Hardware describing languages, high level tools and Synthesis

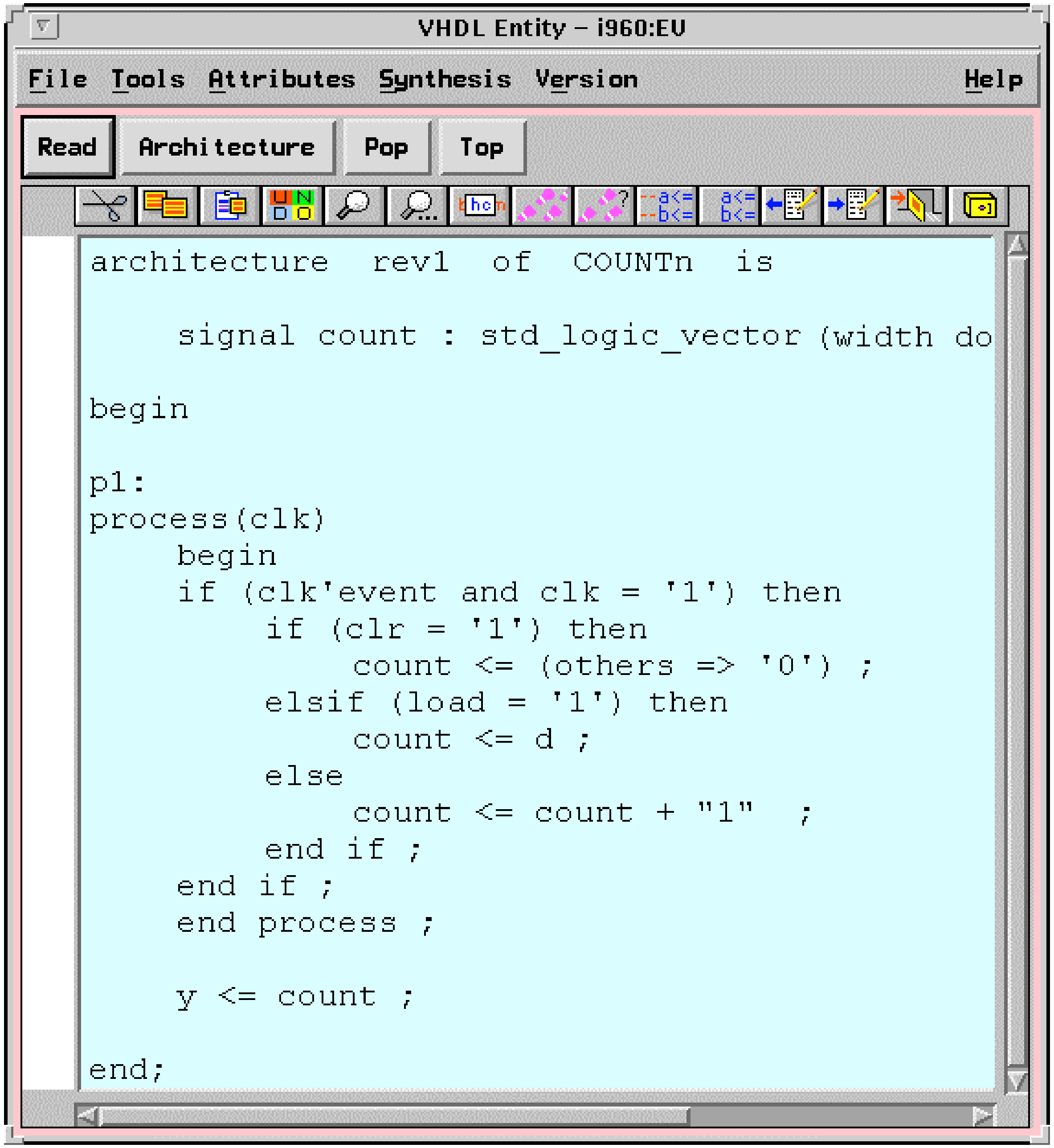
Hardware describing languages (HDL)

- Compiled/Interpreted
  - Compiled:
    - Description compiled into C and then into binary or directly into binary
    - Fast execution
    - Slow compilation
  - Interpreted:
    - Description interpreted at run time
    - Slow execution
    - Fast “compilation”
    - Many interactive features
  - VHDL normally compiled
  - Verilog exists in both interpreted and compiled versions

Data Entry Tools

Design entry

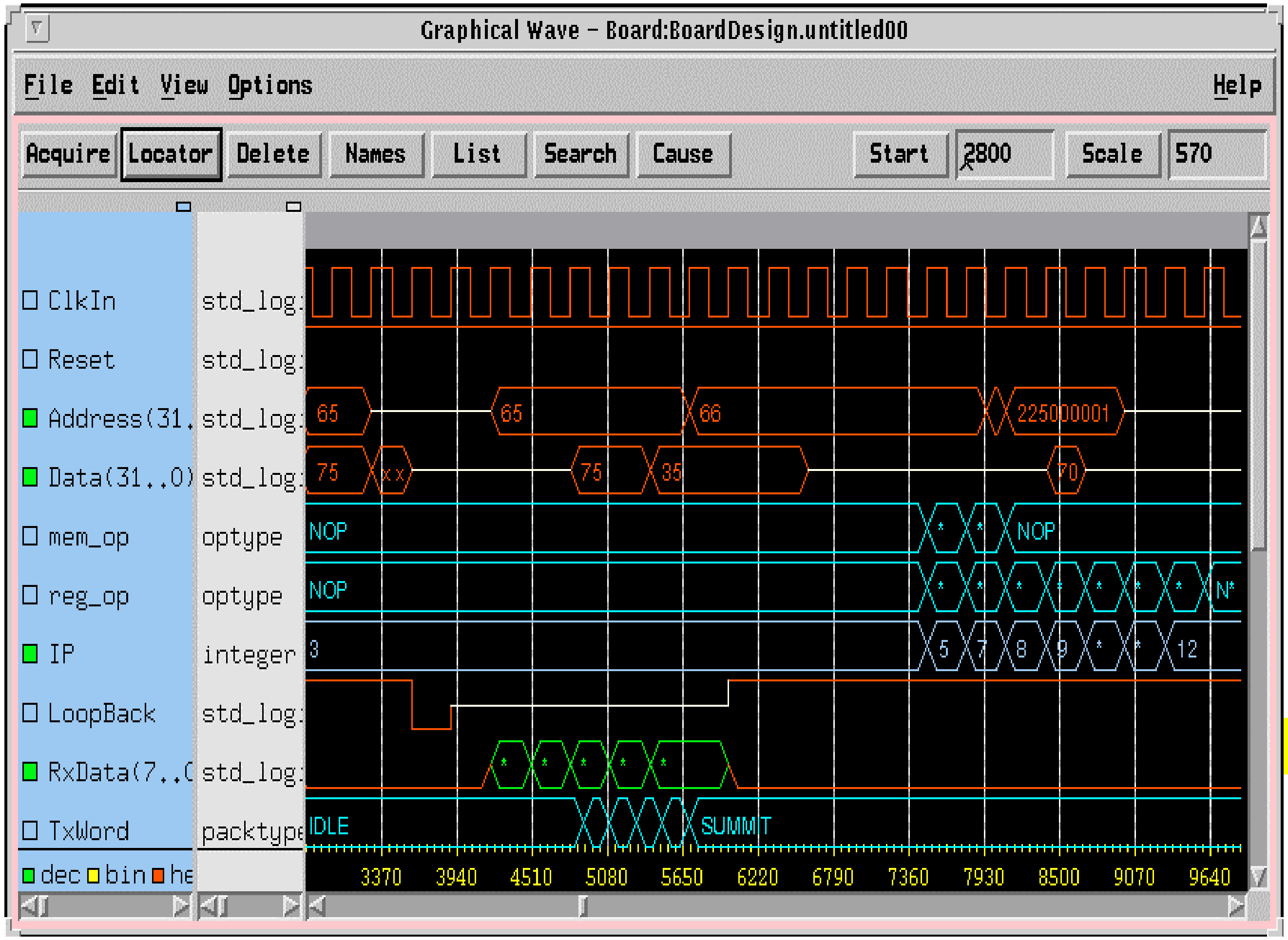
- Text:
  - Tool independent
  - Good for describing algorithms
  - Bad for getting an overview of a large design


VHDL Entity – i960:EU
File Tools Attributes Synthesis Version Help
Read Architecture Pop Top

```
architecture  rev1  of  COUNTn  is

     signal count : std_logic_vector (width do

begin

p1:
process(clk)
     begin
     if (clk'event and clk = '1') then
          if (clr = '1') then
               count <= (others => '0') ;
          elsif (load = '1') then
               count <= d ;
          else
               count <= count + "1"  ;
          end if ;
     end if ;
     end process ;

     y <= count ;

end;
```

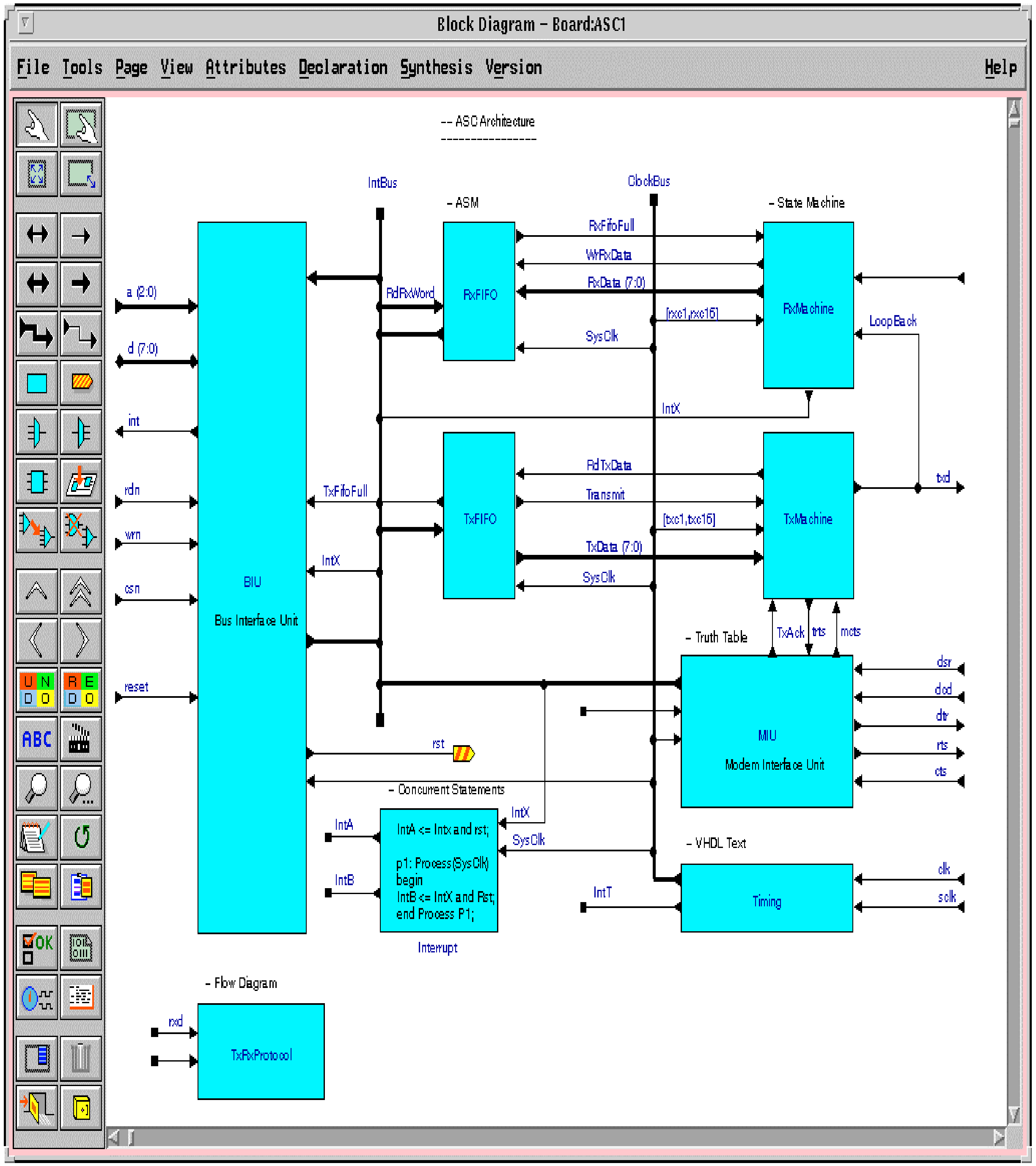
Add-on tools

- Block diagrams to get overview of hierarchy
- Graphical description of final state machines (FSM)
  - Generates synthesizable HDL code
- Flowcharts
- Language sensitive editors
- Waveform display tools

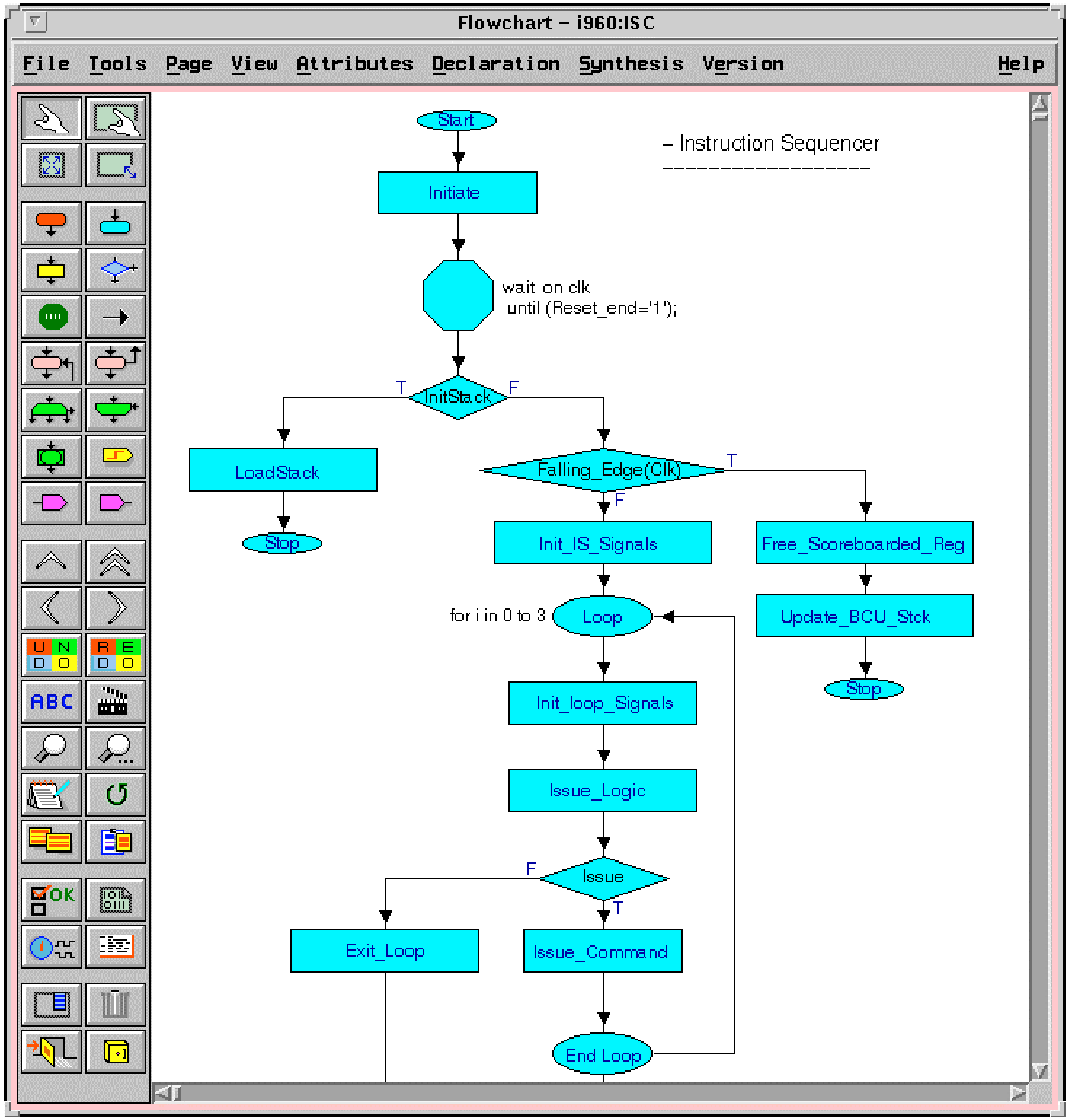


From Visual HDL, Summit design

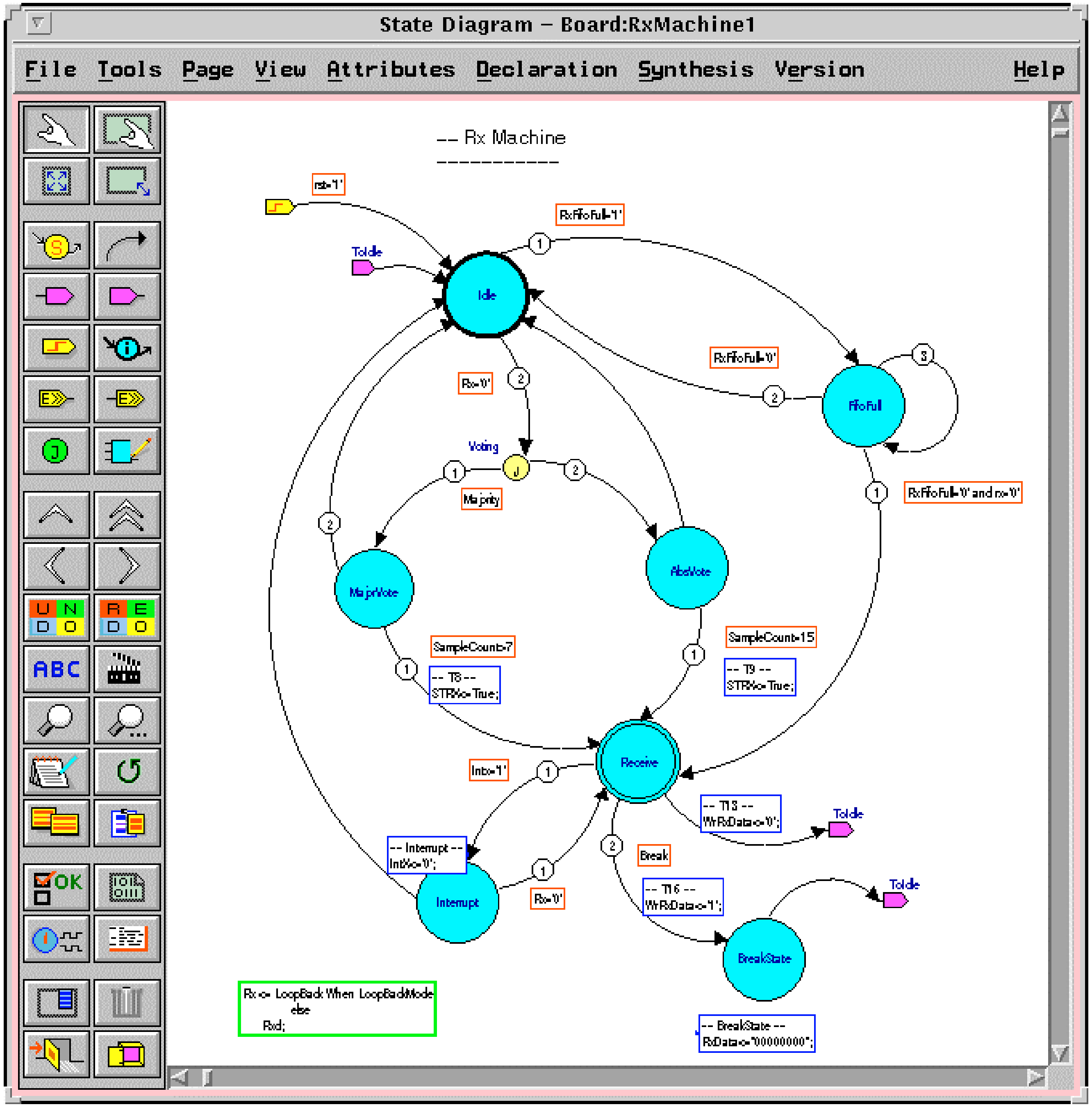
From Visual HDL, Summit design



From Visual HDL, Summit design



From Visual HDL, Summit design



Synthesis

Synthesis

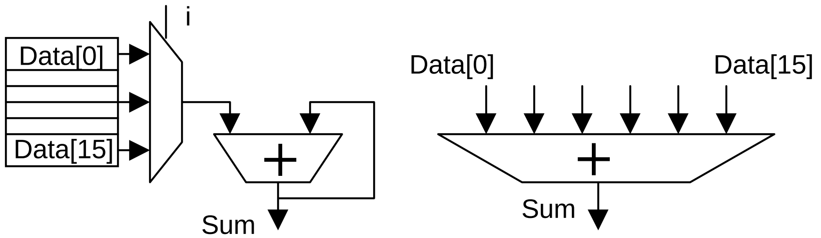
Algorithm

0% technology dependent

```
For i = 0 ; i = 15
      sum = sum + data[I]
```

Architecture

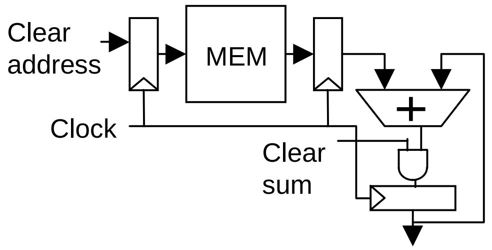
10% technology dependent



Behavioral synthesis

Register level

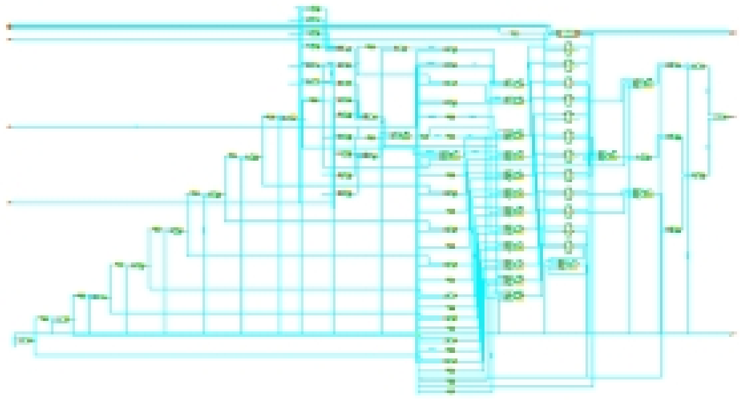
20% technology dependent



Logic synthesis

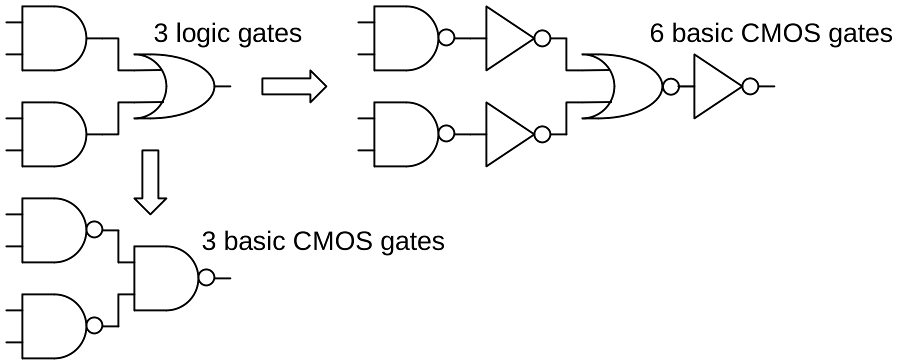
Gate level

100% technology dependent



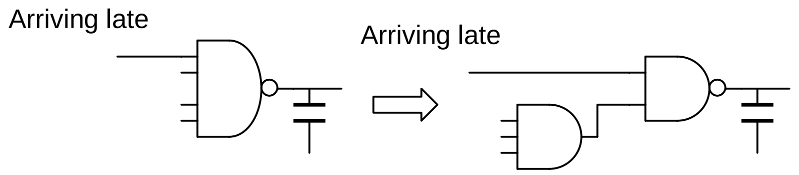
Logic synthesis

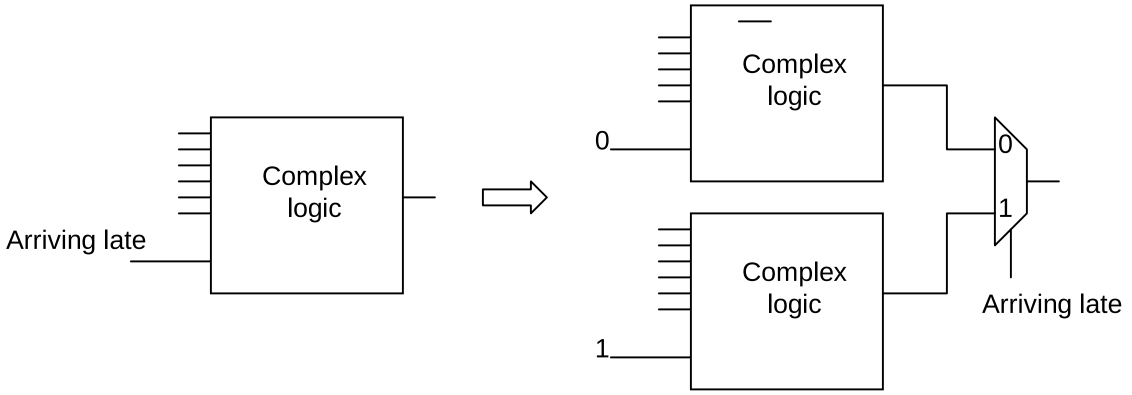
- HDL compilation (from VHDL or Verilog)
  - Registers: Where storage is required
  - Logic: Boolean equations, if-then-else, case, etc.
- Logic optimization
  - Logic minimization (similar to Karnaugh maps)
  - Finds logic sharing between equations
  - Maps into gates available in given technology
  - Uses local optimization rules



Timing optimization

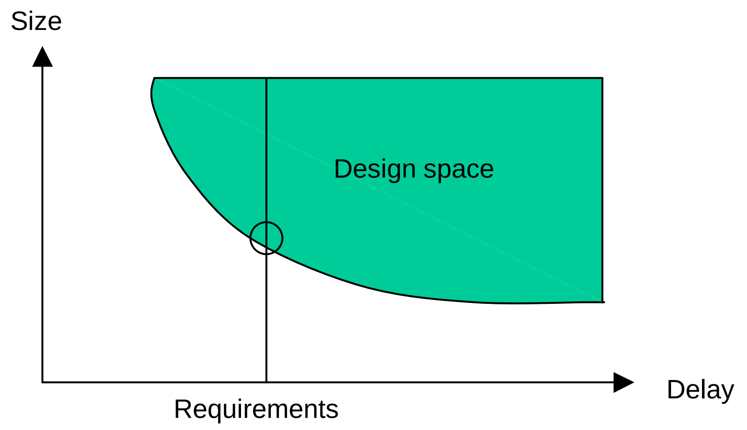
- Estimate loading of wires
- Defined timing constraints (clock frequency, delay, etc.)
- Perform transformations until all constraints fulfilled





Combined timing - size optimization

– Smallest circuit complying to all timing constraints



– Best solution found as a combination of special optimization algorithms and evaluation of many alternative solutions
(Similar to simulated annealing)

Synthesis

- Problems in synthesis
  - Dealing with “single late signal”
  - Mapping into complex library elements
  - Regular data path structures:
    - Adders: ripple carry, carry look ahead, carry select,etc.
    - Multipliers, etc.

    Use special guidance to select special adders, multipliers, etc..

  Performance of sub-micron technologies are dominated by wiring delays (wire capacitance)
- Synthesis in many cases does a better job than a manually optimized logic design.

    (in much shorter time)

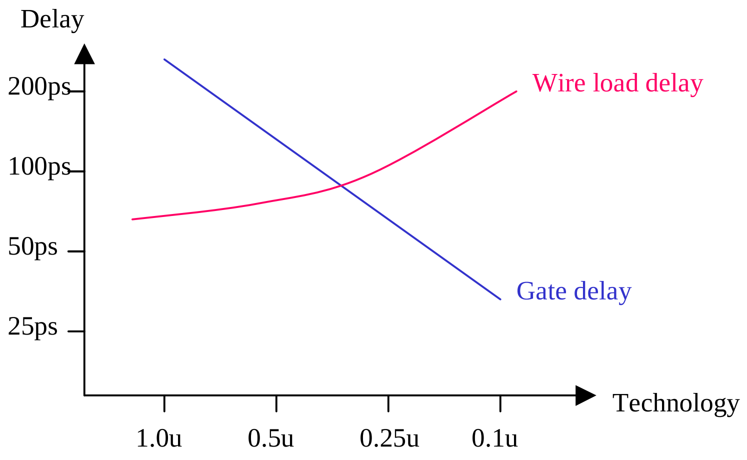
Wire loading

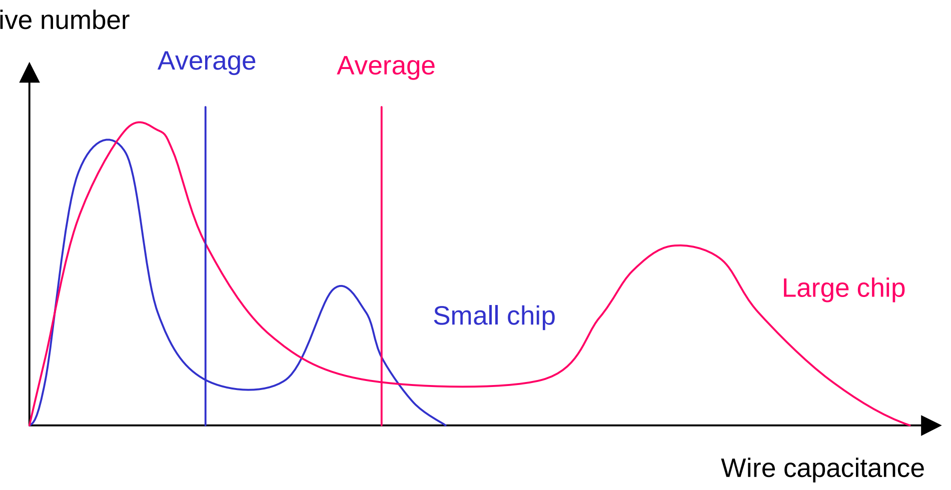
Timing optimization is based on a wire loading model.

Loading of gate = input capacitance of following gates + wire capacitance

Gate loading known by synthesizer

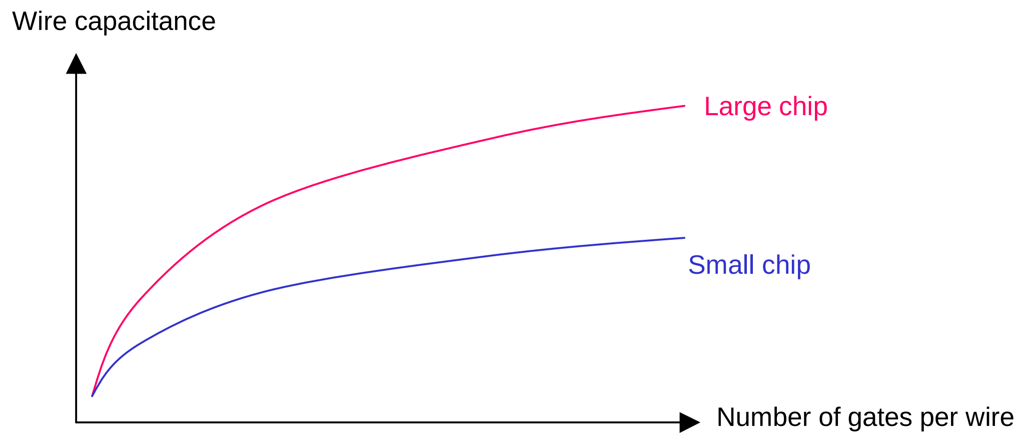
Wire loading must be estimated





Estimate wire capacitance

- Estimate wire capacitance from number of gates connected to wire.



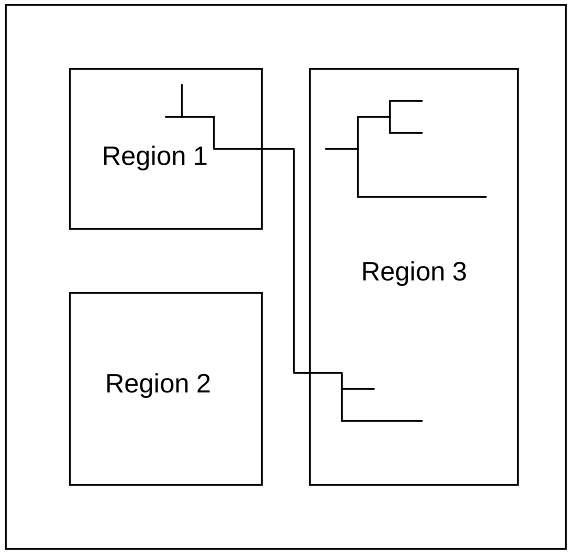
Advantage:	Simple model
Disadvantage:	Bad estimate of long wires (which limits circuit performance)

Using floor plan

- Estimate using floor plan

Inside local region:
Estimate as function of number
of gates and size of region

Between regions:
Use estimate of physical distance
between routing regions.



Advantage:	Realistic estimate
Disadvantage:	Synthesizer most work with complete design

Iteration and Timing

- Iteration
  - Synthesis with crude estimation
  - P&R with extraction of real loading
  - Re-synthesize starting from real loads
  - Repeat X times
- Timing driven P&R
  - Synthesize with crude estimation
  - Use timing calculations from synthesis to control P&R
- Integration of synthesis and P&R
  - Floor planning - timing driven - iteration

Performance - synthesis - layout - timing

Synthesis in the future

- Integration of synthesis and P&R
- Synthesizable standard modules (processor, PCI interface, Digital filters, etc.)
- Automatic insertion of scan path for production testing.
- Synthesis for low power
- Synthesis of self-timed circuits (asynchronous)
- Behavioral synthesis
- Formal verification

Introduction To Mentor Graphics BOLD Browser

- Mentor Graphics BOLD browser allows to access:
  - manuals,
  - Design Architect which is used for schematic capture,
  - and QuicksimII which is used for simulating digital logic and microcomputer circuits.
- To add the package type: ‘addpkg’ at the prompt.
- Note the number to the left of the Mentor, and enter it.

Connecting To Mentor Tools

- For the first time to add package to your account : Type ‘addpkg’ command at the Unix prompt.
- When the list appears find Mentor and note the number besides it.
- Enter the number and press return key.

Invoking The Design Manager

- Design manger allows to access all the available tools from one integrated environment.
- After logging in type :
- *‘ setenv MGC_WC/u/your_user_name’* at the prompt and press return.
- After this type ‘dmgr’ which will invoke Design Manager.

Invoking Design Architect

- After you invoke Design Manager, the central portion of the window shows the entry for each file in your home directory, and left side shows icon for the Mentor Graphics tools.
- Using scroll bar find icon for Design Architect and left click it.

Opening A Sheet And Setting Grid Spacing

- To open a Sheet click on Open Sheet icon from Session_palette of the Design Architect, which appears at the right corner.
- Enter the component name which corresponds to the filename in the open sheet form.
- A window will appear with grid of dots and plus signs.
  - The default setting is 0.25 inches mark off the dots and plus sign.

Opening The Sheet And Setting Grid Spacing.

- With the default setting the plus sign mark off and dots mark off are 0.25 inches.
- You can change the setting by putting the cursor on Setup entry along the Design Architect window and clicking on page entry, and writing the new setting in the page form’s Pin Space box.
- To change the grid spacing click on Grid entry of Setup entry and repeat the procedure.

Opening The Sheet And Setting The Grid Spacing

- Click on Open Sheet icon from Session_palette section of Architect window which appears a the right hand corner.
- On the open sheet form, type the name of the component, and click O.K.
- After which a window appears which has dots and plus sign.

Adding A Border And Title

- To add a Rectangular border
  - Edit entry   Edit Command   Add command   Rectangular Box.
- Repeat the same procedure to add the Title box.
- To have a center line of the title box
  - Edit   Edit Command   Add Command   Line.
- To put some text in the title block
  - Edit   Edit Command   Add Command   Text.

Adding Text

- To add some text in the title box:
  - Edit   Edit Commands   Add Comment   Text Command.
- The default height for the text is small, to change it:
  - Edit   Edit Command   Change Attributes   Text   Height Command

Getting And Placing The Library Component

- Drawing the schematic, we use the component from two libraries which are accessed through the Libraries entry at the top of the Design Architect window.
- For Logic devices we use the models from Logic Modeling Corporation.
- For Vcc,ground,portin and portout connectors we use Mentor Graphics misc_lib.

Library Component

- Logic Modeling Corporation Components : Libraries entry LMC SmartModels
- Clicking will list major LMC component sublibraries,in the right side of the window.
  - Choose the appropriate logic entry.
- To go back to the previous menu from the current one of the logic entry, put the cursor in a blank section of the window containing the list, with the help of right key pop up the palette & go back.

Library Component

- Palette menu doesn’t have a default scroll bar.
- To add one , pop up the Palette menu and choose Show Scroll bars.
- Gates are chosen from the library according to the need.
- Portin, portouts ,Vcc components are added from Mentor misc Library.
  - To get this library: Libraries Menu   MGC_Digital Library
- The list will appear at the right hand corner.
- Choose misc_lib entry.

Drawing Nets (Wires)

- To draw wires : Add Menu ➡ Wire.
- When the menu disappears the cursor changes to
- Move the cursor where to connect the wires.
- Press Esc key/cancel when done otherwise you will be in the ✚ net drawing mode.
- To exit from the net editor press the cancel button at the bottom of the schematic window.
- To make a back up of the sheet, use File ➡ Save

Checking And
Quitting The Sheet

- Before plotting the schematic check the sheet by Sheet $\Rightarrow$ With Default.
- If report shows no errors , close the report.If it shows some then correct them.
- Save your sheet and then for quitting the Design Architect Menu $\Rightarrow$ Quit.
- This will close the Design Architect window but “orphan” command will be left, to close it, move the cursor into the window,hold Control key & press C.
  - use same for Design Manager.

Setting Up The Design Viewpoint

- Mentor Graphics define the Design Viewpoint s a set of rules used to establish the configuration of your design.
- Steps to create the proper design viewpoint reference for the QuickSimII simulator & the Logic Modeling Corp. models.
- Design Manager ➡ Design Viewpoint Editor (DVE) ➡ Double click
- When window appear ➡ OPEN VPT on right & click.

Starting QuickSim And Setting Up window

- For starting : Design Manager Tools ➡QuickSim
- In the top box of the window type filename.
  - Timing Mode ➡ Delay ➡ Detail of Delay ➡ Visible
- Read the contents of the small Model Message window after QuickSim window appears.
  - it should just contain a short message identifying Logic Modeling Corporation.
- Choose QuickSim from blank region of the window and then open sheet

Starting QuickSim

- Add a window which displays logic analyzer type traces of your stimulus signals and the resulting output signal(S)
- To start QuickSim ➡ Add ➡ Traces ➡choose specified.
- Add traces form appears .
  - click on Named signals and then write names.
- In Trace window note that the signal names along the left side and time scale along the bottom .
  - Very similar to the timing display of a logic analyzer.

Starting QuickSim

- Helpful to have the input and output signals displayed in a list format as well as in the waveform format.
- To produce list:
  QuickSim $\Rightarrow$ Add $\Rightarrow$ List menu $\Rightarrow$ choose specified.
- When form appear click $\Rightarrow$ Named signals $\Rightarrow$ Fill the signal names.

Creating Input Stimulus Signals For Verifying The Logic

- To thoroughly test the logic of the circuit you need to apply each of the possible combinations to the inputs and check if the output is correct.
- The most common way to generate input stimulus signals in QuickSim is with **Forced** commands.
  - you can force a single value on a signal line or force sequence of values(multiple) on a signal line or force a repetitive signal called a “clock” on a signal line.

Running A Simulation And Viewing The Results

- To run the simulation pop up $\rightarrow$ QuickSim $\rightarrow$ Run Simulation Menu $\rightarrow$ Choose Until Time.
- Write a time in the Until Time box form for one complete input waveform.
- Input and Output waveforms should appear in the trace window and a truth table type display should appear in the list window.
- To Change the view & the scale of the trace.
  – Popup QuickSim $\rightarrow$ View menu $\rightarrow$ Zoom out/in.

Running the Simulation

- Study the the waveform to determine the glitch and scroll the display so that you can see the input signal transition which causes the glitch.
- Activate the list window which shows an entry for each time where an input or output signal changes.
- Use the list entry to find the maximum propagation delay for the circuit.

Adding Probes To Circuit And Running The Simulation.

- To see the trace of the signal on a particular line when run the simulation from 0 time, we can add probe to the output of the gate.
  - For this choose $\rightarrow$ QuickSim $\rightarrow$ Add menu $\rightarrow$ Probe.
- To add a trace for the probe: QuickSim Add menu $\rightarrow$ Trace $\rightarrow$ Choose Specified.
- To delete the trace: $\rightarrow$ QuickSim Delete.

Conclusion

In this project we have discussed the functional decomposition methodology. The point of this method is to split a big design into small pieces, so it makes easier to achieve any design.

For our approach, we will use this methodology then we will use Renoir to capture our diagram blocks, states, truth tables, then we will generate the respective VHDL files needed to burn a Programmable logic Devices such as FPGA’S.
Finally we will perform the test necessary to confirm our expectations.

Creating And Compiling The VHDL Model

- We use Mentor Tools to create schematics for circuit designs containing parts from the Logic Modeling,Inc. libraries and simulate those designs.
- For the cases where a commercial model is not available for a particular device, we can use VHDL.
- The VHDL model can be simulated directly or included in a design containing commercial models and simulated along with these.

Creating And Compiling The VHDL Model

- To start with the VHDL here are the steps:
- Set the environment variables and bring up dmgr.
- Invoke Design Architect.
- Type the VHDL code.
- After you finished with the code, compile it with compile option from the menu.
- While compiling if the error comes,locate the errors ,by putting the cursor on the error message in the compiler window and click to select .

Generating A Schematic Symbol For A VHDL Model

- Open the Design Architect session palette and choose OPEN SYMBOL.
- In the form type ‘andor’ as the component Name. Symbol which represents your VHDL design will appear in the Symbol Editor window.
- Pop up : Setup ➡ Grid ➡ Report ➡ Color ➡ Choose Grid.
  - Enter 1 in the Grids per Pin box & 10 in the Major Multiple box.
- To get some space around the symbol zoom out.

Generating A Schematic Symbol

- To give the name to the symbol:
  - Add menu ➡ Properties.
- When form appears type MODEL in the new property Name box, and ‘andor’ in the Property Value box.
  - Pop up : check menu defaults.
- Pop up : File menu➡Save ➡ Default registration.
- Again check, and close the symbol editor window.

Mentor Entry Tool Renoir

Mentor Graphics Renoir

Introduction

Renoir from Mentor Graphics

Copyright Mentor Graphics Corporation 1996-1999. All rights reserved.
Unpublished, licensed software. Confidential and proprietary information
which is the property of Mentor Graphics Corporation or its licensors
© ILOG inside

Renoir is a graphical design entry tool that automatically creates synthesizable HDL code (Verilog or VHDL) from schematic blocks, state machines, flow charts, truth tables, or block diagrams for ASICs or FPGAs.

Included with Renoir is a new tool called HDL2Graphics, which will generate graphical blocks, state diagrams, or flow diagrams from Verilog or VHDL code.

Mentor Graphics Renoir™ is a next generation HDL graphical design environment that can generate Verilog and VHDL from Moore/Mealy state, flow chart, truth table and block diagrams for FPGA, ASIC and IC.

It is available on **UNIX** (SUN/HP) and **Windows NT/95/98** platforms.

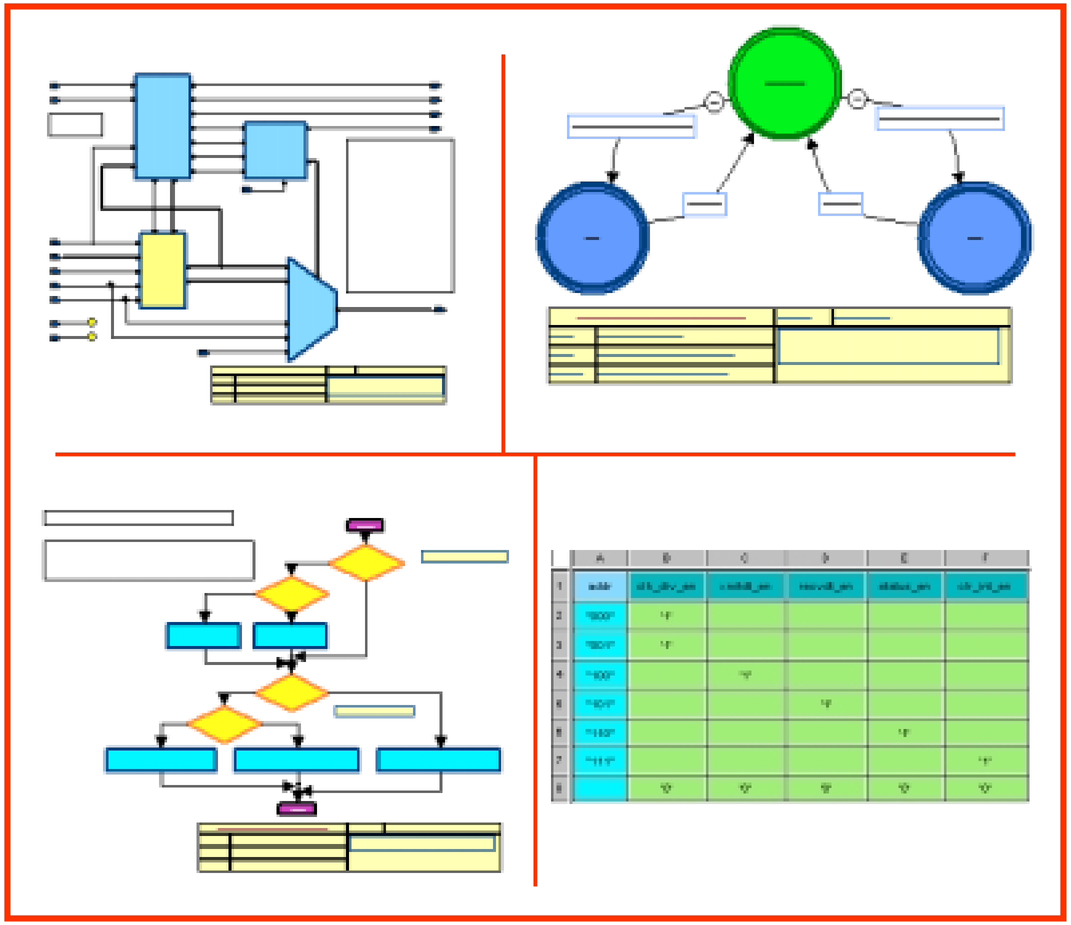
It includes interfaces to logic synthesis, digital simulation, and HW-SW co-verification tools.

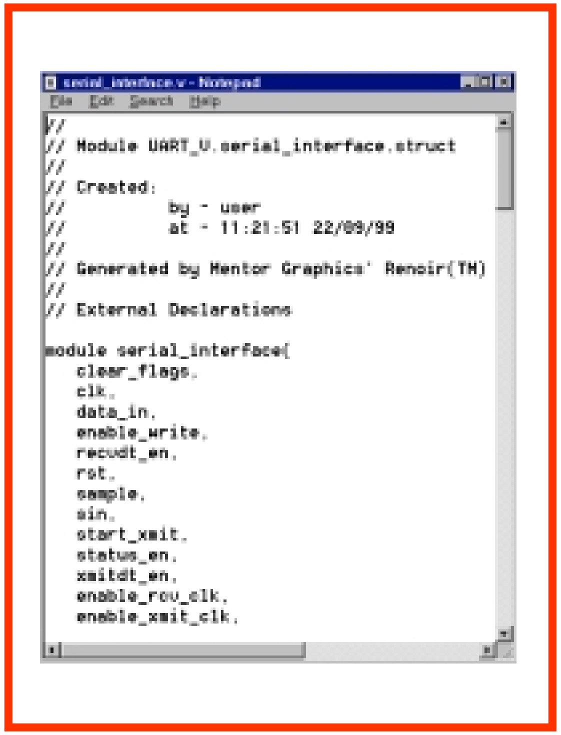
With **HDL2Graphics™** you can turn HDL text, **VHDL** and **Verilog Intellectual Property (IP)** into graphical diagrams.

HDL2Graphics™ is a no-cost option, there are no barriers to unleashing your productivity.

Renoir supports team design using **RCS** and all the major revision control environments, designers can easily manage complex HDL designs, comprising hundreds or thousands of HDL files.

Renoir

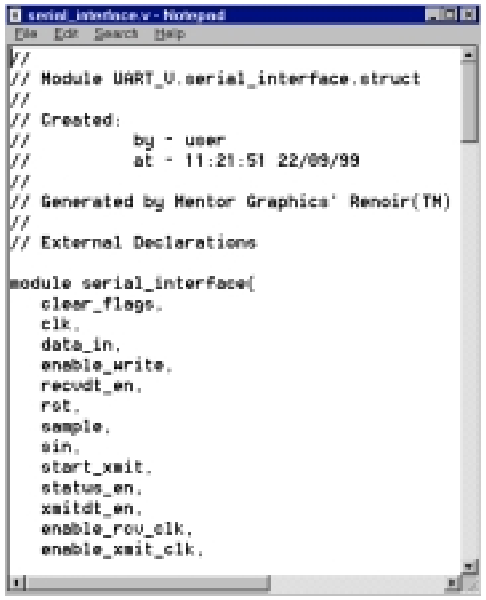




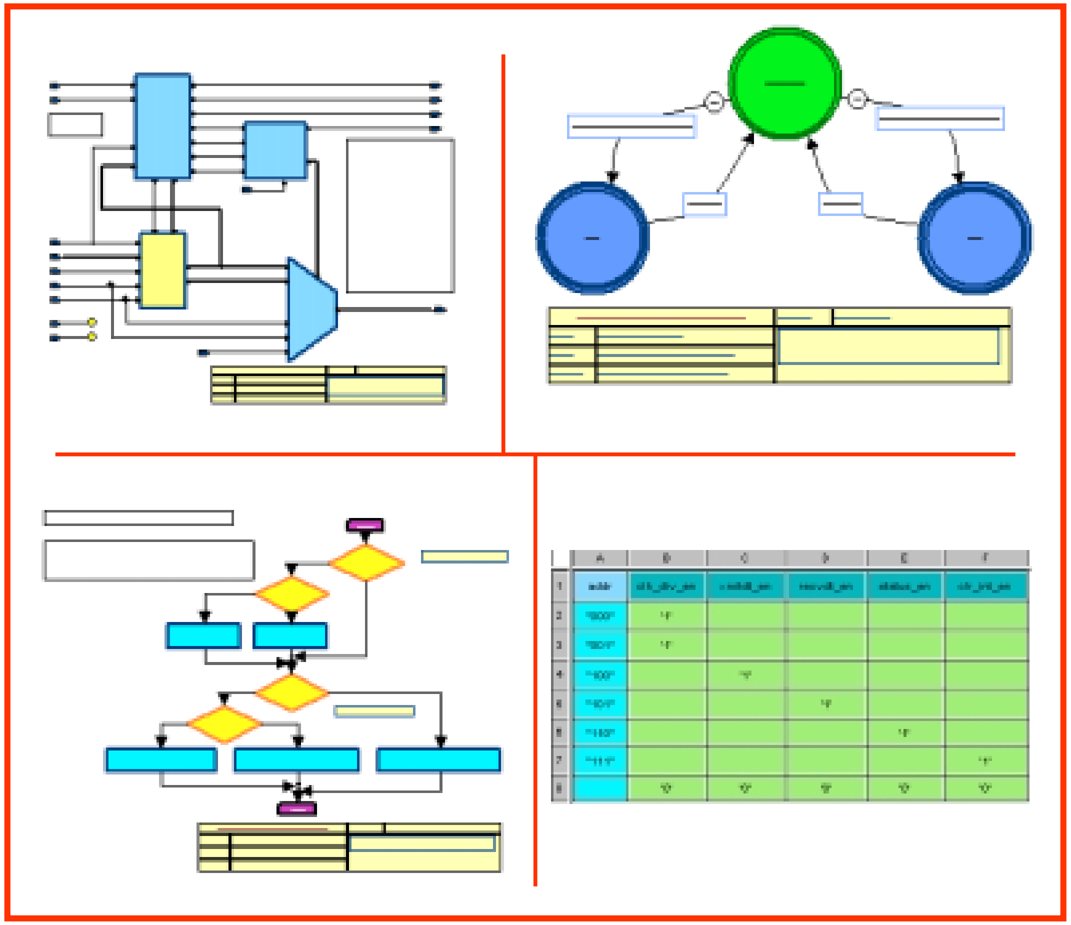
```
//
// Module UART_V.serial_interface.struct
//
// Created:
//            by - user
//            at - 11:21:51 22/09/99
//
// Generated by Mentor Graphics' Renoir(TM)
//
// External Declarations

module serial_interface(
   clear_flags,
   clk,
   data_in,
   enable_write,
   recvdt_en,
   rst,
   sample,
   sin,
   start_xmit,
   status_en,
   xmitdt_en,
   enable_rcv_clk,
   enable_xmit_clk,
```

Renoir will automatically generate HDL code directly from various graphical descriptions, providing an HDL design flow into any HDL simulation or synthesis tool that accepts Verilog or VHDL.

HDL2Graphics



```
//
// Module UART_U.serial_interface.struct
//
// Created:
//            by - user
//            at - 11:21:51 22/09/99
//
// Generated by Mentor Graphics' Renoir(TM)
//
// External Declarations

module serial_interface(
   clear_flags,
   clk,
   data_in,
   enable_write,
   recvdt_en,
   rst,
   sample,
   sin,
   start_xmit,
   status_en,
   xmitdt_en,
   enable_rcv_clk,
   enable_xmit_clk,
```



HDL2Graphics provides the capability to "reverse" the process and enter HDL code into the tool to view the state machines, truth tables, etc… that comprise your design.

Renoir

- The real power of this tool is seen when users are just beginning to design using an HDL.
- For users who are more familiar with graphical entry, this tool provides an effective method by which to transition into writing HDL code.
  - It shows the user exactly how their graphical block corresponds to HDL code.
- This is becoming more and more necessary as designs increase to over a million + gates.

Renoir

- When coupled with HDL2Graphics, Renoir becomes an invaluable tool to assist in the debugging of a large design.
- By importing and converting your Verilog or VHDL source code into graphical blocks, it becomes very easy to locate and fix bugs or errors.
- Because all graphical blocks created by HDL2Graphics are fully editable, the engineer can make changes in the graphical block, and then re-generate the HDL code.

Renoir

•Renoir is completely compatible with most major EDA software tools that support Verilog and VHDL.
•Engineers can create high-level hierarchical designs in Renoir and import them into their favorite Simulator or Synthesis tool.
  •Because of this, Renoir makes a very good addition to an engineers tool set.

•For more information on Renoir, please visit their website:

http://www.mentor.com/renoir/

•There are several self-running demos that show you how to operate Renoir and HDL2Graphics. Also available is a free “demo” copy of Renoir.
• All features of the tool are fully functional, however you will not be able to save your work.

•A complete version of HDL2Graphics is available free of charge on their website.

Xilinx Alliance & Foundation

•Xilinx sells two FPGA/CPLD place and route tools called Alliance and Foundation.

•Both tools will program any Xilinx chip, however the tools in Alliance form a subset of the ones found inside Foundation.

•The Foundation tool provides the user with design entry and synthesis tools, where the Alliance package provides interfaces with other EDA tools.

Xilinx P&R tools

- What do these tools do?
- The Xilinx tool set are essentially Place and Route tools.
- This means that they take the gate level netlist provided by another EDA tool, such as the synthesis tool Synplify, and further break the design down into blocks of logic.
- These broken down blocks of logic "map" into the standard Xilinx architectural logic block called the Configurable Logic Block, or CLB.
- Each CLB has a block of SRAM and some registers, as well as some general purpose combinatorial gates.

Xilinx P&R tools

The Xilinx tool uses it’s knowledge of the specific device that the designer has chosen, to properly place each broken down block into specific CLB’s, and route signals accordingly.

Xilinx P&R tools (cont.)

•So, the Xilinx tool Places and Routes your design so that it will fit inside your FPGA or CPLD.

•The tool also allows you to enter constraints on the design.

•For example, if a certain signal must be on a specific pin on the FPGA, you would “lock” the pin number down and associate it with the specific signal associated with that pin.

•Then, when the tool is performing it’s place and route, it will keep these requirements in mind while placing your design.

•In fact, it will try very hard to fit your design inside the chip so as to meet all constraints.

Xilinx P&R tools (cont.)

- Another constraint that can be entered is a timing constraint.
- If a certain signal has to be to a piece of logic within a certain time frame, you can enter a “timing constraint”, and the P&R tool will try to alter it’s placement and routing to ensure that you will meet your timing spec.
- If the tool fails to meet your desired timing, there are several internal tools that can help you to alter something so that your design meets spec.
- The most popular of these are:
    - Re-entrant router,
    - the Floorplanner.

Xilinx P&R tools (conclusion)

For more information on the Xilinx tool set, please visit their website.

There are many design examples and tips on how to use their tools effectively.

Http://www.xilinx.com

Sources

- J. Christiansen,
- CERN - EP/MIC
- Jorgen.Christiansen@cern.ch

- Andrew Iverson