Design For Testability Features of the SUN Microsystems Niagara2 CMP/CMT SPARC Chip

Robert Molyneaux, Tom Ziaja, Hong Kim,
Shahryar Aryani, Sungbae Hwang, Alex Hsieh

SUN Microsystems
5300 Riata Park Court
Austin, Tx. 78727

SUN Microsystems
410 N. Mary Ave
Sunnyvale, Ca. 94085

Abstract

The Niagara2 System-on-Chip is SUN Microsystem's latest processor in the Eco-sensitive CoolThreads line of multi-threaded servers. This DFT survey of the Niagara2 chip introduces the RAWWCas memory test, a Hybrid Flop Design and a fast efficient bitmapping architecture called DMO. It also showcases some excellent DFT results for this challenging system-on-chip design project.

1.0 Introduction

The Niagara2 SPARC chip is truly a Chip Multiprocessor, Core Multithreaded, system on a chip. It has eight processor cores each supporting eight threads. The cores run at 1.4GHz and are connected to 4MB of on-chip L2 cache. The L2 cache connects to 4 on-chip DRAM controllers which directly interface to a pair of fully-buffered DIMM channels. Additionally N2 has an on-chip Network Interface Unit with two 1 Gb/10 Gb ethernet MACs, and an on-chip PCI-EX controller.

The system on chip nature of Niagara2 poses a number of DFT challenges. There are eight different clock domains within the chip and a mixture of full custom, semi-custom, and ASIC design styles. SERDES blocks [3] are hard IP, acquired complete with their own DFT architecture which must be integrated into the overall Niagara2 DFT architecture. The entire functional IO space of the chip is high speed SERDES rendering testing with functional vectors difficult and limited. This makes providing high quality stuck-at and transition test vectors imperative.There are more than 300 SRAMs of various sizes and functionalities each of which has an MBIST test plan. There are over one million flops in the design managed in over forty different scan string configurations and three different clocking scenarios. Welcome to 65nm SOC design!

2.0 DFT Overview

Niagara2 has a rich set of test and debug features. It is designed with a robust level sensitive scan architecture and a variety of scan configurations to support many different test and debug activities. The stuck-at test coverage requirement is 99% and the transition test requirement is 95%. Every SRAM and CAM on chip is tested with at-speed MBIST. The MBIST is rich with diagnostic modes to aid failure analysis, backside

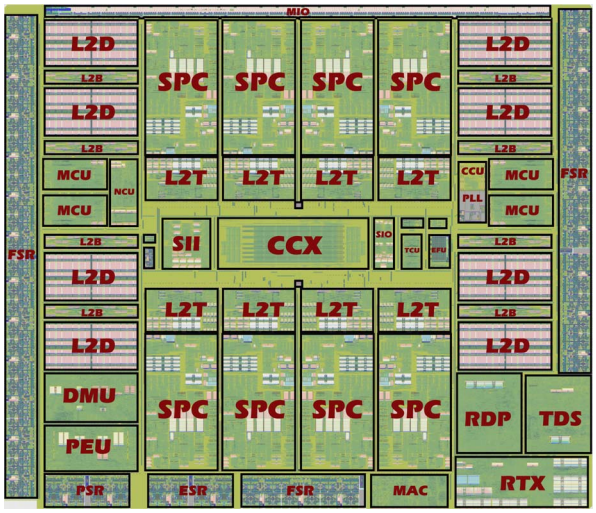


Figure 1.0.1: Niagara 2 Floorplan

probing, and failure isolation. The SPARC cores have LBIST and there are shadow scan registers scattered throughout the chip that capture state and can then be accessed via JTAG while the chip is running in system. There is support for at-speed transition test and a novel mechanism for fast efficient bitmapping of embedded SRAMs.

3.0 Scan Chains

The more than million flops on Niagara2 are organized into 32 manufacturing scan chains. Every flop on the chip resides on one of these chains. There is a second slightly different manufacturing scan chain configuration to accommodate at-speed transition testing. In addition to the two manufacturing scan chain configurations there are eighty four JTAG scan chain configurations. These are intended for in-system use during debug and characterization activities. Each of the 32 manufacturing scan chains can be placed between TDI and TDO for in-system JTAG scan access. The manufacturing scan chain that contains the JTAG block in the Test Control Unit (TCU) is understandably altered for JTAG access. Additionally all chains can be concatenated and placed between TDI and TDO. There are thirty five MBIST scan chains. These short MBIST chains are intended for rapid programming of MBIST configuration registers for use during SRAM diagnostic activities. These MBIST chains also provide access to

1-4244-1128-9/07/$25.00

detailed failure information upon completion of an MBIST run. Shadow registers scattered throughout the chip have sixteen JTAG scan chains dedicated to accessing them for in-situ peeks at the internal state of a running system.

4.0 Hybrid flop design

The test clocking methodology for Niagara2 is LSSD. The basic edge triggered flip-flop is favored by designers due to its single high speed clock operation but level sensitive scanning is favored for test due to its high reliability and immunity to hold time failures.

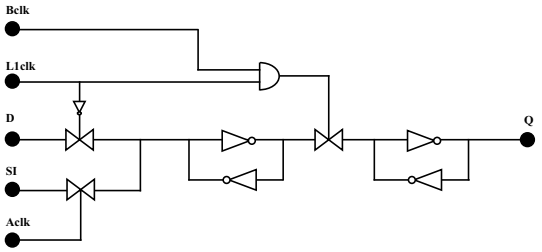


Figure 4.0.1: Hybrid Design: Level Sensitive Scannable Flip-Flop

The Niagara2 chip employs a hybrid flop design that operates as an edge triggered flip-flop for functional clocking but uses two lower speed level sensitive test clocks for scanning. The hybrid flop also supports level sensitive operation in the event a functional hold time violation is suspected of causing circuit failure. During debug activities the chip can be clocked as shown in table 4.0.2

	Time →				
	Clocks Off	Load Master	Clocks Off	Load Slave	Clocks Off
L1clk	1	0	1	1	1
Aclk	0	0	0	0	0
Bclk	0	0	0	1	0

Table 4.0.2: Level Sensitive Clocking for Debug

5.0 Stuck-At Test Coverage

The Niagara2 chip achieved 97.5% stuck-at test coverage. Further processing of the fault list to account for logic tested by MBIST brings the test coverage to nearly 98.5%.

A few simple testability guidelines were implemented at the beginning of the project and the results are evident in the good test coverage results on this large chip. Library elements are required to be 100% testable. SRAMs and pass-gate muxs also receive special attention.

Shadow logic surrounding embedded SRAMs is often a testability problem in microprocessors. Early in the design phase DFT, Logic design, and SRAM circuit designers got together to address the shadow logic issue. It turned out that the circuit designers preferred to bound the inputs and outputs of SRAMs with flops in order to better characterize their circuits. On the other hand the logic designers preferred to have the SRAMs completely unbounded so they could take advantage of unused cycle slack. The compromise decision was made to design all SRAMs with input flops thereby bounding the inputs for ease of characterization. The outputs would remain unbounded so the logic designers could insert logic to use any left over cycle time after an SRAM access. This played very well into the DFT cause. Shadow logic is nonexistent in the design upstream from any SRAMs. There is only an issue of shadow logic downstream from some SRAMs. Some SRAMs are simple and can easily be modeled as a Fastscan CRAM. Others are equipped with a data bypass function and still others drive known values when not performing a read. These behaviors can be modeled to provide a mechanism to test most if not all the downstream shadow logic.

Pass-gate muxs are a recurring testability issue. Early in the design phase these circuits were not available in the library. Only after logic design was finished and a round of timing had been completed were pass-gate muxs made available in the library to solve timing and space issues. All pass-gate muxs are equipped with a weak pull down transistor on the output node to provide testability for the select logic. Additionally the selects are equipped with priority encoding logic that resolves multiple active selects before they reach the mux select inputs. This relieves the ATPG tool from having to determine a safe state for all the tri-state nodes for every test vector. This is a tremendous performance boost for Fastscan. The priority encoder on the select inputs also provides for the all-off state guaranteeing s-a-1 testability of all the selects.

6.0 Transition and Path Delay Testing

Transition test coverage on Niagara 2 is approximately 82%. The many different clock domains added to the difficulty of creating transition test vectors as well as inhibiting high coverage. To avaoid multi-domain race conditions transition vectors used only one clock per vector. Because of this, clock domain crossings were not tested with transition test vectors.

In addition to transition testing Niagara 2 implements path delay testing for the cores. More than 15,000 paths in each core are tested at speed with Fastscan derived path delay test vectors. Because the chip IO is exclusively high speed SERDES the ability to perform functional testing is limited. This heightens the importance of high quality at-speed transition and path delay testing.

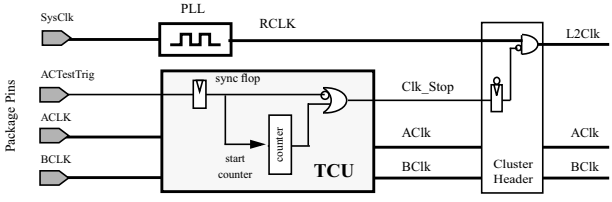


Figure 6.0.1: Transition Test Control

Transition testing and path delay testing utilize the same clock control mechanism. The fundamental design of the transition test controller is similar to that described in Pyron and Molyneaux, et al. [2] The two key factors are to keep the on-chip PLL running throughout the entire testing process and to decouple the relatively slow tester from the high speed signals needed to test the on-chip logic. In order to accomplish the first goal it is necessary to establish a different scan configuration than that used for DC stuck-at testing. It is necessary to remove from the scan chains any flops needed to keep the PLL locked and running. An ACTest Pin is added to the package in order to signal the TCU that transition testing is taking place. In order to accomplish the second goal a trigger and control circuit is designed to allow the ATE to trigger the test in an asynchronous manner. This scheme also allows the ATE to perform scanning with the same ACLK, BCLK, and ScanEn signals as it uses in DC stuck-at testing at any convenient speed.

Figure 6.0.1 shows a simplified diagram of the transition test control. Throughout the testing process the PLL is locked and running and the RClk is toggling at 1.4GHz. The transition test control logic controls the clock stop signals in the chip. The initial state of clock stop is asserted; this stops the L2Clk. The ATE asserts ScanEn and toggles ACLK and BCLK thereby loading a test vector into the chip. Along with the test vector the test control counter is also loaded with the desired count. When scanning is complete ScanEn is deasserted and ACTestTrig is asserted. This causes the stop signal to be deasserted and the test control counter to begin decrementing. Note that the clock to the test control counter is always running so it is not affected by its own action of starting and stopping the clock. When the counter reaches zero it reasserts the stop signal and the test is complete. The ATE can then unload the test results and load in the new vector and counter value.

7.0 Support for SRAM Access

All of the embedded SRAMs in Niagara2 are tested via MBIST however during silicon bringup activities it is necessary to read and write them from the JTAG port for effective debug. The process used to access the SRAMs during debug is referred to as Macro Test since it relies upon the Macro Test facility in the Fastscan ATPG tool. Briefly the process involves scanning the flops bounding the SRAM to the state required to perform the desired operation at the target address and then generating a clock to execute the operation within the SRAM. This scan and clock sequence can be repeated to read or write the contents of an entire SRAM if desired.

One of the challenges to implementing Macro Test in Niagara2 is created by the basic architecture of the SRAMs. Each SRAM is equipped with scannable input flops which is very good for testability of logic driving the SRAM. At the rising edge of a functional clock the SRAM is designed to operate on the data present at its boundary, point A, which is before the scannable input flops. The fact that each SRAM has boundary flops allows the logic designers to perform a full cycle of logic work between the previous flop stage and the boundary of the SRAM. Because of this design it is necessary to understand the upstream logic and to appropriately load the upstream flops to perform a Macro Test operation. This adds an undesirable level of complexity to the SRAM access process.

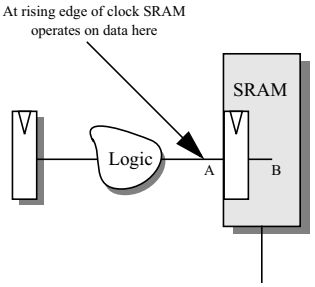


Figure 7.0.1: SRAM access via Macro Test

This complexity has been removed by putting the input flops into scan only mode during Macro Test. This prevents them from updating when the rising clock edge occurs and they hold their scan value. In this way the Macro Test user can set up the boundary flops directly to read or write the SRAMs. This is achieved by routing a special Scan_Enable signal to the SRAM input flops called SE_SCANCOLLAR_IN. Within the TCU there is a Macro Test mode bit which can be set by the user at the beginning of Macro Test operations. When the Macro Test mode bit is set the SE_SCANCOLLAR_IN signal is continually asserted. This allows setting the input flops via scan and also prevents them from updating when a functional clock is applied.

8.0 Memory BIST

There are 80 MBIST engines in Niagara2 dedicated to servicing more than 300 unique SRAMs. The SRAMs all have unique testing requirements driven by address size, bank and way configurations, data output widths, clocking specifics, and other factors resulting in each of the engines being unique to the SRAMs it is servicing. The generic MBIST to SRAM interface is illustrated in figure 8.0.1.

Note that the MBIST data bus is only 8 bits wide regardless of the width of the data input bus of the SRAM. The 8 bits are fanned out across the actual width of the SRAM data bus at the SRAM port. This saves a substantial amount of wiring since the engine is not usually immediately adjacent to the SRAM. The comparators can easily be shared between neighboring SRAMs further reducing test hardware and wiring between SRAMs and engine.

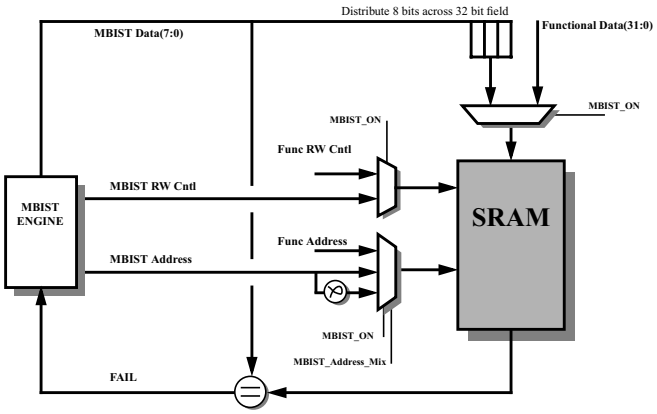


Figure 8.0.1: MBIST Engine to SRAM interface

8.1 What's all the RAWWCas!

The March C- memory march test [6], as shown in figure 8.1.1, forms the basis of the test algorithm used on Niagara2. The March C- test provides excellent DC test qualities and in the AC realm it provides a good write recovery test but not a read recovery test or a weak bit test.

$$\{\Uparrow(W_1);\ \Uparrow(R_1W_0);\ \Uparrow(R_0W_1);\ \Downarrow(R_1W_0);\ \Downarrow(R_0W_1);\ \Downarrow(R_1);\}$$

Figure 8.1.1: March C- Memory Test

The read recovery test executes a read operation of one data value immediately followed by a read operation of the opposite data value. The weak bit test targets bitcells that may be incapable of driving the bitlines correctly during a read operation. The worst case scenario for a read operation is defined as one where each of the bitcells in the target word are at the opposite polarity than all other bitcells on those bitlines. The read should be immediately preceded by a write of opposite polarity on the same bitlines. This test is dubbed Read After Write Worst Case (RAWWCas).

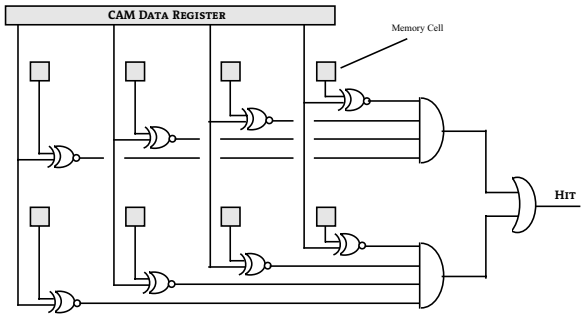
$$\{\Uparrow(W_1);\ \Uparrow(R_1W_0);\ \Uparrow(R_0W_1);\ \Downarrow(R_1W_0);\ \Downarrow(R_0W_1);\ \Downarrow(R_1);$$
$$\Uparrow(W_0W_1{}^*R_0R_1{}^*W_1);\ \Uparrow(W_0);\ \Downarrow(W_1W_0{}^*R_1R_0{}^*W_0);\}$$

Figure 8.1.2: March C- Memory Test with RAWWCas

The first six march elements in the test shown in figure 8.1.2 are the basic March C-, the last three march elements are the RAWWCas test together with a read recovery test. After march element six is complete the SRAM contains all ones. March element seven is composed of five actions. Action one writes zeros to all bits of a target location. The target location is now at a polarity opposite from all other locations in the SRAM. Action two writes all ones to a non-target location that occupies the same bitlines as the target location. The asterisk indicates that the actions occurs at a non-target location. Action three reads the zeros from the target location. At this point we have executed a read immediately after a write of opposite polarity with the worst case background: RAWWCas. Action four reads the ones from the non-target location to provide a read recovery bitline test. Finally action five writes the target location to all ones to match the background and the test is ready to move on to the next target in the address space. When march element seven is complete the SRAM is initialized to the opposite polarity in element eight and the RAWWCas test is repeated in element nine with the opposite test polarity.

8.2 CAM Test

Each CAM in Niagara2 can be read and written like a RAM. Figure 8.2.1 illustrates the test model we used for CAMs. The March C- with RAWWCas test is applied to each CAM during the RAM test phase. Once RAM testing is complete CAMs are further exercised in order to test the logic gates that generate the HIT signal.



$$\{\Uparrow(W_0);\Uparrow(W_1CAM_1W_0);\Uparrow(W_1);\Uparrow(W_0CAM_0W_1);\Uparrow(CAM_{walking0});\}$$

Figure 8.2.1: 4-bit 2-row CAM Model with CAM Test Algorithm

March element one initializes the CAM to all zeros. Element two writes ones to the target word and loads the CAM data register with ones. The CAM is activated and the HIT signal is checked to verify that it is active. March element two completes by restoring the target word to all zeros and moving on to the next address. At the end of element three all XNOR gates are verified to drive one for the one-one input condition. Element three initializes the CAM to all ones and element four checks for matches the same as element two. In this case the XNOR gates should drive one in response to the zero-zero input condition. Element five is designed to verify that each XNOR gate can detect a mismatch. All the memory cells are left in the all one state at the completion of element four. Element five loads the CAM data register with all ones except a single zero in the MSB. The CAM is activated and the HIT signal is checked to be inactive. The CAM data register is then reloaded with the zero shifted right and a one placed in the MSB. By walking a single zero across the entire CAM field in this way each XNOR gate can be verified to cause a mismatch and be reflected at the HIT signal.

8.3 MBIST Features

Address Mix - Most SRAMs have multiple physical addressing mechanisms such as row decoders, column decoders, bank or way selects. It is desirable to test each of these address decoders at speed during the MBIST process. Consider a 128 entry SRAM with 64 rows and 2 columns. Let the SRAM address bits [5:0] be the row address and bit [6] be the column select. The MBIST engine produces MBIST address bits [6:0] mapped to SRAM address bits [6:0]. As the MBIST engine performs its testing it will always treat MBIST address bit [0] as the LSB and it will change its value with each new target address. With this mapping the SRAM row decoder will be exercised at speed being made to change with every target address change. The column decoder being mapped to the MSB will only change every 64 address changes. When the entire test is complete the MBIST engine will assert the Address Mix signal and repeat the test. The Address Mix signal will cause the address mux to map MBIST address bits [0,6:1] to SRAM address bits [6:0]. This configuration results in the column decoder being exercised at speed, changing with each target address change while the row decoder changes only every other target address change. Refer to the symbol in front of the address mux in figure 8.0.1.

Data Values - The MBIST engines have four sets of data values programmed for use: AA/55, CC/33, 99/66, FF/00. In default MBIST mode the engine will deliver an eight bit data value AA wherever 1 appears in the data field of the algorithm and 55 wherever 0 appears. The test is run twice with this pair or data values, once in standard addressing mode the second time with Address Mix active. When the second run is complete the data value set is changed to CC/33 and the test is run twice again, then 99/66 and finally FF/00. There is a user data register in the MBIST engine that can be loaded with an arbitrary eight bit value. When MBIST is run in user data mode the value in the user data register will be used along with its bitwise complement. In this mode the test is complete after executing with this one data set.

User Address - The MBIST engine is equipped with three user registers: Address_Start, Address_Stop, and Address_Increment to exercise control over addressing. Using these registers the user can specify a specific address range to exercise as well as the size of increments to be taken through that space. This feature provides an excellent means of detecting intermittent failures in a know address space. In conjunction with the loop feature it also provides means to create a tight at-speed loop around a failing address thus facilitating efficient back side probing.

Loop - The MBIST engine can be put into endless loop mode. This is especially useful for backside probing activities.

BISI - The MBIST engine has a Built-In Self Initialization Mode. This is run as part of Power On Reset to initialize all SRAMs to the zero state. BISI consists of a write of all zeros to each address.

8.4 Some Unique Challenges

The MBIST interface and test algorithms described in the preceding sections constitute what we have come to call the "Vanilla" MBIST solution. Perhaps half of the SRAMs on Niagara2 received the vanilla treatment while the remainder of the SRAMs had some unique complicating feature requiring substantial customization of the MBIST engine servicing them.

8.4.1 Test Hardware Reduction

In an effort to minimize test hardware, comparators are shared between neighboring SRAMs whenever possible. Additionally, SRAMs with a data output bus larger than 64 bits are tested in segments with the entire memory test being performed as many times as is needed to cover all segments. Segmentation is performed by introducing a mux in the SRAM test data output path as shown in figure 8.4.1.1. In this way the test overhead is constrained to a single 64 bit comparator bank plus muxing. Wiring from SRAM to comparator is kept to just 64 wires.

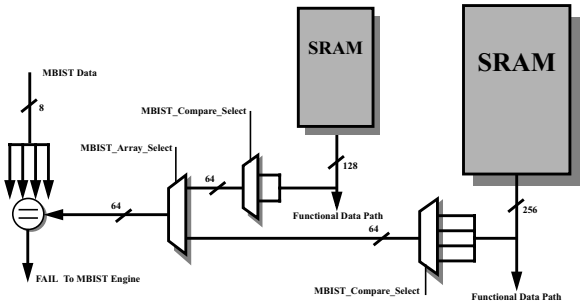


Figure 8.4.1.1: Partitioning data paths and sharing a comparator bank

8.4.2 FIFO Memories

Niagara2 has more than 200Kbits of FIFO memories which are used to pass data between two different on-chip clock domains hence their read and write clocks operate at different frequencies. This poses a challenge to the vanilla MBIST architecture which expects the read port, the write port, and the MBIST engine to all reside in the same clock domain. In order to keep the problem manageable it was decided to equip each FIFO write port with a custom clock mux. A mux control signal from the MBIST engine gives the read clock control over the write port during MBIST operation. In this way the FIFO operates entirely in the read clock domain during MBIST and of course the engine is placed in the read clock domain.

8.4.3 Double-Pumped Memories

The Network Interface Unit (NIU) requires a number of SRAMs that are capable of being read and written in the same cycle. This is usually implemented by designing a two port SRAM. The Niagara2 memory team has come up with a solu-

tion to this need that uses high speed one port SRAMs that are used in other parts of the chip. A 2X clock domain is created inside the SRAM custom block and the memory is run at twice the clock speed as the surrounding logic. In this way a read and write operation can occur on consecutive clock cycles within the SRAM custom block and appear to the surrounding logic as if they had been executed on the same clock cycle. Since the MBIST engines supporting these double-pumped memories are designed as part of the NIU logic, they runs at the normal NIU clock frequency. The vanilla MBIST algorithm has therefore been modified to test simultaneous read and write operations. March elements 2-5 in the test algorithm shown in figure 8.1.2 consist of consecutive Read and Write operations at the same address. For these double pumped arrays those operations occur simultaneously.

8.4.4 L2Data Cache: Two Cycle Access

The 4MB L2Data Cache is partitioned into eight banks with each bank partitioned into two sub-banks, top and bottom. Circuit specifications of this SRAM prohibit access to the same sub-bank on consecutive cycles. Top and bottom sub-banks do however share some address decoder circuitry and so it is desirable to exercise consecutive cycle accesses on one then the other sub-bank. The test algorithm of 8.1.2 was modified to jump back and forth between the top sub-bank and the bottom sub-bank on consecutive cycles. The modified algorithm is shown here in figure 8.4.4.1. A superscript has been added to each operation to indicate top or bottom sub-bank.

$$\{\Uparrow(W_1{}^{T}W_1{}^{B});\ \Uparrow(R_1{}^{T}R_1{}^{B}W_0{}^{T}W_0{}^{B});\ \Uparrow(R_0{}^{T}R_0{}^{B}W_1{}^{T}W_1{}^{B});$$
$$\Downarrow(R_1{}^{T}R_1{}^{B}W_0{}^{T}W_0{}^{B});\ \Downarrow(R_0{}^{T}R_0{}^{B}W_1{}^{T}W_1{}^{B});\ \Downarrow(R_1{}^{T}R_1{}^{B});$$
$$\Uparrow(W_0{}^{T}W_0{}^{B}W^{*}_1{}^{T}W^{*}_1{}^{B}R_0{}^{T}R_0{}^{B}R^{*}_1{}^{T}R^{*}_1{}^{B}W_1{}^{T}W_1{}^{B});$$
$$\Uparrow(W_0{}^{T}W_0{}^{B});\ \Downarrow(W_1{}^{T}W_1{}^{B}W^{*}_0{}^{T}W^{*}_0{}^{B}R_1{}^{T}R_1{}^{B}R^{*}_0{}^{T}R^{*}_0{}^{B}W_0{}^{T}W_0{}^{B});\}$$

Figure 8.4.4.1: Memory Test for L2Data Cache

8.4.5 CAM data needs full data input bus

One of the challenging aspect during the implementation of CAMBIST was how to provide the lookup-data for the walking 0 and walking 1tests without routing a full width data bus from the MBIST engine to the CAM. We created a CAMBIST data register local to each CAM. Each CAMBIST data register is configured as a shift register and is also equipped with initialization circuitry. Two control signals from the MBIST engine to the CAMBIST data register provide for all needed functions in the walking 0 and walking 1 tests. The INIT control signal causes the register to load all ones with a zero in the MSB. A SHIFT signal causes the register to shift its contents right with a wrap from LSB to MSB. A separate CAM signal activates the CAM operation to perform the test. By using this method we were able to avoid routing a large amount of data signals from the mbist engine to CAMs and still achieved effective CAM tests.

8.4.6 TCAM

There is a Ternary CAM located in the NIU block which is responsible for ethernet packet processing. Each TCAM cell has a mask-bit(m) and a data-bit(d). When a mask bit is set to'0' a match occurs regardless of the status of the data bit. The mask and data bits share common data-in and data-out lines for read and write operations. In the event of a multiple row hit, priority encoder logic reports the smallest matched address.

An MBIST engine has been designed to test the read, write, and compare operations of the TCAM. The read/write basic operations are covered in the same manner as described in section 8.1. The TCAM's compare operation is exercised by implementing the algorithm shown in table 8.4.6.1 based on the vanilla CAM test presented in section 8.2 Each test sequence is designed such that it can be independently run for characterization and bring-up purposes.

Test Sequence	Explanation
$\Uparrow(W_{m1})$	Set all mask bits to 1
$\Uparrow(W_{d0});\ \Uparrow(W_{d1}CAM_1W_{d0});$	Match with all data 1's
$\Uparrow(W_{d1});\ \Uparrow(W_{d0}CAM_0W_{d1});$	Match with all data 0's
$\Uparrow(W_{d1});\ \Uparrow(CAM_{walking0});$	Mismatch: walking 0 across bits
$\Uparrow(W_{d0});\ \Uparrow(CAM_{walking1});$	Mismatch: walking 1 across bits
$\Uparrow(W_{d1});\ \Uparrow(W_{m0}CAM_0W_{m1});$	With mask bits 0 and opposite values in data bits, still should match!
$\Uparrow(W_{d1});\ \Uparrow(CAM_1W_{d0});$	Testing priority encoding: hit-addr = current_address
$\Uparrow(W_{d0});\ \Uparrow\{CAM_0R_{m1}\};$	Simultaneous compare and read; read should also happen; hit-addr = 0

Table 8.4.6.1: TCAM Test Sequence

9.0 Direct Memory Observe

Bitmapping of embedded SRAMs is a critical activity in the early stages of chip bringup, especially in a new technology. MBIST is an effective test mechanism but does not lend itself unassisted to bitmapping activity. Muench et al describe an MBIST+ procedure that calls for running MBIST to a stopping point then scanning out the read data from a scannable register. This action is repeated moving the stopping point with each run to access subsequent SRAM locations. The most notable feature of this procedure is the excessive amount of time it takes. We have designed a fast efficient mechanism to take advantage of the existing MBIST resources for bitmapping applications. By simply routing the data output bus from the embedded SRAM to dedicated package pins the output data can be Directly Observed at the tester during MBIST; Direct Memory Observe (DMO). This has the desirable qualities of being an at-speed stimulus of the SRAM along with easy from-

the-pins observation for the tester. There are two major challenges to applying this approach: excessive global wiring and a speed mismatch between SRAMs and IO.

The largest arrays on the chip were targeted for DMO. The ICache and DCache in the cores, the L2Data and L2Tag which are partitioned into eight banks as seen in figure 1.0.1, and fifteen of the largest SRAMs in the Ethernet Network Interface Unit. Bit reduction is performed on the MBIST legs of all of the SRAM data output buses as described in section 8.4.1. DMO data buses from SRAMs within a cluster are muxed together before leaving the cluster. In this way the DMO data bus from any cluster back to the TCU is no more than forty bits. The DMO data bus from the TCU to the IO block is forty bits.

The IO blocks for the debug port operate at 350MHz while the SRAMs in the cores and L2 operate at 1.4GHz; a speed mismatch. To solve this problem the TCU is equipped with time multiplexing logic. It takes the SRAM data in at speed and performs sampling based on a user programable register. In one configuration it may sample and hold 1.4GHz SRAM data for four clock cycles and thereby present a data rate of 350MHz to the IOs; within the capability of the IOs to respond and the tester to strobe. In order to get all the SRAM data to the IOs it is necessary to run the MBIST four times changing the sampling offset while keeping the same sampling rate for each run.

10.0 Logic BIST

The SPARC cores in Niagara2 are equipped with STUMPS [1] architecture LBIST with individual MISR channel masking. For purposes of LBIST, the cores are partitioned into fourteen scan chains with the longest chain being about eight thousand flops. The on chip PLL is locked and running during the execution of LBIST in order to provide for at-speed testing. Built into each cluster clock header is a clock stop function used by the TCU to stop clocks in response to a debug event in preparation for a scan dump. The LBIST engine controls the stop function to the core cluster header. It then performs enough scan cycles to fill all the scan chains. Once the chains are filled the LBIST engine deasserts the clock stop signal for two core clock cycles and the at-speed test is executed. An interesting chicken-and-egg challenge arose when we realized that the LBIST logic itself is part of the core cluster and was stopping its own clocks as part of the test sequence. It was necessary to separate the LBIST logic from the core logic with regards to cluster clocking control in order to make the scheme function. Extra care was taken to control skew between the core clock domain and this tiny new LBIST clock domain. The clock stop signal is the only at-speed signal that crosses the LBIST/core domain boundary.

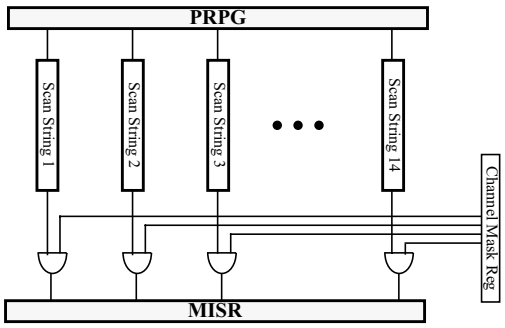


Figure 10.0.1: LBIST Block Diagram for SPARC Cores

In order to calculate LBIST test coverage the core is modeled in Fastscan as if all clocking and scan controls are available at the model boundary. The outputs of the PRGP are treated as virtual scan inputs and the inputs to the MISR are treated as virtual scan outputs. A single stuck-at pattern is generated and written out giving us the topography of the LBIST scan chains. Then a simple PRPG emulator is written in C allowing us to determine the contents of the scan chains at the end of every LBIST scan phase. The contents are then written out as vectors and the final set of vectors are fault simulated in Fastscan. MISR values will be determined empirically. Final LBIST coverage numbers are not yet available.

11.0 1149.6 Support

PCI-E and Ethernet require AC-coupled differential interconnections on their SERDES interface. IEEE Std. 1149.6 extends Std. 1149.1 boundary scan structures and methods in order to ensure simple, robust, and minimally intrusive boundary-scan testing of advanced digital networks [4]. Niagara2 supports IEEE Std. 1149.6. with both AC-test instructions, EXTEST_PULSE and EXTEST_TRAIN. There is also a 210 pin debug port which utilizes standard IO signaling. The debug port IOs support JTAG 1149.1 boundary scan testing.

12.0 Summary

The SUN Microsystems Niagara2 System on a Chip is a significant DFT challenge. With well chosen testability guidelines in place the team has been able to achieve greater than 98% stuck-at test coverage. Embedded SRAMs are covered completely by at-speed MBIST equipped with a rich feature set supporting debug, bitmapping, and failure analysis. We have introduced a number of original DFT solutions. A hybrid flop design that combines the design advantages of edge triggered flip-flops with the hold time immunity of LSSD master-slave latch designs. We have presented the RAWWCas weak bit memory test. We have also presented Direct Memory Observe, a useful combination of MBIST with direct pin access to greatly facilitate embedded SRAM bitmapping. Clock domain management across the chip posed a significant challenge for debug clock control and the reliable application of scan test vectors crossing clock domain boundaries. LBIST

embedded in reusable IP such as the processor core creates the possibility for completely independent self test of that block in future integrations.

Acknowledgments

The authors would like to acknowledge SUN colleagues Ray Heald and PJ Tan for their role in developing the RAWW-Cas memory test. And Paul Dickinson for his participation in the development and productization of the Direct Memory Observe feature.

References

[1] P. H. Bardell and W. H. McAnney, Self-Testing of Multichip Logic Modules, Proceedings ofthe 1982 IEEE Interntional Test Conference, Nov. 1982, pp. 200-204.

[2] C.Pyron, M.Alexander, J.Golab, G. Joos, B.Long, R.Molyneaux, R.Raina, N.Tendolkar, DFT Advances in the Motorola MPC7400, a PowerPC G4 Microprocessor, Proceedings of the 1999 IEEE International Test Conf., Sept. 1999, pp141

[3] I.Robertson, G.Hetherington, T.Leslie, I.Parulkar and R.Lesnikoski, Testing High-Speed, Large Scale Implementation of SerDes I/Os on Chips, Proceedings IEEE International Test Conference, 2005.

[4] IEEE Std 1149.6-2003, IEEE Standard for Boundary- Scan Testing of Advanced Digital Networks

[5] R.Muench, T.Munns, W.C.Shields, Bitmapping the PowerPC 604 Cache Using ABIST, Teradyne User Group Conf, April 1996.

[6] A. J. van de Goor, Testing Semiconductor Memories: Theory and Practice, John Wiley & Sons, New York, USA, 1991.