Application Report
SLOA166A–February 2012–Revised June 2014

# Using Texas Instruments Tag-it™ HF-I Transponder Technology for NFC Vicinity Applications

*Joshua Wyatt, Kostas Aslanidis, Ulrich Denk, Klemens Sattlegger* *Embedded RF & MCU-RF*

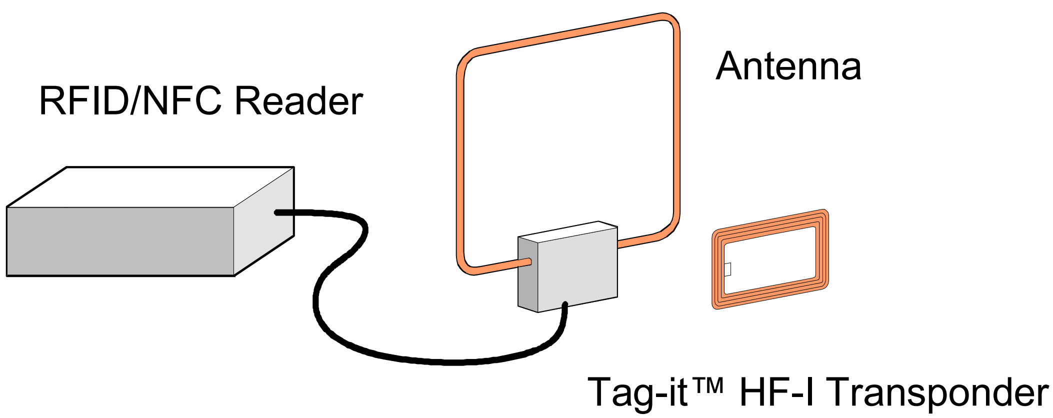
**ABSTRACT**

Texas Instruments' Tag-it™ HF-I family of transponders and silicon (inlays, encapsulated devices, and wafers) consist of 13.56-MHz high-frequency (HF) devices that are compliant with the ISO/IEC15693 and ISO/IEC18000-3 (Mode 1) global open standards.

The NFC Forum has specified the NFC data exchange format (NDEF) for using tag memory to store text, a uniform resource identifier (URI), or images (portable network graphics (PNG) or joint photographic experts group (JPEG) file types). This application report is intended to guide users and developers who are interested in also using them for NFC vicinity applications.

The Tag-it HF-I transponder inlay and silicon wafer offerings are well suited for a variety of applications including but not limited to: product authentication, library applications, supply chain management, asset management, and ticketing/stored value applications.

Texas Instruments TRF796x and TRF797x family of RFID/NFC reader/writer ICs coupled with any of the broad portfolio of Texas Instruments microcontrollers (that is, MSP430™, C2000™, Stellaris®, Sitara™, DaVinci™, or OMAP™) can be used with the above mentioned transponders, inlays, or transponder ICs to create complete RFID/NFC system.



**Contents**



**List of Figures**



**List of Tables**



# 1 Tag-it HF-I Memory Maps

## *1.1 Plus Devices*

The Tag-it HF-I Plus Transponder IC offers a full 2048 bits (256 bytes) of user memory, plus a 64 bit UID, DSFID and AFI fields, along with factory and user lock bits. User data is written to and read from memory blocks, which use a non-volatile EEPROM silicon technology. The user memory is organized in 64 individual 4 byte blocks (memory locations 0-63). Each block is separately programmable by the factory or the user and can be locked to protect data from modification. Once the data has been locked, it cannot be changed. The memory map for this device is shown in Figure 1.

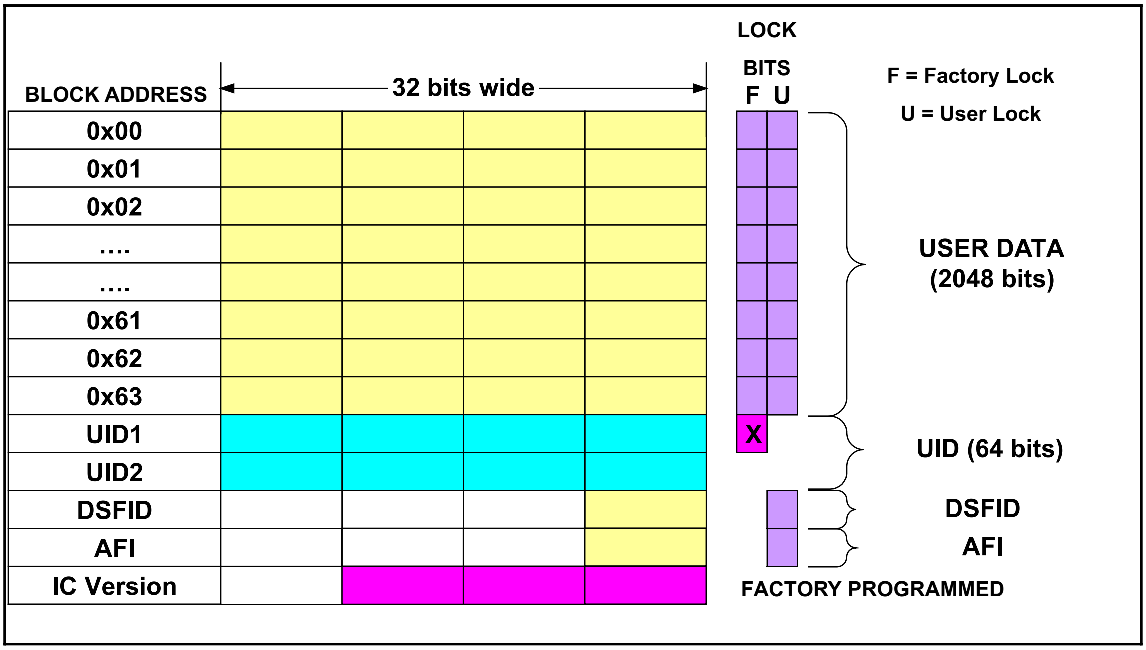



**Figure 1. Tag-it HF-I Plus Memory Map**

## 1.2 Pro Devices (with password write protect and kill features)

The Tag-it HF-I Pro Transponder IC offers a full 256 bits (32 bytes) of user memory, a 64 bit UID, an AFI field, plus a Password field along with factory and user lock bits. User data is written to and read from memory blocks, which use a non-volatile EEPROM silicon technology. The user memory is organized in eight individual 4 byte blocks (memory locations 0-7). Each block is separately programmable by the factory or the user and can be locked to protect data from modification. Once the password field is written and locked, data in blocks that have been locked can be changed by using the Write Block with Password Command. The memory map for this device is shown in Figure 2. This device also has a Kill command implemented for sensitive applications with “life cycles”, so it can be rendered completely unreadable.

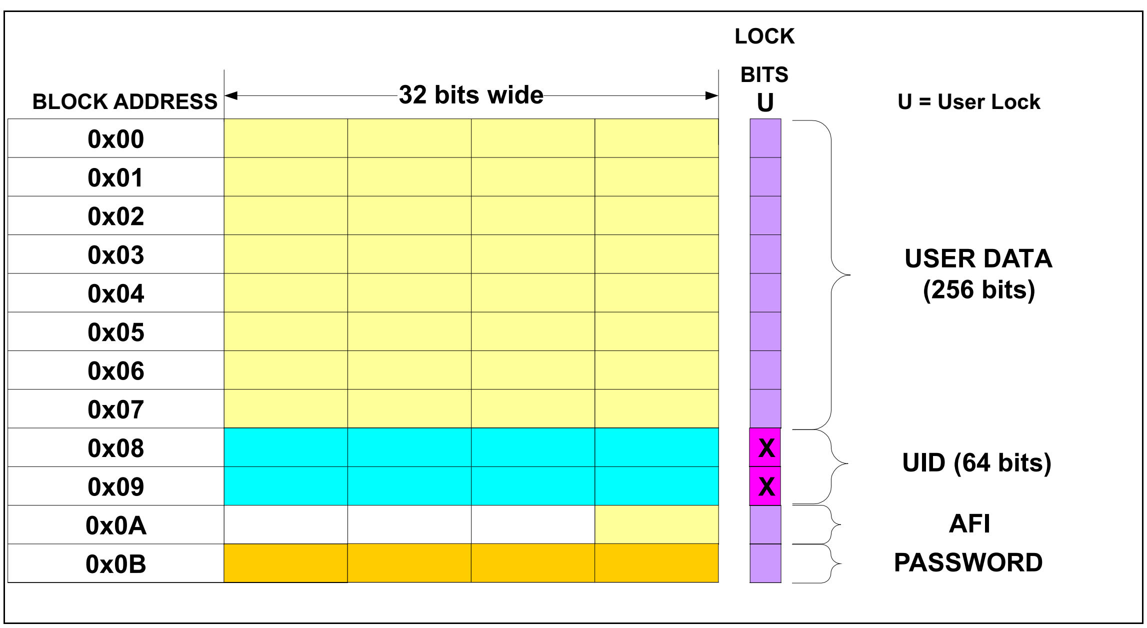



**Figure 2. Tag-it HF-I Pro Memory Map**

### 1.3 Standard Devices

The Tag-it HF-I Standard Transponder IC also offers a full 256 bits (32 bytes) of user memory, a 64 bit UID, AFI field, along with factory and user lock bits. User data is written to and read from memory blocks, which use a non-volatile EEPROM silicon technology. The user memory is organized in eight individual 4 byte blocks (memory locations 0-7). Each block is separately programmable by the factory or the user and can be locked to protect data from modification. Once the data has been locked, it cannot be changed. The memory map for this device is shown in Figure 3.

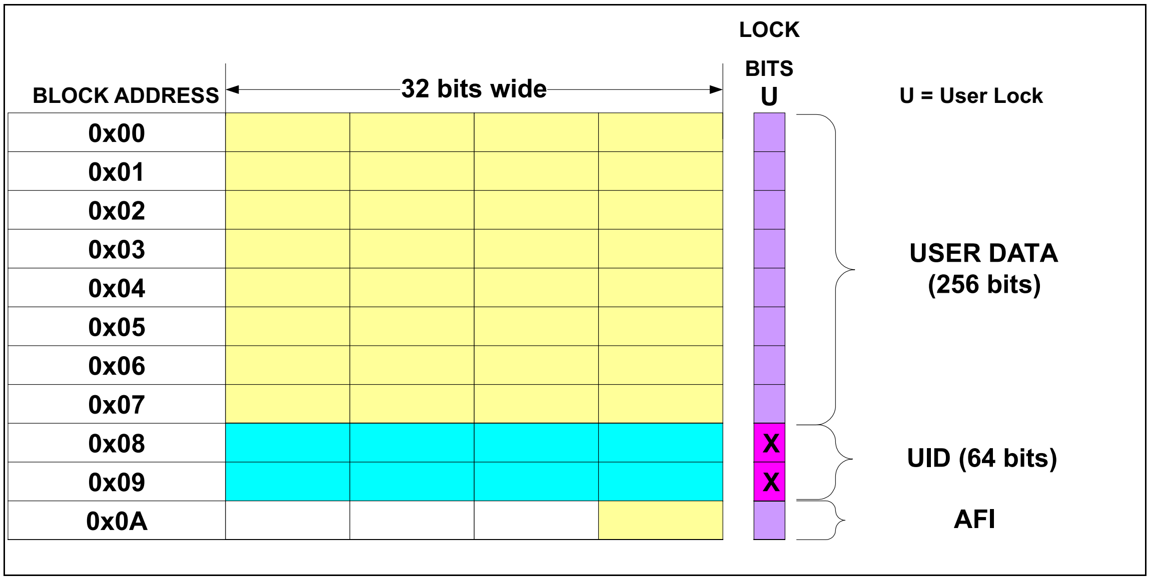



**Figure 3. Tag-it HF-I Standard Memory Map**

## 2 Tag-it HF-I Unique Identification (UID) Coding

The 64 bit UID is defined by the ISO15693 standard and contains two mandatory bytes to indicate that it is an ISO15693 device (MSByte = 0xE0) and the manufacturer of the silicon (next MSByte = 0x07 = TI, see ISO7816-6 for full list of silicon vendors recognized).

Texas Instruments UID coding complies and also uses the third most significant byte to indicate the product ID/functionality, as shown in Figure 4.

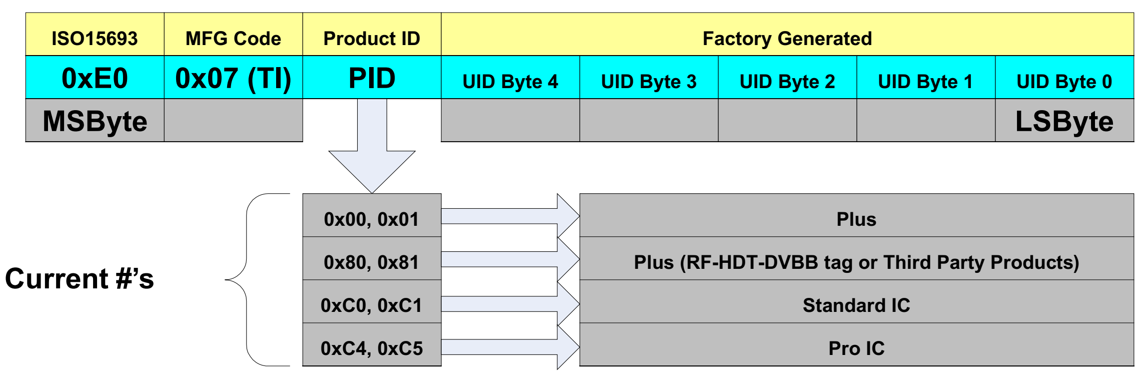



**Figure 4. Tag-it HF-I UID Coding (all products)**

# 3 Tag-it HF-I User Memory Contents (as shipped)

Texas Instruments, as part of the 100% final test of each transponder or IC wafer shipped, programs all user memory blocks, DSFID and AFI bytes to 0x00, then reads back and does a compare to ensure the memory cells are functional. This means that anyone wanting to utilize TI Tag-it HF-I transponders in an NFC application must first NDEF format the devices so they will be recognized.

# 4 Tag-it HF-I Implemented Commands

## 4.1 Plus Devices

**Table 1. Command Set for Tag-it HF-I Plus Transponders (1)**

| Request | Request Code | Inventory | Addressed | Non Addressed | Select | AFI | OptionFlag |
|---|---|---|---|---|---|---|---|
| ISO 15693 Mandatory Commands | | | | | | | |
| Inventory | 0x01 | √ | - | - | - | √ | 0 |
| Stay Quiet | 0x02 | - | √ | - | - | - | 0 |
| ISO 15693 Optional Commands | | | | | | | |
| Read_Single_Block | 0x20 | √ | √ | √ | √ | √ | 0/1 |
| Write_Single_Block | 0x21 | - | √ | √ | √ | - | 1 |
| Lock_Block | 0x22 | - | √ | √ | √ | - | 1 |
| Read_Multi_Blocks | 0x23 | √ | √ | √ | √ | √ | 0/1 |
| Write_Multi_Blocks | 0x24 | - | - | - | - | - | - |
| Select Tag | 0x25 | - | √ | - | - | - | 0 |
| Reset to Ready | 0x26 | - | √ | √ | √ | - | 0 |
| Write_AFI | 0x27 | - | √ | √ | √ | - | 1 |
| Lock_AFI | 0x28 | - | √ | √ | √ | - | 1 |
| Write DSFID | 0x29 | - | √ | √ | √ | - | 1 |
| Lock DSFID | 0x2A | - | √ | √ | √ | - | 1 |
| Get_System_info | 0x2B | √ | √ | √ | √ | √ | 0 |
| Get_M_Blk_Sec_St | 0x2C | √ | √ | √ | √ | √ | 0 |
| TI Custom Commands | | | | | | | |
| Write_2_Blocks | 0xA2 | - | √ | √ | √ | - | 1 |
| Lock_2_Blocks | 0xA3 | - | √ | √ | √ | - | 1 |

(1) √ = Implemented, - = Not applicable, 0/1 = Option Flag needed

**NOTE:** The Option Flag (Bit 7) of the ISO/IEC15693-3 defined Request Flags must be set to 1 for all Write and Lock commands to respond properly. For reliable programming, a programming time ≥ 10 ms is recommended before you send the end of frame (EOF) to request the response from the Transponder.

## 4.2 Pro Devices (with password write protect feature)

**Table 2. Command Set for Tag-it™ HF-I Pro Transponders** (1)

| Request | Request Code | Inventory | Addressed | Non Addressed | AFI | OptionFlag |
|---|---|---|---|---|---|---|
| ISO 15693 Mandatory Commands | | | | | | |
| Inventory | 0x01 | √ | - | - | √ | 0/- |
| Stay Quiet | 0x02 | - | √ | - | - | 0/- |
| ISO 15693 Optional Commands | | | | | | |
| Read_Single_Block | 0x20 | - | √ | √ | - | -/1 |
| Write_Single_Block | 0x21 | - | √ | √ | - | -/1 |
| Lock_Block | 0x22 | - | √ | √ | - | -/1 |
| TI Custom Commands | | | | | | |
| Kill | 0xA4 | - | √ | - | - | -/1 |
| WriteSingleBlockPwd | 0xA5 | - | √ | - | - | -/1 |

(1) √ = Implemented, - = Not applicable, 0/1 = Option Flag needed

**NOTE:** The Option Flag (Bit 7) of the ISO/IEC15693-3 defined Request Flags must be set to 1 for all Write and Lock commands to respond properly. For reliable programming, a programming time ≥ 10 ms is recommended before you send the end of frame (EOF) to request the response from the Transponder.

## 4.3 Standard Devices

**Table 3. Command Set for Tag-it HF-I Standard Transponders** (1)

| Request | Request Code | Inventory | Addressed | Non Addressed | AFI | OptionFlag |
|---|---|---|---|---|---|---|
| ISO 15693 Mandatory Commands | | | | | | |
| Inventory | 0x01 | √ | - | - | √ | 0/- |
| Stay Quiet | 0x02 | - | √ | - | - | 0/- |
| ISO 15693 Optional Commands | | | | | | |
| Read_Single_Block | 0x20 | - | √ | √ | - | -/1 |
| Write_Single_Block | 0x21 | - | √ | √ | - | -/1 |
| Lock_Block | 0x22 | - | √ | √ | - | -/1 |

(1) √ = Implemented, - = Not applicable, 0/1 = Option Flag needed

**NOTE:** The Option Flag (Bit 7) of the ISO/IEC15693-3 defined Request Flags must be set to 1 for all Write and Lock commands to respond properly. For reliable programming, a programming time ≥ 10 ms is recommended before you send the end of frame (EOF) to request the response from the Transponder.

# 5 NDEF Formatting of Tag-it HF-I Transponders

Before any ISO15693 device can be used as NFC-V tag, it must be formatted appropriately so that it can be recognized as NFC-V transponder with NDEF capabilities. This can be accomplished simply by issuing a write single block (with the correct values, see below) to Block 0 of the transponder to be formatted.

The NDEF Capability Container is a four byte field in block 0 of TI Tag-it HF-I transponders. Within this block (Block 0), in the context of using it as an NFC tag, the individual bytes have meaning as shown below in Figure 5 and Table 4.

| | CC0 | CC1 | CC2 | CC3 |
|---|---|---|---|---|
| Block 0 | 0x00 | 0x00 | 0x00 | 0x00 |

**Figure 5. Unformatted Capability Container (Block 0)**

**Table 4. Capability Container Byte Definitions**

| Byte Name | Byte Number | Bit(s) | Function |
|---|---|---|---|
| CC0 | 0 | 0:07 | NDEF Message Present (Magic Number) |
| CC1 | 1 | 6:07 | Version Number (Major) |
| | | 4:05 | Version Number (Minor) |
| | | 2:03 | Read Access Condition (will always be 00b) |
| | | 0:01 | Write Access Condition (can be changed) |
| CC2 | 2 | 0:07 | Total User Memory Size of Transponder, including CCBlock. (CC2 (convert hex value to decimal first) x 8 = User Memory Size in bytes). |
| CC3 | 3 | 3:07 | RFU |
| | | 2 | 1: CC2 Overflow (Memory Size exceeds 2040 bytes) |
| | | | 0: CC2 value is sufficient |
| | | 1 | RFU |
| | | 0 | 1: IC supports Read Multiple Blocks Command (0x23) |
| | | | 0: IC does not support Read Multiple Blocks Command (0x23), must use Read Single Block Command (0x20) |

## 5.1 *Plus Devices*

Based on Table 4, the correct values to be programmed into Block 0 of the Plus transponder IC for NDEF Formatting the Capability Container are shown in Figure 6.

| | CC0 | CC1 | CC2 | CC3 |
|---|---|---|---|---|
| Block 0 | 0xE1 | 0x40 | 0x20 | 0x01 |
| | NDEF Message Present (Magic Number) | Version #, Read/Write Access Conditions | User Memory = CC2 value x 8 = 256 bytes | Supports Read Multiple Blocks |

**Figure 6. Formatted Capability Container Byte (Block 0)**

## 5.2 *Pro Devices (with password write protect feature)*

Based on Table 4, the correct values to be programmed into Block 0 of the Pro transponder IC for NDEF Formatting the Capability Container are shown in Figure 7.

| | CC0 | CC1 | CC2 | CC3 |
|---|---|---|---|---|
| Block 0 | 0xE1 | 0x40 | 0x04 | 0x00 |
| | NDEF Message Present (Magic Number) | Version #, Read/Write Access Conditions | User Memory = CC2 value x 8 = 32 bytes | Does not support Read Multiple Blocks |

**Figure 7. Formatted Capability Container Byte (Block 0)**

## 5.3 Standard Devices

Based on Table 4, the correct values to be programmed into Block 0 of the Standard transponder IC for NDEF Formatting the Capability Container are shown in Figure 8.

| | CC0 | CC1 | CC2 | CC3 |
|---|---|---|---|---|
| Block 0 | 0xE1 | 0x40 | 0x04 | 0x00 |
| | NDEF Message Present (Magic Number) | Version #, Read/Write Access Conditions | User Memory = CC2 value x 8 = 32 bytes | Does not support Read Multiple Blocks |

**Figure 8. Formatted Capability Container Byte (Block 0)**

## 5.4 Empty NDEF Message TLV and TLV Terminator

To complete the NDEF formatting procedure on the transponder, Block 1 must then also be filled with correct values to indicate that it has an empty NDEF message along with the terminator. A Type, Length and Value (TLV) data format is used here. See Table 5 for format field definitions and Table 6 type byte definitions to complete formatting.

**Table 5. TLV Byte Definitions**

| | Type | Length | Value |
|---|---|---|---|
| Length | 1 byte | 1:3 byte(s) | = Length indicated in previous byte |
| Description | TLV Block Type (see Table 6) | Length of NDEF Message (0x00 : 0xFFFE) (0xFFFF and above is RFU) | Record Header, NDEF Message Content and TLV Terminator (see Table 7) |

**Table 6. TLV Byte Definitions**

| TLV Byte Name | Value | Description |
|---|---|---|
| Null TLV | 0x00 | Pad |
| Lock Control TLV | 0x01 | |
| Memory Control TLV | 0x02 | |
| NDEF Message TLV | 0x03 | Transponder contains NDEF Message |
| Proprietary TLV | 0xFD | |
| Terminator TLV | 0xFE | End of Message |

The basic example of an NDEF formatted (but empty) Plus transponder is shown in Figure 9. A formatted, but also empty example of a Pro or Standard transponder, is shown in Figure 10.

| | CC0 | CC1 | CC2 | CC3 |
|---|---|---|---|---|
| Block 0 | 0xE1 | 0x40 | 0x20 | 0x01 |
| | NDEF Message Present (Magic Number) | Version #, Read/Write Access Conditions | User Memory = CC2 value x 8 = 32 bytes | Does not support Read Multiple Blocks |
| Block 1 | 0x03 | 0x00 | 0xFE | 0x00 |
| | NDEF Message Present | Empty | TLV Terminator | Start of Available Memory |

**Figure 9. NDEF Formatted, Empty Tag-it HF-I Plus Transponder**

| | CC0 | CC1 | CC2 | CC3 |
|---|---|---|---|---|
| Block 0 | 0xE1 | 0x40 | 0x04 | 0x00 |
| | NDEF Message Present (Magic Number) | Version #, Read/Write Access Conditions | User Memory = CC2 value x 8 = 32 bytes | Does not support Read Multiple Blocks |
| Block 1 | 0x03 | 0x00 | 0xFE | 0x00 |
| | NDEF Message Present | Empty | TLV Terminator | Start of Available Memory |

**Figure 10. NDEF Formatted, Empty Tag-it HF-I Pro or Standard Transponder**

### 5.5 NDEF Message Programming Example

In the previous sections, the basic NDEF formatting of the transponder was completed. The next step to take would be to program an NDEF message onto the transponder so it will be ready to serve as an NFC device in the field. Table 7 shows the Value Byte definitions as specified in the NFC Data Exchange Document (NDEF) under the heading of Record Header Layout.

**Table 7. Value (Record Header Layout) Byte Definitions**

| NAME | BIT Placement | Function |
|---|---|---|
| Message Begin (MB) | 1xxx xxxx (Bit 7, MSbit) | Indicates the start of the NDEF message |
| Message End (ME) | x1xx xxxx (Bit 6) | Indicates the end of the NDEF message |
| Chunk Flag (CF) | xx1x xxxx (Bit 5) | Indicates that the following message is either the first record chunk or a middle record chunk of a chunked payload. |
| Short Record (SR) | xxx1 xxxx (Bit 4) | Indicates the PAYLOAD_LENGTH field is a single byte (octet) |
| Short Record (IL) | xxxx 1xxx (Bit 3) | Indicates the ID_LENGTH field is present in the header as a single byte (octet), if this is a 0, this field is omitted from the header and the record |
| Type Name Format (TNF) | xxxx x111 (Bit 0:2, LSbit) | Indicates the structure of the value of the TYPE Field. (See Table 8.) |

**Table 8. Type Name Format (TNF) Field Values**

| Type Name Format (TNF) | Value (in binary and decimal) |
|---|---|
| Empty | 000 = 0 |
| NFC Forum well known type (NFC RTD) | 001 = 1 |
| Media Type as defined in RFC 2046 (RFC 2046) | 010 = 2 |
| Absolute URI as defined in RFC 3986 (RFC 3986) | 011 = 3 |
| NFC Forum external type (NFC RTD) | 100 = 4 |
| Unknown | 101 = 5 |
| Unchanged (see NDEF_1.0 document, section 2.3.3 – used for middle or terminating chunks) | 110 = 6 |
| RFU (Reserved) | 111 = 7 |

Building up the transponder user memory further as an NFC device properly formatted and with a URI record, the following examples can be shown, as seen in Figure 11 and Figure 12. The first example shows URI Record Type programmed on the transponder user memory for pointing towards http://www.google.com and the second example shows URI record type programmed on the transponder user memory for pointing towards http://www.ti.com.

Where (from Block 1, per NDEF and RTD_URI Specifications):

0x03 = NDEF Message Present, 0x13 = Length, 0xD1 = Record Header, 0x01 = Type Length, 0x0B = Payload Length, 0x55 = Record Type (U), 0x01 = URI Header ID = http://www., next ten bytes are 0x67, 0x6F, 0x6F, 0x67, 0x6C, 0x65, 0x2E, 0x63, 0x6F, 0x6D to represent “google.com” (in hex), then the TLV Terminator 0xFE, to end the message.

| | CC0 | CC1 | CC2 | CC3 |
|---|---|---|---|---|
| Block 0 | 0xE1 | 0x40 | 0x04 | 0x00 |
| | NDEF Message Present (Magic Number) | Version #, Read/Write Access Conditions | User Memory = CC2 value x 8 = 32 bytes | Does not support Read Multiple Blocks |
| Block 1 | 0x03 | 0x13 | 0xD1 | 0x01 |
| | NDEF Message Present | Length (19 bytes) | Record Header | Type Length |
| Block 2 | 0x0B | 0x55 | 0x01 | 0x67 |
| | Payload Length | Record Type U (URI) | URI Header Identifier | g |
| Block 3 | 0x6F | 0x6F | 0x67 | 0x6C |
| | o | o | g | l |
| Block 4 | 0x65 | 0x2E | 0x63 | 0x6F |
| | e | . | c | o |
| Block 5 | 0x6D | 0xFE | 0x00 | 0x00 |
| | m | TLV Terminator | Empty (Don’t Care) | Empty (Don’t Care) |

**Figure 11. URI Record Example for Tag-it HF-I Pro or Standard Transponders**

The following shows similar being done with the Plus transponder, where (also from Block 1, per NDEF and RTD_URI Specifications):

0x03 = NDEF Message Present, 0x0B = Length, 0xD1 = Record Header, 0x01 = Type Length, 0x07 = Payload Length, 0x55 = Record Type (U), 0x01 = URI Header ID = http://www., next six bytes are 0x74, 0x69, 0x2E, 0x63, 0x6F, 0x6D to represent “ti.com” (in hex), then the TLV Terminator 0xFE, to end the message.

| | CC0 | CC1 | CC2 | CC3 |
|---|---|---|---|---|
| Block 0 | 0xE1 | 0x40 | 0x20 | 0x01 |
| | NDEF Message Present (Magic Number) | Version #, Read/Write Access Conditions | User Memory = CC2 value x 8 = 32 bytes | Does not support Read Multiple Blocks |
| Block 1 | 0x03 | 0x0B | 0xD1 | 0x01 |
| | NDEF Message Present | Length (11 bytes) | Record Header | Type Length |
| Block 2 | 0x07 | 0x55 | 0x01 | 0x74 |
| | Payload Length | Record Type U (URI) | URI Header Identifier | t |
| Block 3 | 0x69 | 0x2E | 0x63 | 0x6F |
| | i | . | c | o |
| Block 4 | 0x6D | 0xFE | 0x00 | 0x00 |
| | m | TLV Terminator | Empty (Don’t Care) | Empty (Don’t Care) |

**Figure 12. URI Record Example for Tag-it HF-I Plus Transponder**

## 6 References


- ISO/IEC15693-2 (Vicinity Cards - Air Interface and Initialization) http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=39695 (this is ISO web store - link to obtain the spec)
- ISO/IEC15693-3 (Vicinity Cards - Anti-Collision and Transmission Protocol) http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=43467 (this is ISO web store - link to obtain the spec)
- ISO/IEC7816 (Register of IC Manufacturers) ISO Recognized Manufacturing IDs http://www.iso.org/iso/iso_catalogue/catalogue_tc/catalogue_detail.htm?csnumber=38780 (this is ISO web store - link to obtain the spec)
- NFC Data Exchange Format (NDEF) (NFCForum-TS-NDEF_1.0) http://www.nfc-forum.org/specs/spec_license (this is click through license agreement for non-members to obtain the spec, same for all the other licenses below).
- NFC Record Type Definition (NFCForum-TS-RTD_1.0) http://www.nfc-forum.org/specs/spec_license. Generic Control Record Type Definition (NFCForum-TS-GenericControl_RTD_1.0)

- Generic Control Record Type Definition (NFCForum-TS-GenericControl_RTD_1.0) http://www.nfc-forum.org/specs/spec_license
- URI Record Type Definition (NFCForum-TS-RTD_URI_1) http://www.nfc-forum.org/specs/spec_license
- Text Record Type Definition (NFCForum-TS-RTD_Text_1.0) http://www.nfc-forum.org/specs/spec_license
- Smart Poster Record Type Definition (NFCForum-SmartPoster_RTD_1.0) http://www.nfc-forum.org/specs/spec_license
- Signature Record Type Definition (NFCForum-TS-Signature_RTD-1.0) http://www.nfc-forum.org/specs/spec_license
- Type 2 Tag Operation Specification (NFCForum-TS-Type-2-Tag_1.1) http://www.nfc-forum.org/specs/spec_license
- *Tag-it™ HF-I Plus Transponder Inlays Reference Guide* (SCBU004)
- *Tag-it™ HF-I Transponder IC TMS37112 Reference Guide* (SCBU011)
- *Tag-it ™ HF-I Plus Transponder Chip/Inlays Extended Commands and Options Reference Guide* (SCBU003)
- *Tag-it™ HF-I Pro Transponder Inlays Reference Guide* (SCBU009)
- *Tag-it HF-I Pro Transponder IC Reference Guide* (SCBU045)
- *Tag-it™ HF-I Standard Transponder Inlays Reference Guide* (SCBU006)
- *Tag-it HF-I Standard Transponder IC Reference Guide* (SCBU047)

## Revision History

**Changes from Original (January 2012) to A Revision** **Page**



NOTE: Page numbers for previous revisions may differ from page numbers in the current version.

## IMPORTANT NOTICE



| **Products** | | **Applications** | |
|---|---|---|---|
| Audio | www.ti.com/audio | Automotive and Transportation | www.ti.com/automotive |
| Amplifiers | amplifier.ti.com | Communications and Telecom | www.ti.com/communications |
| Data Converters | dataconverter.ti.com | Computers and Peripherals | www.ti.com/computers |
| DLP® Products | www.dlp.com | Consumer Electronics | www.ti.com/consumer-apps |
| DSP | dsp.ti.com | Energy and Lighting | www.ti.com/energy |
| Clocks and Timers | www.ti.com/clocks | Industrial | www.ti.com/industrial |
| Interface | interface.ti.com | Medical | www.ti.com/medical |
| Logic | logic.ti.com | Security | www.ti.com/security |
| Power Mgmt | power.ti.com | Space, Avionics and Defense | www.ti.com/space-avionics-defense |
| Microcontrollers | microcontroller.ti.com | Video and Imaging | www.ti.com/video |
| RFID | www.ti-rfid.com | | |
| OMAP Applications Processors | www.ti.com/omap | **TI E2E Community** | e2e.ti.com |
| Wireless Connectivity | www.ti.com/wirelessconnectivity | | |

Mailing Address: Texas Instruments, Post Office Box 655303, Dallas, Texas 75265