Chapter 6

Timing Diagrams

6.1 INTRODUCTION

The behavior of logic systems is usually described and investigated through the use of timing diagrams. These diagrams display, along a time axis, voltages or logic levels at various points in a digital circuit and can be used to indicate both the functional relationships and time delays which exist between inputs and outputs. The speed with which an input change causes a corresponding change in the output is often a critically important design parameter, and the timing diagram is a valuable aid in the investigation of its impact on system performance. A laboratory instrument widely used in troubleshooting logic hardware is the logic analyzer, which is essentially a multichannel storage oscilloscope that displays several timing waveforms simultaneously. Another valuable tool, particularly in the design phase, is the logic simulator which consists of software that runs on a computer workstation and is capable of producing families of related timing diagrams from schematic or Boolean inputs. Logic simulators are valuable to designers since they permit the validation of system performance prior to commitment to hardware.

There are two basic types of timing diagrams. First, there is the microtiming diagram which is concerned with the propagation delays encountered in individual gates. It is used to detect and display undesirable conditions such as “glitches” and unstable oscillations due to feedback. Second, there is the macrotiming diagram which is concerned with the time relationships between signals at various points in the system on a time scale large enough to make the consideration of individual gate delays unnecessary. Various gate input and output waveforms are displayed for a fixed interval, usually with reference to a timing or clock waveform.

6.2 MICROTIMING DIAGRAMS

In digital systems, the hardware implementation of logic gates involves electronic devices which switch between conductive and nonconductive states. It takes a small but finite time for such switching to occur because of electron charge storage and conduction effects which cause a measurable delay between the application of a voltage level change at a gate input and an appropriately recognizable response at the gate's output. This time interval, called the propagation-delay time (\( t_{pd} \)), is specified by hardware manufacturers on their data sheets and tends to be a constant which is applicable to all gates of the same type.

For the purposes of microtiming logic analysis, every gate in the system is often assumed to have an identical unit propagation delay. The microtiming diagram takes these propagation delays into account to show the behavior of a system following a specified change in one or more of its inputs, assumed to occur at \( t = 0 \).

Microtiming Diagram Preparation

1. The initial values of signals (either logic or voltage) applied to all system inputs must be known and recorded for \( t = 0^- \), the time immediately prior to an input change at \( t = 0 \).
2. Using these values, the state of each internal gate is determined and entered on the timing diagram.
3. A specific input signal is allowed to change its logic state, and this change is entered on the timing diagram at \( t = 0 \).
4. All connections linking the changed signal to the input(s) of internal gates are identified by examination of the logic diagram.

5. Each of these gates is examined to see if its logic state will be altered by the change(s) in step 4. If it is, the response is entered on the diagram in the next succeeding propagation-delay interval. Note that since all gates are presumed to have identical delays, the timing diagram is constructed in discrete steps equal to the unit propagation-delay time \( t_{pd} \).

6. Each changed gate output, if any, is now treated as an input change for any gate(s) to which it is connected, and step 5 is repeated.

7. The process terminates when no more gate outputs are found to change.

**EXAMPLE 6.1** Assume that the physical devices (gates and inverters) in the logic circuit of Fig. 6-1 have identical propagation delays. All the inputs and outputs in this example are considered to be HT and, initially, A, B, C, and D are all 1s. At \( t = 0 \), C changes to a logic 0. Draw the appropriate logic microtiming diagram, and then convert it to display voltage waveforms.

\[ \text{Fig. 6-1} \]

Separation of the various waveforms (particularly if they are hand drawn) is made somewhat less confusing if shading is placed beneath the logic 1 states as shown in Fig. 6-2.

**Logic Waveforms**

Variable C, which changes from a logic 1 to a logic 0 at \( t = 0 \), is seen to connect to gates G_1, G_4, and G_5 as indicated by the notation "XG_1, G_4, G_5" in Fig. 6-2. We examine each of these in turn. Gate G_1 is a voltage inverter. Since it appears in conjunction with a slash mark, its output must change to a logic 1 when C goes to a 0. Note that if an inverter is used solely to match half arrows (as indicated by the absence of an associated slash mark), then no logical inversion occurs and the hardware inverter simply passes on a logic change at its input after a delay of one time interval. Gate G_4 is an AND gate, so its output will go to logic 0 if any of its inputs become 0. Thus G_4 will change. Gate G_5 will not be affected by the change to C since one of its inputs is already 0. The changes in the outputs G_1 and G_4 are entered in the diagram at time mark 1, one time-delay interval following the stimulus which caused them (the change in C).
We now look at the gates which have just changed. Gate G₁ is connected only to G₃; G₄ is seen to affect only G₆. Gate G₃ will go to a 1 since all three inputs will be 1s immediately following time interval 1. Gate G₆ will go to a 0 since all three inputs will be 0s immediately following time interval 1. The changes in G₃ and G₆ are entered at time mark 2.

Repeating the preceding process, we note that only the connection between G₃ and G₆ remains to be evaluated: Gate G₆ (F) will be expected to return to a logic 1 since all three inputs will again be 1s. The process terminates since the last gate to change is connected to no others.

Voltage Waveforms

The outputs of G₁, G₂, and G₆ are HT, and, consequently, their logic and voltage waveshapes will be identical. Gates G₃, G₄, and G₅ have LT outputs making it necessary to invert their logic waveforms to obtain voltage equivalents as shown in Fig. 6-3.

6.3 HAZARDS

The logic represented by the diagram in Fig. 6-1 is easily determined to be

\[ F = A\overline{C}D + BC + A'C. \]

This Boolean expression predicts that when, initially, \( A = B = C = D = 1 \), \( F \) will be 1 and will remain unaltered even though \( C \) changes to 0. The waveforms, however (Fig. 6-2), indicate a temporary condition where output \( F \) momentarily goes LOW and then returns to a proper HIGH state as required by the logic. This short-term aberration, due to unequal propagation delays in parallel logic paths, is called a hazard (sometimes informally referred to as a “glitch”).

Since a variable \( C \) is not available as an input, its requirement by the logic mandates the addition of a hardware voltage inverter which adds an extra unit of delay in its signal path. This, unfortunately,
produces the hazard shown in Fig. 6-2 which arises from the difference in delay between the paths for C and for C' which converge at G₆.

This type of hazard can be recognized on a K map. Consider the map for the function of Example 6.1 (see Fig. 6-4).

In this map, the transition from initial state 1 to the final state 1 is seen to occur across the boundary of two coverings which do not overlap. Generally, in order to determine whether the change in any single variable will result in a hazard condition, one need only look for adjacent initial and final states on the map which are isolated in separate coverings.

The situation may be fixed with relative ease. A hazard cover is added which overlaps source and destination coverings as is shown in Fig. 6-5. This results in an added term in the equation for F and a corresponding increase in the hardware required.

It is important to note that while the map can be used to detect hazards in combinational logic arising from the difference in path lengths between a variable and its complement, there are other classes of timing problems which are not so easily detected (refer to Probs. 6.6 and 6.7). Furthermore, whereas the map may indicate two input states which will be involved in a hazard, it cannot identify
which is the initial one. It is quite possible that a glitch will occur when going from state $S_1$ to state $S_2$, but not in the other direction as shown in Prob. 6.1.

Glitches are often unavoidable and, unless countered by proper design, they can cause serious errors in digital circuits. A powerful means of coping with them involves the implementation of strobing or clocking techniques as discussed in Sec. 7.4.

6.4 MACROTIMING DIAGRAMS

Macrotiming diagrams are used to display the outputs of various gates relative to independent inputs or some reference such as a system clock pulse train. They are primarily concerned with circuit behavior viewed on a time scale considerably larger than that used to study propagation-delay phenomena. The effects of individual gate delays are often omitted intentionally or become insignificantly small compared to logic-related sequential transitions shown in the diagram.

EXAMPLE 6.2 The HT inputs A, B, C, and D to the logic shown in Fig. 6-6 vary with time as shown at the top of Fig. 6-7. Assuming that the circuit responds instantaneously to input changes (that is, device propagation delays cannot be distinguished on the time scale used) sketch the output voltage waveshapes F and G.

---

**Fig. 6-6** Logic diagram.
From the logic diagram, $F = AB + (C + D)'$. Using De Morgan's theorem on the second term yields $F = AB + C'D'$; that is, $F$ will be TRUE if either $A$ and $B$ are both TRUE or $C$ and $D$ are both FALSE. These conditions are met three times for the given inputs and, since $F$ is HT, its corresponding voltage waveform will conform to the logic, as shown in Fig. 6-7.

Output $G$ is seen to equal $B'(C + D)'$ or, equivalently, $G = B'C'D'$. Thus, $G$ will be TRUE only if $B$, $C$, and $D$ are all FALSE and, since $G$ is LT, its voltage waveform will be the inverse of that implied by the logic.

### 6.5 Timing Simulations

A major tool for confirming logic design functionality is a computer logic simulator which permits the designer to emulate the breadboarding and testing phases of a design without the need for hardware. Though not a substitute for actual prototyping, the simulator speeds the design process by permitting the early detection of basic flaws and timing problems. Sitting at a workstation monitor, an engineer "assembles" the hardware circuit by a process called schematic capture in which the circuit diagram is drawn on the screen by selecting logic components from a software library and interconnecting them by using a mouse or similar pointing tool which permits convenient positioning of objects on the computer screen. A typical screen display from a simulator package called LogicWorks™ from Capilano Computing is shown in Fig. 6-8 which depicts the circuit of Example 6.1.

On the screen shown in Fig. 6-8, the pull-down menu which provides access to the gate library is shown. In the present case, the symbol for NAND has been chosen to emphasize that the analysis will be a hardware simulation. Though most current simulators make use of positive logic conventions, an increasing number are beginning to incorporate mixed-logic symbols, and future versions of LogicWorks™ are expected to do the same.

As a designer creates the circuit schematic, interconnection paths are stored in the computer as a net list. The simulation software applies selected signals and/or fixed levels at the network inputs and monitors responses at outputs and any other internal points desired. In Fig. 6-8, inputs $A$, $B$, and $D$ are connected to logic 1 via operable switches selected from the I/O menu. Input $C$ is caused to step from 1 to 0 by drawing its waveform (using the mouse) in a timing window, a portion of which is shown in Fig. 6-9. Following a reset command, the simulation is started and the waveforms shown are generated in less
than a second. Compare the results with the manual timing diagram of Fig. 6-3, and note confirmation of the predicted glitch in output F.

In most simulation software, it is possible to vary the delays of selected components and to go glitch hunting. In the current example, all gate delays were set to 10 time units prior to running the simulation. Some software includes glitch-detection routines which predict hazard conditions.
6.6 FEEDBACK IN COMBINATIONAL CIRCUITS

Figure 6-10 shows a combinational logic circuit which contains feedback. In this case, two of the output variables (Y₁ and Y₂) are functions of several variables, including themselves. In other words, in a circuit with feedback, the value of a given variable is affected by the variable itself.

![Combinational Logic Circuit Diagram](image)

Fig. 6-10

Under certain conditions, such circuits can be unstable since the arrival of a feedback signal can cause the output to change which, in turn, will modify the input and cause the output to change again, and so on. If a K-map hazard analysis predicts a glitch, we might expect the probability of continuous oscillation between two states, the rate being dependent upon circuit propagation delays.

We see from Fig. 6-10 that

\[
Y₂ = Y₁Y₂R'S' + S
\]
\[
Y₁ = Y₁Y₂R' + RS + Y₁S
\]

K maps for Y₁ and Y₂ are shown in Fig. 6-11 where we observe, in the Y₂ map, that there are nonoverlapping coverings involving adjacent 1s. This indicates that a glitch is to be expected in Y₂ when R = 0 and S changes, warning us that we would be well advised to analyze this circuit further to check for oscillatory behavior.

It is obvious that, as circuit designs become more complex and incorporate feedback, the construction of microtiming diagrams by hand becomes a burdensome task. The computer simulator makes the job vastly easier, permitting designers to check out circuits much more completely before investing in
(a) Simulation schematic

(b) Predicted waveforms (microtiming diagram)

Fig. 6-12
hardware. As an example, consider the case where the feedback circuit of Fig. 6-10 is constructed with NAND hardware. The predicted instability is easily demonstrated by a LogicWorks™ simulation as shown in Fig. 6-12.*

Note that, as predicted by the K maps, instability occurs in Y₂ when R = 0 and S changes.

Solved Problems

6.1 Consider the NAND implementation of the function \( F = (AB)' + AD \) shown in Fig. 6-13. Assuming that all gates have the same time delays, draw a logic microtiming diagram for the case where A changes while B = D = 1.

Prior to beginning a step-by-step signal trace, it is advisable to determine the underlying logic. This may be easily done by applying the conversion process discussed in Prob. 4.10 which provides us with the required mixed-logic equivalent reproduced in Fig. 6-14.

Microtiming diagram generation is described in Example 6.1. The connections radiating from each variable change are traced, the affected gate or gates are identified, and the resulting logic level changes (if any) are entered in the diagram one time-delay interval following the stimulus. The resulting timing

* The simulation of circuits with feedback can sometimes fail to function properly because of the lack of unambiguous initial conditions. A brief discussion of how this problem is handled when using LogicWorks™ can be found in App. C.
diagram is shown in Fig. 6-15. The given Boolean expression leads us to believe that F will remain equal to 1 regardless of the state of A, and, as expected, there is no response to a 0-to-1 change in input A. Contrary to Boolean prediction, however, a 1-to-0 change in A produces a glitch two time-delay intervals following the input transition. This result may be compared with simulation waveforms if all LT outputs are inverted (see Fig. 6-16).

![Timing Diagram](image)

**Fig. 6-15** Microtiming logic levels.

![Timing Simulation](image)

**Fig. 6-16** Timing simulation. Note the glitch in F following a transition in A from 1 to 0.

6.2 For the circuit in Prob. 6.1, draw the K map, add a hazard covering to eliminate the glitch, and discuss its impact on the hardware.

Application of De Morgan's theorem to the given function yields $F = A' + B' + AD$. The K map with encirclements corresponding to this Boolean form is shown in Fig. 6-17 where the glitch may be readily identified as adjacent 1s in isolated coverings as indicated. A hazard covering can be added over the two indicated 1s, or it may be extended over the entire right-hand column. The former arrangement requires an additional AND gate. The latter yields a simple Boolean term, but it requires an inverter to balance the half arrows, so, in the present case, there is probably not much reason to choose one method over the other. If some other specification or design rule doesn't tip the balance, a professionally engineered coin toss is permissible.
6.3 Assuming that all hardware (gates and inverters) in Fig. 6-18 have the same time delay, sketch the complete logic microtiming diagram. Initially, all input variables are 1s. At \( t = 0 \), C changes to a 0. All inputs and outputs are HT.

Since A, B, and D remain 1s, it is relatively easy to trace the logic. When C changes, it directly affects gates \( G_1 \), \( G_4 \), and \( G_5 \) which, if they respond, will do so after one time-delay interval. Gate \( G_3 \) does not change state since its lower input remains at logic 0. When gate \( G_4 \) changes, it can't affect \( G_6 \) because \( G_7 \) is at logic 1. However, inverter \( G_7 \) goes to logic 0 in the third time interval in response to the change in gate \( G_3 \), causing \( G_6 \) (and F) to go to 0 in interval 4. The logic microtiming diagram is shown in Fig. 6-19.

It is interesting to compare this circuit to that of Fig. 6-1 which differs only in the absence of inverter \( G_7 \). Though, in Fig. 6-18, the reconvergent paths for variable C have three time delays in the upper branch and only one in the lower, no glitch occurs because of the logic structure.

6.4 The circuit shown in Fig. 6-20 is a common interconnection of gates called a latch which will be described in detail in Chap. 7. Note the cross-coupled feedback arrangement where the output of one gate is used as an input for the other. Assume that the output states are as shown and both inputs are initially at logic 0. At time \( t = 0 \), input R changes to a 1. Given that both gates have equal delay times, sketch the microtiming logic diagram.

Because of the logical inversions, the feedback inputs to gates \( G_1 \) and \( G_2 \) are 1 and 0, respectively. With both external inputs at logic 0, the given outputs are seen to be consistent. When R changes, it affects \( G_2 \) which goes to a logic 1 after one time-delay interval. This causes \( G_1 \)'s feedback input to become 0, and,
Fig. 6-19  Logic waveforms.

Fig. 6-20