Modelling Flip-flop Delay Dependencies in Timing Analysis

Abstract

In this paper, we have modelled the flip-flop clock to output delay dependency on the data arrival time and introduced this phenomenon in timing analysis. Traditionally, finding the minimum clock period of a flip-flop based sequential design was based on the assumption that the setup-time and clock to output delay of a flip-flop are constant and hence each stage of the pipeline can be analyzed independently. However, it is well known that the delay of a flip-flop depends on the data arrival time at its input and hence there exists an interdependence among different pipeline stages. The problem of finding the minimum clock period of such a coupled system is a non-trivial problem. In this paper, we formulate the problem of finding the minimum clock period of a flip-flop based sequential circuit accounting for these dependencies. We show that the problem is a non-linear convex optimization problem. We propose three different solution approaches and compare their results on ISCAS '89 sequential benchmark circuits. Modeling these data arrival time dependencies we have seen a consistent decrement of approximately 50-60ps compared to the traditional approach using constant setup-time and flip-flop delays. We also show how the analysis can be extended to account for hold time constraints for short paths in the circuit.

1. Introduction

Static timing analyzers are widely used to verify the behavior of large digital circuit designs in various stages of design. They have also become the core engine used inside circuit optimization tools, such as the transistor and gate sizing tools and logic synthesis. Ever since the frequency of operation of circuits has gained marketing focus, accurate and efficient computation of a circuit's clock period is one of the most important tasks in design.

Flip-flop delay is an increasing concern for high performance circuit designers. Cycle times have been shrinking dramatically driven by both faster gate delays and by more aggressive designs using fewer gates in a single pipeline stage. Thus, flip-flop delay accounts for a significant portion of the clock period and therefore, accurate modelling of this delay is critical. Also, less pessimistic STA algorithms based on better models help reduce the overall clock period of the circuit, by aiding better design and verification techniques.

Traditionally, the task of determining the minimum clock period of a flip-flop based sequential circuit has consisted of finding the longest combinational path between any two flip-flops of the circuit and adding setup-time, flip-flop delay and clock skew to it. (Note that the effect of clock jitter on clock period has been neglected in the entire paper without the loss of generality.) The above formulation is based on the assumption that the setup time of a flip-flop is fixed and the clock to output delay of a flip-flop is constant. This results in the major simplification that each pipeline stage can be considered independently from every other pipeline stage of the circuit. However, it is well known that the delay of a flip-flop is in fact a function of the difference between the clock and the data arrival time, as shown in Figure 1. We define T_{Diff} as the difference between the clock and data arrival time and refer to T_{ff} as the clock to output delay of the flip-flop. As shown in Figure 1, T_{ff} increases with decreasing T_{Diff} until the flip-flop goes metastable and T_{ff} approaches infinity. Hence, both T_{ff} and T_{Diff} of a flip-flop must be considered as variables with mutual

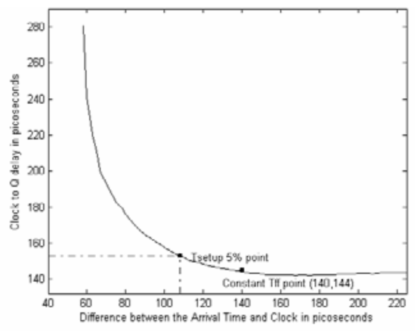

Figure 1. Flip-flop delay dependence on arrival time.

dependencies rather than being treated as constants. We show in this paper that the simplification of using a fixed setup time and constant flip-flop delay may yield a pessimistic minimum clock period, and it is therefore necessary to consider these interdependencies between pipeline stages. The problem of finding the clock period of circuit considering this dependency has similarity with the problem of finding the clock period of a latch based sequential circuit in [1][10][11] except for the fact that this problem is non-linear while latter in can be solved linearly.

In this paper, we model the flip-flop T_{ff} dependencies on data arrival times and propose a new timing analysis algorithm that accounts for these dependencies, which to our knowledge is the first effort to address this problem formally. We describe the problem of determining the minimum clock period as non-linear optimization problem and prove that it is convex. We propose three solution methods, a sequential quadratic programming approach, simple rectilinear manhattan decent approach, a satisfiability approach, and have compared their performance in terms of their runtime and number of iterations on ISCAS'89 sequential benchmark circuits. We have also discussed their ease of integration with the existing static timing analyzers. We compared the exact solution obtained from the proposed formulation with the traditional approach using different T_{setup} times and observed a reduction in the clock period of approximately 50-60ps, which is equivalent to approximately one fan out of four delay in this technology.

The treatment of T_{ff} as a variable also has implications on the short path constraints of a given circuit. As the data arrival time gets closer to the clock the T_{ff} increases thus reducing the number of hold time violations. We therefore show how our analysid can be extended to determine the lowest frequency that fulfills all the hold-time constraints. The long path analysis therefore finds the maximum frequency for which the long path constraints are met whereas the short path analysis finds the minimum frequency for which all short path constraints are satisfied. For any functionally correct design, its is therfore neccessary for $f_{short} < f_{long}$. This results in a window of valid operating frequency as shown in Figure 2.

The remainder of the paper is organized as follows. Section 2 discusses the traditional method of finding the clock period and

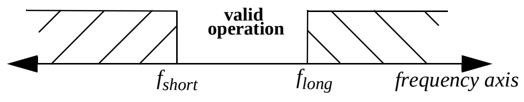

Figure 2. Window of valid operating frequencies

the model of the flip-flop delay. Section 3 discusses the non-linear problem formulation and discusses its properties. Section 4 discusses three different approaches used to solve this non-linear problem. Section 5 describes the short path problem and its analysis, Section 6 presents results and Section 7 concludes this paper.

2. Background and Delay Model

In this section, we describe the traditional model used to compute the clock period of a circuit. We then present a new model for flip-flop clock to output delay Vs. data arrival time. We end the section with an example that demostrates that the clock period computed based on traditional model is pessimistic. We introduce the following nomenclature.

Nomenclature

T_c	Clock period of the circuit.
T_{setup}	Setup time for the flip-flop.
T_{ff}	Flip-flop clock to output delay.
T_{hold}	Hold time of flip-flop.
T_{skew}	Characterized skew of the clock network.
n	Number of flip-flops in the circuit.
$T_{c,ij}$	Effective delay between i^{th} and j^{th} flip-flop.
$T_{Logic,ij,max}$	Maximum combinational delay between i^{th} and j^{th} flip-flops.
$T_{Logic,ij,min}$	Minimum combinational delay between the i^{th} & j^{th} flip-flop.
$T_{Diff,j}$	Difference between clock and data arrival at j^{th} flip-flop.
$T_{Diff,ij}$	Difference between the clock and data arrival for a path between i^{th} and j^{th} flip-flops.

Traditionally, computing clock period of a sequential circuit is performed as described below.

$$T_{c,\,ij} = T_{setup} + T_{ff,i} + T_{Logic,ij,max} + T_{skew} \qquad \text{(EQ 1)}$$

$$T_c = max\{T_{c,\,ij}\}\ \forall(i, j \le n) \qquad \text{(EQ 2)}$$

T_{setup} is defined as the difference between the clock and the arrival time where T_{ff} increases by 5% from its nominal value as suggested in [2] and is shown as Figure 1.

The fundamental assumption in the above timing model is that the setup time and the flip-flop delay are constant, with the flip-flop going metastable and the delay approaching infinity as soon as the difference between the clock and arrival time is less than T_{setup}. However, as shown in Figure 1, the transition between proper functionality and metastability is not as discrete as this model describes. The delay of a flip-flop increases gradually as the difference between the arrival time and clock decreases, creating a dependency of T_{ff} on the data arrival time.

In order to include this effect of variable flip-flop delay in timing analysis, it is important to accurately model the delay of a flip-flop as a function of its data arrival time. Based on the model used to resolve the metastability of a latch we propose the model below in EQ 3 for the flip-flop delay.

$$T_{ff} = A\exp(B \cdot T_{Diff}) + C \qquad \text{(EQ 3)}$$

where A,B,C are real non-zero constants.

This model can be easily fit based on the nonlinear least square method using the trust region dogleg algorithm, with weighted robust regression[12][3].

In Figure 3, we show a comparison between the proposed model in EQ3 and the HSPICE delay for a D-flip-flop from an industrial 0.18 micron library with a loading of 50 ff at 75° C. We fitted the proposed model for four different flip-flop cells from a comercial library. Table 1 summarizes the quality of the fit for different loads, with input slope of 1.8V/.1ns and temperature of 70° C. In Table 1, RMSE is the Root Mean Square Error whereas

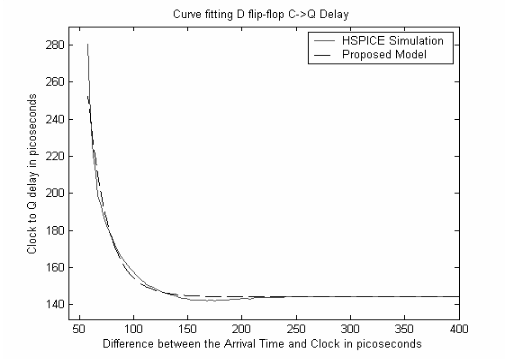

Figure 3. Proposed Model for Flip-Flop Delay

Table 1. Quality of Fit for Flip-flop Delay Model

Flip-flops types in library	Output Loads	Error	
		R-Square	RMSE (ps)
DFFNX1	.1pf	.99476	1.4113
	.05pf	.99483	1.4023
DFFX2	.1pf	.99913	.50861
	.05pf	.99919	.49134
DFFHQX4	.1pf	.98554	1.9571
	.05pf	.98858	1.9542
DFFHQXL	.1pf	.99435	0.8251
	.05pf	.99619	.67761

R-square is a statistical metric used to denote how successful the fit is in explaining the variation of the curve, about its mean. The ideal value for R-square is 1. From Table 1 it is evident that the model mentioned in EQ3 is a good match for describing the flip-flop delay dependence on its data arrival time.

We now show with an example how the traditional approach computes a pessimistic clock period. Consider a hypothetical cyclical sequential circuit as shown in Figure 4. For the sake of simplicity, we assume that both flip-flops in this circuit are of the same kind with the same loading and identical delay dependency curves. Furthermore, we assume that the circuit delay parameters are as follows: T_{ff} = 150ps, T_{Logic1}= 700ps, T_{setup} = 108ps, T_{skew} = 0ps.

Based on these parameters and applying EQ 1 we obtain:

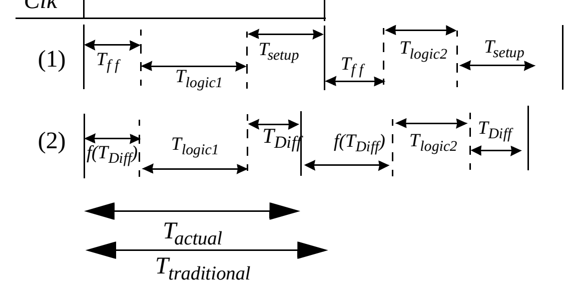

Figure 4. Typical Sequential Circuit and its timing

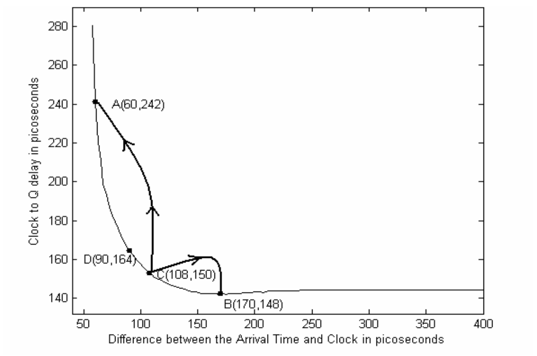

Figure 5. Effect On Tc considering variable Flip-flop delays

T_{c12} = 700+150+108 = 958ps; T_{c21} = 560+150+108 = 818ps

Applying EQ 2 we get: $T_c = max$ {958, 818} = 958 ps

However, it is possible to decrease T_{Diff} for the first pipeline stage which would translate into a higher T_{ff} for *Dff2*. Since, the clock period of the circuit is determined by the critical path in stage 1, it is possible to increase $T_{ff,2}$ for the non-critical path in stage 2. As shown in Figure 5, it is possible for the *Dff1* to operate at point B and *Dff2* to operate at point A rather than both operating at the point C as suggested by the traditional approach. The time period after considering these new operating points is calculated as follows:

T_{c12} = 700+150+60 = 910ps; T_{c21} = 560+242+108 = 910ps

$T_c = max$ {910, 910} = 910ps

Modelling of the flip-flop delay dependency on the arrival time therefore has resulted in a reduction of the clock period by 48ps. Thus, the actual silicon can run faster than predicted and the mismatch is solely due to the inadequacy of the traditional model. It is important to note that uncertainity in the data arrival time has already been modelled by conservative delay characterization of combinational path between flip-flops.

3. Problem Formulation and Properties

We now formulate the problem of finding the clock period of a flip-flop based sequential circuit that takes into account the delay dependency on the arrival time relative to its clock:

$$T_{c,\,ij} = T_{Diff,\,ij} + T_{ff,\,i} + T_{Logic,\,ij,max} + T_{skew} \forall(i, j \le n) \quad \text{(EQ 4)}$$

$$T_{Diff,\,j} = min\{T_{Diff,\,ij}\} \quad \forall(i \le n) \quad \text{(EQ 5)}$$

$$T_{ff,\,j} = A \cdot \exp\{\dot{B} \cdot T_{Diff,\,j}\} + C \quad \text{(EQ 6)}$$

$$T_c = \{max\{T_{c,\,ij}\}\} \quad \forall(i, j \le n) \quad \text{(EQ 7)}$$

$$Objective\ :\ min\{T_c\} \quad \text{(EQ 8)}$$

The objective in the above formulation is to minimize T_c. through the optimal allocation of all $T_{Diff,ij}$ in the circuit. From the above formulation it is clear that T_c is a convex function in terms of the optimization variables $T_{Diff,ij}$. Though the proof of this convexity is straightforward, it has significant implication on the optimization problem and hence we include the proof in this paper.

Theorem 1. T_c is a convex function with $T_{Diff,ij}$ as variables.

Proof:

We first observe that $T_{Diff,j}$ in (EQ5) is a convex function of variables $T_{Diff,ij}$ as the minimum function retains convexity if its arguments are convex. Also, $T_{ff,j}$ (EQ6) is convex as $T_{Diff,j}$ is convex and exponential functions are convex. From this follows that $T_{c,ij}$ (EQ4) is convex as the sum of convex functions is a convex function and consequently we find that T_c (EQ7) is a convex function as $T_{c,ij}$ is convex and the maximum function also retains convexity if its arguments are convex. □

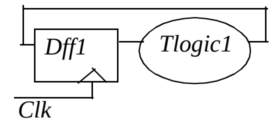

Figure 6. Equivalent circuit for Figure 4

We now examine the properties of this optimization function both when the delay of the logic stages are balanced and when they are unbalanced.

Balanced Logic.

We consider the circuit as shown in Figure 4. Since both the pipeline stages are identical, we can replace the circuit by a single pipeline stage as shown in the Figure 6, as far as the task of finding clock period is concerned. Note that the circuit in Figure 6 is simply the unrolled version of the circuit in Figure 4. The clock period for the circuit in Figure 4 is given by:

$$T_c = T_{Logic,\,11} + T_{Diff,\,11} + A \cdot \exp(B \cdot T_{Diff,\,11}) \quad \text{(EQ 9)}$$

To determine the minimum T_c we must satisfy $\frac{dT_c}{dT_{Diff,i}} = 0$. Applying this criteria in EQ9, we obtain the condition $\frac{dT_{ff,\,i}}{dT_{Diff,\,i}} = -1$. This means that the flip-flop operates at the point where the slope of the curve in Figure 5 reaches -1, as shown by point D. This is intuitively clear since we can only trade-off T_{Diff} for T_{ff} until the increase in T_{ff} exceeds the decrease in T_{Diff} leading to the optimal operating point of $\frac{dT_{ff,\,i}}{dT_{Diff,\,i}} = -1$. Operating beyond this point would add more T_{ff} than decreasing T_{Diff} and hence would increasing T_c.

The optimization space for T_c as a function of $T_{Diff,1}$ and $T_{Diff,2}$ for the circuit in Figure 7 with balanced loads is shown in Figure 7. Clearly, T_c is convex and its symmetric nature is due to the balanced stage delays $T_{Logic,1}$ and $T_{Logic,2}$. The optimization space can be viewed as the maximum of two intersecting surfaces defined by $T_{c,12}$ and $T_{c,21}$. From, this follows the useful observa-

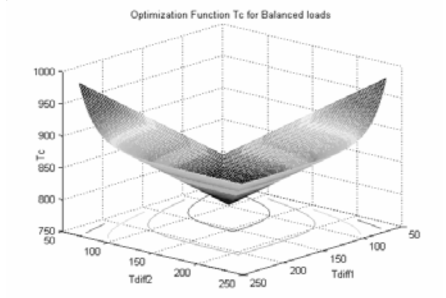

Figure 7. Optimization Function Tc with Balanced Loads

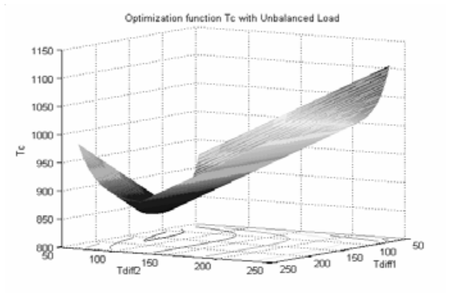

Figure 8. Optimization Function Tc for Unbalanced Loads

tion that for the designs that are very balanced the traditional method of defining T_{setup} and T_{ff} should be replaced with the definition of T_{setup} and T_{ff} corresponding to the negative unity gain point of T_{ff} Vs T_{Diff} curve as shown by point D in Figure 5.

Unbalanced logic:

We now examine the case where the delay of the logic stages is unbalanced, allowing reallocation between the flip-flop delays. Assume in Figure 8 $T_{Logic,2} > T_{Logic,1}$ then, the arrival time of the flip-flop2 can be increased thus increasing its delay. Generally, it is possible to decrease the T_{Diff} for the critical path while this decrease does not cause the delay of the next stage pipeline to exceed the critical path itself. It is possible to show that the optimal allocation for T_{Diff} for minimum T_c will fulfill the condition: $\prod_i \left|\frac{dT_{ffi}}{dT_{Diffi}}\right| = 1$. Figure 8 shows the optimization space T_c with unbalanced loads. In this case, the minimum occurs for a smaller $T_{Diff,2}$ and larger $T_{Diff,1}$.

4. Proposed Solution Approaches

In this section, we describe three approaches used to solve the above non-linear problem of finding the minimum valid clock period.

1. Sequential Quadratic Programming (SQP)

We solved the above problem using a non-linear optimizer, treating the above optimization function as the minmax problem. We use a Sequential Quadratic Programming (SQP) method in MATLAB [4]. Modifications were made to the line search and Hessian. In the line search an exact merit function ([6] and [7]) is replaced with the merit function proposed by [8] and [9]. The line search is terminated when the merit function shows improvement. This method, while very general is not very suited for integration with a timing analyzer. The method operates on a matrix that represents the connectivity of the circuit where each element E_{ij} represents the maximum delay of the combinational path between flip-flop j and flip-flop i. Each entry in the matrix therefore requires a separate timing analysis run, leading to a high run time. However, the results from this approach were used as reference to check the results obtained with the other two approaches.

2. Rectilinear Manhattan Decent (RMD)

As the optimization function is convex, any decent in downward direction approaches to the global minimum of the function. We therefore repeatedly decrease the variable T_{Diff} to which T_c is most sensitive, by a small step. In the circuit, the most sensitive T_{Diff} can be identified by simply finding the critical path and selecting the T_{Diff} of the capturing flip-flop. Each time the variable T_{Diff} is decreased, the delay of the critical stage and the subsequent stage are updated and a new critical path is selected. We compare the new T_c with the previous T_c in each iteration and continue iterating until T_c increases by a greater amount that the step size. The steps of the algorithm are as follows.

1. *Calculate the critical path.*
2. *Decrease corresponding T_{Diff} by a small step and update the timing of the critical stage and the subsequent stage.*
3. *If the increase in new T_c is less than the step size, goto step 1 else, previous T_c was optimal.*

3. Satisfiability Approach (SA)

This method performs a bounded binary search on the optimization space by testing whether the circuit can satisfy a given clock period. We start with an initial upper and lower bound on T_c and terminate the optimization process when the bounds converge. This approach is described as follows:

a *$T_{upperbound} = max\{T_{Logic,ij}\} + K$ (where K is a constant that ensures the upper boundness)*

b *$T_{lowerbound} = max\{T_{Logic,ij}\}$*

c *$T_{current} = mean\ \{T_{upperbound}, T_{lowerbound}\}$*

d *Pick an initial pipeline stage between two flip-flops and calculate T_{Diff} as the difference of $T_{current}$ and $(T_{ff} + T_{logicij})$ for the corresponding pipeline.*

e *Using this T_{Diff}, calculate T_{ff} based on EQ3 for the next flip-flop and propagate this effect through the circuit.*

f *If at any stage T_{Diff} becomes negative, then the circuit is not able to function at $T_{current}$. Change $T_{lowerbound} = T_{current}$ and goto step 3.*

g *Else if the propagation of T_{Diff} in the circuit converges, change $T_{current} = T_{upperbound}$ and goto step 3.*

h *Continue iteration until $T_{upperbound}$ and $T_{lowerbound}$ converge.*

It is important to note that in this method we compute T_{Diff} based on $T_{current}$ such that for all stages $T_{c,ij} = T_{current}$. By computing T_{Diff}s in this manner, we ensure that every path in the circuit is critical. Each time we change a T_{Diff}, based on a change in $T_{current}$, we impact the flip-flop delay of the next stage propagate this change through the whole circuit. If there are cyclic paths in the circuit, iterations may be required before the circuit stabilizes. The circuit is said to satisfy the clock period $T_{current}$, if all the T_{Diff}s remain unchanged from the previous iteration with none of them are negative. If a T_{Diff} becomes negative during the iterations, the circuit cannot meet $T_{current}$.

We now show that the optimization method reaches the global minimum. Figure 9 shows the entire optimization space T_c. By, calculating T_{Diff}s based on $T_{current}$, as mentioned earlier, we ensure that every path in the circuit is critical. This corresponds to solutions in the optimization space that lie on the black line in Figure 9. Henceforth, we shall refer this black line as the critical line. It is clear that the global minimum is on this critical line. Also, in the above proposed method we effectively test whether there exist some intersection point between the $T_{current}$ surface and critical line. In each iteration we change the bounds and hence $T_{current}$ in such a way that we descend down the critical line is as indicated by the arrow in Figure 9. Hence with each iteration our bounds converge to the minimum point. This method can be argued as a variant of steepest decent algorithm. Figure 9 shows the critical

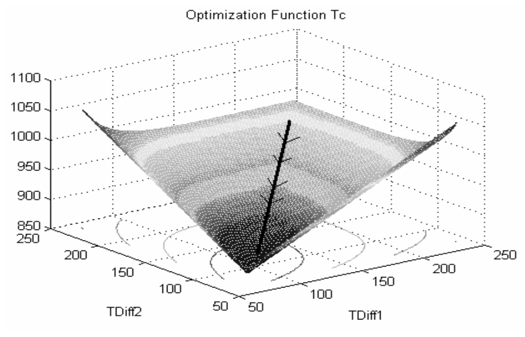

Figure 9. Optimization Function T_c

line for a circuit as shown in Figure 9 with balanced loading. It is noteworthy to clarify that the critical line might be a curve in a multidimensional optimization space for T_c with unbalanced loads.

5. Short Path Problem Formulation and its Analysis

In this section, we first mention the traditional method of identifying short paths and then discuss the implication of variable flip-flop delays on this problem. We then formulate the problem of finding the maximum clock period such that all the short path constraints are met.

In a traditional analysis, a hold time violation is reported when the sum of the flip-flop clock to output delay and the minimum combinational logic delay between the pipeline registers is less than the sum of characterized clock skew and hold time of a flip-flop:

$$T_{ff,i} + T_{Logic,ij,min} < T_{skew} + T_{hold,i} \quad \forall(i, j \le n) \qquad \text{(EQ 10)}$$

It is important to mention that the hold time of a flip-flop is a constant with respect to data arrival time. As suggested earlier in section 2, the clock to output delay of a flip-flop is not a constant but a variable, and a function of data arrival. As the data arrival time gets closer the clock, the delay of the flip-flop increases as shown in Figure 3. This increase in T_{ff} helps reduce the short path violations as evident from EQ10. Thus, as we decrease the clock period, the flip-flop delay increases and more flip-flops in a design will meet short path constraint.

We illustrate this effect using the circuit shown in Figure 4 with following parameters, T_{skew} = 35ps, T_{hold} = 225ps, $T_{Logic,1}$ = 50ps Traditional analysis assumes that Dff1 always operates at point C in Figure 5. With this operation point a short path violation is reported for the above case. But it is quite possible for DFF1 to operates at point A, where the short path violation no longer exists. In this case, the short path is analyzed pessimistically by the traditional analysis, where as in other cases, the traditional analysis may be optimistic.

We now formulate the problem of finding the maximum clock period of a flip-flop based sequential circuit with no hold time violations that takes into account flip-flop delay dependency on the arrival time relative to its clock.

$$T_{c,\,ij} = T_{Diff,\,ij} + T_{ff,\,i} + T_{Logic,\,ij,min} \quad \forall(i, j \le n) \qquad \text{(EQ 11)}$$

$$T_{Diff,\,j} = max\{T_{Diff,\,ij}\} \quad \forall(i \le n) \qquad \text{(EQ 12)}$$

$$T_{ff,\,j} = A \cdot \exp\{B \cdot T_{Diff,j}\} + C \qquad \text{(EQ 13)}$$

$$T_{ff,i} + T_{Logic,ij,min} < T_{hold,i} + T_{skew} \quad \exists(i \le n) \qquad \text{(EQ 14)}$$

$$T_c = max\{T_{c,\,ij}\} \quad \forall(i \le n) \qquad \text{(EQ 15)}$$

$$Objective\ :\ min\{T_c\} \qquad \text{(EQ 16)}$$

Note that the above formulation finds the *minumum* clock period such that there does exist at least one short path violation. Hence T_c-δ is the *maximum* clock period such that all hold time constaints are met.

The solution of this problem as suggested in EQ16 is the lowest frequency for which all the short path constraints are satisfied whereas the solution of former problem in EQ8 is the highest frequency for which all the long path constraints are satisfied. In Figure 2 these two frequency constraints f_{short} and f_{long} are shown. For any properly functional design $f_{short} < f_{long}$ to allow a region of correct operation. By padding the short path delays, we essential lower the lowest short path frequency and the amount of padding determines the size of the feasible region of operation.

6. Results

The discussed methods discussed in Section 4 were implemented and tested on ISCAS '89 sequential benchmark circuits. We used the netlist as given by cbl.ncsu.edu and optimized them using *SYNOPSIS Design Analyzer*[9] keeping the sequential circuit elements intact. Table 2 shows the runtime and the number iterations for the three approaches discussed in Section 4. Table 2 presents the following information in column order: Circuit name, number of flip-flops in that circuit, runtime for Sequential Quadratic Programming (SQP), runtime for Rectilinear Manhattan Decent (RMD), runtime for Satisfiability Approach (SA), number of iterations for Rectilinear Manhattan Decent and number of iterations in Satisfiability Approach. The three solutions reached equivalent solutions, given the numerical accuracy of the analysis.

In Table 3, we compare the exact solution obtained by modelling flip-flop delay dependencies on the data arrival time with the

Table 2. Comparing Runtime and Iterations of Three Approaches

		Runtime in seconds			Iterations	
Benchmarks	**#FF**	**SQP**	**RMD**	**SA**	**RMD**	**SA**
s953	29	3.726	0.4110	0.02	46	21
s9234_1	211	245.1	6.88	0.04	367	21
s9234	228	207.05	4.3970	0.06	205	21
s838_1	32	4.87	0.14	0.02	46	21
s713	19	1.112	0.10	0.01	29	21
s641	19	1.21	0.091	0.02	29	21
s526n	21	1.522	0.11	0.01	36	21
s526	21	1.492	0.1	0.01	45	21
s400	21	1.452	0.16	0.01	147	21
s420_1	16	1.02	0.11	0.02	186	21
s382	21	1.252	0.11	0.01	64	21
s349	15	0.752	0.10	0.02	26	21
s344	15	0.731	0.08	0.01	26	21
s27	3	0.13	0.11	0.02	22	19
s13207_1	638	1192	23.405	0.23	26	21
s13207	669	1222	24.612	0.75	26	21
s15850_1	534	1021	19.89	0.22	100	21
s1238	18	0.531	0.06	0.04	26	21

traditional method when using various fixed T_{setup} times on the delay curve (Figure 1). Table 3, column 2 shows the result obtained by modelling the flip-flop delay dependency using the non-linear problem formulation and column 3, 4, 5 shows the results obtained with traditional timing analysis using the 5% delay point mentioned in Section 2, the delay change curve point where d/dx = -1 mentioned in Section 4 and shown as D in Figure 5, and a constant T_{ff} delay point as shown in Figure 1. The results show significant improvement by modelling the flip-flop delay dependency over the traditional approach.

In addition to this we did compare our predicted highest long path frequency with HSPICE simulation. This simulation was done on some very smaller circuits that allowed for full SPICE simulation. The flip-flop delays were characterized using the same slopes at the input as seen in the HSPICE. The results are tabulated in Table 4. We see that the predicted clock period matches that with HSPICE results with a maximum deviation of 11ps and average deviation of 5ps.

7. Conclusion

In conclusion, we have presented a new timing model for computing the clock period of flip-flop based sequential circuits. The timing analysis based on this models is more accurate as compared to the traditional models as it accounts for flip-flop delay dependencies on the data arrival time. We showed that the problem of finding the maximum clock period of a flip-flop based sequential circuit considering these dependencies is a nonlinear but convex optimization problem. We proposed three approaches to solve this problem. Modelling flip-flop delay dependencies on data arrival time, we obtained a decrease in clock period of 50-60ps, which in current circuits with shallower pipelines can be a significant part of the clock period. We have also formulated the problem of finding the lowest frequency such that all the short path constraints are met. Together these two formulation form the basis of a static timing analyzer that models flip-delay dependencies.

Table 3. Clock Period calculated by Modelling flip-flop delay dependency and Traditional approach Using various different T_{setup} time (ps).

Benchmarks	New Approach	5% T_{setup} Time	d/dx = -1 T_{setup} Time	Constant T_{ff} T_{setup} Time
s953	2774.2	2828	2825	2854
s9234_1	4504.2	4578	4555	4584
s9234	4484.2	4538	4535	4564
s838_1	6531.2	6588	6585	6614
s713	5404.2	5358	5355	5384
s641	5404.2	5358	5355	5384
s526n	2654.2	2708	2705	2734
s526	2634.2	2688	2685	2714
s400	2254.2	2308	2305	2334
s420_1	3574.22	3628	3625	3654
s382	2594.2	2648	2645	2674
s349	3264.2	3318	3315	3344
s344	3264.2	3318	3315	3344
s27	1192.2	1198	1195	1224
s13207_1	5034.2	5088	5085	5114
s13207	4994.2	5048	5045	5074
s15850_1	6344.22	6448	6445	6474
s1238	3344.22	3398	3395	3424

Table 4. Comparison of HSPICE and Predicted results for long path analysis.

Circuit	Load Type	Hspice Result	Predicted Model	Mismatch
Circuit A	Balanced	2270ps	2265ps	-5ps
Circuit B	Unbalanced	2310ps	2313ps	3ps
Circuit C	Balanced	692ps	681ps	-11ps
Circuit D	Unbalanced	2290ps	2291ps	1ps

References

[1] Sakallah, K.A; Mudge, T.N; Olukotun O.A "Analysis and Design of Latch-Controlled Synchronous Digital Circuits" *IEEE Trans. CAD*, Vol.11 Issue3, pp. 322-333, Mar. 92

[2] Makovic, D; Nikolic, B; Bordersen R,W "Design and Analysis of Low Energy Flip-flop", *ISLPED*, pp. 52-54, Aug. 2001.

[3] Branch, M.A., T.F. Coleman, y. Li, "A Subspace, Interior, and Conjugate Gradient Method for Large Scale Bound Constrained Minimization Problems," *SIAM Journal on Scientific Computing*, Vol. 21, Number 1. pp 1-23, 1999.

[4] Brayton, R.K., S.W. Director, G.D. Hachtel, and L.Vidigal, "A New Algorithm for Statistical Circuit Design Based on Quasi-Newton Methods and Function Splitting," *IEEE Trans. Circuits and Systems*, Vol. CAS-26, pp. 784-794, Sept. 1979.

[5] Grace, A.C.W., "Computer-Aided Control System Design Using Optimization Techniques," Ph.D. Thesis, University of Wales, Bangor, Gwynedd, UK, 1989

[6] Han, S.P., "A Globally Convergent Method For Nonlinear Programming," *Journal of Optimization Theory and Applications*, Vol. 22, p. 297, 1977.

[7] Madsen, K. and H. Schjaer-Jacobsen, "Algorithms for Worst Case Tolerance Optimization," *IEEE Transactions of Circuits and Systems*, Vol. CAS-26, Sept. 1979

[8] Powell, M.J.D., "A Fast Algorithm for Nonlineary Constrained Optimization Calculations," *Numerical Analysis*, ed. G.A. Watson, *Lecture Notes in Mathematics*, Springer Verlag, Vol. 630, 1978.

[9] www.synopsys.com. *Design Analyzer* user Manuals

[10] Szymanski, T.G; Shenoy, N.; "Verifying Clock Schedules" *ICCAD*, pp 124-131, Nov. 1992

[11] Szymanski, T.G; "Computing Optimal Clock Schedules" *DAC*, pp. 399-404 June '92.

[12] More, J.J, D.C. Sorensen, "Computing a Trust Region Step," *SIAM Journal on Scientific and Statistical Computing*, Vol. 3,, pp 553-572, 1983