The Alien® Higgs™4 integrated circuit (ICs) supports a range of optional serialization schemes. These use unique numbers programmed into Higgs-4 by Alien at IC manufacturing time and result in minimizing serialization efforts across large enterprises.

**Problem**

- Customers want relief from complex enterprise wide serialization challenges
- There is a need to have a common serialization scheme across multiple sites, converter machines and suppliers
- Implement this today, with current technology
- Brand-owners, who own serialization, could benefit from understanding the implementation options available to them

**Background:**

The Electronic Product Code (EPC) is designed as a universal identifier that provides a unique identity for every physical object anywhere in the world, for all time (the product and the instance of that product). Its structure is defined in the EPCglobal Tag Data Standard (see [http://www.gs1.org/gsmp/kc/epcglobal/tds](http://www.gs1.org/gsmp/kc/epcglobal/tds)). For the purpose of this paper, we simplify the EPC into two conceptual parts. The first identifies the product type or Stock Keeping Unit (SKU). The other portion is a serial number that defines a unique number for that product instance. More simply put, there is portion that is a number that represents the product (e.g. pair of blue socks, medium size) and a portion that is a serial number that tells us that this is the blue medium sock number 12,476,996 (the unique instance number).

Bar codes have the product SKU but not the serial number. So the EPC can be thought of as:

![Figure 1 - GTIN and SGTIN](image)

**Figure 1 - GTIN and SGTIN**
Terminology:

- The non-changing (product) identifier is known as the Global Trade Item Number (GTIN). This only identifies the product type or stock-keeping unit (SKU) rather than an individual instance of a particular product type.
- The serial number must be added to the GTIN to identify the specific instance of that product within the EPC. This is theSerialized Global Trade Item Number (SGTIN).

If there is only one manufacturer of blue/medium socks and only one brand owner of the socks, adding the serial number is a simple affair of starting at zero and incrementing the serial number for every pair of socks produced. However, the complications occur when (in any or all combinations of):

- There are more than one supplier of the product (the brand owner is using multiple product sources)
- Each supplier is using multiple machines in parallel to manufacture that one product
- Each supplier has distributed manufacturing in different regions or even different countries
- The supplier may even sub-contract manufacturing themselves to multiple suppliers
- The supplier may be supplying the same product to more than one brand owner

The problem is ... how do you add a serial number, given any combination of real-world complexities described above...

- Without a single central database (since more than one organization is involved)
- Doing this efficiently
  - Assigning 50 million serial numbers to each machine may be inefficient if some machines are used more than others
  - Who assigns the serial number ranges? The brand owner may not be able to do this if the supplier is supplying the same product to more than one brand-owner.
- Without duplication of the serial number
  - What happens when machine 72 malfunctions? What serial number was last used? Which of the last 160 products were correctly programmed (on which serial number do I restart the machine to avoid duplication and not waste serial numbers)?
  - Two machines are started with the same or overlapping serial numbers. User error in entering a start number on one machine could result in hundreds of thousands of duplicates before the error is discovered.

These examples are not exhaustive, but meant to approximate some of the challenges involved.

The final challenge is that the serial portion of the SGTIN is only 38 bits (as defined by the EPCglobal Tag Data Standard). This provides a maximum of $2^{38}$ possible serial numbers (almost 275 billion). While this sounds a lot, if you are to start splitting this bit-range across a large supply chain, across multiple production lines, across multiple machines, the actual number of serial numbers useable becomes much smaller even with some complex central serial number management. Remember, that the specific supply chain, lines and machines may be different per product which can make serial management completely impractical or at least unwieldy (for example, what happens to the serial number scheme when a new supplier is added or a manufacturing line is removed?).

<table>
<thead>
<tr>
<th>Header</th>
<th>Filter</th>
<th>Partition</th>
<th>Company Prefix</th>
<th>Item Reference</th>
<th>Serial Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 bits</td>
<td>3 bits</td>
<td>3 bits</td>
<td>20-40 bits</td>
<td>24-4 bits</td>
<td>38 bits</td>
</tr>
</tbody>
</table>

Table 1 - EPC SGTIN-96 Format as Defined by the EPCglobal Tag Data Standard V1.6
The Challenge:

There is agreement in the industry, coordinated through the EPCGlobal group within GS1 (http://www.gs1.org/epcglobal/), that the “brand owner” owns the challenge of serialization.

The challenge is to provide a mechanism, as simple, efficient and as cost effective as possible, to allow brand-owners to add a serial number to the Electronic Product Code (EPC) of their products (creating SGTIN), without duplication.

Options:

There are two sources for the serial number, either from another part of the RFID chip that already contains a unique serial number, or from a computer managing these serial numbers on behalf of the brand-owner (known as “IT Based Serialization”). There are pro’s and con’s for using each approach and the solution selected may have more to do with whether the brand-owner wants to define the scheme and manage the complexities themselves or whether the brand-owner wishes to push that responsibility elsewhere. The responsibility may be pushed to a converter/printer or to the chip manufacturer (and the converter/printer may choose to use some of the chip serialization in a hybrid approach).

For non-IT based serialization, the serial number can be created by a 3rd party, such as printer/converter, or by the chip vendor, in this case, Alien.

Chip-Based Serialization:

For Alien customers using Higgs-4 IC, there are two “chip-based” serialization schemes that can be used.

- Use a pre-encoded serial number already in the EPC (Higg-4 only)
  - There is no need to keep a database to ensure serial numbers are unique since this is guaranteed by Alien at manufacturing time of the Higgs-4 IC.
  - Alien has agreed with a number of other RFID IC manufacturers to use Multi-vendor Chips based Serialization (MCS). This is explained further below.
- Create a new serial number from an internal unique serial number that resides outside of the EPC memory
  - Portions of a unique serial number (not yet present in the EPC fields), can be manually copied by the printer/converter to the EPC.
  - The printer/converter will need to manage how this is accomplished. This paper described what needs to be done to create this.

Pre-encoded Chip-Based Serialization:

The Higgs-4 IC provides an optional\(^1\) solution to serialization that requires minimal overhead and removes the need for a brand-owner to manage a database of serial numbers. In order to understand the details we will first discuss the memory available in Higgs-4 IC (and for completeness we show Higgs-3 IC as well).

---

\(^1\) Optional since it may be overwritten by the brand-owner if they do not wish to use pre-encoded chip-based serialization.
Alien’s Higgs-4 IC’s contains two banks of memory that are relevant to the conversation.

- The EPC bank (shown green below), specifically the EPC# field within it (shown dark green), is where we would like to create an EPC number that contains the Serialized Global Trade Item Number (SGTIN). This is owned by the brand owner.
- The TID bank (light orange), specifically the UTID (Unique Tag ID – dark orange) which is a 64 bit pre-encoded serial number that is guaranteed by Alien to be unique. Other IC manufacturers may have a similar field or fields, sometimes smaller or scattered around multiple locations within their chip. So, Alien’s serial number is unique compared to any other Alien chip but there may be a similar serial number from another chip vendor i.e. it is unique to Alien chips only.

<table>
<thead>
<tr>
<th>Bank</th>
<th>Description</th>
<th>Higgs-3 Bank Address</th>
<th># of Bits</th>
<th>Higgs-4 Bank Address</th>
<th># of Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>User</td>
<td>User</td>
<td>00h – 1FFh</td>
<td>512</td>
<td>00h – 7Fh</td>
<td>128</td>
</tr>
<tr>
<td>TID</td>
<td>Device Configuration</td>
<td>60h – BFh</td>
<td>96</td>
<td>60h – BFh</td>
<td>96</td>
</tr>
<tr>
<td></td>
<td>Unique Tag ID Unalterable</td>
<td>20h – 5Fh</td>
<td>64</td>
<td>20h – 5Fh</td>
<td>64</td>
</tr>
<tr>
<td></td>
<td>TID EPC/TMD/TMDID/TMN</td>
<td>00h – 1Fh</td>
<td>32</td>
<td>00h – 1Fh</td>
<td>32</td>
</tr>
<tr>
<td>EPC</td>
<td>EPC #</td>
<td>20h – 7Fh</td>
<td>96</td>
<td>20h – 9Fh</td>
<td>128</td>
</tr>
<tr>
<td></td>
<td>EPC-PC</td>
<td>10h – 1Fh</td>
<td>16</td>
<td>10h – 1Fh</td>
<td>16</td>
</tr>
<tr>
<td></td>
<td>EPC-CRC</td>
<td>00h – 0Fh</td>
<td>16</td>
<td>00h – 0Fh</td>
<td>16</td>
</tr>
<tr>
<td>Reserved</td>
<td>RES-Access Pwd, EPC optional</td>
<td>20h – 3Fh</td>
<td>32</td>
<td>20h – 3Fh</td>
<td>32</td>
</tr>
<tr>
<td></td>
<td>RES-Kill Pwd</td>
<td>00h – 1Fh</td>
<td>32</td>
<td>00h – 1Fh</td>
<td>32</td>
</tr>
</tbody>
</table>

The concept is simple. The goal is to take the product SKU (more accurately the GTIN) and add a serial number (which is 38 bits by definition of the standard) to create the SGTIN (serialized GTIN) in the EPC. The GTIN is defined by the product and does not change from tag to tag (for the same product). If we have a unique number in the serial portion for all Alien tags (regardless of the GTIN) then the combination of GTIN plus this serial number is also unique. In reality, with $2^{38}$ bits, after 275 billion products, the numbers will restart and a duplicate will be created. However, it is unlikely this will happen in the lifetime of most products in the market today.

As Alien *already* has a unique serial number in the Higgs-4 IC (in the UTID), Alien is providing the EPC pre-encoded (pre-programmed) with the lower bits from the UTID serial number in the EPC (with one addition, see MCS below). During manufacturing time of the IC, the UTID serial number has the lower bits copied into the EPC. This means that the only thing the “customer” of the tag has to do is to copy the SKU (non-changing portion or GTIN) into the tag. To speed this up, Alien also provides a couple of additional Higgs-4 capabilities:

- **BlastWrite™** - Writes a constant bit pattern of any word length (such as the product SKU or GTIN) starting at any word aligned memory location (in this case the static EPC fields) to multiple-chips in one command. This allows the GTIN to be written to 50 or 100 tags in one command. Alien has tested BlastWrite writing at a rate equivalent to 3,600 tags per minute. This assumes that all writes are word-aligned.
- **QuickWrite™** - Writes a bit pattern of any word length starting at any word aligned memory location (to a single chip)
So, the Higgs-4 IC comes pre-encoded with the serial number and BlastWrite and QuickWrite are used to quickly set the non-changing part of the EPC speeding up and simplifying the serialization of tags and ICs. To simplify the discussion above, we conveniently simplified the serial number to a single field. The Higgs-4 IC actually has a two part serial number as defined by the Multi-vendor Chips based Serialization (MCS) schema.

**Using the Pre-Encoded Higgs-4 IC Multi-vendor Chips-based Serialization (MCS)**

The description above purposely did not mention MCS which is included as part of the “serial portion” of the EPC when encoded by Alien. The serial number provided by Alien in the UTID is unique to Alien but not to other vendors. To further assist customers using chip-based serialization, several UHF RFID IC vendors, including Alien, have agreed to an approach that splits a small header that defines who the chip vendor is with the serial number described above. The header is 3 bits which leaves 35 bits for the serial portion. This modifies Table 1 to look as follows:

<table>
<thead>
<tr>
<th>Header</th>
<th>Filter</th>
<th>Partition</th>
<th>Company Prefix</th>
<th>Item Reference</th>
<th>Serial Number</th>
</tr>
</thead>
<tbody>
<tr>
<td>8 bits</td>
<td>3 bits</td>
<td>3 bits</td>
<td>20-40 bits</td>
<td>24-4 bits</td>
<td>38 bits</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Non Changing Product SKU Portion (GTIN)**

The 38 bit MCS serial number (including the MCS header) allows brand-owners to source their chips from more than one passive UHF RFID vendor and use the same serialization scheme and have confidence that the serial number used will remain unique regardless of the IC source (as long as the source is an MCS compliant supplier). The MCS table that is used to look-up the supplier (as of May 2012) is defined as follows (new vendors may be added over time):

<table>
<thead>
<tr>
<th>Number Range</th>
<th>Usage</th>
</tr>
</thead>
<tbody>
<tr>
<td>000 (open)</td>
<td>Can be allocated to any number sources. Ranges can be merged or further sub-divided for chip or IT sources. (5/8 of total space)</td>
</tr>
<tr>
<td>001 (open)</td>
<td></td>
</tr>
<tr>
<td>010 (open)</td>
<td></td>
</tr>
<tr>
<td>011 (open)</td>
<td></td>
</tr>
<tr>
<td>100 (open)</td>
<td></td>
</tr>
<tr>
<td>101 (Impinj)</td>
<td>101 &lt;followed by 35 serial bits from Impinj chip&gt;</td>
</tr>
<tr>
<td><strong>110 (Alien)</strong></td>
<td><strong>110 &lt;followed by 35 serial bits from Alien chip&gt;</strong></td>
</tr>
<tr>
<td>111 (NXP)</td>
<td>111 &lt; followed by 35 serial bits from NXP chip&gt;</td>
</tr>
</tbody>
</table>

*Table 4- MCS Look-up Table to that Defines the 3-bit Header*
As of the time of writing this whitepaper, only Alien is **pre-encoding** the MCS header and serial number in to chips (Higgs-4 IC only). Some vendors do not create the EPC serialization at manufacturing time and leave the creation of the MCS header and serial number to the printer/converter. Use of the MCS compliant serial number provided by Alien is optional. If the brand-owner wishes to use a different scheme, they can simply overwrite the EPC serial number with something different.

So in summary, the MCS compliant serial number creation looks like this:

---

### Creating a Custom (Non-Pre-Encoded) Chip Based Serial Number in the EPC

Some vendors or brand-owners may wish to create their own (EPC based SGTIN) serial number scheme (not use MCS) but still use the fact that Alien chips are provided with a unique serial number in the UTID. An example may be that the brand-owner prefers a 32 bit serial number as this makes writing to tags more efficient. The following describes which bits need to be extracted from the UTID and placed into the EPC# field of the EPC bank to achieve this.
In Higgs-4, all that is necessary is to pull out an existing serial number from the low 32 to 38 bits from the UTID (Unique tag ID, part of bank 2, which is called the TID bank) and write these into the low-order bits of the EPC. The UTID words are words 2, 3, 4 and 5, starting with word zero at the beginning of the bank. The user should reserve some bits in the serial number to allow for different tag manufacturers. Using more bits will allow more time until the serialization scheme rolls over, even 32 bits will be unique for years.

The creation of a custom chip-based serial number looks like this:

**Conclusion**

Alien provides a suite of options for tag serialization for enterprise wide tag management:

- An optional **pre-encoded** Multi-Chip vendor Serialization (MCS) schema for the Higgs-4 IC;
- The capability for a converter/printer to create a custom serialization scheme but leveraging from the unique UTID serial number in the Alien Higgs-4 IC, or
- To create a separate IT based serialization scheme