TEXAS INSTRUMENTS

# Reader Series 4000

## *S4100 Multi-Function Reader Module RF-MGR-MNMN*

# *Tag-it™ Library Reference Guide*

11-06-21-708 October 2003

## First Edition - October 2003

This is the first edition of this manual. It describes the **TI Series 4000 Reader.**

It contains a description of the following reader module:

S4100 Multi-Function Reader Module P/N: **RF-MGR-MNMN-N0**

*Texas Instruments (TI) reserves the right to make changes to its products or services or to discontinue any product or service at any time without notice. TI provides customer assistance in various technical areas, but does not have full access to data concerning the use and applications of customer's products*

*Therefore, TI assumes no liability and is not responsible for customer applications or product or software design or performance relating to systems or applications incorporating TI products. In addition, TI assumes no liability and is not responsible for infringement of patents and/or any other intellectual or industrial property rights of third parties, which may result from assistance provided by TI.*

*TI products are not designed, intended, authorized or warranted to be suitable for life support applications or any other life critical applications which could involve potential risk of death, personal injury or severe property or environmental damage.*

# Read This First

## About This Manual

This reference guide for the Series 4000 Multi-Function (13.56 MHz & 134.2 KHz) Reader is designed for use by TI customers who are engineers experienced with RFID Systems and Radio Frequency Identification Devices (RFID).

| Device Name | Boot Loader Firmware Version |
|---|---|
| RF-MGR-MNMN-N0 | 1.02 |

**The Regulatory, safety and warranty notices that must be followed are provided in Chapter 2.**

## Conventions

The following pictograms and designations are used in the operating instructions:

**WARNING:**

**A WARNING IS USED WHERE CARE MUST BE TAKEN, OR A CERTAIN PROCEDURE MUST BE FOLLOWED, IN ORDER TO PREVENT INJURY OR HARM TO YOUR HEALTH.**

**CAUTION:**

**This indicates information on conditions, which must be met, or a procedure, which must be followed, which if not needed could cause permanent damage to the system.**

**Note:**

Indicates conditions, which must be met, or procedures which must be followed, to ensure proper functioning.

**Information:**

Indicates conditions or procedures that should be followed to ensure proper functioning of the system.

## If You Need Assistance

Application Centers are located in Europe, North and South America, the Far East and Australia to provide direct engineering support.

For more information, please contact your nearest TIRIS Sales and Application Center. The contact addresses can be found on our home page: http://www.tirfid.com.

## Numerical Representations

Unless otherwise noted, numbers are represented as decimal.

Hexadecimal numbers are represented with the suffix $_{16}$, e.g. $A5F1_{16}$

Binary numbers are represented with the suffix $_2$, e.g. $1011_2$

Byte representations: the least significant bit (lsb) is bit 0 and the most significant bit (msb) is bit 7.

## Document Overview

Chapter 1

# Tag-it™ Library

**Topic** **Page**

## 1.1 Tag-It™ Library Overview

The following sections define and detail the Protocol functionality in the Tag-it™ Module of the MFR Base Application. This information includes Tag-it™ Protocol Commands and the data/parameters associated with them.

The MFR Module supports multiple entity libraries. Request Packets are received and parsed by the Application Layer. Some of the Request packets are directed to the Application Layer and others to a specific entity module. The <*Cmd1*> field of the packet is used to route the request to the correct library entity. The Entity ID (the <*Cmd1*> field) for the Tag-it™ Library is $05_{16}$.

In many of the Tag-it™ response packets, a "Tag-it™ Response Flags" field is present. This corresponds to four bits that are present in all Tag-it™ RF responses. These four "Response Flags" are right justified (i.e. shifted to the least-significant bits) by the Tag-it™ Library and placed in a single-byte field, often denoted as <*RespFlags*>. The bit flags must be checked by the host in order to properly decode a response packet, as its contents vary according to flag settings. The format of the Tag-it™ Response Flags field is shown in figure 1-1.

**Tag-it™ Response Flags**

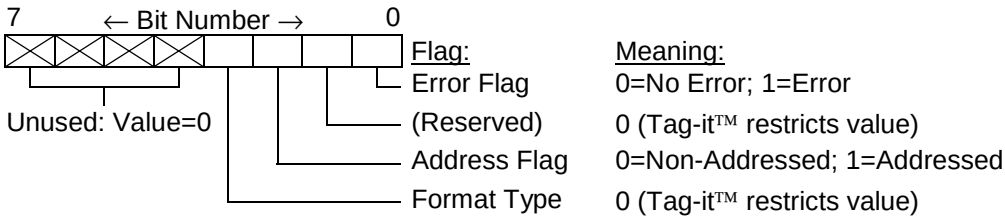


**Figure 1-1: Tag-it™ Response Flags**

All MFR Base Application response packets contain a <*Status*> field reflecting the success or failure of the request. The standard error codes for this field are found in the Base Application Protocol Rference Guide

The MFR Module allows the host to pass data directly to the Entity Modules. The Application Layer simply parses the Request Packet and passes the information directly to the selected library. The information in the Data fields is RF Format specific. The Application Layer doesn't verify if the data is valid, just that it has the right length and correct BCC (LRC). The Tag-it™ Library entity is responsible for validation of the data information in the Data Layer of the request packet. This document details each Protocol Command that provides access to the Tag-it™ Library functions. The following sections address each of these Protocol Commands and provide example request and response packets.

## Find Token Request ($41_{16}$)

In executing the Find Token Request command, the host application can send the MFR Base Application a Request Packet to check if a token is present. This packet contains a loop count that sets the maximum number of times the MFR Module Base Application searches for that token. This function allows a great deal of flexibility. It is possible to search for a variety of transponders or a single type of token. If multiple transponders are selected, the function returns the first token that is read. The Application continues to loop as specified, unless it detects a token. The Application doesn't return every type of token, just the first one read.

After a successful read, the MFR Module Base Application responds with **ERROR_NONE** in the <*Status*> followed by token's RF Technology Type and token data. The RF poll stops once a valid token is found. If a valid token is not found within the number of loops selected, the MFR Module Base Application responds with an **ERROR_TOKEN_NOT_PRESENT** <*Status*> field. The RF Poll stops once the loops are complete.

The Tag-it™ token search is handled by the function Find_Token_TagIt(). Note that this function is not called from the Application Entity if Tag-it™ is not in the Priority table unless the request is directed specifically to the Tag-it™ library. It is not called if a different token is discovered prior to reaching the Tag-it™ format in the priority table.

**Information:**

- Find Token request is implemented internally as a multiple "SID Poll" and "Slot Marker" commands to arbitrate collisions.
- The Application Entity (<*Cmd1*>=01) limits the number of Tag-it™ transponders reported to 1, but if multiple transponders are detected, the entity responds with a <*Status*> of ERROR_COLLISION_DETECT.
- The Tag-it™ Library Entity (<*Cmd1*>=05) limits the number of Tag-it™ transponders reported to 16 and concatenates their responses; if more transponders are detected, the entity responds with a <*Status*> of ERROR_COLLISION_DETECT.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 41 | Find Token |
| LoopCount | 1 | 00 – FF | Number of attempts to find token (0 ⇒ Loop until next request) |

Response Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 41 | Find Token |
| Status | 1 | 00 – FF | Standard Error Codes |
| EntityId | 1 | 05 | Tag-it™ Library Entity ID |
| SID 1 | 0 / 4 | 00 – FF | Simultaneous ID (Serial No.) of Token |
| | | ≡ | |
| SID n[(1)] | 0 / 4 | 00 – FF | Simultaneous ID (Serial No.) of Token |

(1) $n \leq 16$

**Information:**

This command is also valid with Entity ID =$01_{16.}$ In this case, the Application Layer looks for Transponders in the order they are defined in the Priority Table and stops when the first Token is found. This may not be a Tag-it ™ Token. Where the Entity ID = $05_{16}$, only Tag-it™ Transponders will be looked for.

Find Token Example: The request packet specifies 10 loops.

Request Packet: (**01 09 00 03 05 41 0A 45 BA**)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 09 00 | Packet Length 9 bytes |
| DeviceID | 03 | Device is MFR Module |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 41 | Find Token Request |
| LoopCount | 0A | 10 loops maximum |
| BCC | 45 BA | LRC and ~LRC |

Response Packet: (**01 12 00 03 05 41 00 05 01 0A 55 69 01 0A 5E 55 66 99**)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 12 00 | Packet Length 18 bytes |
| DeviceID | 03 | Device is MFR Module |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 41 | Find Token Request |
| Status | 00 | ERROR_NONE |
| EntityID | 05 | Tag-it™ Library Entity ID |
| SID 1 | 01 0A 55 69 | Simultaneous ID 1 (MSB 1st) |
| SID 2 | 01 0A 5E 55 | Simultaneous ID 2 (MSB 1st) |
| BCC | 66 99 | LRC and ~LRC |

## Pass-Through Request ($45_{16}$)

The Pass-Though Request Packet provides a way for a host to have direct communications to an Entity Module library. The Pass-Through Request packet specifies the Entity to direct the data to and then provides data that the library can send directly to its token. This function provides a way to access the low level RF protocol directly, bypassing any abstraction provided by the library and Application Layer.

The Tag-it™ library entity allows direct access to the RF interface via the Pass-Through request. When the Tag-it™ entity receives a Pass-Through request, the module configures the HF ASIC to the Tag-it™ specific RF scheme. The data in the Pass-Through request is not interpreted and it has meaning only to the token(s) that may receive it. Therefore the process that initiated the request must know the structure and data content of the transponder protocol.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 45 | Pass Through |
| NumBits | 2 | 0000 – FFFF | Number of bits in next field (LSB 1st) |
| Data | 0 – n(1) | 00 – FF | Data to send via RF channel |

(1) Length 'n' = Number of bytes to contain <*NumBits*> number of data bits. If the number of bits received is not a multiple of 8, the last byte in the <*Data*> field is padded with trailing zeros (i.e. valid data is in most-significant bits).

Response Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 45 | Pass Through |
| Status | 1 | 00 – FF | Standard Error Codes |
| NumBits | 2 | 0000 – FFFF | Number of bits in next field (LSB 1st) |
| Data | 0 – n(1) | 00 – FF | Tag-it™ Reply data bits (if received) |

(1) Length 'n' = Number of bytes to contain <*NumBits*> number of data bits. If the number of bits received is not a multiple of 8, the last byte in the <*Data*> field is padded with trailing zeros (i.e. valid data is in most-significant bits).

Tag-it™ is a bit oriented protocol. Therefore, unlike library layer commands, the tester has to create the Request Packets by packing bits into bytes and inserting them into the data field of MFR protocol. . Both the request and the response contain a command code, which specifies the operation to be (or that was) performed by the transponder. These transponder protocol details are specified in the Tag-it™ Transponder Protocol Reference Manual. This document is under the reference section on the RFID home page (http://www.ti.com/tiris).

**Information:**

The *Tag-it ™* CRC value is 16-bits. This is derived from a bit-wise computation process, as opposed the byte-wise method, used to compute all other CRC values referenced in this document.

.

The following table explains how to decode a standard Tag-it™ request protocol packet:

| Transponder Request Protocol Frame | | | |
|---|---|---|---|
| **Field Name** | **Bits** | **Definition** | |
| Request Code | 2 | $00_2$ = This is a request from the Reader to the Transponder | |
| Command Code | 8 | The Reader indicates which command the Transponder should perform. For parameter and data details, refer to Transponder Commands. | |
| | | $01_{16}$ | Get_Block: Request that the Transponder send the data contained in the specified block. |
| | | $03_{16}$ | Get_Version: Request for the Transponder to send version of its IC. |
| | | $05_{16}$ | Put_Block: Request for the Transponder to program data into the specified block in its memory. |
| | | $07_{16}$ | Put_Block_Lock: This is a combination of the Put_Block and the Lock_Block Command. |
| | | $08_{16}$ | Lock_Block: Request for the Transponder to permanently user-lock the specified block. |
| | | $0A_{16}$ | SID_Poll: Request for all Transponders in the Reader's antenna field to send back their SID address and chip characteristics. |
| | | $0B_{16}$ | Quiet: Request for a specific Transponder, as specified by the Address field, to "stay quiet" until a request other than an SID_POLL is received. |
| Format Type | 1 | Indicates the format of the request. Only fixed format is supported at this time. | |
| | | $0_2$ | = value must be set to zero for fixed format. |
| Address Flag | 1 | Indicates whether or not the Request is addressed to a specific Transponder. | |
| | | $0_2$ | = Non-Addressed |
| | | $1_2$ | = Addressed |
| Reserved | 1 | $0_2$ = value must be set to zero | |
| SID Address | 32 | Address of the Transponder to which the Request is sent. This field should only be present if the Address Flag field is set to "1". | |
| Parameters | 256 max | Request dependent. For the parameter and data details, refer to the Transponder Commands Descriptions. | |
| CRC | 16 | The calculated value of the CCITT CRC. | |

The following table explains is how to decode a standard Tag-it™ response protocol packet:

| Transponder Response Protocol Frame | | | |
|---|---|---|---|
| **Field Name** | **Bits** | **Definition** | |
| Response Code | 2 | $11_{16}$ = This is a Response from the Transponder to the Reader. | |
| Command Code | 8 | The Transponder responds with the Command Code it performed. Corresponding data is found in the parameter and data field. For the data layout, refer to Transponder Commands. | |
| | | $01_{16}$ | Get_Block: the Transponder responds with data from the specified block. |
| | | $03_{16}$ | Get_Version: the Transponder responds with the version information of its IC. |
| | | $05_{16}$ | Put_Block: the Transponder responds after it stored program data into the specified block in its memory. |
| | | $07_{16}$ | Put_Block_Lock: this is a combined Put_Block and Lock_Block Command. |
| | | $08_{16}$ | Lock_Block: the Transponder responds after it permanently sets the user-lock on the specified block |
| | | $0A_{16}$ | SID_Poll: the individual SIDs and IC characteristics from all the Transponders in the Reader's antenna field. |
| | | $0B_{16}$ | Quiet: the Transponder does not Respond until it receives an addressed Request. |
| Format Type | 1 | The Transponder indicates the format of the Request. Only fixed format is supported at this time. | |
| | | $0_2$ | = value must be set to zero for fixed format |
| Address Flag | 1 | The Transponder sets this flag to indicate whether the Address field will contain the address of the Transponder sending the Response. | |
| | | $0_2$ | = Non-Addressed |
| | | $1_2$ | = Addressed |
| Reserved | 1 | $0_2$ | = Value must be set to zero |
| Error Flag | 1 | The Transponder indicates whether the Request was successful. | |
| | | $0_2$ | = Success. The Request was completed successfully. |
| | | $1_2$ | = An error occurred. The error code is present in the Error Code field. |
| SID address | 32 | Contains the address of the Transponder which is sending back the Response. This field will only be present if the Address Flag field is set to "1". This can be used to confirm the identity of the Transponder. | |
| Error Code | 8 | The Transponder returns the error code. This field will only be present if the Error Flag field is set to "1". | |
| Data | 256 max | Request dependent. For details, refer to Transponder Commands. | |
| CRC | 16 | The calculated value of the CCITT CRC. | |

Pass-Through Example: Send a valid Tag-it™ Get block (Block 2) request with the SID.
Request Packet: (**01 13 00 03 05 45 45 00 00 50 0A 22 AC C0 15 0E F0 EB 14)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 13 00 | Packet Length 19 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 45 | Pass through command |
| NumBits | 45 00 | Length of RF data (69 bits) |
| Data | $00_2$<br>$01_{16}$<br>$0_2$<br>$1_2$<br>$0_2$<br>$01_{16}$ $44_{16}$ $55_{16}$ $98_{16}$<br>$02_{16}$<br>$A1DE_{16}$<br>$000_2$ | Request Code<br>Command Code: Get Block<br>Fixed Format Flag<br>Addressed Mode Flag<br>Reserved Flag<br>Simultaneous ID (MSB 1st)<br>Block Number 2<br>CRC-CCITT (MSB 1st)<br>Pad unused bits with zeros |
| BCC | EB 14 | LRC and ~LRC |

Response Packet: **(01 18 00 03 05 45 00 68 00 C0 50 05 11 56 60 09 12 34 56 78 05 52 D6 29)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 18 00 | Packet Length 24 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 45 | Pass through command |
| Status | 00 | ERROR_NONE |
| NumBits | 68 00 | 104 bits of RF data returned |
| Data | $11_2$<br>$01_{16}$<br>$0_2$<br>$1_2$<br>$0_2$<br>$0_2$<br>$01_{16}$ $44_{16}$ $55_{16}$ $98_{16}$<br>$02_{16}$<br>$01_2$<br>$12_{16}$ $34_{16}$ $56_{16}$ $78_{16}$<br>$0552_{16}$ | Response Code<br>Command Code: Get Block<br>Fixed Format Flag<br>Addressed Mode Flag<br>Reserved Flag<br>Error Flag<br>Simultaneous ID (MSB 1st)<br>Block Number 2<br>Lock Status (User Locked)<br>Block Data (MSB 1st)<br>CRC-CCITT (MSB 1st) |
| BCC | D6 29 | LRC and ~LRC |

Pass-Through Put Block Example: write $AAAAAAAA_{16}$ to block 2.
Below is an example of the degree of flexibility (and thus responsibility) of using the Pass-Through command to perform a Tag-it™ Put Block Request. Since the write operation is a combination of command and programming burst, the command requires 3 to 4 steps to complete. Firstly, the RF transmitter must be manually turned on so that the actual pass-through execution will keep the transmitter on upon exit. Then, the actual command is sent (to which the token does not reply by design). This is followed by the Slot Marker / End-Of-Frame Request which provides the programming burst. Finally, the application may choose to turn off the transmitter or leave it activated. The first 3 steps are shown below:

Transmitter On Request Packet: (**01 08 00 03 05 48 47 B8**)
Transmitter On Response Packet: (**01 09 00 03 05 48 00 46 B9**)

Pass Through Request Packet: (**01 13 00 03 05 45 45 00 01 40 1D 55 55 55 57 2E 98 FC 03)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 13 00 | Packet Length 19 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 45 | Pass through command |
| NumBits | 45 00 | Length of RF data (69 bits) |
| Data | $00_2$<br>$05_{16}$<br>$0_2\ 0_2\ 0_2$<br>$03_{16}$<br>$AA_{16}\ AA_{16}\ AA_{16}\ AA_{16}$<br>$E5D3_{16}$<br>$000_2$ | Request Code<br>Command Code: Put Block<br>Tag-it™ Request Flags<br>Block Number 2<br>Data to put (MSB 1st)<br>CRC-CCITT (MSB 1st)<br>Pad unused bits with zeros |
| BCC | FC 03 | LRC and ~LRC |

Pass Through Response Packet: (**01 01 09 00 03 05 45 01 4A B5**)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 09 00 | Packet Length 9 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 45 | Pass through command |
| Status | 01 | ERROR_TOKEN_NOT_PRESENT |
| BCC | 4A B5 | LRC and ~LRC |

Slot Marker / End-Of-Frame Request Packet: **(01 09 00 03 05 67 01 68 97)**

Slot Marker / End-Of-Frame Response: **(01 0B 00 03 05 67 00 05 00 6E 91)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 0B 00 | Packet Length 11 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 67 | Pass through command |
| Status | 00 | ERROR_NONE |
| CmdCode | 05 | Command Code: Put Block |
| RespFlags | 00 | Tag-it™ Response Flags |
| BCC | 6E 91 | LRC and ~LRC |

## Transmitter On Request ($48_{16}$)

Transmitter On Request is used to turn the transmitter on, for a specific entity. The request packet specifies in the <*Cmd1*> field the intended entity. The application layer firmware redirects the request to the intended entity. The entity then sets up the appropriate hardware and software configuration needed to implement its specific details and then turns the transmitter on.

The Tag-it™ Transmitter On request firstly configures the hardware ports to their proper data directions (input/output), sets up pull-up resistors for input ports and sets up the

correct logic levels at the output ports. The transmitter itself is turned on and then the entity communicates with the HF ASIC to set up the Tag-it™ protocol related register settings in the HF ASIC.

The request packet has no data bytes. The <*Cmd1*> field indicates the entity intended to turn on the transmitter. The response status byte indicates the success/error status.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 48 | Transmitter On |

Response Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 1 | 48 | Transmitter On |
| Status | 1 | 00 – FF | Standard Error Codes |

Transmitter ON Example:
Request Packet: (**01 08 00 03 05 48 47 B8**)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 08 00 | Packet Length 8 bytes |
| DeviceID | 03 | Device is MFR Module |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 48 | Transmitter On Request |
| BCC | 47 B8 | LRC and ~LRC |

Response Packet: (**01 09 00 03 05 48 00 46 B9**)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 09 00 | Packet Length 8 bytes |
| DeviceID | 03 | Device is MFR Module |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd 2 | 48 | Transmitter On Request |
| Status | 00 | ERROR_NONE |
| BCC | 46 B9 | LRC and ~LRC |

## Transmitter Off Request ($49_{16}$)

Transmitter Off Request is used to turn the transmitter off, for a specific entity. The request packet specifies in the <*Cmd1*> field the intended entity. The application layer firmware redirects the request to the intended entity. The entity then sets up the appropriate hardware and software configuration needed to implement its specific details and then turns the transmitter off.

The Tag-it™ entity turns the transmitter off and returns a response. The request packet consists of no data bytes. The <*Cmd1*> field indicates that the entity intended to turn off the transmitter. The response status byte indicates the success/error status.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 49 | Transmitter Off |

Response Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 1 | 49 | Transmitter Off |
| Status | 1 | 00 – FF | Standard Error Codes |

Transmitter OFF Example:
Request Packet: (**01 08 00 03 05 49 46 B9**)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 08 00 | Packet Length 8 bytes |
| DeviceID | 03 | Device is MFR Module |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 49 | Transmitter Off Request |
| BCC | 46 B9 | LRC and ~LRC |

Response Packet: (**01 09 00 03 05 49 00 47 B8**)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 09 00 | Packet Length 9 bytes |
| DeviceID | 03 | Device is MFR Module |
| Cmd1 | 05 | Tag-it™ Library Entity ID |
| Cmd2 | 49 | Transmitter Off Request |
| Status | 00 | ERROR_NONE |
| BCC | 47 B8 | LRC and ~LRC |

## Get Block Request ($61_{16}$)

The Get Block Request gets the data from one memory block of the responding token. In addition to this data, a Block Security Status byte is returned. This byte indicates the write-protection status of the block specified (i.e. Unlocked, (User/Factory) Locked, etc.).

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 61 | Get Block |
| BlkNum | 1 | 00 – FF | Specifies block that is to be read |
| SID(1) | 0 / 4 | 00 – FF | Simultaneous ID (Serial No.) of token |

(1) If SID field is not present, all transponders in read zone are addressed

Response Packet:

RplyDat

| | Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|---|
| | Cmd1 | 1 | 05 | Tag-it™ Entity ID |
| | Cmd2 | 1 | 61 | Get Block |
| | Status | 1 | 00 – FF | Standard Error Codes |
| (1) | Cmd Code | 1 | 01 | Command Code: Get Block |
| | Resp Flags | 1 | 00 / 04 | Tag-it™ Response Flags |
| | SID[A] | 0 / 4 | 00 – FF | Simultaneous ID of token |
| | BlkNum | 1 | 00 – FF | Block Number |
| | Lock Status | 1 | 00 – 03 | Lock Status |
| | Blk Data | n[B] | 00 – FF | Block Data (msb 1st) |

[A] Present when 'Address Flag' bit is set in <*RespFlags*>
[B] The value of 'n' varies with the block size for the specific token read

— OR —

| | | | | |
|---|---|---|---|---|
| (2) | CmdCode | 1 | 01 | Command Code: Get Block |
| | RespFlags | 1 | 01 / 05 | Tag-it™ Response Flags |
| | SID[C] | 0 / 4 | 00 – FF | Simultaneous ID of token |
| | ErrorResp | 1 | 00 – FF | Error Code |

[C] Present when 'Address Flag' bit is set in <*RespFlags*>

— OR —

| | | | | |
|---|---|---|---|---|
| (3) | NoData | 0 | — | (See <*Status*> field) |

[1] Response when token responds with block data
[2] Response when 'Error Flag' is set in <*RespFlags*>
[3] No Data returned due to condition described in <*Status*> field

Get Block Example:

Request Packet: (**01 0D 00 03 05 61 01 00 AC 60 E5 43 BC)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 0D 00 | Packet Length 13 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 61 | Get Block |
| BlkNum | 01 | Block Number 1 |
| SID | 00 AC 60 E5 | Simultaneous ID of token |
| BCC | 43 BC | LRC and ~LRC |

Response Packet: **(01 15 00 03 05 61 00 01 04 00 AC 60 E5 01 00 12 34 56 78 56 A9)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 15 00 | Packet Length 21 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 61 | Get Block |
| Status | 00 | ERROR_NONE |
| CmdCode | 01 | Command Code: Get Block |
| RespFlags | 04 | Tag-it™ Response Flags |
| SID | 00 AC 60 E5 | Simultaneous ID of token |
| BlkNum | 01 | Block Number 1 |
| LockStatus | 00 | Block is unlocked |
| BlkData | 12 34 56 78 | Block Data (msb 1st) |
| BCC | 56 A9 | LRC and ~LRC |

## Get IC Version Request ($62_{16}$)

The Get IC Version Request acquires information on the properties of a responding token. These properties include IC version and manufacturer information as well as the number and size of memory blocks available.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 62 | Get IC Version |
| SID(1) | 0 / 4 | 00 – FF | Simultaneous ID (Serial No.) of token |

(1) If SID field is not present, all transponders in read zone are addressed

Response Packet:

| | Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|---|
| | Cmd1 | 1 | 05 | Tag-it™ Entity ID |
| | Cmd2 | 1 | 62 | Get IC Version |
| | Status | 1 | 00 – FF | Standard Error Codes |
| RplyDat | StdResp(1) | 11 | 03 | Byte 1: Command Code: Get IC Ver. |
| | | | 00 / 04 | 2: Tag-it™ Response Flags |
| | | | 00 – FF | 3-6: Simultaneous ID of token(2) |
| | | | 00 – 7F | 7: Manufacturer Code |
| | | | 00 – FF | 8-9: Chip Version |
| | | | 00 – 1F | 10: Block Bytes minus one |
| | | | 00 – FF | 11: Number of Blocks minus one |
| | — OR — | | | |
| | NoData(3) | 0 | - | *(See* <Status> *field)* |

(1) Response when token responds with no error
(2) SID is present regardless of the setting of the 'Address Flag' in <*RespFlags*>
(3) No Data returned due to condition described in <*Status*> field

Get IC Version Example:
Request Packet: (**01 08 00 03 05 62 6D 92)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 08 00 | Packet Length 8 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 62 | Get Block |
| BCC | 6D 92 | LRC and ~LRC |

Response Packet: **(01 14 00 03 05 62 00 03 00 01 0A 55 5D 01 00 05 03 07 71 8E)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 14 00 | Packet Length 20 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 62 | Get Block |
| Status | 00 | ERROR_NONE |
| CmdCode | 03 | Command Code: Get IC Ver. |
| RespFlags | 00 | Non-addressed mode |

| | | |
|---|---|---|
| SID | 01 0A 55 5D | Simultaneous ID of token |
| Manufacturer Code | 01 | Texas Instruments Inc. |
| IC Version | 00 05 | Version 5 |
| Block Bytes minus one | 03 | Four bytes per block |
| Number of Blocks minus one | 07 | Eight blocks |
| BCC | 71 8E | LRC and ~LRC |

## Put Block Request ($63_{16}$)

The Put Block Request writes data to one memory block of the addressed token(s). In order to successfully write data, the Host must know the size of the memory block of the token. This information is available through the 'Get IC Version' Request or 'SID Poll' sequence requesting version data. A corrupted response or lack of response does not necessarily indicate a failure to perform the write operation. Additionally, multiple transponders may process a non-addressed request.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 63 | Put Block |
| BlkNum | 1 | 00 – FF | Specifies block that is to be written |
| BlkBits | 1 | 00 – FF | Number of bits in next field minus one |
| BlkData | 1 – 32(1) | 00 – FF | Data to be written to specified block(2) |
| SID(3) | 0 / 4 | 00 – FF | Simultaneous ID (Serial No.) of token |

(1) Length specified by previous field
(2) Data is interpreted most-significant-bit 1st. If a byte contains < 8 valid bits, they are read from left-to-right (i.e. most-significant-bit 1st).
(3) If SID field is not present, all transponders in read zone are addressed

Response Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 1 | 63 | Put Block |
| Status | 1 | 00 – FF | Standard Error Codes |
| CmdCode | 0(1) / 1 | 05 | Command Code: Put Block |
| RespFlags | 0(1) / 1 | 00 – 05 | Tag-it™ Response Flags |
| SID(2) | 0 / 4 | 00 – FF | Simultaneous ID of token |
| ErrorResp(3) | 0 / 1 | 00 – FF | Error Code |

(1) No Data returned due to condition described in <*Status*> field
(2) Present when 'Address Flag' bit is set in <*RespFlags*>
(3) Present when 'Error Flag' is set in <*RespFlags*>

Put Block Example:
Request Packet: (**01 12 00 03 05 63 04 1F FF AA CC DD 01 0A 55 5D 2A D5)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 12 00 | Packet Length 18 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 63 | Put Block |
| BlkNum | 04 | Block Number 4 |
| BlkBits | 1F | 32 bits per block |
| BlkData | FF AA CC DD | Data to write (MSB 1st) |
| SID | 01 0A 55 5D | Simultaneous ID of token |
| BCC | 2A D5 | LRC and ~LRC |

Response Packet: **(01 0F 00 03 05 63 00 05 04 01 0A 55 5D 69 96)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 0F 00 | Packet Length 15 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 63 | Put Block |
| Status | 00 | ERROR_NONE |
| CmdCode | 05 | Command Code: Put Block |
| RespFlags | 04 | Addressed Mode, No Error |
| SID | 01 0A 55 5D | Simultaneous ID of token |
| BCC | 69 96 | LRC and ~LRC |

## Put Block Lock Request ($64_{16}$)

The Put Block Lock Request writes data to one memory block of the addressed token(s) and locks that block from further write operations. In order to successfully write data, the Host must know the size of the memory block of the token. This information is available through the 'Get IC Version' Request or 'SID Poll' sequence requesting version data. A corrupted response or lack of response does not necessarily indicate a failure to perform the write-lock operation. Additionally, multiple transponders may process a non-addressed request.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 64 | Put Block Lock |
| BlkNum | 1 | 00 – FF | Specifies block that is to be written |
| BlkBits | 1 | 00 – FF | Number of bits in next field minus one |
| BlkData | 1 – 32(1) | 00 – FF | Data to be written to specified block(2) |
| SID(3) | 0 / 4 | 00 – FF | Simultaneous ID (Serial No.) of token |

(1) Length specified by previous field
(2) Data is interpreted most-significant-bit 1st. If a byte contains < 8 valid bits, they are read from left-to-right (i.e. most-significant-bit 1st).
(3) If SID field is not present, all transponders in read zone are addressed

Response Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 1 | 64 | Put Block Lock |
| Status | 1 | 00 – FF | Standard Error Codes |
| CmdCode | 0(1) / 1 | 07 | Command Code: Put Block Lock |
| RespFlags | 0(1) / 1 | 00 – 05 | Tag-it™ Response Flags |
| SID(2) | 0 / 4 | 00 – FF | Simultaneous ID of token |
| ErrorResp(3) | 0 / 1 | 00 – FF | Error Code |

(1) No Data returned due to condition described in <*Status*> field
(2) Present when 'Address Flag' bit is set in <*RespFlags*>
(3) Present when 'Error Flag' is set in <*RespFlags*>

Put Lock Block Example:
Request Packet: (**01 12 00 03 05 64 08 1F AA AA CC DD 01 44 55 98 FF 00)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 12 00 | Packet Length 18 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 64 | Put Block Lock |
| BlkNum | 08 | Block # 8 (Out of Range) |
| BlkBits | 1F | 32 bits per block |
| BlkData | AA AA CC DD | Data to write (MSB 1st) |
| SID | 01 44 55 98 | Simultaneous ID of token |
| BCC | FF 00 | LRC and ~LRC |

If the block is already locked, a response similar to the one below results:

Response Packet: **(01 10 00 03 05 64 00 07 05 01 44 55 98 10 E9 16)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 10 00 | Packet Length 16 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 64 | Put Block Lock |
| Status | 00 | ERROR_NONE |
| CmdCode | 07 | Command Code: Put Block Lock |
| RespFlags | 05 | Addressed Mode, Error Flag |
| SID | 01 44 55 98 | Simultaneous ID of token |
| ErrorResp | 10 | Block Not Available |
| BCC | E9 16 | LRC and ~LRC |

## Lock Block Request ($65_{16}$)

The Lock Block Request write-protects one memory block of the addressed token(s). A corrupted response or lack of response does not necessarily indicate a failure to perform the lock operation. Additionally, multiple transponders may process a non-addressed request.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 65 | Lock Block |
| BlkNum | 1 | 00 – FF | Specifies block that is to be locked |
| SID(1) | 0 / 4 | 00 – FF | Simultaneous ID (Serial No.) of token |

(1) If SID field is not present, all transponders in read zone are addressed

Response Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 1 | 65 | Lock Block |
| Status | 1 | 00 – FF | Standard Error Codes |
| CmdCode | 0[1] / 1 | 08 | Command Code: Lock Block |
| RespFlags | 0[1] / 1 | 00 – 05 | Tag-it™ Response Flags |
| SID[2] | 0 / 4 | 00 – FF | Simultaneous ID of token |
| ErrorResp[3] | 0 / 1 | 00 – FF | Error Code |

(1) No Data returned due to condition described in <*Status*> field
(2) Present when 'Address Flag' bit is set in <*RespFlags*>
(3) Present when 'Error Flag' is set in <*RespFlags*>

Lock Block Example:
Request Packet: (**01 09 00 03 05 65 01 6A 95)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 09 00 | Packet Length 9 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 65 | Lock Block |
| BlkNum | 01 | Block Number 1 |
| BCC | 6A 95 | LRC and ~LRC |

Response Packet: **(01 0B 00 03 05 65 00 08 00 61 9E)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 0B 00 | Packet Length 11 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 65 | Lock Block |
| Status | 00 | ERROR_NONE |
| CmdCode | 08 | Command Code: Lock Block |
| RespFlags | 00 | Non-Addressed, No Error |
| BCC | 61 9E | LRC and ~LRC |

## SID Poll Request ($66_{16}$)

The SID Poll Request is used to acquire the Simultaneous ID's of Tag-it™ transponders. This request decreases the likelihood of a data collision by forcing transponders to respond in 1 of 16 slots based on a portion of their SID's. To perform a slotted sequence, the 'Slot Marker / End-Of-Frame' Request is used in conjunction with this request. Any collision that does occur can be further arbitrated using the Anti-collision mask in an algorithm outlined in the Tag-it™ Transponder Protocol Reference Manual.

**Information:**
Tag-it™ transmitter remains **ON** in order to preserve the token states changed by the request.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 66 | SID Poll |
| ReqVersion | 1 | 00 – FF | 0 : Request only SID in reply<br>(other) : Request version data also |
| MskLen | 1 | 00 – 20 | Number of bits in next field |
| MskVal(1) | 0 – 4 | 00 – FF | Anti-collision mask per Tag-it™ spec. |

(1) Data is interpreted most-significant-bit 1st. If a byte contains < 8 valid bits, they are read from left-to-right (i.e. most-significant-bit 1st).

Response Packet:

| | Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|---|
| | Cmd1 | 1 | 05 | Tag-it™ Entity ID |
| | Cmd2 | 1 | 66 | SID Poll |
| | Status | 1 | 00 – FF | Standard Error Codes |
| RplyDat (1) | CmdCode | 1 | 0A | Command Code: SID Poll |
| | RespFlags | 1 | 00 – 0F | Tag-it™ Response Flags |
| | SID | 4 | 00 – FF | Simultaneous ID of token |
| | VersionDat | 0 / 5(A) | 00 – 7F | Byte 1: Manufacturer Code |
| | | | 00 – FF | 2-3: Chip Version |
| | | | 00 – 1F | 4: Block Bytes minus one |
| | | | 00 – FF | 5: Number of Blocks minus one |
| | — OR — | | | |
| RplyDat (2) | CmdCode | 1 | 0A | Command Code: SID Poll |
| | RespFlags | 1 | 00 – 01 | Tag-it™ Response Flags |
| | ErrorResp | 1 | 00 – FF | Error Code |
| | — OR — | | | |
| RplyDat (3) | NoData | 0 | — | (See <*Status*> field) |

(A) <*VersionDat*> present when the <*ReqVersion*> field in corresponding 'SID Poll' Request is non-zero

(1) Response when token responds with SID
(2) Response when token responds with error
(3) No Data returned due to condition described in <*Status*> field

SID Poll Example:

Request Packet: **(01 0E 00 03 05 66 00 1D 00 00 00 00 72 8D)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 0E 00 | Packet Length 14 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 66 | SID Poll |
| ReqVersion | 00 | Do not request version info. |
| MskLen | 1D | 29 bit mask size (invalid) |
| MskVal | 00 00 00 00 | Mask bits |
| BCC | 72 8D | LRC and ~LRC |

Response Packet: **(01 0C 00 03 05 66 00 0A 01 1F 79 86)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| Packet Length | 0C 00 | Packet Length 12 bytes |
| Device ID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 66 | SID Poll |
| Status | 00 | ERROR_NONE |
| CmdCode | 0A | Command Code: SID Poll |
| RespFlags | 01 | Non-Addressed, Error Flag |
| ErrorResp | 1F | Command not allowed |
| BCC | 79 86 | LRC and ~LRC |

## Slot Marker / End-Of-Frame Request ($67_{16}$)

The Slot Marker / End-Of-Frame Request is used in conjunction with other requests as part of an exchange sequence. Its purpose is to send the Tag-it™ End-Of-Frame marker to transponders within read range and await a response. In the SID sequence, the EOF is interpreted as the "Slot Marker" for anti-collision. The EOF marker is also used in conjunction with pass-thru write and lock operations in order to complete that operation and poll the token for a response.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 05 | Entity ID |
| Cmd2 | 1 | 67 | Slot Marker / End-Of-Frame |
| FmtReply | 1 | 00 – FF | 0 : Request no formatting<br>(other) : Align fields & perform CRC |

Response Packet:

RplyDat

| | Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|---|
| | Cmd1 | 1 | 05 | Tag-it™ Entity ID |
| | Cmd2 | 1 | 67 | Slot Marker / End-Of-Frame |
| | Status | 1 | 00 – FF | Standard Error Codes |
| (1) | CmdCode | 1 | 0A | Command Code: SID Poll |
| | RespFlags | 1 | 00 – 0F | Tag-it™ Response Flags |
| | SID | 4 | 00 – FF | Simultaneous ID of token |
| | VersionDat | 0 / 5(A) | 00 – 7F | Byte 1: Manufacturer Code |
| | | | 00 – FF | 2-3: Chip Version |
| | | | 00 – 1F | 4: Block Bytes minus one |
| | | | 00 – FF | 5: Number of Blocks minus one |
| | — OR — | | | |
| (2) | CmdCode | 1 | 05 – 08 | Command Code |
| | RespFlags | 1 | 00 – 05 | Tag-it™ Response Flags |
| | SID(B) | 0 / 4 | 00 – FF | Simultaneous ID of token |
| | ErrorResp(C) | 0 / 1 | 00 – FF | Error Code |
| | — OR — | | | |
| (3) | RawData | 0 - X | 00 – FF | Raw Tag-it™ protocol bits |
| | — OR — | | | |
| (4) | NoData | 0 | — | (See <*Status*> field) |

(A) <*VersionDat*> present when the <*ReqVersion*> field in corresponding 'SID Poll' Request is non-zero

(B) Present when 'Address Flag' bit is set in <*RespFlags*>

(C) Present when 'Error Flag' is set in <*RespFlags*>

(1) Response when Slot Marker/EOF used with 'SID Poll'

(2) Response when Slot Marker/EOF used with pass-thru Write/Lock Operation
(3) Unformatted Tag-it™ bits when <*FmtReply*> = 0 (CRC performed by host)
(4) No Data returned due to condition described in <*Status*> field

Slot Marker/ EOF Example:

Request Packet: **(01 0A 00 03 05 66 00 00 6B 94)** (SID Poll: Slot #00)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 0A 00 | Packet Length 10 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 66 | SID Poll |
| ReqVersion | 00 | Do not request version info. |
| MskLen | 00 | 0 bits in mask (∴ no MskVal field) |
| BCC | 6B 94 | LRC and ~LRC |

Response Packet: **(01 09 00 03 05 66 01 69 96)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| Packet Length | 09 00 | Packet Length 9 bytes |
| Device ID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 66 | SID Poll |
| Status | 01 | ERROR_TOKEN_NOT_PRESENT |
| BCC | 69 96 | LRC and ~LRC |

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #01)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 09 00 | Packet Length 9 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 67 | Slot Marker / EOF |
| FmtReply | 01 | Align fields & perform CRC |
| BCC | 68 97 | LRC and ~LRC |

Response Packet: **(01 09 00 03 05 67 01 68 97)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| Packet Length | 09 00 | Packet Length 9 bytes |
| Device ID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 67 | Slot Marker / EOF |
| Status | 01 | ERROR_TOKEN_NOT_PRESENT |
| BCC | 68 97 | LRC and ~LRC |

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #02)

Response Packet: **(01 09 00 03 05 67 01 68 97)**

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #03)

Response Packet: **(01 09 00 03 05 67 01 68 97)**

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #04)

Response Packet: **(01 09 00 03 05 67 01 68 97)**

≡

Response Packet: **(01 09 00 03 05 67 01 68 97)**

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #09)

Response Packet: **(01 0F 00 03 05 67 00 0A 00 01 0A 55 69 52 AD)**

| **Field** | **Contents** | **Summary** |
|---|---|---|
| SOF | 01 | Start of Frame |
| Packet Length | 0F 00 | Packet Length 15 bytes |
| Device ID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 67 | Slot Marker / EOF |
| Status | 00 | ERROR_NONE |
| CmdCode | 0A | Command Code: SID Poll |
| RespFlags | 00 | Non-Addressed, No Error |
| SID | 01 0A 55 69 | Simultaneous ID of token |
| BCC | 52 AD | LRC and ~LRC |

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #0A)

Response Packet: **(01 09 00 03 05 67 01 68 97)**

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #0B)

Response Packet: **(01 09 00 03 05 67 01 68 97)**

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #0C)

Response Packet: **(01 09 00 03 05 67 01 68 97)**

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #0D)

Response Packet: **(01 09 00 03 05 67 01 68 97)**

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #0E)

Response Packet: **(01 0F 00 03 05 67 00 0A 00 01 0A 55 5E 65 9A)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| Packet Length | 0F 00 | Packet Length 15 bytes |
| Device ID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 67 | Slot Marker / EOF |
| Status | 00 | ERROR_NONE |
| CmdCode | 0A | Command Code: SID Poll |
| RespFlags | 00 | Non-Addressed, No Error |
| SID | 01 0A 55 5E | Simultaneous ID of token |
| BCC | 65 9A | LRC and ~LRC |

Request Packet: **(01 09 00 03 05 67 01 68 97)** (Slot Marker: Slot #0F)

Response Packet: **(01 09 00 03 05 67 01 68 97)**

— END 16-SLOT SID POLL SEQUENCE —

## Quiet Request ($68_{16}$)

The Quiet Request is used to silence a token, preventing it from responding to any non-addressed or 'SID Poll' related requests. The token will, however, respond to requests with matching SID. As there is no response to this request from the receiving token, only request status and errors are reported.

**Information:** The Tag-it™ transmitter remains **ON** in order to preserve the token states changed by this request.

Request Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 04 | Entity ID |
| Cmd2 | 1 | 68 | Quiet |
| SID(1) | 0 / 4 | 00 – FF | Simultaneous ID of token |

(1) If SID field is not present, all transponders in read zone are silenced

Response Packet:

| Field | Length (Bytes) | Value Range (Hex) | Description |
|---|---|---|---|
| Cmd1 | 1 | 04 | Tag-it™ Entity ID |
| Cmd2 | 1 | 68 | Quiet |
| Status | 1 | 00 – FF | Standard Error Codes |

Quiet Example:
Request Packet: **(01 0C 00 03 05 68 00 AC 60 E5 4A B5)**

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 0C 00 | Packet Length 12 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 68 | Quiet |
| SID | 00 AC 60 E5 | Simultaneous ID of token |
| BCC | 4A B5 | LRC and ~LRC |

Response Packet: (**01 09 00 03 05 68 00 66 99**)

| Field | Contents | Summary |
|---|---|---|
| SOF | 01 | Start of Frame |
| PacketLen | 09 00 | Packet Length 9 bytes |
| DeviceID | 03 | Device is MFR |
| Cmd1 | 05 | Tag-it™ Entity ID |
| Cmd2 | 68 | Quiet |
| Status | 00 | ERROR_NONE |
| BCC | 66 99 | LRC and ~LRC |

Chapter 2

# Regulatory and Warranty Notices

**Topic** **Page**

## 2.1 FCC Conformity

The Series 4000 Multi-Function Reader is an intentional radiator. The transmitter portion operates at 13.56 MHz and is subject to FCC Part 15, Subpart C, “Intentional Radiator,” paragraph 15.225 (13.553-13.567MHz). Radiated emissions from the device are subject to the limits in Section 15.209 of the Rules outside of the 13.56 +/- 0.007 MHz band.

**Note:**

Any device or system incorporating the Series 4000 reader, in full or in part, needs to obtain FCC certification as part of the system within which this reader unit resides. A system containing this product may be operated only under an experimental license or final approval issued by the relevant approval authority. Before any such device or system can be marketed, an equipment authorization must be obtained form the relevant approval authority.

## 2.2 ETSI Conformity

Any device or system incorporating the Series 4000 reader, in full or in part, may need to comply with European Standard EN300330. It is the responsibility of each system integrator to have their complete system tested and to obtain approvals as required from the local authorities before operating or selling this system.

## 2.3 CE Conformity

Any device or system incorporating the Series 4000 reader, in full or in part, may need to have a CE Declaration of Conformity stating that it meets European EMC directive 99/5/EC. This must be issued by the system integrator or user of such a system prior to marketing or operating it in the European community.

## 2.4 Warranty and Liability

The "General Conditions of Sale and Delivery" of Texas Instruments Incorporated or a TI subsidiary apply. Warranty and liability claims for defect products, injuries to persons and property damages are void if they are the result of one or more of the following causes:

- Improper use of the reader module.
- Unauthorized assembly, operation and maintenance of the reader module.
- Operation of the reader modules with defective and/or non-functioning safety and protective equipment.
- Failure to observe the instructions during transport, storage, assembly, operation, maintenance and setting up of the reader modules.
- Unauthorized changes to the reader modules.
- Insufficient monitoring of the reader modules' operation or environmental conditions.
- Improperly conducted repairs.
- Catastrophes caused by foreign bodies and acts of God.