SEMI E30.3-0698
TESTING EQUIPMENT SPECIFIC EQUIPMENT MODEL (TSEM)

CONTENTS

List of Tables

SEMI E30.3-0698
TESTING EQUIPMENT SPECIFIC EQUIPMENT MODEL (TSEM)

1 Purpose

1.1 This document establishes a Specific Equipment Model for testing equipment (TSEM). The TSEM consists of equipment characteristics and behaviors that apply to this class of equipment and are required to be implemented in addition to the fundamental requirements and additional capabilities specified in the Generic Equipment Model (GEM), SEMI E30. The intent of this document is to facilitate the integration of testing equipment into an automated semiconductor factory. This document accomplishes this by defining an operational model for testing equipment as viewed by a factory automation controller. This definition provides a standard host interface and equipment operational behavior.

2 Scope

2.1 The scope of this document is limited to the definition of testing equipment behavior as perceived by a Semiconductor Equipment Communications Standard (SEMI E5) host that complies with SEMI E30. The document defines the view of the equipment through the SECS communications link but does not define the internal operation of the equipment. It includes a specific processing state model as the basis for the behavior of all equipment of this class.

2.2 This document requires that the SEMI E30 fundamental requirements and applicable additional capabilities have been implemented on the test equipment. This document expands SEMI E30 Standard requirements and capabilities in the areas of the processing state model, collection events, alarm documentation, remote commands, variable items, and process program management.

3 Limitations

3.1 *Communications*

3.1.1 It is required that any TSEM-compliant equipment follow the Communications State Model in SEMI E30. In addition, TSEM-compliant equipment shall support the High-Speed Messaging Service (SEMI E37) Communication Standard sending SEMI E5 messages over TCP/IP to maximize the amount of data available for monitoring from this class of equipment. This specification deals only with the behavior of the tester in communicating with the host. It is recognized that the tester may also have a communications link with a test handler or prober and that the handler and/or prober may also have a communications link with the host. This specification is intentionally non-specific on the communications link requirements between handler and tester to allow the user the greatest amount of flexibility in specific factory configurations.

3.2 *Multi-Head and Multi-Site TSEM Implementation*

3.2.1 This SEM makes some demands and assumptions about the tester when it supports multiple test heads containing multiple sites per head.These requirements are as follows:

- Test systems that are capable of operating more than one virtual (or logical) test system at a time provide the virtual tester configuration data via the VirtualConfig variable specified in Section 8.
- The identification for multi-head and multi-site data (data variable, status variables, events, etc.) will be provided via list structures in SEMI E5 messages as detailed in Section 8.
- In the case where equipment supports more than one virtual (or logical) tester, all events and data items must have distinct CEIDs, DVIDs, and SVIDs for each virtual (or logical) tester.

4 Referenced Documents

4.1 *SEMI Standards*

SEMI E4 — SEMI Equipment Communications Standard 1 Message Transfer (SECS-I)

SEMI E5 — SEMI Equipment Communications Standard 2 Message Content (SECS-II)

SEMI E10 — Standard for Definition and Measurement of Equipment Reliability, Availability, and Maintainability (RAM)

SEMI E30 — Generic Model for Communications and Control of SEMI Equipment (GEM)

SEMI E37 — High-Speed SECS Message Services (HSMS) Generic Services

SEMI E37.1 — High-Speed SECS Message Services Single-Session Mode (HSMS-SS)

SEMI E58 — Automated Reliability, Availability, and Maintainability Standard (ARAMS): Concepts, Behavior, and Services

5 Terminology

5.1 *alignment location* — Site location that individual packaged units are aligned to at the process-site (e.g., electrical test).

5.2 *calibration fixture* — Any electromechanical fixture required to perform system calibration. May consist of multiple components with different part and serial numbers.

5.3 *class* — Classes represent the most coarse view of the test results. At a minimum, there should be two classes defined for each process program: one representing good devices and another representing failed devices.

5.4 *class, hard-bin, and soft-bin* — Equipment is to maintain DVVALs which provide three levels of granularity for test results: class, hard-bin, and soft-bin. Classes, hard-bins, and soft-bins are expected to be defined within a process program, and their names are made available as DVVALs. When device testing has completed, the process program is to determine the class, hard-bin, and soft-bin with which the device is to be associated, based on the results of the testing. Finally, a summation of the number of devices associated with each class, hard-bin, and soft-bin is also maintained throughout the execution of a process program, and these are also made available as DVVALs.

5.5 *diagnostic fixture* — Any electromechanical fixture required to perform system diagnostics. May consist of multiple components with different parts and serial numbers.

5.6 *execution area* — The area from which a current copy of the process program instructions is executed.

5.7 *hard-bin* — Hard-bins represent the typical view of the test results. Within a process program, each hard-bin is associated with a single class. Generally, multiple hard-bins are associated with a particular class.

5.8 *kit* — Specific items of hardware and software as specified by the equipment manufacturer that adapt the equipment for a specific unit or unit package.

5.9 *leadfinger (or substrate connector lead)* — (1) In ceramic packages, an area of refractory metal that has been plated and is designated for the attachment to a process-site. (2) The area of the unit designated for the attachment to a process-site.

5.10 *leadframe* — A sheet metal framework upon which a chip (sometimes chips) is attached, wire-bonded, then molded with plastic epoxy or with ceramic and/or metal.

5.11 *off-line programming (OLP) utility* — Utility to create, edit, and format process programs on a computer as opposed to creating process programs at the equipment.

5.12 *soft-bin* — Soft-bins represent the most detailed view of the test results. Within a process program, each soft-bin is associated with a single hard-bin. Generally, multiple soft-bins are associated with a particular hard-bin.

5.13 *system calibration* — Test system process required to bring the test system into compliance with the test system manufacturer's system specifications.

5.14 *test executive* — The tester software which controls test program execution.

5.15 *test head* — A resource of the tester. The electromechanical interface between the device/unit and the tester.

5.16 *test-site* — A specific site on a test head.

5.17 *test-board* — The electromechanical interface necessary to enable temporary electrical contact between the device/unit to be tested and the tester resource. May consist of multiple components.

5.18 *testing equipment* — An equipment class generally consisting of integrated mechanisms and controls for performing electrical tests of packaged devices and/or wafer die during the manufacturing process.

5.19 *unit* — The functional integrated circuit (or chip) that is to be electrically tested.

5.20 *virtual (or logical) tester* — That portion or portions of the complete test system that is capable of operating as an independent tester in accordance with the state model shown in this document. For a single test system with one test head and a single test-site on

the test head, the physical and virtual (or logical) tester are the same. In a two-headed test system, where each head can execute a unique process program autonomously and there is only one test-site on each head, two virtual (or logical) testers may be operating at the same time. If there are multiple test-sites on multiple heads, each capable of autonomous execution of a unique program, there are as many virtual (or logical) testers in operation as there are autonomous sites. The number of virtual (or logical) testers operating at any time depends on how the test system is currently configured (hardware and software) and may range from one to the maximum capability of the particular test system.

6 Requirements

6.1 *State Models*

6.1.1 The purpose is to define the equipment-specific processing state model and other state models necessary to portray the expected operational states of the equipment to enable host tracking and control in place of a local operator.

6.2 *State Model Requirements*

6.2.1 The processing state models in this document are required for implementing a TSEM-compliant tester, in addition to the required state models in SEMI E30. A state model consists of a processing state model diagram, processing state definitions, and a processing state transitions table. A state model represents the host's view of the tester, but not necessarily the actual tester operations. All TSEM state model transitions shall be mapped sequentially into the actual equipment events that satisfy the requirements of those transitions. In certain implementations, the tester may enter a state and have already satisfied all of the conditions required by the TSEM state model for transition to another state. In this situation, the tester makes the required transition without any additional actions.

6.2.2 Some equipment may need to include additional states. However, any additional states must not change the TSEM-defined state transitions. All expected transitions between TSEM states must occur.

6.3 *TSEM Process State Model*

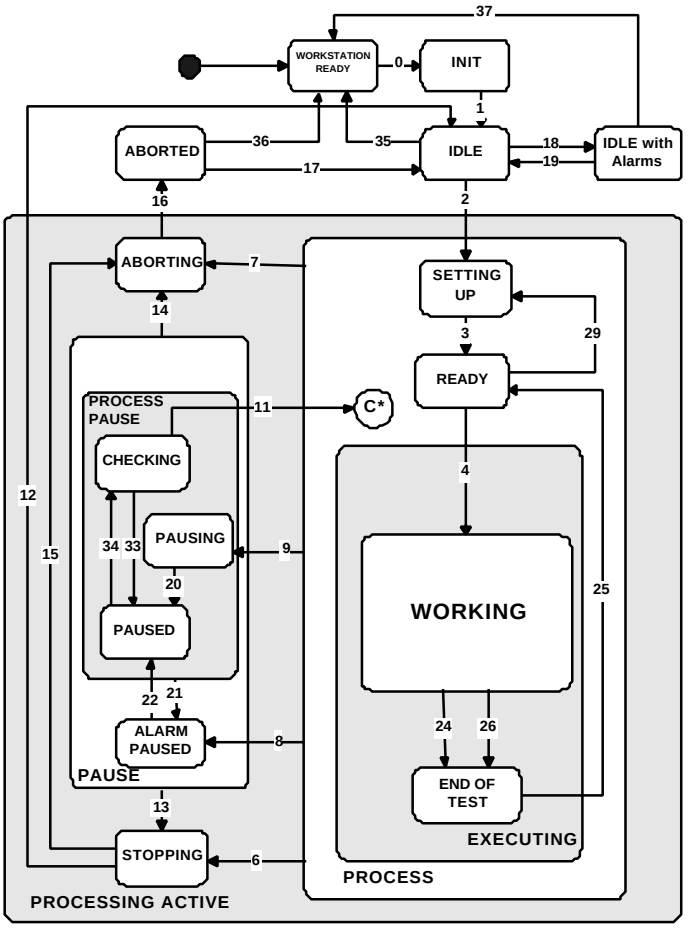

Figure 1
TSEM Processing State Model

6.4 *Description of Tester Processing States*

6.4.1 *ABORTED* — All activity is suspended as a result of an ABORT command. Any alarm and abort conditions must be cleared and verified by an operator before exit from this state.

6.4.2 *ABORTING (PROCESSING ACTIVE Sub-State)* — The tester has received an ABORT command. All activity is suspended. The tester is taking appropriate action to bring itself and material to a "safe" state where possible. Unit or Lot data may be invalid or not available.

6.4.3 *ALARM PAUSED (PAUSE Sub-State)* — An alarm has occurred in the PROCESS or PROCESS PAUSE states, and the tester is waiting for the alarm to be cleared.

6.4.4 *CHECKING (PROCESS PAUSE Sub-State)* — The tester verifies that updates made to the process program are valid. This is a similar procedure to that which is done in SETTING UP. At the successful completion of verification, a transition is made to the process state, based on the process model condition table.

6.4.5 *EXECUTING (PROCESS Sub-State*) — The tester is processing material automatically and can continue to do so without external intervention. This state may include interaction with the host or operator.

6.4.6 *IDLE* — Awaiting a command. IDLE is free of ALARMS and error conditions. A program may or may not be loaded in the execution space during this state.

6.4.7 *IDLE with ALARMS* — An alarm has occurred in the IDLE state, and the tester is waiting for all alarms to be cleared.

6.4.8 *INIT* — Tester initialization is occurring.

6.4.9 *PAUSED (PROCESS PAUSE Sub-State)* — The PROCESS state has been suspended, and the tester is waiting for a command (RESUME, STOP, or ABORT). In this state, the operator may correct error conditions and modify some conditions of the current Process Program selection.

6.4.10 *PAUSING (PROCESS PAUSE Sub-State)* — The current state will be suspended at the completion of the current unit(s), if any, and the tester will be brought to a "safe state."

6.4.11 *PROCESS (PROCESS Sub-State)* — This state is the parent of those sub-states that refer to the preparation and execution of a process program.

6.4.12 *PROCESS PAUSE (PAUSE Sub-State)* — The tester is free of alarm conditions in the PAUSE state.

6.4.13 *READY (PROCESS Sub-State)* — The tester is ready to begin processing and is awaiting a START command.

6.4.14 *SETTING UP (PROCESS Sub-State)* — The tester is satisfying conditions so that processing can begin. This includes the initialization of any process programs and process program-specific calibration. This may be accomplished independently by the tester or may require interaction with the operator and/or host.

6.4.15 *STOPPING (PROCESSING ACTIVE Sub-State)* — The tester has completed a Process Program or has been instructed to stop processing and shall do so at the next opportunity. All necessary cleanup is completed within this state with regard to material, data, control system, etc. Data is preserved. Any error condition is cleared before exiting from this state.

6.4.16 *END OF TEST (PROCESS Sub-State)* — The UNITS testing is complete.

6.4.17 *WORKING (EXECUTING Sub-State)* — The tester is processing a specific unit or units.

6.4.18 *WORKSTATION READY* — The tester workstation is running and ready for tester initialization. The START EXEC remote command is valid in this state.

6.5 *TSEM Processing State Transitions Table*

Table 1. Processing State Transitions Table

#	*Current State*	*Trigger*	*New State*	*Actions*	*Comments*
0	WORK-STATION-READY	Power on	INIT	None	None
1	INIT	All tester initialization is complete with no alarms or error conditions.	IDLE	None	None
2	IDLE	A Process Program is selected.	SETTING UP	Tester-dependent	Commit has been made to setup.
3	SETTING UP	All setup activity has completed, and the tester is ready to receive a START command.	READY	None	The selected Process Program is available for execution.
4	READY	The tester, operator, or host executes a START command, and auto-start is enabled.	WORKING	Begins testing the unit(s) at the test-site(s).	WORKING is an EXECUTING sub-state.
6	PROCESS	The tester has received a STOP command.	STOPPING	The tester completes the current unit(s) before entering the STOPPING state.	The tester begins its cleanup procedure.
7	PROCESS	The tester has received an ABORT command from operator, host, or self-generated.	ABORTING	The tester is put in a "safe" state.	Unit or lot data may be invalid or not available.
8	PROCESS	An alarm occurs.	ALARM PAUSED	PROCESS activity is suspended, and the tester is waiting for all alarms to be cleared.	ALARM PAUSED is a PAUSE sub-state.
9	PROCESS	The tester has received a PAUSE command.	PAUSING	The current state is suspended at the completion of the current unit(s). Any necessary actions to put the tester in a "safe" state will be performed.	PAUSING is a PROCESS PAUSE sub-state.

Table 1. Processing State Transitions Table

11	CHECKING	Parameter checking completes successfully.	STATE based on conditional table.	None	This is a conditional re-entry to the PROCESS state. (See Table 2.)
12	STOPPING	The tester cleanup is complete, and the tester is free of alarms.	IDLE	None	Data is preserved and is valid.
13	PAUSE	The tester has received a STOP command.	STOPPING	The tester proceeds with cleanup.	Data is preserved and is valid.
14	PAUSE	The tester has received an ABORT command.	ABORTING	Any unsafe condition is resolved, if possible.	Data may be invalid or unavailable.
15	STOPPING	The tester has received an ABORT command.	ABORTING	Any unsafe condition is resolved, if possible.	Data may be invalid or unavailable.
16	ABORTING	Unsafe conditions have been resolved, where possible.	ABORTED	The tester is waiting for alarm and ABORT conditions to be cleared.	The only state change allowed is to IDLE. Data is preserved and is valid.
17	ABORTED	An operator has verified that all alarms and abort conditions have been cleared.	IDLE	None	The IDLE state is a "clean" state.
18	IDLE	An alarm is set.	IDLE with ALARMS	The tester waits for all alarms to be cleared.	None
19	IDLE with ALARMS	All alarms have been cleared.	IDLE	None	The IDLE state is free of alarms.
20	PAUSING	The tester has completed Processing the Current unit(s) and achieved a "safe" condition.	PAUSED	The tester is waiting for a command (RESUME, STOP, or ABORT).	None
21	PROCESS PAUSE	An alarm is set.	ALARM PAUSED	The tester waits for all alarms to be cleared or for a STOP or ABORT command.	None
22	ALARM PAUSED	All alarms are cleared.	PAUSED	The tester is waiting for a command (RESUME, STOP, or ABORT).	None

Table 1. Processing State Transitions Table

24	WORKING	The processing of the current unit(s) has completed normally.	END OF TEST	The tester processes end of test data.	"Normal" completion of the test program execution.
25	END OF TEST	Tester is ready to receive a new start of test command.	READY	Waiting for start of next test.	
26	WORKING	The processing of the current unit(s) has completed abnormally.	END OF TEST	The tester processes end of test data.	"Abnormal" completion of the test program execution, etc.
29	READY	New Lot is received by tester.	SETTING UP	The tester performs setup based on the new command.	None
33	CHECKING	Error detected in a new parameter setting.	PAUSED	The tester waits for the parameter correction by operator or host.	None
34	PAUSED	A RESUME command with variable parameters was received.	CHECKING	Validation of the process program parameters begins.	None
35	IDLE	The tester executive has been stopped by the operator.	WORKSTATION READY	Waiting for a START EXECUTIVE command.	None
36	ABORTED	The tester executive has been aborted by the operator.	WORKSTATION READY	Waiting for a START EXECUTIVE command.	None
37	IDLE with ALARMS	The tester executive has been stopped by the operator.	WORKSTATION READY	Waiting for an ALARM Clear and START EXECUTIVE command.	None

6.6 *Process Model Conditions Table*

Table 2. Process Resume Conditions

Condition	*Next State*
Checking determines that process program conditions were changed.	SETTING UP
Previous State WORKING.	END OF TEST
Previous State READY.	READY
Previous State was SETTING UP.	SETTING UP

7 Collection Event List

7.1 *Requirements*

7.1.1 ALL SEMI E30-required Events are required by the TSEM. Since a Processing State Model is required by the TSEM, all state transitions are required Events.

7.1.2 All SEMI E30-required events associated with the SEMI E30 Control, Communications, Alarm, and Spooling State Models are required.

7.1.3 This section of the TSEM lists only those collection events that are not associated with a change of state or those requiring specific data variables (DVVALs) or Reports defined in the TSEM.

7.2 *Collection Event Tables* — The first table contains required events and associated reports. The second table contains required events and associated data variables.

Table 3. Processing State Transitions Requiring Report Levels

Transition	*From State*	*To State*	*Required DVVALs or Report*
SETUP COMPLETE (3)	SETTING UP	READY	Setup Report

Table 4. Other Required Collection Events

Collection Event Name	*Required DVVALs*
LotComplete	See Lot Report.
BoardChg	TestBoardID, CalFixtureID, or DiagFixture ID (Valid in CHECKING and SETTING UP.)
DockStatusChange	DockingStatus
BinDataAvailable	See Bin-Data Report.
SubLotComplete	See SubLot Report.
LotStart	Lot/SubLot Start Report
SubLotStart	Lot/SubLot Start Report

8 Variable Items

The purpose of this section is to define the list of variable items required by the TSEM. Values of these variables will be available to the host through collection event reports and host status queries.

8.1 *Requirements*

8.1.1 All generic variable items defined in SEMI E30 are required by all TSEM equipment.

8.1.2 Variable items required by TSEM are categorized as follows:

- *Common Variables (CVs)* — Variables common to all testers.
- *Configuration-Specific Variables (CSVs)* — Variables associated with a specific configuration of the above equipment class.

8.1.3 Any supplier-defined variables shall be documented in the same format used by this document. The following minimum information is required:

<variable name> **Class:** <ECV, SV, or DVVAL> **Format:** <SML>

Description: <If class = DVVAL, description must contain statement of when data is valid.>

<If format = ASCII, then a length is required. It is assumed to be left-justified unless otherwise noted.>

8.2 *Data Types*

8.2.1 Equipment Constants (ECVs) can be changed by the host using S2,F15. The operator may be able to change some values, but the equipment does not change the values on its own. The value of an equipment constant may be queried by the host at any time, using the S2,F13/14 transaction. They reside in non-volatile memory of the equipment. Equipment constants remain in effect until they are overwritten either by manual entry or by a S2,F15 (NEW EQUIPMENT CONSTANT SEND).

8.2.2 Equipment constants have various uses in TSEM, including the following:

- Equipment offsets that match the performance of several pieces of equipment that would otherwise perform differently due to inherent manufacturing differences. Examples are home values and motion axis scaling factors.
- Setting the configuration of the equipment to allow for different material specifications, equipment options, material flows, frequency of automatic functions, etc. An example is yield check frequency.
- Managing optional machine features. Examples are constants that indicate whether optional features such as automated media stackers are present and control the configuration and function of these optional subsystems when they are present.

8.2.3 Status Variables (SVs) are valid at all times. An SV may not be changed by the host but may be changed by the equipment or operator. The value of status variables may be queried by the host at anytime using the S1,F3/4 or S6,F19/20 transactions.

8.2.4 DVVALs are variables that are valid only upon the occurrence of specific collection events. An attempt to read a variable item at the wrong time does not generate an error, but the data reported may not have relevant meaning.

8.2.5 *Data Item Requirements for Multi-Head, Multi-Site Equipment* — The identification for multi-head and multi-site data (variable items, status variables, events, etc.) is addressed in this specification through the use of status variables. In Table 5, the subscript "v" is used to denote the number of virtual testers, "h" is used to denote the number of tester heads, "s" to denote the number of tester head sites, and "b" to denote the number of bins or classes.

8.3 *Variable Item Table*

Table 5. Variable Item Table

Variable Name	*Category*	*Description*	*Class*	*Format*	*Comments*
Physical Tester Group					
BaseConfig	CV	Base Tester Configuration listing all physical heads and sites.	SV	L, h TestHeadID L, s TestBoardSiteID	Valid in all states. Contains number of possible heads and sites for the tester.
ConfigInfo	CSV	Configuration Information	SV	A[256]	Valid in all states.
ConfigInfoType	CSV	Configuration information source (0 = Auto, 1 = Manual/File)	SV	U4	Valid in all states.
DatalogConfig	CV	Data Log Configuration	SV	A[256]	Valid in all states.
EquipSerialID	CV	Identification of Equipment	SV	A[1..40]	Valid in all states.
HandlerComStatus	CV	Status of comm link between handler/s (0 = Disabled, 1 = 1-way enabled, 2 = 2-way enabled, 3 = Not communicating)	SV	U4	Valid in all states.
LightPoleStatus	CSV	Color/status (i.e., Red/flash)	SV	A[1..16]	Valid in all states.
VirtualConfig	CSV	Current Virtual Configuration listing all virtual IDs.	DVVAL	L, v ...VirtualID L, h TestHeadID L, s TestBoardSiteID	Valid in PROCESS states. Contains active heads and sites for tester. List by virtual tester, head, and site.
Virtual Tester Group					
CalDate	CV	Date of last successful calibration.	SV	A[16]	Valid in all states. List by head.
CalInterval	CS	Time limit between calibrations.	SV	A[16]	Valid in all states. List by head.
CalStatus	CV	Status of last calibration (1 = OK, 0 = Failure)	SV	U4	Valid in all states. List by head.
ClassName	CV	Name tag for high-level class information.	DVVAL	A[1..40]	Valid at BIN-DATA-AVAILABLE state. List by bin.

Table 5. Variable Item Table

DiagStatus	CV	Status of last diagnostic (1 = OK, 0 = Failure)	SV	U4	Valid in all states. List by head.
HardBinName	CV	Test-Site Bin Out Name	DVVAL	A[1..40]	Valid at BIN-DATA-AVAILABLE Event. List by bin.
LotProcessingTime	CV	The time since start of current Lot.	DVVAL	A[16]	Valid in Process-sub-state.
OperatorID	CV	Current Operator ID	ECV	A[1..40]	Valid in all states.
ProductID	CV	ID of product for which tester is currently configured.	DVVAL	A[1..40]	Valid in PROCESS states.
SoftBinName	CV	Test-Site Category Name	DVVAL	A[1..40]	Valid at BIN-DATA-AVAILABLE Event. List by bin.
TestSiteInterval	CV	Interval count to generate event for a specific site $_{(n)}$ test.	ECV	U4	Valid in all states.
VirtualID	CV	ID of each virtual configuration.	SV	U4	
Test Head Group					
CalFixtureID	CV	ID of calibration fixture in current configuration.	SV	A[1..40]	Valid in all states. List by head.
DiagDate	CV	Date of last diagnostic execution.	SV	A[16]	Valid in all states. List by head.
DiagFixtureID	CV	ID of diagnostic fixture in current configuration.	SV	A[1..40]	Valid in all states. List by head.
DiagFixtureList	CV	List of current diagnostic, calibration, and test boards in current configuration.	SV	L, h DiagFixtureID, CalFixtureID, TestBoardIDList	Valid in all states. List by head.
DockingStatus	CSV	Information on handler/ docking status (0 = Yes, 1 = No)	SV	U4	Valid in all states. List by head.
HeadConfig	CV	Number of sites currently configured per head.	SV	U4	Valid in PROCESS states. List by head.
LotID	CV	Lot Identification	DVVAL	A[1..40]	Valid in all states.
StartTestPortID	CSV	Start Test Source (i.e., hand, keyboard, host)	SV	A[1..40]	Valid in all states. List by head.
SubLotID	CV	SubLot Identification	DVVAL	A[1..40]	Valid in all states.

Table 5. Variable Item Table

TestBoardID	CV	ID of current test board.	SV	A[1..30]	Valid in all states. List by head.
TestBoardIDList	CV	List of IDs in current test board configuration.	SV	A[1..32]	Valid in READY state. List by head.
TestHeadID	CV	The ID of a test head.	SV	U4	Valid in READY state. List by head.
TestHeadStatus	CV	(2 = Not Available, 1 = enabled, 0 = disabled)	SV	U4	Valid in IDLE state. List by head.
Test-Site Group					
ClassCnt	CV	Current count for a particular class.	DVVAL	U4	Valid in PROCESS states. List by head, bin, and site.
ClassID	CV	High-level class ID	DVVAL	U4	Valid in PROCESS states. List by head, bin, and site.
DeviceUnitID	CV	Unit Serial Number	DVVAL	U4	Valid in all states. List by head and site.
ExecutionCnt	CV	Number of test executives since last reset for the current PPID and current LOT at this test-site.	DVVAL	U4	Valid at END OF TEST sub-state. List by head and site.
HardBinCnt	CV	Test-Site Bin Out Count	DVVAL	U4	Valid in PROCESS states. List by head, bin, and site.
HardBinID	CV	Test-Site Bin Out Number	DVVAL	U4	Valid in PROCESS states. List by head, bin, and site.
LotUnitOutput	CSV	Bin output for a specific site by lot.	DVVAL	U4	Valid in PROCESS states. List by head and site.
SoftBinCnt	CV	Test-Site Category Count	DVVAL	U4	Valid in PROCESS states. List by head, bin, and site.
SoftBinID	CV	Test-Site Category Number	DVVAL	U4	Valid in PROCESS states. List by head, bin, and site.
SiteContacts	CSV	Number of contacts for a specific site.	DVVAL	U4	Valid in PROCESS states. List by head and site.
SubLotUnitOutput	CV	Bin output for a specific site by SubLot.	DVVAL	U4	Valid in PROCESS states.

Table 5. Variable Item Table

TestBoardSiteContacts	CSV	Number of contacts for a specific site since last reset.	DVVAL	U4	Valid in PROCESS states. List by head and site.
TestBoardSiteID	CV	X, Y location within testboard or probe card. (First element is the X coordinate, and second element is the Y coordinate.)	SV	U4(x)	Valid in all states. List by head and site.
TestBoardSiteInserts	CSV	Insertion-count on a test board site.	DVVAL	U4	Valid in PROCESSING state. List by head and site.
TestBoardSiteStatus	CV	Test Board Availability (1 = enabled, 0 = disabled)	DVVAL	U4	Valid in PROCESS states. List by head and site.
TestBoardStatus	CV	Test Board Availability (1 = enabled, 0 = disabled)	DVVAL	U4	Valid in PROCESS states. List by head and site.

8.4 *TSEM-Required Reports* — The reports below are required as "canned" or preconfigured reports by TSEM. TSEM does not require the equipment to guarantee the accuracy of data identified in these reports outside the PROCESSING ACTIVE state defined in the TSEM process state model.

8.4.1 *Setup Report* — Table 6 contains variables that are required to be available at the setup complete event.

Table 6. Setup Variables

Variable Name	*Notes*
DiagFixtureList	
LotID	
PPExecName	(per SEMI E30)
ProductID	Current
OperatorID	
TestBoardSiteStatus	
TestHeadID	
HandlerComStatus	
DockingStatus	

8.4.2 *Lot Complete Report* — Table 7 contains variables that must be available and reported at the completion of a lot.

Table 7. Lot Complete Variables

Variable Name	*Notes*
LotID	
PPExecName	(per SEMI E30)
DiagFixtureList	
SoftBinCnt	
HardBinCnt	
ClassCnt	
TestHeadID	
ContactCnt	
ExecutionCnt	
ProductID	
LotUnitOutput	
LotContacts	
LotProcessingTime	
OperatorID	

8.4.3 *SubLot Complete Report* — Table 8 below contains variables that must be available and reported at the completion of a SubLot.

Table 8. SubLot Complete Variables

Variable Name	*Notes*
SubLotID	
LotID	
PPExecName	(per SEMI E30)
DiagFixtureList	
SoftBinCnt	
HardBinCnt	
ClassCnt	
TestHeadID	
SubLotContacts	
SubLotUnitOutput	
ExecutionCnt	
ProductID	
OperatorID	

8.4.4 *Bin-Data-Report* — Table 9 contains variables that must be available once bin data is available.

Table 9. Bin-Data Variables

Variable Name	*Notes*
UnitID	
TestBoardSiteID	
HardBinID	
SoftBinID	
ClassID	

8.4.5 *Process Report* — Table 10 contains variables that must be available when the equipment is in the PROCESSING state.

Table 10. Process Variables

Variable Name	*Notes*
OperatorID	
LotID	
SubLotID	
PPExecName	(per SEMI E30)
OperationType	
DockingStatus	
HandlerCommStatus	
TestBoardSiteStatus	
LotProcessingTime	

8.4.6 *Lot/SubLot Report* — Table 11 contains variables that must be available at the completion of the lot or SubLot events.

Table 11. Lot/SubLot Start Variables

Variable Name	*Notes*
OperatorID	
LotID	
SubLotID	
PPExecName	(per SEMI E30)
DiagFixtureList	
TestHeadID	

8.4.7 *Calibration Report* — Table 12 below contains variables that must be available at the completion of a calibration.

Table 12. Calibration Variables

Variable Name	*Notes*
CalibrationInt	
PPExecName	(per SEMI E30)
OperatorID	
CalStatus	
TestHeadID	
CalFixtureID	

8.4.8 *Diagnostic Report* — Table 13 below contains variables that must be available at the completion of a diagnostic.

Table 13. Diagnostic Variables

Variable Name	*Notes*
OperatorID	
PPExecName	(per SEMI E30)
DiagFixtureList	
TestHeadID	
DiagStatus	

9 Process Program Management

9.1 *Process Program Requirements*

9.1.1 The TSEM requires that the GEM capability of process program management be fully supported for this class of equipment. The TSEM also requires that the process program have a structure that enables the user to build process programs with default conditions that can be overridden for a run. The concepts of process program structure are discussed in the following sections. The TSEM also requires the following:

- Minimum, maximum, and default parameter values must be defined for all process programs.
- *Verification* — When a process program is downloaded to the equipment, the program syntax must be verified by the equipment manufacturer. The process program may be rejected or may fail verification if the equipment is not in an allowable state to accept process program downloads (i.e., IDLE or SETUP).
- *Validation* — The downloaded process parameters must by type- and range-checked before execution.
- PPBODY — The contents of the downloaded process program body may contain the explicit parameters and data necessary for the runtime process program, or the body may contain reference information (i.e., PATH location) on where the explicit data is stored. In the latter case, it is required that the equipment combine or compile the reference data prior to the verification step.
- An error message must be generated from the tester if the process program parameters are outside the range of the machine capability.
- Diagnostic and calibration routines are considered process programs and must be verified and validated the same as a typical process program.

9.2 *Process Program Structure*

9.2.1 A tester process program must contain the following information:

- *Flow Information* — This provides information such as execution order of individual tests.
- *Parametric Information* — This section provides information such as AC/DC/IDC levels and timing.
- *Parameter Options/Values* — This section provides information such as test temperature, part frequency, etc.
- *Data Log Information* — This section provides information such as scope of data collection.
- *Functional Test Information* — This section provides information about vector and patterns.

9.2.2 This information must be supplied in a format that can be referenced as a complete process program. It is emphasized that the TSEM does not specify the exact number, data types, and format of this process program.

9.3 *Methods of Process Program Creation* — The method by which an equipment manufacturer creates a process program may be unique to that manufacturer. However, it is required that the customer at least be given both of the following options for the creation of a process program.

9.3.1 *Off-Line Development* — Using this method, the customer is given a set of software tools (process program compilers, decompilers, and debuggers) that will enable him/her to generate or create a process program using the above mentioned information (flow, parameter, functional test, etc.). The newly generated process program then is downloaded onto a specific tester, is verified, and is now ready to be selected and executed locally by the operator or remotely by the host computer. If this process is used, the supplied software tools should closely mimic or simulate a tester so that a user can create a complete process program. In many situations, minor adjustments or tweaks may be needed to the process program on the equipment before it is completely ready for execution.

9.3.2 *On-Line Development* — The second option made is to enable the user to download the above-mentioned information (tables or files) onto the equipment and create the actual process program on the equipment itself.

10 Remote Commands

The purpose of this section is to identify remote commands, command parameters, and valid commands versus states in the processing state models.

10.1 *Requirements*

- The equipment must support the GEM-required remote commands. (Some of the SEMI E30-required remote commands are restated here to define TSEM-specific requirements.)
- All the remote commands defined by TSEM are required.
- The alphanumeric strings defined by TSEM for remote commands (RMCD) and command parameters (CPNAME) are required.
- If additional remote commands are supported, then the "Remote Command vs. Valid States" matrix must be generated for these additional commands. Place an "X" in the table for each state in which a given command is valid.

10.2 *Remote Command Descriptions*

10.2.1 *ABORT* — This command terminates the current processing. ABORT makes no guarantee about completion of the current test(s). Lot level data will be preserved. Levels of ABORT may be specified (see Table 14 for details).

10.2.2 *PAUSE* — This command transitions the tester to the PAUSING process state when the current test(s) completes processing.

10.2.3 *RESUME* — This command resumes processing from the point where the process was PAUSED. This command is only recognized if the tester is in the PAUSED or CHECKING state.

10.2.4 *PP-SELECT* — This command instructs the tester to copy the indicated Process Program from non-volatile storage to the tester's Process Program execution area. Process Program Variable Parameters can be specified in this command which modify the default values for these Variable Parameters in the Process Program. Process program verification (CHECKING state) must occur when variable parameters accompany this command.

10.2.5 *START* — This command is only available to the host or operator when a process program has been selected and the tester is in the READY processing state. The START command instructs the tester to initiate processing. Parameters can be specified in this command.

10.2.6 *STOP* — This command completes the current test(s), stops in a safe condition, and returns to the IDLE processing state. STOP has the intent of bringing about a normal termination after completion of the current test(s). Parameters can be specified in this command. Lot level data will be preserved.

10.2.7 *UNLOAD-PGM* — This command instructs the tester to unload the specified process program, or programs in the case of multiple heads, from the execution area. Parameters can be specified in this command.

10.2.8 *NEW-LOT* — This command instructs the tester to treat the next units to be processed as a new lot. A new-lot ID and process program variables must accompany the new lot command. This command will force subsequent units to be considered part of the lot. Parameters can be specified in this command.

10.2.9 *CLOSE-LOT* — This command instructs the tester to close the current lot. The next lot will require the setup procedure to be performed.

10.2.10 *CALIBRATE* — This command instructs the tester to execute its calibration program. Parameters can be specified in this command.

10.2.11 *START-EXEC* — This command instructs the tester to start the tester executive.

10.2.12 *STOP-EXEC* — This command instructs the tester to stop the tester executive. Lot level data will be preserved.

10.2.13 *RUN-DIAGNOSTICS* — This command instructs the tester to run the specified preventive maintenance diagnostic routine. Parameters can be specified in this command.

10.2.14 *RUN-CONTINUITY* — This command instructs the tester to run the specified continuity routine. Valid in PAUSE state only. Parameters can be specified in this command.

10.2.15 *ENABLE-SITE* — This command instructs the tester to enable a specified test board site(s). Parameters can be specified in this command.

10.2.16 *DISABLE-SITE* — This command instructs the tester to disable a specified test head site. Parameters can be specified in this command.

10.2.17 *RESET-SITE-CNT* — This command instructs the tester to reset the counts on the specified board test board site(s) to zero.

10.3 *Associated Remote Command Parameters*

Table 14. Remote Command Descriptions

Command	*Parameter*			
Name	*Name*	*Opt./Req.*	*Description*	*Format*
ABORT	"VIRTUALID"	OPT	ID of virtual tester to abort.	U4
	"LOTCLOSE"	OPT	The tester will close out the current lot and enter the ABORTING state.	A[7]
CALIBRATE	"PPID"	OPT	The ID of the program to be used.	A[80]
CLOSE-LOT	"VIRTUALID"	OPT	ID of virtual tester.	U4
DISABLE-SITE	"TESTHEADID"	REQ	ID of test head containing the site.	U4
	"TESTSITEID"	REQ	ID of test-site to be disabled.	U4
	"VIRTUALID"	OPT	ID of virtual tester.	U4
ENABLE-SITE	"VIRTUALID"	OPT	ID of virtual tester.	U4
	"TESTHEADID"	REQ	ID of test head containing the site.	U4
	"TESTSITEID"	REQ	ID of test-site to be enabled.	U4
NEW-LOT	"VIRTUALID"	OPT	ID of virtual tester.	U4
	"LOTID"	REQ	ID of New LOT.	A[1..40]
PAUSE	"VIRTUALID"	OPT	ID of virtual tester.	U4

Table 14. Remote Command Descriptions

PP-SELECT	"VIRTUALID"	OPT	ID of virtual tester.	U4
	"LOTID"	OPT	Lot to be processed with this program.	A[1..40]
RESET-SITE-CNT	"VIRTUALID"	OPT	ID of virtual tester.	U4
	"TESTHEADID"	REQ	ID of test head containing the site.	U4
	"TESTSITEID"	REQ	ID of test-site to be reset.	U4
RESUME	"VIRTUALID"	REQ	ID of virtual tester.	U4
RUN-CONTINUITY	"PPID"	OPT	The ID of the program to be used.	A[80]
RUN-DIAGNOSTICS	"PPID"	OPT	The ID of the program to be used.	A[80]
START	None			
START-EXEC	None			
STOP	"VIRTUALID"	REQ	ID of virtual tester.	U4
	"CLOSELOT	OPT	Automatically close lot.	BOOL
STOP-EXEC	"VIRTUALID"	REQ	ID of virtual tester.	U4
UNLOAD-PGM	"VIRTUALID"	OPT	ID of virtual tester to unload.	U4
	"PPID"	REQ	ID of program to unload.	A[80]
	"ALL"	OPT	Will unload all test programs.	BOOL

10.4 *Remote Commands and TSEM Process Model Mapping* — Table 15 illustrates the relationship between remote commands and states of the TSEM processing state model. An "X" indicates that a command is valid for use in this state. If a remote command is attempted during a non-valid state, the equipment would reject the remote command.

Table 15. Remote Commands vs. Process States

COMMAND	ABORT	START-EXEC	STOP-EXEC	NEW-LOT	PAUSE	PP-SELECT	UNLOAD-PGM	RESUME	START	STOP
PROCESSING STATE										
INIT									X	
IDLE				X	X			X		
IDLE with Alarms								X		
PROCESSING ACTIVE										
PROCESS										
SETTING UP	X					X	X			X
READY	X	X				X	X			X
ABORTING										
STOPPING										X
EXECUTING										
WORKING	X					X				X
ENDOFTEST	X					X				X
PROCESS PAUSE										
PAUSING										X
PAUSED	X		X				X			X
CHECKING										X
ALARM PAUSED	X									X
WORKSTATION READY									X	
ABORTED										

Table 16. Remote Commands vs. Process States #2

COMMAND / PROCESSING STATE	CALIBRATE	RUN-DIAGNOSTICS	CLOSE-LOT	RUN-CONTINUITY	RESET-SITE-COUNT
INIT					
IDLE	X	X	X	X	X
IDLE with Alarms					
PROCESSING ACTIVE					
PROCESS					
SETTING UP	X		X		
READY	X	X	X		
ABORTING					
STOPPING					
EXECUTING					
WORKING					
ENDOFTEST					
PROCESS PAUSE					
PAUSING					
PAUSED	X	X	X		
CHECKING					
ALARM PAUSED					
WORKSTATION READY					
ABORTED					

11 Scenarios

The purpose of this section is to document possible TSEM-specific scenarios illustrating the possible virtual configurations. The example below is for a single tester configured as two virtual testers configured as follows:

```
Physical Tester #1  EquipSerialID = T1000A
2 Virtual Equipment Instances,
         where VirtualID #10 represents
         Tester #0, Head #0, Sites 0 - 3.

         where VirtualID #20 represents
         Tester #0, Head #0, Sites 4 - 7,
         Head #1, ALL Sites (1-3).

Status Variable Request Scenario
Status Variable Request       S1,F3 -->
L,1
  1. U4 1001 (SVID for VirtualConfig)
                                      <--S1,F4 (status variable return)
                                            L,1
                                              1. L,2 = # of virtual testers
                                                 1. L,2
                                                    1. 10 (VirtualID = 10)
                                                    2. L,1 = # of heads for Tester 10
                                                       1. L,2
                                                          1. 0 (TestHeadID)
                                                           2. L,3 = # of sites for
                                                                head 1
                                                            1. 01 = TestBoardSiteID #1
                                                            2. 02 = TestBoardSiteID #2
                                                            3. 03 = TestBoardSiteID #3
                                                 2. L,2
                                                    1. 20 (VirtualID = 20)
                                                    2. L,2 = # of heads for Tester 20
                                                       1. L,2
                                                          1. 0 (TestHeadID)
                                                           2. L,4 = # of sites for
                                                              head 0
                                                            1. 04 = TestBoardSiteID #4
                                                            2. 05 = TestBoardSiteID #5
                                                            3. 06 = TestBoardSiteID #6
                                                            4. 07 = TestBoardSiteID #7
                                                       2. L,2
                                                          1. 1 (TestHeadID)
                                                           2.  L,3 = # of sites for
                                                               head 1
                                                            1. 01 = TestBoardSiteID #1
                                                            2. 02 = TestBoardSiteID #2
                                                            3. 03 = TestBoardSiteID #3
```

11.1 *Normal Run Scenario* — This is an error-free run of a single lot, with no additional lots queued. The Host determines the available resources of the tester by requesting the status variable, VirtualConfig. This variable returns a list of virtual testers and the resources assigned to each of them.

COMMENT	**HOST**	**EQUIPMENT**	**COMMENT**

The Host would initiate a PP_SELECT using an available resource. PP_SELECT requires at least 2 parameters (PPID and VirtualID). Optional parameters are not shown.

```
Host Command Send              S2,F41-->
1. PP-SELECT
2. L,2
   1. L,2
         1. PPID
         2. A[80] "/home/recipes/tester_recipe"
   2. L,2
         1. VirtualID
         2. U4 10

                                        <--S2,F42     (host command acknowledge)
                                                         L,2
                                                            1. HCACK "0"
                                                            2. L,0
```

The tester then starts sending several events back to the host to identify transitions and non-transition events.

```
                                        <--S6,F11             (Event #2 Setting Up)
                                S6,F12-->
                                        <--S6,F11             (Event #3 Ready)
                                S6,F12-->
```

The host then sends a start to the tester. The START command requires the VirtualID only.

```
Host Command Send              S2,F41-->
1. START
2. L,1
   1. L,2
         1. VirtualID
         2. U4 10

                                        <--S2,F42     (host command acknowledge)
                                                         L,2
                                                            1. HCACK "0"
                                                            2. L,0
```

The tester then starts sending several events back to the host to identify transitions and non-transition events.

```
                                    <--S6,F11            (Event #4 Working)
                                S6,F12-->
                                    <--S6,F11            (Event # EndOfTest)
                                S6,F12-->
                                    <--S6,F11            (Event # EndOfTest)
                                S6,F12-->
                                    <--S6,F11            (Bin data available)
                                S6,F12-->
                                    <--S6,F11            (Event #25 Ready)
                                S6,F12-->
```

After several of these starts and events sequences, the tester notifies the host that the SubLot is complete and then receives close-lot and stop commands from the host.

```
                                    <--S6,F11            (Sub-lot complete)
                                S6,F12-->

Host Command Send               S2,F41-->
1. CLOSE_LOT
2. L,0

                                    <--S2,F42    (host command acknowledge)
                                                         L,2
                                                            1. HCACK "0"
                                                            2. L,0

                                    <--S6,F11            (Lot-complete)
                                S6,F12-->
Host Command Send               S2,F41-->
1. STOP
2. L,0

                                    <--S2,F42    (host command acknowledge)
                                                         L,2
                                                            1. HCACK "0"
                                    <--S6,F11               (Event #12 IDLE)
                                S6,F12-->
```

11.2 *Run Diagnostics/Calibration Scenario*

```
Run Diag/Calibration
Host Command Send              S2,F41-->
1. CALIBRATE
2. L,1
   1. L,2
         1. START
         2. A[80] "DIAG-ID"

                                          <--S2,F42     (host command acknowledge)
                                                  L,2
                                                     1. HCACK "0"
                                                     2. L,0
                                          <--S6,F11     (Event #2)
                                S6,F12-->
                                          <--S6,F11     (Event #3)
                                S6,F12-->

Host Command Send              S2,F41-->
 1. START
2. L,0

                                          <--S2,F42     (host command acknowledge)
                                                  L,2
                                                     1. HCACK "0"

                                          <--S6,F11             (Event #4)
                                S6,F12-->
                                          <--S6,F11             (Event #24)
                                S6,F12-->
                                          <--S6,F11             (Diag Report)
                                S6,F12-->

Host Command Send              S2,F41-->
1. STOP
2. L,0

                                          <--S2,F42     (host command acknowledge)
                                                  L,2
                                                     1. HCACK "0"
```

12 Additional SEMI E30 Requirements

The purpose of this section is to specify any GEM additional capabilities that are required to be supported by this class of equipment.

12.1 *Requirements* — The following GEM additional capabilities required by TSEM are:

- Establish Communications
- Dynamic Event Report Configuration
- Variable Data Collection
- Status Data Collection
- Alarm Management
- Remote Control
- Equipment Constants
- Process Program Management
- Equipment Terminal Services
- Clock
- Spooling
- Control (Host-Initiated)

13 TSEM Unique Capabilities

The purpose of this section is to specify additional capabilities required for the TSEM that are unique to this class of equipment.

13.1 *Test Handling Equipment Common Data* — The purpose of this subsection is to specify test handling variable item data and event data that must be available to the host during the electrical test process. Because equipment configurations vary regarding handling and test equipment, passing control and process information to the host system also can vary. By providing this information from either class of equipment, a greater variety of configurations is available to the user. The handler data identified in this section must be made accessible from the tester manufacturer's interface (i.e., placeholder IDs, events, and commands). The validity of the data will depend on the specific field configuration of the equipment. For example, if the field configuration places the handler as the primary contact to the host, the data identified in this section would not be needed, and the TSEM capability would be disabled. On the other hand, if the field configuration places the tester as the primary contact to the host, the data identified in this section would be necessary, and the TSEM capability then would be enabled.

13.2 *Variable Item Requirements* — Table 17 identifies variable items that must be available from the tester in addition to those identified in Section 8.

Table 17. Common Handler Variable Items

Variable Name	*Category*	*Description*	*Class*	*Format*	*Comments*
EquipID (Handler)	CV	Identification of Handler Equipment (per Head)	SV	A[16]	Valid in all states.
LotID	CV	Lot ID	SV	A[16]	Valid in EXECUTING.
OperatorID	CSV	Current Operator ID	ECV	A[24]	Valid in all states.

13.3 *Collection Event Data Item Requirements* — Table 18 identifies common collection events that the tester must be able to provide to the host, if available from the handler.

Table 18. Common Handler Collection Events

Collection Event Name	*Event #*
LotComplete	Equipment Specific #
SetupComplete	Equipment Specific #
LotStart	Equipment Specific #

13.4 *Host Access to Tester Data Log Information* — TSEM requires the equipment manufacturer to make data log information available to the host via the SEMI E5, Stream 13 data set message. Because equipment configurations vary for data log content and format, the only requirement TSEM makes is that the equipment manufacturer document the format and content of data log information used by the equipment and make that information available across the communications interface via Stream 13. (See SEMI E5.)

RELATED INFORMATION 1

NOTE: This related information is not an official part of SEMI E30.3 and is not intended to modify or supercede the official standard. These notes are presented as possible methods for SEM implementations and are included only as reference material.

R1-1 TSEM ARAMS Sub-State Codes

For TSEM implementations which are also compliant with SEMI E58, the equipment will support the following ARAMS sub-states:

A. PRODUCTIVE

1. 1XYY "PRD/Initialization for process program"

2. 1XYY "PRD/Testing units"

3. 1XYY "PRD/Determining result information"

4. 1XYY "PRD/Cleanup for process program"

B. STANDBY

1. 2XYY "SBY/Waiting for process program selection"

2. 2XYY "SBY/Waiting for start test"

3. 2XYY "SBY/Waiting for user input, equipment-initiated"

4. 2XYY "SBY/Waiting for user input, user-initiated"

NOTE: The *X* sub-state codes are reserved by SEMI E58 for standard codes and the *Y* is for supplier usage.

R1-1.1 *Mapping of TSEM Processing State Model to ARAMS State Model* — The TSEM Processing state model and the ARAMS state model are separate models which must both be maintained and supported if the equipment is to be both TSEM-compliant equipment and ARAMS-compliant. Although these state models are separate, there is a definite relationship between the two. All TSEM-compliant equipment which is also ARAMS-compliant will support the following mapping between the TSEM Processing state model and the ARAMS state model. This mapping only applies while the equipment is performing its intended function, that is, while it is in the Manufacturing superstate defined by ARAMS.

R1-1.2 *ARAMS State Mapping to TSEM Processing States*

Table 19. ARAMS/TSEM Processing State Transitions Table

#	*Current State*	*Trigger*	*New State*	*ARAMS Actions*	*Comments*
0	WORK-STATION-READY	Power on.	INIT	Based on previous ARAMS.	Based on previous ARAMS state.
1	INIT	All tester initialization is complete with no alarms or error conditions.	IDLE	SBY/Waiting for PP selection.	Equipment is capable of performing the intended function.
2	IDLE	A Process Program is selected.	SETTING UP	PRD/Initialization for PP selection.	None
3	SETTING UP	All setup activity has completed, and the tester is ready to receive a START command.	READY	SBY/Waiting for start test.	Start Test may be received.
4	READY	The handler, operator, or host executes a START command, and auto-start is enabled.	WORKING	PRD/Testing Units	None
6	PROCESS	The tester has received a STOP command.	STOPPING	PRD/Cleanup	None
7	PROCESS	The tester has received an ABORT command from operator, host, or self-generated.	ABORTING	PRD/Cleanup	None
8	PROCESS	An alarm occurs.	ALARM PAUSED	UDT	Not capable of performing intended function.
9	PROCESS	The tester has received a PAUSE command.	PAUSING	No state change.	PAUSING is incomplete, so no change.
11	CHECKING	Parameter checking completes successfully.	STATE based on conditional table.	Condition based on conditional state.	Reference # 2, 3, 4, 24, 26, and 29.
12	STOPPING	The tester cleanup is complete, and the tester is free of alarms.	IDLE	SBY/Waiting for PP selection.	Equipment is capable of performing the intended function.

Table 19. ARAMS/TSEM Processing State Transitions Table

13	PAUSE	The tester has received a STOP command.	STOPPING	PRD/Cleanup	None
14	PAUSE	The tester has received an ABORT command.	ABORTING	UDT or PRD Cleanup.	If uncleared alarms exist, then ARAMS state is UDT, else cleanup.
15	STOPPING	The tester has received an ABORT command.	ABORTING	UDT or no state change.	If uncleared alarms exist, then ARAMS state is UDT, else stopping.
16	ABORTING	Unsafe conditions have been resolved, where possible.	ABORTED	SBY/Waiting for input or UDT.	If uncleared alarms exist, then ARAMS state is UDT, else waiting for clear.
17	ABORTED	An operator has verified that all alarms and abort conditions have been cleared.	IDLE	SBY/Waiting for PP selection.	None
18	IDLE	An alarm is set.	IDLE with ALARMS	UDT	Equipment is NOT capable of performing the intended function.
19	IDLE with ALARMS	All alarms have been cleared.	IDLE	SBY/Waiting for PP selection.	Equipment is capable of performing the intended function.
20	PAUSING	The tester has completed processing the Current unit(s) and achieved a safe condition.	PAUSED	SBY/Waiting for input.	Waiting for resume.
21	PROCESS PAUSE	An alarm is set.	ALARM PAUSED	UDT	Equipment is NOT capable of performing the intended function.
22	ALARM PAUSED	All alarms are cleared.	PAUSED	SBY/Waiting for input.	Alarms cleared, waiting for resume.
24	WORKING	The processing of the current unit(s) has completed normally.	END OF TEST	PRD	ARAMS sub-state is active until the BIN-Data Available event occurs or the next start test is received.

Table 19. ARAMS/TSEM Processing State Transitions Table

25	END OF TEST	Tester is ready to receive a new start of test command.	READY	Either SBY/Waiting for start test or PRD determining results.	PRD/Determining sub-state has entered at END OF TEST. This sub-state is active until BIN-Data Available or until the next start test.
26	WORKING	The processing of the current unit(s) has completed abnormally.	END OF TEST	PRD determining results.	The sub-state is active until BIN-Data Available or until the next start test.
29	READY	New Lot is received by tester.	SETTING UP	PRD/Initialization for PP selection.	New lot requires process program initialization.
33	CHECKING	Error detected in a new parameter setting.	PAUSED	No state change.	None
34	PAUSED	A RESUME command with variable parameters was received.	CHECKING	No state change.	None
35	IDLE	The tester executive has been stopped by the operator.	WORKSTATION READY	No state change.	The equipment may not be aware of this transition. ARAMS specifies the rules to be followed upon reinitialization.
36	ABORTED	The tester executive has been aborted by the operator.	WORKSTATION READY	No state change.	The equipment may not be aware of this transition. ARAMS specifies the rules to be followed upon reinitialization.
37	IDLE with ALARMS	The tester executive has been stopped by the operator.	WORKSTATION READY	No state change.	The equipment may not be aware of this transition. ARAMS specifies the rules to be followed upon reinitialization.

R1-1.3 *Additional ARAMS Capabilities* — ARAMS specifies fundamental ARAMS requirements which must be met to be ARAMS-compliant. ARAMS also specifies additional capabilities which may be provided by ARAMS-compliant equipment. All TSEM-compliant equipment which is also ARAMS-compliant is required to provide the following additional capabilities as defined by ARAMS:

a. Dynamic Event Report Configuration

b. Accumulator Data

c. User-Generated ARAMS Sub-State Table(s)

d. Equipment-Generated ARAMS Sub-State Table(s)

e. User-Generated ARAMS Symptom Table(s)

NOTICE: These standards do not purport to address safety issues, if any, associated with their use. It is the responsibility of the user of these standards to establish appropriate safety and health practices and determine the applicability of regulatory limitations prior to use. SEMI makes no warranties or representations as to the suitability of the standards set forth herein for any particular application. The determination of the suitability of the standard is solely the responsibility of the user. Users are cautioned to refer to manufacturer's instructions, product labels, product data sheets, and other relevant literature respecting any materials mentioned herein. These standards are subject to change without notice.

The user's attention is called to the possibility that compliance with this standard may require use of copyrighted material or of an invention covered by patent rights. By publication of this standard, SEMI takes no position respecting the validity of any patent rights or copyrights asserted in connection with any item mentioned in this standard. Users of this standard are expressly advised that determination of any such patent rights or copyrights, and the risk of infringement of such rights, are entirely their own responsibility.