Metastable Behavior in Digital Systems

Lindsay Kleeman and Antonio Cantoni
University of Newcastle, New South Wales

Fault-free digital circuits may malfunction when asynchronous inputs have critical timing combinations that result in metastable operation. This mode of failure is often overlooked in digital system design and reliability analysis. Here, we survey developments in the study of metastable behavior and identify their relevance to digital system design and reliability, and we describe and evaluate a number of techniques for reducing the probability of metastable failure.
Any digital circuits with asynchronous inputs are susceptible to failure even when all their components are fault-free. They may fail as a result of metastable operation when their inputs have critical timing combinations. Metastable operation of a digital circuit is a malfunction in which the circuit lingers indefinitely between two stable states because of marginal triggering of the circuit. Failures resulting from metastable behavior are particularly troublesome and mysterious because they are intermittent, random, and virtually untraceable.

Most digital systems have asynchronous inputs since they must interact with external events that generate input changes that are random with respect to internal system activities. For example, because the timing of people making telephone calls through a telephone exchange is independent of internal exchange switching events, the exchange has asynchronous inputs. The problem of processing asynchronous inputs occurs throughout digital systems. Hence, it is important to understand and constrain metastable unreliability.

Here we will review developments in the study of metastable behavior in digital circuits and we will present and analyze techniques for improving metastable reliability.

**EXAMPLES**

A digital circuit that clearly demonstrates the problems involved in processing asynchronous inputs is the arbiter. An arbiter is a decision circuit that uniquely allocates a shared resource to one of a number of competing autonomous requesters. For example, a multiprocessor memory employs an arbiter circuit to ensure that only one processor at a time accesses the memory. The inputs to an arbiter are request signals and the outputs are acknowledge signals, and one of each type of signal is associated with every requester. Only one acknowledge output is asserted at a time, but the asynchronous request inputs of an arbiter may be asserted at any time regardless of other inputs and the internal state of the arbiter circuit. Thus simultaneous or nearly simultaneous requests can occur.

The time between requests from different requesters may be arbitrary. The arbiter is expected to resolve these requests so that only one request at a time is satisfied, no matter how small the time interval between requests is. Because request timing will continuously vary, a knife edge decision must be made at some point. A deviation to either side of this point, no matter how small, should result in a single request being serviced. In other words, for the arbiter to operate perfectly the sharpness of the knife edge must be infinite. We will see later that for physically realizable circuits this cannot be achieved within a finite decision time.

We should compare the case of asynchronous inputs with that of synchronous inputs. Synchronous inputs to a system obey strict timing relations governed by a reference such as a system clock. An example of this is shown in Figure 1. A synchronous input to a flip-flop derived from the other parts of the system changes only within a specific part of the system cycle clock, thus satisfying setup and hold time relations required by the flip-flop. In other words, the input does not change within an interval of time defined in relation to the sampling clock edge by the flip-flop's setup and hold times. Using bounds on delays and the required timing relations for inputs to logic elements, one can design a synchronous circuit to operate reliably in the sense that device timing requirements are met. In the canonical form of a synchronous circuit shown in Figure 2, the synchronous inputs are derived from within the system and hence are incorporated in the canonical form without being shown explicitly. A synchronous input must be derived from the system clock in order that it have the required timing relation to the clock. The input or domain and hold time required by the flip-flop, namely the setup and hold time requirements, are satisfied when the following relations hold:

\[ t_{SU} = T_C + t_{HST} + t_{LST}, \]

\[ t_{HD} = t_{HST} + t_{LST}, \]

where \( T_C \) is the system clock period, \( t_{SU} \) is the maximum combinational circuit delay, \( t_{LST} \) is the minimum combinational circuit delay, \( t_{HST} \) is the maximum flip-flop propagation delay, \( t_{LST} \) is the minimum flip-flop propagation delay, \( t_{SU} \) is the setup time required by the flip-flop, and \( t_{HD} \) is the hold time required by the flip-flop. One must ensure that the clock period is long enough to be able to satisfy the setup time and hold time required by the flip-flop. One must also ensure that the clock period is long enough to be able to satisfy the setup and hold time conditions (Equation 1) to be satisfied and that the combinational circuit has been designed with sufficient minimum delay for the hold time condition (Equation 2) to be satisfied.

The design of circuits with asynchronous inputs is not as straightforward, for example, setup and hold time conditions cannot always be satisfied for a flip-flop that has asynchronous

![Figure 1. Setup and hold time conditions satisfied by a synchronous input.](image-url)
3

Figure 2. Synchronous circuit (canonical form). The synchronous inputs are derived from within the circuit.

Figure 4. Runt pulse generated from two well-formed inputs to an AND gate.

Figure 3. D-type flip-flop sampling a changing data input. The changing input causes the flip-flop to enter a metastable state—that is, a state in which its output oscillates between logic levels.

inputs and also requires a clock input, since input changes can occur at any time with respect to the system clock. When input changes occur in the interval of time defined by the setup and hold times of the flip-flop, the setup and hold time constraints are violated. This situation is shown in Figure 3, where a D-type flip-flop samples a changing data input and as a result enters a state in which its output lingers or oscillates between logic levels—a metastable state.

If a flip-flop that does not require a clock input, such as an RS flip-flop, is used, other timing relations required by the flip-flop, such as a minimum separation between the release of the R pulse and the release of the S pulse, may not be satisfied. Moreover, asynchronous inputs can generate misshapen or undersized pulses known as runt pulses. Figure 4 shows how a runt pulse can be generated from two well-shaped inputs to an AND gate. Runt pulses on the input of, for example, an RS flip-flop may give rise to marginal triggering and thus to metastable operation.

Figure 5. Timer employing a divided clock derived from the CPU clock. The timer may generate interrupts that are inadvertently timed so as to always occur in the vicinity of the interrupt sampling instant.

In the initial design of large, complex, distributed synchronous systems, the effect of using divided down clocks and the presence of large propagation delays are often overlooked. These can combine to induce, in a nondeterministic way, marginal triggering of devices. For example, a timer employing a divided clock derived from the CPU clock may generate inadvertently timed interrupts that always occur in the vicinity of the interrupt sampling instant. Figure 5 shows such a system, where the delay from a CPU positive clock edge to interrupt sampling is $t_1$, the clock divider introduces a delay of $t_2$, and the timer asserts its output, a delay $t_3$, after a positive clock edge. Marginal triggering of the interrupt sampling mechanism within the CPU can occur if

$$t_0 + t_1 = t_0 + k t_c$$

where $k$ is an integer. Thus, marginal triggering is induced by synchronism, and the probability of failure when Equation 3 holds may be higher than if an asynchronous timer was employed. Metastability can also occur in a fully synchronous system when power supply disturbances occur. The disturbances act to extend circuit delays, and the extended delays can result in violation of timing constraints, especially in circuits with tight timing tolerances.

**THE METASTABLE STATE**

Let us consider the behavior of a flip-flop under marginal triggering conditions. Similar behavior can be exhibited by any device with at least two stable states and two input contingencies that drive the device to either stable state. A flip-flop is said to be marginally triggering when its output fails to settle to a logically defined state (0 or 1) within its maximum normal delay time. The normal maximum delay time is defined to be the maximum delay required for the output to reach a logically defined state under specified conditions on, for example, setup and hold times. The
Attempts have been made to design a perfect synchronizer. However, careful examination reveals that these designs are indeed subject to failure caused by metastable behavior—or synchronizer failure, as we will henceforth call it. This is particularly evident in the analyses done by Chaney and Moinan. It is generally accepted that a perfect synchronizer cannot be physically realized.

We can use a generic linear system, such as a dynamic system described by differential equations, to analyze any physical system that performs synchronization. Maimo has shown that any system having certain basic properties—which we will describe below—cannot avoid exhibiting metastable behavior. The system is assumed to be fully described by its state and is represented by a state transition function that has as arguments the initial system state, system output functions, and time. The value of the state transition function is the system state at a given time. The state transition function is assumed to obey several reasonable axioms that physical systems are assumed to conform to:

- The system state does not depend on future inputs.
- The system's future behavior is dependent only on the current system state and subsequent inputs, with this relationship being time-invariant.
- The state transition function is continuous with respect to the initial state and time (but not necessarily with respect to the system input functions).

There are also assumed to be at least two stable regions of the state space: one under "idling" inputs, and another input that moves the state from one stable region to the other.

For example, in the case of an RS flip-flop, one stable region corresponds to the reset state, while the other corresponds to the set state; with the idling inputs corresponding to the R and S inputs low (inactive). The terms stable and unstable are used in the sense that if the state is perturbed just outside the stable region then the state will return to the stable region under the idling inputs. Obviously the stable region is a function of the set of idling inputs.
The state transition function is assumed to be time-invariant, although some have claimed that the theory can be developed without this assumption. Under the assumptions given thus far, Marino shows that if there is a range of input functions that results in the system state not reaching either of the stable regions of states within a settling time \( T \), where settling inputs are applied during the settling time. The settling time, \( T \), can be arbitrarily large, with the width of the range of input functions that \( T \) is, which corresponds to the notion of metastable behavior. However, the inputs in the derivation of this result were allowed to contain discontinuities with respect to time (that is, jumps or steps), to prove a similar result with inputs that have bounded first derivatives, Marino introduces the additional assumption that the state transition function is continuous with respect to the input functions. The condition of the input functions having bounded first derivatives can be generalized to any continuous set of input functions, and thus can incorporate virtually any conceivable set of asynchronous input contingencies that can occur in practice. We discuss this matter elsewhere.\(^7\)

Marino's results are directly relevant to physically realizable systems, since we believe that his assumptions apply to all real systems. The most important assumption is the continuity of the system state with respect to the inputs, time, and initial state. The fundamental result is that continuous systems cannot consistently make discrete decisions within a finite time based on non-discrete or continuously variable inputs. This digital system—which intrinsically relies on discrete representation of information—cannot perform with perfect reliability within a finite time when it must process inputs whose timing is crucial in the decision process and yet can take on a continuum of configurations.

Note that for metastable behavior to be unobservable, it is not sufficient just for inputs to be asynchronous; there must also be two input contingencies that drive the system to different states, and a continuum of input contingencies between them. For example, a single-input digital circuit that counts asynchronous occurrences of input edges separated by an even number of clock periods can be operated without metastable failure, since the system state is not dependent on the timing or the state of the system.

However, if the restriction of a minimum separation between input edges is lifted, then the system will not behave consistently and will no longer be useful. In practice, this is a real issue and cannot be ignored.

IEEE DESIGN & TEST OF COMPUTERS
To achieve high reliability, one should design circuits in which the worst case of probability of metastable failure can be estimated with confidence. Otherwise, another degree of unreliability is introduced, namely, uncertainty in estimating reliability. We believe one cannot design a system to be extremely reliable without providing a way to reasonably establish the limits of probability of failure. A design is very reliable only if the technique for evaluating the probability of failure is very reliable. Thus, it is essential to establish such a technique.

FIRST-ORDER MODEL

We use the first-order model of a flip-flop proposed by Veenstra[11] in this analysis. It is shown in Figure 6. Voltages $v_1$ and $v_2$ are assumed to be generated within the flip-flop at time $t=0$ by the sampling process $v_s$ representing the output of the flip-flop. The solution to the differential equations for the circuit, for $v_1$, is

$$v_1 = \frac{v_{02} - v_{01}}{2} e^{\frac{t}{RC}} + \frac{v_{02} + v_{01}}{2} e^{-\frac{t}{RC}} \tag{5}$$

Neglecting the decaying exponential term of Equation 4 and letting $v_1 = v_{02} - v_{01}/2$ and $\tau = RC/(A-1)$, we obtain

$$v_1 = v_2 e^\tau \tag{6}$$

For marginal triggering conditions $v_1$ is assumed to be uniformly distributed between $-v_2$ and $v_2$, where $v_2$ is the “boundary” of the flip-flop’s linear region of operation. Beyond $v_2$, the output is assumed to be defined as one of the two logic levels. The assumption that $v_2$ is uniformly distributed can be made because the clock edge samples a uniformly random input $v_s$, as shown in Figure 7.

---

6

December 1987
The probability of the metastable behavior lasting longer than $t$, $P(t > t)$, is the probability that $v_1 < v_2$, at $t$. The voltage at $t$ can be mapped back to its initial voltage in a monotonic way, since the relation shown in Equation 5 is monotonic. This is illustrated in Figure 8.

Thus
\[ P(t > t) = e^{-t / T} \]
for uniform distribution of $v_1$.

Because the voltages at time $t$ are always uniformly distributed when $v_1$, the superimposed small-amplitude unbiased noise on $v_1$ has a negligible effect on the distribution of $v_2$. Consider two adjacent voltage regions. The number of states forced out of a region because of noise will be replaced by an equal number forced in from the other region. This qualitative argument has been verified experimentally by Coateson and Warm and Veendrick.

**EXPERIMENTAL MODEL**

Experimental measurements made on flip-flops at $10^{-4}$ have led to the verification of an exponential model for predicting the failure rate arising from metastable behavior. This model was first presented by Rosenberger and Charney and Charney. It is described here in terms of an RS flip-flop but can be applied to any flip-flop.

Suppose the time between the completion of the reset pulse and the completion of the reset pulse is $t_1$, and $t_1$ has a uniform probability density $p(t)$ over an interval $t_2 < t_1 < t_3$. Then

\[
p(t)dt = \begin{cases} 
0 & t < t_2 \\
\frac{1}{t_3 - t_2} & t_2 \leq t \leq t_3 \\
0 & t > t_3
\end{cases}
\]

where the flip-flop is set by $t_1 = t_2$, and reset by $t_1 = t_3$, within a normal propagation delay.

For $t_1$ uniformly distributed over this interval, $P(t_1)$ is defined as the probabil-
ity that the flip-flop output has not reached a logically defined and stable value at time \( t = 0 \) after triggering. Experiments have shown that for \( t > b \) this probability can be represented as

\[ P(t) = 1 - e^{- \frac{t}{T}} \]  

\[ Y(t) = 1 - e^{- \frac{t}{T}} \]  

where \( b \) is the parameter \( t \) and the parameters \( t \) and \( T \) depend on the particular flip-flop.

The parameter \( b \) is determined experimentally by using data to determine the minimum value of setting times \( t = b \). This is done so the parameters in Equation 7 will be constant.

Various values for the parameters of Equation 7 are given by Chaney, for available flip-flops. In Chaney's work we can see that there is a large variation in TA (equivalent to \( t \) in our discussion) and \( T \) within the same flip-flop type, resulting in many orders-of-magnitude differences in the calculated examples of the mean time during which the synchronizer is unaligned (M9T). This dramatic range of performance is particularly disturbing in light of trying to design for a particular error rate; therefore, designs should be very conservative to guarantee a failure rate below a minimum value.

In time-critical or high-reliability applications, individual testing of the flip-flop may be worthwhile. A practical way of doing this is to incorporate test circuitry on the same chip as the synchronizer. The testing can be partially automated so that a fast determination of a flip-flop's suitability can be made at fabrication. A design similar to that proposed by Rosenberger and Chaney can be employed.

We should also note in Veendrick and Chaney, the extreme variation in failure rate with respect to the setting time. Doubling the setting time from 20 to 40 ms results in an improvement of more than ten orders of magnitude in M9T for some flip-flops.

**Synchronizer failure rate**

How does one calculate the failure rate of a synchronizer? Let us take as an example a synchronizer consisting of a D-type flip-flop with its data connected to the asynchronous input to be synchronized by the clock (Figure 9).
where the approximation is due to not always being an exact multiple of \( T \). For \( 12 > T \), this can be neglected, giving

\[
\text{Prob of failure} \approx \frac{1}{2} e^{-ABk}\frac{T}{\lambda T} = e^{-ABk} (12)
\]

where \( T = 1/T \). Equation 12 shows that the occurrence of synchronization failures is a Poisson process—that is, that the interarrival times of failures are exponentially distributed. We can now calculate the mean failure rate, \( \lambda \), and the mean time between failures, \( \lambda \tau \), where \( \lambda = 1/\tau \),

\[
\lambda = \frac{T}{\lambda T} = \frac{T}{\lambda T} \quad (15)
\]

\[
\lambda = \frac{T}{\lambda T} = \frac{T}{\lambda T} \quad (16)
\]

Recall that \( T \) is the time interval after the sampling clock edge during which the synchronizer can recover, and \( \lambda \) is a synchronizer device time constant. Note the exponential dependence of \( T \) and \( \lambda \) in Equations 15 and 16. This explains the enormous variation possible in the MTRF when modeled variations occur in \( T \) or \( \lambda \).

**IMPROVING PERFORMANCE**

Several techniques can be used to improve synchronizer performance. They include using fast devices, extending the settling time, using a phase-locked loop, and employing masking and redundancy.

**FAST DEVICES**

Several researchers have used tunnel diodes to increase the speed of flip-flops—that is, to give them small values of \( \tau \). Using fast logic for synchronizers and slower logic for the remainder of the system represents a solution to the problem of achieving high reliability in synchronization. However, when a system is built for maximum speed, fast logic is likely to be used in all parts of the system and the system’s speed is determined by the slowest part. Thus, as speed is increased in high-speed systems, the ratio \( \tau/T \) is no longer present and alternative strategies for achieving high synchronization reliability must be sought.

**EXTENDED SETTLING TIME**

A simple technique for achieving a certain synchronization reliability is to allow adequate settling time after each synchronization event. As shown earlier, exponential improvements in metastable reliability can be gained by increasing the ratio of the settling time to \( \tau \), the device constant.

For example, in a synchronous system, this can be achieved by cascading \( N + 1 \) flip-flops to obtain a delay of \( N \) clock periods before a sampled input is seen by the rest of the system. (Figure 10.) During the \( N \)-clock periods, a latched input value has the opportunity to resolve to either valid logic value as it is shifted through the synchronizer flip-flops.

Another scheme that allows approximately \( N \)-clock periods of settling time is shown in Figure 11. The system clock is divided by \( N-1 \) to achieve an uninterrupted settling time of \( N \)-clock periods between \( FF_1 \) and \( FF_2 \) and an additional clock period between \( FF_2 \) and \( FF_3 \). The function of \( FF_2 \) is to elimin-
rate the delay of the divider circuit, $T_d$, from the final synchronized input available to the system.

To compare the performance of the scheme shown in Figure 10 to that of the scheme shown in Figure 11, let us adopt a simple model for the triggering of metastable behavior in cascaded flip-flops. We assume that the clock edge sampling process transfers the internal state of FF$_1$ to FF$_N$ with a propagation delay $T_p$, as shown in Figure 12. The value of $T_p$ should be about the same as the value of a few gate delays. Although this model takes a rather elementary view of the sampling process, it serves as a starting point for analysis and discussion.

The failure of a synchronizer is considered to occur when the final flip-flop (FF$_N$ in Figure 11 and FF$_{prev}$ in Figure 10) samples an unresolved value. All flip-flops are assumed to have identical characteristics.

The cascaded flip-flop synchronizer of Figure 10 can be thought of as one flip-flop with a settling time $T_s$, where

$$T_s = NT_p - (N-1)T_d$$

and $T$, is the system clock period.

Using $T_s$ from Equation 14 and the same exponential input edge distribution that we used when we showed how to calculate the synchronizer failure rate, we obtain the mean time between failures for the cascaded flip-flop synchronizer, MTBF, where

$$MTBF = T_0^T \left( \frac{N!T_p}{(N-1)!T_d} \right) e^{-T_0/T_s}$$

Applying the metastable cascaded flip-flop model again, we observe that the divided clock synchronizer behaves like a single flip-flop with settling time $T_s$ where

$$T_s = NT_p - T_d$$

Note that for the divided clock synchronizer, the input sampling period is no longer $T$, but $(N-1)T$. Thus, from Equations 14 and 17, for the divided clock synchronizer.
For $N \geq 2$ and $\tau \gg \tau_0$, Equation 10 indicates that the divided clock synchronizer performs better than the cascaded flip-flop synchronizer. In terms of the total number of hardware components, the divided clock scheme is better for large $N$, particularly when multiple asynchronous inputs need to be synchronized. The disadvantage of the divided clock scheme is its $(N-1)$ times slower sampling frequency, which may cause loss of information that the cascaded flip-flop scheme would capture.

Comparing Equations 16 and 19 with Equation 18, we see that increasing the settling time of a synchronizer can result in an exponential improvement in metastable reliability. The drawback is the increased delay, which may be a serious problem in time-critical systems. A compromise between speed and reliability must be reached. This compromise can be very good if manufacturers produce synchronized flip-flops with guaranteed maximum values of the critical device time constant, $\tau$, as opposed to the highly uncertain values provided by existing devices. Devices are being produced that are claimed to be metastable-hardened—that is, to have small, but unspecified, $\tau$ values.

**Example:** Let us say that a microprogrammed system has a 40-msec clock period and that an asynchronous input with mean time between edges of 200 nsec has an exponential distribution of edge interarrival times. The input is synchronized to the system by means of flip-flops with $\tau = 20$ nsec. In this case, it is not unlikely worst case.

IEEE DESIGN & TEST OF COMPUTERS
Suppose a simple synchronizer—which has one clock period of settling time—is used. Applying Equation 16, we obtain an MTBF of 114 ms. Suppose now a cascaded flip-flop synchronizer is employed to obtain two clock periods of settling time and suppose the flip-flop transfer delay, \( t_s \), is equal to 10 ns. Then, from Equation 16, we obtain an MTBF of 7.4 days. For high-reliability applications, this MTBF would be too short.

A cascaded flip-flop synchronizer with three clock periods of settling time, however, would yield an MTBF of 66,000 years. And a divided clock synchronizer, with four clock periods of settling time and with a divider delay of 10 ns, would yield from Equation 16 an MTBF of 122,000 years. At the latter two levels of reliability, the difference in performance is purely academic.

**METASTABLE DETECTION**

The use of metastable detection and a clock that can be paused has been suggested as a way to reduce the probability of synchronization failure. This scheme is shown in Figure 15. Each flip-flop is assumed to have an extra output, \( M_i \), that is asserted while the flip-flop is in a metastable state. Assertion of \( M_i \) delays (pauses) the next clock event. When the flip-flop settles, \( M_i \) is deasserted, and the next clock event is no longer delayed. This can be thought of as synchronizing the system to the input. This scheme reduces the problem of detecting the metastable state.

Philouze, Allen, and Freeman have suggested that a metastable detector can be implemented using a level detection device. Other metastable detectors can be based on delayed outputs, proximity of complementary outputs, and, in CMOS synchronizers, power supply current monitoring.

Figure 15 shows an implementation of a metastable clock. This circuit pauses and restarts the clock without resetting the flip-flop, provided the delay in detecting a metastable state after a sampling clock edge, \( t_{cp} \), satisfies

\[ t_{cp} \geq t_s + t_{setup} \]

where \( t_{setup} \) is the setup time required by \( FF_i \) and \( t_s \) is the positive clock pulse width at most one positive edge on the clock.

Another difficulty is that under a pauseable clock scheme the time needed to complete a processing task is indeterminately extended and failure can be defined in terms of a task finishing beyond a time limit. Comparisons with fixed-clock schemes suggest the pauseable clock scheme is more reliable under certain conditions, but this implicitly assumes that the reliability problems of a pauseable system clock can be overcome.

**SCHMITT TRIGGER SYNCHRONIZER**

Let us analyze two synchronizer circuits (Figures 15 and 16) to determine the probability of synchronization failure in such. Figure 15 represents the simpler synchronizer, one consisting of two D-type flip-flops. This circuit allows the output of \( FF_i \) one clock period in which to settle before it is sampled by \( FF_o \), to form the output of the synchronizer. Figure 16 shows a Schmitt synchronizer. This circuit attempts to improve the performance of the simpler synchronizer by incorporating a Schmitt trigger on the output of \( FF_i \) to filter a

![Figure 15. A simple synchronizer, consisting of two D-type flip-flops.](image1.png)

![Figure 16. A Schmitt synchronizer, consisting of the simple synchronizer with a Schmitt trigger placed between the two flip-flops.](image2.png)
possible metastable state in FF, before that state can affect FF. We will model the metastable behavior of the flip-flops using the first-order model discussed earlier.

Associated with each synchronizer circuit is a range of values of \( V_s \) (the initial voltage sampled on the positive clock edge) that gives rise to failure of the synchronizer. (Note that \( V_s = 0 \) corresponds to the unstable equilibrium point between the '0' and '1' logic levels.) Failure is defined to occur when FF, of each synchronizer, samples a voltage between \(-V_n\) and \(+V_n\). By comparing the range of \( V_s \) that gives rise to failure of the simpler synchronizer with the range of \( V_s \) that gives rise to failure of the Schmitt synchronizer, we can determine the probability of failure for each synchronizer when the asynchronous input has the same electronic properties for each.

Let us first determine the range of \( V_s \) that gives rise to failure of the simple synchronizer. A failure occurs if the output voltage \( V(t) \) of FF, satisfies the relation

\[
-V_n < V(t) < V_n
\]

where \( T \) is the clock period. If we substitute \( V(T) \) for \( V_s \) in Equation 5, Equation 20 becomes

\[
-V_n < V(T) < V_n
\]

From Equation 21, we obtain the maximum size of \( R_s \), the size of the range of values of \( V_s \) that give rise to failure.

\[
R_s = \frac{2V_n}{T}
\]

Now let us determine the range of \( V_s \) that gives rise to failure of the Schmitt trigger synchronizer. For this analysis, we must adopt a model for the behavior of the Schmitt trigger. This model is shown in Figure 17.

The two thresholds of the Schmitt trigger are assumed to be at \( +V_n \) and \( -V_n \). Thus, if the Schmitt trigger has an output of logic '0', then the threshold is \( +V_n \) and if it has an output of logic '1', then the threshold is \( -V_n \). The input output passes through a threshold, the output changes accordingly but is delayed by \( T \), and has a rise time of \( T \), as shown in Figure 17.
The negative edge behavior is assumed to be similar.

Note that the Schmitt trigger is the rectifying device that is used to convert the analog input into a digital output. The Schmitt trigger is designed to have a hysteresis, which means that the trigger point for the input voltage is different from the trigger point for the output voltage. This hysteresis helps to reduce the effects of noise and interference on the output of the Schmitt trigger.

The Schmitt trigger is designed to have a hysteresis of approximately 500 mV. This means that the trigger point for the input voltage is approximately 500 mV lower than the trigger point for the output voltage. This hysteresis helps to reduce the effects of noise and interference on the output of the Schmitt trigger.

The Schmitt trigger is used in a variety of applications, including timing circuits, analog-to-digital converters, and comparator circuits. The Schmitt trigger is a simple and effective way to convert an analog signal into a digital signal.

The Schmitt trigger is designed to have a hysteresis of approximately 500 mV. This means that the trigger point for the input voltage is approximately 500 mV lower than the trigger point for the output voltage. This hysteresis helps to reduce the effects of noise and interference on the output of the Schmitt trigger.

The Schmitt trigger is used in a variety of applications, including timing circuits, analog-to-digital converters, and comparator circuits. The Schmitt trigger is a simple and effective way to convert an analog signal into a digital signal.
A redundant synchronizer with high n will be more reliable than one with low n. However, we have determined that for any nontrivial combinational function satisfying certain reasonable properties, the general redundant system—at any value of n—cannot improve on the simple synchronizer in reducing the probability of synchronization failure due to metastable behavior.

Metastable behavior in digital circuits is unavoidable and produces a dramatic range of failure rates. Designers should give special attention to it, particularly if they are attempting to build highly reliable systems. They should develop techniques for accurately predicting system reliability and should exploit techniques for reducing the probability of synchronization failure due to metastable behavior.

Figure 19.
A synchronizer that uses redundant flip-flops and voting among their outputs to reduce the probability of failure.

Figure 20.
Generalized structure of synchronizers that use clock phasing, redundant flip-flops, and voting.
Acknowledgments

We thank the Computer Society's reviewers and editors for their comments and suggestions. We also acknowledge the support of our work, and of the writing of this article, by the Australian Computer Research Board.

References


Lindsay Klemm has been a member of the IEEE since 1960. Klemm earned a degree in electrical engineering in 1953 and a degree in mathematics in 1963, both with first-class honors and university medals, from the University of Western Australia. In 1967, he received the Ph.D. also from the University of Western Australia, in 1967. He is a member of the IEEE.

For Further Reading


Antonio Cantor is a project director for the COTS Communications Project, Ltd., Perth, Western Australia, and is also a visiting professor of electrical engineering at the University of Western Australia, Perth, Australia. Previously, he was a professor of computer science at the University of Western Australia, where he taught courses in computer science and computer engineering. He is a member of the IEEE.

Antonio Cantor is a project director for the COTS Communications Project, Ltd., Perth, Western Australia, and is also a visiting professor of electrical engineering at the University of Western Australia, Perth, Australia. Previously, he was a professor of computer science at the University of Western Australia, where he taught courses in computer science and computer engineering. He is a member of the IEEE.

Questions about the article can be directed to Cantor at the Department of Electrical and Computer Engineering, University of Western Australia, Nedlands, Western Australia 6009.