Learn the OpenAccess API

Using Python

Initial Contribution By

James Masters
Intel - 2013
Updates & Additions
by Silicon Integration Initiative - 2020

Section 6 - The Design

- oaDesign
- oaBlock
- oaModule
- oaCell
- oaCellView

Overview

- Three classes encapsulate OA design data:
  - oaDesign: top-level object that deals with basic operations
    - Library, cell, and view names
    - Saving/closing operations
    - Wraps exactly one oaBlock (for our purposes)
  - oaBlock:
    - Contains actual design related objects (e.g. shapes)
  - oaModule
    - Represents logical data
    - Can have hierarchy (Module in a module, in a module, …)
- Other classes encapsulate lib, cell, view, and cell view data as stored on disk (Data Management – “DM”)
  - oaLib, oaCell, oaView, oaCellView
  - Interact directly with files stored on disk
  - Remove cell or cell view from library

Creating oaDesign

- All access to an oaDesign is done through an access mode character in the “open” function
  - Common modes:
    - Read “r”
    - Write “w” (overwrites all existing data and starts fresh)
    - Append “a” (edits existing design – will create if it doesn’t exist)
  - Primary access modes are read (“r”) and append (“a”)
- The oaViewType must be known for new designs
  - oaViewType enumeration wrapper is used to retrieve a view type
  - Not required for opening existing designs
  - See oaReservedViewType for a list of reserved types

```
# open existing schematic design
mydes = oa.oaDesign.open("mylib", "top", "schematic", "r")

# create new layout design
ml = oa.oaViewType.get("maskLayout")
test_des = oa.oaDesign.open("mylib", "test", "layout", ml, "a")
```

Accessing Lib, Cell, and View Names

- The library, cell, and view names can be accessed from the oaDesign object:

```
print "Lib: %s" % test_des.getLibName()
print "Cell: %s" % test_des.getCellName()
print "View: %s" % test_des.getViewName()
```

Accessing oaBlock from oaDesign

- An oaBlock is accessed through an oaDesign
  - Create new oaBlock (only need to create once whenever a new oaDesign is created)

```
test_blk = oa.oaBlock.create(test_des)
```

  - Access top oaBlock from an existing oaDesign

```
test_blk = test_des.getTopBlock()
```

Accessing the Module from oaDesign

- An oaModule is accessed through an oaDesign

```
test_mod = oa.oaModule.create(design)
```

- An oaModule can be accessed through the design

```
test_mod = design.getTopModule()
```

- An oaModule can be created multiple times within a design, or within another module
  - The database is updated upon a module save() unlike the oaBlock which is constantly updating the database with changes

Lab 6.1 - Create an oaDesign and an oaBlock

- Goal - Become familiar with oaDesign and oaBlock
- Write a script to:
  1. Create a design called “mycell” within library mylib
  2. Create a new oaBlock
  3. save the design
  4. Create a design called “bad” containing an oaBlock
  5. Discard (purge) the design
     - Look at the contents of the mylib library directory.
     - Both mycell and bad exist though bad has a layout file that is zero bytes
     - Note: you did not save the design after the purge yet the layout file is empty

compare your script to labs/6.1/createDesign.py

oaCell, oaView, and oaCellView DM Objects

- File system operations are done using Data Management (DM) containers – oaLib, oaCell, oaView, oaCellView
- Can loop through cells, views, or cell views within a library

```
for cv in lib.getCellViews():
    cell = cv.getCell()
    view = cv.getView()
    ...
```

- Primary file can be retreived using getPrimary() to get the file name, size, location, etc:

```
dmfile =  cv.getPrimary()
dmfile_path  = dmfile.getPath()
```

- A cell can be removed from disk using oaCell.destroy()

```
cell.destroy()
```

oaLib Access

- Library “write” access is needed before any changes are allowed to be made to a library (e.g. removing a cell, view, etc.)
- Obtaining access to the lib object:

```
lib.getAccess('write')
```

Lab 6.2: Navigate DM Objects

- Goals:
  - Understand the need for “write” library access
  - Become familiar in working with DM objects
  - Understand how to remove a cell view from a library
- Delete empty cell views in a library
  - Open “mylib” library
  - Loop through each cell view in the library
  - Get the primary file of the cell view
  - If the primary file of the cell view is empty, then remove it
  - Extra credit – detect when no more cell views exist for a given cell and remove that cell as well

compare your script to labs/6.2/deleteDesign.py

Section 6 Summary

- Become familiar with dealing with DM objects
  - Create a Design
  - Creating an oaBlock and an oaModule
  - Purge and delete a design

Silicon Integration Initiative

www.si2.org

For details contact Marshall Tiner

Director of Production Standards

mtiner@si2.org