GEORGE A. RILEY (S.B. #118304) – griley@omm.com
MARK E. MILLER (S.B. #130200) – markmiller@omm.com
PETER OBSTLER (S.B. #171623) – pobstler@omm.com
CHRISTOPHER D. CATALANO (S.B. #208606) – ccatalano@omm.com
LUANN L. SIMMONS (S.B. #203526) – lsimmons@omm.com
O'MELVENY & MYERS LLP
Embarcadero Center West
275 Battery Street
San Francisco, CA 94111-3305
Telephone: (415) 984-8700
Facsimile: (415) 984-8701

Attorneys for Defendant
MAGMA DESIGN AUTOMATION, INC.

UNITED STATES DISTRICT COURT

NORTHERN DISTRICT OF CALIFORNIA

SAN FRANCISCO DIVISION

SYNOPSYS, INC., a Delaware Corporation,

Plaintiff,

v.

MAGMA DESIGN AUTOMATION, INC., a Delaware Corporation, AND LUKAS VAN GINNEKEN,

Defendants.

Case No. C04-03923 MMC

DECLARATION OF CARL SECHEN IN SUPPORT OF MAGMA'S MOTIONS FOR SUMMARY JUDGMENT AND IN OPPOSITION TO SYNOPSYS'S MOTIONS FOR PARTIAL SUMMARY JUDGMENT

I, Carl Sechen, declare:

1. I make this declaration based on my personal knowledge of the facts stated herein, and, if called as a witness, I could and would testify to these facts.

Qualifications

2. I hold a Bachelors degree in Electrical Engineering from the University of Minnesota, a Masters degree in Electrical Engineering from M.I.T., and a Ph.D. degree in Electrical Engineering from the University of California, Berkeley. I was an Assistant Professor of Electrical Engineering at Yale University from 1986 to 1990, and an Associate Professor of Electrical Engineering at Yale University from 1990 to 1992. I was an Associate Professor of Electrical Engineering at the University of Washington from 1992 to 1999. Since 1999 I have been a Professor of Electrical Engineering at the University of Washington. I was named a Fellow of the IEEE (Institute for Electrical and Electronic Engineers) in 2002.

3. I have graduated 17 Ph.D. students and am currently advising 10 Ph.D. students studying in my VLSI Design and CAD Lab at the University of Washington. In my 19-year career as a professor, I have written successfully funded research proposals for more than $10 million.

4. I have authored or co-authored 144 papers, one book, and two patents. I co-founded InternetCAD, Inc. in 1993, an integrated circuit placement and routing tool vendor.

5. A more detailed account of my work experience, qualifications and list of publications is included in my Curriculum Vitae, which is attached as Exhibit CCC.

6. In the last nine years, I have been deposed as an expert witness in two cases: (1) Cadence v. Avanti, US District Court (California); and (2) Trans Logic v. Hitachi, US District Court (Oregon).

7. I am being compensated for my time in this matter at my customary rate of $250/hour plus expenses. My compensation is not dependent on the outcome of this action.

8. I expect to testify in this case on the matters set forth in this declaration as well as on other matters that may arise in this litigation including, but not limited to, matters raised in expert reports submitted by the defendants in this case.

Scope of Tasks and Materials Reviewed

9. I have been retained by O'Melveny & Myers LLP, counsel for defendant Magma Design Automation, Inc. ("Magma"), to address issues relating to certain information from plaintiff Synopsys, Inc. ("Synopsys") and disclosures by Magma relating to Magma's technology, as well as other public disclosures.

10. In reaching my conclusions and preparing this report, I have reviewed or considered the following documents:

a. March 1991 Sutherland/Sproull article entitled *Speed on the Back of an Envelope* ("Sutherland et al."), a true and correct copy of which is attached hereto as Exhibit N;

b. November 5, 1995 Grodstein et al. article entitled *A Delay Model for Logic Synthesis of Continuously-Sized Networks* ("Grodstein, et al."), a true and correct copy of which is attached hereto as Exhibit P;

c. Two Synopsys draft patent applications entitled *System and Method for Constant Delay Synthesis* and *Method for Achieving Timing Closure of Digital Networks and Method for Area Optimization of Digital Networks Under Timing Closure*, true and correct copies of which are attached hereto as Exhibits TT and UU;

d. August 1996 van Ginneken ICCAD Paper Complete Draft entitled *Driving on the Left Hand Side of the Performance Speed-way* ("DrivingLHS"), a true and correct

copy of which is attached hereto as Exhibit U;

e. September 26, 1996 ICCAD '96 Advance Program ("ICCAD'96 brochure, intended tutorial abstract"), a true and correct copy of which is attached hereto as Exhibit Q;

f. November 13, 1996 Otten Slides Presented at ICCAD '96 entitled *Tuning for Speed* ("OttenSlides ICCAD '96"), a true and correct copy of which is attached hereto as Exhibit R;

g. Slides entitled *Gain-based synthesis* ("Magma98 (Lukas 'Gain based synthesis' slides)"), a true and correct copy of which is attached hereto as Exhibit A;

h. April 6, 1998 Otten paper entitled *Global Wires Harmful?* ("OttenISPD'98 (Global Wires Harmful?)"), a true and correct copy of which is attached hereto as Exhibit S;

i. June 15, 1998 Otten et al. paper entitled *Planning for Performance* ("OttenDAC'98 (Planning for Performance)"), a true and correct copy of which is attached hereto as Exhibit T;

j. van Ginneken paper entitled *Size Independent Synthesis* ("Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis)"), a true and correct copy of which is attached hereto as Exhibit VV;

k. van Ginneken slides from April 1999 ISPD Panel entitled *Synthesis Driven Layout* ("Lukas Slides of 4/22/99"), a true and correct copy of which is attached hereto as Exhibit B;

l. April 28, 1999 Magma press release ("Magma (press release, April 28, 1999)"), a true and correct copy of which is attached hereto as Exhibit K;

m. November 1999 Magma white paper entitled *Overview of Magma's FixedTiming*

Methodology ("Magma (Nov. 99 white paper)"), a true and correct copy of which is attached hereto as Exhibit C;

n. Groeneveld slides presented at ASP-DAC 2000 Panel entitled *Timing closure* ("Magma (ASPDAC2000Groeneveld)"), a true and correct copy of which is attached hereto as Exhibit D;

o. Groeneveld slides presented at EDP 2000 Workshop entitled *Design closure* ("Magma (EDPapril2000Groeneveld)"), a true and correct copy of which is attached hereto as Exhibit E;

p. Groeneveld slides presented at a DAC 2000 Panel entitled *Design Closure, hope or hype?* ("Magma (DAC2000PanelGroeneveld)"), a true and correct copy of which is attached hereto as Exhibit G;

q. Groeneveld slides presented at a DAC 2000 Tutorial entitled *Gain-based synthesis* ("Groeneveld DACtutorial2000") a true and correct copy of which is attached hereto as Exhibit F;

r. Publicly available versions of the deposition transcript of Lukas van Ginneken from April 26, 2005 and April 27, 2005.

11. In addition the documents listed above, I prepared this report based upon my education, experience and knowledge of the industry.

Analysis

12. Logic synthesis refers to the translation of high level descriptions of the functions that an integrated circuit must perform into an interconnected set of logic gates. (A logic gate performs a simple logical function, such as comparing two signals and producing a result.) Physical design refers to the actual physical placement and wiring of the logic gates on a silicon chip. Once the logic gates are placed and interconnected, each gate performs its specified

function and communicates the result to the next gate. The time that it takes for the gate to carry out its function and communicate the result is referred to as the delay. As the demand or "load" on a gate increases, the delay increases. Under the concept of constant delay, however, the delay for each gate is determined at the beginning of the design process and held constant throughout the remainder of the process. Increases in a gate's load imposed by changes in the design are accommodated by increasing the size of the gate to provide more electrical current so that the delay remains constant.

13. In preparing this declaration, I reviewed the two Synopsys patent applications and the paper entitled "Driving on the Left-Hand Side of the Performance Speedway" (collectively "the Synopsys Materials") to determine what information they disclose. I have found that all of the information disclosed in the Synopsys Materials can be decomposed into the following 11 concepts and techniques.

14. **Constant Delay.** The concept of holding the delay associated with each gate constant during logic synthesis and physical design.

15. **Constant Delay Synthesis.** The concept of applying constant delay to the synthesis of digital circuits (including structuring and mapping).

16. **Constant Delay Set Via Optimal Gain.** The concept of selecting the best gain for each gate and using that gain to determine the constant delay associated with that gate. (Gain is the ratio of a gate's output load to its input load.)

17. **Buffer Insertion For Area Minimization.** The concept of inserting a buffer (a gate that performs no logical function but boosts signal strength) for area minimization in the constant delay paradigm (to achieve the optimal stage effort and therefore the optimal gain for a stage).

18. **Sizing Driven Placement.** The concept of changing cell sizes during iterative placement with the objective of holding the delays of each cell constant.

19. **Net Weight Placement.** The concept of computing a weight for each net that reflects the degree to which additional load impacts overall circuit area, and applying those net weights during placement. (A net refers to the wiring between the output of one gate and an input of one or more other gates.)

20. **Continuous Gate Sizing.** The concept of employing continuous sizing of a gate to maintain a constant delay for that gate during logic synthesis and physical design.

21. **Discrete Gate Sizing.** The concept of employing discrete gate sizes with the objective of maintaining a constant delay for that gate during logic synthesis and physical design.

22. **Area Minimization.** The concept of formulating an equation that calculates the area of a circuit and using that equation to minimize the area while maintaining constant delay.

23. **Area Estimation.** The concept of computing a weight for each net and using those net weights to estimate circuit area in the constant delay paradigm.

24. **Stretching Constant Delays.** The concept of adjusting (or stretching, compressing, trimming, etc.) the constant delay for gates during the logic synthesis and physical design.

Comparison Of Concepts To Magma Disclosures

25. I next compared the concepts that I had identified in the Synopsys Materials with information in Exhibits A, B, C, D, E, F, G, and K, (collectively "the Magma Materials"). I understand that the Magma Materials were disclosed in publications and presentations made by Magma.

26. **Constant Delay.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of holding the

delay associated with each gate constant during logic synthesis and physical design.

➢ Synopsys Patent Application #1, Claim 1, Page 7: "… a) logic synthesis of the digital network, using network delay as an optimization goal, where the delay of each gate is presumed constant. b) Placement of the gates in a two dimensional plane. c) Sizing of the gates in the network such that the network meets the cycle delay as computed by step a)." ➢ DrivingLHS, page 1, left column and right column: "In the ICCAD proceedings of 1995, authors from Digital Equipment Corporation published two papers, in which they propose a synthesis method based on a delay model for continuously sizable libraries. In the first paper [1] Joel Grodstein et al. observe that if continuous sizing of gates is permitted, the delays of the gates can be held constant. A gate is sized by multiplying all transistor widths by the same *gate size*. … The delay of a gate can be held constant by increasing the width of the transistors proportionally to the capacitive load that needs to be driven. "In [1] Grodstein et al. propose to keep the delay constant, while expressing the size as a function of the load." "In a companion paper [2] Eric Lehman et al. use the constant delay model in a mapping algorithm." "In an earlier paper [5] the constant delay model is applied to fanout optimization." "The constant delay model is not really different than conventional models. The difference merely is which variable is held constant, when the load on a gate changes." "The essence of all these papers is that they reverse the causality between the delay and gate size. Normally, the *delay* of a gate is expressed as a function of its capacitive *load* and the *size* of the gate is	➢ Lukas Slides of 4/22/99, slide 4 "Freeze delays" prior to placement; "physical design must maintain timing" ➢ Groeneveld DACtutorial2000, slides 29, 30, 41 29: "fixed timing methodology" 30: "fixed timing" 41: a clear approach for how to set target delays ("delay budget") for each cell in a design 41: "Translate delay budgets into gains" ➢ Magma (press release, April 28, 1999), page 1 "… Magma's patent-pending FixedTiming™ methodology …" "system that achieves optimum timing sign-off (timing closure) at the *beginning* of the physical design process …" ➢ Magma (Nov. 99 white paper), pages 1, 3, 6, 12 Page 1: "This backgrounder puts the technology and ideas behind Magma's FixedTiming approach …" Page 1: "… FixedTiming which provides timing sign-off at the beginning of the physical design flow." Page 3: "FixedTiming methodology freezes the delays before physical design … final cell sizes will be determined during a sizing driven placement … fixed delays can be met based on the real wire load." Page 6: "In a nutshell, the FixedTiming methodology consists of the following major steps … 1 … Target libraries are read, as well as timing constraints set. … 2 … The design is … mapped onto SuperCells. All delays in the circuit are determined and frozen. 3 … The cells are placed and simultaneously sized to meet

assumed to be a given constant. … ➢ DrivingLHS, page 2, left column: "In the new formulation, the delay becomes a parameter of the design, which is adjusted by an optimization algorithm, and its effect on gate size is observed using load analysis. Since delay is now constant with respect to size and load, this approach will be called the constant delay model. Note that the delay of a gate does not need to remain constant throughout synthesis, but a change in delay is a 'conscious' design decision." ➢ DrivingLHS, page 3, left column: "It is interesting though that under these assumptions the delays on a path should be the same, apart from the intrinsic delay. It also should be noted that in the constant delay model, the *delay is constant by definition*."	the timing based on the actual loads." Page 6: "SuperCells are functional place holder cells that have a fixed delay but variable size. Blast Fusion automatically abstracts SuperCells out of the target library. … SuperCell that has a fixed delay, but variable area." Page 12: " … timing is constant …" "'Sizing Driven Placement' technology is the first to address this entirely new paradigm. It combines placement, gain driven sizing, re-optimization, buffer insertion and congestion avoidance routing all in one step." ➢ Magma (DAC2000PanelGroeneveld), pages 3, 5 3 "Fixed delay up-front, fix gate size later" 5: "Magma Fixedtiming" "Actively managing wire delay: Through automatic sizing (sizing-driven placement); Through buffer insertion" ➢ Magma (ASPDAC2000Groeneveld), pages 2, 12, 16, 19, 20, 28 2: "Fix timing up-front" "True integration of Synthesis with Place&Route" "Integrated solution, covering the entire flow" 12: "Timing is fixed, As a result, cell sizes change. But large cells and small cells cancel out: some get bigger, others smaller" 16: "We know the optimal size-ratio for delay…. without knowing the *exact* values" "We can use this to fix delay before the parasitics are known!" 19: "'Sizing-driven' placement" "The gain ratio (=Cout/Cin) is maintained is placement; Sizes change *during* placement; As a result, delay is (almost) constant" 20: "Summary FixedTiming" "Delay fixed; Cell Area unknown; Sum of areas determines chip size. (Additive)" 28: "Magma FixedTiming sizes right" "All cells have exactly the right strength for the capacitance they drive" ➢ Magma (EDP2000Groeneveld), pages

	8, 10, 14, 15 8: "Magma FixedTiming" timing + parasitics = size 10: "Timing is fixed, As a result, cell sizes change. But large cells and small cells cancel out: some get bigger, others smaller" 14: "'Sizing-driven' placement" "The gain ratio (=Cout/Cin) is maintained is placement; Sizes change *during* placement; As a result, delay is (almost) constant" 15: "Summary FixedTiming" "Delay fixed; Cell Area unknown; Sum of areas determines chip size. (Additive)" ➢ Magma98 (Lukas 'Gain based synthesis' slides), Pages 2, 4, 6 2: "Sutherlands Theory of Logical Effort" 2: "Constant Delay and Timing Closure" 4: "The variable part of the delay is the same for all gates regardless of function" 6: "Constant Delay: Pick delays upfront; Use a gain dependent delay model; Compare all gates to the inverter; The initial delay gives the best gain without inserting buffers"

27. **Constant Delay Synthesis.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of applying constant delay to the synthesis of digital circuits (including structuring and mapping).

➢ Synopsys Patent Application #2, Claim 1, Page 20 Constant delay synthesis consisting of "a method for the structuring and mapping of an unmapped digital network." "… using network slack as an optimization goal, where network slack is calculated assuming that the delay of the cells of the network is constant with respect to load." "c) Estimation of the area of the network based on net load." ➢ Synopsys Patent Application #2, Claim 13, Page 22 "(globally optimal mapping)" … b1) a	➢ Lukas Slides of 4/22/99, slide 6 "Layout Driven Synthesis" "Technology dependent synthesis: sizing, cloning, buffering, resynthesis, remapping" ➢ Groeneveld DACtutorial2000, slides 53, 54 70 53: some details on gain-based mapping 54: overall Magma constant delay flow 70: Cloning and restructuring ➢ Magma (press release, April 28, 1999), page 4

traversal of the network from primary inputs and registers from left to right, while choosing at each cell the fastest matching books from all available matching books, using the constant delays of the books and the fastest arrival times of the fanins of the matching book. b2) a traversal of the network from right to left, while choosing at each cell the fastest matching book from the candidates selected during the previous traversal." ➢ Synopsys Patent Application #2, Claim 20, Page 23 "A method for the placement and sizing of cells of a mapped digital network … d) Choosing a target delay for each cell; e) Computing the network slack using the target delays; f) Placement of the cells of the network; g) Sizing of the cells of the network such that the network meets the network slack as computed by step b)."	"Magma physical optimization techniques include many of the traditional optimization functions that are performed during the front-end design flow such as re-structuring, collapsing, re-mapping and cloning and adds additional techniques such as SuperCell™ continuous dynamic sizing and active load management." ➢ Magma (Nov. 99 white paper), pages 8, 12 Page 8: "Logic re-mapping and restructuring at various stages during the design." "Advanced 'sizing driven' placement tool …" "FixedTiming takes care of automatic buffer insertion and cell resizing." Page 8: "Automatic library analysis tools for qualifying individual library cells and building SuperCell." Page 12: " … timing is constant …" "'Sizing Driven Placement' technology is the first to address this entirely new paradigm. It combines placement, gain driven sizing, re-optimization, buffer insertion and congestion avoidance routing all in one step." ➢ Magma (DAC2000PanelGroeneveld), page 9 "Innovative Gain-based synthesis provides early feasibility feedback" ➢ Magma (ASPDAC2000Groeneveld), page 2 "Fix timing up-front" "True integration of Synthesis with Place&Route" "Integrated solution, covering the entire flow" ➢ Magma (EDP2000Groeneveld), page 3 "True integration of Synthesis with Place&Route" ➢ Magma98 (Lukas 'Gain based synthesis' slides), Page 16 "Fast mapping, sizing, placement algorithms"

28. **Constant Delay Set Via Optimal Gain.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of selecting the best gain for each gate and using that gain to determine the constant delay associated with that gate.

➢ Synopsys Patent Application #1, Claim 2, Page 7 "(Choice of constant delay) Claim 2, Page 7: "The method of claim 1, where the parameter C/S for each type of gate is chosen before step 1) and is used during step a) to calculate the cycle delay, said parameter being chosen to have the largest possible value such that a long chain of identical gates of this type, each gate in the chain having identical value of parameter C/S, said chain cannot have simultaneously improved delay and improved gain by adding a buffer at some point to the same chain, even when the parameter C/S is chosen optimally after adding the buffer." ➢ Synopsys Patent Application #2, Claim 3, Page 20 "the delay of the book being chosen by choosing a load size ratio C/S for each book, which is independent of the network." ➢ Synopsys Patent Application #2, Claim 5, Page 20 "… a parameter C/S for each book is chosen to have the largest possible value such that a long chain of identical books each cell in the chain having identical value of parameter C/S, said chain cannot have simultaneously improved delay and improved gain by adding a buffer at some point to the same chain, even when the parameter C/S is chosen optimally after adding the buffer."	➢ Lukas Slides of 4/22/99, slide 5 5: theory of logical effort (Sutherland et al.) ➢ Groeneveld DACtutorial2000, slide 56 "library analysis" shows how to determine initial intended delay ➢ Magma (press release, April 28, 1999), page 7 Page 7: "Timing sign-off is determined at the beginning of the physical design flow using a combination of optimization techniques and logical effort abstractions." Page 7: "Blast Fusion assigns cell drive strengths to every cell in the design (not just cells on selected critical paths) after determining the actual wire loads from the layout." ➢ Magma (Nov. 99 white paper), pages 3, 5, 6, 8, 14 Page 3: "FixedTiming methodology freezes the delays before physical design … final cell sizes will be determined during a sizing driven placement … fixed delays can be met based on the real wire load." Page 3: "FixedTiming employs the concepts of logical effort and gain that Magma has applied for the first time ever to a comprehensive EDA tool." Page 5: "The 'electrical effort' h is called the gain of the gate: it is the ratio of the output and the input capacitances. … This ratio of capacitances can be chosen within limits beforehand, and can be kept constant by gate sizing throughout the following

➢ Synopsys Patent Application #2, Claim 20, Page 23 "A method for the placement and sizing of cells of a mapped digital network … d) Choosing a target delay for each cell; e) Computing the network slack using the target delays; f) Placement of the cells of the network; g) Sizing of the cells of the network such that the network meets the network slack as computed by step b)." ➢ DrivingLHS, page 6, right column, Section 8 ("Library analysis"): "A better method is the continuous buffering assumption, based on the ideas of [3]." "The delay of an amplifier in the technology is chosen as follows. Consider a chain of N amplifiers, and a given gain requirement $H = h^N$ for them. For which stage h is the delay $D = N(p + gh)$ minimal? Since $N = \ln(H)/\ln(h)$ the delay is $D = \ln(H)/\ln(h)\,(p + gh)$. Now lets minimize the delay by choosing h: $\frac{\partial D}{\partial h} = \ln(H)\left[\left(\frac{1}{\ln(h)} - \frac{1}{\ln^2(h)}\right)g - \frac{1}{\ln^2(h)}\frac{1}{h}p\right] = 0$ "Which simplifies to $g\ln(h) - g - p/h = 0$. For a CMOS inverter, $p = ag$, which gives us $a = h\ln(h) - h$. for a between 0.8 and 2 the gain h is about 3.4 to 4.4. The answer given by [8] is that the stage gain should be e=2.718… this analysis ignored the parasitic delay p of the inverters."	design steps. … maintain that timing fixed through final layout." Page 6: "In a nutshell, the FixedTiming methodology consists of the following major steps … 1 … Target libraries are read, as well as timing constraints set. … 2 … The design is … mapped onto SuperCells. All delays in the circuit are determined and frozen. 3 … The cells are placed and simultaneously sized to meet the timing based on the actual loads." Page 6: "SuperCells are functional place holder cells that have a fixed delay but variable size. Blast Fusion automatically abstracts SuperCells out of the target library. … SuperCell that has a fixed delay, but variable area." Page 8: "Logic re-mapping and restructuring at various stages during the design." "Advanced 'sizing driven' placement tool …" "FixedTiming takes care of automatic buffer insertion and cell resizing." Page 8: "Automatic library analysis tools for qualifying individual library cells and building SuperCell." Page 14: " … Magma's patent pending effort based delay models." ➢ Magma (ASPDAC2000Groeneveld), pages 16, 18, 19 16: "We know the optimal size-ratio for delay…. without knowing the *exact* values" "We can use this to fix delay before the parasitics are known!" 18: Sutherland (1991) logical effort 19: "'Sizing-driven' placement" "The gain ratio (=Cout/Cin) is maintained is placement; Sizes change *during* placement; As a result, delay is (almost) constant" ➢ Magma (EDP2000Groeneveld), pages 13, 14 13: Sutherland (1991) logical effort 14: "'Sizing-driven' placement" "The gain ratio (=Cout/Cin) is maintained is placement; Sizes change *during* placement;

	As a result, delay is (almost) constant" ➢ Magma98 (Lukas 'Gain based synthesis' slides), Pages 6, 12 6: "Constant Delay: Pick delays upfront; Use a gain dependent delay model; Compare all gates to the inverter; The initial delay gives the best gain without inserting buffers" 12: "Methodology: Pick delays to meet delay constraints; Optimize gain; Check that system gain is OK; Meet the delays by sizing"

29. **Buffer Insertion For Area Minimization.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of inserting a buffer (a gate that performs no logical function but boosts signal strength) for area minimization in the constant delay paradigm (to achieve the optimal stage effort and therefore the optimal gain for a stage).

➢ Synopsys Patent Application #1, Claim 6, Page 9 "… the buffers being inserted on non-critical paths, as determined by subtracting the delay of the buffer from the slack of the path, while estimating the area reduction …" ➢ Synopsys Patent Application #2, Claims 6-7, Page 21 "… buffer insertion …, the buffers being inserted on paths with positive slack, as determined by subtracting the delay of the buffer from the slack of the path." "… the buffer is inserted if area is saved." ➢ DrivingLHS, page 5, left column, section 6 Determining whether or not a buffer should be added: "Additionally, a buffer can save area. This can be checked using area analysis. Assume a buffer *b* is added at the output of	➢ Lukas Slides of 4/22/99, slide 6 6: "Layout Driven Synthesis" "Technology dependent synthesis: sizing, cloning, buffering, resynthesis, remapping" ➢ Groeneveld DACtutorial2000, slide 60 "Buffering, cloning and restructuring are used to maintain delay during placement" ➢ Magma (press release, April 28, 1999), page 9 "FixedTiming methodology solves the problem because it delays fundamental decisions such as cell sizing (for all cells in the design) and wire loading (width, spacing, and buffering) until very late in the flow." ➢ Magma (Nov. 99 white paper), pages 8, 12 Page 8: "Logic re-mapping and restructuring at various stages during the

gate *g*. Let c' be the updated net loads, then $\Delta A = a^T (c - c')$. The area of the buffer is $A_b - a_b c_b$, which is added to the circuit. The buffer saves area if $a^T (c - c') > a_b c_b$."	design." "Advanced 'sizing driven' placement tool …" "FixedTiming takes care of automatic buffer insertion and cell resizing." Page 8: "Automatic library analysis tools for qualifying individual library cells and building SuperCell." Page 12: " … timing is constant …" "'Sizing Driven Placement' technology is the first to address this entirely new paradigm. It combines placement, gain driven sizing, re-optimization, buffer insertion and congestion avoidance routing all in one step." ➢ Magma (DAC2000PanelGroeneveld), page 5 "Magma Fixedtiming" "Actively managing wire delay: Through automatic sizing (sizing-driven placement); Through buffer insertion" ➢ Magma (ASPDAC2000Groeneveld), page 22 22: handling discrete libraries, cloning, buffering ➢ Magma98 (Lukas 'Gain based synthesis' slides), Page 13 "Inserting buffers always costs delay; Inserting buffers saves area"

30. **Sizing Driven Placement.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of changing cell sizes during iterative placement with the objective of holding the delays of each cell constant.

➢ Synopsys Patent Application #1, Claims 7, 8; Pages 9, 10 Claim 7: "… where step b) [placement] is performed by repeated partitioning steps, partitioning the gates in the network into two or more groups, each group being assigned to an subdivision of the plane, alternating the partitioning steps with	➢ Groeneveld DACtutorial2000, slides 43, 49, 60 43: "The gain ratio is maintained during placement" 49: "But large cells and small cells cancel out: some get bigger, others smaller" 60: "Buffering, cloning and restructuring are used to maintain delay during

sizing steps …" Claim 8: iterative placement improvement: "… repeatedly changing the location of one or two gates at a time, while performing a sizing step c) after each location change." ➢ Synopsys Patent Application #2, Claims 20-23, Page 23 Claim 20: "A method for the placement and sizing of cells of a mapped digital network … d) Choosing a target delay for each cell; e) Computing the network slack using the target delays; f) Placement of the cells of the network; g) Sizing of the cells of the network such that the network meets the network slack as computed by step b)." Claim 21: Executing step f) in Claim 20 "where step f) is performed in gradual steps, each step being followed by a sizing step g)" Claim 22: Placement by partitioning, a sizing step being executed after each iteration. Claim 23: Placement by iterative improvement (moving one or two cells at a time), a sizing step being executed after each iteration.	placement" ➢ Magma (press release, April 28, 1999), page 5 Page 5: "The physical engines included in Blast Fusion include a force-directed placer, detailed placer, global router, track router, and patent-pending "virtual gridless" detailed router." ➢ Magma (Nov. 99 white paper), pages 3, 6, 8, 12, 15 Page 3: "FixedTiming methodology freezes the delays before physical design … final cell sizes will be determined during a sizing driven placement … fixed delays can be met based on the real wire load." Page 3: "FixedTiming employs the concepts of logical effort and gain that Magma has applied for the first time ever to a comprehensive EDA tool." Page 6: "In a nutshell, the FixedTiming methodology consists of the following major steps … 1 … Target libraries are read, as well as timing constraints set. … 2 … The design is … mapped onto SuperCells. All delays in the circuit are determined and frozen. 3 … The cells are placed and simultaneously sized to meet the timing based on the actual loads." Page 6: "SuperCells are functional place holder cells that have a fixed delay but variable size. Blast Fusion automatically abstracts SuperCells out of the target library. … SuperCell that has a fixed delay, but variable area." Page 8: "Logic re-mapping and restructuring at various stages during the design." "Advanced 'sizing driven' placement tool …" "FixedTiming takes care of automatic buffer insertion and cell resizing." Page 8: "Automatic library analysis tools for qualifying individual library cells and building SuperCell." Page 12: " … timing is constant …" "'Sizing Driven Placement' technology is the first to address this entirely new paradigm. It combines placement, gain

	driven sizing, re-optimization, buffer insertion and congestion avoidance routing all in one step." Page 15: "Magma's FixedTiming technology solves the timing closure problem by establishing timing as an absolute constraint." "Blast Fusion assigns cell drive strengths to every cell in the design (not just on selected critical paths) after determining the actual wire loads from the layout." ➢ Magma (DAC2000PanelGroeneveld), pages 4, 5 4: "Actively managing wire delay: Through automatic sizing (sizing-driven placement)" 5: "Magma Fixedtiming" "Actively managing wire delay: Through automatic sizing (sizing-driven placement); Through buffer insertion" ➢ Magma (ASPDAC2000Groeneveld), page 19 "'Sizing-driven' placement" "The gain ratio (=Cout/Cin) is maintained is placement; Sizes change *during* placement; As a result, delay is (almost) constant" ➢ Magma (EDP2000Groeneveld), page 14 "'Sizing-driven' placement" "The gain ratio (=Cout/Cin) is maintained is placement; Sizes change *during* placement; As a result, delay is (almost) constant" ➢ Magma98 (Lukas 'Gain based synthesis' slides), Page 11 "Use realistic timing constraints; They must be met before placement; Need sizing driven placement"

31. **Net Weight Placement.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of computing a weight for each net that reflects the degree to which additional load impacts overall circuit area,

and applying those net weights during placement.

➢ Synopsys Patent Application #1, Claims 9,10, Page 10 deal with net weights for placement. "… the weight being proportional to the area divided by the typical load." ➢ Synopsys Patent Application #2, Claim 30, Page 24 ("Weighted Placement") … f1) The calculation of net weights that reflect the change of network area due to sizing as a function of net length; f2) Placement of the cells of the network where the weighted network net length is used as a placement objective, the weighted network net length being the sum of the weighted net lengths of all nets, each net length being multiplied by a weight."	➢ Magma98 (Lukas 'Gain based synthesis' slides), Pages 9, 10 9: Circuit area $A = a^T c$ is given as the usual equation for total area (*a* is the area sensitivity of a gate) 9: Gives formula for calculating net weights: "Weights are calculated from inputs to outputs" $A = a^T c = \sum a_i c_i$ $w_i = \sum_{j \in fanin} \frac{w_j}{h_{ij}} + a_i$ **w** = **Mw** + **a** 9: $A = a^T c = w^T L$, where L is the imposed capacitance, w is the weight vector 10: "Placement formulation: A = wL; The active area of the circuit can be expressed as a weighted sum of wire lengths" 10: "Note: Placement programs naturally like to optimize a weighted sum of wire lengths"

32. **Continuous Gate Sizing.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of employing continuous sizing of a gate to maintain a constant delay for that gate during logic synthesis and physical design.

➢ Synopsys Patent Application #1, Claim 3, Page 8 "The method of claim 2, where step c) is comprised of steps: d) Determining the parameter C/S for each gate according to delay and gain constraints on the network. e) Determining the size S for each gate by traversing the network, in the direction opposite to the direction of data flow, while calculating the load of each gate, the size S of said gate being determined by the parameter C/S and the load C of said gate, the load on the preceding gate being determined by the size of the fanout gates, iterating step e) until convergence."	➢ Lukas Slides of 4/22/99, slides 11 "sizing must be unrestricted" ➢ Magma (press release, April 28, 1999), pages 7, 9 Page 7: "Timing sign-off is determined at the beginning of the physical design flow using a combination of optimization techniques and logical effort abstractions." Page 7: "Blast Fusion assigns cell drive strengths to every cell in the design (not just cells on selected critical paths) after

➢ Synopsys Patent Application #2, Claim 15, Page 22 "(sizing algorithm) … starting at the primary outputs and traversing the network in the direction opposite to the data flow, while calculating the load of a cell, by summing over the fanout of the cell, the product of the load of the fanout cell divided by the gain plus the net load of the cell." ➢ Synopsys Patent Application #2, Claim 20, Page 23 "A method for the placement and sizing of cells of a mapped digital network … d) Choosing a target delay for each cell; e) Computing the network slack using the target delays; f) Placement of the cells of the network; g) Sizing of the cells of the network such that the network meets the network slack as computed by step b)." ➢ Synopsys Patent Application #2, Claims 24-25, Page 23-24 Claim 24: "(sizing) … g1) calculation of the net length based on the available placement information; g2) calculation of the capacitive load of the cells using the net length; g3) calculation of the sizes from the capacitive load." Claim 25: "(sizing algorithm) … starting at the primary outputs and traversing the network in the direction opposite to the data flow, while calculating the load a cell, by summing over the fanout of the cell, the product of the load of the fanout cell divided by the gain plus the net load of the cell."	determining the actual wire loads from the layout." Page 9: "FixedTiming methodology solves the problem because it delays fundamental decisions such as cell sizing (for all cells in the design) and wire loading (width, spacing, and buffering) until very late in the flow." ➢ Magma (Nov. 99 white paper), pages 3, 4, 6, 8 Page 3: "FixedTiming methodology freezes the delays before physical design … final cell sizes will be determined during a sizing driven placement … fixed delays can be met based on the real wire load." Page 3: "FixedTiming employs the concepts of logical effort and gain that Magma has applied for the first time ever to a comprehensive EDA tool." Page 4: "Instead of a fluctuating delay, Magma's FixedTiming methodology changing cell sizes." Page 6: "In a nutshell, the FixedTiming methodology consists of the following major steps … 1 … Target libraries are read, as well as timing constraints set. … 2 … The design is … mapped onto SuperCells. All delays in the circuit are determined and frozen. 3 … The cells are placed and simultaneously sized to meet the timing based on the actual loads." Page 6: "SuperCells are functional place holder cells that have a fixed delay but variable size. Blast Fusion automatically abstracts SuperCells out of the target library. … SuperCell that has a fixed delay, but variable area." Page 8: "Logic re-mapping and restructuring at various stages during the design." "Advanced 'sizing driven' placement tool …" "FixedTiming takes care of automatic buffer insertion and cell resizing." Page 8: "Automatic library analysis tools for qualifying individual library cells and building SuperCell."

	➢ Magma (DAC2000PanelGroeneveld), pages 4, 5 4: "Actively managing wire delay: Through automatic sizing (sizing-driven placement)" 5: "Magma Fixedtiming" "Actively managing wire delay: Through automatic sizing (sizing-driven placement); Through buffer insertion" ➢ Magma (ASPDAC2000Groeneveld), pages 11, 12, 19, 20, 28 11: timing + parasitics = size 12: "Timing is fixed, As a result, cell sizes change. But large cells and small cells cancel out: some get bigger, others smaller" 19: "'Sizing-driven' placement" "The gain ratio (=Cout/Cin) is maintained is placement; Sizes change *during* placement; As a result, delay is (almost) constant" 20: "Summary FixedTiming" "Delay fixed; Cell Area unknown; Sum of areas determines chip size. (Additive)" 28: "Magma FixedTiming sizes right" "All cells have exactly the right strength for the capacitance they drive" ➢ Magma (EDP2000Groeneveld), pages 8, 10, 14, 15 8: "Magma FixedTiming" timing + parasitics = size 10: "Timing is fixed, As a result, cell sizes change. But large cells and small cells cancel out: some get bigger, others smaller" 14: "'Sizing-driven' placement" "The gain ratio (=Cout/Cin) is maintained is placement; Sizes change *during* placement; As a result, delay is (almost) constant" 15: "Summary FixedTiming" "Delay fixed; Cell Area unknown; Sum of areas determines chip size. (Additive)"

	➢ Magma98 (Lukas 'Gain based synthesis' slides), Pages 12, 15, 16 12: "Methodology: Pick delays to meet delay constraints; Optimize gain; Check that system gain is OK; Meet the delays by sizing" 15: "Post placement: Pick exact discrete sizes" 16: "Fast mapping, sizing, placement algorithms"

33. **Discrete Gate Sizing.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of employing discrete gate sizes with the objective of maintaining a constant delay for that gate during logic synthesis and physical design.

➢ DrivingLHS, page 5, right column, Section 7 ("Discrete sizing algorithms"): "By using discrete gates in parallel, integer multiples of the gates can be created. … a duplicated gate can drive exactly twice the load at exactly the same delay. Different sizes can be mixed to create non-integer multiples."	➢ Lukas Slides of 4/22/99, slide 10 parallel sizing (for more drive strength) ➢ Groeneveld DACtutorial2000, slides 57-59, 64, 67 57-59: library discretization issues 60: "Buffering, cloning and restructuring are used to maintain delay during placement" 61: optimum wire buffering 64: paralleling cells for increased drive 67: What makes a good DSM library (few inputs, lots of drives, no multi-stage gates) ➢ Magma (ASPDAC2000Groeneveld), page 22 22: handling discrete libraries, cloning, buffering ➢ Magma98 (Lukas 'Gain based synthesis' slides), Page 15 15: "Post placement: Pick exact discrete sizes"

34. **Area Minimization.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of formulating an equation that calculates the area of a circuit and using that equation to minimize the area while maintaining constant delay.

➢ Synopsys Patent Application #1, Claim 5, Pages 8-9 "(Area estimation)" Claim 5, pages 8-9: "The method of claim 2, where the logic synthesis of step a) uses network area as an optimization goal, in addition to network delay, the network area being estimated by setting the parameter C/S for each gate according to the method of claim 2, followed by the method of step e) being used to estimate the sizes of the gates, and hence the area of the network." ➢ Synopsys Patent Application #2, Claim 1, Page 20 Constant delay synthesis consisting of "a method for the structuring and mapping of an unmapped digital network." "… using network slack as an optimization goal, where network slack is calculated assuming that the delay of the cells of the network is constant with respect to load." "c) Estimation of the area of the network based on net load." ➢ Synopsys Patent Application #2, Claim 14, Page 22 "(area estimation) … c3) calculation of the sizes from the capacitive load. c4) calculation of the network area by summation of the product of the book area times the cell size." ➢ DrivingLHS, page 4, right column, section 5 "Let q_i be the unscaled load, which consists of the wire load and any primary output load. The capacitance c_i at the output of a gate i is calculated as the unscaled load, plus the sum of the capacitance on the	➢ Magma98 (Lukas 'Gain based synthesis' slides), Pages 8, 9 8: Contains the usual equation for the capacitance at the output of gate i, where L is the imposed capacitance, h_{ij}'s are the gains: (1) $c_i = \sum_{j \in fanout} \frac{c_j}{h_{ij}} + L_i$ (2) $c = Mc + L$ (3) $c = (I-M)^{-1}L$ 9: Circuit area $A = a^T c$ is given as the usual equation for total area (a is the area sensitivity of a gate) 9: Gives formula for calculating net weights: "Weights are calculated from inputs to outputs" 9: $A = a^T c = w^T L$, where L is the imposed capacitance, w is the weight vector

fanout gates, divided by the gain h_{ij} of input i of fanout gate j. $$c_i = q_i + \sum_{j \in fanout(i)} \frac{c_j}{h_{ij}}$$ Using matrix notation, **c** is the vector of gate capacitances, **q** is the vector of unscaled gate loads, and **R** is the matrix of the reciprocal gain. The diagonal of **R** is 0 if we assume that no cell has an output connected to an input of itself. $$R = \begin{bmatrix} 0 & \frac{1}{h_{12}} & \frac{1}{h_{13}} & \dots \\ \frac{1}{h_{21}} & 0 & \frac{1}{h_{23}} & \dots \\ \frac{1}{h_{31}} & \frac{1}{h_{32}} & 0 & \dots \\ \dots & \dots & \dots & 0 \end{bmatrix}$$ The load equation can now be written as **c** = **q** + **Rc**. The capacitance can be solved as the fixed point of this equation: **c** = (**I**-**R**)$^{-1}$**q**. A positive real solution exists if the largest eigenvalue of **R** is less than 1: $\lambda_1 < 1$. To calculate the area of the network, it is assumed that the area of a gate depends linearly on the load of that gate. The *area sensitivity* of a gate with respect to its output load a_i (vector **a**) is the marginal increase in area as a result of an increase in load. Since the area is assumed to depend linearly on the load, this is the area of gate i driving 1 unit of load. The area of the network is now A = **a**^{T}**c**."	

35. **Area Estimation.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of computing a weight for each net and using those net weights to estimate circuit area in the constant delay paradigm.

➢ Synopsys Patent Application #2, Claim 8, Page 21 " … area savings are estimated using net weights which reflect the change of network area due to sizing as a function of net length."	➢ Magma98 (Lukas 'Gain based synthesis' slides), Pages 9, 10 9: Gives formulas for calculating net weights: $A = a^T c = \sum a_i c_i$

➢ Synopsys Patent Application #2, Claim 9, Page 21 "… calculation of net weights is performed by starting at the primary inputs and traversing the network in the direction of the data flow, while calculating the net weight of a cell by summing over the fanin of the cell the product of the net weight of the fanin cell divided by the gain plus the cells area/load sensitivity, as W_i = sum_j W_j / g_ij" ➢ Synopsys Patent Application #2, Claim 31, Pages 25-26 ("Calculation of net weights") … step f1) is performed by starting at the primary inputs and traversing the network in the direction of the data flow, while calculating the net weight of a cell by summing over the fanin of the cell the product of the net weight of the fanin cell divided by the gain plus the cells area/load sensitivity, as W_i = sum_j W_j / g_ij"	$w_i = \sum_{j \in \text{fanin}} \frac{w_j}{h_{ij}} + a_i$ $\mathbf{w} = \mathbf{M}\mathbf{w} + \mathbf{a}$ $A = \mathbf{a}^T\mathbf{c} = \mathbf{w}\mathbf{L}$ "Weights are calculated from inputs to outputs" 9: $A = a^T c = w^T L$, where L is the imposed capacitance, w is the weight vector 10: "Placement formulation: A = wL; The active area of the circuit can be expressed as a weighted sum of wire lengths" 10: "Note: Placement programs naturally like to optimize a weighted sum of wire lengths"

36. **Stretching Constant Delays.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Magma Materials (right column) of the concept of adjusting (or stretching, compressing, trimming, etc.) the constant delay for gates during the logic synthesis and physical design.

➢ Synopsys Patent Application #2, Claims 10-12, Page 21 Claim 10: "(stretching) … a1) choosing a delay for each, book, … a3) adjusting the delay of each cell based on the slack." Claim 11: "… the delay of each cell is adjusted equally among the stages which have the same slack." Claim 12: "… the delay is adjusted on each path, such that the slack of each path becomes 0." ➢ DrivingLHS, page 5, left and right column	➢ Magma98 (Lukas 'Gain based synthesis' slides), Page 14 14: "Delay trimming: Zero slack algorithm"

"Stretching is a method for area optimization by changing the delays of the gates. Note that here delay is an independent variable and size, load and area depend on the delay." "Stretching the delay of a gate increases its gain and decreases its size, thus saving area." "Section 2 [due to Sutherland, et al.] suggests an interesting and simple approach to stretching. According to the simplified path sizing problem in that section, the effort delay of each stage should be equal. Assuming that the effort delay of each stage already was equal, this means that the delay of each stage needs to be increased by the same amount, regardless of function, load or size."	

Comparison Of Concepts To Public Disclosures

37. I next compared the concepts that I had identified in the Synopsys Materials with information in Exhibits N, P, Q, R, S, T, and VV (collectively "the Public Materials").

38. **Constant Delay.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of holding the delay associated with each gate constant during logic synthesis and physical design.

➢ Synopsys Patent Application #1, Claim 1, Page 7: "… a) logic synthesis of the digital network, using network delay as an optimization goal, where the delay of each gate is presumed constant. b) Placement of the gates in a two dimensional plane. c) Sizing of the gates in the network such that the network meets the cycle delay as computed by step a)." ➢ DrivingLHS, page 1, left column and right column: "In the ICCAD proceedings of 1995, authors from Digital Equipment Corporation published two papers, in	➢ OttenSlides ICCAD '96, slides 32, 37, 49, 50, 51: 32: "If C_L/C_{in} is kept constant, delay doesn't vary either" (constant delay) 37: "Principle of uniform stage effort" (gain) (Sutherland/Sproul) 49: formula for area minimization Area = $\mathbf{a}^T\mathbf{c}$, where $\mathbf{c} = (\mathbf{I} - \mathbf{F})^{-1}\mathbf{q}$ ($\mathbf{q}$ is imposed cap.) 49: "constant delay" 50: "delays are constant" 51: "implicit sizing" "no iterative timing analysis" ➢ ICCAD'96 brochure, intended tutorial abstract, page 24:

which they propose a synthesis method based on a delay model for continuously sizable libraries. In the first paper [1] Joel Grodstein et al. observe that if continuous sizing of gates is permitted, the delays of the gates can be held constant. A gate is sized by multiplying all transistor widths by the same *gate size*. … The delay of a gate can be held constant by increasing the width of the transistors proportionally to the capacitive load that needs to be driven.

"In [1] Grodstein et al. propose to keep the delay constant, while expressing the size as a function of the load."

"In a companion paper [2] Eric Lehman et al. use the constant delay model in a mapping algorithm."

"In an earlier paper [5] the constant delay model is applied to fanout optimization."

"The constant delay model is not really different than conventional models. The difference merely is which variable is held constant, when the load on a gate changes."

"The essence of all these papers is that they reverse the causality between the delay and gate size. Normally, the *delay* of a gate is expressed as a function of its capacitive *load* and the *size* of the gate is assumed to be a given constant. …

- DrivingLHS, page 2, left column:

"In the new formulation, the delay becomes a parameter of the design, which is adjusted by an optimization algorithm, and its effect on gate size is observed using load analysis. Since delay is now constant with respect to size and load, this approach will be called the constant delay model. Note that the delay of a gate does not need to remain constant throughout synthesis, but a change in delay is a 'conscious' design decision."

- DrivingLHS, page 3, left column:

"It is interesting though that under these assumptions the delays on a path should be the same, apart from the intrinsic delay.

- "We will assume that, using custom layout cell generators, continuous gate sizing is possible."
- "We propose to make delay an independent variable, and remove both load and gate size from the delay equation"
- "This enables new formulations of many existing optimization algorithms and potentially affects the entire path from high level synthesis down to cell layout."
- "With constant delay, the size of a gate varies as a function of its load."
- "We will extend Sutherland & Sproul's theory of "Logical Effort" and apply it to logic synthesis."
- "In this context, we review the mapping algorithm of Eric Lehman, et al."

- OttenISPD'98 (Global Wires Harmful?), page 4:

Page 4 (describes constant delay or "fixed delays"):
"logic synthesis … should deliver a gate list with a fixed delay for every gate. Layout synthesis should produce a network in each gate causes exactly that delay. [6] calls that *constant delay synthesis*."
Page 4, right column:
"This leads to a new paradigm in synthesis [6,11- this is the ICCAD tutorial by Lukas]: any delay imposed by synthesis can be realized, provided that the sizes of the gates can be continuously adjusted."
"Under the constant delay paradigm …"

- OttenDAC'98 (Planning for Performance), page 1, page 5 (section 4):

Page 1, abstract: "… propose some solutions for a 'constant delay' methodology.
Page 5, Section 4 ("Layout Synthesis"):
" … layout synthesis should realize the functional blocks in such a way that the delays in the blocks do not exceed their

It also should be noted that in the constant delay model, the *delay is constant by definition.*"	timing budgets, or rather keep them right on target. … Layout synthesis should produce a network in which each gate causes exactly that delay. This is called *constant delay synthesis* [6 – Lehman, et al.]. Given a fixed delay for a gate, its size becomes a function of the output capacitance." "… buffer insertion can only serve as an area reduction trick." Page 5, Section 4.1 ("Fixed delay"): describes Sutherland's results and introduces the notion of fixed delay synthesis. "The important observation is that τ can be kept constant by fixing $f = C_{in}/C_L$. This leads to a new paradigm in synthesis [6,9]: any delay imposed by synthesis can be realized, provided that the sizes of the gates can be continuously adjusted. In [6] the authors show that the size of a gate varies linearly with the load under constant delay. This enables size assignment after logic synthesis has fixed the *scaling factor f* for all gates …" ➢ Grodstein, et al., page 1: Page 1, abstract, left column: "We present a new delay model for use in logic synthesis. A traditional model treats the area of a library cell as constant and makes the cell's delay a linear function of load. Our model is based on a different, but equally fundamental linearity in the equation relating area, delay, and load: namely, we may keep a cell's delay constant by making its area a linear function of load." Page 1, left column: "We propose a new model for continuously-sized CMOS gates. In this model, a cell's delay will be held constant. As the cell's load changes, the cell's size automatically grows exactly enough to hold delay constant; making its area a function -- in fact a linear function -- of load." Page 1, right column: "Our own application is for continuously-sized, full custom designs. However, the delay model is also applicable to other methodologies,

	such as high-end standard cell, where there are many sizes of each cell. Essentially, it applies to any technology where cell sizing to obtain a desired delay is viable." Page 1, right column: "Constant-delay modeling has been used frequently in technology-independent algorithms." Page 1, right column: "Our approach is different. We use the constant-delay model in a technology-mapped realm as the synthesis tool's most exact library representation." Page 1, right column: "A companion paper [14] takes a different approach. Just as previous works have used a simple delay model to explore many different structurings of a technology independent network, [141 uses our delay model's computational simplicity to explore a wide space of network restructurings in the mapped realm." ➢ Sutherland et al. ➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), pages 1, 2, 3: Page 1, left column, abstract: "This paper applies to the concept of 'logical effort' to logic synthesis. The concept of logical effort is used to motivate the size independent delay model for logic synthesis. Methods to determine … initial delays for library cells are discussed. Area analysis and delay optimization methods based on 'electrical effort' or gain are discussed." "… continuous gate sizing as the best way to use the methodology, it shows that discrete libraries can also be supported. The method uses delay/gain as an independent variable as opposed to gate size which is traditionally used." Page 1, left column: "In the ICCAD proceedings of 1995, two papers were published by authors from Digital Equipment Corporation, which proposed a synthesis method based on a delay model

	for continuously sizable libraries. In the first paper [1] Joel Grodstein et al. observe that if continuous sizing of gates is permitted, the delays of the gates can be held constant. A gate is sized by multiplying all transistor widths by the same *gate size*. … The delay of a gate can be held constant by increasing the width of the transistors proportionally to the capacitive load that needs to be driven. "In [1] Grodstein et al. propose to keep the delay constant, while expressing the size as a function of the load." Page 1, left column: "In a companion paper [2] Eric Lehman et al. use the constant delay model in a mapping algorithm." Page 1, left column: "In an earlier paper [5] the constant delay model is applied to fanout optimization." Page 1, right column: What is alleged to be new over Sutherland and Grodstein is 'stretching' and 'compressing': "There is a significant difference between the constant delay model as proposed in [1, Grodstein] and the size independent model which is derived from [3, Sutherland, et al.] and applied to synthesis in this paper. In the constant delay model, the delay is determined during initial library analysis and kept constant throughout synthesis. In the size independent delay model on the other hand, the library is analyzed to determine the size independent delay model for the library as well as for determining initial conditions, but the delay of a gate may be changed during the course of synthesis as determined by appropriate delay optimization techniques. Page 1, right column: "The size independent model is no different than conventional models. The difference merely is which variable is held constant, when the load on a gate changes." Page 1, right column:

	"The essence of all the papers discussed is that they reverse the relationship between the delay and gate size. Normally, the *delay* of a gate is expressed as a function of its capacitive *load* and the *size* of the gate is assumed to be a given constant." Page 2, left column: "In the new formulation, the delay becomes a parameter of the design, which is adjusted by an optimization algorithm, and its effect on gate size is observed using area analysis. Since delay is determined independent of size, this approach will be called the *size independent delay model*. Note that the delay of a gate does not need to remain constant throughout synthesis, but a change in delay is a 'conscious' design decision." Page 3, left column: "It is interesting though that under these assumptions the delays on a path should be the same, apart from the intrinsic delay." Page 3, right column (Section 4, "Size independent synthesis"): "Evaluating a synthesis change in the network becomes easier: since the delays are size independent in the new model, the amount of time to recalculate slacks can be reduced."

39. **Constant Delay Synthesis.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of applying constant delay to the synthesis of digital circuits (including structuring and mapping).

➢ Synopsys Patent Application #2, Claim 1, Page 20 Constant delay synthesis consisting of "a method for the structuring and mapping of an unmapped digital network." "… using network slack as an optimization goal, where network slack is calculated assuming that the delay of the cells of the network is constant with respect to load." "c) Estimation of the area of the network based on net load."	➢ OttenSlides ICCAD '96, slide 51 51: "implicit sizing" "no iterative timing analysis" ➢ ICCAD'96 brochure, intended tutorial abstract, page 24 • "We will assume that, using custom layout cell generators, continuous gate sizing is possible."

➢ Synopsys Patent Application #2, Claim 13, Page 22 "(globally optimal mapping)" … b1) a traversal of the network from primary inputs and registers from left to right, while choosing at each cell the fastest matching books from all available matching books, using the constant delays of the books and the fastest arrival times of the fanins of the matching book. b2) a traversal of the network from right to left, while choosing at each cell the fastest matching book from the candidates selected during the previous traversal." ➢ Synopsys Patent Application #2, Claim 20, Page 23 "A method for the placement and sizing of cells of a mapped digital network … d) Choosing a target delay for each cell; e) Computing the network slack using the target delays; f) Placement of the cells of the network; g) Sizing of the cells of the network such that the network meets the network slack as computed by step b)."	• "We propose to make delay an independent variable, and remove both load and gate size from the delay equation" • "This enables new formulations of many existing optimization algorithms and potentially affects the entire path from high level synthesis down to cell layout." • "With constant delay, the size of a gate varies as a function of its load." • "We will extend Sutherland & Sproul's theory of "Logical Effort" and apply it to logic synthesis." • "In this context, we review the mapping algorithm of Eric Lehman, et al." ➢ OttenISPD'98 (Global Wires Harmful?), page 4, right column "This leads to a new paradigm in synthesis [6,11- this is the ICCAD tutorial by Lukas]: any delay imposed by synthesis can be realized, provided that the sizes of the gates can be continuously adjusted." "Under the constant delay paradigm …" ➢ OttenISPD'98 (Global Wires Harmful?), page 5, left column "Technology mapping can be also efficient under the paradigm." "… layout synthesis is capable of realizing gates with a priori imposed delays." ➢ OttenDAC'98 (Planning for Performance), page 5 (section 4) • Page 5, Section 4 ("Layout Synthesis"): " … layout synthesis should realize the functional blocks in such a way that the delays in the blocks do not exceed their timing budgets, or rather keep them right on target. … Layout synthesis should produce a network in which each gate causes exactly that delay. This is called *constant delay synthesis* [6 – Lehman, et al.]. Given a fixed delay for a gate, its size becomes a function of the output capacitance." "… buffer insertion can only serve as an

	area reduction trick." • Page 5, Section 4.1 ("Fixed delay"): describes Sutherland's results and introduces the notion of fixed delay synthesis. "The important observation is that τ can be kept constant by fixing $f = C_{in}/C_L$. This leads to a new paradigm in synthesis [6,9]: any delay imposed by synthesis can be realized, provided that the sizes of the gates can be continuously adjusted. In [6] the authors show that the size of a gate varies linearly with the load under constant delay. This enables size assignment after logic synthesis has fixed the *scaling factor f* for all gates …" ➢ Grodstein, et al., page 1 • Page 1, abstract, left column: "We present a new delay model for use in logic synthesis. A traditional model treats the area of a library cell as constant and makes the cell's delay a linear function of load. Our model is based on a different, but equally fundamental linearity in the equation relating area, delay, and load: namely, we may keep a cell's delay constant by making its area a linear function of load." • Page 1, left column: "We propose a new model for continuously-sized CMOS gates. In this model, a cell's delay will be held constant. As the cell's load changes, the cell's size automatically grows exactly enough to hold delay constant; making its area a function -- in fact a linear function -- of load." • Page 1, right column: "Our own application is for continuously-sized, full custom designs. However, the delay model is also applicable to other methodologies, such as high-end standard cell, where there are many sizes of each cell. Essentially, it applies to any technology where cell sizing to obtain a desired delay is viable."

	• Page 1, right column: "Constant-delay modeling has been used frequently in technology-independent algorithms." • Page 1, right column: "Our approach is different. We use the constant-delay model in a technology-mapped realm as the synthesis tool's most exact library representation." • Page 1, right column: "A companion paper [14] takes a different approach. Just as previous works have used a simple delay model to explore many different structurings of a technology independent network, [141 uses our delay model's computational simplicity to explore a wide space of network restructurings in the mapped realm." ➢ Sutherland et al. ➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), page 3 (Sections 4, 5) "Size independent synthesis"): "Evaluating a synthesis change in the network becomes easier: since the delays are size independent in the new model, the amount of time to recalculate slacks can be reduced."

40. **Constant Delay Set Via Optimal Gain.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of selecting the best gain for each gate and using that gate to determine the constant delay associated with that gate.

➢ Synopsys Patent Application #1, Claim 2, Page 7 "(Choice of constant delay) Claim 2, Page 7: "The method of claim 1, where the parameter C/S for each type of gate is chosen before step 1) and is used during step a) to calculate the cycle delay, said parameter being chosen to have the largest possible value such that a long chain of identical gates of this type, each	➢ OttenDAC'98 (Planning for Performance), page 5, section 4.1 ("Fixed Delay") Describes Sutherland's results and introduces the notion of fixed delay synthesis. "The important observation is that τ can be kept constant by fixing $f = C_{in}/C_L$. This leads to a new paradigm in synthesis [6,9]:

gate in the chain having identical value of parameter C/S, said chain cannot have simultaneously improved delay and improved gain by adding a buffer at some point to the same chain, even when the parameter C/S is chosen optimally after adding the buffer." ➢ Synopsys Patent Application #2, Claim 3, Page 20 "the delay of the book being chosen by choosing a load size ratio C/S for each book, which is independent of the network." ➢ Synopsys Patent Application #2, Claim 5, Page 20 "… a parame ter C/S for each book is chosen to have the largest possible value such that a long chain of identical books each cell in the chain having identical value of parameter C/S, said chain cannot have simultaneously improved delay and improved gain by adding a buffer at some point to the same chain, even when the parameter C/S is chosen optimally after adding the buffer." ➢ Synopsys Patent Application #2, Claim 20, Page 23 "A method for the placement and sizing of cells of a mapped digital network … d) Choosing a target delay for each cell; e) Computing the network slack using the target delays; f) Placement of the cells of the network; g) Sizing of the cells of the network such that the network meets the network slack as computed by step b)." ➢ DrivingLHS, page 6, right column, Section 8 ("Library analysis"): "A better method is the continuous buffering assumption, based on the ideas of [3]." "The delay of an amplifier in the technology is chosen as follows. Consider a chain of N amplifiers, and a given gain requirement $H = h^N$ for them. For which stage h is the delay $D = N(p + gh)$	any delay imposed by synthesis can be realized, provided that the sizes of the gates can be continuously adjusted. In [6] the authors show that the size of a gate varies linearly with the load under constant delay. This enables size assignment after logic synthesis has fixed the *scaling factor f* for all gates …" ➢ Sutherland et al. ➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), pages 2, 3, Section 10 (pages 5, 6) Page 2, left column: "In the new formulation, the delay becomes a parameter of the design, which is adjusted by an optimization algorithm, and its effect on gate size is observed using area analysis. Since delay is determined independent of size, this approach will be called the *size independent delay model*. Note that the delay of a gate does not need to remain constant throughout synthesis, but a change in delay is a 'conscious' design decision." Page 3, left column: "It is interesting though that under these assumptions the delays on a path should be the same, apart from the intrinsic delay." Page 3, right column (Section 4, "Size independent synthesis"): "Evaluating a synthesis change in the network becomes easier: since the delays are size independent in the new model, the amount of time to recalculate slacks can be reduced." • Page 5, right column, Section 10 (Library analysis): "A better method is the continuous buffering assumption, based on the ideas of [3]." [Sutherland, et al.] Then, this follows Sutherland, et al., as well: "The delay of an amplifier in the technology is chosen as follows. Consider a chain of N amplifiers, and a given gain requirement $H = h^N$ for them. For which stage h is

minimal? Since $N = \ln(H)/\ln(h)$ the delay is $D = \ln(H)/\ln(h)\ (p + gh)$. Now lets minimize the delay by choosing h: $\frac{\partial D}{\partial h} = \ln(H)\left[\left(\frac{1}{\ln(h)} - \frac{1}{\ln^2(h)}\right)g - \frac{1}{\ln^2(h)}\frac{1}{h}p\right] = 0$ "Which simplifies to $g\ln(h) - g - p/h = 0$. For a CMOS inverter, $p = ag$, which gives us $a = h\ln(h) - h$. for a between 0.8 and 2 the gain h is about 3.4 to 4.4. The answer given by [8] is that the stage gain should be e=2.718… this analysis ignored the parasitic delay p of the inverters."	the delay $D = N(p + gh)$ minimal? Since $N = \ln(H)/\ln(h)$ the delay is $D = \ln(H)/\ln(h)\ (p + gh)$. Now lets minimize the delay by choosing h: $\frac{\partial D}{\partial h} = \ln(H)\left[\left(\frac{1}{\ln(h)} - \frac{1}{\ln^2(h)}\right)g - \frac{1}{\ln^2(h)}\frac{1}{h}p\right] = 0$ "Which simplifies to $g\ln(h) - g - p/h = 0$. For an inverter, $p = ag$, which gives us $\ln(h) = h + a$. For a self-load between 0.7 and 0.9 for the amplifiers. If this is not ignored, the answer is closer to 3.6."

41. **Buffer Insertion For Area Minimization.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of inserting a buffer (a gate that performs no logical function but boosts signal strength) for area minimization in the constant delay paradigm (to achieve the optimal stage effort and therefore the optimal gain for a stage).

➢ Synopsys Patent Application #1, Claim 6, Page 9 "… the buffers being inserted on non-critical paths, as determined by subtracting the delay of the buffer from the slack of the path, while estimating the area reduction …" ➢ Synopsys Patent Application #2, Claims 6-7, Page 21 "… buffer insertion …, the buffers being inserted on paths with positive slack, as determined by subtracting the delay of the buffer from the slack of the path." "… the buffer is inserted if area is saved." ➢ DrivingLHS, page 5, left column, section 6	➢ OttenDAC'98 (Planning for Performance), page 5, section 4 • Page 5, Section 4 ("Layout Synthesis"): " … layout synthesis should realize the functional blocks in such a way that the delays in the blocks do not exceed their timing budgets, or rather keep them right on target. … Layout synthesis should produce a network in which each gate causes exactly that delay. This is called *constant delay synthesis* [6 – Lehman, et al.]. Given a fixed delay for a gate, its size becomes a function of the output capacitance." "… buffer insertion can only serve as an area reduction trick." ➢ Sutherland et al.

Determining whether or not a buffer should be added: "Additionally, a buffer can save area. This can be checked using area analysis. Assume a buffer *b* is added at the output of gate *g*. Let c' be the updated net loads, then $\Delta A = a^T (c - c')$. The area of the buffer is $A_b - a_b c_b$, which is added to the circuit. The buffer saves area if $a^T (c - c') > a_b c_b$."	➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), page 4 • Page 4 (right column), Section 7 ("Area optimization"), contains this on determining whether or not a buffer should be added: "Additionally, a buffer must save area. This can be checked using area analysis. Assume a buffer *b* is added at the output of gate *g*. Let c' be the updated net loads, then $\Delta A = a^T (c - c')$. The area of the buffer is $A_b - a_b c_b$, which is added to the circuit. The buffer saves area if $a^T (c - c') > a_b c_b$.

42. **Sizing Driven Placement.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of changing cell sizes during iterative placement with the objective of holding the delays of each cell constant.

➢ Synopsys Patent Application #1, Claims 7, 8; Pages 9, 10 Claim 7: "… where step b) [placement] is performed by repeated partitioning steps, partitioning the gates in the network into two or more groups, each group being assigned to an subdivision of the plane, alternating the partitioning steps with sizing steps …" Claim 8: iterative placement improvement: "… repeatedly changing the location of one or two gates at a time, while performing a sizing step c) after each location change." ➢ Synopsys Patent Application #2, Claims 20-23, Page 23 Claim 20: "A method for the placement and sizing of cells of a mapped digital network … d) Choosing a target delay for each cell; e) Computing the network slack using the target delays; f) Placement of the cells of the network; g) Sizing of the cells of the network such that the network meets the network slack as computed by step b)." Claim 21: Executing step f) in Claim 20 "where step f) is performed in gradual steps, each step being followed by a sizing	➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), page 5 Section 8 (page 5, left column) entitled "Placement and size independence". "The quality of the placement solution relies on the relative net weighting metric used" "Since placement depends strongly on the areas of the cells and their physical dimensions, any attempt to use the size independent network for placement must ensure that the area estimation is carried out with sufficient accuracy. Placement algorithms can then be modified to including sizing. For example one may consider a bipartitioning placement program that begins sizing the gates after a certain number of cuts based on Steiner estimates for wire length."

step g)" Claim 22: Placement by partitioning, a sizing step being executed after each iteration. Claim 23: Placement by iterative improvement (moving one or two cells at a time), a sizing step being executed after each iteration.	

43. **Net Weight Placement.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of computing a weight for each net that reflects the degree to which additional load impacts overall circuit area, and applying those net weights during placement.

➢ Synopsys Patent Application #1, Claims 9,10, Page 10 deal with net weights for placement. "… the weight being proportional to the area divided by the typical load." ➢ Synopsys Patent Application #2, Claim 30, Page 24 ("Weighted Placement") … f1) The calculation of net weights that reflect the change of network area due to sizing as a function of net length; f2) Placement of the cells of the network where the weighted network net length is used as a placement objective, the weighted network net length being the sum of the weighted net lengths of all nets, each net length being multiplied by a weight."	➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), page 5 Section 8 (page 5, left column) entitled "Placement and size independence". "The quality of the placement solution relies on the relative net weighting metric used" "Size independent synthesis provides a natural form of net weights. If we choose to associate with each net, the area sensitivity of the gate driving it, then the objective function in the placement problem (or a component of it) measures the total active area required to meet the target constraints. Since placement depends strongly on the areas of the cells and their physical dimensions, any attempt to use the size independent network for placement must ensure that the area estimation is carried out with sufficient accuracy.

44. **Continuous Gate Sizing.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of employing continuous sizing of a gate to maintain a constant delay for that gate during logic synthesis and physical design.

➢ Synopsys Patent Application #1, Claim 3, Page 8 "The method of claim 2, where step c) is comprised of steps: d) Determining the parameter C/S for each gate according to delay and gain constraints on the network. e) Determining the size S for each gate by traversing the network, in the direction opposite to the direction of data flow, while calculating the load of each gate, the size S of said gate being determined by the parameter C/S and the load C of said gate, the load on the preceding gate being determined by the size of the fanout gates, iterating step e) until convergence." ➢ Synopsys Patent Application #2, Claim 15, Page 22 "(sizing algorithm) … starting at the primary outputs and traversing the network in the direction opposite to the data flow, while calculating the load of a cell, by summing over the fanout of the cell, the product of the load of the fanout cell divided by the gain plus the net load of the cell." ➢ Synopsys Patent Application #2, Claim 20, Page 23 "A method for the placement and sizing of cells of a mapped digital network … d) Choosing a target delay for each cell; e) Computing the network slack using the target delays; f) Placement of the cells of the network; g) Sizing of the cells of the network such that the network meets the network slack as computed by step b)." ➢ Synopsys Patent Application #2, Claims 24-25, Page 23-24 Claim 24: "(sizing) … g1) calculation of the net length based on the available placement information; g2) calculation of the capacitive load of the cells using the net length; g3) calculation of the sizes from the capacitive load." Claim 25: "(sizing algorithm) … starting at the primary outputs and traversing the network in the direction opposite to the	➢ OttenSlides ICCAD '96, slide 52 "how to generate continuously sizeable libraries?" ➢ Sutherland et al. ➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), pages 4, 5 Page 4, right column: Also: "Stretching is a method for area optimization by changing the delays of the gates. Note that here delay is an independent variable and size, load and area depend on the delay. Stretching the delay of a gate increases its gain and decreases its size, thus saving area. Stretching is like a continuous sizing algorithm, except that the stretching algorithm changes delays as design decisions and uses area as an objective function. Here the delays of the gates are increased to save area." Page 5, left column: "The core assumption of the size independent delay model approach is that each gate can be sized linearly to a given load. If the sizes of the transistors can continuously be adjusted, this is close to accurate."

data flow, while calculating the load a cell, by summing over the fanout of the cell, the product of the load of the fanout cell divided by the gain plus the net load of the cell."	

45. **Discrete Gate Sizing.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of employing discrete gate sizes with the objective of maintaining a constant delay for that gate during logic synthesis and physical design.

➢ DrivingLHS, page 5, right column, Section 7 ("Discrete sizing algorithms"): "By using discrete gates in parallel, integer multiples of the gates can be created. … a duplicated gate can drive exactly twice the load at exactly the same delay. Different sizes can be mixed to create non-integer multiples."	➢ OttenSlides ICCAD '96, slide 52 "how to handle discrete libraries?" ➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), pages 1, 5 Page 1, left column, abstract: "This paper applies to the concept of 'logical effort' to logic synthesis. The concept of logical effort is used to motivate the size independent delay model for logic synthesis. Methods to determine … initial delays for library cells are discussed. Area analysis and delay optimization methods based on 'electrical effort' or gain are discussed." "… continuous gate sizing as the best way to use the methodology, it shows that discrete libraries can also be supported. The method uses delay/gain as an independent variable as opposed to gate size which is traditionally used." Page 5, Section 9 (Discrete sizing algorithms): "An algorithm is needed to round the computed continuous sizes to the discrete sizes of the library. Discretization converts a network all of whose gates have only delays assigned to them (say by a delay optimization program), into a network with all gates consisting of sized gates from a given library."

46. **Area Minimization.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of formulating an equation that calculates the area of a circuit and using that equation to minimize the area while maintaining constant delay.

➢ Synopsys Patent Application #1, Claim 5, Pages 8-9 "(Area estimation)" Claim 5, pages 8-9: "The method of claim 2, where the logic synthesis of step a) uses network area as an optimization goal, in addition to network delay, the network area being estimated by setting the parameter C/S for each gate according to the method of claim 2, followed by the method of step e) being used to estimate the sizes of the gates, and hence the area of the network." ➢ Synopsys Patent Application #2, Claim 1, Page 20 Constant delay synthesis consisting of "a method for the structuring and mapping of an unmapped digital network." "… using network slack as an optimization goal, where network slack is calculated assuming that the delay of the cells of the network is constant with respect to load." "c) Estimation of the area of the network based on net load." ➢ Synopsys Patent Application #2, Claim 14, Page 22 "(area estimation) … c3) calculation of the sizes from the capacitive load. c4) calculation of the network area by summation of the product of the book area times the cell size." ➢ DrivingLHS, page 4, right column, section 5 "Let q_i be the unscaled load, which consists of the wire load and any primary output load. The capacitance c_i at the output of a gate i is calculated as the unscaled load, plus the sum of the capacitance on the	➢ OttenSlides ICCAD '96, slide 49 formula for area minimization: 49: c_i: capacitance at output i: $$c_i = q_i + \sum_{j \in \text{fanout}(i)} \frac{c_j}{h_{ij}}$$ Area = $\mathbf{a}^T\mathbf{c}$, where $\mathbf{c} = (\mathbf{I} - \mathbf{F})^{-1}\mathbf{q}$ ($\mathbf{q}$ is imposed capacitance) ➢ OttenISPD'98 (Global Wires Harmful?), page 4, right column Presents the usual formula for area estimation: "If we collect the imposed capacitances in a vector $\mathbf{q}$ and the capacitances at the output in a vector $\mathbf{c}$ we obtain the following relation: $$\left(I - Nf^D\right)c = q$$ The matrix N has the zero/non-zero pattern of the (directed) gate-to-gate incidence matrix and can contain the individual relations between the input and output capacitance. The area of the network is then equal $$\left(a \bullet f\right)^T c = \left(a \bullet f\right)^T \left(I - Nf^D\right)^{-1} q$$ Based on this relation it is not difficult to find out whether inserting buffers does save area. Of course, buffers can only be inserted if they do not cause violations in the overall time constraints. For buffers do introduce additional delay locally which is acceptable only if some slack is available there." ➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), pages 4, 5 Page 4, (Section 6, "Area Estimation"): The key concepts, including their development, was also presented in Otten's slides. "Alternatively by assuming the registers are not end points and can be scaled (which is

fanout gates, divided by the gain h_{ij} of input i of fanout gate j. $$c_i = q_i + \sum_{j \in fanout(i)} \frac{c_j}{h_{ij}}$$ Using matrix notation, **c** is the vector of gate capacitances, **q** is the vector of unscaled gate loads, and **R** is the matrix of the reciprocal gain. The diagonal of **R** is 0 if we assume that no cell has an output connected to an input of itself. $$R = \begin{bmatrix} 0 & \frac{1}{h_{12}} & \frac{1}{h_{13}} & \dots \\ \frac{1}{h_{21}} & 0 & \frac{1}{h_{23}} & \dots \\ \frac{1}{h_{31}} & \frac{1}{h_{32}} & 0 & \dots \\ \dots & \dots & \dots & 0 \end{bmatrix}$$ The load equation can now be written as **c** = **q** + **Rc**. The capacitance can be solved as the fixed point of this equation: **c** = (**I**-**R**)$^{-1}$**q**. A positive real solution exists if the largest eigenvalue of **R** is less than 1: $\lambda_1 < 1$. To calculate the area of the network, it is assumed that the area of a gate depends linearly on the load of that gate. The *area sensitivity* of a gate with respect to its output load a_i (vector **a**) is the marginal increase in area as a result of an increase in load. Since the area is assumed to depend linearly on the load, this is the area of gate i driving 1 unit of load. The area of the network is now $A = \mathbf{a}^T\mathbf{c}$."	typically not the case), one can use the following theoretical formulation." $$R = \begin{bmatrix} 0 & \frac{1}{h_{12}} & \frac{1}{h_{13}} & \dots \\ \frac{1}{h_{21}} & 0 & \frac{1}{h_{23}} & \dots \\ \frac{1}{h_{31}} & \frac{1}{h_{32}} & 0 & \dots \\ \dots & \dots & \dots & 0 \end{bmatrix}$$ "Where q_i be the unscaled load, which consists of the $$c_i = q_i + \sum_{j \in fanout(i)} \frac{c_j}{h_{ij}}$$ wire load and any primary output load. The capacitance c_i at the output of a gate i is calculated as the unscaled load, plus the sum of the capacitance on the fanout gates, divided by the gain h_{ij} of input i of fanout gate j. Using matrix notation, **c** is the vector of gate capacitances, **q** is the vector of unscaled gate loads, and **R** is the matrix of the reciprocal gain. The diagonal of **R** is 0 if we assume that no cell has an output connected to an input of itself. The load equation can now be written as **c** = **q** + **Rc**. The capacitance can be solved as the fixed point of this equation: **c** = (**I**-**R**)$^{-1}$**q**. A positive real solution exists if the largest eigenvalue of **R** is less than 1: $\lambda_1 < 1$. The *area sensitivity* of a gate with respect to its output load a_i (vector **a**) is the marginal increase in area as a result of an increase in load. Since the area is assumed to depend linearly on the load, this is the area of gate i driving 1 unit of load. The area of the network is now $A = \mathbf{a}^T\mathbf{c}$."

47. **Area Estimation.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of computing a weight for each net and using those net weights to estimate circuit area in the constant delay paradigm.

➢ Synopsys Patent Application #2, Claim 8, Page 21 " … area savings are estimated using net weights which reflect the change of network area due to sizing as a function of net length." ➢ Synopsys Patent Application #2, Claim 9, Page 21 "… calculation of net weights is performed by starting at the primary inputs and traversing the network in the direction of the data flow, while calculating the net weight of a cell by summing over the fanin of the cell the product of the net weight of the fanin cell divided by the gain plus the cells area/load sensitivity, as W_i = sum_j W_j / g_ij" ➢ Synopsys Patent Application #2, Claim 31, Pages 25-26 ("Calculation of net weights") … step f1) is performed by starting at the primary inputs and traversing the network in the direction of the data flow, while calculating the net weight of a cell by summing over the fanin of the cell the product of the net weight of the fanin cell divided by the gain plus the cells area/load sensitivity, as W_i = sum_j W_j / g_ij"	➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), page 5 "The quality of the placement solution relies on the relative net weighting metric used" "Size independent synthesis provides a natural form of net weights. If we choose to associate with each net, the area sensitivity of the gate driving it, then the objective function in the placement problem (or a component of it) measures the total active area required to meet the target constraints. Since placement depends strongly on the areas of the cells and their physical dimensions, any attempt to use the size independent network for placement must ensure that the area estimation is carried out with sufficient accuracy. Placement algorithms can then be modified to including sizing. For example one may consider a bipartitioning placement program that begins sizing the gates after a certain number of cuts based on Steiner estimates for wire length."

48. **Stretching Constant Delays.** The following chart lists the disclosures in the Synopsys Materials (left column) and in the Public Materials (right column) of the concept of adjusting (or stretching, compressing, trimming, etc.) the constant delay for gates during the logic synthesis and physical design.

➢ Synopsys Patent Application #2, Claims 10-12, Page 21 Claim 10: "(stretching) … a1) choosing a delay for each, book, … a3) adjusting the delay of each cell based on the slack." Claim 11: "… the delay of each cell is adjusted equally among the stages which have the same slack." Claim 12: "… the delay is adjusted on each	➢ Lukas, Kudva, Shenoy '98 (Size Indep. Synthesis), pages 1, 2, 3, 4 Page 1, right column: What is alleged to be new over Sutherland and Grodstein is 'stretching' and 'compressing': "There is a significant difference between the constant delay model as proposed in [1, Grodstein] and the size independent model which is

path, such that the slack of each path becomes 0." ➢ DrivingLHS, page 5, left and right column "Stretching is a method for area optimization by changing the delays of the gates. Note that here delay is an independent variable and size, load and area depend on the delay." "Stretching the delay of a gate increases its gain and decreases its size, thus saving area." "Section 2 [due to Sutherland, et al.] suggests an interesting and simple approach to stretching. According to the simplified path sizing problem in that section, the effort delay of each stage should be equal. Assuming that the effort delay of each stage already was equal, this means that the delay of each stage needs to be increased by the same amount, regardless of function, load or size."	derived from [3, Sutherland, et al.] and applied to synthesis in this paper. In the constant delay model, the delay is determined during initial library analysis and kept constant throughout synthesis. In the size independent delay model on the other hand, the library is analyzed to determine the size independent delay model for the library as well as for determining initial conditions, but the delay of a gate may be changed during the course of synthesis as determined by appropriate delay optimization techniques. Page 2, left column: "In the new formulation, the delay becomes a parameter of the design, which is adjusted by an optimization algorithm, and its effect on gate size is observed using area analysis. Since delay is determined independent of size, this approach will be called the *size independent delay model*. Note that the delay of a gate does not need to remain constant throughout synthesis, but a change in delay is a 'conscious' design decision." Page 3, right column (Section 4, "Size independent synthesis"): "Evaluating a synthesis change in the network becomes easier: since the delays are size independent in the new model, the amount of time to recalculate slacks can be reduced." Page 4, right column: Also: "Stretching is a method for area optimization by changing the delays of the gates. Note that here delay is an independent variable and size, load and area depend on the delay. Stretching the delay of a gate increases its gain and decreases its size, thus saving area. Stretching is like a continuous sizing algorithm, except that the stretching algorithm changes delays as design decisions and uses area as an objective function. Here the delays of the gates are increased to save area."

Summary and Conclusions

49. As illustrated in the first set of charts above (paragraphs 26-36), each of the 11 concepts in the Synopsys Materials was disclosed at least once in the Magma Materials, and many concepts were disclosed multiple times in the Magma Materials.

50. As illustrated in the second set of charts above (paragraphs 38-48), each of the 11 concepts in the Synopsys Materials was disclosed at least once in the Public Materials, and many concepts were disclosed multiple times in the Public Materials.

51. The paper "Size Independent Synthesis" by van Ginneken, Kudva and Shenoy includes all of the concepts in the paper "Driving on the Left Hand Side of the Performance Speedway" by van Ginneken. "Size Independent Synthesis" by van Ginneken, Kudva and Shenoy has only one section that contains any significant new content over the previous van Ginneken paper, and that is Section 8 (page 5, left column) entitled "Placement and size independence".

52. Constant delay was not a new concept when Grodstein wrote his paper (for example, see the comment on page 1, right column). Applying constant delay to the technology dependent case was introduced by Grodstein, et al. Sutherland, et al. in combination with Grodstein, et al. provides the theory of logical effort plus the notion of constant delay, applied to the technology dependent case. This combination teaches at least part of all of the concepts disclosed in the Synopsys Materials.

53. The paper "Driving on the Left Hand Side of the Performance Speedway" acknowledges that: "The constant delay model is not really different than conventional models. The difference merely is which variable is held constant, when the load on a gate changes." Compare this to what Otten says in his ICCAD'96 tutorial slides: slide 52: "don't be fooled: the problem did not change but the viewpoint did!"

54. The paper "Size Independent Synthesis" by van Ginneken, Kudva, and Shenoy concedes that logical effort and the notion of gain was introduced by Sutherland et al. It further concedes that Grodstein et al. introduced constant delay, and the notion of constant delay synthesis. The paper claims to introduce the concept of stretching and compressing the delays. However, Otten's ICCAD tutorial had already revealed the concept of stretching.

55. Magma was very, very open with what they were working on, from presentations at conferences, trade shows, discussions with professors (such as myself). It would be inconceivable to me that Synopsys would not have known they were using logical effort plus constant delay synthesis.

56. Submitting a paper to a conference for publication is universally understood to be contrary to protecting trade secrets. A panel of experts reviews the papers without having to formally sign any type of nondisclosure agreement, nor is there any type of requirement to destroy the papers after reviewing them. There is simply no mechanism in place to protect trade secrets once a paper is submitted for possible presentation at a conference. This is universally known to those who submit papers.

I declare under penalty of perjury under the laws of the United States that the foregoing is true and correct. Executed this 10th day of June, 2005 in San Francisco, CA.

Carl Sechen