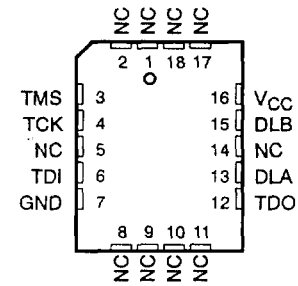
- **Member of Texas Instruments SCOPE™ Family of Testability Products**
- **IEEE 1149.1 Serial Test Bus Compatible**
- **Organization . . . 2048 x 8-Bit Flash Memory**
- **TCK Frequency (V_{CC} ± 10%)**
  '29F816-06 . . . 6.25 MHz
- **5-V Program/Erase/Read Operation**
- **4 Flash-Erasable Blocks (128, 384, 512, and 1024-Byte Size)**
- **Software Sequence Write/Erase Protection**
- **Lockbits**
- **Self-Timed Write/Erase Cycles**
- **Streaming Read/Write Modes**
- **32-Byte Page Programming Mode**
- **CMOS Technology**
- **Single 5-V Power Supply (± 10% Tolerance)**
- **18-Pin Leadless-Ceramic Chip Carrier Package (FG Suffix)**
- **Operating Free-Air Temperature Range – 55°C to 125°C**

FG PACKAGE†
(TOP VIEW)



† Package is shown for pinout reference only.

PIN NOMENCLATURE	
TMS	Test Mode Select
TCK	Test Clock
TDI	Test Data In
TDO	Test Data Out
DLA	Disable Lock A
DLB	Disable Lock B
V_{CC}	5-V Power Supply
GND	Ground
NC	No internal connection

description

The SCOPE Diary is a 16 384-bit, programmable storage device that can be electrically block-erased and reprogrammed. The SCOPE Diary is fabricated using HVCMOS FLOTOX technology for high reliability and very low power dissipation. It performs the erase/program operations automatically with a single 5-V supply voltage, and it can program a single byte or up to 32 bytes in one cycle.

All SCOPE Diary operations are accomplished via a 4-wire Test Access Port (TAP) interface. This interface complies with the IEEE 1149.1 Serial Test Bus standard (JTAG). The interface consists of two control signals: Test Mode Select (TMS) and Test Clock (TCK); and two test data pins: Test Data In (TDI) and Test Data Out (TDO). The JTAG Test Access Protocol defines how this 4-wire test bus is used to scan in instructions and data, execute instructions, and scan out the resulting data.

All test information is serially loaded into the chip via TDI and out of the chip via TDO. Three mandatory JTAG components are added to the Flash EEPROM array: a TAP controller, a set of test data registers, and an instruction register.

The TAP controller interfaces both the test data registers and the instruction register to the 4-wire test bus. The test data registers load and/or capture test data. The instruction register selects the test data register(s) to be accessed and the test to be performed. There are three types of test data registers: the Data Scan Registers (DSR), the Bypass Register (BR), and the Device Identification Register (IDR).

SCOPE is a trademark of Texas Instruments Incorporated.

The SCOPE Diary is divided into four independently flash-erasable blocks. These blocks are configured as 128, 384, 512, and 1024 bytes in size. These blocks can be prevented from being programmed or erased by programming any or all of the four write-once lockbits.

The SCOPE Diary features internal circuitry for self-timed programming, self-timed erasing, and completion polling. In the erased state, all bits are at a logical 1. To reprogram, all memory bits in a selected block are erased first, and then those bits (now logical 1s) are programmed accordingly. The SCOPE Diary supports a page programming mode that allows programming of up to 32 bytes in one cycle. During programming and erasing, the completion status is available, allowing the system to begin a new operation before the maximum specified timeout.

An on-chip power supply reference comparator protects the SCOPE Diary from write and erase commands during power up and power down. During normal operation, software sequences protect against inadvertent program and erase commands.

The SCOPE Diary is offered in an 18-pin leadless ceramic chip carrier package (FG suffix). It is characterized for operation from – 55°C to 125°C.

The SCOPE Diary is available in a 1000-cycle endurance version.

terms

clock

The term *clock* refers to the system test clock used by the controller and its target(s). The clock is input on TCK.

DMA

The SCOPE Diary supports the Direct Memory Access (DMA) extension to the 1149.1 standard. The DMA mode enables a continuous stream of bits to be scanned in or out of the SCOPE Diary.

host

The term *host* refers to the device directing the activity of the SCOPE Diary.

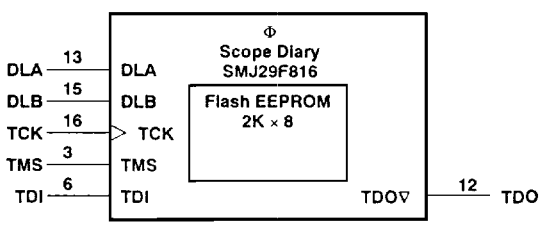
JTAG

The Joint Test Action Group (JTAG) is the originator of IEEE Standard 1149.1.

SCOPE

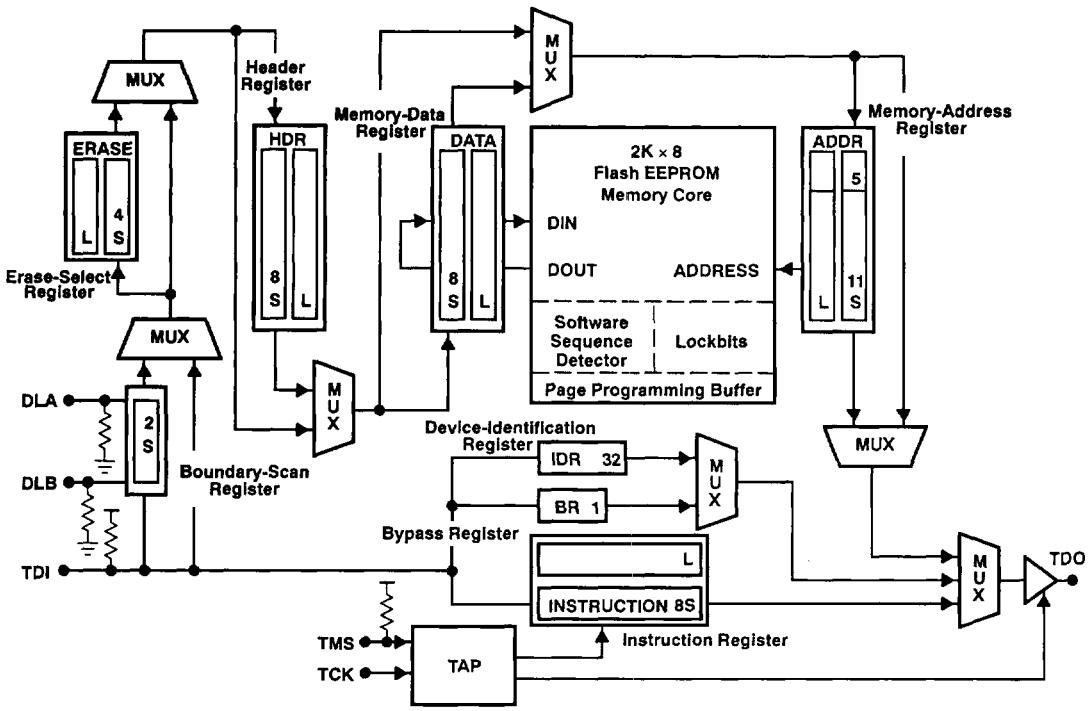
System Controllability and Observability Partitioning Environment (SCOPE) is the family name for Texas Instruments testability products.

logic symbol†



† This symbol is in accordance with ANSI/IEEE Std 91-1984 and IEC Publication 617-12-1991.

functional block diagram



Terminal Functions

Pin Name	Pin #	I/O	Description
TMS	3	I	Test Mode Select. Controls transition of TAP finite state machine. This input is sampled on the rising edge of TCK.
TCK	4	I	Test Clock. Input clock to TAP finite state machine. All changes in state are synchronous to the test clock TCK.
TDI	6	I	Test Data In. Data input to the internal register scan path. Data on this pin is sampled on the rising edge of TCK.
TDO	12	O	Test Data Out. Data output from the internal register scan path. Data is updated on this pin on the falling edge of TCK.
DLA	13	I	Disable Lock A. Controls lockbit functionality for memory array block 0. When DLA = V_{IL}, the state of lockbit 0 (LCK0) determines whether block 0 can be erased or programmed. When DLA = V_{IH}, block 0 can be erased or programmed regardless of the state of lockbit 0. When DLA = V_H ($V_H >> V_{CC}$), the SCOPE Diary enters a special manufacturing test mode.
DLB	15	I	Disable Lock B. Controls lockbit functionality for memory blocks 1, 2, and 3. When DLB = V_{IL}, the states of lockbits 1, 2, and 3 (LCK1, LCK2, LCK3) determine whether their respective blocks (1, 2, and 3) can be erased or programmed. When DLB = V_{IH}, blocks 1, 2, and 3 can be erased or programmed, regardless of the state of their associated lockbits.
V_{CC}	16	I	5-V Power Supply. (± 10% operating power supply connection.)
GND	7	I	Ground reference

Internal registers

Note that the most significant bit is farthest from the output (TDO) in all internal registers.

Instruction

The instruction register is an 8-bit shift register with parallel inputs to monitor the SCOPE Diary status. The most significant bit (7) is a parity bit. The SCOPE Diary status is loaded into the instruction register during the *Capture-IR* controller state (see Figure 1). During the *Shift-IR* state, the status bits are shifted out as a new SCOPE Diary instruction is scanned into the instruction register.

Bit #	7	6	5	4	3	2	1	0
	IRP	P/EBS	SSS	LMS	VMS	P/ECES	0	1
	R-1†	R-0	R-0	R-0	R-0	R-0	R-0	R-1

†R = Read, -n = Value after reset

Bit 0: Always loaded with 1.

Bit 1: Always loaded with 0

Bit 2: **P/ECES – Program/Erase Contention Error Status**
0 = No error detected.
1 = Attempt to write to SCOPE Diary during busy state.

Bit 3: **VMS – Verify Mode Status**
0 = Normal operating mode.
1 = SCOPE Diary is in either program-verify or erase-verify mode. This bit will remain set until exit-verify software sequence is issued.

Bit 4: **LMS – Lock Mode Status**
0 = Normal operating mode.
1 = SCOPE Diary is in lockbit mode.

Bit 5: **SSS – Software Sequence Status**
0 = Normal operating mode.
1 = Valid software sequence detected. The bit will be set within 2 μs after the SCOPE Diary detects a valid software sequence. The bit will remain set until one of the following occurs:
a) The sequence timer expires.
b) The active program or erase cycle is complete.
c) The CLRSWS command is issued.

Bit 6: **P/EBS – Program/Erase Busy Status**
0 = Normal operating mode.
1 = Busy state. The SCOPE Diary is executing a self-timed program or erase operation. The bit will be set within 2 μs after the BEGOPS instruction is executed. This bit will remain set until the operation is complete.

Bit 7: **IRP – Instruction Register Parity**
All valid commands to the instruction register are even parity.
0 = Parity error detected in previously loaded instruction. The SCOPE Diary will automatically place the BYPASS register into the data register scan path.
1 = No parity error in previously loaded instruction.

Figure 1. Instruction Register Status

boundary-scan

The boundary-scan register is a 2-bit register. Bit 0 of this register is connected to DLA; bit 1 is connected to DLB. This register can only be used to sample the connected inputs; therefore, values stored in the boundary-scan register during the *Update-DR* controller state will not be applied to the internal core logic.

device-identification

The device identification register returns the following 32-bit code when interrogated with the IDCODE command: *0000102Fh*. The device ID register is selected into the scan path during power-on reset or upon entering the *Test-Logic-Reset* state.

bypass

The bypass register is a 1-bit register. It allows data to transfer from TDI to TDO in one TCK clock cycle. The bypass register is selected into the scan path when a parity error is detected during the *Shift-IR* state.

memory-data

The memory-data register is an 8-bit register used to load data into the memory array during write operations. This register is also used to sample data from the memory array during read operations. The parallel-scan load path is connected to the memory core data outputs. The output of the register latch is connected to the data input of the memory core. The operation of the register is shown in Table 1.

Table 1. Memory-Data Register Operation

Opcode	Capture-DR	Shift-DR	Update-DR
DMARD	Memory Data to Scan	Data Stream from Array	Scan to Register Latch
DMAWR	Memory Data to Scan	Data Stream to Array	Scan to Register Latch
BYTERD	Memory Data to Scan	Normal Shift Operation	Scan to Register Latch
BYTEWR	Memory Data to Scan	Normal Shift Operation	Scan to Register Latch
ISTEST	Register Latch to Scan	Normal Shift Operation	Scan to Register Latch

memory-address

The memory-address register is a 16-bit register used to address the Flash EEPROM array during read and write operations. Bits 10 – 0 are used to address the Flash memory array. Bits 14 – 0 are used to address the software sequence detector. The operation of the register is shown in Table 2.

Table 2. Memory-Address Register Operation

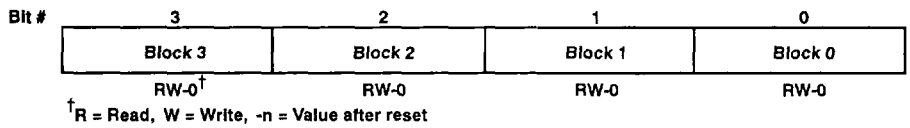
Opcode	Capture-DR	Shift-DR	Update-DR
LDADDR	Register Latch to Scan	Normal Operation	Scan to Register Latch
DMARD	Hold	Auto-Increment	Hold
DMAWR	Hold	Data Stream to Array	Hold
BYTERD	Register Latch to Scan	Normal Operation	Scan to Register Latch
BYTEWR	Register Latch to Scan	Normal Operation	Scan to Register Latch
ISTEST	Register Latch to Scan	Normal Operation	Scan to Register Latch

page programming buffer

The programming pages begin on 32-byte boundaries. Data being written to the SCOPE Diary is stored in the 32-byte page programming buffer until the memory-array programming cycle begins. The page buffer address mechanism does not automatically recognize page programming buffer loads that cross a page boundary. Bits 10 – 5 of the last address presented to the page programming buffer will be used as the page pointer when the memory array programming cycle begins. After an initial data value is loaded into the page programming buffer, all remaining bytes within the page programming buffer are initialized to FFh.

erase-select

The erase-select register is a 4-bit register used to select the Flash memory block(s) that will be erased during an erase cycle. Each bit in the register maps to one of the memory blocks (see Figure 2). To select a block for erasure, set the block's corresponding memory-control bit to logic 1. The operation of the register is shown in Table 3.



†R = Read, W = Write, -n = Value after reset

Bit 0: **Block 0 Erase Enable (address 0000 – 007F)**
0 = Erase disable
1 = Erase enable

Bit 1: **Block 1 Erase Enable (address 0080 – 01FF)**
0 = Erase disable
1 = Erase enable

Bit 2: **Block 2 Erase Enable (address 0200 – 03FF)**
0 = Erase disable
1 = Erase enable

Bit 3: **Block 3 Erase Enable (address 0400 – 07FF)**
0 = Erase disable
1 = Erase enable

Figure 2. Erase-Select Register

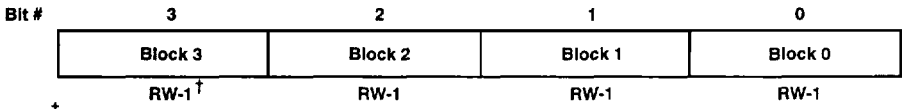
Table 3. Erase-Select Register Operation

Opcode	Capture-DR	Update-DR
ERABLK	Register Latch to Scan	Scan to Register Latch
ISTEST	Register Latch to Scan	Scan to Register Latch

lockbits

The lockbit register contains *four one-time-programmable, non-erasable bits*. The lockbits map one-to-one to the blocks in the array (bit 0 maps to block 0). The lockbit register is not located on the scan path; it is internal to the memory core. It can be accessed using the memory-address and memory-data registers.

To prevent a block from being programmed or erased, program a logic 0 in the block's corresponding bit position. Read and write operations to the lockbits are selected by the SETLOCK instruction. To program the lockbits, execute the DMAWR or BYTEWR instruction sequences while in the lock mode. The lockbit register is shown in Figure 3.



†R = Read, W = Write, -n = Initial value

Bit 0: **Block 0 Lock Enable (address 0000 – 007F)**
0 = Block program and erase disable
1 = Block program and erase enable

Bit 1: **Block 1 Lock Enable (address 0080 – 01FF)**
0 = Block program and erase disable
1 = Block program and erase enable

Bit 2: **Block 2 Lock Enable (address 0200 – 03FF)**
0 = Block program and erase disable
1 = Block program and erase enable

Bit 3: **Block 3 Lock Enable (address 0400 – 07FF)**
0 = Block program and erase disable
1 = Block program and erase enable

Figure 3. Lockbit Register

header

The header register is an 8-bit register used to control the mode of operation during a DMAWR instruction. The register is cleared to zero on power up and upon entering the *Test-Logic-Reset* state. When the register is cleared (all bits to logic 0), the SCOPE Diary uses a state-transition mode to synchronize the DMA write operation. If the register is not cleared, the contents will be used as a shift data input pattern match to synchronize the start of the DMA write operation.

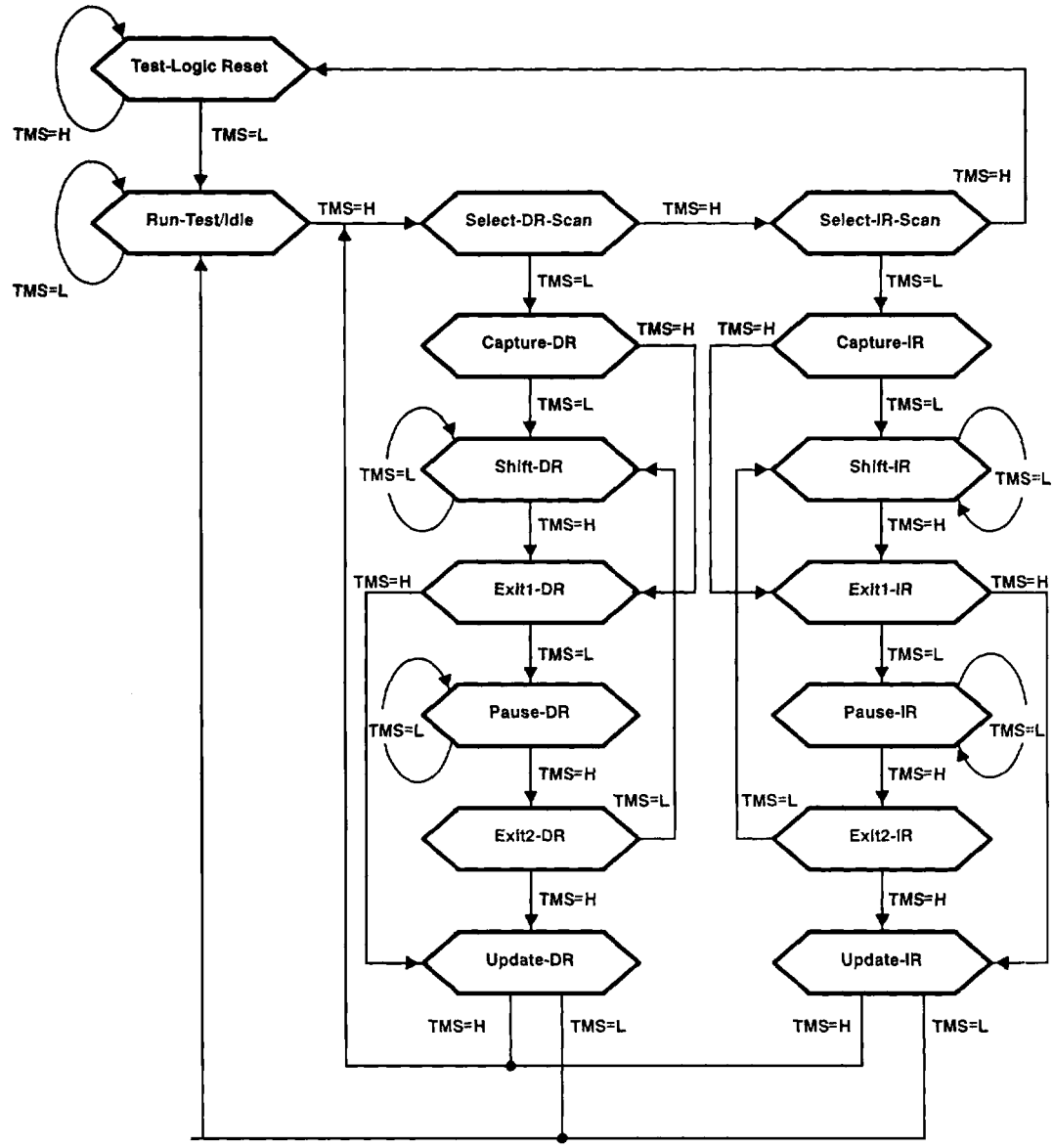


Figure 4. TAP State Diagram

TAP state diagram description (see Figure 4)

The SCOPE Diary TAP controller accepts TCK and TMS signals compatible with IEEE Standard 1149.1. There are six stable states (indicated by a looping arrow) and ten transient states (indicated by two exiting arrows) in the diagram. A stable state is defined as a state the TAP can retain for consecutive TCK cycles. Any other state is a transient state.

There are two main paths through the state diagram; one accesses selected data registers, and one accesses the instruction register.

Test-Logic-Reset

In this state, the test logic is inactive, and an internal reset signal is applied to all registers in the SCOPE Diary. During SCOPE Diary operation, the TAP returns to the *Test-Logic-Reset* state in no more than five TCK cycles if TMS is high. The TMS pin has an internal pullup that forces it to a high level when it is left unconnected or when a board defect causes it to be open-circuited.

Run-Test/Idle

The TAP *must* pass through this state before executing any test operations. The TAP may retain this state indefinitely. No registers are modified while the SCOPE Diary is in the *Run-Test/Idle* state.

Select-DR-Scan, Select-IR-Scan

No specific function is performed in these states. TAP exits them on the next TCK cycle.

Capture-DR

Selected data registers are placed in the scan path (between TDI and TDO). The current instruction determines whether or not the data is loaded or captured into the scan path. The TAP exits the state on the rising edge of TCK.

Shift-DR

In this state, data is shifted serially through the selected data registers, from TDI to TDO, on each TCK cycle. The first shift occurs after the first TCK cycle after entering this state. (No shifting occurs during the TCK cycle in which the TAP changes from *Capture-DR* to *Shift-DR* or from *Exit2-DR* to *Shift-DR*.)

In *Shift-DR*, on the falling edge of TCK, TDO goes from the high-impedance state to the active state. If the TAP has not passed through the *Test-Logic-Reset* state since the last scan operation, TDO enables to the level present before it was last disabled. If the TAP has passed through the *Test-Logic-Reset* state since the last operation, TDO enables to a high level.

Exit1-DR, Exit2-DR

These are temporary states used to end the shifting process. It is possible to return to the *Shift-DR* state from either *Exit1-DR* or *Exit2-DR* without recapturing the data registers. TDO changes from the active state to the high-impedance state on the falling edge of TCK as the TAP changes from *Shift-DR* to *Exit1-DR*.

Pause-DR

The TAP can remain in this state indefinitely. The *Pause-DR* state allows you to suspend and resume shift operations without losing data.

Update-DR

In the *Update-DR* state, the current instruction determines whether or not the latches in the selected data registers are updated with data from the scan path.

TAP state diagram description (continued)

Capture-IR

In the *Capture-IR* state, the instruction register is preloaded with the IR status word, and then it is placed in the scan path. The TAP exits the state on the rising edge of TCK.

Shift-IR

In this state, data is shifted serially through the instruction register, from TDI to TDO, on each TCK cycle. The first shift occurs after the first TCK cycle after entering this state. (No shifting occurs during the TCK cycle in which the TAP changes from *Capture-IR* to *Shift-IR* or from *Exit2-IR* to *Shift-IR*.) In *Shift-IR*, on the falling edge of TCK, TDO goes from the high-impedance state to the active state.

Exit1-IR, Exit2-IR

These are temporary states used to end the shifting process. It is possible to return to the *Shift-IR* state from either *Exit1-IR* or *Exit2-IR* without recapturing the instruction register. TDO changes from the active state to the high-impedance state on the falling edge of TCK as the TAP changes from *Shift-IR* to *Exit1-IR*.

Pause-IR

The TAP can remain in this state indefinitely. The *Pause-IR* state allows you to suspend and resume shift operations without losing data.

Update-IR

In the *Update-IR* state, the instruction register latches are updated with the new instruction from the scan path.

instructions

standard SCOPE instructions

The SCOPE Diary supports a subset of the standard SCOPE instruction set. The defined instructions are shown in Table 4. All other SCOPE instructions select the default BYPASS instruction.

Table 4. Standard SCOPE Instructions

Opcode	Code	Description
BYPASS	FFh	Select Bypass Register
EXTEST	00h	External Boundary Test (see Note 1)
IDCODE	81h	ID Register Scan
SAMPLE	82h	Boundary Sample

NOTE 1: During operation, the EXTEST instruction behaves identically to the SAMPLE instruction.

SCOPE Diary-specific instructions

The SCOPE Diary supports specific instructions to control the operation of the Flash EEPROM array. The defined instructions are shown in Table 5. All undefined opcodes select the BYPASS instruction.

Table 5. SCOPE Diary-Specific Instructions

Opcode	Code	Description
BEGOPS	69h	Begin Operation in Progress
BYTERD	63h	Byte Read
BYTEWR	E4h	Byte Write
CLRERR	6Ah	Clear Conflict Error Flag
CLRLOCK	66h	Exit Lock Mode
CLRSWS	EBh	Clear Software Sequence
DMARD	E1h	DMA Read
DMAWR	E2h	DMA Write
ERABLK	E7h	Erase Block Register Select
ISTEST	6Ch	Internal Self Test
LDADDR	60h	Load Address Register
LOADHDR	E8h	Header Register Select
SETLOCK	65h	Enter Lock Mode

BEGOPS Begin Operation in Progress

Scan Path TDI → bypass → TDO

Description The BEGOPS instruction is used to initiate a program, erase, or verify mode operation after the appropriate software sequence has been issued. This instruction must be executed within 6 ms of the last write operation, and the software sequence status bit in the instruction register must be set, or the selected operation will not begin. If the time-out condition is not met, the software sequence commands must be re-issued. Once the BEGOPS instruction is loaded, it is not executed until the diary is placed in the *Run-Test/Idle* state.

BYPASS Select Bypass Register

Scan Path TDI → bypass → TDO

Description The BYPASS instruction conforms to the 1149.1 BYPASS instruction. The one-bit bypass register is selected in the scan path. A logic 0 is loaded in the bypass register during the *Update-DR* state.

BYTERD Byte Read

Scan Path TDI → memory-data → memory-address → TDO

Description The BYTERD instruction is used to read the value stored in a memory array location. During the read operation, the contents of the memory-address register point to the value. This value is captured in the memory-data register during the *Update-DR* state.

BYTEWR Byte Write

Scan Path TDI → memory-data → memory-address → TDO

Description The BYTEWR instruction performs two operations. It can write 8-bit values into both the software sequence detector and the page programming buffer. The contents of the memory-address register and the contents of the memory-data register are presented to the memory core during the *Update-DR* state. On the rising edge of TCK, upon leaving the *Update-DR* state, an internal write signal is applied to either the software sequence detector or the page programming buffer.

CLRERR Clear Conflict Error Flag

Scan Path TDI → bypass → TDO

Description The CLRERR instruction is used to reset the program/erase conflict flag. The conflict flag (status bit 2 in the instruction register) will be set if any write operations are issued while the SCOPE Diary is programming or erasing. After the conflict flag is set, the SCOPE Diary won't recognize any sequence commands. The conflict flag will remain set until the CLRERR instruction is executed.

CLRLOCK Exit Lock Mode

Scan Path TDI → bypass → TDO

Description The CLRLOCK instruction is used to exit the lock mode. When the lock mode is disabled, all read and programming operations are directed to the memory array. The normal mode is indicated when status bit 4 is cleared in the instruction register.

CLRSWS Clear Software Sequence

Scan Path TDI → bypass → TDO

Description The CLRSWS instruction is used to clear software sequence operations. The instruction will reset or cancel any software sequence up until the BEGOPS instruction is executed. The CLRSWS instruction will also clear status bit 5 (valid software sequence detected) in the instruction register. The CLRSWS instruction will not interrupt an erase or program operation once the operation has started.

DMARD DMA Read

Scan Path TDI → (ignored) / memory-data → TDO

Description The DMARD instruction is used to perform streaming data reads from the Flash EEPROM memory array. During the read operation, upon entering the *Shift-DR* state, the contents of the memory array will be shifted out beginning with the currently addressed location. The memory-address register is automatically incremented on each byte boundary while performing the DMARD operation. Input data on the TDI pin is discarded and does not pass through to the TDO output pin.

DMAWR DMA Write

Scan Path TDI → memory-data → memory-address → TDO

Description The DMAWR instruction allows a streaming method of writing address/data pairs to the SCOPE Diary. During the *Shift-DR* state, the SCOPE Diary will automatically generate write strobes to the memory core on each 24-bit address/data pair boundary. The SCOPE Diary supports two modes of synchronizing the write operation with the incoming address/data pairs; state-transition mode and stream-header mode. The contents of the header register determine the selected mode.

ERABLK Erase Block Register Select

Scan Path TDI → erase-block → TDO

Description The ERABLK instruction is used to access the erase-block select register. Data loaded into the ERABLK register is presented to the memory core during the *Update-DR* state.

EXTEST External Boundary Test

Scan Test TDI → boundary-scan → TDO

Description The EXTEST instruction is used to check the board connectivity of the DLA and DLB input pins. During an EXTEST operation, DLA and DLB inputs to the internal control logic can be sampled by the scan path, but not driven.

IDCODE ID Register Scan

Scan Path TDI → id → TDO

Description The IDCODE instruction is used to read the device identification data. During the *Capture-DR* state, the 32-bit device identification code (0000102Fh) is loaded into the ID register. The IDCODE instruction is automatically loaded during SCOPE Diary power-on reset or upon entry to the *Test-Logic-Reset* state.

ISTEST Internal Self Test

Scan Path TDI → boundary-scan → erase-block → header → memory-data → memory-address → TDO

Description The ISTEST instruction is used to test scan path data registers. During the *Capture-DR* state, all of the register latched values are transferred to the scan path (except the boundary scan register which transfers the values of DLA and DLB to the scan path).

LDADDR Load Address Register

Scan Path TDI → memory-address → TDO

Description The LDADDR instruction is used to load the memory-address register. The 16-bit value loaded from the scan path points to an address and is presented to the memory array during the *Update-DR* state.

LOADHDR Header Register Select

Scan Path TDI → header → TDO

Description The LOADHDR instruction is used to access the header register. Loading any value from 01h to FFh selects header mode synchronization during DMA write operations. Loading the header register with 00h selects state-transition mode synchronization for DMA write operations. During the LOADHDR operation, the header register is selected into the DR scan path.

SAMPLE Boundary Sample

Scan Path TDI → boundary-scan → TDO

Description The SAMPLE instruction is used to check the board connectivity of the DLA and DLB input pins. During a SAMPLE operation, DLA and DLB inputs to the internal control logic can be sampled by the scan path, but not driven.

SETLOCK Enter Lock Mode

Scan Path TDI → memory-data → TDO

Description The SETLOCK instruction is used to enable the lock mode. When the lock mode is enabled, read and programming operations are directed to the lockbits. The lock mode operation is indicated when status bit 4 is set in the instruction register. The SCOPE Diary will remain in the lock mode until the CLRLOCK instruction is executed. While in the lock mode, all read operations capture the state of the lockbits in the data-memory register. While reading the lockbits, the four most significant bits are set to logic 1.

operation

TAP state controller

Operation of the TAP state controller conforms to the IEEE 1149.1 Serial Test Bus standard. The state flow diagram is shown in Figure 4 on page 8.

loading and executing instructions

All bus sequences that load and execute instructions start with the TAP in the *Run-Test/Idle* state. To initialize the TAP to *Run-Test/Idle* from any other state, apply the 6-cycle sequence shown in Table 6.

Table 6. TAP Reset Sequence

Cycle	1	2	3	4	5	6
TMS	1	1	1	1	1	0
TCK						
TDI†	X	X	X	X	X	X
TDO	(See Note 2)	HI-Z	HI-Z	HI-Z	HI-Z	HI-Z
TAP State	Undefined	Undefined	Undefined	Undefined	Test-Logic-Reset	Run-Test/Idle

† X denotes a don't care.

NOTE 2: TDO will become high-impedance on falling edge of TCK.

sequence timing

The SCOPE Diary contains internal timing logic to simplify programming and erase operations. Once the host initiates a programming or erasing operation, that operation will automatically continue to completion. The host does not need to intervene until the operation is finished. To check the status of the operation, poll status bit 6 of the instruction register.

software sequence

The host initiates all of the SCOPE Diary's internal memory operations by issuing a sequence of address/data pairs (forming a specific software sequence) to the SCOPE Diary. The correct address/data pairs must be received in a specific order and within a specific time period to be recognized as a valid software sequence by the SCOPE Diary. Once a sequence has begun, the SCOPE Diary starts an internal sequence timer. Each consecutive address/data pair must be received within a 6 ms time period. After each address/data pair, the timer is reset to receive the next sequence pair. If the time between consecutive address/data pairs exceeds the timer limit, the internal state of the sequence detector will be reset, and the host must re-issue the software sequence from the beginning. If the SCOPE Diary detects a valid software sequence, status bit 5 of the instruction register will be set within 2 μs and will remain set as long as the SCOPE Diary is unlocked for the operation. The host may terminate a software sequence at any point by either letting the internal time limit expire, or by issuing a CLRSWS command. The software sequences recognized by the SCOPE Diary are shown in Table 7.

Table 7. SCOPE Diary Software Sequences

Operation	Address/Data Pair Sequence
Programming	5555h / AAh 2AAAh / 55h 5555h / A0h
Erasing	5555h / AAh 2AAAh / 55h 5555h / 80h 5555h / AAh 2AAAh / 55h 5555h / 10h
Program-Verify	5555h / AAh 2AAAh / 55h 5555h / B0h
Erase-Verify	5555h / AAh 2AAAh / 55h 5555h / D0h
Exit-Verify	5555h / AAh 2AAAh / 55h 5555h / F0h

page programming buffer

The page programming buffer is a 32-byte buffer that the host loads with the data to be programmed into the memory array. This buffer is internal to memory and can be accessed using the memory-address and memory-data registers. The page programming buffer is automatically selected by internal control logic after it detects a valid program software sequence. The contents of this buffer are automatically set to FFh, so any bits not specifically cleared by the host will not be programmed. Up to 32 bytes can be programmed in one cycle.

Address/data pairs must be loaded into the page programming buffer within the same time constraints as the software sequence. If the sequence timer is allowed to expire during a page programming buffer load, the internal control logic will terminate the programming operation and clear the software sequence detector (indicated by status bit 5 in the instruction register). During a programming operation, data that has been loaded into the internal page programming buffer is automatically transferred into the memory array.

operation initiation

The SCOPE Diary differs from typical software sequence-controlled memory devices because the selected programming or erasing operation does not automatically begin at the end of the internal sequence time out. To initiate the selected operation, the host must issue the BEGOPS command to the SCOPE Diary and enter the *Run-Test/Idle* state before the internal sequence timer expires. If the timer expires, the internal sequence detector will be cleared, and the selected operation must be re-initiated from the beginning. Status bit 6 in the instruction register indicates a successful program or erase operation. This bit will be set within 2 μs after the BEGOPS instruction is executed.

reset

The SCOPE Diary test bus logic is cleared either by internal circuitry at power-up, or by entry to the *Test-Logic-Reset* state. All internal data scan path registers are set to logic 0, and the instruction register is loaded with the IDCODE instruction. Entering the *Test-Logic-Reset* state will not clear a pending software sequence or interrupt an executing self-timed program or erase cycle.

erase-verify mode

The erase-verify mode allows the host to verify the adequacy of erasure. Once the SCOPE Diary has been placed in the verify mode, it will remain in that state (indicated in the instruction register when status bit 3 is a logical 1) until the exit-verify mode sequence has been issued. When in the erase-verify mode, the internal voltage applied to the read select lines (wordlines) is reduced by a preset margin. To verify that the array has been erased, the host reads the memory block and checks that all bits are set to logic 1.

program-verify mode

The program-verify mode allows the host to verify the adequacy of programming. Once the SCOPE Diary has been placed in the program-verify mode, it will remain in this state (indicated in the instruction register when status bit 3 is a logical 1) until the exit-verify mode sequence has been issued. When in the program-verify mode, the internal voltage applied to the read-select lines (wordlines) is increased by a preset margin. To verify that a programming operation was successful, the host reads the previously programmed locations and checks that the data values are correct.

JTAG extensions

DMA read

The DMA read mode allows any number of sequential bits to be read from the SCOPE Diary while remaining in the *Shift-DR* state. During a DMA read operation, the contents of the memory array will be shifted out beginning with the address location contained in the memory-address register. Upon entry to the *Shift-DR* state, an internal modulo 8 counter is triggered. This counter is used to increment the contents of the memory-address register on byte boundaries. After the data from the last byte in the memory array has been read, the next data will be read from the byte at the beginning of the memory array.

DMA write

The DMA write mode simplifies data transfer to the SCOPE Diary. This mode allows data to be continuously streamed into the SCOPE Diary while remaining in the *Shift-DR* state. Compared to normal modes of data transfer, the DMA write extensions enable systems with a large number of devices in the scan path to realize a significant reduction of clock cycles.

In the DMA write mode, an internal modulo 24 counter is used to automatically transfer address/data pairs to the memory core while bypassing the *Update-DR* state. To initiate a DMA write data transfer, the internal modulo 24 counter must be triggered (synchronized) when the first bit of an address/data pair is at the TDI input pin. The SCOPE Diary supports two methods of DMA synchronization: state-transition mode and header mode. The host determines which method of DMA synchronization is used.

state-transition mode

The host selects state-transition mode by clearing the header register (all bits to logic 0). When the state-transition mode is selected, incoming scan path data is ignored during first entry to the *Shift-DR* state. The first entry to *Pause-DR* indicates proper alignment at the TDI input pin of the first address/data pair. Re-entry to the *Shift-DR* state triggers the modulo 24 counter and enables the address/data pair to be written to the memory core. Address/data pairs can then be streamed continuously to the SCOPE Diary with internal transfers occurring automatically on 24-bit boundaries.

header mode

The host selects the header mode by loading the header register with a value from 01h to FFh. When the header mode is selected, incoming scan path data is ignored until a byte (matching the contents of the header register) arrives indicating the arrival of valid address/data pairs. When this header byte is detected, the internal modulo 24 counter is triggered. Address/data pairs can then be streamed continuously to the SCOPE Diary with internal transfers occurring automatically on 24-bit boundaries.

In either state-transition or header mode, the host places the SCOPE Diary in the *Update-DR* state to end a DMA write operation. Because placing the SCOPE Diary in the *Update-DR* state ends the operation, the host must *never* place the SCOPE Diary in this state until the DMA write operation is complete. The host may place the SCOPE Diary in the *Pause-DR* state at any time.

operation examples

Note that in this section, the letter "n" denotes a value from 0h to Fh, and the letter "x" denotes a don't care.

reading examples

reading using the byte mode

Step 1. Load the BYTERD instruction.
Step 2. Scan in 16-bit address = *nnnn* and 8-bit data = *xx*.
Step 3. Scan out 16-bit address = *nnnn* and 8-bit data = *nn*.

reading using the DMA mode

Step 1. Load the LDADDR instruction.
Step 2. Scan in 16-bit address = *nnnn*.
Step 3. Load the DMARD instruction.
Step 4. Loop in *Shift-DR* to shift out a stream of 8-bit memory data values, the address register is automatically incremented on byte boundaries.

lockbit examples

reading lockbits using the byte mode

Step 1. Load the SETLOCK instruction.
Step 2. Load the BYTERD instruction.
Step 3. Scan in 16-bit address = 0000 and 8-bit data = *xx*.
Step 4. Scan out 16-bit address = 0000 and 8-bit data = F*n*.
Step 5. Load the CLRLOCK instruction.

reading lockbits using the DMA mode

Step 1. Load the SETLOCK instruction.
Step 2. Load the DMARD instruction.
Step 3. Scan out the 8-bit lock value = F*n*.
Step 4. Load the CLRLOCK instruction.

programming lockbits using the byte mode

Step 1. Load the SETLOCK instruction.
Step 2. Load the BYTEWR instruction.
Step 3. Scan in address = 5555 and data = AA, go to *Run-Test/Idle*.
Step 4. Scan in address = 2AAA and data = 55; go to *Run-Test/Idle*.
Step 5. Scan in address 5555 and data = A0; go to *Run-Test/Idle*.
Step 6. Scan in address = 0000, and data = F*n*; go to *Run-Test/Idle*.
Step 7. Load the BEGOPS instruction; go to *Run-Test/Idle*.
Step 8. Load the CLRLOCK instruction.

programming lockbits using the DMA mode

Step 1. Load the SETLOCK instruction.
Step 2. Load the DMAWR instruction.
Step 3. Synchronize SCOPE Diary using either state-transition mode or header mode.
Step 4. Loop in *Shift-DR* to scan in address = 5555 and data = AA.
Step 5. Continue looping in *Shift-DR* to scan in address = 2AAA and data = 55.
Step 6. Continue looping in *Shift-DR* to scan in address = 5555 and data = A0.
Step 7. Continue looping in *Shift-DR* to scan in address = 0000 and data = F*n*.
Step 8. Load the BEGOPS instruction; go to *Run-Test/Idle*.
Step 9. Load the CLRLOCK instruction.

flash erase examples

erasing a block using the byte mode

Step 1. Load the ERABLK instruction.
Step 2. Scan in the 4-bit erase-block-select value = *n*.
Step 3. Load the BYTEWR instruction.
Step 4. Scan in address = 5555 and data = AA; go to *Run-Test/Idle*.
Step 5. Scan in address = 2AAA and data = 55; go to *Run-Test/Idle*.
Step 6. Scan in address = 5555 and data = 80; go to *Run-Test/Idle*.
Step 7. Scan in address = 5555 and data = AA; go to *Run-Test/Idle*.
Step 8. Scan in address = 2AAA and data = 55; go to *Run-Test/Idle*.
Step 9. Scan in address = 5555 and data = 10; go to *Run-Test/Idle*.
Step 10. Poll the SCOPE Diary until valid sequence is detected.
Step 11. Load the BEGOPS instruction; go to *Run-Test/Idle*.

erasing a block using the DMA mode

Step 1. Load the ERABLK instruction.
Step 2. Scan in the 4-bit erase-block-select value = *n*.
Step 3. Load the DMAWR instruction.
Step 4. Synchronize the SCOPE Diary using either state-transition mode or header mode.
Step 5. Loop in *Shift-DR* to scan in address = 5555 and data = AA.
Step 6. Continue looping in *Shift-DR* to scan in address = 2AAA and data = 55.
Step 7. Continue looping in *Shift-DR* to scan in address = 5555 and data = 80.
Step 8. Continue looping in *Shift-DR* to scan in address = 5555 and data = AA.
Step 9. Continue looping in *Shift-DR* to scan in address = 2AAA and data = 55.
Step 10. Continue looping in *Shift-DR* to scan in address = 5555 and data = 10.
Step 11. Poll SCOPE Diary until valid sequence is detected.
Step 12. Load the BEGOPS instruction; go to *Run-Test/Idle*.

verifying block erasure using the byte mode

select the erase-verify mode:

Step 1. Load the BYTEWR instruction.
Step 2. Scan in address = 5555 and data = AA; go to *Run-Test/Idle*.
Step 3. Scan in address = 2AAA and data = 55; go to *Run-Test/Idle*.
Step 4. Scan in address = 5555 and data = D0; go to *Run-Test/Idle*.

read out the erased block:

Step 5. Load the BYTERD instruction.
Step 6. Scan in 16-bit address = *nnnn* and 8-bit data = *xx*.
Step 7. Scan out 16-bit address = *nnnn* and 8-bit data = FF; at the same time, scan in 16-bit address = *nnnn*+1 and data = *xx*.
Step 8. Repeat Step 7 until entire block is read. All bits will be a logic 1 if the block is properly erased.

exit the erase-verify mode:

Step 9. Load the BYTEWR instruction.
Step 10. Scan in address = 5555 and data = AA; go to *Run-Test/Idle*.
Step 11. Scan in address = 2AAA and data = 55; go to *Run-Test/Idle*.
Step 12. Scan in address = 5555 and data = F0; go to *Run-Test/Idle*.

verifying block erasure using the DMA mode

select the erase-verify mode:

Step 1. Load the DMAWR instruction.
Step 2. Synchronize the SCOPE Diary using either state-transition mode or header mode.
Step 3. Loop in *Shift-DR* to scan in address = 5555 and data = AA.
Step 4. Continue looping in *Shift-DR* to scan in address = 2AAA and data = 55.
Step 5. Continue looping in *Shift-DR* to scan in address = 5555 and data = D0.

read out the erased block:

Step 6. Load the LDADDR instruction.
Step 7. Scan in 16-bit data starting address = *nnnn* of the block you want to verify.
Step 8. Load the DMARD instruction.
Step 9. Loop in *Shift-DR* to shift out a stream of 8-bit memory data values from the addressed block. All bits will be a logic 1 if the block is properly erased.

exit the erase-verify mode:

Step 10. Load the DMAWR instruction.
Step 11. Synchronize the SCOPE Diary using either state-transition mode or header mode.
Step 12. Loop in *Shift-DR* to scan in address = 5555 and data = AA.
Step 13. Continue looping in *Shift-DR* to scan in address = 2AAA and data = 55.
Step 14. Continue looping in *Shift-DR* to scan in address = 5555 and data = F0.

verifying programming using the byte mode

select the program-verify mode:

Step 1. Load the BYTEWR instruction.
Step 2. Scan in address = 5555 and data = AA; go to *Run-Test/Idle*.
Step 3. Scan in address = 2AAA and data = 55; go to *Run-Test/Idle*.
Step 4. Scan in address = 5555 and data = B0; go to *Run-Test/Idle*.

read out the programmed data:

Step 5. Load the BYTERD instruction.
Step 6. Scan in 16-bit address = *nnnn* and 8-bit data = *xx*.
Step 7. Scan out 16-bit address = *nnnn* + 1 and 8-bit data = *nn*; at the same time, scan in 16-bit address=*nnnn* + 1 and data = *xx*.
Step 8. Repeat Step 7 until desired memory locations are read and verified.

exit the program-verify mode

Step 9. Load the BYTEWR instruction.
Step 10. Scan in address = 5555 and data = AA; go to *Run-Test/Idle*.
Step 11. Scan in address = 2AAA and data = 55; go to *Run-Test/Idle*.
Step 12. Scan in address = 5555 and data = F0; go to *Run-Test/Idle*.

verifying programming using the DMA mode

select the program-verify mode:

Step 1. Load the DMAWR instruction.
Step 2. Synchronize the SCOPE Diary using either state-transition mode or header mode.
Step 3. Loop in *Shift-DR* to scan in address = 5555 and data = AA.
Step 4. Continue looping in *Shift-DR* to scan in address = 2AAA and data = 55.
Step 5. Continue looping in *Shift-DR* to scan in address = 5555 and data = B0.

read out the programmed data:

Step 6. Load the LDADDR instruction.
Step 7. Scan in 16-bit starting address = *nnnn* of the data you want to verify.
Step 8. Load the DMARD instruction.
Step 9. Loop in *Shift-DR* to shift out a stream of 8-bit memory data values starting from the addressed location. Verify that the output data stream matches the programmed data.

exit the program-verify mode:

Step 10. Load the DMAWR instruction.
Step 11. Synchronize the SCOPE Diary using either state-transition mode or header mode.
Step 12. Loop in *Shift-DR* to scan in address = 5555 and data = AA.
Step 13. Continue looping in *Shift-DR* to scan in address = 2AAA and data = 55.
Step 14. Continue looping in *Shift-DR* to scan in address = 5555 and data = F0.

programming examples

programming a single byte using the byte mode

Step 1. Load the BYTEWR instruction.
Step 2. Scan in address = 5555 and data = AA; go to *Run-Test/Idle*.
Step 3. Scan in address = 2AAA and data = 55; go to *Run-Test/Idle*.
Step 4. Scan in address = 5555 and data = A0; go to *Run-Test/Idle*.
Step 5. Scan in address = *nnnn* and data = *nn*; go to *Run-Test/Idle*.
Step 6. Load the BEGOPS instruction; go to *Run-Test/Idle*.

programming a single byte using the DMA mode

Step 7. Load the DMAWR instruction.
Step 8. Synchronize the SCOPE Diary using either state-transition mode or header mode.
Step 9. Loop in *Shift-DR* to scan in address = 5555 and data = AA.
Step 10. Continue looping in *Shift-DR* to scan in address = 2AAA and data = 55.
Step 11. Continue looping in *Shift-DR* to scan in address = 5555 and data = A0.
Step 12. Continue looping in *Shift-DR* to scan in address = *nnnn* and data = *nn*.
Step 13. Load the BEGOPS instruction; go to *Run-Test/Idle*.

programming a page using the byte mode

Step 1. Load the BYTEWR instruction.
Step 2. Scan in address = 5555 and data = AA; go to *Run-Test/Idle*.
Step 3. Scan in address = 2AAA and data = 55; go to *Run-Test/Idle*.
Step 4. Scan in address = 5555 and data = A0; go to *Run-Test/Idle*.
Step 5. Scan in address = *nnnn* and data = *nn*; go to *Run-Test/Idle*.
Step 6. Go to Step 5 while there are address/data pairs to load within the 32-byte page.
Step 7. Load the BEGOPS instruction, go to *Run-Test/Idle*.

programming a page using the DMA mode

Step 1. Load the DMAWR instruction.
Step 2. Synchronize the SCOPE Diary using either state-transition mode or header mode.
Step 3. Loop in *Shift-DR* to scan in address = 5555 and data = AA.
Step 4. Continue looping in *Shift-DR* to scan in address = 2AAA and data = 55.
Step 5. Continue looping in *Shift-DR* to scan in address = 5555 and data = A0.
Step 6. Continue looping in *Shift-DR* to scan in address = *nnnn* and data = *nn*.
Step 7. Go to Step 6 while there are address/data pairs to load within the 32-byte page.
Step 8. Load the BEGOPS instruction; go to *Run-Test/Idle*..

absolute maximum ratings over operating free-air temperature range (unless otherwise noted)†

Supply voltage, V_{CC} (see Note 3) ..... – 0.6 V to 7 V
Input voltage range: All except DLA (see Note 3) ..... – 0.6 V to 6.5V
Input voltage range: DLA (see Note 3) ..... – 0.6 V to 15 V
Output voltage (see Note 3) ..... – 0.6 V to V_{CC} + 0.6V
Operating free-air temperature range ..... – 55°C to 125°C
Storage temperature range ..... – 65°C to 125°C

† Stresses beyond those listed under "absolute maximum ratings" may cause permanent damage to the device. These are stress ratings only and functional operation of the device at these or any other conditions beyond those indicated under "recommended operating conditions" is not implied. Exposure to absolute-maximum-rated conditions for extended periods may affect device reliability.

NOTE 3: Voltage values are with respect to GND (substrate).

recommended operating conditions

			MIN	NOM	MAX	UNIT
V_{CC}	Supply voltage		4.5	5	5.5	V
V_{IH}	High-level input voltage	TTL	2		V_{CC} + 1	V
		CMOS	V_{CC} – 0.2		V_{CC} + 0.2	
V_{IL}	Low-level input voltage	TTL	– 0.5		0.8	V
		CMOS	GND – 0.2		GND + 0.2	
T_A	Operating free-air temperature		– 55		125	°C
	Endurance cycles				1000	
	Programming operations				100 000	

electrical characteristics over full ranges of recommended operating conditions (unless otherwise noted)

PARAMETER			TEST CONDITIONS	MIN	TYP†	MAX	UNIT
V_{OH}	High-level output voltage		I_{OH} = – 2 mA	2.4			V
V_{OL}	Low-level output voltage		I_{OL} = 2.1 mA			0.4	V
I_I	Input current (leakage)	DLA, DLB	V_I = 2.4 V		75	165	µA
		DLA, DLB	V_I = 0 V			±10	
		TDI, TMS, TCK	V_I = 0.4		– 10	– 50	
		TDI, TMS, TCK	$V_I = V_{CC}$ = 5.5 V			±10	
I_O	Output current (leakage)		V_O = 0.1 to V_{CC}			±10	µA
I_{CC1}	V_{CC} average supply current (active read)		t_{cycle} = 160 ns, outputs open			20	mA
I_{CC2}	V_{CC} average supply current (active write)		t_{cycle} = 15 ms			15	mA

† Typical values are at T_A = 25°C and nominal voltages.

capacitance over recommended ranges of supply voltage and operating free-air temperature, f = 1 MHz‡

PARAMETER		TEST CONDITIONS	MIN	TYP†	MAX	UNIT
C_I	Input capacitance	V_I = 0, f = 1 MHz		4	7	pF
C_O	Output capacitance	V_O = 0, f = 1 MHz		8	12	pF

† Typical values are at T_A = 25°C and nominal voltage.
‡ Capacitance measurements are made on sample basis only

switching characteristics over full ranges of recommended operating conditions

PARAMETER		MIN	MAX	UNIT
t_{DA}	TDO valid from falling edge of TCK		74	ns
t_{DZ}	TDO disable time from falling edge of TCK		45	ns

timing requirements over recommended ranges of supply voltage and operating free-air temperature

		MIN	MAX	UNIT
t_{CYC}	TCK cycle time	160		ns
$t_{W(TCKH)}$	Pulse duration, TCK high	50		ns
$t_{W(TCKL)}$	Pulse duration, TCK low	70		ns
$t_{SU(TMS)}$	TMS input setup time	15		ns
$t_{IH(TMS)}$	TMS input hold time	5		ns
$t_{SU(TDI)}$	TDI input setup time	6		ns
$t_{IH(TDI)}$	TDI input hold time	15		ns

internal timing requirements

	PARAMETER	MIN	MAX	UNIT
t_{SSS}	Software sequence status bit valid from software sequence		2	μs
t_{PEBS}	Program erase busy status bit valid from BEGOPS execution		2	μs
t_{ST}	Sequence timer limit		6	ms
t_{ERA}	Erase cycle time		15	ms
t_{PGM}	Program cycle time		15	ms

PARAMETER MEASUREMENT INFORMATION

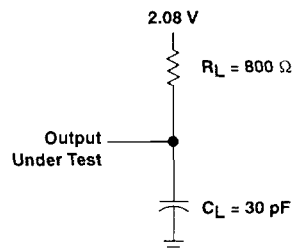
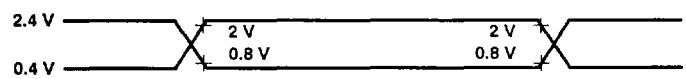
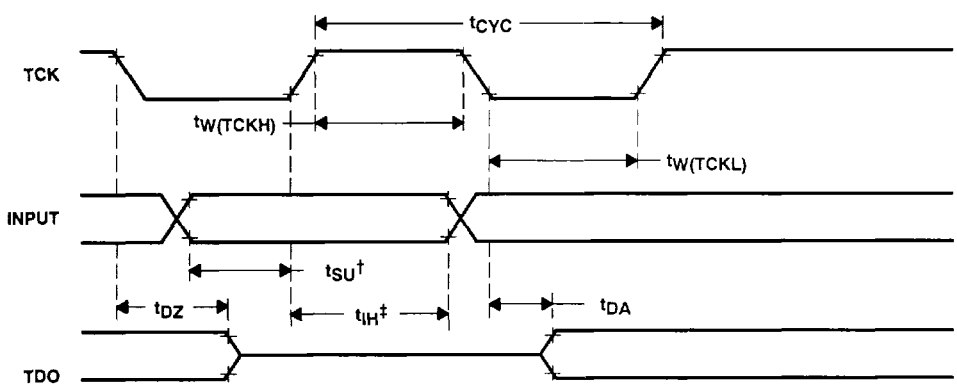


Figure 5. AC Test Output Load Circuit

AC testing input/output wave forms



AC testing inputs are driven at 2.4 V for logic high and 0.4 for logic low. Timing measurements are made at 2 V for logic 1 and 0.8 V for logic 0 for both inputs and outputs. Each device should have a 0.1 μF ceramic capacitor connected between V_{CC} and GND as close as possible to the device pins.



† t_{SU} represents TDI input setup time and TMS input setup time.
‡ t_{IH} represents TDI input hold time and TMS input hold time