IBIS-AMI and Co-Optimization

Todd Westerhoff, Walter Katz, and Mike LaBonte

December 2014

SiSoft™
We Are Signal Integrity

Optimization

http://en.wikipedia.org/wiki/Mathematical_optimization

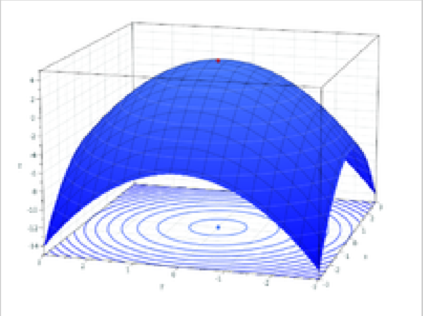
Mathematical optimization

From Wikipedia, the free encyclopedia
(Redirected from Optimization)

"Optimization" and "Optimum" redirect here. For other uses, see Optimization (disambiguation) and Optimum (disambiguation).

In mathematics, computer science, economics, or management science, **mathematical optimization** (alternatively, **optimization** or **mathematical programming**) is the selection of a best element (with regard to some criteria) from some set of available alternatives.[1]

In the simplest case, an optimization problem consists of maximizing or minimizing a real function by systematically choosing input values from within an allowed set and computing the value of the function. The generalization of optimization theory and techniques to other formulations comprises a large area of applied mathematics. More generally, optimization includes finding "best available" values of some objective function given a defined domain (or a set of constraints), including a variety of different types of objective functions and different types of domains.



Graph of a paraboloid given by $f(x, y) = -(x^2 + y^2) + 4$. The global maximum at (0, 0, 4) is indicated by a red dot.

We Are Signal Integrity

Other Terms …

The page "Co-Optimization" does not exist.
see whether the topic is already covered.
For search help, please visit Help:Searching.

Co-Optimization

Link Trainer

From Wikipedia, the free encyclopedia
(Redirected from Link Training)

The term **Link Trainer,** also known as the "Blue box" and "Pilot Trainer"[1] is commonly used to refer to a series of flight simulators produced between the early 1930s and early 1950s by the **Link Aviation Devices, Inc**, founded and headed by Ed Link, based on technology he pioneered in 1929 at his family's business in Binghamton, New York. These simulators became famous during World War II, when they were used as a key pilot training aid by almost every combatant nation.

Link trainer in use at a British Fleet Air Arm station in 1943

The original Link Trainer was created in 1929 out of the need for a safe way to teach new pilots how to fly by instruments. A former organ and nickelodeon builder, Link used his knowledge of pumps, valves and bellows to create a flight simulator that responded to the pilot's controls and gave an accurate reading on the included instruments. More than 500,000 US pilots were trained on Link simulators,[2] as were pilots of nations as diverse as Australia, Canada, Germany, United Kingdom, Israel, Japan, Pakistan and the USSR.

The Link Flight Trainer has been designated as a *Historic Mechanical Engineering Landmark* by the American Society of Mechanical Engineers.[2] The Link Company, now the Link Simulation & Training division of L-3 Communications, continues to make aerospace simulators.[3]

Link Training

Backchannel (disambiguation)

From Wikipedia, the free encyclopedia

A **backchannel** is a real-time online conversation using networked computers that takes place alongside live spoken remarks.

Backchannel may also refer to:

- Return channel, a low-speed, or less-than-optimal, telecommunications transmission channel in the opposite direction to the main channel
- Backchanneling, the method a malicious malware program uses to secretly communicate to command and control servers from a compromised computer
- Track II diplomacy, an unofficial channel of communication between states or other political entities
- Backchannel (linguistics), listener responses that can be both verbal and non-verbal in nature
- Backchanneling, an organizational practice in business that involves bypassing recognized or official chains of command in order to create vulnerability[*clarification needed*] at the level(s) skipped

Back Channel

Scenario #1: SerDes IP **with** Hardware Backchannel

- Model SerDes hardware that uses run-time communication to co-optimize TX & RX settings
  - AMI models implement hardware optimization protocol as closely as possible to predict how hardware will converge
  - Implies that results are optimization path-dependent and may be a local (i.e. non-global) optimum

- Designer's Questions:
  - Will this link converge?
  - Do we need specific presets to ensure convergence?
  - What will the trained margins be?

Scenario #1 Requirements

1. TX & RX models emulate hardware protocol
   - Models must communicate at simulation run-time
2. Report optimized margins (eye height, width, etc.)
3. Cross-vendor support
4. Report optimized IP settings (taps, etc.)
5. Constrain solution based on IP capabilities
6. Init and GetWave-based optimization
7. Support hardware starting point (presets)
8. Probes work correctly (TX EQ from TX)

Scenario #2: SerDes IP **without** Hardware Backchannel

- Optimize TX/RX setting for SerDes hardware that **does not** use hardware run-time optimization
  - AMI models perform co-optimization **beyond** what hardware can do at run-time
  - Implies that results are optimization path-independent and seek to provide a global optimum
  - Implies optimization **in addition to** what a device can do by itself (i.e. DFE tap adaptation)

- Designer's Questions:
  - Can this link work with this IP?
  - How should IP be configured?
  - What will the margins be?

Why Scenario #2 ?

- Provide a starting point for lab validation
- Optimize settings for individual links at the system level
  - As opposed to “bucketing by length”
- Eliminate need to run “blind sweep” simulations

Scenario #2 Requirements

1. Report optimized IP settings and margins
2. Cross-vendor support
3. Provable methodology
   - Ability to validate results independently
4. High performance optimization
   - Optimize 4,000 links overnight
5. Constrain solution based on IP capabilities
6. Probes work correctly
7. User-selectable optimization criteria
   - Defined objectives and constraints

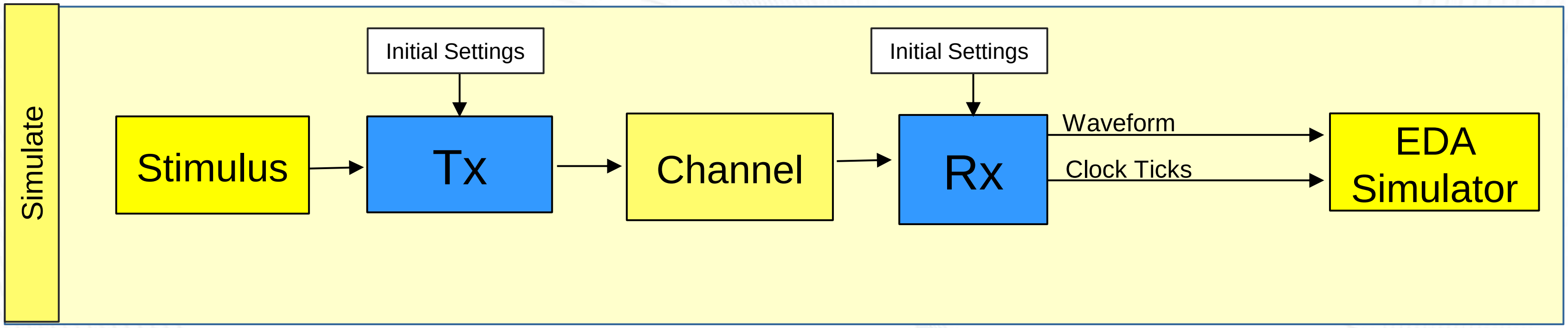
Terminology

- **Adaptation** – When AMI models change their behavior (EQ, clock recovery) on a bit by bit basis in time-domain simulation
- **Eye Quality Metric (EQM)** – Computed measure of eye quality at the RX decision point, such that different scenarios can be evaluated and the best case identified
- **Self-Optimization** – When an RX adjusts it internal behavior based on an internal EQM to optimize its settings
- **Co-Optimization** – Simultaneous adjustment of TX / RX settings to optimize EQM, usually at the RX decision point
- **Co-Optimization by proxy** – Where one device provides the EQ that would normally be provided by another. In AMI this normally occurs with TX EQ disabled and the RX providing the TX (LTI) equalization in the TX's place

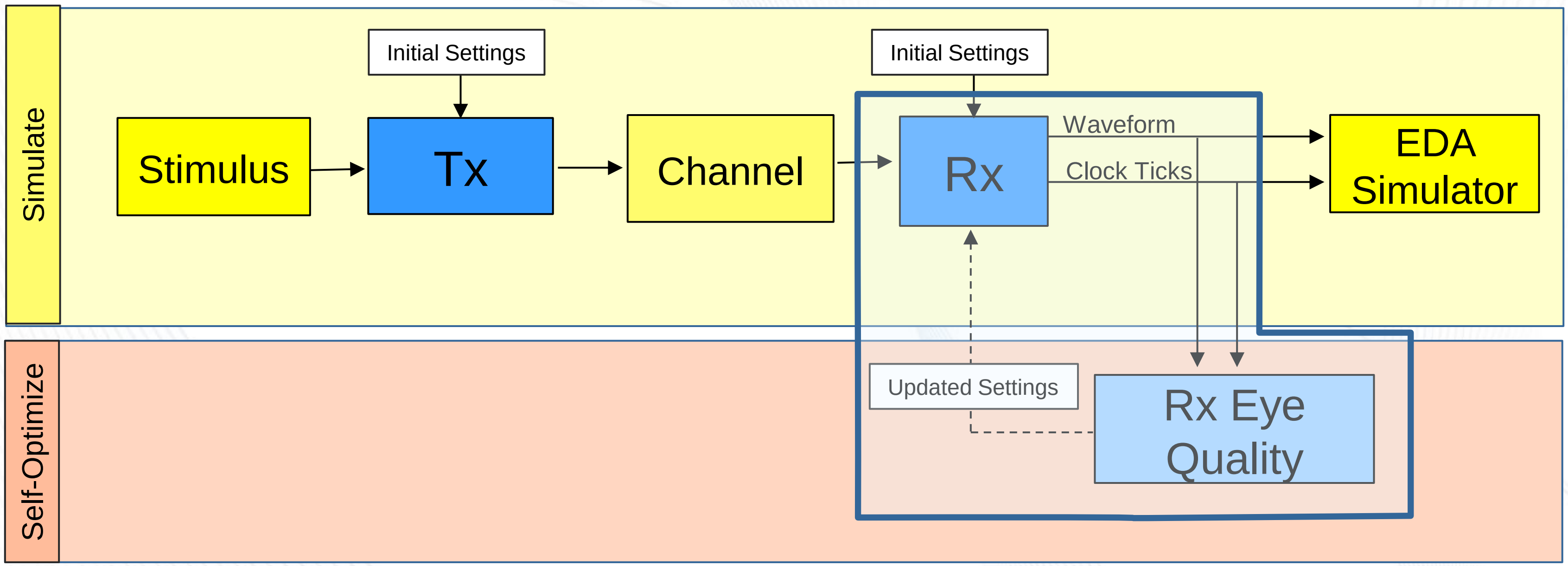
Key Questions

- What is being optimized?
- Who does the optimizing?
- What optimization algorithm?
- Local or global optimum?

No Optimization



Self-Optimization

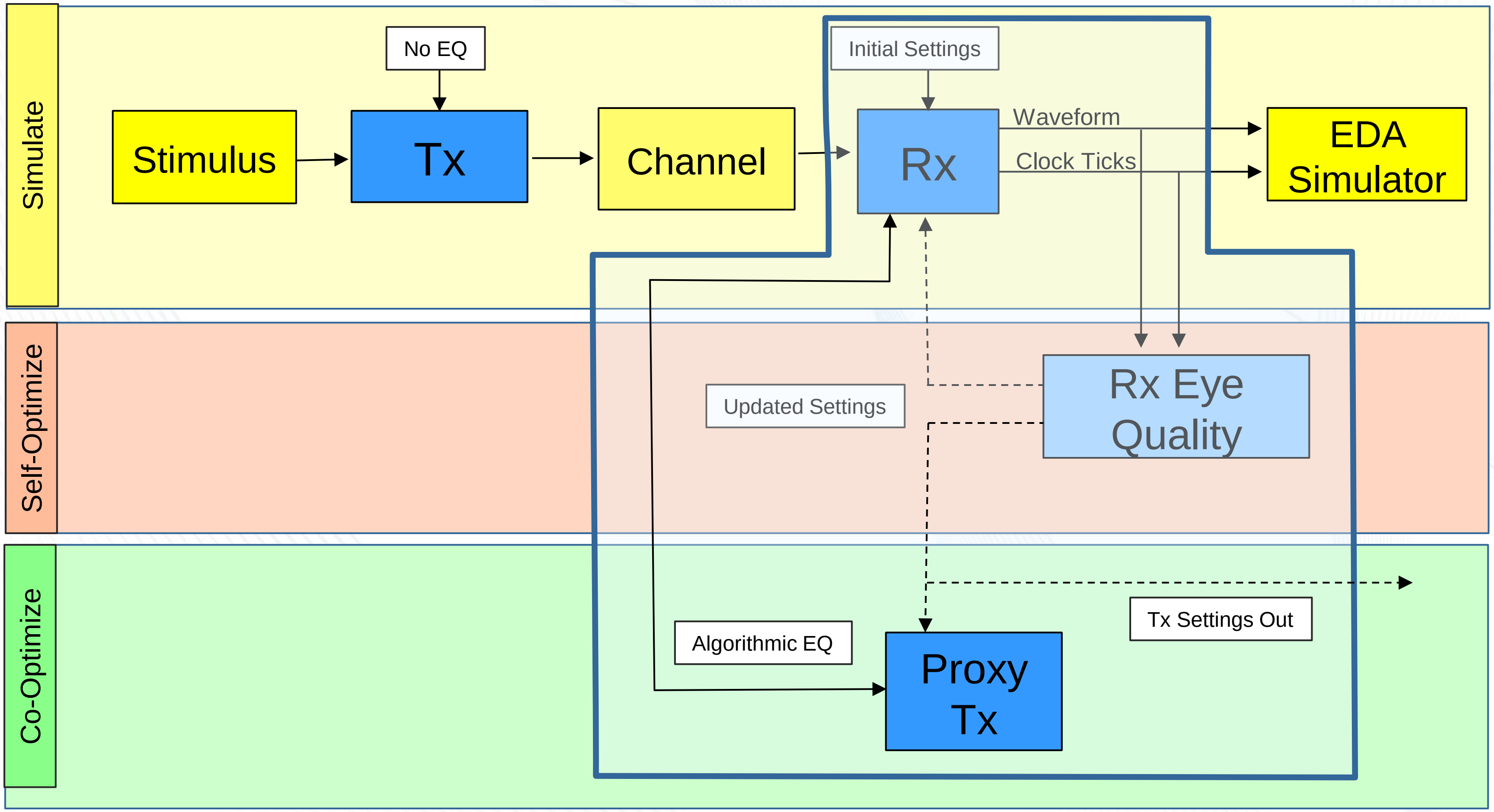


Legend: EDA AMI

Co-Optimization by Proxy

- Special case using matched pair models:
  - TX equalization is disabled
  - RX provides TX equalization in place of TX
  - RX co-optimizes itself and proxy TX settings
  - Optimized TX settings are reported
  - Subsequent analysis used to verify TX settings
- Requires no changes to existing AMI flows
- RX has detailed knowledge of TX EQ
  - Only works for paired TX / RX models
  - Limited / no cross-vendor support

Co-Optimization by Proxy



Legend: EDA AMI

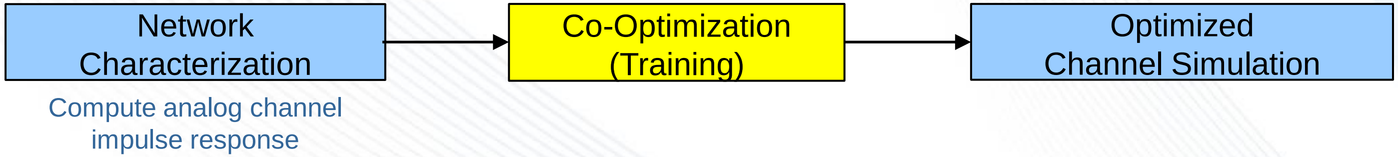
SiSoft We Are Signal Integrity

Co-Optimization - Analysis Flow

- In hardware, co-optimization (link training) precedes normal system operation.
- We believe that simulation-based co-optimization should follow the same pattern

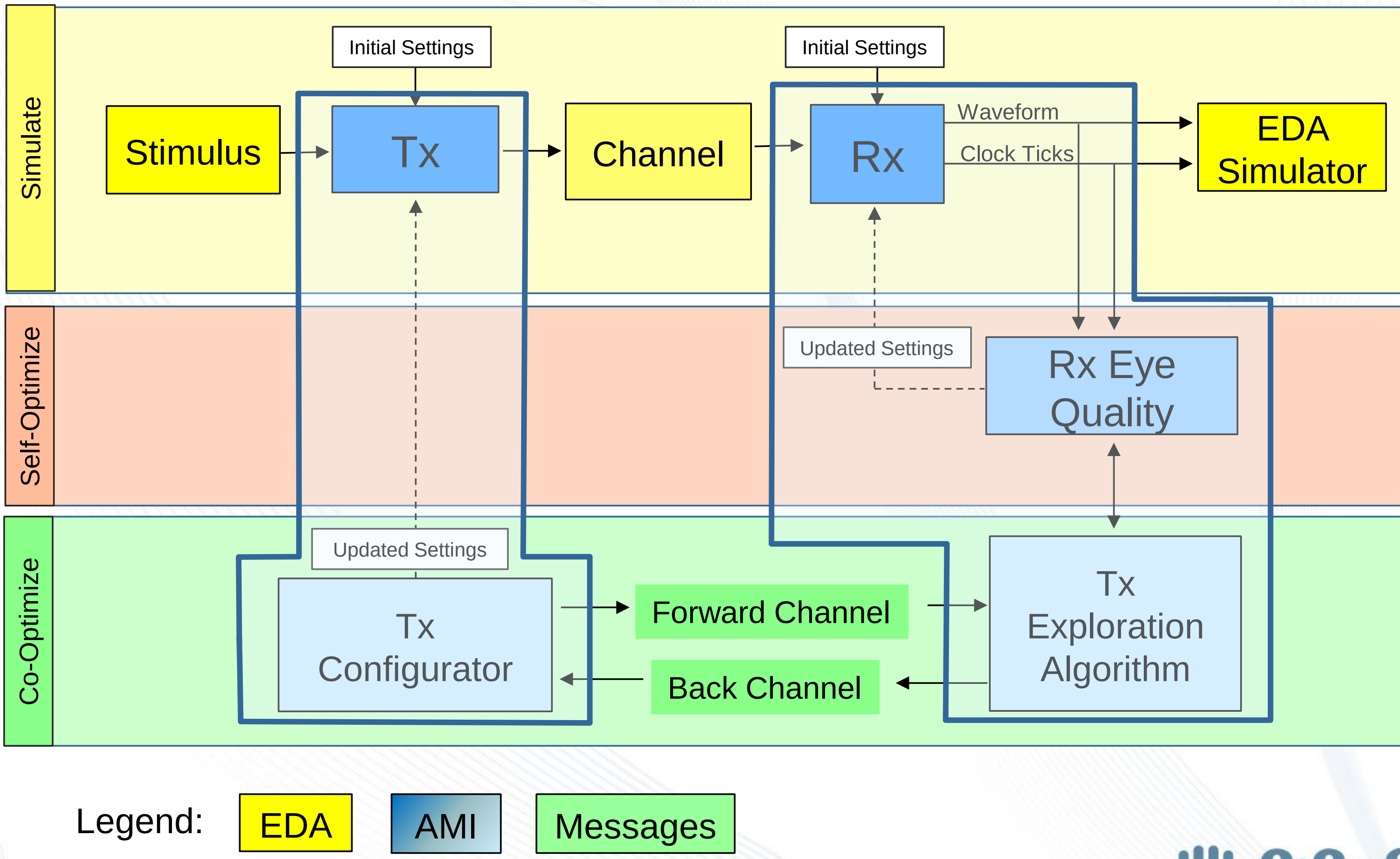
Normal

Co-Optimization



We Are Signal Integrity

Co-Optimization



Key Points

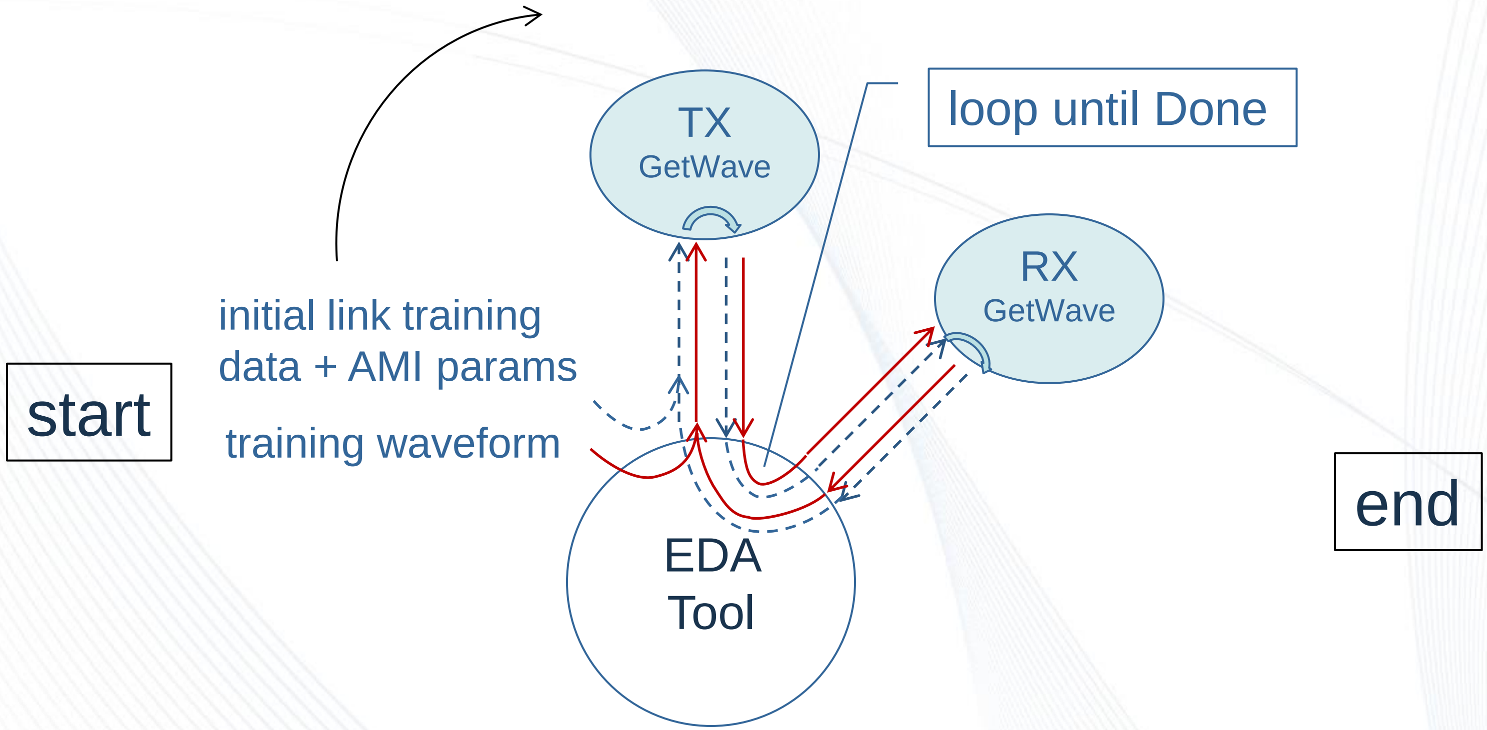
- What is being optimized?
  - Eye Quality –quantitative metric determined by the RX model based on looking at its own behavior. Does not have to be rigorously defined or even reported.
- Who does the optimizing?
  - The RX model
- What optimization algorithm?
  - Scenario #1 – an algorithm internal to the RX model that follows the hardware protocol as closely as possible
  - Scenario #2 – an algorithm internal to the RX model
- Local or Global optimum?
  - Scenario #1: Local
  - Scenario #2: Local or Global

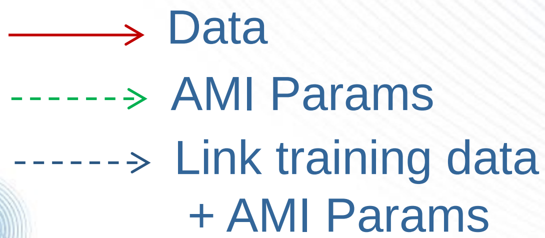
SiSoft™ We Are Signal Integrity

Analysis Mode Requirements

- Emulate hardware training protocol as closely as possible (Scenario 1, requirement 1)
  - Requires training be performed using bit by bit waveform processing (Getwave)
- High performance Co-Optimization (Scenario 2, requirement 4)
  - Requires training be performed using impulse response processing (Init)

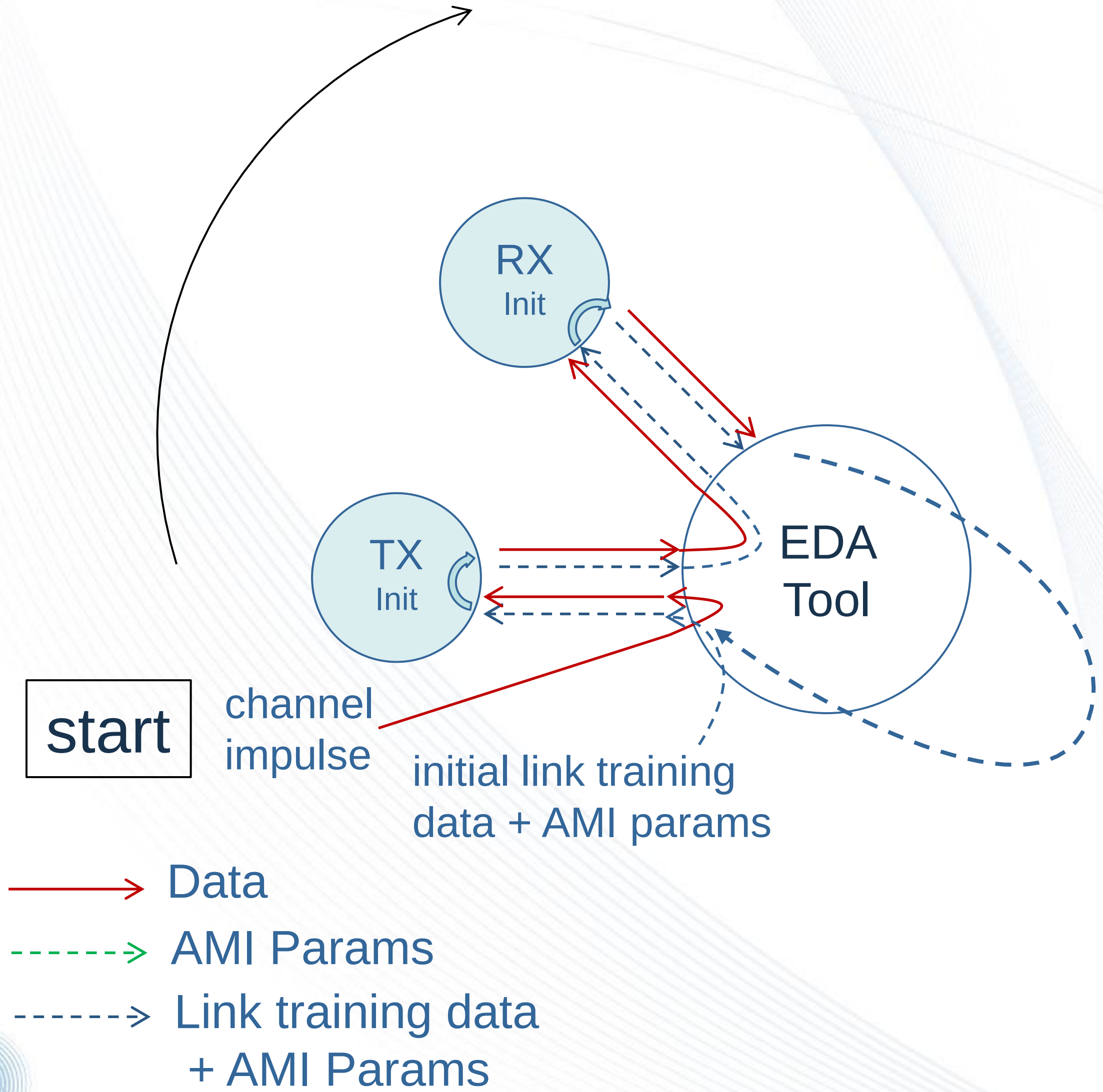
Time Domain Link Training





SiSoft We Are Signal Integrity

Statistical Link Training

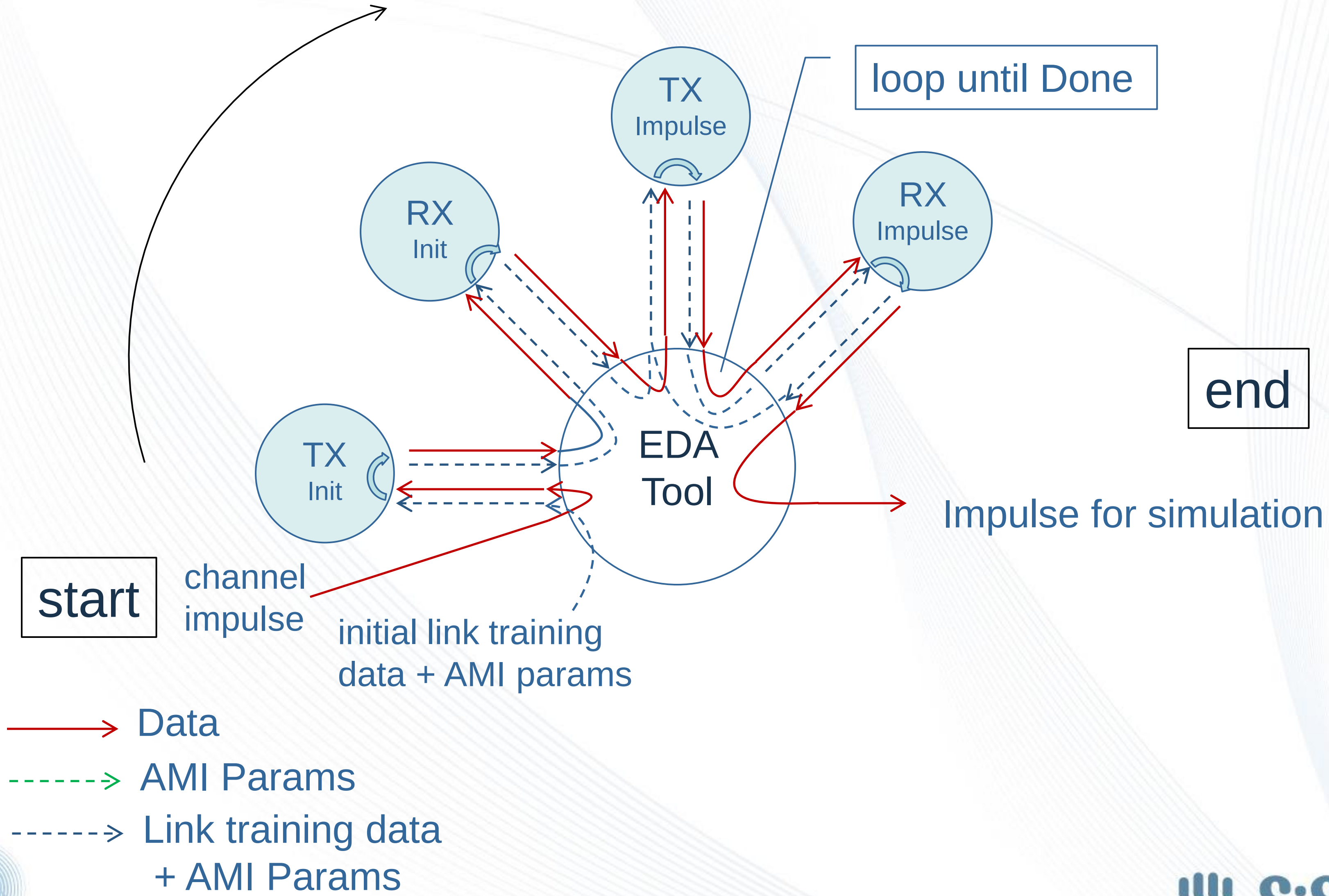


Problem: We need to call AMI_Init again, but it can be called only once.

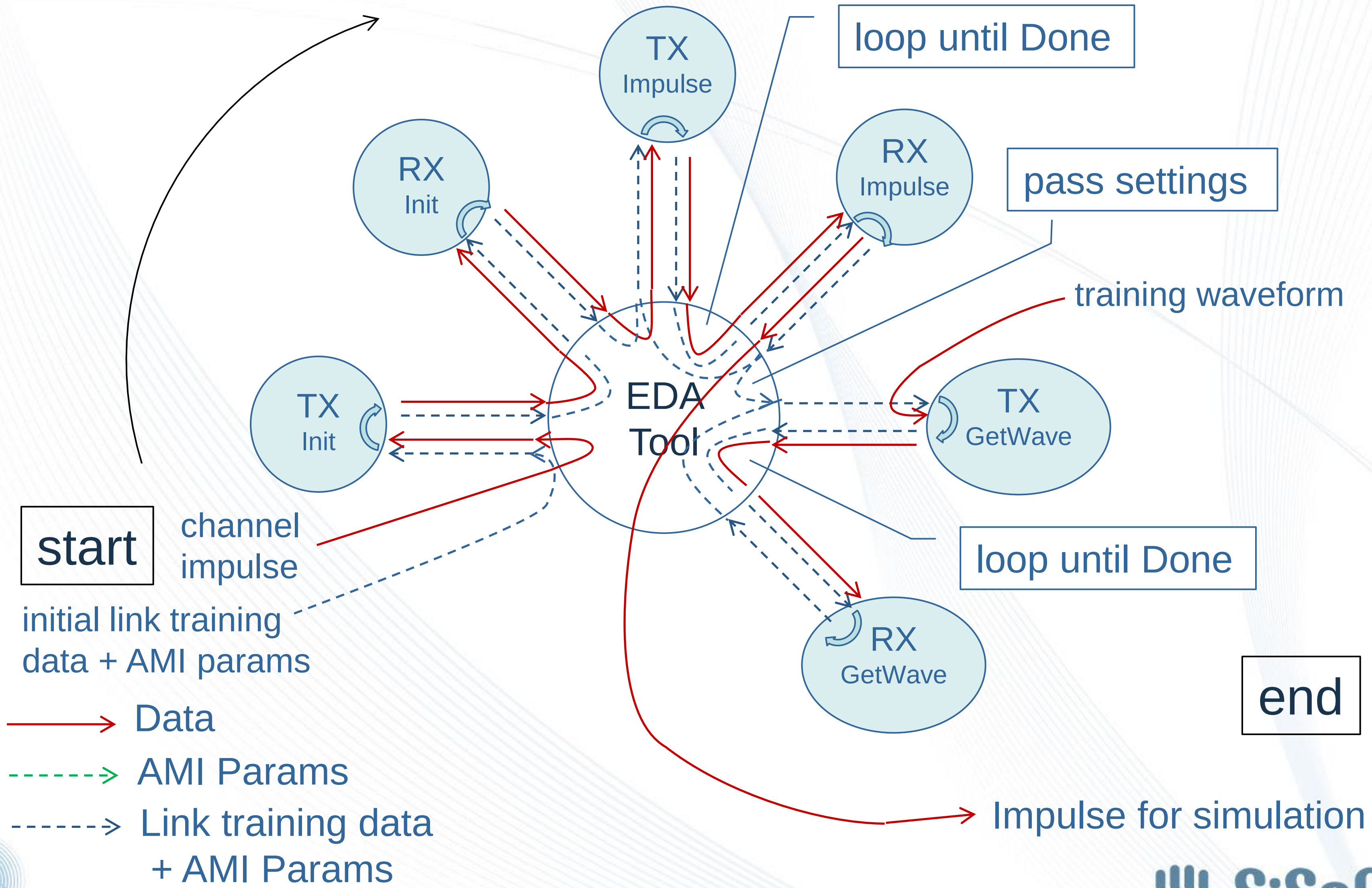
Solution: New function ***AMI_Impulse*** with same signature as AMI_Init:

- ✗ Initialize
- ✓ Params In/Out
- ✓ Impulse response

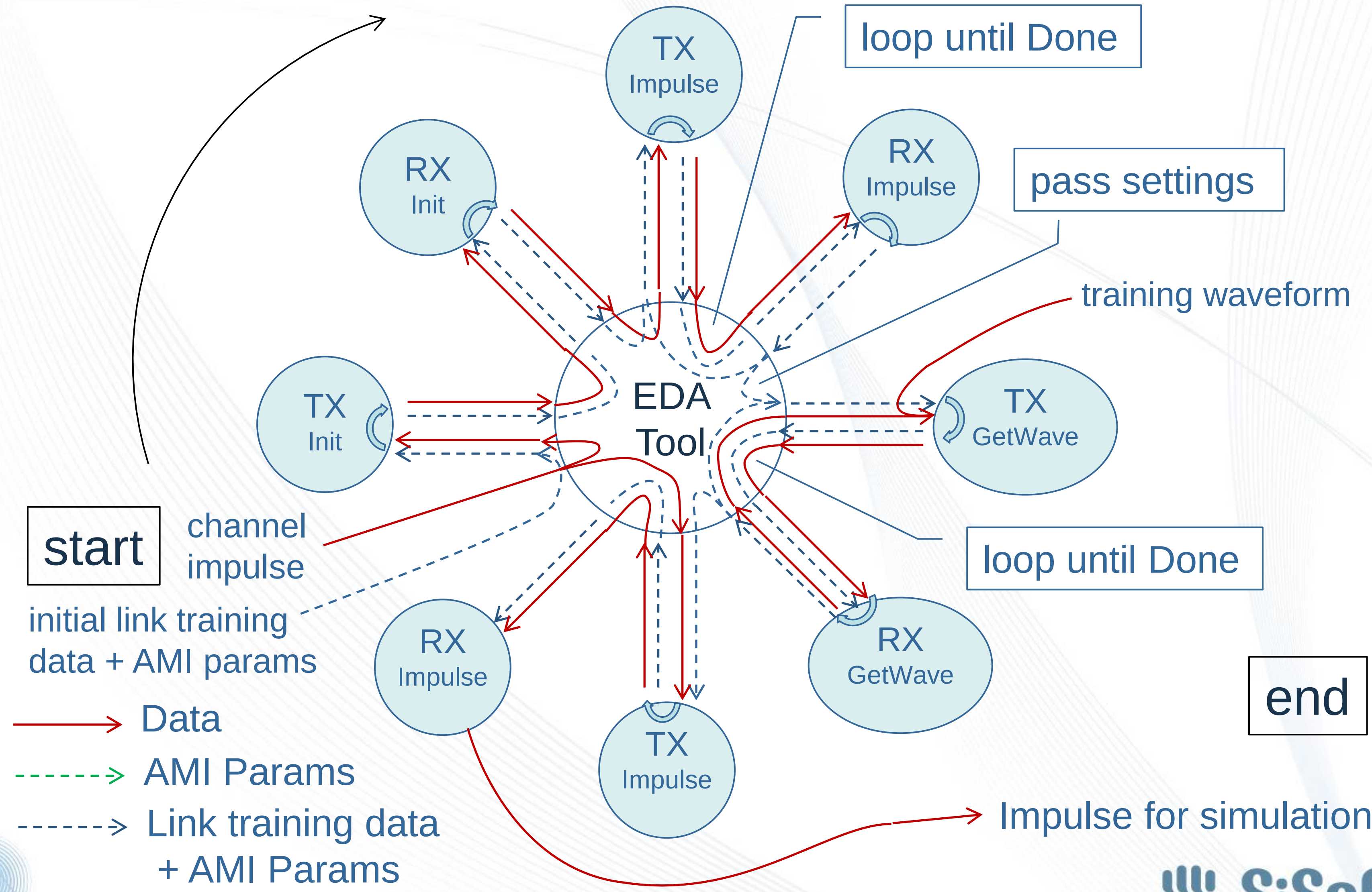
Statistical Link Training (again)



Statistical / Time-Domain Link Training



Enhanced Statistical Link Training



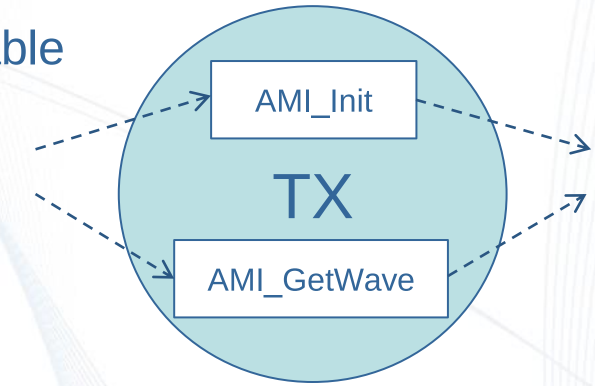
Key Questions

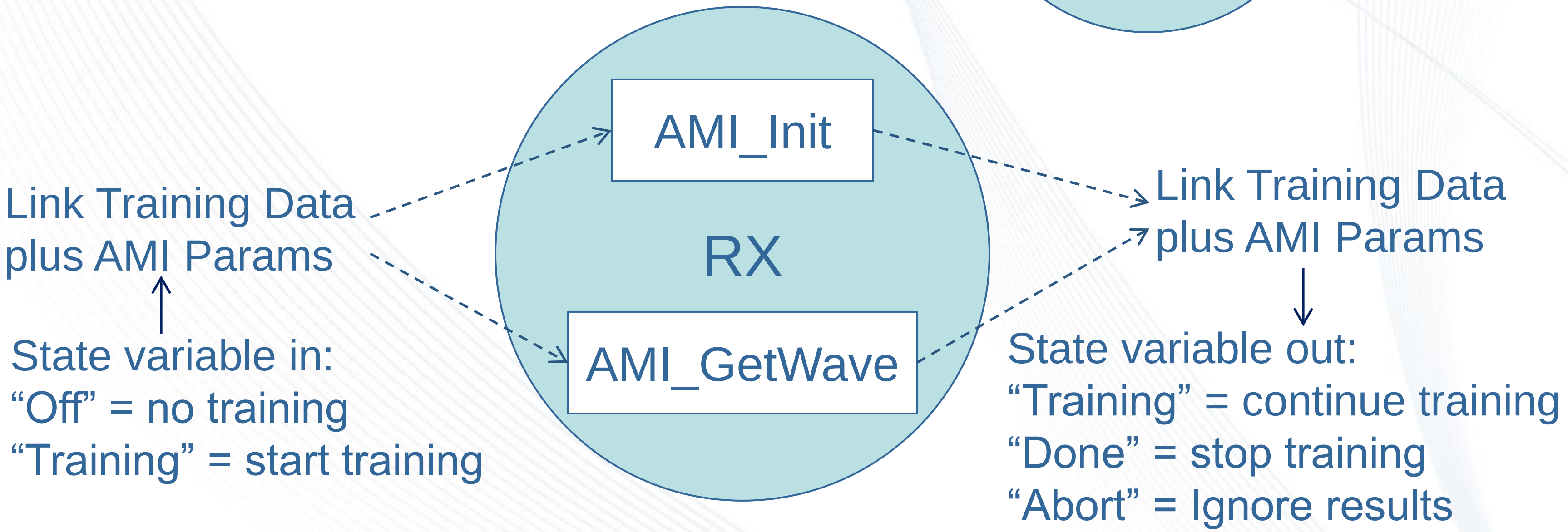
- How does training start? stop?
- How are EQ adjustments communicated?
- How can device constraints (min/max tap settings, resolution) be taken into account?

Link Training Flow Overview

TX also reads and writes Link Training data

- TX does not change state variable





SiSoft
We Are Signal Integrity

DLL Function Signature Changes

Changes in red

- AMI_Init
  - Input: Parameters_In
  - Output: Parameters_Out
- AMI_Impulse
  - New function; signature identical to AMI_Init
  - Does not initialize, can be called repeatedly
  - Input: Parameters_In
  - Output: Parameters_Out
- AMI_GetWave
  - Input: BIRD 128 changes Parameters_Out to Parameters_InOut
  - Output: Parameters_InOut

SiSoft We Are Signal Integrity

SiSoft™

We Are Signal Integrity