AADL Networking Annex
Status Report

Brendan Hall, Alexey Khoroshilov, Tiyam Robati

SAE AS-2C AADL Winter Meeting 2016

Recent Version

The most recent version available here:

https://gitlab.com/sae_as2c/networking-annex/wikis/home

AFDX Trademark Question

1.0 INTRODUCTION

1.1 Purpose of Document

The purpose of this document is to define a deterministic network: Avionics Full Duplex Switched Ethernet (AFDX™). AFDX™ is a trademark of Airbus and is used with permission. This document also highlights the additional performance requirements of avionics systems, within the context of AFDX.

This specification allows:

- System integrators to design flight critical systems using AFDX
- Equipment designers to specify equipment interoperable with AFDX

Open Questions

- AFDX Trademark
- Feedback?
- TTEthernet?

Old Action Items

- AI1. ~~Update AFDX reference model with the new example that demonstrates how to represent asymmetrical Network A-B designs in AADL.~~ **Responsible: Alexey**
- AI2. Add to reference model a model of end to end protocol on top of AFDX virtual links. **Responsible: Brendan**
- AI3. Describe safety properties of AFDX components from the reference model with EMV2. **Responsible: Brendan**
- AI4. Propose recommendations how to model ICD in AADL. **Responsible: Brendan**
- AI5. Update TTEthernet reference model and organize its review by TTTech. **Responsible: Tiyam, Brendan**
- AI6. ~~Prepare a plan of the Networking Annex text document.~~ **Responsible: Alexey**

Next Steps

- How to proceed?

Backup Slides

Alexey Khoroshilov
khoroshilov@ispras.ru

http://masiw.ispras.ru

Scope of Networking Annex

- AFDX as defined in ARINC 664p7
- Deterministic Ethernet as defined by ARINC 664 and SAE AS6802 (TTEthernet)

Goal

- to provide recommendations how to model ARINC 664 and SAE AS6802 networks in AADL
- so we could achieve tools interoperability on the models

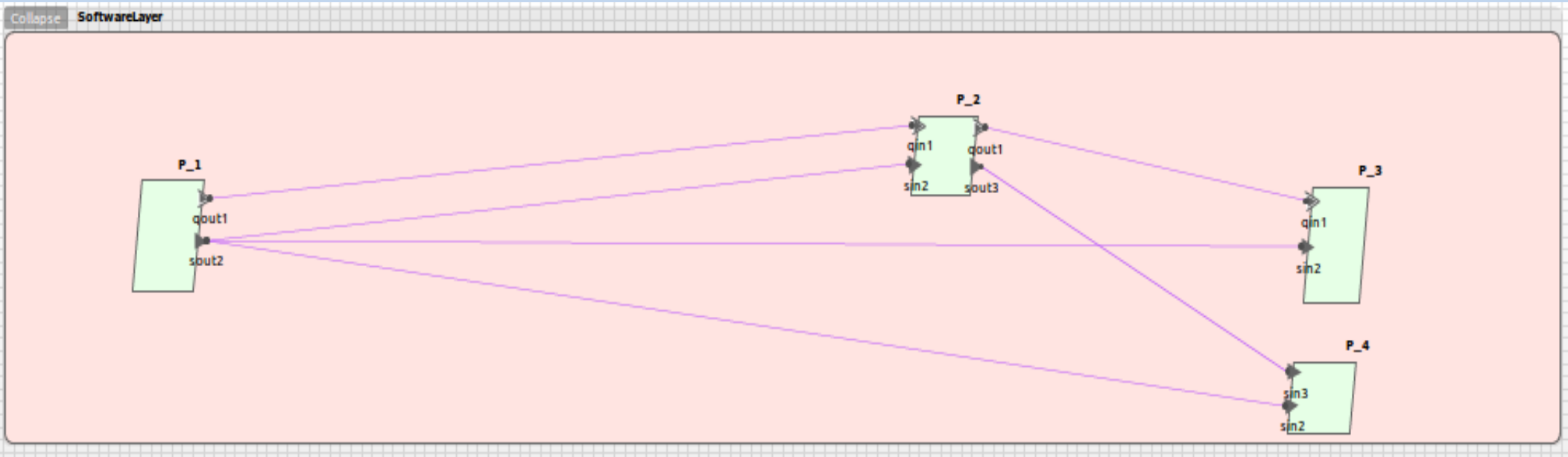
Target Use Cases

- **UC-1) Latency (timing) analysis of network**
- **UC-2) Buffer capacity analysis**
- **UC-3) Consistency analysis**
- UC-4) Synthesis and trade-off analysis
- UC-5) Safety analysis
- *UC-6) Configuration generation*

AFDX Model Variants

- Pre-Synthesis model
  - Input data for AFDX virtual links generation
- Basic model
  - Represents internal structure of AFDX network
  - Ignores redundancy
  - Works well for symmetrical Network A and B
- Detailed model
  - Represents internal structure of AFDX network including asymmetrical redundancy configurations
- Model with safety related properties

Software Layer



Let us suppose, we have a software system to be communicated via AFDX network.

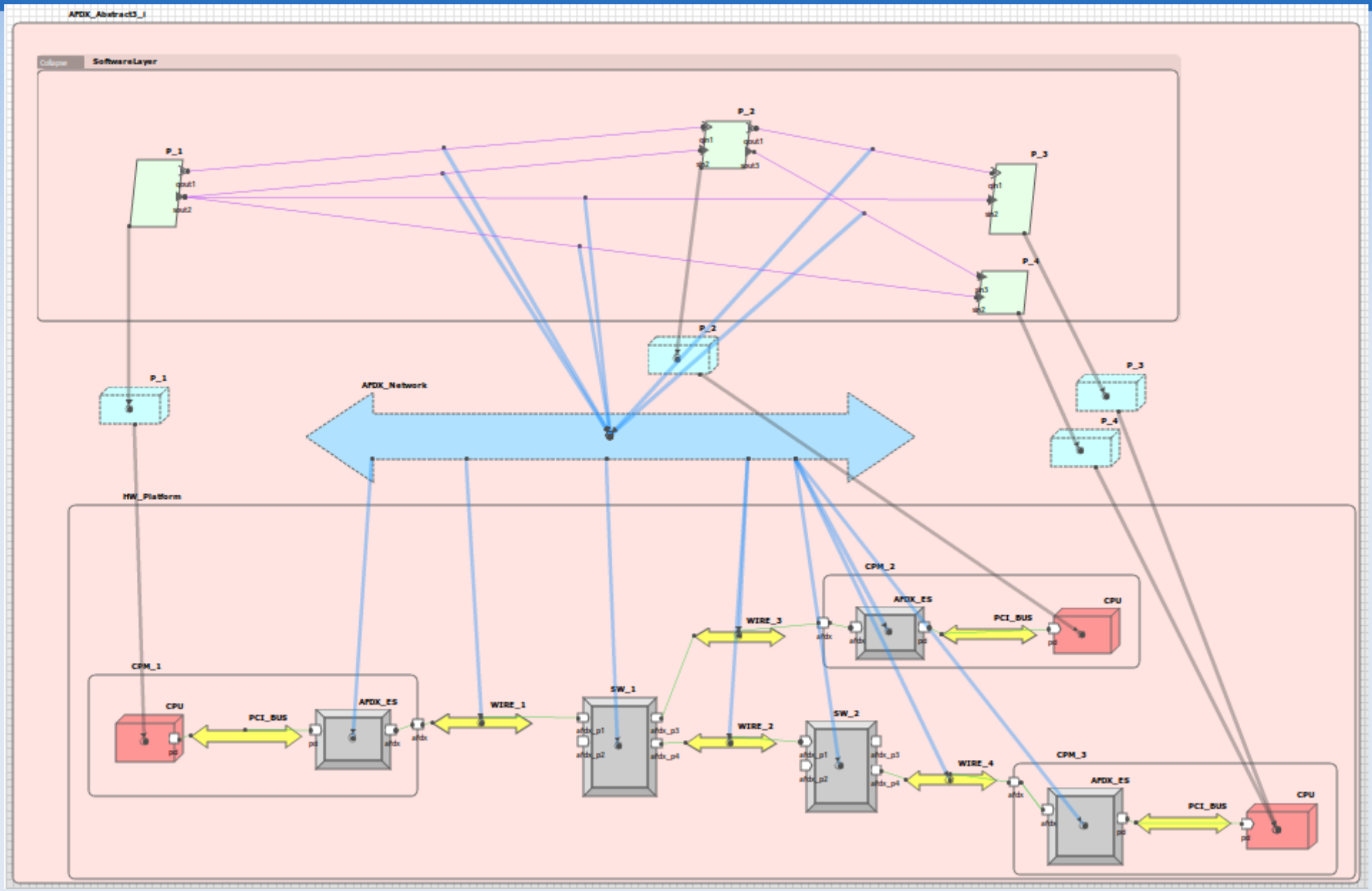
AFDX Model Variants

- **Pre-Synthesis model**
  - **Input data for AFDX virtual links generation**
- Basic model (no redundancy)
  - Model internal structure, ignore redundancy
  - Works for symmetrical Network A and B
- Detailed model (redundancy support)
  - Model internal structure including redundancy
- Model with Safety related properties

Pre-Synthesis Model

- The model describing an input for synthesis algorithm that automatically generates AFDX virtual links and their attributes meeting requirements (e.g., latency on connections, flows, etc.).
- The target for synthesis algorithm is AFDX_Network virtual bus.
- Connections bound to it should go via AFDX Network that is formed by components of HW_Platform AFDX_Network bounds to.

Pre-Synthesis Model



AFDX_Network is used to define a target for AFDX configuration synthesis.

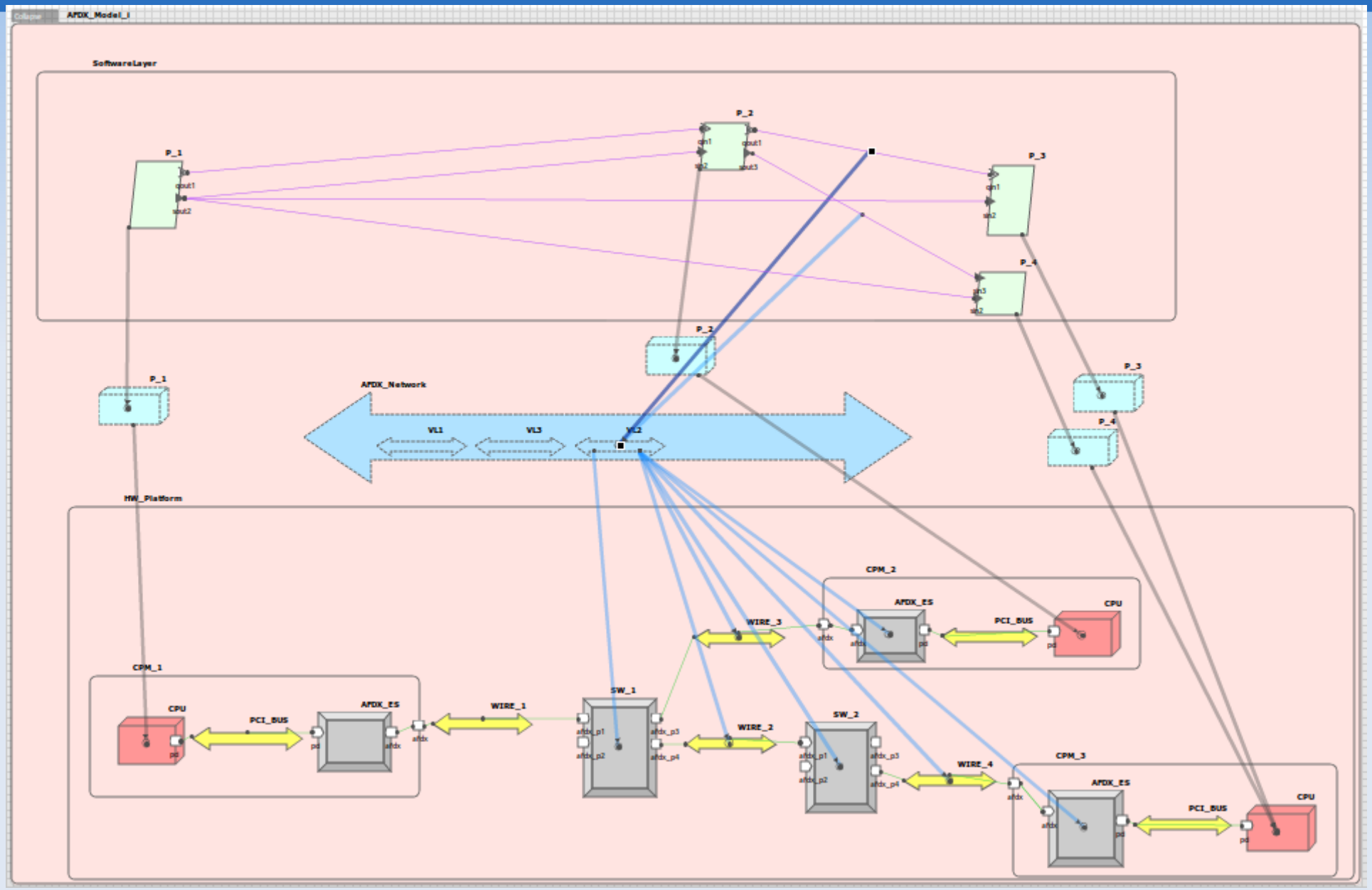
Pre-Synthesis Model

```
-------------------------------------------------------------------
----------------------AFDX_Model-----------------------------------
-------------------------------------------------------------------
  SYSTEM AFDX_PreSynthesis
  END AFDX_PreSynthesis;
  SYSTEM IMPLEMENTATION AFDX_PreSynthesis.i
    SUBCOMPONENTS
      Software_Layer : SYSTEM Software_Layer::Software_Layer.i;
      HW_Platform : SYSTEM HW_Platform.i;
      -- ARINC-653 Configuration
      P_1 : VIRTUAL PROCESSOR;
      P_2 : VIRTUAL PROCESSOR;
      P_3 : VIRTUAL PROCESSOR;
      P_4 : VIRTUAL PROCESSOR;
      -- AFDX Configuration
      AFDX_Network : VIRTUAL BUS AFDX::Network;
    PROPERTIES
      -- Maps processes to processors
      Actual_Processor_Binding => (reference(P_1)) applies to Software_Layer.P_1;
      Actual_Processor_Binding => (reference(P_2)) applies to Software_Layer.P_2;
      Actual_Processor_Binding => (reference(P_3)) applies to Software_Layer.P_3;
      Actual_Processor_Binding => (reference(P_4)) applies to Software_Layer.P_4;
      Actual_Processor_Binding => (reference(HW_Platform.CPM_1.CPU)) applies to P_1;
      Actual_Processor_Binding => (reference(HW_Platform.CPM_2.CPU)) applies to P_2;
      Actual_Processor_Binding => (reference(HW_Platform.CPM_3.CPU)) applies to P_3;
      Actual_Processor_Binding => (reference(HW_Platform.CPM_3.CPU)) applies to P_4;
      -- Map connections to virtual links
      Actual_Connection_Binding => (reference(AFDX_Network))
        applies to Software_Layer.CQ_1_2, Software_Layer.CS_1_2,
                   Software_Layer.C_2_3, Software_Layer.C_1_3,
                   Software_Layer.C_1_4, Software_Layer.C_2_4;
      -- Map AFDX Network to HW elements
      Actual_Connection_Binding => (reference(HW_Platform.CPM_1.AFDX_ES),reference(HW_Platform.CPM_2.AFDX_ES),
                                    reference(HW_Platform.CPM_3.AFDX_ES),
                                    reference(HW_Platform.WIRE_1),reference(HW_Platform.WIRE_2),
                                    reference(HW_Platform.WIRE_3),reference(HW_Platform.WIRE_4),
                                    reference(HW_Platform.SW_1),reference(HW_Platform.SW_2))
        applies to AFDX_Network;
  END AFDX_PreSynthesis.i;
```

AFDX Model Variants

- Pre-Synthesis model
  - Input data for AFDX virtual links generation
- **Basic model**
  - **Represents internal structure of AFDX network**
  - **Ignores redundancy**
  - **Works well for symmetrical Network A and B**
- Detailed model
  - Represents internal structure of AFDX network including asymmetrical redundancy configurations
- Model with safety related properties

Basic Model



Virtual Links are represented as virtual buses within AFDX_Network.

Basic Model

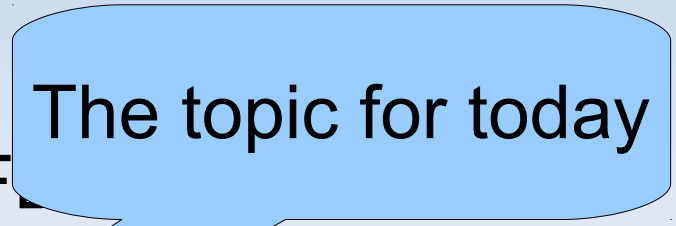
```
-------------------------------------------------------------------
--- 2. AFDX Network Configuration
-------------------------------------------------------------------
  VIRTUAL BUS IMPLEMENTATION AFDX_Network.i
    SUBCOMPONENTS
      VL1 : VIRTUAL BUS AFDX::Virtual_Link;
      VL2 : VIRTUAL BUS AFDX::Virtual_Link;
      VL3 : VIRTUAL BUS AFDX::Virtual_Link;
    PROPERTIES
      -- Setup virtual links configuration
      AFDX_Properties::BAG => 1 ms applies to VL1,VL2,VL3;
      AFDX_Properties::Lmax => 1518 Bytes applies to VL1;
      AFDX_Properties::Lmax => 512 Bytes applies to VL2,VL3;
      AFDX_Properties::SkewMax => 1 ms applies to VL1,VL2,VL3;
  END AFDX_Network.i;
```

Basic Model

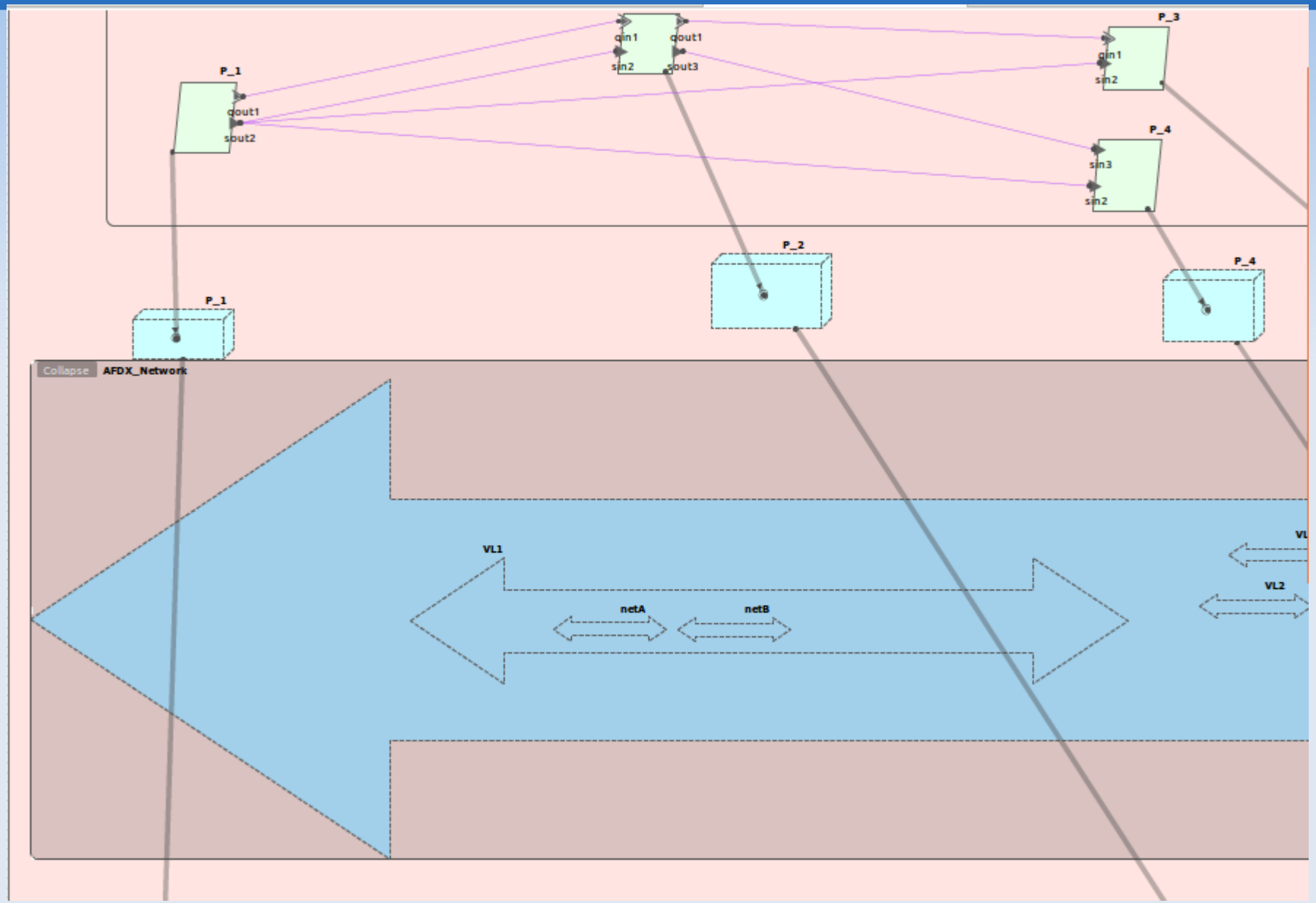
```
-- Define virtual link configuration
Actual_Connection_Binding => (reference(HW_Platform.CPM_1.AFDX_ES),reference(HW_Platform.WIRE_1),reference(HW_
                              reference(HW_Platform.WIRE_3),reference(HW_Platform.CPM_2.AFDX_ES))
  applies to AFDX_Network.VL1;
Actual_Connection_Binding => (reference(HW_Platform.CPM_2.AFDX_ES),reference(HW_Platform.WIRE_3),reference(HW_
                              reference(HW_Platform.WIRE_2),reference(HW_Platform.SW_2),
                              reference(HW_Platform.WIRE_4),reference(HW_Platform.CPM_3.AFDX_ES))
  applies to AFDX_Network.VL2;
Actual_Connection_Binding => (reference(HW_Platform.CPM_1.AFDX_ES),reference(HW_Platform.WIRE_1),reference(HW_
                              reference(HW_Platform.WIRE_2),reference(HW_Platform.SW_2),
                              reference(HW_Platform.WIRE_3),reference(HW_Platform.CPM_2.AFDX_ES),
                              reference(HW_Platform.WIRE_4),reference(HW_Platform.CPM_3.AFDX_ES))
  applies to AFDX_Network.VL3;
-- Map connections to virtual links
Actual_Connection_Binding => (reference(HW_Platform.CPM_1.PCI_BUS),reference(AFDX_Network.VL1),reference(HW_P
  applies to Software_Layer.CQ_1_2;
Actual_Connection_Binding => (reference(HW_Platform.CPM_1.PCI_BUS),reference(AFDX_Network.VL3),reference(HW_P
  applies to Software_Layer.CS_1_2;
Actual_Connection_Binding => (reference(HW_Platform.CPM_1.PCI_BUS),reference(AFDX_Network.VL2),reference(HW_P
  applies to Software_Layer.C_2_3;
Actual_Connection_Binding => (reference(HW_Platform.CPM_1.PCI_BUS),reference(AFDX_Network.VL3),reference(HW_P
  applies to Software_Layer.C_1_3;
Actual_Connection_Binding => (reference(HW_Platform.CPM_1.PCI_BUS),reference(AFDX_Network.VL3),reference(HW_P
  applies to Software_Layer.C_1_4;
Actual_Connection_Binding => (reference(HW_Platform.CPM_2.PCI_BUS),reference(AFDX_Network.VL2),reference(HW_P
  applies to Software_Layer.C_2_4;
-- Switch configuration
AFDX_Properties::VL_Route_Table => ([
      vl => reference (AFDX_Network.VL1);
      jitter => 8 us;
      priority => high;
      accountingPolicy => frame;
                          ],
                          [
      vl => reference (AFDX_Network.VL2);
      jitter => 16 us;
      priority => low;
      accountingPolicy => frame;
```

AFDX Model Variants

- Pre-Synthesis model
  - Input data for AFDX virtual links generation
- Basic model
  - Represents internal structure of AF
  - Ignores redundancy
  - Works well for symmetrical N work A and B
- **Detailed model**
  - **Represents internal structure of AFDX network including asymmetrical redundancy configurations**
- Model with safety related properties



The First Attempt



netA and netB virtual buses inside VL, to be bound to HW devices.

The First Attempt

```
VIRTUAL BUS IMPLEMENTATION Virtual_Link.dup
  SUBCOMPONENTS
    netA : VIRTUAL BUS Virtual_Link;
    netB : VIRTUAL BUS Virtual_Link;
END Virtual_Link.dup0;
```

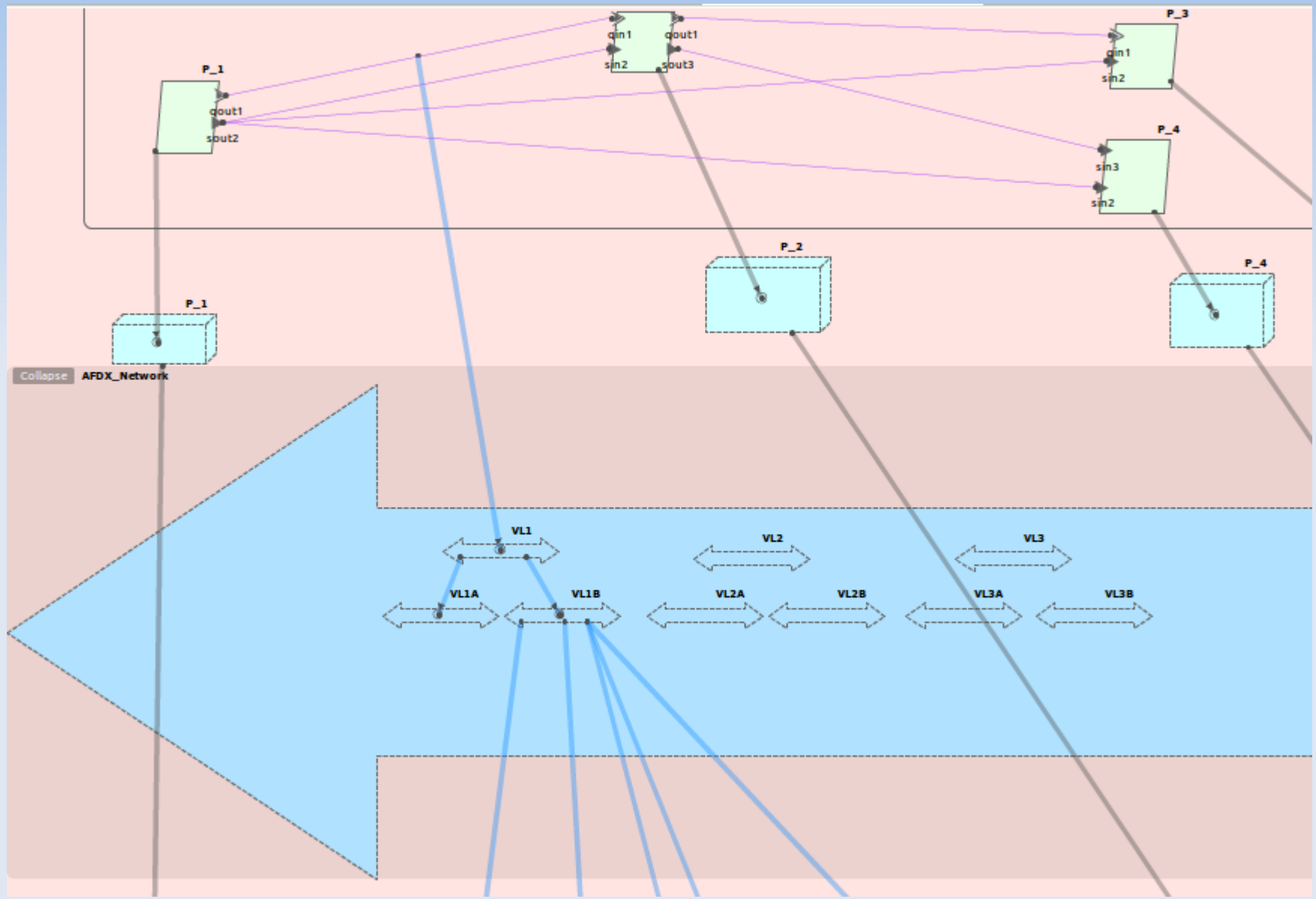
netA and netB virtual buses inside VL, to be bound to HW devices.

The First Attempt – Problems

```
VIRTUAL BUS IMPLEMENTATION Virtual_Link.dup
  SUBCOMPONENTS
    netA : VIRTUAL BUS Virtual_Link;
    netB : VIRTUAL BUS Virtual_Link;
END Virtual_Link.dup0;
```

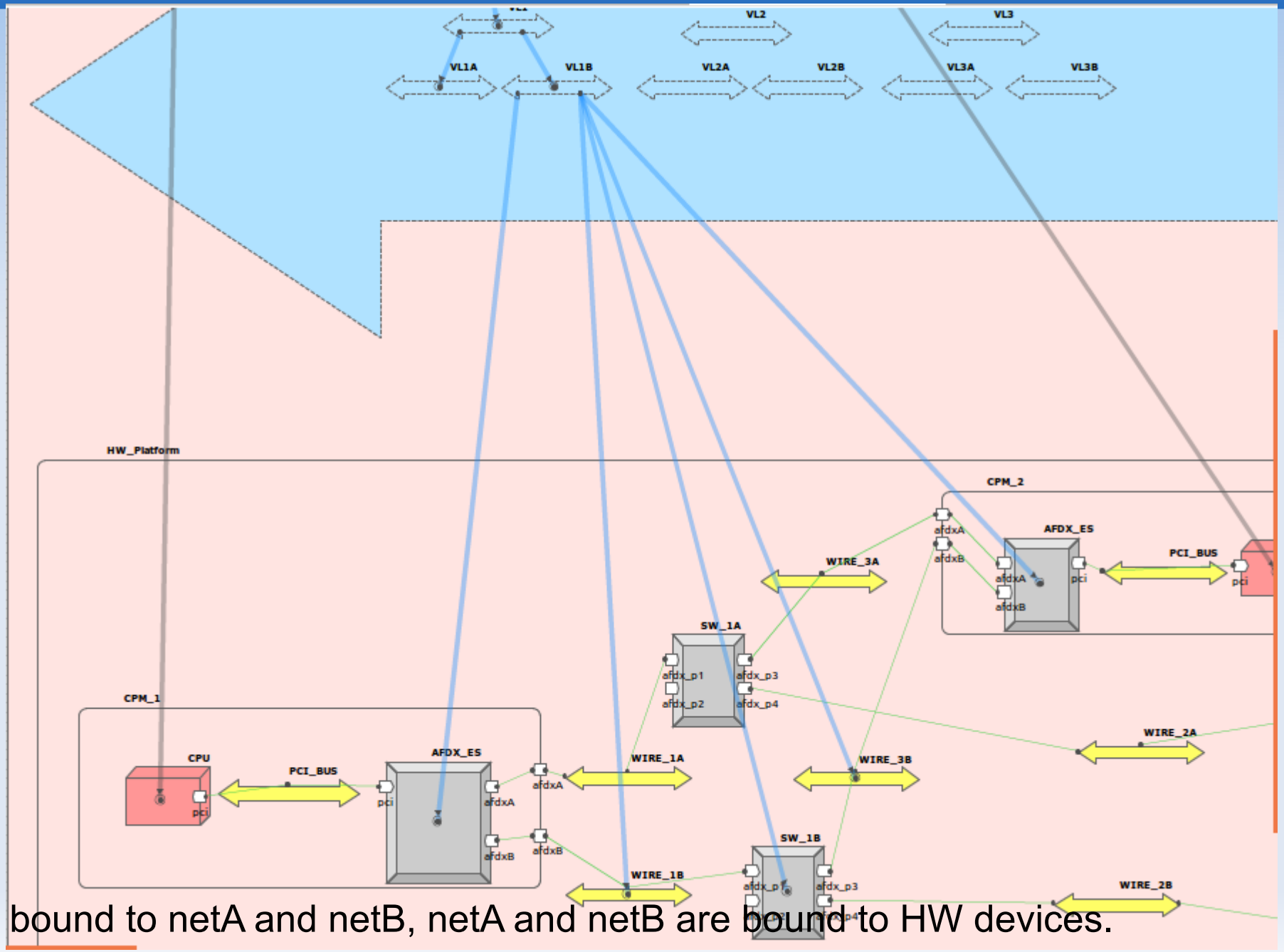
- According semantics of AADL 2.1 netA and netB are bound to VL
  - netA of VL provides resources to VL
    - then VL should be bond to netA
  - netA of VL is subchannel of VL
    - then netA could be bond to VL

An Alternative



VL is bound to netA and netB, netA and netB are bound to HW devices.

An Alternative



VL is bound to netA and netB, netA and netB are bound to HW devices.

An Alternative

```
VIRTUAL BUS IMPLEMENTATION Virtual_Link.A
  PROPERTIES
    AFDX_Properties::networkSelector => A;
END Virtual_Link.A;

VIRTUAL BUS IMPLEMENTATION Virtual_Link.B
  PROPERTIES
    AFDX_Properties::networkSelector => B;
END Virtual_Link.B;
```

An Alternative

```
-------------------------------------------------------------------
--- 2. AFDX Network Configuration
-------------------------------------------------------------------
  VIRTUAL BUS IMPLEMENTATION AFDX_Network.i
    SUBCOMPONENTS
      -- Virtual links
      VL1 : VIRTUAL BUS AFDX::Virtual_Link.dup;
      VL2 : VIRTUAL BUS AFDX::Virtual_Link.dup;
      VL3 : VIRTUAL BUS AFDX::Virtual_Link.dup;
      -- Virtual link path in network A-B
      VL1A : VIRTUAL BUS AFDX::Virtual_Link.A;
      VL1B : VIRTUAL BUS AFDX::Virtual_Link.B;
      VL2A : VIRTUAL BUS AFDX::Virtual_Link.A;
      VL2B : VIRTUAL BUS AFDX::Virtual_Link.B;
      VL3A : VIRTUAL BUS AFDX::Virtual_Link.A;
      VL3B : VIRTUAL BUS AFDX::Virtual_Link.B;
    PROPERTIES
      -- Setup binding A-B virtual links to paths in network A-B
      Actual_Connection_Binding => (reference(VL1A),reference(VL1B))
        applies to VL1;
      Actual_Connection_Binding => (reference(VL2A),reference(VL2B))
        applies to VL2;
      Actual_Connection_Binding => (reference(VL3A),reference(VL3B))
        applies to VL3;
      -- Setup virtual links configuration
      AFDX_Properties::BAG => 1 ms applies to VL1,VL2,VL3;
      AFDX_Properties::Lmax => 1518 Bytes applies to VL1;
      AFDX_Properties::Lmax => 512 Bytes applies to VL2,VL3;
      AFDX_Properties::SkewMax => 1 ms applies to VL1,VL2,VL3;
  END AFDX_Network.i;
```

EMV2 – Virtual Link

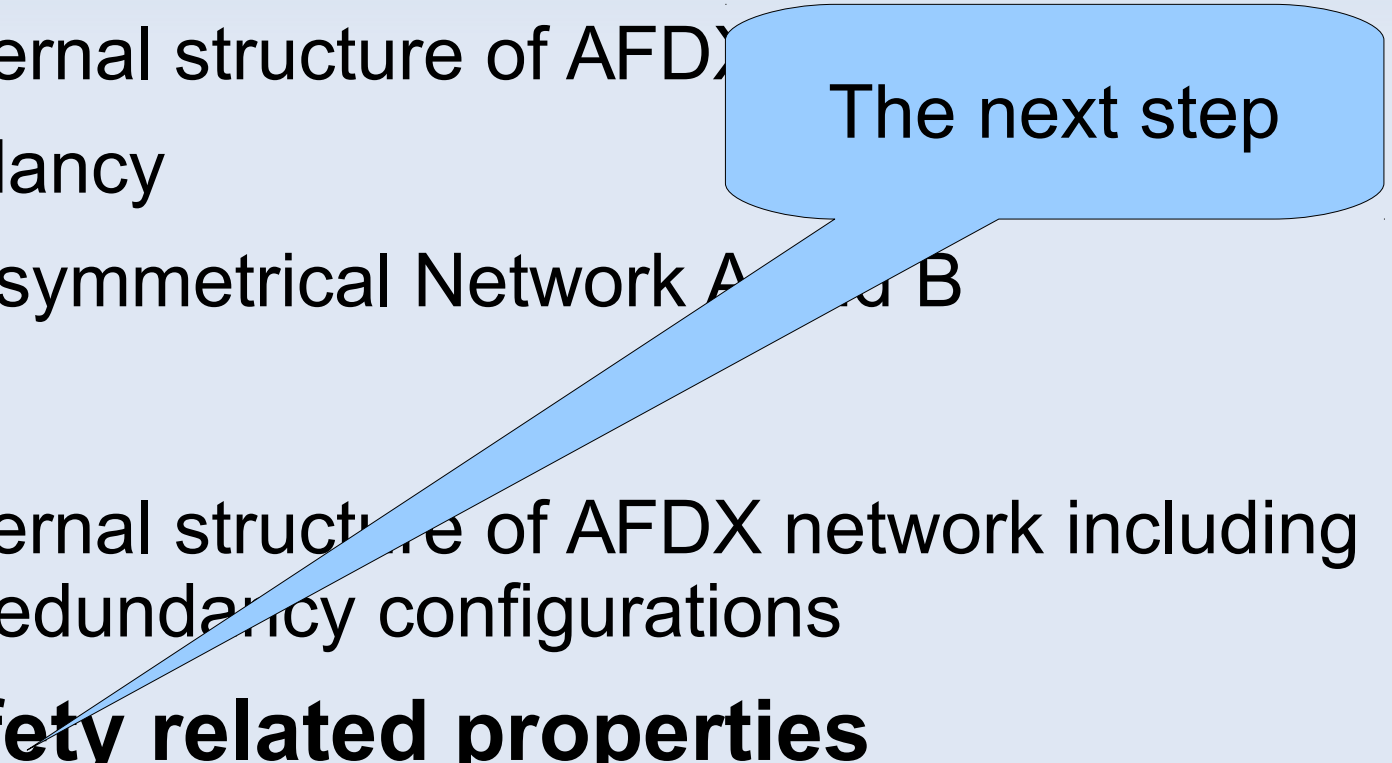
```
  VIRTUAL BUS IMPLEMENTATION Virtual_Link.dup
  annex EMV2 {**
    use types     AFDXErrorLib;
    use behavior  AFDXErrorLib::Twostate;

    error propagations
        connection      : in  propagation {NoService};
        bindings        : out propagation {NoService};
    end propagations;

    component error behavior
    transitions
        fail      : Operational -[ 2 ormore( connection{NoService} ) ]-> Failed;
    propagations
        f : Failed -[]-> bindings(NoService);
    end component;
**};
  END Virtual_Link.dup;
```

AFDX Model Variants

- Pre-Synthesis model
  - Input data for AFDX virtual links generation
- Basic model
  - Represents internal structure of AFDX
  - Ignores redundancy
  - Works well for symmetrical Network A and B
- Detailed model
  - Represents internal structure of AFDX network including asymmetrical redundancy configurations
- **Model with safety related properties**



Update of Basic AFDX Model

```
-------------------------------------------------------------------------
--- 1.1 AFDXWire
-------------------------------------------------------------------------
  BUS Wire
  END Wire;

  BUS TwinWire
  END TwinWire;

-------------------------------------------------------------------------
--- 1.2 AFDXSwitch
-------------------------------------------------------------------------
  ABSTRACT Switch
  END Switch;

  ABSTRACT TwinSwitch
  END TwinSwitch;
```

```
ABSTRACT End_System1
  FEATURES
    afdx : REQUIRES BUS ACCESS TwinWire;
END End_System1;

ABSTRACT End_System2
  FEATURES
    afdxA : REQUIRES BUS ACCESS Wire;
    afdxB : REQUIRES BUS ACCESS Wire;
END End_System2;
```

AFDX Reference Models

- Available at github:

  https://github.com/khoroshilov/aadl-networking-refmodel

- My try to build TTEthernet model is also there

Thank you!

Alexey Khoroshilov
khoroshilov@ispras.ru

http://masiw.ispras.ru