ZEBRA

HOST API FOR RFID MODULE OF THE DS9808-R SCANNER

Revisions

REV	DESCRIPTION	DATE
A	Release Document	9/21/11
A	Update tables on pages 8, 9, 11, 13, and 14.	1/18/12

TABLE OF CONTENTS

1 Introduction

1.1 Purpose

The purpose of this document is to describe the host Application Program Interface (API) for the advanced reading and writing of RFID tags using the Durango RFID Module with the DS9808 Sierra Digital Scanner.

1.2 Terms and Definitions

- EPC Electronic Product Code
- RFID Radio-frequency identification

1.3 References

[ref 1] Zebra DS9808 Digital Scanner Product Ref Guide,
part number 72E-112999-xx

[ref 2] Zebra Scanner SDK v1.0, SCANNERSDK100.0043
http://www.zebra.com/software

[ref 3] [ref 3] EPCglobal-uhfc1g, Tag Protocol – UHF Class-1 Gen-2
http://www.epcglobalinc.org

2 Description

2.1 System Overview

The DS9808 scanner provides imager based scanning for hands-free operation (presentation) and both imager and laser based scanning for hand-held operation.

The D9808 may optionally be equipped with the D9808-R RFID Module (based on the Sailfish Module), an RFID radio engine intended to enable mobile computers and bar code scanners to read and modify Electronic Product Code (EPC) Gen2 UHF RFID tags.
The DS9808 offers several host interfaces; some which behave like simple input devices, some which offer intelligent control to the connected host.

The host control of the advanced RFID capabilities of the D9808 with RFID makes use of the existing RSM protocol by adding new attributes for the setup and execution of RFID functions.

3 Setup

3.1 Using with USB OPOS

To use the RFID control with the Zebra Scanner SDK (see [ref 2])
you may use the Scanner WMI Sample Application (Scanner_WMI_test.exe).
This program offers the “SetAttributes” and “GetAttributes” methods which are needed to access the RFID attributes. All examples in this document assume you are using this sample application.

The DS9808-R scanner must have a USB cable connection and should be configured with the USB Device Type set to one of the following:

- USB OPOS Handheld
- IBM Handheld USB
- IBM Table Top USB

Refer to the DS9808 Product Reference Guide (see [ref 1]) for details and configuration bar codes.

4 RSM Attributes for RFID

The RFID API is expressed in a series of RSM attributes.

ATTRIBUTE NUMBER	ATTRIBUTE NAME	RSM TYPE	SIZE (BYTES)	ACCESS	DESCRIPTION
35001	RFID_LAST_TAG_ID	‘A’	34	R	The EPC Tag ID of the last tag reported. (size-encoded binary)
35002	RFID_TAG_ID	‘A’	34	W	The EPC Tag ID of the tag to be operated upon. (size-encoded binary)
35003	RFID_BANK	‘B’	1	W	Desired Tag Bank: 0 = reserved, 1 = EPC, 2 = TID, 3 = User
35004	RFID_DATA	‘A’	66	RW	Buffer for read, write, and lock (size-encoded binary)
35005	RFID_OFFSET	‘W’	2	W	Word offset into tag buffer
35006	RFID_LENGTH	‘W’	2	W	Words of data to read from tag buffer. 0 means entire bank
35007	RFID_PASSWORD	‘A’	4	W	Binary password for privileged operations
35008	RFID_COMMAND	‘B’	1	W	Execute command: 1 = Read 2 = Write 3 = Lock 4 = Kill
35009	RFID_CMD_STATUS	‘W’	2	R	Resulting status from executing a command
35010	RFID_TAG_CACHE	‘W’	2	RW	Internal Tag Cache Size: Read for current cache size Write 0 to clear the cache

For “size-encoded” binary data, the first two bytes contain the length (MSB, LSB) for the data to be considered (needed because the RSM attributes are fixed size).

4.1 RFID_LAST_TAG_ID – Last Reported Tag

As a convenience, the RFID_LAST_TAD_ID may be used to get the raw EPC of the last tag reported by normal tag reading operation of the D9808 (normally for an RFID_READ event, the scanner issues a 2-tone beep).

As an example, to get the RFID_LAST_TAD_ID attribute after the scanner reports the tag 3005FB63AC1F3681EC880469.

METHOD	INPUT	ATTVALUELIST
GetAttributes	<attrib_list>35001</attrib_list>	<attrib_list> <attribute> <id>35001</id> <datatype>A</datatype> <permission>R</permission> <value>0x00 0x0c 0x30 0x05 0xfb 0x63 0xac 0x1f 0x36 0x81 0xec 0x88 0x04 0x69 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 </value> </attribute> </attrib_list>

In raw RSM protocol:

```
Send: 00 08 02 00 88 B9 FF FF
Recv: 00 33 02 00 88 B9 41 01 42 00 22 00 00 00 0C 30 05 FB 63 AC 1F 36 81
EC 88 04 69 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 FF
FF FF FF
```

4.2 RFID_CMD_STATUS - Common Operation Response

All operations (read, write, kill, lock) return a status in the RFID_CMD_STATUS attribute:

0x0000	Success
0x0001	No RFID module
0x0002	Tag Not Found
0x0003	Timeout
0x0004	Tag CRC Error
0x01xx	Tag Backscatter Error, LSB indicates the error_code as per EPC Protocol
0x02xx	Tag Access error. LSB indicates the error code
0x03xx	Bad Parameter, the LSB indicates which parameter: 1 = Command 2 = Tag_ID 3 = Bank 4 = Data 5 = Offset 6 = Password

4.3 RFID_COMMAND

The RFID_COMMAND attribute is used to execute the various tag operations. Each operation has “parameter” attributes that should be setup prior to executing the command.

4.3.1 Tag Read Operation

The Tag Read operation requires the following attributes to be set using RSM SetAttributes:

ATTRIBUTE NUMBER	ATTRIBUTE	DESCRIPTION
35002	RFID_TAG_ID	EPC code of desired tag
35003	RFID_BANK	Desired memory bank of tag
35005	RFID_OFFSET	Word offset into the memory bank
35006	RFID_LENGTH	Number of words to read
35007	RFID_PASSWORD	Optional access password
35008	RFID_COMMAND	1 = “read”

The Tag Read operation is activated by the SetAttribute of the RFID_COMMAND attribute.

The result of the operation may be retrieved by the RSM GetAttributes of the following attributes. If the RFID_CMD_STATUS attribute indicates success, then the RFID_DATA attribute will have the requested data.

ATTRIBUTE NUMBER	ATTRIBUTE	DESCRIPTION
35009	RFID_CMD_STATUS	Resulting status from the read
35004	RFID_DATA	Buffer for read

4.3.1.1 Example Read

As an example, to read the entire EPC bank of the tag with EPC of 3005FB63AC1F3681EC880469.

METHOD	INPUT	ATTVALUELIST
SetAttributes	<attrib_list> <attribute> <!--**RFIDTagID**--> <id>35002</id> <datatype>A</datatype> <value>0x00 0x0c 0x30 0x05 0xFB 0x63 0xAC 0x1F 0x36 0x81 0xEC 0x88 0x04 0x69 </value> </attribute> <attribute> <!--**RFIDBank**--> <id>35003</id> <datatype>B</datatype> <value>1</value> </attribute> <attribute> <!--**RFIDOffset**--> <id>35005</id> <datatype>W</datatype> <value>0</value> </attribute> <attribute> <!--**RFIDLength**--> <id>35006</id> <datatype>W</datatype> <value>0</value> </attribute> <attribute> <!--**RFIDCommand**--> <id>35008</id> <datatype>B</datatype> <value>1</value> </attribute> </attrib_list>	n/a
GetAttributes	<attrib_list>35009,35004</attrib_list>	<attrib_list> <attribute> <id>35009</id> <name>"RFIDCmdStatus"</name> <datatype>W</datatype> <permission>R</permission> <value>0</value> </attribute> <attribute> <id>35004</id> <name>"RFIDData"</name> <datatype>A</datatype> <permission>R W P</permission> <value>0x00 0x10 0xA0 0x4B 0x30 0x00 0x30 0x05 0xFB 0x63 0xAC 0x1F 0x36 0x81 0xEC 0x88 0x04 0x69 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 </value> </attribute> </attrib_list>

In raw RSM protocol:

```
Send: 00 33 05 00 88 BA 41 00 42 00 0E 00 00 00 0C 30 05 FB 63 AC 1F 36 81
EC 88 04 69 88 BB 42 00 01 88 BD 57 00 00 00 88 BE 57 00 00 00 88 C0 42 00
01 FF FF
Recv: 00 04 05 00
Send: 00 0A 02 00 88 C1 88 BC FF FF
Recv: 00 59 02 00 88 C1 57 01 00 00 88 BC 41 03 42 00 42 00 00 00 10 AE 57
30 00 30 05 FB 63 AC 1F 36 81 EC 88 04 69 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF FF
```

As a second example, to read the PC word (second word, whose value is 0x3000) from the EPC buffer of the same tag:

METHOD	INPUT	ATTVALUELIST
SetAttributes	<attrib_list> <attribute> <!--**RFIDTagID**--> <id>35002</id> <datatype>A</datatype> <value>0x00 0x0c 0x30 0x05 0xFB 0x63 0xAC 0x1F 0x36 0x81 0xEC 0x88 0x04 0x69 </value> </attribute> <attribute> <!--**RFIDBank**--> <id>35003</id> <datatype>B</datatype> <value>1</value> </attribute> <attribute> <!--**RFIDOffset**--> <id>35005</id> <datatype>W</datatype> <value>1</value> </attribute> <attribute> <!--**RFIDLength**--> <id>35006</id> <datatype>W</datatype> <value>1</value> </attribute> <attribute> <!--**RFIDCommand**--> <id>35008</id> <datatype>B</datatype> <value>1</value> </attribute> </attrib_list>	n/a
GetAttributes	<attrib_list>35009,35004</attrib_list>	<attrib_list> <attribute> <id>35009</id> <name>"RFIDCmdStatus"</name> <datatype>W</datatype> <permission>R</permission> <value>0</value> </attribute> <attribute> <id>35004</id> <name>"RFIDData"</name> <datatype>A</datatype> <permission>R W P</permission> <value>0x00 0x02 0x30 0x00

METHOD	INPUT	ATTVALUELIST
		0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 0x00 </value> </attribute> </attrib_list>

In raw RSM protocol:

```
Send: 00 33 05 00 88 BA 41 00 42 00 0E 00 00 00 0C 30 05 FB 63 AC 1F 36 81
EC 88 04 69 88 BB 42 00 01 88 BD 57 00 00 01 88 BE 57 00 00 01 88 C0 42 00
01 FF FF
Recv: 00 04 05 00
Send: 00 0A 02 00 88 C1 88 BC FF FF
Recv: 00 59 02 00 88 C1 57 01 00 00 88 BC 41 03 42 00 42 00 00 00 02 30 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00 00
00 00 00 00 00 00 00 00 00 00 00 00 FF FF FF FF
```

4.3.2 Tag Write Operation

The Tag Write operation requires the following attributes to be set using RSM SetAttributes:

ATTRIBUTE NUMBER	ATTRIBUTE	DESCRIPTION
35002	RFID_TAG_ID	EPC code of desired tag
35003	RFID_BANK	Desired memory bank of tag
35004	RFID_DATA	data to write
35005	RFID_OFFSET	Word offset into the memory bank
35007	RFID_PASSWORD	Optional access password
35008	RFID_COMMAND	2 = “write”

The Tag Write operation is activated by the SetAttribute of the RFID_COMMAND attribute.

The result of the operation may be retrieved by the RSM GetAttributes of the RFID_CMD_STATUS attribute.

ATTRIBUTE NUMBER	ATTRIBUTE	DESCRIPTION
35009	RFID_CMD_STATUS	Resulting status from the Write

4.3.2.1 Example Write

As an example, to write 0x1234 into the second word of the user bank of the tag with EPC of 3005FB63AC1F3681EC880469.

METHOD	INPUT	ATTVALUELIST
SetAttributes	<attrib_list> <attribute> <!--**RFIDTagID**--> <id>35002</id> <datatype>A</datatype> <value>0x00 0x0c 0x30 0x05 0xFB 0x63 0xAC 0x1F 0x36 0x81 0xEC 0x88 0x04 0x69 </value> </attribute> <attribute> <!--**RFIDBank**--> <id>35003</id> <datatype>B</datatype> <value>3</value> </attribute> <attribute> <!--**RFIDOffset**--> <id>35005</id> <datatype>W</datatype> <value>1</value> </attribute> <attribute> <!--**RFIDData**--> <id>35004</id> <datatype>A</datatype> <value>0x00 0x02 0x12 0x34 </value> </attribute> <attribute> <!--**RFIDCommand**--> <id>35008</id> <datatype>B</datatype> <value>2</value> </attribute> </attrib_list>	n/a
GetAttributes	<attrib_list>35009</attrib_list>	<attrib_list> <attribute> <id>35009</id> <name>"RFIDCmdStatus"</name> <datatype>W</datatype> <permission>R</permission> <value>0</value> </attribute> </attrib_list>

In raw RSM Protocol:

```
Send: 00 33 05 00 88 BA 41 00 42 00 0E 00 00 00 0C 30 05 FB 63 AC 1F 36 81
EC 88 04 69 88 BB 42 00 03 88 BD 57 00 00 01 88 BC 41 00 42 00 04 00 00 00
02 12 34 88 C0 42 00 02 FF FF
Recv: 00 04 05 00
Send: 00 08 02 00 88 C1 FF FF
Recv: 00 0E 02 00 88 C1 57 01 00 00 FF FF FF FF
```

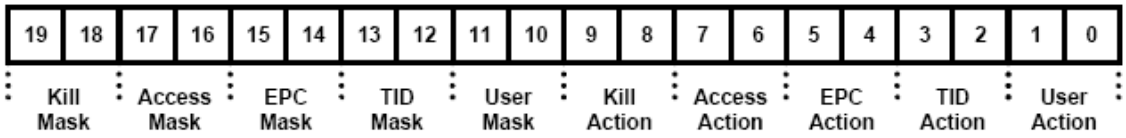
4.3.3 Tag Lock Operation

The Tag Lock operation requires the following attributes to be set using RSM SetAttributes:

ATTRIBUTE NUMBER	ATTRIBUTE	DESCRIPTION
35002	RFID_TAG_ID	EPC code of desired tag
35004	RFID_DATA	Lock configuration
35007	RFID_PASSWORD	Access password (required)
35008	RFID_COMMAND	3 = “Lock”

The Lock Configuration is 4 bytes, defined in the EPC Protocol Spec as:

Lock-Command Payload



Masks and Associated Action Fields

	Kill pwd		Access pwd		EPC memory		TID memory		User memory	
	19	18	17	16	15	14	13	12	11	10
Mask	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write	skip/ write
	9	8	7	6	5	4	3	2	1	0
Action	pwd read/ write	perma lock	pwd read/ write	perma lock	pwd write	perma lock	pwd write	perma lock	pwd write	perma lock

pwd-write	permalock	Description
0	0	Associated memory bank is writeable from either the **open** or **secured** states.
0	1	Associated memory bank is permanently writeable from either the **open** or **secured** states and may never be locked.
1	0	Associated memory bank is writeable from the **secured** state but not from the **open** state.
1	1	Associated memory bank is not writeable from any state.
pwd-read/write	**permalock**	**Description**
0	0	Associated password location is readable and writeable from either the **open** or **secured** states.
0	1	Associated password location is permanently readable and writeable from either the **open** or **secured** states and may never be locked.
1	0	Associated password location is readable and writeable from the **secured** state but not from the **open** state.
1	1	Associated password location is not readable or writeable from any state.

The password is required for the Tag Lock operation and must match the access password of the tag (bytes 4-7 of the reserved bank).

The Tag Lock operation is activated by the SetAttribute of the RFID_COMMAND attribute.

The result of the operation may be retrieved by the RSM GetAttributes of the RFID_CMD_STATUS attribute.

ATTRIBUTE NUMBER	ATTRIBUTE	DESCRIPTION
35009	RFID_CMD_STATUS	Resulting status from the Kill

4.3.3.1 Example Tag Lock Operation

As an example, to hide the kill password (bytes 0-3 of the reserved bank) of the tag with EPC of 3005FB63AC1F3681EC880469, assuming the access password (bytes 4-7 of the reserved bank) is 0x87654321.

To setup the lock configuration, set:

"kill pwd" mask = 11

"kill pwd" action = 10

Therefore the lock configuration is 0x000c0200.

METHOD	INPUT	ATTVALUELIST
SetAttributes	<attrib_list> <attribute> <!--**RFIDTagID**--> <id>35002</id> <datatype>A</datatype> <value>0x00 0x0c 0x30 0x05 0xFB 0x63 0xAC 0x1F 0x36 0x81 0xEC 0x88 0x04 0x69 </value> </attribute> <attribute> <!--**RFIDData**--> <id>35004</id> <datatype>A</datatype> <value>0x00 0x04 0x00 0x0c 0x02 0x00 </value> </attribute> <attribute> <!--**RFIDPassword**--> <id>35007</id> <datatype>A</datatype> <value>0x87 0x65 0x43 0x21</value> </attribute> <attribute> <!--**RFIDCommand**--> <id>35008</id> <datatype>B</datatype> <value>3</value> </attribute> </attrib_list>	n/a
GetAttributes	<attrib_list>35009</attrib_list>	<attrib_list> <attribute> <id>35009</id> <name>"RFIDCmdStatus"</name> <datatype>W</datatype> <permission>R</permission> <value>0</value> </attribute> </attrib_list>

4.3.4 Tag Kill Operation

The Tag Kill operation requires the following attributes to be set using RSM SetAttributes:

ATTRIBUTE NUMBER	ATTRIBUTE	DESCRIPTION
35002	RFID_TAG_ID	EPC code of desired tag
35007	RFID_PASSWORD	Kill password (required non-zero)
35008	RFID_COMMAND	4 = “kill”

The password is required for the Tag Kill operation and must match the kill password of the tag (bytes 0-3 of the reserved bank). Note that, as per the EPC protocol spec, if the kill password is zero the tag cannot be killed.

The Tag Kill operation is activated by the SetAttribute of the RFID_COMMAND attribute.

The result of the operation may be retrieved by the RSM GetAttributes of the RFID_CMD_STATUS attribute.

ATTRIBUTE NUMBER	ATTRIBUTE	DESCRIPTION
35009	RFID_CMD_STATUS	Resulting status from the Kill

4.3.4.1 Example Tag Kill Operation

As an example, to kill the tag with EPC of 3005FB63AC1F3681EC880469, assuming the kill password (bytes 0-3 of the reserved bank) is 0x12345678.

METHOD	INPUT	ATTVALUELIST
SetAttributes	<attrib_list> <attribute> <!--**RFIDTagID**--> <id>35002</id> <datatype>A</datatype> <value>0x00 0x0c 0x30 0x05 0xFB 0x63 0xAC 0x1F 0x36 0x81 0xEC 0x88 0x04 0x69 </value> </attribute> <attribute> <!--**RFIDData**--> <id>35004</id> <datatype>A</datatype> <value>0x00 0x04 0x00 0x0c 0x02 0x00 </value> </attribute> <attribute> <!--**RFIDPassword**--> <id>35007</id> <datatype>A</datatype> <value>0x12 0x34 0x56 0x78</value> </attribute> <attribute> <!--**RFIDCommand**--> <id>35008</id>	n/a

METHOD	INPUT	ATTVALUELIST
	<datatype>B</datatype> <value>4</value> </attribute> </attrib_list>	
GetAttributes	<attrib_list>35009</attrib_list>	<attrib_list> <attribute> <id>35009</id> <name>"RFIDCmdStatus"</name> <datatype>W</datatype> <permission>R</permission> <value>0</value> </attribute> </attrib_list>

4.4 RFID_TAG_CACHE – Tag Cache Operations

As a diagnostic tool, the RFID_TAG_CACHE may be used to read the current tag cache size or to flush the tag cache. Reading this attribute returns the current number of the unique RFID tags in the cache. Writing any value to this attribute causes the cache to be cleared (flushed).

✓ **Note:** Use caution when clearing the tag cache as all tags in range will be read on the very next inventory. If automatic reading is enabled, this will be immediately.

4.4.1 Example Read Cache Size

METHOD	INPUT	ATTVALUELIST
GetAttributes	<attrib_list>35010</attrib_list>	<attrib_list> <attribute> <id>35010</id> <name>""</name> <datatype>W</datatype> <permission>R W</permission> <value>2</value> </attribute> </attrib_list>

In raw RSM protocol:

```
Send: 00 08 02 00 88 C2 FF FF
Recv: 00 0E 02 00 88 C2 57 03 00 02 FF FF FF FF
```

4.4.2 Example Clear Tag Cache

METHOD	INPUT	ATTVALUELIST
SetAttributes	<attrib_list> <attribute> <id>35010</id> <datatype>W</datatype> <value>0</value> </attribute> </attrib_list>	

In raw RSM protocol:

```
Send: 00 0A 05 00 88 C2 57 03 00 00
Recv: 00 04 05 00
```

Zebra Technologies Corporation
Corporate Headquarters
3 Overlook Point
Lincolnshire, IL 60069
www.zebra.com