# ZAATS

## Zebra Advanced Asset Tracking System

# Tag Data and Numbering Guide

MN-003199-01

## Copyright



## Terms of Use

- Proprietary Statement

  This manual contains proprietary information of Zebra Technologies Corporation and its subsidiaries ("Zebra Technologies"). It is intended solely for the information and use of parties operating and maintaining the equipment described herein. Such proprietary information may not be used, reproduced, or disclosed to any other parties for any other purpose without the express, written permission of Zebra Technologies.
- Product Improvements

  Continuous improvement of products is a policy of Zebra Technologies. All specifications and designs are subject to change without notice.
- Liability Disclaimer

  Zebra Technologies takes steps to ensure that its published Engineering specifications and manuals are correct; however, errors do occur. Zebra Technologies reserves the right to correct any such errors and disclaims liability resulting therefrom.
- Limitation of Liability

  In no event shall Zebra Technologies or anyone else involved in the creation, production, or delivery of the accompanying product (including hardware and software) be liable for any damages whatsoever (including, without limitation, consequential damages including loss of business profits, business interruption, or loss of business information) arising out of the use of, the results of use of, or inability to use such product, even if Zebra Technologies has been advised of the possibility of such damages. Some jurisdictions do not allow the exclusion or limitation of incidental or consequential damages, so the above limitation or exclusion may not apply to you.

## Revision History

Changes to the original guide are listed below:

| Change | Date | Description |
|---|---|---|
| -01 Rev A | 5/2019 | Initial Release |

# Table of Contents

# About This Guide

## Introduction

This guide explains how the EPC memory bank in an RFID tag is used in a ZAATS system to identify an item that has an RFID tag attached to it and also provides guidelines on how to encode RFID tags used to identify items to achieve the highest levels of performance and interoperability.

## Scope

The general topic of tag data and numbering is a complex subject that impacts how the ZAATS RFID data is consumed and processed by the end user system. This guide describes both the general framework for how this data is structured consistent with standardized methods, and then prescribes in detail how RFID tag data should be structured to achieve the functionality and performance specific to ZAATS.

## Chapter Descriptions

Topics covered in this guide are as follows:

- Overview provides an introduction to tag numbering and describes the benefits of following a structured approach to encoding RFID tags in a ZAATS system. The most common international standards used to specify tag data structures are also listed.
- GS1 Encoding provides an overview of GS1 numbering methods, including an introduction to the GS1 Tag Data Standard (TDS).
- ISO Encoding provides an overview of ISO numbering methods.
- Encoding Guidelines for Single Tag Assets provides guidelines for end users to use either GS1 or ISO methods to develop internal numbering schemes for assets using a single RFID tag using standardized methods.
- Encoding Guidelines for Multi-Tag Assets provides guidelines for end users to use either GS1 or ISO methods to develop internal numbering schemes for fixed assets using multiple RFID tags using standardized methods.

## Notational Conventions

The following conventions are used in this document:

- “reader” refers to the Zebra ATR7000 RFID reader.
- **Bold** text is used to highlight the following:
  - Dialog box, window and screen names
  - Drop-down list and list box names
  - Check box and radio button names
  - Icons on a screen
  - Key names on a keypad
  - Button names on a screen.
- Bullets (•) indicate:
  - Action items
  - Lists of alternatives
  - Lists of required steps that are not necessarily sequential.
- Sequential lists (e.g., those that describe step-by-step procedures) appear as numbered lists.

## Related Documents and Software

The following documents provide background information related to tag data and numbering:

- ZAATS Deployment Guide, (p/n MN-003195-xx)
- CLAS Server and Software Installation Guide, (p/n MN-003197-xx)
- CLAS API Developer Guide, (p/n MN-003198-xx)

For the latest version of these guides and software, visit: [www.zebra.com/support](www.zebra.com/support)

## Service Information

If you have a problem with your equipment, contact Zebra Global Customer Support for your region. Contact information is available at: [zebra.com/support](zebra.com/support).

When contacting support, please have the following information available:

- Serial number of the unit
- Model number or product name
- Software type and version number.

Zebra responds to calls by email, telephone or fax within the time limits set forth in support agreements.

If your problem cannot be solved by Zebra Customer Support, you may need to return your equipment for servicing and will be given specific directions. Zebra is not responsible for any damages incurred during shipment if the approved shipping container is not used. Shipping the units improperly can possibly void the warranty.

If you purchased your Zebra business product from a Zebra business partner, contact that business partner for support.

## Provide Documentation Feedback

If you have comments, questions, or suggestions about this guide, send an email to EVM-Techdocs@zebra.com.

# Overview

## Introduction

The use of sound numbering practices is an essential element of a successful implementation of the ZAATS system. Though ZAATS can estimate the location of tags with any EPC ID, this guide strongly advises end-users to use internationally standardized numbering systems and to adopt the methods described in this guide. This will ensure that the ZAATS system performs as expected, especially in environments which may be exposed to assets containing tags that are not part of the end-user specific, closed-loop RTLS application.

Additionally, for items that have multiple tags (for example, forklifts or trolleys), ZAATS can combine the information from these multiple tags and produce a more accurate location of the item or indicate in which direction the item is pointing.

In summary, the benefits of following this guide include:

- Ensuring unique numbering of assets (tags) and avoid duplicate IDs which may corrupt system data
- Reducing the susceptibility to tagged assets outside of the end-user system degrading the real-time performance of ZAATS
- Allowing end users to more accurately track critical fixed assets, such as forklifts, trolleys, etc.

## Gen2 Background

The ATR7000 and ZAATS system is designed to work exclusively with RFID tags conforming to the GS1 EPC™ Gen2 air interface standard, more commonly referred to as the Gen2 protocol or Gen2 standard. In many applications, including ZAATS, the ISO/IEC 18000-63 air interface standard is functionally equivalent to the GS1 Gen2 standard. These standards have enabled a market for UHF RFID tags and readers by ensuring that readers and tags will be able to communicate and interoperate with each other regardless of the tag or reader manufacturer.

The Gen2 protocol relies on passive tags that are powered solely by the energy they absorb that is transmitted by the readers and identify themselves in an orderly manner. During this interrogation, the tags respond with all or part of their EPC Memory bank. A portion of the EPC memory bank is the Electronic Product Code (EPC) ID. In general, RFID tag memory is logically divided into four banks as follows:

**Bank 1 – Reserved Memory:** Contains Kill and Access passwords, if passwords are implemented.

**Bank 2 – EPC Memory:** Contains the EPC ID and other protocol control words defined in the Gen2 standard. The length of the EPC ID can range between 0 and 496 bits. The most common EPC length is 96 bits, and the Gen2 protocol has no restrictions on the contents of the EPC ID.

**NOTE:** In ISO applications, the EPC ID is commonly referred to as the UII.

**Bank 3 – TID Memory:** Contains the Tag ID; i.e. class identification and other tag or manufacturer-specific information. This memory is permalocked at the time of manufacture and cannot be modified.

**Bank 4 – User Memory:** Optional memory that allows end users to store asset specific information directly in the tag beyond the simple “license plate” of the EPC ID.

The structure of tag memory for Bank 2 (EPC Memory) is shown below in Table 1. It includes three distinct parts: the 16-bit CRC, the 16-bit Protocol Control (PC) bits, and the EPC itself (typically 96 bits for the most commonly available RFID tags).

**Table 1** EPC Tag Memory

| | | 00110 | 0 | 0 | 0/1 | 00000000 | |
|---|---|---|---|---|---|---|---|
| | ... | Length | UMI | XPC | Toggle | AFI | ... |
| | CRC (16 bits) | PC (16 Bits) | | | | | EPC |
| Memory Address | $00_h$ $0F_h$ | $10_h$ $4F_h$ | $15_h$ | $16_h$ | $17_h$ | $18_h$ $1F_h$ | $20_h$ $7F_h$ |

The “Toggle Bit” of the PC (Bit 0x17 of the EPC memory bank) indicates how to interpret the encoded data on the tag (either GS1 encoding or ISO encoding). If the Toggle Bit is ‘0’ this indicates that the tag’s EPC memory is encoded according to the GS1 EPC Tag Data Standard (TDS v1.11). If the Toggle Bit is ‘1’ then the EPC (UII) memory is according to ISO/IEC 15961 and ISO/IEC 15962.

# Numbering Systems

The two most prevalent set of standards that address the unique identification and serialization of RFID tags have been developed by GS1 and ISO. These two organizations have created separate, yet complementary schemes for tag data and numbering, either of which can achieve the objective of ensuring interoperability across the global supply chain and that a unique identity can be prescribed to each item (asset).

Note that in a strict sense it is not possible to guarantee absolute uniqueness of any item, and there is no guarantee that all users of RFID tags follow responsible guidelines. The probability of uniqueness, however, and achieving expected levels of system performance are very high when following the methods described in this guide.

The following chapter, GS1 Encoding, describes the framework of tag data and numbering using GS1 standardized methods. The chapter ISO Encoding describes the framework of tag data and numbering using ISO standardized methods.

## Common Standards Used to Define Tag Data and Numbering

End users who are involved in the definition of numbering of RFID tags for their organizations should consult the many published standards that exist. The following list is a summary of the standards most often used to help define internal data standards and numbering:

- GS1 System Architecture Document, Release 8.0, February 2019
- GS1 EPC$^{TM}$ Radio-Frequency Identity Protocols Generation-2 UHF RFID Standard; Specification for RFID Air Interface Protocol for Communications at 860 MHz - 960 MHz, Release 2.1, July 2018
- ISO/IEC 18000-63:2013; Information technology — Radio frequency identification (RFID) for item management - Part 63: Parameters for air interface communications at 860 MHz to 960 MHz Type C.
- GS1 EPC Tag Data Standard; defines the Electronic Product Code$^{TM}$ and specifies the memory contents of Gen 2 RFID Tags v1.11, September 2017
- ISO/IEC 15961-2; Information technology — Radio frequency identification (RFID) for item management: Data protocol — Part 2: Registration of RFID data constructs
- ISO/IEC 15961-3; Information technology — Radio frequency identification (RFID) for item management: Data protocol — Part 3: RFID data constructs
- ISO/IEC 15962:2013; RFID for item management: Data protocol — Data encoding rules & logical memory functions

# GS1 Encoding

## Introduction

The GS1 methodology is the most common form of identification used in both barcode and RFID applications. As described in the previous chapter, the Electronic Product Code (EPC) is a universal identifier for any physical object. This chapter provides an overview of the GS1 Tag Data Standard (TDS) and the guidelines for encoding of data on RFID tags that helps ensure uniqueness in item identification in a ZAATS system.

The GS1 EPC Tag Data Standard (TDS v1.11) defines the Electronic Product Code (EPC), including its correspondence to GS1 keys (described in the next section) and other existing codes. TDS also specifies data that is carried on Gen 2 RFID tags, including the EPC, User Memory data, control information, and tag manufacture information. Due to the large amount of information that can be stored in an EPC ID, the ability to identify unique instances of each item in a structured way forms the foundation for managing asset data and ensuring unambiguous identification of assets tracked by ZAATS.

## GS1 ID Keys

To give companies efficient ways to access information about items in their supply chains, and to share this information with other companies, GS1 defined a series of identification (ID) keys. GS1 identification keys are the foundation of the GS1 system and can be encoded on barcodes or RFID tags to enable companies to assign standard identifiers to products, documents, physical locations, and more. GS1 ID keys are globally unique, therefore, they can be shared between companies, increasing supply chain visibility for trading partners.

There are four types of keys defined by GS1 and they are summarized below:

- **Class 1:** Keys administered by GS1 and fully under its control.
- **Class 2:** Keys whose framework is controlled by GS1 and for which a portion of the identification capacity is allocated to an identification scheme administered by an external agency.
- **Class 3:** Keys fully administered and controlled outside of GS1, although, are supported in some part or parts of the GS1 system. A Class 3 key allows end users to take advantage of GS1 technology, although, without the expectation of the full global interoperability of Class 1 or Class 2 keys.
- **Class 4:** Keys that are entirely outside the GS1 system.

The data encoded in a common UPC barcode is an example of a data encoded using a GS1 Class 1 ID key. Specifically, the UPC-A barcode contains data encoded as a Global Trade Item Number (GTIN) key. The GTIN helps identify consumer goods, pharmaceuticals, medical devices, and other items at a unit level. For example, the GTIN can be used to identify an 8 ounce can of Classic Coke. GTIN also incorporates the ISBN number that uniquely identifies different books. For example, the first edition of Harry Potter and the Sorcerer’s Stone by JK Rowling published by Scholastic in the United States has a unique GTIN.

In addition to the GTIN, GS1 ID keys cover all sorts of other information that could be encoded. For example, physical locations as GLN (Global Location Number); pallets of items as SSCC (Serial Shipping Container Code); or office equipment, transport equipment, IT equipment or vehicles as GIAI (Global Individual Asset Identifier). With these globally unique Class 1 ID keys, anyone that scans the ID will be able to identify the object being scanned. There is no ambiguity if the GS1 ID keys are honored. This allows companies to use the ID as a key into a database and access information about the item. This key would work across different databases created by different companies.

## GS1 Tag Data Standard Specifics

GS1 Tag Data Standard (TDS v1.11) defines specifications for the EPC and its correspondence to GS1 Keys. Due to the large amount of information that can be stored in an EPC ID, the ability to identify unique instances of each item is now available. For example, the Serialized GTIN (SGTIN) is used to assign a unique identity to an instance of an item. The GTIN doesn't distinguish between the millions of the identical Coke cans that are produced each year or the millions of copies of that Harry Potter edition that were published. Each can or book gets the same GTIN. However, by using RFID and the SGTIN, it is possible to assign each can of Coke or each published copy of Harry Potter its own unique serialized identifier.

In addition to the SGTIN, the TDS describes how to encode the other GS1 ID keys into the EPC ID of a Gen2 compliant tag as well as other encoding schemes. Three of the schemes described in TDS that are relevant for ZAATS are the Serial Shipping Container Code (SSCC), the Global Individual Asset Identifier (GIAI), and the General Identifier (GID). The SSCC is an encoding scheme for logistics handling units, e.g. pallet loads or the aggregate contents of shipping containers. The GIAI is an encoding scheme used for assigning a unique identifier to a specific fixed asset, e.g. a forklift. The GID is a catch-all for encoding schemes that are independent of the TDS and provides end-users with a method to define their own numbering structures, while still complying with the overall objectives of the GS1 TDS that ensures all end users an orderly use of RFID.

Before deploying any ZAATS system, end-users should obtain their GS1 company prefix or contact GS1 to obtain a new prefix. The GS1 Company Prefix (GCP) is a unique string of four to twelve digits used to issue GS1 identification keys. The GCP is the first stage in creating a GS1 number scheme and can be obtained from your local GS1 Member Organization (MO), based on your company's numbering requirements.

## SSCC-96

SSCC is a GS1 Class 1 key. Table 2 illustrates the encoding of tags that adhere to the SSCC-96 encoding scheme. Note, the 0x31 in the EPC header field is what indicates the tag is programmed using this format. Table 3 lists the partition values (P) that are used with SSCC-96, the value which depends on the GS1 Company Prefix of the end-user. The note that the largest end users with 20-bit prefixes can use up to 38 bits of serialization (over 274 billion unique EPCs available).

**Table 2** SSCC-96 Encoding Table

| Scheme | SSCC-96 | | | | | |
|---|---|---|---|---|---|---|
| | | | SSCC | | | (Reserved) |
| Field Description | EPC Header | Filter | Partition | Company Prefix | Asset Reference | |
| # of Bits | 8 | 3 | 3 | 20-40 | 38-18 | 24 |
| | 0011 0001 | See TDS | See Table 3 | | | 0000...0000 |

**Table 3** SSCC-96 Partition Table

| | GS1 Company Prefix | | Individual Asset Reference | |
|---|---|---|---|---|
| **Partition Value (P)** | **Bits (M)** | **Digits (L)** | **Bits (N)** | **Digits** |
| 0 | 40 | 12 | 18 | 5 |
| 1 | 37 | 11 | 21 | 6 |
| 2 | 34 | 10 | 24 | 7 |
| 3 | 30 | 9 | 28 | 8 |
| 4 | 27 | 8 | 31 | 9 |
| 5 | 24 | 7 | 34 | 10 |
| 6 | 20 | 6 | 38 | 11 |

Note that the filter value field in the EPC is additional control information that may be included in the tag's EPC memory. The intended use of the filter value is to allow an RFID reader to select or to de-select RFID tags corresponding to certain physical objects or to make it easier to read the desired tags in an environment where there may be other tags present. For example, if the goal is to read the single tag on a pallet, and it is expected that there may be hundreds or thousands of item-level tags present, the performance of the ZAATS system may be improved by allowing the ATR7000 readers to select only pallet tags and de-select item-level tags.

# GIAI-96

GIAI is also a GS1 Class 1 key. Table 4 illustrates the encoding of tags that adhere to the GIAI-96 encoding scheme. Note, the 0x34 in the EPC header field is what indicates the tag is programmed using this format. Table 5 lists the partition values (P) that are used with GIAI-96. Note that with the GIAI-96 encoding scheme, even the smallest end users with 40-bit prefixes can use up to 42 bits of serialization (over 4 trillion unique EPCs available). Therefore, if using GS1 numbering methodology, the recommended encoding scheme for identifying fixed assets with the ZAATS system is GIAI-96. Refer to section 14.5.5 of the TDS for more information on encoding GIAI-96.

**Table 4** GIAI-96 Encoding Table

| Scheme | GIAI-96 | | | | |
|---|---|---|---|---|---|
| **Field Description** | **EPC Header** | **Filter** | **GIAI** | | |
| | | | **Partition** | **Company Prefix** | **Individual Asset Reference** |
| **# of Bits** | **8** | **3** | **3** | **20-40** | **62-42** |
| | 0011 0100 | 000 | See Table 3 | | |

**Table 5** GIAI-96 Partition Table

| Partition Value (P) | GS1 Company Prefix | | Individual Asset Reference | |
|---|---|---|---|---|
| | Bits (M) | Digits (L) | Bits (N) | Digits |
| 0 | 40 | 12 | 42 | 13 |
| 1 | 37 | 11 | 45 | 14 |
| 2 | 34 | 10 | 48 | 15 |
| 3 | 30 | 9 | 52 | 16 |
| 4 | 27 | 8 | 55 | 17 |
| 5 | 24 | 7 | 58 | 18 |
| 6 | 20 | 6 | 62 | 19 |

# GID-96

GID is a GS1 Class 3 key. There is only one encoding scheme for GID, illustrated in Table 6. Note, the 0x35 in the EPC header field is what indicates the tag is programmed using this format. There is no partition table when using GID-96. Refer to section 14.5.11 of the TDS for more information on encoding GID-96.

The Object Class is used by the ZAATS system and will be described in later detail in the subsequent chapters of this guide.

**Table 6** GID-96 Encoding Table

| Scheme | GID-96 | | | |
|---|---|---|---|---|
| Field Description | EPC Header | General Manager Number | Object Class | Serial Number |
| # of Bits | 8 | 28 | 24 | 36 |
| | 0011 0101 | See below | See below | See below |

The General Manager Number identifies an organizational entity (essentially a company, manager or other organization) that is responsible for maintaining the numbers in subsequent fields – Object Class and Serial Number. GS1 assigns the General Manager Number to an organization and ensures that each General Manager Number is unique. Note that a General Manager Number is not a GS1 Company Prefix. A General Manager Number may only be used in GID EPCs. To obtain a General Manager Number, contact GS1.

The Object Class and Serial Number are both used by the ZAATS system and will be described in greater detail in subsequent chapters of the guide. The end user has full control over assignment of the Serial Number.

# ISO Encoding

## Introduction

Another common standardization method used for RFID is based on ISO standards ISO/IEC 15961, ISO/IEC 15962, and ISO/IEC 15963 and related standards to facilitate global exchange of goods among trading partners. While such methods are perfectly suitable for tag data and numbering methods, they are generally beyond the scope of this document. This chapter instead describes a simplified, closed-system methodology for tag numbering that is based on ISO methods, and is compatible with both GS1 and open-loop ISO methodologies.

## Simplified ISO Encoding Method

The encoding methods described below are similar to the methods described in the chapter on GS1 encoding. One important difference between GS1 and ISO is related to the terminology used to describe the unique identifier. In GS1 the unique identifier is referred to as an Electronic Product Code, or EPC. In ISO terminology, the same unique identifier is referred to as a Unique Item Identifier, or UII.

As mentioned above, it is possible to indicate the use of proprietary (closed-system) encoding methods without violating open-system standards when using an ISO/IEC 18000-63 (i.e. Gen 2) tag. When correctly used, the information at the start of the EPC memory bank (in the PC bits) can indicate, in an ISO-compliant manner, that the EPC memory bank is encoded according to “private” rules and cannot be interpreted by systems outside of the end user’s domain.

To use this method:

1. The tag’s Toggle Bit in the EPC memory bank is set to ‘1’ indicating the use of ISO encoding, and specifically indicating that the next eight bits on the tag constitute an ISO-compliant Application Family Indicator (AFI).
2. The tag’s eight AFI bits are then set to all zeros indicating that the AFI is “not configured” and should not be interpreted in an open-system manner. Note that other eight-bit AFI values 0x01 to 0x0F are defined as “assigned to closed system environments” in ISO/IEC 15961-3 and may also be used for this purpose. The use of AFIs from 0x01 to 0x0F follow the specific rules defined in ISO/IEC 15961-3 and ISO/IEC 15693-3.
3. The remainder of the EPC memory bank is encoded according to either user-defined rules; i.e. using the guidelines described in this document; or using any published data standard.

It is important to understand that responsible closed-system encoding can only be accomplished using the above methods. If the tag’s Toggle Bit is set to ‘0’ instead of ‘1’, then the tag will appear to outside systems as intended for GS1 applications and will appear as a corrupt or incorrectly encoded tag. This practice could disrupt the data systems of your trading partners or any RFID system that comes into contact with your RFID tagged items and assets. By following these guidelines, you will demonstrate your commitment to best practices and the responsible use of RFID.

## ISO Encoding Data Structure Used in ZAATS

Table 7 illustrates the generic data structure format used for encoding tags using ISO methods in ZAATS. This data structure has the advantage of being very flexible, offers over 64k ZAATS specific IDs (to be used to uniquely identify the end user system) and over $2^{64}$ UII codes. The use of the ZAATS ID reduces the likelihood of “pollution” with other closed-systems employing similar methods.

**Table 7** ISO Closed-System Encoding Table Used in ZAATS

| Scheme | ISO Closed-System (96 bit) | | |
|---|---|---|---|
| Field Description | ZAATS ID | Object Class | Serial Number |
| # of Bits | 16 | 16 | 64 |
| | | | |

Details on how the ZAATS ID, Object Class, and Serial Number are assigned and used by ZAATS are described in the chapters that follow.

# Encoding Guidelines for Single Tag Assets

## Introduction

This chapter describes the recommend tag and numbering methods to be used with single-tagged assets.

## Single Tag Encoding

Though ZAATS can estimate the location of tags with any EPC ID, following either the GS1 or ISO data encoding, as described in this section, will ensure the best performance of ZAATS. This will ensure that the ZAATS system performs as expected, especially in environments which may be exposed to assets containing tags that are not part of the end-user specific, closed-loop RTLS application. When the encoding schemes described below are adhered to, only the appropriate tags are read. In this way, extraneous reads are avoided and the end-user tags that are meant to be identified and located are read more frequently, thereby improving overall system performance.

ZAATS is able to distinguish between the appropriate and extraneous tags by utilizing the structures described in the previous chapters. For GS1 applications, ZAATS can be configured to only process tags with a specific GS1 Company Prefix or General Manager Number. Similarly, when using ISO encoding, ZAATS can be configured to only process tags with their Toggle bit set to ‘1’ and that use a specific ZAATS ID data structure. Details of how ZAATS is configured with the specific information it needs, GS1 or ISO, can be found in the CLAS API Developer Guide (p/n MN-003198-xx).

## Encoding with GS1

When using the GS1 Tag Data Standard, each tag must be encoded as per the standard. As described in the Gen2 Background section of the Overview chapter, the Toggle bit in the PC section of the EPC must be set to ‘0’. If GID-96 tags are used, all tags to be processed in the system should be encoded with GID-96 and share the same General Manager Number. Tags encoded with all other coding schemes in the TDS (e.g. GIAI-96 and SSCC) to be processed within ZAATS must share the same GS1 Company Prefix. This allows ZAATS to differentiate between ZAATS related tags and extraneous tags.

With the exception of the GID-96, all of the coding schemes in the TDS (including SSCC and GIAI-96) place the GS1 Company prefix in the same location. It is, therefore, preferred to use one (or more) of these coding schemes (as opposed to the GID-96 coding scheme).

## Encoding with ISO

As described in the Gen2 Background of the Overview chapter, when using ISO encoding, the Toggle bit in the PC section of the EPC must be set to '1' and the remaining 8 bits of the PC (the AFI) must be set appropriately as described above in the ISO Encoding chapter. All tags to be processed within ZAATS shall share the same ZAATS ID. This allows ZAATS to differentiate between ZAATS related tags and extraneous tags.

## Configure CLAS with the Encoding Type and Company-Specific Identifiers

By default, the CLAS software does not take into consideration encoding methods or company-specific identifiers. This enables out-of-box functionality. However, to ensure best practices, it is strongly recommended that the methodologies outlined above are followed.

After selecting the encoding type, the following parameters should be configured into the CLAS software:

- Encoding Type: GS1 SSCC-96, GS1 GID-96, or ISO-96
- GS1 Partition and Company Prefix or General Manager Number: Only used for GS1 encoding
- ZAATS ID: Only used for ISO-96

Please refer to the CLAS API Developer Guide, (p/n MN-003198-xx) for information on how the specific parameters are configured into the CLAS software.

# Encoding Guidelines for Multi-Tag Assets

## Introduction

This chapter describes the process of placing multiple RFID tags on a single object, typically a fixed asset such as a forklift or trolley.

By default, the ZAATS system will try and locate every tag in the field of view of the ATR7000 in the system and report the Tag ID (or EPC) and x-y-z location. However, it may be advantageous to tag certain objects with multiple tags on different parts of the object. Tagging an asset with more than one tag potentially has multiple benefits:

- It increases the likelihood of reading a tag on the object.
- It allows for combining the location estimates from each of the tags to provide a location estimate for the object with higher confidence compared to the confidence of each individual tag location estimate.
- It gives the system an ability to determine the direction of the object by performing a rigid body transformation.

One drawback of tagging an asset with multiple tags is obvious, the cost is higher. However, for fixed assets that are permanently affixed with tags, this cost is negligible. The second less obvious drawback is that multiple tags take longer to read. For example, if every asset in an end-user system had two RFID tags applied to it, the location accuracy would indeed increase, however, the update rate would decrease by 50%. Therefore, choosing to tag items with multiple tags should be done judiciously and only when the benefits outweigh the drawbacks.

To take advantage of the Multi-Tag combining provided by ZAATS, the following steps are required:

1. Define and record the relative locations of each tag for each fixed asset type.
2. Decide on an encoding type – GIAI-96 or GID-96 if GS1 encoding methods are used, or ISO.
3. Encode tags and place on the assets to be tracked.
4. Configure ZAATS with the encoding type and tag locations for each asset type/class.

These steps are described in the sections that follow.

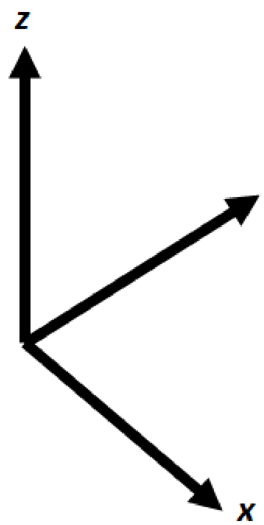
## Define and Record the Relative Locations of Multi-Tag Assets

### Geometry

To enable the benefits outlined above, ZAATS needs to know the relative locations on the object and the tag IDs at each of these locations. The tags may be placed anywhere on the multi-tagged object, however, in order for the ZAATS system to combine information from individual tags to produce a composite location, the relative positions of the tags on the object must be consistent, known, and configured into the ZAATS system.

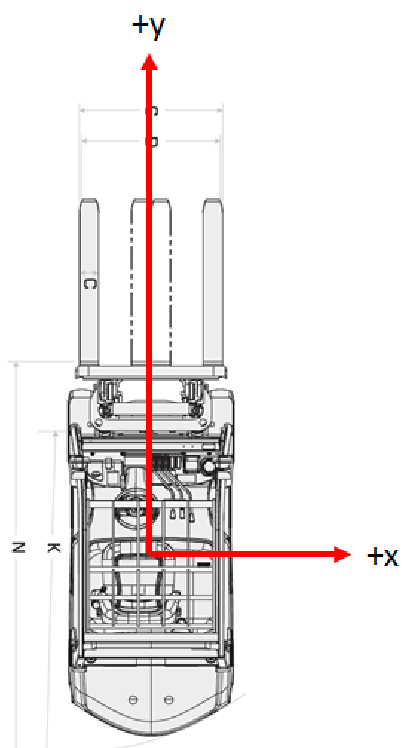
When recording the relative positions of tags on an item, the orientation of the axes should follow the right-hand rule as shown in Figure 1. Assuming that the asset is referenced to its x-y location and it is sitting on the floor at z=0, the x-y-z origin for the object should be defined such that the center of the object is at (0,0,0). The units (feet/meters) for the relative tag locations should follow the convention used when configuring ZAATS.

**Figure 1** Orientation of 3-dimensional Axis




For assets with an obvious forward orientation, such as a forklift, it is important that the direction of the forks be in the positive y direction and the positive x direction be in the direction going to the right of the forks, as shown in Figure 2. In this way, the direction of the forklift will indicate the direction of the forks. For example, if the forks are aligned with the y-axis of ZAATS, the direction of the forklift will be reported as 0 degrees. If the forklift is rotated to the right, the direction will be reported by ZAATS as 90 degrees.

**Figure 2** Forklift Orientation



## Recording Tag Locations on Fixed Assets

Note, when tagging assets in this manner with multiple tags, only a single Tag ID and x-y-z location are reported by ZAATS, the location being the estimation of actual asset location using all relevant tag reads associated with the individual asset. As stated earlier, the direction of the asset is also available if the system is configured to report it as metadata.

As an example, we consider the case were a Toyota Forklift model 8FGCUxx is tagged with eight RFID tags in the locations shown in Figure 3. The tag placement geometry of this particular forklift is summarized in Table 8.

**Figure 3** Example of a Forklift as a Multi-tagged Asset


**Table 8** Example of Tag Locations on Multi-Tagged Item

| Multi-Tag Object Type | Toyota 8FGCUxx | | |
|---|---|---|---|
| Multi-Tag Object ID | 0xabc | | |
| Tag Location ID and Location (Feet) | X | Y | Z |
| 1 | -1.75 | +3.25 | +6.725 |
| 2 | +1.75 | +3.25 | +6.725 |
| 3 | -1.75 | -3.25 | +6.725 |
| 4 | +1.75 | -3.25 | +6.725 |
| 5 | -1.75 | -1.00 | +2.00 |
| 6 | +1.75 | -1.00 | +2.00 |
| 7 | -1.50 | -4.50 | +4.00 |
| 8 | +1.50 | -4.50 | +4.00 |

Details of how to define the numbering and data structure for the Multi-Tag Object ID are described in the Guidelines for Encoding Tags section. The process of defining each Multi-Tag Object Type with commensurate Tag Locations should be repeated for each Multi-Tagged item.

# Decide on an Encoding Type

Presumably, the encoding type has been defined based on organizational requirements and choice of numbering methodologies. It is important that whatever encoding methods, GS1 TDS or ISO, are used consistently for all assets (i.e. single tagged assets and multi-tagged assets) in a deployment.

# Guidelines for Encoding Tags

## Overview of Multi-Tag Encoding

ZAATS differentiates between a single tag and multi-tag asset via information contained in fixed and known locations within the tag memory data structure based on the encoding type (GS1 or ISO). For GID-96 and ISO, ZAATS uses the Object Class field for this purpose. For GIAI-96, ZAATS uses 16 bits of the Individual Asset Reference to serve as an Object Class, with the remainder of the bits used for Serial Number or zero fill. The Multi-Tag Object ID and Tag Location ID shown in Table 8 are both encoded into two sub-fields of the appropriate Object Class.

The Multi-Tag Object ID is a 12-bit value that indicates a particular class of multi-tagged object. For example, a specific make and model of a forklift. As an example, the Object ID 0xabc may be used to identify a Toyota model 8FGCUxx forklift and the Object ID 0xdef may be used to identify a Crown model SC604x forklift. All Multi-Tag objects of the same class will have the same Multi-Tag Object ID which indicates that they have the same geometry. This is how ZAATS knows how to interpret the tags based on the relative locations of all tags associated with the particular class of multi-tagged object.

For each particular class of a multi-tagged object, the locations of the multiple tags are indicated by the Tag Location ID. Valid values are between 1 and 15, inclusive (allowing up to 15 tags per item). When configuring ZAATS with multi-tag information, the Tag Location ID maps to the relative location of the tag on the Multi-Tag Object. The number 0 is reserved and will be used when ZAATS reports the ID and location of the object as a whole.

**NOTE:** All single-tag objects are assigned an Object Class with the value 0x0000. That way ZAATS only applies multi-tag logic to the appropriate assets. This true for both GS1-based and ISO-based encoding methods.

Finally, to uniquely identify different objects of the same Multi-Tag Object class, a serial number must be encoded in the tag ID (i.e. the Serial Number of Individual Asset Reference of the EPC or UII). There may be many multi-tagged objects with the same Multi-Tag Object Class that exist in ZAATS. The Serial Number distinguishes each multi-tagged object of the Multi-Tag Object Class from each other. It can be thought of as a license plate for each item within a class of Multi-Tag objects.

## Encoding with GIAI-96

When using the GS1 Tag Data Standard, each tag must be encoded as per the standard. As described in the Gen2 Background section of the Overview chapter, the Toggle bit in the PC section of the EPC must be set to 0.

When using GIAI-96 for the purposes of multi-tag RTLS, as explained earlier, the Individual Asset Reference of the GIAI-96 structure includes information to identify each specific asset; i.e. the unique combination of Multi-Tag Object ID, Tag Location ID, and Serial Number. Due to the variable length of the Individual Asset Reference in GIAI-96, this information must always be encoded in the shortest possible Individual Asset Reference length of 42 bits. For Partition values greater than 0 (i.e. Individual Asset Reference greater than 42 bits), the most significant bits must be padded with zeros. Finally, each multi-tagged item has a unique serial number (assigned by the end user). A total of $2^{24}$ serial numbers (approximately 16 million) per object class are available. This should be more than sufficient for any organization to uniquely identify their RFID tagged fixed assets. The format of the Individual Asset Reference field of the GIAI-96 is shown in Figure 4.

**Figure 4** Individual Asset Reference Field of the GIAI-96 for Multi-Tag Items

| Scheme | GIAI-96 | | | | |
|---|---|---|---|---|---|
| Field Description | EPC Header | Filter | GIAI | | |
| | | | Partition | Company Prefix | Individual Asset Reference |
| # of Bits | 8 | 3 | 3 | 20-40 | 62-42 |
| | 0011 0100 | 000 | See Table 5 | | |

| GIAI-96 Individual Asset Reference | | | | |
|---|---|---|---|---|
| Field Description | Zero Fill | Multi-Tag Object ID | Tag Location ID | Serial Number |
| # of Bits | 22-2 | 12 | 4 | 24 |
| | 0 | 0x000: Reserved for Single tag objects | 0x0: Reserved for reporting composite object | |

As an example, assume that an end-user with GS1 Company Prefix 0x1122334455 uses Toyota 8FGCUxx forklifts in their operation and decides to track these fixed assets using the multi-tag capability of ZAATS. Based on the forklift geometry and need for high accuracy location tracking with directionality, the end-user decides to use 8 RFID tags per forklift (see Figure 3 and Table 8). Below is a summary of a specific forklift belonging to that class:

Partition Value: 0

GS1 Company Prefix: 0x1122334455

Multi-Tag Object ID for Toyota 8FGCUxx forklift: 0xabc

8 tags affixed to the object (to all objects of the same class)

Serial Number for the specific forklift: 0xaabbcc

The previous example would result in the following tag encodings for the eight RFID tags affixed to the specific locations of the forklift:

**Table 9** Forklift Tag Encodings

| Tag | Full EPC ID | Components of EPC | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Individual Asset Frame | | | |
| | | EPC Header | Filter | Partition | GS1 Company Prefix | Reserv ed | Multi-Tag Object ID | Tag Location ID | Serial Number |
| | 96 bits | 8 bits | 3 bits | 3 bits | 40 bits | 4 bits | 12 bits | 4 bits | 24 bits |
| 1 | 0x34004488cd1154abc1aabbcc | 34 | 0 | 0 | 1122334455 | 0 | abc | 1 | aabbcc |
| 2 | 0x34004488cd1154abc2aabbcc | 34 | 0 | 0 | 1122334455 | 0 | abc | 2 | aabbcc |
| 3 | 0x34004488cd1154abc3aabbcc | 34 | 0 | 0 | 1122334455 | 0 | abc | 3 | aabbcc |
| 4 | 0x34004488cd1154abc4aabbcc | 34 | 0 | 0 | 1122334455 | 0 | abc | 4 | aabbcc |
| 5 | 0x34004488cd1154abc5aabbcc | 34 | 0 | 0 | 1122334455 | 0 | abc | 5 | aabbcc |
| 6 | 0x34004488cd1154abc6aabbcc | 34 | 0 | 0 | 1122334455 | 0 | abc | 6 | aabbcc |

**Table 9** Forklift Tag Encodings

| Tag | Full EPC ID | Components of EPC | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | | | Individual Asset Frame | | | |
| | | EPC Header | Filter | Partition | GS1 Company Prefix | Reserv ed | Multi-Tag Object ID | Tag Location ID | Serial Number |
| | 96 bits | 8 bits | 3 bits | 3 bits | 40 bits | 4 bits | 12 bits | 4 bits | 24 bits |
| 7 | 0x34004488cd1154abc7aabbcc | 34 | 0 | 0 | 1122334455 | 0 | abc | 7 | aabbcc |
| 8 | 0x34004488cd1154abc8aabbcc | 34 | 0 | 0 | 1122334455 | 0 | abc | 8 | aabbcc |
| See note | 0x34004488cd1154abc0aabbcc | 34 | 0 | 0 | 1122334455 | 0 | abc | 0 | aabbcc |

**NOTE:** The last item in the table is not a specific tag. It illustrates the reported EPC of the composite object that contains the Multi-Tag Object ID 0xabc, '0' as the Tag Location ID, and 0xaabbcc as the Serial Number of the specific forklift.

## Encoding with GID-96

When using the GS1 Tag Data Standard, each tag must be encoded as per the standard. As described in the Gen2 Background section of the Overview chapter, the Toggle bit in the PC section of the EPC must be set to 0.

When using GID-96 for the purposes of multi-tag RTLS, the Multi-Tag Object ID and the Tag Location ID are encoded directly into the Object Class of the GID-96 structure. The Object Class field will be zero filled with 8 bits to conform to the 24-bit Object Class field size.

The format of the Individual Asset Reference field of the GID-96 is shown in Figure 5.

**Figure 5** Object Class Field of the GID-95 for Multi-Tag Items

| Scheme | GID-96 | | | |
|---|---|---|---|---|
| Field Description | EPC Header | General Manager Number | Object Class | Serial Number |
| # of Bits | 8 | 28 | 24 | 36 |
| | 0011 0101 | See below. | See below. | See below. |

| GID-96 Object Class | | | |
|---|---|---|---|
| Field Description | Zero-Fill | Multi-Tag Object ID | Tag Location ID |
| # of Bits | 8 | 12 | 4 |
| | 0x00 | 0x000: Reserved for Single tag Objects | 0x0: Reserved for reporting composite object |

For the purposes of the multi-tag RTLS, each multi-tagged item has a unique serial number.

As an example:

General Manager Number: 0x1111222

Multi-Tag Object ID for Toyota 8FGCUxx forklift: 0xabc

8 tags affixed to the object

Serial Number for this particular Multi-Tag Object ID: 0xaaaabbbbc

Results in the following tag encodings:

**Table 10** GID-96 Tag Encodings

| Tag | Full EPC ID | Components of EPC | | | | |
|---|---|---|---|---|---|---|
| | | | | Object Class | | |
| | | EPC Header | General Manager Number | Multi Tag Object ID | Tag Location ID | Serial Number |
| | 96 bits | 8 bits | 28 bits | 20 bits | 4 bits | 36 bits |
| 1 | 0x35111122200abc1aaaabbbbc | 35 | 1111222 | 00abc | 1 | aaaabbbbc |
| 2 | 0x35111122200abc2aaaabbbbc | 35 | 1111222 | 00abc | 2 | aaaabbbbc |
| 3 | 0x35111122200abc3aaaabbbbc | 35 | 1111222 | 00abc | 3 | aaaabbbbc |
| 4 | 0x35111122200abc4aaaabbbbc | 35 | 1111222 | 00abc | 4 | aaaabbbbc |
| 5 | 0x35111122200abc5aaaabbbbc | 35 | 1111222 | 00abc | 5 | aaaabbbbc |
| 6 | 0x35111122200abc6aaaabbbbc | 35 | 1111222 | 00abc | 6 | aaaabbbbc |
| 7 | 0x35111122200abc7aaaabbbbc | 35 | 1111222 | 00abc | 7 | aaaabbbbc |
| 8 | 0x35111122200abc8aaaabbbbc | 35 | 1111222 | 00abc | 8 | aaaabbbbc |
| See note | 0x35111122200abc0aaaabbbbc | 35 | 1111222 | 00abc | 0 | aaaabbbbc |

**NOTE:** When reporting Tag ID and location of a multi-tagged asset, the reported EPC of the composite object contains the Multi-Tag Object ID, ‘0’ as the Tag Location ID, and the Serial Number of the object.

## Encoding with ISO

When using ISO encoding, as mentioned in the Gen2 Background section of the Overview chapter, the Toggle bit in the PC section of the EPC shall be set to 1 and the remaining 8 bits of the PC (the AFI) shall be set appropriately.

When using ISO encoding for the purposes of RTLS, the ZAATS ID is a unique identifier that identifies the end user system. The ZAATS Object Class includes information to identify the Multi-Tag Object ID of the object and the Tag Location ID.

When using ISO 96-bit encoding for the purposes of multi-tag RTLS, the Multi-Tag Object ID and the Tag Location ID are encoded directly into the Object Class of the ISO structure (without zero fill).

The format of the Object Class for ISO encoding will be as follows:

**Figure 6** Object Class ISO Encoding Format

| Scheme | ISO Closed-System (96 bit) | | |
|---|---|---|---|
| Field Description | ZAATS ID | Object Class | Serial Number |
| # of Bits | 16 | 16 | 64 |
| | | | |

| ZAATS Object Class | | |
|---|---|---|
| Field Description | Multi-Tag Object ID | Tag Location ID |
| # of Bits | 12 | 4 |
| | 0x000: Reserved for Single tag Objects | 0x0: Reserved for reporting composite object |

For the purposes of the multi-tag RTLS, each multi-tagged item has a unique serial number.

As an example:

ZAATS ID: 0x2eb1

Multi-Tag Object ID for Toyota 8FGCUxx forklift: 0xabc

8 tags affixed to the object

Serial Number for this particular Multi-Tag Object ID: 0x0123456789abcdef

Results in the following tag encodings:

**Table 11** Tag Encodings for ISO

| Tag | Full EPC ID | Components of EPC | | | |
|---|---|---|---|---|---|
| | | ZAATS ID | ZAATS Object Class | | Serial Number |
| | | | Multi-Tag Object ID | Tag Location ID | |
| | 96 bits | 16 bits | 12 bits | 4 bits | 64 bits |
| 1 | 0x2eb1abc10123456789abcdef | 2eb1 | abc | 1 | 0123456789abcdef |
| 2 | 0x2eb1abc20123456789abcdef | 2eb1 | abc | 2 | 0123456789abcdef |
| 3 | 0x2eb1abc30123456789abcdef | 2eb1 | abc | 3 | 0123456789abcdef |
| 4 | 0x2eb1abc40123456789abcdef | 2eb1 | abc | 4 | 0123456789abcdef |
| 5 | 0x2eb1abc50123456789abcdef | 2eb1 | abc | 5 | 0123456789abcdef |
| 6 | 0x2eb1abc60123456789abcdef | 2eb1 | abc | 6 | 0123456789abcdef |
| 7 | 0x2eb1abc70123456789abcdef | 2eb1 | abc | 7 | 0123456789abcdef |
| 8 | 0x2eb1abc80123456789abcdef | 2eb1 | abc | 8 | 0123456789abcdef |
| See note | 0x2eb1abc00123456789abcdef | 2eb1 | abc | 0 | 0123456789abcdef |

**NOTE:** When reporting Tag ID and location of a multi-tagged asset, the reported EPC of the composite object contains the Multi-Tag Object ID, '0' as the Tag Location ID, and the Serial Number of the object.

# Configure CLAS With the Encoding Type and Tag Locations for Each Asset Type/Class

How CLAS is configured to specify the encoding type and company-specific identifier was described in the previous chapter. If using the multi-tag feature of ZAATS, then after all asset types and classes have been identified and data structures defined, the system will need to be aware that these classes exist in order to report asset location data accordingly. The following list of parameters should be configured into the CLAS software:

- Multi-Tag Encoding Type: GS1 GIAI-96, GS1 GID-96, or ISO-96 (Must be ISO-96 if ISO-96 is used for single tag encoding)
- Multi-Tag Object ID and Tag locations: One for each asset class.

Please consult the CLAS API Developer Guide, (p/n MN-003198-xx) for information on how the specific parameters are configured into the CLAS software.