



E5-88

SEMI EQUIPMENT COMMUNICATIONS STANDARD 2 MESSAGE CONTENT (SECS-II)

1. Introduction

1.1 *Intent* – The SEMI Equipment Communications Standard Part 2 (SECS-II) defines the details of the interpretation of messages exchanged between intelligent equipment and a host. This specification has been developed in cooperation with the Japan Electronic Industry Development Association Committee 12 on Equipment Communications.

1.1.1 It is the intent of this standard to be fully compatible with SEMI Equipment Communications Standard E4 (SECS-I). It is also the intent to allow for compatibility with alternative message transfer protocols. The details of the message transfer protocol requirements are contained in Section 3.

1.1.2 It is the intent of this standard to define messages to such a level of detail that some consistent host software may be constructed with only minimal knowledge of individual equipment. The equipment, in turn, may be constructed with only minimal knowledge of the host.

1.1.3 The messages defined in the standard support the most typical activities required for IC manufacturing. The standard also provides for the definition of equipment specific messages to support those activities not covered by the standard messages. While certain activities can be handled by common software in the host, it is expected that equipment specific host software may be required to support the full capabilities of the equipment.

1.2 *Overview* – SECS-II gives form and meaning to messages exchanged between equipment and host using a message transfer protocol, such as SECS-I.

1.2.1 SECS-II defines the method of conveying information between equipment and host in the form of messages. These messages are organized into categories of activities, called streams, which contain specific messages, called functions. A request for information and the corresponding data transmission is an example of such an activity.

1.2.2 SECS-II defines the structure of messages into entities called items and lists of items. This structure

allows for a self-describing data format to guarantee proper interpretation of the message.

1.2.3 The interchange of messages is governed by a set of rules for handling messages called the transaction protocol. The transaction protocol places some minimum requirements on any SECS-II implementation.

1.3 *Application* – SECS-II applies to equipment and hosts used in the manufacturing of semiconductor devices. Examples of the activities supported by the standard are transfer of control programs, material movement information, measurement data, summarized test data, and alarms.

1.3.1 The minimum compliance to this standard involves meeting the few constraints outlined in Section 5. It is expected that a given piece of equipment will require only a subset of the functions described in this standard. The number of functions and the selection of functions will depend upon the equipment capabilities and requirements. For each piece of equipment, the exact format for each function provided must be documented according to the form outlined in Section 7.

1.3.2 It is assumed that the equipment will define the messages used in a particular implementation of SECS-II. It is assumed the host will support the equipment implementation.

1.4 *Applicable Documents*

1.4.1 ANSI¹
X3.4-1977 Code for Information Interchange (ASCII)

1.4.2 IEEE²
754 Standard for Binary Floating Point Arithmetic

1.4.3 SEMI Specifications³
E4 SEMI Equipment Communications Standard I – Message Transfer (SECS-I)

¹ ANSI, 1430 Broadway, New York, NY 10018

² IEEE Service Center, 445 Hoe Lane, Piscataway, NJ 08854

³ SEMI, 805 E. Middlefield Road, Mountain View, CA 94043



1.4.4 The Japan Electronic Industry Development Association (JEIDA)
JIS-6226 JIS 8-bit coded character set

2. Selected Definitions

2.1 The following brief definitions refer to sections providing further information.

block – a physical division of a message used by the message transfer protocol. (see 3.3)

conversation – a sequence of related messages. (see 5.4)

conversation timeout – an indication that a conversation has not completed properly. (see 5.4.1)

device ID – a number between 0 and 32767 used in identifying the particular piece of equipment communicating with a host. (see 3.4.1)

equipment – the intelligent system which communicates with a host.

function – a specific message for a specific activity within a stream. (see 4.2)

host – the intelligent system which communicates with the equipment.

interpreter – the system that interprets a primary message and generates a reply when requested. (see 3.2)

item – a data element within a message. (see 6.2)

item format – a code used to identify the data type of an item. (see 4.2)

list – a group of items. (see 6.3)

message – a complete unit of communication. (see 3.2)

message header – information about the message passed by the message transfer protocol. (see 3.4)

multi-block message – a message sent in more than one block by the message transfer protocol. (see 3.3.2)

originator – the creator of a primary message. (see 3.2)

packet – a physical division of a message used by the message transfer protocol. (see 3.3)

primary message – an odd numbered message. Also, the first message of a transaction. (see 3.2, 4.2)

reply – the particular secondary message corresponding to a primary message. (see 3.2)

secondary message – an even numbered message. Also the second message of a transaction. (see 3.2, 4.2)

single-block message – a message sent in one block by the message transfer protocol. (see 3.3.1)

stream – a category of messages. (see 4.1)

transaction – a primary message and its associated secondary message, if any. (see 5.2)

transaction timeout – an indication from the message transfer protocol that a transaction has not completed properly. (see 3.5)

3. The Message Transfer Protocol

3.1 *Intent* – SECS-II is fully compatible with the message transfer protocol defined by SECS-I. It is the intent of this standard to allow for compatibility with alternative message transfer protocols. The purpose of this section is to define the requirements of the interaction between an application using SECS-II and the message transfer protocol. The methods used to implement these requirements are not covered as a part of this standard. The terms used in this standard are those used by SECS-I. Equivalent terms may be different for other message transfer protocols.

3.2 *Messages* – The message transfer protocol is used to send messages between equipment and host. The message transfer protocol must be capable of sending a primary message, indicating whether a reply is requested; and, if a reply is requested, it must be capable of associating the corresponding secondary message or reply message with the original primary message. The term originator will refer to the creator of the original primary message. The term interpreter will refer to the entity that interprets the primary message at its destination and generates a reply when requested.

3.3 *Blocking Requirements* – The message transfer protocol must support the following SECS-II message blocking requirements.

3.3.1 *Single-block messages* – SECS-II requires that certain messages be sent in a single block or single packet by the message transfer protocol. Those messages defined in this standard as single-block SECS-II messages must be sent in a single-block or packet. The method used by the application software to tell the message transfer protocol that a particular message must be sent as a single block is not covered as part of this standard. For compatibility with SECS-I, the maximum length allowed for a single-block SECS-II message is 244 bytes. The minimum requirement for the message transfer protocol is to be able to send single-block SECS-II messages.

3.3.2 *Multi-Block Messages* – For compatibility with SECS-I, SECS-II messages that are longer than 244 bytes are referred to as multi-block messages. Also, certain SECS-II messages are allowed to be multi-block messages even if they otherwise meet the single block length requirements. An application may require an

message data be sent in blocks of specified lengths. The method used by the application software to tell the message transfer protocol how to block a message is not covered as part of this standard.

3.4 Message Header – The message transfer protocol must provide the following information, called the message header, with every message. Only the content of the message header is defined by this standard. The exact format of the message header passed between the application and the message transfer protocol is not covered as part of this standard.

NOTE – in SECS-I, this information is contained in the 10-byte header of each block of a message.

3.4.1 Device ID – The message transfer protocol must be capable of identifying the device ID (0-32767) which indicates the source or destination of a message.

3.4.2 Stream and Function – The message transfer protocol must be capable of identifying to SECS-II a minimum 15-bit message identification code. In SECS-II, messages are identified by a stream code (0-127, 7 bits) and a function code (0-255, 8 bits). Each combination of stream and function represents a distinct message identification.

3.4.3 Reply Requested – The message transfer protocol must be capable of identifying whether a reply is requested to a primary message.

3.5 The Transaction Timeout – It is presumed that the message transfer protocol will notify SECS-II in the event of failure to receive an expected reply message within a specified transaction timeout period.

NOTE – In SECS-I, a transaction timeout occurs if either the reply timeout (T3) is exceeded before the first block of a reply message is received or if the inter-block time-out (T4) is exceeded before an expected block of a multi-block message is received.

3.6 Multiple Open Transactions – This standard allows, but does not require, the support of more than one concurrent open transaction.

4. Streams and Functions

4.1 Streams – A stream is a category of messages intended to support similar or related activities.

4.2 Functions – A function is a specific message for a specific activity within a stream. All the functions used

in SECS-II will follow a numbering convention corresponding to primary and secondary message pairs. All primary messages will be given an odd numbered function code. The reply message function code is determined by adding one to the primary message function code. The even numbered function following a primary message which requests no reply is reserved and is not to be used. Function code 0 is reserved in all streams for aborting transactions as described in Section 7.4.

4.3 Stream and Function Allocation – Some of the stream and function code combinations are reserved for this standard while others are available for user definition. The stream and function codes reserved for this standard are as follows:

In Stream 0, Functions 0-255.

In Streams 1-63, Functions 0-63.

In Streams 64-127, Function 0.

The stream and function codes available for user definition are as follows:

In Streams 1-63, Functions 64-255.

In Streams 64-127, Functions 1-255.

4.3.1 The stream and function code assignment can also be represented by the diagram shown in Figure 1.

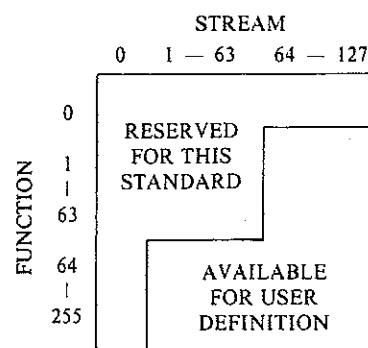


Figure 1
Stream and Function Allocation



4.3.2 The reserved codes assigned by this standard are listed in Section 7. It is recognized that there will be user needs beyond the specific definitions given in this standard. In these situations, the streams and functions reserved for user definition should be used subject to the guidelines for minimum compliance outlined in Section 5.

5. Transaction and Conversation Protocols

5.1 *Intent* – For an implementation to be in compliance with SECS-II it must meet the minimum transaction requirements outlined in this section. The conversation protocols serve to further define the use and interaction between transactions.

5.2 *Transaction Definition* – A transaction forms the basis for all information exchanges in SECS-II. A transaction consists of either a primary message for which no reply is requested, or a primary message which requests a reply together with its corresponding secondary message. Secondary messages cannot request a reply.

5.3 *Transaction Level Requirements* – The following are the requirements to comply with the SECS-II protocol at the transaction level:

1. Respond to S1,F1 with S1,F2 as described in Section 7.5.
2. For any received message that cannot be processed by the equipment, send the appropriate error message on Stream 9. As described in Section 7.13, S9-F1, F3, F5, F7, or F11 are possible.
3. Format any other supported messages according to Section 7.
4. Upon detection of a transaction timeout at the equipment send S9,F9 to the host.
5. Upon receipt of function 0 as a reply to a primary message, terminate the related transaction. No error message should be sent to the host by the equipment.

5.4 *Conversation Protocols* – A conversation is a series of one or more related transactions used to complete a specific task. A conversation should include all transactions necessary to accomplish the task and leave both the originator and interpreter free of resource commitments at its conclusion.

5.4.1 *Conversation Timeout* – A conversation timeout is used to indicate that a conversation has not completed properly. A conversation timeout is application dependent and the methods used for detecting conversation timeouts are not covered as part of this standard. A conversation timeout will terminate further action on the conversation and will allow for the clearing of any com-

mitted resources. Upon detection of a conversation timeout at the equipment, S9,F13 should be sent to the host.

5.4.2 *Types of Conversations*. There are seven types of conversations which characterize all information exchanges in SEC-II:

1. A primary message with no reply is the simplest conversation. This message must be a single-block SEC-II message. The originator must assume that the interpreter responds to the message. This conversation is used where the originator can do nothing if the message is rejected.
2. If the interpreter has data that the originator wants, the data are requested with a primary message and the data returned to the originator as a reply message. It is assumed that the originator requesting the data is prepared to receive the amount of data returned. This is the request/data conversation.
3. If the originator wishes to send data in a single-block SECS-II message to the interpreter, then the originator sends the data and expects an acknowledgment from the interpreter. This is the send/acknowledge conversation.
4. If the originator has a multi-block SECS-II message to send for a particular exchange, then the originator must receive permission from the interpreter prior to sending the data. The first transaction requests permission to send and the interpreter either grants or denies permission. If permission is granted, the originator sends the data and the interpreter replies appropriately. This is the inquire/grant/send/acknowledge conversation. Between the inquire and the send, the interpreter may commit some resources in preparation for the data. Consequently, a conversation timeout may be set by the interpreter at a time dependent upon the application, at which time the interpreter will free its resources and send an S9,F13 error message to the originator. Note that under the definition of S9,F13 in this standard, only the equipment should generate an error message to the host under these conditions.
5. There is a conversation related to the transfer of file data between equipment and host. This conversation is described in detail in Stream 11. (see Section 7.15)
6. There is a conversation related to the handling of material between equipment. This conversation is described in detail in Stream 4. (see Section 7.8)
7. The originator may request information from the interpreter which requires some time to obtain; e.g. operator input is required. The first transaction requests the information and the interpreter re-



sponds in 1 of 3 ways: 1) the information is returned, 2) the interpreter indicates that the information cannot or will not be obtained, or 3) the interpreter indicates that the information will be obtained and returned in a subsequent transaction, as specified for this conversation. For case number 3, the interpreter will initiate the subsequent transaction when the information is available

Case 3 is the request/acknowledge/send/acknowledge transaction.

The originator of the request/acknowledge/send/acknowledge conversation may commit some resources in anticipation of the send/acknowledge transaction. Consequently, a conversation timeout may be set by the originator at a time dependent on the application. On timeout, the originator will free its resources and restart the conversation with the 'request', or send an S9, F13 error message. Note that under the definition of S9, F13 in this standard, only the equipment should generate an error message to the host under these conditions.

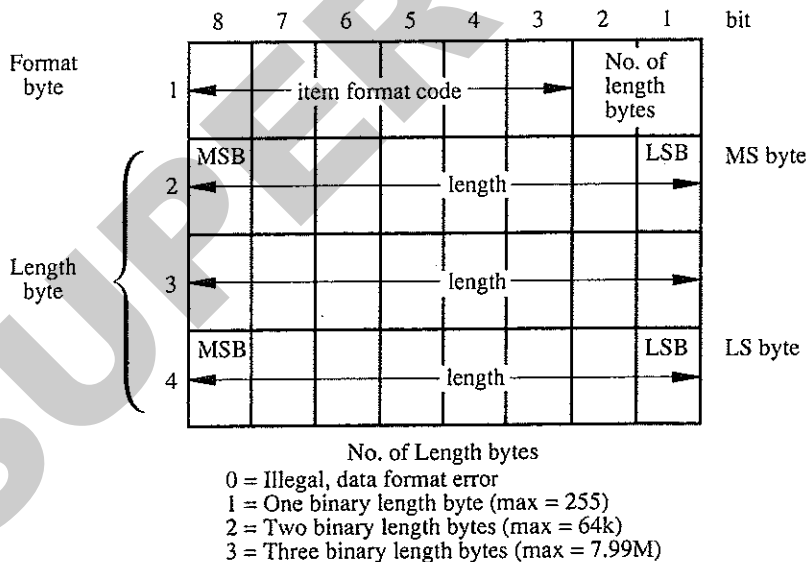
5.4.3 The key words, request, data, send, acknowledge,

inquire, and grant are used in the function names as an aid to understanding the relationship between the messages and the conversation. Single message transactions do not use these words.

6. Data Structures

6.1 *Intent* – All information transmitted according to this standard will be formatted using two data structures, items and lists. These data structures define the logical divisions of the message, as distinct from the physical divisions of the message transfer protocol. They are intended to provide a self-describing internal structure to messages passed between equipment and host.

6.2 *Item* – An item is an information packet which has a length and format defined by the first 2, 3, or 4 bytes of the item. These first bytes are called the item header (IH). The item header consists of the format byte and the length byte(s) as shown in Figure 2. Bits one and two of the item header tell how many of the following bytes refer to the length of the item. This feature allows for long items without requiring the byte overhead for shorter items. The item length refers to the number of bytes following the item header, called the item body (IB), which is the actual data of the item. The item length



Length refers to the number of bytes except in a list format where it refers to the number of elements (items or lists) in the list.

Figure 2
Item and List Header



Table 1
Item Format Codes

Format Code		Meaning
Binary bit-876543	octal	The data after the heading has the following form.
000000	00	LIST (length in elements)
001000	10	Binary
001001	11	Boolean
010000	20	ASCII ¹
010001	21	JIS-8 ¹
011000	30	8-byte integer (signed) ²
011001	31	1-byte integer (signed)
011010	32	2-byte integer (signed) ²
011100	34	4-byte integer (signed) ²
100000	40	8-byte floating point ³
100100	44	4-byte floating point ³
101000	50	8-byte integer (unsigned) ²
101001	51	1-byte integer (unsigned)
101010	52	2-byte integer (unsigned) ²
101100	54	4-byte integer (unsigned) ²

¹Non-printing characters are equipment specific.

²Most significant byte sent first.

³IEEE P754. The byte containing the sign bit is sent first.

NOTE – Changes in integer format codes may conflict with earlier implementations.

refers only to the item body not including the item header, so the actual number of bytes in the message for one item is the item length plus 2, 3, or 4 bytes for the item header. All bytes in the item body are in the format specified in the format byte.

6.2.1 A zero length in the format byte is illegal and produces an error. A zero length in the item length bytes has a special meaning as defined in the detailed message definitions.

6.2.2 Bits 3 through 8 of the format byte of the item header define the format of the data which follows. Of the 64 possible formats, fourteen are defined as shown in Table 1. Format code 0 is called a list and is defined in E5 6.3. The remaining 14 item formats define unspecified binary, code 10 (octal); Boolean, code 11 (octal); ASCII character strings, code 20 (octal); JIS-8 character strings, code 21 (octal); signed integer, codes 30, 31, 32, 34 (octal); floating point, codes 40, 44 (octal); and unsigned integer, codes 50, 51, 52, 54 (octal). These formats are used for groups of data that have the same representation in order to save repeated item headers. Signed integers will be two's complement values. Floating point numbers will conform to the IEEE standard 754. Boolean values will be byte quantities with zero being equivalent to false and non-zero being equivalent to true.

6.3 *List* – A list is an ordered set of elements, where an element can be either an item (6.2) or a list. The list header (LH) has the same form as an item header with format type 0. However, the length bytes refer to the number of elements in the list rather than to the number of bytes. The list structure allows grouping items of related information which may have different formats into a useful structure.

6.3.1 A zero length in the format byte is illegal and produces an error. A zero length in the list length bytes has a special meaning as defined in the detailed message definitions.

6.4 *Example Data Structures* – The data structures for different types of items are illustrated in the following examples.

(a) An item contains one binary code 10101010.
bit
87654321

00100001 length byte
00000001 1 byte long
10101010 data byte

(b) An item contains three ASCII characters ABC.
01000001 Item ASCII, 1 length byte



00000011 Three bytes long
 01000001 ASCII A
 01000010 ASCII B
 01000011 ASCII C

(c) An item contains three binary numbers in 2-byte form. 01101001 Item, 2-byte integers 00000110 6 bytes total (6/2=3 integers)

xxxxxxx MSByte number x
 xxxxxx LSByte number x
 yyyyyyy MSByte number y
 yyyyyyy LSByte number y
 zzzzzzz MSByte number z
 zzzzzzz LSByte number z

(d) An item contains one 4-byte IEEE floating point number.

10010001 Item, 4-byte floating point
 00000100 4 bytes (4/4=1 number)
 ffff ffff
 ffff ffff Floating point number
 ffff ffff in IEEE 754
 ffff ffff

(e) A message is sent from device 42 telling the host that the temperature at point T1 has exceeded a preset process limit. The message ID is stream 5, function 1, and the data consists of a list of three items. The first item is a code for the alarm set and the alarm category code. The second item is the equipment-specific alarm number for this alarm, e.g. 17. The third item is a string of text giving a brief description of the alarm, e.g. "T1 HIGH." No reply is requested. The complete message including the header is as follows:

10000000 R=1 (to the host)
 01000010 Device Code=42
 00000101 Stream 5, W=0 10 byte
 00000001 Function 1 Header
 10000000 E=1
 00000001 Block 1
 00000000
 00000000 System bytes=0
 00000000
 00000001 List
 00000011 3 Elements
 00100001 Item, 1-byte, next byte length
 00000001 1 byte long
 10000100 Alarm set, category 4
 01100101 Item, 1-byte integer, next byte length
 00000001 1 byte long
 00010001 Alarm 17
 01000001 Item, ASCII, next byte length
 00000111 7 characters
 01010100 ASCII T
 00110001 ASCII 1
 00100000 ASCII space
 01001000 ASCII H
 01001001 ASCII I
 01000111 ASCII G
 01001000 ASCII H

The entire message contains 17 bytes of data, 10 bytes of header, 1 byte length (not shown), and 2 bytes checksum (not shown) for a total of 30 bytes. At 9600 baud transmission, the message would be sent in 31 milliseconds.



6.5 Data Item Dictionary

ABS

Format: 10

Any binary string

Where Used: S2F25, S2F26

ACDS

Format: 32, 52

After Command Codes

Vector of all command codes which defined command must succeed within the same block.

Where Used: S7F22

ACKC13

Format: 10

Return code for secondary messages, 1 byte

- 0 = O.K.
- 1 = ERROR: Try later
- 2 = ERROR: Unknown Data Set Name
- 3 = ERROR: legal Checkpoint value
- 4 = ERROR: Too many open Data Sets
- 5 = ERROR: Data set open too many times
- 6 = ERROR: No open Data Set
- 7 = ERROR: Cannot continue
- 8 = ERROR: End of Data
- > 8 = other error
- 9-63 reserved

Where Used: S13F2, S13F4, S13F6, S13F8

ACKC3

Format: 10

Acknowledge code, 1 byte

- 0 = accepted
- 1-63 reserved

Where Used: S3F6, S3F8, S3F10

ACKC5

Format: 10

Acknowledge code, 1 byte

- 0 = accepted
- > 0 = error, not accepted
- 1-63 reserved

Where Used: S5F2, S5F4



ACKC6

Format: 10

Acknowledge code, 1 byte

- 0 - accepted and done
- > 0 - error, not done
- 1-63 reserved

Where used: S6F2, S6F4, S6F10

ACKC7

Format: 10

Acknowledge code, 1 byte

- 0 - Accepted
- 1 - Permission not granted
- 2 - Length error
- 3 - Matrix overflow
- 4 - PPID not found
- 5 - Mode unsupported
- > 5 - Other error
- 6-63 reserved

Where Used: S7F4, S7F12, S7F14, S7F16, S7F18, S7F24, S7F32

ACKC7A

Format: 31, 51

Acknowledge Code, 1 byte

- 0 = Accepted
- 1 = MDLN is inconsistent
- 2 = SOFTREV is inconsistent
- 3 = Invalid CCODE
- 4 = Invalid PPARM value
- 5 = Other error (described by ERRW7)
- 6-63 reserved

Where Used: S7F27

ALCD

Format: 10

Alarm code byte

- bit 8 = 1 means alarm set
- bit 8 = 0 means alarm cleared
- bits 7 - > 1 is alarm category code
 - 0 = not used
 - 1 = Personal safety
 - 2 = Equipment safety
 - 3 = Parameter control warning
 - 4 = Parameter control error
 - 5 = Unrecoverable error
 - 6 = Equipment status warning
 - 7 = Attention flags
 - 8 = Data integrity
 - > 8 = other categories
- 9-63 reserved

Where used: S5F1, S5F6



ALED

Format: 10

Alarm enable/disable code, 1 byte

bit 8 = 1 means enable alarm

bit 8 = 0 means disable alarm

Where Used: S5F3

ALID

Format: 3(),5()

Alarm identification

Where Used: S5F1, S5F3, S5F5, S5F6

ALTX

Format: 20

Alarm text limited to 40 characters

Where Used: S5F1, S5F6

BCDS

Format: 32, 52

Before Command Codes

Vector of all command codes which defined command must precede within the same block.

Where Used:

BCEQU

Format: 20, 51

Bin code equivalents

Array of all codes that are to be processed. Must be the same format as BINLT and NULBC. Zero length indicates all codes be sent.

Where Used: S12F4

BINLT

Format: 20, 51

The Bin List

Is an array of bin values. Format must be the same as used in NULBC and BCEQU.

Where Used: S12F7, S12F9, S12F11, S12F14, S12F16, S12F18

BLKDEF

Format: 31, 51

Block Definition

Blocks define the range for before/after code checking (specified by BCDS, IBCDS, NBCDS, ACDS, IACDS, and NACDS). BLKDEF specifies whether the command being defined starts a block (+1), terminates a block (-1) or is within the body of a block (0). All other values are invalid. The outermost block of a process program is implicit and is bounded by the first and last command of the process program. Before/after checks for a particular command are performed with all other commands in the same block and at the same nesting level. For the purpose of before/after checking, a set of commands making up a block is considered to be a single body command of its containing block. This command has the before/after restrictions of the start block command which begins the block.

Where Used: S7F22



BPD
Boot program Data

Format: 10

Where Used: S8F2

BYTMAX
Byte Maximum

Format: 3(),5()

Maximum length of process program. A value of zero indicates no limit. Negative values are invalid.

Where Used: S7F22

CCODE
Command Code

Format: 32, 52

Each command code corresponds to a unique process operation the machine is capable of performing.

Where Used: S7F22, S7F23, S7F26, S7F31

CEID
Collected event ID

Format: 3(), 20, 5()

Where Used: S6F3, S6F8, S6F9

CKPNT
Checkpoint as defined by the sending system

Format: 54

Where Used: S13F3, S13F6

CMDA
Command acknowledge code

Format: 31, 51

0 = completed or done
1 = command does not exist
2 = cannot perform now
> 2 = other equipment specific error
3-63 reserved

Where Used: S2F22, S2F28

CMDMAX
Command Maximum

Format: 3(),5()

Maximum number of commands to be allowed in a process program. A value of zero indicates no limit. Negative values are invalid.

Where Used: S7F22



CNAME

Format: 20

Command Name L = 16 characters

Text string unique among other CNAMEs in PCD which describes the processing done by the equipment for the corresponding CCODE.

Where Used: S7F22

COLCT

Format: 5()

Column count in die increments

Where Used: S12F1, S12F4

CSAACK

Format: 10

Equipment Acknowledgement code, 1 byte

- 0 = everything correct
- 1 = busy
- 2 = invalid spid
- 3 = invalid data
- > 3 = equipment specific error
- 4-63 reserved

Where Used: S2F8

DATAID

Format: 3(), 20, 5()

Data ID

Where Used: S6F3, S6F5, S6F7, S6F8, S6F9

DATALLENGTH

Format: 3(), 5()

Total bytes to be sent

Where Used: S6F5

DATLC

Format: 51

Data location

Location of invalid data represented in bytes, measured from the start of the message in question excluding all header bytes.

Where Used: S12F9

DSID

Format: 3(), 20, 5()

Data set ID

Where Used: S6F3, S6F8, S6F9



DSNAME

Format: 20

The name of the Data Set

The minimum length is zero. The maximum is 50 characters.

Where Used: S13F1, S13F2, S13F3, S13F4

DSPER

Format: 20

Data sample period

hhmmss, 6 bytes

Where Used: S2F23

DUTMS

Format: 20

Die Units of Measure

Use units description per E5 Section 9.

Where Used: S12F1, S12F4

DVNAME

Format: 3(),20,5()

Data value name

Where Used: S6F3, S6F8

DVVAL

Format: 3(),20,4(),5()

Data value

Where Used: S6F3, S6F8, S6F9

EAC

Format: 10

Equipment acknowledge code, 1 byte

0 = Acknowledge

1 = Denied. At least one constant does not exist

2 = Denied. Busy

3 = Denied. At least one constant out of range

> 3 = Other equipment specific error

4-63 reserved

Where Used: S2F16

ECDEF

Format: 3(),20,5()

Equipment constant default value

Where Used: S2F30



ECID
Equipment Constant ID

Format: 3(),20,5()

Where Used: S2F13, S2F15, S2F29, S2F30

ECMAX
Equipment constant maximum value

Format: 3(),20,5()

Where Used: S2F30

ECMIN
Equipment constant minimum value

Format: 3(),20,5()

Where Used: S2F30

ECNAME
Equipment constant name

Format: 20

Where Used: S2F30

ECV
Equipment Constant Value

Format: 3(),20,5()

Where Used: S2F14, S2F15

EDID
Expected data Identification

Format: 10,3(),20,5()

Three possible responses.

MEXP EDID
S02F03, <SPID>
S03F13, <PTN>
S07F03, <PPID>

Where Used: S9F13

EMID
Equivalent material ID (16 bytes maximum)

Format: 10, 20

Where Used: S3F9

EPD
Executive program data

Format: 10

Where Used: S8F4

**ERRW7**

Format: 20

Text string describing error found in process program

Where Used: S7F27

FFROT

Format: 52

Film Frame Rotation

In degrees from the bottom CW (Bottom equals zero degrees). Zero length indicates not used.

Where Used: S12F1, S12F3

FILDAT

Format: 10, 20

Data from the Data Set

The maximum length is the RECLen from Open Data Set Data (DSOD).

Where Used: S13F6

FNLOC

Format: 52

Flat/Notch Location

In degrees from the bottom CW (Bottom equals zero degrees). Zero length indicates not used.

Where Used: S12F1, S12F3, S12F4

FRMLEN

Format: 3(),5()

Formatted Process Program Length

If greater than zero, indicates PPID is available as a formatted process program and its length in bytes. If zero, the PPID is not available as a formatted process program. Negative values are invalid.

Where Used: S7F34

GRANT

Format: 10

Grant code, 1 byte

- 0 = positive response, load OK
- 1 = busy, try again
- 2 = no space
- 3 = duplicate name
- > 3 = equipment specific error code
- 4-63 reserved

Where Used: S2F1



GRANT6

Format: 10

Permission to send, 1 byte

- 0 = Permission granted
- 1 = Busy, try again
- 2 = Not interested
- > 2 = Other errors
- 3-63 reserved.

Where Used: S6F6

GRNTI

Format: 10

Grant code, 1 byte

- 0 = Positive response, transfer ok
- 1 = Busy, try again
- 2 = No space
- 3 = Map too large
- 4 = Duplicate ID
- 5 = Material ID not found
- 6 = Unknown map format
- > 6 = Error
- 7-63 reserved

Where Used: S12F6

HANDLE

Format: 3(), 5()

Logical unit or channel

Where Used: S13F3, S13F4, S13F5, S13F6, S13F7, S13F8

IACDS

Format: 32, 52

Immediately After Command Codes

Vector of all command codes which defined command must immediately succeed within the same block.

Where Used: S7F22

IBCDS

Format: 32, 52

Immediately Before Command Codes

Vector of all command codes which defined command must immediately precede within the same block.

Where Used: S7F22



IDTYP

Format: 10

Id type

- 0 = Wafer Id
- 1 = Wafer Cassette Id
- 2 = Film Frame Id
- > 2 = Error
- 3-63 Reserved

Where Used: S12F1, S12F3, S12F4, S12F5, S12F7, S21F9, S12F11, S12F13, S12F14, S12F15, S12F16, S12F17, S12F18

LENGTH

Format: 3(),5()

Length of the service program or process program in bytes

Where Used: S2F1, S7F1, S7F29

LLIM

Format: 3(),4(),5()

Lower limit for numeric value.

Where Used: S7F22

LOC

Format: 10

Material location code, 1 byte

Where Used: S2F27, S3F2

MAPER

Format: 10

Map Error

- 0 = ID not found
- 1 = Invalid Data
- 2 = Format error
- > 2 = Invalid error
- 3-63 Reserved

Where Used: S12F19

MAPFT

Format: 10

Map data format type

- 0 = Row format
- 1 = Array format
- 2 = Coordinate format
- > 2 = Error
- 3-63 Reserved

Where Used: S12F3, S12F5



MDACK

Format: 10

Map data acknowledge

- 0 = Map received
- 1 = Format error
- 2 = No ID match
- 3 = Abort/discard map
- > 3 = Error
- 4-63 Reserved

Where Used: S12F8, S12F10, S12F12

MDLN

Format: 20

Equipment Model Type, 6 bytes max

Where Used: S1F2, S7F22, S7F23, S7F26, S7F31

MEXP

Format: 20

Message expected in the form SxxFyy where x is stream and y is function

Where Used: S9F13

MF

Format: 10, 20

Material format code 1 byte for Format 10

Items with format 10 will be encoded as follows:

- 1 = quantities in wafers
- 2 = quantities in cassette
- 3 = quantities in die or chips
- 4 = quantities in boats
- 5 = quantities in ingots
- 6 = quantities in leadframes
- 7 = quantities in lots
- 8 = quantities in magazines
- 9 = quantities in packages
- 10 = quantities in plates
- 11 = quantities in tubes
- 12 = quantities in waferframes
- 13-63 reserved

Items with format 20 will be a unit identifier for one of the special SECS generic units, as specified in Section 9.

Where Used: S3F2, S3F4, S3F5, S3F7

MHEAD

Format: 10

SECS message block header associated with message block in error

Where Used: S9F1, S9F3, S9F5, S9F7, S9F11



MID
Material ID

Format: 10, 20

16 Characters maximum

Where Used: S2F27, S3F2, S3F4, S3F7, S3F9, S4F1, S4F3, S4F5, S4F7, S4F9, S4F11, S4F13, S4F15, S4F17, S7F7, S7F8, S7F10, S7F11, S7F13, S7F35, S7F36, S12F1, S12F3, S12F4, S12F5, S12F7, S12F9, S12F11, S12F13, S12F14, S12F15, S12F16, S12F17, S12F18

MIDAC

Format: 10

Material ID Acknowledge Code, 1 byte

- 0 = accepted
- 1 = Invalid port number
- 2 = material is not present at identified port
- > 2 = error
- 3-63 reserved

Where Used: S12F14

MIDRA

Format: 10

Material ID Request Acknowledge, 1 byte

- 0 = acknowledge, MID follows
- 1 = acknowledge, will not send MID
- 2 = acknowledge, will send MID later in S3, F13
- 3-63 reserved

Where Used: S3F12, S3F14

MLCL

Format: 5()

Message length

Defined by message size in (bytes).

Where Used: S12F4, S12F5

MMODE

Format: 10

Matrix mode select, 1 byte

- 1 = Host source mode
Use S7F7 and S7F8 to define the process program to be used.
- 2 = Local source mode
Use the matrix defined by S7F9, S7F10, S7F11 and S7F13 to define the process program to be used. The equipment will initialize to local source mode at power up. The equipment will default to local source mode from host source mode in the event of loss of communication with the host.
- 3 = Host immediate mode
Use the current process program for all material unless changed by S7F1-F4. The timing of the mode change is equipment specific.

Where Used: S7F15



NACDS Format: 32, 52
Not After Command Codes
Vector of all command codes which defined command may not succeed within the same block.

Where Used: S7F22

NBCDS Format: 32, 52
Not Before Command Codes
Vector of all command codes which defined command may not precede within the same block

Where Used: S7F22

NULBC Format: 20, 51
Null bin code value
This value is the bin code value that is used for no die at a location. (For X/Y coordinate format the ASCII value is a value with "n" length. For all other map formats, ASCII is a single byte per bin with "n" length per item thus the total number of bins is the length. i.e. Length "n" = 10 for ASCII format is 10 single byte bin codes.) The format used must be the same as the one used for BINLT and BCEQU. Zero length indicates not used.

Where Used: S12F1, S12F3, S12F4

ORLOC Format: 10
Origin Location
implicit value of (0,0)
0 = Center die of wafer
(Determined by:
$$\frac{\text{row or column count} + 1}{2}$$

truncated)
1 = Upper right
2 = Upper left
3 = Lower left
4 = Lower right
> 4 = Error
5-63 Reserved
Zero length indicates not available

Where Used: S12F1, S12F3, S12F4



PDFLT

Format: 11,20,3(),4(),5()

Parameter Default Value

Specifies default value and data type of parameter. If no defaults are provided, item data length will be zero. For numeric or Boolean data, the default item may be a multi-valued vector if the parameter itself can be a vector.

If RQPAR is false:

Position of data in a default item is significant. When obtaining a default value to be used in the Nth position of a vector parameter, the value is obtained from the Nth position of the default item. If the default item has L entries, no default value will be provided for the L+1,...,PMAX parameter entries.

If RQPAR is true:

The length of the default vector (L) specifies the minimum number of entries which must be entered for the parameter. If > PMAX only PMAX entries are required.

Where Used: S7F22

PFGD

Format: 10

Predefined form code, 1 byte

Where Used: S6F9

PMAX

Format: 3(), 5()

Parameter Count Maximum

Maximum amount of data to be accepted by the host for this parameter. When a conflict arises between value of PMAX and length of PDFLT, PMAX takes precedence.

For numeric and Boolean parameters:

PMAX < 0 is invalid.

PMAX = 0 specifies there is no upper bound.

PMAX = 1 specifies a single value is expected.

PMAX > 1 specifies a vector of values is expected with a maximum of PMAX entries.

For string parameters:

PMAX < 0 is invalid

PMAX = 0 specifies there is no upper bound.

PMAX > 0 maximum length of parameter string.

Where Used: S7F22

PNAME

Format: 20

Parameter Name <= 16 characters

Text string identifying the parameter value expected by its parent process command.

Where Used: S7F22

PPARMn

Format: 11,20,3(),4(),5()

Process Parameter

Numeric or Boolean SECS data item, single or multiple value, or text string which provides information required to complete the process command to which the parameter refers.

Where Used: S7F23, S7F26, S7F31



PPBODY

Format: 10,20,3(),5()

Process program body

The process program describes to the equipment, in its own language, the actions to be taken in processing the material it receives.

Where Used: S7F3, S7F6, S7F36

PPGNT

Format: 10

Process program grant status, 1 byte

- 0 = OK
- 1 = Already have
- 2 = No space
- 3 = Invalid PPID
- 4 = Busy, try later
- 5 = Will not accept
- > 5 = Other error
- 6-63 reserved

Where Used: S7F2, S7F30

PPID

Format: 10, 20

Process program ID

Limited to 16 bytes.

The format used in the PPID will be host dependent. For internal use of the equipment, the PPID can be treated as a unique binary pattern. If the local equipment is not prepared to display the transmitted code, the display should be in hexadecimal form.

Where Used: S2F27, S7F1, S7F3, S7F5, S7F6, S7F8, S7F10, S7F11, S7F13, S7F17, S7F20, S7F23, S7F25, S7F26, S7F27, S7F31, S7F33, S7F34, S7F36

PRAXI

Format: 10

Process axis

- 0 = Rows (X-axis), top, increasing
- 1 = Rows (X-axis), top, decreasing
- 2 = Rows (X-axis), bottom, increasing
- 3 = Rows (X-axis), bottom, decreasing
- 4 = Columns (Y-axis), left, increasing
- 5 = Columns (Y-axis), left, decreasing
- 6 = Columns (Y-axis), right, increasing
- 7 = Columns (Y-axis), right, decreasing
- > 7 = Error
- 8-63 Reserved

Where Used: S12F1, S12F3



PRDCI
Process Die Count
Number of die to be processed or number of die which have been processed. Zero length indicates not used.

Format: 5()

Where Used: S12F1, S12F4

PTN
Material Port number, 1 byte

Format: 10

Where Used: S3F11, S3F12, S3F13, S4F1, S4F3, S4F5, S4F7, S4F9, S4F11, S4F13, S4F15, S4F17

QUA
Quantity in format, 1 byte

Format: 10

Where Used: S3F2, S3F4, S3F5, S3F7

RAC
Reset acknowledge, 1 byte
0 = reset to be done
1 = reset denied
> 1 = other errors
2-63 reserved

Format: 31, 51

Where Used: S2F20

RCMD
Remote command code or string

Format: 20, 31, 51

Where Used: S2F21

READLN
Maximum length to read

Format: 3(), 5()

Where Used: S13F5

RECLN
Maximum length of a Discrete record.

Format: 3(), 5()

Where Used: S13F4

REPGSZ
Reporting group size

Format: 20, 3(), 5()

Where Used: S2F23



RESC
Resolution code for numeric data
1 = Absolute. Value may be specified to nearest increment of RESV.
2 = Significant Digits. Value may be specified with no more significant digits than RESV allows.

Format: 31, 51

Where Used: S7F22

RESV
Resolution value for numeric data
If RESC = 1
then RESV contains smallest increment allowed for parameter. Format: 3(), 4(), 5()
If RESC = 2
then RESV contains number of significant digits allowed for parameter value. Format: 3(), 5()

Where Used: S7F22

RIC
Reset code, 1 byte
0 = not used
1 = power up reset
> 1 other reset conditions
2-63 reserved

Format: 31, 51

Where Used: S3F19

ROWCT
Row count in die increments

Format: 5()

Where Used: S12F1, S12F4

RPSEL
Reference Point Select
(number of reference point from 0-n).

Format: 51

Where Used: S12F1, S12F4

RQCMD
Required Command
True - Command must be specified
False - Command is optional

Format: 11

Where Used: S7F22



RQPAR

Format: 11

Required Parameter

True- Parameter must be specified
False- Parameter is optional

Where Used: S7F22

RRACK

Format: 10

Request to Receive Acknowledge code, 1 byte

0 = acknowledge, OK (note that 'OK' differs from 'ready')
1 = invalid port number
2 = requested material is not at identified port
3 = busy. Try again
4 = sender does not have permission to perform this operation
5-63 reserved

Where Used: S4F18

RSACK

Format: 10

Ready to Send Acknowledge code, 1 byte

0 = acknowledge, OK (note that 'OK' differs from 'ready')
1 = invalid port number
2 = port is already occupied
3 = busy, unable to move material at this time. Try again
4 = receiver does not have permission to perform this operation
5-63 reserved

Where Used: S4F2

RSINF

Format: 3()

Starting location for row or column

This item consists of 3 values (x,y direction). If direction value is "-" it equals decreasing direction.
If the value is "+" it equals increasing direction. Direction must be a non-zero value.

Where Used: S12F7, S12F14

RTYPE

Format: 3(),5()

Type of record

0 = Stream
1 = Discrete
2-63 = reserved

Where Used: S13F4



SDACK

Format: 10()

Map set-up data acknowledge

- 0 = Received data
- > 0 = Error
- 1-63 Reserved

Where Used: S12F2

SDBIN

Format: 10

Send bin information flag

- 0 = Send bin information
- 1 = Don't send bin information
- > 1 = Error
- 2-63 Reserved

Where Used: S12F17

SEQNUM

Format: 5(),3()

Command Number

Value which identifies a unique process program command by its position in the list of commands relative to the first. For the first command of the process program, SEQNUM is 1.

Where Used: S7F27

SFCD

Format: 10

Status form code, 1 byte

Where Used: S1F5, S1F7

SHEAD

Format: 10

Stored header related to the transaction timer

Where Used: S9F9

SMPLN

Format: 3(),5()

Sample number

Where Used: S6F1

SOFTREV

Format: 20

Software revision code 6 bytes maximum

Where Used: S1F2, S7F22, S7F23, S7F26, S7F31



SPAACK

Format: 10

Equipment acknowledgement code, 1 byte

0 = everything correct

1 = Invalid data

> 1 = equipment specific error

2-63 reserved

Where Used: S2F4

SPD

Format: 10

Service program data

Where Used: S2F3, S2F6

SPID

Format: 20

Service program ID, 6 characters

Where Used: S2F1, S2F5, S2F6, S2F9, S2F12

SPR

Format: Device dependent

Service program results

Where Used: S2F10

STEMP

Format: 20

String template

ASCII text string acceptable to equipment as a parameter value. A data string matches a template string if the data string is at least as long as the template and each character of the data string matches the corresponding character of the template. A null list indicates all user data is acceptable to the machine.

Where Used: S7F22

STIME

Format: 20

Sample time, 12 bytes

yymmddhhmmss

Where Used: S6F1

STRP

Format: 3()

Starting position in die coordinate position. Must be in (X,Y) order

Where Used: S12F9, S12F16



SV

Format: 20,3(),4(),5()

Status variable value

Where Used: S1F4, S6F1

SVID

Format: 20, 3(), 5()

Status variable ID

Status variables may include any parameter that can be sampled in time such as temperature or quantity of a consumable.

Where Used: S1F3, S1F11, S1F12, S2F23

SVNAME

Format: 20

Status Variable Name

Where Used: S1F12

TEXT

Format: 10,20,3(),5()

A single line of characters

Where Used: S10F1, S10F3, S10F5, S10F9

TIAACK

Format: 10

Equipment acknowledgement code, 1 byte

- 0 = everything correct
- 1 = too many SVIDs
- 2 = no more traces allowed
- 3 = invalid period
- > 3 = equipment specific error
- 4-63 reserved

Where Used: S2F24

TID

Format: 10

Terminal number, 1 byte

- 0 = single or main terminal
- >1 = additional terminals at the same equipment

Where Used: S10F1, S10F3, S10F5, S10F7

TIME

Format: 20

Time of day, 12 bytes yymmddhhmmss

Where Used: S2F18



TOTSMP

Format: 20,3(),5()

Total samples to be made

Where Used: S2F23

TRID

Format: 20, 3(), 5()

Trace request ID

Where Used: S2F23, S6F1

TSIP

Format: 10

Transfer status of input port, 1 byte

- 1 = Idle state
- 2 = Prep state
- 3 = Track on state
- 4 = Stuck in Receiver state
- 5-63 reserved

Where Used: S1F10

TSOP

Format: 10

Transfer status of output port, 1 byte

- 1 = Idle state
- 2 = Prep state
- 3 = Track on state
- 4 = Stuck in Sender state
- 5 = Completion state
- 6-63 reserved

Where Used: S1F10

TTC

Format: 3(),5()

Time to completion

Where Used: S3F4

ULIM

Format: 3(),4(),5()

Upper limit for numeric value.

Where Used: S7F22



UNFLEN

Format: 3(),5()

Unformatted Process Program Length

If greater than zero, indicates PPID is available as an unformatted process program and its length in bytes. If zero, the PPID is not available as an unformatted process program. Negative values are invalid.

Where Used: S7F34

UNITS

Format: 20

Units identifier

As allowed by E5 section 9.

Where Used: S1F12, S2F30, S7F22

XDIES

Format: 4(),5()

X-axis die size (index)

Where Used: S12F1, S12F4

XYPOS

Format: 3()

X and Y Coordinate Position. Must be in (X,Y) order.

Where Used: S12F11, S1218

YDIES

Format: 4(),5()

Y-axis die size (index)

Where Used: S12F1, S12F4

SUPERSEDED

7. Message Detail

7.1 *Intent* – This section defines a number of specific functions in different streams which can be used as a basis for communication between host and equipment. The functions are defined in the form of transaction message pairs according to the transaction level requirements specified in Section 5.

7.2 The functions are described in a standard form

which involves specification of the number, name, single or multiple block, direction of communication, nature of reply required, description, variable definition, and the detailed structure of the message in terms of lists and items. Double lines separate streams and single lines separate transactions to aid readability.

7.3 The abbreviations used in each transaction are as follows:

Define:	Variables to be defined.	Required formats (octal) 3() means all formats starting with 3
	S = single, M = multi-block	
	short mnemonic	valid direction
	Stream n	H = host, E = equipment
	Function m (odd)	reply option
	Sn, Fm name of function (XXX)	S, H→E , reply(required)
	(prime)	M H←E [reply] (optional)
		H→E "blank" (never)
Description:	A description of the action generated by the function.	
Structure:	Detailed structure showing lists and defined items. Lists are denoted by a capital L followed by the length separated by a comma. The individual elements in the list are numbered on separate lines. Nested lists are indented to emphasize the structure. The detailed form of the items is given in the define section at the beginning of the transaction. The symbols "<" and ">" are used to enclose each item in the structure definition and implies that there is an item header.	
Exceptions:	Special cases in the structure that have a different meaning.	
Sn,Fm+1 Name of function (secondary)		(same structure as above except never with reply)

7.4 Stream 0 and Function 0 - Stream 0 is always defined as not used since a 0 is the most likely error. No functions are defined in stream 0.

7.4.1 Function 0 exists in all streams and has the same special meaning in each stream. A function 0 message closes a transaction, so that the originator will not have

to wait for a transaction timeout to proceed. Function 0 is sent in lieu of the expected secondary message when the interpreter cannot, because of a transmission error or some other reason, respond with the expected reply. It is not a requirement that the interpreter send function 0 to close transaction.



7.5 Stream Equipment Status – This stream provides a means for exchanging information about the status of the equipment, including its current mode, depletion of

various consumable items, and the status of transfer operations.

S1,F0 Abort Transaction (SIF0)

S,H<—>E

Description: Used in lieu of an expected reply to abort a transaction. Function 0 is defined in every stream and has the same meaning in every stream.

Structure: Header only

S1,F1 Are You There Request (R)

S,H<—>E,reply

Description: Establishes if the equipment is on line. A function 0 response to this function means the communication is inoperative. In the equipment, a function 0 is equivalent to a timeout on the receive timer after issuing S1,F1 to the host.

Structure: Header only

S1,F2 On Line Data (D)

S,H<—>E

Description: Data signifying that the equipment is alive.

Structure: L,2

1. <MDLN>
2. <SOFTREV>

Exceptions: The host sends a zero length list to the equipment.

S1,F3 Selected Equipment Status Request (SSR)

S,H—>E,reply

Description: A request to the equipment to report selected values of its status.

Structure: The following structure is approved for all item formats and should be used by all new implementations:

L,n

1. <SVID1>

n. <SVIDn>

The following structure is included for compatibility with previous implementations and may only be used for items of format 3() and 5():

<SVID1, . . . , SVIDn>

Exceptions: A zero length item means report all SVID's.



S1,F4 Selected Equipment Status Data (SSD)

M,H<—E

Description: The equipment reports the value of each SVID requested in the order requested. The host remembers the names of values requested.

Structure: L,n

1. <SV1>

n. <SVn>

Exceptions: If n = 0, no response can be made. A zero length returned for SVi means that SVIDi does not exist.

S1,F5 Formatted Status Request (FSR)

S,H—>E,reply

Description: A request for the equipment to report the status according to a predefined fixed format.

Structure: <SFCD>

S1,F6 Formatted Status Data (FSD)

M,H<—E

Description: The equipment reports the value of status variables according to the SFCD.

Structure: Depends upon the structure specified by the status form.

Exceptions: A zero length item means that no report can be made.

S1,F7 Fixed Form Request (FFR)

S,H—>E,reply

Description: A request for the form used in S1,F6

Structure: <SFCD>

S1,F8 Fixed Form Data (FFD)

M,H<—E

Description: The form is returned with the name of each value and the corresponding data format item having a zero length as a two element list in the place of each single item to be returned in S1,F6.

Format: Depends upon the form being specified.

Exceptions: A zero length item means the form is unavailable.

S1,F9 Material Transfer Status Request (TSR)

S,H—>E,reply

Description: A request to report the status of all material ports to the host.

Structure: Header only

S1,F10 Material Transfer Status Data (TSD)

M,H<—E

Description: The equipment reports to the host the transfer status of all material ports.

Structure: L,2

1. <TSIP1, . . . , TSIPn>

2. <TSOP1, . . . , TSOPn>

Exceptions: A zero length item means there are no such ports. A zero length list means there are no ports.



S1,F11 Status Variable Namelist Request

S,H—>E,reply

Description: A request to the equipment to identify certain status variables.

Structure: L,n

1. <SVID1>

n. <SVIDn>

Exception: A zero length means report all SVIDs.

S1,F12 Status Variable Name List Reply

M,H<—E

Description: The equipment reports to the host the name and units of the requested SV's.

Structure: L,n

1. L,3

1. <SVID1>

2. <SVNAME1>

3. <UNITS1>

2. L,3

n. L,3

1. <SVIDn>

2. <SVNAMEn>

7.6 Stream 2 Equipment Control and Diagnostics – Messages which deal with control of the equipment from the host. This includes all remote operations and equipment self-diagnostics and calibration but specifi-

cally excludes the control operations which are associated with material transfer (see stream 4), loading of executive and boot programs (stream 8), and all file and operating system calls (stream 10, 13).

S2,F0 Abort Transaction (S2F0)

S,H<—>E

Description: Save form as S1F0

S2,F1 Service Program Load Inquire (SPI)

S,H<—>E, reply

Description: Either the host or equipment wants to send the specified program.

Structure: L,2

1. <SPID>

2. <LENGTH>

S2,F2 Service Program Load Grant (SPG)

S,H<—>E

Description: Provides permission to load.

Structure: <GRANT>



Documents

Document Sharing Hub



S2,F3 Service Program Send (SPS)

M,H<—>E,reply

Description: The data associated with the S2,F1 inquire is sent. If S2,F3 is multi-block, it must be preceded by the S2,F1/S2,F2 Inquire/Grant transaction.

Structure: <SPD>

S2,F4 Service Program Send Acknowledge (SPA)

S,H<—>E

Description: Acknowledge or error.

Structure: <SPAACK>

S2,F5 Service Program Load Request (SPR)

S,H<—>E,reply

Description: A service program is requested.

Structure: <SPID>

S2,F6 Service Program Load Data (SPD)

M,H<—>E

Description: A service program is sent.

Structure: <SPD>

Exceptions: A zero length item means that the requested program cannot be returned.

S2,F7 Service Program Run Send (CSS)

S,H—>E,reply

Description: Start the requested program.

Structure: <SPID>

S2,F8 Service Program Run Acknowledge (CSA)

S,H<—E

Description: Acknowledge or error.

Structure: <CSAACK>

S2,F9 Service Program Results Request (SRR)

S,H—>,reply

Description: Ask for results of service program.

Structure: <SPID>

S2,F10 Service Program Results Data (SRD)

M,H<—E

Description: Get the results back.

Structure: <SPR>

Exceptions: A zero length items means SPR does not exist.

S2,F11 Service Program Directory Request (SDR)

S,H<—>E,reply

Description: There may be more than one service program.

Structure: Header only.

S2,F12 Service Program Directory Data (SDD)

S,H<—>E

Description: A list of service program names.

Structure: L,n



1. <SPID1>

.

.

n. <SPIDn>

Exceptions: If $n = 0$, there are no service programs.

S2,F13 Equipment Constant Request (ECR)

S,H→E,reply

Description: Constants such as for calibration, servo gain, alarm limits, data collection mode and other values that are changed infrequently can be obtained using this message.

Structure: The following structure is approved for all item formats and should be used by all new implementations:

L,n

1. <ECID1>

.

.

n. <ECIDn>

The following structure is included for compatibility with previous implementations and may only be used for items of format 3() and 5(:

<ECID1, . . . , ECIDn>

Exceptions: If $n = 0$, then no response exists. A zero length ECVi means that ECIDi does not exist.

S2,F14 Equipment Constant Data (ECD)

M,H←E

Description: Data Response to S2F13 in the order requested.

Structure: L,n

1. <ECV1>

2. <ECV2>

.

n. <ECVn>

Exceptions: If $n = 0$, then no response exists. A zero length ECVi means that ECIDi does not exist.

S2,F15 New Equipment Constant Send (ECS)

S,H→E,reply

Description: Change one or more equipment constants.