SEMI E40-0702
STANDARD FOR PROCESSING MANAGEMENT

This standard was technically approved by the Global Information & Control Committee and is the direct responsibility of the North American Information & Control Committee. Current edition approved by the North American Regional Standards Committee on April 30, 2002. Initially available at www.semi.org June 2002; to be published July 2002. Originally published in 1995; previously published November 2001.

1 Purpose

1.1 Automated management and command of material processing in equipment is a crucial aspect enabling factory automation. This standard addresses the communications needs within the semiconductor manufacturing environment with respect to the processing of material in an equipment.

1.2 This standard specifies the application of the appropriate processing to specified material received at the processing agent. It describes the concepts of material processing, the behavior of the equipment in relation to processing, and the messaging services which are needed to accomplish the task.

1.3 The communications services defined here enable standards-based interoperability of independent systems. They allow application software to be developed that can assume the existence of these services and allow software products to be developed which offer them.

1.4 Implementation of automated processing management will help eliminate misprocessing of material. The adoption of the standards described will greatly reduce the effort required to integrate compliant equipment components and reduce time to set up for processing. Compliance requires a minimal but specific set of standard services.

2 Scope

2.1 The scope of this standard is automated material processing based on discrete processing jobs. It provides the functionality required for process management for modules within a cluster tool. It may be applied to sub-systems of other multi-resource equipment, as well as to host control of many types of equipment.

2.2 This standard supports individual management of jobs for identical processing of material within a group and concurrent processing of independent groups. Where material contains other material (such as carriers containing wafers), processing may be specified in terms of either material type.

2.3 A simple tuning mechanism is provided for limited feedforward and feedback control between process steps. A method is defined for taking advantage of recipe variable parameters. This is not expected to satisfy all closed loop control requirements. Other mechanisms are anticipated with greater flexibility for late tuning and handling complex data.

2.4 This standard does not provide services for receiving material for processing, or disposing of it after processing is complete. Automation of material transfer is assumed to be provided through other services, such as those defined in applicable SEMI standards.

2.5 This standard presents a solution from the concepts and behavior down to the messaging services. It does not define the messaging protocol.

2.6 A messaging service includes the identification that a message shall be exchanged and a definition of the data which is contained in that message. It does not include information on the structure of the message, how the data is represented within the message, or how the message is exchanged. This additional information is contained within the message protocol.

2.7 The defined services may be applied to multiple protocols. Information on the mapping of processing management services to special protocols (e.g., SECS-II) are added as adjunct standards.

2.8 The services assume a communications environment in which a reliable connection has been established between the user of the services and the provider of the services. Establishing, maintaining, releasing a connection, and handling communication failures are beyond the scope of this standard.

2.9 This standard does not purport to address safety issues, if any, associated with its use. It is the responsibility of the users of this standard to establish appropriate safety and health practices and determine the applicability of regulatory limitations prior to use.

3 Referenced Standards

3.1 *SEMI Standards*

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

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

SEMI E39 — Object Services Standard: Concepts, Behavior, and Services

SEMI E53 — Event Reporting

NOTE 1: As listed or revised, all documents cited shall be the latest publications of adopted standards.

4 Terminology

4.1 The following definitions are arranged in alphabetical order. Some definitions use terms defined elsewhere within this section. No references beyond this section should be necessary for a basic understanding of these terms.

4.2 *Definitions*

4.2.1 *agent* — an intelligent system within a factory that provides one or more service resources and uses the services of other agents. A generalization of host, equipment, cell, cluster, cluster module, station controller, and work station. Agents are associated with a physical system or a collection of physical systems, including computer platforms.

4.2.2 *form* — type of data representing information contained in an object attribute or service message parameter.

4.2.3 *fundamental requirements* — the requirements for information and behavior that must be satisfied for compliance to a standard. Fundamental requirements apply to specific areas of application, objects, or services.

4.2.4 *post-conditioning* — activities performed by the processing resource after departure of the material being processed but related to the processing of that material (e.g., cleanup).

4.2.5 *pre-conditioning* — activities performed by the processing resource before arrival of the material being processed but related to the processing of that material.

4.2.6 *processing agent* — an intelligent system within a factory which is independently capable of providing manufacturing value added to material.

4.2.7 *processing resource* — an entity within a processing agent which provides the manufacturing value added to material.

4.2.8 *process job* — a material processing job for a processing resource specifying and tracking the processing to be applied to the material.

4.2.9 *recipe* — the pre-planned and reusable portion of the set of instructions, settings, and parameters under control of a processing resource that determines the processing environment seen by the material. Recipes may be subject to change between runs or processing cycles.

4.2.10 *recipe executor* — a component of a module that stores and executes recipes.

4.2.11 *recipe namespace* — a logical management domain with the responsibility for the storage and management of recipes. It ensures the uniqueness of recipe identifiers and provides services pertaining to recipes stored within that domain.

4.2.12 *service* — the set of messages and definition of the behavior of a service-provider that enables remote access to a particular functionality.

4.2.13 *service-provider* — the software control entity that is the provider of a particular functionality which may be accessible remotely.

4.2.14 *service-user* — the software control entity that is the user of any of the related services.

4.2.15 *supervisor* — an entity or entities having supervisory control responsibilities for one or more processing resource. It is the service-user of the processing management services.

4.2.16 *tuning* — specification of information which supplements the pre-defined recipe used to achieve the particular process goals.

4.3 *Data Type*

4.3.1 *boolean* — may take on one of two possible values, equating to TRUE or FALSE.

4.3.2 *enumerated* — may take on one of a limited set of possible values. These values may be given logical names, but they may be represented by any single-item data type.

4.3.3 *form* — type of data: positive integer, unsigned integer, integer, enumerated, boolean, text, formatted text, structure, list, ordered list.

4.3.4 *formatted text* — a text string with an imposed format. This could be by position, by use of special characters, or both.

4.3.5 *integer* — may take on the value of any negative or unsigned integer. Messaging protocol may impose a limit on the range of possible values.

4.3.6 *list* — a set of one or more items that are all of the same form (one of the above forms).

4.3.7 *ordered list* — a list for which the order in which items appear is significant.

4.3.8 *positive integer* — may take the value of any positive whole number. Messaging protocol may impose a limit on the range of possible values.

4.3.9 *structure* — a complex structure consisting of a specific set of items, of possibly mixed data types, in a specified arrangement.

4.3.10 *text* — a text string. Messaging protocol may impose restrictions, such as length or ASCII representation.

4.3.11 *unsigned integer* — may take the value of any positive integer or zero. Messaging protocol may impose a limit on the range of possible values.

5 Conventions

5.1 *Harel State Model* — This document uses the Harel State Chart notation to describe the dynamic behavior of the objects defined. An overview of this notation is presented in an appendix of SEMI E30. The formal definition of this notation is presented in *Science of Computer Programming 8*, "Statecharts: A Visual Formalism for Complex Systems," by D. Harel, 1987.

5.1.1 The Harel notation does not include the concept of "creation" and "deletion" of state models to represent transient entities. The "job" described in this document is such an entity, where each new job created uses a copy of the same state model. In this document, an oval is used to denote the creation of an entity and also the deletion of that entity.

5.1.2 Transition tables are provided in conjunction with the state diagrams to describe explicitly the nature of each state transition. A transition contains columns for Transition #, Current State, Trigger, New State, Action(s). The "trigger" (column 3) for the transition occurs while in the "current" state. The "actions" (column 5) include a combination of (1) actions taken upon exit of the current state, (2) actions taken upon entry of the new state, and (3) actions taken which are most closely associated with the transition. No differentiation is made.

5.1.3 The state models included in this standard are a requirement for Processing Management compliance. A state model consists of a state model diagram, state definitions, and a state transition table. When using SEMI E30, E53 or similar style collection events, all state transitions in this standard, unless otherwise specified, shall correspond to collection events.

5.1.4 A state model represents the host's view of the equipment, and does not necessarily describe the internal equipment operation. When using collection events, all Processing Management state model transitions shall be mapped sequentially into the appropriate internal equipment collection events that satisfy the requirements of those transitions. In certain implementations, the equipment may enter a state and have already satisfied all of the conditions required by the Processing Management state models for transition to another state. In the case, the equipment makes the required transition without any additional actions in this situation.

5.2 *Object Attribute Representation* — The object information models for standardized objects will be supported by an attribute definition table with the following column headings:

Attribute Name	*Definition*	*Access*	*Requirement*	*Form*
The formal text name of the attribute.	Description of the information contained.	RO or RW	Y or N	(see below)

5.2.1 The Access column uses RO (Read Only) or RW (Read and Write) to indicate the access that service-users have to the attribute.

5.2.2 A 'Y' or 'N' in the requirement (Rqmt) column indicates if this attribute must be supported in order to meet fundamental compliance for the service.

5.2.3 The Form column is used to indicate the format of the attribute. (See Section 4.1 for definitions.)

5.3 *Service Message Representation*

5.3.1 *Service Resource Definition* — A service resource definition table defines the specific set of messages for a given service group, as shown in the following table:

Message Service Name	*Type*	*Description*
Message Name	N or R	The intent of the service.

5.3.1.1 Type can be either N = Notification or R = Request.

5.3.1.2 Notification type messages are initiated by the service provider, and the provider does not expect to get a response from the consumer/subscriber.

5.3.1.3 Request messages are initiated by a service consumer or subscriber. Request messages ask for data or an activity from the provider. Request messages expect a specific response message (no presumption on the message content).

5.3.2 *Service Parameter Dictionary* — A service parameter dictionary table defines the parameters used in a service, as shown in the following table:

Parameter	*Form*	*Description*
Parameter X	Data Type	A parameter called X is B in A.

5.3.2.1 A row is provided in the table for each parameter of the service. The first column contains the name of the parameter. This is followed by columns describing the form and contents of the corresponding primitive.

5.3.2.2 The Form column is used to indicate the type of data contained in a parameter. (See Section 4.2 for definitions.)

5.3.2.3 The Description column in the Service Parameter Dictionary table describes the meaning of the parameter, the values it can take on, and any inter-relationships with other parameters.

5.3.2.4 To prevent the definition of numerous parameters named "XxxList," this document adopts the convention of referring to the list as "(List of) Xxx." In this case, the definition of the variable Xxx will be given, not of the list. The term "list" indicates a collection (or set) of zero or more items of the same data type. Where a list is used in both the request and the response, the list order in the request is retained in the response. A list must contain at least one element unless zero elements are specifically allowed.

5.3.3 *Service Message Definition* — A service message definition table defines the parameters used in a service, as shown in the following table:

Parameter	*Req/Ind*	*Rsp/Cnf*	*Description*
Parameter X	(see below)	(see below)	A description of the service.

5.3.3.1 The columns labeled Req/Ind and Rsp/Cnf link the parameters to the direction of the message. The message sent by the initiator is called the "Request." The receiver terms this message the "Indication" or the request. The receiver may then send a "Response," which the original sender terms the "Confirmation."

5.3.3.2 The following codes appear in the Req/Ind and Rsp/Cnf columns and are used in the definition of the parameters (e.g., how each parameter is used in each direction):

M	**Mandatory Parameter** — Must be given a valid value.
C	**Conditional Parameter** — May be defined in some circumstances and undefined in others. Whether a value is given may be completely optional or may depend on the value of the other parameter.
U	**User-Defined Parameter**
-	The parameter is not used.
=	(for response only) Indicates that the value of this parameter in the response must match that in the primary (if defined).

6 Overview

6.1 Processing management is concerned with the processing of material by a processing resource. Its principle function is to ensure that material delivered to the processing resource is processed with the correct recipe. It defines the services needed by a supervisor (service-user) to initiate and track processing of a particular material. It also defines commands which affect the processing operation.

6.1.1 The processing resource is the entity which adds manufacturing value to the material. It may take several forms, including the processing element of a cluster tool process module or the entity managing processing for a complete stand-alone equipment. The processing agent is considered to be the provider of the processing services.

6.1.2 Process management allows for pre-conditioning before material arrival and post-conditioning after material departure. A simple tuning mechanism provides support for limited feedforward and feedback control. The tuning, applied at process initiation, sets recipe variable parameters.

6.1.3 The services are fully defined in terms of the functionality provided by the processing agent (service-provider) and, as such, do not dictate the architecture of the supervisor (service-user).

6.1.4 This standard describes the concepts and processing model on which the communications are based, followed by the detailed behavioral model used. It then describes the standard object attributes and message services in detail.

6.2 *Compliance* — Compliance to this standard includes adherence to all stated requirements in this document where implemented. Standard services are to be used where related functionality is required. This includes defined message services and state models.

6.2.1 Some capabilities are not required to be supported for compliance, such as queuing, multiple concurrent jobs, material groups, manual start, pause/resume, and tuning. Required capabilities are indicated throughout the document and are also listed in the Fundamental Requirements section.

6.2.2 A processing agent shall provide the fundamental requirements, plus the set of optional services, appropriate to achieve effective processing management for the particular hardware architecture and automated processing requirements.

7 Concepts

7.1 *Material Processing Model* — Processing management ensures that the appropriate processing is applied to a particular material by a processing resource through the definition of a process job. The process job provides a widely applicable supervisory control capability for automated processing of material in equipment, irrespective of the particular process being used.

7.1.1 This standard assumes that, given the material and the recipe specification, the processing resource is capable of independently achieving the required processing objectives.

7.1.2 Processing management does not provide services for material movement, but the service-provider does need to coordinate its activities with regard to the receiving and sending of material, thereby maintaining system integrity.

7.2 *Process Job* — The process job is a dynamic object specified by the process supervisor (service-user) to effect material processing by the processing resource. The high-level job contains all the information required by the processing resource to achieve processing of the material, once it arrives, without further intervention by the supervisor.

7.2.1 The process job encompasses up to four sequential phases:

- processing resource pre-conditioning before material arrival,
- material and processing resource preparation for processing,
- material processing, and
- processing resource post-conditioning after material departure.

7.2.2 The material processing phase is the only phase in which the material is altered and is the only required phase.

7.2.3 This standard specifies services for the creation, control (pausing, aborting, etc.) and tracking of the process job. It does not define the low-level control of processing since this is application-dependent. The processing resource performing a process job is responsible for doing whatever is appropriate to achieve its processing objectives, as specified by the recipe and tuning parameters.

7.2.4 The material specified in a process job may be the actual single material elements to be processed or a container, such as in the case of a cassette of wafers.

7.2.5 The process job lifecycle may extend beyond the active processing of the material. It may exist from before material arrival, through setup and processing, and until after material departure. This allows for material processing-related pre-conditioning of the processing resource before the material is received and for processing resource post-conditioning (e.g., cleanup) after material is sent. Pre-conditioning and post-conditioning support is not a fundamental requirement.

7.2.6 The processing resource may provide process job queuing in order to offer flexibility in systems where work is pre-scheduled or the order of material arrival is unknown. Queuing is the acceptance of multiple process jobs in advance of performing the processing activities. Queuing is generally needed to support more

complex systems requiring concurrent and consecutive jobs (see below). The jobs are listed in the queue in the order created. Execution order may be significant, such as consecutive jobs on the same material. Queuing is not a fundamental requirement.

7.3 *Relation to Material Movement* — Processing Management does not provide services for receiving material into the processing agent domain for processing or sending the material away after processing is complete.

7.3.1 Processing depends on the presence of the material, and material departure depends on process completion. There is also an interdependency requiring synchronization with material movement if processing resource pre-conditioning and post-conditioning are applied. The equipment is responsible for maintaining integrity between material transfer and processing.

7.3.2 Material movement management is outside the scope of this standard but may be achieved using applicable SEMI standards.

7.4 *Processing Description* — The description of the processing to be applied by the Process Job is crucial. The description may be supplied in the form of a Process Recipe (see SEMI E42) or a Process Program (see SEMI E30). This specification will define messages referencing only Process Recipes. Where there are special considerations for using Process Programs instead of a Process Recipe, they will be noted.

7.4.1 The process job includes a processing description (a recipe or process program) identifier that shall be unique within the domain of the processing agent. The type and content of the processing description must be appropriate for the processing resource and the type of material.

7.4.2 Creation and management of recipes and process programs is outside the scope of this standard.

7.5 *Process Tuning* — Feedforward and feedback control between process steps is becoming increasingly important for process tuning in stabilizing processes, such as those which lack in-situ metrology, and demand increasing yields (or performance). The process tuning requirements differ considerably with application, and there is little consensus on any particular method. It is not the intention of this standard to provide comprehensive support for process tuning, but rather to provide a simple mechanism which may be extended. Support for recipe tuning is not a fundamental requirement.

7.5.1 Processing management provides a mechanism for specification of the type of recipe method to be applied. Two methods are defined in the standard: RecipeID only, and RecipeID and Variables. User-defined methods may also be used, requiring all communicating entities to have a common understanding of the particular definition and application requirements.

7.5.2 The RecipeID only method accepts the identifier of the recipe to be applied but no additional tuning parameters. The application recipe body is not precluded from defining any application tuning, but there is no standardized support.

7.5.3 The RecipeID and Variables method provides a simple process tuning mechanism at process job creation to support limited run to run feedforward and feedback control. It defines the VariableTuning method, which supplies a list of variable names and values in the process job create. This sets variables defined in the recipe. Each variable name shall be one of the exposed variable definitions supported in recipe management, and its value shall fall within the range specified in the variable definition.

7.5.4 Recipe parameter names (RecipeVarName) shall be specified using the nomenclature defined for ‘Object Specification’ (ObjSpec) in SEMI E39 (Object Services Standard) within the scope of SEMI E40 (Note: see OBJSPEC in SEMI E5). This use of the ObjSpec nomenclature is required to unambiguously identify parameters within some complex recipes (e.g., cluster tool process recipes). If the specification of a recipe parameter is unambiguous, then the form of the ObjSpec may be simplified to just the parameter name. Using the ObjSpec nomenclature for SEMI E40 recipe parameter names requires an object model to describe equipment recipe structure.

7.5.5 Figure 1 shows an example of a typical set of cluster tool hierarchical recipe relationships. A processing parameter might be specified through Sequence Step AB, Process Recipe BB, Process Step CB, to Process Parameter DB. In this example, the object specifier for Process Parameter DB would be:

“Sequence:Erma>SequenceStep:AB>ProcessRecipe:
BB>ProcessStep:CB>ProcessParameter:DB>”

or

“Erma>AB>BB>CB>DB>”

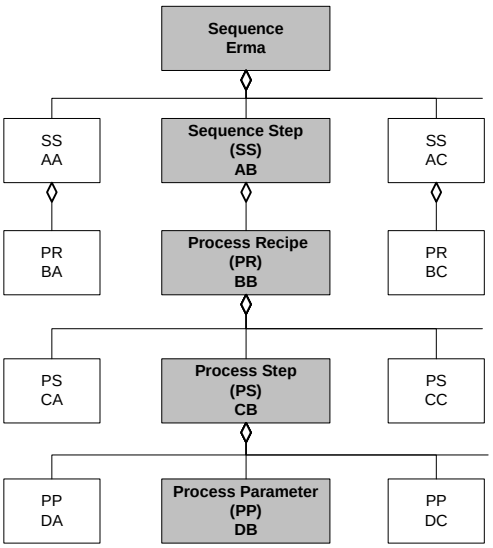

Figure 1
Cluster Tool Sequence Recipe Hierarchical Relationships

7.6 *Processing Material Groups* — Many equipment architectures require concurrent processing of groups of material. A single process job can control a group of material, with certain restrictions. All material in the group needs to be of the same type and processed identically. The processing of each material should be dependent on the arrival of the group. That is, the full material group shall be received by the equipment before processing begins and may not depart until all processing is complete. This ensures that the process job remains a simple logical control mechanism while maintaining robust control, good coordination with material movement, and effective material data tracking.

7.6.1 Two common examples of allowable material group processing are:

a) an equipment receiving a cassette of wafers to be processed identically. All the wafers in the group are received together, in the cassette. The processing of the wafers may or may not be simultaneous but can only proceed once the cassette has arrived. All wafer processing shall complete before the cassette may be removed. Note that this may alternatively be specified as a simple process job with the cassette as the material; and

b) a cluster tool batch process module processing wafers together. Wafers are received singly. Processing of the wafer group in the batch chamber begins when all the wafers specified in the process job have arrived. Wafers are sent after processing is complete.

7.7 *Concurrent Process Jobs* — Concurrent process jobs is the situation where multiple jobs are active (not queued) at the same time. A single process job is not always appropriate for concurrent processing of multiple material. In such cases, multiple concurrent jobs shall be supported by the processing resource. Support of concurrent process jobs is not a fundamental requirement.

7.7.1 An example of concurrent wafer processing which may not be achieved with a single process job is a carousel type cluster tool process module. The processing of a single wafer is not dependent on the arrival of the group. In this case, concurrent process jobs would be created, one for each wafer, even if they were to be processed identically. This allows effective control and tracking of the process jobs.

7.7.2 Processing management considers concurrent process jobs to be logically independent of one other. They are distinguished by unique job identifiers. Concurrent jobs may not apply to the same material because the processing resource may not have more than one active process job associated with a particular material.

7.7.3 There may be interdependencies between concurrent jobs due to resource availability and equipment hardware architecture.

7.8 *Consecutive Process Jobs* — Consecutive process jobs is a situation where multiple process jobs are applied to material while it is in the processing resource. The process jobs requested on the same material are maintained in the order received, and a subsequent job becomes active once the previous one has completed material processing.

7.8.1 A process job normally specifies all the processing to be applied to the material during a single visit to the processing resource. For example, a process job for a cluster tool would specify all the processing in the multiple sequential steps in the various process modules of the tool. Certain situations may require application of subsequent process jobs to material without it leaving the processing resource.

7.8.2 Processing management requires that a subsequent process job on the same material does not interrupt the previous process job processing. The previous process job terminates immediately once it completes active material processing, even though the material has not left the processing resource, and it is superceded by the subsequent job. The material becomes associated with the superceding process job. This allows sequential processing and maintains a single active association between material and process job.

7.9 *Process Job without Material* — This standard is primarily intended for the management of material pro-

cessing. However, it permits application of a process job to a processing resource which contains no processing material. This may be used to achieve processing resource conditioning which is not related to a specific material.

7.9.1 The process job has the normal control characteristics, except that it has no dependency on material arrival and terminates at the end of active processing. Support of process jobs without material is not a fundamental requirement.

8 Behavior

8.1 This section provides a high-level definition of the communications between the supervisor and the processing resource needed to achieve material processing. It does not define the message detail, concentrating instead on the concepts. The message detail is addressed in Section 10, Messaging Services.

8.2 *Process Job Communications*

8.2.1 *Process Job Control Messaging* — The control message flow for normal operation is presented in Figure 1. The arrows represent significant information exchange.

8.2.1.1 A detailed description of each message used in normal operation follows.

8.2.1.2 *PR Job Create* — The supervisor requests that the processing resource perform the specified process job. This request may be acted upon immediately, or queued for later execution if the processing resource is busy or the order of material arrival unknown. If the processing resource does not support queuing or the queue is full, the request may be rejected. The request shall supply a process specification in which the supervisor supplies such information as:

- identification of the material to be processed,
- the recipe defining the processing, and
- whether processing will be started manually (optional, normal operation is automatic).

8.2.1.2.1 Upon receipt of the PR Job Create request and before acknowledging, the processing resource checks the process specifications to ensure that they are valid (i.e., that the specified parameters (recipe, material, manual start) are sufficient) and have legal values for the capabilities of the processing resource. Depending on its ability to queue jobs, it may also check such dynamic information as availability of the processing resource to receive the material or presence of correct material, etc., to determine whether to accept or reject.

8.2.1.2.2 The processing resource sets the process start attribute if automatic start is requested (normal operation).

8.2.1.3 *PR Job Create Acknowledge* — The processing resource informs the supervisor that the requested job is accepted or rejected, and if rejected, supplies error codes and textual reasons for the failure.

8.2.1.4 *PR Job Setup* — The processing resource reports that the process job is active and setting up for process. It may have been on the queue or have just been created. If the material is not already present, the processing resource performs any pre-conditioning required then awaits material arrival. Upon arrival of all the material, it prepares for processing and automatically initiates processing if the process start attribute is set.

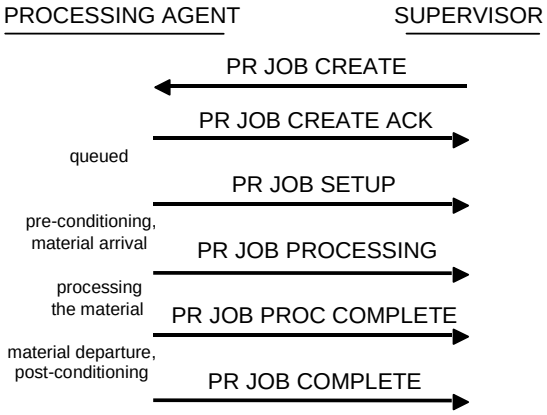

Figure 2
Process Job Message Flow

8.2.1.5 *PR Job Processing* — The processing resource reports that material processing has commenced.

8.2.1.6 *PR Job Processing Complete* — The processing resource reports that material processing is completed and that the material is available for removal.

8.2.1.7 *PR Job Complete* — The processing resource declares the process job to be complete once it has completed processing the material, the material has departed, and any required processing resource post-conditioning has completed. This message is also used when a process job ends abnormally. The message provides information on the success or failure of the processing and, if failed, supplies error codes and textual reasons for the failure.

8.2.2 *Process Job Informational Events* — There are process job related events which may be of significance to the service-user. These are designated as “collection events” and shall be available for event reporting. This section describes those collection events and shows how they fit into the chronology of a process job. These collection events, as specified by the PRJobEvent

service, shall be implemented per definition of one of the following standards, SEMI E30, E40, E53 or similar style events, as required by the service-user. The equipment may also optionally implement the collection events per the remaining standards. However, the service-user shall only utilize one of the standards for collection event implementations. The selection mechanism by the service-user is equipment specific. For example, it may be part of an equipment power-on process or via an equipment specific equipment constant.

8.2.2.1 If SEMI E30, E53 or similar style collection events are used for PRJobEvent, the equipment must also implement events for the process job milestones as defined by the PRJobAlert service.

8.2.2.2 Collection event messages provide valuable information but are not strictly required to perform material processing. Therefore, it is expected that some message protocol implementations will provide a method by which the service-user may disable those events which are not needed in a particular implementation. For SEMI E40 style events, the activation for disabling the events is user specific. For SEMI E30, E53 or similar style events, the activation is as defined in those standards.

8.2.2.3 *PR Job Waiting for Material* — The processing resource reports that all required pre-conditioning has completed and that it cannot proceed until the process job material arrives. It is considered to be awaiting material arrival if it is not aware of activities in progress with the aim of receiving all or part of the material. This event may be generated only during process job setup. This event requires, at a minimum, PRJobID and Timestamp as its data.

8.2.2.4 *PR Job State Change* — The processing resource reports that it has changed state. All state transitions in the state model Figure 4 shall trigger this collection event. These events require, at a minimum, variables PRJobID, PRJobState, and TimeStamp (E30).

8.2.3 *Process Job Extended Messaging* — In this section, the extended messaging of Abort/Stop/Cancel, Pause/Resume and manual Start Process is added to the normal messaging described above. The only extended functionality required to be supported in processing management is Abort.

8.2.3.1 A detailed description of each message used in extended operation follows.

8.2.3.2 *PR Job Abort* — The supervisor may command the processing resource to abort a process job at any time. The goal of the abort command is to end the process job activities as quickly as possible. This includes halting all processing of material in progress, which may result in an unknown material condition. Abort is intended for use when serious problems are detected and further damage needs to be prevented. The abort command terminates the process job. In many cases, error recovery may be required before normal operation may continue and subsequent jobs can be executed. For processing equipment a part of the error recovery procedure may require the removal of substrates belonging to the aborted process job that still reside in the equipment. This is determined by the processing agent, which may use applicable service standards to handle the exception.

8.2.3.2.1 The abort command takes precedence over the stop, cancel, and pause commands. If the specified process job is queued, the abort command acts identically to a cancel command.

8.2.3.3 *PR Job Stop* — The supervisor may command the processing resource to stop a process job at any time. The stop command terminates the job in an orderly manner. The object of the stop command is to cease the current activity at the next safe, convenient point, preserving material integrity. In the situation of processing equipment, this convenient point may require that all related substrates are sent to their output destination. This implies that each material is either processed as specified in the recipe or not at all. As stop terminates the job, a new process job is needed to continue processing the material in the processing resource. If restart of the same job is needed, pause and resume should be used instead of stop. A new job is required if additional processing is needed after a job is stopped.

8.2.3.3.1 If the specified process job is queued, the stop command acts identically to a cancel command.

8.2.3.4 *PR Job Cancel* — The supervisor may cancel a process job which has not yet become active (e.g., a job which is queued). Cancel is used when the supervisor would like to remove a process job — to reschedule, for example — but does not want to affect process job activities already in-progress. A cancelled job is removed from the queue and ceases to exist. No physical action is associated with canceling a process job. If the specified process job is active, the processing resource shall reject this command.

8.2.3.5 *PR Job Pause* — The supervisor may issue a command to pause a process job at any time. A pause command shall cause the processing resource to continue to the first safe, continuable, pausing place and then cease activity. The activity may cease only at points that allow for resumption of the activity (see the resume command) such that material integrity is maintained and the processing goals are accomplished. Note that a paused process job may be aborted or stopped as an alternative to the resume command. In

this case, the stop command may cause the equivalent of a resume in order to ensure material integrity (either fully processed or not processed at all) upon termination of the process job.

8.2.3.6 *PR Job Resume* — The resume command is used to continue a previously paused process job activity.

8.2.3.7 *PR Job Create* — The PR Job Create request as previously decribed is for normal automatic process job operation. If the process start attribute is not set, the processing resource waits for a manual start from the supervisor before processing the material. The control message flow for manual start is presented in Figure 3.

8.2.3.8 *PR Job Waiting for Start* — The processing resource is ready to process once the material has arrived and has been prepared for processing. The processing resource reports that it is ready to process and is waiting for start, if the process job PRProcessStart attribute is not set. The PRProcessStart attribute is not set if the job is defined to start manually and the start command has not yet been received. The WAITING FOR START state shall be a safe condition which maintains material integrity. If the process job is stopped or aborted while in this state, the material will not have been altered.

8.2.3.9 *PR Job Start Process* — The supervisor which has defined a process job to start manually issues a PR Job Start Process to allow processing of the material to proceed when the processing resource is ready. The start may be issued at any time after process job creation. On receiving the start command, the processing resource sets the process start attribute and starts processing if it is already in the WAITING FOR START state.

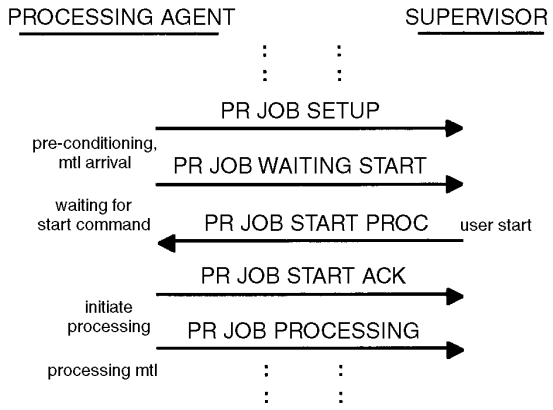

Figure 3
Manual Start Message Flow

8.2.3.10 *PR Job Start Acknowledge* — The processing resource responds to the supervisor that the requested start process is accepted or rejected and, if rejected, supplies errorcodes and textual reasons for failure.

8.3 *Process Job State Model* — The process job is a transient entity. It is created on request of the supervisor, executes, and then is deleted by the processing resource. The job usually spans the time period from shortly before material is physically delivered to the processing resource, through the processing, and until shortly after material is taken away.

8.3.1 *Process Job State Model Diagram* — Figure 4 shows the Process Job State Model diagram.

8.3.2 *Process Job State Descriptions* — The detailed state definitions follow.

8.3.2.1 *ABORTING (ACTIVE Substate)* — While the PR Job is in the ABORTING substate, the processing resource is performing an abort or an optional error recovery procedure. The abort procedure will cause immediate termination of the processing. It is the responsibility of the processing resource to cease physical activity as quickly as possible, having achieved a safe condition.

NOTE 2: For processing equipment the termination may have to be followed by an error recovery procedure with which remaining substrates can be brought to the output destination.

8.3.2.2 *ACTIVE* — ACTIVE is the parent state of all substates where the context of an active process job execution exists.

8.3.2.3 *EXECUTING (ACTIVE Substate)* — EXECUTING is the parent state of those substates that refer to the preparation and execution of a process job.

8.3.2.4 *SETTING UP (EXECUTING Substate)* — While the PR Job is in the SETTING UP substate, the processing resource performs pre-conditioning, awaits material arrival, and prepares for material processing. Pre-conditioning includes all operations in the processing resource, which are required by the recipe in advance of material arrival.

8.3.2.4.1 In cases where the material is already present, it is simply prepared for processing. If pre-conditioning (without material present) is required to achieve the processing goals specified by the recipe, the job fails and terminates.

8.3.2.5 *PAUSE (ACTIVE Substate)* — While the PR Job is in the PAUSE substate the processing resource is suspending or has suspended activity.

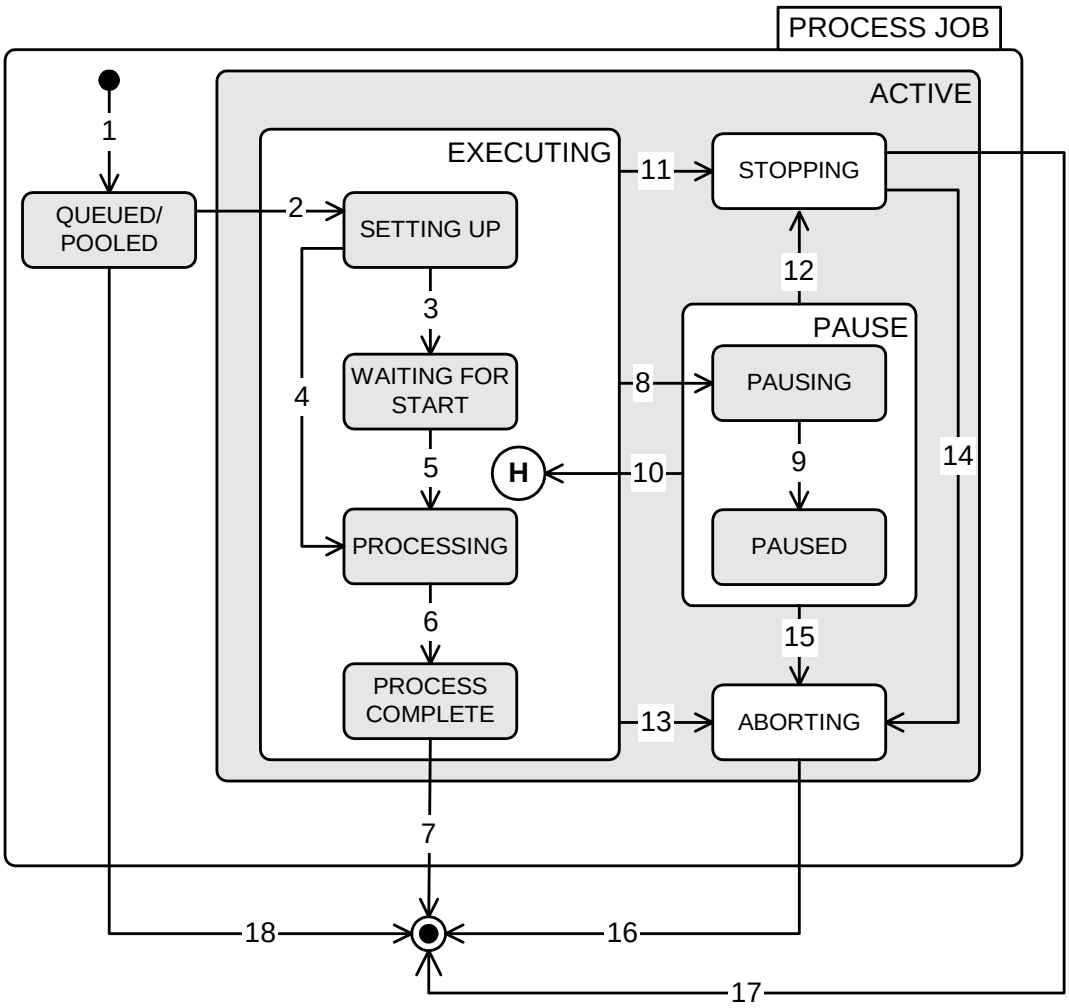

Figure 4
Process Job State Model

8.3.2.6 *PAUSED (PAUSE Substate)* — While the PR Job is in the PAUSED substate all processing resource activity has ceased. The PR Job is awaiting a RESUME (or STOP or ABORT) command.

8.3.2.7 *PAUSING (PAUSE Substate)* — While the PR Job is in the PAUSING substate, the processing resource continues to the first safe, continuable pausing place and then ceases activity. The activity may only cease at points that allow for resumption of the activity such that material integrity is maintained and the processing goals are accomplished.

8.3.2.8 *PROCESSING (EXECUTING Substate)* — While the PR Job is in the PROCESSING substate, the processing resource is doing the actual material processing using the equipment recipe(s) specified by the PR Job.

8.3.2.9 *PROCESS COMPLETE (EXECUTING Substate)* — While the PR Job is in the PROCESS COMPLETE substate the processing resource has completed processing all material specified by the PR Job. When all material removed from the processing resource, processing resource performs any required post-conditioning. Post-conditioning includes all operations in the processing resource after material departure, which are required by the recipe.

8.3.2.9.1 In cases where the process job is superseded by another process job on the same material and post-conditioning is not required, the first job terminates successfully while the material is still present. If post-conditioning is required, the second job may not supersede and remains on the queue.

8.3.2.10 *QUEUED/POOLED* — While the PR Job is in the QUEUED/POOLED substate, the process job has been accepted by the processing resource through a PR

Job Create/Acknowledge transaction (such as PRJobCreate, PRJobCreateEnh, PRJobDuplicateCreate, and PRJobMultiCreate) and is awaiting execution. One or more jobs may be in this state depending upon specific equipment capabilities. That is, if equipment does not support job queuing, then only one PR Job may be in this state at a time. If equipment does support job queuing, then the number of jobs that may be in this state must at least be equal to the number of load ports on the equipment. Advanced equipment job management capabilities require multiple jobs (greater than two) be in this state per load port.

8.3.2.10.1 The order that jobs become active is dependent upon whether the equipment supports job queuing and/or job pooling. Example methods for job activation include FIFO (first-in/first-out) order, material arrival order, and host ordering of jobs (provided by additional services). The equipment may support only one method of selecting jobs for activation, or more than one method allowing only one method to be used at a time or more than one method to be used at a time.

8.3.2.10.2 All process jobs pass through this state. If the processing resource supports queuing/pooling, jobs may remain in this state for prolonged periods. In any case, a process job remains queued/pooled until the material positions (of the processing resource) needed for the process are available or are already occupied by the material to be processed.

8.3.2.11 *STOPPING (ACTIVE Substate)* — While the PR Job is in the STOPPING substate, the processing resource is performing a stop procedure to terminate processing in an orderly manner. It is the responsibility of the processing resource to cease the current activity at the next safe, convenient point, preserving material integrity. For processing equipment this may require sending all related substrates to its output destination. This implies that each material is processed as specified in the recipe or not at all.

8.3.2.12 *WAITING FOR START (EXECUTING Substate)* — The substate WAITING FOR START is used only in manual start process jobs. It is entered once SETUP is complete and a PR Job Start Process has not yet been received by the processing resource. Manual start is defined by the supervisor in PR Job Create.

8.3.2.12.1 The job remains in this state, ready to process the material, until the PR Job Start Process is received or Abort or Stop terminates the job.

8.3.3 *Process Job State Transitions* — The detailed state definitions are defined in Table 1.

Table 1 Process Job State Transition Table

#	*Current State*	*Trigger*	*New State*	*Action(s)*
1	(no state)	The processing resource accepts a Process Job create request.	QUEUED/ POOLED	1. The job is placed the job queue/pool. 2. Acknowledge the Process Job creation.
2	QUEUED/ POOLED	The processing resource has been allocated to the Process Job.	SETTING UP	1. The job is removed from the queue/pool. 2. PR Job Setup event is triggered. 3. All required resource preconditioning is performed. 4. When job material arrives all material preparation is performed.
3	SETTING UP	Job material is present AND the processing resource is ready to start the process job AND PRProcessStart attribute is not set.	WAITING FOR START	PR Job Waiting for Start event is triggered.
4	SETTING UP	Material is present and ready for processing. PRProcessStart attribute is set.	PROCESSING	1. PR Job Processing event is triggered. 2. Material is processed.
5	WAITING FOR START	Job Start directive	PROCESSING	1. PR Job Processing event is triggered. 2. Material is processed.

#	*Current State*	*Trigger*	*New State*	*Action(s)*
6	PROCESSING	Material processing completed.	PROCESS COMPLETE	1. PR Job Processing Complete event is triggered. 2. The processing resource performs all required resource post-conditioning. 3. Await material departure.
7	PROCESS COMPLETE	Job material departed the processing resource AND resource post-conditioning completed, OR superceded by another process job on the same material.	(no state)	1. PR Job Complete event is triggered. 2. The process job is deleted.
8	EXECUTING	The processing resource initiated a process pause action. (it received a PAUSE command or initiated an internal pause)	PAUSING	The processing resource pauses at the first convenient time.
9	PAUSING	The processing resource paused the job.	PAUSED	None.
10	PAUSE	The processing resource resumed the job.	EXECUTING	The processing resource resumes the activity that was paused.
11	EXECUTING	The processing resource initiated a process stop action. (it received a STOP command or initiated an internal stop)	STOPPING	The processing resource stops the current execution activity at the first convenient time.
12	PAUSE	The processing resource initiated a process stop action. (it received a STOP command or initiated an internal stop)	STOPPING	The processing resource stops the current execution activity at the first convenient time.
13	EXECUTING	The processing resource initiated a process abort action. (it received an ABORT command or initiated an internal abort)	ABORTING	The processing resource terminates the current execution activity immediately.
14	STOPPING	The processing resource initiated a process abort action. (it received an ABORT command or initiated an internal abort)	ABORTING	The processing resource terminates the current execution activity immediately.
15	PAUSE	The processing resource initiated a process abort action. (it received an ABORT command or initiated an internal abort)	ABORTING	The processing resource terminates the current execution activity immediately.
16	ABORTING	The processing resource abort procedure is complete and for some processing equipment the related substrates are moved out as part of the error recovery.	(no state)	1. PR Job Complete event is triggered. 2. The process job is deleted.
17	STOPPING	The processing resources stop procedure completed.	(no state)	1. PR Job Complete event is triggered. 2. The process job is deleted.
18	QUEUED/ POOLED	“CANCEL,” “ABORT,” or “STOP” command received.	(no state)	1. Remove the process job from the queue/pool 2. PR Job Complete event is triggered. 3. Delete the process job.

9 Object Definitions

9.1 Processing management defines one standard object, the Process Job.

9.2 *Process Job Object Definition* — The process job is a dynamic object created by the processing resource as requested by the supervisor. It tracks progress of the operations required and is deleted by the processing resource automatically upon completion. The process job is uniquely identified by the PRJobID attribute. The object attribute notation used in the table below is described in Conventions, Section 5.2.

9.2.1 The attributes in Table 2 shall be accessible using Object Services Standard (SEMI E39).

Table 2 Process Job Attributes

Attribute Name	*Definition*	*Rqmt*	*Access*	*Form*
ObjID	An identifier for the service user. It is set when the process job is created.	Y	RO	Text
ObjType	The object type.	Y	RO	Text: "PROCESSJOB"
PauseEvent	List of event identifiers that cause the equipment to automatically transition to the PAUSING/PAUSED states when one of the listed events is triggered.	N	RO	List of: EventID
PRJobState	A unique sub-state of the job according to the process job state model in Figure 4.			Enumerated: QUEUED/POOLED SETTING UP WAITING FOR START PROCESSING PROCESS COMPLETED EXECUTING PAUSING PAUSE STOPPING ABORTING
PRMtlNameList	List of identifiers of the material being processed.	Y	RO	List of: PRMtlName
PRMtlType	Identifies the type of material being processed.	Y	RO	Enumerated
PRProcessStart	Indicates that the processing resource start processing immediately when ready.	N	RO	Boolean: TRUE — Automatic start FALSE — Manual start
PRRecipeMethod	Indication of recipe specification type, whether using is applied and which method is used.	Y	RO	Enumerated: Recipe only Recipe with Variable Tuning
RecID	Identifier of the recipe applied.	Y	RO	Text
RecVariableList	List of variables supporting a recipe method.	N	RO	List of: RecipeVariable

9.2.2 A number of the ProcessJob attributes are composite data types. The constituent data is defined in Table 3.

Table 3 Attribute Data Definitions

Data Itentifier	*Description*	*Form*
PRMtlName	Textual identifier of the material being processed.	Text
RecipeVariable	Variables supporting a recipe method.	Structure composed of: RecipeVarName RecipeVarValue
RecipeVarName	The name of the recipe variable.	Text
RecipeVarValue	Value of the recipe variable.	Depends on variable

10 Messaging Services Detail

10.1 This section defines the messaging services required to implement the processing management concepts. The messages were introduced in Section 8.1. These services are independent of the messaging protocol used. They may be mapped to SECS-II (SEMI E5) or to other comparable protocols.

10.1.1 These messaging services define the messages to be used, the nature of the parameters contained within the messages, and data type of the parameters. Not defined here is the internal structure of the actual messages as transferred, including order of the parameters and how various data structures and data types are represented.

10.1.2 The service message notation used in the tables below is described in Conventions, Section 5.3.

10.2 *Service List* — The following messages are exchanged between host and equipment for the purpose of accomplishing processing management tasks.

Table 4 Service List

Message Name	*Type*	*Description*
PRGetAllJobs	R	Get a list of the jobs and their states for all jobs which have not completed.
PRGetSpace	R	Get the number of jobs which can currently be created on the resource.
PRJobAlert	N	Notification by the processing resource that the process job is setting up, processing, completed process, or that the job is completed.
PRJobCommand	R	Command which affects the process job.
PRJobCreate	R	Supervisor (service-user) request that a process job be performed.
PRJobCreateEnh	R	User request for job to be done. User assigns a unique job identifier.
PRJobDequeue	R	Removes (deletes) process job(s) from the queue.
PRJobDuplicateCreate	R	Create a set of similar process jobs. User assigns unique job identifiers.
PRJobEvent	N	Notification by the processing resource that a process-related event has occurred.
PRJobMultiCreate	R	Create several jobs which may be dissimilar. User assigns unique job identifiers.
PRJobSetRecipeVariable	R	User request for setting a new value to one of more recipe variable parameters.
PRJobSetStartMethod	R	Create a set of similar process jobs. User assigns unique job identifiers.
PRSetMtrlOrder	R	Request the service to use a specific methodology for processing order.

10.3 *Parameter Dictionary*

Table 5 Parameter Dictionary, Part 1

Parameter Name	*Definition*	*Form: Possible Values*
CmdParameter	Parameter supporting a command type.	Structure composed of: CmdParmName CmdParmValue
CmdParmName	The name of the parameter.	Text
CmdParmValue	Value of the parameter.	Varies per parameter
ErrorCode	Contains the code for the specific error found.	Enumerated: *PRJobCreate, PRJobCreateEnh, PRDuplicateCreate, PRJobMultiCreate:* • Parameters improperly specified • Insufficient parameters specified • Unsupported option requested • Busy (no queueing or queue full) *PRJobCreateEnh, PRDuplicateCreate, PRJobMultiCreate:* • Object identifier in use *PRJobCommand:* • Parameters improperly specified • Insufficient parameters specified • Unsupported option requested

Parameter Name	*Definition*	*Form: Possible Values*
		• Command not valid for current state *PRJobComplete:* • No material altered • Material partially processed • All material processed • Recipe specification related error • Failed during processing • Failed while not processing • Failed due to lack of material • Job aborted • Job stopped • Job cancelled
ErrorText	Text in support of the error code.	Text
PRAck	Indicates whether the activity was successful.	Boolean: TRUE — Successful FALSE — Unsuccessful
PRCmdName	Indicates which process job command to perform.	Text: ABORT STOP CANCEL PAUSE RESUME STARTPROCESS
PREventData	Data related to the specific event.	Varies per parameter reported
PREventID	Identifier of the specific event which occurred.	Enumerated: Unique collection event ID for Waiting for Material and Process Job State Change events.

Table 6 Parameter Dictionary, Part 2

Parameter Name	*Definition*	*Form: Possible Values*
PRJobID	The unique identifier for a process job. It can be accessed as the ObjID attribute of the process job. The host may provide this identifier to the processing resource. In this case, the host must guarantee uniqueness of job identifiers for all the process jobs in the PROCESS JOB STATE in the equipment.	Text.
PRJobList	List of process job identifiers and their states.	List of Structure PRJobID PRJobState
PRJobMilestone	Process job milestone.	Enumerated: PR Job Setup PR Job Processing PR Job Processing Complete PR Job Complete PR Job Waiting for Start
PRJobSpace	Used to indicate the number of jobs that can currently be created for the processing resource.	Integer

Parameter Name	*Definition*	*Form: Possible Values*
PRJobState	A unique state of the process job according to the process job state model.	Enumerated: QUEUED/POOLED SETTING UP WAITING FOR START PROCESSING PROCESS COMPLETE PAUSING PAUSED STOPPING ABORTING
PRMtlName	Textual identifier of the material being processed.	Text: Unique for each material with respect to the processing agent.
PRMtlType	Identifies the type of material being processed.	Enumerated.
PRMtrlOrder	Defines the order by which material in the process jobs material list will be processed.	Enumeration: ARRIVAL – process whichever material first arrives. OPTIMIZE – process in an order that maximizes throughput. LIST – follow the order in the list.
PRPauseEvent	Variable containing information which is transferred to the corresponding PRJob attribute. Shall conform to event identifiers as defined in either SEMI E30 or E53.	(list of) text
PRProcessStart	Indicates that the processing resource start processing immediately when ready.	Boolean: TRUE — Automatic Start FALSE — Manual Start
PRRecipe	Specification of the process job recipe.	Structure composed of: PRRecipeMethod RecID (List of) RecipeVariable
PRRecipeMethod	Indication of recipe specification type, whether tuning is applied and which method is used.	Enumerated: Recipe only Recipe with VariableTuning
PRStatus	Reports the acceptance or rejection of a requested operation.	Structure composed of: PRAck (List of) Status
RecID	Identifier of the recipe applied.	Text: Unique with respect to the processing agent.
RecipeVariable	Variables supporting a recipe method.	Structure composed of: RecipeVarName RecipeVarValue
RecipeVarName	The name of the recipe variable.	Text: Depends on recipe
RecipeVarValue	Value of the recipe variable.	Depends on variable
Status	Reports any errors found.	Structure composed of: ErrorCode ErrorText
Timestamp	Event date and time.	Text: yyyymmddhhmmsscc

10.4 *Service Detail* — The tables below define the parameters for each service. In some cases, parameters have additional detail which is defined in the parameter definition section.

10.4.1 *PRJobCreate*

Table 7 PRJobCreate Service Detail

PRJobCreate Service Detail Section

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
PRJobID	-	M	The processing agent assigns the unique identifier which is used in all subsequent process job communications.
PRMtlType	M	-	
(List of) PRMtlName	M	-	All material shall be of the same material type. This is an ordered list and indicates the order in which the process job should process the material, if it is single wafer processing equipment.
PRRecipe	M	-	
PRProcessStart	M	-	Indicates auto or manual start.
PRStatus	-	M	

PRRecipe Parameter Detail Section

PRRecipeMethod	M	-	
RecID	M	-	The process job recipe identifier shall be unique within the domain of the processing agent.
(List of) Recipe Variable	C	-	Parameters required depend on the recipe method selected.

PRStatus Parameter Detail Section

PRAck	-	M	Indication of acceptance to perform the job.
(List of) Status	-	C	Error information, required if PRAck is unsuccessful.

10.4.2 *PRJobCreateEnh*

Table 8 PRJobCreateEnh Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
PRJobID	M	M	User supplied Job ID. Must be unique among jobs known by the processing resource or else the resource shall reject the create request.
PRMtlType	M	-	May be NULL when no material is processed.
(List of) PRMtlName	M	-	An ordered list that associates a set materials with process conditions (process programs or recipes).
PRRecipe	M	-	This is a structure.
PRProcessStart	M	-	Indicates auto or manual start.
PRPauseEvent	M	-	If null, then processing will not be automatically paused.
PRStatus	-	M	Indicates success or failure.

10.4.3 *PRJobDuplicateCreate* — This service creates multiple process jobs. Each job will be identical (a duplicate) in terms of doing the same processing on all the material that must be identical. This means the same values of PRRecipe and PRProcessStart are applied to each job that is created by this service. Creates a series of process jobs useful for single wafer processing.

Table 9 PRJobDuplicateCreate Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
(List of) PRJobDupSpec	M	C	Ordered list of user supplied Job ID's and material. Structure: PRJobID PRMtlName
PRMtlType	M	-	Enumeration that Indicates type of material.
PRRecipe	M	-	This is a structure.
PRProcessStart	M	-	Indicates auto or manual start.
PRPauseEvent	M	-	If null, then processing will not be automatically paused.
(list of) PRJobID	-	C	Shall be used if PRJobSpec is not returned.
PRStatus	-	M	Indicates success or failure.

10.4.4 *PRJobMultiCreate* — This service creates multiple process jobs. Each job can be created uniquely.

Table 10 PRJobMultiCreate Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
(List of) PRJobMultiSpec	M	C	An ordered list of job specifications as follows: (list of) Structure: PRJobID PRRecipe PRProcessStart (List of) PRMtlName
PRMtlType	M	-	Enumeration that indicates type of material.
(list of) PRJobID	-	C	Shall be used if PRJobSpec is not returned.
PRStatus	-	M	Indicates success or failure.

10.4.5 *PRJobDequeue* — Remove one or more jobs from the queue. PRStatus shall indicate any jobs which could not be removed because they either did not exist or were in the PR JOB ACTIVE state.

Table 11 PRJobDequeue Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
PRJobList	M	-	(List of) PRJobID
PRStatus	-	M	Indicates success or failure.

10.4.6 *PRJobCommand* — All of the process job commands described in Section 8.2.3 are communicated using the PRJobCommand service. The commands are Abort, Stop, Cancel, Pause, Resume, and Start Process. This standard does not specify any required parameters. Abort is the only command which is required to be supported.

Table 12 PRJobCommand Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
PRJobID	M	-	Identifies the process job on which to perform the command.
PRCmdName	M	-	
(List of) CmdParameter	C	-	Dependent on the command selected.
PRStatus	-	M	

10.4.7 *PRJobAlert* — Notification of process job milestones achieved by the processing resource are communicated using the PRJobAlert service. Process job milestones, which are described in Section 8.2.1, are events which are important to the control and tracking of the process job. The milestones required to be supported are PR Job Setup, PR Job Processing, PR Job Processing Complete, and PR Job Complete. An additional milestone, PR Job Waiting for Start, is used with the manual start option.

Table 13 PRJobAlert Service Detail

Parameter	*Req/Ind*	*Comment*
Timestamp	M	
PRJobID	M	Identifies the process job on which the milestone has been reached.
PRJobMilestone	M	
PRStatus	M	

10.4.8 *PRJobEvent* — Process job informational event notification, which is described in Section 8.2.2, is communicated using the PRJobEvent service. These are defined for Waiting for Material and Process Job State Change events. Support for informational events is not required.

Table 14 PRJobEvent Service Detail

Parameter	*Req/Ind*	*Comment*
PREventID	M	
Timestamp	M	
PRJobID	M	Identifies the process job which generated the event.
PREventData	C	

10.4.9 *PRJobSetRecipeVariable* — Sends a request to change the settings for a list of recipe variable parameters. Implementation of this service is optional.

Table 15 PRJobSetRecipeVariable Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
PRJobID	M	-	
RecVariableList	M	-	(list of) RecipeVariable
PRStatus	-	M	Indicates success or failure. Failure is if a variable can't be set. A List of variables which could not be set is returned in the status.

10.4.10 *PRJobSetStartMethod* — Sends a request to change the start method for job(s). This request will fail if a specified job is not in the QUEUED/POOLED state. Implementation of this service is optional.

Table 16 PRJobSetStartMethod Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
PRJobList	M	-	(List of) PRJobID
PRProcessStart	M	-	Indicates auto or manual start.
PRStatus	-	M	Indicates success or failure.

10.4.11 *PRGetAllJobs* — This message shall return a list containing job identifiers and the associated states of those jobs for all jobs which have not completed.

Table 17 PRGetAllJobs Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
PRJobList	-	M	

10.4.12 *PRGetSpace* — This message shall return the remaining number of jobs that can be created for the processing resource.

Table 18 PRGetSpace Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
PRJobSpace	-	M	

10.4.13 *PRSetMtrlOrder* — Request the Processing Management Service to use a specific strategy for the order in which materials are processed.

Table 19 PRSetMtrlOrder Service Detail

Parameter	*Req/Ind*	*Rsp/Cnf*	*Comment*
PRMtrlOrder	M	-	Sets the value for the strategy the service will use.
PRAck	-	M	Indicates success or failure.

10.5 *Mapping of Semantics to Syntax* — Table 20 provides the correspondence between the message semantics defined in Section 8.2 and the syntax as defined in Section 10.4. The use of '.req', '.rsp', or '.nfy' suffixes shows the direction of message flow. '.req' is a message request from the service user to the service provider. '.rsp' is a response message from the service provider to the service user. '.nfy' is a notification from the service provider to the service user.

Table 20 Correspondence of Message Semantics to Syntax

Parameter	*Comment*
PR Job Create	PRJobCreate.req
PR Job Create Acknowledge	PRJobCreate.rsp
PR Job Setup	PRJobAlert.nfy (PRJobMilestone=PR Job Setup)
PR Job Processing	PRJobAlert.nfy (PRJobMilestone=PR Job Processing)
PR Job Processing Complete	PRJobAlert.nfy (PRJobMilestone=PR Job ProcessingComplete)
PR Job Complete	PRJobAlert.nfy (PRJobMilestone=PR Job Complete)
PR Job Waiting for Material	PRJobEvent.nfy (PREventID=Waiting for Material)
PR Job State Changes	PRJobEvent.nfy (PREventID=Process Job State Change)
PR Job Abort	PRJobCommand.req (PRCmdName=ABORT)
PR Job Stop	PRJobCommand.req (PRCmdName=STOP)
PR Job Cancel	PRJobCommand.req (PRCmdName=CANCEL)
PR Job Pause	PRJobCommand.req (PRCmdName=PAUSE)
PR Job Resume	PRJobCommand.req (PRCmdName=RESUME)
PR Job Waiting for Start	PRJobAlert.nfy (PRJobMilestone=PR Job Waiting for Start)
PR Job Start Process	PRJobCommand.req (PRCmdName=STARTPROCESS)
PR Job Start Acknowledge	PRJobCommand.rsp (PRSatus.PRAck=TRUE)

11 Variable Data

11.1 The purpose of this section is to define the list of variable data requirements for Processing Management compliant equipment. Values of these variables are available to the host via collection event reports.

11.2 *Variable Data Definitions* — The identifier and all other attributes of the ProcessJob object shall be available for inclusion in event reports associated with it. The following attributes are most likely to be used: PRJobID, PRJobState, RecID, RecVariableList and PRMtlNameLIst.

12 Compliance

12.1 Processing management defines the standard services available to achieve job-based material processing in equipment. The capabilities supported allow flexible management of automated processing encompassing many process types. Only a subset of these capabilities may be needed for a particular implementation.

12.2 *Fundamental Requirements* — All processing agent implementations shall support the fundamental requirements. These have been indicated in the appropriate sections of the document and are listed together below:

- Create and execute a single process job to completion, given:
  - a single material of the appropriate type, uniquely identified.
  - a unique recipe identifier for which the corresponding recipe can be found.
- Report the process job milestones: Setup, Processing, Processing Complete, and Job Complete.
- Detect and report the success or failure of the process job, indicating complete, partial, or non-processing of the material.
- Support Abort of the process job at all times, immediately ceasing activity and terminating the process job.
- Maintain the data of required process job attributes indicated in Table 2.
- Reject requests with incomplete or invalid parameters.
- Reject requests for capabilities not supported.
- Implement the services and messages with the exception of those required for the Optional capabilities.

12.2.1 Satisfying fundamental requirements may not provide sufficient flexibility or performance for some equipment. In such cases, fundamental functionality should be supplemented by optional capabilities as appropriate to the needs of the system.

12.3 *Additional Capabilities* — Optional capabilities defined or enabled in this standard include:

- Processing resource pre-conditioning and post-conditioning.
- Stop, Pause, and Resume of a process job.
- Manual process start.
- Process job queuing and Cancel on a queued job.
- Process tuning.
- Processing of material groups.
- Multiple concurrent process jobs.
- Multiple consecutive process jobs in a single visit.
- Process job with no material.
- Notification of waiting for material and of process job state changes.
- Implement PRJobCreateEnh, PRJobMultiCreate, and PRJobDuplicateCreate.

12.3.1 The services are defined with mechanisms to reject unsupported services and options should they be requested. This improves robustness and enables sophisticated service-users to adjust their requests to the capabilities of the particular processing agent.

12.4 Table 21 provides a checklist for Processing Management (PM) compliance.

Table 21 PM Compliance Statement

Fundamental PM Requirements	*PM Section*	*Implemented*	*PM Compliant*
Single Process Job Execution	8.2	☐ Yes ☐ No	☐ Yes ☐ No
Process Job Milestones	8.2.1 (except 8.2.1.2,3)	☐ Yes ☐ No	☐ Yes ☐ No
Process Job Failure Indication	8.2.1.7	☐ Yes ☐ No	☐ Yes ☐ No
Abort Command	8.2.3.2	☐ Yes ☐ No	☐ Yes ☐ No
Process Job Object Implementation	8.3, 9	☐ Yes ☐ No	☐ Yes ☐ No
Reject Invalid/Incomplete Parameters	8.2.1.2.1	☐ Yes ☐ No	☐ Yes ☐ No
Reject Unsupported Capabilities	11.2	☐ Yes ☐ No	☐ Yes ☐ No
Services Implementation (not per Additional)	10	☐ Yes ☐ No	☐ Yes ☐ No
Additional PM Capabilities	*PM Section*	*Implemented*	*PM Compliant*
Resource Pre/Post-conditioning		☐ Yes ☐ No	☐ Yes ☐ No
Stop, Pause and Resume Commands	8.2.3.3,4,5	☐ Yes ☐ No	☐ Yes ☐ No
Manual Process Start	8.2.3.9	☐ Yes ☐ No	☐ Yes ☐ No
Process Job Queuing	8.3	☐ Yes ☐ No	☐ Yes ☐ No
Process Tuning	7.5, 10.4.9	☐ Yes ☐ No	☐ Yes ☐ No
Processing of Material Groups	7.6	☐ Yes ☐ No	☐ Yes ☐ No
Multiple Concurrent Process Jobs	7.7	☐ Yes ☐ No	☐ Yes ☐ No
Multiple Consecutive Process Jobs	7.8	☐ Yes ☐ No	☐ Yes ☐ No
Process Job with No Material	7.9	☐ Yes ☐ No	☐ Yes ☐ No
Event Notification	8.2.2	☐ Yes ☐ No	☐ Yes ☐ No
Enhanced Job Creation	10.4.2	☐ Yes ☐ No	☐ Yes ☐ No
Multiple Job Creation	10.4.3	☐ Yes ☐ No	☐ Yes ☐ No
Duplicate Job Creation	10.4.4	☐ Yes ☐ No	☐ Yes ☐ No

SEMI E40.1-1101
SECS-II SUPPORT FOR PROCESSING MANAGEMENT STANDARD

This standard was technically approved by the Global Information & Control Committee and is the direct responsibility of the North American Information & Control Committee. Current edition approved by the North American Regional Standards Committee on August 27, 2001. Initially available at www.semi.org September 2001; to be published November 2001. Originally published in 1996; previously published July 2001.

1 Purpose

1.1 This document maps the services and data of SEMI E40 to SECS-II streams and functions and data definitions.

2 Scope

2.1 This is the standard way to implement the Processing Management, which provides remote control of wafer processing, using the SECS-II message protocol.

2.2 This standard does not purport to address safety issues, if any, associated with its use. It is the responsibility of the users of this standard to establish appropriate safety and health practices and determine the applicability of regulatory limitations prior to use.

3 Referenced Standards

3.1 *SEMI Standards*

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

SEMI E40 — Standard for Processing Management

4 Terminology

4.1 None.

5 Mapping of Processing Services

Table 1 Processing Management Messages Mapping

Service Message Name	*Stream, Function*	*SECS-II Name*
PRJobCreate request	S16F3,F4	Process Job Create Request/Acknowledge
PrJobCommand request	S16F5,F6	Process Job Command Request/Acknowledge
PRJobAlert notify	If E30 style events: S6F11,F12 If E40 style alerts: S16F7,F8 If E53 style events: S6F11,F12 S6F13,F14	If E30 style events: Event Report Send/Acknowledge If E40 style alerts: Process Job Alert Notify/Confirm If E53 style events: Event Report Send/Acknowledge Annotated Event Report Send/Ack
PRJobEvent notify	If E30 style events: S6F11,F12 If E40 style events: S16F9,F10 If E53 style events: S6F11,F12 S6F13,F14	If E30 style events: Event Report Send/Acknowledge If E40 style events: Process Job Event Notify/Confirm If E53 style events: Event Report Send/Acknowledge Annotated Event Report Send/Ack
PRSetMtrlOrder request	S16F29,30	Process Job Set Material Order
PRJobCreateEnh	S16F11/F12	PRJobCreateEnh
PRJobDuplicate-Create	S16F13/F14	PRJobDuplicateCreate
PRJobMultiCreate	S16F15/F16	PRJobMultiCreate
PRJobDequeue	S16F17/F18	PRJobDequeue
PRGetAllJobs	S16F19/F20	PRGetAllJobs
PRGetSpace	S16F21/F22	PRGetSpace
PRJobSetRecipe-Variable	S16F23/F24	PRJobSetRecipeVariable
PRJobSetStart-Method	S16F25/F26	PRJobSetStartMethod

6 Mapping of Processing Parameter

Table 2 Data Item Mapping

Service Parameter	*SECS-II Data Item*
PrJobID	PRJOBID
PRMtlType	MF
PRMtlName	MID
PRAck	ACKA
PRRecipeMethod	PRRECIPEMETHOD
RecID	RCPSPEC
RecipeVarName	RCPPARNM
RecipeVarValue	RCPPARVAL
PRProcessStart	PRPROCESSSTART
PRCmdName	PRCMDNAME
PRJobMilestone	If E30 style events: CEID If E40 style alerts: PRJOBMILESTONE If E53 style events: CEID
PRJobState	PRSTATE
Timestamp	TIMESTAMP
PREventID	If E30 style events: CEID If E40 style events: PREVENTID If E53 style events: CEID
CmdParmName	CPNAME
CmdParmVal	CPVAL
PRMtrlOrder	PRMTRLORDER
ErrorCode	ERRCODE
ErrorText	ERRTEXT
PREventData	V (SV, ECV, DVVAL)
PRJobSpace	PRJOBSPACE
PRPauseEvent	PRPAUSEEVENT

7 Variable Data Item Mapping

7.1 This section shows the specific SECS-II data classes, and formats needed for SECS-II implementations of SEMI E40 variable data. According to SEMI E40 section 11, all ProcessJob object attributes are to be available as variables for Process Job state transition events. These variables will be of SEMI E5 data item DVVAL.

8 Implementation Details

8.1 *Use of Object Services* — Several capabilities of the Processing Management Services are accessed through the Object Services Standard. Whan a Process Job has been created, (PRJOBID is valid), then its attributes can be read and written using the Object Services GetAttr and SetAttr messages.

8.1.1 E39 Object Services shall be used for access to ProcessJob attributes. The GetAttr service may be used for all ProcessJob attributes and the SetAttr service may be used on only those attributes whose Access is set to RW.

8.2 *Multi-Block Messages* — Processing Management Services is protocol independent and therefore, makes no mention of SECS-II multi-block access and grant messages. When these Service use the SECS-II protocol, then S16F3,F5 shall be preceded by an S16F1/S16F2 access request/grant message exchange when the message will be multi-block.

9 SECS-II Attribute Definitions

9.1 *Process Job Object SECS-II Attributes Definitions* — The following are the SECS-II structure definitions for the E40 ProcessJob object.

Table 3 Process Job SECS-II Attribute Definitions

Attribute Name	*Attribute Data Form: SECS-II Structure*
"ObjID"	<PRJOBID> (Conforms to the restrictions of ObjID as specified in SEMI E39.1, Section 6.)
"ObjType"	"ProcessJob"
"PauseEvent"	L,n n=number of collection events 1. <$CEID_1$> … n. <$CEID_n$> CEID restricted to format U()
"PRJobState"	<PRSTATE> PRJobState PRSTATE enumerated as follows: 51 (U1) Enumerations: 0 – QUEUED/POOLED 1 – SETTING UP 2 – WAITING FOR START 3 – PROCESSING 4 – PROCESS COMPLETE 5 – (Reserved) 6 – PAUSING 7 – PAUSED 8 – STOPPING 9 – ABORTING
"PRMtlNameList"	When MF = 13 (0x0d) (carriers) L,n n=number of carriers 1. L,2 1. <$CARRIERID_1$> 2. L,j j=number of slots 1. <$SLOTID_1$> : j. <$SLOTID_j$> : n. L,2 1. <$CARRIERID_n$> 2. L,k k=number of slots 1. <$SLOTID_1$> : k. <$SLOTID_k$> When MF = 14 (0x0e) (substrate) L,n n=number of material (substrate) 1. <MID_1> substrate ID … n. <MID_n> MID restricted to format A
"PRMtlType"	<MF> MF restricted to format B **MF Value** **Description** 13 (0x0d) carriers (e.g. FOUP, SMIF pod, cassette) 14 (0x0e) substrate (e.g. wafer, mask, flat panel) other not valid for E40 material

Attribute Name	*Attribute Data Form: SECS-II Structure*
"PRProcessStart"	<PRPROCESSSTART>
"PRRecipeMethod"	<PRRECIPEMETHOD>
"RecID"	<RCPSPEC>
"RecVariableList"	L,n n=number of recipe variables 1. L,2 1. <$RCPPARNM_1$> 2. <$RCPPARVAL_1$> … n. L,2 1. <$RCPPARNM_n$> 2. <$RCPPARVAL_n$>