International Post Corporation

IPC RFID STANDARD FOR INDENTIFYING POSTAL ITEMS BASED ON THE UPU S10 CODE, USING THE ISO/IEC 18000-63 PROTOCOL

Version 1.0

22 February 2017

IPC RFID standard for: Identifying postal items based on the UPU S10 code, using the ISO/IEC 18000-63 protocol

Version 1.0

Contents

Introduction

The UPU S10 code has been used to identify postal items for a number of years. Until now, the only means of data capture has been by representing the code in a bar code and capturing this with a scanner.

The need for increased automation and to capture data from multiple postal items almost simultaneously can be achieved by using RFID (radio frequency identification) using an acceptable standardised approach. This IPC standard provides such an approach for encoding information on RFID tags that increases the cost-effectiveness of the technology within postal services. Particularly, this will be through greater interoperability of RFID tags and equipment, and enhance support for resource sharing between postal services.

The encoding structures that have been adopted for this IPC standard have the potential to be used in a fully interoperable manner for other applications in future.

1 Scope

This IPC standard defines rules for encoding the UPU S10 code in radio frequency identification (RFID) tags. The tags and other artefacts shall comply with ISO/IEC 18000-63 (previously known as ISO/IEC 18000-6 Type C) operating in the UHF frequency. The encoding rules are based on ISO/IEC 15962, which uses an object identifier structure to identify those elements. The current edition of this IPC standard defines the rules for encoding a Unique Item Identifier in a specific Memory Bank known as MB 01. It also provides rules for encoding other relevant optional data in a separate Memory Bank known as MB 11. Each of these memory banks is addressable using the different commands of the appropriate RFID technology.

Rules are also defined for efficient selection of postal items using criteria that can be implemented in the RFID interrogator.

Although the encoding on the RFID tag is different from the encoding of the S10 in a linear bar code, the input and output formats are identical. This allows RFID data capture and bar code data capture to be interoperable and to work concurrently in the same system.

2 Normative references

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

UPU S10, *Identification of postal items – 13-character identifier*

IPC Interconnect Harmonised Labelling - Letter Package Label specifications

ISO/IEC 15962, *Information technology — Radio frequency identification (RFID) for item management — Data protocol: data encoding rules and logical memory functions*

ISO/IEC 18000-63, *Information technology — Radio frequency identification for item management — Part 63: Parameters for air interface communications at 860 MHz to 960 MHz Type C*

ISO/IEC 18046-1, *Information technology -- Radio frequency identification device performance test methods -- Part 1: Test methods for system performance*

ISO/IEC 18046-2, *Information technology -- Radio frequency identification device performance test methods -- Part 2: Test methods for interrogator performance*

ISO/IEC 18046-3, *Information technology — Radio frequency identification device performance test methods — Part 3: Test methods for tag performance*

ISO/IEC 18047-6, *Information technology — Radio frequency identification device conformance test methods — Part 6: Test methods for air interface communications at 860 MHz to 960 MHz*

3 Terms and definitions

3.1
access method
mechanism that declares the ISO/IEC 15962 encoding rules and formatting rules applied to encoded data

NOTE For this IPC standard, the term is only relevant to Memory Bank 11, containing optional data elements

3.2
air interface protocol
rules of communication between an RFID interrogator and the RFID tag of a particular type, covering: frequency, modulation, bit encoding and command sets

3.3
AFI
application family identifier
mechanism used in the data protocol and the **air interface protocol** to select a class of RFID tags relevant to an application, or aspect of an application, and to ignore further communications with other classes of RFID tags with different identifiers

NOTE For this IPC standard, the term is only relevant to Memory Bank 01, containing the data elements comprising the UII

3.4
arc
specific branch of an object identifier tree, with new arcs added as required to define a particular object

NOTE The top three arcs of all object identifiers are compliant with ISO/IEC 9834-1, ensuring uniqueness.

3.5
data format
mechanism used in the data protocol to identify how **object identifiers** are encoded on the RFID tag, and (where possible) identify a particular data dictionary for the set of relevant object identifiers for that application

3.6
DSFID
data storage format identifier
code that consists of, at least, the **access method** and **data format**

NOTE For this IPC standard, the term is only relevant to Memory Bank 11, which is currently used for encoding optional data elements.

3.7
MB
memory bank
designated name of a **segmented memory structure**

NOTE For the ISO/IEC 18000-63 tag the memory banks are: 00, 01, 10, and 11 using binary notation

3.8
logical memory
array of contiguous bytes of memory acting as a common software representation of the RFID tag memory accessible by an application and to which the object identifiers and data objects are mapped in bytes

3.9
object identifier
value (distinguishable from all other such values), which is associated with an object

3.10
S10 identifier
identifier that identifies postal items

3.11
S10 format
13-character string in the format of 2 alpha characters, 9 numeric characters, and 2 alpha characters

NOTE 1 The format is made up of four components, comprising of two characters for the service indicator, an 8-digit serial number, a check digit, and an ISO 3166 2-alpha country code.

NOTE 2 The format and permitted code is fully defined in UPU S10, Identification of postal items – 13-character identifier.

3.12
segmented memory structure
memory storage that is separated into separate elements and requires multiple addressing elements for access

NOTE This has the same meaning as partitioned memory, a term used in some documents.

3.13
UII
unique item identifier
encodable data when combined with an object identifier prefix that renders the combination unique within the rules of the application domain

4 Symbols and abbreviations

IEC	International Electrotechnical Commission
IPC	International Post Corporation
ISO	International Organization for Standardization
MHz	Mega Hertz
RFID	Radio Frequency Identification
UHF	Ultra High Frequency *NOTE For RFID this is 860 MHz to 960MHz*
UPU	Universal Postal Union
URN	Uniform Resource Name

5 RFID technology requirements

5.1 RFID air interface protocol

The air interface for compliant RFID tags and interrogators is specified in ISO/IEC 18000-63. There are different national and regional radio regulations for the use of RFID within the UHF frequency spectrum. It is essential to comply with such regulations, as follows:

— To meet with international requirements RFID tags should be able to operate between 860 MHz and 960 MHz, but shall comply with the national or regional requirements.

— RFID interrogators, or readers, shall operate at the nationally or regionally prescribed frequency within the 860 MHz to 960 MHz range. As a general guide:

  — Europe operates at the lower end: 865 MHz to 868 MHz

  — North America operates in the mid range: 902 MHz to 928 MHz

  — Japan operates at the upper end: 952 MHz to 958 MHz

More precise details are provided at http://www.gs1.org/docs/epcglobal/UHF_Regulations.pdf.

5.2 RFID tag

5.2.1 General tag features

ISO/IEC 18000-63 RFID tags have what is known as a segmented memory structure, where four different memory banks are supported and separately addressable. Using binary notation, the memory banks (MBs) are:

00	–	for passwords
01	–	for the unique item identifier
10	–	for tag identification, which can include serialisation

11 – for additional user data, which in the case of this IPC standard will include the optional data

Memory is organised in a 16-bit word unit for commands to read and write the data, but the actual memory structure is left to the chip manufacturer to decide on how this is implemented.

5.2.2 RFID tag memory parameter requirements

The ISO/IEC 18000-63 tag has four memory banks as described above. The following parameters are relevant to the tag specification relevant to this IPC standard:

— MB 00 is generally provided with memory capacity for the Kill password and Access password. Neither password is required for this IPC standard. If the tag supports passwords, then these shall remain at the default zero value.

— MB 01 is a mandatory requirement for ISO/IEC 18000-63 and shall have a minimum memory capacity to encode a UII of 96 bits.

— MB 10 is a mandatory requirement for ISO/IEC 18000-63. There is no requirement for the encoding by the IC manufacturer to be serialised, although this may be for some implementations.

— MB 11 should be provided to ensure the capability of encoding the optional data elements. The memory capacity required depends on the choice of data element to be encoded and the length of that encoding. A postal operator that opts to have an RFID without MB 11 risks losing the full benefits of this IPC standard. This memory bank should be capable of being selectively locked.

5.2.3 Declaring the memory parameters

ISO/IEC 18000-63 defines a number of parameters that are fixed, such as the fact that the unit for reading and writing is a 16-bit word. However, many features and parameters are left to the choice of the IC manufacturer. There is no air interface requirement to read a chip id as a basic part of the protocol to select and read the RFID tag. The 18000-63 tag has, in MB 10, a code that identifies the IC manufacturer (or designer) and model.

The memory requirements for this S10 standard are defined in Annex A.1. To achieve interoperability in postal operations, IPC has adopted a set of test methods for the performance requirements for passive UHF RFID tags to qualify for postal operations (see 5.6 and Annex A.2).

5.3 RFID interrogator (RFID reader)

RFID interrogators shall support all memory banks so that tags with three or four memory banks and different sized memory are all interoperable.

In order to achieve interoperability, RFID interrogators shall be based on open architecture RFID standards as defined in 5.5, 5.7 and 5.8. This means that any one manufacturer's reading/writing equipment shall be able to read or write to any other manufacturer's RFID tags, and that any manufacturer's RFID tags shall be able to be read and/or programmed by any other manufacturer's reader/writer.

5.4 Required air interface commands

Table 1 identifies the mandatory and optional commands that are requirements for RFID for item management applications and therefore for this IPC standard. Interrogators and tags claiming compliance with this standard shall comply with the item management requirements provided in the table.

Table 1 - Required commands and their codes

Function	Command code (binary)	ISO/IEC 18000-63 basic type	Required for this IPC standard
QueryRep	00	Mandatory	This is a RF level command and part of system set up.
ACK	01	Mandatory	This is a RF level command and part of system set up.
Query	1000	Mandatory	This is a RF level command and part of system set up.
QueryAdjust	1001	Mandatory	This is a RF level command and part of system set up.
Select	1010	Mandatory	This command is used to select tags by using the AFI for MB 01, and possibly the DSFID for MB 11. It is also used in the IPC standard for an interrogator level *rapid reading* procedure (see Clause 10)
Reserved	1011	N/A	
NAK	11000000	Mandatory	This is a RF level command and part of system set up.
Req_RN	11000001	Mandatory	This is a RF level command and used to communicate with a particular tag.
Read	11000010	Mandatory	This command is used to read 16-bit words from the nominated memory bank, unless the memory area is read-locked (e.g. passwords).
Write	11000011	Mandatory	This command is used to write a single word to a nominated address in a nominated memory bank. It is not possible to write to a locked word, and this means that writing to MB 10 is impossible at the application level.
Kill	11011100	Mandatory	This command is not required for tags to support this IPC standard. The command shall be supported by interrogators to enable interoperability with other IPC standards.
Lock	11000101	Mandatory	This command is use to lock or permalock the individual passwords, or the entire MB 01, or the entire MB 11.
Access	11000110	Optional	This command is not required for tags to support this IPC standard. The command shall be supported by interrogators to enable interoperability with other IPC standards.
BlockWrite	11000111	Optional	This command should be supported by interrogators, and may be supported by the RFID tag.
BlockErase	11001000	Optional	This command should be supported by interrogators, and may be supported by the RFID tag.
BlockPermalock	11001001	Optional	This command is used to selectively lock the encoding on the tag or to read the permalock status from the tag. The command can be applied to MB 01 and MB 11. The command shall be supported by interrogators, and should be supported by the RFID tag.

Although ISO/IEC 18000-63:2013 indicates that the *Kill* command can be used to re-commission a tag, this feature has been withdrawn from later editions of the air interface protocol.

NOTE No tag products are known to support this feature.

The use of the *BlockPermalock* command is not defined in this first edition of this IPC standard. However, interrogators shall support the command to achieve interoperability with other IPC standards or future editions of this standard. Because tags are single use for this IPC standard, they may be able to support the command.

5.5 Air interface conformance

The air interface conformance shall be tested in accordance with the procedures of ISO/IEC 18047-6.

5.6 Tag performance

Where there are requirements to test tag performance, these shall be done in accordance with ISO/IEC 18046-3. Additionally tags shall comply with the IPC set of test methods for passive UHF RFID tags (see Annex **Error! Reference source not found.**).

5.7 Interrogator performance

Where there are requirements to test interrogator (reader) performance, these shall be done in accordance with ISO/IEC 18046-2.

5.8 System performance

Where there are requirements to test system performance, these shall be done in accordance with ISO/IEC 18046-1.

5.9 RFID data protocol

The process rules of ISO/IEC 15962 shall be used to encode and decode data from the RFID tag. In particular, the following constraints shall apply:

- Encoding in MB 01 shall comply with the ISO/IEC 15962 rules for a Monomorphic-UII and specifically the URN Code 40 rules. Encoding in MB 01 is mandatory with the rules as defined in 8.4.

- Encoding in MB 11 shall comply with the No-Directory access method and be used for encoding the optional data elements defined in 8.7.

- MB 00 is intended for passwords. As the passwords are not required for this IPC standard, encoding in MB 00 shall not be implemented.

- No encoding is possible in MB 10.

MB 11 is defined as optional in ISO/IEC 18000-63, and therefore not all RFID tags include this memory bank. Increasingly MB 11 is incorporated in RFID tag products, so should be used to support the encoding rules defined in this IPC standard.

6 Data Protocol

6.1 Data protocol overview

The data shall be written to, and read from, the RFID tag using facilities functionally equivalent to the commands and responses defined in ISO/IEC 15961-1. The encoded byte stream on the RFID tag shall be encoded in accordance with the rules of ISO/IEC 15962. These rules are implemented automatically through a system that has both ISO/IEC 15961-1 and ISO/IEC 15962 as part of the complete data protocol.

6.2 Data constructs

6.2.1 Overview

ISO/IEC 15961-2 requires that a set of RFID data constructs be registered for applications that use the data protocol. The four RFID data constructs are described in 6.2.2 to 6.2.5, together with their particular code values that have been assigned by the ISO/IEC 15961 Registration Authority for use by IPC.

6.2.2 AFI

The AFI is a single byte code used as a tag selection mechanism across the air interface to minimize the extent of communication transaction time with tags that do not carry the relevant AFI code.

The AFI value **A0**$_{\mathrm{HEX}}$ has been assigned under the registration of ISO/IEC 15961-2 explicitly for use for IPC standards. This distinguishes postal items from all other items using RFID in item management systems. This avoids the risk of an RFID reader in another domain reading the RFID tag on a postal item and confusing the encoded content with data for its own application. It also enables a postal system to ignore items that carry a different AFI code or no AFI code (such as a GS1 EPC product code), possibly from a domain of a postal client (e.g. any content within a postal item).

The AFI is encoded in MB 01 (see 8.4.2). For this IPC standard, the AFI declares that the UII that is encoded in MB 01 is a Monomorphic-UII.

NOTE Unlike other ISO/IEC 15962 encoding schemes, Monomorphic-UIIs do not require the DSFID and some syntax to be encoded. All the requirements are declared by the AFI.

No other value of AFI shall be used in MB 01. This is to ensure that the rules registered for the data constructs according to ISO/IEC 15961-2 are consistently applied.

6.2.3 DSFID

If there is encoding in MB 11, the DSFID shall be encoded in the first byte. The DSFID has two component parts relevant to this IPC standard:

— the data format, as defined in 6.2.4, which is represented in the last five bits of the DSFID;

— the access method, which is represented in the first two bits of the DSFID, and which determines how data is structured in MB 11 on the RFID tag. The access method that is currently defined for this IPC standard 00_2 = No-Directory, where the encoded bytes are concatenated in a continuous byte stream.

Other access methods have been included ISO/IEC 15962. This IPC standard shall not support any additional access method without a formal amendment. Such an amendment shall include a migration path for the introduction and support of a new access method.

Locking the DSFID results in both the access method and data format being permanently set for the RFID tag. Any decision to lock or unlock the DSFID needs to consider the advice provided in 8.7.2.

6.2.4 Data format

The data format is used as a mechanism to enable object identifiers to be encoded in a truncated or short form. The data format value **14 (xxx01110$_2$)** has been assigned under the registration of ISO/IEC 15961-2 explicitly for postal use. The data format is part of a single byte value defined as the DSFID and defined in 6.2.3.

For this IPC standard, the DSFID, and therefore the data format, are only encoded in MB 11.

6.2.5 Object identifier for postal applications

The object identifier structure used in the RFID data protocol ensures that each data element is unique not only within a domain such as a postal system, but between all domains. The object identifier may be split into two component parts. The Relative-OID, as defined in 6.3, only distinguishes between data elements within a particular domain, whereas by prefixing this with a Root-OID the data element becomes unique within all object identifiers. The common Root-OID that has been assigned under the registration of ISO/IEC 15961-2 explicitly for IPC standards is:

1.0.15961.14

For all object identifiers specified in this IPC standard, only the Relative-OID will need to be encoded.

6.3 The URN Structure

The Uniform Resource Name provides a means for extending the use of RFID beyond the base data capture. It provides a means to use:

— the Internet to enable searches from any computer with the appropriate browser rules,

— various layers of RFID communication standards from the device interface to the application and data exchange layers.

The generic URN structure for IPC is:

urn:oid:1.0.15961.14.{IPCApplicationType}.{UniqueID}

1.0.15961.14 is part of the registration with ISO, and is not encoded in the RFID tag, but declared by the AFI. This is called the root-OID, and ensures that any IPC encoding in RFID tags and with the subsequent processing remains unambiguous.

The Relative-OID arc, value **A**, is assigned by IPC to distinguish RFID tags that encode the S10 code from any other IPC RFID application standard. The arc, value **A**, is re-created by the RFID decoding process.

The specific URN structure for this IPC RFID S10 standard is:

urn:oid:1.0.15961.14.A.{S10Code}

A postal operation may retain the UII in this format and add **1.0.15961.14.** as a prefix, or extract the S10 code depending on the business operation. Retaining the full OID structure, comprising of all arcs is useful where a system needs to distinguish between different OID structures or use resolver systems and other URN based systems. Extracting the S10 code achieves interoperability with bar code data capture and existing UPU message structures. Both approaches may be used in the same operation to meet particular system requirements.

7 Data elements

7.1 Unique item identifier (UII)

The unique item identifier (UII) is a mandatory data element to be encoded in Memory Bank 01 of an ISO/IEC 18000-63 RFID tag, which has a segmented memory structure. The UII shall be encoded using the rules defined in ISO/IEC 15962 for a Monomorphic-UII, which declares the Object identifier and encoding scheme directly from the AFI. Specifically, the encoding shall comply with the URN Code 40 encoding rules as defined in ISO/IEC 15962.

NOTE 1 The Relative-OID in the UII is part of the data payload and therefore does not need to be encoded separately, nor is a DSFID or precursor required for MB 01. However, these features are required for encoding in MB 11.

The UII for this IPC standard shall comprise these components:

— the IPC ApplicationType for S10 codes: the letter **A**,

— a ‘dot’ separator (the ‘dot’ is also known as a ‘full stop’ or ‘period’, ISO/IEC 8859-1 code point 2E$_{HEX}$),

— the S10 code.

NOTE 2 The ApplicationType is a mechanism that IPC can use to address other RFID applications and maintain full interoperability with this S10 standard.

The UII shall be locked to prevent various forms of digital vandalism and to ensure proof of source from a particular postal operator. The procedure for locking MB 01 is defined in Annex B.

7.2 Optional data elements encoded in MB 11

7.2.1 Alignment with the IPC Interconnect Harmonised Label

At the date of publication of this IPC S10 RFID standard, the Interconnect Harmonised Label mainly addresses bar code. However, the inclusion of a printed symbol to indicate the presence of an associated RFID tag implies that some alignment is possible and desirable. Furthermore, it enables a postal operator to implement RFID in a compatible manner.

The Interconnect Harmonised Label has a number of data elements that can be encoded in MB 11 of the RFID tag. These are defined in 7.2.2 to 7.2.5 and may be selected to meet the requirements of the postal operator or commercial e-seller producing the label and encoding the RFID tag.

The RFID tag may be part of the label stock, or a separate component at the choice of the original encoding organisation.

7.2.2 International delivery postal code

This is intended for international post. If encoded it shall be input in the format:

<ISO 3166-1 2-alpha country code> <domestic postal code of delivery address>

The Relative-OID **10** is assigned to the international delivery postal code.

EXAMPLES:
GB GL46RA
CA H1Y 1C7
US 63366-9700
US 90001
US NY 10118
AU NSW 2103

NOTE *A regional code, as in the last two examples may be included, but it might be redundant, as is the case for the United States.*

For encoding on the RFID tag, any space between the country code, regional code (if used) and the postal code should be omitted to reduce the use of memory. This also applies to the space between component parts of postal codes for Canada and the United Kingdom because the structure is unambiguously defined. Other characters should be included, such as the hyphen used in the Zip+4 code structure for the United States.

7.2.3 Associated documents

This optional data element is used to identify up to three documents associated with the postal item.

The Relative-OID **11** is assigned to the data element to identify associated documents.

Any of the 3-character codes from UPU Code List 184 may be encoded, although preference shall be given to any of the CN, CP, and EMS forms. The 2-character codes defined in UPU Code List 184 shall not be encoded.

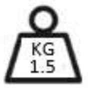
EXAMPLE 1: Based on UPU Code List 184, form CN 23 has the code **U23**

EXAMPLE 2: Assume that a parcel has a CN 22 customs declaration form as part of the label and includes a despatch note and invoice inserted by the e-seller. The encoding string would be:

U22750INV

7.2.4 Gross weight of item

Where the weight of an item is included on the Interconnect Harmonised Label, it may also be encoded on the RFID tag. On the label it appears as a symbol:



On the printed label the weight is always expressed in kilograms (kg) to one decimal place.

For encoding on the RFID tag the decimal point is removed to enable numeric encoding. For example, if the weight as shown in the symbol is 4.7 kg, then the value 47 (without any leading zeros) is input to the encoding process. All weights up to 999.8 kg can be encoded accurately. In the unlikely event of a greater weight, then the value 9999 is encoded.

The Relative-OID **16** is assigned to the data element to identify the gross weight.

7.2.5 Transport instructions

The Interconnect Harmonised Label has a number of symbols that convey transport instructions. Gross weight has already been addressed in 7.2.4.

An optional data element may be used to signal the instructions in a single byte code. This shall be encoded using the application-defined compaction rule, which is used for a bit-mapped code (see 0). If the label has a printed symbol, then the bit is set to **'1'** in the bit map. The structure is shown in Figure 1.

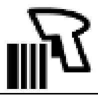
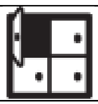
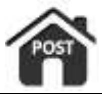
Printed symbol					Reserved for future assignments			
Instruction	Scan barcode	Signature required	Deliver to parcel locker	Deliver to pickup location				
Bit value when symbol is printed	1	1	1	1	0	0	0	0
Bit value when symbol is not printed	0	0	0	0	0	0	0	0

Figure 1 - Bit-map for transport instructions

EXAMPLE If a parcel is to be delivered to a pick up location and the barcode needs to be scanned, then the bit-mapped code is 10010000

The Relative-OID **17** is assigned to the data element to identify the bit map for the transport instructions.

7.2.6 Additional dedicated codes for e-sellers

The Interconnect Harmonised Label provides support for additional printed information for the e-seller. This principle is carried forward in this IPC RFID standard. Two Relative-OID values are assigned for the e-seller printing the label and encoding the RFID tag. The Relative-OIDs **126** and **127** are assigned for this purpose. The semantics of the data elements are only relevant to the e-seller and have no meaning in the postal services.

7.2.7 Additional dedicated code for the original postal processing centre

The data elements defined in 7.2.2 to 7.2.5 might not address all the requirements of the original postal processing centre. If additional data is required, Relative-OID **125** is assigned for this purpose.

The postal service may also request that, if possible, this data element is encoded by the e-seller when printing the label and encoding the RFID tag.

The semantics of this data element is only relevant to the original postal processing centre and has no meaning to other postal services.

7.2.8 Encoding and locking MB 11

The initial encoding of MB 11 of the RFID tag shall be aligned and synchronised with the production of the Interconnect Harmonised Label, whether the RFID tag is an integral part of the blank label stock or a separate entity. MB11 should be locked at this stage. However, by mutual arrangement, for example between e-seller original postal processing centre, the MB 11 may be left temporarily unlocked.

If the RFID tag is left unlocked during the initial label production, then the original postal processing centre may append any of the data elements defined in 7.2.2 to 7.2.5 and / or the code defined in 7.2.7 to the existing encoding. At this stage the tag shall be locked.

If an RFID tag is part of the Interconnect Harmonised Label, or an IPC RFID tag is associated with the postal item, then the label shall include the relevant symbol as shown in Figure 2.

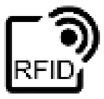


Figure 2 - RFID symbol

7.3 Summary of data elements assigned to this IPC standard

The set of data elements supported by this IPC standard is outlined in

Table 2.

Table 2 – List of data elements

Relative OID	Name of the data element	Encoded in MB	Encoding Status	Display format / Comments
A	S10 code	01	Mandatory	A.{S10 code}
10	International delivery postal code	11	Optional	<ISO 3166 2-alpha country code> <domestic postal code of delivery address>
11	Associated documents	11	Optional	Based on up to three codes from the UPU Code List 184
12 - 15				Reserved for other IPC applications
16	Gross weight of item	11	Optional	Weight in kg up to 999.8, encoded as a numeric string without the decimal point
17	Transport instructions	11	Optional	Bit-map to align with transport instruction symbols
125	Postal service Internal use	11	Optional	This is a free format data element for use by the original postal processing centre
126	E-seller internal use	11	Optional	This is a free format data element for use by the e-seller
127	E-seller internal use	11	Optional	This is a free format data element for use by the e-seller

8 ISO/IEC 15962 encoding rules

8.1 General

The memory of an ISO/IEC 18000-63 tag is divided into four memory banks as defined in 5.2. Three of the memory banks can be encoded, whereas MB 10 is written to by the manufacturer of the integrated circuit and thereafter is read-only.

Memory is organised in 16-bit words, and a word is the minimum unit that can be written to the tag or read from the tag. Commands are addressed in word number starting at 0_{HEX}. However, some of the structures of memory are defined as bit locations with the first bit in each memory bank identified as 00_{HEX}.

There are no standard air interface commands to determine which words are locked; ISO/IEC 18000-63 simply states "A Tag’s lock bits cannot be read directly; they can be inferred by attempting to perform other memory operations."

The logical memory is the software equivalent of the structure of the memory on the RFID tag itself. It is a mechanism used in ISO/IEC 15962 to represent all the encoding for a tag, including processes that need to be implemented for locking or selectively locking data. Once structured, the content of the logical memory can be passed to air interface protocol commands as the data ‘payload’ or to invoke other actions, like locking.

The following clauses identify the structure and rules as applicable for this IPC standard.

8.2 Structure of MB 00

8.2.1 Supported passwords

This memory bank is used to store passwords. The 32-bit Kill password is stored at locations 00_{HEX} to $1F_{HEX}$. The un-programmed value of this password is a 32-bit zero string. An interrogator can use the Kill password to kill a tag and render it unresponsive thereafter.

The 32-bit Access password is encoded at location 20_{HEX} to $3F_{HEX}$. The default un-programmed value is a 32-bit zero string. A tag with a non-zero Access password requires the interrogator to issue this password before subsequent processing with the tag memory.

8.2.2 Kill password

Because the Kill password is not required for normal postal processing, it shall not be used in this IPC standard.

8.2.3 Access password

The Access password is not required for this IPC standard. This is because selective locking of data elements using the *BlockPermalock* air interface command is not supported in this edition of this IPC standard.

8.3 Structure of MB 01

This memory bank contains the UII and associated syntax. The first word at memory address location 00_{HEX} to $0F_{HEX}$ contains a stored CRC-16. This is automatically generated when the tag is processed and the rules for that are beyond the scope of this IPC standard. The second word contains a protocol control word at memory locations 10_{HEX} to $1F_{HEX}$ as shown in Table 3, which shows the encoding for this IPC standard in the last row.

Table 3 - Structure of Protocol Control Word

Protocol Control Word bits 10_{HEX} to $1F_{HEX}$															
Length indicator					UMI	XPC	NSI	ISO Application Family Identifier (AFI)							
10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F
0	0	1	0	0	0 = not used 1 = encoded	0 (N/A)	1 (ISO)	1	0	1	0	0	0	0	0

The structure is significant and relevant to this IPC standard as follows:

— A UII length field is encoded in memory locations 10_{HEX} to 14_{HEX}. The value shown in Table 3 represents the length of the encoding for the UII. The value of the length field should be calculated automatically as defined in ISO/IEC 18000-63.

— A user memory indicator (UMI) is held in location 15_{HEX}. The different editions of ISO/IEC 18000-63 result in different ways that this may be set:

  — The tag manufacturer sets this to **0** when there is no MB 11 present on the tag, or to **1** when MB 11 is available for encoding. This option was introduced in the ISO/IEC 18000-63:2015 edition.

  — It can be set by the tag, based on encoding being successfully encoded in MB 11. This option has been in various editions of ISO/IEC 18000-63.

  — It can set by the interrogator. Some interrogators might require the value to be input from the application. This option has been in earlier editions of ISO/IEC 18000-63.

NOTE An additional point to consider is the different development procedures and lifespans of tags and interrogators. Older interrogators might not support the latest tag features.

Annex A.1 provides information of the capability of different ISO/IEC 18000-63 tags compliant with IPC RFID standards.

— An extended protocol control indicator (XPC) is stored in location 16_{HEX}. The function of this bit is beyond the scope of this IPC standard. If used in an IPC standard in future, it would be calculated automatically as defined in ISO/IEC 18000-63.

— A numbering system identifier (NSI) is encoded in memory location 17_{HEX}. This shall be encoded with the value '1' to indicate that the following eight bits are the AFI.

— Bit locations 18_{HEX} to $1F_{HEX}$ shall encode the AFI with the bit values **10100000_2.**

The encoding of the Unique Item Identifier starts at bit location 20_{HEX}. Encoding and decoding needs to be invoked for complete 16 bit words. The value of the UII length field (in memory locations 10_{HEX} to 14_{HEX}) is generated automatically as defined in ISO/IEC 18000-63.

8.4 Encoding in MB 01

8.4.1 Components

MB 01 encodes the AFI and the UII. The encoding rules for these two components are defined in the following sub-clauses. Although shown separately the encoding should be implemented in one action.

8.4.2 Encoding the AFI

The AFI is encoded as part of the protocol control word in bit locations 18_{HEX} to $1F_{HEX}$. It shall be preceded by a '1' in location 17_{HEX} to enable tags encoded to ISO rules to be distinguished from those encoded to GS1 EPC rules.

NOTE An e-seller can use RFID tags using GS1 EPC encoding rules. The assignment of a specific AFI for IPC standards means that other tags can be ignored.

In the absence of any more specific procedures for a creating air interface commands, the 16-bit string defined in Table 3, shall be used to construct the encoding of MB 01 bit positions 10_{HEX} to $1F_{HEX}$.

8.4.3 Encoding the UII

The Monomorphic-UII shall be encoded using the URN Code 40 encoding rules as defined in Annex C. The AFI declares the encoding scheme. In turn this means that a DSFID does not need to be encoded in MB 01. The URN Code 40 encoding rules support variable length input without requiring a length to be encoded. However, given the fixed length of the S10 code, the length will be the same for all tags.

These are the encoding steps:

1. Because a Monomorphic-UII is used, the encoder input is A.{S10Code}, noting that {S10Code} is replaced by the 13-character S10 code

2. Submit the resultant character string to the URN Code 40 encoder. The encoder takes as input a 3-character string and converts it to a 16-bit string. The process is repeated until the encoding is completed.

3. The resultant byte string is encoded from bit location 20_{HEX}. Because URN Code 40 encoding is always over 16-bit units, it is already aligned with the 16-bit word boundary of MB 01. As the S10 code is 13 characters long, and is preceded by the two characters '**A.**', the 15-character string is encoded in five 16-bit words ending at bit location $6F_{HEX}$.

EXAMPLE	Basic structure:	A.{S10Code}	
	Step 1:	**A.RY013000415CH**	
	Step2a:	A.R encodes as	0000101010110011_2 = $0AB3_{HEX}$
	Step2b:	Y01 encodes as	1010000100010000_2 = $A110_{HEX}$
	Step2c:	300 encodes as	1101001100001111_2 = $D30F_{HEX}$
	Step2d:	041 encodes as	1100000011110000_2 = $C0F0_{HEX}$

Step2e: 5CH encodes as 1101101101000001_2 = $DB41_{HEX}$

Step3: The resultant byte string is: $0AB3A110D30FC0F0DB41_{HEX}$

An expanded version of this example, showing the detailed calculations for each 16-bit string is given in Annex C.2.2.

8.4.4 Rules for writing and locking MB 01

MB 01 does not support any form of selective locking, therefore the entire memory bank shall be locked. This ensures that the tag is in a read-only state with the UII being protected from accidental or deliberate changes.

There are no commands to determine if MB 01 is locked (see 8.1).

The procedure to lock MB 01 does require some precision in the sequence of commands to encode data on the tag. For example, for many tags the encoding in MB 11 needs to be started so that the UMI bit can be set automatically by the tag. Only after all of the relevant encoding has been done should any attempt be made to lock MB 01.

The structure of MB 01 and the locking functionality has some implications for this IPC standard:

— If the AFI is encoded prior to the encoding of the UII, MB 01 shall not be locked at this stage to avoid the tag being rendered useless.

— If any of the optional data elements are to be encoded in MB 11, this shall first be encoded to ensure that the user memory indicator (UMI) in location 15_{HEX} is correctly set.

— If no encoding of optional data elements is ever expected, then the encoding of all the data in MB 01 is completed and locked as defined in Annex B.

8.5 Structure and use of MB 10

MB 10 (also known as TID memory) encodes information that identifies the manufacturer or designer of the integrated circuit and the model number. These can be used to provide some information about the tag's capability.

ISO/IEC 18000-63 compliant integrated circuits that have been developed more recently can also include a serialised component in the TID.

Once the integrated circuit manufacturer has encoded the TID, it is generally locked and therefore is in a read-only state.

8.6 Structure of MB 11

This memory bank contains the optional data elements and associated syntax. The first byte at memory address location 00_{HEX} to 07_{HEX} contains the encoded DSFID with the value **00001110_2**. The next byte beginning at memory address location 08_{HEX} encodes the precursor of the first data set.

As with the other memory banks, the basic *Lock* air interface protocol command only supports locking of the entire memory bank.

For this initial edition of this IPC S10 standard, the *BlockPermalock* command shall NOT be used.

There are no commands to determine if MB 11 is locked or selectively locked (see 8.1).

The DSFID can be read without reading any other data encoded in MB 11, by invoking an air interface *Read* command and set this to only read the first word.

8.7 Encoding in MB 11

8.7.1 General MB 11 rules

The encoding rules are designed to achieve a combination of flexibility and efficiency for the bytes that are encoded on the RFID tag. In particular:

— data is compacted efficiently using a defined set of compaction techniques that reduce the encoding on the RFID tag and across the air interface;

— data formatting minimizes the encoding of the object identifiers on the RFID tag and on the air interface, but still provides complete flexibility for identifying specific data without the recourse to rigid message structures.

The syntax associated with the encoding rules effectively creates a self-defining message structure within MB 11. This allows optional data from the application data dictionary to be selected. It also enables variable length data to be encoded efficiently, and for different formats of data (e.g. numeric or alphanumeric) to be encoded as efficiently as possible and intermixed in the same RFID system. The rules of ISO/IEC 15962 make it possible to correctly interpret the data on the RFID tag without any prior knowledge of what is encoded on the tag. This is an important feature that enables interoperability of devices, and allows this IPC standard to add new data elements in future without changes to the equipment. It also allows an individual postal operator to vary the choices of data elements without the need for any major update.

These are the requirements and recommendations for particular features:

— All interrogators shall support the encoding and decoding of MB 11.

— Any encoding and / or decoding software shall support all the data elements defined for MB 11.

A postal operator or commercial e-seller may choose to use tags that support MB 11. In turn, the postal operator and e-seller may choose which data elements from

Table 2 to encode in MB 11, and choose the sequence in which they are encoded.

8.7.2 Encoding sequence

The DSFID (see 8.7.3) shall be encoded as the first byte of MB 11. The DSFID shall not be encoded on its own because it will either have to be overwritten (because ISO/IEC 18000-63 tag encodes a minimum of a 16-bit word), or require that word to be locked and probably others to be locked.

The first data set (see 8.7.5) shall begin its encoding in the second byte of MB 11 (i.e. the second half of the first word). If the Relative-OID of a data set is in the range 1 to 14 it is encoded according to the rules of 8.7.6. If the Relative-OID is in the range 15 to 127 it is encoded to the rules of 8.7.7.

The data sets may be encoded in any sequence at the choice of the original encoding organisation.

8.7.3 Configuring the DSFID

The DSFID consists of two components:

— the access method;

— the data format.

The data format is specified in 6.2.4 and the access method is defined in 6.2.3. These bit values are combined to create the appropriate DSFID byte value as shown in Table 4.

Table 4 - Relevant DSFID value

Bit sequence			DSFID byte
Access method[a]	**Reserved**	**Data format**	
00	0	01110	0E

[a] 00 = No directory, where the encoded bytes are concatenated in a continuous byte stream.

8.7.4 Data compaction

The data elements in

Table 2 are subjected to the standard ISO/IEC 15962 compaction. When the instructions are sent to an ISO/IEC 15962 compliant encoder to compact the data, it automatically selects the most efficient compaction scheme for each data element presented. It also enables shorter codes to be represented (generally) in fewer bytes. This allows alphanumeric or numeric code structures to be used flexibly. UTF-8 may be supported for Relative-OIDs 125 to 127, with the only penalty being that more complex character sets require more encoding space on the RFID tag.

The exceptions to this automatic selection rule are:

- that as Relative-OID 17 for the transport instructions (see 7.2.5) is bit based, the application shall declare that the Application-defined compaction scheme is being used. This ensures that the bit structure remains consistent up to the point of decoding.
- that if Relative-OIDs 125 to 127 are encoded using UTF-8, the application shall declare that the UTF-8 string compaction scheme is being used. This ensures that the UTF-8 string remains unchanged up to the point of decoding.

The compaction schemes are identified on the RFID tag by a 3-bit code which is included as part of the precursor (see 8.7.6). The full set of compaction schemes and their code is shown in Table 5.

Table 5 - ISO/IEC 15962 No-Directory compaction schemes

Code	Name	Description
000	Application-defined	As presented by the application
001	Integer	Integer
010	Numeric	Numeric string (from “0” to “9”)
011	5-bit code	Uppercase alphabetic
100	6-bit code	Uppercase, numeric, etc.
101	7-bit code	US ASCII
110	Octet string	Unaltered 8 bit (default = ISO/IEC 8859-1)
111	UTF-8 string	External compaction to ISO/IEC 10646

8.7.5 General rules for creating the encoded data set(s)

The encoding of the Relative-OID and data object on the RFID follows a particular sequential structure defined in ISO/IEC 15962. The next two sub-clauses define the basic rules that are relevant to this IPC standard.

NOTE ISO/IEC 15962 defines other rules, for example for encoding full object identifiers, which are not relevant to this IPC standard.

8.7.6 Data set for Relative-OID value 1 to 14

NOTE Some Relative-OID values in the range 1 to 14 are reserved for identifying the IPCApplicationType encoded as part of the UII.

The structure of an encoded data set with the Relative-OID value 1 to 14 consists of the following components:

- a precursor, i.e. a single byte that in this case encodes the compaction scheme and the Relative-OID;
- the length of the compacted data object;
- the compacted data object.

This structure is shown in Figure 3.

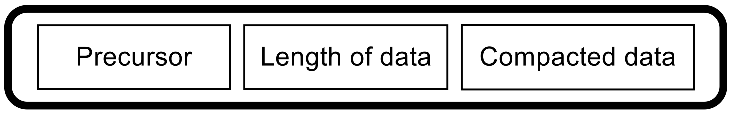


Figure 3 - ISO/IEC data set with Relative-OID values 1 to 14

Some of the data elements defined in this IPC standard have Relative-OID values (1 to 14). These are directly encoded in the precursor (see Table 6), and this reduces the amount of memory required for the encoding.

Table 6 - Bit position of precursor components

Precursor bit positions							
7	6	5	4	3	2	1	0
Offset	Compaction code			Object identifier			

The offset bit in the precursor is only set to "1" if an offset byte is encoded on the RFID tag. The offset byte is not relevant to this IPC S10 standard, therefore shall be set to "0".

NOTE The offset byte is used as part of the procedure to block align data where selective locking is applied to MB 11. Selective locking is not supported in this edition of this IPC standard.

8.7.7 Data set for OID value 15 to 127

The precursor only provides 4 bits for encoding the object identifier. It is only capable of directly encoding Relative-OID values from 1, which encodes as 0001_2, to 14, which encodes as 1110_2. For Relative-OID values between 15 and 127 the last four bits of the precursor are set = 1111_2. This bit string signals that the Relative-OID has to be explicitly encoded as a separate component (a single byte) in the data set, as shown in Figure 4.

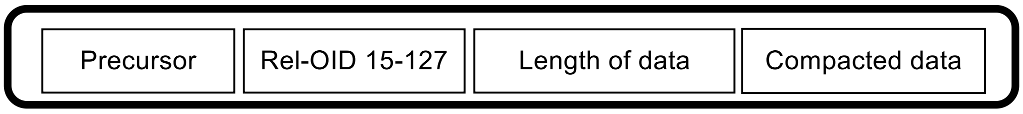


Figure 4 - ISO/IEC data set with Relative-OID values 15 to 127

The value that is encoded for the Relative-OID is the value offset by –15. This means that Relative-OID 15 is encoded as 15-15 = 0 = 00_{HEX}. The highest Relative-OID that can be encoded in this manner is Relative-OID 127, encoded as 127-15 = 112 = 70_{HEX}.

8.7.8 Selecting the data elements to encode

The encoding organisation should decide on the number and sequence of data elements to encode. The only guidance is that the more relevant data elements should be encoded before the less relevant ones. This is to enable a tag with a given MB 11 memory capacity to be selected. This is particularly important if the data elements are of variable length where it might be difficult to predict the memory required. This way, if there is insufficient memory the less significant data elements will not be encoded. An encoding example for MB 11 is shown in Annex D.

8.7.9 The logical memory for MB 11

Irrespective of whether one data set or multiple data sets are to be encoded in MB 11 the encoded bytes are formatted in the logical memory in a structure that is compliant with the specific tag architecture. Because the size of memory differs between manufacturers and even between model versions, this formatting is an essential feature of the encoding rules to achieve interoperable RFID tags. This enables any RFID tags to be considered as candidates for encoding to this IPC standard that can claim compliance with ISO/IEC 18000-63, but differ between each other within the options permitted by the air interface standard.

EXAMPLE 1 RFID tags may have different sizes of memory for MB 11.

EXAMPLE 2 Some tags are able to transfer multiple blocks across the air interface in write and read transactions, others only to transfer single blocks.

A compliant ISO/IEC 15962 encoding process shall:

- identify the memory capacity of MB11;
- report as an error if the encoding requirement exceeds the available memory capacity.

Once the logical memory has been populated, single or multiple words are written across the air interface.

When reading data from the RFID tag, the logical memory is populated word by word. Decoding of an RFID tag with a No-Directory access method is done in the sequence in which the Relative-OIDs are presented, but the data object only needs decoding if its Relative-OID is selected by the application command.

8.7.10 Locking MB 11

This clause defines the procedure to follow to lock the DSFID and a contiguous sequence of data sets. In other words:

DSFID {1st data set} (2nd data set} {3rd data set} ... {final data set}

This first edition of this IPC standard only supports locking all of MB 11. This is achieved during label production or may be postponed until the original postal processing centre first handles the item (see 7.2.8).

The air interface *Lock* command shall be used to lock MB 11.

9 Decoding to ISO/IEC 15962 rules

9.1 Decoding MB 01

9.1.1 AFI

The AFI should be used as part of the air interface *Select* command, and as such is not read. Only tags with the IPC assigned AFI **$\mathbf{A0_{HEX}}$** will respond. Tags with other AFI values, or with no AFI value (e.g. with data encoded to GS1 EPC rules) will be ignored.

If there is a requirement to read the AFI, e.g. for diagnostic purposes, then the protocol control word is read. An RFID tag that conforms to this IPC standard has its last 9 bits with this value: **$\mathbf{110100000_2}$.** The leading ‘1’ bit is required to indicate that the following 8 bits are the AFI.

9.1.2 Decoding and processing the Monomorphic-UII

Decoding the URN Code 40 byte string is achieved by taking each 16-bit word from the encoded Monomorphic-UII and converting it to characters. The UII for this IPC standard is always 80 bits long, which means that five 16-bit words will be returned.

The first 16-bit word should be retained in a binary or hexadecimal format to check that it is a valid S10 code conforming to this IPC standard.

- In hexadecimal, the first byte of the first word shall be $0A_{Hex}$.
  - If so, proceed with the decoding
  - If not, the encoding is wrong

- In binary, the first 8-bits of the first word shall be 00001010_2.
  - If so, proceed with the decoding
  - If not, the encoding is wrong

Once the UII has been shown to be valid, decode each of the five words in succession, using the inverse of the URN Code 40 rules defined in Annex C.

The following additional validity checks may be applied to the decoding. If applied to each word, then the decoded:

- first word is always: alpha, '.', alpha.
- second word is always alpha, numeric, numeric.
- third word is all-numeric.
- fourth word is all numeric.
- fifth word is numeric, alpha, alpha.

If the validity check is applied to the entire decode, then the decoded string is:

A.{alpha}{9 numeric}{2 alpha}
where the first two characters are always the letter 'A' followed by the 'dot' character, and the characters in {} comply with the S10 code structure.

9.1.3 S10 bar code interoperability

To maintain interoperability with S10 bar code and message structures, the process shall strip off the leading two characters '**A.**'

9.1.4 URN interoperability

To maintain interoperability with a URN structure, the 15-character string is retained and shall be prefixed by **1.0.15691.14.** (note that the final 'dot' is required), thus creating a unique object identifier.

9.2 Decoding MB 11

9.2.1 DSFID

The DSFID is encoded as the first byte in MB 11. Therefore it will only be encoded if there is other encoding in MB 11. It can only be read by using an air interface *Read* command to read either the first 16-bit word in MB 11, or by processing this word in a *Read* command designed to return multiple words. The DSFID shall have the value **$0E_{HEX}$**, or **00001110_2** if the processing is carried out at the bit string level. Tags with other DSFID values shall be ignored as they do not conform to the IPC standard. So too, shall any subsequent encoding.

The first two bits declare the No-Directory access method for the remainder of the encoding. The last five bits declare that the data complies with this IPC standard. The third bit is not relevant to this IPC standard.

9.2.2 Decoding a No-Directory data set

NOTE This clause defines decoding rules for this IPC standard. These rules are a perfect subset of the ISO/IEC 15962 rules, which are more complex to address features not required for this IPC standard. A compliant ISO/IEC 15962 decoder can support all the rules below. The following rules are presented for developers addressing this IPC standard.

Each data element is encoded in a data set as defined in 8.7.6 and 8.7.7. The first data set shall abut the DSFID and its encoding shall start in the second byte (or from bit location 08_{HEX}) of the first word in MB 11.

The first byte of any data set is the precursor (see 8.7.6), whatever the Relative-OID value. Table 6 shows the structure of the precursor. It shall be decoded into its component parts, and there is an advantage of processing in this sequence.

- The Relative-OID value 1 to 14 is directly encoded in bit positions 3 to 0, or the second hexadecimal character of the precursor.
  - If the value is 0000_2 (0_{HEX}), then the encoding is in error.
  - If the value is 1111_2 (F_{HEX}), then the Relative-OID is greater than 15.
  - Otherwise the Relative-OID is the decimal equivalent of the binary value.
- The compaction code (see Table 5) is encoded in bit positions 6 to 4.
- The offset bit is encoded in bit position 7. For this edition of this IPC standard, the offset bit = $\mathbf{0_2}$.

If the precursor indicates that the Relative-OID is in the range 15 to 127, then the byte immediately following the precursor declares the value of the Relative-OID. The hexadecimal value is converted to decimal, and 15 is added to the value. Thus value 00_{HEX} = 0 +15 = Relative-OID 15.

For both data set structures as defined in 8.7.6 and 8.7.7, the length byte declares the length of the compacted data. If the length byte has a value greater than $7F_{HEX}$, then the encoding is in error.

The compacted byte string is decoded using an inverse process to the encoding process declared in Table 5 and fully defined in ISO/IEC 15962. The exception to this rule is that Relative-OID **17**, which encodes the transportation instructions. As this is a bit-based code it is transferred to the application layer for interpretation.

9.2.3 The end of encoding

Decoding proceeds until all data sets are decoded. The end of encoding is signalled by a byte value 00_{HEX} where the next precursor would otherwise be expected. A robust decoding procedure may check for a short series of bytes with the value 00_{HEX}, possibly no more than one or two 16-bit words.

10 Rapid reading of the UII

10.1 Overview

The following clauses define a set of methods that can be used to rapidly read various parts of the UII from the tag at the interrogator layer, without the requirement to decode that data. The methods make use of the air interface *Select* command to read and process particular strings of 'raw' bits.

10.2 Structure and purpose of the *Select* command

The ISO/IEC 18000-63 air interface *Select* Command is issued at the beginning of an inventory round.

A ***Select*** command can be used prior to a ***Query*** command to specify which tag population to select. It may be all IPC tags to include test tags, receptacle assets and other postal products; or it could be a subset of a particular IPC standard as defined in the following sub-clauses. Only those tags that hear the ***Select*** commands and meet the criteria defined by the ***Select*** command(s) can be instructed to participate or not participate in an inventory round. This means that not only does the ***Select*** command focus on the tags required at a given data capture point, but also that all other tags that do not fulfil the requirements are effectively ignored.

The following sub-clauses define the ***Select*** command parameters for this IPC receptacle asset standard. A technical explanation of this command is provided in Annex E.

10.3 Fast select

This method is used to select RFID tags with S10 codes from any others. This method excludes tags from all other domains and tags encoded to conform to other IPC standards, some of which have still to be developed.

The *Select* command shall be constructed with 13 bits from bit positions 17_{HEX} to 23_{HEX} of MB 01 with the Mask value $\mathbf{1101000000000_2}$ as part of the command structure defined in Table 7. This represents the bit that identifies the encoding as being complaint to ISO rules, the AFI assigned to IPC and the first 4 bits of the 16-bit word that identifies that this is a S10 code.

Table 7 - *Select* command parameters for S10 codes

	Command	Target	Action	MemBank	Pointer	Length	Mask	Truncate	CRC-16
# of bits	4	3	3	2	EBV	8	13	1	16
description	1010	100	001	01	00010111	00001101	1101000000000	0	

NOTE ‘EBV’ stands for Extended Bit Vector, which is a method used in ISO/IEC 18000-63 tags to be able to encode and bit string in a self declaring manner. In this case each 8-bit block comprises of the lead or extension bit followed by 7 bits that represent the numeric value. If the extension bit = 0 then it is the last block. If the extension bit = 1, then it is followed by another block.

EXAMPLES	decimal 127	01111111
	decimal 128	10000001 00000000
	decimal 16384	10000001 10000000 00000000

Invoking the *Select* command results in the selection of all tags that conform to this IPC S10 standard. With the *Select* set with these parameters, the complete UII is returned.

10.4 Select to Service Indicator

The Service Indicator comprises the first two alphabetic characters of the S10 code. However for many Service Indicators, just the first character is sufficient to distinguish between major product types.

In situations where it is important to identify items belonging to a particular class of Service, the *Select* command shall be constructed with 25 bits from bit positions 17_{HEX} to $2F_{HEX}$ of MB 01 with the Mask value as defined in Table 8 as part of the command structure defined in Table 9.

Table 8 - Mask values for different types of product

Type of product	Service indicator 1st character	Mask value Bit string
EMS	E	1101000000000101010100110
Insured letter post	V	1101000000000101010110111
Registered letter post, not insured	R	1101000000000101010110011
Recorded delivery, neither registered nor insured	A	1101000000000101010100010
Express letter post	L	1101000000000101010101101
Items other than the above, but subject to custom control, i.e. bearing a CN 22 or CN 23	U	1101000000000101010110110
Parcel post	C	1101000000000101010100100

Table 9 - *Select* command parameters for particular product types

	Command	Target	Action	MemBank	Pointer	Length	Mask	Truncate	CRC-16
# of bits	4	3	3	3	EBV	8	23	1	16
description	1010	100	001	01	00010111	00011001	See Table 8	0	

This results in the selection of all tags that conform to the particular service indicator code. With the *Select* set with these parameters, the complete UII is returned.

10.5 Reading the UII as a raw bit string

There can be situations where there is an opportunity to avoid decoding the UII and simply read the UII as a binary string or hexadecimal string (depending in the output of the interrogator). This method can be used where the RFID tag on item is certain to conform to this IPC standard, e.g. by invoking the methods in 10.3 or 10.4. These methods deliver the raw UII.

As an example in a sortation system: the raw UII is captured, then decoded, then a routing decision is assigned to the item. The application database will assign the decision to the decoded UII (i.e. the S10 code or the object identifier), then that decision can also be associated with the raw UII. Any subsequent reading of the RFID tag only requires the raw UII to be matched with the decision.

Annex A(informative) -- Information about tag compliance

A.1 Memory requirements

These are the requirement guidelines to fully support this IPC S10 standard:

MB 00 This memory bank support passwords in ISO/IEC 18000-63 compliant products. The Access password and Kill password are not required for this standard.

MB 01 Memory capacity is measured from bit position 20_{HEX}. This memory bank encodes the UII, i.e. the S10 code. Although this only requires 80 bits of memory, most tags support at least 96 bits of UII memory. Generally unused memory cannot be used, but the length indicator in the protocol word is used to restrict air interface transmission to the data that is encoded.

MB 10 This memory bank contains details of the chip manufacturer, model number and often a unique serial number. IPC has a list of compliant tags identified by manufacturer / model number. The memory bank in not generally required for this IPC S10 standard, except to confirm compliance, and to possibly use the serial number for diagnostic purposes.

MB 11 Tags with this memory bank are recommended. The encoded example in Annex D requires 240 bits. Other configurations of data elements will require less or more memory.

A.2 Performance requirements

IPC has established an RFID tag conformance standards, to which all tags shall conform. It also maintains a list of tags that meet to performance requirements. More details can be obtained from rfid@ipc.be.

Annex B(normative) -- Locking procedure for MB 01 with encoding in MB 11

B.1 Overview

This annex defines two procedures based on how the UMI bit (bit position 15_{HEX}) is set in the protocol control word of MB11. It is therefore necessary to establish the tags features by using the resources defined in Annex A.1.

B.2 UMI set by the chip manufacturer

In this case the manufacturer of the integrated circuit (RFID chip) sets the UMI to'1' if the tag has encoding capacity in MB 11. All this does is declare that memory is present.

Once the fact has been established that the UMI is pre-encoded, if the intention is to encode data in MB 11 then it simply means that the sequence of invoking the write command for each memory bank is not critical.

NOTE Setting the UMI this way has an impact on the reading process. Reading a UII from MB 01 no longer provides information that data is actually encoded in MB 11.

B.3 UMI set by the tag or interrogator

In this case the tag has an internal automatic process that sets the UMI to '1', based on the value of the DSFID in MB 11.

Previous editions of ISO/IEC 18000-63, and GS1 EPC Class 1 Gen 2, provided a method where the interrogator and not the tag created the UMI. In other words the bit UMI bit had to be included as part of the write command. As there are tags and readers in circulation that support the previous editions (as recently as 2013), the following procedure addresses both options.

1. Write the DSFID in MB 11. If the other input details are known, other data sets may be encoded at the same time.
2. By mutual agreement between an e-seller undertaking the initial encoding and the local postal service, MB11 may be left unlocked until that postal service appends any additional data elements.
3. Consider locking the DSFID and any data elements using the *Lock* air interface command. This will lock all of MB 11 and will make it impossible to make any future changes to the encoding.
4. Create the bit string for the Protocol Control word starting at bit location 15 with {10110100000} to encode the UMI bit, the NSI bit and the AFI (see 8.3). This is the safer of two options defined in previous editions of ISO/IEC 18000-63 (and GS1 EPC Class 1 Gen 2) because it supports tags that do not automatically generate the UMI bit. A tag that does will generally over-write the lead '1' bit in the string with an automatically generated '1' bit.
5. Create the UII.
6. Append the UII (step 4) to the Protocol Control word bits (step 3) and write to MB 01. It might be necessary to pad with five leading bits {00000} to complete the structure of the Protocol Control word.
7. Lock MB 01.

NOTE: Although the length of the UII is both known and constant, the rules of the ISO/IEC 18000-63 protocol result in these length bits being determined automatically.

WARNING — If the DSFID is not encoded before MB 01 (containing the AFI and UII) then using MB 11 becomes extremely difficult because the UMI bit in the protocol control word will indicate that it contains no encoding.

Annex C(normative) -- Monomorphic-UII and URN Code 40 encoding

C.1 Monomorphic-UII

The Monomorphic-UII encoding scheme is designed to achieve a simple encoding of a Unique Item Identifier when encoded in a memory area dedicated to this function, and where no additional data is encoded in the same memory area. All of the features can be self-declaring through the registration of the particular AFI code values under the rules of ISO/IEC 15961-2. The following conditions apply:

— The Monomorphic-UII is only capable of being encoded in a tag memory architecture that supports the separate encoding of a UII. For this IPC standard, the tag is one that conforms to ISO/IEC 18000-63.

— An AFI assigned to a particular Monomorphic-UII shall not support the encoding of any additional item-related data in the same memory area. If item-related data is required for the application, then this shall be encoded in MB 11. For IPC standards, the AFI is $A0_{HEX}$.

— The AFI fully defines all the arcs of the Object-Identifier for communications in the commands of ISO/IEC 15961 and in the ISO/IEC 24791 standards. For IPC standards, the Object Identifier for the UII is 1.0.15961.14.

— Each AFI declares which of the encoding schemes, as defined in ISO/IEC 15962 is used. For this IPC standard, this is URN Code 40. As this encoding scheme is based on a sequence of 16-bit encoding units, there is no requirement to encode the length of the encoding. This is declared by the length bits in the protocol control word and 'calculated' by hardware rules for the ISO/IEC 18000-63 air interface protocol.

— The Monomorphic-UII does not require the encoding of a DSFID in MB 01. For this IPC standard, a DSFID is still required to be encoded in MB 11, when optional data elements are encoded.

C.2 URN Code 40 encoding

This particular encoding is designed to support the encoding of a hierarchical URN with the appropriate separators between the hierarchical components. This encoding scheme provides a method to encode data compliant with the **urn:oid namespace scheme** and extended to cover the Unique Item Identifier, as used in the ISO RFID data protocol. Therefore, when the URN is decoded from the RFID tag it is presented in a structure that is compatible with that is required for resolving on the Internet.

The UII shall be encoded using the URN Code 40 encoding defined in the following sub-clauses.

C.2.1 Basic Character Set

The encoding process takes a string of three data characters (from Table 10) and compacts these into two bytes. The PAD character is used in the final string where there are fewer than three data characters.

Three URN Code 40 values (the last column in the table) are encoded as a 16-bit value (msb first). Three URN Code 40 values (C_1, C_2, C_3) shall be encoded as:

$$(1600*C_1) + (40*C_2) + C_3 + 1$$

This process produces a value in the range 1 to 64000, which is converted to hexadecimal in the range 0001_{HEX} to $FA00_{HEX}$.

The procedure continues for each triplet of input characters.

Table 10 - URN Code 40 Character Set

Graphic Symbol	Name	HEX Code	8-bit code	URN Code 40 (decimal)
	PAD			0
A	Capital letter A	41	01000001	1
B	Capital letter B	42	01000010	2
C	Capital letter C	43	01000011	3
D	Capital letter D	44	01000100	4
E	Capital letter E	45	01000101	5
F	Capital letter F	46	01000110	6
G	Capital letter G	47	01000111	7
H	Capital letter H	48	01001000	8
I	Capital letter I	49	01001001	9
J	Capital letter J	4A	01001010	10
K	Capital letter K	4B	01001011	11
L	Capital letter L	4C	01001100	12
M	Capital letter M	4D	01001101	13
N	Capital letter N	4E	01001110	14
O	Capital letter O	4F	01001111	15
P	Capital letter P	50	01010000	16
Q	Capital letter Q	51	01010001	17
R	Capital letter R	52	01010010	18
S	Capital letter S	53	01010011	19
T	Capital letter T	54	01010100	20
U	Capital letter U	55	01010101	21
V	Capital letter V	56	01010110	22
W	Capital letter W	57	01010111	23
X	Capital letter X	58	01011000	24
Y	Capital letter Y	59	01011001	25
Z	Capital letter Z	5A	01011011	26
–	Hyphen-Minus	2D	00101101	27
.	Full stop	2E	00101110	28
:	Colon	3A	00101110	29
0	Digit zero	30	00110000	30
1	Digit one	31	00110001	31
2	Digit two	32	00110010	32
3	Digit three	33	00110011	33
4	Digit four	34	00110100	34
5	Digit five	35	00110101	35
6	Digit six	36	00110110	36
7	Digit seven	37	00110111	37
8	Digit eight	38	00111000	38
9	Digit nine	39	00111001	39

C.2.2 Encoding Example

Using the example in 8.4.3, the encoding can be shown in more detail:

Step 1 Input the character string **A.RY013000415CH**

Step 2a The encoding process begins by taking the first three characters, which are the Relative-OID for this S10 standard **A**, followed by the 'dot' separator, followed by the first letter of the S10 service indicator.

Example: **A.R**
These are converted to their URN Code 40 table values and then compacted as follows, where the letter **A** has the decimal value **1** from Table 10, the 'dot' has the decimal value **28** and the letter **R** has the decimal value **18.** Using the equation in C.2.1 the first 16-bit string can be calculated:

$1600*1 + 40*28 + 18 + 1 = 2739_{10} = 0000101010110011_2 = 0AB3_{HEX}$

Step 2b The encoding process continues by taking the next three characters, which are the second letter of the S10 service indicator followed by the first two digits of the serial number.

Example: **Y01**
From the table letter Y= 25, 0 = 30, 1 = 31. Using the equation, the second 16-bit string can be calculated:

$1600*25 + 40*30 + 31 + 1 = 41232_{10} = 1010000100010000_{2} = \text{A110}_{\text{HEX}}$

Step 2c The encoding process continues by taking the next three characters, which are the third, fourth and fifth digits of the serial number.

Example: **300**
From the table 3 = 33, 0 = 30, 0 = 30. Using the equation, the third 16-bit string can be calculated:

$1600*33 + 40*30 + 30 + 1 = 54031_{10} = 1101001100001111_{2} = \text{D30F}_{\text{HEX}}$

Step 2d The encoding process continues by taking the next three characters, which are the sixth, seventh and eighth digits of the serial number.

Example: **041**
From the table 0 = 30, 4 = 34, 1 = 31. Using the equation, the fourth 16-bit string can be calculated:

$1600*30 + 40*34 + 31 + 1 = 49392_{10} = 1100000011110000_{2} = \text{C0F0}_{\text{HEX}}$

Step 2e The encoding process continues by taking the final three characters, which are the check digit of the serial number followed by the 2-alpha country code.

Example: **5CH**
From the table 5 = 35, C = 3, H = 8. Using the equation, the fifth 16-bit string can be calculated:

$1600*35 + 40*3 + 8 + 1 = 56129_{10} = 1101101101000001_{2} = \text{DB41}_{\text{HEX}}$

Step3: The resultant byte string is: $\text{0AB3A110D30FC0F0DB41}_{\text{HEX}}$

This encoding example is shown graphically in Annex F.

Annex D(informative) MB 11 encoding examples

D.1 General considerations

This annex shows the encoding of a hypothetical set of data elements compliant with this IPC standard.

The processes are presented in a manner to help the reader understand how input data is converted to encoded bytes on the RFID tag. This is done progressively for each data element, but it should be borne in mind that software compliant with ISO/IEC 15962 might have different processes to achieve the same end result.

D.2 Input assumptions

D.2.1 The RFID tag

Sufficient memory capacity exists in MB 11 to support the example. In a real encoding process, the encoding software should report an error when there is insufficient memory. This IPC standard does not define the error message because different compliant ISO/IEC 15962 can have different processes.

The DSFID is required to be encoded as the first byte of MB 11. The next byte is part of the first data set.

D.2.2 The input data

The data elements to be encoded are described in Table 11.

Table 11 - Example data elements

Data Element	Sequence	Rel-OID	Format	Example data
International postal code	1st	10	Variable length alphanumeric	US63366-9700
Associated documents	2nd	11	Variable length alphanumeric based on up to three UPU Code List 184 items	U22750INV
Gross weight	3rd	16	Numeric format of weight in hectograms (i.e. kg to one place of decimal)	47 (for 4.7 kg)
Transport instructions	4th	17	Bitmap, if '1' invoke 1st bit = scan bar code 2nd bit = signature 3rd bit = postal locker 4th bit = deliver to pick up location 5th to 8th bit = reserved	10010000

D.3 Encoding the data elements

D.3.1 General

The Relative-OID values in the range 1 to 14 have their value directly encoded in the Precursor as defined in 8.7.6. The Relative-OID values in the range 15 to 127 have their value encoded in the byte after the Precursor as defined in 8.7.7.

The structure of the precursor is as defined in Table 6.

D.3.2 Encoding the international postal code

Using the example from Table 11, the encoding process is as follows:

1. Input the character string **U22750INV**

2. After parsing the input string, the ISO/IEC 15962 automatically chooses the 6-bit compaction scheme, code 100

3. Output the compacted bytes **553DB3CF6DADE77C30**

4. The encoder counts the number of bytes and adds this as the length **09553DB3CF6DADE77C30**

5. The encoder constructs the Precursor as in Table 6
   a. Position 7 = 0
   b. Positions 6 to 4 = 100 to indicate 6-bit compaction
   c. Position 3 to 0 = 1010 to indicate the Relative-OID 10

   This results in the bit string 01001010 = $4A_{HEX}$

6. Construct the complete data set **4A09553DB3CF6DADE77C30**

D.3.3 Encoding the codes for the associated documents

Using the example from Table 11, the encoding process is as follows:

1. Input the character string **US63366-9700**

2. After parsing the input string, the ISO/IEC 15962 automatically chooses the 6-bit compaction scheme, code 100

3. Output the compacted bytes **572CB7D7024E5A**

4. The encoder counts the number of bytes and adds this as the length **07572CB7D7024E5A**

5. The encoder constructs the Precursor as in Table 6
   a. Position 7 = 0
   b. Positions 6 to 4 = 100 to indicate 6-bit compaction
   c. Position 3 to 0 = 1011 to indicate the Relative-OID 11

   This results in the bit string 01001011 = $4B_{HEX}$

6. Construct the complete data set **4B07572CB7D7024E5A**

D.3.4 Encoding the gross weight

Using the example from Table 11, the encoding process is as follows:

1. Input the character string **47** to represent 4.7kg

2. After parsing the input string, the ISO/IEC 15962 automatically chooses the integer compaction scheme, code 001.

3. Output the compacted bytes **2F**

4. The encoder counts the number of bytes and adds this as the length **012F**

5. The encoder constructs the Precursor as in Table 6
   a. Position 7 = 0
   b. Positions 6 to 4 = 001 to indicate integer compaction
   c. Position 3 to 0 = 1111 to indicate the Relative-OID is greater than 15

This results in the bit string 00011111 = **1F**HEX

6. The Relative-OID 16 needs to be encoded immediately after the Precursor, as follows:
   16 -15 = 1 = **01**HEX

7. Construct the complete data set **1F01012F**

D.3.5 Encoding the transport instructions

Using the example from Table 11, the encoding process is as follows:

1. Input the bit string **10010000**

2. Because this is a bit map, the application needs to declare that the application-defined compaction scheme, code 000, shall be used.

3. The bit string is converted to hexadecimal resulting in the byte **90**

4. The encoder counts the number of bytes and adds this as the length **0190**

5. The encoder constructs the Precursor as in Table 6
   a. Position 7 = 0
   b. Positions 6 to 4 = 000 to indicate application-defined compaction
   c. Position 3 to 0 = 1111 to indicate the Relative-OID is greater than 15

   This results in the bit string 00001111 = **0F**HEX

6. The Relative-OID 17 needs to be encoded immediately after the Precursor, as follows:
   17 -15 = 2 = **02**HEX

7. Construct the complete data set **0F020190**

D.3.6 Assembling the data sets and adding the DSFID

The data sets are concatenated in the required sequence. For this example, this is as defined in 8.7.2.

4A09553DB3CF6DADE77C30 4B07572CB7D7024E5A 1F01012F 0F020190

NOTE The spaces are to clarify the boundaries between the data sets, but are not encoded.

The encoded data sets are preceded by the DSFID, which is **0E** (see 8.7.3), resulting in a string of bytes represented as 16-bit words:

0E4A 0955 3DB3 CF6D ADE7 7C30 4B07 572C B7D7 024E 5A1F 0101 2F0F 0201 90

In this example, the final word is padded with the byte 00HEX to align with a word boundary, resulting in an encoding of 15 words, requiring an RFID tag with at least 240 bits of memory in MB 11.

D.4 Locking MB 11

Once all the data sets have been encoded, all of MB 11 shall be locked. The timing and location of invoking locking MB 11 needs to take account of any shared responsibility between an e-seller and the first postal operation.

Annex E (informative) ISO/IEC 18000-63 *Select* command

Successive ***Select*** commands can be used prior to a ***Query*** command to widen tag population selection to include test tags, assets and other postal products. Only those tags that hear the ***Select*** commands and meet the criteria defined by the ***Select*** command(s) can be instructed to participate or not participate in an inventory round.

When a tag meets selection criteria its **Select flag** gets asserted or de-asserted depending on the **Action** variable state issued in the ***Select*** command. When a ***Query*** command is sent after the ***Select*** command(s) at the beginning of a tag inventory round, it can request only those tags with **Select** and specific **Inventory** flags set to participate. The ***Query*** command also sets a session number S0 to S3 to support tag communication with multiple readers.

The ***Select*** command (**Error! Reference source not found.**) includes the following parameters:

— **Target**: 3 bits: indicates which flags, SL (select flag) or inventoried and sessions flags are changed as a result of meeting ***Select*** command filter requirements.

— **Action**: 3 bits: This parameter is set to determine the tag response as shown in Table 13. Put more simply:

  a) if **Target** specifies SL flag is changed – whether the SL flag is set (asserts) or resets (de-asserts) if a tag meets selection criteria

  b) if **Target** specifies inventory flag change the options are change to A or to B.

— **MemBank:** Specifies whether the Mask applies to UII, MB 11 or TID memory. The ***Select*** command filter mask components can operate on the UII, MB11 or TID memory. Successive ***Select*** commands prior to a **Query** command can extend filtering for content in multiple memory banks.

— **Filter Parameters:** The filter parameters include **Pointer, Length** and **Mask. Pointer** and **Length** describe the memory range**.** The **Pointer** references a bit address using **EBV** formatting. **Length** defines the mask length in bits. **Mask** contains a bit string that a Tag compares against the memory location that begins at the pointer and ends **Length** bits later.

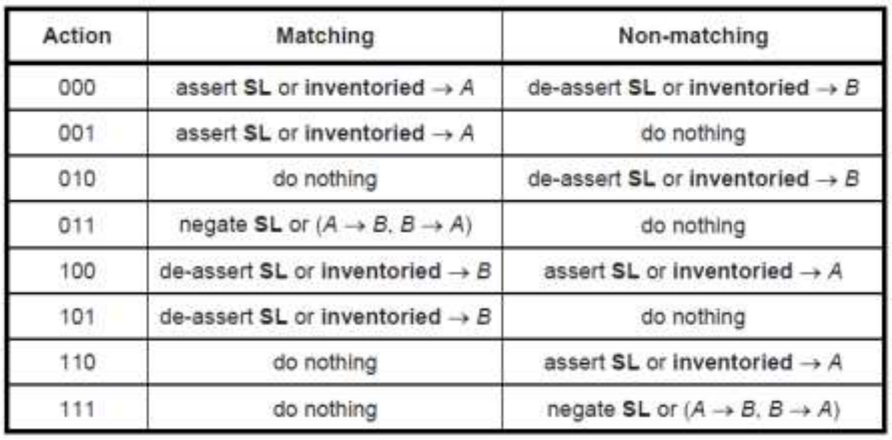
— **Truncate:** The **Truncate** function operator only works on UII backscatter response. If the ***Select*** Command has the **Truncate** parameter selected or asserted and if a subsequent **Query** command specifies **Sel** Paramaters (**Sel**=10 or **Sel**=11) then a matching tag will truncate its reply to an acknowledgement, **ACK**, to the portion of the UII immediately following the **Mask** followed by the **StoredCRC**. If a user is to apply or use successive ***Select*** commands and desires to truncate the response the user must assert or set **Truncate** in the last **Select** command prior to sending a **Query** command. The **Target** parameter must be set (100) such that tags will set **Sel** flag as a result of matching the **Mask** and the **Mask** ends with in the UII.

  *NOTE: Not all reader vendors support the **Truncate** function of the **Select** command. As this is not required for this IPC receptacle asset standard it should be set to **0** to disable the function.*

Table 12 - ISO/IEC 18000-63 Select command parameters

	Command	Target	Action	MemBank	Pointer	Length	Mask	Truncate	CRC-16
# of bits	4	3	3	2	EBV	8	Variable	1	16
description	1010	000: **Inventoried** (S0) 001: **Inventoried** (S1) 010: **Inventoried** (S2) 011: **Inventoried** (S3) 100: **SL** 101: RFU 110: RFU 111: RFU	See Table 13	00: Reserved 01: UII 10:TID 11:User	Starting Mask address	Mask length (bits)	Mask value	0: Disable truncation 1: Enable truncation	

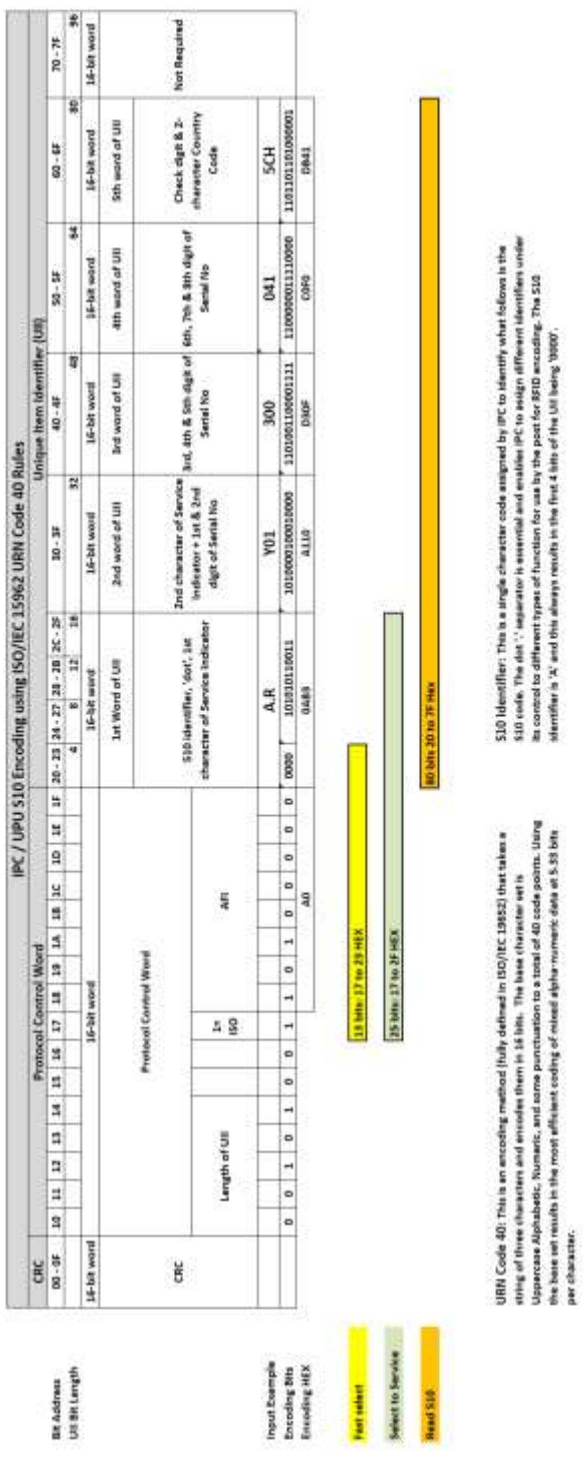
Table 13 - ISO/IEC 18000-63 tag response to Action parameter



Action	Matching	Non-matching
000	assert **SL** or **inventoried** → *A*	de-assert **SL** or **inventoried** → *B*
001	assert **SL** or **inventoried** → *A*	do nothing
010	do nothing	de-assert **SL** or **inventoried** → *B*
011	negate **SL** or (*A* → *B*, *B* → *A*)	do nothing
100	de-assert **SL** or **inventoried** → *B*	assert **SL** or **inventoried** → *A*
101	de-assert **SL** or **inventoried** → *B*	do nothing
110	do nothing	assert **SL** or **inventoried** → *A*
111	do nothing	negate **SL** or (*A* → *B*, *B* → *A*)

Annex F (informative) – Bit mapping of S10 UII encoding example

This Annex illustrates the bit and byte mapping described in 8.4 and shows the encoding of the AFI (see 8.4.2) and the worked example of the UII (see 8.4.3), together with the tag selection process defined in 10.



IPC / UPU S10 Encoding using ISO/IEC 15962 URN Code 40 Rules

	CRC	Protocol Control Word																Unique Item Identifier (UII)								
Bit Address	00 - 0F	10	11	12	13	14	15	16	17	18	19	1A	1B	1C	1D	1E	1F	20 - 23	24 - 27	28 - 2B	2C - 2F	30 - 3F	40 - 4F	50 - 5F	60 - 6F	70 - 7F
UII Bit Length																		4	8	12	16	32	48	64	80	96
	16-bit word	16-bit word																16-bit word				16-bit word	16-bit word	16-bit word	16-bit word	16-bit word
	CRC	Protocol Control Word																1st Word of UII				2nd word of UII	3rd word of UII	4th word of UII	5th word of UII	Not Required
		Length of UII							1= ISO	AFI								S10 identifier, 'dot', 1st character of Service Indicator				2nd character of Service Indicator + 1st & 2nd digit of Serial No	3rd, 4th & 5th digit of Serial No	6th, 7th & 8th digit of Serial No	Check digit & 2-character Country Code	
Input Example																		A.R				Y01	300	041	SCH	
Encoding Bits		0	0	1	0	1	0	0	1	1	0	1	0	0	0	0	0	0000	101010110011			1010000100010000	1101001100001111	1100000011110000	1101101101000001	
Encoding HEX										A0								0AB3				A110	D30F	C0F0	DB41	

Fast select	13 bits: 17 to 29 HEX
Select to Service	25 bits: 17 to 2F HEX
Read S10	80 bits 20 to 7F Hex

URN Code 40: This is an encoding method (fully defined in ISO/IEC 19852) that takes a string of three characters and encodes them in 16 bits. The base character set is Uppercase Alphabetic, Numeric, and some punctuation to a total of 40 code points. Using the base set results in the most efficient coding of mixed alpha-numeric data at 5.33 bits per character.

S10 Identifier: This is a single character code assigned by IPC to identify what follows is the S10 code. The dot '.' separator is essential and enables IPC to assign different identifiers under its control to different types of function for use by the post for RFID encoding. The S10 identifier is 'A' and this always results in the first 4 bits of the UII being '0000'.

Bibliography

IPC Interconnect Harmonised Labelling – Letter Package Label specifications

UPU Standards glossary (accessible at http://www.upu.int)

UPU Code List 184 Document type

ISO 3166-1, *Codes for the representation of names of countries and their subdivisions – Part 1. Country codes*

ISO/IEC 15961-1, *Information technology — Radio frequency identification (RFID) for item management: Data protocol — Part 1: Application interface*