Boundary Scan Speeds Static Memory Tests

Jim Coleman and Richard Thorpe
Texas Instruments Incorporated
Semiconductor Group

SCTA030

Reprinted from ELECTRONIC DESIGN February 1993

BOUNDARY SCAN SPEEDS STATIC MEMORY TESTS

THE BOUNDARY-SCAN STANDARD CAN SUPPLY ACCESS NEEDED TO CONTROL BUILT-IN SELF-TEST FUNCTIONS.

JIM COLEMAN and RICHARD THORPE
Texas Instruments Inc., Semiconductor Group, Test Technology Center, 6500 Chase Oaks Blvd., MS 8407, Plano, TX 75086; (214) 575-2577.

The steady increase in memory densities used in microprocessor-based systems has stretched functional test times, as large memories require time-consuming tests for adequate fault coverage. Moreover, using high-density ASICs with embedded memory limits physical test access. Wide buses and high bus fan out further complicate testing by limiting component fault isolation. Fortunately, the IEEE-1149.1 boundary-scan standard offers a solution to the problem of testing static memory.

The boundary-scan standard was initially conceived to test board interconnections without the need for physical probing. The standard requires the inclusion of boundary-scan functions on ICs, and its success is evidenced by willingness of silicon vendors to add that capability to their new products. Boundary-scan devices available now include general microprocessors, digital-signal processors, field-programmable gate arrays, ASIC libraries (standard cell and gate arrays), and bus-interface components.

Designers of microprocessor-based applications typically buffer the processor's address and data buses to solve electrical loading or isolation problems. If the bus-interface parts are IEEE-1149.1 compatible, then boundary-scan functions can be used to test the memory.

One technique uses boundary-scan instructions to scan in the address value and the data value, set a memory strobe active to perform the memory access, and set the memory strobe inactive to complete the cycle. Because these multiple-scan steps must be done for each memory address, this technique is very scan-intensive and therefore very time consuming. For extremely large memory arrays, the process could require millions of IEEE-1149.1 scan operations and take hundreds of minutes to perform.

A better solution, which can speed up test execution time by a factor of hundreds, employs IEEE-1149.1-controlled built-in self-test (BIST) with off-the-shelf components or ASIC macros. With either method, the fault detection and isolation provided by a given memory test will depend on the stimulus patterns used. To compare the two techniques, a 256-by-8-bit memory array and associated bus-interface and control logic was constructed *(Fig. 1)*. This configuration allows for explicit read/write operations using the IEEE-1149.1 Extest and Sample instructions. The IEEE-1149.1 components, which have a BIST capability controlled by boundary-scan methods, can also perform the memory read/write operations.

Two tests were run on the memory. The first explicitly scanned in the RAM array address, data, and strobe signal; the second executed IEEE-1149.1-controlled BIST, which generated the address, data, and strobe signals automatically at the test clock (TCK) rate of 6.25 MHz. The boundary-scan technique solved the problem of direct physical access, but was time consuming *(Table 1)*. The second test, however, clearly showed the advantage of IEEE-

TABLE 1: BOUNDARY SCAN VERSUS CONTROLLED BIST

Mode	256 accesses	1,000,000 accesses
IEEE 1149.1 (Extest & Sample)		
Time to apply	4.8 sec.	332.00 min.
Scans	512	2,000,000
Patterns	512	2,000,000
IEEE 1149.1 (with BIST capability)		
Time to apply	0.011 sec.	0.75 min.
Scans	7	28,000
Patterns	512	2,000,000

1149.1-controlled BIST. That is, by using IEEE-1149.1-controlled BIST circuitry embedded within the functional logic surrounding large memory arrays, designers can closely emulate the actual functional characteristics and timing speeds of the memory being accessed.

This article describes how to implement this solution using Texas Instruments Scope bus-interface components. These are off-the-shelf devices that offer the IEEE-1149.1-controlled BIST functionality needed to test static memory. The Scope devices also supply the electrical signal conditioning and buffering a design engineer would typically design around a microprocessor.

Generating Patterns

First, a few comments regarding deterministic and algorithmic patterns. Using algorithmically generated patterns for memory-array tests is an accepted engineering practice commonly used in software-based built-in test (BIT) code. Deterministic patterns that can't be generated with a BIST algorithmic circuit would require more memory to store the patterns than the memory array that's being tested.

Various memory-testing algorithms are available. Each technique specializes in detecting and isolating particular memory faults. Once the engineer analyzes the memory fault classes and testing approaches applicable to a given design, the appropriate BIST structures can be integrated into the processor, ASIC, or bus-interface components.

The size of the memory array is important in determining the desired width of the BIST structure bounding the address and data buses. An octal bus-interface component can generate only 256 unique patterns from its BIST circuit. So, if two 8-bit octals are used on a 16-bit bus, the BIST must be executed 256 times to cover a 64-kbit memory space.

The Scope 8-bit octal bus interface components provide flexible, generic types of BIST structures that support several memory-testing algorithms. The 8-bit biCMOS technology (BCT) devices have BIST structures for pseudo-random pattern generation (PRPG), parallel signature analysis (PSA), concurrent PSA/PRPG, and Toggle/Sample. The 8-bit advanced biCMOS technology (ABT) versions have the same capabilities as the BCT parts, and add a Count-up (256 patterns) function at the outputs. The 18- and 20-bit Widebus Scope bus interface components generate 256-kbit and 1-Mbit unique patterns, respectively, in one BIST execution. These parts also have advanced BIST capabilities, including the Count-up function of the ABT devices.

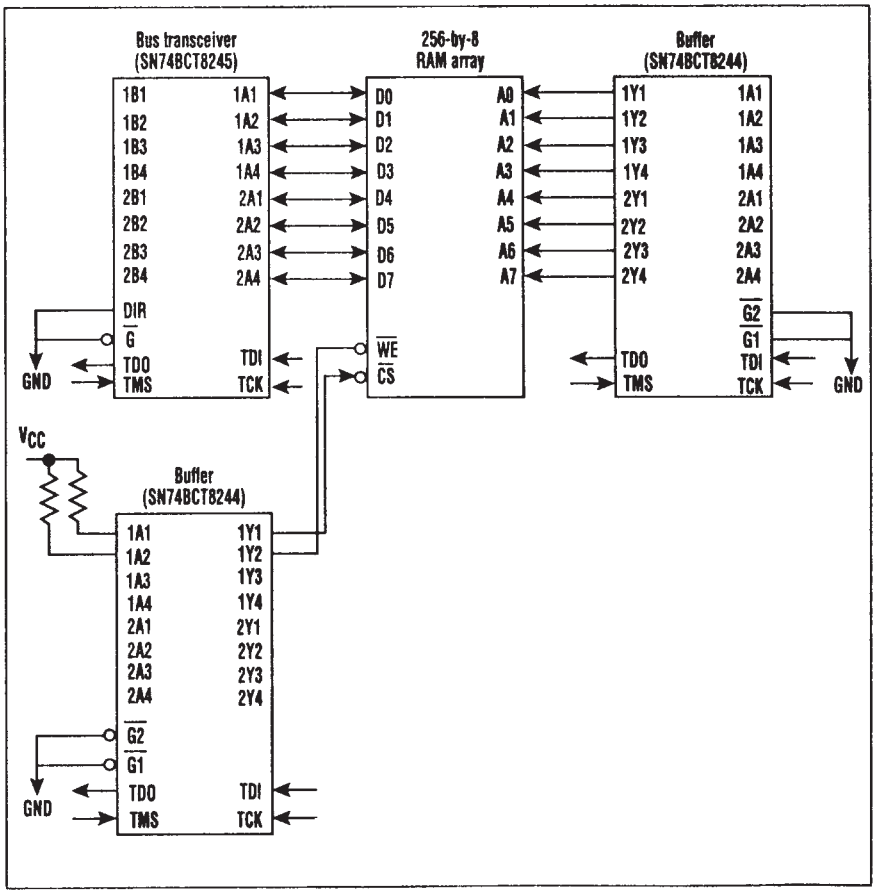
TABLE 2: SCAN SEQUENCE FOR PRPG

Step	Type	Data scanned into device	Register accessed
1	IR scan	READBN opcode	Instruction
2	DR scan	Initial seed (16 bits)	Boundary register
3	IR scan	SCANCN opcode	Instruction
4	DR scan	b'01' (PRPG)	Boundary control
5	IR scan	Runt opcode	Instruction
6	IR scan	Bypass opcode	Instruction

The example circuit uses the BCT octal bus interface components, so a review of their BIST functions is appropriate at this point. Specifically, the following descriptions of PRPG, PSA, combined PSA/PRPG, and Toggle/Sample patterns apply to the BCT8240, BCT8244, BCT8245, BCT8373, and BCT8374.

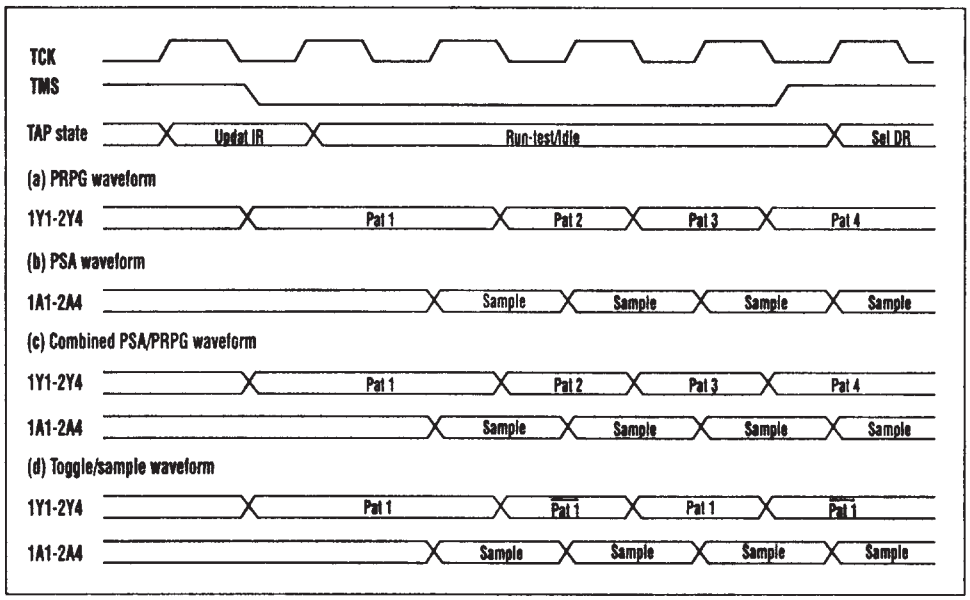
For PRPG, the patterns are generated at the functional outputs. The user should select an initial seed value and scan it into the boundary-scan register before performing the scanning sequence required to place the device into PRPG *(Table 2)*. After the Instruction Register (IR) scan of step 5 is completed and the test access port (TAP) has entered the Run-Test/Idle state, the device outputs begin generating pseudo-random patterns.

The timing relationship of these patterns to TCK and test mode select



1. A SIMPLE CIRCUIT CONSISTING of a memory array and associated bus-interface and control logic was constructed in order to evaluate two memory test methods.

BOUNDARY-SCAN MEMORY TESTS



2. THIS timing diagram shows the relationship between the IEEE-1149.1 signals—TCK, TMS, and TAP state—and the different types of test data signals that can be used to test the memory.

(TMS) is given in Figure 2a. The Scope octal's outputs change value after the falling edge of TCK while the TAP is in Run-Test/Idle and the current instruction register is loaded with the instruction Runt.

All Scope octals use an internal 16-bit linear feedback shift register (LFSR) for PRPG. The devices must receive a total of 65,535 TCKs in order to generate every sequence combination of 8-bit (octal) output values. The sequence of these values is pseudo-random, based on the initial seed value written into the boundary-scan register. The seed can be any value between x0001 and xFFFF. A seed of x0000 causes the LFSR to remain at x0000. After 65,535 TCKs, the output pattern sequence begins to repeat.

The 16-bit PRPG function can be used to generate input patterns for the data bus when performing memory write operations. This function would not normally be used for generating address patterns because it would take more than 256 TCKs to ensure that all 256 memory locations were accessed.

USING PSA

When the PSA function is used, data appearing on the eight functional data inputs is compressed into a 16-bit signature. As with PRPG, an initial seed value must be scanned into the boundary-scan register before the scanning sequence required to place the device into PSA is performed *(Table 3)*. The compression occurs after the IR scan of step 5 is completed and the TAP enters the Run-Test/Idle state.

The timing relationship of the data sampling to TCK is shown in Figure 2b. The A-inputs of the Scope octal devices are sampled on the rising edge of TCK, while the TAP is in Run-Test/Idle and the current instruction register is loaded with the instruction Runt.

The PSA function uses the 16-bit LFSR to generate the 16-bit signature, based on the initial seed value loaded in the boundary-scan register. The 16-bit seed can be any value between x0000 and xFFFF. The seed value chosen will affect the validity of the signature and the detection of faulty patterns. Therefore, the engineer should carefully study the nature of the LFSR's signature analysis in order to understand aliasing conditions. Aliasing occurs when multiple input-pattern combinations produce the same final signature. If that happens, certain memory data faults could be masked and thus go undetected.

The PSA mode would typically be used when verifying the validity of known data previously written to a block of memory. Using the 16-bit PSA function of a Scope octal device reduces the chance of aliasing.

As noted, the Scope octals can also combine PRPG and PSA operations. In this mode, the devices simultaneously generate pseudo-random patterns on the outputs while compressing a signature on the inputs. The IEEE-1149.1 scan operations needed to set up a device for combined PSA/PRPG operation are similar to those for the PSA mode *(Table 3, again)*. One difference is that the 16-bit seed value is split into two 8-bit seeds, one for PRPG and the other for PSA (step 2). In addition, the boundary-control register should be loaded with a b'11' value (step 4). After the IR scan of step 5 is completed and the TAP enters the Run-Test/Idle state, the octal's outputs begin generating PRPG patterns and the inputs are compressed. The timing relationship of these operations to TCK is shown in Figure 2c.

Because the two 8-bit seeds split the 16-bit LFSR, only 255 TCKs are required to generate every PRPG output value, and the device generates only an 8-bit PSA signature. After 255 TCKs, the output PRPG sequence repeats itself. Because of the nature of an LFSR, the eight outputs will never be all zeros. Because only 8 bits of the LFSR are used for PSA, the possibility for aliasing increases. Consequently, the 8-bit PRPG func-

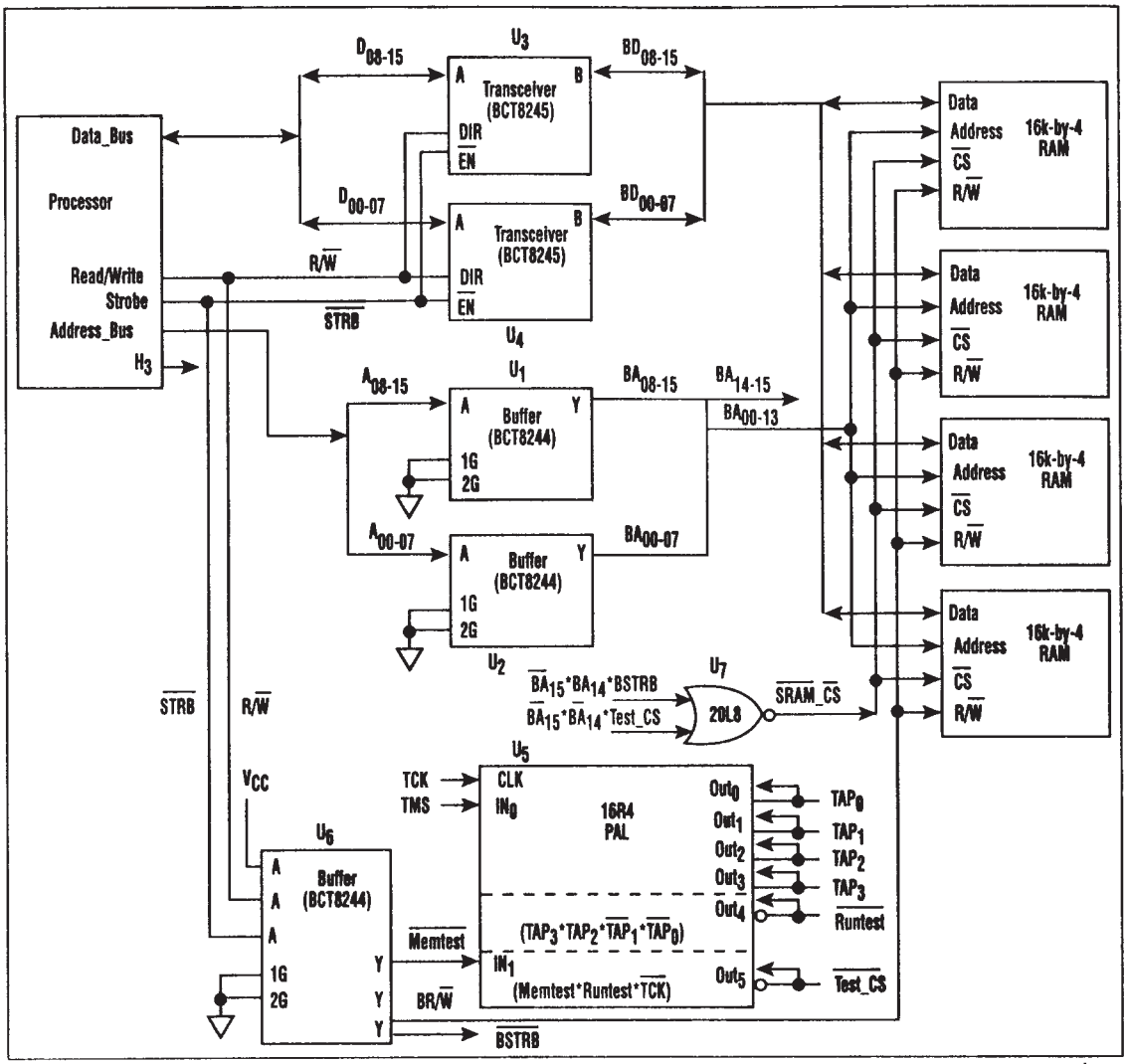
TABLE 3: SCAN SEQUENCE FOR PSA

Step	Type	Data scanned into device	Register accessed
1	IR scan	READBN opcode	Instruction
2	DR scan	Initial seed (16 bits)	Boundary Register
3	IR scan	SCANCN opcode	Instruction
4	DR scan	b'10' (PSA)	Boundary Control
5	IR scan	Runt opcode	Instruction
6	IR scan	READBN opcode	Instruction
7	DR scan	Resulting signature	Boundary Register

BOUNDARY-SCAN MEMORY TESTS

tion could be used to produce memory addresses, since 255 address values can be generated within 255 TCKs. But the 8-bit PSA function would not be recommended for sampling memory data because of the greater possibility of aliasing.

In the Toggle/Sample mode, the octal device generates a toggle pattern at the functional outputs while the inputs are sampled. Once again, the scan operations used to set up the octal for Toggle/Sample mode are similar to those for PSA setup. Two 8-bit seeds are entered (step 2). The first is the initial value of the toggle pattern at the outputs, and the second isn't used. To configure the octal for this BIST function, the boundary control register should be loaded with a b'00' value (step 4). As in the other modes, the octal's outputs begin to toggle after the IR scan of step 5 is completed and the TAP enters the Run-Test/Idle state.



3. IN THE EXAMPLE TEST CIRCUIT, several bus-interface devices partition the address and data buses to facilitate boundary scan around the microprocessor.

The timing relationship of these patterns to TCK is shown in Figure 2d. The outputs of the Scope octal change value after the falling edge of TCK, and the TAP is in Run-Test/Idle and the current instruction register is loaded with the instruction Runt. The toggling output function could also be used for generating memory data input patterns, such as 55/AA, FF/00, and so on.

An example circuit consists of a generic microprocessor, a static RAM array, two PALs, and several Scope octal ICs *(Fig. 3)*. The microprocessor has a 16-bit address bus and a 16-bit data bus, as well as read/write (R/W) and address strobe ($\overline{STRB}$) control signals. The Scope octal devices partition the address and data buses to facilitate boundary scan around the microprocessor. Specifically, U_1 and U_2 (BCT8244 types) buffer the microprocessor's address bus, and U_3 and U_4 (BCT8245 types) are transceivers on the microprocessor's data bus. U_6 (a BCT8244) buffers the processor's R/W signal and other control signals.

TABLE 4: MEMORY MAP AND ADDRESS DECODER

Address range BA_{00}-BA_{15}	Memory selected
0000-3FFF	Not defined
4000-7FFF	RAM array (to be tested)
8000-FFFF	Not defined

!SRAMCS = ((!BA15 & BA14 & BSTRB) # (!BA15 & BA14 & TSTCS))

For simplicity, the example circuit doesn't detail the data-transfer acknowledgement logic. The signal H_3 is the microprocessor's functional clock, from which all functional read/writes are timed *(Fig. 4)*.

The circuit's memory map and address decoder equation, including the buffered upper address lines, BA_{14}, and BA_{15}, are defined in Table 4. To select the memory resource to be tested by the BIST function, BA_{14} and BA_{15} must be controlled properly to generate the necessary chip select. The 20L8 PAL, U_7, generates a chip select to the RAM array, $\overline{SRAM\_CS}$.

The logic equation generating $\overline{SRAM\_CS}$ has two product terms

BOUNDARY-SCAN MEMORY TESTS

(P-Terms). The first generates the "functional operation" chip select; the second supports IEEE-1149.1 BIST testing. When BA_{15} is a logic 0 and BA_{14} is a logic 1 and $\overline{STRB}$ is a logic 0, the output $\overline{SRAM\_CS}$ is a logic 0. This enables the RAM array (via the chip-select pin).

The second term is logically ORed with the first and supports the BIST operation. U_1 (the octal that buffers the upper address bus) must be loaded with the Extest instruction and must drive the upper address lines, BA_{00-15}. The output $\overline{SRAM\_CS}$ is a logic 0 when BA_{15} is a logic 0, BA_{14} is a logic 1, and $\overline{Test\_CS}$ is a logic 0. The lower address lines, BA_{00-07}, are driven by U_2's PRPG BIST circuit.

The 16R4 PAL, U_5, generates the $\overline{Test\_CS}$ signal at the appropriate time to allow the IEEE-1149.1 BIST functions to time properly with the RAM circuitry. This device holds three logic components.

The first logic component has two inputs—TCK and TMS. The TMS signal is sampled on every rising edge of TCK, just as in other IEEE-1149.1 devices. By sampling TMS, the PAL logic can monitor the TAP state of the IEEE-1149.1 scan bus. The TAP has 16 possible states, encoded into four outputs defined as TAP_0-TAP_3 *(Table 5)*.

The PAL's second logic component generates the output signal $\overline{Runtest}$. This signal is asserted (logic 0) whenever the TAP is in the Run-Test/Idle state. This signal, which indicates when the TAP is in Run-Test/Idle state, is important. When the TAP is in this state and the octal is loaded with the Runt instruction, the BIST circuitry generates pseudo-random patterns or compresses a PSA signature.

The third logic component actually generates the $\overline{Test\_CS}$ signal. While the octals are performing the BIST in the Run-Test/Idle state, this PAL needs to know when the Runt instruction is loaded into the octals. A scannable signal, $\overline{Memtest}$, is used to accomplish this. U_6, a BCT8244 octal buffer, drives this signal.

The equation for $\overline{Test\_CS}$ is:

$\overline{Test\_CS}$ = (Memtest * Runtest * $\overline{TCK}$)

Software performing the IEEE-1149.1 scan operations must set the $\overline{Memtest}$ signal low at the same time that the Runt instruction is loaded into the octals performing PRPG and PSA.

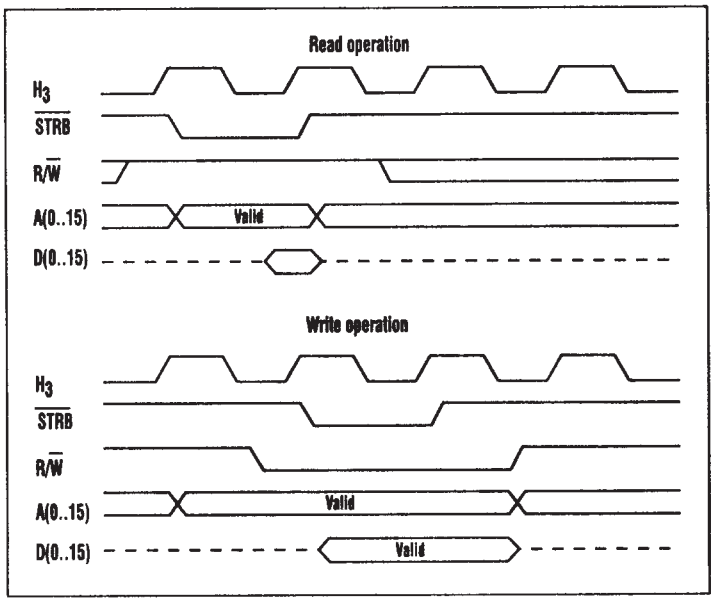
The designer must be able to shut off the microprocessor or hold it in a reset state while the test is executing. Thus, the processor should be controllable via an IEEE-1149.1 scannable register. Table 6 defines the scan operations required to perform BIST memory writes to RAM addresses 4001-40FF, using pseudo-code to generate both the address and data patterns. Device U_2 generates pseudo-random addresses on signals BA_{00-07}, and devices U_3 and U_4 generate pseudo-random data on signals BD_{00-15} *(Table 6)*.

The following steps execute the BIST memory write operation:

Step 1 loads all Scope octals with a READBN instruction, which allows access to the octal boundary-scan register (BSR) while the octals remain in their functional mode.

Step 2 involves a Data Register scan that initializes the BSRs of each octal device. U_1 (BA_{08-15}) is set up with the desired RAM memory address and U_2 (BA_{00-07}) is loaded with a PRPG seed value. Both U_3 (BD_{08-15}) and U_4 (BD_{00-07}) are loaded with a PRPG seed value. U_5 is set up with the Memtest signal asserted (logic 0), and the read/write-signal (BR/$\overline{W}$) logic level selecting a write (logic 0) operation.

Step 3 is an Instruction Register scan that loads the octals U_2, U_3, and U_4 with the SCANCN instruction. This instruction allows access to the boundary control register (BCR). U_1



4. THE READ/WRITE WAVEFORMS for the example circuit show the relationship between the microprocessor's functional clock, H_3, and the appropriate signals.

TABLE 5: DEVICE U5 TAP STATE DEFINITION

Output				
TAP_3	TAP_2	TAP_1	TAP_0	TAP State
1	1	1	1	Strap
1	1	0	0	Run-Test/Idle
0	1	1	1	Select_DR
0	1	0	0	Select_IR
0	1	1	0	Capture_DR
0	0	1	0	Shift_DR
0	0	0	1	$Exit_1$_DR
0	0	1	1	Pause_DR
0	0	0	0	$Exit_2$_DR
0	1	0	1	Update_DR
1	1	1	0	Capture_IR
1	0	1	0	Shift_IR
1	0	0	1	$Exit_1$_IR
1	0	1	1	Pause_IR
1	0	0	0	$Exit_2$_IR
1	1	0	1	Update_IR

!RUNTEST = (TAP_3 * TAP_2 * !TAP_1 * !TAP_0)

BOUNDARY-SCAN MEMORY TESTS

TABLE 6: PSEUDO-CODE FOR MEMORY WRITE OPERATIONS

Sequence and type	BA_{08}-BA_{15} U_1	BA_{00}-BA_{07} U_2	BD_{08}-BD_{15} U_3	BD_{00}-BD_{07} U_4	$\overline{\text{Memtest}}$ and BR/W U_5
			**Shut off the embedded microprocessor		
1. IR	READBN	READBN	READBN	READBN	READBN
2. DR	$BA_{15}=0$ $BA_{14}=1$ $BA_{08\text{-}13}=000000$	LFSR seed	LFSR seed (set direction A → B)	LFSR seed	BR/W = 0 $\overline{\text{Memtest}}=0$
3. IR	Bypass	SCANCN	SCANCN	SCANCN	Bypass
4. DR	Bypass 0	PSA/PRPG 11	PSA/PRPG 11	PSA/PRPG 11	Bypass 0
5. IR	Extest	Runt	Runt	Runt	Extest
			**Hold in 1149.1 Run-Test/Idle state for 255 Test Clocks (TCKs)		
6. IR	READBN	READBN	READBN	READBN	READBN
7. DR	$BA_{15}=0$ $BA_{14}=1$ $BA_{08\text{-}13}=000001$	LFSR seed	LFSR seed (set direction A → B)	LFSR seed	BR/W = 0 $\overline{\text{Memtest}}=0$

and U_5 are loaded with the Bypass instruction.

Step 4 is a Data Register scan that loads the U_2, U_3, and U_4 BCRs with the PSA/PRPG code (11).

Step 5 is an Instruction Register scan that loads U_2, U_3, and U_4 with the Runt instruction and loads U_1 and U_5 with the Extest instruction. Extest allows the previously loaded values for BA, BR/$\overline{W}$, and $\overline{\text{Memtest}}$ to be asserted when the TAP enters the Update-IR state. When the TAP enters the Runtest/Idle state, U_2, U_3, and U_4 begin generating PRPG patterns *(Fig. 5)*.

Step 6 is an Instruction Register scan that puts the octals into their functional mode. It also deasserts U_5's Memtest signal (logic 1), preventing Test_CS from being generated when the Runtest/Idle state is reentered.

Step 7 is a Data Register scan that sets up the memory write operation for the next block of memory.

Similarly, the scan operations needed for memory read operations read from RAM addresses 4001-40FF. U_2 generates pseudo-random addresses on address bus signals $BA_{00\text{-}07}$, and devices U_3 and U_4 compress a signature from data bus signals $BD_{00\text{-}15}$ *(Table 7)*.

The following steps are required in order to execute the memory read operation:

Step 1 loads the octals with the READBN instruction, which allows access to the octal BSR while the octals remain in their functional mode.

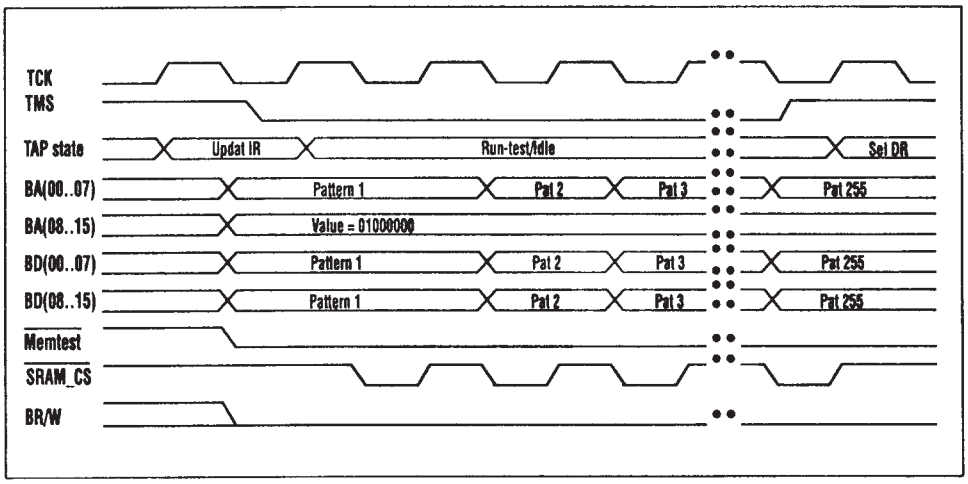
Step 2 is a Data Register scan that initializes the Scope octals' BSRs. U_1 ($BA_{08\text{-}15}$) is set up with the desired RAM memory address and U_2 ($BA_{00\text{-}07}$) is loaded with a PRPG seed value. Both U_3 ($BD_{08\text{-}15}$) and U_4 ($BD_{00\text{-}07}$) are loaded with a PSA seed value. U_5 is set up with the Memtest signal asserted (logic 0), and the read/write signal BR/W logic level selects read (logic 1).

Step 3 is an Instruction Register scan that loads octals U_2, U_3, and U_4 with the SCANCN instruction, permitting access to the boundary control register. U_1 and U_5 are loaded with the Bypass instruction.

Step 4 is a Data Register scan that loads the U_2, U_3, and U_4 boundary control register with the PSA/PRPG code (11) and PSA code (10).

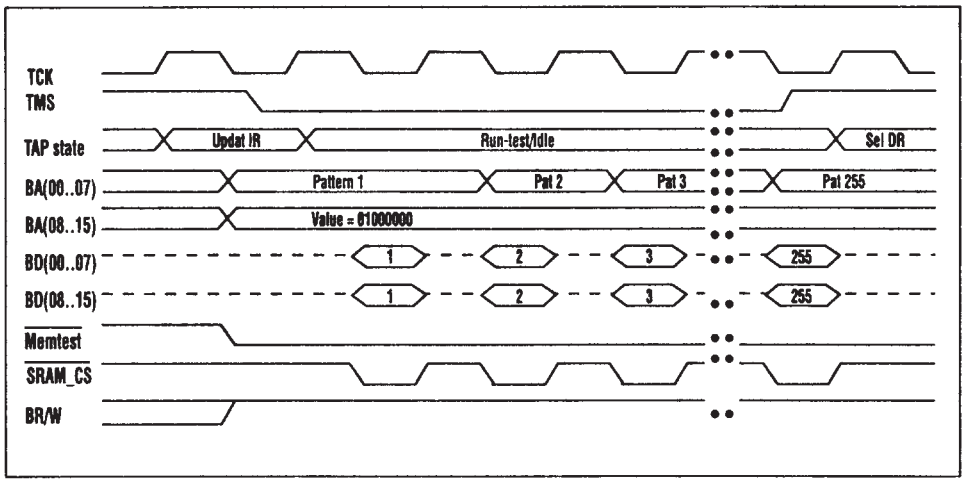
Step 5 is an Instruction Register scan that loads U_2, U_3, and U_4 with the Runt instruction and loads U_1 and U_5 with the Extest instruction. The Extest instruction allows the previously loaded values for $BA_{08\text{-}15}$, BR/$\overline{W}$, and $\overline{\text{Memtest}}$ to be asserted when the TAP enters the Update-IR state. When the TAP enters the Runtest/Idle state, U_2 begins generating PRPG patterns and U_3 and U_4 begin compressing a signature.

Step 6 is an Instruction Register scan that puts the octals into functional mode. It also deasserts octal device U_5's Memtest signal (logic 1),



5. THE BIST-GENERATED PRPG patterns for memory write operations are created after U_2, U_3, and U_4 are loaded with the Runt instruction, and the TAP enters the Run-test/Idle state.

BOUNDARY-SCAN MEMORY TESTS



6. THE BIST-GENERATED waveforms for memory read operations are created after the Data Register scan that reads the signature from octals U_3 and U.

preventing Test_CS from being generated when the Runtest/Idle state is reentered.

Step 7 is a Data Register scan that reads the signature from octals U_3 and U_4. In adition, it sets up the memory read operation for the next block of memory.

The BIST-generated waveforms for the memory read operation are shown in Figure 6.

The designer can repeat these procedures to access other locations in the RAM array by changing the value of U_1's boundary-scan register output signals, BA_{08-13}. Because U_1 operates in the Extest mode, software has to increment and scan out each of these addresses.

During Read cycles, the data bus, BD_{00-15}, becomes valid during the assertion of SRAM_CS (after the RAM access time). The bus is sampled by devices U_3 and U_4 on the rising edge of TCK, which is when SRAM_CS is being deasserted. The address bus changes value after the falling edge of TCK (after the propagation time, TCK-to-Q valid). The calculation of the minimum TCK period must consider the address setup time, RAM access time, and the octal data setup and hold times.

OTHER CONSIDERATIONS

The 8-bit Scope octals offer two practical patterns for generating BIST addresses to a block of memory: 8-bit PRPG and Binary Count-up (for ABT parts only). The devices also supply four ways of writing data to a block of memory: 8-bit PRPG, 16-bit PRPG, Toggle, and Binary Count-up. The parts allow for reading BIST data from a block of memory by using 16-bit PSA data compression.

The use of 8-bit PRPG for address generation has a minor weakness in that it does not include address 00. Designers can ignore this shortcoming or do a separate boundary-scan procedure for address 00 in each memory block tested. The Binary Count-up algorithm accesses every address in the memory block.

One type of fault not fully covered by PRPG, or by one pass of any data pattern, is a single stuck bit in a memory cell. To guarantee detection of this fault, the test must write and read back a data pattern and its complement on two successive BIST executions. Only the Toggle algorithm can do this. The other data patterns require many more BIST operations.

Regardless of the address and data patterns used, all BIST memory blocks are read back using 16-bit PSA data compression. If the data patterns created a unique signature for each data byte within all the memory blocks, the tests should detect the general area and/or type of error. For example, if an error occurs in the same byte of every block, a data line probably has a short or an open. If all the bytes in a memory

TABLE 7: PSEUDO-CODE FOR MEMORY READ OPERATIONS

Sequence and type	BA_{08}-BA_{15} U_1	BA_{00}-BA_{07} U_2	BD_{08}-BD_{15} U_3	BD_{00}-BD_{07} U_4	$\overline{\text{Memtest}}$ and BR/W U_6
**Shut off the embedded microprocessor					
1. IR	READBN	READBN	READBN	READBN	READBN
2. DR	$BA_{15}=0$ $BA_{14}=1$ $BA_{08-13}=000000$	LFSR seed	LFSR seed (set direction B → A)	LFSR seed	BR/W=1 $\overline{\text{Memtest}}=0$
3. IR	Bypass	SCANCN	SCANCN	SCANCN	Bypass
4. DR	Bypass 0	PSA/PRPG 11	PSA 10	PSA 10	Bypass 0
5. IR	Extest	Runt	Runt	Runt	Extest
**Hold in 1149.1 Run-Test/Idle state for 255 Test Clocks (TCKs)					
6. IR	READBN	READBN	READBN	READBN	READBN
7. DR	$BA_{15}=0$ $BA_{14}=1$ $BA_{08-13}=000001$		Read signature (and load next LFSR seed)	Read signature	BR/W=1 $\overline{\text{Memtest}}=0$

BOUNDARY-SCAN MEMORY TESTS

block are in error, an address line may have a problem. If only one byte in a block has a error, a memory cell may be bad, especially if complementary data does not cause an error.

Once the general area or type of error is known, the designer can write a simple program using TI's Asset boundary-scan development tools that will read and write to any selected address. Using various addresses and data patterns, this program can help determine the exact location of the problem.

For larger memory arrays, the Widebus interface parts can be used. These parts allow a larger block of data (up to 65,536 addresses) to be tested in one BIST execution. Also, one device can generate or read two data bytes. Although this capability increases test speed, it reduces resolution in locating a particular fault.

The techniques discussed in this article are for testing of static devices (RAM, ROM, EPROM, etc.). However, with some minor changes depending on the type of device being tested, similar methods can be used to test dynamic RAMs. The main consideration is the refresh cycle time required by the DRAM. One alternative is to make the BIST for each memory block short enough that a burst refresh before and after each BIST can keep the memory refreshed. Another technique is to design the address generator part of the BIST circuit so that it is tied to all the DRAM row-address lines. Then the DRAM will be refreshed automatically while the BIST is running.

Jim Coleman, hardware development manager for Texas Instruments Semiconductor Group's Test Technology Center, received a BS in computer engineering from the Rochester Institute of Technology, Rochester, N.Y.

Richard Thorpe, a member of the group technical staff for the Test Technology Center, holds a BS and MS in electrical engineering from the University of Texas at Austin.