Learn the OpenAccess API
Using Python

©Silicon Integration Initiative
Initial Contribution By

James Masters
Intel - 2013
Updates & Additions
by Silicon Integration Initiative - 2020
Section 8: `oaPRBoundary`, `oaLayerBlockage`, `oaTrackPattern`

- OA has special objects for certain design concepts:
  - Cell boundary (`oaPRBoundary`)
  - Grids (`oaTrackPattern`)
  - Routing blockages/keepouts/obstructions (`oaLayerBlockage`)
Design content is typically encapsulated in a cell boundary
- OA has an `oaPRBoundary` object
- Only one `oaPRBoundary` object is allowed per block
- The `oaPRBoundary` object accepts a list of points making up a polygon (although boundaries are often just a rectangle)

# Note – for brevity and because the built-in UU to DBU
# conversion functions are still beta, we will use our own UU to  # DBU
# conversion functions previously mentioned in Section 7.

```python
bndpts = pts2dbu([0.0, 0.0], [0.0, 10.0], [10.0, 10.0], [10.0, 0.0])

# Create the cell boundary
bnd = oa.oaPRBoundary.create(block, bndpts)
```
A layer blockage can be used to block layers
- Focus in training will be on “routing” blockages, but a few other layer blockage types exist (see oaBlockageType)
- A routing blockage tells routers to not route in a given area

```
# Create a M2 routing blockage
m2 = oa.oaLayer.find(tech, "m2").getNumber()
blkpts = pts2dbu( [[0.0, 0.0], [0.0, 20.0], [10.0, 20.0], [10.0, 0.0]] )
m2blk = oa.oaLayerBlockage.create(block, 'routing', m2, blkpts)
```
An oaTrackPattern is a formula which specifies where routing on a given layer will go.

Track pattern includes/excludes:
- Includes: orientation (horiz/vert), layer, num tracks, begin point, pitch
- Excludes: allowed width(s), expected wire net names, end points

Orientation (horiz/vert) is in the direction of the pitch (not the track orientation)

```python
ml = oa.oaLayer.find(tech, "m1").getNumber() horiz = True
oa.oaTrackPattern.create(blk, horiz, cd2dbu(0.5), 5, cd2dbu(0.2), ml)
```
oaTrackPattern (Complex)

- Complex grids can be created by using multiple track patterns
- Need to calculate number of tracks to end at a certain point
- Example below:
  - Metal orientation will be vertical (so use “horizontal” track pattern)
  - Power/Ground track will use wider metal width and start the pattern at 0.20 offset from the origin (note metal width is not stored on Track Patterns)
  - Signal track will be at 0.50, 0.70, 0.90, 1.10
  - Entire pattern set repeats at 1.20 pitch
Calculating Number of oaTrackPattern Tracks

- There is no end location in oaTrackPatterns, instead the number of patterns is used to stop but must be calculated
- Example function on calculating number of tracks

```python
def tpnum(begin, pitch, end):
    # Calculate number of tracks to use given begin, pitch, end
    num = int((end - begin) / pitch)
    remain = (end - begin) - (num * pitch)
    tot = num
    if remain > sys.float_info.epsilon:
        # add another tp for the remainder
        tot += 1
    if num > 0:
        # num is number of pitches - add 1 since 2 tp in one pitch
        tot += 1
    return(tot)

# Note: Machine epsilon gives an upper bound on the relative error due to rounding in floating point arithmetic.
```
Finding Special Objects in an oaBlock

• Earlier, we found out that shapes can be queried using `oaBlock::getShapes()`
  – This will not return the special objects mentioned in this section
• Use the following to get the special objects mentioned in this section from an oaBlock:
  – `oaPRBoundary::find`
  – `oaBlock::getBlockages` (be sure to check the blockage type)
  – `oaBlock::getTrackPatterns`
Lab 8.1: Create OA Objects in an oaBlock

• Goal - Become familiar in working with oaPRBoundary, oaLayerBlockage, and oaTrackPattern objects

• Write a script to create oaBlock to hold special OA objects:
  1. Create layout oaDesign called “objects” under library “mylib”
  2. Create an oaPRBoundary with lower-left at 0,0 that is 10um square
  3. Create an oaLayerBlockage with lower-left at 0,0 that is 3um square
  4. Create vertically oriented (“horizontal”) oaTrackPatterns for “m1” using the track pattern sequence on the “complex” track pattern slide and fit within the 10um square boundary

Compare your script to labs/8.1/objects.py
Section 8 Summary

- Learned about, and used, the `oaPRBoundary` object
- Demonstrated use of `oaLayerBlockage` object to block routing
- Explored `oaTrackPattern` object and how it is defined
- Discovered techniques for accessing these objects
Silicon Integration Initiative

www.si2.org

For details contact Marshall Tiner
Director of Production Standards

mtiner@si2.org