Low Level Reader Protocol (LLRP) 1.1 Conformance Requirements Document

Approved October 13, 2010

Disclaimer

EPCglobal Inc™ is providing this document as a service to interested industries. This document was developed through a consensus process of interested parties.

Although efforts have been to assure that the document is correct, reliable, and technically accurate, EPCglobal Inc makes NO WARRANTY, EXPRESS OR IMPLIED, THAT THIS DOCUMENT IS CORRECT, WILL NOT REQUIRE MODIFICATION AS EXPERIENCE AND TECHNOLOGICAL ADVANCES DICTATE, OR WILL BE SUITABLE FOR ANY PURPOSE OR WORKABLE IN ANY APPLICATION, OR OTHERWISE. Use of this document is with the understanding that EPCglobal Inc has no liability for any claim to the contrary, or for any damage or loss of any kind or nature.

Abstract

This document outlines the approach to conformance testing for the EPCglobal Low Level Reader Protocol (LLRP) 1.1 specification. The objective of an LLRP conformance certification program is to test and certify solution providers' implementations of the EPCglobal LLRP interface v1.1. Certification of LLRP conformance provides confidence for buyers in the operational capability of a specific product's implementation of the LLRP interface, while providing solution providers a benchmark to assure product functionality.

Status of this document

This section describes the status of this document at the time of its publication. Currently this document is fully approved by the Reader Operations Working Group and has completed final processing within the GS1 EPCglobal Standards Development Process. In the future other documents may supersede this document. The latest version of this document can be viewed at http://www.epcglobalinc.org/standards/llrp with the proviso that you are a GS1 EPCglobal subscriber.

Comments on this document should be sent to the EPCglobal Software Action Group Reader Operations Working Group mailing list at GS1help@gs1.org.

Table of Contents

1 Introduction

Technical implementations of the Low Level Reader Protocol (LLRP) specification may vary due to distinct interpretations of the specification and/or use of proprietary technologies when developing systems that implement the EPCglobal Architecture Framework. Conformance testing provides a mechanism to ensure that solutions adhere to, and are compatible with, the specified standard. A Low Level Reader Protocol (LLRP) Conformance Certification Program provides solution providers a benchmark to assure product functionality according to the LLRP specification, while imparting confidence on potential buyers in the operational capability of a specific product's implementation of the LLRP interface.

LLRP certification represents an endorsement that helps solution provider differentiate their products and services within the marketplace. Certification of LLRP conformance instills both product recognition and a level of public confidence sought by corporate supply chains looking to partner with a solution provider of EPCglobal standard compliant products. Implementation of an LLRP certification program will:

- Help move the industry toward RFID Interoperability
- Accelerate LLRP and EPC Implementations
- Publicly identify product vendors who support the EPCglobal standards.

The focus of this program will be both software and hardware product conformance to the EPCglobal LLRP 1.0.1 Interface Specification. The Low Level Reader Protocol (LLRP) specification describes an interface through which client applications may obtain low-level access to air protocol specific features on an RFID Reader. The design of the interface recognizes that a LLRP implementation may be a software component built independent from a physical hardware device. Or, the implementation may be embedded within an RFID reader. This program places no restrictions on this aspect of an LLRP implementation.

The EPCglobal Reader Operations working group is responsible for defining the LLRP Certification test scenarios that the authorized testing agency will use in developing a test harness and associated test scripts.

2 Scope

An LLRP Conformance Certification Program will focus on testing a given applicant's implementation of the LLRP interface and its conformance to the LLRP 1.0.1 Specification. Test case requirements and benchmark definitions, documented herein, have been developed by the EPCglobal Reader Operations working group.

An LLRP Conformance Certification Program is NOT intended to test the performance, reliability, or scalability of the tested product. And, an LLRP Conformance Certification Program is NOT required to test a hardware device. An applicant's implementation of the LLRP interface MAY be strictly software. However, in this case, the applicant must provide a Reader simulator suitable to executing the test scenarios defined by the LLRP Conformance Certification Program.

3 Program Overview

The LLRP Certification Program will be offered by a certified testing laboratory to solution providers enrolled in the certification program.

Program Implementation and Certificate definition are to be defined by EPCglobal US and a chosen Testing Laboratory.

An EPCglobal LLRP Conformance Certification Program will focus on testing the following aspects of the LLRP interface:

- Support for querying a Reader for its capabilities.
- Support for querying and setting a Reader's configuration.
- Support for Reader inventory and access operations.
- Support for Reader reporting of events and reader operations (i.e., tag data).
- Support and proper handling of error conditions.
- Support for EPCglobal UHF Gen2 air protocol.
- Support for the binary encoding and TCP transport by the specification.

The conformance tests may not be exhaustive, but should be representative of capabilities needed for a successful LLRP implementation. The tests should be defined to be platform independent, and should not require products to be implemented on any particular system or platform.

4 Terminology

This document adopts terminology developed by the World Wide Web Consortium [W3C-Conformance]:

- *Certificate Issuer* The organization that issues certificates of conformance, namely, EPCglobal.
- *Testing Laboratory* An organization that carries out certification testing on behalf of the Certificate Issuer
- *Specification* An EPCglobal specification for which conformance is tested.
- *Implementation Under Test (IUT)* A submission of hardware and/or software for which certification is sought by an EPCglobal subscriber.
- *System Under Test (SUT)* The IUT together with any other apparatus required to carry out the test.
- *Test Method* A description of the test that is applied to the SUT. There may be more than one Test Method available for a given LLRP 1.0.1 specification requirement, each providing a different level of conformance testing.

- *Test Report* A Test Report contains the results of the testing effort. The test report should provide enough information that, if necessary, the testing effort could be duplicated. The testing report should contain:
    - a complete description of the IUT,
    - the name of the Testing Laboratory,
    - the signature of a Testing Laboratory official,
    - the date that the testing was completed,
    - the name and version number of the Test Method
    - the results of the Test Method
    - an unambiguous statement indicating pass or fail.[1]
- *LLRP Conformance Certification Program:* An EPCglobal US sponsored Software/Hardware solution certification program measuring LLRP 1.1 conformance.
- *Certificate of Conformance:* The certificate of conformance is typically a summation of the Test Report. Since it is often used in the procurement process, it includes information most pertinent between the buyer and the seller.[1]

5 Submission Requirements

Solution providers who wish to submit their product(s) for testing must submit the following to the testing laboratory:

- An Implementation Under Test (IUT). This may take one of the following forms:
    - Software or hardware that implements LLRP Reader interface and can report tag and EPC information necessary to conduct the conformance tests below.
    - Any other kind of system that implements the LLRP interface, including (but not limited to) LLRP implementations embedded in RFID readers or other devices.
- A document that includes a statement for each requirement listed in Section 6.1 Mandatory Requirements Matrix and defined to be verified "By Design". Sufficient for each statement is a validation letter stating that the product's implementers acknowledge the requirement and that they confirm that the product is designed to satisfy the requirement. Supporting material may be included with these statements such as the following information:
    - One or more images from the product's User's Guide confirming that the product is designed to include one or more features that satisfy the requirement.
    - Internal test results performed on the IUT that demonstrate the "By Design" requirement.

[1] From W3C-Conformance

- Internal log files captured from the IUT that demonstrate the “By Design” requirement.
- Design documents for the IUT showing satisfaction of the “By Design” requirements.
- Other relevant material

6 LLRP 1.1 Functional Requirements

The LLRP 1.1 Specification defines specific functionality that a valid LLRP Implementation must provide. The following tables outline the specific requirements that must be tested as defined by the LLRP 1.1 specification. Each test requirement entry references the LLRP 1.1 Specification and the test case requirement (TCR) used to verify functionality as defined in section 8 of this document.

6.1 Mandatory Requirements Matrix

The following table outlines the mandatory requirements for an LLRP implementation as defined by the LLRP 1.0.1 Specification. Some entries within this table are marked as mandatory, but are conditionally required by the specification only if the device advertises the corresponding LLRP capability.

Req. No.	**Protocol SubClause**	**Requirements (Requirements, Command, …)**	**Applies to (ref)**	**How Verified (by Demonstration or by Design)**
M0	6.1.1.2	Trigger operation	R	See M69, M70, M71, M72, M74, M75, M76, M79, M80, M82-M90, M116
M1	7.1.2	Process messages in order	R	By Design
M2	7.1.3.1.1	Only UTC or Uptime	R	By Design
M3	7.1.3.1.1.1	UTC parameter	RC	7.1.1
M4	7.1.3.1.1.2	Uptime parameter	RC	7.1.1
M5	7.1.4	Air protocol ID	C	Not Tested *(client requirement only)*
M6	7.1.5	Generic error message	R	7.4.1
M7	8.1	Unsupported custom message	R	7.4.1
M8	8.2	Unsupported custom parameter	R	7.4.1
M9	8.2	Clients accept custom parameters	C	Not Tested *(client requirement only)*
M10	8.5	Ranges and enumerations cannot be extended	RC	Not Tested *(client requirement only)*
M11	9.1.1	Ver field set to 2 prior to successful	C	Not Tested *(client*

		negotiation		*requirement only)*
M12	9.1.1	Negotiated Version	C	Not Tested *(client requirement only)*
M13	9.1.1	StatusCode of M_UnsupportedVersion	C	Not Tested *(client requirement only)*
M14	9.1.1	GET_SUPPORTED_VERSION	RC	7.2.1
M15	9.1.2	Advertised SupportedVersion	RC	7.2.1
M16	9.1.2	Ver field set to 2 prior to successful negotiation	RC	7.2.1
M17	9.1.2	Negotiated Version	RC	7.2.1
M18	9.1.2	GET_SUPPORTED_VERSION_RESPONSE	RC	7.2.1
M19	9.1.3	Message transmission count	C	Not Tested *(client requirement only)*
M20	9.1.3	Ver field set to 2	C	Not Tested *(client requirement only)*
M21	9.1.3	StatusCode of M_UnsupportedVersion	C	Not Tested *(client requirement only)*
M22	9.1.3	SET_PROTOCOL_VERSION	RC	7.2.1
M23	9.1.4	StatusCode of M_UnsupportedVersion	RC	7.2.1
M24	9.1.4	StatusCode of M_UnexpectedMessage	RC	7.2.1
M25	9.1.4	StatusCode of M_Success	RC	7.2.1
M26	9.1.4	Ver field set to 2	RC	7.2.1
M27	9.1.4	SET_PROTOCOL_VERSION_RESPONSE	RC	7.2.1
M28	10.1.1	GET_READER_CAPABILITIES	RC	7.3.1
M29	10.1.2	GET_READER_CAPABILITIES_RESPONSE	RC	7.3.1
M30	10.2.1	GeneralDeviceCapabilities	RC	7.3.1
M31	10.2.1.1	MaximumReceiveSensitivity	R	Not Tested *(optional feature)*
M32	10.2.1.2	ReceiveSensitivityTableEntry	R	7.3.1
M33	10.2.1.3	PerAntennaReceiveSensitivityRange	C	Not Tested *(client requirement only)*
M34	10.2.1.4	PerAntennaAirProtocol	RC	7.3.1
M35	10.2.1.5	GPIOCapabilities	RC	7.3.1
M36	10.2.2	LLRPCapabilities	RC	7.3.1

M37	10.2.2	Support at least one ROSpec, InventoryParameterSpec/AISpec, OpSpec/AccessSpec	R	7.3.1, 7.6.1, 7.8.1
M38	10.2.3	AirProtocolLLRPCapabilities	RC	7.3.1
M39	10.2.3	No more than one reference to any air protocol capabilities parameter	C	Not Tested *(client requirement only)*
M40	10.2.4	RegulatoryCapabilities	RC	7.3.1
M41	10.2.4.1	UHFBandCapabilities	RC	Not Tested *(client requirement only)*
M42	10.2.4.1	No more than one reference to any air protocol	RC	Not Tested *(client requirement only)*
M43	10.2.4.1.1	TransmitPowerLevelTableEntry	RC	Not Tested *(client requirement only)*
M44	10.2.4.1.2	FrequencyInformation	RC	Not Tested *(client requirement only)*
M45	10.2.4.1.2.1	FrequencyHopTable	RC	Not Tested *(client requirement only)*
M46	10.2.4.1.2.1	FixedFrequencyTable	RC	Not Tested *(client requirement only)*
M47	10.2.4.1.3	RFSurveyFrequencyCapabilities	RC	Not Tested *(client requirement only)*
M48	11.1.1	ADD_ROSPEC	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1,
M49	11.1.1	CurrentState set to disabled	C	Not Tested *(client requirement only)*
M50	11.1.1	Error response to CurrentState not disabled	R	7.5.1
M51	11.1.2	ADD_ROSPEC_RESPONSE	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M52	11.1.3	DELETE_ROSPEC	RC	7.6.1, 7.8.1
M53	11.1.3	ROSpecID value of zero indicates to delete all ROSpecs	R	By Design
M54	11.1.4	DELETE_ROSPEC_RESPONSE	RC	7.6.1, 7.8.1
M55	11.1.5	START_ROSPEC	RC	7.6.1, 7.8.1
M56	11.1.6	START_ROSPEC_RESPONSE	RC	7.6.1, 7.8.1, 7.9.1, 7.11.1,
M57	11.1.7	STOP_ROSPEC	RC	7.6.1
M58	11.1.8	STOP_ROSPEC_RESPONSE	RC	7.6.1

M59	11.1.9	ENABLE_ROSPEC	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M60	11.1.9	ROSpecID value of zero indicates to enable all ROSpecs	R	By Design
M61	11.1.10	ENABLE_ROSPEC_RESPONSE	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M62	11.1.11	DISABLE_ROSPEC	RC	7.8.1
M63	11.1.11	ROSpecID value of zero indicates to disable all ROSpecs	R	By Design
M64	11.1.12	DISABLE_ROSPEC_RESPONSE	RC	7.8.1
M65	11.1.13	GET_ROSPECS	RC	7.8.1
M66	11.1.14	GET_ROSPECS_RESPONSE	RC	7.8.1
M67	11.1.14	ROSpecs reported in order they were added	R	By Design
M68	11.2.1	ROSpec	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M69	11.2.1.1	ROBoundarySpec	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M70	11.2.1.1.1	ROSpecStartTrigger	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1,
M71	11.2.1.1.1	PeriodicTriggerValue present when trigger type=2	C	Not Tested *(client requirement only)*
M72	11.2.1.1.1	GPITriggerValue present when trigger type=3	C	Not Tested *(client requirement only)*
M73	11.2.1.1.1.1	UTC not supported, return error	R	By Design
M74	11.2.1.1.1.1	PeriodicTriggerValue	RC	7.13.1
M75	11.2.1.1.1.2	Start trigger timeout value ignored	R	By Design
M76	11.2.1.1.1.2	GPITriggerValue	RC	Not Tested *(optional feature)*
M77	11.2.1.1.1.2	Return zero for numGPIs	RC	Not Tested *(optional feature)*
M78	11.2.1.1.1.2	Send Error message	RC	Not Tested *(optional feature)*
M79	11.2.1.1.2	ROSpecStopTrigger	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1

M80	11.2.1.1.2	GPITriggerValue present when trigger type=2	C	Not Tested *(client requirement only)*
M81	11.2.2	AISpec	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M82	11.2.2.1	AISpecStopTrigger	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M83	11.2.2.1	Support GPITrigger if NumGPIs > 0	R	Not Tested *(optional feature)*
M84	11.2.2.1	Duration trigger ignored when type <> 1	R	By Design
M85	11.2.2.1	GPITrigger present when type=2	C	Not Tested *(client requirement only)*
M86	11.2.2.1	TagObservationTrigger present when type=3	C	Not Tested *(client requirement only)*
M87	11.2.2.1.1	TagObservationTrigger	RC	7.9.1
M88	11.2.2.1.1	NumberOfTags ignored when type<>0	R	By Design
M89	11.2.2.1.1	NumberOfAttempts ignored when type<>2	R	By Design
M90	11.2.2.1.1	T ignored when type <>1	R	By Design
M91	11.2.2.2	InventoryParameterSpec	R	7.6.1, 7.8.1, 7.9.1, 7.10.1,7.11.1, 7.13.1
M92	11.2.3.1	Duration ignored when type <> 1	RC	Not Tested *(optional feature)*
M93	11.2.3.1	Duration greater than zero when type = 1	RC	Not Tested *(optional feature)*
M94	11.2.3.1	N ignored when type <> 2	RC	Not Tested *(optional feature)*
M95	11.2.3.1	N greater than zero when type = 2	RC	Not Tested *(optional feature)*
M96	11.2.4	final parameter in the ListOfSpecs	C	Not Tested *(client requirement only)*
M97	11.2.4	at least one AISpec, RFSurveySpec, or Custom parameter precede	C	Not Tested *(client requirement only)*
M98	11.2.4	LoopSpec	RC	7.7.1
M99	11.2.4	Value of 0 means execution of unlimited ListOfSpecs	RC	7.7.1
M100	12.1.1	ADD_ACCESSSPEC	RC	7.8.1
M101	12.1.1	CurrentState set disabled	C	Not Tested *(client requirement only)*

M102	12.1.1	Error response to CurrentState not disabled	R	7.8.1
M103	12.1.2	ADD_ACCESSSPEC_RESPONSE	RC	7.8.1
M104	12.1.3	DELETE_ACCESSSPEC	RC	7.8.1
M105	12.1.4	DELETE_ACCESSSPEC_RESPONSE	RC	7.8.1
M106	12.1.5	ENABLE_ACCESSSPEC	RC	7.8.1
M107	12.1.5	AccessSpecID value of zero enables all AccessSpecs	R	By Design
M108	12.1.6	ENABLE_ACCESSSPEC_RESPONS E	RC	7.8.1
M109	12.1.7	DISABLE_ACCESSSPEC	RC	7.8.1
M110	12.1.7	AccessSpecID value of zero disables all AccessSpecs	R	By Design
M111	12.1.8	DISABLE_ACCESSSPEC_RESPONS E	RC	7.8.1
M112	12.1.9	GET_ACCESSSPECS	RC	7.8.1
M113	12.1.10	GET_ACCESSSPECS_RESPONSE	RC	7.8.1
M114	12.1.10	AccessSpecs reported in order they were added	R	By Design
M115	12.2.1	AccessSpec	RC	7.8.1
M116	12.2.1.1	AccessSpecStopTrigger	RC	7.8.1
M117	12.2.1.2	Execute first matching AccessSpec	R	By Design
M118	12.2.1.2	Reader stops execution of the AccessSpec after OpSpec fails	R	By Design
M119	12.2.1.2.1	Conditional SHALLs regarding ClientRequestOpSpec parameter	RC	Not Tested *(optional feature)*
M120	12.2.2	Conditional SHALL regarding OpSpec processing order		Not Tested *(optional feature)*
M121	13.1.1	GET_READER_CONFIG	RC	7.6.1
M122	13.1.2	GET_READER_CONFIG_RESPONS E	RC	7.6.1
M123	13.1.2	Return one instance of antenna parameters	RC	7.6.1
M124	13.1.2	Conditional SHALL regarding GPI		Not Tested *(optional feature)*
M125	13.1.3	Configuration value duration	R	By Design
M126	13.1.3	SET_READER_CONFIG	RC	7.5.1
M127	13.1.4	SET_READER_CONFIG_RESPONS E	RC	7.5.1

M128	13.1.5	CLOSE_CONNECTION	RC	7.1.1
M129	13.1.5	Close response	RC	7.1.1
M130	13.1.6	CLOSE_CONNECTION_RESPONSE	RC	7.1.1
M131	13.1.6	Close connection	R	7.1.1
M132	13.2.1	State value change conditions	R	By Design
M133	13.2.1	LLRPConfigurationStateValue	RC	7.6.1
M134	13.2.2	Identification Parameter	RC	7.6.1
M135	13.2.2	MAC address encoding	R	By Design
M136	13.2.3	Conditional SHALLs regarding GPOWriteData	R	Not Tested *(optional feature)*
M137	13.2.4	KeepaliveSpec	RC	7.14.1
M138	13.2.4	No keepalives sent by reader	RC	7.14.1
M139	13.2.6	AntennaConfiguration	RC	7.6.1
M140	13.2.6.1	RFReceiver Parameter	RC	Not Tested *(optional parameter)*
M141	13.2.6.2	RFTransmitterParameter	RC	Not Tested *(optional parameter)*
M142	13.2.6.3	Conditional SHALLs regarding GPIPortCurrentState	RC	Not Tested *(optional feature)*
M143	14	Reporting	R	By Design
M144	14	Reader send new data	R	By Design
M145	14	Duplicate the tag report data	R	By Design
M146	14.1.1	GET_REPORT	RC	7.13.1
M147	14.1.2	RO_ACCESS_REPORT	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M148	14.1.3	KEEPALIVE	RC	7.14.1
M149	14.1.4	KEEPALIVE_ACK	RC	7.14.1
M150	14.1.5	READER_EVENT_NOTIFICATION	RC	7.1.1, 7.3.1, 7.5.1, 7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M151	14.2.1	ROReportSpec	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1

M152	14.2.1	N ignored when trigger type=0	R	By Design
M153	14.2.1.1	TagReportContentSelector	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M154	14.2.2	AccessReportSpec	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M155	14.2.3	TagReportData	RC	7.8.1, 7.9.1
M156	14.2.3	Sticky parameter values	RC	By Design
M157	14.2.3.1	Multiple SHALLs for Accumulation rules	R	By Design
M158	14.2.3.2	EPCData Parameter	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M159	14.2.3.3	ROSpecID Parameter	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M160	14.2.3.4	SpecIndex Parameter	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M161	14.2.3.5	InventoryParameterSpecID Parameter	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M162	14.2.3.6	AntennaID Parameter	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M163	14.2.3.7	PeakRSSI Parameter	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M164	14.2.3.8	ChannelIndex Parameter	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M165	14.2.3.9	FirstSeenTimestampUTC	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M166	14.2.3.10	FirstSeenTimestampUptime	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M167	14.2.3.11	LastSeenTimestampUTC	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M168	14.2.3.12	LastSeenTimestampUptime	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1

M169	14.2.3.13	TagSeenCount > 65535	R	By Design
M170	14.2.3.13	TagSeenCount Parameter	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M171	14.2.3.15	AccessSpecID Parameter	RC	7.8.1,
M172	14.2.5	ReaderEventNotificationSpec	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.13.1
M173	14.2.5.1	EventNotificationState	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.13.1
M174	14.2.6	ReaderEventNotificationData	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.13.1
M175	14.2.6.1	Multiple SHALLs for Event ordering requirements	R	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.13.1
M176	14.2.6.2	HoppingEvent	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.13.1
M177	14.2.6.4	ROSpecEvent	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.13.1
M178	14.2.6.6	Buffer overflow event	R	By Design
M179	14.2.6.9	AISpecEvent	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.13.1
M180	14.2.6.11	ConnectionAttemptEvent	RC	7.1.1
M181	14.2.6.12	ConnectionCloseEvent	R	7.1.1
M182	14.2.6.12	Reader send no additional messages	R	By Design
M183	14.2.6.12	Reader ignore any messages received	R	By Design
M184	14.2.6.13	SpecLoopEvent	RC	7.7.1
M185	14.2.6.13	reader report 4294967295	RC	By Design
M186	15	Reader discards errant message	R	By Design
M187	15	Reader returns at least one error parameter	R	7.5.1
M188	15	Reader response to unsupported message type	R	7.5.1
M189	15	Reader response to unsupported version	R	7.5.1
M190	15	Reader response to ERROR_MESSAGE	R	7.5.1
M191	15.1.1	ERROR_MESSAGE	RC	7.5.1
M192	15.2.1	Multiple SHALLs for LLRP status codes	RC	7.5.1
M193	15.2.2	LLRPStatus Parameter	RC	7.5.1

M194	15.2.2.1	FieldError Parameter	RC	Not Tested *(optional parameter)*
M195	15.2.2.2	ParameterError Parameter	RC	Not Tested *(optional parameter)*
M196	16.2.1.1.1	C1G2LLRPCapabilities	RC	7.3.1
M197	16.2.1.1.1	Readers support at least one filter per query	R	By Design
M198	16.2.1.1.2	UHFC1G2RFModeTable	RC	Not Tested *(optional parameter)*
M199	16.2.1.1.2.1	UHFC1G2RFModeTableEntry	RC	Not Tested *(optional parameter)*
M200	16.2.1.1.2.1	Tari support	R	Not Tested *(optional parameter)*
M201	16.2.1.1.2.1	Spectral Mask Indication	R	Not Tested *(optional parameter)*
M202	16.2.1.1.2.1	EPC HAG T&C Conformance	R	Not Tested *(optional parameter)*
M203	16.2.1.2.1	Tag inventory state-aware behavior	R	Not Tested *(optional feature)*
M204	16.2.1.2.1	C1G2InventoryCommand	RC	7.8.1
M205	16.2.1.2.1.1	C1G2Filter	RC	7.8.1
M206	16.2.1.2.1.1.1	C1G2TagInventoryMask	RC	7.8.1
M207	16.2.1.2.1.1.2	C1G2 TagInventoryStateAware FilterAction	RC	Not Tested *(optional feature)*
M208	16.2.1.2.1.1.3	C1G2TagInventoryStateUnaware FilterAction	RC	7.8.1
M209	16.2.1.2.1.2	C1G2RF Control	RC	7.8.1
M210	16.2.1.2.1.3	Tag inventory state-aware behavior	R	Not Tested *(optional feature)*
M211	16.2.1.2.1.3	C1G2SingulationControl	RC	Not Tested *(optional parameter)*
M212	16.2.1.2.1.3.1	C1G2TagInventoryStateAware SingulationAction	RC	Not Tested *(optional feature)*
M213	16.2.1.3.1	C1G2TagSpec	RC	7.8.1

M214	16.2.1.3.1.1	C1G2TargetTag	RC	7.8.1
M215	16.2.1.3.2.1	C1G2Read	RC	7.8.1
M216	16.2.1.3.2.2	C1G2Write	RC	7.8.1
M217	16.2.1.3.2.3	C1G2Kill	RC	7.15.1
M218	16.2.1.3.2.4	Set CanSupportTagRecommissioning to false	RC	Not Tested *(optional feature)*
M219	16.2.1.3.2.4	Reader return error message	RC	Not Tested *(optional feature)*
M220	16.2.1.3.2.5	C1G2Lock	RC	7.15.1
M221	16.2.1.3.2.5.1	C1G2LockPayload	RC	7.15.1
M222	16.2.1.3.2.6	Conditional SHALL regarding block erase	R	Not Tested *(optional feature)*
M223	16.2.1.3.2.7	Conditional SHALL regarding block write	R	Not Tested *(optional feature)*
M224	16.2.1.3.2.8	Conditional SHALL regarding block permalock	R	Not Tested *(optional feature)*
M225	16.2.1.3.2.9	Conditional SHALL regarding block permalock status	C	Not Tested *(optional feature)*
M226	16.2.1.5.1	C1G2EPCMemorySelector	RC	7.8.1
M227	16.2.1.5.2	C1G2PC	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M228	16.2.1.5.3	C1G2XPCW1	RC	Not Tested *(optional feature)*
M229	16.2.1.5.4	C1G2XPCW2	RC	Not Tested *(optional feature)*
M230	16.2.1.5.5	C1G2CRC	RC	7.6.1, 7.8.1, 7.9.1, 7.10.1, 7.11.1, 7.13.1
M231	16.2.1.5.7.2	C1G2WriteOpSpecResult	RC	7.8.1
M232	16.2.1.5.7.2	Non-zero result	R	By Design
M233	16.2.1.5.7.3	C1G2KillOpSpecResult	RC	7.15.1
M234	16.2.1.5.7.4	C1G2RecommissionOpSpecResult	RC	Not Tested *(optional feature)*
M235	16.2.1.5.7.5	C1G2LockOpSpecResult	RC	7.15.1
M236	16.2.1.5.7.6	Conditional SHALL regarding block erase	R	Not Tested *(optional feature)*
M237	16.2.1.5.7.7	Conditional SHALL regarding block write	R	Not Tested *(optional feature)*

M238	16.2.1.5.7.8	Conditional SHALL regarding block permalock	R	Not Tested *(optional feature)*
M239	16.2.1.5.7.9	Conditional SHALL regarding block permalock status	R	Not Tested *(optional feature)*
M240	17	Multiple SHALLs for Binary encoding rules	RC	By Design
M241	17	Clients ignore reserved bits	C	Not Tested *(client requirement only)*
M242	17	Readers error on non-zero reserved bits	R	7.5.1
M243	17	Both Clients and Readers SHALL set reserved bits to zero	RC	By Design
M244	17.1	Reserve bits set to zero	RC	By Design
M245	17.2.1	TLV and TV encoding	RC	By Design
M246	17.2.1.1	TLV parameter reserved bits	RC	By Design
M247	17.2.7.3	EPCData vs EPC96 parameter	RC	Not Tested *(not enforced)*
M248	18	Reader's transmitter enable restrictions	R	By Design
M249	19	Multiple SHALLs for future versions		Not Tested
M250	20	Reader maintains configuration state	R	By Design
M251	20.1.1	Initiate or accept connections	RC	7.1.1
M252	20.1.1	Reader reply, ConnectionAttemptEvent	RC	7.1.1
M253	20.1.1	Client sends no information until ConnectionAttemptEvent received	C	Not Tested *(client requirement only)*
M254	20.1.2	Reader limits to single connection (momentary connection allowed)	R	7.1.1
M255	20.1.2	Multiple SHALLs for duplicate connection management	R	Not Tested
M256	20.1.3	Version remain constant for the duration of the connection	RC	7.2.1
M257	20.2.1	Conditional SHALL regarding TLS	RC	Not Tested *(optional feature)*
M258	20.2.2	SHALL in INFORMATIVE section	RC	Not Tested *(optional feature)*

6.2 Optional Requirements Matrix

LLRP conformance certification will not test optional requirements.

7 Test Case Requirements

An LLRP Conformance Certification Program will test an Implementation Under Test (IUT) according to predefined test case requirements that have been designed to isolate and test specific features and functions of the LLRP 1.0.1 Specification. While these test case requirements are not exhaustive, they test all the mandatory features that are required by the specification.

For Reader test cases, the IUT can be either a device that includes an embedded implementation of LLRP or it can be a software component implementing LLRP. The testing laboratory is responsible for providing test software that acts as the Client.

For Reader test cases, the term “Reader” refers to the LLRP Reader end-point being tested (either a software component or a hardware device). The term “Send” is an instruction to send a message to the Reader. The term “Receive” indicates that a message should be received from the Reader.

For each test case, in the “Expected Results” column, the term “Verify” is used to indicate a procedure for verifying that a Reader or Client is conformant with one or more requirements. In this same column the term “Confirm” is used to indicate a condition that is perquisite to completing a verification procedure.

In general, test case timing values are parameterized. A certification applicant can submit an IUT with a specification of timing values to be used by the testing laboratory during certification testing. For any timing parameters not specified by the applicant, the testing laboratory will use the default timing values specified by this document. For a complete list of the timing parameters, see Section 8.

The following conventions are used when describing the test cases:

Terminology	Definition
Successful Response	The term successful XXX_RESPONSE is used within the test cases. A successful response is a response containing an LLRPStatusParameter whose StatusCode equals zero (M_Success)
Unsuccessful Response	The term unsuccessful XXX_RESPONSE is used within the test cases. An unsuccessful response is a response containing an LLRPStatusParameter whose StatusCode is not equal to zero.
Basic AISpec	Some tests cases use ROSpecs to cause inventory operations. When the details of the AISpec are not clarified in the test case, the following AISpec parameters will be sent: • AISpecStopTrigger Parameter containing o AISpecStopTriggerType=0 corresponding to a Null stop condition • AntennaIDs: This list will contain a single antennaID of 1. • One InventoryParameterSpecs Parameter o InventoryParameterSpecID = 1 o ProtocolID = 1
Basic AccessSpec	Some test cases use AccessSpecs to cause access operation on tags. When the details of the AccessSpec are not clarified in the test case, the following AccessSpec parameters will be sent: • An AccessSpecID of 1 • An AntennaID of 0 (all) • A ProtocolID of 1 (Gen2) • A Current State of 0 (false) • An ROSpecID of 0 • An AccessSpecStopTrigger Parameter containing o AccessSpecStopTriggerType of 1 o OperationCountValue of 1. • No AccessReportSpec • An AccessCommandOperation (e.g., read, write, kill etc) is to be defined by the test case
TagSpec to match all EPC values	In one or more test cases, a TagSpec to match all EPC values is required. Unless specified by the test case, this TagSpec is a C1G2TagSpecParameter and has the following values: • C1G2TargetTagParameter TagPattern1 containing: o M=1 o Pointer=0 o Length=0 o TagMask=zero length bit array o TagData=zero length bit array o Match=TRUE (1) • No C1G2TargetTagParameter TagPattern2
ROSpec filter to match all EPC values	In one or more test cases, a ROSpec is created with a filter value to match all EPCs. Unless specified by the test case, this filter will take the following form: An InventoryParameterSpec that contains • An InventoryParameterSpecID of 1 • A ProtocolID of 1 (Gen2) • A Single AntennaConfiguration Parameter containing: o AntennaID=0

	o No RFReceiverSettings o No RfTransmitterSettings o A single C1G2InventoryCommand Parameter containing the following ▪ TagInventoryStateAware=False ▪ No C1G2SingulationControl Parameter ▪ No C1G2RFControl Parameter ▪ A single C1G2 Filter Parameter containing: • A single C1G2TagInventoryMask Parameter containing: o MB=1 o Pointer=0 o Length=0 o TagMask=0 zero length bit array • T=0 • A single C1G2TagInventoryStateUnawareFilterAction Parameter containing: o Action=0
OpSpec to write an EPC value	In one or more test cases, an OpSpec is used to write an EPC value into an unlocked tag. Unless specified by the test case, this OpSpec will take the following form: • A C1G2WriteParameter containing o OpSpecID=1 o MB=1 (EPC memory) o WordPtr=1 (skip CRC) o WriteData=0x3000 0000 0000 0000 0000 0000 0000 o AccessPassword=0 (no password required on tag)
OpSpec to read EPC memory	In one or more test cases, an OpSpec is used to read EPC memory from a tag. Unless specified by the test case, this OpSpec will take the following form: • A C1G2ReadParameter containing o OpSpecID=1 o MB=1 (EPC memory) o WordPtr=0 o WordCount=0 (all) o AccessPassword=0 (no password required on tag)
OpSpec to lock write operations on EPC memory	In one or more test cases, an OpSpec is used to lock EPC memory on a tag. Unless specified by the test case, this OpSpec will take the following form: • A C1G2LockParameter containing o OpSpecID=1 o A single C1G2LockPayload parameter containing: ▪ Priviledge=0 (Read/Write Lock) ▪ DataField=2 (EPC memory) o AccessPassword=0x00000001
OpSpec to kill an EPC tag	In one or more test cases, an OpSpec is used to kill a tag. Unless specified by the test case, this OpSpec will take the following form: • A C1G2KillParameter containing o OpSpecID=1 o KillPassword=0x00000001

7.1 Test Case Requirement 1 – TCP Connections

7.1.1 Test Case Requirement 1 – Reader

<table>
<tr><th colspan="3">TCP Connections</th></tr>
<tr><td colspan="3"><b>TPId</b>: TCR-R1</td></tr>
<tr><td colspan="3"><b>Requirement Purpose</b>: This Test Case Requirement confirms TCP connection capabilities of a Reader.</td></tr>
<tr><td colspan="3"><b>Requirements:</b> M3<b>,</b> M4<b>,</b> M128<b>,</b> M129<b>,</b> M130<b>,</b> M131<b>,</b> M150<b>,</b> M180<b>,</b> M181<b>,</b> M251<b>,</b> M252<b>,</b> M254</td></tr>
<tr><td colspan="3"><b>Pre-test conditions</b>:
• No connection established between Reader and Client.</td></tr>
<tr><th>Step</th><th>Step description</th><th>Expected results</th></tr>
<tr><td>1</td><td>Setup the Reader to initiate a connection to a Client. Invoke the Reader to connect.</td><td>Verify that the Reader sends a READER_EVENT_NOTIFICATION message with a ConnectionAttemptEvent parameter with status set to <i>Success</i> (0). Verify that the version number reports LLRP 1.0.1. Confirm that the Client accepts the connection.</td></tr>
<tr><td>2</td><td>Invoke the Client to send GET_READER_CONFIG where RequestedData=7.</td><td>Verify that a successful GET_READER_CONFIG_RESPONSE message is received from the Reader. Record the LLRPConfigurationStateValue reported.</td></tr>
<tr><td>3</td><td>Invoke the Client to connect to the Reader on the same port.</td><td>Verify that the Reader does not establish a second connection.</td></tr>
<tr><td>4</td><td>Invoke the Client to send a CLOSE_CONNECTION.</td><td>Verify that a successful CLOSE_CONNECTION_RESPONSE is received from the reader. Verify that the reader closes the connection in R1 (default=10) seconds without sending any additional data.</td></tr>
<tr><td>5</td><td>Setup the Reader to accept connections from the Client. Invoke the Client to connect to the Reader.</td><td>Confirm that the Reader accepts the connection. Verify that a READER_EVENT_NOTIFICATION message is received from the Reader with a ConnectAttemptEvent parameter where status=0.</td></tr>
<tr><td>6</td><td>Invoke the Client to connect to the Reader on the same port.</td><td>Verify that the Reader does not establish a second connection.</td></tr>
</table>

7	Invoke the Reader to close the connection to the client established in step #4	Verify that the reader sends a READER_EVENT_NOTIFICATION message with ReaderEventNotificationData parameter containing a ConnectionCloseEventParameter. Verify that the connection is closed within R1 (default=10) seconds without sending any additional data.

7.2 Test Case Requirement 2 – Protocol Version Management

7.2.1 TCR-2 Reader

Protocol Version Management		
TPId: TCR-R2		
Requirement Purpose: This Test Case Requirement confirms that the Reader correctly implements version management messages and version negotiation capabilities.		
Requirements: M14, M15, M16, M17, M18, M22, M23, M24, M25, M26, M27		
Pre-test conditions: • An established TCP connection between Reader IUT and Client test software.		
Step	**Step description**	**Expected results**
1	Send GET_SUPPORTED_VERSION with Ver field set to 2.	Verify that a successful GET_SUPPORTED_VERSION_RESPONSE is received with Ver field set to 2. Verify that the message and its parameters are correctly encoded.
2	Send SET_SUPPORTED_VERSION with Ver field set to 2 and ProtocolVersion set to 1.	Verify that a successful SET_SUPPORTED_VERSION_RESPONSE is received with Ver field set to 2 and an LLRPStatus parameter with the StatusCode = 0 (M_Success).
3	Send GET_SUPPORTED_VERSION with Ver field set to 1.	Verify that a successful GET_SUPPORTED_VERSION_RESPONSE is received with Ver field set to 1.
4	Send SET_SUPPORTED_VERSION with Ver field set to 2 and ProtocolVersion set to 2.	Verify that a successful SET_SUPPORTED_VERSION_RESPONSE is received with Ver field set to 2 and an LLRPStatus parameter with the StatusCode != 0 (M_UnexpectedMessage).
5	Restart the Reader IUT.	

6	Send GET_SUPPORTED_VERSION with Ver field set to 2.	Verify that a successful GET_SUPPORTED_VERSION_RESPONSE is received with Ver field set to 2. Record the SupportedVersion field.
7	Send SET_SUPPORTED_VERSION with Ver field set to 2 and ProtocolVersion set to SupportedVersion recorded in step 5.	Verify that a successful SET_SUPPORTED_VERSION_RESPONSE is received with Ver field set to 2 and an LLRPStatus parameter with the StatusCode = 0 (M_Success).
8	Restart the Reader IUT.	
9	Send GET_SUPPORTED_VERSION with Ver field set to 2.	Verify that a successful GET_SUPPORTED_VERSION_RESPONSE is received with Ver field set to 2. Record the SupportedVersion field.
10	Send SET_SUPPORTED_VERSION with Ver field set to 2 and ProtocolVersion set to one greater than SupportedVersion recorded in step 8.	Verify that a successful SET_SUPPORTED_VERSION_RESPONSE is received with Ver field set to 2 and an LLRPStatus parameter with the StatusCode != 0 (M_UnsupportedVersion). Verify that the messge contains no sub-parameter.

7.3 Test Case Requirement 3 – Get Reader Capabilities

7.3.1 TCR-3 Reader

<table>
<tr><th colspan="3">Get Reader Capabilities</th></tr>
<tr><td colspan="3"><b>TPId</b>: TCR-R3</td></tr>
<tr><td colspan="3"><b>Requirement Purpose</b>: This Test Case Requirement confirms that the Reader correctly handles LLRP capabilities messages and responses.</td></tr>
<tr><td colspan="3"><b>Requirements:</b> M28, M29, M30, M32, M34, M35, M36, M37, M38, M40, M150, M196</td></tr>
<tr><td colspan="3"><b>Pre-test conditions</b>:
• An established TCP connection between Reader IUT and Client test software.</td></tr>
<tr><td><b>Step</b></td><td><b>Step description</b></td><td><b>Expected results</b></td></tr>
<tr><td>1</td><td>Send GET_READER_CAPABILITIES where RequestedData=0.</td><td>Verify that a successful GET_READER_CAPABILITIES_RESPONSE is received with all capabilities parameters. Verify that the message and its parameters are correctly encoded.</td></tr>
</table>

7.4 Test Case Requirement 4 – Custom Messages and Custom Parameters

7.4.1 TCR-4 Reader

Custom Messages and Custom Parameters		
TPId: TCR-R4		
Requirement Purpose: This Test Case Requirement confirms the Reader's proper handling of custom messages and custom parameters.		
Requirements: M6, M7, M8		
Pre-test conditions: • An established TCP connection between Reader IUT and Client test software.		
Step	**Step description**	**Expected results**
1	Send a correctly formed custom message unknown to the Reader.	Verify that the Reader responds with an ERROR_MESSAGE containing an LLRPStatusParameter with StatusCode != 0.
2	Send GET_READER_CAPABILITIES where RequestedData=0. Include with this message a correctly formed custom parameter unknown to the Reader.	Confirm that an unsuccessful GET_READER_CAPABILITIES_RESPONSE is received. Verify that the response contains no other parameters (i.e. the requested capabilities).

7.5 Test Case Requirement 5 – Errors

7.5.1 TCR-5 Reader

Errors		
TPId: TCR-R5		
Requirement Purpose: This Test Case Requirement confirms the Reader's proper handling of error conditions.		
Requirements: M50, M126, M127, M150, M187, M188, M189, M190, M191, M192, M193, M242		
Pre-test conditions: • An established TCP connection between Reader IUT and Client test software.		
Step	**Step description**	**Expected results**

1	Send SET_READER_CONFIG with a KeepaliveSpec parameter where KeepaliveTriggerType=2.	Confirm that a SET_READER_CONFIG_RESPONSE is received. Verify that the response includes an LLRPStatus parameter with the StatusCode != 0 (M_Success).
2	Send GET_READER_CAPABILITIES where RequestedData=5.	Confirm that a GET_READER_CAPABILITIES_RESPONSE is received. Verify that the response includes an LLRPStatus parameter with the StatusCode!= 0 (M_Success).
3	Send SET_READER_CONFIG with a LLRPConfigurationStateValue parameter.	Confirm that a SET_READER_CONFIG_RESPONSE is received. Verify that the response includes an LLRPStatus parameter with the StatusCode!= 0 (M_Success).
4	Send SET_READER_CONFIG with no parameters and no fields.	Confirm that a SET_READER_CONFIG_RESPONSE is received. Verify that the response includes an LLRPStatus parameter with the StatusCode!= 0 (M_Success).
5	Send SET_READER_CONFIG with two KeepaliveSpec parameters.	Confirm that SET_READER_CONFIG_RESPONSE is received. Verify that the response includes an LLRPStatus parameter with the StatusCode!= 0 (M_Success).
6	Send SET_READER_CONFIG with an unknown parameter (i.e., parameter type =1000).	Confirm that SET_READER_CONFIG_RESPONSE is received. Verify that the response includes an LLRPStatus parameter with the StatusCode!= 0 (M_Success).
7	Send an unknown message (i.e., message type = 1000).	Confirm that ERROR_MESSAGE is received. Verify that the response includes an LLRPStatus parameter with StatusCode!= 0 (M_Success).
8	Send SET_READER_CONFIG with a KeepaliveSpec parameter where KeepaliveTriggerType=1. Include an Uptime parameter in the KeepaliveSpec parameter.	Confirm that a SET_READER_CONFIG_RESPONSE is received. Verify that the response includes an LLRPStatus parameter with the StatusCode!= 0 (M_Success).
9	Send GET_READER_CONFIG where the LLRP version is reported other than LLRP 1.0.1	Confirm that ERROR_MESSAGE is received with: • the version the same as the received message • a matching message ID • an LLRPStatusParameter with the StatusCode set to M_UnsupportedVersion Verify that this message contains no sub-parameters. Verify that no GET_READER_RESPONSE is received.
10	Send an ERROR_MESSAGE to the Reader.	Verify that the reader does not generate a response.

11	Send a GET_READER_CAPABILITIES with Reserved Bits (Rsvd) set to 1.	Confirm that ERROR_MESSAGE is received.

7.6 Test Case Requirement 6 – Read Operations and Reporting

7.6.1 Test Case Requirement 6 – Reader

<table>
<tr><th colspan="3">Read Operations and Reporting</th></tr>
<tr><td colspan="3"><b>TPId</b>: TCR-R6</td></tr>
<tr><td colspan="3"><b>Requirement Purpose</b>: This Test Case Requirement confirms that the Reader correctly performs read operations and provides requested reports.</td></tr>
<tr><td colspan="3"><b>Requirements:</b> M37, M48, M51, M52, M54, M55, M56, M57, M58, M59, M61, M68, M69, M70, M79, M81, M82, M91, M121, M122, M123, M133, M134, M139, M147, M150, M151, M153, M154, M158, M159, M160, M161, M162, M163, M164, M165, M166, M167, M168, M170, M172, M173, M174, M175, M176, M177, M179, M227, M230</td></tr>
<tr><td colspan="3"><b>Pre-test conditions</b>:
• An established TCP connection between Reader IUT and Client test software.
• One or more UHF Gen2 tags in the field-of-view of the Reader.[2]
• No ROSpecs or AccessSpecs are defined in the Reader.</td></tr>
<tr><td><b>Step</b></td><td><b>Step description</b></td><td><b>Expected results</b></td></tr>
<tr><td>1</td><td>Send SET_READER_CONFIG where the default ROReportSpec and AccessReportSpec are set to report all data values at the end of the ROSpec with N=0. Set the ReaderEventNotificationSpec to enable ROSpec and AISpec event notification (EventType=2 and EventType=6) and disable all other event notifications.</td><td>Confirm that successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.</td></tr>
<tr><td>2</td><td>Send GET_READER_CONFIG where RequestedData=4.</td><td>Confirm that a successful GET_READER_CONFIG_RESPONSE message is received. Verify that default ROReportSpec matches the ROReportSpec set in step #1.</td></tr>
<tr><td>3</td><td>Send GET_READER_CONFIG where RequestedData=6.</td><td>Confirm that a successful GET_READER_CONFIG_RESPONSE message is received. Verify that default AccessReportSpec matches the AccessReportSpec set in step #1.</td></tr>
</table>

[2] The number and location of number of UHF Gen2 tags should be selected such that the Reader IUT is able to execute the ROSpec in the given duration. The Reader IUT may use only one antenna to ensure the execution of the ROSpec in the stipulated duration.

4	Send GET_READER_CONFIG where RequestedData=5.	Confirm that a successful GET_READER_CONFIG_RESPONSE message is received. Verify that default ReaderEventNotificationSpec matches the ReaderEventNotificationSpec set in step #1.
5	Send GET_READER_CONFIG where RequestedData=7.	Confirm that a successful GET_READER_CONFIG_RESPONSE message is received. Record the LLRPConfigurationStateValue reported.
6	Send ADD_ROSPEC with a basic AISpec and null triggers.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received.
7	Send GET_READER_CONFIG where RequestedData=7.	Confirm that successful GET_READER_CONFIG_RESPONSE message is received. Record the LLRPConfigurationStateValue reported. Verify that this value does not match the state value recorded in step #5.
8	Send ENABLE_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #6.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received.
9	Send GET_ROSPECS.	Confirm that successful GET_ROSPECS_RESPONSE message is received. Verify that the ROSpec matches the ROSpec set in step #6 except that the ROSpec state is enabled.
10	Send START_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #6.	Confirm that successful START_ROSPEC_RESPONSE message is received. Record the start time. Confirm that READER_EVENT_NOTIFICATION message for ROSpec start is received.
11	Wait for R6.11 (default=10) seconds.	This wait time is arbitrary so long as the Reader has enough time to complete at least one attempt to read tags.
12	Send STOP_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #6.	Confirm that successful STOP_ROSPEC_RESPONSE message is received. Verify that READER_EVENT_NOTIFICATION messages for AISpec end and ROSpec end events are received in this respective order. Verify that RO_ACCESS_REPORT message is received after the AISpec end event report and before that ROSpec end event report. Verify that these reports are correctly encoded and that the EPC of the tag in the FOV is present.
13	Send DELETE_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #6.	Confirm that successful DELETE_ROSPEC_RESPONSE message is received. Verify that this message is correctly encoded.

14	Send GET_ROSPECS.	Confirm that a successful GET_ROSPECS_RESPONSE message is received. Verify that the ROSpec added in step #6 is not present in the ROSpecs reported.
15	Send GET_READER_CONFIG where RequestedData=7.	Confirm that successful GET_READER_CONFIG_RESPONSE message is received. Record the LLRPConfigurationStateValue reported. Verify that this value does not match the LLRPConfigurationStateValue recorded in step #7.
16	Send GET_READER_CONFIG where RequestedData=0, AntennaID=0, GPIPortNum=0, GPOPortNum=0.	Confirm that successful GET_READER_CONFIG_RESPONSE message is received and contains all mandatory parameters.

7.7 Test Case Requirement 7 – Reader Operations in Loop

7.7.1 Test Case Requirement 7 – Reader

Read Operations in Loop		
TPId: TCR-R7		
Requirement Purpose: This Test Case Requirement confirms that the Reader correctly executes read operations and reporting in loops.		
Requirements: M98, M99, M184		
Pre-test conditions: • An established TCP connection between Reader IUT and Client test software. • One or more UHF Gen2 tags in the field-of-view of the Reader[3]. • No ROSpecs or AccessSpecs are defined in the Reader.		
Step	**Step description**	**Expected results**
1	Send SET_READER_CONFIG where the default ROReportSpec and AccessReportSpec are set to report all data values at the end of the ROSpec with N=0. Set the ReaderEventNotificationSpec to enable SpecLoop event notification (EventType=9) and disable all other event notifications.	Confirm that successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
2	Send ADD_ROSPEC with a basic AISpec and a LoopSpec with LoopCount 1.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received.

[3] The number and location of number of UHF Gen2 tags should be selected such that the Reader IUT is able to execute the ROSpec in the given duration. The Reader IUT may use only one antenna to ensure the execution of the ROSpec in the stipulated duration.

3	Send ENABLE_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #2.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received.
4	Send START_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #2.	Confirm that successful START_ROSPEC_RESPONSE message is received. Record the start time.
5	Wait for R7.5 (default=10) seconds.	This wait time is arbitrary so long as the Reader has enough time to complete one attempt to read tags. Confirm that READER_EVENT_NOTIFICATION message for Spec Loop Event is received with LoopCount 1 and ROSpecID that is sent with ADD_ROSPEC in step #2. Verify that RO_ACCESS_REPORT message is received. Verify that these reports are correctly encoded and that the EPC of the tag(s) in the FOV is present.
6	Send DELETE_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #2.	Confirm that successful DELETE_ROSPEC_RESPONSE message is received.
7	Send ADD_ROSPEC with a basic AISpec and a LoopSpec with LoopCount 0.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received.
8	Send ENABLE_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #7.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received.
9	Send START_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #7.	Confirm that successful START_ROSPEC_RESPONSE message is received. Record the start time.
10	Wait for R7.10 (default=20) seconds.	This wait time is arbitrary so long as the Reader has enough time to complete more than one attempt to read tags. Confirm that READER_EVENT_NOTIFICATION message for Spec Loop Event is received with increasing LoopCount and ROSpecID that is sent with ADD_ROSPEC in step #7.
11	Send STOP_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #7.	Confirm that successful STOP_ROSPEC_RESPONSE message is received. Verify that RO_ACCESS_REPORT message is received. Verify that these reports are correctly encoded and that the EPC of the tag in the FOV is present.
12	Send DELETE_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #7.	Confirm that successful DELETE_ROSPEC_RESPONSE message is received.

7.8 Test Case Requirement 8 – Access Operations and Reporting

7.8.1 Test Case Requirement 8 – Reader

Access Operations and Reporting

TPId: TCR-R8

Requirement Purpose: This Test Case Requirement confirms that the Reader correctly performs reader access operations and provides requested reports.

Requirements: M37, M48, M51, M52, M54, M55, M56, M59, M61, M62, M64, M65, M66, M68, M69, M70, M79, M81, M82, M91, M100, M102, M103, M104, M105, M106, M108, M109, M111, M112, M113, M115, M116, M147, M150, M151, M153, M154, M155, M158, M159, M160, M161, M162, M163, M164, M165, M166, M167, M168, M170, M171, M172, M173, M174, M175, M176, M177, M179, M204, M205, M206, M208, M209, M213, M214, M215, M216, M226, M227, M230, M231

Pre-test conditions:

- An established TCP connection between Reader IUT and Client test software.
- One or more unlocked UHF Gen2 tags in the field-of-view of the Reader.[4]
- No ROSpecs or AccessSpecs are defined in the Reader.

Step	**Step description**	**Expected results**
1	Send SET_READER_CONFIG where the default ROReportSpec and AccessReportSpec are set to report all data values at the end of the ROSpec. Set the ReaderEventNotificationSpec to enable ROSpec and AISpec event notification.	Confirm that a successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
2	Send ADD_ROSPEC with a basic AISpec, no filter and start trigger is set to R8.2a (default=10) second offset time and stop trigger is set to R8.2b (default=10) second duration.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
3	Send ADD_ACCESSSPEC containing a basic AccessSpec with an ROSpecID value of 0. Set the OpSpec to write an EPC value and the TagSpec to match all EPC values. Set execution count =1.	Confirm that a successful ADD_ACCESSSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
4	Send ENABLE_ACCESSSPEC with AccessSpecID from step #3.	Confirm that a successful ENABLE_ACCESS_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.

[4] The number and location of number of UHF Gen2 tags should be selected such that the Reader IUT is able to execute the ROSpec in the given duration. The Reader IUT may use only one antenna to ensure the execution of the ROSpec in the stipulated duration.

5	Send ENABLE_ROSPEC with ROSpecID from step #2.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
6	Wait R8.6 (default=20) seconds for ROSpec to start and stop.	Confirm that READER_EVENT_NOTIFICATION messages for ROSpec start, AISpec end and ROSpec end events are received in this respective order. Confirm that a successful RO_ACCESS_REPORT message is received. Verify that the message and its parameters are correctly encoded. Verify the message includes a TagReportData parameter which includes a C1G2WriteOpSpecResult parameter with result=0.
6a	Send GET_ACCESSSPEC	Confirm that a successful GET_ACCESSSPECS_RESPONSE is received and the spec created in step #3 has been deleted
7	Send DELETE_ROSPEC with ROSpecID from step #2.	Confirm that a successful DELETE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
8	Send ADD_ROSPEC with basic AISpec and filter to match all EPC values. Set the ROSpec start trigger to be periodic every R8.8a (default=10) seconds. Set the stop trigger to duration R8.8b (default=1) second.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
9	Send ADD_ACCESSSPEC with a basic AccessSpec using the ROSpecID from step #8. Set the OpSpec to read EPC memory and the TagSpec to match all EPC values. Set execution count =0.	Confirm that a successful ADD_ACCESSSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
10	Send ENABLE_ACCESSSPEC with AccessSpecID from step #9.	Confirm that a successful ENABLE_ACCESS_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
11	Send ENABLE_ROSPEC with ROSpecID from step #8.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
12	Send START_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #8.	Confirm that a successful START_ROSPEC_RESPONSE message is received. Record the start time.

13	Wait for R8.13 (default=20) seconds.	Confirm that at least one instance of READER_EVENT_NOTIFICATION messages for ROSpec start, AISpec end and ROSpec end events are received in this respective order. Confirm that at least one RO_ACCESS_REPORT message is received. Verify that the message and its parameters are correctly encoded. Verify the message includes a TagReportData parameter which includes a C1G2ReadOpSpecResult parameter with result=0 and EPC value that matches the EPC value in the TagReportData.
14	Send DISABLE_ACCESSSPEC with AccessSpecID from step #9.	Confirm that a successful DISABLE_ACCESS_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
15	Wait for R8.15 (default=20) seconds.	Confirm that at least one instance of READER_EVENT_NOTIFICATION messages for ROSpec start, AISpec end and ROSpec end events are received in this respective order. Confirm that at least one RO_ACCESS_REPORT message is received. Verify that the message and its parameters are correctly encoded. Verify the message includes a TagReportData parameter which does not include a C1G2ReadOpSpecResult parameter.
16	Send DISABLE_ROSPEC with ROSpecID from step #8.	Confirm that a successful DISABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
17	Wait for R8.17 (default=20) seconds.	Confirm that no RO_ACCESS_REPORT messages are received.
18	Send GET_ACCESSSPECS.	Confirm that a successful GET_ACCESSSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded. Verify that the AccessSpec returned matches the AccessSpec created in step #9.
19	Send GET_READER_CONFIG where RequestedData=7.	Confirm that a successful GET_READER_CONFIG_RESPONSE message is received. Record the LLRPConfigurationStateValue reported.
20	Send DELETE_ACCESSSPEC with AccessSpecID from step #9.	Confirm that a successful DELETE_ACCESSSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.

21	Send ADD_ACCESSSPEC using a basic AccessSpec and the ROSpecID from step #8. Set the OpSpec to read EPC memory and the TagSpec to match all EPC values. Set execution count =0	Confirm that a successful ADD_ACCESSSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
22	Send GET_ACCESSSPECS.	Confirm that a successful GET_ACCESSSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded. Verify that the AccessSpec created in step #9 is not reported and verify that the AccessSpec created in step #21 is reported.
23	Send GET_READER_CONFIG where RequestedData=7.	Confirm that a successful GET_READER_CONFIG_RESPONSE message is received. Record the LLRPConfigurationStateValue reported. Verify that this value does not match the LLRPConfigurationStateValue recorded in step #19.

7.9 Test Case Requirement 9 – Tag Observations, Count-based Triggering

7.9.1 Test Case Requirement 9 – Reader

Tag Observations, Count-based Triggering

TPId: TCR-R9

Requirement Purpose: This Test Case Requirement confirms that the Reader correctly performs read operations based upon tag observation, count-based triggering.

Requirements: M48, M51, M56, M59, M61, M68, M69, M70, M79, M81, M82, M87, M91, M147, M150, M151, M153, M154, M155, M158, M159, M160, M161, M162, M163, M164, M165, M166, M167, M168, M170, M172, M173, M174, M175, M176, M177, M179, M227, M230

Pre-test conditions:

- An established TCP connection between Reader IUT and Client test software.
- No tags in the field-of-view of the Reader.
- The Reader is configured without any ROSpecs or AccessSpecs.

Step	Step description	Expected results
1	Send SET_READER_CONFIG where the default ROReportSpec and AccessReportSpec are set to report all data values at the end of the ROSpec. Set the ReaderEventNotificationSpec to enable ROSpec and AISpec event notification.	Confirm that a successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.

2	Send ADD_ROSPEC with a basic AISpec, no filter and triggers=null. Set the AISpec stop trigger tag count=2 with timeout set to R9.2 (default=30) seconds.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
3	Send ENABLE_ROSPEC with ROSpecID from step #2.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
4	Send START_ROSPEC with ROSpecID from step #2.	Confirm that a successful START_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded. Record start time. Confirm that READER_EVENT_NOTIFICATION message for ROSpec start is received.
5	Wait R9.5 (default=40) seconds for AISpec and ROSpec to stop.	Verify that no RO_ACCESS_REPORT message is received or an RO_ACCESS_REPORT containing no TagReportDataParameters is received. Confirm that READER_EVENT_NOTIFICATION messages for AISpec end and ROSpec end events are received in this respective order.
6	Send START_ROSPEC with ROSpecID from step #2.	Confirm that a successful START_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded. Record start time. Confirm that READER_EVENT_NOTIFICATION message for ROSpec start is received.
7	Present two tags to the Reader within R9.7 (default=30) seconds of step #6.	Confirm that READER_EVENT_NOTIFICATION messages for AISpec end and ROSpec end events are received in this respective order. Confirm that one RO_ACCESS_REPORT message is received. Verify that the message and its parameters are correctly encoded. Verify that the sum of the TagSeenCount parameter(s) included in the TagReportData parameter(s) in the RO_ACCESS_REPORT equals 2.
8	Send SET_READER_CONFIG and change the default ROReportSpec to report on every tag (N=1).	Confirm that a successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.

9	Send START_ROSPEC with ROSpecID from step #2.	Confirm that a successful START_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded. Record start time. Confirm that READER_EVENT_NOTIFICATION message for ROSpec start is received.
10	Present two tags to the Reader within R9.10 (default=30) seconds of step #7.	Confirm that READER_EVENT_NOTIFICATION messages for AISpec end and ROSpec end events are received in this respective order. Confirm that two RO_ACCESS_REPORT message are received. Verify that the messages and their parameters are correctly encoded. Verify the messages include only one TagReportData parameter and the TagSeenCount parameter is also 1.

7.10 Test Case Requirement 10 – Immediate Triggering

7.10.1 Test Case Requirement 10 – Reader

<table>
<tr><th colspan="3">Immediate Triggering</th></tr>
<tr><td colspan="3"><b>TPId</b>: TCR-R10</td></tr>
<tr><td colspan="3"><b>Requirement Purpose</b>: This Test Case Requirement confirms that the Reader correctly performs read operations based upon immediate triggering.</td></tr>
<tr><td colspan="3"><b>Requirements:</b> M48, M51, M59, M61, M68, M69, M70, M79, M81, M82, M91, M147, M150, M151, M153, M154, M158, M159, M160, M161, M162, M163, M164, M165, M166, M167, M168, M170, M172, M173, M174, M175, M176, M177, M179, M227, M230</td></tr>
<tr><td colspan="3"><b>Pre-test conditions</b>:
• An established TCP connection between Reader IUT and Client test software.
• One or more tags in the field-of-view of the Reader.[5]
• The Reader is configured without any ROSpecs or AccessSpecs.</td></tr>
<tr><td><b>Step</b></td><td><b>Step description</b></td><td><b>Expected results</b></td></tr>
</table>

[5] The number and location of number of UHF Gen2 tags should be selected such that the Reader IUT is able to execute the ROSpec in the given duration. The Reader IUT may use only one antenna to ensure the execution of the ROSpec in the stipulated duration.

1	Send SET_READER_CONFIG where the default ROReportSpec and AccessReportSpec are set to report all data values at the end of the ROSpec. Set the ReaderEventNotificationSpec to enable ROSpec and AISpec event notification.	Confirm that a successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
2	Send ADD_ROSPEC with no filter, a basic AISpec, start trigger=immediate and stop trigger set to duration of R10.2 (default=5) seconds.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
3	Send ENABLE_ROSPEC with ROSpecID from step #2.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
4	Wait for stop trigger for R10.4 (default=5) seconds.	Confirm that READER_EVENT_NOTIFICATION messages are received for AISpec end events and ROSpec start/stop events. Verify that the messages and their parameters are correctly encoded. Confirm that an RO_ACCESS_REPORT message is received. Verify that the message and its parameters are correctly encoded.

7.11 Test Case Requirement 11 – AISpec Stop Trigger

7.11.1 Test Case Requirement 11 – Reader

AISpec Stop Trigger		
TPId: TCR-R11		
Requirement Purpose: This Test Case Requirement confirms that the Reader correctly performs read operations using AISpec stop triggers based upon tag observations.		
Requirements: M48, M51, M56, M59, M61, M68, M69, M70, M79, M81, M82, M91, M147, M150, M151, M153, M154, M158, M159, M160, M161, M162, M163, M164, M165, M166, M167, M168, M170, M227, M230		
Pre-test conditions: • An established TCP connection between Reader IUT and Client test software. • No tags in the field-of-view of the Reader. • The Reader is configured without any ROSpecs or AccessSpecs.		
Step	**Step description**	**Expected results**

1	Send SET_READER_CONFIG where the default ROReportSpec and AccessReportSpec are set to report all data values on every tag (N=1) or at the end of the ROSpec. Set the ReaderEventNotificationSpec to enable AISpec event notification.	Confirm that a successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
2	Send ADD_ROSPEC with no filter, start/stop triggers=null. Include 1 basic AISpec augmented with stop trigger= tag observation (no tags seen for R11.2a (default=5) seconds / timeout set to R11.2b (default=20) seconds.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
3	Send ENABLE_ROSPEC with ROSpecID from step #2.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
4	Send START_ROSPEC with ROSpecID from step #2.	Confirm that a successful START_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
5	Wait for R11.5 (default=30) seconds.	Confirm that a READER_EVENT_NOTIFICATION message is received after R11.2a seconds. Verify that the message and its parameters are correctly encoded. Verify that no tag data is reported.
6	Send START_ROSPEC with ROSpecID from step #2.	Confirm that a successful START_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
7	Present a tag to the Reader for R11.7a (default=2) seconds and then remove the tag from the Reader's FOV. Wait for another R11.5 (default=30) seconds.	Confirm that one or more RO_ACCESS_REPORT messages are received while the tag is in the reader's FOV, and that the tag is correctly reported. Verify that the message(s) and its parameters are correctly encoded. Verify that after R11.7b (default=7) seconds a READER_EVENT_NOTIFICATION reports the AISpec end event.

7.12 Test Case Requirement 12– Omitted

Test Case 10 was intentionally omitted. This section is left blank.

7.13 Test Case Requirement 13 – Polled Reporting

7.13.1 Test Case Requirement 13 – Reader

Polled Reporting

TPId: TCR-R13

Requirement Purpose: This Test Case Requirement confirms that the Reader correctly handles read operations with polled reporting.

Requirements: M48, M51, M59, M61, M68, M69, M70, M74, M79, M81, M82, M91, M146, M147, M150, M151, M153, M154, M158, M159, M160, M161, M162, M163, M164, M165, M166, M167, M168, M170, M172, M173, M174, M175, M176, M177, M179, M227, M230

Pre-test conditions:

- An established TCP connection between Reader IUT and Client test software.
- One or more tags in the field-of-view of the Reader.[6]
- The Reader is configured without any ROSpecs or AccessSpecs.

Step	Step description	Expected results
1	Send SET_READER_CONFIG where the default ROReportSpec and AccessReportSpec are set to null and to report all parameters.	Confirm that a successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
2	Send ADD_ROSPEC with a basic AISpec, no filter, start trigger=immediate and stop trigger set to a duration of R13.2 (default=5) seconds.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
3	Send ENABLE_ROSPEC with ROSpecID from step #2.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
4	Wait for stop triggers R13.4 (default=30) seconds.	Verify that the messages and their parameters are correctly encoded. Verify that no reports are received prior to the timeout.

[6] The number and location of number of UHF Gen2 tags should be selected such that the Reader IUT is able to execute the ROSpec in the given duration. The Reader IUT may use only one antenna to ensure the execution of the ROSpec in the stipulated duration.

5	Send GET_REPORT	Confirm that an RO_ACCESS_REPORT message is received. Verify that the message and its parameters are correctly encoded.

7.14 Test Case Requirement 14 – Keepalives

7.14.1 Test Case Requirement 14 – Reader

<table>
<tr><th colspan="3">Keepalives</th></tr>
<tr><td colspan="3"><b>TPId</b>: TCR-R14</td></tr>
<tr><td colspan="3"><b>Requirement Purpose</b>: This Test Case Requirement confirms that the Reader correctly handles keepalive processing.</td></tr>
<tr><td colspan="3"><b>Requirements:</b> M137, M138, M148, M149</td></tr>
<tr><td colspan="3"><b>Pre-test conditions</b>:
• An established TCP connection between Reader IUT and Client test software.
• One or more tags in the field-of-view of the Reader.[7]
• The Reader is configured without any ROSpecs or AccessSpecs.</td></tr>
<tr><th>Step</th><th>Step description</th><th>Expected results</th></tr>
<tr><td>1</td><td>Send SET_READER_CONFIG with KeepaliveSpec parameter where KeepaliveTriggerType=1 and it includes the PeriodicTriggerValue parameter where period= R14.1 (default=5) seconds. Disable all ROSpecs, and event notifications.</td><td>Confirm that a successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.</td></tr>
<tr><td>2</td><td>Wait 2xR14.1+1 seconds.</td><td>Confirm that at least two KEEPALIVE messages are received. Verify that the message and its parameters are correctly encoded.</td></tr>
<tr><td>3</td><td>Send SET_READER_CONFIG with KeepaliveSpec parameter where KeepaliveTriggerType=0.</td><td>Confirm that a successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.</td></tr>
<tr><td>4</td><td>Wait 2xR14.1+1 seconds.</td><td>Verify that no KEEPALIVE messages are received.</td></tr>
</table>

[7] The number and location of number of UHF Gen2 tags should be selected such that the Reader IUT is able to execute the ROSpec in the given duration. The Reader IUT may use only one antenna to ensure the execution of the ROSpec in the stipulated duration.

7.15 Test Case Requirement 15 – Lock and Kill Access Operations

7.15.1 Test Case Requirement 15 – Reader

Lock and Kill Access Operations

TPId: TCR-R15

Requirement Purpose: This Test Case Requirement confirms that the Reader correctly handles the OpSpec parameters C1G2Lock and C1G2Kill.

Requirements: M217**,** M220**,** M221**,** M233**,** M235

Pre-test conditions:

- An established TCP connection between Reader IUT and Client test software.
- One unlocked UHF Gen2 tags in the field-of-view of the Reader with access and kill password set to 0x00000001.
- No ROSpecs or AccessSpecs are defined in the Reader.

Step	**Step description**	**Expected results**
1	Send SET_READER_CONFIG where the default ROReportSpec and AccessReportSpec are set to report all data values at the end of the ROSpec	Confirm that a successful SET_READER_CONFIG_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
2	Send ADD_ROSPEC with a basic AISpec, no filter and start trigger is set to R15.2a (default=10) second offset time and stop trigger is set to R15.2b (default=10) second duration.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
3	Send ADD_ACCESSSPEC using the ROSpecID from step #2. Set the OpSpec to write-lock EPC memory and the TagSpec to match all EPC values. Set execution count =1.	Confirm that a successful ADD_ACCESSSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
4	Send ENABLE_ACCESSSPEC with AccessSpecID from step #3.	Confirm that a successful ENABLE_ACCESS_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
5	Send ENABLE_ROSPEC with ROSpecID from step #2.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.

6	Wait R15.6 (default=20) seconds for ROSpec to start and stop.	Confirm that at least one RO_ACCESS_REPORT message is received. Verify that the message and its parameters are correctly encoded. Verify the message includes a TagReportData parameter which includes a C1G2LockOpSpecResult parameter with result=0.
7	Send DELETE_ROSPEC with ROSpecID from step #2.	Confirm that a successful DELETE_ROSPEC message is sent. Verify that the message and its parameters are correctly encoded.
8	Send ADD_ROSPEC with a basic AISpec, no filter and start trigger is set to R15.2a (default=10) second offset time and stop trigger is set to R15.2b (default=10) second duration.	Confirm that a successful ADD_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
9	Send ADD_ACCESSSPEC using the basic AccessSpec and ROSpecID from step #8 to write an EPC value and the TagSpec to match all EPC values. Set execution count =1.	Confirm that a successful ADD_ACCESSSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
10	Send ENABLE_ACCESSSPEC with AccessSpecID from step #9.	Confirm that a successful ENABLE_ACCESS_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
11	Send ENABLE_ROSPEC with ROSpecID from step #8.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
12	Send START_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #8.	Confirm that a successful START_ROSPEC_RESPONSE message is received. Record the start time.
13	Wait for R15.13 (default=20) seconds.	Confirm that at least one RO_ACCESS_REPORT message is received. Verify that the message and its parameters are correctly encoded. Verify the message includes a TagReportData parameter which includes a C1G2WriteOpSpecResult parameter with result != 0.
14	Send DISABLE_ROSPEC with ROSpecID from step #8.	Confirm that a successful DISABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
15	Send ADD_ACCESSSPEC using the ROSpecID from step #8. Set the OpSpec to kill the EPC tags and the TagSpec to match all EPC values. Set execution count =1.	Confirm that a successful ADD_ACCESSSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.

16	Send ENABLE_ACCESSSPEC with AccessSpecID from step #17	Confirm that a successful ENABLE_ACCESS_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
17	Send ENABLE_ROSPEC with ROSpecID from step #8.	Confirm that a successful ENABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
18	Send START_ROSPEC where ROSpecID is that sent with ADD_ROSPEC in step #8.	Confirm that a successful START_ROSPEC_RESPONSE message is received. Record the start time.
19	Wait for R15.19 (default=20) seconds.	Confirm that at least one RO_ACCESS_REPORT message is received. Verify that the message and its parameters are correctly encoded. Verify the message includes a TagReportData parameter which includes a C1G2KillOpSpecResult parameter with result = 0 (Success). Record the EPC (s) of the tag(s) killed.
20	Send DISABLE_ROSPEC with ROSpecID from step #8.	Confirm that a successful DISABLE_ROSPEC_RESPONSE message is received. Verify that the message and its parameters are correctly encoded.
21	Send DELETE_ROSPEC with ROSpecID from step #8	Confirm that a successful DELETE_ROSPEC message is sent. Verify that the message and its parameters are correctly encoded.
22	Repeat TCR-R11.	Verify that either 1) the tag reported killed in TCR-R13 step 19 is not reported in the RO_ACCESS_REPORT message resulting from TCR-R11 step 5, or 2) no RO_ACCESS_REPORT is returned by the Reader resulting from TCR-R11 step 5.

8 Default timeout values

The following default values will be used for testing unless a table with alternate values is provided with the IUT.

Identifier	Reference	Default Value	Description
R1	7.1.1	10 seconds	Used for allowing time for the reader to close a connection
R6.11	7.6.1	10 seconds	Used to allow the Reader time to

			singulate tags
R7.5	7.7.1	10 seconds	Used to allow the Reader time to singulate tags
R7.10	7.7.1	20 seconds	Used to allow the Reader time to singulate tags for at least twice
R8.2a	7.8.1	10 seconds	Used to set the offset for an ROSpec start trigger
R8.2b	7.8.1	10 seconds	Used to set the duration for an ROSpec Inventory event
R8.6	7.8.1	20 seconds	The time for the ROSpec to start and complete. Should be as long or longer than R8.2a+ R8.2b
R8.8a	7.8.1	10 seconds	Value for the periodic start trigger rate in seconds
R8.8b	7.8.1	1 second	Length of the periodic RoSpec duration in seconds
R8.13	7.8.1	20 seconds	Length to wait for reader to receive access report
R8.15	7.8.1	20 seconds	Length to wait for reader to receive access report
R8.17	7.8.1	20 seconds	Length to wait for reader to not receive access report
R9.2	7.9.1	30 seconds	Duration of AISpec for Tag count based reporting
R9.5	7.9.1	40 seconds	Time to wait for AISpec to stop.
R9.7	7.9.1	30 seconds	Max time to wait before placing tags in the field.
R9.10	7.9.1	30 seconds	Max time to wait before placing tags in the field.
R10.2	7.10.1	5 seconds	Duration of the ROSpec
R10.4	7.10.1	5 seconds	Duration to wait for ROSpec stop trigger.
R11.2a	7.11.1	5 seconds	No tags seen for timeout
R11.2b	7.11.1	20 seconds	ROSpec timeout
R11.5	7.11.1	30 seconds	Verification timeout.

R11.7a	7.11.1	2 seconds	Tag time in field of view
R11.7b	7.11.1	7 seconds	Max time before RO Report is received. Should be as long as R11.2a + R11.7a
R13.2	7.13.1	5 seconds	Stop trigger duration
R13.4	7.13.1	30 seconds	Wait for stop trigger timeout
R14.1	7.14.1	5 seconds	Keepalive configuration time
R15.2a	7.15.1	10 seconds	RoSpec start trigger offset
R15.2b	7.15.1	10 seconds	ROSpec duration
R15.6	7.15.1	20 seconds	Wait for ROSpec to complete
R15.13	7.15.1	20 seconds	Wait for ROSpec to complete
R15.19	7.15.1	20 seconds	Wait for ROSpec to complete

9 References

[W3C-Conformance] D. Dardailler, “Conformance Testing and Certification Model for W3C Specifications,” W3C Note, http://www.w3.org/QA/2002/01/Note-qa-certif-20020102.html, January 2002.

[LLRP-Spec] EPCglobal, The Low Level Reader Protocol (LLRP) Specification, Version 1.1, Candidate Specification, May 27, 2010.

10 Acknowledgement of Contributors and of Companies Opt’d-in during the Creation of this Standard (non-normative)

Disclaimer

Whilst every effort has been made to ensure that this document and the information contained herein are correct, EPCglobal and any other party involved in the creation of the document hereby state that the document is provided on an “as is” basis without warranty, either expressed or implied, including but not limited to any warranty that the use of the information herein with not infringe any rights, of accuracy or fitness for purpose, and hereby disclaim any liability, direct or indirect, for damages or loss relating to the use of the document.

Below is a list of active participants and contributors in the development of the LLRP 1.1 Conformance Requirements. This list does not acknowledge those who only monitored the process without contributing or those who chose not to have their name listed here.

An “active participant” for the purpose of this list is an individual who corresponded using the Working Group mailing list or who attended one or more face-to-face or teleconference meetings of the Working Group.

Name	Company	Key Role
Delaney, Chris	Impinj	Editor
Frey, Mark	GS1 EPCglobal, Inc.	Facilitator/Process Manager
Gangl, Gerhard	7iD (formerly EOSS GmbH)	
Molina, Victor	Symbol	Co-Chair

The following list enumerates, in alphabetical order by company name, all companies that signed the EPCglobal IP Policy and the opt-in agreement for the EPCglobal Working Group that created the LLRP 1.1 standard and its associated Conformance Requirements.

7iD (formerly EOSS GmbH)
Afilias Limited
Auto-ID Labs - ICU
CAEN RFID SRL
GS1 Australia
GS1 China
GS1 EPCglobal, Inc.
GS1 Germany (CCG)
GS1 Global Office
GS1 Taiwan (EAN)
IBM Corporation
Impinj, Inc.
Invengo Information Technology Co.,Ltd.
LIT (Institute of Logistics Information Technology)
Lyngsoe Systems
MET Laboratories
Psion Teklogix Inc.
R R Indústria E Comércio De Etiquetas Ltda
Supply Insight, Inc.
Symbol Technologies Inc, a Motorola Co.
TagSys
Tibco Software, Inc