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.

0740-7475/87/1200-04$1.00©1987

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

Many 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 system design; 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 multiport memory employs an arbiter circuit to ensure that only one processor at a time accesses it.

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 the 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 clock cycle, 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 constraints required by the n flip-flops, namely the setup and hold time requirements, are satisfied when the following relations hold:

$$t_{setup} \leq T_c - (t_{cmax} + t_{pdmax}) \quad (1)$$

$$t_{hold} \leq t_{cmin} + t_{pdmin} \quad (2)$$

where T_c is the system clock period, t_{cmax} is the maximum combinational circuit delay, t_{cmin} is the minimum combinational circuit delay, t_{pdmax} is the maximum flip-flop propagation delay, t_{pdmin} is the minimum flip-flop propagation delay, t_{setup} is the setup time required by the flip-flops, and t_{hold} is the hold time required by the flip-flops. One must ensure that the clock period is long enough to enable the setup time condition (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

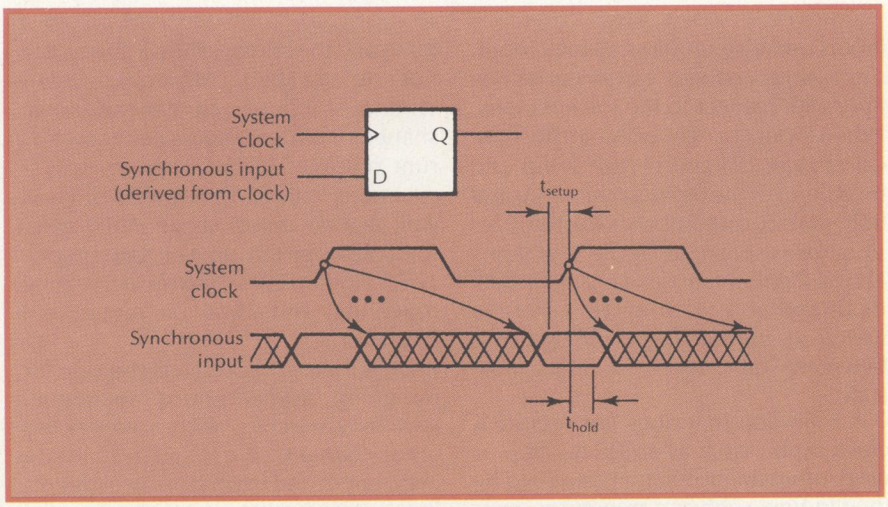


Figure 1. Setup and hold time conditions satisfied by a synchronous input.

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

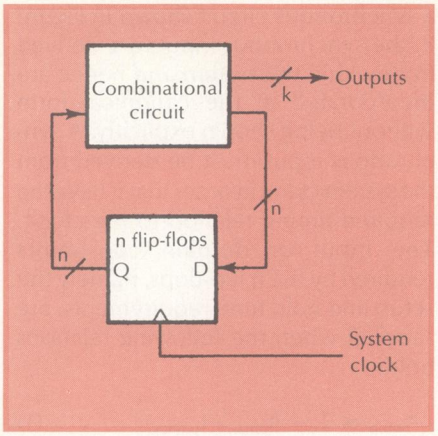


Figure 2. Synchronous circuit (canonical form). The synchronous inputs are derived from within the circuit.

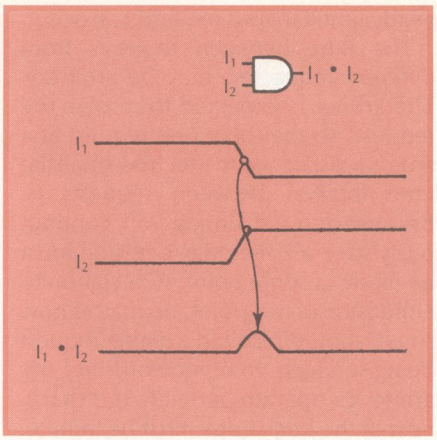


Figure 4. Runt pulse generated from two well-formed inputs to an AND gate.

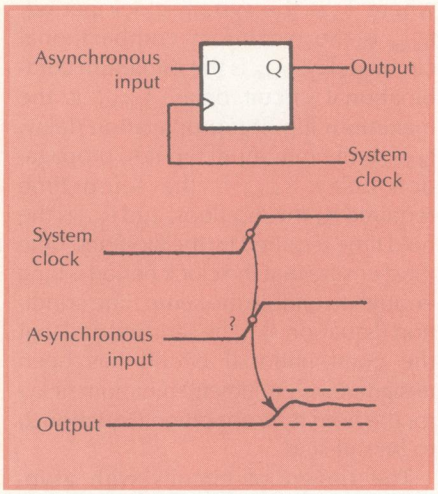


Figure 3. D-type flip-flop sampling a changing data input. The changing input causes the flip-flop to enter a metastable state—that is, a state in which its output oscillates between logic levels.

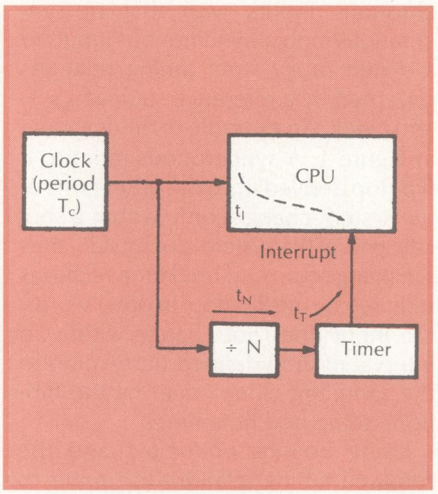


Figure 5. Timer employing a divided clock derived from the CPU clock. The timer may generate interrupts that are inadvertently mistimed so as to always occur in the vicinity of the interrupt sampling instant.

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.[1,2] 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.

Unger has proposed a technique for designing asynchronous sequential switching circuits with unrestricted input changes.[3] It uses a special state assignment and inertial delay elements in the state branches. An ideal inertial delay only passes pulses greater than a set width. However, all available inertial delay elements require input constraints in order to perform as required. The input constraints may be violated by asynchronous inputs, as demonstrated by Marino.[4]

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 nonrandom way, marginal triggering of devices. For example, a timer employing a divided clock derived from the CPU clock may generate inadvertently mistimed 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_I, the clock divider introduces a delay of t_N, and the timer asserts its output, a delay t_T, after a positive clock edge. Marginal triggering of the interrupt sampling mechanism within the CPU can occur if

$$t_N + t_T \cong t_I + kT_c \qquad (3)$$

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.

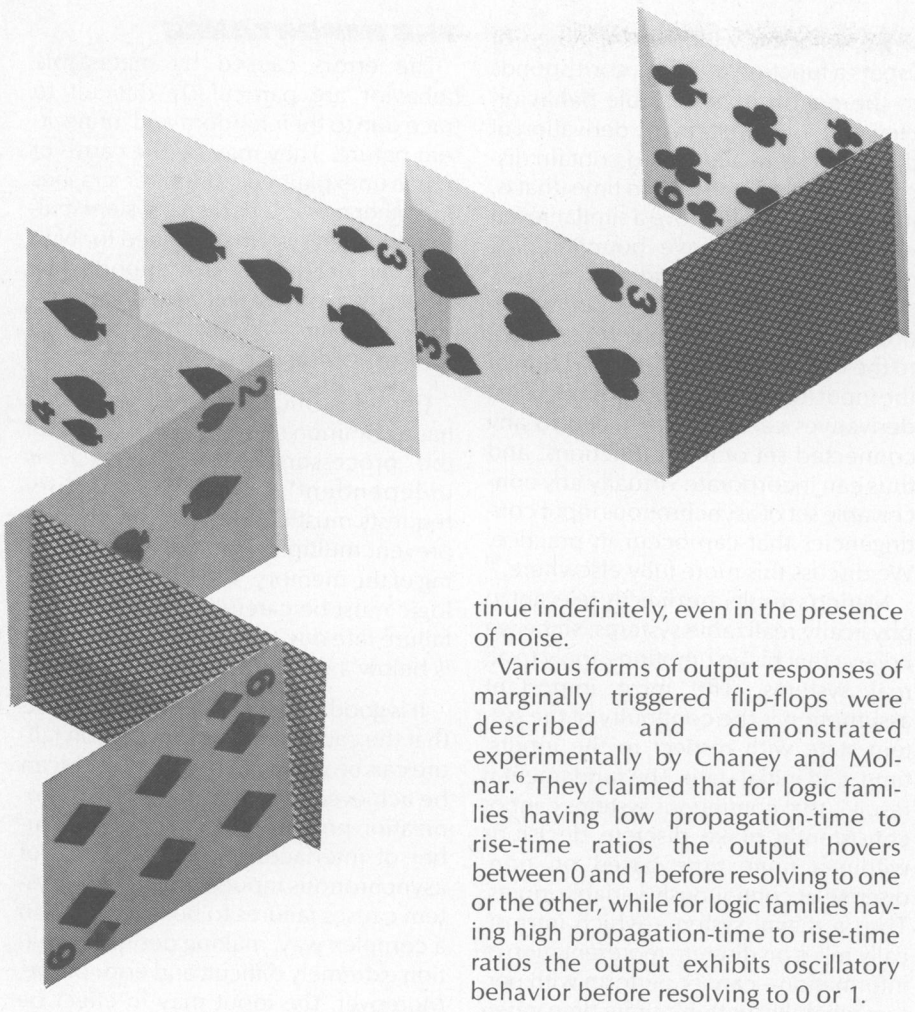
Metastability can also occur in a fully synchronous system when power supply disturbances occur.[5] The disturbances act to extend circuit delays, and the extended delays can result in violation of timing constraints, especially in circuits with tight timing tolerances.

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

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.



acceptable maximum delay is effectively determined by the components of a logic family and by the appropriately specified setup and hold times. Marginal triggering can occur in a D-type flip-flop when the data input changes near the sampling clock edge, or in an RS-type flip-flop when reset and set pulses are released almost simultaneously or contain runt pulses.

Under marginal triggering conditions a flip-flop can enter a metastable state, meaning the flip-flop can be in the vicinity of an unstable equilibrium state, with its output lingering between the 0 and 1 voltage levels. A displacement about an unstable equilibrium brings about forces that tend to destroy the equilibrium rather than restore it. However, unbiased noise has a negligible statistical effect on the duration of a metastable state in a flip-flop.

Catt demonstrated qualitatively that an unstable equilibrium state exists in a bistable flip-flop and can be induced by a critical marginal triggering of the flip-flop.[2] Moreover, he claimed that a metastable state in a flip-flop can continue indefinitely, even in the presence of noise.

Various forms of output responses of marginally triggered flip-flops were described and demonstrated experimentally by Chaney and Molnar.[6] They claimed that for logic families having low propagation-time to rise-time ratios the output hovers between 0 and 1 before resolving to one or the other, while for logic families having high propagation-time to rise-time ratios the output exhibits oscillatory behavior before resolving to 0 or 1.

THE UNAVOIDABILITY

A common approach to designing a sequential circuit with asynchronous inputs is to use a synchronous sequential circuit and place an interface between it and the asynchronous inputs. The interface changes the asynchronous inputs to synchronous inputs by sampling them with the system clock. This sampling process is known as synchronization and the interface is called a synchronizer. The problem of designing a perfect synchronizer—one not subject to metastable behavior—is equivalent to the problem of designing a perfect circuit with asynchronous inputs, since each can be built from the other. Considerable academic research has been done on the problem of designing reliable synchronizers and estimating their probability of failure due to metastable behavior.[4,6-15] The problem has also been recognized by component manufacturers, and they have given special consideration to the synchronization of asynchronous inputs.[16,17]

Attempts have been made to design a perfect synchronizer.[7,8] However, careful examination reveals that these designs are indeed subject to failure caused by metastable behavior—or synchronization failure, as we will henceforth call it. This is particularly evident in the analyses done by Chaney and Molnar.[6] 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. Marino has shown that any system having certain basic properties—which we will describe below—cannot avoid exhibiting metastable behavior.[9] 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 input 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 considered 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 under "idling" inputs, and also an 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 set state while the other corresponds to the reset state, with the idling inputs corresponding to the R and S inputs low (inactive). The term stable is 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.

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

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 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 idling inputs are applied during the settling time.

The settling time, T, can be arbitrarily large, with the width of the range of inputs a function of T. This 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 connected set of input functions, and thus can incorporate virtually any conceivable set of asynchronous input contingencies that can occur in practice. We discuss this more fully elsewhere.[10]

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 nondiscrete or continuously variable inputs. Thus a 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 unavoidable, it is not sufficient just for the 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 at least a minimum time can be designed to operate without metastable failure, since the system state is not dependent on the timing or asynchronism of the input.

However, if the restriction of a minimum separation between input edges is lifted, then two edges very close together will not be seen by the circuit and two edges much further apart will count as two. In this situation metastable behavior is unavoidable due to the continuum of edge interarrival times possible between no effect and two counts.

Glossary of symbols

Δt	Metastable aperture width
f_c	Clock frequency
FF	Flip-flop
h	Lower bound on settling time for validity of exponential model of metastable behavior
λ	Mean data edge rate of exponential probability distribution
MFR	Mean failure rate
MTBF	Mean time between failures
N	Delay in synchronizer (integer number of clock periods)
prob(A)	Probability of event A
R_1, R_2	Ranges of v_0 that give rise to failure
t	General time variable
t′, T	Flip-flop settling time
T_0	Time parameter for aperture width model
T_c, t_c	Clock period
t_{hold}	Hold time for data input to clocked flip-flop
t_r	Rise/fall time of Schmitt trigger
t_s	Propagation delay of Schmitt trigger
t_{setup}	Setup time for data input to clocked flip-flop
τ	Device time constant for exponential escape from metastable region of flip-flop
v_0	Flip-flop sampled initial voltage with respect to unstable equilibrium point
v_1	Flip-flop output voltage with respect to unstable equilibrium point
v_d	Boundary voltage of linear operation of flip-flop with respect to unstable equilibrium point

THE IMPORTANCE

The errors caused by metastable behavior are particularly difficult to trace due to their random and intermittent nature. They may be the cause of many unexplained computer crashes and other mysterious digital system malfunctions.[6] In systems designed for high reliability, it is particularly important to reduce this form of failure to an acceptable level and identify the locations where it can occur.

Consider a multiprocessor system that has a common memory accessed by all the processors, each of which is independently clocked. Memory requests must be reliably arbitrated to prevent multiple simultaneous accessing of the memory. Thus, the arbitration logic must be carefully designed so its failure rate due to metastable behavior is below a maximum tolerable value.

It is good practice to design circuits so that the cause of a synchronization failure can be readily determined. This can be achieved by confining the synchronization processes to a minimum number of interfaces. The ad hoc use of asynchronous inputs throughout a system causes failures to be distributed in a complex way, making design evaluation extremely difficult and error prone. Moreover, the input may in effect be synchronized more than once in the system, increasing the probability of failure.

Even without metastable failure, an input may be interpreted differently by different parts of the system because of clock skew effects, thereby causing inconsistencies within the logic. Hence, synchronization must be done before any decisions are made using the input.

It is important that independent inputs be synchronized if possible. Otherwise, relations assumed to hold between variables before synchronization may be destroyed by the synchronization process, and thus inconsistencies may occur in later processing of the inputs.

If two asynchronous inputs are complementary, for example, then after a marginal or near-marginal synchronization event they may no longer be complementary because their sampling times may have become slightly different when they were changing. For this simple problem, the solution is to synchronize one input only and derive the

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

other synchronized input from it using an inverter.

Unnecessary synchronization can also be avoided by masking the appropriate inputs with state variables when the next state transition does not depend on the variables being masked.[11]

Dependencies arise naturally among multibit asynchronous inputs such as those from analog-to-digital converters or parallel input data lines. The synchronization of multibit asynchronous inputs can be achieved in several ways:

- By coding so that only one bit changes at a time (that is, by using a Gray code).
- By designing the asynchronous source of the multibit data to provide handshaking with a single additional output signal to indicate "data valid." The data valid bit must be inactive for an appropriate setup and hold time window around changes in the asynchronous multibit data. The data and data valid bit are synchronized (a device hardened against metastability is used for the data valid synchronization, of course), and the system uses only synchronized data with an active synchronized "data valid."
- By synchronizing the device to the system so it is not asynchronous! For example, one can make an A/D converter provide synchronous data by ensuring appropriate triggering from the system clock and accounting for all delays.

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.

FAILURE PROBABILITY

The analysis of failure rates of flip-flops used as synchronizers has been the subject of several papers.[1,12,13] The basic approach is to consider the behavior of the flip-flop around its metastable state or unstable equilibrium point, where the flip-flop state is in its linear region. When the state leaves this region, the flip-flop is assumed to quickly resolve its output to either stable logic state, and hence the nonlinear behavior can be assumed to be of little consequence to the metastable behavior.

FIRST-ORDER MODEL

We use the first-order model of a flip-flop proposed by Veendrick[13] in this analysis. It is shown in Figure 6. Voltages v_{01} and v_{02} are assumed to be generated within the flip-flop at time t=0 by the sampling process; v_1 represents the output of the flip-flop. The solution to the differential equations for the circuit, for v_1, is

$$v_1 = \frac{(v_{01}-v_{02})}{2}\cdot e^{\frac{(A-1)}{RC}t} + \frac{(v_{01}+v_{02})}{2}\cdot e^{-\frac{(A+1)}{RC}t} \quad (4)$$

Neglecting the decaying exponential term of Equation 4 and letting $v_0=(v_{01}-v_{02})/2$ and $\tau=RC/(A-1)$, we obtain

$$v_1 = v_0\, e^{\frac{t}{\tau}} \quad (5)$$

For marginal triggering conditions v_0 is assumed to be uniformly distributed between $-v_d$ and $+v_d$, where v_d is the "boundary" of the flip-flop's linear region of operation. Beyond $\pm v_d$ the output is assumed to be defined as one of the two logic levels. The assumption that v_0 is uniformly distributed can be made because the clock edge samples a uniformly rising data input generating v_0, as shown in Figure 7.

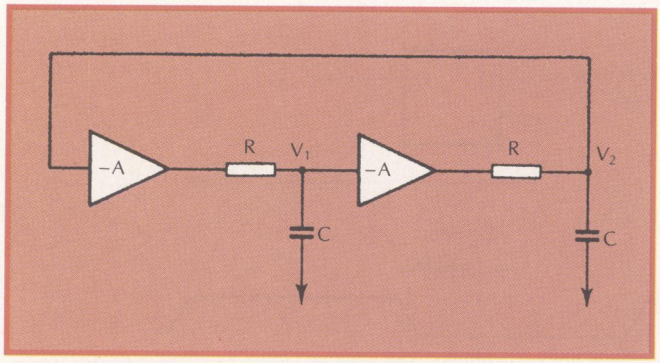


Figure 6. First-order model of a flip-flop.

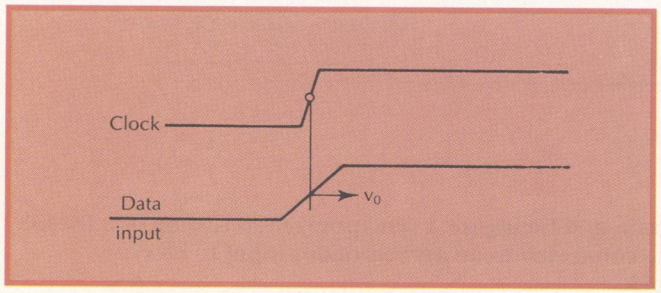


Figure 7. Clock edge sampling a uniformly rising data input. The sampling generates v_0.

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

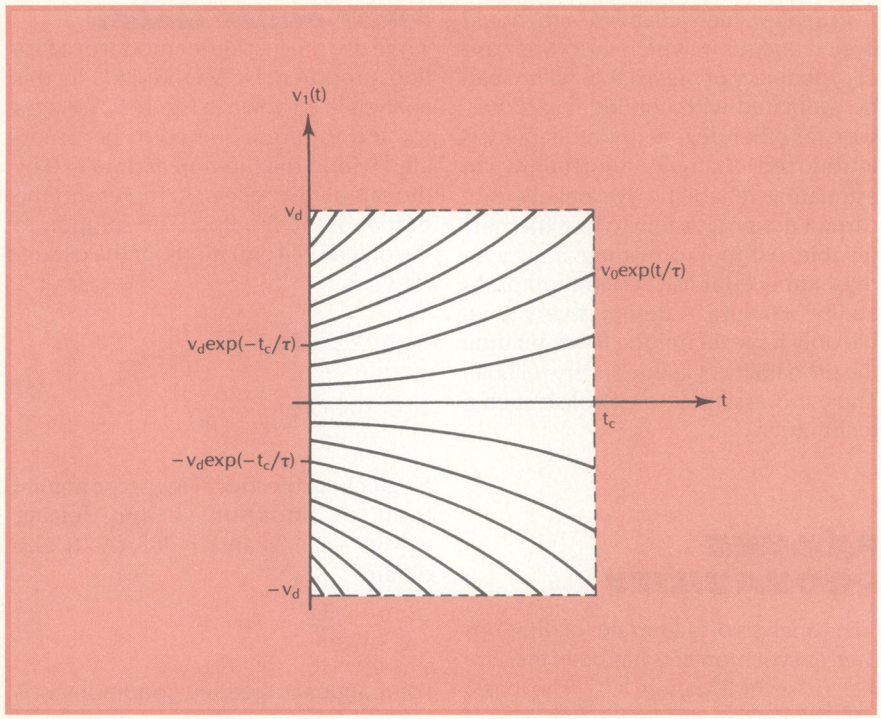


Figure 8. Voltage at t_c monotonically mapped back to its initial voltage.

The probability of the metastable behavior lasting longer than t_c, $P(t>t_c)$, is the probability that $|v_1|<v_d$ at $t=t_c$. The voltage at t_c 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

$$\begin{aligned}&\text{Prob}\,(|v_1(t_c)|<v_d)\\&=\text{prob}\,(|v_0|<v_d e^{-\frac{t_c}{\tau}}\end{aligned}\qquad(6)$$

that is,

$$P(t>t_c)=e^{-\frac{t_c}{\tau}}$$

by uniform distribution of v_0.

Because the voltages at time t are always uniformly distributed when between $\pm v_d$, superimposed small-amplitude unbiased noise on v_1 has a negligible effect on the distribution of v_1. Consider two adjacent voltage regions. The number of states forced out of one 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 Couranz and Wann[1] and Veendrick.[13]

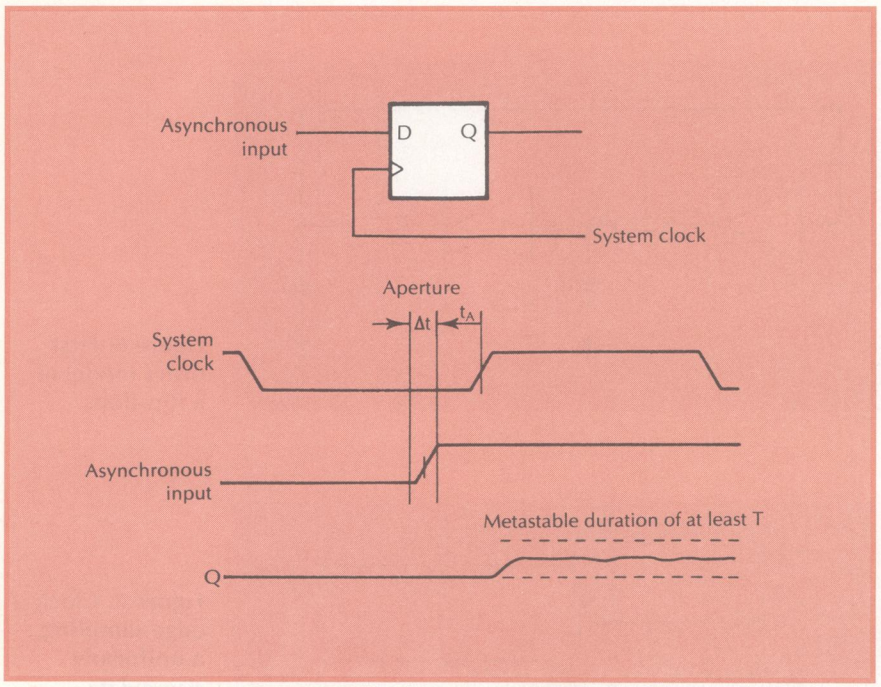


Figure 9. Block diagram and timing for a synchronizer consisting of a D-type flip-flop with its data connected to an asynchronous input to be synchronized by the clock.

EXPONENTIAL MODEL

Experimental measurements made on flip-flops[13-15] 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 Chaney[14] and Chaney.[15] 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 set pulse and the completion of the reset pulse is t_d, and t_d has a uniform probability density $p(t_d)$ over an interval $t_a \le t_d \le t_b$. Then

$$p(t_d)=\begin{cases}0 & t_d<t_a\\ 1/(t_b-t_a)=1/\delta & t_a\le t_d\le t_b\\ 0 & t_d>t_b\end{cases}$$

where the flip-flop is set by $t_d=t_a$ and reset by $t_d=t_b$ within a normal propagation delay.

For t_d uniformly distributed over this interval, $F(t')$ is defined as the probabil-

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

ity that the flip-flop output has not reached a logically defined and stable value at a time t′ after triggering. Experiments have shown that for $t' > h$ this probability can be represented as

$$F(t') = (T_0/\delta)\, e^{-\frac{t'}{\tau}} \tag{7}$$

where $\delta = t_b - t_a$ and the parameters τ and T_0 depend on the particular flip-flop.

The parameter h is determined experimentally by using data to determine the minimum value of settling times $t'(=h)$; this is done so the parameters in Equation 7 will be constant.

Various values for the parameters of Equation 7 are given by Chaney[15] for available flip-flops. In Chaney's work we can see that there is a large variation in TAU (equivalent to τ in our discussion) and T_0 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 unresolved (MTSU). 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 testing 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[14] can be employed.

We should also note in Veendrick[13] and Chaney[15] the extreme variation in failure rate with respect to the settling time t′. Doubling the settling time from 20 ns to 40 ns results in an improvement of more than ten orders of magnitude in MTSU 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).

For every clock event (0-to-1 clock transition) there is a region of time called an aperture. If a data edge occurs within the aperture, the output of the flip-flop remains unresolved longer than a time T. The width of the aperture, Δt, is a function of T and the parameters of the flip-flop (the parameters used in Equation 7). Thus,

$$\Delta t = T_0\, e^{-\frac{T}{\tau}} \tag{8}$$

for $T > h$.

The exact location of the aperture with respect to the 0-to-1 clock edge, t_A or $-t_A$, is not important, but its width is, since the width determines the probability of an occurrence of a data edge causing metastable behavior. The edges of the aperture may not be as sharply defined in real flip-flops as they are here, but our model is a useful approximation of the real situation and agrees with experimental measurements of flip-flop failure rates.

To evaluate the failure rate of our synchronizer, we begin by assuming that data edges occur randomly in time with an exponential distribution of time between data edges. That is, we assume that

$$\text{prob(no data edge in } [t_1, t_2]) = e^{-(t_2 - t_1)\lambda} \tag{9}$$

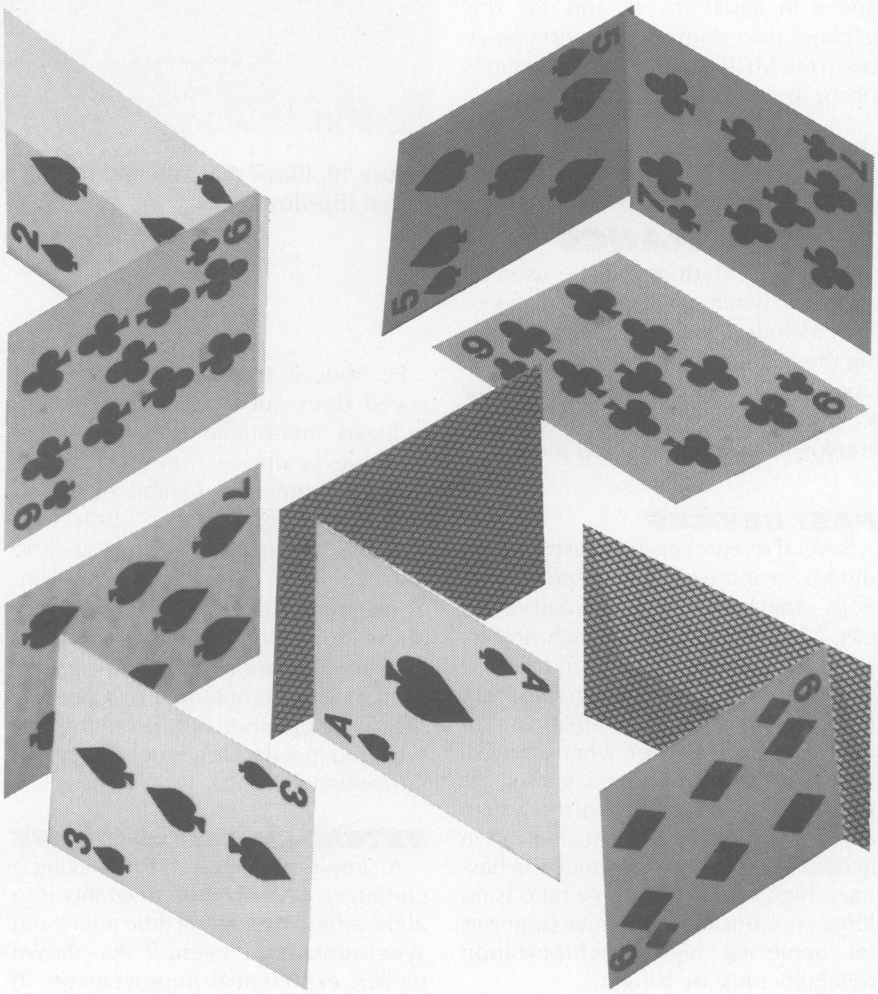
where λ is the mean data edge rate of the exponential probability distribution.

Hence the probability p_{nf} of the flip-flop not failing (that is, of it settling within T) as a result of a synchronization event is given by

$$p_{nf} = e^{-\lambda \Delta t} \tag{10}$$

If we assume a constant clock period of T_c and frequency f_c $(=1/T_c)$, and we also assume that clock events are independent, it follows that

$$\text{prob(no failure in } [0,t]) \cong (e^{-\lambda \Delta t})^{\frac{t}{T_c}} \tag{11}$$



Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

where the approximation is due to t not always being an exact multiple of T_c. For $t >> T_c$ this can be neglected, giving

$$\text{prob(no failure in } [0,t]) = e^{-\lambda \Delta t f_c} \quad (12)$$

where $f_c = 1/T_c$. Equation 12 shows that the occurrence of synchronization failure is a Poisson process—that is, that the interarrival times of failures are exponentially distributed. We can now calculate the mean failure rate, MFR, and the mean time between failures, MTBF, where MTBF=1/MFR:

$$MFR = \lambda f_c T_0 e^{-\frac{T}{\tau}} \quad (13)$$

$$MTBF = \frac{T_c e^{\frac{T}{\tau}}}{\lambda T_0} \quad (14)$$

Recall that T is the time interval after the sampling clock edge during which the synchronizer can resolve, and τ is a synchronizer device time constant. Note the exponential dependence on T and τ in Equations 13 and 14. This explains the enormous variation possible in the MTBF when modest variations occur in τ or T.

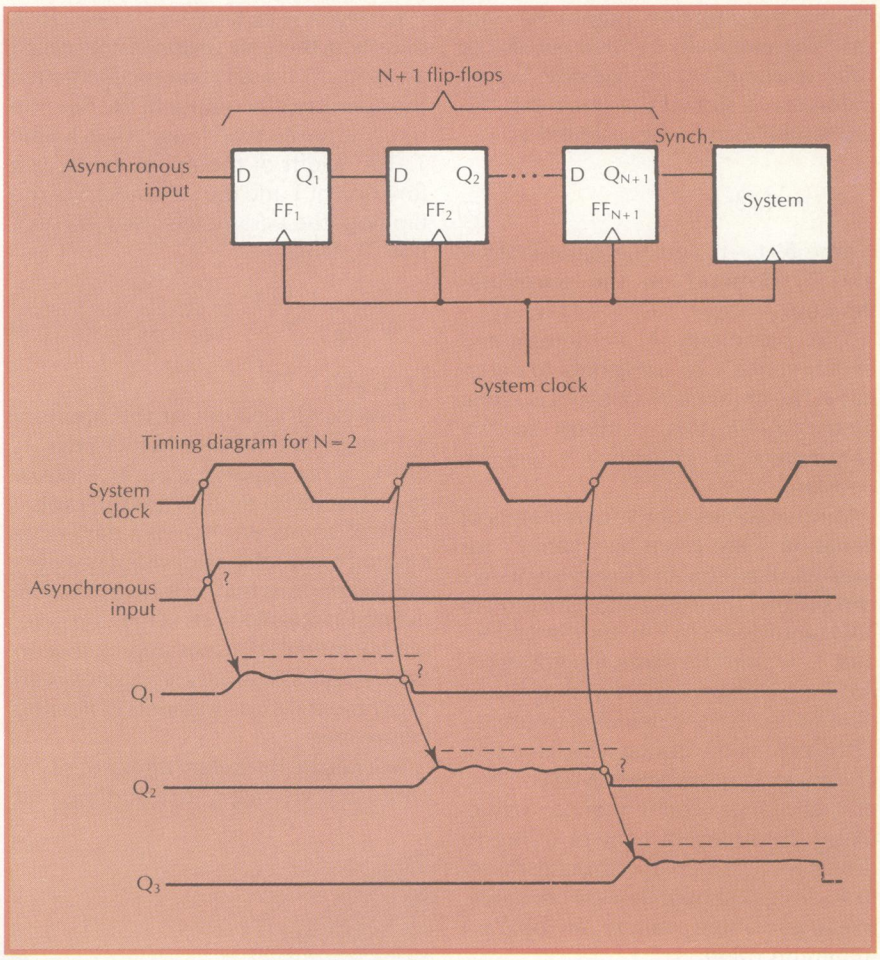


Figure 10. Block diagram and timing for a synchronizer consisting of cascaded flip-flops.

IMPROVING PERFORMANCE

Several techniques can be used to improve synchronizer performance. They include using fast devices, extending the settling time, using a pausable clock and metastable detection, using a Schmitt trigger synchronizer, 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 τ.[18] 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 timings scaled down accordingly. Thus, the advantage in having a high settling-time-to-τ ratio is no longer present and alternative strategies for achieving high synchronization reliability must be sought.

Pechoucek suggests that improved speed does not necessarily result in reduced metastable durations.[12] For example, a device may have a fast switching time yet exhibit extended oscillatory metastable behavior because of underdamping in the device's linear region of operation. Thus, greater device speed does not always correspond to a low probability of synchronization failure. Test results similar to those obtained by Chaney[15] should be obtained before confidence is placed in a device's synchronization performance.

EXTENDED SETTLING TIME

A simple technique for achieving a certain synchronization reliability is to allow adequate settling time after each synchronization event.[11] As shown earlier, exponential improvements in metastable reliability can be gained by increasing the ratio of the settling time to τ, 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 marginal 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−1) clock periods between FF_1 and FF_2 and an additional clock period between FF_2 and FF_3. The function of FF_3 is to elimi-

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

nate 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 transferring of metastable behavior in cascaded flip-flops. We assume that the clock edge sampling process transfers the internal state of FF_i to FF_{i+1} 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_3 in Figure 11 and FF_{N+1} 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_1, where

$$T_1 = NT_c - (N-1)t_p \tag{15}$$

and T_c is the system clock period.

Using T_1 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, $MTBF_1$, where

$$MTBF_1 = \frac{e^{\frac{T_1}{\tau}}}{\lambda T_0} = \frac{e^{\frac{NT_c}{\tau}}}{\lambda T_0\, e^{\frac{(N-1)t_p}{\tau}}} \tag{16}$$

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_2, where

$$T_2 = NT_c - t_p - t_d \tag{17}$$

Note that for the divided clock synchronizer, the input sampling period is no longer T_c but $(N-1)T_c$. Thus, from Equations 14 and 17, for the divided clock synchronizer,

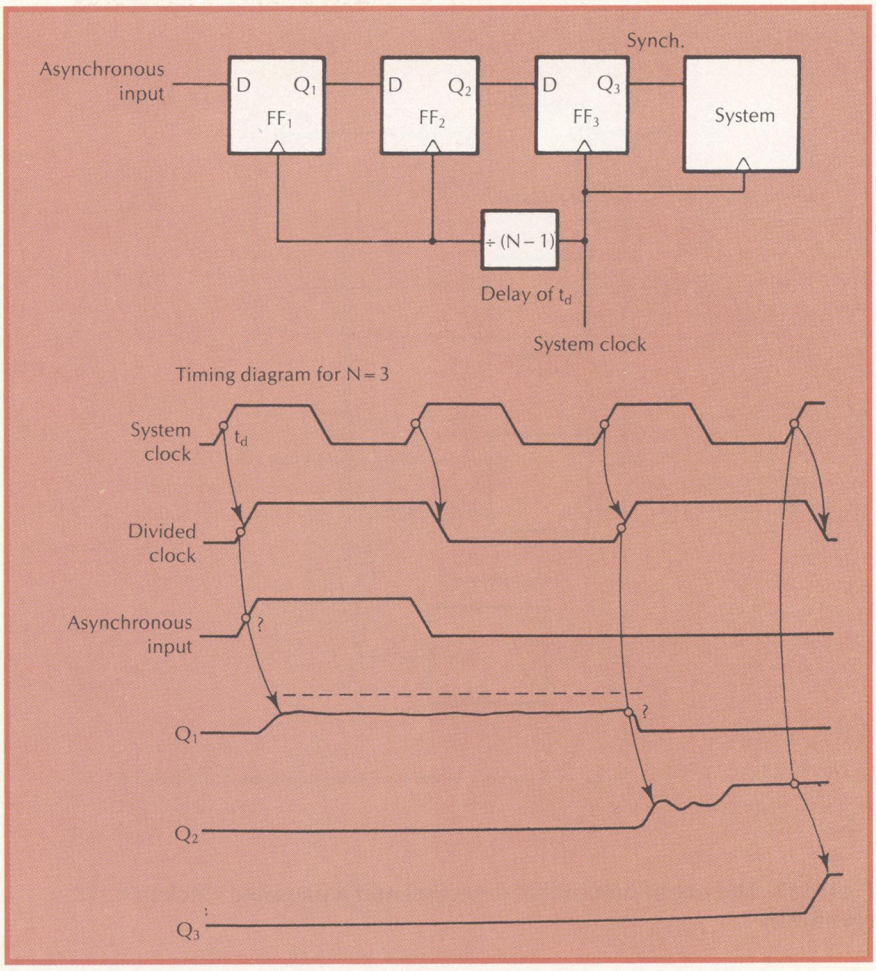


Figure 11. Block diagram and timing for a synchronizer consisting of cascaded flip-flops used with a divided clock.

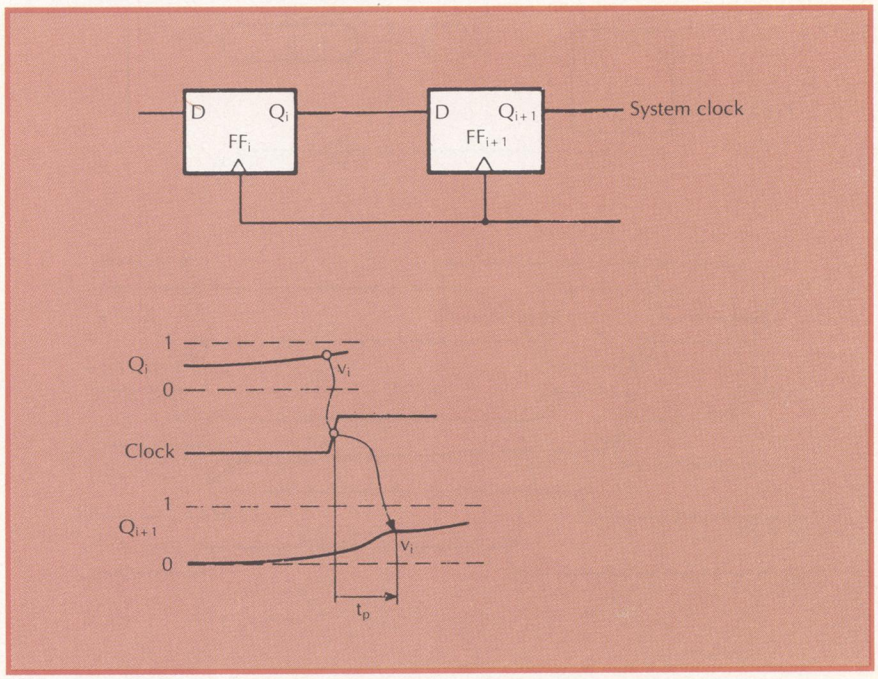


Figure 12. Model of the transferal of metastable behavior in cascaded flip-flops.

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

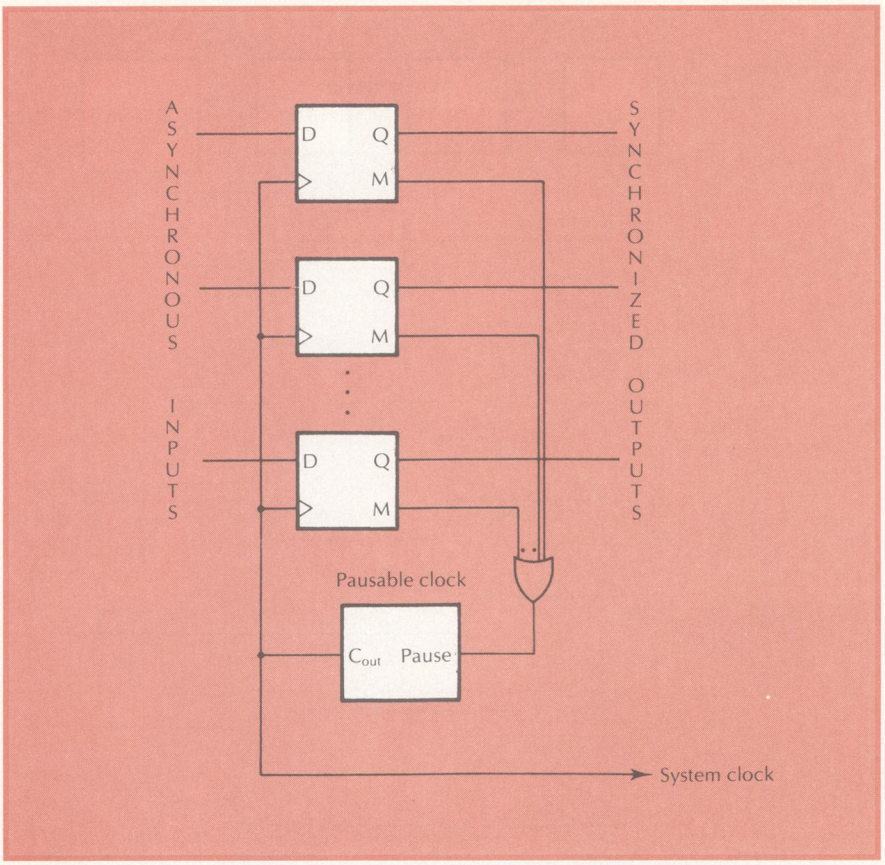


Figure 13. The use of metastable detectors and a pausable clock in a synchronizer.

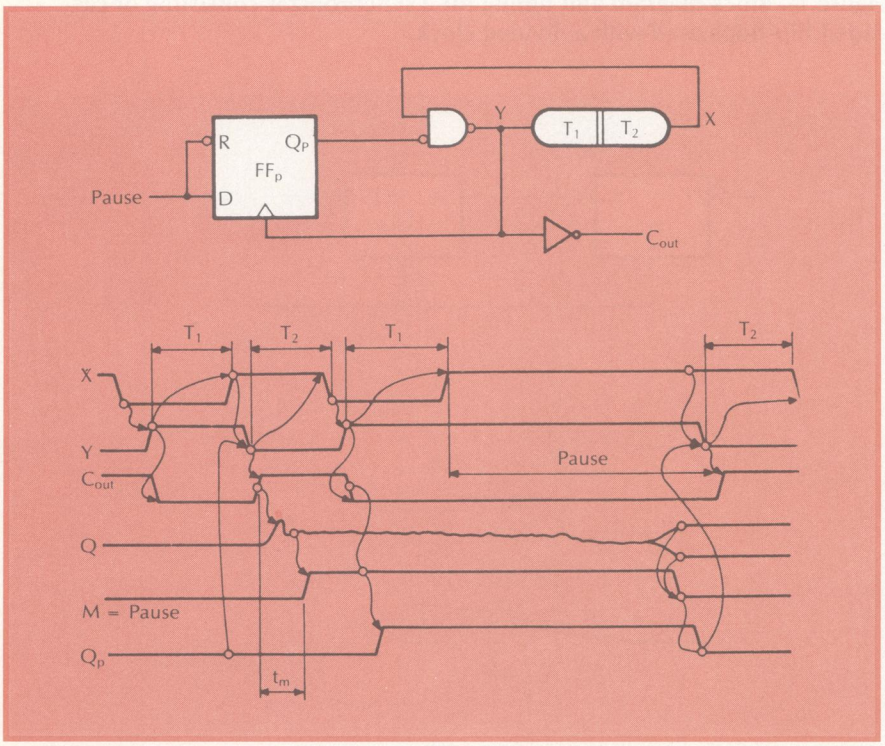


Figure 14. Block diagram and timing of an implementation of a pausable clock.

$$MTBF_2 = (N-1)T_c \frac{e^{\frac{T_2}{\tau}}}{\lambda T_0} = \frac{(N-1)T_c e^{\frac{NT_c}{\tau}}}{\lambda T_0 e^{\frac{t_p + t_d}{\tau}}} \quad (18)$$

And from Equations 16 and 18,

$$\frac{MTBF_2}{MTBF_1} = (N-1)\frac{e^{\frac{(N-2)t_p}{\tau}}}{e^{\frac{t_d}{\tau}}} \quad (19)$$

For $N > 2$ and $t_p \cong t_d$, Equation 19 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 it to miss information that the cascaded flip-flop scheme would capture.

Comparing Equations 16 and 18 with Equation 14, we see that increasing the settling time of a synchronizer can result in an exponential improvement in metastable reliability. The drawback is the introduced delay, which can 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 will produce synchronizing flip-flops with guaranteed maximum values of the critical device time constant, τ, as opposed to the highly uncertain values provided by existing devices.[15] Devices are being produced that are claimed to be metastable-hardened—that is, to have small, but unspecified τ values.

Example: Let us say that a microprogrammed system has a 40-ns clock period and that an asynchronous input with mean time between edges of 200 ns has an exponential distribution of edge interarrival times. The input is synchronized to the system by means of flip-flops with $\tau = 2$ ns and $T_0 = 20$ μs (a not unlikely worst case[15]).

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

Suppose a simple synchronizer—which has one clock period of settling time—is used. Applying Equation 14, we obtain an MTBF for it of 194 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_p 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, also with three clock periods of settling time and with a divider delay of 10 ns, would yield (from Equation 18) an MTBF of 132,000 years. At the latter two levels of reliability, the difference in performance is purely academic.

METASTABLE DETECTION

The use of metastable detectors and a clock that can be paused has been suggested as a way to reduce the probability of synchronization failure.[12,19,20] This scheme is shown in Figure 13.

Each flip-flop is assumed to have an extra output, M, that is asserted while the flip-flop is in a metastable state. Assertion of M delays (pauses) the next clock event. When the flip-flop settles, M 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 requires a means of detecting the metastable state.

Pechoucek,[12] Adams et al.,[21] and Freeman et al.[22] have suggested that a metastable detector can be implemented using a level detection device. Other metastable detectors can be based on delayed outputs,[23] proximity of complementary outputs,[20] and, in CMOS synchronizers, power supply current monitoring.[22]

Figure 14 shows an implementation of a pausable clock. This circuit pauses and restarts the clock without runt pulses, provided the delay in detecting a metastable state after a sampling clock edge, t_M, satisfies

$$t_M + t_{setup} \leq T_2$$

where t_{setup} is the setup time required by FF_p and T_2 is the positive clock pulse width. At most one positive edge on the pause input is allowed to occur after each synchronization.

The pausable clock scheme has several disadvantages. First, some systems rely on a constant clock rate for correct operation and hence cannot use a pausable clock. Second, it may not be possible to design metastable detectors that are not subject to runt pulses on the output when oscillatory metastable behavior occurs. For example, a simple threshold device may fail under oscillatory metastable behavior; examples of such failure can be found in Chaney and Molnar's work.[6] And even if the metastable behavior is nonoscillatory, the reduced noise margin caused by metastable detectors may be a source of unreliability.

The problem of spurious outputs from the metastable detector is particularly serious because the performance of the pausable system clock is directly dependent on the metastable detector's outputs. The failure of the system clock is more serious than the failure of a single synchronizer because the effects of a system clock failure are more widespread.

Another difficulty is that under a pausable clock scheme the time needed to complete a processing task is indeterminantly extended and failure can be defined in terms of a task finishing beyond a time limit. Comparisons with fixed-clock schemes[19] suggest the pausable clock scheme is more reliable under certain conditions, but this implicitly assumes that the reliability problems of a pausable 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 each. Figure 15 represents the simpler synchronizer, one consisting of two D-type flip-flops. This circuit allows the output of FF_1 one clock period in which to settle before it is sampled by FF_2 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_1 to filter a

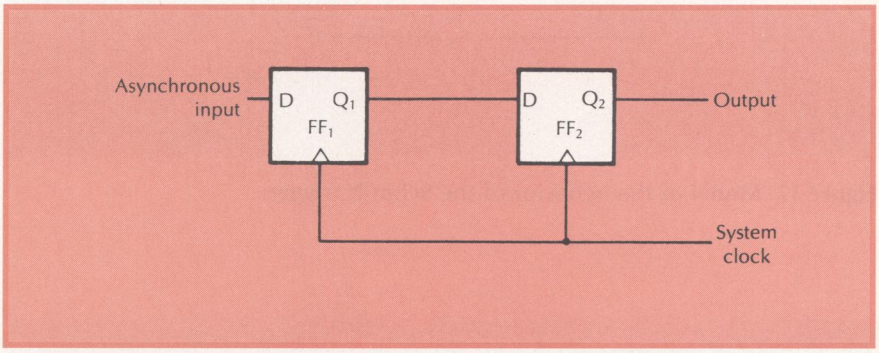


Figure 15. A simple synchronizer, consisting of two D-type flip-flops.

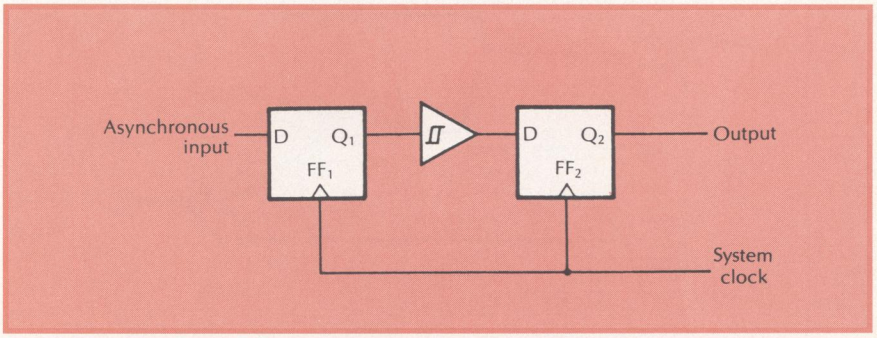
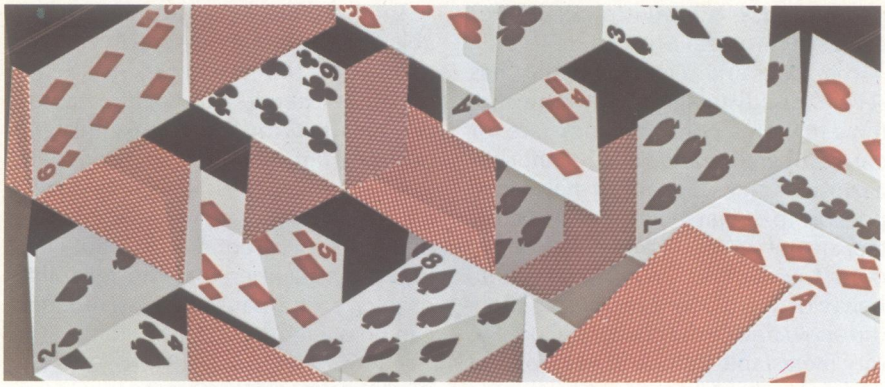


Figure 16. A Schmitt synchronizer, consisting of the simple synchronizer with a Schmitt trigger placed between the two flip-flops.

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.



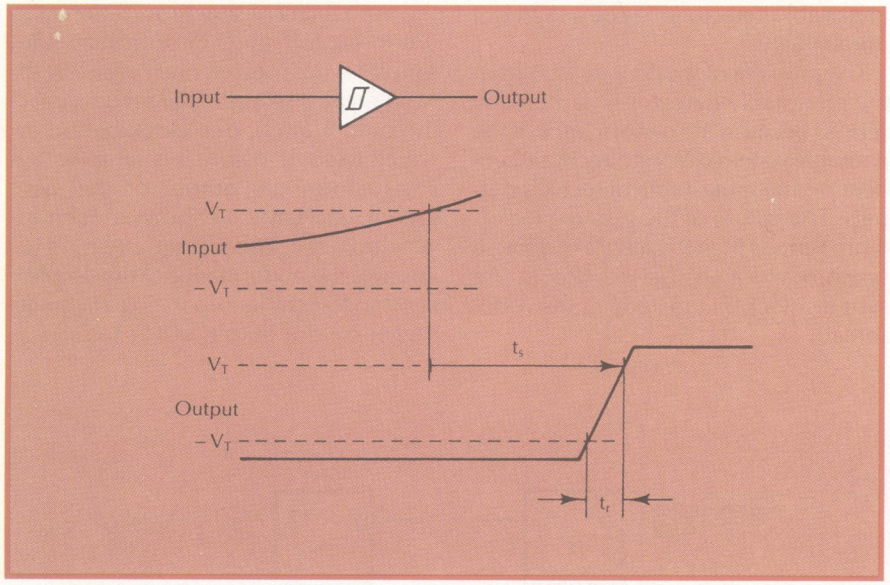


Figure 17. Model of the behavior of the Schmitt trigger.

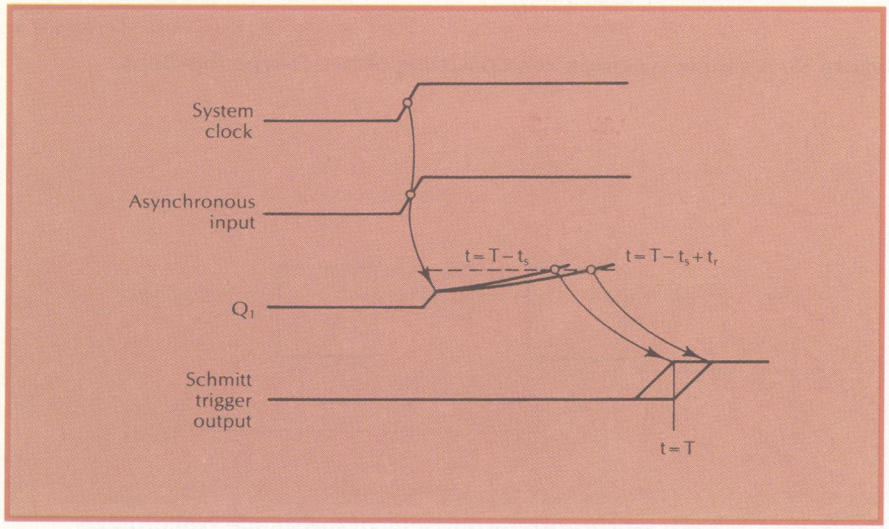


Figure 18. Failure of the Schmitt synchronizer.

possible metastable state in FF_1 before that state can affect FF_2.

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_0 (the initial voltage sampled on the positive clock edge) that give rise to failure of the synchronizer. (Note that $v_0 = 0$ corresponds to the unstable equilibrium point between the 0 and 1 logic levels.) Failure is defined to occur when FF_2 of each synchronizer samples a voltage between $-v_d$ and $+v_d$. By comparing the range of v_0 that gives rise to failure of the simpler synchronizer with the range of v_0 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 stochastic properties for each.

Let us first determine the range of v_0 that gives rise to failure of the simple synchronizer. A failure occurs if the output voltage, $v(t)$, of FF_1 satisfies the relation

$$-v_d < v(T) < v_d \tag{20}$$

where T is the clock period. If we substitute $v(T)$ for v_1 in Equation 5, Equation 20 becomes

$$-v_d e^{-\frac{T}{\tau}} < v_0 < v_d e^{-\frac{T}{\tau}} \tag{21}$$

From Equation 21, we obtain for the simple synchronizer R_1, the size of the range of values of v_0 that give rise to failure:

$$R_1 = 2v_d e^{-\frac{T}{\tau}} \tag{22}$$

Now let us determine the range of v_0 that gives rise to failure of the Schmitt 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 $+v_d$ and $-v_d$. That is, if the Schmitt trigger has an output of logic 0, then the threshold is v_d, and if it has an output of logic 1, then the threshold is $-v_d$. Once the input passes through a threshold, the output changes accordingly but is delayed by t_s and has a rise time of t_r, as shown in Figure 17.

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

The negative edge behavior is assumed to be similar.

Note that nothing is assumed about the behavior of the Schmitt trigger when the output is undefined. The first-order approximation assumed for the flip-flops results in monotonic inputs to the Schmitt trigger; hence, issues relating to marginal triggering of the Schmitt trigger[4] are avoided.

The Schmitt synchronizer fails when the output of FF_1, v(t), passes through the Schmitt trigger thresholds, v_d and $-v_d$, at a time that causes the Schmitt trigger output to be in transition at t=T. This is illustrated by the timing diagram in Figure 18 for positive edge transitions.

From Figure 18 we can see that failure occurs on a positive edge output if

$$v(t) = v_d \tag{23}$$

for $T-T_s < t < T-t_s+t_r$. Substituting for v(t) from Equation 6 and rearranging Equation 23 gives

$$v_d e^{-\frac{T-t_s+t_r}{\tau}} < v_0 < v_d e^{-\frac{T-t_s}{\tau}} \tag{24}$$

For the negative edge, the range of values of v_0 that give rise to failure is given by

$$-v_d e^{-\frac{T-t_s}{\tau}} < v_0 < -v_d e^{-\frac{T-t_s+t_r}{\tau}} \tag{25}$$

From Equations 24 and 25, we obtain for the Schmitt synchronizer the value R_2, the sum of the sizes of the two intervals of v_0 that result in failure:

$$R_2 = 2v_d e^{-\frac{T-t_s}{\tau}}\{1-e^{-\frac{t_r}{\tau}}\} \tag{26}$$

If it is assumed that the values of v_0 are uniformly distributed between $-v_d$ and $+v_d$ under marginal triggering conditions, then R_2/R_1 represents the ratio of the probability of failure of the Schmitt synchronizer to the probability of failure of the simple synchronizer. Note that the probability of marginal triggering in both synchronizers is equal since the asynchronous input is the same stochastic process in both cases.

From Equations 22 and 26 it follows that

$$\frac{R_2}{R_1} = e^{\frac{t_s}{\tau}}\{1-e^{-\frac{t_r}{\tau}}\} \tag{27}$$

If it is assumed that the flip-flop FF_1 and the Schmitt trigger are in the same logic family, then $(t_s/\tau) >> 1$. This follows since the propagation delay of a Schmitt trigger, t_s, is comparable with a gate delay, whereas τ is typically an order of magnitude smaller than a gate delay. For example, Chaney shows that for a 74S74 flip-flop, τ is at most 1.7 ns and that for a Schmitt trigger of the same logic family—a 74S11—the propagation delay is typically 8.5 ns.[15] Even if a very high speed comparator with compatible STTL output is used, the propagation delay is of the same order.

Moreover, the rise and fall times of the Schmitt trigger are at least as long as τ (usually much longer), and hence $t_r/\tau > 1$. Thus, we can conclude that $R_2/R_1 >> 1$, which implies that the simple synchronizer performs much better than the Schmitt synchronizer. The significant factor is the loss of settling time resulting from the propagation delay of the Schmitt trigger exponentially increasing the probability of synchronization failure while the possibly short rise time of the Schmitt trigger is at best only linearly decreasing it.

We have shown that the Schmitt synchronizer has an opposite effect on the probability of failure than intended. We have used the first-order model, which takes a rather simplistic view of the metastable behavior of a flip-flop. Experiments have shown that for the simple synchronizer the first-order model gives estimates of the probability of failure of flip-flops consistent with probabilities of failure of flip-flops in real circuits. However, it is not clear how applicable the model is to the analysis of more complex synchronizers such as the Schmitt synchronizer.

Under the first-order model, the effects of noise on the flip-flop output and the possibility of non-monotonic output behavior (for example, oscillations such as those observed by Chaney and Molnar[6]) are not taken into account. Nevertheless, we believe that even though the effects of these departures from the ideal first-order behavior of a flip-flop are not taken into account, they tend to increase the probability of failure for the Schmitt synchronizer in any case since they cause the output of the Schmitt trigger to become less reliable. For this reason we stand by our result—that the simple synchronizer performs better than the Schmitt synchronizer—and we believe that it applies to real circuits.

REDUNDANCY AND MASKING

One can use redundant hardware to mask component failures.[24] Here, we consider whether redundancy and masking can improve the reliability of synchronization. As will become evident from the results presented, synchronization failure due to metastable behavior cannot be treated as a hardware component failure in the usual sense, and redundancy and masking techniques are ineffective.

A synchronizer can use triple hardware redundancy and voting to suppress the anomalous behavior of a synchronizing element (Figure 19). In such a synchronizer, the outputs of three synchronizing elements, IFF_1, IFF_2, and IFF_3, are voted on after one clock period of settling time. If one flip-flop is unresolved due to metastable operation while the other two resolve to the same logic value, the voter circuit masks the metastable state. It does so because it is insensitive to changes in one input when the other two inputs agree. However, if an input data edge occurs that results in two flip-flops resolving to different values, and if the third flip-flop enters a metastable state, then the voter output may be undefined, since changes in the third input affect the voter output in this situation. Thus, the redundant synchronizer does not mask all occurrences of metastable behavior in three synchronizing flip-flops.

This deficiency can be eliminated by phasing the clock on the input flip-flops and increasing the number of flip-flops and voters. We can view a synchronizer thus modified as a general system consisting of n synchronizing elements that have arbitrarily phased synchronous clocks and that feed a combinational function (Figure 20). (Note that the redundant synchronizer discussed above is an example of this general system in which $n=3$, $\tau_1=\tau_2=\tau_3=0$, and the combinational function is a majority

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

voter.) 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.[25]

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.

We have examined a number of such techniques here. They include the use of fast devices, extended decision time, a pausable clock, a Schmitt synchronizer, and redundancy and masking. Our evaluations show that the use of extended decision time is the best technique for lowering the probability of synchronization failure.

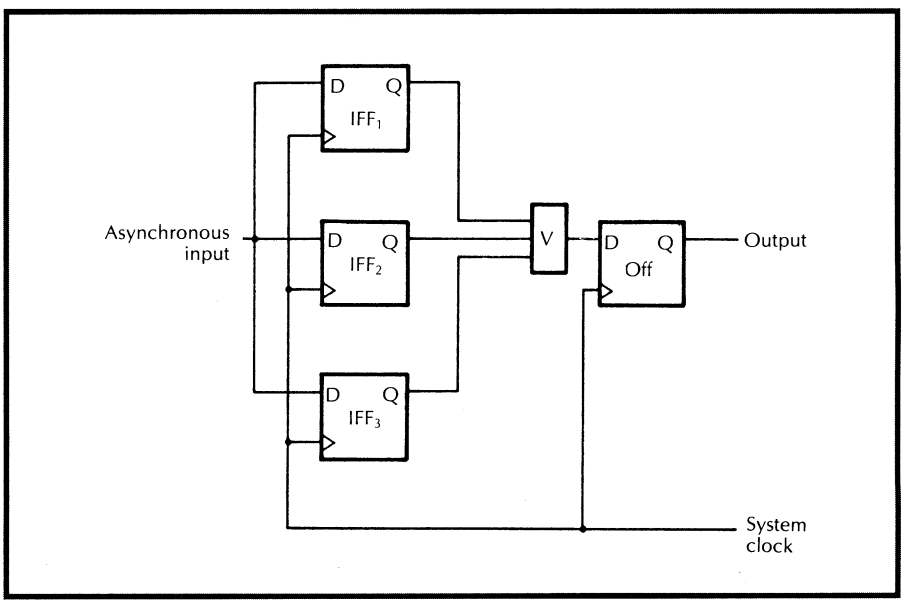


Figure 19. A synchronizer that uses redundant flip-flops and voting among their outputs to reduce the probability of failure.

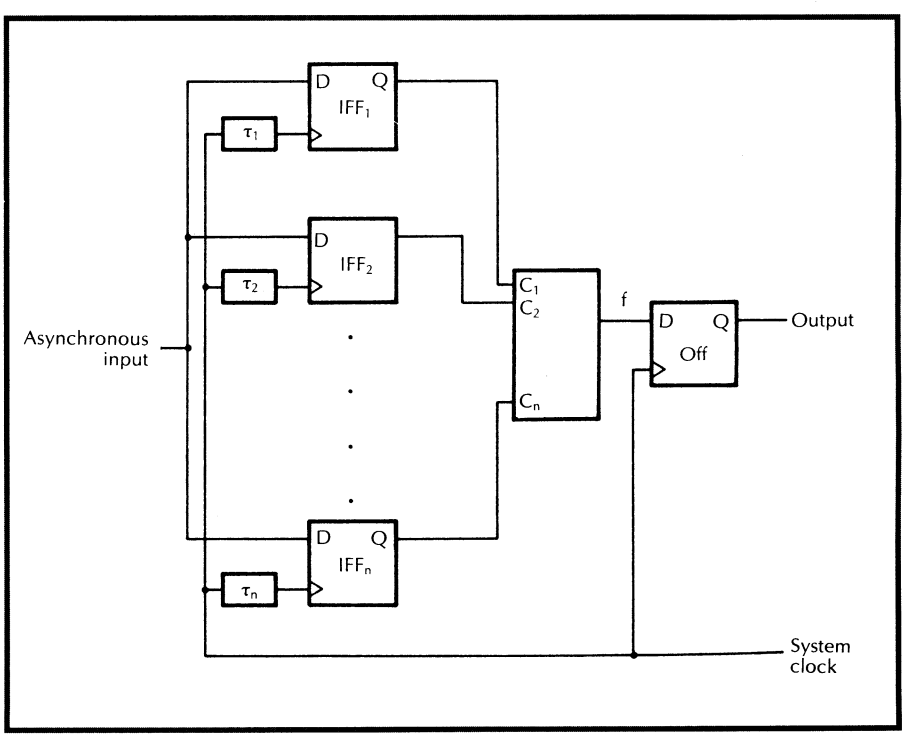


Figure 20. Generalized structure of synchronizers that use clock phasing, redundant flip-flops, and voting.

Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.

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

1. G.R. Couranz and D.F. Wann, "Theoretical and Experimental Behavior of Synchronizers Operating in the Metastable Region," *IEEE Trans. Computers*, June 1975, pp. 604-616.
2. I. Catt, "Time Loss Through Gating of Asynchronous Logic Signal Pulses," *IEEE Trans. Electronic Computers*, Feb. 1966, pp. 108-111.
3. S.H. Unger, "Asynchronous Sequential Switching Circuits With Unrestricted Input Changes," *IEEE Trans. Computers*, Dec. 1971, pp. 1437-1444.
4. L.R. Marino, "The Effect of Asynchronous Inputs on Sequential Network Reliability," *IEEE Trans. Computers*, Nov. 1977, pp. 1082-1090.
5. M. Cortes et al., "Properties of Transient Errors Due to Power Supply Disturbances, " *Proc. Int'l Conf. on Circuits and Systems*, May 1986..
6. T.J. Chaney and C.E. Molnar, "Anomalous Behavior of Synchronizer and Arbiter Circuits," *IEEE Trans. Computers*, Apr. 1973, pp. 421-422.
7. T.J. Chaney, "Comments on 'A Note on Synchronizer and Interlock Maloperation'," *IEEE Trans. Computers*, Oct. 1979, pp. 802-804.
8. W.I. Fletcher, *An Engineering Approach to Digital Design*, Prentice-Hall, Englewood Cliffs, N.J., 1980, p. 484.
9. L.R. Marino, "General Theory of Metastable Operation," *IEEE Trans. Computers*, Feb. 1981, pp. 107-115.
10. L. Kleeman and A. Cantoni, "On the Unavoidability of Metastable Behavior," *IEEE Trans. Computers*, Jan. 1987, pp. 109-111.
11. C. Mead and L. Conway, *Introduction to VLSI Systems*, Addison-Wesley, Reading, Mass., 1980, chap. 7.
12. M. Pechoucek, "Anomalous Response Times of Input Synchronizers," *IEEE Trans. Computers*, Feb. 1976, pp. 133-139.
13. H.J.M. Veendrick, "The Behavior of Flip-Flops Used as Synchronizers and Prediction of Their Failure Rate," *IEEE J. Solid-State Circuits*, Apr. 1980, pp. 168-176.
14. F.U. Rosenberger and T.J. Chaney, "Flip-Flop Resolving Time Test Circuit," *IEEE J. Solid-State Circuits*, Aug. 1982, pp. 731-738.
15. T.J. Chaney, "Measured Flip-Flop Responses to Marginal Triggering," *IEEE Trans. Computers*, Dec. 1983, pp. 1207-1209.
16. *Microsystems Components Handbook*, Vol. 1, Appendix B, Intel Corp., Santa Clara, Calif., 1984, pp. 3-315.
17. D.A. Laws and P. Alfke, *Bipolar Microprocessor Logic and Interface Data Book*, Advanced Micro Devices, Sunnyvale, Calif., 1983, pp. 8-4 to 8-7.
18. G. Elineau and W. Wiesbeck, "A New JK Flip-flop for Synchronizers," *IEEE Trans. Computers*, Dec. 1977, pp. 1277-1278.
19. W. Lim and J.R. Cox, Jr., "Clocks and the Performance of Synchronisers," *Proc. IEE*, Mar. 1983, pp. 57-64.
20. M.J. Stuck and J.R. Cox, Jr., "Synchronization Strategies," *Proc. Caltech Conf. on VLSI*, Jan. 1979, pp. 375-393.
21. R.L. Adams, D.R. Castaldo, and G.W. Kurtz, "Real-Time Detection of Latch Resolution Using Threshold Means," US Patent 3,515,998, June 1970.
22. G.G. Freeman et al., "Two CMOS Metastable Sensors," CRC Tech. Report No. 86-4, Dept. of Electrical Engineering and Computer Science, Stanford Univ., May 1986.
23. P.A. Stoll, "How to Avoid Synchronizer Problems," *VLSI Design*, Nov. 1982, pp. 56-59.
24. J.F. Wakerly, "Transient Failure in Triple Modular Redundancy Systems With Sequential Modules," *IEEE Trans. Computers*, May 1975, pp. 570-573.
25. L. Kleeman and A. Cantoni, "Can Redundancy and Masking Improve the Performance of Synchronizers?," *IEEE Trans. Computers*, July 1986, pp. 643-646.

For further reading

Chaney, T.J., S.M. Ornstein, and W.M. Littlefield, "Beware of the Synchronizer," *Digest of Papers—Compcon 72*, Sept. 1972, pp. 317-319.

Cortes, M., and E.J. McCluskey, "Modelling Power-Supply Disturbances in Digital Circuits," *Proc. Int'l Solid-State Circuits Conf.*, 1986.

Fleischhammer, W., and D. Dortok, "The Anomalous Behavior of Flip-flops in Synchronizer Circuits," *IEEE Trans. Computers*, Mar. 1979, pp. 273-276.

Kameyama, M., and T. Higuichi, "Design of Dependent-Failure-Tolerant Microcomputer System Using Triple Modular Redundancy," *IEEE Trans. Computers*, Feb. 1980, pp. 202-206.

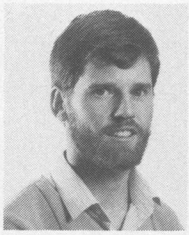
Kinniment, D.J., and J.V. Woods, "Synchronisation and Arbitration Circuits in Digital Systems," *Proc. IEE*, Oct. 1976, pp. 961-966.

Lacroix, G., P. Marchegay, and G. Piel, "Comments on 'The Anomalous Behavior of Flip-flops in Synchronizer Circuits'," *IEEE Trans. Computers*, Jan. 1982, pp. 77-78.

Landan, T., "Asynchronous Timing in Logic Systems," *Digital Processes*, 1976, pp. 157-162.

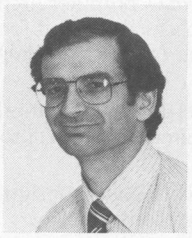
TIBPAL16XX Data Sheet (No. D3023), Texas Instruments, Dallas, Texas, May 1987, pp. 13-14. (Provides design parameters for calculating the probability of failure due to metastability.)

Wakerly, J.F., "Microcomputer Reliability Improvement Using Triple Modular Redundancy," *Proc. IEEE*, Jan. 1976, pp. 889-895.



Lindsay Kleeman has been a lecturer in the Department of Electrical Engineering at Monash University, Clayton, Victoria, Australia, since 1986. His interests include metastable failure, fault-tolerant hardware and software design, hardware correctness proving, VLSI design, the design of asynchronous hardware for reliability and testability, self-timed systems, and performance evaluation.

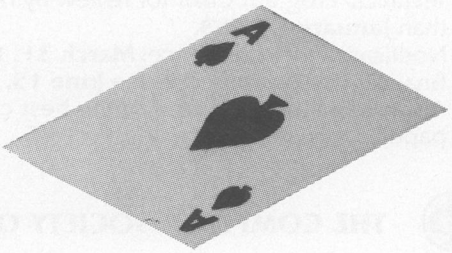
Kleeman received a degree in electrical engineering in 1982 and a degree in mathematics in 1983, both with first-class honors and university medals, from the University of Newcastle, Shortland, New South Wales, Australia. He received the PhD, also from the University of Newcastle, in 1987. He is a member of the IEEE.



Antonio Cantoni is projects director for QPSX Communications Pty. Ltd., Perth, Western Australia, and is also a visiting professor of electrical engineering at the University of Western Australia, Nedlands. Previously, he was a professor of computer engineering at the University of Newcastle. He is interested in adaptive systems, digital communication, and industrial electronics. He regularly acts as a consultant to industry, advising companies developing electronic and microprocessor-based systems.

Cantoni received the BE, with first-class honors, and the PhD from the University of Western Australia in 1968 and 1972, respectively. He is a senior member of the IEEE.

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



Authorized licensed use limited to: TU Wien Bibliothek. Downloaded on July 17,2023 at 15:36:31 UTC from IEEE Xplore. Restrictions apply.