Efficient Minarea Retiming of Large Level-Clocked Circuits*

Naresh Maheshwari
Department of Electrical & Computer Engineering
Iowa State University, Ames IA 50011
naresh@iastate.edu

Sachin S. Sapatnekar
Department of Electrical & Computer Engineering
University of Minnesota, Minneapolis, MN 55455
sachin@ece.umn.edu

Abstract

Delay-constrained area optimization is an important step in synthesis of VLSI circuits. Minimum area (minarea) retiming is a powerful technique to solve this problem. The minarea retiming problem has been formulated as a linear program; in this work we present techniques for reducing the size of this linear program and efficient techniques for generating it. This results in an efficient minarea retiming method for large level-clocked circuits (with tens of thousands of gates).

1 Introduction

Timing optimization plays a vital role in the synthesis of VLSI circuits. One method that is of great interest to the design and CAD community is the procedure of retiming [1], which relocates the memory elements in a circuit without changing its functionality, to optimize some cost function. The problem of finding the minimum clock period is called *minperiod retiming*. Since minperiod retiming pays no regard to the area overhead it can significantly increase the number of memory elements in the circuit. Hence a method for minimizing the number of memory elements while satisfying a target clock period is needed. This is called *minarea retiming*.

The memory elements in a circuit may be either edge-triggered *flip-flops* (FF's) or level-sensitive *latches*. In a level-clocked circuit the latch is transparent during the active period of the clock, and the delay through a combinational logic path can be longer than one clock cycle, as long as it is compensated by shorter paths delays in the subsequent cycles. This transparent nature of the latch provides more flexibility both in terms of the minimum clock period achievable and the minimum number of memory elements needed. Unfortunately this transparency also complicates the analysis of level-clocked circuitry because data can ripple through several stages of memory elements before its propagation is complete. This makes the design and optimization of level-clocked circuits an intricate task, and the need of good automation tools acute.

As a result several efforts have been made to perform minperiod retiming on level-clocked circuits, based on the Leiserson-Saxe approach [1], e.g. [2, 3, 4]. Of these only TIM [3] provides a method of minarea retiming. The problem is formulated as a linear program (LP) similar to the one in [1]. Due to the transparent nature of the latches the number of constraints in this LP is extremely large, almost $\frac{|G|^2}{2}$ for a circuit with $|G|$ gates. This places heavy time and space requirements on minarea retiming and TIM is not capable of retiming large circuits for minimum area. The objective of this work, called Minaret-L, is to be able to retime level-clocked circuits with tens of thousands of gates in reasonable time.

An efficient method for minarea retiming of edge-triggered circuits was presented in [5]. This was achieved by utilizing the observation that in edge-triggered circuits, if a sub-path satisfies the timing constraints, then any path containing this sub-path will also satisfy the timing constraints. Unfortunately this is not true in level-clocked circuits, because of the the transparent nature of latches. Therefore the techniques of [5] cannot be applied to level-clocked circuits.

The ASTRA algorithm in [6], presented a different approach to retiming edge-triggered circuits utilizing the retiming-skew relation. This relation was extended to level-clocked circuits in [7], which presented an algorithm for minperiod retiming of level-clocked circuits under a general multi-phase clocking scheme. Both of these methods are capable of minperiod retiming of large circuits but do not address the harder problem of minarea retiming.

The work in [8] presented efficient techniques to obtain bounds on the variables of the minarea LP for edge-triggered circuits. It then used these bounds to further reduce the size of the LP and the time required

*This work was supported in part by National Science Foundation award MIP-9502556, a Lucent Technologies DAC Graduate Scholarship and a Iowa State University Computation Center Grant.

for generating the LP. Utilizing the retiming-skew relation for level-clocked circuits from [7], bounds on the variables of the minarea LP are obtained in this work, along the lines of [8]. However due to the transparent nature of latches, unlike edge-triggered circuits, the techniques of [8] cannot be used to reduce the time required to generate the LP in level-clocked circuits. This presents a major hurdle in retiming large level-clocked circuits for minimum area, because in the absence of any efficiency-enhancing technique, the minarea LP cannot be generated in a reasonable time. This work also presents new techniques for reducing the time taken to generate the minarea LP in level-clocked circuits.

The rest of the paper is organized as follows: Section 2 presents the required background, followed by minarea retiming method of TIM [3] in Section 3. We then present our efficient minarea retiming method in Section 4. Section 5 presents experimental results, and Section 6 concludes the paper.

2 Background

2.1 The Clock Model

As in [3] a k-phase clock is a set of k periodic signals $\Pi = \langle \phi_1, \gamma_1, \phi_2, \gamma_2, \ldots \phi_k, \gamma_k \rangle$, where ϕ_i is the active duration of phase i and γ_i is the gap between the falling edge of phase i and the rising edge of phase $(i+1)$. We denote the duration of phase i by $\pi_i = \phi_i + \gamma_i$. We overload the symbol Π to also denote the clock period $\Pi = \sum_{i=1}^{k} \pi_i$. A clocking scheme is symmetric if all phases have the same duration and active intervals, i.e. if $\phi_i = \phi \ \forall i = 1, \cdots, k$ and $\gamma_i = \gamma \ \forall i = 1, \cdots, k$. Thus for a k phase symmetric clocking scheme $\Pi = k \cdot \pi$ and $\pi = \phi + \gamma$. In this work we consider only symmetric clocking schemes.

2.2 The retiming model

As in [1], a sequential circuit is represented by a directed graph, $G(V, E)$, where each vertex v corresponds to a gate, and a directed edge e_{uv} represents a connection from the output of gate u to the input of gate v, through zero or more latches. Each vertex has a fixed delay $d(v)$. Each edge has associated with it a weight $w(e_{uv})$ and a width $\beta(e_{ij})$. The weight is the number of latches between the output of gate u and the input of gate v. The width of an edge is the area cost of placing one latch on it.

A retiming is a labeling of the vertices $r : V \rightarrow Z$, where Z is the set of integers. The retiming label $r(v)$ for a vertex v represents the number of latches moved from its output towards its inputs.

The weight of an edge e_{uv} after retiming, denoted by $w_r(e_{uv})$ is given by $w_r(e_{uv}) = w(e_{uv}) + r(v) - r(u)$. One may define the weight $w(p)$ of any path $p : u \rightsquigarrow v$, originating at vertex u and terminating at vertex v, as the sum of the weights on the edges on p, and its delay $d(p)$ as the sum of the delays of the vertices on p. Similarly $w_r(p)$ is the sum of the weights on the edges on p after retiming, and is given by

$$w_r(p) = w(p) + r(v) - r(u) \tag{1}$$

2.3 Alternate view of retiming

In [7] the term *Global Departure Time (GDT)* is defined for each latch as the latest departure time of data signal from that latch, with reference to the arrival time at the primary inputs in a global time frame. GDT is a very approximate analogy of "skew" used for edge-triggered circuits in [6]. A relation between GDT and retiming, similar to the one between skew and retiming for edge-triggered circuits [6] is also presented in [7]. This relation is used to solve the minperiod retiming problem for level-clocked circuits, by mapping it to the clock skew optimization problem. The GDT's so obtained are then used to get the actual retiming. Moving a latch from the inputs of a gate to its outputs is equivalent to increasing the GDT of that latch by an amount equal to the delay of the gate. Likewise, a motion from outputs to the inputs is equivalent to reducing the GDT by the gate delay. A value of GDT between $-\phi$ and 0 is considered allowable, since it corresponds to zero skew. The GDT's are reduced by relocating latches across gates obtaining the retimed circuit.

3 Minarea Retiming

In this section we present the method used in TIM [3] for minarea retiming of level-clocked circuits. The (constrained) minarea retiming problem is formulated as an LP. The objective function represents the number of latches in the circuit and the constraints ensure that the retimed circuit satisfies the target clock period. We will first derive these constraints and then present the LP.

3.1 Conditions for proper clocking

For a level-clocked circuit to be properly clocked the delay between any two gates should be less than the time available [4], i.e.,

$$d(p) \leq (w_r(p) + 1) \cdot \pi + \phi \tag{2}$$

After substituting Equation 1, this constraint can be rewritten as

$$r(u) - r(v) \leq w(p) - \frac{d(p)}{\pi} + 1 + \frac{\phi}{\pi} \tag{3}$$

Clearly if there are multiple paths from u to v only the tightest constraint (minimum right hand side)

is irredundant. We denote the minimum value of $w(p) - \frac{d(p)}{\pi}$ over all paths from u to v by $\delta(u,v)$, i.e.,

$$\delta(u,v) = \min_{\forall p: u \rightsquigarrow v} \left(w(p) - \frac{d(p)}{\pi} \right) \quad (4)$$

Let us define $\Delta(u,v)$ as

$$\Delta(u,v) = \left\lfloor \delta(u,v) + \frac{\phi}{\pi} + 1 \right\rfloor \quad (5)$$

Since the retiming variables $r(u)$ and $r(v)$ are integers, we can rewrite Equation (3) as

$$r(u) - r(v) \leq \Delta(u,v) \quad (6)$$

3.2 The minarea LP

The minarea retiming problem for a target period Π can be formulated as the following LP [3]:

$$\min \sum_{v \in (V)} \left[\left(\sum_{\forall j \in FI(v)} \beta(e_{jv}) - \sum_{\forall j \in FO(v)} \beta(e_{vj}) \right) \cdot r(v) \right]$$

$$\begin{aligned} \text{subject to} \quad & r(u) - r(v) \leq w(e_{uv}) && \forall e_{uv} \in E \quad (7) \\ & r(u) - r(v) \leq \Delta(u,v) && \forall u, v \ \exists \ p : u \rightsquigarrow v \\ & -\infty \leq r(u) \leq \infty && \forall v \in V \end{aligned}$$

The significance of the objective function and the constraints is as follows.

- The objective function represents the number of latches added to the retimed circuit in relation to the original circuit, taking into account maximal latch sharing [1] at the output of a gate. $\beta(e_{ij})$ is the cost of a latch on edge e_{ij}.Hence the cost coefficient of gate v gives the number of latches added to the circuit if it is retimed by one unit.
- The first set of constraints ensures that the weight e_{uv} of each edge (i.e., the number of latches between the output of gate u and the input of gate v) after retiming is nonnegative. We will refer to this set as the *circuit constraint* set C_c.
- The second set of constraints ensures that after retiming, every path satisfies the proper timing constraint of Equation (2). This set, being dependent on the clock period, is referred to as the *period constraint* set C_p. Notice that for every gate we have a period constraint to every other gate reachable from itself, therefore the number of period constraints is almost $\frac{|G|^2}{2}$ for a circuit with $|G|$ gates.

As in [1] this LP is also the dual of a minimum cost network flow problem.

4 Efficient Minarea Retiming

The minarea retiming method presented in Section 3 is unable to handle large circuits. Firstly, the size of the LP in Equation (7) is very large, and secondly, the time required to generate the period constraints is very high. In this section, we will present techniques for reducing the size of the minarea LP, and for efficiently generating the period constraints. We use an efficient network simplex algorithm from [8] to solve this LP. This algorithm could solve a LP with 70,000 variables and 17 million constraints in under 9 minutes.

4.1 Reduced minarea LP

The work in [8] modified the methods in [6] to obtain bounds on the r variables for edge-triggered circuits. These bounds where then used to reduce the size of the minarea LP. In this work we modify the methods in [7] along the same lines to obtain bounds on the r variables for level-clocked circuits of the form $L_v \leq r(v) \leq U_v$. The variable set V of the LP in Equation (7) can now be reduced to $V' \subseteq V$ the reduced variable set, where $V' = \{v \in V | U_v \neq L_v\}$ The circuit constraint set (C_c) and the period circuit constraint set (C_p) are also reduced to obtain the reduced circuit constraint set $C'_c \subset C_c$, and the reduced period constraint set $C'_p \subseteq C_p$. This reduction in the constraint sets is obtained by dropping redundant constraints identified by the following rule from [8].

Rule 1 Any constraint (i,j) of the form $r(i) - r(j) \leq c_{ij}$ is redundant in the presence of the bounds and can be dropped if $U_i - L_j \leq c_{ij}$.

This gives us the following reduced LP which has the same optimal solution as the LP in Equation (7):

$$\min \sum_{v \in V'} \left[\left(\sum_{\forall j \in FI(v)} \beta(e_{jv}) - \sum_{\forall j \in FO(v)} \beta(e_{vj}) \right) \cdot r(v) \right] \quad (8)$$

$$\begin{aligned} \text{subject to} \quad & r(u) - r(v) \leq w(e_{uv}) && \forall (u,v) \in C'_c \\ & r(u) - r(v) \leq \Delta(u,v) && \forall (u,v) \in C'_p \\ & L_u \leq r(u) \leq U_u && \forall u \in V' \end{aligned}$$

4.2 Generating the constraints

A major portion of the computational effort in retiming a level-clocked circuit for minimum area is spent in generating the period constraints set. We now describe efficient techniques for generating this constraint set.

The generation of period constraints requires computation of the δ values (see Equation 4 for all-pairs of gates in the circuit. These δ values can

be obtained by re-weighting each edge e_{ij} with $w'(e_{ij}) = w(e_{ij}) - \frac{d(i)}{\pi}$ and computing all-pair shortest paths. We use Johnson's algorithm [9] which has $O(|V|)$ memory requirement, since $O(|V|^2)$ memory is not practical for large circuits. Johnson's algorithm first re-weights all edges to ensure nonnegative edge weights. The shortest paths between all pair of gates can then be computed by running Dijkstra's algorithm for each gate as source.

Let us consider a particular run of Dijkstra's algorithm with gate a as the source, and let b be a gate to which the shortest path $\delta(a,b)$ has been obtained. Let c be any other gate in the circuit, reachable from gate b.

$$\begin{aligned} \text{By definition,} \quad & r(a) - r(b) \leq & U_a - L_b \\ \text{If} \quad & U_a - L_b & \leq \delta(a,b), \\ \text{then} \quad & r(a) - r(b) \leq & \delta(a,b). \end{aligned} \tag{9}$$

From Equation (5) and Equation (6)

$$r(b) - r(c) \leq \delta(b,c) + \frac{\phi}{\pi} + 1,$$

which when combined with Equation (9) gives

$$r(a) - r(c) \ \leq \ \delta(a,b) + \delta(b,c) + \frac{\phi}{\pi} + 1 \tag{10}$$

If the shortest path from gate a to gate c does not go through gate b then $\delta(a,b) + \delta(b,c) \geq \delta(a,c)$ and we do not need to process the fanouts of gate b to obtain $\delta(a,c)$. On the other hand, if the shortest path from gate a to gate c is indeed through gate b then $\delta(a,b) + \delta(b,c) = \delta(a,c)$ and Equation (10) is same as the period constraint $r(a) - r(c) \leq \Delta(a,c)$. If $U_a - L_b \leq \delta(a,b)$ then this period constraint is redundant. In either case we need not process the fanouts of gate b. Since this is true for any c, reachable from gate b, we get the following rule.

Rule 2 If during the shortest path calculations from source a using the Dijkstra's algorithm, for any gate b we have $U_a - L_b \leq \delta(a,b)$, we do not need to process the fanouts of gate b.

Notice that unlike TIM, our approach does not compute the full (all-pairs) shortest path matrix, because of the pruning provided by Rule 2.

To increase the efficiency of period constraint generation further, we reuse some of the computations performed in obtaining the δ values. The idea is motivated by the fact that in many practical circuits a high percentage of gates are single-fanout gates. Consider any such gate a with the single fanout b. For gate a, the shortest path to every other gate must go through gate b, which implies that $\delta(a,c) = w'(e_{a,b}) + \delta(b,c)$. Therefore we can obtain the shortest paths from gate a by simply adding $w'(e_{a,b})$ to the shortest paths from gate b. Thus if we somehow ensure that shortest paths from gate b are obtained before those from gate a, we will save one complete execution of Dijkstra's algorithm (for gate a as source). We perform a preprocessing step to ensure that we process gate b before gate a. We found that we could obtain up to 50% savings in the time required to generate the period constraints using this technique.

4.3 Additional constraint reduction

Due to the transparent nature of level-sensitive latches the constraint set for level-clocked circuits is much larger than for the corresponding edge-triggered circuit. We now describe some efficient rules for reducing the period constraint set C'_p further.

Consider three gates a, b and c, such that gate b lies on the path from gate a to gate c.

If gate b is a fanin of gate c then we have

$$\begin{aligned} C1: \quad & r(a) - r(b) \leq & \Delta(a,b) \\ C2: \quad & r(b) - r(c) \leq & w(e_{bc}) \\ C3: \quad & r(a) - r(c) \leq & \Delta(a,c) \end{aligned}$$

If $\Delta(a,b) + w(e_{bc}) \leq \Delta(a,c)$ then constraint $C3$ is redundant and can be dropped. This leads us to the following rule

Rule 3 If b and c are two gates reachable from gate a, such that gate b is a fanin of gate c and $\Delta(a,b) + w(e_{bc}) \leq \Delta(a,c)$ then the period constraint from gate a to gate c is redundant and can be dropped.

If gate b is a fanout of gate a then we have

$$\begin{aligned} C4: \quad & r(a) - r(b) \leq & w(e_{ab}) \\ C5: \quad & r(b) - r(c) \leq & \Delta(b,c) \\ C6: \quad & r(a) - r(c) \leq & \Delta(a,c) \end{aligned}$$

If $w(e_{ab}) + \Delta(b,c) \leq \Delta(a,c)$ then constraint $C6$ is redundant and can be dropped. This leads us to the following rule.

Rule 4 If gate b is a fanout of gate a and gate c is some gate reachable from gate a, then if $w(e_{ab}) + \Delta(b,c) \leq \Delta(a,c)$ then the period constraint from gate a to gate c is redundant and can be dropped.

Rule 1 and Rule 2 prune the constraints because the information in the bounds on r variables makes some constraints redundant. Rule 3 and Rule 4 on the other hand prune the constraints because of the

discrete nature of the Δ values. These rules can be generalized to include implication by more than two constraints; these generalized rules will, however, be computationally expensive to apply. Rules 3 and 4 on the other hand, can be efficiently applied, to drop redundant constraints as they are generated. This is because we generate the period constraints from one gate (say gate a) at a time, therefore both $\Delta(a, b)$ and $\Delta(a, c)$ are available in the same iteration, and Rule 3 can be efficiently applied. Because of the reuse of δ computations, if gate b is the fanout of a single-fanout gate a we have both $\Delta(a, c)$ and $\Delta(b, c)$ available at the same time and Rule 4 can be efficiently applied.

5 Experimental Results

We present results for the larger circuits in ISCAS-89 benchmark suite, and some other large circuits (myex1 through myex3) created by combining ISCAS-89 circuits using our approach, Minaret-L. As in [3], to obtain level-clocked circuits, we replaced each edge-triggered FF in the ISCAS-89 circuit by two level-sensitive latches.

In Table 1 we show for each circuit, the number of gates $|G|$, the number of latches, the problem size, and the execution time. The circuits are retimed for the minimum possible period, and the number of latches for both minperiod retiming [7] (initial) and minarea retiming by Minaret-L (for the same clock period) obtained after taking into account the maximum register sharing [1] are shown. For almost all circuits minarea retiming reduces the number of latches in the circuit by a factor of two to three as compared to minperiod retiming, even though both retime the circuit for the same clock period. This underscores the importance of minarea retiming.

We compare the size of the LP in Minaret-L given in Equation (8) and the original LP in Equation (7) by presenting the number of variables and constraints in both, and the percentage reduction as R_v and R_c respectively. It can be seen that up to three orders of magnitude reduction is obtained in the number of constraints by using Minaret-L. The number of unpruned constraints grow at the rate of $O(|G|^2)$ and our pruning techniques reduce this rate of growth significantly.

The CPU time shown is on a DEC AXP system 3000/900 workstation with 256M RAM, and includes the time spent in generating the bounds, generating the LP and solving it. The time required to generate the LP dominates the total execution time signifying the importance of the efficiency enhancing techniques of Section 4.2. Although the reduction in LP size obtained by the bounds is significant, the time spent in obtaining them was an insignificant fraction of the total CPU time (e.g., 15 seconds for myex3). The small execution time of Minaret-L highlights the effectiveness of our techniques. Minaret-L can retime large level-clocked circuit for minimum area in time comparable to that required for retiming edge-triggered circuits, thus bringing the techniques for retiming level-clocked circuits at par with the state of art in retiming techniques for edge-triggered circuits.

Figure 1 to Figure 4 show area-delay tradeoff curves obtained from ASTRA [7] and Minaret-L, for some ISCAS89 circuits. Since ASTRA pays no attention to minimizing the number of latches, there is no reason for the area-delay curve to be monotonic. The Minaret-L curve on the other hand must, by definition, be monotonic.

6 Conclusion

A fast algorithm for minarea retiming of large level-clocked circuits has been presented. The entire ISCAS-89 benchmark suite could be retimed in minutes. This work unifies the two approaches to retiming of level-clocked circuits, i.e., the TIM [3] and the ASTRA [7] approaches. This unification together with the other techniques (Rule 2, reuse of δ computation, Rule 3 and Rule 4) leads to an efficient method of generating a much smaller LP, with two to three orders of magnitude less constraints. It makes it feasible to retime large level-clocked circuits (over 56,000 gates) for minimum area in very reasonable time (under 1.5 hours). To put this in perspective, the largest level-clocked circuit for which minarea retiming results had been published in the past had less than 400 gates [3].

References

[1] C. E. Leiserson and J. B. Saxe, "Retiming synchronous circuitry," *Algorithmica*, vol. 6, pp. 5–35, 1991.

[2] N. Shenoy, R. K. Brayton, and A. Sangiovanni-Vincentelli, "Retiming of circuits with single phase transparent latches," in *Proceedings of the IEEE International Conference on Computer Design*, pp. 86–89, 1991.

[3] M. C. Papaefthymiou and K. H. Randall, "TIM: A timing package for two-phase, level-clocked circuitry," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 497–502, 1993.

[4] B. Lockyear and C. Ebeling, "Optimal retiming of level-clocked circuits using symmetric clock schedules," *IEEE Transactions on Computer-Aided Design*, vol. 13, pp. 1097–1109, Sept. 1994.

[5] N. Shenoy and R. Rudell, "Efficient implementation of retiming," in *Proceedings of the IEEE/ACM International Conference on Computer-Aided Design*, pp. 226–233, 1994.

[6] S. S. Sapatnekar and R. B. Deokar, "Utilizing the retiming skew equivalence in a practical algorithm for retiming large circuits," *IEEE Transactions on Computer-Aided Design*, vol. 15, pp. 1237–1248, Oct. 1996.

[7] N. Maheshwari and S. S. Sapatnekar, "A practical algorithm for retiming level-clocked circuits," in *Proceedings of the IEEE International Conference on Computer Design*, pp. 440–445, 1996.

[8] N. Maheshwari and S. S. Sapatnekar, "An improved algorithm for minimum-area retiming," in *Proceedings of the ACM/IEEE Design Automation Conference*, pp. 2–7, 1997.

[9] T. H. Cormen, C. E. Leiserson, and R. L. Rivest, *Introduction to Algorithms*. New York, NY: McGraw-Hill, 1990.

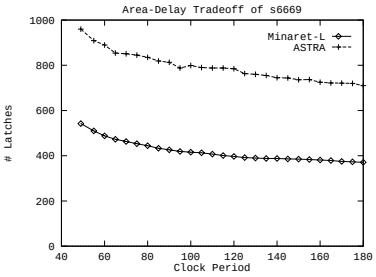

Figure 2: Area-Delay Tradeoff of s6669

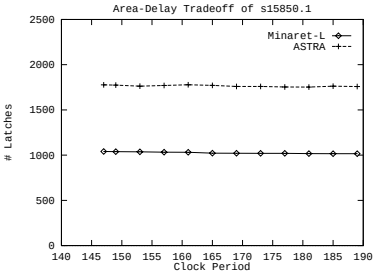

Figure 3: Area-Delay Tradeoff of s15850.1

Table 1: Minarea retiming by Minaret-L

Circuit	$\|G\|$	# Latches		# Variables			# Constraints			CPU Time
		Minarea	Initial	Minaret-L	Original	R_v	Minaret-L	Original	R_c	
s3384	1,685	337	638	2,006	2,166	7.39%	55,980	761,365	92.65%	2.56s
s4863	2,342	234	473	2,706	2,995	9.65%	72,451	5,481,911	98.68%	5.36s
s5378	2,779	286	480	2,970	3,664	18.94%	31,765	4,595,645	99.31%	3.22s
s6669	3,080	542	960	3,735	4,100	8.90%	20,841	1,923,524	98.92%	6.17s
s13207.1	7,791	890	1,795	7,656	9,180	16.60%	55,395	22,908,799	99.76%	18.61s
s15850.1	9,617	1,041	1,777	9,013	11,332	20.46%	69,142	39,493,334	99.83%	45.82s
s35932	16,065	3,523	4,144	20,264	21,716	6.69%	189,068	130,080,328	99.85%	1m:7.26s
s38584.1	19,253	2,852	7,558	20,590	23,390	11.97%	127,488	293,482,797	99.96%	1m:57.52s
s38417	21,370	2,766	4,938	25,735	25,923	0.73%	2,446,798	149,492,588	98.36%	6m:26.99s
myex1	28,946	3,891	9,065	30,489	34,417	11.41%	154,603	504,055,977	99.97%	6m:37.48s
myex2	40,661	5,551	13,820	48,560	49,214	1.33%	3,638,182	819,701,299	99.56%	31m:16.52s
myex3	56,761	9,041	17,019	70,000	70,414	0.59%	8,207,036	1,624,913,333	99.50%	1h:19m:43.07s

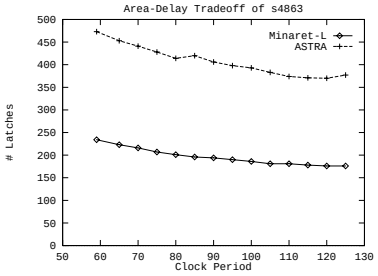

Figure 1: Area-Delay Tradeoff of s4863

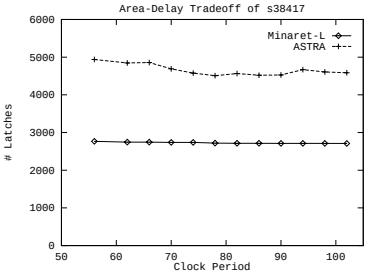

Figure 4: Area-Delay Tradeoff of s38417