

Ignition 8.3 User Manual — SECS/GEM Module

Generated 2026-10-12 from
<https://www.docs.inductiveautomation.com/docs/8.3/>

SECS/GEM

NOTE

The SECS/GEM dropdown is available only when the SECS/GEM module is installed.

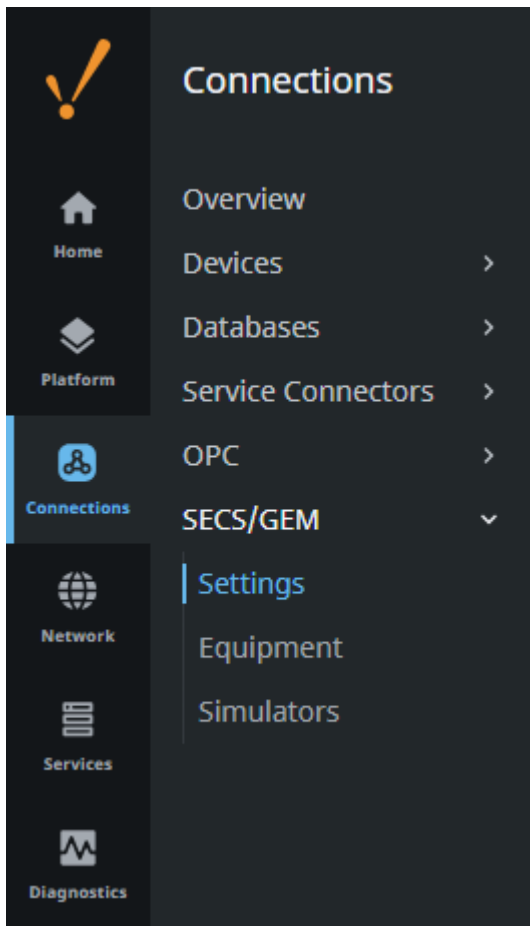
The SEMI Equipment Communications Standard/Generic Equipment Model (SECS/GEM) module enables Ignition projects and third party applications to communicate with semiconductor fab equipment. The module is an implementation of the GEM standard (SEMI E30-0307), the SECS-II standard (SEMI E5-0712) and the HSMS standard (SEMI E37-0308).

SECS/GEM Host

The module is implemented as a host that can talk to one or more tools via equipment connections configured within the Gateway. While it has some built-in simulation functionality to aid with initial setup, the module serves as a host only, and cannot respond as if it were equipment.

Communication with equipment is in the form of messages defined by the SECS-II standard and referred to as SECS messages or just messages. By use of a simple definition file, any SECS message (including custom ones) can be supported.

Once the module is installed, a new header is available under the Connections section of the Gateway that allows you to configure connections to the equipment, or set up simulators that are used for testing.



Equipment Connections

The **equipment connections** are what allow Ignition to communicate with the specialized semiconductor fab equipment. Equipment connections are setup within the Connections section of the Gateway. Connections can be made over Ethernet using TCP/IP (HSMS protocol) or over a direct serial connection (SECS-I protocol) using a RS-232 cable.

The equipment connections also require the use of a database, which will store sent and received messages for audit purposes, and to hold some configuration data. Each equipment connection connects to one tool and one database and transfers SECS messages between the Gateway to the tool. Any number of equipment connections can be configured in the Gateway as long as your license allows it, and different equipment connections can be configured to use the same database connection or different database connections.

SECS Definition Language (SDL) File

The **SDL file** is where SECS messages are defined. Any SECS message defined in this file can be sent and received. All SECS messages that are sent and received are validated against this file. If a message is not defined or doesn't match a definition then it is not sent, and it is instead inserted into the Errors table as a validation error along with information about why it didn't validate.

Each Equipment Connection has one SDL file and a default SDL file will be used if an alternate is not specified. The SDL file definitions are based on the GEM and SECS standards.

Simulators

The SECS/GEM module includes the capability to create and configure **equipment simulators**. The simulators are not a full implementation of the GEM standard. Equipment Connections can connect to simulators and they can exchange various messages. A range of SECS messages are supported by the simulators. This capability exists for getting started with the SECS/GEM module and for testing applications.

SECS/GEM Message Basics

Once a connection has been properly configured to a tool, messages can be sent between Ignition and the equipment. These messages are often complex, which is why SECS messages are often in the form of a JSON string, which can contain any number of lists and name/value pairs. The messages are typically sent and received using specialized scripting functions within Ignition, such as **system.secsgem.sendRequest** and **system.secsgem.getResponse**, that the SECS/GEM module adds.

To send a message, a user calls the `system.secsgem.sendMessage` function in a Python script within an Ignition Gateway or client, and passes the SECS message as **JSON** text to the function. A transaction id is returned, which can then be passed to the `system.secsgem.getResponse` function. This function returns the response from the tool as JSON text. However, some SECS messages can be handled differently, such as those that are sent repeatedly from the equipment like status variables. Those messages can be configured to fire Message Handler scripts in the Gateway or Client when the message is received, which can handle the constant flow of messages in a consistent manner. Alternately, the messages can be configured to create Tags with their values, updating the Tags anytime a new message comes in.

Definitions

The SECS/GEM module uses some unique terms that are not used in the rest of Ignition:

Term	Description
SECS/GEM	The SECS/GEM module is named after two standards the SECS-II standard (SEMI E5-0712) and the GEM standard (SEMI E30-0307). The SECS-II standard defines messages that can be exchanged between a host and an equipment. SECS stands for Semi Equipment Communications Standard. The GEM standard defines ways for equipment to implement functionality using SECS messages. GEM is short for Generic Model for Communications and Control of Manufacturing Equipment.
HSMS Protocol	A TCP/IP based message transfer protocol for sending SECS messages. The SECS/GEM module uses HSMS or SECS-I for sending messages.
Equipment	A semiconductor device that sends and receives SECS messages. This is also called a tool.
Equipment Connection	An Equipment Connection defines how Ignition communicates with a specific piece of equipment. It includes the communication settings for the equipment and an associated database connection used for configuration, auditing, and diagnostics.
Stream	A category of SECS messages.
Function	A specific SECS message in a stream.
Request Message	Also primary message. A SECS message having an odd numbered function that originates a communication and may require a response message.
Response Message	Also reply message. Also secondary message. A SECS message with an even numbered function that is a response to a request message.
SDL File	Also SECS Definition Language file. A file consisting of definitions of SECS messages and definitions of items used in SECS messages. This file is used to validate SECS messages that are sent and received and to add various functionality having to do with them.
SECS-1	A serial-based message transfer protocol used for sending SECS messages. The SECS-I standard is SEMI E4.

Term	Description
SECS Message	Also just message. A communication message in the format specified by the SECS-II standard that is used to communicate with an equipment.
Transaction	A request message and its response message if required.
TransactionID	Also TxID. Every request message has a unique 32-bit integer identifier that identifies the transaction. Every response message has the same TransactionID as the request message it responds to. In the HSMS and SECS-1 protocols, this is called System Bytes.

Using the SECS/GEM Module

This page walks you through setting up a basic SECS/GEM environment using a simulator and a demo project. By the end of this tutorial, you will have a working equipment connection and be able to send and view SECS/GEM messages in Ignition.

Downloading and Installing the Module

You can download the SECS/GEM module from the [Ignition Downloads](#) page under the **Device Connectivity** section. After downloading the module, install it from the Gateway by navigating to **Platform > System > Modules**. For detailed installation steps, refer to [Installing or Upgrading a Module](#).

NAMING CONVENTIONS

Before getting started, establish a consistent naming convention for simulators, equipment connections, databases, and table prefixes. This makes it easier to identify related components when working with multiple tools and environments.

Configuring the Database Connection

To set up a database connection for use with the SECS/GEM module, go to the Gateway and navigate to **Connections > Databases > Connections**, then create a new database connection. In this tutorial, the database connection name used is **SecsGem**.

The SECS/GEM module supports common relational databases such as MySQL, Microsoft SQL Server, and Oracle. The SECS/GEM Demo Project used later in this tutorial currently supports MySQL only. For more information about database configuration, see [Connecting to Databases](#).

Configuring the Simulator

The simulator can be used to emulate SECS/GEM equipment for development and testing without requiring access to physical equipment. To set up a simulator, follow the steps below.

1. On the Gateway, go to **Connections > SECS/GEM > Simulators**.
2. Click **Create Simulator +**.

3. Configure the following fields:

- **Name:** Enter a name for the simulator. For example, **SimEquip**.
- **Active Port:** Use **5000**.
- **Passive Port:** Use **5000**.
- **Device ID:** Use **0**, unless you need a different unique equipment identifier.
- **Connection Mode:** Use **Passive**.

MULTIPLE CONFIGURED SIMULATORS

If multiple simulators are configured, each simulator must have a unique Active and Passive port number.

4. Click **Create Simulator**.

Create Simulator

GENERAL

Name

SimEquip

Choose a name for this simulator.

Description

Enabled

☐ Disabling a simulator will prevent communication with an Equipment Connection.

Active IP Address

localhost

IP Address of Equipment Connection to connect to. Used when simulator is in Active mode.

Active Port

5000

Port number of Equipment Connection to connect to. Used when simulator is in Active mode.

Passive IP Address

Cancel

Create Simulator

After creation, the simulator appears in the list with a status of **Not Connected** or **Connecting**. The status changes to **Communicating** once an equipment connection is configured to connect to the simulator. For more information, see the [SECS/GEM Simulator](#) page.

TRIAL MODE RESET

If the SECS/GEM module is running in trial mode, simulator and equipment communication stops when the Gateway trial period expires after two hours. Reset the trial mode from the Gateway to resume communication. For more information, see the [Trial Period](#) section.

Creating an Equipment Connection

After configuring the simulator, or when connecting to physical equipment, you'll next need to create an equipment connection. To create an equipment connection, follow the steps below.

1. On the Gateway, navigate to the **Connections > SECS/GEM > Equipment** page.
2. Click **Create Equipment Connection +**.
3. Select **Ethernet/HSMS Connections** for this example, then click **Next**.
4. Configure the following fields:
 - **Equipment Name:** Enter a name for the connection. For example, **Equip_MySim**.
 - **Database Connection:** Select the database connection used for SECS/GEM message storage. For example, **SecsGem**.
 - **Active Port:** Use **5000**. This must match the simulator configuration.
 - **Passive Port:** Use **5000**.
5. Click **Create Equipment Connection**.

Create Equipment Connection

Create Equipment Connection

Choose Equipment Type

Setup Equipment Connection

MAIN

*Required

Name *

Equip_MySim

The name of this Equipment Connection.

Description

Enabled

☒ Disabling a connection will prevent communication to the target equipment.

Connection mode

Active

Method used to connect to equipment.

Active IP Address *

Back

Create Equipment Connection

After creation, the equipment connection appears in the list with a status of Initializing. When the status changes to Communicating, the connection to the simulator or equipment is established and ready to exchange SECS/GEM messages. For more information, see the [SECS/GEM Equipment Connections](#) page.

Connections

Filter equipment...

View 20

Create Equipment Connection +

NAME	SDL FILE	TYPE	SENT MESSAGES	RECEIVED MESSAGES	STATUS	
{itemName}	messages.sdl	Serial/SECS-I	0	0	INITIALIZING	
Equip_MySim	messages.sdl	Ethernet/HSMS	0	0	COMMUNICATING	

Showing 2 of 2 items

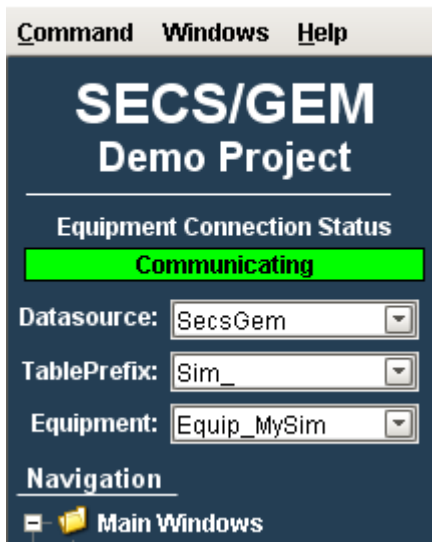
Using the SECS/GEM Demo Project

You can use the SECS/GEM Dem Project to explore messaging and visualization features once you have an equipment connection. The demo project includes prebuilt views for sending SECS messages and inspecting responses. The demo project is available from the [SECS/GEM Demo Project](#) page. For information on importing projects from the Exchange, see [Ignition Exchange](#).

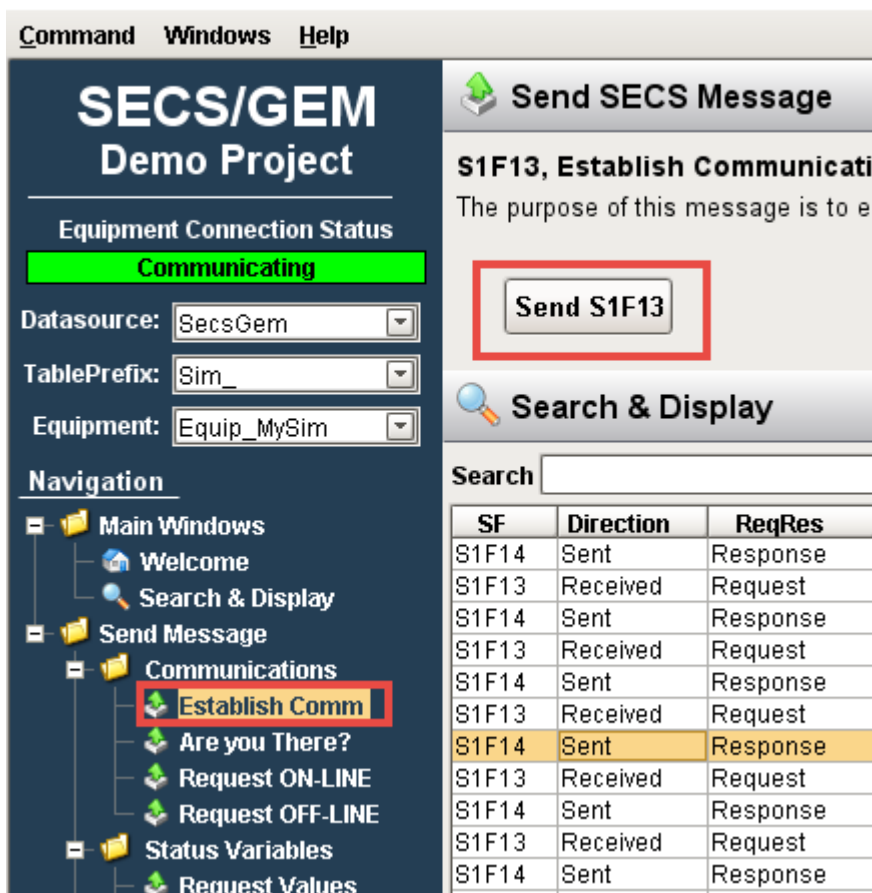
After restoring the demo project, launch a Client from the Gateway under the **Home** section. For more information, see [Launching Clients](#).

When the Client opens, select the following values from the dropdown menus in the upper-left corner of the application:

- **Datasource:** Select **SecsGem**.
- **TablePrefix:** Select **Sim_**.
- **Equipment:** Select **Equip_MySim**.



You can now view and send SECS messages. From the navigation tree, select **Establish Comm**, then click **Send S1F13**. Messages appear in the **Search & Display** table.



Display Tabs

The Display tabs show detailed information for each SECS request and response. The available tabs are **Simple**, **Tree**, **Table**, **JSON**, and **Python**. Select a request or response from the **Search & Display** table, then switch tabs to view the message in different formats.

The examples below show the response message **S1F14** returned from the **Establish Comm** request (**S1F13**).

Simple Tab

The **Simple** tab provides a human-readable summary of the SECS message. It displays the message name, a short description, and key header information such as whether a reply is expected and the message direction. For the **S1F14 – Establish Communications Request Acknowledge** message, the Simple tab indicates that the request was **accepted** by the equipment.

SimpleTreeTableJSONPython

S1F14, Establish Communications Request Acknowledge
Accept or deny Establish Communications Request (S1F13).

Reply expected: False

Direction: Response from Equipment Connection sent to Equipment

Body:

Establish Communications

Acknowledge Code

Accepted

Equipment Model Type

Software Revision Code:

Tree Tab

The **Tree** tab displays the SECS message as a hierarchical structure. The message is divided into **header** and **body** sections, which can be expanded to view individual fields and nested items. This view is useful for understanding how message elements are organized and how lists and values relate to one another.

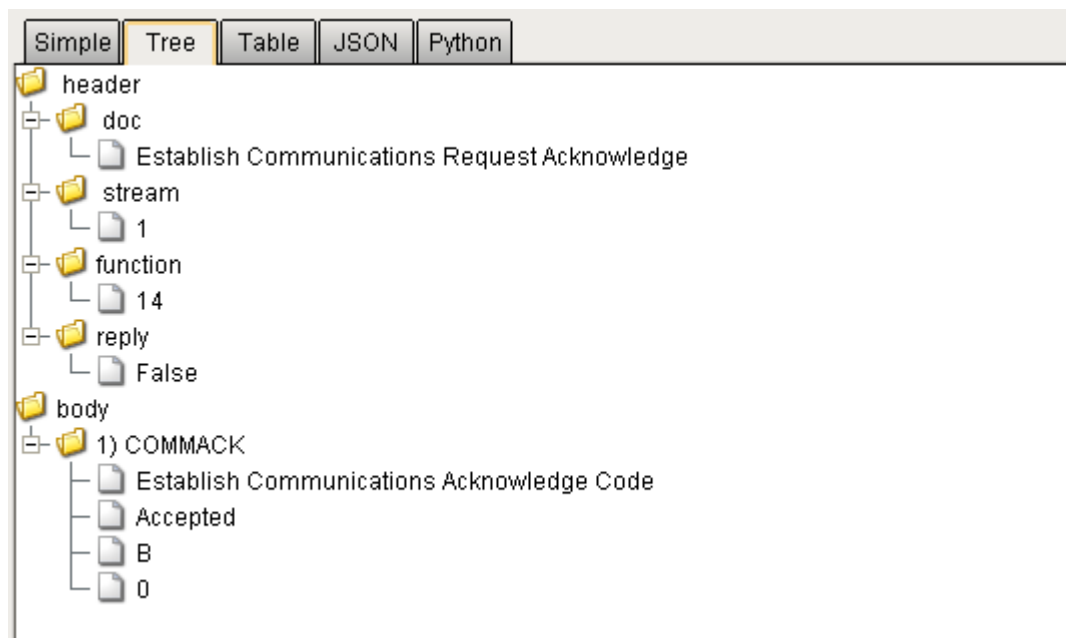


Table Tab

The **Table** tab presents the message data in a tabular format. Each row represents a single message element and includes the element path, description, data format, and value. This view is helpful for quickly scanning message contents or locating specific fields.

Simple	Tree	Table	JSON	Python
path	doc	cod...	format	value
header/doc	Establish Communications Request Acknowl...			
header/stream				1
header/function				14
header/reply				False
body/1) COMMACK	Establish Communications Acknowledge Code	Accept...	B	0

JSON Tab

The **JSON** tab shows the message in the JSON format used internally by Ignition. This is the same structure used when sending and receiving SECS messages through scripting. This view is useful when comparing received messages to SDL definitions or when writing scripts that construct or parse SECS messages.

Simple	Tree	Table	JSON	Python
--------	------	-------	------	--------

```
{
  "body": [
    {
      "codeDesc": "Accepted",
      "doc": "COMMACK, Establish Communications Acknowledge Code",
      "format": "B",
      "value": 0
    },
    []
  ],
  "header": {
    "doc": "Establish Communications Request Acknowledge",
    "function": 14,
    "reply": false,
    "stream": 1
  }
}
```

Python Tab

The **Python** tab displays the message as a Python dictionary. This reflects how the message appears when accessed in Python scripts, such as when handling responses or processing messages in Message Handlers. This view is useful for understanding how to reference message fields programmatically.

Simple	Tree	Table	JSON	Python
--------	------	-------	------	--------

```
{u'body': [{u'codeDesc': 'Accepted',
            u'doc': 'COMMACK, Establish Communications Acknowledge Code',
            u'format': 'B',
            u'value': 0},
          []],
 u'header': {u'doc': 'Establish Communications Request Acknowledge',
            u'function': 14,
            u'reply': False,
            u'stream': 1}}
```

SECS/GEM Equipment Connections

Connection Types

Equipment connections can be configured using either Ethernet or serial communication, depending on what the equipment supports. Ethernet connections use the HSMS protocol (SEMI E37) over TCP/IP and are the preferred option due to higher performance and reliability. Serial connections use the SECS-I protocol (SEMI E4) over an RS-232 connection and require the equipment to be directly connected to the Ignition Gateway server.

Equipment Connection Properties

When setting up an equipment connection through either ethernet or serial, there are various properties that can be configured. For more details on adding a connection, see the [Creating an Equipment Connection](#) example.

Common Properties

Property Name	Description
Equipment Name	The name of the equipment connection.
Equipment Description	A description of the equipment connection.
Enabled	Enables or disables communication between the Gateway and the equipment. Default is TRUE.
Device ID	A unique integer identifier defined by the equipment. Default is 0.
Database Connection	The database connection used for configuration data and message auditing.
Database Table Prefix	An optional prefix added to database table names to distinguish records for different equipment connections.
SECS Definition Language (SDL) File	The SDL file used to validate SECS messages. If not specified, the default messages.sdl file is used.
T3 Reply Timeout	The number of seconds to wait for an expected SECS reply message. Default is 45.

Ethernet/HSMS Connection Properties

These properties apply only to equipment connections that use the Ethernet/HSMS connection type.

Property Name	Description
Active IP Address	IP Address of the equipment to connect to when in Active mode. Default is localhost.
Active Port	The Port number of the equipment to connect to when in Active mode, which must match the port number used to create the Simulator. Default is 5000.
Passive IP Address	IP Address of the equipment to connect to when in Passive mode. Default is localhost.
Passive Port	The Port number that an Equipment will connect to if the SECS/GEM module is in passive mode. Default is 5000.
Connection Mode	The method used to connect to Equipment. • ACTIVE: Will attempt to connect to the equipment at the given Active IP Address and Active Port. Default. • PASSIVE: Will listen for a connection from the equipment at the given Passive IP Address and Passive Port. • ALTERNATING: Will switch between Active mode and Passive mode until a connection is made.
T5 Connect Separation Timeout	The number of seconds which must elapse between successive attempts to connect to the equipment after disconnection. Default is 10.
T6 Control Transaction Timeout	The number of seconds which a control transaction (such as LinkTest or Select) may remain open before it is considered a communications failure. Default is 5.
T7 Not Selected Timeout	The number of seconds which a TCP/IP connection can remain in NOT SELECTED state (i.e. no HSMS activity) before it is considered a communications failure. This timeout is only used for Passive Connection Mode. Default is 10.
T8 Network Intercharacter Timeout	The maximum number of seconds between successive bytes of a single HSMS message before it is considered a communications failure. This applies to HSMS messages received from the equipment. Default is 5.

Property Name	Description
Keep Alive Timeout	The number of seconds interval for sending LinkTest control messages for testing a connection. Automatically reconnects if the test fails. A value of 0 will disable it. Default is 300.

Serial/SECS-I Connection Properties

These properties apply only to equipment connections that use the Serial/SECS-I connection type.

Property Name	Description
Serial Port Name	The name of the serial port. Default is COM1.
Data Bit Rate	The bit rate of data that is being sent and received. Default is BR_9600. Possible values are: - BR_9600 - BR_4800 - BR_2400 - BR_1200 - BR_19200 - BR_150
T1 Inter-Character Timeout	The maximum number of seconds allowed for interruptions between characters being sent. Default is 0.5.
T2 Protocol Timeout	The maximum number of seconds for a lack of protocol response. Default is 10.
T4 Inter-Block Timeout	The maximum number of seconds allowed for interruptions in multi-block messages. Default is 45.
Retry Limit	The maximum number of send retries allowed. Default is 3.

Database Tables

SECS/GEM creates several database tables to store equipment connection information, message history, and errors. These tables support auditing and diagnostics and are not required for normal module operation.

The tables created for an equipment connection depend on the selected database connection and optional table prefix.

 CAUTION

Ignition does not perform automatic maintenance on SECS/GEM database tables. Some tables can grow large over time, so it's recommended that users manage retention, pruning, or partitioning as needed.

SECSGEM_EquipmentInfo

The **SECSGEM_EquipmentInfo** table stores summary information for all equipment connections that use the same database connection. This table is created once per database connection and does not use a table prefix.

Each row represents a single equipment connection.

Column Name	Datatype	Description
Equipment	String	The name of the equipment connection.
Prefix	String	The table prefix used by the equipment connection.
Status	String	The current status of the equipment connection.
SDL File	String	The SDL file associated with the equipment connection. If none is uploaded, the default messages.sdl file is used.

Messages

The Messages table stores SECS messages that are sent or received by an equipment connection. If a table prefix is specified, it is included in the table name. Multiple equipment connections can share the same prefix and write to the same Messages table.

prefixMessages

Column Name	Datatype	Description
ID	Auto-incrementing integer	The unique row identifier.
Equipment	String	The name of the equipment connection.
StreamFunction	String	The SECS Stream and Function. For example, S1F1 .
Direction	String	Indicates whether the message was sent or received.
RequestResponse	String	Indicates whether the message is a request or a response.
CommonID	String	An identifier used to associate related messages.
TxID	Integer	The transaction identifier for the message.
Reply	Integer	Indicates whether the message expects a reply.
Message	String	The SECS message represented as JSON.
TimeSentReceived	DateTime	The date and time the message was sent or received.

Errors

The Errors table records errors that occur during SECS/GEM communication. If a table prefix is specified, it is included in the table name. Multiple equipment connections can write to the same Errors table when they share a prefix.

prefixErrors

Column Name	Datatype	Description
ID	Auto-incrementing integer	The unique row identifier.
Equipment	String	The name of the equipment connection.
StreamFunction	String	The SECS Stream and Function associated with the error, if applicable.
ErrorType	String	The category of error, such as connection, timeout, or validation errors.
Error	String	The error message.
Time	DateTime	The date and time the error occurred.

SECS/GEM Equipment Page

The SECS/GEM Equipment page displays all configured equipment connections, their current status, and message activity.

Attribute	Description
Active Connections	The total number of active equipment connections.
Errored Connections	The total number of errored equipment connections.
Aggregate Throughput	The combined message throughput across all connections.
Name	The equipment connection name.
Type	The type of equipment connections, either Ethernet/HSMS Connection or Serial/SECS-I Connection.
Sent Messages	The total number of messages sent to the equipment.
Received Messages	The total number of messages received from the equipment.
Status	The current connection status. If a connection becomes faulted or disconnected, its status is shown as Not Connected .

SECS/GEM Equipment

Status

ACTIVE CONNECTIONS

1 / 1

ERRORED CONNECTIONS

0

AGGREGATE THROUGHPUT

0

per second

Connections

Filter equipment connections...

View20

Create Equipment Connection +

☐ NAME ↑	TYPE	SENT MESSAGES	RECEIVED MESSAGES	STATUS	
☐ Equip_MySim	Ethernet/HSMS Connection	2	1	COMMUNICATING	

Showing 1 of 1 items

SECS/GEM Equipment Details

Clicking the three dots menu to the right of an equipment connection and selecting **View Details** will show more detailed information for that particular piece of equipment.

Messages Sent

Attribute	Description
Throughput	The messages per second the system is currently sending.
Average	The average messages sent per second from when the equipment was first enabled.
Total	The total messages sent.

Messages Received

Attribute	Description
Throughput	The messages per second the system is currently receiving.
Average	The average messages received per second from when the equipment was first enabled.
Total	The total messages received.



Equip_MySim Details

Messages Sent

Last to: S1F14

THROUGHPUT

0

Messages Per Second

AVERAGE

0

Messages Per Second

TOTAL

6

Messages Received

Last from: S1F13

THROUGHPUT

0

Messages Per Second

AVERAGE

0

Messages Per Second

TOTAL

3

SECS Definition Language (SDL) File

The SECS Definition Language (SDL) file defines the structure and content of SECS messages for an equipment connection. All sent and received messages are validated against the SDL file, and messages that fail validation are logged to the Errors table for the equipment connection. The SDL file uses JSON syntax to define SECS datatypes, items, and messages.

Each equipment connection can use a single SDL file. If no SDL file is specified, the **default messages.sdl** file is used. The default SDL file is based on the SECS-II standard (SEMI E5-0712) and the GEM standard (SEMI E30-0307).

SDL files can be edited directly in the Gateway using the built-in SDL editor, or downloaded from the SECS/GEM Equipment page, edited externally, and re-uploaded.

SDL File Format

The SDL file is formatted using JSON and follows a consistent structure. The file contains four main sections:

- **doc**: Provides documentation for the SDL file. Including clear documentation is recommended to make future maintenance easier.
- **formats**: Defines the SECS datatypes used by items. Only datatypes referenced by items need to be included.
- **items**: Defines reusable SECS items used in message bodies. Items are grouped alphabetically by name.
- **messages**: Defines SECS messages by stream and function, including message direction, body structure, and behavior.

Example SDL File

```

{
  "doc": ["The doc for the file."],
  "formats": {
    "A": { "doc": "ASCII" },
    "I2": { "doc": "Signed Integer Two Bytes" }
  },
  "items": {
    "A": {
      "ALTX": {
        "doc": ["ALTX, Alarm Text"],
        "formats": ["A"],
        "max": 120
      }
    }
  },
  "messages": {
    "S1": {
      "doc": ["Stream 1 Equipment Status"],
      "S1F1": {
        "doc": ["Are You There Request"],
        "block": "single",
        "direction": "h<->e",
        "reply": true
      }
    }
  }
}

```

SECS Datatype Formats

The formats section defines the datatypes used by items in the SDL file. Formats may reference single datatypes or consolidate multiple related datatypes into a single logical format, which can simplify item definitions. For example, multiple integer sizes can be grouped under a single I format.

SDL Formats Definition Example

```
{
  "A": { "doc": "ASCII" },
  "B": { "doc": "Binary" },
  "I": {
    "doc": "Signed Integer",
    "formats": ["I1", "I2", "I4", "I8"]
  },
  "F": {
    "doc": "Floating Point",
    "formats": ["F4", "F8"]
  }
}
```

SDL Item Definitions

The items section defines all SECS items that can be referenced by message bodies. If a message references an item that is not defined in the SDL file, the message will fail validation.

SDL Item Definition Example

```
{
  "A": {
    "ACKC7": {
      "doc": ["ACKC7, Acknowledge Code"],
      "formats": ["B"],
      "bytes": 1,
      "codes": {
        "0": "Accepted",
        "1": "Permission not granted",
        ">": [6, "Other error"]
      }
    }
  }
}
```

Each item definition may include properties such as documentation, allowed formats, byte length, enumerated codes, and optional validation constraints.

SDL Message Definitions

The messages section defines all SECS messages supported by the equipment connection. Messages are grouped by stream, with each function defining the message direction, structure, and behavior.

SDL Message Definition Example

```
{
  "S6F1": {
    "doc": [
      "Trace Data Send",
      "Sent by the equipment to report trace data samples."
    ],
    "block": "multiple",
    "direction": "h<-e",
    "reply": false,

    "CommonID": ["TRID"],
    "realtime": true,
    "SQLTags": true,

    "body": [
      "TRID",
      "SMPLN",
      "STIME",
      ["SV", "..."]
    ]
  }
}
```

SDL Message Defaults

The messages section typically starts with a default set of properties before the list of SECS messages. These default property values will be used in the message when the property is not present in the message definition.

SDL Message Defaults

```
{
  "defaults":{
    "doc":[
      "Default values for properties that are missing."
    ],
    "block":"single",
    "reply":false,
    "direction":"h<->e",
    "body":null
  }
}
```

SDL File Editor

You can edit an equipment connection's SDL file directly from the SECS/GEM Equipment page. Open the three dots menu for the equipment connection and select **Edit File**. After editing and saving the changes for the SDL file, the equipment connection must be restarted for the changes to take effect.

Default SDL File

The default **message.sdl** file is used when no other SDL file is specified.

The messages.sdl Default File.

```

{
  "doc":["SECS stream/functions and items are listed and defined in this
document."],

  "formats":{
    "A":{"doc":"ASCII"},
    "B":{"doc":"Binary"},
    "B64":{"doc":"Binary encoded as Base64 string"},
    "J":{"doc":"JIS-8"},
    "L":{"doc":"List"},
    "Bool":{"doc":"Boolean"},
    "Local":{"doc":"Two Byte Character String"},
    "U1":{"doc":"Unsigned Integer One Byte"},
    "U2":{"doc":"Unsigned Integer Two Bytes"},
    "U4":{"doc":"Unsigned Integer Four Bytes"},
    "U8":{"doc":"Unsigned Integer Eight Bytes"},
    "I1":{"doc":"Signed Integer One Byte"},
    "I2":{"doc":"Signed Integer Two Bytes"},
    "I4":{"doc":"Signed Integer Four Bytes"},
    "I8":{"doc":"Signed Integer Eight Bytes"},
    "F4":{"doc":"Floating Point Four Bytes"},
    "F8":{"doc":"Floating Point Eight Bytes"},
    "I":{"doc":"Signed Integer", "formats":["I1","I2","I4","I8"]},
    "U":{"doc":"Unsigned Integer", "formats":["U1","U2","U4","U8"]},
    "F":{"doc":"Floating Point", "formats":["F4","F8"]}},

  "items":{
    "A":{
      "ACCESSMODE":{
        "doc":["ACCESSMODE, Load Port Access Mode"],
        "formats":["U1"],
        "codes":{
          "0":"Manual",
          "1":"Auto"}},
      "ACKA":{
        "doc":["ACKA, Indicates success of a request"],
        "formats":["Bool"],
        "codes":{
          "0":"False",
          "1":"True"}},
      "ACKC5":{
        "doc":["ACKC5, Acknowledge Code"],
        "formats":["B"],
        "bytes":1,
        "codes":{
          "0":"Accepted",
          ">":["0","Error, not accepted"]}},
      "ACKC6":{

```

```

    "doc":["ACKC6, Acknowledge Code"],
    "formats":["B"],
    "bytes":1,
    "codes":{
        "0":"Accepted",
        ">":["0","Error, not accepted"]}},
"ACKC7":{
    "doc":["ACKC7, Acknowledge Code"],
    "formats":["B"],
    "bytes":1,
    "codes":{
        "0":"Accepted",
        "1":"Permission not granted",
        "2":"Length error",
        "3":"Matrix overflow",
        "4":"PPID not found",
        "5":"Mode unsupported",
        "6":"Command will be performed with completion signaled
later",
        ">":["6","Other error"]}},
"ACKC10":{
    "doc":["ACKC10, Acknowledge Code"],
    "formats":["B"],
    "bytes":1,
    "codes":{
        "0":"Accepted for display",
        "1":"Message will not be displayed",
        "2":"Terminal not available"}},
"ALCD":{
    "doc":["ALCD, Alarm Code Byte",
        "bit 8 = 1 means alarm set",
        "bit 8 = 0 means alarm cleared",
        "bit 7-1 is alarm category"],
    "formats":["B"],
    "bytes":1,
    "codes":{
        "0":"Not Used",
        "1":"Alarm Cleared, Personal Safety",
        "2":"Alarm Cleared, Equipment Safety",
        "3":"Alarm Cleared, Parameter Control Warning",
        "4":"Alarm Cleared, Parameter Control Error",
        "5":"Alarm Cleared, Irrecoverable Error",
        "6":"Alarm Cleared, Equipment Status Warning",
        "7":"Alarm Cleared, Attention flags",
        "8":"Alarm Cleared, Data integrity",
        "><":["8,128","Alarm Cleared, Other categories"],
        "128":"Not Used",
        "129":"Alarm Set, Personal Safety",
        "130":"Alarm Set, Equipment Safety",

```

```
        "131": "Alarm Set, Parameter Control Warning",
        "132": "Alarm Set, Parameter Control Error",
        "133": "Alarm Set, Irrecoverable Error",
        "134": "Alarm Set, Equipment Status Warning",
        "135": "Alarm Set, Attention flags",
        "136": "Alarm Set, Data integrity",
        ">": [136, "Alarm Cleared, Other categories"]}],
    "ALED": {
        "doc": ["ALED, Alarm Enable/Disable Code"],
        "formats": ["B"],
        "bytes": 1,
        "codes": {
            ">": [127, "Enable Alarm"],
            "<": [128, "Disable Alarm"]}
    },
    "ALID": {
        "doc": ["ALID, Alarm Identification"],
        "formats": ["U"]
    },
    "ALTX": {
        "doc": ["ALTX, Alarm Text"],
        "formats": ["A"],
        "max": 120
    },
    "ATTRDATA": {
        "doc": ["ATTRDATA, Contains a specific attribute value for a
specific object."],
        "formats":
[{"format": "L", "item": "ATTRDATA"}, "B", "Bool", "A", "F", "U", "I"]
    },
    "ATTRID": {
        "doc": ["ATTRID, Identifier for an attribute for a specific type of
object."],
        "formats": ["A", "U"]
    },
    "C": {
        "CAACK": {
            "doc": ["CAACK, Carrier Action Acknowledge Code"],
            "formats": ["U1"],
            "codes": {
                "0": "Acknowledge, command has been performed",
                "1": "Invalid command",
                "2": "Can not perform now",
                "3": "Invalid data or argument",
                "4": "Acknowledge, request will be performed with completion
signaled later by an event",
                "5": "Rejected. Invalid state",
                "6": "Command performed with errors"}
        },
        "CARRIERACTION": {
            "doc": ["CARRIERACTION, Specifies the action requested for a
carrier"],
            "formats": ["A"]
        },
        "CARRIERID": {
            "doc": ["CARRIERID, The identifier of a carrier"],
```

```
    "formats":["A"]},
  "CATTRDATA":{
    "doc":["CATTRDATA, The value of a carrier attribute"],
    "formats":
[{"format":"L","item":"CATTRDATA"},"B","Bool","A","F","U","I"]},
    "CATTRID":{
      "doc":["CATTRID, The name of a carrier attribute"],
      "formats":["A"]},
    "CCODE":{
      "doc":["CCODE, Command Code",
        "Each command code corresponds to a unique process operation
the machine is capable of performing."],
      "formats":["A","I2","I4","U2","U4"]},
    "CEED":{
      "doc":["CEED, Collection event or trace enable/disable code"],
      "formats":["Bool"],
      "bytes":1,
      "codes":{
        "0":"Disable",
        "1":"Enable"}}},
    "CEID":{
      "doc":["CEID, Collected Event ID"],
      "formats":["U"]},
    "COMMACK":{
      "doc":["COMMACK, Establish Communications Acknowledge Code"],
      "formats":["B"],
      "bytes":1,
      "codes":{
        "0":"Accepted",
        "1":"Denied, Try Again"}}},
    "CPACK":{
      "doc":["CPACK, Command Parameter Acknowledge Code"],
      "formats":["B"],
      "bytes":1,
      "codes":{
        "1":"Parameter Name (CPNAME) does not exist",
        "2":"Illegal Value specified for CPVAL",
        "3":"Illegal Format specified for CPVAL",
        ">":["3","Other equipment specific error"]}},
    "CPNAME":{
      "doc":["CPNAME, Command Parameter Name"],
      "formats":["A"],
      "max":40},
    "CPVAL":{
      "doc":["CPVAL, Command Parameter Value"],
      "formats":["A","B","Bool","U","I"]}},
  "D":{
    "DATAID":{
      "doc":["DATAID, Data ID"],
```

```
    "formats":["U"]},
    "DALENGTH":{
        "doc":["DALENGTH, Total bytes to be sent"],
        "formats":["U"]},
    "DRACK":{
        "doc":["DRACK, Define Report Acknowledge Code",
            "If an error condition is detected the entire message is
rejected (i.e., partial changes are not allowed)."],
        "formats":["B"],
        "bytes":1,
        "codes":{
            "0":"Accepted",
            "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"]}},
    "DSPER":{
        "doc":["DSPER, Data sample period.",
            "DSPER has two allowable formats:",
            "Format 1: hhmmss, 6 bytes",
            "Format 2: hhmmsscc, 8 bytes",
            "Where hh is hours, mm is minutes, ss is seconds, and cc is
centiseconds.",
            "Equipment shall either (1) support only Format 1, or (2)
support both Format 1 and Format 2.",
            "Equipment shall document which formats it accepts.",
            "Equipment which supports Format 2 need not necessarily
support a minimum DSPER of 1 centisecond,",
            "nor a trace resolution of 1 centisecond, but equipment
suppliers shall document its trace performance limits."],
        "formats":["A"],
        "pattern":"^[0-9]{6}|[0-9]{8}$"}},
    "E":{
        "EAC":{
            "doc":["EAC, Equipment Acknowledge Code"],
            "formats":["B"],
            "bytes":1,
            "codes":{
                "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"]}},
        "ECDEF":{
            "doc":["ECDEF, Equipment Constant Default value"],
            "formats":
[{"format":"L","item":"ECDEF"},"A","B","Bool","J","F","U","I"]},
        "ECID":{
```

```
    "doc":["ECID, Equipment Constant ID"],
    "formats":["U"]},
  "ECMAX":{
    "doc":["ECMAX, Equipment Constant Maximum value"],
    "formats":["A","B","Bool","J","F","U","I"]},
  "ECMIN":{
    "doc":["ECMIN, Equipment Constant Minimum value"],
    "formats":["A","B","Bool","J","F","U","I"]},
  "ECNAME":{
    "doc":["ECNAME, Equipment Constant Name"],
    "formats":["A"]},
  "ECV":{
    "doc":["ECV, Equipment Constant Value"],
    "formats":
[{"format":"L","item":"ECV"},"A","B","Bool","J","F","U","I"]},
  "EDID":{
    "doc":["EDID, Expected Data Identification"],
    "formats":["A","B","U","I"]},
  "ERACK":{
    "doc":["ERACK, Enable/Disable Event Acknowledge Code"],
    "formats":["B"],
    "bytes":1,
    "codes":{
      "0":"Accepted",
      "1":"Denied. At least one CEID does not exist",
      ">":["1","Other Errors"]}},
  "ERRCODE":{
    "doc":["ERRCODE, Code identifying an error"],
    "formats":["U"],
    "codes":{
      "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",
"48": "Load port does not exist",
"49": "Load port already in use",
"50": "Missing Carrier",
"32768": "Action will be performed at earliest opportunity",
"32769": "Action can not be performed now",
"32770": "Action failed due to errors",
"32771": "Invalid command",
"32772": "Client Alr",
"32773": "Duplicate ClientID",
"32774": "Invalid ClientType",
"32775": "IncompatibleVersions",
"32776": "Unrecognized ClientID (Client not currently
connected)",
"32777": "Failed (Completed Unsuccess-fully)",
"32778": "Failed (Unsafe) – External intervention required",
"32779": "Sensor-Detected Obstacle",
"32780": "Material Not Sent",
"32781": "Material Not Received",
"32782": "Material Lost",
```

```
        "32783": "Hardware Failure",
        "32784": "Transfer Cancelled" }},
    "ERRTEXT": {
        "doc": ["ERRTEXT, Text string describing the error noted in the
corresponding ERRCODE"],
        "formats": ["A"],
        "max": 120 }},
    "G": {
        "GRANT": {
            "doc": ["GRANT, Grant code"],
            "formats": ["B"],
            "bytes": 1,
            "codes": {
                "0": "Permission Granted",
                "1": "Busy, Try Again",
                "2": "No Space Available",
                "3": "Duplicate DATAID",
                ">": [3, "Equipment Specific Error Code"] }},
        "GRANT6": {
            "doc": ["GRANT6, Permission to send"],
            "formats": ["B"],
            "bytes": 1,
            "codes": {
                "0": "Permission granted",
                "1": "Busy, try again",
                "2": "Not interested",
                ">": [2, "Other err"] }},
    "H": {
        "HACK": {
            "doc": ["HACK, Host Command Parameter Acknowledge Code"],
            "formats": ["B"],
            "bytes": 1,
            "codes": {
                "0": "Acknowledge, command has been performed",
                "1": "Command does not exist",
                "2": "Cannot perform now",
                "3": "At least one 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" }},
    "L": {
        "LENGTH": {
            "doc": ["LENGTH, Length of the service program or process program
in bytes."],
            "formats": ["U"] },
        "LRACK": {
            "doc": ["LRACK, Link Report Acknowledge Code"],
            "formats": ["B"],
```

```
    "bytes":1,
    "codes":{
        "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"]}},
    "M":{
        "MDLN":{
            "doc":["MDLN, Equipment Model Type"],
            "formats":["A"],
            "max":20},
        "MEXP":{
            "doc":["MEXP, Message expected in the form SxxFyy where x is
stream and y is function."],
            "formats":["A"],
            "max":6},
        "MF":{
            "doc":["MF, Material Format code",
                "Items with format A will be a unit identifier for one of the
special SECS generic units, as specified in § 12."],
            "formats":["B","A"],
            "bytes":1,
            "codes":{
                "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":"Quantities in carriers",
                "14":"Quantities in substrates"}},
        "MHEAD":{
            "doc":["MHEAD, SECS message block header associated with message
block in error."],
            "formats":["B"]},
        "MID":{
            "doc":["MID, Material ID"],
            "formats":["B","A"],
            "max":80}},
    "O":{
```

```
"OBJACK":{
  "doc":["OBJACK, Acknowledge code"],
  "formats":["U1"],
  "codes":{
    "0":"Successful completion of requested data",
    "1":"Error"}},
"OBJSPEC":{
  "doc":["OBJSPEC, 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."],
  "formats":["A"]},
"OBJTYPE":{
  "doc":["OBJTYPE, Identifier for a group or class of objects",
  "All objects of the same type must have the same set of
attributes available."],
  "formats":["A","U"]},
"OFLACK":{
  "doc":["OFLACK, Acknowledge code for OFF-LINE request"],
  "formats":["B"],
  "bytes":1,
  "codes":{
    "0":"OFF-LINE Acknowledge"}},
"ONLACK":{
  "doc":["ONLACK, Acknowledge code for ON-LINE request"],
  "formats":["B"],
  "bytes":1,
  "codes":{
    "0":"ON-LINE Accepted",
    "1":"ON-LINE Not Allowed",
    "2":"Equipment Already ON-LINE"}}},
"P":{
  "PARAMNAME":{
    "doc":["PARAMNAME, The name of a parameter in a request"],
    "formats":["A"]},
  "PARAMVAL":{
    "doc":["PARAMVAL, The value of the parameter named in PARAMNAME",
    "Values that are lists are restricted to lists of single items
of the same format type."],
    "formats":
```

```
[{"format":"L","item":"PARAMVAL"},"A","B","Bool","F","U","I"]},
  "PORTACTION":{
    "doc":["PORTACTION, tThe action to be performed on a port"],
    "formats":["A"]},
  "PPARM":{
    "doc":["PPARM, Process Parameter"],
    "formats":["A","Bool","F","U","I"]},
  "PPBODY":{
    "doc":["PPBODY, 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."],
    "formats":["A","B","B64","U","I"]},
  "PPGNT":{
    "doc":["PPGNT, Process Program Grant status"],
    "formats":["B"],
    "bytes":1,
    "codes":{
      "0":"OK",
      "1":"Already have",
      "2":"No space",
      "3":"Invalid PPID",
      "4":"Busy, try later",
      "5":"Will not accept",
      ">":["5","Other error"]}},
  "PPID":{
    "doc":["PPID, Process Program ID",
    "The format used in the PPID will be host dependent.",
    "For internal use of the equipment, the PPID can be treated as
a unique B pattern.",
    "If the local equipment is not prepared to display the
transmitted code, the display should be in hexadecimal form."],
    "formats":["A"],
    "max":120},
  "PRJOBID":{
    "doc":["PRJOBID, Text string which uniquely identifies a process
job"],
    "formats":["A"]},
  "PRPAUSEEVENT":{
    "doc":["PRPAUSEEVENT, The list of event identifiers, which may be
sent as an attribute value to a process job.",
    "When a process job encounters one of these events it will
pause, until it receives the PRJobCommand RESUME."],
    "formats":[{"format":"L","item":"CEID"}]},
  "PRPROCESSSTART":{
    "doc":["PRPROCESSSTART, Indicates that the process resource start
processing immediately when ready."],
    "formats":["Bool"],
    "codes":{
      "0":"Manual Start",
```

```
        "1": "Automatic Start" }},
    "PRRECIPEMETHOD": {
        "doc": ["PRRECIPEMETHOD, Indicates the recipe specification type, whether tuning is applied and which method is used."],
        "formats": ["U1"],
        "codes": {
            "1": "Recipe only",
            "2": "Recipe with variable tuning" }},
    "PRSTATE": {
        "doc": ["PRSTATE, Enumerated value"],
        "formats": ["U1"]},
    "PTN": {
        "doc": ["PTN, Material Port number"],
        "formats": ["B", "U1"],
        "bytes": 1}},

    "R": {
        "RCMD": {
            "doc": ["RCMD, Remote Command"],
            "formats": ["A"]},
        "RCPPARNM": {
            "doc": ["RCPPARNM, The name of a recipe variable parameter."],
            "formats": ["A"],
            "max": 256},
        "RCPPARVAL": {
            "doc": ["RCPPARVAL, The initial setting assigned to a recipe variable parameter."],
            "formats":
[{"format": "L", "item": "RCPPARVAL"}, "A", "B", "Bool", "F", "U", "I"],
            "max": 80},
        "RCPSPEC": {
            "doc": ["RCPSEC, Recipe specifier. The object specifier of a recipe."],
            "formats": ["A"]},
        "REPGSZ": {
            "doc": ["REPGSZ, Reporting Group Size"],
            "formats": ["U"]},
        "RPTID": {
            "doc": ["RPTID, Report ID"],
            "formats": ["U"]}},

    "S": {
        "SHEAD": {
            "doc": ["SHEAD, Stored header related to the transaction timer"],
            "formats": ["B"]},
        "SLOTID": {
            "doc": ["SLOTID, Used to reference material by slot (a position that holds material/substrates) in a carrier.",
                "This item may be implemented as an array in some messages."],
            "formats": ["U1"]},
```

```
"SMPLN":{
  "doc":["SMPLN, Sample Number"],
  "formats":["U","I"]},
"SOFTREV":{
  "doc":["SOFTREV, Software Revision Code"],
  "formats":["A"],
  "max":20},
"STIME":{
  "doc":["STIME, Sample Time",
    "12 or 16 bytes, or Extended format as specified by the
TimeFormat equipment constant value setting.",
    "12-byte format: YYMMDDhhmmss",
    "16-byte format: YYYYMMDDhhmmsscc",
    "Extended (max 32 byte) format: format YYYY-MM-
DDThh:mm:ss.sTZD (see SEMI E148)"],
  "formats":["A"],
  "max":32},
"SV":{
  "doc":["SV, Status Variable"],
  "formats":
[{"format":"L","item":"SV"},"B","Bool","A","J","F","U","I"]},
"SVID":{
  "doc":["SVID, Status Variable ID",
    "Status variables may include any parameter that can be
sampled in time such as temperature or quantity."],
  "formats":["U"]},
"SVNAME":{
  "doc":["SVNAME, Status Variable Name"],
  "formats":["A"]},
"T":{
  "TEXT":{
    "doc":["TEXT, a single line of characters."],
    "formats":["A","B","Local","U","I"]},
  "TIACK":{
    "doc":["TIACK, Time Acknowledge Code"],
    "formats":["B"],
    "bytes":1,
    "codes":{
      "0":"OK",
      "1":"Error, not done"}},
  "TIAACK":{
    "doc":["TIAACK, Equipment Acknowledgement Code"],
    "formats":["B"],
    "bytes":1,
    "codes":{
      "0":"Everything correct",
      "1":"Too many SVIDs",
      "2":"No more traces allowed",
      "3":"Invalid period",
```

```

        "4": "Unknown SVID specified",
        "5": "Invalid REPGSZ",
        ">": [63, "Equipment specified error"]}},
    "TID": {
        "doc": ["TID, Terminal number"],
        "formats": ["B"],
        "bytes": 1,
        "codes": {
            "0": "Single or main terminal",
            ">": [0, "Additional terminals at the same equipment"]}},
    "TIME": {
        "doc": ["TIME, Time of day",
            "12, 16 bytes, or Extended format as specified by the
TimeFormat equipment constant value setting.",
            "12-byte format YYMMDDhhmmss",
            "16-byte format YYYYMMDDhhmmsscc",
            "Extended (max 32 byte) format YYYY-MM-DDThh:mm:ss.sTZD (see
SEMI E148)"],
        "formats": ["A"],
        "max": 32},
    "TOTSMP": {
        "doc": ["TOTSMP, Total Samples"],
        "formats": ["U"]},

    "TRID": {
        "doc": ["TRID, Trace Request ID."],
        "formats": ["U"]}},
    "U": {
        "UNITS": {
            "doc": ["UNITS, Units Identifier"],
            "formats": ["A", "B"]}},
    "V": {
        "V": {
            "doc": ["V, Variable data"],
            "formats":
[{"format": "L", "item": "V"}, "A", "B", "Bool", "J", "F", "U", "I"]},
        "VID": {
            "doc": ["VID, Variable ID"],
            "formats": ["U"]}}},
    "messages": {

        "defaults": {
            "doc": ["Default values for properties that are missing."],
            "block": "single",
            "reply": false,
            "direction": "h<->e",
            "body": null},

        "S1": {

```

```
    "doc":["Stream 1 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."],

    "S1F0":{
        "doc":["Abort Transaction",
            "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."],
        "block":"single","direction":"h<->e"},

    "S1F1":{
        "doc":["Are You There Request",
            "Establishes if the equipment is on-line.",
            "A function 0 response to this message 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."],
        "block":"single","direction":"h<->e","reply":true,
        "autoReply":{"header":
{"stream":1,"function":2,"reply":false},"body":[]}},

    "S1F2":{
        "doc":["On Line Data",
            "Data signifying that the equipment is alive.",
            "Exception: The host sends a zero-length list to the
equipment."],
        "block":"single",
        "or":[
            {"direction":"h<-e","body":["MDLN","SOFTREV"]},
            {"direction":"h->e","body":[]}]},

    "S1F3":{
        "doc":["Selected Equipment Status Request",
            "A request to the equipment to report the values of selected
status variables.",
            "Exception: A zero-length list means report all SVIDs."],
        "block":"single","direction":"h->e","reply":true,
        "body":["SVID","..."]},

    "S1F4":{
        "doc":["Selected Equipment Status Data",
            "The equipment reports the value of each status variable
requested in the order requested.",
            "The host remembers the names of values requested.",
            "Exception: A zero-length list item for SV means that SVID
does not exist."],
```

```

        "block": "multiple", "direction": "h<-e",
        "body": ["SV", "..."]},

    "S1F11": {
        "doc": ["Status Variable Namelist Request",
            "A request to the equipment to report the name and units of
selected status variables.",
            "Exception: A zero length means report all SVIDs."],
        "block": "single", "direction": "h->e", "reply": true,
        "body": ["SVID", "..."]},

    "S1F12": {
        "doc": ["Status Variable Namelist Reply",
            "The equipment reports to the host the name and units of the
requested status variables.",
            "Exception: Zero-length A items for both SVNAME and UNITS
indicates that the SVID does not exist."],
        "block": "multiple", "direction": "h<-e",
        "body": [{"SVID", "SVNAME", "UNITS"}, "..."]},

    "S1F13": {
        "doc": ["Establish Communications Request",
            "The purpose of this message is to establish communication
between an equipment and host at an application level.",
            "The purpose of this message is to provide a formal means of
initializing communications at a logical application level both on",
            "power-up and following a break in communications. It should
be the following any period where host and Equipment SECS",
            "applications are unable to communicate. An attempt to send an
Establish Communications Request (S1,F13) should be repeated",
            "at programmable intervals until an Establish Communications
Acknowledge (S1,F14) is received within the transaction timeout",
            "period with an acknowledgement code accepting the
establishment.",
            "Exception: The host sends a zero-length list to the
equipment."],
        "block": "single", "reply": true,
        "autoReply": {"header":
{"stream": 1, "function": 14, "reply": false}, "body": [{"format": "B", "value": 0}, []]},
        "or": [
            {"direction": "h->e", "body": []},
            {"direction": "h<-e", "body": ["MDLN", "SOFTREV"]}]}},

    "S1F14": {
        "doc": ["Establish Communications Request Acknowledge",
            "Accept or deny Establish Communications Request (S1F13).",
            "MDLN and SOFTREV are on-line data and are valid only if
COMMACK = 0.",
            "Exception: The host sends a zero-length list for item 2 to

```

```
the equipment."],
    "block": "single",
    "or": [
        {"direction": "h<-e", "body": ["COMMACK", ["MDLN", "SOFTREV"]]},
        {"direction": "h->e", "body": ["COMMACK", []]}],

    "S1F15": {
        "doc": ["Request OFF-LINE",
            "The host requests that the equipment transition to the OFF-
LINE state."],
        "block": "single", "direction": "h->e", "reply": true,

    "S1F16": {
        "doc": ["OFF-LINE Acknowledge"],
        "block": "single", "direction": "h<-e",
        "body": "OFLACK"},

    "S1F17": {
        "doc": ["Request ON-LINE",
            "The host requests that the equipment transition to the ON-
LINE state"],
        "block": "single", "direction": "h->e", "reply": true,

    "S1F18": {
        "doc": ["ON-LINE Acknowledge"],
        "block": "single", "direction": "h<-e",
        "body": "ONLACK"},

    "S1F23": {
        "doc": ["Collection Event Namelist Request",
            "This function allows the host to retrieve information about
what collection event IDs are available in the equipment",
            "and which data values (DVVALs) are valid for each collection
event.",
            "Exception: A zero-length list means send information for all
CEIDs."],
        "block": "single", "direction": "h->e", "reply": true,
        "body": ["CEID", "..."]},

    "S1F24": {
        "doc": ["Collection Event Namelist",
            "The equipment reports to the host the information of the
collection events and associated VIDs of the CEIDs",
            "requested with the S1F23 message.",
            "A listed VID can be conditionally or unconditionally
associated with the CEID;",
            "it is the responsibility of the equipment supplier to
document whether the conditional VIDs are reported with S1F24.",
            "Exception: When both CENAME and the list of associated VIDs
```

```
are zero-length items, this indicates that ECID does not exist."],
    "block":"multiple","direction":"h<-e",
    "body":[["CEID","CENAME",["VID","..."]],"..."]}],

    "S2":{
        "doc":["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 specifically",
            "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",
            "(Streams 10, 13). See also continuations in Stream 17."],

        "S2F0":{
            "doc":["Abort Transaction",
                "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."],
            "block":"single","direction":"h<->e"},

        "S2F13":{
            "doc":["Equipment Constant Request",
                "A request to the equipment to report the values of selected
constants.",
                "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.",
                "Exception: A zero-length list means report all ECVs according
to a predefined order."],
            "block":"single","direction":"h->e","reply":true,
            "body":["ECID","..."]},

        "S2F14":{
            "doc":["Equipment Constant Data",
                "The equipment reports the value of each constant requested in
the order requested.",
                "Exception: A zero-length list item for ECV means that ECID
does not exist. The list format for this data item is not allowed, except in this
case."],
            "block":"multiple","direction":"h<-e",
            "body":["ECV","..."]},

        "S2F15":{
            "doc":["New Equipment Constant Send",
                "Change one or more equipment constants."],
            "block":"single","direction":"h->e","reply":true,
            "body":[["ECID","ECV"],"..."]},
```

```
"S2F16":{
  "doc":["New Equipment Constant Acknowledge",
    "Acknowledges or denies modification of equipment constants.",
    "If EAC contains a non-zero error code, the equipment should
not change any of the ECIDs specified in S2F15."],
  "block":"single","direction":"h<-e",
  "body":"EAC"},

"S2F17":{
  "doc":["Date and Time Request",
    "Get the date and time from the equipment.",
    "Useful to check equipment time base or for equipment to
synchronize with the host time base."],
  "block":"single","direction":"h<->e","reply":true},

"S2F18":{
  "doc":["Date and Time Data",
    "Actual time data from equipment.",
    "Exception: A zero-length item means no time exists"],
  "block":"single","direction":"h<->e",
  "body":"TIME"},

"S2F23":{
  "doc":["Trace Initialize Send",
    "Report the current values of selected status variables at a
regular interval of time.",
    "Status variables exist at all times. This function provides a
way to sample a subset of those status variables as a function of time.",
    "The trace data is returned on S6,F1 and is related to the
original request by the TRID Multiple trace requests may be made to",
    "that equipment allowing it. If equipment receives S2F23 with
the same TRID as a trace function that is currently in progress,",
    "the equipment should terminate the old trace and then
initiate the new trace. A trace function currently in progress may be",
    "terminated by S2,F23 with TRID of that trace and TOTSMF =
0.",
    "If S2F23 is multi-block, it must be preceded by the
S2F39/S2F40 Inquire/Grant transaction. Some equipment may support",
    "only single-Block S6F1, and may refuse a S2F23 message which
would cause a multi-block S6F1.",
    "Each equipment shall document its trace performance limits.
The Host Computer shall not send an S2F23 which exceeds the",
    "equipment's performance limits, or the equipment may operate
incorrectly.",
    "TRID = Trace Request ID, DSPER = Data Sample Period, TOTSMF =
Total samples to be made.",
    "REPGSZ = Reporting Group Size, SVID = Status Variable ID"],
  "block":"multiple","direction":"h->e","reply":true,"CommonID":
```

```

["TRID"],
    "body":["TRID","DSPER","TOTSMP","REPGSZ",["SVID","..."]]},

"S2F24":{
    "doc":["Trace Initialize Acknowledge"],
    "block":"single","direction":"h<-e",
    "body":{
        "doc":["Equipment Acknowledgment Code"],
        "formats":["B"],
        "bytes":1,
        "codes":{
            "0":"Everything correct",
            "1":"Too many SVIDs",
            "2":"No more traces allowed",
            "3":"Invalid period",
            "4":"Unknown SVID specified",
            "5":"Invalid REPGSZ",
            ">":["63","Equipment specified error"]}}},

"S2F29":{
    "doc":["Equipment Constant Namelist Request",
        "A request to the equipment to report information about
selected constants.",
        "This function allows the host to retrieve basic information
about what equipment constants are available in the equipment.",
        "Exception: A zero-length list means send information for all
ECIDs."],
    "block":"single","direction":"h->e","reply":true,
    "body":["ECID","..."]]},

"S2F30":{
    "doc":["Equipment Constant Namelist",
        "The equipment reports to the host information about certain
constants.",
        "Exception: Zero-length A items for ECNAME, ECMIN, ECMAX,
ECDEF, and UNITS indicates that the ECID does not exist."],
    "block":"multiple","direction":"h<-e",
    "body":["ECID","ECNAME","ECMIN","ECMAX","ECDEF","UNITS"],"..."]]},

"S2F31":{
    "doc":["Date and Time Set Request",
        "Set the date and time in the equipment.",
        "Useful to synchronize the equipment time with the host time
base."],
    "block":"single","direction":"h->e","reply":true,
    "body":"TIME"},

"S2F32":{
    "doc":["Date and Time Set Acknowledge"],

```

```
"block":"single","direction":"h<-e",
"body":"TIACK"},

"S2F33":{
  "doc":["Define Report",
    "The purpose of this message is for the host to define a group
of reports for the equipment.",
    "The type of report to be transmitted is designated by a
BOOLEAN \"Equipment Constant\".",
    "An \"Equipment Constant Value\" of \"False\" means that an
\"Event Report\" (S6F11) will be sent, and a value of \"True\" means",
    "that an \"Annotated Event Report\" (S6F13) will be sent. If
S2F33 is Multi-block, it must be preceded by the S2F39/S2F40 Inquire/Grant
transaction.",
    "Exception 1: A list of zero-length following DATAID deletes
all report definitions and associated links. See S2F35 (Link Event/Report).",
    "#Exception 2: A list of zero-length following RPTID deletes
report type RPTID. All CEID links to this RPTID are also deleted."],
  "block":"multiple","direction":"h->e","reply":true,
  "body":["DATAID",[["RPTID",["VID","..."]],"..."]]},

"S2F34":{
  "doc":["Define Report Acknowledge"],
  "block":"single","direction":"h<-e",
  "body":"DRACK"},

"S2F35":{
  "doc":["Link Event Report",
    "The purpose of this message is for the host to link n reports
to an event (CEID).",
    "These linked event reports will default to 'disabled' upon
linking.",
    "That is, the occurrence of an event would not cause the
report to be sent until enabled.",
    "See S2F37 for enabling reports.",
    "If S2F35 is Multi-block, it must be preceded by the
S2F39/S2F40 Inquire/Grant transaction.",
    "#Exception: A list of zero length following CEID deletes all
report links to that event."],
  "block":"multiple","direction":"h->e","reply":true,
  "body":["DATAID",[["CEID",["RPTID","..."]],"..."]]},

"S2F36":{
  "doc":["Link Event Report Acknowledge",
    "If an error condition is detected the entire message is
rejected (i.e., partial changes are not allowed)."],
  "block":"single","direction":"h<-e",
  "body":"LRACK"},
```

```
"S2F37":{
  "doc":["Enable/Disable Event Report",
    "The purpose of this message is for the host to enable or
disable reporting for a group of events (CEIDs).",
    "The reporting for unlisted (CEIDs) is not affected.",
    "Exception: A list of zero length following <CEED> means all
CEIDs."],
  "block":"single","direction":"h->e","reply":true,
  "body":["CEED",["CEID","..."]]},

"S2F38":{
  "doc":["Enable/Disable Event Report Acknowledge",
    "If an error condition is detected the entire message is
rejected, i.e., partial changes are not allowed."],
  "block":"single","direction":"h<-e",
  "body":["ERACK"]},

"S2F39":{
  "doc":["Multi-block Inquire",
    "If a S2,F23, S2,F33, S2,F35, S2,F45, or S2,F49 message is
more than one block, this transaction must precede the message."],
  "block":"single","direction":"h->e","reply":true,
  "body":["DATAID","DATALENGTH"]]},

"S2F40":{
  "doc":["Multi-block Grant",
    "Grant permission to send multi-block message."],
  "block":"single","direction":"h<-e",
  "body":["GRANT"]},

"S2F41":{
  "doc":["Host Command Send",
    "The Host requests the Equipment perform the specified remote
command with the associated parameters."],
  "block":"single","direction":"h->e","reply":true,
  "body":["RCMD",[["CPNAME","CPVAL"],"..."]]},

"S2F42":{
  "doc":["Host Command Acknowledge",
    "If command is not accepted due to one or more invalid
parameters (i.e., HCAK = 3),",
    "then a list of invalid parameters will be returned containing
the parameter name and reason for being invalid."],
  "block":"single","direction":"h<-e",
  "body":["HCAK",[["CNAME","CPACK"],"..."]]}},

"S3":{
  "doc":["Stream 3 Materials Status",
    "The functions of the material status stream are used to
```

```
communicate information and actions related to material",
    "including carriers and material-in-process, time-to-completion
information, and extraordinary material occurrences."],

    "S3F0":{
        "doc":["Abort Transaction",
            "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."],
        "block":"single","direction":"h<->e"},

    "S3F15":{
        "doc":["Materials Multi-Block Inquire",
            "This message requests permission to send a multi-block
message based upon a maximum length of the total message.",
            "It must be sent prior to sending any multi-block primary
message in Stream 3."],
        "block":"single","direction":"h->e","reply":true,
        "body":["DATAID","DATALENGTH"]},

    "S3F16":{
        "doc":["Materials Multi-Block Grant",
            "This message grants or denies permission to send a multi-
block primary message in Stream 3."],
        "block":"single","direction":"h<-e",
        "body":"GRANT"},

    "S3F17":{
        "doc":["Carrier Action Request",
            "This message requests an action to be performed for a
specified carrier.",
            "If multi-block, this message must be preceded by the
S3F15/F16 transaction.",
            "Exception: If variable list has no items, then no carrier
attributes are included.",
            "If CARRIERID is not a zero-length item, then PTN may be
omitted (a zero-length item)",
            "ATTRID and ATTRDATA may be substituted for CATTRID and
CATTRDATA respectively.",
            "ReticlePodLocationID may be used as one of <CATTRID> when the
CARRIERACTION is PodRelease and the carrier is not at a Load Port."],
        "block":"multiple","direction":"h->e","reply":true,
        "body":["DATAID","CARRIERACTION","CARRIERID","PTN",
[[["CATTRID","CATTRDATA"],"..."]]],

    "S3F18":{
        "doc":["Carrier Action Acknowledge",
            "This message acknowledges the carrier action request.",
            "Exception: If variable list contains no items then no errors
```

```
exist"],
    "block":"single","direction":"h<-e",
    "body":["CAACK",[["ERRCODE","ERRTEXT"],"..."]]},

    "S3F25":{
        "doc":["Port Action Request",
            "This message requests an action be performed for a port.",
            "Exception: If variable list contains no items, then no
parameters are provided."],
        "block":"single","direction":"h->e","reply":true,
        "body":["PORTACTION","PTN",[["PARAMNAME","PARAMVAL"],"..."]]},

    "S3F26":{
        "doc":["Port Action Acknowledge",
            "This message acknowledges the port action request.",
            "Exception: If variable list contains no item then no errors
exist."],
        "block":"single","direction":"h<-e",
        "body":["CAACK",[["ERRCODE","ERRTEXT"],"..."]]},

    "S3F27":{
        "doc":["Change Access",
            "The Host requests the Equipment to change the Access Mode for
the specified Load Ports.",
            "ACCESSMODE specifies the desired Access Mode. PTN specifies a
desired Load Port Number.",
            "If variable list contains no items then the command applies
to all Load Ports on the equipment.",
            "If any specified port is already in the specified Access
Mode, then the Equipment shall accept the command,",
            "and toggle all loadports to specified mode.",
            "If the Equipment is unable to change one or more of specified
Port(s) to the specified Access Mode,",
            "then the Equipment shall accept the command (with appropriate
response acknowledgement)",
            "and shall change only the Access Mode of those Port(s)
allowed by the equipment,",
            "supplying the host with an indication that not all ports were
successfully changed."],
        "block":"single","direction":"h->e","reply":true,
        "body":["ACCESSMODE",[["PTN"],"..."]]},

    "S3F28":{
        "doc":["Change Access Acknowledge",
            "If the command is successful, CAACK = 0, and variable list
size = 0.",
            "If the command was successful for some ports, CAACK = 6, and
variable list size > 0."],
        "block":"single","direction":"h<-e",
```

```
    "body":["CAACK",[[{"PTN","ERRCODE","ERRTEXT"},"..."]]]},

    "S5":{
        "doc":["Stream 5 Exception Handling",
            "This stream contains messages regarding binary and analog
equipment exceptions.",
            "Exceptions are classified into two categories: errors and
alarms."],

        "S5F0":{
            "doc":["Abort Transaction",
                "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."],
            "block":"single","direction":"h<->e"},

        "S5F1":{
            "doc":["Alarm Report Send",
                "This message reports a change in or presence of an alarm
condition.",
                "One message will be issued when the alarm is set and one
message will be issued when the alarm is cleared.",
                "Irrecoverable errors and attention flags may not have a
corresponding clear message."],
            "block":"single","direction":"h<-e","reply":true,
            "autoReply":{"header":
{"stream":5,"function":2,"reply":false},"body":{"format":"B","value":0}},
            "body":["ALCD","ALID","ALTX"]},

        "S5F2":{
            "doc":["Alarm Report Acknowledge",
                "Acknowledge or error."],
            "block":"single","direction":"h->e",
            "body":["ACKC5"]},

        "S5F3":{
            "doc":["Enable/Disable Alarm Send",
                "This message will change the state of the enable bit in the
equipment.",
                "The enable bit determines if the alarm will be sent to the
host.",
                "Alarms which are not controllable in this way are unaffected
by this message.",
                "Exception: A zero-length item for ALID means all alarms."],
            "block":"single","direction":"h->e","reply":true,
            "body":["ALED","ALID"]},

        "S5F4":{
            "doc":["Enable/Disable Alarm Acknowledge",
```

```
        "Acknowledge or error."],
        "block":"single","direction":"h<-e",
        "body":"ACKC5"}},

    "S5F5":{
        "doc":["List Alarms Request",
            "This message requests the equipment to send binary and analog
alarm information to the host.",
            "A zero-length item means send all possible alarms regardless
of the state of ALED."],
        "block":"single","direction":"h->e","reply":true,
        "body":"ALID..."}},

    "S5F6":{
        "doc":["List Alarm Data",
            "This message contains the alarm data known to the
equipment.",
            "A zero-length item returned for ALCD or ALTX means that value
does not exist."],
        "block":"multiple","direction":"h<-e",
        "body":[["ALCD","ALID","ALTX"],"..."]}},

    "S6":{
        "doc":["Stream 6 Data Collection",
            "This stream is intended to cover the needs of in-process
measurements and equipment monitoring."],

        "S6F0":{
            "doc":["Abort Transaction",
                "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."],
            "block":"single","direction":"h<->e"},

        "S6F1":{
            "doc":["Trace Data Send",
                "This function sends samples to the host according to the
trace setup done by S2F23.",
                "Trace is a time-driven form of equipment status.",
                "Even if S6F1 is multi-block, it is not preceded by an
Inquire/Grant transaction, because the Host S2F23 is an implicit grant.",
                "Some equipment may support only single-block S6F1, and may
refuse an S2F23 (Trace Initiate Send) message which would cause a multi-block
S6,F1.",
                "Exception: A zero-length STIME means no value is given and
that the time is to be derived from SMPLN along with knowledge of the request.",
                "TRID = Trace Request ID, SMPLN = Sample Number, STIME =
Sample Time, SV = Status Variable Value"],
            "block":"multiple","direction":"h<-e","or":[{"reply":true},
```

```
{"reply":false}],
    "CommonID":["TRID"],"realtime":true,"SQLTags":true,
    "autoReply":{"header":
{"stream":6,"function":2,"reply":false},"body":{"format":"B","value":0}},
    "body":["TRID","SMPLN","STIME",["SV","..."]]},

    "S6F2":{
        "doc":["Trace Data Acknowledge",
            "Acknowledge or error."],
        "block":"single","direction":"h->e",
        "body":"ACKC6"},

    "S6F5":{
        "doc":["Multi-block Data Send Inquire",
            "If the discrete data report S6F3, F9, F11, F13 can involve
more than one block, this transaction must precede the transmission."],
        "block":"single","direction":"h<-e","reply":true,
        "autoReply":{"header":
{"stream":6,"function":6,"reply":false},"body":{"format":"B","value":0}},
        "body":["DATAID","DATALENGTH"]]},

    "S6F6":{
        "doc":["Multi-block Grant"],
        "block":"single","direction":"h->e",
        "body":"GRANT6"},

    "S6F11":{
        "doc":["Event Report Send",
            "Sends a report when an event occurs.",
            "The purpose of this message is for the equipment to send a
defined, linked, and enabled group of reports to the host upon the",
            "occurrence of an event (CEID).",
            "If S6F11 is Multi-block, it must be preceded by the
S6,F5/S6,F6 Inquire/Grant transaction.",
            "Exception: If there are no reports linked to the event a
'null' report is assumed.",
            "A zero-length list for number of reports means there are no
reports linked to the given CEID."],
        "block":"multiple","direction":"h<-e","reply":true,
        "CommonID":["DATAID"],"realtime":true,"SQLTags":["DATAID","CEID"],
        "autoReply":{"header":
{"stream":6,"function":12,"reply":false},"body":{"format":"B","value":0}},
        "body":["DATAID","CEID",[[["RPTID",["V","..."]],"..."]]},

    "S6F12":{
        "doc":["Event Report Acknowledge"],
        "block":"single","direction":"h->e",
        "body":"ACKC6"},

```

```
"S6F15":{
  "doc":["Event Report Request",
    "The purpose of this message is for the host to demand a given
report group from the equipment."],
  "block":"single","direction":"h->e","reply":true,
  "body":"CEID"},

"S6F16":{
  "doc":["Event Report Data",
    "Equipment sends reports linked to given CEID to host.",
    "Identical to structure of S6F11.",
    "A zero-length list for number of reports means there are no
reports linked to the given CEID."],
  "block":"multiple","direction":"h<-e",
  "body":["DATAID","CEID",[["RPTID",["V","..."]],"..."]]},

"S6F19":{
  "doc":["Individual Report Request",
    "The purpose of this message is for the host to request a
defined report from the equipment."],
  "block":"single","direction":"h->e","reply":true,
  "body":"RPTID"},

"S6F20":{
  "doc":["Individual Report Data",
    "Equipment sends variable data defined for the given RPTID to
the host."],
  "block":"multiple","direction":"h<-e",
  "body":["V","..."]]},

"S7":{
  "doc":["Stream 7 Process Program Management",
    "The functions in this stream are used to manage and transfer
process programs.",
    "Process programs are the equipment-specific descriptions that
determine the procedure to be",
    "conducted on the material by a single piece of equipment.",
    "Methods are provided to transfer programs as well as establish
the link between the process program",
    "and the material to be processed with that program."],

  "S7F0":{
    "doc":["Abort Transaction"],
    "block":"single","direction":"h<->e"},

  "S7F1":{
    "doc":["Process Program Load Inquire",
      "This message is used to initiate the transfer of a process
program or to select from stored programs.",
      "The message may be used to initiate the transfer of an
```

```
unformatted process program (S7F3/S7F4)",
    "or a formatted process program (S7F23/S7F24),
(S7F31/S7F32)."],
    "block": "single", "direction": "h<->e", "reply": true,
    "body": ["PPID", "LENGTH"]},

    "S7F2": {
        "doc": ["Process Program Load Grant",
            "This message gives permission for the process program to be
loaded."],
        "block": "single", "directions": "h<->e",
        "body": "PPGNT"},

    "S7F3": {
        "doc": ["Process Program Send",
            "The program is sent.",
            "If S7F3 is multi-block, it must be preceded by the S7F1/S7F2
Inquire/Grant transaction."],
        "block": "multiple", "direction": "h<->e", "reply": true,
        "body": ["PPID", "PPBODY"]},

    "S7F4": {
        "doc": ["Process Program Acknowledge"],
        "block": "single", "direction": "h<->e",
        "body": "ACKC7"},

    "S7F5": {
        "doc": ["Process Program Request",
            "This message is used to request the transfer of a process
program."],
        "block": "single", "direction": "h<->e", "reply": true,
        "body": "PPID"},

    "S7F6": {
        "doc": ["Process Program Data",
            "Exception: A zero-length list means request denied."],
        "block": "multiple", "direction": "h<->e",
        "body": {"or": [{"PPID", "PPBODY"}, []]}},

    "S7F17": {
        "doc": ["Delete Process Program Send",
            "This message is used by the host to request the equipment to
delete process programs from equipment storage.",
            "Exception: If variable list size = 0, delete all."],
        "block": "single", "direction": "h->e", "reply": true,
        "body": ["PPID", "..."]},

    "S7F18": {
        "doc": ["Delete Process Program Acknowledge"],
```

```
        "block": "single", "direction": "h<-e",
        "body": "ACKC7"},

    "S7F19": {
        "doc": ["Current EPPD Request",
            "This message is used to request the transmission of the
current equipment process program directory (EPPD).",
            "This is a list of all the PPIDs of the process programs
stored in the equipment."],
        "block": "single", "direction": "h->e", "reply": true},

    "S7F20": {
        "doc": ["Current EPPD Data",
            "This message is used to transmit the current EPPD."],
        "block": "multiple", "direction": "h<-e",
        "body": ["PPID", "..."]},

    "S7F23": {
        "doc": ["Formatted Process Program Send",
            "This message allows movement of formatted process programs
between a piece of equipment and its host system.",
            "The values of MDLN and SOFTREV are obtained from the PCD used
to generate the process program.",
            "If S7F23 is multi-block, it must be preceded by the S7F1/F2
Inquire/Grant transaction."],
        "block": "multiple", "direction": "h<->e", "reply": true,
        "body": ["PPID", "MDLN", "SOFTREV", [{"CCODE",
["PPARM", "..."]}], "..."]]},

    "S7F24": {
        "doc": ["Formatted Process Program Acknowledge",
            "Acknowledges reception of a formatted process program at its
destination and whether the process program",
            "was accepted by the interpreter.",
            "A returned status of \"accepted\" by the interpreter means
only that the message is understood.",
            "The validity of the contents of the process program is
determined through a separate transaction (S7F27/S7F28)."],
        "block": "single", "direction": "h<->e",
        "body": "ACKC7"},

    "S7F25": {
        "doc": ["Formatted Process Program Request",
            "This message is used by either equipment or host to request a
particular process program from the other."],
        "block": "single", "direction": "h<->e", "reply": true,
        "body": "PPID"},

    "S7F26": {
```

```
        "doc":["Formatted Process Program Data",
            "This message transfers a process program in response to a
request for the PPID.",
            "The values of MDLN and SOFTREV are obtained from the PCD used
to generate the process program.",
            "Exception: A zero length list indicates the request was
denied."],
        "block":"multiple","direction":"h<->e",
        "body":["PPID","MDLN","SOFTREV",["CCODE",
["PPARM","..."]],"..."]]},

    "S9":{
        "doc":["Stream 9 System Errors",
            "This stream provides a method of informing the host that a
message block has been received",
            "which cannot be handled or that a timeout on a transaction
(receive) timer has occurred.",
            "The messages indicate either a Message Fault or a Communications
Fault has occurred but",
            "do not indicate a Communications Failure has occurred."],

        "S9F0":{
            "doc":["Abort Transaction",
                "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."],
            "block":"single","direction":"h<->e"},

        "S9F1":{
            "doc":["Unrecognized Device ID",
                "The device ID in the message header did not correspond to any
known device ID in the node detecting the error."],
            "block":"single","direction":"h<-e",
            "body":"MHEAD"},

        "S9F3":{
            "doc":["Unrecognized Stream Type",
                "The equipment does not recognize the stream in the message
header."],
            "block":"single","direction":"h<-e",
            "body":"MHEAD"},

        "S9F5":{
            "doc":["Unrecognized Function Type",
                "The equipment does not recognize the function in the message
header."],
            "block":"single","direction":"h<-e",
            "body":"MHEAD"},
    }
```

```
"S9F7":{
  "doc":["Illegal Data",
    "This message indicates that the stream and function were
recognized, but the message format was incorrect."],
  "block":"single","direction":"h<-e",
  "body":"MHEAD"},

"S9F9":{
  "doc":["Transaction Timer Timeout",
    "This message indicates that the host failed to respond to a
request within the expected amount of time. The transaction has been aborted.",
    "It is up to the host to respond to this error in an
appropriate manner to keep the system operational."],
  "block":"single","direction":"h<-e",
  "body":"SHEAD"},

"S9F11":{
  "doc":["Data Too Long",
    "This message to the host indicates that the equipment has
been sent more data than it can handle."],
  "block":"single","direction":"h<-e",
  "body":"MHEAD"},

"S9F13":{
  "doc":["Conversation Timeout",
    "Data were expected but none were received within a reasonable
length of time. Resources have been cleared."],
  "block":"single","direction":"h<-e",
  "body":["MEXP","EDID"]}},

"S10":{
  "doc":["Stream 10 Terminal Services",
    "The functions of this stream is to pass textual messages
between operator terminals attached to",
    "processing and/or testing equipment and the host. The
equipment makes no attempt to interpret",
    "the text of the message, but merely passes it from terminal
keyboard to the host or from the host to the display of the terminal.",
    "Management of human response times to information displayed
on terminals is the responsibility of the host."],

  "S10F0":{
    "doc":["Abort Transaction",
      "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."],
    "block":"single","direction":"h<->e"},

  "S10F3":{
```

```
"doc":["Terminal Display, Single"],
"block":"single","direction":"h->e","reply":true,
"body":["TID","TEXT"]},

"S10F4":{
  "doc":["Terminal Display, Single Acknowledge"],
  "block":"single","direction":"h<-e",
  "body":["ACKC10"]},

"S14":{
  "doc":["Stream 14 Object Services",
    "The functions in this stream are used for generic functions
concerning objects,",
    "including obtaining information about objects and setting values
for an object."],

  "S14F0":{
    "doc":["Abort Transaction",
      "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."],
    "block":"single","direction":"h<->e"},

    "S14F9":{
      "doc":["Create Object Request",
        "This message is used to request an object owner to create an
object instance.",
        "OBJSPEC specifies the object owner.",
        "Exception: If OBJSPEC is a null-length item, no object
specifier is provided.",
        "If variable list size = 0, no specific attribute settings are
requested for the new object."],
      "block":"multiple","direction":"h<->e","reply":true,
      "body":["OBJSPEC","OBJTYPE",["ATTRID","ATTRDATA"],"..."]},

      "S14F10":{
        "doc":["Create Object Acknowledge",
          "This message is used to acknowledge the success or failure of
creating the new object specified.",
          "If successful, OBJSPEC is the object specifier of the new
object.",
          "The list of attributes returned is dependent upon the type of
object specified.",
          "Exception: If OBJSPEC is a null-length item, no object was
created.",
          "If the attribute list size = 0, then no attributes of the new
```

```

object are returned.",
    "If the errors list size = 0, no errors were detected."],
    "block":"multiple","direction":"h<->e",
    "body":["OBSPEC",
        [["ATTRID","ATTRDATA"],"..."],
        ["OBJACK",[["ERRCODE","ERRTEXT"],"..."]]]}],

"S16":{
    "doc":["Stream 16 Processing Management",
        "This stream provides protocol for a set of messages that enable
the control of material processing at equipment and equipment resources."],

    "S16F0":{
        "doc":["Abort Transaction",
            "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."],
        "block":"single","direction":"h<->e"},

    "S16F1":{
        "doc":["Multi-block Process Job Data Inquire",
            "If any of Processing Management messages are larger than one
block, then this transaction must precede that message."],
        "block":"single","direction":"h->e","reply":true,
        "body":["DATAID","DATALENGTH"]},

    "S16F2":{
        "doc":["Multi-block Process Job Data Grant",
            "Message to indicate if permission is granted to transmit a
multi-block Job Data message."],
        "block":"single","direction":"h<-e",
        "body":["GRANT"]},

    "S16F15":{
        "doc":["PRJobMultiCreate",
            "Use this single message to Create Multiple Process Jobs, each
of which may be unique in its association of material to process recipe.",
            "If multi-block, this message must be preceded by the S16F1/F2
transaction.",
            "Exception: The list for specifying material is empty when no
material is specified for the process job.",
            "The form of data for item 3 depends on the value in MF."],
        "block":"multiple","direction":"h->e","reply":true,
        "body":["DATAID",[["PRJOBID","MF",{ "or":[[["CARRIERID",
["SLOTID","..."]],"..."],
["MID","..."]]]],
["PRRECIPEMETHOD","RCPSPEC",
[["RCPPARNM","RCPPARVAL"],"..."]],["PRPROCESSSTART","PRPAUSEEVENT"]]]}],

```

```

    "S16F16":{
        "doc":["PRJobMultiCreate Acknowledge",
            "This message acknowledges the request and reports any errors
in the creation of a process job.",
            "ERRTEXT contains the identifier of process jobs that were not
created.",
            "Exception: If the list of errors is empty then no errors
exist."],
        "block":"single","direction":"h<-e",
        "body":[["PRJOBID","..."],["ACKA",
[["ERRCODE","ERRTEXT"],"..."]]]},

    "S16F17":{
        "doc":["RJobDequeue",
            "Used to remove process jobs from the equipment for jobs that
have not begun processing.",
            "Exception: If the list size = 0, then de-queue all."],
        "block":"single","direction":"h->e","reply":true,
        "body":["PRJOBID","..."]},

    "S16F18":{
        "doc":["PRJobDequeue Acknowledge",
            "Acknowledge the request to de-queue and report any errors.",
            "ERRTEXT will contain the identifier of any jobs that were not
de-queued.",
            "Exception: If error list size = 0, no errors exist."],
        "block":"single","direction":"h<-e",
        "body":[["PRJOBID","..."],["ACKA",
[["ERRCODE","ERRTEXT"],"..."]]]},

    "S16F19":{
        "doc":["PRGetAllJobs",
            "Requests the equipment to return a list of process jobs which
have not completed.",
            "They may be running or waiting to run."],
        "block":"single","direction":"h->e","reply":true},

    "S16F20":{
        "doc":["PRGetAllJobs Send",
            "Returns the requested list of process jobs.",
            "Exception: If the list of jobs = 0, then no process jobs are
running or waiting to run."],
        "block":"single","direction":"h<-e",
        "body":[["PRJOBID","PRSTATE"],"..."]}]
}
}

```

SECS/GEM Messages

SECS Message Basics

The SECS-II standard defines how information is exchanged between equipment and a host using messages. Messages are organized into categories called **streams**, which contain specific messages called **functions**. Each SECS message is identified by a stream and function number, written as **S#F#**. For example, **S1F13** refers to Stream 1, Function 13.

Requests and Responses

SECS messages are exchanged as requests and responses:

- **Request** (primary message): Originates from the host, which in Ignition is the Gateway. Request messages use odd-numbered functions, such as **S1F1**.
- **Response** (secondary message): Originates from the equipment in reply to a request. Response messages use even-numbered functions, such as **S1F2**.

A request may expect a single response or no response, depending on the message definition. This request-and-response pattern forms the basis of SECS/GEM communication.

SECS Message Format

SECS messages are transmitted using binary SECS protocols. Within Ignition, messages are represented as structured JSON objects for scripting and configuration purposes. Each message includes a header that specifies the stream, function, and whether a response is expected. Messages may also include a body containing one or more data items, which can be nested lists.

SECS Message Example

The following example shows an **S1F11** request represented as JSON within Ignition:

S1F11 Request Message

```
{
  "header":{
    "doc":"Status Variable Namelist Request",
    "stream":1,
    "function":11,
    "reply":true
  },
  "body":[
    {
      "doc":"SVID, Status Variable ID",
      "format":"U4",
      "value":4
    }
  ]
}
```

To send this request from a script, use the [sendRequest](#) function. This function returns a transaction ID, which is used to retrieve the corresponding response.

Python - Send an S1F11 request to the Equipment

```
body = [{"format": "U4", "value": 4}]
transactionID = system.secsgem.sendRequest("S1F11", True, body, "Equip4")
```

A response to the above request would look similar to the following:

S1F12 Response Message

```

{
  "header": {
    "doc": "Status Variable Namelist Reply",
    "stream": 1,
    "function": 12,
    "reply": false
  },
  "body": [
    [
      {
        "doc": "SVID, Status Variable ID",
        "format": "U4",
        "value": 4
      },
      {
        "doc": "SVNAME, Status Variable Name",
        "format": "A",
        "value": "RandomBinaryData"
      },
      {
        "doc": "UNITS, Units Identifier",
        "format": "A",
        "value": "B"
      }
    ]
  ]
}

```

Responses are retrieved using the [getResponse](#) function.

Python - Get an S1F12 response from the Equipment

```

response = system.secsgem.getResponse(transactionID, "Equip4")
print response

```

This request-and-response pattern forms the foundation of SECS/GEM communication. In addition to retrieving responses through scripting, messages can also be processed using tags or Message Handlers.

NOTE

To send or receive a specific SECS message, the corresponding stream and function must be defined in the equipment connection's SDL file.

SECS Messages through Tags

SECS messages that contain data can be configured to generate tags. This is commonly used for recurring data, such as trace data returned in **S6F1** messages. When enabled, tags are created automatically under the SECS/GEM Realtime Tag Provider for the equipment connection. These tags are read-only and do not support Tag History, alarming, or manual editing.

NOTE

Realtime tags are not persistent. If the Gateway restarts, the tags must be recreated.

To enable tag creation for a message, the message definition in the SDL file must be configured accordingly. The default SDL file is preconfigured to generate tags for **S6F1** messages.

Simulator S2F23 Body

```
[
  {
    "doc": "TRID, Trace Request ID.",
    "format": "U4",
    "value": 1
  },
  {
    "doc": "DSPEP, Data sample period.",
    "format": "A",
    "value": "000002"
  },
  {
    "doc": "TOTSMP, Total Samples",
    "format": "U4",
    "value": 10
  },
  {
    "doc": "REPGSZ, Reporting Group Size",
    "format": "U4",
    "value": 1
  },
  [
    {
      "doc": "SVID Status Variable ID",
      "value": 1,
      "format": "U4"
    },
    {
      "doc": "SVID Status Variable ID",
      "value": 2,
      "format": "U4"
    },
    {
      "doc": "SVID Status Variable ID",
      "value": 3,
      "format": "U4"
    }
  ]
]
```

Tags Example Using the Simulator

This example configures the **S2F14** message to create tags that can be viewed in the Tag Provider. It assumes the equipment connection is connected to a simulator and is using the default SDL file. To capture other messages as tags, the SDL file must be configured to allow tag creation. In this example, the SDL file is updated so that **S2F14** generates tags.

1. On the Gateway, go to the **Connections** section and navigate to **SECS/GEM > Equipment**.
2. Expand the **three dots menu** for the equipment connection and select **Edit File** under **SDL File**.
3. Locate the **S2F14** message in the list.
 - i. Add **SQLTags** as an Auto type object with a value of **true** directly under the S2F14 root.
 - ii. Create an array called **CommonID** directly under the S2F14 root.
 - a. Add **0: ECV** to the **CommonID** array as an Auto type object.
4. Save the SDL file and restart the equipment connection.
5. After the equipment connection is reestablished, send an **S2F13** message with the body shown below:

S2F13 Body

```
[
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 200
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 201
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 203
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 205
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 206
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 207
  }
]
```

SECS Messages through Message Handlers

Instead of generating tags, SECS messages can invoke a custom Python script using a Message Handler. The Message Handler must have the `onSecsGemRealtimeUpdate` name, though it can be specified for both the Client Event Scripts and the Gateway Event Scripts. Within the Message Handler, the SECS message will pass information into the payload object.

MESSAGE HANDLER LIMITATIONS

Message Handlers can only process messages that are allowed to originate from equipment, such as **S6F1**. Messages that originate only from the host cannot trigger Message Handlers.

Payload Key	Description
CommonID	The identifier used to associate related messages.
Equipment	The name of the equipment connection.
Direction	Indicates whether the message was sent or received.
RequestResponse	Indicates whether the message is a request or response.
TxID	The transaction identifier.
Reply	Indicates whether the message expects a reply.
Message	The message contents represented as JSON.

This script can handle the message in any way such as storing data in the database or writing to tags. The example below will take the information and create a logger object with it that will then be printed to the console.

Python - Example of a onSecsGemRealtimeUpdate Message Handler

This message handler is very simple in that it will create a logger with the SECS message data that prints to the console.

```
commonID = payload['CommonID']
equipment= payload['Equipment']
direction = payload['Direction']
reqResp = payload['RequestResponse']
txId = payload['TxID']
reply = payload['Reply']
message = payload['Message']

msg = "CommonID=" + commonID
msg += ", Equipment=" + equipment
msg += ", Direction=" + direction
msg += ", RequestResponse=" + reqResp
msg += ", TxID=" + str(txId)
msg += ", Reply=" + str(reply)
msg += ", Message=" + message
logger = system.util.getLogger("SECSGEM.Gateway.onSecsGemRealtimeUpdate")
logger.info(msg)
```

Once an `onSecsGemRealtimeUpdate` Message Handler has been specified, it needs to be specified within the Gateway. On the Gateway, navigate to **Connections > SECS/GEM > Settings** to access the SECS/GEM Settings page and specify which project holds the Gateway Message Handler and which project holds the Client Message Handler. These can be the same project or different projects.

Message Handler Example Using the Simulator

This example configures the **S2F14** message to call a Gateway Message Handler. It assumes the equipment connection is connected to a simulator and is using the default SDL file. To capture messages using `onSecsGemRealtimeUpdate`, the SDL file must be configured to allow the message to be routed through the Message Handler. In this example, the SDL file is updated so that **S2F14** is processed by the Message Handler.

1. Create a Gateway Message Handler called **onSecsGemRealtimeUpdate**.
2. On the Gateway, go to the **Connections** section and navigate to **SECS/GEM > Settings**.
3. Specify the project that contains the Gateway Message Handler in the **Gateway Message Handler Project** setting.
4. Navigate to **SECS/GEM > Equipment**.
5. Expand the **three dots menu** for the equipment connection and select **Edit File**.
6. Locate the **S2F14** message in the list.
 - i. Add **realtime** as an Auto type object with a value of **true** directly under the S2F14 root.
 - ii. Create an array called **CommonID** directly under the S2F14 root.
 - a. Add **0: ECV** to the **CommonID** array as an Auto type object.
7. Save the SDL file and restart the equipment connection.

8. After the equipment connection is reestablished, send an **S2F13** message with the body shown below:

S2F13 Body

```
[
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 200
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 201
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 203
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 205
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 206
  },
  {
    "doc": "ECID, Equipment Constant ID",
    "format": "U4",
    "value": 207
  }
]
```

Custom SECS Message Response Handlers

For messages that expect a response, SECS/GEM automatically sends a response based on the SDL configuration. You can override this behavior by creating a custom Message Response Handler, which allows responses to be generated entirely through Python scripting. Using the data given to us in the

payload, the Message Handler can then manipulate it in any way before calling the `system.secsgem.sendResponse` function, which will send the response back to the equipment.

NOTE

Only one custom response can be configured per stream and function for an equipment connection. Custom response handlers are supported for messages received over the SECS-I (serial) protocol.

Payload Key	Description
Equipment	The name of the equipment connection.
TxID	The transaction identifier.
SystemBytes	The system bytes associated with the response.
Message	The message contents represented as JSON.

To configure a custom response handler, open **Connections > SECS/GEM > Equipment**, expand the **three dots menu** for the equipment connection, and select **Custom Responses**. Each entry maps a stream and function to a Gateway Message Handler.

On the Message Response Handlers page, we can setup a link between the stream function response we want to intercept and which Message Handler will be used to handle it. Each Message Response Handler has the following settings.

Setting	Description
Enabled	If true, the specified script message handler will be fired when the specified StreamFunction is encountered.
Intercept Stream Function	The SECS stream and function type of the message that will be intercepted. Use a value such as "S6F2"
Gateway Message Handler	The name of the script message handler that will create the SECS response message. This message handler must be created in a project to provide custom responses, and must be a Gateway Event Script message handler.
Project Name	The project that contains the script message handler specified above.

SECS/GEM Simulator

Overview

The SECS/GEM module includes the capability to add and configure equipment simulators. Equipment simulators include basic functionality and can handle a number of SECS messages, allowing you to model a simulator after an existing tool. You can also add hard-coded responses to messages that do not exist in the simulator.

Simulators are created from the Gateway under **Connections > SECS/GEM > Simulators**. Simulator creation is typically performed as part of initial setup when configuring a SECS/GEM environment. For step-by-step instructions on creating a simulator and connecting it to an equipment connection, see the [Using the SECS/GEM Module](#) page.

Simulator SDL File

A simulator is driven by data defined in its SDL file. This file uses the same general structure as the SDL file for equipment connections, but also includes simulator-specific configurations such as default control states, status variables, equipment constants, reports, collection events, and optional hard-coded behavior. The simulator SDL file is read when the simulator starts.

The simulator SDL file can be edited directly, but creating or modifying objects manually can take significant time. A Sim Variables panel provides structured access to the simulator SDL file and is the primary interface for viewing and modifying simulator configuration. Any changes made through the Sim Variables panel are written to the SDL file.

Follow the steps below to modify the simulator SDL file from the Gateway:

1. Navigate to **Connections > SECS/GEM > Simulators**.
2. Click the **three dots menu** next to the simulator you want to modify and select **Sim Variables**.
3. The SDL file is displayed in a tree view, allowing you to browse its contents and edit individual object values. Click **Save Changes** when finished.

After saving changes to the simulator SDL file, the simulator must be restarted for the updates to take effect. For additional simulator behavior and information, see the [Additional SECS/GEM Simulator Features](#) page.

Default Messages

The simulator implements the GEM Communications State Model and the GEM Control State Model, and can respond to the following requests by default:

- **S1F1** - Are You Online
- **S1F3** - Selected Equipment Status Request
- **S1F11** - Status Variable Namelist Request
- **S1F13** - Establish Communications Request
- **S1F15** - Request OFFLINE
- **S1F17** - Request ONLINE
- **S2F13** - Equipment Constant Request
- **S2F15** - New Equipment Constant Send
- **S2F17** - Data and Time Request
- **S2F23** - Trace Initialize Send
- **S2F29** - Equipment Constant Namelist Request
- **S2F31** - Date and Time Set Request
- **S2F33** - Define Report

For example, sending **S1F15** transitions the simulator to an OFFLINE state. In this state, most messages are rejected with **S1F0**. Sending **S1F17** transitions the simulator back to an ONLINE state. Control state changes can trigger event reports, as defined in the simulator SDL file.

Although additional messages can be defined in the simulator SDL file, the simulator only responds automatically to the messages listed above. Messages that are not handled by the simulator are rejected unless an **Echo Response** is configured.

Simulator Properties

When using a simulator, there are a number of properties that can be configured.

Property Name	Description
Name	Name of the simulator.
Description	Description of the simulator.
Enabled	Enables or disables the simulator. A disabled simulator prevents communication with the equipment connection. Default is TRUE.
Active IP Address	IP address of the equipment connection when using Active mode. Default is localhost.
Active Port	Port used in Active mode. Must be unique when running multiple simulators. Default is 5000.
Passive IP Address	IP address Ignition listens on when using Passive mode. Default is localhost.
Passive Port	Port used in Passive mode. Must be unique when running multiple simulators. Default is 5000.
Device ID	Unique equipment identifier. Must be an integer. Default is 0.
Connection Mode	Connection method: Active, Passive, or Alternating. Default is PAS.

Advanced Properties

Property Name	Description	Default
T3 Reply Timeout	Specifies the seconds that the simulator will wait for an expected SECS message reply.	45
T5 Connect Separation Timeout	Specifies the seconds which must elapse between successive attempts to connect to an equipment connection after disconnection.	10
T6 Control Transaction Timeout	Specifies the seconds which a control transaction (such as LinkTest or Select) may remain open before it is considered a communications failure.	5
T7 Not Selected Timeout	Seconds which a TCP/IP connection can remain in NOT SELECTED state (i.e., no HSMS activity) before it is considered a communications failure. This timeout is only used in Passive mode.	10
T8 Network Intercharacter Timeout	Maximum seconds between successive bytes of a single HSMS message which may expire before it is considered a communications failure. This applies to HSMS messages received from an equipment connection.	5

Additional SECS/GEM Simulator Features

Default Variables

Each simulator is initialized with a set of status variables and equipment constants intended for demonstration and testing. These objects are active within the simulator, meaning standard SECS/GEM requests, such as **S1F3**, can be used to read their definitions and current values. Equipment constant values can also be updated using the **S2F15** request.

Status variables and equipment constants are defined in the simulator SDL file under the **data** section. Any changes made in the **Sim Variables** panel are written to the simulator SDL file and take effect after the simulator is restarted. For more information on how these objects are structured, see the [Simulator SDL File](#) section.

Status Variables

Status variables define values that represent the current state of the simulator, such as operating mode, control state, or sampled measurements. Each status variable includes an ID, name, units, format, and value. Optional event triggers can be defined so that a change in value generates a collection event. Status variables are stored in the simulator SDL file and edited in the "Status Variables" section of the Sim Variables panel. Definitions captured from an existing tool using the **S1F11** request can be used as a reference when defining status variables in the simulator. Since **S1F11** does not include format or value information, formats and values must still be configured manually.

Example Status Variable Definition

The example below shows a single status variable entry as it appears in the simulator SDL file. To use this example, add a new entry under "Status Variables" in the Sim Variables panel and adjust the values as needed for your simulator.

```
{
  "svid": 2,
  "name": "ControlState",
  "units": "states",
  "format": "A",
  "value": "ON-LINE/REMOTE",
  "eventTriggers": [
    {
      "ceid": 1,
      "newValue": "ON-LINE/REMOTE"
    },
    {
      "ceid": 2,
      "newValue": "ON-LINE/LOCAL"
    }
  ]
}
```



EXAMPLE DETAILS

- **svid** uniquely identifies the status variable.
- **format** defines the SECS data type returned in responses.
- **eventTriggers** specify which collection events are fired when the value changes.

After saving changes, restart the simulator for the variable to be available to SECS requests such as **S1F3**.

Equipment Constants

Equipment constants define configurable parameters that typically change infrequently, such as limits, operating modes, or timeouts. Constants include minimum and maximum values, units, formats, and default values. Equipment constants are stored in the simulator SDL file in the "Equipment Constants" section and can be read or modified using SECS requests like **S2F13** and **S2F15**.

Definitions captured from an existing tool using the **S2F29** request can be used as a reference when defining equipment constants in the simulator.

Example Equipment Constant Definition

The example below shows a single equipment constant entry. To use it, add a new entry under "Equipment Constants" in the Sim Variables panel and adjust the values as needed for your simulator.

```
{
  "ecid": 201,
  "name": "EstablishCommunicationsTimeout",
  "minValueFormat": "U2",
  "minValue": 0,
  "maxValueFormat": "U2",
  "maxValue": 10000,
  "format": "U2",
  "defaultValue": 15,
  "units": "seconds"
}
```

EXAMPLE DETAILS

- **ecid** uniquely identifies the equipment constant.
- **minValue** and **maxValue** define valid limits.
- **defaultValue** is used when the simulator starts or resets.

Restart the simulator after saving changes for the constant to be applied.

Echo Responses

Some SECS/GEM messages are not implemented by default in the simulator. Echo Responses allow you to define a fixed response for a specific request message. An Echo Response maps an incoming request to a predefined response message. This allows the simulator to return valid responses for unsupported messages or to emulate tool-specific behavior. Echo Responses are defined in the simulator SDL file and edited through the Sim Variables panel, where they appear alongside other simulator configuration data.

When adding support for new messages, you may also need to define the request and response messages in the messages section of both the simulator SDL file and the equipment connection SDL file.

Example Echo Response Definition

The example below defines a response to an **S1F1** request. Add this definition to the Echo Responses section of the simulator SDL. Once applied, the simulator returns a fixed **S1F2** reply whenever it receives **S1F1**.

```

{
  "request": {
    "stream": 1,
    "function": 1
  },
  "response": {
    "header": {
      "stream": 1,
      "function": 2,
      "reply": false
    },
    "body": [
      {
        "format": "A",
        "value": "SimEquip",
        "doc": "MDLN"
      },
      {
        "format": "A",
        "value": "1.0",
        "doc": "SOFTREV"
      }
    ]
  }
}

```

Event Runs

Event Runs allow the simulator to generate S6F11 Event Report Send messages on a defined schedule. An Event Run consists of one or more events that execute in sequence, each with a configurable delay. Each event in an Event Run contains a complete **S6F11** message definition, which is sent when the event executes. Event Runs are defined in the simulator SDL file and edited through the Sim Variables panel. They are commonly used to simulate periodic reporting or state-driven events. Event Runs must be started manually. While an Event Run is active, events are sent in order until the run completes or is cancelled.

Example Event Run Message Body

The example below shows a complete **S6F11** message body used by an Event Run. To use this example, paste the message body into an event definition and adjust IDs and values to match your simulator configuration.

All referenced **CEIDs**, **RPTIDs**, and variables must already exist in the simulator SDL file for the message to be sent successfully.

```
{
  "header": {
    "stream": 6,
    "function": 11,
    "reply": true
  },
  "body": [
    {
      "format": "U4",
      "value": 2,
      "doc": "DATAID"
    },
    {
      "format": "U4",
      "value": 3,
      "doc": "CEID"
    },
    [
      [
        {
          "format": "U4",
          "value": 1,
          "doc": "RPTID"
        },
        [
          {
            "format": "A",
            "value": "ON-LINE/REMOTE",
            "doc": "V"
          }
        ]
      ]
    ]
  ]
}
```

Basic SECS/GEM Troubleshooting

Equipment Connections

If you are unable to create an equipment connection after configuring the connection settings, verify that the **Active Port** and **Passive Port** values are not already in use. Each equipment connection and simulator must use unique port numbers. Ports cannot be shared between multiple connections.

To verify port usage:

1. Navigate to **Connections > SECS/GEM > Equipment** on the Gateway.
2. Locate the equipment connection and open the **three dots menu**.
3. Select **Edit**.
4. Confirm that the Active and Passive Port values are not used by another equipment connection or simulator.

If the ports are already in use, update them to unique values and save your changes. Once the ports are available and correctly configured, the equipment connection status should transition to **Communicating**.

DATABASE TABLES ARE NOT BEING CREATED ON EQUIPMENT CONNECTION STARTUP

If you are using MySQL, ensure that the latest MySQL JDBC driver is installed in Ignition. You can download the driver from the MySQL website or from the Gateway under **Connections > Databases > Settings**.

Logging

Each equipment connection has an associated logger that records diagnostic information. Equipment connection loggers and their current severity levels can be viewed from the Gateway under **Diagnostics > Logs**.

By default, only messages with a severity of **INFO**, **WARNING**, or **ERROR** are logged. Additional diagnostic information can be captured by lowering the log level. At the **TRACE** level, all SECS messages sent and received by the equipment connection are logged.

To change the log level:

1. Open **Diagnostics > Logs**.
2. Click the **Settings** icon to open the Log Configuration panel.
3. Locate the logger associated with the equipment connection.
4. Select a new severity level from the dropdown.

wrapper.log File

Log messages can be viewed both in the Gateway **Diagnostics** section and in Ignition's log file, **wrapper.log**. When troubleshooting unexpected behavior, enabling verbose logging and reviewing the wrapper.log file often reveals validation errors, connection issues, or unexpected message handling.

The wrapper.log file is located in different directories depending on the operating system:

- **Windows:** Program Files\Inductive Automation\Ignition\logs
- **Linux:** /var/log/ignition
- **macOS:** /usr/local/ignition/logs

On Linux systems, the following command can be useful for monitoring log messages in real time:

```
tail -f /var/log/ignition/wrapper.log
```

Failure to Send a SECS Message

All SECS messages sent or received by an equipment connection are validated against the configured **SECS Definition Language (SDL)** file. A common cause of message failures is a mismatch between the message being sent or received and its definition in the SDL file.

If a message fails validation:

- The message is inserted into the **Errors** database table for the equipment connection.
- A validation error is logged, describing why the message failed.
- Details are written to both the equipment connection diagnostic logs and the wrapper.log file.

To resolve validation errors, update the SDL file to ensure that the message structure, formats, and items match the expected definition. The SDL file can be edited directly from the Gateway using the built-in SDL editor on the **Equipment** page, or downloaded, edited externally, and uploaded again.