The Global Language of Business

GS1 Low Level Reader Protocol (LLRP) Standard

The interface protocol between RFID Readers and Clients called low-level as it provides control of RFID air protocol operation timing and access to air protocol command parameters.

Release 2.0, Ratified, Jan 2021

Document Summary

Document Item	Current Value
Document Name	GS1 Low Level Reader Protocol (LLRP) Standard
Document Date	Jan 2021
Document Version	2.0
Document Issue	
Document Status	Ratified
Document Description	The interface protocol between RFID Readers and Clients called low-level as it provides control of RFID air protocol operation timing and access to air protocol command parameters.

Contributors

First Name	Last Name	Organisation
Koji	Asano	GS1 Japan
Mahdi	Barati	GS1 Iran
Xavier	Barras	GS1 France
Benjamin	Bekritsky	Zebra Technologies Corporation
Tony	Ceder	Charmingtrim
Henk	Dannenberg	NXP Semiconductors
Jeanne	Duckett	Avery Dennison RFID
Hussam	El-Leithy	GS1 US
Thomas	Frederick	Clairvoyant Technology LLC (Co-Chair)
Guido	Freijomil	GS1 Argentina
Nicole	Golestani	GS1 Canada
James	Goodland	NXP Semiconductors
Danny	Haak	Nedap
Steve	Halliday	RAIN RFID Alliance
Roula	Karam	Antares Vision
Steven	Keddie	GS1 Global Office
Kimmo	Keravuori	GS1 Finland
Martin	Kirisits	7iD Technologies GmbH
Michael	Koch	Zebra Technologies Corporation (Co-Chair)
Blair	Korman	Johnson & Johnson
Zhimin	Li	GS1 China
Noriyuki	Mama	GS1 Japan
Scott	McMillan	Clairvoyant Technology LLC
Zubair	Nazir	GS1 Canada
Alice	Nguyen	GS1 Vietnam
Falk	Nieder	European EPC Competence Center GmbH (EECC)

First Name	Last Name	Organisation
Geoff	O'Connell	GS1 UK
Thiago	Perez Rojas	GS1 Argentina
Bijoy	Peter	GS1 India
Neil	Piper	GS1 Global Office
Josef	Preishuber-Pflugl	CISC Semiconductor GmbH
Albertus	Pretorius	LicenSys Pty Ltd
Craig Alan	Repec	GS1 Global Office
J. John	RYU	GS1 Global Office (Facilitator)
Yuki	Sato	GS1 Japan
Klaus	Schmitt	Robert Bosch GmbH
Georg	Schwering	European EPC Competence Center GmbH (EECC)
Jim	Springer	EM Microelectronic
Matt	Stanton	Impinj, Inc
Oleg	Tanchuk	Quake Global
Claude	Tetelin	GS1 Global Office (Co-Lead Editor)
Ricardo	Verza Amaral Melo	GS1 Brasil
Thorsten	Vogel	race result AG
Amber	Walls	GS1 US
Sajan	Wilfred	Zebra Technologies Corporation (Co-Lead Editor)
Owen	Williams	Clairvoyant Technology LLC
Roman	Winter	GS1 Germany
Connie	Wong	GS1 Canada
Ruoyun	Yan	GS1 China

Log of Changes

Release	Date of Change	Changed By	Summary of Change
1.0	Oct 2010		EPCglobal edition
2.0	Dec 2020	LLRP MSWG	eBallot Version WR 19-274. Mission Specific Work Group which undertook a review and update based on prototype testing

Disclaimer

Table of Contents

Audience for this document

The target audience for this specification includes:

- RFID Network Infrastructure vendors
- Reader vendors
- EPC Middleware vendors
- System integrators

1 (Informative) Glossary

This section provides a non-normative summary of terms used within this specification.

Term	Meaning
Access	The operation of communicating with (reading from and/or writing to) a Tag. An individual Tag must be uniquely identified prior to access. Access comprises multiple commands, some of which employ one-time-pad based cover- coding of the R=>T link.
Air Interface	The complete communication link between a Reader and a Tag including the physical layer, collision arbitration algorithm, command and response structure, and data-coding methodology.
Antenna	An atomic, specifically-addressable RF transmission and/or reception device used for communication with RFID tags. For the purposes of this spec, multiplicity of antenna is going to be referred to as antennas.
Capabilities	The set of intrinsic Reader properties relevant to protocol operation. This may include physical, functional, or protocol support information.
Compatibility	A general term used to describe a consistency of terminology and/or operation between one or more specifications and/or implementations. It is specifically not intended to be used to define expectations on protocol operation. The proper term for this is 'Interoperability' as defined below.
Configuration	Data and parameters to control specific operation of a Reader or the Client that is typically instantiated at boot time or as a result of specific management actions on a timescale much greater than the operations of LLRP. It is possible that certain parameters may be controlled via LLRP and have corresponding default configuration parameters.
Client	From the perspective of LLRP, a *Client* is synonymous with *Controller* (see below). The specification uses the term Client to identify the endpoint opposite to the Reader.
Controller	The function that implements the Reader Protocol (Interface) opposite the Reader (i.e., an LLRP endpoint). In the GS1 System Architecture, this function could comprise part of the Filtering & Collection (Role), but it may be implemented in a wide range of devices, including dedicated RFID infrastructure, Readers, and middleware running on server hardware.
GPI	General purpose input
GPO	General purpose output
Interference	There are two types of interference that impact a RFID system's operating capacity: Reader-to-tag, and Reader-to- Reader. Reader-to-tag interference happens when a tag receives signals of comparable strengths from more than one Reader at the same time. This causes the tag to respond arbitrarily to the Readers, and makes its state unpredictable. Reader-to-Reader interference happens when a Reader in the midst of listening to a tag's reply at a particular frequency, receives signals much stronger than the tag's reply, from another Reader operating at the same frequency at the same time. This causes the Reader's receiver logic to not be able to correctly decode the tag's reply. Both these interference scenarios can potentially degrade the system performance.
Interoperability	The ability for two implementations of protocol endpoints to properly function with each other. Proper function may require negotiation of supported capabilities between the two endpoints.
Interrogator	Synonymous with Reader. The EPCglobal Class-1 Gen-2 air protocol specification refers to Readers as Interrogators. However, since the term Reader is included in the title of this specification *Low Level Reader Protocol*, the term Reader is used instead of Interrogator.

Term	Meaning
Inventory	The operation of identifying Tags. A Reader begins an inventory round by transmitting a Query command (Query starts the round) in one of four sessions. One or more Tags may reply. The Reader detects a single Tag reply and requests the PC, EPC, and CRC-16 from the Tag. Inventory comprises multiple commands. An inventory round operates in one and only one session (defined below) at a time.
LLRP	Low Level Reader Protocol
LLRP connection	Instance of LLRP between the Reader and the Client.
LLRP endpoint	The endpoints of a LLRP instance (i.e., either a Reader or a Client).
Q	A parameter that a Reader uses to regulate the probability of Tag response. A Reader commands Tags in an inventory round to load a Q-bit random (or pseudo-random) number into their slot counter; the Reader may also command Tags to decrement their slot counter. Tags reply when the value in their slot counter is zero. Q is an integer in the range (0,15); the corresponding Tag-response probabilities range from $2^0 = 1$ to $2^{-15} = 0.000031$.
Q algorithm	A collision-arbitration algorithm where Tags load a random (or pseudo-random) number into a slot counter, decrement this slot counter based on Reader commands, and reply to the Reader when their slot counter reaches zero.
Reader	The function that implements the RFID Reader (Role) in the GS1 System Architecture. It is one of the two endpoints of the Reader Protocol (Interface) which is, for the purposes of this specification, LLRP. The Reader comprises of one or more antennas which are used to communicate with RFID tags. Note that a Reader can not only read RFID tags, it can perform other operations on tags such as write and kill.
Receive Sensitivity	Receiver sensitivity is a measure of the weakest tag signal an RFID reader is able to detect and demodulate. Changing this affects the minimum detectable signal (MDS) so as to prevent weaker responses from tying up the receiver. The other commonly used term for such a control is squelch.
Select	The operation of choosing a tag population for inventory and access. A Select command may be applied successively to select a particular Tag population based on user-specified criteria. This operation is analogous to selecting records from a database.
Session	An inventory process comprising a Reader and an associated Tag population. A Reader chooses one of four sessions and inventories Tags within that session. The Reader and associated Tag population operate in one and only one session for the duration of an inventory round. For each session, Tags maintain a corresponding inventoried flag. Sessions allow Tags to keep track of their inventoried status separately for each of four possible time-interleaved inventory processes, using an independent inventoried flag for each process.
Singulation	Identifying an individual Tag in a multiple-Tag environment.
Spec	The document uses the term 'Spec' to denote the parameter specification for an operation.
UTC	Coordinated Universal Time (UTC) is the international time standard as maintained by the Bureau International des Poids et Mesures (BIPM).

2 Introduction

This document specifies an interface between RFID Readers and Clients. The design of this interface recognises that in some RFID systems, there is a requirement for explicit knowledge of RFID air protocols and the ability to control Readers that implement RFID air protocol communications. It also recognises that coupling control to the physical layers of an RFID infrastructure may be useful for the purpose of mitigating RFID interference. The interface described herein, and the functionality it implies, is called "Low Level Reader Protocol," or LLRP.

Following are the responsibilities of this interface:

- Provide means to command an RFID Reader to inventory tags (read the EPC codes carried on tags), read tags (read other data on the tags apart from the EPC code), write tags, and execute other protocol-dependent access commands (such as 'kill' and 'lock' from EPCglobal Class 1 Generation 2).
- Provide means for robust status reporting and error handling during tag access operations.

- Provide means for conveying tag passwords necessary to effect commands that may require them, such as the 'Kill' command in the EPCglobal Class 1 Generation 2 UHF Air Interface Protocol.
- Provide means to control the forward and reverse RF link operation to manage RF power levels and spectrum utilisation, and assess RF interference, among RFID Readers in a system.
- Provide means to control aspects of Tag Protocol operation, including protocol parameters and singulation parameters.
- Provide means to facilitate the addition of support for new air protocols.
- Provide means for the retrieval of Reader device capabilities.
- Provide means for vendors of Reader devices to define vendor-specific extensions to the protocol in a manner that is non-interfering among vendors, and which, to the extent possible, is vendor-administered.

In addition LLRP is "regulatory requirements-aware," such that its functions are applicable in regulatory jurisdictions worldwide.

The overall organisation of this specification is as follows: - General Overview (sections 3-6); Abstract Model (sections 1-16, 18), which describes the protocol, its message types and contents without specifying the protocol syntax; Binary Encoding (section *16*), which specifies the syntax for representing the abstract protocol; Transport Binding (section *19*), which specifies the mechanism for delivery of protocol messages; Informative Descriptions (sections 1-19). Guidelines for adding support of a new air protocol to LLRP are presented in section *16.1*.

3 Role within the GS1 System Architecture

The RFID infrastructure part of the "Share" layer of the GS1 System Architecture, consists of network that participates in the management (e.g., read/write/lock) and transmission of tag data. Consumers of tag data (e.g., end-user applications) are considered client network elements. Network elements between the tag and the clients form the conduit to transport tag data over the network to the applications and convey tag operational commands over the network to the tags. Elements relevant to the LLRP specification are tags and readers, supported by the filtering and collection.

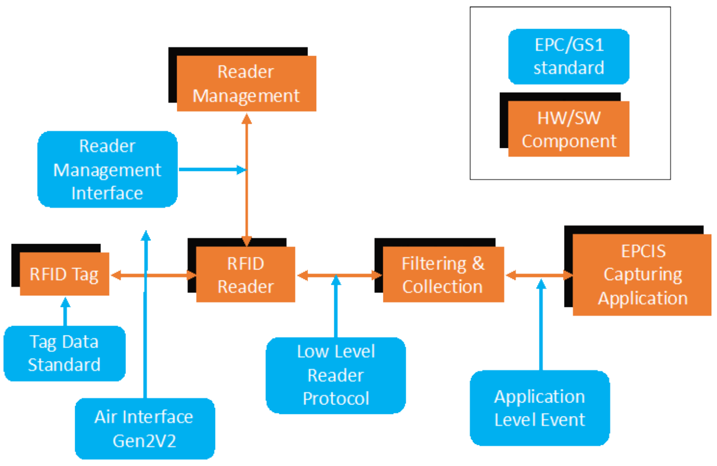

Figure 1: LLRP in the GS1 System Architecture

Figure 1 illustrates the position of LLRP in the "Share" layer of the GS1 System Architecture between filtering & collection and the RFID reader.

The responsibilities of the elements and interfaces below filtering and collection can be classified into three broad functional groups: tag data processing (*data path*), reader device management (*management path*) and reader control and coordination (*control path*).

LLRP facilitates reader control and coordination by exposing air protocol relevant control knobs to filtering and collection.

The physical and logical requirements for the communication between the reader and the tag are defined by the air protocol. Specifically, the air protocol defines the signalling layer of the communication link, reader and tag operating procedures and commands, and the collision arbitration (or “singulation”) scheme to identify a specific tag in a multiple-tag environment. One such air protocol is GS1's EPC “Gen2” UHF standard. EPC/RFID tag memory is logically separated into four distinct banks: reserved memory (MB 00), EPC memory (MB 01), TID memory (MB 10) and User memory (MB 11). The physical memory map of a tag is vendor-specific. The air protocol commands that access memory have a parameter that selects the bank, and an address parameter to select a particular memory location within that bank.

The fundamental operations a reader performs on a tag population are inventory and access. Inventory is the operation of identifying tags and comprises multiple air protocol commands. Using the singulation scheme, the reader detects a single tag reply and requests the EPC memory contents from that tag. Access is the operation of communicating with (reading from and/or writing to) a tag. An individual tag must be uniquely identified prior to access. Similar to the inventory operation, access comprises multiple air protocol commands. In addition, a reader can choose a subset of a tag population for inventory and access. This is referred to as a select operation in the air interface protocol. The select operation is used to pre-select and/or de-select a particular tag population for the subsequent inventory and/or access operation. This helps focus the operations on the desired subset of tags, and also thins the tag population participating in the singulation operation, thereby improving the overall singulation rate.

It is anticipated that overall system performance may be optimised by tuning the RF, singulation and air protocol parameters within and across readers. Performance can be further optimised if the tuning is done cognizant of the RF environment in the vicinity of the reader.

The LLRP interface between the client and the reader facilitates the management of reader devices to mitigate reader-to-tag and reader-to-reader interference and maximize the efficiency of singulation and data operations over the tag population. This is achieved by enabling reader operation at the full performance level of the air interface. In addition, LLRP provides the interface to transport the results of RF monitoring (i.e., RF survey) if the reader device is capable of performing that function.

Other applications may perform additional operations on EPC/ RFID tag data. Operations may range from capturing EPCs to performing other tag access operations exposed by the air interface protocol, including read, write, kill, lock, etc. Multiple application requirements translate into a set of access operations that a reader or a set of readers performs on tags as and when they are in the field of view. The LLRP interface provides a scalable mechanism to manage the access operations at the reader devices.

Finally, scalable device management capabilities are critical for operations of a large network of reader devices. The LLRP interface facilitates device status and error reporting, as well as discovery of device capabilities.

4 Terminology and Typographical Conventions

Within this specification, the terms SHALL, SHALL NOT, SHOULD, SHOULD NOT, MAY, NEED NOT, CAN, and CANNOT are to be interpreted as specified in Annex G of the ISO/IEC Directives, Part 2, 2001, 4th edition [ISODir2]. When used in this way, these terms will always be shown in ALL CAPS; when these words appear in ordinary typeface they are intended to have their ordinary English meaning. However in this document only a subset of the terms listed above SHALL be used. The subset of acceptable terms includes the following: SHALL, SHALL NOT and MAY. The terms SHOULD, SHOULD NOT, NEED NOT, CAN, and CANNOT, SHALL NOT be used.

All sections of this document, with the exception of section 1-3, are normative, except where explicitly noted as non-normative. All figures within the document are non- normative unless otherwise specified.

The following typographical conventions are used throughout the document: ALL CAPS type is used for the special terms from [ISODir2] enumerated above. ALL_CAPS_UNDERSCORE type is used for LLRP message names.

CamelBackType is used for LLRP parameter and data field names.

`Monospace` type is used to denote programming language, UML, and XML identifiers, as well as for the text of XML documents.

5 Overview of LLRP

LLRP is specifically concerned with providing the formats and procedures of communications between a Client and a Reader. The LLRP protocol data units are called messages. Messages from the Client to the Reader include getting and setting configuration of Readers, capabilities discovery of Readers and managing the inventory and access operations at the Readers. Messages from the Reader to the Client include the reporting of Reader status, RF survey, and inventory and access results.

LLRP is an application layer protocol and does not provide retransmission, or reordering facilities. State consistency between the Client and the Reader is critical for the correct functioning of the system. Using LLRP messages, the Client updates the Reader state which includes Reader configuration parameters, dynamically created data structures (e.g., ROSpecs, AccessSpecs, etc), and possibly vendor defined data. For this reason, LLRP requires acknowledgements for the Client to Reader transactions – this provides a fail-safe mechanism at the LLRP layer to cope with network error situations. Also, to cope with intermittent connections, a Client can request a Reader's configuration state to confirm that a Reader's state is consistent with the Client after the Client reconnects (see LLRPConfigurationStateValue in section *13.2.1*). The Reader to Client messages are primarily reports, status notifications or keepalives. Only the keepalives are acknowledged by the Client.

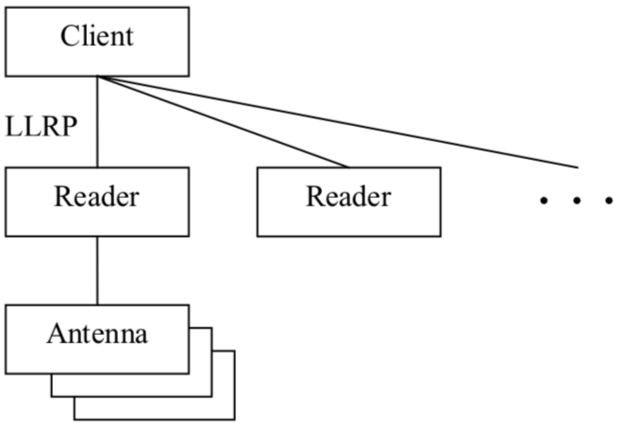

Figure 2: LLRP Endpoints

As shown in Figure 2, from LLRP's perspective, a Reader contains a collection of one or more antennas. Moreover, Readers as used in this specification may not necessarily be in one-to-one correspondence with hardware devices.

5.1 Typical LLRP Timeline

The typical timeline of a LLRP connection has two phases: version negotiation, and runtime.

5.1.1 Version Negotiation

There are multiple documented versions of LLRP, and a Client and a Reader must negotiate a mutually supported protocol version for each connection. When a LLRP connection is first established, both Client and Reader assume LLRP version 1. With the exception of the version negotiation messages themselves, only version 1 messages are sent by the Client and Reader until the version negotiation process is complete. Version negotiation is performed using version 2 messages and is initiated by the Client. The Client requests the supported protocol version from the Reader, determines an appropriate version for the connection, and instructs the reader as to the protocol to use. Once a version has been established using this procedure, all subsequent messages are sent using the negotiated protocol version. The details of this procedure can be found in section 9, with detailed message sequence charts found in section *19.1.3*

A typical version negotiation timeline between a Client and a Reader is depicted in Figure 3.

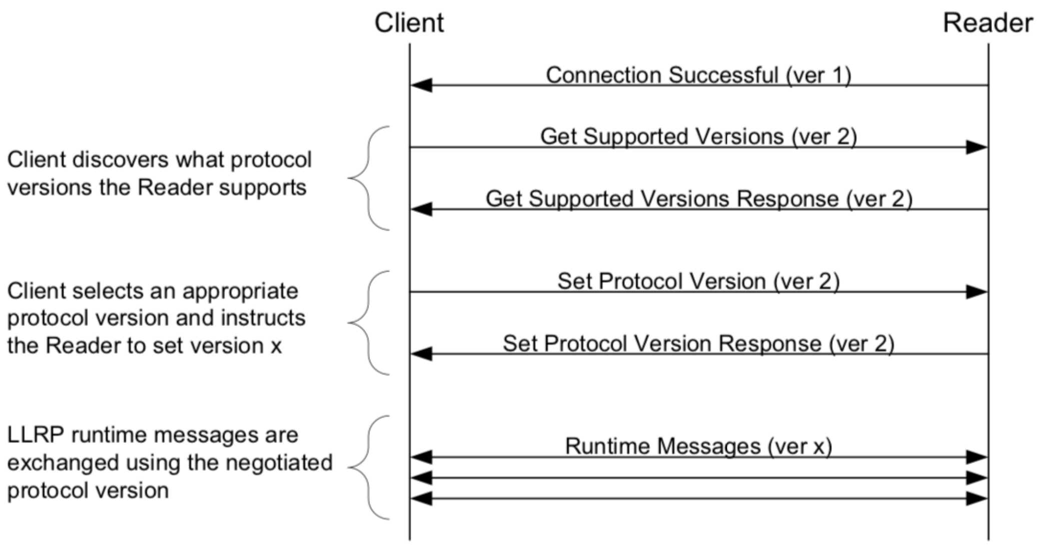
Client
Reader
Connection Successful (ver 1)
Get Supported Versions (ver 2)
Get Supported Versions Response (ver 2)
Client discovers what protocol versions the Reader supports
Set Protocol Version (ver 2)
Set Protocol Version Response (ver 2)
Client selects an appropriate protocol version and instructs the Reader to set version x
Runtime Messages (ver x)
LLRP runtime messages are exchanged using the negotiated protocol version

Figure 3: Typical LLRP Version Negotiation Timeline

5.1.2 Runtime Operation

LLRP runtime operation consists of the following phases of execution:

- Capability discovery
- Device configuration
- [optional] Inventory and access operations setup
- Inventory cycles executed

If tag conditions matched, access operations will be executed during inventory cycle execution. Access operations include reading, writing, and locking tag memory, killing tags, etc.

- RF Survey operations executed
- Reports returned to the Client

A typical timeline of both LLRP and air protocol interactions between a Client, a Reader and a population of tags is depicted in Figure 4.

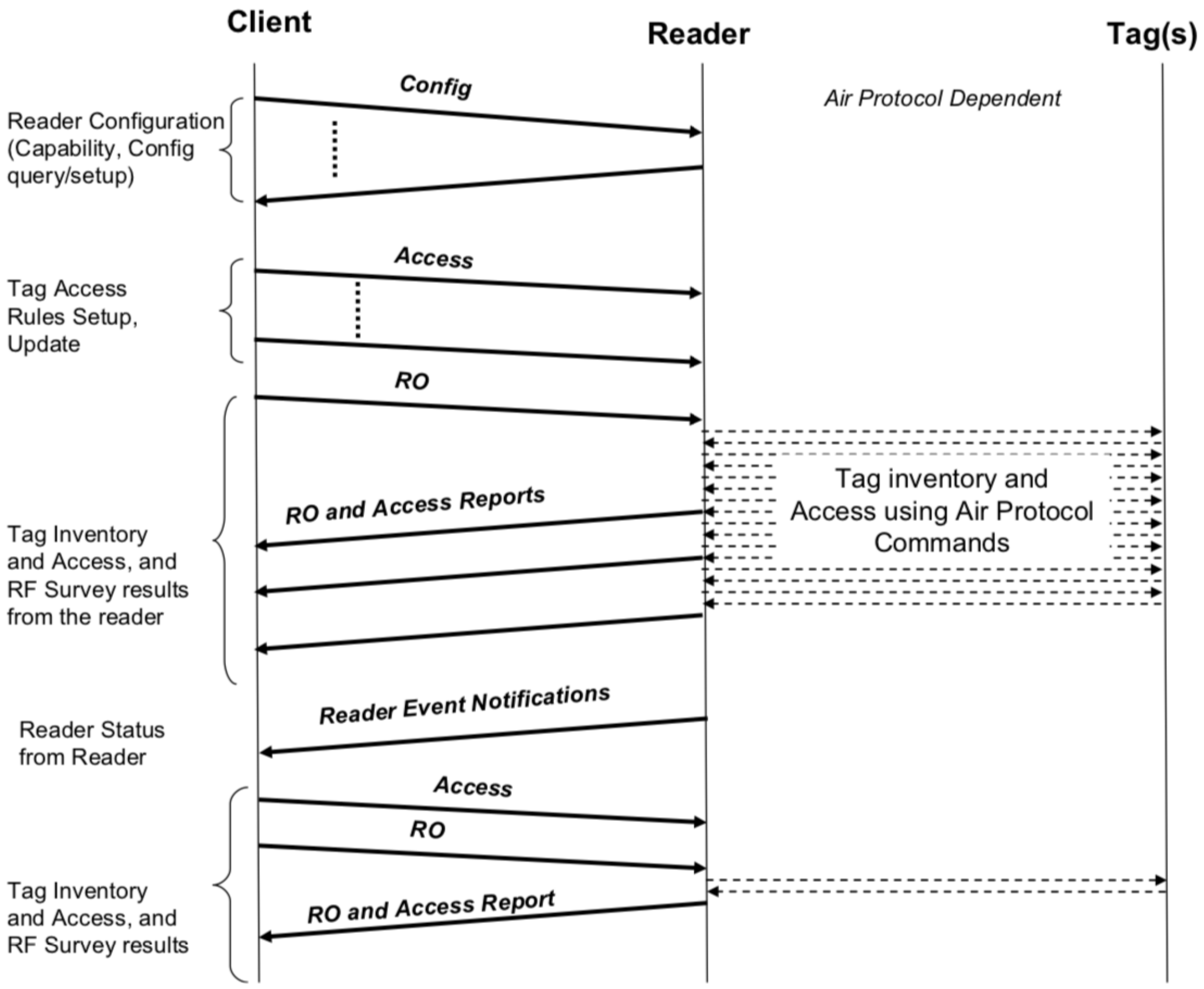
Client
Reader
Tag(s)
Config
Air Protocol Dependent
Reader Configuration (Capability, Config query/setup)
Access
Tag Access Rules Setup, Update
RO
RO and Access Reports
Tag inventory and Access using Air Protocol Commands
Tag Inventory and Access, and RF Survey results from the reader
Reader Event Notifications
Reader Status from Reader
Access
RO
RO and Access Report
Tag Inventory and Access, and RF Survey results

Figure 4: Typical LLRP Runtime Timeline

6 LLRP Operation

LLRP uses protocol data units called messages (See section *7.1.2* for details) to communicate between the Client and the Reader. Using LLRP, the Client updates or retrieves configuration of the Reader; by doing so, it controls the Reader's operation. This section provides an overview of the abstract model of the LLRP interface, and the data structures used in LLRP to and from the Reader.

Section *1* presents an informative description of the LLRP object model based upon UML notation.

6.1 Inventory, RF Survey and Access Operations

LLRP is based upon an abstraction of RFID air protocols and their respective commands. There are two principal concepts to the LLRP abstraction of RF operations by a Reader:

1) Reader Operations, and 2) Access operations. The remainder of this section provides a detailed description of these LLRP concepts.

Reader Operations (RO) define the parameters for operations such as Antenna Inventory and RF Survey. Access Operations define the parameters for performing data access operations to and from a tag.

The timing control of an operation is specified using boundary specification, which specifies how the beginning (using start trigger) and the end (using stop trigger) of the operation is to be determined.

An *antenna inventory* (AI) is the smallest unit of interaction between a Reader and tags in the antenna's *field-of-view (FOV)*. An *InventoryParameterSpec* defines the parameters to be used during the inventory operation including protocol, protocol-specific parameters, and RF parameters. During an AI, the tags in the FOV of the antennas are singulated using air protocol commands based on the contents of the *InventoryParameterSpec*. An *AISpec* binds a stop trigger and a set of antennas to a set of *InventoryParameterSpecs*, and is identified by a one based index called the SpecIndex. The stop trigger defines the termination condition of the aggregate *AISpec* operation comprising of N * M antenna inventory operations, where N and M are the cardinality of the antenna set and *InventoryParameterSpecs* set respectively. For example, if there is a single antenna and a single *InventoryParameterSpec* defined in an *AISpec*, the AI operation specified by the <antenna, *InventoryParameterSpec*> tuple is bounded by the stop trigger specification.

It should be noted that the stop trigger specification of each individual AI is not specified, which means the Reader is not limited to execute the AIs in the order in which they appear in an AISpec. The timing control and the sequencing of the individual AIs within an AISpec will be determined by the Reader.

RF Survey is an operation during which the Reader performs a scan and measures the power levels across a set of frequencies at an antenna. The RF survey operational parameters are described in a RFSurveySpec and it defines the survey operation at a single antenna. It comprises an identifier for the spec, an antenna identifier, stop trigger and set of parameters for the survey operation.

A *Reader Operation (RO)* describes the operations to be executed at one or more antennas of the Reader. A *RO* comprises at least one Spec, where a Spec is either an *AISpec* or a *RFSurveySpec*. If a *RO* comprises multiple Specs, each Spec is an *AISpec* or a *RFSurveySpec*. Each RO's operational parameters are described in a *ROSpec*. The *ROSpec* contains a spec identifier, the boundary specification for the entire RO operation, priority, a list of *AISpecs* and/or *RFSurveySpecs*, and optionally a reporting specification. The reporting specification defines the contents of RO Report and the trigger conditions when to send the inventory report and survey report. The order of *AISpec* and *RFSurveySpec* execution within a *ROSpec* is the order in which they appear in the *ROSpec*.

Figure 5 illustrates the statechart of a *ROSpec*. The *ROSpec* has three states: Disabled, Inactive and Active. The Client configures a new *ROSpec* using an ADD_ROSPEC message for the *ROSpec*. The *ROSpec* starts at the Disabled state waiting for the ENABLE_ROSPEC message for the *ROSpec* from the Client, upon which it transitions to the Inactive state. The ROSpec does not respond to start or stop triggers in the Disabled state. The Client disables a *ROSpec* using a DISABLE_ROSPEC message for the *ROSpec*.

The *ROSpec* transitions from the Inactive state to the Active state when ROSpecStartCondition occurs for the *ROSpec*. The *ROSpec* transitions back to the inactive state when ROSpecDoneCondition happens.

```
ROSpecStartCondition = ROSpecStartTrigger or START_ROSPEC

ROSpecDoneCondition = AllSpecsDone or ROSpecStopTrigger or preempted or
(STOP_ROSPEC message for the ROSpec from the Client)
```

The *ROSpec* when undefined is no longer considered for execution. The Client undefines the *ROSpec* using a DELETE_ROSPEC message for the *ROSpec*.

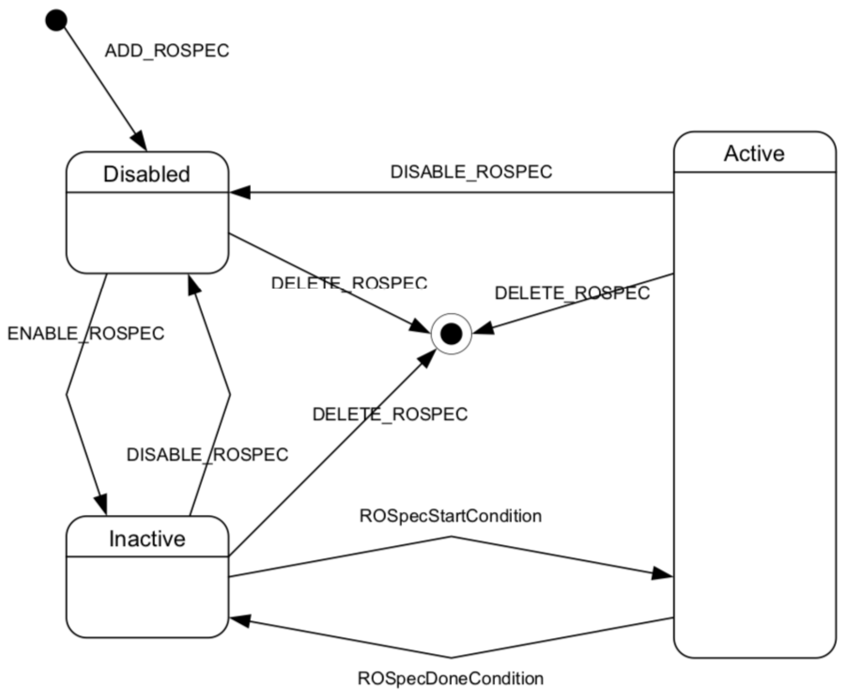

Figure 5: ROSpec Statechart

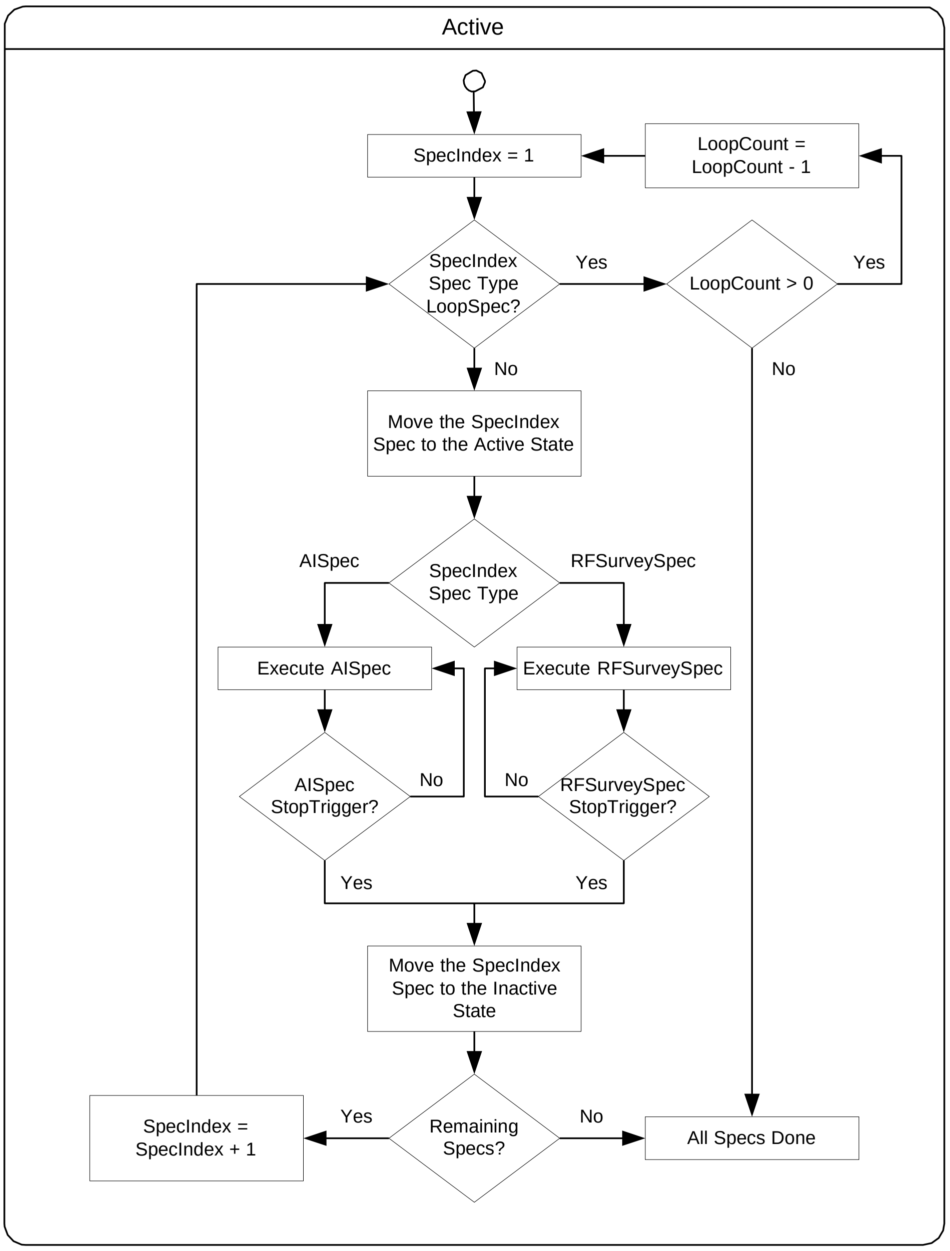
Active
SpecIndex = 1
LoopCount = LoopCount - 1
SpecIndex Spec Type LoopSpec?
Yes
LoopCount > 0
Yes
No
No
Move the SpecIndex Spec to the Active State
AISpec
SpecIndex Spec Type
RFSurveySpec
Execute AISpec
Execute RFSurveySpec
AISpec StopTrigger?
No
No
RFSurveySpec StopTrigger?
Yes
Yes
Move the SpecIndex Spec to the Inactive State
SpecIndex = SpecIndex + 1
Yes
Remaining Specs?
No
All Specs Done

Figure 6: ROSpec Active State

LLRP supports configuring multiple *ROSpecs*. Each *ROSpec* has a priority field. The default is for all the *ROSpecs* to have the same priority. Since the start trigger for the *ROSpec* can be an asynchronous event, there may be situations where a *ROSpec's* start trigger event occurs when the Reader is busy executing another *ROSpec*. The Client, when setting up a *ROSpec,* can set the appropriate priority so that a high priority *ROSpec* can preempt a currently active lower priority *ROSpec* and start execution as soon as the ROSpecStartCondition for the higher priority, inactive *ROSpec* occurs. The *ROSpec* that got preempted transitions to the Inactive state.

Figure 7 illustrates the AISpec statechart. When the parent ROSpec moves to the active state, each AISpec in the ROSpec starts at the inactive state. During an active ROSpec's execution, when an inactive AISpec is selected for execution, that AISpec moves to the active state. If there are multiple antennas and InventoryParameterSpecs in that AISpec, the Reader picks the next <antenna, InventoryParameterSpec> to execute. In the figure, the ID of the selected antenna is A, and the protocol for the selected InventoryParameterSpec is P. The Reader starts tag singulation for air protocol P on antenna A using the operational parameters specified in the InventoryParameterSpec. This involves one or more air protocol commands from the Reader via the antenna to the tags in the antenna's FOV. The tags get singulated and each tag's EPC information is received by the antenna. If further tag memory operations are to be performed, such as writing or reading other memory regions, it will be performed at this point. As illustrated in Figure 8, these access operations are interleaved with the execution of an AISpec. Access operations are described using AccessSpecs. AccessSpecs describe the tags (TagSpec) on which some operations are to be performed, the operations to be performed (OpSpec), the boundary specification, and optionally a reporting specification for the Access operation. The AccessSpec may contain antenna information at which this access operation needs to be executed and contains the air protocol to be used to perform the access operations. In addition, to accommodate scenarios where an access operation needs to be performed only during a particular ROSpec execution, the AccessSpec optionally contains the ROSpec information. There can be one or more AccessSpecs set up at the Reader.

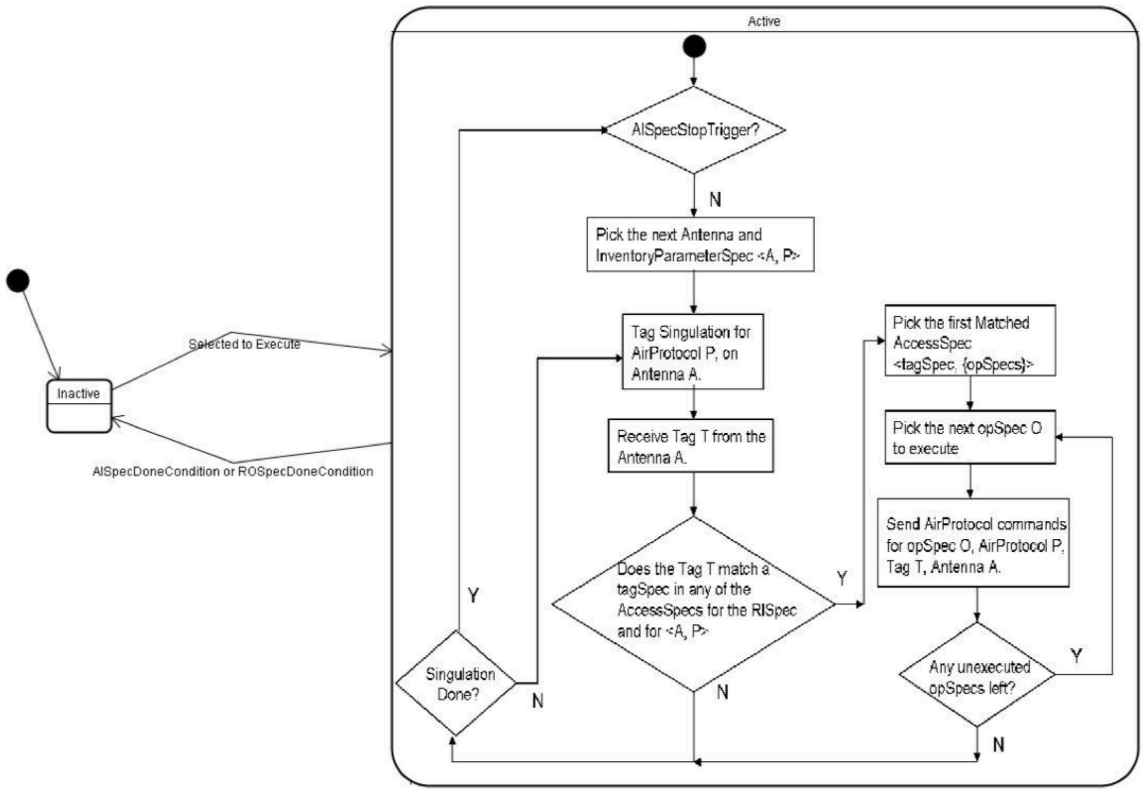
Active
AISpecStopTrigger?
N
Pick the next Antenna and InventoryParameterSpec <A, P>
Tag Singulation for AirProtocol P, on Antenna A.
Receive Tag T from the Antenna A.
Does the Tag T match a tagSpec in any of the AccessSpecs for the RISpec and for <A, P>
Y
N
Pick the first Matched AccessSpec <tagSpec, {opSpecs}>
Pick the next opSpec O to execute
Send AirProtocol commands for opSpec O, AirProtocol P, Tag T, Antenna A.
Any unexecuted opSpecs left?
Y
N
Singulation Done?
Y
N
Inactive
Selected to Execute
AISpecDoneCondition or ROSpecDoneCondition

Figure 7: Antenna Inventory Spec States

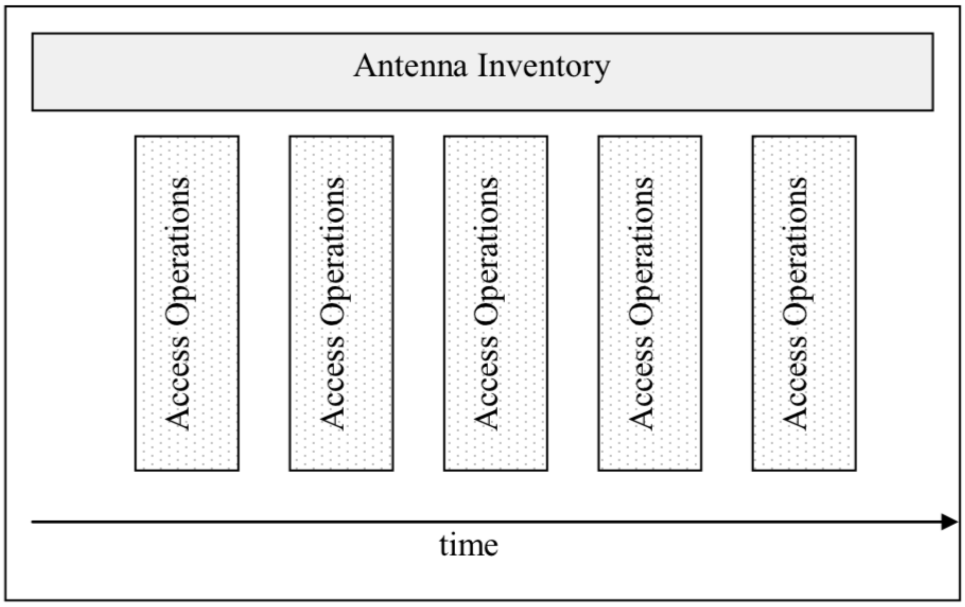

Figure 8: Access Operations Interleaved in an Antenna Inventory Operation

In Figure 7, when tags are received as a result of singulation, a check is performed to determine if the received tag matches the TagSpec defined in any of the Active (See statechart in Figure 10) AccessSpecs. In case there are multiple AccessSpecs that get matched during a tagSpec lookup, the Reader will execute the first AccessSpec that matches, where the ordering of the AccessSpecs is the order in which the AccessSpecs were created by the Client.

When an AccessSpec is executed, the set of operations as specified in OpSpecs of the AccessSpec are performed on the tag, which results in one or more air protocol commands and responses transacted between the Reader and the tag via antenna A over air protocol P. In order to support cases where the Reader needs to query the Client for further information to complete the operation on the tag, there is an OpSpec called the ClientRequestOpSpec.

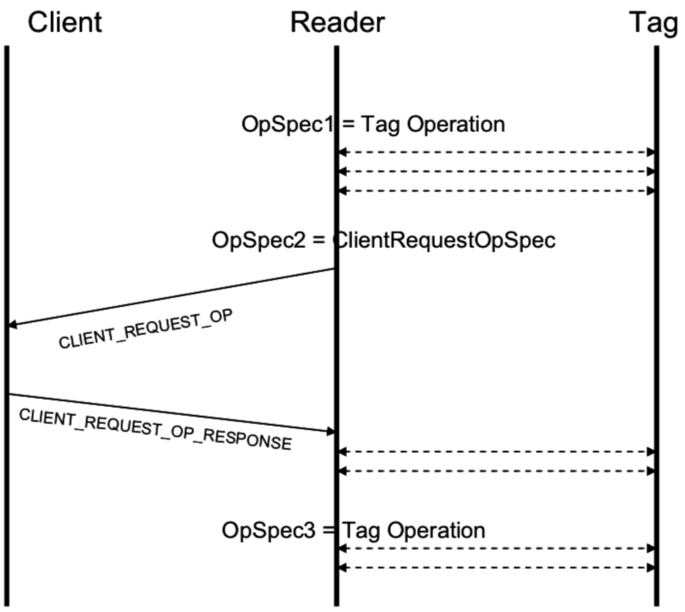

Figure 9: Client Request OpSpec

Figure 9 illustrates the message interaction between the Client, Reader and Tag for a ClientRequestOpSpec. For OpSpecs that are not ClientRequestOpSpec, the Reader performs the operations on the tag using the the air protocol commands. If an OpSpec is of the

ClientRequestOpSpec, the Reader sends the result of the ongoing AccessSpec till that point in a CLIENT_REQUEST_OP message, so that the Client has all the relevant information to send a response. The client response is carried in a CLIENT_REQUEST_OP_RESPONSE message. This message is the set of OpSpecs that the reader should execute. The reader continues to execute the OpSpecs within an AccessSpec until all opSpecs have been executed or until an error occurs. When execution completes, the reader resumes the inventory operation.

The AISpec transitions back to the inactive state when AISpecDoneCondition occurs or when the parent *ROSpec*'s ROSpecDoneCondition occurs.

AISpecDoneCondition = AISpecStopTrigger

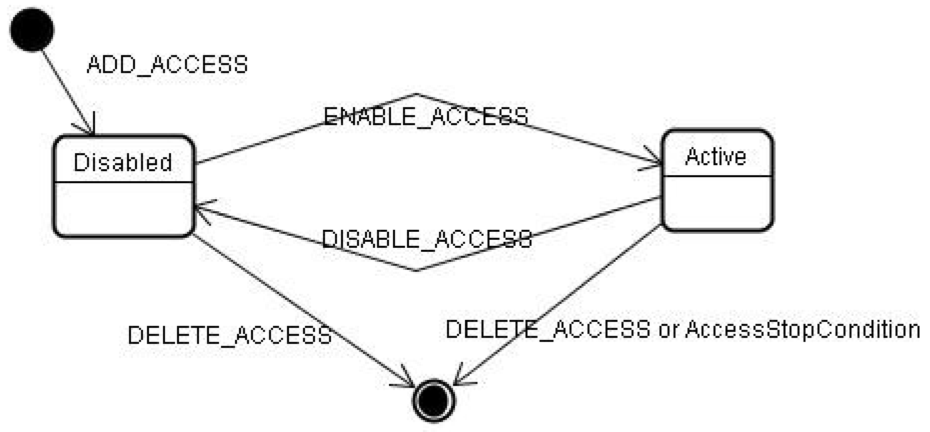

Figure 10: Access Spec States

Figure 10 illustrates the *AccessSpec's* states. The Client configures an *AccessSpec* using an ADD_ACCESS_SPEC message for the *AccessSpec*. The *AccessSpec* starts at the Disabled state, waiting for an ENABLE_ACCESS_SPEC message from the Client for that *AccessSpec*, upon which it enters the Active state. It is only in the Active state that the *AccessSpec* is considered for execution. The Client can disable an *AccessSpec* using a DISABLE_ACCESS_SPEC message for the *AccessSpec*. The *AccessSpec* when undefined is no longer considered for execution. The Client undefines the *AccessSpec* using a DELETE_ACCESS_SPEC message for the *AccessSpec*.

In order for the Reader to take a local action to limit the validity of an *AccessSpec*, the Client can configure a stop trigger for the *AccessSpec*. An example use case of the stop trigger is when an AccessSpec is defined on all the antennas, and the desired behavior is to operate on the tag only once, the first time it is seen at any antenna. When the AccessStopCondition occurs, the *AccessSpec* transitions to undefined and is no longer considered for execution.

AccessStopCondition = AccessSpecStopTrigger

Figure 11 illustrates the *RFSurveySpec* statechart. When the parent *ROSpec* moves to the active state, each *RFSurveySpec* in the *ROSpec* starts at the inactive state. During an active *ROSpec's* execution, when an inactive *RFSurveySpec* is selected for execution, that *RFSurveySpec* moves to the active state. In the active state, the Reader executes the survey operation as specified by the *RFSurveySpec*. The *RFSurveySpec* transitions back to the inactive state when the RFSurveySpecDoneCondition occurs or when the parent *ROSpec*'s ROSpecDoneCondition occurs

RFSurveyStopCondition = RFSurveySpecStopTrigger

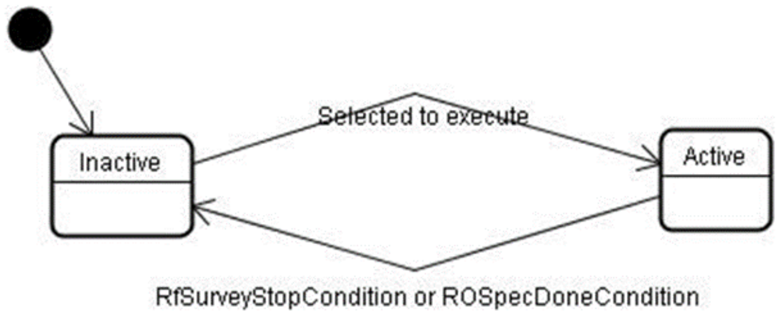

Figure 11: RFSurveySpec States

In summary, the Reader operation and Access operation specific data structures pass the following information between a Client and a Reader:

ROSpec: Details of a Reader operation

- ROSpecID: This identifier is generated by the Client. This identifier is used by the Client to perform operations on this ROSpec, like start, stop, enable, disable and delete. Reports that are generated as a result of the execution of this ROSpec also carry this identifier.
- ROBoundarySpec:
  - ROSpecStartTrigger, ROSpecStopTrigger: This is the start and stop trigger for this ROSpec. The triggers that are specifiable for a ROSpec are listed in Table 1.
- Priority: This is the priority of this ROSpec.
- CurrentState: This is the current state of the ROSpec – disabled, inactive, active. This field is kept up to date by the Reader based on the ROSpec's current state.
- Set of Specs: Each Spec is an *AISpec*, *RFSurveySpec*, *LoopSpec*, or *Custom*. The Specs are executed in the order in which it is defined in the ROSpec. The position of the Spec (AISpec, RFSurveySpec, or Custom) in the ROSpec is called the SpecIndex. The SpecIndex is used during reporting to identify the spec inside of ROSpec whose execution generated the data in the report. The numbering of SpecIndex is 1 based.
- ROReportingSpec: If specified, this defines when to send the results of this ROSpec, and also the contents and format of the report

AISpec: Details of one of more antenna inventory operations

- AISpecStopTrigger: This is the stop trigger for the AISpec. The triggers that are specifiable for an AISpec are listed in Table 1.
- Set of Antenna IDs: This is the set of antennas at which the inventory operations described in the InventoryParameterSpecs are executed. If there are N antennas and M InventoryParameterSpecs, the Reader will execute the M inventory operations at each of the specified antennas. Thus, in aggregate, the Reader will execute N * M *AI*s (Antenna inventory operations). The ordering of the *AI*s is determined by the Reader.
- Set of InventoryParameterSpecs: There can be one or more InventoryParameterSpecs specified as part of the AISpec. Collectively, they are bound by the AISpecStopTrigger. The order in which the antenna inventory operations described as <Antenna, InventoryParameterSpec> are executed is determined in a proprietary manner inside the Reader.

InventoryParameterSpec: Operational parameters for an inventory using a single air protocol.

- InventoryParameterSpecID: This identifier is generated by the Client. Reports that are generated as a result of the execution of this InventoryParameterSpec carry this identifier.
- Air Protocol: This is the air protocol that is used to inventory the tags in the field of view of the antenna.
- Set of Antenna Configuration Settings: Each Antenna Configuration setting comprises of

    - Antenna ID: The identifier of the antenna
    - RFTransmitterSettings: This describes the configuration of the transmitter during the inventory operation.
    - RFReceiverSettings: This describes the configuration of the receiver during the inventory operation.
    - AirProtocolInventoryCommandSettings parameters: This describes the configuration of the air protocol parameters for the inventory operation.

RFSurveySpec: Details of a RF Survey operation

- RFSurveySpecID: This identifier is generated by the Client. Reports that are generated as a result of the survey operation carry this identifier.
- RFSurveySpecStopTrigger: This is the stop trigger for the RFSurveySpec. The triggers that are specifiable for a RFSurveySpec are listed in Table 1.
- AntennaID: This is the antenna at which the survey operation is to be executed.
- StartFrequency: This is the starting channel for which power levels need to be measured during this RF survey operation.
- EndFrequency: This is the ending channel for which power levels need to be measured during this RF survey operation. The RF survey operation is performed on frequency channels between the specified Start Frequency and End frequency.

LoopSpec: Instructs the Reader to execute the first Spec in the Set of Specs.

- LoopCount: This value instructs the reader on the number of times to loop through the Set of Specs within the ROSpec.

AccessSpec: Details of an access operation.

- AccessSpecID: This identifier is generated by the Client upon creation of this AccessSpec. This identifier is used by the Client to perform operations on this AccessSpec, like start, stop and delete. Reports that are generated as a result of the execution of this AccessSpec also carry this identifier.
- AntennaID: This is the identifier of the antenna for whose tag observations this AccessSpec is executed.
- Air Protocol: This is the air protocol used to perform access operations on the tag.
- ROSpecID: This is the identifier of the ROSpec during whose tag observations this AccessSpec is executed.
- CurrentState: This is the current state of the AccessSpec – disabled, active. This field is kept up to date by the Reader based on the AccessSpec’s current state.
- AccessSpecStopTrigger: If specified, this is the trigger to undefine the AccessSpec upon the occurrence of the stop trigger.
- AccessCommand: This parameter is used to configure the air protocol parameters for the access operation. At a minimum, this specifies the tag filters for which the access operations are to be performed, and the list of operations to be performed on the tag.
    - TagSpec: This describes the tag filters and is specified in terms of the air protocol’s tag memory layout.
    - List of OpSpecs: This is specified in terms of the air protocol’s tag access operations. The order of execution is determined by the order in which it is configured in the AccessSpec.
- AccessReportSpec: If specified, this defines when to send the results of this AccessSpec, and also the contents and format of the report.

6.1.1 Operation Triggers

This section describes the triggers that can be configured using LLRP to control the various operations.

6.1.1.1 Summary

The specific triggers used to control the various operations are presented in a tabular fashion.

Table 1: Operation Triggers

Trigger Name	ROSpecStart	ROSpecStop	AISpecStop	AccessSpecStop	RFSurveySpec Stop
GPI Trigger	X	-	-	-	-
GPI Trigger with Timeout	-	X	X	-	-
N attempts	-	-	X	-	-
N tag observations	-	-	X	-	-
No tag observations for t ms	-	-	X	-	-
Immediate	X	-	-	-	-
Null	X	X	X	X	X
Time Based Periodic	X	-	-	-	-
Time Based Duration	-	X	X	-	X
Operation Count	-	-	-	X	X

6.1.1.2 Reader Operation Triggers

The triggers SHALL operate as follows:

- Null: When used as a start or a stop trigger, it implies no start or stop conditions have been specified, respectively.
- Immediate: This is used as a start trigger. Operations using this trigger will start immediately.
- Time-based: There are two different types of time-based triggers defined in LLRP periodic and duration.
  - Periodic: This is used as a start trigger. This is specified using UTC time [UTC], offset and period. For one-shot inventory, period is set to 0, and for periodic inventory operation, period > 0. If UTC time is not specified, the first start time is determined as (time of message receipt + offset), else, the first start time is determined as (UTC time + offset). Subsequent start times = first start time + k * period (where, k > 0).
  - Duration: This is used as a stop trigger.
- Tag observation based: There are three different types of tag-observation based triggers defined in LLRP. They are all used only as stop triggers. Each of these trigger types have a timeout value. So the trigger event happens when either the tag observation event happens or the timeout expires.
  - Upon seeing N tags, or timeout.
  - Upon seeing no more new tags for t milliseconds, or timeout
  - N attempts to see all the tags in the field of view, or timeout. Such an attempt to see the tags energised by a reader is defined as an inventory round (as defined by the EPC Radio-Frequency Identity Protocols Generation-2 UHF RFID Standard, Specification for RFID Air Interface Protocol for Communications at 860 MHz – 960 MHz, Release 2.1). An inventory round is initiated by a Query command and terminated by either a subsequent Query command (which also starts a new inventory round), a Select command, or a Challenge command.

- External events: These are due to events received at Reader interfaces like signal transition on a GPI port or a message on the network port.
  - GPI event at a GPI port, or a timeout
  - Client triggers: A Client can instruct the Reader to start/stop a particular operation using LLRP messages.
- Operation count: This is used as a stop trigger for RFSurvey. This trigger limits the number of times the Reader takes survey measurements across the specified frequency range.

AI and RFSurvey specs do not contain start triggers. The first spec (AISpec or RFSurveySpec) starts when the ROSpec enters the active state. The kth Spec in the ROSpec starts immediately after the completion of the k-1th Spec.

When Null is specified as a stop trigger for a Spec ((either AISpec or RFSurveySpec), the execution of the Spec is stopped only when the parent ROSpec's ROSpecDoneCondition occurs.

6.1.1.3 Access Operation Triggers

AccessSpecs do not contain start triggers. An AccessSpec when enabled using ENABLE_ACCESS_SPEC will transition to the active state. There is only one type of stop trigger for controlling the validity of an AccessSpec:

- Operation count: This is used as a stop trigger. This trigger is useful to limit the number of times the instance of the operation is executed during its lifetime.

6.2 Reporting, Event Notification and Keepalives

The results of the inventory, access and RF survey operations, will be sent by the Reader to the Client in the form of reports. Using LLRP, the Client is capable of setting up the triggers that determine when the report is to be sent by the Reader, and also the contents and format of the report. The report message is RO_ACCESS_REPORT. The triggers and report contents can be configured in one of the following ways:

- Differently for each ROSpec and AccessSpec when creating them using the ADD_ROSPEC and ADD_ACCESSSPEC messages, respectively.
- Global default using the SET_READER_CONFIG message.

Table 2 summarises the triggers available in LLRP to control when the RO report and the AccessReport is to be generated and sent by the Reader.

Table 2: Reporting Triggers

Trigger Name	ROReport	AccessReport
None	X	-
(Upon N tags or End of Spec), where Spec = AISpec or RFSurveySpec	X	-
Upon N tags or End of ROSpec	X	-
(Upon N seconds or milliseconds or End of Spec), where Spec = AISpec or RFSurveySpec	X	
Upon N seconds or milliseconds or End of ROSpec	X	
End of AccessSpec execution	-	X
Whenever ROReport is generated for the RO that triggered the execution of this AccessSpec	-	X

In addition to data reports, the Client can configure the Reader to enable or disable notification of events as and when it happens at the Reader. Some examples of events are frequency hop, buffer overflow, etc.

In order to monitor the LLRP-layer connectivity with the Reader, the Client can configure the Reader to send Keepalives periodically. The Keepalive message is acknowledged by the Client, using which, the Reader can also monitor the LLRP-layer connectivity with a Client. The Keepalives can be disabled. If enabled, the periodicity of the message is specified by the Client.

7 Messages, Parameters and Fields

LLRP is a message-oriented protocol made up of data elements called protocol data units. This section provides the details of each message type and parameter type, and expresses them in an abstract manner. The section starts with an overview of the message types and parameters, where the messages are grouped into separate functional groups.

7.1 Overview

LLRP provides an extensible mechanism to support existing and new air protocols. It is achieved by decoupling messages from parameters – using a common message structure across air protocols, and providing extensibility in the form of parameters.

7.1.1 Formatting Conventions and Data Types

LLRP messages and parameters are defined using the graphical notation below.

The contents of a LLRP message are listed within a box with a double line border such as this.

The contents of a LLRP message parameter are listed with a box with a single line border such as this.

Figure 12: Box Formats for Messages and Parameters

Contained within the box is an ordered list of sub-parameters and fields contained within the message or parameter. The field/parameter names are shown in **boldface**, followed by the data type and a brief description of the field/parameter when necessary. Fields with values that are restricted to a subset of the range of their data type have their possible and legal values shown in *italics* below the field name.

Fields are composed of one of the following basic data types:

Bit – An integer with only two possible values, 0 or 1

Bit Array – A sequence of bits.

Byte Array – A sequence of bytes.

Boolean – A field that can take the values TRUE or FALSE.

Integer – An integer can take any whole number. When this value is used in the abstract specification, the *Possible Values* element will specify the possible and legal value for a particular field.

Short Array – A sequence of unsigned short integers

Signed Integer – A signed integer can take any whole number value between -2^{31} through $2^{31}-1$ inclusive. Within the abstract specification, the *Possible Values* element will enumerate any restrictions beyond these limits for a particular field.

Signed Short Integer – A signed short integer can take any whole number value between -2^{15} through $2^{15}-1$ inclusive. Within the abstract specification, the Possible Values element will enumerate any restrictions beyond these limits for a particular field.

Unsigned Integer – An unsigned integer is a value that is between 0 through 2^{32}-1 inclusive. Within the abstract specification, the *Possible Values* element will enumerate any restrictions beyond these limits for a particular field.

Unsigned Long Integer – An unsigned long integer is a value that is between 0 through 2^{64}-1 inclusive. Within the abstract specification, the *Possible Values* element will enumerate any restrictions beyond these limits for a particular field.

Unsigned Short Integer – An unsigned short integer is a value that is between 0 through 2^{16}-1 inclusive. Within the abstract specification, the *Possible Values* element will enumerate any restrictions beyond these limits for a particular field.

UTF-8 String – A sequence of UTF-8 [UTF8] encoded characters.

In addition to the basic types, fields can be defined as 'lists' of a basic type. A list is an ordered set of a basic type. The order is preserved by all bindings.

7.1.2 Messages

Each Message contains:

- Version value that indicates the version of the protocol for this message. Table 3 shows how the value in the version field maps to the LLRP protocol version.

Table 3: Version Field to Protocol Version Mapping

Version Field	Protocol Spec Version
1	LLRP v1.0.1
2	LLRP v1.1
3	LLRP v2.0

- Message Type value that uniquely identifies it within a protocol message.
- Message ID: The Reader behavior SHALL be based upon starting the processing of messages in the order received over LLRP, however, the completion of execution of the message processing MAY not necessarily be in the same order inside the Reader. Hence, the Reader responses to the messages may be in a different order than the order of the Client messages. The Message ID is to facilitate multiple outstanding messages/requests from Client or Reader. The communications between the Client and the Reader is primarily of a request- response type - requests/commands from the Client to the Reader, and response from the Reader to the Client. The Message ID is used to associate a Reader response with the original Client message.
- In addition, it may contain mandatory or optional parameters.

7.1.3 Parameters

LLRP Parameters are used to communicate specific details of LLRP operation in LLRP Messages. Each Parameter contains:

- Parameter Type value that uniquely identifies it within a Message.
- In addition, it may contain individual fields or sub-parameters.

Compliance requirement: Compliant Readers and Clients SHALL use this enumeration.

AntennaID, ROSpecID, AccessSpecID, GPIPort, GPOPort: These fields are identifiers for LLRP-related objects within the Reader. For example, AntennaID is the identifier of the antenna; ROSpecID is the identifier of the ROSpec. The objects are indexed from 1. A value of non-zero for a field is a specific instance of the respective object. A value of zero means all instances of the respective object.

7.1.4 Functional Grouping

The LLRP messages are grouped into:

- **Protocol version management**: Messages that discover supported protocol versions, or set the protocol version for the current connection. They include:
  - GET_SUPPORTED_VERSION
  - GET_SUPPORTED_VERSION_RESPONSE
  - SET_PROTOCOL_VERSION
  - SET_PROTOCOL_VERSION_RESPONSE
- **Reader device capabilities**: Messages that query Reader capabilities. They include
  - GET_READER_CAPABILITIES
  - GET_READER_CAPABILITIES_RESPONSE
- **Reader operations control**: Messages that control the Reader's air protocol inventory and RF operations. They include
  - ADD_ROSPEC
  - ADD_ROSPEC_RESPONSE
  - DELETE_ROSPEC
  - DELETE_ROSPEC_RESPONSE
  - START_ROSPEC
  - START_ROSPEC_RESPONSE
  - STOP_ROSPEC
  - STOP_ROSPEC_RESPONSE
  - ENABLE_ROSPEC
  - ENABLE_ROSPEC_RESPONSE
  - DISABLE_ROSPEC
  - DISABLE_ROSPEC_RESPONSE
  - GET_ROSPECS
  - GET_ROSPECS_RESPONSE
- **Access control**: Messages that control the tag access operations performed by the Reader. They include
  - ADD_ACCESSSPEC
  - ADD_ACCESSSPEC_RESPONSE
  - DELETE_ACCESSSPEC
  - DELETE_ACCESSSPEC_RESPONSE
  - ENABLE_ACCESSSPEC
  - ENABLE_ACCESSSPEC_RESPONSE
  - DISABLE_ACCESSSPEC
  - DISABLE_ACCESSSPEC_RESPONSE
  - GET_ACCESSSPECS
  - GET_ACCESSSPECS_RESPONSE
  - CLIENT_REQUEST_OP
  - CLIENT_REQUEST_OP_RESPONSE

- **Reader device configuration**: Messages that query/set Reader configuration, and close LLRP connection. They include
  - GET_READER_CONFIG
  - GET_READER_CONFIG_RESPONSE
  - SET_READER_CONFIG
  - SET_READER_CONFIG_RESPONSE
  - CLOSE_CONNECTION
  - CLOSE_CONNECTION_RESPONSE
- **Reports**: These are messages that carry different reports from the Reader to the Client. Reports include Reader device status, tag data, RF analysis report. They include
  - GET_REPORT
  - RO_ACCESS_REPORT
  - READER_EVENT_NOTIFICATION
  - KEEPALIVE
  - KEEPALIVE_ACK
  - ENABLE_EVENTS_AND_REPORTS
- **Custom Extension**: This is a common mechanism for messages that contain vendor defined content.
  - CUSTOM_MESSAGE
- **Errors:** Typically the errors in the LLRP defined messages are conveyed inside of the responses from the Reader. However, in cases where the message received by the Reader contains an unsupported message type, or a CUSTOM_MESSAGE with unsupported parameters or fields, the Reader SHALL respond with this generic error message.
  - ERROR_MESSAGE

LLRP parameters are used to communicate specific settings of LLRP operation in the messages. A parameter contains one or more fields, and in some cases also may nest one or more other parameters.

Typically, each message type has its own set of parameters; however, there may be exceptions in some cases, where two different message types use the same parameter because they require the same setting exposed by the parameter.

7.1.5 LLRP Messages and Actions

This section describes the corresponding LLRP-related actions in the Reader upon receiving the various LLRP protocol messages. Figure 13 uses UML synchronous messaging notation. Messages are asynchronous.

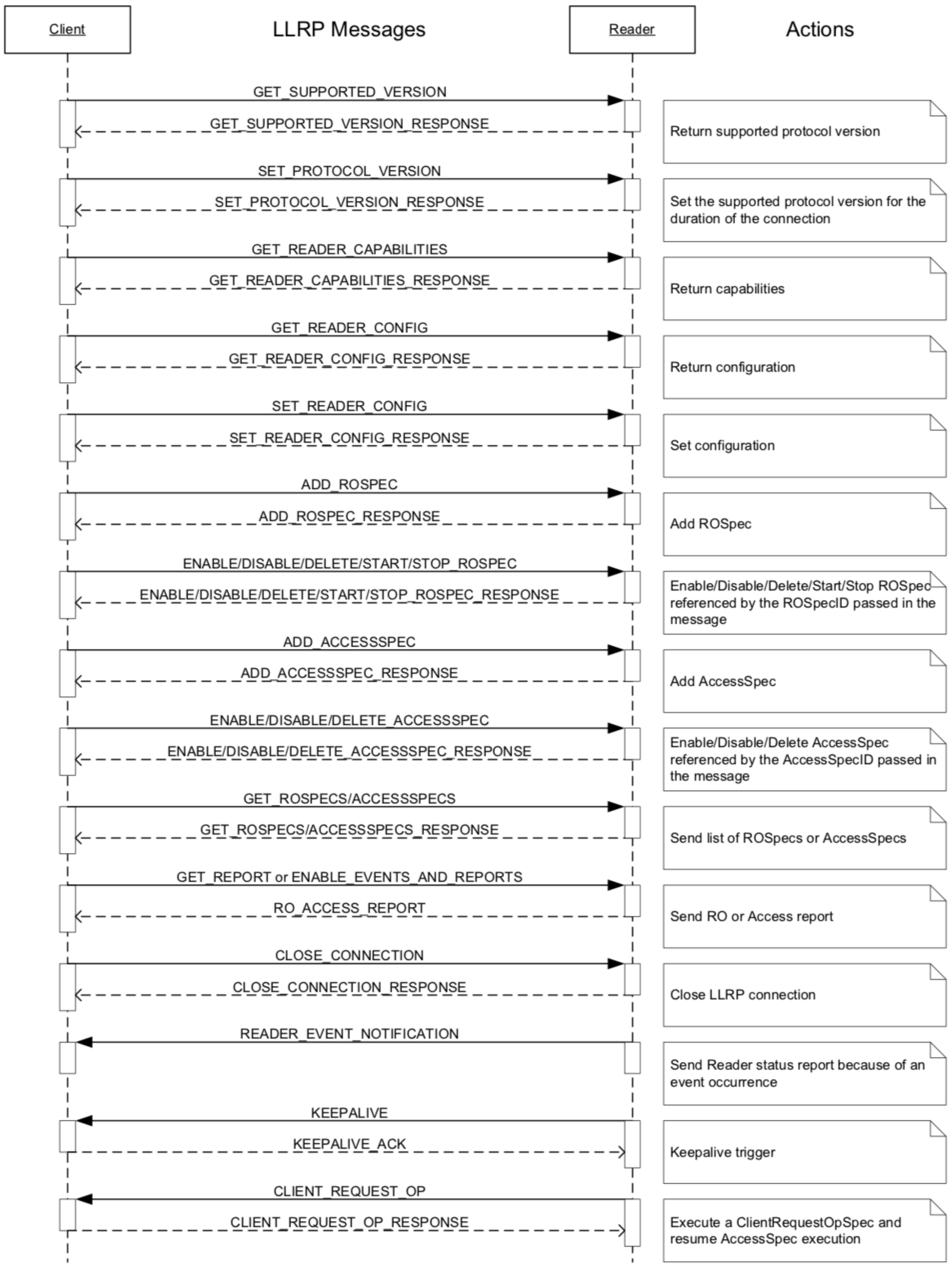
Client
LLRP Messages
Reader
Actions
GET_SUPPORTED_VERSION
GET_SUPPORTED_VERSION_RESPONSE
Return supported protocol version
SET_PROTOCOL_VERSION
SET_PROTOCOL_VERSION_RESPONSE
Set the supported protocol version for the duration of the connection
GET_READER_CAPABILITIES
GET_READER_CAPABILITIES_RESPONSE
Return capabilities
GET_READER_CONFIG
GET_READER_CONFIG_RESPONSE
Return configuration
SET_READER_CONFIG
SET_READER_CONFIG_RESPONSE
Set configuration
ADD_ROSPEC
ADD_ROSPEC_RESPONSE
Add ROSpec
ENABLE/DISABLE/DELETE/START/STOP_ROSPEC
ENABLE/DISABLE/DELETE/START/STOP_ROSPEC_RESPONSE
Enable/Disable/Delete/Start/Stop ROSpec referenced by the ROSpecID passed in the message
ADD_ACCESSSPEC
ADD_ACCESSSPEC_RESPONSE
Add AccessSpec
ENABLE/DISABLE/DELETE_ACCESSSPEC
ENABLE/DISABLE/DELETE_ACCESSSPEC_RESPONSE
Enable/Disable/Delete AccessSpec referenced by the AccessSpecID passed in the message
GET_ROSPECS/ACCESSSPECS
GET_ROSPECS/ACCESSSPECS_RESPONSE
Send list of ROSpecs or AccessSpecs
GET_REPORT or ENABLE_EVENTS_AND_REPORTS
RO_ACCESS_REPORT
Send RO or Access report
CLOSE_CONNECTION
CLOSE_CONNECTION_RESPONSE
Close LLRP connection
READER_EVENT_NOTIFICATION
Send Reader status report because of an event occurrence
KEEPALIVE
KEEPALIVE_ACK
Keepalive trigger
CLIENT_REQUEST_OP
CLIENT_REQUEST_OP_RESPONSE
Execute a ClientRequestOpSpec and resume AccessSpec execution

Figure 13: LLRP Messages and Reader Actions

8 Custom Extension

LLRP supports vendor extensions for defining commands and parameters within certain commands. All LLRP bindings support these extension mechanisms.

8.1 CUSTOM_MESSAGE

This message carries a vendor defined format from Reader to Client or Client to Reader. In addition to the version and messageID, the custom message also carries the information below.

CUSTOM_MESSAGE

Vendor Identifier: Unsigned Integer. IANA Private Enterprise Number

Message Subtype: Integer

Possible Values: 0-255.

Data: vendor specific format

No requirements are made as to the content or parameters contained within the Data portion of these messages. Clients MAY ignore CUSTOM_MESSAGEs. Readers SHALL accept CUSTOM_MESSAGE and return an ERROR_MESSAGE if CUSTOM_MESSAGE is unsupported by the Reader or the CUSTOM_MESSAGE contains fields and/or parameters that are unsupported by the Reader.

8.2 Custom Parameter

Certain Messages and Parameter Sets within LLRP allow for the insertion of vendor defined parameters. These custom parameters have the following format.

Custom Parameter

Vendor Identifier: Unsigned Integer**.** IANA Private Enterprise Number

Parameter Subtype: Unsigned Integer

Data: vendor specific format

Clients SHALL accept messages (except for CUSTOM_MESSAGE) that contain custom parameters but MAY ignore all custom parameters within these messages. Readers SHALL accept messages (except for CUSTOM_MESSAGE) that contain custom parameters and SHALL return an error when such parameters are unsupported.

8.3 Custom Extension in Commands

The following commands allow one or more custom Parameters in their message structure:

GET_READER_CAPABILITIES GET_READER_CONFIG GET_READER_CAPABILITIES_RESPONSE GET_READER_CONFIG_RESPONSE SET_READER_CONFIG

8.4 Custom Extension in Individual LLRP Parameters

LLRP only allows extension to parameters where the parameter set is defined with a custom Parameter type in the abstract model. All custom extension points will be marked in the abstract standard using the notation

Custom Extension Point List: List of <custom Parameter> [optional]

The following example illustrates a fictitious parameter that allows the embedding of custom extension parameters.

Example Parameter

Field1: Unsigned Integer

relatedData: Example Sub Parameter

Custom Extension Point List: List of <custom Parameter> [optional]

This example shows that the Example Parameter could contain an optional custom parameter that must adhere to the custom Parameter format.

8.5 Allowable Parameter Extension

All parameter values are specified within the abstract binding. A Reader or Client SHALL NOT extend the range of fields defined within the abstract specification unless the possible values indicate ranges for user defined options.

For example, the Indentification Parameter defines a field to carry the ID type.

IDType: Integer

```
Possible Values:

        IDType   ID
        ------   --
          0    MAC address
          1    EPC
```

A Client or Reader adhering to the standard SHALL generate an **IDType** field with only those values shown (0-1). A Reader or Client implementation SHALL generate an error upon receiving a value outside this range.

9 Protocol Version Management

The messages within this category deal with the discovery of supported protocol versions, and the selection of a protocol version for the current LLRP connection. This sequence is referred to as *version negotiation*. For the purposes of version negotiation, the definition of a LLRP connection is left to the underlying transport layer (see section *19.1* more information).

9.1 Messages

9.1.1 GET_SUPPORTED_VERSION

This message is sent from the Client to the Reader. Clients MAY send this message at any time during a LLRP connection.

Because this message is used for version negotiation and this procedure was introduced in LLRP 1.1 (version 2), this message SHALL be sent with its Ver field set to 2 prior to a successfully negotiated protocol version. Once a protocol version has been established, this message SHALL be sent with the negotiated version. Clients that receive an ERROR_MESSAGE in response to this message with a StatusCode of M_UnsupportedVersion SHALL assume the Reader only supports LLRP 1.0.1 (version 1).

Compliance requirement: Compliant Readers and Clients SHALL implement this message

GET_SUPPORTED_VERSION

9.1.2 GET_SUPPORTED_VERSION_RESPONSE

This is the response message from the Reader to the GET_SUPPORTED_VERSION message. The response contains the LLRPStatus Parameter and the version information supported by the Reader. The SupportedVersion field is inclusive starting with LLRP version 1.0.1 (version 1). For example, as advertised SupportedVersion value of 5 by the Reader SHALL indicate that the Reader supports versions [1-5] inclusive.

Because this message is used for version negotiation and this procedure was introduced in LLRP 1.1 (version 2), this message SHALL be sent with its Ver field set to 2 prior to a successfully negotiated protocol version. Once a protocol version has been established, this message SHALL be sent with the negotiated version.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_SUPPORTED_VERSION_RESPONSE

Response: LLRPStatus Parameter

CurrentVersion: Unsigned Byte. The currently negotiated protocol version.

SupportedVersion: Unsigned Byte. The maximum supported protocol version.

9.1.3 SET_PROTOCOL_VERSION

This message is sent from the Client to the Reader to set the protocol version for the current connection. Clients SHALL send this message either zero or one times during a LLRP connection, assuming the SET_PROTOCOL_VERSION_RESPONSE from the Reader indicates that the protocol selection was successful. Once a successful protocol version has been established this message SHALL not be sent again.

Because this message is used for version negotiation and this procedure was introduced in LLRP 1.1 (version 2), this message SHALL be sent with its Ver field set to 2. Clients that receive an ERROR_MESSAGE in response to this message with a StatusCode of M_UnsupportedVersion SHALL assume the Reader only supports LLRP 1.0.1 (version 1).

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

SET_PROTOCOL_VERSION

ProtocolVersion: Unsigned Byte. The desired protocol version.

9.1.4 SET_PROTOCOL_VERSION_RESPONSE

This is the response message from the Reader to the SET_PROTOCOL_VERSION message. The response contains only the LLRPStatus Parameter. If the version requested by the Client is not supported by the Reader, the Reader SHALL respond with an LLRPStatus StatusCode of M_UnsupportedVersion, and no sub-parameters (such as FieldError or ParameterError). If a negotiated version has already been established by a successful SET_PROTOCOL_VERSION message, the Reader SHALL respond with a StatusCode of M_UnexpectedMessage. If the Client requests protocol version 1, the Reader SHALL respond with a StatusCode of M_Success.

Because this message is used for version negotiation and this procedure was introduced in LLRP 1.1 (version 2), this message SHALL be sent with its Ver field set to 2.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

SET_PROTOCOL_VERSION_RESPONSE

Response: LLRPStatus Parameter.

10 Reader Device Capabilities

There are four broad categories of capabilities that are advertised by the Reader: general device, LLRP, regulatory, and air protocol capabilities.

10.1 Messages

10.1.1 GET_READER_CAPABILITIES

This message is sent from the Client to the Reader. The Client is able to request only a subset or all the capabilities from the Reader.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_READER_CAPABILITIES

RequestedData: Integer

Possible Values:

```
    Value        Requested Data
    ------      ----------------
      0          All
      1          General Device Capabilities
      2          LLRP Capabilities
      3          Regulatory Capabilities
      4          Air Protocol LLRP Capabilities
```

Custom Extension Point List: List of <custom Parameter> [optional]

10.1.2 GET_READER_CAPABILITIES_RESPONSE

This is the response from the Reader to the GET_READER_CAPABILITIES message. The response contains the LLRPStatus Parameter and the list of parameters for the requested capabilities conveyed via RequestedData in the GET_READER_CAPABILITIES message.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_READER_CAPABILITIES_RESPONSE

Response Data: Set of LLRP Parameters.

Possible Values: The possible members are

<LLRPStatus Parameter>, and,

one or more from the set

< GeneralDeviceCapabilities Parameter,

LLRPCapabilities Parameter,

RegulatoryCapabilities Parameter,

AirProtocolLLRPCapabilities Parameter >.

Custom Extension Point List: List of <custom Parameter> [optional]

10.2 Parameters

10.2.1 GeneralDeviceCapabilitiesParameter

This parameter carries the general capabilities of the device like supported air protocols, version of the Reader firmware, device hardware and software information, and receive sensitivity table.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

GeneralDeviceCapabilities Parameter

Device manufacturer name: Unsigned Integer. The IANA Private Enterprise Number (PEN).

Model name: Unsigned Integer

Firmware version: UTF-8 String

Maximum number of antennas supported: Unsigned Short Integer

CanSetAntennaProperties: Boolean. If set to true, the Client can set antenna properties (section *13.2.5*), else, the Client can not set it, but only query it using GET_READER_ CONFIG.

Maximum Receive Sensitivity: <MaximumReceiveSensitivity Parameter> [optional]

Receive Sensitivity Table: List of <ReceiveSensitivityTableEntry Parameter>

Per Antenna Receive Sensitivity Range: List of <PerAntennaReceiveSensivityRange Parameter>

Air protocol supported per antenna: N instances of<PerAntennaAirProtocol Parameter>, where N = Maximum number of antennae supported.

GPIO Support: <GPIO Capabilities Parameter>

HasUTCClockCapability: Boolean. If set to true, the Reader reports time based on UTC timestamps (section *7.1.3.1.1.1*) in its reports, else, the Reader reports time based on Uptime (section *7.1.3.1.1.2*) in its reports

10.2.1.1 MaximumReceiveSensitivity Parameter

This parameter specifies the maximum receive sensitivity supported by the Reader. Readers that allow control of receive sensitivity advertise values relative to this maximum sensitivity (see section *10.2.1.2*) and SHALL implement this parameter. If the Reader does not allow control of receive sensitivity, this parameter MAY be omitted.

MaximumReceiveSensitivity Parameter

Maximum sensitivity value: Signed Short Integer. The value is in absolute dBm.

10.2.1.2 ReceiveSensitivityTableEntry Parameter

This parameter specifies the index into the Receive Sensitivity Table for a receive sensitivity value. The receive sensitivity is expressed in dB and the value is relative to the maximum sensitivity (see section *10.2.1.1*). If the Reader does not allow control of receive sensitivity, a table of one entry is returned, the entry having the value of zero.

If the Reader allows control of receive sensitivity and the Reader also supports multiple antennas where the antennas can have different receive sensitivity values, then the Receive Sensitivity Table should be a set of values representing the union of sensitivity values for all antennas.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ReceiveSensitivityTableEntry Parameter

Index: Unsigned Short Integer

Receive sensitivity value: Integer. The value is in dB relative to the maximum sensitivity.

Possible Values: 0 to 128.

10.2.1.3 PerAntennaReceiveSensitivityRange Parameter

For a particular antenna, this parameter specifies the Reader's valid index range in the Receive Sensitivity Table. A Reader should report this parameter if the Reader allows control of receive sensitivity (i.e., the Reader reports a Receive Sensitivity Table with more than one entry) and the Reader supports multiple antennas where the antennas can have different receive sensitivity values.

If this parameter is omitted, then the Client SHALL assume that for all of the Reader's antennas the index range is the same as in the Receive Sensitivity Table.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

PerAntennaReceiveSensitivityRange Parameter

Antenna ID: Unsigned Short Integer

Possible Values:

1 to N, where N is the maximum number of antennas supported by the device.

ReceiveSensitivityIndexMin: Unsigned Short Integer

Possible Values:

0 to S, where S is the number of Receive Sensitivity Table entries reported by the Reader.

ReceiveSensitivityIndexMax: Unsigned Short Integer

Possible Values:

Mn to S, where Mn is the ReceiveSensitivityIndexMin and S is the number of Receive Sensitivity Table entries reported by the Reader.

10.2.1.4 PerAntennaAirProtocol Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

PerAntennaAirProtocol Parameter

Antenna ID: Unsigned Short Integer

Possible Values:

1 to N, where N is the maximum number of antennas supported by the device.

Air protocols supported: List of Protocol Ids enumerated based on Table 4.

10.2.1.5 GPIOCapabilities Parameter

This parameter describes the GPIO capabilities of the Reader. A value of zero for NumGPIs indicates that the Reader does not have general purpose inputs. A value of zero for NumGPOs indicates that the Reader does not have general purpose outputs.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

GPIOCapabilities Parameter

NumGPIs: Unsigned Short Integer. Number of general purpose inputs supported by the device.

NumGPOs: Unsigned Short Integer. Number of general purpose outputs supported by the device.

10.2.2 LLRPCapabilities Parameter

This parameter describes the LLRP protocol capabilities of the Reader. These include optional LLRP commands and parameters, capacities of data structures used in LLRP operations, and air protocol specific capabilities used by LLRP.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter. Readers MAY support RFSurvey, MAY support tag inventory state aware singulation, MAY support UTC clocks, MAY support buffer fill warning reports, MAY support EventAndReportHolding upon reconnect, and MAY support ClientRequestOpspec. Readers SHALL support at least one ROSpec, one AISpec per ROSpec, one InventoryParameterSpec per AISpec, one AccessSpec, and one OpSpec per AccessSpec.

LLRPCapabilities Parameter

CanDoRFSurvey: Boolean. If set to true, the Reader can perform RFSurvey operations (section *11.2.3*).

CanDoTagInventoryStateAwareSingulation: Boolean. If set to true, the Reader can support tag inventory state aware singulation.

CanReportBufferFillWarning: Boolean. If set to true, the Reader can report buffer fill warning in the reader event notification (section *14.2.6.5*).

MaxNumROSpecs: Integer. If zero, there is no limit. This is the maximum number of ROSpecs that can be configured at the Reader.

MaxNumSpecsPerROSpec: Integer. If zero, there is no limit. This is the maximum number of Specs (either AISpec or RFSurveySpec) that can be configured as part of a ROSpec at the Reader.

MaxNumInventoryParameterSpecsPerAISpec: Integer. If zero, there is no limit. This is the maximum number of InventoryParameterSpecs that can be configured per AISpec.

MaxPriorityLevelSupported: Integer. This is the maximum priority level supported in the reader. If set to less than or equal to 1, the Reader has no preemption support.

Possible Values: 0-7.

MaxNumAccessSpecs: Integer. If zero, there is no limit. This is the maximum number of AccessSpecs that can be configured at the Reader.

MaxNumOpSpecsPerAccessSpec: Integer. If zero, there is no limit. This is the maximum number of OpSpecs that can be configured per AccessSpec at the Reader.

SupportsClientRequestOpSpec: Boolean. If set to true, the Reader supports client request OpSpecs (section *12.2.1.2.1*).

ClientRequestOpSpecTimeout: Unsigned Short Integer (in milliseconds). The time the Reader will wait for the CLIENT_REQUEST_OP_RESPONSE from the Client after sending a RO_ACCESS_REPORT message upon executing the ClientRequestOpSpec OpSpec. This field is valid only if the Reader supports ClientRequestOpSpec (section *12.2.1.2.1*). If this field is 0, there is no limit

SupportsEventAndReportHolding: Boolean. If set to True, the Reader supports the EventsAndReports Parameter and the ENABLE_EVENTS_AND_REPORTS message. If set to false, the Reader does not support the ENABLE_EVENTS_AND_REPORTS message or the EventsAndReports Parameter.

10.2.3 C1G2LLRPCapabilities Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter. Readers SHALL support at least one select filter per query.

C1G2LLRPCapabilities Parameter

CanSupportBlockErase: Boolean

CanSupportBlockWrite: Boolean

CanSupportBlockPermalock: Boolean

CanSupportTagRecommissioning: Boolean DEPRECATED/(RFU)

CanSupportUMIMethod2: Boolean
CanSupportXPC: Boolean
MaxNumSelectFiltersPerQuery: Unsigned Short Integer. If set to zero, it indicates there is no maximum limit.
CanSupportUntraceable: Boolean. Indicates support for Untraceable command
CanSupportChallenge: Boolean. Indicates support for Challenge command
CanSupportAuthenticate: Boolean. Indicates support for Authenticate command

CanSupportAuthComm: Boolean. Indicates support for AuthComm

This parameter carries the RF regulation specific attributes. They include regulatory standard, frequency band information, power levels supported, frequencies supported, and any air protocol specific values that are determined based on regulatory restriction.

The regulatory standard is encoded using two Integer fields, <Country Code, Communications standard> and it specifies the current operational regulatory mode of the device. This should not be used to reflect the ability to operate in regulatory environments which require configuration different from the current. This version of the LLRP protocol will have support for only the UHF band.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

10.2.4 RegulatoryCapabilitiesParameter

RegulatoryCapabilities Parameter

Country Code: Unsigned Short Integer. This field carries the numeric code of the country as specified in ISO 3166 [ISO3166]. 0 means unspecified.

Communications Standard: Unsigned Short Integer. This field carries the enumerations of the communications standard as specified below.

Value	"DEPRECATED" if no longer applicable	Country/Region or Regulation	Frequency in MHz boundaries		Power in W	ERP or EIRP	Technique
			lower	upper			
0		unspecified					
1		United States FCC §15.247	902	928	4	EIRP	FHSS
2		ETSI 302-208					
3	DEPRECATED	ETSI 300-220			0.5		
4		Australia	918	926	1	EIRP	
5		Australia	920	926	4	EIRP	
6	DEPRECATED	Japan ARIB STD T89					
7		Hong Kong	865	868	2	ERP	
8	DEPRECATED	Taiwan: DGT LP0002					
9	DEPRECATED	Korea: MIC Article 5-2					
10		902-928 MHz 4 W EIRP FHSS	902	928		EIRP	FHSS
11		ETSI 302-208 "Lower Band"	865	868	2	ERP	
12		Brazil	902	907.5	4	EIRP	FHSS
13		China	840.5	844.5	2	EIRP	FHSS
14		China	920.5	924.5	2	EIRP	FHSS
15		Hong Kong, China	920	925	4	EIRP	FHSS
16		Israel	915	917	2	EIRP	
17	DEPRECATED	Japan	952	954	4	EIRP	LBT
18	DEPRECATED	Japan	952	955	0.02	EIRP	LBT
19		865-868 MHz 0.5W ERP	865	868	0.5	ERP	
20		Korea, Rep.	917	920.8	4	EIRP	FHSS or LBT
21		Korea, Rep.	917	923.5	0.2	EIRP	FHSS or LBT
22	DEPRECATED	Malaysia	866	869	2	ERP	
23		Malaysia	919	923	2	ERP	
24		New Zealand	864	868	6	EIRP	FHSS
25		Singapore	866	869	0.5	ERP	
26		Singapore	920	925	2	ERP	
27		South Africa	915.4	919	4	EIRP	FHSS
28		South Africa	919.2	921	4	EIRP	non-modulated
29		Taiwan	922	928	1	ERP	FHSS
30		Taiwan	922	928	0.5	ERP	FHSS
31		Thailand	920	925	4	EIRP	FHSS
32		Venezuela	922	928	4	EIRP	FHSS
33		Vietnam	866	869	0.5	ERP	
34		Vietnam	920	925	2	ERP	
35		Japan	916.7	920.9	4	EIRP	
36		Japan	916.7	923.5	0.5	EIRP	LBT
37		Brazil	915	928	4	EIRP	FHSS
38		New Zealand	920	928	6	EIRP	FHSS
39-65535		Reserved for Future Use					

UHFBandCapabilities: <UHFBandCapabilities Parameter> [optional]
Custom Extension Point List: List of <custom Parameter> [optional]

Note: The list of regulations can be found at *https://www.gs1.org/sites/default/files/docs/epc/uhf_regulations.pdf*

10.2.4.1 UHFBandCapabilities Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

UHFBandCapabilities Parameter

TransmitPowerTable: List of <TransmitPowerLevelTableEntry Parameter>

Frequency Information: <FrequencyInformation Parameter>

RFSurveyFrequencyCapabilities: <RFSurveyFrequencyCapabilities Parameter> [optional]

UHF_RFModeTable: List of LLRP Parameter.

Possible Values:

Each air protocol's UHF RF mode table is expressed as a different LLRP parameter. Each protocol SHALL be referenced not more than once. The air protocol's UHF RF mode table capabilities LLRP Parameters are defined in section *16.1*

10.2.4.2 TransmitPowerLevelTableEntry Parameter

This parameter specifies the index into the TransmitPowerLevelTable for a transmit power value. The transmit power is expressed in dBm*100 to allow fractional dBm representation and is the conducted power at the connector of the Reader.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

TransmitPowerLevelTableEntry Parameter

Index: Integer

Possible Values: 0-255

Transmit power value: Signed short integer. Transmit power expressed in dBm*100 to allow fractional dBm representation.

10.2.4.2.1 FrequencyInformation Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

Frequency Information Parameter

Hopping: Boolean

Freq Hop Info: Zero or more instances of <FrequencyHopTable Parameter>. This is transmitted only when Hopping = true.

Fixed Freq Info: At most one instance of <FixedFrequencyTable>. This is transmitted only when Hopping = false.

10.2.4.2.2 FrequencyHopTable Parameter

This parameter carries the frequency hop table parameters. This is used for Readers operating in regions with frequency hopping regulatory requirements. If the Reader is capable of storing multiple hop tables, the Reader may send all of them to the Client. Each hop table contains:

- HopTableID which is the index of the frequency hop table returned by the Reader.
- This is followed by a list of the frequencies (in kHz) in hop table order. The one- based position of a frequency in the list is defined as its ChannelIndex (i.e. the first frequency is referred to as ChannelIndex one).

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter when operating in frequency hopping regulatory regions.

FrequencyHopTable Parameter

HopTableID : Integer

Possible Values: 0 - 255

Frequency Hop List: List of unsigned integers. Frequency in kHz.

If multiple frequency hop tables are supported by the Reader, each table can be sent using a separate Frequency Hop Table Parameter.

10.2.4.2.3 FixedFrequencyTable Parameter

This parameter carries the fixed frequency list that can be used by the Reader. The one- based position of a frequency in the list is defined as its ChannelIndex (i.e. the first frequency is referred to as ChannelIndex one).

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter when operating in fixed frequency regulatory regions.

Fixed Frequency Parameter

Frequency List: List of unsigned integers. Frequency in kHz.

10.2.4.3 RFSurveyFrequencyCapabilities Parameter

This parameter describes the Reader's range of supported receive frequencies. This specifies the lower and upper limit of frequencies allowed in an RFSurveySpec.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter if CanDoRFSurvey is reported as "true" in the LLRPCapabilties parameter.

RFSurveyFrequencyCapabilities Parameter

MinimumFrequency: Unsigned Integer. Minimum receive frequency (in kHz) supported by the Reader.

MaximumFrequency: Unsigned Integer. Maximum receive frequency (in kHz) supported by the Reader.

10.2.4.4 UHFC1G2RFModeTable Parameter

This parameter carries the set of C1G2 RF modes that the Reader is capable of operating.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

UHFC1G2RFModeTable Parameter

UHFC1G2RFModeSet: List of <UHFC1G2RFModeTableEntry Parameter>

10.2.4.4.1 UHFC1G2RFModeTableEntry Parameter

This parameter carries the information for each UHFC1G2 RF mode. A mode that has been tested for conformance by the EPCglobal Hardware Action Group's Testing and Conformance (HAG T&C) group, is indicated using a conformance flag.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

UHFC1G2RFModeTableEntry Parameter

Mode identifier: Unsigned Integer. This is a Reader defined identifier that the client may use to set the Gen2 operating parameters.

DR Value: Integer. Divide ratio.

Possible Values:

```
        Value         DR
        -----         --
         0           8
         1           64/3
```

BDR Value: Integer. Backscatter data rate in bps.

Possible Values:

40000 – 640000 bps

M value: Integer. Modulation.

Possible Values:

```
       Value         M
        ------       ---
         0           FM0
         1           2
         2           4
         3           8
```

Forward link modulation: Integer

Possible Values:

```
       Value         Modulation
       ------       ------------
         0           PR-ASK
         1           SSB-ASK
         2           DSB-ASK
```

PIE Value: Integer. One thousand times the ratio of data-0 symbol length and data-1 symbol length in pulse-interval encoding. The C1G2 spec specifies a ratio range of 1.5 – 2.0. (see section 6.3.1.2.4 in [C1G2]).

Possible Values:

1500-2000

MinTariValue: Integer. Minimum Tari time in nanoseconds (see section 6.3.1.2.4 in [C1G2])

Possible Values:

6250-25000

MaxTariValue: Integer. Maximum Tari time in nanoseconds. (see section 6.3.1.2.4 in [C1G2]).

Possible Values:

6250-25000

StepTariValue: Integer. Tari Step size in nanoseconds.(see section 6.3.1.2.4 in [C1G2])

Possible Values:

0 – 18750 nsec

Spectral Mask Indicator: Integer. Spectral mask characteristics of the mode. The Reader SHALL advertise this value if and only if the spectral mask value is valid for all the Tari steps in the range.

Possible Values:

```
Value        Modulation
------      ------------
  0          Unknown
  1          SI – Meets [C1G2] Single-Interrogator Mode
             Mask
  2          MI – Meets [C1G2] Multi-Interrogator Mode Mask
  3          DI - Meets [C1G2] Dense-Interrogator Mode Mask
```

EPC HAG T&C Conformance: Boolean. This flag indicates if the Reader vendor has received the certification for the parameter sets specified in this mode. The Reader SHALL set this flag to true only if the Reader vendor has received EPCglobal conformance for this mode as specified in EPCglobal Testing and Conformance.

Note: This version of LLRP standard recommends reader vendors to support RF mode table entries that support a single Tari value. Such table entries will have MinTariValue equal to MaxTariValue, and StepTariValue will be 0. Clients may choose such an RF mode table entry using C1G2RFControl parameter of C1G2InventoryCommand, by specify ModeIndex field of C1G2RFControl parameter to index of such a table entry and Tari field of C1G2RFControl parameter set to 0.

11 Reader Operation (RO)

This section presents the messages and the parameters used by the Client for specifying RO.

11.1 Messages

11.1.1 ADD_ROSPEC

An ADD_ROSPEC message communicates the information of a *ROSpec* to the Reader. LLRP supports configuration of multiple ROSpecs. Each ROSpec is uniquely identified using a ROSpecID, generated by the Client. The *ROSpec* starts at the Disabled state waiting for the ENABLE_ROSPEC message for the *ROSpec* from the Client, upon which it transitions to the Inactive state.

The Client SHALL add a ROSpec in a Disabled State – i.e., CurrentState field in the ROSpec Parameter (section *11.2.1*) SHALL be set to disabled. If the CurrentState value is different than disabled, an error SHALL be returned in the ADD_ROSPEC_RESPONSE (e.g. P_FieldError).

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

ADD_ROSPEC

ROSpec: ROSpec Parameter

11.1.2 ADD_ROSPEC_RESPONSE

This is the response by the Reader to an ADD_ROSPEC message. If all the parameters specified in the ADD_ROSPEC command are successfully set, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

ADD_ROSPEC_RESPONSE

Response: LLRPStatus Parameter

11.1.3 DELETE_ROSPEC

This command is issued by the Client to the Reader. This command deletes the ROSpec at the Reader corresponding to ROSpecID passed in this message.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

DELETE_ROSPEC

ROSpecID: Unsigned Integer. The identifier of the ROSpec to delete. 0 indicates to delete all ROSpecs.

11.1.4 DELETE_ROSPEC_RESPONSE

This is the response by the Reader to a DELETE_ROSPEC command. If there was a ROSpec corresponding to the ROSpecID that the Reader was presently executing, and the Reader was successful in stopping that execution, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

DELETE_ROSPEC_RESPONSE

Response: LLRPStatus Parameter

11.1.5 START_ROSPEC

This message is issued by the Client to the Reader. Upon receiving the message, the Reader starts the ROSpec corresponding to ROSpecID passed in this message, if the ROSpec is in the enabled state.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

START_ROSPEC

ROSpecID: Unsigned Integer. The identifier of the ROSpec to start.

Possible Values: 0 is disallowed.

11.1.6 START_ROSPEC_RESPONSE

This is the response by the Reader to a START_ROSPEC command. If there was a ROSpec corresponding to the ROSpecID in the enabled state, and the Reader was able to start executing that ROSpec, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message

START_ROSPEC_RESPONSE

Response: LLRPStatus Parameter

11.1.7 STOP_ROSPEC

This message is issued by the Client to the Reader. Upon receiving the message, the Reader stops the execution of the ROSpec corresponding to the ROSpecID passed in this message. STOP_ROSPEC overrides all other priorities and stops the execution. This basically moves the ROSpec's state to Inactive. This message does not the delete the ROSpec.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

STOP_ROSPEC

ROSpecID: Unsigned Integer. The identifier of the ROSpec to stop.

Possible Values: 0 is disallowed.

11.1.8 STOP_ROSPEC_RESPONSE

This is the response by the Reader to a STOP_ROSPEC command. If the Reader was currently executing the ROSpec corresponding to the ROSpecID, and the Reader was able to stop executing that ROSpec, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

STOP_ROSPEC_RESPONSE

Response: LLRPStatus Parameter

11.1.9 ENABLE_ROSPEC

This message is issued by the Client to the Reader. Upon receiving the message, the Reader moves the ROSpec corresponding to the ROSpecID passed in this message from the disabled to the inactive state.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

ENABLE_ROSPEC

ROSpecID: Unsigned Integer. The identifier of the ROSpec to enable. If set to 0, all ROSpecs are enabled.

11.1.10 ENABLE_ROSPEC_RESPONSE

This is the response by the Reader to a ENABLE_ROSPEC command. If there was a ROSpec corresponding to the ROSpecID, and the Reader was able to enable that ROSpec, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

ENABLE_ROSPEC_RESPONSE

Response: LLRPStatus Parameter

11.1.11 DISABLE_ROSPEC

This message is issued by the Client to the Reader. Upon receiving the message, the Reader moves the ROSpec corresponding to the ROSpecID passed in this message to the disabled state.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

DISABLE_ROSPEC

ROSpecID: Unsigned Integer. The identifier of the ROSpec to disable. If set to 0, all ROSpecs are disabled.

11.1.12 DISABLE_ROSPEC_RESPONSE

This is the response by the Reader to a DISABLE_ROSPEC command. If there was a ROSpec corresponding to the ROSpecID, and the Reader was able to disable that ROSpec, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

DISABLE_ROSPEC_RESPONSE

Response: LLRPStatus Parameter

11.1.13 GET_ROSPECS

This is the request from the Client to the Reader to retrieve all the ROSpecs that have been configured at the Reader.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_ROSPECS

11.1.14 GET_ROSPECS_RESPONSE

This is the response by the Reader to a GET_ROSPECS command. If there are no ROSpecs configured at the Reader, the response is just the LLRPStatus parameter with the success code. Else, a list of ROSpec parameter is returned by the Reader, along with the success code in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_ROSPECS_RESPONSE

Status: LLRPStatus Parameter

Response: List of <ROSpec Parameter> that are in the order in which they are added.

11.2 Parameters

11.2.1 ROSpec Parameter

This parameter carries the information of the Reader inventory and survey operation.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ROSpec Parameter

ROSpecID: Unsigned Integer; 0 is an illegal ROSpecID for a ROSpec.

Priority: Integer. Lower numbered priority values are given higher priority.

Possible Values: 0-7.

CurrentState: Integer

Possible Values:

```
Value   Definition
-----   ----------
 0      Disabled
 1      Inactive
 2      Active
```

ROBoundarySpec: ROBoundarySpec Parameter

ListOfSpecs: List of LLRP Parameters

Possible Values:

Each parameter can be either an <AISpec Parameter>, a <RFSurveySpec Parameter>, a <LoopSpec Parameter>, or a Custom Parameter.

ROReportSpec: ROReportSpec Parameter [optional] (section *14.2.1*)

11.2.1.1 ROBoundarySpec Parameter

This parameter carries the lifetime of the command, ROStartTrigger and ROStopTrigger parameters.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ROBoundarySpec Parameter

ROSpecStartTrigger: ROSpecStartTrigger Parameter

ROSpecStopTrigger: ROSpecStopTrigger Parameter

11.2.1.1.1 ROSpecStartTrigger Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ROSpecStartTrigger Parameter

ROSpecStartTriggerType: Integer

Possible Values:

```
Value   Definition
-----   ----------
  0    Null – No start trigger. The only way to start the
       ROSpec is with a START_ROSPEC from the Client.
  1    Immediate
  2    Periodic
  3    GPI
```

PeriodicTriggerValue: PeriodicTriggerValue Parameter [Optional]. This parameter SHALL be present when ROSpecStartTriggerType = 2.

GPITriggerValue: GPITriggerValue Parameter [Optional]. This parameter SHALL be present when ROSpecStartTriggerType = 3.

11.2.1.1.2 PeriodicTriggerValue Parameter

Periodic trigger is specified using UTC time, offset and period.

For one-shot inventory, period is set to 0, and for periodic inventory operation period > 0. If UTC time is not specified, the first start time is determined as (time of message receipt

+ offset), else, the first start time is determined as (UTC time + offset). Subsequent start

times = first start time + k * period (where, k > 0).

If the Reader does not support UTC clock (as indicated by HasUTCClockCapability), and it receives the UTC time as part of the PeriodicTriggerValue parameter from the Client, the Reader SHALL return an error.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter. Compliant Readers and Clients MAY implement the UTCTimestamp parameter.

PeriodicTriggerValue Parameter

UTC Time: <UTCTimestamp Parameter> [Optional]

Offset: Unsigned Integer. Time offset specified in milliseconds.

Period: Unsigned Integer. Time period specified in milliseconds

11.2.1.1.3 GPITriggerValue Parameter

This trigger is tied to an event on the General Purpose Input (GPI) of the Reader. The event is represented as a boolean type, and it is up to the internal implementation of the Reader to map exact physical event to a boolean type. For example, a 0 →1 and a 1 →0 transition on an input pin of the Reader could be mapped to a boolean true and a boolean false event respectively.

This trigger parameter has a timeout value field. The timeout is useful for specifying a fail-safe timeout when this trigger is used as a stop trigger. When the timeout is 0, it indicates that there is no timeout. When used as a start trigger, the timeout value SHALL be ignored.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter. Readers that do not support GPIs SHALL return zero for numGPIs in the capabilities discovery. If the Client sets up the GPI trigger for such a Reader, the Reader SHALL send an error message for the ADD_ROSPEC message and not add the ROSpec.

GPITriggerValue Parameter

GPIPortNum: Unsigned Short Integer.

Possible Values: 1-65535. Zero is invalid.

GPIEvent: Boolean. The Boolean value that causes a GPI event to trigger.

Timeout: Unsigned Integer. Trigger timeout in milliseconds. If set to zero, it indicates there is no timeout.

11.2.1.1.4 ROSpecStopTrigger Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ROSpecStopTrigger Parameter

ROSpecStopTriggerType: Integer

Possible Values:

```
         Value   Definition
         -----   ----------
           0    Null – Stop when all Specs are done (including any
                looping as required by a LoopSpec parameter), or when
                preempted, or with a STOP_ROSPEC from the Client.
           1    Duration – Stop after DurationTriggerValue
                milliseconds, or when all Specs are done
                (including any looping as required by a LoopSpec
                parameter), or when preempted, or with a STOP_ROSPEC
                from the Client.
           2    GPI with a timeout value – Stop when a GPI "fires", or
                after Timeout milliseconds, or when all Specs are done
                (including any looping as required by a LoopSpec
                parameter), or when preempted, or with a STOP_ROSPEC
                from the Client.
```

DurationTriggerValue: Duration in milliseconds. This field is ignored when ROSpecStopTriggerType != 1.

GPITriggerValue: GPITriggerValue Parameter [Optional]. This parameter SHALL be present when ROSpecStopTriggerType = 2.

11.2.2 AISpec Parameter

This parameter defines antenna inventory operations.

Compliance Requirement: Compliant Readers and Clients SHALL implement this parameter.

AISpec Parameter

AISpecStopTrigger: <AISpecStopTrigger Parameter>

AntennaIDs: Short Array. If this set contains an antenna ID of zero, this AISpec will utilise all the antennas of the Reader.

InventoryParameterSpecs: <List of InventoryParameterSpec Parameter>
Custom Extension Point List: List of <custom Parameter> [Optional]

11.2.2.1 AISpecStopTrigger Parameter

This parameter defines the stop (i.e., terminating boundary) of an antenna inventory operation.

Compliance Requirement: Compliant Readers and Clients SHALL implement this parameter. If a Reader reports NumGPIs (see GPIO Capabilities Parameter) greater than zero, then the Reader SHALL support GPI Trigger.

AISpecStopTrigger Parameter

AISpecStopTriggerType: Integer

Possible Values:

```
Value   Definition
-----   ----------
  0    Null – Stop when ROSpec is done.
  1    Duration
  2    GPI with a timeout value
  3    Tag observation
```

Duration Trigger: Unsigned Integer. Duration of AISpec in milliseconds. This field SHALL be ignored when AISpecStopTriggerType != 1.

GPI Trigger : GPITrigger value Parameter [Optional]. This field SHALL be present when AISpecStopTriggerType = 2.

TagObservation Trigger : TagObservation Trigger Parameter [Optional]. This field SHALL be present when AISpecStopTriggerType = 3.

11.2.2.1.1 TagObservationTrigger Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

Tag ObservationTrigger Parameter

TriggerType: Integer

Possible Values:

```
Value   Modulation
-----   ----------
  0    Upon seeing N tag observations, or timeout. The
       definition of an "observation" is vendor specific.
  1    Upon seeing no more new tag observations for T ms, or
       timeout. The definition of an "observation" is vendor
       specific.
  2    N attempts to see all tags in the FOV, or timeout.
  3    Upon seeing N unique tag observations, or timeout.
  4    Upon seeing no more new unique tag observations for T ms,
       or timeout.
```

NumberOfTags: Unsigned Short Integer. This field SHALL be ignored when TriggerType != 0 and TriggerType != 3.

NumberOfAttempts; Unsigned Short Integer. This field SHALL be ignored when TriggerType != 2.

T : Unsigned Short Integer. Idle time between tag responses in milliseconds. This field SHALL be ignored when TriggerType != 1 and TriggerType != 4.

Timeout : Unsigned Integer; Trigger timeout value in milliseconds. If set to zero, it indicates that there is no timeout.

11.2.2.2 InventoryParameterSpec Parameter

This parameter defines the inventory operation to be performed at all antennas specified in the corresponding AISpec. This parameter is composed of an InventoryParameterSpecID, a ProtocolID, and zero or more optional antenna configuration parameters. Antenna configurations for antennas not indicated by the AntennaIDs within the AISpec are ignored by the reader.

Compliance Requirement: Compliant Readers and Clients SHALL implement this parameter.r

InventoryParameterSpec Parameter

InventoryParameterSpecID: Unsigned Short Integer. 0 is illegal.

ProtocolID: Integer. Enumeration based on Table 4.

AntennaConfiguration: List of <AntennaConfiguration Parameter> (section *13.2.6*) [Optional]

Custom Extension Point List: List of <Custom Parameter> [Optional]

11.2.3 RFSurveySpec Parameter

This parameter defines RF Survey operations. RF Survey is an operation during which the Reader performs a scan and measures the power levels across a set of frequencies at an antenna. This parameter defines the identifier of the antenna where this survey is to be performed, the duration of the survey operation (specified via stop trigger), and the range of frequencies to measure power levels of.

Compliance Requirement: Compliant Readers and Clients MAY implement this parameter.

RFSurveySpec Parameter

Antenna ID: Unsigned Short Integer.

Possible Values: 1 to N, where N is the maximum number of antennas supported by the Reader.

RFSurveySpecStopTrigger: RFSurveySpecStopTrigger parameter

StartFrequency: Unsigned Integer. The start (lower bound) frequency to survey specified in kHz. The Reader's supported frequency range is reported via the RFSurveyFrequencyCapabilities in the UHFBandCapabilities parameter of the GET_READER_CAPABILITIES_RESPONSE message.

EndFrequency: Unsigned Integer in kHz. The end (upper bound) frequency to survey specified in kHz.

Custom Extension Point List: List of <custom Parameter> [Optional]

11.2.3.1 RFSurveySpecStopTrigger Parameter

This parameter defines the stop trigger for RF Survey operations.

Compliance Requirement: Compliant Readers and Clients MAY implement this parameter.

RFSurveySpecStopTrigger Parameter

StopTriggerType: Integer

Possible Values:

```
        Value   Definition
        -----   ----------
          0      Null
          1      Duration
          2      N iterations through the frequency range
```

Duration: Unsigned Integer; The maximum duration of the RFSurvey operation specified in milliseconds. This field SHALL be ignored when StopTriggerType != 1. When StopTriggerType = 1, the value SHALL be greater than zero.

N: Unsigned Integer. The maximum number of iterations through the specified frequency range. This field SHALL be ignored when StopTriggerType != 2. When StopTriggerType = 2, the value SHALL be greater than zero.

11.2.4 LoopSpec Parameter

This parameter instructs the Reader to loop execution of the ROSpec, starting at SpecIndex 1. If present in a ROSpec's ListOfSpecs, this parameter SHALL be the final parameter in the ListOfSpecs, and at least one AISpec, RFSurveySpec, or Custom parameter SHALL preceed this parameter in the ListOfSpecs.

Compliance Requirement: Compliant Readers and Clients SHALL implement this parameter.

LoopSpec Parameter

LoopCount: Unsigned Integer. The number of times to loop through the ROSpec's ListOfSpecs. A value of 0 means unlimited (execute ListOfSpecs until the ROSpecStopTrigger fires).

12 Access Operation

This section presents the messages and the parameters used by the Client for specifying access operation

12.1 Messages

12.1.1 ADD_ACCESSSPEC

This command creates a new AccessSpec at the Reader. The *AccessSpec* starts at the Disabled state waiting for the ENABLE_ACCESSSPEC message for the *AccessSpec* from the Client, upon which it transitions to the Active state. The AccessSpecID is generated by the Client.

The Client SHALL add an AccessSpec in a Disabled State – i.e., CurrentState field in the AccessSpec Parameter (section *12.2.1*) SHALL be set to false. If the CurrentState value is different than false, an error SHALL be returned in the ADD_ACCESSSPEC_RESPONSE (e.g. P_FieldError).

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

ADD_ACCESSSPEC

AccessSpec: AccessSpec parameter

12.1.2 ADD_ACCESSSPEC_RESPONSE

This is the response by the Reader to an ADD_ACCESSSPEC command. If the parameters passed in that ADD_ACCESSSPEC command were successfully accepted and set at the Reader, then the success code is returned in the LLRPStatus parameter. However, if the *AccessSpec* was not successfully created at the Reader, the Reader sends a LLRPStatus parameter describing the error in the message.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

ADD_ACCESSSPEC_RESPONSE

Response: LLRPStatus Parameter

12.1.3 DELETE_ACCESSSPEC

This command is issued by the Client to the Reader. The Reader deletes the AccessSpec corresponding to the AccessSpecId, and this AccessSpec will stop taking effect from the next inventory round.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

DELETE_ACCESSSPEC

AccessSpecID : Unsigned Integer.

Possible Values: If set to 0, all AccessSpecs are deleted.

12.1.4 DELETE_ACCESSSPEC_RESPONSE

This is the response by the Reader to a DELETE_ACCESSSPEC command. If there was an AccessSpec at the Reader corresponding to the AccessSpecID passed in the DELETE_ACCESSSPEC command, and the Reader was successful in deleting that AccessSpec, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

DELETE_ACCESSSPEC_RESPONSE

Response: LLRPStatus Parameter

12.1.5 ENABLE_ACCESSSPEC

This message is issued by the Client to the Reader. Upon receiving the message, the Reader moves the AccessSpec corresponding to the AccessSpecID in this message from the Disabled state to the Active state. The Reader executes this access-spec until it gets a DISABLE_ACCESSSPEC or a DELETE_ACCESSSPEC from the Client. The AccessSpec takes effect with the next (and subsequent) inventory rounds.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

ENABLE_ACCESSSPEC

AccessSpecID: Unsigned Integer. If set to 0, all AccessSpecs are enabled.

12.1.6 ENABLE_ACCESSSPEC_RESPONSE

This is the response by the Reader to a START_ACCESSSPEC command. If there was an AccessSpec corresponding to the AccessSpecID, and the Reader was able to move that AccessSpec from the disabled to the active state, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

ENABLE_ACCESSSPEC_RESPONSE

Response: LLRPStatus Parameter

12.1.7 DISABLE_ACCESSSPEC

This message is issued by the Client to the Reader. Upon receiving the message, the Reader stops the execution of the AccessSpec corresponding to AccessSpecID in this message. This basically moves the AccessSpec's state to Disabled. This message does not delete the AccessSpec. The AccessSpec will stop taking effect from the next inventory round.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

DISABLE_ACCESSSPEC

AccessSpecID: Unsigned Integer. If set to 0, all AccessSpecs are disabled.

12.1.8 DISABLE_ACCESSSPEC_RESPONSE

This is the response by the Reader to a STOP_ACCESSSPEC command. If the Reader was currently executing the AccessSpec corresponding to the AccessSpecID, and the Reader was able to disable that AccessSpec, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

DISABLE_ACCESSSPEC_RESPONSE

Response: LLRPStatus Parameter

12.1.9 GET_ACCESSSPECS

This is the request from the Client to the Reader to retrieve all the AccessSpecs that have been configured at the Reader.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_ACCESSSPECS

12.1.10 GET_ACCESSSPECS_RESPONSE

This is the response by the Reader to a GET_ACCESSSPECS command. If there are no AccessSpecs configured at the Reader, the response is just the LLRPStatus parameter with the success code. Else, a list of <AccessSpecID, AccessSpec parameter> is returned by the Reader, along with the LLRPStatus parameter containing the success code. The order of the AccessSpecs listed in the message is normatively the order in which the AccessSpecs were created at the Reader.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_ACCESSSPECS_RESPONSE

Status: LLRPStatus Parameter

Response: List of <AccessSpec Parameter>. The ordering of the AccessSpecs in this list is the order in which the AccessSpecs were created at the Reader.

12.1.11 CLIENT_REQUEST_OP

This message is sent by the Reader to the Client upon executing a ClientRequestOpSpec OpSpec (section 12.2.1.2.1). This message carries the TagReportData (section *14.2.3*) that contains information collected for the tag which includes singulation results and the results of OpSpecs executed till that point.

Compliance requirement: Compliant Readers and Clients MAY implement this message.

CLIENT_REQUEST_OP

TagReport: <TagReportData Parameter> (section *14.2.3*))

12.1.12 CLIENT_REQUEST_OP_RESPONSE

This is the response by the Client to the Reader. This is in response to the CLIENT_REQUEST_OP sent by the Reader due to the execution of a ClientRequestOpSpec. This is a response to the CLIENT_REQUEST_OP message; thus, the messageID in this message is the messageID of the CLIENT_REQUEST_OP.

Compliance requirement: Compliant Readers and Clients MAY implement this message. Readers that do not support ClientRequestOpSpec MAY ignore this message.

CLIENT_REQUEST_OP_RESPONSE

Response: ClientRequestResponse Parameter

12.2 Parameters

12.2.1 AccessSpec Parameter

This parameter carries information of the Reader access operation.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

AccessSpec Parameter

AccessSpecID: Unsigned Integer. 0 is illegal.

Antenna ID: Unsigned Short Integer. If 0, this spec is operational on all antennas.

ProtocolID: Integer.
Possible Values: Enumeration based on Table 4.

CurrentState: Boolean. This is the current state of the AccessSpec. false = Disabled, true = Active.

ROSpecID: Unsigned Integer. If 0, this spec is operational for all ROSpecs.

AccessSpecStopTrigger: AccessSpecStopTrigger Parameter

Access Command Operation: AccessCommand Parameter

AccessReportSpec: AccessReportSpec Parameter [Optional]

Custom Extension Point List: List of <custom Parameter> [Optional]

12.2.1.1 AccessSpecStopTrigger Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

AccessSpecStopTrigger Parameter

AccessSpecStopTriggerType: Integer

Possible Values:

```
Value   Definition
-----   ----------
  0     Null - No stop trigger defined.
  1     Operation count
```

OperationCountValue: Unsigned Short Integer. A count to indicate the number of times this Spec is executed before it is deleted. If set to 0, this is equivalent to no stop trigger defined.

12.2.1.2 AccessCommand Parameter

This parameter defines the air protocol access-specific settings. It contains a TagSpec and an OpSpec Parameter. The TagSpec specifies the tag filters in terms of air protocol specific memory capabilities (e.g., memory banks, pointer and length). The OpSpec specifies all the details of the operations required for the air protocol specific access operation commands.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

AccessCommand Parameter

TagSpec: LLRP Parameter *Possible*

Values:

Each air protocol's TagSpec parameter is expressed as a different LLRP Parameter. The air protocol specific TagSpec LLRP Parameters are defined in section 16.1 This field carries a single TagSpec parameter corresponding to the air protocol referenced by the ProtocolID in the AccessSpec Parameter.

OpSpec: List of LLRP Parameters

Possible Values:

Each parameter can be either an air protocol specific OpSpec LLRP Parameter, a <ClientRequestOpSpec Parameter>, or a Custom Parameter.

Regarding the air protocol specific OpSpec LLRP Parameter: Each air protocol's OpSpec parameter is expressed as a different LLRP Parameter. The air protocol specific OpSpec LLRP Parameters are defined in section 16.1. The list of OpSpecs in this field is comprised of OpSpec parameters corresponding to the air protocol referenced by the ProtocolID in the AccessSpec Parameter.

Custom Extension Point List: List of <Custom Parameter> [Optional]

In case there are multiple AccessSpecs that get matched during a TagSpec lookup, the Reader SHALL only execute the first enabled AccessSpec that matches, where the ordering of the AccessSpecs is the order in which the AccessSpecs were created by the Client.

The order of execution of OpSpecs within an AccessSpec is the order in which the OpSpecs were set up in the AccessSpec. If an OpSpec execution fails, the Reader SHALL stop the execution of the AccessSpec.

12.2.1.3 C1G2TagSpec Parameter

This parameter describes the target tag population on which certain operations have to be performed. This Parameter is similar to the selection C1G2Filter Parameter described earlier. However, because these tags are stored in the Reader's memory and ternary comparisons are to be allowed for, each bit i in the target tag is represented using 2 bits - bit i in mask, and bit i in tag pattern. If bit i in the mask is zero, then bit i of the target tag is a don't care (X); if bit i in the mask is one, then bit i of the target tag is bit i of the tag pattern. For example, "all tags" is specified using a mask length of zero.

This parameter can carry up to two tag patterns. If more than one pattern is present, a Boolean AND is implied. Each tag pattern has a match or a non-match flag, allowing (A and B,!A and B, !A and !B, A and !B), where A and B are the tag patterns.

The tagSpec contains:

- TagPattern1
- TagPattern2

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2TagSpec Parameter

TagPattern1: <C1G2TargetTag Parameter>

TagPattern2: <C1G2TargetTag Parameter> [optional]

12.2.1.3.1 C1G2TargetTag Parameter

If Length is zero, this pattern will match all tags regardless of MB, pointer, mask and data.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2TargetTag Parameter

MB: Integer. Memory bank.

Possible Values: 0-3.

Pointer: Unsigned Short Integer. The address of the first (msb) bit against which to apply the Tag Mask and compare with the value.

TagMask : Bit array

TagData: Bit array

Match: Boolean

12.2.1.3.2 C1G2 OpSpec Parameters

This section describes the C1G2 specific OpSpec parameters that are sent as part of the AccessSpec. Each OpSpec parameter has an OpSpecID that is used when reporting results of the operation.

C1G2Read Parameter

MB is the memory bank to use. WordPtr is the starting word address. WordCount is the number of 16-bit words to be read. Following is text reproduced from the C1G2 specification regarding WordCount=0. [If WordCount = 0, the tag backscatters the contents of the chosen memory bank starting at WordPtr and ending at the end of the bank, unless MB = 1, in which case the Tag shall backscatter the EPC memory contents starting at WordPtr and ending at the length of the EPC specified by the first 5 bits of the PC if WordPtr lies within the EPC, and shall backscatter the EPC memory contents starting at WordPtr and ending at the end of EPC memory if WordPtr lies outside the EPC.

Access Password is the password used by the Reader to transition the tag to the secure state so that it can read protected tag memory regions. For example, the Tag's Reserved memory is locked but not permalocked, meaning that the Reader must issue the access password and transition the Tag to the secured state before performing the read operation.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2Read Parameter

OpSpecID: Unsigned Short Integer

MB: Integer. Memory bank.

Possible Values: 0-3

WordPtr: Unsigned Short Integer. The word addresss of the first word to read from the chosen memory bank.

WordCount: Unsigned Short Integer

AccessPassword: Unsigned Integer

C1G2Write Parameter

MB is the memory bank to use. WordPtr is the starting word address. Write Data is the data to be written to the tag. Word Count is the number of words to be written. Depending on the word count, the Reader may have to execute multiple C1G2 air protocol Write commands. Access Password is the password used by the Reader to transition the tag to the secure state so that it can write to protected tag memory regions.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2Write Parameter

OpSpecID : Unsigned Short Integer

MB: Integer. Memory bank.

Possible Values: 0-3

WordPtr: Unsigned Short Integer. The word addresss of the first word to be written to the chosen memory bank.

Write Data: Short array. The data to write to the chosen memory bank.

AccessPassword: Unsigned Integer

C1G2Kill Parameter

Kill Password is the value of the kill password to be used or set.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2Kill Parameter

OpSpecID : Unsigned Short Integer

Kill Password: Unsigned Integer

C1G2Lock Parameter

This parameter contains the definition of the access privilege updates (read/write/permalock) to be performed in various locations of the memory. The five data fields for which we can define access control using the lock command are: Kill Password, Access Password, EPC memory, TID memory and User memory. The access privilege updates are expressed as a list of C1G2LockPayload Parameters, one for each memory location.

The Access Password provides the password to enter the *secured* state. A Reader can perform a lock operation on a tag only if the tag is in the *secured* state. The tag enters the secured state only using the Access Password (if a non-zero value).

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2Lock Parameter

OpSpecID : Unsigned Short Integer

LockCommandPayloadList: List of <C1G2LockPayload Parameter>

Access Password: Unsigned Integer

C1G2LockPayload Parameter

This parameter contains the definition of the access privilege updates (read/write/permalock) to be performed for a single location of the tag memory. The five data fields for which we can define access control using the lock command are: Kill Password, Access Password, EPC memory, TID memory and User memory.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2LockPayload Parameter

OpSpecID : Unsigned Short Integer

Privilege: Integer. Value indicates the access privilege to be applied.
Possible Values:

```
Value   Access Privilege
-----   ----------------
   0    Read/Write
   1    Permalock
   2    Permaunlock
   3    Unlock
```

DataField: Unsigned Integer. Value indicates to which data field the access privilege will be applied.
Possible Values:

```
Value   Field
-----   ------
   0    Kill Password
   1    Access Password
   2    EPC Memory
   3    TID Memory
   4    User Memory
```

C1G2BlockErase Parameter

MB is the memory bank to use. WordPtr is the starting word address. Word Count is the number of 16-bit words to be read. Access Password is the password used by the Reader to transition the tag to the secure state so that it can erase protected tag memory regions.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2BlockErase SHALL set CanSupportBlockErase to false in C1G2LLRPCapabilities. If such a Reader receives an ADD_ACCESSSPEC with an AccessSpec that contained this OpSpec parameter, the Reader SHALL return an error for that message and not add the AccessSpec.

C1G2BlockErase

Parameter **OpSpecID** :

Unsigned Short Integer **MB**:

Integer. Memory bank.

Possible Values: 0-3

WordPtr: Unsigned Short Integer. Word address of first word to be erased.

Word Count: Unsigned Short Integer. Number of words to erase.

Access Password: Unsigned Integer

C1G2BlockWrite Parameter

MB is the memory bank to use. WordPtr is the starting word address. Word Count is the number of 16-bit words to be written. Depending on the word count, the Reader may have to execute multiple C1G2 air protocol block write commands. Write Data is the data to be written to the tag. Access Password is the password used by the Reader to transition the tag to the secure state so that it can write to protected tag memory regions.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2BlockWrite SHALL set CanSupportBlockWrite to false in

C1G2LLRPCapabilities. If such a Reader receives an ADD_ACCESSSPEC with an AccessSpec that contained this OpSpec parameter, the Reader SHALL return an error for that message and not add the AccessSpec.

C1G2BlockWrite

Parameter **OpSpecID** :

Unsigned Short Integer **MB**:

Integer. Memory bank.

Possible Values: 0-3

WordPtr: Unsigned Short Integer. Word address of first word to be written.

Write Data: Short array

Access Password: Unsigned Integer

C1G2BlockPermalock Parameter

MB (MemBank), BlockPointer, and BlockMask fields are equivalent to the Class1 Gen2 BlockPermalock fields. AccessPassword is the password required to put the tag into the secured state as required to execute the BlockPermalock command.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2BlockPermalock SHALL set CanSupportBlockPermalock to false in C1G2LLRPCapabilities. If such a Reader receives an ADD_ACCESSSPEC with an AccessSpec that contained this OpSpec parameter, the Reader SHALL return an error for that message and not add the AccessSpec.

C1G2BlockPermalockParameter

OpSpecID : Unsigned Short Integer

MB: Integer. Memory bank.

Possible Values: 0-3

BlockPointer: Unsigned Short Integer. Specifies the starting address for BlockMask in units of 16 blocks.

BlockMask: Unsigned Short Integer Array. The blocks to lock, starting at BlockPointer and ending ((16*(BlockMask array length)) – 1) blocks later.

AccessPassword: Unsigned Integer

C1G2GetBlockPermalockStatus Parameter

This parameter retrieves the BlockPermalock status from a tag. MB (MemBank), BlockPointer, and BlockRange fields are equivalent to the Class1 Gen2 BlockPermalock fields. AccessPassword is the password required to put the tag into the secured state as required to execute the BlockPermalock command.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2BlockPermalock SHALL set CanSupportBlockPermalock to false in C1G2LLRPCapabilities. If such a Reader receives an ADD_ACCESSSPEC with an AccessSpec that contained this OpSpec parameter, the Reader SHALL return an error for that message and not add the AccessSpec.

C1G2GetBlockPermalockStatus Parameter

OpSpecID : Unsigned Short Integer

MB: Integer. Memory bank.

Possible Values: 0-3

BlockPointer: Unsigned Short Integer. Specifies the starting address to retrieve in units of 16 blocks.

BlockRange: Unsigned Short Integer. The range of blocks to retrieve, starting at BlockPointer and ending ((16*BlockRange) – 1) blocks later.

AccessPassword: Unsigned Integer

C1G2Authenticate

Clients can initialise crypto operations with supported tag using C1G2Authenticate OpSpec parameter.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2Authenticate

Parameter OpSpecID:

Unsigned Short Integer

S: Integer: Store or Send the reply for authenticate command

Possible Values:

0. Store
1. Send

L: Integer: Enable or disable inclusion of length in tag's reply

Possible Values:

0. Omit length from reply
1. Include length in reply

CSI: Integer: Specifies crypto suite to use for the Challenge

Message: Bit array: Crypto message with parameters for authentication

See C1G2AuthenticateStatusOpSpecResult parameter in *14.2.9.10.10* for response to C1G2Authenticate operation

C1G2AuthComm

Clients can perform authenticated communication with supported tag using C1G2AuthCommOpSpec parameter.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2AuthComm

Parameter OpSpecID:

Unsigned Short Integer

L: Integer

Enable or disable inclusion of length in tag's reply

Possible Values:

0. Omit length from tag reply
1. Include length in tag reply

Message: Bit array

Crypto message with parameters for authenticated communication

See C1G2AuthCommStatusOpSpecResult parameter in *14.2.9.10.10* for response to C1G2AuthComm operation

C1G2SecureComm

Clients can perform secure communication with supported tag using C1G2SecureComm OpSpec parameter.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2SecureComm

Parameter OpSpecID:

Unsigned Short Integer

S: Integer. Send or store reply

Possible values:

0: Send reply

1: Store reply

L: Integer. Enable or disable inclusion of length in tag's reply

Possible Values:

0. Omit length from tag reply
1. Include length in tag reply

Message: Bit array :Crypto message with parameters for secure communication

See C1G2SecureCommStatusOpSpecResult parameter in *14.2.9.10.10* for response to C1G2SecureComm operation

C1G2ReadBuffer

Clients can read response of a crypto operation from tag that supports *ReadBuffer* command using C1G2SecureComm OpSpec parameter.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2ReadBuffer

Parameter OpSpecID:

Unsigned Short Integer

W: Integer. Word address pointer to read

N: Integer. Number of bits to read

See C1G2ReadBufferOpSpecResult parameter in 14.2.9.10.10 for response to C1G2ReadBuffer operation.

C1G2Untraceable Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2Untraceable SHALL set CanSupportUntraceable to false in C1G2LLRPCapabilities. If such a Reader receives an ADD_ACCESSSPEC with an AccessSpec that contained this OpSpec parameter, the Reader SHALL return an error for that message and not add the AccessSpec.

C1G2Untraceable Parameter

OpSpecID : Unsigned Short Integer

U: Unsigned character. Value indicates state of U bit in XPC_W1
Possible Values:

```
          Value   State
          -----   ----------
             0    De-assert U bit
             1    Assert U bit
```

EPC: Unsigned Character. Show/Hide EPC and set new length for EPC

MSB bit: show/hide EPC.

5 LSB bits: New EPC length field (new L bits for Stored PC)

TID: Integer. 2 bits to show/hide TID memory

Possible Values:

```
          Value   Meaning
          -----   ----------
             0    Hide none
             1    Hide some
             3    Hide all
```

User: Integer. 1 bit to show/hide USER memory

Possible Values:

```
          Value   Meaning
          -----   ----------
             0    View
             1    Hide
```

Range: Integer. 2 bits to control tag's operating range

Possible Values:

```
Value   Meaning
-----   ----------
  0     Normal
  1     Toggle temporarily
  2     Reduced
```

AccessPassword: Unsigned Integer

C1G2KeyUpdate

Clients can update secure communication keys used in tag with C1G2KeyUpdate OpSpec parameter.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2KeyUpdate

Parameter OpSpecID:

Unsigned Short Integer

S: Integer: Store or Send the reply for KeyUpdate command

Possible Values:

0. Store
1. Send

L: Integer: Enable or disable inclusion of length in tag's reply

Possible Values:

0. Omit length from reply
1. Include length in reply

KeyID: Integer: Specifies identity of key to be updated

Message: Bit array: Crypto message with parameters for authentication

See C1G2KeyUpdateStatusOpSpecResult parameter in *14.2.9.10.10* for response to C1G2KeyUpdate operation

C1G2TagPrivilege

Clients can read or modify key privileges using C1G2TagPrivilege OpSpec parameter.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2TagPrivilege

Parameter OpSpecID:

Unsigned Short Integer

S: Integer: Store or Send the reply for TagPrivilege command

Possible Values:

0. Store

1. Send

L: Integer: Enable or disable inclusion of length in tag’s reply

Possible Values:

0. Omit length from reply

1. Include length in reply

A: Integer: Action

Possible Values:

0. Read

1. Modify

T: Integer: Target

Possible Values:

0. Access password

1. Key

KeyID: Integer: Specifies identity of key

Privilege: Bits: Privilege bit mask. Refer Class 1 Gen 2 Air Interface Protocol Standard for details.

See C1G2TagPrivilegeStatusOpSpecResult parameter in *14.2.9.10.10* for response to C1G2TagPrivilege operation

ClientRequestOpSpec Parameter

This parameter is sent as part of the possible values for the AccessSpec OpSpec list. One or more ClientRequestOpSpec operations may be performed on a tag in succession. Upon executing a ClientRequestOpSpec Parameter, a Reader will immediately send the CLIENT_REQUEST_OP message to the Client. This CLIENT_REQUEST_OP message carries the TagReportData (section *14.2.3*)_*bookmark159* that contains information collected for the tag which includes singulation results and the results of OpSpecs executed till that point.

A global timeout is associated with this request. If the Client does not return a ClientRequestResponse within the *ClientRequestOpSpecTimeout* (LLRP Capabilities) period, or the AirProtocolOpSpec List is empty in the ClientRequestResponse, the execution of the AccessSpec is cancelled.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support ClientRequestOpSpec SHALL set SupportClientRequestOpSpec to false in LLRPCapabilities. If such a Reader receives an ADD_ACCESSSPEC with an AccessSpec that contained this OpSpec parameter, the Reader SHALL return an error for that message, and not add the AccessSpec.

ClientRequestOpSpec Parameter

OpSpecID: Unsigned Short Integer.

Possible Values: 0 is an illegal value.

12.2.2 ClientRequestResponse Parameter

This parameter describes the list of OpSpecs that the Reader has to execute on the tag for which a Client request was initiated. The AccessSpecID is the identifier of the AccessSpec that had the Client request; the EPC data is the singulated data of the tag for which this Client request was initiated. The AirProtocolOpSpec list contained in the ClientRequestResponse SHALL be processed as the next OpSpecs sent over the air interface. If the AirProtocolOpSpec List is empty, then the execution of the AccessSpec specified by AccessSpecID is cancelled.

Compliance requirement: Compliant Readers MAY implement this parameter. Readers that do not support ClientRequestOpSpec MAY ignore this parameter.

ClientRequestResponse Parameter

AccessSpecID: Unsigned Integer. The ID of the AccessSpec that triggered this request.

EPCdata: <EPCData Parameter>. The electronic product code of the RFID tag that triggered this request.

AirProtocolOpSpecList: List of LLRP OpSpec Parameter.

Possible Values:

Each air protocol's OpSpec parameter is expressed as a different LLRP Parameter. The air protocol specific OpSpec LLRP Parameters are defined in section 16.1. This field carries a list of OpSpec parameters corresponding to the air protocol referenced by ProtocolID in the AccessSpec that generated the Client request.

13 Reader Device Configuration

This section contains the messages and parameters for getting and setting configuration.

13.1 Messages

13.1.1 GET_READER_CONFIG

This command is issued by the Client to get the current configuration information of the Reader. The Requested Data passed in the command represents the parameter(s) of interest to the Client that has to be returned by the Reader.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_READER_CONFIG

RequestedData : Integer

Possible Values:

```
Value        Requested Data
------       ----------------
  0          All
  1          Identification
  2          AntennaProperties
  3          AntennaConfiguration
  4          ROReportSpec
  5          ReaderEventNotificationSpec
  6          AccessReportSpec
  7          LLRPConfigurationStateValue
```

```
        8          KeepaliveSpec
        9          GPIPortCurrentState
        10         GPOWriteData
        11         EventsAndReports
```

AntennaID: Unsigned Short Integer. This field is ignored when RequestedData != 0 or 2 or 3. If the AntennaID is 0, get antenna information (AntennaProperties, AntennaConfiguration) for all antennas.

GPIPortNum: Unsigned Short Integer. This field is ignored when RequestedData != 0 or 9. If the GPIPortNum is 0, get GPI port current state for all GPI ports.

GPOPortNum: Unsigned Short Integer. This field is ignored when RequestedData != 0 or 10. If the GPOPortNum is 0, get GPO port current state for all GPO ports.

Custom Extension Point List: List of <custom Parameter> [Optional]

13.1.2 GET_READER_CONFIG_RESPONSE

This is the response by the Reader to the GET_READER_CONFIG message. The response is the LLRPStatus Parameter and the list of configuration parameters based on the RequestedData in GET_READER_CONFIG. If the GET_READER_CONFIG message did not have any errors, the success code is returned in the LLRPStatus parameter, and in addition the requested configuration parameters are returned. If there is an error, the appropriate error code is returned in the LLRPStatus parameter. The response contains at most one instance of each configuration parameter except for two cases, which are as follows:

- If RequestedData is 0, 2 or 3, and AntennaID is set to 0 in the GET_READER_CONFIG message, the Reader SHALL return one instance of AntennaProperties Parameter or AntennaConfiguration Parameter per requested antenna.
- If RequestedData is 0 or 9 (10), and GPIPortNum (GPOPortNum) is set to 0 in the GET_READER_CONFIG message, and, if the Reader supports GPI (GPO), the Reader SHALL return one instance of GPIPortCurrentState (GPOWriteData) Parameter per requested GPI Port (GPO Port).

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_READER_CONFIG_RESPONSE

Status: LLRPStatus Parameter

Response Data: Set of LLRP Parameters.

Possible Values: The possible members are zero or more of

{< LLRPConfigurationStateValue Parameter>,

<ReaderEventNotificationSpec Parameter>,

<Antenna Properties Parameter>,

<Antenna Configuration Parameter>,

<ROReportSpec Parameter>,

<AccessReportSpec Parameter>,

<Identification Parameter>,

<KeepaliveSpec Parameter>,

<GPIPortCurrentState Parameter>,

<GPOWriteData Parameter>,

<EventsAndReports Parameter>

}

Custom Extension Point List: List of <custom Parameter> [Optional]

13.1.3 SET_READER_CONFIG

This command is issued by the Client to the Reader. This command sets the Reader configuration using the parameters specified in this command. Values passed by the SET_READER_CONFIG SHALL apply for the duration of the LLRP connection, or until the values are changed by additional SET_READER_CONFIG messages.

For example, ROReportSpec defines the reporting of ROReport format and trigger for a ROSpec. ROReportSpec sent as part of SET_READER_CONFIG becomes the default ROReportSpec for the Reader. A ROReportSpec sent as part of ROSpec in the ADD_ROSPEC command overrides the default value for that ROSpec. However, in cases where there is no ROReportSpec specified in a ROSpec sent as part of ADD_ROSPEC, that particular ROSpec inherits the default ROReportSpec.

The data field ResetToFactoryDefault informs the Reader to set all configurable values to factory defaults before applying the remaining parameters.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

SET_READER_CONFIG

ResetToFactoryDefault: Boolean. If true, the Reader will set all configurable values to factory defaults before applying the remaining parameters.

Configuration Data: Set of LLRP Parameters

Possible Values: The possible members of the set are

{<ReaderEventNotificationSpec Parameter>,

<Antenna Properties Parameter>,

<Antenna Configuration Parameter>,

<ROReportSpec Parameter>,

<AccessReportSpec Parameter>,

<KeepaliveSpec Parameter>,

<GPOWriteData Parameter>,

<GPIPortCurrentState Parameter>,

<EventsAndReports Parameter>}

Custom Extension Point List: List of <custom Parameter> [Optional]

13.1.4 SET_READER_CONFIG_RESPONSE

This is the response by the Reader to a SET_READER_CONFIG command. If all the parameters specified in the SET_READER_CONFIG command are successfully set, then the success code is returned in the LLRPStatus parameter. If there is an error, the appropriate error code is returned in the LLRPStatus parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

SET_READER_CONFIG_RESPONSE

Response: < LLRPStatus Parameter>

13.1.5 CLOSE_CONNECTION

This command is issued by the Client to the Reader. This command instructs the Reader to gracefully close its connection with the Client. Under normal operating conditions, a Client SHALL attempt to send this command before closing an LLRP connection. A Client should wait briefly for the Reader to respond with a CLOSE_CONNECTION_RESPONSE.

Upon receipt of this command, the Reader SHALL respond with the CLOSE_CONNECTION_REPONSE message and it should then attempt to close the connection between the Reader and Client.

Having executed a CLOSE_CONNECTION command, a Reader MAY persist its configuration state as defined by the ReaderConfigurationStateValue parameter specified in section *13.2.1*.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

CLOSE_CONNECTION

13.1.6 CLOSE_CONNECTION_RESPONSE

This is the response by the Reader to a CLOSE_CONNECTON command from the Client. Upon receiving a CLOSE_CONNECTION command, the Reader SHALL attempt to send this response to the Client. After attempting to send this response, the Reader SHALL close its connection with the Client.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

CLOSE_CONNECTION_RESPONSE

Status: <LLRPStatus Parameter>

13.2 Parameters

13.2.1 LLRPConfigurationStateValue Parameter

This parameter, LLRPConfigurationStateValue, is a 32-bit value which represents a Reader's entire LLRP configuration state including: LLRP configuration parameters, vendor extension configuration parameters, ROSpecs, and AccessSpecs. A Reader SHALL change this value only:

- Upon successful execution of any of the following messages:
  - ADD_ROSPEC
  - DELETE_ROSPEC
  - ADD_ACCESSSPEC
  - DELETE_ACCESSSPEC

  - SET_READER_CONFIG
  - Any CUSTOM_MESSAGE command that alters the reader’s internal configuration.
- Upon an automatically deleted AccessSpec due to completion of OperationCountValue number of operations (section 12.2.1.1).

A Reader SHALL not change this value when the CurrentState of a ROSpec or AccessSpec changes.

The mechanism used to compute the LLRP configuration state value is implementation dependent. However, a good implementation will insure that there’s a high probability that the value will change when the Reader’s configuration state changes.

It is expected that a Client will configure the Reader and then request the Reader’s configuration state value. The Client will then save this state value. If this value does not change between two requests for it, then a Client may assume that the above components of the LLRP configuration have also not changed.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter. When requested by a Client, the Reader SHALL compute a state value based upon the Reader’s current configuration state. Upon each request, the Reader SHALL return the same state value provided a Client has not altered the Reader’s configuration state between requests. Aside from this requirement, the computation of the state value is implementation dependent.

LLRPConfigurationStateValue Parameter

LLRPConfigurationStateValue: Unsigned Integer

13.2.2 Identification Parameter

This parameter carries an identification parameter that is unique within the local administration domain. The identifier could be the Reader MAC address or EPC. The IDType defines the type of the identification value contained in this Parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

Identification Parameter

IDType: Integer

Possible Values:

```
IDType        ID
------        --
  0           MAC address
  1           EPC
```

Reader ID: Byte array. If IDType=0, the MAC address SHALL be encoded as EUI-64.[EUI64]

13.2.3 GPOWriteData Parameter

This parameter carries the data pertinent to perform the write to a general purpose output port.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support GPOs SHALL set NumGPOs in the GPIOCapabilities to zero. If such a Reader receives a SET_READER_CONFIG with

GPOWriteData Parameter, the Reader SHALL return an error message and not process any of the parameters in that message.

GPOWriteData Parameter

GPO Port Number : Unsigned Short Integer. 0 is invalid.

GPO Data: Boolean. The state to output on the specified GPO port.

13.2.4 KeepaliveSpec Parameter

This parameter carries the specification for the keepalive message generation by the Reader. This includes the definition of the periodic trigger to send the keepalive message.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

KeepaliveSpec Parameter

KeepaliveTriggerType: Integer

Possible Values:

```
Value   Definition
-----   ----------
  0    Null – No keepalives SHALL be sent by the Reader
  1    Periodic
```

PeriodicTriggerValue: Integer. Time interval in milliseconds. This field is ignored when KeepaliveTriggerType is not 1.

13.2.5 AntennaProperties Parameter

This parameter carries a single antenna's properties. The properties include the gain and the connectivity status of the antenna.The antenna gain is the composite gain and includes the loss of the associated cable from the Reader to the antenna. The gain is represented in dBi*100 to allow fractional dBi representation.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

AntennaProperties Parameter

AntennaID: Unsigned Short Integer

Possible Values:

1 to N, where N is the maximum number of antennas supported by the device.

AntennaGain: Signed short integer. The gain of the antenna in dBI*100 (dB relative to Isotropic) to allow for fractional dBi representation.

AntennaConnected: Boolean. False = not connected, True = connected.

13.2.6 AntennaConfiguration Parameter

This parameter carries a single antenna's configuration and it specifies the default values for the parameter set that are passed in this parameter block. The scope of the default values is the antenna. The default values are used for parameters during an operation on this antenna if the parameter was unspecified in the spec that describes the operation.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

AntennaConfiguration Parameter

Antenna ID: Unsigned Short Integer. If set to zero, this configuration applies to all the antennas.

RFReceiverSettings: <RFReceiver Parameter> [Optional]

RFTransmitterSettings: <RFTransmitter Parameter> [Optional]

AirProtocolInventoryCommandSettings: List of LLRP parameters. [Optional]

Possible Values:

Each air protocol's inventory command parameter is expressed as a different LLRP Parameter. The air protocol specific inventory command LLRP Parameters are defined in section *16.1*. This field is a list of inventory command LLRP Parameters, one per air protocol, that the Client would like to use as the default inventory command setting for inventory operations using the air protocol on this antenna.

Custom Extension Point List: List of <custom Parameter> [Optional]

13.2.6.1 RFReceiver Parameter

This Parameter carries the RF receiver information. The Receiver Sensitivity defines the sensitivity setting at the receiver. The value is the index into the ReceiveSensitivityTable (section *10.2.1.1*).

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

RFReceiver Parameter

ReceiverSensitivity: Unsigned Short Integer - an index into the ReceiveSensitivity Table (section *10.2.1.1*)

13.2.6.2 RFTransmitter Parameter

This Parameter carries the RF transmitter information. The Transmit Power defines the transmit power for the antenna expressed as an index into the TransmitPowerTable (section *10.2.4.2*). The HopTableID is the index of the frequency hop table to be used by the Reader is used when operating in frequency-hopping regulatory regions. This field is ignored in non-frequency-hopping regulatory regions. The ChannelIndex is the one-based channel index in the FixedFrequencyTable to use during transmission (section *10.2.4.2.3*) and is used when operating in non-frequency-hopping regulatory regions. This field is ignored in frequency-hopping regulatory regions.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

RFTransmitter Parameter

Transmit Power: Unsigned Short Integer - an index into the Transmit Power table.

HopTableID : Unsigned Short Integer

ChannelIndex : Unsigned Short Integer. This is the index of the frequency to use.

13.2.6.3 C1G2InventoryCommand Parameter

This parameter defines the C1G2 inventory-specific settings to be used during a particular C1G2 inventory operation. This comprises of C1G2Filter Parameter, C1G2RF Parameter and C1G2Singulation Parameter and C1G2Challenge. It is not necessary that the Filter, RF Control, Singulation Control or Challenge Parameters be specified in each and every inventory command. They are optional parameters. If not specified, the default values in the Reader are used during the inventory operation. If multiple C1G2Filter parameters are encapsulated by the Client in the C1G2InventoryCommand parameter, the ordering of the filter parameters determines the order of C1G2 air-protocol commands (e.g., Select command) generated by the Reader. C1G2Filter parameters included in the C1G2InventoryCommand parameter of a SET_READER_CONFIG message replace any existing filters configured on the Reader. Client implementations may use a “null filter” (see section 13.2.6.3.1) to delete existing filters on a Reader.

The TagInventoryStateAware flag is used to determine how to process all the C1G2Filter and C1G2Singulation parameters in this command. At a functional level, if the Client is managing the tag states during an inventory operation (i.e., the Client is specifying Class1 Gen2 tag Select command Target and Action values), then it will set that flag to true and pass the appropriate fields in the C1G2 Filter and C1G2 Singulation parameters. If a reader set CanDoTagInventoryStateAwareSingulation to False in LLRPCapabilities (section 10.2.2), then the Reader SHALL ignore the TagInventoryStateAware flag.

Each C1G2Challenge parameter in the optional list of C1G2Challenge commands translate to issuing of individual Challenge command over the air. Each of the C1G2Challenge parameter is processed and corresponding Challenge command issued over the air, before any of the C1G2 Filter Parameter is processed.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2InventoryCommand Parameter

TagInventoryStateAware: Boolean

C1G2 Filter : List of <C1G2Filter Parameter> [if absent, use default]

C1G2 RF: <C1G2RFControl Parameter> [if absent, use default]

C1G2 Singulation Control : <C1G2SingulationControl Parameter> [if absent, use default]

C1G2 Challenge : <List of C1G2Challenge Parameter> [if absent, do not issue Challenge]

Custom Extension Point List: List of <Custom Parameter> (optional)

13.2.6.3.1 C1G2Filter Parameter

This parameter carries information specific to C1G2 filter (in particular, the parameters for the select command) operation, and are optionally sent with each inventory command from the Client to the Reader. This sets up the target tag population that gets inventoried. For an inventory operation with multiple filters, multiple instances of filter parameters are sent. A filter parameter contains the following fields:

- Target tag mask: This contains the information for the tag memory data pattern used for the select operation.
- T: This value is set if the Client is interested in only a truncated portion of the tag to be backscattered by the tag. The portion that gets backscattered includes the portion of the tag ID following the mask. This bit has to be set only in the last filter-spec.
- TagInventoryStateAwareFilterAction: This is used if the TagInventoryStateAware flag is set to true in the InventoryParameterSpec.
- TagInventoryStateUnawareFilterAction: This is used if the TagInventoryStateAware flag is set to false in the InventoryParameterSpec.
- ExtendedOnTime: Specifies time in milliseconds to keep continuous wave transmission ON after issuing air protocol *Select* command. This parameter will be ignored if reader cannot support such a capability.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2Filter Parameter

Target Tag Mask: <C1G2TagInventoryMask Parameter>

T: Integer

Possible Values:

```
        Value   Truncate action
        -----   ---------------
          0       Unspecified: The Reader decides what truncate action
                  to take.
          1       Do not truncate
          2       Truncate
```

TagInventoryStateAwareAction: C1G2TagInventoryStateAwareFilterAction Parameter (optional)

TagInventoryStateUnawareAction: C1G2TagInventoryStateUnawareFilterAction Parameter (optional)

ExtendedOnTime: ExtendedOnTime Parameter (optional)

Parameter specifies time in milliseconds to maintain continuous wave on after issuing over-the-air *Select* command

C1G2TagInventoryMask Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2TagInventoryMask Parameter

MB: Integer. C1G2 Tag memory bank.

Possible Values:

1-3. The mask used for the C1G2 select command applies only to EPC, TID or User memory, and not to Reserved memory (MB 0).

Pointer: Unsigned Short Integer. The first (msb) bit location of the specified memory bank against which to compare the TagMask.

TagMask: Bit array. The pattern against which to compare. If this array is empty (0 length), the tag mask is considered a “null filter” and will match all tags.

C1G2TagInventoryStateAwareFilterAction Parameter

This parameter is used by the Client to manage the tag states during an inventory operation. In order to use this parameter during inventory, the TagInventoryStateAware flag is set to true in the InventoryParameterSpec. This parameter contains:

- Target: This value indicates which flag in the tag to modify – whether the SL flag or its inventoried flag for a particular session.
- Action describes the action for matching and non-matching tags. The actions are specific about the tag-inventory states - e.g., do nothing, assert or deassert SL, assign inventoried S0/S1/S2/S3 to A or B.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support tag inventory state aware singulation SHALL set CanDoTagInventoryStateAwareSingulation to false in LLRPCapabilities.

C1G2TagInventoryStateAwareFilterAction Parameter

Target: Integer

Possible Values:

```
        Value     Definition
        -----     ----------
          0       SL
          1       Inventoried state for session S0
          2       Inventoried state for session S1
          3       Inventoried state for session S2
          4       Inventoried state for session S3
```

Action : Integer

Possible Values:

```
        Value     Definition
        -----     ----------
          0       Matching tags: assert SL or inventoried state → A.
                  Non-matching tags: deassert SL or inventoried state →B.
          1       Matching tags: assert SL or inventoried state →A. Non-
                  matching tags: do nothing
          2       Matching tags: do nothing. Non-matching tags: deassert
                  SL or inventoried state →B
          3       Matching tags: negate SL or (A→B, B→A) Non-matching
                  tags: do nothing
          4       Matching tags: deassert SL or inventoried state → B
                  Non-matching tags: assert SL or inventoried state →A
          5       Matching tags: deassert SL or inventoried state → B
                  Non-matching tags: do nothing
          6       Matching tags: do nothing. Non-matching tags: assert SL
                  or inventoried state → A
          7       Matching tags: do nothing. Non-matching tags: negate SL
                  or (A→B, B→A)
```

C1G2TagInventoryStateUnawareFilterAction Parameter

This parameter is used by the Client if it does not want to manage the inventoried state of tags. Using this parameter, the Client instructs the Reader about the tags that should and should not participate in the inventory action. In order to use this parameter during inventory, the TagInventoryStateAware flag is set to false in the InventoryParameterSpec. This parameter contains:

- Action describes the action for matching and non-matching tags. However, the action is simply specifying whether matching or non-matching tags partake in this inventory. The Reader is expected to handle the tag inventory states to facilitate this.

In this parameter, Action=Select means include the corresponding tags in the Inventory, and Action=Unselect means exclude the corresponding tags in the Inventory.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter

C1G2RFControl Parameter

ModeIndex: Unsigned Integer. This is an index into the UHFC1G2RFModeTable.

Tari: Integer. Value of Tari to use for this mode specified in nsec. This is specified if the mode selected has a Tari range. If the selected mode has a range, and the Tari is set to zero, the Reader implementation picks up any Tari value within the range. If the selected mode has a range, and the specified Tari is out of that range and is not set to zero, an error message is generated.

Possible Values:

0 or 6250-25000 nsec

```
Value      Matching Tags       Non-matching Tags
-----      -------------       -----------------
  0           Select               Unselect
  1           Select               Do nothing
  2           Do nothing           Unselect
  3           Unselect             Do nothing
  4           Unselect             Select
  5           Do nothing           Select
```

13.2.6.3.2 C1G2RF Control Parameter

This Parameter carries the settings relevant to RF forward and reverse link control in the C1G2 air protocol. This is basically the C1G2 RF Mode and the Tari value to use for the inventory operation.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

13.2.6.3.3 C1G2SingulationControl Parameter

This C1G2SingulationControl Parameter provides controls particular to the singulation process in the C1G2 air protocol. The singulation process is started using a Query command in the C1G2 protocol. The Query command describes the session number, tag state, the start Q value to use, and the RF link parameters. The RF link parameters are specified using the C1G2RFControl Parameter (see section *13.2.6.3.2*). This Singulation Parameter specifies the session, tag state and description of the target singulation environment. The following attributes are specified to provide guidance to the Reader for the singulation algorithm:

- Tag transit time: This is the measure of expected tag mobility in the field of view of the antenna where this inventory operation is getting executed.
- Tag population: This is the expected tag population in the field of view of the antenna.

In addition, the Singulation Parameter allows setting of the following:

- Session ID: This is the C1G2 session number that the tags use to update the inventory state upon successful singulation.
- TagInventoryStateAwareSingulationAction: This is used if the TagInventoryStateAware flag is set to true in the InventoryParameterSpec.
  - I: This is the inventoried state of the target tag population in the selected session and it corresponds to the *Target* field of the Class1 Gen2 Query command. Only tags that match the session state participate in the inventory round. If the TagInventoryStateAware flag is false, then the Reader ignores this field, and its up to the Reader implementation to determine the value of I used in the inventory round.
  - S: This is the state of the SL flag in the tag and it corresponds to the *Sel* field of the Class1 Gen2 Query command. Only tags that match that tag state participate in the inventory round. If the TagInventoryStateAware flag is false, then the Reader ignores this field, and its up to the Reader implementation to determine the value of S used in the inventory round.

If a reader sets CanDoTagInventoryStateAwareSingulation to False in LLRPCapabilities (section*10.2.2*), it SHALL ignore the TagInventoryStateAwareSingulationAction field.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2SingulationControl Parameter

Session: Integer. Session number to use for the inventory operation.

Possible Values: 0-3

Tag population: Unsigned Short Integer. An estimate of the tag population in view of the RF field of the antenna.

Tag transit time: Unsigned Integer. An estimate of the time a tag will typically remain in the RF field of the antenna specified in milliseconds.

TagInventoryStateAwareSingulationAction: <C1G2TagInventoryStateAwareSingulationAction Parameter> (optional)

C1G2TagInventoryStateAwareSingulationAction Parameter

See C1G2SingulationControl above for a description of this parameter.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support tag inventory state aware singulation SHALL set CanDoTagInventoryStateAwareSingulation to false in LLRPCapabilities.

C1G2TagInventoryStateAwareSingulationAction Parameter

I : Integer

Possible Values:

```
Value   Definition
-----   ----------
  0     State A
  1     State B
```

S : Integer

Possible Values:

```
Value   Definition
-----   ----------
  0     SL
  1     ~SL
```

S_All : Integer

If set to zero, reference the S field. If set to one, the S field is ignored.

Possible Values:

```
Value   Definition
-----   ----------
  0     No
  1     All
```

13.2.6.3.4 C1G2Challenge Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2Challenge Parameter

L: Integer

Enable or disable inclusion of length in tag's stored reply

Possible Values:

0. Omit length from stored reply
1. Include length in stored reply

E: Integer

Enable or disable transmission of tag's response to *Challenge* command immediately after transmission of its EPC during inventory

Possible Values:

0. Do not concatenate response to EPC
1. Concatenate response to EPC

CSI: Integer

Specifies crypto suite to use for the Challenge

Message: Bit array

Crypto message with parameters for authentication

ExtendedOnTime: Integer

Time in milliseconds to maintain continuous wave on after issuing over-the-air *Challenge* command

13.2.6.4 GPIPortCurrentState Parameter

This Parameter carries the current configuration and state of a single GPI port. In a SET_READER_CONFIG message, this parameter is used to enable or disable the GPI port using the GPIConfig field; the GPIState field is ignored by the reader. In a GET_READER_CONFIG message, this parameter reports both the configuration and state of the GPI port.

When a ROSpec or AISpec is configured on a GPI-capable reader with GPI start and/or stop triggers, those GPIs must be enabled by the client with a SET_READER_CONFIG message for the triggers to function.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support GPIs SHALL set NumGPIs in the GPIOCapabilities to zero. If such a Reader receives a GET_READER_CONFIG with a GPIPortCurrentState Parameter, the Reader SHALL return an error message and not process any of the parameters in that message.

GPIPortCurrentState Parameter

GPIPortNum: Unsigned Short Integer. Zero is illegal.

GPIConfig : Boolean (0 for disabled, 1 for enabled)

GPIState : Integer (ignored in SET_READER_CONFIG messages)

Possible Values:

```
Value   Definition
-----   ----------
```

```
              0     GPI state is low
              1     GPI state is high
              2     GPI state is unknown
```

13.2.6.5 EventsAndReports Parameter

This parameter controls the behavior of the Reader when a new LLRP connection is established. In a SET_READER_CONFIG message, this parameter is used to enable or disable the holding of events and reports upon connection using the HoldEventsAndReportsUponReconnect field. In a GET_READER_CONFIG message, this parameter reports the current configuration. If the

HoldEventsAndReportsUponReconnect is true, the reader will not deliver any reports or events (except the ConnectionAttemptEvent) to the Client until the Client issues an ENABLE_EVENTS_AND_REPORTS message. Once the ENABLE_EVENTS_AND_REPORTS message is received the reader ceases its hold on events and reports for the duration of the connection.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

EventsAndReports Parameter

HoldEventsAndReportsUponReconnect: Boolean. (False does not hold reports and events, True holds reports and events)

14 Reports, Notifications and Keepalives

This section describes the messages and parameters used in reports, event notifications and keepalives that are generated by the Reader and sent to the Client.

The Reader SHALL send reports only when

- A reporting trigger (ROReportTrigger or AccessReportTrigger) generates a report while a connection is open, or
- In response to an explicit Client request (GET_REPORT or ENABLE_EVENTS_AND_REPORTS), or
- A notification event occurs and the event is enabled.

The triggers may be specified per ROSpec and AccessSpec using ROReportSpec and AccessReportSpec parameters. In a report, the Reader SHALL send new data (results of ROSpecs and/or AccessSpecs) acquired since the last report message. The tag report data generated by the AccessReport trigger SHALL NOT duplicate the tag report data generated by the ROReportTrigger, and vice-versa.

14.1 Messages

14.1.1 GET_REPORT

This message is issued by the Client to the Reader to get the tag reports. In response to this message, the Reader SHALL return tag reports accumulated. If no reports are available to send as a response to a GET_REPORT message, the Reader MAY return an empty RO_ACCESS_REPORT message.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

GET_REPORT

14.1.2 RO_ACCESS_REPORT

This message is issued by the Reader to the Client, and it contains the results of the RO and Access operations. The ROReportSpec and AccessReportSpec parameters define the contents and triggers for this message.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

RO_ACCESS_REPORT
InventoryAccessReportData: List of <TagReportData Parameter> [Optional]
RFSurveyReportData: List of <RFSurveyReportData Parameter> [Optional]
Custom Extension Point List: List of <custom Parameter> [Optional]

14.1.3 KEEPALIVE

This message is issued by the Reader to the Client. This message can be used by the Client to monitor the LLRP-layer connectivity with the Reader. The Client configures the trigger at the Reader to send the Keepalive message. The configuration is done using the KeepaliveSpec parameter (section *13.2.4*).

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

KEEPALIVE

14.1.4 KEEPALIVE_ACK

A Client SHALL generate a KEEPALIVE_ACK in response to each KEEPALIVE received by the Reader. If the Reader fails to receive multiple consecutive KEEPALIVE_ACK responses to its KEEPALIVE requests, the Reader MAY assume the client connection is defunct and can be closed.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

KEEPALIVE_ACK

14.1.5 READER_EVENT_NOTIFICATION

This message is issued by the Reader to the Client whenever an event that the Client subscribed to occurs. The pertinent event data is conveyed using the ReaderEventNotificationData parameter.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

READER_EVENT_NOTIFICATION
ReaderEventNotificationData: ReaderEventNotificationData Parameter

14.1.6 ENABLE_EVENTS_AND_REPORTS

This message can be issued by the Client to the Reader after a LLRP connection is established. The Client uses this message to inform the Reader that it can remove its hold on event and report messages. Readers that are configured to hold events and reports on reconnection (See section *13.2.6.5*) respond to this message by returning the tag reports accumulated (same way they respond to GET_REPORT (See section *14.1.1*).

Compliance requirement: Compliant Readers and Clients MAY implement this message.

ENABLE_EVENTS_AND_REPORTS

14.2 Parameters

14.2.1 ROReportSpec Parameter

This Parameter carries the Reader inventory and RF survey reporting definition for the antenna. This parameter describes the contents of the report sent by the Reader and defines the events that cause the report to be sent.

The ROReportTrigger field defines the events that cause the report to be sent.

The TagReportContentSelector parameter defines the desired contents of the report. The ROReportTrigger defines the event that causes the report to be sent by the Reader to the Client.

See section *14.2.6* for details about the order that reports are to be sent with respect to Reader event notifications.

Custom extensions to this parameter are intended to specify summary data to be reported as an extension to the RO_ACCESS_REPORT message (see section *14.1.2*).

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ROReportSpec Parameter

ROReportTrigger: Integer

Possible Values:

```
Value   Definition
-----   ----------
   0    None
   1    (Upon N TagReportData Parameters or End of AISpec) Or
        (End of RFSurveySpec) - N=0 is unlimited
   2    Upon N TagReportData Parameters or End of ROSpec - N=0
        is unlimited
   3    Upon N seconds or (End of AISpec or End of RFSurveySpec)
        – N=0 is unlimited
   4    Upon N seconds or End of ROSpec – N=0 is unlimited.
   5    Upon N milliseconds or (End of AISpec or End of
        RFSurveySpec) – N=0 is unlimited
   6    Upon N milliseconds or End of ROSpec – N=0 is unlimited
   7    Upon N inventory rounds or End of ROSpec - N=0 is
        unlimited. (If N=1, the TagSeenCount parameter in
        TagReportData can be used to
```

N: Unsigned Short Integer. When ROReportTrigger = 1 or 2, this is the number of TagReportData parameters present in a report before the report trigger fires. When ROReportTrigger = 3 or 4, this is the number of seconds since the last report was generated before the report trigger fires. When ROReportTrigger = 5 or 6, this is the number of milliseconds since the last report was generated before the report trigger fires. If N = 0, there is no limit on either the number of TagReportData parameters, or the time since the last report was generated. This field SHALL be ignored when ROReportTrigger = 0.

ReportContents: <TagReportContentSelector Parameter>

Custom Extension Point List: List of <Custom Parameter> [Optional]

14.2.1.1 TagReportContentSelector Parameter

This parameter is used to configure the contents that are of interest in TagReportData. If enabled, the field is reported along with the tag data in the TagReportData.

Custom extensions to this parameter are intended to specify data related to each tag that is to be reported as an extension to the TagReportData parameter (see section 14.2.3).

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

TagReportContentSelector

EnableROSpecID: Boolean **EnableSpecIndex:**

Boolean **EnableInventoryParameterSpecID:**

Boolean **EnableAntennaID:** Boolean

EnableChannelIndex: Boolean

EnablePeakRSSI: Boolean

EnableFirstSeenTimestamp: Boolean

EnableLastSeenTimestamp: Boolean

EnableTagSeenCount: Boolean

EnableCryptoResponse: Boolean

14.2.1.2 C1G2EPCMemorySelector Parameter

AirProtocolSpecificEPCMemorySelector: LLRP parameter.

Possible Values:

Each air protocol's EPC memory selector parameter is expressed as a different LLRP Parameter. The air protocol specific EPC memory selector LLRP Parameters are defined in section 16.1 This field is the EPC memory selector LLRP Parameter corresponding to the air protocol referenced by the ProtocolID in the ROSpec that the ROReportSpec is part of.

EnableAccessSpecID: Boolean

Custom Extension Point List: List of <Custom Parameter> [Optional]

This parameter is used to determine what contents are of interest in the C1G2EPC memory bank for reporting. If enableCRC, enablePC, and enableXPC are set to false, then only the EPC is returned in the RO Report. If enablePC is set to true, then the PC bits are returned with the EPC in the RO Report. If enableCRC is set to true, then the CRC bits are returned with the EPC in the RO Report. If enableXPC is set to true, then the XPC bits are returned with the EPC in the RO Report in the C1G2XPCW1 and C1G2XPCW2 parameters, depending on what was backscattered by the tag. If enableXPC is set to true, Client implementations SHALL accept C1G2XPCW1 and C1G2XPCW2 in the RO Report.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2EPCMemorySelector

enablePC: Boolean **enableCRC:**

Boolean **enableXPC**: Boolean

14.2.2 AccessReportSpec Parameter

This parameter sets up the triggers for the Reader to send the access results to the Client. In addition, the Client can enable or disable reporting of ROSpec details in the access results.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

AccessReportSpec

AccessReportTrigger: Integer

Possible Values:

```
Value   Definition
-----   ----------
   0    Whenever ROReport is generated for the RO that
        triggered the execution of this AccessSpec.
   1    End of AccessSpec (immediately upon completion of
        the access operation)
```

14.2.3 TagReportData Parameter

This report parameter is generated per tag per accumulation scope. The only mandatory portion of this parameter is the EPCData parameter. If there was an access operation performed on the tag, the results of the OpSpecs are mandatory in the report. The other sub-parameters in this report are optional. LLRP provides three ways to make the tag reporting efficient:

1. Allow parameters to be enabled or disabled via TagReportContentSelector (section 14.2.1.1) in TagReportSpec.
2. If an optional parameter is enabled, and is absent in the report, the Client SHALL assume that the value is identical to the last parameter of the same type received. For example, this allows the Readers to not send a parameter in the report whose value has not changed since the last time it was sent by the Reader.
3. Allow accumulation of tag reports. See next section for details of accumulation.

Compliance Requirement: Compliant Readers and Clients SHALL implement this parameter.

TagReportData Parameter

EPCData: <EPCData Parameter>

ROSpecID: <ROSpecID Parameter> [Optional]

SpecIndex: <SpecIndex Parameter> [Optional]

InventoryParameterSpecID: <InventoryParameterSpecID Parameter> [Optional]

AntennaID: <AntennaID Parameter> [Optional]

PeakRSSI: <PeakRSSI Parameter> [Optional]

ChannelIndex: <ChannelIndex Parameter> [Optional]

FirstSeenTimestampUTC: <UTCFirstSeenTimestamp Parameter> [Optional]

FirstSeenTimeStampUptime: <UptimeFirstSeenTimestamp Parameter> [Optional]

LastSeenTimestampUTC: <UTCLastSeenTimestamp Parameter> [Optional]

LastSeenTimeStampUptime: <UptimeLastSeenTimestamp Parameter> [Optional]

TagSeenCount: <TagSeenCount Parameter> [Optional]

AirProtocolTagData: LLRP Parameters (e.g., C1G2EPC-PC, C1G2EPC-CRC) [Optional]

Possible Values:

Each air protocol's AirProtocolTagData parameter is expressed as a different LLRP Parameter. The air protocol specific AirProtocolTagData LLRP Parameters are defined in section *16.1*. This field is the AirProtocolTagData LLRP Parameter corresponding to the air protocol referenced by the ProtocolID of the InventoryParameterSpec during whose execution this tag was observed.

AccessSpecID: <AccessSpecID Parameter> [Optional]

OpSpecResultList: List of LLRP parameters [Optional]

Possible Values of each LLRP Parameter:

Air protocol specific OpSpecResult parameter, <ClientRequestOpSpecResult Parameter>, or Custom Parameter.

Regarding the air protocol specific OpSpecResult parameter: Each air protocol's OpSpecResult parameter is expressed as a different LLRP Parameter. The air protocol specific OpSpecResult LLRP Parameters are defined in section *16.1* This field is a list of OpSpecResult LLRP Parameters corresponding to the air protocol referenced by the ProtocolID of the AccessSpec.

Custom Extension Point List: List of <Custom Parameter> [Optional]

14.2.3.1 Accumulation of TagReportData

A Reader MAY accumulate multiple tag reports into a single tag report.. If a Reader accumulates, the Reader SHALL follow the accumulation rules specified in this section.

The following specifies the rules for accumulating multiple tag observations into a single TagReportData:

- EPCData:
  - The Reader SHALL not accumulate tag reports that do not have the same EPCData value.
- OpSpecResultList:
  - The Reader SHALL not accumulate tag reports that do not have the same value for the OpSpec results in the OpSpecResultList.
- SpecID, SpecIndex, InventoryParameterSpecID, AntennaID, AirProtocolTagData, AccessSpecID:
  - These fields are optional, and their reporting can be enabled by the Client. If the Client has enabled one or more fields listed above, the Reader SHALL not accumulate tag reports that do not have the same value for all the enabled fields.
- FirstSeenTimestamp, LastSeenTimestamp, PeakRSSI, TagSeenCount, ChannelIndex

These fields are optional, and their reporting can be enabled by the Client. If the field is enabled, the Reader sets the value of these fields as follows:

- FirstSeenTimestamp: The Reader SHALL set it to the time of the first observation amongst the tag reports that get accumulated in the TagReportData.
- LastSeenTimestamp: The Reader SHALL set it to the time of the last observation amongst the tag reports that get accumulated in the TagReportData.
- PeakRSSI: The Reader SHALL set it to the maximum RSSI value observed amongst the tag reports that get accumulated in the TagReportData.
- ChannelIndex: The Reader MAY set it to the index of the first channel the tag was seen.
- TagSeenCount: The Reader SHALL set it to the number of tag reports that get accumulated in the TagReportData.

14.2.3.2 EPCData Parameter

This parameter carries the EPC identifier information.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

EPCData Parameter
EPC: Bit array

14.2.3.3 ROSpecID Parameter

This parameter carries the ROSpecID information.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ROSpecID Parameter
ROSpecID: Unsigned Integer

14.2.3.4 SpecIndex Parameter

This parameter carries the SpecIndex information. The SpecIndex indicates the item within the ROSpec that was being executed at the time the tag was observed.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

SpecIndex Parameter
SpecIndex: Unsigned Short Integer

14.2.3.5 InventoryParameterSpecID Parameter

This parameter carries the InventoryParameterSpecID information.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

InventoryParameterSpecID Parameter
InventoryParameterSpecID: Unsigned Short Integer

14.2.3.6 AntennaID Parameter

This parameter carries the AntennaID information.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

AntennaID Parameter

AntennaID: Unsigned Short Integer

14.2.3.7 PeakRSSI Parameter

This parameter carries the PeakRSSI information.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

PeakRSSI Parameter

PeakRSSI: Signed Integer. The peak received power of the EPC backscatter in dBm.

Possible Values:

-128 to +127

14.2.3.8 ChannelIndex Parameter

This parameter carries the one-based ChannelIndex value.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ChannelIndex Parameter

ChannelIndex: Unsigned Integer

Possible Values: 1 to 255.

14.2.3.9 FirstSeenTimestampUTC Parameter

This parameter carries the FirstSeenTimestamp information in UTC.

Compliance requirement: Compliant Readers and Clients that have UTC clocks SHALL implement this parameter.

FirstSeenTimestampUTC Parameter

Microseconds: Unsigned long Integer. This is the time elapsed since the Epoch (00:00:00 UTC, January 1, 1970) measured in microseconds.

14.2.3.10 FirstSeenTimestampUptime Parameter

This parameter carries the FirstSeenTimestamp information in Uptime.

Compliance requirement: Compliant Readers and Clients that do not have UTC clocks SHALL implement this parameter. Compliant Readers and Clients that have UTC clocks MAY implement this parameter.

FirstSeenTimestampUptime Parameter

Microseconds: Unsigned long Integer. This is the time elapsed since boot, measured in microseconds.

14.2.3.11 LastSeenTimestampUTC Parameter

This parameter carries the LastSeenTimestamp information in UTC.

Compliance requirement: Compliant Readers and Clients that have UTC clocks SHALL implement this parameter.

LastSeenTimestampUTC Parameter

Microseconds: Unsigned long Integer. This is the time elapsed since the Epoch (00:00:00 UTC, January 1, 1970) measured in microseconds.

14.2.3.12 LastSeenTimestampUptime Parameter

This parameter carries the LastSeenTimestamp information in Uptime.

Compliance requirement: Compliant Readers and Clients that do not have UTC clocks SHALL implement this parameter. Compliant Readers and Clients that have UTC clocks MAY implement this parameter.

LastSeenTimestampUptime Parameter

Microseconds: Unsigned long Integer. This is the time elapsed since boot, measured in microseconds.

14.2.3.13 TagSeenCount Parameter

This parameter carries the tag seen count information. If TagSeenCount > 65535 for the report period, the reader SHALL report 65535.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

TagSeenCount Parameter

Count: Unsigned Short Integer

14.2.3.14 C1G2PC Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2PC Parameter

PC bits: Unsigned Short Integer

14.2.3.15 C1G2XPCW1 Parameter

This parameter is included within a TagReportData parameter if enableXPC is set to true and the tag backscattered XPC_W1.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter. Readers that do not support C1G2XPC SHALL set CanSupportXPC to false in C1G2LLRPCapabilities.

C1G2XPCW1 Parameter

XPC_W1: Unsigned Short Integer

XPC_W1 value is interpreted by client as a 16-bit bit field:

MSB															LSB
XEB	RFU	RFU	RFU	SA	SS	FS	SN	B	C	SLI	TN	U	K	NR	H

Starting from MSB the meaning of XPC_W1 bits is defined in table below:

Bit indicator	Descriptor	Settings
XEB	XPC_W2 indicator	0: Tag has no XPC_W2 or all bits of XPC_W2 are zero-valued 1: Tag has an XPC_W2 and at least one bit of XPC_W2 is nonzero
RFU	RFU	RFU and fixed at zero
RFU	RFU	RFU and fixed at zero
RFU	RFU	RFU and fixed at zero
SA	Sensor Alarm indicator (defined by ISO/IEC 18000-63)	0: Tag is not reporting an alarm condition or does not support the SA flag 1: Tag is reporting an alarm condition
SS	Simple Sensor indicator (defined by ISO/IEC 18000-63)	0: Tag does not have a Simple Sensor 1: Tag has a Simple Sensor
FS	Full Function Sensor indicator (defined by ISO/IEC 18000-63)	0: Tag does not have a Full Function Sensor 1: Tag has a Full Function Sensor
SN	Snapshot Sensor indicator (defined by ISO/IEC 18000-63)	0: Tag does not have a Snapshot Sensor 1: Tag has a Snapshot Sensor
B	Battery-Assisted Passive indicator	0: Tag is passive or does not support the B flag 1: Tag is battery-assisted
C	Computed response indicator	0: ResponseBuffer is empty or Tag does not support a ResponseBuffer 1: ResponseBuffer contains a response
SLI	SL indicator	0: Tag has a deasserted SL flag or does not support the SLI bit 1: Tag has an asserted SL flag
TN	Notification indicator	0: Tag does not assert a notification or does not support the TN bit 1: Tag asserts a notification
U	Untraceable indicator	0: Tag is traceable or does not support the U bit 1: Tag is untraceable
K	Killable indicator	0: Tag is not killable by *Kill* command or does not support the K bit 1: Tag can be killed by *Kill* command.
NR	Nonremovable indicator	0: Tag is removable from its host item or does not support the NR bit 1: Tag is not removable from its host item

Bit indicator	Descriptor	Settings
H	Hazmat indicator	0: Tagged item is not hazardous material or Tag does not support the H bit 1: Tagged item is hazardous material

14.2.3.16 C1G2XPCW2 Parameter

This parameter is included within a TagReportData parameter if enableXPC is set to true and the tag backscattered XPC_W2.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter. Readers that do not support C1G2XPC SHALL set CanSupportXPC to false in C1G2LLRPCapabilities.

C1G2XPCW2 Parameter

XPC_W2: Unsigned Short Integer

14.2.3.17 1G2CRC Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2CRC Parameter

CRC: Unsigned Short Integer

14.2.4 ClientRequestOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

ClientRequestOpSpecResult Parameter

OpSpecID: Unsigned Short Integer. 0 is illegal.

14.2.5 AccessSpecID Parameter

This parameter carries the AccessSpecID information.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

AccessSpecID Parameter

AccessSpecID: Unsigned Integer

14.2.6 CryptoResponse Parameter

This parameter carries cryptographic response message to a Challenge command, when such a response is concatenated to EPC and back scattered by the tag during inventory. See *C1G2Challenge Parameter* in section *13.2.6.3.4* for details on enabling such immediate response from the tag.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

CryptoResponse Parameter

Message: Bit array

14.2.7 RFSurveyReportData Parameter

This describes the content of the RF Survey Report.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

RFSurveyReportData Parameter

ROSpecID: <ROSpecID Parameter> [Optional]

SpecIndex: <SpecIndex Parameter> [Optional]

FrequencyPowerLevelList: List of <FrequencyRSSILevelEntry Parameter>
Custom Extension Point List: List of <custom Parameter> [Optional]

14.2.7.1 FrequencyRSSILevelEntry Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

FrequencyRSSILevelEntry Parameter

Timestamp: Either <UTCTimestamp Parameter> or <Uptime Parameter>

Frequency: Unsigned Integer. The frequency on which the measurement was taken, specified in kHz.

Bandwidth: Unsigned Integer. The measurement bandwidth of the measurement in kHz.

Average RSSI: Integer in dBm. The average power level observed at this frequency.

Possible Values:

-128 to + 127

Peak RSSI: Integer in dBm. The peak power level observed at this frequency.

Possible Values:

-128 to + 127

14.2.8 ReaderEventNotificationSpec Parameter

This parameter is used by the Client to enable or disable notification of one or more Reader events. Notification of buffer overflow events and connection events (attempt/close) are mandatory, and not configurable.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ReaderEventNotificationSpec Parameter

EventNotificationSpecTable: List of <EventNotificationState Parameter>

14.2.8.1 EventNotificationState Parameter

This parameter is used to enable or disable notification of a single Reader event type.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

EventNotificationState Parameter

EventType:

Possible Values:

```
        Value   Definition
        -----   ----------
          0      Upon hopping to next channel (e.g., in FCC
                 regulatory region)
          1      GPI event
          2      ROSpec event (start/end/preempt)
          3      Report buffer fill warning
          4      Reader exception event
          5      RFSurvey event (start/end)
          6      AISpec event (end)
          7      AISpec event (end) with singulation details
          8      Antenna event (disconnect/connect)
          9      SpecLoop event
```

NotificationState: Boolean; enable = true, disable = false.

14.2.9 ReaderEventNotificationData Parameter

This parameter describes the contents of the event notification sent by the Reader, and defines the events that cause the notification to be sent. Event notification messages may be sent by the Reader due to connection establishment/closing event, critical events such as hopping, fault-detection in a Reader functional block, buffer overflow, due to the activation of a Reader accessory trigger input (e.g. motion detection), or due to performance monitoring events such as abnormalities in the RF environment.

Timestamp is the time that the events reported occurred.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

EventNotificationState Parameter
ReaderEventNotificationData Parameter

Timestamp: Either <UTCTimestamp Parameter> or <Uptime Parameter>

Events: List of events.

Possible Values: The possible members of the list are

{

<HoppingEventParameter>,

<GPIEvent Parameter>,

<ROSpecEvent Parameter>,

<ReportBufferLevelWarningEvent Parameter>,

<ReportBufferOverflowErrorEvent Parameter>,

<ReaderExceptionEvent Parameter>,

```
  <RFSurveyEvent Parameter>,
  <AISpecEvent Parameter>,
  <AntennaEvent Parameter>,
  <ConnectionAttemptEvent Parameter>,
  <ConnectionCloseEvent Parameter>,
  <SpecLoopEvent Parameter>
}
```

Custom Extension Point List: List of <custom Parameter> [optional]

14.2.9.1 Requirements for Ordering of Event Reporting

LLRP assumes a reliable stream transport mechanism. Messages sent through LLRP will arrive in the order that they were sent over the transport and binding utilised. Status events within the same message SHALL be ordered chronologically.

Status events delivered by reader event notifications are useful, especially in conjunction with the tag report data. The following describes the requirements of the reader event notifications ordering with respect to the ordering of tag reports and Reader Event Notifications.

The following requirements are made on the ordering of Event Parameters with respect to each other and to tag report Parameters. These statements apply if the respective status events and report triggers are enabled.

If the start of an ROSpec is triggered by a GPI, the GPIEvent Parameter SHALL be sent before the ROSpecEvent Parameter signaling the start of the ROSpec.

If the end of an ROSpec is triggered by a GPI, the GPIEvent Parameter SHALL be sent before the ROSpecEvent Parameter signaling the end of the ROSpec.

If an ROSpec contains one or more AISpecs, the ROSpecEvent parameter signaling the end of an ROSpec SHALL be sent after the AISpecEvent Parameter signaling the end of the last AISpec within that ROSpec.

If one ROSpec pre-empts another ROSpec, the ROSpecEvent parameter signaling the preemption of the first ROSpec SHALL be sent before the ROSpecEvent parameter signaling the start of the next ROSpec.

Tag data received during an ROSpec execution SHALL be sent between the ROSpecEvent parameter signaling the start of the ROSpec and the ROSpecEvent parameter signaling the end or preemption of the ROSpec if the ROReportTrigger is not set to 'None'.

Tag data received during an AISpec execution SHALL be sent before the AISpecEvent Parameter signaling the end of the AISpec if the ROReportTrigger is not 'None' or 'end of RO Spec'

Tag data received during the time on a channel SHALL be sent after the HoppingEvent parameter that announced this channel and before the next HoppingEvent parameter when the ROReportTrigger is not 'None' and N=1.

14.2.9.2 HoppingEvent Parameter

A Reader reports this event every time it hops frequency.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

HoppingEvent Parameter

HopTableID: Unsigned Short Integer

NextChannelIndex: Unsigned Short Integer. This is the one-based ChannelIndex of the next channel to which the Reader is going to change.

14.2.9.3 GPIEvent Parameter

A reader reports this event every time an enabled GPI changes GPIstate.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

GPIEvent Parameter **GPIPortNumber**:

Unsigned Short Integer **GPIEvent:** Boolean –

True/False.

14.2.9.4 ROSpecEvent Parameter

This parameter carries the ROSpec event details. The EventType could be start or end of the ROSpec.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ROSpecEvent Parameter

ROSpecID: Unsigned Integer. This is the ID of the ROSpec that started, ended or got preempted.

EventType: Integer

Possible Values:

```
Value   Definition
-----   ----------
  0      Start of ROSpec
  1      End of ROSpec
  2      Preemption of ROSpec
```

PreemptingROSpecID: Integer. This field is ignored when EventType != 2. This field carries the ID of the preempting ROSpec.

14.2.9.5 ReportBufferLevelWarningEvent Parameter

A Reader can warn the Client that the Reader's report buffer is filling up. A Client can act upon this warning by requesting report data from the Reader, thereby freeing the Reader's report memory resources.

ReportBufferLevelWarningEvent Parameter

ReportBufferPercentageFull: Integer

Possible Values: 0-100

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. A Reader MAY send a report buffer level warning event whenever the Reader senses that its report memory resources are running short. The buffer level at which a warning is reported is Reader implementation dependent. A Client MAY act upon a report buffer level warning event by requesting report data from the Reader and thereby free report memory resources in the Reader.

14.2.9.6 ReportBufferOverflowErrorEvent Parameter

A Reader reports a buffer overflow event whenever report data is lost due to lack of memory resources.

ReportBufferOverflowErrorEvent Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter. A Reader SHALL report a buffer overflow event whenever report data is lost due to lack of memory resources.

14.2.9.7 ReaderExceptionEvent Parameter

The reader exception status event notifies the client that an unexpected event has occurred on the reader. Optional parameters provide more detail to the client as to the nature and scope of the event.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

ReaderExceptionEvent Parameter

ROSpecID: <ROSpecID Parameter> [Optional]

SpecIndex: <Spec Index Parameter> [Optional]

InventoryParameterSpecID: <InventoryParameterSpecID Parameter> [Optional]

AntennaID: <AntennaID Parameter> [Optional]

AccessSpecID: <AccessSpecID Parameter> [Optional]

OpSpecID: <OpSpecID Parameter> [Optional] **Message**:

UTF-8 String

Custom Extension Point List: List of <custom Parameter> [Optional]

14.2.9.7.1 OpSpecID Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

OpSpecID Parameter

OpSpecId: Unsigned Short Integer. 0 is illegal.

14.2.9.8 RFSurveyEvent Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

RFSurveyEvent Parameter

ROSpecID: Unsigned Integer. The identifier of the ROSpec that contains the RFSurveySpec.

SpecIndex: Unsigned Short Integer. The index of the spec in the ROSpec.

EventType: Integer

Possible Values:

```
Value   Definition
-----   ----------
  0      Start of RFSurvey
  1      End of RFSurvey
```

14.2.9.9 AISpecEvent Parameter

This parameter carries the AISpec event details. The EventType is the end of the AISpec. When reporting the end event, the AirProtocolSingulationDetails MAY be reported if it is supported by the Reader and EventType of 7 has been enabled (section *14.2.8.1*).

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

AISpecEvent Parameter

ROSpecID: Unsigned Integer. The identifier of the ROSpec that contains the AISpec.

SpecIndex: Unsigned Short Integer. The index of the spec in the ROSpec.

EventType: Integer

Possible Values:

```
Value   Definition
-----   ----------
  0      End of AISpec
```

AirProtocolSingulationDetails: LLRP parameter [Optional]

Possible Values:

Each air protocol's AirProtocolSingulationDetails parameter is expressed as a different LLRP Parameter. The air protocol specific AirProtocolSingulationDetails LLRP Parameters are defined in section *14.2.9.9.1*. This field is the AirProtocolSingulationDetails LLRP Parameter corresponding to the air protocol referenced by the ProtocolID of the InventoryParameterSpec upon whose execution completion this event report was generated.

14.2.9.9.1 C1G2SingulationDetails Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2SingulationDetails Parameter

NumCollisionSlots: Unsigned Short Integer. The number of slots detected as collided over the duration of this report.

NumEmptySlots: Unsigned Short Integer. The number of slots detected as empty over the duration of this report.

14.2.9.10 C1G2 OpSpec Results

14.2.9.10.1 C1G2ReadOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2ReadOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

ReadData: Short Array. The data read from the RFID tag.

Result: Integer

Possible Values:

```
Value   Definition
-----   ----------
 0    Success
 1    Non-specific tag error
 2    No response from tag
 3    Non-specific reader error
 4    Memory overrun error
 5    Memory locked error
 6    Incorrect password error
 7    Not supported
```

14.2.9.10.2 C1G2WriteOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2WriteOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

NumWordsWritten: Unsigned Short Integer. The number of words written as a result of this OpSpec. If the number of words written is not equal to the length of the data pattern to write, the Result below SHALL be non-zero.

Result: Integer

Possible Values:

```
Value   Definition
-----   ----------
 0    Success
 1    Tag memory overrun error
 2    Tag memory locked error
 3    Insufficient power to perform memory-write
      operation
 4    Non-specific tag error
 5    No response from tag
 6    Non-specific reader error
 7    Incorrect password error
 8    Not supported
```

14.2.9.10.3 C1G2KillOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2KillOpSpecResult Parameter

OpSpecID: Unsigned Short Integer **Result**:

Integer

Possible Values:

```
          Value   Definition
          -----   ----------
             0    Success
             1    Zero kill password error
             2    Insufficient power to perform kill
                  operation
             3    Non-specific tag error
             4    No response from tag
             5    Non-specific reader error
             6    Incorrect password error
```

14.2.9.10.4 C1G2LockOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

C1G2LockOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
            Value   Definition
            -----   ----------
             0    Success
             1    Insufficient power to perform lock
                  operation
             2    Non-specific tag error
             3    No response from tag
             4    Non-specific reader error
             5    Incorrect password error
             6    Tag memory overrun error
             7    Tag memory locked error
             8    Not supported
```

14.2.9.10.5 C1G2BlockEraseOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2 Block Erase SHALL set CanSupportBlockErase to false in C1G2LLRPCapabilities

C1G2BlockEraseOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
Value   Definition
-----   ----------
   0    Success
   1    Tag memory overrun error
   2    Tag memory locked error
   3    Insufficient power to perform block erase
        operation
   4    Non-specific tag error
   5    No response from tag
   6    Non-specific reader error
   7    Incorrect password error
   8    Not supported
```

14.2.9.10.6 C1G2BlockWriteOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2 Block Write SHALL set CanSupportBlockWrite to false in C1G2LLRPCapabilities

C1G2BlockWriteOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

NumWordsWritten: Unsigned Short Integer

Result: Integer

Possible Values:

```
Value   Definition
-----   ----------
   0    Success
   1    Tag memory overrun error
   2    Tag memory locked error
   3    Insufficient power to perform memory-write
        operation
   4    Non-specific tag error
   5    No response from tag
   6    Non-specific reader error
   7    Incorrect password error
   8    Not supported
```

14.2.9.10.7 C1G2BlockPermalockOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2 Block Permalock SHALL set CanSupportBlockPermalock to false in C1G2LLRPCapabilities.

C1G2BlockPermalockOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
Value   Definition
-----   ----------
   0    Success
   1    Insufficient power to perform block
        permalock operation
   2    Non-specific tag error
   3    No response from tag
   4    Non-specific reader error
   5    Incorrect password error
   6    Tag memory overrun error
   7    Not supported
```

14.2.9.10.8 C1G2GetBlockPermalockStatusOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2 Block Permalock SHALL set CanSupportBlockPermalock to false in C1G2LLRPCapabilities.

C1G2GetBlockPermalockStatusOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

PermalockStatus: Unsigned Short Integer Array. Specifies the Permalock status of each block requested.

Result: Integer
Possible Values:

```
Value   Definition
-----   ----------
   0    Success
   1    Non-specific tag error
   2    No response from tag
   3    Non-specific reader error
   4    Incorrect password error
   5    Tag memory overrun error
   6    Not supported
```

14.2.9.10.9 C1G2UntraceableStatusOpSpecResult Parameter

Compliance requirement: Compliant Readers and Clients MAY implement this parameter. Readers that do not support C1G2 Untraceable SHALL set CanSupportUntraceable to false in C1G2LLRPCapabilities. If such a Reader receives an ADD_ACCESSSPEC with an AccessSpec that contained this OpSpec parameter, the Reader SHALL return an error for that message and not add the AccessSpec.

C1G2UntraceableOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
Value   Definition
-----   ----------
 0    Success
 1    Insufficient power to perform
      operation
 2    Non-specific tag error
 3    No response from tag
 4    Non-specific reader error
 5    Not Supported
 6    Insufficient privilege
 7    Tag memory overrun error
```

14.2.9.10.10 C1G2AuthenticateOpSpecResult Parameter

This parameter carries the crypto response message for an Authenticate command.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2AuthenticateOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
Value Definition
----- ----------
 0    Success
 1    Insufficient power
 2    Non-specific tag error
 3    No response from tag
 4    Non-specific reader error
 5    In process, still working
 6    In process success, stored response without length
 7    In process success, stored response with length
 8    In process success, send response without length
 9    In process success, send response with length
 10   In process error, stored response without length
 11   In process error, stored response with length
 12   In process error, send response without length
 13   In process error, send response with length
 14   Not supported
 15   Insufficient privilege
 16   Crypto suite error
 17   Response buffer overflow error
 18   Security timeout
 19   Other error
```

Response: Bit array

14.2.9.10.11 C1G2AuthCommOpSpecResult Parameter

This parameter carries the crypto response message for an AuthComm command.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2AuthCommOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
Value Definition
----- ----------
 0    Success
 1    Insufficient power
 2    Non-specific tag error
 3    No response from tag
 4    Non-specific reader error
 5    In process, still working
 6    In process success, stored response without length
 7    In process success, stored response with length
 8    In process success, send response without length
 9    In process success, send response with length
 10   In process error, stored response without length
 11   In process error, stored response with length
 12   In process error, send response without length
 13   In process error, send response with length
 14   Not supported
 15   Insufficient privilege
 16   Crypto suite error
 17   Security timeout
```

Response: Bit array

14.2.9.10.12 C1G2SecureCommOpSpecResult Parameter

This parameter carries the crypto response message for a SecureComm command.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2SecureCommOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
Value Definition
----- ----------
 0    Success
 1    Insufficient power
 2    Non-specific tag error
 3    No response from tag
 4    Non-specific reader error
 5    In process, still working
 6    In process success, stored response without length
 7    In process success, stored response with length
 8    In process success, send response without length
 9    In process success, send response with length
 10   In process error, stored response without length
```

```
                 11   In process error, stored response with length
                 12   In process error, send response without length
                 13   In process error, send response with length
                 14   Not supported
                 15   Insufficient privilege
                 16   Crypto suite error
                 17   Response buffer overflow error
                 18   Security timeout
```

Response: Bit array

14.2.9.10.13 C1G2ReadBufferOpSpecResult Parameter

This parameter carries the crypto response message for a ReadBuffer command.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2ReadBufferOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
                Value Definition
                ----- ----------
                 0    Success
                 1    Non-specific tag error
                 2    No response from tag
                 3    Non-specific reader error
                 4    Response buffer overflow error
```

Response: Bit array

14.2.9.10.14 C1G2KeyUpdateOpSpecResult Parameter

This parameter carries the response for KeyUpdate command.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2KeyUpdateOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
                Value Definition
                ----- ----------
                 0    Success
                 1    Insufficient power
                 2    Non-specific tag error
                 3    No response from tag
                 4    Non-specific reader error
                 5    In process, still working
                 6    In process success, stored response without length
```

```
          7    In process success, stored response with length
          8    In process success, send response without length
          9    In process success, send response with length
          10   In process error, stored response without length
          11   In process error, stored response with length
          12   In process error, send response without length
          13   In process error, send response with length
          14   Not supported
          15   Insufficient privilege
          16   Crypto suite error
          17   Response buffer overflow error
          18   Security timeout
          19   Command not encapsulated
```

Response: Bit array

14.2.9.10.15 C1G2TagPrivilegeOpSpecResult Parameter

This parameter carries the response for TagPrivilege command.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

C1G2TagPrivilegeOpSpecResult Parameter

OpSpecID: Unsigned Short Integer

Result: Integer

Possible Values:

```
         Value Definition
         ----- ----------
          0    Success
          1    Insufficient power
          2    Non-specific tag error
          3    No response from tag
          4    Non-specific reader error
          5    In process, still working
          6    In process success, stored response without length
          7    In process success, stored response with length
          8    In process success, send response without length
          9    In process success, send response with length
          10   In process error, stored response without length
          11   In process error, stored response with length
          12   In process error, send response without length
          13   In process error, send response with length
          14   Not supported
          15   Insufficient privilege
          16   Crypto suite error
          17   Response buffer overflow error
          18   Command not encapsulated
```

KeyID: Integer. Key identifier

Privilege: Bits. Defining the privilege of key

14.2.9.11 AntennaEvent Parameter

This event is generated when the Reader detects that an antenna is connected or disconnected.

Compliance requirement: Compliant Readers and Clients MAY implement this parameter.

AntennaEvent Parameter

AntennaID: Unsigned Short Integer

EventType: Integer

Possible Values:

```
Value   Definition
-----   ----------
  0      Antenna disconnected
  1      Antenna connected
```

14.2.9.12 ConnectionAttemptEvent Parameter

This status report parameter establishes Reader connection status when the Client or Reader initiates a connection. See section *19.1*, TCP Transport, for more details regarding the use of this report.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ConnectionAttemptEvent Parameter

Status: Integer

Possible Values:

```
Value   Definition
-----   ----------
 0       Success
 1       Failed (a Reader initiated connection already exists)
 2       Failed (a Client initiated connection already exists)
 3       Failed (any reason other than a connection already exists
 4       Another connection attempted
```

14.2.9.13 ConnectionCloseEvent Parameter

This status report parameter informs the Client that, unsolicited by the Client, the Reader will close the connection between the Reader and Client. Before the Reader closes a connection with the Client that is not solicited by the Client, the Reader SHALL first attempt to send a READER_EVENT_NOTIFICATION containing this parameter to the Client.

Once the Reader sends this event to the Client, the Reader SHALL close the connection to the Client. This is also to say that, once the Reader sends this event, the Reader SHALL send no additional messages to the Client and the Reader SHALL ignore any messages received from the Client until another new connection is established.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ConnectionCloseEvent Parameter

14.2.9.14 SpecLoopEvent Parameter

This status report parameter informs the Client that a ROSpec ending with a LoopSpec parameter has finished executing all items in its ListOfSpecs, and will continue by executing the first item in its ListOfSpecs. This event is generated once for each complete loop through a ROSpec's ListOfSpecs. If LoopCount > 4294967295 for the current inventory, the reader SHALL report 4294967295.

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

SpecLoopEvent Parameter

ROSpecID: Unsigned Integer. The identifier of the ROSpec that has looped execution of its ListOfSpecs.

LoopCount: Unsigned Integer. The number of times execution of the ROSpec's ListOfSpecs has been completed. The first time a SpecLoopEvent is generated by an ROSpec, the LoopCount is 1.

15 Errors

This section describes the errors that are solely based on LLRP protocol message parsing. The Reader SHALL discard the message if there is at least one error in the message, or cannot be fully processed. In addition, no portion of the message containing an error SHALL be executed by the Reader. In case the message has one or more errors, the Reader SHALL return at least one error parameter for one of the errors. The Reader MAY return more than one error parameter, one for each error. The errors are conveyed using a combination of 'generic error codes', a pointer to the culprit parameter/field, and a description of the error encoded as a string of UTF-8 characters.

Typically the errors in the LLRP defined messages are conveyed inside of the responses from the Reader. However, in cases where the message received by the Reader contains an unsupported message type, or a CUSTOM_MESSAGE with unsupported parameters or fields, the Reader SHALL respond with the ERROR_MESSAGE.

When a Reader or Client receives a command or notification with a version that is not supported, the receiver SHALL send an ERROR_MESSAGE in reply consisting of: A version that is the same as the received message, the message ID that matches the received message, and an LLRPStatusParameter with the ErrorCode set to M_UnsupportedVersion. This message SHALL contain no sub-parameters (such as Field Error, Parameter Error).

Readers and Clients SHALL not respond to an ERROR_MESSAGE.

15.1 Messages

15.1.1 ERROR_MESSAGE

This message is issued by the Reader to the Client, and it contains the LLRPStatus parameter that describes the error in the message.

Compliance requirement: Compliant Readers and Clients SHALL implement this message.

ERROR_MESSAGE

Error: <LLRPStatus Parameter>

15.2 Parameters

First, the error codes are presented, and then later, error parameters are presented that identify the culprit field in the message.

15.2.1 LLRP Status Codes

Status can be a success or one of the error conditions. This section lists a set of generic error conditions that, in combination with the identifier of the culprit field, conveys the error condition. The codes are broken into four scopes: message, parameter, field and device. The device code indicates that the error is in the Reader device rather than the message, parameter or field.

StatusCode	Name	Scope	Description
0	M_Success	**Message**	This code SHALL indicate that the message was received and processed successfully.
100	M_ParameterError		This code SHALL indicate that an error occurred with a parameter of this message.
101	M_FieldError		This code SHALL indicate that an error occurred with a field of this message.
102	M_UnexpectedParameter		This code SHALL indicate that an unexpected parameter was received with this message.
103	M_MissingParameter		This code SHALL indicate that a required parameter was missing from this message.
104	M_DuplicateParameter		This code SHALL indicate that a parameter, for which there must only be one instance at the Reader, was seen more than once in this message.
105	M_OverflowParameter		This code SHALL indicate that the maximum number of instances of the parameter has been exceeded at the Reader.
106	M_OverflowField		This code SHALL indicate that the maximum number of instances of the field has been exceeded at the Reader.
107	M_UnknownParameter		This code SHALL indicate that an unknown parameter was received in the message.
108	M_UnknownField		This code SHALL indicate that the field is unknown or not found at the Reader.
109	M_UnsupportedMessage		This code SHALL indicate that an unsupported message type was received.
110	M_UnsupportedVersion		This code SHALL indicate that the LLRP version in the received message is not supported by the Reader.
111	M_UnsupportedParameter		This code MAY indicate that the Parameter in the received message is not supported by the Reader.
112	M_UnexpectedMessage		This code SHALL indicate that the message received was unexpected by the Reader.

StatusCode	Name	Scope	Description
200	P_ParameterError	**Parameter**	This code SHALL indicate that an error occurred with a parameter of this parameter.
201	P_FieldError		This code SHALL indicate that an error occurred with a field of this parameter.
202	P_UnexpectedParameter		This code SHALL indicate that an unexpected parameter was received with this message.
203	P_MissingParameter		This code SHALL indicate that a required parameter was missing from this parameter.
204	P_DuplicateParameter		This code SHALL indicate that a parameter, for which there must only be one instance, was seen more than once in this parameter.
205	P_OverflowParameter		This code SHALL indicate that the maximum number of instances of the parameter has been exceeded at the Reader.
206	P_OverflowField		This code SHALL indicate that the maximum number of instances of the field has been exceeded at the Reader.
207	P_UnknownParameter		This code SHALL indicate that an unknown parameter was received with this message.
208	P_UnknownField		This code SHALL indicate that the field is unknown or not found at the Reader.
209	P_UnsupportedParameter		This code SHALL indicate that an unsupported parameter was received.
300	A_Invalid	**Field**	This code SHALL indicate that the field value was considered invalid for a non specific reason. An example is a message with invalid SpecID for a ROSpec or
301	A_OutOfRange		This code SHALL indicate that the field value did not fall within an acceptable range.
401	R_DeviceError	**Reader**	This code MAY indicate that there is a problem on the Reader rather than with a message, parameter, or field.

15.2.2 LLRPStatus Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

LLRPStatus Parameter

StatusCode: Integer.

Possible Values:

See the error code table (section *15.2.1*) for possible values within the Message, Parameter or Field scope.

FieldError: <FieldError Parameter> [Optional]

ParameterError: <ParameterError Parameter> [Optional]
ErrorDescription: UTF-8 String

15.2.2.1 FieldError Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

FieldError Parameter

FieldNum: Integer. Field number for which the error applies. The fields are numbered after the order in which they appear in the parameter or message body.

Possible Values: 0-65535

ErrorCode: Integer.

Possible Values:

See the error code table (section *15.2.1*) for possible values within the Argument scope.

15.2.2.2 ParameterError Parameter

Compliance requirement: Compliant Readers and Clients SHALL implement this parameter.

ParameterError Parameter

ParameterType: Integer. The parameter type that caused this error.

Possible Values: 0 - 1023

ErrorCode: Integer.

Possible Values:

See the error code table (section *15.2.1*) for possible values within the Parameter scope.

FieldError: <FieldError Parameter> [Optional]

ParameterError: <ParameterError Parameter> [Optional]

16 Binary Encoding for LLRP

This section contains the specific formats and operations for the binary encoding of the Low Level Reader Protocol. All fields and parameters must be encoded in the order shown in the diagrams in this section. This section does not contain information that has been generalised in the main body of the document. Refer to sections 8-16 for the description of the messages and the parameters and fields in the messages.

The binary encoding is based on a stream of octets. Each octet represents 8 bits of information. Octets within the data stream are serialised according to the particular transport mechanism over which this binding is carried. Octet numbering shown in this section is in network order. For example, in Figure 14, the first octet that a LLRP endpoint receives contains Rsvd, Ver and 2 bits of Message type.

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type										Message Length [31:16]															

Message Length [15:0]	Message ID[31:16]
Message ID[15:0]	
Message Value	

Figure 14: Network Order

Figure 15 illustrates the bit order inside the fields.

2	1	0	2	1	0	9	8	7	6	5	4	3	2	1	0	31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16
Rsvd			Ver			Message Type										Message Length [31:16]															

Figure 15: Bit Order in Fields

Integer numbers SHALL be encoded in network byte order with the most significant byte of the integer being sent first (big-Endian). Bit arrays are aligned to the most significant bit. Bit arrays are padded to an octet boundary. Pad bits SHALL be ignored by the Reader and Client.

The length of all messages within the binary encoding SHALL be multiples of octets. This means all parameters within the binary encoding SHALL be multiples of octets. This includes any custom or vendor specific parameter. All the messages and parameters in this section have been padded with zero to ensure that the length is a multiple of octets.

Reserved Bits

Reserved bits are added to fields where necessary to preserve octet-alignment of the binary LLRP stream. Future versions of LLRP may use reserved bits to extend certain types of fields (like Boolean, assuming the reserved bits are towards the MSb of the field), or to add new fields. Reserved bits are subject to the following rules:

- Clients SHALL ignore reserved bits.
- Readers SHALL error on non-zero reserved bits.
- Both Clients and Readers SHALL set reserved bits to zero.

Notations

Inside a message or a parameter,

- If a parameter X is denoted simply as X, it means that X is mandatory and appears once in the message.
- If a parameter X is denoted as X (0-n), it means that X is optional in the message, and it can appear multiple times in the message.
- If a parameter X is denoted as X (0-1), it means that X is optional in the message and that it can appear at most once in the message.
- If a parameter X is denoted as X (1-n), it means that X is mandatory and can appear multiple times in the message.

Negative Numbers

Negative numbers are represented using twos complement notation.

16.1 Messages

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
Message Value																															

Reserved bits: 3 bits

The reserved bits are reserved for future extensions. All reserved bits in messages SHALL be set to 0.

Ver: 3 bits

The version of LLRP. Implementations of LLRP based on this specification are using the value 0x2 except where explicitly noted otherwise (see Table 3 for the mapping between the values for this field and the protocol versions). Other values are reserved for future use.

Message Type: 10 bits

The type of LLRP message being carried in the message.

Message Length: 32 bits

This value represents the size of the entire message in octets starting from bit offset 0 of the first word. Therefore, if the Message Value field is zero-length, the Length field will be set to 10.

Message ID: 32 bits

As stated earlier, the communications between the Client and the Reader are primarily of a request- response type - requests/commands from the Client to the Reader, and responses from the Reader to the Client. In order to facilitate multiple outstanding commands/requests from the Client, LLRP uses a Message sequence number in each message. The Message sequence number is used to correlate a response with the original request. This sequence number is local to the LLRP channel.

Message Value: variable length

Dependent on the Message Type.

16.1.1 GET_SUPPORTED_VERSION

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type =46										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															

See section 9.1.1

16.1.2 GET_SUPPORTED_VERSION_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type =56										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																CurrentVersion								SupportedVersion							
LLRPStatus Parameter																															

See section 9.1.2

16.1.3 SET_PROTOCOL_VERSION

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1

Rsvd			Ver			Message Type =47										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																ProtocolVersion															

See section 9.1.3

16.1.4 SET_PROTOCOL_VERSION_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type =57										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 9.1.4

16.1.5 GET_READER_CAPABILITIES

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type =1										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																RequestedData															
Custom Parameter (0-n)																															

See section 10.1.1

16.1.6 GET_READER_CAPABILITIES_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 11										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															
GeneralDeviceCapabilities Parameter (0-1)																															
LLRPCapabilities Parameter (0-1)																															
RegulatoryCapabilities Parameter (0-1)																															
AirProtocolLLRPCapabilities Parameter (0-1)																															
Custom Parameter (0-n)																															

See section 10.1.2

16.1.7 ADD_ROSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 20										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															

ROSpec Parameter																															

See section 11.1.1

16.1.8 ADD_ROSPEC_RESPONSE

0											1										2										3	
0	1	2	3		4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd				Ver			Message Type = 30										Message Length [31:16]															
	Message Length [15:0]																Message ID[31:16]															
	Message ID[15:0]																															
	LLRPStatus Parameter																															

See section 11.1.2

16.1.9 DELETE_ROSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 21										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																ROSpecID[31:16]															
ROSpecID[15:0]																															

See section 11.1.3

16.1.10 DELETE_ROSPEC_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type =31										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 11.1.4

16.1.11 START_ROSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 22										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																ROSpecID[31:16]															
ROSpecID[15:0]																															

See section 11.1.5

16.1.12 START_ROSPEC_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 32										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section *11.1.6*

16.1.13 STOP_ROSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 23										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																ROSpecID[31:16]															
ROSpecID[15:0]																															

See section *11.1.7*

16.1.14 STOP_ROSPEC_RESPONSE

0											1										2										3	
0	1	2	3	4	5	6	7	8	9		0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver				Message Type = 33										Message Length [31:16]															
	Message Length [15:0]																Message ID[31:16]															
	Message ID[15:0]																															
	LLRPStatus Parameter																															

See section *11.1.8*

16.1.15 ENABLE_ROSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 24										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																ROSpecID[31:16]															
ROSpecID[15:0]																															

See section *11.1.9*

16.1.16 ENABLE_ROSPEC_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 34										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 11.1.10

16.1.17 DISABLE_ROSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 25										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																ROSpecID[31:16]															
ROSpecID[15:0]																															

See section 11.1.11

16.1.18 DISABLE_ROSPEC_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 35										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 11.1.12

16.1.19 GET_ROSPECS

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 26										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															

See section 11.1.13

16.1.20 GET_ROSPECS_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 36										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															
ROSpec Parameter (0-n)																															

See section 11.1.14

16.1.21 ADD_ACCESSSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 40										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
AccessSpec Parameter																															

See section 12.1.1

16.1.22 ADD_ACCESSSPEC_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 50										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 12.1.2

16.1.23 DELETE_ACCESSSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 41										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																AccessSpecId[31:16]															
AccessSpecId[15:0]																															

See section 12.1.3

16.1.24 DELETE_ACCESSSPEC_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 51										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 12.1.4

16.1.25 ENABLE_ACCESSSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 42										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															

Message ID[15:0]																AccessSpecId[31:16]															
AccessSpecId[15:0]																															

See section 12.1.5

16.1.26 ENABLE_ACCESSSPEC_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 52										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 12.1.6

16.1.27 DISABLE_ACCESSSPEC

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 43										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																AccessSpecId[31:16]															
AccessSpecId[15:0]																															

See section 12.1.7

16.1.28 DISABLE_ACCESSSPEC_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 53										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 12.1.8

16.1.29 GET_ACCESSSPECS

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 44										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															

See section 12.1.9

16.1.30 GET_ACCESSSPECS_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 54										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															
AccessSpec Parameter (0-n)																															

See section *12.1.10*

16.1.31 CLIENT_REQUEST_OP

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 45										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
TagReportData Parameter																															

See section *12.1.11*

16.1.32 CLIENT_REQUEST_OP_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 55										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
ClientRequestResponse Parameter																															

See section *12.1.12*

16.1.33 GET_REPORT

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 60										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															

See section *14.1.1*

16.1.34 RO_ACCESS_REPORT

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 61										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
TagReportData Parameter (0-n)																															
RFSurveyReportReportData Parameter (0-n)																															
Custom Parameter (0-n)																															

See section 14.1.2.

16.1.35 KEEPALIVE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 62										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															

See section 14.1.3

16.1.36 KEEPALIVE_ACK

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 72										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															

See section 14.1.4

16.1.37 READER_EVENT_NOTIFICATION

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 63										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
ReaderEventNotificationData Parameter																															

See section 14.1.5

16.1.38 ENABLE_EVENTS_AND_REPORTS

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 64										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															

See section *14.1.6*

16.1.39 ERROR_MESSAGE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 100										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section *15.1.1*

16.1.40 GET_READER_CONFIG

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 2										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																Antenna ID															
RequestedData								GPIPortNum																GPOPortNum[15:8]							
GPOPortNum[7:0]																															
Custom Parameter (0-n)																															

See section *13.1.1*

16.1.41 GET_READER_CONFIG_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 12										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															
Identification Parameter (0-1)																															
AntennaProperties Parameter (0-n)																															
AntennaConfiguration Parameter (0-n)																															
ReaderEventNotificationSpec Parameter (0-1)																															
ROReportSpec Parameter (0-1)																															
AccessReportSpec Parameter (0-1)																															
LLRPConfigurationStateValue Parameter (0-1)																															
KeepaliveSpec Parameter (0-1)																															
GPIPortCurrentState Parameter (0-n)																															

GPOWriteData Parameter (0-n)																															
EventsAndReports Parameter (0-1)																															
Custom Parameter (0-n)																															

See section 13.1.2

16.1.42 SET_READER_CONFIG

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 3										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																R	Reserved														
ReaderEventNotificationSpec Parameter (0-1)																															
AntennaProperties Parameter (0-n)																															
AntennaConfiguration Parameter (0-n)																															
ROReportSpec Parameter (0-1)																															
AccessReportSpec Parameter (0-1)																															
KeepaliveSpec Parameter (0-1)																															
GPOWriteData Parameter (0-n)																															
GPIPortCurrentState Parameter (0-n)																															
EventsAndReports Parameter (0-1)																															
Custom Parameter (0-n)																															

Abbreviations

R - ResetToFactoryDefaults See section 13.1.3

16.1.43 SET_READER_CONFIG_RESPONSE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 13										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 13.1.4

16.1.44 CLOSE_CONNECTION

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 14										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															

See section 13.1.5.

16.1.45 CLOSE_CONNECTION_RESPONSE

0										1										2										3	

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 4										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																															
LLRPStatus Parameter																															

See section 13.1.6

16.1.46 CUSTOM_MESSAGE

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Rsvd			Ver			Message Type = 1023										Message Length [31:16]															
Message Length [15:0]																Message ID[31:16]															
Message ID[15:0]																Vendor Identifier [31:16]															
Vendor Identifier [15:0]																Message Subtype															
Vendor Specified Payload																															

See section 8.1

16.2 LLRP Parameters

LLRP parameters are defined in the following subsections with the exception that the air protocol specific LLRP parameters are defined in section 17.3. The binary encoding of LLRP uses two different encodings of parameters: Type-length-value (TLV) encoded parameters, and Type-value (TV) encoded parameters. The TV encoding is only used for encoding parameters that are fixed-length, and are in Reports and Notifications from the Reader. The use of a compact encoding (i.e., TV) for the Reports and Notifications helps improve the network efficiency.

16.2.1 TLV and TV Encoding of LLRP Parameter

The type of encoding (TLV or TV) is determined based on the value of bit 0 in the parameter header. All TLV-encoded Parameters SHALL have a 0 in bit 0 of the header. All TV-encoded Parameters SHALL have a 1 in bit 0 of the header.

16.2.1.1 TLV-Parameters

LLRP TLV-Parameters have the following encoding structure.

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Parameter Type										Parameter Length															
Parameter Value																															

Reserved bits: 6 bits

The reserved bits are reserved for future extensions. All reserved bits SHALL be set to 0.

Parameter Type: 10 bits

This is the type of LLRP parameter being carried in the message. The parameter number space for the TLV-parameters is 128 – 2047. The number space 0-127 is reserved for TV-parameters.

Parameter Length: 16 bits

This value represents the size of the entire parameter in bytes starting from bit offset 0 of the first word. Therefore, if the Parameter Value field is zero-length, the Parameter Length field will be set to 4.

Parameter Value: variable length

Dependent on the Parameter Type.

16.2.1.1.1 Encoding Guidelines for TLV-Parameters

The following rules apply to TLV-Parameters:

- Parameters may contain mandatory and optional fields.
- Parameter fields may be passed by value or by sub-parameter.
- Mandatory fields will always be present and optional fields may or may not be present.
- Mandatory fields of fixed length will be passed by value only, using the order, size and alignment defined in this document.
- A mandatory field of variable length must be passed by value if it is the only field, mandatory or optional, of variable length in that parameter.
- A parameter with multiple mandatory or optional fields of variable length must pass them as sub-parameters.
- A parameter containing a field of variable length being passed by value may not contain sub-parameters.
- Optional fields will always be passed as sub-parameters.

The following rules apply to sub-parameters:

- Sub-parameters follow all parameter rules.
- A sub-parameter is a parameter that is encompassed within the length of a preceeding parameter and adds to the dataset of the encapsulating parameter.

```
ParameterA-length---------------------------+

Data                                        |
        ParameterB-length---+               |
        Data               <-+              |
        ParameterC-length---------------+   |
        Data                            |   |
                ParameterD-length---+   |   |
                Data               <-+ <-+ <-+
```

- Sub-parameters may be mandatory or optional.

16.2.1.2 TV-Parameters

LLRP TV-Parameters have the following encoding structure.

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Parameter Type																														
Parameter Value																															

Parameter Type: 8 bits

This is the type of LLRP parameter being carried in the message. The parameter number space for the TV-parameters is 1 – 127. The number space 128-2047 is reserved for TLV-parameters.

Parameter Value: variable length

Dependent on the Parameter Type.

16.2.1.2.1 Encoding Guidelines for TV-Parameters

The following rule applies to TV-Parameters:

- TV-Parameters cannot contain sub-parameters (TLV or TV-Parameters).

16.2.2 General Parameters

16.2.2.1 UTCTimestamp Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 128										Length															
Microseconds [63:32]																															
Microseconds [31:0]																															

See section 11.2.1.1.2

16.2.2.2 Uptime Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 129										Length															
MicroSeconds [63:32]																															
Microseconds [31:0]																															

See section 11.2.1.1.2

16.2.3 Reader Device Capabilities Parameters

16.2.3.1 GeneralDeviceCapabilities Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 137										Length															
MaxNumberOfAntennaSupported																C	T	Reserved													
Device manufacturer name																															
Model Name																															
FirmwareVersionByteCount																															
Reader Firmware Version: Variable length UTF-8 String																															
ReceiveSensitivityTableEntry Parameter (1-n)																															
PerAntennaReceiveSensitivityRange Parameter (0-n)																															
GPIOCapabilities Parameter																															
PerAntennaAirProtocol Parameter (1-n)																															
MaximumReceiveSensitivity Parameter (0-1)																															

See section 10.2.1

Abbreviations

C –
CanSetAntennaProperties
T –
HasUTCClockCapability

16.2.3.1.1 MaximumReceiveSensitivity Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 363										Length															
Maximum Sensitivity Value																															

See section 10.2.1.1

16.2.3.1.2 ReceiveSensitivityTableEntry Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 139										Length															
Index																Receive Sensitivity Value															

See section 10.2.1.2.

16.2.3.1.3 PerAntennaReceiveSensitivityRange Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 149										Length															
AntennaID																ReceiveSensitivityIndexMin															
ReceiveSensitivityIndexMax																															

See section 10.2.1.3.

16.2.3.1.4 PerAntennaAirProtocol Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 140										Length															
AntennaId																NumProtocols															
ProtocolID#1								ProtocolID#2								…..								ProtocolID#P							

See section 10.2.1.4.

16.2.3.1.5 GPIOCapabilities Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 141										Length															
NumGPIs																NumGPOs															

See section 10.2.1.5.

16.2.3.2 LLRPCapabilities Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 142										Length															
C	R	S	T	H	Reserved			MaxPriorityLevelSupported								ClientRequestOpSpecTimeout															
MaxNumROSpecs																															
MaxNumSpecsPerROSpec																															
MaxNumInventoryParameterSpecsPerAISpec																															
MaxNumAccessSpecs																															
MaxNumOpSpecsPerAccessSpec																															

Abbreviations

C – CanDoRFSurvey
R – CanReportBufferFillWarning
S – SupportsClientRequestOpSpec
T – CanDoTagInventoryStateAwareSingulation
H – SupportsEventAndReportHolding
MaxNumPriority –

MaxNumPriorityLevelsSupported

See section 10.2.2

16.2.3.3 AirProtocolLLRPCapabilities Parameter

See section 10.2.3

There is no separate binary encoding for AirProtocolLLRPCapabilities. Each Air protocol's capabilities are expressed in a different LLRP Parameter. Refer to section 10.2.3 for air protocol specific capability

parameters. For example, the C1G2LLRPCapabiltities Parameter (section 10.2.3) carries the C1G2 air protocol capabilities.

16.2.3.4 RegulatoryCapabilities Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 143										Length															
Country Code																Communications Standard															
UHFBandCapabilities Parameter (0-1)																															
Custom Paramter (0-n)																															

See section 10.2.4

16.2.3.4.1 UHFBandCapabilities Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 144										Length															
TransmitPowerLevelTableEntry Parameter (1-n)																															
FrequencyInformation Parameter																															
UHFRFModeTable Parameter (1-n)																															

RFSurveyFrequencyCapabilities Parameter (0-1)																															

See section 10.2.4.1

TransmitPowerLevelTableEntry Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 145										Length															
Index																TransmitPowerValue															

See section 10.2.4.2

FrequencyInformation Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 146										Length															
H	Reserved																														
FrequencyHopTable Parameter (0-n)																															
FixedFrequencyTable (0-1)																															

Abbreviations

H – Hopping

See section 10.2.4.2.1

FrequencyHopTable Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 147										Length															
HopTableId								Reserved								NumHops															
Frequency#1																															
….																															
Frequency#n																															

NumHops: Number of entries in the List of

Frequencies.

See section 10.2.4.2.2

FixedFrequencyTable Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 148										Length															
NumFrequencies																Frequency#1[31:16]															
Frequency#1[15:0]																….															
….																															
Frequency#n [15:0]																															

NumFrequencies: Number of entries in the List of Frequencies.

See section 10.2.4.1.2.2.

RFSurveyFrequencyCapabilities Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 365										Length															
MinimumFrequency																															
MaximumFrequency																															

See section 10.2.4.1.3.

16.2.4 Reader Operations Parameters

16.2.4.1 ROSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 177										Length															
ROSpecID																															
Priority								CurrentState																							
ROBoundarySpec Parameter																															
SpecParameter (1-n) [See notes below]																															
ROReportSpec Parameter (0-1)																															

Notes

Each SpecParameter can be one of the following types: AISpec Parameter or RFSurveySpec Parameter or LoopSpec Parameter or Custom Parameter.

See section 11.2.1

ROBoundarySpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 178										Length															
ROSpecStartTrigger Parameter																															
ROSpecStopTrigger Parameter																															

See section 11.2.1.1

ROSpecStartTrigger Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 179										Length															
ROSpecStartTriggerType																															
PeriodicTriggerValue Parameter (0-1)																															
GPITriggerValue Parameter (0-1)																															

See section 11.2.1.1

PeriodicTriggerValue Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 180										Length															
Offset																															
Period																															
UTCTimestamp Parameter (0-1)																															

See section 11.2.1.1

GPITriggerValue Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 181										Length															
GPIPortNum																E	Reserved							Timeout[31:24]							
Timeout [23:0]																															

Abbreviations

E – GPIEvent

See section 11.2.1.1

ROSpecStopTrigger Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 182										Length															
ROSpecStopTriggerType								DurationTriggerValue[31:8]																							
DurationTriggerValue[7:0]																															
GPITriggerValue Parameter (0-1)																															

See section 11.2.1.1.

16.2.4.2 AISpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 183										Length															
AntennaCount																AntennaID#1															
……																AntennaID#n															
AISpecStopTrigger Parameter																															
InventoryParameter Spec Parameter (1-n)																															
Custom Parameter (0-n)																															

See section 11.2.2

16.2.4.2.1 AISpecStopTrigger Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 184										Length															
AISpecStopTriggerType								DurationTrigger[31:24]																							

DurationTrigger[7:0]																															
GPITriggerValue Parameter (0-1)																															
TagObservationTrigger Parameter (0-1)																															

See section *11.2.2.1*

TagObservationTrigger Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 185										Length															
TriggerType								Reserved								NumberOfTags															
NumberOfAttempts																T															
Timeout																															

See section 11.2.2.1.1.

InventoryParameterSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 186										Length															
InventoryParameterSpecID																ProtocolID															
AntennaConfigurationParameter (0-n)																															
Custom Parameter (0-n)																															

See section *11.2.2.2*

16.2.4.3 RFSurveySpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 187										Length															
AntennaID																StartFrequency[31:16]															
StartFrequency[15:0]																EndFrequency[31:16]															
EndFrequecy[15:0]																															
RFSurveySpecStopTrigger Parameter																															
Custom Parameter (0-n)																															

See section *11.2.3*

16.2.4.3.1 RFSurveySpecStopTrigger Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 188										Length															
StopTriggerType								Duration [31:24]																							
Duration [7:0]								N[31:8]																							
N[7:0]																															

See section *11.2.3.1*

16.2.4.4 LoopSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 355										Length															
LoopCount																															

See section 11.2.4

16.2.5 Access Operation Parameters

16.2.5.1 AccessSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 207										Length															
AccessSpecID																															
AntennaId																ProtocolId								C	Reserved						
ROSpecID																															
AccessSpecStopTrigger Parameter																															
AccessCommand Parameter																															
AccessReportSpec Parameter (0-1)																															
Custom Parameter (0-n)																															

Abbreviations

C – CurrentState

See section 12.2.1.

16.2.5.1.1 AccessSpecStopTrigger Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 208										Length															
AccessSpecStopTrigger								OperationCountValue																							

See section 12.2.1.1

16.2.5.1.2 AccessCommand Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 209										Length															
TagSpecParameter [See notes below]																															
OpSpecParameter (1-n) [See notes below]																															
Custom Parameter (0-n)																															

Notes

TagSpecParameter is the air protocol specific tag spec parameter. For C1G2, it is C1G2TagSpec Parameter.

Each OpSpecParameter can be one of the following types: Air protocol specific OpSpec (e.g., C1G2OpSpec Parameter), ClientRequestOpSpec Parameter, or Custom Parameter.
See section 12.2.1.2

16.2.5.1.3 ClientRequestOpSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 210										Length															
OpSpecID																															

See section 12.2.1.2

16.2.5.1.4 ClientRequestResponse Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 211										Length															
AccessSpecID																															
EPCDataParameter																															
OpSpecParameter (0-n) [See notes below]																															

Notes

Each OpSpecParameter is an Air protocol specific opspec (e.g., C1G2OpSpec Parameter).

See section 12.2.2

16.2.6 Configuration Parameters

16.2.6.1 LLRPConfigurationStateValue Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 217										Length															
LLRPConfigurationStateValue																															

See section 13.2.1

16.2.6.2 Identification Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 218										Length															
IDType								ByteCount																							
Reader ID(Variable length)																															

See section 13.2.2

16.2.6.3 GPOWriteData Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 219										Length															
GPO Port Number																W	Reserved														

Abbreviations

W – GPO Data

See section 13.2.3

16.2.6.4 KeepaliveSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 220										Length															
KeepaliveTriggerType								TimeInterval																							
TimeInterval																															

See section 13.2.4

16.2.6.5 AntennaProperties Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 221										Length															
C	Reserved							AntennaId																AntennaGain[15:8]							
AntennaGain[7:0]																															

Abbreviations

C – Antenna connected

See section 13.2.5

16.2.6.6 AntennaConfiguration Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 222										Length															
AntennaId																															
RFReceiver Parameter (0-1)																															
RFTransmitter Parameter (0-1)																															
AirProtocolInventoryCommandSettings Parameter (0-n)																															
Custom Parameter (0-n)																															

Notes:

Each AirProtocolInventoryCommandSettingsParameter instance is an Air protocol specific Parameter (e.g., C1G2InventoryCommand Parameter).

See section 13.2.6

16.2.6.7 RFReceiver Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 223										Length															
Receiver Sensitivity																															

See section 13.2.6.1

16.2.6.8 RFTransmitter Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 224										Length															
HopTableId																ChannelIndex															
TransmitPower																															

See section 13.2.6.2.

16.2.6.9 GPIPortCurrentState Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 225										Length															
GPIPortNum																C	Reserved							GPIState							

Abbreviations

C – GPIConfig

See section 13.2.6.3.

16.2.6.10 EventsAndReports Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 226										Length															
H	Reserved																														

Abbreviations

H – HoldEventsAndReportsUponReconnect

See section 13.2.6.4

16.2.7 Reporting Parameters

16.2.7.1 ROReportSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 237										Length															
ROReportTrigger								N																							
TagReportContentSelector Parameter																															
Custom Parameter (0-n)																															

See section 14.2.1.

16.2.7.1.1 TagReportContentSelector Parameter

0										1										2										3	

0	1	2	3	4	5	6	7	8	9	0	1	3		4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 238										Length															
R	I	P	A	C	R	F	L	T	S	**Y**	Reserved																				
AirProtocolSpecificEPCMemorySelectorParameter (0-n) [See notes below]																															
Custom Parameter (0-n)																															

Abbreviations
R – EnableROSpecID
I – EnableSpecIndex
P – EnableInventoryParameterSpecID
A – EnableAntennaID
C – EnableChannelIndex
R – EnablePeakRSSI
F – EnableFirstSeenTimestamp
L – EnableLastSeenTimestamp
T – EnableTagSeenCount

S – EnableAccessSpecID

Y - EnableCryptoResponse

Notes:
Each instance of AirProtocolSpecificEPCMemorySelectorParameter is one of the air protocol specific selector parameters (e.g., C1G2EPCMemorySelector Parameter).

See section *14.2.1.1.*

16.2.7.2 AccessReportSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 239										Length															
AccessReportTrigger																															

See section *14.2.2.*

16.2.7.3 TagReportData Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 240										Length															
EPCDataParameter [See notes below]																															
ROSpecID Parameter (0-1)																															
SpecIndex Parameter (0-1)																															
InventoryParameterSpecID Parameter (0-1)																															
AntennaID Parameter (0-1)																															
PeakRSSI Parameter (0-1)																															
ChannelIndex Parameter (0-1)																															
FirstSeenTimestampUTC Parameter (0-1)																															
FirstSeenTimestampUptime Parameter (0-1)																															
LastSeenTimestampUTC Parameter (0-1)																															
LastSeenTimestampUptime Parameter (0-1)																															
TagSeenCount Parameter (0-1)																															
AirProtocolTagDataParameter (0-n)[See Notes below]																															
AccessSpecID Parameter (0-1)																															
OpSpecResultParameter (0-n) [See notes below]																															
CryptoResponse(0-1)																															

Custom Parameter (0-n)																															

Notes:
The EPCDataParameter is either the EPCData Parameter or EPC-96 Parameter. The EPCData Parameter SHALL be used for encoding a non-96 bit EPC, whereas the EPC-96 Parameter SHALL be used for encoding a 96-bit EPC.

The AirProtocolTagDataParameter is one or more air protocol specific tag data parameters (e.g., C1G2PC, C1G2XPCW1, C1G2XPCW2, and C1G2CRC). In the C1G2 case, each parameter, C1G2PC, C1G2XPCW1, C1G2XPCW2, and C1G2CRC, is optional in the TagReportData Parameter.

OpSpecResultParameter: Either an air protocol specific OpSpec result parameter (e.g., C1G2OpSpecResult Parameter), a ClientRequestOpSpecResult Parameter, or a Custom Parameter.

See section *14.2.3*

16.2.7.3.1 EPCData Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 241										Length															
EPCLengthBits																															
EPC																															

EPCLengthBits: Number of bits in the EPC.
See section *14.2.3.2.*

16.2.7.3.2 EPC-96 Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=13							EPC[95:72]																							
EPC[71:40]																															
EPC[39:8]																															
EPC[7:0]																															

See section *14.2.3.2*.

16.2.7.3.3 ROSpecID Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=9							ROSpecID[31:8]																							
ROSpecID[7:0]																															

See section *14.2.3.3.*

16.2.7.3.4 SpecIndex Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=14							SpecIndex																							

See section *14.2.3.4.*

16.2.7.3.5 InventoryParameterSpecID Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=10							InventoryParameterSpecId																							

See section 14.2.3.5.

16.2.7.3.6 AntennaID Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=1							AntennaId																							

See section 14.2.3.6.

16.2.7.3.7 PeakRSSI Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=6							PeakRSSI																							

See section 14.2.3.7.

16.2.7.3.8 ChannelIndex Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=7							ChannelIndex																							

See section 14.2.3.8.

16.2.7.3.9 FirstSeenTimestampUTC Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=2							Microseconds [63:40]																							
Microseconds [39:8]																															
Microseconds[7:0]																															

See section 14.2.3.9.

16.2.7.3.10 FirstSeenTimestampUptime Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=3							Microseconds[63:40]																							
Microseconds [39:8]																															
Microseconds[7:0]																															

See section 14.2.3.10.

16.2.7.3.11 LastSeenTimestampUTC Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=4							Microseconds[63:40]																							
Microseconds[39:8]																															
Microseconds[7:0]																															

See section 14.2.3.11.

16.2.7.3.12 LastSeenTimestampUptime Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=5							Microseconds[63:40]																							
Microseconds[39:8]																															
Microseconds[7:0]																															

See section 14.2.3.12.

16.2.7.3.13 TagSeenCount Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=8							TagCount																							

See section 14.2.3.13.

16.2.7.3.14 ClientRequestOpSpecResult Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=15							OpSpecID																							

See section 14.2.3.14.

16.2.7.3.15 AccessSpecID Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=16							AccessSpecID[31:8]																							
AccessSpecID[7:0]																															

See section 14.2.3.15.

16.2.7.3.16 C1G2AuthenticateOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 374										Length															
Result								OpSpecID																RespBitLen[31:16]							
RespBitLen[15:0]								Response																							

RespBitLen: Number of bits in Response
Response: Response message
See section 14.2.9.10.10

16.2.7.3.17 C1G2AuthCommOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 375										Length															
Result								OpSpecID																RespBitLen[31:16]							
RespBitLen[15:0]								Response																							

RespBitLen: Number of bits in Response
Response: Response message

See section 14.2.9.10.11

16.2.7.3.18 C1G2SecureCommOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 376										Length															
Result								OpSpecID																RespBitLen[31:16]							
RespBitLen[15:0]								Response																							

RespBitLen: Number of bits in Response
Response: Response message
See section 14.2.9.10.12

16.2.7.3.19 C1G2ReadBufferOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 377										Length															
Result								OpSpecID																RespBitLen[31:16]							
RespBitLen[15:0]								Response																							

RespBitLen: Number of bits in Response
Response: Response message
See section 14.2.9.10.13

16.2.7.3.20 C1G2KeyUpdateOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 378										Length															
Result								OpSpecID																RespBitLen[31:16]							
RespBitLen[15:0]								Response																							

RespBitLen: Number of bits in Response
Response: Response message
See section 14.2.9.10.10

16.2.7.3.21 C1G2TagPrivilegeOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 379										Length															
Result								OpSpecID																Reserved							T
KeyID								Privilege																							

Length: Number of bits in Message
T: 0 – Access Password target, 1 – Key target
KeyID – Key Identifier
Privilege – Privilege bits. Refer Class 1 Gen 2 Air Interface Protocol standard for details

See section 14.2.9.10.10

16.2.7.3.22 CryptoResponse Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 290										Length															
RespBitLen																															
Message																															

RespBitLength: Number of bits in Message
Message: Response message
See section *14.2.6*

16.2.7.3.23 RFSurveyReportData Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 242										Length															
ROSpecID Parameter (0-1)																															
SpecIndex Parameter (0-1)																															
FrequencyRSSILevelEntry Parameter (1-n)																															
Custom Parameter (0-n)																															

See section *14.2.3.15.*

16.2.7.3.24 FrequencyRSSILevelEntry Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 243										Length															
Frequency																															
Bandwidth																															
Average RSSI								Peak RSSI																							
TimestampParameter [See notes below]																															

Notes:
TimestampParameter: Either UTCTimestamp Parameter or UptimeParameter.

See section *14.2.4.1*

16.2.7.3.25 ReaderEventNotificationSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 244										Length															
EventNotificationState Parameter(1-n)																															

See section *14.1.5*

16.2.7.3.26 EventNotificationState Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 245										Length															
EventType																S	Reserved														

Abbreviations:
S – NotificationState

See section 14.2.5.1.

16.2.7.4 ReaderEventNotificationData Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 246										Length															
TimestampParameter [See notes below]																															
HoppingEvent Parameter (0-1)																															
GPIEvent Parameter (0-1)																															
ROSpecEvent Parameter (0-1)																															
ReportBufferLevelWarningEvent Parameter (0-1)																															
ReportBufferOverflowErrorEvent Parameter (0-1)																															
ReaderExceptionEvent Parameter (0-1)																															
RFSurveyEvent Parameter (0-1)																															
AISpecEvent Parameter (0-1)																															
AntennaEvent Parameter (0-1)																															
ConnectionAttemptEvent Parameter (0-1)																															
ConnectionCloseEvent Parameter (0-1)																															
SpecLoopEvent Parameter (0-1)																															
Custom Parameter (0-n)																															

Notes:
TimestampParameter: Either UTCTimestamp Parameter or Uptime

Parameter.

See section 14.2.9 bookmark179

16.2.7.4.1 HoppingEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 247										Length															
HopTableID																NextChannelIndex															

See section 14.2.6.2.

16.2.7.4.2 GPIEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 248										Length															

GPIPortNumber																E	Reserved														

Abbreviations

E – GPIEvent

See section *14.2.6.3.*

16.2.7.4.3 ROSpecEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 249										Length															
EventType								ROSpecID[31:8]																							
ROSpecID[7:0]								PreemptingROSpecID[31:8]																							
PreemptingROSpecID[7:0]																															

See section *14.2.6.4.*

16.2.7.4.4 ReportBufferLevelWarningEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 250										Length															
ReportBufferPercentageFull																															

See section *14.2.6.5.*

16.2.7.4.5 ReportBufferOverflowErrorEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 251										Length															

See section *14.2.6.6.*

16.2.7.4.6 ReaderExceptionEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 252										Length															
Message String ByteCount																															
Message: Variable length UTF-8String																															
ROSpecID Parameter (0-1)																															
SpecIndex Parameter (0-1)																															
InventoryParameterSpecID Parameter (0-1)																															
AntennaID Parameter (0-1)																															
AccessSpecID Parameter (0-1)																															
OpSpecID Parameter (0-1)																															
Custom Parameter (0-n)																															

See section *14.2.6.7.*

OpSpecID Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=17							OpSpecID																							

See section 13.2.5.7.1.

16.2.7.4.7 RFSurveyEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 253										Length															
EventType								ROSpecID[31:8]																							
ROSpecID[7:0]								SpecIndex[15:0]																							

See section 14.2.6.7.1.

16.2.7.4.8 AISpecEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 254										Length															
EventType								ROSpecID[31:8]																							
ROSpecID[7:0]								SpecIndex[15:0]																							
AirProtocolSingulationDetailsParameter (0-1) [See notes below]																															

See section 14.2.6.9.

Notes:

AirProtocolSingulationDetailsParameter is one of the air protocol specific singulation parameters (e.g., C1G2SingulationDetails Parameter).

16.2.7.4.9 AntennaEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 255										Length															
EventType								AntennaID																							

See section 14.2.6.10.

16.2.7.4.10 ConnectionAttemptEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 256										Length															
Status																															

See section 14.2.6.11.

16.2.7.4.11 ConnectionCloseEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 257										Length															

See section 14.2.6.12.

16.2.7.4.12 SpecLoopEvent Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 356										Length															
ROSpecID																															
LoopCount																															

See section 14.2.6.13.

16.2.8 LLRP Error Parameters

16.2.8.1 LLRPStatus Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 287										Length															
StatusCode																Error Description ByteCount															
Error Description: Variable length UTF-8 String																															
FieldError Parameter (0-1)																															
ParameterError Parameter (0-1)																															

See section 15.2.2.

16.2.8.1.1 FieldError Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 288										Length															
FieldNum																ErrorCode															

See section 15.2.2.1.

16.2.8.1.2 ParameterError Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 289										Length															
ParameterType																ErrorCode															
Field Error Parameter (0-1)																															
Parameter Error Parameter (0-1)																															

See section 15.2.2.2.

16.2.9 Custom Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1

Reserved						Type=1023										Parameter Length															
Vendor ID																															
Subtype																															
VendorParameter Value																															

See section 8.2.

16.3 Air Protocol Specific Parameters

This section defines air protocol specific parameter encodings. There is a separate subsection here for each air protocol defined by LLRP.

16.3.1 Class-1 Generation-2 (C1G2) Protocol Parameters

The Class-1 Generation-2 (C1G2) Air Protocol is specified by the EPCglobal Class-1 Generation-2 UHF RFID Protocol v1.1.0 specification.

The following subsections specify LLRP air protocol specific parameter encodings.

- Capabilities Parameters
- Reader Operations Parameters
- Access Operation Parameters
- Configuration Parameters
- Reporting Parameters

16.3.1.1 Capabilities Parameters

This section of air protocol specific parameters corresponds to LLRP parameters encodings specified in section 10.2.

16.3.1.1.1 C1G2LLRPCapabilities Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type =327										Length															
E	W	P	R	U	X	Rsvd		MaxNumSelectFiltersPerQuery																UT	CH	AU	AC	SC	KU	TP	RB
EP	Reserved																														

Abbreviations

E – CanSupportBlockErase
W – CanSupportBlockWrite
P – CanSupportBlockPermalock

R – CanSupportTagRecommissioning - DEPRECATED / RFU

U – CanSupportUMIMethod2
X - CanSupportXPC
UT – CanSupportUntraceable
CH- CanSupportChallenge
AU – CanSupportAuthenticate
AC – CanSupportAuthComm
SC – CanSupportSecureComm
KU – CanSupportKeyUpdate
TP – CanSupportTagPrivilege
RB – CanSupportReadBuffer

EP – CanSupportExtendedPowerOnPeriod

See section 10.2.3

16.3.1.1.2 UHFC1G2RFModeTable Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 328										Length															
UHFC1G2RFModeTableEntry Parameter (1-n)																															

See section 10.2.4.4

UHFC1G2RFModeTableEntry Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 329										Length															
Mode identifier																															
R	C	Reserved						Mod								FLM								M							
BDR Value																															
PIE Value																															
MinTariValue																															
MaxTariValue																															
StepTariValue																															

Abbreviations

R – DR Value

M– Spectral Mask Indicator

Mod – M value / Modulation

FLM – Forward Link Modulation

C – EPC HAG T&C Conformance

See section 10.2.4.4.1

16.3.1.2 Reader Operations Parameters

This section of air protocol specific parameters corresponds to LLRP parameters encodings specified in section 11.2

16.3.1.2.1 C1G2InventoryCommand Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 330										Length															
S	Reserved																														
C1G2Filter Parameter (0-n)																															
C1G2RFControl Paremeter (0-1)																															
C1G2SingulationControl Parameter (0-1)																															
C1G2Challenge Parameter (0-1)																															
Custom Parameter (0-n)																															

Abbreviations

S – TagInventoryStateAware

See section 13.2.6.3

C1G2Filter Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type =331										Length															
T		Reserved																													
C1G2TagInventoryMask Parameter																															
C1G2TagInventoryStateAwareFilterAction Parameter (0-1)																															
C1G2TagInventoryStateUnawareFilterAction Parameter (0-1)																															
ExtendOnTime(0-1)																															

See section 13.2.6.3.1

C1G2TagInventoryMask Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 332										Length															
MB		Reserved						Pointer[15:0]																MaskBitCount[15:8]							
MaskBitCount[7:0]																															
Tag Mask																															

See section 13.2.6.3.1

C1G2TagInventoryStateAwareFilterAction Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 333										Length															
Target								Action																							

See section 13.2.6.3.1

C1G2TagInventoryStateUnawareFilterAction Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 334										Length															
Action																															

See section 13.2.6.3.1

C1G2RFControl Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 335										Length															
ModeIndex																Tari															

See section 13.2.6.3.1

C1G2SingulationControl Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type=336										Length															
S		Reserved						TagPopulation																TagTransitTime[31:24]							
TagTransitTime[23:0]																															
C1G2TagInventoryStateAwareSingulationAction Parameter (0-1)																															

Abbreviations:
S –Session
See section 13.2.6.3.1

C1G2TagInventoryStateAwareSingulationAction Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 337										Length															
I	S	A																													

Abbreviations
A – S_All
See section 13.2.6.3.1

C1G2Challenge Parameter

0											1										2										3	
0	1	2	3	4	5		6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved							Type = 366										Length															
Reserved							L	E	CSI								MsgLen															
	Message																															
	ExtendOnTime																															

L – 0: Omit length from reply, 1: Include length in reply
E – 0: Do not transmit result with EPC, 1: Transmit result with EPC
CSI – Crypto suite identifier
MsgLen: Length of message
Message: Message dependent on CSI
ExtendedOnTime: Time in milliseconds to extend continuous wave transmission after issuing Challenge command
See section 13.2.6.3.4.

ExtendedOnTime Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 381										Length															
Time																															

See section 13.2.6.3.1

16.3.1.3 Access Operation Parameters

This section of air protocol specific parameters corresponds to LLRP parameters encodings specified in section 12.2

16.3.1.3.1 C1G2TagSpec Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 338										Length															
C1G2TargetTag Parameter																															
C1G2TargetTag Parameter (0-1)																															

See section 12.2.1.3

C1G2TargetTag Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 339										Length															
MB	M	Resvd						Pointer																MaskBitCount[15:8]							
MaskBitCount[7:0]																															
Tag Mask																															
DataBitCount																															
Tag Data																															

Abbreviations

M – Match.

See section 12.2.1.3.1

16.3.1.3.2 C1G2 OpSpecs

C1G2Read Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 341										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																MB		Reserved						WordPointer[15:8]							
WordPointer[7:0]								WordCount																							

See section 12.2.1.3.2

C1G2Write Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 342										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																MB		Reserved						WordPointer[15:8]							
WordPointer[7:0]								WriteDataWordCount																							
Write Data																															

See section 12.2.1.3.2

C1G2Kill Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 343										Length															
OpSpecID																KillPassword[31:16]															
KillPassword[15:0]																															

See section 12.2.1.3.2

C1G2Lock Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 344										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																															
C1G2LockPayload Parameter (1-n)																															

See section 12.2.1.3.2

C1G2LockPayload Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 345										Length															
Privilege								DataField																							

See section 12.2.1.3.2

C1G2BlockErase Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 346										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																MB		Reserved						WordPointer[15:8]							
WordPointer[7:0]								WordCount																							

See section 12.2.1.3.2

C1G2BlockWrite Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 347										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																MB		Reserved						WordPointer[15:8]							
WordPointer[7:0]								WriteDataWordCount																							
Write Data																															

See section 12.2.1.3.2

C1G2BlockPermalock Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 358										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																MB		Reserved						BlockPointer[15:8]							
BlockPointer[7:0]								BlockMaskWordCount																							
BlockMask																															

See section 12.2.1.3.2

C1G2GetBlockPermalockStatus Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 359										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																MB		Reserved						BlockPointer[15:8]							
BlockPointer[7:0]								BlockRange																							

See section 12.2.1.3.2

C1G2Untraceable Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 380										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																U	Reserved							EPC							
TID		User	Range		Resvd																										

U –	0: De-assert U bit in XPC_WI
	1: Assert U bit in XPC_W1
EPC[5:0] – MSB bit[5] -	0: show memory above EPC
	1: hide memory above EPC
LSB bits [4:0] –	New EPC length field (new L bits)
TID[1:0] -	00: Hide none
	01: Hide some
	10: Hide all
	11: RFU
User -	0: View
	1: Hide
Range[1:0] -	00: Normal
	01: Toggle temporarily
	10: Reduced
	11: RFU

See section 12.2.1.3.2

C1G2Authenticate Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 367										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																S	L	Reserved						CSI							
MsgLen																Message															

Message																															

S – 0: Store reply, 1: Send reply
L – 0: Omit length from reply, 1: Include length in reply
CSI – Crypto suite identifier
MsgLen: Length of message
Message: Message dependent on CSI
See section 12.2.1.3.2

C1G2AuthComm Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 368										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																L	Reserved							MsgLen							
MsgLen								Message																							
Message																															

L – 0: Omit length from reply, 1: Include length in reply
MsgLen: Length of message
Message: Crypto message
See section 14.2.9.10.11

C1G2SecureComm Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 369										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																S	L	Reserved						MsgLen							
MsgLen								Message																							
Message																															

S – 0: Store reply, 1: Send reply
L – 0: Omit length from reply, 1: Include length in reply
CSI – Crypto suite identifier
MsgLen: Length of message
Message: Crypto message
See section 12.2.1.3.2

C1G2ReadBuffer Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 370										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																WordPtr															
NumBits																															

WordPtr: Word pointer address
NumBits: Number of bits to read starting from WordPtr
See section 14.2.9.10.10

C1G2KeyUpdate Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 372										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																S	L	Reserved						KeyID							
MsgLen																Message															
Message																															

S – 0: Store reply, 1: Send reply
L – 0: Omit length from reply, 1: Include length in reply
KeyID – Key identifier
MsgLen: Length of message
Message: Message dependent on CSI
See section 12.2.1

C1G2TagPrivilege Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 373										Length															
OpSpecID																AccessPassword[31:16]															
AccessPassword[15:0]																S	L	A	T	Reserved				KeyID							
Privilege																															

S – 0: Store reply, 1: Send reply
L – 0: Omit length from reply, 1: Include length in reply
A – 0: Read action, 1: Modify action
T – 0: Targets access password, 1: Targets key
KeyID – Key identifier
Privilege: Privilege bit mask. Refer TagPrivilege command in Class 1 Gen 2 Standard
See section 13.2.6.3.40

16.3.1.4 Configuration Parameters

This section of air protocol specific parameters corresponds to LLRP parameters specified in section 13.2 The only air protocol specific parameter is the AirProtocolInventoryCommandSettings parameter in the AntennaConfiguration (section 13.2.6).

16.3.1.5 Reporting Parameters

This section of air protocol specific parameters corresponds to LLRP parameters encodings specified in section 14.2.

16.3.1.5.1 C1G2EPCMemorySelector Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 348										Length															
C	P	X	Reserved																												

Abbreviations

C – EnableCRC
P – EnablePCBits
X – EnableXPCBits

See section 14.2.1.2

16.3.1.5.2 C1G2PC Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=12							PC-Bits																							

See section 14.2.3.14

16.3.1.5.3 C1G2XPCW1 Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=19							XPC_W1																							

See section 14.2.3.15

16.3.1.5.4 C1G2XPCW2 Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=20							XPC_W2																							

See section 14.2.3.16

16.3.1.5.5 C1G2CRC Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=11							CRC																							

See section 14.2.3.17

16.3.1.5.6 C1G2SingulationDetails Parameter (TV-Encoding)

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
1	Type=18							NumCollisionSlots																NumEmptySlots[15:8]							
NumEmptySlots[7:0]																															

See section 14.2.9.9.1

16.3.1.5.7 C1G2 OpSpec Results

C1G2ReadOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 349										Length															
Result								OpSpecID																ReadDataWordCount[15:8]							
ReadDataWordCount[7:0]																															
ReadData																															

See section 14.2.9.10.1

C1G2WriteOpSpecResult Parameter

0										1										2										3	

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 350										Length															
Result								OpSpecID																NumWordsWritten[15:8]							
NumWordsWritten[7:0]																															

See section 14.2.9.10.2

C1G2KillOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 351										Length															
Result								OpSpecID																							

See section 14.2.9.10.3

C1G2LockOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 352										Length															
Result								OpSpecID																							

See section 14.2.9.10.4

C1G2BlockEraseOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 353										Length															
Result								OpSpecID																							

See section 14.2.9.10.5

C1G2BlockWriteOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 354										Length															
Result								OpSpecID																NumWordsWritten[15:8]							
NumWordsWritten[7:0]																															

See section 14.2.9.10.6

C1G2BlockPermalockOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 361										Length															
Result								OpSpecID																							

See section 14.2.9.10.7

C1G2GetBlockPermalockStatusOpSpecResult Parameter

0										1										2										3	

0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 362										Length															
Result								OpSpecID																StatusWordCount[15:8]							
StatusWordCount[7:0]								PermalockStatus																							

See section 14.2.9.10.8

C1G2UntraceableOpSpecResult Parameter

0										1										2										3	
0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1	2	3	4	5	6	7	8	9	0	1
Reserved						Type = 364										Length															
Result								OpSpecID																							

See section 14.2.9.10.9

16.4 Listing of Message and Parameter Types

This section lists the parameter and message types used in the binary encoding.

Table 5: Message Listing

Message Name	Type
GET_SUPPORTED_VERSION	46
GET_SUPPORTED_VERSION_RESPONSE	56
SET_PROTOCOL_VERSION	47
SET_PROTOCOL_VERSION_RESPONSE	57
GET_READER_CAPABILITIES	1
GET_READER_CAPABILITIES_RESPONSE	11
ADD_ROSPEC	20
ADD_ROSPEC_RESPONSE	30
DELETE_ROSPEC	21
DELETE_ROSPEC_RESPONSE	31
START_ROSPEC	22
START_ROSPEC_RESPONSE	32
STOP_ROSPEC	23
STOP_ROSPEC_RESPONSE	33
ENABLE_ROSPEC	24
ENABLE_ROSPEC_RESPONSE	34
DISABLE_ROSPEC	25
DISABLE_ROSPEC_RESPONSE	35
GET_ROSPECS	26
GET_ROSPECS_RESPONSE	36
ADD_ACCESSSPEC	40

ADD_ACCESSSPEC_RESPONSE	50
DELETE_ACCESSSPEC	41
DELETE_ACCESSSPEC_RESPONSE	51
ENABLE_ACCESSSPEC	42
ENABLE_ACCESSSPEC_RESPONSE	52
DISABLE_ACCESSSPEC	43
DISABLE_ACCESSSPEC_RESPONSE	53
GET_ACCESSSPECS	44
GET_ACCESSSPECS_RESPONSE	54
CLIENT_REQUEST_OP	45
CLIENT_REQUEST_OP_RESPONSE	55
GET_REPORT	60
RO_ACCESS_REPORT	61
KEEPALIVE	62
KEEPALIVE_ACK	72
READER_EVENT_NOTIFICATION	63
ENABLE_EVENTS_AND_REPORTS	64
ERROR_MESSAGE	100
GET_READER_CONFIG	2
GET_READER_CONFIG_RESPONSE	12
SET_READER_CONFIG	3
SET_READER_CONFIG_RESPONSE	13
CLOSE_CONNECTION	14
CLOSE_CONNECTION_RESPONSE	4
CUSTOM_MESSAGE	1023
Reserved for ISO/IEC 24791-5	900-999

Table 6: Parameter Listing

Parameter Name	Type	TV-Encoded?
UTCTimeStamp	128	
Uptime	129	
GeneralDeviceCapabilities	137	
MaximumReceiveSensitivity	363	
ReceiveSensitivityTableEntry	139	
PerAntennaAirProtocol	140	
GPIOCapabilities	141	
LLRPCapabilities	142	
RegulatoryCapabilities	143	

UHFBandCapabilities	144	
TransmitPowerLevelTableEntry	145	
FrequencyInformation	146	
FrequencyHopTable	147	
FixedFrequencyTable	148	
PerAntennaReceiveSensitivityRange	149	
RFSurveyFrequencyCapabilities	365	
ROSpec	177	
ROBoundarySpec	178	
ROSpecStartTrigger	179	
PeriodicTriggerValue	180	
GPITriggerValue	181	
ROSpecStopTrigger	182	
AISpec	183	
AISpecStopTrigger	184	
TagObservationTrigger	185	
InventoryParameterSpec	186	
RFSurveySpec	187	
RFSurveySpecStopTrigger	188	
LoopSpec	355	
AccessSpec	207	
AccessSpecStopTrigger	208	
AccessCommand	209	
ClientRequestOpSpec	210	
ClientRequestResponse	211	
LLRPConfigurationStateValue	217	
Identification	218	
GPOWriteData	219	
KeepaliveSpec	220	
AntennaProperties	221	
AntennaConfiguration	222	
RFReceiver	223	
RFTransmitter	224	
GPIPortCurrentState	225	
EventsAndReports	226	
ROReportSpec	237	
TagReportContentSelector	238	

AccessReportSpec	239	
TagReportData	240	
EPCData	241	
EPC-96	13	X
ROSpecID	9	X
SpecIndex	14	X
InventoryParameterSpecID	10	X
AntennaID	1	X
PeakRSSI	6	X
ChannelIndex	7	X
FirstSeenTimestampUTC	2	X
FirstSeenTimestampUptime	3	X
LastSeenTimestampUTC	4	X
LastSeenTimestampUptime	5	X
TagSeenCount	8	X
ClientRequestOpSpecResult	15	X
AccessSpecID	16	X
RFSurveyReportData	242	
FrequencyRSSILevelEntry	243	
ReaderEventNotificationSpec	244	
EventNotificationState	245	
ReaderEventNotificationData	246	
HoppingEvent	247	
GPIEvent	248	
ROSpecEvent	249	
ReportBufferLevelWarningEvent	250	
ReportBufferOverflowErrorEvent	251	
ReaderExceptionEvent	252	
OpSpecID	17	X
RFSurveyEvent	253	
AISpecEvent	254	
AntennaEvent	255	
ConnectionAttemptEvent	256	
ConnectionCloseEvent	257	
SpecLoopEvent	356	
LLRPStatus	287	
FieldError	288	

ParameterError	289	
CryptoResponse	290	
Custom	1023	
C1G2LLRPCapabilities	327	
UHFC1G2RFModeTable	328	
UHFC1G2RFModeTableEntry	329	
C1G2InventoryCommand	330	
C1G2Filter	331	
C1G2TagInventoryMask	332	
C1G2TagInventoryStateAwareFilterAction	333	
C1G2TagInventoryStateUnawareFilterAction	334	
C1G2RFControl	335	
C1G2SingulationControl	336	
C1G2TagInventoryStateAwareSingulationAction	337	
C1G2TagSpec	338	
C1G2TargetTag	339	
C1G2Read	341	
C1G2Write	342	
C1G2Kill	343	
Reserved	357	
C1G2Lock	344	
C1G2LockPayload	345	
C1G2BlockErase	346	
C1G2BlockWrite	347	
C1G2BlockPermalock	358	
C1G2GetBlockPermalockStatus	359	
C1G2EPCMemorySelector	348	
C1G2PC	12	X
C1G2XPCW1	19	X
C1G2XPCW2	20	X
C1G2CRC	11	X
C1G2SingulationDetails	18	X
C1G2ReadOpSpecResult	349	
C1G2WriteOpSpecResult	350	
C1G2KillOpSpecResult	351	
Reserved	360	
C1G2LockOpSpecResult	352	

C1G2BlockEraseOpSpecResult	353	
C1G2Challenge	366	
C1G2BlockWriteOpSpecResult	354	
C1G2BlockPermalockOpSpecResult	361	
C1G2GetBlockPermalockStatusOpSpecResult	362	
C1G2Untraceable	380	
C1G2UntraceableOpSpecResult	364	
C1G2Authenticate	367	
C1G2AuthComm	368	
C1G2SecureComm	369	
C1G2ReadBuffer	370	
C1G2KeyUpdate	372	
C1G2TagPrivilege	373	
C1G2AuthenticateOpSpecResult	374	
C1G2AuthCommOpSpecResult	375	
C1G2SecureCommOpSpecResult	376	
C1G2ReadBufferOpSpecResult	377	
C1G2KeyUpdateOpSpecResult	378	
C1G2TagPrivilegeOpSpecResult	379	
ExtendOnTime	381	
Reserved for ISO/IEC 24791-5	900-999	

17 Transmitter Behavior of a Reader

A Reader SHALL enable its transmitter only under the following conditions:

- When an ROSpec is in the active state.
- Between a GET/SET_READER_CONFIG containing a RequestedData field with value 0 (All) or 2 (Antenna Properties) and the corresponding GET/SET_READER_CONFIG_RESPONSE.

18 Future Versions of LLRP

To ensure continued viability of the protocol, all backwards compatible versions of LLRP shall be implemented with the following restrictions.

- In all future versions of LLRP, the following SHALL be prohibited:
    - New mandatory parameters in existing messages or parameters
    - New fields in existing messages or parameters
    - Changes to any existing field in messages or parameters
    - Changes to any messages involved in version negotiation
- In all future versions of LLRP, new functionality SHALL be implemented by one of the following:
    - New optional parameters in existing messages or parameters

- New extension points in existing messages or parameters.
- Extending existing enumerated types
- Using existing reserved bits
- Adding new messages
- Using custom extensions

19 Connection and Transport

The Reader SHALL maintain LLRP configuration state during an LLRP connection.

The Reader MAY maintain configuration or data state when a connection fails, or across LLRP connections.

19.1 TCP Transport

LLRP end-to-end communications based on TCP/IP connections SHALL be implemented in accordance with the requirements specified in this section. These requirements are defined as the LLRP *TCP Transport*.

19.1.1 Connection Establishment

Readers SHALL be able to both initiate and accept LLRP TCP connections. Readers MAY be configured such that, at any given time, they only either initiate or accept an LLRP connection. If so, the mechanism for configuring a Reader to either initiate or accept an LLRP connection is not specified by LLRP.

Clients SHALL be able both to initiate and accept LLRP TCP connections. Clients MAY be configured such that, at any given time, they only either initiate or accept an LLRP connection. If so, the mechanism for configuring a Client to either initiate or accept an LLRP connection is not specified by LLRP.

For Readers and Clients, that are configured to accept connections, the default port is 5084, as established by IANA (see http://www.iana.org/assignments/port-numbers), but other ports can be used.

When a TCP connection (called the *established connection*) is initiated by either the Reader or the Client, the Reader SHALL reply with a status report message before communicating any other information. This report's status parameter, ConnectionAttemptEvent, SHALL be set to indicate connection success (see section *14.2.9.12*). No other parameters may be contained within this message. The Client SHALL not send any information to the Reader until this status report message is received.

19.1.2 Duplicate Connection Management

Readers SHALL limit communications to a single established connection on a Reader IP address and TCP port. Readers MAY momentarily accept TCP connections (called *momentary connections*) in addition to the Reader's one established connection on a Reader IP address and TCP port. If a momentary connection is accepted, then the Reader SHALL send a status report message on the Reader's established connection. This report's status parameter, ConnectionAttemptEvent, SHALL be set to indicate that another connection was attempted (see section *14.2.9.12*). If this action results in a TCP error, then the Reader MAY close the established connection and then treat the momentary connection as a new established connection. In this case, the Reader SHALL

reply with a status report message on the newly created established connection, as specified above, indicating connection success.

If the established connection is not closed, then the Reader SHALL reply on the momentary connection with a status report message. This report's status parameter, ConnectionAttemptEvent, SHALL be set to indicate connection failure. The Reader SHALL use the appropriate connection failed status value as defined in section 14.2.9.12. Once the connection failure message is sent, the Reader SHALL close the momentary connection.

The following UML sequence diagrams illustrate different scenarios of a Reader and Client initiating TCP connections.

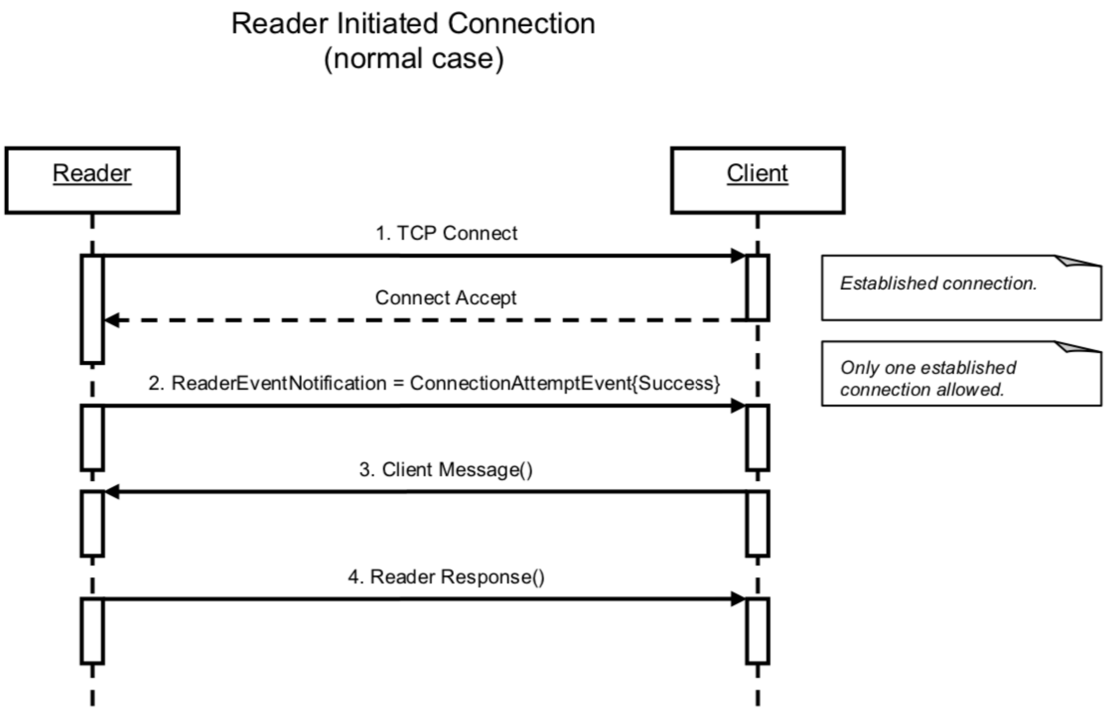
Reader Initiated Connection
(normal case)
Reader
Client
1. TCP Connect
Connect Accept
Established connection.
Only one established connection allowed.
2. ReaderEventNotification = ConnectionAttemptEvent{Success}
3. Client Message()
4. Reader Response()

Figure 16: Reader Initiated Connection (Normal)

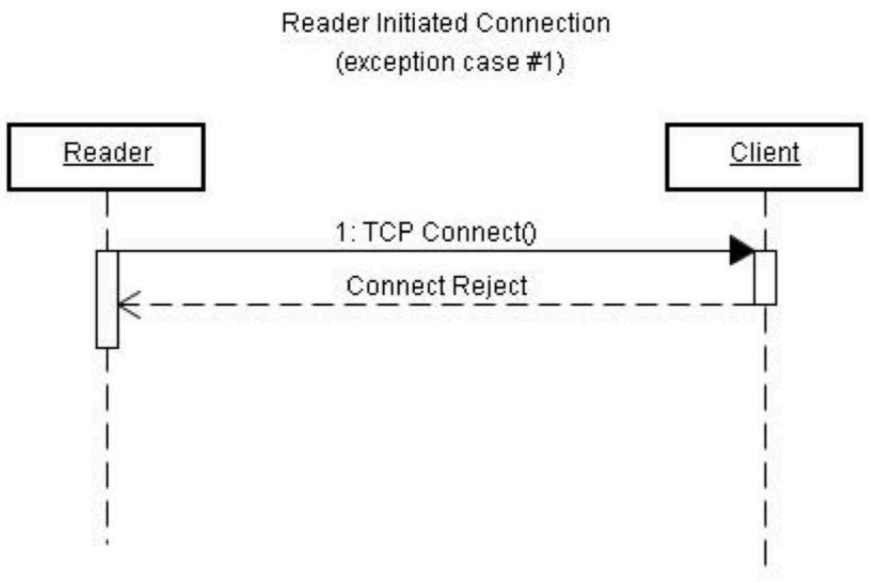

Figure 17: Reader Initiated Connection (Exception)

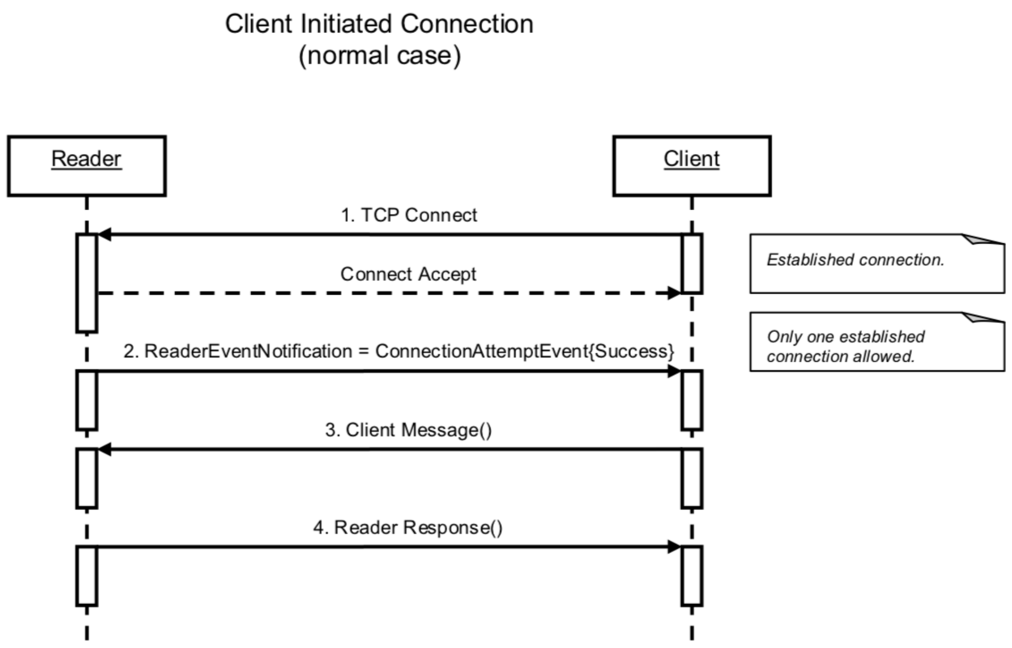
Client Initiated Connection
(normal case)
Reader
Client
1. TCP Connect
Connect Accept
2. ReaderEventNotification = ConnectionAttemptEvent{Success}
3. Client Message()
4. Reader Response()
Established connection.
Only one established connection allowed.

Figure 18: Client Initiated Connection (Normal)

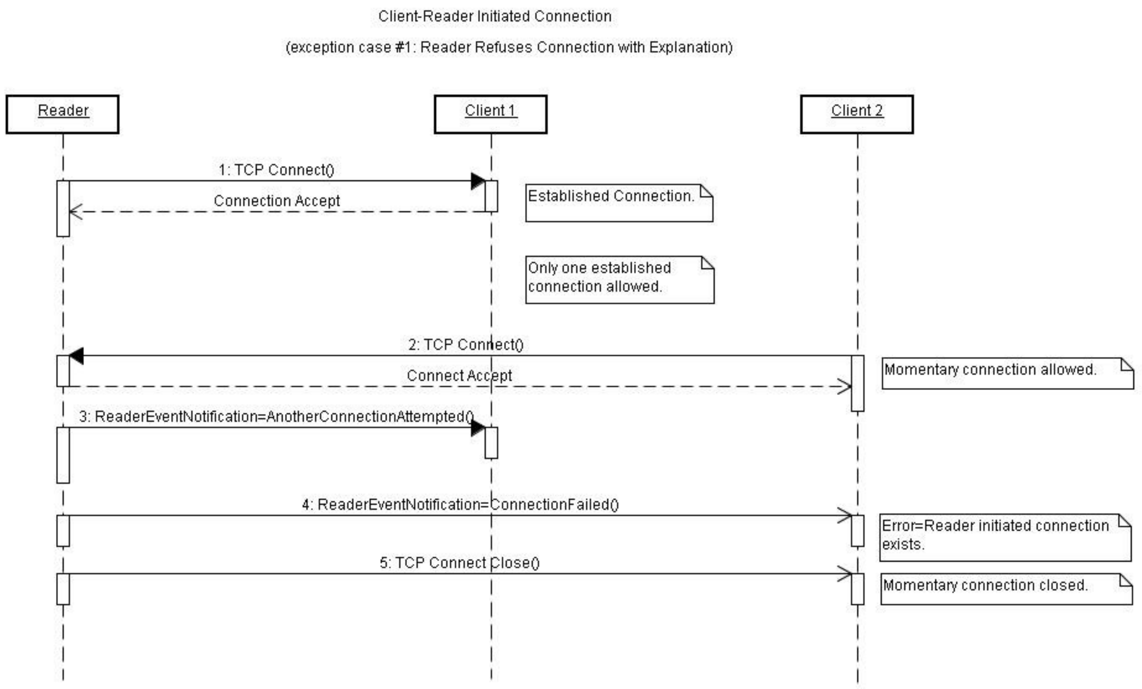
Client-Reader Initiated Connection
(exception case #1: Reader Refuses Connection with Explanation)
Reader
Client 1
Client 2
1: TCP Connect()
Connection Accept
Established Connection.
Only one established connection allowed.
2: TCP Connect()
Connect Accept
Momentary connection allowed.
3: ReaderEventNotification=AnotherConnectionAttempted()
4: ReaderEventNotification=ConnectionFailed()
Error=Reader initiated connection exists.
5: TCP Connect Close()
Momentary connection closed.

Figure 19: Client Initiated Connection (Exception #1)

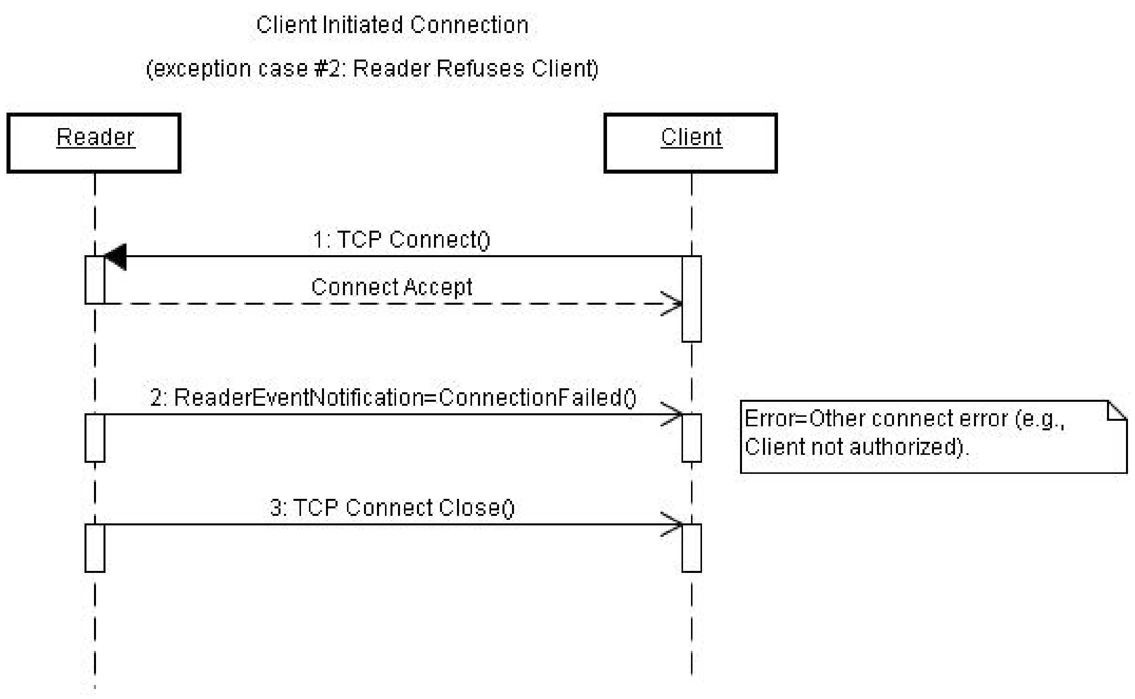
Client Initiated Connection
(exception case #2: Reader Refuses Client)
Reader
Client
1: TCP Connect()
Connect Accept
2: ReaderEventNotification=ConnectionFailed()
Error=Other connect error (e.g., Client not authorized).
3: TCP Connect Close()

Figure 20: Client Initiated Connection (exception #2)

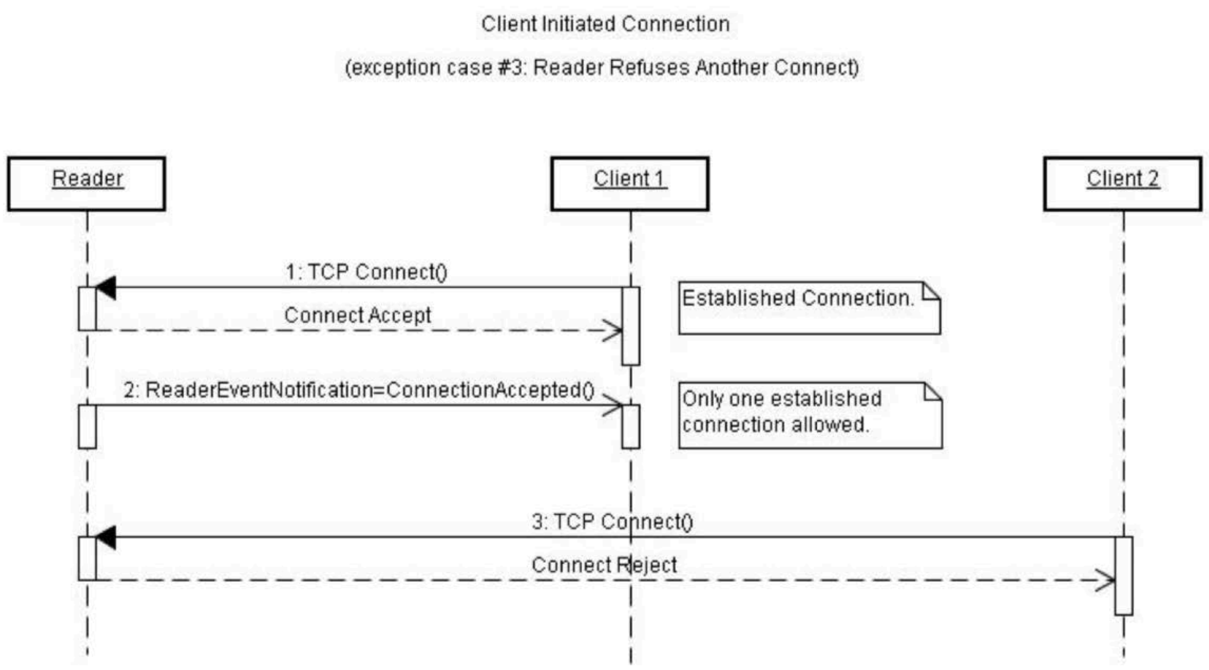
Client Initiated Connection
(exception case #3: Reader Refuses Another Connect)
Reader
Client 1
Client 2
1: TCP Connect()
Connect Accept
Established Connection.
2: ReaderEventNotification=ConnectionAccepted()
Only one established connection allowed.
3: TCP Connect()
Connect Reject

Figure 21: Client Initiated Connection (exception #3)

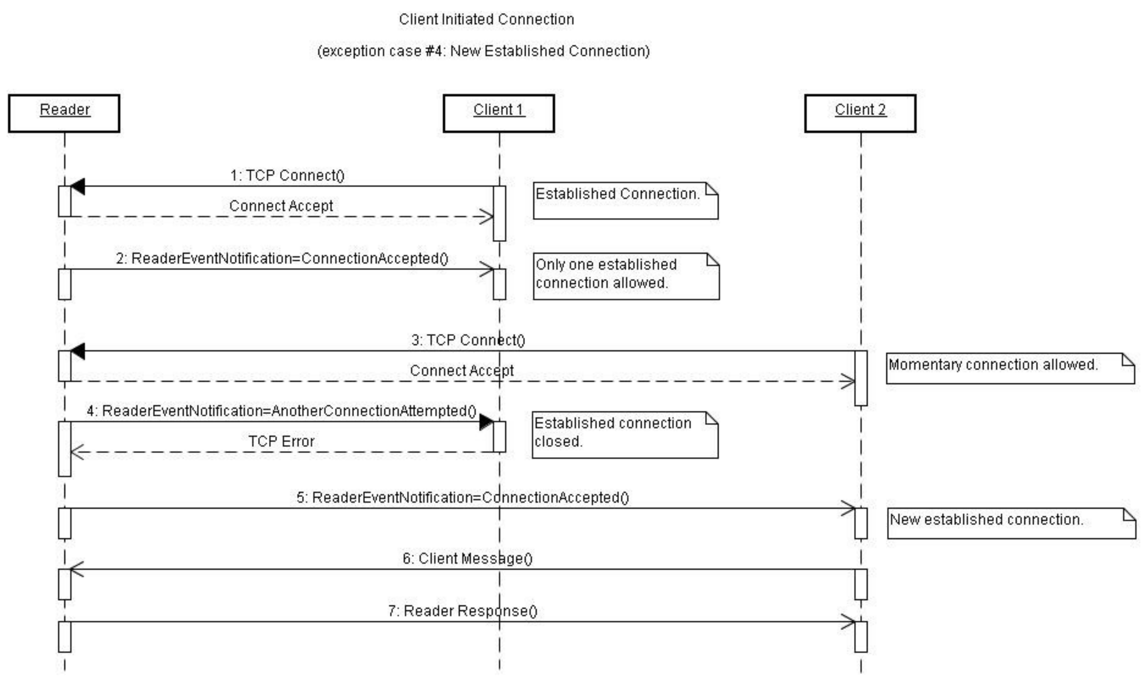
Client Initiated Connection
(exception case #4: New Established Connection)
Reader
Client 1
Client 2
1: TCP Connect()
Connect Accept
Established Connection.
2: ReaderEventNotification=ConnectionAccepted()
Only one established connection allowed.
3: TCP Connect()
Connect Accept
Momentary connection allowed.
4: ReaderEventNotification=AnotherConnectionAttempted()
TCP Error
Established connection closed.
5: ReaderEventNotification=ConnectionAccepted()
New established connection.
6: Client Message()
7: Reader Response()

Figure 22: Client Initiated Connection (exception #4)

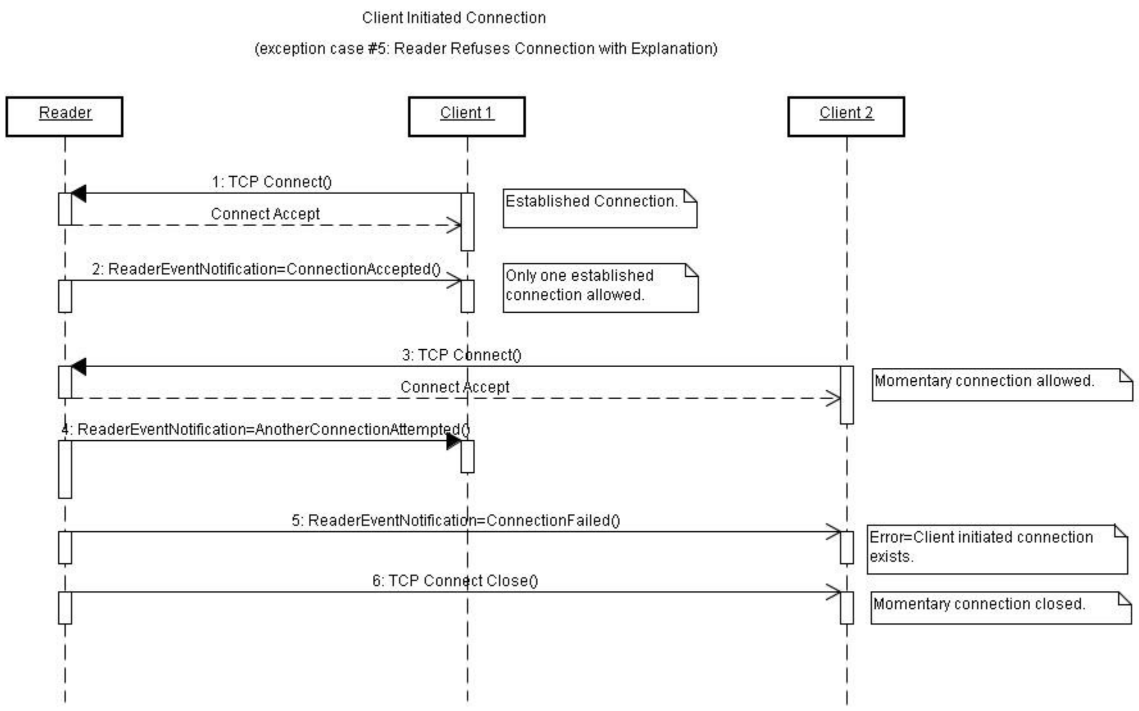
Client Initiated Connection
(exception case #5: Reader Refuses Connection with Explanation)
Reader
Client 1
Client 2
1: TCP Connect()
Connect Accept
Established Connection.
2: ReaderEventNotification=ConnectionAccepted()
Only one established connection allowed.
3: TCP Connect()
Connect Accept
Momentary connection allowed.
4: ReaderEventNotification=AnotherConnectionAttempted()
5: ReaderEventNotification=ConnectionFailed()
Error=Client initiated connection exists.
6: TCP Connect Close()
Momentary connection closed.

Figure 23: Client Initiated Connection (exception #5)

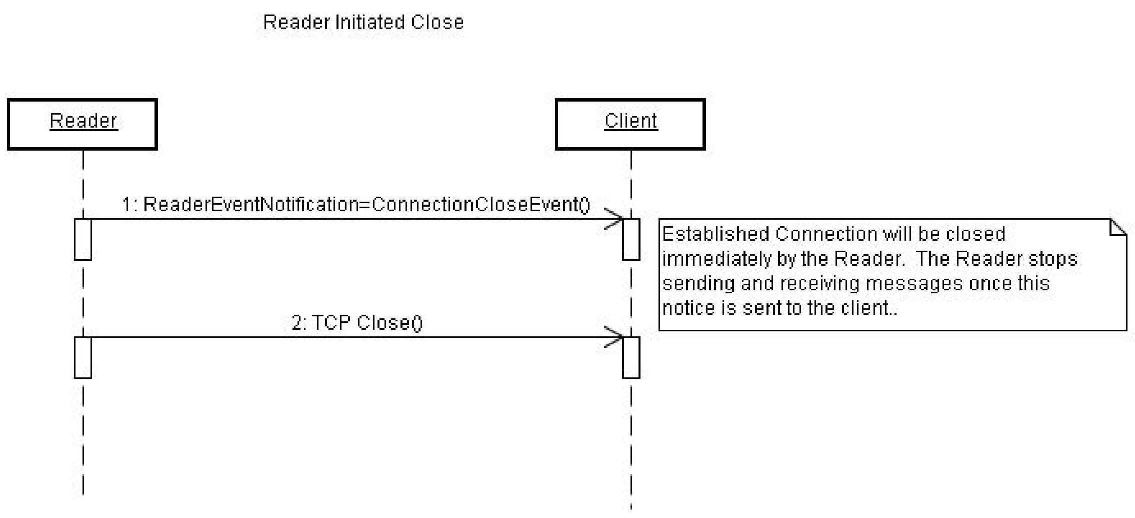
Reader Initiated Close
Reader
Client
1: ReaderEventNotification=ConnectionCloseEvent()
Established Connection will be closed immediately by the Reader. The Reader stops sending and receiving messages once this notice is sent to the client..
2: TCP Close()

Figure 24: Reader Initiated Close

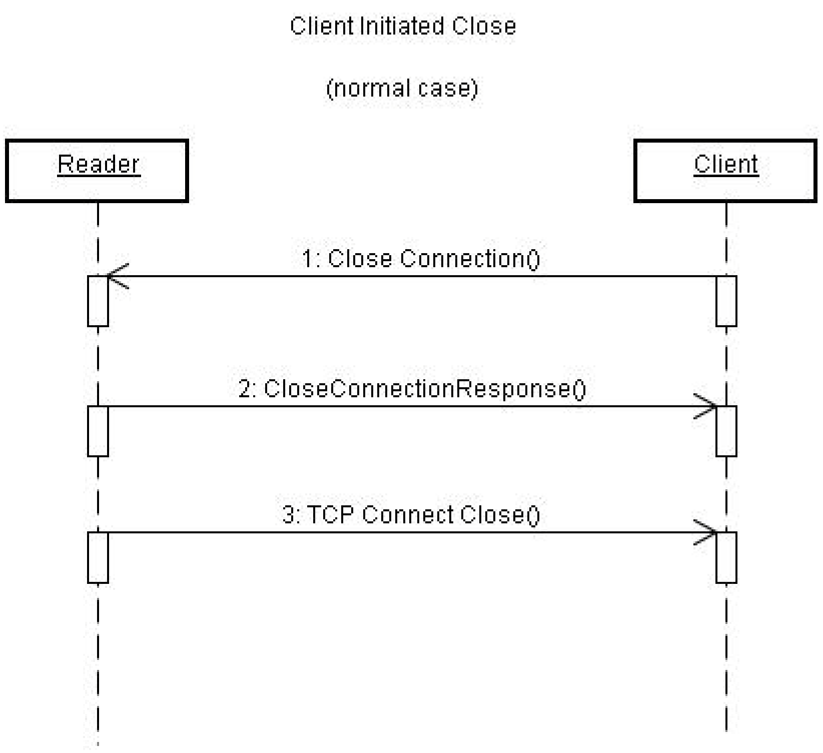

Figure 25: Client Initiated Close (Normal)

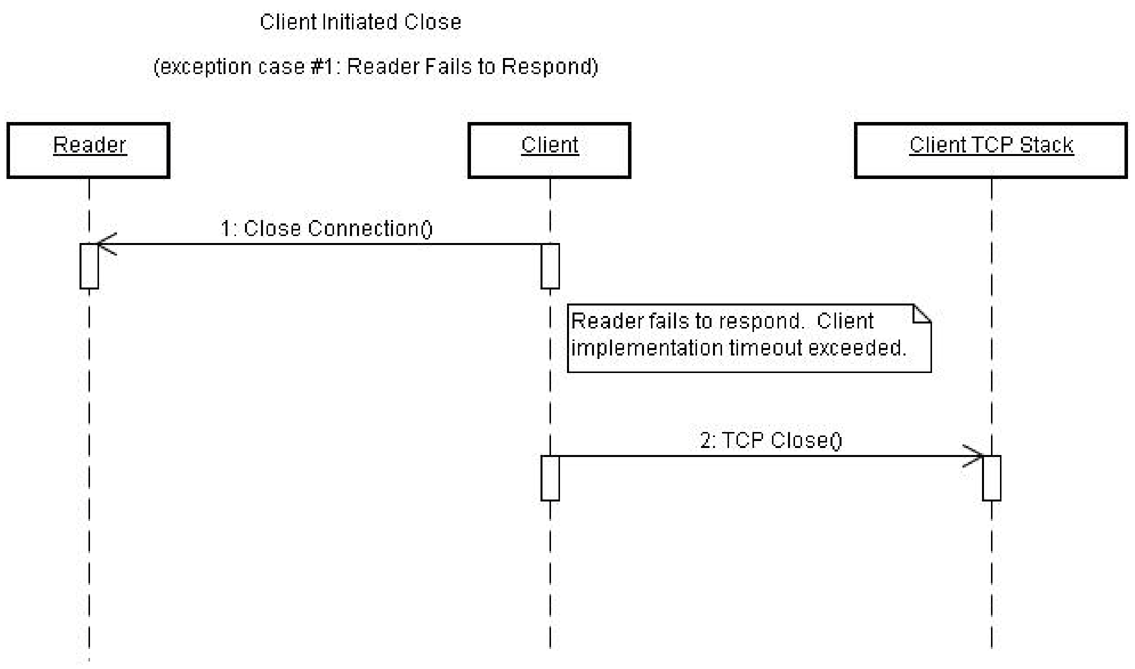
Client Initiated Close
(exception case #1: Reader Fails to Respond)
Reader
Client
Client TCP Stack
1: Close Connection()
Reader fails to respond. Client implementation timeout exceeded.
2: TCP Close()

Figure 26: Client Initiated Close (Exception)

19.1.3 Version Negotiation

LLRP version negotiation consists of the Client discovering the Reader's supported protocol versions, selecting an appropriate, mutually supported version, and setting the version. Once selected, this version SHALL remain constant for the duration of the connection; renegotiation is not supported. For details regarding this process, see section 9.

Because version negotiation was introduced in LLRP 1.1, there exist scenarios where mismatched Client and Reader implementations must still successfully negotiate a protocol version. The following UML sequence diagrams illustrate different scenarios of a Reader and Client negotiating a protocol version.

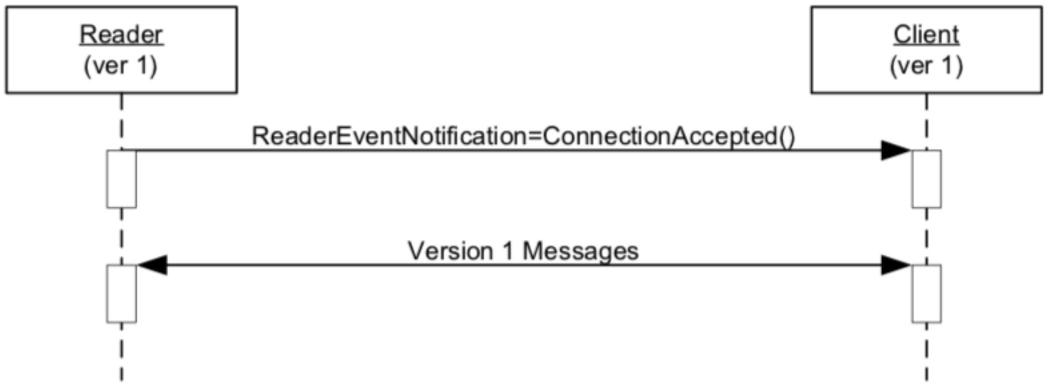

Figure 27: Reader version 1, Client version 1

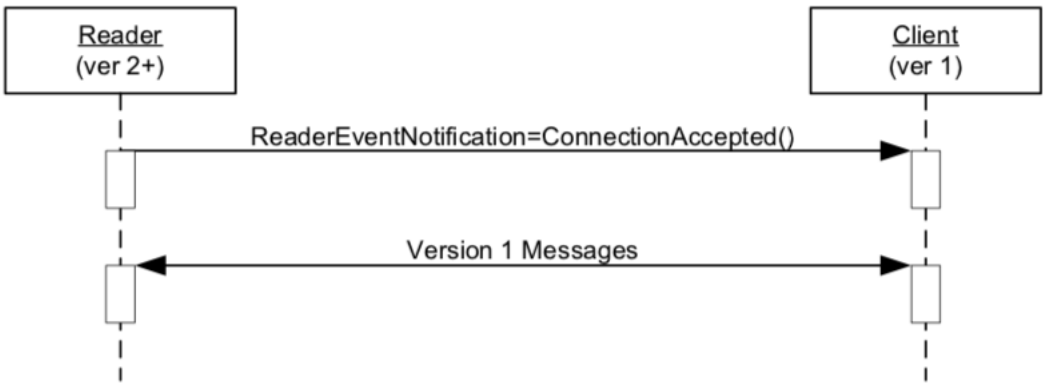

Figure 28: Reader version 2+, Client version 1

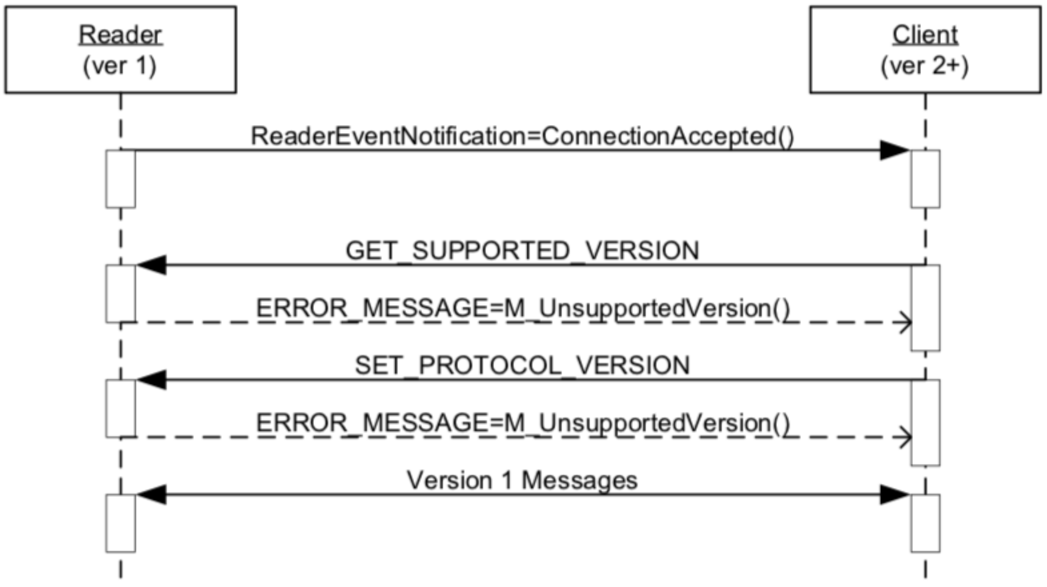
Reader
(ver 1)
Client
(ver 2+)
ReaderEventNotification=ConnectionAccepted()
GET_SUPPORTED_VERSION
ERROR_MESSAGE=M_UnsupportedVersion()
SET_PROTOCOL_VERSION
ERROR_MESSAGE=M_UnsupportedVersion()
Version 1 Messages

Figure 29: Reader version 1, Client version 2+

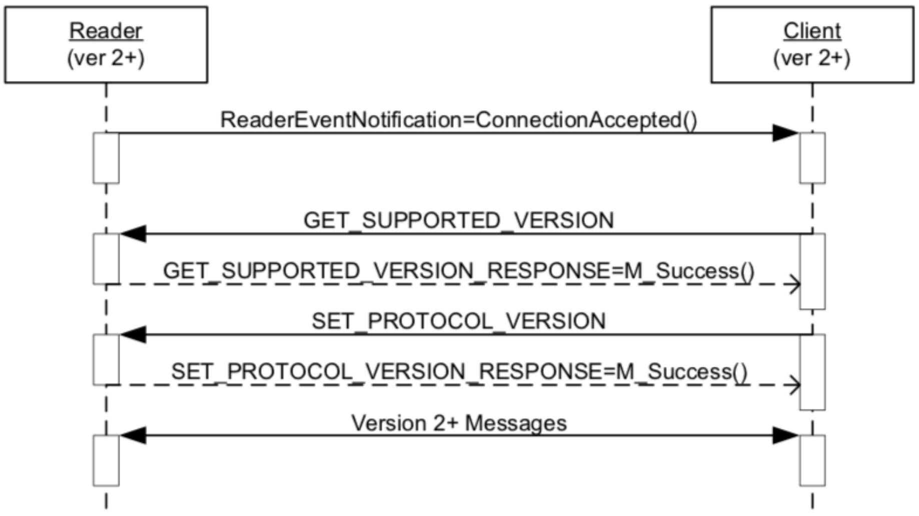
Reader
(ver 2+)
Client
(ver 2+)
ReaderEventNotification=ConnectionAccepted()
GET_SUPPORTED_VERSION
GET_SUPPORTED_VERSION_RESPONSE=M_Success()
SET_PROTOCOL_VERSION
SET_PROTOCOL_VERSION_RESPONSE=M_Success()
Version 2+ Messages

Figure 30: Reader version 2+, Client version 2+

Version Negotiation can only occur once per LLRP connection (see section 9.1.3). The following UML diagram illustrates two separate LLRP connections based on the TCP transport, and how version negotiation occurs.

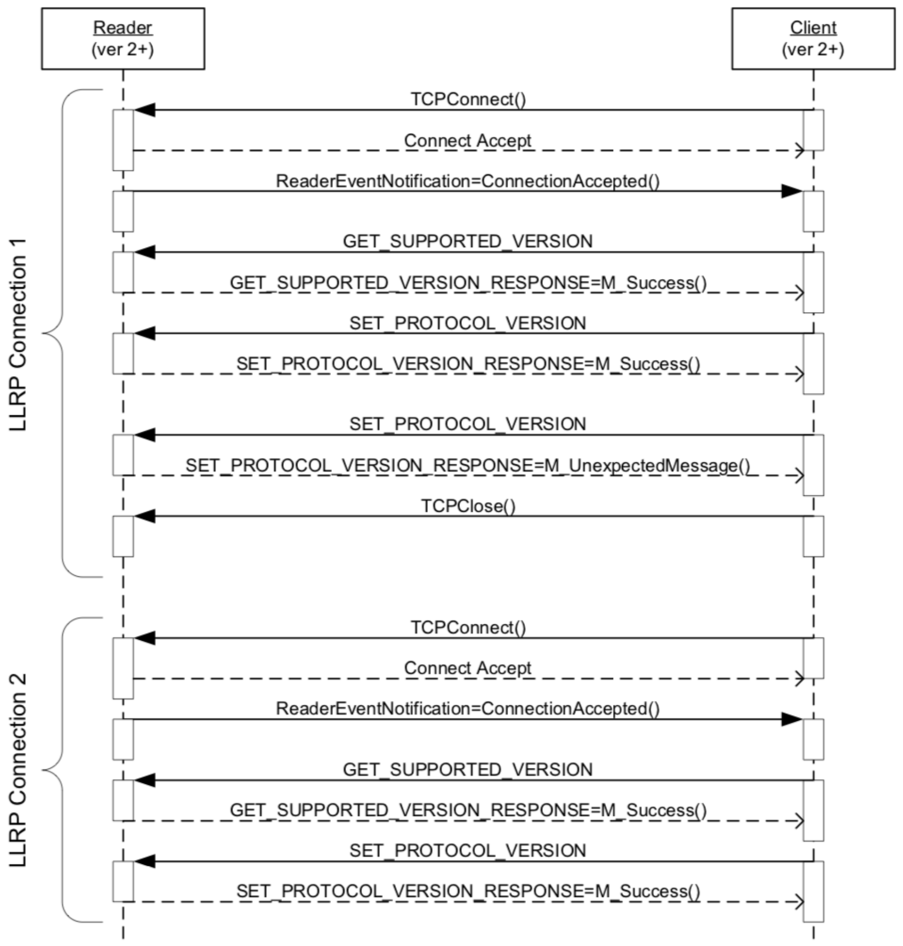
Reader (ver 2+)
Client (ver 2+)
LLRP Connection 1
TCPConnect()
Connect Accept
ReaderEventNotification=ConnectionAccepted()
GET_SUPPORTED_VERSION
GET_SUPPORTED_VERSION_RESPONSE=M_Success()
SET_PROTOCOL_VERSION
SET_PROTOCOL_VERSION_RESPONSE=M_Success()
SET_PROTOCOL_VERSION
SET_PROTOCOL_VERSION_RESPONSE=M_UnexpectedMessage()
TCPClose()
LLRP Connection 2
TCPConnect()
Connect Accept
ReaderEventNotification=ConnectionAccepted()
GET_SUPPORTED_VERSION
GET_SUPPORTED_VERSION_RESPONSE=M_Success()
SET_PROTOCOL_VERSION
SET_PROTOCOL_VERSION_RESPONSE=M_Success()

Figure 31: Version Negotiation across LLRP connections

19.2 Security in TCP Transport

This section describes the security aspects for LLRP connections running over a TCP transport binding. Refer to the previous section for any TCP connection related requirements.

19.2.1 Normative section

The LLRP Client and LLRP Reader MAY implement TLS. The LLRP Client and LLRP Reader MAY use a different port for TLS LLRP connections and non-TLS LLRP connections.

The LLRP Client MAY be capable of operating in a mixed deployment, where it communicates using TLS with a set of Readers and just plain TCP with a different set of Readers. In such mixed deployments, the LLRP Client MAY use different ports for TLS and non-TLS LLRP connections. The default port for TLS-LLRP connections is 5085, as established by IANA (see http://www.iana.org/assignments/port-numbers), but other ports can be used.

The LLRP endpoint that initiates the TLS connection MAY be the same LLRP endpoint that initiated the underlying TCP connection.

The LLRP endpoints SHALL use at least TLS 1.3 [TLS13].

If the Reader or Client uses X.509 certificates[X509] for authentication, the certificates SHALL be compliant with the EPCglobal Security2 working group specification [SEC2].

19.2.2 Informative Section

19.2.2.1 Overview of TLS

The TLS protocol provides privacy and data integrity between two authenticated communicating applications. TLS is a light weight transport protocol and has been proven to be reliable and secure by the use of millions of real users for many years. The strength of TLS can be chosen by the cipher suite negotiated by the two communicating parties through a flexible mechanism during the handshaking.

TLS is particularly useful for TCP based applications. First, a TLS client initiates a connection with the TLS server. After a TLS connection is established, the applications can use the transport connection like an ordinary TCP connection, while having the added value that the data is protected and that both parties are mutually authenticated.

For interoperability, a TLS client and server have to implement at least one common cipher suite. The credentials required for mutual authentication depend on the suite negotiated. For example, if the negotiated suite is using RSA for key exchange, then the server must own a server certificate (with private key) for RSA encryption purposes while the client must have a client certificate (with private key) for RSA signing purposes. Further, each side must have the root Certificate Authority (CA) certificates to verify the certificates presented by the peers. TLS also requires each party to present the CA certificates (except the root) that directly and indirectly issue the certificate.

19.2.2.2 Threat Analysis for LLRP

With TLS being used for Reader and Client communication, the following protections are provided, assuming that the credentials for the TLS client and server are not stolen:

- Readers only talk to authorised LLRP Clients;
- LLRP Clients only talk to authorised Readers;
- No other party can read the LLRP messages (privacy protection) or inject/modify messages without being detected (integrity protection).

Note that the strength of protection depends on the negotiated cipher suite.

19.2.2.3 Configuration Elements for TLS

In order to use TLS for LLRP, the following information has to be configured and/or provisioned at each entity (Reader or Client):

- **TLS enabled:** Yes or no. If TLS is not enabled, the rest of the information need not be configured and the LLRP endpoint (Reader or Client) SHALL use TCP directly.
- **TLS role:** Whether the LLRP endpoint is playing the TLS client or the TLS server role. A TLS client initiates a TCP connection to jump start TLS handshaking. A TLS server passively listens on the TCP server port.
- **Preferred list of cipher suites**: A TLS client proposes the list of cipher suites to the TLS server during TLS handshaking. The TLS server will pick one suite from the proposed list if it is also in the preferred list maintained by the server. In TLS, the order of suites in the proposed list has no significance. Also, it is up to the server's local policy to select when there are multiple choices.
- **Certificates and private keys**: A TLS server needs a server certificate (with private key) for TLS server authentication. A TLS client needs a client certificate (with private key) for TLS client authentication. In each case, all the CA certificates (except the root) in the chain have to be available.
- **Root CA certificates**: A TLS server needs to maintain the root CA certificate of the client certificate. This is used for verifying client certificates. A TLS client needs to maintain the root CA certificate of the server certificate. This is used for verifying server certificates.

- **List of authorised devices:** Each TLS server MAY have a list of authorised TLS clients that can connect to it. Likewise, each TLS client MAY have a list of authorised TLS servers that it can connect to.

The configuration and/or provisioning of a LLRP endpoint is out of the scope of TLS and LLRP. Provisioning is important but does not affect the interoperability of LLRP. Vendors should have the flexibility to choose the most cost-effective ways (for provisioning and protecting provisioned credentials) based on designs, available technologies, potential threats, security requirements, and so on. This is a topic that should be addressed in DCI.

19.2.2.4 Why different TLS server port?

It is recommended that the TLS server should listen to a TCP port different from that for non-TLS mode for the following reasons:

- If one of the endpoints has to be deployed behind firewalls, IT managers are more willing to open a port they know only TLS traffic can pass through.
- Without using a different port, a non-TLS server may be confused by the TLS Client-Hello handshaking message.
- Without using a different port, a TLS server may be confused by the LLRP application message (non-TLS handshaking message).
- Without using a different port, for each new TCP connection, a server in a mixed environment (TLS and non-TLS) may have to wait a few moments to see if a Client-Hello message ever arrives from the client before it can conclude whether it is a TLS connection or not.
- Without using a different port, it is potentially harder to implement a hybrid server if the server relies on third-party libraries for handling TLS. This is because the server application has to read the first message from the client to know if it is a TLS connection. It may be difficult for the TLS library to take over a connection after the TLS Client-Hello message has been consumed.

However, if a deployment in totality is only TLS or only non-TLS, the LLRP endpoint can be configured only as a TLS server or non-TLS server exclusively, then there should be no problem using the same port, as long as a non-TLS server can ignore TLS handshaking messages from a TLS client and as long as a TLS server can ignore non- TLS handshaking messages from a non-TLS client.

A (Informative) TCP Keepalives

The TCP specification doesn’t specify any specific handling of idle connections, where there is no data being transmitted by either end for a prolonged period of time. However, in some TCP implementations, there is an option called TCP-keepalive which may be turned on. If turned on, TCP-keepalive packets are sent only during periods of inactivity, on a configurable interval. If the connection is still valid, the other end responds with a segment containing an ack. If the connection is not valid the other end will reply with a connection reset (RST) and the connection is closed by this end.

Due to events like network failures, or Client failures, half connections may remain at the Reader because the TCP connection was not cleanly terminated. If the Reader doesn’t implement TCP-keepalive, the only way to recover (i.e., reconnect to the Reader) may be to reboot the Reader.

However, there are Readers for which intermittent connectivity may be a normal mode of operation – e.g., mobile Readers, handheld Readers. When connectivity is lost for these devices, the use of TCP-keepalive acts negatively and closes the TCP session prematurely before the TCP session would have timed out. If keepalives were not used, the mobile Reader would just start sending LLRP messages as soon as the link layer is re-established without requiring a re-establishment of the TCP session as long as the TCP session did not timeout.

B (Informative) References

[ISODir2] ISO, “Rules for the structure and drafting of International Standards (ISO/IEC Directives, Part 2, 2001, 4th edition),” July 2002.

[C1G2] Class 1 Generation 2 UHF Air Interface Protocol Standard

https://www.gs1.org/standards/epc-rfid/uhf-air-interface-protocol

[ALE 1.0] EPCglobal Application Level Events (ALE) 1.0 Specification

https://www.gs1.org/standards/epc-rfid/ale

[**ISO3166**] ISO, “Codes for the Representation of Names of Countries”. http://www.iso.org/iso/en/prods-services/iso3166ma/index.html

[**UTC**] IETF RFC 3339, “Date and Time on the Internet: Timestamps”, July 2002. http://www.ietf.org/rfc/rfc3339.txt

[**EUI64**] "Guidelines For 64-bit Global Identifier (EUI-64)", http://standards.ieee.org/db/oui/tutorials/EUI64.html

[TLS30] IETF RFC 8446, "The Transport Layer Security (TLS) Protocol Version 1.3" , *https://tools.ietf.org/html/rfc8446*

[**X509**] ITU-T Recommendation X.509 (1997 E): Information Technology - Open Systems Interconnection - "The Directory Authentication Framework". 1988.

[**UTF8**] IETF RFC 3629, “UTF-8, a transformation format of ISO 10646”, November 2003. See also Annex D of ISO 10646-1:2000. http://www.ietf.org/rfc/rfc3629.txt

[**HK**] OFTA 1049,

http://www.ofta.gov.hk/en/standards/hktaspec/hkta1049.pdf#search=%22OFTA%201049

%22

[**TW**] DGT LP0002,

http://www.dgt.gov.tw/English/Type-approval/8.4/LP0002/LP0002-940324.pdf

[**JPN**] ARIB STD T89, http://www.arib.or.jp/english/html/overview/st_j.html

[**KOR**] MIC Article 5-2, (Radio Equipment for RFID/USN): Technical Requirements for the radio equipment for passive RFID using the frequency range of 908.5~914MHz.