CSL TCP/IP Network Interface (Ethernet and Wi Fi) Specification for Fixed Intelligent Reader (R2000 Based) Family:

CS463/CS203X/CS468XJ

Version 1.9

November 21, 2023

CSL: The One-Stop-Shop for RFID Solutions

Release Notes

Date	Ver	Author	Remark
2020 05 04	1.0	Albert Lai	Convert to new CS463/CS203X/CS468X/CS468XJ/CS206 only
2020 07 16	1.1	Albert Lai	Add inventory cycle begin command state packet
2020 09 07	1.2	Albert Lai	Delete old unsupported commands
2020 09 11	1.3	Albert Lai	Update Appendix A RFID Commands
2020 09 18	1.4	Albert Lai	Clean up wordings
2020 09 24	1.5	Albert Lai	Add Antenna Cycle Begin packet and MAC Error 0x0336
2020 11 26	1.6	Albert Lai	Add FM13DT160 HST commands and MAC registers
2021 08 10	1.7	Albert Lai	Add GPI Interrupt 00 and 01 (Chapter 4: UDP Based Control Commands)
2021 09 15	1.8	Albert Lai	Add reminder on packet version number in uplink inventory packet
2023 11 21	1.9	Carlson Lam	Remove the naming convention on “Unified API” to avoid confusions

1 Content

2 Introduction

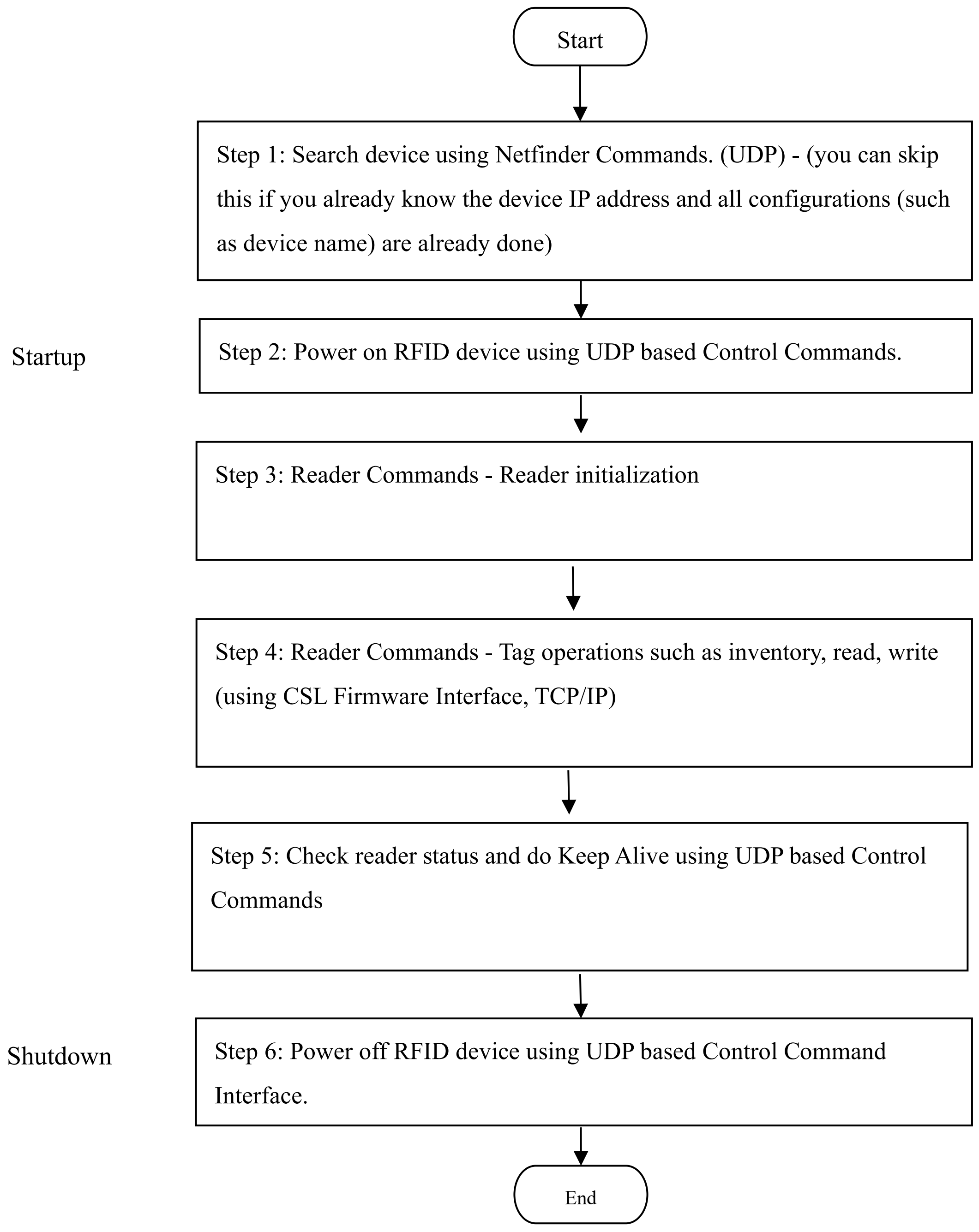
2.1 Fixed Readers

The CSL fixed readers consist of the new intelligent reader series.

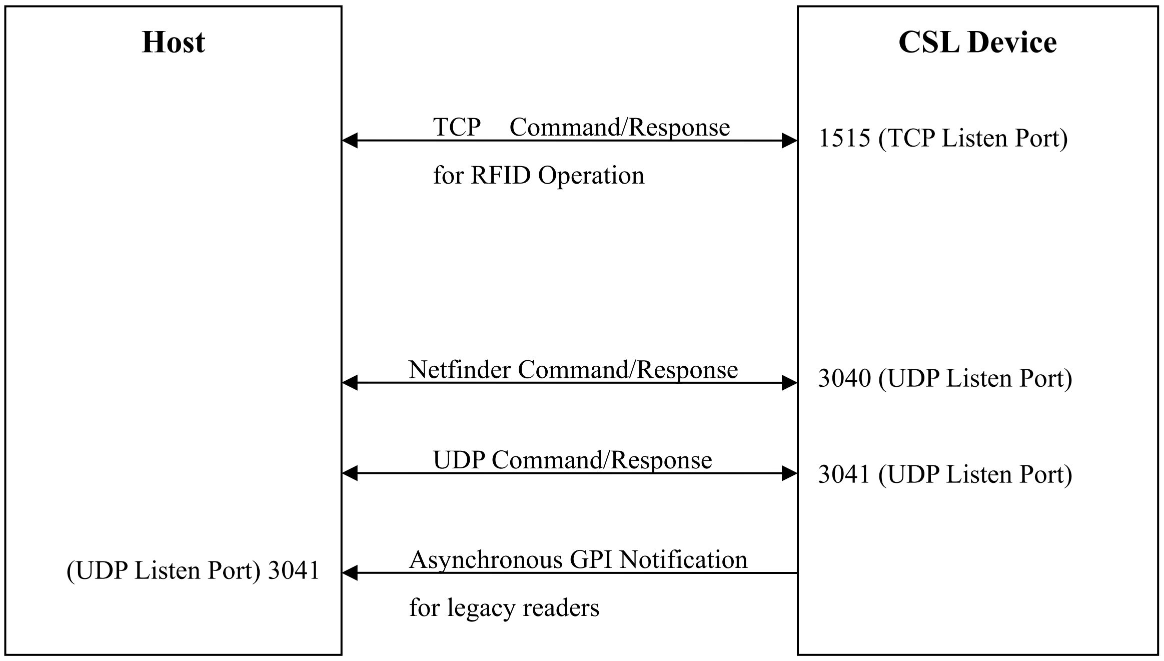
New Intelligent Reader Series: CS463, CS468XJ, CS468X, CS203X, CS206;

2.2 Command Sequences Overview

Before performing ISO 18000-6C tag operations via the CSL Network Low Level Firmware Interface, there will be a sequence of operation as the following 5 steps:


Start
Step 1: Search device using Netfinder Commands. (UDP) - (you can skip this if you already know the device IP address and all configurations (such as device name) are already done)
Startup
Step 2: Power on RFID device using UDP based Control Commands.
Step 3: Reader Commands - Reader initialization
Step 4: Reader Commands - Tag operations such as inventory, read, write (using CSL Firmware Interface, TCP/IP)
Step 5: Check reader status and do Keep Alive using UDP based Control Commands
Shutdown
Step 6: Power off RFID device using UDP based Control Command Interface.
End

The network port connection diagram between the CSL reader and the control server is as below. The TCP ports 1515 is used for data communications. The UDP ports 3040 and 3041 on CSL device are used for Netfinder command and UDP control command communication. For legacy readers, if the GPI interrupt setting is enabled, an asynchronous GPI notification message will be sent to the host at UDP port 3041 after GPI port is triggered.



This document describes the packet format used by CSL Network Firmware Interface to present data via the external interface. Note that there are 3 types of commands defined here: Netfinder Commands, Control Commands and Reader Commands. Chapter 2, 3 and 4 describes each type respectively.

2.3 Minimum Software Requirements

For all the updated functions, it is recommended the following firmware versions are in the reader:

Legacy Readers:

Firmware version:	1.5.35 or higher
8051 Bootloader version:	2.9.5 or higher
8051 App version:	2.18.59 or higher

2.4 Related Documents

1. CSL Firmware Command Specification (Appendix A)
2. EPC Radio Frequency Identity Protocols Class-1 Gen-2 UHF RFID Protocol for Communications at 860 MHz – 960 MHz Version 1.2.0

3 Netfinder Commands

Netfinder packets are UDP packets that provide the following functions:

- Search CSL device on the network

These packets are all transmit using User Datagram Protocol (UDP) with port 3040.

Notes:

1. **You can skip the Netfinder commands step if the CSL device IP address is already known and fixed, and you do not need to change any configurations such as device name.**

2. **Netfinder commands are available only when TCP port 1515 is NOT connected.**

3.1 Search Packet Format

The search packet is recommended to be transmitted by UDP broadcast with port 3040 so that all CSL readers under the same network can receive the search packet (on the reader side, since they are multi-protocol readers, their access mode need to be set to CSL Unified API before they can be "seen" by netfinder commands). Table 1 explains the UDP broadcast.

Table 1: Search Packet Fields

Byte Offset(s)	**Name**	**Description**
0	pkt_type	Packet specific type Currently = 0x00
1	reserved	Reserved. Set as zero.
3:2	rand_no	A random number value to prevent the generation of duplicated traffic. 0x0000 – 0x7FFF are valid. Must use a new random number for each new search.

When a CSL device has correctly received the search packet, a search reply packet will be sent out. Table 2 explains the reply packets.

Table 2: Search Reply Packet Fields

Byte Offset(s)	Name	Description
0	pkt_type	Packet specific type Currently = 0x01
1	mode	Operation mode. 0 = normal mode, 1 = bootloader mode.
3:2	rand_no	A random number value to prevent the generation of duplicated traffic. Must be the same as the search packet rand_no field. Otherwise ignore this packet.
5:4	day_on_powered	The day field from powered on time
6	hour_on_powered	The hour field from powered on time.
7	minute_on_powered	The minute field from powered on time.
9:8	day_on_network	The day field from network on time
10	hour_on_ network	The hour field from network on time.
11	minute_on_ network	The minute field from network on time.
12	second_on_powered	The second field from powered on time.
13	second_on_network	The second field from network on time.
19:14	mac	Reader's MAC address
23:20	ip	Reader's IP address
27:24	trusted_host_ip	Trusted host's IP address
28	trusted_host_mode	0 = normal mode (any host IP can connect TCP) 1 = trusted host mode (only trusted host IP can connect TCP)
29	connection_mode	1 = USB mode. 2 = Ethernet mode. 3 = RS232 mode. 4 = DBUART mode.
30	dhcp_retry	DHCP retry count.
31	ip_mode	0 = DHCP. 1 = static IP.
35:32	subnet	Subnet mask
39:36	gateway	Gateway address
n:40	name	1st null terminated string is the device name. 2nd null terminated string is the description. 3rd null terminated string is the name of power on time.

		4th null terminated string is the name of network on time.
n+1	gateway_check	Gateway check reset mode. 1 = on. Otherwise = off (default).

4 Control Commands

UDP based Control Commands are UDP commands that provide custom defined control to CSL device. These commands should be sent out to CSL device in a broadcast manner. **These packets are all transmitted using User Datagram Protocol (UDP) with port 3041.**

4.1 UDP Based Control Commands Packet Format

UDP Port is 3041. All UDP based Control Commands are recommended to be sent out in a broadcast manner. The UDP commands actually can be sent in either broadcast or unicast mode. As broadcast is more robust in some network situation, it is suggested to use broadcast mode.

Table 3: UDP Based Control Commands packet field for UDP Port 3041

Byte Offset	Name	Description
0	header	= 0x80
4:1	destination	= destination's IP address (the CSL device you want to walk to)
5	length	= no. of value byte
6	command	0x01 = check status 0x05 = check GPI status for GPI0 and GPI1 0x06 = check GPO status for GPO0 and GPO1 0x07 = set GPO0 state 0x08 = set GPO1 state 0x09 = set GPI0 interrupt mode 0x0A = set GPI1 interrupt mode 0x0B = check GPI interrupt mode 0x0E = check API mode 0x15 = get bootloader version 0x16 = get image version 0x17 = set TCP notification mode (once set, if there is no tag data, it will send a packet of 8 byte of 0x98, i.e. 16 hex numbers: 0x9898989898989898 to the host every 0.5 second) 0x18 = check GPI status for GPI2 and GPI3 0x19 = check GPO status for GPO2 and GPO3 0x20 = set GPO2 state 0x21 = set GPO3 state
n:7	value	**Command code** **value**

		0x07	0 = off 1 = on
		0x08	0 = off 1 = on
		0x09	0 = off 1 = raising-edge trigger 2 = falling-edge trigger 3 = any trigger
		0x0A	0 = off 1 = raising-edge trigger 2 = falling-edge trigger 3 = any trigger
		0x17	0 = off 1 = on
		0x20	0 = off 1 = on
		0x21	0 = off 1 = on

Remark:

1. If GPI interrupt mode is turned on, asynchronous GPI notification return packet for check GPI status Command 0x05 command return will be sent out (to UDP port 3041) when trigger is detected.

Table 4: UDP Based Control Commands return packet field for Command 0x01

Byte Offset	Name	Description
0	header	= 0x81
1	flag	0x00 = failure 0x01 = success
2	length	= 10
3	command	= 0x01
4	power	RF board power status. 0 = off, 1 = on
5	error_reset	Error reset mode. 0 = off, 1 = on
6	udp_keepalive	UDP keepalive mode. 0 = off, 1 = on (this value is always off, i.e. = 0, now because UDP Keepalive is no longer supported)
7	connection	TCP connection status. 0 = TCP listening, non-zero = TCP connected
11:8	time	Elapsed time since last sent out low level interface packet. 1st byte = day 2nd byte = hour 3rd byte = minute 4th byte = second
12	inv_crc_filter	Inventory CRC error filter mode. 0 = off, 1 = on
13	tcp_notification	TCP notification mode. 0 = off, 1 = on

Table 5: UDP Based Control Commands return packet field for Command 0x07, 0x08, 0x17, 0x20, 0x21

Byte Offset	Name	Description
0	header	= 0x81
1	flag	0x00 = failure 0x01 = success
2	length	= 0
3	command	= input command code

Table 6: UDP Based Control Commands return packet field for Command 0x05

Byte Offset	Name	Description
0	header	= 0x81
1	flag	0x00 = failure 0x01 = success
2	length	= 2
3	command	= 0x05
4	GPI0	GPI0 state (0 or 1). **0xFF = invalid**
5	GPI1	GPI1 state (0 or 1). **0xFF = invalid**

Table 7: UDP Based Control Commands return packet field for Command 0x06

Byte Offset	Name	Description
0	header	= 0x81
1	flag	0x00 = failure 0x01 = success
2	length	= 2
3	command	= 0x06
4	GPO0	GPO0 state (0 or 1)
5	GPO1	GPO1 state (0 or 1)

Table 8: UDP Based Control Commands return packet field for Command 0x0B

Byte Offset	Name	Description
0	header	= 0x81
1	flag	0x00 = failure 0x01 = success
2	length	= 2
3	command	= 0x0B
4	GPI0 mode	GPI0 interrupt mode
5	GPI1 mode	GPI1 interrupt mode

Table 9: UDP Based Control Commands return packet field for Command 0x0E

Byte Offset	Name	Description
0	header	= 0x81
1	flag	0x00 = failure 0x01 = success
2	length	= 1
3	command	= 0x0E
4	API mode	0 = high level API mode 1 = low level API mode

Table10: UDP Based Control Commands return packet field for Command 0x15, 0x16

Byte Offset	Name	Description
0	header	= 0x81
1	flag	0x00 = failure 0x01 = success
2	length	= 4
3	command	= input command code
7:4	version	4 bytes version number

Table 11: UDP Based Control Commands return packet field for Command 0x18

Byte Offset	Name	Description
0	header	= 0x81
1	flag	0x00 = failure 0x01 = success
2	length	= 2
3	command	= 0x18
4	GPI0	GPI2 state (0 or 1). **0xFF = invalid**
5	GPI1	GPI3 state (0 or 1). **0xFF = invalid**

Table 12: UDP Based Control Commands return packet field for Command 0x19

Byte Offset	Name	Description
0	header	= 0x81
1	flag	0x00 = failure 0x01 = success
2	length	= 2
3	command	= 0x19
4	GPO0	GPO2 state (0 or 1)
5	GPO1	GPO3 state (0 or 1)

5 Reader Commands

Reader commands are TCP/IP commands that provide commands to the RFID reader board. **These packets are all transmitted using Transmission Control Protocol (TCP) with port 1515.** The format is defined in Appendix A. There are 2 main categories of Reader Commands:

1. Reader Initialization
2. Tag Operations

5.1 Reader Initialization

There are 4 steps within Reader Initialization that MUST be done:

Step 1: Open Radio

Step 2: Set RF Power

Step 3: Set Channel

Step 4: Set Profile

5.1.1 Radio Open

Packet Name	Command Data	Description
ABORT control command	40:03:00:00:00:00:00:00	Control command to abort operation.
ABORT control command return	40:03:bf:fc:bf:fc:bf:fc	Command response of abort operation.

5.1.2 Set RF Power

Packet Name	Command Data	Description
REG_REQ, Write Register	70:01:01:07:00:00:00:00	ANT_PORT_SEL (See appendix A)
REG_REQ, Write Register	70:01:06:07:2c:01:00:00	ANT_PORT_POWER (See appendix A)

5.1.3 Set Channel (For ETSI only)

Step 1: Disable all available channels

Step 2: Enable only the desired channel.

Example of disable channel 0.

Packet Name	Command Data	Description
REG_REQ	70:01:01:0c:00:00:00:00	FREQCH_SEL (see appendix A)
REG_REQ	70:01:02:0c:00:00:00:00	FREQCH_CFG (see appendix A)

Example of enable channel 1.

Packet Name	Command Data	Description
REG_REQ	70:01:01:0c:01:00:00:00	FREQCH_SEL (see appendix A)
REG_REQ	70:01:02:0c:01:00:00:00	FREQCH_CFG (see appendix A)

5.1.4 Set Profile

Packet Name	Command Data	Description
REG_REQ	70:01:60:0b:02:00:00:00	CURRENT_PROFILE(see appendix A)
REG_REQ	70:01:00:f0:19:00:00:00	HST_CMD (see appendix A)
Command-Begin Packet	02:00:00:80:02:00:00:00: 19:00:00:00:13:41:00:00	See appendix A.
Command-End Packet	02:00:01:80:02:00:00:00: 16:41:00:00:00:00:00:00	See appendix A.

5.2 *Tag Operations: Inventory*

There are 3 steps within Inventory that MUST be done:

Step 1: Set Inventory Parameters

Step 2: Set Inventory Algorithm

Step 3: Start/Stop Inventory

5.2.1 Set Inventory Parameters

Packet Name	Command Data	Description
REG_REQ	70:01:00:07:ff:ff:00:00	ANT_CYCLES (See appendix A)
REG_REQ	70:01:00:09:00:00:00:00	QUERY_CFG (See appendix A)

5.2.2 Set Inventory Algorithm

REG_REQ	70:01:02:09:01:00:00:00	INV_SEL (See appendix A)
REG_REQ	70:01:03:09:f7:00:50:03	INV_ALG_PARM_0 (See appendix A)
REG_REQ	70:01:01:09:01:00:00:00	INV_CFG (See appendix A)

5.2.3 Start/Stop Inventory

REG_REQ	70:01:00:f0:0f:00:00:00	HST_CMD (See appendix A)
Command-Begin Packet	02:01:00:80:02:00:00:00: 0f:00:00:00:61:44:00:00	See appendix A.
Inventory-Response Packet	02:00:05:80:07:00:00:00: 73:44:00:00:81:5f:83:06: 00:00:00:00:30:00:10:00: 00:00:00:00:00:00:00:00: 06:87:71:34	See appendix A. (This is the tag info during inventory)
ABORT control command	40:03:00:00:00:00:00:00	This is the command you use to stop inventory.
ABORT control command return	40:03:bf:fc:bf:fc:bf:fc	This is the command response you will receive.

Note: After the “ABORT” command to stop inventory, a 2 seconds delay is required for the reader to clear buffer before it can execute another command

5.3 Tag Operations: Read

There are 4 steps within Read that MUST be done:

Step 1: Set Inventory Parameters

Step 2: Set Inventory Algorithm. (Assume only 1 tag in front of reader)

Step 3: Select the desired tag.

Step 4: Start reading the tag data.

The followings example assumes reading TID bank.

5.3.1 Step 1 – Set Inventory Parameters

Packet Name	Command Data	Description
REG_REQ	70:01:00:07:01:00:00:00	ANT_CYCLES (See appendix A)
REG_REQ	70:01:00:09:80:01:00:00	QUERY_CFG (See appendix A)

5.3.2 Step 2 – Set Inventory Algorithm (Fixed Q = 0)

REG_REQ	70:01:02:09:00:00:00:00	INV_SEL (See appendix A)
REG_REQ	70:01:03:09:00:00:00:00	INV_ALG_PARM_0 (See appendix A)
REG_REQ	70:01:05:09:00:00:00:00	INV_ALG_PARM_2 (See appendix A)

5.3.3 Step 3 – Select the desired tag (e.g.111122223333444455556666)

REG_REQ	70:01:01:08:09:00:00:00	TAGMSK_DESC_CFG (See appendix A 1)
REG_REQ	70:01:02:08:01:00:00:00	TAGMSK_BANK (See appendix A)
REG_REQ	70:01:03:08:20:00:00:00	TAGMSK_ PTR (See appendix A)
REG_REQ	70:01:04:08:60:00:00:00	TAGMSK_ LEN (See appendix A)
REG_REQ	70:01:05:08:11:11:22:22	TAGMSK_ 0_3 (See appendix A)
REG_REQ	70:01:06:08:33:33:44:44	TAGMSK_ 4_7 (See appendix A)
REG_REQ	70:01:07:08:55:55:66:66	TAGMSK_ 8_11 (See appendix A)
REG_REQ	70:01:01:09:40:40:00:00	INV_CFG (See appendix A)

5.3.4 Step 4 – Start reading tag data

REG_REQ	70:01:02:0a:02:00:00:00	TAGACC_BANK (See appendix A)
REG_REQ	70:01:03:0a:00:00:00:00	TAGACC_PTR (See appendix A)
REG_REQ	70:01:04:0a:02:00:00:00	TAGACC_CNT (See appendix A)
REG_REQ	70:01:06:0a:00:00:00:00	TAGACC_ACCPWD (See appendix A)
REG_REQ	70:01:00:f0:10:00:00:00	HST_CMD (See appendix A)

5.3.5 Responses from reader

Command-Begin Packet	02:00:00:80:02:00:00:00: 10:00:00:00:d6:8b:00:00	See appendix A.
Inventory-Response Packet	02:00:05:80:07:00:00:00: ec:8b:00:00:00:00:00:00: 00:00:00:00:30:00:11:11: 22:22:33:33:44:44:55:55: 66:66:18:35:	See appendix A. (This is the selected tag)
Tag-Access Packet	01:00:06:00:04:00:00:00: f0:8b:00:00:c2:00:00:00: 00:00:00:00:e2:00:10:50	See appendix A. (this is the memory bank to be read)
Command-End Packet	02:00:01:80:02:00:00:00: f5:8b:00:00:00:00:00:00	See appendix A.

5.4 Tag Operations: Write

There are 4 steps within Write that MUST be done:

Step 1: Set Inventory Parameters

Step 2: Set Inventory Algorithm. (Assume only 1 tag in front of reader)

Step 3: Select the desired tag.

Step 4: Start writing the tag data.

The followings example assumes reading TID bank.

5.4.1 Set Inventory Parameters

Packet Name	Command Data	Description
REG_REQ	70:01:00:07:01:00:00:00	ANT_CYCLES (See appendix A)
REG_REQ	70:01:00:09:80:01:00:00	QUERY_CFG (See appendix A)

5.4.2 Set Inventory Algorithm (Fixed Q = 0)

REG_REQ	70:01:02:09:00:00:00:00	INV_SEL (See appendix A)
REG_REQ	70:01:03:09:00:00:00:00	INV_ALG_PARM_0 (See appendix A)
REG_REQ	70:01:05:09:00:00:00:00	INV_ALG_PARM_2 (See appendix A)

5.4.3 Select the desired tag (e.g.111122223333444455556666)

REG_REQ	70:01:01:08:09:00:00:00	TAGMSK_DESC_CFG (See appendix A 1)
REG_REQ	70:01:02:08:01:00:00:00	TAGMSK_BANK (See appendix A)
REG_REQ	70:01:03:08:20:00:00:00	TAGMSK_ PTR (See appendix A)
REG_REQ	70:01:04:08:60:00:00:00	TAGMSK_ LEN (See appendix A)
REG_REQ	70:01:05:08:11:11:22:22	TAGMSK_ 0_3 (See appendix A)
REG_REQ	70:01:06:08:33:33:44:44	TAGMSK_ 4_7 (See appendix A)
REG_REQ	70:01:07:08:55:55:66:66	TAGMSK_ 8_11 (See appendix A)
REG_REQ	70:01:01:09:40:40:00:00	INV_CFG (See appendix A)

5.4.4 Start writing tag EPC data (e.g.000022223333444455556666)

REG_REQ	70:01:01:0a:0f:00:00:00	TAGACC_DESC_CFG (See appendix A)
REG_REQ	70:01:02:0a:01:00:00:00	TAGACC_BANK (See appendix A)
REG_REQ	70:01:03:0a:00:00:00:00	TAGACC_PTR (See appendix A)
REG_REQ	70:01:04:0a:06:00:00:00	TAGACC_CNT (See appendix A)
REG_REQ	70:01:06:0a:00:00:00:00	TAGACC_ACCPWD (See appendix A)
REG_REQ	70:01:09:0a:00:00:02:00	TAGWRDAT_0 (See appendix A)

REG_REQ	70:01:0a:0a:22:22:03:00	TAGWRDAT_1 (See appendix A)
REG_REQ	70:01:0b:0a:33:33:04:00	TAGWRDAT_2 (See appendix A)
REG_REQ	70:01:0c:0a:44:44:05:00	TAGWRDAT_3 (See appendix A)
REG_REQ	70:01:0d:0a:55:55:06:00	TAGWRDAT_4 (See appendix A)
REG_REQ	70:01:0e:0a:66:66:07:00	TAGWRDAT_5 (See appendix A)
REG_REQ	70:01:00:f0:11:00:00:00	HST_CMD (See appendix A)
Command-Begin Packet	02:00:00:80:02:00:00:00: 11:00:00:00:d6:8b:00:00	See appendix A.
Tag-Access Packet	01:00:06:00:03:00:00:00: 10:76:01:00:c3:00:00:00: 00:00:00:00	See appendix A.
Command-End Packet	02:00:01:80:02:00:00:00: f5:8b:00:00:00:00:00:00	See appendix A.

5.5 Enable G2XM and G2XL EAS-Bit by ChangeEAS Command

1. The tag must have a non-zero access password value preset to it.
2. Select the tag so that the tag enters Ready State
3. Set reader MAC register 0xA01 bit 22 to 1 for enable
4. Send ChangeEAS command: HST_CMD: 0x00000026
5. A Tag Access Response reply with success should come back
6. The G2XM or G2XL tag now has the EAS Alarm enabled so that whenever it receives a EAS_Alarm command it will respond with a 64 bit ID and will be decoded by reader. See next section 4.6 to see how that can be done – notice that with CSL reader it will loop this EAS_Alarm internally without the need for user application to send this EAS_Alarm repeatedly, which is not practical.

5.6 Tag Operations: EAS Alarm

One can configure specific port to do EAS alarm detection and other ports to do normal inventory. In other words, a single reader can perform multiple functions. The only point to be careful is the cycling time to the EAS alarm detection port must be short enough so that the EAS alarm tag would not move too fast through the gate so that it is missed.

There are 3 steps within EAS alarm that MUST be done:

Step 1: Set Port Parameters

Step 2: Start EAS detection & Inventory

Step 3: Stop EAS detection & Inventory

5.6.1 Set Port Parameters

in this example, Port 0 is selected to do normal inventory

Packet Name	Command Data	Description
REG_REQ	70:01:01:07:00:00:00:00	ANT_PORT_SEL (See appendix A)
REG_REQ	70:01:02:07:01:00:00:00	ANT_PORT_CFG (See appendix A)

5.6.2 Set Port Parameters

in this example, Port 1 is selected to detect EAS alarm tag – it will automatically loop to send EAS_Alarm command

Packet Name	Command Data	Description
REG_REQ	70:01:01:07:01:00:00:00	ANT_PORT_SEL (See appendix A)
REG_REQ	70:01:02:07:01:00:10:00	ANT_PORT_CFG (See appendix A)

5.6.3 Start/Stop EAS detection & Inventory

note that from Port 0 we receive "inventory-response packets, while in Port 1 we receive "tag-access packets

REG_REQ	70:01:00:f0:0f:00:00:00	HST_CMD (See appendix A)
Command-Begin Packet	02:01:00:80:02:00:00:00: 0f:00:00:00:61:44:00:00	See appendix A.
Inventory-Response Packet	02:00:05:80:07:00:00:00: 73:44:00:00:81:5f:83:06: 00:00:00:00:30:00:10:00:	See appendix A. (This is the tag info during inventory of port 0)

	00:00:00:00:00:00:00:00: 06:87:71:34	
Tag-Access Packet	01:00:06:00:03:00:00:00: 10:76:01:00:04:00:01:00: 00:00:00:00	See appendix A. (EAS is detected in port 1)
ABORT control command	40:03:00:00:00:00:00:00	This is the command you use to stop EAS detection.
ABORT control command return	40:03:bf:fc:bf:fc:bf:fc	This is the command response you will receive.

Notes:

1. After the "ABORT" command, a 2 seconds delay is required for the reader to clear buffer before it can execute another command

6 Common Issues

6.1 Keep Alive

Keep alive, a very common feature of low level TCP/IP stacks, available from PC server, is often not available from embedded system (the host processor side).　Therefore, the host processor does not have an easy means to detect network connectivity issues and automatically recovering from it.　Because of that, the host processor cannot easily distinguish between the situation of no tags and the situation of no connection.

The CSL device provides 1 type of Keep Alive:

TCP Keep Alive:　for host processor where its TCP stack has keep alive, this is the easiest to do.　Typical host processors with this TCP Keep Alive includes all Windows based servers (WinXP, Win7, etc.)

Appendix A - CSL RFID Firmware Command Specification

A.1 Introduction

To configure and control the RFID module, one has to:

1) Understand the differences between low level API and high level API (Section A.2)
2) How to read and write MAC firmware registers (Section A.3)
3) Know the definition of each MAC firmware registers (Section A.4)
4) Use HST_CMD commands to tell the RFID module to start various operations (end of Section A.4) – sending a command to MAC is achieved by writing the command to a special MAC firmware register 0xF000, hence it is described in Section A.2.
5) Know the definition of R2000 registers. Need to use MAC Bypass Read and Write HST_CMD 0x05 and 0x06 to access them. (Section A.5)
6) Know the definition of OEM registers. Need to use OEM Write and Read HST_CMD 0x02 and 0x03 to access them. (Section A.6)
7) Receive Command State Packets to monitor RFID module operation status and receive RFID tag info (Section A.7)
8) Use Control Command to stop operations. (Section A.8)

There are 3 types of registers that the application needs to access (read or write):

1) MAC registers: these are the most common one to be accessed. Use **REG_REQ** to access.

2) OEM register: only 1 register, 0xA2, is of concern to the application. Use **HST_CMD 0x02 and 0x03** to access

3) R2000 register: only 1 register, 0x0450, is of concern to the application. Use MAC Bypass **HST_CMD 0x05 and 0x06** to access.

A.2 High Level API vs Low Level API

Due to historical reasons, the CSL byte protocol has a high level API version and a low level API version. The differences are only in Byte 0: Packet Version (pkt_ver) byte, Byte 1: Access Flag (access_flg) byte and Byte 2 & 3: the Packet Type (pkt_type) word of each packet. These are specially remarked in the following pages so that the differences can be easily understood. The reader itself actually accepts both high level and low level API seamlessly. However, the responses going back to the PC host can be either high level or low level API depending on customer's own choice (which is usually affected by what they historically used).

All CS108 and CS463 reader are set default to high level API ex-factory. Therefore for new developer it is simpler just to assume High Level API. Although the examples in Appendix C are shown in Low Level API format, it is quite easy to convert them to High Level API. Just refer to the remaining Appendices A.3-8 to know how to convert between them.

Whether the reader behaves as a High Level API reader or a Low Level API reader is defined in an OEM location 0x000000A2: 0 means high level; 1 means low level. Default is high level. One can change that by writing to OEM address using HST_CMD 0x00000002 (need to first write the desired OEM address to MAC register OEM_ADDR; write the value you want to set to MAC register OEM_DATA, and then after that write the value 0x00000002 to MAC register HST_CMD (0xF000))

As mentioned above, whether a packet is high level or low level is mainly determined by pkt_ver, which is the first byte of the byte stream communication packet (so called byte 0):

Downlink/Uplink	**pkt_ver**	**Packet Category/Types**
Downlink	**70(low level)** **00/01(high level – read/write)**	MAC Register Read or Write Request, For high level, 00 means Read, 01 means Write For low level, Read and Write are determined by access_flg, the second byte: 00 in second byte means read, 01 in second byte means write

Uplink	**70(low level)**
	00(high level)
		MAC Register Read Response

Uplink	**02(low level)**
	01(high level)
		R2000 IC command state packets such as command begin, command end, etc.

Uplink	**03 (same for high level or low level)**	inventory packet
Uplink	**04 (same for high level or low level)**	compact mode inventory packet
Uplink	**01 (same for high level or low level)**	Tag Access packet (actually tag access packet is also a command state packet – we need to describe this in a separate row because tag access packets for low level API and high level API BOTH use pkt_ver = 01)
Downlink and Uplink	**40 (same for high level or low level)**	Abort command
Uplink	**01 (same for high level or low level)**	OEM Register Read Response and R2000 Register Read Response packet

Note that pkt_ver for the last 5 rows are the same whether the reader is set to high level or low level.

Note that although pkt_ver = 01 appears in 3 uplink rows above, but their pkt_type are different. So they can be differentiated.

So, if you are tracking the packets in Wireshark, and just looking at the byte 0, packet version, and the logic is

Downlink (from PC to reader):

Whenever you see 70/00/01:	MAC register read or write request and response

Uplink (from reader to PC):

Whenever you see 70/00:	MAC register read response
Whenever you see 02/01/03/04:	Command state packets (e.g., inventory packet, compact mode inventory packet, tag access packets)

Uplink or Downlink:

Whenever you see 40:	Abort command and Abort Response command

To further differentiate them, you need to also look at byte 1: flag; and byte 2 &3: packet type.

In summary, the process of decoding basic category of packets requires knowledge of Byte 0, 1, 2 and 3. Byte 0 is packet version, Byte 1 is flag, Byte 2 and 3 are packet type.

A.3 MAC Register Access Packet (Downlink)

REG_REQ packet format

These packets are sent from Host to firmware to access (read or write) a firmware register. This is a Downlink packet.

Byte Offset(s)	**Name**	**Description**
0	**Header (pkt_ver)**	**0x70 for low level** **(for high level API, Byte 0 is used to define write or read: write = 1, read = 0)**
1	access_flg	**For low level:** **0 = read** **1 = write** **(for high level API, Byte 1 is always 0)**
3:2	reg_addr	16bit address of the register to be written. **Note that Byte 3 is the high byte, Byte 2 is the low byte !!!!! Example, an address of 0xF000 will become 00FO in the packet.***
7:4	reg_data	For *write register* this is the 32 bits of data to be written. For *read register* set to zeros. **Again, note the reverse order of byte placement. Byte 7 is the first byte, byte 6 is second byte, and so on.***

***Note that for RFID firmware commands and replies, when a field is more than 1 byte, it is always REVERSELY POPULATED. For example, if a field contains 2 bytes, then the first byte is the LOW BYTE, and second byte is the HIGH BYTE. If a field contains 3 bytes, then the first byte is the LSB, second byte is the mid byte, third byte is the MSB. Likewise for fields containing 4 bytes, 5 bytes, etc. This is called byte-swapped.**

REG_RESP packet format

These are response packets returned to the host while the firmware is not in command state. **This response packet only comes back when the operation is Read register. There is no response if the operation is write register.** This is an uplink packet.

Byte Offset(s)	**Name**	**Description**
0	**header (pkt_ver)**	**0x70 for Low Level (0x00 for High Level)**
1	reserved	Read as zero
3:2	reg_addr	16 bit address of the register to be read (echo back for confirmation and easy tracking)
7:4	reg_data	These are the 4 bytes of register data you want to read

A.4 MAC Registers Description

The following are the relevant MAC registers that can be read or written to according to A.1.

Reminder: please note that for RFID firmware commands and replies, when a field is more than 1 byte, it is always REVERSELY POPULATED (byte-swapped). For example, if a field contains 2 bytes, then the first byte is the LOW BYTE, and second byte is the HIGH BYTE. If a field contains 3 bytes, then the first byte is the LSB, second byte is the mid byte, third byte is the MSB. Likewise for fields containing 4 bytes, 5 bytes, etc.

FIRMWARE_VER

Holds the firmware version information

reg_addr = 0x0000

Bits	**Name**	**Description**
11:0	Patch Version	12 bit Patch version number of firmware image
23:12	Minor Version	12 bit Minor version number of firmware image
31:24	Major Version	8 bit Major version number of firmware image

IMPINJ_EXTENSIONS (TagFocus and FastID)

This register provides support for Impinj specific extensions

reg_addr = 0x0203

Bits	**Name**	**Description**
3:0	blockwrite_mode	Determines the maximum number of words to write per BlockWrite transaction with the tag. 0 = Auto-detect (Default). One or two word BlockWrite will be determined automatically. 1 = Force one word BlockWrite. Unconditionally use one word BlockWrites in all cases. 2 = Force two word BlockWrite. Unconditionally use two word BlockWrites in all cases. A protocol error will occur if the tags in the field do not support this feature. 3-15 = Reserved for future use
4:4	tag_focus	If this feature is enabled, once a tag has been singulated it will remain out of the tag population (the tag's session 1 inventoried flag remains in B state) until the inventory operation is complete. This feature is only effective in conjunction with other inventory controls Session=S1, target=A only (toggle disabled)

		0=disabled 1=enabled
5:5	fast_id	If this feature is enabled and an Impinj tag is in the field, then the 6-word TID will be returned along with the EPC when the tag is singulated. The first 5 bits of PC will be automatically adjusted to account for the length increase. 0=disabled 1=enabled
31:6	reserved	reserved

PROTSCH_SMIDX

Indicate the FCC or ETSI country set. **(read only)**

reg_addr = 0x0300

Bits	Name	Description
31:0	country	0 = FCC 1 = ETSI

PROTSCH_SMCFG

Specify the configuration bits for the current protocol state machine.

reg_addr = 0x0301

FCC Country:

Bits	Name	Description
31:0	reserved	read / write as zero

ETSI Country:

Bits	Name	Description
0	en_lbt	enable LBT. 0 = off 1 = on
1	en_scan	enable Scan Mode - if an interferer is detected by the LBT logic the next enabled frequency channel is selected. LBT will also be performed on the new channel before use. The logic is repeated until a channel is found with no interferer. **Recommended to set 1 when LBT is on.**
2	reserved	read / write as zero
3	dis_eoff	“disable early cw off” when no tags found. In other words, don’t turn off CW when tags aren’t immediately found. This is useful for applications

		with fast moving tag populations (ex. conveyor applications). Where the device may need to continuously look for tags without lowering the carrier before the TX ON time limit is reached.
7:4	reserved	read / write as zero
15:8	lbt_rssi_thr	8 bit rssi threshold value. During LBT operation, RSSI readings >= to this threashold indicate to the LBT logic that an interferer is present. **Recommended value is 0x2F**.
31:16	reserved	read / write as zero

PROTSCH_SMCFG_SEL

Used to select the desired bank for the PROTSCH_SMCFG register

reg_addr = 0x0304

Bits	**Name**	**Description**
31:0	index	For ETSI, set to 1 before changing PROTSCH_SMCFG register.

ANT_CYCLES

Specify the number of times the enabled logical antenna port should be cycled through in order to complete protocol command execution. In a single antenna cycle, the firmware scans all available logical antennas starting at #0.

reg_addr = 0x0700

Bits	**Name**	**Description**
15:0	cycles	The number of antenna cycles to be completed for command execution 0x0001 = once cycle through enabled antennas (Non-continuous mode) 0xFFFF = cycle forever until a ABORT is received (Continuous mode)
17:16	mode	Antenna sequence mode **(for CS468 only)** 0x00 = normal mode (antenna switch from port 0 to port 15 sequentially) 0xX1 = sequence mode (antenna switch based on the sequence stored in OEM table address 0xA7-0xAC) 0x1X = smart check mode (quick switch to next enabled antenna if no tag is detected.)
23:18	sequence size	Sequence size of sequence mode. Max = 48. **(for CS468 only)**
24	freq_agile	Frequency Agile mode. 0 = disable 1 = enable
31:25	reserved	reserved - read / write as 0

ANT_PORT_SEL

Select the antenna port.

reg_addr = 0x0701

.Bits	Name	Description
31:0	port_idx	zero based index of descriptor to select for port. Valid values are 0x0 through 0xF.

ANT_PORT_CFG

Sets bits in this register to indicate the configuration of the logical antenna specified by the ANT_PORT_SEL register.

reg_addr = 0x0702

Bit	Name	Description
0	Enable	1=logical antenna enabled 0=logical antenna disabled
1	Inv_mode	0 = Global mode (use global parameters). CS108 must set as 0. 1 = Local mode (use port dedicated parameters)
3:2	Inv_algo	Inventory algorithm to use.
7:4	StartQ	Starting Q value. 0 - 15
8	Profile_mode	0 = Global mode (use last CURRENT_PROFILE parameters). CS108 must set as 0. 1 = Local mode (use port dedicated parameters)
12:9	Profile	0-3
13	Freq_mode	0 = Global mode (use first enabled frequency). CS108 must set as 0. 1 = Local mode (use port dedicated frequency)
19:14	Freq_chan	Frequency channel
20	EAS_enable	1=EAS detection enabled 0=EAS detection disabled
31:21	RFU	reserved - read / write as 0

ANT_PORT_DWELL

Set the dwell time for the logical antenna selected by ANT_PORT_SEL.

reg_addr = 0x0705

Bits	Name	Description
31:0	time_ms	number of milliseconds to communicate on this antenna during a

		given Antenna Cycle 0x00000000 indicates that dwell time should not be used.

ANT_PORT_POWER

Set the output power for the logical antenna selected by ANT_PORT_SEL.

reg_addr = 0x0706

Bits	**Name**	**Description**
31:0	power	power in steps of 0.1dBm (Maximum value is 300)

ANT_PORT_INV_CNT

Set how many inventory rounds should be executed for the logical antenna selected by ANT_PORT_SEL.

reg_addr = 0x0707

Bits	**Name**	**Description**
31:0	count	number of inventory rounds for current port 0x00000000 indicates that inventory round count should not be used.

Power up defaults for each bank (bank selected using ANT_PORT_SEL register):

ANT_PORT_SEL	**ANT_PORT_CFG**	**ANT_PORT_DWELL**	**ANT_PORT_POWER**	**ANT_PORT_INV_CNT**
0x0	0x00000001	0x000007D0	0x0000012C	0x00002000
0x1	0x00000000	0x000007D0	0x0000012C	0x00002000
0x2	0x00000000	0x000007D0	0x0000012C	0x00002000
0x3	0x00000000	0x000007D0	0x0000012C	0x00002000
0x4	0x00000000	0x000007D0	0x0000012C	0x00002000
0x5	0x00000000	0x000007D0	0x0000012C	0x00002000
0x6	0x00000000	0x000007D0	0x0000012C	0x00002000
0x7	0x00000000	0x000007D0	0x0000012C	0x00002000
0x8	0x00000000	0x000007D0	0x0000012C	0x00002000
0x9	0x00000000	0x000007D0	0x0000012C	0x00002000
0xA	0x00000000	0x000007D0	0x0000012C	0x00002000
0xB	0x00000000	0x000007D0	0x0000012C	0x00002000
0xC	0x00000000	0x000007D0	0x0000012C	0x00002000
0xD	0x00000000	0x000007D0	0x0000012C	0x00002000
0xE	0x00000000	0x000007D0	0x0000012C	0x00002000
0xF	0x00000000	0x000007D0	0x0000012C	0x00002000

TAGMSK_DESC_SEL

Write this register to select which *Select* descriptor and corresponding mask register set to access.

Total of 8 selects can be **AND** together.

reg_addr = 0x0800

Power up default = 0x00000000

Bit	**Name**	**Description / Values**
2:0	desc_idx	Index of the Select descriptor to be used.
31:3	reserved	reserved - read / write as zero

TAGMSK_DESC_CFG

Specify the parameters for a *Select* operation. When an inventory operation is performed and the tag select feature is enabled, these descriptors will be scanned. Each descriptor that has its enable bit set will cause a select command to be issued prior to the inventory.

reg_addr = 0x0801

Power up default = 0x00000000

Bit	**Name**	**Description / Values**
0	Enable	1 descriptor enabled, 0 descriptor disabled

3:1	sel_target	*Target* value for *Select* operations.
		0 = Inventoried(S0);
		1 = Inventoried(S1);
		2 = Inventoried(S2);
		3 = Inventoried(S3);
		4 = SL; **(Recommend to set this value)**

6:4	sel_action	what action to perform on *inventories* or *SL* flags as indicated to tags during *Select* operation. **Recommend to set zero**.
7:7	trunk_en	truncate enable bit value - passed to underlying Select operation.
15:8	delay	CW hold time in ms after Select command.
31:16	reserved	reserved - write as zero

sel_action:

sel_action	**Tag Matching**	**Tag Not-Matching**
000	assert **SL** or **inventoried** → *A*	deassert **SL** or **inventoried** → *B*
001	assert **SL** or **inventoried** → *A*	do nothing
010	do nothing	deassert **SL** or **inventoried** → *B*
011	negate **SL** or (*A* → *B*, *B* → *A*)	do nothing
100	deassert **SL** or **inventoried** → *B*	assert **SL** or **inventoried** → *A*
101	deassert **SL** or **inventoried** → *B*	do nothing
110	do nothing	assert **SL** or **inventoried** → *A*
111	do nothing	negate **SL** or (*A* → *B*, *B* → *A*)

TAGMSK_BANK

Specify which memory bank is applied to during *Select.*

18000-6C memory banks list as follows:

0x0 - Reserved

0x1 - EPC

0x2 - TID

0x3 - USER

reg_addr = 0x0802

Power up default = 0x00000000

Bits	**Name**	**Description**
1:0	mask_bank	Mask bank selection
31:2	reserved	reserved - read / write as 0

TAGMSK_PTR

Specify the bit offset in tag memory at which the configured mask will be applied during *Select*.

Used in conjunction with TAGMSK_BANK, TAGMSKPTR and TAGMSK[0,31] registers.

reg_addr = 0x0803

Power up default = 0x00000000

Bits	**Name**	**Description**
31:0	ptr	offset to be used with the mask. 0x00000000 - 0xFFFFFFFF supported

TAGMSK_LEN

Specify the number of valid bits in TAGMSK[0,31] registers that will be sent with the underlying *Select* command.

reg_addr = 0x0804

Power up default = 0x00000000

Bits	**Name**	**Description**
7:0	length	number of valid bits the combined TAGMSK[0,31] registers.
31:8	reserved	reserved - read / write as 0

TAGMSK_0_3

Write the tag mask data to the TAGMSK_xx registers for use with subsequent *Select* commands.

reg_addr = 0x0805

Power up default = 0x00000000

When the R1000 firmware executes a *Select* command it scans the tag mask registers as a byte array, starting with the low order byte of the TAGMSK_0_3 and continues until it scans the number of bits specified in the TAGMSK_LEN register. For a non-8 bit aligned tag mask the final bits are left justified (high order bits) in the final byte. Consider the following example:

Tag mask = {0xFF, 0x07}

Length = 0xB

1) Host writes 0x000070FF to register TAGMSK_0_3

2) Host writes 0x0000000B to register TAGMSK_LEN

3) When firmware executes a *Select* operation it will build the mask by scanning 0xB bits from TAGMSK_0_3 starting at the low byte. The final, non-8 bit alligned mask bits, are taken from the second byte of TAGMSK_0_3 using the logic : final_bits = (8-(TAGMSK_LEN % 8)).

Bits	**Name**	**Description**
31:8	reserved	reserved - read / write as 0

TAGMSK_4_7

See TAGMSK_0_3 for details.

reg_addr = 0x0806

Power up default = 0x00000000

TAGMSK_8_11

See TAGMSK_0_3 for details.

reg_addr = 0x0807

Power up default = 0x00000000

TAGMSK_12_15

See TAGMSK_0_3 for details.

reg_addr = 0x0808

Power up default = 0x00000000

TAGMSK_16_19

See TAGMSK_0_3 for details.

reg_addr = 0x0809

Power up default = 0x00000000

TAGMSK_20_23

See TAGMSK_0_3 for details.

reg_addr = 0x080A

Power up default = 0x00000000

TAGMSK_24_27

See TAGMSK_0_3 for details.

reg_addr = 0x080B

Power up default = 0x00000000

TAGMSK_28_31

See TAGMSK_0_3 for details.

reg_addr = 0x080C

Power up default = 0x00000000

QUERY_CFG

Configure parameters used in underlying Query and inventory operations.

reg_addr = 0x0900

Bit	Name	Description / Values
3:0	reserved	Reserved - read / write as zero
4	query_target	Starting *Target* argument (A or B) to underlying *Query* 0 = A; 1 = B.
6:5	query_session	*Session* argument to underlying *Query* 0 = S0; 1 = S1; 2 = S2; 3 = S3.
8:7	query_sel	*Select* argument to underlying *Query* 0 = All; 1 = All; 2 = ~SL; 3 = SL; **Recommend to Set 0 for inventory operation.** **Recommend to Set 3 for tag select operation. Reference to INV_CFG and TAGMSK_DESC_CFG.**
31:9	reserved	reserved - read / write as zero

INV_CFG

Inventory configuration. Configure parameters used in underlying inventory operations

reg_addr = 0x0901

Power up default = 0x00000001

Bit	Name	Description / Values
5:0	Inv_algo	Inventory algorithm to use.
13:6	match_rep	Stop after "N" tags inventoried (zero indicates no stop)
14	tag_sel	1 = enable tag select prior to inventory, read, write, lock or kill. 0 = no select issued.
15	disable_inventory	Do not run inventory.
17:16	tag_read	0 = no tag read issued. 1 = enable read 1 bank after inventory. 2 = enable read 2 banks after inventory
18	crc_err_read	0 = disable crc error read 1 = enable crc error read
19	QT_mode	0 = disable QT temporary read private EPC 1 = enable QT temporary read private EPC
25:20	tag_delay	Time delay for each tag in ms (use to reduce tag rate), 6 bits, binary. For Bluetooth normal mode inventory (see byte 26 below), this should be set to 30. For Bluetooth compact mode inventory (see byte 26 below), this should be set to 0 or some small values, most desirably less than 7. For USB, this should be set to 0.
26	inv_mode	0 = normal mode 1 = compact mode. See Inventory-Response Packet (Compact mode)
27	brand_ID	0 = disable brand ID for Ucode8 1 = enable brand ID for Ucode8*
31:28	reserved	reserved - read / write as zero

*NXP Ucode 8 has a new feature where a Brand ID can be saved ex-factory for brand owners of the world (e.g. LV). This Brand ID is 16 bits long (2 bytes, 4 hex numbers). When you select EPC bank with offset=204h, length=1, mask=1b, then the inventory results in a LONGER number (PC becomes bigger!!) with the last 4 hex numbers being the Brand ID. The actual data is scrambled and will only be descrambled if you set bit 27 of INV_CFG register to 1.

INV_SEL

Select which set of algorithm parameter registers to access.

reg_addr = 0x0902

Bits	Name	Description
31:0	desc_idx	zero based index of descriptor to access 0 through 3.

INV_ALG_PARM_0

The algorithm that will be used for the next Inventory command is specified by the register INV_CFG. The definition of each register varies depending on the algorithm chosen. For instance, if you wish to set the parameters for algorithm 1, then set INV_SEL to 1 and load PARM as specified.

reg_addr = 0x0903

Power up defaults for each bank (bank selected using HST_INV_SEL register):

Algorithm #	Power up Value
0x0	0x00000004
0x3	0x000040F4

The following tables indicate how each algorithm interprets the value for its bank:

Algorithm 0 (Fixed Q)

Bits	Name	Description
3:0	Q value	Q value that algorithm 0 will use when running an inventory

Algorithm 3 (Dynamic Q Algorithm 3)

Bits	Name	Description
3:0	StartQ	Starting Q value
7:4	MaxQ	Maximum Q value
11:8	MinQ	Minimum Q Value
19:12	Tmult	Threshold multiplier. This is a fixed point fraction with the decimal point between bit 2 and 3. The field looks like bbbbbb.bb which allows fractional values of ½, ¼ and ¾.

INV_ALG_PARM_1

The algorithm that will be used for the next Inventory command is specified by the register INV_CFG. The definition of each register varies depending on the algorithm chosen. For instance, if you wish to set the parameters for algorithm 1, then set INV_SEL to 1 and load PARM as specified.

reg_addr = 0x0904

Power up defaults for each bank (bank selected using HST_INV_SEL register):

Algorithm #	Power up Value
0x0	0x00000000
0x1	0x00000000
0x2	0x00000000
0x3	0x00000000

The following tables indicate how each algorithm interprets the value for its bank:

All Algorithms

Bits	Name	Description
7:0	Retry	Number of times to retry a query / query rep sequence for the session/target before flipping the target or exiting. For example, if q is 2 then there will be one query and 4 query reps for each retry.

INV_ALG_PARM_2

The algorithm that will be used for the next Inventory command is specified by the register INV_CFG. The definition of each register varies depending on the algorithm chosen. For instance, if you wish to set the parameters for algorithm 1, then set INV_SEL to 1 and load PARM as specified.
reg_addr = 0x0905

Power up defaults for each bank (bank selected using HST_INV_SEL register):

Algorithm #	Power up Value
0x0	0x00000003
0x1	0x00000001
0x2	0x00000001
0x3	0x00000001

The following tables indicate how each algorithm interprets the value for its bank:

Algorithm 0

Bits	Name	Description
0	Toggle	If set to one, the target will flip from A to B or B to A after all rounds have been run on the current target. This is done after no tags have been read if continuous mode is selected and all retry's have been run.

1	RunTillZero	Continue running inventory rounds until a round is completed without reading any tags.

Algorithm 1-3

Bits	**Name**	**Description**
0	Toggle	If set to one, the target will flip from A to B or B to A after all rounds have been run on the current target. This is done after no tags have been read if continuous mode is selected and all retry's have been run.

TAGACC_DESC_CFG

Tag access configuration register

reg_addr = 0x0A01

Power up default = 0x00000000

Bits	**Name**	**Description**
0	Verify	Verify after write. **Must set as 1 if writing tag.**
5:1	Retry	Number of time to retry the operation if failed
31:6	Reserved	reserved - read / write as zero

TAGACC_BANK

Specify which memory bank is to be accessed during the various *access* operations.

18000-6C memory banks list follows:

0x0 - Reserved

0x1 - EPC

0x2 - TID

0x3 - USER

reg_addr = 0x0A02

Power up default = 0x00000001

Bits	**Name**	**Description**
1:0	acc_bank	Tag memory bank selection for Tag *Accesses*
3:2	acc_bank2	Tag memory bank selection for second bank read. **Used for INV_CFG tag_read = 2 only. Otherwise must be set to 0.**
31:4	reserved	reserved - read / write as 0

TAGACC_PTR

Specify the offset (16 bit words) in tag memory for tag accesses (read and write).

reg_addr = 0x0A03

Power up default = 0x00000002

For INV_CFG tag_read = 0.

Bits	**Name**	**Description**
31:0	ptr	offset to be used in tag accesses. 0x00000000 - 0xFFFFFFFF supported

For INV_CFG tag_read = 1 or 2

Bits	**Name**	**Description**
15:0	ptr	offset to be used in first bank tag accesses. 0x0000 – 0xFFFF supported
31:16	ptr2	offset to be used in second bank tag accesses. 0x0000 – 0xFFFF supported

TAGACC_CNT

Write this register to specify the number of 16 bit words that should be accessed when issuing read or write commands. Note that the value zero which is used to specify the maximum number of words contained in a bank is not supported at this time.

Reg_addr = 0x0A04

Power up default = 0x00000001

Bits	**Name**	**Description**
7:0	length	number of 16 bit words to read/write – maximum value: Read: 255, Write:32
15:8	length2	number of 16 bit words to read second bank. **Used for INV_CFG tag_read = 2 only. Otherwise must be set to 0.**
31:16	reserved	reserved - read / write as 0

TAGACC_LOCKCFG

Write this register prior to issuing the "Lock" command (0x12).

reg_addr = 0x0A05

Power up default = 0x00000000

Bits	**Name**	**Description**
9:0	action	lock command action bits
19:10	mask	lock command mask bits
31:20	reserved	reserved - write as 0

Mask defines which bank to execute the locking, action defines what type of lock or unlock commands to carry out. For details please reference EPC Air Interface document.

TAGACC_ACCPWD

Set this register to the access password that is specified in the reserved tag memory prior to issuing access commands.

reg_addr = 0x0A06

Power up default = 0x00000000

Bits	Name	Description
15:0	AWORD0	Access password value - word 0
31:16	AWORD1	Access password value - word 1

TAGACC_KILLPWD

Set this register to the desired password value prior to issuing the "Kill" command (0x13).

reg_addr = 0x0A07

Power up default = 0x00000000

Bits	Name	Description
15:0	KWORD0	Kill password value - word 0
31:16	KWORD1	Kill password value - word 1

TAGWRDAT_SEL

Used to access the tag write buffer.

The buffer is set up as a 16 register array.

reg_addr = 0x0A08

Power up default = 0x00000000

Bits	Name	Description
2:0	bank	Bank select value of TAGWRDAT. Can be 0 or 1 only. 0 for TAGWRDAT_0 - TAGWRDAT_15; 1 for TAGWRDAT_16 - TAGWRDAT_31; (same reg_addr as TAGWRDAT_0 - TAGWRDAT_15)
31:3	reserved	Reserved, write as zero

For Block Write,

Bits	Name	Description

2:0	bank	Bank select value of TAGWRDAT. Can be 0 to 7
		0 TAGWRDAT_0 to TAGWRDAT_15;
		1 TAGWRDAT_16 to TAGWRDAT_31;
		2 TAGWRDAT_32 to TAGWRDAT_47

		3 TAGWRDAT_48 to TAGWRDAT_63 4 TAGWRDAT_64 to TAGWRDAT_79 5 TAGWRDAT_80 to TAGWRDAT_95 6 TAGWRDAT_96 to TAGWRDAT_111 7 TAGWRDAT_112 to TAGWRDAT_127
31:3	reserved	Reserved, write as zero

TAGWRDAT_0 - TAGWRDAT_15

Set these registers to valid data prior to issuing the "Write" command (0x11).

reg_addr = 0x0A09 - 0x0A18

Power up default = 0x00000000

Bits	Name	Description
15:0	data_word	data to be written to during underlying *write* access.
31:16	offset	This value plus the value of TAGACC_PTR gives the word offset from start of the bank specified by TAGACC_BANK to write the data.

For Block Write,

Bits	Name	Description
15:0	Word 0	Tag Block Write Command - Low order data to be written to during underlying write access
31:16	Word 1	Tag Block Write Command - High order data to be written to during underlying write access.

CURRENT_PROFILE

Specify the selector of the active profile. The active profile may be changed by writing the selector of the desired link profile, and executing the "update the link profile" command (command 0x19).

reg_addr = 0x0B60

Power up default = 0x00000002

Bits	Name	Description
7:0	Profile selector	valid values are 0 through 3* (see Appendix C for Profile descriptions)
31:8	RFU	Reserved for future use

FREQCH_SEL

Select the frequency channel.

reg_addr = 0x0C01

Bits	Name	Description
31:0	Freq Channel	Frequency Channel selector - valid values are 0 through 49. Note that FCC scans all 50 channels; ETSI only scans channels 0 to 14.

FREQCH_CFG

Indicate which frequency channels are enabled. The frequency channel selected by writing the desired channel number to the FREQCH_SEL register, and then writing the desired bits to this register.
When a channel is disabled, the firmware will skip over it when selecting the next channel to use. Note that, for the ETSI, that the first enabled channel is the only channel used until *Scan Mode* is enabled in the PROTSCH_SMCFG register.
reg_addr = 0x0C02

Bits	Name	Description
0	status	1=channel enabled, 0=channel disabled
31:1	reserved	Reserved, write as zero

FREQCH_DESC_PLLDIVMULT

Configure the frequency for the corresponding frequency channel. The host must write HST_RFTC_FRQCH_SEL to the desired channel number and then configure the bits in this register. As the protocol scheduler iterates through enabled frequency channels, it passes the value in this register to the RF Control module so that when carrier is turned on it will be at the desired frequency.
Reg_addr = 0x0C03

Bits	Name	Description
15:0	MULTRAT	PLL multiply ratio
24:16	DIVRAT	PLL divide ratio
31:25	RFU	Reserved for future use

The relevant equations are:

CWFrequency = 24MHz * MULTRAT / (DIVRAT * 4)

AUTHENTICATE_CFG

Authenticate command configuration .
reg_addr = 0x0F00

Bits	Name	Description
0	SenRep	0: store 1: send

1	IncRepLen	0: Omit length from reply 1: Include length in reply
9:2	CSI	CSI value write as zero
21:10	Length	length of message in bits (must be multiple of 8)
31:22	reserved	reserved - write as 0

AUTHENTICATE_MSG0

Address: 0x0F01

Bit	**Name**	**Description / Values**
31:0	Message	Upper 32 bits of message

AUTHENTICATE_MSG1

Address: 0x0F02

Bit	**Name**	**Description / Values**
31:0	Message	2nd 32 bits of message

AUTHENTICATE_MSG2

Address: 0x0F03

Bit	**Name**	**Description / Values**
31:0	Message	3rd 32 bits of message

AUTHENTICATE_MSG3

Address: 0x0F04

Bit	**Name**	**Description / Values**
31:0	Message	Lower 32 bits of message

READBUFFER_PTR

ReadBuffer command starting address .

reg_addr = 0x0A03

Bits	Name	Description
31:0	prt	16-bit word starting address

READBUFFER_LEN

ReadBuffer command length .

reg_addr = 0x0A04

Bits	Name	Description
11:0	length	Number of bits to read (max 256 bits)
31:12	reserved	reserved - write as 0

UNTRACEABLE_CFG

Untraceable command configuration .

reg_addr = 0x0F05

Bits	Name	Description
1:0	Range	00: normal 01: toggle temporarily 10: reduced 11: RFU
2	User	0: view 1: hide
4:3	TID	00: hide none 01: hide some 10: hide all 11: RFU
10:5	EPC	MSB (show/hide): 0: show memory above EPC 1: hide memory above EPC 5 LSBs (length): New EPC length field (new **L** bits)
11	U	0: Deassert **U** in XPC_W1 1: Assert **U** in XPC_W1
31:12	reserved	reserved - write as 0

INV_CYCLE_DELAY

Delay time between inventory cycle.

reg_addr = 0x0F0F

Bits	Name	Description
31:0	cycle_delay	Time delay in-between each inventory cycle in ms (use to reduce tag rate) The values should be between 0 to 2000. 0 means fastest tag rate.

Reminder: When writing this register, please be careful in the packetizing process that the Least Significant Byte (LSB) should be placed first, Most Significant Byte (MSB) should be placed last. This is the so called "Reversely Populated" principle as mentioned at the beginning of this chapter.

HST_CMNDIAGS

Common diagnostics configuration register

reg_addr = 0x0201

Power Up default = 0x00000210

For Vibrator Operation, MUST CHANGE to 0x00000010 (disable Command Active Packet !!!)

Bits	Name	Description
0:0	en_diags	Enable diagnostics packets
1:1	en_status	Enable status packets
2:2	en_cycle	Enable protocol module end of cycle statistics packets only
3:3	en_round	Enable protocol module end of round statistics packets only
4:4	en_invresp	Enable sending of the inventory response packets during access commands
5:5	en_cwdiags	Enable sending of carrier diagnostics packets only
6:6	en_accessdiags	Enable sending of access diagnostic packets
7:7	en_randomcwdiags	Enable sending of random CW diagnostics packets only
8:8	en_csmdiags	Enable sending of core state machine diagnostics packets only
9:9	en_commandactive	Enable periodic output of Command Active packet for long running commands, in the absence of any other host interface output
31:10	Reserved_31_10	Reserved

AMBIENTTEMP

Ambient temperature of the reader.

reg_addr = 0x0B06

Bits	Name	Description
31:0	temperature	Ambient temperature, in units of degrees Celsius. This is a two's complement representation of a signed 32-bit integer. If the temperature is positive, i.e., MSB is 0, then the value is directly the temperature.

Reminder: Right after power up, this MAC register has value 0 if you read it. You must at least do 1 round of inventory before this register has meaningful value.

PATEMP

RF power amplifier temperature.

reg_addr = 0x0B0A

Bits	Name	Description
31:0	temperature	RF power amplifier temperature, in units of degrees Celsius. This is a two's complement representation of a signed 32-bit integer. If the temperature is positive, i.e., MSB is 0, then the value is directly the temperature.

Reminder: Right after power up, this MAC register has value 0 if you read it. You must at least do 1 round of inventory before this register has meaningful value.

FM13DT160_CMDCFGPAR

Cmd Cfg parameter

reg_addr = 0x0117

Bit	Name	Description / Values
31:0	cmd_cfg	Store the cmd cfg parameter for the FM13DT160 command

FM13DT160_REGADDRPAR

Reg addr parameter

reg_addr = 0x0118

Bit	Name	Description / Values
15:0	Reg addr	Store the reg addr parameter for the FM13DT160 command

FM13DT160_WRITEPAR

Write para parameter

reg_addr = 0x0119

Bit	Name	Description / Values
15:0	Write para	Store the write para parameter for the FM13DT160 command

FM13DT160_PWDPAR

Password parameter

reg_addr = 0x011A

Bit	Name	Description / Values
31:0	Password	Store the Password parameter for the FM13DT160 command

FM13DT160_STOBLOCKADDPAR

Store Block Address parameter

reg_addr = 0x011B

Bit	Name	Description / Values
7:0	Store Block Address	Store Block Address parameter for the FM13DT160 Get Temperature command

FM13DT160_STARTADDRPAR

Start addr parameter

reg_addr = 0x011C

Bit	Name	Description / Values
15:0	Start addr	Start addr parameter for the FM13DT160 Read/Write Memory command

FM13DT160_READWRITELENPAR

Read/Write len parameter

reg_addr = 0x011D

Bit	Name	Description / Values
15:0	Len	Read/Write length parameter for the FM13DT160 Read/Write Memory command (Read: Max 512; Write: Max 4)

FM13DT160_DATAPAR

Data parameter

reg_addr = 0x011E

Bit	Name	Description / Values
31:0	Data	Data parameter for the FM13DT160 Write Memory command

HST_CMD (Downlink)

HST_CMD is host command register. Host software writes one of the supported command values to this register and causes the reader firmware to execute the new command. Once you write the values into this register, the reader will execute the command. This is a "register write to enable action" methodology to send a command to another IC.

reg_addr = 0xF000

Bits	Name	Description
31:0	Command	Command value

Command Set Summary

When the host wishes to issue a command to the reader firmware, it writes a *command value* to the HST_CMD register. The host is then required to successively read *Response Packets.* First it should receive a *Command begin Packet*. Then it should receive *Response Packets*. At the end it should receive a *Command End Packet*. The *Command End Packet* signifies completion of the command and, if the reader firmware is not in an error state, it is then ready to execute more commands. If there is an error, the command end packet will contain that and the host can analyze it and check against the MAC error table in Appendix B.

Command Value (for HST_CMD register 0xF000)	**Command Mnemonic**	**Description**
0x00000002	CMD_WROEM	Write OEM register (set OEM_ADDR and OEM_DATA before you do this)
0x00000003	CMD_RDOEM	Read OEM register (set OEM_ADDR before you do this)
0x00000005	CMD_MBPRDREG	MAC Bypass Register Read - directly read R2000 register (set MAC_BYPASS_ADDR before you do this)
0x00000006	CMD_MBPWRREG	MAC Bypass Register Write - directly write R2000 register (set MAC_BYPASS_ADDR and MAC_BYPASS_DATA before you do this)
0x0000000F	CMD_18K6CINV	Start Inventory
0x00000010	CMD_18K6CREAD	Read
0x00000011	CMD_18K6CWRITE	Write

0x00000012	CMD_18K6CLOCK	Lock
0x00000013	CMD_18K6CKILL	Kill
0x00000014	CMD_SETPWRMGMTCFG	Set Power Management Configuration
0x00000019	CMD_UPDATELINKPROFILE	Change the link profile
0x0000001F	CMD_18K6CBLOCKWRITE	BlockWrite
0x00000026	CUSTOMNXPCHANGEEAS	ChangeEAS for G2XM and G2XL IC of NXP
0x00000050	CMD_18K6AUTHENTICATE	Authenticate
0x00000051	CMD_18K6READBUFFER	ReadBuffer
0x00000052	CMD_18K6UNTRACEABLE	Untraceable
0x00000053	CUSTOMMFM13DTREADMEMORY	FM13DT160 Read Memory
0x00000054	CUSTOMMFM13DTWRITEMEMORY	FM13DT160 Write Memory
0x00000055	CUSTOMMFM13DTAUTH	FM13DT160 Auth
0x00000056	CUSTOMMFM13DTGETTEMP	FM13DT160 Get Temperature
0x00000057	CUSTOMMFM13DTSTARTLOG	FM13DT160 Start Logging
0x00000058	CUSTOMMFM13DTSTOPLOG	FM13DT160 Stop Logging
0x00000059	CUSTOMMFM13DTWRITEREG	FM13DT160 Write Reg
0x0000005A	CUSTOMMFM13DTREADREG	FM13DT160 Read Reg
0x0000005B	CUSTOMMFM13DTDEEPSLEEP	FM13DT160 Deep Sleep
0x0000005C	CUSTOMMFM13DTOPMODECHK	FM13DT160 Op Mode Check
0x0000005D	CUSTOMMFM13DTINITIALREGFILE	FM13DT160 Initial Regfile
0x0000005E	CUSTOMMFM13DTLEDCTRL	FM13DT160 Led Ctrl

Note that whenever an HST_CMD is sent using MAC Register 0xF0000, then the reader will execute the requested actions, and send back first a Command Begin packet, then certain number of operation-response packets, and then ending with a Command End packet.

OEM_ADDR

Sets this address register to indicate which address in the OEM table should be accessed with a subsequent read OEM command.   You must set this value BEFORE you execute HST_CMD 0x02 or 0x03

Address: 0x0500

Bits	Name	Description
15:0	OEM_ADDR	The address (or index) of the OEM data to be accessed
31:16	Reserved	Reserved

OEM_DATA

Use this register to save the actual data you want to write to a certain OEM address as defined by OEM_ADDR above.   You must set this value BEFORE you execute HST_CMD 0x02

Address: 0x0501

Bits	Name	Description
31:0	OEM_DATA	The OEM data to be written

MAC_BYPASS_ADDR

Sets this address register to indicate which address in the OEM table should be accessed with a subsequent read OEM command.   You must set this value BEFORE you execute HST_CMD 0x02 or 0x03

Address: 0x0400

Bits	Name	Description
11:0	R2000_ADDR	The address (or index) of the R2000 Register to be accessed
31:12	Reserved	Reserved

MAC_BYPASS_DATA

Use this register to save the actual data you want to write to a certain OEM address as defined by OEM_ADDR above.   You must set this value BEFORE you execute HST_CMD 0x02

Address: 0x0401

Bits	Name	Description
31:0	OEM_DATA	The OEM data to be written

A.5 R2000 Registers Description

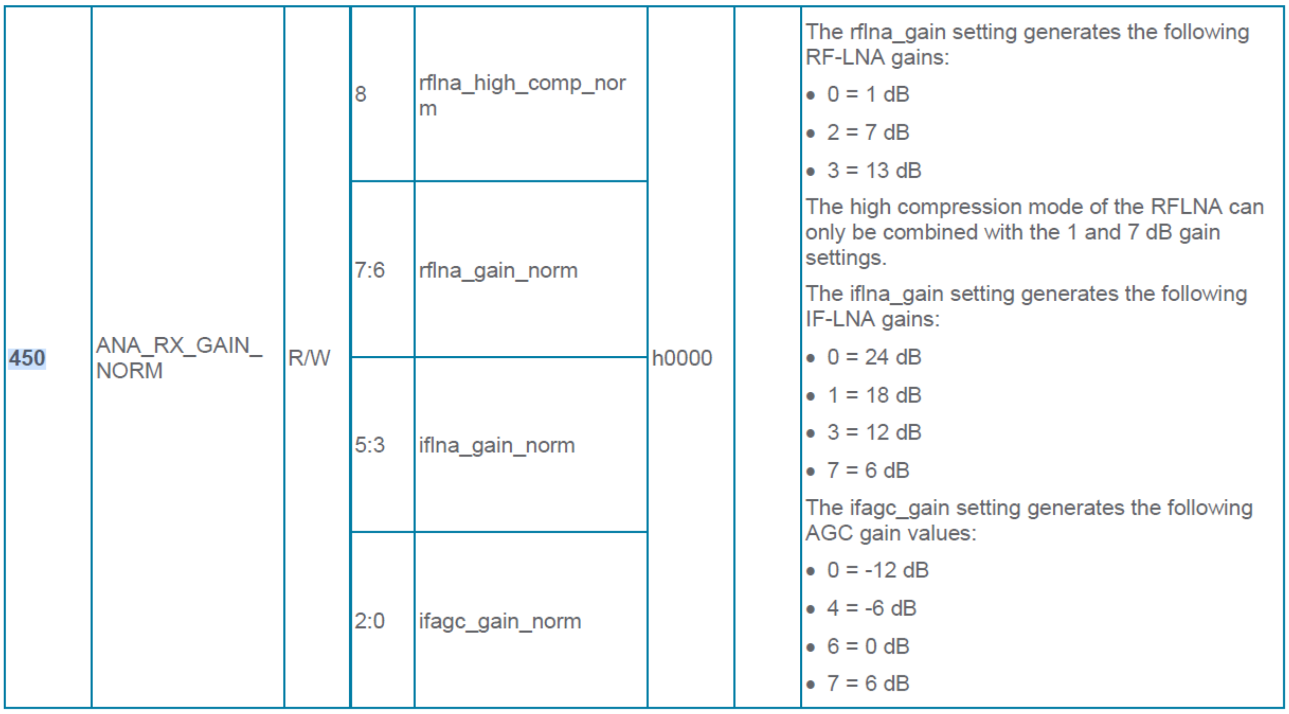
This R2000 register contains RF High Compression Mode setting, RF LNA Gain, IF LNA Gain, AGC Gain Range for application configure. Use **MAC Bypass Write and Read HST_CMD (i.e., 0x05 and 0x06)** to access this R2000 register.

ANA_RX_GAIN_NORM

Configures RF LNA Gain, RF LNA High Compression Mode ON/OFF, IF LNA Gain

IF AGC Gain

Address: 0x0450



450	ANA_RX_GAIN_NORM	R/W	8	rflna_high_comp_norm	h0000		The rflna_gain setting generates the following RF-LNA gains: • 0 = 1 dB • 2 = 7 dB • 3 = 13 dB The high compression mode of the RFLNA can only be combined with the 1 and 7 dB gain settings. The iflna_gain setting generates the following IF-LNA gains: • 0 = 24 dB • 1 = 18 dB • 3 = 12 dB • 7 = 6 dB The ifagc_gain setting generates the following AGC gain values: • 0 = -12 dB • 4 = -6 dB • 6 = 0 dB • 7 = 6 dB
			7:6	rflna_gain_norm			
			5:3	iflna_gain_norm			
			2:0	ifagc_gain_norm			

A.6 OEM Registers Description

HIGHLOWLEVEL

Holds the high low level configuration of the reader

reg_addr = 0x000000A2

Bits	**Name**	**Description**
31:0	High Low Level	0 is high level; 1 is low level

A.7 Command State Packet (Uplink)

Command state packets come back from RFID reader module to the host processor to report important information.

Command-Begin Packet

The command-begin packet indicates the start of a sequence of packets for an ISO 18000-6C tag-protocol operation (i.e. inventory etc.). The type of command executed by the RFID radio module determines which data packets appear, and in what order they appear, between the command-begin/end packet pair.

Table: Command-Begin Packet Fields

Byte Offset(s)	**Name**	**Description**
0	**pkt_ver**	**0x02 for low level** **(or 0x01 for high level)**
1	flags	0 = Non-continuous mode 1 = Continuous mode
3:2	pkt_type	0x8000 for low level (or 0x0000 for high level)
5:4	pkt_len	Length = 0x0002
7:6	reserved	Reserved. Read as zero.
11:8	command	The command that initiated the packet sequence: 0x0000000F - Start Inventory
15:12	ms_ctr	Firmware millisecond counter when the operation started.

Command-End Packet

The command-end packet indicates the end of sequence of packets for an ISO 18000-6C tag-protocol operation. A command-end packet is always used to terminate a packet sequence regardless of the fact that a tag-access operation is completed successfully or not. If not successful, a possible reason is a MAC error has happened. The MAC error type is stored in Byte 12 and 13 (in reverse order) of the Command End Packet Fields.

Table: Command-End Packet Fields

Byte Offset(s)	**Name**	**Description**
0	**pkt_ver**	**0x02 for low level** **(0x01 for high level)**
1	flags	Reserved. Read as zero.
3:2	pkt_type	0x8001 for low level (or 0x0001 for high level)
5:4	pkt_len	Length = 0x0002
7:6	reserved	Reserved. Read as zero.
11:8	ms_ctr	Firmware millisecond counter when the operation ended.
13:12	status	The completion status of the operation. Values are: 0x0000 – Success. See Appendix B for RFID Firmware Error Codes table.
14	error_port	If status is non-zero, this reports the error port. Otherwise read as zero.
15	reserved	Reserved. Read as zero.

Inventory-Response Packet

The ISO 18000-6C inventory-response packet contains the data a tag backscatters during the tag-singulation phase. This data is generated for tag inventories as well as ISO 18000-6C tag-access operations (i.e. read, write, etc.). Assuming a valid CRC, the data contains the PC+EPC+CRC16 received during the singulation of a tag.

Table: Inventory-Response Packet Fields

Byte Offset(s)	Name	Description
0	**pkt_ver**	**Currently = 0x03 (for both high level and low level)** **(note that this is different from R1000 based readers such as CS469, CS203, CS468INT)**
1	flags	Flags
3:2	pkt_type	0x8005 for low level (or 0x0005 for high level)
5:4	pkt_len	Length = variable (greater than or equal to 3)
7:6	reserved	Reserved. Read as zero.
11:8	ms_ctr	Firmware millisecond counter tag was inventoried.

Bits	**Name**	**Description**
0:0	CRC	0 = Valid CRC 1 = Invalid CRC
1:1	ANA_CTL Format	0 = R1000 Format 1 = R2000 Format
3:2	FastID Data	0 = No FastID data in packet 1 = Monza 4 FastID data included (12 bytes) 2-3 = Reserved
4:4	Phase Data	0 = No Phase Data (0) 1 = Phase Data available
5:5	Reserved	Reserved
7:6	Pad Bytes	Number of padding bytes to Inv_Data to force 32-bit packet boundary. Padding bytes are always at the end of the packet.

12	wb_rssi	Wideband Receive Signal Strength Indicator.
		Value Conversion to dB:
		Mantissa = 3:0
		Exponent = 7:4
		Mantissa_Size = 4
		20 * log10 (2^Exponent * (1 + Mantissa / 2^Mantissa_Size))
		Example: Value 0x48
		Mantissa = 8
		Exponent = 4
		20 * log (2^4 * (1 + 8 / 2^4)) = 27.60

13	nb_rssi	Narrowband Receive Signal Strength Indicator.
		Value Conversion to dB:
		Mantissa = 2:0
		Exponent = 7:3
		Mantissa_Size = 3
		20 * log10 (2^Exponent * (1 + Mantissa / 2^Mantissa_Size))
		Example: Value 0x48
		Mantissa = 0
		Exponent = 9
		20 * log (2^9 * (1 + 0 / 2^3)) = 54.19

14	phase	The Phase Data Bits 6:0 represents signed phase value at the time the EPC is received. Bit 7 is reserved and always 0. Phase is not available on R1000. For R2000, only bit 0 to bit 5 are used. Bit 6 is discarded. Phase is only available for profiles where the following is true: 8 * 2^Miller-Number / BLF >= 20.0 us. (BLF is backscatter Link frequency) Phase in Radian = <bit 0 to bit 5> x 2 x Pie / 128 Phase in Degree = Phase in Radian x 180 / Pie
15	chidx	Current channel index.
16	data1_count	DATA1 16 bit word length
17	data2_count	DATA2 16 bit word length
19:18	port	Current antenna port.
n:20	inv_data	The data that was backscattered by the tag (i.e. PC + EPC + CRC16) during tag singulation. The data is presented in the same format as it is transmitted over the air from the tag to the RFID radio module – i.e. the data has not been changed to match the endianness of the host

		processor. The total length of this field (in bytes) can be determined by the following formula: ((pkt_len – 3) * 4) – ((flags >> 6) & 3) If tag_read in INV_CFG is 1, the backscattered data is in format PC+EPC+DATA1+CRC16. EPC length is ((PC >> 11) * 2) If tag_read in INV_CFG is 2, the backscattered data is in format PC+EPC+DATA1+DATA2+CRC16. EPC length is ((PC >> 11) * 2)

Inventory-Response Packet (Compact mode)

This is Inventory-Response packet when compact mode is set in INV_CFG. Note the important difference marked in bold red.

Table: Inventory-Response Packet (Compact mode) Fields

Byte Offset(s)	Name	Description
0	**pkt_ver**	**Currently = 0x04 (for both high level and low level)**
1	flags	Flags <table><tr><th>Bit(s)</th><th>Description</th></tr><tr><td>0</td><td>0 = no CRC error 1 = CRC error</td></tr><tr><td>7:1</td><td>Reserved. Read as zero.</td></tr></table>
3:2	pkt_type	0x8005 for low level (or 0x0005 for high level)
5:4	pkt_len	Payload length in **byte (!!!!!!!)**
6	Antenna Port #	Port, indexing from 0 to 15. For CS463, 0 is Port 1, 1 is Port 2, 2 is Port 3, 3 is Port 4. All the payloads in n:8 below are for this antenna port only.
7	Reserved	Reserved. Read as zero
n:8	payload	The payload data includes one or multiple tag data. Each tag data is in format: PC(2 bytes)+EPC(EPC length)+NB_RSSI(1 byte). EPC length is ((PC >> 11) * 2). Narrowband Receive Signal Strength Indicator (NB_RSSI). Value Conversion to dB: Mantissa = 2:0 Exponent = 7:3 Mantissa_Size = 3 20 * log10 (2^Exponent * (1 + Mantissa / 2^Mantissa_Size)) Example: Value 0x48 Mantissa = 0 Exponent = 9 20 * log (2^9 * (1 + 0 / 2^3)) = 54.19

Tag-Access Packet

The ISO 18000-6C tag-access packet indicates the result of the tag-access command upon the ISO 18000-6C tag. Valid tag access commands are as follows:

- Read
- Write
- Kill
- Lock

If a tag operation is simply limited to an inventory operation, ISO 18000-6C tag operation packets will not appear in the packet sequence.

Table: Tag-Access Packet Fields

Byte Offset(s)	Name	Description
0	**pkt_ver**	**Currently = 0x01 (for both high level and low level)**
1	flags	Flags

Bit(s)	Description
0	Error flag: 0 = Access operation succeeded 1 = An error occurred. If one of the following error-specific bit fields does not indicate an error, the error code appears in the data field.
1	Tag backscatter error flag: 0 = Tag did not backscatter an error. 1 = Tag backscattered an error. See error_code field.
2	ACK timeout flag: 0 = Tag responded within timeout. 1 = Tag failed to respond within timeout.
3	CRC invalid flag: 0 = CRC was valid 1 = CRC was invalid
5:4	Reserved. Read as zero.
7:6	Number of padding bytes added to end of tag access data to force packet to end on 32-bit boundary.

Byte Offset(s)	Name	Description
3:2	pkt_type	Packet type value = 0x0006 **(for both high level and low level)**
5:4	pkt_len	Length = variable (greater than or equal to 3)
7:6	reserved	Reserved. Read as zero.

11:8	ms_ctr	Firmware millisecond counter tag was inventoried.
12	command	ISO 18000-6C access command: 0xC2 – Read 0xC3 – Write 0xC4 – Kill 0xC5 – Lock 0xC7 – Block Write 0x04 – EAS
13	error_code	If the tag backscattered an error (i.e. the tag backscatter error flag is set), this value is the error code that the tag backscattered. Values are: 0x00 – general error (catch-all for errors not covered by codes) 0x03 – specified memory location does not exist of the PC value is not supported by the tag 0x04 – specified memory location is locked and/or permalocked and is not writeable 0x0B – tag has insufficient power to perform the memory write 0x0F – tag does not support error-specific codes
15:14	port	Current antenna port.
19:16	reserved	Reserved. Read as zero.
n:20	data	If there were no errors, and the ISO 18000-6C tag-access operation was a read (i.e. the command field contains 0xC2), this field contains the data that was read from the specified tag memory bank. The data should be treated as a sequence of bytes and is presented in the same format as it is transmitted over the air from the tag to the radio module – i.e. the data has not been changed to match the endianness of the host Firmware processor. The length of this field can be determined by the following formula: ((pkt_len – 3) * 4) – ((flags >> 6) & 3) If there were no errors, and the ISO 18000-6C tag-access operation was anything besides a read, then this field will not be present. If the error flag is set, and none of the error-specific bits are set in the flags field, this field contains a 32-bit error code. The error code can be one of the following: 0x0000 = No error 0x0001 = Handle mismatch 0x0002 = CRC error on tag response 0x0003 = No tag reply

		0x0004 = Invalid password 0x0005 = Zero kill password 0x0006 = Tag lost 0x0007 = Command format error 0x0008 = Read count invalid 0x0009 = Out of retries 0xFFFF = Operation failed

Antenna-Cycle-End Packet

The antenna-cycle-end packet indicates the end of one iteration through all enabled antennas.

Table: Antenna-Cycle-End Packet Fields

Byte Offset(s)	Name	Description
0	**pkt_ver**	**0x02 for low level (or 0x01 for high level)**
1	flags	Reserved. Read as zero.
3:2	pkt_type	0x8007 for low level (or 0x0007 for high level)
5:4	pkt_len	Length = 0x0000
7:6	reserved	Reserved. Read as zero.

Inventory-Cycle-Begin Packet

The inventory-cycle-begin packet **is normally not sent back**, but only if MAC Error 0x0336 happens. MAC Error 0x0336 means the requested forward power level is not achieved during power ramp. The MAC Error 0x0336 code is in Byte 7:6

Table: Inventory-Cycle-Begin Packet Fields

Byte Offset(s)	Name	Description
0	**pkt_ver**	**0x01 for high or low level**
1	flags	Reserved. Read as zero.
3:2	pkt_type	0x000A for high level or low level
5:4	pkt_len	Length = 0x0000
7:6	Error Code	MAC Error 0x0336
11:8	MS_Ctr	Firmware millisecond counter when antenna usage stopped

OEM Registers Read Response Packet

The OEM Register Read Response packet indicates the OEM values in an OEM address, which comes back after an HST_CMD OEM Read operation.

Table: OEM Register Read Response Packet Fields

Byte Offset(s)	Name	Description
0	**pkt_ver**	**0x01 for both low level and high level**
1	flags	Reserved. Read as zero.
3:2	pkt_type	Packet type value = 0x3007
5:4	pkt_len	Length = 0x0002
7:6	reserved	Currently = 0xFF
11:8	address	Address of requested OEM configuration register
15:12	Data	Data in requested OEM configuration register

OEM Register Write Response Packet

No response after OEM register write command other than the command begin and command end response.

Therefore application must execute an OEM Register Read command to confirm the write.

Since OEM registers are flash memory, a good practice is to wait 100 msec after sending the write OEM register command, and then do a read OEM register.

R2000 Register Read Response Packet

The R2000 Register Read Response packet indicates the values in an R2000 register address, which comes back after an HST_CMD MAC Bypass Read operation.

Table: R2000 Register Read Response Packet Fields

Byte Offset(s)	**Name**	**Description**
0	**pkt_ver**	**0x01 for both low level and high level**
1	flags	Reserved. Read as zero.
3:2	pkt_type	Packet type value = 0x3005
5:4	pkt_len	Length = 0x0001
7:6	reserved	Currently = 0xFF
9:8	address	Address of requested R2000 configuration register
11:10	Data	Data in requested R2000 register

R2000 Registers Write Response Packet

No response after R2000 register write command other than the command begin and command end response.

Therefore application must execute an R2000 Register Read command to confirm the write.

A.8 Control Command (Downlink & Uplink)

There are 2 control commands that are the same for high level or low level API:

Abort (Downlink)

Abort Response (Uplink)

Byte 0 = pkt_ver = 40   (for both low level and high level)

Byte 1 = 03

Packet Name	**Direction**	**Command Data**	**Description**
ABORT control command	**Downlink**	**40**:03:00:00:00:00:00:00	Control command to abort operation. Explanation: 4003000000000000 means abort operation.
ABORT control command response	**Uplink**	**40**:03:bf:fc:bf:fc:bf:fc	Command response of abort operation. 4003bffcbffcbffc means abort success

Appendix B – Firmware Error Codes

The "status" field of the Command End Packet comes back with error code field. If this field is non-zero then there are errors. Again, this is reversed, so byte 13 is the high byte, byte 12 is the low byte.

Code (hex number)	Description
0x0102	Set by the USB interface module when an unsupported descriptor *TYPE* has been requested by the host (i.e. not a device, string, configuration descriptor type. This may be due to compatibility problems with the USB host.
0x0103	Set by the USB interface module when an unsupported device descriptor index has been requested by the Host.
0x0104	Set by the USB interface module when it is unable to transmit the response to a request on USB endpoint 0 (aka control endpoint). This may be due to compatibility or synchronization problems with the USB host.
0x0105	RESERVED
0x0106	Set by the USB interface module when higher level firmware requests an unsupported buffer length. This may be due to a firmware build error or corrupted firmware in flash.
0x0107	This is set by the Host interface module when the underlying physical interface module returns an unknown error code on receive from the host. This may be due to a firmware build issue, corrupted firmware image or corrupted SRAM due to errant Intel® R1000 Firmware code.
0x0108	This is set by the Host interface module when the underlying physical interface module returns an unknown error code on transmit to the Host. This may be due to a firmware build issue, corrupted firmware image or corrupted SRAM due to errant code.
0x0109	This is set when the Host interface code detects that its internal state machine out of sync. This could be due to a corrupted firmware image or corrupted SRAM due to errant Intel® R1000 Firmware code.
0x010A	RESERVED
0x010B	Set by the host interface module when an invalid Intel® R1000 Firmware register read or write is attempted (either by the host or internally by the Intel® R1000 Firmware).

0x010C	RESERVED
0x010D	This is set by the host interface module during initialization if it is unable to retrieve USB string descriptors from non-volatile memory (i.e. flash) OEM configuration area. This may be due to a corrupt or unformatted OEM Configuration area. It may also be due to a firmware build issue if the OEM configuration definition is out of sync with the Intel® R1000 Firmware code.
0x010E	This is set when the host attempts to *write* a value to a selector type register that is out of range for that selector.
0x010F	Some firmware tried to send a packet to the host that was too long for the underlying host interface code.
0x0110	Not currently set by Intel® R1000 Firmware.
0x0111	Set by the low level host interface logic if an upper level requests an unsupported raw mode. This may occur if the system is corrupted.
0x0112	Set by the low level host interface logic if a system corrupt occurs and the link manager can not determine the current link state.
0x0113	Set by the low level host interface logic if an unknown / unsupported control command is received from the host. This may occur if the host logic and the Intel® R1000 Firmware. logic are out of sync. in terms of the lowest level host interface (UART, USB)
0x0114	This is set if the upper layer host logic attempts to receive data and the lower layer cannot support the buffer size requested. This will happen if the system is corrupted.
0x0115	Set by the low level host interface logic if a control command is received from the host while in raw mode - which is not allowed. This would happen if the host caused the the Intel® R1000 Firmware to enter non-volatile memory update mode, which uses raw mode, and then the host proceeded to issue control commands.
0x0116	Set by the host interface module at boot time if the OEM configuration area is specifying an unsupported host interface.
0x0300	This is set during the PLL lock logic when a bounds check fails while checking the frequency channel configuration registers.
0x0301	This is set if an unsupported frequency hopping mode is detected - during the PLL lock logic.
0x0302	This is set if the PLL fails to lock.
0x0303	This is set when the RFTC module's AUX ADC function times out waiting for an ADC conversion.
0x0304	This is set when the RFTC module times out waiting for Intel® UHF RFID Transceiver R1000 to indicate RX or TX filter tuning is complete.
0x0305	This is set when the RFTC module detects ambient temperature sensor indicates too

	hot.
0x0306	This is set when the RFTC module detects transceiver temperature sensor indicates too hot.
0x0307	This is set when the RFTC module detects PA temperature sensor indicates too hot.
0x0308	This is set when the RFTC module detects that the delta between the PA temperature and the ambient temperature is too great.
0x0309	This is set when the reverse power level is too high as measured by the configured reverse power level threshold in the register set.
0x030A	This is set when an incorrect current gain setting is passed into the IFLNA gain adjustment logic. May indicate corrupted code.
0x030B	Returned by RFTC code when errors occur in transmitting a bit over the RF interface
0x030C	Returned by RFTC code when errors occur in transmitting a buffer of bytes over the RF interface
0x030D	Returned by RFTC code when errors occur in transmitting an “end of transfer” command over the RF interface
0x030E	Returned by RFTC code when errors occur in transmitting a “preamble” command over the RF interface
0x030F	Returned by RFTC code when errors occur in transmitting a “frame-sync” command over the RF interface
0x0310	Indicates that the RF Transceiver (Intel® UHF RFID Transceiver R1000) failed to set expected ISR bits in a timely fashion. Indicates a failure in either the RFTC state machine logic or in the RF Transceiver state machine logic.
0x0311	This is set when invalid link parameters are detected when the filter tuning logic is run.
0x0312	This indicates a failure in either the RFTC state machine logic or in the RF Transceiver state machine logic. This error can only occur if the RF Transceiver (Intel® UHF RFID Transceiver R1000) starts filling its RX FIFO with received data, but fails return the requested number of bits in a timely fashion.
0x0313	Not currently in use. May occur in the future when switching between link profiles if some of the required information is not properly coded in the Intel® R1000 Firmware.
0x0314	May occur when switching between link profiles if the set of Intel® UHF RFID Transceiver R1000 registers that correspond to the requested link profile’s ID_High/ID_Low cannot be found. For the time being, since profiles are compiled into the code, this would be an indication of a corrupted data segment or a build-time fault.
0x0315	Internal error. The error is the direct result of the Intel® R1000 Firmware having to do a “dBm to linear” conversion on a dBm measurement that is outside the range of -99dBm through +45dBm. It the unlikely event that this error is encountered, it is probably the result of a faulty RF Peak Detector, a bug in the code that computes the dBm value from the RF Peak Detector ADC reading, or a faulty external PA circuit.
0x0316	If, during RF power-ramping, it is determined that the RF power at the antenna port

	has momentarily exceeded 35dBm, or has exceeded 33dBm steady-state, this error will be thrown. Encountering this error is often the result of an incorrect calibration of the “gross gains”. See Intel® R1000 Firmware command 0x1B for more information on how to calibrate the system.
0x0317	Internal error that may occur if memory is corrupted.
0x0318	Indicates that the target power (in Intel® R1000 Firmware Virtual Register 0x706) is higher than the maximum allowed output power, which is +33dBm.
0x0319	Indicates that the specified ADC reference voltage is outside the allowed range.
0x031A	Indicates that the measured value of the antenna-sense resistor (reported in the Intel® R1000 Firmware Virtual Register 0x703) exceeds the threshold specified (specified in the Intel® R1000 Firmware Virtual register 0xB12). To determine which antenna was disconnected, the list of enabled antennas will need to be scanned for the one exceeding the threshold (this is done by iterating through all valid selectors in register 0x701 and examining the MAC_ANT_DESC_STAT register at address 0x703.
0x031B	Indicates that the OEMCFG’s HW_OPTIONS_FORMAT value is not recognized by the RFTC subsystem
0x031C	Indicates that the forward power detection option found in OEMCFG’s HW_OPTIONS0 field is not recognized by the RFTC subsystem
0x031D	Indicates that the reverse power detection option found in OEMCFG’s HW_OPTIONS0 field is not recognized by the RFTC subsystem
0x031E	Indicates that the DRM Filter option found in OEMCFG’s HW_OPTIONS0 field is not recognized by the RFTC subsystem
0x031F	Indicates that ambient temperature sensor option found in OEMCFG’s HW_OPTIONS0 field is not recognized by the RFTC subsystem
0x0320	Indicates that PA temperature sensor option found in OEMCFG’s HW_OPTIONS0 field is not recognized by the RFTC subsystem
0x0321	Indicates that transceiver temperature sensor option found in OEMCFG’s HW_OPTIONS0 field is not recognized by the RFTC subsystem
0x0322	Indicates that antenna-sense resistor sensor option found in OEMCFG’s HW_OPTIONS0 field is not recognized by the RFTC subsystem
0x0323	The range specified for the IF LNA AGC gain limits is bad. Either the “min” is higher than the “max”, or the min or max setting is incorrect.
0x0324	When invoking the CMD_LPROF_RDXCVRREG or CMD_LPROF_WRXCVRREG commands, one of the arguments is the selector of a valid link profile (in the Intel® R1000 Firmware release 1.0 and 1.1, profile selectors 0 through 5 are valid). New link profile selectors cannot be created through these commands, so if a selector outside this range is passed, the RFTC_ERR_LPROFBADSELECTOR error will be generated.
0x0325	The R1000 transceiver has about 421 register addresses between 0x0000 and

	0x0500. One of the arguments to the CMD_LPROF_RDXCVRREG or CMD_LPROF_WRXCVRREG commands is the transceiver register address to configure. If the address passed is not a valid R1000 transceiver address, this error will be thrown.
0x0326	Not all valid R1000 transceiver addresses may be configured through the link profiles. The excluded addresses include those registers which are read-only (refer to the R1000 register map), and the indirect address for the R2T command register: 0x0105.
0x0327	Set by the RFTC module if an unsupported RFLNA gain level is requested. This will happen if the protocol scheduler's cycle granular configuration is enabled and the user has specified an unsupported gain level in the HST_PROTSCH_CYCCFG_DESC_ADJ1 banked register
0x0328	Set by the RFTC module if an unsupported IFLNA gain level is requested. This will happen if the protocol scheduler's cycle granular configuration is enabled and the user has specified an unsupported gain level in the HST_PROTSCH_CYCCFG_DESC_ADJ1 banked register
0x0329	Set by the RFTC module if an unsupported AGC/MIXER gain level is requested. This will happen if the protocol scheduler's cycle granular configuration is enabled and the user has specified an unsupported gain level in the HST_PROTSCH_CYCCFG_DESC_ADJ1 banked register
0x032A	Set by the RFTC module if an unsupported compensation option is detected at OEMCFG address 0xA1.
0x0336	This error is not in Command End packet, but in Inventory-Cycle-Begin packet. This error will not cause inventory to stop. This error is generated when the requested forward power level is not achieved during power ramp.

Appendix C – RFID Reader Firmware Command Sequence Examples

C.1 RFID Reader Initialization Example

There are 4 steps within Reader Initialization that MUST be done:

Step 1: Open Radio
Step 2: Set RF Power
Step 3: Set Channel
Step 4: Set Profile

Radio Open

This command ensures the Radio is alive.

Packet Name	**Direction**	**Command Data**	**Description**
ABORT control command	Downlink	40:03:00:00:00:00:00:00	Control command to abort operation. Explanation: 4003
ABORT control command return	Uplink	40:03:bf:fc:bf:fc:bf:fc	Command response of abort operation.

Set RF Power

Packet Name	Direction	Command Data	Description
REG_REQ*	Downlink	70:01:01:07:00:00:00:00	ANT_PORT_SEL (See appendix A*)
REG_REQ	Downlink	70:01:06:07:2c:01:00:00	ANT_PORT_POWER (See appendix A)

*REG_REQ means register requests, which can either be read or write.

**Appendix A describes how to read and write registers and the defined content within each register. For example, ANT_PORT_SEL is a register. You can read from it to see what the current setting is. You can write to it to change it to whatever setting you want. The registers contain the configuration for the RFID reader to follow in operation.

Set Channel

Set channel means setting the frequency channels. Note that this is only needed for readers where the frequency set is NOT locked. If the version has the frequency channels locked, then there is no need to do this Set Channel operation. For example, if your reader is CS108-2, that means it is FCC and the frequency channels are locked, complying with USA government requirements. Then you do not need to set channel. Please refer to Appendix K for reader model versions versus frequency channel set.

The following steps are followed:

Step 1: Disable all available channels
Step 2: Enable only the desired channel.

Example of disabling channel 0.

Packet Name	Direction	Command Data	Description
REG_REQ	Downlink	70:01:01:0c:00:00:00:00	FREQCH_SEL (see appendix A)
REG_REQ	Downlink	70:01:02:0c:00:00:00:00	FREQCH_CFG (see appendix A)

Example of enable channel 1.

Packet Name	Direction	Command Data	Description
REG_REQ	Downlink	70:01:01:0c:01:00:00:00	FREQCH_SEL (see appendix A)
REG_REQ	Downlink	70:01:02:0c:01:00:00:00	FREQCH_CFG (see appendix A)

Set Profile

Packet Name	Direction	Command Data	Description
REG_REQ	Downlink	70:01:60:0b:01:00:00:00	CURRENT_PROFILE (see appendix A) Explanation: 7001 means write to register. Address is 0B60 which is the address that stores the current profile. Profile is to be set to 00000001, which is Profile 1 for CS108.
REG_REQ	Downlink	70:01:00:f0:19:00:00:00	HST_CMD (see appendix A) Explanation: 7001 means write to register. F000 is the HST_CMD address (note the byte reversal). Command is 00000019 (always reversing bytes) which means change profile. So the action is change profile.
Command-Begin Packet	Uplink	02:00:00:80:02:00:00:00:19:00:00:00:13:41:00:00	See appendix A. This is the packet that would come back from RFID module. It signals RFID module is starting to do the command above (i.e. change profile).
Command-End Packet	Uplink	02:00:01:80:02:00:00:00:16:41:00:00:00:00:00:00	See appendix A. This is the packet that would come back from RFID module. It signals it has completed the command of changing profile.

C.2 Tag Operations: Inventory Example

There are 3 steps within Inventory that MUST be done:

Step 1: Set Inventory Parameters

Step 2: Set Inventory Algorithm

Step 3: Start/Stop Inventory

Set Inventory Parameters

Packet Name	Directions	Command Data	Description
REG_REQ	Downlink	70:01:00:07:ff:ff:00:00	ANT_CYCLES (See appendix A)
REG_REQ	Downlink	70:01:00:09:00:00:00:00	QUERY_CFG (See appendix A)

Set Inventory Algorithm

REG_REQ	Downlink	70:01:02:09:01:00:00:00	INV_SEL (See appendix A)
REG_REQ	Downlink	70:01:03:09:f7:00:50:03	INV_ALG_PARM_0 (See appendix A)
REG_REQ	Downlink	70:01:01:09:01:00:00:00	INV_CFG (See appendix A)

Start Inventory

REG_REQ	Downlink	70:01:00:f0:0f:00:00:00	HST_CMD (See appendix A)
Command-Begin Packet	Downlink	02:01:00:80:02:00:00:00:0f:00:00:00:61:44:00:00	See appendix A.
Inventory-Response PacketS	Uplink	02:00:05:80:07:00:00:00:73:44:00:00:81:5f:83:06:00:00:00:00:30:00:10:00:00:00:00:00:00:00:00:00:00:06:87:71:34	See appendix A. (This is the tag info during inventory. The reader will keep on sending tag data inventoried back to the host processor until you stop inventory)

Stop Inventory

ABORT control command	Downlink	40:03:00:00:00:00:00:00	This is the command you use to stop inventory.
ABORT control command return	Uplink	40:03:bf:fc:bf:fc:bf:fc	This is the command response you will receive for stopping inventory.

Note: After the "ABORT" command to stop inventory, a 2 seconds delay is required for the reader to clear buffer before it can execute another command

C.3 Tag Operations: Read Example

There are 4 steps within Read that MUST be done:

Step 1: Set Inventory Parameters

Step 2: Set Inventory Algorithm. (Assume only 1 tag in front of reader)

Step 3: Select the desired tag.

Step 4: Start reading the tag data.

The followings example illustrates how to read the TID bank.

Step 1: Set Inventory Parameters

Packet Name	Command Data	Description
REG_REQ	70:01:00:07:01:00:00:00	ANT_CYCLES (See appendix A)
REG_REQ	70:01:00:09:80:01:00:00	QUERY_CFG (See appendix A)

Step 2: Set Inventory Algorithm (Fixed Q = 0)

REG_REQ	70:01:02:09:00:00:00:00	INV_SEL (See appendix A)
REG_REQ	70:01:03:09:00:00:00:00	INV_ALG_PARM_0 (See appendix A)
REG_REQ	70:01:05:09:00:00:00:00	INV_ALG_PARM_2 (See appendix A)

Step 3: Select the desired tag

(e.g.111122223333444455556666)

REG_REQ	70:01:01:08:09:00:00:00	TAGMSK_DESC_CFG (See appendix A 1)
REG_REQ	70:01:02:08:01:00:00:00	TAGMSK_BANK (See appendix A)
REG_REQ	70:01:03:08:20:00:00:00	TAGMSK_ PTR (See appendix A)
REG_REQ	70:01:04:08:60:00:00:00	TAGMSK_ LEN (See appendix A)
REG_REQ	70:01:05:08:11:11:22:22	TAGMSK_ 0_3 (See appendix A)
REG_REQ	70:01:06:08:33:33:44:44	TAGMSK_ 4_7 (See appendix A)
REG_REQ	70:01:07:08:55:55:66:66	TAGMSK_ 8_11 (See appendix A)
REG_REQ	70:01:01:09:40:40:00:00	INV_CFG (See appendix A)

Step 4: Start reading tag data of the selected tag

REG_REQ	70:01:02:0a:02:00:00:00	TAGACC_BANK (See appendix A)
REG_REQ	70:01:03:0a:00:00:00:00	TAGACC_PTR (See appendix A)
REG_REQ	70:01:04:0a:02:00:00:00	TAGACC_CNT (See appendix A)
REG_REQ	70:01:06:0a:00:00:00:00	TAGACC_ACCPWD (See appendix A)
REG_REQ	70:01:00:f0:10:00:00:00	HST_CMD (See appendix A)

Responses from reader

Command-Begin Packet	02:00:00:80:02:00:00:00: 10:00:00:00:d6:8b:00:00	See appendix A.
Inventory-Response Packet	02:00:05:80:07:00:00:00: ec:8b:00:00:00:00:00:00: 00:00:00:00:30:00:11:11: 22:22:33:33:44:44:55:55: 66:66:18:35:	See appendix A. (This is the selected tag)
Tag-Access Packet	01:00:06:00:04:00:00:00: f0:8b:00:00:c2:00:00:00: 00:00:00:00:e2:00:10:50	See appendix A.
Command-End Packet	02:00:01:80:02:00:00:00: f5:8b:00:00:00:00:00:00	See appendix A.

C.4 Tag Operations: Write Example

There are 4 steps within Write that MUST be done:

Step 1: Set Inventory Parameters

Step 2: Set Inventory Algorithm. (Assume only 1 tag in front of reader)

Step 3: Select the desired tag.

Step 4: Start writing the tag data.

The followings example assumes reading TID bank.

Set Inventory Parameters

Packet Name	Command Data	Description
REG_REQ	70:01:00:07:01:00:00:00	ANT_CYCLES (See appendix A)
REG_REQ	70:01:00:09:80:01:00:00	QUERY_CFG (See appendix A)

Set Inventory Algorithm (Fixed Q = 0)

REG_REQ	70:01:02:09:00:00:00:00	INV_SEL (See appendix A)
REG_REQ	70:01:03:09:00:00:00:00	INV_ALG_PARM_0 (See appendix A)
REG_REQ	70:01:05:09:00:00:00:00	INV_ALG_PARM_2 (See appendix A)

Select the desired tag (e.g.111122223333444455556666)

REG_REQ	70:01:01:08:09:00:00:00	TAGMSK_DESC_CFG (See appendix A 1)
REG_REQ	70:01:02:08:01:00:00:00	TAGMSK_BANK (See appendix A)
REG_REQ	70:01:03:08:20:00:00:00	TAGMSK_ PTR (See appendix A)
REG_REQ	70:01:04:08:60:00:00:00	TAGMSK_ LEN (See appendix A)
REG_REQ	70:01:05:08:11:11:22:22	TAGMSK_ 0_3 (See appendix A)
REG_REQ	70:01:06:08:33:33:44:44	TAGMSK_ 4_7 (See appendix A)
REG_REQ	70:01:07:08:55:55:66:66	TAGMSK_ 8_11 (See appendix A)
REG_REQ	70:01:01:09:40:40:00:00	INV_CFG (See appendix A)

Start writing EPC data (e.g.000022223333444455556666)

REG_REQ	70:01:01:0a:0f:00:00:00	TAGACC_DESC_CFG (See appendix A)
REG_REQ	70:01:02:0a:01:00:00:00	TAGACC_BANK (See appendix A)
REG_REQ	70:01:03:0a:00:00:00:00	TAGACC_PTR (See appendix A)
REG_REQ	70:01:04:0a:06:00:00:00	TAGACC_CNT (See appendix A)
REG_REQ	70:01:06:0a:00:00:00:00	TAGACC_ACCPWD (See appendix A)
REG_REQ	70:01:09:0a:00:00:02:00	TAGWRDAT_0 (See appendix A)
REG_REQ	70:01:0a:0a:22:22:03:00	TAGWRDAT_1 (See appendix A)
REG_REQ	70:01:0b:0a:33:33:04:00	TAGWRDAT_2 (See appendix A)
REG_REQ	70:01:0c:0a:44:44:05:00	TAGWRDAT_3 (See appendix A)
REG_REQ	70:01:0d:0a:55:55:06:00	TAGWRDAT_4 (See appendix A)
REG_REQ	70:01:0e:0a:66:66:07:00	TAGWRDAT_5 (See appendix A)
REG_REQ	70:01:00:f0:11:00:00:00	HST_CMD (See appendix A)
Command-Begin Packet	02:00:00:80:02:00:00:00: 11:00:00:00:d6:8b:00:00	See appendix A.
Tag-Access Packet	01:00:06:00:03:00:00:00: 10:76:01:00:c3:00:00:00: 00:00:00:00	See appendix A.
Command-End Packet	02:00:01:80:02:00:00:00: f5:8b:00:00:00:00:00:00	See appendix A.

C.5 Tag Operations: Search Tag Example

There are 4 steps within Search that MUST be done:

Step 1: Set Inventory Parameters

Step 2: Set Inventory Algorithm.

Step 3: Select the desired tag.

Step 4: Start searching the tag.

The followings example illustrates how to read the TID bank.

Set Inventory Parameters

Packet Name	Command Data	Description
REG_REQ	70:01:00:07:01:00:00:00	ANT_CYCLES (See appendix A)
REG_REQ	70:01:00:09:80:01:00:00	QUERY_CFG (See appendix A)

Set Inventory Algorithm (Fixed Q = 0)

REG_REQ	70:01:02:09:00:00:00:00	INV_SEL (See appendix A)
REG_REQ	70:01:03:09:00:00:00:00	INV_ALG_PARM_0 (See appendix A)
REG_REQ	70:01:05:09:00:00:00:00	INV_ALG_PARM_2 (See appendix A)

Select the desired tag (e.g.111122223333444455556666)

REG_REQ	70:01:01:08:09:00:00:00	TAGMSK_DESC_CFG (See appendix A 1)
REG_REQ	70:01:02:08:01:00:00:00	TAGMSK_BANK (See appendix A)
REG_REQ	70:01:03:08:20:00:00:00	TAGMSK_ PTR (See appendix A)
REG_REQ	70:01:04:08:60:00:00:00	TAGMSK_ LEN (See appendix A)
REG_REQ	70:01:05:08:11:11:22:22	TAGMSK_ 0_3 (See appendix A)
REG_REQ	70:01:06:08:33:33:44:44	TAGMSK_ 4_7 (See appendix A)
REG_REQ	70:01:07:08:55:55:66:66	TAGMSK_ 8_11 (See appendix A)
REG_REQ	70:01:01:09:40:40:00:00	INV_CFG (See appendix A)

Start Searching

REG_REQ	Downlink	70:01:00:f0:0f:00:00:00	HST_CMD (See appendix A)
Command-Begin Packet	Downlink	02:01:00:80:02:00:00:00:0f:00:00:00:61:44:00:00	See appendix A.
Inventory-Response Packet**s**	Uplink	02:00:05:80:07:00:00:00:73:44:00:00:81:5f:83:06:00:00:00:00:30:00:11:11:22:22:33:33:44:44:55:55:66:66:71:34	See appendix A. (This is the tag info if tag is found. The reader will keep on sending tag data inventoried back to the host processor until you stop inventory)

Stop Searching

ABORT control command	Downlink	40:03:00:00:00:00:00:00	This is the command you use to stop inventory.
ABORT control command return	Uplink	40:03:bf:fc:bf:fc:bf:fc	This is the command response you will receive for stopping inventory.

Note: After the "ABORT" command to stop inventory, a 2 seconds delay is required for the reader to clear buffer before it can execute another command

C.6 Tag Operations: Lock Example

There are 4 steps within Lock that MUST be done:

Step 1: Set Inventory Parameters

Step 2: Set Inventory Algorithm. (Assume only 1 tag in front of reader)

Step 3: Select the desired tag.

Step 4: Start locking the access password memory.

The followings example assumes locking access password memory.

Set Inventory Parameters

Packet Name	Command Data	Description
REG_REQ	70:01:00:07:01:00:00:00	ANT_CYCLES (See appendix A)
REG_REQ	70:01:00:09:80:01:00:00	QUERY_CFG (See appendix A)

Set Inventory Algorithm (Fixed Q = 0)

REG_REQ	70:01:02:09:00:00:00:00	INV_SEL (See appendix A)
REG_REQ	70:01:03:09:00:00:00:00	INV_ALG_PARM_0 (See appendix A)
REG_REQ	70:01:05:09:00:00:00:00	INV_ALG_PARM_2 (See appendix A)

Select the desired tag (e.g.111122223333444455556666)

REG_REQ	70:01:01:08:09:00:00:00	TAGMSK_DESC_CFG (See appendix A 1)
REG_REQ	70:01:02:08:01:00:00:00	TAGMSK_BANK (See appendix A)
REG_REQ	70:01:03:08:20:00:00:00	TAGMSK_ PTR (See appendix A)
REG_REQ	70:01:04:08:60:00:00:00	TAGMSK_ LEN (See appendix A)
REG_REQ	70:01:05:08:11:11:22:22	TAGMSK_ 0_3 (See appendix A)
REG_REQ	70:01:06:08:33:33:44:44	TAGMSK_ 4_7 (See appendix A)
REG_REQ	70:01:07:08:55:55:66:66	TAGMSK_ 8_11 (See appendix A)
REG_REQ	70:01:01:09:40:40:00:00	INV_CFG (See appendix A)

Start locking access password memory (assume access password is 0x11223344)

REG_REQ	70:01:01:0a:0f:00:00:00	TAGACC_DESC_CFG (See appendix A)
REG_REQ	70:01:05:0a:80:00:03:00	TAGACC_LOCKCFG (See appendix A)
REG_REQ	70:01:06:0a:44:33:22:11	TAGACC_ACCPWD (See appendix A)
REG_REQ	70:01:00:f0:12:00:00:00	HST_CMD, 12 for locking (See appendix A)
Command-Begin Packet	02:00:00:80:02:00:00:00: 11:00:00:00:d6:8b:00:00	See appendix A.
Tag-Access Packet	01:00:06:00:03:00:00:00: 10:76:01:00:c5:00:00:00: 00:00:00:00	See appendix A.
Command-End Packet	02:00:01:80:02:00:00:00: f5:8b:00:00:00:00:00:00	See appendix A.

Appendix D: Reader Modes (Link Profiles)

The reader modes (link profiles) are different for the legacy reader series and the new intelligent reader series because they use different RFID IC.

Legacy Reader Series (CS203/CS468INT/CS469) – Profile 2 is the typically used profile.

There are 5 link profiles. Profile 2 is the typically used profile.

Profile	0	1	2 (typically used profile)	3	4	5
R-T Modulation	DSB-ASK	DSB-ASK	PR-ASK	PR-ASK	DSB-ASK	PR-ASK
Tari (µs)	25.00	12.50	25.00	25.00	6.25	25.00
R-T speed (kbps)	40	80	40	40	160	40
PIE	2:1	2:1	1.5:1	1.5:1	1.5:1	1.5:1
Pulse Width (uS)	12.50	6.25	12.50	12.50	3.13	12.50
T-R LF (kbps)	40	160	250	300	400	250
T-R Modulation	FM0	Miller-2	Miller-4	Miller-4	FM0	Miller-2
Divide Ratio	8	8	64/3	64/3	8	64/3
T-R Data Rate (kbps)	40	80	62.5	75	400	125

New Intelligent Reader Series (CS463/CS468XJ/CS468X/CS203X/CS206) – Profile 1 is the typically used profile.

There are 4 link profiles: 0, 1, 2, 3. **Profile 1 is the typically used profile.** Only 1 profile is active at any time in the CSL device. The purpose of each link profile is explained below. These definitions correspond to different application or physical scenarios. The user should try out each profile to see which one gives best performance.

Reader Mode/ Link Profile	**0**	**1 (typically used profile)**	**2**	**3**
Definition	Best Multipath Fading Resistance	Longest Read Range, Dense Reader Mode	Read Range and Throughput, Dense Reader Mode	Maximum Throughput
R-T Modulation	DSB-ASK	PR-ASK	PR-ASK	DSB-ASK
Tari (μs)	25.00	25.00	25.00	6.25
X	1.00	0.50	0.50	0.50
PW (Pulse Width in usec)	12.50	12.50	12.50	3.13
RTcal (usec)	75.00	62.50	62.50	15.63
TRcal (usec)	200.00	85.33	71.11	20.00
DR (Divide Ratio)	8	64/3	64/3	8
T-R Modulation	FM0	Miller-4	Miller-4	FM0
TRExt	1	1	1	1
Link Frequency(LF) (KHz)	40	250	300	400
Data Rate (Kbps)	40	62.5	75	400

Appendix E: Sessions

Session is a concept of the EPC Global Standard to allow a tag to respond to multiple readers inventorying it at the same time, each using a different session number.

There are 4 possible sessions: S0, S1, S2, S3.

The user however has to be careful because these 4 sessions have different behaviors, notably how the tag flag "persists" in time. A tag, before being inventoried or when just after being powered on, has a flag of State A. When it is inventoried, the flag will go to State B. The tag flag will stay in State B until the tag powers off or the persistence time is up.

A reader can declare it only wants to inventory flag A, so that after a tag is inventoried and its flag gone to State B, it will no longer respond to further inventory rounds – until the end of the persistence time.

Now for S0, S1, S2 and S3, the persistence times are DIFFERENT! Because of that, one has to be very careful in choosing which session to use.

Session	**Tag Flags Persistence Time**
S0	Tag Energized: indefinite Tag Not Energized: none
S1	Tag Energized: 0.5 second < Persistence Time < 5 seconds Tag Not Energized: 0.5 second < Persistence Time < 5 seconds
S2	Tag Energized: indefinite Tag Not Energized: 2 seconds < Persistence Time
S3	Tag Energized: indefinite Tag Not Energized: 2 seconds < Persistence Time

Appendix F: Tag Population and Q

Tag Population is the RFID tag population that is to be inventoried. To be more precise, it is the population of tags that can be "seen" by the RFID reader.

Q is an EPC Global Standard concept related to the way a group of tags is inventoried. When a reader broadcasts its desire to inventory tags, it sends out a Q value. The tag will, based on that Q, calculate a certain number and define that as the number of repeated inventories the reader will do. Basically, the relationship of Inventory Repeats and Q is:

Inventory Repeats = 2^Q

The tag will then choose by random a certain number less than this Inventory Repeats. When the reader starts doing inventory, the tag will then respond at that repeat number.

In other words, the Inventory Repeats should correspond to Tag Population:

Tag Population = Inventory Repeats = 2^Q

For example, if there are 8 tags, then in theory the Q can be 3, and if each tag chooses a number different from that of the other 7 (miraculously, of course), then the 8 tags will be inventoried in an orderly manner in turn.

Of course this will never happen, as the tags will easily choose a number the same as that of another one, and a collision will occur.

Therefore, it is a normal practice to have a bigger Q, such as 4 in this case, so that the 8 tags would have a lower chance of choosing the same number.

Therefore, reversing the equation, ideally, we can have:

Q = INTEGER(LOG_2(Tag Population))

But in reality, we need some headroom, so that:

Q = INTEGER(LOG_2(Tag Population x 2)+1

Appendix G: Query Algorithm

There are 2 types of Query Algorithm: Fixed Q and Dynamic Q.

For Fixed Q, the Q value does not change. In other words, the expected Tag Population does not change.

For Dynamic Q, the Q value changes adaptively: when there are a lot of inventory repeats where no tags respond, the reader will interpret that there are not that many RFID tags in the ~~front~~ environment, and hence it is more efficient to change the Q to a smaller value. When there are a lot of inventory repeats where the reader receives data but they do not satisfy checksum, meaning there is heavy collision, then the reader will interpret that there are too many RFID tags in the environment and hence it is better to increase the value of Q. Dynamic Q algorithm is a way to allow the RFID reader to adapt to different amount of RFID tags being seen by the reader. The idea is that if there are a smaller number of tags in the environment, then the Q can be reduced and the reader can collect all the tag data faster.

Appendix H: Target

Target refers to the target flag that the reader wants to inventory. There are 2 possible flags of an RFID tag: State A and State B.

When an RFID tag is first powered up, it has a flag of State A. After it is inventoried, the state of the flag becomes State B.

The tag will only go back to State A if either it is powered off and powered on again, or if its persistence time has run up (See Appendix B).

For each round of inventory, the reader sends out a notification to the environment which tag flag state it wants to inventory. It can inventory State A, or it can inventory State A and State B alternatively from one round of inventory to the next round of inventory.

In theory, it is a good thing to inventory only State A. The reason being that those tags that have been inventoried should not respond again, and will hence quickly reduce the amount of collisions between tags. Generally in practice if you set inventory to State A only, the inventory of large amount of tags can be faster.

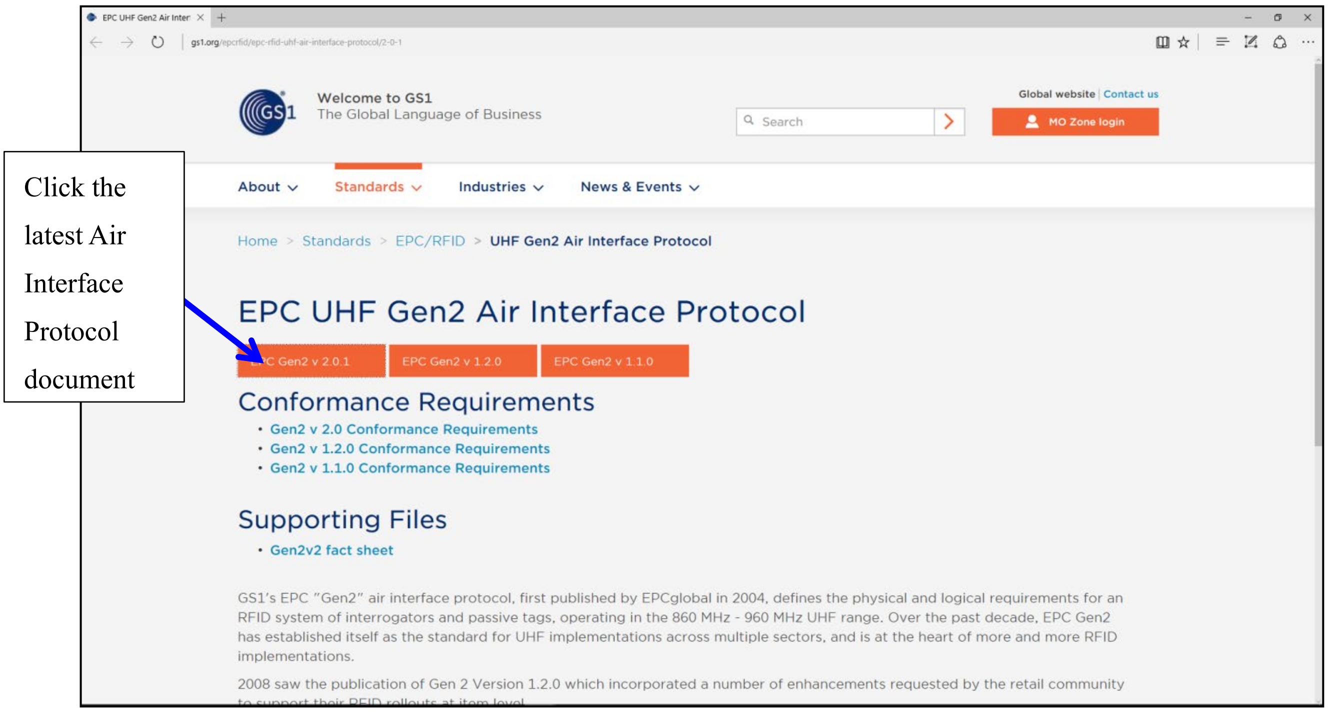
The only catch is that when a tag responds to the reader, it does not know another tag is colliding with it. It sends out the response and thinks it has done the job, hence transitioning to flag State B. So in such a case, the tag will not respond to further inventory queries by the reader, even though its response has been lost due to collisions. Because of that, sometimes the user will set the inventory to target State A in one inventory round, and then State B in the next round, and vice versa, and so on. This is called A/B Toggle or A & B Dual Target or simply Dual Target.

Appendix I: Security

There are 4 actions you can apply on the memory inside an RFID tag:

1) Lock
2) Unlock
3) Permanent Lock
4) Permanent Unlock

You can obtain an EPC Global document which can be downloaded from the EPC Global website that explains this: https://www.gs1.org/epcrfid/epc-rfid-uhf-air-interface-protocol/2-0-1.
Once there, press the button showing the latest air interface protocol document and click on it to get the pdf file.


Click the latest Air Interface Protocol document
EPC UHF Gen2 Air Inter
gs1.org/epcrfid/epc-rfid-uhf-air-interface-protocol/2-0-1
Welcome to GS1
The Global Language of Business
Search
Global website | Contact us
MO Zone login
About
Standards
Industries
News & Events
Home > Standards > EPC/RFID > UHF Gen2 Air Interface Protocol
EPC Gen2 v 2.0.1
EPC Gen2 v 1.2.0
EPC Gen2 v 1.1.0

EPC UHF Gen2 Air Interface Protocol

Conformance Requirements

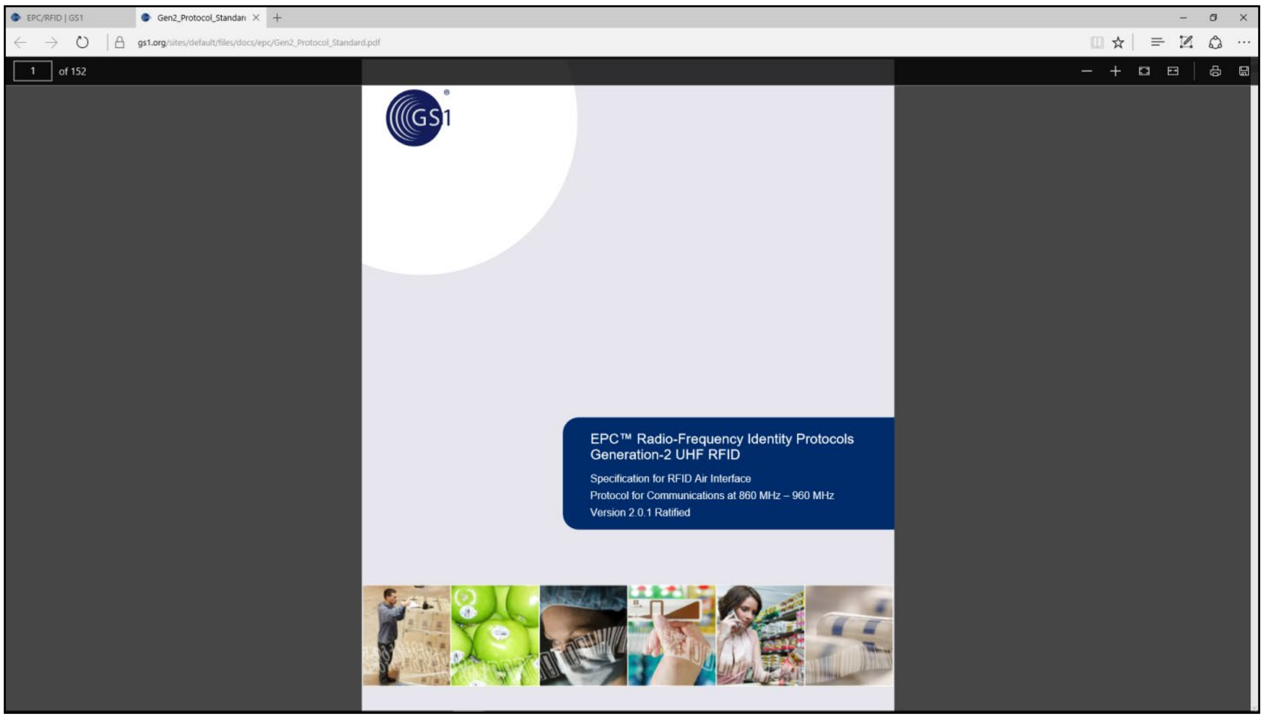
- Gen2 v 2.0 Conformance Requirements
- Gen2 v 1.2.0 Conformance Requirements
- Gen2 v 1.1.0 Conformance Requirements

Supporting Files

- Gen2v2 fact sheet

GS1's EPC "Gen2" air interface protocol, first published by EPCglobal in 2004, defines the physical and logical requirements for an RFID system of interrogators and passive tags, operating in the 860 MHz - 960 MHz UHF range. Over the past decade, EPC Gen2 has established itself as the standard for UHF implementations across multiple sectors, and is at the heart of more and more RFID implementations.

2008 saw the publication of Gen 2 Version 1.2.0 which incorporated a number of enhancements requested by the retail community to support their RFID rollouts at item level.


EPC/RFID | GS1
Gen2_Protocol_Standard
gs1.org/sites/default/files/docs/epc/Gen2_Protocol_Standard.pdf
1
of 152
GS1

EPC™ Radio-Frequency Identity Protocols
Generation-2 UHF RFID

Specification for RFID Air Interface
Protocol for Communications at 860 MHz – 960 MHz
Version 2.0.1 Ratified

For the Access Password and Kill Password the security locking affects both reading and writing.

For the EPC memory bank and the User memory bank, the security locking affects only writing.

For the TID memory bank, since we are the user and not the manufacturing vendor, there is no security action that can be applied. It has been permanently unlocked in the factory and it cannot be changed.

Appendix J – Tag Focus

Tag Focus is a special feature of Impinj tag IC. When enabled, and when the reader is using Session S1 and Target A to query the tag, the tag will, once inventoried, remain in Flag B until the inventory is completed.

This is in contrast to the normal EPC query using S1 and Target A, where the tag will only remain in Flag B for 2 to 5 seconds – this time being defined as persistence time.

The original purpose of EPC S1 and Target A is so that those tags that have been inventoried before would not come back and be inventoried again so quickly, so that more time slots are available for other tags that have not been inventoried yet. However, the time of 2 to 5 seconds are simply too short if there are many tags in the environment being illuminated by the reader. This is particularly true if the reader is "seeing" a whole bunch of tagged items in a warehouse, where there may be more than 1000 tags the reader can "see" at any moment. The end result is before all the tags have been inventoried, the early inventoried tags, having passed 5 seconds after it was first inventoried, join back to be inventoried!!

In the early days of EPC standard, a few hundred tags being inventoried is already a rather unthinkable matter, and so 2 to 5 seconds of persistence time is enough. Nowadays, with the ever improving sensitivity of tags, 5 seconds is not enough.

With Impinj Tag Focus enabled, the tag simply would not respond again until the inventory is completely over.

Appendix K – Fast ID

Fast ID is a special feature of Impinj tag IC. When enabled, a normal inventory will bring back BOTH Bank 1 EPC Bank and Bank 2 TID Bank. The first 5 bits of PC will be automatically enlarged to accommodate the additional length of the Bank 2 TID Bank. Since Impinj Tag IC has 96 bits Bank 2, so that is the added length. When one receives this lengthened inventory return, one has to separate out the TID data and the EPC data. TID data is at the end of the returned data. In other words, Bank 2 is tagged at the end of the original EPC data.

Appendix L – Receive Path Parameters

The receive path of CSL Reader consists of RF and IF Low Noise Amplifier and Automatic Gain Control Amplifier. The gains of these can be controlled.

Here are the 4 parameters:

1) RF LNA High Compression Mode:	0 or 1; if RF LNA Gain is 13, this must be 0. Default is 1
2) RF LNA Gain:	1, 7, or 13 dB. If this is set to 13 dB, then RF High Compression Mode must be 0. Default is 1
3) IF LNA Gain:	24, 18, 12, 6 dB. Default is 24
4) AGC Gain Range:	-12, -6, 0, 6 dB. Default is -6

Appendix M: Models & Regulatory Region

There are various models, denoted by the alphanumeric key to the right of the dash after the "CSXXX-", here denoted by "**N**". The applicable regulatory regions for each model are described below:

N=1:	865-868 MHz for Europe ETSI, Russia, Mid-East countries, 865-867 MHz for India
N=2:	902-928 MHz, FCC, for USA, Canada and Mexico. Hopping frequencies locked
N=2 AS:	920-926 MHz, Australia. Hopping frequencies locked
N=2 NZ:	921.5-928 MHz, New Zealand. Hopping frequencies locked
N=2 OFCA:	920-925 MHz, Hong Kong. Hopping frequencies locked
N=2 RW:	920-928 MHz, Rest of the World, e.g. Philippines, Brazil, Peru, Uruguay, etc.
N=4:	922-928 MHz, Taiwan
N=7:	920-925 MHz, China
N=8:	916.7-920.9 MHz, Japan
N=9:	915-921 MHz, Europe Upper Band

Appendix N: Technical Support

All technical support should be sent to the following email:

info@convergence.com.hk