VLSI TECHNOLOGY, INC.

ASIC SOFTWARE TOOLS DATA BOOK

ASIC Division

VLSI TECHNOLOGY, INC.

ASIC SOFTWARE TOOLS DATA BOOK

VLSI TECHNOLOGY, INC.

ASIC SOFTWARE TOOLS DATA BOOK

ASIC Division

VLSI TECHNOLOGY, INC.

CONTENTS

Introduction

Overall VLSI Tools System

General Description

Overview

The Express^{tm} ASIC Design System from VLSI Technology is an advanced set of CAE tools covering every phase of ASIC design, from conception and logic design through physical verification. Along with VLSI's extensive libraries, VLSI's Tools can be used for both gate array and cell-based designs, including any combination of standard cells, compiled blocks, Megacells, and full custom blocks.

The VLSI Tools greatly enhance design productivity, significantly decreasing the overall design time for complex and high integration ASIC's. The Tools are highly automated and simple to use, while also being highly flexible. They are therefore suitable for both advanced and novice users. The Tools' speed and efficiency allow them to handle designs ranging up to very high complexities -- over 100,000 gates, while resulting in extremely dense layouts.

The VLSI Tools are available in packages ranging in power from a logic-only design system to a full-scale ASIC CAD system. The Tools are available on a number of hardware platforms, including workstations and compute engines from Apollo, Sun, DEC, Hewlett Packard and Ridge.

The VLSI Tools also contain programs which support the automatic transfer of whole or partial designs in either direction between the VLSI Tools and a growing list of other design tools and environments. The list includes third-party CAE workstations, simulators, and hardware accelerators.

The VLSI Tools support a number of input devices, including mouses and tablets, besides keyboards.

The VLSI Tools are highly integrated, with a consistent, friendly user interface. The Tools have many general features in common, as described on the next page.

Features

Convenient User Interface

The VLSI Tools are highly integrated and have a consistent window-oriented, menu-driven user interface, making them easy to learn and use. All tools have similar commands with similar names, syntax and semantics. All windows and pop-up menus have similar graphic and command structures and have been designed to minimize the number of discrete button clicks and keystrokes.

Integrated Database

A common database with a simple abstract structure underlies all of the Tools. All pieces and types of design information are stored uniformly in objects called *cells*. Each tool can accept data in many forms, automatically converting between cell types if necessary.

Communication Between Windows

An arbitrary number of windows can be open at once, on the same or different parts of your design. A change made to a cell in one window is automatically reflected in other windows containing that cell. Changes in lower level cells are automatically reflected at the top level.

Flexible Window Manipulation

A number of convenient features exist for manipulating graphical windows. Individual windows, with work in progress, can be parked off to the side to avoid cluttering the screen, and then conveniently reopened. Windows can also be reframed or moved conveniently. Similarly, even pop-up menus can be easily moved, to avoid obscuring something you might wish to see while making selections.

Powerful Zoom

The graphical windows have a variety of powerful and convenient zoom and pan mechanisms to change your current viewing scale and location. Each window has a graphical zoom pane which shows an outline representing your current view relative to the design's total extent. You can zoom in or out by framing an area in the zoom pane, or by framing an area in the actual view. There are also commands to zoom to the bounding box, the last zoomed view, a named previous view, or an explicit absolute scale. You can pan using the zoom pane, or by specifying source and destination points in the actual view. Similarly, all text windows have common convenient scrolling mechanisms.

Flexible Command Language

Non-graphic tools have a flexible yet concise command language. Command abbreviations are recognized; you need only type enough characters to disambiguate the command. Often, you may also define macro-like aliases for commands and their parameter combinations, and any command that takes a list of node names as a parameter also takes wild card and exclude patterns. You can get help at any point by typing '?'.

Aborting Long Operations

Most operations in VLSI Tools are quick, even when used interactively. However, operations that are a bit longer by their very nature, such as the complete compaction of a full complex chip, can often be interrupted in progress without leaving the editing session, by typing a simple abort sequence.

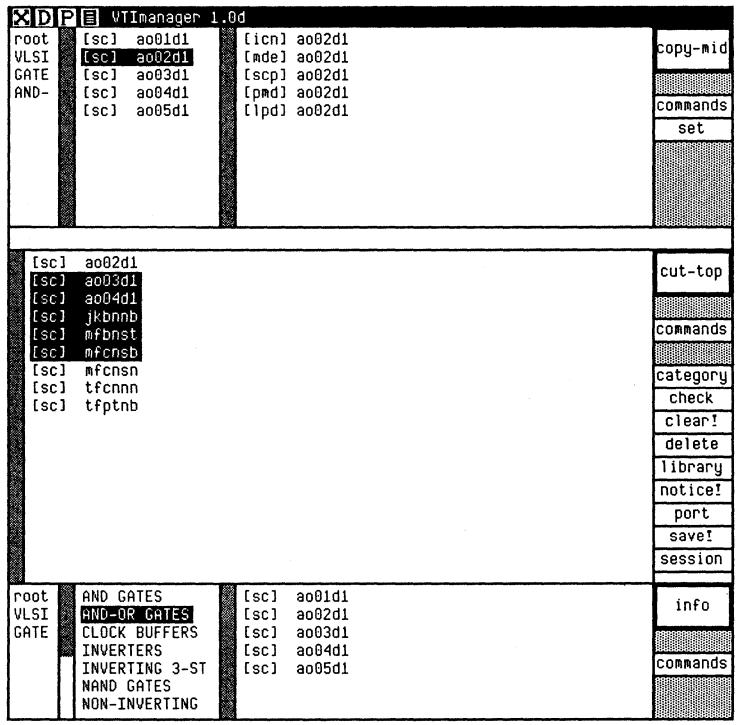
Interactive or Batch Mode

The VLSI Tools can be run graphically or in a textual shell environment, allowing them to be run interactively or with command files. Most graphic tools also have a non-graphic command interface that allows them to be accessed from the shell environment. Conversely, the VLSI shell can be accessed from the graphics environment, through VLSI's terminal emulator window.

System Support Tools

VLSI Cell Manager

Hierarchical Database Manager for IC Design Projects



Overview

VTImanager, VLSI's Cell and Project Manager, is an integral part of VLSI's tools that is responsible for all aspects of managing the design database. It consists of three parts which interact closely.

The first part is the *data manager* that underlies all of VLSI's tools. You need never interact directly with the data manager; rather, every VLSI tool interacts with it automatically on your behalf. The data manager is responsible for storing your design files in the file systems of the computers that you use, for finding these files there when they are needed, and for controlling simultaneous access to these files by other designers.

The second part is the *browser*, a pane attached to the top of every window which displays the various parts of your design for selection and use.

The third component is the *manager window*, a dedicated window that you use to control the operation of the cell manager and for major restructuring of the way that the browser presents your designs to you. Some of the commands in this window are also available directly in the browser pane in each window.

VTImanager provides extensive support for multi-person designs and individual experimentation. You can gain exclusive access to parts of a design in order to alter them, while other designers continue to use the old unchanged versions. When the alterations are completed to your satisfaction, you can release them to the other designers on the project. You do not need to get exclusive access to large parts of a design in order to alter just a small portion of it.

Component libraries can easily be shared among several designs. You select the libraries you wish to access by setting your search path. Public component libraries, which are frozen to maintain their integrity, can be stored in a particularly efficient way which uses less disk space and can be accessed in a faster manner.

Several versions of most design files are preserved. The oldest versions are deleted automatically.

VTImanager provides a simple mechanism for moving parts of, or even whole, designs to different computers, other designers or archive media. Little interaction is involved to restore the design at the destination or from archive.

Features

Structured Database

VTImanager provides a consistent, operating-system independent, abstract view of the design database. All pieces and types of design data are stored uniformly in objects called *cells*, and cells are grouped together into *libraries*, which can be shared among several designers and projects.

Search Path

A *search path* mechanism allows individual designers and projects to make use of multiple libraries, and to selectively override cells that have multiple variations.

Graphical Interface

Available cells on the search path are displayed graphically, in a *browser* pane at the top of each VLSI Tools window. The browser provides the means for cell selection in the various windows, so you don't have to type cell names. It also conveniently provides in each window a subset of the commands of the general manager window.

Hierarchical Database

The cells can be structured into hierarchical browser categories independently of the libraries or directories in which they are located, and independent of any design hierarchy.

Integrated Database

The browser pane and underlying data manager are common to all VLSI tools. VTImanager provides a unified data base and user interface for all these tools.

Manager Window

There is a dedicated VTImanager window with more extensive capabilities than available in the individual browsers. This window provides the main interface for all the housekeeping operations associated with the maintenance of libraries and setting up projects. The VTImanager window also provides a sophisticated way to reorganize browser category structures.

Cell Management

The underlying data manager keeps track of various cell attributes and manages the cells accordingly. These attributes include both static attributes, such as creation date, technology and cell type, and dynamic attributes, such as whether the cell has been modified in the current session.

Project Management

VTImanager provides a number high-level project management facilities which support multi-person designs. For example, the checkout mechanism allows you to get exclusive access to a cell to alter it, while other designers on the project continue to use the old unchanged version. Conversely, VTImanager lets you notice whether any cells have been altered by the other designers while you are in the midst of your session, so it is not necessary to lock all cells between the cell being altered and the root of the cell hierarchy.

Read-Only Libraries

Libraries which are unchanging can be stored in a particularly efficient way which uses less disk space and is accessible faster than normal. This also protects such libraries from inadvertent permanent modifications, while still allowing temporary override using the search path mechanism.

Import/ Export

The export and import mechanism allows any number of cells to be conveniently moved to another library or computer in a single file transfer. All administrative information is automatically transferred along with the cells, minimizing the amount of interaction needed to restore the design and make use of the cells at the destination.

Backups

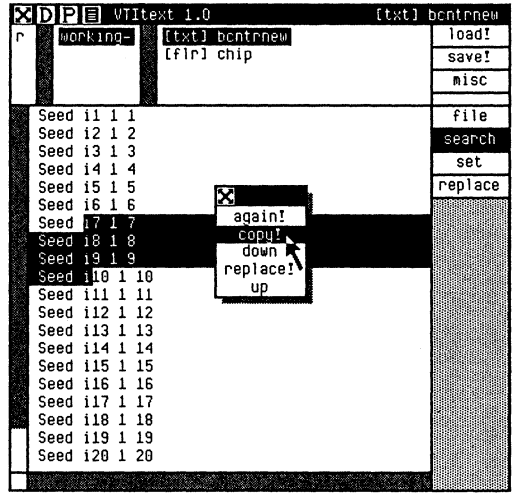
Backup versions of critical cells are kept. The number of versions kept is under user control. Versions older than that are automatically deleted as newer ones are created.

Interactive or Batch Mode

Many of the VTImanager window functions are also available in the terminal environment, in which VTImanager can be used interactively or with command files.

VLSI Text Editor

Editor for Manipulating Text Files



Overview

VTItext, VLSI's graphical Text Editor, is a screen-based text editor integrated into the VLSI Tools environment. As with all other VLSI editors, you give commands to the editor with the mouse and menus. This makes text manipulation very rapid and easy.

Features

Mouse Oriented

The mouse is used for relocating the cursor, invoking commands and moving blocks of text within the editor. The mouse can be used to manually scroll to any point in the text with just one or two clicks, instead of many repeated keystrokes. This makes the text editor much easier to use and faster than conventional keyboard-based text editors.

Tools Integration

VTItext is fully integrated into the VLSI Tools environment. This allows you to access the common VLSI Tools browser database and quickly and easily edit any text files produced or used by other tools without leaving the Tools environment.

Flexible Input and Output

You can read either cells from VTItext's browser or standard text files from outside, and write them back as either cells or files, in any combination. This feature can be used to move files into a VTImanager library, or to move cells from a library to an outside directory.

Flexible 2D Scrolling

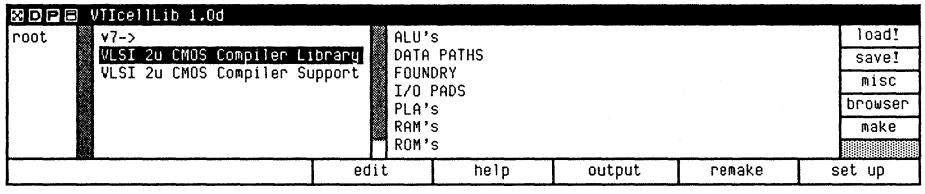
Unlike most text editors, VTItext offers convenient horizontal scrolling and arbitrary line widths, in addition to vertical scrolling and unlimited length. Besides manual scrolling, automatic horizontal and vertical scrolling occurs to keep the cursor visible.

Auto Wrap-around

You can set VTItext to automatically begin a new line at the closest preceding word break when the line you are typing gets too long.

VLSI Cell Library Window

Cell Library Compiler Interface and Parameter Editor



Overview

VTIcellLib, VLSI's Cell Library Window, is the compilation interface to VLSI's libraries, which include VLSI's high-level Datapath and State Machine Compilers, as well as VLSI's Module Compiler Library. VTIcellLib is used to specify parameter values and generate various forms of compiled data for these libraries. Compiled data includes physical layout, gate-level netlists, or schematic symbols with behavioral models. In some cases, the parameters include a reference to a higher-level specification.

VTIcellLib can also be used to specify parameters and generate schematic symbols for user-written behavioral models.

Features

Saved Values

VTIcellLib enables you to save and recall sets of parameter values for later use or further editing.

Menu Interface

You select a parameter graphically from a menu. If the parameter has a discrete set of possible values, then those values are also graphically selected. Otherwise, values may be typed in.

Default Values

Parameters have default values, to quicken value specification.

Range Checking

Parameters with a range of values are immediately range-checked upon value entry, prior to compilation.

Hierarchical Parameters

Certain parameters are displayed for value specification only when they are relevant, depending on the value of other parameters.

Compiled Output

Once parameter values have been specified, several forms of compiled output data can be generated. These include:

- **Layout and Phantom.** Optimized high-density layout in the form of CIF (CalTech Intermediate Format) can be generated. A phantom cell, which lacks the internal geometry, is also generated. The phantom cell enables quicker verification of the higher-level design by VTIextract, VTInetCompare and VTIdrc, because the internal, compiled geometry is already known to be correct.

- **Portable Netlist.** A netlist of primitive cells from VLSI's Portable Library can be generated in the case of many compilers, for implementation in a gate array or standard cells in a variety of process technologies.

- **Simulation Netlist.** A gate-level simulation netlist for modeling the compiled layout can be generated in the case of very complex compilers.

- **Model Schematic.** A schematic symbol can be quickly generated for either a compiler or a user-written behavioral model. In either case, the symbol represents the combination of a behavioral model with the specified parameter values.

Quick Check

The input descriptions for VLSI's high-level compilers, which include the Datapath Compiler and State Machine Compiler, can be checked for proper format or syntax without invoking a full compilation.

Area Estimate

A quick estimate of compiled layout area, along with other information, can be produced for complex compilers, without invoking a full compilation.

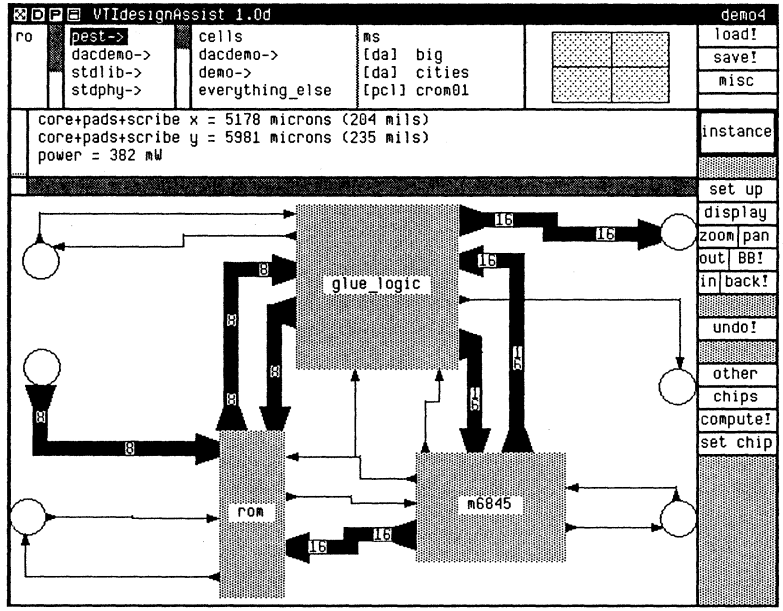
Interactive or Batch Mode

You may use VTIcellLib interactively or with command files, in either the graphics or terminal environments. VTIcellLib can generate layout immediately or produce a command file to do so later.

Logic Design and Analysis Tools

VLSI Design Assistant

System Partitioning and Design Evaluation Tool



Overview

VTIdesignAssist, VLSI's graphical Design Assistant, is an expert tool for system partitioning and design evaluation. It may be used for high-level "what-if" considerations at the very early stage of a design. It analyzes high-level trade-offs involved in partitioning a design and choosing an implementation technology, to ensure the optimal design approach and minimize system cost.

For example, the Design Assistant can help answer such questions as:

- Should I use a Megacell or a standard part?
- If the ROM is 1K larger, will the chip still fit in an inexpensive plastic package?
- Will multiplexing the data bus make a difference or is the design already core-limited?
- How much power will the chip dissipate?
- Will the design fit in a 10K gate-array?

VTIdesignAssist evaluates ASIC alternatives from a rough block diagram of the design. The diagram contains blocks representing whatever level of implementation detail is known, such as simple gate count, a list of TTL parts or other cells, a captured netlist, compiler specifications or actual layout. From this data, VTIdesignAssist enumerates the various implementation alternatives for each chip and estimates several characteristics of the final chip for each alternative, including die size and power consumption. It also recommends the optimum package in each specified package family.

VTIdesignAssist's area analysis takes into account the known or estimated size of leaf cells, compiled blocks, gate array macros, gate array bases and so forth. In addition, it takes into account estimated routing, including standard-cell routing, inter-block routing and routing from the core to the pads. It also automatically distinguishes between core-limited and pad-limited designs, and takes the appropriate pad set into account.

The power consumption estimate takes into account operating frequency of both the pad and the core, and other factors. Besides number of pins, the package choice takes into account other factors such as die size and power dissipation.

VTIdesignAssist aids in the partitioning process at both the block and chip levels. Each block can be assigned to any of the chips in the chip set or, like a standard part, none at all. VTIdesignAssist automatically adjusts its estimates, including die size, to take into account pad requirements for inter-chip versus on-chip signals.

The analysis performed by VTIdesignAssist is driven from a user-extendable set of rules that describes the available technologies, cell libraries, routing approximation algorithms, decision criteria, and so on.

Features

System Partitioning

VTIdesignAssist aids in the system partitioning process at both the block and chip levels. For multi-chip designs, VTIdesignAssist automatically tracks pin requirements for inter-chip signals as components and signals are moved during the analysis. This includes an estimate of the number of power and ground pins needed.

Complex Analysis

VTIdesignAssist's analysis considers standard cells, compilers, megacells, gate array macro's and bases, standard cell and interblock routing, pad routing, gate array occupancy, choice of pad set, operating frequency, and available packages. VTIdesignAssist lists the possible implementation alternatives for each chip, along with close estimates of die size, power consumption, and optimum packages for each alternative.

Interactive Block Diagrams

VTIdesignAssist works from generalized block diagrams and so allows the analysis of very high-level trade-offs. The diagram is entered in a very interactive way that is consistent with VTIschematic and makes changes very easy, so that different design configurations can be analyzed.

Graphical Feedback	Blocks are displayed dynamically in their relative estimated size on the chip. This gives immediate floorplanning feedback.
Variety of Input	Design feasibility can be explored using whatever level of implementation detail is available. Input could be as simple as gate count, a list of TTL parts or other cells, or a captured netlist with instances and incomplete wiring. For yet greater accuracy, input could be as detailed as a complete netlist, compiler specification, or physical layout.
Technology Independence	All the technology-dependent information is contained within a machine-independent collection of rules containing algorithmic information as well as list-oriented data.
Extendibility	The rule file can be tailored to your own needs or augmented for other available technologies.
Schematic Output	VTIdesignAssist writes out top-level schematics for each chip, including all the pads and any necessary drivers or level-shifters. It also writes a schematic template for each block, including the connectors to the block, thus facilitating top-down design.

Documentation

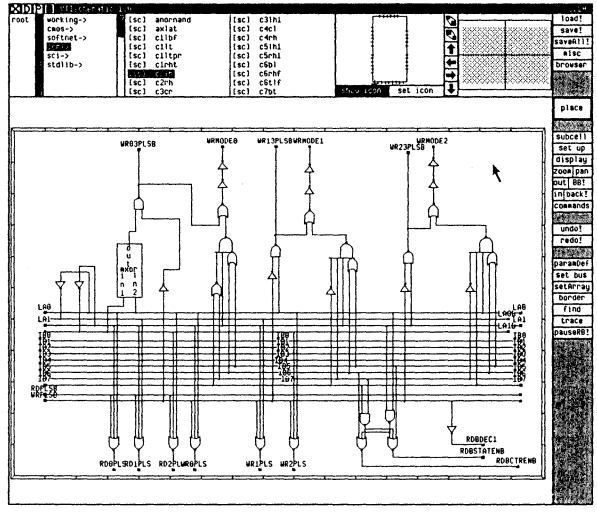
The block diagram can be plotted, and the results of the analysis are available in hardcopy, plain or as a quote form, as well as on-screen.

Undo

Most commands in VTIdesignAssist can be undone.

VLSI Schematic Editor

High Level Design Entry Program



Overview

VTIschematic, VLSI's graphical Schematic Editor, is a tool for front-end logic capture that provides you with the ability to create, view, edit and plot schematic diagrams. VTIschematic is a hierarchical tool, enabling you to create schematics that range from a complete chip description to the the gate or transistor level. Within VTIschematic, you can place and connect cells from any of VLSI's libraries, as well as behavioral models created using VTImodel. You can also place and connect standard logic gates or new schematics which you've built from a combination of any of these components.

VTIschematic diagrams provide input to many other VLSI tools, including those used for simulation, timing verification, chip compilation and netcompare.

Features

Hierarchy

With VTIschematic, you can place schematic cell instances in schematic cells, building a hierarchy to describe the final schematic. In this way, VTIschematic, along with the companion tool VTIicon, supports both top-down and bottom-up design styles along with a combination of either.

Parameters

Electrical parameters can be defined for any schematic, at any level. For example, many cells in VLSI's libraries have parameters handled in this way. Some behavioral model parameters also show up as schematic parameters. Parameter values for each instance of a schematic can then be set with fixed values or they may be inherited from higher level schematics.

Push/Pop

You can easily push down into a schematic subcell and return, to conveniently traverse the schematic hierarchy for editing. Changes to lower level schematic cells are automatically reflected at the top level.

Powerful Commands

VTIschematic provides a variety of powerful editing commands. These include an intelligent move command which retains existing connections, as well as commands to cut and paste whole regions.

Buses

VTIschematic supports buses, along with bus breakouts and bus connections.

Arrays

VTIschematic lets you conveniently and flexibly place arrays of instances or regions.

Dynamic Feedback

VTIschematic's user interface gives instantaneous graphical feedback, such as rubber-banding, for pending operations.

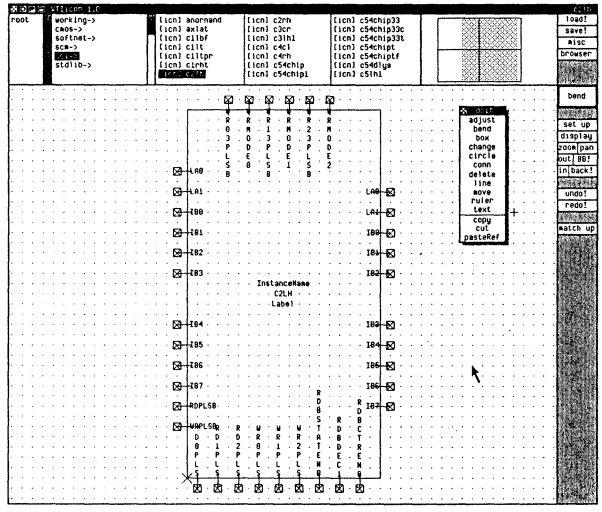
Immediate Checking

VTIschematic performs edit-time checks to prevent accidental shorting of incompatible or conflicting signals or buses.

Simulation Feedback	VTIschematic is tightly coupled with VTIsim, for convenient circuit debugging. Nodes or instances selected in VTIsim can automatically be found and highlighted in VTIschematic.
Text Annotation	Comments and general text annotation may be added to any part of a schematic to clarify the design.
Undo	The last several commands can be undone or redone.
Plots	VTIschematic generates plot files for Versatec raster plotters and Hewlett-Packard pen plotters.
Automatic Icon Generation	VTIschematic automatically generates icons to represent schematic instances in higher level schematics. These icons may be customized with VTIicon.
Interactive or Batch Mode	You may use VTIschematic interactively or with command files, in either the graphics or terminal environments.

VLSI Icon Editor

A Tool for Editing Schematic Symbols



Overview

VTIicon, VLSI's graphical Icon Editor, provides you with the ability to create, view, edit and plot schematic symbols, termed *icons*, for VTIschematic. You can create icons from scratch, or customize icons that the VLSI Tools automatically create, so that your schematics use familiar symbols and notations. VTIicon can also be used as a general-purpose graphic drawing tool.

Features

Powerful Commands

VTIicon provides a variety of powerful editing commands. Arcs, circles, lines, rectangles and other objects are easily created and manipulated. Whole areas can be cut and pasted.

Text Annotation

Comments and general text annotation may be added to any part of an icon to clarify the design. Several specialized text fields are also available which can be changed in VTIschematic.

Undo

The last several commands can be undone or redone.

Plotting

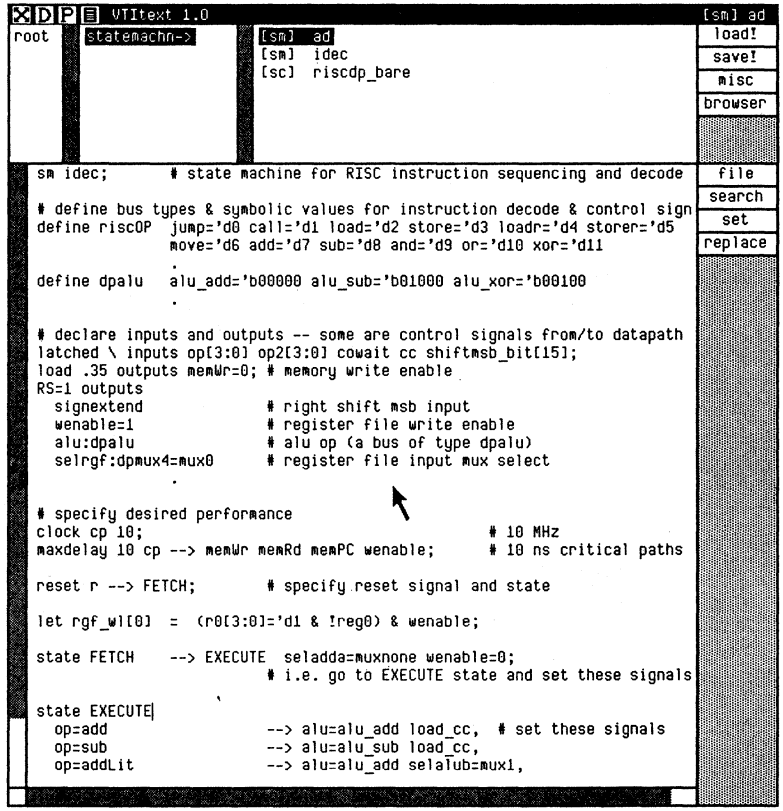
VTIicon generates plot files for Versatec raster plotters and Hewlett-Packard pen plotters.

Automatic Schematic Generation

VTIicon can generate an empty schematic from a finished icon. This allows the designer to draw a schematic using this icon before any of the sub-blocks have been implemented, facilitating a top-down design.

VLSI State Machine Compiler

ASIC Logic Synthesizer and Optimizer



Overview

VLSI's State Machine Compiler, a logic synthesizer for ASIC's, is used to design, implement and optimize state machines and random logic. A state machine is a block of logic whose outputs (and next state) at any one time depend not only on the inputs at that time, but also on its current internal stored state.

State machines can be used for sequencers, controllers, or just pure combinational logic, such as decoders, with no stored state at all. They can also be used as on-chip PLD (Programmable Logic Device) replacements, for high integration.

The State Machine Compiler takes a flexible yet simple high-level language description of the desired logic and performs logic synthesis and optimization within user-specified performance constraints. It efficiently translates the high-level description to a netlist of primitives from VLSI's Portable Library, for implementation in a gate array or standard cells. It can also generate a PLA (Progammable Logic Array) code file for input to VLSI's PLA compiler. The State Machine Compiler aggressively optimizes the logic taking into account the available library and target implementation, using advanced optimization techniques.

The state machine description language is very readable, convenient and easy to use. For each relevant combination of inputs in each state, the language description specifies the desired outputs and next state, using high-level equations, buses and symbolic constants. The language and resulting state machines that can be produced are very flexible, with many useful options.

You access the State Machine Compiler through VTIcellLib or a special interface to the Compiler's PLA Optimizer. You then simulate the Compiler's output in VTIsim, using the behavioral or gate-level models

supplied with VLSI's Portable Library or PLA Compiler. The Compiler also includes a high-level behavioral model, so you can simulate your state machine directly from its description, without actually making the final netlist or code file, in order to quickly check out that description.

Features

Wide Applicability

The State Machine Compiler can be used for a wide variety of applications, including any type of control or glue logic, with or without internal state. It can implement the most general forms of state machines and can even implement circuits beyond those normally thought of as state machines, such as pure combinational logic. It can also be used for PLD replacement.

Datapath

The State Machine Compiler smoothly complements VLSI's Datapath Compiler when used to generate datapath control and decode logic, making use of special provisions for that purpose. (See VLSI's separate Datapath Compiler Datasheet for more information on the Datapath Compiler.)

High-Level Language Input

The State Machine Compiler accepts input in the form of a high-level language description. The language is very flexible, with many features:

- **High-Level Equations.** You describe combinational outputs and transitions between states using high-level equations.
- **Buses and Symbolic Names.** The inputs and outputs can be buses. You can use symbolic constants for bus values, and various operations such as comparison or assignment are allowed on such buses and constants.

Similarly, states are represented by symbolic names rather than actual bit codes. Such high-level symbolic names contribute to readability, ease of description, consistency and correctness.

- **Flexible Performance Constraints.** In the state description, you may specify performance requirements in the form of a maximum operating frequency or individual delays for one or more critical paths. You can specify maximum delays from inputs to outputs, from inputs to the state register (the setup times for the machine), from the state register to outputs, or from state register to state register (the operating frequency). To ensure meeting the performance requirements in the actual system, information about external circuit characteristics, such as how strongly the inputs are driven and how heavily loaded the outputs are, may also be specified.

- **Flexible Clocking.** You can have the machine change state on the rising or falling edge of the clock. You can also put latches on inputs and outputs, again clocked on either edge, to further control the timing of the machine's signals.

- **Flexible Reset.** You can specify which state to go to on reset, the initial values of the input and output latches on reset, and whether reset is active high or low.

- **Simple Description.** There are several options which can be used to further simplify the state machine description by minimizing the number of outputs whose values must explicitly be specified for each transition. For example, default values can be declared for outputs, whose values then need only be specified when they differ

from default. Outputs can also have RS flip-flops, with the effect that their values need only be specified when they change.

- **Don't Cares.** To enhance the quality of the optimization, don't care conditions can be explicitly specified.

- **Intermediate Equations.** To simplify the description, you can give intermediate logic equations for signals that are only used inside the machine, such as those that are fanned out to many places.

Portability and Technology Independence

The State Machine Compiler can be used for either gate array or cell-based design, in a variety of technologies. From a single high-level description, output is available as a netlist of Portable primitives for implementation in a gate array or as standard cells, or as a custom-density layout block in the form of a PLA. The high-level description is therefore technology-independent. It is easily migrated between implementation styles and process technologies.

Netlist Compilation and Advanced Optimization

When compiling a netlist, the State Machine Compiler Aggressively optimizes the logic for the target Portable library using several advanced techniques. Among them are

- PLA-like optimization,

- factorization into a multi-level network, and

- application of library-dependent transformation rules, which ensures use of the most efficient gates available in the target library.

The Compiler optimizes first with regard to speed, to meet your specifications, and then area. The resulting implementation is comparable to and often better than aggressively hand-optimized circuits, and is done in a fraction of the time, without error.

PLA Output and Optimization

The Compiler has a PLA Optimizer which can take a state machine description and produce an optimized PLA code file to implement the combinational logic for the state machine. VLSI's PLA Optimizer can also take as input an independently-generated bare PLA code file.

Automatic State Assignment

The State Machine Compiler performs automatic state assignment (the assignment of bit codes to represent each state in the state register), using a heuristic for further optimization. The states can also be manually assigned if desired.

High-Level Model

The State Machine Compiler includes a high-level behavioral model which serves as an interpretive facility for state machine emulation and efficient simulation. You can directly simulate your state machine in VTIsim from its description, without actually synthesizing and optimizing the logic. This is useful up-front during the design process for quickly checking out the description. Special debug features are available, such as feedback using the symbolic names. For example, the model tells the name of the state it is in at any one instant.

VLSI Screener

Netlist Screener and Design Review Tool

Overview

VTIscreen, VLSI's Netlist Screener, analyzes your logic design and identifies gate or block-level electrical rule violations, for both gate array and standard cell designs. It also provides several design summary reports, including a utilization report for gate array designs.

VTIscreen can be run any time after the schematic netlist is created. You run it at least once before submitting a gate array design for placement and routing. You also run VTIscreen routinely earlier during the design process in order to keep track of your array utilization and to catch any errors as soon as possible. VTIscreen identifies connectivity errors and other potential problems.

VTIscreen also analyzes wire lengths and generates a back annotation file with either pre-route predictive capacitances or post-route actual capacitances. This file can then be used for back annotation in simulation or timing verification. A report comparing the actual to predicted capacitances is also produced.

Finally, VTIscreen interfaces to VTIvector, its companion vector screener and translator, by producing a file containing logical I/O signal information needed by that tool.

Features

Summary Reports

VTIscreen provides several summary reports. These include design statistic and complexity reports, containing information on number of interconnects, cell instances, gate equivalents, and so forth, as well as a gate array utilization report on pad and internal cell utilization, VTIscreen saves the reports in a disk file as well as printing them on-screen.

Checking

VTIscreen checks for certain connectivity errors and other potential problems in your design, such as oddly or dangerously connected or otherwise misused inputs, outputs, parallel cells, clock buffers, pad drivers and level shifters.

Back Annotation

VTIscreen has a companion program that analyzes wire lengths and generates a back annotation file in a variety of formats, with either pre-route predictive capacitances or post-route actual capacitances. The program also produces a report comparing the actual to predicted capacitances.

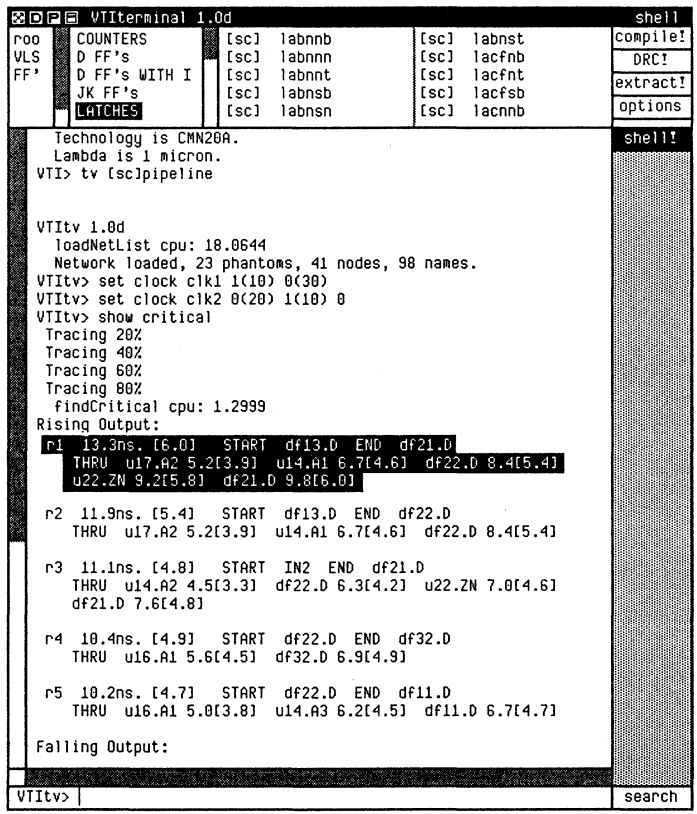
Interactive or Batch Mode

You may use VTIscreen interactively or with command files, in either the graphics or terminal environments, from VLSI's text shell or terminal emulator window.

Logic and Timing Verification Tools

VLSI Timing Verifier

Static Timing Analyzer



Overview

VTItv, VLSI's Timing Verifier, is a gate or block-level static timing analyzer for synchronous circuits. It is useful for a system designer who is designing a performance-critical circuit which is complex enough that verifying all timing purely using simulation would be difficult. No knowledge of transistor-level design is required. VTItv analyzes either the entire design at once, for global system checking, or certain paths as defined by the user, for more specific analysis.

VTItv does a static timing analysis. No input or case vectors are needed and no assertions need be added to the circuit for the analysis.

VTItv works with synchronous systems, systems that are composed of alternating stages of storage elements and combinational logic. For simplicity, system clocks are explicitly declared rather than derived.

VTItv is typically used to find and rectify potential timing problems after the overall functionality of the design has been verified with logic simulation. Identified critical nodes and paths can then be examined more closely during timing simulation, using actual data. These paths can also be weighted during physical design, to minimize wiring capacitances. VTItv assists you in filtering your design, so you can focus your attention on the most critical parts of the circuit.

VTItv can also be used to quickly re-verify designs which are ready for fabrication or which have been changed to a new technology. This is particularly useful when switching among gate arrays and cell-based technologies using VLSI's Portable Library.

Features

Flexible Analysis

VTItv performs a flexible, extensive variety of analyses, on either the entire design at once or paths that you specify. These analyses include:

- **Critical Paths.** VTItv identifies the n most critical paths in your circuit, and reports the slack in non-critical paths.

- **Cycle Time.** VTItv identifies the minimum possible period for any clock, or the maximum operating frequency for your overall circuit, accurate to within a tolerance you specify.

- **Delay.** VTItv reports the delay along any particular path you specify. Delays waiting for clock transitions are taken into account.

- **Longest Path.** VTItv identifies the n "worst" paths from input to output of the entire circuit -- the paths with the greatest delay, that may limit the speed of the circuit and contribute to a critical path, or the paths with the least delay, which are used to check for hold time violations. Delays waiting for clock transitions are taken into account.

Checking

VTItv performs a variety of additional checks, including:

- **Delay.** VTItv checks that the delay along a given path is within the tolerance you specify.
- **Setup and Hold.** VTItv checks for any setup and/or hold time violations.
- **Clock Skew.** VTItv checks that the skew between any two clock signals is within the tolerance you specify.

Separate Interconnect Delays

Interconnect delays are broken out and separately reported from total delays (which are also made up of intrinsic gate delays). This allows you to see the delay over which you have some control.

Back Annotation

VTItv accepts a netlist containing interconnect capacitances, or the capacitances for a previously loaded netlist can be explicitly separately given. Thus, the capacitances can be independently loaded from a file produced by other VLSI tools, for back annotation of predicted or actual routing capacitances.

Separate Rise and Fall Times

VTItv distinguishes between rising and falling signals, with separate delays.

Node States

Node states can be specified to be stable, rising or falling only, besides the usual changing both ways, thus causing irrelevant circuit transitions to be ignored. This is useful for eliminating from consideration impossible paths, or little-used paths which result from reset or similar signals. It can also be used for performing a case analysis.

Input Transition Times

You can set the relative transition times of circuit inputs. VTItv reports the input transition times necessary to satisfy setup and hold constraints.

Derating Factors

You can specify a derating factor for VTItv to apply to all delays, so you may examine the effect of different process variables and operating conditions on system timing.

Path Names

VTItv lets you assign meaningful names or other abbreviations to paths, clock edges, and commands. This makes your input more convenient to enter, and makes your output more readily understandable. New paths that VTItv lists out in response to a command are automatically assigned names, so you can conveniently reference them in subsequent commands without entering the entire path name.

Easy to Learn

VTItv has a command syntax similar to that of VTIsim.

Wild Cards

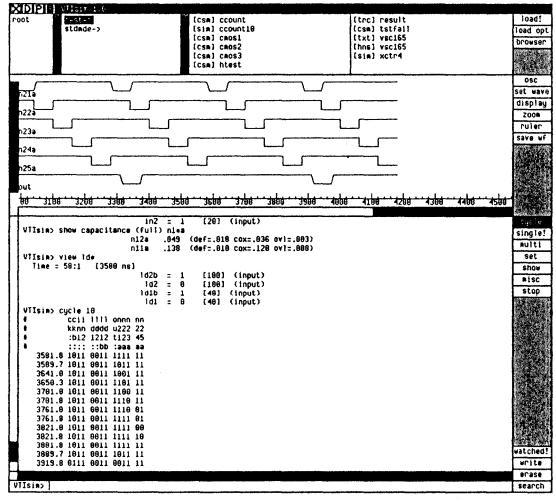
Any command that takes a list of node names as a parameter also takes wild card and exclude patterns.

Interactive or Batch Mode

You may use VTItv interactively or with command files, in either the graphics or terminal environments, from VLSI's text shell or terminal emulator window.

VLSI Simulator

Logic and Timing Simulator



Overview

VTIsim, VLSI's Logic Simulator, is an interactive, mixed-mode logic and timing simulator, running in the VLSI Tools window or terminal environments. Circuits to be simulated may contain a mixture of transistor-level, gate-level and behavioral-level components.

VTIsim takes as input circuit descriptions produced by other VLSI tools, such as VTIschematic, VTIextract, VTIchipComp, VTIcompose and VTIsticks. It also takes externally generated netlists that have been written directly in VLSI format or that have been converted to that format using VTIexchange. The level of representation to be used for each circuit component may be altered using switches produced by VTIswitch. Behavioral-level components are described in the VTImodel modeling language. Input vectors and commands can be provided interactively or by drivers, either user-written or produced by VTItest or VTIvector.

Output is produced both in text format and as a waveform display. Waveforms may be viewed during simulation, plotted, or saved for later redisplay.

Node states are represented internally as continuous voltages (in tenths of a volt) and are displayed to the user in terms of four logic values: `high`, `low`, `intermediate` and `unknown`. Behavioral models may also drive nodes with three output pin strengths (normal, weak and 3-stated). For transistor logic, the notion of strength is replaced by real-valued resistances, allowing in effect a continuous gradation of strengths.

At the transistor level, both CMOS and HMOS technologies are supported. VTIsim models each transistor as a resistive switch, which is either off or turned on to some degree which depends on the voltages on its terminals. This resistive model is used to compute the voltage at each node, and, together with node capacitances, to estimate delays.

Features

Input from Other Tools

Circuit descriptions may be those produced by VTIschematic, VTIcompose, VTIchipComp, VTIextract, VTIsticks and VTImodel, or they may be externally generated netlists written in VLSI format or converted to that format using VTIexchange.

MOS Technology

VTIsim supports MOS technologies: CMOS or HMOS, VLSI process or user-defined.

Mixed Mode

Circuit descriptions may contain a mixture of behavioral models, gates and transistors. Individual portions of the circuit can be switched at load-time among these levels of representation.

Timing or Logic Mode

VTIsim offers a choice of delay calculations. Delays can be based on a very accurate detailed timing model, or on a simpler unit-delay-like model. In either mode, explicit rise and fall delays may be assigned to individual nodes.

Universal Timing Model

Transistors, gates and behavioral models share a detailed timing model. Node transitions are ramps or steps. Ramped nodes rise over a period of time. A ramp in progress may be interrupted by new events, so glitches or other temporary fluctuations of node voltages appear clearly.

Time Range

The event queue has a resolution of .1 nanosecond and a range greater than 100 hours.

Multi Drivers

VTIsim resolves contention between multiple outputs writing to a single node.

Transistor Simulation

VTIsim simulates at the switch level with functional and timing results.

- Node states are represented internally as continuous voltages (in tenths of a volt) and are displayed to the user in terms of four logic values: `high`, `low`, `intermediate` and `unknown`. The user may specify threshold voltages for high and low logic values on a per-node basis.
- Transistors are modeled as a resistive net, thus enabling the resolution of multiple drivers.
- In timing mode, accurate delay estimates are based on transistor sizes, actual gate voltages, and node capacitances.

- A threshold-drop computation in conjunction with the resistive net model detects nodes not driven to good logic levels.

- Charge-sharing. Stored charge may alter the state of adjacent nodes, including nodes which are driven by circuit components.

Gate-Level Simulation

Logic gates can be represented by simulator gate primitives, for efficient simulation.

- Gate primitives are available for simple logic gates (inverter, AND, NAND, OR, NOR, XOR, XNOR) having up to seven inputs.

- The gate primitives model pin capacitances and load-dependent rise and fall delays.

- Gate simulation is based on three logic values: `high`, `low` and `unknown`.

Behavioral Models

Behavioral models are written in the VTImodel behavioral modeling language.

- Behavioral models are supplied with VLSI's cell libraries. You may also write your own models using VTImodel.

- Behavioral models can model timing very accurately, including propagation delay, ramps, spike filtering (inertial delay), and setup and hold checking.

- Behavioral model output drives are modeled in terms of three strengths (normal, weak and 3-stated), thus enabling the resolution of multiple drivers.

- Behavioral models can transfer high-level data (strings and numerical), aiding top-down design.

Command Interface

The VTIsim command interface has the following features:

- **Flexible Input.** VTIsim accepts graphical or keyboard command input.

- **Command Files.** Stimuli may be provided interactively or by a driver in the form of a command file. Multiple stimulus files may be loaded in parallel, with control alternating back and forth.

- **Simulation Driver Modules.** For ultimate flexibility, the simulator may also be driven by a user-written program in the Mainsail language.

- **Wild Cards.** Any command that takes a list of node names as a parameter also takes wild card and exclude patterns.

- **Breakpoints.** You may tell the simulator to halt when a specified node changes to specified value. You have the choice of halting immediately at that time or at the end of the then current clock phase.

- **Interactive Examination.** The simulation may be paused at any time and the status of nodes may be set or examined, including nodes which are currently ramping. The network structure may also be explored.

- **Flexible Reporting.** Logic values of specified watched nodes may be reported according to a variety of trace formats and frequencies. They may be reported tabularly or individually, at the end of every clock interval or dynamically whenever they change.

- **Output Testing.** Logic values can be dynamically compared to expected results.

- **Flexible Clocks.** Any number of independent clocks (signals with repeated patterns) may be defined, with varying phase and cycle times.

- **Vectors.** An arbitrary list of nodes can be defined as a vector, facilitating input and display of data. For each vector, you have choice of binary, octal, decimal or hexadecimal format.

Toggle Reporting

VTIsim can report which nodes or percentage thereof have not toggled, thus providing a form of test grading.

Static Checks

Commands check for unused inputs, undeclared outputs, nodes that cannot be driven high or low, nodes with more than one driver, undeclared power nodes, self-gated transistors, wells not connected to power, and other dangerous conditions or electrical rule violations.

Save and Restore

Simulator state may be saved (checkpointed) at any time, and later restored to this state, either in the same simulation or later. Saved data includes the state of all nodes and internal states of functional models.

Interactive or Batch Mode

VTIsim can be run interactively or in batch mode, in either the graphics or terminal environments. You may issue commands and set input values interactively at each step, or have the commands be provided by a driver. In batch mode you have the option to create a waveform history for subsequent interactive display.

Interactive Circuit Modification

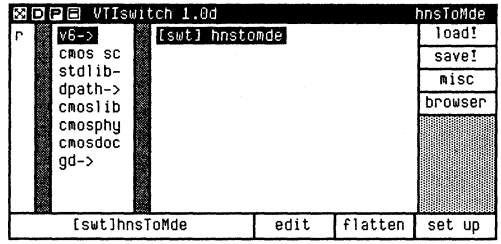
You may make certain circuit changes, such as adding a new component, changing the capacitance of a node, or setting an explicit delay on a node, at any time. For example, this is useful for back annotation of a netlist with independent predicted or actual routing capacitances.

Graphical Waveform Feedback

Output can be viewed graphically, as oscilloscope-like waveforms, or textually, in several trace modes and formats. Waveforms can be displayed (and measured) graphically during the simulation, they can be saved and recalled for later viewing, or they can be derived from a trace file. Waveforms can also be plotted.

VLSI Switch

Hierarchical Netlist Substitution Tool



Overview

VTIswitch, VLSI's Switch Window, provides you with the ability to create, view, edit and save switches. A *switch* is a specification of one or more automatic substitutions of one type of cell or netlist for another. When you load, expand or translate hierarchical netlists, you use switches to:

- Substitute the netlist of one implementation of a cell for the netlist of another implementation or another cell.
- Substitute a behavioral model in place of an expanded netlist.
- Control the depth to which the hierarchy is expanded, or "flattened".

You use a switch when you load a netlist in VTIsim or some other tool, or when you write a VLSI netlist using VTIexchange. You can also explicitly flatten a netlist directly within VTIswitch.

Switches allow you to change the result of the substitutions independent of the source netlist, without having to edit the source of that netlist. You can make any of these substitutions on an instance-by-instance basis, or for all instances of a cell.

Features

Netlist Substitution	You can substitute one netlist for another. The substituted netlist can be from a different implementation of the same cell, including a different type of implementation done in a different VLSI Tool, or it can be a netlist from a different cell.
Gate-Level Netlist Substitution	You can substitute a netlist of gate level primitives for another netlist, or vice versa. You can also substitute a transistor netlist for a gate-level primitive.
Behavioral Model Substitution	You can substitute a behavioral model for a netlist or vice versa.
Substitution for Individual Instances	You can substitute for all instances of specific cells, or only for selected instances of those cells, optionally using wild cards.

Control Over Depth of Flattening

You can direct the flattener to stop expanding when it encounters specific cells, for either all or selected instances of those cells.

Implicit or Explicit Flattening

Netlists may be flattened automatically upon loading into VTIsim or some other tool, or they may be flattened directly in VTIswitch, either way under the control of switches.

Source Independent

VTIswitch allows you to make substitutions without having to edit the source of a netlist.

Interactive or Batch Mode

You may create and edit switches interactively or with command files, in either the graphics or terminal environments.

VTImodel

Behavioral Modeling Language

Overview

VTImodel, VLSI's Behavioral Modeling Language, is a language with which to create behavioral models for use with VTIsim, VLSI's mixed-mode simulator. The language consists of a set of procedures and definitions embedded in the programming language MAINSAIL.

VTImodel serves two important functions. First, it aids top-down design by allowing you to model the functionality of circuits and subcircuits before they are implemented. Second, it provides improvement in simulation performance by allowing the substitution of behavioral models for detailed circuit descriptions.

Features

Use With VTIsim

Models written with VTImodel can be simulated in VTIsim, an interactive, mixed-mode simulator with waveform display.

Parameters

Models can have arbitrary parameters. The parameters can specify timing constants, data file names, widths of buses and other pin configuration data, execution options, and so on.

Flexible Pin Configuration

A single model can represent multiple pin configurations. For example, a counter model might have parameters that control the number of bits in the counter, whether or not it has PRESET and CLEAR inputs, and whether these inputs are active high or active low (with names to match).

Detailed Timing

VTImodel supports a sophisticated timing model, including propagation delay, ramp times that vary with capacitive loading, and control over spike filtering (inertial delay). Simpler min/max and unit delay paradigms are also supported.

Setup and Hold Checking

You can check setup and hold times on input pins. The checking can be automatic (whenever the model accesses a pin's value) or by explicit calls to the check procedures.

Nine-State Logic

You can set model outputs with strength as well as value information. The VTIsim algorithm for computing a node's value takes into account both transistors and model outputs on the node.

3-Valued Functions

VTImodel has predefined procedures for operations on 3-valued data, that is, data whose value can be *high*, *low* or *unknown*.

High-Level Data Transfer

Model inputs and outputs can be strings or numerical values as well as buses or single bits. This is useful for top-down design.

Embedded Programming Language

You can use the full power of the MAINSAIL programming language to write models.

Interface With Schematic Editor

You can automatically generate a schematic symbol that represents a model, and use that symbol in the VTIschematic window.

Switching Between Model and Implementation

At simulation time, you can substitute a model for the corresponding logical or physical implementation, or vice versa. You can simulate a design that includes a mixture of models and implementations.

VLSI Test Package

Test Description Language and Test Generation Tool

Overview

VTItest, VLSI's Test Generation Language, is a test development tool that consists of a test description language with interpreter, and a test program generator. It is used to generate both simulation drivers and complete hardware test programs, from a single high-level description.

You use VTItest's test description language to describe the physical characteristics, timing, stimuli patterns, and expected responses of your design in a high-level modular fashion. This description is independent of simulator or tester characteristics. VTItest then translates the language description into commands for specific simulators or testers.

VTItest bridges the gap between design and test, enabling you to develop a device and its test program simultaneously. VTItest notifies you early in the design process about any tester-specific details that would affect the test of your device, so that you can make the optimal design and test trade-offs up front while the device is still being developed.

VTItest identifies any tester limitations exceeded by the high-level test description, such as the number of accessible test pins or the placement of timing generators, and suggests ways to work around them. The vectors derived from the description may also be further screened by VTIvector for adherence to VLSI's more specific guidelines. In the process, you become familiar with both testing problems and their solutions, helping you design testable circuits.

The test program generated by VTItest is complete and can be executed on the tester without manual intervention. It contains all timing generator, strobe, and register specifications, as well as all pattern-loading, dc parametric and summary routines, in addition to the functional test vectors.

In contrast to simulation post-processor approaches to test generation, both simulation and test are driven from the same description. Therefore, the

test vectors generated by VTItest have a 1:1 correspondence with the vectors used during simulation, including timing. All critical information is preserved and not lost, resulting in a high-quality test, and highly self-sufficient design documentation for test engineers.

To further close the loop between simulation and test, the responses predicted by the simulator can be automatically incorporated into the test program as expected responses.

Features

High-Level Language

You describe the physical characteristics, timing, stimuli patterns, and expected responses of the device under development using a high-level language. VLSI's test language is powerful and flexible, yet simple, so you can efficiently develop a simulation or test program. For example, the language contains looping constructs and provides the capability to pass parameters between modules.

Modularity

The test language is based on a modular format to simplify description. The description is partitioned into small, manageable modules, of various types, each describing a well-defined portion of the test or simulation procedure. For example, one type of module contains pin definitions, one type specifies timing parameters, and one type defines DC test conditions.

Variety of Outputs

VTItest directly supports several simulators and testers, which currently include VTIsim and several Sentry testers, and supports an even greater variety in conjunction with VTIvector.

Simulator and Tester Independence

The test description is simulator and tester independent. Simulator-specific and tester-specific limitations are identified by VTItest during translation from the language description to the commands that drive a specific simulator or tester. VTItest also suggests ways to work around the simulator or tester limitations.

Testability

VTItest enables you to develop a device and its test program simultaneously. It notifies you early in the design process about any tester-specific details and practical limitations that would affect the test of your device, so that you can make the optimal design and test trade-offs up front while the device is still being developed.

Flexibility

VTItest may be used to develop "at-speed" simulations and tests, or it may be used in conjunction with VTIvector to further screen the description and produce vectors that conform to VLSI's more specific test guidelines.

Complete Test Program

The test program generated by VTItest is complete and can be executed on the tester without manual intervention. It contains all timing generator, strobe, and register specifications, as well as all pattern-loading, dc parametric and summary routines, in addition to the functional test vectors.

Predefined Test Routines

In the test description, you can access predefined test routines that measure DC parametrics such as output voltages, input leakage currents, and power pin leakage currents. You can explicitly specify the measurement and forcing function values, or you can select a set of user-defined default values.

1:1 Correspondence

The test vectors generated by VTItest have a 1:1 correspondence with the vectors used during simulation, and all timing information is preserved during the test program generation. This is because both simulation and test are driven from the same description, as opposed to simulation post-processing approaches, which often lose critical information. Furthermore, VTItest can incorporate into the test program the responses predicted by simulation.

Device Documentation

The test description is highly readable and serves as an up-to-date source of documentation, since it is also the ultimate source used to drive the simulator and generate the test program. The test description contains sufficient information for data sheet creation or communication to test engineers. You can also augment this information with comments in the test description that are to be placed in the generated simulation driver and test program files, where the comments can be extremely valuable during simulation and test debugging.

Interactive or Batch Mode

You may use VTItest interactively or with command files, in either the graphics or terminal environments, from VLSI's text shell or terminal emulator window. VTItest runs on top of VTIsim; as such it can produce VTIsim command files or drive VTIsim interactively.

VLSI Vector Processor

Vector Screener and Translator

Overview

VTIvector, VLSI's Vector Screener and Translator, is a general-purpose vector conversion, comparison, and test generation program. It converts vectors, containing both logical and timing information, between VLSI's Vector Interchange Format (VIF) and a wide range of other simulator and tester formats. This allows the transfer of vectors back and forth between the VLSI Tools and other simulators and design environments, such as hardware accelerators or third-party CAE workstations. Using VTIvector, you can automatically convert a trace file from one simulator into an input driver for resimulation on another simulator, or you can create test vectors and full test programs for a variety of testers.

VTIvector contains a vector rule checker, to ensure that the vectors pass VLSI's recommended test guidelines as well as any restrictions imposed by destination simulators, testers or formats. VTIvector also contains a comparison utility that compares trace files from multiple simulations or simulators.

The main conversion paths supported by VTIvector are:

- Conversion from simulator output trace files to VLSI format.
- Conversion from VLSI format to simulation drivers.
- Conversion from VLSI format to test programs.

Being flexible yet simple, VTIvector provides you with a number of options for each of these paths. For example, you can modify your vectors as they are converted, by scaling or altering their timing.

VTIvector also contains several other features to help convert vectors. For example, through an interface with VTIexchange and VTIscreen, VTIvector

derives certain logical information from your design netlist that it needs to interpret the vectors, and which may often be lacking from the trace files.

VTIvector also has a vector composition utility for building complete test programs out of test sequences for lower-level blocks.

Features

Variety of Input Trace Formats

VTIvector can read a growing list of simulation vector trace file formats, currently including VTIsim, Mach1000, Hilo, Mentor, Daisy, Tegas, and Fairlog. Many of these trace types have several formatting variations which VTIvector handles, For example, VTIvector handles both dynamic and tabular VTIsim formats.

Variety of Output Simulation Driver Formats

VTIvector can generate simulation drivers for a growing list of simulators, currently including VTIsim (trace file or waveform format), Mach1000, Hilo, and Daisy.

Complete Simulation Drivers

The driver written by VTIvector is complete with a "header" section in addition to the input vectors. The header section declares watched signals and can also be output independently, to be used with vectors produced by other sources such as VTItest. For simulators such as VTIsim that support dynamic comparison of outputs with expected results, VTIvector also incorporates the output vectors and corresponding test into the driver.

Variety of Test Program Formats

VTIvector can write complete test programs for a growing list of hardware testers, currently including several Sentry and Sentry-compatible testers, as well as the MegaOne tester.

Complete Test Program

The test program written by VTIvector includes full parametric testing, as well as a functional test with specified channel assignments and all required timing definitions. The test program can also contain specified at-speed critical path checks. Circuit initialization for the parametric tests is automatically derived from the functional vectors.

Print On Change

VTIvector can read and write vectors with either print-on-change or print-on-strobe formats. Print-on-change formats contain one vector for every signal change within a cycle, thereby preserving detailed timing information. Print-on-strobe formats contain at most one vector for every cycle.

Timing Modification

VTIvector supports several options to perform timing transformations on your vectors as they are converted. You can scale all input timing by a specified factor, or you can insert a standard delay between each input transition to create a purely synchronous test. You can also completely override the original timing information for the conversion.

Vector Screening

VTIvector screens trace vectors for adherence to VLSI's recommended test guidelines. These guidelines ensure that your vectors conform to tester constraints and that they can be easily converted to reliable test programs. VTIvector warns about irregularities in the vectors and dangerous conditions which could result in a bad test or simulation. such as illegal timing, unknowns on inputs, glitches on outputs, or illegal conditions on bidirectional nodes.

Extensive Error Checking

Besides screening for adherence to test guidelines, VTIvector also performs several other checks upon conversion. VTIvector checks for conversion-specific conditions which would hamper conversion, such as syntax errors, unrecognized names, or vectors in one format which fail to conform to the vector format to which they are being converted.

Vector Comparison

Using VTIvector, you can compare two sets of vectors, causing all logical and timing discrepancies between the two versions to be flagged. You can compare all signals, or only certain ones, and you can specify a timing tolerance factor for the comparison. VTIvector also produces a merged set of vectors, in which all discrepancies in the values between the two sets are replaced with don't-cares. This feature is useful, for example, to compare worst case versus best case simulations, and consequently mask out output nodes that are unstable over process variations.

Vector Composition

VTIvector has a utility to automatically combine test programs for individual blocks with test vectors for the top-level interconnect, to create a single integrated test program, using test multiplexing methodology. VLSI Technology provides canned test programs for its large function blocks such as Megacells.

Interface With Design Netlist

Besides reading a trace file, VTIvector derives certain logical information from the design netlist, in order to identify I/O signals and their characteristics, and to resolve node names.

Interactive or Batch Mode

You may use VTIvector interactively or with command files, in either the graphics or terminal environments, from VLSI's text shell or terminal emulator window.

VLSI Spice

MOS Device-Level Circuit Simulator

Overview

VTIspice, VLSI's SPICE Circuit Simulator, is an enhanced version of Berkeley's SPICE 2G.6 that has been optimized for the detailed simulation and analysis of MOSFET circuits. It is a general-purpose circuit simulation program for nonlinear dc, nonlinear transient, and linear ac analysis. Besides MOSFETs, circuits may contain capacitors, resistors, diodes, voltage sources, and other devices.

VTIspice is used to accurately analyze the detailed electrical characteristics of a circuit, such as speed and power consumption. It is most commonly used when you wish to look very closely at one small section or critical path of your design. It is particularly useful for verifying a custom-designed circuit, such as that designed using VTIlayout or VTIsticks.

Your VTIspice input file may be generated using a combination of many methods. Netlists produced by other VLSI tools can be automatically converted into VTIspice input format, taking the desired process corner and operating conditions into account. You can also specify an independent file of further control commands to be incorporated into the input file. This is particularly useful when resimulating multiple versions of the same circuit. Finally, you can edit the resulting input file to further customize it.

If you wish to simulate only a portion of your circuit, such as one or more critical paths, you can direct the conversion process to isolate that portion. Irrelevant portions of the circuit are trimmed away and do not appear in the input file.

Features

Accurate MOSFET Models

The model parameters used in VTIspice are generated by actual data from test chips fabricated at VLSI. Some model parameters have been added to more accurately represent the silicon.

Berkeley SPICE Compatibility

VTIspice is an enhanced version of Berkeley's SPICE Version 2G. It supports all the circuit types and features as Berkeley SPICE, plus more.

Convergence

Many convergence problems present in the raw Berkeley code have been corrected or improved.

Diagnostics

VTIspice produces a diagnostic file that gives information and suggestions about possible errors in your input file.

Netlist Conversion

Netlists produced by other VLSI tools are automatically converted to VTIspice format, facilitating input file preparation. The desired process corner and operating conditions for the simulation may be specified prior to the conversion. You can also specify an independent file of further commands to be incorporated into the input file.

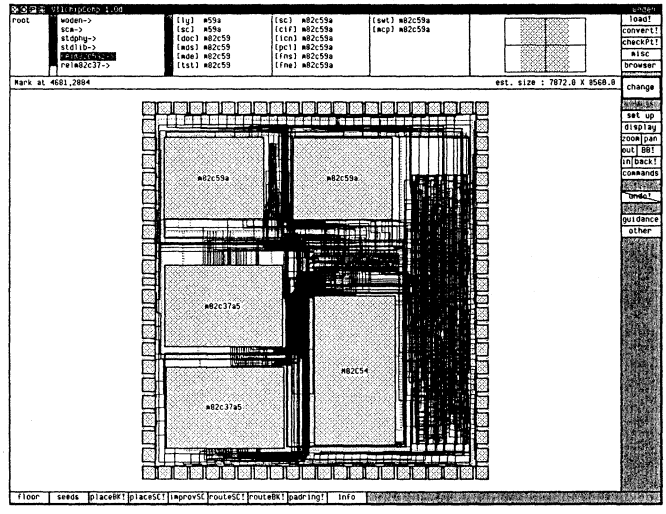
Netlist Trimming

Whole netlists may be simulated, or just portions thereof, with minimum overhead.

Physical Design Tools

VLSI Chip Compiler

Automatic Placement and Routing Tool for Chip Assembly



Overview

VTIchipComp, VLSI's Chip Compiler, is a highly automated system for chip assembly. It is an integrated arbitrary block and standard cell placement and routing system with an interactive floorplanning capability. Input to the system is by a schematic netlist that may contain any mixture of standard cells and multiple arbitrary blocks (such as compiled blocks or Megacells).

In the floorplanning stage, the user defines general rectangular areas for standard cell placement and the relative locations and orientations of these areas and the arbitrary blocks. Standard cell areas with vertical and horizontal rows can be mixed. Analysis tools are provided to evaluate the efficiency of the floorplan based on interconnect routing density.

After the floorplan is established, VTIchipComp automatically places and routes the standard cells, and routes the interconnections between the defined standard cell areas and the arbitrary blocks, using a compaction algorithm to minimize the area of the chip. VTIchipComp automatically partitions the standard cells among standard cell areas for physical optimization, independently of the logical partitioning in the schematic, as if to "pour" the cells in to fill the gaps between the arbitrary blocks. If an entire chip is to be compiled, VTIchipComp also automatically assembles a padring around the chip.

Most automatic features can also be manually controlled to whatever degree is desired. The user may control VTIchipComp's placement of critical cells by placing net weights in the schematic, or by specifying seed placements during the compilation. Any number of standard cells, I/O connectors and pads may be seed-placed. The user may also give "suggestions" within the floorplan on how he would like critical nets to be routed. Specifying this guidance simply involves drawing a rough path with symbolic wires, and optionally, indicating a desired width for each wire.

If the schematic netlist contains pads, VTIchipComp's padring generator automatically selects from multiple pad sets, based on the optimal pitch for a pad-limited versus core-limited design. Corner pad cells are used to reduce area even further. Power busses in the padring are automatically sized, based on core power consumption, or the user may preset their widths. The interconnection between the padring and the core is also automatically routed.

The compiled chip is directly suitable for fabrication. VTIchipComp also generates the compiled cell in either VTIchipComp, VTIcompose or VTIlayout format, for higher-level assembly or further optimization. In addition, VTIchipComp outputs a back annotation command file with wiring capacitances for accurate post-route performance verification in VTIsim or VTItv.

Features

Cell or Complete Chip

VTIchipComp is capable of compiling either a cell that can be used as a building block with other cells, or it can compile a whole chip with a padring.

Flexible Floorplanning

VTIchipComp allows flexible interactive floorplanning of the arbitrary blocks and standard cell areas prior to compilation. You can move or alter the orientation of each block and standard cell area. You can also provide arbitrary constraints to control the size, shape, and other floorplan attributes of each standard cell area, if desired. VTIchipComp includes analysis functions that provide you with feedback on the efficency of your floorplan, allowing you to improve the floorplan in a directed manner.

Seed Placement

You can specify the locations for any standard cell instances, I/O connectors or pads by seed-placing them. Additionally, there are two types of seed placements for standard cells, soft and hard. In the case of a soft standard cell seed placement, the cell is placed at its seed location during initial placement, and its location is free to change during placement improvement. In a hard seed placement, the cell is fixed forever at its seed location -- it will not move during the placement improvement phase.

Net Weights

You can specify net weights for critical nets in your schematic, the net weight property sheet, or in a text file. VTIchipComp places cells on weighted nets closer together.

Automatic Standard Cell Placement

After evaluating the input netlist, floorplan, net weights, and seed placements, VTIchipComp partitions the standard cells into standard cell areas independent of the logical partitioning in the schematic. It generates an initial placement of the standard cells which minimizes the total connecting wire length and routing congestion.

Standard Cell Placement Improvement

VTIchipComp improves the initial placement of standard cells in the standard cell areas. by iteratively rearranging the locations and orientations of standard cells to reduce wire length yet further. You can control the amount of effort spent on this phase.

Powerful Standard Cell Routing

Following standard cell placement, all standard cell areas are routed. Afterwards, each area is treated as an arbitrary block. The standard cell router has several features in common with the global router, such as multiple layers, variable-width wires, channel compaction and via elimination. In addition, the standard cell router has several features specific for optimization of standard cell areas, including:

- **Routing Over Cells.** The router identifies open routing tracks within the standard cells for the second routing layer, and routes over the cells using these locations.

- **Feed-Through Insertion.** The router automatically inserts feed-through cells into the standard cell rows, to provide yet more open locations for routing, or to otherwise space out the rows, if necessary.

- **Row-End Cell Insertion.** VTIchipComp automatically puts in guard ring cells at the ends of the standard cell rows for safe CMOS design.

- **Power Connection Cell Insertion.** In standard cell areas, the router automatically inserts power and ground connection cells for tying signals to VDD or VSS.

Global Routing Guidance

Global routing guidance lets the user control where parts of certain nets are to go. Guidance may be useful for critical wires or ones that go to many places, such as power buses. Guidance is specified by drawing segments between the blocks in the floorplan. The guidance may be complete or partial, and may optionally specify the width of each segment. Complete guidance for a net means that guidance for all its segments is specified. Partial guidance means that the guidance only covers a portion of the net. In this case, the global router completes the net automatically.

Powerful Global Routing

During global routing, VTIchipComp routes the arbitrary blocks according to the connections in the input netlist. The global router has many features to optimize routing area, including:

- **Multiple Layers.** The global router uses multiple layers, normally metal1 and metal2.

- **Variable-Width Routing.** You can specify unique widths for power buses, critical signals, or other non-standard "wide" wires, and you can specify varying widths for different segments within a wire.

- **Channel Compaction** Instead of using a fixed pitch for each routing track, VTIchipComp compacts all the routed wires in each channel, inserting jogs where necessary. This results in the smallest possible channel routing area -- usually a 5% to 20% saving.

- **Via Elimination** When compared with other existing channel routers, VTIchipComp's router generates routes with much fewer vias and shorter wire length even without via elimination. Nevertheless, the router goes through an extra step to further eliminate vias and unnecessary layer changes, to improve the timing characteristics and process yields of the chip.

- **Gridless routing.** The global router can route arbitrary blocks whose connectors need not be laid out on a routing grid.

Padring Generation

VTIchipComp automatically chooses between pad types suited for pad-limited or core-limited designs, and generates the padring to satisfy bonding requirements and user specifications. VTIchipComp then routes signal wires and power buses from the pads to the core of the chip.

Corner Pads

In the case of a pad-limited design, VTIchipComp is capable of generating a padring with corner pads, further reducing chip size.

Power Bus Sizing

VTIchipComp automatically sizes power buses, according to power consumption and specified operating frequency, or uses a preset width that you specify.

Back Annotation

VTIchipComp can generate a back annotation command file that contains the capacitances of the routed wires, for accurate post-route performance verification.

Flexible Sequence of Operation

After executing any of the seven major steps in VTIchipComp (loading a netlist, block placement, standard cell area initial placement, standard cell area placement improvement, standard cell area routing, arbitrary block routing, and padring generation), you can go back to the result of any previous step and discard the results of the steps in between. Also, VTIchipComp allows you to checkpoint your cell at any time, or save an intermediate result under a different name.

Variety of Outputs

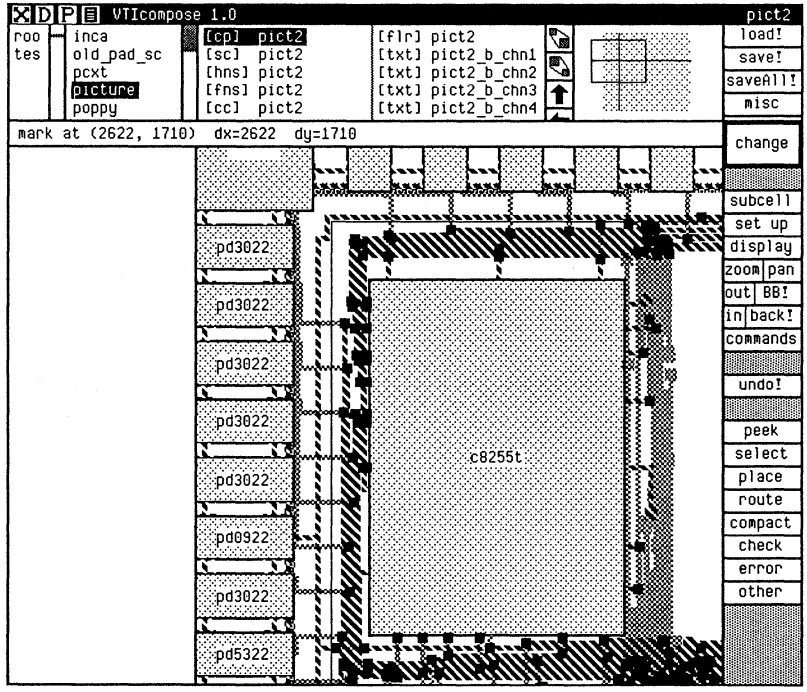
VTIchipComp can output chip designs in a variety of formats suitable for input to other VLSI tools, including VTIcompose and VTIlayout. The top-level cell, the core, and the standard cell areas can independently be output in each of these formats.

Interactive or Batch Mode

You may use VTIchipComp interactively or with command files, in either the graphics or terminal environments.

VLSI Composition Editor

Symbolic Chip Assembly Tool



Overview

VTIcompose, VLSI's Composition Editor, is a graphical chip assembly tool. It is a semi-automatic multiple block placement and routing system that retains connectivity information. It is used during the physical design stage, and supports a wide range of design styles and methodologies. These include top-down or bottom-up design, semi-manual placement with any combination of manual and fully automatic routing, and interconnection by abutment with automatic stretching. VTIcompose is particularly useful when used in combination with VTIchipComp, to further optimize VTIchipComp's routing.

VTIcompose automatically routes the interconnections specified in a schematic netlist or another composition cell. The router routes around arbitrary-sized rectangular blocks having connectors on their boundaries and also deals with previously drawn interconnections, including those entered manually. Instances and wires made by the automatic operations can be modified manually.

VTIcompose displays *unrealized interconnections* between cells, logical connections for which the actual physical routing has not been done yet, or which have been broken due to a change in lower level cells. This helps you evaluate the efficiency of the floorplan and improve it based on interconnect routing density and congestion. It also allows actual routing to be deferred for some or all interconnections.

All wires are represented as stick-type, symbolic interconnections without physical width. VTIcompose includes a symbolic layout compactor which spaces all wire segments and instances of cells in minimum area to meet the design rules. VTIcompose also automatically inserts and deletes contacts between wiring layers when you manually draw wires. In addition, the compactor can stretch sticks or composition cells to satisfy abutment.

Composition cells may be plotted, written as CIF (CalTech Intermediate Format), placed in VTIlayout VTIcompose also directly generates simulation netlists, including interconnect capacitances, without invoking VTIextract.

Features

Technology Independent

VTIcompose reads a technology file which defines valid wiring layers, minimum wire widths, contact structures and so forth. Therefore, you can use VTIcompose with any one of a variety of technologies, and you need not know the design rules.

Hierarchy

VTIcompose is hierarchical. You can place and interconnect cells which may themselves contain instances of other cells, to any depth of hierarchy.

Phantom Cells

VTIcompose facilitate top-down design, through the use of phantom cells, which represent parts of the chip that have not yet been designed. Phantom cells can be used to start planning a chip and building a general floorplan, before all the pieces of the chip have been finished. The internals of the phantom cells can then be specified as the design becomes more refined, Phantom cells may be created, deleted and edited with VTIcompose.

Variety of Output

Besides creating a composition cell, VTIcompose can write CIF, which includes the entire hierarchy. VTIcompose can also directly generate a simulation netlist, including interconnect capacitances, without invoking VTIextract. Besides simulation or net comparison, the netlist can be used as router input for another version of this composition cell.

Many Different Kinds of Cells

You can hierarchically place and interconnect any cells which contain some form of geometric layout (CIF, layout and sticks, as well as composition). These include cells designed in other VLSI Tools editors, cells from VLSI's libraries, or cells defined outside the VLSI Tools system and read in as CIF. You make instances of these cells and wire them together without regard to the kind of data in the cells.

Manual Routing With Automatic Features

Being symbolic, VTIcompose makes even manual routing semi-automatic. VTIcompose automatically sizes the segments to match existing connectors or wires. It also inserts contacts when you change wiring layers, or removes contacts as required when you delete segments. The contacts are automatically sized according to contacted wire widths, and are made up of multiple contact cuts when appropriate. When you move a segment, other segments attached to it automatically move or change length to maintain the connection, thus minimizing the chance of accidentally breaking a connection.

Checking

VTIcompose optionally checks that you do not short wires with different node names, reducing the chance of accidentally making incorrect connections.

Metal Maximization

VTIcompose can maximize metal along a wire, even if manually drawn, thereby minimizing undesirable wiring layers. VTIcompose changes as much of the wire to metal as possible, inserting jumpers under existing metal, while maintaining design-rule correctness.

Automatic Placement

VTIcompose can do a quick initial placement from a schematic. It places the blocks of the design at the same relative locations as that of the icons in the schematic.

Unrealized Interconnect

VTIcompose allows you to specify and/or view unrealized interconnections, logical connections for which physical paths might not have yet been drawn or routed. Unrealized interconnect is helpful during the initial stages of design, such as floorplanning, and can be used later to invalidate routes without totally deleting the connection. You can specify unrealized interconnect manually or load connections from a schematic netlist or another composition cell.

Automatic Routing

VTIcompose includes a general-purpose arbitrary-block symbolic router, which automatically routes interconnections specified in a schematic netlist or another composition cell. It routes all the interconnections or just selected ones, taking into account previously realized partial or whole interconnections, including those entered manually. VTIcompose can display the interconnect either symbolically or as actual layout, before or after compaction.

Compaction

VTIcompose provides a symbolic layout compactor, which automatically spaces all wires and instances to minimum design rule spacing, as defined in a technology file. The compactor also minimizes the length of segments on poor wiring layers.

Flypaper Instances

Wire segments over the top of cell instances can be treated as fixed relative to those instances during compaction. This is particularly useful for wiring over the top of an instance to make a connection to an interior node.

User Compaction Constraints

You may direct the compactor by defining individual constraints on relative positioning of features, such as exact, minimum or maximum separations between points. You can graphically view the critical path through the cell, to guide you in specifying useful constraints. Overconstrained parts of the cell cause error flags which you can also view graphically.

Layout Constraint Violation Check

VTIcompose can quickly check for violated constraints between pieces of layout in your cell without doing a full compaction. You can use this feature as a partial design rule check, for spacing rules. Places where the compactor would insert more space are flagged, indicating a possible design rule violation.

Error Handling

You can quickly review compaction or other VTIcompose command errors and step through them. VTIcompose automatically pans the display to each error location, indicates the location with a black mark or error bars, and reports the error message. Similarly, VTIcompose overlays VTIdrc and VTInetCompare error flags on top of your design, enabling you to conveniently locate, view and step through such errors in context.

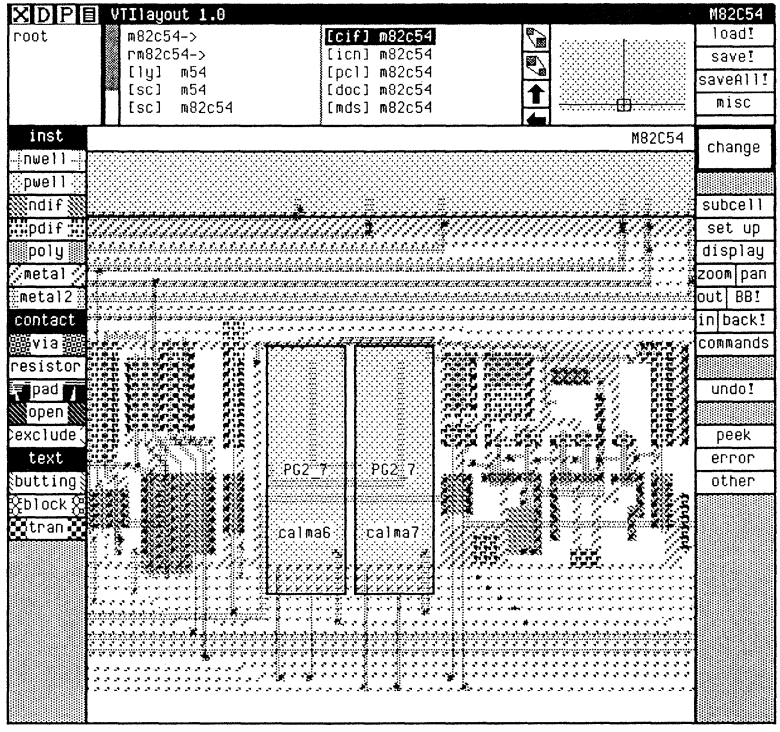
Cut and Paste

With cut and paste, you can delete, copy and move whole areas of a composition cell intact, either within that cell or between cells. Cutting and pasting between different cells provides a convenient way to build a hierarchy from a flat cell or flatten a hierarchy into a cell.

Group Select	VTIcompose lets you select whole groups of items to be operated on at once, by graphically pointing, or by naming with wild cards.
Arrays	VTIcompose lets you conveniently and flexibly place arrays of instances.
Undo	Most commands in VTIcompose can be undone.
Interactive or Batch Mode	You may use VTIcompose interactively or with command files, in either the graphics or terminal environments.

VLSI Layout Editor

Geometric Layout Tool for Cell Design or Chip Assembly



Overview

VTIlayout, VLSI's Layout Editor, is a graphical tool for hierarchical cell design or chip assembly at the mask level. It makes drawing cells and interfacing them together very easy. Using the edit-in-place feature, you can edit cells anywhere in the hierarchy while viewing the context in which they are used. This greatly simplifies the task of interfacing cells.

While VTIlayout can be used for complete chip layout and assembly, it can also be used effectively in conjunction with the other VLSI physical design tools. Cells designed in VTIlayout can be automatically assembled in VTIchipcomp or VTIcompose. VTIlayout can also be used by the experienced designer to modify cells produced by other means, such as those symbolically-generated in VTIsticks or VTIcompose. VTIlayout can also write cells in CIF (CalTech Intermediate Format), and layout cells can be plotted.

VTIlayout is also useful in conjunction with VLSI's verification tools, such as VTIdrc, VTInetcompare and VTIextract. VTIlayout provides a convenient interface for quickly examining error output from VTIdrc and VTInetCompare, or running VTIdrc interactively on the spot. Through VTIextract, cells designed in VTIlayout can be written as netlists.

Features

Hierarchy

VTIlayout is hierarchical. You can place and interconnect cells which may themselves contain instances of other cells, to any depth of hierarchy.

Many Different Kinds of Cells

You can hierarchically place and interconnect any cells which contain some form of geometric layout (CIF, composition, and sticks, as well as regular layout). These include cells designed in other VLSI Tools editors, cells from VLSI's libraries, or cells defined outside the VLSI Tools system and read in as CIF.

Edit In Place

In VTIlayout, you can edit a layout subcell other than the top-level cell, anywhere in the hierarchy, in the context in which it is used. This greatly simplifies the task of drawing cells that must interface together, because you can clearly see the interface and can conveniently change from editing one cell to the other.

Arrays

VTIlayout lets you conveniently and flexibly create, modify and flatten arrays of instances.

Convenient Painting

VTIlayout lets you fill a box by simply pointing to any layout on the screen that is on the layers you want, as well as by pointing to a layer menu.

Arbitrary Polygons

VTIlayout supports polygons with edges at arbitrary angles. Alternatively, it can snap the edges to multiples of 45 or 90 degrees.

Wire Drawing

VTIlayout has a powerful yet simple user interface for drawing wires. It can draw wires left, right, or center-justified, at arbitrary angles. It can also snap wire segments to multiples of 45 or 90 degrees.

Cut and Paste

With cut and paste, you can delete, copy and move whole areas of layout either within a cell or between cells. Cutting and pasting between different cells provides a convenient way to build a hierarchy from a flat cell or flatten a hierarchy into a cell.

Undo

Most commands in VTIlayout can be undone.

Interactive or Batch Mode

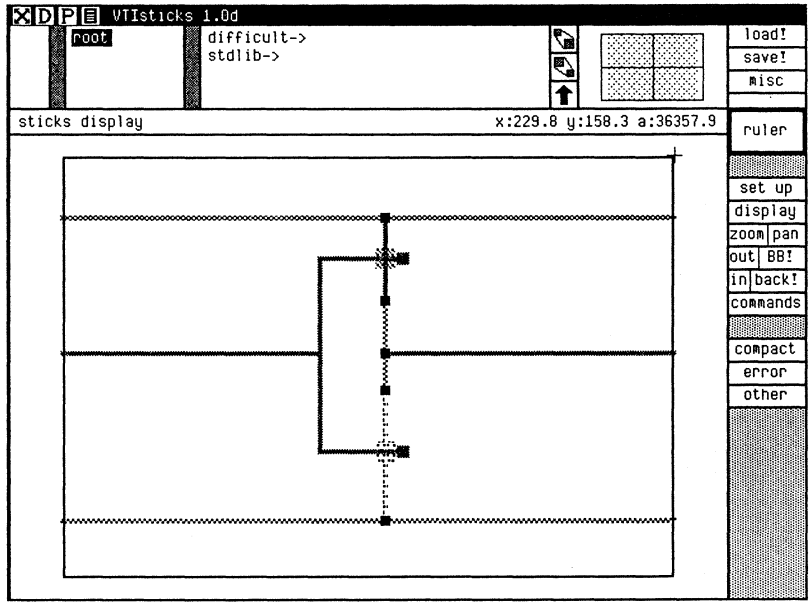
You may use VTIlayout interactively or with command files, in either the graphics or terminal environments.

Error Handling

VTIlayout has a convenient user interface for viewing VTIdrc and VTInetCompare errors. VTIlayout overlays the error flags on top of your design, and automatically pans the display to each error location, so that you can quickly locate and step through the errors in context. VTIdrc can also be run directly in VTIlayout on any selected area of the design.

VLSI Sticks Editor

Symbolic Layout Tool for Leaf Cell Design



Overview

VTIsticks, VLSI's graphical Sticks Editor, provides you with the ability to conveniently describe layout by creating and editing stick diagrams. A *stick diagram* describes the layout topology of circuits symbolically. Using thin lines to represent physical wires of unspecified width in various layers, the diagram describes circuit connectivity and relative area locations. Objects such as transistors, contacts and connectors are automatically derived from the configuration of the lines and immediately displayed as symbols. VTIsticks automatically fills out the layout, while compacting it to minimum area, to meet the design rules. The compaction can be manually controlled to the degree desired, by specifying constraints.

VTIsticks is a convenient tool for physical leaf cell design. These cells can then be used in VLSI's higher level chip assembly tools, such as VTIcompose. VTIsticks stick diagrams may be plotted, written as CIF (CalTech Intermediate Format), placed in VTIlayout or VTIcompose, or edited in VTIlayout for further detail modifications. VTIsticks also directly generates simulation netlists, including node capacitances, without invoking VTIextract.

Stick diagrams are a quick, easy and reliable way to sketch layout, because they do not contain details. They are also thereby uncluttered and easy to understand. VTIsticks can easily be used by the system designer, because he need not know the design rules. The details are automatically derived and filled in by VTIsticks, guaranteeing correctness.

Features

Symbolic Layout

With VTIsticks, you can describe a circuit symbolically, using stick diagrams. This is a higher level way to describe a circuit layout than at the polygon level, because the designer need not fill in all the details of the actual layout.

Design Rule Independent

Stick diagrams are independent of the design rules, which are handled automatically by VTIsticks via a technology file. The designer need not know the design rules.

Device Recognition

Objects such as transistors, contacts and connectors need not be drawn. They are implied and are automatically recognized and symbolically displayed by VTIsticks.

Compactor

VTIsticks includes a compactor which converts stick diagrams into actual layout. The compactor automatically spaces the layout to satisfy the design rules in minimum area. VTIsticks can display the resulting circuit in a number of ways, including symbolically, as actual layout before or after gap patching, or showing the critical path.

User Compaction Constraints

You may direct the compactor by defining individual constraints on relative positioning of features, such as exact, minimum or maximum separations between points.

Variety of Output

Besides creating a sticks cell, VTIsticks can write CIF. VTIsticks can also directly generate a simulation netlist, including node capacitances, without invoking VTIextract.

VLSI CalCif and CifCal

Data Conversion Programs Between CIF and Calma GDSII

Overview

CalCIF

VLSI's `CalCIF` utility converts a file in Calma `GDSII` Stream format to a file in CIF format (CalTech Intermediate Format). The input can be in level 3, 4 or 5 of Calma's specification. This utility currently operates on VAX/VMS, Apollo/Aegis and Ridge/ROS. It takes the input Calma file from tape and writes the CIF file out to disk. It can also read from a disk file containing a tape image.

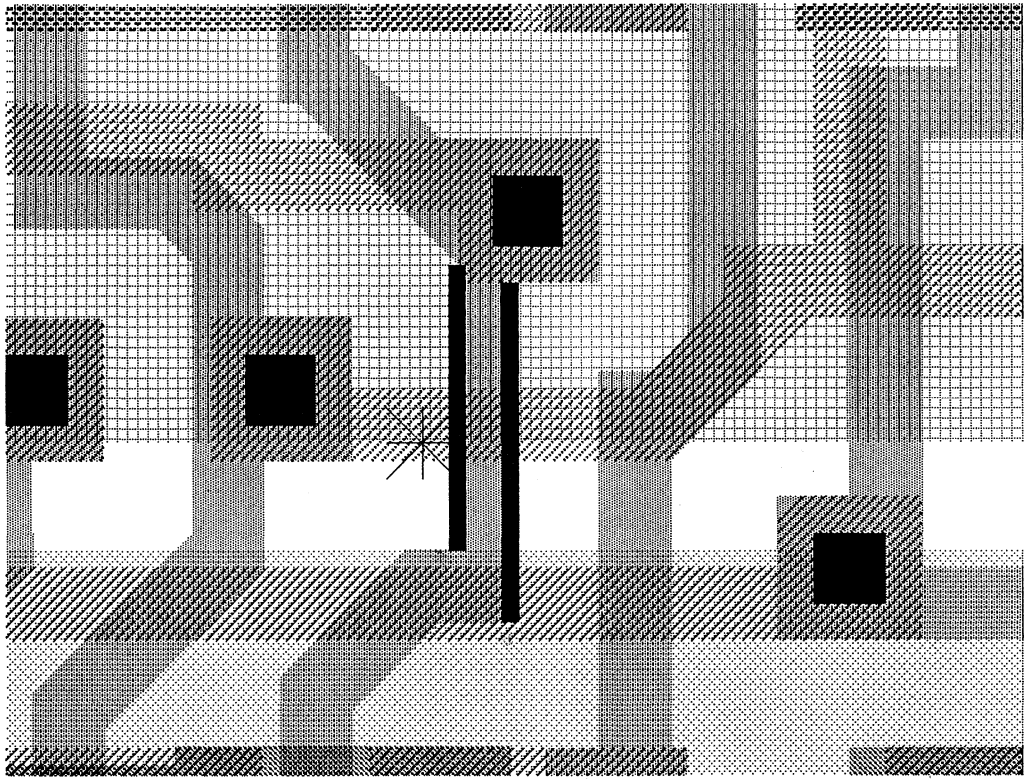
CIFCal

VLSI's `CIFCal` utility converts a file in CIF format to a file in Calma `GDSII` Stream format, level 3. This level is also compatible with levels 4 and 5. The utility currently operates on VAX/VMS, Apollo/Aegis and Ridge/ROS. It takes the input CIF file from disk and writes the Calma file out to magtape.

Physical Verification Tools

VLSI DRC

Design Rule Checker



Overview

VTIdrc, VLSI's geometric Design Rule Checker, checks whether designs satisfy certain constraints on design aspects, such as wire widths and wire separation. The constraints checked are those necessary for the fabrication process in the selected technology.

VTIdrc accepts as input any cell which contains some form of geometric layout (CIF, layout, composition or sticks cells). It checks for design rule violations, and indicates errors with graphical representations in an error cell, which can contain the errors alone or merged with the CIF. The error cell can be plotted, or viewed in VTIlayout or VTIcompose.

Features

Variety of Input

VTIdrc accepts as input any physical type of cell, including CIF, layout, composition and sticks cells. Besides a complete cell, VTIdrc can also be performed on a selected area, directly in VTIlayout.

Technology Independence

VTIdrc can check designs that are done in a variety of technologies. It checks the design according to rules contained in a separate technology file for the selected technology.

Non-Orthogonal Geometry

VTIdrc accepts non-orthogonal geometry with arbitrary angles.

Phantom DRC

The design may contain phantom cells, for quicker, hierarchical checking. Phantom cells lack internal geometry, which is known to be correct. VTIdrc checks only the interfaces to such cells.

Exclusion

Areas in the design can be excluded from checking using a special exclude layer, to speed checking.

Recovery

VTIdrc has the capability to recover from system interruptions and resume where it left off.

Interactive or Batch Mode

You may use VTIdrc interactively or with command files, in either the graphics or terminal environments. VTIdrc is accessible from VLSI's text shell or terminal emulator window, or directly from VTIlayout.

Graphical Feedback

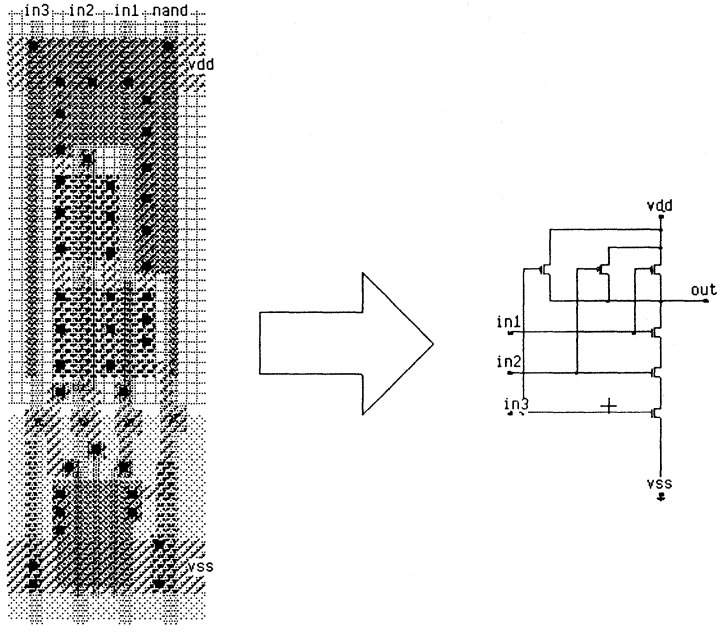
If there are any design rule violations, VTIdrc creates an error cell with graphical flags to represent the errors. The error cell can be loaded as an overlay in VTIlayout or VTIcompose for convenient viewing, location and stepping through of the errors in context. Similarly, VTIdrc creates a version of the error cell with the original design already merged in, for plotting.

Runtime Feedback

As VTIdrc executes, it reports the general class of checks it is working on and a total count of the design rule errors it finds at each step.

VLSI Extractor

A Tool for Circuit Extraction from Physical Data



Overview

VTIextract, VLSI's Circuit Extractor, converts completed layout to a transistor netlist, which is used to close the verification loop between the layout of a design and its specification.

VTIextract accepts as input any cell which contains some form of geometric layout (CIF, layout, composition or sticks cells), and produces a flat netlist describing the circuit. The netlist contains all the nodes and transistors in the circuit, annotated with node names. It also contains the width and length of each transistor, the capacitance on each node, and stray capacitances between nodes.

The netlist can be used as input to VTIsim (VLSI's Mixed-Mode Simulator) or VTIspice for simulating the circuit, to VTInetcompare (VLSI's Netlist Comparison Program) for comparing the extracted circuit to the schematic, and to a variety of other useful tools and utilities.

Features

Transistors

VTIextract recognizes arbitrary transistors, including annular and interdigitated ones and multiple transistors tied together, besides rectangular ones. VTIextract calculates the equivalent length and width of each transistor.

Capacitances

VTIextract calculates the capacitance on each node, recognizing capacitances of various types. The primary types are diffusion junction diode capacitance and regular conductor capacitance (including interconnect as well as gate input capacitance). For both of these types, both bottom area, or bulk, capacitances and sidewall, or fringing, capacitances are considered.

Stray Capacitances

Optionally, more detailed stray capacitances may be calculated, including coupling capacitances between crossing layers.

Analog Capability

VTIextract can extract marked capacitor, resistor and diode components.

Checking

VTIextract performs several important checks, including checks for:

- **Shorts.** VTIextract checks for shorts between certain nodes.
- **Partitioned Nodes.** VTIextract warns about partitioned nodes, two distinct electrical nodes that have the same name and thus may be intended by the user to be the same node. This check is particularly useful for ensuring that power routing is complete.

Recovery

VTIextract has the capability to recover from system interruptions and resume where it left off.

Variety of Input

VTIextract accepts as input any physical type of cell, including CIF, layout, composition and sticks cells.

Phantom Extract

The design may contain phantom cells, for quicker, hierarchical extraction and verification. Phantom cells lack internal geometry, which is known to be correct.

Discontinuity

Areas in the design can be marked to be ignored by VTIextract, using a special discontinuity layer. The discontinuity layer acts like a scalpel, excising all geometry that it covers. It can be used to mask off areas which you do not want extracted, such as logos, or to separate two nodes that would otherwise be shorted through a high-resistance path such as a well-resistor.

Interactive or Batch Mode

You may use VTIextract interactively or with command files, in either the graphics or terminal environments, from VLSI's text shell or terminal emulator window.

VLSI Netcompare

Netlist Comparison Tool

Overview

VTInetCompare, VLSI's Netlist Comparison Window, is a verification tool that compares circuits described by two netlists, in order to verify that the two circuits are the same or logically equivalent. Typically:

- One of these netlists comes from the VLSI schematic editor and describes the circuit as intended.
- The other netlist comes from VTIextract and represents the circuit as actually laid out.

You can also compare two extracted netlists or two schematic netlists.

Though the netlists might describe the circuit at the transistor level, VTInetCompare compares the circuit at the logic level. It recognizes logically equivalent circuits, whose irrelevant details need not be checked and flagged as errors.

Optionally, VTInetCompare does check detailed transistor width and length values and node capacitances.

VTInetCompare supports all MOS technologies.

Features

Logical Equivalence	VTInetCompare recognizes simple transistor stacks as logically reversible and therefore equivalent, so that they need not be checked and flagged as errors.
WL Checks	Optionally, VTInetCompare checks the width and length of transistors in the two netlists to verify that they match within a specified tolerance.
Capacitance Checks	Optionally, VTInetCompare checks node capacitances to verify that the capacitances in one netlist do not exceed those in the other netlist by more than a specified tolerance.
Analog Capability	VTInetCompare can handle capacitor, resistor and diode components.

Identification	VTInetcompare automatically identifies matching nodes. You may also give VTInetCompare starting matches in case of ambiguities.
Phantom Net-Compare	The netlist may contain phantom cells, for quicker, hierarchical checking. Phantom cells lack internal geometry, which is known to be correct.
Wild Cards	Any command that takes a list of node names as a parameter also takes wild card and exclude patterns.
Interactive or Batch Mode	You may use VTInetCompare interactively or with command files, in either the graphics or terminal environments.
Interactive Error Examination	VTInetCompare helps you diagnose problems interactively. For example, there are commands to interactively examine errors and to explore the two networks by displaying what is connected to a selected node.

Graphical Feedback

VTInetCompare optionally produces an error cell with graphical flags to represent the errors. This cell can be loaded as an overlay in VTIlayout or VTIcompose, allowing the errors to be automatically stepped through one at a time, so that you can quickly locate the problem devices and nodes in the context of your layout.

VLSI ERC

Electrical Rule Checker

Overview

VTIerc, VLSI's electrical rule checker, quickly spots dangerous circuit configurations and roughly checks circuit performance. It is a time-saving circuit-level analysis tool for the systems designer.

Static errors such as nodes not driven and unusual transistor configurations are quickly and clearly reported. In addition, an estimate of the RC delay for each node warns of nodes which might not be able to meet system speed requirements.

Features

Quick Checking

VTIerc runs very fast, with little overhead compared to a more general analysis tool, since VTIerc is optimized specifically for electrical rule checking.

Extensive Checking

VTIerc checks a wide variety of circuit conditions, including but even more extensive than those checked by VTIsim. VTIerc's checks include:

- hanging or unconnected nodes
- nodes that cannot be driven
- nodes or transistors that do not drive anything
- nodes with too many threshold drops
- transistors oddly connected, in one of many ways
- transistors otherwise used incorrectly, depending on their type and technology
- transistors whose size exceeds some limit

Figure of Merit

VTIerc reports a *Figure of Merit*, a rough indication of the delay on each node, derived from a simple RC calculation. This helps you find nodes which may have performance problems, without simulating.

Easy to Learn

VTIerc has a command syntax exactly like that of VTIsim.

Wild Cards

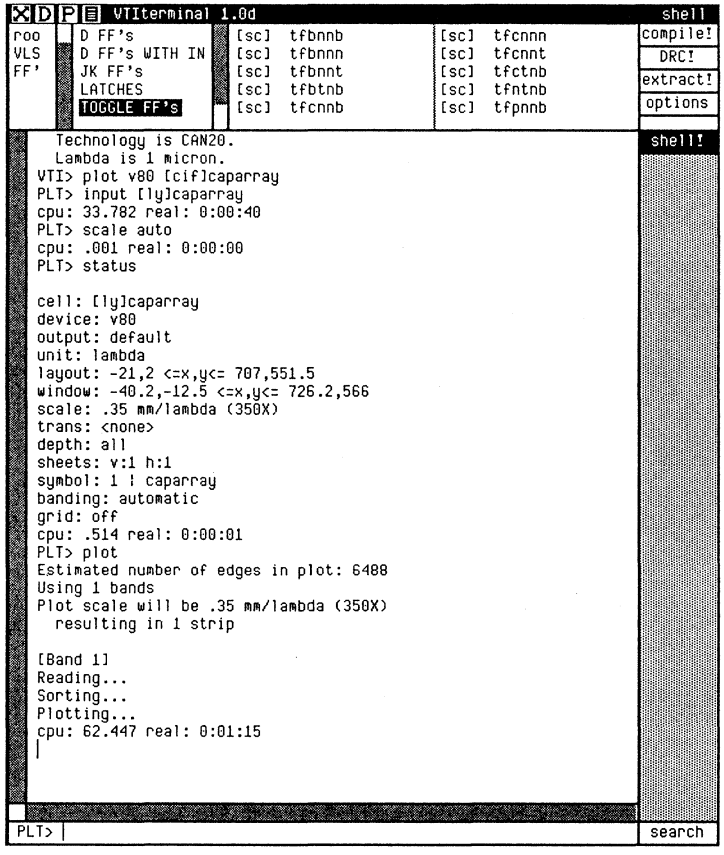
Any command that takes a list of node names as a parameter also takes wild card and exclude patterns.

Interactive or Batch Mode

You may use VTIerc interactively or with command files, in either the graphics or terminal environments, from VLSI's textual shell or terminal emulator window.

VLSI Plot Package

Plotting Interface for VLSI Tools



Overview

VTIplot, VLSI's Plotting Interface, is a plot utility for plotting layout and schematics on a variety of plotters. It can accept as input any cell which contains some form of geometric layout (CIF, layout, composition or sticks cells), as well as schematic and icon plot files created by VTIschematic and VTIicon.

The plotting program can be dynamically configured to route its output directly to the plotter or to a file you specify. You can select from a variety of options in order to tailor the plotting package to your needs.

Features

Variety of Hardware

VTIplot supports a variety of plotters, including the Versatec black-and-white and color series, as well as the Hewlett Packard series of HPGL color pen plotters.

Flexibility

VTIplot allows you to control the form of the plot with great flexibility. A specific symbol or subsection of the symbol can be plotted, as well as specific layers. Node names, connectors, or the cell abutment box can be added. Scale can be set in a variety of ways, including automatically. Multi-sheet plots are supported. Many other options are available.

No Size Limitations

VTIplot can plot files of any size, subject only to disk space limitations.

Interactive or Batch Mode

You may use VTIplot interactively or with command files, in either the graphics or terminal environments, from VLSI's text shell or terminal emulator window.

VLSI TECHNOLOGY, INC.

VLSI Technology, Inc.
ASIC Division
1109 McKay Drive
San Jose, CA 95131
408-434-3100
TLX: 278807
FAX: 408-263-2511