An IEEE 1149.1 Based Logic/Signature Analyzer in a Chip

Lee Whetsel
Texas Instruments Incorporated
Senior Member Technical Staff
Semiconductor Group

SCTA034

TEXAS INSTRUMENTS

AN IEEE 1149.1 BASED LOGIC/SIGNATURE ANALYZER IN A CHIP

Lee Whetsel
Member Group Technical Staff
Texas Instruments Inc.
P.O. box 869305, M/S 8407
Plano, Texas 75086

ABSTRACT

This paper describes an IEEE 1149.1 based test IC that emulates the functions of logic and signature analysis test instruments. These ICs can be used at the board or multi-chip module level to provide an embedded method of monitoring circuits at-speed. This paper assumes the reader has a basic understanding of the IEEE 1149.1 standard[1,2].

INTRODUCTION

Test instruments, such as logic analyzers, have traditionally been used to test the at-speed interaction of functioning ICs on board designs. These test instruments gain access to the circuit under test by physically contacting the circuit using a probing mechanism. The use of these test instruments to test functioning circuitry can reveal timing sensitive and/or intermittent failures that would otherwise not be detectable in a nonfunctional test environment. The ability of these test instruments to synchronize up with and observe the at-speed operation of electronic circuits, have made them an invaluable asset in a wide range of testing applications.

With the increasing use of high-speed, state-of-the-art integrated circuits in combination with the miniaturized substrates on which they are assembled, the physical access between an external test instrument and a circuit under test is being severely reduced and in some cases completely eliminated. New test approaches such as the IEEE 1149.1 boundary scan standard provide a method to regain electrical access to miniaturized circuits and substrates through the use of an IC resident test port and boundary scan architecture. The 1149.1 standard provides an excellent method of testing the structural integrity of the wiring interconnects between ICs on a common substrate[3,4]. In addition, 1149.1 can be used to test individual ICs while they are in a nonfunctional mode.

However, the 1149.1 standard cannot be used effectively for at-speed functional testing of ICs or circuits. The standard does provide a test instruction, referred to as Sample/Preload, that allows the boundary scan register to take a snapshot sample of the data entering and leaving a functioning IC. While a specific application of the Sample/Preload instruction has been described[5], its general use suffers due to problems not addressed in the standard[6].

One problem with the Sample/Preload instruction is that there is no prescribed method of synchronizing the sample operation with the operation of the host IC. Sampling data asynchronously is a hit and miss proposition that serves no useful purpose. Another problem is that there is no prescribed method of qualifying when to execute the sample operation in a functioning system. Sampling data synchronously but at random does little to support testing. The solutions to these potentially challenging problems is left up to the user of 1149.1.

A new approach, therefore, is required to provide a method of functionally testing the at-speed operation of miniaturized electronic circuits. The approach described in this paper overcomes the loss of functional test access to state-of-the-art circuitry through the use of an IC designed specifically for embedded at-speed testing applications. This test IC is a member of TIs SCOPE™ family of testability components[7,8], and is referred to as a Digital Bus Monitor (DBM). The DBM can be implemented in board or multi-chip module designs and coupled to critical functional IC bus signals to provide a method of non-intrusively monitoring the functional operation of the circuit.

When the DBM is enabled via serial input from the 1149.1 test bus, it synchronizes up with the functional circuitry to perform data trace and/or data compaction on the at-speed data flow between the functional ICs of the circuit. Following the test, the trace data and/or signature collected can be accessed via the 1149.1 test bus for processing.

The advantage offered by the DBM is that it enables the use of traditional at-speed test approaches without having to physically probe the electronic circuit being tested. Also, since the DBMs are embedded in the product and accessible via the 1149.1 test bus, the tests they provide are reusable throughout the life cycle of the product. For example the DBMs can be used for at-speed testing of the product at the assembly site, then later reused during other phases of the products life cycle such as; hardware/software integration and debug, at-speed system testing, environmental chamber testing, and field testing and diagnostics.

DBM ARCHITECTURE

The DBM is a 28 pin device designed in compliance with the rules set forth in the IEEE 1149.1 boundary scan standard. The DBM supports all required 1149.1 boundary scan test instructions as well as a rich set of special test instructions supporting its data trace and signature analysis test operations. The data trace and signature analysis operations can operate from control input from the 1149.1 test bus or from control generated internal to the DBM.

The DBM IC architecture of Figure 1 has input pins for receiving 16 data inputs (D0-15), 3 clock inputs (CK1,CK2,CK3), an event qualification input (EQI), an output pin for outputting an event qualification output (EQO), and a bidirectional pin for inputting or outputting a polynomial input/output signal (PIO). In addition, the DBM is coupled to the 1149.1 test bus via the test data input (TDI), test data output (TDO), test

mode select (TMS), and test clock (TCK) pins, to provide a standard serial interface for input and output.

The DBM consists of a 1Kx16-bit trace memory for test data storage, a 16-bit parallel signature analysis (PSAR) register for test data compaction, a 16-bit comparator for pattern detection, an event qualification module (EQM) for at-speed test control, a programmable clock interface (PCI) for clock conditioning, a boundary scan register for interconnect testing, and a 1149.1 circuit block consisting of a control register for instruction amplification, a bypass register for abbreviating the scan path through the DBM, an instruction register for instruction storage, and the 1149.1 test access port (TAP).

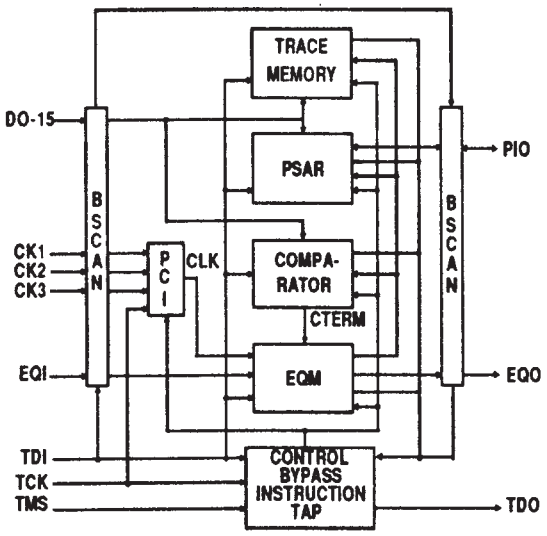

FIGURE 1 DBM Architecture

1149.1 Circuitry

The TAP along with the instruction, bypass, and boundary scan registers are test circuits required in the 1149.1 standard. The TAP receives the 1149.1 TMS and TCK bus control signals and provides the interface through which an external test bus controller can input control to shift data through the IC from the TDI input to the TDO output, or effectuate an 1149.1 controlled test operation. The instruction register provides storage for test instructions shifted into the IC. The bypass register provides a single bit scan path through the IC when 1149.1 testing is not being performed. The boundary scan register consists of a series of cells associated with each IC input and output pin. When placed in their test mode, the boundary cells can be accessed via the TAP to output test data from the IC's outputs and receive test data into the IC's inputs, in accordance to the IEEE 1149.1 Extest instruction.

Trace Memory

The trace memory receives the 16-bit data bus, serial input from the TDI pin, control input from the TAP and EQM, and outputs serial data to the TDO pin. The trace memory is a 16-bit by 1024 location static RAM. When disabled the trace memory powers down and requires minimum power to maintain its contents. When enabled the trace memory can be used to store the 16-bit data input to the DBM in response to control input from either the TAP or EQM. The DBM can receive and store 16-bit data words at up to 40Mhz.

The trace memory of Figure 2 consists of a 1Kx16-bit static RAM, a 16-bit data input/output register (DIOR), a 10-bit address register (AREG), and read/write control circuitry. The read/write control circuitry receives control input from the TAP and EQM, and outputs control to the DIOR, AREG, and RAM. The selection of which control input is used, TAP or EQM, is determined by the instruction in the 1149.1 instruction register.

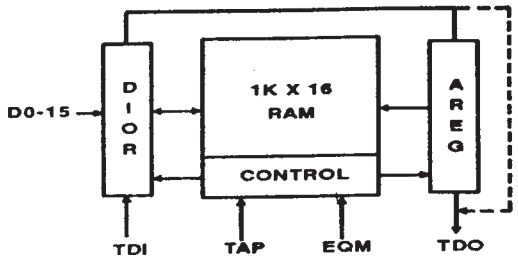

FIGURE 2 Trace Memory

When trace operations are controlled from the 1149.1 TAP, data input on the data bus is clock into the DIOR register then transferred into the addressed RAM location during each TCK while the TAP is in its RT/IDLE state. After each data pattern is written to RAM, the AREG is incremented to address the next RAM location.

When trace operations are controlled from the EQM, data input on the data bus is clocked into the DIOR register during each CLK output from the PCI. However, the data is only transferred into the RAM when the EQM is qualified, via input from either the internal CTERM or external EQI signal, to execute one of its eight selectable test protocols. After a data pattern is transferred into the RAM, the AREG is incremented to address the next RAM location.

Trace Memory Read/Write Modes

The RAM has two modes of serial read/write access, 1149.1 scan access mode and direct memory access mode (DMA). The RAM access mode is selected via an instruction loaded into the instruction register. In the 1149.1 scan access mode, the TAP inputs or outputs RAM data via the DIOR and AREG registers by cycling through multiple data register scan operations. During each scan operation the AREG addresses a RAM location to be written to or read from via the DIOR. Each RAM read or write cycle requires one 1149.1 data register scan operation.

In the DMA mode, the DIOR is configured as either a serial input/parallel output register for DMA write operations or a parallel input/serial output register for DMA read operations, and the AREG is configured as a counter. During DMA write operations, a DMA controller embedded in the TAP, outputs control to cause serial data to be shifted into the DIOR from the TDI input. After the DIOR has received a predetermined number of bits, the DMA controller inputs control to cause the RAM

to accept the parallel data from the DIOR, the AREG to increment, and the DIOR to continue inputting serial data. These steps of serially inputting data to the DIOR, writing the data into the RAM, and incrementing the AREG are repeated until the RAM is full.

During DMA read operations, the DMA controller inputs control to cause the DIOR to parallel load data from the RAM. After the data is loaded, the DMA controller inputs control to cause the AREG to increment, and the DIOR to shift data out via the TDO output. When the last bit is shifted out, the DMA controller repeats the steps of parallel loading RAM data into the DIOR, incrementing the AREG, and shifting data out of the DIOR until the RAM data is read.

The reason for developing the DMA mode for RAM access is that it significantly reduces read/write access time to embedded memories over using the 1149.1 scan approach, especially in a system environment where multiple devices precede and follow the target device. The time savings is brought about by the fact that the DMA mode allows a single, but extended 1149.1 data register scan cycle to stream serial data continuously into or out of a device's embedded memory without interruption.

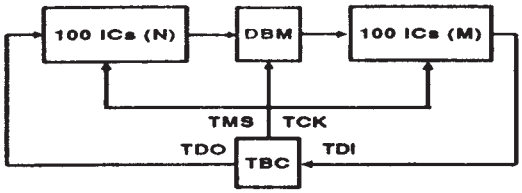

FIGURE 3 Trace Memory Access Example

To illustrate the time savings using the DMA mode to access the DBM's memory over using the 1149.1 data register scan mode, calculations are made on the example system scan path environment illustrated in Figure 3. The scan path consists of a test bus controller (TBC), a DBM, 100 1149.1 compliant ICs (N) between the TDO output from the TBC and the TDI input to the DBM, and 100 1149.1 compliant ICs (M) between the TDO output from the DBM and TDI input to the TBC.

In both the 1149.1 (1) and DMA (2) RAM access mode calculations, assume a 5Mhz TCK and that the ICs preceding and following the DBM are in their Bypass mode. Since the scan path is symmetrical the read and write access times are identical for both DMA and 1149.1 access modes, so only a read access calculation will be performed on each. A comparison of the calculated RAM access times, shows the DMA access time (3.3ms) to be 93% less than the 1149.1 access time (47ms).

(1) 1149.1 Access Time = $[(L_{sp} x S_{st} + D_{st}) x T_{ck}] x M_{pd}$ = 47 ms

Where: L_{sp} = Length of scan path
= (N+DBM+M) = (100+26+100)
S_{st} = TAP Shift state
D_{st} = TAP Dead states (4)
T_{ck} = TCK period (200ns)
M_{pd} = Memory pattern depth (1024)

(2) DMA Access Time = $(L_{dior} x M_{pd} x S_{st} + D_{st} + L_m) x T_{ck}$ = 3.3 ms

Where: L_{dior} = Length of DIOR register (16)
L_m = Length of M scan path (100)
S_{st} = TAP Shift state
D_{st} = TAP Dead states (4)
T_{ck} = TCK period (200ns)
M_{pd} = Memory pattern depth (1024)

Trace Memory Self-Test

The trace memory was designed with a self-testing capability that enables quick verification of the DBM's RAM in both an IC production and field test environment. The self-test is executed in two steps. The first step loads the RAM contents with test patterns, and the second step compacts the contents of the RAM into a signature.

The first step, RAM initialization, can be accomplished using one of two methods. One method uses RAM upload instructions designed to allow the data scanned into the DIOR to be automatically uploaded into the RAM for 1024 TCKs while the TAP is in its RT/IDLE state. Two RAM upload instructions are provided to allow data to be uploaded into the RAM in either a fixed or toggled format. This automatic RAM upload method is also used to initialize the RAM prior to performing a data trace operation. The other method of initializing the RAM uses the DMA input method. While the DMA input mode takes longer to load the RAM than the RAM upload instructions, it allows deterministic test patterns to be input to the RAM for more thorough testing.

The second step, RAM compaction, is accomplished using a RAM data compaction instruction. The data compaction instruction enables the RAM contents to be read and compacted into a 16-bit signature for 1024 TCKs while the TAP is in it's RT/IDLE state. During this instruction, the DIOR operates as a multiple input signature register (MISR) and the AREG operates as a self incrementing RAM address counter.

If desired, deterministic testing of the RAM can be achieved efficiently using the DMA input and DMA output instructions to stream data into and out of the RAM. The RAMs data retention can be tested by disabling the RAM to its low power mode after it has be initialized, then enabling the RAM and testing its contents using either the data compaction or DMA output instruction.

PSA Register

The PSAR receives the 16-bit data bus, serial input from the TDI pin, control input from the TAP and EQM, and outputs serial data to the TDO pin. Also the PSAR is coupled to a bidirectional pin serving as a polynomial input or output (PIO) to allow cascading the PSARs of multiple DBMs. The PSAR is used to sample or compact data input to the DBM via the data bus in response to control input from either the TAP or EQM. The selection of which control input is used, TAP or EQM, is determined by the instruction in the 1149.1 instruction register. The PSAR can receive and compact data words at up to 40 Mhz.

The PSAR in Figure 4 consists of a 16-bit MISR and data bit masking circuitry. The polynomial tap connections in

the MISR are programmable via input from the control register in the 1149.1 circuit block of Figure 1, to enable use of any polynomial feedback equation during data compaction operations[9]. Also the bit masking circuitry receives masking control input from the control register to selectively mask one or more of the data bus input signals from being compacted into the MISR. This bit masking feature allows the MISR to operate as a parallel, serial, or selectable input signature analyzer. Also the bit masking provides a method of diagnosing why a parallel signature fails, by repeating the test on individual or selected groups of non-masked inputs[9].

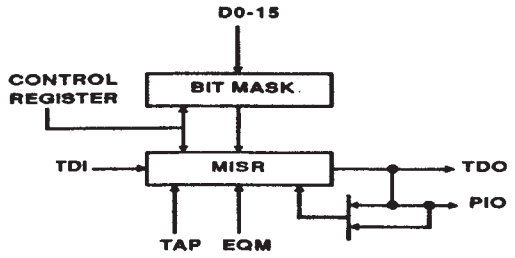

FIGURE 4 PSA Register

When PSAR operations are controlled from the 1149.1 TAP, data input on the data bus is compacted into the MISR during each TCK while the TAP is in its RT/IDLE state. When PSAR operations are controlled from the EQM, data input on the data bus is compacted into the MISR when the EQM is qualified, via input from either the internal CTERM or external EQI signal, to execute one of its eight selectable test protocols. After either data compaction operation is complete, the signature can be shifted out of the MISR by loading in an PSAR read instruction and performing a data register scan operation to shift out the contents of the MISR.

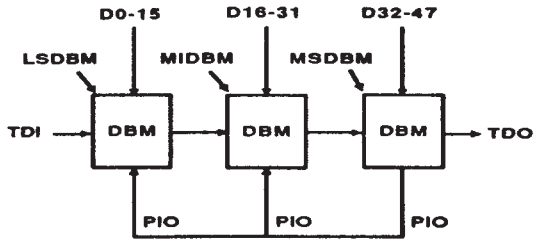

FIGURE 5 Cascading DBMs

The PSARs of multiple DBMs can be cascaded in 16 bit increments to form wider parallel signature analysis registers. For example, Figure 5 illustrates three DBMs cascaded to form a 48-bit PSAR. During cascaded PSAR operation, the least significant DBM's (LSDBM) TDI input is disabled, the TDO output is enabled to output the quotient from the MISR, and the PIO pin is configured as a polynomial input to the MISR. The middle DBM's (MIDBM) TDI input is enabled to receive the TDO output from the LSDBM, the TDO output is enable to output the quotient from the MISR, and the PIO pin is configured as a polynomial input to the MISR. The most significant DBM's (MSDBM) TDI is enabled to receive the TDO output from the MIDBM, the TDO output is disabled, and the PIO pin is configured as a quotient output from the MISR.

Since the MISR in each DBM is designed using an internal type polynomial feedback circuit, any number of MIDBMs can be placed between a LSDBM and MSDBM without incurring the delays associated with external type polynomial feedback circuits.

Comparator

The comparator receives the 16-bit data bus, serial input from the TDI pin, control input from the TAP and EQM, and outputs a compare term (CTERM) to the EQM and serial data to the TDO pin. The comparator is used during EQM controlled data trace and compaction test operations to detect patterns input to the DBM via the data bus and output a CTERM signal to the EQM in response to a detection.

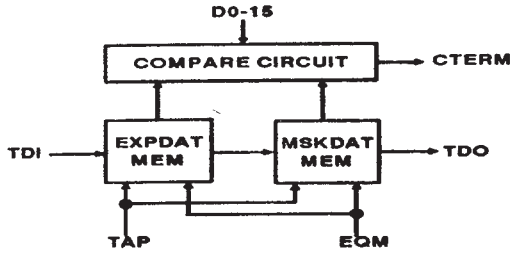

FIGURE 6 Comparator

The comparator in Figure 6 consists of a 16-bit compare circuit, and two 16x16 memories used to store expected data and mask data patterns. The memories are accessible for data input/output using either 1149.1 data register scan or DMA access modes described in the trace memory. The memories provide storage of 16 levels of 16-bit expected and mask data, enabling the comparator to perform up to 16 levels of pattern detection. The mask data memory provides selective bit masking on each of the 16 pattern detection operations. Masked bits become don't cares and are not used in the compare operation.

Programmable Clock Interface

The PCI receives clock inputs from the CK1, CK2, CK3, and TCK pins, control input from the control register, and outputs a conditioned clock signal to the EQM. The PCI is programmable via input from the control register to couple one of the clock inputs or its complement to the PCI CLK output, or combine the clock inputs using Boolean AND and OR functions and couple the combined clock signal or its complement to the PCI CLK output. The purpose of the PCI is to minimize clock gating and selection logic external to the DBM. For example, the PCI clock inputs can programmed to accept Motorola or Intel type read/write bus control without any external interfacing logic.

Event Qualification Module

The EQM is a test control macro that has been developed by Texas Instruments to provide a standard method of enabling an IC's test circuitry during normal operation of the IC[6]. The EQM receives control input from the TAP, clock input from the PCI, external event input from the EQI pin, internal event input from the CTERM signal, serial data input from the TDI pin, and outputs event signals to the EQO pin and serial data to the TDO pin.

The EQM of Figure 7 consists of a test controller, an event multiplexer, and a scan path consisting of command (CMD), loop counter (LPCNT), and event counter (EVCNT) sections. The test controller operates synchronous to the CLK output from the PCI to execute one of eight predefined test protocols. The protocol to be executed is input to the test controller via the CMD register. The LPCNT register serves as a programmable 16-bit counter enabling a protocol to be looped up to 2^{16} times. Each protocol is executed in response to event inputs from the event multiplexer. The event multiplexer can be set via the CMD register to select the event input to come from either the internal CTERM signal from the comparator or from an external signal input via the EQI pin.

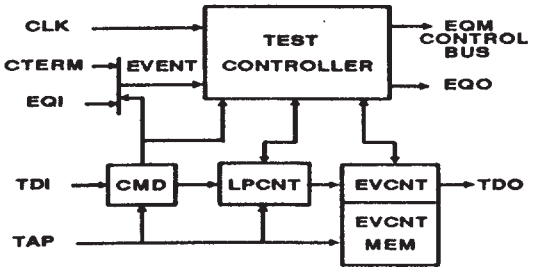

FIGURE 7 Event Qualification Module

The EVCNT register serves as a programmable 16-bit counter enabling a protocol to be started or stopped on the Nth occurrence of predetermined events. Also the EVCNT register can be used within a protocol to enable testing for a programmed number of CLK cycles, from N = 1 to 2^{16}. To allow for multiple Nth events and multiple clocked test operations within a test protocol, a 16x16 EVCNT memory is coupled to the EVCNT register. The EVCNT memory can be accessed for input and output using either 1149.1 scan operations or DMA operations as described in the trace memory section. When an existing count has expired in the EVCNT register, the test controller uploads a new count pattern into the EVCNT register. Each protocol can use up to 16 different count patterns during the execution of a test operation.

During EQM controlled data trace and/or data compaction operations, the EQM outputs control to the trace memory and PSAR via the EQM control bus. Also the test controller outputs test status or internal event signals off chip via the EQO pin. In the status output mode, test status signals can be output from the EQO pin to indicate, among other things, (1) test-in-progress or (2) end-of-test. These status outputs can be monitored by external controllers or testers to determine the state of the test being performed. Alternately, these test status signals can be inspected via 1149.1 instruction register scan operations.

In the event output mode, the CTERM signal from the comparator can be output from the EQO pin to indicate the occurrence of an internal event. The event signal output from the EQO pin can be used for triggering an external tester, or to input an event to a neighboring ICs EQM to initiate a test protocol operation. The action of outputting an internally detected event on the EQO pin of one IC to be received by the EQI input of another IC enables EQMs to cross trigger one another. This cross triggering scheme can be expanded to where multiple ICs can participate via an external voting circuit to generate a global cross triggering mechanism between EQO outputs and EQI inputs.

EQM Test Protocols

The eight EQM test protocols are summarized in the following protocol test action code statements. In each protocol, "M" references a count value from 1 to 2^{16} in the 16-bit LPCNT, "N" references 1 of 16 stored count values (1 to 2^{16}) that can be loaded into the 16-bit EVCNT, "test" refers to a data trace and/or data compaction operation, "event" refers to a signal input to the EQM from either the CTERM output of the comparator or the EQI input pin, and "clock" refers to the CLK output from the PCI circuit.

Protocol 1: For M times do;
On N^{th} event do test
End of test

Protocol 2: For M times do;
During N^{th} event do test
End of test

Protocol 3: For M times do;
On N^{th} event start test
On N^{th} event stop test
End Of Test

Protocol 4: For M times do;
On N^{th} event start test
On N^{th} event stop test
after N clocks
End of test

Protocol 5: For M times do;
On N^{th} event, test for
N clocks
End of test

Protocol 6: For M times do;
On N^{th} event,
Pause N clocks
Then test for N clocks
End of test

Protocol 7: On N^{th} event, test for N clocks
Then for M-1 times do;
Pause for N clocks
Test for N clocks
End of test

Protocol 8: On N^{th} event, For M times do;
Pause for N clocks
Test for N clocks
End of test

DBM TEST APPLICATIONS

As described previously, the DBM can operate in two test modes, an off-line test mode controlled by the TAP, and an on-line test mode controlled by the EQM. In the off-line test mode the circuitry to be monitored needs to be placed in a nonfunctional mode and preferably under 1149.1 control. In the on-line test mode the circuitry to be monitored may remain in its normal functional mode while being tested by the DBM.

In Figure 8, a typical circuit is shown consisting of a processor, RAM, ROM, I/O port, buffering/transceiving

circuits, and two DBMs. This circuit could be realized on either a printed circuit board or multi-chip module substrate, and may be one of many such circuits in an electronic system. During functional operation, the processor outputs address and control to enable data transfers between it and one of the other devices in the circuit. DBMs are coupled to the processors address, data, and control buses.

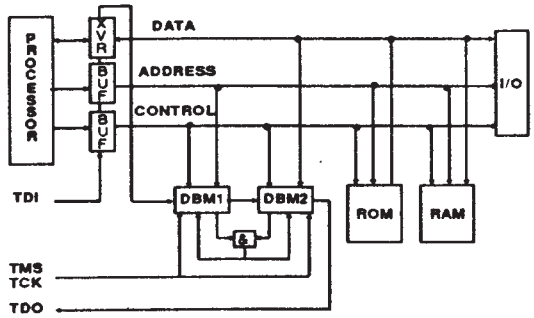

FIGURE 8 DBM Application Example

While non-testing buffers/transceivers could have been used in this circuit, 1149.1 compliant ones are implemented to provide embedded control and observability of the processors 16-bit data, 16-bit address, and control buses. The 8-bit buffers and transceivers used, like the DBMs, are members of TIs SCOPE family of testability components and are referred to as SCOPE octals[7].

In addition to their buffering and transceiving functions, the octals include circuitry supporting both 1149.1 boundary scan and built-out-self-test features. The term built-out-self-test (BOST) is probably one the reader may not be familiar with and is coined here to differentiate from the common terminology of built-in-self-test (BIST). BIST is a pseudorandom based test structure designed primarily to test circuits inside an IC, whereas BOST is a pseudorandom based test structure designed primarily to test circuits outside an IC.

While in their functional mode, the octals allow the data, address, and control bus signals to pass through unobstructed. However, when placed in their test mode, the octals can execute all required 1149.1 scan based test operations as well as additional pseudorandom based BOST features[8,10,11]. One of the BOST features included in the octals that is pertinent to the DBM's off-line test mode description is the ability to generate and output pseudorandom test patterns in response to control input from the 1149.1 test bus.

DBM Off-Line Test Examples

The primary objective of an off-line test of the circuit in Figure 8 is to verify the structural integrity of the interconnections between the ICs. Most manufacturing failures are those related to shorts and opens in either the wiring interconnects between ICs or the solder connection between ICs and their associated substrate footprints. A secondary objective of the off-line test is to verify the basic functionality of each component in the circuit.

1 – Off-Line Shorts Testing

To achieve the shorts test part of the primary objective, the octals as well as the DBMs are scanned into their 1149.1 external test (Extest) mode. In the Extest mode the devices are operable to output test patterns from their outputs and receive test patterns into their inputs under control input from a test bus controller on the 1149.1 test bus. During the shorts test, the octals are scanned to output a sequence of test patterns, such as walking ones, on the data, address, and control buses. Concurrently, the DBMs are scanned to receive the test patterns from the octals and output them to the test bus controller for inspection.

Alternately, a shorts test on the data and address buses could be accomplished by enabling the octals to generate and output pseudorandom patterns while the DBMs receive and store the patterns into their trace memory and/or compact the patterns into their PSAR. After the test is complete, the test bus controller accesses the stored test pattern response and/or signature from the DBMs for processing. This shorts test may provide a reduced test time over the scan based shorts test, since the test patterns are generated by the octals, not scanned into the octals.

2 – Off-Line Opens Testing

To achieve the opens test part of the primary objective, the octals are scanned into their 1149.1 Extest mode and operated under control input from the test bus controller to emulate processor read and write operations to the RAM, ROM, and I/O port. Good connectivity between the octals and the RAM and I/O port is verified by successful read and write operations. Good connectivity between the octals and the ROM is verified by successful read operations.

3 – Off-Line Component Testing

To achieve the secondary objective of the off-line test, the functionality of the RAM, ROM, and I/O port must be verified. The I/O port can be easily verified by scanning the octals into their Extest mode and inputting and outputting a sequence of data to the I/O port via multiple 1149.1 scan cycles.

The RAM can be tested by first loading it with test patterns, then reading back the test patterns to verify they were input and output correctly. While there are ways to automate the loading of the RAM using the BOST features of the octals, the loading here is performed by scanning the octals into their Extest mode, then scanning them multiple times to provide the address, data and control required to load the RAM.

After the RAM is loaded, the octals are scanned to setup the RAM read part of the test. During RAM reads, the output of the data octals are disabled, the control octal is set to Extest mode and outputs control for read operations, the higher order byte addressing octal is set to Extest mode and addresses a 256 location address range within the RAMs memory space, and the lower order byte addressing octal is set to output pseudorandom patterns within the specified 256 location RAM memory space during the TAP RT/IDLE state. Also DBM2 is scanned to enable its trace memory to operate during the TAP RT/IDLE state. During the RT/IDLE state, the low order address octal outputs

addresses on the falling edge of the TCK and DBM2 stores RAM output data on the rising edge of the TCK.

After 254 TCKs the test bus controller moves the octals and DBM2 out of the RT/IDLE state, scans the higher order address octal to point to the next 256 location address range within the RAM memory space, then moves the devices back into the RT/IDLE state for 254 more TCKs. The 255th TCK of each address sequence is produced upon moving out of the RT/IDLE state. This process is repeated four times to fill the trace memory of DBM2.

After the trace memory fills, the test bus controller inputs a DMA output instruction into the DBM and streams out the contents of the trace memory. The test bus controller must read the first (>00) memory location of each 256 location address range using the Extest mode to complete the test. This additional step is required since pseudorandom generators cannot produce an all zero output. This entire process is repeated until all the RAM data has been read.

Testing of the ROM is simpler than the RAM since no preloading of test data is required. During ROM testing the octals and DBM2 are setup and operated as described in the RAM read test description. The only difference is that the ROM will be located in a different memory map space than the RAM.

If a faster ROM or RAM read access is desired, the data read from either could be compacted into a 16-bit signature using DBM2's PSAR. After the data is compacted, the signature is accessed by the test bus controller and compared to an expected value. The reduction in test time is brought about by the ability of the PSAR to compact the entire contents of the ROM or RAM before outputting the signature to the test bus controller.

Additional SCOPE buffering/transceiving devices are being developed. These devices will be in 8, 16, and 18-bit configurations, and will provide both pseudorandom and binary counting output modes. The binary count output mode will eliminate the missing read operations to "all zero" address locations associated with the pseudorandom output mode. Also, if 16-bit devices were used in this example, the RAM/ROM read operations would be much more efficient since they would not have to be interrupted every 256 locations to scan in the next higher 256 location address range.

DBM On-Line Test Examples

The objective of an on-line test of the circuit in Figure 8 is to verify that the circuit as well as the controlling software functions properly at its rated speed. If the circuit passes an off-line structural and component test, there is a good chance it will operate at-speed. However, subtle problems associated with the interaction of the ICs in response to software control cannot be adequately tested in an off-line test environment.

To provide an embedded on-line testing method, the DBMs can be controlled via their internal EQMs to monitor the circuits functional operation. During on-line testing the octals remain in their functional mode and do not participate in the test. The 16-bit data bus input of DBM1 is connected to the processors address bus to allow triggering on and tracing of address patterns. The 16-bit data bus input of DBM2 is connected to the processors data bus to allow triggering on and tracing of data patterns. The CK 1, 2 and 3 inputs of both DBMs are connected to the processors control bus to allow test operations to be enabled and synchronized with read, write, or read/write bus transactions. The TAP pins of the DBMs are connected to the 1149.1 test bus to allow an external test bus controller to input test instructions, monitor test status, and receive test results.

To expand the DBMs event qualification capability, the EQO outputs from both DBMs are input to an And gate, and the EQI inputs to both DBMs receive the output from the And gate. This is a standard configuration for ICs equipped with EQMs and allows the qualification of a test operation to be determined by events detected in DBM1, DBM2 or both DBM 1 and 2.

Since the DBMs have eight protocols from which a monitoring operation may be controlled, the first step is to determine the type of test to be performed, then select a protocol best suited for the test. For example, if the test is to trace data bus transactions occurring between two particular addresses, protocol 3 would suffice. Alternately, if the test is to sample a single data pattern in response to the occurrence of a particular data and/or address pattern, then protocol 1 one may be used. Moreover, if the test is to compact data appearing on the data bus for a predetermined number of bus cycles in response to the occurrence of a particular data and/or address pattern, then protocol 5 may be used. The range of defined protocols provides the user with a very flexible triggering environment through which a test operation may be effectuated.

After choosing the appropriate test protocol, the control and data required by the protocol is input to the DBMs via the 1149.1 test bus. Following this protocol setup procedure, an instruction is input to the DBMs, via the 1149.1 test bus, to initiate the on-line test operation. Once the DBMs receive the on-line test instruction, their operation is autonomous from the 1149.1 test bus. During test, the test bus controller can monitor the state of the DBMs by accessing test status information via 1149.1 instruction register scan operations.

When the test bus controller receives an end-of-test status output from the DBMs, it performs an instruction register scan operation to input an instruction to access the test results. If a data trace operation was performed, an instruction enabling a read of the trace memory is input. If a data compaction operation was performed, an instruction enabling a read of the signature is input. After the test bus controller receives the test result data, the data may be processed as required.

In some applications, the test result data may need only to indicate a pass or fail condition using the signature analysis test mode. In other applications, the test data result may be used to help diagnose hardware/software integration problems using the data trace mode. To facilitate human interpretation of the test result data, the test bus controller may output trace data to a video monitor for display in a standard logic analyzer waveform format.

To simplify the use of the DBM's on-line test features, software support is being developed in TIs ASSET™ diagnostic system[12,13]. The software support will

include a windowing environment through which on-line test operations can be easily setup and executed, and the results displayed in a waveform format. From a users perspective the interface to DBMs will be similar to that used by traditional logic analyzer testing instruments.

The following example applications provide insight into some of the types of on-line test operations the DBM can perform on the example circuit of Figure 8.

1 – On-Line Testing During An Event

As part of the normal operation of the circuit, the software program controlling the action of the processor periodically addresses the I/O port to enable multiple transfers of data into and out of the circuit. The DBMs can non-intrusively trace and/or compact the circuits data I/O transfers by using DBM1 to provide the triggering function and DBM2 to provide the data trace and/or compaction function. In this example a trace operation is selected.

To setup any EQM controlled test operation, three basic decisions need to be made, (1) protocol selection, (2) protocol control input selection, and (3) number of times (M) to repeat the protocol. Since the I/O port uses only one address in the processors memory map space, Protocol 2 best fits the protocol selection for this test application (decision 1). Since the PCI circuit of the DBMs can be programmed to respond to read, write, or read/write control input from the processor, the data to be traced can be defined as input only, output only, or both input and output. To test both directions of data transfer through the I/O port, the PCI is selected to enable tracing of both input and output data (decision 2). Assuming each I/O operation transfers 100 data words and it is desired to fill the trace memory, but not overflow it, the trace operation is set to repeat 10 times (M=10), to allow 1000 I/O transfers to be traced (decision 3).

With these setup decisions made, the 1149.1 test bus controller inputs the required data and control to the DBMs to execute the I/O trace test. After starting the test, the test bus controller monitors the test status via instruction scan operations. During the test DBM1 continuously monitors the address bus for the I/O port address. When DBM1 detects the address, it outputs an EQO signal to the And gate, which in turn inputs the signal to the EQI input of DBM2. While the EQI input signal is present, DBM2 stores the I/O data transfers into its trace memory. When DBM1 detects a different address, it stops outputting the EQO signal which terminates the first I/O trace operation of DBM2.

The second through tenth repetitions of the I/O trace operation are performed exactly as the first. At the end of the tenth I/O trace operation, the end-of-test status bit is set in DBM2. When the test bus controller detects the end-of-test status, it accesses the trace data stored in DBM2 using either the 1149.1 scan or DMA output modes and either processes the data directly or displays it in a desired format for human interpretation.

2 – On-Line Testing Between Two Events

During program execution the processor reads and writes data and instruction words to the RAM. If desired, the DBMs can be used to trace and/or compact selectable R/W transfers between processor and RAM memory. In this example the transfers are traced. The RAM R/W trace operation differs from the I/O trace operation in that RAM is accessed via a plurality of sequential addresses within the processors memory map. Since the RAM has a different addressing scheme than the I/O port, a different protocol is required for the trace operation.

During RAM R/W tracing the DBMs need to window the trace operation between a first address that starts the RAM R/W trace operation and a second address that stops the RAM R/W trace operation. Protocols 3 and 4 support test operations bounded by starting and stopping events or in this case addresses. Protocol 4 enables additional trace operations to continue for a specified number of bus cycles after the stop event, whereas Protocol 3 stops the trace operation in response to the stop event.

Since only the data within the window is to be traced, Protocol 3 is selected for the R/W trace operation. Also, during the RAM R/W trace operation, the address associated with each data word is traced, to simplify interpretation of the data by the user. Since Protocol 3 enables the trace operation in DBM 1 and 2 to be repeated a specified number of times (M) with up to 8 pairs of different start and stop addresses (events), it is possible to perform multiple RAM R/W traces within the protocol.

For the RAM R/W trace operation both DBMs are setup to; (1) perform Protocol 3 operations, (2) trace both address and data for both read and write transfers, and (3) repeat traces for 5 times. With these setup decisions made, the 1149.1 test bus controller inputs the required data and control to the DBMs to execute the RAM R/W trace test. After starting the test, the test bus controller monitors the test status via instruction scan operations.

During the test DBM1 continuously monitors the address bus for the first trace start address. When DBM1 detects the address, it outputs an EQO signal to the EQI input of DBM2. In response to the first trace start address, DBM1 starts tracing addresses and monitoring for the first trace stop address, and DBM2 starts tracing data. When DBM1 detects the first trace stop address it signals DBM2 via the EQO to EQI signal path. In response to the first trace stop address, both DBMs stop their trace operations and DBM1 starts monitoring the address bus for the second trace start address.

The second through fifth trace operations are performed exactly as the first. At the end of the fifth trace operation, the end-of-test status bit is set in DBMs 1 and 2. When the test bus controller detects the end-of-test status, it accesses the trace data stored in DBMs 1 and 2 using either the 1149.1 scan or DMA output modes and either processes the data directly or displays it in a desired format for human interpretation. Since both address and data were traced, the display can associate each data word transfer with a particular address.

3 – On-Line Testing In Response To An Event

In some monitoring applications it may be desired to trace and/or compact data words transferred between the processor and I/O, RAM, and ROM in response to a single predetermined address and/or data pattern. To enable this type of test operation, Protocols 1, 5, 6, and 7

should be considered. Protocol 1 enables a single data trace and/or compaction operation in response to an event. Protocol 5 enables data trace and/or compaction for a specified number of bus cycles in response to an event. Protocol 6 enables the steps of (1) pausing for a specified number of bus cycles and (2) performing a data trace and/or compaction operation for a specified number of bus cycles. Protocol 7 enables the steps of (1) performing a data trace and/or compaction operation for a specified number of bus cycles and (2) pausing for a specified number of bus cycles. Each protocol can be repeated M times.

One application of this type of test protocol would be to use Protocol 1 to take a single snapshot sample of the data bus in response to a particular address pattern. In this application DBM1 would detect a predetermined address and output an event signal to DBM2. In response to the event input, DBM2 would execute a Protocol 1 operation to sample and store the pattern on the data bus. After the test, the test bus controller accesses the sampled pattern from DBM2 for displaying. While the other protocols of this type offer expanded test control features, many hardware and software debug tasks can be resolved by viewing a single pattern in response to a particular circuit state or condition.

Another application of this type of test protocol would be to use Protocol 5 in combination with a self test program residing in the ROM to compact the contents of the ROM into a signature. The self test program would be designed to cause the processor to sequentially read the contents of the ROM from the ROM's starting address to ending address. Prior to invoking the self test, DBM1 would be setup to detect the occurrence of a read operation to the ROM's starting address and to output an event signal to DBM2. Also, DBM2 would be setup to perform a data compaction operation on each ROM read cycle in response to an event input signal from DBM1 using Protocol 5. The N variable in Protocol 5 would be set to equal the address range of the ROM. This testing scenario assumes the processor can upload the self-test code from ROM prior to starting the ROM read operations so that the self-test code executes internal to the processor, not external to the processor. After the self test completes, the data compacted in DBM2 can be accessed by the test bus controller to compare the ROM signature against a known good signature.

Still another application of this type of test protocol would be to use Protocol 6 to respond to an event and perform a data trace and/or compaction operation after a predetermined delay. For example, if the software program transfers into an area of code that is to be tested, and it is not desired to start the test when the transfer first occurs, a delay can be setup in Protocol 6 to suspend testing for a selected number of bus cycles (N). After the delay, the data can be traced and/or compacted for a selected number of bus cycles (N). With the ability of Protocol 6 to repeat the steps of pausing and testing M times, the data trace and/or compaction operation can be accurately mapped into only those areas of code to be tested, skipping over areas of code not to be tested.

Protocol 7 is the inverse of Protocol 6 in that it allows testing to begin immediately in response to an event, then paused for a specified number of bus cycles before once again testing for a number of bus cycles. The inverse test scenario of that described for Protocol 6 can be envisioned for testing using Protocol 7.

4 - On-Line Failure Monitoring

While the previous examples have been associated with test operations setup and executed by a test bus controller for the purpose of testing the at-speed operation of the circuit, the DBMs can be used in a different way to achieve still another embedded monitoring capability. To set this scenario up assume that the data inputs from the I/O port indicates some external condition that the circuit is monitoring. Normally the inputs can be monitored and processed without problems. However, if an unexpected input is received that the program is unable to process, it transfers into an area of code designed to recover from such failures and to output status via the I/O port that it has encountered the failure. When attempting to diagnose the problem it would be beneficial to have a history of the circuits operation before and after the occurrence of the failure condition that caused the problem.

To provide a means of monitoring for failure conditions and for creating a history of the circuits operation before and after the failure, the DBMs can be setup using Protocol 4 to continuously trace the data and address buses while the circuit is operating. Protocol 4 is a windowing type protocol that allows the trace operation to be started in response to a first predetermined event, and stopped N clocks after the occurrence of a second predetermined event. When using Protocol 4 as control for failure monitoring, the trace functions of both DBMs are started in response to some arbitrary address/data pattern that occurs normally during the operation of the circuit. However, the second address/data pattern that initiates the termination of the trace operation is specified to be the particular address/data pattern that transfers code execution into the failure recovery and status output routine.

With the DBMs setup in this mode, they continuously input and store address and data patterns into their 1Kx16 trace memory during each bus cycle. During this test scenario, the trace memory is enabled to overflow or wraparound as it fills with data. If no failure is encountered, Protocol 4 never terminates. However, if a failure occurs the DBMs will stop storing address and data patterns into the trace memory after a specified number of bus cycles (N). The specified number of bus cycles to continue tracing after detection of a failure is defined by the user and allows partitioning the trace memory of DBMs 1 and 2 into two sections, one for pre-failure circuit history and the other for post-failure circuit history.

For example, if Protocol 4 is set to terminate the trace operation 512 bus cycles after detecting a failure, half the DBMs trace memory (512 locations) will contain information about the circuits operation before the failure and half will contain information about the circuits operation after the failure. By increasing the number of bus cycles to execute following the failure, the trace memory will contain more post-failure information. Likewise, by decreasing the number of bus cycles to execute following the failure, the trace memory will contain more pre-failure information.

A test bus controller monitoring the DBMs status via instruction register scans can determine if the Protocol 4 operation terminates. If a termination is found, the test bus controller can access the address and data patterns

stored in trace memories of DBM1 and DBM2 and provide this circuit history information to aid in diagnosing the cause of the failure.

On-Line Testing of System Sub-Circuits

In the examples described, the DBM's on-line test modes have been shown to provide an improved method of testing the at-speed operation of the circuit in Figure 8. If the test scenario is expanded to encompass a complete system comprising multiple sub-circuits similar to the one in Figure 8, the benefit of the DBM's on-line testing features becomes even more significant.

While it is important to verify the functional operation of each sub-circuit in a system, it is even more important to verify the complete functional operation of the entire system. If DBMs were included in each sub-circuit of a system, their testing capabilities could be used to functionally test each sub-circuit, then harnessed together to provide a method of concurrently testing the functional operation of the entire system.

Such a capability would be equivalent to having logic analyzers embedded and coupled to all critical signals and buses in each buried sub-circuit within a system. System level on-line monitoring could be achieved by setting up the DBMs in each sub-circuit to perform an at-speed test operation in response to a predetermined or expected system level transaction.

For example, if the system sub-circuits are similar to the one in Figure 8, and they communicate together at the system level via their I/O port, the DBMs of each sub-circuit could be setup to monitor the I/O port for an expected input or output transaction. In response to the expected transaction, the DBMs of each sub-circuit could simultaneously execute a data trace on the local operation of each sub-circuit. After this system level test is completed, a test bus controller could access the traced data from each sub-circuit and process the data to determine how each sub-circuit responded during the predetermined system level transaction.

The power of such a test approach comes from the fact that the functional operation of a plurality of sub-circuits can be monitored concurrently and non-intrusively during normal system operation. Thus the functional interaction between multiple embedded sub-circuits in a system can be tested to verify that each sub-circuit as well as the entire system operates properly.

SUMMARY

While the off-line and on-line test features described in this paper require adding DBMs to boards/multi-chip module substrates, the benefit gained may, in some applications, offset the additional package/die penalty incurred. Without including the DBMs in the example circuit, the type of test operations described would have to be performed using external test instruments and probing mechanisms. If the circuit is embedded within a system, access for probing may be a difficult and time consuming proposition, if possible at all.

The decision whether to use DBMs in a product is influenced by many factors such as; product type, manufacturing costs, test requirements, and life cycle costs. In any case, the DBM offers a test option currently not available to the industry, and provides a novel method of testing state-of-the-art technologies that are resistant to traditional test approaches.

CONCLUSION

This paper has described the architecture and operation of an 1149.1 based digital bus monitor (DBM) IC. The ability of the DBM to monitor the at-speed signal transfers between ICs in real time provides a method of monitoring the functional operation of circuits assembled on board and multi-chip module substrates. Such tests can be used to reveal timing sensitive and/or intermittent failures that would otherwise not be detectable without the use of external testers and mechanical probing fixtures. This test approach may, in some cases, reduce the cost to manufacture and support a product by reducing the need for test equipment in both factory and field environments.

REFERENCES

1. IEEE Std 1149.1 1990, A Standard Test Access Port and Boundary Scan Architecture

2. C. Maunder, R. Tulloss, "The Test Access Port and Boundary Scan Architecture," IEEE Computer Society Press 1990.

3. P. Hansen, "The Impact of Boundary Scan on Board Test Strategies," Proc. ATE & Instrumentation Conference West, 1989, pp 35-40.

4 K. Parker, "Production Board Testing in a Boundary Scan Environment," Proc. ATE & Instrumentation Conference West, 1990, pp 135-141.

5. M. Lefebvre, "Functional Test & Diagnosis: A Proposed JTAG Sample Mode Scan Tester," Proc. IEEE International Test Conference, 1990, pp 294-303.

6. L. Whetsel, "Event Qualification – A Gateway To At-Speed System Testing," Proc. IEEE International Test Conference, 1990, pp 135-141.

7. Texas Instruments, "Advanced Logic and Bus Interface Logic, Data Book," 1990, sec. 11, pp. 1-264, Customer Services.

8 L. Whetsel, "JTAG Devices Simplify Board Level Design for Testability," Proc. IEEE Wescon, 1989, pp 294-299

9. L. Whetsel, "Modular ASIC Test Cells for Boundary Test Applications," Proc. Government Microcircuit Applications Conference, 1989, pp 497-500.

10. A. Halliday, G. Young, A Crouch, "Prototype Testing Simplified by Scannable Buffers and Latches," Proc. IEEE International Test Conference, 1989, pp.174-181

11. L. Whetsel, G. Young, "Hardware Based Extensions to the JTAG Architecture," Proc. ATE & Instrumentation Conference West, 1990, pp. 1-10.

12. Texas Instruments, "Test And Emulation Primer," 1989, Customer Services.

13. Texas Instruments, "ASSET Technical Overview Data Sheet," Customer Services.

TRADEMARKS