2023 DESIGN AND VERIFICATION™ DVCON CONFERENCE AND EXHIBITION JAPAN

IP-XACT IEEE-1685 入門から最新情報まで

Koji Nakamura (Arteris IP)

ARTERIS IP

accellera SYSTEMS INITIATIVE

Agenda

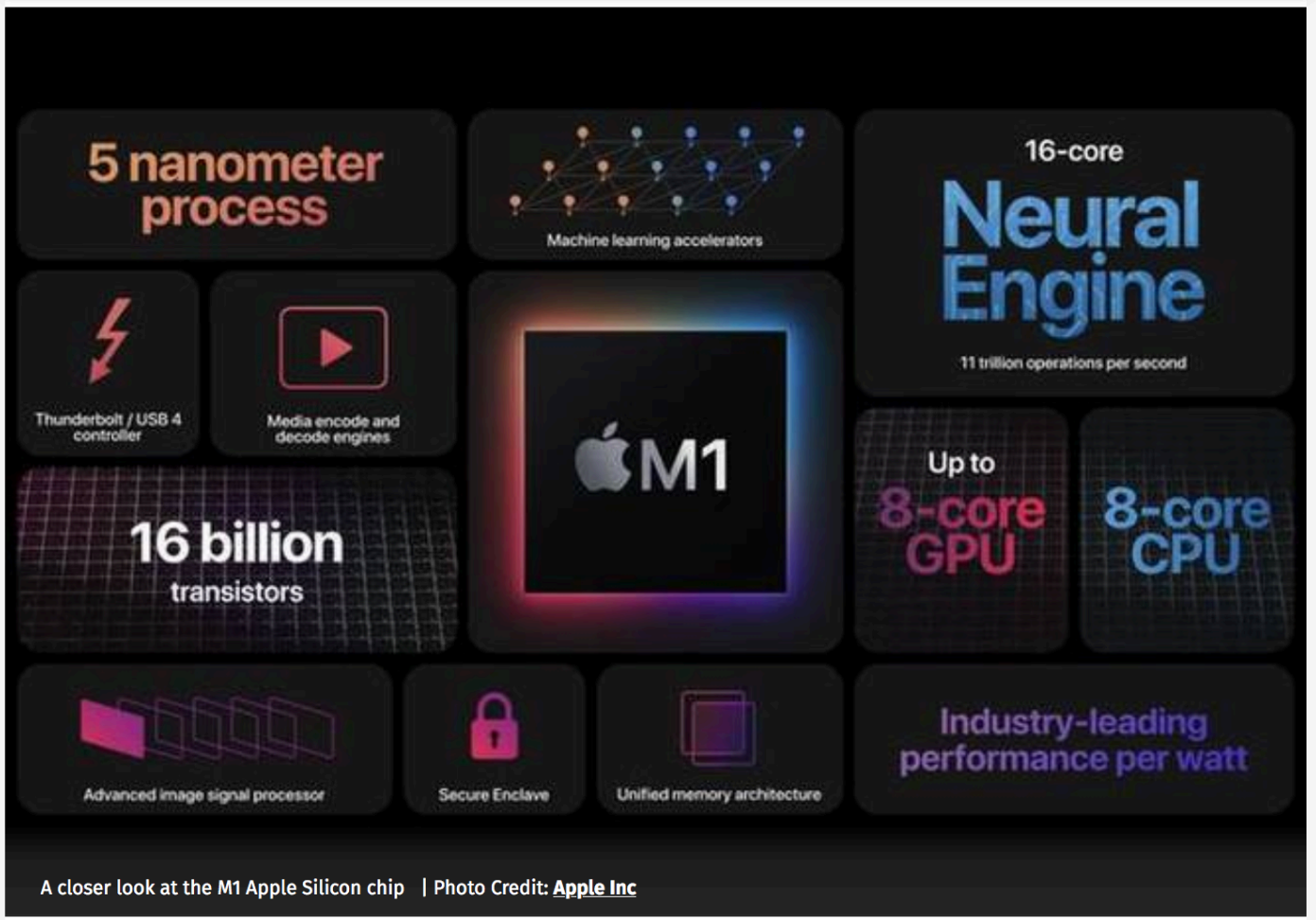
- Why IP-XACT and why we need Tools?
- Why IP-XACT 2022?

Why IP-XACT?

why we need it and why we need tools

Why IP Reuse?

- Too complex to do it by hand
- Reuse work to save time and effort (and avoid errors)
- Simplify the delivery of IP to stakeholders



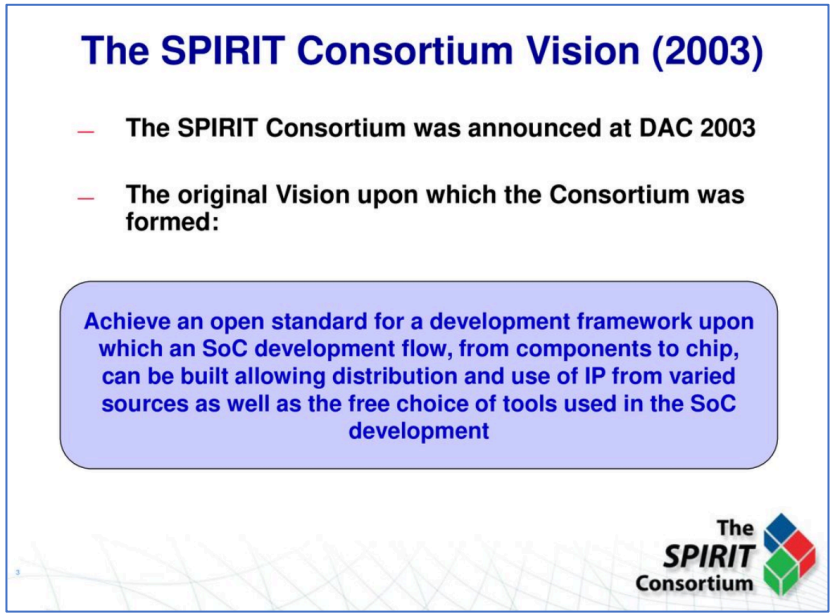
A closer look at the M1 Apple Silicon chip | Photo Credit: Apple Inc

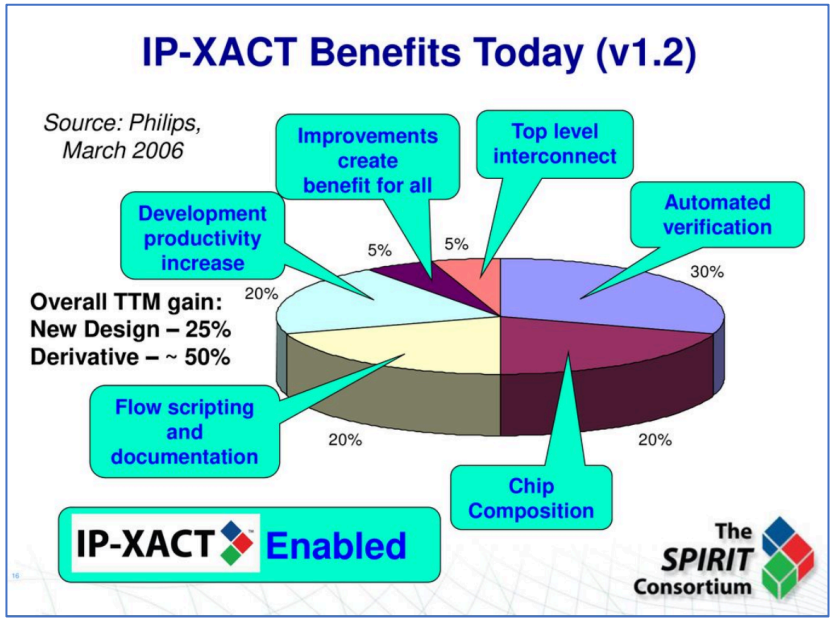
Why IP-XACT?

- Standard to exchange and retarget IP to multiple vendors
- Electronic documentation (easy to parse and process)
- Automate the repetitive tasks by writing Portable generators (using standard API: TGI)
- Easier to certify flows

IP-XACT: the origins

- 20 years old (started in 2003)
- The first (Spirit) consortium was created with IP providers (ARM, Synopsys), SoC integrators (NXP, ST) and EDA vendors (Cadence, Mentor, Synopsys)
- Original XML schema donated by Mentor
  - Main objective was to give a SW view of a HW SoC to align the SW (memory) architecture with the HW (structural) architecture
- Schema updated to cover 2 main additional objectives:
  - Early SoC assembly (required by the SoC integrators)
  - Single source of IP information (required by the IP Vendors) to avoid N qualification of the same IP for N different EDA Vendors

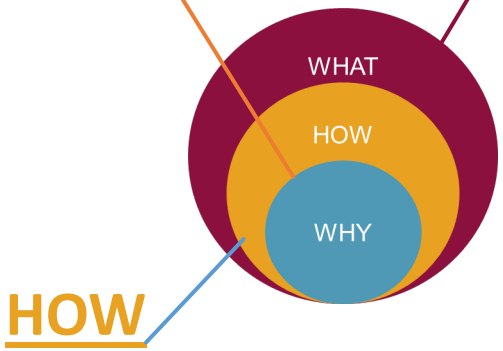




IPXACT: Why / How / What

WHY

- Simplify IP Exchange and Reuse
- Automate tedious and low interest tasks
- Manage Growing complexity
- Limit dependency to IP / EDA providers

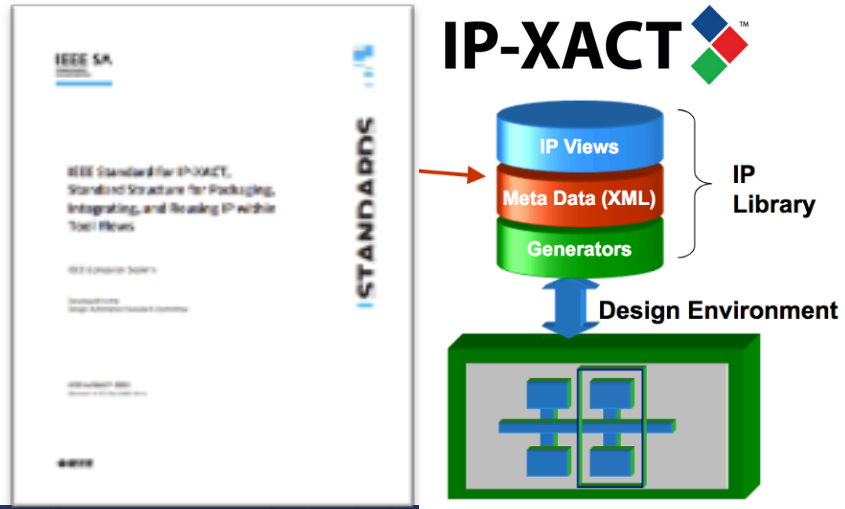


HOW

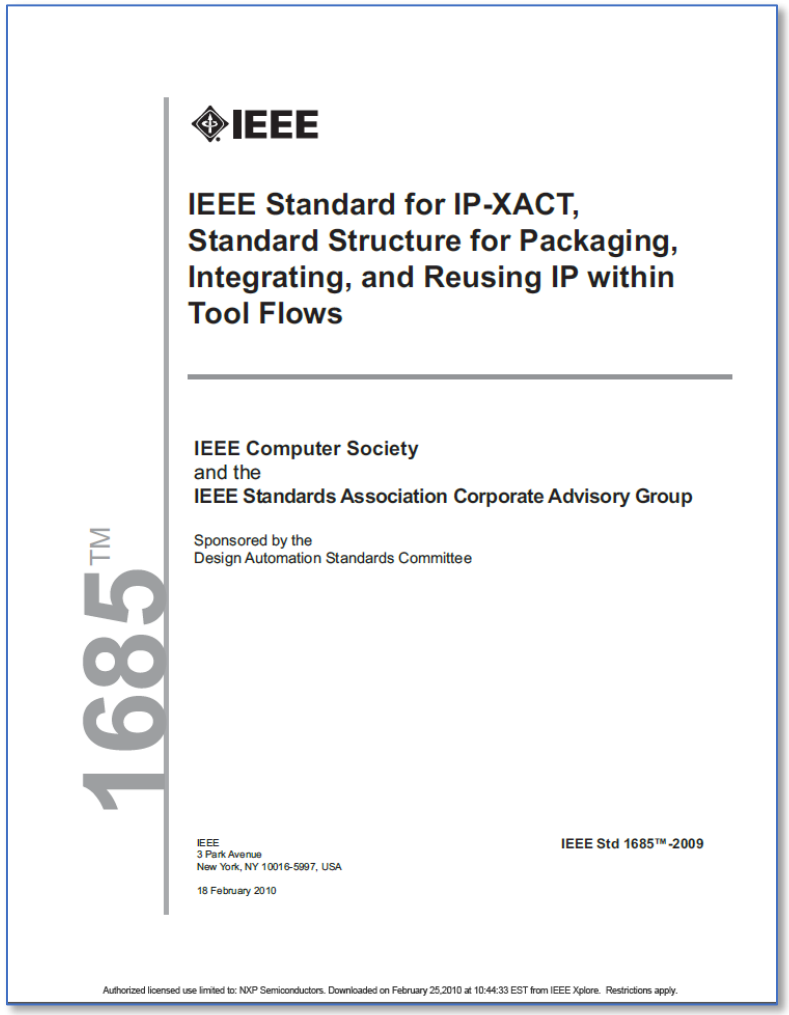
- IEEE standard
- Create a semantic descriptive model
- Promote a large adoption
- Enhance and coexist with other languages

WHAT

- IEEE 1685-20xx PDF
  - Semantic model description
  - Semantic Rules
  - API
- Accellera
  - XML Schema + TGI WSDL/OpenAPI
  - XSLT for up conversion
  - Examples
  - User guide
  - Official extensions



IP-XACT Basics - Quick Reminder



IEEE

IEEE Standard for IP-XACT, Standard Structure for Packaging, Integrating, and Reusing IP within Tool Flows

IEEE Computer Society
and the
IEEE Standards Association Corporate Advisory Group

Sponsored by the
Design Automation Standards Committee

1685™

IEEE
3 Park Avenue
New York, NY 10016-5997, USA

18 February 2010

IEEE Std 1685™-2009

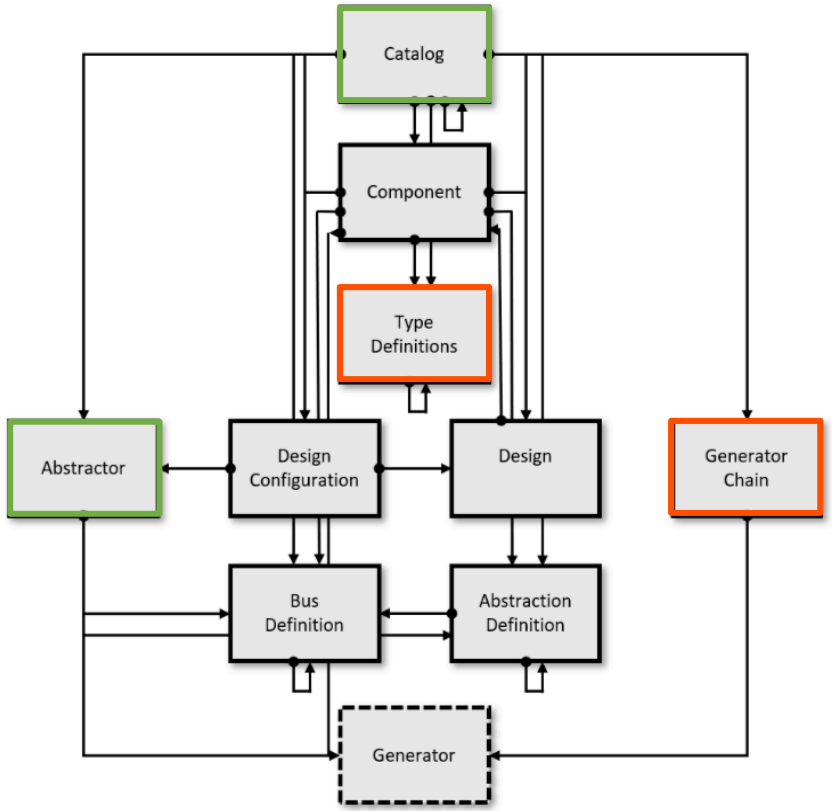


Figure 2—IP-XACT object interactions

Each object identified by a Unique ID:
VLNV

IP-XACT Concepts:

- BusDefinition
- AbstractionDefinition
- Component / Abstractor
  - Ports
  - BusInterface
  - Registers / Fields
  - Filesets / Files
  - Views
  - Modes
- Design
  - Component instances
  - AdhocConnection
  - Interconnection
- DesignConfiguration
  - View selection
  - Parameters configuration
- TypeDefinitions
- GeneratorChain + TGI
- Catalog

IP-XACT Basics - Quick Reminder

The IP-XACT Specification

- Is design language neutral
- Is design tool neutral

Why is this so important?

IP shall be integrated in different D&V flows using different Tool Vendors

- Is efficient
- Is proven
- Is built on the existing XML (W3C) standard
- Includes a standardized API for generator integration (TGI)
- Validated and released in accordance with the IEEE policies

- IP-XACT is Not Yet Another Electronic Language
  - Does not replace existing HDL (VHDL, Verilog, SystemVerilog, SystemRDL, SystemC, C++…). But refers to them.
  - Forms an electronic databook exposing IP most significant characteristics
  - Provides independence over EDA Tools thanks to its standard language neutral API: **TGI**

IP-XACT is an XML format…

Component instances and connections in a Design

```
    </spirit:componentInstance>
    <spirit:componentInstance>
        <spirit:instanceName>urgu</spirit:instanceName>
        <spirit:componentRef spirit:library="Leon2RTL" spirit:name="rgu" spirit:vendor="spiritconsortium.org" spirit:version="1.2"/>
        <spirit:configurableElementValues>
            <spirit:configurableElementValue spirit:referenceId="ClockPeriod">100</spirit:configurableElementValue>
            <spirit:configurableElementValue spirit:referenceId="ClockPulseDuration">50</spirit:configurableElementValue>
            <spirit:configurableElementValue spirit:referenceId="ClockPulseOffset">50</spirit:configurableElementValue>
            <spirit:configurableElementValue spirit:referenceId="ClockPulseValue">1</spirit:configurableElementValue>
        </spirit:configurableElementValues>
    </spirit:componentInstance>
</spirit:componentInstances>
<spirit:interconnections>
    <spirit:interconnection>
        <spirit:name>defaultid4490051</spirit:name>
        <spirit:activeInterface spirit:busRef="AHBmaster" spirit:componentRef="uproc"/>
        <spirit:activeInterface spirit:busRef="MirroredMaster0" spirit:componentRef="uahbbus"/>
    </spirit:interconnection>
    <spirit:interconnection>
        <spirit:name>defaultid4490067</spirit:name>
        <spirit:activeInterface spirit:busRef="ambaAHB" spirit:componentRef="udma"/>
        <spirit:activeInterface spirit:busRef="MirroredMaster1" spirit:componentRef="uahbbus"/>
    </spirit:interconnection>
    <spirit:interconnection>
        <spirit:name>defaultid4490084</spirit:name>
        <spirit:activeInterface spirit:busRef="AHBmaster" spirit:componentRef="i_ahb2ahb_1"/>
        <spirit:activeInterface spirit:busRef="MirroredMaster2" spirit:componentRef="uahbbus"/>
    </spirit:interconnection>
    <spirit:interconnection>
        <spirit:name>defaultid4490101</spirit:name>
        <spirit:activeInterface spirit:busRef="AHBSlave" spirit:componentRef="uahbram"/>
        <spirit:activeInterface spirit:busRef="MirroredSlave0" spirit:componentRef="uahbbus"/>
```

Component memoryMap / Registers

```
<spirit:memoryMaps>
    <spirit:memoryMap>
        <spirit:name>ambaAPB</spirit:name>
        <spirit:addressBlock>
            <spirit:name>defaultid4489950</spirit:name>
            <spirit:baseAddress spirit:resolve="immediate" spirit:format="long" spirit:rangeType="float" spirit:prompt="Base Address:">0</spiri
            <spirit:range spirit:format="long">16</spirit:range>
            <spirit:width spirit:id="width" spirit:format="long">32</spirit:width>
            <spirit:register>
                <spirit:name>data</spirit:name>
                <spirit:displayName>data</spirit:displayName>
                <spirit:description>Data read/write register</spirit:description>
                <spirit:addressOffset>0x0</spirit:addressOffset>
                <spirit:size>32</spirit:size>
                    it:volatile>true</spirit:volatile>
                    it:access>read-write</spirit:access>
                    register>
                    egister>
                    it:name>status</spirit:name>
                <spirit:displayName>status</spirit:displayName>
                                                    pirit:description>
                                                    essOffset>
                                                    s>
```

Component parameters

```
                <spirit:constraintSets>
                    <spirit:constraintSet spirit:constraintSetId="default">
                        <spirit:timingConstraint spirit:clockEdge="rise" spirit:clockName="virtual_clk">75.0</spirit:timingConstraint>
                    </spirit:constraintSet>
                </spirit:constraintSets>
            </spirit:wire>
        </spirit:port>
    </spirit:ports>
    <spirit:modelParameters>
        <spirit:modelParameter spirit:dataType="boolean">
            <spirit:name>EXTBAUD</spirit:name>
            <spirit:value spirit:id="EXTBAUD" spirit:resolve="user" spirit:choiceRef="EXTBAUDChoice" spirit:configGroups="requiredConfig" spirit:
        </spirit:modelParameter>
    </spirit:modelParameters>
</spirit:model>
<spirit:choices>
    <spirit:choice>
        <spirit:name>EXTBAUDChoice</spirit:name>
        <spirit:enumeration spirit:text="false">false</spirit:enumeration>
        <spirit:enumeration spirit:text="true">true</spirit:enumeration>
    </spirit:choice>
</spirit:choices>
<spirit:fileSets>
    <spirit:fileSet>
        <spirit:name>fs-vhdlSource</spirit:name>
        <spirit:file>
            <spirit:name spirit:resolve="immediate" spirit:format="string" spirit:rangeType="float">../../common/target.vhd</spirit:name>
            <spirit:fileType>vhdlSource</spirit:fileType>
            <spirit:logicalName>leon2uart_lib</spirit:logicalName>
        </spirit:file>
```

IP-XACT is an XML format... you do not want to edit by hand

Component instances and connections in a Design

```
    </spirit:componentInstance>
    <spirit:componentInstance>
        <spirit:instanceName>urgu</spirit:instanceName>
        <spirit:componentRef spirit:library="Leon2RTL" spirit:name="rgu" spirit:vendor="spiritconsortium.org" spirit:version="1.2"/>
        <spirit:configurableElementValues>
            <spirit:configurableElementValue spirit:referenceId="ClockPeriod">100</spirit:configurableElementValue>
            <spirit:configurableElementValue spirit:referenceId="ClockPulseDuration">50</spirit:configurableElementValue>
            <spirit:configurableElementValue spirit:referenceId="ClockPulseOffset">50</spirit:configurableElementValue>
            <spirit:configurableElementValue spirit:referenceId="ClockPulseValue">1</spirit:configurableElementValue>
        </spirit:configurableElementValues>
    </spirit:componentInstance>
</spirit:componentInstances>
<spirit:interconnections>
    <spirit:interconnection>
        <spirit:name>defaultid4490051</spirit:name>
        <spirit:activeInterface spirit:busRef="AHBmaster" spirit:componentRef="uproc"/>
        <spirit:activeInterface spirit:busRef="MirroredMaster0" spirit:componentRef="uahbbus"/>
    </spirit:interconnection>
    <spirit:interconnection>
        <spirit:name>defaultid4490067</spirit:name>
        <spirit:activeInterface spirit:busRef="ambaAHB" spirit:componentRef="udma"/>
        <spirit:activeInterface spirit:busRef="MirroredMaster1" spirit:componentRef="uahbbus"/>
    </spirit:interconnection>
    <spirit:interconnection>
        <spirit:name>defaultid4490084</spirit:name>
        <spirit:activeInterface spirit:busRef="AHBmaster" spirit:componentRef="i_ahb2ahb_1"/>
        <spirit:activeInterface spirit:busRef="MirroredMaster2" spirit:componentRef="uahbbus"/>
    </spirit:interconnection>
    <spirit:interconnection>
        <spirit:name>defaultid4490101</spirit:name>
        <spirit:activeInterface spirit:busRef="AHBSlave" spirit:componentRef="uahbram"/>
        <spirit:activeInterface spirit:busRef="MirroredSlave0" spirit:componentRef="uahbbus"/>
```

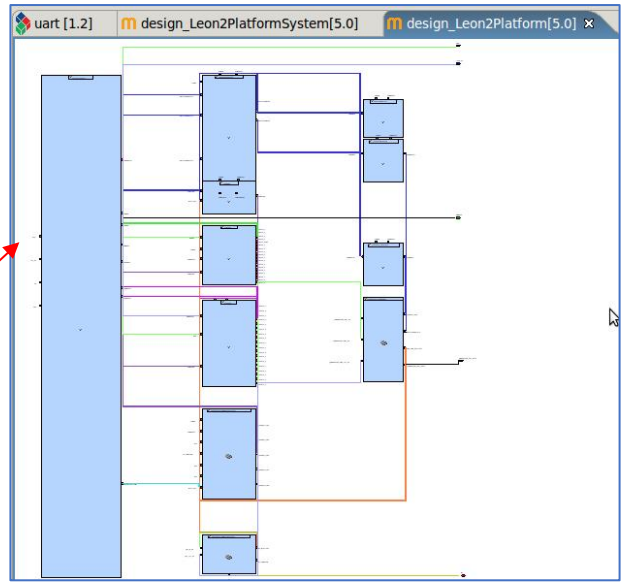
Component memoryMap / Registers

```
<spirit:memoryMaps>
    <spirit:memoryMap>
        <spirit:name>ambaAPB</spirit:name>
        <spirit:addressBlock>
            <spirit:name>defaultid4489950</spirit:name>
            <spirit:baseAddress spirit:resolve="immediate" spirit:format="long" spirit:rangeType="float" spirit:prompt="Base Address:">0</spiri
            <spirit:range spirit:format="long">16</spirit:range>
            <spirit:width spirit:id="width" spirit:format="long">32</spirit:width>
            <spirit:register>
                <spirit:name>data</spirit:name>
                <spirit:displayName>data</spirit:displayName>
                <spirit:description>Data read/write register</spirit:description>
                <spirit:addressOffset>0x0</spirit:addressOffset>
                <spirit:size>32</spirit:size>
                    it:volatile>true</spirit:volatile>
                    it:access>read-write</spirit:access>
                    register>
                    egister>
                    it:name>status</spirit:name>
```

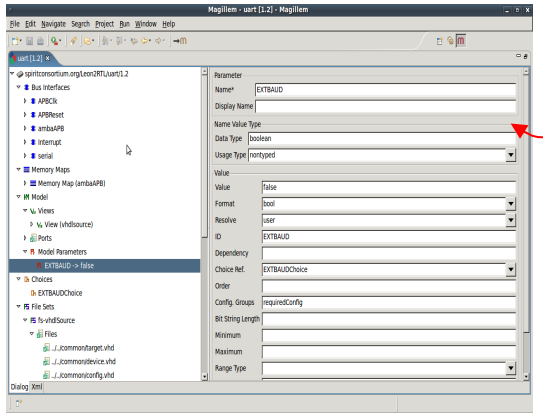
Component parameters

```
            <spirit:constraintSets>
                <spirit:constraintSet spirit:constraintSetId="default">
                    <spirit:timingConstraint spirit:clockEdge="rise" spirit:clockName="virtual_clk">75.0</spirit:timingConstraint>
                </spirit:constraintSet>
            </spirit:constraintSets>
        </spirit:wire>
    </spirit:port>
</spirit:ports>
<spirit:modelParameters>
    <spirit:modelParameter spirit:dataType="boolean">
        <spirit:name>EXTBAUD</spirit:name>
        <spirit:value spirit:id="EXTBAUD" spirit:resolve="user" spirit:choiceRef="EXTBAUDChoice" spirit:configGroups="requiredConfig" spirit:
    </spirit:modelParameter>
</spirit:modelParameters>
</spirit:model>
<spirit:choices>
    <spirit:choice>
        <spirit:name>EXTBAUDChoice</spirit:name>
        <spirit:enumeration spirit:text="false">false</spirit:enumeration>
        <spirit:enumeration spirit:text="true">true</spirit:enumeration>
    </spirit:choice>
</spirit:choices>
<spirit:fileSets>
    <spirit:fileSet>
        <spirit:name>fs-vhdlSource</spirit:name>
        <spirit:file>
            <spirit:name spirit:resolve="immediate" spirit:format="string" spirit:rangeType="float">../../common/target.vhd</spirit:name>
            <spirit:fileType>vhdlSource</spirit:fileType>
            <spirit:logicalName>leon2uart_lib</spirit:logicalName>
        </spirit:file>
```

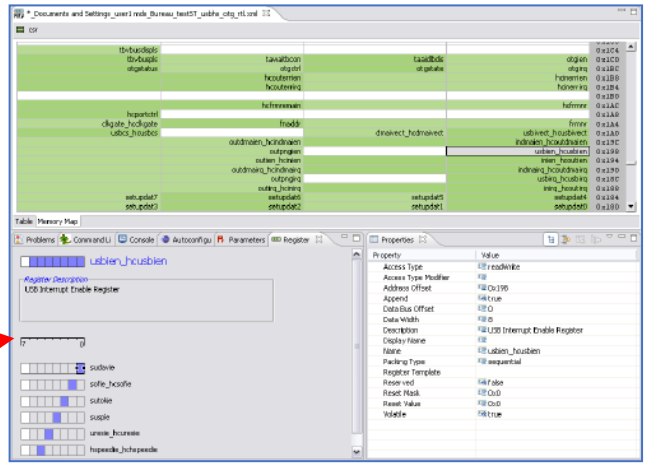
Connectivity Editor



IP-XACT Editor

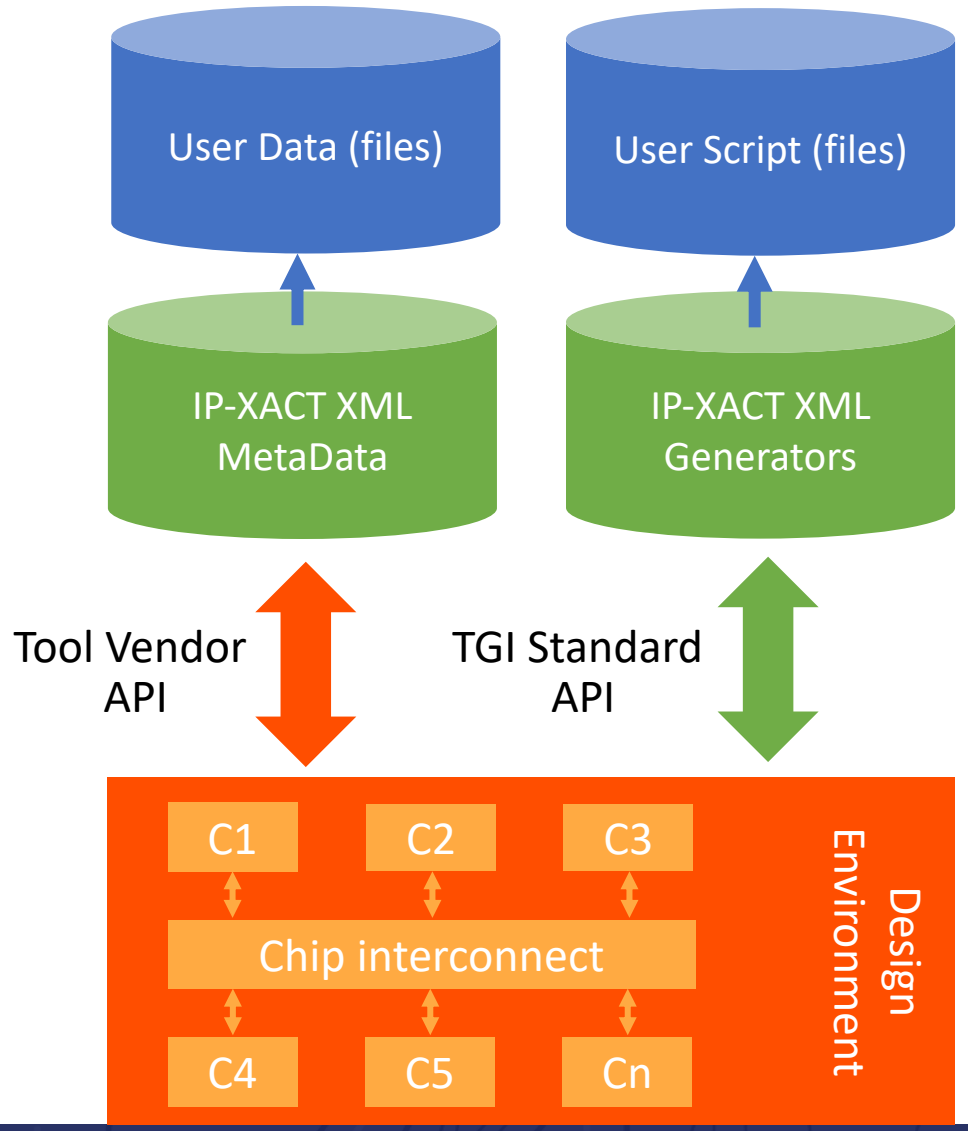


Register Editor



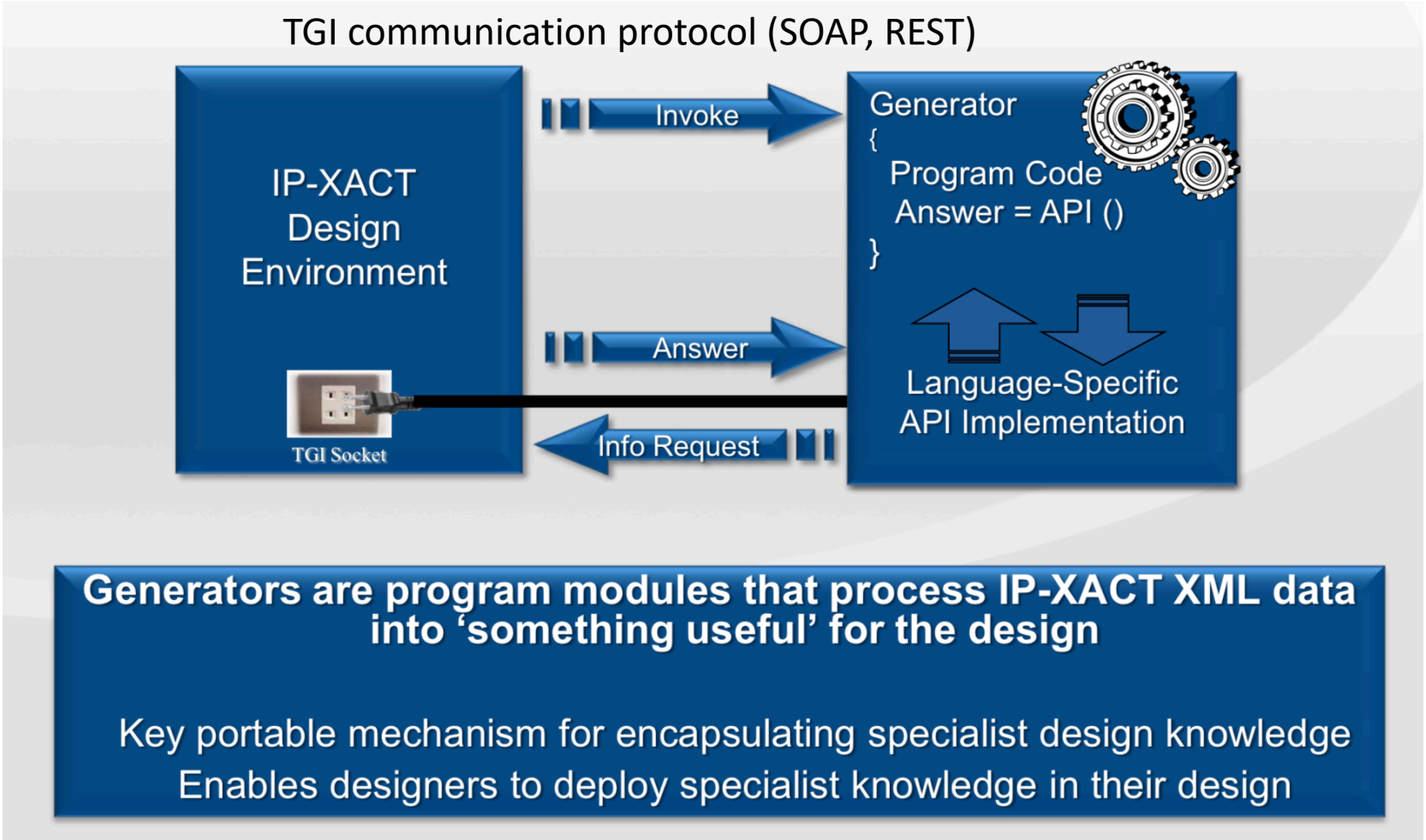
IP-XACT is an XML format... you can process

- You can write Portable (standard) Generators that can be plugged in any IP-XACT Design Environment
- All IP-XACT Compliant tool shall support the TGI
- Generators can be used to expand / customize the Tool features as you desire



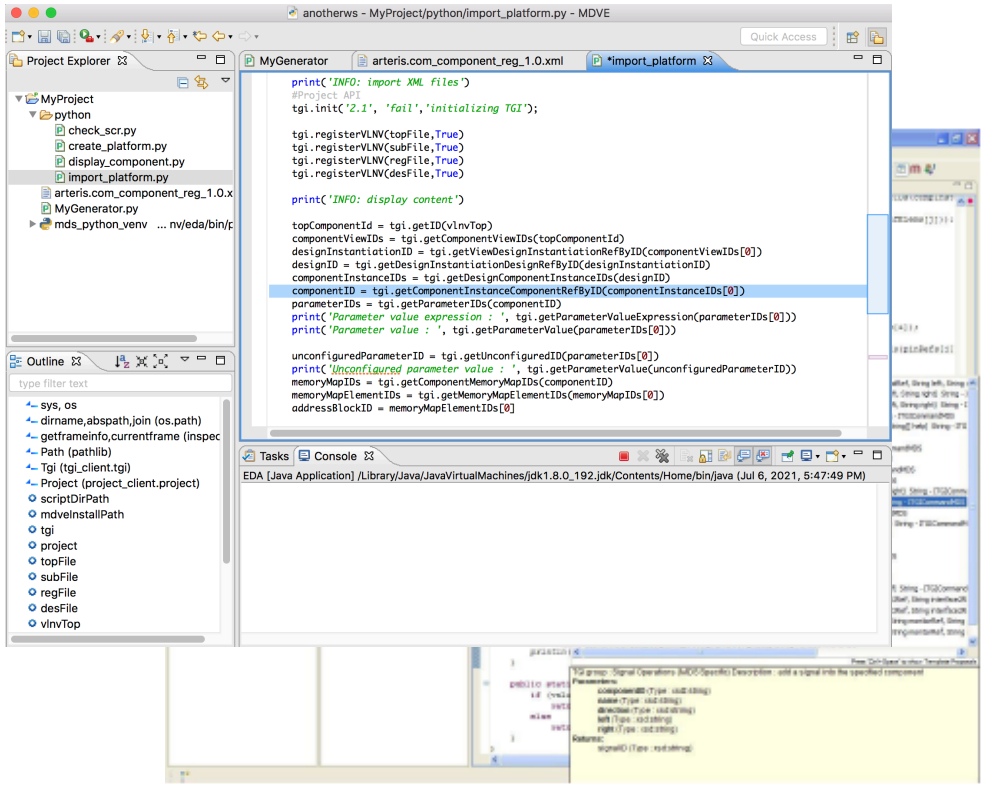
IP-XACT processing using TGI

- Generators can be grouped into generator Chains and invoked from the Design Environment
  - Combining individual generators enables the creation of a custom functionality or flow
- Generators can also be attached to Components
  - Only activated when the component is added to a Design



Need tools to help writing (TGI) generators

- Write scripts or code (generators) to automate the flow
- The tool provides
  - Code writing assistance with integrated documentation.
  - On the fly compilation: no syntax errors
  - Native TGI API support
  - Java + Python + TCL development environment providing:
    - Completion assistance,
    - debugging environment (Step by Step, tracing, hot code replace…)



Why IP-XACT 2022?

THE NEW Standard

From IEEE 1685-2009 to 2014 to 2022

IP-XACT 2022 benefits for users

- Why adopting the new IP-XACT standard IEEE 1685-2022?
  - More straightforward and more complete schema to support todays and tomorrow's complex IP and SoCs
  - Includes all of 2009 and 2014
- Why IEEE 1685-2014 was not broadly adopted?
  - Conditionality was complex to implement and validate (more specifically at design level)
  - Still many features were missing to represent complex memory and connectivity objects
  - Limited support from IP and Tool vendors
- Compared to IP-XACT 2009, the new proposed IP-XACT standard includes
  - Better support of Memory objects definition
    - external definitions, arrays, conditional accesses, resets, sharing, aliasing and broadcasting…
  - Better support of Connectivity
    - tie, broadcast, SystemVerilog interfaces, structs and unions, VHDL records, SystemC sockets...
  - Better support of Parameters propagation and Expressions
    - replace Xpath by SystemVerilog expressions, mix multiple IP views
  - Support of editing API
    - TGI extended

IP-XACT 2022 major features (vs. 2009)

- New root objects:
  - Catalog: like the Magillem 2009 catalog, but now in standard
  - TypeDefinitions: new object including (nested) definitions of all memory objects

9.1 Type definitions

An IP-XACT *typeDefinitions* is the central placeholder for the definition of memory-related objects meta-data. A typeDefinitions lists definitions of field access policies, fields, registers, register files, address blocks, banks, memory maps, and memory remaps. These definitions can be configured and referenced in components and other typeDefinitions.

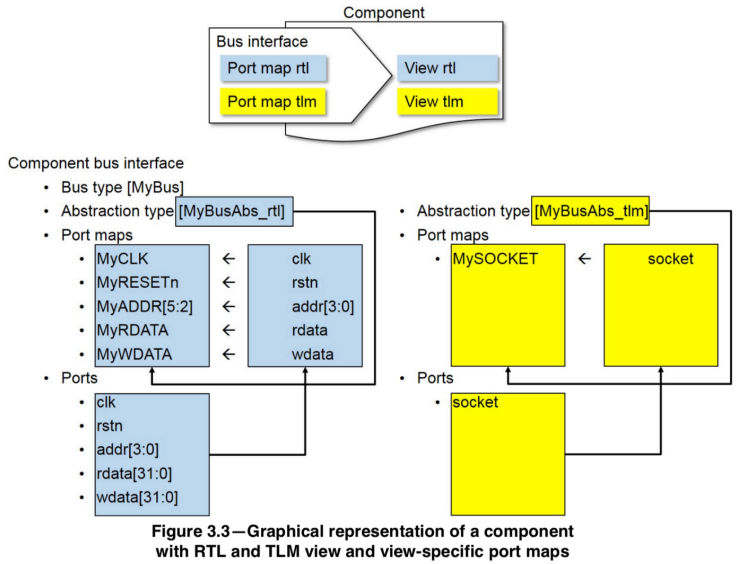


Figure 3.3—Graphical representation of a component with RTL and TLM view and view-specific port maps

- General changes:
  - Expressions (SV vs. XPath)
  - Added Assertions
  - Added support for multiple leaf and hierarchical views
  - Added AMS and Power support (added power domains and links)
  - Generalize Vendor Extensions
  - Conditionality (isPresent) defined as a Vendor Extension
  - Added name group (with VLNV, displayName, shortDescription and description), on all top-level objects

IP-XACT 2022 major features (vs. 2009)

- Changes in Protocols (abstractionDef):
  - Support Serial and Multiplexed busses (add packets on logical ports)
  - Describe matching widths
  - Added new qualifiers on logical ports (isValid, isInterrupt, isRequest, isResponse…)
- Changes in Component ports, interfaces and views:
  - Added Structured Ports (to support SVI, Unions and VHDL Records) **-> Example**
  - Added Qualifiers on Ports (isInterrupt, isReset, isClock…)
  - Added Parameters on Ports
  - Changed Transactional ports to support SystemC TLM sockets
  - Linked Ports and register Fields **-> Example**
  - Added HDL properties in views (packageName, libraryName, architectureName…)
  - Added Runtime model parameters
  - View dependant interfaces/ports and parameters

Example: Structured port – SV interface

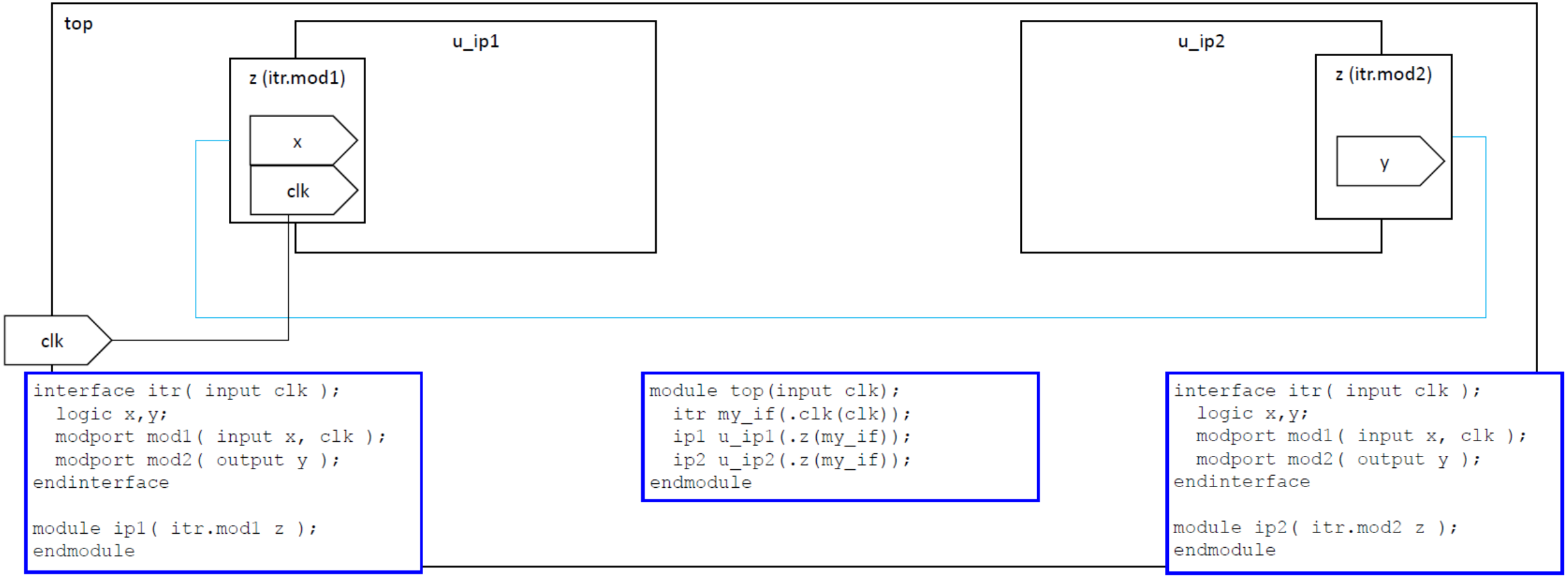
SV interface

```
name=z
structured
        interface
        subPort, isIO=true
                name=clk
                wire
                        direction=in
        subPort
                name=x
                wire
                        direction=inout
        structPortTypeDef
                typeName=itr
                role=mod1
```

```
interface itr( input clk );
  logic x,y;
  modport mod1( input x, clk );
  modport mod2( output y );
endinterface


module ip( itr.mod1 z );
endmodule
```

Example: Structured port – SVI connection




```
interface itr( input clk );
  logic x,y;
  modport mod1( input x, clk );
  modport mod2( output y );
endinterface

module ip1( itr.mod1 z );
endmodule
```

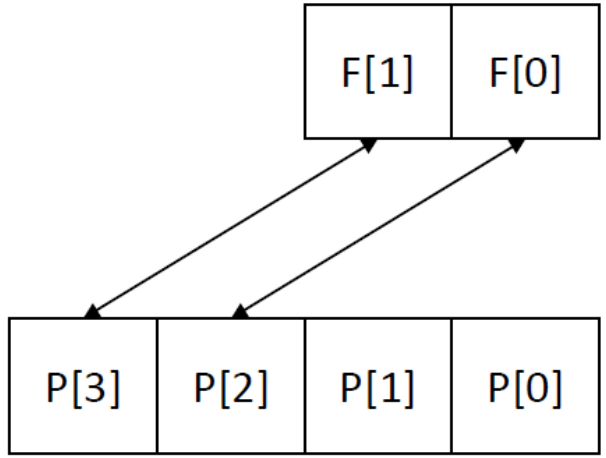
```
module top(input clk);
  itr my_if(.clk(clk));
  ip1 u_ip1(.z(my_if));
  ip2 u_ip2(.z(my_if));
endmodule
```

```
interface itr( input clk );
  logic x,y;
  modport mod1( input x, clk );
  modport mod2( output y );
endinterface

module ip2( itr.mod2 z );
endmodule
```

Example: Field map

```
register
        name=R
        field
                name=F
                bitOffset=0
                bitWidth=2
port
        name=P
        vector=[3:0]
        fieldMap
                fieldSlice
                        registerRef=R
                        fieldRef=F
                range=[3:2]
                modeRef=FUNC
```



IP-XACT 2022 major features (vs. 2009)

- Changes in Component Memory objects:
  - Improved HDL access Handle, including string expressions with index variables
  - Added modes, modes condition (using SV expression) and mode references permitting the support of secure registers **-> Example**
  - Added access policies and access restrictions
  - Added field and register aliasing & broadcasting
  - Extended fields enumerated values, and added enumerations references
  - Added Reset at bitField level + support for Multiple resets
  - Added CPU memory map using references to addressSpaces and segments of addressSpaces
  - Added arrays (dim), indices, stride and bitStride to memory objects

Example: Mode-dependent register access

component modes

```
mode
        name=TEST
        portSlice=myPortSlice
                portRef=test_mode
        condition=$ipxact_port_value(myPortSlice)==1
mode
        name=FUNC
        portSlice=myPortSlice
                portRef=test_mode
        condition=$ipxact_port_value(myPortSlice)==0
```

register access policy

```
register
        name=myReg
        addressOffset=0
        size=32
        accessPolicy
                modeRef=TEST
                access=read-write
        accessPolicy
                modeRef=FUNC
                access=read-only
```

A condition expression can contain terms for port slice values, register field slice values, and mode condition values

IP-XACT 2022 major features (vs. 2009)

- Changes in Design connectivity and Design configuration:
  - Improved Design connections: top feedthrough, top tied ports, interconnections (N vs. 2)…
  - Support exclude logical port mechanism in interface connection
  - Support isVirtual instance (that should not be netlisted)
  - Added Parameters propagation through hierarchy
- TGI
  - Base (get)
  - Extended (create/set/add/remove)
  - Supports REST (in addition to SOAP) communication protocol
- SCRs
  - + 100 SCR added vs. 2009

質疑応答