4.2.3.5 *Scenario*

Host Initiates Trace Report:

```
COMMENT                                    HOST  EQUIPMENT                          COMMENT

Trace Data initialization
requested                     S2,F23-->
                                      <--S2,F24     Acknowledge, trace initiated
                                                    [DO] TOTSMP REPGSZ times
                                                    [DO] REPGSZ many times: collect
                                                     SVID_1,...SVID_n data, delay
                                                     time by DSPER.
                                                    [END_DO]
                                      <--S6,F1      Send SV_1,...SV_n
Acknowledge receipt            S6,F2-->
                                                    [END_DO]
Optional:
Request trace termination prior to
completion (TOTSMP = 0)       S2,F23-->
                                      <--S2,F24     Acknowledge premature termination
```

4.2.4 *Limits Monitoring* — This capability relates to the monitoring of selected equipment variables and has three primary aspects:

— Defines a standard set of monitoring zones and limits.

— Provides for reporting to the host when selected equipment variables transition between monitoring zones.

— Empowers the host to modify the values of the variable limit attributes for these same selected equipment variables.

4.2.4.1 *Purpose* — The limits monitoring capability provides the host a means of monitoring equipment conditions by a flexible, efficient, and asynchronous method which is consistent across equipment. It eliminates the need for constant polling of equipment by the host for current status values. Further, this capability allows the host to implement changes in the monitoring range as needed. This capability has application to both production operation and diagnostic/testing scenarios, and it also has applicability to statistical process control.

4.2.4.2 *Definitions*

LimitVariable — DVVAL containing the VID of a specific equipment variable for which a zone transition collection event has been generated.

EventLimit — DVVAL containing the LIMITID of the limit crossed by LimitVariable.

TransitionType — DVVAL which defines the direction of the zone transition which has occurred: 0 = transition from lower to upper zone, 1 = transition from upper to lower zone.

Limit — Used in this section to represent the set of variable limit attributes that completely describe a variable monitoring "barrier." The attributes include VID, Units, UPPERDB, LOWERDB, LIMITMAX, and LIMITMIN. In some contexts it may be interpreted more narrowly as the combination of UPPERDB and LOWERDB.

$LIMITID_n$ — Refers to the identifier of a specific limit (as defined by UPPERDB and LOWERDB) among the set of limits for a monitored equipment variable. LIMITIDs are consecutively numbered, beginning at one through the number of limits possible (seven minimum).

Monitoring Zone — A subset of the possible range of values for a variable of interest to the host. A single limit divides the range into two zones. Multiple limits may be combined to divide the range even further.

Zone Transition — The movement of a variable value from one monitoring zone to another. This transition is a collection event and has a corresponding CEID.

Deadband — An overlap of two zones implemented to prevent constant zone transitions by a variable sitting on or near a limit (i.e., "chattering").

UPPERDB — A variable limit attribute that defines the upper boundary of the deadband of a limit.[18] The value applies to a single limit (LIMITID) for a specified VID.

18 The format and units must be the same as the format of the variable being monitored.

Thus, UPPERDB and LOWERDB as a pair define a limit.

LOWERDB — A variable limit attribute that defines the lower boundary of the deadband of a limit.[18] The value applies to a single limit (LIMITID) for a specified VID. Thus, UPPERDB and LOWERDB as a pair define a limit.

UPPER ZONE — The range of values lying above a limit.

LOWERZONE — The range of values lying below a limit.

LIMITMAX — The maximum value for any limits of a specific equipment variable. This value is set by the equipment manufacturer and typically coincides with the maximum value allowed for the monitored variable.

LIMITMIN — The minimum value for any limits of a specific equipment variable.[19] This value is set by the equipment manufacturer and typically coincides with the minimum value allowed for the monitored variable.

Undefined — When used in reference to variable limits, it indicates that monitoring/reporting of zone transitions involving that particular limit are disabled.

4.2.4.3 *Description* — The limits monitoring capability provides the host with a minimum of seven configurable limits or barriers that may be applied to selected equipment status variables (SV's) of the types floating point, integer, and boolean. When one of these barriers is crossed, a collection event is generated to alert the host to a change in monitoring zone or state of the monitored variable. These seven limits may be combined in a variety of ways to match the needs of the host system.[19] An illustration of a combination of five of the limits to provide one type of variable monitoring is shown in Figure 4.2.1.[20] This section describes the key aspects of limits monitoring. Detailed implementation examples of limits monitoring are provided as Application Note A.7.

NOTE 6: While the SEMI E5 standard allows SV's to be lists, such variable lists are not allowed under this capability.

4.2.4.3.1 *Monitoring Limit Characteristics* — A limit is defined by a set of attributes that include the variable (VID) to which the limit corresponds, the units of that variable, the maximum and minimum possible values of the limit (LIMITMAX and LIMITMIN) and the specific borders of the limit (UPPERDB and LOWERDB). See Figure 4.2.2. There is a limitation to the values of UPPERDB and LOWERDB which may be stated as:

LIMITMAX≥UPPERDB≥LOWERDB≥LIMITMIN

A limit divides the possible range of variable values into two parts, the upper zone and the lower zone. At any time, the monitored variable is considered to be in one and only one of these zones. However, as Figure 4.2.2 shows, these two zones have an area of overlap. This is called the deadband.

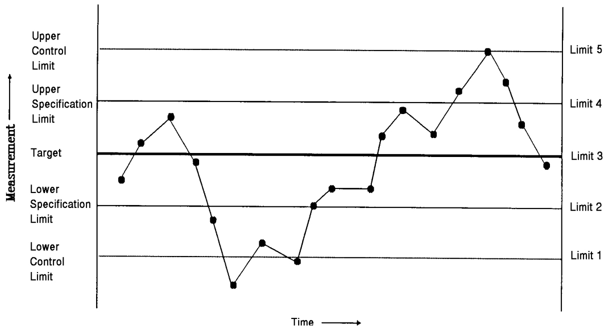

Figure 4.2.1
Limit Combination Illustration: Control Application

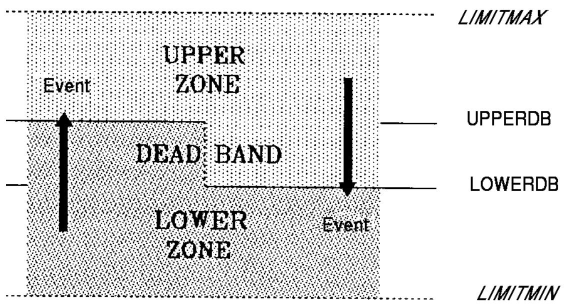

Figure 4.2.2
Elements of One Limit

The deadband is a key concept of limits monitoring, especially for floating point variables. Its purpose is to prevent a phenomenon known as chattering — the repeated changing of zones due to small, rapid fluctuations in variable value while near the zone boundary. In practice, the value of a variable must reach the opposite boundary of the deadband before a zone transition can occur. Thus, if a variable's value

19 Note that while at least seven limits per variable are available from the equipment, the host need not use all seven.

20 This illustration shows the reading which might be available to the equipment, not the limit excursions reported to the host. Reporting is covered later in the section.

reaches the UPPERDB and transitions into the upper zone, it will not return to the lower zone until it falls back to the LOWERDB. The difference between UPPERDB and LOWERDB should always be greater than the typical amplitude of those fluctuations deemed to be insignificant. In some cases, the width of the deadband may set to zero (i.e., UPPERDB = LOWERDB). At first glance, this would seem to make indeterminate the current zone when an integer value sits on the limit. This is not the case, however, when movement of the value is considered. To illustrate, an example is given, assuming that UPPERDB = LOWERDB = 100. The list shows consecutive readings of the variable and the resultant zone:

99	Lower Zone (Initial Reading)
101	Upper Zone (Zone Transition)
100	Lower Zone (Zone Transition)
100	Lower Zone
99	Lower Zone
100	Upper Zone (Zone Transition)

Transition from one zone into another generates a collection event, as might be reported via S6,F11. The host has the option of receiving notification by enabling event reporting for the event. For each variable that has monitoring capability, one CEID is reserved to indicate zone transitions for that variable. To aid in the determination of the nature of a transition event, three DVVAL's have been defined:

LimitVariable — The VID of the monitored variable to which the collection event refers.

EventLimit — Contains the LIMITID of the limit reached or crossed by LimitVariable.

TransitionType — Defines the direction of the zone transition which has occurred: 0 = transition from lower to upper zone, 1 = transition from upper to lower zone.

Sampling frequency is an important element of limits monitoring and should be considered during equipment specification. If changes in variable value are relatively fast compared to sampling frequency, it is possible for some zone transitions to be missed or for multiple zone transitions to occur between readings. Since it is possible for zone transitions to occur "simultaneously" or for limits to be identically defined, the DVVAL EventLimit has been defined to allow for a list of multiple zone transitions of a variable to be reported with a single collection event.

It also should be emphasized that a single CEID is used to report transitions in both directions across a limit. Thus, reporting for one direction but not the other cannot be configured.

The functionality of each limit for each variable can be described with the state model shown in Figure 4.2.3. Below, the three states are described more fully, followed by a table defining the transitions.

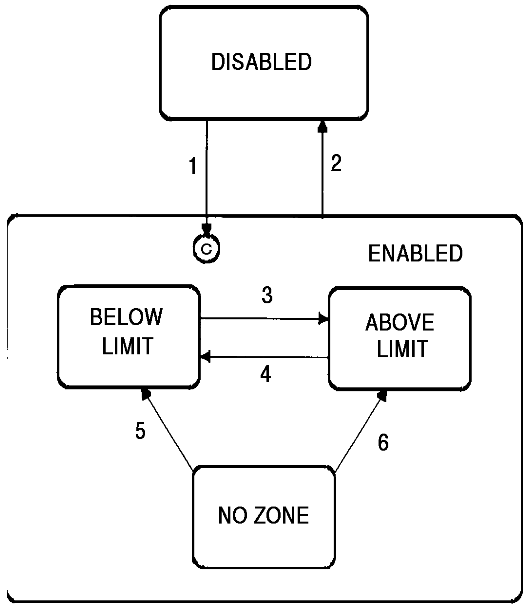

Figure 4.2.3
Limit State Model

ABOVE LIMIT

A variable is considered to be above a limit when its value increases to equal or exceed the upper boundary of the deadband, UPPERDB. The significance attached to this state is a function of the host's usage.

BELOW LIMIT

A variable is considered to be below a limit when its value decreases to equal or fall below the lower boundary of the deadband, LOWERDB. The significance attached to this state is a function of the host's usage.

NO ZONE

In some circumstances it is possible for the variable value to be in neither the upper zone nor the lower zone. This may occur upon definition of a new limit or upon equipment startup when the value of the variable lies in the deadband. In this case, the active state of the limit is considered to be NO ZONE. The limit shall remain in this state until the variable value reaches either boundary of the deadband.

4.2.4.3.2 *Modification of Limit Values* — Values for the monitoring limits on any monitored variable may be modified by the host using the message transaction S2,F45/F46 (Define Variable Limit Attributes). The equipment must reject any S2,F45 message which contains limit information which conflicts with the following rules:

— LIMITMAX≥UPPERDB≥LOWERDB≥LIMIT-MIN;

— If either UPPERDB or LOWERDB is defined, both must be defined; if either UPPERDB or LOWERDB is undefined, both must be undefined.

The first rule is defined and graphically depicted in Figure 4.2.2. The second rule refers to the host's ability to turn any limit "on" or "off". While a minimum of seven limits must be available for each monitored variable, it will be common for the host application to require less than seven or even none of the limits be used. The limits not needed can be disabled by leaving the values for UPPERDB and LOWERDB "undefined". Limits may be disabled for a VID or for all monitored VIDs by using zero length lists in the S2,F45 message.

All monitored variables must be one of three types: integer, floating point, or Boolean. This may be accomplished by using the following formats: 11, 20, 3(), 4(), 5().

NOTE 7: The binary format is not allowed. If the ASCII format is used, the equipment shall perform a conversion into one of the numeric types before performing any value comparisons, both for limit validations and zone transitions.

4.2.4.3.3 *Limit Values Request* — The host may request the current limit values for a specified VID using the message transaction S2,F47/F48 (Variable Limit Attribute Request).

4.2.4.4 *Requirements*

— A minimum of seven limits per monitored variable must be available.

— One CEID per monitored variable must be supplied for zone transition reporting.

— All limit definitions must be kept in non-volatile storage.

— The equipment must enforce the limit validation rules defined above.

— The specification and documentation of which variables may be monitored with this capability is the responsibility of the equipment manufacturer based on the specific instance of equipment. This subject also may be addressed by equipment models of classes of semiconductor equipment.

Table 4.2 Limit State Transition Table

#	*Current State*	*Trigger*	*New State*	*Action*	*Comment*
1	DISABLED	Limit attributed defined w/ S2,F45.	ENABLED	None	The substate of ENABLED is determined by the current value of the monitored variable.
2	ENABLED	Limit attributes set to undefined w/ S2,F45.	DISABLED	None	None
3	BELOW LIMIT	Variable Increased to be ≥UPPERDB	ABOVE LIMIT	None	Zone Transition.
4	ABOVE LIMIT	Variable decreases to be ≤LOWERDB	BELOW LIMIT	None	Zone Transition.
5	NO ZONE	Variable decreases to be ≤LOWERDB	BELOW LIMIT	None	Zone Transition.
6	NO ZONE	Variable increases to be ≥UPPERDB	ABOVE LIMIT	None	Zone Transition.

4.2.4.5 *Scenarios* — Zone Transition Event occurs in equipment:

```
COMMENTS                         HOST   EQUIPMENT                                  COMMENTS

Send enabled event report to host as shown in Section 4.2.1.
```

Host defines Limit Attributes:

```
COMMENTS                         HOST   EQUIPMENT                                  COMMENTS

[IF] S2,F45 is Multi-block
[THEN] Send Multi-block inquire S2,F39-->
                                        <--S2,F40    Multi-block grant.
[END IF]
Host defines new variable limit
attributes.                     S2,F45-->
                                        <--S2,F46    Equipment acknowledges host
                                                     request.
```

Host queries equipment for current Limits:

```
COMMENTS                         HOST   EQUIPMENT                                  COMMENTS

Host queries equipment          S2,F47-->
for current variable limit
attribute definitions.
                                        <--S2,F48    Equipment returns report
                                                     containing requested variable
                                                     limit attribute values.
```

4.2.5 Status Data Collection

4.2.5.1 *Purpose* — This capability allows the host to query the equipment for selected status information and is useful in synchronizing with equipment status.

4.2.5.2 *Definitions*

Status variable value (SV) — A data item containing the value of a status variable. See SEMI E5 for a full definition of this data item.

Status variable ID (SVID) — A unique identifier of a status variable. See SEMI E5 for a full definition of this data item.

4.2.5.3 *Detailed Description* — The host may query equipment status by specifying the desired SVID's. Upon such a request, the equipment sends the host the value of the selected status variables. The host also may request the description (name and units) of any or all available status variables.

4.2.5.4 *Requirements*

— The equipment manufacturer must provide unique SVID's for the various status variables (SV) available for data collection in the equipment.

— All status data is available for status data collection. See Section 5.2, Variable Item List, for a list of status variables.

— All SV's must contain valid data whenever provided to the host.

4.2.5.5 *Scenario*

Request Equipment Status Report:

```
COMMENT                           HOST   EQUIPMENT                                   COMMENT

Host requests report of selected
status variable values            S1,F3-->
                                         <--S1,F4     Equipment responds with the
                                                      requested status variable
                                                      data.
```

Request Equipment Status Variable Name list:

```
COMMENT                           HOST   EQUIPMENT                                   COMMENT

Host requests equipment to identify
selected status variables.       S1,F11-->
                                         <--S1,F12    Equipment responds with the
                                                      requested status variable
                                                      descriptions.
```

4.2.6 *On-line Identification*

4.2.6.1 *Purpose* — Implementation of SEMI E5 (a GEM Fundamental Requirement) requires the equipment to accept the S1,F1 from the Host at any time while it is ONLINE and COMMUNICATING, and respond with S1,F2. The On-line Identification capability describes the host-initiated scenario. The equipment-initiated scenario is used for a different purpose and is defined in sections 3.3 and 4.12 describing the GEM "Control Model".

4.2.6.2 *Definitions*

Equipment Model Type (MDLN) — ASCII string containing the equipment model. See SEMI E5 for a full definition of this data item.

Equipment Software Revision Code (SOFTREV) — ASCII string containing the equipment software revision. See SEMI E5 for a full definition of this data item.

4.2.6.3 *Detailed Description* — On-line Identification allows the host to verify the presence and identity of the equipment.

4.2.6.4 *Requirements* — (Host-Initiated) An S1,F2 response from the equipment must provide MDLN and SOFTREV information which reflects the hardware and software configuration of the equipment.

SOFTREV must uniquely identify different releases of equipment software. Any change in equipment software must result in a corresponding change to SOFTREV.

The equipment-initiated S1,F1 is not required except as described in sections 3.3 and 4.12 describing the GEM "Control Model".

4.2.6.5 *Scenario:*

Host Initiated

```
COMMENT                           HOST   EQUIPMENT                                   COMMENT

Are you there?                    S1,F1-->
                                         <--S1,F2     Equipment replies with MDLN and
                                                      SOFTREV.
```

4.3 *Alarm Management* — The alarm management capability provides for host notification and management of alarm conditions occurring on the equipment.

4.3.1 *Purpose* — Historically, a precise definition of an equipment alarm has been absent. Consequently, differing interpretations have resulted in inconsistent implementations. This is addressed by providing a more rigorous definition (see definition in Section 4.3.2 below) of an alarm.

In addition, it is often important for equipment to report more extensive information to the host than has been available in the S5,F1/F2 (Alarm Report Send/ Acknowledge) transaction. The data required in such cases is very dependent on equipment type, host information requirements, and alarm situation. This issue is addressed by providing event reporting methods that are tied to alarm state changes.

Lastly, the alarm management capability provides mechanisms for:

— Reporting the time of an alarm state change,

— Uploading a list of alarm texts,

— Enabling and disabling the notification of specific alarms, and

— Host query of alarms set and enabled status on the equipment.

4.3.2 *Definitions*

Alarm — An alarm is related to any abnormal situation on the equipment that may endanger people, equipment, or material being processed. Such abnormal situations are defined by the equipment manufacturer based on physical safety limitations. Equipment activities potentially impacted by the presence of an alarm shall be inhibited.

Note that exceeding control limits associated with process tolerance does not constitute an alarm nor do normal equipment events such as the start or completion of processing.

AlarmsEnabled — Status value consisting of a list of the alarm ID's currently enabled for reporting to the host. See SEMI E5 for a full definition of this variable data item.

AlarmsSet — Status value consisting of a list of the alarm ID's currently in the ALARM SET (or unsafe) state. See SEMI E5 for a full definition of this variable data item.

ALCD — Alarm code data item used in the S5,F1 (Alarm Report Send) and S5,F6 (List Alarm Data) messages. This code is divided into two parts, the alarm set/cleared bit and the 7 bit alarm category code. Only the set/cleared bit is used—bit 8 = 1 means alarm set, = 0 means alarm cleared. The alarm category code is not used. See SEMI E5 for a full definition of this data item.

ALID — Alarm identifier. See SEMI E5 for a full definition of this data item.

ALTX — Data item contained in the S5,F1 and S5,F6 messages containing a brief textual description of an alarm. See SEMI E5 for a full definition of this data item.

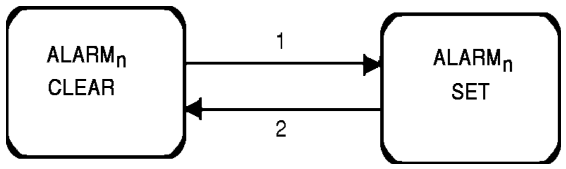

Figure 4.3
State Diagram for Alarm ALID_n

4.3.3 *Detailed Description* — Two alarm notification mechanisms are defined to achieve the flexibility necessary for the reporting required by host systems. First, stream 5 alarm reporting enables a brief, yet fixed, method for notification of alarm occurrences using the S5,F1/F2 transaction. Second, to address the host's potential need for more extensive and flexible data reporting, two collection events ("alarm set" and "alarm cleared") are defined for each possible alarm condition on the equipment to allow the use of event data collection mechanisms. In the latter case, reports are sent by the equipment using the Event Report/ Acknowledge transaction (see Section 4.2 on event data collection).

The alarm_n detected and cleared events are derived from the state model for an alarm (see Figure 4.3 and Table 4.3.1). In this model each of n alarms can be in one of two possible states, either ALARM CLEAR or ALARM SET. The transition from the ALARM CLEAR to the ALARM SET state is defined as the collection event "Alarm_n detected" (transition 1). Conversely, the transition from ALARM SET to ALARM CLEAR is defined as the collection event "Alarm_n cleared" (transition 2).

NOTE 8: The alarm capability is intended as an addition to standard safety alarms (e.g., lights, horns). There is no intent to replace direct operator notification of such problems, nor is there the expectation that the host can prevent or directly address such alarms.

Table 4.3.1 Alarm State Transition Table

#	*Current*	*Trigger*	*New State*	*Action*	*Comment*
1	ALARM_n CLEAR	Alarm_n is detected on the equipment.	ALARM_n SET	Initiate local actions (if any) to ensure safety. Update AlarmsSet and ALCD_n values. Generate and issue alarm message if enabled.	Inhibited activities require operator or host intervention prior to resuming.
2	ALARM_n SET	Alarm_n is no longer detected on the equipment.	ALARM_n CLEAR	Update AlarmsSet and ALCD_n values. Generate and issue alarm message if enabled.	Inhibited activities require operator or host intervention prior to resuming.

The equipment manufacturer is responsible for identifying all alarms on their equipment by:

— Applying the above alarm definition,

— Consulting Application Note A.3 for examples of alarms for various equipment configurations,

— Noting that the presence of an alarm typically requires some action or intervention before resuming safe operation of the equipment, and by

— Referring to Table 4.3.2 below which delineates the differences between events and alarms.

Table 4.3.2

EVENT	*ALARM*
Any occurrence detectable by the equipment.	Related to only those occurrences that are abnormal, undesirable, AND endanger people, equipment, or physical material being processed.
Certain events may trigger a state transition(s).	Each alarm has an associated two-state state model: ALARM SET (or unsafe) and ALARM CLEAR (or safe).
Equipment activities are not necessarily inhibited by the occurrence of an event (unless it is associated with an alarm or intentional inhibit).	The presence of an alarm inhibits equipment activities to ensure safe operation until the alarm condition is cleared.
Certain events may occur in an expected sequence.	Alarms may occur at any time.

4.3.3.1 *Enable/Disable Alarms* — Upon request from the host, the equipment shall enable or disable reporting of certain alarms. Enabling or disabling a given alarm shall impact the communication of both the alarm set and clear messages equally (i.e., turn them both on or both off). This is not the case for enabling/disabling of the associated collection events, where the alarm-set and alarm-cleared events can be enabled and disabled separately. The current enable/disable settings must be stored in non-volatile memory. Changes to the enable/disable settings must be reflected in the AlarmsEnabled status value.

NOTE 9: The alarm itself is not being enabled or disabled, but the reporting of the alarm through SECS-II messages is being enabled or disabled.

4.3.3.2 *Send Alarm Report* — Upon detecting a change in the status of a given alarm, an associated alarm state model shall transition to the opposite state. Following initiation of local actions necessary to ensure safety, the equipment must update the AlarmsSet and associated ALCD values and send an alarm message and/or an event message to the host assuming one or both are enabled. Alarm messages must be sent before the corresponding event messages if both are enabled.

NOTE 10: The ALCD is divided into two parts, the alarm set/cleared bit and the 7 bit alarm category code. Only the set/cleared bit is used—bit 8 = 1 means alarm set, = 0 means alarm cleared. The alarm category code is not used.

4.3.3.3 *List Alarm Text* — Upon request from the host, the equipment sends values of alarm text associated with a specified list of alarm ID's using the S5,F5/F6 (List Alarms Request/Data) transaction.

4.3.3.4 *List Currently Set Alarms* — The host may obtain a listing of currently set alarms by employing any data collection method specified by GEM and referencing the variable data item called "AlarmsSet" (e.g., including " AlarmsSet in a report definition). When reported, AlarmsSet must contain a list of all currently set alarms. Each alarm set or clear occurrence must cause a change to the AlarmsSet status value prior to reporting it to the host.

4.3.3.5 *List Currently Enabled Alarms* — The host may obtain a listing of currently enabled alarms by employing any data collection method specified by GEM and referencing the variable data item called "AlarmsEnabled" (e.g., including AlarmsEnabled in a report definition). When reported, AlarmsEnabled must contain a list of all alarm ID's currently enabled for reporting. The equipment must update the value of AlarmsEnabled upon corresponding change(s) to the enable/disable settings.

4.3.4 *Requirements*

— A set of alarms relating to the physical safety limits of operator, equipment, or material being processed must be defined for the equipment by the equipment manufacturer.

— The equipment must maintain all enable/disable states and report definitions for alarms and collection events in non-volatile memory.

— Each alarm defined on the equipment must have a brief description of its meaning, an associated unique alarm identifier (ALID), alarm text (ALTX), an alarm status (ALCD) and two unique collection event identifiers (CEIDs), one for set and one for cleared.

— Enabled alarm reports must be sent prior to corresponding enabled event reports.

4.3.5 *Scenarios*

NOTE 11: Consult event reporting sections of this document for descriptions of enabling, disabling, and sending collection event reports.

Enable/Disable Alarms:

```
COMMENTS                         HOST   EQUIPMENT                            COMMENT

Enable/disable Alarm             S5,F3-->
                                        <-- S5,F4    Acknowledge
```

Upload Alarm Information:

```
COMMENTS                         HOST   EQUIPMENT                            COMMENT

Request alarm data/text          S5,F5-->
                                        <-- S5,F6    Send alarm data/text
```

Send Alarm Report:

```
COMMENTS                         HOST   EQUIPMENT                            COMMENT

Alarm occurrence detected by the equipment.
                                        <-- S5,F1    Send alarm report (if enabled).
Acknowledge                      S5,F2-->
                                        <-- S6,F11   Send event report (if enabled).
Acknowledge                     S6,F12-->
```

4.4 *Remote Control*

4.4.1 *Purpose* — This capability provides the host with a level of control over equipment operations.

4.4.2 *Definitions*

Host Command Parameter (CPNAME/ CPVAL/CEPVAL) — A parameter name/value associated with a particular host command (S2,F41/S2,F49). The equipment manufacturer must provide unique names (CPNAMEs) for any supported command parameters. Command parameters are not specified in this document but are left to equipment manufacturers to define. Equipment models of specific classes of semiconductor equipment also may address this issue. Note that if there are no associated parameters a zero-length list is sent. The data item CEPVAL, which can be defined as a list, allows grouping of related parameters within a main parameter. If the CEPVAL is defined as a single (non-list) item, then it is the equivalent of a CPVAL.

The uses of OBJSPEC in the header structure of the S2,F49 Enhanced Remote Command allows the equipment supplier to define a set of unique identifiers for different objects within the equipment such as: equipment sub-systems, sub-system components, processing stations, ports, and exchange stations.

4.4.3 *Description* — The equipment responds to host commands that provide the following functions relative to individual equipment implementations:

— Start processing

— Select a process program or recipe

— Stop processing

— Temporarily suspend processing

— Resume processing

— Abort processing

Additional commands may be implemented by the equipment manufacturer (e.g., vent chamber, clear material, open door).

Remote commands shall be interpreted as "request action be initiated" rather than "do action." The equipment may then respond via S2,F42/S2,F50 with HCACK = 4 if the command "is going to be performed." This alleviates any transaction timeouts for commands that may take a long time to perform. The completion of the action initiated by the remote command (i.e., HCACK = 0 or 4) must result in either a state transition or other action that generates a collection event upon normal/abnormal completion.

The format for all remote commands is ASCII, with a maximum length of 20 characters. The character set is restricted to the printable characters (hexadecimal 21 through 7E). Note that spaces are not allowed.

The following remote commands (RCMDs), if implemented on the equipment, shall be supported as described below (see Section 3.4 for a description of Equipment Processing States).

NOTE 12: The terms "current cycle" and "safe break point" used below are to be defined by the supplier or within the models of classes of semiconductor equipment.

START — This command is available to the host when a process program or recipe has been selected and the equipment is in the "ready" processing state. The START command instructs the equipment to initiate processing. Variable parameter settings may be included as name/value command parameters CPNAME/CPVAL/CEPVAL.

PP-SELECT — This command instructs the equipment to make the requested process program(s) available in the execution area. The process programs (PPIDs) are specified via the command parameter list. A status variable (PPExecName) contains the PPID of the process program(s) currently selected.

RCP-SELECT — This command uses the Enhanced Remote Command S2,F49 to instruct the equipment to prepare the requested recipes for execution in the execution area. The recipes and variable parameters are specified via command parameter lists. Each recipe specification may be accompanied by new variable parameter settings, if any, in the command parameter list. A status variable RcpExecName contains the recipe specifiers or identifiers of the recipes currently selected.

STOP — Command to complete the current cycle, stop in a safe condition and return to the "idle" processing state. Stop has the intent of stopping the process. The equipment is not required to support the continuation of processing. Stop leaves material either fully processed or partially processed so that the processing can be later completed. For example, for a single wafer process tool, five wafers have been processed while the remaining wafers remain unprocessed.

PAUSE — Command to suspend processing temporarily at the next safe break point. Pause has the intent of resuming the process at the same point where it was paused. The process may be RESUMED, STOPPED, or ABORTED while in a PAUSED condition. RESUME shall be able to continue the process from the same point where it was paused.

RESUME — Command to resume processing from the point where the process was paused.

ABORT — Command to terminate the current cycle prior to its completion. Abort has the intent of immediately stopping the process and is used because of abnormal conditions. Abort makes no guarantee about the subsequent condition of material. In the above example, the wafers being processed at the time of the abort may not be completely processed. Other AbortLevels > 1 may be defined by the manufacturer or addressed by models of specific classes of semiconductor equipment.

CPNAME = AbortLevel, CPVAL = 1 means terminate current cycle at the next "safe break point," retrieve all material, stop in a safe condition and return to the idle state in the processing state machine.

4.4.4 *Requirements*

— The following Remote Commands, as defined under Descriptions, must be implemented on equipment to satisfy minimum requirements for this capability:

  — START

  — STOP

— The RCMD value for all commands supported on the equipment must be recognized if sent with all upper-case characters (e.g., "STOP", "START", "PP-SELECT", "PAUSE", etc.). In addition to accepting strings with all upper-case characters, the equipment can optionally accept strings with all lower-case characters or mixed-case strings. The equipment documentation should describe whether or not the optional lower-case or mixed-case strings are supported.

— Stream 2 currently provides for Host Command Send and Enhanced Remote Command. The equipment shall support one or both methods, based on appropriateness.

— The Enhanced Remote Command is used to address size, complexity, or the need to target a specific sub-system within the equipment, (i.e., processing station, port, exchange station, material handler, chamber).

4.4.5 *Scenarios*

4.4.5.1 *Host Command Send Scenario*

```
COMMENTS                          HOST   EQUIPMENT                           COMMENT

Host Command                 Send S2,F41-->
                                         <-- S2,F42   Host Command Acknowledge

                                                      [IF] Command Accepted
                                                      (HCACK = 0 or 4)

                                         <-- S6,F11   [THEN] Event Report-state
                                                      change or other collection
                                                      event occurrence.
Event Report Acknowledge          S6,F12-->
```

4.4.5.2 *Enhanced Remote Command Scenario*

```
COMMENTS                       HOST   EQUIPMENT                          COMMENT

Enhanced Remote Command       S2,F49-->
                                      <--S2,F50    EnhancedRemote Command
                                                   Acknowledge

                                                   [IF] Command Accepted
                                                   (HCACK = 0 or 4)

                                      <--S6,F11    [THEN] Event Report-state
                                                   change or other collection
                                                   event occurrence.

Event Report Acknowledge      S6,F12-->
```

4.5 *Equipment Constants*

4.5.1 *Purpose* — This capability provides a method for the host to read and to change the value of selected equipment constants on the equipment.

4.5.2 *Definitions* — None.

4.5.3 *Description* — This capability allows the host to reconfigure equipment constants to support a variety of situations. The following functions are included:

Host Sends Equipment Constants — Allows the host to change the value of one or more equipment constants.

Host Equipment Constant Request — Allows the host to determine the current value of equipment constants.

Host Equipment Constant Namelist Request — Allows the host to retrieve basic information about the equipment constants available at the equipment.

4.5.4 *Requirements*

— Equipment constants must be stored in non-volatile memory.

— The equipment must be in a “safe” condition to accept new constant(s) settings as defined by the equipment manufacturer.

— The equipment must provide a collection event to alert the host whenever an equipment constant is changed by the operator. Information indicating which constant was changed shall be available for the event report.

4.5.5 *Scenarios*

Host Sends Equipment Constants:

```
COMMENTS                       HOST   EQUIPMENT                          COMMENTS

Host sends                    S2,F15-->
equipment constants                   <-- S2,F16   EAC = 0 equipment sets
                                                   constants
```

Host Equipment Constants Request:

```
COMMENTS                       HOST   EQUIPMENT                          COMMENTS

Host constant request         S2,F13-->
                                      <-- S2,F14   Equipment constant data

NOTE: This capability also can be accomplished using S6,F19 & S6,F20. See Section
4.2.
```

Host Equipment Constant Namelist Request:

```
COMMENTS                         HOST   EQUIPMENT                            COMMENTS

Host constant namelist         S2,F29-->
request                                <-- S2,F30   Equipment constant namelist
```

Operator Changes Equipment Constant

```
COMMENTS                                 HOST  EQUIPMENT                     COMMENTS

                                                    Operator changes equipment
                                                    constant at equipment
                                                    operator console.
                                       <-- S6,F11   Equipment reports equipment
                                                    constant change.
Host acknowledges event        S6,F12-->
```

4.6 *Process Program Management* — Process programs and recipes must be managed through interaction between the equipment and the host system.

4.6.1 *Purpose* — Process program management provides a means to transfer process programs or recieps, and to share the management of those process programs or recipes, between the host and equipment.

4.6.2 *Definitions*

PPError — A text data value with information about verification errors of a process program that failed verification. If the equipment provides an event for recipe verification and/or recipe verification failure, then PPError shall be a DVVAL. Otherwise, PPError shall be an SV.

PPFormat — A variable (SV) indicating the type or types of process programs and recipes that are supported.

1 = Unformatted process programs

2 = Formatted process programs

3 = Both unformatted and formatted process programs

4 = Execution recipes

5 = Large unformatted process programs

6 = Large formatted process programs

7 = Both large unformatted and large formatted process programs

8 = Large execution recipes

and a combination of these formats. See SEMI E5 for a compete list.

4.6.2.1 *Definitions for Process Programs*

Process Program — A process program is the pre-planned and reusable portion of the set of instructions, settings, and parameters under control of the equipment that determine the processing environment seen by the manufactured object and that may be subject to change between runs or processing cycles.

Process Program Identifier — A text string (PPID) used to identify a process program.

Formatted Process Program — A process program that is presented as an ordered sequence of command codes with their associated parameters as dictated by S7,F23, and S7,F26. Where formatted process programs are supported, equipment must also provide information sufficient to allow a user at the host to create, display, modify, and partially verify their contents (for example, that information provided in S7,F22).

Unformatted Process Program — An unformatted process program is transferred without structure as the single data item PPBODY (refer to SEMI E5 for a complete description of PPBODY).

Process Program Change Event — The collection event associated with the occurrence of the creation, modification, or deletion of a process program by the operator.

PPChangeName — A data value (DVVAL) containing the PPID of the process program affected by the event Process Program Change Event. See SEMI E5 for a full definition of this variable data item.

PPChangeStatus — The action taken on the process program named by PPChangeName. This variable is valid for the collection event Process Program Change Event. See SEMI E5 for a full definition of this variable data item.

PPExecName — The status variable containing the PPID(s) of the currently selected process program(s). See SEMI E5 for a full definition of this variable data item.

PP-SELECT — The remote command used to select one or more process programs for execution. The process programs are specified by PPID via the command parameter list.

Process Program Verification — Verification is syntax checking of a process program. Verification ensures only that a process program is structured correctly. It does not ensure that the program has the correct parameters to run a particular process or product (see Process Program Validation). Equipment supporting unformatted process programs should provide a variable DVVAL PPError that provides information to the user concerning the error or errors when an attempt to verify a process program fails.

NOTE 14: It may not be possible for the equipment to verify unformatted process programs other than to check the size of the program and internal program checksums. Equipment has no standard means of indicating the type of error encountered in an unformatted process program.

Process Program Validation — Validation is type-and-range checking of parameters in a process program, and is performed after verification.

4.6.2.2 *Definitions for Recipes*

Execution recipe — A type of recipe stored by the equipment for purposes of editing, verification, and execution.

For complete definitions of execution recipes and their standard attributes, see SEMI E42, Section 6.

Execution Recipe Change Event — The collection event associated with the occurrence of the modification or deletion of an execution recipe stored by the equipment.

Note 15: A recipe is modified whenever its body is changed.

New Execution Recipe Event — The collection event associated with the creation of a new execution recipe at the equipment.

Object form recipe — A recipe with body in a proprietary format that may be presented without structure.

RcpChangeName — A data value (DVVAL) containing the identifier of the recipe affected by the event Execution Recipe Change Event or New Execution Recipe Event. See the SEMI E5 Standard for a full definition of this variable data item.

RcpChangeStatus — The action taken on the recipe named by RcpChangeName. This variable is valid for the collection event Execution Recipe Change Event or New Execution Recipe Event. See the SEMI E5 Standard for a full definition of this variable data item.

RcpExecName — The status variable containing the specifiers of the currently selected recipe(s). See the SEMI E5 Standard for a full definition of this variable data item.

RCP-SELECT — The remote command used to select one or more recipes for execution. See Section 4.4.3.

Recipe Attribute — Information about the recipe that is transferred with the recipe as a name/value pair. The value may be a single item or a list.

Recipe — A recipe contains both a set of instructions, settings, and parameters that the equipment uses to determine the processing environment (its body or process program) and a set of attributes that provide information about the recipe, such as the date and time the body was last changed.

SEMI E42 defines two types of recipes: *managed recipes* and *execution recipes*. For purposes of GEM, the term *recipe* refers to an *execution recipe* only.

Recipe identifier — A recipe identifier is a formatted text string (RCPID) used to identify the recipe.

Recipe specifier — A formatted text string (RCPSPEC) used in messages to indicate a specific recipe. A recipe specifier includes the recipe identifier. It may also include additional information, such as the name of the specific component of the equipment where the recipe is to be executed (e.g. a process chamber) and the name of a recipe repository on the host.

Recipe Verification — Verification is syntax checking of a recipe's body. Verification ensures that a recipe body is structured correctly and has the correct syntax. It may also provide a check of semantics. It does not ensure that the body has the correct parameters to run a particular process or product (see Recipe Validation).

NOTE 16: Unverified recipes shall be verified upon download.

Recipe Validation — Validation is type-and-range checking of parameters in a recipe, and is performed when the recipe is selected for execution. The recipe may be correct in its syntax and semantics but should fail validation if it can not be executed with the current equipment configuration.

Source form recipe — A recipe with a body that is presented as an ordered sequence of text. A source form recipe may be created and edited off-line to the equipment. Definition of syntax requirements shall be documented, in order to allow proper off-line editing.

Variable Parameters — Variable parameters are recipe parameters that are defined in the body of the recipe and whose run-time values may be set outside of the recipe when the recipe is selected for execution and/or when processing is started. Both the host and the operator may specify new settings as a parameter name/value pair.

Variable Parameter Definition — A variable parameter definition has three parts: the name of the variable parameter, its default setting, and restrictions on the run-time value selected. Variable parameter definitions are stored in the recipe attribute "Parameters".

4.6.3 *Description*

4.6.3.1 *Process Program Description*

Process programs allow the equipment's process, and/or the parameters used by that process, to be set and modified by the engineer to achieve different results. Different process programs maybe required for different products, while often the same process program will be used for all lots of a given product. The engineer must be able to create such programs, to modify current programs, and to delete programs from equipment storage.

For the host to ensure that the proper process programs are in place at the equipment, there must be a means of transferring them from equipment to host and from host to equipment. The host also may need to delete process programs from the equipment's storage to make room for a process program to be downloaded. In addition, the host must be kept informed whenever a local change occurs in the contents or status of a process program.

Both formatted and unformatted process programs may be uploaded and downloaded. This capability provides for both host- and equipment-initiated transfers. The equipment-initiated transfer may be used at the request of the process engineer or operator at the equipment.

If a process program exists with the same PPID as the one given in the SECS-II message, the old process program must be replaced. The PPID in the e process program in non-volatile storage.

4.6.3.2 *Recipe Description*

Specifications in Section 4.6.3.1 apply to recipes as well as process programs, with the following differences:

- A recipe contains a body corresponding to a process program. In addition, it contains attributes defined for execution recipes in SEMI E42, Section 6. Recipe attributes are transferred whenever the recipe is downloaded or uploaded.
- The same SECS-II messages are used for all execution recipes, regardless of the internal structure of the recipe body.
- If an execution recipe already exists with the same identifier as the one given in the SECS-II message, the downloaded recipe shall be rejected (not stored) unless the host has specified a "forced overwrite" in the data item RCPOWCODE.
- A recipe currently being edited shall be protected from inadvertent change or overwriting by a recipe with the same identifier that is downloaded during this time. If the downloaded recipe is accepted (stored), the equipment shall require the operator either to save the edited recipe to a new (unused) identifier or to discard it.
- For the equipment to initiate either an upload or download of a recipe, it shall request the host to initiate an upload or download procedure. In addition, it may be necessary to also specify the name of the repository (recipe namespace) at the host.

4.6.3.3 *Large Process Programs and Recipes*

4.6.3.3.1 Process programs and recipes for certain types of equipment, such as metrology or inspection equipment, contain images and are, therefore, very large. Such process programs and recipes cannot fit into a single multi-block SECS-II message that has a maximum theoretical size of 7,995,148 text bytes. Furthermore, process programs and recipes may also require some preparation prior to transfer.

4.6.3.3.2 The commonly used Stream 7 and Stream 15 transfer functions require that the entire process program be sent in a single message. Thus, it is not possible to send a large process program or recipe with such messages. However, there is an alternative set of Stream 7 and Stream 15 functions that supports large transfers. These messages invoke Stream 13 Data Set Transfer Protocol messages to transfer large process programs (or recipes) by using a sequence of read messages. The completion of such a read transaction is indicated by "ERROR: End of Data". The Data Set Transfer Protocol does not set any limit on the size of the data set.

4.6.4 *Requirements*

- The equipment manufacturer shall provide a method to create, modify, and delete process programs or recipes. This method shall exist on either the equipment or on a separate computing system.

- A CEID shall be defined for a collection event for the creation, the deletion, or the modification (completion of an editing session) of a process program (Process Program Change Event). For recipes, there are two separate CEIDs and collection events, one for the creation of a new recipe (New Execution Recipe Event) and one when a recipe is changed or deleted (Execution Recipe Change Event). A New Execution Recipe Event shall occur whenever a new recipe identifier is created through download, edit, copy, or rename operations. A Execution Recipe Change Event shall occur whenever the body of an existing recipe is modified.
- The name (identifier) that the engineer or operator uses to refer to the process program or recipe is the same as the identifier used by the host.
- Upon request from the host or operator, the equipment shall perform the following actions with regard to process programs and recipes stored in non-volatile storage: upload, download, delete, and list current equipment process program or recipe directory.
- The equipment shall be able to store in non-volatile memory the number of process programs or recipes sufficient to execute three unique process cycles. For example, if a wire-bonder requires both an “ALIGN” process program and a “BOND” process program for a full process cycle, then it must provide non-volatile storage for at least three pairs of process programs. These stored process programs or recipes may not be modified in any way by the execution process, nor may the execution process be affected by the modification of any process program or recipes in storage, either by downloading or by local editing, while that process program or recipe~~s~~ is being executed.
- The equipment must provide verification and validation of all downloaded process programs and recipes.
- Stream 7 provides for formatted and unformatted process programs, while Stream 15 provides for recipes. The equipment must support at least one of these three methods.
- The equipment supplier shall document any restrictions on the length or test format of PPID. The maximum length allowed by equipment may be less than that allowed by SECS-II.
- Where recipes are supported, the following requirements also apply:
    - The variable PPFormat shall be provided to indicate to the user the messages supported by the equipment.
    - The recipe and its attributes shall comply to the requirements for execution recipes as defined in SEMI E42, Section 6.

4.6.5 *Scenarios for Process Programs*

4.6.5.1 *Process Program Creation, Editing, or Deletion Process Program Created, Edited, or Deleted by Operator*

```
COMMENTS                          HOST   EQUIPMENT                          COMMENTS

                                                      New process program created,
                                                      edited, or deleted by operator
                                                      at equipment.
                                                      PPChangeName = PPID
                                                      PPChangeStatus = 1 (Created)
                                                                     = 2 (Edited)
                                                                     = 3 (Deleted)
                                                      [IF] CEID for Process Program
                                                      Change Event enabled
                                                      [THEN]
                                         <-- S6,F11   Send Event Report
Event Report Acknowledge        S6,F12-->
                                                      [END_IF]
```

Process Program Deletion by the Host:

```
COMMENTS                          HOST   EQUIPMENT                           COMMENTS

Delete Process Program Send     S7,F17-->
                                        <-- S7,F18   Delete Process Program
                                                     Acknowledge.
                                                     The process program is removed
                                                     from non-volatile storage.
```

4.6.5.2 *Process Program Directory Request*

```
COMMENTS                          HOST   EQUIPMENT                           COMMENTS

Current EPPD Request            S7,F19-->
                                        <-- S7,F20   Current EPPD Data
```

4.6.5.3 *Process Program Upload*

Host-Initiated Process Program Upload — Formatted:

COMMENTS	**HOST**	**EQUIPMENT**	**COMMENTS**
Formatted Process Program Request	S7,F25-->		
		<-- S7,F26	Formatted Process Program Data
Large Formatted Process Program Request	S7,F43-->		
		<-- S7,44	Large Formatted Process Program Acknowledge
			Acknowledge may be “accepted and will be performed later with completion signaled”. When the equipment is ready to send:
		<-- S13,F1	Send Data Set Send
Send Data Set Send Acknowledge	S13,F2-->		
Open Data Set request	S13,F3-->		
		<-- S13,F4	Open Data Set Data
[START] Repeat until ACKC13 indicates error:			
Read Data Set Request	S13,F5-->		
[END]		<-- S13,F6	Read Data set Data
			[IF] CEID for upload event enabled [THEN] [IF] ACKC13 is "ERROR: END OF DATA" [Then]
Event Report Acknowledge		<-- S6,F12	Send Event Report "successful upload"
	S6,F12-->		[ELSE]
Event Report Acknowledge		<-- S6,F11	Send Event Report "bad upload"
	S6,F12-->		
			[END_IF] [END_IF]
Close Data Set Send	S13,F7-->		
		<-- S13,F8	Close Data Set Acknowledge

Host-Initiated Process Program Upload — Unformatted:

```
COMMENTS                          HOST   EQUIPMENT                            COMMENTS
Process Program Request           S7,F5-->
                                         <-- S7,F6[21]    Process Program Data

Large Process Program Request     S7,F41-->
                                          <-- S7,F42  Large Process Program
                                                      Acknowledge

                                                       Acknowledge may be “accepted
                                                       and will be performed later
                                                       with completion signaled”.
                                                       When the equipment is ready
                                                       to send:

                                          <-- S13,F1   Send Data Set Send

Send Data Set Send                S13,F2-->
Acknowledge

Open Data Set request             S13,F3-->
                                          <-- S13,F4   Open Data Set Data
[START]
Repeat until ACKC13 indicates
error:
Read Data Set Request             S13,F5-->
[END]                                     <-- S13,F6   Read Data set Data
                                                       [IF]
                                                       CEID for upload event enabled
                                                       [THEN]
                                                       [IF]
                                                       ACKC13 is "ERROR: END OF
                                                       DATA"
                                                       [Then]
                                          <-- S6,F11   Send Event Report
Event Report Acknowledge          S6,F12-->            “successful upload”
                                                       [ELSE]
                                          <-- S6,F11   Send Event Report "bad
Event Report Acknowledge          S6,F12-->             upload"
                                                       [END_IF]
                                                       [END_IF]
Close Data Set Send               S13,F7-->
                                          <-- S13,F8   Close Data Set Acknowledge
```

21 If the process program does not exist, a zero-length list will be sent.

Equipment Initiated Process Program Upload — Formatted:

```
COMMENTS                          HOST   EQUIPMENT                          COMMENTS

                                                     [IF] Process program is multi-
                                                          block
                                                     [THEN]
                                         <-- S7,F1   Process Program Load Inquire
Process Program Load Grant        S7,F2-->
                                                     [END_IF]
                                         <-- S7,F23  Formatted Process Program Send
Formatted Process Program
Acknowledge                      S7,F24-->
                                         <-- S7,F39  Large Formatted Process Program
                                                     Send
Large Formatted Process          S7,F40-->
Program Acknowledge                      <-- S13,F1  Send Data Set Send

Send Data Set Send               S13,F2-->
Acknowledge

Open Data Set request            S13,F3-->
                                         <-- S13,F4  Open Data Set Data

[START] repeat until
ACKC13 is error:
Read Data Set Request            S13,F5-->
[END]                                    <-- S13,F6  Read Data set Data

                                                     [IF]
                                                     CEID for upload event enabled
                                                     [THEN]
                                                     [IF]
                                                     ACKC13 is "ERROR: END OF DATA"
                                                     [Then]
                                         <-- S6,F11  Send Event Report
Event Report                     S6,F12-->           “successful upload”
Acknowledge                                          [ELSE]
                                         <-- S6,F11  Send Event Report “bad upload”
Event Report                     S6, F12-->
Acknowledge                                          [END_IF]
                                                     [END_IF]
Close Data Set Send              S13,F7-->
                                         <-- S13,F8  Close Data Set Acknowledge
```

Equipment-Initiated Process Program Upload — Unformatted:

```
COMMENTS                          HOST   EQUIPMENT                      COMMENTS

                                                      [IF] Process program is multi-
                                                      block
                                                      [THEN]
                                         <-- S7,F1    Process Program Load Inquire
Process Program Load Grant        S7,F2-->
                                                      [END_IF]
                                         <-- S7,F3    Process Program Send
Process Program Acknowledge       S7,F4-->

                                         <-- S7,F37   Large Process Program Send
Large Process Program            S7,F38-->
Acknowledge
                                         <-- S13,F1   Send Data set Send
Send Data Set Send               S13,F2-->
Acknowledge

Open Data Set request            S13,F3-->
                                         <-- S13,F4   Open Data Set Data
[START] Repeat until
ACKC13 is error:
Read Data Set Request            S13,F5-->
[END]                                    <-- S13,F6   Read Data set Data

                                                      [IF]
                                                      CEID for upload event enabled
                                                      [THEN]
                                                      [IF]
                                                      ACKC13 is "ERROR: END OF DATA"
                                                      [Then]
                                         <-- S6,F11   Send Event Report
Event Report Acknowledge         S6,F12-->            “successful upload”
                                                      [ELSE]
                                         <-- S6,F11   Send Event Report “bad upload”
Event Report Acknowledge         S6,F12-->            [END_IF]
                                                      [END_IF]
Close Data Set Send              S13,F7-->
                                         <-- S13,F8   Close Data Set Acknowledge
```

4.6.5.4 *Process Program Download*

NOTE 17: Formatted process programs must be verified immediately following any download.

While the Process Program Load Inquire/Grant transaction (S7,F1/F2) is required only for the transfer of multi-block process programs, its use is recommended prior to all host-initiated downloads. It provides a means of verifying process program size.

Host-Initiated Process Program Download — Formatted:

```
COMMENTS                          HOST   EQUIPMENT                          COMMENTS

[IF]
Process program is multi-block
[THEN]
Process Program Load Inquire   S7,F1[22] -->
                                       <-- S7,F2    Process Program Load Grant
[ENDIF]
Formatted Process Program Send  S7,F23-->
                                       <--S7,F24    Formatted Process Program
                                                    Acknowledge Verify process
                                                    program
                                                    [IF] S7,F27 is multi-block
                                                    [THEN]
                                       <--S7,F29    Process Program Verification
                                                    Inquire
                                 S7,F30->           Process Program Verification
                                                    Grant
                                                    [END IF]
                                       <--S7,F27    Process Program Verification
                                                    Send
Process Program Verification
Acknowledge                     S7,F28-->

Large Formatted Process         S7,F39-->
Program Send
                                       <-- S7,F40   Large Formatted Process Program
                                                    Acknowledge

                                       <-- S13,F3   Open Data Set Request
Open Data Set Data              S13,F4-->
                                                    [START] repeat until ACKC13 is
                                                     error:
                                       <-- S13,F5   Read Data Set Request
Read Data Set Data              S13,F6-->
                                                    [END]

                                       <-- S13,F7   Close Data Set Send
Close Data Set                  S13,F8-->
Acknowledge
                                                    [IF]
                                                    S7,F27 is multi-block
                                                    [THEN]
                                       <-- S7,F29   Process Program Verification
                                                    Inquire
Process Program                 S7,F30-->
Verification Grant                                  [END_IF]

                                       <-- S7,F27   Process Program Verification
                                                    Send
Process Program                 S7,F28-->
Verification Acknowledge
```

22 S7,F1 should be used only to request permission to transfer a multi-block formatted or unformatted process program. It should not be used to select a process program. For selecting a process program for execution, the remote command PP-SELECT should be used.

Host-Initiated Process Program Download — Unformatted:

```
COMMENTS                           HOST   EQUIPMENT                          COMMENTS

[IF]
Process program is multi-block
[THEN]
Process Program Load Inquire   S7,F1[21] -->
                                         <--S7,F2     Process Program Load Grant
[END_IF]
Process Program Send             S7,F3-->
                                         <--S7,F4     Process Program Acknowledge

Large Process Program           S7,F37-->
Send
                                         <--S7,F38   Large Process Program
                                                      Acknowledge

                                         <--S13,F3    Open Data Set Request

                                S13,F4-->
Open Data Set Data                                    [START] repeat until ACKC13 is
                                                      error:
                                         <--S13,F5    Read Data Set Request
                                S13,F6-->
Read Data set Data                                    [END]

                                         <--S13,F7    Close Data Set Send
                                S13,F8-->
Close Data Set Acknowledge
                                                      [IF]
                                                      S7,F27 is multi-block
                                                      [THEN]
                                                      Process Program Verification
                                         <--S7,F29    Inquire
Process Program                 S7,F30-->
Verification Grant                                    [END_IF]

                                         <--S7,F27    Process Program Verification
                                                      Send
Process Program                 S7,F28-->
Verification Acknowledge
```

Equipment-Initiated Process Program Download — Formatted:

```
COMMENTS                         HOST   EQUIPMENT                          COMMENTS

                                        <--S7,F25    Formatted Process Program
                                                     Request
Formatted Process Program Data  S7,F26-->
                                                     [IF] S7,F27 is multi-block
                                                     [THEN]
                                        <--S7,F29    Process Program Verification
                                                     Inquire
                                S7,F30-->            Process Program Verification
                                                     Grant
                                                     [END IF]
                                        <--S7,F27    Process Program Verification
                                                     Send
Process Program Verification
Acknowledge                     S7,F28-->
                                        <--S7,F43    Large Formatted Process Program
Large Formatted Process         S7,F44-->            Program Request
Program Acknowledge
                                        <--S13,F3    Open Data Set Request

                                S13,F4-->
Open Data Set Data                                   [START] repeat until ACK13 is
                                                     error:
                                        <--S13,F5    Read Data Set Request
                                S13,F6-->
Read Data Set Data                                   [END]

                                        <--S13,F7    Close Data Set Send
                                S13,F8-->
Close Data Set
Acknowledge                                          [IF]
                                                     S7,F27 is multi-block
                                                     [THEN]
                                        <--S7,F29    Process Program Verification
                                                     Inquire
Process Program                                      [END_IF]
 Verification Grant
                                        <--S7,F27    Process Program Verification
                                S7,F28-->            Send

Process Program
Verification Acknowledge
```

Equipment-Initiated Process Program Download — Unformatted:

```
COMMENTS                         HOST   EQUIPMENT                      COMMENTS

                                        <--S7,F5     Process Program Request
Process Program Send             S7,F6-->

                                        <--S7,F41    Large Process Program Request
Large Process Program           S7,F42-->
Program Acknowledge

                                        <--S13,F3    Open Data Set Request

Open Data Set Data              S13,F4-->
                                                     [START] repeat until ACKC13 is
                                                     error:
                                        <--S13,F5    Read Data Set Request

Open Data Set Data              S13,F6-->
                                                     [END]

                                        <--S13,F7    Close Data Set Send

Close Data Set Acknowledge      S13,F8-->
                                                     [IF]
                                                     S7,F27 is multi-block
                                                     [THEN]
                                        <--S7,F29    Process Program Verification
Process Program                 S7,F30-->            Inquire
Verification Grant                                   [END_IF]

                                        <--S7,F27    Process Program
Process Program                 S7,F28-->            Verification Send
Verification Acknowledge
```

4.6.6 Scenarios for Recipes

4.6.6.1 *Recipe Creation, Editing, or Deletion*

Recipe Created by Operator:

```
COMMENTS                         HOST   EQUIPMENT                      COMMENTS
                                                     New recipe created by  operator
                                                     at equipment
                                                     RcpChangeName  =  RCPID
                                                     RcpChangeStatus  =  1 (Created)
                                                     [IF] CEID for New Execution
                                                     Recipe Event enabled
                                                     [THEN]
                                        <--S6,F11    Send Event Report
Event Report Acknowledge        S6,F12-->
                                                     [END_IF]
```

Recipe Edited or Deleted by Operator:

```
COMMENTS                          HOST   EQUIPMENT                           COMMENTS
                                                    New recipe edited, or deleted
                                                    at equipment
                                                    RcpChangeName  =  RCPID
                                                    RcpChangeStatus  =  2
                                                    (Modified) or 5 (Deleted)
                                                    [IF] CEID for Execution
                                                    Execution Recipe Change Event
                                                    enabled
                                                    [THEN]
                                       <--S6,F11    Send Event Report
Event Report Acknowledge        S6,F12-->
                                                    [END_IF]
```

Recipe Deletion by the Host:

```
COMMENTS                          HOST   EQUIPMENT                           COMMENTS
Recipe Delete Request          S15,F35-->
                                       <--S15,F36   Recipe Delete Acknowledge. The
                                                    recipe is removed from non-
                                                    volatile storage.
```

4.6.6.2 *Recipe Directory Request*

Host requests a list of identifiers of currently stored recipes.

```
COMMENTS                          HOST   EQUIPMENT                           COMMENTS
GetAttr Request                 S14,F1-->
                                       <--S14,F2    GetAttr Data
```

4.6.6.3 *Recipe Upload*

Host-Initiated Recipe Upload:

```
COMMENTS                        HOST   EQUIPMENT                         COMMENTS
Recipe Upload Request          S15,F31-->
                                        <--S15,F32   Recipe Upload Data
Large Recipe Upload Request     S15,F51-->
                                        <--S15,F52   Large Recipe Upload Acknowledge

                                                      Acknowledge may be “accepted
                                                      and will be performed later
                                                      with completion signaled”.
                                                      When the equipment is ready to
                                                      send:

                                        <--S13,F1     Send Data Set Send

Send Data Set Send               S13,F2-->
Acknowledge

Open Data Set request            S13,F3-->
                                        <--S13,F4     Open Data Set Data

[START] repeat until
ACKC13 is error:
Read Data Set Request            S13,F5-->
[END]                                   <--S13,F6     Read Data Set Data

                                                      [IF]
                                                      CEID for upload event enabled
                                                      [THEN]
                                                      [IF]
                                                      ACKC13 is "ERROR: END OF DATA"
                                                      [Then]
                                        <--S6,F11     Send Event Report “successful
Event Report Acknowledge         S6,F12-->            upload”
                                                      [ELSE]
                                        <--S6,F11     Send Event Report “bad upload”
Event Report Acknowledge         S6,F12-->            [END_IF]
                                                      [END_IF]

Close Data Set Send              S13,F7-->
                                        <--S13,F8     Close Data Set Acknowledge
```

Equipment-Initiated Recipe Upload:

```
COMMENTS                          HOST   EQUIPMENT                        COMMENTS
                                                      RCPCMD  =  Upload
                                                      [IF] multi-block request
                                         <--S15,F1    [THEN] Recipe Management  Multi-
                                                      block inquire
Recipe Management Multi-        S15,F2-->
block Grant
                                                      [END_IF]
                                         <--S15,F21   Recipe Action Request
Recipe Action Acknowledge      S15,F22-->
Host requests upload           S15,F31-->
                                         <--S15,F32   Recipe Upload Data
                                                      RCPCMD = Upload
                                                      [IF] multi-block request
                                         <--S15,F1    [THEN]
                                                      Recipe Management
Recipe Management               S15,F2-->             Multi-block Inquire
Multi-block Grant                                     [END_IF]
                                         <--S15,F21   Recipe Action Request
Recipe Action Acknowledge      S15,F22-->

                               S15,F51-->
Large Recipe Upload
Request
                                         <--S15,F52   Large Recipe Upload
                                                      Acknowledge

                                         <--S13,F1    Send Data Set Send

Send Data Set Send              S13,F2-->
Acknowledge

Open Data Set Request           S13,F3-->
                                         <--S13,F4    Open Data Set Data

[START] repeat until
ACKC13 is error:
Read Data Set Request           S13,F5-->
[END]                                    <--S13,F6    Read Data Set Data

                                                      [IF]
                                                      CEID for upload event enabled
                                                      [THEN]
                                                      [IF]
                                                      ACKC13 is "ERROR: END OF DATA"
                                                      [Then]
                                         <--S6,F11    Send Event Report “successful
                                                      upload”
Event Report Acknowledge        S6,F12-->             [ELSE]
                                         <--S6,F11    Send Event Report “bad upload”

Event Report Acknowledge        S6,F12-->             [END_IF]
                                                      [END_IF]

Close Data Set Send             S13,F7-->
                                         <--S13,F8    Close Data Set Acknowledge
```

4.6.6.4 *Recipe Download*

The Recipe Management Multi-block Inquire/Grant transaction (S15,F1/F2) is required for the transfer of multi-block recipes. However, its use is recommended prior to all downloads, as it provides recipe size to the equipment.

NOTE 18: If the data item RCPOWCODE is TRUE in S15,F27, then a pre-existing recipe with the same identifier shall be overwritten.

Host-Initiated Recipe Download:

```
COMMENTS                          HOST   EQUIPMENT                         COMMENTS
[IF]
Recipe is multi-block
[THEN]

Recipe Management Multi-
block Inquire                   S15,F1-->

                                        <--S15,F2    Recipe Management Multi-block
                                                     Grant
[END IF]
Recipe Download Request        S15,F27-->
                                                     [IF] RCPOWCODE  =  TRUE
                                                     delete any pre-existing recipe
                                                     with the same identifier before
                                                     storing new recipe
                                                     [END_IF]
                                        <--S15,F28   Recipe Download Acknowledge
Large Recipe Download          S15,F49-->
Request
                                                     [IF] RCPOWCODE = TRUE
                                                     delete any pre-existing recipe
                                                     with the same identifier before
                                                     storing new recipe
                                                     [END_IF]
                                        <--S15,F50   Large Recipe Download
                                                     Acknowledge

                                        <--S13,F3    Open Data Set Request

Open Data Set Data              S13,F4-->
                                                     [START] repeat until ACKC13 is
                                                     error:
                                        <--S13,F5    Read Data Set Request
Read Data Set Data              S13,F6-->
                                                     [END]

                                        <--S13,F7    Close Data Set Send
Close Data Set Acknowledge      S13,F8-->

                                                     [IF] multi-block request
                                        <--S15,F1    [THEN]
Recipe Management               S15,F2-->            Recipe Management
Multi-block Grant                                    Multi-block Inquire
                                                     [END_IF]
                                        <--S15,F53
                                                     Recipe Verification Send
Recipe Verification            S15,F54-->
Acknowledge
```

Equipment-Initiated Recipe Download:

```
COMMENTS                          HOST   EQUIPMENT                            COMMENTS
                                                      RCPCMD  =  Download
                                                      [IF] multi-block request
                                         <--S15,F1    [THEN] Recipe Management Multi-
                                                      block inquire
Recipe Management Multi-block   S15,F2-->
Grant
                                                      [END_IF]
                                         <--S15,F21   Recipe Action Request

Recipe Action Acknowledge      S15,F22-->
[IF]
Recipe is multi-block
[THEN]

Recipe Management Multi-block   S15,F1-->
Inquire
                                         <--S15,F2    Recipe Management Multi-block
                                                      Grant
[END IF]
Recipe Download Request        S15,F27-->
                                                      [IF] RCPOWCODE  =  TRUE
                                                      delete any pre-existing recipe
                                                      with the same identifier before
                                                      storing new recipe
                                                      [END_IF]
                                         <--S15,F28   Recipe Download Acknowledge
                                                      RCPCMD = Download
                                                      [IF] multi-block request
                                         <--S15,F1    [THEN]
Recipe Management               S15,F2-->             Recipe Management
Multi-block Grant                                     Multi-block Inquire
                                                      [END_IF]

                                         <--S15,F21   Recipe Action Request
Recipe Action Acknowledge      S15,F22-->

Large Recipe Download          S15,F49-->
Requests
                                                      [IF] RCPOWCODE = TRUE
                                                      delete any pre-existing recipe
                                                      with the same identifier before
                                                      storing new recipe
                                                      [END_IF]

                                         <--S15,F50   Large Recipe Download
                                                      Acknowledge
Open Data Set Data              S13,F4-->
                                         <--S13,F3    Open Data Set Request

                                                      START] repeat until ACKC13 is
                                                      error:
                                         <--S13,F5    Read Data Set Request
Read Data Set Data              S13,F6-->
                                                      [END]

                                         <--S13,F7    Close Data Set Send
```

```
Close Data Set Acknowledge      S13,F8-->
                                                  [IF] multi-block request
                                                  [THEN]
                                        <--S15,F1    Recipe Management
Recipe Management               S15,F2-->         Multi-block Inquire
Multi-block Grant                                 [END_IF]

                                        <--S15,F53 Recipe Verification Send
Recipe Verification            S15,F54-->
Acknowledge
```

4.6.6.5 *Recipe Verification*

Host requests equipment to verify a recipe that it has stored:

```
COMMENTS                         HOST   EQUIPMENT                          COMMENTS
Recipe Verify Request          S15,F29-->
                                                  Equipment verifies recipe
                                        <--S15,F30   Recipe Verify Data
```

4.7 *Material Movement* — The material movement capability includes the physical transfer of material (WIP, tools, expendable materials, etc.) between equipment, buffers, and storage facilities. The transfer of material can be performed by operators, AGV robots, tracks, or dedicated fixed material handling equipment.

4.7.1 *Purpose* — This capability is limited in implementation, serving to notify the host of the appearance or removal of material at the equipment's ports.

4.7.2 *Definitions*

Port — A point or area on the equipment at which a change in equipment ownership of material may occur.

4.7.3 *Description* — This capability consists of alerting the host whenever material is sent or received from any of the ports on the equipment. Event-specific information, such as port identification and material identification, also may be useful, but definition of these and other related DVVAL's are left to the implementation.

4.7.4 *Requirements* — The equipment must supply two CEIDs, one to report when material is sent from any port and the other to report when material is received at any port.

4.7.5 *Scenarios*

```
COMMENTS                         HOST   EQUIPMENT                          COMMENTS

                                                  Material is sent or received at
                                                  an equipment port.
                                        <--S6,F11    Send collection event to host.
Host acknowledges.              S6,F12-->
```

4.8 *Equipment Terminal Services* — Equipment Terminal Services allows the host to display information on the equipment's display device or the operator of the equipment to send information to the host.

4.8.1 *Purpose* — Equipment Terminal Services allows the factory operators to exchange information with the host from their equipment workstations.

4.8.2 *Definitions* —Message Recognition: a positive action by the equipment operator indicating the operator has viewed the text of a host initiated message.

4.8.3 *Detailed Description* — The equipment must be capable of displaying information passed to it by the host for the operator's attention. The information, or an indication of a message, must remain on the equipment's display until the operator indicates message recognition. Message recognition results in a collection event that informs the host that the operator has actually viewed the information.

The equipment must be capable of passing information to the host that has been entered from the operator's equipment console. This information is intended for host applications and is not processed by the equipment.

The equipment has no responsibility for interpreting any of the data passed to or from the host using this method.

4.8.4 *Requirements*

— Any new Terminal Display message sent by the host shall overwrite an unrecognized message at the same equipment terminal.

— The equipment must provide a display device capable of displaying at least 160 characters to the operator.

— The equipment must provide a mechanism for displaying information sent to it by the host.

— The equipment must provide an indicator to notify the operator when an unrecognized message is present.

— The equipment must provide a mechanism for the operator to indicate message recognition (e.g., push button, terminal function).

— The equipment must provide a means for alpha numeric data entry that can be used by the operator.

— The equipment must support operator entry of at least 160 characters per message.

— The equipment must have a mechanism to send operator-entered messages to the host.

— The equipment must support single-block messages as a minimum. Support of multi-block messages is optional.

— A Terminal Display message received by the equipment with a zero length TEXT data item shall be accepted and replace any previous unrecognized message, but shall not itself be considered an unrecognized message. This provides a method of clearing an unrecognized message and turning off the unrecognized message indicator.

4.8.5 *Scenarios*

Host sends information to an equipment's display device:

```
COMMENTS                         HOST   EQUIPMENT                     COMMENTS

Host sends textual information  S10,F3-->
to equipment for display to the
operator on terminal x.
                                        <--S10,F4    Equipment acknowledges request
                                                     to display text. (Equipment
                                                     sets unrecognized message
                                                     indicator.)
                                                     Operator indicates message
                                                     recognition.
                                                     (Equipment clears unrecognized
                                                     message indicator.)
                                        <--S6,F11    Message recognition event.
                                                     (See Section 4.2.1, Event Data
                                                     Collection, for details.)
Host acknowledges Optional:     S6,F12-->
                                        <--S10,F1    Operator responds with text
                                                     via terminal x.
Host acknowledges receipt       S10,F2-->
of operator text.
```

Host sends information to an equipment's display device and then overwrites the information before operator recognizes message:

```
COMMENTS                          HOST   EQUIPMENT                             COMMENTS

Host sends textual information  S10,F3-->
to equipment for display to
the operator on terminal x.
                                        <--S10,F4    Equipment acknowledges request
                                                     to display text. (Equipment
                                                     sets unrecognized message
                                                     indicator.)
Host sends textual information  S10,F3-->
to equipment for display to the
operator on terminal x. This
message overwrites the first one
sent by the host since it is
still unrecognized.
                                        <--S10,F4    Equipment acknowledges request
                                                     to display text. (Equipment
                                                     sets unrecognized message
                                                     indicator.)
                                                     Operator indicates message
                                                     recognition. (Equipment clears
                                                     unrecognized message
                                                     indicator.)
                                        <--S6,F11    Message recognition event. (See
                                                     Section 4.2.1, Event Data
                                                     Collection, for details.)
Host acknowledges               S6,F12-->
```

Operator sends information to the host:

```
COMMENTS                          HOST   EQUIPMENT                             COMMENTS
                                        <--S10,F1    Operator sends textual
                                                     information via equipment
                                                     terminal x.
Host acknowledges receipt of    S10,F2-->
operator initiated message.
Optional:
Host responds with information  S10,F3-->
for display to the operator
on terminal x.
                                        <--S10,F4    Equipment acknowledges receipt
                                                     of request to display text.
                                                     (Equipment sets unrecognized
                                                     message indicator.)
                                                     Operator indicates message
                                                     recognition.
                                        <--S6,F11    Message recognition event.
                                                     (See Event data collection for
                                                     details.)
Host acknowledges               S6,F12-->
```

Host sends a multi-block display message:

```
COMMENTS                         HOST   EQUIPMENT                         COMMENTS

Send information               S10,F5-->
                                        <--S10,F6    Accepted or denied.
                                                     [IF]
                                                     Message from host is multi-
                                                     block and multi-block is not
                                                     supported by the equipment,
                                                     [THEN]
                                        <--S10,F7    Send Multi-block Not Allowed.
                                                     [END_IF]
```

4.9 *Error Messages*

4.9.1 *Purpose* — Error messages provide the host with information describing the reason for a particular message or communication fault detected by the equipment.

4.9.2 *Definitions*

Communication Fault — Refer to Section 2 for the definition.

Message Fault — Refer to Section 2 for the definition.

4.9.3 *Detailed Description* — The equipment must inform the host that it cannot process a message due to an incorrect:

— device ID,

— message stream type,

— message function,

— message format, or

— data format.

The equipment must inform the host if the message has more data than it can handle.

The equipment must inform the host if the equipment's transaction timer expires.

The equipment shall treat the above conditions as application-level errors and shall not take any further action on any message in error.

Error messages are invoked whenever the equipment detects communication or message faults.

4.9.4 *Requirements* — Support of all Stream 9 messages.

4.9.5 *Scenario*

Message Fault Due to Unrecognized Device ID:

```
COMMENTS                         HOST   EQUIPMENT                         COMMENTS

Host sends a message.            Sx,Fy-->
                                                     Equipment detects an
                                                     unrecognized device ID within
                                                     the message from the host.
                                        <--S9,F1     Equipment reports to the host
                                                     that an "unrecognized device
                                                     ID" was detected in the
                                                     received message.
```

Message Fault Due to Unrecognized Stream Type:

COMMENTS	HOST	EQUIPMENT	COMMENTS
Host sends a message.	Sx,Fy-->		
			Equipment detects an unrecognized stream type within the message from the host.
		<--S9,F3	Equipment reports to the host that an “unrecognized stream type” was detected in the received message.

Message Fault Due to Unrecognized Function Type:

COMMENTS	HOST	EQUIPMENT	COMMENTS
Host sends a message.	Sx,Fy-->		
			Equipment detects an unrecog-nized function type within the message from the host.
		<--S9,F5	Equipment reports to the host that an “unrecognized function type” was detected in the received message.

Message Fault Due to Illegal Data Format:

COMMENTS	HOST	EQUIPMENT	COMMENTS
Host sends a message.	Sx,Fy-->		
			Equipment detects illegal data format within the message from the host.
		<--S9,F7	Equipment reports to the host that “illegal data format” was detected in the received message.

Communication Fault Due to Transaction Timer Timeout:

COMMENTS	HOST	EQUIPMENT	COMMENTS
			Equipment does not receive an expected reply message from the host and a transaction timer timeout occurs.
		<--S9,F9	Equipment reports to the host that a transaction timer timeout occurred.

Message fault due to data too long:

```
COMMENTS                         HOST   EQUIPMENT                           COMMENTS

Host sends a message.            Sx,Fy-->
                                                    Equipment detects that the
                                                    message from the host contains
                                                    more data than it can handle.
                                        <--S9,F11   Equipment reports to the host
                                                    that “data too long“ was
                                                    detected in the received
                                                    message.
```

Communication Fault Due to Conversation Timeout:

```
COMMENTS                         HOST   EQUIPMENT                           COMMENTS

Host sends a message.            Sx,Fy-->
                                        <--Sx,Fy+1  Equipment sends reply.
                                                    Equipment is now expecting a
                                                    specific message from the host
                                                    as a result of the previous
                                                    transaction. Equipment has not
                                                    received the expected message
                                                    from the host and a
                                                    conversation timeout occurs.
                                        <--S9,F13   Equipment reports to the host
                                                    that a conversation timeout
                                                    occurred.
```

4.10 *Clock* — The clock capability enables host management of time-related activities and occurrences associated with the equipment and across multiple pieces of equipment.

4.10.1 *Purpose* — The purpose of the clock capability is to enable time stamping of collection event and alarm reports. Time stamping is useful for resolving relative order of event/alarm occurrences and scheduling of equipment activities by the host.

The ability for the host to instruct the equipment to set an internal clock to a specified time value, and for the equipment to request the current date and time, is needed for effective time management and synchronization between host and equipment.

4.10.2 *Definitions*

Clock — Clock is a status variable containing the current value of time at the equipment. Clock may be included in report definitions and/or queried separately by the host. See SEMI E5 for a full definition of this variable data item and its required formatting.

TIME — TIME is a data item contained in messages used by the host to set time at the equipment and by the equipment or host to request the current time from the other. (See SEMI E5 for a full definition of this data item.)

4.10.3 *Detailed Description* — The clock capability assumes the existence of a relative time reference on the equipment. This time reference is used as a basis for updating the time value of an equipment status variable called “Clock.” The time reference must reflect the current time to within a resolution range of seconds to centiseconds (refer to the format for Clock in the SEMI E5 Standard). The purpose of time stamping with centiseconds is to resolve the order in which nearly simultaneous events occur rather than to provide a more precise record of the time of day at which they occurred. Where more than one event occurs within a given period of clock resolution, the centiseconds reported in the time stamps for these events must reflect the actual order in which the events were detected. Equipment with a clock resolution of less than a second should report centiseconds. Otherwise, centiseconds should be assigned to reflect the relative order in which events were detected. Equipment unable to resolve time to less than a second and unable to reflect the relative order in which events were detected may report centiseconds as “00”.

The host employs the "Date and Time Set Request" message (S2,F31) to initialize the value of Clock to the value contained in the TIME data item. Similarly, the equipment may employ the "Date and Time Request" message (S2,F17) to obtain a new initialization time for Clock. As before, the value of TIME returned by the host is used to set Clock. Note, in the event that the precision of TIME is seconds and that for Clock is centiseconds, in both cases the initial value of Clock shall contain "00" for its Centisecond digits upon initialization. Additionally, for any field in TIME that is not supported by the equipment, the local value of this field is equipment dependent. For example, Equipment that cannot resolve time to less than a second might round or ignore centiseconds and always set the Centisecond field to "00".

4.10.4 *Requirements*

— The resolution and update rate of the internal time reference must be sufficient to distinguish between two nearly simultaneous collection events and/or alarms.

— The equipment supplier shall provide documentation describing the resolution of the internal time reference.

— The equipment supplier shall provide documentation describing how centisecond values are assigned, including the case of unresolvable simultaneous events.

4.10.5 *Scenarios*

Equipment Requests TIME (Optional Scenario):

COMMENTS	HOST	EQUIPMENT	COMMENTS
		<--S2,F17	Equipment requests a time value from the host.
Host responds with a TIME value	S2,F18-->		
			Equipment sets its internal time reference to the value of TIME received from the host.

Host Instructs Equipment to Set Time:

COMMENTS	HOST	EQUIPMENT	COMMENTS
Host instructs equipment to set its time.	S2,F31-->		
		<--S2,F32	Equipment sets its internal time reference to the value of TIME received from the host and acknowledges completion.

Host Requests Equipment's Current Time Value:

COMMENTS	HOST	EQUIPMENT	COMMENTS
Host requests equipment time.	S2,F17-->		
		<--S2,F18	Equipment returns its internal time reference value to the host.

4.11 *Spooling* — Spooling is a capability whereby the equipment can queue messages intended for the host during times of communication failure and subsequently deliver these messages when communication is restored. Spooling is limited to primary messages of user-selected streams.

4.11.1 *Purpose* — The purpose of spooling is to provide a method for retaining equipment message data that might otherwise be lost due to communication failure. The motive for producing this functionality is to retain valuable data used to track material and to improve product quality. The spooling capability fills a gap in the SEMI E5 standard. In the past, without a spooling capability, the equipment has typically discarded messages that could not be delivered, or turned messaging off altogether. It is intended that the host initiate the spool unload process immediately following the reestablishment of communications.

4.11.2 *Definitions*

MaxSpoolTransmit — An equipment constant containing the maximum number of messages that the equipment shall transmit from the spool in response to an S6,F23 "Transmit Spooled Messages" request. If MaxSpoolTransmit is set to zero, no limit is placed on the messages sent from the spool. Multi-block inquire/grant messages are not counted in this total.

OverWriteSpool — A boolean equipment constant used to indicate to the equipment whether to overwrite data in the spool area or to discard further messages whenever the spool area limits are exceeded.

Send Queue — Refers to the queue into which equipment generated SECS messages are placed in preparation for transmission to the host.

Spool — The spool is an area of non-volatile storage in which the equipment stores certain messages that cannot be delivered to the host (when the equipment is in the NOT COMMUNICATING substate of COMMUNICATIONS ENABLED). The spool area can be thought of as a sequential "ring" buffer. The term spool is also used to denote the action of placing messages into the spool area.

SpoolCountActual — A status variable used to keep a count of the messages actually stored in the equipment's spool area. Multi-block inquire/grant messages are not spooled and not included in this count.

SpoolCountTotal — A status variable used to keep a count of the total number of primary messages directed to the spool, regardless of whether placed or currently retained in the spool. Multi-block inquire/grant messages are not spooled and not included in this count.

SpoolFullTime — A status variable containing the timestamp when the spool last became full. If the spool was not filled during the last spooling period, this will contain a time value prior to the current SpoolStartTime.

SpoolStartTime — A status variable containing the timestamp from when spooling was last activated.

4.11.3 *Description*

4.11.3.1 *Spooling State Model* — There are two major states of spooling: SPOOL INACTIVE and SPOOL ACTIVE. SPOOL ACTIVE has two components: SPOOL UNLOAD and SPOOL LOAD. These are each broken into substates. A description of all spooling states, substates, and applicable state transitions follows. The POWER OFF and POWER ON parent states are common to all equipment subsystems. They are shown here to illustrate the retention of spooling context during a power down situation.

NOTE 20: Disabling SECS communications does not affect the current spooling state since no messages are generated until communications are subsequently enabled. Spooling is effectively frozen in this case.

POWER OFF

The equipment has lost power for any reason (e.g., power failure, power switch set to off).

POWER ON

The equipment is powered up.

SPOOL INACTIVE

This is the normal operating mode. No spooling occurs. The spool area is empty. Primary SECS-II messages are transmitted normally.

SPOOL ACTIVE

All primary SECS-II messages ready for sending and for which spooling is enabled (see S2,F43) are directed to the spool area. All other primary messages, except Stream 1, are discarded. The equipment shall attempt to send any secondary messages that are generated and discard these messages should the attempt to send fail.

Spool state transitions from SPOOL INACTIVE to SPOOL ACTIVE if the communications state changes from COMMUNICATING to NOT COMMUNICATING (Communication State Transition Table, #14) or from WAIT CRA to WAIT DELAY (Communication STATE Transition Table, #6).

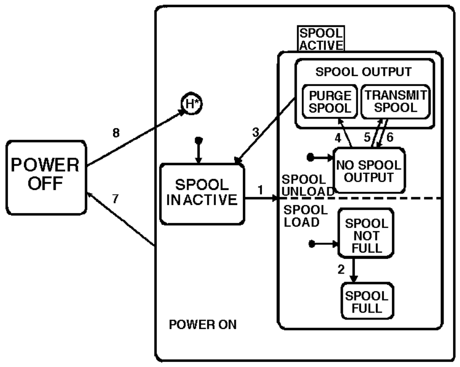

Figure 4.11
Spooling State Diagram

Once communications are established, the host must initiate the spool unload sequence to restore full functionality (see below). Since the equipment will deliver secondary messages, the host may inquire for information or send commands as needed.

The SPOOL ACTIVE state has two AND states: SPOOL LOAD and SPOOL UNLOAD. This means that they operate independently, though sharing data and some state change stimuli. See Section 3.1 for explanation of state model terms and notation.

SPOOL LOAD

The SPOOL LOAD component enters messages into the spool area. It is divided into two substates: SPOOL NOT FULL and SPOOL FULL. SPOOL NOT FULL is the default entry substate of the parent state SPOOL LOAD.

SPOOL NOT FULL

As primary SECS-II messages are directed to the spool area, the equipment shall "write" the SECS-II message to the end of the spool. Status variables SpoolCountTotal and SpoolCountActual shall be incremented each time a message is placed in the spool area.

SPOOL FULL

In this state, all of the allocated spooling area is filled. Choice of the following options shall be controlled by the setting of a Boolean equipment constant called "OverWriteSpool." The first message to be dealt with is that which could not be fit into the spool prior to transition from SPOOL NOT FULL (see transition table below).

OverWriteSpool is True: The equipment deletes as many of the "oldest" records (e.g., SECS-II messages) contained in the spool area necessary to make space for the new message and then adds the message. Status variable SpoolCountTotal shall be incremented whenever a message is submitted to the spool area. Status variable SpoolCountActual shall be manipulated to keep an accurate count of the number of messages contained in the spool area. For example, if it is necessary to delete three messages in the spool area to spool one message, SpoolCountActual would have three subtracted and then one added to the total.

OverWriteSpool is False: Any subsequent primary messages shall be discarded. When such a message is discarded, SpoolCountTotal is incremented.

SPOOL UNLOAD

The SPOOL UNLOAD component of SPOOLACTIVE deals with movement of messages out of the spool. It has an active substate (SPOOL OUTPUT) and a passive substate (NO SPOOL OUTPUT). NO SPOOL OUTPUT is the default entry substate, since the equipment is NOT COMMUNICATING at the time spooling is initiated. When communications between equipment and host are restored, there is an opportunity for the host to recover spooled messages. No action is taken until the host initiates the spool output process via the S6,F23 (Request Spooled Data). The host has the option to either receive the spooled messages (see substate TRANSMIT SPOOL) or discard all messages in the spool (see substate PURGE SPOOL).

NO SPOOL OUTPUT

In this state, no messages are removed from the spool.

SPOOL OUTPUT

The SPOOL OUTPUT state encompasses the removal of messages from the spool. Its substates are TRANSMIT SPOOL and PURGE SPOOL.

TRANSMIT SPOOL

The host elects to receive all messages contained in the spool area. The equipment is expected to keep track of the oldest record (i.e., message) within the spool area. When communications are re-established with the host and transmission of the spool area is started, the oldest record must be the first record transmitted, then the next oldest record, etc. There is no prioritization of messages to be sent from the spool.

As each spooled message is successfully transmitted to the host, it is removed from the spool area upon successful completion of the transaction. SpoolCountActual is decremented as each message is removed from the spool. The equipment shall transmit messages only from the spool area until all spooled messages have been completely transmitted to the host.

Flow control of the spool transmit process is achieved in two ways. First, only one open transaction on the equipment is allowed during spool unload. Thus, if a message requires a reply, the equipment shall wait for that reply before transmitting the next spooled message. Messages which require no reply may be transmitted sequentially as rapidly as the message transfer mechanism will allow.

The second flow control method is to allow the host to limit the maximum number of messages sent from the spool in response to the S6,F23 request. An equipment constant named MaxSpoolTransmit may be set by the host to achieve this behavior. If MaxSpoolTransmit is set to five, for example, the equipment will send the first five messages from the spool and then transition to the NO SPOOL OUTPUT state, awaiting the next S6,F23 request. There is no event report generated when MaxSpoolTransmit is reached. The host is responsible for determining this situation by a) counting the messages received, b) timing out waiting for the next message, c) inquiring to the equipment for the curent value of the SpoolCountActual status variable, or d) some combination of the above. If MaxSpool-Transmit is set to zero, the spool shall be transmitted completely in response to S6,F23.

Normal spooling continues during the spool transmit process. If the SPOOL LOAD component should transi-tion to SPOOL FULL, it shall not have any effect on the SPOOL UNLOAD component. Once full, the spool cannot make the transition back to SPOOL NOT FULL except via the SPOOL INACTIVE state. Space made available due to the spool unload process shall not be used in this case.

When a multi-block message is to be transmitted from the spool, any required inquire/grant transaction shall be initiated. If the host's response denies permission to send the multi-block message, the equipment shall discard that message and continue with the transmit process. This sequence shall count as one message in the MaxSpoolTransmit count.

There is one area where SPOOL LOAD and SPOOL UNLOAD may interact: When the spool is full and OverWriteSpool is True. During the spool transmit process, spooled messages are being removed and new primary messages are being written to the spool. These new messages are overwriting the oldest messages available, unless the unload process has freed sufficient spool space. There is a possibility that the unload and overwrite processes may compete for control of the same message area. For example, if the spool holds messages ABCDE, with A oldest and E newest, A might be sent to the host, B (and the space from A) overwritten by the new message F, C sent to the host, D and E (and the space from message C) overwritten by G, etc. The loss of continuity may be "disorienting" to the host program receiving the messages. It is expected that the unload process will be fast relative to the generation of new messages, so that this occurrence will be rare.

Should a communication failure occur during the spool transmit process, spooling shall continue as before the transmit process began. However, the spool unload sequence shall terminate (i.e., transition to NO SPOOL OUTPUT will occur — see transition table below).

PURGE SPOOL

The equipment shall discard all messages in the spool and, when the spool is empty, zero SpoolCountActual.

Spooling State Transitions

A table follows detailing all spooling state transitions as presented in the state transition diagram.

Table 4.11 Spooling State Transition

#	*Current State*	*Trigger*	*New State*	*Action*	*Comment*
1	SPOOL INACTIVE	Communication state changes from COMMUNICATING to NOT COMMUNICATING or from WAIT CRA to WAIT DELAY and Enable Spool is true.	SPOOL ACTIVE	SpoolCountActual and SpoolCountTotal are initialized to zero. Any open transactions with the host are aborted. SpoolStartTime (SV) is set to current time. Alert the operator that spooling is active.	The default state in each AND substate is entered. The message which could not be sent remains in the send queue and is dealt with in Spool Active state. The collection event Spooling Activated has occurred.
2	SPOOL NOT FULL	Message generate which will not fit into spool area.	SPOOL FULL	SpoolFullTime (SV) is set to current time. Alert the operator that spool is full.	The message which would not fit into the spooling area is dealt with after the transition. No collection event is generated.
3	SPOOL OUTPUT	Spool area emptied.	SPOOL INACTIVE	Spooling process disabled. Alert the operator that spooling has been terminated.	The collection event Spooling Deactivated has occurred. Transition from the AND substate Spool Unload component occurs.
4	NO SPOOL OUTPUT	S6,F23 received w/RSDC = 1.	PURGE SPOOL	No action.	Initiates purging process. No collection event is generated since this is based on host request.
5	NO SPOOL OUTPUT	S6,F23 received w/RSDC = 0.	TRANSMIT SPOOL	No action.	Initiates message transmission from spool. No collection event is generated since this is based on host request.
6	TRANSMIT SPOOL	Communication failure or MaxSpoolTransmit reached.	NO SPOOL OUTPUT	Spool transmission process suspended.	If communications failure, the event Spool Transmit Failure has occurred. No collection event generated for MaxSpoolTransmit reached.
7	POWER ON	Equipment power source discontinued.	POWER OFF	No action.	Spooling context has been maintained in non-volatile storage prior to this transition.
8	POWER OFF	Equipment power source restored.	POWER ON	Spooling context restored from non-volatile memory.	If spooling were active prior to power down, it shall continue. If TRANSMIT SPOOL were active at powerdown, transition #6 is expected to follow since communications state is initially NOT COMMUNICATING.

4.11.3.2 *Enabling Spooling* — The equipment shall provide the host with the ability to enable and disable Spooling for any message (except Stream 1 messages (i.e., S1,F1, S1,F13)) via the S2,F43/F44 transaction. Spooling may be enabled for an entire Stream, for individual messages within a stream, or for any combination of the two. Streams and Functions not referenced in this message are not spooled. Spooling can be totally disabled by sending an S2,F43 with a zero length list for the first item (see S2,F43 definition). In addition, the equipment shall provide an equipment constant, EnableSpooling, to allow setting the enable or disable of spooling. NOTE: When EnableSpooling is false, the SPOOL state cannot transition from SPOOLINACTIVE to SPOOLACTIVE. Changing EnableSpooling does not change the spool state, purge the spool or change streams and functions enabled for spooling. Once the equipment is "SPOOL ACTIVE", the host must initiate the spool unload sequence to restore full functionality even though “Enable Spooling” has changed to false.

4.11.4 *Requirements* — The following items are required to support the spooling capability:

— At a minimum, the equipment shall reserve for spooling non-volatile storage with sufficient capacity to store all of the primary SECS-II messages that would occur during a normal processing cycle.

— While spooling is enabled, the equipment shall reply to primary messages sent by the host with the appropriate secondary message.

— Secondary messages which cannot be delivered shall be discarded, never spooled.

— All spooling-related status variables and setup information (as per S2,F43) must be stored in non-volatile memory along with any other information required for the potential unloading of the spool area after a power loss.

— Upon powerup, the equipment shall retain all spooling context from the time the equipment was last shutdown or reset. This means that spooling, if previously active, continues upon system powerup.

— The Equipment must reject any message that attempts to set “Spooling” for Stream 1.

— If a multi-block primary message need for the inquire/grant is to be sent during SPOOL ACTIVE, the message should be placed in the spool and the grant resolved during spool transmit.

4.11.5 *Scenarios*

Define the Set of Messages to be Spooled:

This Scenario is used to set up the list of messages that the equipment should spool (or by defining none, to disable spooling).

```
COMMENTS                         HOST   EQUIPMENT                          COMMENTS

Host defines messages to be     S2,F43-->
spooled in case of communications
failure.
                                        <--S2,F44    Equipment acknowledges setup.
```

Define the Maximum Number of Messages to Send in Response to S6,F23:

This Scenario sets the value of the equipment constant MaxSpoolTransmit.

```
COMMENTS                         HOST   EQUIPMENT                          COMMENTS

Host sends value for            S2,F15-->
equipment constant MaxSpoolTransmit.
                                        <--S2,F16    Equipment acknowledges
                                                     equipment constant change.
```

Request or Delete Spooled Data (MaxSpoolTransmit = 0):

This Scenario is used to initiate the transfer of the spooled data from the equipment to the host or to purge the spool.

```
COMMENTS                        HOST   EQUIPMENT                           COMMENTS

                Communications were lost and then re-established.
Host requests data that
includes spool-related
status variables.                S1,F3-->
                                        <--S1,F4     Send status data.
NOTE: S1,F3 is one of
various methods that
could be used. Request
or delete spooled data.         S6,F23-->
                                        <--S6,F24    Request spooled data
                                                     acknowledgement.
                                                     [IF] RSDC = 0 (Spool data
                                                     requested.)
                                                     [THEN]
                                                     The appropriate Streams and
                                                     Functions are used to transmit
                                                     the spooled data to the host.
                                                     [ELSE_IF] RSDC = 1
                                                     [THEN]
                                                     Spool data discarded.
                                                     [END_IF]
                                        <--S6,F11    Spooling Deactivated event
                                                     report sent.
Acknowledge                     S6,F12-->
```

Request or Delete Spooled Data (MaxSpoolTransmit > 0):

This Scenario shows the affect of MaxSpoolTransmit < SpoolCountActual on the Spool Transmit process. For the purpose of illustration, the value of MaxSpoolTransmit is 5 and the SpoolCountActual is 8 (at the time communications are re-established). No messages are added to the Spool during the transmit process.

```
COMMENTS                          HOST   EQUIPMENT                        COMMENTS

                Communications were lost and then re-established.
Host requests data that
includes spool-related
status variables.                 S1,F3-->
                                         <--S1,F4     Send status data (e.g.
                                                      SpoolCountActual = 8, MaxSpool-
                                                      Transmit = 5).
Host requests spooled data
(RSDC=0).                        S6,F23-->
                                         <--S6,F24    Request spooled data
                                                      acknowledgement.
                                                      The five oldest messages in the
                                                      Spool are transmitted to the
                                                      host. Spooling remains active.
Host recognizes that MaxSpoolTransmit
is reached.
Host requests additional         S6,F23-->
spooled data (RSDC = 0).
                                         <--S6,F24    Request spooled data
                                                      acknowledgement.
                                                      The three remaining messages
                                                      are transmitted from the spool.
                                         <--S6,F11    Spooling Deactivated event
                                                      report sent.
Acknowledge                      S6,F12-->
```

4.12 *Control* — The control-related capabilities allow for configuration and manipulation of the control state model. In this way the host and/or user may modify the equipment's control-related behavior.

4.12.1 *Purpose* — This section complements the CONTROL state model description found in Section 3.3. It defines the requirements for implementation of this model.

4.12.2 *Definitions* — None.

4.12.3 *Description*

4.12.3.1 *Control Configuration* — The control state model has two areas of configuration. The first area is related to default entry states of the state model. Upon system initialization, the system must activate either the ON-LINE or OFF-LINE state. Upon entry to OFF-LINE, the system must in turn activate one of the substates of OFF-LINE (EQUIPMENT OFF-LINE, ATTEMPT ON-LINE, or HOST OFF-LINE). In both these cases, the user shall configure the equipment to make the choices appropriate to that factory. Entry to the ON-LINE state also involves a choice of substates. In this case, the equipment reads the front panel REMOTE/LOCAL switch to determine the appropriate state.

The second area of configuration involves the transition to be made if the ON-LINE attempt should fail. The model may be set to transition to either HOST OFF-LINE or to EQUIPMENT OFF-LINE should the S1,F1 transaction be terminated unsuccessfully. Choosing HOST OFF-LINE allows the host to cause the equipment to transition to ON-LINE when the host becomes ready. This is accomplished via the message S1,F17 (see below).

4.12.3.2 *Changing Control State* — In the control state model, both the operator and the host can affect the control state. The operator retains ultimate authority to set the equipment OFF-LINE by means of an OFF-LINE switch mechanism. The operator also can cause the equipment to attempt to go ON-LINE. Under some circumstances, the host can initiate the transition to ON-LINE.

If the operator requests ON-LINE, the equipment will send an S1,F1 to the host. The host may confirm ON-

LINE with an S1,F2 or deny ON-LINE by sending an S1,F0.[23]

When the equipment is ON-LINE, the host may request that it transition to OFF-LINE. It will transition into the HOST OFF-LINE substate. When the equipment HOST OFF-LINE state is active, the host may request that it transition to ON-LINE. The combination of these two allow the host to cycle the equipment between ON-LINE and OFF-LINE.

Only the operator may change the ON-LINE substate (REMOTE or LOCAL).

4.12.4 *Requirements*

— The equipment shall supply a method for configuring the default CONTROL state to be activated upon system initialization. The choice of states must be among ATTEMPT ON-LINE, EQUIPMENT OFF-LINE, HOST OFF-LINE, and ON-LINE.

— The equipment shall supply a method for configuring which state should be activated when the attempt to go ON-LINE fails. The option is a transition to either the HOST OFF-LINE state or the EQUIPMENT OFF-LINE state.

— The equipment shall supply a momentary switch which will initiate the transition to OFF-LINE and another which will begin the process to go ON-LINE. Discrete position switches shall not be used. These should be designed so that they may not be actuated simultaneously. The switch may be mounted on the front panel or be available via keyboard input at the operator console.

— The equipment shall supply a discrete two-position switch which the operator may use to indicate the desired substate for ON-LINE (i.e., REMOTE or LOCAL). The switch may be mounted on the front panel or be available via keyboard input at the operator console. If implemented in software, this setting shall be retained in non-volatile storage.

— The equipment shall supply an indicator on the front panel which displays the full identification of the current CONTROL state/substate (e.g., OFF-LINE/ATTEMPT ON-LINE). This may be accomplished either with labelled display lights or via the operator console display. It is recommended that this indicator be visible at all times.

— The equipment shall supply a status variable that contains the current state/substate of the CONTROL state model.

— Whenever the ON-LINE/REMOTE state is active and the operator issues a command to the equipment, the equipment shall cause an “operator command issued” event.

23 If there is no host response (i.e. reply timeout), the equipment shall treat it as a denial.

4.12.5 *Scenarios*

4.12.5.1 *Operator-Initiated Scenarios*

Host Accepts ON-LINE:

```
COMMENTS                           HOST   EQUIPMENT                           COMMENTS

    Operator actuates ON-LINE switch when equipment OFF-LINE state is active.

                                          <--S1,F1     Equipment requests ON-LINE.
Host grants ON-LINE.               S1,F2-->
                                          <--S6,F11    “Control State LOCAL (or
                                                       REMOTE)” event.
Acknowledge.                      S6,F12-->
```

Host Denies ON-LINE:

```
COMMENTS                           HOST   EQUIPMENT                           COMMENTS

    Operator actuates ON-LINE switch when equipment OFF-LINE state is active.

                                          <--S1,F1     Equipment requests ON-LINE.
Host denies ON-LINE.               S1,F0-->
```

Operator Sets OFF-LINE:

```
COMMENTS                           HOST   EQUIPMENT                           COMMENTS

    Operator actuates OFF-LINE switch when equipment ON-LINE state is active.

                                          <--S6,F11    “Equipment requests OFF-LINE”
                                                       event.
Acknowledge.                      S6,F12-->
```

Operator Sets REMOTE:

```
COMMENTS                           HOST   EQUIPMENT                           COMMENTS

                       Operator sets switch from LOCAL to REMOTE.

                                          <--S6,F11    “Control State REMOTE” event.
Acknowledge.                      S6,F12-->
```

Operator Sets LOCAL:

```
COMMENTS                           HOST   EQUIPMENT                           COMMENTS

                       Operator sets switch from REMOTE to LOCAL.

                                          <--S6,F11    “Control State LOCAL” event.
Acknowledge.                      S6,F12-->
```

4.12.5.2 *Host-Initiated Scenarios*

Host Sets OFF-LINE:

```
COMMENTS                         HOST   EQUIPMENT                         COMMENTS

Host requests OFF-LINE.         S1,F15-->
                                                    [IF] Equipment is OFF-LINE
                                                    [THEN]:
                                        <--S1,F0    Equipment does not process
                                                    requests.
                                                    [ELSE] Equipment ON-LINE
                                        <--S1,F16   Equipment acknowledges request
                                                    and transitions to OFF-LINE.
                                        <--S6,F11   “Equipment OFF-LINE” event.
Acknowledge                     S6,F12-->
                                                    [END_IF]
```

Host Sets ON-LINE:

```
COMMENTS                         HOST   EQUIPMENT                         COMMENTS

Host requests ON-LINE.          S1,F17-->
                                                    [IF] Equipment is HOST OFF-LINE
                                                    state not active.
                                                    [THEN]
                                        <--S1,F18   Equipment denies request
                                                    (ONLACK = 1).
                                                    [ELSE] Equipment HOST OFF-LINE
                                                    state is active.
                                        <--S1,F18   Equipment acknowledges request
                                                    (ONLACK != 1).
                                        <--S6,F11   “Control state LOCAL (or
                                                     REMOTE)” event.
                                                     (only if ONLACK = 0)
Acknowledge                     S6,F12-->
                                                    [END_IF]
```

5 Data Items

The following sections specify which data items and variable data items are required.

Except for the specified format restrictions, all data items and variable data items follow the definitions contained in SEMI E5.

5.1 *Data Item Restrictions* — The following is a subset of the data items used by SECS-II messages specified in this standard. Each data item listed in this section is restricted in its SEMI E5-defined usage. Most are limited to a single format from their standard list of formats. Data items used by SECS-II messages contained in this document, but which have no restrictions are not duplicated here.

NOTE 21: One data item, ALCD, is restricted in other than its format. Take note of this restriction as described below.

NOTE 22: The equipment supplier shall document any restrictions on length or format of CPNAME. Suppliers shall document the behavior of spaces in a CPNAME. The maximum length of CPNAME shall be 40. This change became effective in September 1995.

ACKC7A	Format: 51
ALCD	Only bit 8 (alarm set/cleared) of the binary byte is used. Bits 1–7, denoting alarm category, are not used.
ALID	Format: 5()
CCODE	Format: 20, 32, 34, 52, 54
CEID	Format: 5()
CPNAME	Format: 20
DATAID	Format: 5()
DATALENGTH	Format: 5()
ECID	Format: 5()
LENGTH	Format: 5()
PPID	Format: 20
RCMD	Format: 20
REPGSZ	Format: 5()
RPTID	Format: 5()
SEQNUM	Format: 52
SMPLN	Format: 5()
SVID	Format: 5()
TEXT	Format: 20
TOTSMP	Format: 5()
TRID	Format: 5()
VID	Format: 5()

5.2 *Variable Item List* — The following variable data items from the Variable Item Dictionary in SEMI E5 are required. Format restrictions are noted.

DVVAL's:

AlarmID Format: 5()
EventLimit
LimitVariable
PPChangeName Format: 20
PPChangeStatus
PPError
RcpChangeName
RcpChangeStatus
TransitionType

ECV's:

EstablishCommunicationsTimeout
MaxSpoolTransmit
OverWriteSpool
TimeFormat

SV's:

AlarmsEnabled
AlarmsSet
Clock
ControlState
EventsEnabled
PPError
PPExecName Format: 0,20
PPFormat
PreviousProcessState
ProcessState
RcpExecName
SpoolCountActual
SpoolCountTotal
SpoolFullTime
SpoolStartTime

6 Collection Events

Table 6.1 provides the list of collection events required to support the capabilities addressed within this standard. Also shown are typical variable data that would most likely be included in the associated collection event report and a reference to the event trigger and to the appropriate section of the standard.

This list does not represent all events that might be needed to properly monitor/control equipment. Many events are unique to the specific equipment characteristics. The needed additions are a matter for other standards and for collaboration between equipment supplier and user.

See Section 5.2 for further detail of variable data items.

Table 6.1 GEM-Defined Collection Events

Event Designation	*Typical Variable Data*	*Reference*
Control-Related Events:		Section 3.3
Equipment OFF-LINE	ControlState, Clock	ON-LINE->OFF-LINE
Control State LOCAL	ControlState, Clock	REMOTE->LOCAL or OFF-LINE->LOCAL
Control State REMOTE	ControlState, Clock	LOCAL->REMOTE or OFF-LINE->REMOTE
Operator Command Issued	OperatorCommand	Operator Activity while REMOTE state is active.
Processing-Related Events:	Note: Any transition in the implemented processing state model must have a corresponding collection event.	Section 3.4
Processing Started	Clock, PreviousProcessState	Entry into EXECUTING state.
Processing Completed	Clock, PreviousProcessState	Normal exit of EXECUTING state.
Processing Stopped	Clock, PreviousProcessState	Result of STOP command from host or operator.
Processing State Change	Clock, ProcessState, PreviousProcessState	Any processing state transition.
Alarm Management Events:		Section 4.3
Alarm$_n$Detected	Clock, AlarmID, AlarmsSet, Associated variable data	ALARM$_n$CLEAR->ALARM$_n$SET
Alarm$_n$Cleared	Clock, AlarmID, AlarmsSet	ALARM$_n$SET->ALARM$_n$CLEAR
Equipment Constant Events:		Section 4.5
Operator Equipment Constant Change	ECID	Operator activity
Limits Monitoring:		Section 4.2.4
Limit Zone Transition$_n$ (separate CEID per variable)	Clock, LimitVariable, EventLimit, Transition Type	Entry into BELOW LIMIT or ABOVE LIMIT states.
Process Program Management Events:		Section 4.6
Process Program Change	PPChangeName, PPChangeStatus	Operator activity
Process Program(s) Selected	PPExecName	Operator/Host activity
Material Movement Events:		Section 4.7
Material Received	Clock	
Material Removed	Clock	
Spooling Events:		Section 4.11
Spooling Activated	SpoolStartTime	SPOOL INACTIVE->SPOOL ACTIVE
Spooling Deactivated	SpoolCountTotal	SPOOL OUTPUT->SPOOL INACTIVE
Spool Transmit Failure	Clock, SpoolCountActual SpoolCountTotal	TRANSMIT SPOOL->NO SPOOL OUTPUT
Terminal Services Events:		Section 4.8
Message Recognition	Clock	Operator
New Execution Recipe Event	RcpChangeName, RcpChangeStatus	Section 4.6.2.2
Execution Recipe Change Event	RcpChangeName, RcpChangeStatus	Section 4.6.2.2
Successful Upload	DataSetName	Sections 4.6.5.3, 4.6.6.3
Bad Upload	DataSetName	Sections 4.6.5.3, 4.6.6.3