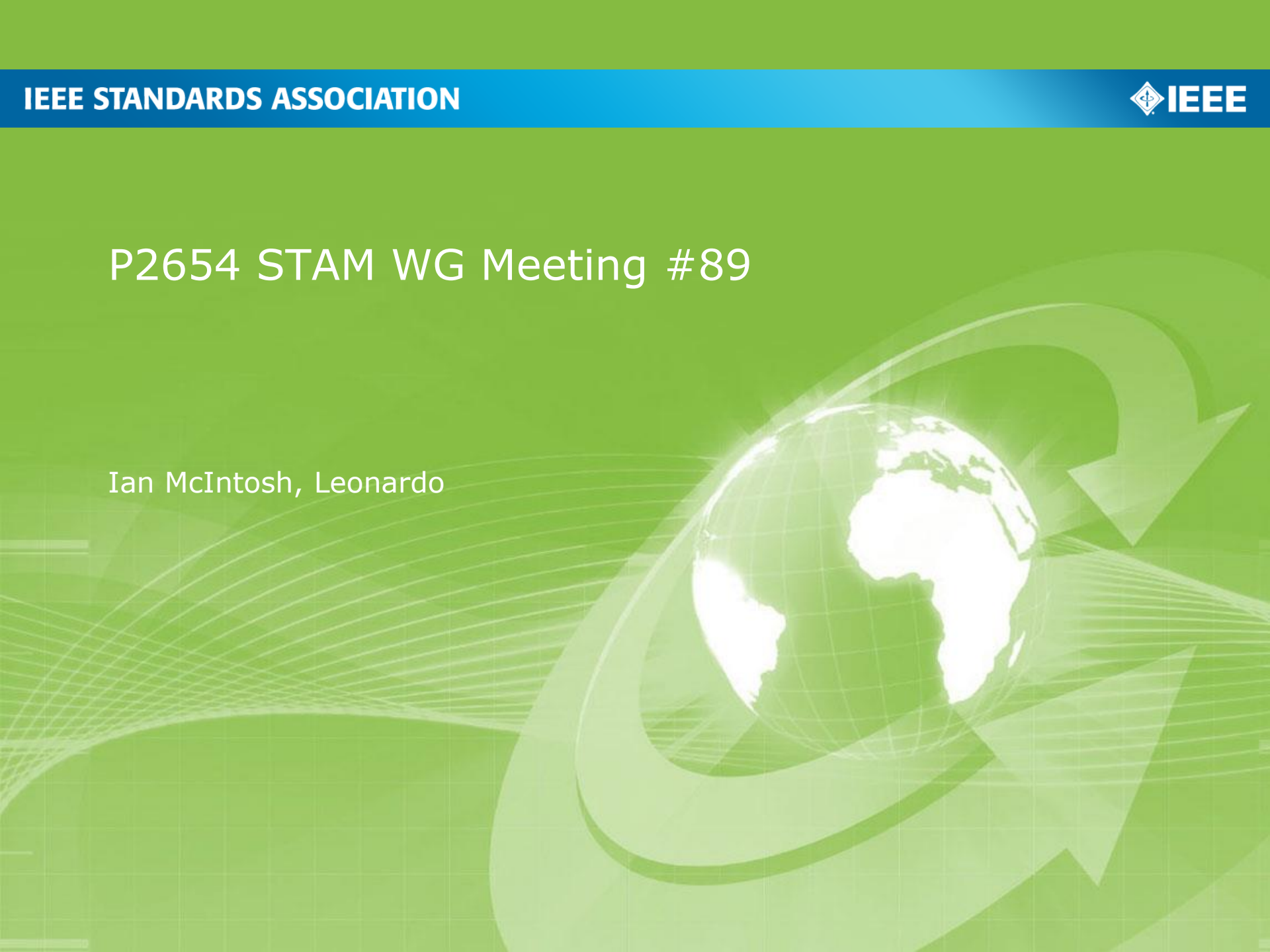
P2654 STAM WG Meeting #89

Ian McIntosh, Leonardo



Compliance with IEEE Standards Policies and Procedures

Subclause 5.2.1 of the *IEEE-SA Standards Board Bylaws* states, "While participating in IEEE standards development activities, all participants…shall act in accordance with all applicable laws (nation-based and international), the IEEE Code of Ethics, and with IEEE Standards policies and procedures."

The contributor acknowledges and accepts that this contribution is subject to

- The IEEE Standards copyright policy as stated in the *IEEE-SA Standards Board Bylaws*, section 7, http://standards.ieee.org/develop/policies/bylaws/sect6-7.html#7, and the *IEEE-SA Standards Board Operations Manual*, section 6.1, http://standards.ieee.org/develop/policies/opman/sect6.html
- The IEEE Standards patent policy as stated in the *IEEE-SA Standards Board Bylaws*, section 6, http://standards.ieee.org/guides/bylaws/sect6-7.html#6, and the *IEEE-SA Standards Board Operations Manual*, section 6.3, http://standards.ieee.org/develop/policies/opman/sect6.html

IEEE P2654
System Test Access Management
Ian McIntosh (chair)

Working Group Meeting #89			
Date: 2020-12-07			
Author(s):			
Name	**Affiliation**	**Phone [optional]**	**Email [optional]**
Ian McIntosh	Leonardo		

2. Agenda

1. Roll Call
2. Agenda
3. IEEE Patent and Copyright Slides
4. Review and approve previous minutes: November 30
5. Review open action items
6. Inter-group Collaboration
7. Discussion Topics:
    a. Update on WG elections
    b. Review standard outline and figures - see if there are any "quick wins"
8. Any other business
9. Key Takeaways from today's meeting
10. Glossary terms from this meeting
11. Schedule next meeting
12. Topic for next meeting
13. Reminders
14. List new action items
15. Adjourn

Participants have a duty to inform the IEEE

- Participants shall inform the IEEE (or cause the IEEE to be informed) of the identity of each holder of any potential Essential Patent Claims of which they are personally aware if the claims are owned or controlled by the participant or the entity the participant is from, employed by, or otherwise represents

- Participants should inform the IEEE (or cause the IEEE to be informed) of the identity of any other holders of potential Essential Patent Claims

Early identification of holders of potential Essential Patent Claims is encouraged

Ways to inform IEEE

- **Cause an LOA to be submitted to the IEEE-SA (patcom@ieee.org); or**
- **Provide the chair of this group with the identity of the holder(s) of any and all such claims as soon as possible; or**
- **Speak up now and respond to this Call for Potentially Essential Patents**

If anyone in this meeting is personally aware of the holder of any patent claims that are potentially essential to implementation of the proposed standard(s) under consideration by this group and that are not already the subject of an Accepted Letter of Assurance, please respond at this time by providing relevant information to the WG Chair

Other guidelines for IEEE WG meetings

- All IEEE-SA standards meetings shall be conducted in compliance with all applicable laws, including antitrust and competition laws.
  - Don't discuss the interpretation, validity, or essentiality of patents/patent claims.
  - Don't discuss specific license rates, terms, or conditions.
    - Relative costs of different technical approaches that include relative costs of patent licensing terms may be discussed in standards development meetings.
      - Technical considerations remain the primary focus
  - Don't discuss or engage in the fixing of product prices, allocation of customers, or division of sales markets.
  - Don't discuss the status or substance of ongoing or threatened litigation.
  - Don't be silent if inappropriate topics are discussed … do formally object.

For more details, see *IEEE-SA Standards Board Operations Manual*, clause 5.3.10 and *Antitrust and Competition Policy: What You Need to Know* at http://standards.ieee.org/develop/policies/antitrust.pdf

Patent-related information

The patent policy and the procedures used to execute that policy are documented in the:

- *IEEE-SA Standards Board Bylaws*
  (http://standards.ieee.org/develop/policies/bylaws/sect6-7.html#6)
- *IEEE-SA Standards Board Operations Manual*
  (http://standards.ieee.org/develop/policies/opman/sect6.html#6.3)

Material about the patent policy is available at
http://standards.ieee.org/about/sasb/patcom/materials.html

If you have questions, contact the IEEE-SA Standards Board Patent Committee Administrator at patcom@ieee.org

IEEE-SA Copyright Policy

By participating in this activity, you agree to comply with the IEEE Code of Ethics, all applicable laws, and all IEEE policies and procedures including, but not limited to, the IEEE-SA Copyright Policy.

- Previously Published material (copyright assertion indicated) shall not be presented/submitted to the Working Group nor incorporated into a Working Group draft unless permission is granted.

  Prior to presentation or submission, you shall notify the Working Group Chair of previously Published material and should assist the Chair in obtaining copyright permission acceptable to IEEE-SA.
- For material that is not previously Published, IEEE is automatically granted a license to use any material that is presented or submitted.

IEEE-SA Copyright Policy

- The **IEEE-SA Copyright Policy** is described in the *IEEE-SA Standards Board Bylaws* and *IEEE-SA Standards Board Operations Manual*
  - IEEE-SA Copyright Policy, see
    Clause 7 of the *IEEE-SA Standards Board Bylaws*
    https://standards.ieee.org/about/policies/bylaws/sect6-7.html#7
    Clause 6.1 of the *IEEE-SA Standards Board Operations Manual*
    https://standards.ieee.org/about/policies/opman/sect6.html

- IEEE-SA Copyright Permission
  - https://standards.ieee.org/content/dam/ieee-standards/standards/web/documents/other/permissionltrs.zip
- IEEE-SA Copyright FAQs
  - http://standards.ieee.org/faqs/copyrights.html/

- IEEE-SA Best Practices for IEEE Standards Development
  - http://standards.ieee.org/develop/policies/best_practices_for_ieee_standards_development_051215.pdf

- Distribution of Draft Standards (see 6.1.3 of the SASB Operations Manual)
  - https://standards.ieee.org/about/policies/opman/sect6.html

4. Review and approve minutes

Working Group Meeting #88, November 30

Draft circulated November 30.

Correction: In 12, the nominal topic for this meeting was not updated.

Attendees:

Ian McIntosh (Leonardo)
Eric Cormack (DFT Solutions) (joined 11:25)
Terry Duepner (National Instruments)
Heiko Ehrenberg (GOEPEL Electronics)
Brian Erickson (JTAG Technologies)
Peter Horwood (Digital Development Consultants Ltd)
Joel Irby (AMD)
Richard Pistor (Curtiss-Wright)
Jon Stewart (Dell)
Brad Van Treuren (VT Enterprises Consulting Services)
Louis Ungar (A.T.E. Solutions)
Carl Walker (Cisco Systems)
Zdenek Plíva (Technical University of Liberec) (left mid-meeting)

5. Review open action items

Action Item Register:

http://files.sjtag.org/PostStudyGroup/ActionItemRegister.xlsx

Format of action number is

[Meeting#.Action# within that meeting]

[88.1] Ian: Check which members still need to set up access to iMeet.

- Reminder emails will go to regular members yet to set up access

6. Inter-group Collaboration

General:

- P1149.7 ballot complete, 75% response target met with 86% approval (75/75 rule applies). 83 comments to resolve.
- P1450 Standard Test Interface Language revision PAR approved to Dec. 2024
- P1450.1 STIL Extensions for Semiconductor Design Environments revision PAR approved to Dec. 2024
- P1450.6 STIL Core Test Language revision PAR approved to Dec. 2024

7. Discussion Topics

7.a Update on WG elections

7.b Review standard outline and figures

- See if there are any "quick wins"

General References:

- Definitions from forum: http://forums.sjtag.org/viewforum.php?f=40
- Reference Pack (previous material): http://files.sjtag.org/P2654WG/P2654_Reference_Pack.pptx

Wrap-up items

8. Any other business
9. Today's Key Takeaways
10. Glossary terms from this meeting
11. Schedule next meeting
    - December 14 (TBC)
    - Final meeting(s) of 2020, first meeting of 4 January 2021
12. Topic for next meeting
    - Update on WG elections
    - Review standard outline and figures
        - See if there are any "quick wins"
13. Reminders
    - December 13 closure for nominations
14. List new action items
15. Adjourn

Definitions

Device: Used in the sense of "something made or adapted for a particular purpose", may be an individual component or a larger assembly (e.g. COTS board/module) – A lowest level "leaf" of the system, and end target instrument or collection of instruments

Component: An individual part, typically mounted on a PCB, that behaves other than transparently on the data passing through it – Essentially a part that doesn't meet the requirements of a "device" but has an effect on what's needed to perform a test

P2654 "Actors"

End user*: Person (or sub-assembly) that wishes to execute a test

Test Developer*: Creator of tests for the End user and who may use test tools

Test Tool Vendor: Provider of software tools to aid test creation

System Integrator: Person that aggregates sub-assemblies (boards, devices) to form a system

Circuit Designer†: Person creating boards or sub-assemblies

Device Vendor†: Person designing/supplying a low-level part to be used by circuit designers

EDA Tool Vendor: Provider of software tools to aid circuit/device design

* Roles may be combined in "interactive" applications

† Possibly there is overlap between these?

PAR Scope and Purpose

Scope: This standard addresses use/ reuse of test assets in system context by: 1) defining a representation for behavioral descriptions of pertinent sub-assembly interfaces and of relevant data and protocol transformations; 2) defining methods for utilizing such representations to enhance management of and access to said test assets. In conjunction with existing methods for test access and test management, this will allow the coordination and control of a variety of digital interfaces to devices, boards, and sub-systems to extend test access to board and system levels. This standard does not replace or provide an alternative to existing test interface standards, but aims instead to enable their usage throughout the hierarchy of systems.

Purpose: The purpose of this standard is to facilitate a means to seamlessly integrate component access topologies, interface constraints, and other dependencies at the board and system level by using standardized descriptions focusing on topology, interfaces and behavior (as opposed to physical structure). This will ease the burden on those preparing test, maintenance and support applications, including Automatic Test Pattern Generation (ATPG), in particular where the application requires to co-ordinate control of and data transfer through multiple interfaces and/or protocols. Typically, the providers of these conforming descriptions are the producers of integrated circuits, printed circuit boards or sub-systems, including, for example, intellectual property cores in a System on Chip (SoC), with digital interfaces that are intended to be used in an automated fashion within a larger assembly. This standard will also include a methodology to ensure access to particular destination registers in the correct time order.

P2654 in a nutshell

In essence, it is the "glue" that binds the test instruments to the testable system' interface boundary

It doesn't define the test instruments or how they behave

It doesn't define the test application

It doesn't (explicitly) define a UI at the system boundary – that's derived from the behaviours of the constituent parts and the available external interfaces

For each actor…

To what extent do they need to be aware of P2654?

What will they get out of P2654?

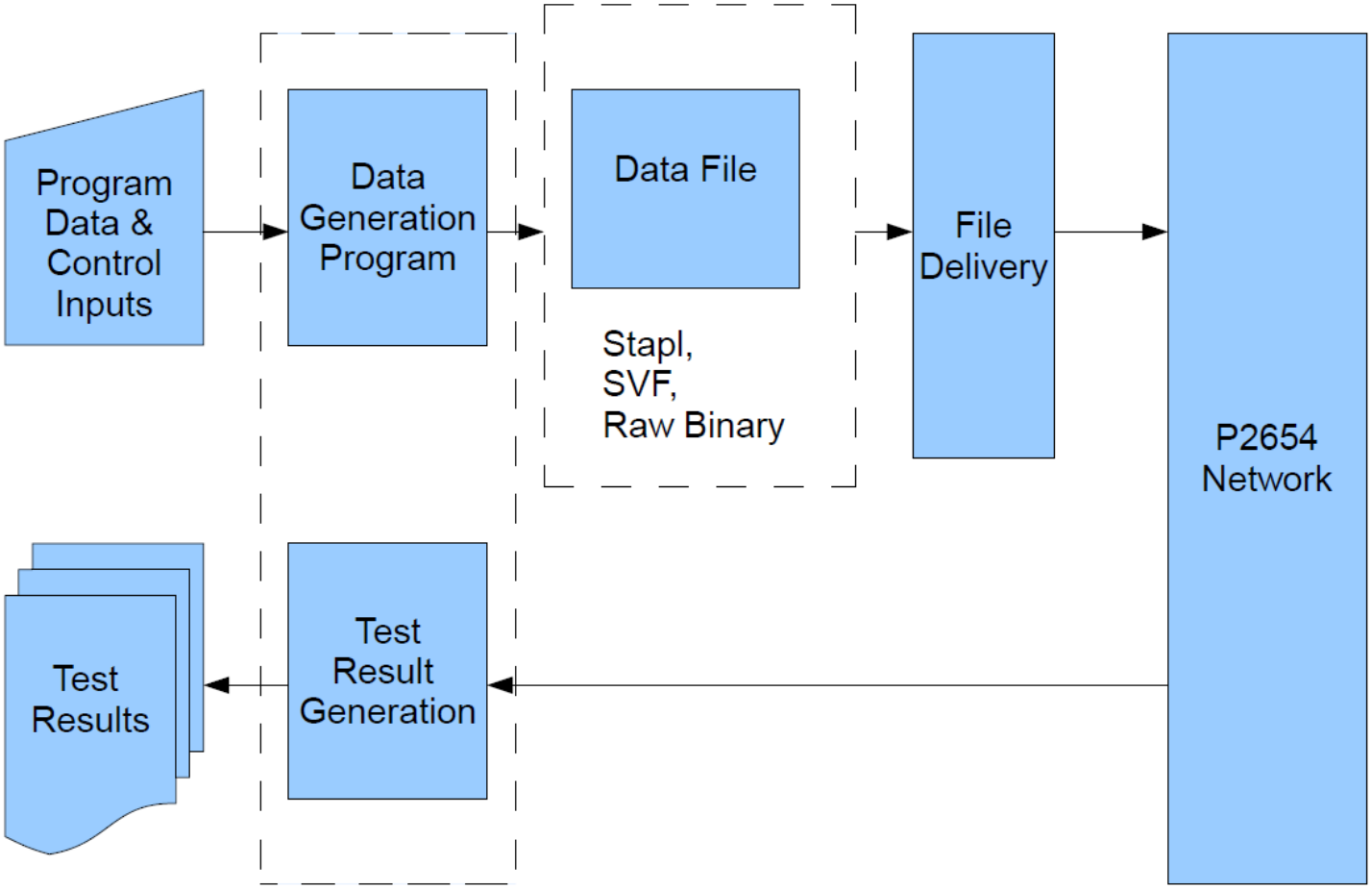
What (if anything) do they need to create for P2654 to work?

What (if anything) do they need to know (from other actors) in order to use P2654?

What (if anything) do they already know that applies to P2654 needs?

How will they use P2654?

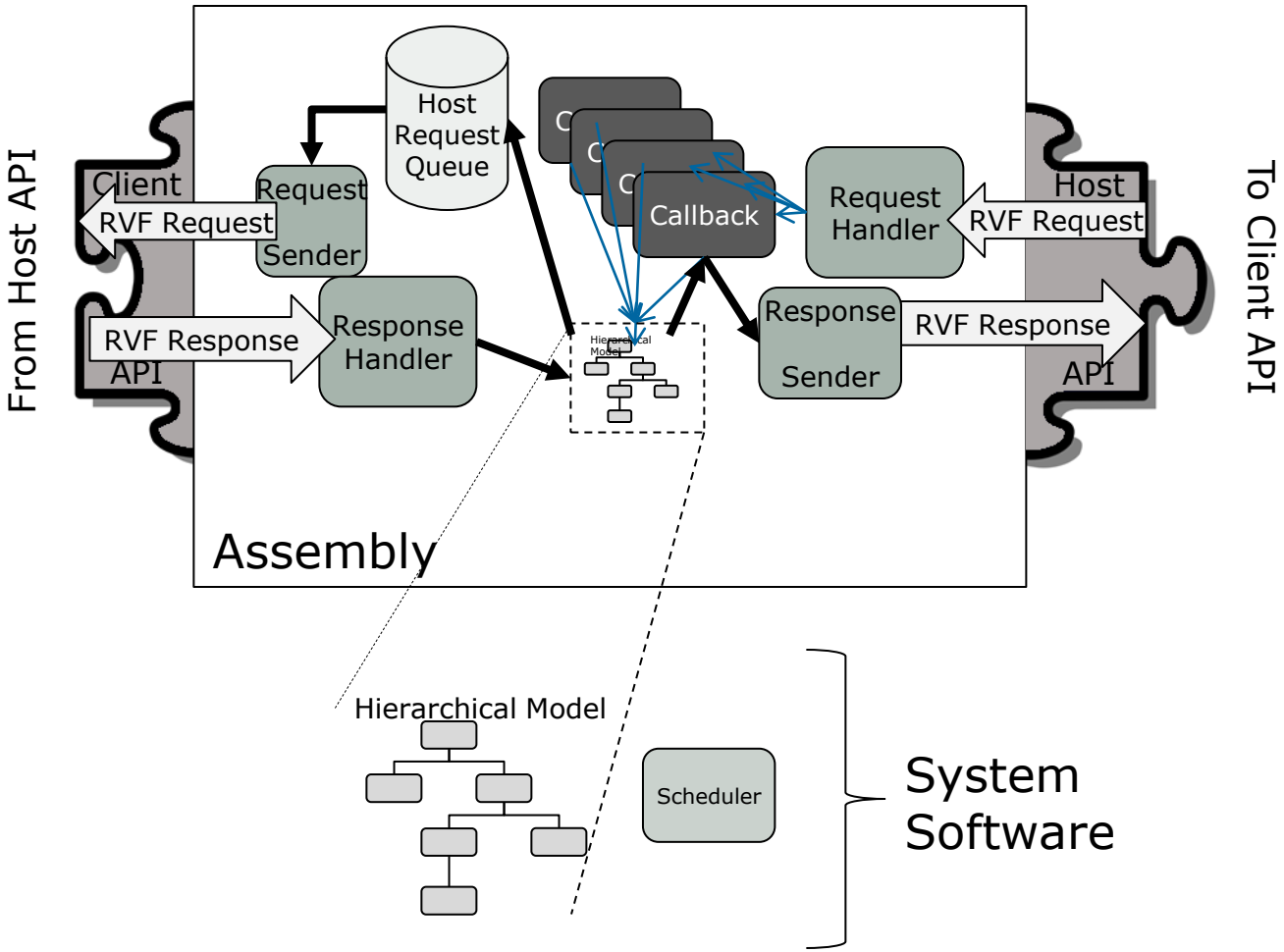
Vector Generation Interactions



Untitled (2)

- Input to test delivery generation can be TCL, or vendor specfic
- Dependent on the tool flow the generation and delivery may be in a single operation or split in to separate actions

Architecture1



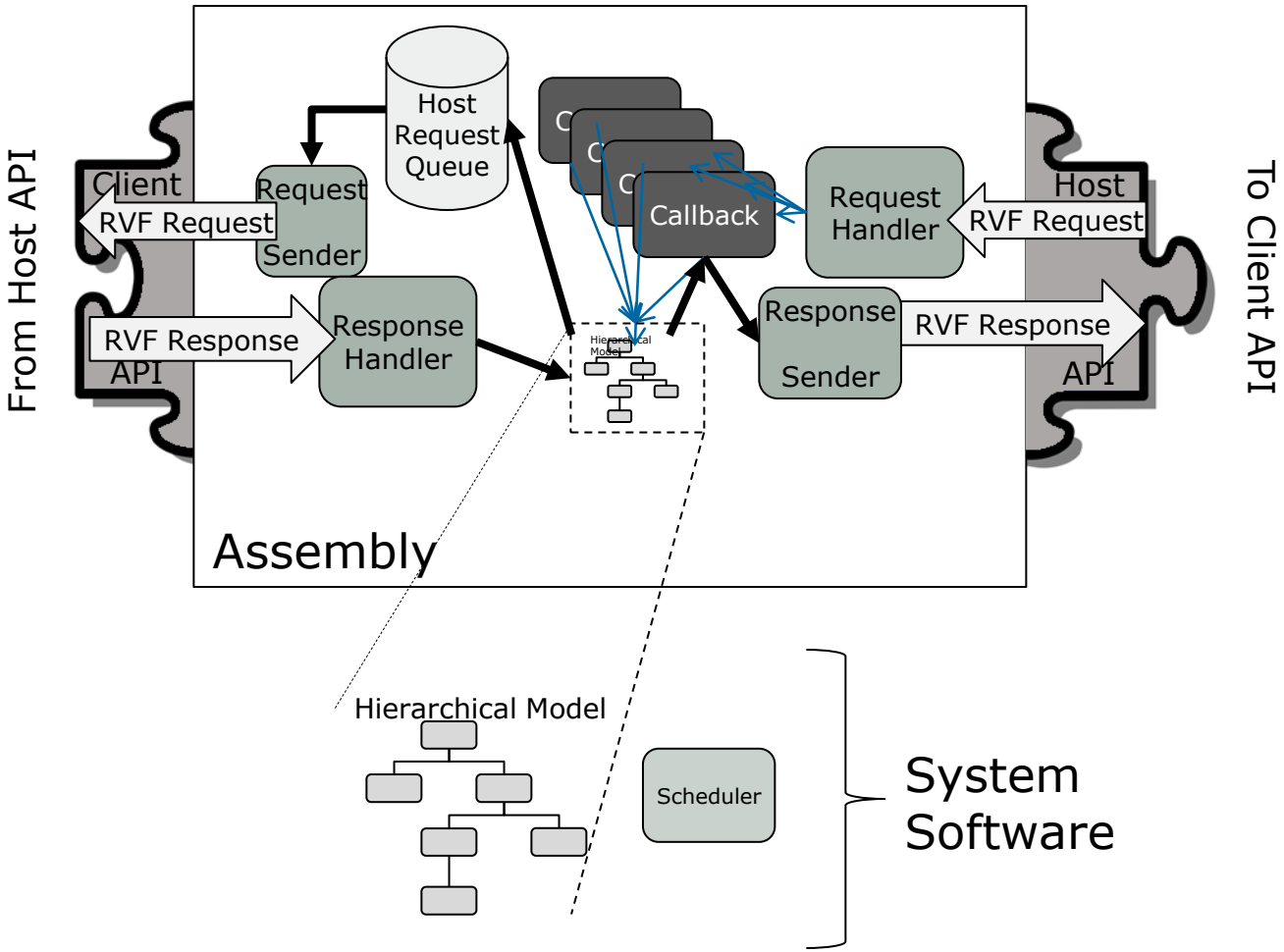
Discussion 20200824

- Product Data Management System (Leonardo)
  - Parts: Something that you make
  - Component: Something that you buy
- Additional Actors
  - Device IP Integrator
  - Device Designer
- Actors have different perspectives of P2654
- An Actor may never see to concept of P2654 when applying tests
- P2654 is the glue between the top and bottom ends of a design
- See P2654 in a nutshell slide from meeting pack
- P2654 is not defining an interface, but querying for capability may require a software interface definition (define method or just principle)

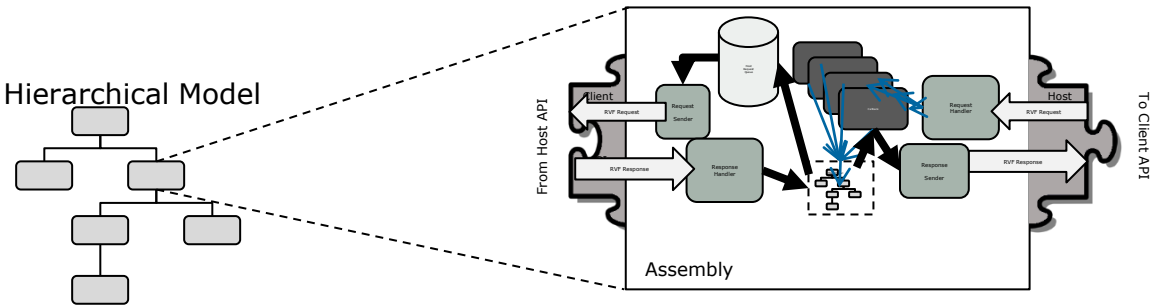
Discussion 20200824

- Data File in diagram is really Data Stream
- File delivery is really the Data Stream to driver for P2654 Network interface provided by integrator/tool vendor
- P2654 really resided in the Data Generation Program and Test Result Generation boxes. The other boxes are outside the scope of P2654
- P2654 is concerned with the data stream interacting with the network driver interface. System or tool designer is responsible for integrating the data stream with the hardware interface.

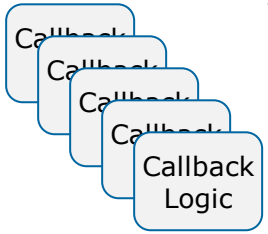
Architecture1



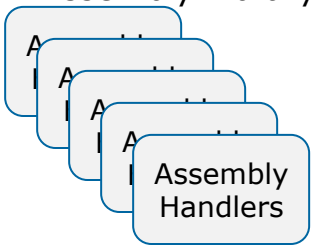
Context1

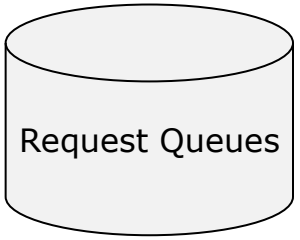


Callback Library



Assembly Library





Modeling Issues

- PDL references entities using hierarchical path names based on context of the ICL module addressed (iWrite mbist.r1 0x23)
- This path is a relative path for the context of device, board, and system as the PDL is intended to be reused for all instances of that ICL module in the design
- Thus, the absolute path to the target instance, giving the application proper context to which instance to reference, is required to resolve the overall context at run-time by the application or data generator
- This is subtly different than resolving the context by the retargeter or transformation callback
- Transformation logic is a reusable callback for all instances of the target entity
- Transformation logic needs to be able to resolve context itself separate from the application context based on the target instance being used

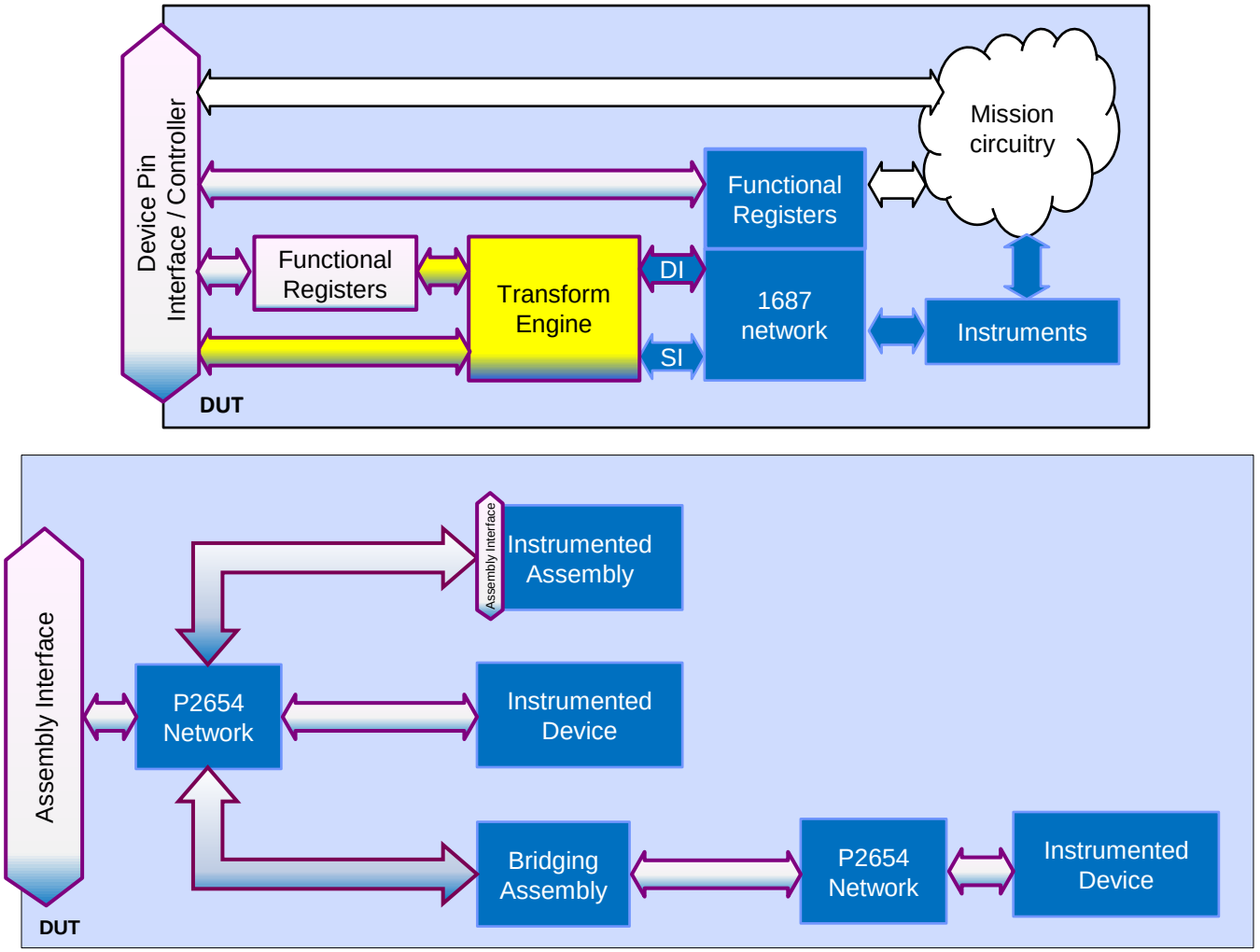
Modeling Issues

- How do we indicate what model elements are necessary for transformation logic as parameters/attributes(e.g., address filtered out of data or indirectly as set by writing a Register as separate scan operation?)?
- When activating a path to a leaf register, do you:
  - Automatically cascade up a branch sequentially activating each branch level from top down towards the target register?
  - Or, do you require manual specification by the user to the model within the application code?
  - (1687 Retargeter automatically enables SIBs on a 1687 Network)
- How do you model dependencies between model elements, like registers? (e.g., Selection of active TAP TDR is dependent on the value of the TAP IR. Writing to TAP IR or TAP DR is dependent on the TAP state.) Dependencies:
  - Associated Register(s)
  - Associated state
  - Associated signal value

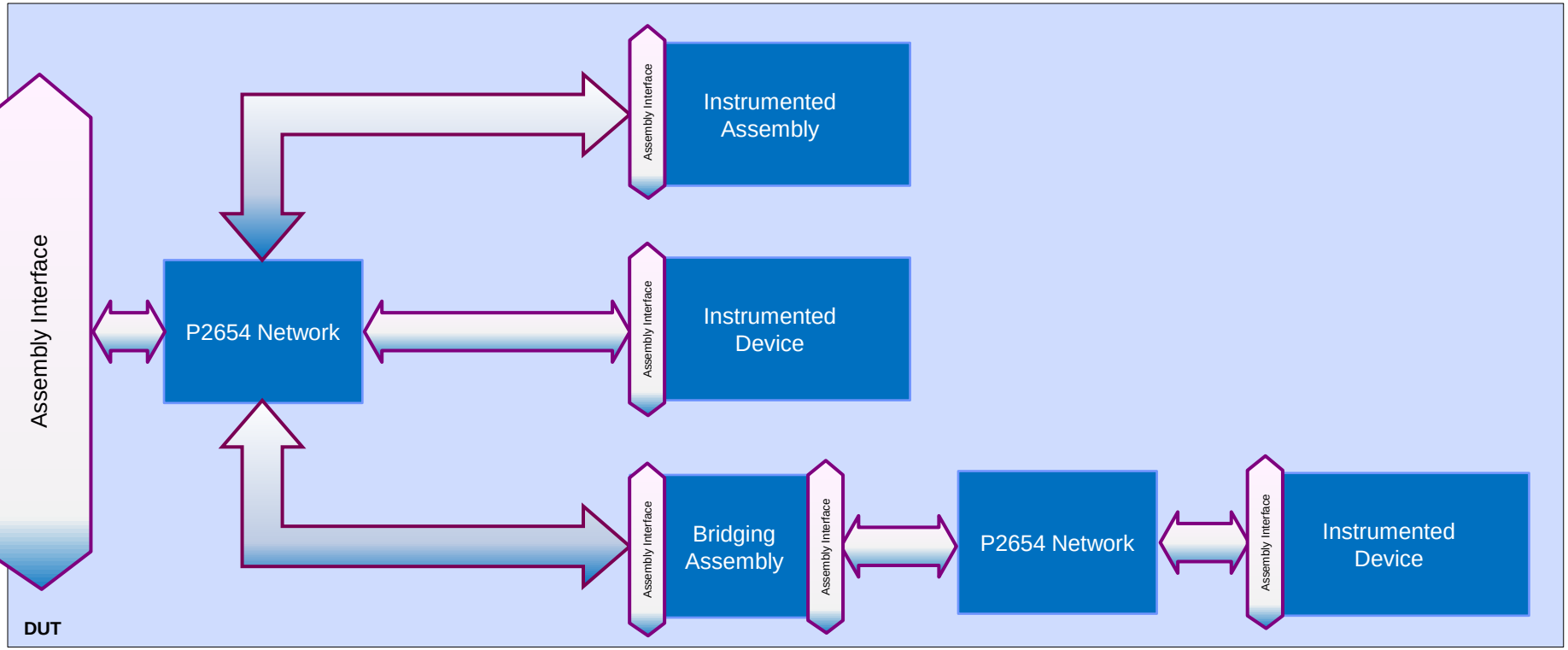
ICL Describes Structural Primitives

- IEEE 1687 Structural Primitives
  - DataInput Register
  - DataOutput Register
  - Missing DataInOutRegister!
  - ScanRegister
  - ScanMUX
    - Associated Control Signal(s)
    - Associated Selection Register
      - Binary
      - OneHot
      - OneHot –No IDLE
      - N-Hot
      - N-Hot – no IDLE
  - DataMUX (Same control associations)

Context1

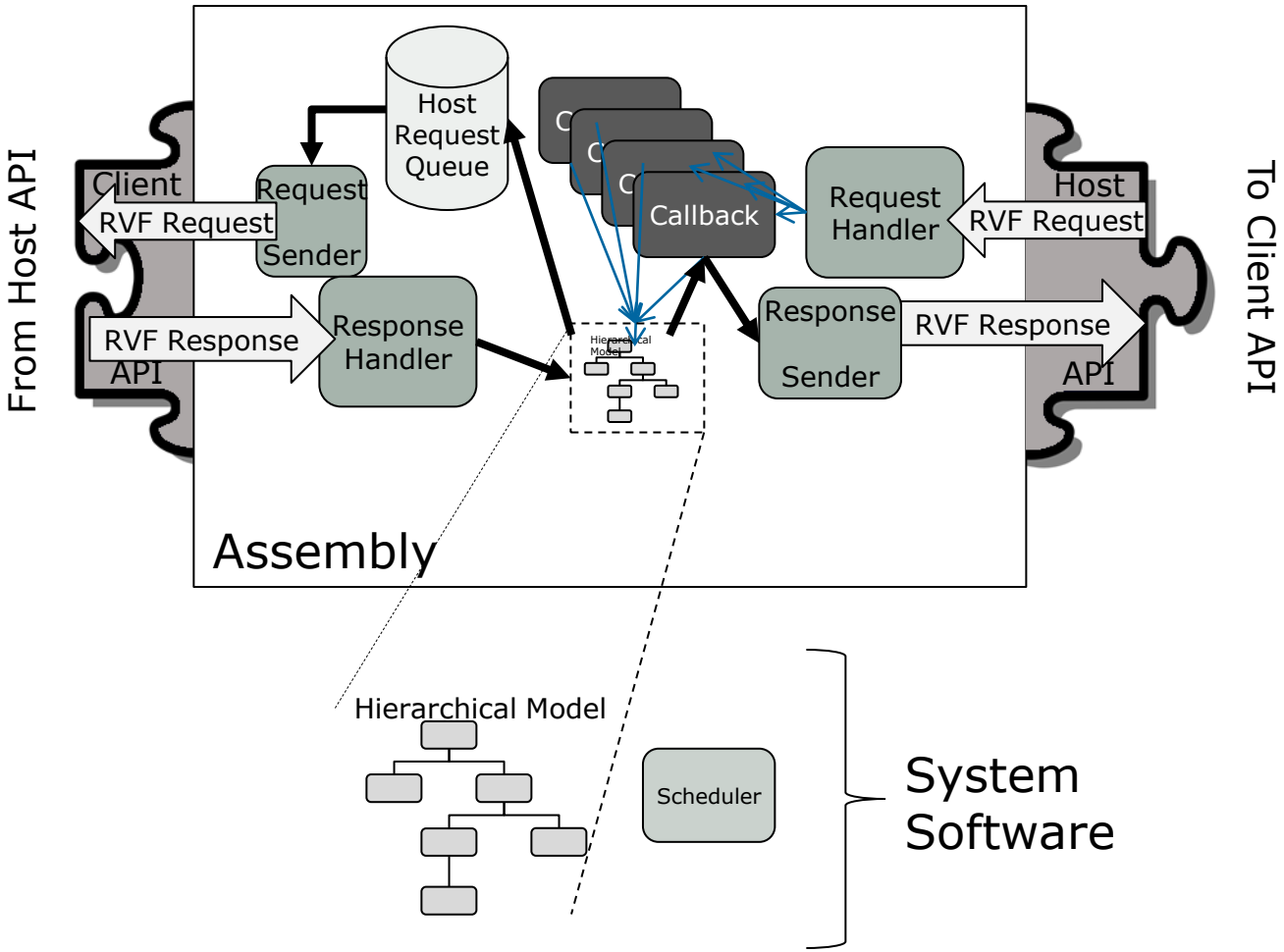

Device Pin Interface / Controller
Mission circuitry
Functional Registers
Functional Registers
Transform Engine
DI
SI
1687 network
Instruments
DUT
Assembly Interface
Assembly Interface
Instrumented Assembly
P2654 Network
Instrumented Device
Bridging Assembly
P2654 Network
Instrumented Device
DUT

Context2 – Physical Layer Diagram?

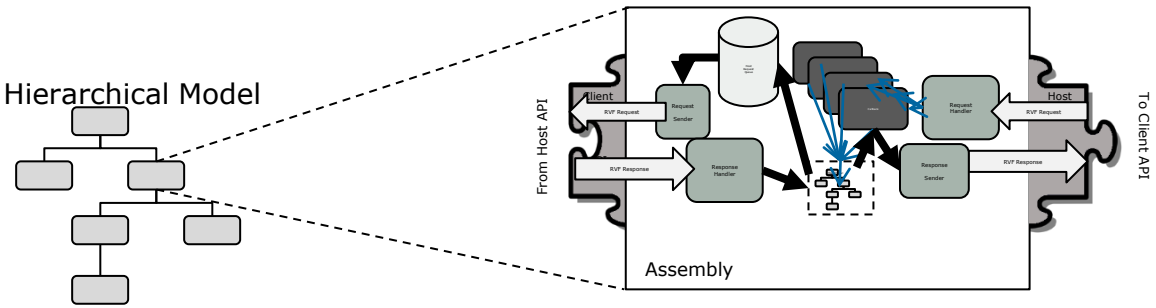


- Do we represent context from a physical or modeling perspective?
- Can P2654 Network be inferred from CAD data or does it need a new description format like 1149.7 does with HSDL.7?
- What are the types of P2654 Networks? Serial? Bused? Parallel/Digital?
- Or should we represent data message context with host and client interfaces binding assemblies over P2654 Networks? Or is that a diagram of the Access and Data Link Layers?

Architecture1



Context1



Callback Library

Callback Logic

Assembly Library

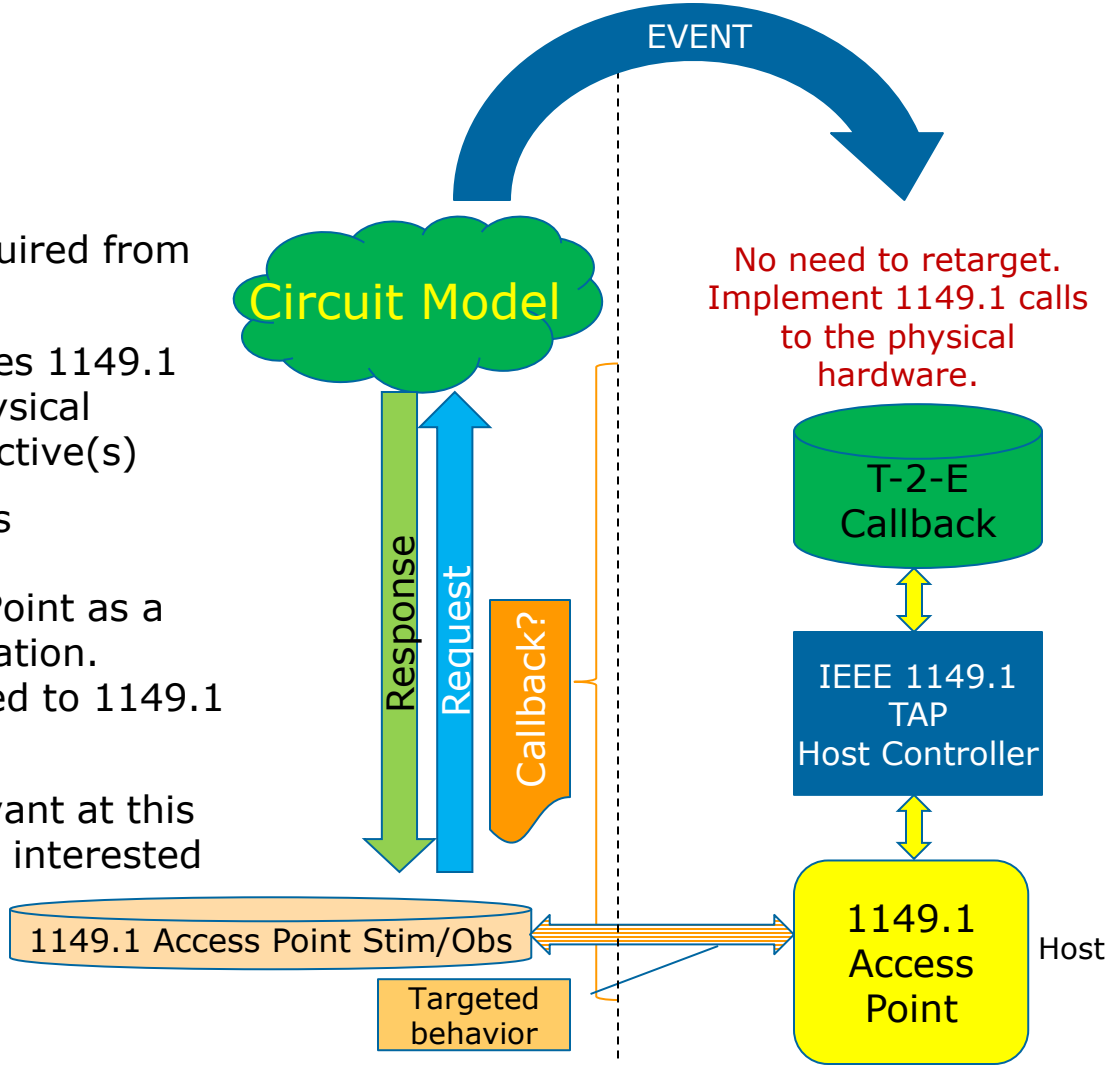
Assembly Handlers

Request Queues

Two Types of Modeling Approaches

Top: Request and Response Model (Bottom Up Approach)

- 1149.1 Stim/Obs define Requests/Responses required from the next higher level
- Pass 5: Request translates 1149.1 Stim/Obs in terms of physical 1149.1 Access Point directive(s)
- Pass 5: Response returns values observed at the 1149.1 physical Access Point as a result of a stimulus operation. Response is un-retargeted to 1149.1 scope.
- Instrument PDL is irrelevant at this level of retargeting: only interested in control of 1149.1 Access Point



Modeling Issues

- PDL references entities using hierarchical path names based on context of the ICL module addressed (iWrite mbist.r1 0x23)
- This path is a relative path for the context of device, board, and system as the PDL is intended to be reused for all instances of that ICL module in the design
- Thus, the absolute path to the target instance, giving the application proper context to which instance to reference, is required to resolve the overall context at run-time by the application or data generator
- This is subtly different than resolving the context by the retargeter or transformation callback
- Transformation logic is a reusable callback for all instances of the target entity
- Transformation logic needs to be able to resolve context itself separate from the application context based on the target instance being used

Modeling Issues

- How do we indicate what model elements are necessary for transformation logic as parameters/attributes(e.g., address filtered out of data or indirectly as set by writing a Register as separate scan operation?)?
- When activating a path to a leaf register, do you:
  - Automatically cascade up a branch sequentially activating each branch level from top down towards the target register?
  - Or, do you require manual specification by the user to the model within the application code?
  - (1687 Retargeter automatically enables SIBs on a 1687 Network)
- How do you model dependencies between model elements, like registers? (e.g., Selection of active TAP TDR is dependent on the value of the TAP IR. Writing to TAP IR or TAP DR is dependent on the TAP state.) Dependencies:
  - Associated Register(s)
  - Associated state
  - Associated signal(s) value

ICL Describes Structural Primitives

- IEEE 1687 Structural Primitives (Figure 39 from IEEE 1687)
  - DataInput Register
  - DataOutput Register
  - Missing DataInOutRegister!
  - ScanRegister
  - ScanMUX
    - Associated Control Signal(s)
    - Associated Selection Register (from Michele's SIT)
      - Binary
      - OneHot
      - OneHot –No IDLE
      - N-Hot
      - N-Hot – no IDLE
  - DataMUX (Same control associations)

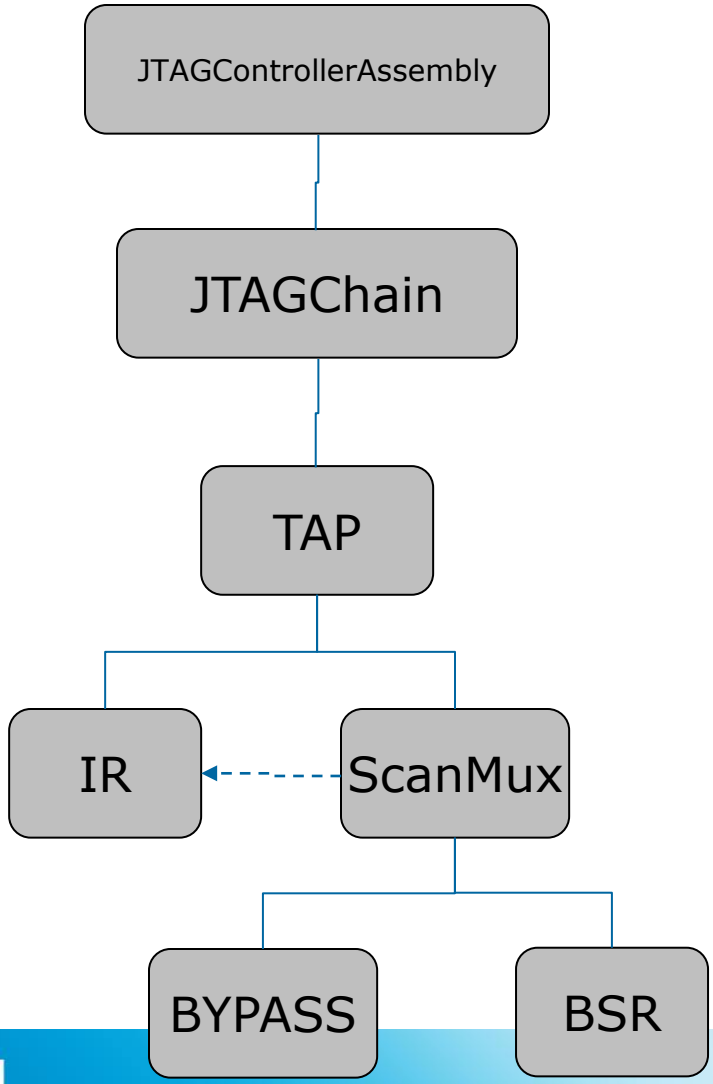
Transformation Callback Logic Blocks

- Requested Command Callback: Used to
  - Decode RVF message sent from client
  - Transform request into necessary ordered requests for the host
  - Update system model state for specific Assembly instance with requested data to be synchronized with hardware (write) based on request data
  - Wait for necessary response data from host to complete processing
- Response Command Callback: Used to
  - Decode RVF message sent from host
  - Reverse transform response into subsets of data required by the clients
  - Update system model state for specific Assembly instance based on response data synchronized with hardware (read) based on response data

Transformation Callback Logic Blocks

- Shared Block
  - Common data shared between Request Callback and Response Callback for handoff of data between callbacks (useful if each are running is separate threads)
  - Shared utility procedures used by both callbacks
- Optionally (Depending on design implementation)
  - System management or scheduling process completing deferred actions from Request Callbacks and Response Callbacks
  - Triggered by execution of iApply in a PDL domain

P2654Board1 Design



Alternate Use Cases/Examples/Illustrations

Benchmark Test Cases

- TCP/IP example for System/Sub-Systems
  - Michele's Automated Testing Flow Paper
- SM-Bus/IPMI over I2C/TCP-IP/RS-232/USB to Baseboard Management Controller (BMC) running software (bulletin board shared variables to set and get)[ssh, Telnet, rsh possible too]
- OpenCores.org SPI to I2C RTL (Bridge example)
- Brad's JTAG to I2C RTL (Bridge example)
- I2C to bit-bang IEEE 1687 as in what P1687.1 is proposing
- Example using binary black box algorithm as extension to description of transformation
- Bottom-up control flow (Instrument register control)
- Top-down control flow (Terry's System request)

What needs to be described by standard?

- Topographical relationship between hierarchical layers
  - Includes Assembly instance specific details for nodes
  - Includes AccessInterface instance specific details for edges
  - Example is Michele's Simplified ICL Template (SIT)
- Description of algorithm to be applied at each Node
  - Generalized for use by each instance of the Assembly
  - Parameterized by information in the Topology relationship description for each instance
- Description of common elements for an Assembly for all instances of the same configuration (e.g., what BSDL does for 1149.1)

Transformation and RVF-to-RVF interface boundaries Reflecting on Black Box and fully described cases

- Leaf descriptions (hand-off from other standards) and not hierarchical transformational descriptions
  - Described and not prescribed
  - Describes behavior of use and not implementation of design
  - May be an obfuscation of the behavior
  - STAM describes sequence of events and not necessarily the timing of the events (not cycle based).
  - Allow IEEE 1149.1-2013/1687 PDL event cycle compatibility
    - 1149.1-2013 BSDL extensions for TDR path descriptions
    - 1687 ICL for timing and access to TDR (Scan Registers) and Data Registers (Parallel Instrument Registers)
  - Defines what has to be passed via RVF to the P2654 tool flow
- STAM is agnostic to purpose of data and only required to transfer data at the right time in the right sequence
- STAM is agnostic to what standard (official or ad hoc) are being used at the leaf nodes of the P2654 model

Transformation and RVF-to-RVF interface boundaries
Message Transformation

- Brad's demo packets
  - Universal Identifier – UID (unique for an assembly instance – a node in the topology tree slide 41)
  - Command – String
  - Payload – binary data formatting parameters and data required for a callback
- There are 2 aspects for the packet
  - That which is specific to a tool implementation (not part of the standard) – UID
  - That which is descriptive and not prescriptive as part of the interface: Callback name, Callback parameters, Data
- UID across physical hardware boundaries (system to sub-system) may require a way of exporting the remote UID and aliased for the local model to use. Does this make UID required for the standard?

Transformation and RVF-to-RVF interface boundaries
Message Transformation

- Brad's simplified transformation engine diagram was shown (see slide 47)
- Ideally, looking at Jeff Rearick's gRPC description slides from his 1687.1 demo would be helpful
- Need to look at Strategy and Plug-in Software Design Patterns (google search for these for details)

Simplified AccessInterface and Node Diagram