



SEMI E5-0997

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

CONTENTS

1 Introduction

- 1.1 Intent
- 1.2 Overview
- 1.3 Application
- 1.4 Applicable Documents

2 Selected Definitions

3 The Message Transfer Protocol

- 3.1 Intent
- 3.2 Messages
- 3.2 Blocking Requirements
- 3.2 Message Header
- 3.2 The Transaction Timeout
- 3.2 Multiple Open Transaction

4 Streams and Functions

- 4.1 Streams
- 4.2 Functions
- 4.3 Stream and Function Allocation

5 Transaction and Conversation Protocols

- 5.1 Intent
- 5.2 Transaction Definition
- 5.3 Transaction Level Requirements
- 5.4 Conversation Protocols

6 Data Structures

- 6.1 Intent
- 6.2 Item
- 6.3 List
- 6.4 Example Data Structures
- 6.5 Data Item Dictionary
- 6.6 Variable Item Dictionary
- 6.7 Object Dictionary

7 Message Detail

- 7.1 Intent
- 7.4 Stream 0 and Function 0
- 7.5 Stream 1 Equipment Status
 - S1,F0 Abort Transaction (S1F0)
 - S1,F1 Are You There Request (R)
 - S1,F2 On Line Data (D)
 - S1,F3 Selected Equipment Status Request (SSR)
 - S1,F4 Selected Equipment Status Data (SSD)
 - S1,F5 Formatted Status Request (FSR)
 - S1,F6 Formatted Status Data (FSD)
 - S1,F7 Fixed Form Request (FFR)
 - S1,F8 Fixed Form Data (FFD)

- S1,F9 Material Transfer Status Request (TSR)
- S1,F10 Material Transfer Status Data (TSD)
- S1,F11 Status Variable NameList Request (SVNR)
- S1,F12 Status Variable NameList Reply (SVNRR)
- S1,F13 Establish Communications Request (CR)
- S1,F14 Establish Communications Request Acknowledge (CRA)
- S1,F15 Request OFF-LINE (ROFL)
- S1,F16 OFF-LINE Acknowledge (OFLA)
- S1,F17 Request ON-LINE (RONL)
- S1,F18 ON-LINE Acknowledge (ONLA)

Macro Level Messages

- S1,F19 Get Attribute (GA)
- S1,F20 Attribute Data (AD)

- 7.6 Stream 2 Equipment Control and Diagnostics
 - S2,F0 Abort Transaction (S2F0)
 - S2,F1 Service Program Load Inquire (SPI)
 - S2,F2 Service Program Load Grant (SPG)
 - S2,F3 Service Program Send (SPS)
 - S2,F4 Service Program Send Acknowledge (SPA)
 - S2,F5 Service Program Load Request (SPR)
 - S2,F6 Service Program Load Data (SPD)
 - S2,F7 Service Program Run Send (CSS)
 - S2,F8 Service Program Run Acknowledge (CSA)
 - S2,F9 Service Program Results Request (SRR)
 - S2,F10 Service Program Results Data (SRD)
 - S2,F11 Service Program Directory Request (SDR)
 - S2,F12 Service Program Directory Data (SDD)
 - S2,F13 Equipment Constant Request (ECR)
 - S2,F14 Equipment Constant Data (ECD)
 - S2,F15 New Equipment Constant Send (ECS)
 - S2,F16 New Equipment Constant Acknowledge (ECA)
 - S2,F17 Date and Time Request (DTR)
 - S2,F18 Date and Time Data (DTD)
 - S2,F19 Reset/Initialize Send (RIS)
 - S2,F20 Reset Acknowledge (RIA)
 - S2,F21 Remote Command Send (RCS)
 - S2,F22 Remote Command Acknowledge (RCA)
 - S2,F23 Trace Initialize Send (TIS)
 - S2,F24 Trace Initialize Acknowledge (TIA)
 - S2,F25 Loopback Diagnostic Request (LDR)
 - S2,F26 Loopback Diagnostic Data (LDD)
 - S2,F27 Initiate Processing Request (IPR)
 - S2,F28 Initiate Processing Acknowledge (IPA)
 - S2,F29 Equipment Constant Namelist Request (ECNR)



S2,F30 Equipment Constant Namelist (ECN)
S2,F31 Date and Time Set Request (DTS)
S2,F32 Date and Time Set Acknowledge (DTA)
S2,F33 Define Report (DR)
S2,F34 Define Report Acknowledge (DRA)
S2,F35 Link Report/Event (LER)
S2,F36 Link Report/Event Acknowledge (LERA)
S2,F37 Enable/Disable Event Report (EDER)
S2,F38 Enable/Disable Event Report Acknowledge (EERA)
S2,F39 Multi-block Inquire (MBI)
S2,F40 Multi-block Grant (MBG)
S2,F41 Host Command Send (HCS)
S2,F42 Host Command Acknowledge (HCA)
S2,F43 Reset Spooling Streams and Functions (RSSF)
S2,F44 Reset Spooling Acknowledge (RSA)
S2,F45 Define Variable Limit Attributes (DVLA)
S2,F46 Variable Limit Attribute Acknowledge (VLAA)
S2,F47 Variable Limit Attribute Request (VLAR)
S2,F48 Variable Limit Attribute Send (VLAS)
S2,F49 Enhanced Remote Command
S2,F50 Enhanced Remote Command Acknowledge

7.7 Stream 3 Material Status

S3,F0 Abort Transaction (S3F0)
S3,F1 Material Status Request (MSR)
S3,F2 Material Status Data (MSD)
S3,F3 Time to Completion Request (TCR)
S3,F4 Time to Completion Data (TCD)
S3,F5 Material Found Send (MFS)
S3,F6 Material Found Acknowledge (MFA)
S3,F7 Material Lost Send (MLS)
S3,F8 Material Lost Acknowledge (MLA)
S3,F9 Material ID Equate Send (IES)
S3,F10 Material ID Equate Acknowledge (IEA)
S3,F11 Material ID Request (MIDR)
S3,F12 Material ID Request Acknowledge (MIRA)
S3,F13 Material ID Send
S3,F14 Material ID Acknowledge

7.8 Stream 4 Material Control

S4,F0 Abort Transaction (S4F0)
S4,F1 Ready to Send Material (RSN)
S4,F2 Ready to Send Acknowledge (RSA)
S4,F3 Send Material (SMN)
S4,F4 Not Used
S4,F5 Handshake Complete (HCN)
S4,F6 Not Used
S4,F7 Not Ready to Send (ABN)
S4,F8 Not Used
S4,F9 Stuck in Sender (SSN)
S4,F10 Not Used
S4,F11 Stuck in Receiver (SRN)
S4,F12 Not Used

S4,F13 Send Incomplete Timeout (SIN)
S4,F14 Not Used
S4,F15 Material Received (MRN)
S4,F16 Not Used
S4,F17 Request to Receive (RTR)
S4,F18 Request to Receive Acknowledge (RRA)

Macro Level Messages

S4,F19 Transfer Job Create (TJ)
S4,F20 Transfer Job Acknowledge (TJA)
S4,F21 Transfer Job Command (TJC)
S4,F22 Transfer Command Acknowledge (TCA)
S4,F23 Transfer Job Alert (TJA)
S4,F24 Transfer Alert Acknowledge (TLA)
S4,F25 Multi-Block Inquire (MB14)
S4,F26 Multi-Block Grant (MBG4)

Micro Level Messages

S4,F27 Handoff Ready (HR)
S4,F28 Not Used
S4,F29 Handoff Command (HC)
S4,F30 Not Used
S4,F31 Handoff Command Complete (HCC)
S4,F32 Not Used
S4,F33 Handoff Verify (HV)
S4,F34 Not Used
S4,F35 Handoff Cancel Ready (HCR)
S4,F36 Not Used
S4,F37 Handoff Cancel Ready Acknowledge (HCA)
S4,F38 Not Used
S4,F39 Handoff Halt (HH)
S4,F40 Not Used
S4,F41 Handoff Halt Acknowledge (HHA)
S4,F42 Not Used

7.9 Stream 5 Exception Handling

S5,F0 Abort Transaction (S5F0)
S5,F1 Alarm Report Send (ARS)
S5,F2 Alarm Report Acknowledge (ARA)
S5,F3 Enable/Disable Alarm Send (EAS)
S5,F4 Enable/Disable Alarm Acknowledge (EAA)
S5,F5 List Alarms Request (LAR)
S5,F6 List Alarm Data (LAD)
S5,F7 List Enabled Alarm Request
S5,F8 List Enabled Alarm Data
S5,F9 Exception Post Notify (EXPN)
S5,F10 Exception Post Confirm (EXPC)
S5,F11 Exception Clear Notify (EXCN)
S5,F12 Exception Clear Confirm (EXCC)
S5,F13 Exception Recover Request (EXRR)
S5,F14 Exception Recover Acknowledge (EXRA)
S5,F15 Exception Recovery Complete Notify (EXRCN)
S5,F16 Exception Recovery Complete Confirm (EXRCC)
S5,F17 Exception Recovery Abort Request (EXRAR)



S5,F18	Exception Recovery Abort Acknowledge (EXRAA)	S7,F16	Matrix Mode Select Acknowledge (MMA)
7.10 Stream 6	Data Collection	S7,F17	Delete Process Program Send (DPS)
S6,F0	Abort Transaction (S6F0)	S7,F18	Delete Process Program Acknowledge (DPA)
S6,F1	Trace Data Send (TDS)	S7,F19	Current EPPD Request (RER)
S6,F2	Trace Data Acknowledge (TDA)	S7,F20	Current EPPD Data (RED)
S6,F3	Discrete Variable Data Send (DVS)	S7,F21	Equipment Process Capabilities Request (PCR)
S6,F4	Discrete Variable Data Acknowledge (DVA)	S7,F22	Equipment Process Capabilities Data (PCD)
S6,F5	Multi-block Data Send Inquire (MBI)	S7,F23	Formatted Process Program Send (FPS)
S6,F6	Multi-block Grant (MBG)	S7,F24	Formatted Process Program Acknowledge (FPA)
S6,F7	Data Transfer Request (DDR)	S7,F25	Formatted Process Program Request (FPR)
S6,F8	Data Transfer Data (DDD)	S7,F26	Formatted Process Program Data (FPD)
S6,F9	Formatted Variable Send (FVS)	S7,F27	Process Program Verification Send (PVS)
S6,F10	Formatted Variable Acknowledge (FVA)	S7,F28	Process Program Verification Acknowledge (PVA)
S6,F11	Event Report Send (ERS)	S7,F29	Process program Verification Inquire (PVI)
S6,F12	Event Report Acknowledge (ERA)	S7,F30	Process Program Verification Grant (PVG)
S6,F13	Annotated Event Report Send (AERS)	S7,F31	Verification Request Send (VRS)
S6,F14	Annotated Event Report Acknowledge (AERA)	S7,F32	Verification Request Acknowledge (VRA)
S6,F15	Event Report Request (ERR)	S7,F33	Process Program Available Request (PAR)
S6,F16	Event Report Data (ERD)	S7,F34	Process Program Availability Data (PAD)
S6,F17	Annotated Event Report Request (AERR)	S7,F35	Process Program for MID Request (PPMR)
S6,F18	Annotated Event Report Data (AERD)	S7,F36	Process Program for MID Data (PPMD)
S6,F19	Individual Report Request (IRR)	7.12 Stream 8	Control Program Transfer
S6,F20	Individual Report Data (IRD)	S8,F0	Abort Transaction (S8F0)
S6,F21	Annotated Individual Report (AIR)	S8,F1	Boot Program Request (BPR)
S6,F22	Annotated Individual Report Data (AIRD)	S8,F2	Boot Program Data (BPD)
S6,F23	Request Spooled Data (RSD)	S8,F3	Executive Program Request (EPR)
S6,F24	Request Spooled Data Acknowledgement Send (RSDAS)	S8,F4	Executive Program Data (EPD)
S6,F25	Notification Report Send	7.13 Stream 9	System Errors
S6,F26	Notification Report Send Acknowledge	S9,F0	Abort Transaction (S9F0)
S6,F27	Trace Report Send (TRS)	S9,F1	Unrecognized Device ID (UDN)
S6,F28	Trace Report Send Acknowledge (TRSA)	S9,F2	Not Used
S6,F29	Trace Report Request (TRR)	S9,F3	Unrecognized Stream Type (USN)
S6,F30	Trace Report Data (TRD)	S9,F4	Not Used
7.11 Stream 7	Process Program Management	S9,F5	Unrecognized Function Type (UFN)
S7,F0	Abort Transaction (S7F0)	S9,F6	Not Used
S7,F1	Process Program Load Inquire (PPI)	S9,F7	Illegal Data (IDN)
S7,F2	Process Program Load Grant (PPG)	S9,F8	Not Used
S7,F3	Process Program Send (PPS)	S9,F9	Transaction Timer Timeout (TTN)
S7,F4	Process Program Acknowledge (PPA)	S9,F10	Not Used
S7,F5	Process Program Request (PPR)	S9,F11	Data Too Long (DLN)
S7,F6	Process Program Data (PPD)	S9,F12	Not Used
S7,F7	Process Program ID Request (PIR)	S9,F13	Conversation Timeout (CTN)
S7,F8	Process Program ID Data (PID)	S9,F14	Not Used
S7,F9	M/P M Request (MMR)		
S7,F10	M/P M Data (MMD)		
S7,F11	M/P M Update Send (UMS)		
S7,F12	M/P M Update Acknowledge (UMA)		
S7,F13	Delete M/P M Entry Send (DES)		
S7,F14	Delete M/P M Entry Acknowledge (DEA)		
S7,F15	Matrix Mode Select Send (MMS)		



7.14 Stream 10 Terminal Services

- S10,F0 Abort Transaction (S10F0)
- S10,F1 Terminal Request (TRN)
- S10,F2 Terminal Request Acknowledge (TRA)
- S10,F3 Terminal Display, Single (VTN)
- S10,F4 Terminal Display, Single Acknowledge (VTA)
- S10,F5 Terminal Display, Multi-block (VTN)
- S10,F6 Terminal Display, Multi-block Acknowledge (VMA)
- S10,F7 Multi-block Not Allowed (MNN)
- S10,F8 Not Used
- S10,F9 Broadcast (BCN)
- S10,F10 Broadcast Acknowledge (BCA)

7.15 Stream 11 Host File Services (Deleted)

7.16 Stream 12 Wafer Mapping

- S12,F0 Abort Transaction (S12F0)
- S12,F1 Map Set-Up Data Send (MSDS)
- S12,F2 Map Set-up Data Acknowledge (MSDA)
- S12,F3 Map Set-up Data Request (MSDR)
- S12,F4 Map Set-up Data (MSD)
- S12,F5 Map Transmit Inquire (MAPTI)
- S12,F6 Map Transmit Grant (MAPTG)
- S12,F7 Map Data Send Type 1 (MDS1)
- S12,F8 Map Data Acknowledge Type 1 (MDA1)
- S12,F9 Map Data Send Type 2 (MDS2)
- S12,F10 Map Data Acknowledge Type 2 (MDA2)
- S12,F11 Map Data Send Type 3 (MDS3)
- S12,F12 Map Data Acknowledge 3 (MDA3)
- S12,F13 Map Data Request Type 1 (MDR1)
- S12,F14 Map Data Type 1 (MD1)
- S12,F15 Map Data Request Type 2 (MDR2)
- S12,F16 Map Data Type 2 (MD2)
- S12,F17 Map Data Request Type 3 (MDR3)
- S12,F18 Map Data Type 3 (MD3)
- S12,F19 Map Error Report Send (MERS)
- S12,F20 Not Used

7.17 Stream 13 Data Set Transfers

- S13,F0 Abort Transaction (S10F0)
- S13,F1 Send Data Set Send (DSSS)
- S13,F2 Send Data Set Acknowledge (DSSA)
- S13,F3 Open Data Set Request (DSOR)
- S13,F4 Open Data Set Data (DSOD)
- S13,F5 Read Data Set Request (DSRR)
- S13,F6 Read Data Set Data (DSRD)
- S13,F7 Close Data Set Send (DSCS)
- S13,F8 Close Data Set Acknowledge (DSCA)
- S13,F9 Reset Data Set Send (DSRS)
- S13,F10 Reset Data Set Acknowledge (DSRA)
- S13,F11 Data Set Object Multi-Block Inquire (DSOMGI)
- S13,F12 Data Set Object Multi-Block Grant (DSOMBG)
- S13,F13 Table Data Send (TDS)
- S13,F14 Table Data Acknowledge (TDA)
- S13,F15 Table Data Request (TDR)

S13,F16 Table Data (TD)

7.18 Stream 14 Object Services

- S14,F0 Abort Transaction (S14F0)
- S14,F1 GetAttr Request (GAR)
- S14,F2 GetAttr Date (GAD)
- S14,F3 SetAttr Request (SAR)
- S14,F4 SetAttr Data (SAD)
- S14,F5 GetType Request (GTR)
- S14,F6 GetType Data (GTD)
- S14,F7 GetAttrName Request (GANR)
- S14,F8 GetAttrName Data (GAND)
- S14,F9 Create Object Request (COR)
- S14,F10 Create Object Acknowledge (CAO)
- S14,F11 Delete Object Request
- S14,F12 Delete Object Acknowledge (DOA)
- S14,F13 Object Attach Request (OAR)
- S14,F14 Object Attach Acknowledge (OAA)
- S14,F15 Attached Object Action Request (AOAR)
- S14,F16 Attached Object Action Acknowledge (AOAA)
- S14,F17 Supervised Object Action Request (SOAR)
- S14,F18 Supervised Object Action Acknowledge (SOAA)

7.19 Stream 15 Recipe Management

- S15,F0 Abort Transaction (S15F0)
- S15,F1 Recipe Management Multi-block Inquire
- S15,F2 Recipe Management Multi-block Grant
- S15,F3 Recipe Namespace Action Request
- S15,F4 Recipe Namespace Action Acknowledge
- S15,F5 Recipe Namespace Rename Request
- S15,F6 Recipe Namespace Rename Acknowledge
- S15,F7 Recipe Space Request
- S15,F8 Recipe Space Data
- S15,F9 Recipe Status Request
- S15,F10 Recipe Status Data
- S15,F11 Recipe Version Request
- S15,F12 Recipe Version Data
- S15,F13 Recipe Create Request
- S15,F14 Recipe Create Acknowledge
- S15,F15 Recipe Store Request
- S15,F16 Recipe Store Acknowledge
- S15,F17 Recipe Retrieve Request
- S15,F18 Recipe Retrieve Data
- S15,F19 Recipe Rename Request
- S15,F20 Recipe Rename Acknowledge
- S15,F21 Recipe Action Request
- S15,F22 Recipe Action Acknowledge
- S15,F23 Recipe Descriptor Request
- S15,F24 Recipe Descriptor Data
- S15,F25 Recipe Parameter Update Request
- S15,F26 Recipe Parameter Update Acknowledge
- S15,F27 Recipe Download Request
- S15,F28 Recipe Download Acknowledge
- S15,F29 Recipe Verify Request



S15,F30 Recipe Verify Acknowledge
S15,F31 Recipe Unload Request
S15,F32 Recipe Unload Data
S15,F33 Recipe Select Request
S15,F34 Recipe Select Acknowledge
S15,F35 Recipe Delete Request
S15,F36 Recipe Delete Acknowledge
S15,F37 DRNS Segment Approve Action Request
S15,F38 DRNS Segment Approve Action Acknowledge
S15,F39 DRNS Recorder Segment Request
S15,F40 DRNS Recorder Segment Acknowledge
S15,F41 DRNS Recorder Modify Request
S15,F42 DRNS Recorder Modify Acknowledge
S15,F43 DRNS Get Change Request
S15,F44 DRNS Get Change Request Data
S15,F45 DRNS Manager Segment Change Approval Request
S15,F46 DRNS Manager Segment Approval Acknowledge
S15,F47 DRNS Manager Rebuild Request
S15,F48 DRNS Manager Rebuild Acknowledge

7.20 Stream 16 Processing Management

S16,F1 Multi-block Process Job Data Inquire (PRJI)
S16,F2 Multi-block Process Job Data Grant (PRJG)
S16,F3 Process Job Create Request (PRJCR)
S16,F4 Process Job Create Acknowledge (PRJCA)
S16,F5 Process Job Command Request (PRJCMDR)
S16,F6 Process Job Command Acknowledge (PRJCMDA)
S16,F7 Process Job Alert Notify (PRJA)
S16,F8 Process Job Alert Confirm (PRJAC)
S16,F9 Process Job Event Notify (PRJE)
S16,F10 Process Job Event Confirm (PRJEC)

7.21 Stream 17 Equipment Control and Diagnostics

S17,F1 Data Report Create Request (DRC)
S17,F2 Data Report Create Acknowledge (DRCA)
S17,F3 Data Report Delete Request (DRD)
S17,F4 Data Report Delete Acknowledge (DRDA)
S17,F5 Trace Create Request (TRC)
S17,F6 Trace Create Acknowledge (TRCA)
S17,F7 Trace Delete Request (TRD)
S17,F8 Trace Delete Acknowledge (TRDA)
S17,F9 Collection Event Link Request (CELR)
S17,F10 Collection Event Link Acknowledge (CELA)
S17,F11 Collection Event Unlink Request (CEUR)
S17,F12 Collection Event Unlink Acknowledge (CEUA)
S17,F13 Trace Reset Request (TRR)

S17,F14 Trace Report Reset Acknowledge (TRRA)

8 Message Documentation

- 8.1 Intent
- 8.2 Standard Form SECS-II Document

9 Units of Measure

- 9.1 Intent
- 9.2 Units Symbol
- 9.3 Compliance
- 9.4 SECS-II Units of Measure Identifiers

Application Notes

- A1. The General Node Transaction Protocol
- A2. Some Suggested Message Usage
- A3. Notes on SECS-II Data Transfers
- A4. Process Programs
- A5. Suggested Baseline SECS Equipment Implementation



NOTES

SUPERS EDED

This page intentionally left blank.



SEMI E5-0997

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 equipment implementation.

1.4 *Applicable Documents*

1.4.1 *ANSI*¹

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

1.4.2 *IEEE*²

754 — Standard for Binary Floating Point Arithmetic

¹ ANSI, 1430 Broadway, New York, NY 10018

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



1.4.3 SEMI Specifications³

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

SEMI E6 — Facilities Interface Specifications Guide-line and Format

1.4.4 The Japan Electronic Industry Development Association (JEIDA) has requested that the SECS-II standard incorporate support for the JIS-8 codes for data exchange. This code would allow support for katakana characters in Japanese implementations of SECS-II.

JIS 8-bit Coded Character Set (JIS-6226) for information Interchange, Japanese Industrial Standards.

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 Section 3.3).

conversation — A sequence of related messages (see Section 5.4).

conversation timeout — An indication that a conversation has not completed properly (see Section 5.4.1).

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

equipment — The intelligent system which communicates with a host.

function — A specific message for a specific activity within a stream (see Section 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 Section 3.2).

item — A data element within a message (see Section 6.2).

item format — A code used to identify the data type of an item (see Section 6.2).

list — A group of items (see Section 6.3).

message — A complete unit of communication (see Section 3.2).

message header — Information about the message passed by the message transfer protocol (see Section 3.4).

multi-block message — A message sent in more than one block by the message transfer protocol (see Section 3.3.2).

originator — The creator of a primary message (see Section 3.2).

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

primary message — An odd numbered message. Also, the first message of a transaction (see Sections 3.2 and 4.2).

reply — The particular secondary message corresponding to a primary message (see Sections 3.2 and 4.2).

secondary message — An even-numbered message. Also the second message of a transaction (see Sections 3.2 and 4.2).

single-block message — A message sent in one block by the message transfer protocol (see Section 3.3.1).

stream — A category of messages (see Section 4.1).

transaction — A primary message and its associated secondary message, if any (see Section 5.2).

transaction timeout — An indication from the message transfer protocol that a transaction has not completed properly (see Section 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 SEMI, 805 E. Middlefield Road, Mountain View, CA 94043



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. Certain older implementations may impose application-specific requirements on block sizes for certain incoming messages. Beginning with the 1988 revision of the standard, new applications may not impose application-specific requirements on incoming block sizes. Applications implemented before 1988 may impose such requirements.

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 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 timeout (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.

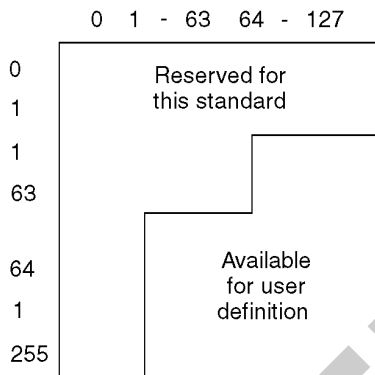


Figure 1
Stream and Function Allocation

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 S1F1 with S1F2 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 S9F9 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 committed resources. Upon detection of a conversation timeout at the equipment, S9F13 should be sent to the host.

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

1. A primary message with no reply is the simplest conversation. This message must be a single-block SECS-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 unformatted data sets between equipment and host. This conversation is described in detail in Stream 13. (See Section 7.17.)
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 responds 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 S9F13 error message. Note that under the definition of S9F13 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 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.

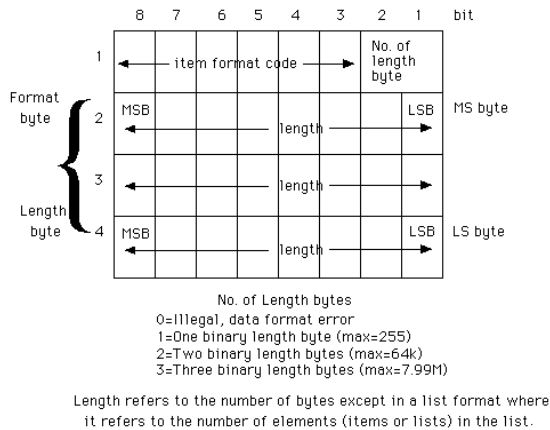


Figure 2
Item and List Header

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, fifteen 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 which 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, which is defined in the detailed message definitions.

Format Code	Meaning
Binary	The data after the heading
Bit 876543 Octal	has the following form
000000 00	LIST (length in elements)
001000 10	Binary
001001 11	Boolean
010000 20	ASCII ¹
010001 21	JIS-8
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 754. The byte containing the sign bit is sent first.

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



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  Item, binary, 1 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 signed integer form.

```
01101001 Item, 2-byte integers
00000110 6 bytes total (6/2=3 integers)

xxxxxxx MSByte number x
xxxxxxx 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)
ffffffff  Floating point number
ffffffff  in IEEE 754
ffffffff
```



- e. A message is sent from device 66 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)
00101110	Device Code=66
00000101	Stream 5, W=0
00000001	Function 1
10000000	E=1
00000001	Block 1
00000000	
00000000	System bytes=0
00000000	
00000000	
00000001	List
00000011	3 Elements
00100001	Binary Item next byte length
00000001	1 byte long
00000100	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, 1-byte length (not shown), 10 bytes of header, 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 — This section defines the data items used in the standard SECS-II messages described in Section 7, Message Detail.

Name: A unique mnemonic name for this data item. This name is used in message definitions.

Format: The allowable item format codes which can be used for this standard data item. Item format codes are shown in octal, as described in Table 1, Item Format Codes. The notation "3()" indicates any of the signed integer formats (30, 31, 32, 34). The notation "4()" indicates any of the floating point formats (40, 44). The notation "5()" indicates any of the unsigned integer formats (50, 51, 52, 54). The notation "0" indicates that a list with user-defined structure may be used. Where more than one format is shown, a given implementation can use any of the formats specified.

Description: A description of the data item, with the meanings of specific values.

Where Used: The standard messages in which this data item appears.



ABS

Format: 10

Any binary string

Where Used: S2F25, F26

ACDS

Format: 32, 52

After Command Codes

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

Where Used: S7F22

ACKA

Format: 11

Indicates success of a request:

TRUE is successful else FALSE

Where Used: S5F14, F15, F18, S16F2, F4, F6, S17F4, F8, F14

ACKC3

Format: 10

Acknowledge code, 1 byte

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

Where Used; S3F6, F8, F10

ACKC5

Format: 10

Acknowledge code, 1 byte

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

Where Used; S5F2, F4

ACKC6

Format: 10

Acknowledge code, 1 byte

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

Where Used; S6F2, F4, F10, F12, F14



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, F12, F14, F16, F18, F24, F32

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

ACKC10

Format: 10

Acknowledge Code, 1 byte

- 0 = Accepted for display
- 1 = Message will not be displayed
- 2 = Terminal not available
- 3-63 reserved

Where Used: S10F2, F4, F6, F10



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: Illegal 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
- 9 = ERROR: Handle in Use
- >10 = ERROR: Pending Transaction
- 11-127 reserved

Where Used: S13F2, F4, F6, F8

AGENT

Format: 20

Where Used: S15F11, 12, 21, 22, 25

ALCD

Format: 10

Alarm code byte

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

Where Used: S5F1, F6

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, F3, F5, F6

ALTX

Format: 20

Alarm text limited to 40 characters

Where Used: S5F1, F6

ATTRDATA

Format: 0,10,20,3(),4(),5()

Contains a specific attribute value for a specific object.

Where Used: S1F20, S13F14, F16, S14F1, F2, F3, F4, F9, F10, F11, F12, F13, F14, F15, F16, F17, F18

ATTRID

Format: 20,5()

Identifier for an attribute for a specific type of object.

Where Used: S1F19, S13F14, F16, S14F1, F2, F3, F4, F8, F9, F10, F11, F12, F13, F14, F15, F16, F17, F18

ATTRRELN

Format: 51

The relationship that a specified qualifying value has to the value of an attribute of an object instance (the value of interest):

- 0 = The qualifying value is equal to the value of interest,
- 1 = The qualifying value is not equal to the value of interest,
- 2 = The qualifying value is less than the value of interest,
- 3 = The qualifying value is less than or equal to the value of interest,
- 4 = The qualifying value is greater than the value of interest,
- 5 = The qualifying value is greater than or equal to the value of interest,
- 6 = The qualifying value is present (contained in the set of) the value of interest,
- 7 = The qualifying value is absent (not contained in the set of) the value of interest,
- >7 = reserved.

Where Used: S14F1

BCDS

Format: 32, 52

Before Command Codes

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

Where Used: S7F22



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: S12F3, F4

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, F9, F11, F14, F16, F18

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

Format: 10

Boot program Data

Where Used: S8F2

BYTMAX

Format: 3(),5()

Byte Maximum

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

Where Used: S7F22



CCODE

Format: 32, 52

Command Code

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

Where Used: S7F22, F23, F26, F31

CEED

Format: 11

Collection event or trace enable/disable code, 1 byte

FALSE=Disable

TRUE=Enable

| Where Used: S2F37, S17F5

CEID

Format: 20, 3(), 5()

Collected event ID

| Where Used: S2F35, F37, S6F3, F8, F9, F11, F13, F15, F17, S17F5, F9, F10, F11, F12

CEPACK

Format: 0, 51

Command Enhanced Parameter Acknowledge. If a specific value of CPNAME is defined to have a CEPVAL that is a LIST, then CEPACK shall have the same structure as the corresponding list format of CEPVAL as used in S2F63. Otherwise CEPACK will a 1 byte integer. Enumerated:

0 = no error

1 = Parameter name (CPNAME) does not exist

2 = Illegal value specified for CEPVAL

3 = Illegal format specified for CEPVAL

4 = Parameter name (CPNAME) not valid as used

5-63 reserved

| Where Used: S2F50

CEPVAL

Format: 0, 10, 11, 20, 21, 3(), 4(), 5()

Command Enhanced Parameter Value. A specific application of CEPVAL shall always be identified with a specific value of CPNAME. A CEPVAL has the following forms: a single (non-list) value (eg. CPVAL), a list of single items of identical format and type, or a list of items of the form.

L, 2

1. CPNAME

2. CEPVAL

| Where Used: S2F49



CKPNT

Format: 54

Checkpoint as defined by the sending system

Where Used: S13F3, F6

CMDA

Format: 31, 51

Command acknowledge code

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

Where Used: S2F22, F28

CMDMAX

Format: 3(),5()

Command Maximum

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 = 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, F4

COLHDR

Format: 20

Text description of contents of TBLELT. 1 - 20 characters

Where Used: S13F13, F16



COMMACK

Format: 10

Establish Communications Acknowledge Code, 1 byte

- 0 = Accepted
- 1 = Denied, Try Again
- 2-63 reserved

Where Used: S1F14

CPACK

Format: 10

Command Parameter Acknowledge Code, 1 byte

- 1 = Parameter Name (CPNAME) does not exist
- 2 = Illegal Value specified for CPVAL
- 3 = Illegal Format specified for CPVAL
- >3 = Other equipment-specific error
- 4-63 reserved

Where Used: S2F42

CPNAME

Format: 20, 3(), 5()

Command Parameter Name

Where Used: S2F41, F42, F49, F50, S4F21, F29, S16F5

CPVAL

Format: 10, 11, 20, 21, 3(), 5()

Command Parameter Value

Where Used: S2F41, F49, S4F21, F29, S16F5

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: 20, 3(), 5()

Data ID

Where Used: S2F33, F35, F39, F45, F49, S4F19, F25, S6F3, F5, F7, F8, F9, F11, F13, F16, F18, F27, S13F11, F13, F15, S15F27, F29, F33, F35, F37, F39, F41, F43, F45, F47, S16F1, F3, F5, S17F1, F5, F9



DATALENGTH

Format: 3(),5()

Total bytes to be sent

Where Used: S2F39, S4F25, S6F5, S13F11, S16F1

DATASRC

Format: 20

Object type for Data Source Objects.

Where Used: S14F1, F3, F6, F7, F8, S17F1

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: S12F19

DRACK

Format: 10

Define Report Acknowledge Code, 1 byte

- 0 = Accept
- 1 = Denied. Insufficient space
- 2 = Denied. Invalid format
- 3 = Denied. At least one RPTID already defined
- 4 = Denied, At least VID does not exist
- >4 = Other errors
- 5-63 reserved

Where Used: S2F34

DSID

Format: 20, 3(), 5()

Data set ID

Where Used: S6F3, F8, F9

DSNAME

Format: 20

The name of the Data Set

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

Where Used: S13F1, F2, F3, F4



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, F4

DVNAME

Format: 3(),20,5()

Data value name

Where Used: S6F3, F8

DVVAL

Format: 0, 10, 11, 20, 21, 3(), 4(), 5()

Data value

Where Used: S6F3, F8, F9

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: 10, 11, 20, 21, 3(), 4(), 5()

Equipment constant default value

Where Used: S2F30

ECID

Format: 3(), 20, 5()

Equipment Constant ID

Where Used: S2F13, F15, F29, F30



ECMAX

Format: 10, 11, 20, 21, 3(), 4(), 5()

Equipment constant maximum value

Where Used: S2F30

ECMIN

Format: 10, 11, 20, 21, 3(), 4(), 5()

Equipment constant minimum value

Where Used: S2F30

ECNAME

Format: 20

Equipment constant name

Where Used: S2F30

ECV

Format: 10, 11, 20, 21, 3(), 4(), 5()

Equipment Constant Value

Where Used: S2F14, F15

EDID

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

Expected data Identification

Three possible responses.

MEXP	EDID	EDID
S02F03, <SPID>	A[6]	
S03F13, <PTN>	B[1]	
S07F03, <PPID>	A[16], B[16]	

Where Used: S9F13

EMID

Format: 10, 20

Equivalent material ID (16 bytes maximum)

Where Used: S3F9

EPD

Format: 10

Executive program data

Where Used: S8F4



EQNAME

Format: 20

A unique ASCII equipment identifier assigned by the factory to the equipment. Limited to a maximum of 80 characters.

Where Used: S4F29

ERACK

Format: 10

Enable/Disable Event Report

Acknowledge Code, 1 byte

- 0 = Accepted
- 1 = Denied. At least one CEID does not exist
- <1 = Other Errors
- 2-63 reserved

Where Used: S2F38

SUPERSEDED



ERRCODE

Format: 5()

Code identifying an error

0	=	No error
1	=	Unknown object in Object Specifier
2	=	Unknown target object type
3	=	Unknown object instance
4	=	Unknown attribute name
5	=	Read-only attribute - access denied
6	=	Unknown object type
7	=	Invalid attribute value
8	=	Syntax error
9	=	Verification error
10	=	Validation error
11	=	Object identifier in use
12	=	Parameters improperly specified
13	=	Insufficient parameters specified
14	=	Unsupported option requested
15	=	Busy
16	=	Not available for processing
17	=	Command not valid for current state
18	=	No material altered
19	=	Material partially processed
20	=	All material processed
21	=	Recipe specification related error
22	=	Failed during processing
23	=	Failed while not processing
24	=	Failed due to lack of material
25	=	Job aborted
26	=	Job stopped
27	=	Job cancelled
28	=	Cannot change selected recipe
29	=	Unknown event
30	=	Duplicate report ID
31	=	Unknown data report
32	=	Data report not linked
33	=	Unknown trace report
34	=	Duplicate trace ID
35	=	Too many data reports
36	=	Sample period out of range
37	=	Group size too large
38	=	Recovery action currently invalid
39	=	Busy with another recovery currently unable to perform the recovery
40	=	No active recovery action
41	=	Exception recovery failed
42	=	Exception recovery aborted
43	=	Invalid table element
44	=	Unknown table element
45	=	Cannot delete predefined
46	=	Invalid token
47	=	Invalid parameter
0-63	=	reserved

Where Used: S1F20, S4F20, F22, F23, F33, F35, S5F14, F15, F18, S13F14, F16, S14F2, F4, F6, F8, F10, F12, F14, F16, F18, S15F28, F30, F32, F34, F36, F38, F40, F42, F44, F48, S17F2, F4, F6, F8, F10, F12, F14



ERRTEXT

Format: 20

Text string describing the error noted in the corresponding ERRCODE. Limited to 80 characters maximum.

Where Used: S1F20, S4F20, F22, F23, F33, F35, S5F70, F71, F74, S13F14, F16, S14F2, F4, F6, F8, F10, F12, F14, F16, F18, S15F28, F30, F32, F34, F36, F38, F40, F42, F44, F48, S17F4, F8, F14

ERRW7

Format: 20

Text string describing error found in process program

Where Used: S7F27

EVNTSRC

Format: 20

Object type for Event Source Objects.

Where Used: S17F5, F9, F10, F11, F12

EXID

Format: 20

Unique identifier for the exception. Maximum length of 20 characters.

Where Used: S5F9, F11, F13, F14, F15, F17, F18

EXMESSAGE

Format: 20

Text which describes the nature of the exception.

Where Used: S5F9, F11

EXRECVRA

Format: 20

Text which specifies a recovery action for an exception. Maximum length of 40 bytes.

Where Used: S5F9, F13

EXTYPE

Format: 20

Text which identifies the type of an exception. It is usually a single word of text.

"ALARM"
"ERROR"

Where Used: S5F9, F11, S14F1, F2, F8

FCNID

Format: 51

Function Identification

Where Used: S2F43, F44



FFROT

Format: 52

Film Frame Rotation

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

Where Used: S12F1, F3

FILDAT

Format: 10, 20

Data from the Data Set

The maximum length is the RECLen from Open Data Set Data.

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, F3, F4

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 program. Negative values are invalid.

Where Used: S7F34

GRANT

Format: 10

Grant code, 1 byte

- 0 = Permission Granted
- 1 = Busy, Try Again
- 2 = No Space Available
- 3 = Duplicate DATAID
- >3 = Equipment Specific Error Code
- 4-63 reserved

Where Used: S2F2, F40, S4F26, S13F12, S16F2



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

GRNT1

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, F4, F5, F6, F7, F8

HCACK

Format: 10

Host Command Parameter Acknowledge Code, 1 byte

- 0 = Acknowledge, command has been performed
- 1 = Command does not exist
- 2 = Cannot perform now
- 3 = At least parameter is invalid
- 4 = Acknowledge, command will be performed with completion signaled later by an event
- 5 = Rejected, Already in Desired Condition
- 6 = No such object exists
- 7-63 reserved

Where Used: S2F42, F50

HOACK

Format: 11

Conveys whether the corresponding handoff activity succeeded (=True) or failed (=False)

Where Used: S4F33, F35



HOCANCELACK

Format: 51

Tells whether the cancel ready message was accepted or rejected

- 0 = Cancel Ready Accepted.
- 1 = Atomic Transfer Unknown
- 2 = Cancel Ready Rejected - Handoff Begun.

Where Used: S4F13

HOCMDNAME

Format: 20

Identifier for the handoff command to be executed

Where Used: S4F31

HOHALTACK

Format: 51

Tells whether the halt command was accepted or rejected

- 0 = Halt Accepted.
- 1 = Atomic Transfer Unknown.
- 2-63 reserved

Where Used: S4F41

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, F3, F4, F5, F7, F9, F11, F13, F14, F15, F16, F17, F18

LENGTH

Format: 3(),5()

Length of the service program or process program in bytes

Where Used: S2F1; S7F1, F29

LIMITACK

Format: 10

Acknowledgment code for variable limit attribute set, 1 byte

- 1 = LIMITID does not exist
- 2 = UPPERDB > LIMITMAX
- 3 = LOWERDB < LIMITMIN
- 4 = UPPERDB < LOWERDB
- 5 = Illegal format specified for UPPERDB or LOWERDB
- 6 = ASCII value cannot be translated to numeric
- 7 = Duplicate limit definition for this variable
- >7 = Other equipment-specific error
- 8-63 reserved

Where Used: S2F46

LIMITID

Format: 10

The identifier of a specific limit in the set of limits (as defined by UPPERDB and LOWERDB) for a variable to which the corresponding limit attributes refer, 1 byte.

Where Used: S2F45, F46, F48

LIMITMAX

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

The maximum allowed value for the limit values of a specific variable. The equipment manufacturer should specify this value, which would typically coincide with the maximum value of the variable being monitored. The format must match that of the referenced variable.

Where Used: S2F48



LIMITMIN

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

The minimum allowed value for the limit values of a specific variable. The equipment manufacturer should specify this value, which would typically coincide with the minimum value of the variable being monitored. The format must match that of the referenced variable.

Where Used: S2F48

LINKID

Format: 54

Used to link a completion message with a request that an operation be performed. LINKID is set to the value of RMOPID in the initial request except for the last completion message to be sent, where it is set to zero.

Where Used: S6F25, S15F22, F30

LLIM

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

Lower limit for numeric value

Where Used: S7F22

LOC

Format: 10

Machine material location code, 1 byte

Where Used: S2F27, S3F2

LOWERDB

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

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

Where Used: S2F45, F48

LRACK

Format: 10

Link Report Acknowledge Code, 1 byte

- 0 = Accepted
- 1 = Denied. Insufficient space
- 2 = Denied. Invalid format
- 3 = Denied. At least one CEID link already defined
- 4 = Denied. At least one CEID does not exist
- 5 = Denied. At least one RPTID does not exist
- >5 = Other errors
- 6-63 reserved

Where Used: S2F36



LVACK

Format: 10

Variable limit definition acknowledge code, 1 byte. Defines the error with the limit attributes for the referenceVID.

- 1 = Variable does not exist
- 2 = Variable has no limits capability
- 3 = Variable repeated in message
- 4 = Limit value error as described in LIMITACK
- 5-63 reserved

Where Used: S2F46

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, F5

MCINDEX

Format: 5()

Identifier used to link a handoff command message with its eventual completion message. Corresponding messages carry the same value for this data item.

Where Used: S4F31, F33

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, F10, F12



MDLN

Format: 20

Equipment Model Type, 6 bytes max

Same data as returned by S1, F2

Where Used: S1F2; S7F22, F23, F26, F31

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 by 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 waterframes
- 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, F4, F5, F7

MHEAD

Format: 10

SECS message block header associated with message block in error

Where Used: S9F1, F3, F5, F7, F11

MID

Format: 10, 20

Material ID

16 Characters maximum

Where Used: S2F27, S3F2, F4, F7, F9, F12, F13, S4F1, F3, F5, F7, F9, F11, F13, F15, F17, S7F7, F8, F10, F11, F13, F35, F36, S12F1, F3, F4, F5, F7, F9, F11, F13, F14, F15, F16, F17, F18



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: S3F14

MIDRA

Format: 10

Material ID Acknowledge Code, 1 byte

- 0 = acknowledge, MID follows
- 1 = acknowledge, will not send MID
- 2 = acknowledge, will send MID later in S3F13
- 3-63 reserved

Where Used: S3F12

MLCL

Format: 5()

Message length

Defined by message size in (bytes)

Where Used: S12F4, F5

MMODE

Format: 10

Matrix mode select, 1 byte

- 1 = Host source mode
Use S7F7 and F8 to define the process program to be used.
- 2 = Local source mode
Use the matrix defined by S7F9, F10, F11 and F13 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, F3, F4

OBJACK

Format: 51

Acknowledge code:

0 = Successful completion of requested data.
1 = Error.
>1 reserved.

Where Used: S14F2, F4, F6, F8, F10, F12, F14, F16, F18

OBJCMD

Format: 51

Specifies an action to be performed by an object:

0 = reserved
1 = attach to requestor
2 = detach from requestor (requires authorization token)
3 = reattach to requestor
4 = set attributes (requires authorization token)
>4 reserved

OBJID

Format: 20, 5()

Identifier for an object

Where Used: S1F19, S14F1, F2, F3, F4



OBJSPEC

Format: 20

A text string that has an internal format and that is used to point to a specific object instance. The string is formed out of a sequence of formatted substrings, each specifying an object's type and identifier. The substring format has the following four fields:

object type, colon character ":", object identifier, greater-than symbol ">"

where the colon character ":" is used to terminate an object type and the "greater than" symbol ">" is used to terminate an identifier field. The object type field may be omitted where it may be otherwise determined. The final ">" is optional.

Where Used: S2F49, S13F11, F13, F15, S14F1, F3, F5, F7, F9, F10, F11, F13, F15, F16, F17, S15F43, F47

OBJTOKEN

Format: 54

Token used for authorization.

Where Used: S14, S15F37, F39, F41, F43

OBJTYPE

Format: 20, 5()

Identifier for a group or class of objects. All objects of the same type must have the same set of attributes available.

Where Used: S1F19, S14F1, F3, F6, F7, F8

OFLACK

Format: 10

Acknowledge code for OFF-LINE request.

0 = OFF-LINE Acknowledge
1-63 reserved

Where Used: S1F16

ONLACK

Format: 10

Acknowledge code for ON-LINE request.

0 = ON-LINE Accepted
1 = ON-LINE Not Allowed
2 = Equipment Already ON-LINE
3-63 reserved

Where Used: S1F18

OPID

Format: 5()

Operation ID. A unique integer generated by the requestor of an operation, used where multiple completion confirmations may occur.

Where Used: S6F25, S15F21, F29, F30, F37, F41, F44, F46



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, F3, F4

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-varied 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

PFCD

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 invalid

PMAX = 0 specifies there is no upper bound:

PMAX = 1 specified 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 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

PPARM

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, F26, F31

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, F6, F36



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, F30

PPID

Format: 10, 20

Process program ID.

Limited to a maximum of 80 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, F3, F5, F6, F8, F10, F11, F13, F17, F20, F23, F25, F26, F27, F31, F33, F34, F36, S9F13

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, F3



PRCMDNAME

Format: 20

Commands sent to a process job:

“START”
“STOP”
“PAUSE”
“RESUME”
“ABORT”
“CANCEL”

Where Used: S16F5

PRDCT

Format: 5()

Process Die Count

Number of die to be processed or number of die which have been processed. (Zero length indicates not used.)

Where Used: S12F1, F4

PREVENTID

Format: 5()

Processing related event identification:

- 1 - Waiting for material
- 2 - Job state change

Where Used: S16F9

PRJOBID

Format: 20

Text string which uniquely identifies a process job.

Where Used: S16F4, F5, F6, F7, F9

PRJOBMILESTONE

Format: 5()

Notification of Processing status shall have one of the following values:

- 1 - Job Setup
- 2 - Job Processing
- 3 - Job Processing Complete
- 4 - Job Complete
- 5 - Job Waiting for Start

Where Used: S16F7

PTN

Format: 10

Material Port number, 1 byte

Where Used: S4F1, F3, F5, F7, F9, F11, F13, F15, F17



QUA

Format: 10

Quantity in format, 1 byte

Where Used: S3F2, F4, F5, F7

RAC

Format: 31, 51

Reset acknowledge, 1 byte

- 0 = reset to be done
- 1 = reset denied
- > = other errors
- 2-63 reserved

Where Used: S2F20

RCMD

Format: 20, 31, 51

Remote command code or string

Where Used: S2F21, F41, F49

RCPATTRDATA

Format: 0, 10, 11, 20, 3(), 4(), 5()

The contents (value) of a recipe attribute.

Where Used: S6F25, S15F13, F15, F18, F27, F28, F30, F32

RCPATTRID

Format: 20

The name (identifier) of a non-identifier recipe attribute

Where Used: S6F25, S15F13, F15, F18, F27, F28, F30, F32

RCPBODY

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

Recipe body

Where Used: S15F13, F15, F18, F27, F32

RCPCCLASS

Format: 20

Recipe class

Where Used: S15F11



RPCCMD

Format: 51

Indicates an action to be performed on a recipe

5 = delete
8 = unprotect
9 = protect
10 = verify
11 = link
12 = unlink
13 = certify
14 = de-certify
15 = download
16 = upload
0-4, 6-7, 17-63 reserved

Where Used: S15F21, F22

RCPDEL

Format: 51

0 = delete
1 = deselect
> 1 reserved

Where Used: S15F35

RCPDESCLTH

Format: 5()

The length in bytes of a recipe section

Where Used: S15F24

RCPDESCNM

Format: 20

Identifies a type of descriptor of a recipe: "ASDesc", "BodyDesc", "GenDesc".

Where Used: S15F24

RCPDESCTIME

Format: 20

The timestamp of a recipe section, in the format "YYYYMMDDhhmmsscc".

Where Used: S15F24

RCPID

Format: 20

Recipe identifier. Formatted text conforming to the requirements of OBJSPEC.

Where Used: S15F21, F23, F28, F29, F30, F33, F35, F37, F41, F44



RCPNAME

Format: 20

Recipe name

Where Used: S15F11

RCPNEWID

Format: 20

The new recipe identifier assigned as the result of a copy or rename operation.

Where Used: S15F19, F41, F44

RCPOWCODE

Format: 11

Indicates whether any pre-existing recipe is to be overwritten (=TRUE) or not (=FALSE) on download.

Where Used: S15F27

RCPPARNM

Format: 20

The name of a recipe variable parameter. Maximum length of 40 characters.

Where Used: S15F25, F33, S16F3

RCPPARRULE

Format: 20

The restrictions applied to a recipe variable parameter setting. Maximum length of 80 characters.

Where Used: S15F25

RCPPARVAL

Format: 10, 11, 20, 3(), 4(), 5()

The initial setting assigned to a recipe variable parameter. Text form restricted to maximum of 80 characters.

Where Used: S15F25, F33, S16F3

RCPRENAME

Format: 11

Indicates whether a recipe is to be renamed (= TRUE) or copied (= FALSE).

Where Used: S15F19



RCPSECCODE

Format: 10

Indicates the sections of a recipe requested for transfer or being transferred:

- 1 = generic attributes only
- 3 = generic attributes and body
- 4 = all agent-specific datasets (zero or more)
- 7 = generic attributes, body, and all agent-specific datasets
- 8 = single agent-specific dataset (zero or one)
- 11 = generic attributes, body, and single agent-specific datasets
- All other values reserved

Where Used: S15F15, F16, F17

RCPSECNM

Format: 20

Recipe section name: "Generic", "Body", or "ASDS".

Where Used: S15F15, F18

RCPSPEC

Format: 20

Recipe specifier. The object specifier of a recipe.

Where Used: S15F1, F9, F13, F15, F17, F19, F27, F28, F31, F32, F45

RCPSTAT

Format: 51

The status of a managed recipe

- 0 = does not exist
- 8 = unprotected
- 9 = protected

Where Used: S15F10

RCPUPDT

Format: 11

Indicates if an existing recipe is to be updated (= True) or a new recipe is to be created (= False).

Where Used: S15F13

RCPVERS

Format: 20

Recipe version

Where Used: S15F10, F12

READLN

Format: 3(), 5()

Maximum length to read

Where Used: S13F5



RECLEN

Format: 3(), 5()

Maximum length of a Discrete record

Where Used: S13F4

REFP

Format: 3()

Reference Point

Where Used: S12F1, F4

REPGSZ

Format: 20, 3(), 5()

Reporting group size

Where Used: S2F23, S17F5

RESC

Format: 31, 51

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

Where Used: S7F22

RESPEC

Format: 20

Object specifier for the recipe executor

Where Used: S15F29, F33, F35

RESV

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

Resolution value for numeric data

If RESC=1, then RESV contains smallest increment allowed for parameter

Where Used: S7F22

RIC

Format: 31, 51

Reset code, 1 byte

- 0 = not used
- 1 = power up reset
- >1 other reset conditions
- 2-63 reserved

Where Used: S2F19



RMACK

Format: 51

Conveys whether a requested action was successfully completed, denied, completed with errors, or will be completed with notification to the requestor.

- 0 = successful completion
- 1 = cannot perform action
- 2 = completed with errors
- 3 = action will be completed and notification sent
- 4 = no request for this action exits

Where Used: S15F4, F6, F8, F10, F12, F14, F16, F18, F20, F22, F24, F25, F28, F30, F32, F34, F36, F38, F40, F42, F44, F48

RMCHGSTAT

Format: 5()

Indicates the change that occurred for an object.

- 0 = no change
- 1 = created
- 2 = updated
- 3 = stored (new)
- 4 = replaced
- 5 = deleted
- 6 = copied (new object)
- 7 = renamed
- 8 = unprotected
- 9 = protected
- 10 = verified
- 11 = linked
- 12 = unlinked
- 13 = certified
- 14 = de-certified
- 15 = selected
- 16 = deselected

Where Used: S15F25



RMCHGTYPE

Format: 5()

Indicates the type of change for a recipe.

- 0 = no change
- 1 = create
- 2 = update
- 5 = delete
- 6 = copy (new object)
- 7 = rename
- 8 = unprotect
- 9 = product
- 10 = verify
- 11 = link
- 12 = unlink
- 13 = certify
- 14 = de-certify
- 15 = change generic attribute
- 16 = change agent-specific attribute
- 17 = change both generic and agent-specific attributes

Where Used: S15F37, F41, F44, F46, F47, F48

RMDATASIZE

Format: 5()

The maximum total length, in bytes, of a multi-block message, used by the receiver to determine if the anticipated message exceeds the receiver's capacity.

Where Used: S15F1

RMGRNT

Format: 10

Grant code, used to grant or deny a request. 1 byte.

- 0 = permission granted
- 1 = cannot accept now, try again
- 2 = no space
- 3 = request is on hold
- 4-64 reserved

Where Used: S15F2, F37, F46

RMNEWNS

Format: 20

New name (identifier) assigned to a recipe namespace

Where Used: S15F5



RMNSCMD

Format: 51

Action to be performed on a recipe namespace

1 = created
5 = deleted
0, 2-4, 6-63 reserved.

Where Used: S15F3

RMNSSPEC

Format: 20

The object specifier of a recipe namespace

Where Used: S15F3, F5, F7, F11, F21, F23, F25, F47

RMRECSPEC

Format: 20

The object specifier of a distributed recipe namespace recorder.

Where Used: S15F39, F41, F43

RMREQUESTOR

Format: 11

Set to TRUE if initiator of change request was an attached segment. Set to FALSE otherwise.

Where Used: S15F41, F44, F46

RMSEGSPEC

Format: 20

The object specifier of a distributed recipe namespace segment.

Where Used: S15F37, F39, F41, F44, F46, F49

RMSPACE

Format: 5 ()

The amount of storage available for at least one recipe in a recipe namespace, in bytes.

Where Used: S15F8

ROWCT

Format: 5()

Row count in die increments

Where Used: S12F1, F4

RPSEL

Format: 51

Reference Point Select.

Number of reference point from 0-n.

Where Used: S12F1, F4