

SEMI E183-0123

SPECIFICATION FOR RICH INTERACTIVE TEST DATABASE (RITdb)

This Standard was technically approved by the Automated Test Equipment Global Technical Committee. This edition was approved for publication by the global Audits and Reviews Subcommittee on May 5, 2022. Available at www.semiviews.org and www.semi.org in January 2023; originally published November 2021.

NOTICE: The section symbol (§) is used to reference a numbered section, paragraph, or header and all subordinate paragraphs, headers, and embedded material (i.e., EXCEPTIONs, lists) therein. The paragraph symbol (§) is used to reference a particular numbered paragraph and embedded material therein. When duplicated (i.e., §§ and §§§) the symbols are used to reference more than one section or paragraph, respectively.

NOTICE: Paragraphs entitled ‘NOTE:’ are not an official part of this Standard or Safety Guideline and are not intended to modify or supersede the official Standard or Safety Guideline. These have been supplied by the global technical committee to enhance the usage of the Standard or Safety Guideline.

Table of Contents

1 Purpose	3
2 Scope	3
3 Limitations.....	3
4 Referenced Standards and Documents	3
5 Terminology	3
6 Resources, Organizations, and Standards	4
7 Conventions.....	4
8 General Requirements for RITdb	6
9 Overview	6
10 General Requirements for RITdb	7
11 Identity Requirements.....	8
12 Provenance Management.....	9
13 RITdb Transport Requirements	9
14 RITdb Containers Requirement.....	12
15 Requirements for Use Cases in RITdb Containers	13
16 RITdb Container Contents Requirements.....	13
17 Requirements for Customizing spec.ritdb.....	14
18 Event Message Body Requirements	14
19 Event Message Topic Requirements for Subscribing to and Publishing Events	15
20 Streaming RITdb Container Requirements.....	15
Appendix 1 Statement of Compliance	17
Appendix 2 The Spec.Ritdb Database Structure.....	21
Appendix 3 Cbor Conventions	22
Related Information 1 Design Objectives.....	24
Related Information 2 RITdb Container Object Model Description.....	27

List of Tables

Table 1 Requirement Identifiers	4
Table 2 Example Table with Requirements.....	5
Table 3 Table Topic Tokens Order.....	10
Table 4 RITdb Reserved Topics.....	12
Table 5 Column Definition for Table ‘ritdb1’	12
Table 6 Allowed Data Types	13

List of Figures

Figure 1 Overall Architecture Diagram	7
Figure 2 RITdb Global Scope.....	25
Figure 3 RITdb Attribute types.....	27

1 Purpose

- 1.1 This Specification describes a data and event sharing and streaming methodology for semiconductor test and related operations.
- 1.2 The scheme fully supports legacy equipment and allows equipment vendors the flexibility to provide additional data items via extensions.
- 1.3 Real time, historical, and widely distributed data are fully supported.
- 1.4 Provenance of all data is supported via customizable metadata along with cryptographic hashing and signing.
- 1.5 Bidirectional data and messages are supported to enable smart manufacturing.

2 Scope

- 2.1 This Specification describes data and event communication to/from/within a semiconductor test cell/floor.
- 2.2 This Specification describes RITdb, a data container which can hold a wide range of legacy data, modern extensions and user defined extensions based on relational database technology. This Standard supports provenance via customizable metadata and machine validation.
- 2.3 This Specification describes a real time publish subscribe event messaging system optimized for semiconductor manufacturing and based on a Machine-to-Machine Internet of Things (M2M-IoT) protocol which is an easily extensible and efficient machine connectivity.

NOTICE: SEMI Standards and Safety Guidelines do not purport to address all safety issues associated with their use. It is the responsibility of the users of the documents to establish appropriate safety and health practices, and determine the applicability of regulatory or other limitations prior to use.

3 Limitations

- 3.1 This Specification does not mandate performance metrics or expectations as that is a property of implementations.

4 Referenced Standards and Documents

4.1 *ISO® Standards*¹

ISO 8859-1 — ASCII limited version of UTF-8

NOTICE: Unless otherwise indicated, all documents cited shall be the latest published versions.

5 Terminology

5.1 *Abbreviations and Acronyms*

- 5.1.1 *CBOR* — concise binary object representation
- 5.1.2 *IoT* — Internet of Things
- 5.1.3 *M2M* — machine-to-machine
- 5.1.4 *MDIR* — metadata indexed repository
- 5.1.5 *MQTT* — Message Queuing Telemetry Transport
- 5.1.6 *Pub/Sub* — publish-subscribe
- 5.1.7 *RITdb* — Rich Interactive Test Database
- 5.1.8 *UTF-8* — Unicode Transformation Format
- 5.1.9 *SQL* — Structured Query Language

¹ International Organization for Standardization, ISO Central Secretariat, Chemin de Blandonnet 8, CP 401-1214 Vernier, Geneva, Switzerland; Telephone: +41.22.749.01.11, Fax: +41.22.733.34.30, <http://www.iso.org>. ISO trademark is owned by International Organization for Standardization.

5.1.10 *SQLite* — Single File Database

5.2 Definitions

5.2.1 *broker* — a message routing entity which supports a publish subscribe communications network.

5.2.2 *custom.ritdb* — a spec.ritdb file with custom specification definitions.

5.2.3 *derived event* — a higher level opinion on the state of the environment based on combining history, facts, models, and analysis.

5.2.4 *domain* — a broker or cluster's identity and coverage scope.

5.2.5 *fact* —the immutable representation of some facet of the total environment's state. Facts are determined by a measurement or state transition.

5.2.6 *metadata* — a collection of attributes, defining the provenance of a RITdb object, stored external to the object.

5.2.7 *provenance* — the chronology of the contents, ownership, custody, or location of a RITdb object.

5.2.8 *run* — set of data with a common provenance.

5.2.9 *spec.ritdb* — a RITdb file containing specification definitions.

5.2.10 *subscription* — a topic which is listened on.

5.2.11 *usecase* — a defined subset of the standard applicable to a use defined set of requirements.

6 Resources and Organizations

6.1 Important resources and organizations used in the generation and definition of this Specification are found in:

6.1.1 *Single File Database (SQLite)* — <https://www.sqlite.org/index.html>

6.1.2 *Message Queuing Telemetry Transport (MQTT)* — <https://mqtt.org/>

7 Conventions

7.1 Requirements Identification

7.1.1 The following notation specifies the structure of requirement identifiers.

7.1.1.1 The following requirements prefix format is used at the beginning of requirement text. Refer to Table 1 for the format notation of the requirements prefix.

- [E183.00-RQ-0000-00]

7.1.1.2 To mark the end of the requirement text, the following suffix format is used.

- [/RQ]

7.1.1.3 Requirements in the body text are highlighted with a border and light green background (may appear gray in black and white printouts) as shown below.

[Esss.ss-RQ-nnnnn-nn] Requirement text. [/RQ]

Table 1 Requirement Identifiers

<i>Format Notation</i>	<i>Purpose</i>
Esss.ss	SEMI Standards Specification identifier. Examples: E87.00, E87.01, E134.00.
RQ	Indicates this is a requirement identifier.
nnnnn	Unique five-digit number within this Specification. 90000–99999 are reserved for use by SEMI.
.nn	Two-digit number that indicates version level of the requirement (a value of .00 is used for the first version of a requirement).
/RQ	Indicates the end of a requirement.

7.1.1.4 Note that the requirements in Appendix 1 use the reserved requirement numbers noted in Table 1 because they are repeated in other SEMI Standards that use this convention.

7.1.2 Requirements in tables are delimited in one of two ways.

7.1.2.1 Where the requirement occupies an entire row in the table, the requirement identifier (RequirementID) is placed in column 1. No '[/RQ]' is used to mark the end of the requirements text in this case. The table may also contain rows that are not requirements. In this case, the column 1 cell is left blank.

7.1.2.2 Non-requirements text related to a requirement row may be included in an adjacent row. The relationship between the rows is indicated by a broken line separating the two.

7.1.2.3 Where the requirement occupies only one cell, the text in the cell includes the RequirementID prefix and suffix similar to requirements in the body text.

7.1.2.4 No cell in a table will contain both requirement and non-requirement text.

7.1.2.5 Cells that contain requirements are shaded light green.

7.1.2.6 Table 2 provides an example of requirements in a table. Note that in this example, the same requirement is presented in two alternative formats. In practice, mixing the two approaches in the same table is not typically done.

Table 2 Example Table with Requirements

<i>RequirementID</i>	<i>Statement</i>	<i>Additional Detail</i>
Esss.ss-RQ-nnnnnn-nn	The light shall be the sole blue light.	There shall be no similar color in the light panel.
	The blue light is typically placed at the leftmost position in the light panel.	
	[Esss.ss-RQ-nnnnnn-nn] The light shall be the sole blue light. There shall be no similar color in the light panel. [/RQ]	The blue light is typically placed at the leftmost position in the light panel.

7.1.3 Only text marked with the RequirementID is a requirement of this Specification.

7.1.3.1 Clarifications, examples, and related recommendations may be provided near a requirement, but are not part of the requirement.

7.1.3.2 Note that the word 'should' is used in some non-requirements text, where it denotes a recommendation or a best practice, not a requirement.

7.1.3.3 Note that the words 'must' and 'will' are used in the Implementation Overview section to illustrate the logical result of requirements specified in the prerequisites and requirements section. Such statements should not be considered requirements.

7.1.4 Parent-child relationships of requirements are noted in Appendix 1, Table A1-1. Where a parent requirement includes conditions or selection criteria, these are passed to their children. For example, if a parent requirement is stated to apply only to photolithography equipment, any child requirements also apply only to such equipment. The condition need not be restated for each child requirement.

7.1.4.1 Compliance to a parent requirement includes compliance to all of its child requirements.

7.1.4.2 Some cells or rows in a table may contain headings that are assigned a RequirementID. These are parent requirements for other requirements in the table—typically for the rows that immediately follow that heading.

7.2 Use of a Structured Query Language (SQL) database for the standard. The majority of this Standard is provided as an SQLite database. This database is referred to in this Standard as spec.ritdb.

7.2.1 *SQL Script Conventions* — Each section which references the standard database will provide SQL code to display the portion of the database referred to in that section. This script can be found in the ‘scripts’ table of the spec.ritdb as described in Appendix 5.

7.2.2 *Usecase* — Each script has one or more attributes with name ‘USECASE’. When a script is referred to in this Standard all scripts with a matching usecase value to the specified usecase are indicated.

7.2.3 *Script Id* — Each script shall have a unique ID held in attribute named ‘SCRIPT_ID’.

7.2.4 *Validation* — spec.ritdb is designed to enable machine validation of the contents of a compliant RITdb container against the standard along with custom.ritdb extensions. Script examples are provided which demonstrate partial validations.

8 General Requirements for RITdb

8.1 The following is the top-level parent requirement of all others in this Specification.

[E183.00-RQ-00001-00] Compliance to this Specification shall require compliance to all other identified requirements in this Specification, including complementary files. [/RQ]

8.2 spec.ritdb File

8.2.1 This Specification includes a spec.ritdb file, ‘SEMIE183-00-spec.ritdb’ file, that provides the normative definitions of the types and methods described here.

[E183.00-RQ-00002-00] Implementations shall use the Complementary File ‘SEMIE183-00-spec.ritdb’ file that accompanies this Specification. The Complementary File is an official part of and accompanies this Specification. The contents of this spec.ritdb file constitutes a core part of this Specification. [/RQ]

8.2.2 This Specification also includes Various Materials. These Various Materials are not an official part, but accompany this Specification. Use of these Various Materials is optional and is not required for compliance with this Specification.

8.2.3 The file SEMI E183-00-spec-helper.xlsx is an export of the SEMI E183-00-spec.ritdb contents into an excel file. The excel file can be used to view the spec.ritdb file without the need for an SQLite browser. This file is provided for the reader’s convenience and is not meant to be used by implementors.

NOTE 1: The spreadsheet has been provided as an illustrative example in the Supplementary Materials: Various Materials section of the SEMI Standards Web site.

9 Overview

9.1 Test cells generate large amounts of disparate but related data which must be collected, shared, and analyzed with the intent of improving the performance, quality, or efficiency of each operation. This data is both real time and historical with sizes ranging from a few bytes to many megabytes. Real time support is necessary to provide immediate responses to new data while maintaining history to analyze changes which lead to failures. Both requirements can be met through the use of message streams based on standardized addressing and content. This Standard covers the messaging scheme for both historical and immediate data along with the message body formats including for both streaming and archival data containers.

9.2 Figure 1 shows an overall view of the RITdb architecture. All participants are interconnected via a M2M-IoT based publish/subscribe (Pub/Sub) network (the lines/arrows in Figure 1).

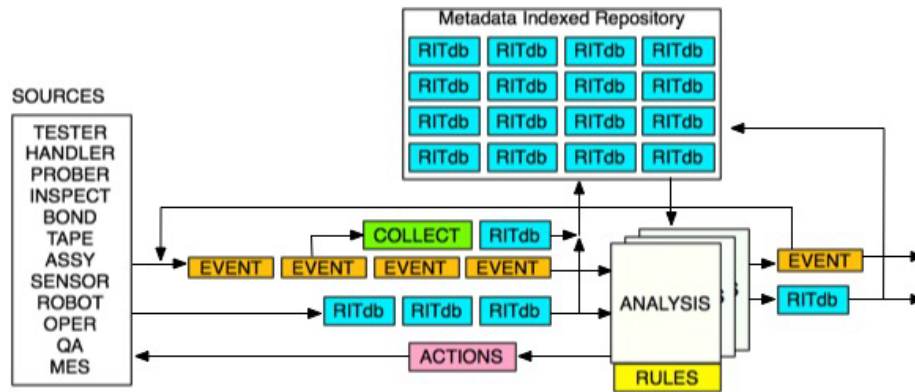


Figure 1
Overall Architecture Diagram

9.2.1 Participant Roles

9.2.1.1 *Events* — Events (orange boxes) are temporal messages. Events may be ‘facts’ as created by sources or ‘derived’ as created by rule and analysis systems.

9.2.1.2 *Actions* — Actions (purple boxes) are requests to act.

9.2.1.3 *RITdb* — RITdb (light blue boxes) represents immutable and persistent containers of events and data.

9.2.1.4 *Pub/Sub Infrastructure* — A peer to peer means to communicate containers and events. Shown by the lines and arrows.

9.2.1.5 *Sources* — Sources have two roles. They publish ‘facts’ in the form of RITdb containers and events. They also subscribe to messages requesting actions based on the execution of rules.

9.2.1.6 *Collect* — Collectors (green boxes) aggregate messages into RITdb containers.

9.2.1.7 *Metadata Indexed Repository (MDIR)* — The MDIR of RITdb containers, maintains the history of RITdb containers along with the metadata which defines each containers provenance.

9.2.1.8 *Analysis* — The analysis subsystem(s) compares facts, RITdb containers and history against rules to generate actions, new RITdb containers and derived events.

10 General Requirements for RITdb

10.1 As RITdb describes a machine to machine communications architecture a more limited character set was indicated than the full Unicode Transformation Format (UTF-8). When appropriate this Standard uses a single byte subset of UTF-8 referred to a RITdb_ASCII.

[E183.00-RQ-00003-00] RITdb_ASCII shall be strings of encoding ISO 8859-1 with byte values of 0x01 to 0x7f.
[RQ]

10.2 Maps (collections of key value pairs) are used extensively in the RITdb Standard. To facilitate M2M some limits have been placed on the structure of the keys by defining a RITdb_KEY_STRING type.

[E183.00-RQ-00004-00] RITdb_KEY_STRING shall be case sensitive RITdb_ASCII constrained to A-Z, a-z, 0-9, ‘_’, ‘.’. [RQ]

[E183.00-RQ-00005-00] The first character of a RITdb_KEY_STRING shall be from the characters A-Z or a-z.
[RQ]

10.3 Dates and times are also used throughout RITdb so we limited the time strings to two formats. Once for coarse time and one for fine.

[E183.00-RQ-00006-00] TIME DATE FORMAT shall be YYYY-MM-DDTHH24:MI:SS or YYYY-MM-DDTHH24:MI:SS.sss per RFC 3339. [/RQ]

[E183.00-RQ-00007-00] BASE36 ENCODING(base36) – Integer encoded as a constrained RITdb_ASCII string. Each base36 digit is represented by a character within 0-9 and A-Z where 0 represents decimal 0 and Z represents decimal 35. [/RQ]

[E183.00-RQ-00008-00] HEX_STRING – Binary encoded as hex using RITDB_ASCII constrained to the characters 0123456789ABCDEF. [/RQ]

[E183.00-RQ-00009-00] RITdb_TIMESTAMP 6 characters of base36 encoded seconds since UTC 1/1/1990T00:00:00 plus two characters of base 36 encoded sequence for a total of 8 bytes. [/RQ]

10.4 RITdb_HASH SHA1 and SHA256 are allowed.

[E183.00-RQ-00010-00] SHA1 shall be represented as a 20-byte array left padded with zero as a 40 character hex string. SHA256 shall be represented as a 32-byte array left padded with zero as a 64 character hex string. [/RQ]

10.5 *RITdb_IdentityToken*

[E183.00-RQ-00011-00] A RITdb_IdentityToken shall consist of 8 bytes encoded using RITdb_ASCII further limited to the upper-case alpha characters plus the digits 0-9, where the first byte is an upper case alpha character of A-Z. [/RQ]

[E183.00-RQ-00012-00] Self-created (self-signed) identities shall have the character X as the first character with the following 7 characters being the base36 encoding of a random number. [/RQ]

10.6 *Object_Creation_Token* — There is a need for easy to generate locally unique object identifiers. For convenience the first half represents the signer and the latter half the creation time.

[E183.00-RQ-00013-00] An Object_Creation_Token shall be the RITdb_identityToken for the entity signing the object concatenated with a RITdb_TIMESTAMP. [/RQ]

11 Identity Requirements

11.1 Key to all forms of data analysis are assumptions made about the quality of data. One of the key parts of data quality is the ability to trust the source and modifiers of data and other items within the infrastructure. Within the RITdb infrastructure two tokens are provided to determine identity.

11.2 RITdb_Identity_Token is the identifier of a participant in the infrastructure (§ 10.5).

11.3 *Object_Creation_Token* — The signer of a RITdb shall provide an identity string for that container which is unique within the scope for which the data will be trusted (§ 10.6).

[E183.00-RQ-00014-00] Each signer shall generate a monotonically increasing in time RITdb_TIMESTAMP. [/RQ]

[E183.00-RQ-00015-00] For initial object creation the RITdb_TIMESTAMP shall be within 5 minutes of current UTC time. [/RQ]

[E183.00-RQ-00016-00] For mutated or forked objects the time portion of the RITdb_TIMESTAMP for the new object shall occur after the time portion of the RITdb_TIMESTAMP of the source object. [/RQ]

[E183.00-RQ-00017-00] Cryptographic public/private key pair – Each signer of objects shall have a public key accessible by a reference to its RITdb_IdentityToken built using the RSA algorithm with key size of 512 bits or greater. [/RQ]

[E183.00-RQ-00018-00] Content Hash – Content shall be identified by a RITdb_hash function of its contents. [/RQ]

[E183.00-RQ-00019-00] The signature of an object shall be a byte array consisting of the content hash of the metadata followed with the content hash of the contents with the combination encrypted by the signer's private key. [/RQ]

12 Provenance Management – Metadata

12.1 The intent of this Standard is to provide wide access to data containers via a distributed data infrastructure. To maintain data integrity these data items are identified by the hash of their content instead of the traditional file path-based identity. This approach requires a means to describe the content without accessing the referenced data structure. A common approach to this is referred to as 'provenance' which is based on the use of metadata.

12.2 *Metadata* — Metadata is expressed as key value pairs where the key are encoded as RITdb_KEY_STRINGs and values are encoded as RITdb_ASCII strings. The entire metadata for an object is then concise binary object representation (CBOR) encoded into and bound to its object via cryptographic techniques. Metadata may be grouped into different categories. During the encoding process these categories are kept separate and may be encrypted independent of other groups. Groups can include information on content, source, analysis, integrity along with custom attributes. Two categories are defined by this standard (public, identity) via key construction.

[E183.00-RQ-00020-00] Identity metadata attribute names shall be RITdb_KEYSTRINGS and start with a lower cases letter and contain at least one period. Attribute values shall be RITdb_ASCII. [/RQ]

[E183.00-RQ-00021-00] All required metadata items shown by executing the script with ID xxx shall be included in an RITdbContainers metadata. [/RQ]

[E183.00-RQ-00022-00] Public Metadata – Public metadata names shall be unencrypted RITdb_ASCII camel case alphanumeric characters starting with an uppercase letter. [/RQ]

[E183.00-RQ-00023-00] Binary Format for Metadata – a binary object representing metadata shall be CBOR encoded as shown in Appendix 3. [/RQ]

13 RITdb Transport Requirements

13.1 M2M-IoT Pub/Sub Network

13.1.1 Data sharing within RITdb is implemented via a publish subscribe, machine to machine communications protocol based on MQTT. This protocol has addressing based on topics with support for raw binary bodies.

13.1.2 *Client ID* — All clients to the Pub/Sub network shall have a network wide unique identifier. It is recommended that a Client ID consist of 23 RITdb_ASCII characters of which at least 10 are based on a random number. Allowed characters are '0123456789abcdefghijklmnopqrstuvwxyzABCDEFGHIJKLMNOPQRSTUVWXYZ'.