Von einer VHDL-Beschreibung zum Layout

Eine Einführung in Silicon Ensemble

Projekt von
Marc-Sebastian Fiedler

Gliederung

1. Synthese der VHDL-Beschreibung
2. Vorbereitungen für Silicon Ensemble
3. Arbeiten mit Silicon Ensemble

VHDL-Beschreibung

- Angeben der Bibliotheken für die Synthese in der Toplevel-Beschreibung

```
-- synopsys translate_off
library umcl18u250t2_typ;
library umcl18u300t2_typ;
library umcl18u350t2_typ;
-- synopsys translate_on
```

- umcl18u250t2 für die Standardzelllogik
- umcl18u300t2 bzw. umcl18u350t2 für die Padzellen

VHDL-Beschreibung (2)

- Einfügen der Padzellen als Komponenten

```
…
component C3I40
  port(
    PAD : in    STD_LOGIC;
    DI  : out   STD_LOGIC);
end component;
…
bus: for i in 0 to 7 generate
     A_pad : C3I40 port map (
              summand_a(i), a(i));
end generate bus;
```

VHDL-Beschreibung (3)

- Welche Padzellen stehen zur Verfügung ?
- Pads aus der umcl18u300t2 Bibliothek:
  - C3I40 Input Pad
  - C3O10 Output Pad
  - C3B10 Bidirektionale Pad
  - C18C32 Clock Pad
  - VVSS Ground Pad
  - VVDD Power Pad
  - ANIO Analog Pad
- Pads aus der umcl18u350t2 Bibliothek haben immer noch ein „w“ zusätzlich vor dem Namen:
  - WC3I40 Input Pad

Synthese

- Synthese der VHDL-Beschreibung
- Durch Synopsys Design-Analyzer
  - Dazu wird am besten ein Verzeichnis z.B. synopsys für die Synthese angelegt
  - In diesem Verzeichnis sollte ein work Verzeichnis angelegt werden
  - die zwei folgenden Dateien werden benötigt, z.B. aus der VHDL-Übung:
    - .synopsys_dc.setup
    - .synopsys_vss.setup
  - In ihnen muss die Zielbibliothek für die UMC18-Zielarchitektur eingebunden werden

Synthese (2)

- .synopsys_dc.setup wie folgt anpassen:

```
link_library ={
    /opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/design_compiler/umcl18u250t2_typ.db, \
    /opt/des_kits/UMC/0.18/UMCL18U300D2_1.3/design_compiler/umcl18u300t2_typ.db, \
    /opt/des_kits/UMC/0.18/UMCL18U350D2_1.2/design_compiler/umcl18u350t2_typ.db}
target_library ={
    /opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/design_compiler/umcl18u250t2_typ.db, \
    /opt/des_kits/UMC/0.18/UMCL18U300D2_1.3/design_compiler/umcl18u300t2_typ.db, \
    /opt/des_kits/UMC/0.18/UMCL18U350D2_1.2/design_compiler/umcl18u350t2_typ.db}
symbol_library ={
    /opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/design_compiler/umcl18u250t2.sdb, \
    /opt/des_kits/UMC/0.18/UMCL18U300D2_1.3/design_compiler/umcl18u300t2.sdb, \
    /opt/des_kits/UMC/0.18/UMCL18U350D2_1.2/design_compiler/umcl18u350t2.sdb}
```

Synthese (3)

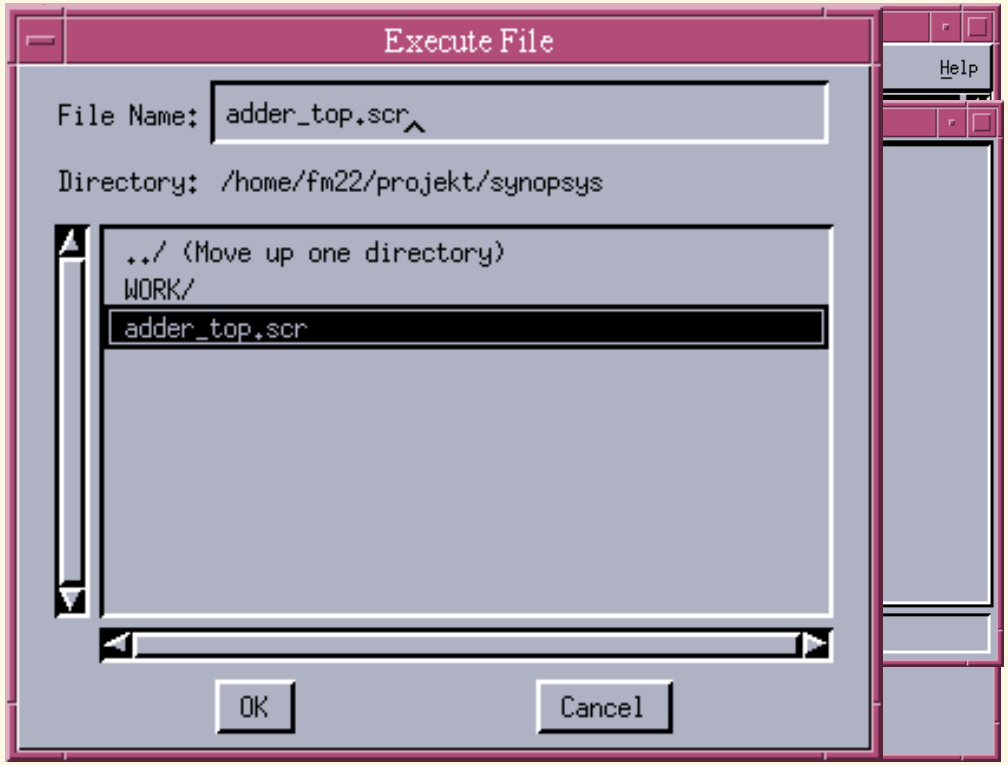
- In die .synopsys_vss.setup müssen folgende Pfade hinzugefügt werden:

umcl18u250t2_typ : /opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/design_compiler
umcl18u300t2_typ : /opt/des_kits/UMC/0.18/UMCL18U300D2_1.3/design_compiler
umcl18u350t2_typ : /opt/des_kits/UMC/0.18/UMCL18U350D2_1.2/design_compiler

Synthese (4)

- Starten des Design-Analyzers mit design_analyzer
- Setup -> Command Window öffnet das Kommandofenster
- Eingabe einzelner Kommandos
- Ausführen eines Skripts für die Synthese durch

  Setup -> Execute Script



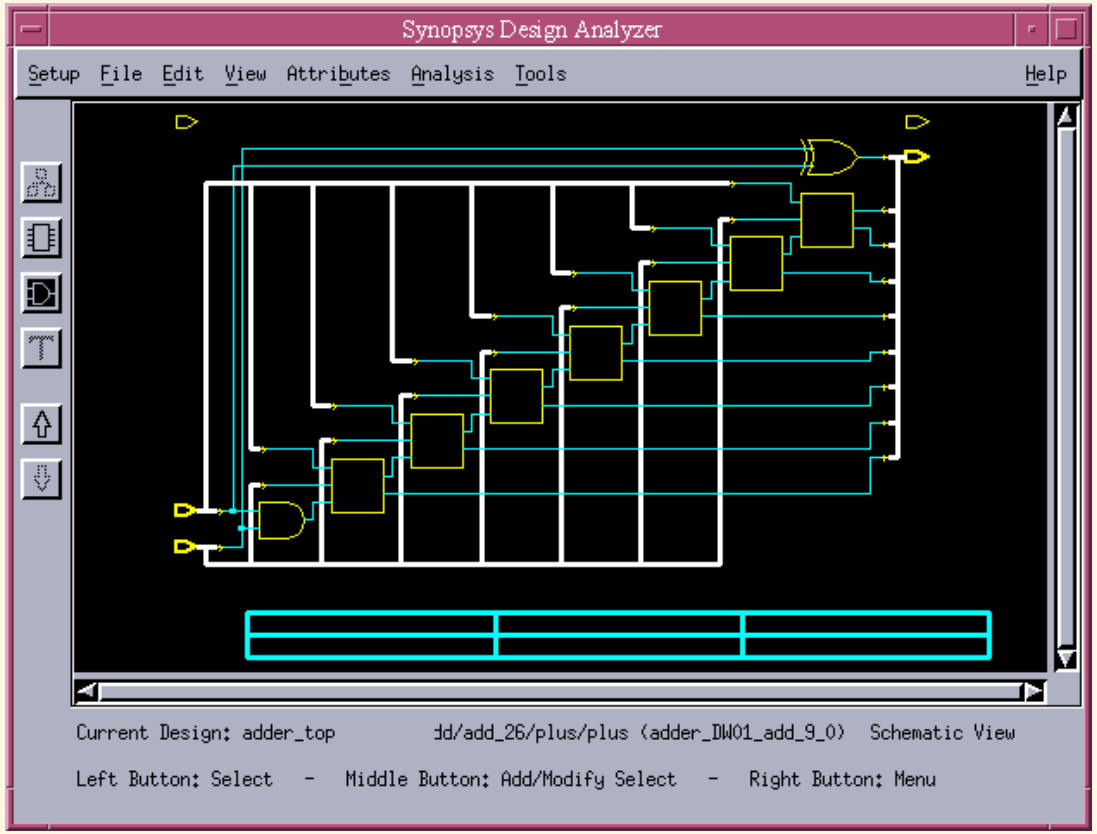
Synthese (5)

- Schreiben und ausführen eines Skripts:

```
TOP = adder_top
remove_design -all
analyze -format vhdl -lib work adder.vhd
analyze -format vhdl -lib work TOP + ".vhd“
elaborate TOP -arch TOP + "_arch" -lib WORK -update
current_design TOP
uniquify
set_port_is_pad "*“
remove_constraint -all
compile -map_effort med
change_names -rule verilog
write -format verilog -hierarchy -output TOP + ".v"
```

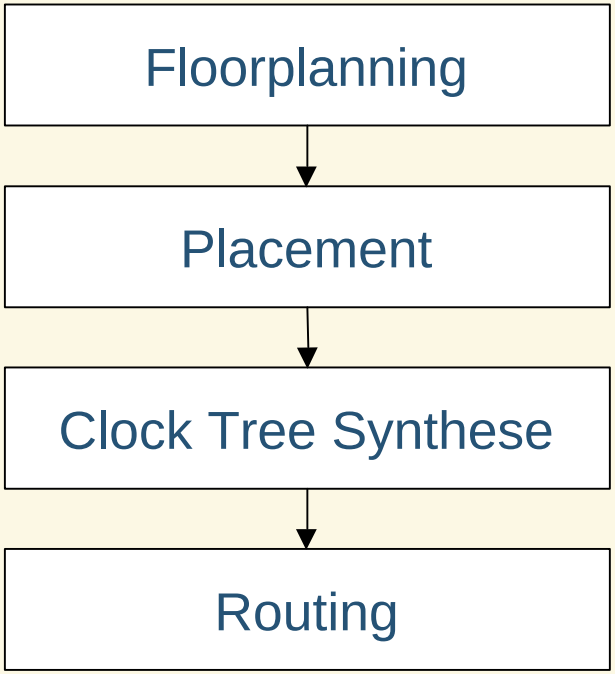
Synthese (6)

- Und man erhält eine Schaltung
- Und eine Verilognetzliste für das Erstellen eines Layouts z.B. in Silicon Ensemble

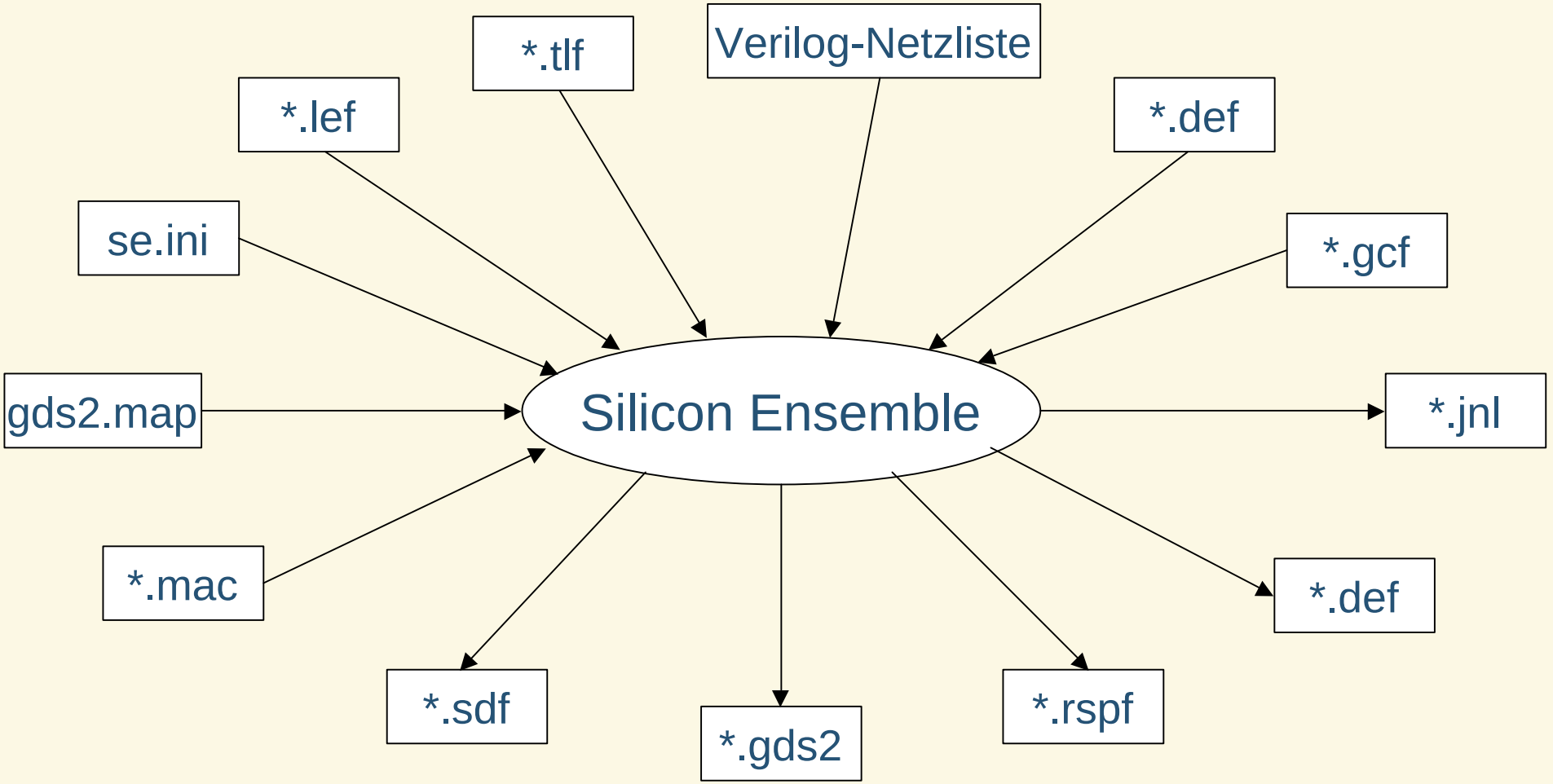


Silicon Ensemble

- Silicon Ensemble ist ein Layouttool für den Standardzellenentwurf
- Folgende Arbeitsschritte sind möglich:



Silicon Ensemble – Dateien



Netzliste

- Enthält alle Instanzen und deren Verbindungen zueinander
- meist in Verilogformat

LEF

- Library Exchange Format
- wird für alle Arten von Zellen benötigt
- ist technologieabhängig
- beinhaltet unter anderem:
  - die Größe der Zelle,
  - die existierenden Pins,
  - die Blockages,
  - die mögliche Zellausrichtungen
- beinhaltet aber keine Timinginformationen der Zellen

TLF

- Timing Library File
- Enthält die Timinginformationen der Zellen
- Für jede Zelle muss auch ein Timing-View vorhanden sein
- Verzögerungszeit wird durch eine Tabelle bestimmt
  - In Abhängigkeit von der Flankensteilheit des Eingangssignals und der Last am Ausgang
- Unterscheidung in drei Fälle
  - Worst
  - Typical
  - Best

TLF (2)

- Unterscheidung ergibt sich aus der Spannung, Temperatur und Prozessstreuung

	Worst-Case	Typical-Case	Best-Case
Spannung [V]	1,62	1,8	1,98
Temperatur [°C]	0	25	125
Prozessstreuung	0,8	1	1,2

- Wichtig für die Timinganalyse
  - Dort wird zwischen einem Setup- und Holdfehler unterschieden

DEF

- Design Exchange Format
- beinhaltet alle Informationen aus der Netzliste, sowie zusätzliche Informationen aus der Layout-Sicht, z. B.:
  - die Größe des Chips
  - die Koordinaten der platzierten Zellen
  - sowie deren Ausrichtung
  - Spezialnets
  - Routing des Designs

Vorbereitungen für SE

- Hinzufügen der Pfade in die .cshrc

source /opt/cadence/ic4.46/start.sh
setenv PATH /opt/cadence/dsmse5.3/tools/dsm/bin:
/opt/cadence/dsmse5.3/tools/bin:
/opt/cadence/dsmse5.3/tools/dfII/bin:$PATH

- Oder ihr kopiert euch die .cshrc von mir:
  /home/fm22/.csrhc

Vorbereitungen für SE (2)

- Anlegen der folgenden Verzeichnisstruktur
- /home/user/cadence_se/ work/ dbs/

  …/netlist/

  …/def/

  …/lef/
- Initialisierungsdateien se.ini und gds2.map aus /opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/silicon_ensemble ins …/cadence_se/work/ Verzeichnis kopieren

Vorbereitungen für SE (3)

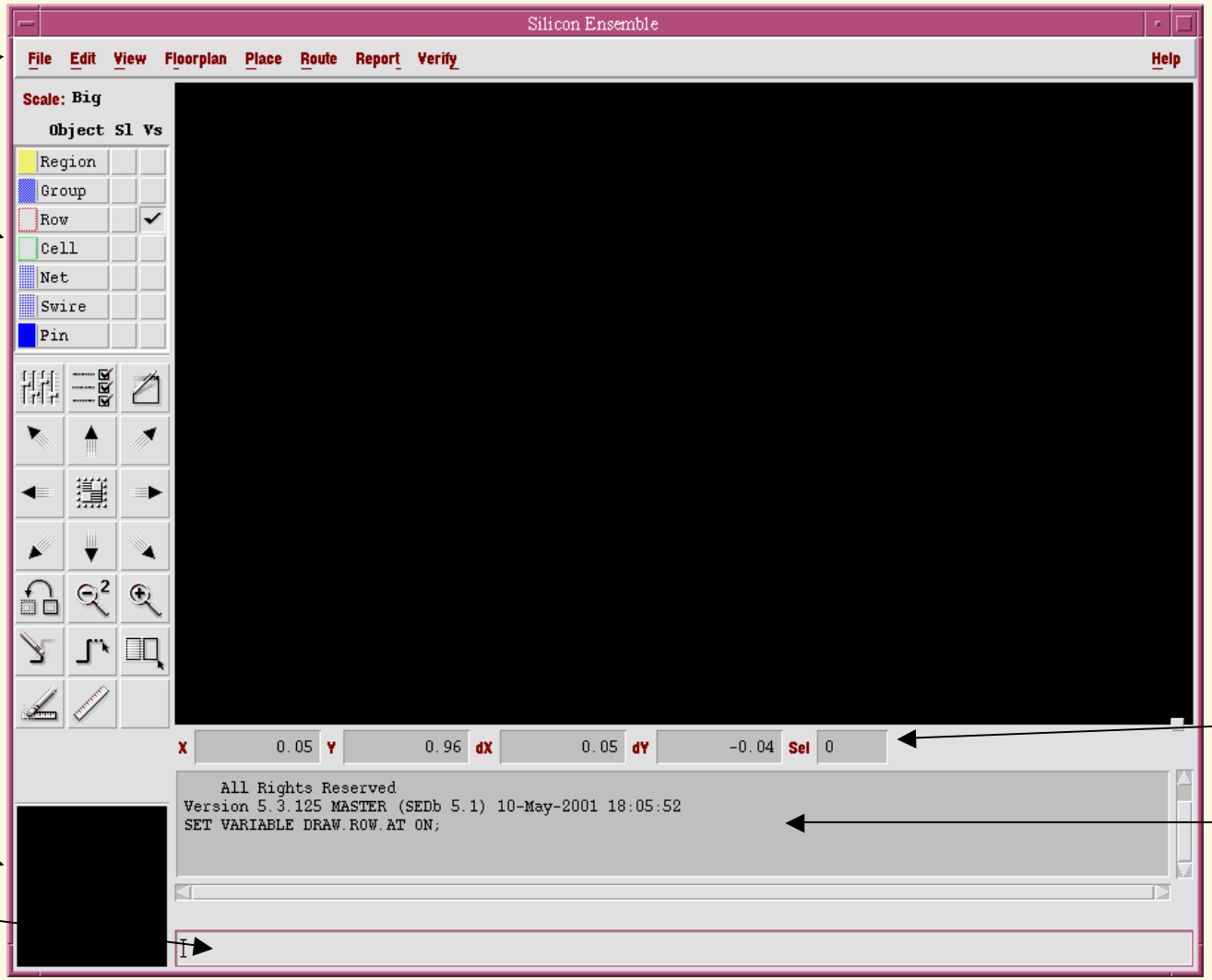
- Zum Importierten der TLF-Datei wird eine GCF-Datei (Gobal Constraint File) verwendet
  - Enthält einen Verweis auf die TLF-Dateien der Zellen
  - Kopieren der umcl18u250t2_typ_sample.gcf aus dem Pfad: /opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/tlf/
  - Anpassen der Pfade wie folgt:

```
(extension "TLF_FILES"
    ( /opt/des_kits/UMC/0.18/UMCL18U300D2_1.3/tlf/umcl18u300t2_bc_3.1.tlf
      /opt/des_kits/UMC/0.18/UMCL18U300D2_1.3/tlf/umcl18u300t2_typ_3.1.tlf
      /opt/des_kits/UMC/0.18/UMCL18U300D2_1.3/tlf/umcl18u300t2_wc_3.1.tlf
      /opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/tlf/umcl18u250t2_bc.tlf
      /opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/tlf/umcl18u250t2_typ.tlf
      /opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/tlf/umcl18u250t2_wc.tlf
    ) )
```

Starten von Silicon Ensemble

- Wechseln ins angelegte work-Verzeichnis
- Dort in die Unix-Shell sedsm zum starten eingeben
- sedsm –h zeigt unterschiedliche Startoptionen an
- sedsm –m=200 startet Silicon Ensemble mit einem Speicherbedarf von 200MB

Silicon Ensemble



Menü

Ansichtsfenster

Tools

Übersichtsfenster

Kommandozeile

Koordinaten

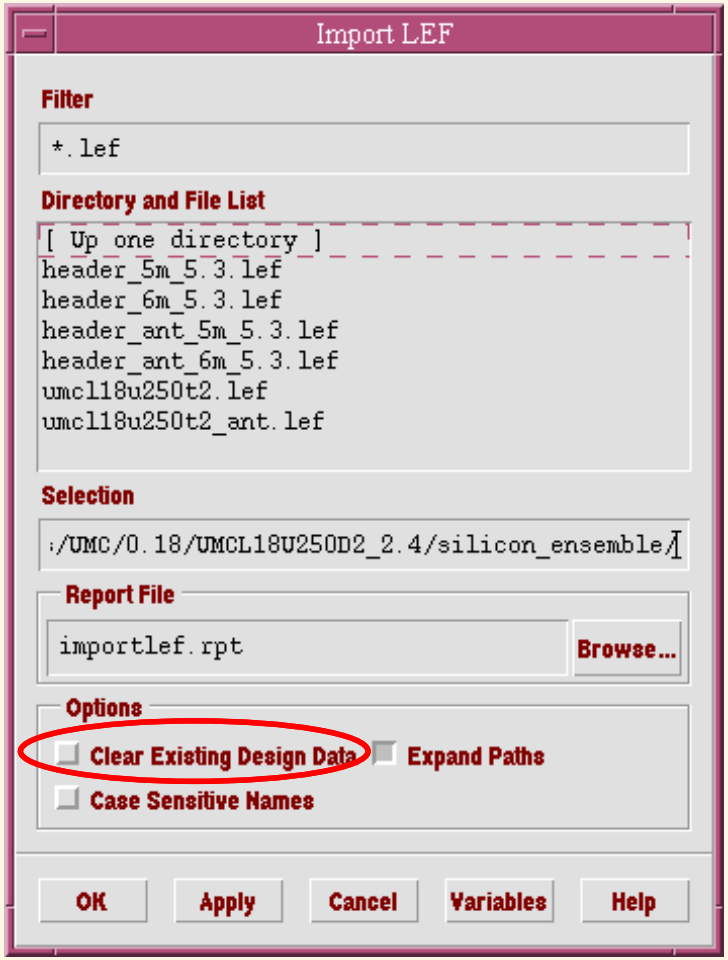
Kontrollausgabe

Importieren des Designs

- Reihenfolge des Importierens ist Wichtig
  1. LEF-Dateien für Standard- und Padzellen importieren
  2. TLF-Dateien
  3. Verilognetzliste

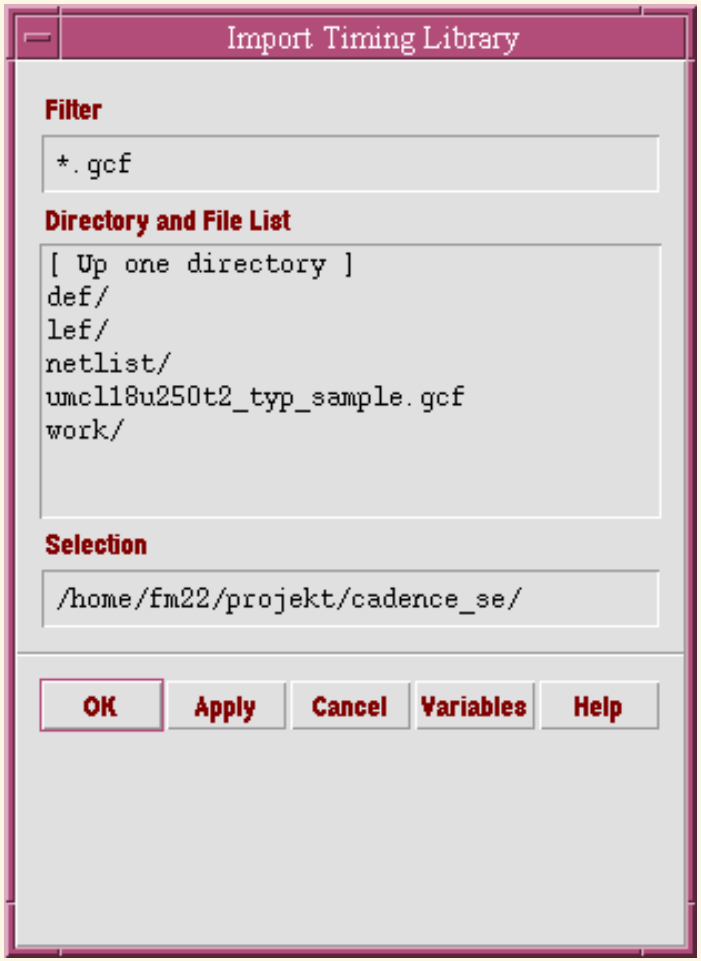
Importieren der LEF-Dateien

- File -> Import -> LEF
- In den folgenden Pfad wechseln:
/opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/silicon_ensemble/
- Clear Existing Design Data aktivieren
- header_6m_5.3.lef -> Apply
- Clear Existing Design Data deaktivieren
- umcl18u250t2.lef -> Apply
- in gleicher Weise ist das Importieren für die Padzellen durchzuführen



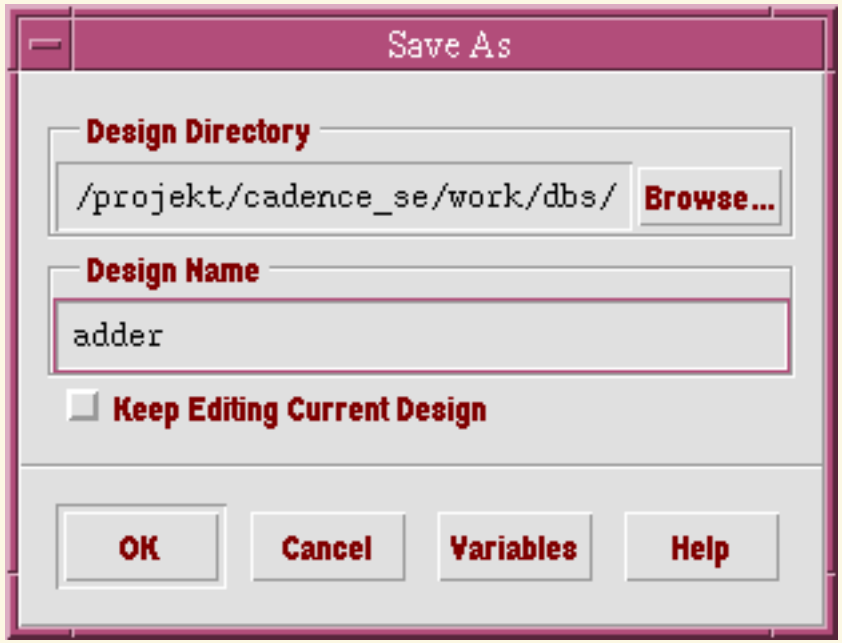
Importieren der TLF-Dateien

- File -> Import -> Timing Library
- In den …/cadence_se/ Pfad wechseln
- umcl18u250t2_typ_sample.gcf
- OK



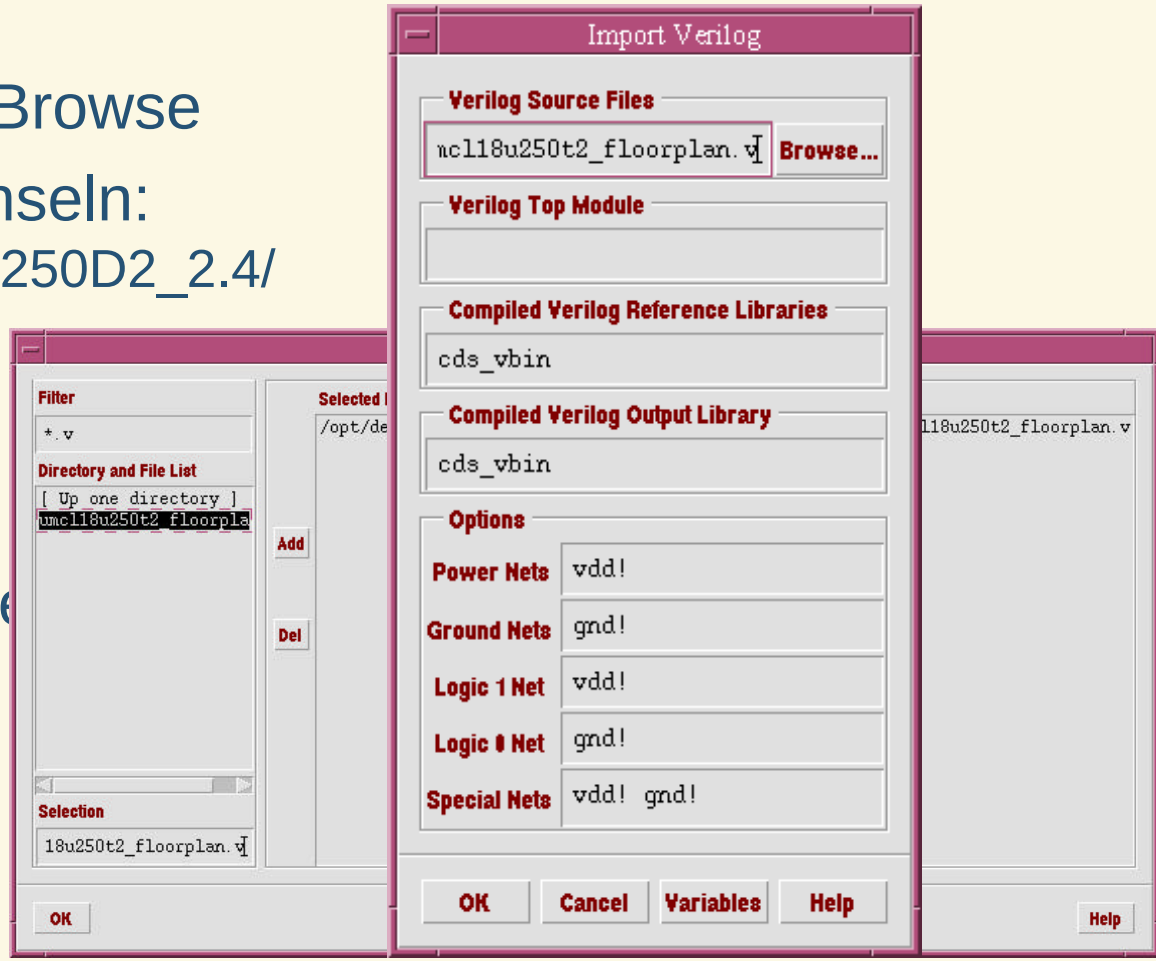
Design speichern

- File -> Save as
- standardmäßig speichert Cadence im angelegten dbs Verzeichnis
- Design Namen eingeben
- OK



Importieren der Verilognetzliste

- File -> Import -> Verilog -> Browse
- In den folgenden Pfad wechseln:
/opt/des_kits/UMC/0.18/UMCL18U250D2_2.4/silicon_ensemble/
- Datei umcl18u250t2_floorplan.v
- Add -> OK
- Verilog Top Module freilasse
- Evtl. Netze verändern
- OK



Importieren der Verilognetzliste (2)

- File -> Import -> Verilog -> Browse
- In den ../cadence_se/netlist/ Pfad wechseln
- Verilognetzliste auswählen
- Add -> OK
- im Verilog Top Module das TopEntity eintragen
- Evtl. Netze verändern
- OK

Alles importiert

- Damit sind alle notwendigen Designinformationen importiert
- Hier empfiehlt es sich das Design erneut zu speichern

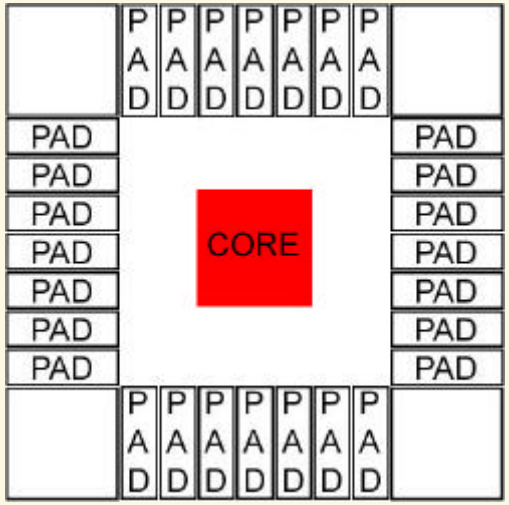
Floorplanning

- Festlegen der Chipgröße
- Erzeugung der Rows
- Platzierung der Pads
- Platzierung von Makroblöcken (Memory, PLL, DAC…)
- Powerverdrahtung legen

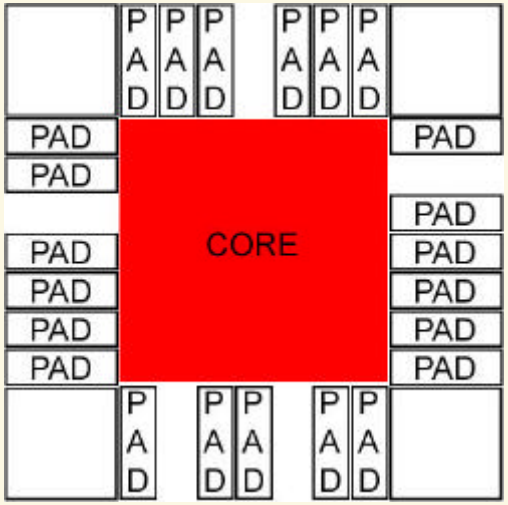
Floorplanning – Chipgröße

- Wodurch wird die Größe des Chips bestimmt ?

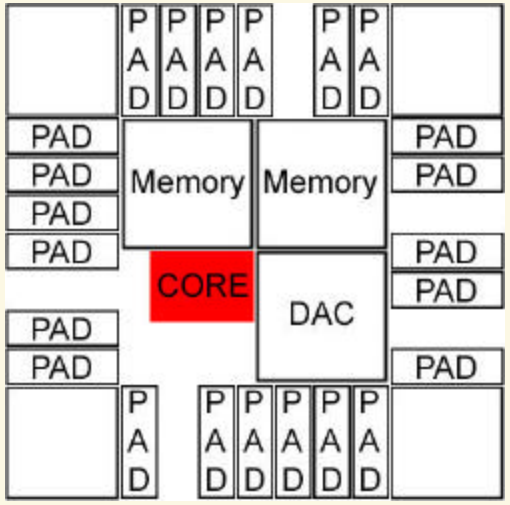
PAD-limitiert



CORE-limitiert



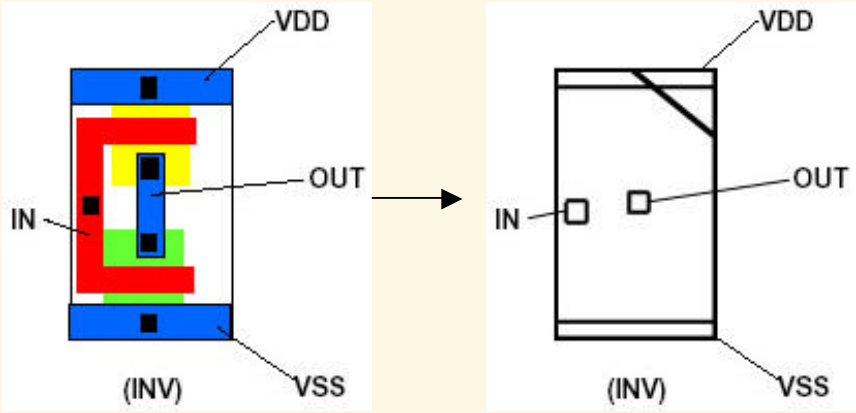
BLOCK-limitiert

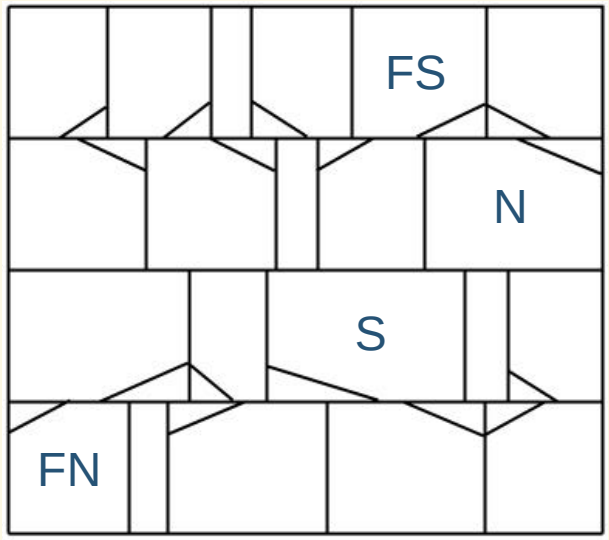


- Existieren unterschiedlich große Padzellen
  - umcl18u300t2 für PAD-limitiert
  - umcl18u350t2 für CORE-limitiert

Floorplanning – Rows

- Standardzellen
  - – konstante Höhe
  - – Breite abhängig von der Standardzelle
  - – wird „AbstractView“ verwendet, weil nicht alle Layoutinformationen benötigt werden
- Rows
  - – sind Zeilen in denen die Standardzellen platziert werden
  - – 4 Ausrichtungsmöglichkeiten der Standardzellen:
    - North, South, FlipNorth und FlipSouth





Floorplanning – Initialisierung

- Floorplan -> Initialize Floorplan
- Festlegung der Chipgröße
- Abstand zwischen Pads und Coregebiet
- Rows
  - Row Utilization ist das Verhältnis des Platzbedarfs der Std.zellen zum Gesamtrowgebiet in Prozent
  - Rows können direkt aneinander oder getrennt platziert werden
  - Rows können geflipt werden
- OK

Floorplanning – Pads

- Platzierung wird von Hand durchgeführt, d.h. die Koordinaten und die Ausrichtung der Zellen wird festgelegt
- Dazu kann eine DEF-Datei erzeugt werden, in der schon die Signalpads enthalten sind
- File -> Export -> DEF
- Cells aktivieren
- Dateinamen angeben
- OK

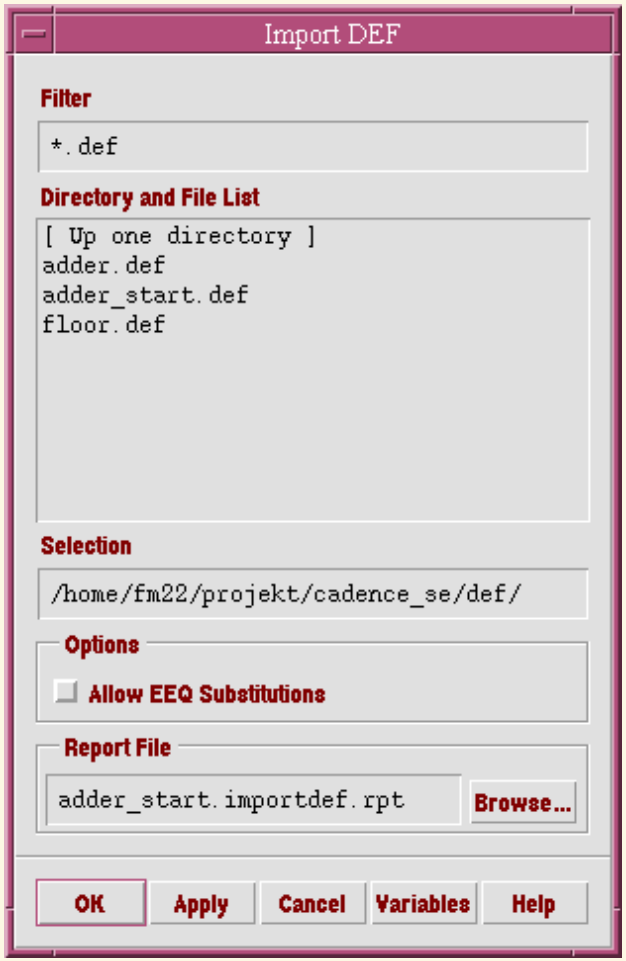
Floorplanning – Pads (2)

- Anpassen der DEF-Datei:

```
COMPONENTS 35 ;
…
- U100 CORNER + PLACED ( -444800 150100 ) E ;
- U101 CORNER + PLACED ( -444800 -424400 ) N ;
- U12 C3O10B + PLACED ( -102000 -424400 ) N ;
- U13 C3I80U + PLACED ( -60000 -424400 ) N ;
- U104 VVSS + PLACED ( 20000 150000 ) S ;
- U105 VVDD + PLACED ( 60000 150000 ) S ;
…
END COMPONENTS
```

Floorplanning – Pads (3)

- File -> Import -> DEF
- ins ../cadence_se/def/ Verzeichnis wechseln
- Dort die geschriebene DEF-Datei auswählen
- OK



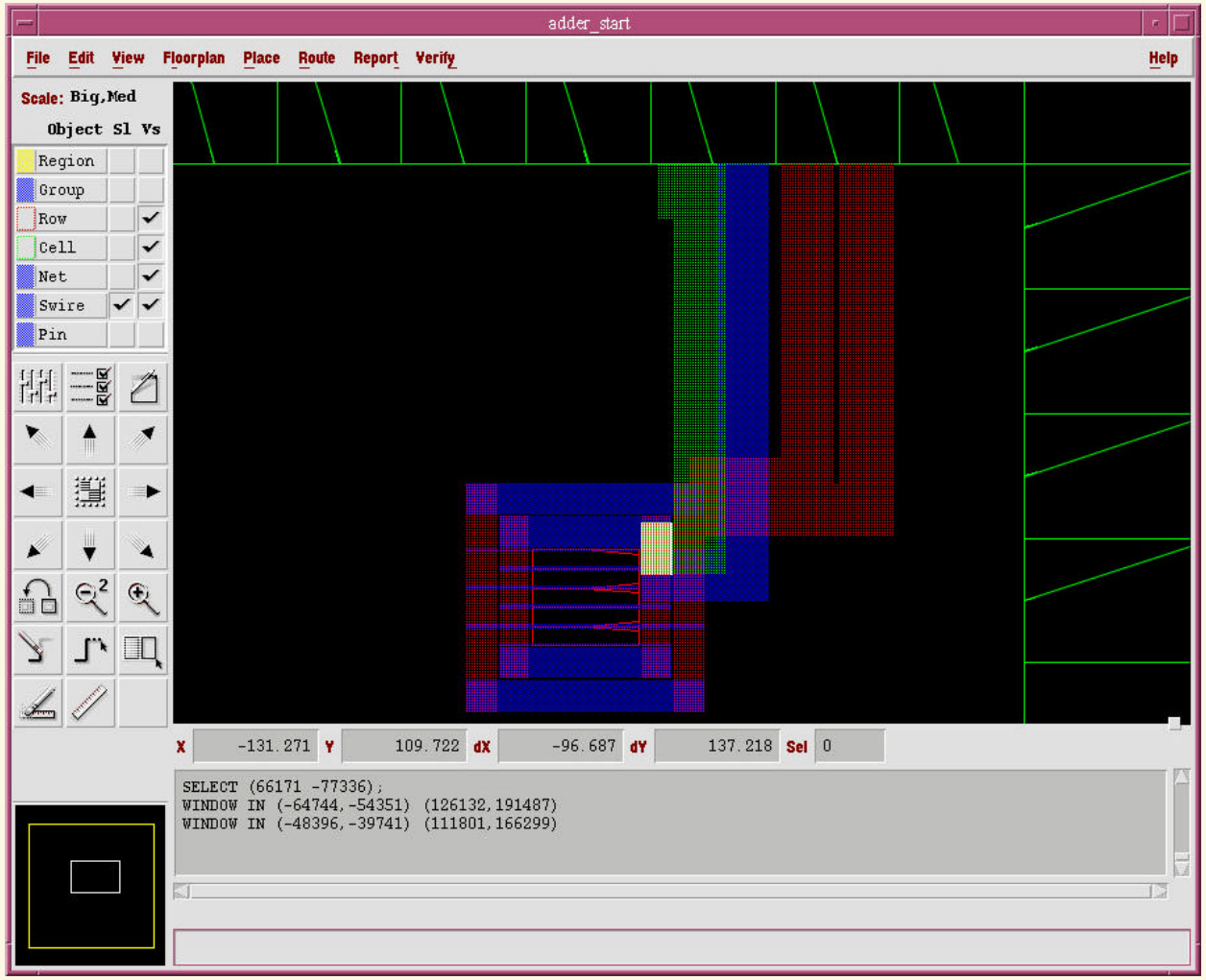
Floorplanning – Power

- Erzeugen von Powerringen
- Route -> Plan Power
- Add Rings
  - Ringe um das Coregebiet
  - Ringe um Blöcke
  - Namen der Netze
  - Breite der Leitungen
  - Position
- OK

Floorplanning – Power (2)

- Route -> Connect Ring
  - Stripe
  - Block
  - IO Pad
  - IO Ring
  - Followpinrouting
  - Auch hier müssen die Netznamen angeben werden
- OK

Floorplan

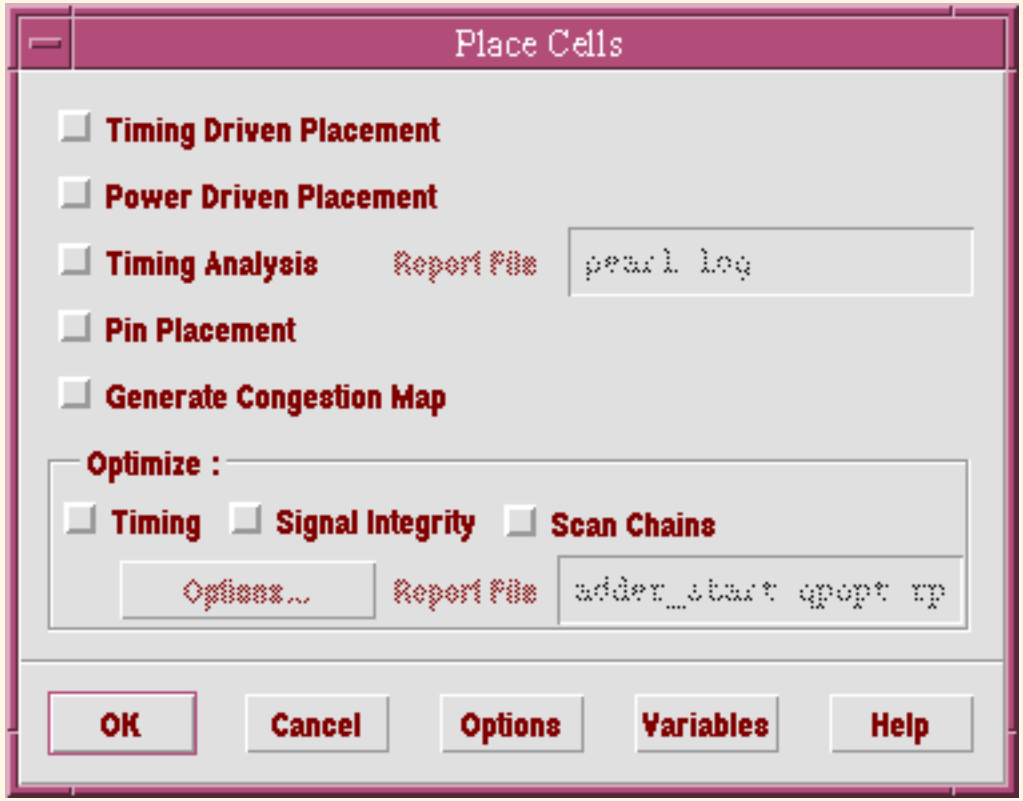

adder_start
File Edit View Floorplan Place Route Report Verify Help
Scale: Big,Med
Object Sl Vs
Region
Group
Row
Cell
Net
Swire
Pin
X -131.271 Y 109.722 dX -96.687 dY 137.218 Sel 0
SELECT (66171 -77336);
WINDOW IN (-64744,-54351) (126132,191487)
WINDOW IN (-48396,-39741) (111801,166299)

Placement

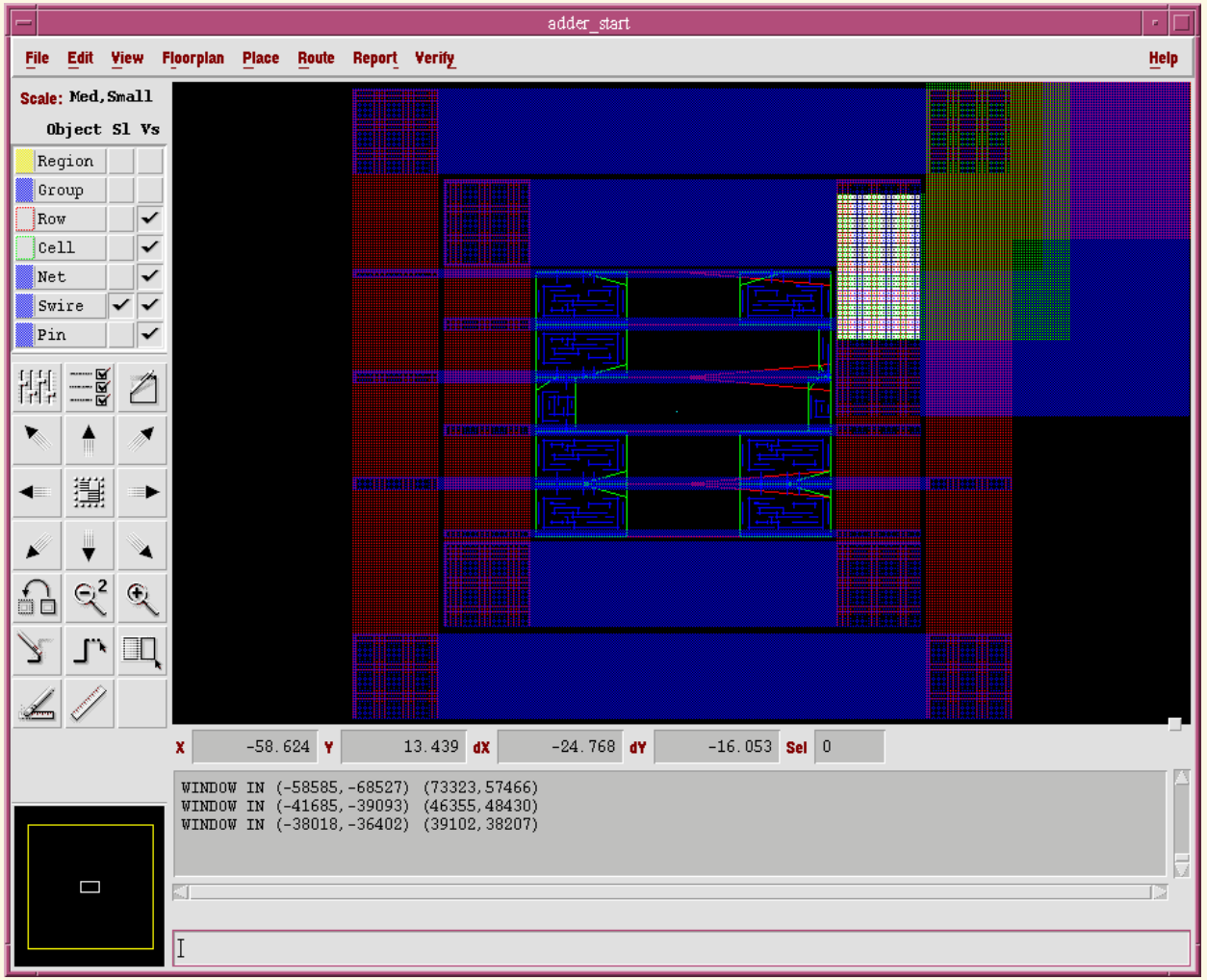
- Hierbei werden alle Standardzellen platziert, so dass
  - die Verbindungsleitungen möglichst kurz werden
  - das Design verdrahtbar ist
  - das vorgegebene Timing erreicht wird
- Dabei werden die Standardzellen in die Rows platziert
- Verschiedene Optionen
  - Timing Driven Placement
  - Congestion Map

Placement (2)

- Place –> Place Cells
- OK



Placement (3)


adder_start
File Edit View Floorplan Place Route Report Verify Help
Scale: Med,Small
Object Sl Vs
Region
Group
Row
Cell
Net
Swire
Pin
X -58.624 Y 13.439 dX -24.768 dY -16.053 Sel 0
WINDOW IN (-58585,-68527) (73323,57466)
WINDOW IN (-41685,-39093) (46355,48430)
WINDOW IN (-38018,-36402) (39102,38207)

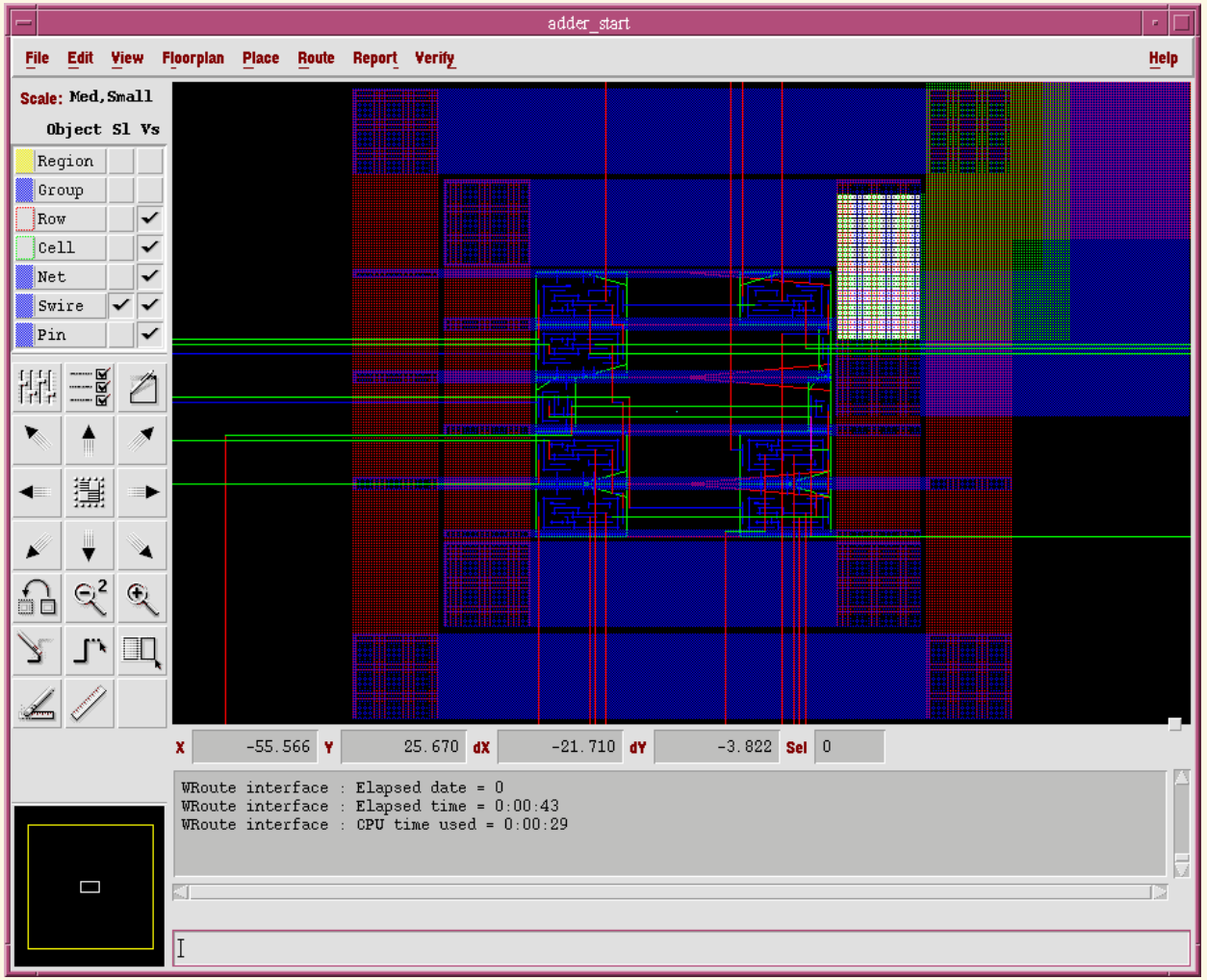
Routing

- Verdrahtung der Signalleitungen zwischen den Standardzellen
- In bis zu 6 Metalllayern
- WarpRoute
  - Gobal Routing
  - Final Routing

Routing (2)

- Route -> WRoute
- Global und Final Route aktivieren
- OK

Routing (3)


adder_start
File Edit View Floorplan Place Route Report Verify Help
Scale: Med,Small
Object Sl Vs
Region
Group
Row
Cell
Net
Swire
Pin
X -55.566 Y 25.670 dX -21.710 dY -3.822 Sel 0
WRoute interface : Elapsed date = 0
WRoute interface : Elapsed time = 0:00:43
WRoute interface : CPU time used = 0:00:29

Abschließende Schritte

- Timing Analyse
- Physical Verification
  - Design Rule Check
  - Layout vs. Schematic