cdn LIVE EMEA 2011

DEF File Export Considering Overlapping Structures

Andreas Krinke

Dresden University of Technology, IFTE
Dresden, Germany

Ralf Eckhard Stephan

Robert Bosch GmbH
Reutlingen, Germany

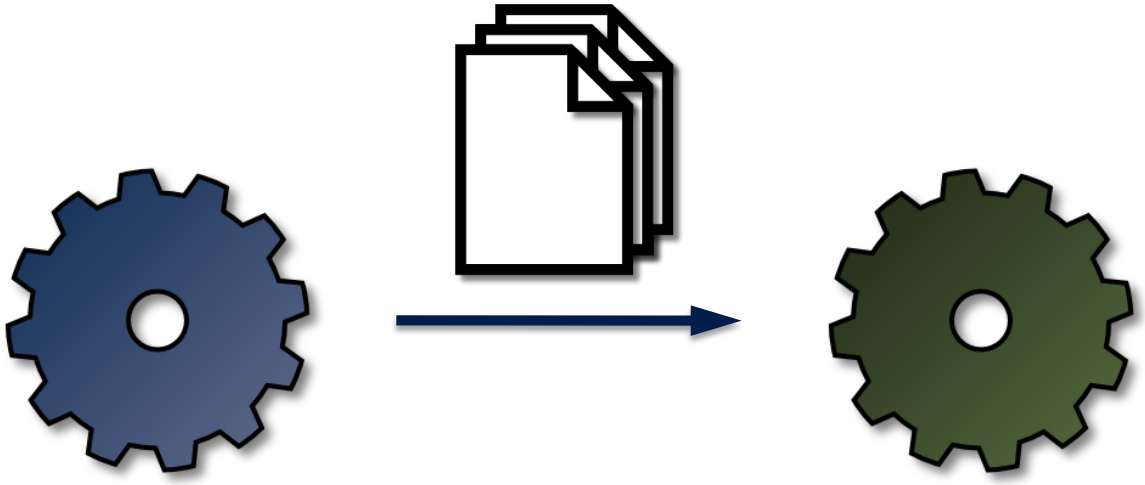
Munich, May 5, 2011

Outline

- **Introduction**
- **Approach**
- **Implementation**
- **Challenges**
  - Automatic Shape Conversion
  - Design data consistency
- **Conclusion**

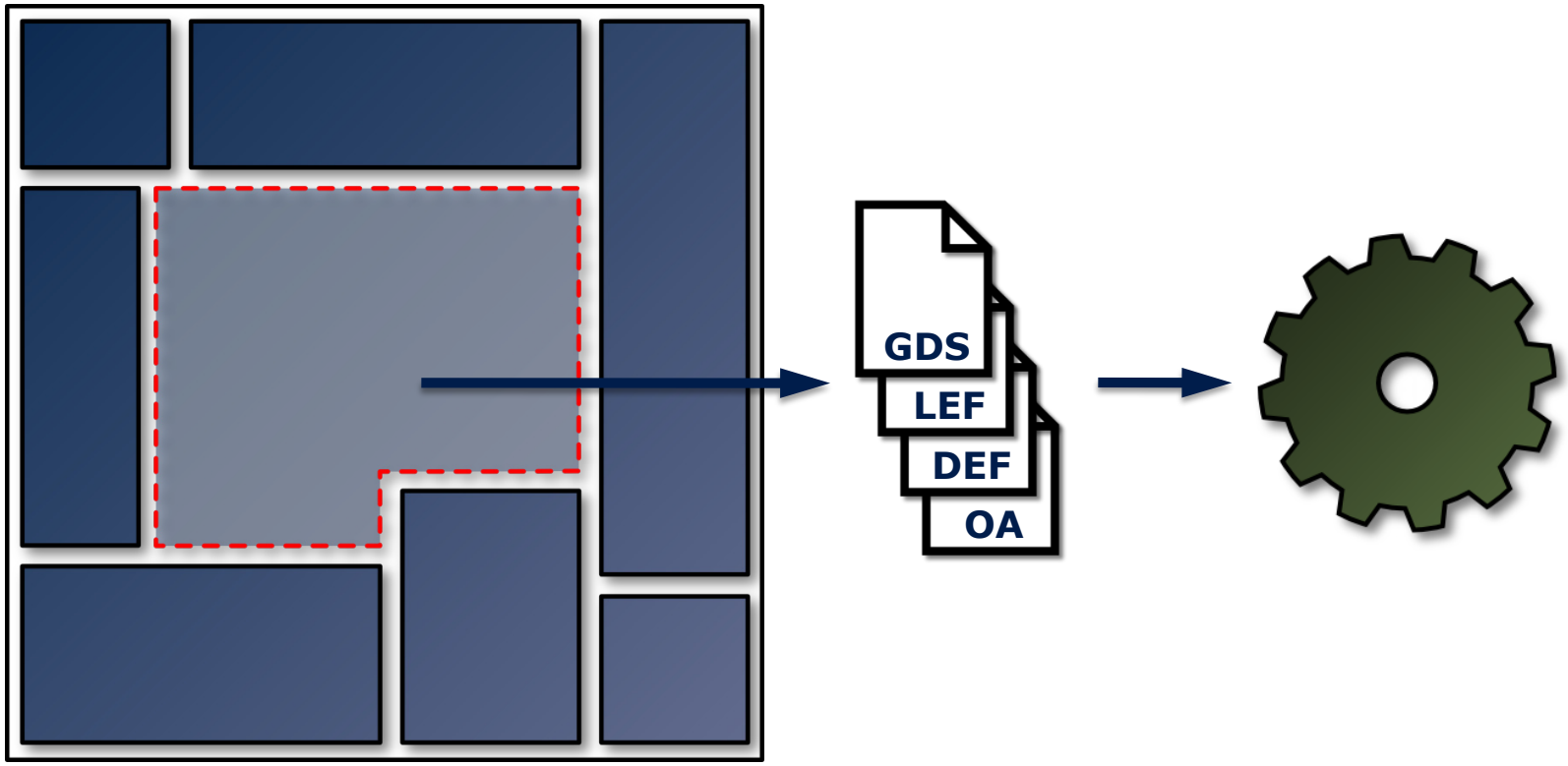
A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Introduction



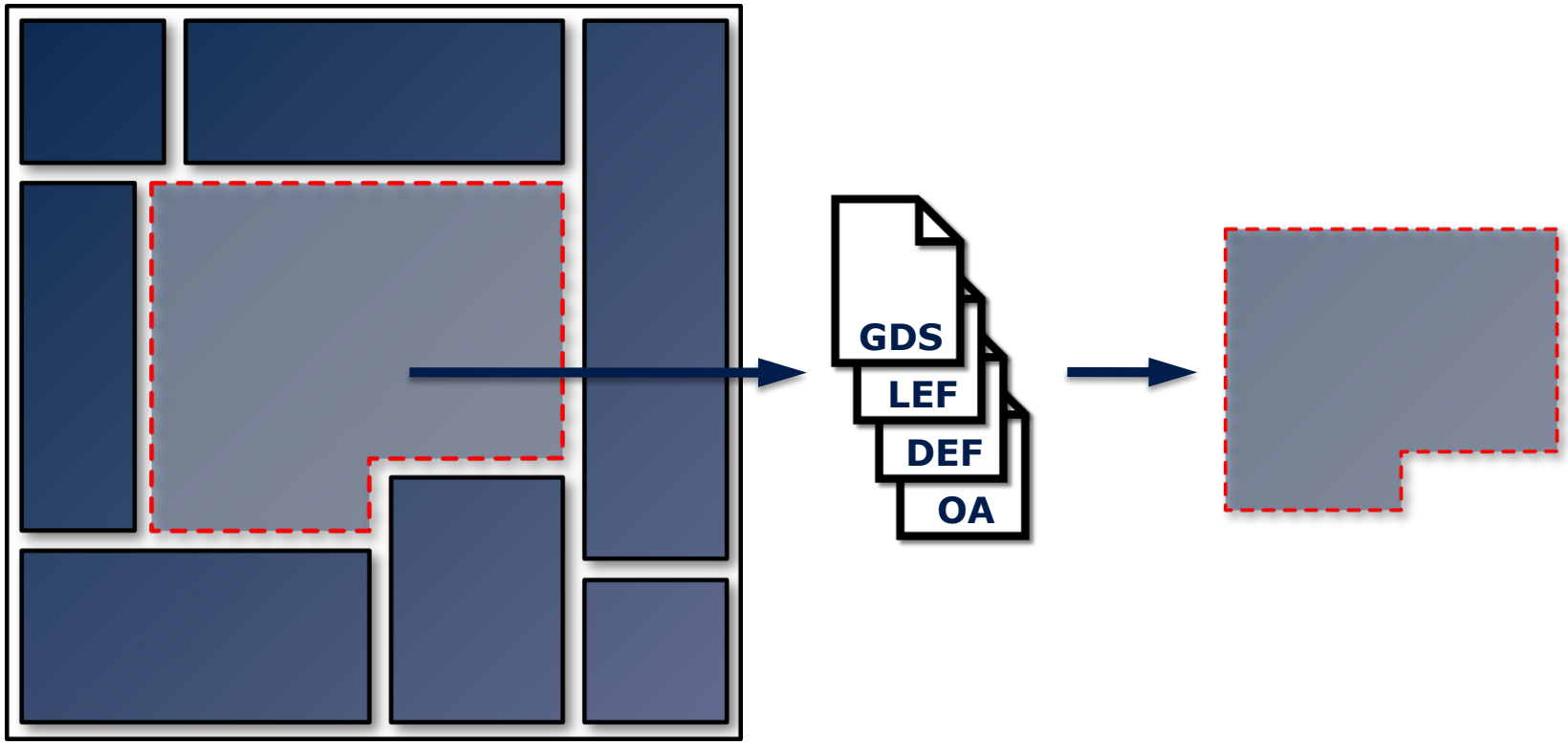
A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Introduction



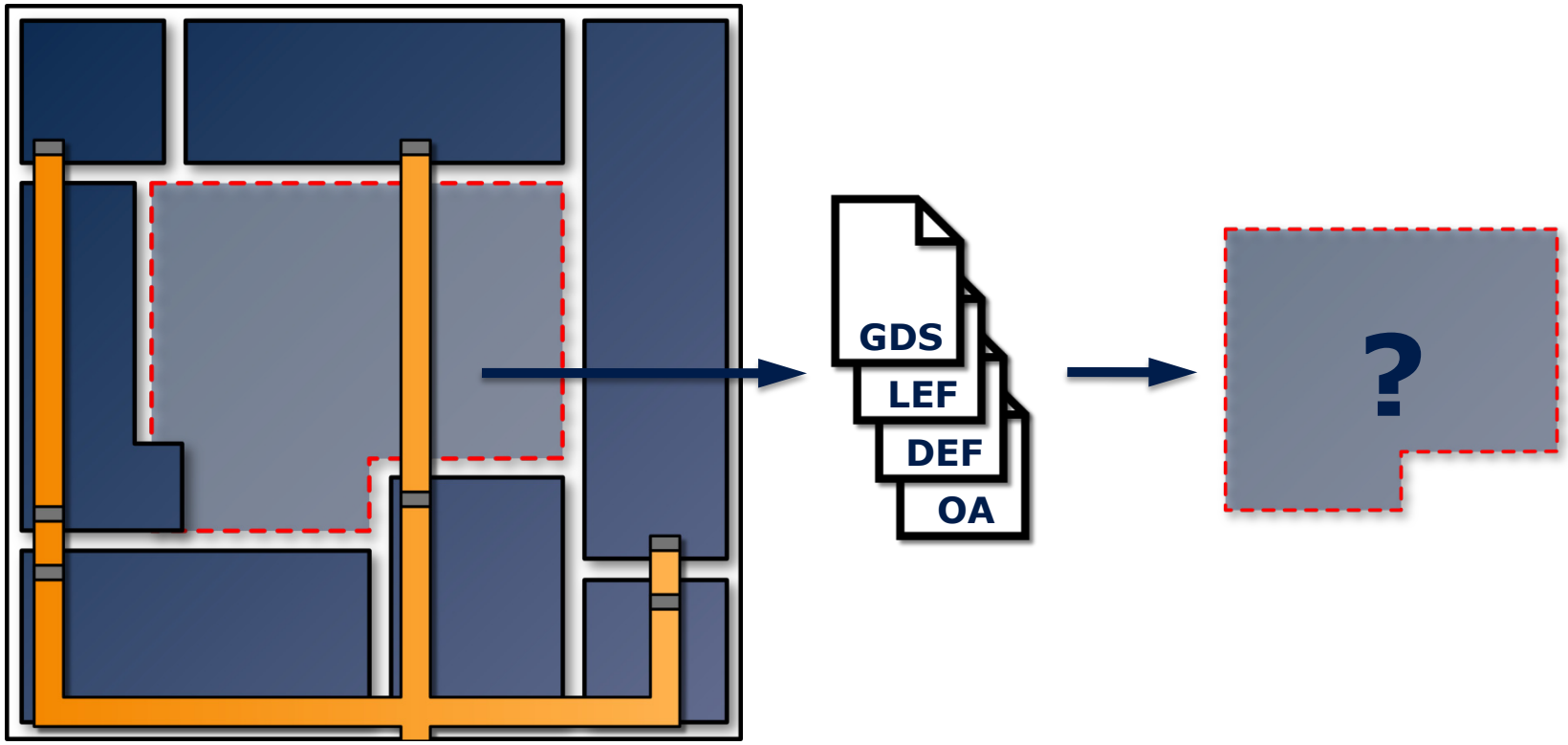
A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Introduction



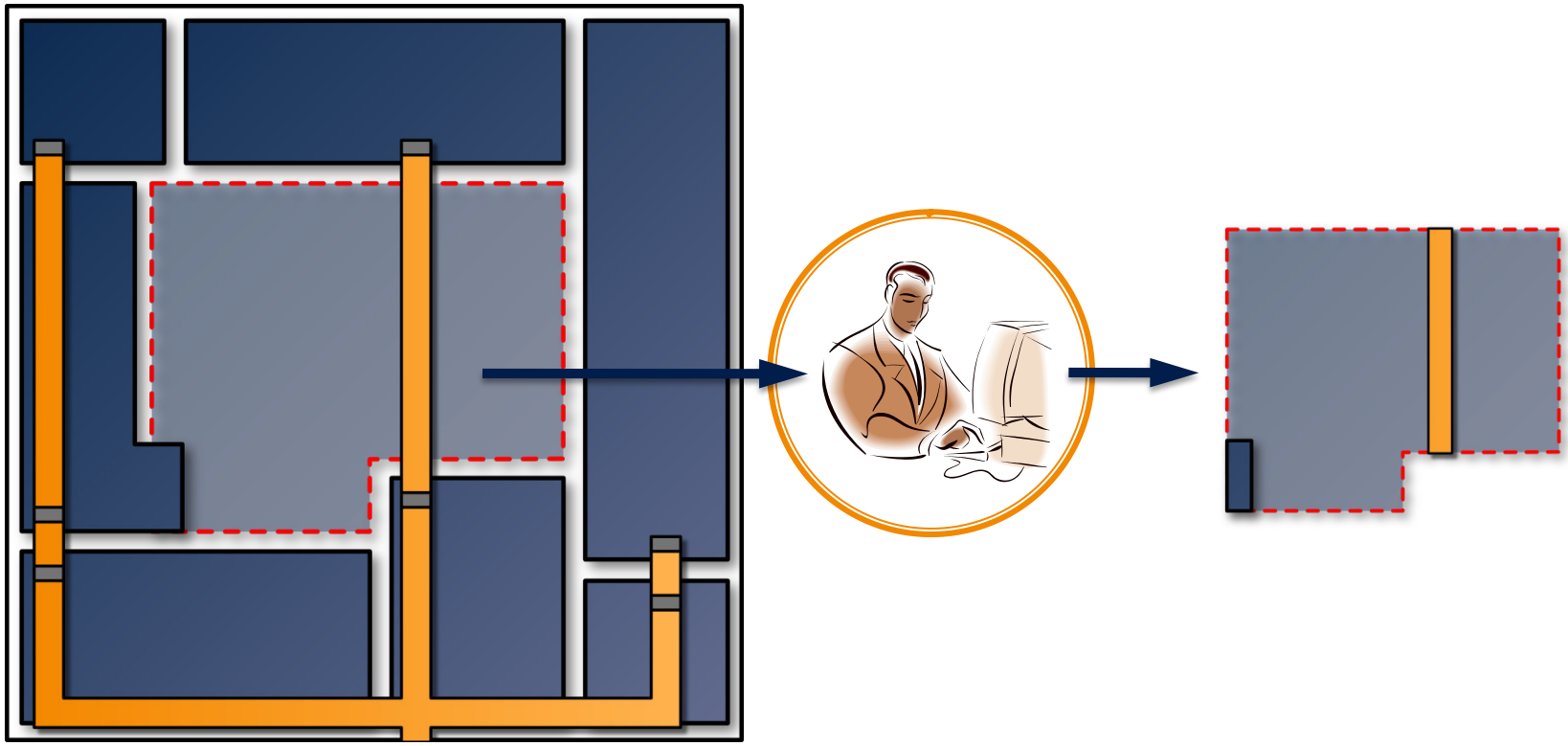
A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Introduction



A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Introduction



A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

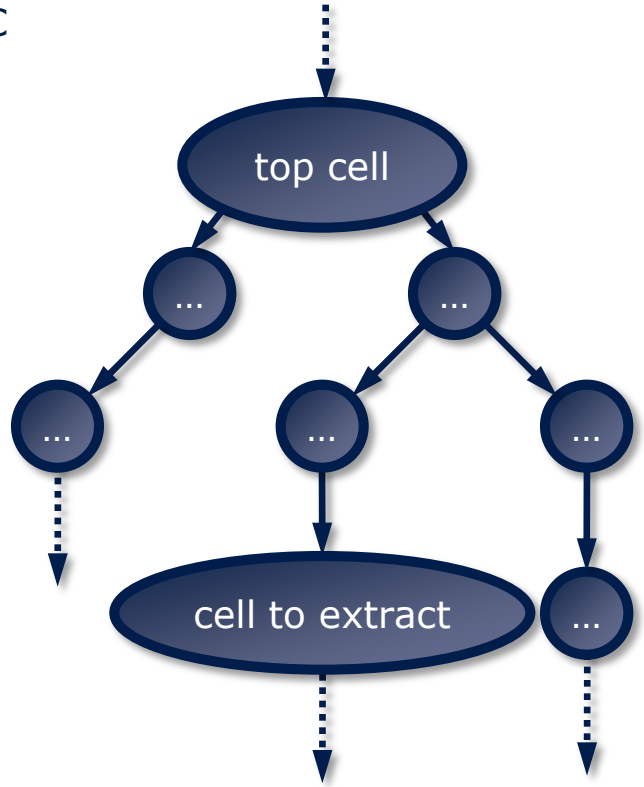
Outline

- Introduction
- **Approach**
- Implementation
- Challenges
  - Automatic Shape Conversion
  - Design data consistency
- Conclusion

A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Approach

- Creation of a SKILL script for Cadence CIC
- Input:
  1. Cell to extract,
     e.g. the digital core
  2. Arbitrary higher-level cell,
     e.g. the top cell
- Output:
  - DEF file of the cell including all overlapping shapes



A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Approach

- Start traversing the design hierarchy at the top cell
- For the current cell:
  - Calculate and store intersection of all shapes with the boundary
  - Add all child instances to the queue
- Finally, write all shapes to the DEF file

Secondary Goals

- No modification of design data
- Short runtime (seconds to a few minutes at most)

A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Outline

- Introduction
- Approach
- **Implementation**
- Challenges
  - Automatic Shape Conversion
  - Design data consistency
- Conclusion

A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Implementation – Preparations

- Get the path between the cells

```
path = geFindPathToCellViewInst(topcell cell)
```

- Get the transformation lists

```
cell2top = geGetInstTransform(path)
top2cell = icDbInvertTransform(cell cell2top)
```

- Copy the boundary to the top cell

```
dbCopyFig(boundary topcell cell2top)
```

- Load routing layers from tech file

```
techGetViaLayers(techfile)
techGetPrRoutingLayers(techfile)
```


A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Implementation – Main Loop (1)

- Start at the top cell

```
queue = list(list(topcell list(0:0 "R0" 1.0))
```

- Remove first element from queue
- Process all shapes
  - For each layer: Copy shapes to the top cell

```
copy = dbCopyFig(shape topcell transformation)
```

  - Calculate and store the intersection with the boundary

```
shapes = dbLayerAnd(topcell … boundary copies)
```


A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Implementation – Main Loop (2)

- Process all instances (ignoring mosaics)
  - Transform bounding box to top cell coordinates

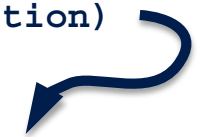
    ```
    dbTransformBBox(instance~>bBox transformation)
    ```

  - Calculate transformation list of all children

    ```
    dbConcatTransform(instance~>transform transformation)
    ```

  - If bounding box intersects:

    ```
    queue = cons(list(instance~>master newTransformation) queue)
    ```



A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Implementation – Main Loop (3)

- Process all mosaics
  - Copy mosaic to top cell

    ```
    copy = dbCopyFig(mosaic topcell transformation)
    ```

  - Flatten the copy

    ```
    dbFlattenInst(copy 1)
    ```

  - Add intersecting instances to the queue

    ```
    queue = cons(list(instance~>master instance~>transform) queue)
    ```

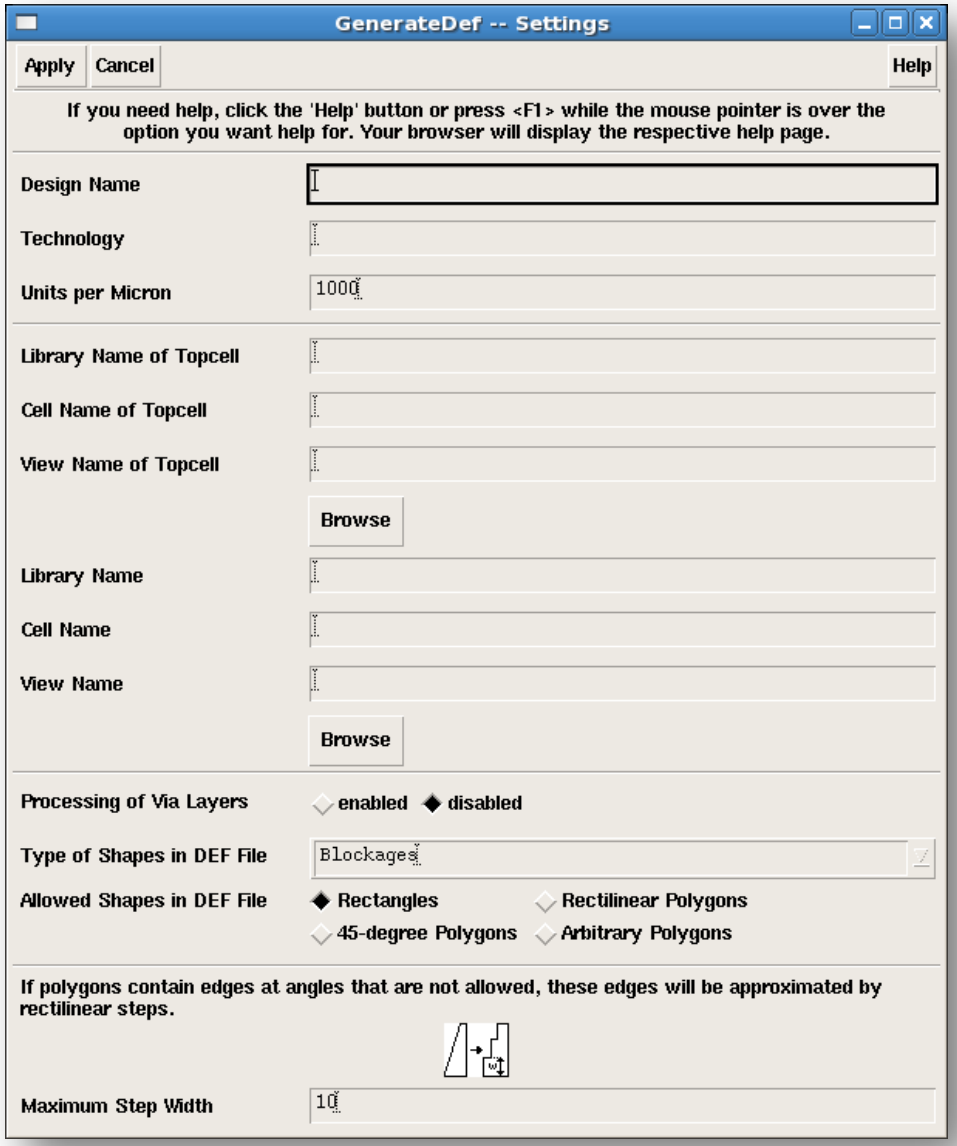
- Restart main loop: Process next instance in queue

A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Graphical User Interface

Higher-ranking cell →

Cell to extract →



Outline

- Introduction
- Approach
- Implementation
- **Challenges**
  - Automatic Shape Conversion
  - Design data consistency
- Conclusion

A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

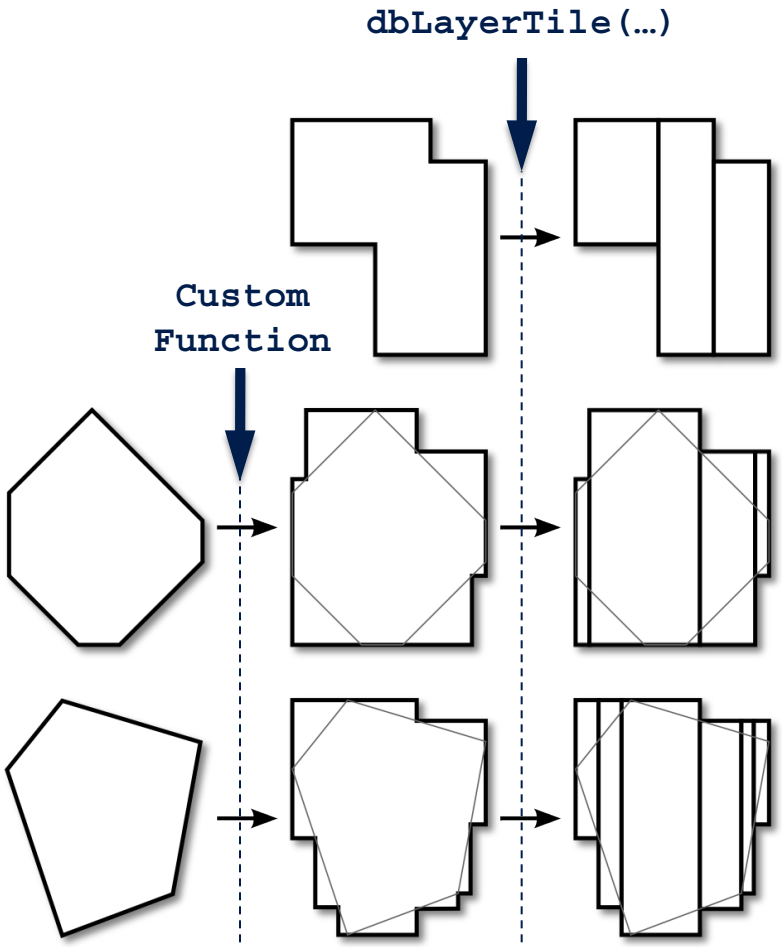
Challenges

- Problem: Multiple versions of the DEF standard exist
- EDA tools support different versions with varying properties
- Main difference: Supported shape types
  - Rectangles
  - Rectilinear polygons
  - Octilinear polygons
  - (Arbitrary polygons)
- Conversion of unsupported shape types is necessary

A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Shape Conversion

- Allowed shape types are configurable
- Automatic conversion of forbidden shape types
- Conversion to rectilinear polygons:
  - Iterating over all edges
  - If angle not allowed: Approximation with rectilinear steps
  - Maximum step width configurable



A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Design data consistency

- Goal: No modification of design data
- Top cell view opened in scratch mode:

```
cV = dbOpenCellViewByType(libName cellName viewName nil "s")
```

- Problem: Every window displaying this cell view records changes
- 1st Solution: Unroll all changes
- 2nd Solution:

```
dbClose(cV)
if(member(cV dbGetOpenCellViews()) then
    ; refreshing the cell view discards all edits
    dbRefreshCellView(cV)
) ; if
```


A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Outline

- Introduction
- Approach
- Implementation
- Challenges
  - Automatic Shape Conversion
  - Design data consistency
- **Conclusion**

A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

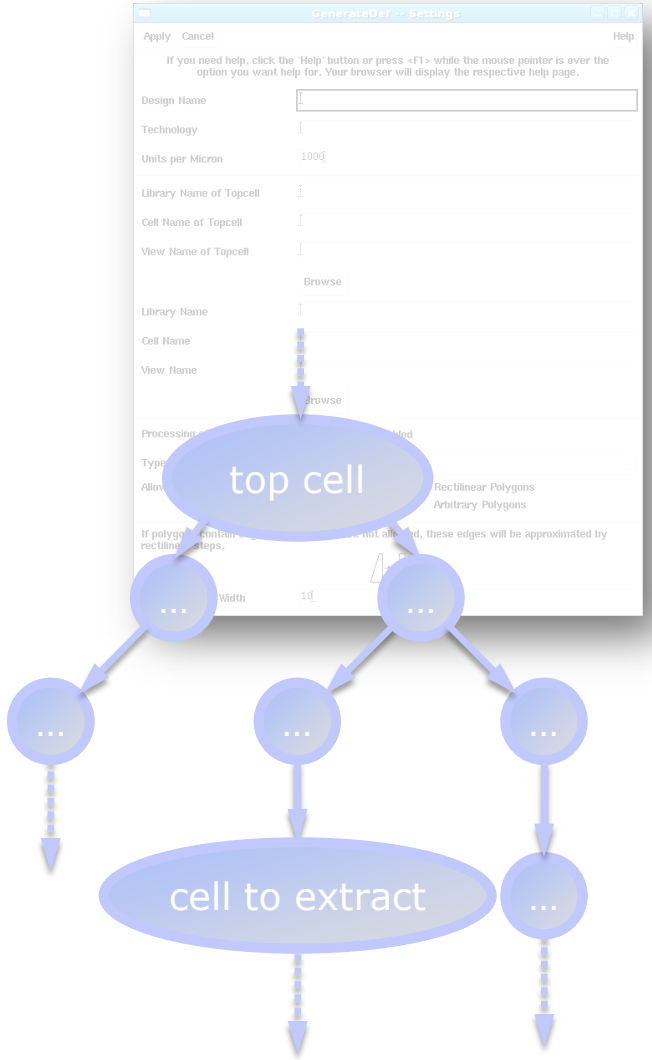
Conclusion

- Resulting DEF file contains *all* overlapping shapes
- All shapes are completely located within the bounds of the cell
- Automatic approximation of unsupported shapes
- No modification of design data
- Short runtime

- Shorter average design time (about 3 man days per design)
- Increase of design quality by avoiding potential DRC and LVS errors

A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany

Thank You!



A. Krinke, R. E. Stephan, "DEF File Export Considering Overlapping Structures", *CDNLive! 2011*, May 2011, Munich, Germany