Path-based UPF Strategies:
Optimally Manage Power on your Designs

Progyna Khondkar, Siemens EDA,
Progyna_Khondkar@mentor.com

Abstract*-*UPF 3.0 and 3.1 LRM combined, introduces path-based semantics for isolation, level-shifter, and repeater strategies in conjunction with -sink and -diff_supply_only TRUE commands. This feature explicitly defines paths from source to sink domains for which any of the above strategies applies. Before path-based semantics were introduced, isolation, level-shifter, and repeater strategies utilized ad hoc methodologies, such as port splitting, which made power management very difficult. For example, when an isolation strategy is specified on ports and a port has a fanout going to multiple receiving logic supplies, the port-based semantics would place isolation cells on all paths. Contrast this with path-based semantics, which will place isolation cells in paths that go to specified sinks or receiving logic in the strategy. This paves the way to optimally manage power on any design. Evidently, path-based methodology adoption is not straight forward as it depends on the contents of strategies. For example, isolation strategies, -location fanout, self, and parent pose extra complexity to imply a strategy according to expectations. This paper addresses the complexity of adopting path-based strategies through numerous examples and real designs. This paper will also help the UPF user to smoothly transition from a port-based ad hoc to a path-based standard methodology and to understand how isolation, level-shifter, repeater strategies, and cells are inferred between source and sink power domains.

Key words: Port-based, Path-based, Port-splitting, Net-splitting, Redundant, Optimal, Area, Power

I. INTRODUCTION

UPF 3.0 and 3.1 LRM together define a path-based specification for UPF protection cells, like isolation (ISO), Level-Shifter (LS), and Repeater (REPTR) strategies defined in a user's design UPF file. The semantics combine the **-sink** <*PD/supply*> and **–diff_supply_only TRUE** explicitly, which defines the path from the source to the sink domain for which the strategy applies. It is important to remember that UPF 3.1 also introduces target insertion port semantics to facilitate net-splitting to honor this path-based specification.

UPF 3.1 emphasized on **-sink** <PD/Supply> along with **-diff_supply_only TRUE** to ensure minimal inferring of UPF protection cells on the specified sinks or receiving logic in the strategy. Precedence rules from UPF 3.1 also play vital roles to take all above into account. As noted earlier, port-based semantics implied port-splitting and when a port has multiple fanout going to multiple receiving logic supplies, then port-based semantics will split the port to place ISO cells that will apply to all paths. In contrast, path-based semantics will split nets relevant to the port and place ISO cells in paths that go to specified sinks in the strategy. For example, let us consider the following ISO strategies applied between PD1 (instance u1) and PD2 (instance u2), as shown in Figure 1.

```
set_isolation ISO1 -elements {u1}
set_isolation ISO2 -elements {u1} -sink {VDD2}
set_isolation ISO3 -elements {u1} -sink {VDDt}
```

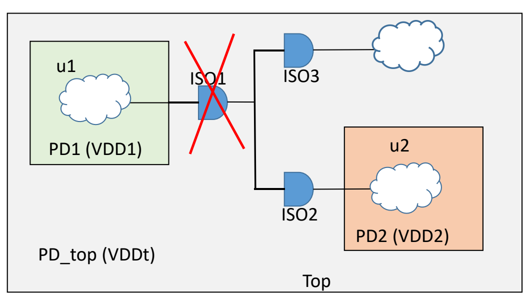

Figure 1. Example of path-based semantics

Here, based on UPF 3.1 precedence rules, it is very straightforward to comprehend path-based semantics from Figure 1, as shown in list 1 below.

List 1: Comprehending path-based semantics

- `ISO1 strategy has lowest precedence`
- `ISO2 and ISO3 have same precedence however they are covering different sinks (different paths so no conflict)`
- `The tool drops ISO1 due its lower precedence`
- `The tool honors ISO2 and ISO3.`
- `In each case, ISO cells are placed as close to the sink domain as possible.`
- `Net splitting facilitates this placement`

```
set_isolation strategy_name
[-source <source_domain_name | source_supply_ref >]
[-sink <sink_domain_name | sink_supply_ref >]
[-location <self | other | parent | fanout>]
[-clamp_value <0 | 1 | Z | latch | value | {<0 | 1 | Z | latch | value>*}>]
```

Figure 2. Generic ISO strategy

In general, ISO is required when source logic goes OFF and sink logic is in an ON state. Hence, if ISO is not inserted properly at the right place on the path, it can lead to a functional failure. On the other hand, if ISO is placed at a location where it is not required, it is redundant, wasting area and consuming unnecessary power. UPF commands specify where the ISO needs to be applied and the domain at which the cell needs to be inserted. It is important to note that path-based semantics are applied on HighConn of parents and covers all the paths. The HighConn context is shown in Figure 3.

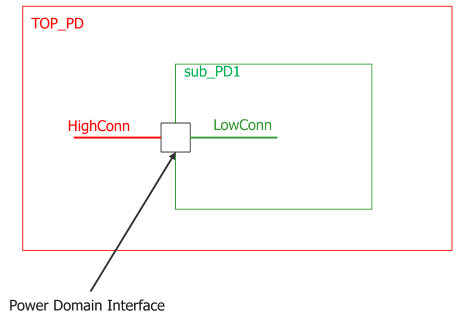

Figure 3. HighConn context between parent (TOP_PD) vs. child (sub_PD1)

On the contrary, port-based semantics can result in *collateral damage* when applying UPF protection cells on ports as shown in Figure 4.

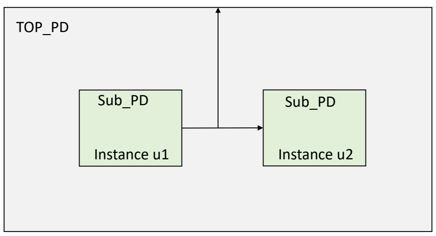

Figure 4. Explaining the *collateral damage* from port-based semantics

In Figure 4, there is TOP_PD with two sub blocks of instance u1 and u2 in the sub_PD power domain. The output from instance u1 goes to the top and to instance u2 (arrows indicate). In fact, there is only one power domain crossing: sub_PD of instance u1 to TOP_PD. However, in port-based semantics with an ISO scheme which includes "-domain PD_Sub -source PD_Sub.primary -sink PD_top.primary", port-splitting will occur and two ISO cells will be inferred or inserted (instead of one); one for each path. Although an intelligent structural checker may flag the redundant ISO on the sub_PD to sub_PD crossing, the sub_PD path is already considered to be *collateral damage* according to UPF LRM 3.1.

Since port-splitting is an error, we propose a warning mechanism in implementing a path-based algorithm discussed in section II. Such a warning alerts users by informing them that the isolation strategy which would have isolated the port on the path from sub_PD (source) to TOP_PD (sink) will result in splitting the port and (since port splitting is not allowed), a single isolation cell will be inserted that will isolate both the TOP_PD (sink) and PD_sub (sink).

Looking at the UPF LRMs IEEE 1801-2009 or 2.0, IEEE 1801-2013 or 2.1, IEEE 1801-2015 or 3.0, and IEEE 1801-2018 or 3.1, the semantical aspects that drive path-based insertion of UPF protection cells (e.g. ISO, LS, REPTR) are summarized below.

List 2: Summary of path-based semantical aspects

- Port splitting is still not allowed in UPF 3.1 standard
- If –location fanout is specified, the target insertion port is the port on the location domain boundary closest to the receiving logic. This is the location domain port that is driven by the port to which the strategy applies.
- If –location parent or –location self is specified, then target insertion port is the port specified in the strategy.
  NOTE: This clarification makes it possible for the isolation cell to be inserted along the path and not necessarily at the port
- If –sink and/or –diff_supply_only TRUE is specified (or if –diff_supply_only FALSE is not specified) in the strategy, then the inserted isolation cell shall only affect the receivers that are powered by a supply set that is different from the driving supply set
  NOTE: The wording above allows tools to do "net splitting" and to place the isolation cell close to the relevant port while still isolating only the path that needs the isolation.
- The 3.1 standard makes it clearer now that a single port with multiple fanout to different sinks may not be implementable depending on the location domain specified in the strategy, e.g., use of –location self and in some scenarios -location parent

II. COMPLEXITIES OF ADOPTION PATH-BASED STRATEGIES

The Path-Based Strategies methodology applies to isolation, level shifter, or repeater strategies on a per path basis in the design. The strategy that applies to a port on a path, specifies the power intent for that path exclusively. If no strategy applies to a port on a path, then the power intent (for isolation, level shifter, or repeater) will not infer any cell by the port for that path. This methodology allows insertion of isolation, level shifter, and repeater cells closest to, and connected to, the target insertion port within the extent of the location domain and follows the listed semantics shown here in List 3:

List 3: Path based semantics

- For an isolation, level shifter, and repeater cell - if users specify **-sink** in the strategy, then the inserted cell affects receivers that are powered by a supply set that matches the specified supply set.
- For an isolation cell - if users specify **-diff_supply_only TRUE** in the strategy, then the inserted cell affects receivers that are powered by a supply set that does not match the driving supply set.
- **NOTE**: In the case of multiple paths to different receiving supplies, care should be taken when specifying the location domain for isolation, level shifter, or repeater cell inferred for one path may affect another path. It shall be an error if the isolation power intent cannot be implemented without duplicating ports. Although the standard specifies the outline of the rules but it is the algorithm that is implemented in a tool to interpret and infer strategies.

The proposed algorithm in this paper provides a mechanism for a simulation tool to display the above "NOTE" as a warning message –

- When the methodology cannot implement the isolation, level shifter, or repeater power intent without duplicating ports, and
- Then inserts isolation, level shifter, or repeater cell in the specified location (self, parent, or fanout) and in the path to the target receivers.

The methodology considers following mechanism to target a strategy-based cell insertion as explained below.

- For isolation and level shifter cells, the target insertion point depends on whether the user specified the **-location** fanout option or not:
    - When user specifies **-location fanout,** the target insertion port is the port on the location domain boundary that is closest to the receiving logic. If the receiving logic is in a macro cell instance, the target insertion port is the input port of that macro cell instance, on the lower boundary of the location domain.
    - Otherwise, the target insertion port is the location domain port that is driven by the port to which the strategy applies.
    - When user does not specify **-location fanout**, the target insertion port is the port of the location domain that is (for the -self domain) or corresponds to (for the parent or child domain) the port to which the strategy applies.
- For a repeater cell, the target insertion port is the port to which the strategy applies.

So, it's evident — that the path-based strategy and cell inferring are the defacto standards and allow users to manage power specification in optimal way.

Let us explain the complex algorithm in a very simple way through an example so users can understand inferring strategies during simulation, so that expectations from spec and the outcome of simulation match. Consider the following design example for different ISO strategies:

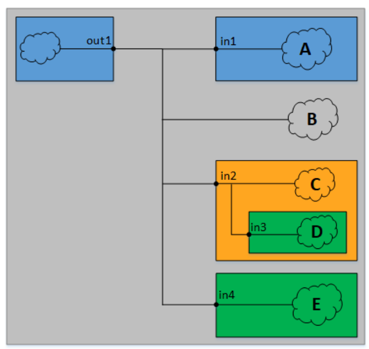

Figure 5. Generic view of the example

*A. Case 1: When users specifying ISO strategy: **set_isolation** iso1 **-domain** Blue **-sink** Green*

The outcome algorithm will process the strategy as follows.

- For **-location self**: The algorithm cannot implement the isolation strategy that affects only the target receiver (green domain). A tool can give a warning message to take special care when specifying location on a case of multiple paths to different receiving supplies and inserts the isolation cell in the **-self** domain and in the path to the green domain. This is shown in Figure 6.

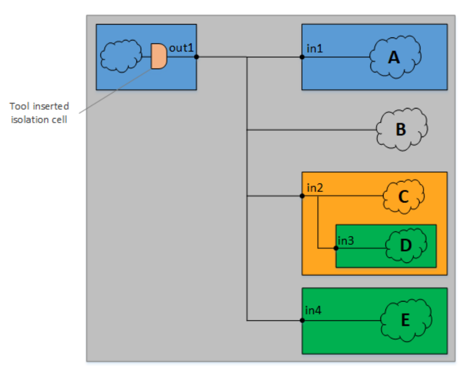

Figure 6. Outcome of algorithm for isolation cell inferring for **-location self** (Case 1)

NOTE: ***Both port- and path-based semantics will place ISO cell in the same location and a warning will show up.***

- For **-location parent**: The algorithm cannot implement the isolation strategy that affects only the target receiver (green domain). A tool can give a warning message (as shown above) and inserts the isolation cell in the parent domain and in the path to the green domain. This is shown in Figure 7.

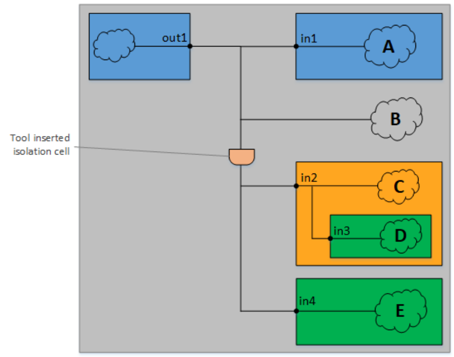

Figure 7. Outcome of algorithm for isolation cell inferring for **-location parent** (Case 1)

NOTE: *For port-based semantics, ISO cell will be placed outside of out1 and a warning will show up.*

- For **-location fanout**: The algorithm can implement the isolation strategy that affects only the target receivers (-sink). This is shown in Figure 8.

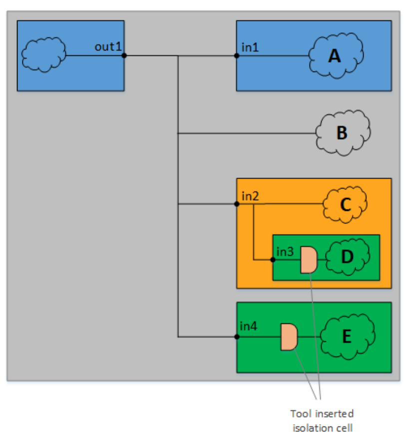

Figure 8. Outcome of algorithm for isolation cell inferring for **-location fanout** (Case 1)

NOTE: *Both port- and path-based semantics will place ISO cell in the same location and a warning will show up.*

B. Case 2: When users specifying ISO strategy: ***set_isolation*** *iso2* ***-domain*** *Blue* ***-sink*** *Orange*
The outcome algorithm will process the strategy as follows.

- For **-location self**: The algorithm cannot implement the isolation strategy that affects only the target receiver (orange domain). A tool can give a warning message (as shown above) and inserts the isolation cell in the self-domain and in the path to the orange domain. This is shown in Figure 9.

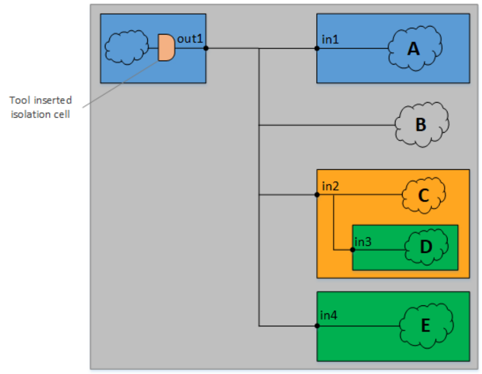

Figure 9. Outcome of algorithm for isolation cell inferring for **-location self** (Case 2)

NOTE: *Both port- and path-based semantics will place ISO cell in the same location and a warning will show up.*

- For **-location parent**: The algorithm cannot implement the isolation strategy that affects only the target receiver (orange domain). A tool can give a warning message (as shown above) and inserts the isolation cell in the parent domain and in the path to the orange domain. This is shown in Figure 10.

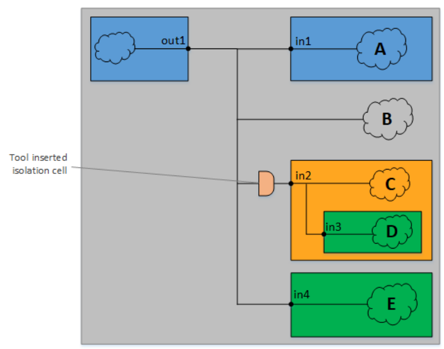

Figure 10. Outcome of algorithm for isolation cell inferring for **-location parent** (Case 2)

NOTE: *For port-based semantics ISO cell will be placed outside of out1 and a warning will show up.*

- For **-location fanout**: The algorithm cannot implement the isolation strategy that affects only the target receivers (-sink). This is shown in Figure 11.

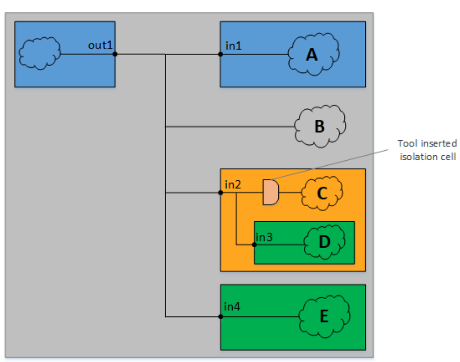

Figure 11. Outcome of algorithm for isolation cell inferring for **-location fanout** (Case 2)

NOTE: *For port-based semantics, ISO cell will be placed on path of C & D inside of orange sink near in2 and a warning will show up.*

C. Case 3: When users specify ISO Strategy: ***set_isolation*** *iso3* ***-domain*** *Blue* ***-diff_supply_only*** *TRUE*
The outcome algorithm will process the strategy as follows.

- For **-location self**: The algorithm cannot implement the isolation strategy that affects only the target receivers (gray, green, and orange domains). A tool can give a warning message (as shown above) and inserts the isolation cell in the self-domain and in the path to the gray, green, and orange domains. This is shown in Figure 12.

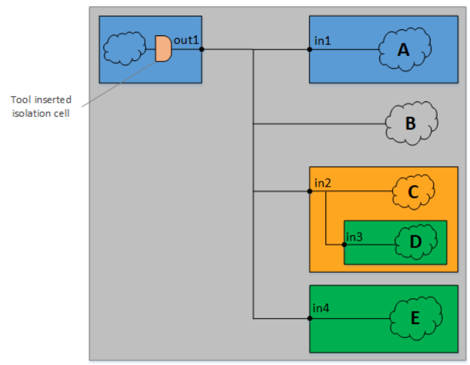

Figure 12. Outcome of algorithm for isolation cell inferring for **-location self** (Case 3)

NOTE: *Both port- and path-based semantics will place ISO cell in the same location and a warning will show up.*

- For **-location parent**: The algorithm implements the isolation strategy that affects only the target receivers (-diff_supply_only TRUE). This is shown in Figure 13.

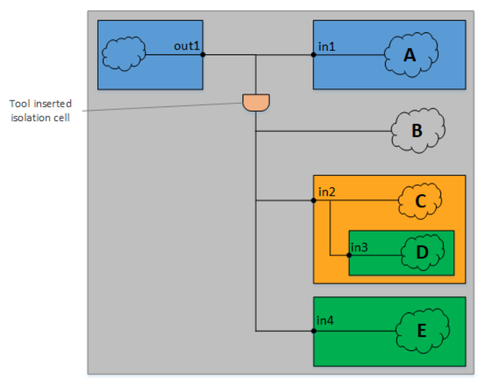

Figure 13. Outcome of algorithm for isolation cell inferring for **-location parent** (Case 3)

NOTE: *For port-based semantics ISO cell will be placed outside of out1 and warning will show up.*

- For **-location fanout**: The algorithm implements the isolation strategy that affects only the target receivers (-diff_supply_only TRUE). This is shown in Figure 14.

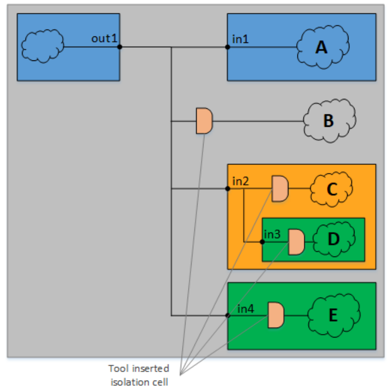

Figure 14. Outcome of algorithm for isolation cell inferring for **-location fanout** (Case 3)

NOTE: *For port-based semantics, ISO cell will be placed outside of out1, inside of orange sink near in2, before D near in3, and a warning will show up.*

It is evident from the above three case studies that, on a real design, **-location self** shows similar placements of ISO cells. However, for **-location parent** and **-location fanout**, the port- and path-based semantics differ significantly. This difference allows users to comprehend optimal and accurate designation of ISO cells from a very early stage of the design phase, at the RTL, which is bound to comply on down the implementation phase of UPF protection cells, whether it is during synthesis or the place&route stages.

III. Optimally Manage Power on your Designs

The previous section addresses the complexity of adopting path-based strategies through numerous examples. This section further demonstrates how path-based algorithms comply with UPF 3.1 standards and their implication on real design examples, so that UPF users can smoothly transition from port-based ad hoc to a path-based standard methodology and understand how isolation, level-shifter, and repeater strategies and cells are inferred between source and sink power domains.

A. Why ISO not before instance uL1C and uL1D but only before instance uL1B?

In this real design example, we consider a case study with **-location parent** as shown in Figure 15 below. The simple construction of the design with power domain distribution is shown also on the right hand side of Figure 15.

<table>
<tr>
<td>

```
set_isolation ISO_PDA \
 -domain PDA \
 -isolation_supply_set ss_PD0 \
 -clamp_value 0 \
 -applies_to outputs \
 -diff_supply_only TRUE \
 -isolation_signal {uPMU/ISOEN} \
 -isolation_sense low \
 -location parent
```

</td>
<td>

```
Power Domain: /tb/uTOP/PDA, File: design.upf(5).
   Creation Scope: /tb/uTOP
   Extents:
    1. Instance : /tb/uTOP/uL1C
    2. Instance : /tb/uTOP/uL1B
Primary Supplies: Supply Set handles: primary: /tb/uTOP/ss_PD1

Power Domain: /tb/uTOP/PD0, File: design.upf(4).
    Creation Scope: /tb/uTOP
    Extents:
    1. Instance : /tb/uTOP
    2. Instance : /tb/uTOP/uL1B, Lower Boundary
    3. Instance : /tb/uTOP/uL1C, Lower Boundary
 Primary Supplies: Supply Set handles: primary: /tb/uTOP/ss_PD0
NOTE: Sub hierarchical modules are
/tb/uTOP/uL1C (PDA)
tb/uTOP/uL1C/uL1Ca/uL1Cb (PDA)
tb/uTOP/uL1D (PD0)
tb/uTOP/uL1E (PD0)
tb/uTOP/uPMU (PD0)
```

</td>
</tr>
</table>

Figure 15 Design UPF and part of power domains scopes and extents

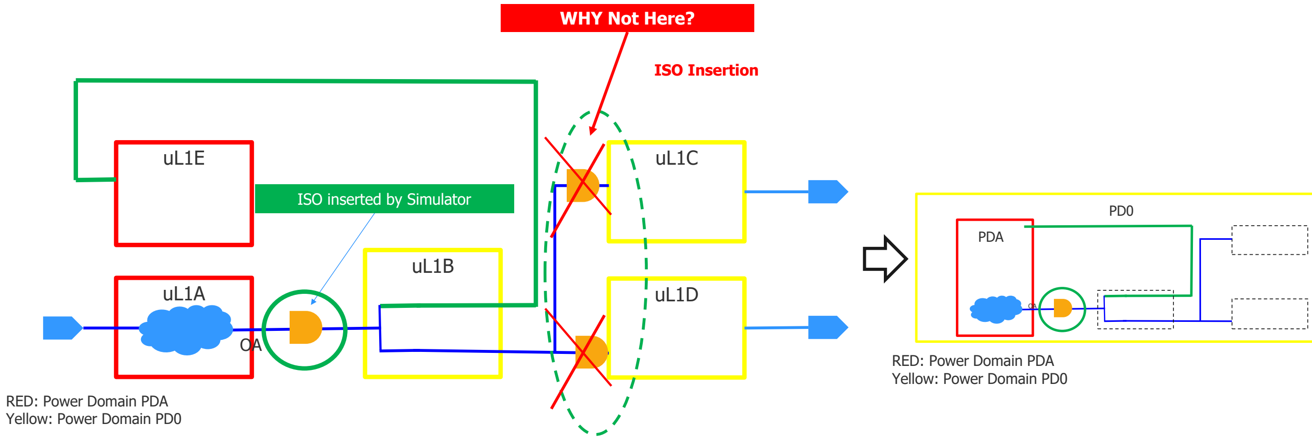

Figure 16. Structural representation of power domains scopes and extents of the design from Figure 15

Here red instances are within power domain PDA and yellow are within PD0. When **set_isolation –location parent** is specified on a port, the RTL simulator does not insert isolation cell insertion anywhere in parent instance. Based on the discussed algorithm, the tool only considers ports at **<u>HighConn</u>** in **parent** instance for isolation cell insertion. Obviously, the implemented algorithm does not consider adding ISOs in the **green dotted circle**. In this testcase, a tool used for the experiments gives following warning because port-splitting is a violation.

```
** Warning: test.upf(41): (vopt-9927) Isolation strategy 'ISO_PDA' specified on port '/TOP/uL1A/OA' can't
be applied without splitting the port '/TOP/uL1A/OA'. To avoid port splitting,  cell will be inserted
for the paths that go to target receiver supply.
```

B. What happens to ISO for glue logic?

In this real design example, we consider a case study with **-location fanout** — actually a real **"**heterogeneous fanout**"** scenario. The UPF and design blocks (in modules and glue logic) and relevant power domains are shown in Figure 17. One of the big concerns here are what will happen to ISO placement if there is glue logic in the design on the fanout path.

```
set_isolation ISO_PD2_out_0 \
 -domain PD2 \
 -isolation_supply_set ss_PD1 \
 -clamp_value 0 \
 -applies_to outputs \
 -diff_supply_only TRUE

set_isolation_control ISO_PD2_out_0 \
 -domain PD2 \
 -isolation_signal {PMU/ISO} \
 -isolation_sense low \
 -location fanout
```

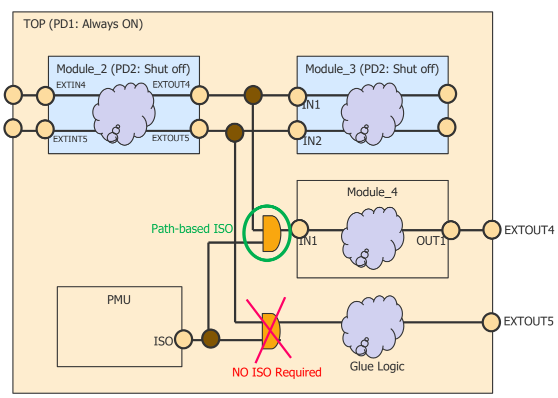

Figure 17. The UPF, design blocks (in modules and glue logic), and relevant power domains

For glue logic we implicate enable/disable options to allow/disallow ISO before glue-logic. From Figure 17, its evident that the Module_3 output ports are not directly connected to glue-logic, but they are connected to top-level module output ports. Also note that the **Module_3** output ports are not path based, there is only a path going out from the Module_3 output ports. Hence, based on user discretion, "tool option to disable ISO before glue-logic" has no effect on these ports. Such a disabling ISO option will take effect only when:

- Port has multiple paths and corresponding ISO strategy is not applicable to one of these paths.
- Port is directly connected to glue-logic.

Hence, ISO cell will continue to be inserted as shown in green circle because for Module_3, output ports driving supply is ss_PD2 and receiving supply is ss_PD1. However, disabled ISO before glue-logic will not insert isolation cell in glue-logic path of a port — if the port has multiple paths and corresponding isolation strategy is not applicable to at least one of the paths. This is shown by red crossed line and texts in Figure 17.

IV. CONCLUDING REMARKS

As noted earlier, UPF protection cells like ISOs are required when source logic goes OFF and sink logic is in an ON state. Hence, if ISO is not inserted properly at the right path, then it can lead to functional failure. On the other hand, if ISO is placed at a location where it is not required, it is redundant, wastes area, and consumes unnecessary power. It is evident now that path-based semantics allow users to manage power on any design in a very efficient and optimal way, compared to the legacy port-based semantics. With real design examples, we have presented the complete path-based algorithm implemented in simulation tools through case studies. The simulation tool implicated with the proposed algorithm applies isolation, level shifter, and repeater strategies on a per path basis on any design. The strategy that applies to a port on a path specifies the power intent for that path exclusively. With the semantic incorporated, the tool inserts an isolation, level shifter, and repeater cell closest to, and connected to, the target insertion port within the extent of the location domain and follows the listed semantics:

- For an isolation, level shifter, and repeater cell — if a user specifies -sink in the strategy, then the inserted cell affects receivers that are powered by a supply set that matches the specified supply set.

- For an isolation cell — if a user specifies **-diff_supply_only TRUE** in the strategy, then the inserted cell affects receivers that are powered by a supply set that does not match the driving supply set.

In the case of multiple paths to different receiving supplies, care should be taken when specifying the location domain as an isolation, level shifter, or repeater cell inferred, since one path may affect another path. A power aware or low power simulation tool should display a warning message (as shown in section II) if it cannot implement the isolation, level shifter, or repeater power intent without duplicating ports, and inserts the isolation, level shifter, or repeater cell in the specified location (self, parent, or fanout) and in the path to the target receivers. So **-sink** and **-source** directly manipulate **-location**, and obviously from UPF 3.1, specifies the path on which isolation, level shifter, and repeater must be applied. In the case that the semantics and implementation algorithms are not incorporated, then tools should error out.

V. REFERENCES

[1] IEEE Std. 1801™-2009 "IEEE Standard for Design and Verification of Low-Power Integrated Circuits", 6 March 2013.
[2] IIEEE Std. 1801™-2013 "IEEE Standard for Design and Verification of Low-Power Integrated Circuits", IEEE Computer Society, 29 May 2013.
[3] IEEE Std 1801™-2015 "IEEE Standard for Design and Verification of Low Power, Energy Aware Electronic Systems IEEE Computer Society, 05 Dec 2015.
[4] IEEE Std 1801™-2018 "IEEE Standard for Design and Verification of Low Power, Energy Aware Electroinc Systems" IEEE Computer Society, 29 March 2019