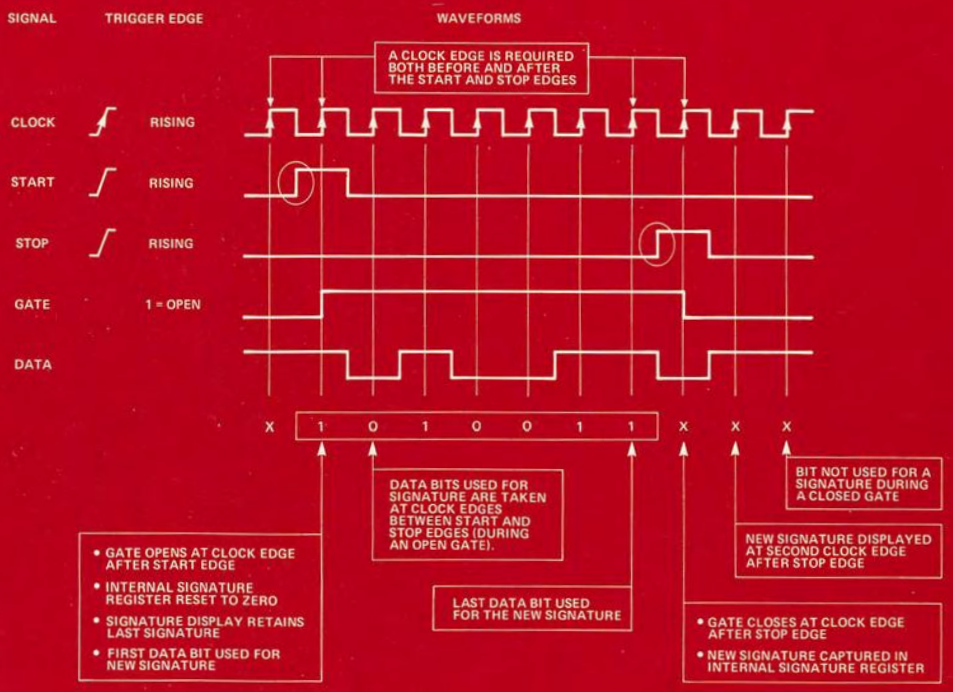
Application Note 222-4

GUIDELINES FOR SIGNATURE ANALYSIS

Understanding the Signature Measurement



HEWLETT PACKARD

APPLICATION NOTE 222-4

GUIDELINES FOR SIGNATURE ANALYSIS
Understanding the Signature Measurement

This Application Note shows how Hewlett-Packard Signature Analyzers take signature measurements. It contains guidelines for controlling the gate through the Start, Stop, and Clock inputs. It shows how measurements of three-state nodes are treated, and more. This information applies to both design and retrofit situations.

FORWARD

ABOUT DIGITAL TROUBLESHOOTING

Microprocessors have revolutionized your product line. Your products are smarter, faster, friendlier and more competitive because they take advantage of μP-based control and computation. They are also harder to build, harder to test and harder to fix when they fail. Complex bus structures and timing relationships have practically obsoleted the scope/voltmeter signal tracing techniques so effective on analog products. The need to enhance the testability and serviceability of your digital products is acute. So is the need for specialized digital troubleshooting equipment.

ABOUT SIGNATURE ANALYSIS

To address these needs, Hewlett-Packard has developed the Signature Analysis technique, as well as a Signature Analyzer product line, for component-level troubleshooting of microprocessor-based products. A Signature Analyzer detects and displays the unique digital signatures associated with the data nodes in a circuit under test. By comparing these actual signatures to the correct ones, a troubleshooter can back-trace to a faulty node. By designing or retrofitting S.A. into digital products, a manufacturer can provide manufacturing test and field service procedures for component-level repair, without dependence on expensive board-exchange programs.

ABOUT THIS PUBLICATION

This application note is aimed at assisting designers, test engineers and others in designing or retrofitting their digital products for Signature Analysis testability and serviceability. It shows how Hewlett-Packard Signature Analyzers take signature measurements. While there are many different ways to control the signature measurement, there are a few common ways that apply to most microprocessor-based products. We've compiled a list of these common measurement control methods in the form of guidelines. These guidelines resulted from suggestions of some of the hundreds who have used signature analysis for the past few years. They are designed to show how to create measurements that result in stable and repeatable signatures, so that correct signatures can be documented in a troubleshooting procedure. When that goal is met, then incorrect, unrepeatable, or unstable signatures measured during troubleshooting, will accurately indicate incorrect or intermittant circuit behavior, allowing fast fault analysis to the component level.

ABOUT OTHER PUBLICATIONS

Application Note 222-0, HP Publication 02-5952-7593, is a complete index to the latest Signature Analysis publications. It lists all other application notes currently available in the AN 222 series about Signature Analysis. They cover a wide range of interests, from how to design or retrofit Signature Analysis into digital systems, to the cost reductions that can be expected in production test and field service by doing so. It also lists all data sheets for the complete line of Hewlett-Packard Signature Analysis products, plus other related publications about digital troubleshooting.

CONTENTS

SECTION **PAGE**

SECTION 1

Introduction

Contents

This application note explains how Hewlett-Packard Signature Analyzers take signature measurements. Here is what is covered in this note:

1. How the GATE, or measurement cycle, is controlled, including simplified examples that demonstrate basic gating concepts. Some application examples are also given to demonstrate the concepts in a real situation.
2. How data on three-state nodes is interpreted by the signature analyzer during a measurement cycle.
3. How the signature analyzer responds to a reset, and how the HOLD feature works in conjunction with reset.
4. How the signature analyzer is affected by noise on any of its inputs.
5. The signature analyzer specifications.
6. Ways the measurement cycle can be controlled by the product under test, through a combination of hardware and software, during the execution of a circuit stimulus program.

Examples of typical circuit stimulus programs are the subject of additional Case Study notes in this Application Note series. They show typical examples of stimulus program software and the circuits that get stimulated. The Case Studies also contain additional examples of how the GATE is controlled. See Application Note 222-O, "An Index to Signature Analysis Publications," HP Publication 02-5952-7593, for a complete list of current application notes in the AN 222 series.

Organization

Each section of this application note covers a measurement concept by showing simple examples of START, STOP, CLOCK, and DATA waveforms. In some cases, actual application examples will also be shown. The title for each section reflects the way in which most people phrase questions about how the signature analyzer takes a measurement. This organization allows the note to be used as a quick reference guide when a specific question comes up about a measurement concept. But the organization also allows someone unfamiliar with the signature measurement process, to read through it, without having to refer back and forth to different sections.

The simplified examples are intended only to demonstrate and clarify a measurement concept. They do not necessarily reflect an actual measurement. For example, the START, STOP, CLOCK and DATA waveforms are simplified so that they could easily be drawn in diagrams. GATE times were shortened, and the CLOCK is shown with a fixed frequency. However, GATES can be much longer than shown (GATE length is a subject covered in a section of the same name), and the CLOCK need not be constant. The CLOCK can be symmetrical as shown in most of the examples, but it need not be.

SECTION 2

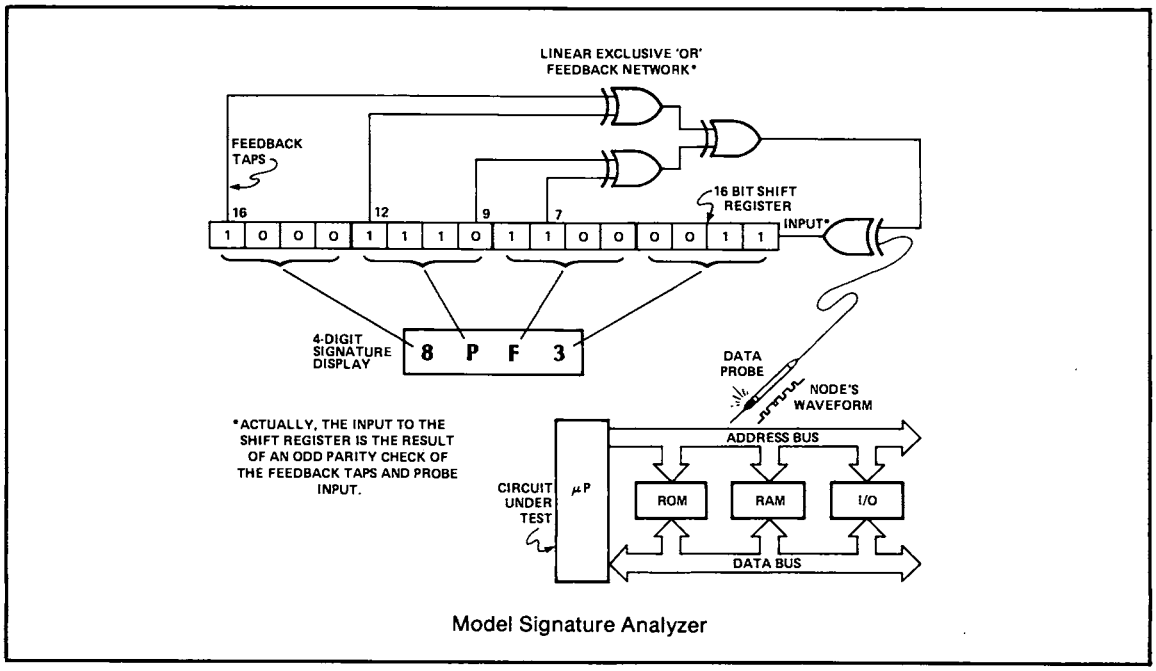
Signature Analyzer Characteristics

Here is a list of the four major characteristics of Hewlett-Packard Signature Analyzers that determine the signature value resulting from a signature measurement cycle. Other signature analyzers may or may not get the same signatures, depending on how they treat these and other characteristics. The first two characteristics are shown in the Figures 2.1 and 2.2. The third and fourth characteristics are covered in the remaining sections of this application note.

1. The internal 16-bit signature register feedback taps.
2. The signature display characters.
3. The gating characteristics.
4. The input logic thresholds.

The Signature Register

Figure 2.1 shows the model for the 16-bit internal signature register including the four feedback tap positions. It also shows how the signature analyzer compresses a node's waveform into a four digit signature display. More on the operation of the shift register and its accuracy of error detection can be found in Application Note 222-2.



Model Signature Analyzer

Figure 2.1

The Signature Display Character Set

Figure 2.2 shows the sixteen characters that form each of the four digits of the signature display. The character set has been changed from standard hexidecimal, becasue the signature analyzer uses a seven-segment display. The seven-segment display was chosen to make the characters as large as economically possible, for easy readability during troubleshooting. The new character set eliminates the confusion between the small letter b and the number 6, or the capital letter B and the number 8. During troubleshooting, measured signatures are compared against documented ones. There is no diagnostic information in the signature itself. So it doesn't matter WHAT the characters are, as long as they are EASY to compare. The new character set makes the signature instantly recognizable for fast troubleshooting.

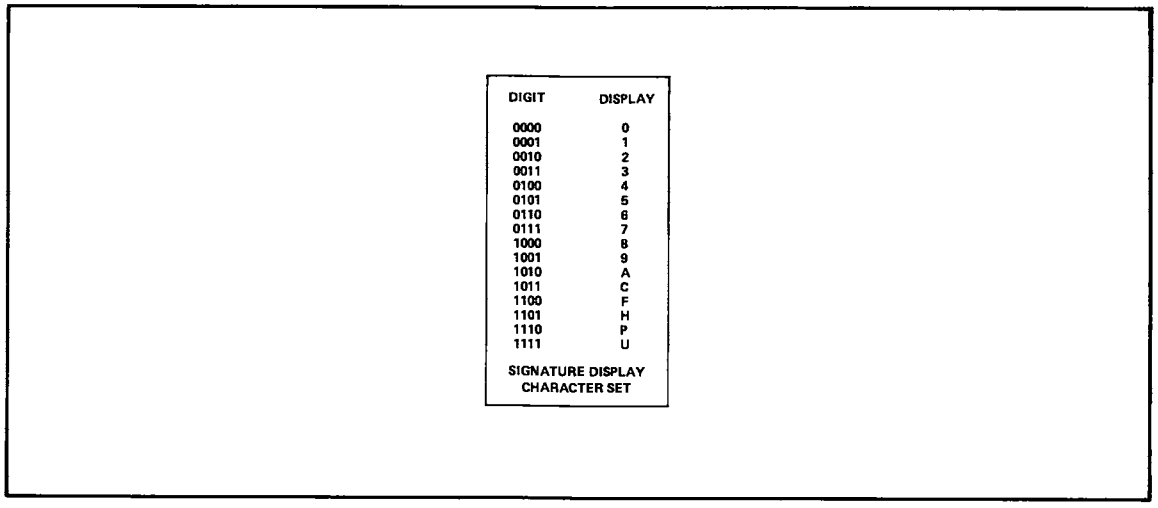


Figure 2.2

Gating Characteristics and Input Thresholds

The gating characteristics are covered in sections three through ten of this application note. Input threshold is the subject of section 11 and Appendix A.

SECTION 3

Basic Gate Operation

Definition of the Gate

The measurement cycle of the signature analyzer is controlled through the GATE. When the GATE is open, the analyzer is taking a new signature measurement through the DATA probe input. The signature from the previous measurement cycle remains displayed. When the GATE closes, the analyzer stops taking the new signature and then displays it.

The GATE is not an input to the signature analyzer. It is an internal state whose operation is shown by the GATE LIGHT. The GATE LIGHT is shown in Figure 3.1. It turns on when the GATE opens, and turns off when it closes. The GATE LIGHT will blink at about 10 Hz if the GATE cycles faster than that.

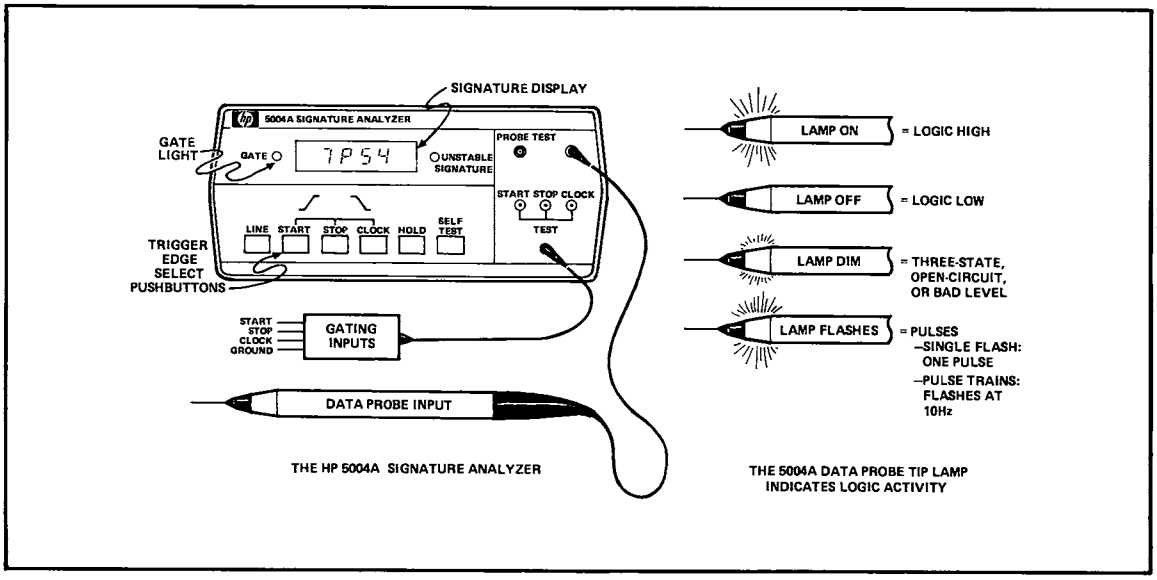
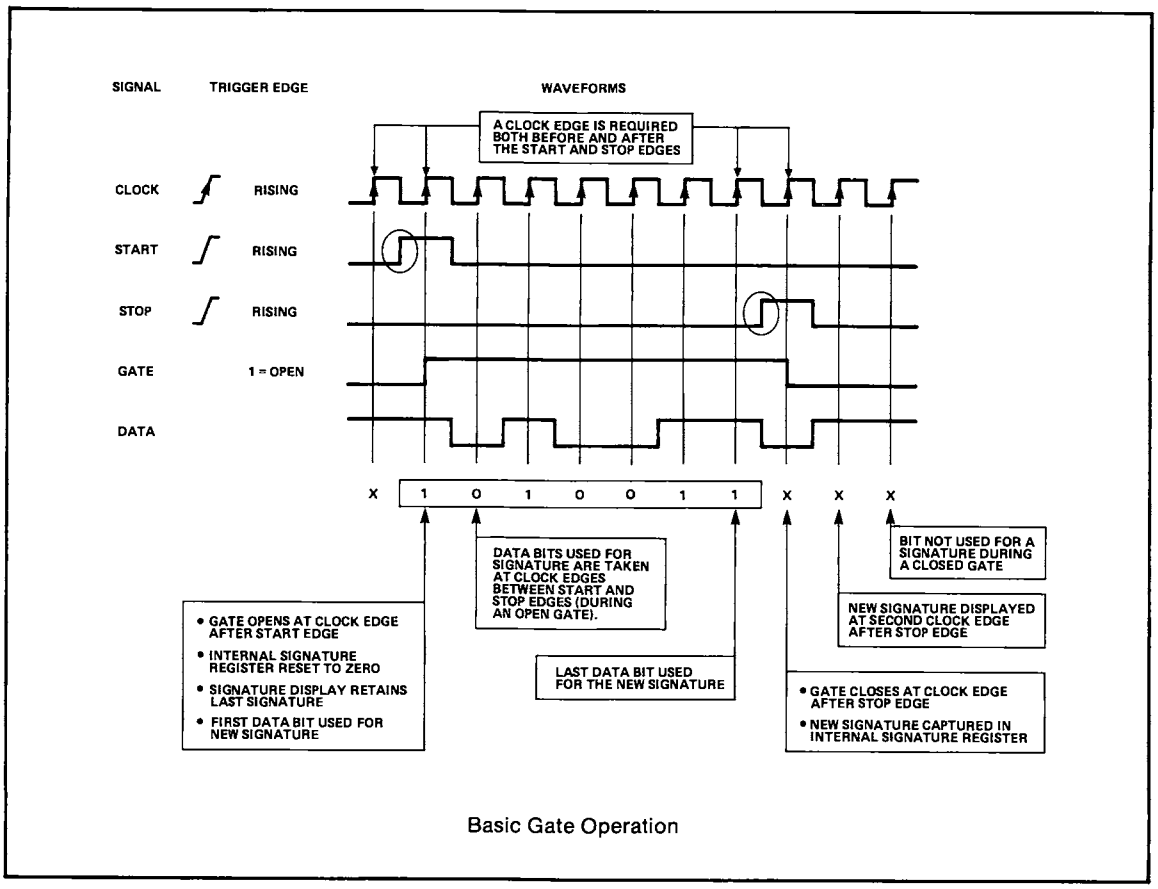


Figure 3.1

The START, STOP, CLOCK and DATA Inputs

The GATE is controlled by the three gating inputs: START, STOP and CLOCK. Basic GATE control is shown in Figure 3.2. START and STOP are used to open and close the GATE at selected trigger edges. The trigger edges for START, STOP and CLOCK are selected by pushbuttons on the instrument, and can be either the rising or falling edge of the input signal, in any combination. From now on, any reference to a START, STOP or CLOCK edge will mean the edge as selected by these pushbuttons.



Basic Gate Operation

Figure 3.2

The CLOCK synchronizes the analyzer to the device being tested. The logic level of the waveform measured by the DATA probe input is sampled at the CLOCK edge and is ignored at all other times. These logic levels are used as data bits that are compressed into the signature. The START and STOP inputs are also synchronously detected by the CLOCK. The GATE opens at the CLOCK edge following the START edge, and not at the START edge itself. Similarly, the GATE closes at the CLOCK edge following the STOP edge, instead of directly at the STOP edge.

First and Last Data Bits Sampled For a Signature

The first DATA bit sampled for use in a signature occurs coincident with the GATE opening at the CLOCK edge following the START edge. The last DATA bit used occurs at the CLOCK edge PRIOR to the STOP edge. In other words, DATA is sampled at every CLOCK edge BETWEEN the START and STOP edges. DATA is not used for a signature when the GATE closes at the CLOCK edge following the STOP edge. The signature is displayed at the SECOND CLOCK edge after the STOP edge.

There are many different ways to control the GATE of the signature analyzer. For instance, both START and STOP can be connected to the same signal or different ones. And they can be triggered on the same or different edges. START usually opens the GATE and STOP closes it, but the GATE can toggle open and closed as well. The GATE can be as long or as short as required. START and STOP can even have multiple edges. The remaining guidelines explain each one of these characteristics and more.

SECTION 4

Getting the Gate To Open and Close

The Interaction of START, STOP and CLOCK

There must be a CLOCK edge both before and after the START and STOP edges, as shown in Figure 3.2. The GATE opens at the CLOCK edge following the START edge, and not at the START edge itself. Similarly, the GATE closes at the CLOCK edge following the STOP edge, instead of directly at the STOP edge. A CLOCK edge is also required before both the START and STOP edges, because the signature analyzer compares the logic states of START and STOP from one CLOCK edge to the next, and detects a change in state as an edge. The result of that comparison determines whether the GATE will open or close. For example, consider START selected for a rising edge trigger, the GATE initially closed, and START initially at logic low. At each CLOCK edge, START remains low, so a trigger edge is not detected, and the GATE remains closed. Now, before the next CLOCK edge, suppose START goes high. At the next CLOCK edge, START will be detected as high. The signature analyzer detects the change in state from low (at one CLOCK edge) to high (at the next CLOCK edge) as a rising edge, and opens the GATE at the CLOCK edge. See Figure 3.2.

Situations That Cause the Gate Not To Open or Close

If the GATE does not open or close as expected, it's probably because there is not a CLOCK edge both before and after the START and STOP edges. Here are four situations that demonstrate this, along with ways to get the GATE to open and close as required.

1: No CLOCK before or after the START or STOP edge.

In Figure 4.1, START and STOP are connected to the same signal and trigger on opposite edges. There is no CLOCK edge to detect the logic LOW level of START/STOP, so the rising edge for START does not open the GATE. Similarly the falling edge for STOP won't close the GATE if it starts open.

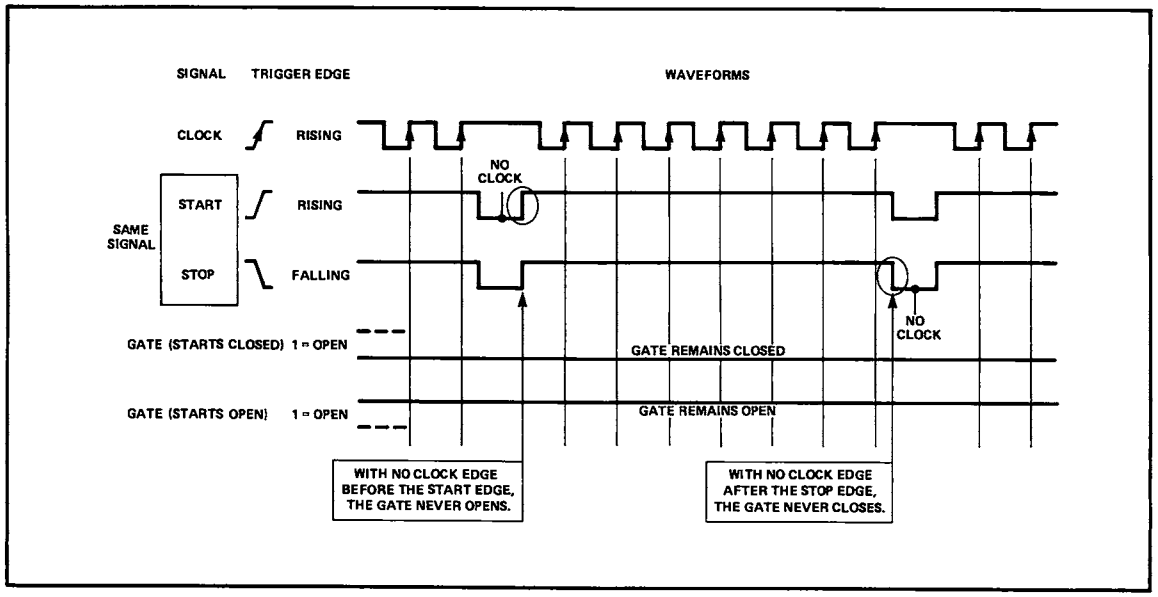
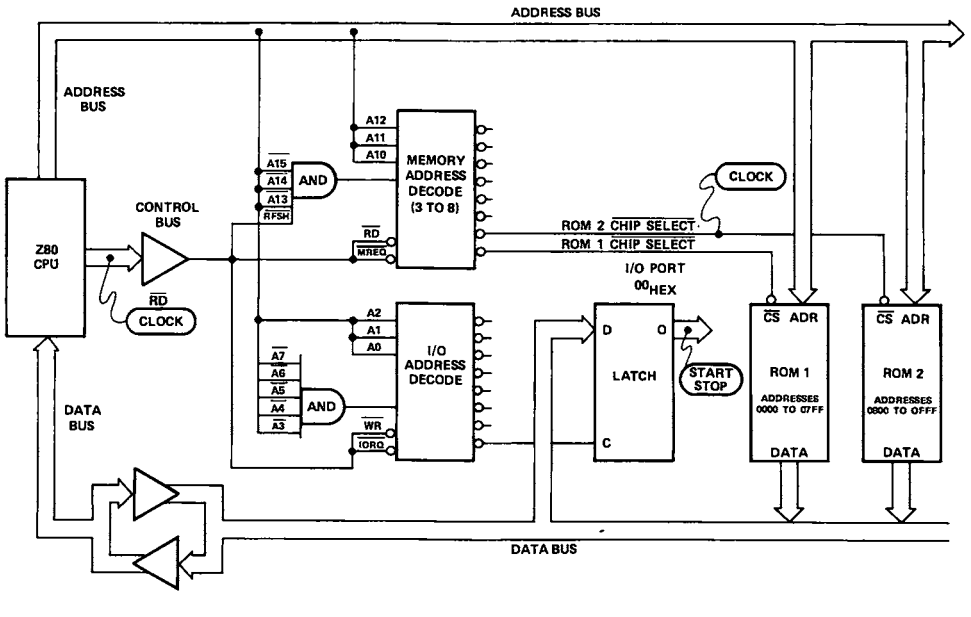


Figure 4.1

Here's an example of this situation from an actual application. In the Zilog Z80 microprocessor-based system of Figure 4.2, START and STOP are connected to a bit in a latch addressed as an I/O port. START is selected to trigger on a rising edge, and STOP triggers on a falling edge. The START and STOP edges are generated by a combination of this hardware, and the execution of the ROM stimulus program of Figure 4.3. The bit in the latch is set to logic one at the beginning of the program to open the GATE, and reset to logic zero at the end to close the GATE.



Simplified Z80 Microprocessor-Based System

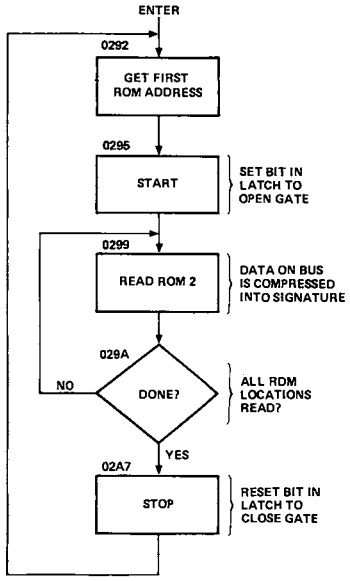
Figure 4.2

The ROM stimulus program of Figure 4.3 is stored in ROM #1, and is used to exercise ROM #2. This allows signature analysis to be used to troubleshoot ROM #2 and associated circuits such as address decoders. The program simply reads all locations of ROM #2 onto the data bus. Before the first ROM #2 location is read, the program sets the bit in the latch to open the GATE. Now all ROM #2 data that is read onto the bus will be used for a signature. After reading all ROM #2 locations, the program resets the bit to close the GATE, then returns to the beginning to open the GATE and start over.

Z80 ASSEMBLY CODE

HEX ADRS	CONTENTS	LABEL	INSTRUCTION		COMMENTS
0292	210008	START:	LD	HL, 0800	; H-L POINTS TO START OF ROM.
0295	AF		XOR	A	; A=0
0296	3C		INC	A	; A=1
0297	D300		OUT	00H, A	; OPEN GATE BY SETTING LATCH.
0299	7E	ROM:	LD	A, (HL)	; READ ROM.
029A	23		INC	HL	; INCREMENT ROM POINTER.
029B	7D		LD	A, L	; HAVE 256 BYTES OF ROM.
029C	FE00		CPI	00H	; BEEN READ?
029E	C29902		JP	NZ, ROM	; NO. READ MORE ROM.
02A1	7C		LD	A, H	; YES, BUT HAVE 2K BYTES.
02A2	FE10		CPI	10H	; OF ROM BEEN READ?
02A4	C29902		JP	NZ, ROM	; NO. READ MORE ROM.
02A7	AF	STOP:	XOR	A	; YES. A=0
02A8	D300		OUT	00H, A	; CLOSE GATE BY RESETTING LATCH.
02AA	C39202		JP	START	; JUMP TO BEGINNING OF LOOP.

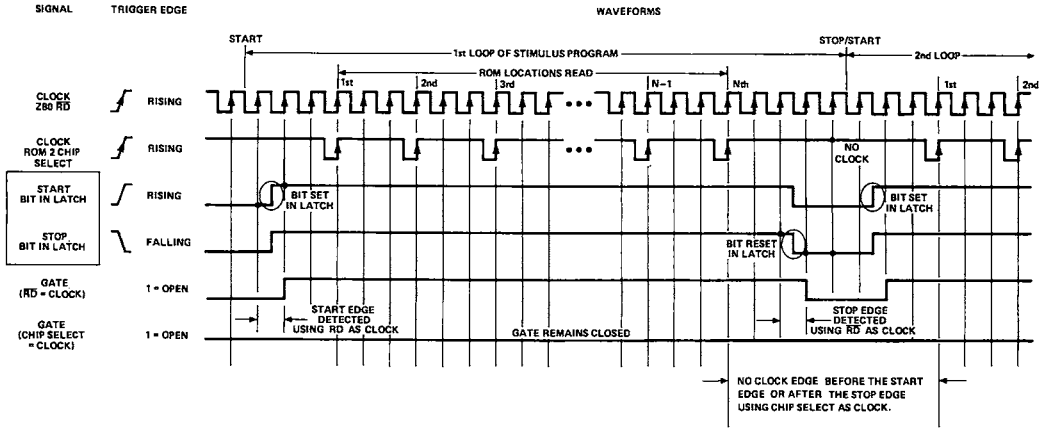
NOTE: THE LATCH THAT IS SET AND RESET TO OPEN AND CLOSE THE GATE IS LOCATED AT I/O PORT ADDRESS 00_H.



ROM Stimulus Program

Figure 4.3

If $\overline{RD}$ from the Z80 is used as a CLOCK for the signature analyzer, then the START and STOP edges will be detected, and the GATE will open and close as shown in Figure 4.4. But if the chip select for ROM #2 is used as the CLOCK (which combines the address decode for ROM #2 and the $\overline{RD}$ line from the Z80), the GATE will not open. This is because a CLOCK edge does not occur between the program steps where the bit is reset and set again. ROM #2 is not being read during that time, so the chip select line, and therefore the CLOCK, remains inactive.



ROM Stimulus Program Timing Diagram Showing Start and Stop Edge Detection

Figure 4.4

Why not connect CLOCK to $\overline{RD}$ instead of the chip select for ROM #2 and be satisfied? The answer depends on the data that should, and should not be, clocked into the signature analyzer. In this example, a $\overline{RD}$ CLOCK occurs every time the processor reads data from ROM #2. Therefore, if a signature is taken on the data bus, then it will be composed of data from ROM #2 as intended. But a $\overline{RD}$ CLOCK also occurs every time the processor fetches instructions from ROM #1 during program execution. Therefore, the signature is also composed of data from ROM #1 as well as ROM #2. This is satisfactory as long as the contents of ROM #1 is known to be good, or can be verified by some other means such as FREERUN. (The contents of ROM #1 had better be good or else the program will not execute properly, and will not exercise ROM #2.)

What if it's necessary to clock data into the signature analyzer **only** when a device is exercised by the program (in order to keep data from being entered into the signature analyzer during any other time)? For instance, imagine that this example program was used to stimulate RAM, instead of ROM, and that the RAM was accessed by both the processor and another device. This occurs in a CRT terminal where the processor stores characters in RAM, and the CRT regularly accesses the RAM to put the characters on the screen. This interaction of the processor and the CRT with RAM, is random with respect to each other. If the CLOCK is connected to $\overline{RD}$, then the data sampled by the signature analyzer will be a combination of data as accessed by the processor (during the stimulus program), and data as accessed by the CRT (during refresh of the screen). Since this combination of data is random, the signature will be unstable. To get a stable signature, the CLOCK is connected to the chip select of the RAM. The chip select combines the $\overline{RD}$ line with the processor's address decode for the RAM. The chip select line is active only when the RAM is accessed by the processor. Now the signature analyzer will sample data from the RAM only while the RAM is read during the stimulus program.

In the original ROM example, if the chip select for ROM #2 is used as a CLOCK, then only data from ROM #2 will be used for a signature. Data from ROM #1 will be ignored. However, when the chip select for ROM #2 is used as a CLOCK, then the START and STOP edges are not detected. This is because the chip select is not active, and therefore there's no CLOCK edge, between the START and STOP edge generation. To create the required CLOCK edge, a minor step could be added to the program that reads the first ROM location as shown in Figure 4.5.

Z80 ASSEMBLY CODE

	HEX ADRS	CONTENTS	LABEL	INSTRUCTION		COMMENTS
	0292	210008	START:	LD	HL, 0800	; H-L POINTS TO START OF ROM.
*	0295	7E		LD	A, (HL)	; READ FIRST ROM LOCATION.
	0296	AF		XOR	A	; A=0
	0297	3C		INC	A	; A=1
	0298	D300		OUT	00H, A	; OPEN GATE BY SETTING LATCH.
	029A	7E	ROM:	LD	A, (HL)	; READ ROM.
	029B	23		INC	HL	; INCREMENT ROM POINTER.
	029C	7D		LD	A, L	; HAVE 256 BYTES OF ROM.
	029D	FE00		CPI	00H	; BEEN READ?
	029F	C29A02		JP	NZ, ROM	; NO. READ MORE ROM.
	02A2	7C		LD	A, H	; YES, BUT HAVE 2K BYTES.
	02A3	FE10		CPI	10H	; OF ROM BEEN READ?
	02A5	C29A02		JP	NZ, ROM	; NO. READ MORE ROM.
	02A8	AF	STOP:	XOR	A	; YES. A=0
	02A9	D300		OUT	00H, A	; CLOSE GATE BY RESETTING LATCH.
	02AB	C39202		JP	START	; JUMP TO BEGINNING OF LOOP.

NOTE: THE LATCH THAT IS SET AND RESET TO OPEN AND CLOSE THE GATE IS LOCATED AT I/O PORT ADDRESS 00_H.

* EXTRA PROGRAM STEP

ROM Stimulus Program With Step Added To Detect the Start and Stop Edges

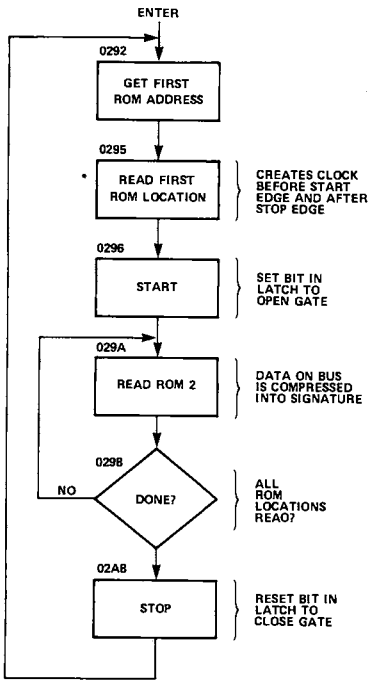
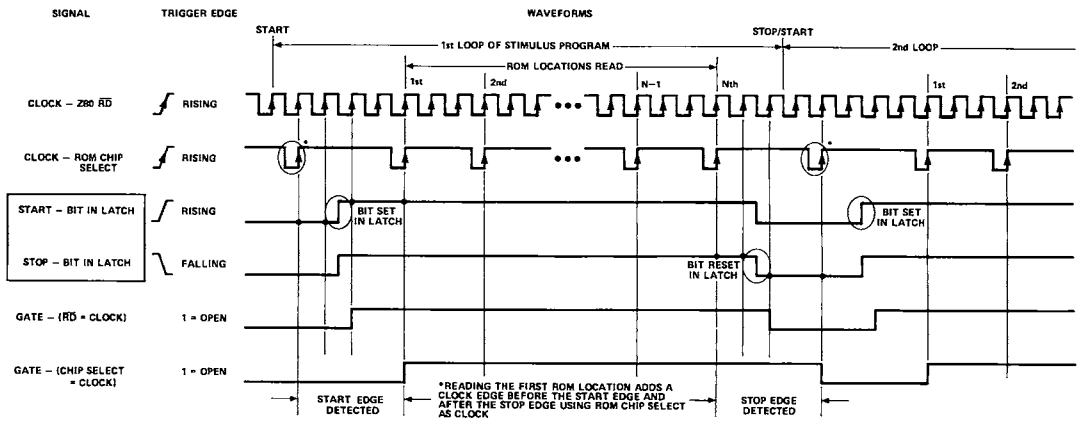


Figure 4.5

This inserts a clock pulse into the CLOCK waveform as shown in Figure 4.6. Now a CLOCK edge occurs both before and after the START and STOP edges, and the GATE opens and closes as required.

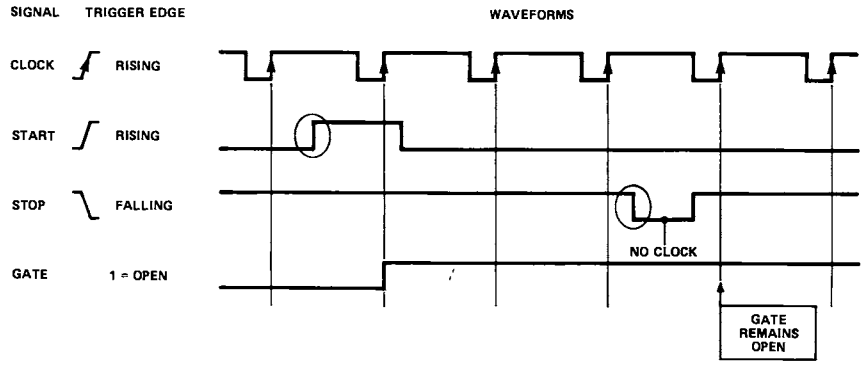


ROM Stimulus Program Timing Diagram Showing Start and Stop Edge Detection

Figure 4.6

2: The START or STOP pulse is too short for the CLOCK frequency.

In the example of Figure 4.7, the STOP pulse is too short for the slower CLOCK frequency. The GATE remains open because the STOP edge is not detected. This can occur when START and STOP are connected to address decodes, and pulsed at the beginning and end of the stimulus program. If a CLOCK is chosen that is slower than the pulses, such as a UART's transmitter clock input, then the pulses will remain undetected. To get the GATE to open and close, choose a higher frequency CLOCK, or create new START and STOP edges by controlling a bit in a latch as shown in the previous example, or using the address decode pulses to set and reset a flip-flop or latch directly.

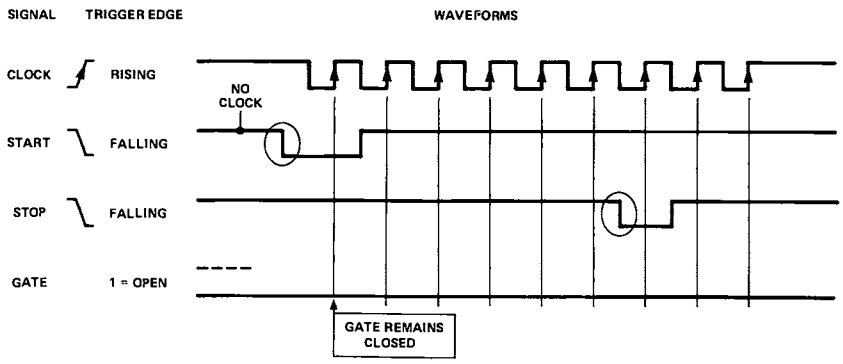


The STOP Edge Is Not Detected, Leaving the GATE Open,
Because the STOP Pulse Is Too Short For the CLOCK Frequency

Figure 4.7

3: The CLOCK turns on after the START edge.

In figure 4.8, the START edge is not detected because it occurs before the first CLOCK edge. This can happen in board test systems, where START is the signal from the board tester that turns on the CLOCK to the board under test. Here, selecting the other edge for START will allow the GATE to open. Choosing a CLOCK that is running before the START edge will also ensure its detection.

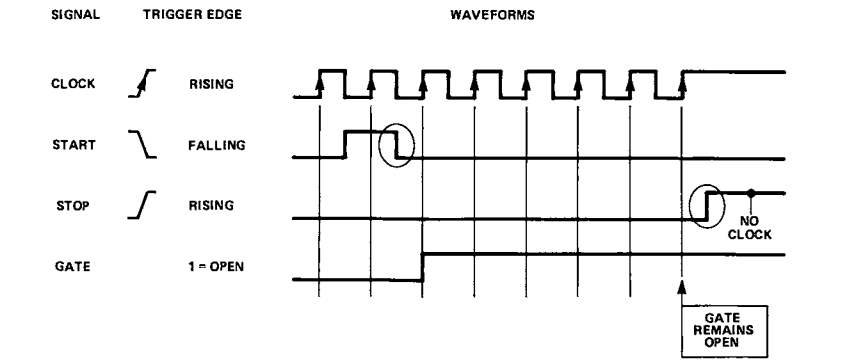


The START Edge Is Not Detected, and the GATE Never Opens, Because the CLOCK Turns On After the START Edge

Figure 4.8

4: The CLOCK turns off before the STOP edge.

In Figure 4.9, the STOP edge is not detected because it occurs after the last CLOCK edge. In this example, STOP is the terminal count output of a counter. When the terminal count occurs, the counter's clock is turned off. Connecting CLOCK to the counter's clock will cause the STOP edge to remain undetected. To ensure detection, the CLOCK should be moved to a clock that runs after the STOP edge.



The STOP Edge Is Not Detected, Leaving the GATE Open, Because the CLOCK Turns Off Before the STOP Edge

Figure 4.9

SECTION 5

Two Ways To Control The Gate

There are two basic ways to control the GATE. One way is to have the START edge open the GATE and a separate STOP edge close it. Another way is to toggle the GATE open and closed.

START Opens the GATE, STOP Closes It

Here are two examples where the START edge always opens the GATE, and the STOP edge always closes it. In the first example of Figure 5.1, this happens when:

1. START and STOP are connected to DIFFERENT signals.
2. START and STOP trigger on EITHER edge.
3. The START and STOP edges DO NOT occur between the same two CLOCK edges.

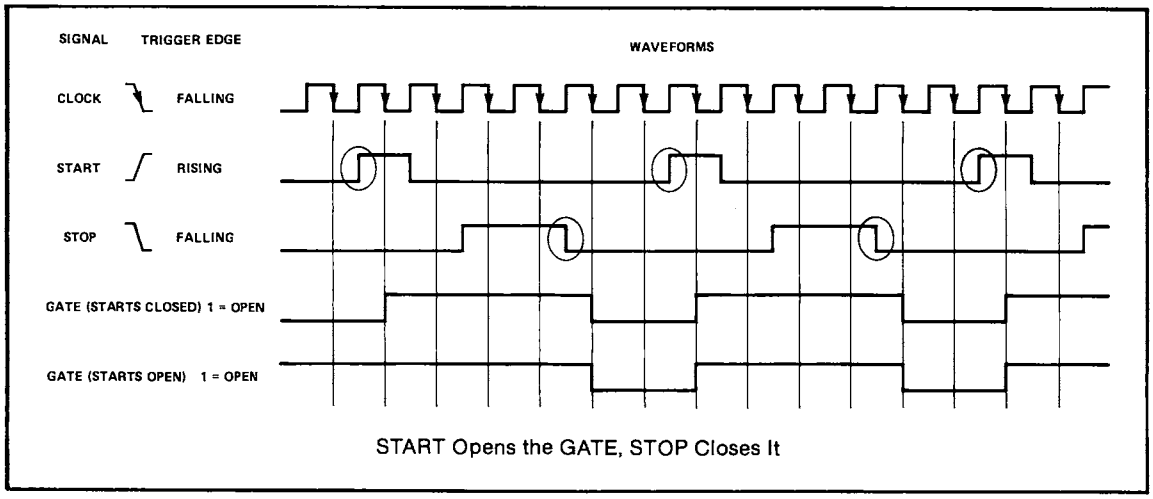


Figure 5.1

In the second example of Figure 5.2:

1. START and STOP are connected to the SAME signal.
2. START and STOP trigger on OPPOSITE edges.

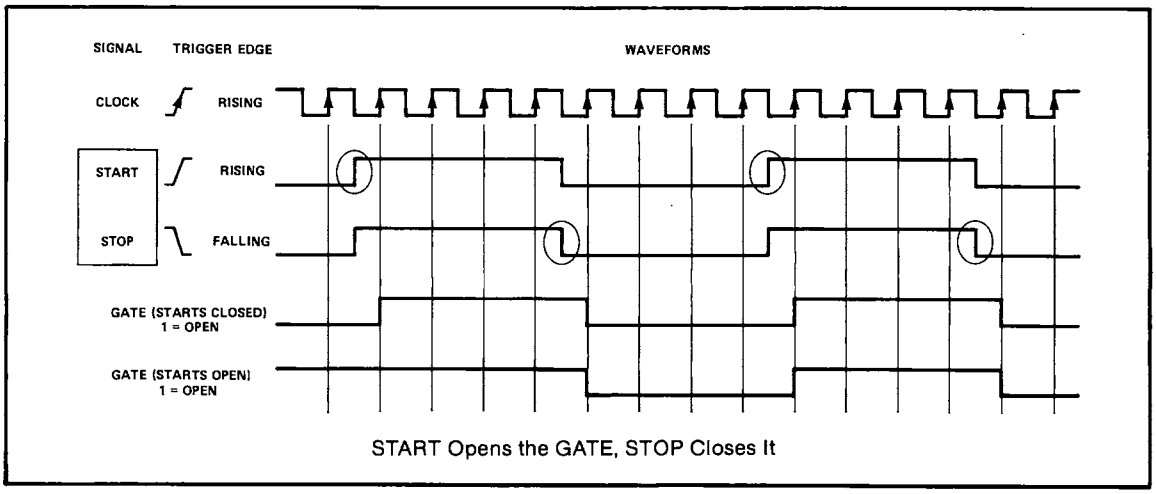
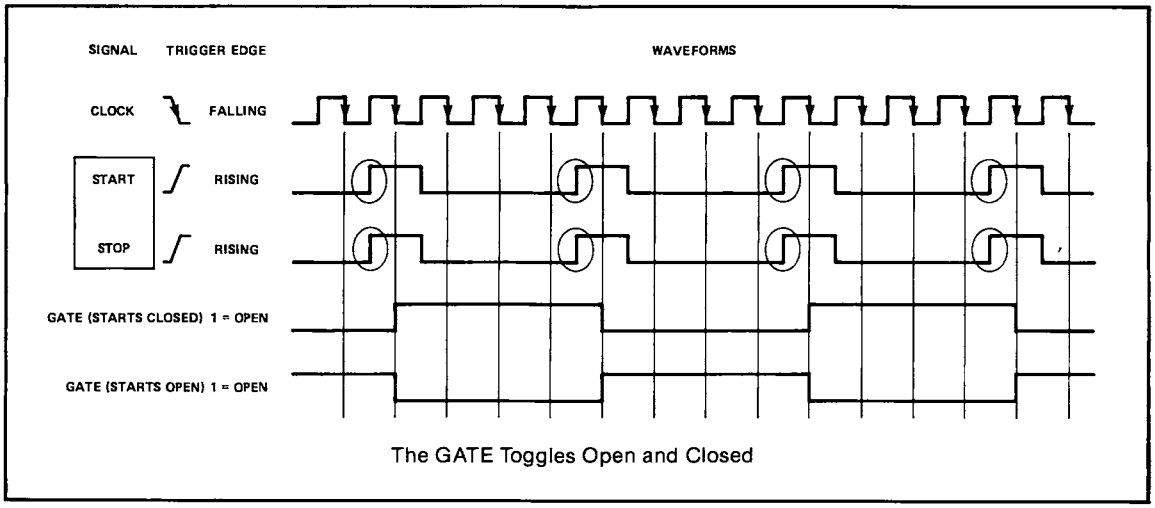


Figure 5.2

The GATE Toggles Open and Closed

In these examples, the GATE toggles because the START and STOP edges occur between the same two CLOCK edges. In the first example of Figure 5.3:

1. START and STOP are connected to the SAME signal.
2. START and STOP both trigger on the SAME edge.

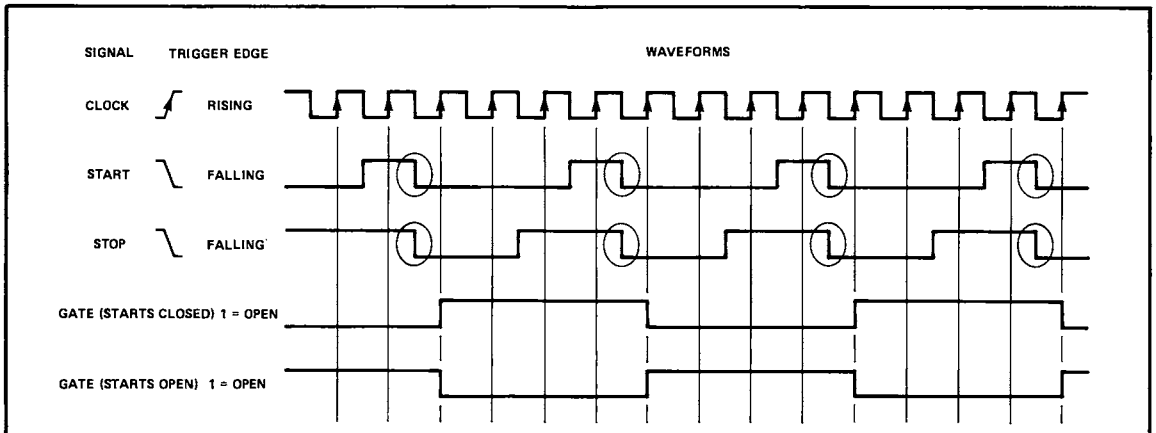


The GATE Toggles Open and Closed

Figure 5.3

In the second example of Figure 5.4:

1. START and STOP are connected to DIFFERENT signals.
2. START and STOP trigger on EITHER edge.
3. The START and STOP edges occur between the SAME two CLOCK edges.

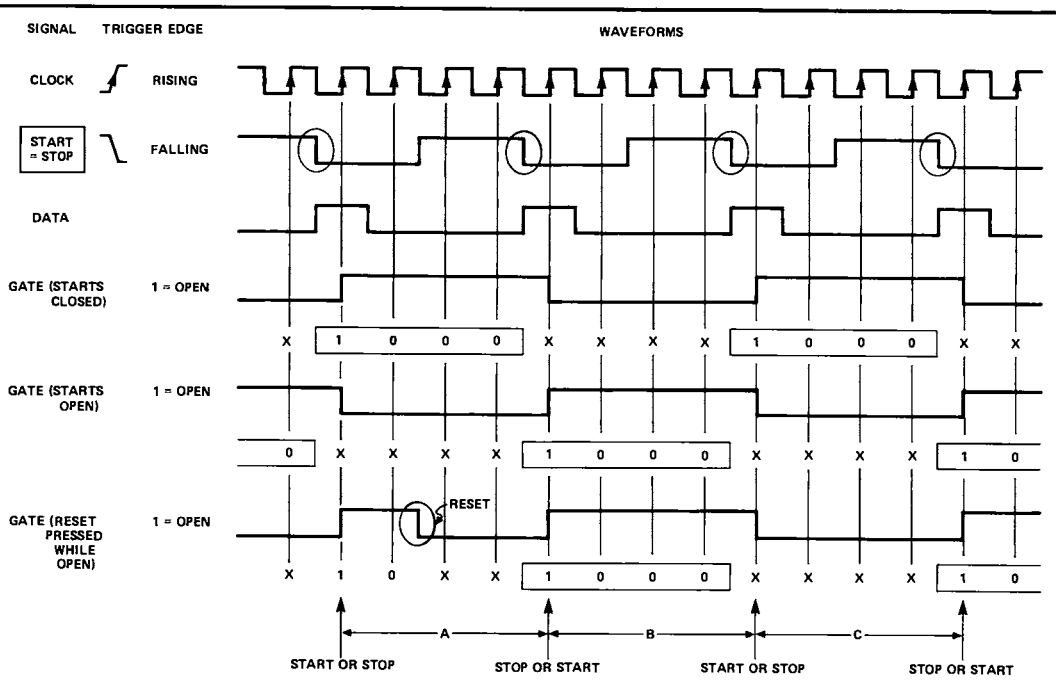


The GATE Toggles Open and Closed

Figure 5.4

When toggling the GATE, keep the same number of CLOCK edges, and the same pattern of DATA bits, between alternate trigger edges, so that the GATE can open arbitrarily at either edge. This insures repeatable signatures. Figures 5.5 through 5.7 show this.

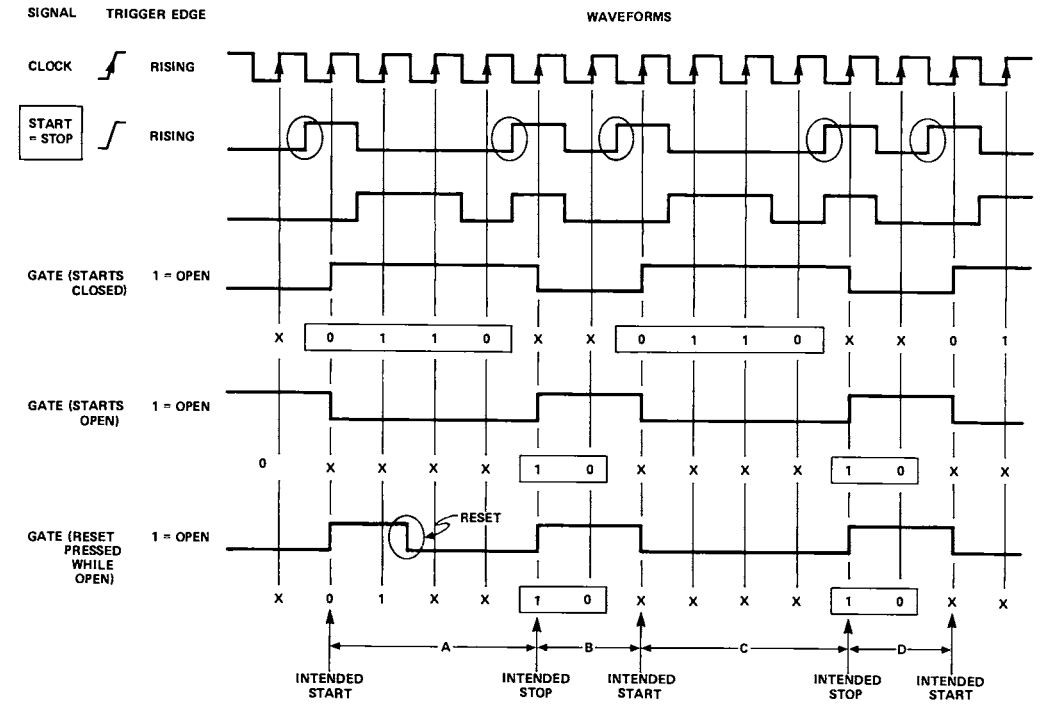
In Figure 5.5, the intended signature is always measured, regardless of the initial GATE state, even if the signature analyzer is RESET while the GATE is open. The reason? The number of CLOCK edges between any two trigger edges is identical, so that the GATE opens at any edge and still measures the same data (e.g., potential GATE times A, B, and C are identical).



The GATE Can Toggle Open Or Closed At Either Edge and Result In a Repeatable Signature, Because the Number Of CLOCK Edges and the Patterns Of Data Bits Are Always the Same Between START/STOP Edges

Figure 5.5

In Figure 5.6, the GATE is intended to be open during times A and C. However, if the GATE starts open (which can occur if RESET is pressed during A or C), then the GATE will continue to toggle at each new trigger edge, but will be open when it should be closed and vice versa. This will result in signatures taken during times B and D when the GATE is intended to be closed. The reason? The times between any two trigger edges are not the same. The GATE cannot open arbitrarily at any trigger edge, as it could in Figure 5.5, and still measure the same data.

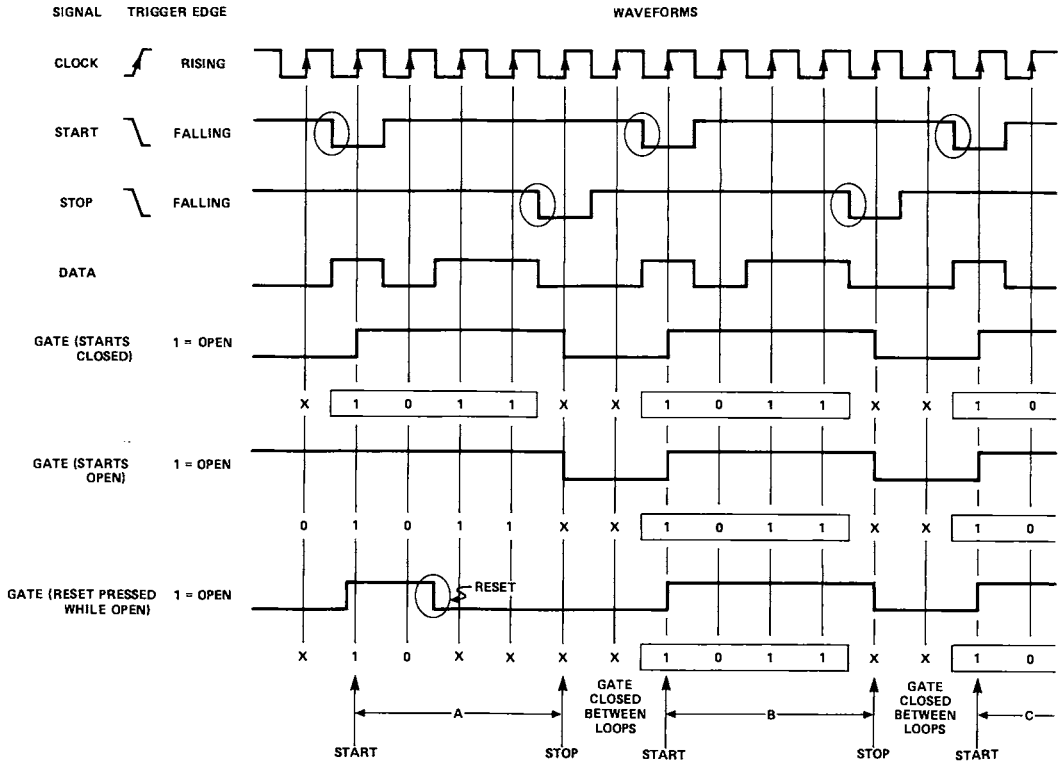


Signature Differs Depending On When GATE Opens

Figure 5.6

There are at least two ways to make the GATE open and close when intended. One way is to keep START and STOP on the same signal and continue to toggle the GATE. However, create an edge at the beginning of the stimulus program or time period of interest to open the GATE. Do not create another edge (for STOP) at the end of the program as in Figure 5.6. Instead, cause the program to return to the beginning where the same edge will then close the GATE for the next execution of the loop. The GATE will then continue to toggle as shown in Figure 5.5.

Another way is to have START open the GATE and STOP close it, as shown in Figure 5.7. Do this by connecting START and STOP to separate signals. Create one START edge at the beginning of the stimulus program or time period of interest to open the GATE. Create one STOP edge to close the GATE at the end of the program. Even pressing RESET will not change the times the GATE will open. This is especially useful if the GATE must be closed for an extended period between loops of the stimulus.



START Opens the GATE, STOP Closes It

Figure 5.7

Gate Lengths

Short GATES Mean Fast Troubleshooting

Keeping GATE cycle times under about ½ second allows the troubleshooter to quickly jump from node to node, making the total troubleshooting time very fast. Even if the probe should accidently slip from the node, then when the probe is again solidly on the node, it will be a maximum wait of one second for the correct signature. When the GATE is two seconds or longer, it could seem like a long wait for the signature. If GATES are longer than two seconds, consider splitting the stimulus program into smaller sections, so that each one runs with shorter GATES.

The easiest way to determine the GATE length is to measure it empirically. That is, run the stimulus and observe the GATE light. For GATE cycle times longer than 100 msec, the light is a direct indication of GATE length. For cycle times less than 100 msec, the light will flash at about 10 Hz.

How Long Does The GATE Need To Be Open?

The GATE needs to be open only as long as it takes to stimulate the measured node through all of its functional states. This may require stimulating the node through more than both logic states. If this is done, then the signature analyzer will compress all the data on a node into a compact signature. The signature then contains all the information about the correct functioning of the device driving the node. Be sure that the CLOCK will sample all of the functional states of the node, otherwise a functional error, such as a single bit error, could go undetected. See Figure 6.1.

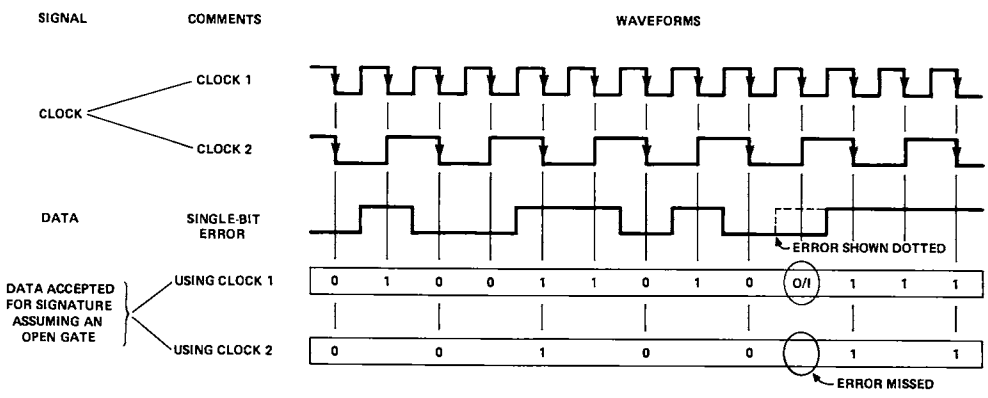


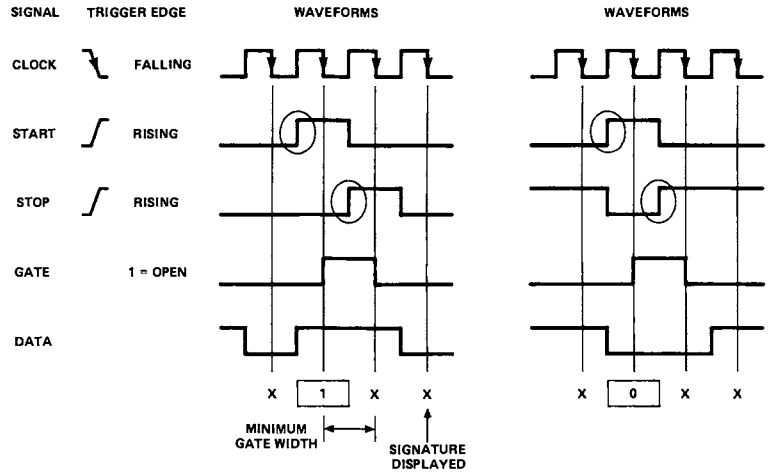
Figure 6.1

Single bit errors are guaranteed to result in a signature change for that node, as long as the error bit has been clocked into the signature analyzer. Multiple bit errors will result in a signature change for that node with a probability of 99.998%. In other words, functional failures in the device driving the node will be detected with a probability of 99.998%, worst case, assuming the error bits have been clocked into the signature analyzer, and that the device has been exercised through all of its functions.

It's unnecessary to artificially lengthen the GATE to increase the accuracy of error detection. The signature analyzer's accuracy of error detection is 100% for 16 or less data bits clocked into the analyzer. For more than 16 bits, the accuracy remains at 99.998%, worst case. This means that GATES can be as short or as long as necessary. Application Note 222-2 contains a complete discussion of the accuracy of error detection.

Minimum and Maximum Timing For An Open GATE

At least one CLOCK edge must occur between the START edge and the STOP edge, in order to open then close the GATE. The GATE opens for one CLOCK edge, while one data bit is clocked into the signature analyzer. The new signature will be displayed at the SECOND CLOCK edge after the STOP edge (i.e., after the GATE closes). See Figure 6.2.



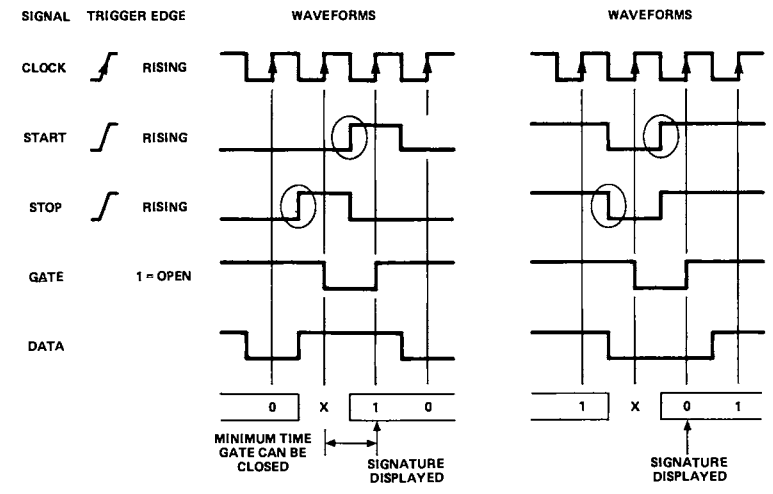
Two Examples Of GATE Open For Minimum Time

Figure 6.2

Any number of CLOCK edges can occur between a START edge and a STOP edge. There is no upper limit. Even the number of possible signatures (65,536) is not an upper limit. The signature analyzer will continue to gather a signature as long as the GATE is open. However, the new signature will not be displayed until the SECOND CLOCK edge after the STOP edge (i.e., not until the GATE closes).

Minimum and Maximum Timing For A Closed GATE

At least one CLOCK edge must occur between the last STOP edge and the next START edge, in order to close then open the GATE. The GATE will close for that CLOCK edge, and the data bit at that edge will not be used for the signature. The signature will be displayed at the SECOND CLOCK edge after the STOP edge (i.e., when the GATE opens again). See Figure 6.3.



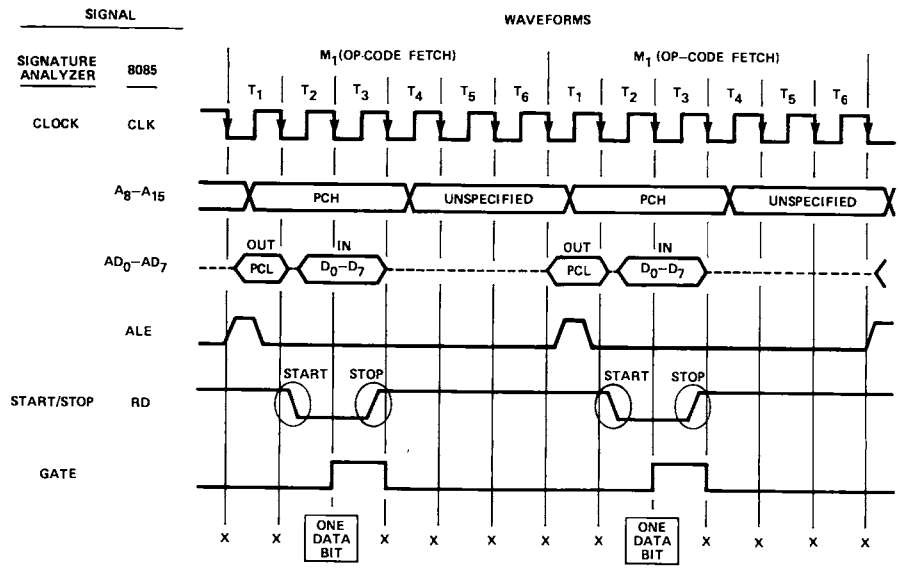
Two Examples Of GATE Closed For Minimum Time

Figure 6.3

Any number of CLOCK edges can occur between the last STOP edge and the next START edge. There is no upper limit. The signature analyzer will not gather a signature as long as the GATE is closed. The last signature will be displayed at the SECOND CLOCK edge after the STOP edge (i.e., after the GATE closed), and will continue to be displayed as long as the GATE is closed.

Creating A Synchronous Logic Probe

A minimum GATE of one CLOCK cycle allows the signature analyzer to be used as a synchronous logic probe. This can be very useful in determining the logic state of a node at an exact instant in time. The need for a synchronous logic probe measurement was discovered when trying to FREERUN microprocessors with multiplexed address and data buses, such as the Intel 8085. When the 8085 FREERUNS, an address is placed onto the multiplexed bus by the 8085 during the first half of each instruction fetch cycle. When the 8085 switches the bus to read the instruction, a NOP or similar instruction is forced onto the bus, so that the microprocessor simply increments to the next memory address, and so on. The address incrementation acts as a stimulus to memory devices within the system, so that signature analysis can be used for troubleshooting. (For a thorough discussion of FREERUN, see Application Note 222.) If there is a fault on the multiplexed bus, such as a short to ground, or short to another line, then two things could happen that would cause the 8085 not to stimulate the memory devices properly. First, the instruction could be changed by the fault, so that the microprocessor no longer increments the address bus sequentially. Second, the address lines could be changed by the fault such that they would not stimulate all memory devices. It would be easy to find the fault if it could be determined what instruction was being read into the microprocessor. To find out, it's necessary to separate address output, from data input, on the multiplexed bus. This can be done by sampling the bus only during the time the 8085 is reading it. Figure 6.4 shows how the signature analyzer can be used to display each bit of the instruction as if it were static on the bus. The signature will either be 0000 or 0001, corresponding to the logic state of the bus at the moment it was sampled.



The signature analyzer can be used as a synchronous logic probe to display each bit of the instruction as if it were static on the bus. Here, the 8085 is FREERUNNING, so the instruction is the same for each M_1 cycle.

Figure 6.4

For most other cases, GATE times will be longer than one bit. Otherwise, the node has not been allowed to cycle through all of it's possible functional states.

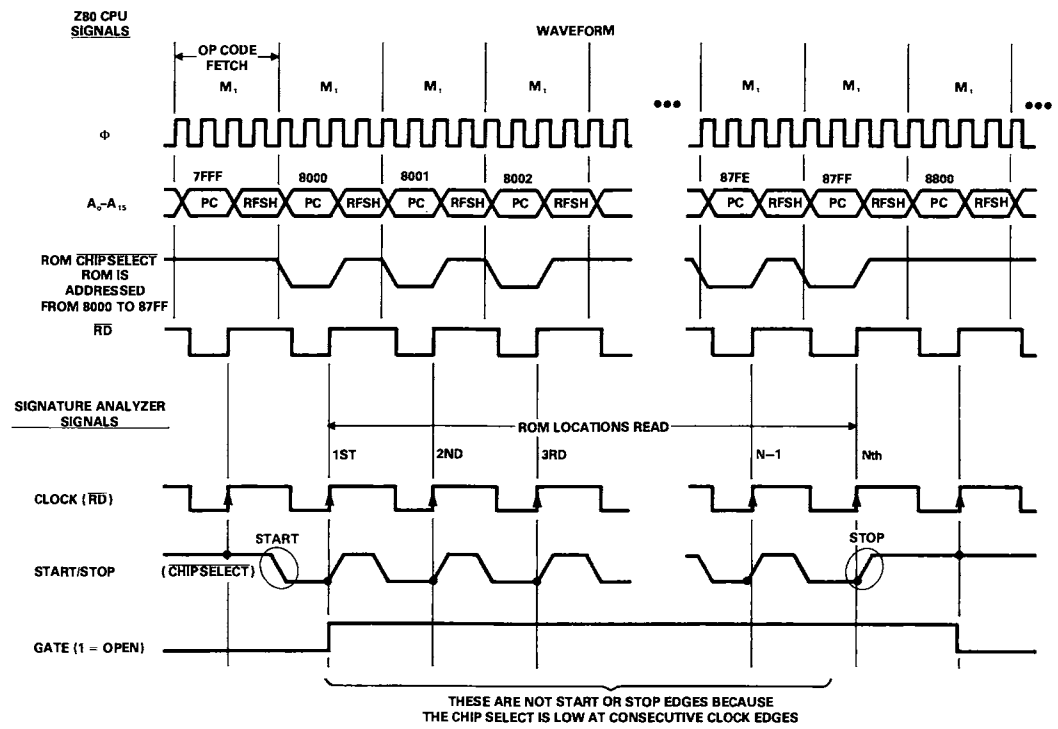
Multiple START and STOP Edges

If there is another START edge between any START edge and a STOP edge, then there are multiple START edges. Similarly, if there is at least one additional STOP edge between any STOP edge and a START edge, then there are multiple STOP edges. This assumes two things. First, that the START and STOP inputs are connected to separate signals. Second, that the START and STOP edges are detected by the CLOCK. The next paragraph shows how to determine if the second condition is true. If both conditions are true, then the first START edge after the GATE closes will open the GATE. Further START edges have no effect. The first STOP edge after the GATE opens will close the GATE. Further STOP edges have no effect. The GATE opens again at the first (next) START edge after the GATE closes. Resetting the signature analyzer, or using the HOLD feature, may affect the operation of the GATE. See Section Eight.

Are There Really Multiple START or STOP Edges?

Signals sometimes seem to have more than one edge when they really don't. For example, when FREERUNNING a microprocessor such as the Zilog Z80, START and STOP are frequently connected to ROM chip selects or their equivalents. This creates a GATE that's open only while ROM data is on the bus, allowing ROM's and associated circuits to be verified and troubleshot.

The chip selects become active at each address, and then inactive again during address changes. This creates multiple edges that at first seem to be multiple START and STOP edges. But when $\overline{RD}$ is used as a CLOCK, only one START and STOP edge actually occur. See Figure 7.1.



What Sometimes Seem To Be Multiple START/STOP Edges Really Aren't

Figure 7.1

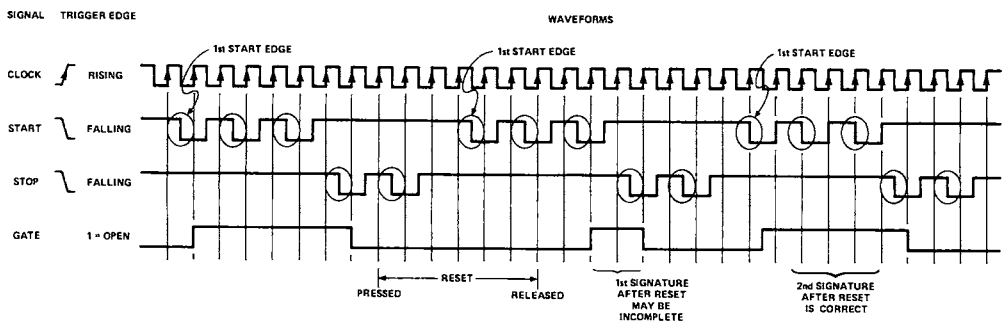
Reset and Hold

Resetting The Signature Analyzer

The HP Model 5004A Signature Analyzer is reset by pressing the RESET button on the DATA probe. The signature analyzer is usually reset for one of the following reasons:

1. It's a convenient way to cause the signature analyzer to display the V_{CC} (all ones) signature without touching the DATA probe to V_{CC}.
2. It's the only way to get the signature analyzer to take another signature while it's in the HOLD mode.
3. It's a way to reset the display and close the GATE, just after placing the DATA probe on a node, so that the next signature displayed is the correct one. This is usually done when GATE cycle times are long.

When there are multiple START edges, the first signature displayed after reset may not be the correct one. As shown in Figure 8.1, when the signature analyzer is RESET, the GATE closes, and will remain closed, until the next START edge. If the GATE opens again at the first of multiple START edges, then the first signature will be a correct (complete) one. If the GATE reopens at one of the multiple START edges other than the first one, then the first signature will be a partial (incorrect) one. In either case, the second signature after RESET will always be correct (complete), assuming the GATE repeats and the signature analyzer is not in the HOLD mode.



The First Signature After RESET May Be Incorrect
If There Are Multiple START Edges

Figure 8.1

The HOLD Feature

The HOLD feature causes the HP 5004A Signature Analyzer to take only ONE signature after RESET. To use the HOLD feature, press the HOLD button IN. Place the DATA probe on the node, and then press RESET for a new signature. The next signature displayed will be held in the display, even if the probe is removed from the node. This feature is handy when the probe must be removed from the node temporarily to see the display. This can happen when troubleshooting a product that's in an awkward location. The HOLD feature also makes it convenient to document signatures in the troubleshooting procedure. After the signature is captured in the display, the probe can be set down so that the signature can be written. HOLD can even be used to capture signatures of single-cycle events that occur when the product is first turned on. Note that a second signature is not taken in the HOLD mode. If the HOLD feature is intended to be used, then START signals must have only one edge. If this is done, then the "held" signature will always be correct (complete).

SECTION 9

Getting Correct, Repeatable, and Stable Signatures

While creating an SA circuit stimulus, a few circuit conditions can cause unstable or unrepeatable signatures. This section outlines their causes and how to recognize them. The remaining guidelines show how to eliminate them from the SA stimulus design so that incorrect signatures measured during troubleshooting accurately indicate a fault.

Correct and Incorrect Signatures

Correct signatures are defined as the signatures documented in the troubleshooting procedure for a product. An incorrect signature is defined as a signature displayed on the signature analyzer, that doesn't match the documented one for the node being probed. If the correct signatures are repeatable, then measuring an incorrect signature during troubleshooting will accurately indicate an incorrect waveform for that node. This allows the troubleshooter to quickly back-trace through the circuit following incorrect waveforms to the faulty node. The fault is found at the point where back-tracing further results in correct signatures again. For instance, if the signature for an output of a device is incorrect, signatures are taken on the inputs for that device. If the input signatures are correct, then the fault has been isolated to the output node. If any input signature is incorrect, back-tracing continues along that signal path.

A troubleshooter is concerned only if the signatures are correct or incorrect. He must believe that the signatures documented for the product are correct. That is, he assumes that a known good product will yield all of the signatures as documented in the troubleshooting procedure. The person that creates and checks the signature analysis stimulus is concerned about assuring that signatures to be documented in the troubleshooting procedure are really correct. Correct signatures must be repeatable.

The creator of the stimulus also is concerned about whether all possible product faults will yield incorrect signatures. Two things determine this. First, the accuracy of error detection of the measurement, and second, the extent of the functional stimulus of the product. The signature analysis measurement gives a 99.998% worst-case probability of detecting an error, as long as the error appears in the waveform for that node, and the error bit(s) have been clocked into the signature analyzer. The error will appear in the waveform only if the functional stimulus causes it to be there.

Repeatable and Unrepeatable Signatures

Repeatable signatures are defined as getting the same signature each and every time the signature measurement is made, no matter when it is made, if:

1. Identical nodes in the product are measured.
2. Identical known good products are tried.
3. Identical signature measurements are performed.
4. Identical signature analysis stimulus is run.
5. Identical signature analyzers are used (i.e., signature analyzers with compatible characteristics. See Section 2 for a description of the characteristics.)

Unrepeatable signatures are defined as getting a different signature any time the same node in a known good product is measured, in the same way, when running the same stimulus. One of the goals of the signature analysis circuit stimulus creator, is to eliminate unrepeatable signatures from the product.

Making Sure Signatures Are Repeatable

Unrepeatable signatures usually occur because some circuit element, such as a flip-flop or RAM, has not been initialized before the SA stimulus begins. They also can occur if the GATE toggles open at the wrong time. The following steps will help uncover repeatable signatures in a product. They should be performed to verify ALL signatures in a product BEFORE they are considered correct.

1. Turn the product under test off then on again.
2. Reset the signature analyzer, or touch the DATA probe to ground or V_{CC} before measuring a node.
3. Take signatures in as many other known good products as time and effort permits.

Turning the product under test off then on again determines if any devices are not being initialized before the SA stimulus begins. For example, a flip-flop that is not set or reset at power-on or at the beginning of the SA stimulus loop can cause signatures to differ. RAM also must be initialized with some known pattern before the SA stimulus begins if any nodes associated with that RAM are measured during the execution of the stimulus.

Resetting the signature analyzer determines two things. First, that the GATE toggles open at the intended time. See Section Five. Second, that the node being sampled is always in the third state. If a node is always in the third state, the signature will either be all zeroes, or the V_{CC} signature, depending on the last valid logic state of the previous node measured. If an all zeroes signature has been documented for the three-state node, WITHOUT a notation that the signature could ALSO be the V_{CC} signature, then the signature will appear to be unrepeatable. (This is a special case where the signature is unrepeatable, but can only be one of two possible signatures.) Reset causes the signature analyzer to use a logic one as the last valid logic state. This is the same as if the probe had been touched to V_{CC} before measuring the three-state node. See Sections Eight and Eleven.

Taking signatures in as many known good products as possible determines if all uninitialized devices have been eliminated from the signature measurement. It also determines if all recommended operating conditions of the signature analyzer have been met. Using another signature analyzer can also help determine this. For example, if the specification for the setup time is being violated, but the signature analyzer actually is performing better, then another signature analyzer may help uncover this. See Appendix A for setup and hold times.

Here's an example of a situation that can cause unrepeatable signatures. On the data bus of a FREERUNNING microprocessor system, there may be data from ROM, RAM and I/O devices. The ROM data is always the same, but RAM data will differ each time the system is turned off then on again. This is because the processor has no way to initialize RAM by storing a pattern in it. Therefore, the data from RAM and other uninitialized devices must be eliminated from a signature measurement of the data bus. This will result in the same DATA (ROM data only) being measured each time the GATE is open and therefore results in repeatable signatures. This can be done in two ways. One way is to move START and STOP to ROM chip selects (or their equivalents), to create a GATE that is open only while ROM data is on the bus. Another way to do this is to choose a CLOCK that samples data on the bus only when ROM data is present.

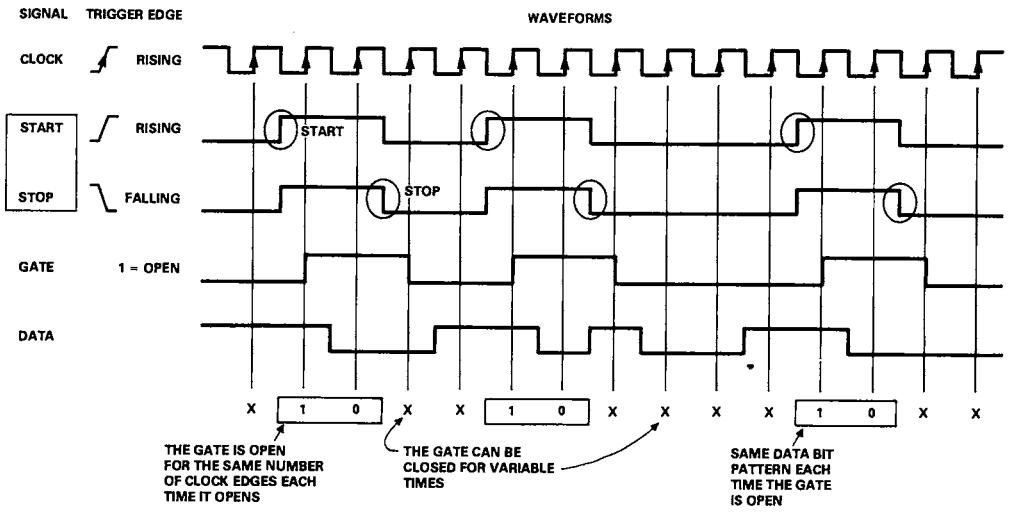
Stable and Unstable Signatures

The definition of stable and unstable signatures depends on the measurement of two signatures in a row. Stable signatures occur when two adjacent signature measurements result in the same signature for all measurements. Unstable signatures occur when the signatures of any two adjacent measurement cycles differ. An unstable signature is indicated by the UNSTABLE SIGNATURE LIGHT. The light turns on for approximately 100 milliseconds whenever a signature taken during ANY GATE cycle differs from the signature taken during the immediately previous GATE cycle. If unstable signatures occur for every GATE cycle, the light will blink at $\leq$10 Hz. Unstable signatures are not of concern while the signature analyzer is in the HOLD mode. This is because only one signature is taken after RESET is pressed. However, the UNSTABLE LIGHT will flash as the signature display changes from 0000 after RESET, to a captured signature other than 0000.

Sometimes the four-digit signature display will indicate an unstable signature by slowly and/or constantly changing, or changing so fast that it becomes unintelligible. However, with fast GATES and an intermittant circuit fault, intermittantly unstable signatures can occur without a noticeable signature display change. In those cases, the UNSTABLE SIGNATURE LIGHT will be the only indication of an unstable signature and the intermittant circuit.

Eliminating Unstable Signatures

Unstable signatures usually occur because the GATE length created by the stimulus is not the same from loop to loop, or the response of the circuit to the stimulus is not the same from loop to loop. For stable and repeatable signatures, the GATE must be open for the same number of CLOCK edges each time it opens. The DATA bits sampled by CLOCK edges during an open GATE must also be identical (same logic levels and time relationship) from one open GATE to the next. While the GATE is closed, the number of CLOCK edges can vary as well as the DATA pattern. See Figure 9.1. To check DMA cycles that transfer the same data each time, connect START to a line that signals when DMA begins. In a Zilog Z80-based system, this might be HOLD ACKNOWLEDGE. This control line from the Z80 signals to an external device, such as a DMA controller, that it now can take control of the bus. Connect STOP to a similar signal that occurs at the end of the DMA cycle. See Figure 9.1.



Some Conditions For Stable and Repeatable Signatures

Figure 9.1

Unstable signatures can sometimes be documented for nodes if the signature changes by only a few numbers. For instance, if a signature is unstable, but only changes between two numbers, then both can be documented. Then, either one measured during troubleshooting will indicate correct circuit function, while any other signature will indicate a fault.

SECTION 10

About Noise

Most signal noise does NOT result in incorrect, unstable, or unrepeatable signatures, because the signature analyzer synchronously detects logic state changes on START, STOP and DATA at CLOCK edges. This section shows the effects of two different types of noise, defined below, on each of these four inputs.

Synchronous Noise Definition

Synchronous noise is any noise on a signal that can be predicted with respect to a CLOCK edge. It is usually caused by clocking data synchronously, through a digital system. It consists of momentary transitions, or multiple logic state changes, through the logic threshold opposite to the signal's defined stable state, between two CLOCK edges. These transitions occur after the CLOCK edge that caused them, but are absent at the next CLOCK edge. Here, "at the CLOCK edge" means the signal is at its defined state during the data setup time of the signature analyzer.

Asynchronous Noise Definition

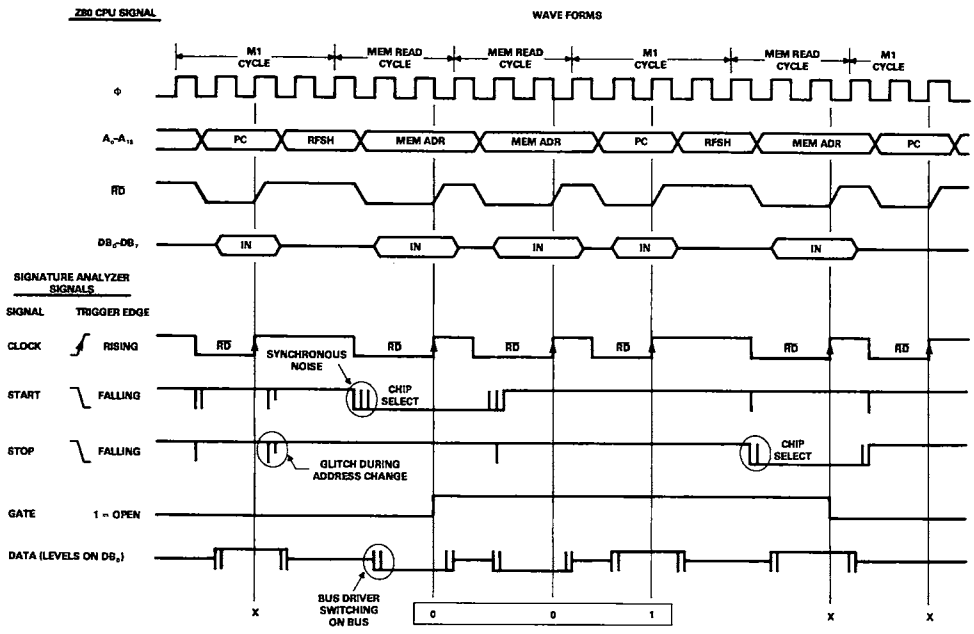
Asynchronous noise is any noise on a signal that occurs randomly with respect to the CLOCK. This noise is usually referred to as a "glitch" and can occur even at a CLOCK edge. It consists of momentary transitions, or multiple logic state changes, through the logic threshold opposite to the signal's defined stable state, between two CLOCK edges. These transitions will be detected as an incorrect logic state only if they occur exactly at the CLOCK edge of the signature analyzer.

Noise On START, STOP and DATA

Synchronous noise on START and STOP is ignored by the signature analyzer, as shown in Figure 10.1. This is a real advantage when using address decoders or chip selects as START and STOP, even though they frequently are noisy during address changes. But since a CLOCK can be used that occurs only after the address bus is stable, the noise does not get detected and will not open or close the gate incorrectly at the noise edges. The gate opens or closes only at a CLOCK edge when the START or STOP edge has been synchronously detected.

Synchronous noise on the DATA input is also ignored. This is an advantage when taking signatures on high-current data bus lines which are noisy while drivers switch on and off the bus. Only valid logic states at CLOCK edges will be used as DATA bits for the signatures.

Asynchronous noise on START, STOP and DATA can cause incorrect, unstable, or unrepeatable signatures if it occurs at the CLOCK edge. This is because it can violate the setup time of the CLOCK as defined in the Recommended Operating Conditions of Appendix A.



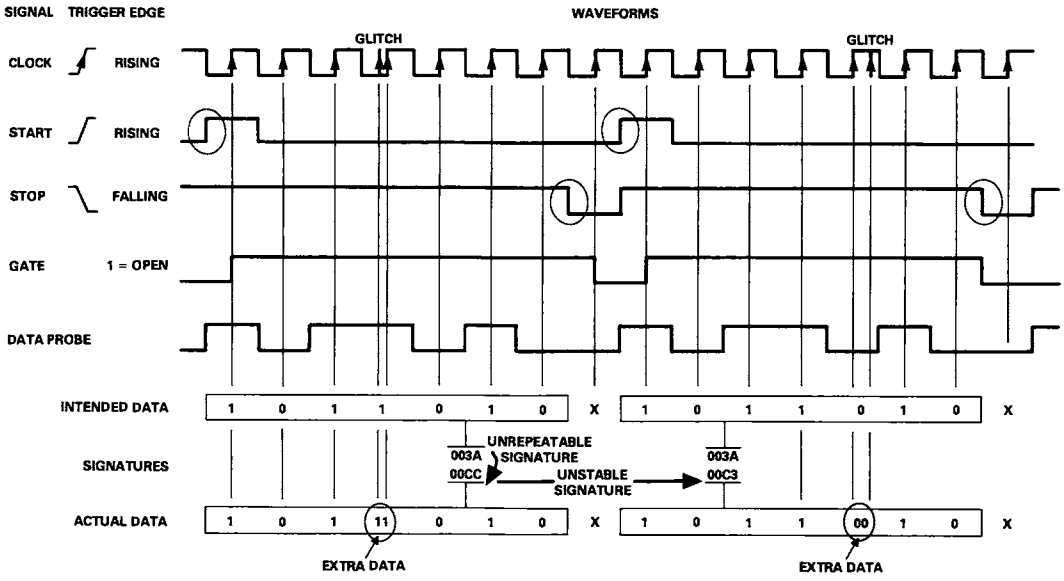
Synchronous Noise Is Ignored In This Z80-Based Microprocessor System

Figure 10.1

CLOCK Noise

Any transition through the single logic threshold of the CLOCK input is defined as a CLOCK edge. Synchronous or asynchronous noise on the CLOCK results in extra CLOCK edges to the signature analyzer, as shown in Figure 10.2.

If the extra CLOCK edges occur while the gate is closed, then the signature may or may not be affected depending on the activity on START. If the extra CLOCK edges do not detect a START edge, then the gate will remain closed and the signature will be correct. However, if the extra CLOCK edges do accidentally detect a START edge, then the gate will open early. The gate length will then vary resulting in an unstable or unrepeatable signature. If the extra CLOCK edges occur while the gate is open, the length of the gate will change, resulting in unstable or unrepeatable signatures.



Noise On The Clock Input Results In Unstable and Unrepeatable Signatures

Figure 10.2

Ground Noise

The ground leads of the signature analyzer should be connected to circuit grounds that are free of excessive noise. Normally only the GATING POD ground lead need be connected to the circuit under test. However, when measuring high-frequency DATA (usually above 5 MHz), an additional ground lead connected to the DATA probe ground point will help reduce any noise induced through a long ground lead path.

About Three-State Nodes

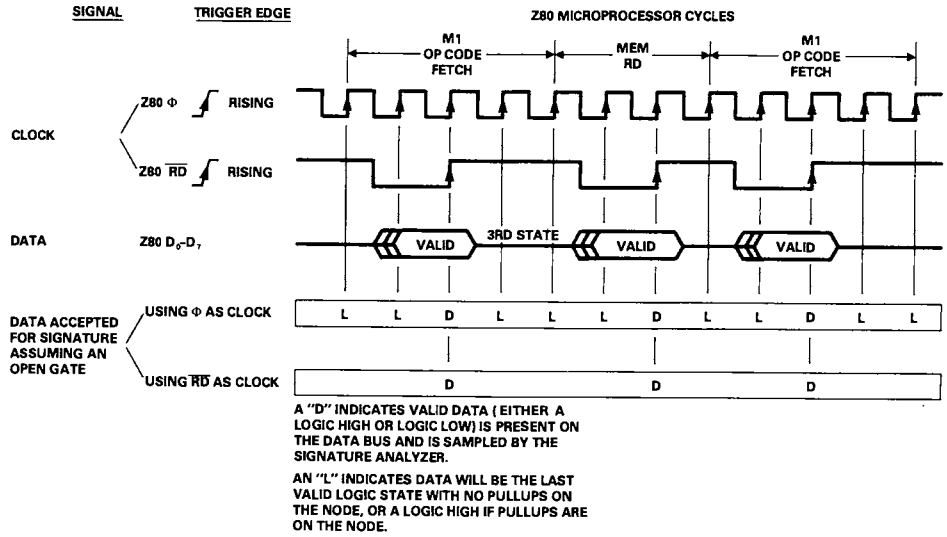
When three-state nodes are probed with a Hewlett-Packard Signature Analyzer, stable and repeatable signatures can be obtained, even though the TTL level of the node is not defined during the third state. Two things allow this to happen:

1. A CLOCK can be chosen to sample data on a three-state node only when it is in a defined state. When using such a CLOCK, the node is NOT sampled during the third state.
2. If such a CLOCK is not used, then the DATA probe and the front-end logic of the signature analyzer will define a third state sampled by the CLOCK as the last valid logic level on the node before it went into the third state.

Here's the details.

Using A CLOCK That Samples Only Defined DATA

In most applications, it's easy to choose a CLOCK that samples a three-state node only when valid data is present, and not when the node is in the third state. For example, a microprocessor's data bus can enter the third state between op code fetches, memory references, and I/O operations as the bus direction is turned around from writing to reading a device. Usually there is a control output from the microprocessor (e.g., a read or write line), that defines when data is valid on the bus. Using this control line as the CLOCK synchronizes the signature analyzer, so that only valid logic levels on the bus are sampled and third-state levels are ignored. See Figure 11.1.



A Clock Can Be Chosen To Sample DATA On a Three-State Node Only When It Is In a Defined State

Figure 11.1

Using A CLOCK That Samples A Node During The Third State

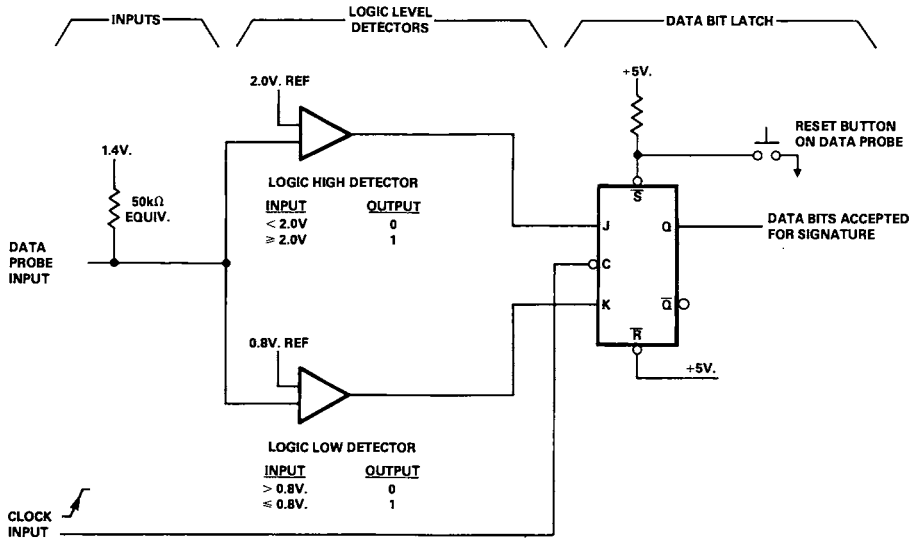
Sometimes it's necessary to use a CLOCK that samples a node when it's in the third state. This can happen when a synchronous CLOCK is not available or when a node remains in the third state for a long time. The signature then depends on whether or not there's an external pullup (or pulldown), on the node.

Signatures For Three-State Nodes Without Pullups

Stable and repeatable signatures can be obtained on three-state nodes without pullups using a Hewlett-Packard Signature Analyzer. A pullup does not need to be added to the node during system design, nor does a pullup need to be added to the DATA probe during troubleshooting. This is because the signature analyzer will continue to use the last valid logic state as a DATA bit when the node enters the third state. There are three parts to the signature analyzer that allow this to happen:

1. An internal reference inside the DATA probe pulls the node to 1.4 volts during the third state.
2. Dual threshold logic level detectors that define logic LOW as 0.8 volts or less, logic HIGH as 2.0 volts or greater, and anything in between LOW and HIGH as the third state.
3. A clocked DATA bit latch.

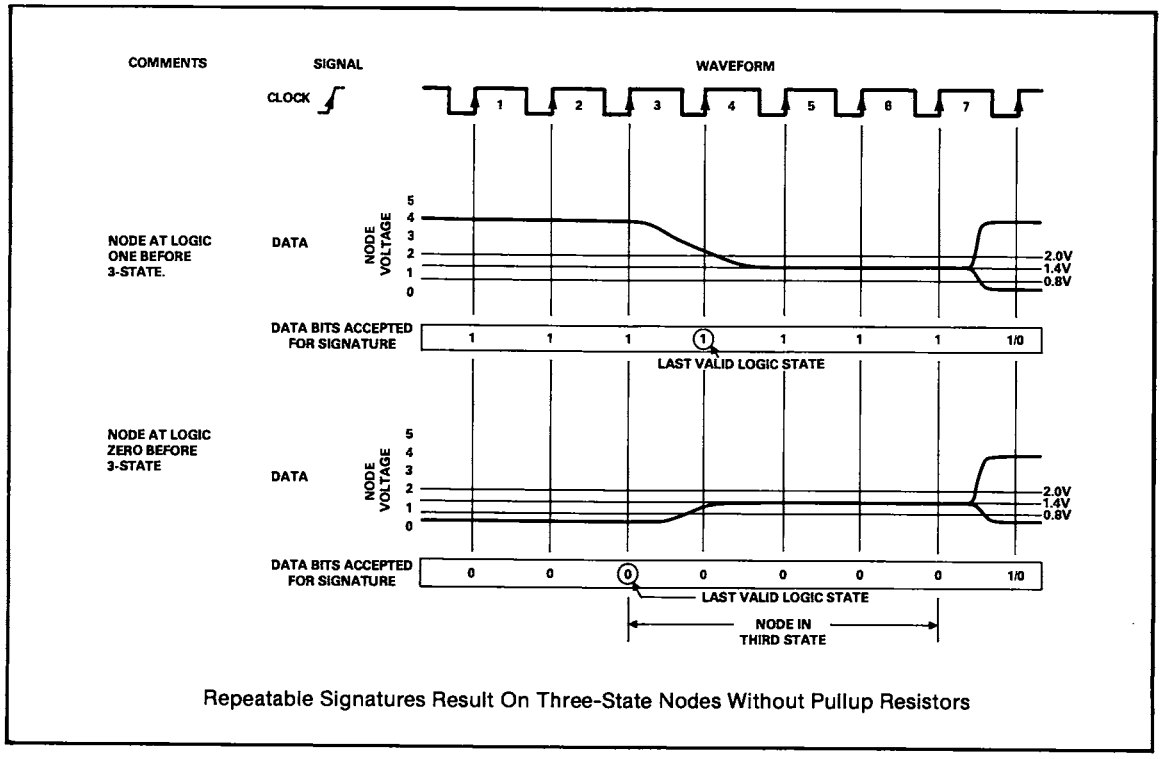
Figure 11.2 shows a simplified diagram of the input circuit of the HP Model 5004A Signature Analyzer. The DATA bits accepted for a signature are latched into a flip-flop at each CLOCK edge. If the voltage at the DATA probe input is 2.0 volts or greater, the logic level converter on the J input to the flip-flop causes the flip-flop to set to logic one at the next CLOCK edge. If the probe voltage is 0.8 volts or less, the converter on the K input causes the flip-flop to reset to logic zero at the next CLOCK edge. If the probe voltage is between 0.8 volts and 2.0 volts, neither logic level converter is active and the flip-flop neither sets nor resets.



Front-End For the HP Model 5004A Signature Analyzer

Figure 11.2

When the node enters the third state, the DATA probe's internal reference pulls the node to 1.4 volts. If the last valid logic state of the node was a high, then the reference pulls the node from logic high to 1.4 volts during the third state and does not allow the node's voltage to drop into the logic zero detection level of 0.8 volts or less. Similarly, a logic low is pulled up to 1.4 volts and a logic high is never detected. This is shown in Figure 11.3. In other words, the last valid logic state remains in the DATA latch and continues to be used for a signature while the node is in the third state. If a node remains in the third state during the entire measurement, then the signature will be all zeroes or the V_{CC} signature depending on whether the last bit on the last node measured was a one or a zero.



Repeatable Signatures Result On Three-State Nodes Without Pullup Resistors

Figure 11.3

Getting Vcc Signatures Without Probing Vcc

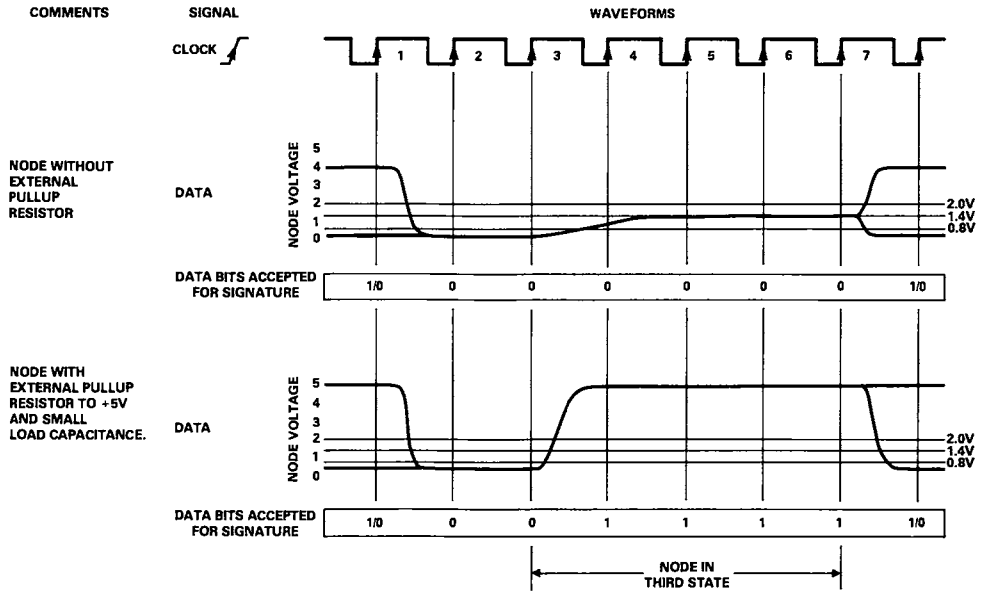
The RESET button on the DATA probe sets the DATA latch to a logic one so that if the probe is left floating (no node is being probed or the node is always in the third state), then all ones are clocked into the signature analyzer as if the probe were placed on +5VDC. This results in a Vcc (or +5V, or all ones) signature.

Noise On A Three-State Node

The signature analyzer's treatment of three-state nodes without pullups has been optimized, and proven with several years' experience. The DATA probe input has high enough impedance to reduce circuit loading of the node during the active state and low enough impedance to reduce most noise induced on the node during the third state. This treatment results in stable and repeatable signatures in most cases. However, because of their high impedance, three-state nodes are susceptible to excessive noise induced through crosstalk with high current carrying lines. Although unlikely, excessive noise could cause the logic threshold detectors to switch states exactly at a CLOCK edge and cause an erroneous DATA bit to be sampled resulting in an incorrect, unstable or unrepeatable signature. Section Ten shows how noise on any input affects the signature measurement.

Signatures For Three-State Nodes With Pullups

Stable and repeatable signatures can also be obtained for three-state nodes with pullup resistors in most cases. This is because the pullups on three-state nodes will override the signature analyzer's internal reference and pull the node up to logic high during the third state. This results in logic ones being used as DATA bits during the third state instead of the last valid logic state, as was the case for a node without pullups. See Figure 11.4.



Repeatable Signatures Can Be Obtained On Three-State Nodes With Pullups and Small Load Capacitance

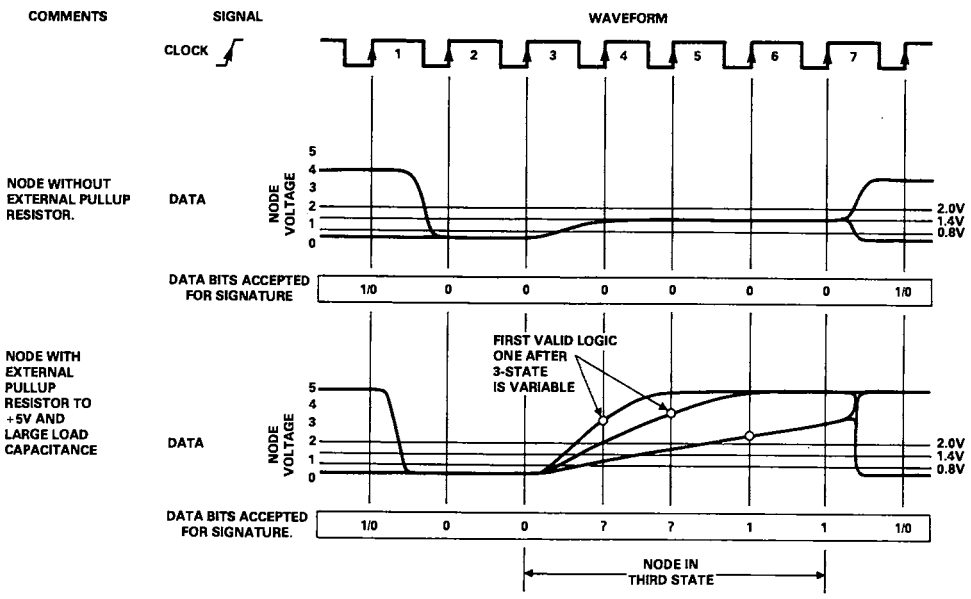
Figure 11.4

However, it's not always possible to add pullups to a three-state node. This may be because the pullups may load the circuit too much when the node is active. Or perhaps the pullups were not designed into the circuit and they cannot be added later. If there are pullups on the node, or if an external pullup is placed on the DATA probe of the signature analyzer, be careful that the signal risetimes remain fast enough.

Nodal Capacitance And Pullups Can Cause Slow Risetimes

Without pullups on the node, it doesn't matter how long it takes for the DATA probe to pull the node to 1.4 volts. Risetimes are not a concern since the DATA bit latch will not change state during the third state. However, external pullups or pulldowns on the node, combined with large nodal capacitance, can cause slow signal risetimes to logic one (or fall times to zero), resulting in erroneous DATA being detected. This can cause incorrect, unstable, or unrepeatable signatures in some cases.

In the case of pullups on the node, the node's voltage will eventually be detected as a logic high during the third state, but the point at which the data will be ones instead of zeroes depends on how fast the node's voltage rises to +5VDC. This is shown in Figure 11.5. The node's RC time constant will determine the rise time. If the risetime is always the same, or if the signal is guaranteed to rise above the logic high threshold before the next CLOCK edge (including time for setup), then a stable signature will result. However, the RC time constant depends on a fixed pullup resistor (if it is designed into the board), or a variable one (if it is placed on the DATA probe during troubleshooting). The RC time constant also depends on the node capacitance that could vary from unit to unit, or even on the same unit over time, or if a part is replaced. The capacitance can also vary, particularly on a large bus, due to the number of devices on the bus, the variable capacitance of each device due to manufacturing variations, the bus length, including backplanes and optional PC boards, and so on.



It's Harder To Get Repeatable, Stable Signatures On Three-State Nodes With Pullups and Large Node Capacitance

Figure 11.5

If unstable or unrepeatable signatures result on a node with a pullup resistor, first verify that a slow risetime is the problem. If it is, then try to find a CLOCK that will sample only valid DATA on the node and will ignore the third state. If this is not possible, then calculate the node's worst case capacitance and choose a pullup resistor that will insure the node's voltage will be above 2.0 volts before the next CLOCK. Be sure all devices on the bus have sufficient drive capability with the new resistor.

If risetime does not seem to be a problem, then check if there is excessive noise synchronous to the CLOCK. If so, eliminate the source of the noise, or choose a CLOCK that will ignore it.

APPENDIX A

Hewlett-Packard Model 5004A Signature Analyzer

Any edge for CLOCK, with any trigger edge selected for START or STOP, in any combination			Min.	Nom.	Max.	Unit	See Fig.
Setup time, t_{s-}	t_{ss}	START	25			ns	A.1
		STOP	25				
	t_{sd}	DATA	15				
Hold time, t_{h-}	t_{hs}	START	0				
		STOP	0				
	t_{hd}	DATA	0				
Clock frequency			0		10	MHz	
Width of CLOCK pulse, t_w high or low			50			ns	
Input impedance to 1.4V		START		50		kΩ	–
		STOP		50		kΩ	
		CLOCK		7		pf	
		DATA		7		pf	
V_{ih} High-level input voltage		DATA	1.6	2.0		V	A.2
V_{il} Low-level input voltage				0.8	1.2	V	
V_{ref} Single logic threshold range		START	0.8	1.4	2.0	V	A.3
		STOP	0.8	1.4	2.0	V	
		CLOCK	0.8	1.4	2.0	V	
Hysteresis band around single logic threshold		START		.1		V	
		STOP		.1		V	
		CLOCK		.1		V	
Overload protection, all inputs		continuous	−150		150	V	–
		intermittent	−250		+250	V	
		≤1 min.		250		VAC	

Table A.1—Recommended Operating Conditions

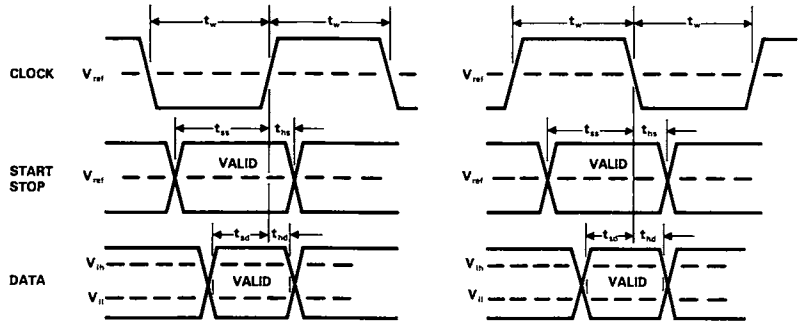


Figure A.1—Voltage Waveforms

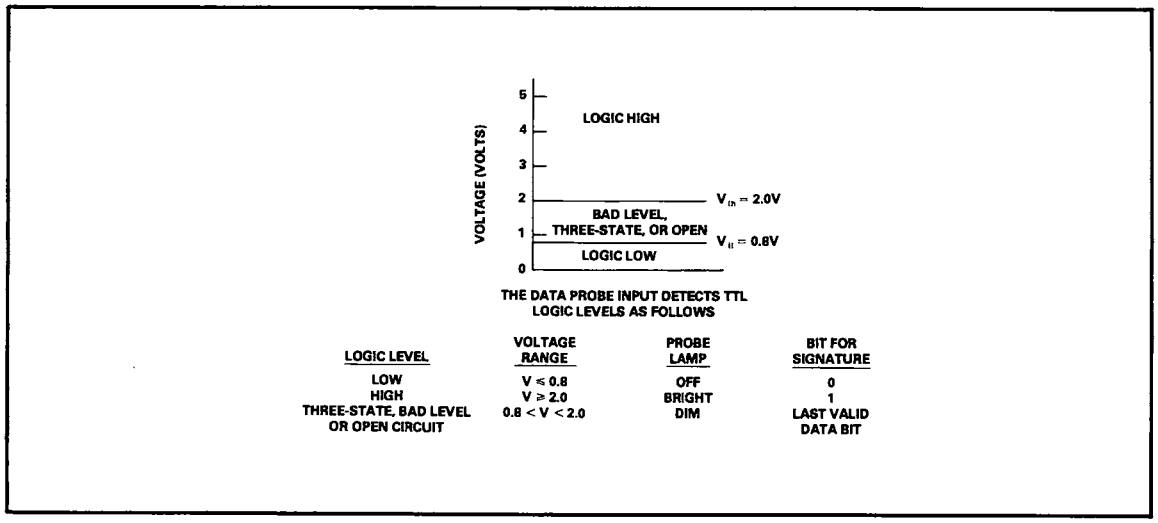


Figure A.2—Data Probe Dual Thresholds (Nominal)

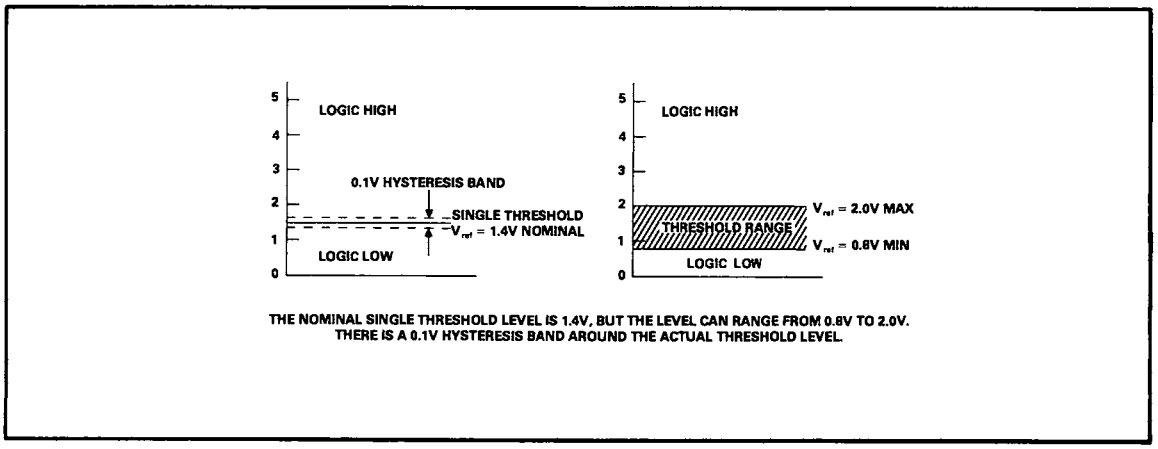


Figure A.3—START, STOP and CLOCK Single Thresholds

HEWLETT PACKARD

For more information, call your local HP Sales Office or nearest Regional Office: **Eastern** (201) 265-5000; **Midwestern** (312) 255-9800; **Southern** (404) 955-1500; **Western** (213) 970-7500; **Canadian** (416) 678-9430. Ask the operator for instrument sales. Or write Hewlett-Packard, 1501 Page Mill Road, Palo Alto, CA 94304. **In Europe:** Hewlett-Packard S.A., 7, rue du Bois-du-Lan, P.O. Box, CH 1217 Meyrin 2, Geneva, Switzerland. **In Japan:** Yokogawa-Hewlett-Packard Ltd., 29-21, Takaido-Higashi 3-chome, Suginami-ku, Tokyo 168.

02-5952-7595 **JANUARY 1981** **PRINTED IN USA**