Rapid and Accurate Latch Characterization via Direct Newton Solution of Setup/Hold Times

Shweta Srivastava, Jaijeet Roychowdhury
Department of Electrical and Computer Engineering, University of Minnesota
Email: jr,shwetas@umn.edu

ABSTRACT

Characterizing setup/hold times of latches and registers, a crucial component for achieving timing closure of large digital designs, typically occupies months of computation in industries such as Intel and IBM. We present a novel approach to speed up latch characterization by formulating the setup/hold time problem as a scalar nonlinear equation $h(\tau) = 0$ derived using state-transition functions, and then solving this equation by Newton-Raphson (NR). The local quadratic convergence of NR results in rapid improvements in accuracy at every iteration, thereby significantly reducing the computation needed for accurate determination of setup/hold times. We validate the fast convergence and computational advantage of the new method on transmission gate and C^{2}MOS latch/register structures, obtaining speedups of 4-10$\times$ over the current standard of binary search.

I. INTRODUCTION

Finding the setup and hold times of latches and registers is a crucially important prerequisite for static and dynamic timing analysis of digital circuits [1]–[3]. As devices shrink, clock speeds become ever faster, and design margins become increasingly squeezed in high-speed digital systems, it becomes important to determine these quantities with the highest possible accuracy to ensure that timing analysis makes neither unduly optimistic nor pessimistic predictions [2], [4]. Optimism in setup/hold times can cause failure of the fabricated circuits, while pessimism leads to inferior performance.

Determining latch setup/hold times is intrinsically more computationally challenging than finding, *e.g.*, delays of combinatorial gates. The main reason is that whereas *linear* gate/interconnect models [5] often produce acceptable approximations for combinatorial gates, *memory effects* and *nonlinearities* are crucial for latch dynamics. Indeed, central to the setup/hold time problem is to determine the onset of metastability [6], a fundamentally nonlinear dynamical phenomenon peculiar to latches and bistable circuits. As a result, detailed transistor-level transient simulation [7] of latch circuits using accurate (and computationally expensive) device models is essential for the latch setup/hold problem. The prevalent technique for setup/hold time characterization is to investigate clock-to-q delay[1] for various trial setup/hold skews via a series of transient simulations embedded in a binary search process.

The computational expense of latch characterization in current industrial practice (especially in large microprocessor design houses [8], [9]) cannot be overstated. Setup/hold times need to characterized for every register/cell of every standard cell library, each typically containing hundreds or thousands of cells, for all process-voltage-temperature (PVT) corners. It is not uncommon, today, for months to be spent running simulations on large computer clusters simply for determining setup/hold times of latches and registers. Therefore, even relatively modest reductions in computation can have a disproportionately positive domino impact on microprocessor design cycles. As we describe in more detail later in the paper, the new setup/hold time characterization method presented here results in speed improvements of some $4 \sim 10\times$, depending on the accuracy desired.

The currently prevalent binary search method for finding setup/hold skews may be considered to be a guided trial-and-error approach for finding a data-vs-clock skew τ that leads to the onset of metastability. In this paper, we adapt ideas from mixed-signal/RF simulation [10]–[12] to propose a new technique for finding this skew *directly*, by expressing it as the solution to a scalar nonlinear equation $h(\tau) = 0$, where τ can be the setup or the hold time. Our formulation uses the nonlinear state-transition function [13], [14] of the differential equations describing the latch, and incorporates a threshold condition to detect onset of metastability.

We then solve the equation $h(\tau) = 0$ numerically using the well-known Newton-Raphson (NR) method [15]. Because NR has the property of quadratic convergence as it approaches the solution, it is able to "zoom in" on the correct solution (*i.e.*, increase accuracy) much more rapidly than binary search. As a result, it provides significant computational advantage over binary search in higher accuracy regimes, as we demonstrate in Section IV. We note that the computational advantage results in spite of the fact that NR on $h(\tau) = 0$ requires computation of the derivative $\frac{dh}{d\tau}$ – indeed, this extra gradient information is a core differentiator against binary search and is crucially responsible for NR's convergence. (Readers familiar with RF/mixed-signal simulation will note connections with shooting methods [10]–[12] and with transient sensitivity computation [16], [17] in Section III, which contains mathematical and algorithmic details of the technique).

We validate the new method in detail using two prototypical register designs: a transmission-gate based master-slave register and a C^{2}MOS edge-triggered register. Our experiments confirm that NR's gradient-directed quadratic convergence results in speed advantages (of $\sim 4 - 10\times$), especially at higher accuracies.

The remainder of the paper is organized as follows. We first provide some background on the latch setup/hold time problem in Section II. In Section III, we develop the new state-transition equation based formulation around general nonlinear differential algebraic equations (DAEs) of latches/registers to express the problem of finding setup/hold times as $h(\tau) = 0$; we then present a detailed description of our NR-based numerical algorithm to solve this equation. In Section IV, we validate the new method on prototypical register structures and provide detailed comparisons against binary search techniques.

II. TERMINOLOGY AND BRIEF BACKGROUND

Latches and edge-triggered registers are crucial and ubiquitous building blocks in all digital designs. Typically, each register has a clock line and a data line [18] . For reliable transfer of data through the register, the data line must be stable a certain amount of time (known as the *setup time*) prior to the active clock edge. Similarly, the data input line must be held stable for a certain amount of time (known as *hold time*) after the active clock edge to avoid problems in sampling the data. The active clock edge is defined as the transition

[1]See Section II for an explanation of setup/hold times, clock-to-q delay and other relevant concepts.

978-3-9810801-2-4/DATE07 © 2007 EDAA

edge of the clock at which data transfers occur; it is the *low to high/high to low* transition for a positive-triggered/negative-triggered register.

A concept often used in the context of latch setup/hold times is the *clock-to-q delay*, *i.e.*, the delay measured from the 50% transition of the active clock edge to the 50% transition of the output of the register. Setup skew is the delay from the 50% data transition edge to the 50% active clock transition edge; similarly, hold skew is the delay from the 50% active clock transition edge to the 50% data transition edge.

Clock-to-q delay is often plotted against setup/hold skew [6], [18]–[20]; this plot is generated using a series of transient simulation, one for each setup/hold skew. Generation of this plot is typically followed by the extraction of one special setup skew point on the plot, for which clock-to-q delay increases by a certain amount, *e.g.*, a typical number is 10%. This setup skew point is taken as the *setup time*. We will revisit this process of characterization in more detail in Section IV (see, for example, Fig. 3(a) for a graphical representation of the above procedure).

This manual characterization process, typically automated using binary search, provides an estimate of setup time but lacks accuracy since the determination of setup time is performed via interpolation from the plot drawn from data obtained from simulations. Refinement of setup time is typically performed by identifying an interval around the estimate of setup time from the plot and running many transient simulations in that interval in a sequence that follows binary search methodology to narrow down to the more exact value of setup time corresponding to the 10% increase in clock-to-q delay. A similar procedure applies for the characterization of hold time.

III. Nonlinear state-transition formulation for register setup and hold time determination

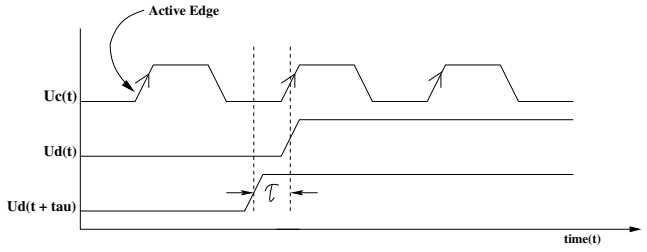


Fig. 1. Clock and Data Waveform.

In this section, we establish the formulation of finding setup time as an equation and then describe the numerical algorithm to solve it using Newton-Raphson. The methodology developed here will be generic in nature and can be applied for the characterization of hold time also.

A. Setup-time Formulation

We begin with a nonlinear circuit system that can be represented by the following vector differential algebraic equation [14].

$$\frac{d}{dt}\vec{q}(\vec{x})+\vec{f}(\vec{x})+\vec{b}u(t)=0. \tag{1}$$

In (1), for an order n system, $\vec{x}\in\mathbb{R}^n$ is the state vector of internal node voltages and branch currents; $\vec{q}\in\mathbb{R}^n$ and $\vec{f}\in\mathbb{R}^n$ are the charge/flux and the current terms respectively and $\vec{b}u(t)\in\mathbb{R}^n$ represents all the input source voltages.

A typical register consists of many transistors, a clock source and a single or multiple data source(s); it is therefore a nonlinear circuit system. Now, suppose $u_c(t)$ is a scalar waveform representing the clock of the register and $u_d(t+\tau)$ (*obtained by shifting $u_d(t)$ towards its left by τ time.*) represents the data waveform of the register. Here we make an assumption that $u_d(t)$ is the waveform with only one transition going from high to low or low to high and its transition edge coincides with the active edge of $u_c(t)$ (*refer* Fig. 1). The rationale behind this choice of $u_d(t)$ comes from the fact that we are trying to establish the methodology for the determination of setup time and therefore a data waveform like $u_d(t+\tau)$ is needed to monitor the output waveform of the register for different values of τ's, which is typically understood as the setup skew of the data waveform w.r.t the clock. The limiting value of τ, for which the rise time of the output waveform *i.e.* clock-to-output delay (t_{C-Q}) increases by a certain amount determines the setup time of the register. t_{C-Q} is the delay measurement from the 50% transition of the active clock edge to the 50% transition of the output waveform.

If we separate the input sources of a register into a clock input source and a data input source, the differential equation for it can be written as following based on the pattern of (1):

$$\frac{d}{dt}\vec{q}(\vec{x})+\vec{f}(\vec{x})+\vec{b}_c u_c(t)+\vec{b}_d u_d(t+\tau)=0. \tag{2}$$

In order to detect the limiting value of τ, we will monitor a single output waveform, given by $\vec{c}^T\vec{x}$. Here, $\vec{c}$ will typically be a unit vector which selects an output node. The typical behavior of the output waveform for different values of τ's is shown in Fig. 2(a). It can be noted from Fig. 2(a) that the clock-to-output delay increases as the setup skew decreases; output waveform even fails to complete the transition for much lower values of τ's as it happens for τ_3 and τ_4 shown in the figure. Apart from this, it is also true for any kind of register that if the setup skew is larger than a certain amount then the clock-to-output delay is independent of setup skew, this constant clock-to-output delay is the characteristic of any register and can be termed as the "*characteristic clock-to-output delay*".

We are interested in finding a value of τ, for which the t_{C-Q} of the output waveform increases by a certain amount compared to its characteristic clock-to-output delay, *i.e.* the output waveform corresponding to the τ is only able to reach at some value r, less than the 50% of its final value at time t_f, where t_f denotes the time at which the output waveform reaches to its 50% transition for large setup skews.

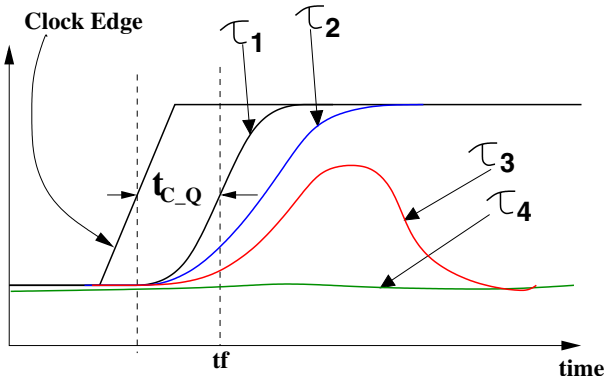
The above situation is also explained graphically in Fig. 2(b), in which two output waveforms corresponding to setup skews τ_1 and τ_2 are shown, having t_{C-Q_1} and t_{C-Q_2} clock-to-output delay respectively. Here we assume that t_{C-Q_1} is the characteristic clock-to-output delay and therefore independent of setup skew of value τ_1 and larger than that. As defined above, t_f is the time at which the output waveform, w.r.t. setup skew equal to or greater than τ_1, reaches to its 50% transition. The other output waveform w.r.t. τ_2 manages to reach at value r, much less than the 50% of its final value, at time t_f. Therefore, τ_2 qualifies as the setup time according to our specification. The next paragraph outlines how to find the value of τ_2 (*setup time*) given r and t_f.

Let $\vec{\phi}(t;\vec{x}_0,t_0=0,\tau)$ be the state-transition function of (2). We assume that the initial condition $\vec{x}_0=\vec{x}(t=t_0)$ is fixed at some value. Therefore, r, t_f and $\vec{x}_0$ are known and fixed quantities, the only unknown is τ and the condition we are seeking to satisfy is that the output is at value r at time t_f, *i.e.*, $\vec{c}^T\vec{x}(t_f)=r$. Writing this in terms of the state transition function, we obtain

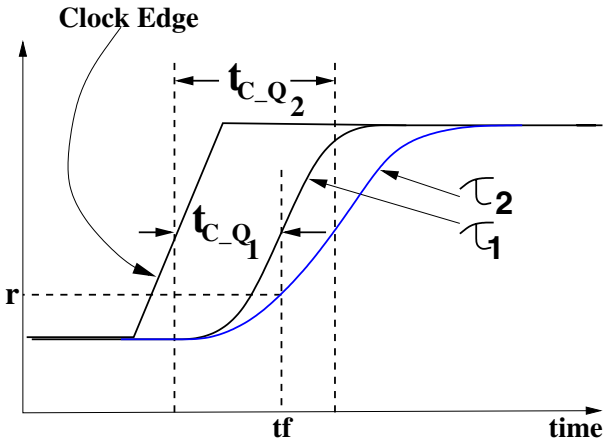
$$\begin{aligned}\vec{c}^T\vec{\phi}(t_f;\vec{x}_0,0,\tau)-r=0,\\ \text{or } \vec{c}^T\vec{\phi}_\tau(\tau)-r=0,\\ \text{where } \vec{\phi}_\tau(\tau)\equiv\vec{\phi}(t_f;\vec{x}_0,0,\tau).\end{aligned} \tag{3}$$

Hence, the nonlinear equation which we need to solve to get the optimal value of τ is

$$\boxed{h(\tau)\equiv\vec{c}^T\vec{\phi}_\tau(\tau)-r=0.} \tag{4}$$



(a) Output for $\tau_1 > \tau_2 > \tau_3 > \tau_4$.



(b) Clock to Q delay.

Fig. 2. Behavior of output waveform for different setup skews.

B. Newton-Raphson Methodology

In this paper, we propose to solve the nonlinear equation (4) via Newton-Raphson and this subsection, in particular, discusses all the details in relation to run NR on (4).

We first note that (4) is a scalar equation in a scalar unknown. In order to solve it via NR, we need to be able to do two things: 1) evaluate $h(\tau)$ given any τ and 2) evaluate $\frac{dh(\tau)}{d\tau}$ for any τ. Evaluation of $h(\tau)$ can be done by running a transient simulation with the given τ and then evaluating (4). To compute $\frac{dh(\tau)}{d\tau}$, we need to evaluate $\frac{d}{d\tau}\vec{\phi}(t_f;\vec{x}_0,0,\tau)$. We next develop the procedure to do this.

First, we write out (2) with all dependencies on τ shown explicitly for clarity:

$$\frac{d}{dt}\vec{q}(\vec{x}(t,\tau))+\vec{f}(\vec{x}(t,\tau))+\vec{b}_c u_c(t)+\vec{b}_d u_d(t+\tau)=0. \tag{5}$$

Next, noting that $\frac{d\vec{\phi}}{d\tau}$ is simply $\frac{d\vec{x}(t,\tau)}{d\tau}$, we differentiate the entire equation with respect to τ, interchanging the order of differentiation w.r.t. t and τ in the first term :

$$\begin{aligned}0&=\frac{d}{d\tau}\left[\frac{d}{dt}\vec{q}(\vec{x}(t,\tau))+\vec{f}(\vec{x}(t,\tau))+\vec{b}_c u_c(t)+\vec{b}_d u_d(t+\tau)\right]\\&=\frac{d}{dt}\left[\frac{d}{d\tau}\left[\vec{q}(\vec{x}(t,\tau))\right]\right]+\frac{d}{d\tau}\left[\vec{f}(\vec{x}(t,\tau))\right]+\vec{b}_d u_d'(t+\tau)\\&=\frac{d}{dt}\left[\frac{d\vec{q}(t,\tau)}{d\vec{x}}\frac{d\vec{x}}{d\tau}\right]+\frac{d\vec{f}(t,\tau)}{d\vec{x}}\frac{d\vec{x}}{d\tau}+\vec{b}_d u_d'(t+\tau).\end{aligned} \tag{6}$$

Since we want to evaluate $\frac{dh(\tau)}{d\tau}$ at any given value of τ, *e.g.* at τ^*, we can define the following terms for the ease of understanding.

$$\begin{aligned}C^\dagger(t)&=\left.\frac{d\vec{q}(t,\tau)}{d\vec{x}}\right|_{\tau=\tau^*},\\G^\dagger(t)&=\left.\frac{d\vec{f}(t,\tau)}{d\vec{x}}\right|_{\tau=\tau^*},\\\text{and}\quad \vec{m}^\dagger(t)&=\left.\frac{d\vec{x}(t,\tau)}{d\tau}\right|_{\tau=\tau^*}.\end{aligned} \tag{7}$$

Rewriting (6) for $\tau=\tau^*$, we get:

$$\frac{d}{dt}\left(C^\dagger(t)\vec{m}^\dagger(t)\right)+G^\dagger(t)\vec{m}^\dagger(t)+\vec{b}_d u_d'(t+\tau^*)=0 \tag{8}$$

Solution of (8) can easily be obtained using any integration method, *e.g.* BE, TRAP *etc.* [13], [14]. We write the solution of (8) using BE as

$$\begin{aligned}&\frac{C^\dagger(t_i)\vec{m}^\dagger(t_i)-C^\dagger(t_{i-1})\vec{m}^\dagger(t_{i-1})}{t_i-t_{i-1}}\\&\quad+G^\dagger(t_i)\vec{m}^\dagger(t_i)+\vec{b}_d u_d'(t_i+\tau^*)=0,\end{aligned} \tag{9}$$

which can be simplified as

$$\vec{m}_i^\dagger=\left(\frac{C_i^\dagger}{\Delta t}+G_i^\dagger\right)^{-1}\left(\frac{C_{i-1}^\dagger}{\Delta t}\vec{m}_{i-1}^\dagger-\vec{b}_d u_d'(t_i+\tau^*)\right). \tag{10}$$

Subscript i denotes to the fact that the corresponding quantity has been evaluated at $t=t_i$; $\Delta t=t_i-t_{i-1}$ is the time step used in the integration.

We need to know the value of $\vec{m}_0^\dagger$ to start the integration process and it turns out that we can safely take $\vec{m}_0^\dagger$ to be equal to $\vec{0}$. The reason for this choice is that $\vec{x}_0=\vec{x}(t=t_0)$ will not change for any particular value of τ and thus enabling $\vec{m}_0$ to take the value $\vec{0}$.

Evaluating (10) from $t=t_1$ to $t=t_f$ (*i.e.* $i\in 1,2,\ldots,f$) will give $\vec{m}_f^\dagger=\left.\frac{d\vec{\phi}_\tau}{d\tau}\right|_{t=t_f,\tau=\tau^*}$.

And finally we can obtain the scalar

$$\left.\frac{dh(\tau)}{d\tau}\right|_{t=t_f,\tau=\tau^*}=\vec{c}^T\left.\frac{d\vec{\phi}_\tau}{d\tau}\right|_{t=t_f,\tau=\tau^*}. \tag{11}$$

Next subsection presents the algorithm for running NR on (4).

C. Numerical Algorithm

In this subsection, we'll outline the procedure of finding the optimal value of $\tau=\tau_s$, referred as the setup time of register, such that it satisfies (4). (4) is rewritten below for clarity.

$$h(\tau)=\vec{c}^T\vec{\phi}_\tau(\tau)-r=0. \tag{12}$$

The above written equation has to be evaluated at time $t=t_f$.

The algorithm for finding τ_s is as follows:

1) Initialize τ, $\vec{x}$ and $\vec{m}$.
   a) $\tau=\tau_0$.
      τ_0 can be chosen as any positive value; a good choice will always be less than the time period of the clock. A better guess of τ_0 will approximate to some previously known setup time of the similar kind of registers.
   b) $\vec{x}(t=0,\tau)=\vec{x}_0(\tau)$.
      $\vec{x}_0(\tau)$ can be made same for all values of τ *e.g.* equal to $\vec{x}_0^*$; where $\vec{x}_0^*$ can assume any arbitrary value.
      A good choice of $\vec{x}_0(\tau)$ will be the DC operating point for that particular value of τ; here $\vec{x}_0(\tau)$ will differ for different values of τ's.

c) $\vec{m}(t=0,\tau)=\vec{m}_0(\tau)$.
As explained previously, $\vec{m}_0(\tau)$ will be initialized to $\vec{0}$ because $\vec{x}_0$ doesn't change for a particular value of τ.

2) Start the Newton-Raphson procedure.

For an iteration index j:

a) Divide $t=0$ to t_f in N points: t_0, t_1, ...,t_{N-1}. For each $i \in \{0,\ldots,N-1\}$, compute the following:
τ_j is the value of τ being used for the iteration index j.

i) Compute $\vec{x}_{ji}$ using (2) (reproduced below).
Here, $\vec{x}_{ji}$ denotes to the fact that the quantity $\vec{x}$ is being evaluated at time t_i for the iteration index j. This terminology will hold for other quantities too.

$$\frac{d}{dt}\vec{q}(\vec{x})+\vec{f}(\vec{x})+\vec{b}_c u_c(t)+\vec{b}_d u_d(t+\tau)=0. \quad (13)$$

(13) can be solved using any integration method like BE, TRAP *etc.* [13], [14].

ii) After having obtained $\vec{x}_{ji}$'s, compute the following:

$$C_{ji}=\left.\frac{d\vec{q}(\vec{x})}{d\vec{x}}\right|_{\vec{x}=\vec{x}_{ji}} \quad \text{and} \quad G_{ji}=\left.\frac{d\vec{f}(\vec{x})}{d\vec{x}}\right|_{\vec{x}=\vec{x}_{ji}}. \quad (14)$$

iii) Compute $\vec{m}_{ji}$ using (10) as follows.

$$\vec{m}_{ji}=\left(\frac{C_{ji}}{t_i-t_{i-1}}+G_{ji}\right)^{-1}\left(\frac{C_{j(i-1)}}{t_i-t_{i-1}}\vec{m}_{j(i-1)} -\vec{b}_d u'_d(t_i+\tau_j)\right). \quad (15)$$

We have now obtained $\vec{x}_{j(N-1)}$ and $\vec{m}_{j(N-1)}$.

b) Calculate $h(\tau_j)$ defined in (4) as follows.

$$h(\tau_j)=\vec{c}^T\vec{x}_{j(N-1)}-r. \quad (16)$$

c) Check convergence of NR using reltol and abstol [21].
If NR has converged, then we have obtained the optimal value of τ as τ_j. Stop Here.
Otherwise calculate $\frac{dh(\tau)}{d\tau}$ defined in (11) as follows:

$$\left.\frac{dh(\tau)}{d\tau}\right|_{\tau=\tau_j}=\vec{c}^T\vec{m}_{j(N-1)}. \quad (17)$$

d) Calculate τ_{j+1} and increment j.

$$\tau_{j+1}=\tau_j-\frac{h(\tau_j)}{\left.\frac{dh(\tau)}{d\tau}\right|_{\tau=\tau_j}}, \quad \text{and} \quad j=j+1. \quad (18)$$

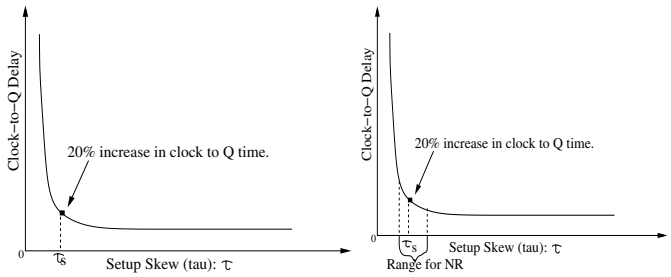
Go to step (a) for the next iteration of NR.

The above described procedure will be applied for the computation of τ_s in the next section.

IV. Application of NR Methodology and Results

In this section, we'll first review the existing method for the setup/hold time characterization. We'll then show how Newton-Raphson is useful for the fast and more precise computation of setup time. At the end of this section we'll compare the results obtained from the NR method against the binary search method to emphasize the usefulness of Newton-Raphson based methodology for latch/register characterization.

In one of the current existing methodology for the setup time characterization: clock-to-output delay is first plotted against setup skew (*a typical plot of t_{C-Q} vs setup skew is shown in Fig. 3(a)*), then the setup skew for which clock-to-q delay increases by a certain amount *e.g.* by 20% is measured and considered as the setup time. We must note here that the clock-to-q delay changes very rapidly



(a) Setup time characterization.

(b) Region for Newton Raphson for setup time characterization.

Fig. 3. Setup skew vs clock-to-output delay

near to the setup time, making it very sensitive to small setup skew changes in the neighborhood of setup time; therefore, to capture the setup time with high accuracy, it requires many simulations to run in the vicinity of setup time τ_s.

If we have identified the region around the setup time, *i.e.* we have found two nearby setup skews such that it contains τ_s, then we can refine the search for the setup time (using the criteria of 20% increase in t_{C-Q}) applying the binary search method in that interval. One such interval is shown in Fig. 3(b) which contains the setup time point τ_s.

We propose here to apply the Newton-Raphson method in the above identified setup skew interval to quickly narrow down to the setup time. Since, NR has the quadratic convergence near to the solution, we expect to get to the result much faster as compared to the binary search method which has the linear convergence.

A. Transmission gate based master-slave register

We have chosen the simple transmission gate master-slave positive-edge triggered register for the verification purpose. The transmission gate master-slave register is shown in Fig. 4.

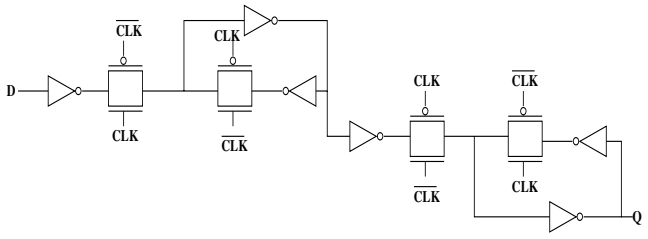


Fig. 4. Transmission Gate Based Positive-edge triggered master-slave register.

The clock waveform($u_c(t)$) used in the register is chosen such that it makes the transition between 0V and 2.5V with a period of 10ns. It has the rise/fall time of 0.1ns and has initial delay of 1ns. Therefore, the active clock transition edge starts at 1ns, 11ns, 21ns...so on. The data waveform($u_d(t)$) makes a low to high transition from 0V to 2.5V with a rise time of 0.1ns at 11ns coinciding its transition edge with the active clock edge of the clock. *$u_c(t)$ and $u_d(t)$ are of the same type as shown in Fig. 1.*

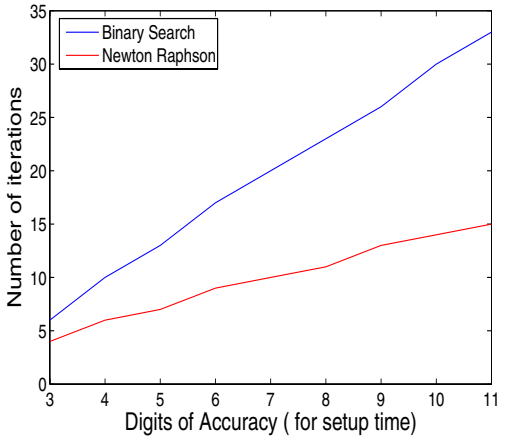
For a large setup skew(τ), when we can assume that the data is being latched reliably and clock-to-q delay is independent of τ, the output waveform rises to its 50% value *i.e.* 1.25V at 11.625ns. If the setup time corresponds to that value of τ, for which the output waveform rises only to 1V(20% less than 1.25V) at 11.625ns, then we set $r=1V$ and $t_f=11.625ns$ in (4) and call the corresponding setup skew as τ_s, which is the setup time of the register in this case.

After having set the value for r and t_f, we need a suitable initial guess of τ to start the Newton Raphson method. Initial guess of τ

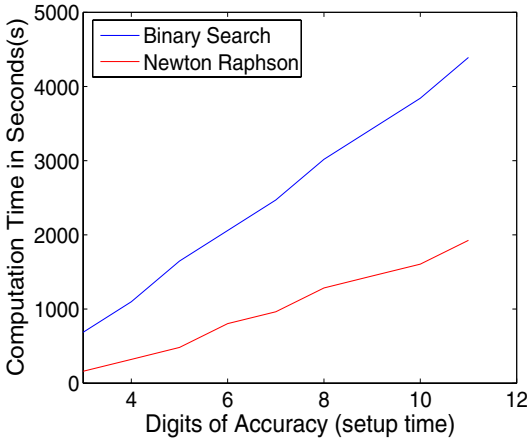
proves to be critical for the convergence of NR and should be chosen near to the solution within the convergence range of NR.

To find a good initial guess of τ: We first start with a setup skew interval $[\tau_L, \tau_R]$, where register latches the data properly for τ_L and fails to latch for τ_R. Therefore, this interval will contain τ_s. We then narrow down the setup skew interval surrounding τ_s (as shown in Fig. 3(b)) using binary search method untill the interval length falls in the convergence range of NR.

Convergence range of NR varies with types of registers, rise/fall time of data and clock waveforms *etc.*. Sensitivity of output waveform to the changes in setup skew, affects considerably the convergence range of NR. Convergence range will be small for high sensitive output and large for less sensitive output.



(a) Number of Iterations vs Digits of Accuracy.



(b) Computation Time vs Digits of Accuracy.

Fig. 5. Digits of accuracy: Newton Raphson vs binary search method.

For the circuit in Fig. 4, Newton-Raphson has very small convergence range, approximating to 5-8ps. Therefore, we narrowed down the setup skew interval to 5 ps before applying NR on it. We can then take the initial guess of τ either as τ_R or τ_L to start the NR method. After having the values for τ_0(*initial guess*), r and t_f, we apply NR(*following the algorithm described in previous section*) to reach at the solution τ_s. The solution τ_s thus obtained equals $1.838398001980504e-10$ seconds; this solution has 10 digits of accuracy and it took only 12 iterations of NR. In contrast to this, Binary Search method took 32 iterations to give the solution of τ_s with 10 digits of accuracy. Obviously, NR excels when high accuracy is needed. We have plotted a graph showing number of iterations vs digits of accuracy to compare the Binary Search and the NR methodologies.

The plots in Fig. 5(a) shows how many number of iterations of NR is needed as compared to Binary Search to add one extra bit of accuracy in the value of τ_s. It is clear from the graph that NR manages to increase bits of accuracy in much less number of iterations than the Binary Search method, therefore it reaches very fast to the more accurate value of setup time.

Though, the number of computations required in one iteration of NR is more than the number of computations needed in one iteration of Binary Method, NR becomes more and more efficient as the digits of accuracy needed in the setup time increases. In fact, the computation time per iteration for NR increases mainly due to the evaluation of equation (15) in each iteration. Since equation (15) involves the inverse computation of the matrix $(\frac{C_{ji}}{t_i - t_{i-1}} + G_{ji})$, it looks expensive in terms of time consumption. But, effectively, the inverse computation of this matrix is of the order $O(\ n^a)$ $(a \in [1,2])$ because of the sparsity. For the circuit shown in Fig. 4, NR took 160.5 seconds per iteration while Binary Search took 137.2 seconds. The plots shown in Fig. 5(b) gives the comparison of computation time for NR and Binary Search method. Clearly, NR consumes less time than Binary Search for each additional bit of accuracy in spite of its larger computation time per iteration. The vertical difference between the plots shown in Fig. 5(b) gets bigger and bigger, validating our assumption that NR will perform better with each additional bit of accuracy.

B. C^2MOS Positive-edge triggered master-slave register.

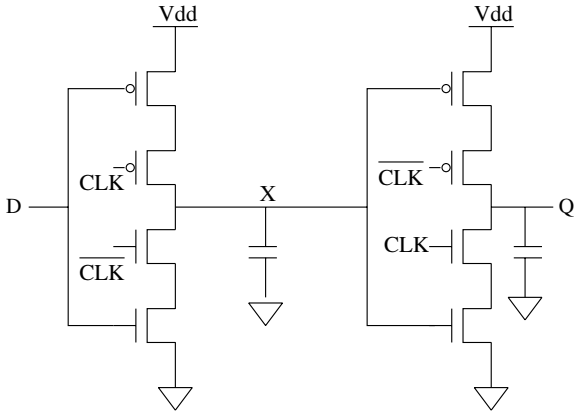
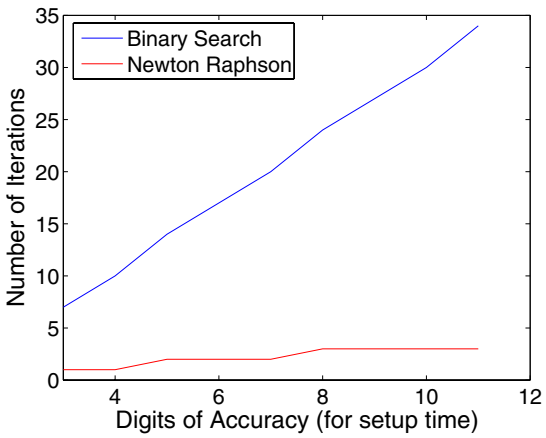


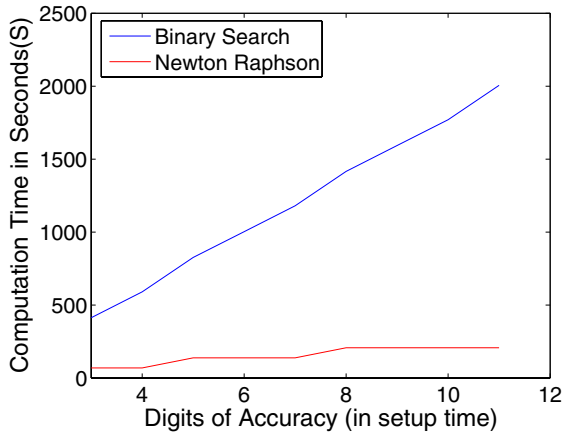
Fig. 6. C^2MOS Positive-edge triggered master-slave register.

To further validate the use of NR as a fast characterization method, we extracted the setup time information for the circuit shown in Fig. 6 using NR. We have taken the same clock and data waveform ($u_c(t)$ and $u_d(t)$) as used for the previous register. The output of this register rises to its 50% value *i.e.*, 1.25V at 11.171ns for a large setup skew. As we did in the previous example, here also we assume that the setup time corresponds to that value of τ, for which the output rises only to 1V(20% less than 1.25V) at 11.171ns, thus we set $r = 1V$ and $t_f = 11.171ns$ in (4).

The convergence range of NR for this register is found to be approximately 120-150ps, indicating that the output of this register is less sensitive to the changes in setup skews as compared to the output of the transmission-gate based master-slave register. Therefore, initial value of τ for this register can be chosen anywhere within the setup skew interval of length 120-150ps containing τ_s (*setup time*). After having the numerical values for r, t_f and τ_0, we apply NR and get the setup time of the register 2.514017726429020e-10 seconds, which is accurate upto 10 digits. The plots shown in Fig. 7(a) and Fig. 7(b)



(a) Number of Iterations vs Digits of Accuracy.



(b) Computation Time vs Digits of Accuracy.

Fig. 7. Digits of accuracy: Newton Raphson vs binary search method.

shows the number of iterations and computation time against digits of accuracy respectively. Once again, it is evident from the plots that NR becomes more and more efficient than Binary Search with each additional bit of accuracy.

Usefulness of NR will also be more prominent if we already have some idea of the setup skew interval surrounding the τ_s, then it only requires to run few iterations of NR to extract the setup time of high accuracy. It turns out that we in fact, have a fair idea of such setup skew interval from the previous characterization of standard cell library containing similar registers. Hence, NR can be less time consuming and thus very useful as it features more accuracy in less number of iterations, in the characterization of setup/hold time.

Conclusions and Future Directions

We have presented a new method that directly solves for latch setup and hold times by expressing them as solutions to a scalar nonlinear equation, and solving this equation exploiting the quadratic convergence properties of Newton-Raphson. We have validated the new technique on two different registers and compared them against binary search, confirming speedups in the range of $4-10\times$.

Current/future work is focused on working together with Intel to implement and evaluate the proposed technique in an industrial timing closure flow. We are also developing extensions of the method that take advantage of NR's quadratic convergence for multivariate (*i.e.*, vector) unknowns, to dramatically reduce setup/hold time calculations for simultaneously finding both setup and hold times, as well as exploring the more complex setup/hold skew landscape for multi-input registers and groups of registers. We anticipate that our technique will rapidly gain adoption in industry to become the standard for latch/register setup/hold characterization for cell libraries over process-voltage-temperature variations.

Acknowledgments

We thank Chirayu Amin, who explained the latch setup/hold problem in clear and minimalist fashion during a visit to Intel's SCL in Portland, Oregon; and Chandramouli Kashyap for subsequent discussions. We also acknowledge Sani Nassif, who several years ago had mentioned this problem in the context of devising specialized fast latch macromodels.

References

[1] W. Roethig. Library characterization and modeling for 130 nm and 90 nm soc design. *Proceedings of the IEEE International SOC Conference*, pages 383–386, September 2003.

[2] Emre Salman, Ali Dasdan, Feroze Taraporevala, Kayhan Kucukcakar, and Eby G. Friedman. Pessimism reduction in static timing analysis using interdependent setup and hold times. *Proceedings of the 7th International Symposium on Quality Electronic Design*, page 6, March 2006.

[3] D. Patel. Charms: Characterization and modeling system for accurate delay prediction of asic designs. *Proceedings of the IEEE Custom Integrated Circuits Conference*, pages 9.5.1–9.5.6, May 1990.

[4] R. W. Phelps. Advanced library characterization for high-performance asic. *Proceedings of the IEEE Custom International ASIC conference*, pages 15–3.1 – 15–3.4, September 1991.

[5] J. Sridharan and T. Chen. Gate delay modeling with multiple input switching for static(statistical) timing analysis. *Electronic Device Letters, IEEE*, 22(4):309–311, July 2001.

[6] M. Shoji. *Theory of CMOS Digital Circuits and Circuits Failures*. Princeton University Press, 1992.

[7] L.W. Nagel. *SPICE2: a computer program to simulate semiconductor circuits*. PhD thesis, EECS Dept., Univ. Calif. Berkeley, Elec. Res. Lab., 1975. Memorandum no. ERL-M520.

[8] S. Nassif. Personal communication, 2005.

[9] C. Amin and C. Kashyap. Personal communications, 2006.

[10] T.J. Aprille and T.N. Trick. Steady-state analysis of nonlinear circuits with periodic inputs. *Proc. IEEE*, 60(1):108–114, January 1972.

[11] S. Skelboe. Computation of the periodic steady-state response of nonlinear networks by extrapolation methods. *IEEE Trans. Ckts. Syst.*, CAS-27(3):161–175, March 1980.

[12] R. Telichevesky, K. Kundert, and J. White. Efficient AC and noise analysis of two-tone RF circuits. In *Proc. IEEE DAC*, pages 292–297, 1996.

[13] C.W. Gear. *Numerical initial value problems in ordinary differential equations*. Prentice-Hall series in automatic computation. Prentice-Hall, Englewood Cliffs, N.J., 1971.

[14] L. O. Chua and P. M. Lin. *Computer-Aided Analysis of Electronic Circuits: Algorithms and Computation Technigues*. Prentice-Hall, Englewood Cliffs, NJ, 1975.

[15] W.H. Press, S.A. Teukolsky, W.T. Vetterling, and B.P. Flannery. *Numerical Recipes – The Art of Scientific Computing*. Cambridge University Press, 1989.

[16] U. Choudhury. Sensitivity Computation in SPICE3. Master's thesis, EECS Dept., Univ. Calif. Berkeley, Elec. Res. Lab., December 1988.

[17] D. Hocevar, P. Yang, T. Trick, and B. Epler. Transient Sensitivity Computation for MOSFET Circuits. *IEEE Trans. CAD*, 4(4):609–620, October 1985.

[18] Jan M. Rabaey. *Digital Integrated Circuits*. Prentice-Hall, 2004.

[19] V. Stojanovic and V.G. Oklabdzija. Comparative analysis of master-slave latches and flip-flops for high-performance and low-power systems. *IEEE Journal of Solid-Sate Circuits*, 34(4):536–548, April 1999.

[20] N. Weste and D. Harris. *CMOS VLSI Design*. Addison Wesley, 2004.

[21] Thomas L. Quarles. *SPICE 3B.1 User's Guide*. University of California, Berkeley, EECS Industrial Liaison Program, University of California, Berkeley California, 94720, April 1987.