TwIRTee design exploration with Capella and IP-XACT

Philippe Cuenot (IRT Saint-Exupéry / Continental)

Bassem OUNI (IRT Saint-Exupéry)

Pierre Gaufillet (IRT Saint-Exupéry / Airbus)

AIRBUS

Continental

accellera SYSTEMS INITIATIVE

2016 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION EUROPE

Agenda

- Introduction
- Electronic Design process
- Capella to IP-XACT transformation
- TwIRTee rover use case
- Conclusion and perspective

Capella

IP-XACT

Introduction – Capella

- Guidance for **Model Based System Engineering method** (Arcadia)

- Allows interrelated activities for:
  - Needs analysis and modeling
  - Requirement engineering
  - Architecture building and validation

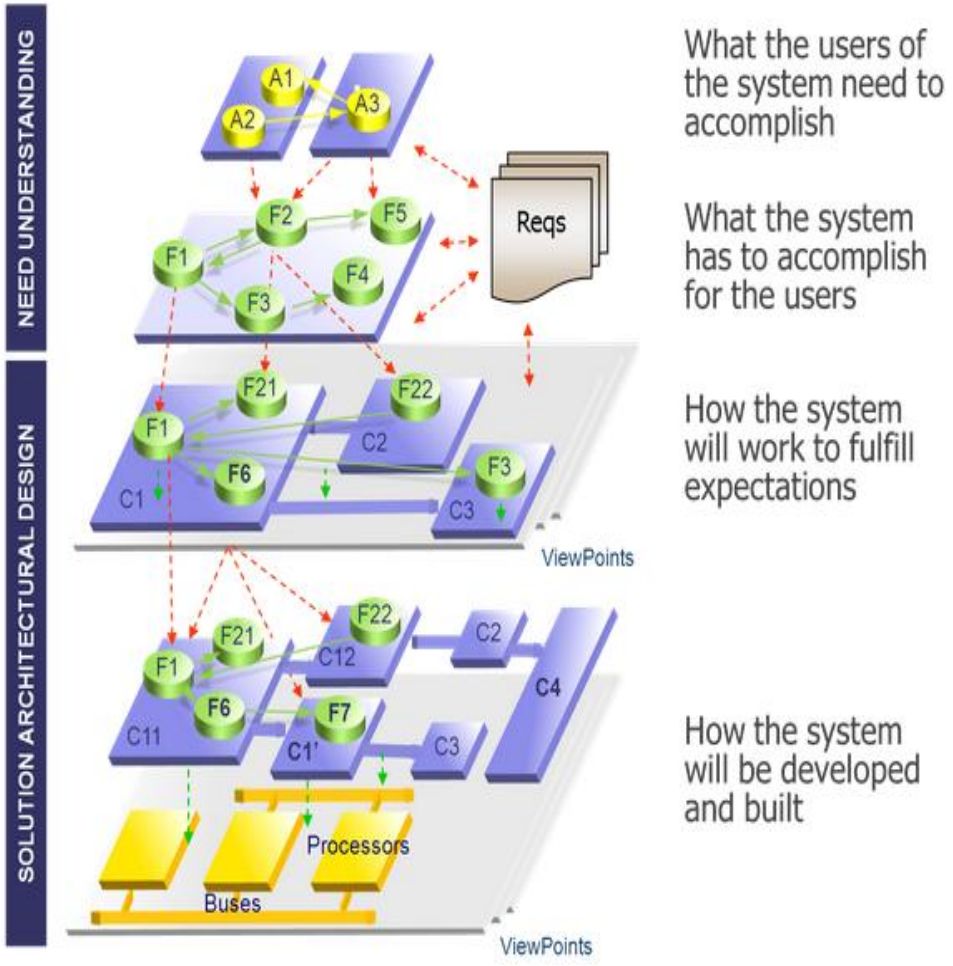
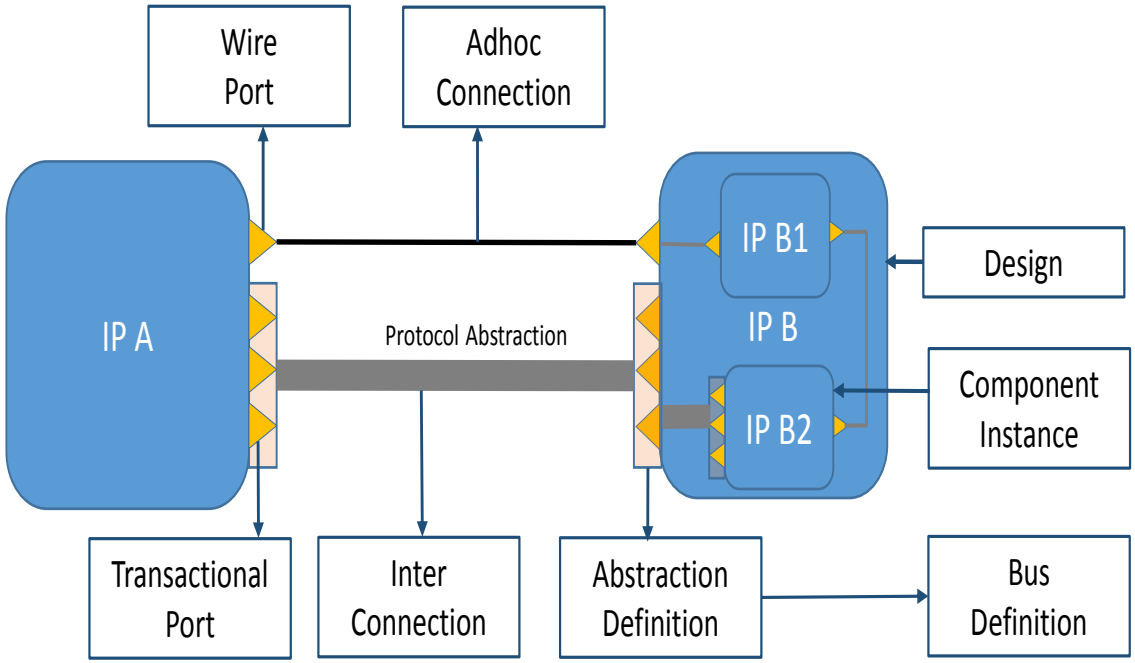


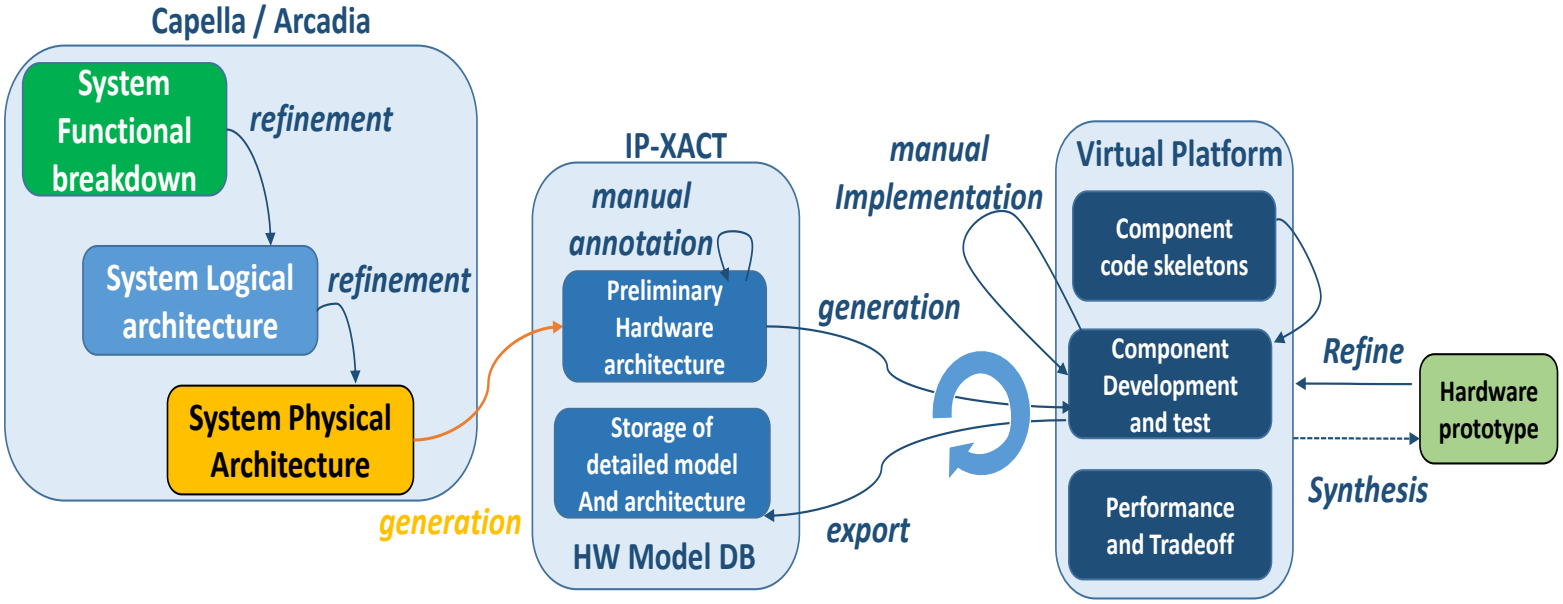
Image from Thales and PFRC presentation Eclipse CON 2015

Introduction – IP-XACT relation

- XSD file for hardware design description
- Matching concept of hardware element of Capella Physical Architecture for hardware



Electronic Design Process



- **System breakdown** in **Capella** down to hardware topology
- **Configuration and assembly** in **IP-XACT**
- **Electronic simulation** with **virtual platform**

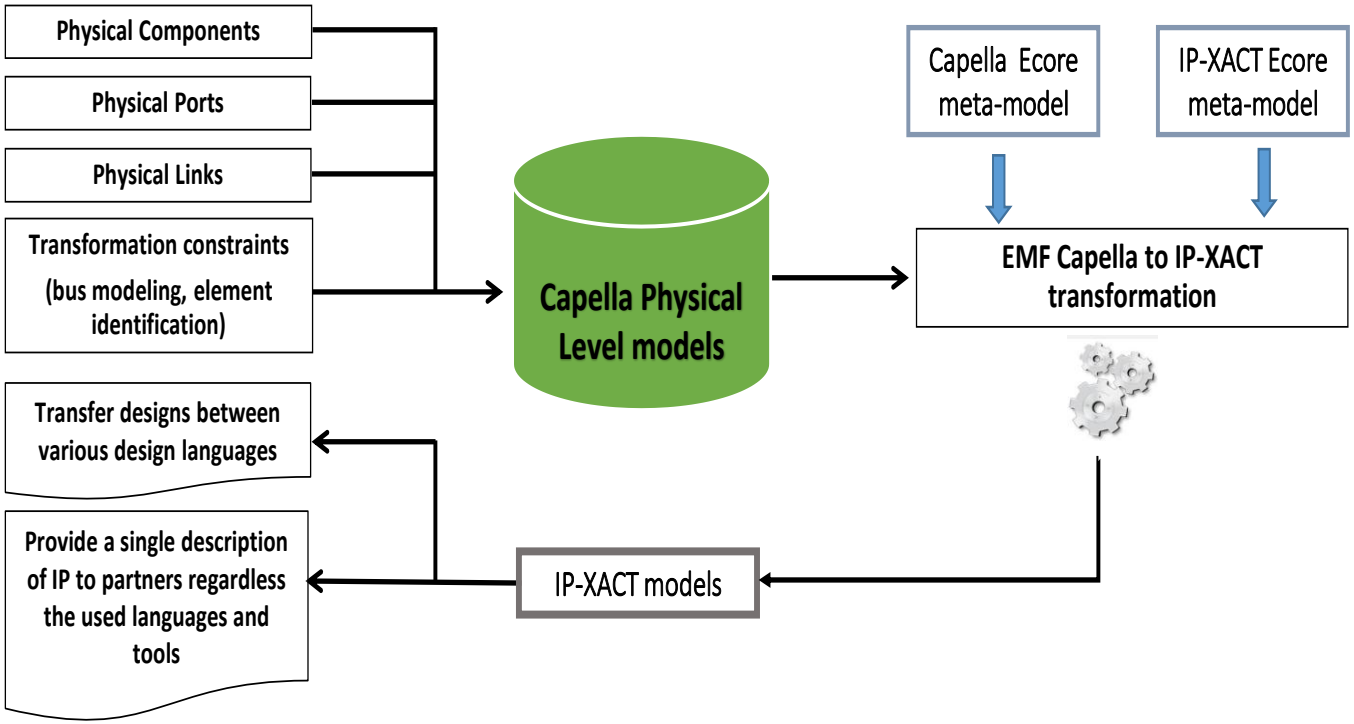
Electronic Design Process

- Typical scenario from Capella physical architect to Virtual Platform simulation (SystemC)
  - Assembly of **SoC connected via serial bus**
  - **Assembly of IP component** interconnected with memory mapped bus and signal (to build a SoC)
  - **White box component design** (targeting Virtual Platform simulation)

- IP-XACT to Virtual Platform transformation is supported by several commercial solutions.

Transforming Capella into IP-XACT

- Uses of **Eclipse Modeling Framework**
- Transforms **hardware elements** of physical architecture



Capella to IP-XACT transformation

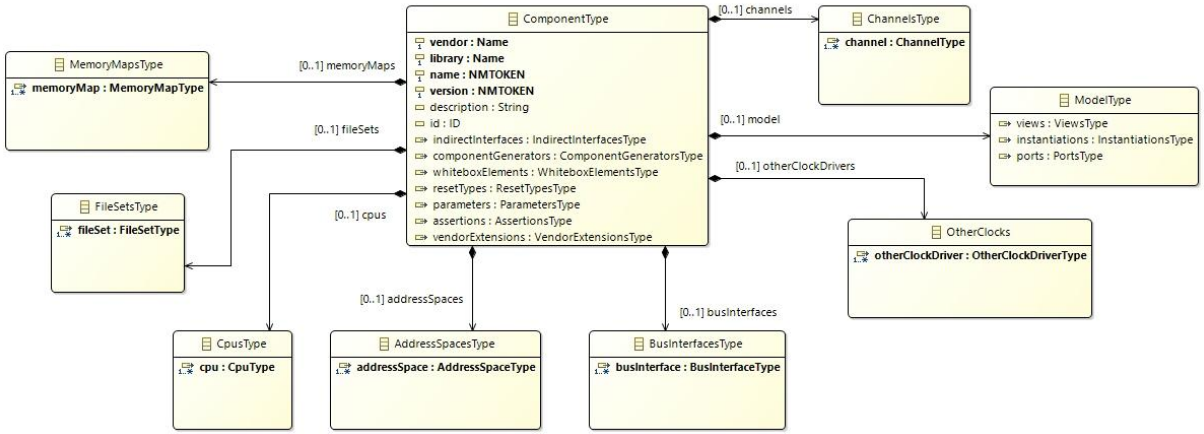
- Mapping concept

Capella	Capella model condition	IP-XACT
Physical Component	Node or Execution Unit	CPU type *cpu*
Physical Component	Others	Component (or design) *Component - Port*
Physical Port	None	Transactional port *Port - Transactional*
Physical Link	Category attribute	Inter Connection *Interconnection/BusRef* Bus Definition / Bus Abstraction *Bus Interface – Abstraction Type – Port Map* *Bus Type – Port*

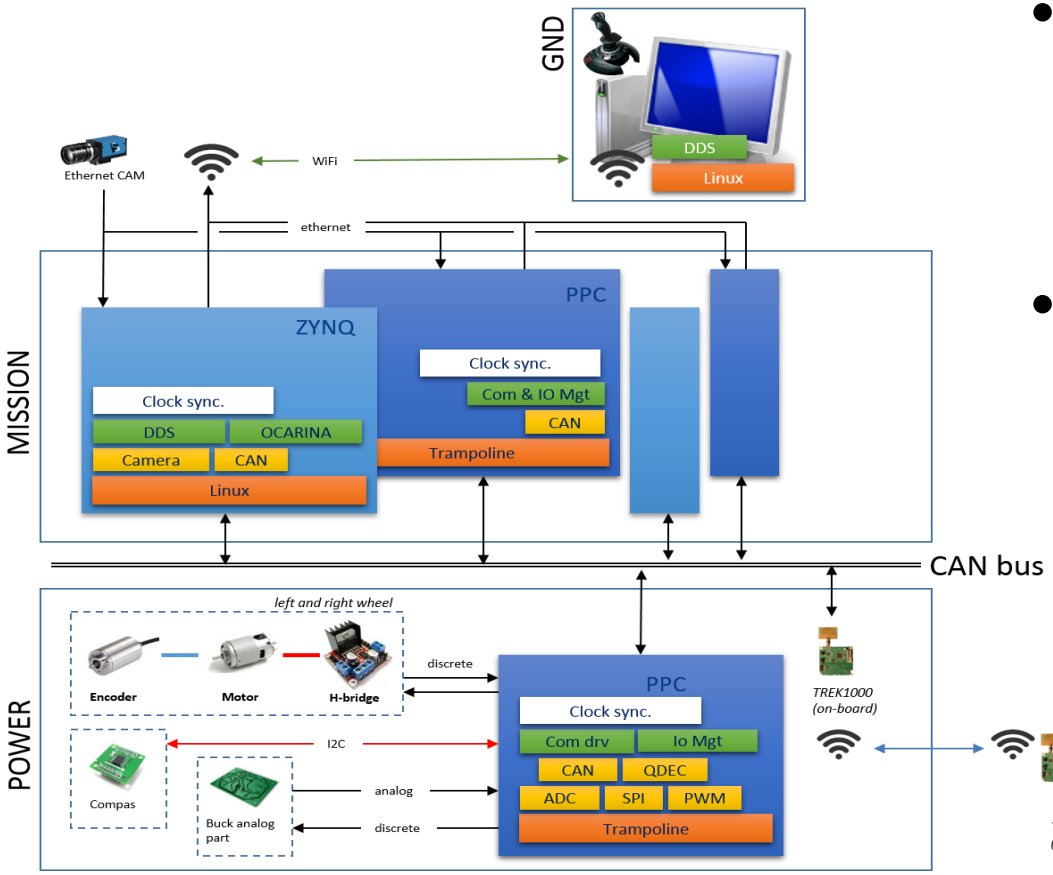
- IP-XACT bus element and connections are build by exploration of component links and connections (segmentation with synthetic port remapped to final bus interface)

Transforming Capella into IP-XACT

- Toward IP-XACT meta-model
  - Fixes a few bugs in the IP-XACT official XSD (IEEE 1685-2014)
  - Builds IP-XACT meta-model from XSD with EMF import tool
  - Generates a simple IP-XACT model editor with EMF
- Benefit to standardize meta-model rather than XSD (and rely on XMI).



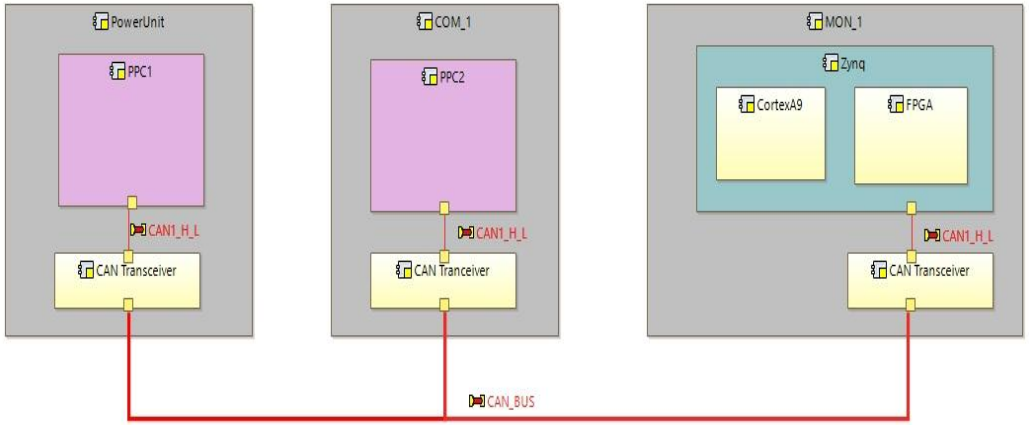
TwIRTee Rover use case



- Safety critical
  - Autonomous trailer
  - Avoid collision
- Electronic architecture
  - PPC : standard SoC
  - Zynq: SoC with FPGA
  - CAN and Ethernet Bus
  - Camera, sensors
  - Redundancy

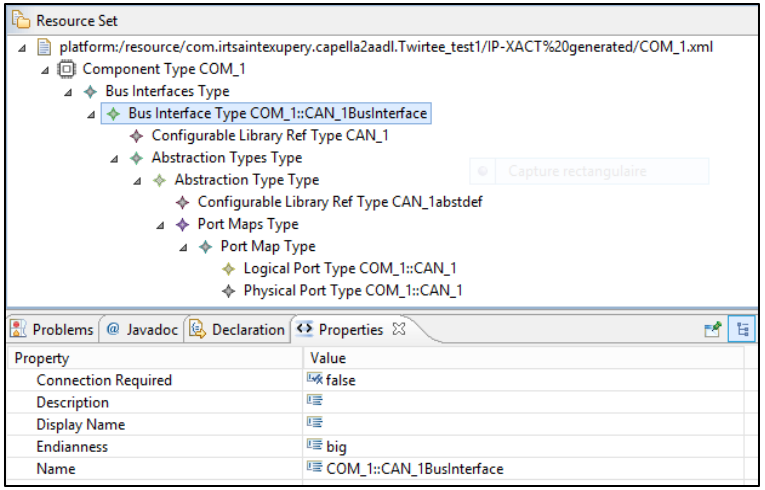
TwIRTee Rover use case

- Use Case 1
  - Assembly of 3 MPC5674F SoCs connected by CAN




```
<?xml version="1.0" encoding="UTF-8"?>
<ipxact:component xmlns:ipxact="http://www.accellera.org/XMLSch
  <ipxact:vendor>COM_1</ipxact:vendor>
  <ipxact:library>COM_1</ipxact:library>
  <ipxact:name>COM_1</ipxact:name>
  <ipxact:version>COM_1</ipxact:version>
  <ipxact:busInterfaces>
    <ipxact:busInterface>
      <ipxact:name>COM_1::CAN_1BusInterface</ipxact:name>
      <ipxact:busType library="CAN_1" name="CAN_1" vendor="CAN_
      <ipxact:abstractionTypes>
        <ipxact:abstractionType>
          <ipxact:abstractionRef library="CAN_1abstdef" name="C
          <ipxact:portMaps>
```

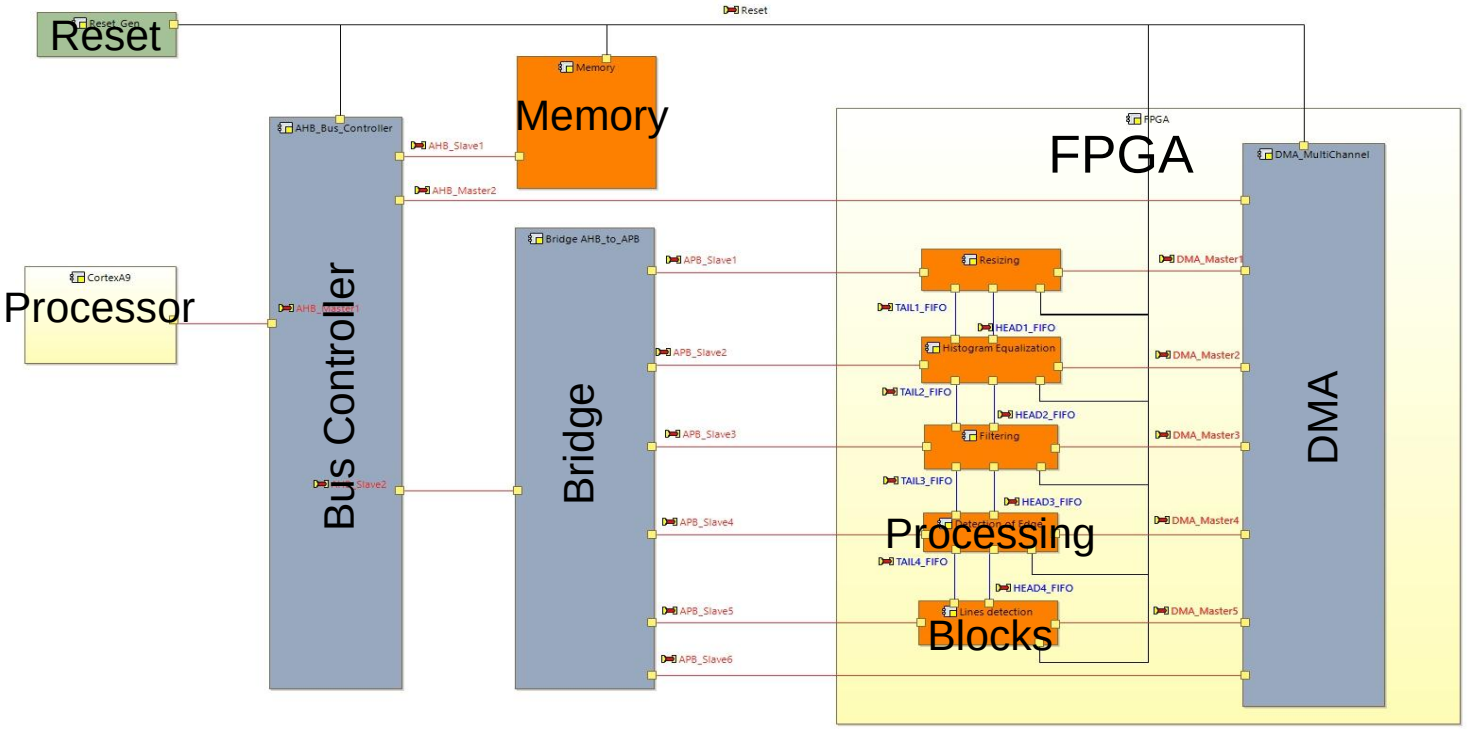
IP-XACT XML file



Eclipse EMF tree view

TwIRTee Rover use case

- Use Case 2
  - Memory mapped interconnected components
  - Manual capture of memory mapping, Register for Blocks…



Conclusion and perspective

- **Tool prototype operational**
  - Bus connection algorithm control seen as work around due to Capella missing concept (extension possible)
  - Actual limitation on simple signal connection
  - perspective for introduction of low level driver information

- Introduction of **AADL** in design flow
  - Integrate software real time properties
  - Unified shared design between software and hardware
  - But requires minor AADL evolutions for specific system/hardware concepts.

Thank you for your attention

Questions ?