TIMING DRIVEN PLACEMENT USING COMPLETE PATH DELAYS

Wilm E. Donath, Reini J. Norman, Bhuwan K. Agrawal, Stephen E. Bello,
Sang Yong Han, Jerome M. Kurtzberg, Paul Lowy, and Roger I. McMillan

IBM Corporation

Thomas J. Watson Research Center, Yorktown Heights, New York 10598
Data Systems Division, Kingston, New York 12401

The Timing Drive Placement (TDP) system balances wirability and timing constraints so that the final released design meets timing criteria. This is achieved by dynamically evaluating the timing of critical paths during placement. TDP is significant because convergence to a timed wirable solution early in the physical design cycle is achieved, or else it becomes apparent that logic changes are required.

INTRODUCTION

In the past, physical layout programs were designed primarily for obtaining a wirable layout on a chip, without regard to performance considerations. This was satisfactory as long as the delay attributable to wire length was small compared with the delay of the circuit itself. However, the problem of meeting timing specifications is especially severe for today's dense CMOS FET technologies where typically 50% of a circuit's delay is attributable to inter-circuit wiring capacitance. Consequently, a placement which is wirable often contains paths with excessive delays.

Recently, several approaches have been developed which optimize not only for wirability, but also for timing. Reference Hau87 contains an excellent discussion of the problem and a more detailed description of alternate approaches than we present here.

An early approach (Wo178) consisted of first completing placement, then performing timing analysis. If timing verification indicated that several paths did not meet their timing requirements, individual circuits were allowed to change their location. More recent approaches have in common that they limit net capacitance. In one approach, the placement program is instructed to preferentially shorten nets contained in critical paths as determined by timing analysis (Tei86). Reference Hau87 describes a placement method that places capacitance limits on all nets as determined by timing analysis.

However, timing is not determined by the behavior of individual nets, but rather by combinations of nets which we call paths. In general, for the method of limiting individual net capacitances to work properly, an averaging effect must be postulated — some nets on a path may exceed their allowance, which must be compensated for by other nets that do not exhaust their allowance. Capacitance limits do not address this problem directly, because the placement algorithm optimizes nets without regard to paths.

Most recently a hierarchical approach to correct this problem has been described (Mar89); this aproach uses capacitive weights, effectively based on timing analysis, which are recomputed at each level of the hierarchy.

We now introduce a new approach whereby the placement process is divided into a global step and a detailed step. During the global placement process the timing of the chip is evaluated dynamically together with the wirability. The result is a global placement, which controls the detailed placement process that follows. The major advantage of TDP is that timing and wirability are accurately predicted early in the design effort, without expending time and energy in the detailed physical design portion of the design cycle.

PROCESS

This paper describes a methodology for standard cell or gate array designs (Ald87). For the purposes of design only, the chip image is divided into equal-sized rectangles called precincts. The size of the rectangles is chosen to be small enough so that one can still obtain resistance and capacitance estimates with sufficient accuracy to predict the timing performance of the chip. For example, for 9.4 MM chips we use 120 precincts.

The logic is partitioned into segments, each of which fits into one of the precincts. The segments are then placed into the precincts to form an initial global placement.

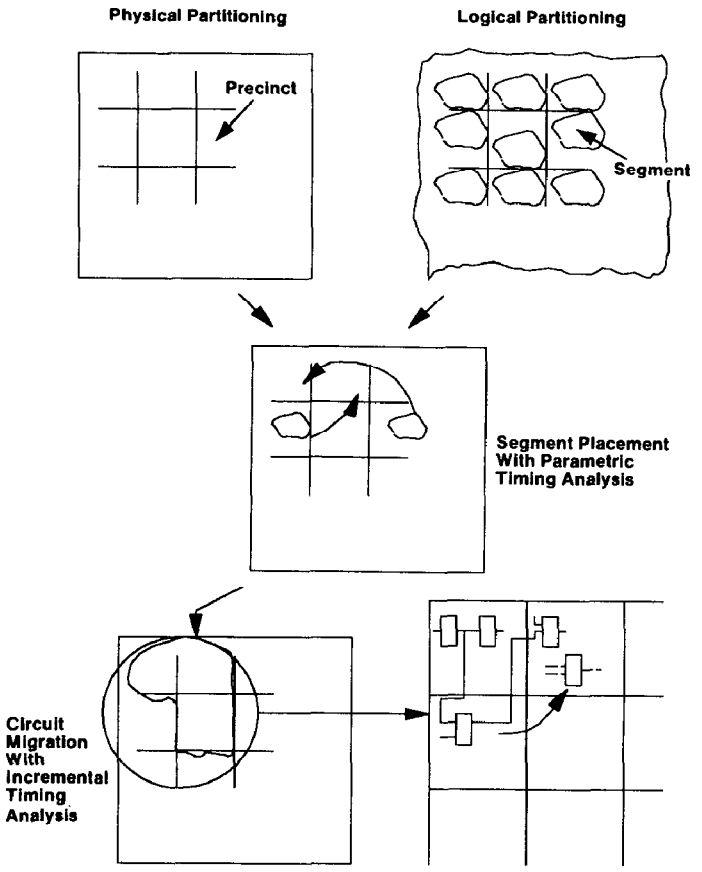
In this initial global placement phase (also called segment placement) we include in the placement evaluation function — in addition to wiring metrics — a metric that is essentially a set of equations for timing critical paths as a function of placement. After each move, the equations are re-evaluated; this dynamic evaluation is called "parametric timing analysis". Since the circuits within a segment will remain together during the segment placement phase, circuit delays within the segment may be computed based on a small capacitance adder that approximates the wire length of the final placement with reasonable accuracy. Similarly, inter-segment nets can be assigned a larger capacitance adder because it is known that on average these "global" nets are larger than the intra-segment or "local" nets. Thus early in the design cycle a more accurate estimate of chip timing than that based on fanout alone is obtained.

After segment placement, both wirability and timing are improved by moving or "migrating" individual circuits from precinct to precinct. After each trial move affecting a critically timed path, a timing analysis ("incremental timing analysis") is performed on that path to determine acceptability. A complete timing analysis is performed after each major step. Figure 1 is a high level diagram of the process.

The result is a global placement made of precinct areas and an assignment of each circuit on the chip to a precinct. The detailed placement phase honors the global placement assignment, and optimizes the placement of circuits within each precinct area.

We will now discuss in more detail the significant parts of the process; namely timing analysis, partitioning, and placement.

Figure 1. TDP Process



TIMING ANALYSIS

The principals of timing analysis have been well documented in previous papers (Hit82, Hau87). TDP's timing analysis function satisfies the following requirements:

1. Compute delays reasonably fast. In our case, the delay equation for each circuit is a 5 term equation with variables for transition times, and load capacitance; an additional term is used for RC delay.
2. Accommodate different capacitance calculations for wire length:
   a. Without placement or partitioning information — i.e. based only on an estimate of capacitance per connection.
   b. With partitioning information, where the wire capacitance is computed as K1 x fanout for global connections and K2 x fanout for local connections. This term can be estimated from the wire length estimation method developed by one of us (Don79, Hel84). For a wiring capacitance of .2 ff/um, K2 is .133 pf/fanout. Roughly, K1 is five times the size of K2.
   c. With global placement information, where the wiring capacitance can be estimated fairly accurately for the global connections using a minimal spanning tree (Pre88) to connect precinct centers. This is somewhat conservative on average, but is balanced by various effects that occur during detailed placement and routing (i.e. some nets — because of wirability constraints — are elongated during either placement of routing).
3. Include an estimate of the resistance in the net for computation of the RC term for the delay calculation steps after partitioning. After partitioning but before the segment placement step, the resistance is estimated as a constant times the estimated capacitance. After segment placement, wire resistance is calculated using the center of the precinct for each circuit I/O location and the minimal spanning tree algorithm as is done for capacitance.
4. Work incrementally (in addition to batch operation) — i.e. when a circuit changes its assigned precinct —the timing analysis should re-evaluate only those transitions, arrival times, and expected arrival times, which are affected by the change.
5. Compute the slack for each path.

PARAMETRIC TIMING ANALYSIS

Path analysis for global placement was developed based on the two level hierarchy. We do not have to be concerned about nets contained within segments, since their delays will stay constant during segment placement. Since the delays of global nets are not known before global placement, a set of slack equations in terms of the delays on global nets must be prepared. This is accomplished in two steps by:

1. Deriving a simple delay equation for each global net, which is a two-piece linear equation. This takes into account the possibility of inserting a buffer circuit when the capacitance becomes large. We subtract the nominal wiring delay based on partitioning, so that this equation represents a correction for the delay of the net. The equation is:

   C<Ccrit: DC = a*(C — Cnom)

   C>Ccrit: DC = b*(C — Cnom)

   where C is the total capacitance of the net i.e. wire plus circuit capacitance, Ccrit is the capacitance above which the buffer is inserted, DC is the delay correction for the delay from the partitioning estimate, a and b are the slopes of the delay curve for the buffered and unbuffered circuits, and Cnom is the nominal capacitance based on the partitioning step.

   We now denote by "term" the parameterized delay equation above together with the set of segments that make up the end points of the global net.
2. Propagating the terms through the network to obtain the set of equations that the segment placement program uses.

In principle, the second step could yield an exponential number of slack equations because of the multiplicity of paths on a chip. Several measures are required to keep the number of such equations within a manageable limit:

1. The number of different types of different global net delay equations is limited as much as possible.
2. Paths which are identical are recognized and only the worst case arrival times are retained.
3. A significant amount of pruning is done in the propagation step.

For the first step above, all global nets which have the same global configuration (i.e. drivers and sinks are distributed into the same set of segments) are treated simultaneously. The global net with the worst slack is fitted first — and the program attempts to fit the other nets with the same equation. Should the deviation for a net with significant slack become too large, another form of the slack equation is generated for this set of nets.

It is general practice in timing analysis to generate a timing graph from the logic description (Hen73); the nodes of the graph hold arrival times and may refer to input or output pins of the logic circuit, while the directed edges may refer to circuit

delays or wire delays. If the delay of a directed edge includes a term as defined above it is a global edge. If not, it is a local edge.

Only a forward propagation step is required; instead of arrival times, each node of the timing graph contains a set of equations, each of which is the sum of a nominal arrival time plus a set of terms. Identical sets of parameterized delays are recognized with a hashing scheme. Propagation contains three phases:

1. Propagate across a local edge, where only arrival times of the different equations have been changed.
2. Propagate across a global edge. As this occurs, each equation is augmented by an additional term.
3. Merge several different edges, where the equations arrive in an ordered form and — as stated before — only the worst case path is taken for identical cases. Pruning is also done in this phase. The most effective form of pruning is to simply limit the number of paths handled at each node, and keep those with worst case nominal arrival times.

At the capture points (data input to a latch or primary output) the expected arrival times are subtracted from the nominal arrival times. Further subsumption of identical equations is done in a final merge. This final set of equations with their associated slacks is then used by the initial global placement phase (by the segment placement program).

These steps enable the sum of all the slack equations for a typical 9.4 mm chip to be restricted to about 40,000 equations.

INCREMENTAL TIMING ANALYSIS

When a circuit changes its assigned precinct or segment during a migration step, timing analysis is performed immediately to evaluate the move. The timing graph that represents the logic description (described in the previous paragraph) remains largely intact and is updated by the set of changed circuits. Edges (delays) are updated from the new placement or partition and only affected arrival or expected arrival times are recomputed, and the corresponding nodes are updated. Typically, a trial move of one circuit to a different precinct affects 5 to 10 circuits and the evaluation requires less than .1 seconds on a 3090 model 200 CPU.

PARTITIONING

The objective of the partitioning step is to divide the circuits into the required number of segments, minimizing the number of global nets, i.e. nets which span multiple segments. To accomplish this a tree structure of successive bi-partitions is used. At each leaf of each level of the tree the relative size and the minimum and maximum size in cells of each bi-partition are calculated. The algorithm used to minimize the cutset follows the work of Krishnamurthy (Kri84). We use the following features:

1. The starting bi-partition at each leaf is not random but uses the design process (the circuit sequence number) to aid in selection.
2. Nets are divided into two categories, global (those previously cut) and local. Gain vectors and cut sets are maintained for both categories. A higher weight is assigned to cutting a previously uncut net than is assigned to one that is already global.

Good partitioning helps the segment placement phase enormously by reducing the number and types of global nets. Therefore, improvement of this partitioning phase is undertaken through the use of two circuit migration programs:

1. The first program attempts to reduce global nets (if possible) or the average Rent exponent (Pre88) by moving circuits from its segment to any other segment that a net on the circuit goes to.
2. The second program employs incremental timing analysis. Nets are sorted on the basis of worst slack and attempts are made to improve the slack of these critical nets by reducing the number of segments these nets attach to.

After partitioning and improvement we have found that the number of global nets is about 40% of the total nets and the average Rent exponent ranges from .65 to .72 depending on the design.

SEGMENT PLACEMENT

Segment placement uses a simulated annealing algorithm (Kir83). The cost function is a composite of both wiring and timing cost functions. Standard techniques are employed — i.e. an initial set of 10 random placements is used to set the initial temperature. After this, a fixed number of moves (a move is the swap of a segment in one precinct with a segment in another precinct) is tried at each temperature, until that number is exhausted or a given number of moves has been accepted; the temperature then is multiplied by a fraction to determine the next temperature. The assignment of chip I/O's is also evaluated during annealing.

Wiring Cost Function

In determining the cost function for the placement of segments in precincts, both local wiring and global wiring must be evaluated. Local wires connect circuits within the precinct only. Global wires connect circuits that are in different precincts. The cost function is the ratio of the estimated wiring to the available tracks (wiring channels) on the chip.

1. Local wiring

   One can estimate demand for tracks from local wires from wirability theory (Don79, Hel84) as one knows the number of circuits and connections within each segment. When a segment is assigned to a precinct during placement, the precinct gets this value known as the local precinct demand.

2. Global wiring

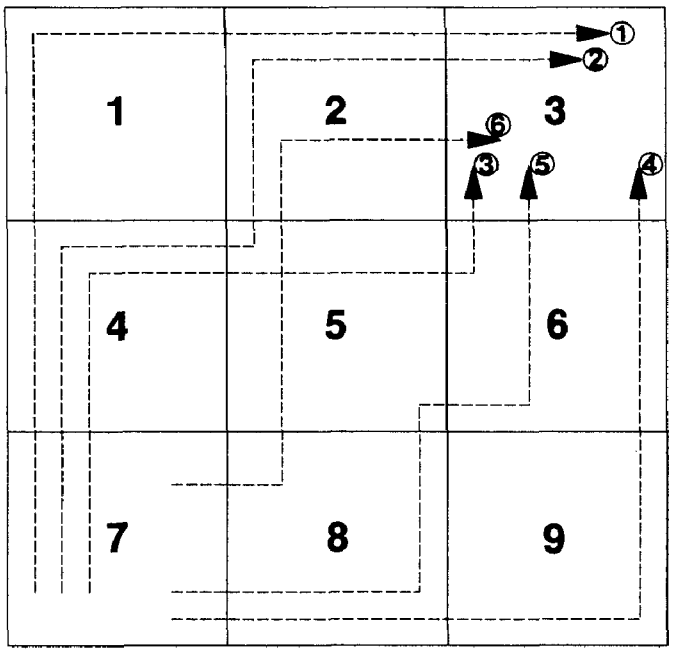
   This calculation is considerably more complicated than the local wiring calculation. We account for global wiring in two ways:

   a. First, account for wires that cross a precinct boundary to connect to a circuit I/O in that precinct; in this case we essentially count the number of global nets crossing these precinct boundaries. This is the cut demand.

   b. Second, account for wires that do not connect to a circuit I/O of the precinct in question, but nonetheless must cross through the precinct to connect circuits in other precincts to one another. A stochastic method is used to compute the demand these global wires have inside the precincts. This is the global precinct demand. Consider a net which must cross some partial combination of seven precincts to arrive at its destination. Please refer to figure 2. To go from precinct 7 to precinct 3, the net would take one of these routes: 4-1-2, 4-5-2, 4-5-6, 8-9-6, 8-5-6, or 8-5-2. The probability that the net will traverse one of these seven non-end point precincts is the number of paths through the precinct divided by the total number of paths (e.g. probability = .5 for precinct 5). This is calculated rapidly using binomial coefficients.

The global precinct demand is added to the local precinct demand to form the precinct demand.

Figure 2. Computing Precinct Demand



For both cut and precinct demand we also know the corresponding tracks available — which we call the capacity. We can have both a cut and a precinct cost function by taking the ratio of demand to capacity for each item. These ratios are raised to some power to reduce the spread between the smallest and largest value and summed to form the cut and precinct cost. Each cost is multiplied by some weighting term and added together to obtain the wiring cost.

The program to evaluate the wirability is able to work incrementally for each change in placement of a segment in segment placement, or a circuit in circuit migration.

Timing Cost Function

To describe how timing is evaluated during the annealing process, we introduce the following definitions:

1. A "Form" gives the constants a,b, and Ccrit used in the delay formulas defined previously in the paragraph on Parametric Timing Analysis
2. A "Term" consists of a Form followed by a segment list.
3. An "Equation" consists of a nominal slack followed by a list of terms.
4. "Baseslack" is a value of slack such that equations yielding slack higher than this value are not considered in placement.

Files containing terms and equations are used to define data structures which permit efficient evaluation of changes. As segments are interchanged, global nets which connect to these segments are evaluated for changed capacitance which results in changed terms and corresponding changes in delay. Changed terms result in changed equations and thereby changes in the slack of these equations.

Each equation contributes to the timing cost if its slack is worse than the base slack; the timing cost is the summation of these differences raised to some power. Not all equations need be considered. If the nominal slack of an equation is sufficiently positive, no segment placement is possible that will enable the slack to degrade to the baseslack and therefore the equation can be dropped. Thus an input parameter (base slack) allows one to reduce the total number of equations considered.

COMPUTER RESOURCES

TDP is written in PLI and operates on a VM system. Most chips require 8 MEG of storage. Chips with more than 35,000 equivalient circuits (an equivalent circuit is a 2 way NAND) require 16 MEG. The CPU time for the entire TDP process for a typical chip of 30,000 equivalent circuits, is 4 hours on an IBM 3090 model 200 CPU, which is a 28 MIP machine. This excludes detailed placement and wiring times. Table 1 shows times of the major steps for 2 designs. During the TDP process, analysis and changes to operating parameters take place and normally a single pass takes 1 - 2 days. It has been our experience that starting with consistent design data about 3 iterations through TDP are required, taking about 6 working days. However after successfully running TDP, the detailed design requires only 1 pass.

TABLE 1. CPU Time

	Chip 4	Chip 5
No. of Equiv. Ckts.	11,500	31,400
No. of Precincts	60	120
Partitioning*	12	39
Timing Analysis*	6	19
Equation Generation*	3	5
Placement*	15	56
Circuit Migration*	8	12

*Times are IBM 3090/200 minutes

RESULTS

It is of prime importance that TDP's wirability and timing estimates accurately predict the final design.

The precinct and cut cost funcitons correlate to the efficacy of the final detailed routing. We have found consistently that if the worst cut cost after the circuit migration for wiring step is less than .8, and if the worst precinct cost is less than 1.3, the chip will be wirable. Typically, when cut costs are below .7, and precinct costs are below 1.0, the detailed routing program routs all wires i.e. no wires remain to be added manually. When the worst cut costs are between .7 and .8, and worst precinct costs are between 1.0 and 1.3 the number of wires to be added manually is between 0 and 100 (Table 2).

Timing will be accurately predictable if:

1. the capacitance estimates from TDP match the final capacitance,
2. the precincts constrain placement effectively,
3. the capacitance limits for critical nets are respected by the router.

TABLE 2. Cut and Precinct Costs vs. Wirability

	Worst Costs				Manual
	Hor. Cut	Vert. Cut	Hor. Pct.	Vert. Pct.	Wires Added
Chip 4	.39	.45	.75	.73	0
Chip 5	.73	.75	1.05	1.11	29
Chip 6	.80	.79	1.34	1.29	104

This is the case. Table 3 illustrates that TDP's timing analysis results — based on the global placement — compare well with the results from a timing analysis run on the final placed and wired chip.

Experiments were run to compare a conventional non-performance oriented approach (non-TDP) with the TDP approach. Both approaches have in common that the IBM

TABLE 3. TDP Slacks vs. Final Slacks*

	TDP (% cycle)	Final (% cycle)	Difference (% cycle)
Path 1	4.6	4.6	0.0
Path 2	8.4	6.5	+1.9
Path 3	8.4	6.2	+2.2
Path 4	8.2	10.0	-1.8

*for 7.5 MM chip

Engineering Design System is used for detailed design including placement (simulated annealing) and wiring (global router and mazerunner). The major difference is that with TDP the global placement controls the detailed placement. Table 4 compares the wirability of TDP with the non-TDP approach based on the number of wires to be added manually in each case.

TABLE 4. Wirability — Number of Manual Wires Required

	Non-TDP	TDP
Chip 4	0	0
Chip 7	0	3
Chip 5	3	29

These additions are used to measure wirability rather than wire length since the chip meets performance constraints. The table indicates that TDP degrades wiring slightly over the conventional approach. Table 5 shows the improvement in slack as a percent of cycle time for 9 chips using TDP versus the non-TDP or conventional design approach. The results were tabulated after detailed wiring but before any manual corrections such as adding unrouted wires or buffer circuits were made. Although all chips except Chip 8 still had some negative slacks after TDP, the distribution of slacks was improved considerably with TDP. This is shown in figure 3 for a typical chip. The vertical axis of the histogram, labeled "no. of endpoints", represents paths on the chip (with negative slacks) that terminate in a capture point. The shift in slack distribution to the left and the improvement of the slowest nets are typical of TDP designs.

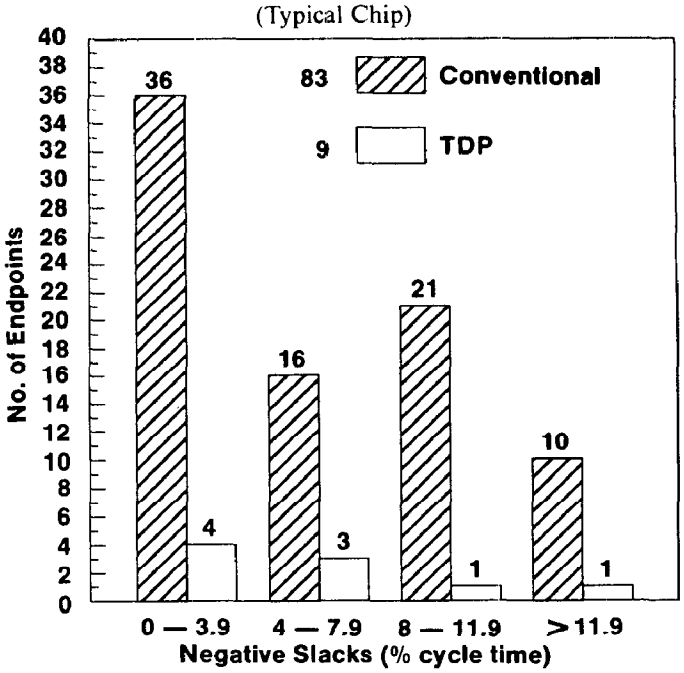
TABLE 5. Cycle Time Improvement

	Worst Slack (% cycle)		Improvement (% cycle)
	Non-TDP	TDP	
Chip 1	-17.3	- 6.4	10.9
Chip 2	-20.4	- 2.9	17.5
Chip 3	-12.9	- 6.6	6.3
Chip 4	- 9.3	- 1.8	7.5
Chip 5	-10.2	- 4.7	5.5
Chip 6	-21.4	- 4.3	17.1
Chip 7	-18.3	- 3.8	14.5
Chip 8	- 1.1	+ 4.4	5.5
Chip 9	-20.0	-13.0	7.0

The complete physical layout process for a 30,000 equivalent circuit chip which is guaranteed to meet all layout guidelines and performance constraints under all operating conditions is a complex process involving placement, wiring,

Figure 3. Slack Distribution

(Typical Chip)



checking and timing analysis. The longest step in this process is detailed wiring which requires about 12 CPU hours for a 9.4 mm chip. It is costly and time consuming to repeat this step. Thus far, sixteen chips have been processed through TDP. The average time to complete the process steps to the first timing analysis run on full detailed data was 12 days, 6 of which were spent completing the steps preceding detailed placement. On average, 2 "minor" timing correction cycles per chip were required. A "minor" correction involves analysis and manual wiring changes and takes about 5 days per cycle. Therefore 22 days on average were required to design the wirable, timed chip. On the other hand, without TDP, we speculate that it would have been necessary to do 2 "major" corrections involving analysis, placement, and wiring for some of the chips. (This has been the case historically.) A major correction requires that placement and wiring be re-done, taking 15 days per cycle. Then, 36 days would be the time to obtain a wirable, timed chip.

CONCLUSION

TDP is effective in achieving the goal of automatically placing and wiring CMOS chips and guaranteeing that they meet timing specifications. Without TDP, chips may have to be depopulated to meet timing requirements. The extra time (6 days average) spent in the TDP phase was small compared to the time saved by not having to replace and rewire the chip (and in the worst case redesign it) to fix timing problems. The designer is able to determine whether a solution exists early in the design cycle, and can either trade off timing with manual effort to wire the chip, or he or she can optimize logic to solve timing problems.

ACKNOWLEDGEMENTS

A large system like TDP requires the effort of many people, working together as a team and we gratefully acknowledge them: Kwok Eng, Frank Kordzikowski, David Sankus, Ralph Wilk, and the members of the Advanced Design Department at IBM Kingston. Throughout the course of the project, we received additional support from Paul Case, Ed Eichelberger, Bob Hatch, Peter Hauge, Bill Heller, Joe Hutt, Cyril Price, Bob Proctor and Dick Spadavecchia.

REFERENCES

(Ald87) A.W. Aldrich, R.F. Keil, J.H. Panner, G.D. Pittman and D.R. Thomas, "A 40K Equivalent Gate CMOS Standard Cell Chip", Proc. of the IEEE 1987 Custom Integrated Circuits Conference, 1987, pp. 248-252.

(Don79) W.E. Donath, "Placement and Average Interconnection Lengths of Computer Logic," IEEE Trans. on Circuits and Systems CAS-26, 1979, pp. 272-276.

(Hau87) P. Hauge, R. Nair and E. Yoffa, "Circuit Placement for Predictable Performance," Proc. of ICCAD '87 - IEEE International Conference on Computer Aided Design, 1987, pp. 88-91.

(Hel84) W.R. Heller, C.G. Hsi and W.F. Mikhail, "Wirability — Designing Wiring Space for Chips and Chip Packages," IEEE Design and Test, Aug. 1984, pp. 43-51.

(Hen73) E.J. Henley and R.A. Williams, "Graph Theory in Modern Engineering," Academic Press, Inc., 111 5th Ave. New York, New York 10003, 1973.

(Hit82) R.B. Hitchcock Sr., G.L. Smith and D.D. Cheng, "Timing Analysis of Computer Hardware," IBM Journal of Research and Development, vol. 26, no. 1, Jan. 1983, pp. 100-105.

(Kir83) S. Kirkpatrick, C.D. Gelatt Jr. and M.P. Vecchi, "Optimization by Simulated Annealing," Science, vol. 220, no. 4598, 1983, pp. 671-680.

(Kri84) B. Krishnamurthy, "An Improved Min-Cut Algorithm for Partitioning VLSI Networks," IEEE Transactions on Computers, vol c-33, no. 5, 1984, pp. 438-446.

(Mar89) M. Marek-Sadowska and S.P. Lin, "Timing Driven Placement", Proc. of ICCAD '89, IEEE International Conference on Computer Aided Design, 1989, pp. 94-97.

(Pre88) B. Preas and M. Lorenzetti, "Physical Design Automation of VLSI Systems", The Benjamin/Cummings Publishing Company, Inc. 2727 Sand Hill Road, Menlo Park, Ca. 94025, 1988.

(Tei86) S. Teig, R.L. Smith, and J. Seaton, "Timing Driven Layout of Cell-Based IC's," VLSI Systems Design, May 1986, pp. 63-73.

(Wol78) P.K. Wolff, A.E. Ruehli, B.J. Agule, J.D. Lesser and G. Goertzel, "Power/Timing: Optimization and Layout Techniques for LSI Chips," Journal of Design Automation and Fault Tolerant and Computing, 1978, pp. 145-164.