PA-APIs: Looking beyond power intent specification formats

Amit Srivastava and Awashesh Kumar
Mentor Graphics Corp. 8005 SW Boeckman Rd. Wilsonville, OR 97070

Abstract

Power Management affects design functionality, hence both tools and users need to be aware of this information. The traditional approaches to query information don't work for power aware designs as information is captured in different formats, UPF, HDL and Liberty. The paper provides details of an abstract data model for Power Aware design and the interface to query the information. This can be used to provide standard access to power management information.

Access to Power Aware Information

Access to power aware information is not just significant for tool developers but also to design and verification engineers. Some typical requirements are:

- Develop utilities for design intent exploration.
  - Generate a customized report of power architecture information for specific exploration. E.g. List of power management cells inserted in the design.
- Develop utilities that help create additional UPF definitions.
  - Understand the power management of an IP and then accordingly create the power intent of the SoC depending upon characteristics of the IP.
- Develop custom checking utilities that can be incorporated into a quality checking flow.
  - Check that an isolation clamp value matches the reset value.
- Create an environment to automatically generate coverage monitors and assertions for power management.

Obstacles to accessing PA Information

There are various challenges to accessing information related to power management.

- Power management information is specified separately from HDL in UPF and Liberty files.
- UPF relies more on tool automation to simplify specification and hence its difficult to inspect UPF files for extracting information.
- UPF provides some Tcl Query commands which are incomplete and inconsistent to be used for accessing information.

Modeling the Power Aware Information

A model that captures the information related to power management for a given design and provides simplified interface to access the information.

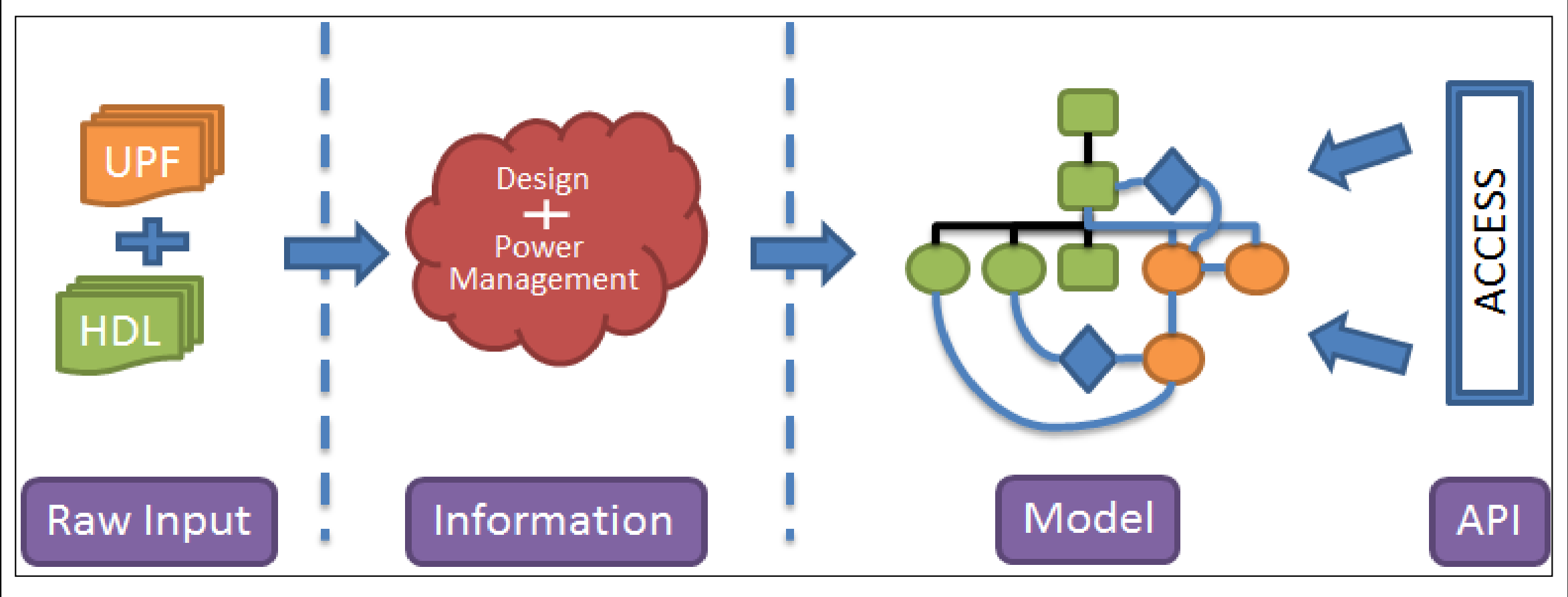

Figure 1: Information Model Flow

PA Information Model

Objects	Attributes
Primary holders of information	Pieces of information present on objects
Accessed by PA Handle	Accessed by Attribute IDs
Classified into three broad groups • UPF Objects • HDL Objects • Relationship Objects	Classified into two groups • Basic Types • String, Integer, Boolean, Float, Enumerated • Complex Types • Handle, List of Handles

PA Handle: A reference to an object in PA Information Model

UPF Objects

- Represents objects created by UPF
  - e.g. Power Domains, Supply Sets, Power States, etc.
- Captures information after application of power intent
- Can represent objects from different UPF versions and other power formats

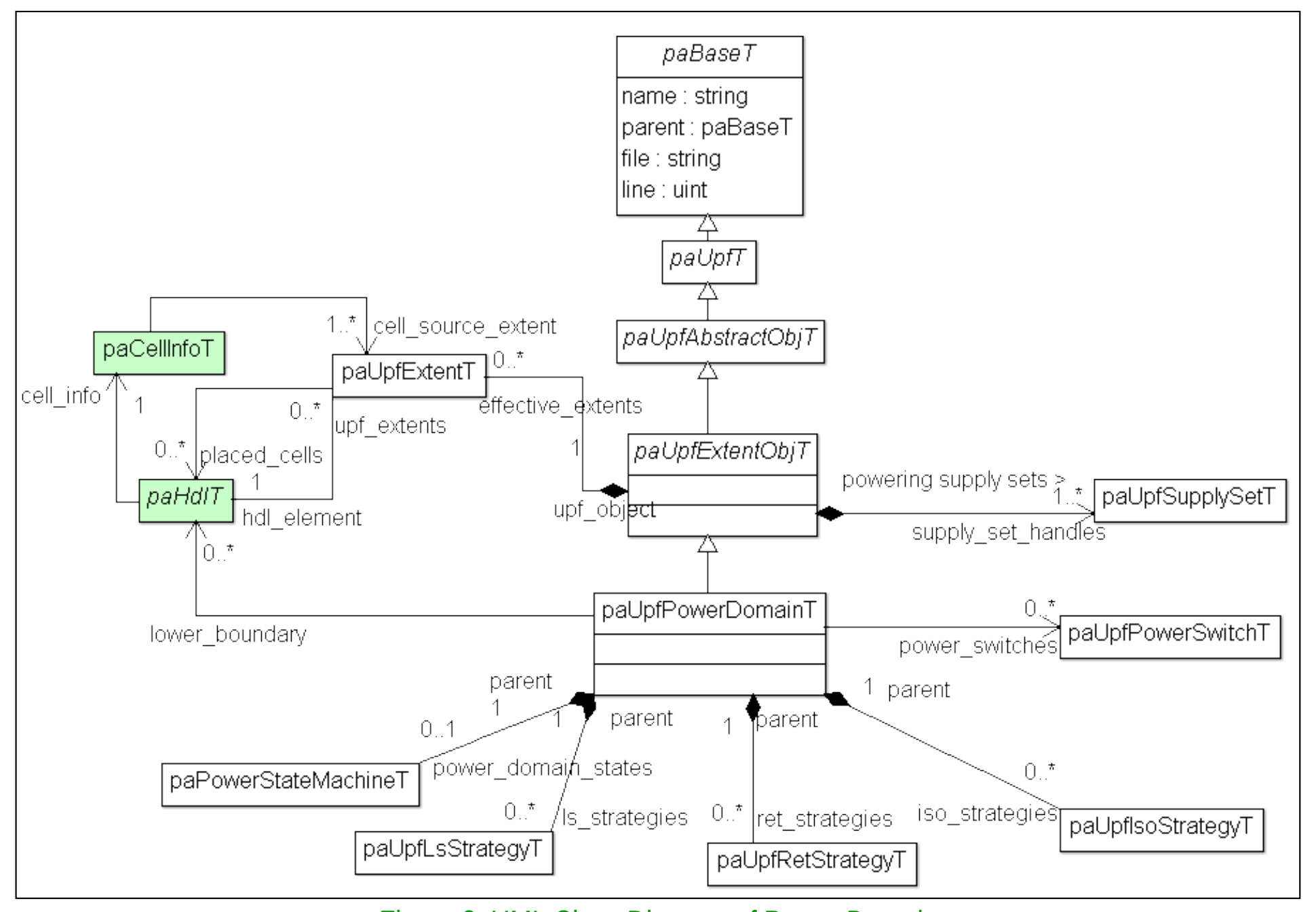

Figure 2: UML Class Diagram of Power Domain

HDL Objects

- Represent objects from HDL design hierarchy
- Object contains abstracted HDL information common across all HDL languages
  - e.g. hierarchical structure, name, size of ports/nets
- Only subset of HDL information is captured
  - necessary to represent power management architecture
  - e.g. extent, control signals, creation scopes, -instance, etc.
- Additional HDL information can be accessed by getting full-hierarchical path of HDL object from PA Handle and then querying from other respective HDL Information Models.

Relationship Objects

- Represent some relationships between other objects
- Captures meta information which is not present in user design
  - e.g. Expressions, extents, cell information etc.

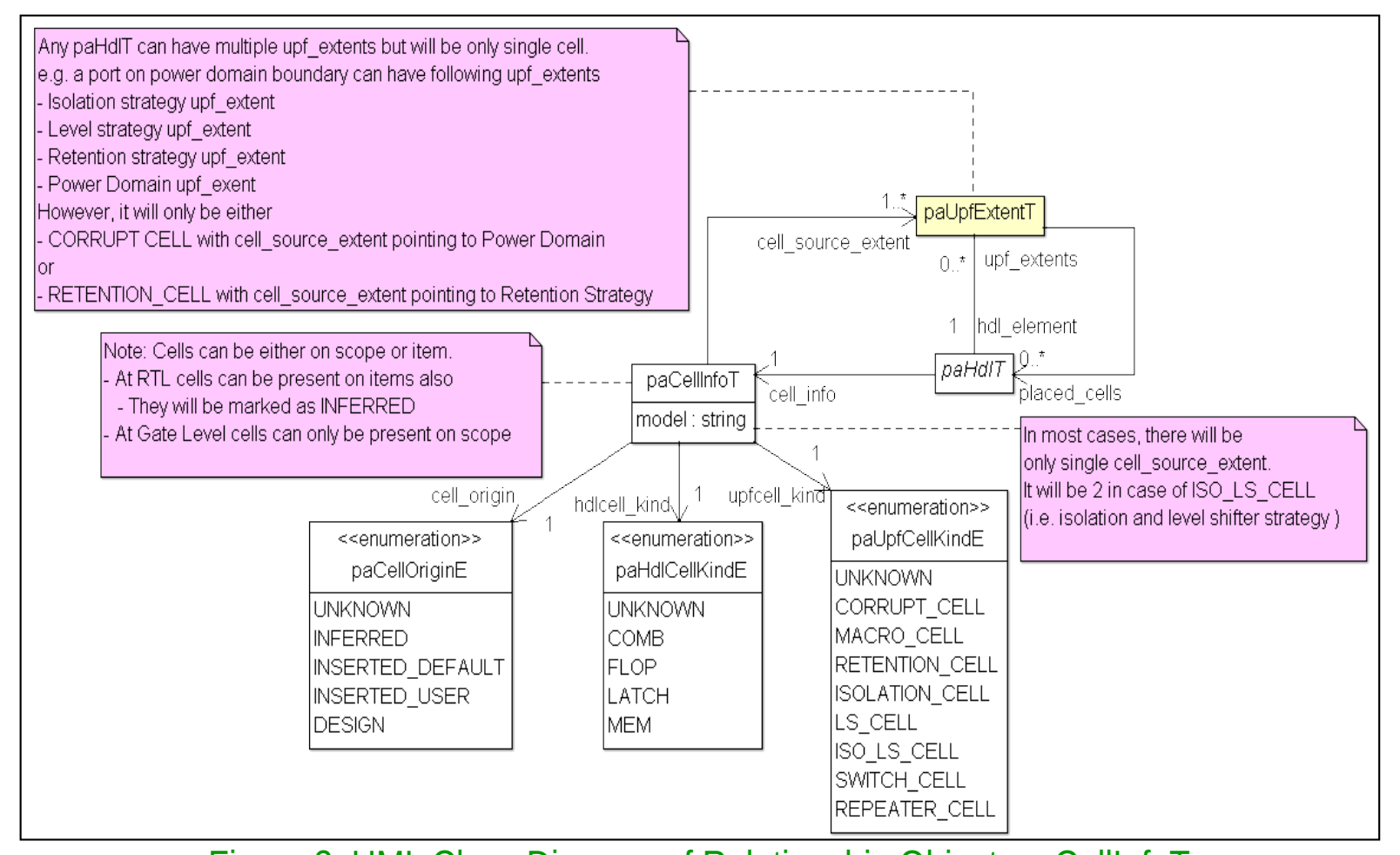

Figure 3: UML Class Diagram of Relationship Object: paCellInfoT

Power Aware Application Programmable Interface (PA-API)

PA-API is the interface to access information from PA-Information Model. It can be implemented in different languages depending upon requirements.

- C Interface
  - Can be used to construct more sophisticated applications
- Tcl Interface
  - Allow querying power management information
- HDL Interface
  - Allow construction of high level testbenches

The PA-API is mainly designed to provide read only access to PA Information. However, there is some requirement to provide API to modify dynamic attributes during simulation in order to construct sophisticated testbenches at higher level of abstraction. This feature is currently under exploration.

C-API

C PA-API		
API Name	**Return Type**	**Description**
pa_GetHandle	paHandleT	Returns a handle of an object corresponding to the given attribute
pa_GetHandleByName	paHandleT	Returns a handle of another object matching a given name
pa_GetInt	int	returns the integer value of the given attribute
pa_GetStr	char*	returns the string value of the given attribute
pa_GetSeqIterator	paIteratorT	returns the iterator to list of objects for the given attribute
pa_GetNext	paHandleT	returns the next handle in a given iterator
pa_GetHandleType	int	returns the object type of a given handle
pa_IsInClass	int	Returns 1 if object belongs to specified class else 0
pa_GetHierPathname	char*	returns the hierarchical path of a given object
pa_IsSameHandle	int	Returns 1 if handles are same else 0

Tcl-API

Tcl PA-API	
API Name	**Description**
pa_get_object_handle_by_name	Search object handle by name
pa_get_attribute	Get value of given attribute present on the handle
pa_get_hierpath	Return the full hierarchical pathname for specified handle
pa_is_in_class	Check if handle belongs to a specified class
pa_is_same_handle	Check if two handles are same

Example Design

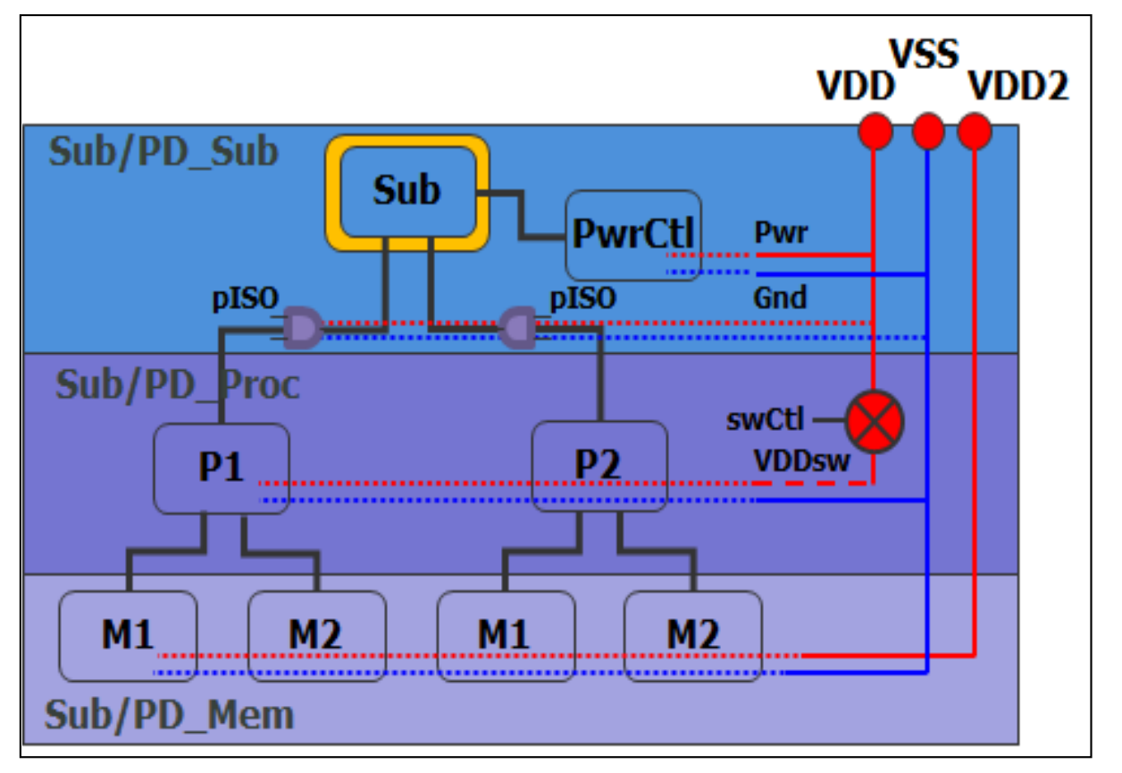

Figure 4: Block Diagram of Example Design

UPF

```
set_scope Sub
create_power_domain PD_Sub \
-include_scope
create_power_domain PD_Proc \
-elements {P1 P2}
create_power_domain PD_Mem \
-elements {P1/M1 P1/M2 P2/M1
P2/M2}
set_isolation ISOproc \
-domain PD_Proc \
-applies_to outputs \
-clamp_value 0 \
-location parent \
-isolation_power_net Pwr \
-isolation_ground_net Gnd
-isolation_signal pISO \
-isolation_sense high
...
```

UML Object Diagram of Information Model

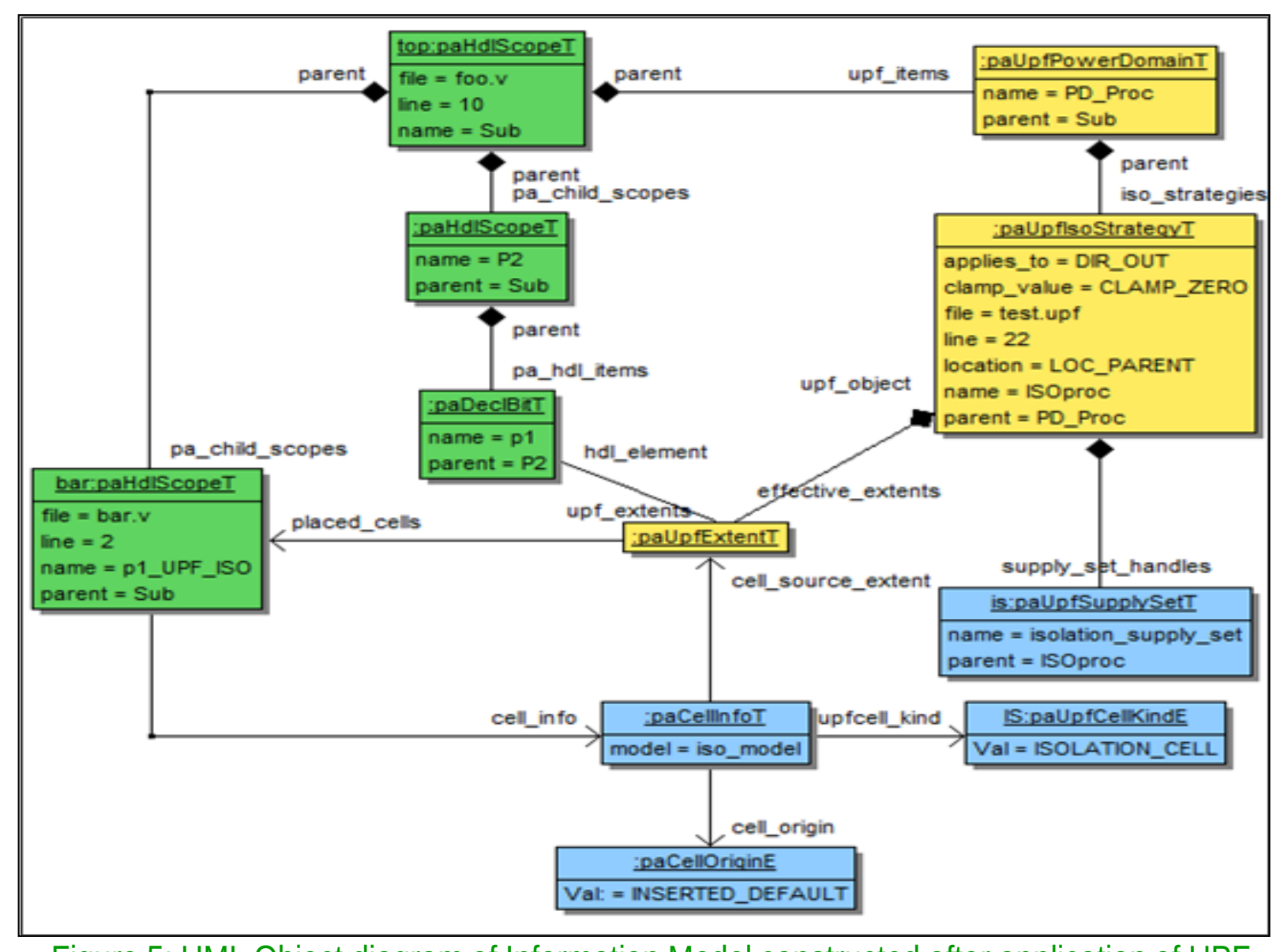

Figure 5: UML Object diagram of Information Model constructed after application of UPF

C-Example: Print Isolation Information

```
pd = pa_GetHandleByName( NULL,“Sub/PD_Proc” );
iso_iter = pa_GetSeqIterator(pd, ISO_STRATEGIES);
pa_IterForeach( iso_iter, iso ) {
  pa_GetStr( iso, NAME );
 //=> “ISOproc”
  pa_GetInt(iso, CLAMP_VALUE, &clamp);
 //=> 0
  ctrl = pa_GetHandle(iso, ISOLATION_CONTROL);
  ctrl_sig = pa_GetHandle(ctrl, CONTROL_SIGNAL);
 //=> pISO
  pa_GetInt(ctrl, SIGNAL_SENSITIVITY, &ctrl_sense);
 //=> SENSE_HIGH
}
port_iter = pa_GetSeqIterator(iso, EFFECTIVE_EXTENTS);
pa_IterForeach(port_iter, port_ex) {
  port = pa_GetHandle(port_ex, HDL_ELEMENT);
  path = pa_GetFullPathname(port, NULL);
//=> /Sub/P2/p1
  cells_iter = pa_GetSeqIterator(port_ex, PLACED_CELLS);
  pa_IterForeach(cells_iter, cell) {
    path = pa_GetFullPathname(cell, NULL);
//=> /Sub/p1_UPF_ISO
  }
}
```

Tcl-Example: Return list of all isolation cells in a given scope

```
proc get_iso_cells {scope}{
  set cells {}
# Check if Valid Handle
  if{[pa_is_in_class -handle scope -class_id CLASS_HDL_SCOPE] == 0} {
    return $cells
  }
# Get Child Scopes
  set children [pa_get_attribute -handle $scope -attribute
CHILD_SCOPES ]
# Iterate over children and check if isolation
  foreach ch $children {
    set cell [pa_get_attribute -handle $ch -attribute CELL_INFO]
    if { $cell ne “” } {
      set cell_type [pa_get_attribute -handle $cell –attribute
UPFCELL_KIND]
      if { $cell_type == ISOLATION_CELL } {
        lappend cells [pa_get_hierpath $cell]
      }
    } else {
      concat $cells [get_iso_cells $cell]
    }
  }
  return $cells
}
puts [get_iso_cell [pa_get_object_handle_by_name –name “/Sub”]]
# => { /Sub/p1_UPF_ISO }
```

Conclusion

The PA-IM along with PA-APIs provides a well-defined structure and simplified access to power management information. This can be used across tools and design flows. It captures the result of the application of power intent and stores the abstract representation of HDL. Hence, it can represent the PA information at different levels of design abstraction – RTL or Gate Level and also from different sources (UPF, HDL, Liberty). The following work has been donated to IEEE P1801 UPF WG and the work is on to include this in UPF Standard.