Input to SystemVerilog Requirements Gathering Meeting

Gary Delp
Gary.Delp@gmail.com,
Gary.Delp@Silverloonsystems.com,
Gary.Delp@p1801.org,
Delp.Gary@mayo.edu
(for connectivity but not attribution)

First – Do no harm

- Gain experience – e.g. 1801
- Potential synergy from:
  - Low Power Abstraction: 1801
  - Reuse composition: 1685
  - Attribute harmonization: IP-XACT Xtension
  - IP Protection: 1735
    - Encryption
    - Rights management
    - Provenance tracking
- Bottom line: collect potential requirement sources
- Study group ⇔ Scope document

1801-2009

IEEE Standard for
Design and Verification of
Low Power Integrated Circuits

A port-supply relationship can be annotated in HDL using the following attributes:

Attribute name: **UPF_related_power_pin** or **UPF_related_ground_pin**.

Attribute value: **"***supply_port_name***"**, where *supply_port_name* is a string whose value is the simple name of a port on the same interface as the attributed port.

Attribute name: **UPF_related_bias_pin**.

Attribute value: **"***list_of_supply_port_names***"**, where *list_of_supply_port_names* is a string whose value is a space-separated list of one or more simple names of port(s) on the same interface as the attributed port.

SystemVerilog or Verilog-2005 example (power_pin):

```
(* UPF_related_power_pin = "my_Vdd" *) output my_Logic_Port;
```

VHDL example (power_pin):

```
attribute UPF_related_power_pin of my_Logic_Port : signal is
"my_Vdd";
```

```
(* UPF_related_power_pin = "my_Vdd" *) output my_Logic_Port;
```

SystemVerilog or Verilog-2005 example (bias_pin):

```
(* UPF_related_bias_pin = "my_VNWELL my_VPWELL" *) output
my_Logic_Port;
```

VHDL example (bias_pin):

```
attribute UPF_related_bias_pin of my_Logic_Port : signal
is "my_VNWELL my_VPWELL";
```

Isolation clamp value port properties can be annotated in HDL using the following attributes:

Attribute name: **UPF_clamp_value**

Attribute value: <**"0"** | **"1"** | **"Z"** | **"latch"** | **"any"** | **"***value***"**>

SystemVerilog or Verilog-2005 example:

```
(* UPF_clamp_value = "1" *) output my_Logic_Port;
```

4.6 Attributes and HDLs

Hardware description languages include a mechanism for specifying properties of objects. These properties are called *attributes*. Certain UPF properties can be annotated directly in HDL source descriptions using attributes. The semantic for properties specified using HDL attributes is the same as the corresponding behavior defined by the UPF command alternative (see Clause 6). Table 1 enumerates the HDL attributes defined for UPF-compliant implementations.

Table 1—Attribute and command correspondence[a]

HDL attribute name	Attribute value specification	Equivalent UPF command arguments	See
UPF_clamp_value	<**"0"** | **"1"** | **"Z"** | **"latch"** | **"any"** | "*value*">	**set_isolation -clamp_value** **set_port_attributes -clamp_value**	6.40 6.45
UPF_sink_off_clamp_value	<**"0"** | **"1"** | **"Z"** | **"latch"** | **"any"** | "*value*">	**set_isolation -sink_off_clamp_value** **set_port_attributes -sink_off_clamp_value**	6.40 6.45
UPF_source_off_clamp_value	<**"0"** | **"1"** | **"Z"** | **"latch"** | **"any"** | "*value*">	**set_isolation -source_off_clamp_value** **set_port_attributes -source_off_clamp_value**	6.40 6.45
UPF_pg_type	*pg_type_value* (see 4.3.4)	**set_port_attributes -pg_type**	6.45
UPF_related_power_pin	*port_name*	**set_pin_related_supply -related_power_pin** **set_port_attributes -related_power_port**	6.44 6.45
UPF_related_ground_pin	*port_name*	**set_pin_related_supply -related_ground_pin** **set_port_attributes -related_ground_port**	6.44 6.45
UPF_related_bias_pin	*port_name_list*	**set_port_attributes -related_bias_ports**	6.45
UPF_retention	<**"required"** | **"optional"**>	**set_retention_elements -retention required** **set_retention_elements -retention optional**	6.49
UPF_simstate_behavior	<**"ENABLE"** | **"DISABLE"**>	**set_simstate_behavior**	6.51
UPF_is_leaf_cell	<**"TRUE"** | **"FALSE"**>	**set_design_attributes -is_leaf_cell**	6.37

[a]Where two HDL attribute names are listed in the same table row, they are aliases for each other; where two UPF command arguments are listed in the same table row, they correspond to the same attribute.

Xact Extensions - rationale

- Propose a graded system of hosting schemas, documentation, generators, and examples of Vendor Extensions
- Vendor extensions may be placed in IP-XACT documents in several defined places. – The current schema provides an excellent framework.
- Sharing the definitions of extensions provides for greater opportunity for interworking
- Vendor extension are used but all sharing mechanisms are currently ad-hoc
- A lightweight mechanism for providing a "share point" will enable adoption and interoperability
- A Grading mechanism, or a set of "gates" will:
  - Set expectations of requirements, stability, and availability
  - Provide early access and later, stable availability
  - Provide a migration path to incorporation in the base schema, but only when necessary.
- Vetted by team appointed by Accellera Board

The SPIRIT Consortium

Potential Stages – under discussion

- Extension Proposal:
  - Posted for XXX meeting cycle - must move to alpha or time-out
  - Use issue tracker to share based on template
- Extension Alpha: Entry criteria – review group allows
  - Posted on web site in alpha space requires that reviewing member to see it?
  - Feature complete
- Extension Beta
  - At least 2 member company champions
- Extension Release:
  - Approval of release group
  - At least 2 users
  - Complete package
  - Schema - provides elements and parameters only for use at the vendor extension points of the base schema
  - Tool implementation? Is generator sufficient? (yes) Does the generator need to be Public?
  - Documentation:
    - (See PSS TGI not included)
    - Semantics of the schema
    - Use Model
    - Semantic checks or inconsistency
      - e.g. endianness - what elements and parameters of the base schema
  - Examples:
    - Include examples of all of the elements in the extension schema.
    - Include examples for all of the use models documented.
    - Include a generator
- Extension Integration into base schema:
  - Activity for SWG
  - Tool interoperability requirement here.
  - May occur at any time based on Schema group, may never be necessary

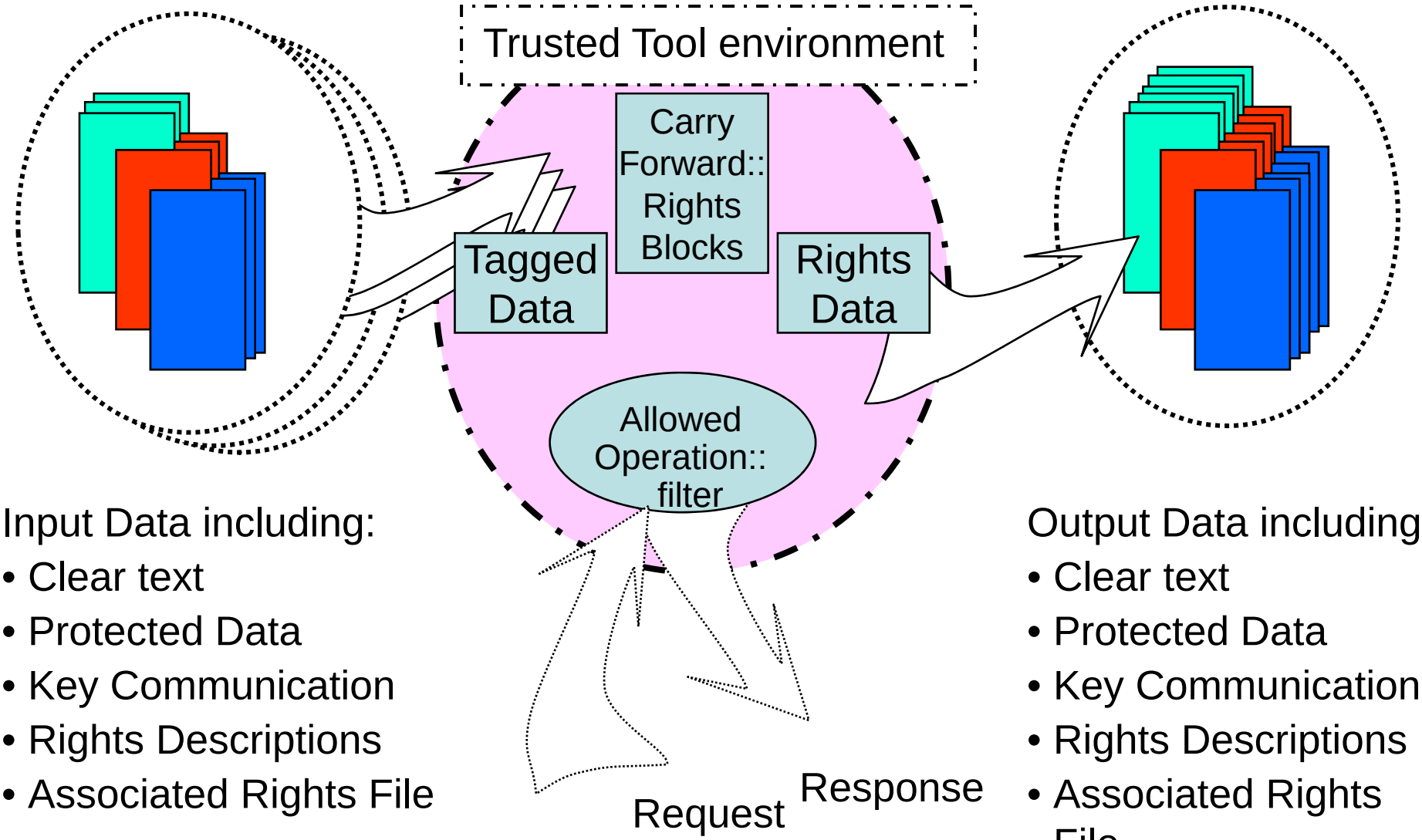
The SPIRIT Consortium

Potential XTs (are there more?)

- Bus Parameter Extensions to be used by OCP would come with:
  - Extension schema
  - Bus Def
  - AbstractionDef
  - Component
  - Design including component
- VSI Tags - provenance
- Power Intent
- Power characteristics

The SPIRIT Consortium

Trusted Tool Flow



Input Data including:

- Clear text
- Protected Data
- Key Communication
- Rights Descriptions
- Associated Rights File

Output Data including:

- Clear text
- Protected Data
- Key Communication
- Rights Descriptions
- Associated Rights File

Schematic of Protected IP Data Flow

First – Do no harm

- Gain experience – e.g. 1801
- Potential synergy from:
  - Low Power Abstraction: 1801
  - Reuse composition: 1685
  - Attribute harmonization: IP-XACT Xtension
  - IP Protection: 1735
    - Encryption
    - Rights management
    - Provenance tracking
- Bottom line: collect potential requirement sources
- Study group ⇔ Scope document