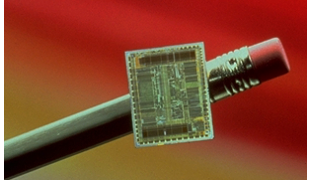


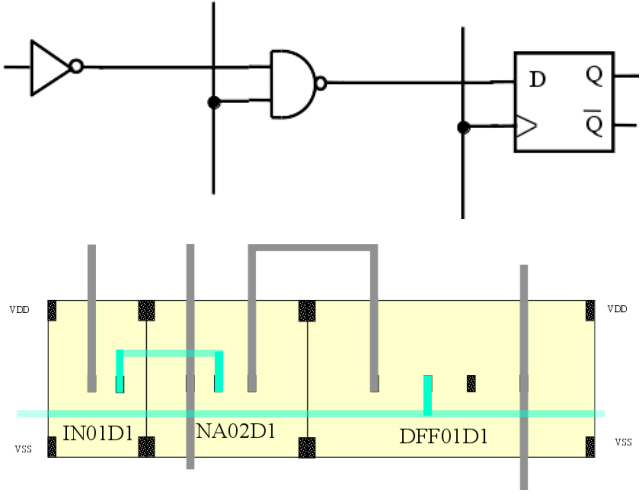
ICS904/CD2IC : Cell Design For Digital Integrated Circuits

L4 : Design Automation

Yves MATHIEU
yves.mathieu@telecom-paris.fr

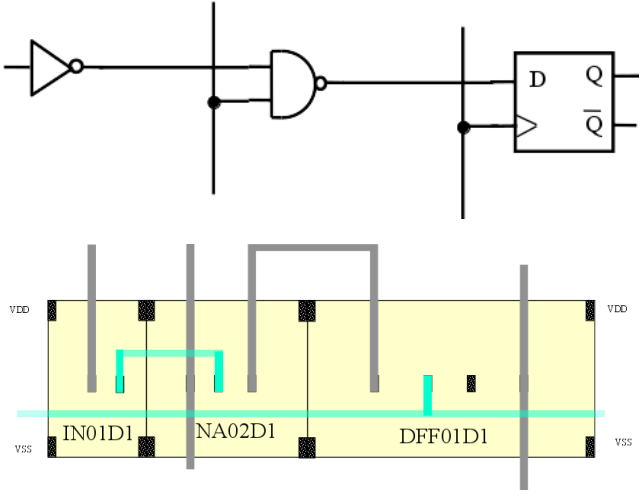
Outline

Refresher : Standard cell principles



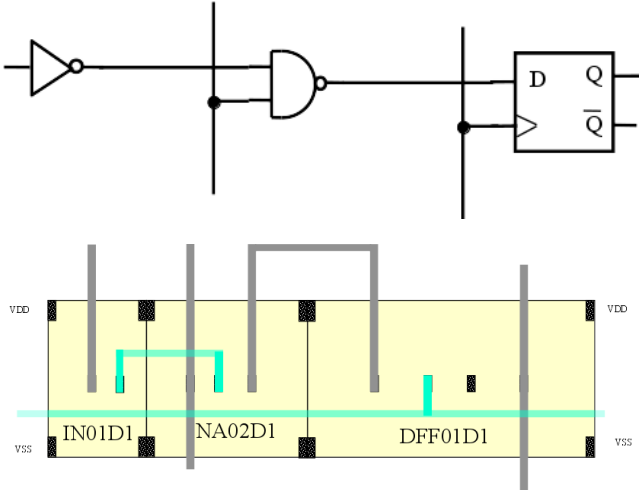
- All cells have same height
- Power supply and Ground connected by abutment.
- Cell design should be free of DRC error.
- Any abutment of any couple of cells should be free of DRC error.
- Wiring inside cell limited to Metal1 level.

Refresher : Standard cell principles



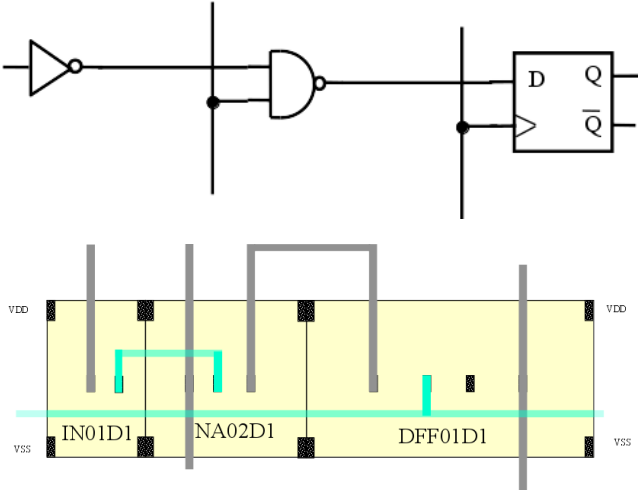
- All cells have same height
- Power supply and Ground connected by abutment.
- Cell design should be free of DRC error.
- Any abutment of any couple of cells should be free of DRC error.
- Wiring inside cell limited to Metal1 level.

Refresher : Standard cell principles



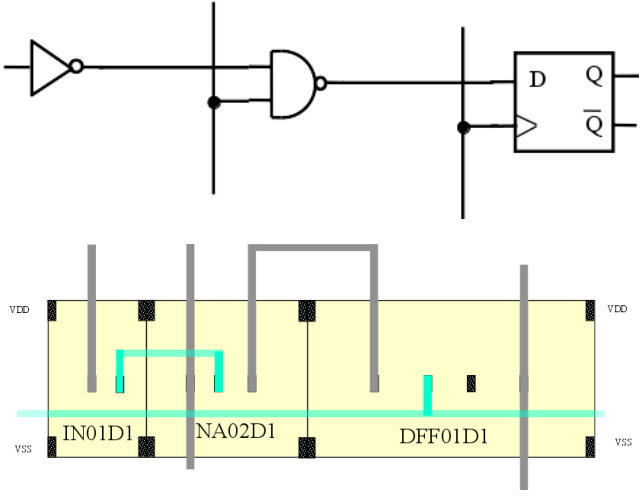
- All cells have same height
- Power supply and Ground connected by abutment.
- Cell design should be free of DRC error.
- Any abutment of any couple of cells should be free of DRC error.
- Wiring inside cell limited to Metal1 level.

Refresher : Standard cell principles



- All cells have same height
- Power supply and Ground connected by abutment.
- Cell design should be free of DRC error.
- Any abutment of any couple of cells should be free of DRC error.
- Wiring inside cell limited to Metal1 level.

Refresher : Standard cell principles



- All cells have same height
- Power supply and Ground connected by abutment.
- Cell design should be free of DRC error.
- Any abutment of any couple of cells should be free of DRC error.
- Wiring inside cell limited to Metal1 level.

Standard cell library

Example : gsclib045 library based on gpdk045 technology

Cell category	Cell list
ADD	Full/Half adders, 2 outputs
AND	up to inputs
AO	AO/AOI up to 6 inputs
BUF	Buffers
FF	DFF (EN,SET,RST,QB...)
INV	Invertor
MX	Multiplexor up to 4 inputs
NAND	up to 4 inputs
NOR	up to inputs
OA	OA/OAI up to 6 inputs
OR	up to 4 inputs
TLAT	Latches (SET,RST,...)
XNOR	up to 3 inputs
XOR	up to 3 inputs

- A wide selection of cells
- Facilitate synthesis tools usage
- Non obvious cells

Standard cell library

Example : gsclib045 library based on gpdk045 technology

Cell category	Cell list
ADD	Full/Half adders, 2 outputs
AND	up to inputs
AO	AO/AOI up to 6 inputs
BUF	Buffers
FF	DFF (EN,SET,RST,QB...)
INV	Invertor
MX	Multiplexor up to 4 inputs
NAND	up to 4 inputs
NOR	up to inputs
OA	OA/OAI up to 6 inputs
OR	up to 4 inputs
TLAT	Latches (SET,RST,...)
XNOR	up to 3 inputs
XOR	up to 3 inputs

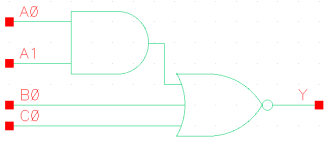
- A wide selection of cells
- Facilitate synthesis tools usage
- Non obvious cells

Standard cell library

Example : gsclib045 library based on gpdk045 technology

Cell category	Cell list
ADD	Full/Half adders, 2 outputs
AND	up to inputs
AO	AO/AOI up to 6 inputs
BUF	Buffers
FF	DFF (EN,SET,RST,QB...)
INV	Invertor
MX	Multiplexor up to 4 inputs
NAND	up to 4 inputs
NOR	up to inputs
OA	OA/OAI up to 6 inputs
OR	up to 4 inputs
TLAT	Latches (SET,RST,...)
XNOR	up to 3 inputs
XOR	up to 3 inputs

- A wide selection of cells
- Facilitate synthesis tools usage
- Non obvious cells



Standard cell library

Buffering

Cell category	Cell list
ADD	X1,X2,X4
AND	X1,X2,X4,X6,X8
AO	X2,X2,X4
BUF	X2,X3,. . .,X16,X20
FF	X1,X2,X4,X8
INV	X2,X3,. . .,X16,X20
MX	X1,X2,X4
NAND	X1,X2,X4,X6,X8
NOR	X1,X2,X4,X6,X8
OA	X1,X2,X4
OR	X1,X2,X4,X6,X8
TLAT	X1,X2,X4,X6,X8
XNOR	X1,X2,X4
XOR	X1,X2,X4

- For synthesis and P&R timing optimisation.
- No way to resize transistors
- Growing sizes of predefined output transistors
- Large choice for simple cells
- Small choice for complex cells
- Extra large choice for INV/BUF (load adaptation)

Standard cell library

Buffering

Cell category	Cell list
ADD	X1,X2,X4
AND	X1,X2,X4,X6,X8
AO	X2,X2,X4
BUF	X2,X3,. . .,X16,X20
FF	X1,X2,X4,X8
INV	X2,X3,. . .,X16,X20
MX	X1,X2,X4
NAND	X1,X2,X4,X6,X8
NOR	X1,X2,X4,X6,X8
OA	X1,X2,X4
OR	X1,X2,X4,X6,X8
TLAT	X1,X2,X4,X6,X8
XNOR	X1,X2,X4
XOR	X1,X2,X4

- For synthesis and P&R timing optimisation.
- No way to resize transistors
- Growing sizes of predefined output transistors
- Large choice for simple cells
- Small choice for complex cells
- Extra large choice for INV/BUF (load adaptation)

Standard cell library

Buffering

Cell category	Cell list
ADD	X1,X2,X4
AND	X1,X2,X4,X6,X8
AO	X2,X2,X4
BUF	X2,X3,. . .,X16,X20
FF	X1,X2,X4,X8
INV	X2,X3,. . .,X16,X20
MX	X1,X2,X4
NAND	X1,X2,X4,X6,X8
NOR	X1,X2,X4,X6,X8
OA	X1,X2,X4
OR	X1,X2,X4,X6,X8
TLAT	X1,X2,X4,X6,X8
XNOR	X1,X2,X4
XOR	X1,X2,X4

- For synthesis and P&R timing optimisation.
- No way to resize transistors
- Growing sizes of predefined output transistors
- Large choice for simple cells
- Small choice for complex cells
- Extra large choice for INV/BUF (load adaptation)

Standard cell library

Buffering

Cell category	Cell list
ADD	X1,X2,X4
AND	X1,X2,X4,X6,X8
AO	X2,X2,X4
BUF	X2,X3,. . .,X16,X20
FF	X1,X2,X4,X8
INV	X2,X3,. . .,X16,X20
MX	X1,X2,X4
NAND	X1,X2,X4,X6,X8
NOR	X1,X2,X4,X6,X8
OA	X1,X2,X4
OR	X1,X2,X4,X6,X8
TLAT	X1,X2,X4,X6,X8
XNOR	X1,X2,X4
XOR	X1,X2,X4

- For synthesis and P&R timing optimisation.
- No way to resize transistors
- Growing sizes of predefined output transistors
- Large choice for simple cells
- Small choice for complex cells
- Extra large choice for INV/BUF (load adaptation)

Standard cell library

Buffering

Cell category	Cell list
ADD	X1,X2,X4
AND	X1,X2,X4,X6,X8
AO	X2,X2,X4
BUF	X2,X3,. . .,X16,X20
FF	X1,X2,X4,X8
INV	X2,X3,. . .,X16,X20
MX	X1,X2,X4
NAND	X1,X2,X4,X6,X8
NOR	X1,X2,X4,X6,X8
OA	X1,X2,X4
OR	X1,X2,X4,X6,X8
TLAT	X1,X2,X4,X6,X8
XNOR	X1,X2,X4
XOR	X1,X2,X4

- For synthesis and P&R timing optimisation.
- No way to resize transistors
- Growing sizes of predefined output transistors
- Large choice for simple cells
- Small choice for complex cells
- Extra large choice for INV/BUF (load adaptation)

Standard cell library

Buffering

Cell category	Cell list
ADD	X1,X2,X4
AND	X1,X2,X4,X6,X8
AO	X2,X2,X4
BUF	X2,X3,. . .,X16,X20
FF	X1,X2,X4,X8
INV	X2,X3,. . .,X16,X20
MX	X1,X2,X4
NAND	X1,X2,X4,X6,X8
NOR	X1,X2,X4,X6,X8
OA	X1,X2,X4
OR	X1,X2,X4,X6,X8
TLAT	X1,X2,X4,X6,X8
XNOR	X1,X2,X4
XOR	X1,X2,X4

- For synthesis and P&R timing optimisation.
- No way to resize transistors
- Growing sizes of predefined output transistors
- Large choice for simple cells
- Small choice for complex cells
- Extra large choice for INV/BUF (load adaptation)

Standard cell library

Speed / Dynamic power / Leakage power trade-off

- Several versions of the library with the same cell layout.
- Based on the available transistor threshold choice.
- A "Standard VT" version for general purpose logic.
- A low leakage version using "High VT" transistors but slower gates...
- A high speed version using "Low VT" transistors but more leakage...
- Synthesizer strategy :
  - Use high speed gates, if needed, on critical paths.
  - Use low leakage gates on paths with relaxed timing constraints.
- "Backbias" versions used to dynamically adapt speed to external conditions.

Standard cell library

Speed / Dynamic power / Leakage power trade-off

- Several versions of the library with the same cell layout.
- Based on the available transistor threshold choice.
- A "Standard VT" version for general purpose logic.
- A low leakage version using "High VT" transistors but slower gates...
- A high speed version using "Low VT" transistors but more leakage...
- Synthesizer strategy :
  - Use high speed gates, if needed, on critical paths.
  - Use low leakage gates on paths with relaxed timing constraints.
- "Backbias" versions used to dynamically adapt speed to external conditions.

Standard cell library

Speed / Dynamic power / Leakage power trade-off

- Several versions of the library with the same cell layout.
- Based on the available transistor threshold choice.
- A "Standard VT" version for general purpose logic.
- A low leakage version using "High VT" transistors but slower gates...
- A high speed version using "Low VT" transistors but more leakage...
- Synthesizer strategy :
  - Use high speed gates, if needed, on critical paths.
  - Use low leakage gates on paths with relaxed timing constraints.
- "Backbias" versions used to dynamically adapt speed to external conditions.

Standard cell library

Speed / Dynamic power / Leakage power trade-off

- Several versions of the library with the same cell layout.
- Based on the available transistor threshold choice.
- A "Standard VT" version for general purpose logic.
- A low leakage version using "High VT" transistors but slower gates...
- A high speed version using "Low VT" transistors but more leakage...
- Synthesizer strategy :
  - Use high speed gates, if needed, on critical paths.
  - Use low leakage gates on paths with relaxed timing constraints.
- "Backbias" versions used to dynamically adapt speed to external conditions.

Standard cell library

Speed / Dynamic power / Leakage power trade-off

- Several versions of the library with the same cell layout.
- Based on the available transistor threshold choice.
- A "Standard VT" version for general purpose logic.
- A low leakage version using "High VT" transistors but slower gates...
- A high speed version using "Low VT" transistors but more leakage...
- Synthesizer strategy :
  - Use high speed gates, if needed, on critical paths.
  - Use low leakage gates on paths with relaxed timing constraints.
- "Backbias" versions used to dynamically adapt speed to external conditions.

Standard cell library

Speed / Dynamic power / Leakage power trade-off

- Several versions of the library with the same cell layout.
- Based on the available transistor threshold choice.
- A "Standard VT" version for general purpose logic.
- A low leakage version using "High VT" transistors but slower gates...
- A high speed version using "Low VT" transistors but more leakage...
- Synthesizer strategy :
  - Use high speed gates, if needed, on critical paths.
  - Use low leakage gates on paths with relaxed timing constraints.
- "Backbias" versions used to dynamically adapt speed to external conditions.

Standard cell library

Speed / Dynamic power / Leakage power trade-off

- Several versions of the library with the same cell layout.
- Based on the available transistor threshold choice.
- A "Standard VT" version for general purpose logic.
- A low leakage version using "High VT" transistors but slower gates...
- A high speed version using "Low VT" transistors but more leakage...
- Synthesizer strategy :
  - Use high speed gates, if needed, on critical paths.
  - Use low leakage gates on paths with relaxed timing constraints.
- "Backbias" versions used to dynamically adapt speed to external conditions.

Standard cell library

Special purpose cells : clock tree, delay cells, clock gating, LowPower Cells...

- During synthesis : clock signals are considered as ideal clocks, the designer defines expected clock properties (period, skew)
- Based on activity simulation, synthesis tool may insert **"clock gating cells"** in order to minimize power consumption
- During P&R phase, a clock tree is generated by the tools in order to full-fill the expected behavior. Specific **clock buffers**, with specific timing constraints are used (very short transition time)
- The P&R tool may insert **"delay cells"** in order to full-fill "hold timing" constraints (course between clk and data between D flip-flops
- In a "multi-domain power supply" designs, **"level adapter cells"** are inserted between cells of different power domains, **"power switch cells"** are inserted in order to switch on/off complete blocks.

Standard cell library

Special purpose cells : clock tree, delay cells, clock gating, LowPower Cells...

- During synthesis : clock signals are considered as ideal clocks, the designer defines expected clock properties (period, skew)
- Based on activity simulation, synthesis tool may insert **"clock gating cells"** in order to minimize power consumption
- During P&R phase, a clock tree is generated by the tools in order to full-fill the expected behavior. Specific **clock buffers**, with specific timing constraints are used (very short transition time)
- The P&R tool may insert **"delay cells"** in order to full-fill "hold timing" constraints (course between clk and data between D flip-flops
- In a "multi-domain power supply" designs, **"level adapter cells"** are inserted between cells of different power domains, **"power switch cells"** are inserted in order to switch on/off complete blocks.

Standard cell library

Special purpose cells : clock tree, delay cells, clock gating, LowPower Cells...

- During synthesis : clock signals are considered as ideal clocks, the designer defines expected clock properties (period, skew)
- Based on activity simulation, synthesis tool may insert **"clock gating cells"** in order to minimize power consumption
- During P&R phase, a clock tree is generated by the tools in order to full-fill the expected behavior. Specific **clock buffers**, with specific timing constraints are used (very short transition time)
- The P&R tool may insert **"delay cells"** in order to full-fill "hold timing" constraints (course between clk and data between D flip-flops
- In a "multi-domain power supply" designs, **"level adapter cells"** are inserted between cells of different power domains, **"power switch cells"** are inserted in order to switch on/off complete blocks.

Standard cell library

Special purpose cells : clock tree, delay cells, clock gating, LowPower Cells...

- During synthesis : clock signals are considered as ideal clocks, the designer defines expected clock properties (period, skew)
- Based on activity simulation, synthesis tool may insert **"clock gating cells"** in order to minimize power consumption
- During P&R phase, a clock tree is generated by the tools in order to full-fill the expected behavior. Specific **clock buffers**, with specific timing constraints are used (very short transition time)
- The P&R tool may insert **"delay cells"** in order to full-fill "hold timing" constraints (course between clk and data between D flip-flops
- In a "multi-domain power supply" designs, **"level adapter cells"** are inserted between cells of different power domains, **"power switch cells"** are inserted in order to switch on/off complete blocks.

Standard cell library

Special purpose cells : clock tree, delay cells, clock gating, LowPower Cells...

- During synthesis : clock signals are considered as ideal clocks, the designer defines expected clock properties (period, skew)
- Based on activity simulation, synthesis tool may insert **"clock gating cells"** in order to minimize power consumption
- During P&R phase, a clock tree is generated by the tools in order to full-fill the expected behavior. Specific **clock buffers**, with specific timing constraints are used (very short transition time)
- The P&R tool may insert **"delay cells"** in order to full-fill "hold timing" constraints (course between clk and data between D flip-flops
- In a "multi-domain power supply" designs, **"level adapter cells"** are inserted between cells of different power domains, **"power switch cells"** are inserted in order to switch on/off complete blocks.

Standard cell library

Special purpose cells : physical only cells...

- Physical only cells are cells that have no logic behavior. They are only needed by physical constraints of the layout
- **Filler cells** are used to fill holes between standard cells. This ensures, supply and well continuity.
- **Well tap cells** are used to connect Nwell and Pwell to the power supply and the ground.
- **Antenna cells** are used to limit the so called **Antenna Effect** during manufacturing
- **Decap cells** are used to limit ground and power bounce often called **IRdrop**.
- ...

Standard cell library

Special purpose cells : physical only cells...

- Physical only cells are cells that have no logic behavior. They are only needed by physical constraints of the layout
- **Filler cells** are used to fill holes between standard cells. This ensures, supply and well continuity.
- **Well tap cells** are used to connect Nwell and Pwell to the power supply and the ground.
- **Antenna cells** are used to limit the so called **Antenna Effect** during manufacturing
- **Decap cells** are used to limit ground and power bounce often called **IRdrop**.
- ...

Standard cell library

Special purpose cells : physical only cells...

- Physical only cells are cells that have no logic behavior. They are only needed by physical constraints of the layout
- **Filler cells** are used to fill holes between standard cells. This ensures, supply and well continuity.
- **Well tap cells** are used to connect Nwell and Pwell to the power supply and the ground.
- **Antenna cells** are used to limit the so called **Antenna Effect** during manufacturing
- **Decap cells** are used to limit ground and power bounce often called **IRdrop**.
- ...

Standard cell library

Special purpose cells : physical only cells...

- Physical only cells are cells that have no logic behavior. They are only needed by physical constraints of the layout
- **Filler cells** are used to fill holes between standard cells. This ensures, supply and well continuity.
- **Well tap cells** are used to connect Nwell and Pwell to the power supply and the ground.
- **Antenna cells** are used to limit the so called **Antenna Effect** during manufacturing
- **Decap cells** are used to limit ground and power bounce often called **IRdrop**.
- ...

Standard cell library

Special purpose cells : physical only cells...

- Physical only cells are cells that have no logic behavior. They are only needed by physical constraints of the layout
- **Filler cells** are used to fill holes between standard cells. This ensures, supply and well continuity.
- **Well tap cells** are used to connect Nwell and Pwell to the power supply and the ground.
- **Antenna cells** are used to limit the so called **Antenna Effect** during manufacturing
- **Decap cells** are used to limit ground and power bounce often called **IRdrop**.
- ...

Standard cell library

Special purpose cells : physical only cells...

- Physical only cells are cells that have no logic behavior. They are only needed by physical constraints of the layout
- **Filler cells** are used to fill holes between standard cells. This ensures, supply and well continuity.
- **Well tap cells** are used to connect Nwell and Pwell to the power supply and the ground.
- **Antenna cells** are used to limit the so called **Antenna Effect** during manufacturing
- **Decap cells** are used to limit ground and power bounce often called **IRdrop**.
- ...

Standard cell library

Refresh on Antenna Effect...

- During manufacturing phases, plasma etching leads to charge accumulation in conductor layers already created.
- If this layers are floating (not connected to drain/sources of transistors or wells) and connected to gates then gate oxide may break !!
- Place and route tools are able to evaluate and correct this problem :
- Evaluation : estimate the risk of breakdown, based on areas of metal/gates/implants connected to the conductor.
- Correction1 : Add and connect specific diodes (**Antenna cells**) in order to avoid floating layers during manufacturing
- Correction2 : Modify routing in order to connect long lines of metal only during the last steps of manufacturing.

Standard cell library

Refresh on Antenna Effect...

- During manufacturing phases, plasma etching leads to charge accumulation in conductor layers already created.
- If this layers are floating (not connected to drain/sources of transistors or wells) and connected to gates then gate oxide may break !!
- Place and route tools are able to evaluate and correct this problem :
- Evaluation : estimate the risk of breakdown, based on areas of metal/gates/implants connected to the conductor.
- Correction1 : Add and connect specific diodes (**Antenna cells**) in order to avoid floating layers during manufacturing
- Correction2 : Modify routing in order to connect long lines of metal only during the last steps of manufacturing.

Standard cell library

Refresh on Antenna Effect...

- During manufacturing phases, plasma etching leads to charge accumulation in conductor layers already created.
- If this layers are floating (not connected to drain/sources of transistors or wells) and connected to gates then gate oxide may break !!
- Place and route tools are able to evaluate and correct this problem :
- Evaluation : estimate the risk of breakdown, based on areas of metal/gates/implants connected to the conductor.
- Correction1 : Add and connect specific diodes (**Antenna cells**) in order to avoid floating layers during manufacturing
- Correction2 : Modify routing in order to connect long lines of metal only during the last steps of manufacturing.

Standard cell library

Refresh on Antenna Effect...

- During manufacturing phases, plasma etching leads to charge accumulation in conductor layers already created.
- If this layers are floating (not connected to drain/sources of transistors or wells) and connected to gates then gate oxide may break !!
- Place and route tools are able to evaluate and correct this problem :
- Evaluation : estimate the risk of breakdown, based on areas of metal/gates/implants connected to the conductor.
- Correction1 : Add and connect specific diodes (**Antenna cells**) in order to avoid floating layers during manufacturing
- Correction2 : Modify routing in order to connect long lines of metal only during the last steps of manufacturing.

Standard cell library

Refresh on Antenna Effect...

- During manufacturing phases, plasma etching leads to charge accumulation in conductor layers already created.
- If this layers are floating (not connected to drain/sources of transistors or wells) and connected to gates then gate oxide may break !!
- Place and route tools are able to evaluate and correct this problem :
- Evaluation : estimate the risk of breakdown, based on areas of metal/gates/implants connected to the conductor.
- Correction1 : Add and connect specific diodes (**Antenna cells**) in order to avoid floating layers during manufacturing
- Correction2 : Modify routing in order to connect long lines of metal only during the last steps of manufacturing.

Standard cell library

Refresh on Antenna Effect...

- During manufacturing phases, plasma etching leads to charge accumulation in conductor layers already created.
- If this layers are floating (not connected to drain/sources of transistors or wells) and connected to gates then gate oxide may break !!
- Place and route tools are able to evaluate and correct this problem :
- Evaluation : estimate the risk of breakdown, based on areas of metal/gates/implants connected to the conductor.
- Correction1 : Add and connect specific diodes (**Antenna cells**) in order to avoid floating layers during manufacturing
- Correction2 : Modify routing in order to connect long lines of metal only during the last steps of manufacturing.

Outline

The goal of test

- Filtering out defective devices during manufacturing.
- The more you wait to detect defaults, the more it costs.
- Test done at wafer level and after packaging.
- **Specification oriented test** : check the conformance to design specification.
- **Application oriented test** : check that the device works properly in its application environment.
- **Structural test** : check that there is no physical defect in the chip.
- **Structural test** is the more efficient et more easy to implement with generic methods.

The goal of test

- Filtering out defective devices during manufacturing.
- The more you wait to detect defaults, the more it costs.
- Test done at wafer level and after packaging.
- **Specification oriented test** : check the conformance to design specification.
- **Application oriented test** : check that the device works properly in its application environment.
- **Structural test** : check that there is no physical defect in the chip.
- **Structural test** is the more efficient et more easy to implement with generic methods.

The goal of test

- Filtering out defective devices during manufacturing.
- The more you wait to detect defaults, the more it costs.
- Test done at wafer level and after packaging.
- **Specification oriented test** : check the conformance to design specification.
- **Application oriented test** : check that the device works properly in its application environment.
- **Structural test** : check that there is no physical defect in the chip.
- **Structural test** is the more efficient et more easy to implement with generic methods.

The goal of test

- Filtering out defective devices during manufacturing.
- The more you wait to detect defaults, the more it costs.
- Test done at wafer level and after packaging.
- **Specification oriented test** : check the conformance to design specification.
- **Application oriented test** : check that the device works properly in its application environment.
- **Structural test** : check that there is no physical defect in the chip.
- **Structural test** is the more efficient et more easy to implement with generic methods.

The goal of test

- Filtering out defective devices during manufacturing.
- The more you wait to detect defaults, the more it costs.
- Test done at wafer level and after packaging.
- **Specification oriented test** : check the conformance to design specification.
- **Application oriented test** : check that the device works properly in its application environment.
- **Structural test** : check that there is no physical defect in the chip.
- **Structural test** is the more efficient et more easy to implement with generic methods.

The goal of test

- Filtering out defective devices during manufacturing.
- The more you wait to detect defaults, the more it costs.
- Test done at wafer level and after packaging.
- **Specification oriented test** : check the conformance to design specification.
- **Application oriented test** : check that the device works properly in its application environment.
- **Structural test** : check that there is no physical defect in the chip.
- **Structural test** is the more efficient et more easy to implement with generic methods.

The goal of test

- Filtering out defective devices during manufacturing.
- The more you wait to detect defaults, the more it costs.
- Test done at wafer level and after packaging.
- **Specification oriented test** : check the conformance to design specification.
- **Application oriented test** : check that the device works properly in its application environment.
- **Structural test** : check that there is no physical defect in the chip.
- **Structural test** is the more efficient et more easy to implement with generic methods.

Structural test

What kind of defects ?

- Hard shorts, Hard open.
- Resistive bridges, Resitive shorts.
- Wiring defects, Component (transistors) defects.
- A **fault** is an undesired behaviour of a chip as a result of a **defect**
- **fault models** should :
  - Accurately reflect the effect of a defect.
  - Represent defects that are typical for the technology used.
  - Be easy to implement in tools.

Structural test

What kind of defects ?

- Hard shorts, Hard open.
- Resistive bridges, Resitive shorts.
- Wiring defects, Component (transistors) defects.
- A **fault** is an undesired behaviour of a chip as a result of a **defect**
- **fault models** should :
  - Accurately reflect the effect of a defect.
  - Represent defects that are typical for the technology used.
  - Be easy to implement in tools.

Structural test

What kind of defects ?

- Hard shorts, Hard open.
- Resistive bridges, Resitive shorts.
- Wiring defects, Component (transistors) defects.
- A **fault** is an undesired behaviour of a chip as a result of a **defect**
- **fault models** should :
  - Accurately reflect the effect of a defect.
  - Represent defects that are typical for the technology used.
  - Be easy to implement in tools.

Structural test

What kind of defects ?

- Hard shorts, Hard open.
- Resistive bridges, Resitive shorts.
- Wiring defects, Component (transistors) defects.
- A **fault** is an undesired behaviour of a chip as a result of a **defect**
- **fault models** should :
  - Accurately reflect the effect of a defect.
  - Represent defects that are typical for the technology used.
  - Be easy to implement in tools.

Structural test

What kind of defects ?

- Hard shorts, Hard open.
- Resistive bridges, Resitive shorts.
- Wiring defects, Component (transistors) defects.
- A **fault** is an undesired behaviour of a chip as a result of a **defect**
- **fault models** should :
  - Accurately reflect the effect of a defect.
  - Represent defects that are typical for the technology used.
  - Be easy to implement in tools.

Structural test

What kind of defects ?

- Hard shorts, Hard open.
- Resistive bridges, Resitive shorts.
- Wiring defects, Component (transistors) defects.
- A **fault** is an undesired behaviour of a chip as a result of a **defect**
- **fault models** should :
  - Accurately reflect the effect of a defect.
  - Represent defects that are typical for the technology used.
  - Be easy to implement in tools.

Structural test

What kind of defects ?

- Hard shorts, Hard open.
- Resistive bridges, Resitive shorts.
- Wiring defects, Component (transistors) defects.
- A **fault** is an undesired behaviour of a chip as a result of a **defect**
- **fault models** should :
  - Accurately reflect the effect of a defect.
  - Represent defects that are typical for the technology used.
  - Be easy to implement in tools.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs

- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs

- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs

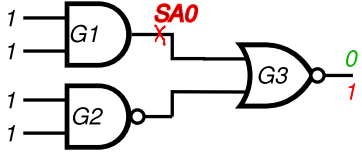
- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs



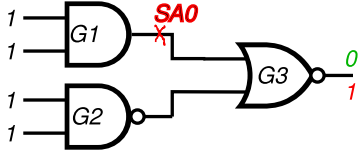
- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs



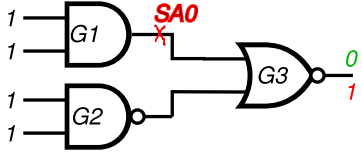
- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs



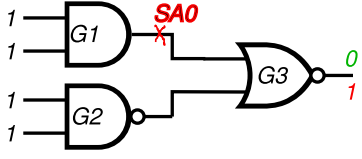
- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs



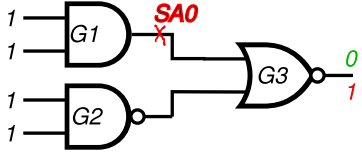
- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs



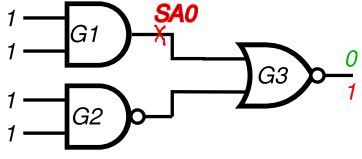
- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs



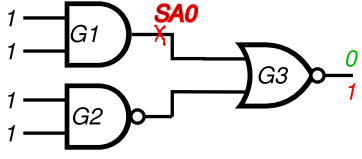
- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs



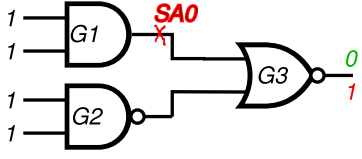
- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs



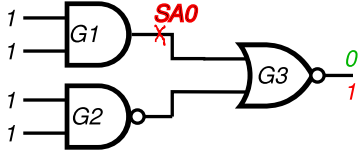
- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

The "stuck-at" fault model

- We assume that the only effect of a defect is a node that is stuck at low or high logic level.
- We assume that there is only one "stuck-at" fault in the circuit.
- All the remaining circuit may be used to test if this fault exists.
- How to detect a stuck-at "0" at node between G1 and G3 ?

- **Controllability**
- Try to put "1" on faulty node.
- Put "1" at G1 output
- Put "1,1" at G1 inputs



- **Observability**
- G3 used to transmit value.
- Put "0" on other G3 input.
- Put "1,1" on G2 inputs.

Structural test

Stuck-at fault model

- Usable for any combinational block based on simple combinational gates.
- Inputs are called **primary inputs**
- Outputs are called **primary outputs**
- For any combinational netlist algorithmic tools can compute a **test program** :
  - The test program is a set of **test vectors**.
  - A test vector is the union of a **stimuli** on primary inputs, and **expected values** on primary outputs.
- A 100% stuck-at fault coverage can be achieved.

Structural test

Stuck-at fault model

- Usable for any combinational block based on simple combinational gates.
- Inputs are called **primary inputs**
- Outputs are called **primary outputs**
- For any combinational netlist algorithmic tools can compute a **test program** :
  - The test program is a set of **test vectors**.
  - A test vector is the union of a **stimuli** on primary inputs, and **expected values** on primary outputs.
- A 100% stuck-at fault coverage can be achieved.

Structural test

Stuck-at fault model

- Usable for any combinational block based on simple combinational gates.
- Inputs are called **primary inputs**
- Outputs are called **primary outputs**
- For any combinational netlist algorithmic tools can compute a **test program** :
  - The test program is a set of **test vectors**.
  - A test vector is the union of a **stimuli** on primary inputs, and **expected values** on primary outputs.
- A 100% stuck-at fault coverage can be achieved.

Structural test

Stuck-at fault model

- Usable for any combinational block based on simple combinational gates.
- Inputs are called **primary inputs**
- Outputs are called **primary outputs**
- For any combinational netlist algorithmic tools can compute a **test program** :
  - The test program is a set of **test vectors**.
  - A test vector is the union of a **stimuli** on primary inputs, and **expected values** on primary outputs.
- A 100% stuck-at fault coverage can be achieved.

Structural test

Stuck-at fault model

- Usable for any combinational block based on simple combinational gates.
- Inputs are called **primary inputs**
- Outputs are called **primary outputs**
- For any combinational netlist algorithmic tools can compute a **test program** :
  - The test program is a set of **test vectors**.
  - A test vector is the union of a **stimuli** on primary inputs, and **expected values** on primary outputs.
- A 100% stuck-at fault coverage can be achieved.

Structural test

Stuck-at fault model

- Usable for any combinational block based on simple combinational gates.
- Inputs are called **primary inputs**
- Outputs are called **primary outputs**
- For any combinational netlist algorithmic tools can compute a **test program** :
  - The test program is a set of **test vectors**.
  - A test vector is the union of a **stimuli** on primary inputs, and **expected values** on primary outputs.
- A 100% stuck-at fault coverage can be achieved.

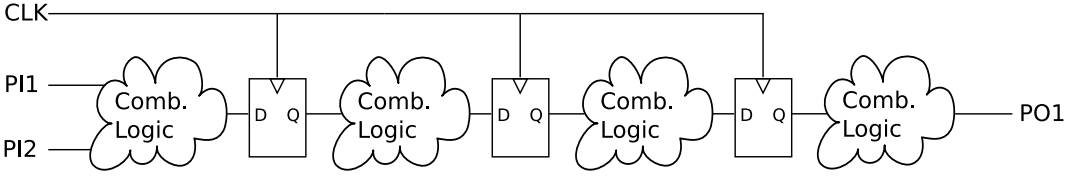
Structural test

Stuck-at fault model

- Usable for any combinational block based on simple combinational gates.
- Inputs are called **primary inputs**
- Outputs are called **primary outputs**
- For any combinational netlist algorithmic tools can compute a **test program** :
  - The test program is a set of **test vectors**.
  - A test vector is the union of a **stimuli** on primary inputs, and **expected values** on primary outputs.
- A 100% stuck-at fault coverage can be achieved.

Structural test

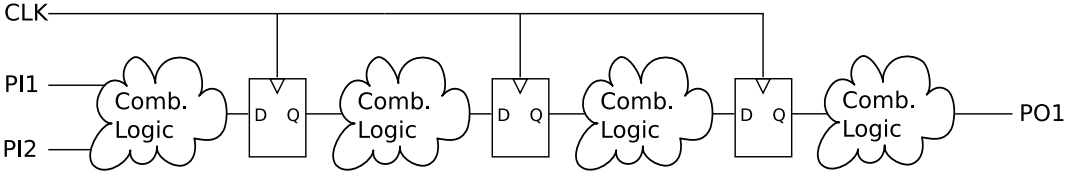
How to get full observability and full controllability in a synchronous circuit ?



- A digital circuit is build of combinational bloc synchronized by D flip-flops.
- Only a few primary inputs are usable.
- Only a few primary outputs are usable.
- Some combinational blocs are completely isolated from PIs or POs

Structural test

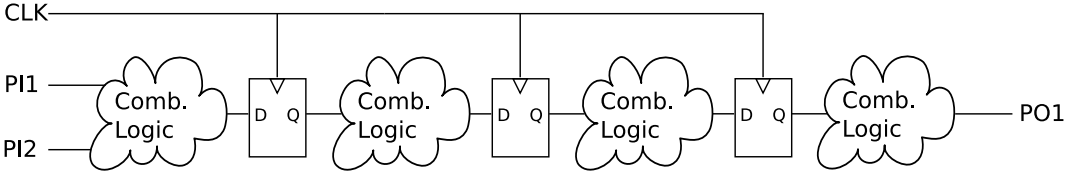
How to get full observability and full controllability in a synchronous circuit ?



- A digital circuit is build of combinational bloc synchronized by D flip-flops.
- Only a few primary inputs are usable.
- Only a few primary outputs are usable.
- Some combinational blocs are completely isolated from PIs or POs

Structural test

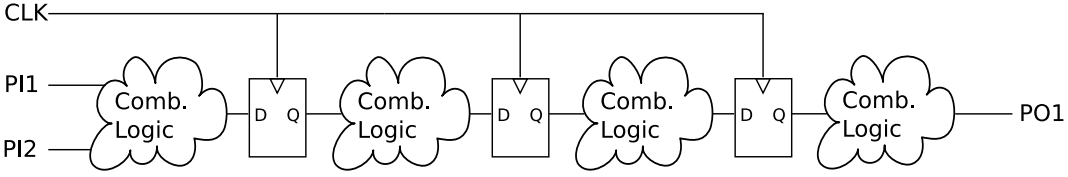
How to get full observability and full controllability in a synchronous circuit ?



- A digital circuit is build of combinational bloc synchronized by D flip-flops.
- Only a few primary inputs are usable.
- Only a few primary outputs are usable.
- Some combinational blocs are completely isolated from PIs or POs

Structural test

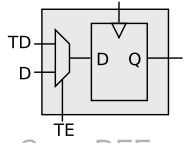
How to get full observability and full controllability in a synchronous circuit ?



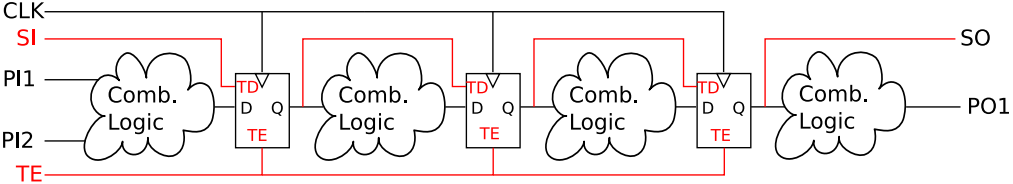
- A digital circuit is build of combinational bloc synchronized by D flip-flops.
- Only a few primary inputs are usable.
- Only a few primary outputs are usable.
- Some combinational blocs are completely isolated from PIs or POs

Structural test

Scan Flip-Flop and Scan-chain



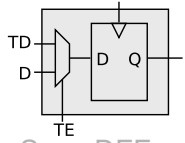
- Each DFF is replaced by a Scan-DFF.
- In test mode (TE=1) D input of the flip_flop is replaced by TD input
- Scan DFF are chained in a long shift register.



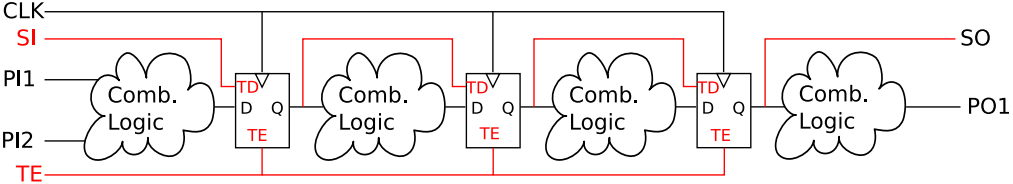
- A new SI serial input is used for controllability.
- A new SO serial output is used for observability.
- A test enable mode is inserted.

Structural test

Scan Flip-Flop and Scan-chain



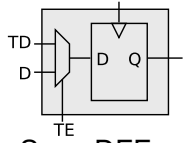
- Each DFF is replaced by a Scan-DFF.
- In test mode (TE=1) D input of the flip_flop is replaced by TD input
- Scan DFF are chained in a long shift register.



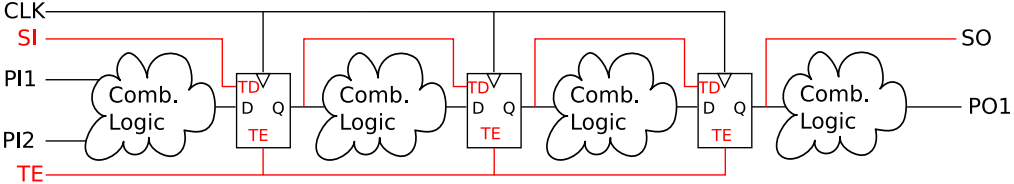
- A new SI serial input is used for controllability.
- A new SO serial output is used for observability.
- A test enable mode is inserted.

Structural test

Scan Flip-Flop and Scan-chain



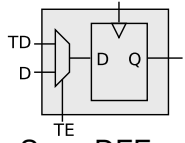
- Each DFF is replaced by a Scan-DFF.
- In test mode (TE=1) D input of the flip_flop is replaced by TD input
- Scan DFF are chained in a long shift register.



- A new SI serial input is used for controllability.
- A new SO serial output is used for observability.
- A test enable mode is inserted.

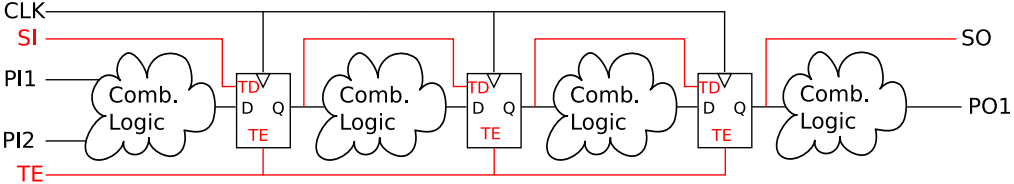
Structural test

Scan Flip-Flop and Scan-chain



- Each DFF is replaced by a Scan-DFF.
- In test mode (TE=1) D input of the flip_flop is replaced by TD input

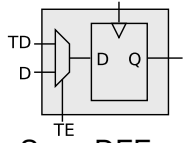
- Scan DFF are chained in a long shift register.



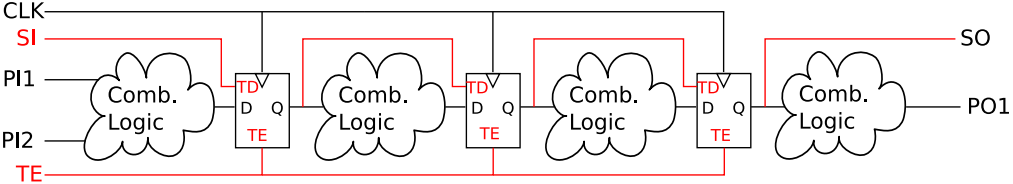
- A new SI serial input is used for controllability.
- A new SO serial output is used for observability.
- A test enable mode is inserted.

Structural test

Scan Flip-Flop and Scan-chain



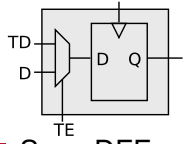
- Each DFF is replaced by a Scan-DFF.
- In test mode (TE=1) D input of the flip_flop is replaced by TD input
- Scan DFF are chained in a long shift register.



- A new SI serial input is used for controllability.
- A new SO serial output is used for observability.
- A test enable mode is inserted.

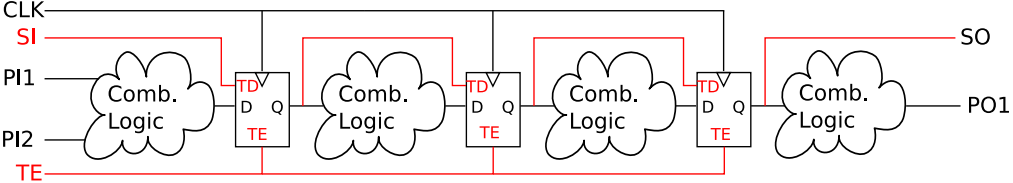
Structural test

Scan Flip-Flop and Scan-chain



- Each DFF is replaced by a Scan-DFF.
- In test mode (TE=1) D input of the flip_flop is replaced by TD input

- Scan DFF are chained in a long shift register.



- A new SI serial input is used for controllability.
- A new SO serial output is used for observability.
- A test enable mode is inserted.

Structural test

Scan Test and design automation

- A test program consists on loops of the following procedure :
  - TE=1 A test vector is loaded via input SI using the shift-register.
  - TE=0 A one cycle computation is done in normal mode. All registers are loaded by computed values.
  - TE=1 The shift-register is dumped via SO, results are compared to expected results.
- Scan chain insertion can be fully automatic (during synthesis)
- Test vector generation can be fully automatic (after synthesis)
- Warning Using scan test reduces performances (lower clock frequency, higher power consumption)

Structural test

Scan Test and design automation

- A test program consists on loops of the following procedure :
  - TE=1 A test vector is loaded via input SI using the shift-register.
  - TE=0 A one cycle computation is done in normal mode. All registers are loaded by computed values.
  - TE=1 The shift-register is dumped via SO, results are compared to expected results.
- Scan chain insertion can be fully automatic (during synthesis)
- Test vector generation can be fully automatic (after synthesis)
- Warning Using scan test reduces performances (lower clock frequency, higher power consumption)

Structural test

Scan Test and design automation

- A test program consists on loops of the following procedure :
  - TE=1 A test vector is loaded via input SI using the shift-register.
  - TE=0 A one cycle computation is done in normal mode. All registers are loaded by computed values.
  - TE=1 The shift-register is dumped via SO, results are compared to expected results.
- Scan chain insertion can be fully automatic (during synthesis)
- Test vector generation can be fully automatic (after synthesis)
- Warning Using scan test reduces performances (lower clock frequency, higher power consumption)

Structural test

Scan Test and design automation

- A test program consists on loops of the following procedure :
  - TE=1 A test vector is loaded via input SI using the shift-register.
  - TE=0 A one cycle computation is done in normal mode. All registers are loaded by computed values.
  - TE=1 The shift-register is dumped via SO, results are compared to expected results.
- Scan chain insertion can be fully automatic (during synthesis)
- Test vector generation can be fully automatic (after synthesis)
- Warning Using scan test reduces performances (lower clock frequency, higher power consumption)

Structural test

Scan Test and design automation

- A test program consists on loops of the following procedure :
  - TE=1 A test vector is loaded via input SI using the shift-register.
  - TE=0 A one cycle computation is done in normal mode. All registers are loaded by computed values.
  - TE=1 The shift-register is dumped via SO, results are compared to expected results.
- Scan chain insertion can be fully automatic (during synthesis)
- Test vector generation can be fully automatic (after synthesis)
- Warning Using scan test reduces performances (lower clock frequency, higher power consumption)

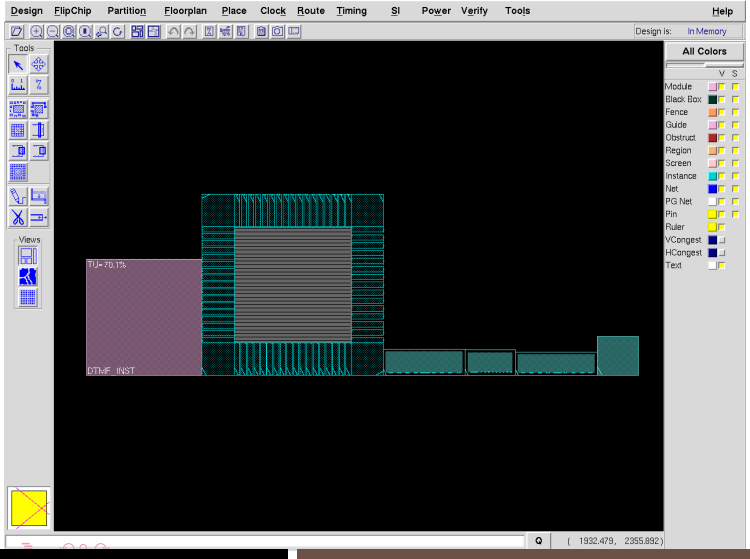
Structural test

Scan Test and design automation

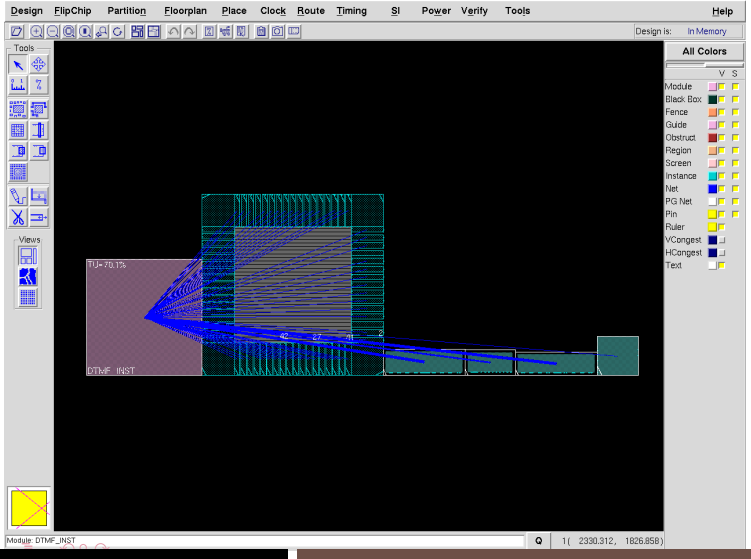
- A test program consists on loops of the following procedure :
  - TE=1 A test vector is loaded via input SI using the shift-register.
  - TE=0 A one cycle computation is done in normal mode. All registers are loaded by computed values.
  - TE=1 The shift-register is dumped via SO, results are compared to expected results.
- Scan chain insertion can be fully automatic (during synthesis)
- Test vector generation can be fully automatic (after synthesis)
- Warning Using scan test reduces performances (lower clock frequency, higher power consumption)

Outline

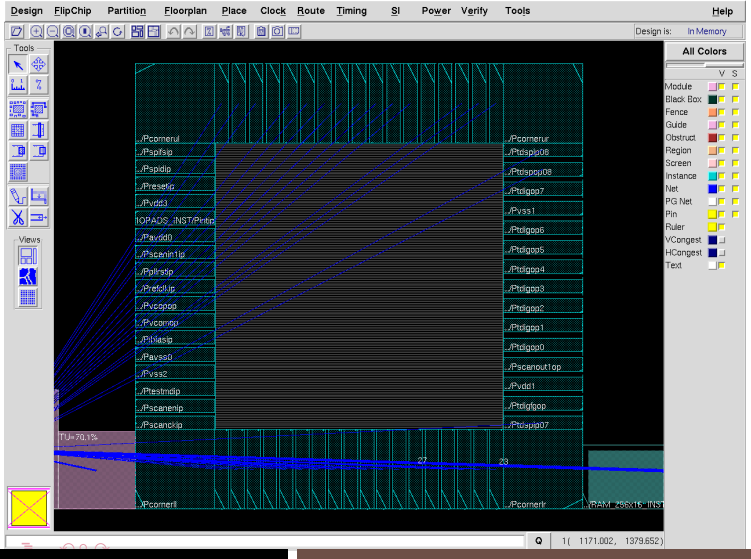
SoC Encounter tool gui.



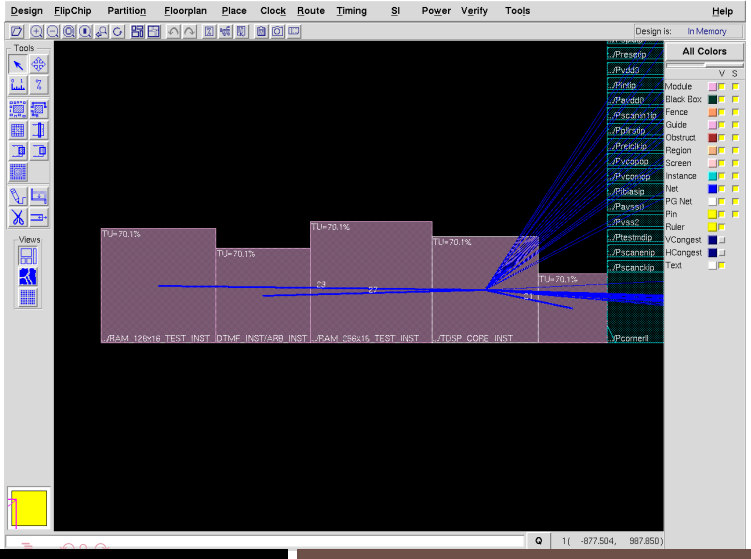
Viewing nets as elastic wires.



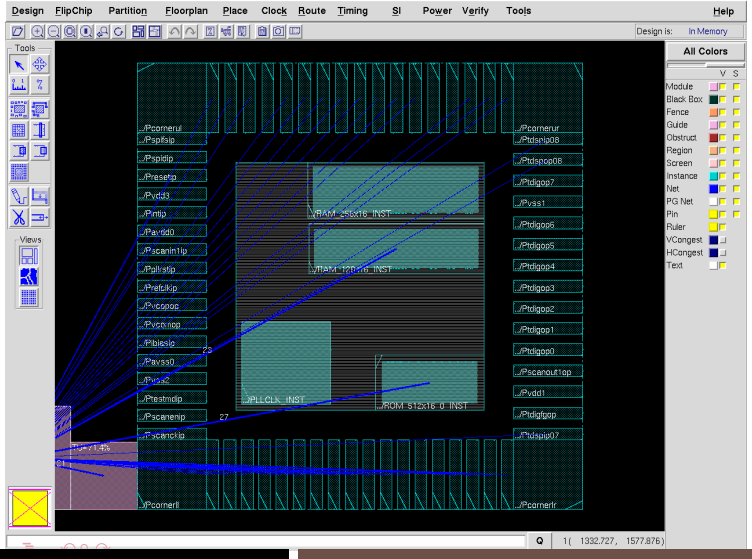
Pads around de core, Corner pads.



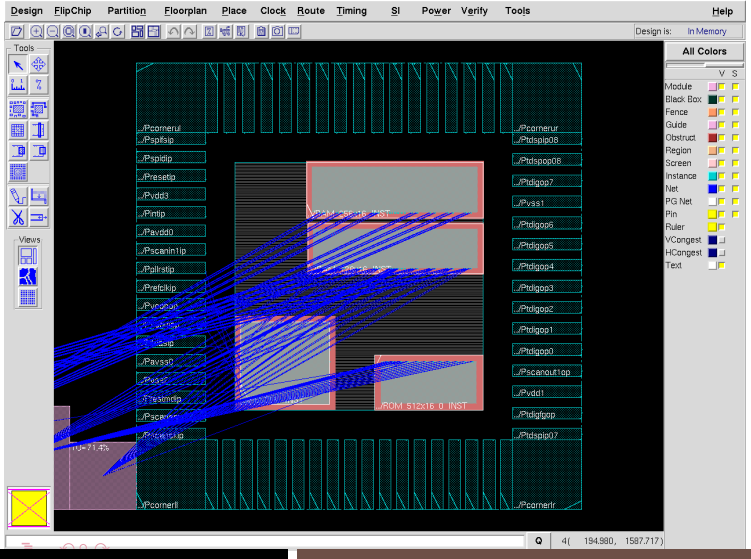
The design hierarchy is preserved in the netlist



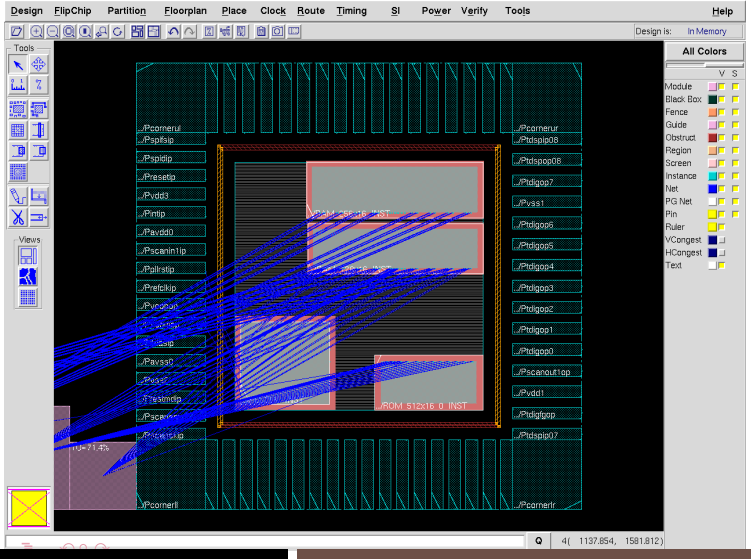
Floorplan : Placement of macro-cells w/o power rings



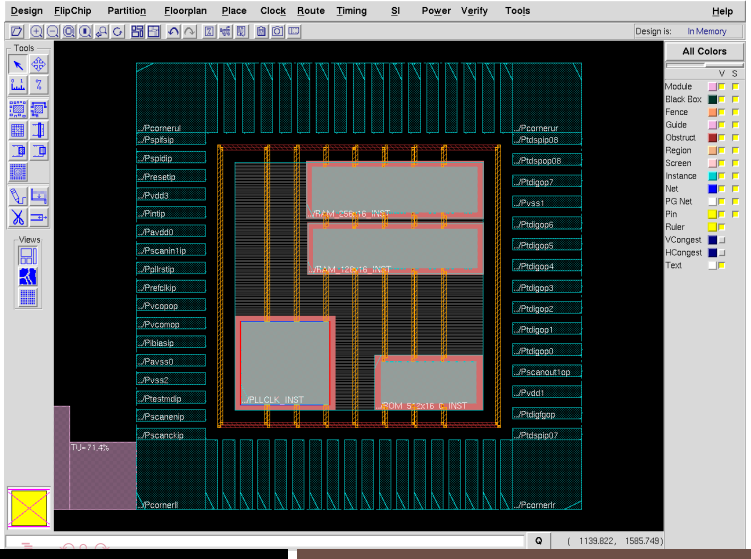
Floorplan : Defining prohibited areas for standard cells



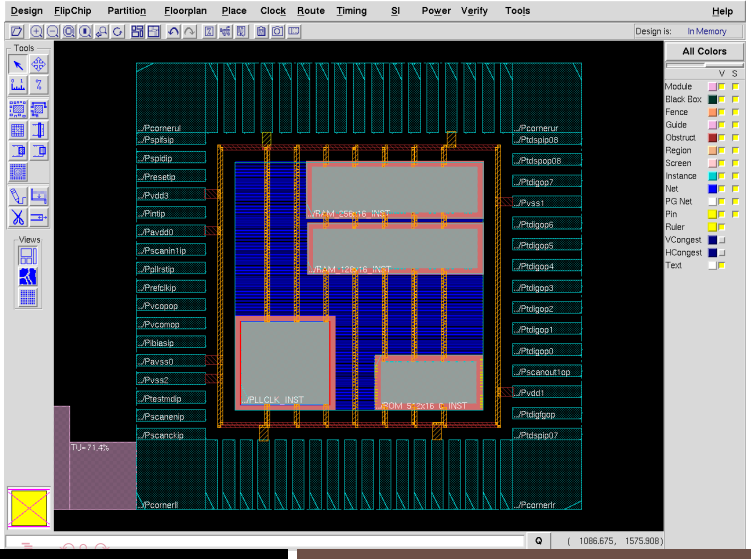
Floorplan : Defining a global power ring



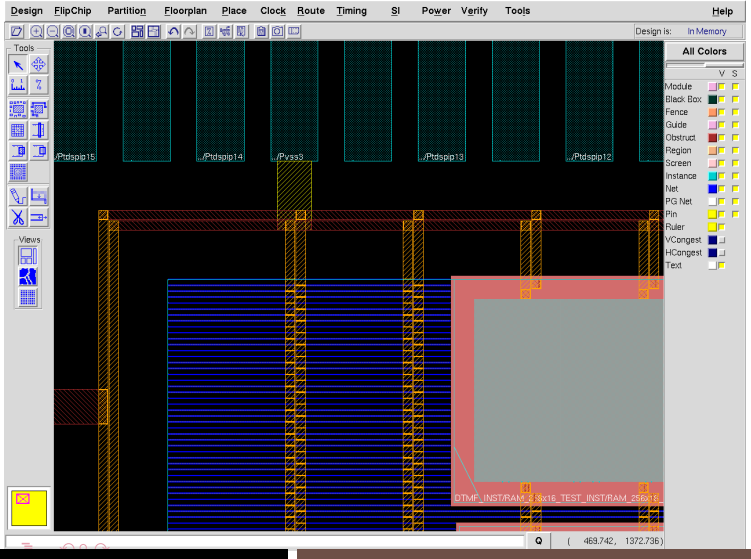
Floorplan : Routing reinforcement power stripes



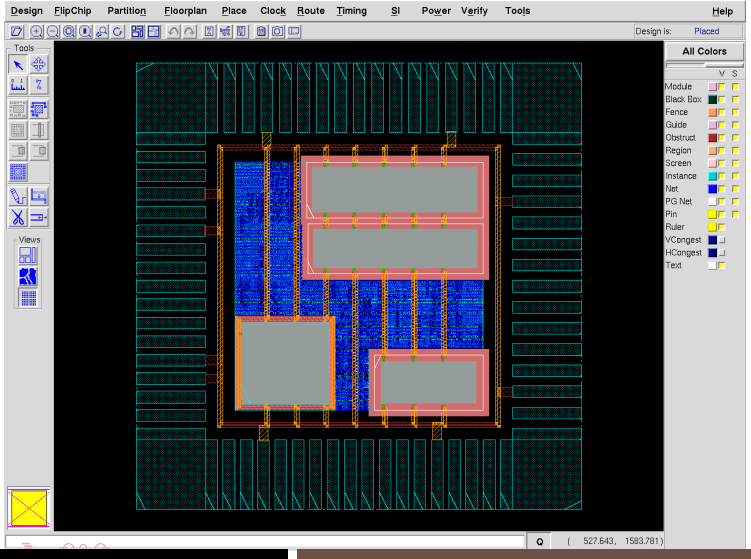
Floorplan : Connecting Standard Cell Raws and Power pads



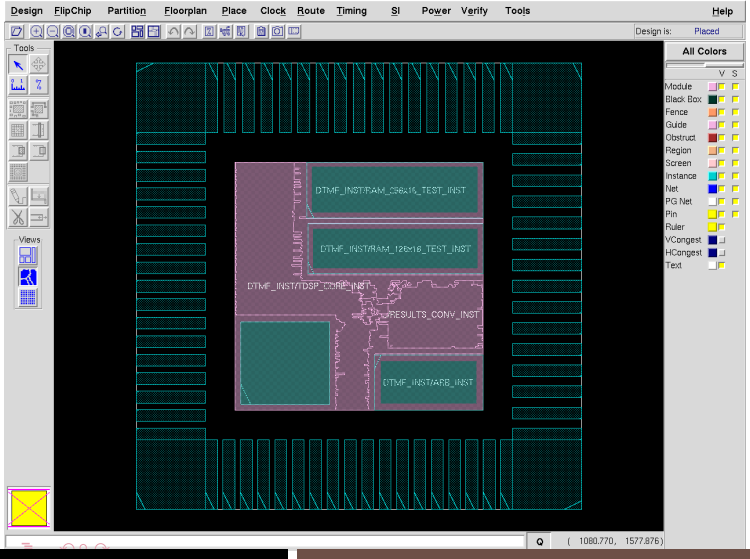
Connection between lines uses arrays of standard size vias.



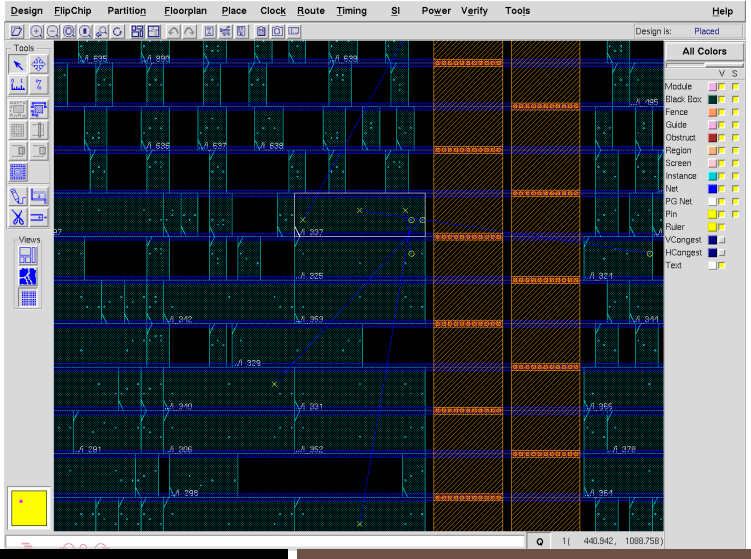
PLACEMENT : Automatic placement of standard cells.



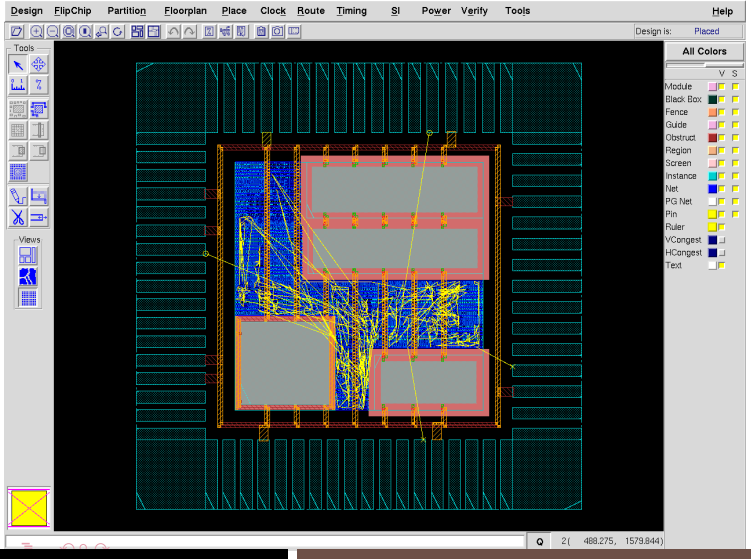
PLACEMENT : Hierarchy analysis, placement guides.



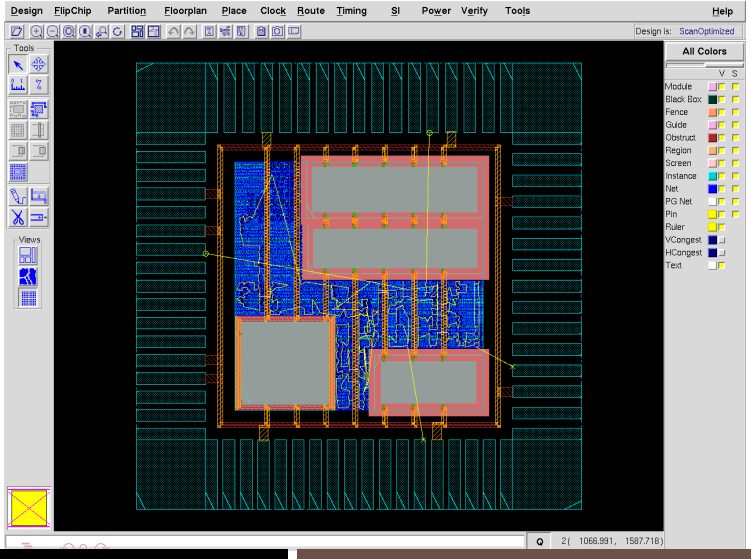
PLACEMENT : Details.



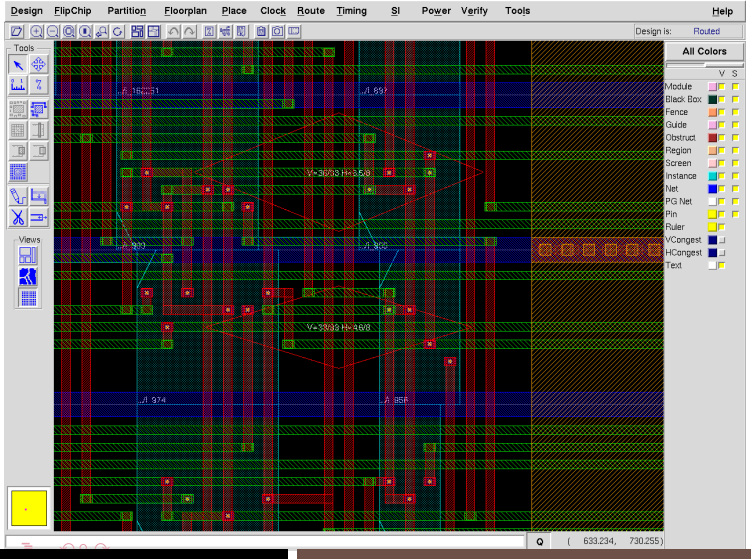
Initial SCAN chain for test



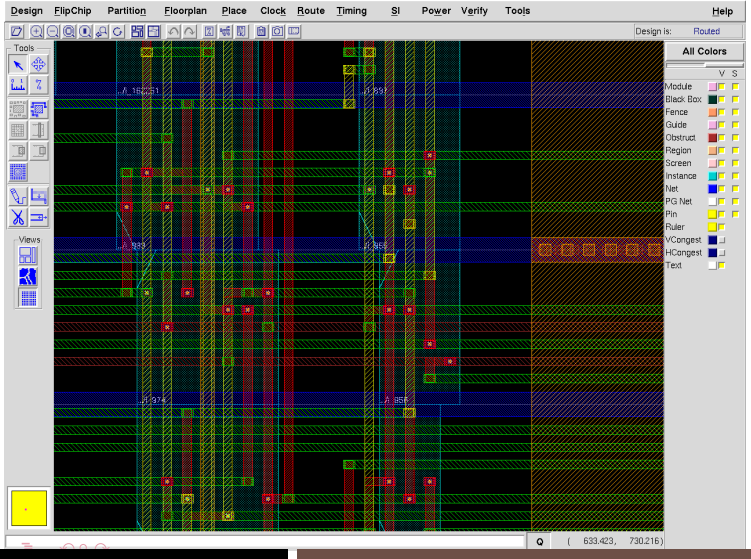
Optimized restructured scan chain



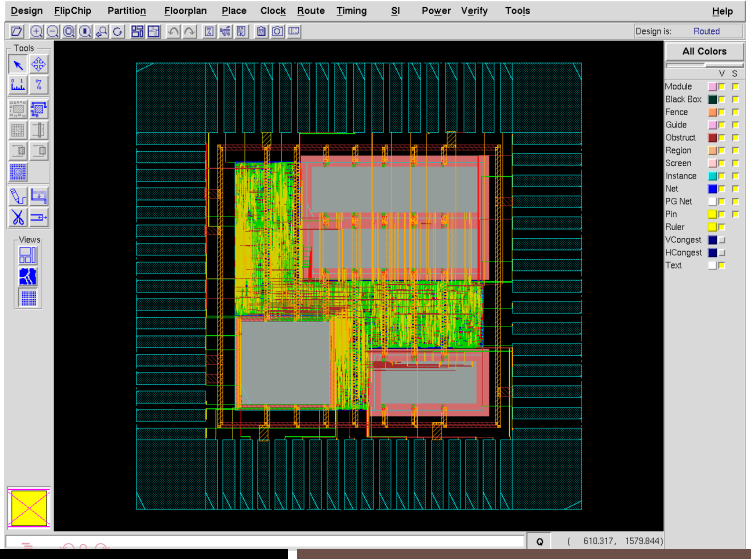
Trial routing : not enough metal levels



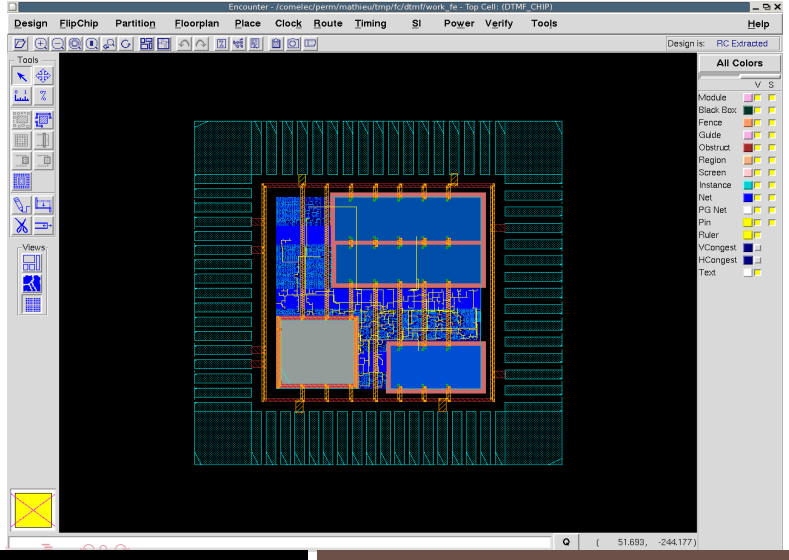
Trial routing : better results with 6 levels



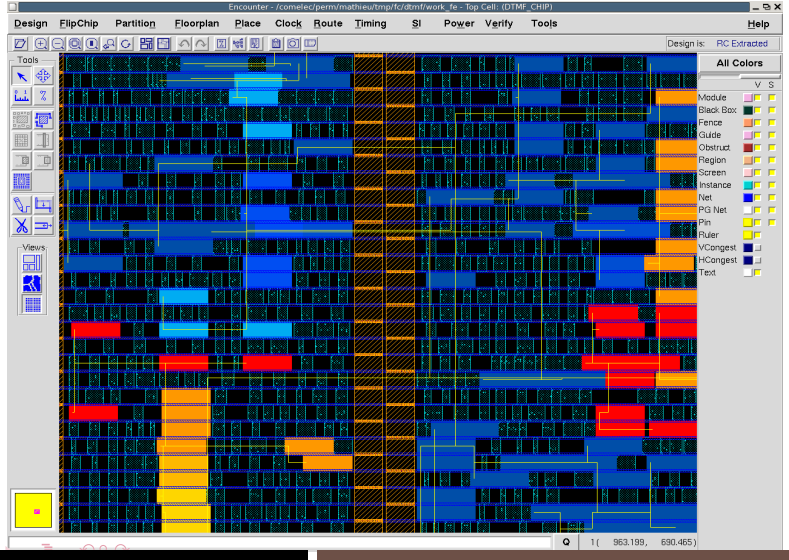
Trial routing : the full circuit



Clock tree synthesis : the tree of buffers



Clock tree synthesis : Skew evaluation on CLK DFF inputs



In place optimization and final routing with clock tree