Open Process Specification

Gilles NAMUR

May 12th, 2014
AGENDA

What is OPS ?

What does OPS Looks like ?

What kind of Automation from OPS ?

- DRC
- Techfile
- FE Device Library
- Device Pcells
Automated PDK Generation – OPS in Action

SI2 OPDK Coalition Board
Chair: Jim Culp, IBM

OpenPDK Technical Steering Group (TSG)
Chair: Gilles Namur, ST

- ESD
- Open Process Specification (OPS)
- Symbols, Callbacks & Parameters (SCP)
- Pcell XML Packaging
- Pcell Common Language Grammar
- Unified Layer Model (Jointly with DFMC)

- OPS to OA TechDB
- OPS to DRC

OpenPDK Working Groups

12-May-14
Several PDK Input formats
Several PDK Generation flows
Several EDA Tools for the same feature to be supported
What is the target of OpenPDK Coalition?

**OpenPDK Coalition Introduction**

A set of open standards to allow an OpenPDK to be created once and then translated into specific EDA vendor tools and specific foundry formats.

Let's Start with PDK Inputs format → Open Process Specification

[http://www.si2.org/?page=1118](http://www.si2.org/?page=1118)
Open Process Specification (OPS) is a standardized format for exchanging all data needed to generate a complete PDK.

This is the SI2 OPS standard
A smart & complete electronic PDK input format

- Allow Automation for PDK generation.
- Eases EDA Vendor sync. with foundries Inputs.

**OPS.xml**

- `<Process>`
- `<Libraries>`
- `<Devices>`
- `<Layers>`
- `<Rules>`
- `<...>`

**Foundries**

**Inputs DB**

**PDK**

**EDA Tools**
What ever is the format of the Database used by the supplier to manage the PDK inputs:

The PDK generation flow remains the same.
Why OPS?

Set of PDK Inputs: DRM & Device Spec

Foundry 1

Foundry 2

Foundry 3

Set of PDK Inputs: DRM & Device Spec (Format C)

OPS.xml

<Process>
<Libraries>
<Devices>
<Layers>
<Rules>
...
...

OPS in Action

Customer A
Design & Validation
With Tool Suite A

Customer B
Design & Validation
With Tool Suite B

Customer C
Design & Validation
With Tool Suite C

PDK 1
For Tool Suite A

PDK 1
For Tool Suite B

PDK 1
For Tool Suite C

PDK 2
For Tool Suite A

PDK 2
For Tool Suite B

PDK 2
For Tool Suite C

PDK 3
For Tool Suite A

PDK 3
For Tool Suite B

PDK 3
For Tool Suite C
### Machine Readable Layer Declaration in OPS.xml

```xml
<ops:RootLayer name="OD" abbreviation="OD" destination="core">
  <ops:ToolMappingNumber number="17" tool="OA"/>
</ops:RootLayer>

<ops:Purpose opsName="drawing" name="drawing" abbreviation="drg" destination="core">
  <ops:ToolMappingNumber number="-1" tool="OA"/>
</ops:Purpose>

<ops:CADLayer name="OD;drawing" alias="OD" opsRole="DIFFUSION" role="DIFFUSION" description="Defines ACTIVE Area" destination="core" expoNumber="1">
  <ops:RefRootLayerName value="OD"/>
  <ops:RefPurposeName value="drawing"/>
  <ops:StreamIO format="GDSII" number="1" dataType="0"/>
  <ops:StreamIO format="OASIS" number="501" dataType="500"/>
  <ops:ToolMappingNumber number="1001" tool="Calibre"/>
  <ops:Display transparencyOrder="1" visible="True" selectable="True" con2ChgLy="True" drgEnbl="True" valid="True" stipple="stipple3" lineStyle="lineStyle0" fill="lime" outline="lime" fillStyle=""/>
</ops:CADLayer>
```

### Human Readable Layer List

<table>
<thead>
<tr>
<th>Root</th>
<th>Purpose</th>
<th>Alias</th>
<th>Destination</th>
<th>Comment</th>
<th># of MASK</th>
<th>Role</th>
<th>GDS #</th>
<th>DAS #</th>
<th>OAS #</th>
<th>OAS dt</th>
<th>LSW order</th>
<th>LSW visible</th>
<th>LSW select</th>
<th>Con 2</th>
<th>Chat</th>
<th>Drg Enbl</th>
<th>Valid</th>
<th>Stipple</th>
<th>Line Style</th>
<th>Fill</th>
<th>Outline</th>
<th>Calibre Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>OD</td>
<td>drawing</td>
<td>OD</td>
<td>core</td>
<td>Defines ACTIVE Area</td>
<td>1</td>
<td>DIFFUSION</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>501</td>
<td>500</td>
<td>1</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>stipple3</td>
<td>lineStyle0</td>
<td>lime</td>
<td>lime</td>
<td>1901</td>
<td></td>
</tr>
<tr>
<td>OD</td>
<td>phi</td>
<td>ODphi</td>
<td>core</td>
<td></td>
<td>1</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>501</td>
<td>500</td>
<td>3</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>black</td>
<td>mediumLine</td>
<td>lime</td>
<td>lime</td>
<td>1903</td>
<td></td>
</tr>
<tr>
<td>OD</td>
<td>net</td>
<td>ODnet</td>
<td>core</td>
<td></td>
<td>1</td>
<td></td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>501</td>
<td>511</td>
<td>4</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>stipple2</td>
<td>lineStyle0</td>
<td>lime</td>
<td>lime</td>
<td>1904</td>
<td></td>
</tr>
<tr>
<td>OD</td>
<td>boundary</td>
<td>ODbndry</td>
<td>core</td>
<td></td>
<td>1</td>
<td></td>
<td>1</td>
<td>1</td>
<td>2</td>
<td>501</td>
<td>512</td>
<td>6</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>black</td>
<td>lineStyle0</td>
<td>lime</td>
<td>lime</td>
<td>1905</td>
<td></td>
</tr>
<tr>
<td>OD</td>
<td>fill_blockage</td>
<td>ODfill_blockage</td>
<td>core</td>
<td></td>
<td>1</td>
<td></td>
<td>1</td>
<td>2</td>
<td>0</td>
<td>501</td>
<td>520</td>
<td>6</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>stipple2</td>
<td>lineStyle0</td>
<td>lime</td>
<td>lime</td>
<td>1906</td>
<td></td>
</tr>
<tr>
<td>Nu</td>
<td>drawing</td>
<td>Nu</td>
<td>core</td>
<td>Defines N/E</td>
<td>0</td>
<td>0</td>
<td>501</td>
<td>500</td>
<td>7</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>black</td>
<td>lineStyle0</td>
<td>orange</td>
<td>orange</td>
<td>1907</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VTH_N</td>
<td>drawing</td>
<td>VTH_N</td>
<td>core</td>
<td>Defined to reflect VTH</td>
<td>0</td>
<td>0</td>
<td>501</td>
<td>500</td>
<td>8</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>stipple35</td>
<td>lineStyle0</td>
<td>gold</td>
<td>gold</td>
<td>1908</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>VTH_P</td>
<td>drawing</td>
<td>VTH_P</td>
<td>core</td>
<td>Defined to reflect VTH</td>
<td>0</td>
<td>0</td>
<td>501</td>
<td>500</td>
<td>9</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>stipple15</td>
<td>lineStyle0</td>
<td>pink</td>
<td>pink</td>
<td>1909</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>OD18</td>
<td>drawing</td>
<td>OD18</td>
<td>core</td>
<td>Defines 18 V</td>
<td>0</td>
<td>0</td>
<td>501</td>
<td>500</td>
<td>10</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>stipple15</td>
<td>lineStyle0</td>
<td>magenta</td>
<td>magenta</td>
<td>1910</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PO</td>
<td>fill</td>
<td>DPO</td>
<td>core</td>
<td>Defines fill</td>
<td>0</td>
<td>0</td>
<td>501</td>
<td>500</td>
<td>11</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>stipple12</td>
<td>lineStyle0</td>
<td>red</td>
<td>red</td>
<td>1911</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>PO</td>
<td>fill</td>
<td>DPO</td>
<td>core</td>
<td>Defines fill</td>
<td>0</td>
<td>0</td>
<td>501</td>
<td>500</td>
<td>12</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>stipple12</td>
<td>lineStyle0</td>
<td>red</td>
<td>red</td>
<td>1912</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

### What does OPS look like?

The provided code snippet in OPS.xml demonstrates a layer declaration for the root layer `OD` with a purpose `drawing` and associated attributes. The layer declaration includes definitions for mask numbers, tool mappings, stream data output formats, display attributes, and other layer-specific properties. This is an example of how automated PDK generation in OPS can be implemented and managed.
What does OPS look likes?

Human Readable DRM

7.3.2 DERIVED GEOMETRIES

- NW1V: NW .not. OD18  
  *NW of core device*
- NW2V: NW .and. OD18  
  *NW of I/O device*
- Gate: OD .and. PO  
  *PO over OD*

Machine Readable OPS.xml

```xml
<ops:DerivedLayer name="Gate" description="PO over OD">
  <ops:Template name="AND">
    <ops:RefCADLayerAlias value="OD"/>
    <ops:RefCADLayerAlias value="PO"/>
  </ops:Template>
</ops:DerivedLayer>

<ops:DerivedLayer name="PP_OD" description="Heavily doped P type OD that does not include transistor channels">
  <ops:Template name="NOT">
    <ops:DerivedLayer name=""/>
    <ops:Template name="AND">
      <ops:RefCADLayerAlias value="OD"/>
      <ops:RefCADLayerAlias value="PP"/>
    </ops:Template>
  </ops:Template>
  <ops:RefCADLayerAlias value="PO"/>
</ops:DerivedLayer>
```
What does OPS look like?

7.3.4.6 Poly (PO)

<table>
<thead>
<tr>
<th>PO Design Rules (POLY)</th>
<th>A</th>
<th>0.040</th>
</tr>
</thead>
<tbody>
<tr>
<td>PO.W.1 PO width</td>
<td>B</td>
<td>0.100</td>
</tr>
<tr>
<td>PO.S.1 PO space</td>
<td>C</td>
<td>0.050</td>
</tr>
<tr>
<td>PO.A.1 PO area</td>
<td>D</td>
<td>0.050</td>
</tr>
<tr>
<td>PO.A.2 PO enclosed area</td>
<td>E</td>
<td>0.150</td>
</tr>
<tr>
<td>PO.W.2 Gate (Transistor) length, under OD16</td>
<td>F</td>
<td>0.140</td>
</tr>
<tr>
<td>PO.S.2 Gate distance to PO in Source/Drain direction</td>
<td>G</td>
<td>0.220</td>
</tr>
<tr>
<td>PO.S.3 PO space on OD, under OD18</td>
<td>H</td>
<td>0.150</td>
</tr>
<tr>
<td>PO.S.4 PO space if at least one PO width is &gt; 0.120 µm (W) and if the parallel run length is &gt; 0.140 µm (L)</td>
<td>M</td>
<td>0.160</td>
</tr>
</tbody>
</table>

Human Readable DRM

Machine Readable OPS.xml

```xml
<ops:Rule name="PO.S.4">
  <opc:Documentation sectionTitle="PO Design Rules (POLY)">
    <opc:Description>PO space if at least one PO width is > 0.120 µm (W) and if the parallel run length is > 0.140 µm (L)</opc:Description>
  </opc:Documentation>
  <ops:Template name="">
    <opc:Parameter name="V1" value="0.120" type="FLOAT"/>
    <opc:Parameter name="V2" value="0.140" type="FLOAT"/>
    <opc:Parameter name="MAIN" value="0.160" type="FLOAT"/>
    <opc:SIUnit prefix="µ" siName="m"/>
  </ops:Template>
</ops:Rule>
```
What does OPS look like?

<table>
<thead>
<tr>
<th>Library</th>
<th>device</th>
<th>description</th>
<th>Parameter</th>
<th>Pin Order</th>
<th>Netlister</th>
</tr>
</thead>
<tbody>
<tr>
<td>demo45</td>
<td>nfet</td>
<td>Regular-Vt FET</td>
<td>w &amp; l</td>
<td>D;G;S;B</td>
<td>CDL</td>
</tr>
</tbody>
</table>

```
<edsDevice name="nfet" libName="demo45" description="Regular-Vt FET" symbolName="nmos3" symbolPinOrderMappingList="D=d;G=g;S=s;B=b" layoutName="nfet" layoutType="PCELL" >
  <edsDeviceParameter name="w" description="Total Gate Width" defaultValue="240n" type="string" units="lengthMetric" >
    <edsDDFDeviceParameter callback="Check_Width_Fet('w)" parseAsNumber="1" parseAsCEL="1" editable='cdfgData->dimensionMode->value="1"'' display="1" storeDefault="1" />
  </edsDeviceParameter>
  <edsDeviceParameter name="l" description="Gate Length" defaultValue="80n" type="string" units="lengthMetric" >
    <edsDDFDeviceParameter editable="1" callback="Check_Length_Fet()" parseAsNumber="1" parseAsCEL="1" display="1" storeDefault="1" />
  </edsDeviceParameter>
  <edsDeviceNetlistingInfos>
    <edsDeviceNetlistingInfo toolName="auCdl" modelName="nfet" netlistProcedure="_ansCdlCompParamPrim" instParameters="(w l)" componentName="mos" termOrder="(d g s b)" namePrefix="M" />
  </edsDeviceNetlistingInfos>
</edsDevice>
```
OPS 1.1

- **OPS v1.0** was targeted at process producing companies (Foundries), to enable study and creation of electronic versions of DRM, to gather early feedback on completeness.

- **OPS 1.1** is the first release targeted for consideration by EDA Vendors complete enough to enable DRC & technology file creation.

  **Contents**
  - Connectivity below M1 (including local interconnect).
  - Manufacturing, DRC & Layout DB grids
  - Improved enumerations
  - Multi Patternning Support in Layers Definition
  - Tool mapping / Interfaces (from Tools Interface WG)

  **Implementation example:** Demo DRM 45nm v1.6 aligned on OPS 1.1
This table describes the connectivity of this metal stack but also the connectivity bellow M1 level.

Derived Layers (Derived Geometries) can be used in the connectivity description.

Ex: **PONET**: PO .not. RPO for *Salicied Poly net*
### Connectivity in OPS 1.1

**XML Representation**

<table>
<thead>
<tr>
<th>Ordering</th>
<th>From</th>
<th>Through</th>
<th>To</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>AP</td>
<td>CB</td>
<td>M7Z</td>
</tr>
<tr>
<td>2</td>
<td>M7Z</td>
<td>VIA6Z</td>
<td>M6Z</td>
</tr>
<tr>
<td>3</td>
<td>M6Z</td>
<td>VIA5Z</td>
<td>TOPMIM</td>
</tr>
<tr>
<td>4</td>
<td>M6Z</td>
<td>VIA5Z</td>
<td>BOTMIM</td>
</tr>
<tr>
<td>5</td>
<td>M6Z</td>
<td>VIA5Z</td>
<td>M5X</td>
</tr>
<tr>
<td>6</td>
<td>M5X</td>
<td>VIA4X</td>
<td>M4X</td>
</tr>
<tr>
<td>7</td>
<td>M4X</td>
<td>VIA3X</td>
<td>M3X</td>
</tr>
<tr>
<td>8</td>
<td>M3X</td>
<td>VIA2X</td>
<td>M2X</td>
</tr>
<tr>
<td>9</td>
<td>M2X</td>
<td>VIA1X</td>
<td>M1</td>
</tr>
<tr>
<td>10</td>
<td>M1</td>
<td>V0</td>
<td>CB</td>
</tr>
<tr>
<td>11</td>
<td>M1</td>
<td>V0</td>
<td>CA</td>
</tr>
<tr>
<td>12</td>
<td>CB</td>
<td>touching</td>
<td>CA</td>
</tr>
<tr>
<td>13</td>
<td>CB</td>
<td>touching</td>
<td>PONET</td>
</tr>
<tr>
<td>14</td>
<td>CA</td>
<td>touching</td>
<td>NPONET</td>
</tr>
<tr>
<td>15</td>
<td>CA</td>
<td>touching</td>
<td>PPONET</td>
</tr>
<tr>
<td>16</td>
<td>NPONET</td>
<td>touching</td>
<td>PPONET</td>
</tr>
<tr>
<td>17</td>
<td>NPONET</td>
<td>touching</td>
<td>NW</td>
</tr>
<tr>
<td>18</td>
<td>PPONET</td>
<td>touching</td>
<td>PW</td>
</tr>
<tr>
<td>19</td>
<td>NW</td>
<td>touching</td>
<td>DNW</td>
</tr>
<tr>
<td>20</td>
<td>PW</td>
<td>NODNW</td>
<td>CHIP</td>
</tr>
</tbody>
</table>

```xml
<xml>
  <title>Connectivity for 7M-5X-2Z-MIM metatization option</title>
  <ConnectObject connectOrdering="1" connectBy="via">
    <FromLayer> <RefRootLayer name="AP"/> </FromLayer>
    <ThroughLayer> <RefRootLayer name="CB"/> </ThroughLayer>
    <ToLayer> <RefRootLayer name="M7Z"/> </ToLayer>
  </ConnectObject>
  ...
</xml>
```
Multi Patterning Support in OPS

Based on the Cadence Contribution to OA:

**Data Model Specification**

*Multi-Pattern Technology Support Document Revision: 1.0 Document*

*Date: 11-07-2012 Template Version 2.5*

© 2012 Cadence Design Systems, Inc. All rights reserved worldwide.

We have then added 3 “information” in the OPS to support Multi Patterning: the “exposition Number”, the “mask Color” and the “lock Status”.

```xml
<ops:CADLayer name="M1_E1" alias="M1_E1" enumRole="CUSTOM" role="" description="Defines first exposure for M1" destination="core" expoNumber="2" maskColor="MaskColor1" lockStatus="locked">
  <ops:RefRootLayer value="M1"/>
  <ops:RefPurpose value="drawing"/>
  <ops:StreamIO format="GDSII" number="21" dataType="234"/>
  <ops:StreamIO format="OASIS" number="521" dataType="734"/>
  <ops:ToolMappingNumber number="1029" tool="Calibre"/>
  <ops:Display transparencyOrder="23" visible="true" selectable="true" con2ChgLy="true" drgEnbl="true" valid="true" stipple="stipple30" lineStyle="lineStyle0" fill="violet" outline="violet" fillStyle=""/>
</ops:CADLayer>
```
ULM - The center of several Si2 standards

Common infrastructure for OpenDFM, OpenPDK and OpenAccess

Open Process Specifications (OPS)
XML Description of layers, rules, regions

ULM Layer Equations
Semantics and Syntax

Open DFM Parser

Bidirectional

Mentor plug-in
Synopsys plug-in
Cadence plug-in

Calibre DRC Runset
ICV DRC Runset
PVS DRC Runset

OPS Introduction
ULM Summary

- The Universal Layer Model
  - Standardizes more than two dozen layer operators
  - Provides an efficient method for custom operators
  - Qualified with test cases by all major DRC engines

- ULM is an infrastructure compatible with multiple Standards
  - Derive layer operations 100% compatible with OpenDFM
  - UML layer ops are 100% compatible with OpenAccess
  - The XML description is 100% compatible with OPS
  - Truth tables and log files used for mask data accounting
A concrete OPS example

- All along the process of definition of OPS, ST has contributed with several DRM example: demo DRM 45nm.
- The example contains a complete DRM.pdf and its associated OPS.xml file aligned with the OPS.xsd defined by SI2 OPDKC OPS WG.

ST contributed with an updated DRM & OPS.xml aligned with OPS1.1
- New Connectivity Table added
- OPS Version
- New Grids information
- Local Interconnect layers added
- Multi Patterned Layers added
- CutSize Vias support added
- New information added in the CAD Layer description.
How could you construct your OPS?

- **Merge Operation**
  - Check the consistency in between the data from both initial OPS
    - If consistent → Merge
    - If not → need source correction = iteration

**Ground Ruler**
- DRM DB
- OPS

**Layer Owner**
- Layer Info
- OPS

**Device Owner**
- Device Spec
- OPS

**Process Eng.**
- Process Data
- OPS

**DRM DB**
- Layer Info
- Device Spec
- Process Data
- DRM DB

**OPS Introduction**

12-May-14
What kind of Automation from OPS?

OPS.xml

<Process> <Libraries> <Devices> <Layers> <Rules> <...> <...>

Enriched OPS.xml

<Process> <Libraries> <Devices> <Layers> <Rules> <Templates> <...> <...>

GroundRuler

DRM DB

OPS

Enriched OPS

Generator

DRC Code

DRC Coder

openDFM Template DB

In-House Template DB

New Template

Reference to

New Template
OPS to DRC Generation

OPS XML

OpenDFM File

ULM Equation

Calibre, PVS or ICV Runsets
OPS to DRC Generation

OPS Connectivity

OpenDFM Connectivity Rules

Automatic generation
OPS to DRC Generation

The Big Picture

Demo Scope

1. OPS2ODFM Translation Input
   - Exactly one line per translation

2. OPS2ODFM Template
   - XML xPaths
   - ODFM Syntax
   - Processing Node

3. OpenDFM Runset
   - OpenDFM file is easily viewed/edited by Verification Engineers

~ 14,000 lines of OPS XML

Si2 XSD Verification Of OPS XML
OPS to DRC Generation

OPS XML to OpenDFM is Simpler

OPS XML

- **ops:Template**
  - `name`: check_space

- **opc:Parameter**
  - `name`: MAIN
  - `value`: 0.080
  - `type`: FLOAT

- **opc:SIUnit**
  - `prefix`: µ
  - `siName`: m

- **ops:RefCADLayerAlias**
  - `value`: OD

OpenDFM

- Clear intent with good semantics and syntax

```
# OpenDFM translation of OPS XML Rule Name: 'OD.S.1'
dfmc::check_space -rule_name OD.S.1 \
  -in_layer OD \
  -space_less_than 80.0 \
  -cmnt "OD space"
```

- Automatic conversion from Micron to Nanometer
What are Composite Rules?

Note: Composite Rules are made up from several parts

OPS XML Template

OpenDFM Template
<ops:Rules>
  <ops:Rule name="OD.S.1">
    <ops:Documentation sectionTitle="OD Design Rules">
      <ops:Description>OD Space</ops:Description>
    </ops:Documentation>
    <ops:Template name="" drcTool="OpenDFM">
      <ops:Parameter name="MAIN" value="0.080" type="FLOAT">
        <ops:SIUnit prefix="\u" siName="m"/>
      </ops:Parameter>
      <ops:RefCADLayerAlias value="OD"/>
    </ops:Template>
    <ops:DRCTools>
      <odfm:OpenDFM>
        <odfm:check_space>
          <odfm:space_less_than attr="value" relPath="opc:Parameter[@name='MAIN']/@value"/>
          <odfm:in_layer layerType="cadLayer0" attr="value" relPath="ops:RefCADLayerAlias/@value"/>
        </odfm:check_space>
      </odfm:OpenDFM>
      <myNS:Customer>
        <myNS:customSpaceCheck>
          <myNS:getMinSpaceValue attr="value" relPath="opc:Parameter[@name='MAIN']/@value"/>
          <myNS:getLayerName layerType="cadLayer0" attr="value" relPath="ops:RefCADLayerAlias/@value"/>
        </myNS:customSpaceCheck>
      </myNS:Customer>
    </ops:DRCTools>
  </ops:Rule>
</ops:Rules>
We now have good experience with translating OPS XML into OpenDFM rules

- 90% of the OPS rules were easy. Usually 1 – 6 OpenDFM functions per DRC check
- 9% were moderately difficult (> 6 functions) but all of the information was directly available
- 1% were ill formed. The information for the check was not directly stored in the OPS XML

We have a handful of proposed changes to discuss, modify and approve

- We have a method to support multiple DRC/LVS/PEX engines in the OPS XML
- Allows the functions OpenDFM, Calibre, ICV and PVS to coexist in the XML
- Either none, one or all of the DRC template formats can exist in the OPS XML

We plan to use attributes that contain relative xPaths from the DRC function to the OPS Data

- All DRC functions only use xPaths to access the data values in the ops:Template
- The DRC functions can NOT contain duplicates of the data in the ops:Template
- A change the data in the ops:Template must immediately be available to all DRC engines
- The xPath concept should also be used by OpenPCell, LVS, PEX, OPEX to access OPS data
- When duplicate data exists in different sections of the XML the data integrity is diminished
- When the golden source OPS XML data is updated then all of the duplicate data and the data derived from this data must all be consistently updated
Techfile
Automation
OPS To Techfile Generation

TARGET

OPS XML file
<Process>
<Libraries>
<Devices>
<Layers>
<Rules>
<...>
<...>

Techfile Generator

Techfile
All Layers

Design Layers
Derived Layers
Via Definitions
Physical Constraints
Tool Directives
OPS To Techfile Generation

Current overview

- OPS XML File
- OPS2 oaTechDB Translator
- GDSII Layer Map File
- OpenAccess oaTechDB
- Display File
- TechLPP
- lppLayer
- lppLayerNum
- lppPurpose
- lppPurposeNum
- lppFlags
- lppPriority
- lppPacketName
- Display (side file; in formats)
  - Display.xml
  - Display.rdf
  - Display.tcl
- GDS (Layer map side file)
  - LayerName
  - layerPurpose
  - GDS layer number
  - GDS data type
What kind of Automation from OPS?

- In-House Template DB
- openDFM Template DB
- New Template

GroundRuler

Techfile Coder

DRM DB

DRC Coder

OPS

Enriched OPS

Generator

Techfile Code

DRC Code

Reference to

Reference to

New Template
OPS To Techfile Generation

```python
def minimumSpacing (layername, values, refRule) :
    _layername = layername[0]
    _mainValue = values[0]
    _refRule = refRule

    template = """spacings(
        ( minSpacing "layername" {mainValue} 'ref '{refrule}"
    ) ;spacings"

    return template.format(layername=_layername, mainValue=_mainValue, refrule=_refRule)
```

**minimumSpacing**: Techfile Constraint Template Documentation

**Definition**: Specifies the minimum spacing between two shapes on the same layer.

**Example**:

<table>
<thead>
<tr>
<th>OD.S.1</th>
<th>OD space</th>
<th>0.080</th>
</tr>
</thead>
</table>

**OPS Introduction** 12-May-14
OPS To Techfile Generation

OPS.XML – Rules Enriched with techfile template name & args

```xml
<ops:Rule name="OD.S.1">
    <opc:Documentation sectionTitle="OD Design Rules">
        <opc:Description>OD space</opc:Description>
    </opc:Documentation>
    <ops:Techfile_Template name="minimumSpacing">
        <ops:RefCADLayerAlias value="OD"/>
        <opc:Parameter name="MAIN" value="0.080"/>
    </ops:Techfile_Template>
    <ops:Template name="">
        <opc:Parameter name="MAIN" value="0.080" type="FLOAT">
            <opc:SIUnit prefix="\mu" siName="m"/>
        </opc:Parameter>
        <ops:RefCADLayerAlias value="OD"/>
    </ops:Template>
</ops:Rule>
```
Techfile vs DRC

OPS Introduction

Gilles NAMUR

12-May-14
OPS to Techfile Optimisation

Enriched OPS.xml

<Process> <Libraries> <Devices> <Layers> <Rules> <Templates > ... ...

Techfile Template

techfile = "<templates>
  {techfile "templates" "files" ("files")}
  {techfile "templates" "files" ...
other techfile "templates" ...

Techfile Optimisator

Primary Techfile with « flat » constraints code

One Rule → One Template & One code / No table

Final Techfile with optimized constraints code

OPS Introduction

12-May-14
OPS Introduction

Next Steps

Bi-Directional Translation

OPS2
oaTechDB
Translator

GDSII
Layer Map
File

OpenAccess
oaTechDB

Display
Files

OPS.xml
<Process>
<Libraries>
<Devices>
<Layers>
<Rules>
<...>

OPS.xsd
The OPS DB is a Python virtual DB.
Relations between object instances are exactly as represented in OPS UML.
**PDK Automation : openLibGen**

- **OpenLibGen** is an ST Internal API & Tools.
- PDK Device Library generation push button flow from an OPS.xml as input.

**OPS XML file**
- `<Process>`
- `<Libraries>`
- `<Devices>`
- `<Layers>`
- `<Rules>`
- `…`
- `…`

**OPS API (Python)**

**Device Library & techfile**

---

**OPS Introduction**

12-May-14
PDK Automation : openLibGen 1/2

- openLibGen generation flow:
  - Library creation.
  - Symbols creation from a reference Symbol Library. (copy, rename and rename the pins)
  - Copy the pcells referenced in the OPS from the Pcells Lib.
  - CDF creation & compilation from OPS (no more template, no other input Data)
  - Copy the callbacks referenced in the OPS from the Callbacks directory. Callbacks are then still coded by the Device Library owner.
  - Netlister views Creation from OPS
  - Device Categories Creation from OPS
  - Techfile Creation & compilation:
    - Automate the layer declaration from a layer list
    - Concat the layer declaration with splitted techfile (vias, controls, constraints, functions) → these part of the techfile are still developed by the techfile owner.
    - Create all the optional techfiles required.
Device parameters from paper spec to library

Paper specification

<table>
<thead>
<tr>
<th>Name</th>
<th>Default value</th>
<th>Abs Min</th>
<th>Abs Max</th>
<th>DK Min</th>
<th>DK Max</th>
<th>Specific Callback</th>
<th>Prompt</th>
<th>Description</th>
<th>LVS/PLS Extracted/compared</th>
<th>Drawn channel width</th>
<th>Extracted/compared</th>
<th>ParseAsNumber</th>
<th>ParseAsCEL</th>
<th>storeDefault</th>
<th>units</th>
</tr>
</thead>
<tbody>
<tr>
<td>w</td>
<td>0.12</td>
<td>0.12</td>
<td>0.12</td>
<td>10</td>
<td>10000</td>
<td>DK_CBmos(w)</td>
<td>yes</td>
<td>yes</td>
<td>both</td>
<td>yes</td>
<td>both</td>
<td>yes</td>
<td>yes</td>
<td>yes</td>
<td></td>
</tr>
</tbody>
</table>

CDF parameter file

cdfCreateParam(cdfId, name "w", prompt "Width (drawn) (um)", units "", type "string", defaultValue "0.12", display "t", editable "t", callback "DK_CBmos(w)", storeDefault "yes", parseAsNumber "yes", parseAsCEL "yes")

Schematic behavior

OPS specification (xml) ➔ xml viewer

OPS specification (xml)

```xml
<edsDeviceParameter name="w">
  <description>Width (drawn) (um)</description>
  <defaultValue>0.12</defaultValue>
  <type>string</type>
  <unit>lengthMetric</unit>
  <callback>DK_CBmos(w)</callback>
  <parseAsNumber>1</parseAsNumber>
  <parseAsCEL>1</parseAsNumber>
  <editable>1</editable>
  <display>1</display>
  <storeDefault>1</storeDefault>
</edsDeviceParameter>
```
Conditional parameter behaviour

<table>
<thead>
<tr>
<th>Name</th>
<th>Default value</th>
<th>Abs Min</th>
<th>Abs Max</th>
<th>Rec Min</th>
<th>Rec Max</th>
<th>Specific Callback</th>
<th>Prompt</th>
<th>Description</th>
<th>LVS/PLS</th>
<th>Type</th>
<th>Applicable to (schema/layout/none/both)</th>
<th>DDF parameters</th>
<th>units</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>DK_GBresRpo(r)</td>
<td>Segment Resistance</td>
<td>&quot;calcmode = Geometry &quot;</td>
<td>yes/yes/yes</td>
<td>string</td>
<td>both</td>
<td>storeDefault &quot;yes&quot;</td>
<td>parseAsNumber &quot;yes&quot;</td>
</tr>
</tbody>
</table>

Paper specification

CDF parameter file

```xml
<edsDeviceParameter name="r" description="Segment Resistance" type="string">
    <edsDDFDeviceParameter callback="DK_GBresRpo('r')" parseAsNumber="1" parseAsCEL="1" editable="true">
        <calcmode>value == "Geometry"</calcmode>
        <display>"1" storeDefault="1" /
    </edsDDFDeviceParameter>
</edsDeviceParameter>
```

OPS specification (xml) - source code example

Schematic behavior: r change capability linked to calcmode
Callback: from specification to library

**Paper specification**

<table>
<thead>
<tr>
<th>Name</th>
<th>Default value</th>
<th>Abs Min</th>
<th>Abs Max</th>
<th>DK value</th>
<th>Abs Min</th>
<th>Abs Max</th>
<th>DK value</th>
</tr>
</thead>
<tbody>
<tr>
<td>r</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
<td>/</td>
</tr>
</tbody>
</table>

Specific Callback:

\[ \text{DK_CBResRpo}(r) \]

**OPS specification (xml)**

```xml
<edsDeviceParameter name="r" description="Segment Resistance" type="string">
  <edsDeviceParameter callback="DK_CBResRpo(r)" parseAsNumber="1" parseAsCEL="1" editable="true" display="true" storeDefault="1" />
</edsDeviceParameter>
```

```xml
<edCreateParam cdId ?name ="r"
  ?prompt ="Segment Resistance"
  ?units ="
  ?type ="string"
  ?defValue ="
  ?display ="true"
  ?editable="true"
  ?calcmode=value ="Geometry"
  ?callback="DK_CBResRpo(r)"
  ?storeDefault="yes"
  ?parseAsNumber="yes"
  ?parseAsCEL="yes"
```

**CDF parameter file**

**Library warning message from callback**

Procedure in native schematic tool language
Symbol configuration

Paper specification

Si2 symbol

Resulting symbol

OPS specification (xml)

Si2 symbol: nmos4

Pin names and order: d g s b
Si2 Symbol name: n(p)mos4

Parameter name | Parameter type | Parameter value
--- | --- | ---
paramLabelSet | display | "w l m n fing"

OPS Introduction
Library category settings

Paper specification
- Category name: diode
- Device description: N(P)+ / P(N)WELL: LP (H)SVT
- Device name: dn(h)svtlp/dp(h)svtlp

OPS specification
```xml
<edsLibraryCategories>
  <edsLibraryCategory name="diode">
    <edsLibraryCategoryDevices list="ansvtlp dpsvtlp dhvtlp dphvtlp" />
  </edsLibraryCategory>
  <edsLibraryCategory name="fet">
    <edsLibraryCategoryDevices list="nsvtlp psvtlp nhvtlp phvtlp" />
  </edsLibraryCategory>
  <edsLibraryCategory name="TJdevices">
    <edsLibraryCategoryDevices list="nsvt18 psvt18 anvt18 dpsvt18" />
  </edsLibraryCategory>
  <edsLibraryCategory name="ILsrar">
    <edsLibraryCategoryDevices list="nsrnl1pd nsrnl1pg psrnl1pu" />
  </edsLibraryCategory>
  <edsLibraryCategory name="resistor">
    <edsLibraryCategoryDevices list="rrporpo rrporpo rpodrpo" />
  </edsLibraryCategory>
</edsLibraryCategories>
```

Resulting category structure
- Everything
  - Uncategorized
  - TJdevices
    - diode
    - fet
    - ILsrar
    - resistor
  - resistor
  - dhvtlp dphvtlp dpsvtlp
Netlisting information behaviour

Paper specification

<table>
<thead>
<tr>
<th>Tool name</th>
<th>Spectre</th>
</tr>
</thead>
<tbody>
<tr>
<td>Model name</td>
<td></td>
</tr>
<tr>
<td>Netlist procedure</td>
<td></td>
</tr>
<tr>
<td>Instance parameters</td>
<td></td>
</tr>
<tr>
<td>Component name</td>
<td></td>
</tr>
<tr>
<td>Device Terminal</td>
<td></td>
</tr>
</tbody>
</table>
| Term order | "d" "g" "s" "b"
| Prefix | X |
| Permute Rule |         |
| Prop mapping | (nil mult m) |
| Term mapping | (nil d l:1 g l:2 s l:3 b l:4) |
| ModeParamExptList |         |

OPS specification (xml)

```
<toolName>spectre</toolName>
<instParameters>
  (w l ad pd as ps nfinger mul s sa sb src first numcos numcod pocus pocode ngcon)
</instParameters>
<termOrder>
  ("d" "g" "s" "b")
</termOrder>
<termMapping>
  (nil d l:1 g l:2 s l:3 b l:4)
</termMapping>
<propMapping>
  (nil mult m)
</propMapping>
```

Spectre netlist generated for the aside mos

// Library name: test
// Cell name: demo
// View name: schematic

```
PMO (net1 net3 net4 net2) psvtlp w=0.12 l=0.04 ad=1.32e-02 pd=3.40e-01
  as=1.32e-02 ps=3.40e-01 nfinger1 mult=1 src first=1 ngcon=1
```

psvtlp

w=0.12
l=0.04
m=1
nfinger=1

Gilles NAMUR

OPS Introduction
From techlayer to maptable and display

Source file for technology layer description

| GL_Name | Layer_Name | Purpose_Name | H# | 10s 15s | 10L 15L | 10S 15S | chidgrp | chidoc | DASID | chidoc | 10S15S | 10L15L | 10S15S | 10L15L | Purpose_Order | LRP_Order | Design_Packet_Name | Vis | Sel | D2ChqL | D2FbrL | Valid | Stipple | LineStyle | FalL
<table>
<thead>
<tr>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>RX</td>
<td>OD</td>
<td>drawing</td>
<td>1</td>
<td>0</td>
<td>4</td>
<td>1</td>
<td>1</td>
<td>130</td>
<td>RX</td>
<td>drawing</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>RX</td>
<td>drawing</td>
<td>ch021</td>
<td>solid</td>
<td>line</td>
<td>RX</td>
<td>drawing</td>
<td>ch222</td>
<td>color2</td>
<td>RX</td>
<td>drawing</td>
<td>ch222</td>
</tr>
<tr>
<td>RX</td>
<td>OD</td>
<td>boundary</td>
<td>1</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td>150</td>
<td>RX</td>
<td>boundary</td>
<td></td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>RX</td>
<td>boundary</td>
<td>ch222</td>
<td>solid</td>
<td>line</td>
<td>RX</td>
<td>boundary</td>
<td>ch222</td>
<td>color2</td>
<td>RX</td>
<td>boundary</td>
<td>ch222</td>
</tr>
<tr>
<td>RX</td>
<td>OD</td>
<td>tile</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td>160</td>
<td>RX</td>
<td>tile</td>
<td>nil</td>
<td>nil</td>
<td>nil</td>
<td>nil</td>
<td>RX</td>
<td>tile</td>
<td>ch222</td>
<td>solid</td>
<td>line</td>
<td>RX</td>
<td>tile</td>
<td>ch222</td>
<td>color2</td>
<td>RX</td>
<td>tile</td>
<td>ch222</td>
</tr>
<tr>
<td>RX</td>
<td>OD</td>
<td>pin</td>
<td>1</td>
<td>6</td>
<td>1</td>
<td></td>
<td></td>
<td>340</td>
<td>RX</td>
<td>pin</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>RX</td>
<td>pin</td>
<td>ch222</td>
<td>solid</td>
<td>line</td>
<td>RX</td>
<td>pin</td>
<td>ch222</td>
<td>color2</td>
<td>RX</td>
<td>pin</td>
<td>ch222</td>
</tr>
<tr>
<td>RX</td>
<td>NW</td>
<td>net</td>
<td>1</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td>400</td>
<td>RX</td>
<td>net</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>RX</td>
<td>net</td>
<td>ch222</td>
<td>solid</td>
<td>line</td>
<td>RX</td>
<td>net</td>
<td>ch222</td>
<td>color2</td>
<td>RX</td>
<td>net</td>
<td>ch222</td>
</tr>
<tr>
<td>RX</td>
<td>NW</td>
<td>drawing</td>
<td>1</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td>48</td>
<td>RX</td>
<td>drawing</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>t</td>
<td>RX</td>
<td>drawing</td>
<td>ch222</td>
<td>solid</td>
<td>line</td>
<td>RX</td>
<td>drawing</td>
<td>ch222</td>
<td>color2</td>
<td>RX</td>
<td>drawing</td>
<td>ch222</td>
</tr>
<tr>
<td>RX</td>
<td>NW</td>
<td>pin</td>
<td>2</td>
<td>1</td>
<td>0</td>
<td></td>
<td></td>
<td>100</td>
<td>RX</td>
<td>pin</td>
<td>nil</td>
<td>nil</td>
<td>nil</td>
<td>nil</td>
<td>RX</td>
<td>pin</td>
<td>ch222</td>
<td>solid</td>
<td>line</td>
<td>RX</td>
<td>pin</td>
<td>ch222</td>
<td>color2</td>
<td>RX</td>
<td>pin</td>
<td>ch222</td>
</tr>
<tr>
<td>RX</td>
<td>NW</td>
<td>net</td>
<td>2</td>
<td>10</td>
<td></td>
<td></td>
<td></td>
<td>120</td>
<td>RX</td>
<td>net</td>
<td>nil</td>
<td>nil</td>
<td>nil</td>
<td>nil</td>
<td>RX</td>
<td>net</td>
<td>ch222</td>
<td>solid</td>
<td>line</td>
<td>RX</td>
<td>net</td>
<td>ch222</td>
<td>color2</td>
<td>RX</td>
<td>net</td>
<td>ch222</td>
</tr>
<tr>
<td>RX</td>
<td>NW</td>
<td>boundary</td>
<td>2</td>
<td>30</td>
<td></td>
<td></td>
<td></td>
<td>110</td>
<td>RX</td>
<td>boundary</td>
<td>nil</td>
<td>nil</td>
<td>nil</td>
<td>nil</td>
<td>RX</td>
<td>boundary</td>
<td>ch222</td>
<td>solid</td>
<td>line</td>
<td>RX</td>
<td>boundary</td>
<td>ch222</td>
<td>color2</td>
<td>RX</td>
<td>boundary</td>
<td>ch222</td>
</tr>
</tbody>
</table>

Generated maptable

<table>
<thead>
<tr>
<th>Packet</th>
<th>OD</th>
<th>1 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>OD</td>
<td>1 30</td>
<td></td>
</tr>
<tr>
<td>OD</td>
<td>1 1</td>
<td></td>
</tr>
<tr>
<td>OD</td>
<td>1 10</td>
<td></td>
</tr>
<tr>
<td>OD</td>
<td>1 20</td>
<td></td>
</tr>
<tr>
<td>NW</td>
<td>2 0</td>
<td></td>
</tr>
<tr>
<td>NW</td>
<td>2 10</td>
<td></td>
</tr>
<tr>
<td>NW</td>
<td>2 20</td>
<td></td>
</tr>
<tr>
<td>NW</td>
<td>2 30</td>
<td></td>
</tr>
</tbody>
</table>

Generated display ressource file

```plaintext
drDEFINE PACKET(
    ;( Display_Name, Packet_Name, Stipple, LineStyle, Fill, Outline )
    ( display NW drawing, blank, solid, orange, orange, orange),
    ( display NW label, dashf2, solid, orange, orange, orange),
    ( display NW pin, blank, thickLine, orange, orange, orange),
    ( display NW boundary, blank, solid, orange, orange, orange),
    ( display RX drawing, dashf3, solid, lime, lime, lime),
    ( display RX label, dashf2, solid, lime, lime, lime),
    ( display RX pin, blank, thickLine, lime, lime, lime),
    ( display RX boundary, blank, solid, lime, lime, lime)
)
```

Generated sections of techfile

```plaintext
;( Layer_Name, Purpose, Packet, Vis, Sel, D2ChqL, D2FbrL, Valid )

; NW

; ( NW, drawing, NW drawing, t, t, t, t )
; ( NW, net, NW label, t, t, t, t )
; ( NW, pin, NW pin, t, t, t, t )
; ( NW, boundary, NW boundary, t, t, t, t )

; OD

; ( OD, drawing, RX drawing, t, t, t, t )
; ( OD, net, RX label, t, t, t, t )
; ( OD, pin, RX pin, t, t, t, t )
; ( OD, tile, RX fill, nil, nil, t, nil, nil )
; ( OD, boundary, RX boundary, t, t, t, t )

; ( Layer_Name, Layer#, Abbreviation )

; NW

; ( NW, 1, OD )
; ( NW, 2, NW )
; ( NW, 3, VTH_N )
; ( NW, 4, VTH_P )
```

**OPS Introduction**
ST OA to OPS Model

**OPS Introduction**
Device Library Documentation for End-User.
Pcells Automation
Problems OpenPCell Addresses

- More productive programming effort
- Write once Pcell and Callback code
- Multi flow support
- Ease of integration into a PDK
- Assure high quality
OPS with its PCells XML Repository solution is language agnostic and with the use of translators, the data can be adopted to any companies proprietary solution.
Open Pcells

XML Pcell Repository
- Ties pcell to PDK
- Defines superMaster
- Operational Parameters
- PDK Parameters
- Code Body
- Integration Testing

Data Definitions

Description Language
- Single grammar
- Pcell and Callback definitions
- Restricted Python Syntax
- Pcell API’s

Function Definitions
OpenPCell Repository - XML

- **Goals:**
  - Standardize an OpenPCell definition for OpenPDK Specification (OPS)
  - Integrate the pcell into the OPS PDK
  - Demonstrate the creation of pcells from the definition in the implementation languages

- **XML definition**
  - Operational parameters (OpParams)
  - Device/PDK integration parameters (PdkParams)
  - Generated Layers
  - Code Body
  - Test benches

- **XSD**
  - Translation from XML into pcell definitions
THANK YOU

QUESTIONS ?