International Post Corporation

IPC RFID STANDARD FOR: IDENTIFYING POSTAL RECEPTACLES BASED ON THE UPU S9 CODE, USING THE ISO/IEC 18000-63 PROTOCOL

Version 1.0

07 January 2019

IPC RFID standard for: Identifying postal receptacles based on the UPU S9 code, using the ISO/IEC 18000-63 protocol

Version 1.0

Contents

Introduction

The UPU S9 code has been used to identify the international movement of receptacles between International Mail Processing Centres (IMPC) for track and trace purposes of the consignments being transferred. Until now, the only means of data capture has been by representing the identifier in a bar code according to the UPU S47 standard and capturing this with a scanner.

RFID technology has advanced in performance and affordability, facilitating increased automation. Automated capture of data from multiple postal items simultaneously is possible by using RFID (radio frequency identification). An acceptable standardised approach is required for successful implementation. This includes implementation of a data construct plan that allows the leveraging of air interface protocols for tag population management. This IPC standard specifies an encoding structure for passive UHF RFID tags that accommodates the S9 data identifier requirements and effective management of tag populations. It is complementary to the other currently published data construct standards in use by the IPC. In particular, this will provide greater interoperability of RFID tags and infrastructure 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.

1 Scope

This IPC standard defines rules for encoding the UPU S9 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.

This first edition of this IPC standard does not specify any data to be encoded in MB 11. IPC reserves the right to specify such data in a future edition. Because the RFID tag is intended to be single use, any changes in future editions will be backward compatible.

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

The encoded data on the RFID tag is identical to the encoded data of the S9 in a linear bar code, resulting in the input and output formats being identical. This allows RFID data capture and bar code data capture to be interoperable and to work concurrently in the same system. The way that the data is encoded is dependent on the rules of the different data carriers: Code 128 symbology for the bar code, the rules of this standard for RFID. Both data carriers are supported by a printed human readable S9 code.

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 S9, *Postal receptacle identifier*

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 edition of this IPC standard, the term is only relevant to Memory Bank 11, for which no data elements have been specified

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 edition of this IPC standard, the term is only relevant to Memory Bank 11, for which no data elements have been specified

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
S9 identifier
identifier that identifies the consignment within the receptacle

3.11
S9 format
29-character string in the format of 15 alpha characters, followed by 14 numeric characters

NOTE The format and permitted code is fully defined in UPU S9, Postal receptacle 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
IMPC International Mail Processing Centre
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 and in some countries additionally in the range 915 MHz to 921 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 https://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 edition of this IPC standard **shall not** 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 160 bits to support this IPC standard.

— 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 is not required for this first edition of this IPC standard.

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 S9 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 used to lock or permalock the individual passwords, or the entire MB 01, or the entire MB 11. It is not required for tags to support this edition of this IPC standard. The command shall be supported by interrogators to enable interoperability with other IPC standards.
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 may be supported by the RFID tag.

Although ISO/IEC 18000-63:2013 and some earlier editions of the standard indicated that the *Kill* command can be used to re-commission a tag, this feature has been withdrawn from ISO/IEC 18000-63:2013.

NOTE No tag products are known to support this re-commissioning 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.

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 A.2).

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.

— MB 11 is intended for additional data. As no data has been specified for MB 11 in this first edition of this IPC standard, encoding in MB 11 shall not be implemented at present.

NOTE A postal operator that opts to have an RFID with MB 11 and encodes data in that tag risks losing the opportunity for future interoperability. Any data that has the potential to be encoded should be reported to IPC for further consideration.

— 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 user encoding is possible in MB 10, as this is reserved for the IC manufacturer.

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**$_{HEX}$ 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

As the DSFID is only relevant to encoding in MB 11, it is not defined in this first edition of this IPC standard. However it is relevant to other IPC standards, e.g. S10.

6.2.4 Data format

The data format is a component of the DSFID. As such it is not defined here. However it is relevant to other IPC standards, e.g. S10.

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 on the RFID tag.

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 **E**, is assigned by IPC to distinguish RFID tags that encode the S9 code from any other IPC RFID application standard. The arc, value **E**, is re-created by the RFID decoding process.

The specific URN structure for this IPC RFID S9 standard is:

urn:oid:1.0.15961.14.E.{S9Code}

A postal operation may retain the UII in this format including **1.0.15961.14.** as a prefix, or extract the S9 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 uses resolver systems and other URN based systems. Extracting the S9 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, should future editions of this IPC specify requirements for encoding in that memory bank.

The UII for this IPC standard shall comprise these components:

- the IPC ApplicationType for S9 codes: the letter **E**,
- a ‘dot’ separator (the ‘dot’ is also known as a ‘full stop’ or ‘period’, ISO/IEC 8859-1 code point $2E_{HEX}$),
- the S9 code.

NOTE 2 The ApplicationType is a mechanism that IPC can use to address other RFID applications and maintain full interoperability with this S9 standard.

Because the S9 code is applied by an IMPC in the country of origin and the consignment is transferred through known routes to the destination IMPC, the UII may be left unlocked.

7.2 Optional data elements encoded in MB 11

7.2.1 First edition of this IPC S9 standard

No data elements have been defined for this first edition of this IPC standard.

7.2.2 Data elements for encoding in MB 11 in future editions

There are potential data elements that may be specified for encoding in MB 11. In keeping with other IPC standards, particularly the S10 standard, similar rules will apply to enable full interoperability. These include:

- the assignment of Relative-OIDs that are different from other IPC standards to avoid any systems clash of data;

— the use of the same DSFID in MB 11 as with other IPC standards to preserve the integrity of IPC RFID standards;

— the use of the ISO/IEC 15962 No-Directory encoding and decoding rules to ensure that existing software investments are protected.

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’.

The following clauses identify the structure and rules as applicable for this IPC S9 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 the S9 code applies to a single transaction between the Origin IMPC and the Destination IMPC resulting in the data being transient until reaching the destination IMPC. Therefore selective locking of data elements using the *BlockPermalock* air interface command is not supported in this edition of this IPC standard. This also applies to future editions of this IPC standard that may define encoding rules for MB 11.

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 2, which shows the encoding for this IPC standard in the last row.

Table 2 - 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	1	0	1	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 2 represents the length of the encoding for the UII. In this case the bits **01010_2** indicate that there are ten 16-bit words starting in bit position 20_{HEX}. The value of the length field should be calculated automatically as defined in ISO/IEC 18000-63.

NOTE 1 A few tags at the beginning of the year will encode in nine 16-bit words and the UII length field will be calculated automatically by the interrogator.

— 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 2 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.

In the absence of any more specific procedures for a creating air interface commands, the 16-bit string defined in Table 2, shall be used to construct the encoding of MB 01 bit positions 10_{HEX} to $1F_{HEX}$. However as the length indicator is generated automatically, for some low value Despatch Numbers, this will differ from the bit values shown in Table 2.

8.4.3 Encoding the UII

The Monomorphic-UII shall be encoded using the URN Code 40 encoding rules as defined in Annex B. 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 S9 code, the length of input data will be the same for all tags.

These are the encoding steps:

1. Because a Monomorphic-UII is used, the encoder input is E.{S9Code}, noting that {S9Code} is replaced by the 29-character S9 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. There are different rules for long numeric strings that apply to this IPC standard. 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. Using the 29-character example S9 code from the UPU standard **DEFRAANLAMSAAUN40027002000258** the example below shows how the data is encoded. Because the S9 code is of a fixed structure of 15 alphabetic characters followed by 14 numeric characters, a compliant URN Code 40 encoder will automatically invoke numeric compaction for part of the numeric string.

4. As the S9 code is 29 characters long, and is preceded by the two characters '**E.**', the 31-character string is encoded in ten 16-bit words ending at bit location BF_{HEX}.

EXAMPLE	Basic structure:	E.{S9Code}	
	Step 1:	Add the **E.** Prefix: E.DEFRAANLAMSAAUN40027002000258	
	Step2	Encode the first 18 characters using the URN Code 40 Basic character set. As the digit **4** (for the last digit of the year) completes an input character triplet, it can be encoded using the Basic Character set encoding rules	
	Step2a:	E.D encodes as	$0010001110100101_2 = 23A5_{HEX}$
	Step2b:	EFR encodes as	$1010000100010000_2 = 2043_{HEX}$
	Step2c:	AAN encodes as	$0000011001110111_2 = 0677_{HEX}$
	Step2d:	LAM encodes as	$0100101100110110_2 = 4B36_{HEX}$
	Step2e:	SAA encodes as	$0111011011101010_2 = 76EA_{HEX}$
	Step2f:	UN4 encodes as	$1000010110010011_2 = 8593_{HEX}$

Step3: The remaining 13-digit string **0027002000258** is converted to hexadecimal by the URN Code 40 encoder, resulting in **064971D382**$_{HEX}$. Any leading zeros need to be retained in the input and output strings.

Step4: The fact that numeric encoding is used needs to be encoded as syntax

Step4a: The hexadecimal byte **FB** indicates that the subsequent encoding is a long string integer

Step4b: The next hexadecimal character indicates the length of the input numeric string from 9 to 24 digits. As the input string is 13 digits long, 13 minus 9 = **4**

Step4c: The final hexadecimal character indicates the length of the output hexadecimal string from 4 to 19 bytes. As the output string is 5 bytes long, 5 minus 4 = **1**

Step4d The resultant byte string for the numeric data is **FB41064971D382**$_{HEX}$

Step5: The resultant byte string for the entire S9, including the IPC S9 application indicator is: **23A5204306774B3676EA8593FB41064971D382**$_{HEX}$

NOTE 1 *Because of the constant structure of the trailing 13 digits of the S9 code, the seventh 16-bit word (steps 4a to 4c) is always* ***FB4n***$_{HEX}$*. Depending on the value of the Despatch Number this could be* ***FB41***$_{HEX}$ *(as in the example above) indicating that the trailing 13 digits is compacted into 5 bytes. The ISO/IEC 15962 encoding rules state that the hexadecimal bytes that in this case follow the 16-bit word* ***FB41***$_{HEX}$ *represent the integer value of the numeric string (msb first). This means that there is no requirement to add a pad byte* ***00***$_{HEX}$ *for the last word as part of the URN Code 40 procedure.*

NOTE 2 *For very low values of the S9 despatch number, the decimal to hexadecimal conversion will require 4 bytes. In the case the numeric indicator word is* ***FB40***$_{HEX}$*, which is then followed by 4 bytes that end on a 16-bit boundary.*

NOTE 3 *As the value of the S9 despatch number increases, the decimal to hexadecimal conversion will require 6 bytes. In the case the numeric indicator word is* ***FB42***$_{HEX}$*, which is then followed by 6 bytes that end on a 16-bit boundary. The threshold point between 5 or 6 bytes is close to the Despatch Number = 1099, but other subsequent component data elementss impact the value of the byte string.*

NOTE 4 *Even with all 13 S9 component values set to their highest values, the encoding will always be achieved in three words.*

An expanded version of a different example, showing the detailed calculations for each 16-bit string is given in Annex B.2.3.

8.4.4 Rules for writing and locking MB 01

MB 01 does not support any form of selective locking. Because the S9 RFID tag is relevant between the origin IMPC and the destination IMPC, there is no requirement to lock the UII. However, the entire memory bank may be locked to ensure 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).

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 and use of MB 11

This first edition of the IPC S9 standard does not specify any encoding in MB 11. Future editions may identify data elements that are either required or optional.

If encoding is specified for MB 11 in future editions, then such encoding shall be interoperable with existing IPC standards.

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 **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 only 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: **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 the 16-bit word string from the encoded Monomorphic-UII and converting it to characters. The UII for this IPC standard is can be 144 bits for a small number of instances at the beginning of the year, which means that nine 16-bit words will be returned. Thereafter it is always 160 bits long, which means that ten 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 S9 code conforming to this IPC standard.

- In hexadecimal, the first byte of the first word shall be 23$_{HEX}$.
  - If so, proceed with the decoding
  - If not, the encoding is wrong

- In binary, the first 8-bits of the first word shall be 00100011$_2$.
  - 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 first six words in succession, using the inverse of the URN Code 40 rules defined in Annex B.

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

- the first word is always: alpha, '.', alpha.
- the second, to fifth word is always alpha, alpha, alpha.
- the sixth word is alpha, alpha, numeric.

The seventh word indicates that the encoding has switched to numeric compaction for the remaining 13 digits of the S9 code.

- In hexadecimal, the first byte of the seventh word shall be **FB**$_{HEX}$

- The second byte declares the structure of the remainder of the S9 code and of the encoded bits:
  - The first hexadecimal character, **4**$_{HEX}$, declares the length of the last 13 digits of the S9 code
  - The second hexadecimal character can have one of three values:

   - **0**$_{HEX}$, declares that the compacted data is contained in four hexadecimal bytes.
   - **1**$_{HEX}$, declares that the compacted data is contained in five hexadecimal bytes. Any trailing **00**$_{HEX}$ byte shall be discarded before converting from hexadecimal to decimal.
   - **2**$_{HEX}$, declares that the compacted data is contained in six hexadecimal bytes.

- Any 16-bit words beyond bit location **BF**$_{HEX}$ are not read because the length of UII bits in the Protocol Word should result in additional words not being read by the interrogator.

9.1.3 S9 bar code interoperability

To maintain interoperability with S9 bar code and message structures, the process shall strip off the leading two characters '**E.**'

9.1.4 URN interoperability

To maintain interoperability with a URN structure, the 31-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.

10 Rapid reading of the UII

10.1 Overview

The following sub-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 S9 tags, S10 tags, 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 receive 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.

Selecting only on the AFI returns all IPC compliant tags. This can be used where there are known to be mixed environments of tags, e.g. to exclude GS1 product based tags, or to exclude IATA tags in an airport. Using the fast select procedure relevant to each particular IPC standard selects tags that are encoded to that standard and excludes other IPC tags.

The following sub-clauses define the ***Select*** command parameters for this IPC S9 standard. A technical explanation of this command is provided in Annex C.

10.3 Fast select

This method is used to select RFID tags with S9 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 **1101000000010**$_2$ as part of the command structure defined in Table 3. 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 S9 code.

Table 3 - *Select* command parameters for S9 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	1101000000010	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 using the parameters defined in Table 3 results in the selection of all tags that conform to this IPC S9 standard. With the *Select* set with these parameters, the complete UII is returned.

10.4 Select to an IMPC of Origin code

While it is possible to create a ***Select*** command to identify a particular IMPC of origin, there are some practical implications:

— Compare to the Fast Select, which only requiring 13 bits for the mask value, this would require a mask of 57 bits.

— With the total number of IMPC codes that are assigned at any point of time a look-up table would be required.

Both these constraints make such a selection process impractical.

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 the receptacle is certain to conform to this IPC standard, e.g. by invoking the methods in 10.3. These methods deliver the raw UII.

This may also be an option in a situation where the interrogator is unable to decode the URN Code 40 byte string to numeric value of the last 13 digits of the S9 code and where this needs to be processed higher up the application stack.

11 RFID label stock

The UPU S47-2 standard defines a specific area, known as Zone E, where the bar code is intended to be printed. To avoid distortion of the bar code by the underlying RFID tag and possible damage of the RFID chip during bar code scanning the RFID tag should not be placed behind Zone E. As Zone E will differ between the preferred landscape format label and the alternative portrait format the label stock needs to be specified by each postal operator. Figure 1 shows the location of Zone E.

←Zone E

Figure 1: S47 landscape label showing the position of Zone E

Annex A(informative) -- Information about tag compliance

A.1 Memory requirements

These are the requirement guidelines to fully support this IPC S9 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 S9 code. This requires 160 bits of memory, reducing the choice of tags that can be used. Tags with a larger UII memory may be used. 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 S9 standard, except to confirm compliance, and to possibly use the serial number for diagnostic purposes.

MB 11 Tags with this memory bank are not required for this first edition of this IPC standard, but may be required for future editions. As this first edition is based on the RFID tags being used only once, any future use of MB 11 can be phased in over a transition period.

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) -- Monomorphic-UII and URN Code 40 encoding

B.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 for encoding in MB 11 if this is specified for future editions of this IPC standard.

B.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 required for resolving on the Internet.

The UII shall be encoded using the URN Code 40 encoding defined in the following sub-clauses. For this IPC S9 standard, the first 14 characters (including the despatch year) are encoded using the Basic Character Set as defined in B.2.1. The remaining 13 decimal digits are converted to hexadecimal using the relevant ISO/IEC 15962 extended encoding rule for long integer strings. The rules relevant to this IPC S9 standard are defined in B.2.2.

NOTE The rules defined in B.2.2 apply because there are always 13 trailing digits in the S9 code, and as such are a specific implementation of the rules for encoding long integer strings. ISO/IEC 15962 defines comprehensive rules for compacting any integer string from 9 digits to 24 digits long. Such rules are only relevant if the intention is to support other non-IPC applications using URN Code 40.

B.2.1 Basic Character Set

The encoding process takes a string of three data characters (from Table 4) 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 4 - 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

B.2.2 Numeric compaction

Numeric compaction shall be invoked for the last 13 digits of the S9 code, starting with the despatch number.

NOTE The numeric digit for the despatch year is included as the third character of the final character triplet encoded using the Basic Character Set as defined in B.2.1.

Although the ISO/IEC 15962 rules for URN Code 40 numeric compaction can encode a numeric string of 9 to 24 digits, this IPC standard only addresses the requirements for a 13-digit string.

The table-driven encoding, as discussed in B.2.1, only uses the double byte values up to FA00$_{HEX}$. As the basic encoding for URN Code 40 is always of a pair of bytes, the lead byte can never have a value FB$_{HEX}$ to FF$_{1EX}$ in the table-driven solution. These byte values are used as syntax to declare different encoding rules. Only FB$_{HEX}$ is relevant for this IPC standard, and is used to signal that a long string integer is encoded. The syntax is shown in Table 5.

Table 5 - Syntax to declare long string integer encoding

	Lead Byte	3rd HEX Character	4th HEX Character
ISO/IEC 15962 Rule	Assigned to numeric compaction	Length of input digit string including leading decimal zeros, minus 9	Number of encoded bytes, minus 4
IPC S9 standard	**FB**$_{HEX}$	13 – 9 = **4**$_{HEX}$	4 – 4 = **0**$_{HEX}$ OR 5 – 4 = **1**$_{HEX}$ OR 6 – 4 = **2**$_{HEX}$

The fourth hexadecimal value is determined by the value of the despatch number and subsequent numeric values in the 13 digit string:

— At despatch number **0004**$_{DEC}$ the fourth character of the syntax can be either **0**$_{HEX}$ or **1**$_{HEX}$ depending on the value of the subsequent decimal digits. Lower despatch numbers will encode over four bytes and higher despatch numbers will encode over five or six bytes.

— At despatch number **1099**$_{DEC}$ the fourth character of the syntax can be either **1**$_{HEX}$ or **2**$_{HEX}$ depending on the value of the subsequent decimal digits. Lower despatch numbers will encode over five bytes and higher despatch numbers will encode over six bytes.

A fully compliant URN Code 40 encoder or decoder will process this automatically.

The 13-digit decimal string, including leading zeros, shall be converted to hexadecimal 8-bit bytes, including leading zero hexadecimal characters. Examples are shown in Table 6.

Table 6 - Examples of encoding

13-digit decimal input (spaces between data elements)	Hexadecimal encoding, including syntax (spaces between 16-bit words)
0002 002 1 0 0024	FB40 7755 9F38
0005 001 0 0 0024	FB41 012A 1534 58
0027 002 0 0 0258	FB41 0649 71D3 82
0488 037 0 0 0046	FB41 71A1 45A3 6E
1234 023 0 1 0175	FB42 011F 5180 CF7F
9999 002 0 0 0053	FB42 0918 12F6 5AB5

B.2.3 Encoding Example

Using a different example from that in 8.4.3, the encoding can be shown in more detail:

Step 1 — Input the character string **E.PTLISAGBLONBACN01234023010175**

Step 2a — The encoding process begins by taking the first three characters, which are the Relative-OID for this S9 standard **E**, followed by the ‘dot’ separator, followed by the first letter of the S9 IMPC of Origin code.

Example: **E.P**
These are converted to their URN Code 40 table values and then compacted as follows, where the letter **E** has the decimal value **5** from Table 4, the ‘dot’ has the

decimal value **28** and the letter **P** has the decimal value **16.** Using the equation in C.2.1 the first 16-bit string can be calculated:

$1600*5 + 40*28 + 16 + 1 = 9137_{10} = 0010001110110001_2 = 23B1_{HEX}$

Step 2b The encoding process continues by taking the next three characters of the IMPC of Origin.

Example: **TLI**
From the table, letter T= 20, L = 12, I = 9. Using the equation, the second 16-bit string can be calculated:

$1600*20 + 40*12 + 9 + 1 = 32490_{10} = 0111111011101010_2 = 7EEA_{HEX}$

Step 2c The encoding process continues by taking the next three characters, which are the last two characters of the IMPC of Origin and the first character of the IMPC of Destination.

Example: **SAG**
From the table S = 19, A = 1, G = 7. Using the equation, the third 16-bit string can be calculated:

$1600*19 + 40*1 + 7 + 1 = 30448_{10} = 0111011011110000_2 = 76F0_{HEX}$

Step 2d The encoding process continues by taking the next three characters, which are the second, third and fourth characters of the IMPC of destination.

Example: **BLO**
From the table B = 2, L = 12, O = 15. Using the equation, the fourth 16-bit string can be calculated:

$1600*2 + 40*12 + 15 + 1 = 3696_{10} = 0000111001110000_2 = 0E70_{HEX}$

Step 2e The encoding process continues by taking the final two characters of the IMPC of Destination, followed by the mail category code.

Example: **NBA**
From the table N = 14, B = 2, A = 1. Using the equation, the fifth 16-bit string can be calculated:

$1600*14 + 40*2 + 1 + 1 = 22482_{10} = 0101011111010010_2 = 57D2_{HEX}$

Step2f The encoding using the URN Code 40 Basic Character Set concludes by taking the two characters of the mail sub-class followed by the last digit of the calendar year.

Example: **CN0**
From the table C = 3, N = 14, 0 = 30. Using the equation, the sixth 16-bit string can be calculated:

$1600*3 + 40*14 + 30 + 1 = 5391 = 0001010100001111_2 = 150F_{HEX}$

Step3: The remaining 13-digit string **1234023010175** is converted to hexadecimal by the URN Code 40 encoder, resulting in **011F5180CF7F**$_{HEX}$. Any leading zeros shall be retained in the input and output strings.

Step4: The fact that numeric encoding is used needs to be encoded.

Step4a: The hexadecimal byte **FB** is part of the syntax that indicates that the subsequent encoding is a long string integer.

Step4b: The next hexadecimal character of the syntax word indicates the length of the input numeric string from 9 to 24 digits. As the input string is always 13 digits long for this IPC standard, 13 minus 9 = **4**

Step4c: The final hexadecimal character of the syntax word indicates the length of the output hexadecimal string from 4 to 19 bytes. For this IPC standard this may be 4, 5, or 6 depending on the value of the Despatch Number. For this particular example as the output string is 6 bytes long, 6 minus 4 = **2**

Step4d The resultant byte string for the numeric data is **FB42011F5180CF7F**$_{HEX}$

Step5: The resultant byte string for the entire S9, including the IPC S9 application indicator is:
23B17EEA76F00E7057D2150FFB42011F5180CF7F$_{HEX}$

This encoding example is shown graphically in Annex D.

Annex C(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 receive 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 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 8. 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 S9 standard it should be set to **0** to disable the function.*

Table 7 - 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 8	00: Reserved 01: UII 10:TID 11:User	Starting Mask address	Mask length (bits)	Mask value	0: Disable truncation 1: Enable truncation	

Table 8 - 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 D (informative) – Bit mapping of S9 UII encoding example

D.1 General

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

D.2 Selecting to the S9

It is possible to select all receptacles with an RFID tag encoding S9 data using a *Select* command with a mask of 13 bits, from bit position 17_{HEX} to 23_{HEX} inclusively as defined 10.3. This is illustrated in Figure 2.

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
																	4	8	12	16
16-bit word	16-bit word																16-bit word			
CRC	Protocol Control Word																1st Word of UII			
	Length of UII							1= ISO	AFI								S9 identifier, 'dot', 1st character of IMPC of Origin			
																	E.D			
	0	1	0	1	0	0	0	1	1	0	1	0	0	0	0	0	0010	001110100101		
									A0								23A5			
Fast Select								13 bits: 17 HEX to 23 HEX												

Figure 2: The mask for the *Select* command

NOTE 1: The length of the UII of 01010_2 in Figure 2 applies to most despatch numbers as shown in D.4 and D.5. However, the length of UII 01001_2 applies to a few despatch numbers as shown in D.3.

NOTE 2: The bit value 0110_2 in bit positions 20_{HEX} to 23_{HEX} is common to all S9 codes, thus enabling the selection of any S9 encoded RFID tag.

D.3 The S9 UII when URN Code 40 numeric compaction requires 4 bytes

An encoding example of an S9 code at the early part of the year is illustrated in Figure 3.

	IPC / UPU S9 Encoding using ISO/IEC 15962 URN Code 40 Rules									
	URN Code 40 Basic Rules						Syntax	URN Code 40 Numeric Compaction		
Bit Address	20 - 2F	30 - 3F	40 - 4F	50 - 5F	60 - 6F	70 - 7F	80 - 8F	90 - 9F	A0 - AF	B0 - BF
UII Bit Length	16	32	48	64	80	96	112	128	144	160
	1st Word of UII	2nd Word of UII	3rd Word of UII	4th Word of UII	5th Word of UII	6th Word of UII	7th Word of UII	8th Word of UII	9th Word of UII	10th Word of UII
	S9 Identifier, 'dot', 1st character of IMPC of Origin	2nd, 3rd, and 4th character of IMPC of Origin	5th and 6th character of IMPC of Origin and 1st character of IMPC of Destination	2nd, 3rd, and 4th character of IMPC of Destination	5th and 6th character of IMPC of Destination, then Mail Category Code	Two character Mail Sub-class, then last digit of calendar year	URN Code 40 Syntax to declare remaining 13 digits	13 digits comprising: 4-digit Despatch number, 3-digit Receptacle Serial Number, 1-digit for Highest numbered receptacle indicator, 1-digit Registered / Insured indicator, 4-digit Receptacle weight		
Input Example	E.D	EFR	AAG	BLA	LAD	UN7	Syntax	0001001100031		
Encoded HEX	23A5	2043	0670	0E62	4B2D	8596	FB40	3BAB	92FF	

Figure 3: The S9 UII example requiring four bytes of URN Code 40 Compaction

This UII structure, only requiring four bytes for numeric compaction, applies only to despatch numbers 0001 to 0003 and some values of 0004. As only four bytes are required for the numeric compaction, the UII is encoded over 144 bits and the length of the UII is automatically declared by interrogator processes in the Protocol Word as 01001_2.

D.4 The S9 UII when URN Code 40 numeric compaction requires 5 bytes

The encoding example in 8.4.3 is illustrated Figure 4.

	IPC / UPU S9 Encoding using ISO/IEC 15962 URN Code 40 Rules									
	URN Code 40 Basic Rules						Syntax	URN Code 40 Numeric Compaction		
Bit Address	20 - 2F	30 - 3F	40 - 4F	50 - 5F	60 - 6F	70 - 7F	80 - 8F	90 - 9F	A0 - AF	B0 - BF
UII Bit Length	16	32	48	64	80	96	112	128	144	160
	1st Word of UII	2nd Word of UII	3rd Word of UII	4th Word of UII	5th Word of UII	6th Word of UII	7th Word of UII	8th Word of UII	9th Word of UII	10th Word of UII
	S9 Identifier, 'dot', 1st character of IMPC of Origin	2nd, 3rd, and 4th character of IMPC of Origin	5th and 6th character of IMPC of Origin and 1st character of IMPC of Destination	2nd, 3rd, and 4th character of IMPC of Destination	5th and 6th character of IMPC of Destination, then Mail Category Code	Two character Mail Sub-class, then last digit of calendar year	URN Code 40 Syntax to declare remaining 13 digits	13 digits comprising: 4-digit Despatch number, 3-digit Receptacle Serial Number, 1-digit for Highest numbered receptacle indicator, 1-digit Registered / Insured indicator, 4-digit Receptacle weight		
Input Example	E.D	EFR	AAN	LAM	SAA	UN4	Syntax	0027002000258		
Encoded HEX	23A5	2043	0677	4B36	76EA	8593	FB41	0649	71D3	82

Figure 4: The S9 UII example requiring five bytes of URN Code 40 Compaction

This UII structure, only requiring five bytes for numeric compaction, applies to despatch numbers 0005 to 1098 and some values of 0004 and 1099. Although only five bytes are required for the numeric compaction, the UII is encoded over 160 bits and the length of the UII is automatically declared by interrogator processes in the Protocol Word as 01010_2. Bits in position $B8_{HEX}$ to BF_{HEX} are not calculated by the ISO/IEC 15962 URN Code 40 compliant software and are assumed to have a null value. Depending on how it has been designed, a compliant ISO/IEC 18000-63 interrogator may only encode up to bit position $B8_{HEX}$ or add the pad 0 bits to bit position BF_{HEX}. Whichever way the UII is encoded has no impact on the URN Code 40 decoder because the syntax word determines that only five bytes need to be processed.

D.5 The S9 UII when URN Code 40 numeric compaction requires 6 bytes

The encoding example in B.2.3 is illustrated Figure 5.

	IPC / UPU S9 Encoding using ISO/IEC 15962 URN Code 40 Rules									
	URN Code 40 Basic Rules						Syntax	URN Code 40 Numeric Compaction		
Bit Address	20 - 2F	30 - 3F	40 - 4F	50 - 5F	60 - 6F	70 - 7F	80 - 8F	90 - 9F	A0 - AF	B0 - BF
UII Bit Length	16	32	48	64	80	96	112	128	144	160
	1st Word of UII	2nd Word of UII	3rd Word of UII	4th Word of UII	5th Word of UII	6th Word of UII	7th Word of UII	8th Word of UII	9th Word of UII	10th Word of UII
	S9 Identifier, 'dot', 1st character of IMPC of Origin	2nd, 3rd, and 4th character of IMPC of Origin	5th and 6th character of IMPC of Origin and 1st character of IMPC of Destination	2nd, 3rd, and 4th character of IMPC of Destination	5th and 6th character of IMPC of Destination, then Mail Category Code	Two character Mail Sub-class, then last digit of calendar year	URN Code 40 Syntax to declare remaining 13 digits	13 digits comprising: 4-digit Despatch number, 3-digit Receptacle Serial Number, 1-digit for Highest numbered receptacle indicator, 1-digit Registered / Insured indicator, 4-digit Receptacle weight		
Input Example	E.P	TLI	SAG	BLO	NBA	CN0	Syntax	1234023010175		
Encoded HEX	23B1	7EEA	76F0	0E70	57D2	150F	FB42	011F	5180	CF7F

Figure 5: The S9 UII example requiring six bytes of URN Code 40 Compaction

This UII structure, requiring six bytes for numeric compaction, applies to all despatch numbers over 1100 and some values of 1099. The UII is encoded over 160 bits and the length of the UII is automatically declared by interrogator processes in the Protocol Word as 01010_2.

Bibliography

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

UPU Code List 108 International Mail Processing Centre (IMPC) codes - Open offices

UPU Code List 115 Mail category codes

UPU Code List 117 Mail sub-class codes

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