

IPL Constraints Working Group Update

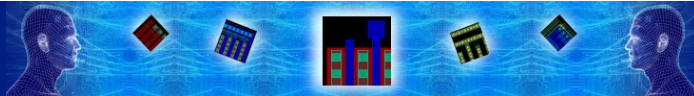
20th EDA Interoperability Developer's Forum

Ed Petrus, Ciranova

James Roberts, Silicon Canvas

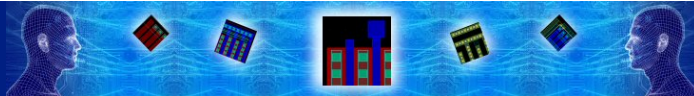
IPL Constraint Working Group

- Industry lacks an open standard for specifying and handling design constraints
- EDA companies to collaborate, create and donate a standard
- Must be open and extensible
- Strong preference for a base level text representation
 - Avoid the need for special APIs to read/write constraints
- Binary, in-memory representation left up to individual companies
 - As long as text in/out is supported



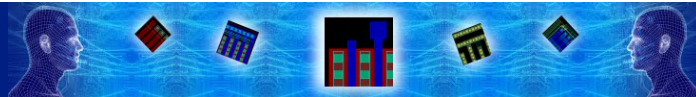
Proposed Activities

- Move quickly to define an open, text based syntax for design constraints
- Need to agree on the nature of the syntax
 - Do we invent a new syntax?
 - Do we borrow a syntax from somewhere?
 - Should we aim for one of the industry standard data definition formats such as XML or YAML?
- Second Step: define specific formats for specific design constraints



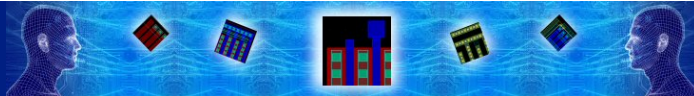
Constraints Syntax Proposal

- Suggest we do not invent yet another syntax
- Narrow the choice down to XML or YAML
- Ciranova has been prototyping constraints in YAML and is open to sharing the knowledge with the IPL group
 - YAML (“Yet Another Markup Language”) is the same idea as XML but simpler
 - More human readable/writable – one can read the text and make sense of the constraints
 - Successfully incorporated a parser into an OA-based tool
 - <http://www.yaml.org>



Example

- The following 2 pages contain a prototype constraint file coded in YAML
 - For a VCO type analog circuit



```
# A YAML-based constraint specification
```

```
#  
# Top-level terminals and power supplies
```

```
#
```

```
Terminals:
```

```
- [Name,      Type,      Param, Operation]  
# -----  
- [VAA,      GlobalNet,  top,      ModifyAdd]  
- [VSS,      GlobalNet,  bottom,   ModifyAdd]  
- [VIN,      GlobalNet,  left,     ModifyAdd]  
- [VREF,     GlobalNet,  left,     ModifyAdd]  
- [FN,       GlobalNet,  right,    ModifyAdd]
```

```
#
```

```
# Global ruleset directive
```

```
#
```

```
Rulesets:
```

```
- [Name,      Appliesto]  
# -----  
- [recommended, []]  
- [default,    [MN3, MP3]]
```

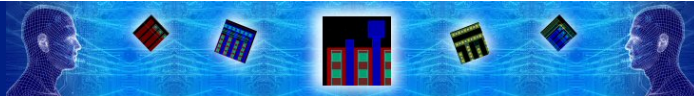
```
#
```

```
# Top-level ContactRings
```

```
#
```

```
contactRings:
```

```
- [StartLayer, EndLayer, ImplantLayer, FillLayers, NetName, xOffset, yOffset]  
# -----  
- [diff,      metall,  pimp,      [],      none,      0.5,      0.5]  
- [diff,      metall,  nimp,      [],      none,      0,        0]  
  
# Multi-fill-layer contact ring  
- [diff,      metall,  nimp,      [nwell,od2],VAA,      0,        0]
```



```
#  
# Combine primitives  
#  
Combines:  
- [Name, Param, Contains, SharedTerms, TargetFixedShape]  
# -----  
- [MN1_MN2, idNmos, [MN1, MN2], [Source, Sub], false]
```

```
# Constraints & Directives Specification - order matters!  
#
```

```
Constraints:
```

- Name: uc0
Type: Halo
Param: {leftOffset: 0.1, rightOffset: 0.1, bottomOffset: 0.1, topOffset: 0.1}
Contains: [RP1_RP2]

- Name: uc1
Type: column
Param: {Alignment: center, YFlipvec: [false, false]}
Contains: [MN1_MN2, uc0]

- Name: uc2
Type: row
Param: {Alignment: bottom, xFlipvec: [false, true]}
Contains: [MN8, MN9]

- Name: uc5
Type: Partition
Param: []
Contains: [MP4, MP6, MP8, MN6]

- Name: uc6
Type: Clone
SourceName: uc7
NameMap: {MP3: MP4, MP5: MP6, MP7: MP8, MN7: MN6}
NetMap: {VSS: VSS, VAA: VAA, N1N181: N1N188, CLKN1: CLKN1, VON: VOP, CLK: CLK}
Contains: [uc5]

- Name: uc7
Type: Partition
Param: []

