Computing Synchronizer Failure Probabilities

Suwen Yang Mark Greenstreet
Department of Computer Science, University of British Columbia

***Abstract*— System-on-Chip designs often have a large number of timing domains. Communication between these domains requires synchronization, and the failure probabilities of these synchronizers must be characterized accurately to ensure the robustness of the complete system. We present a novel approach for determining the failure probabilities of synchronizer circuits. We use numerical intergration to perform large-signal analysis that accounts for the non-linear behaviour of real synchronizer circuits. We complement this with small-signal techniques to characterize behaviours near the metastable equilibrium. This combination overcomes the limitations of traditional techniques: the large-signal analysis accounts for the transfer of metastable behaviour between synchronizer stages; and the small-signal techniques overcome the limitations of numerical accuracy inherent in pure simulation approaches. Our approach is fully automated, is suitable for integration into circuit simulation tools such as SPICE, and enables accurate characterization of extremely small failure probabilities.**

1. Introduction

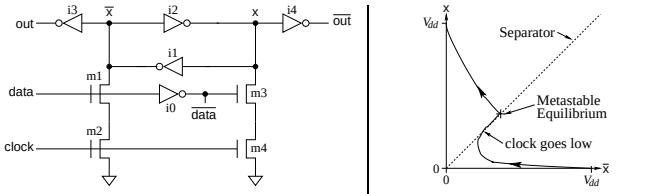
Large integrated circuit designs are being divided into an increasing number of separate time domains. There are several motivations for this trend. First, distributing a global clock across a large chip with low skew is difficult, power intensive, and often unnecessary. Second, the use of independently designed IP ("Intellectual Property") blocks results in modules with different performance requirements and clock speeds. Third, power management techniques such as dynamic voltage scaling introduce even more combinations of clock frequencies into a design. Finally, interfaces to networks, graphics and storage devices, memory, and other processors can each require a separate clock. This proliferation of timing domains leads to a growing number of on-chip synchronizers at their interfaces. While it is impossible to build a perfect synchronizer [1], the probability of failure drops roughly exponentially with the time allotted for synchronization [2], [3]. Calculating these failure probabilities accurately is an essential part of multi-timed design: underestimating the failure probability leads to an unreliable design; conversely, overestimates lead to excessive communication latency and lower system-level performance. This paper presents an efficient and accurate method for determining synchronizer failure probabilities.

The traditional analysis of synchronization failure uses a small-signal linear approximation of the dynamics of the bistable circuit near its metastable equilibrium [4, chap. 7.5]. Such linearization neglects the large signal behaviours that occur as the synchronizer first approaches its metastable equilibrium and in the coupling between stages in multi-stage synchronizers. Thus, the small-signal approach is useful for gaining an intuitive understanding of synchronizer operation, it does not provide accurate estimates of failure probabilities.

Simulation based on numerical integration is the most prevalent way to analyse non-linear circuits. However, the acceptable failure probabilities for real synchronizers are extremely small, and the difference between an input that causes a failure and one that leads to correct resolution is less than the resolution of double-precision floating point numbers. Furthermore, the dynamics of synchronizer circuits lead to numerical instability in the integration routines so that the accuracy of the simulator is much less than the floating point resolution. Thus, simulation is an essential tool for circuit design, but simulations cannot establish the very low failure probabilities required for real designs.

Instead of using simulation or analysis, it is also possible to experimentally measure the failure probability of a synchronizer after it has been fabricated. In [5], Kinniment *et al* describe how they combined a clever experimental set-up with careful statistical accounting to measure the failure probabilities of a 74F5074 flip-flop down to 10^{-12} (assuming a 100 MHz clock, and uniformly distributed input arrival times), corresponding to a MTBF (Mean Time Between Failure) of about three hours. They showed that due to the non-linear effects in the coupling between the master and slave stages of the flip-flop, the actual MTBF was less than that predicted by the simple, linear model by more than a factor of 20000. Similar measurements and results are presented in [6], [7]. The physical data that can be provided by these kinds of experiments is valuable for validating any computational approach for analysing metastable behaviour. However, such physical measurements can only be performed after the synchronizer has been fabricated, and the it must be possible to observe the synchronizer from off-chip. Designers need to be able to determine failure probabilities and evaluate design tradeoffs before their circuits are fabricated. Thus, we need a computational approach for analysing these circuits.

This paper presents a novel method for computing the failure probabilities of synchronizers. We combine small-signal, linear analysis with large-signal, non-linear simulation in a way that exploits the strengths of each to allow us to accurately compute failure probabilities that are much smaller than the floating point precision. Section 2 presents that the dynamical systems perspective that we use for analysing metastability and describes the limitations of traditional methods based on small-signal analysis or numerical integration. Section 3 shows how we combine small-signal analysis with numerical integration to obtain a robust, fully-automated method for computing failure probabilities. In section 4 we illustrate our approach by computing the failure probabilities for several synchronizer



A latch circuit A Metastable Trajectory

Fig. 1. A Simple Latch and Its Metastable Behaviour

circuits.

2. Synchronization

Consider the transparent latch shown in the left part of Figure 1. Metastability can occur if the data input changes at roughly the same time as the clock input makes a high-to-low transition. For example, the right half of Figure 1 illustrates how the latch can function if the data input makes a rising transition as the clock falls. In response to the high value on data, $\overline{\mathsf{x}}$ starts to fall, which causes x to rise with some lag. Metastability occurs if x and $\overline{\mathsf{x}}$ have roughly the same voltage when the clock input falls.

We now consider the metastable latch as a dynamical system. For simplicity, assume that inverters i1 and i2 are identical. Let V_{ms} be the voltage at which the inputs and outputs of these inverters are equal in equilibrium. This balance point is the *metastable equilibrium* of the latch.

When the trajectory is sufficiently close to the metastable point, it can be accurately approximated using a linear model for the circuit dynamics. Let y be a vector with an element for each non-input node of the circuit, and let in be the vector of inputs. An ODE (ordinary differential equation) model for circuit is given by

$$\dot{y} \quad = \quad f(y, in), \tag{1}$$

where $\dot{y}$ is the time derivative of y, and f is the derivative function. Let y_{ms} denote the metastable equilibrium for the synchronizer. For y close to y_{ms} we get:

$$\dot{y} \quad \approx \quad A_{(y_{ms},in)} \cdot (y - y_{ms}) + \dot{y}_{ms} \tag{2}$$

where the $A_{(y_{ms},in)}$ matrix is the Jacobian of f at (y_{ms}, in). The traditional textbook analysis [4, chap. 7.5] assumes that in is constant during the time that metastability is a concern. For typical synchronizer circuits, this is a reasonable approximation as long as the clock signal is constant. We then get

$$y(t) \quad \approx \quad y_{ms} + e^{t \cdot Ams} \cdot (y(t_0) - y_{ms}) \tag{3}$$

If y is n-dimensional, then the matrix $A_{(y_{ms},in)}$ has $n-1$ negative eigenvalues and one positive one. The eigenvectors for the negative eigenvalues span the $n-1$ dimensional space corresponding to the separator near y_{ms}; these decay exponentially with time, bringing the trajectory closer to the metastable equilibrium. The eigenvector for the positive eigenvalue corresponds to the separation of trajectory from the separator – this is the component that grows exponentially with time, eventually bringing the synchronizer to a well-defined logical state. Letting λ_+ denote the value of the positive eigenvalue of $A_{(y_{ms},in)}$. If the state of the synchronizer at time t_0 is uniformly distributed in a small region around y_{ms}, then the probability that the synchronizer has not resolved by time t decreases as $e^{\lambda_+ t}$.

As noted earlier, the simple linear model from equation 3 fails to capture the behaviour of real synchronizers. There are two main reasons for this. First, when a latch goes opaque, its internal state may be near the separator between the stable basins of attraction for the latch but still a significant distance from the metastable equilibrium. The right side of figure 1 illustrates this where the trajectory is along the separator but below the metastable equilibrium when the clock goes low. Second, the assumption that the clock input to the synchronizer is constant while the synchronizer is metastable does not hold for a multi-stage synchronizer. Such a synchronizer attempts to resolve metastability over several clock periods; with each successive edge of the clock, the metastable behaviour effectively moves from one stage of the synchronizer to the next. These large signal swing activities violate the small signal assumptions of linear analysis.

Numerical integration provides an alternative to small signal analysis. For example, HSPICE provides a bisection command to search for metastable equilibria. With bisection, a user can choose two input transition times such that the latch settles high for the first and low for the second. Then, the bisection routine searches for the input transition time that causes the circuit to remain in an unresolved, metastable condition for a prolonged time.

Bisection provides a method for calculating MTBF. Let's say that a design allows time t_s for the output of the synchronizer to settle. Let $t_{early}(t_s)$ and $t_{late}(t_s)$ be the earliest and latest transition times respectively for the input that cause the output to take at least t_s to settle. Define $\Delta_{t_{in}} = t_{late}(t_s) - t_{early}(t_s)$. In words, $\Delta_{t_{in}}(t_s)$ is the width (in seconds) of the window of input events that cause the synchronizer to fail when given time t_s to settle. If the clock frequency is f_c, the rate of input transitions is f_d, and the times of input events are uncorrelated with the clock, we get

$$\text{MTBF}(t_s) \quad = \quad (f_c f_d \Delta_{t_{in}}(t_s))^{-1}. \tag{4}$$

Given a value for t_s, we can use simulation and bisection to compute $t_{early}(t_s)$ and $t_{late}(t_s)$. Using SPICE or similar models, the non-linearities of the circuit are taken into account, overcoming the limitations of analysis based on small-signal models. However, designers typically need very small MTBFs, often specified in the millions of years or more. If f_d is 1GHz, then to achieve a MTBF of one million years, $\Delta_{t_{in}}$ must be less than $10^{-22}(1/f_c)$; this is beyond the numerical resolution of a simulator using double-precision numbers. Furthermore, the ODE for the synchronizer is numerically unstable near a metastable equilibrium to to the positive eigenvalue, λ_+ of the Jacobian. Thus, the accuracy of a numerical integrator will be much less than the numerical resolution of the machine. In the next section, we show how small-signal, linear analysis can be

combined with large-signal numerical integration to compute failure probabilities accurately for very large MTBFs.

3. Combining Large- and Small-Signal Analysis

Our method for computing synchronizer failure probabilities is based on two observations. First, the size of the window of input events for which the synchronizer fails, $\Delta_{t_{in}}$ decreases exponentially with the amount of time that the synchronizer has to settle, t_s. Thus, the trajectories corresponding to input events at $t_{early}(t_s)$ and $t_{late}(t_s)$ will be extremely close to each other during the initial part of the simulation, only diverging from each other when the metastable condition is finally resolved. Second, MTBF depends on the *difference* between $t_{early}(t_s)$ and $t_{late}(t_s)$, the exact values of $t_{early}(t_s)$ and $t_{late}(t_s)$ are not critical. Our approach calculates this difference, $\Delta_{t_{in}}$, directly, and avoids the "small difference of large numbers" problem associated with existing simulation based techniques for analysing metastability.

3.1. Restarting Bisection

Our approach extends the traditional bisection approach. For simplicity, we describe our algorithm for a synchronizer whose data input makes a low-to-high transition. The analysis for transitions in the other direction is equivalent. Initially, we start with times t_{H_0} and t_{L_0} such that the synchronizer settles high when the input transition for data occurs at time t_{H_0} and settles low when the transition is at time t_{L_0}. The times t_{H_0} and t_{L_0} are provided by the designer; they may be widely separated; so finding such times is not difficult. Using bisection, we find a small interval, $[t_{H_1}, t_{L_1}]$ such that the synchronizer settles high if the input transition occurs at time t_{H_1} and low if it occurs at t_{L_1}. We keep the gap between t_{H_1} and t_{L_1} large enough that the trajectories for these two conditions are clearly distinguished by the numerical integrator.

Rather than further bisecting the interval of transition times for data, we *restart* the simulation at a later time. Let $V_1(t_{in}, t)$ give the state vector (voltage on each node of the circuit) at time t when the input changes at time t_{in}. We find a time, t_1, such that for $t_{H_1} < t_{in} < t_{L_1}$ and $0 \le t \le t_1$

$$V_1(t_{in}, t) \approx \frac{t_{L_1} - t_{in}}{t_{L_1} - t_{H_1}} V_1(t_{H_1}, t) + \frac{t_{in} - t_{H_1}}{t_{L_1} - t_{H_1}} V_1(t_{L_1}, t) \quad (5)$$

The main considerations for choosing t_1 are that t_1 must be small enough for the accuracy of the linear approximation to be acceptable and large enough for the analysis to make progress. Section 3.3 describes the selection of t_1 in greater detail and shows that these conditions are easily satisfied in practice. In the following, let $V_{H_1} = V(t_{H_1}, t_1)$, and $V_{L_1} = V(t_{L_1})$.

The points $V_1(t_{H_1}, t_1)$ and $V_1(t_{L_1}, t_1)$ are the endpoints of an interval that we can use for further bisection. Let $\alpha_2 \in [0, 1]$ be the bisection parameter; set the initial voltage state of the circuit to $(1-\alpha_2)V_{H_1} + \alpha_2 V_{L_1}$ and the initial time to t_1; model the data input with a transition at time $(1-\alpha_2)t_{H_1} + \alpha_2 t_{L_1}$.

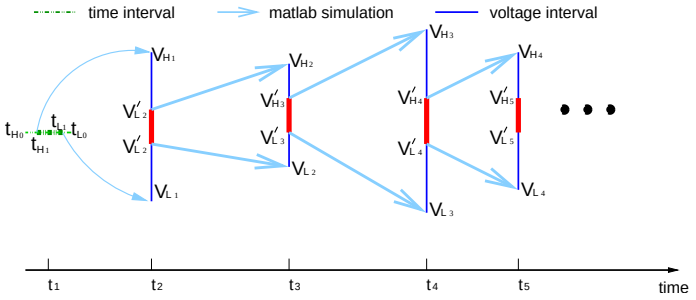


Fig. 2. The Evolution between intervals

We bisect on α_2 to find a small interval, $[\alpha_{H_2}, \alpha_{L_2}]$, such that the synchronizer settles high when simulated from the initial state for α_{H_2} and low when simulated from the initial state for α_{L_2}. Let $V_2(\alpha_2, t)$ give the state vector at time t when simulated from an initial state of $(1-\alpha_2)V_{H_1} + \alpha V_{L_1}$ at time t_1. We now find a time, t_2, such that for $\alpha_{H_2} < \alpha_2 < \alpha_{L_2}$ and $t_1 \le t \le t_2$

$$V_2(\alpha_2, t) \approx \frac{\alpha_{L_2} - \alpha_2}{\alpha_{L_2} - \alpha_{H_2}} V_2(\alpha_{H_2}, t) + \frac{\alpha_2 - \alpha_{H_2}}{\alpha_{L_2} - \alpha_{H_2}} V_2(\alpha_{L_2}, t) \quad (6)$$

Section 3.3 describes the selection of t_2 in more detail. The points $V_2(\alpha_{H_2}, t_2)$ and $V_2(\alpha_{L_2}, t_2)$ are the endpoints of an interval that we can use for further bisection.

Figure 2 illustrates our repeated bisection method. We write $V_{H_{i-1}}$ and $V_{L_{i-1}}$ for the endpoints of the interval that we use for starting the i^{th} round of bisection. We write α_{H_i} and α_{L_i} for the interval that we reach at the end of the i^{th} round, and V'_{H_i} and V'_{L_i} for the corresponding voltage points. We have for $i \ge 2$:

$$\begin{aligned} V'_{H_i} &= (1-\alpha_{H_i})V_{H_{i-1}} + \alpha_{H_i} V_{L_{i-1}} \\ V'_{L_i} &= (1-\alpha_{L_i})V_{H_{i-1}} + \alpha_{L_i} V_{L_{i-1}} \\ V_{H_i} &= V_i(\alpha_{H_i}, t_i) \\ V_{L_i} &= V_i(\alpha_{L_i}, t_i) \end{aligned} \quad (7)$$

where $V_i(\alpha, t)$ is the voltage state reached at time t starting from an initial voltage of $(1-\alpha)V_{H_{i-1}} + \alpha V_{L_{i-1}}$ at time t_{i-1}. Thus, $V_{H_i} = V_i(\alpha_{H_i}, t_i)$ and $V_{L_i} = V_i(\alpha_{L_i}, t_i)$. Because we choose t_i to be small enough to allow for an accurate linear approximation of V_i we have

$$V_i(\alpha, t_i) \approx \frac{\alpha_{L_i} - \alpha}{\alpha_{L_i} - \alpha_{H_i}} V_{H_i} + \frac{\alpha - \alpha_{H_i}}{\alpha_{L_i} - \alpha_{H_i}} V_{L_i}. \quad (8)$$

Rather than attempting to simulate an entire trajectory from the critical transition of the data to the resolution of metastability, our method divides such trajectories into multiple segments. The dynamics of metastable circuits ensures that trajectories that resolve to different logical states will diverge exponentially with time. While this divergence causes serious stability problems when trying to find metastable trajectories by numerical integration alone, we use the divergence to find intervals for restarting our bisection that are larger than their predecessors. Because these divergent trajectories are initially quite close to each other, we can build linear maps from the trajectories at each step of our computation back to trajectories

in earlier steps. The next section describes how these mappings allow us to compute failure probabilities accurately.

3.2. Computing Bounds for $\Delta_{t_{in}}$

Using equation 4, it is sufficient to compute $\Delta_{t_{in}}(t_s)$ to determine the failure probability of a synchronizer that has t_s time units of settling time. For each bisection step, we find trajectories for which metastability takes longer and longer to resolve. These correspond to larger values for t_s and smaller values for $\Delta_{t_{in}}$. The value of t_s can be observed directly from the simulation: we simply note that time of the corresponding output transition (e.g. the time that the output signal crosses $V_{dd}/2$). This section describes how we compute $\Delta_{t_{in}}$. There are two main issues that we address:

1) At most steps, our algorithm bisects a voltage interval. We need to map these initial voltage states back to input transition times in a way that allows accurate calculation of $\Delta_{t_{in}}$.
2) Consider a synchronizer whose input makes a low-to-high transition. If the synchronizer settles high, we can observe t_s directly. On the other hand, if the synchronizer settles low, there may be no change on the output. Thus, to compute $\Delta_{t_{in}}$ we need to find the latest that the input can change and still have the synchronizer settle high. This effectively means finding the perfectly metastable trajectory that never resolves.

For $i \geq j$, let $Back_{i,j}$ map a voltage from the segment that was used to start the i^{th} round of bisection, i.e. (V_{H_i}, V_{L_i}), back to the corresponding voltage on the segment (V_{H_j}, V_{L_j}) if $j > 1$ or the transition time of the data input if $j = 1$. Inverting equations 5 and 8 we get:

$$\begin{aligned} Back_{i,j}(V) &= V, && \text{if } i = j \\ &= \tfrac{\|V_{L_1} - V\| t_{H_1} + \|V - V_{H_1}\| t_{L_1}}{\|V_{L_1} - V_{H_1}\|}, && \text{if } i = j = 1 \\ &= Back_{i-1,j}\left(\tfrac{\|V_{L_i} - V\| V'_{H_{i-1}} + \|V - V_{H_i}\| V'_{L_{i-1}}}{\|V_{L_i} - V_{H_i}\|}\right), && \text{if } i > 1 \end{aligned} \tag{9}$$

We now need to determine the latest time that the input can make a low-to-high transition such that the synchronizer eventually settles to a logical high value. Each bisection phase computes bounds for this time, and these bounds become progressively tighter. For each i, $V'_{H_{i+1}}$ is an initial condition for which the synchronizer settles high and $V'_{L_{i+1}}$ is an initial condition for which it settles low. Let k denote the total number of bisection rounds performed in the analysis. We could use $Back_{k,1}(V'_{H_{k+1}})$ and $Back_{k,1}(V'_{L_{k+1}})$ as lower and upper bounds respectively for the input times that lead to "perfect" metastability. However, this would lead to large numerical errors, because $\Delta_{t_{in}}$ may be very small compared with the time of the input transition.

To avoid these numerical issues, we perform the subtraction as early as possible. For example, when we find $V'_{H_{i+1}}$ we know its settling time – we call this $t_{s,i}$. Now, when we complete our last round of bisection, we have $V'_{H_{k+1}}$. Note that $Back_{k,i}(V'_{H_{k+1}})$ gives the state at time t_i that led to $V'_{H_{k+1}}$ at time t_k. Likewise, $Back_{k,i}(V'_{L_{k+1}})$ gives the state at time t_i that led to $V'_{L_{k+1}}$ at time t_k. Let

$$\begin{aligned} \rho_1 &= \tfrac{t_{L_1} - t_{H_1}}{\|V_{L_1} - V_{H_1}\|}, \\ \rho_i &= \rho_1 \textstyle\prod_{j=2}^{i} \tfrac{\|V'_{L_j} - V'_{H_j}\|}{\|V_{L_j} - V_{H_j}\|}, \quad \text{if } i > 1 \end{aligned} \tag{10}$$

We now have

$$\begin{aligned} & \|Back_{k,i}(V'_{H_{k+1}}) - V'_{H_{i+1}}\| \rho_i \\ \leq\ & \Delta_{t_{in}}(t_{s,i}) \\ \leq\ & \|Back_{k,i}(V'_{L_{k+1}}) - V'_{H_{i+1}}\| \rho_i \end{aligned} \tag{11}$$

This is the formula that we use for computing bounds for $\Delta_{t_{in}}$ in the remainder of this paper.

3.3. Implementation Issues

We implemented the computation described above using MATLAB. We used the simple, short-channel transistor model from [8, chap. 2.5.2] and adjusted the model parameters so that inverter transition times matched those from HSPICE for the 0.18μ TSMC CMOS process. We used MATLAB's ode45, a fourth-order Runga-Kutta integrator, for numerical integration.

Our algorithm has an inherent trade-off between the accuracy of the integrator and the accuracy of the linear approximation. If we continue bisection to produce a very small segments for $(V'_{H_{i+1}}, V'_{L_{i+1}})$ then the linearization will be very accurate, but the results will be sensitive to errors from the integrator. If we use a larger interval, then linearization error will dominate. In our implementation, we bisect until we produce a segment for $(V'_{H_{i+1}}, V'_{L_{i+1}})$ this is roughly one-tenth the length of (V_{H_i}, V_{L_i}).

A similar trade-off occurs in the choice of the t_i's. Using larger values reduces the number of rounds of bisection required to reach a pre-specified t_s or $\Delta_{t_{in}}$, thus reducing the time for the algorithm to run and the impact of integration error. On the other hand, large values for t_i lead to larger linearization errors. In our implementation, we integrate three trajectories at the end of each bisection round. These start from $V'_{H_{i+1}}$, $V'_{L_{i+1}}$, and $(V'_{H_{i+1}} + V'_{L_{i+1}})/2$ respectively. At each time step of the integration, we compare the integrator's value for the trajectory starting from $(V'_{H_{i+1}} + V'_{L_{i+1}})/2$ with the value obtained by linear interpolation from the other two starting points. We chose t_{i+1} as the largest time for which this error is less that 2% of the magnitude of the voltage vectors.

Any method that relies on numerical integration for analysing metastability must address the instability that arises from the positive eigenvalue of the Jacobian, λ_+. The (V_{H_i}, V_{L_i}) segments are parallel to the corresponding eigenvector. Thus, the calculation of $\|V_{L_i} - V_{H_i}\|$ is susceptible to this instability. Instead of calculating the difference explicitly, we compute the small signal sensitivity of the circuit along a trajectory from t_i to t_{i+1}. In particular, we augment the ODE from equation 1 with a matrix $S(t)$ where

$$\begin{aligned} \dot{S} &= (\text{Jac} f(y))S \\ S(t_i) &= I \end{aligned} \tag{12}$$

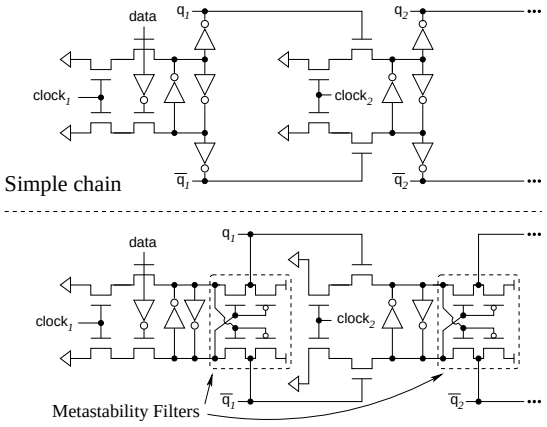


Fig. 3. Synchronizer Chains

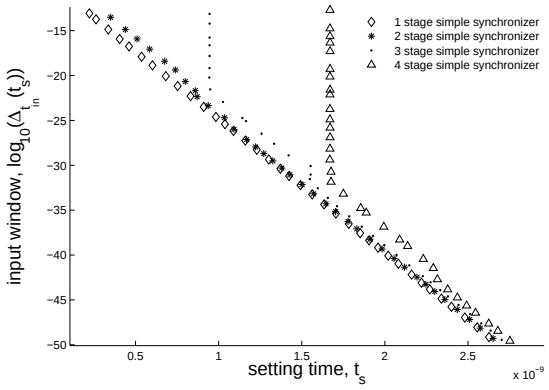


Fig. 4. $\Delta_{t_{in}}(t_s)$ vs. t_s for the simple synchronizer chain

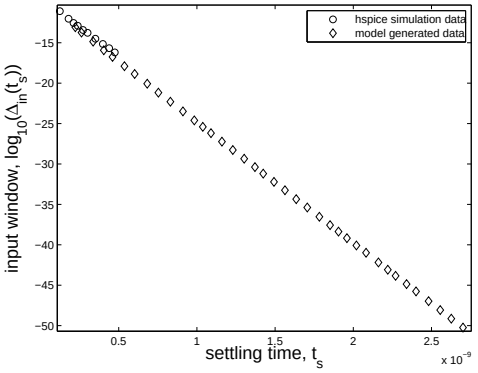


Fig. 5. $\Delta_{t_{in}}(t_s)$ vs. t_s for a one-stage, simple synchronizer

where Jac is the Jacobian operator. We now have that $S(t_{i+1})(i,j) = \partial y_i(t_{i+1})/\partial y_j(t_i)$. This sensitivity matrix allows us to calculate the ρ_i's (including ρ_1) from equation 10 without numerical differencing. Our experimental results show that our method is very robust, and we expect to prove this using standard error analysis techniques in future work.

4. Results

We tested our algorithm by analysing the failure probabilities of chains of synchronizers. Figure 3 shows our two implementations. The chain on the top uses simple inverters to couple the output of one latch to the input of the next. The chain on the bottom uses a "metastability filter" – the n-channel transistors in the filter remain in cut-off until two sides of the cross-coupled inverter pair differ by a n-channel threshold. This ensures that the latch output does not change until metastability has resolved. For all of our measurements, we use a clocks with a period of $1.5ns$ and 50% duty cycle. The phase for each latch in the chain is half of a period later than that for the previous stage. We "freeze" the clocks when the final stage of the chain goes opaque so we can observe settling times greater than the clock period.

Figure 4 plots the input window, $\Delta_{t_{in}}(t_s)$, as a function of the output settling time, t_s, for the chain coupled with simple inverters. We calculate the settling time from the clock edge that makes the first latch in the chain opaque to facilitate comparison of chains of differing numbers of latches. For the multistage chains, the curve drops below the straight line that would be predicted by a simple, small-signal analysis. In a n-stage synchronizer chain, an input transition first affects the output when latch $n-1$ goes opaque and latch n goes transparent. At this time we can observe the metastability of latch $n-1$. Half a clock cycle later, latch n goes opaque, and the synchronizer remains in an unresolved state if this latch becomes metastable as well. This is the "back edge effect" described in [5]. Whereas [5] observed a positive back edge (increased delay) for the 74F5074 flip-flop, our synchronizer exhibits a negative back edge. We conjecture that the inverters coupling the stages have greater gain-bandwidth products than the cross-coupled pairs for the transistor sizes that we simulated.

Figure 5 shows $\Delta_{t_{in}}$ versus t_s for a single stage synchronizer. Here, we compare with values calculated using HSPICE's bisection capability. In the figure, the circles are the HSPICE simulation data and the diamonds were calculated with our approach. We find that in the range where HSPICE can compute a result, the two agree quite well. However, HSPICE can only determine $\Delta_{t_{in}}$ down to 0.2 femtoseconds – if f_c and f_d are both 1GHz, this corresponds to a MTBF of about five milliseconds. In contrast, our method easily calculates $\Delta_{t_{in}}$ to 10^{-50} seconds or less corresponding to an MTBF of greater than 10^{24} years.

Figure 6 compares synchronizer chains with and without metastability filters. We first observe that the simple synchronizer outperforms the synchronizer with the metastability filter. At first, this seemed surprising as a metastable latch in the filtered synchronizer can only corrupt its successor as it exits metastability, whereas a metastable latch in the simple synchronizer is always visible to the next stage. However, the couping inverters in the simple inverter are faster than

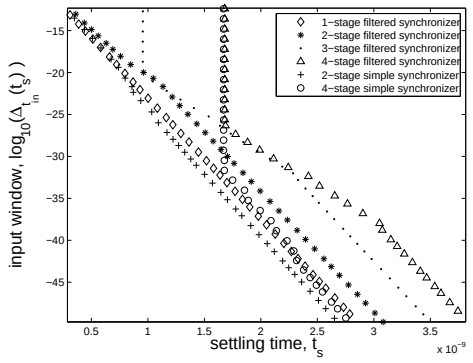


Fig. 6. Comparing the Two Synchronizer Designs

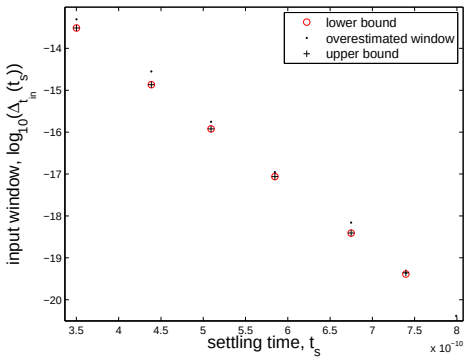


Fig. 7. Upper and Lower bounds for the $\Delta_{t_{in}}$

the metastability filters of the alternative design. This added speed gives the simple design its advantage. This also shows how our approach can be used to confirm or refute proposed design optimizations in regimes that cannot be resolved by traditional simulators. Our second observation is that the filtered synchronizer has a positive back edge, like those described in [5]. Again, we attribute this to the added delays of the filter circuit.

As described in section 3.2, our algorithm computes upper and lower bounds for $\Delta_{t_{in}}$. Figure 7 plots these bounds for a two-stage, simple synchronizer chain. For comparison purposes, we also plot the value of $\Delta_{t_{in}}$ obtained after each bisection round if we use $V'_{L_{i+1}}$ to compute our estimate. Not surprisingly, the latter approach significantly overestimates $\Delta_{t_{in}}$. In contrast, our method computes very tight bounds with the difference between the upper an lower bound only becoming visible at the final data point.

5. Conclusions

We presented a novel method for measuring failure probabilities and MTBF for synchronizer circuits. By combining linear interpolations between nearby trajectories with numerical integration to account for the non-linearities of synchronizer circuits, we overcome the limitations of traditional small-signal analysis or numerical simulation based approaches. This allows us to verify MTBFs of a million years or greater. To the best of our knowledge, our approach is the first to demonstrate the ability to verify realistic reliability requirements for synchronizer circuits. We demonstrated our methods by implementing them in Matlab, using them to compare two designs for synchronizer chains, and comparing our results with those from HSPICE.

Our methods build upon numerical techniques that are already used in circuit simulators such as numerical integration, bisection, and calculation of small-signal sensitivities by augmenting the ODE model. Thus, we believe that our approach is well suited for integration into existing circuit simulators.

Our method is fully automated. We are interested in applying it to other circuits where metastability plays an important role. These include analog-to-digital converters, sense amplifiers, and high-speed digital circuits that with minimal reset circuitry. We presented our approach as a method for computing failure probabilities. It should be possible to extend this approach to also generate traces of metastability failures. We are also interested in extending our approach to automatically optimize transistor sizes for synchronizer circuits.

Simulating metastable behaviour is inherently difficult because of the numerical instability introduces by the positive eigenvalue of the Jacobian operator near the metastable equilibrium. With our approach, we believe that this error shows up in the absolute time of the vulnerability window, but has little impact on the value computed for the width of the window. Our experience with our test cases supports this conjecture, and we plan to perform a formal error analysis to test this conjecture. Such an analysis should also provide a basis for determining the optimal criteria for moving from one bisection phase to the next and to determine the size of the time interval between these phases.

References

[1] T. Chaney and C. Molnar, "Anomalous behavior of synchronizer and arbiter circuits," *IEEE Transactions on Computers*, vol. C-22, no. 4, pp. 421–422, Apr. 1973.

[2] M. Hurtado, "Structure and performance of asymptotically bistable dynamical systems," Ph.D. dissertation, Sever Institute, Washington University, Saint Louis, MO, 1975.

[3] L. Marino, "General theory of metastable operation," *IEEE Transactions on Computers*, vol. C-30, no. 2, pp. 107–115, Feb. 1981.

[4] C. Mead and L. Conway, *Introduction to VLSI Systems*. Addison Wesley, 1979.

[5] D. J. Kinniment, K. Heron, *et al.*, "Measuring deep metastability," in *Proceedings of the Eleventh International Symposium on Asynchronous Circuits and Systems*, Mar. 2006, pp. 2–11.

[6] D. J. Kinniment, A. Bystrov, *et al.*, "Synchronization circuit performance," *IEEE Journal of Solid-State Circuits*, vol. 37, no. 2, pp. 202–209, 2002.

[7] Y. Semiat and R. Ginosar, "Timing measurements of synchronization circuits," in *Proceedings of the Ninth International Symposium on Asynchronous Circuits and Systems*, May 2003, pp. 12–16.

[8] D. A. Hodges, H. G. Jackson, and R. A. Saleh, *Analysis and Design of Digital Integrated Circuits in Deep Submicron Technology*. McGraw Hill, 2004.