.00012
b	-0.01005
c	-0.0305
d	5.73845

$$freqOffset = \boldsymbol{a}.tempXO^3 + \boldsymbol{b}.tempXO^2 + \boldsymbol{c}.tempXO + \boldsymbol{d}$$



From Geoff Garner, "Phase and Frequency Offset, and Frequency Drift Rate Time History Plots Based on New Frequency Stability Data", contribution to IEC/IEEE 60802, March 2021
The calculation of freqOffset is not used in the model but is included for completeness and in case the reader wishes to recreate the example graphs.

Clock Drift Error – Temperature Cycle Parameters

- The model includes a parameter to scale the Clock Drift up or down to emulate less or more accurate XOs. The default of 1 means no scaling; 0.5 results in half the amount; 2 in twice the amount. (The model has a separate parameter for GM node scaling and one for non-GM node scaling, but the equations on subsequent pages refer only to a single parameter of “*scale*”, since all other elements of the equations are the same.)

Error	Default	Unit	Notes
tempMax	85	°C	
tempMin	-20	°C	
tempRampRate	±1	°C/s	Only used for linear temperature ramp
tempRampPeriod	125	s	Only used for sinusoidal and half-sinusoidal temperature ramps
tempHold	30	s	
GMscale	1	-	
nonGMscale	1	-	

Clock Drift Error – Linear Temp Ramp Equations – 1

Input	Default Value	Unit
$\boldsymbol{tempMax}$	85	°C
$\boldsymbol{tempMin}$	-20	°C
$\boldsymbol{tempRampRate}$	±1	°C/s
$\boldsymbol{tempHold}$	30	s

$$tempCyclePeriod = 2 \times \left(\frac{\boldsymbol{tempMax} - \boldsymbol{tempMin}}{\boldsymbol{tempRampRate}} + \boldsymbol{tempHold}\right)$$

$$sectionA = \frac{\boldsymbol{tempMax} - \boldsymbol{tempMin}}{\boldsymbol{tempRampRate}}$$

$$sectionB = sectionA + \boldsymbol{tempHold}$$

$$sectionC = sectionB + sectionA$$

Clock Drift Error – Linear Temp Ramp Equations – 2

$t = \sim U(0, \boldsymbol{tempCyclePeriod})$

$if\,(0 \leq t < sectionA)$

$tempXO = \boldsymbol{tempMin} + \boldsymbol{tempRampRate}.t$

$tempRoC = \boldsymbol{tempRampRate}$

$\boldsymbol{clockDrift} = (3.\boldsymbol{a}.tempXO^2 + 2.\boldsymbol{b}.tempXO + \boldsymbol{c}) \times \boldsymbol{tempRampRate} \times \boldsymbol{scale}$

$if\,(sectionA \leq t < sectionB)$

$tempXO = \boldsymbol{tempMax}$

$tempRoC = 0$

$\boldsymbol{clockDrift} = 0$

$if\,(sectionB \leq t < sectionC)$

$tempXO = \boldsymbol{tempMax} - \boldsymbol{tempRampRate}.(t - sectionB)$

$tempRoC = -\boldsymbol{tempRampRate}$

$\boldsymbol{clockDrift} = -(3.\boldsymbol{a}.tempXO^2 + 2.\boldsymbol{b}.tempXO + \boldsymbol{c}) \times \boldsymbol{tempRampRate} \times \boldsymbol{scale}$

$if\,(sectionC \leq t)$

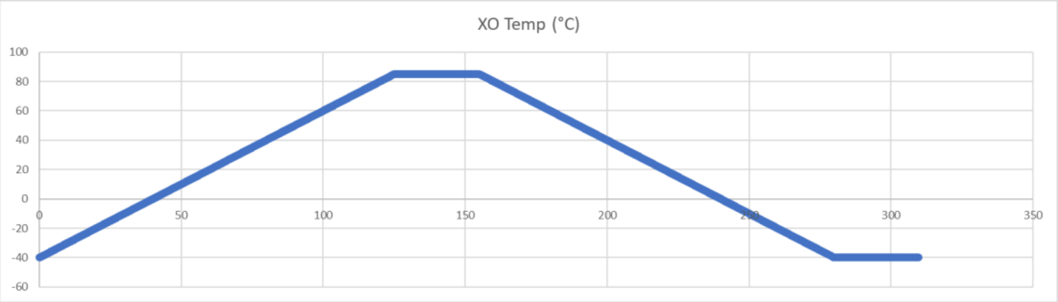
$tempXO = \boldsymbol{tempMin}$

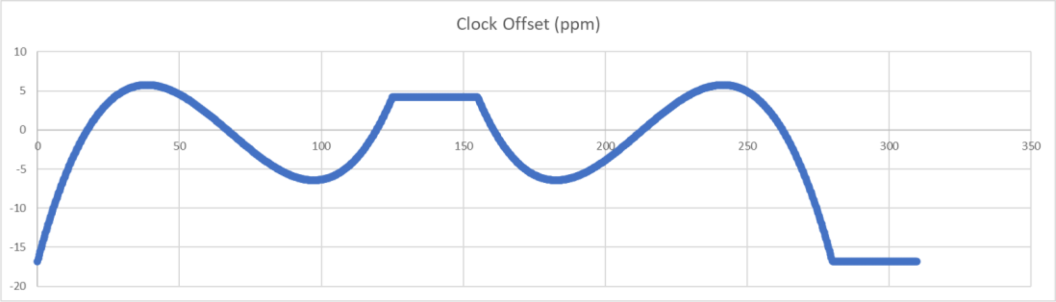
$tempRoC = 0$

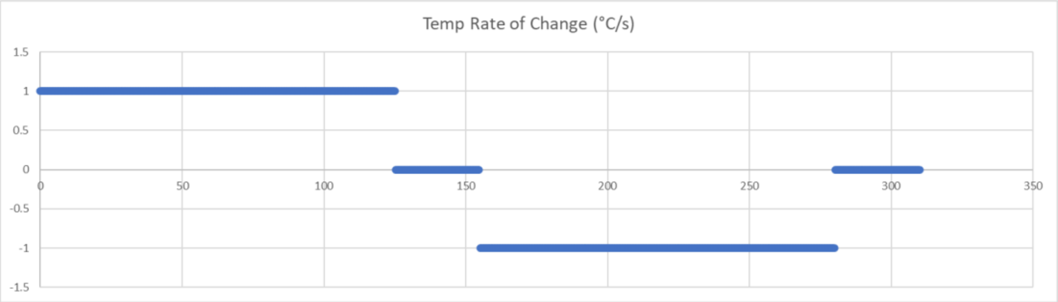
$\boldsymbol{clockDrift} = 0$

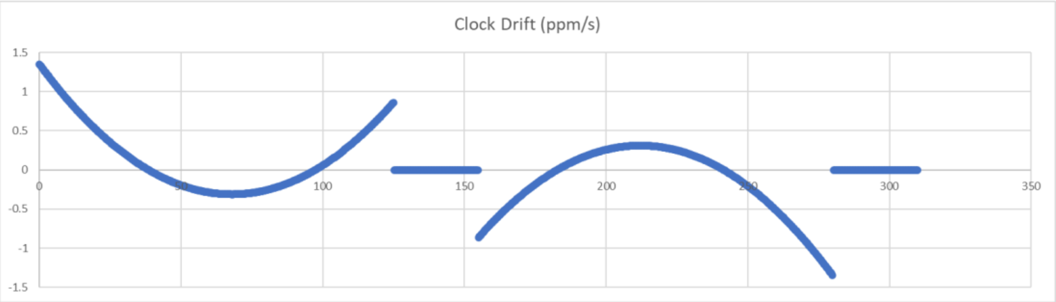
The calculation of tempRoC is not used in the model but is included for completeness and in case the reader wishes to generate the example graphs.

Clock Drift Example – Linear Temperature Ramp: 1°C/s ↕ (125s ↕)









Inputs		
Temp Max	85	°C
Temp Min	-40	°C
Temp Ramp Rate	1	°C/s
Temp Hold	30	s

Temp Rate of Change		
MAX	1.00	°C/s
MIN	-1.00	°C/s

Clock Drift		
MAX	1.35	ppm/s
MIN	-1.35	ppm/s

Clock Drift Error – Sinusoidal Temp Ramp Equations – 1

Input	Default Value	Unit
tempMax	85	°C
tempMin	-20	°C
tempRampPeriod	125	s
tempHold	30	s

$$tempCyclePeriod = 2 \times (\boldsymbol{tempRampPeriod} + \boldsymbol{tempHold})$$

$$tempDeviation = \frac{\boldsymbol{tempMax} - \boldsymbol{tempMin}}{2}$$

$$tempMidpoint = \boldsymbol{tempMax} - tempDeviation$$

$$\omega = \frac{\pi}{\boldsymbol{tempRampPeriod}}$$

$$sectionA = \boldsymbol{tempRampPeriod}$$

$$sectionB = sectionA + \boldsymbol{tempHold}$$

$$sectionC = sectionB + sectionA$$

Clock Drift Error – Sinusoidal Temp Ramp Equations – 2

$t = \sim U(0, \boldsymbol{tempCyclePeriod})$

$if\,(0 \le t < sectionA)$

$tempXO = \boldsymbol{tempMidpoint} - \boldsymbol{tempDeviation}.\cos(\omega.t)$

$tempRoC = \omega.\boldsymbol{tempDeviation}.sin(\omega.t)$

$\boldsymbol{clockDrift} = (3.\boldsymbol{a}.tempXO^2 + 2.\boldsymbol{b}.tempXO + \boldsymbol{c}) \times \left(\omega.\boldsymbol{tempDeviation}.sin(\omega.t)\right).\boldsymbol{scale}$

$if\,(sectionA \le t < sectionB)$

$tempXO = \boldsymbol{tempMax}$

$tempRoC = 0$

$\boldsymbol{clockDrift} = 0$

$if\,(sectionB \le t < sectionC)$

$tempXO = \boldsymbol{tempMidpoint} + \boldsymbol{tempDeviation}.\cos\left(\omega.(t - sectionB)\right)$

$tempRoC = -\omega.\boldsymbol{tempDeviation}.sin\left(\omega.(t - sectionB)\right)$

$\boldsymbol{clockDrift} = -(3.\boldsymbol{a}.tempXO^2 + 2.\boldsymbol{b}.tempXO + \boldsymbol{c}) \times \left(\omega.\boldsymbol{tempDeviation}.sin\left(\omega.(t - sectionB)\right)\right).\boldsymbol{scale}$

$if\,(sectionC \le t)$

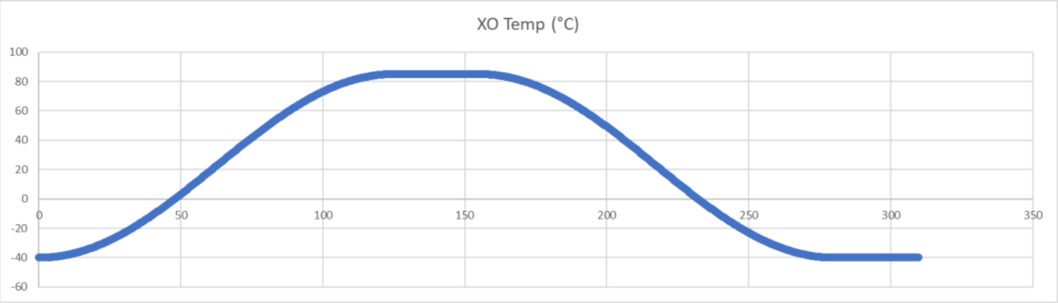
$tempXO = \boldsymbol{tempMin}$

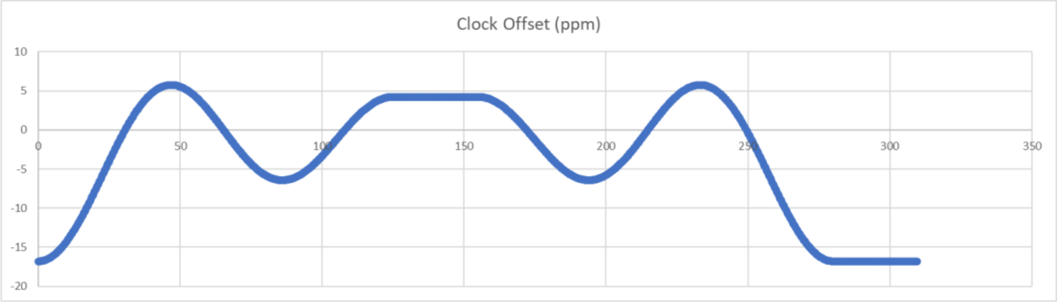
$tempRoC = 0$

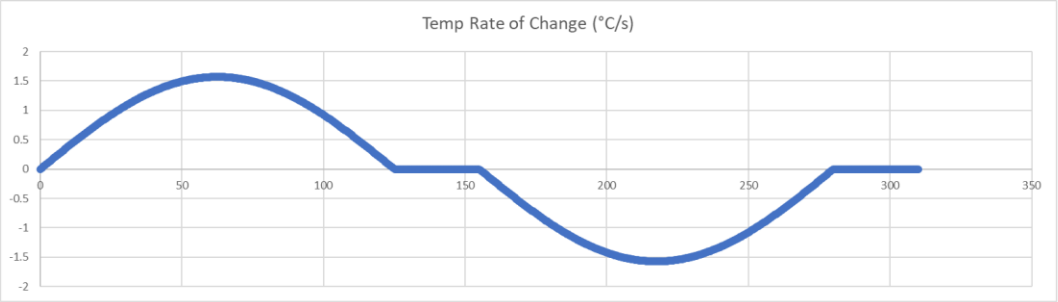
$\boldsymbol{clockDrift} = 0$

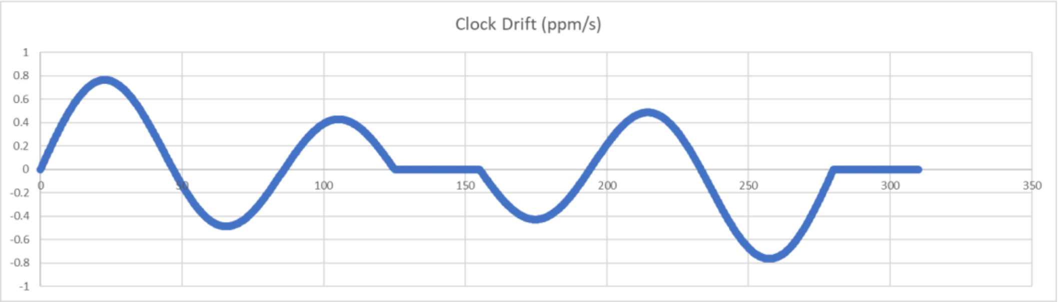
The calculation of tempRoC is not used in the model but is included for completeness and in case the reader wishes to generate the example graphs.

Clock Drift Example –
Sinusoidal Temperature Ramp: 125s ↕









Inputs		
Temp Max	85	°C
Temp Min	-40	°C
Temp Ramp Period	125	s
Temp Hold	30	s

Temp Rate of Change		
MAX	1.57	°C/s
MIN	-1.57	°C/s

Clock Drift		
MAX	0.76	ppm/s
MIN	-0.76	ppm/s

Clock Drift Error – Half-sinusoidal Temp Ramp Equations – 1

Input	Typical Value	Unit
tempMax	85	°C
tempMin	-20	°C
tempRampPeriod	125	s
tempHold	30	s

$$tempCyclePeriod = 2 \times (\boldsymbol{tempRampPeriod} + \boldsymbol{tempHold})$$

$$tempRange = \boldsymbol{tempMax} - \boldsymbol{tempMin}$$

$$\tau = \frac{\pi}{\boldsymbol{tempRampPeriod} \times 2}$$

$$sectionA = \boldsymbol{tempRampPeriod}$$

$$sectionB = sectionA + \boldsymbol{tempHold}$$

$$sectionC = sectionB + sectionA$$

Clock Drift Error – Half-sinusoidal Temp Ramp Equations – 2

$t = \sim U(0, \boldsymbol{tempCyclePeriod})$

$if\,(0 \leq t < sectionA)$

$tempXO = \boldsymbol{tempMin} + \boldsymbol{tempRange}.\sin(\tau.t)$

$tempRoC = \tau.\boldsymbol{tempRange}.cos(\tau.t)$

$\boldsymbol{clockDrift} = (3.\boldsymbol{a}.tempXO^2 + 2.\boldsymbol{b}.tempXO + \boldsymbol{c}) \times (\tau.\boldsymbol{tempRange}.cos(\tau.t)).\boldsymbol{scale}$

$if\,(sectionA \leq t < sectionB)$

$tempXO = \boldsymbol{tempMax}$

$tempRoC = 0$

$\boldsymbol{clockDrift} = 0$

$if\,(sectionB \leq t < sectionC)$

$tempXO = \boldsymbol{tempMax} - \boldsymbol{tempRange}.\sin\big(\tau.(t - sectionB)\big)$

$tempRoC = -\tau.\boldsymbol{tempRange}.cos\big(\tau.(t - sectionB)\big)$

$\boldsymbol{clockDrift} = -(3.\boldsymbol{a}.tempXO^2 + 2.\boldsymbol{b}.tempXO + \boldsymbol{c}) \times \Big(\tau.\boldsymbol{tempRange}.cos\big(\tau.(t - sectionB)\big)\Big).\boldsymbol{scale}$

$if\,(sectionC \leq t)$

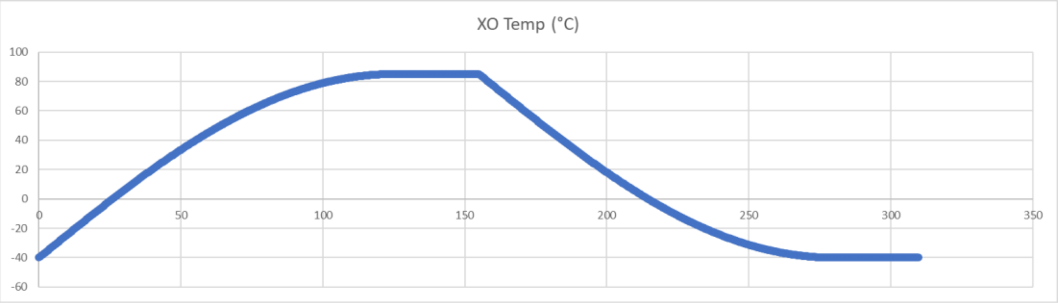
$tempXO = \boldsymbol{tempMin}$

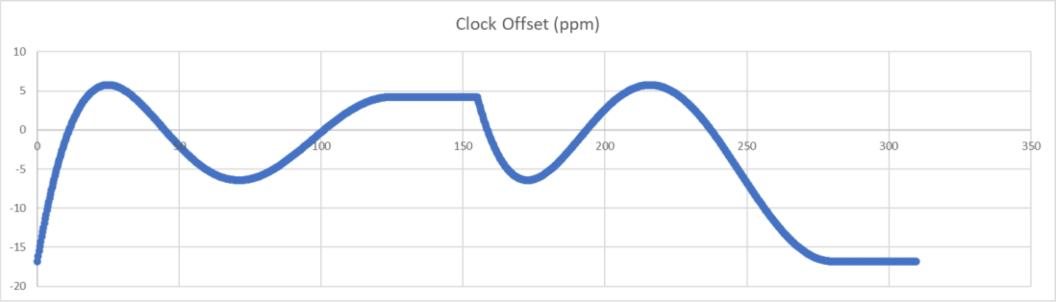
$tempRoC = 0$

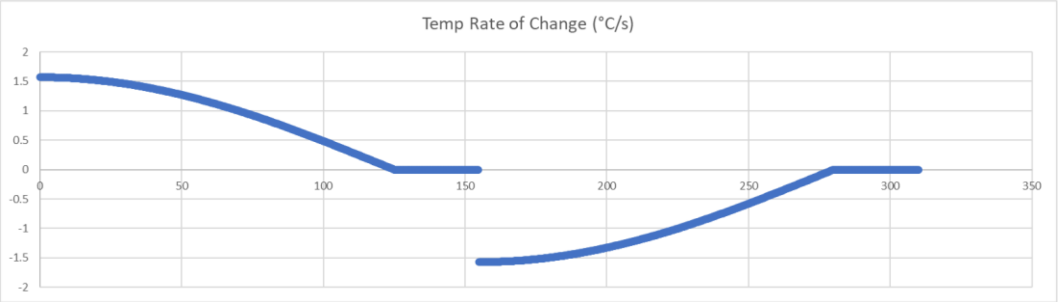
$\boldsymbol{clockDrift} = 0$

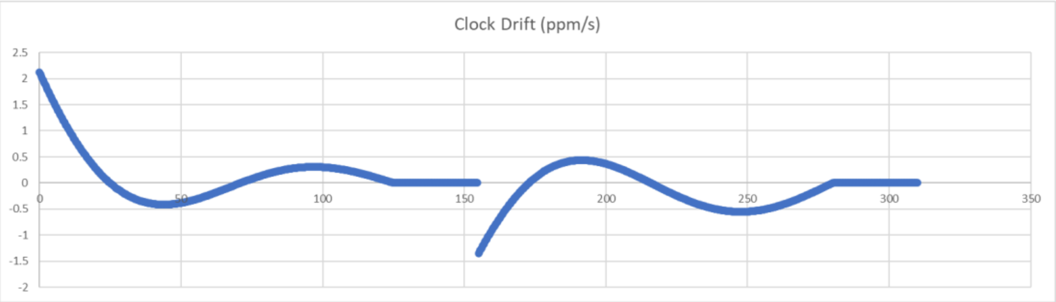
The calculation of tempRoC is not used in the model but is included for completeness and in case the reader wishes to generate the example graphs.

Clock Drift Example – Half-Sinusoidal Temperature Ramp: 125s ↕







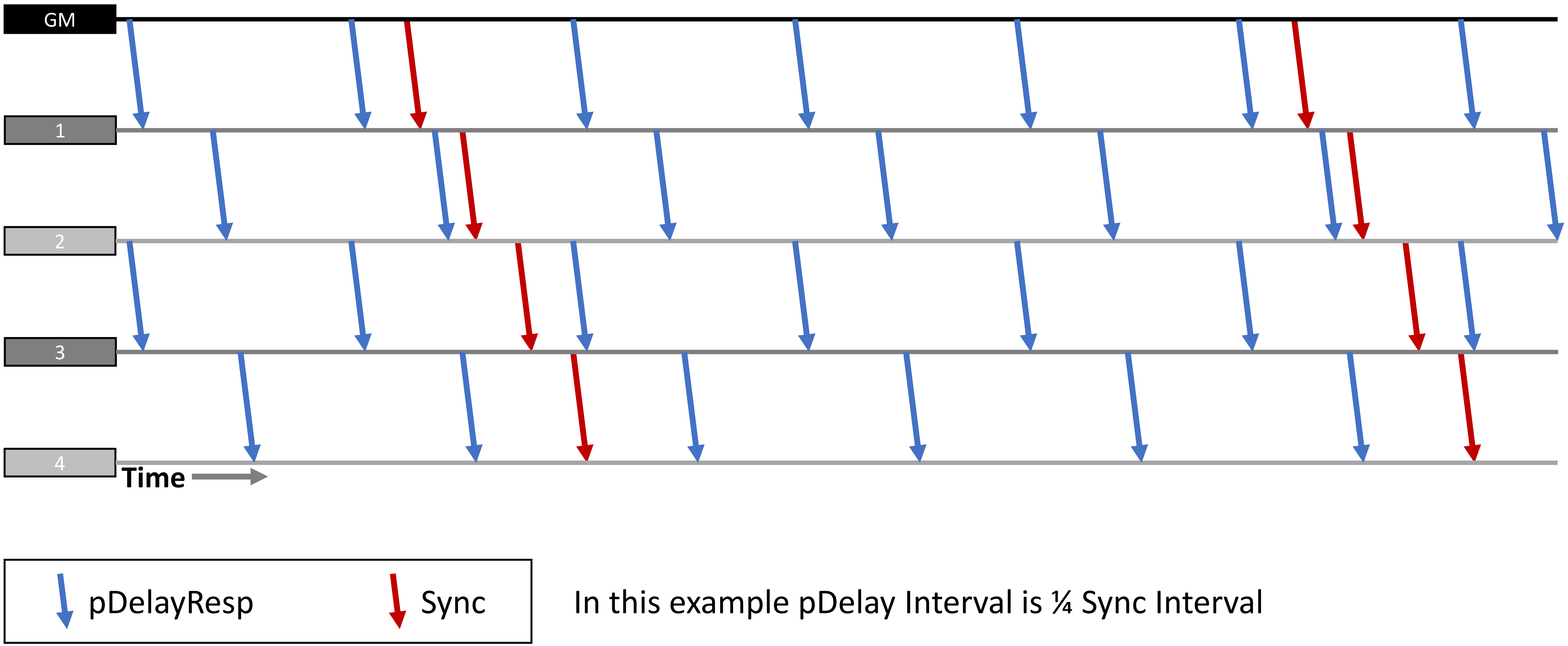


Inputs		
Temp Max	85	°C
Temp Min	-40	°C
Temp Ramp Period	125	s
Temp Hold	30	s

Temp Rate of Change		
MAX	1.57	°C/s
MIN	-1.57	°C/s

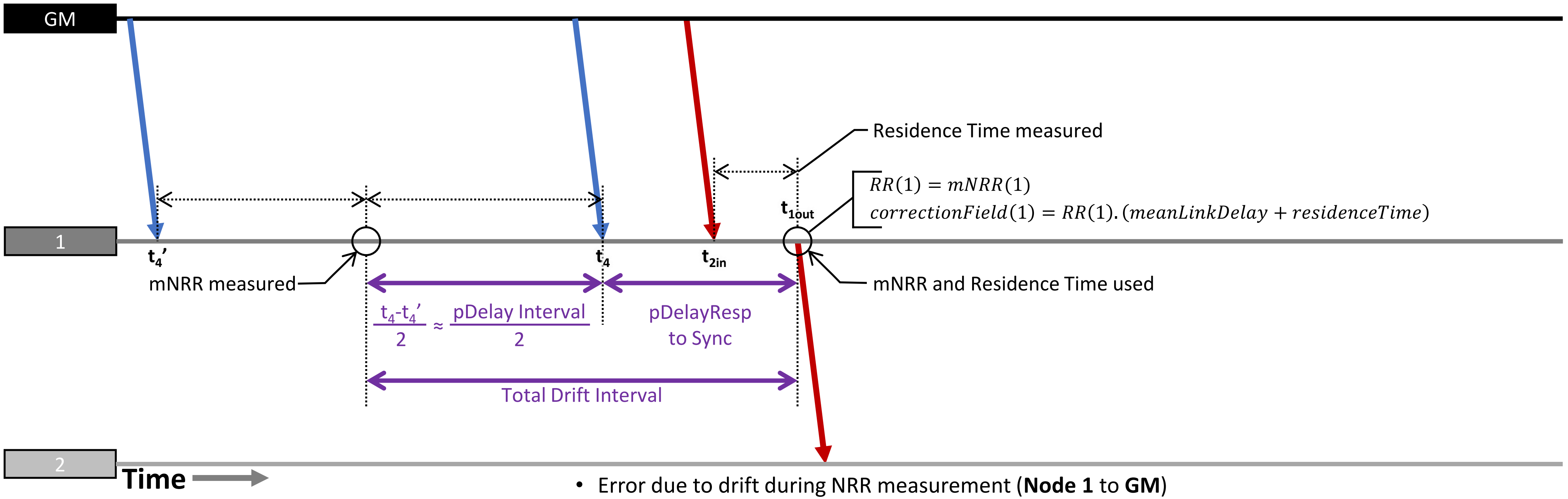
Clock Drift		
MAX	2.12	ppm/s
MIN	-1.35	ppm/s

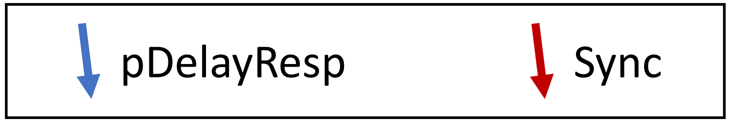
Clock Drift Error – Relevant Intervals 4 Hops



In this example pDelay Interval is ¼ Sync Interval

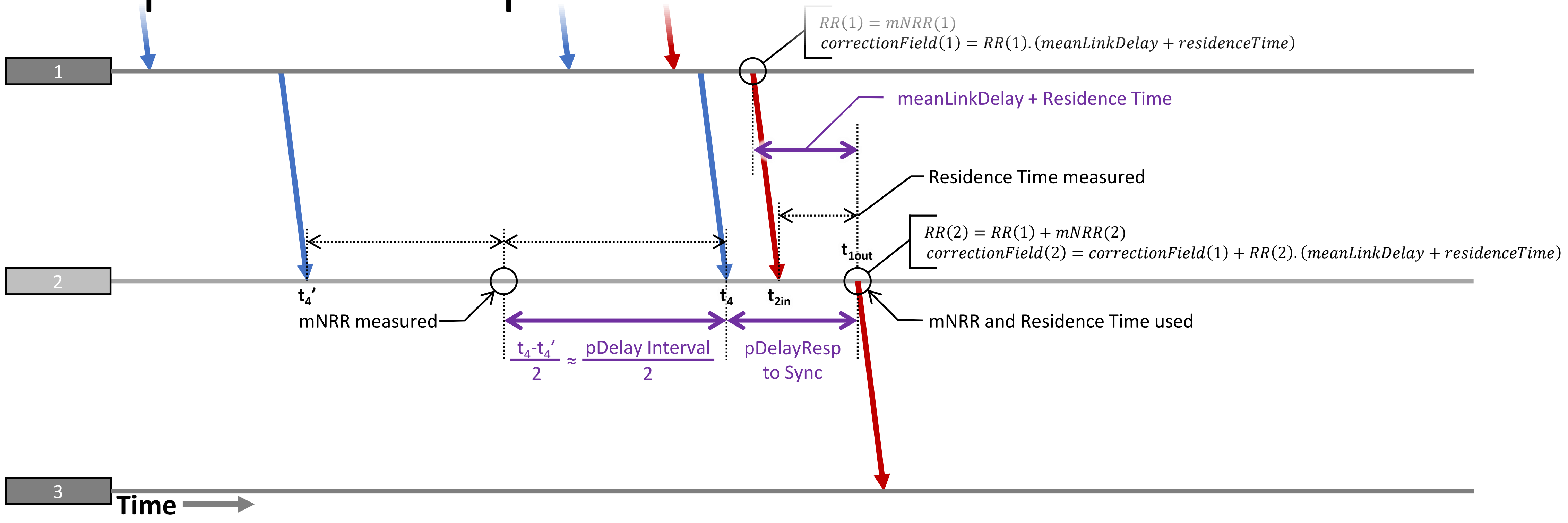
Clock Drift Error – Relevant Intervals
4 Hops – 1st Hop

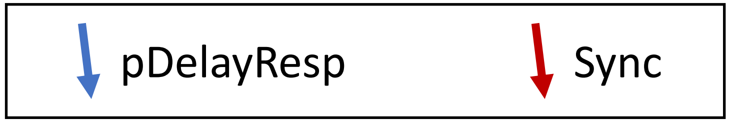




- Error due to drift during NRR measurement (**Node 1** to **GM**)
  - Interval is half t_4-t_4' which is nominally half the pDelay Interval, but actual pDelay Interval varies.
- Error due to drift between measuring and using NRR (**Node 1** to **GM**)
  - Interval is between zero and the maximum pDelay Interval, which is larger than the nominal pDelay Interval.
  - Assumes that pDelayResp arriving between t2in and t1out will trigger new mNRR calculation; not unreasonable as information is included in Follow-up, not Sync, if 2-step Sync is used.
- Error due to drift during Residence Time measurement (**Node 1** to **GM**)
  - meanLinkDelay is measured separately, is much smaller, and can be averaged to remove errors, so is ignored.

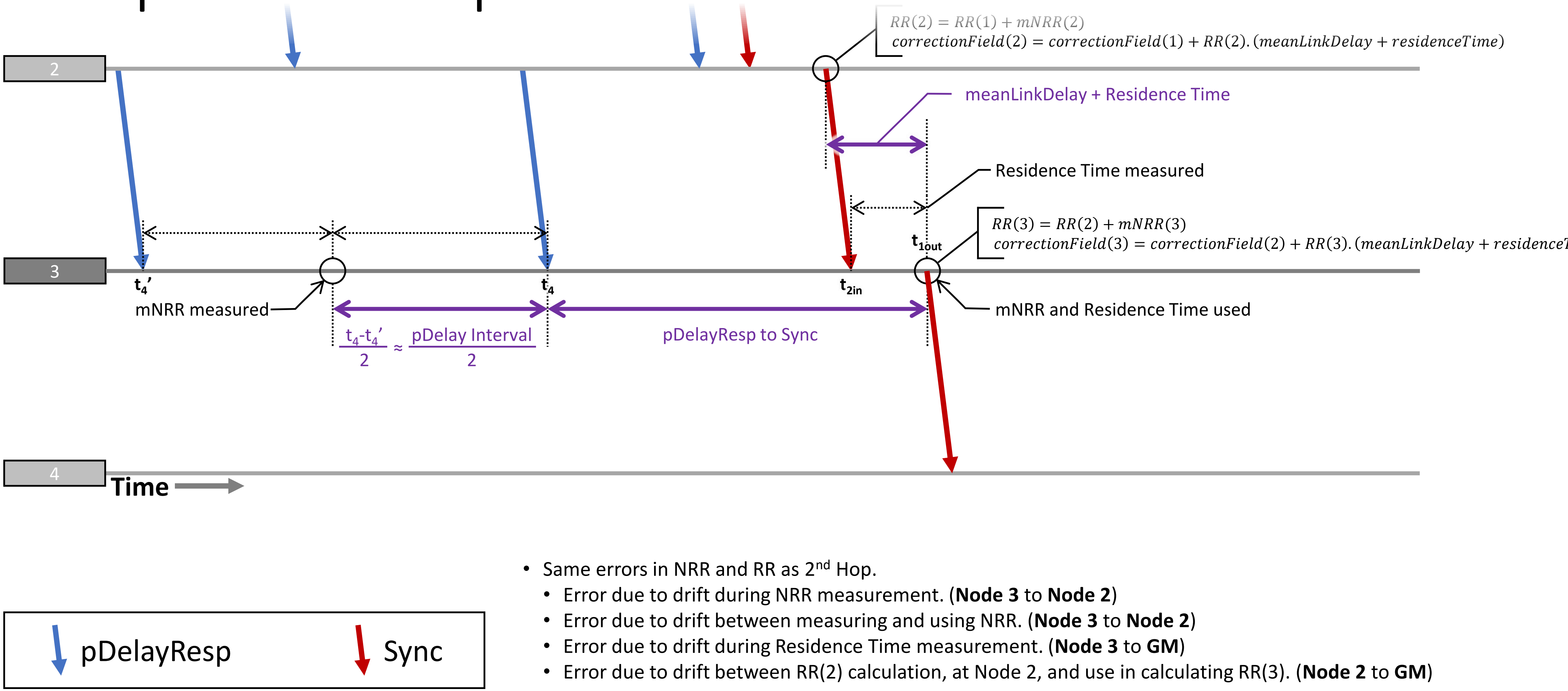
Clock Drift Error – Relevant Intervals
4 Hops – 2nd Hop





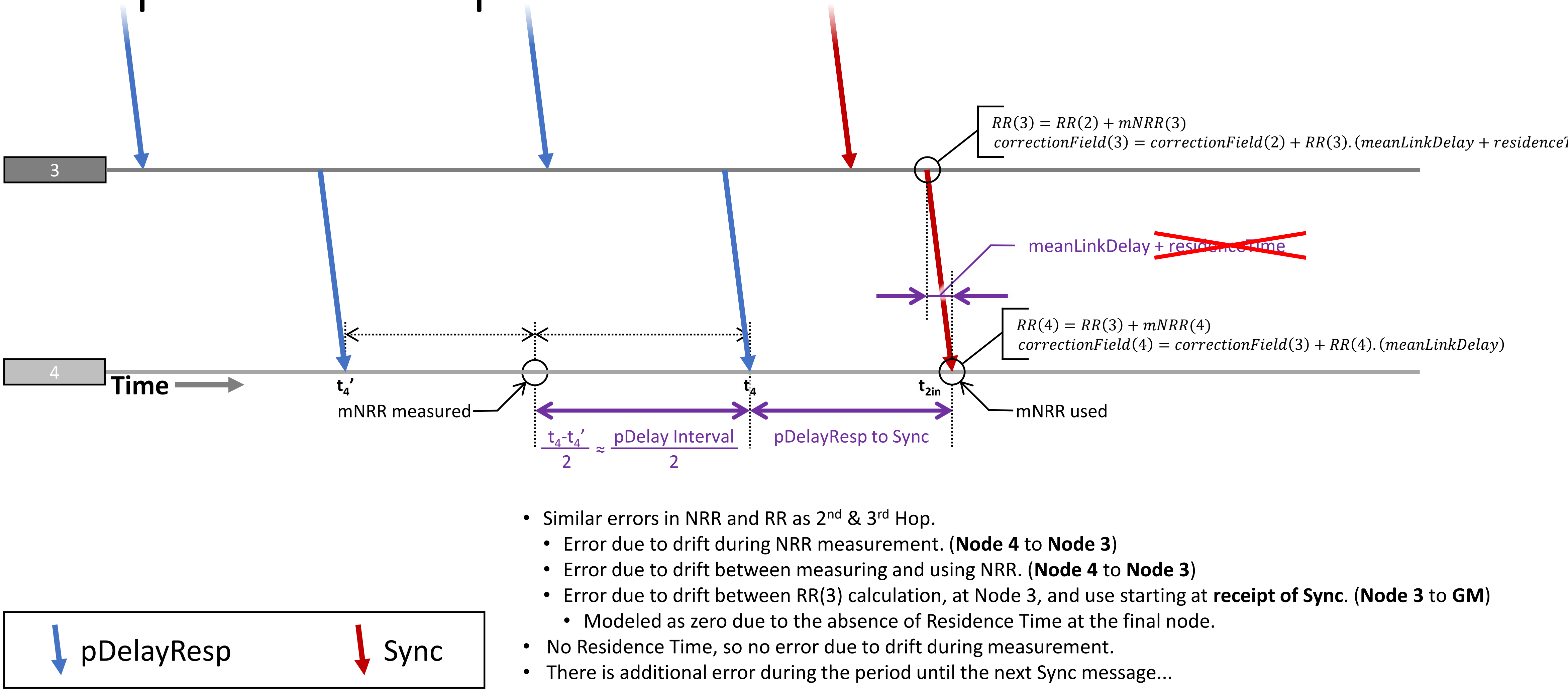
- Same errors in mNRR as 1st Hop.
  - Error due to drift during NRR measurement. (**Node 2** to **Node 1**)
  - Error due to drift between measuring and using NRR. (**Node 2** to **Node 1**)
  - Error due to drift during Residence Time measurement. (**Node 2** to **GM**)
- Additional error from drift between RR(1) calculation, at Node 1, and use in calculating RR(2). (**Node 1** to **GM**)
  - In the model the contribution from meanLinkDelay is ignored; only Residence Time is used.

Clock Drift Error – Relevant Intervals
4 Hops – 3rd Hop



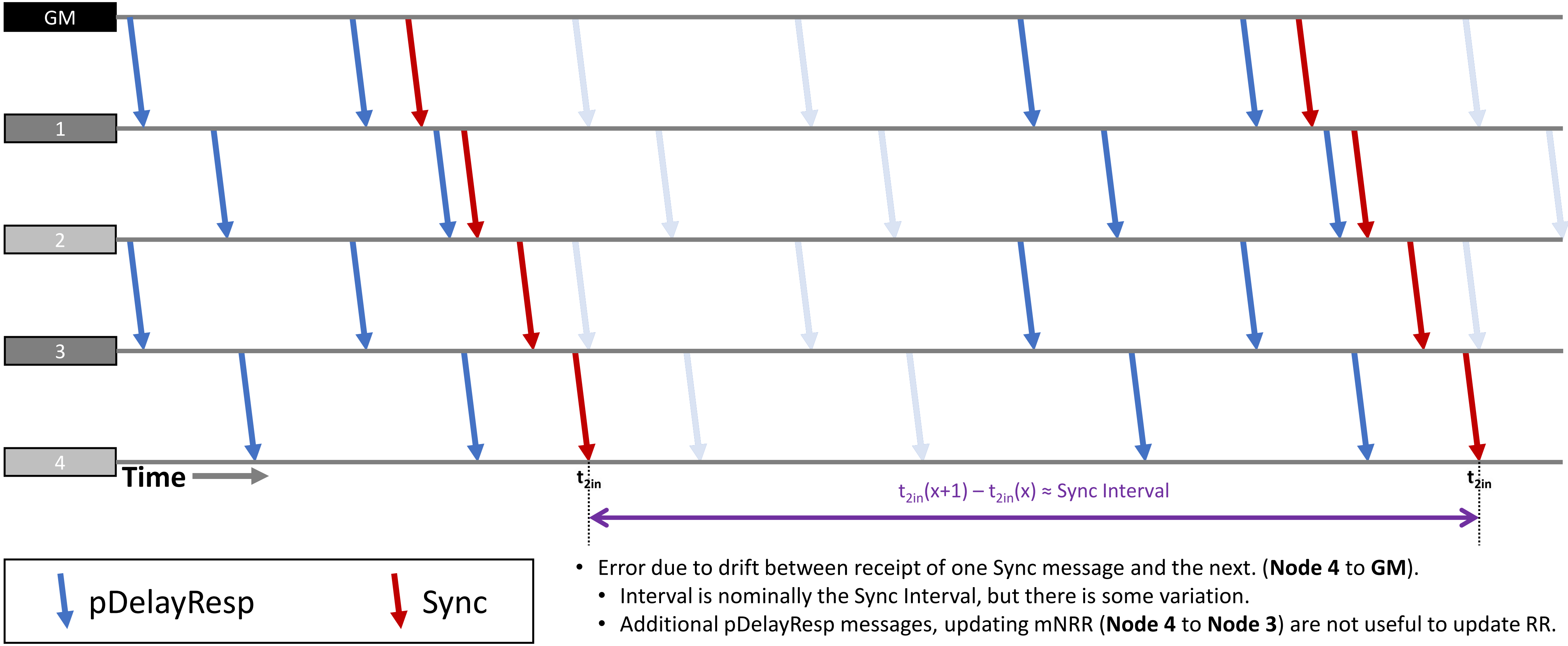
- Same errors in NRR and RR as 2nd Hop.
  - Error due to drift during NRR measurement. (**Node 3** to **Node 2**)
  - Error due to drift between measuring and using NRR. (**Node 3** to **Node 2**)
  - Error due to drift during Residence Time measurement. (**Node 3** to **GM**)
  - Error due to drift between RR(2) calculation, at Node 2, and use in calculating RR(3). (**Node 2 to GM**)

Clock Drift Error – Relevant Intervals
4 Hops – 4th Hop



- Similar errors in NRR and RR as 2nd & 3rd Hop.
  - Error due to drift during NRR measurement. (**Node 4** to **Node 3**)
  - Error due to drift between measuring and using NRR. (**Node 4** to **Node 3**)
  - Error due to drift between RR(3) calculation, at Node 3, and use starting at **receipt of Sync**. (**Node 3** to **GM**)
    - Modeled as zero due to the absence of Residence Time at the final node.
- No Residence Time, so no error due to drift during measurement.
- There is additional error during the period until the next Sync message...

Clock Drift Error – Relevant Intervals
4 Hops – 4th Hop



- Error due to drift between receipt of one Sync message and the next. (**Node 4** to **GM**).
  - Interval is nominally the Sync Interval, but there is some variation.
  - Additional pDelayResp messages, updating mNRR (**Node 4** to **Node 3**) are not useful to update RR.

Clock Drift Error – Relevant Intervals Summary – 1

- There are six relevant intervals...

1. Effective NRR Measurement → Actual NRR Measurement
   - The relevant drift is between the current node's clock and the upstream node's clock.
   - NRR is measured via information from a pair of pDelayResp messages. As Clock Drift is assumed to be linear, the effective measurement point is half-way between the two. The actual measurement point is at receipt of the second message.
   - The interval between the two pDelayResp messages is nominally the pDelay Interval. IEEE 1588 defines the permitted minimum and maximum interval as 90% and 130% of the nominal value. [See IEEE 1588-2019 9.5.13.2]
   - The interval is modelled as a uniform distribution between these two.

$$T_{pdelay2pdelay} = \sim U(\boldsymbol{pdelayInterval}.0.9, \boldsymbol{pdelayInterval}.1.3)$$

   - Note: see section on Algorithmic Improvements & Corrections for how this calculation changes if an older pDelayResp message is used as the first of the pair.

Clock Drift Error – Relevant Intervals Summary – 2

2. Actual NRR Measurement → NRR Use
   - The relevant drift is between the current node's clock and the upstream node's clock.
   - For all hops other than the last, NRR is used when RR for the outgoing Sync message is calculated. For the last hop, NRR is used when RR is calculated for the working clock of the local device to use until the arrival of the next Sync message.
   - In either case, the interval is modelled as a uniform distribution between zero and the interval between receipt of pDelayResp messages, i.e. random depending on the phase between Sync and pDelay messaging.
   - The interval between receipt of pDelayResp messages is modelled in the same way as for the previous error, i.e. between 90% and 130% of the nominal value. [See 802.1AS-2020 9.5.13.2]

Clock Drift Error – Relevant Intervals Summary – 3

3. RR Calc at upstream Node → RR Use at Current Node
    - The relevant drift is between the upstream node's clock and the GM's clock.
    - For all hops other than the last, the interval is the path delay between the upstream node and the current node plus the residence time at the current node. For the last node there is no residence time, so it's only the path delay.
    - The path delay is sufficiently small that it can ignored. [See assumption that it's OK to not model small errors that will be swamped by larger ones.]
    - For all hops other than the last, the delay is modelled as equal to the *residenceTime* parameter. For the last hop it is not modelled, i.e. per the previous point, the error is assumed to be small enough to ignore.

Clock Drift Error – Relevant Intervals Summary – 4

4. Start of Residence Time Measurement → End of Measurement
   - The relevant drift is between the current node's clock and the GM's clock.
   - When calculating the Correction Field, the goal is to measure Residence Time in terms of the GM clock. It is measured using the Local Clock with the interval multiplied by the Rate Ratio to translate it into GM time. There are two sources of error:
     - Error in measuring the interval.
     - Error in Rate Ratio.

     Clock drift during this interval affects the former.
   - It is modelled as equal to the *residenceTime* parameter.

Clock Drift Error – Relevant Intervals Summary – 5

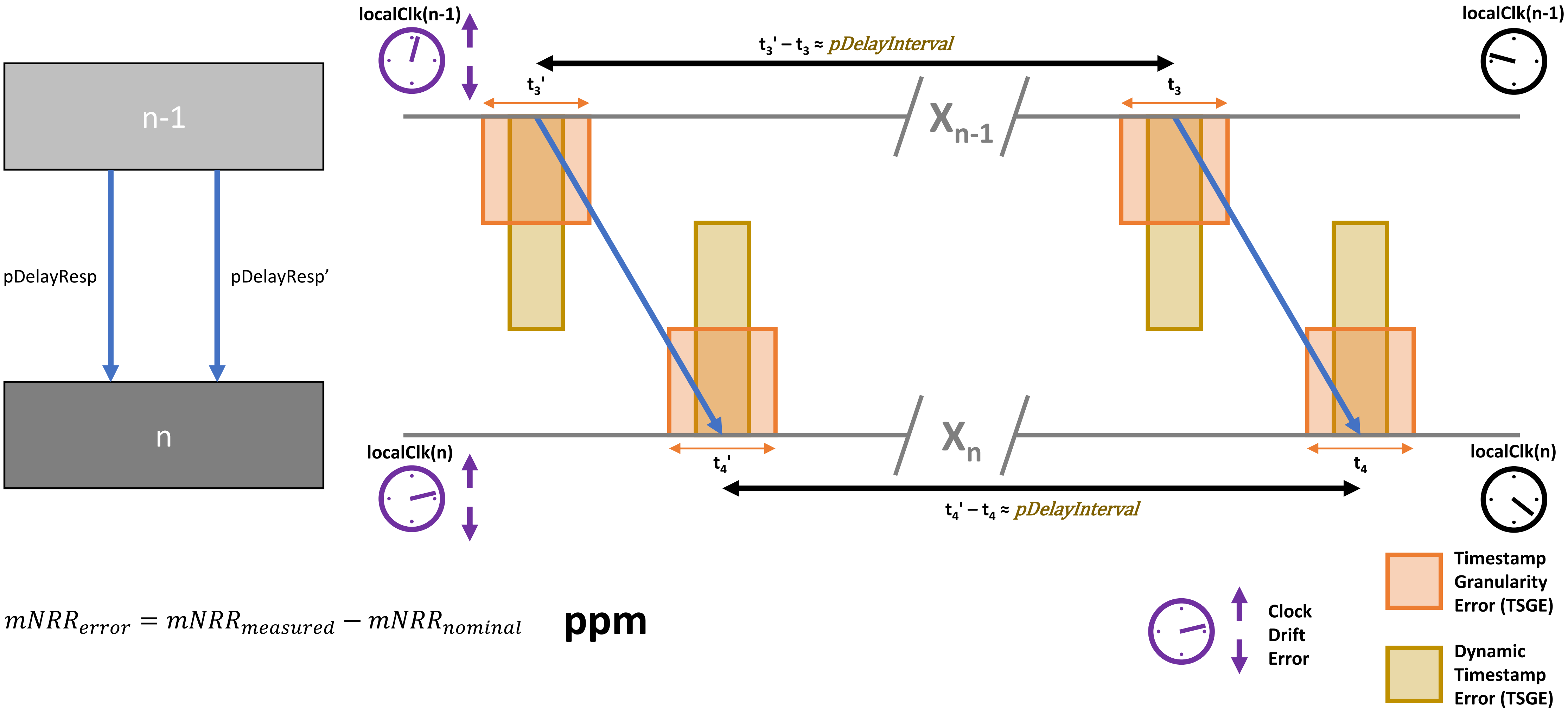
5. RR Calc at Receipt of Sync message → Receipt of next Sync Message
   - The relevant drift is between the current node's clock and the GM's clock.
   - It is only modeled at for the last hop.
   - It is modelled as a gamma distribution with shape 270.5532 and rate 270.5532/*syncInterval*. [See IEEE 1588-2019 9.5.9.2 and [7] slides 8-15.]

$$T_{SyncToSync} = \sim\boldsymbol{\Gamma}\left(270.5532, \frac{270.5532}{\boldsymbol{syncInterval}}\right)$$

6. Drift during measurement of meanLinkDelay (not shown above; see Equations section)
   - The relevant drift is between the current node's clock and the upstream node's clock.
   - It is modeled as equal to the *pDelayTurnaround* parameter.

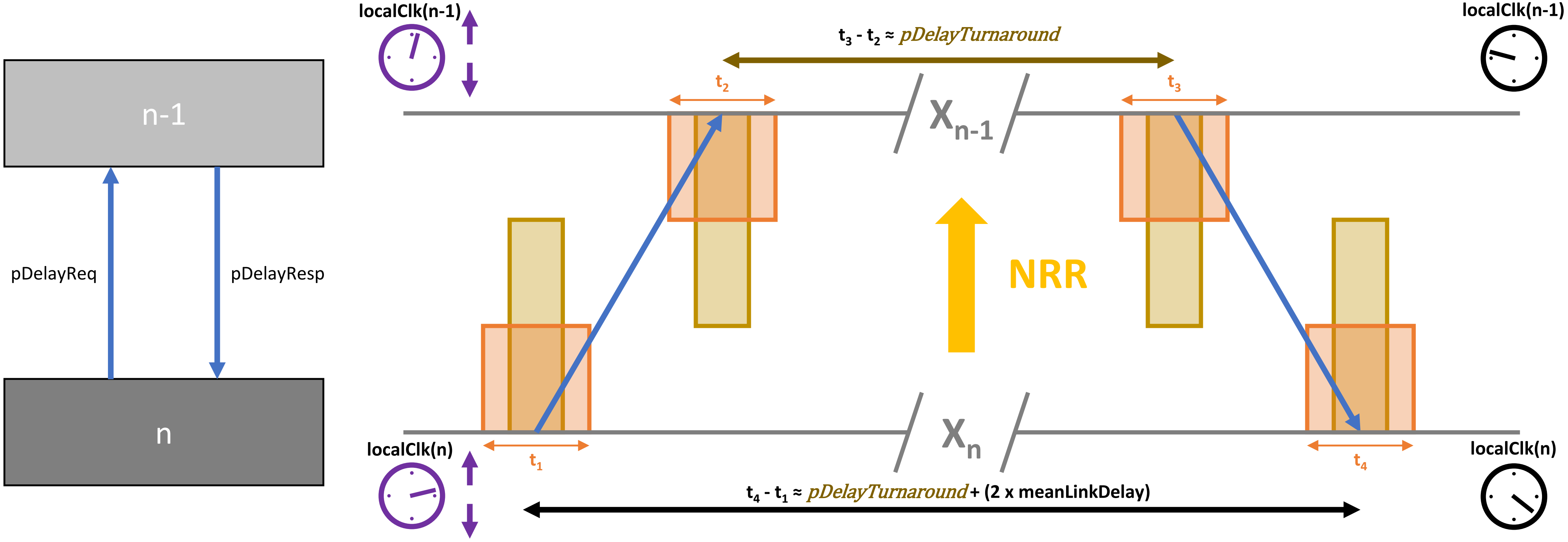
Equations

Errors Measuring NRR

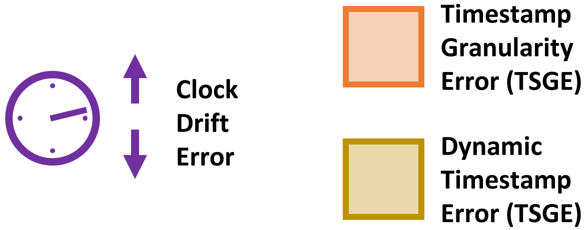


$$mNRR_{error} = mNRR_{measured} - mNRR_{nominal}$$ **ppm**

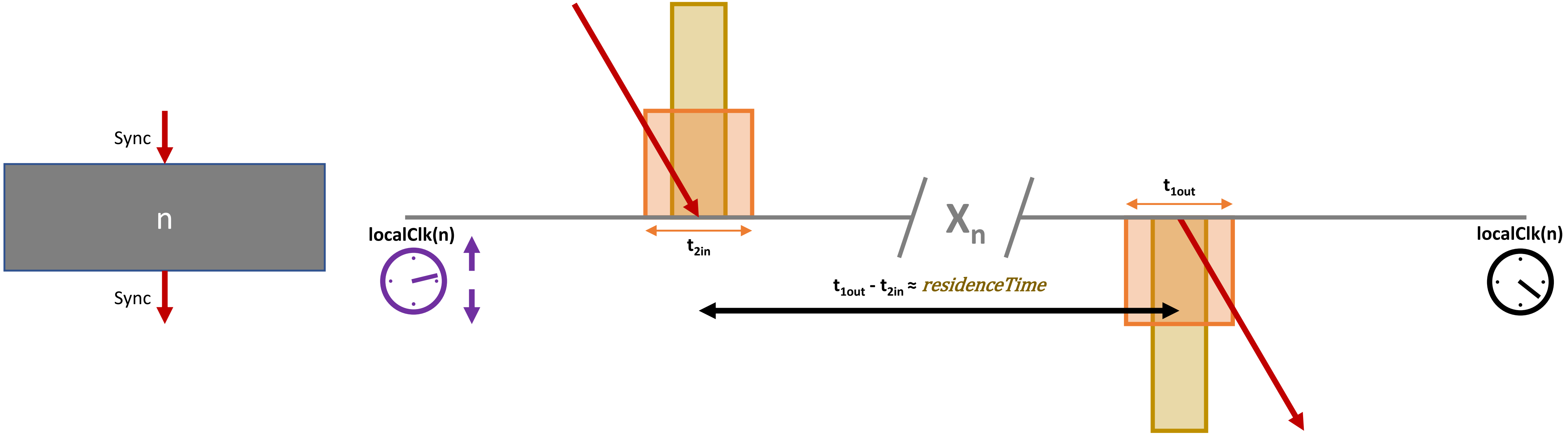
meanLinkDelay Errors



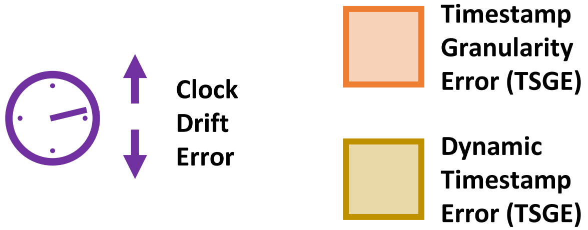
$$meanLinkDelay_{error} = meanLinkDelay_{measured} - meanLinkDelay_{nominal}$$ **ns**



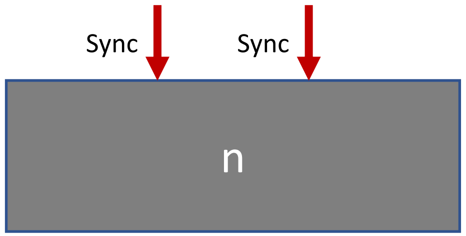
Residence Time

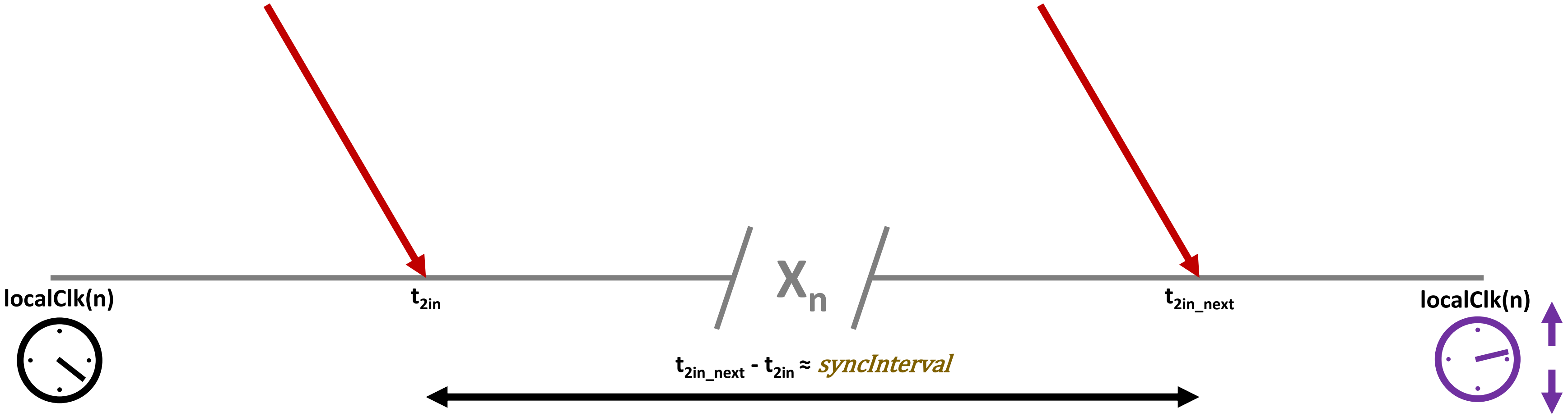


$$residenceTime_{error} = residenceTime_{measured} - residenceTime_{nominal}$$ **ns**

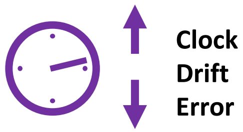


End Station Error

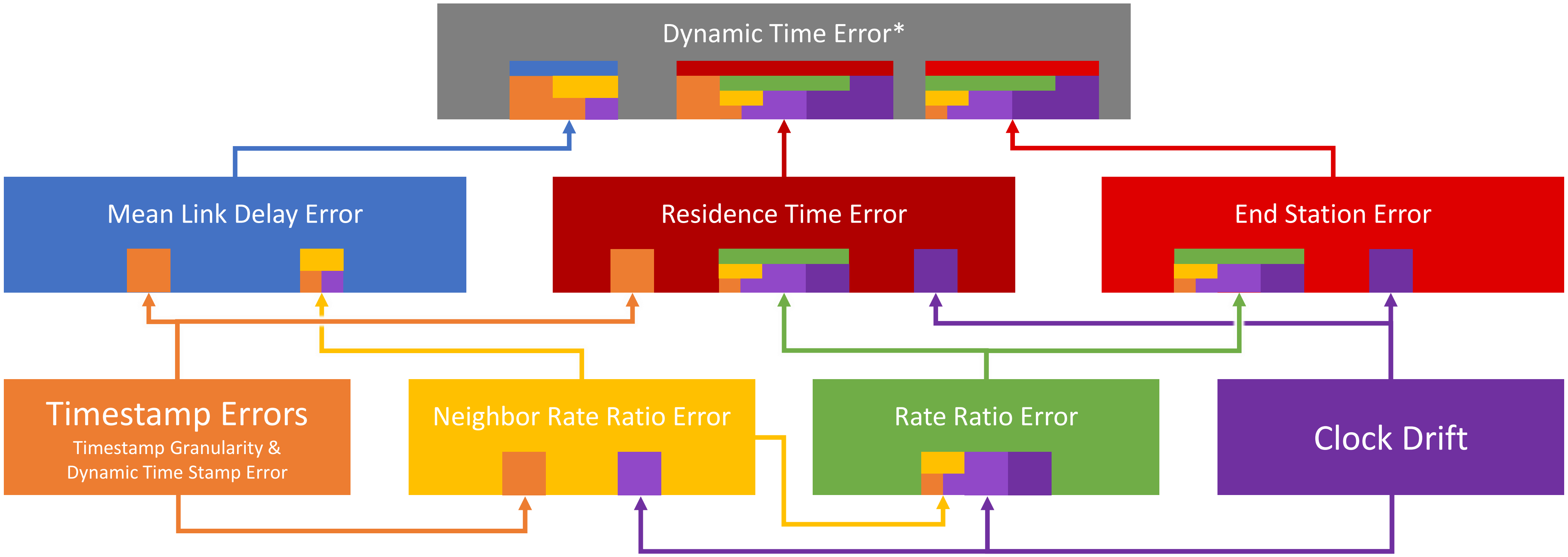




$$endStation_{error} = endStation_{measured} - endStationnomi_{nal}$$ **ns**



Dynamic Time Sync Error Accumulation



All errors in this analysis are caused by either Clock Drift or Timestamp Errors

*DTE based on protocol messaging only. Total DTE at the application level will also depend on ClockMaster, ClockSlave, ClockSource, Clock Target, etc...

Equations – Timestamp Errors

- Eight timestamp errors are generated for each node in each run...
  - $t_{1pderror}$
  - $t_{2pderror}$
  - $t_{3pderror}$
  - $t_{4pderror}$
  - $t_{3pderror}'$
  - $t_{4pderror}'$
  - $t_{2sinerror}$
  - $t_{1souterror}$
- *pd* errors are associated pDelay and pDelayResponse messages used to measure meanLinkDelay. $t_{3pderror}'$ and $t_{4pderror}'$ errors are associated with the previous pDelayResponse message used – along with the most recent one, to measure NRR.
- *sin* and *sout* errors are associated with receiving and transmissing Sync messages.
- All timestamp errors use the same equation. For example...

$$t_{1pderror} = Error_{TSGETX} + Error_{DTSETX}$$

- Each timestamp error is uncorrelated to any other timestamp error.
- See previous section for definition of $Error_{TSGE}$ and $Error_{DTSE}$.

Equations – Clock Drift

- The model tracks the drift of three clocks at each node...
  - $clockDrift_{GM}$ – Clock Drift of the Grandmaster clock
  - $clockDrift_{n}$ – Clock Drift of the current node's clock
  - $clockDrift_{n-1}$ – Clock Drift of the upstream node's clock
- See previous section for details on the equations to generate the clockDrift values
- $clockDrift_{GM}$ is generated once for each run. For the first hop, it is equivalent to $clockDrift_{n-1}$, i.e. for the hop, the "upstream node" is the GM
- For all nodes after the first, $clockDrift_{n}$ is copied to $clockDrift_{n-1}$ before a new value for $clockDrift_{n}$ is generated.
  - Note: the actual implementation uses two vectors and swaps their function between n and n-1 to reduce processing time by eliminating the need to copy data.

Equations – $\mathrm{mNRR}_{\mathrm{error}}$ (Neighbor Rate Ratio) Timestamp Error – 1

$$mNRR_{error} = mNRR_{errorTS} + mNRR_{errorCD}$$ **ppm**

$$mNRR_{errorTS} = mNRR_{measuredTSerror} - mNRR_{nominal}$$ **ppm**

$$= \left(\left(\frac{\left((t_3 + \boldsymbol{t_{3pderror}}) - (t_3' + \boldsymbol{t'_{3pderror}})\right)}{\left((t_4 + \boldsymbol{t_{4pderror}}) - (t_4' + \boldsymbol{t'_{4pderror}})\right)}\right) - 1\right) \times 10^6 - \left(\left(\frac{t_3 - t_3'}{t_4 - t_4'}\right) - 1\right) \times 10^6$$

$$= \left(\left(\frac{t_3 - t_3' + \boldsymbol{t_{3pderror}} - \boldsymbol{t'_{3pderror}}}{t_4 - t_4' + \boldsymbol{t_{4pderror}} - \boldsymbol{t'_{4pderror}}}\right) - 1\right) \times 10^6 - \left(\left(\frac{t_3 - t_3'}{t_4 - t_4'}\right) - 1\right) \times 10^6$$

$$= \left(\left(\frac{t_3 - t_3'}{t_4 - t_4' + \boldsymbol{t_{4pderror}} - \boldsymbol{t'_{4pderror}}}\right) - 1 + \left(\frac{\boldsymbol{t_{3pderror}} - \boldsymbol{t'_{3pderror}}}{t_4 - t_4' + \boldsymbol{t_{4pderror}} - \boldsymbol{t'_{4pderror}}}\right)\right) \times 10^6 - \left(\left(\frac{t_3 - t_3'}{t_4 - t_4'}\right) - 1\right) \times 10^6$$

The ratio X in ppm is (X – 1) x 10^6.
$mNRR_{errorTS}$ and $mNRR_{errorCD}$ are not entirely independent, but the effect of the relationship on $mNRR_{error}$ can be ignored provided the errors are small compared to ***pDelayTurnaround***, which they are for pDelayTurnaround of 1ms or more. See backup for algorithmic proof.

Equations – mNRR$_{error}$ – Timestamp Error

$$mNRR_{errorTS} = \left(\left(\frac{t_3 - t'_3}{t_4 - t'_4 + t_{4pderror} - t'_{4pderror}}\right) - 1 + \left(\frac{t_{3pderror} - t'_{3pderror}}{t_4 - t'_4 + t_{4pderror} - t'_{4pderror}}\right)\right) \times 10^6 - \left(\left(\frac{t_3 - t'_3}{t_4 - t'_4}\right) - 1\right) \times 10^6$$

ppm

$$= \frac{(t_4 - t_4').(t_{3pderror} - t'_{3pderror}) - (t_3 - t'_3).(t_{4pderror} - t'_{4pderror})}{(t_4 - t'_4).\left((t_4 - t'_4) + t_{4pderror} - t'_{4pderror}\right)} \times 10^6$$

$$\approx \frac{T_{pdelay2pdelay} \times 10^6.(t_{3pderror} - t'_{3pderror}) - T_{pdelay2pdelay} \times 10^6.(t_{4pderror} - t'_{4pderror})}{T_{pdelay2pdelay} \times 10^6.(T_{pdelay2pdelay} \times 10^6 + t_{4pderror} - t'_{4pderror})} \times 10^6$$

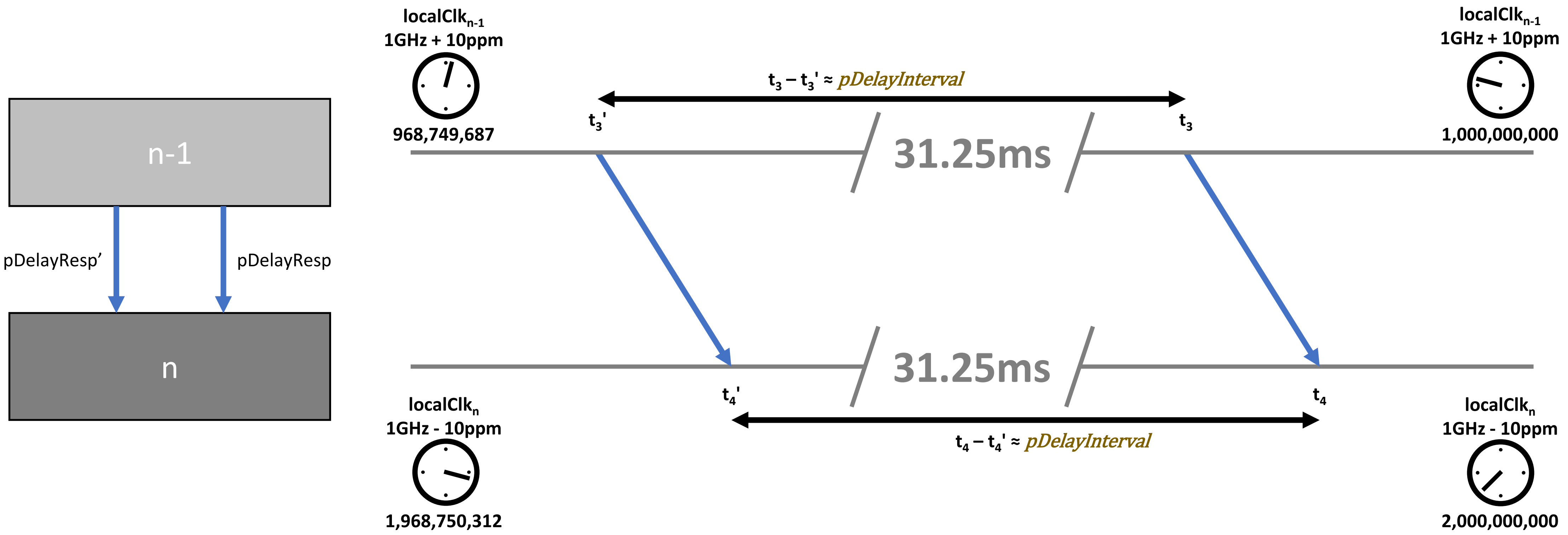
$$= \frac{(t_{3pderror} - t'_{3pderror}) - (t_{4pderror} - t'_{4pderror})}{T_{pdelay2pdelay} \dfrac{t_{4pderror} - t'_{4pderror}}{10^6}}$$

$$\approx \frac{(t_{3pderror} - t'_{3pderror}) - (t_{4pderror} - t'_{4pderror})}{T_{pdelay2pdelay}}$$

The error magnitudes are small relative to the $t_3 - t_3'$ and $t_4 - t_4'$ factors, which are both nominally $T_{pdelay2pdelay}$ (which is in ms, whereas the timestamps are in nanoseconds, hence $T_{pdelay2pdelay}$ x10^6).

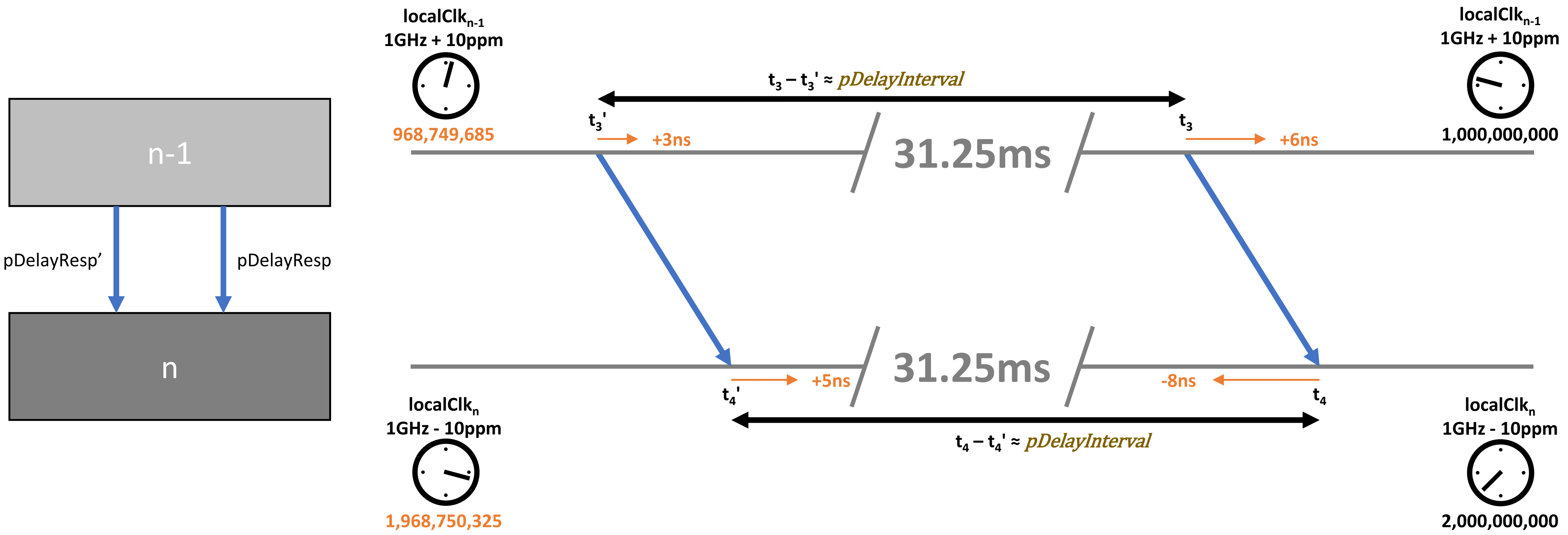
$t_{4pderror} - t_{4pderror}'$ divided by 10^6 on the lower line is small enough relative to $T_{pdelay2pdelay}$ to ignore.

$\text{mNRR}_{\text{error}}$ Timestamp Error Example



$$\text{mNRR @ } t_4' = \frac{(t_3 - t_3')}{(t_4 - t_4')} = \frac{31{,}250{,}313}{31{,}249{,}688} \rightarrow 0.002000\% \;=\; 20.00 \text{ ppm}$$

mNRR$_{error}$ Timestamp Error Example



$$\textbf{mNRR @ } t_4' = \frac{(t_3 - t_3')}{(t_4 - t_4')} = \frac{31{,}250{,}315}{31{,}249{,}675} \rightarrow 0.0020512\% = 20.512 \text{ ppm}$$

$mNRR_{error}$ Timestamp Error Example

- With no Timestamp Error...

$$\textbf{mNRR @ } t_4' = \frac{(t_3 - t_3')}{(t_4 - t_4')} = \frac{31{,}250{,}313}{31{,}249{,}688} \rightarrow 0.002000\% = 20.00 \text{ ppm}$$

- With $t_{4pderror}$ = -8 ns, $t_{4pderror}'$ = +5 ns, $t_{3pderror}$ = +6 ns, $t_{3pderror}'$ = +3 ns, $mNRR_{errorCD}$ = 0.512 ppm

$$\textbf{mNRR @ } t_4' = \frac{(t_3 - t_3')}{(t_4 - t_4')} = \frac{31{,}250{,}315}{31{,}249{,}675} \rightarrow 0.0020512\% = 20.512 \text{ ppm}$$

- From $mNRR_{errorTS}$ equation...

$$mNRR_{errorTS} \approx \frac{\left(t_{3pderror} - t_{3pderror}'\right) - \left(t_{4pderror} - t_{4pderror}'\right)}{T_{pdelay2pdelay}} = \frac{(6-3)-(-8-5)}{31.25} = \frac{16}{31.25} = 0.512 \text{ ppm}$$

Equations – mNRR$_{error}$
Errors due to Clock Drift

$mNRR_{error} = mNRR_{errorTS} + mNRR_{errorCD}$ **ppm**

$mNRR_{errorCD} = mNRR_{measuredCDerror} - mNRR_{nominal}$ **ppm**

$$= \left(\left(\frac{t_3 - (t'_3 + t'_{3CDerror})}{t_4 - (t'_3 + t'_{4CDerror})}\right) - 1\right) \times 10^6 - \left(\left(\frac{t_3 - t'_3}{t_4 - t'_3}\right) - 1\right) \times 10^6$$

$$= \left(\left(\frac{t_3 - t'_3 - t'_{3CDerror}}{t_4 - t'_3 - t'_{4CDerror}}\right) - 1\right) \times 10^6 - \left(\left(\frac{t_3 - t'_3}{t_4 - t'_3}\right) - 1\right) \times 10^6$$

$$= \left(\left(\frac{t_3 - t'_3}{t_4 - {t_4}' - t'_{4CDerror}}\right) - 1 + \left(\frac{-t'_{3CDerror}}{t_4 - {t_4}' - t'_{4CDerror}}\right)\right) \times 10^6 - \left(\left(\frac{t_3 - t'_3}{t_4 - {t_4}'}\right) - 1\right) \times 10^6$$

The ratio X in ppm is (X – 1) x 10^6.

Equations – mNRR$_{error}$ Errors due to Clock Drift

$$mNRR_{errorCD} = \left(\left(\frac{t_3 - t_3'}{t_4 - t_4' - t_{4CDerror}'}\right) - 1 + \left(\frac{-t_{3CDerror}'}{t_4 - t_4' - t_{4CDerror}'}\right)\right) \times 10^6 - \left(\left(\frac{t_3 - t_3'}{t_4 - t_4'}\right) - 1\right) \times 10^6$$ **ppm**

$$= \frac{(t_4 - t_4')(-t_{3CDerror}') - (t_3 - t_3')(-t_{4CDerror}')}{(t_4 - t_4').\left((t_4 - t_4') - t_{4CDerror}'\right)} \times 10^6$$ **ppm**

$$\approx \frac{T_{pdelay2pdelay} \times 10^6.(-t_{3CDerror}') - T_{pdelay2pdelay} \times 10^6(-t_{4CDerror}')}{T_{pdelay2pdelay} \times 10^6.\left(T_{pdelay2pdelay} \times 10^6 - t_{4CDerror}'\right)} \times 10^6$$ **ppm**

$$= \frac{t_{4CDerror}' - t_{3CDerror}'}{T_{pdelay2pdelay} \frac{-t_{4CDerror}'}{10^6}}$$ **ppm**

$$\approx \frac{t_{4CDerror}' - t_{3CDerror}'}{T_{pdelay2pdelay}}$$

The error magnitudes are small relative to the $t_3 - t_3'$ and $t_4 - t_4'$ factors, which are both nominally $T_{pdelay2pdelay}$ (which is in ms, whereas the timestamps are in nanoseconds, hence $T_{pdelay2pdelay} \times 10^6$).

$t_{4CDerror}'$ divided by 10^6 on the lower line is small enough relative to $T_{pdelay2pdelay}$ to ignore.

Equations – mNRR$_{error}$ Errors due to Clock Drift

$$mNRR_{errorCD} = \frac{t'_{4CDerror} - t'_{3CDerror}}{T_{pdelay2pdelay}}$$ **ppm**

$$t'_{4CDerror} = t'_{4measuredCDerror} - t'_{4nominal}$$ **ns**

$$= \left(t_4 - T_{pdelay2pdelay} \times 10^6 \left(1 + \frac{clockOffset_n(t_4) - \frac{\boldsymbol{clockDrift_n}}{2} \times \frac{T_{pdelay2pdelay}}{10^3}}{10^6} \right) \right) - \left(t_4 - T_{pdelay2pdelay} \times 10^6 \left(1 + \frac{clockOffset_n(t_4)}{10^6} \right) \right)$$ **ns**

$$= -T_{pdelay2pdelay} \times 10^6 \frac{-\frac{\boldsymbol{clockDrift_n}}{2} \times \frac{T_{pdelay2pdelay}}{10^3}}{10^6}$$ **ns**

$$= \frac{\boldsymbol{clockDrift_n}.T^2_{pdelay2pdelay}}{2 \times 10^3}$$ **ns**

$$t'_{3CDerror} = \frac{\boldsymbol{clockDrift_{n-1}}.T^2_{pdelay2pdelay}}{2 \times 10^3}$$ **ns**

Equations – mNRR$_{error}$ Errors due to Clock Drift

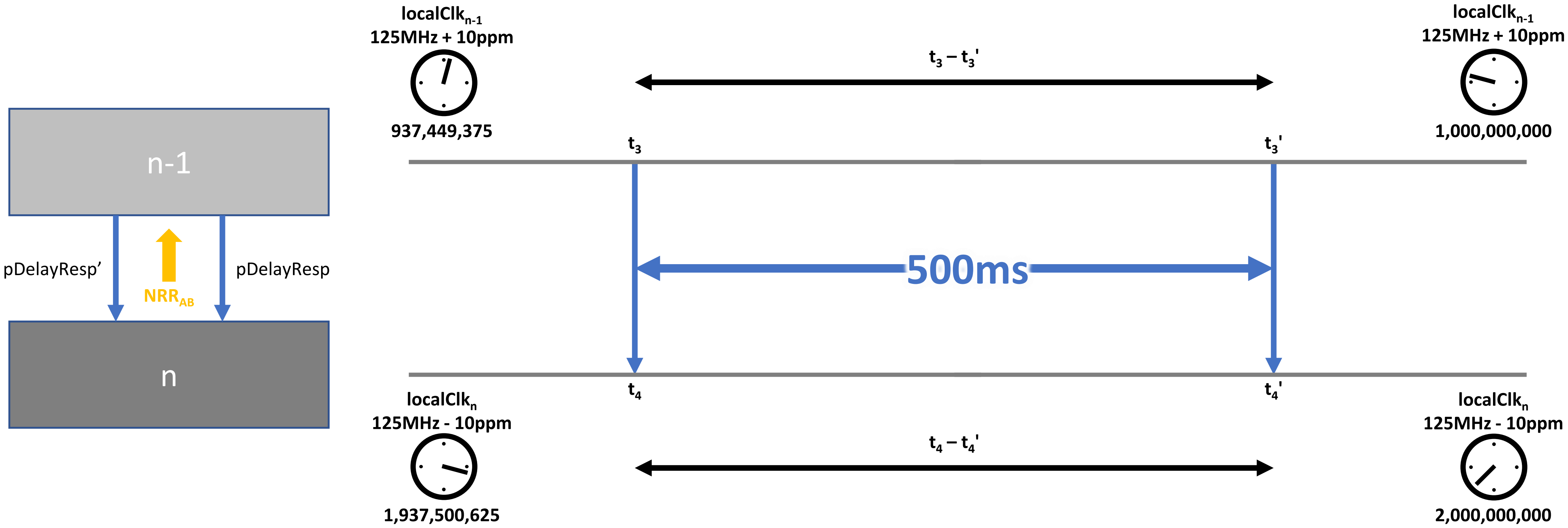
$$mNRR_{errorCD} = \frac{t'_{4CDerror} - t'_{3CDerror}}{T_{pdelay2pdelay}}$$ **ppm**

$$t'_{4CDerror} = \frac{\boldsymbol{clockDrift_n}.T^2_{pdelay2pdelay}}{2 \times 10^3} \qquad t'_{3CDerror} = \frac{\boldsymbol{clockDrift_{n-1}}.T^2_{pdelay2pdelay}}{2 \times 10^3}$$ **ns**

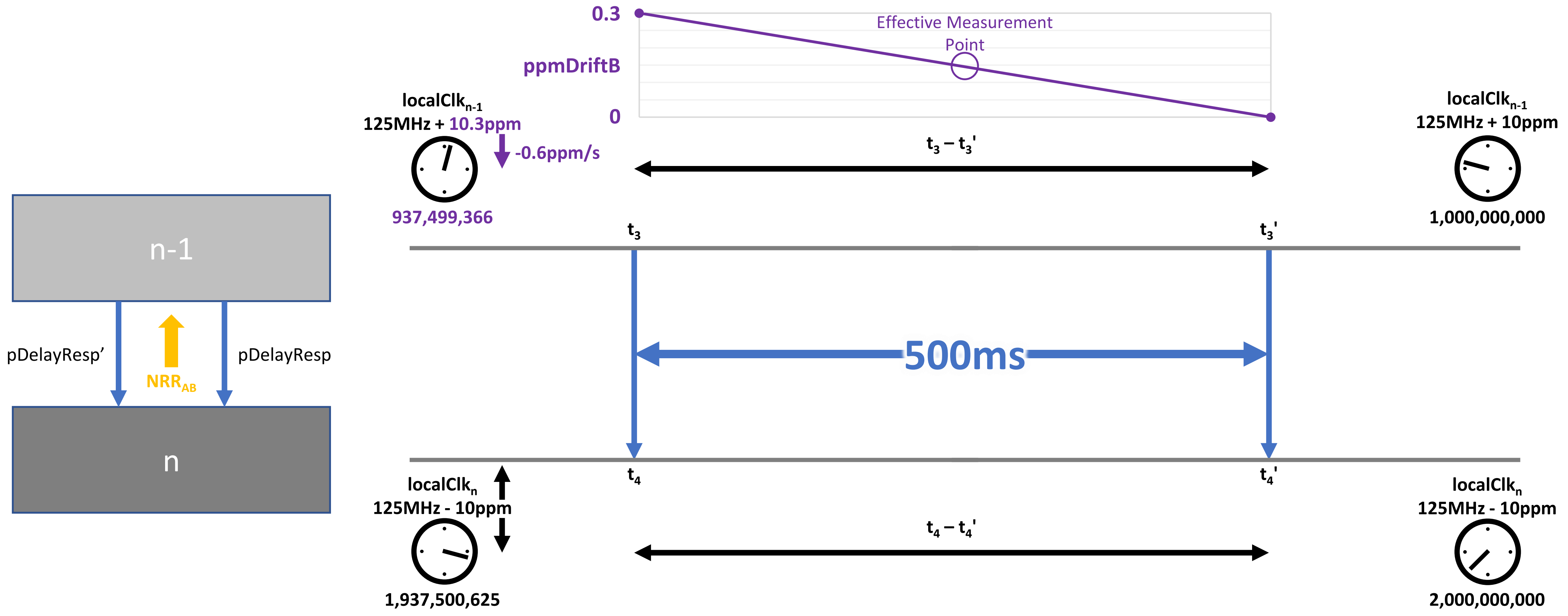
$$mNRR_{errorCD} = \frac{\boldsymbol{clockDrift_{n-1}}.T^2_{pdelay2pdelay} - \boldsymbol{clockDrift_n}.T^2_{pdelay2pdelay}}{2 \times 10^3 \times T_{pdelay2pdelay}}$$ **ppm**

$$= \frac{T_{pdelay2pdelay}(\boldsymbol{clockDrift_n} - \boldsymbol{clockDrift_{n-1}})}{2 \times 10^3}$$ **ppm**

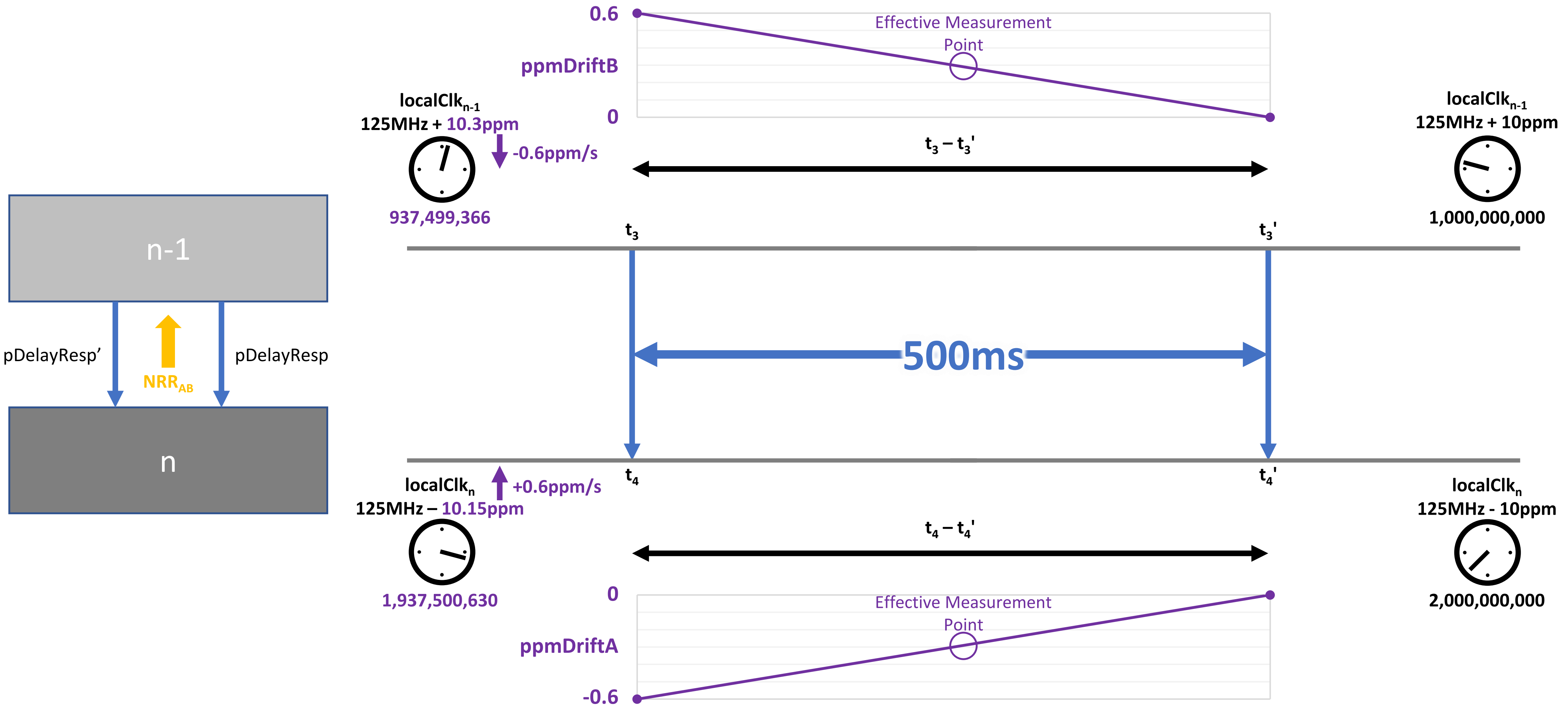
$mNRR_{error}$ Clock Drift Example



$$mNRR\ @\ t_4' = \frac{(t_3 - t_3')}{(t_4 - t_4')} = \frac{62{,}500{,}675}{62{,}499{,}375} \rightarrow 0.0020000\% = 20.000\ \text{ppm}$$



$$\text{mNRR @ } t_4' = \frac{(t_3 - t_3')}{(t_4 - t_4')} = \frac{62{,}500{,}634}{62{,}499{,}375} \rightarrow 0.0020150\% \;=\; 20.150 \text{ ppm}$$



$$\text{mNRR @ } t_4' = \frac{(t_3 - t_3')}{(t_4 - t_4')} = \frac{62{,}500{,}634}{62{,}499{,}370} \rightarrow 0.0020225\% = 20.225 \text{ ppm}$$

$\text{mNRR}_{\text{error}}$ Clock Drift Example

- With no Clock Drift...

$$\textbf{mNRR @ } t_4' = \frac{(t_3 - t_3')}{(t_4 - t_4')} = \frac{62{,}500{,}675}{62{,}499{,}375} \rightarrow \mathbf{0.0020000\%} = \mathbf{20.000\ ppm}$$

- With *clockDrift*$_n$ = 0.3 ppm/s and *clockDrift*$_{n-1}$ = -0.6 ppm/s, $mNRR_{errorCD}$ = 0.225ppm

$$\textbf{mNRR @ } t_4' = \frac{(t_3 - t_3')}{(t_4 - t_4')} = \frac{62{,}499{,}370}{62{,}500{,}634} \rightarrow \mathbf{-0.0020225\%} = \mathbf{20.225\ ppm}$$

- From $mNRR_{errorCD}$ equation...

$$mNRR_{errorCD} = \frac{T_{pdelay2pdelay}(\boldsymbol{clockDrift_n} - \boldsymbol{clockDrift_{n-1}})}{2 \times 10^3} = \frac{500(0.3 + 0.6)}{2 \times 10^3} = \frac{500(0.9)}{2 \times 10^3} = \frac{450}{2 \times 10^3} = 0.225\ ppm$$

Note on Algorithmic Equivalence

- Both the above derivations utilise the following equivalence...

$$c\left(\frac{a}{b+y}-1+\frac{x}{b+y}\right)-c\left(\frac{a}{b}-1\right)=c\,\frac{bx-ay}{b(b+y)}$$

- The detailed steps are as follows...

$$c\left(\frac{a}{b+y}-1+\frac{x}{b+y}\right)-c\left(\frac{a}{b}-1\right)$$

$$=c\left(\frac{a}{b+y}+\frac{-b-y}{b+y}+\frac{x}{b+y}\right)-c\left(\frac{a}{b}+\frac{-b}{b}\right)$$

$$=c\,\frac{a-b-y+x}{b+y}-c\,\frac{a-b}{b}$$

$$=c\,\frac{ab-b^2-by+bx}{b(b+y)}-c\,\frac{a(b+y)-b(b+y)}{b(b+y)}$$

$$=c\,\frac{ab-b^2-by+bx}{b(b+y)}-c\,\frac{ab+ay-b^2-by}{b(b+y)}$$

$$=c\,\frac{ab-b^2-by+bx-ab-ay+b^2+by}{b(b+y)}$$

$$=c\,\frac{bx-ay}{b(b+y)}$$

Equations – RR_{error} – 1

- RR is calculated via an accumulation of NRRs. At each node RR_{error} is the sum of...
  - RR_{error} at the upstream node
  - $mNRR_{error}$
  - Error due to Clock Drift between mNRR calculation and **transmission** of Sync
    - Or, for the last node only, **reception** of Sync
  - Error due to Clock Drift between RR calculation at upstream node and transmission of Sync

Equations – $\mathrm{RR}_{\mathrm{error}}$ – 2

$$RR_{errror}(n) = RR_{error}(n-1) + mNRR_{error} + RR_{errorCD\_NRR2sync} + RR_{errorCD\_RR2sync}$$ **ppm**

$$RR_{errrorCD\_NRR2sync} = \frac{T_{mNRR2sync}(\boldsymbol{clockDrift_{n}} - \boldsymbol{clockDrift_{n-1}})}{10^3}$$ **ppm**

$$T_{mNRR2sync} = \sim U(\boldsymbol{pdelayInterval}.\,0.9, \boldsymbol{pdelayInterval}.\,1.3\,) \times \sim U(0,1)$$ **ppm**

$$RR_{errrorCD\_RR2sync} = \frac{\boldsymbol{residenceTime}(\boldsymbol{clockDrift_{n-1}} - \boldsymbol{clockDrift_{GM}})}{10^3}$$ **ppm**

At the final hop there is no Residence Time, so $RR_{errorCD\_RR2Sync}$ is zero.

Equations – MLD$_{error}$ (Mean Link Delay) – 1

$MLD_{errror} = MLD_{errorTSdirect} + MLD_{errorNRR}$ **ns**

$MLD_{errorTSdirect} = MLD_{measuredTSerror} - MLD_{nominal}$ **ns**

$$= \frac{\left((t_4 + t_{4pderror}) - (t_1 + t_{1pderror})\right) - \left((t_3 + t_{3pderror}) - (t_2 + t_{2pderror})\right)\left(1 - \frac{mNRR}{10^6}\right)}{2} - \frac{(t_4 - t_1) - (t_3 - t_2)\left(1 - \frac{mNRR}{10^6}\right)}{2}$$

$$= \frac{(t_{4pderror} - t_{1pderror}) - (t_{3pderror} - t_{2pderror})\left(1 - \frac{mNRR}{10^6}\right)}{2}$$

$$\approx \frac{(t_{4pderror} - t_{1pderror}) - (t_{3pderror} - t_{2pderror})}{2}$$

Equations – MLD_{error} (Mean Link Delay) – 2

$$MLD_{errror} = MLD_{errorTSdirect} + MLD_{errorNRR}$$ **ns**

$$MLD_{errorNRR} = MLD_{measuredNRRerror} - MLD_{nominal}$$ **ns**

$$= \frac{(t_4 - t_1) - (t_3 - t_2)\left(1 - \frac{mNRR + mNRR_{error}}{10^6}\right)}{2} - \frac{(t_4 - t_1) - (t_3 - t_2)\left(1 - \frac{mNRR}{10^6}\right)}{2}$$

$$= \frac{(t_4 - (t_4 - \boldsymbol{pDelayTurnaround} \times 10^6 - 2.meanLinkDelay) - \left(t3 - \left(t_3 - \boldsymbol{pDelayTurnaround} \times 10^6 \left(1 - \frac{mNRR + mNRR_{error}}{10^6}\right)\right)\right)}{2}$$

$$- \frac{(t_4 - (t_4 - \boldsymbol{pDelayTurnaround} \times 10^6 - 2.meanLinkDelay) - \left(t3 - \left(t_3 - \boldsymbol{pDelayTurnaround} \times 10^6 \left(1 - \frac{mNRR}{10^6}\right)\right)\right)}{2}$$

$$= -\frac{\boldsymbol{pDelayTurnaround}.mNRR_{error}}{2}$$

Equations – RT_{error} (Residence Time) – 1

$$RT_{errror} = RT_{errorTSdirect} + RT_{errorRR} + RT_{errorCDdirect}$$ **ns**

$$RT_{errorTSdirect} = \boldsymbol{t_{1souterror}} - \boldsymbol{t_{2sinerror}}$$ **ns**

$$RT_{errorRR} = RT_{measuredRRerror} - RT_{nominal}$$ **ns**

$$= (t_{1soutNominal} - t_{2sinMeasured}) - (t_{1soutNominal} - t_{2sinNominal})$$

$$= -\left(t_{1soutNominal} - \boldsymbol{residenceTime} \times 10^6\left(1 + \frac{freqOffset}{10^6}\right)\left(1 + \frac{RR + RR_{error}}{10^6}\right)\right)$$

$$+\left(t_{1soutNominal} - \boldsymbol{residenceTime} \times 10^6\left(1 + \frac{freqOffset}{10^6}\right)\left(1 + \frac{RR}{10^6}\right)\right)$$

$$= -\left(-\boldsymbol{residenceTime}.RR_{error}\left(1 + \frac{freqOffset}{10^6}\right)\right)$$

$$= \boldsymbol{residenceTime}.RR_{error} + \frac{\boldsymbol{residenceTime}.RR_{error}.freqOffset}{10^6}$$

$$\approx \boldsymbol{residenceTime} \times RR_{error}$$

Equations – RT_{error} (Residence Time) – 2

$$RT_{errror} = RT_{errorTSdirect} + RT_{errorRR} + RT_{errorCDdirect}$$ **ns**

$$RT_{errorTSdirect} = t_{1souterror} - t_{2sinerror}$$

$$RT_{errorRR} = residenceTime \times RR_{error}$$ **ns**

$$RT_{errorCDdirect} = RT_{measuredCDerror} - RT_{nominal}$$ **ns**

$$= (t_{1soutNominal} - t_{2sinMeasured})\left(1 + \frac{RR}{10^6}\right) - (t_{1soutNominal} - t_{2sinNominal})\left(1 + \frac{RR}{10^6}\right)$$

$$= -\left(t_{1soutNominal} - residenceTime \times 10^6\left(1 + \frac{clockOffset_n(t_{1out}) + \frac{clockDrift_n}{2} \times \frac{residenceTime}{10^3}}{10^6}\right)\right)\left(1 + \frac{RR}{10^6}\right)$$

$$+ \left(t_{1soutNominal} - residenceTime \times 10^6\left(1 + \frac{clockOffset_n(t_{1out}) + \frac{clockDrift_{GM}}{2} \times \frac{residenceTime}{10^3}}{10^6}\right)\right)\left(1 + \frac{RR}{10^6}\right)$$

$$= -\left(-residenceTime \times 10^6 \frac{\frac{clockDrift_n}{2} \times \frac{residenceTime}{10^3}}{10^6}\right)\left(1 + \frac{RR}{10^6}\right) + \left(-residenceTime \times 10^6 \frac{\frac{clockDrift_{GM}}{2} \times \frac{residenceTime}{10^3}}{10^6}\right)\left(1 + \frac{RR}{10^6}\right)$$

$$= \frac{residenceTime^2(clockDrift_n - clockDrift_{GM})}{2 \times 10^3} + \frac{residenceTime^2.RR(clockDrift_n - clockDrift_{GM})}{2 \times 10^9}$$

$$\approx \frac{residenceTime^2(clockDrift_n - clockDrift_{GM})}{2 \times 10^3}$$

Equations – ES_{error} (End Station) – 1

$$ES_{error} = ES_{errorRR} + ES_{errorCDdirect}$$ **ns**

$$ES_{errorRR} = ES_{actualRRerror} - ES_{nominal}$$ **ns**

$$= T_{sync2sync} \times 10^6 \left(1 + \frac{freqOffset}{10^6}\right)\left(1 + \frac{RR + RR_{error}}{10^6}\right) - T_{sync2sync} \times 10^6 \left(1 + \frac{freqOffset}{10^6}\right)\left(1 + \frac{RR}{10^6}\right)$$

$$= T_{sync2sync} \times 10^6 \left(1 + \frac{freqOffset}{10^6}\right)\left(1 + \frac{RR}{10^6}\right) + T_{sync2sync} \times 10^6 \left(1 + \frac{freqOffset}{10^6}\right)\left(\frac{RR_{error}}{10^6}\right) - T_{sync2sync} \times 10^6$$

$$= T_{sync2sync}.RR_{error} + \frac{T_{sync2sync}.RR_{error}.freqOffset}{10^6}$$

$$\approx T_{sync2sync}.RR_{error}$$

$$T_{SyncToSync} = \sim\Gamma\left(270.5532, \frac{270.5532}{syncInterval}\right)$$ **ns**

The error associated with $freqOffset$ is not modelled as it is orders of magnitude smaller than the main error.

Equations – ES_{error} (End Station) – 2

$$ES_{errror} = ES_{errorRR} + ES_{errorCDdirect}$$ **ns**

$$ES_{errorCDdirect} = ES_{actualCDerror} - ES_{nominal}$$ **ns**

$$= T_{sync2sync} \times 10^6 \left(1 + \frac{freqOffset + \frac{\boldsymbol{clockDrift_n}}{2} \times \frac{T_{sync2sync}}{10^3}}{10^6}\right)\left(1 + \frac{RR}{10^6}\right) - T_{sync2sync} \times 10^6 \left(1 + \frac{freqOffset + \frac{\boldsymbol{clockDrift_{GM}}}{2} \times \frac{T_{sync2sync}}{10^3}}{10^6}\right)\left(1 + \frac{RR}{10^6}\right)$$

$$= T_{sync2sync}\left(\frac{\boldsymbol{clockDrift_n} - \boldsymbol{clockDrift_{GM}}}{2} \times \frac{T_{sync2sync}}{10^3}\right)\left(1 + \frac{RR}{10^6}\right)$$

$$= \frac{{T_{sync2sync}}^2(\boldsymbol{clockDrift_n} - \boldsymbol{clockDrift_{GM}})}{2 \times 10^3}\left(1 + \frac{RR}{10^6}\right)$$

$$= \frac{{T_{sync2sync}}^2(\boldsymbol{clockDrift_n} - \boldsymbol{clockDrift_{GM}})}{2 \times 10^3} + \frac{RR.{T_{sync2sync}}^2(\boldsymbol{clockDrift_n} - \boldsymbol{clockDrift_{GM}})}{2 \times 10^9}$$

$$\approx \frac{{T_{sync2sync}}^2(\boldsymbol{clockDrift_n} - \boldsymbol{clockDrift_{GM}})}{2 \times 10^3}$$

The error associated with RR is not modelled as it is orders of magnitude smaller than the main error.

Equations – DTE

- At all hops other than the last...

$$DTE(n) = DTE(n-1) + MLD_{error} + RT_{error}$$ **ns**

- At the last hop...

$$DTE(n) = DTE(n-1) + MLD_{error} + ES_{error}$$ **ns**

Error Contribution Tracking & Graphical Representations

Error Contribution Tracking

- As well as calculating the primary errors required to calculate DTE, the model also tracks the components of each error and how they accumulate.
- This makes is relatively simple to answer questions such as “What is the probability distribution of DTE due to the Timestamp Error related component of Neighbor Rate Ratio?”
- It also enables a graphical representation of how DTE breaks down into it’s contributing components.
- This section describes the equations used to track the components and the graphical representations.

Error Contribution Tracking – Implementation

- For each component or a primary error, the contribution of the component at each hop is calculated. The vector (across all runs) for this is given the suffix $\_X$.
- For each primary error and component, the vector representing the running total of all errors of this type (across all runs) up to and including the current hop, is calculated and given the suffix $\_SUM$
- From the SUM vectors, the following statistical values are calculated and stored in a single vector per primary error or component (one value for each hop representing the statistic across all runs)...
  - Maximum Absolute Value ($\_\mathrm{MAXabs}$)
  - Mean ($\_\mathrm{MEAN}$)
  - Sigma ($\_\mathrm{SIGMA}$), assuming a gaussian distribution (which is not always valid)
- The _X and _SUM vectors are not preserved, other than at the last hop.
- There are 3 exceptions...
  - $\mathrm{mNRR}_{\mathrm{error}}$ does not accumulate, so the SUM values are not calculated, although MAXabs, MEAN and SIGMA values are (based on the $\_X$ error vectors at each hop).
  - RR_{error} is an accumulation of $mNRR$, so $\_X$ values do not exist. (They do, however, exist for some Clock Drift related error components that don't accumulate.)
  - At the last node, Residence Time Error ($\mathrm{RT}_{\mathrm{error}}$) is effectively replaced by End Station Error ($\mathrm{ES}_{\mathrm{error}}$) which is only calculated at the final hop. Combined $\mathrm{RTES}_{\mathrm{error}}$ error vectors track SUM, MAXabs, MEAN and SIGMA (representing the sum of RT errors up to the last-but-one hop, then the sum of RT errors plus the ES errors at the last hop). The SUM vectors for RT at the last-but-one hop are preserved, and the vectors for ES errors at the last hop are also available as well as the combined RTES error vectors (to enable statistical analysis).

Error & Error Components – Naming

Item	Examples using Residence Time	Residence Time errors due to...
Element	RT	Residence Time (no errors); other elements are mNRR, RR, MLD & ES.
Error in that Element	RT_{error}	All underlying components
Component of that error due to another Element	$RT_{errorTS}$ $RT_{errorCD}$ $RT_{errorTSdirect}$ $RT_{errorCDdirect}$ $RT_{errorRR}$	All Timestamp components All Clock Drift components Direct Timestamp components Direct Clock Drift components Rate Ratio Error
Next level down... Component of a component	$RT_{errorRR\_TS}$ $RT_{errorRR\_CDdirect}$ $RT_{errorRR\_NRR}$	All Timestamp components of Rate Ratio Error Direct Clock Drift component of Rate Ratio Error NRR error component of Rate Ratio Error
Next level down... Component of a component of a component	$RT_{errorRR\_NRR\_TS}$ $RT_{errorRR\_NRR\_CD}$	Timestamp component of NRR via Rate Ratio Error Clock Drift component of NRR via Rate Ratio Error

Equations - $\text{mNRR}_{\text{error}}$

Primary Errors	$mNRR_{errorTS\_X} = \frac{(t_{3pderror} - t_{3pderror}') - (t_{4pderror} - t_{4pderror}')}{T_{pdelay2pdelay}}$	**ppm**
	$mNRR_{errorCD\_X} = \frac{T_{pdelay2pdelay}(clockDrift_n - clockDrift_{n-1})}{2 \times 10^3}$	**ppm**
	$mNRR_{error\_X} = mNRR_{errorTS\_X} + mNRR_{errorCD\_X}$	**ppm**
Error Components	$\text{mNRR}_{\text{error}}$ breaks down into a Timestamp Error and an error due to Clock Drift, so there are no additional error components or calculations.	

Equations - RR_{error}

Primary Errors	$RR_{errorCD\_NRR2Sync\_X} = \frac{T_{mNRR2Sync}(\boldsymbol{clockDrift_n} - \boldsymbol{clockDrift_{n-1}})}{10^3}$	**ppm**
	$RR_{errorCD\_RR2Sync\_X} = \frac{\boldsymbol{residenceTime}(\boldsymbol{clockDrift_{n-1}} - \boldsymbol{clockDrift_{GM}})}{10^3}$	**ppm**
	$RR_{error\_SUM}(n) = RR_{error\_SUM}(n-1) + mNRR_{error\_X} + RR_{errorCD\_NRRtoSync\_X} + RR_{errorCD\_RRtoSync\_X}$	**ppm**

Error Components	$RR_{errorNRR\_CD\_SUM}(n) = RR_{errorNRR\_CD\_SUM}(n-1) + mNRR_{errorCD\_X}(n)$	**ppm**
	$RR_{errorCD\_NRR2sync\_SUM}(n) = RR_{errorCD\_NRR2sync\_SUM}(n-1) + RR_{errorCD\_NRR2Sync\_X}(n)$	**ppm**
	$RR_{errorCD\_RR2sync\_SUM}(n) = RR_{errorCD\_RR2sync\_SUM}(n-1) + RR_{errorCD\_RR2Sync\_X}(n)$	**ppm**
	$RR_{errorNRR\_SUM}(n) = RR_{errorNRR\_SUM}(n-1) + mNRR_{error\_X}(n)$	**ppm**
	$RR_{errorTS\_SUM}(n) = RR_{errorTS\_SUM}(n-1) + mNRR_{errorTS\_X}(n)$	**ppm**
	$RR_{errorCD\_SUM}(n) = RR_{errorNRR\_CD\_SUM}(n) + RR_{errorCD\_NRR2sync\_SUM}(n) + RR_{errorCD\_RR2sync\_SUM}(n)$	**ppm**

At the last hop, there is no Residence Time, so $RR_{errorCD\_RR2Sync\_SUM}$ = 0.

Equations – MLD_{error} – Per Hop

Primary Errors

$$MLD_{errrorTSdirect\_X} = \frac{(t_{4pderror} - t_{1pderror}) - (t_{3pderror} - t_{2pderror})}{2}$$ **ns**

$$MLD_{errrorNRR\_X} = -\frac{pDelayTurnaround \, . \, mNRR_{error}}{2}$$ **ns**

$$MLD_{errror\_X} = MLD_{errorTSdirect} + MLD_{errorNRR}$$ **ns**

Error Components

$$MLD_{errrorNRR\_TS\_X} = -\frac{pDelayTurnaround \, . \, mNRR_{error\_TS\_X}}{2}$$ **ns**

$$MLD_{errrorCD\_X} = -\frac{pDelayTurnaround \, . \, mNRR_{error\_CD\_X}}{2}$$ **ns**

$$MLD_{errrorTS\_X} = MLD_{errorTSdirect} + MLD_{errorNRR\_TS\_X}$$ **ns**

Equations – MLD_{error} – Accumulation

Primary Errors	$MLD_{errrorTSdirect\_SUM}(n) = MLD_{errrorTSdirect\_SUM}(n-1) + MLD_{errrorTSdirect\_X}$	**ns**
	$MLD_{errrorNRR\_SUM}(n) = MLD_{errrorNRR\_SUM}(n-1) + MLD_{errrorNRR\_X}$	**ns**
	$MLD_{errror\_SUM}(n) = MLD_{errror\_SUM}(n-1) + MLD_{errror\_X}$	**ns**
Error Components	$MLD_{errrorNRR\_TS\_SUM}(n) = MLD_{errrorNRR\_TS\_SUM}(n-1) + MLD_{errrorNRR\_TS\_X}$	**ns**
	$MLD_{errrorCD\_SUM}(n) = MLD_{errrorCD\_SUM}(n-1) + MLD_{errrorCD\_X}$	**ns**
	$MLD_{errrorTS\_SUM}(n) = MLD_{errrorTS\_SUM}(n-1) + MLD_{errrorTS\_X}$	**ns**

Equations – RT_{error} – Per Hop (Except Last)

Primary Errors	$RT_{errorTSdirect\_X} = t_{1souterror} - t_{2sinerror}$	ns
	$RT_{errrorCDdirect\_X} = \frac{residenceTime^2(clockDrift_n - clockDrift_{GM})}{2 \times 10^3}$	ns
	$RT_{errorRR\_X} = residenceTime \times RR_{error\_SUM}$	ns
	$RT_{error\_X} = RT_{errorTSdirect\_X} + RT_{errorRR\_X} + RT_{errorCDdirect\_X}$	ns

Error Components	$RT_{errrorRR\_NRR\_CD\_X} = residenceTime \times RR_{errorNRR\_CD\_SUM}$	ns
	$RT_{errrorRR\_CD\_NRR2sync\_X} = residenceTime \times RR_{errorCD\_NRR2sync\_SUM}$	ns
	$RT_{errrorRR\_CD\_RR2sync\_X} = residenceTime \times RR_{errorCD\_RR2sync\_SUM}$	ns
	$RT_{errrorRR\_TS\_X} = residenceTime \times RR_{errorTS\_SUM}$	ns
	$RT_{errrorRR\_NRR\_X} = RT_{errrorRR\_NRR\_CD\_X} + RT_{errrorRR\_TS\_X}$	ns
	$RT_{errrorRR\_CD\_X} = RT_{errrorRR\_NRR\_CD\_X} + RT_{errrorRR\_CD\_NRR2sync\_X} + RT_{errrorRR\_CD\_RR2sync\_X}$	ns
	$RT_{errorCD\_X} = RT_{errorCDdirect\_X} + RT_{errrorRR\_CD\_X}$	ns
	$RT_{errorTS\_X} = RT_{errorTSdirect\_X} + RT_{errorRR\_TS\_X}$	ns

Equations – RT_{error} – Accumulation

Primary Errors	$RT_{errorTSdirect\_SUM}(n) = RT_{errorTSdirect\_SUM}(n-1) + RT_{errorTSdirect\_X}$	**ns**
	$RT_{errrorCDdirect\_SUM}(n) = RT_{errrorCDdirect\_SUM}(n-1) + RT_{errrorCDdirect\_X}$	**ns**
	$RT_{errrorRR\_SUM}(n) = RT_{errrorRR\_SUM}(n-1) + RT_{errrorRR\_X}$	**ns**
	$RT_{errror\_SUM}(n) = RT_{errror\_SUM}(n-1) + RT_{errror\_X}$	**ns**
Error Components	$RT_{errrorRR\_NRR\_CD\_SUM}(n) = RT_{errrorRR\_NRR\_CD\_SUM}(n-1) + RT_{errrorRR\_NRR\_CD\_X}$	**ns**
	$RT_{errrorRR\_CD\_NRR2sync\_SUM}(n) = RT_{errrorRR\_CD\_NRR2sync\_SUM}(n-1) + RT_{errrorRR\_CD\_NRR2sync\_X}$	**ns**
	$RT_{errrorRR\_CD\_RR2sync\_SUM}(n) = RT_{errrorRR\_CD\_RR2sync\_SUM}(n-1) + RT_{errrorRR\_CD\_RR2sync\_X}$	**ns**
	$RT_{errrorRR\_TS\_SUM}(n) = RT_{errrorRR\_TS\_SUM}(n-1) + RT_{errrorRR\_TS\_X}$	**ns**
	$RT_{errrorRR\_NRR\_SUM}(n) = RT_{errrorNRR\_SUM}(n-1) + RT_{errrorNRR\_X}$	**ns**
	$RT_{errrorRR\_CD\_SUM}(n) = RT_{errrorRR\_CD\_SUM}(n-1) + RT_{errrorRR\_CD\_X}$	**ns**
	$RT_{errrorCD\_SUM}(n) = RT_{errrorCD\_SUM}(n-1) + RT_{errrorCD\_X}$	**ns**
	$RT_{errrorTS\_SUM}(n) = RT_{errrorTS\_SUM}(n-1) + RT_{errrorTS\_X}$	**ns**

Equations – ES_{error} – Last Hop Only

Primary Errors

$$ES_{errorRR\_X} = T_{sync2sync}.RR_{error\_SUM}$$ ns

$$ES_{errorCDdirect\_X} = \frac{T_{sync2sync}{}^{2}(\boldsymbol{clockDrift_{n}} - \boldsymbol{clockDrift_{GM}})}{2 \times 10^{3}}$$ ns

$$ES_{error\_X} = ES_{errorRR} + ES_{errorCDdirect}$$ ns

Error Components

$$ES_{errorRR\_NRR\_CD\_X} = T_{sync2sync}.RR_{errorNRR\_CD\_SUM}$$ ns

$$ES_{errorRR\_CD\_NRR2sync\_X} = T_{sync2sync}.RR_{errorCD\_NRR2sync\_SUM}$$ ns

$$ES_{errorRR\_CD\_RR2sync\_X} = T_{sync2sync}.RR_{errorCD\_RR2sync\_SUM}$$ ns

$$ES_{errorRR\_CD\_X} = ES_{errorRR\_NRR\_CD\_X} + ES_{errorRR\_CD\_NRR2sync\_X} + ES_{errorRR\_CD\_RR2sync\_X}$$ ns

$$ES_{errorRR\_TS\_X} = T_{sync2sync}.RR_{errorTS\_SUM}$$ ns

$$ES_{errorRR\_NRR\_X} = ES_{errorRR\_TS\_X} + ES_{errorRR\_NRR\_CD\_X}$$ ns

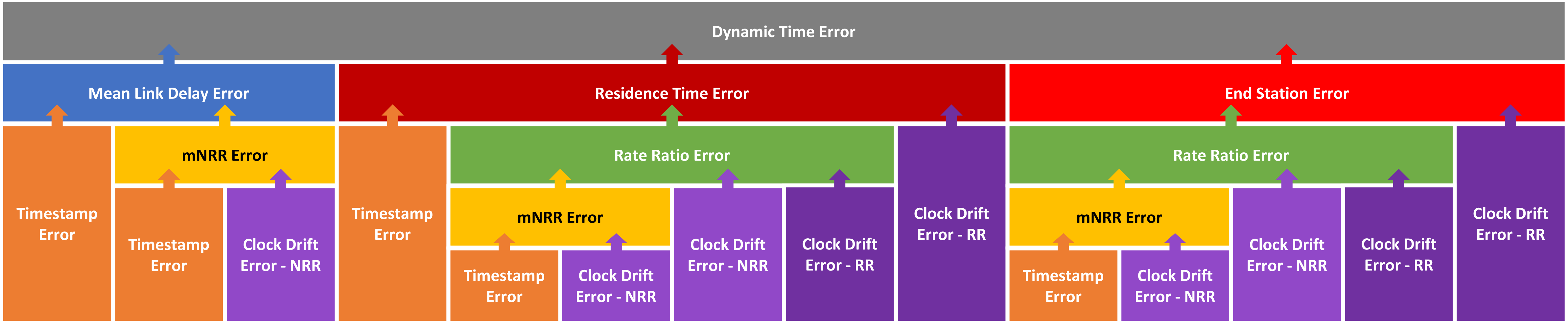
$$ES_{errorCD\_X} = ES_{errorRR\_CD\_X} + ES_{errorCDdirect\_X}$$ ns

Equations – DTE – Per Hop & Accumulation

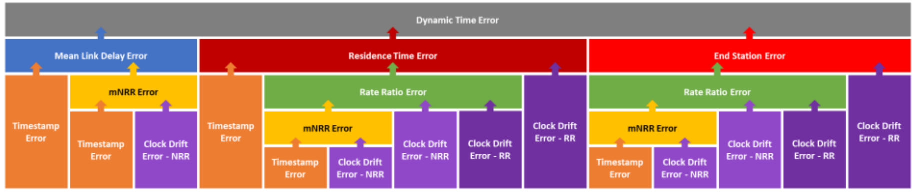
Primary Errors	At all hops other than last:	$DTE_X = MLD_{errror\_X} + RT_{errror\_X}$	**ns**
	At last hop:	$DTE_X = MLD_{error\_X} + ES_{error\_X}$	**ns**
	At all hops:	$DTE_{SUM}(n) = DTE_{SUM}(n-1) + DTE_X$	**ns**

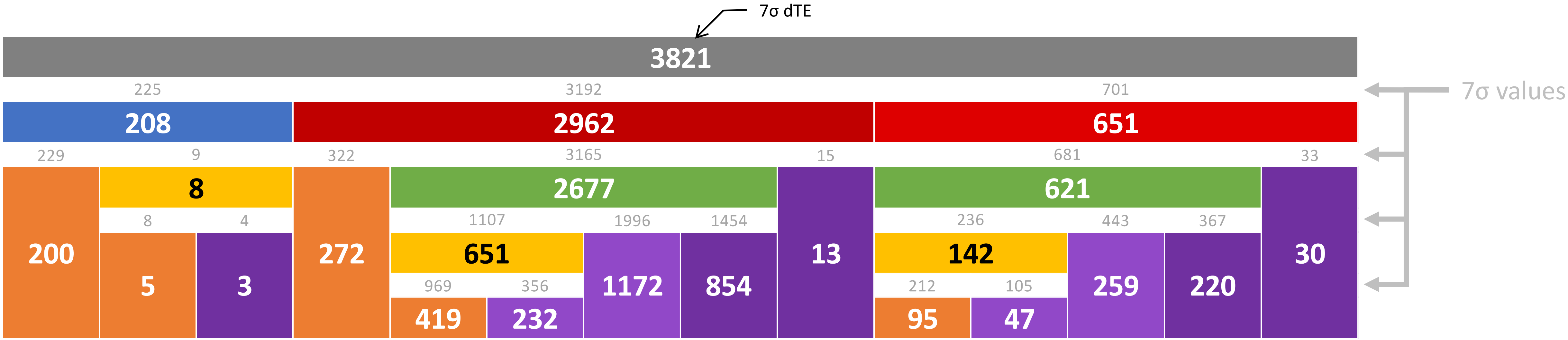
Error Components	At all hops other than last:	$DTE_{CD\_X} = MLD_{errrorCD\_X} + RT_{errrorCD\_X}$	**ns**
		$DTE_{TS\_X} = MLD_{errrorTS\_X} + RT_{errrorTS\_X}$	**ns**
	At last hop:	$DTE_{CD\_X} = MLD_{errrorCD\_X} + ES_{errrorCD\_X}$	**ns**
		$DTE_{TS\_X} = MLD_{errrorTS\_X} + ES_{errrorTS\_X}$	**ns**
	At all hops:	$DTE_{CD\_SUM}(n) = DTE_{CD\_SUM}(n-1) + DTE_{CD\_X}$	**ns**
		$DTE_{TS\_SUM}(n) = DTE_{TS\_SUM}(n-1) + DTE_{TS\_X}$	**ns**

Graphical Representation of Error Accumulation



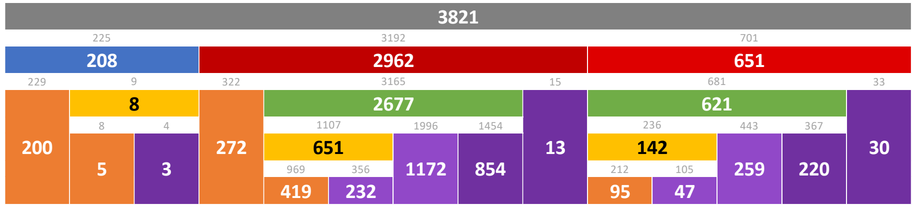
Graphical Representation of Error Accumulation



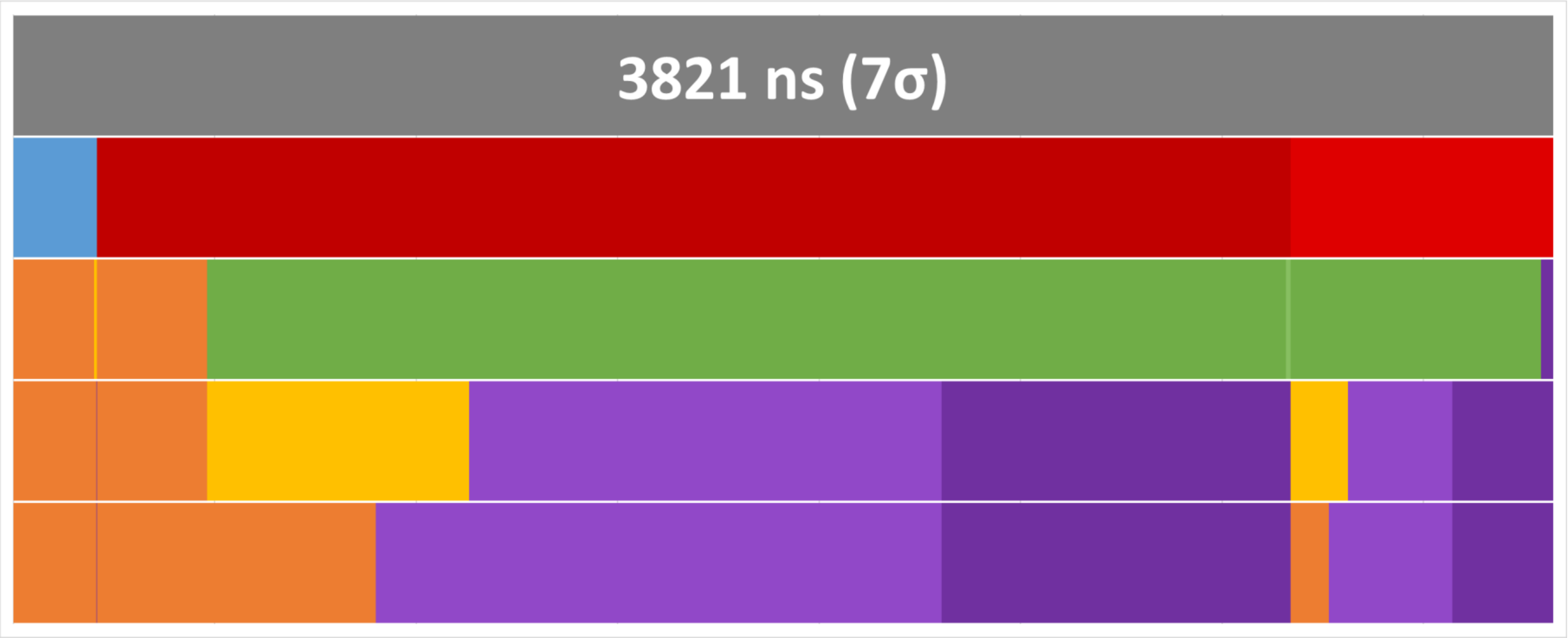


The 7σ dTE value is split repeatedly according to the ratio of 7σ values of underlying errors. 7σ probabilities do not combine via addition so, at each level, the sum of the underlying 7σ values is greater than the value that is being split. Larger errors will often swamp smaller errors, so small errors are, in general, over-represented by this approach. It does, however, provide a useful visualisation of how underlying errors combine to make up the 7σ dTE value.

Graphical Representation of Error Accumulation

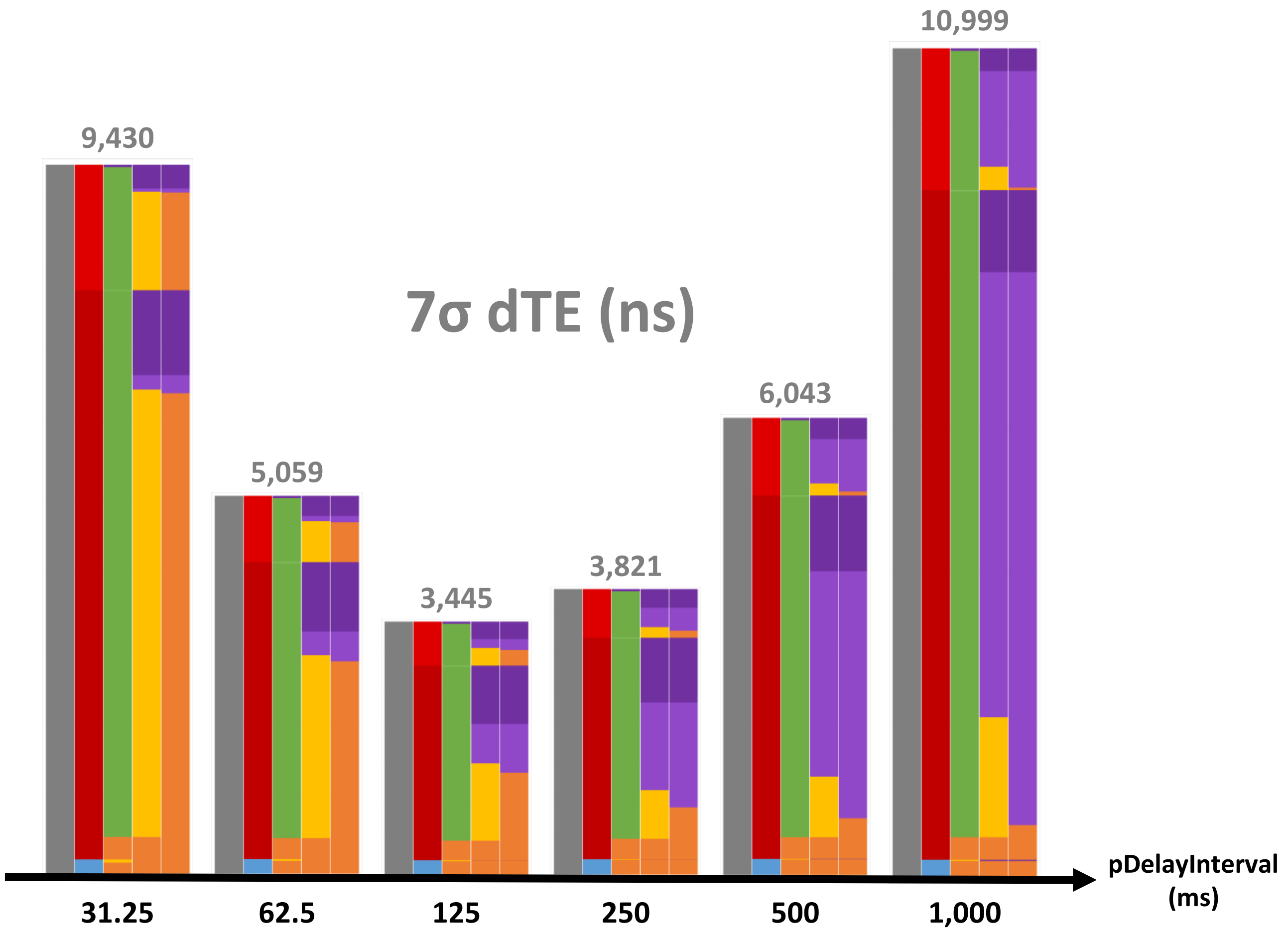


Input Errors		
Drift Type (Linear Temp Ramp)	2	
GM Clock Drift Max	+1.35	ppm/s
GM Clock Drift Min	-1.35	ppm/s
Fraction of GM nodes w/ Drift	80%	
non-GM Clock Drift Max	+1.35	ppm/s
non-GM Clock Drift Min	-1.35	ppm/s
Fraction of non-GM Nodes w/ Drift	80%	
Temp Max	+85.	°C
Temp Min	-40.	°C
Temp Ramp Rate	±1	°C/s
Temp Ramp Period	125	s
Temp Hold Period	30	s
GM Scaling Factor	100%	
non-GM Scaling Factor	100%	
Timestamp Granularity TX	±4	ns
Timestamp Granularity RX	±4	ns
Dynamic Time Stamp Error TX	±4	ns
Dynamic Time Stamp Error RX	±4	ns
Input Parameters		
pDelay Interval	250	ms
Sync Interval	125	ms
pDelay Turnaround Time	10	ms
residenceTime	10	ms
Input Correction Factors		
Mean Link Delay Averaging	0%	
NRR Drift Rate Correction	0%	
RR Drift Rate Error Correction	0%	
pDelayResp → Sync Type (Uniform)	1	
pDelayResp → Sync Max	100%	
pDelayResp → Sync Min	0%	
pDelayResp → Sync Target	10	ms
mNRR Smoothing N	1	
mNRR Smoothing M	1	
Configuration		
Hops	100	
Runs	1,000,000	



pDelayInterval Sensitivity Analysis

Input Errors		
Drift Type (Linear Temp Ramp)	2	
GM Clock Drift Max	+1.35	ppm/s
GM Clock Drift Min	-1.35	ppm/s
Fraction of GM nodes w/ Drift	80%	
non-GM Clock Drift Max	+1.35	ppm/s
non-GM Clock Drift Min	-1.35	ppm/s
Fraction of non-GM Nodes w/ Drift	80%	
Temp Max	+85.	°C
Temp Min	-40.	°C
Temp Ramp Rate	±1	°C/s
Temp Ramp Period	125	s
Temp Hold Period	30	s
GM Scaling Factor	100%	
non-GM Scaling Factor	100%	
Timestamp Granularity TX	±4	ns
Timestamp Granularity RX	±4	ns
Dynamic Time Stamp Error TX	±4	ns
Dynamic Time Stamp Error RX	±4	ns
Input Parameters		
pDelay Interval	VAR	ms
Sync Interval	125	ms
pDelay Turnaround Time	10	ms
residenceTime	10	ms
Input Correction Factors		
Mean Link Delay Averaging	0%	
NRR Drift Rate Correction	0%	
RR Drift Rate Error Correction	0%	
pDelayResp → Sync Type (Uniform)	1	
pDelayResp → Sync Max	100%	
pDelayResp → Sync Min	0%	
pDelayResp → Sync Target	10	ms
mNRR Smoothing N	1	
mNRR Smoothing M	1	
Configuration		
Hops	100	
Runs	1,000,000	



Algorithmic Improvements & Corrections

Aligning pDelayResp with Sync

- Clock drift between measuring NRR (mNRR) and using mNRR during Sync messaging (to calculate RR and then multiply meanLinkDelay + residenceTime by RR) introduces an error:

$$RR_{errorCD\_NRR2Sync} = \frac{T_{mNRR2sync}(\boldsymbol{clockDrift_n} - \boldsymbol{clockDrift_{n-1}})}{10^3} \qquad T_{mNRR2sync} = \sim U(\boldsymbol{pdelayInterval}.0.9, \boldsymbol{pdelayInterval}.1.3) \times \sim U(0, 1)$$

- By aligning pDelayResp with Sync messaging RR_{error} can be reduced.

Aligning pDelayResp with Sync - Parameters

- The Monte Carlo Model offers three approaches to modelling alignment of pDelayResp with Sync, controlled via input parameter pDelayRespSyncAlignMode

Correction Parameter	Default	Unit	Notes
pDelayRespSyncAlignMode	1	-	1: Uniform distribution between a minimum and maximum fraction of $T_{pdelay2pdelay}$ without any alignment 2: Gamma distribution with target *pDelayRespSyncAlignTarget* 3: Gaussian distribution with mean *pDelayRespSyncAlignTarget* and standards deviation *pDelayRespSyncAlignSD*
pDelayRespSyncAlignMax	1	-	Maximum fraction for uniform distribution
pDelayRespSyncAlignMin	0	-	Minimum fraction for uniform distribution
pDelayRespSyncAlignTarget	10	ms	Target value for Gamma and Gaussian distribution
pDelayRespSyncAlignSD	3	ms	Standard deviation for Gaussian Distribution

Aligning pDelayResp with Sync - Parameters

- In mode 1:

$$T_{mNRR2sync} = \sim U(\mathit{\mathbf{pdelayInterval}}.0.9, \mathit{\mathbf{pdelayInterval}}.1.3\,) \times \sim \boldsymbol{U}(pDelayRespSyncAlignMin, pDelayRespSyncAlignMax)$$

- In mode 2:

$$T_{mNRR2sync} = \sim\Gamma\left(270.5532, \frac{270.5532}{pDelayRespSyncAlignTarget}\right)$$

- In mode 3:

$$T_{mNRR2sync} = \sim\boldsymbol{\mathcal{N}}(pDelayRespSyncAlignTarget, pDelayRespSyncAlignSD)$$

mNNRsmoothingN

- The Monte Carlo approach models using timestamp values from older pDelayResp messages via the *mNRRsmoothingN* parameter adjusting *Tpdelay2pdelay*.

Correction Parameter	Default	Unit	Notes
mNRRsmoothingN	1	-	Must be a whole number, minimum value 1.

$$T_{pdelay2pdelay} = \sum_{x=1}^{mNRRsmoothingN} \sim U(\mathbf{pdelayInterval}.0.9, \mathbf{pdelayInterval}.1.3)$$

MLD, NRR and RR Error Correction

- The model includes error correction factors for MLD, and Clock Drift related errors in NRR and RR
  - As the Monte Carlo approach is not a time series simulation, it does not model any of these algorithmic corrections in detail. It instead assumes a percentage effective factor, i.e. how much of the relevant error would be removed.
  - It is assumed that MLD error correction is accomplished via averaging.
  - It is assumed that NRR and RR error correction is accomplished via measuring Clock Drift in the past and – as Clock Drift is relatively consistent over time – compensating for Clock Drift in the future.

MLD Error Correction

Correction Parameter	Default	Unit	Notes
mLinkDelayErrCor	0	-	Value between zero (no error correction) and 1 (error is eliminated)

$$MLD_{errrorTSdirect\_X} = \frac{(t_{4pderror} - t_{1pderror}) - (t_{3pderror} - t_{2pderror})}{2}(1 - \boldsymbol{mLinkDelayErrCor})$$

$$MLD_{errrorNRR\_X} = -\frac{pDelayTurnaround.mNRR_{error}}{2}(1 - \boldsymbol{mLinkDelayErrCor})$$

$$MLD_{errrorNRR\_TS\_X} = -\frac{pDelayTurnaround.mNRR_{error\_TS\_X}}{2}(1 - \boldsymbol{mLinkDelayErrCor})$$

$$MLD_{errrorCD\_X} = -\frac{pDelayTurnaround.mNRR_{error\_CD\_X}}{2}(1 - \boldsymbol{mLinkDelayErrCor})$$

NRR and RR Error Correction

Correction Parameter	Default	Unit	Notes
NRRdriftRateErrorCor	0	-	Value between zero (no error correction) and 1 (error is eliminated)
RRdriftRateErrorCor	0	-	Value between zero (no error correction) and 1 (error is eliminated)

$$mNRR_{errorCD\_X} = \frac{T_{pdelay2pdelay}(clockDrift_n - clockDrift_{n-1})}{2 \times 10^3}(1 - NRRdriftRateErrorCor)$$

$$RR_{errrorCD\_NRR2Sync\_X} = \frac{T_{mNRR2Sync}(clockDrift_n - clockDrift_{n-1})}{10^3}(1 - NRRdriftRateErrorCor)$$

$$RR_{errrorCD\_RR2Sync\_X} = \frac{residenceTime(clockDrift_{n-1} - clockDrift_{GM})}{10^3}(1 - RRdriftRateErrorCor)$$

$$RT_{errrorCDdirect\_X} = \frac{residenceTime^2(clockDrift_n - clockDrift_{GM})}{2 \times 10^3}(1 - RRdriftRateErrorCor)$$

$$ES_{errorCDdirect\_X} = \frac{{T_{sync2sync}}^2(clockDrift_n - clockDrift_{GM})}{2 \times 10^3}(1 - RRdriftRateErrorCor)$$

Thank you!

Equations – mNRR – Independence of TS & CD errors

ppm

$$mNRR_{error} = mNRR_{measuredCDTSerror} - mNRR_{nominal}$$

$$= \left(\left(\frac{\left((t_3 + t_{3pderror}) - (t_{3'} + t_{3pderror}{}' + t_{3CDerror}{}')\right)}{\left((t_4 + t_{4pderror}) - (t_4{}' + t_{4pderror}{}' + t_{4CDerror}{}')\right)}\right) - 1\right) \times 10^6 - \left(\left(\frac{t_3 - t_3{}'}{t_4 - t_4{}'}\right) - 1\right) \times 10^6$$

$$= \left(\left(\frac{t_3 - t_3' + t_{3pderror} - t'_{3pderror} - t_{3CDerror}{}'}{t_4 - t_4' + t_{4pderror} - t'_{4pderror} - t_{4CDerror}{}'}\right) - 1\right) \times 10^6 - \left(\left(\frac{t_3 - t_3{}'}{t_4 - t_4{}'}\right) - 1\right) \times 10^6$$

$$= \left(\left(\frac{t_3 - t_3{}'}{t_4 - t_4{}' + t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'}\right) - 1 + \left(\frac{t_{3pderror} - t_{3pderror}{}' - t_{3CDerror}{}'}{t_4 - t_4{}' + t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'}\right)\right) \times 10^6 - \left(\left(\frac{t_3 - t_3{}'}{t_4 - t_4{}'}\right) - 1\right) \times 10^6$$

$$= \frac{(t_4 - t_4{}')(t_{3pderror} - t_{3pderror}{}' - t_{3CDerror}{}') - (t_3 - t_3{}').(t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}')}{(t_4 - t_4{}')\left((t_4 - t_4{}') + t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'\right)} \times 10^6$$

$$\approx \frac{T_{pdelay2pdelay} \times 10^6 (t_{3pderror} - t_{3pderror}{}' - t_{3CDerror}{}') - T_{pdelay2pdelay} \times 10^6.(t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}')}{T_{pdelay2pdelay} \times 10^6 (T_{pdelay2pdelay} \times 10^6 + t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}')} \times 10^6$$

The error magnitudes are small relative to the $t_3 - t_3{}'$ and $t_4 - t_4{}'$ factors, which are both nominally $T_{pdelay2pdelay}$ (which is in ms, whereas the timestamps are in nanoseconds, hence $T_{pdelay2pdelay}$ x 10^6).

Equations – mNRR – Independence of TS & CD errors

$$mNRR_{error} \approx \frac{T_{pdelay2pdelay} \times 10^6 \left(t_{3pderror} - t_{3pderror}{}' - t_{3CDerror}{}'\right) - T_{pdelay2pdelay} \times 10^6 . \left(t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'\right)}{T_{pdelay2pdelay} \times 10^6 \left(T_{pdelay2pdelay} \times 10^6 + t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'\right)} \times 10^6$$

ppm

$$= \frac{\left(t_{3pderror} - t_{3pderror}{}' - t_{3CDerror}{}'\right) - \left(t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'\right)}{T_{pdelay2pdelay} \times 10^6 + t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'} \times 10^6$$

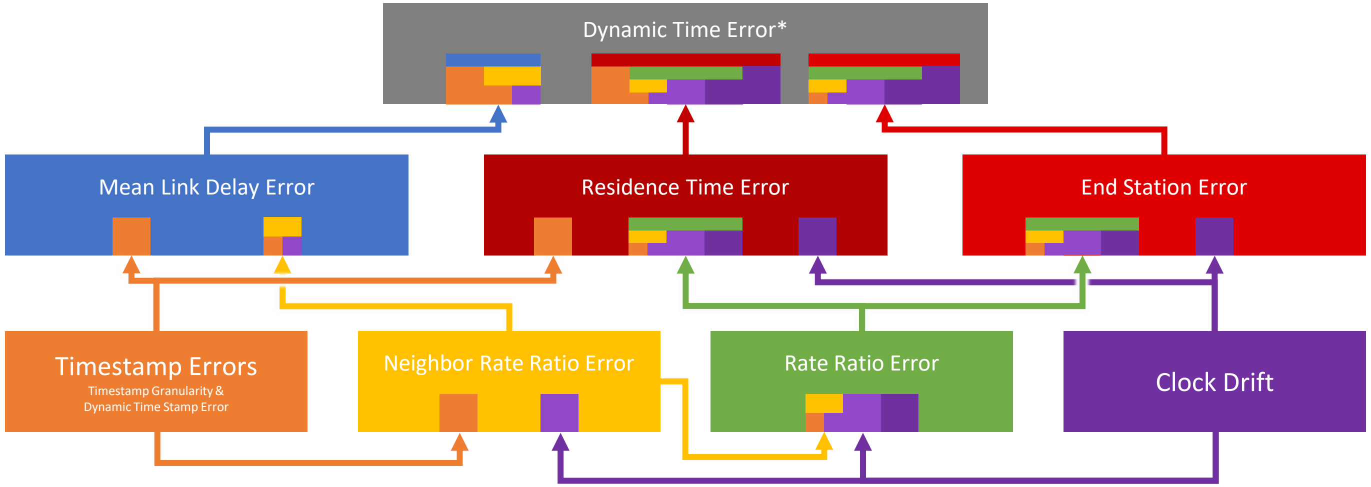
$$= \frac{\left(t_{3pderror} - t_{3pderror}{}' - t_{3CDerror}{}'\right) - \left(t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'\right)}{T_{pdelay2pdelay} + \dfrac{t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'}{10^6}}$$

$$\approx \frac{\left(t_{3pderror} - t_{3pderror}{}'\right) - \left(t_{4pderror} - t_{4pderror}{}'\right)}{T_{pdelay2pdelay}} + \frac{t_{4CDerror}{}' - t_{3CDerror}{}'}{T_{pdelay2pdelay}}$$

$$= mNRR_{errorTS} + mNRR_{errorCD}$$

$t_{4pderror} - t_{4pderror}{}' - t_{4CDerror}{}'$ divided 10^6 by on the lower line is small enough relative to $T_{pdelay2pdelay}$ to ignore.

Colours for Charts



	Lines	Areas	Backgrounds
	7F7F7F	BFBFBF	F2F2F2
	4472C4	B4C7E7	DAE3F3
	C00000	FF8181	FFBDBD
	DE0000	FF9B9B	FFC3C3
	ED7D31	F8CBAD	FBE5D6
D09E00	FFC000	FFE699	FFF2CC
	70AD47	C5E0B4	E2F0D9
	7030A0	C198E0	DFC9EF
	9148C8	C8A3E3	E2CFF1