

Translated English of Chinese Standard: GB/T39852-2021

www.ChineseStandard.net → Buy True-PDF → Auto-delivery.

Sales@ChineseStandard.net

GB

NATIONAL STANDARD OF THE
PEOPLE'S REPUBLIC OF CHINA

ICS 35.040

A 24

GB/T 39852-2021

Electronic product code - Tag data translation

物品电子编码 标签数据转换

Issued on: March 09, 2021

Implemented on: October 01, 2021

Issued by: State Administration for Market Regulation;

Standardization Administration of the PRC.

Table of Contents

Foreword.....	3
1 Scope.....	4
2 Normative references.....	4
3 Terms and definitions	4
4 Abbreviations and symbols	4
5 EPC system and tag data	5
5.1 EPC system	5
5.2 Tag data	8
6 Tag data translation.....	13
6.1 Overview of tag data translation.....	13
6.2 Translation process.....	14
7 TDT tab files.....	18
7.1 Overview of TDT tab files.....	18
7.2 Definition method and additional requirements of TDT tab files.....	19
7.3 Elements and attributes of TDT tab files	37
8 Tag data translation algorithms	43
8.1 Overview	43
8.2 Function algorithm and application program interface (API)	43
Bibliography	52

Electronic product code - Tag data translation

1 Scope

This Standard specifies the methods and requirements for the electronic product code system and tag data, tag data translation, tag data translation tab files, and tag data translation algorithms.

This Standard applies to tag data translation and processing in the process of information exchange and transmission of electronic product code system.

2 Normative references

The following documents are indispensable for the application of this document. For the dated references, only the editions with the dates indicated are applicable to this document. For the undated references, the latest edition (including all the amendments) are applicable to this document.

GB/T 12905 Bar code terminology

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

ISO/IEC 19762 Information technology - Automatic identification and data capture (AIDC) techniques - Harmonized vocabulary

GS1 EPC Tag Data Standard

GS1 General specifications

3 Terms and definitions

The terms and definitions defined in GB/T 12905 and ISO/IEC 19762 apply to this document.

4 Abbreviations and symbols

The following abbreviations and symbols apply to this document.

ABNF: Augmented Backus-Naur Form

The identification and data capture layer realizes the automatic capture of product identification information, which is composed of identification and reader. The identification and data capture layer based on RFID technology is composed of RFID tag and reader. Other technologies are not specified in this Standard.

The RFID tag is an automatic identification carrier loaded with the electronic product code (EPC). It is usually attached to the identified product. It is the unique identification of the product's entire life cycle. It is composed of an antenna and a chip. The EPC is stored in binary form in the RFID tag.

RFID reader is a device that reads or reads-writes the information stored in RFID tag. The reader can be embedded with the LLRP, which can control the reader to read the original tag data, and exchange information with the information system through middleware.

5.1.3 EPC information system layer

The EPC information system layer provides data filtering and data sharing services for the application layer; including the application level event (ALE) interface that implements the filtering function; the data synchronization that implements the data sharing service; the GS1 EANCOM and GS1 eCOM XML standards; and the EPCIS standard that implements the product event information service, etc.

Note 1: GS1 EANCOM is an electronic data exchange specification developed by GS1 and applied to the field of commercial circulation.

Note 2: GS1 eCOM XML is an electronic data exchange specification based on XML language developed by GS1.

5.1.4 Application layer

The application layer refers to various applications, which are built by users for keywords using other forms of EPC or GS1 product code according to their needs.

5.1.5 EPC network service layer

The EPC network service layer provides network services for finding specific applications and specific objects on the Internet, including object naming service (ONS) and discovery service (Discovery), etc. Among them, object naming service (ONS) is a network service mode that identifies and discovers the entity object through a unique electronic product code.

5.2 Tag data

5.2.1 Tag data level

Tag data refers to data in various formats from RFID tag to object naming service (ONS) in the EPC system, including seven formats:

- Binary format (BINARY);
- Tag encoding URI format (TAG_ENCODING);
- Pure identity URI format (PURE_IDENTITY);
- GS1 text data field format (LEGACY);
- GS1 application identifier string format (LEGACY_AI);
- GS1 data transmission string format (ELEMENT_STRING);
- ONS domain name format (ONS_HOSTNAME).

Among them, binary format, tag encoding URI format and pure identity URI format are the three formats of RFID tag encoding. The GS1 text field, GS1 application identifier format and GS1 data transmission string format are the formats for storing and transmitting GS1 data in the EPC system and user application system. ONS domain name format is the format used for tag data to initiate query and retrieval in the object naming service (ONS).

Figure 2 shows the seven tag data levels in the EPC system, as well as the direction of the encoding process and the decoding process. The format level of tag data decreases from top to bottom. The binary memory format of RFID tags is the lowest level. The process of tag data translation from high-level format to low-level format is called the encoding process of tag data translation (the direction toward binary format). Conversely, the process of tag data translation from low-level format to high-level format is called the decoding process of tag data translation (the direction away from binary format).

Tag data translation respectively uses regular expression and augmented Backus-Naur form (ABNF) to define translation rules for different levels of EPC format. Regular expressions are mainly used to match input data and separate different data fields (combinations of bits, numbers, letters) from it. The augmented Backus-Naur form (ABNF) is mainly used to define the formatting rules of the output data; that is, how to combine the output result of the tag data translation process through the original data field, the derived data field, and the constant value.

The regular expression adopts the regular expression grammar that conforms to the Perl language; needs to support zero-length negative lookahead.

Note: The regular expressions in the TDT markup language are not just regular expressions that conform to the XSD specification, because they do not support the zero-length negative lookahead. The regular expression libraries of typical programming languages, such as Perl, Java, C#, and .Net, generally support the zero-length negative lookahead.

The grammar attribute defined in the ABNF format describes how to combine the various fields obtained in the translation process and the related fixed constant values to finally form the output data. In the grammar attribute, the fixed value is wrapped in single quotes. There is no name of the representative data field (original or derived) wrapped in single quotes. During the translation process, these data fields need to be replaced with actual values obtained. Each field (fixed value field and data field) of the grammar attribute is separated by a space. After the replacement and combination are completed, it needs to be deleted.

The sub-element field element <field> of the option element <option> also contains other related constraints and format translation rules, which are used to check or verify whether the EPC has errors.

The above definition mode is relatively complicated, but it increases future scalability. Especially for the URI format, each element is defined independently, which is convenient for check, and can be used to expand the rules when using longer-memory-digit tags in the future.

7.2.3 Judgment of input formats

The tag data translation process shall be able to, based on the input data (string), automatically determine its encoding structure and specific format. Other additional parameters, such as the tag length, can be obtained through the input data (such as input binary format or tag encoding URI format; the format itself has the tag length); or are inputted from outside and selected by the user. In addition, the translation between some formats (such as: between

the GS1 application identifier string format or the GS1 data transmission string format. In this process, some information (such as tag length, filter value, length of GS1 manufacturer identification code, etc.) may be lost. When additional parameters are needed in the decoding process, only 64-bit tags will need to use the translation table between the GS1 manufacturer identification code and the GS1 manufacturer identification code index as an additional parameter. Except for the above situation, the entire decoding process does not require the participation of additional parameters.

In some cases, the encoding process requires the following additional parameters to resolve the input string:

- The length of the GS1 manufacturer identification code;
- The actual tag length;
- Filter value (for example, indicating the packaging level: item level, box level or pallet level).

The tag data translation algorithm shall consider the input and acquisition methods of these additional parameters. Programming can adopt look-up tables or associative arrays (such as Dictionary, Hashtable) that introduce "key-value" pairs into the input parameters; or use appropriate regular expressions to extract directly from the input text string, etc.

In EPC application, GTIN and GLN types of GS1 identification need to provide a serial number as a supplement. In this case, the serial number shall not be transmitted to the tag data translation process by means of additional parameters. The serial number shall be used as a part of the input data. If the input is in GS1 text data field string format, add a ";serial=(serial number)" after GTIN or GLN. If the GS1 application identifier string or GS1 data transmission string is used, the application identifier 21 (serial number) is added for GTIN; the application identifier 254 (serial number) is added for GLN. Other encoding structures (such as SSCC, GRAI, GIAI, GDTI, GSRN) are serialized codes themselves; there is no need to add a serial number part. See Table 1 for examples. 8.2 specifies the application program interface of the tag data translation software, where input parameters only allow: input data, additional parameters, output formats. There is no need to consider individual values and intermediate variables.

numeric range, which a 34-bit binary number can represent, is 0-17,179,869,183, which is beyond the above range. Therefore, it is necessary to add a rule to the binary format, to check whether the serial number part of the SSCC is within a reasonable value range.

In order to avoid the above problems, in the tag data translation, for the field that represents a number or a combination of numbers, the decimalMinimum attribute is used to indicate the minimum possible value; the decimalMaximum attribute is used to indicate the maximum possible value. The field, which represents a combination of numbers and letters, does not require the above attributes. Once the decimalMinimum and decimalMaximum are specified, the tag data translation software needs to use a reasonable algorithm (binary to decimal, text to numeric value, etc.), to convert the field to a numeric value (decimal); and then perform the following check:

$$\text{decimalMinimum} \leq \text{field} \leq \text{decimalMaximum}$$

Once out of bounds, it is necessary to return an error or throw an exception.

7.2.8 Constraint and check requirements for text fields

For text fields, the characterSet attribute will be specified in the field tab <field>, to restrict the range of characters in the field. This attribute is defined by means of regular expressions. In regular expressions, inside the square brackets "[...]" are allowed characters; the asterisk "*" means that, the characters in the above-mentioned character range can appear zero or more times. When checking, it is recommended to add "^" to indicate the starting position of the matching field and "\$" to indicate the ending position of the matching field. For example, "[0-7]* \$" means that, the entire string is composed of a combination of numeric characters 0-7; no other characters are allowed.

Example:

[01]* - Only numeric characters 0 and 1 are allowed;

[0-7]* - Only numeric characters 0 to 7 are allowed;

[0-9]* - Only numeric characters 0 to 9 are allowed;

[0-9 A-Z\-*]* - Only numeric characters 0 to 9, space (ASCII code 32), uppercase letters A to Z, and hyphen "-" are allowed.

Note: There is a space between 9 and A here.

characterSet allows to check whether the characters of the field are within the allowed range during the tag data translation process, to ensure the correct transmission of EPC data. But there are special cases. For example: GRAI's

the length of the character set in the length attribute meets the requirement.

In order to avoid unnecessary duplication of definitions, the verification steps, resolution steps, rule execution steps, and result construction of the tag data translation process shall follow the following rules:

- a) If there is a field element <field> (original data field) in a different option element <option> in a format level <level> element, which contains the definition of padChar attribute, padDir attribute and length attribute; then the same field element <field> (original data field) between all different option elements <option> within the same format level <level> element shall be filled with the same padChar attribute, padDir attribute and length attribute. If there is a rule element <rule> (derived data field) in a format level <level> element, which contains the definition of padChar attribute, padDir attribute and length attribute; then all the same rule elements <rule> (derived data fields) within the same format level <level> element shall be filled with the same padChar attribute, padDir attribute, and length attribute.
- b) If a field element <field> or a rule element <rule> under the format level element <level> of a tag encoding URI format contains the length attribute, padDir attribute, and padChar attribute; then the field (original or derived) corresponding to the format level element <level> of all formats above the tag encoding URI format (such as pure identity URI format, GS1 data transmission string format, text field format, ONS domain name format, etc.) SHALL be filled with the same padChar attribute, padDir attribute and length attribute.
- c) If a field tab <field> or a rule tab <rule> under the format level element <level> of a binary format contains the length attribute, padDir attribute, and padChar attribute; and the padDir attribute and padChar attribute are not defined under the format level element <level> of the tag encoding URI format; this means that, when converting from the code of other formats to binary format, before the translation, the data in non-binary format needs to be filled according to the padding rules; and then the binary translation is completed. Conversely, when decoding from the binary format to other formats, according to the opposite way to this padding rule, the text result after the binary translation shall remove the added pad characters, to get the translation result (if the converted format has other padding rules, further padding needs to be done according to this rules). Here, for any EPC code structure, if the padChar attribute and padDir attribute provisions of the binary and non-binary formats are exactly the same, there is no need to specify the padChar attribute and padDir attribute in the binary format. If in the same TDT definition file, the same padChar attribute and padDir attribute are defined for both the

Step 1.3, According to the user's input or system parameters, set the desired output format.

Step 2, Judge the encoding structure and input format of the input data:

Step 2.1, Traverse all encoding structure elements <scheme> and their format level elements <level>; use the prefixMatch attribute¹⁾ of each format level element <level> to judge whether the input data meets its regulations, and judge:

- a) If the input data does start with the pattern specified by the prefixMatch attribute of the <level>: (letters followed by numbers, single brackets)
 - 1) If the upper encoding mode <scheme> of the format level element <level> is not set with the tagLength attribute, then the upper encoding mode <scheme> of the format level element <level> is recorded as a to-be-selected encoding mode. Record the format <level> as the to-be-selected input format, which is represented by "to-be-selected encoding mode+format level" (recorded as: to-be-selected input format);
 - 2) If the upper encoding mode <scheme> of the format level element <level> is set with the tagLength attribute, and the field data table also has the definition of tagLength, then: If the tagLength attribute is consistent with the value of tagLength in the field data table, record the encoding mode <scheme> and the format <level> as a to-be-selected input format. Otherwise, skip and proceed to the prefixMatch attribute judgment of the next format level element <level>.
- b) If the input data does not meet the prefixMatch attribute of this <level>, skip and continue to the prefixMatch attribute judgment of the next

1) Please note that, the prefixMatch attribute in the TDT tab file provides an optimized method to determine the input format, which is obviously more efficient than regular expressions. When designing tag data translation software, if a format contains both prefixMatch and regular expression: If the input data does not start with prefixMatch, it can be considered that the format does not match the input format, and continue to the next format to verify; if the input data does start with prefixMatch, then the regular expression must be verified to be able to determine whether it matches.

In addition, the prefixMatch attribute in the TDT tab file shall be an actual fixed string, not a regular expression. This is also because many programming languages have string start checking functions (such as startsWith), which are much more efficient than regular expressions. If software designers are willing to use regular expressions to implement the start check function, they can completely construct regular expressions on their own according to the value of prefixMatch. But please pay attention to the grammar (such as using '^' to indicate the beginning) and character escapes (such as using '\\.' to escape '\\.').

encoding URI format or a binary format, and there are multiple to-be-selected input formats, the encoding mode <scheme> is preferred to set the final input format and matching option that contains the tagLength attribute and is consistent with the tagLength of the input data. If the output format is the tag encoding URI format level or the binary format level, the additional parameter tagLength must be set.

Step 3.3, Record the value of the optionKey attribute of the final matching option element <option>, which will be used in step 6.

Step 4, According to the pattern of the matching option element <option> in the final input format, parse the input data and obtain the corresponding data fields:

Step 4.1, Verify whether the regular expression specified by the pattern attribute of the matching option element <option> exactly matches the input data (the previous and the verification can use the results of the previous check): If it matches exactly, start to consider each field element <field> defined under the option element <option>; otherwise, continue to find the matching option of the next to-be-selected input format.

Step 4.2, Use the regular expression specified by the pattern attribute in the option element <option> to match the input data. In the matching process, there will be a bunch of matching groups parsed using regular expression backreference syntax (backreference). The values of these matching groups will be assigned to each original data field one by one (defined by each field element <field> under the option element <option>):

a) Use the above to obtain each matching group. Each original data field is defined by a field element <field>. Each <field> element contains a seq attribute to indicate the matching group corresponding to the data field. Where seq="1" represents matching group 1; seq="2" represents matching group 2, and so on; to complete the assignment process.

b) When the above assignment is completed:

- 1) If the compaction attribute of the <field> element is not defined, or the defined value is "empty", the value of the field is treated as an integer (numeric field). Otherwise it is treated as a string (text field).
- 2) If the input format is binary format (the type attribute of the <level> element is BINARY), the 0-1 string obtained by matching the regular expression is treated as a binary string. If the field is a numeric field, convert the above binary string to an integer. If it is a text field, convert it to a string according to the compaction attribute.

Step 6.2, Under the <level> element, find the option element <option> with the same optionKey attribute value as the option determined in step 3, as the output option used in the output format.

Step 7, Execute various rules in the formatting phase (FORMAT), to obtain the value of the derived data field:

Step 7.1, According to the rules defined by each rule element <rule> in the formatting phase (FORMAT), according to the value of the seq attribute, perform various operations in turn, to generate derived data fields. Be prepared for formatting output results as follows.

Step 7.2, In the process of calculating the derived field, perform the necessary elimination (de-padding) and the necessary padding after translation. And check whether the result conforms to the defined characterSet attribute or decimalMinimum attribute, and decimalMaximum attribute.

Step 7.3, In the case of completely error-free, store the names and values of these derived data fields in the field data table.

Step 8, Use the field data table and the defined output format grammar to generate output results:

Step 8.1, According to the grammar specified by the grammar attribute in the output option <option> of the output format element <level>, generate an output result. This grammar conforms to the provisions of the augmented Backus-Naur form (ABNF) grammar, in which: single quotation marks contain constant values; no single quotation marks contain data field names (i.e. names or keys in the field data table).

Step 8.2, In the field data table, find the values corresponding to the names of all the above data fields. If the output format is a binary format, the above values shall be converted from a decimal integer or a string to a binary. And perform the necessary elimination (de-padding) and the necessary padding after the translation. The rules are shown in Figure 8.

Step 8.3, According to the grammar specified by the grammar attribute, combine the values found above (in accordance with the output format) and the constant values (with the single quotes removed) in the order specified by the grammar (during the combination, the space in the grammar attribute is ignored). Finally, the combined result is output as the output result and output data.

8.2.2.2 Translation interface

This is an excerpt of the PDF (Some pages are marked off intentionally)

Full-copy PDF can be purchased from 1 of 2 websites:

1. <https://www.ChineseStandard.us>

- SEARCH the standard ID, such as GB 4943.1-2022.
- Select your country (currency), for example: USA (USD); Germany (Euro).
- Full-copy of PDF (text-editable, true-PDF) can be downloaded in 9 seconds.
- Tax invoice can be downloaded in 9 seconds.
- Receiving emails in 9 seconds (with download links).

2. <https://www.ChineseStandard.net>

- SEARCH the standard ID, such as GB 4943.1-2022.
- Add to cart. Only accept USD (other currencies - <https://www.ChineseStandard.us>).
- Full-copy of PDF (text-editable, true-PDF) can be downloaded in 9 seconds.
- Receiving emails in 9 seconds (with PDFs attached, invoice and download links).

Translated by: Field Test Asia Pte. Ltd. (Incorporated & taxed in Singapore. Tax ID: 201302277C)

About Us (Goodwill, Policies, Fair Trading...): <https://www.chinesestandard.net/AboutUs.aspx>

Contact: Wayne Zheng, Sales@ChineseStandard.net

Linkin: <https://www.linkedin.com/in/waynezhengwenrui/>

----- The End -----