Volume 08 Issue 01 Published, February 18, 2004 ISSN 1535-864X

Intel® Technology Journal

Intel® Pentium® 4 Processor on 90nm Technology

Library Architecture Challenges for Cell-Based Design

Library Architecture Challenges for Cell-Based Design

Barbara Chappell, Technology and Manufacturing Group, Intel Corporation
Amanda Duncan, Technology and Manufacturing Group, Intel Corporation
Kiran Ganesh, Mobile Platforms Group, Intel Corporation
Manoj Gunwani, Mobile Platforms Group, Intel Corporation
Abhinav Sharma, Mobile Platforms Group, Intel Corporation
Madhu Swarna, Desktop Platforms Group, Intel Corporation

Index words: Cell-Based Design, standard cell library

ABSTRACT

The Intel[®] Pentium[®] 4 processor on 90nm technology is the first Intel microprocessor whose significant portion (~50% of the non-cache devices) was designed using a Cell-Based Design (CBD) methodology. In the CBD methodology, Electronic Design Automation (EDA) tools are used with a library of standard cells to build up a large and complex design. This paper describes the challenges involved in designing a standard cell library to enable the CBD methodology to be applied on a large scale on a chip with an aggressive performance target. Factors critical in enabling CBD on the Intel Pentium 4 processor included the breadth of library content, the physical architecture and design guidelines of the cells, the circuit optimization methodologies, and the functional validation of the cells. In addition to these design concerns, careful modeling for timing, noise, reliability, formal verification, and place and route were required. In this paper, we present an overview of the CBD flow, and we discuss these cell library design and modeling issues.

INTRODUCTION

Cell-Based Design (CBD) refers to a design approach that uses a library of basic building blocks called *cells*. Using cells from the library, larger, more complex functions are realized. In contrast to transistor-level *in situ* customization of cell designs [1], the cells are treated as black box entities by the design and verification tools and are fully characterized for timing, noise, reliability, etc.

[®] Intel and Pentium are registered trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

The use of CBD in the Intel Pentium 4 processor on 90nm technology enabled large "sea-of-cells" designs for improved global optimization and more rapid design convergence. Pre-qualification of the cells and a reduced amount of unique layout contributed to the quality control for the product. The CBD library also aided estimation of chip floor-plan and architectural trade-offs. Early explorations of library architecture in conjunction with 90nm technology pathfinding helped us evaluate the impact of the technology on circuits.

The quality of the library plays a key role in producing a competitive design with the CBD methodology. The goal of this paper is to present some of the technical challenges in cell library design and modeling faced by the designers.

CELL-BASED DESIGN FLOW

The Cell-Based Design (CBD) flow for the Intel Pentium 4 processor consisted of four basic steps:

1. Netlist generation
2. Cell placement
3. Routing
4. Design verification

Due to the very aggressive performance targets of the Intel Pentium 4 processor and other constraints specific to portions of the design, varying degrees of automation were used.

Gate-level netlists were created both by directly synthesizing Register Transfer Level (RTL) code and by manually drawing schematics. Cell placement and interconnect routing were generated using both automatic techniques and manual specification. Design verification included domains such as logic, timing, noise, and reliability.

One of the main challenges in the CBD flow was design convergence. Standard circuit and layout techniques were applied to solve timing problems including max-delay, min-delay, and max slope. Wire-spacing, shielding, buffer sizing, and other solutions were used to address noise issues. Reliability convergence for electromigration and self heat was achieved mainly through slope fixing, wire sizing, and thermal simulation. In addition, the CBD flow had to automatically perform design completion tasks such as scan insertion, scan chain hook-up, clock tree synthesis, and sizing of clock buffers.

The CBD flow was used across the board on the Intel Pentium 4 processor to implement a variety of design blocks. Some designs such as cache, register files, domino and analog circuits were implemented using custom techniques. Table 1 shows the percentage of area, transistor count, and cell count in the CBD and non-CBD sections of the chip, where cell count in the non-CBD areas refers to the number of custom cells.

Table 1: CBD usage in the Pentium® 4 processor

	CBD	Non-CBD
Area	52%	48%
Cell count	44%	56%
Transistor count	50%	50%

CELL LIBRARY DESIGN

Library Content

The library consisted of over 1600 cells, covering over 130 unique logic functions, and associated collateral included over 20 file types. Most cells had variants implemented with different threshold voltage transistors to enable trade-offs between delay and leakage power. The percentage of library content, usage, and effort for different cell types is shown in Figure 1. The library content included complex cells that were aimed at device-dominated, high-speed datapath and control blocks. Buffers and sequentials for use as repeaters and drivers for global nets were also included in the library. Sequentials included latches and flip-flops with a wide variety of functions, and scan and non-scan variants for almost every type. As shown in Figure 1, although the sequentials made up less than 45% of the library, they accounted for more than 65% of the library effort, due to their design and characterization complexity.

A wide range of drive strengths was designed for each logic family. Drive strengths were selected to maximize transistor density within the cells as much as possible, while providing adequate granularity of drive strengths for optimal tuning of paths.

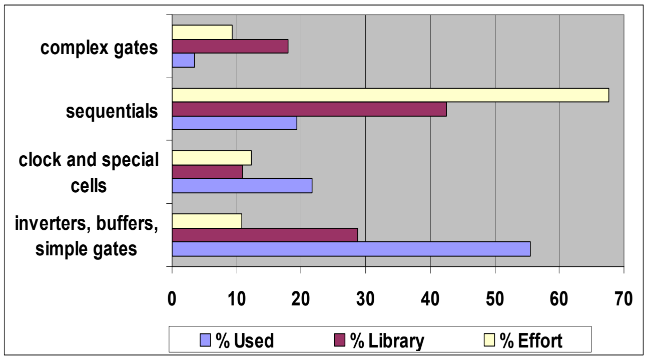

Figure 1: Library percentage content, effort, and usage

Library Architecture

The architectural specification is what distinguishes a CBD library from a random collection of cells. The scope of this specification is broad–including naming convention, physical design template, drive-strength definitions, electrical and physical design guidelines, and methodology for verification and for production of collateral for chip design. For the Intel Pentium 4 processor project, the fundamentals of the library architecture were defined very early in conjunction with the device, design rule, and fabrication technology definition for the 90nm generation [2,3]. It was revised multiple times before production versions of the library were used in the final design convergence of the Intel Pentium 4 processor.

The physical architecture of the library, Figure 2, is row-based. All cells are 15 metal 2 (M2) tracks tall and a variable, but integer, number of metal 3 (M3) tracks wide. It features wide M2 power busing over the devices and 11 M2 tracks for signals. The outer tracks were useful for routing on metal 1 (M1) and poly, thereby minimizing the need for M2 for intra-cell routing, even in complex cells. No M3 was used inside the cells. Pins in the cell were in M1 in nearly all cases, and they conformed to a specification that balanced cell area against block place-and-route efficiency.

The cell template and rules for internal cell routing were carefully designed to restrict the delay due to resistance*capacitance (RC) to an acceptable level, while minimizing cell area and the use of upper levels of metal. For example, RC delay in the polysilicon layer was acceptably low even when **p** or **n** device widths filled the cell height, as illustrated in Figure 2, but would have precluded the use of a wider gate, even if the cell height was taller. Poly routing was used to cross under M1 routing in the cells (as an alternative to M2 routing) only if the resulting RC delay was acceptable. All stages with the fastest delay targets use full metal strapping of source and drain diffusions, but limited use of diffusion without metal strapping was allowed on series-connected devices or other devices with longer delays. Netlists, including resistances, were extracted from the cell layout to ensure performance and margin verification accuracy.

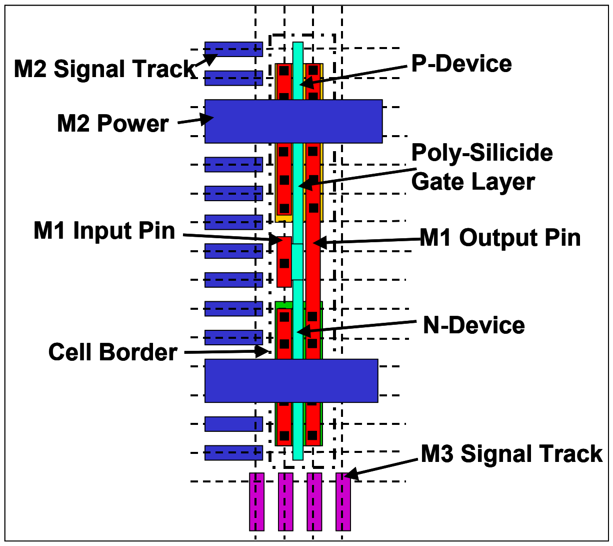

Figure 2: Illustration of standard cell architecture

Cell Design Methodology

P/N ratios for single-stage gates (such as inverter, nand, nor, and-or-invert and or-and-invert) were determined by path-based delay optimizations. Average delay was minimized for a given total gate area, under nominal loading conditions. These ratios were typically determined once per logical family and applied across all of its drive strengths.

Multi-stage gates (such as muxes, xor/xnor gates, etc.) were individually optimized across all drive strengths to minimize objective functions specified by the designers. Typically, the objective was to minimize a power-delay product under nominal loading conditions, while meeting any specified constraints for noise margins, transition times, etc.

Sequential cells are special cases of multi-stage gates. The objective functions for optimization included power-delay trade-offs, noise, charge-sharing, stability, and side-pin (reset/preset/enable) constraints. We modeled each of these constraints under the appropriate skew and voltage conditions. For master-slave flipflops, our key objective was the minimization of its black-hole time (clock-to-out + setup). Prior to the design execution, we performed several experiments to select the optimal activity factors, driver sizes, waveform shapes, noise criteria, length of clock chains, P/N ratio of output driver, and pass-gate size restrictions. We then leveraged these results during the design of all sequential families.

Individual sizing of each drive strength helped achieve optimal transistor sizes. For example, smaller sized latches were more susceptible to noise and less prone to restore time failures, and we sized the transistors appropriately to reflect this.

The library included phase-1 and phase-2 scan versions of all sequential elements. In phase-1 scan sequentials, the clock is in phase with the scan clock, and in phase-2 versions, the clock is out of phase with the scan clock. Min-delay and max-delay constraints from the scan-load/scan-store operations primarily drove the sizing of scan circuitry. Leakage power considerations drove the choice of non-minimum transistor lengths. Scan cells were built by taking the regular sequential elements and adding a scan gadget on top of them. Using the same gadget sizes for all sequential cells helped minimize design work. The combined scan cell was validated for noise and delay constraints. (See “Full Hold-Scan Systems in Microprocessors: Cost/Benefit Analysis” in this issue of the *Intel Technology Journal* for additional details on scan.)

We targeted some family types, such as clock buffers and min-delay cells, for specific operating conditions. They had their unique optimization methods under restricted delay and transition time domains. Cells compatible with Focused Ion Beam editing, de-coupling capacitors, and other layout completion cells were driven by layout and process requirements.

The 90nm process used in the Pentium 4 processor design featured a choice of dual threshold-voltage transistors. These can be used for power-performance trade-offs. In the library, low Vt transistors were primarily used to gain speed-up for the same layout footprint. These cells were generated from the nominal versions by converting selected devices to low Vt, without transistor size changes.

A host of internally developed automation tools helped ensure high productivity even when cells were individually optimized. These included tools for circuit optimization, parasitic and reliability estimation, low Vt variant generation, and layout automation. Once a design was set up for optimization, it propagated through process file revisions, design target changes, creation of design variants, and the addition of new drive strengths with minimal effort.

Library Qualification

Once a library cell is designed, it must be qualified to meet its logic, circuit, and layout design specifications. Logic equivalence tools were used to validate that each cell implemented the logic function for which it was designed. A host of validation checks was performed to guarantee circuit functionality and performance across wide ranges of temperature, power supply, signal slopes, and process skews. Functional checks for circuits covered noise margins, writability, and node recovery times. Performance checks included delay, setup/hold times, and maximum signal slopes.

In addition to manual reviews, cell layout was checked by a cell architecture verification tool to ensure that it was compatible with the place and route tool. The reliability checks on each cell included those for electromigration, self-heat, and IR drop on the power rails.

Multiple library releases were made during the course of the project. Regression tests were used to ensure that a new library release did not significantly perturb the existing state of the design. Pilot blocks were re-created using the new library to gauge the impact on the design before the library release.

CELL LIBRARY MODELING

Modeling for Timing

To ensure accurate modeling of cell timing, the mathematical timing models used by the static timing analysis tools (typically of the form *timing=f(C_L, TT_{in})* where C_L is the external load of the cell and TT_{in} is the transition time of the input) were compared against dynamic simulations at each characterized value of the input parameters. This was done for every cell, and problematic cells were studied for further action, which often simply involved recognizing that the models showed large errors for the input parameter ranges at which the cells were rarely used.

Setup time modeling for latches and flip-flops was done based on a set of criteria that involved constraints on storage node transitions at the clock arrival time and acceptable errors in cell delay timings at setup, with respect to delay timings at infinite separation between data and clock. Ideally, in order to reduce modeling errors we would want zero errors in cell delay timings. However, this approach is impractical as a flip-flop's black hole time (clock-to-out delay plus setup time) may not be optimal at this point. Therefore, a point was chosen for the setup time modeling that gave optimal timing at the cost of an acceptable error between cell delays at setup and infinite setup (see Figure 3).

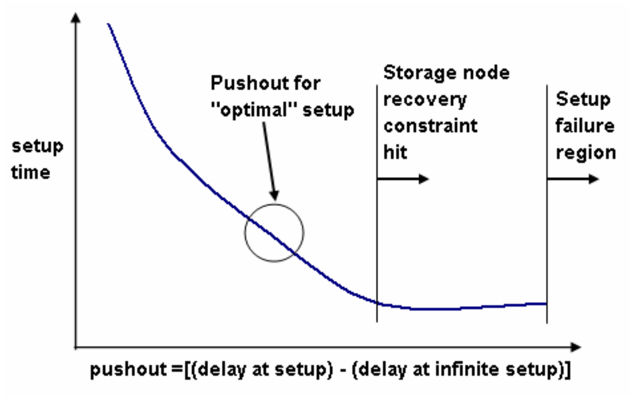

Figure 3: Definition of setup time

Some cells are very sensitive to the environment in which they are placed, which creates timing modeling challenges. One example is a cell that has input pins directly connected to pass-gates. For such cells, large differences in timing due to charge sharing can be seen for timing arcs that involve the switching on of the pass-gates. The charge sharing problem is worse for large pass-gates with weak external drivers. To mitigate this problem, cells with inputs tied directly to pass-gates were designed with restricted pass-gate transistor widths, and they were characterized with the assumption of the weakest possible driver allowed.

Typical cell timing characterization involves a single input transition causing an output transition, but in some instances it is possible that more than one pin transitions simultaneously. Such scenarios were handled for combinational cells by modeling possible increases or decreases in delay when more than one pin transitions at the same time. There are generally a large number of input parameter combinations that could be considered. To constrain the characterization effort, only the most important input parameter combinations were considered, while eliminating those combinations that were found to have a smaller impact on the timing of the cell.

Another interesting case of modeling timing was encountered in the fully decoded multiplexer cells, an example of which is shown in Figure 5. In such cells, only one select pin (sa or sb in the figure) can be on at any given time. Also, select-to-output timing is degraded if one of the select pins transitions high and the other one transitions low simultaneously, relative to the case where only one select pin transitions high with the other one held at a constant low. To model this timing difference, worst-case select-to-output delays were modeled by simultaneous switching of two select pins with one rising and the other falling, and the best case delays were modeled by only one select pin rising.

Modeling for Noise

In the CBD flow, any noise failure on a cell instance can be fixed only by changing the environment around the cell. Therefore, it is important to design cells with good noise immunity while maintaining acceptable timings.

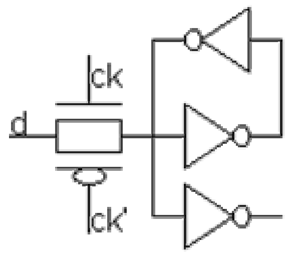

Figure 4: Noise-sensitive latch

An example of a cell in which the noise versus timing trade-off was critical is shown in Figure 4. The latch in the figure is sensitive to noise on both the d and ck input pins. An initial design with a very tight noise constraint turned out to have the same timing as a cell with an extra inverter in the d to output path. A revised noise spec based on a more reasonable assumption of noise attackers gave a design with much better timing. This made the design less robust to noise, but the number of noise failure cases at the CBD level was still manageable. Such cases were used as a reference to come up with suitable noise versus timing design trade-offs for other cells with noise-sensitive architectures. Noise specs were revised for some architectures to give timings that met expectations with the understanding that usage of such cells in CBD would be limited to cases where the noise levels are manageable.

Interconnect parasitics have a big impact on the timing and noise behavior of standard cells, especially when it comes to cross capacitance either between nets within the cell or between nets inside the cell and the ones surrounding the cell. For the library cells, the extraction tool modeled the cross capacitance between nets internal to the cell by looking at the actual geometry and routing of the nets. For the internal net to external net cross capacitance estimations, assumptions were made about what the external routing would look like and the capacitances were extracted based on those assumptions. For timing purposes, the external ends of these capacitances were grounded to prevent unrealistically pessimistic modeling. For noise purposes and for reporting noise behavior of cells to the CBD tools, actual attackers were assumed on the external ends of these capacitors to come up with the worst-case attacker switching scenario. A realistic "derating" of the attacker strength based on the type of layer of the cross capacitance was done to avoid making the results too pessimistic.

Modeling for Reliability

The primary reliability concerns addressed here are Electromigration (EM) and Self-Heating (SH) problems. This was a bigger concern in the 90nm process used for the Intel Pentium 4 processor because of the low thermal conductivity of the low-K inter-layer dielectric.

The analysis at the block level fell under a couple of different categories: (a) validating the cell internals for a given external loading and (b) validating the interconnect between the cells for a certain routing topology. Power grid validation for EM, SH, and voltage drop was a special case of the latter.

At the cell level, the characterization data were analyzed for the product's end-of-life operating condition. The EM/SH characterization data for cells depended on whether a net is a power or non-power (signal) net. For signal nets, the cell characterization data included the maximum average and root mean square (RMS) currents a cell pin could support. For power nets, the characterization produced a model of the cell's power network including resistors and current sinks.

At the block level, the data from cell-level characterization were used to do the roll-ups. The power grid was simulated to test the accuracy of the heat produced by the power nets. The temperature simulator generated a temperature map for the whole die, from which local temperatures for coarse pixels are determined. An EM violation could be waived by reducing the SH current in a neighboring wire that affects the temperature.

Modeling for Formal Verification

The CBD flow required a library rich in content, including cells whose logic did not fit the norm of standard cells in previous libraries.

Cells with constrained inputs needed adequate modeling to ensure that they are handled appropriately. As an example, consider the 2-to-1 multiplexer circuit shown in Figure 5.

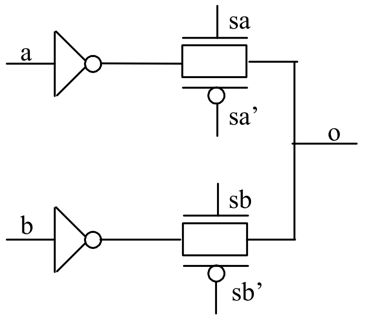

Figure 5: 2-to-1 multiplexer with decoded selects

The logical behavior for this circuit in HDL syntax is expressed as “if sa then a else if sb then b.” However, this description has two pitfalls: (1) it implies priority of the sa select over sb when both are active and (2) when both sa and sb are inactive it implies that the circuit holds its values, i.e., behaves as a latch. Moreover, the circuit implementation is such that both selects cannot be made active simultaneously since that can create a short-circuit path. For correct usage of this cell, the constraint that one and only one of the selects (sa or sb) must be active at all times must be met when using this cell at the block level.

Another example of a cell with constrained inputs is a fast XNOR gate with logic function “a*b + ab*bb” needed for high-speed applications. The a & ab inputs, as well as the b & bb inputs are assumed to be complementary signals. This assumption needed to be propagated correctly to the formal verification flow.

Some circuits such as the scan cells were too large for analysis at the transistor level. This issue was overcome by carefully decomposing the circuit into a hierarchy of logic functions, each of which could then be successfully verified.

Formal verification tools have used such constraints but at higher levels in the design hierarchy. The CBD flow necessitated their use even at the library level so that the cells could be black-boxed for logic validation.

Modeling for Place and Route

The advanced 90nm process technology used in the Intel Pentium 4 processor included complex layout design rules that are not adequately comprehended by current place and route tools.

Traditionally, placement tools assume that the spacing between cell layout polygons and the cell border is greater than or equal to half of the design rule spacing for each layer. Thus, the spacing constraints are satisfied even when cells are abutted against each other. The cell layouts had to be carefully designed and modeled to ensure this. Extra checks were added to the layout verification flow to guard against violations.

During routing, cells are traditionally modeled as abstracts consisting of *terminals* (target connection points for routing) and *obstacles* (areas where routes cannot be placed) This posed a couple of problems in the Intel Pentium 4 processor design, including the following:

1. False design-rule violations on obstacles: The obstacles within cells are assumed to originate from physical wiring within the layout and were expected to satisfy the layout design rules for the layer they were on. However, sometimes only part of the wiring on a net can be marked as a terminal for process or circuit performance reasons. This led to router issues because the unmarked segments created design-rule violations.

   As an example, consider the layout in Figure 6 where the via cover on output net O cannot be marked as terminal because it is too narrow to pass the reliability checks. In this case, routing connections must be made only to the wider segment. The existence of other wiring (Net X) prevents the via cover from being widened to pass the reliability check. The part of the via cover outside the fat segment must therefore be marked as obstacle to enforce this–and that part is not wide enough to meet design rules on its own.

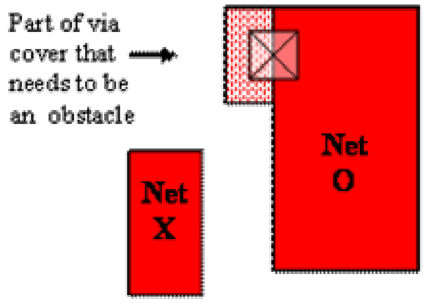

Figure 6: Modeling issue with obstacle

2. Fixed vs. variable spacing constraints: Adequate support for obeying spacing constraints based on local polygon density was not available (similar to the placement issue described above)

The first issue led to the loss of terminal area in some cell abstracts since obstacles had to be expanded to meet minimum width requirements. In some cells, layout rework was needed to satisfy the requirements of the routing tool. For the second issue, a workaround was developed whereby the shape of the obstacle generated was based on the local polygon density of the layer, and the routing tool used a spacing constraint value that ensured no violations were created by it. This workaround required special care during the layout design so that the generated obstacle shape satisfied all design rules as required by the previous limitation mentioned.

CONCLUSION

For CBD to be an effective methodology for a high-performance product like the Intel Pentium 4 processor, many considerations must be addressed during library design in order to use CBD widely without compromising the design. Library richness, in terms of logic functions, drive strengths, and collateral types, as well as an optimized architectural specification, including the physical design template and guidelines, play a key role. Both power and delay must be considered during cell optimization as well as other constraints for specific cell types. Because the cells are treated as black boxes in the CBD flow, it is critical that they are well-tested for functionality and performance and meet the requirements of the place-and-route tool and all process design rules. There are many modeling issues that must be addressed during the design of a library for CBD. Challenges include the treatment of pass-gate inputs, multiple-input switching, trade-offs between timing and noise robustness, cells with problematic logic descriptions, cells too large for formal verification at the transistor level, and complex layout design rules that may not be completely comprehended by the place and route tool. Careful consideration of these and other issues during the construction of the cell library helped enable the CBD methodology to be used to an unprecedented extent in the Pentium 4 processor.

ACKNOWLEDGMENTS

This library is the work of many more contributors than can be included in the author list, including the design engineers and graphics technicians on the library team, the tool developers, the users of the library, and several key technical leaders from both product and technology groups. Special thanks go to Carl Simonsen, who was the founder of much of the methodology used in this standard cell library. The authors also thank Tim Deeter, Deanna Hotchkiss, Donna Medeiros, Vijay Pitchumani, Stephen Rich, and Ian Young for reviewing this paper.

REFERENCES

[1] Northrop, G.A. and Lu, Pong-Fei, “A semi-custom design flow in high-performance microprocessor design,” in *Proceedings of the 2001 Design Automation Conference*, pp. 426-431.

[2] Jan, C.H. et. al., “90nm generation, 300mm wafer low k ILD/Cu interconnect technology,” in *Proceedings of the IEEE 2003 International Interconnect Technology Conference*, pp. 15-17.

[3] Thompson, S. et. al., “A 90nm Logic Technology Featuring 50nm Strained Silicon Channel Transistors, 7 layers of Cu Interconnects, Low k ILD, and 1 um2 SRAM Cell,” in *Proceedings of the 2002 International Electron Devices Meeting*, pp. 61-64.

AUTHORS’ BIOGRAPHIES

Barbara Chappell is an Intel principal engineer in an R&D design organization within the Technology Manufacturing Group. During her eight years with Intel, her contributions have been in the fields of synthesis and library methodologies, in evaluating the impact of process technology on design, and in circuit techniques for high-speed logic. Barbara holds an M.S.E.E. degree from the University of California at Berkeley. Her e-mail address is barbara.a.chappell at intel.com.

Amanda Duncan manages the Standard Logic Implementation group in Intel’s Logic Technology Development organization. She has been with Intel for seven years. Her interests include library design and modeling. She holds B.S., M.S., and Ph.D. degrees in Electrical Engineering from the University of Illinois at Urbana-Champaign. Her e-mail address is amanda.duncan at intel.com.

Kiran Ganesh manages the Cell Libraries Development group in the Design Technology organization. He has been at Intel for seven years. His primary interests include design and modeling of cell libraries and physical design. He holds a Bachelors degree in Electrical Engineering from the Indian Institute of Technology, Madras and a Masters degree in Computer Engineering from Syracuse University. His e-mail address is kiran.ganesh at intel.com.

Manoj Gunwani is a project leader in the Cell Libraries Development group and has been with Intel for seven years. His primary interests are in VLSI library and physical design. He holds Masters degrees in Computer Science and Neuroscience from Syracuse University, a Bachelors degree in Electronics Engineering from IIT, Chennai and also pursued doctoral studies in Electrical Engineering at Stanford University. His e-mail address is manoj.g.gunwani at intel.com.

Abhinav Sharma works as a library design engineer in the Cell Libraries Development group in Design Technology, Intel. His interests are standard cell libraries and their usage by automated synthesis and P&R. He received his MSE degree in Electrical Engineering from Arizona State University and his B. Tech degree in Electronics & Instrumentation from Nagarjuna University, India. His e-mail address is abhinav.sharma at intel.com.

Madhu Swarna is a design automation engineer in the Desktop Products Group. He has been at Intel for seven years. His primary interests are in design for low power, timing analysis and library modeling. He earned his Master’s degree in Computer Science from Texas A&M University in 1997. His e-mail address is madhu.swarna at intel.com.

THIS PAGE INTENTIONALLY LEFT BLANK

For further information visit:
developer.intel.com/technology/itj/index.htm