



How EDA Vendors Approach Interoperability

Synopsys Interoperability Conference November 2006

Aspects of Analog & Custom Interoperability

1. Database Interoperability
2. PCell Interoperability
3. Tool Conventions & Protocols
4. Interoperability with Legacy Data

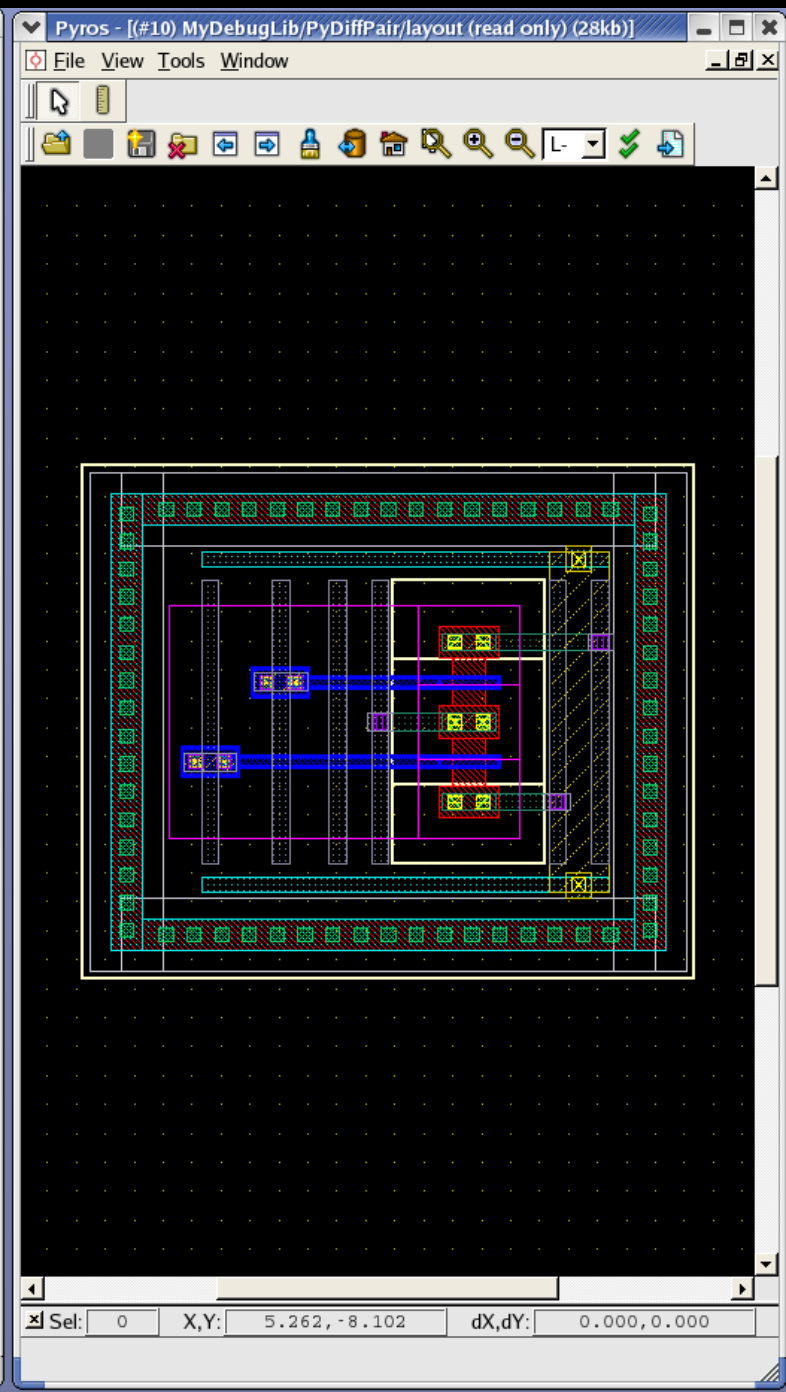
1. Database Interoperability

- OpenAccess has this one handled
 - Supported by every major EDA vendor, with more joining every day
 - Leading tools delivered or announced on OpenAccess
 - No meaningful alternatives
- Kudos to the EDA industry for pulling this one off!

2. PCell Interoperability

- Free, Modern Solution: PyCell Studio
 - Creates interoperable OpenAccess PCells
 - Work with every OpenAccess tool tested to date
 - Modern API, IDE, graphical viewer, interactive DRC
 - Available for free download at www.ciranova.com
- Other techniques also exist

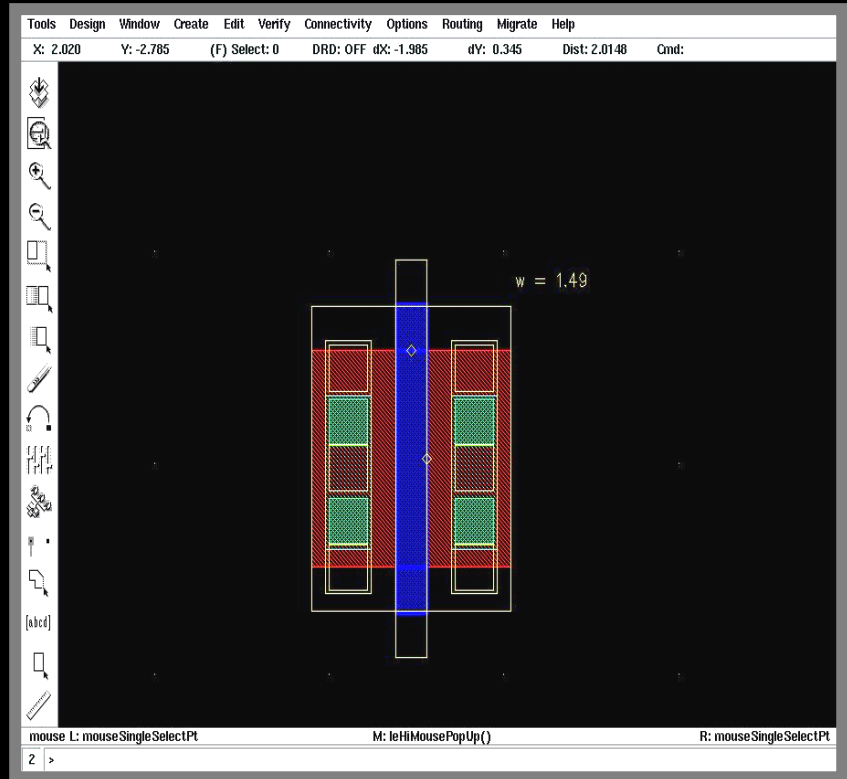
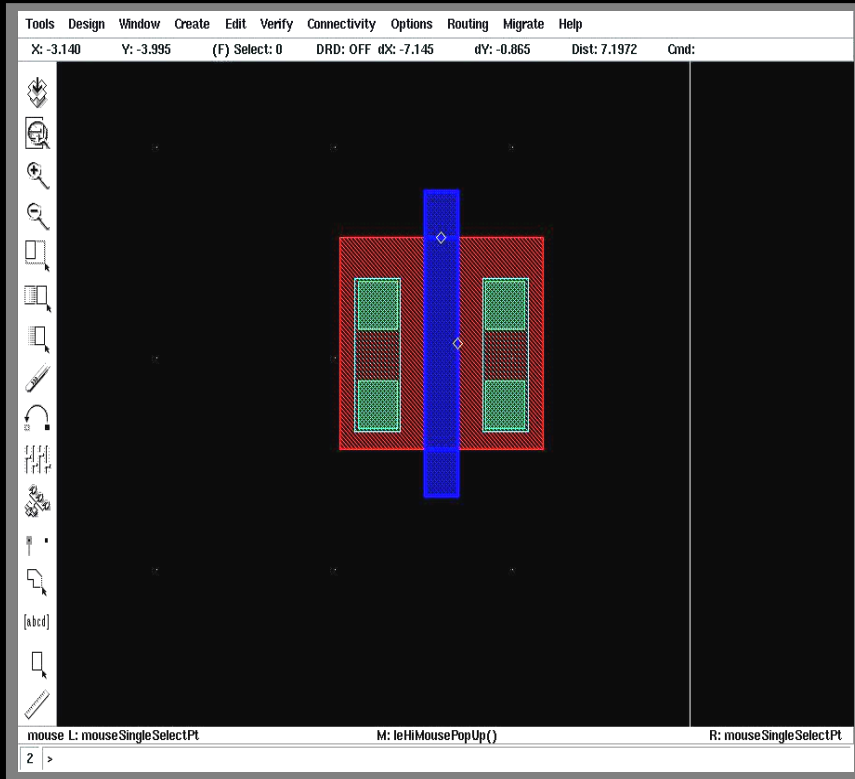
```
Wing IDE: diffPair.py (tmp/file8Ay5L5_cni_felix/MyPyCells)
File Edit Source Project Debug Tools Window CiraNova Help
New Open... Save Save All Cut Copy Paste Undo Redo
Goto Definition Search Indent Dedent Match Indent
Project Properties Break Debug Run To Cursor Stop Pause
Step Into Step Over Step Out Up Stack Down Stack
init__.py diffPair.py
PyDiffPair createRing
554 w2 = self.drainGroup.getBBox().getWidth()
555 l2 = self.shieldTopBar.getBBox().top - self.shieldBottomBar
556 self.drainShieldBar = Bar( self.drainShieldLayer, Direction
557 #overlap the group
558 self.place( self.drainShieldBar, Direction.EAST, self.drain
559
560 #drop contacts between to the side shield
561 c3 = Contact( self.drainShieldLayer, self.shieldLayer, "Shi
562 c3.moveTo( Point( self.drainShieldBar.getBBox().getCenterX(
563 c4 = Contact( self.drainShieldLayer, self.shieldLayer, "Shi
564 c4.moveTo( Point( self.drainShieldBar.getBBox().getCenterX(
565
566
567
568 def createRing(self):
569 """Create guard ring.
570 """
571 ContactRing( self.tech.getLayer( "diff" ), self.strapLayer, "B",
572
573
574
575 def createRouting( self):
576 """Route gates and diffusion contacts to bars.
577 """
578
579 for i in range( self.fingers):
580 Route.StraightLine( self.T1Units[i].instPin( "G" ), self.gat
581 Route.StraightLine( self.T1Units[i].instPin( "D" ), self.dra
582 Route.StraightLine( self.T1Units[i].instPin( "S" ), self.sou
583
584 for i in range( self.fingers):
585 Route.StraightLine( self.T2Units[i].instPin( "G" ), self.gat
586 Route.StraightLine( self.T2Units[i].instPin( "D" ), self.dra
587 Route.StraightLine( self.T2Units[i].instPin( "S" ), self.sou
588
```



3. Conventions & Protocols

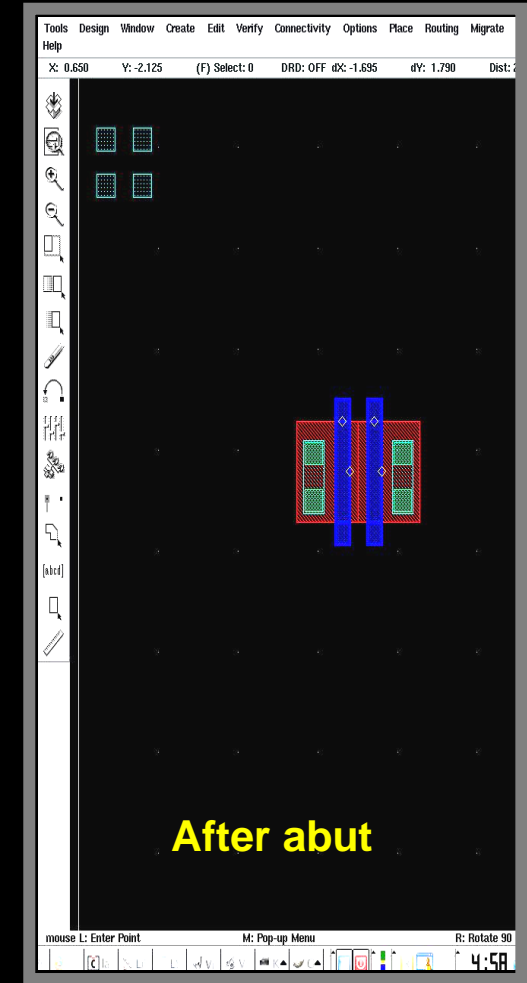
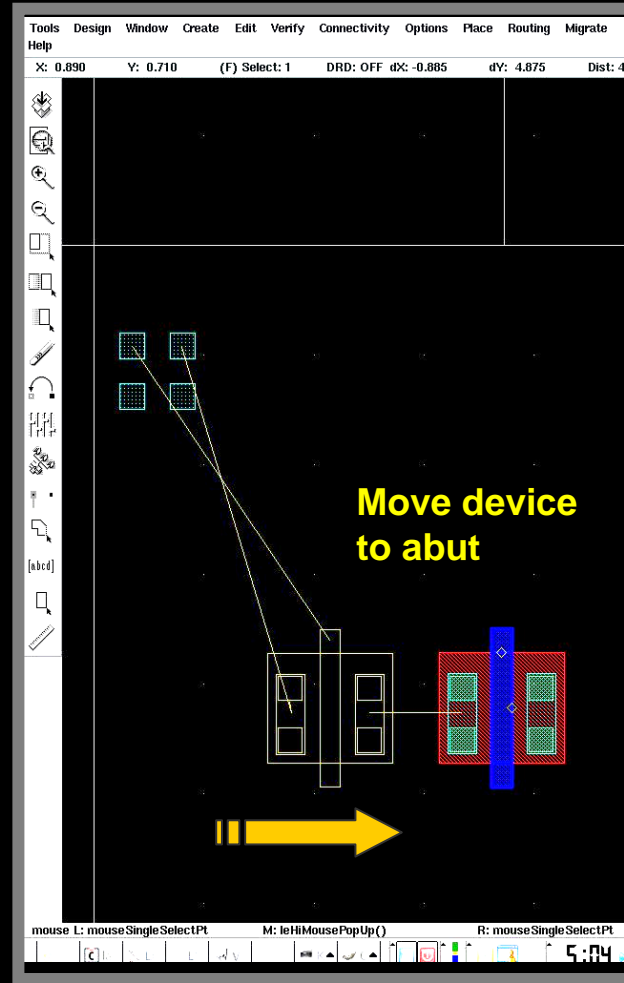
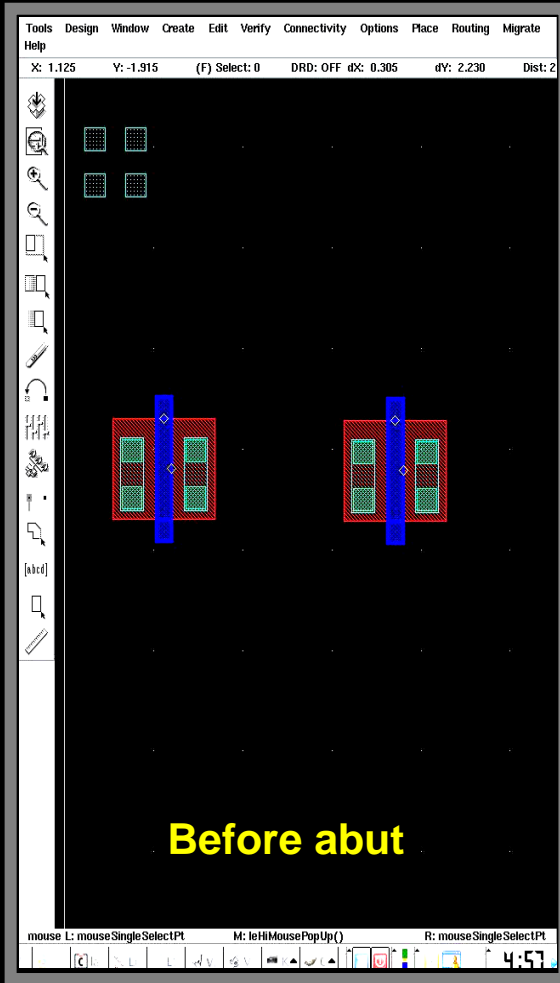
- This one gets a little trickier
- Each application has a different way of handling:
 - Stretch handles
 - Automatic abutment
 - Layer purpose pair associations
 - Communication to other tools (placer, router)
- A truly interoperable solution must address these

Stretch Handles with PyCells



Dragging stretch handle increases gate width & number of source / drain contacts

Automatic Abutment with PyCells



PyCell Interoperability Table

	Read PyCells	Parameterize PyCells	PyCell Stretch Handles	PyCell Auto Abutment
Cadence Virtuoso	✓	✓	✓	✓
AWR Analog Office	✓	✓	2007	2007
Mentor Calibre	✓			
Silicon Canvas Laker	✓	2007	2007	2007
Silicon Navigator RDE	✓	✓	✓	2007

4. Interoperability with Legacy Data

- How do we get SKILL PCells into other tools?
 - This one is the hardest of all
 - SKILL evaluator is proprietary to Cadence
- Translation?
 - Not 100% solution
 - Some people trying it: <https://sourceforge.net/projects/sk2py>
- Emulation?
 - Writing an emulator is a never ending job
 - You must emulate SKILL language & SKILL APIs
- Persistent PCells?