audio.dev CONFERENCE BRISTOL & ONLINE 2026

Model-Based Testing for Audio Devices

How We Test on Hardware Early

Max Bailey & Ray Brien

Introduction



Max Bailey

Software Developer in Test @ Focusrite



Ray Brien

Software Developer in Test @ Focusrite

Previous Talks

Automating Audio Device Testing - A Brief Existential Guide

Joseph Deller & Jake Wignall - ADC22

Focusrite's Hardware Test Platform - Testing Thousands of Little Red Boxes Every Day

Dave Curtis & Adrien Fauconnet - ADC23

Agenda

1. **Test Design** - *Why and how we switched from Script-Based to Model-Based Testing*

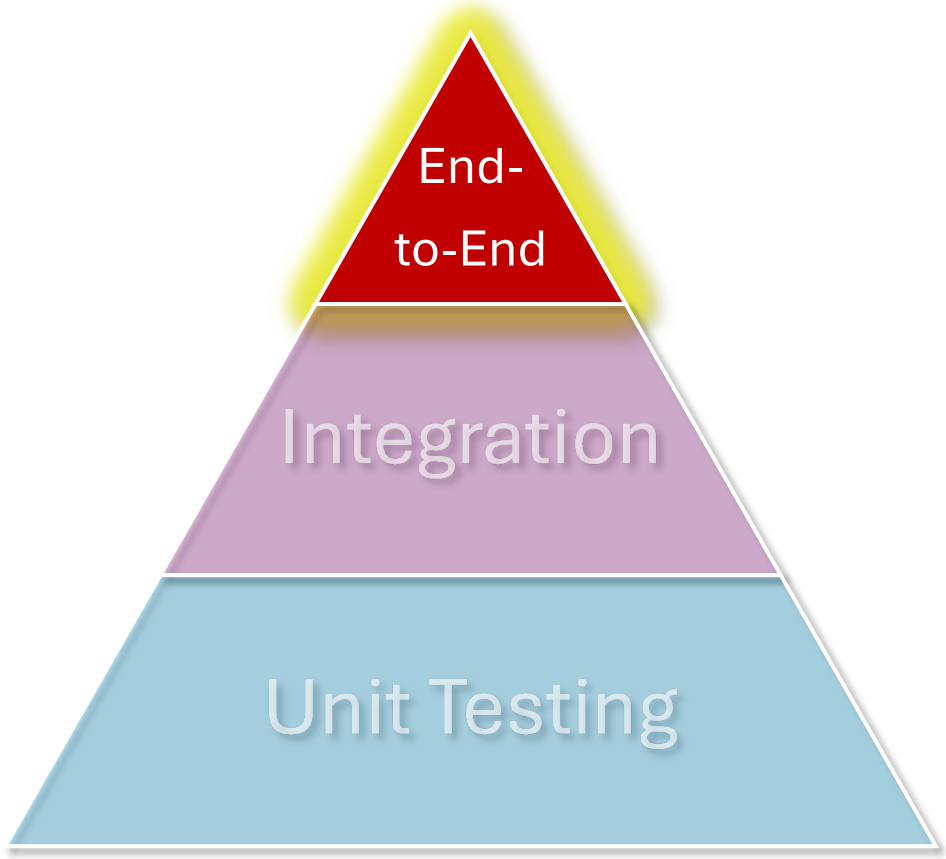
2. **Architecture** - *Putting everything together*

3. **Demo**

4. **Q&A**

Our Focus

- End-to-End
  - Runs on **hardware**
  - Exercising the **complete** system
- Long, **exhaustive** test coverage
- Tests need to be **reliable**, failures need to be **actionable**
- *'Shift-left'* testing for **firmware development**




Focusrite
Scarlett
Focusrite
Focusrite
Focusrite
Focusrite

Focusrite

Exhaustive Firmware Test Library

- Lots of test cases
- 70% **measure audio**

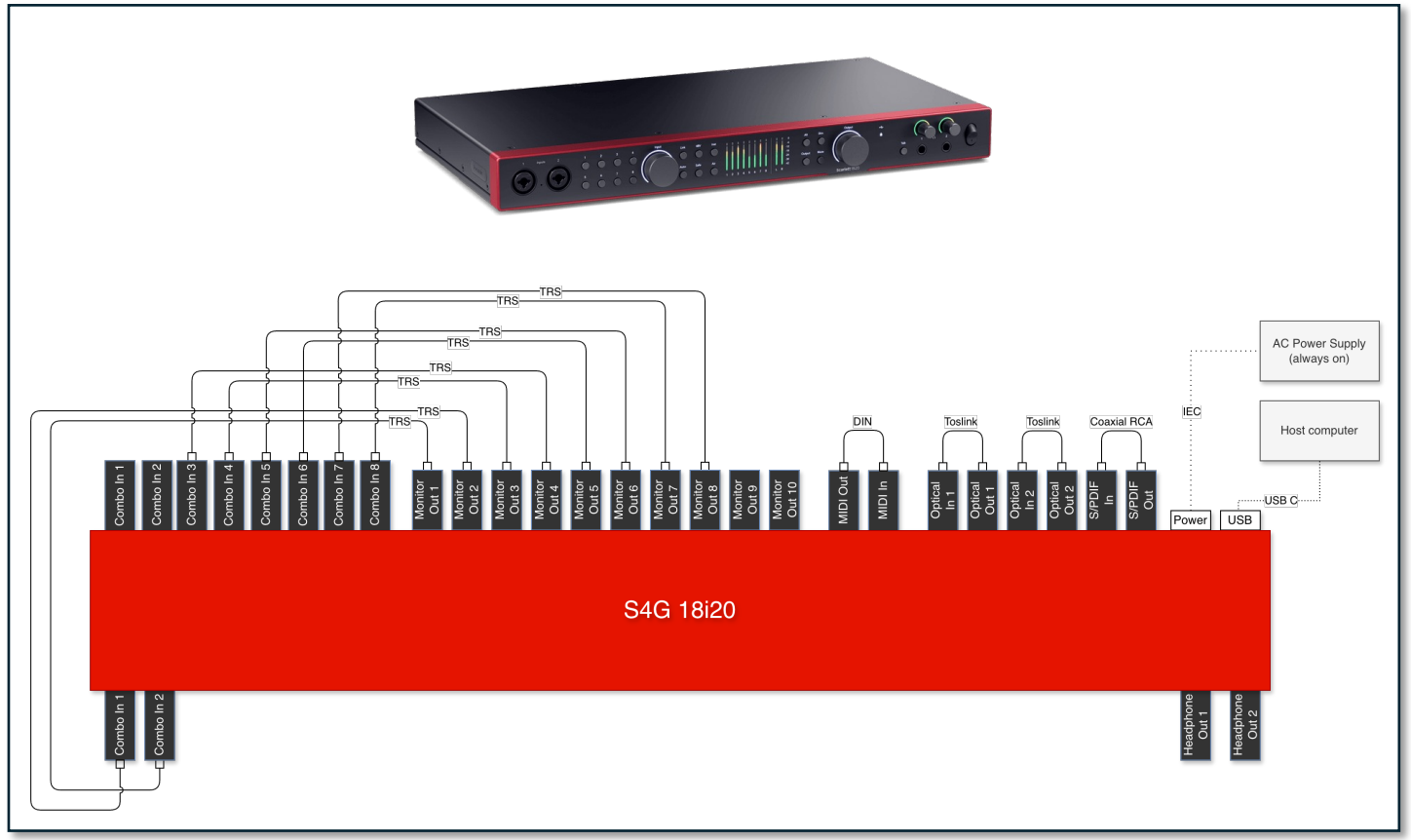
Feature	Description	Time	Audio?
OTA Updates	*Prove the firmware can be updated to and from over USB.*	00:01:47	✗
Factory-Default	*Prove the device can reboot after a factory-default command.*	00:01:00	✗
Easy Start Mode	*Validate the contents of the Easy Start drive.*	00:01:00	✗
MIDI	*Prove MIDI can be sent and received, check data and timing is correct.*	00:01:25	✗
Reset	*Prove the device can reboot with a command.*	00:00:30	✗
Basic Audio	*Validate audio quality across all I/O, at each sample rate. Measure distortion, RMS and RTL.*	00:03:00	✓
Glitch Test	*Prove the device can run for a long time without glitching.*	24:00:00	✓
External Clock Source	*Prove the device can be synced to an external clock source, rather than internal.*	00:03:00	✓
Phantom Power	*Stress toggling phantom power on each preamp. Validate the audio is temporarily muted.*	00:10:00	✓
Preamp Gain	*Stress changing each preamp gain. Validate the correct gain amount is applied to the signal.*	00:10:00	✓
Inst	*Stress toggling input type on each preamp. Validate the audio is temporarily muted.*	00:10:00	✓
Air	*Stress toggling each input's Air. Validate the level changes by the expected amount.*	00:10:00	✓
Routing	*Stress changing routes. Validate the signals appear on the correct channels.*	00:10:00	✓
Mixer	*Stress changing mixes. Validate the signals appear on the correct channels.*	00:10:00	✓
Monitor Control	*Stress changing monitor group features. Validate the levels change accordingly.*	00:10:00	✓

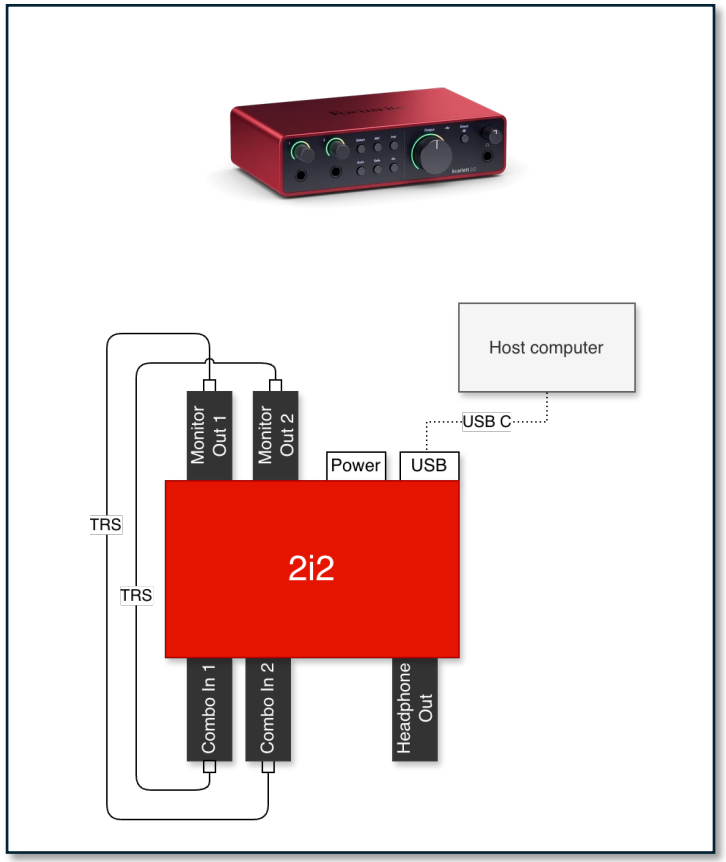
Test Approach

- Black Box
  - Control over USB protocol
    - Like **Focusrite Control 2**
  - Running ‘release’ firmware builds
    - Testing the **release candidate**
    - Building release confidence



Setup Diagrams





ISA C8X

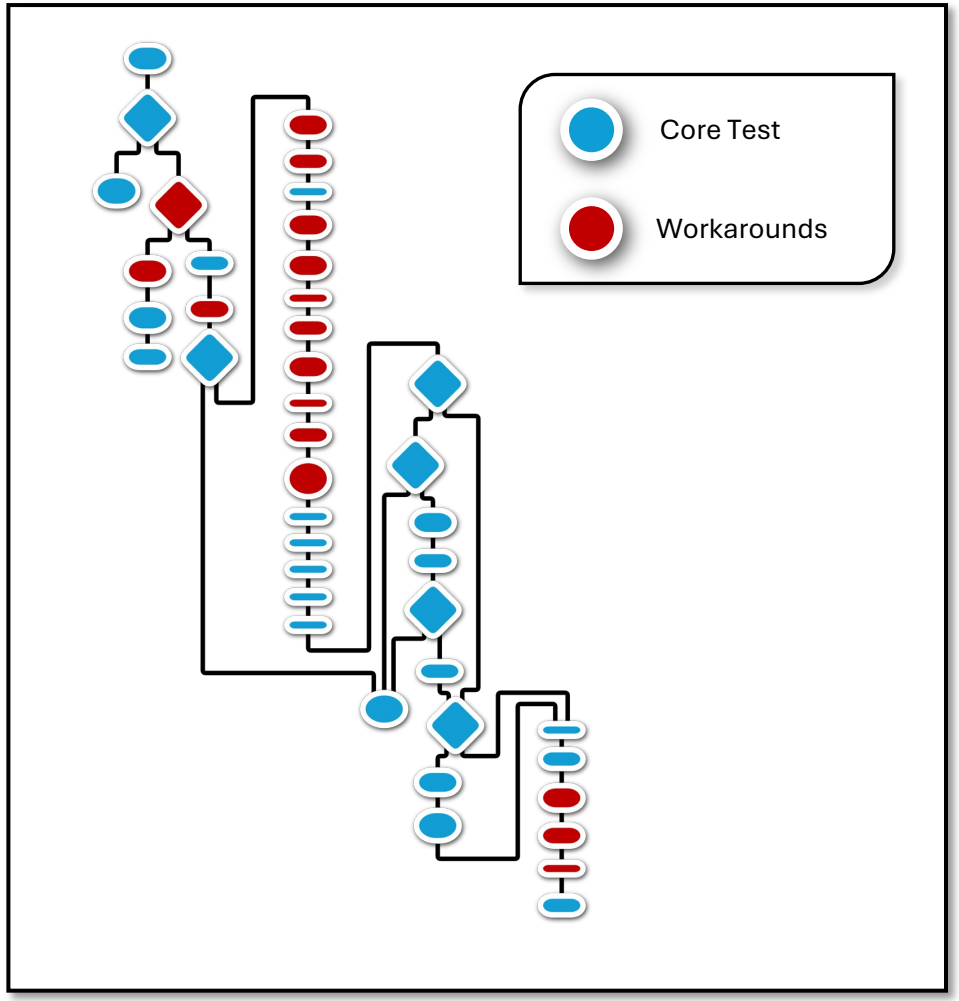


Could We Re-Use Our Tests?

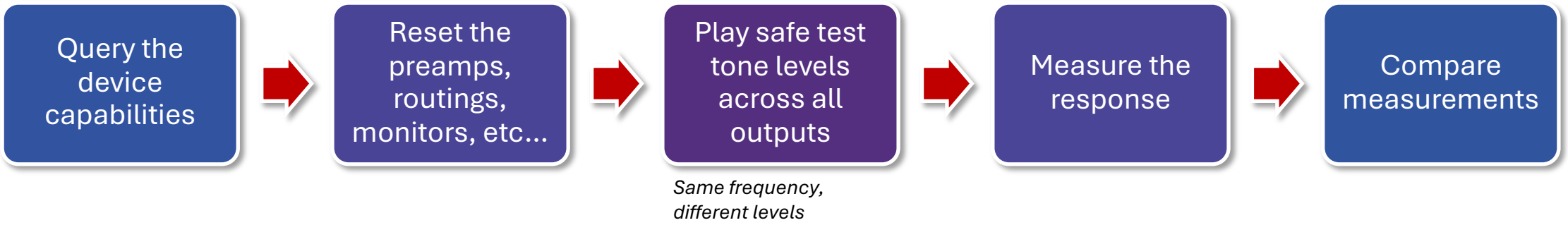
Basic Audio - Audit

- **3,300** lines of code
- **38%** ‘device-specific’ code
  - In-line **workarounds**
- **17** ‘magic numbers’
  - Expected measurements
  - Tolerances
  - Channel indexes

Hard to maintain!



Basic Audio Test Flow



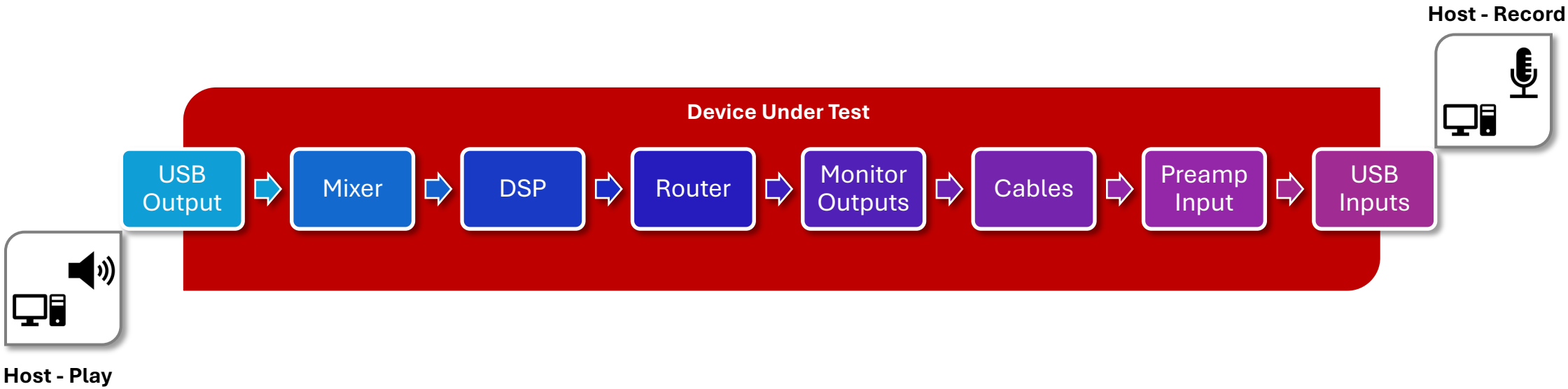
What are the device capabilities?

How to reset the preamps, etc?

What are **safe** test tone *levels*?

What are **expected input** *levels*?

Audio Interfaces



Given the current state, what signal response should I observe?

Test Requirements

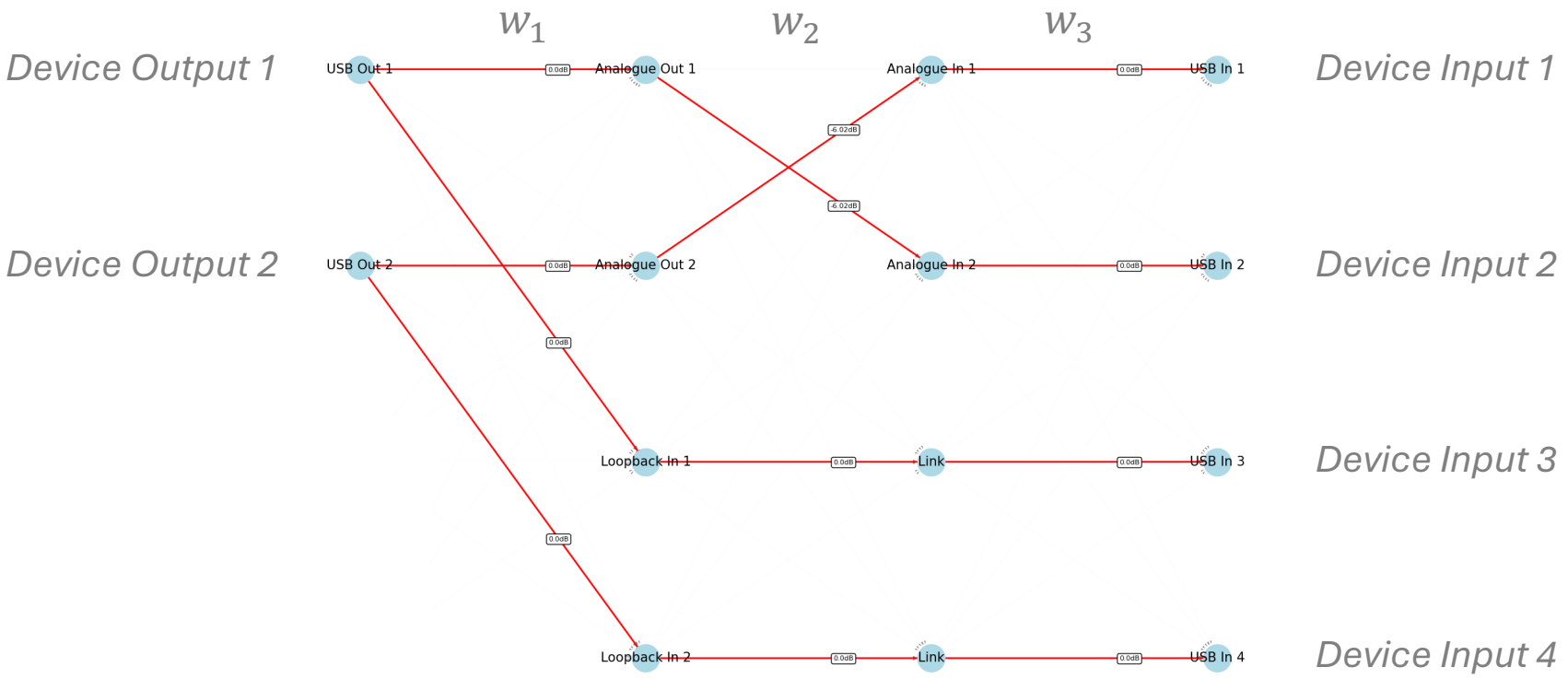
***What** can it do?*

***How** does it do it?*

What's the ***response?***



Network Model V1



Scarlett 2i2 4th Gen

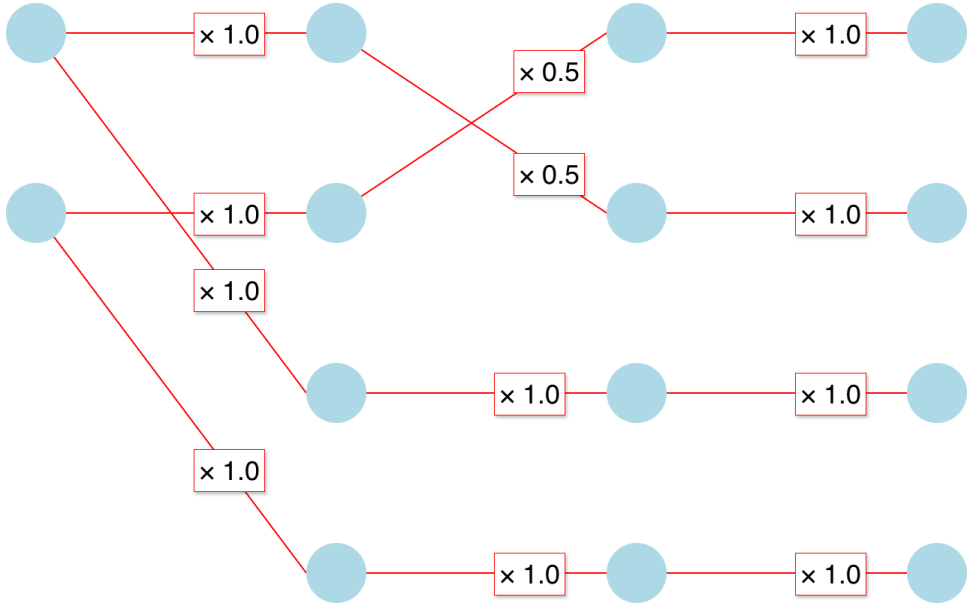
Signals propagate from left to right

Only weights greater than 0.0 shown

Simulating the Model

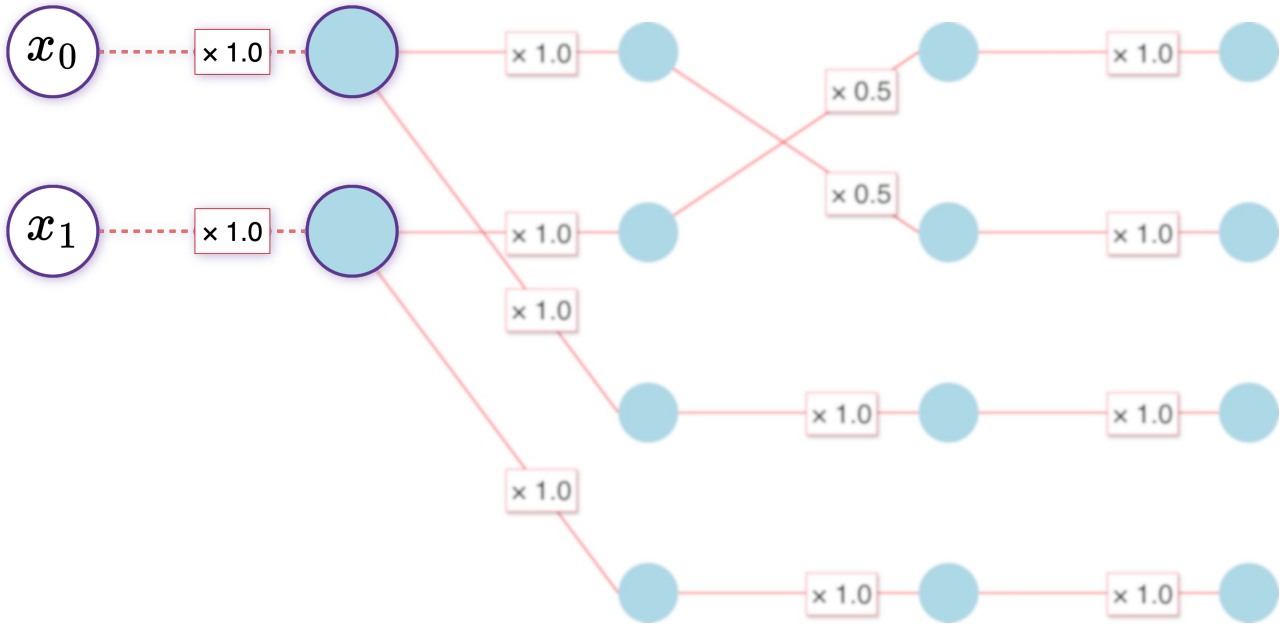
- Directionality
  - Feed-Forward networks are made of **adjacent layers**
  - **Sequential** matrix multiplication
- We’re modelling **signal levels** only
- Between each layer is a **vector of scalars**
  - 0.0 = mute
  - 1.0 = unity gain
  - 0.5 ~= -6dB

Simulating the Model



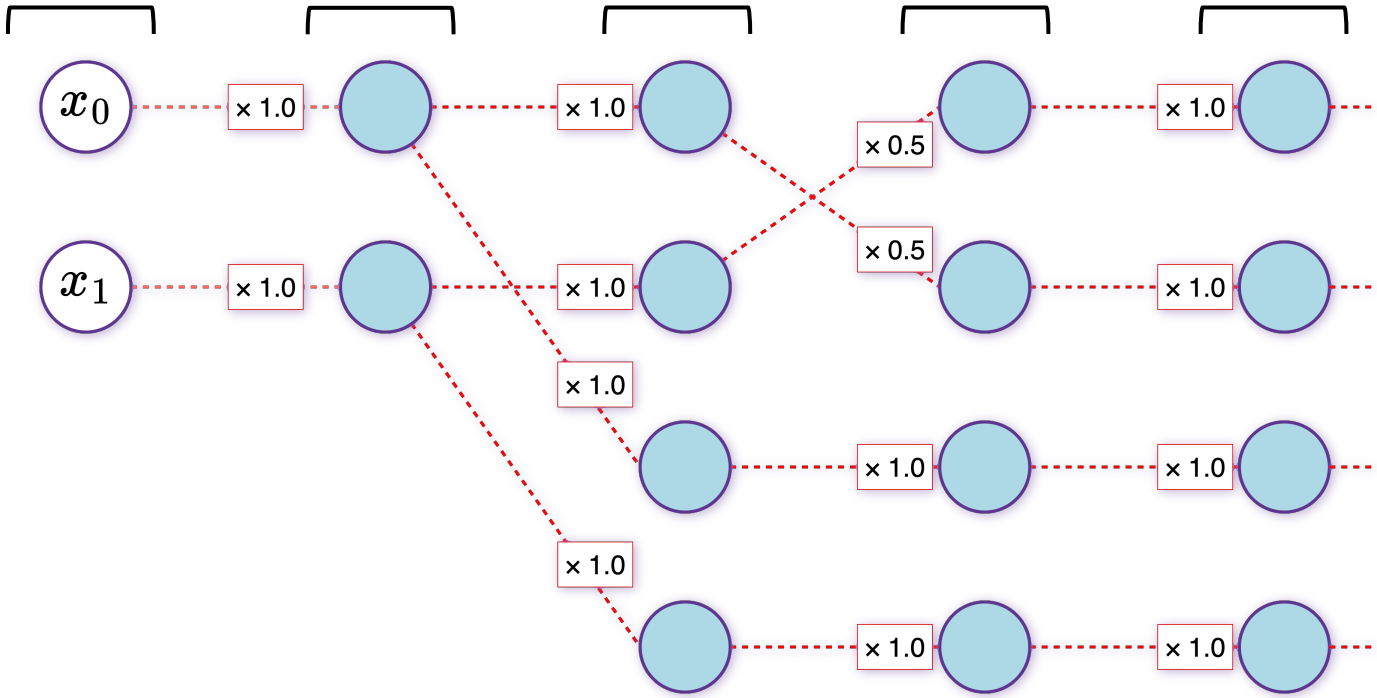
Simulating the Model

1. **Prepend** the network with input signal layer



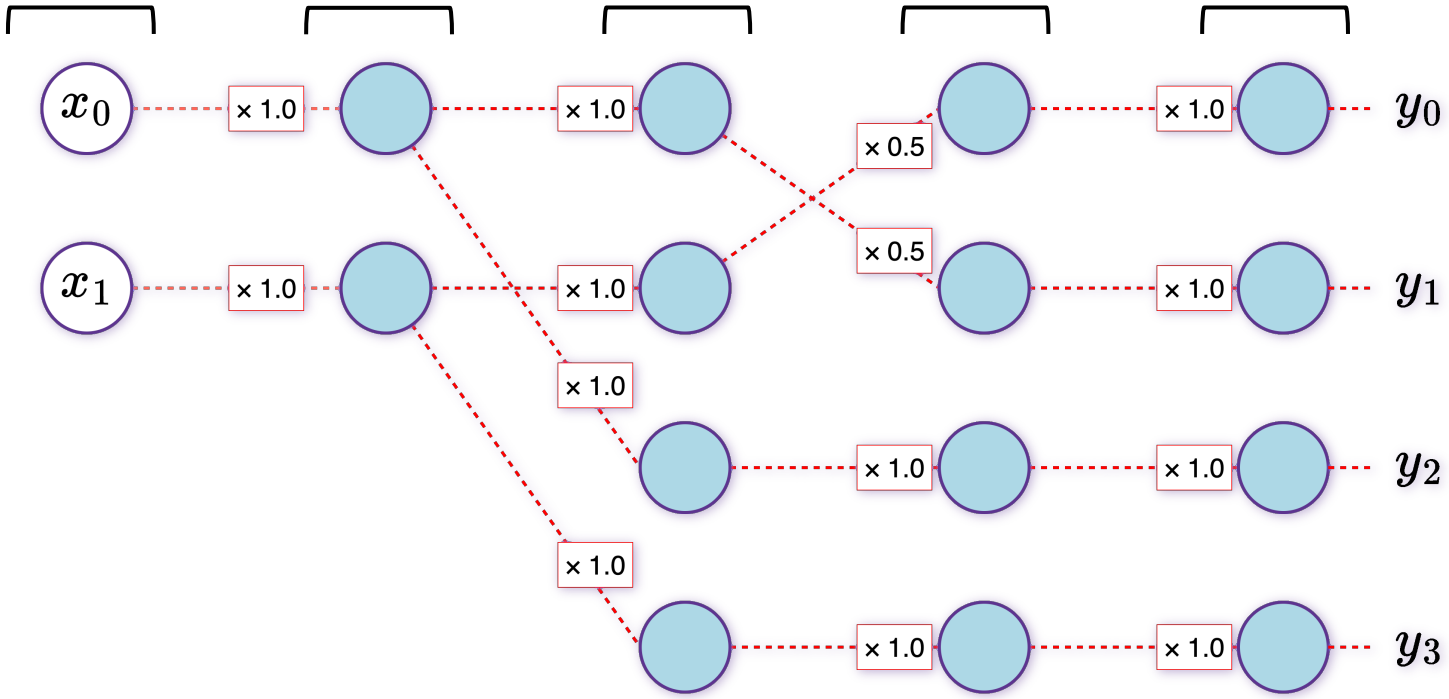
Simulating the Model

1. **Prepend** the network with input signal layer
2. Successively multiply each layer with **matrix multiplication**
   - Signal attenuation is a **linear scalar**, not decibels
   - Signals **accumulate** across connected paths and layers



Simulating the Model

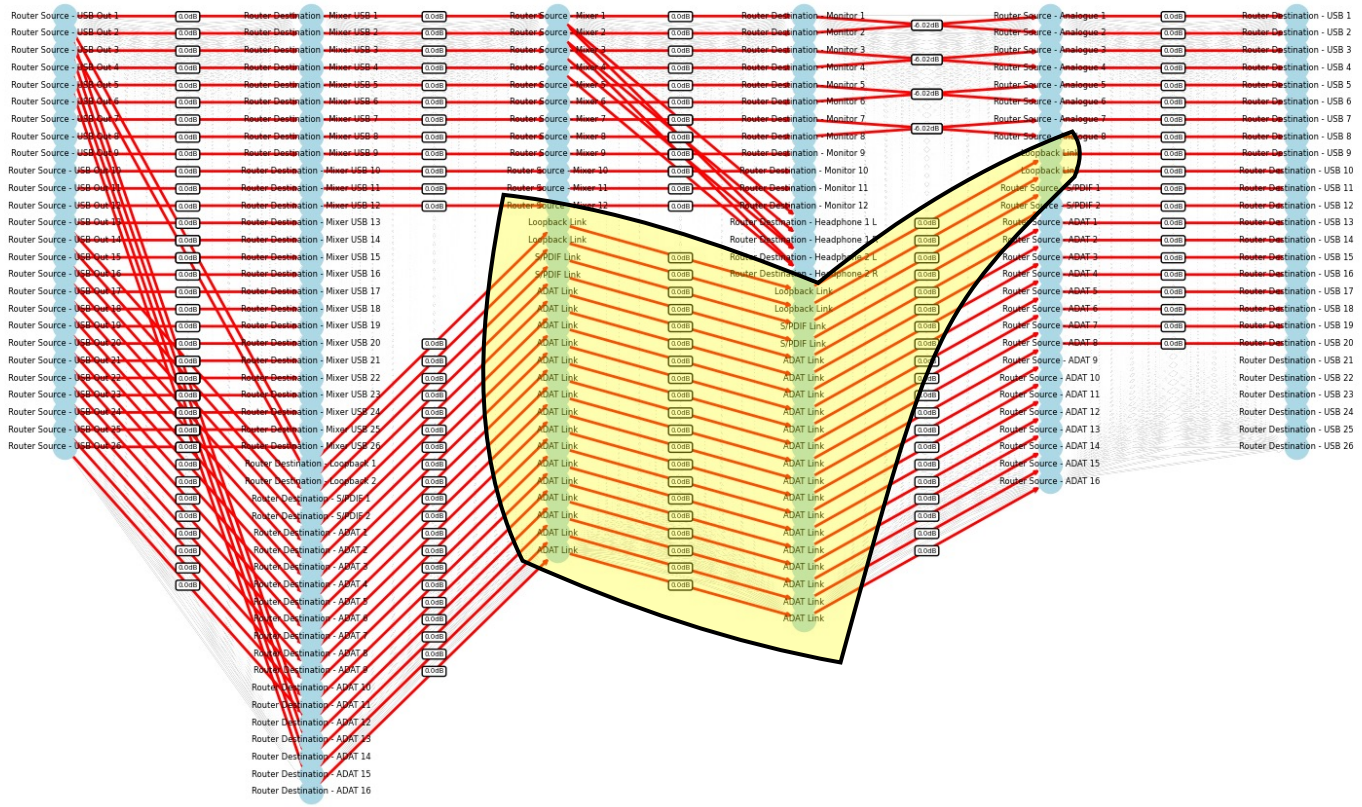
1. **Prepend** the network with input signal layer
2. Successively multiply each layer with **matrix multiplication**
    - Signal attenuation is a **linear scalar**, not decibels
    - Signals **accumulate** across connected paths and layers
- The final matrix multiplication at the last ‘output’ layer gives us the output values
    - Convert back to dB



Network Model Benefits

- ✓ Default state of the DUT is simulated deterministically
- ✓ Graphical debugging
  - ✓ Quickly understand expected device behaviour
- ✓ Reduced complexity in the test script
  - ✓ *Basic Audio* in-line test script was **3,300 lines**
  - ✓ After adding the model, it is **200 lines**
    - ✓ No magic numbers and no in-line device-specific workarounds

Larger Model



ISA C8X

‘Workaround’ nodes highlighted in yellow

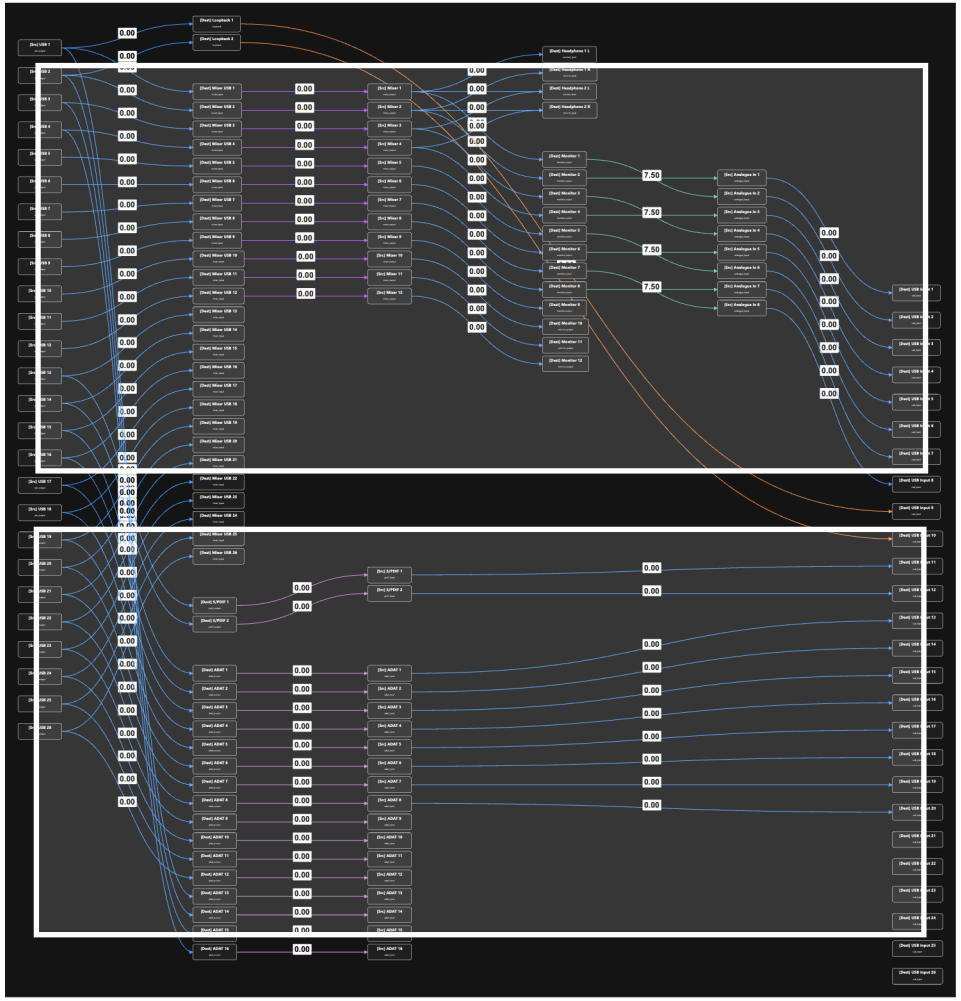
42 ‘workaround’ nodes

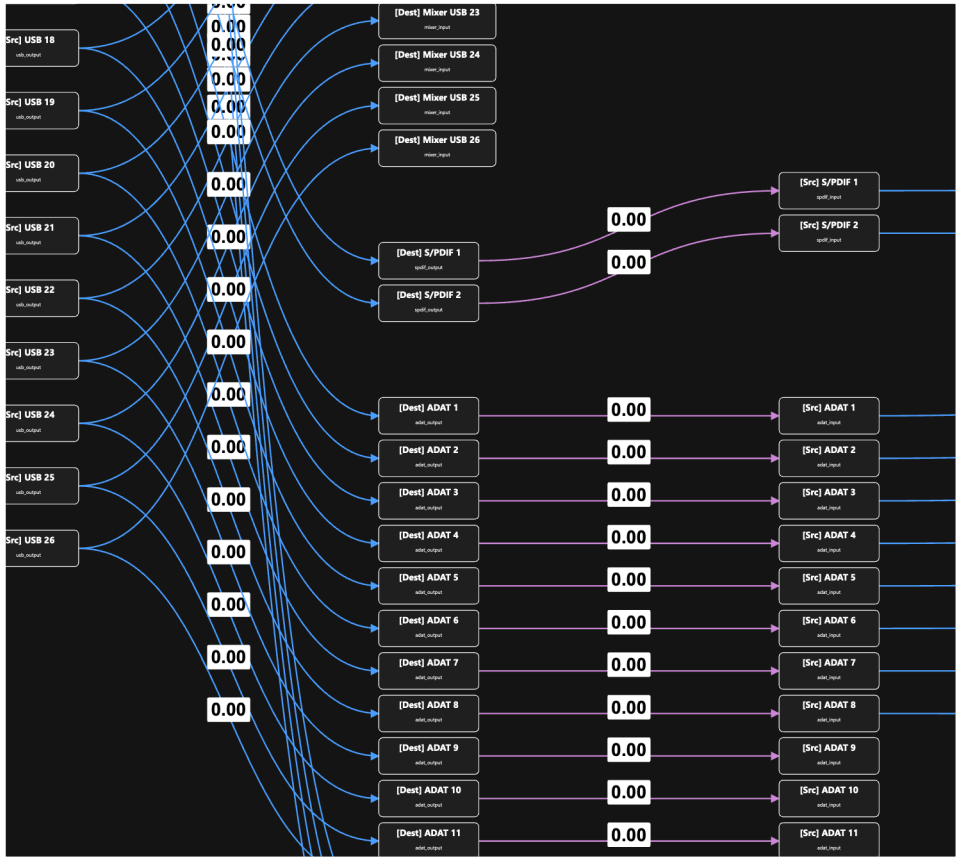
134 meaningful nodes

Feed Forward Network Limitations

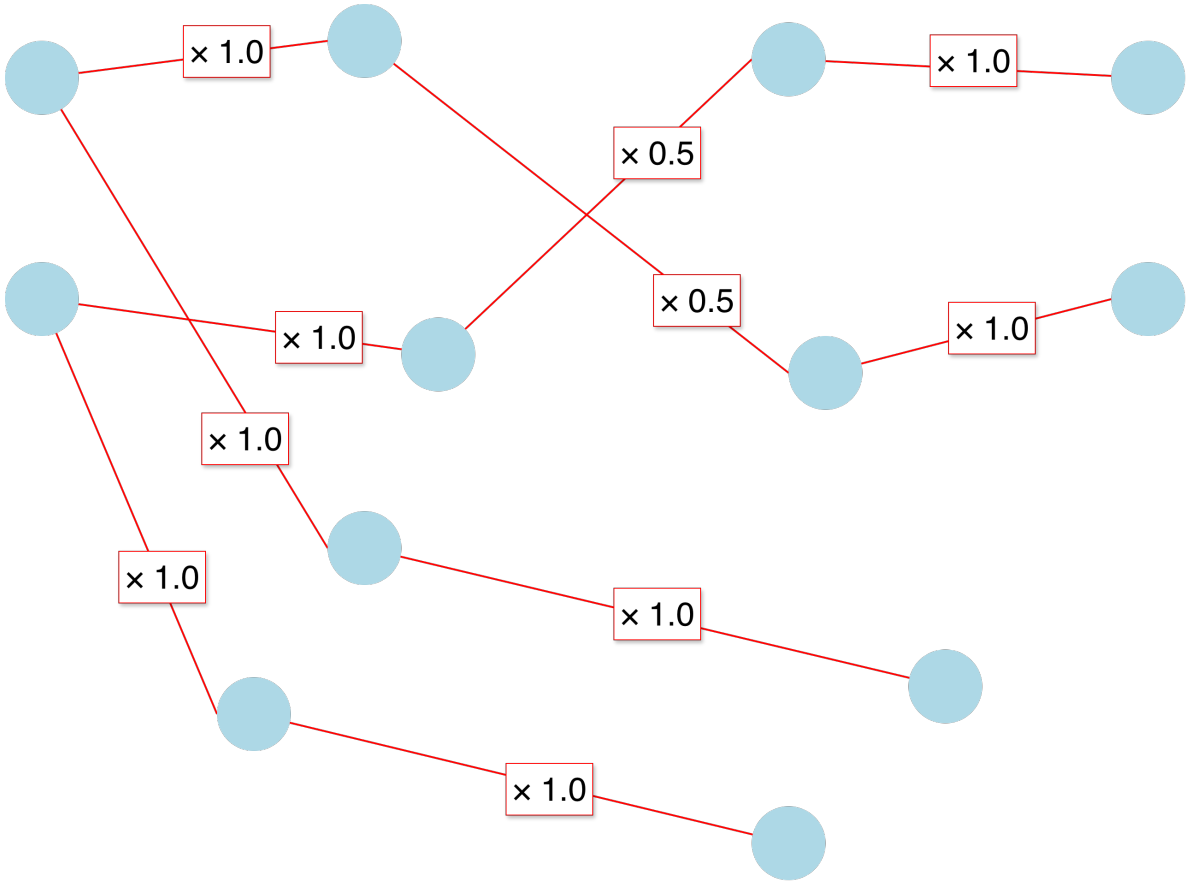
- **Layers** impose limits on configurations
  - Only **adjacent** nodes can be connected
  - ‘Workaround’ nodes are needed to propagate signals through unnecessary layers
- As complexity grows, the number of ‘workaround’ nodes becomes unmaintainable

Graph Model V2



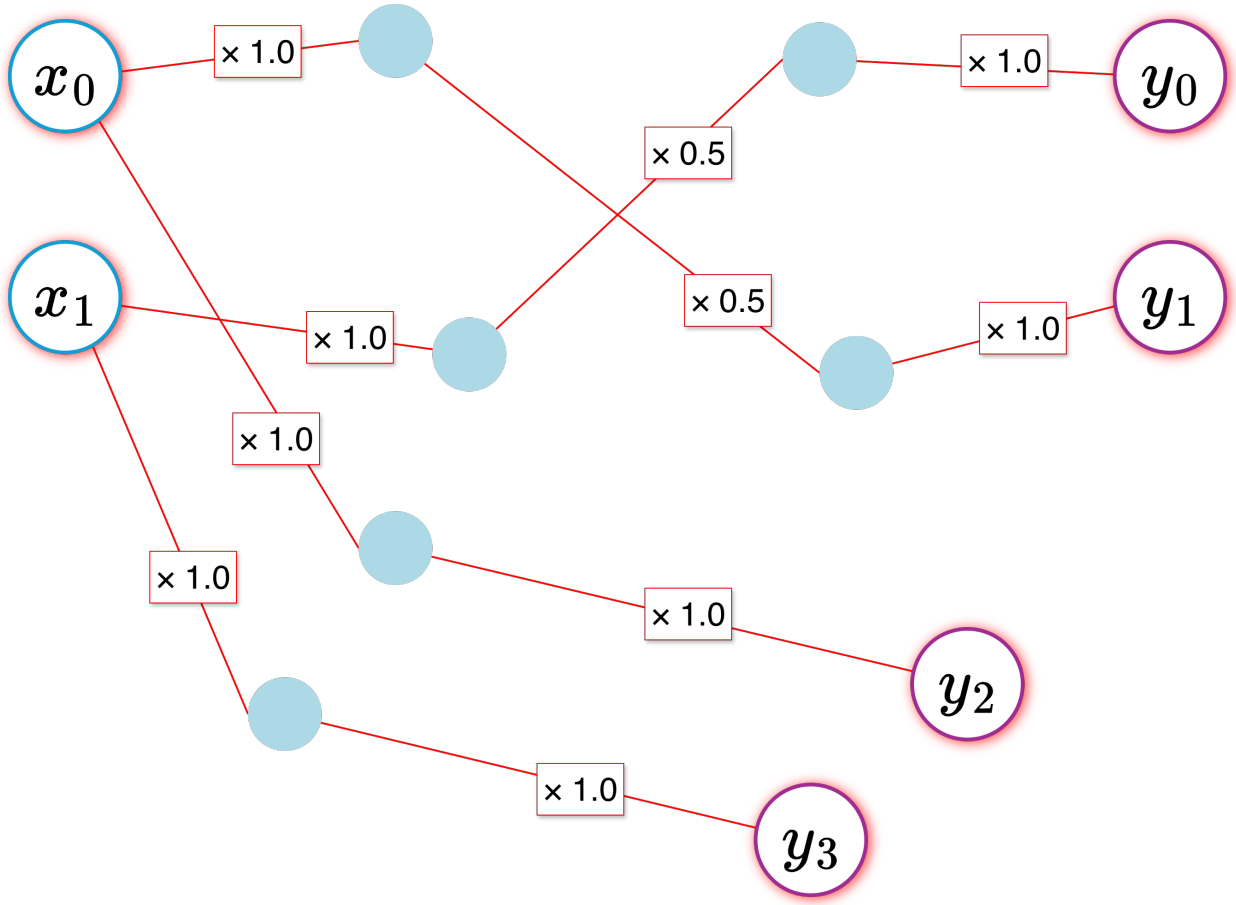


Processing the Network (V2)



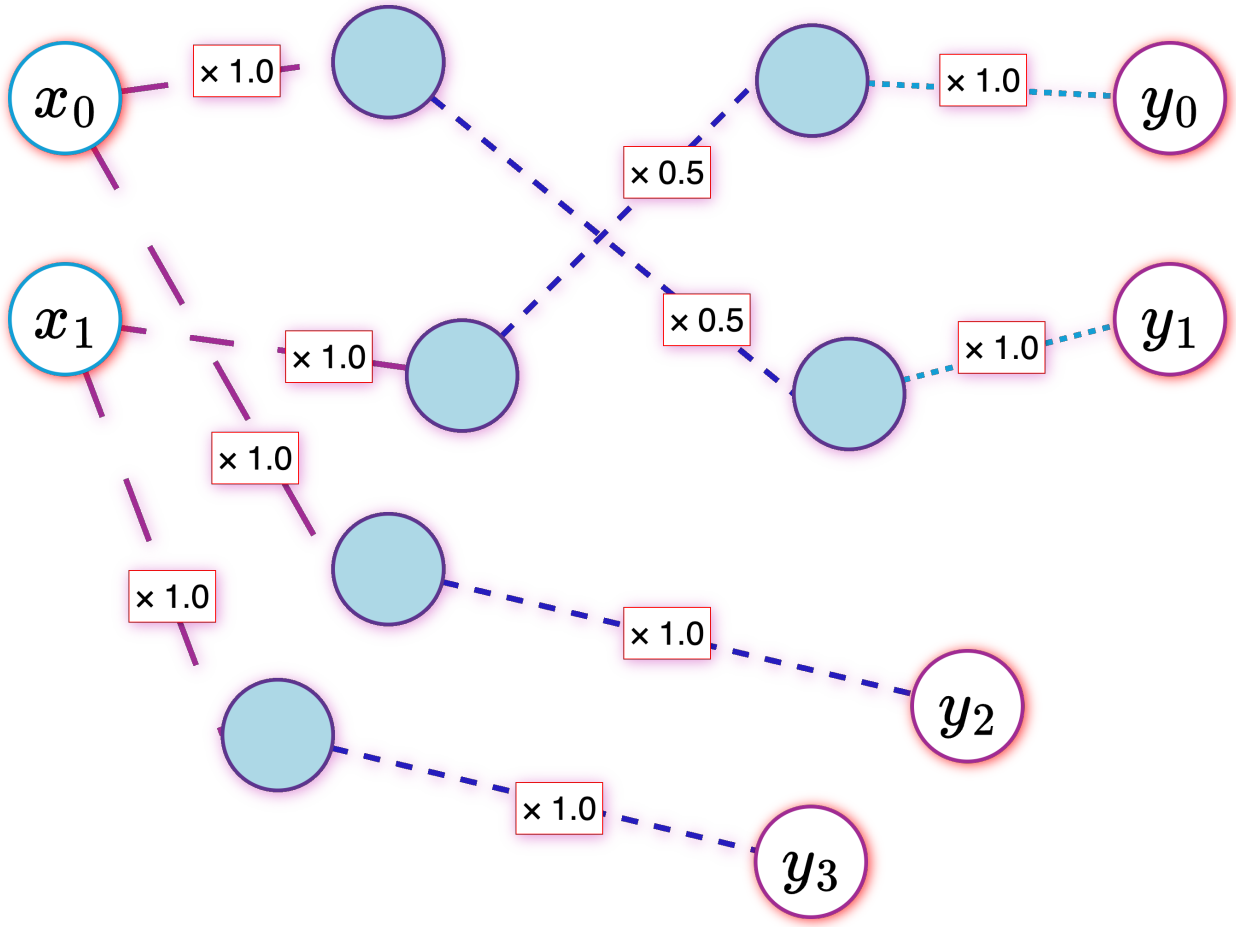
Processing the Network (V2)

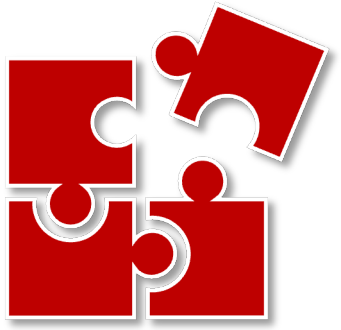
1. Select the **input and output** nodes
   - Convert the input levels to **magnitudes**
   - Place inputs at starting nodes x



Processing the Network (V2)

1. Select the **input and output** nodes
   - Convert the input levels to **magnitudes**
   - Place inputs at starting nodes x
2. Iteratively propagate signals through the network
   1. Scan all nodes, propagating each new signal contribution to the downstream node(s), multiplied by the gain scalar
      - If **multiple paths** lead to the **same node**, their contributions are summed together
   2. Repeat until propagation stops, or iteration limit is reached
3. Values at y now represent the response
   - Convert back to dB



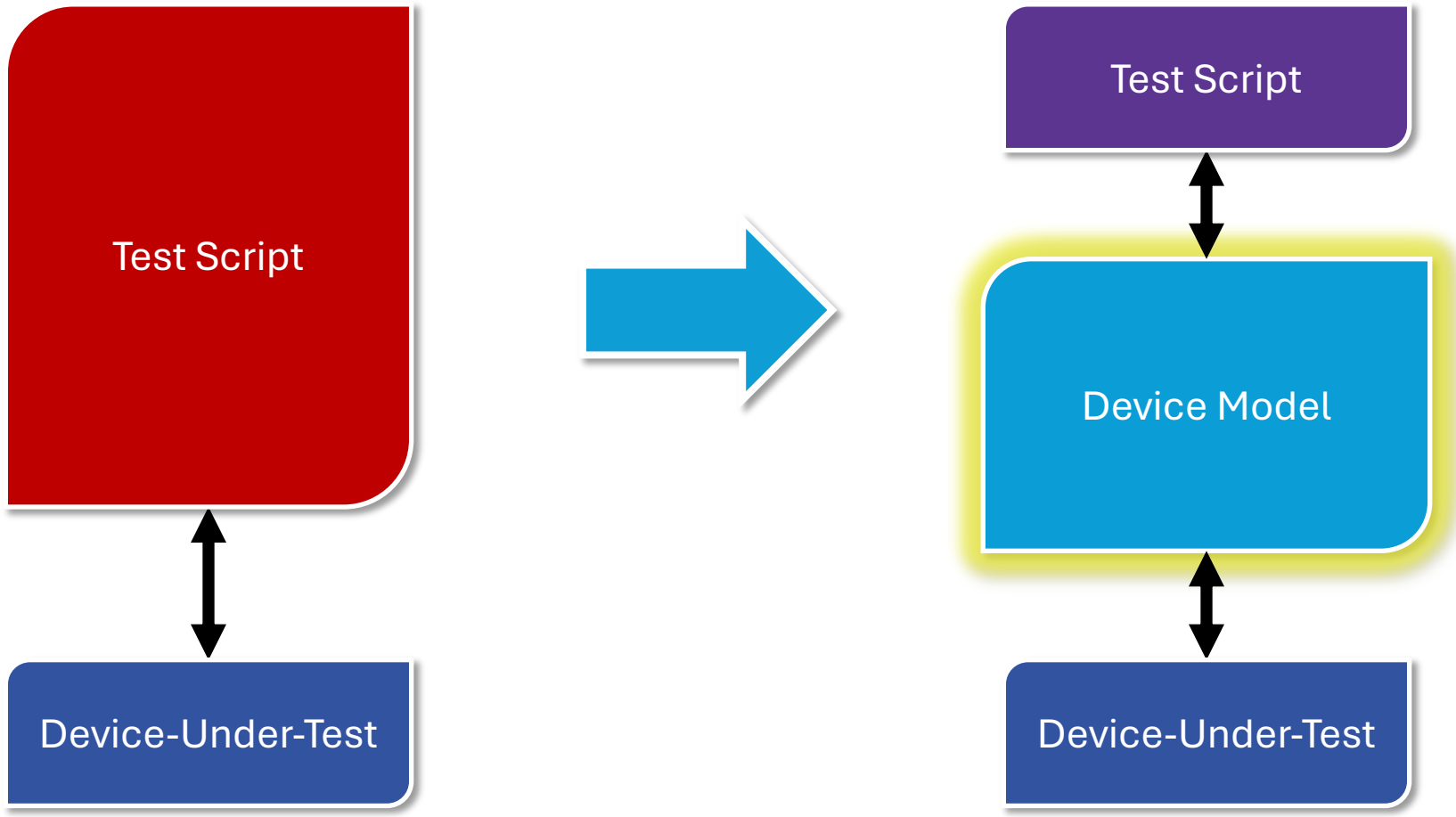


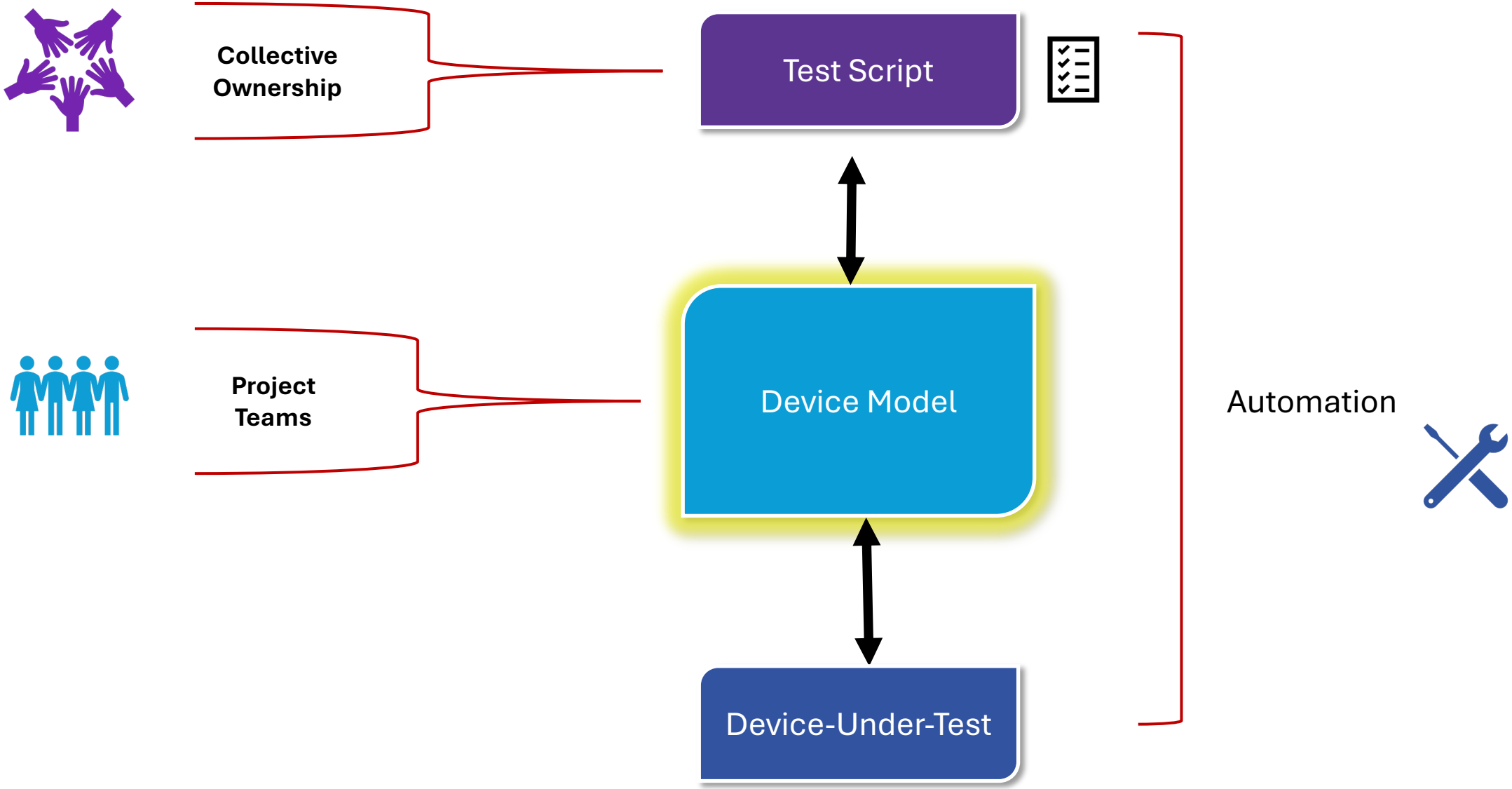
Model-Test Interface

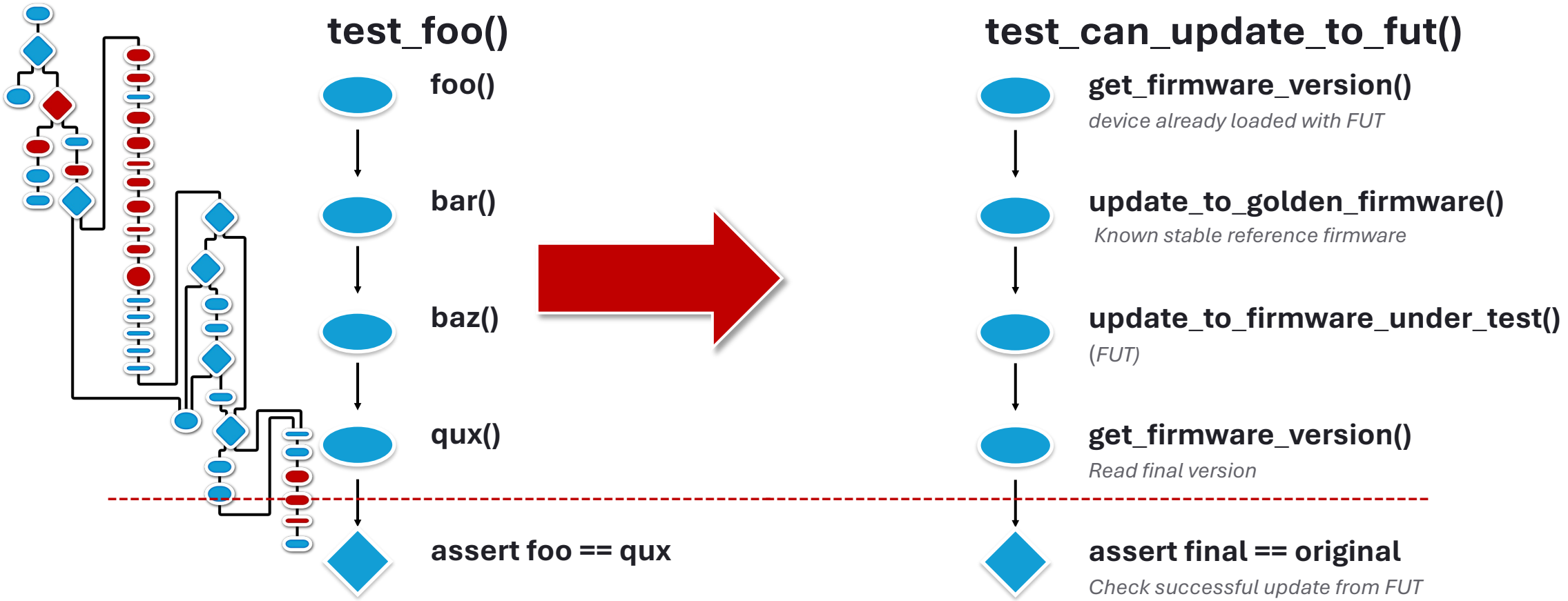
Architecture

Focusrite

Abstraction Layer




Collective Ownership
Project Teams
Test Script
Device Model
Device-Under-Test
Automation


test_foo()
foo()
bar()
baz()
qux()
assert foo == qux
test_can_update_to_fut()
get_firmware_version()
device already loaded with FUT
update_to_golden_firmware()
Known stable reference firmware
update_to_firmware_under_test()
(FUT)
get_firmware_version()
Read final version
assert final == original
Check successful update from FUT

Model-Test Contract

Test

- test_foo():
  - bar()
  - assert bar()

Framework: Abstract base class

```
Class Baz(ABC):
    def bar(self, *args, **kwargs):
        ...
Def
```

Project model

```
Class MyDevice(Bar):
    def bar(self, *args, **kwargs):
        ...
```

Easy model creation

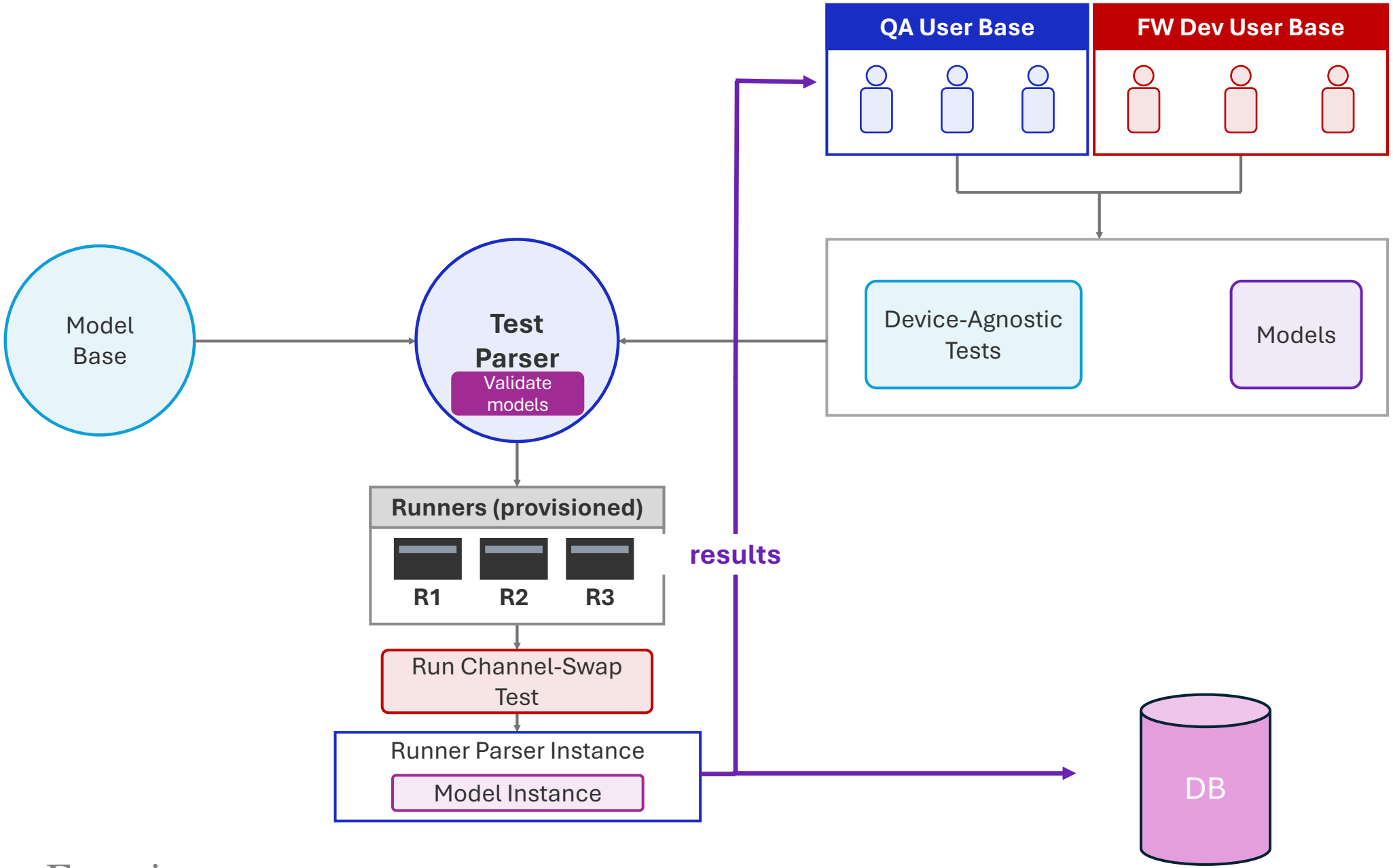


Framework: Abstract base class

```
Class Baz(ABC):
    def bar(self, *args, **kwargs):
        ...
```

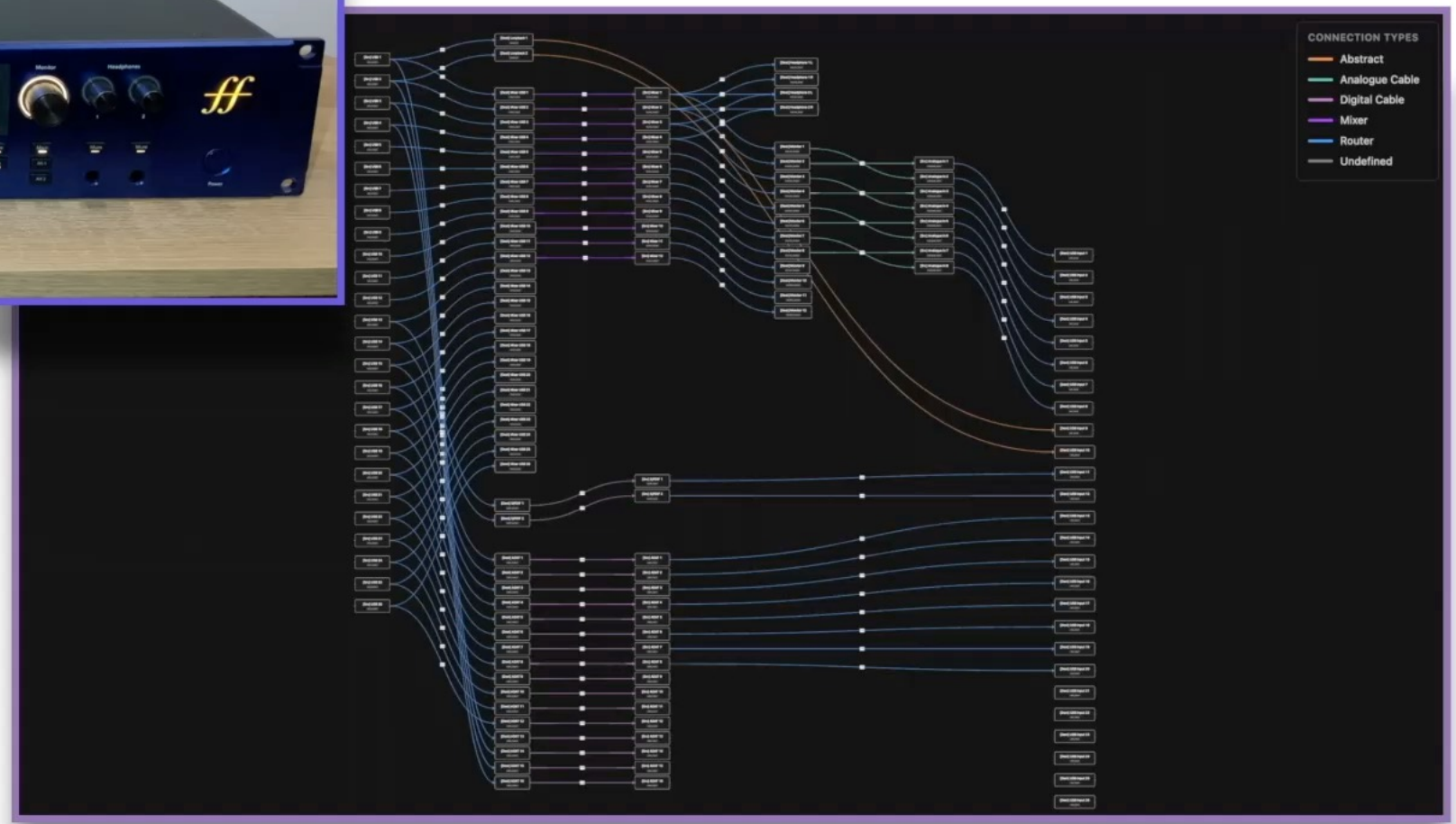
Project model

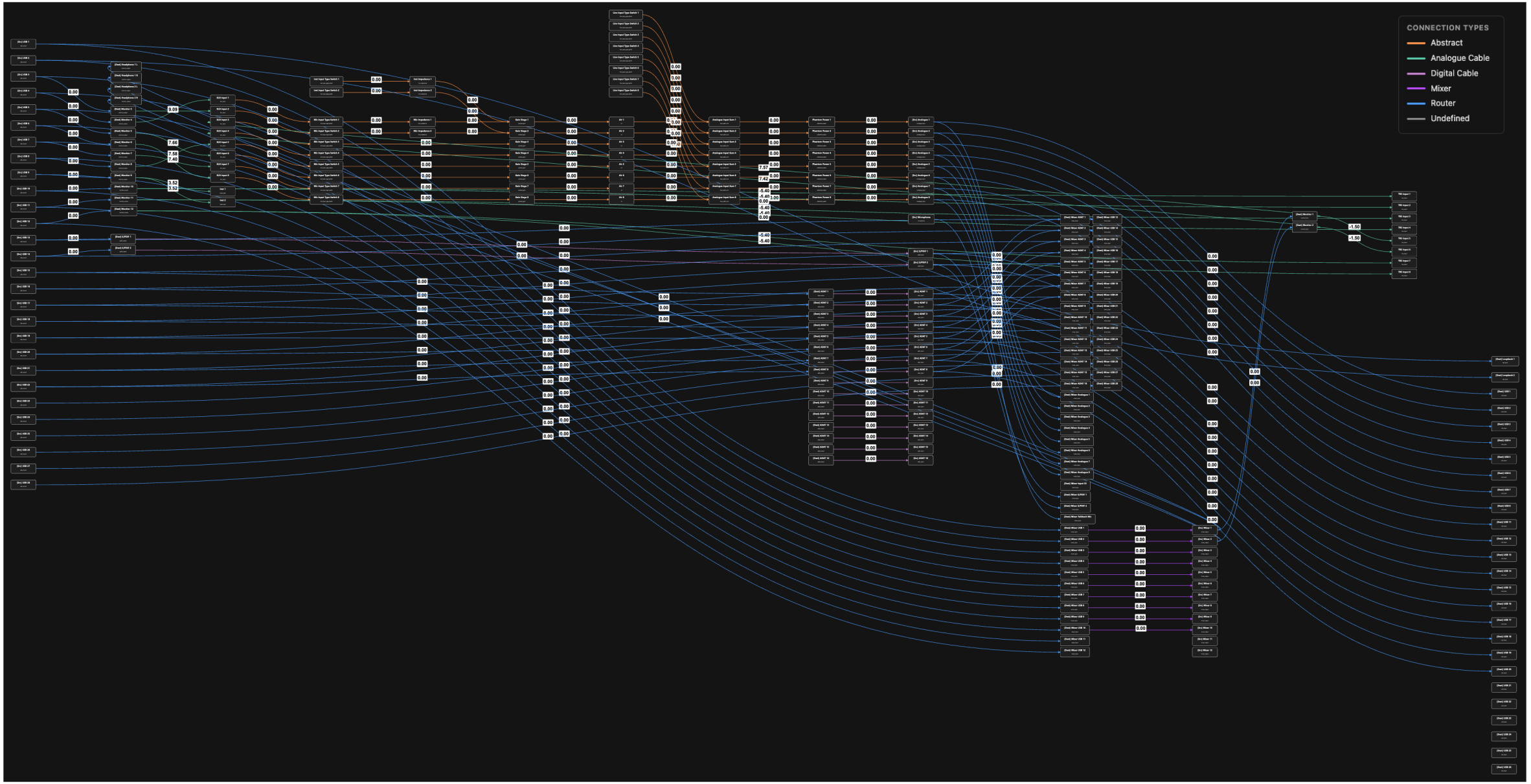
```
Class MyDevice(Bar):
    def bar(self, *args, **kwargs):
        self.do_x(threshold=MY_THRESHOLD)
```


QA User Base
FW Dev User Base
Model Base
Test Parser
Validate models
Device-Agnostic Tests
Models
Runners (provisioned)
R1
R2
R3
results
Run Channel-Swap Test
Runner Parser Instance
Model Instance
DB

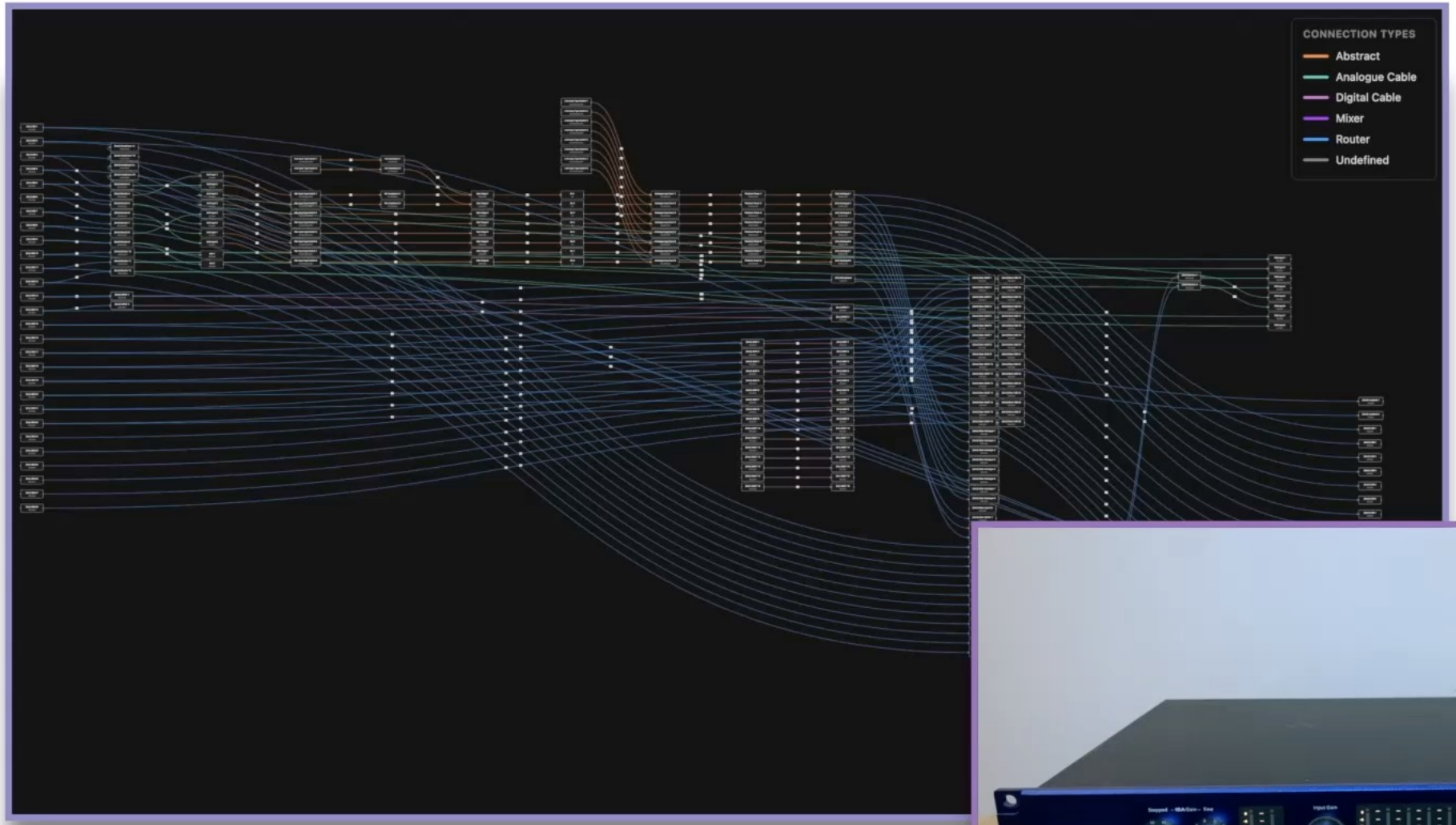
Demo


ISA C8X
Focusrite


CONNECTION TYPES
Abstract
Analogue Cable
Digital Cable
Mixer
Router
Undefined

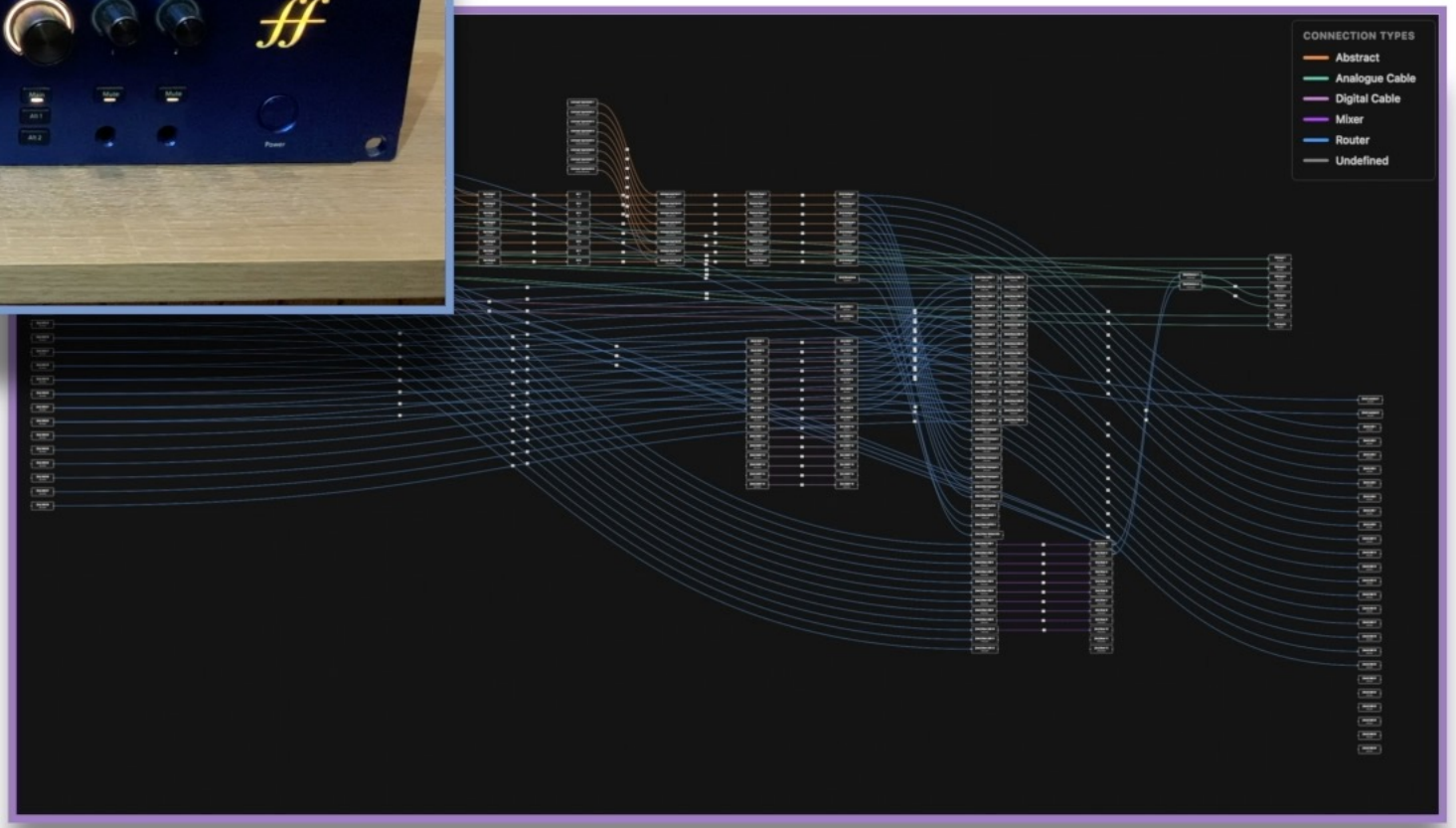

CONNECTION TYPES
Abstract
Analogue Cable
Digital Cable
Mixer
Router
Undefined

Focusrite


CONNECTION TYPES
Abstract
Analogue Cable
Digital Cable
Mixer
Router
Undefined


ISA C8X
Focusrite
ff


ISA C8X
Focusrite
Stepped
ISA Gain
Fine
Input Gain
Monitor
Headphones
ff
ISA1
ISA2
Power


CONNECTION TYPES
Abstract
Analogue Cable
Digital Cable
Mixer
Router
Undefined

```
E           AssertionError: RMS level mismatch (dB)
E             Channel 1: measured=-116.84 dB, expected=-13.13 dB
E           assert [-116.84, -20..., -13.63, ...] == approx([-13.1...-120.0 ± 1.0])
E
E             comparison failed. Mismatched elements: 1 / 26:
E             Max absolute difference: 103.7123313333797
E             Max relative difference: 0.8876440545479262
E             Index | Obtained | Expected
E             0     | -116.84  | -13.127668666620307 ± 1.0
```

Closing Remarks

- What we have:
  - Simulation model
  - Re-usable tests
  - Infrastructure tooling

- Future improvements:
  - Frequency domain modelling
  - Test case parameterisation
    - Automatic test case generation

Thank you

Q&A

Previous Talks

Focusrite's Hardware Test Platform - Testing Thousands of Little Red Boxes Every Day

Dave Curtis & Adrien Fauconnet - ADC23

Automating Audio Device Testing - A Brief Existential Guide

Joseph Deller & Jake Wignall - ADC22