Reference number of working document: **ISO/IEC JTC 1/SC 31 N 0xxx**

Date: 2006-08-16

Reference number of document: **ISO/IEC FDIS 18000-7**

Committee identification: ISO/IEC JTC 1/SC 31/WG 4

SC 31 Secretariat: ANSI

Information technology AIDC techniques — Radio frequency identification for item management — Air interface, Part 7: Parameters for active air interface communications at 433 MHz

Technologie de l'information - techniques d'AIDC - RFID pour la gestion d'article - aérez l'interface, pièce 7 - paramètres pour des communications actives d'une interface d'air de à 433 mégahertz

Document type: International standard
Document subtype: if applicable
Document stage: (50) Approval
Document language: E

Macintosh HD:Users:craigkharmon:Desktop:ISO_IEC_FDIS18000-7_E.doc Basic template BASICEN3 2002-06-01

Contents

Page

Foreword

ISO (the International Organization for Standardization) and IEC (the International Electrotechnical Commission) form the specialized system for worldwide standardization. National bodies that are members of ISO or IEC participate in the development of International Standards through technical committees established by the respective organization to deal with particular fields of technical activity. ISO and IEC technical committees collaborate in fields of mutual interest. Other international organizations, governmental and non-governmental, in liaison with ISO and IEC, also take part in the work. In the field of information technology, ISO and IEC have established a joint technical committee, ISO/IEC JTC 1.

International Standards are drafted in accordance with the rules given in the ISO/IEC Directives, Part 2.

The main task of the joint technical committee is to prepare International Standards. Draft International Standards adopted by the joint technical committee are circulated to national bodies for voting. Publication as an International Standard requires approval by at least 75 % of the national bodies casting a vote.

Joint Technical Committee ISO/IEC JTC 1, Information technology, Subcommittee SC 31, Automatic identification and data capture techniques, prepared ISO/IEC 18000-7

ISO/IEC 18000 consists of the following parts, under the general title *Information technology automatic identification and data capture techniques — Radio frequency identification for item management:*

— *Part 1: Reference architecture and definition of parameters to be standardized*

— *Part 2: Parameters for air interface communications below 135 kHz*

— *Part 3: Parameters for air interface communications at 13,56 MHz*

— *Part 4: Parameters for air interface communications at 2,45 GHz*

— *Part 6: Parameters for air interface communications at 860 MHz to 960 MHz*

— *Part 7: Parameters for active air interface communications at 433 MHz*

Introduction

This part of ISO/IEC 18000 is intended to address RFID devices operating in the 433 MHz frequency band, providing an air interface implementation for wireless, non-contact information system equipment for Item Management applications. Typical applications operate at ranges greater than one meter.

The RFID system includes a host system and RFID equipment (interrogator and tags). The host system runs an application program, which controls interfaces with the RFID equipment. The RFID equipment is composed of two principal components: tags and interrogators. The tag is intended for attachment to an item, which a user wishes to manage. It is capable of storing a Tag serial number and other data regarding the tag or item and of communicating this information to the interrogator. The interrogator is a device, which communicates to tags in its RF communication range. The interrogator controls the protocol, reads information from the tag, directs the tag to store data in some cases, and ensures message delivery and validity. This system uses an active tag.

RFID systems defined by this part of ISO/IEC 18000 provide the following minimum features:

- Identify tag in range
- Read data
- Write data or handle read only systems gracefully
- Selection by group or address
- Graceful handling of multiple tags in the field of view
- Error detection

The International Organization for Standardization (ISO) and International Electrotechnical Commission (IEC) draw attention to the fact that it is claimed that compliance with this document may involve the use of a patent concerning radio-frequency identification technology given in sub-clause 6.2. ISO and IEC take no position concerning the evidence, validity and scope of this patent right.

The holder of this patent right has assured the ISO and IEC that he is willing to negotiate licenses under reasonable and non-discriminatory terms and conditions with applicants throughout the world. In this respect, the statement of the holder of this patent right is registered with ISO and IEC. Information may be obtained from

Patent Number	Patent Title	Patent Holder	Contact	Affected Clause
US 5640151	Communication System for Communicating with Tags	Savi Technology	Ravi Rajapaksi, Chief Technology Officer, Savi Technology, Inc., 615 Tasman Dr., Sunnyvale, CA 94089	Clause 6.2.6
US 5686902	Communication System for Communicating with Tags	Savi Technology	Ravi Rajapaksi, Chief Technology Officer, Savi Technology, Inc., 615 Tasman Dr., Sunnyvale, CA 94089	Clause 6.2.6
EP 0467036	Method and Apparatus for Radio Identification and Tracking	Savi Technology	Ravi Rajapaksi, Chief Technology Officer, Savi Technology, Inc., 615 Tasman Dr., Sunnyvale, CA 94089	Clause 6.2.6
US 6002344	System and method for electronic inventory	Matrics Technology	Rob Sokohl, Sterne, Kessler, Goldstein & Fox P.L.L.C., 1100 New York Avenue, NW, Washington, DC 20005-3934.	Clause 6.2

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights other than those identified above. ISO or IEC shall not be held responsible for identifying any or all such patent rights.

Information technology AIDC techniques - RFID for item management - Air interface, Part 7, Parameters for active air interface communications at 433 MHz

1 Scope

This part of ISO/IEC 18000 defines the air interface for radio-frequency identification (RFID) devices operating as an active RF Tag in the 433 MHz band used in item management applications. The purpose of this part of ISO/IEC 18000 is to provide a common technical specification for RFID devices that may be used by ISO committees developing RFID application standards. This part of ISO/IEC 18000 is intended to allow for compatibility and to encourage inter-operability of products for the growing RFID market in the international marketplace. This part of ISO/IEC 18000 defines the forward and return link parameters for technical attributes including, but not limited to, operating frequency, operating channel accuracy, occupied channel bandwidth, maximum power, spurious emissions, modulation, duty cycle, data coding, bit rate, bit rate accuracy, bit transmission order, and where appropriate operating channels, frequency hop rate, hop sequence, spreading sequence, and chip rate. This part of ISO/IEC 18000 further defines the communications protocol used in the air interface

2 Conformance

The rules for RFID device conformity evaluation are defined in ISO/IEC TR 18047-7.

3 Normative references

The following referenced documents are indispensable for the application of this document. For dated references, only the edition cited applies. For undated references, the latest edition of the referenced document (including any amendments) applies.

ISO/IEC 15963, *Information technology — Automatic identification and data capture techniques — Radio frequency identification for item management — Unique identification for RF tags*

ISO 17363, *Supply chain applications of RFID – Freight containers*

ISO/IEC TR 18047-7 *Information technology — Automatic identification and data capture techniques — RFID device conformance test methods — Part 7: Test methods for active air interface communications at 433 MHz*

ISO/IEC 19762-1, *Information technology — AIDC techniques — Harmonized vocabulary — Part 1: General terms relating to Automatic Identification and Data Capture (AIDC)*

ISO/IEC 19762-3, *Information technology — AIDC techniques — Harmonized vocabulary — Part 3: Radio-Frequency Identification (RFID)*

ITU-T Recommendation V.41 (Extract from the Blue Book), *Data communication over the telephone network, Code-independent error-control system, Appendix I - Encoding and decoding realization for cyclic code system*

4 Terms and definitions

For the purposes of this document, the terms and definitions given in ISO/IEC 19762-1 and ISO/IEC 19762-3 apply.

5 Symbols and abbreviated terms

For the purposes of this document, the symbols and abbreviated terms given in ISO/IEC 19762-1 and ISO/IEC 19762-3 apply.

6 433,92 MHz active narrowband specification

6.1 Physical layer

The RF communication link between interrogator and tag shall utilize a narrow band UHF frequency with the following characteristics:

Carrier Frequency	433,92 MHz +/-20ppm
Modulation Type	FSK
Frequency deviation	+/- 50 kHz
Symbol LOW	fc +50 kHz
Symbol HIGH	fc -50 kHz
Modulation rate	27,7 kHz
Wake up Signal	31,25 kHz sub-carrier tone followed by 10 kHz tone

The Wake up signal shall be transmitted by the interrogator for a minimum of 2,4 seconds to wake up all tags within communication range. The wake up signal shall consist of a 2,4- to 4,8-second 31,25 kHz sub-carrier tone followed by a 0,1-second 10 kHz sub-carrier tone. Upon detection of the Wake-up signal all tags shall enter into the Ready state awaiting a command from the interrogator. See Figure 1. Once woken up, the tag shall stay awake for 30 seconds after the last command received, unless the interrogator otherwise commands the tag.

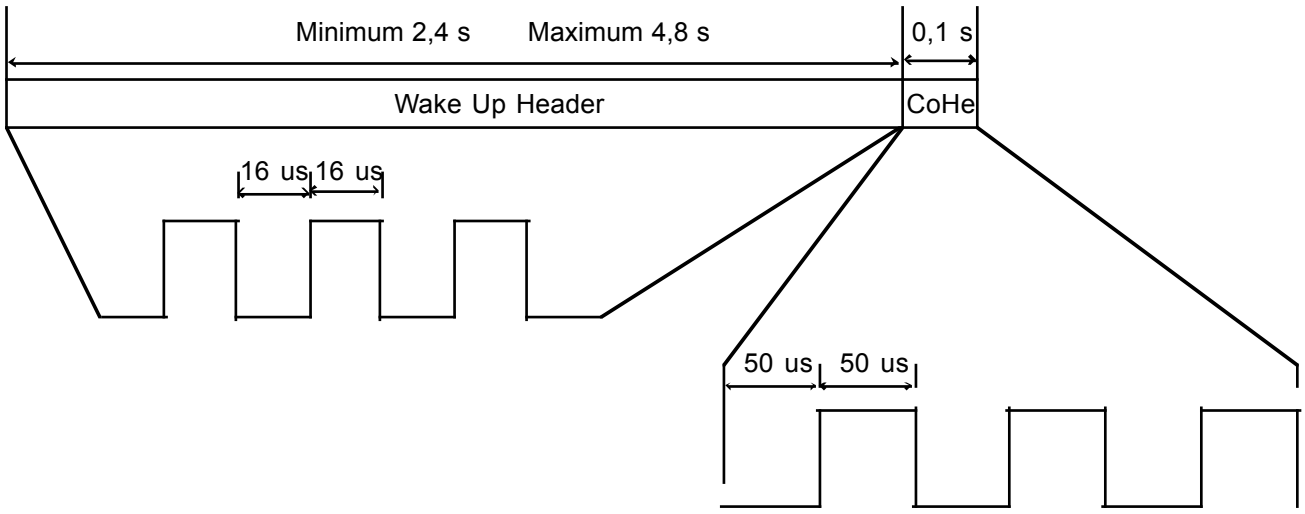


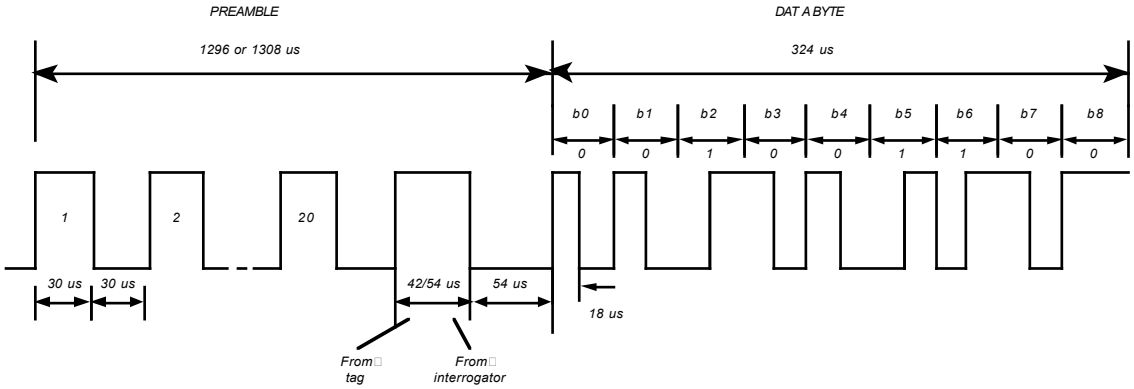
Figure 1 — Wake-up header
(See FDIS_18000-7_Fig1.dxf)

The communication between interrogator and tag shall be of the Master-Slave type, where the interrogator shall initiate communications and then listen for a response from a tag. Multiple response transmissions from tags shall be controlled by the collection algorithm described in section 6.4.

6.2 Data Link layer

6.2.1 General

Data between interrogator and tag shall be transmitted in packet format. A packet shall be comprised of a preamble, data bytes and a final logic low period. The last two pulses of the preamble shall indicate the end of the preamble and beginning of the first data byte. The same two pulses of the preamble also indicate the originator of the data packet. Data bytes shall be sent in Manchester code format. Transmission order shall be most significant byte first; within a byte, the order shall be least significant bit first. Figure 2 illustrates the data communication timing of the preamble and the first byte of a packet.



Note: Data byte transmitted order is most significant byte first; with a byte the order is least significant bit first. Byte shown is code 0x64

Figure 2 — Data communication timing
(See FDIS_18000-7_Fig2.dxf)

6.2.2 Preamble

The preamble shall be comprised of twenty pulses of 60 µs period, 30 µs high and 30 µs low, followed by a final sync pulse which identifies the communication direction: 42 µs high, 54 µs low (tag to interrogator); or 54 µs high, 54 µs low (interrogator to tag).

6.2.3 Data byte format

Data bytes shall be in Manchester code format, comprised of 8 data bits and one stop bit. The bit period shall be 36 µs, the total byte period shall be 324 µs. A falling edge in the centre of the bit-time indicates a 0 bit, a rising edge indicates a 1 bit. The stop bit is coded as a zero bit.

6.2.4 CRC bytes

A CRC checksum shall be calculated as a 16-bit value, initialized with all zeroes (0x0000), over all data bytes (excluding preamble) according to the CCITT polynomial ($x^{16} + x^{12} + x^5 + 1$). The CRC shall be appended to the data as two bytes. Reference: ITU-T Recommendation V.41 (Extract from the *Blue Book*), *Data communication over the telephone network, Code-independent error-control system,* Appendix I - *Encoding and decoding realization for cyclic code system*

6.2.5 Packet end period

A final period of 36 µs of continuous logic low shall be transmitted after the last Manchester encoded bit within the packet..

6.2.6 Interrogator-to-tag message format

Tags shall recognize the interrogator-to-tag message format described in Tables 1 and 2:

Table 1 — Interrogator-to-tag command format (broadcast)

Protocol ID	Packet Options	Packet Length	Interrogator ID	Command Code	Command Arguments	CRC
0x40	1 byte	1 byte	2 bytes	1 byte	N bytes	2 bytes

Table 2 — Interrogator-to-tag command format (point-to-point)

Protocol ID	Packet Options	Packet Length	Tag Manufacturer ID	Tag Serial Number	Interrogator ID	Command Code	Command Arguments	CRC
0x40	1 byte	1 byte	2 bytes	4 bytes	2 Bytes	1 byte	N bytes	2 bytes

•

6.2.6.1 Packet options

Table 3 — Packet options field

Bit							
7	6	5	4	3	2	1	0
Reserved	Reserved	Reserved	Reserved	Reserved	1	0= Broadcast (Tag serial number and Tag manufacturer ID not present) 1= Point to Point (Tag serial number and tag manufacturer ID present)	Reserved

The Packet Options field, described in Table 3, shall be is used to indicate the presence of the Tag serial number and Tag manufacturer fields within the current data packet. If the interrogator wishes to address a single tag by specifying its Tag serial number, Bit 1 of the Packet Options field shall be set, indicating point-to-point communication. In the case in which the interrogator wants to address all tags within its RF communication range, Bit 1 of the Packet Options field shall be cleared to indicate a broadcast message. A broadcast message shall not use the Tag serial number field, which is omitted from the data packet. Reserved bits are for future use. The default value shall be “0”.

6.2.6.2 Protocol ID

The protocol ID field allows different application standards based on 18000-7 (“derived application standards”) to be developed. All derived application standards shall share the same physical layer protocols, but their command/response structure/field and command sets may vary depending on the application. The two basic commands “collection” (i.e., collection with Universal Data Block)” and “sleep and sleep all but” defined in this standard shall be supported by all derived application standards. All other commands required by this standard shall be supported by 18000-7 compliant products, but not necessary by products compliant with derived application standards.

When the interrogator sends out a wake up signal, all tags based on the 18000-7 air interface and derived standards shall wake up.

The interrogator may send out various commands as specified by the application. In the event that the interrogator wants to inventory all the active tags within its range, it shall send out a Collection command as defined in this standard. All tags adhering to 18000-7 or derived application standards shall respond to this basic Collection command. A tag shall respond with the collection response defined by the tag's own application data link layer standard (18000-7 or derived standard). The tags shall also accept the Sleep commands defined in this standard. The co-existence of 18000-7 and derived standards is illustrated in Annex A.

In this standard, the collection command is Collection with Universal Data Block. In the following section, collection command and Collection with Universal Data Block are used interchangeably.

6.2.6.3 Tag Manufacturer ID

The Tag Manufacturer ID is a unique identifier that is issued to each tag manufacturer. The Tag Manufacturer ID is a 16-bit code assigned by the Registration Authority as called out in ISO/IEC 15963. This 16 bit code is a combination of the 15963 Allocation Class "0001 0001" and the 8-bit Issuer UID "xxxxxxxx".

6.2.6.4 Tag Serial Number

The Tag Serial Number is a 32-bit integer that is uniquely assigned to each individual tag during manufacturing. This number cannot be changed and is read only. The Tag Serial Number has no structure and does not contain any information besides uniquely identifying a tag. The Tag Serial Number cannot be reused. Issuance of Tag Serial Numbers may be managed and administered by each manufacturer. The Tag Manufacturer ID and Tag Serial Number together uniquely identify a tag as defined in ISO/IEC 15963. The Tag Serial Number for each manufacturer is four bytes in length. An example of the combined data structure for Tag Manufacturer ID and Tag Serial Number is:

00000001 00010001 xxxxxxxx xxxxxxxxx xxxxxxxx xxxxxxxxx

6.2.6.5 Interrogator ID

The Interrogator ID is a 16-bit integer programmed into the interrogator's non-volatile memory. The Interrogator ID, which can be changed without restriction, is used to route tag responses efficiently through an interrogator network. At the moment the Interrogator ID is changed in an interrogator, any ongoing communication between that interrogator and any tag shall be terminated. An interrogator that receives a tag message containing an Interrogator ID not equal to the Interrogator ID of that interrogator shall not pass the message to the system.

6.2.6.6 Command codes

The Command codes and their function as a Read and/or Write command shall be as listed in Table 4, below. The least significant 7 bits of a command identify its base function; the eighth (MS) bit is set to '0' for a Read function and '1' for a Write function. Codes not identified are reserved. All commands are mandatory unless otherwise specified.

Table 4 — Command codes

Command code (R / W)	Command name	Command type	Mandatory/Optional		Description
			Interrogator	Tag	
0x1F / NA	Collection with Universal Data Block	Broadcast	Mandatory	Mandatory	Collects all Tag IDs and Universal Data Block
NA / 0x15	Sleep	Point to Point	Mandatory	Mandatory	Puts tag to sleep
NA / 0x16	Sleep All But	Broadcast	Mandatory	Mandatory	Puts all tags but one to sleep
0x13 / 0x93	User ID	Point to Point	Mandatory	Optional	Sets user assigned ID (1 – 60 bytes)
0x09 / 0x89	Routing Code	Point to point	Mandatory	Mandatory	Reads and writes routing code
0x0C / NA	Firmware Revision	Point to Point	Mandatory	Optional	Retrieves manufacturer-defined tag firmware revision number
0x0E / NA	Model Number	Point to Point	Mandatory	Optional	Retrieves manufacturer-defined tag model number
0x60 / 0xE0	Read/Write Memory	Point to Point	Mandatory	Optional	Reads and writes user memory
NA / 0x95	Set Password	Point to Point	Mandatory	Optional	Sets tag password (4 bytes long)
0x17 / 0x97	Set Password protect	Point to Point	Mandatory	Optional	Engages/disengages password protection (see section 6.3.4)
NA/ 0x96	Unlock	Point to Point	Mandatory	Optional	Unlocks password protected tag
0x70 / NA	Read Universal Data Block	Point to Point	Mandatory	Mandatory	Reads the Universal Data Block
0x26	Table Create	Point to Point	Mandatory	Optional	Creates a database table
0x26	Table Add Records	Point to Point	Mandatory	Optional	Prepares to add new records to the specified database table
0x26	Table Update Records	Point to Point	Mandatory	Optional	Prepares to modify the specified table records
0x26	Table Update Fields	Point to Point	Mandatory	Optional	Prepares to update the specified fields of a table record
0x26	Table Delete Record	Point to Point	Mandatory	Optional	Deletes existing record from the existing database table
0x26	Table Get Data	Point to Point	Mandatory	Optional	Prepares to retrieve the specified table records
0x26	Table Get Properties	Point to Point	Mandatory	Optional	Gets total number of records and size of each field
0x26	Table Read Fragment	Point to Point	Mandatory	Optional	Retrieves a block of data from a table as initiated by the Table Get Data command
0x26	Table Write Fragment	Point to Point	Mandatory	Optional	Writes a block of data into a table as initiated by the Table Add Records, Table Update Records, or Table Update fields command
0x26	Table Query	Point to Point	Mandatory	Optional	Initiates table search based on the specified criteria
0x11 / NA	Collection Query	Broadcast	Mandatory	Optional	Requests results from Table Query
0xE1 / NA	Beep ON/OFF	Point to Point	Mandatory	Optional	Turns tag's beeper ON or OFF
0x8E	Delete Writeable Data	Point to Point	Mandatory	Optional	Deletes all allocated writeable data on a tag

The Command Type column indicates whether the command is broadcast (does not include Tag Manufacturer ID and Tag serial number in the message) or point-to-point (includes Tag Manufacturer ID and Tag Serial Number in the message).

6.2.6.7 Command arguments

Some commands require additional argument. For those commands where argument is defined, additional data must be supplied with the command. The value of N, which may be zero, and the nature of the data are specific to each command. See section 6.3 for details.

6.2.7 Tag-to-interrogator message format

The tag-to-interrogator message shall use one of two formats depending on the type of message being transmitted to the Interrogator. The tag shall always respond to a command using one of the response formats described below except in the following situations, for which the tag shall not respond:

- the command is explicitly specified in this standard as requiring no response
- the CRC bytes received in the command do not match the CRC checksum that the tag has calculated for the received command packet

There are two possible response formats:

- the Broadcast response message format
- the Point-to-Point response message format

6.2.7.1 Broadcast response message format

The message format shown in Table 5 shall be used in response to Interrogator broadcast commands received by tags within the Interrogator's communication range.

The broadcast command shall be used to collect Tag Manufacturer IDs, Tag Serial Numbers, Routing Codes and optionally application data from the selected group of tags using the batch collection algorithm. See 6.3.1 for more details.

Table 5 — Broadcast response message format

Protocol ID	Tag Status	Packet Length	Interrogator ID	Tag Manufacturer ID	Tag Serial Number	Command Code	Data	CRC
0x40	2 bytes	1 byte	2 bytes	2 bytes	4 bytes	1 byte	N bytes	2 bytes

Tag Status: Indicates various conditions such as response format, tag type, and alert flag. See section 6.2.7.4, **Tag Status,** for more details.

Packet Length: Message length in bytes including CRC byte codes

Interrogator ID: ID of Interrogator: Integer value from 1 to 65535

Tag Manufacturer ID: Unique ID assigned to manufacturer

Tag Serial Number: Unique tag serial number preset during manufacturing

Command Code: Command code (see Table 4) received from the Interrogator

Data: Contains the Universal Data Block that includes the mandatory Routing Code and optionally application data (see Table 6). N is the size, in bytes, of the Universal Data Block.

CRC: CCITT code check bytes

Table 6 — Type ID values

Type ID (1 byte)		Note
0x00 - 0x0A	Reserved	
0x10	Routing code	The routing code is specified in the ISO 17363 standard
0x11	User ID	User ID as specified within this document
0x12 – 0x7F	Reserved	These types are reserved for future tag data elements
0x7F – 0xFF	Future extension	Also reserved

6.2.7.2 Point-to-point response message format

This message format, shown in Table 7, shall be returned to the Interrogator as a response to all point-to-point commands, which require the Tag Serial Number in order to access a particular tag. (The point-to-point commands include all commands except Collection commands).

Table 7 — Tag–to-interrogator response format (point-to-point)

Protocol ID	Tag Status	Packet Length	Interrogator ID	Tag Manufacturer ID	Tag Serial Number	Command Code	Response Data*	CRC
0x40	2 bytes	1 byte	2 bytes	2 bytes	4 bytes	1 byte	N bytes	2 bytes

* This field is command dependent; some commands may or may not need this field

Tag Status: Indicates various conditions such as response format, tag type, and alert flag. See section 6.2.7.4, **Tag Status,** for more details.

Packet Length: Message length in bytes including CRC byte codes

Interrogator ID: ID of Interrogator, an Integer value from 1 to 65535

Tag Manufacturer ID: Unique ID assigned to manufacturer.

Tag Serial Number: Unique tag serial number preset during manufacturing

Command Code: Command code received from the Interrogator

Response Data: Data returned by the tag as a response to an Interrogator's valid command request. The value of N, the length of the data in bytes, is specific to the command. In the event that the tag receives an invalid command, a NACK flag within the Tag Status word will be set and the Response Data will contain an error code of one or more bytes, as described in Table 8.

CRC: CCITT code check bytes

6.2.7.3 Error codes

In response to a point-to-point command a tag may reply with one of the errors listed in Table 8. Errors generated resulting from broadcast commands do not generate responses.

Table 8 — Error code

Error Code	Description
0x01	Invalid Command Code
0x02	Invalid Command Parameter
0x04	Not Found
0x06	Can't Create Object
0x08	Authorization Failure
0x09	Object is Read-Only
0x3f	Implementation Dependent
0x40	Sequence ID Mismatch
0x41	Boundary Exceeded

An error shall consist of a one-byte error code; possibly a one-byte sub-code, depending on the kind of error; possibly one or more bytes of parameter data, also depending on the error; and an optional, manufacturer-defined number of additional data bytes, as shown in Table 9. In the following error definition sections, the optional, manufacturer-defined data bytes are not shown.

Table 9 — General error format

Error Code	Sub-code	Error Data	Manufacturer Data
1 byte	1 byte	N bytes	M bytes

Error Code: a value from Table 8 identifying the kind of error

Sub-code: an optional value that further refines the nature of the error and is specific to the kind error, as defined in the following subsections

Error Data: N bytes of data, where N is zero or greater, whose existence, length, and content depend on the nature of the error, as defined in the following subsections

Manufacturer Data: M bytes of data, where M is zero or greater, whose existence, length, and content are at the discretion of the tag manufacturer

6.2.7.3.1 Invalid command code error

Table 10 shows the structure of this error code.

Table 10 — Invalid command code error

Error Code
0x01

This error as defined in Table10 shall be generated when the tag receives a packet with a command code or class code that is not defined in this standard.

6.2.7.3.2 Invalid command parameter error

Table 11 shows the structure of this error code.

Table 11 — Invalid command parameter error

Error Code	Sub-code	Parameter Offset
0x02	1 byte	1 byte

Sub-code: a code as shown in Table 12 that describes the error more specifically. Following values are defined:

Table 12 — Invalid command parameter error sub-codes

Sub-code	Sub-error Name	Meaning
0x01	Parameter Out of Range	The value of a parameter is not legal
0x02	Too Few Parameters	There are fewer bytes in the Command Arguments field than expected
0x03	Too Many Parameters	There are more bytes in the Command Arguments field than expected

Parameter offset: the offset in bytes from the beginning of the Command Arguments field where the error was detected.

6.2.7.3.3 Not found error

Table 13 shows the structure of this error code.

Table 13 — Not found error

Error Code	Sub-code
0x04	1 byte

Sub-code: a code as shown in Table 14 that describes the error more specifically. Following values are defined:

Table 14 — Not found error sub-codes

Sub-code	Sub-error Name	Meaning
0x01	Table Does Not Exist	There is no existing table for the table ID given
0x02	Record Does Not Exist	There are fewer records than the record number given
0x03	Field Does Not Exist	There are fewer fields than the field number given

6.2.7.3.4 Can’t create object error

Table 15 shows the structure of this error code.

Table 15 — Can’t create object error

Error Code	Sub-code
0x06	1 byte

This error as shown in Table 15 shall be generated upon an unsuccessful attempt to create a database table. The **Sub-code** field in the Table 16 below shall indicate the reasons why the attempt failed.

Table 16 — Can't create object error sub-codes

Sub-code	Sub-error Name	Meaning
0x02	Table Already Exists	The requested table ID is already in use
0x03	Out of Memory	There is insufficient memory in the tag to create the requested table
0x04	Table ID Reserved	The tag restricts usage of the table ID to a particular purpose.

6.2.7.3.5 Authorization failure error

Table 17 shows the structure of this error code.

Table 17 — Authorization failure error

Error Code
0x08

This error as shown in Table 17 shall be generated upon an invalid attempt to access a tag feature protected by a password.

6.2.7.3.6 Object is read-only error

Table 18 shows the structure of this error code.

Table 18 — Object is read-only error

Error Code
0x09

This error as shown in Table 18 shall be generated upon an attempt to modify some tag data entity for which the tag does not allow modifying operations.

6.2.7.3.7 Implementation dependent error

Table 19 shows the structure of this error code.

Table 19 — Implementation dependent error

Error Code	Sub-code
0x43	1 byte

This error code as shown in Table 19 shall be reserved for tag manufacturers to define for general tag behaviour errors not covered by this standard. At a minimum, the tag implementation must include a **Sub-code** field. Additional fields of the error are left to the tag manufacturer to specify.

6.2.7.3.8 Sequence ID mismatch error

Table 20 shows the structure of this error code.

Table 20 — Sequence ID mismatch error

Error Code
0x40

This error as shown in Table 20 shall be generated upon an attempt to delete a record in a database table, passing to the command a sequence Table Full sequence ID of the most recent successful invocation of the Table Delete Record command. (See the Table Delete Record command in section 0.)

6.2.7.3.9 Boundary exceeded error

This error as shown in Table 21 and sub-code shown in Table 22 shall be generated upon an attempt to access a record outside of a valid boundary.

Table 21 — Boundary exceeded error

Error Code	Sub-code
0x41	1 byte

Table 22 — Boundary exceeded error sub-codes

Sub-code	Sub-error Name	Meaning
0x01	Table Full	The table has been filled to the create-time allotment
0x02	Record Does Not Exist	The record has not been added yet
0x03	Fragment Overrun	The write operation completed with still more to write

6.2.7.4 Tag status

The Tag Status field shown in Table 23, included in all tag–to-interrogator messages, shall consist of the following information:

Table 23 — Tag status field format

Bit							
15	**14**	**13**	**12**	**11**	**10**	**9**	**8**
Mode field				Reserved	Reserved	Reserved	Acknowledgement 1 = NACK 0 = ACK

Bit							
7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
Reserved		Tag type			Reserved	Reserved	Service bit

Note: reserved fields are set to a value of “0”

Mode field indicates the format (response to Broadcast command or response to Point-to-Point command) of the response data from the tag. The list of possible values is shown in Table 24.

Table 24 — Tag status field format

Mode field	Mode format code (bit15 - 12)
Broadcast Command	0000
Point to Point Command	0010

Acknowledgment, when clear (‘0’), indicates that the tag has received a valid command (CRC ok and all fields valid) from the Interrogator and processed the command successfully. If set (‘1’), the command was invalid or the tag encountered an error during the processing of the command. Note that as described in section 6.2.7, the tag issues no response in the case of a CRC error.

Tag type is a value assigned by, and meaningful only to, the tag manufacturer. The manufacturer can use this value to indicate manufacturer-defined special features.

Service bit when set ('1') indicates that tag battery is low. Manufacturers or the application standards will define battery-low parameters. The exception to this rule is when the Interrogator performs Collection Query. In this case, a tag will indicate through this bit whether search criteria have been met. When this bit is set ('1') upon response to the Collection Query command, it indicates that a match was found. In all other cases the Service bit indicates the low-battery condition.

6.3 Tag commands

6.3.1 Collection with Universal Data Block (UDB)

The format of the **Collection with Universal Data Block** shall be as shown in Table 25,

Table 25 — Collection with Universal Data Block

Command Code	Windows size	Max Packet Length	Type
0x1F	2 bytes	1 byte	1 byte

- **Window Size:** the number of 57,3 ms intervals to use for each window in the collection algorithm. See section 6.4 for an explanation of the collection algorithm.
- **Max Packet Length:** an integer in the range 1 to 255 inclusive that specifies the maximum size, in bytes, of a response packet. The interrogator uses this parameter to instruct a tag into what sized packets to partition a large Universal Data Block (UDB), given that the size of the whole UDB might exceed the maximum allowable RF transmission packet size as determined by local ordinances.

- **Type** is reserved for future expansion and is currently required to have the value 0x00.

The Collection with UDB command shall be used to collect all Tag Serial Numbers with the content of the UDB data block. The UDB contains data elements that are formatted using a Type, Length and Value annotation scheme. Each data element shall be uniquely identified with a Type ID (see Table 6).

6.3.2 Sleep

To put a tag to **Sleep** the command in Table 26 shall be sent (written) to the tag.

Table 26 — Sleep command format (Write)

Command code
0x15

Upon receiving the **Sleep** command in Table 26, the tag shall enter the Sleep mode. The tag shall not respond to this command or to any subsequent command until the tag is woken again by the Wake-up signal.

6.3.3 Sleep all but

To put all except one tag to **Sleep** the command in Table 27 shall be sent (written) to the tag.

Table 27 — Sleep all but command format (Write)

Command code	Tag Manufacturer ID	Tag Serial Number
0x16	2 bytes	4 bytes

The **Sleep All But** command is a broadcast command used to place all tags into the sleep mode, as with the **Sleep** command of section 6.3.2, except the one tag that matches the provided Tag Manufactures ID and Tag Serial Number. Upon receiving this command, all tags except the one tag that matches the provided Tag Manufactures ID and Tag Serial Number shall enter "sleep" mode.

The tags shall not respond to this command.

6.3.4 Security commands

Access to tag write commands shall be guarded by a *password protection* mechanism that application software can command the tag to engage or disengage. If password protection is engaged, those write commands shall be *non-accessible unless the tag is unlocked*; that is, they will not perform their usual operations but rather respond with an Authorization Failure error. If the password protection is disengaged, the commands are *accessible* – they behave as described in the corresponding sections of this standard without the possibility of an Authorization Failure error. Password protection is engaged and disengaged by means of the **Set Password Protect** command described in section 6.3.4.2. Password protection is disengaged by default.

While password protection is engaged, application software can command the tag to enter the *unlocked* state temporarily. While a tag is unlocked, the password-protected write commands shall be accessible. The tag returns to the *locked* state, in which the password-protected commands shall be non-accessible, when the tag returns to the Sleep mode. The **Unlock** command of section 6.3.4.3 puts the tag into the unlocked state. There is no command to put a tag into the locked state explicitly.

The Table 28 lists the commands that are affected by password protection.

Table 28 — Write commands affected by password protection

Command code	Command name	Description
0x93	User ID	Sets user assigned ID (1 – 60 bytes)
0x89	Routing Code	Writes routing code
0xE0	Write Memory	Writes user memory
0x95*	Set Password*	Sets tag password (4 bytes long)
0x97*	Set Password Protect*	Engages/disengages password protection
0x26	Table Create	Creates a database table
0x26	Table Add Records	Prepares to add new records to the specified database table
0x26	Table Update Records	Prepares to modify the specified table records
0x26	Table Update Fields	Prepares to update the specified fields of a table record
0x26	Table Delete Record	Deletes existing record from the existing database table
0x26	Table Write Fragment	Writes a block of data into a table as initiated by the Table Add Records, Table Update Records, or Table Update fields command
0x8E	Delete Writeable Data	Deletes all allocated writeable data on a tag,

* These commands behave as though password protection were engaged permanently.

6.3.4.1 Security — Set Password

To set the password of a tag, the command in Table 29 shall be sent (written) to the tag.

Table 29 — Set Password command format (write)

Command code	Password
0x95	4 bytes

To the **Set Password** command the tag shall respond (write response) as shown in Table 30.

Table 30 — Set Password command format (write response)

Command code
0x95

This command sets the tag's password. This command requires tag to be first unlocked with the **Unlock** command of section 6.3.4.3 before the command can be accessed. The initial value of the tag's password is 0xFFFFFFFF.

The possible error responses shall be as shown in the Table 31.

Table 31 — Set Password command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	**Password** parameter is missing or the wrong length
0x08	Authorization Failure	**Unlock** command not invoked prior to invocation of this command

6.3.4.2 Security — Set Password Protect

To set a tag to **Password Protect**ed the command in Table 32 shall be sent (written) to the tag.

Table 32 — Set Password Protect command format (write)

Command code	Secure
0x97	1 byte

- **Secure:** a value of 0x01, engages password protection;
- If Secure byte is set to '0x00', tag password protection condition is disengaged

To the **Set Password Protect** command the tag shall respond (write response) as shown in Table 33.

Table 33 — Set password protect command format (write response)

Command code
0x97

This command engages or disengages password protection in the tag. To access this command the tag shall first be unlocked with the **Unlock** command of section 6.3.4.3 regardless of the state of the tag's password protection.

The possible error responses shall be as shown in Table 34.

Table 34 — Set Password Protect command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	Secure parameter is missing or the wrong length
0x08	Authorization Failure	Unlock command not invoked prior to invocation of this command

6.3.4.3 Security — Unlock

To unlock a tag the command in Table 35 shall be sent (written) to the tag.

Table 35 — Unlock command format (write)

Command code	Password
0x96	4 bytes

To the **Unlock** command the tag shall respond (write response) as shown in Table 36.

Table 36 — Unlock command format (write response)

Command code
0x96

This command unlocks the tag. If the supplied password matches tag's password, the tag shall permit the execution of all commands ordinarily non-accessible because of password protection. The tag shall remain in the unlocked state until it receives the Sleep command or 30 seconds has elapsed since the tag received a command.

The possible error responses shall be as shown in Table 37.

Table 37 — Unlock command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	Password parameter is missing or the wrong length
0x08	Authorization Failure	Incorrect password supplied

6.3.5 Transit information commands

6.3.5.1 User ID

To retrieve a tag's User ID the command in Table 38 shall be sent to the tag.

Table 38 — User ID command format (read)

Command code
0x13

To the **User ID** read command the tag shall respond as shown in Table 39.

Table 39 — User ID command format (read response)

Command code	User ID Length	User ID
0x13	1 byte	N bytes

To set a tag's **User ID** the command in Table 40 shall be sent to the tag.

Table 40 — User ID command format (write)

Command code	User ID Length	User ID
0x93	1 byte	N bytes

To the **User ID** write command the tag shall respond as shown in Table 41.

Table 41 — User ID command format (write response)

Command code
0x93

- **User ID Length:** the length, N, in bytes, of the User ID, where N is between 0 and 60 inclusive.
- **User ID:** the contents of the User ID

The User ID is a user-readable and writeable memory whose meaning and size (up to 60 bytes) is user defined. The User ID can be used, for example, to store a value that uniquely identifies the asset to which the tag is affixed. This command sets and gets the size and contents of the User ID. In addition to this command, the Collection with UDB command also retrieves the User ID, except if the **User ID Length** parameter is set to zero, in which case the UDB message will not contain the User ID. The default length of the User ID is zero.

The possible error responses shall be as shown in Table 42.

Table 42 — User ID command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	The length of the User ID parameter does not agree with the User ID Length parameter, or the wrong number of parameter bytes was given
0x08	Authorization Failure	Write command was attempted with password protection engaged and tag in the locked state
0x41	Boundary Exceeded	User ID Length parameter is greater than the maximum, 60

6.3.5.2 Routing Code

To retrieve a tag's Routing Code the command in Table 43 shall be sent to the tag.

Table 43 — Routing Code command format (read)

Command code
0x09

To the **Routing Code** read command the tag shall respond as shown in Table 44.

Table 44 — Routing Code command format (read response)

Command code	Routing Code Length	Routing Code
0x09	1 bytes	N bytes

To set a tag's **Routing Code** the command in Table 45 shall be sent to the tag.

Table 45 — Routing Code command format (write)

Command code	Routing Code Length	Routing Code
0x89	1 byte	N bytes

To the **Routing Code** write command the tag shall respond as shown in Table 46.

Table 46 — Routing Code command format (write response)

Command code
0x89

- **Routing Code Length:** the length, N, in bytes, of the Routing Code, where the value of N is either zero or equal to the value defined by ISO/IEC 15459.

- **Routing Code:** the contents of the Routing Code

The Routing Code is a user-readable and writable memory whose purpose is to store a Country-specific code for asset routing purposes. Note that the Routing Code is part of the tag's response to the Collection with UDB command, except if the **Routing Code Length** parameter is set to zero, in which case the UDB message will not contain the Routing Code. The default length of the Routing Code is zero.

The possible error responses shall be as shown in Table 47.

Table 47 — Routing Code command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	**Routing Code Length** parameter is neither zero nor the value defined by ISO/IEC 15459, or the length of the **Routing Code** parameter does not agree with the **Routing Code Length** parameter, or the wrong number of parameter bytes was given
0x08	Authorization Failure	Write command was attempted with password protection engaged and tag in the locked state

6.3.6 Manufacturing Information Commands (Optional)

The following two commands, which are not mandatory for compliance with this part of ISO/IEC 18000, enable the tag manufacturer to provide manufacturer-defined, immutable information about a tag.

6.3.6.1 Firmware Version (Optional)

To retrieve a tag's **Firmware Version** the command in Table 48 shall be sent to the tag.

Table 48 — Firmware Version command format (read)

Command code
0x0C

To the **Firmware version** command the tag shall respond as shown in Table 49.

Table 49 — Firmware Version command format (read response)

Command code	Firmware version
0x0C	4 bytes

The Firmware Version indicates the tag firmware version.

6.3.6.2 Model Number (Optional)

To retrieve a tag's **Model Number** the command in Table 50 shall be sent to the tag.

Table 50 — Model number command format (read)

Command code
0x0E

To the **Model Number** command the tag shall respond as shown in Table 51.

Table 51 — Model Number command format (read response)

Command code	Model number
0x0E	2 bytes

The Model Number indicates the tag model number.

6.3.7 Memory commands

A tag may provide zero or more bytes of user-readable and writable random-access memory in which the user can store and retrieve user-defined data. Associated with every byte of memory is an *address*, through which that memory byte can be accessed.

6.3.7.1 Write Memory

To write memory the command in Table 52 shall be sent (written) to the tag.

Table 52 — Write Memory command format (write)

Command Code	Number of Bytes	Start Address	Data
0xE0	1 byte	3 bytes	N bytes

To the **Write Memory** command the tag shall respond as shown in Table 53.

Table 53 — Write Memory command format (write response)

Command Code
0xE0

- **Number of Bytes:** N, the number of bytes to write, in the range 1 to 46 inclusive
- **Start Address:** the memory address of the first memory byte to write, in the range 0 to the manufacturer-defined maximum address
- **Data:** the memory contents to write

The **Write Memory** command stores the given data into the user random-access memory for later retrieval with the **Read Memory** command of the next section.

The possible error responses shall be as shown in Table 54.

Table 54 — Write Memory command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	The length of the Data parameter does not agree with the Number of Bytes parameter, or the wrong number of parameter bytes was given
0x08	Authorization Failure	Write command was attempted with password protection engaged and tag in the locked state
0x41	Boundary Exceeded	Number of Bytes parameter is outside its legal range, or Start Address plus Number of Bytes extends beyond the maximum address

6.3.7.2 Read Memory

To read memory the command in Table 55 shall be sent to the tag.

Table 55 — Read Memory command format (read)

Command Code	Number of Bytes to Read	Start Address
0x60	1 byte	3 bytes

To the **Read Memory** command, the tag shall respond as shown in Table 56.

Table 56 — Read Memory command format (read response)

Command Code	Number of Bytes Actually Read	Data
0x60	1 byte	N bytes

- **Number of Bytes to Read:** the number of bytes to read, in the range 1 to 46 inclusive
- **Number of Bytes Actually Read:** N, the number of bytes of data returned in the response, which always agrees with **Number of Bytes to Read**
- **Start Address:** the memory address of the first memory byte to read, in the range 0 to the manufacturer-defined maximum address
- **Data:** the memory contents read

The **Read Memory** command retrieves from the user random-access memory the requested data previously written with the **Write Memory** command of the previous section.

The possible error responses shall be as shown in Table 57.

Table 57 — Read Memory command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	The wrong number of parameter bytes was given
0x41	Boundary Exceeded	**Number of Bytes to Read** parameter is outside its legal range, or **Start Address** plus **Number of Bytes to Read** extends beyond the maximum address

6.3.8 Delete Writeable Data

To delete all allocated writeable data on a tag, the command in Table 58 shall be sent to the tag. Data that is permanent on the tag or that is marked non-writeable is left untouched.

Table 58 — Delete Writeable Data

Command code
0x8A

To the **Delete Writeable Data** command the tag shall respond as shown in Table 59.

Table 59 — Delete Writeable Data (response)

Command code
0x8A

This command restores all user-writeable memory to factory defaults. In particular, the following operations are performed:

- The length of the User ID is reset to zero.
- The length of the Routing Code is reset to zero.
- All database tables are deleted.

The possible error responses shall be as shown in Table 60.

Table 60 — Delete Writeable Data command errors

Error Code	Error Name	Reason
0x08	Authorization Failure	Command was attempted with password protection engaged and tag in the locked state

6.3.9 Read Universal Data Block (UDB)

To read the Universal Data Block the command in Table 61 shall be sent to the tag.

Table 61 — Read UDB

Command Code	Sequence ID	Max Packet Length	Type
0x70	1 byte	1 byte	1 byte

To the **Read Universal Data Block** command, the tag shall respond as shown in Table 62.

Table 62 — Read UDB Response

Command Code	Sequence ID Countdown	Universal Data Block
0x70	1 byte	N bytes

- **Sequence ID:** used by the interrogator to identify a specific packet in order to facilitate error correction during a multi-packet session. The initial value of **Sequence ID** is set to 0. It is incremented after each successful packet reception. The last **Sequence ID** shall be equal to the initial **Sequence ID Countdown** parameter transmitted by the tag.
- **Max Packet Length:** an integer in the range 1 to 255 inclusive that specifies the maximum size, in bytes, of a response packet. The interrogator uses this parameter to instruct a tag into what sized packets to partition a large Universal Data Block (UDB), given that the size of the whole UDB might exceed the maximum allowable RF transmission packet size as determined by local ordinances.
- **Sequence ID Countdown:** the total number of packets that the Interrogator shall retrieve from the Tag. The Sequence ID Countdown parameter transmitted by the Tag starts with the highest index value (equal to the total number of packet less one) and is decremented by one after each successful transmission until it reaches the value of zero.
- **Type** field is reserved for future expansion and currently required to have the value 0x00.

The **Read Universal Data Block** command fetches one packet of the Universal Data Block (UDB) within a sequence of UDB packets.

The possible error responses shall be as shown in Table 63.

Table 63 — Read UDB command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	The Sequence ID parameter is greater than the total number of UDB packets, or Max Packet Length is zero, or the wrong number of parameter bytes was given.

6.3.10 Database table commands (Optional)

The Database Table commands provide basic database table functionality, allowing application software to create one or more tables of varying schemas, perform table updates, and query a table. A tag is not required to implement these commands. The Database Table commands provide no mechanism for performing table joins. The schema and maximum number of records of a Database Table is fixed at creation time.

A table schema consists of a list of field widths, in bytes. Field numbers are indexed starting at 0 for the first field. Every field in a table is untyped; that is, all field value comparisons are performed on a byte-for-byte basis, with equality being established between two field values if all bytes in each value match. One field value is considered “less than” a second field value if for some byte position *p* in the two values, all bytes in the byte range 0 to *p*-1 are equal in the two values, and byte *p* of the first value is less than byte *p* of the second value. In other words, a straight multi-byte value comparison is performed with the first byte being the most significant and the last byte being the least significant.

Table records are indexed starting at 0 for the first record. The *record number* (the record index) does not maintain a fixed relationship with a record. When a record is deleted, all the records whose record number is greater than the deleted record are decremented. Thus, deleting record 4 three times, for example, deletes the records that were numbered 4, 5, and 6 before the first deletion occurred.

Associated with a database table is a table ID, an immutable 2-byte value assigned at table creation time that uniquely identifies a table among all other tables in the tag.

6.3.10.1 Table Create (optional)

When invoking **Table Create** the command in Table 64 shall be sent to the tag.

Table 64 — Table Create

Command Code	Sub Opcode	Table ID	Maximum Number of Records	Number of Fields	Length of Each Field
0x26	0x01	2 bytes	2 bytes	1 byte	N bytes

Where:

- **Table ID** indicates the pre-assigned identifier value for the table. Table ID 0x0000 is reserved.
- **Maximum Number of Records** indicates how many records can ultimately exist in the table in total.
- **Number of Fields** indicates number of the fields per record and its range is 1-32
- **Length of Field** is a byte array of length N bytes, where N is the number of fields, as specified by the **Number of Fields** parameter. Each one-byte element of the byte array indicates the size of a field. The first element of the byte array specifies the length of the first field; the second element specifies the length of the second field, and so forth. The length of a field must lie within the range 1 to 255 inclusive.

To the **Table Create** command the tag shall respond as shown in Table 65.

Table 65 — Table Create response

Command Code
0x26

This command creates a database table consisting of the specified number of fields each having the specified length. Initially after creation, the table has no records.

The possible error responses shall be as shown in Table 66.

Table 66 — Table Create command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	A parameter is missing, or the **Number of Fields** parameter is outside its legal range, or the length of the **Length of Field** array does not match **Number of Fields**, or one or more of the Length of Field elements is zero, or the wrong number of parameter bytes was given.
0x06	Can't Create Object	The **Table ID** is already assigned to an existing table, or the tag does not have sufficient memory, or **Table ID** is 0x0000
0x08	Authorization Failure	Command was attempted with password protection engaged and tag in the locked state

6.3.10.2 Table Add Records (Optional)

When invoking **Table Add Records** the command in Table 67 shall be sent to the tag.

Table 67 — Table Add Records

Command Code	Sub Opcode	Table ID	Number of Records
0x26	0x02	2 bytes	2 bytes

Where:

- **Table ID** indicates the pre-assigned identifier value for the table.
- **Number of Records** indicates the total number of records to add to the table.

To the **Table Add Records** command the tag shall respond as shown in Table 68.

Table 68 — Table Add Records

Command Code	Token
0x26	N bytes

This command instructs the tag to prepare to add the specified number of records to the Table. The record contents are specified with a series of subsequent Table Write Fragment commands.

- **Token** indicates a value used to iteratively write data to the added records. The high-order 4 bits of the first byte of the token indicates the length of the token, not including the first byte, so zero indicates a token length of 1 byte (see Table Write Fragment). The **Token** value of exactly 0x00 is reserved, and indicates an end-of-iteration condition. The structure of a **Token** field is shown below in Table 69:

Table 69 — Token structure

N: Token Length	Token Data
N value in bits 7-4 [Value of N = 0 – 16]	4 low order bits of Token Length byte, then N bytes

The possible error responses shall be as shown in Table 70.

Table 70 — Table Add Records command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	**Number of Records** is zero, or the wrong number of parameter bytes was given
0x04	Not Found	There is no database table associated with the specified table ID
0x08	Authorization Failure	Command was attempted with password protection engaged and tag in the locked state
0x09	Object is Read-Only	**Table ID** is 0x0000, which specifies the read-only query results table
0x41	Boundary Exceeded	The table is too full to accept an additional **Number of Records** new records

6.3.10.3 Table Update Records (Optional)

When invoking **Table Update Records** the command in Table 71 shall be sent to the tag.

Table 71 — Table Update Records

Command Code	Sub Opcode	Table ID	Starting Record Number	Number of Records
0x26	0x03	2 bytes	2 bytes	2 bytes

Where:

- **Table ID** indicates the pre-assigned identifier value for the table.
- **Starting Record Number** indicates the first record to begin updating.
- **Number of Records** indicates the total number of records that will be updated.
- **Token** indicates a value used to iteratively write data to the updated records. The high-order 4 bits of the first byte of the token indicates the length of the token, not including the first byte, so zero indicates a token length of 1 byte (see Table Write Fragment). The **Token** value of exactly 0x00 is reserved, and indicates an end-of-iteration condition. The structure of a **Token** field is shown in Table 69.

To the **Table Update Records** command the tag shall respond as shown in Table 72.

Table 72 — Table Update Records response

Command Code	Token
0x26	N bytes

This command instructs the tag to prepare to update the specified table records. The new record contents are specified with a series of subsequent Table Write Fragment commands.

The possible error responses shall be as shown in Table 73.

Table 73 — Table Update Records command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	**Number of Records** is zero, or the wrong number of parameter bytes was given
0x04	Not Found	There is no database table associated with the specified **table ID**
0x08	Authorization Failure	Command was attempted with password protection engaged and tag in the locked state
0x09	Object is Read-Only	**Table ID** is 0x0000, which specifies the read-only query results table
0x41	Boundary Exceeded	Starting **Record Number plus Number of Records** extends beyond the total number of records in the table

6.3.10.4 Table Update Fields (Optional)

When invoking **Table Update Fields** the command in Table 74 shall be sent to the tag.

Table 74 — Table Update Fields

Command Code	Sub Opcode	Table ID	Record Number	Starting Field Number	Number of Fields
0x26	0x04	2 bytes	2 bytes	1 byte	1 byte

Where:

- **Table ID** indicates the pre-assigned identifier value for the table.
- **Record Number** indicates the record to update.
- **Starting Field Number** indicates the first field to begin updating.
- **Number of Fields** indicates the total number of fields in the specified record that will be updated.
- **Token** indicates a value used to iteratively write data to the updated records. The high-order 4 bits of the first byte of the token indicates the length of the token, not including the first byte, so zero indicates a token length of 1 byte (see Table Write Fragment). The **Token** value of exactly 0x00 is reserved, and indicates an end-of-iteration condition. The structure of a **Token** field is shown in Table 69.

To the **Table Update Fields** command the tag shall respond as shown in Table 75.

Table 75 — Table Update Fields

Command Code	Token
0x26	N bytes

This command instructs the tag to prepare to update the specified fields of a table record. The new field contents are specified with a series of subsequent Table Write Fragment commands.

The possible error responses shall be as shown in Table 76.

Table 76 — Table Update Fields command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	**Number of Fields** is zero, or the wrong number of parameter bytes was given
0x04	Not Found	There is no database table associated with the specified table ID
0x08	Authorization Failure	Command was attempted with password protection engaged and tag in the locked state
0x09	Object is Read-Only	**Table ID** is 0x0000, which specifies the read-only query results table
0x41	Boundary Exceeded	**Record Number** is greater than or equal to the total number of records in the table, or **Number of Fields** plus **Starting Field Number** extends beyond the number of fields in the table

6.3.10.5 Table Delete Record (Optional)

When invoking **Table Delete Record** the command in Table 77 shall be sent to the tag.

Table 77 — Table Delete Record

Command Code	Sub Opcode	Table ID	Sequence ID	Record Number
0x26	0x05	2 bytes	1 byte	2 bytes

Where:

- **Table ID** indicates the pre-assigned identifier value for the table.
- **Sequence ID** is used to ensure a unique transaction. With every invocation of this command, the interrogator should supply a different value for **Sequence ID**. If the interrogator receives no reply from an invocation of the command (due to a communication error, for example) the interrogator may confidently retry deleting the same record, without the danger of an unintentional deletion of a second record, by supplying the same **Sequence ID** as in the first invocation.
- **Record Number** indicates the index number of the record to delete

To the **Table Delete Record** command the tag shall respond as shown in Table 78.

Table 78 — Table Delete Record

Command Code
0x26

This command instructs the tag to delete a single record from the Table. When the record is deleted from the database all records below the deleted one will be shifted upwards by one record size. As a side effect, any iteration tokens that have been generated (by one of the iterating table commands) are made invalid.

The possible error responses shall be as shown in Table 79.

Table 79 — Table Delete Record command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	The wrong number of parameter bytes was given
0x04	Not Found	There is no database table associated with the specified table ID
0x08	Authorization Failure	Command was attempted with password protection engaged and tag in the locked state
0x09	Object is Read-Only	**Table ID** is 0x0000, which specifies the read-only query results table
0x40	Sequence ID Mismatch	The **Sequence ID** parameter is exactly the **Sequence ID** of the most recent successful invocation of the command, indicating a redundant request
0x41	Boundary Exceeded	**Record Number** is greater than or equal to the total number of records in the table

6.3.10.6 Table Get Data (Optional)

When invoking **Table Get Data** the command in Table 80 shall be sent to the tag.

Table 80 — Table Get Data

Command Code	Sub Opcode	Table ID	Starting Record Number	Starting Field Number
0x26	0x06	2 bytes	2 bytes	1 byte

To the **Table Get Data** command the tag shall respond as shown in Table 81.

Table 81 — Table Get Data response

Command Code	Token
0x26	N bytes

Where:

- **Table ID** indicates the pre-assigned identifier value for the table.
- **Starting Record Number** indicates the first record to begin reading.
- **Starting Field Number** indicates the first field to begin reading.
- **Token** indicates a value used to iteratively read record data. The high-order 4 bits of the first byte of the token indicates the length of the token, not including the first byte, so zero indicates a token length of 1 byte (see Table Read Fragment). The **Token** value of exactly 0x00 is reserved, and indicates an end-of-iteration condition. The structure of a **Token** field is shown in Table 69.

The **Table Get Data** command instructs the tag to prepare to read data from a database table starting with a specified record and field. A sequence of **Table Read Fragment** commands performs the actual data reading. Unlike table write commands, **Table Get Data** is an open-ended iteration that terminates either at the application software's choosing or when the end of the table is reached.

The possible error responses shall be as shown in Table 82.

Table 82 — Table Get Data command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	The wrong number of parameter bytes was given
0x04	Not Found	There is no database table associated with the specified table ID, or there is no query result for this table, either because no query was executed on the table or the query results have been made invalid by an intervening modification to the table that was queried or by starting a new query.
0x41	Boundary Exceeded	Starting Record Number is greater than or equal to the total number of records in the table, or Starting Field Number is greater than or equal to the number of fields in the table

6.3.10.7 Table Get Properties (Optional)

When invoking **Table Get Properties** the command in Table 83 shall be sent to the tag.

Table 83 — Table Get Properties

Command Code	Sub Opcode	Table ID
0x26	0x07	2 bytes

To the **Table Get Properties** command the tag shall respond as shown in Table 84.

Table 84 — Table Get Properties response

Command Code	Total Number of Records	Maximum Number of Records	Reserved
0x26	2 bytes	2 bytes	1 bytes

Where:

- **Table ID** indicates the pre-assigned identifier value for the table.
- **Total Number of Records** indicates total number of records in the table.
- **Maximum Number of Records** indicates the maximum allocated records for the table.
- **Reserved** is a byte reserved for future use.

The **Table Get Properties** command retrieves information about the specified table, namely, the number of used records in the table and the capacity of the table.

The possible error responses shall be as shown in Table 85.

Table 85 — Table Get Properties command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	The wrong number of parameter bytes was given
0x04	Not Found	There is no database table associated with the specified table ID

6.3.10.8 Table Read Fragment (Optional)

When invoking **Table Read Fragment** the command in Table 86shall be sent to the tag.

Table 86 — Table Read Fragment

Command Code	Sub Opcode	Request Token	Requested Read Length
0x26	0x08	N bytes	1 byte

To the **Table Read Fragment** command the tag shall respond as shown in Table 87.

Table 87 — Table Read Fragment response

Command Code	Response Token	Actual Read Length	Data
0x26	N bytes	1 byte	N bytes

Where:

- **Request Token** is the token from the prior **Table Get Data** or **Table Read Fragment** command. The high-order 4 bits of the first byte of the token indicates the length of the token, not including the first byte, so zero indicates a token length of 1 byte (see Table Read Fragment). The **Token** value of exactly 0x00 is reserved, and indicates an end-of-iteration condition. The structure of a **Token** field is shown in Table 69.
- **Requested Read Length** is the requested length of data to return.
- **Response Token** is the resulting new token from a successful **Table Read Fragment** command. The high-order 4 bits of the first byte of the token indicates the length of the token, not including the first byte, so zero indicates a token length of 1 byte. The **Token** value of exactly 0x00 is reserved, and indicates an end-of-iteration condition.
- **Actual Read Length** is the length of data actually read.
- **Data** is the actual data read.
- **N** is the number of bytes for data read.

The **Table Read Fragment** command reads a block of data bytes from a database table. Which bytes to read are implied by the **Request Token** parameter created by a prior invocation of the **Table Get Data** command or a previous invocation of this **Table Read Fragment** command.

The possible error responses shall be as shown in Table 88.

Table 88 — Table Read Fragment command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	Request Token is malformed, or Requested Read Length is zero, or the wrong number of parameter bytes was given
0x40	Sequence ID Mismatch	Request Token is properly formed and not 0x00, but is invalid due to an intervening modification of the table to which the Request Token applies
0x41	Boundary Exceeded	Request Token is 0x00

6.3.10.9 Table Write Fragment (Optional)

When invoking **Table Write Fragment** the command in Table 89 shall be sent to the tag.

Table 89 — Table Write Fragment

Command Code	Sub Opcode	Request Token	Data Length	Data
0x26	0x09	N bytes	1 byte	N bytes

To the **Table Write Fragment** command the tag shall respond as shown in Table 90.

Table 90 — Table Write Fragment response

Command Code	Response Token
0x26	N bytes

Where:

- **Request Token** is the token from the prior **Table Add Records**, **Table Update Records**, **Table Update Fields**, or **Table Write Fragment** command. The high-order 4 bits of the first byte of the token indicates the length of the token, not including the first byte, so zero indicates a token length of 1 byte (see Table Read Fragment). The structure of a **Token** field is shown in Table 69.
- **Data Length** is the length of data to write.
- **Response Token** is the resulting new token from a successful **Table Write Fragment** command. The high-order 4 bits of the first byte of the token indicates the length of the token, not including the first byte, so zero indicates a token length of 1 byte. The **Token** value of exactly 0x00 is reserved, and indicates an end-of-iteration condition.

The **Table Write Fragment** command writes a block of data bytes to a database table. Which bytes to write are implied by the **Request Token** parameter created by a prior invocation of the **Table Add Records, Table Update Records,** or **Table Update Fields** command or a previous invocation of this **Table Write Fragment** command.

The possible error responses shall be as shown in Table 91.

Table 91 — Table Write Fragment command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	Request Token is malformed, or Data Length is zero, or the length of the Data parameter does not agree with Data Length, or the wrong number of parameter bytes was given
0x08	Authorization Failure	Command was attempted with password protection engaged and tag in the locked state
0x40	Sequence ID Mismatch	Request Token is properly formed and not 0x00, but is invalid due to an intervening modification of the table to which the Request Token applies
0x41	Boundary Exceeded	Request Token is 0x00, or the Data Length for this request would exceed the length declared in the original Table Add Records, Table Update Records, or Table Update Record Fields command

6.3.10.10 Table Query (Optional)

When invoking **Table Query** the command in Table 92 shall be sent to the tag.

Table 92 — Table Query

Command Code	Sub Opcode	Table ID	Sequence ID	Logical Operator	Field Number	Relational Operator	Data Length	Data
0x26	0x10	2 bytes	1 byte	1 byte	1 byte	1 byte	1 byte	N bytes

Where:

- **Table ID** indicates the pre-assigned identifier value for the table.
- **Sequence ID** field indicates a current operand/operator sequence number. For example, for given expression: (**A** and **B** or **C** and **D)** the operator/operand **Clear A** would represent the first query (Sequence number = 3), **And B** would represent second query (Sequence number = 2), **Or C** would represent third query (Sequence number = 1) and **And D** would represent fourth query (Sequence number = 0),. This Sequence # parameter starts with highest number (equal to total number of operands less one) and is decremented by one for each additional operand until it reaches value of zero.
- **Logical Operator** is used to define search criteria expression with following defined operators 'C' (CLEAR), 'A' (AND), or 'O' (OR). Operator CLEAR is used to reset any previously remaining criteria expression.
- **Field Number** indicates index number of the field within the record (equivalent to rows of the Table). The range of the Field Number is from 1 - 32.
- **Relational Operator** is used to define search criteria expression with following defined operators '=' (EQUAL), '<' (LESS THAN), '>' (GREATER THAN), '!' (NOT).
- **Data Length** indicates length of the Data field in bytes.
- **Data** field specifies data (operand) that is being searched for. This field is 1 to 32 bytes long and accepts wild card symbol '*' as the first byte to indicate that a match should be indicated if the remaining bytes of the parameter is found anywhere within the target field. For example, when searching for word *BOOTS within fields specified as ABCBOOTSXYZ would yield positive match.

To the **Table Query** command the tag shall respond as shown in Table 93.

Table 93 — Table Query response

Command Code	Number of Records matched	Index of first matched record
0x26	2 bytes	2 bytes

This command searches a tag's database (table) for the specific information. For each operand/operator pair (**A** and - **B** or - **C** and - **D**) an individual command is being sent until a full expression is being received by the tag (indicated through Sequence number being equal to zero). The operand in this case can be any word that needs to match in combination with other criteria. For example if searching for BLUE BOOTS within the records that have two fields: **Colour** and **Item Name** the expression would be as follows: 1st query is 'C', 'BLUE', 'A' (where 'C' clears any previous expression, BLUE being operand or word which is searched for within the specified field of the record and 'A' standing for AND logical operator; 2nd query is 'BOOTS' this creating the full expression BLUE and BOOTS).

The result of the search is placed in the non-volatile memory at the end of the Tag database file. The results are saved in the 2 bytes array where each byte pair corresponds to the index (memory address) of the matched record. The end the database file can be calculated based on the total number of records and the length of the record.

The command is protected for queries that are too long to fit in memory or that will run out of the space allocated for queries. In these cases an error indicating that the query is not a complete query is returned.

The logical operators <= and >= are not implemented directly but can be simulated by altering the last bit of the data (with carry over all bytes). The Table Query command can handle sequence of queries logically connected with AND and OR operators (**A** and **B** or **C** and **D**), as long as parenthesis are not required to express the full query.

This command is used to broadcast a database (table) query to all tags. The data base query search criteria is performed in multiple steps by transmitting individual operand/operator pair to all tags. The "Number to Follow" field indicates how many commands need to be received by the tag until a full expression is being transmitted (**A** and **B** or **C** and **D**). So, for instance, the "Number to Follow" field with the A part transmitted would be 3, and the tag would decrement it by one after receiving each subsequent multipart queries: the B, C, and D queries.

Those tags that received full search criteria expression will start internal database search and upon completion will wait for Collection_Query command from Interrogator in order to respond with their search results.

If the tag misses any of the B, C, or D queries the tag will not respond to Collection_Query command.

The result of the search is placed in table 0 (zero). The table has a single field of two-bytes per field (and thus also two bytes per record). Each record corresponds to the query table matched record number. The two-byte record number is stored in the field in MSB first order.

Application can access these memory areas using the **Table Get Data** and **Table Read Fragment** commands on Table ID 0 (zero) in order to obtain the record number of each matched record in its original table.

The possible error responses shall be as shown in Table 94.

Table 94 — Table Query command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	Sequence ID is greater than the maximum number of query operators that the tag supports; or Logical Operator, Field Number, or Relational Operator are outside their legal range of values; or Data Length is zero; or the length of Data does not agree with Data Length, or the wrong number of parameter bytes was given
0x04	Not Found	There is no database table associated with the specified table ID

6.3.10.11 Collection Query (Optional)

When invoking **Collection Query** the command in Table 95 shall be sent to the tag.

Table 95 — Collection Query

Command Code	Window Size	Reserved
0x11	2 bytes	1 byte

This command is executed after all the tags have completed database search upon receiving multiple **Table Query** commands. This command is used to collect the results of the previous database query. Tag will return their Tag serial number with the service bit set if there was a match or set to zero if match was not found.

The result of the search is placed in Table ID 0 (zero). The results are saved in a one-field table of 2 bytes per record length, where each record (each field) indicates the record number of the matched record in its corresponding table.

6.3.11 Beep ON/OFF (Optional)

When invoking **Beep ON/OFF** the command in Table 96 shall be sent to the tag.

Table 96 — Beep ON/OFF

Command Code	Beeper On/Off
0xE1	1 byte

Where:

- **Beeper On/Off** parameter when 0x01 will turn tag's beeper ON or when set to 0x00 will turn tag's beeper OFF.

To the **Beep ON/OFF** command the tag shall respond as shown in Table 97.

Table 97 — Beep ON/OFF response

Command Code
0xE1

The **Beep ON/OFF** command turns the tag's beeper on or off. When the tag's beeper is turned on, the beeper stays on until explicitly turned off or when the tag returns to the Sleep mode.

The possible error responses shall be as shown in Table 98.

Table 98 — Beep ON/OFF command errors

Error Code	Error Name	Reason
0x02	Invalid Command Parameter	Beeper On/Off parameter is missing or the wrong length or outside its legal range of values

6.4 Tag collection and collision arbitration

This standard specifies the method by which the interrogator shall identify and collect the memory contents of one or more tags present in the operating field of the interrogator over a common radio frequency channel. This standard also specifies the method by which the Interrogator shall communicate with individual tags. Communication may be: identify a tag, read the tag data payload, write to tag memory or command the tag to perform a specific function.

General explanation

The collection process involves the arbitration of the tag population so as to reduce or eliminate the collisions that may occur between multiple tag transmissions. Communication with the tag population present takes place in a series of communications periods called batch sessions. Tags do not transmit unless commanded to do so by the interrogator. The Interrogator may conduct an inventory of tags present or communicate with a single individual tag where the interrogator has knowledge of the tag identity.

Tag arbitration uses a mechanism that allocates tag reply transmissions into time slots within a time window referred to as a Listen Period (LP). Listen Periods are contained within one or more collection rounds referred to as Collection Periods (CP). The arbitration process begins when the Interrogator transmits a collection command. On receiving a collection command a tag selects at random a slot in which it will reply. A tag that has selected slot 1 will transmit its reply immediately. A tag that has selected a slot other than the first slot will wait for the appropriate length of time and then transmit. Each slot time is of sufficient duration to accommodate a complete tag reply. The number of slots in a Listen Period is determined by the tag based on the maximum packet length (unit in bytes) and window size (unit of multiples of 57,3ms, for example, window size of 1 means 57.3ms duration) communicated by the Interrogator. The Interrogator may adjust the window size (number of slots) based on the number of collisions or on the number of unused slots within a particular Listen Period. As the tag population is reduced, the Interrogator will reduce the size of subsequent Listen Periods accordingly.

Tag Collection and Arbitration

Refer to figure 3 **Interrogator - Tag communication timing**, figure 4 **Tag Batch Session Elements** and Figure 5 **Collection Sequence and Timing**. A batch session commences when the Interrogator transmits a *Wakeup* signal as defined in clause 6.1. A Batch session shall contain one or more Collection Periods, CP(1), CP(2), . . . , CP(p), . . . , CP(P). A Tag shall not transmit until it is commanded to do so by the Interrogator. A Collection Period shall commence when the Interrogator transmits the broadcast command *Collection with Universal Data Block*. Each Collection Period shall contain the following time periods

- Synchronisation Period, SP – the time period used by the Interrogator to send a broadcast command to the tag population present. The Tag shall synchronize initial slot start time with Interrogator broadcast command.
- Listen Period, LP – the time period in which the tags present, respond to the broadcast command. The Listen Period, LP, shall be divided into a number of time slots TS. The duration of each time slot shall be determined by the Window Size parameter in the *Collection with Universal Data Block* command and the tag response message time duration.
- Acknowledge Period, AP – the time period in which the Interrogator acknowledges a tag response and during which the Interrogator may optionally command the tag to transmit additional data.

Tag Message Length and Slot selection

The tag message length and Slot selection are determined by parameters contained in the broadcast collection command (e.g., *Collection with Universal Data Block).*

The tag reply packet length (message length) shall be determined by the Max Packet Length parameter. The time duration of the slots is given by

- **Slot Size = tag reply packet length + 2 milliseconds**.

The number of slots in a Listen Period shall be determined by the command parameter Window Size according to the equation. Slot Size shall be rounded up to the next nearest milliseconds. The slot guard time is 2 milliseconds that allows the Interrogator to process the data and ready to receive the next tag.

- **Number of slots = (Window Size * 57,3ms) / Slot Size** (where Window Size is the number of 57,3ms intervals in the Listen Period).

The tag shall incorporate a random number generator that the tag shall use to select the slot in which to reply. Number of slots shall be rounded down to the whole number.

When the tag receives the command *Collection with Universal Data Block* it shall:

- set the slot size according to the Max Packet Length parameter contained in the command,
- set the maximum number of slots using the Window Size parameter.
- select a time slot TS(n) in which to reply in range of 1 to D
- transmit its reply when it has waited for the time TS(n) * (Slot Size-1)

The tag shall reply only once during a Collection Period.

Acknowledge and Read Tag data

After the Interrogator has transmitted the *Collection with Universal Data Block* command there will be one of three possible outcomes.

1. The Interrogator does not receive a tag reply within the Listen Period because there were no tags present within the reader interrogation field or because one or more tags present did not recognise the command.
2. One or more tags replied but no tag reply was successfully decoded by the Interrogator, due to message corruption or an invalid Tag CRC. In the case of a collision the Interrogator shall record the collision and continue to listen for further Tag replies in the current Listen Period.
3. The Interrogator receives a valid Tag response without error. The Interrogator shall record the Tag identity and shall continue to listen until the end of the current Listen Period and shall record all valid Tag responses.

When the Listen Period is finished, i.e. at the end of the Window time, the Interrogator shall transmit one or more point-to-point (directed) commands. Refer figure 5 **Collection sequence and timing**. If there is no need to read Tag data then the Interrogator shall transmit a *Sleep* command for each of the Tag identities collected during the preceding Listen Period. If further data is be read from one or more of the Tags, then the Interrogator shall transmit a *Read Universal Data Block* command for each Tag for which Tag data is required.

Alternatively, the Interrogator may transmit the broadcast command *Sleep All But*. This command will cause all Tags present to move into the Sleep state except the individual Tag specified in the *Sleep All But* command parameters Tag Manufacturer ID and Tag Serial Number.

When the Interrogator has acknowledged all the Tags collected during the current Collection Period, the Interrogator may issue a further *Collection with Universal Data Block* and repeat the same process described for the first Collection Period.

If during the preceding Collection Period, the Interrogator senses that the there are too many collisions or that there is a significant amount of empty air time, the Interrogator shall adapt the communication channel bandwidth by transmitting control parameters in the Collection command. If there are too many collisions, the Interrogator shall use a larger Window Size parameter in the *Collection with Universal Data Block* command. If there are few or no collisions, then the Interrogator shall reduce the Window Size parameter.

When the Interrogator receives no further Tag replies in a Collection Period, it shall transmit two further *Collection with Universal Data Block* commands using a small Window Size parameter to ensure that there are no more Tags present in the Interrogator field.

Note:

Tag Batch Session elements (Refer figure 4, Tag Batch Session Elements)

A Batch Session beginning with the *Wakeup* signal contains the following elements.

- Each Batch Session contains P (one or more) Collection Periods. CP(1), CP(2), . . . , CP(p), . . . , CP(P).
- Each Collection Period starts with a broadcast command in the Synchronisation Period SP, followed by a Listen Period LP and ending with an Acknowledgment Period AP. Where the batch contains more than one Collection Period, the
    - Synchronization Period in each Collection Period is referred to as SP(1), SP(2), . . . , SP(p), . . . , SP(P) respectively.
    - Each Listen Period is referred to as LP(1), LP(2), . . . , LP(p), . . . , LP(P) respectively and
    - Each Acknowledge Period is referred to as AP(1), AP(2), . . . , AP(p), . . . , AP(P) respectively.
- Each Listen Period has a number D of Time Slots TS, the number and duration of which are specified in the parameters of the broadcast command. Listen Period LP(1) has D1 Time Slots TS(1,1), TS(1,2), . . . , TS(1,D1), LP(2) has D2 time slots TS(2,1), TS(2,2), ... , TS(2,D2) etc…

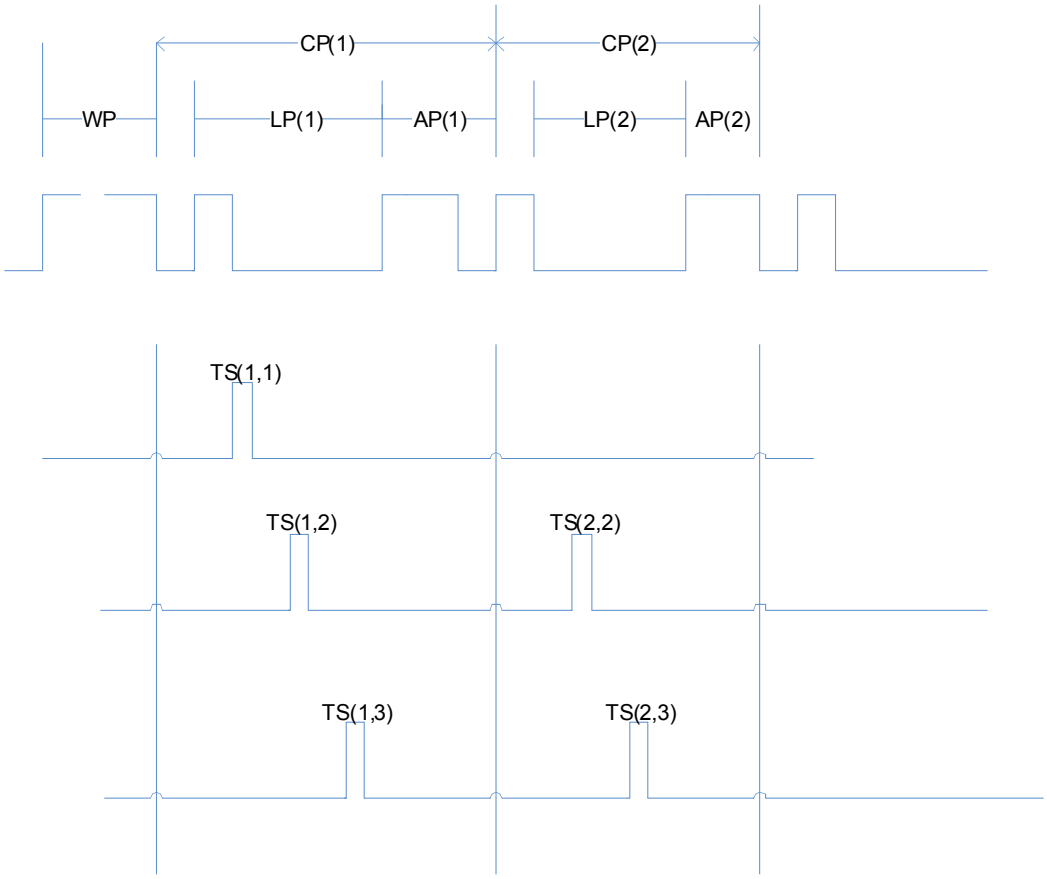


Figure 3 — Interrogator-tag communication timing diagram

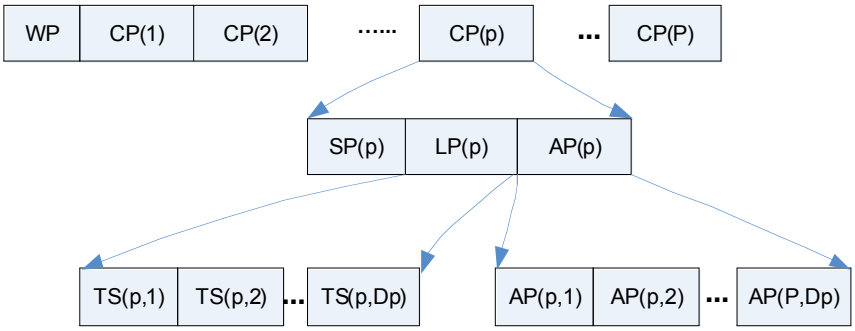


Figure 4 Detailed Interrogator-tag anti-collision scheme timing diagram

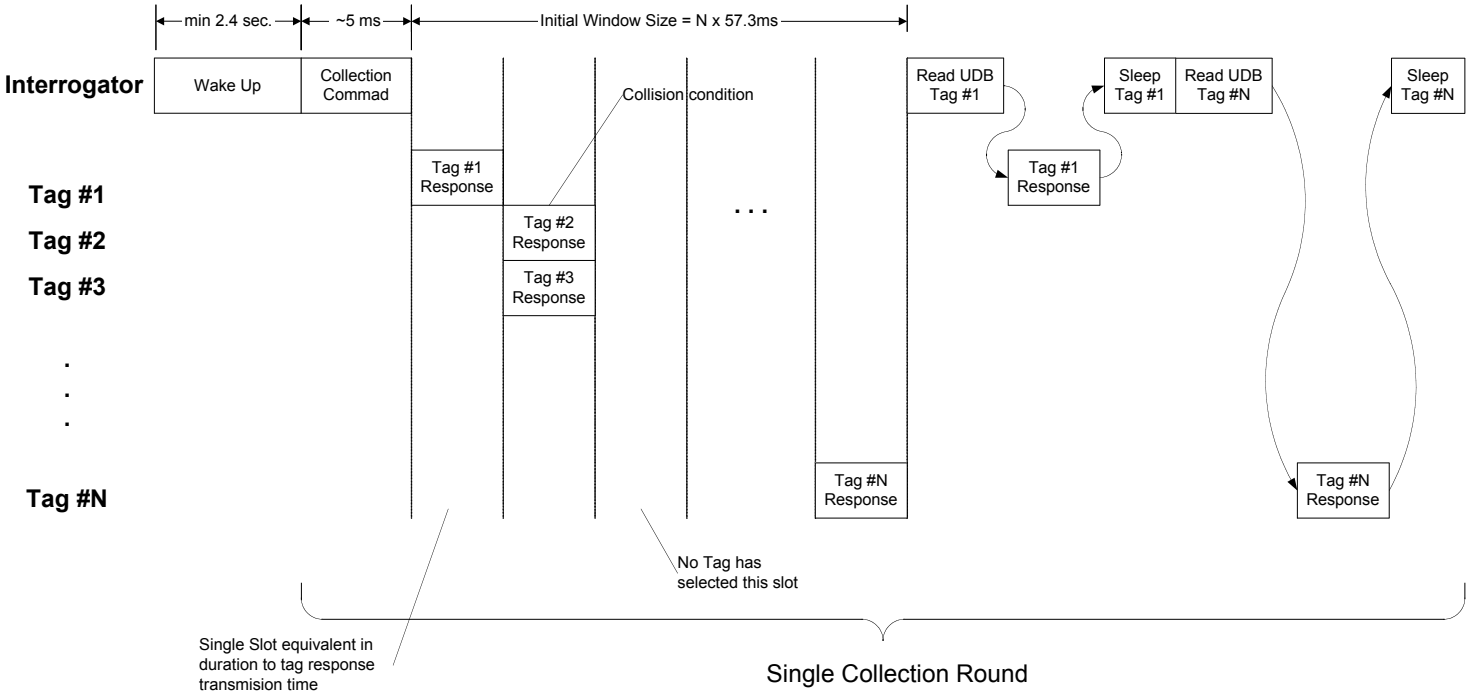


Figure 5 — Collection sequence and timing

6.5 Multi-packet UDB collection with mandatory Routing Code and application data (User ID)

The following section shows a typical handshaking process between the Interrogator and the Tag within a population of tags:

1. Figure 6 illustrates where the interrogator broadcasts following packet to all Tags (starts an inventory of the Tags).

Protocol Code	Packet Options	Packet Length	Interrogator ID	Command Code	Command Arguments	CRC
0x40	8 bits	1 byte	2 bytes	0x1F	N bytes	2 bytes

Window Size	Max Packet Length	Type
2 bytes	1 byte	1 byte

Figure 6 — Interrogator broadcast for the initial collection of the tags

2. Figure 7 illustrates the Tag response to Interrogator's command. The Reply is in the format of Point-to-Point packet (see section 6.2.7.7 Read Universal Data Block for Sequence ID Countdown parameter definition).

Protocol Code	Tag Status	Packet Length	Interrogator ID	Mfg ID	Tag Serial Number	Command Code	Data	CRC
0x40	2 bytes	1 byte	2 bytes	2 bytes	4 bytes	1 byte	N bytes	2 bytes

Sequence ID Countdown	Routing Code Type	Routing Code Length	Routing Code	User ID Type	User ID Length	User ID
1 byte	1 byte	1 byte	M bytes	1 byte	1 byte	N-M-5 - n bytes

Max Packet Length

Figure 7 — Tag response to interrogator's request with the point –to-point packet

3. Figure 8 shows an Interrogator *Read Universal Data Block* command directed to a Tag previously identified during the inventory session. The command is directed to the Tag using the Tag ID discovered in a previous Collection session. Note that the Sequence ID starts with the value 1 because the initial collection command implicitly retrieved the first packet of the UDB message (Sequence ID = 0). See section 6.3.9, *Read Universal Data Block* for Sequence ID Countdown parameter definition.

Protocol Code	Packet Options	Packet Length	Mfg ID	Tag Serial Number	Interrogator ID	Command Code	Command Arguments	CRC
0x40	1 byte	1 byte	2 bytes	4 bytes	2 bytes	0x70	N bytes	2 bytes

Sequence ID	Max Packet Length	Type
1 byte	1 byte	1 byte

Figure 8 — Interrogator's request for the remainder of information with a point–to-point request to a specific tag

4. Figure 9 shows the Tag reply in response to the Interrogator command. The reply contains the second part of the UDB message using the point-to-point (directed) packet format.

Protocol Code	Tag Status	Packet Length	Interrogator ID	Mfg ID	Tag Serial Number	Command Code	Data	CRC
0x40	2 bytes	1 byte	2 bytes	2 bytes	4 bytes	1 byte	N bytes	2 bytes

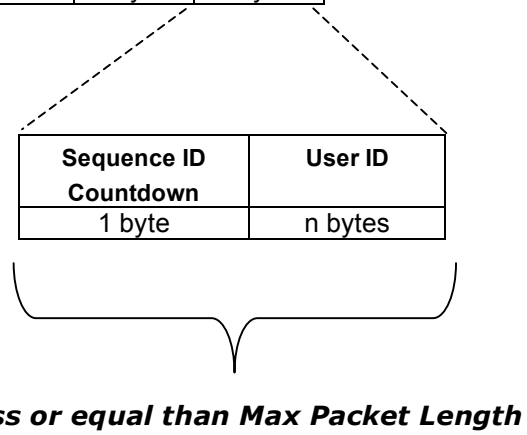


Figure 9 — Tag response to interrogator's request for the remainder of information with a point –to-point request to a specific tag

6.6 Physical and Media Access Control (MAC) parameters

6.6.1 Interrogator to tag link

The interrogator to tag link parameters are summarized in Table 99.

Table 99 — Interrogator to tag link parameters

Ref.	Parameter	Value
Int:1	Operating Frequency range	433,92 MHz
Int:1a	Default Operating Frequency	433,92 MHz
Int:1b	Operating Channels	Not applicable in this mode
Int:1c	Operating Frequency Accuracy	+/-20 ppm
Int:1d	Frequency Hop Rate	Not applicable in this mode
Int:1e	Frequency Hop Sequence	Not applicable in this mode
Int:2	Occupied Channel Bandwidth	200 kHz (transmit)
Int:2a	Minimum Receiver Bandwidth	500 kHz
Int:3	Interrogator Transmit Maximum EIRP	As allowed by local regulations
Int:4	Interrogator Transmit Spurious Emissions	
Int:4a	Interrogator Transmit Spurious Emissions, In-Band	Not applicable in this mode
Int:4b	Interrogator Transmit Spurious Emissions, Out of Band	The interrogator shall transmit in conformance with spurious emissions requirements defined by the country's regulatory authority within which the system is operated.
Int:5	Interrogator Transmitter Spectrum Mask	Not applicable in this mode
Int:6	Timing	
Int:6a	Transmit to Receive Turn Around Time	1 ms
Int:6b	Receive to Transmit Turn Around Time	1 ms
Int:6c	Interrogator Transmit Power On Ramp	1 ms
Int:6d	Interrogator Transmit Power Down Ramp	1 ms
Int:7	Modulation	Frequency Shift Keying (FSK)
Int:7a	Spreading Sequence	Not applicable in this mode
Int:7b	Chip Rate	Not applicable in this mode
Int:7c	Chip Rate Accuracy	Not applicable in this mode
Int:7d	Modulation Index	Not applicable in this mode
Int:7e	Duty Cycle	Not applicable in this mode
Int:7f	FM Deviation	±50 kHz
Int:8	Data Coding	Manchester, 36 µs bit period. Logic one: 18 µs low followed by 18 µs high, logic zero: 18 µs high followed by 18 µs low
Int:9	Bit Rate	27,778 kbit/s
Int:9a	Bit Rate Accuracy	200 ppm
Int:10	Interrogator Transmit Modulation Accuracy	Not applicable in this mode
Int:11	Preamble	

Ref.	Parameter	Value
Int:11a	Preamble Length	Twenty one (21) bits
Int:11b	Preamble Waveform	Square wave as defined in Int:11c
Int:11c	Bit Sync Sequence	20 cycles of 30 µs high, 30 µs low, followed by one cycle 54 µs high, 54 µs low
Int:11d	Frame Sync Sequence	20 cycles of 30 µs high, 30 µs low, followed by one cycle 54 µs high, 54 µs low
Int:12	Scrambling	Not applicable in this mode
Int:13	Bit Transmission Order	Byte: least significant bit (LSB) first Data: most significant byte first
Int:14	Wake-up Process	Yes
Int:15	Polarization	Omni-directional

6.6.2 Tag to interrogator link

The tag to interrogator link parameters are summarized in Table 100.

Table 100 — Tag to interrogator link parameters

Ref.	Parameter	Value
Tag:1	Operating Frequency range	433,92 MHz
Tag:1a	Default Operating Frequency	433,92 MHz
Tag:1b	Operating Channels	Not applicable in this mode
Tag:1c	Operating Frequency Accuracy	+/-20 ppm
Tag:1d	Frequency Hop Rate	Not applicable in this mode
Tag:1e	Frequency Hop Sequence	Not applicable in this mode
Tag:2	Occupied Channel Bandwidth	200 kHz (transmit)
Tag:2a	Minimum Receiver Bandwidth	500 kHz
Tag:3	Transmit Maximum EIRP	5,6 dBm (peak power) or as allowed by local regulations
Tag:4	Transmit Spurious Emissions	
Tag:4a	Transmit Spurious Emissions, In-Band	Not applicable in this mode
Tag:4b	Transmit Spurious Emissions, Out of Band	The tag shall transmit in conformance with spurious emissions requirements defined by the country's regulatory authority within which the system is operated.
Tag:5	Transmit Spectrum Mask	Not applicable in this mode
Tag:6	Timing	
Tag:6a	Transmit to Receive Turn Around Time	1 ms
Tag:6b	Receive to Transmit Turn Around Time	1 ms
Tag:6c	Transmit Power On Ramp	1 ms
Tag:6d	Transmit Power Down Ramp	1 ms
Tag:7	Modulation	Frequency Shift Keying (FSK)
Tag:7a	Spreading Sequence	Not applicable in this mode
Tag:7b	Chip Rate	Not applicable in this mode
Tag:7c	Chip Rate Accuracy	Not applicable in this mode

Ref.	Parameter	Value
Tag:7d	On-Off Ratio	Not applicable in this mode
Tag:7e	Sub-carrier Frequency	Not applicable in this mode
Tag:7f	Sub-carrier Frequency Accuracy	Not applicable in this mode
Tag:7g	Sub-carrier Modulation	Not applicable in this mode
Tag:7h	Duty Cycle	Not applicable in this mode
Tag:7i	FM deviation	±50 kHz (FSK deviation)
Tag:8	Data Coding	Manchester, 36 µs bit period. Logic one: 18 µs low followed by 18 µs high, Logic zero: 18 µs high followed by 18 µs low
Tag:9	Bit Rate	27,778 kbit/s
Tag:9a	Bit Rate Accuracy	200 ppm
Tag:10	Tag Transmit Modulation Accuracy	Not applicable in this mode
Tag:11	Preamble	
Tag:11a	Preamble Length	Twenty one (21) bits
Tag:11b	Preamble Waveform	Square wave as defined in Tag:11c
Tag:11c	Bit Sync Sequence	20 cycles of 30 µs high, 30 µs low, followed by one cycle 42 µs high, 54 µs low
Tag:11d	Frame Sync Sequence	20 cycles of 30 µs high, 30 µs low, followed by one cycle 42 µs high, 54 µs low
Tag:12	Scrambling	Not applicable in this mode
Tag:13	Bit Transmission Order	Byte: least significant bit (LSB) first Data: most significant byte first
Tag:14	(Reserved by committee)	
Tag:15	Polarization	Omni-directional
Tag:16	Minimum Tag Receiver Bandwidth	200 kHz

6.6.3 Protocol parameters

The protocol parameters are summarized in Table 101.

Table 101 — Protocol parameters

Ref.	Parameter Name	Description
P:1	Who talks first	Reader-Talks-First (RTF)
P:2	Tag addressing capability	Yes
P:3	Tag UID	Yes
P:3a	UID Length	32 bit
P:3b	UID Format	binary
P:4	Read size	20 bytes
P:5	Write Size	20 bytes
P:6	Read Transaction Time	25 ms

Ref.	Parameter Name	Description
P:7	Write Transaction Time	30 ms
P:8	Error detection	CCITT 16
P:9	Error correction	None
P:10	Memory size	1 byte to 128 kilobytes
P:11	Command structure and extensibility	8 bits for command

6.6.4 Anti-collision parameters

The anti-collision parameters are summarized in Table 102.

Table 102 — Anti-collision parameters

Ref.	Parameter Name	Description
A:1	Type	Probabilistic
A:2	Linearity (for N tags)	Probabilistic: 0,065*N seconds for 1 <= N <= 3000
A:3	Tag inventory capacity	Probabilistic: 3000

Annex A
(normative)

Co-existence of different application standards based on 18000-7

To ensure co-existence of the revision of the protocol and previous version of protocol, a different protocol ID is selected as shown in Table A.1.

Since different application standards are developed based on the air interface of 18000-7, this standard needs to serve as the baseline.

All the different application standards will share the same physical layer protocols, but their command/response structure/field and command sets may vary depending on the application. Collection with UDB is a collection command. In this annex, we will use collection command and collection with UDB interchangeably. The two basic commands “collection” and “sleep” defined in this standard that shall be supported by all application standards. All the other commands shall be supported 18000-7 compliant products, but not necessary for other application standards compliant products.

When the Interrogator sends out a wake up signal, all tags based on 18000-7 air interface will wakeup.

Then the Interrogator may sends outs different commands as specified by the application. In the case that the Interrogator wants to find out all the active tags out there, it shall send out a collect command as defined in this standard. All the tags defined by the 18000-7 based application standards shall respond to this basic “collection” command. The tag respond with the collect response defined in its application data link layer standard. The tags shall also accept the “Sleep” commands defined in this specification.

Table A.1 — Protocol ID

Protocol ID	Standards
0x31	18000-7 version 0
0x40	18000-7 version 1
0x80	18185 eSeal standard
0xC0	17363 Shipment Tag

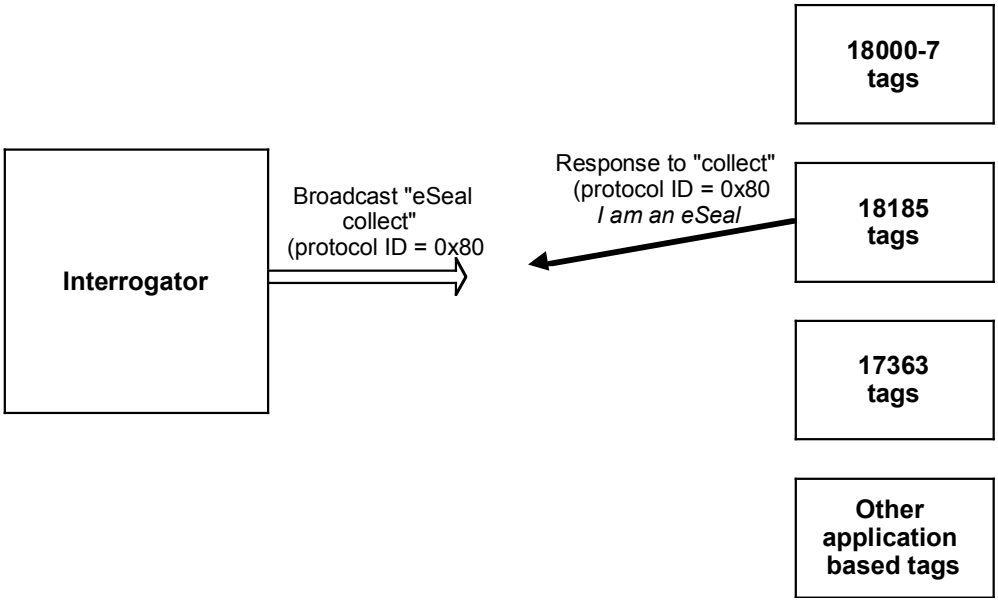


Figure A.1 — Broadcast / response for specific Protocol ID

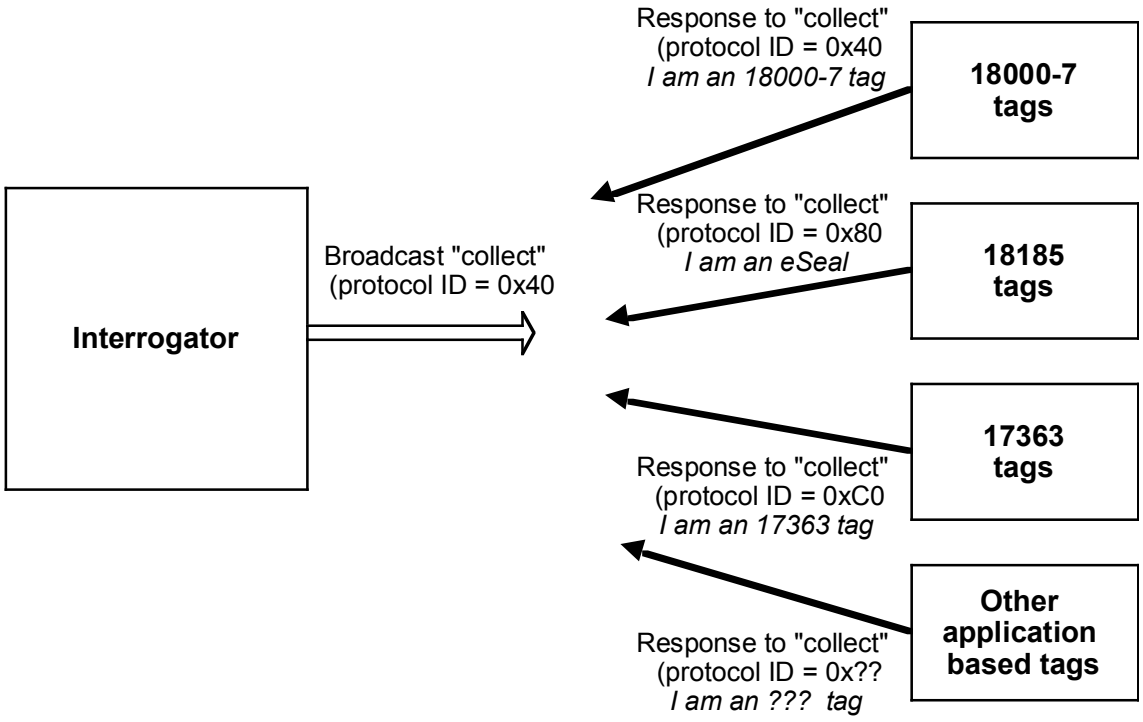


Figure A.2 — Coexistence of different application tags

Bibliography

[1] ISO/IEC Directives, Part 2: *Rules for the structure and drafting of International Standards*, 2001

[2] ISO/IEC 18000-1, *Information technology automatic identification and data capture techniques — Radio frequency identification for item management —Part 1, Reference architecture and definition of parameters to be standardized*

[3] ISO/IEC 18001, *Information technology — Automatic identification and data capture techniques — Radio frequency identification for item management — Application Requirements Profiles*

[4] ISO/IEC 18046, *Information technology — Automatic identification and data capture techniques — RFID device performance test methods*

[5] *CCITT Red Book,* Volume VIII, International Telecommunications Union, Geneva, 1986. *Recommendation V.41, "Code-Independent Error Control System*

[6] ERC/REC 70-03 - Relating to the Use of Short Range Devices (SRD), Annex 1, Band E; European Radio-communications Committee (ERC), August 2005

[7] US Code of Federal Regulations (CFR) Title 47, Chapter I, Part 15, Section 15.231. "Periodic operation in the band 40.66–40.70 MHz and above 70 MHz"; U.S. Federal Communications Commission, 1 Oct. 1999