MaxReader CLI User Guide

Rev Number 1.05
Date October 31st, 2011

RELEASE

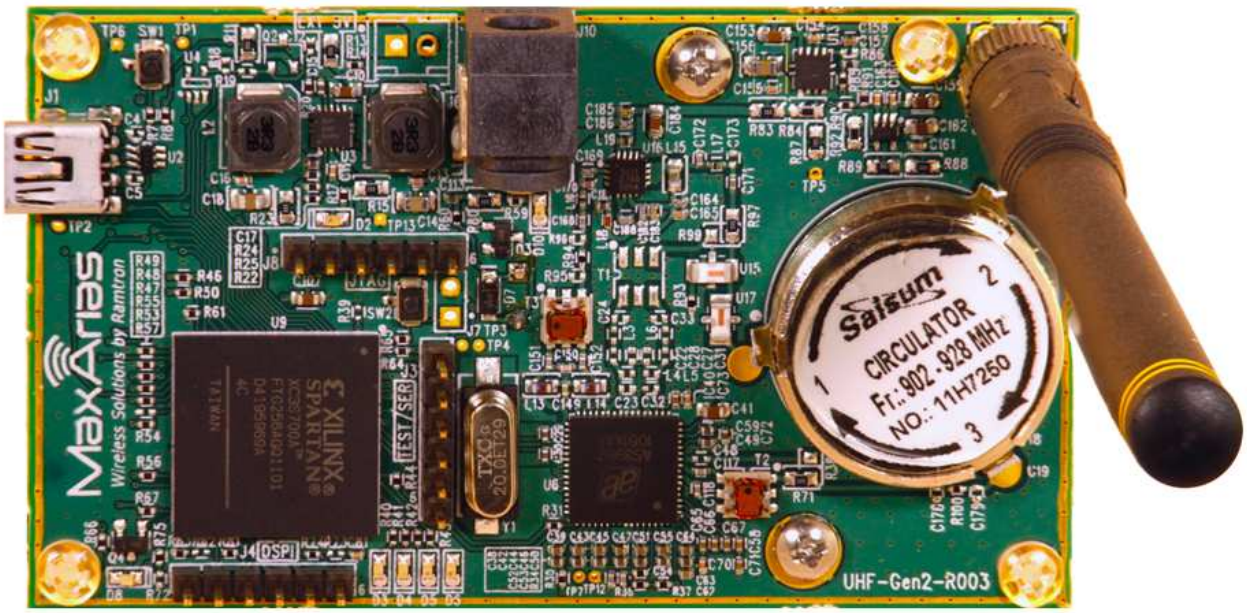


Table of Contents

1 Overview

This document provides a complete description of all commands available in the MaxReader Command Line Interpretter (CLI). The CLI is accessed through a terminal interface such as Windows ® Hyperterminal for Windows XP, Windows 7, or other terminal emulation programs, such as PuTTY.

HyperTerminal may be obtained from www.hilgraeve.com supporting Windows 7.

PuTTY may be downloaded online from www.PuTTY.org. Wide support for PuTTY is available for other operating systems as well.

2 NOMENCLATURE

CLI Command Nomenclature	Description
G2*	All CLI Gen2 commands and CLI Gen2 command sequences shall start with a “G2” command header.
G2*X	CLI Gen2 commands using EBV-formatted extended addressing shall include the character “X” in the command name.
DS*	All DSPI commands shall start with a “DS” command header.
*ON	CLI commands for enabling functions shall use an “ON” command suffix.
*OFF	CLI commands for disabling functions shall use an “OFF” command suffix.

3 READER SETUP

3.1 Hardware Setup

The MaxReader requires the following to be connected to it prior to operation:

1. 5V DC power cable,
2. USB cable, and
3. Antenna.

The MaxReader obtains all the power it requires from the external power supply – no USB bus power is required.

3.2 USB Driver Installation

Refer to the Quick Startup Guide for detailed instructions to install the USB VCP driver and configure the terminal interface.

Connect the 5V power cable to the MaxReader. After the reader has initialized, the buzzer should emit a short chirp. The Power On, FPGA ready, and authenticate LEDs should all be illuminated.

When connecting the MaxReader to the computer's USB port using a mini B USB cable for the first time, the correct USB drivers will need to be installed. Allow Windows to automatically install the required USB serial port drivers. In the event that the serial port drivers cannot install automatically correctly, these drivers may either be located on a CD, or downloaded online at www.ftdichip.com/Drivers/VCP.htm . The current version of the Virtual COM Port (VCP) USB driver is 2.08.14.

Once the USB driver has successfully installed, MaxReader's buzzer should emit a rapid rising tone indicating the USB has been connected to the board. If the USB cable is disconnected from MaxReader, the buzzer should emit a rapid falling tone.

3.3 Terminal Setup

3.3.1 Quick Setup

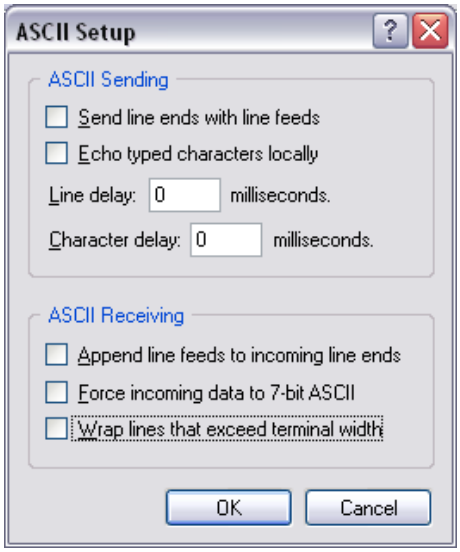
The file `MaxReader.ht` should be included in the software package provided in the MaxArias Development Kit. This file has all the required setting pre-set with the exception of the COM port.

1. Double-click on `MaxReader.ht` to launch Hyperterminal. An informational message will likely pop up indicating that Hyperterminal was not able to connect to COM*n*. Select `OK`.
2. Select the "Properties" icon in Hypertermal (far-right icon below the menu bar). In the "Connect using" drop-down menu, Hyperterminal will list all the available COM ports available to it. Select the COM port that the reader is connected to and select `OK`.
3. Select Hyperterminal's "Call" icon. If the COM port is available, Hyperterminal should now have a link to the reader.
4. Press `ENTER` 3 or 4 times. The "CLI" prompt should appear along with the CLI title bar.

3.3.2 Manual Setup

The computer terminal must be configured as shown below.

COM port settings:	
Baud rate	460,800
Data bits	8
Parity	None
Stop bits	1
Flow control	Hardware
	CTS/RTS



COM port properties:	
Emulation	ANSIW
Input translation	Standard JIS

The MaxReader supports two baud rates: 115,200baud and 460,800 baud. The reader's baud rate is auto-sensing – once the title banner in the CLI has been displayed, the baud rate is fixed until either the power has been cycled to the reader or a hard reset is transmitted through the CLI interface.

4 COMMAND LINE INTERPRETER (CLI) DEFINITIONS

This section defines the CLI instruction syntax. A complete instruction short-form can be found in APPENDIX A.

Commands may be entered into the CLI in either upper-case or lower-case – commands shown in this document shall be displayed in upper-case. Command parameters are shown in braces and are delimited by a space. Most CLI commands that include parameters can be repeated without the necessity of re-entering the parameters. For example, if the data value x1234 is written to the tag's USER memory through the Gen2 interface to addresses x10, x11, and x12, this can be accomplished by the following set of instructions:

G2WRITE 10 1234

G2WRITE 11

G2WRITE 12

Hence, once a parameter has been defined, it remains that value until updated by a subsequent instruction.

Currently, MaxReader is configured to work with a single tag at a time. It is possible to query multiple tags in the reader's RF field, however sub-optimal performance should be expected.

4.1 Control Instructions

MaxReader control instructions are used to adjust the reader's hardware and functional settings. They include enabling RF, buzzer, setting the Gen2 link frequency, controlling the display mode, etc. The following sections provide detail on all CLI control instructions.

CLI Command	Description
ASINIT	Initialize RF front end
RFON	Enable RF
RFOFF	Disable RF
SPKRON	Enable buzzer
SPKROFF	Disable buzzer
SCRIPTON	Enable scripting
SCRIPTOFF	Disable scripting
SCRIPTDELAY	Command delay insertion
RFCHAN	RF channel select
ATTEN	RF output attenuation
LF40K	Link frequency set to 40kHz
LF160K	Link frequency set to 160kHz
M0	Display mode 0
M1	Display mode 1
M2	Display mode 2
M3	Display mode 3
M4	Display mode 4
M5	Display mode 5
M6	Display mode 6
T	Gen2 protocol timer
TESTPAGE	Test port page select
EBVCONV	EBV conversion calculator

4.1.1 ASINIT: Front-End RF Hardware Initialization

Syntax	**ASINIT**
Function	AS3992 initialization
Parameters	None
Cmd Reply	None
Description	Command must always be performed once after startup and prior to any Gen2 command to initialize the AS3992 hardware.

4.1.2 RFON: RF Enable

Syntax	**RFON**
Function	Enable RF
Parameters	None
Cmd Reply	None
Description	Enables the RF carrier output of the reader. This is required for subsequent Gen2 instructions to a RFID tag. The red LED indicator D6 is turned on when the RF output is enabled.

4.1.3 RFOFF: RF Disable

Syntax	**RFOFF**
Function	Disable RF
Parameters	None
Cmd Reply	None
Description	Disables the RF carrier output of the reader. The red LED indicator D6 is turned off when the RF output is disabled. The RF may be disabled when no Gen2 communication is required – note that this will reset the state of the tag since it no longer is receiving RF power.

4.1.4 SPKRON: Enable Buzzer

Syntax	**SPKRON**
Function	Enable reader buzzer
Parameters	None
Cmd Reply	None
Description	Enables the buzzer chirp when a successful Gen2 RFID command has completed. The buzzer is enabled by default.

4.1.5 SPKROFF: Disable Buzzer

Syntax	`SPKROFF`
Function	Disable reader buzzer
Parameters	None
Cmd Reply	None
Description	Disables the buzzer from chirping when a successful Gen2 RFID command has completed.

4.1.6 SCRIPTON: Enable Scripting

Syntax	`SCRIPTON`
Function	Enables scripting
Parameters	None
Cmd Reply	None
Description	When scripting is enabled, the reader will not transmit the entered CLI command text to the terminal until a carriage return is entered. This is required when command scripting or other software is used to transmit CLI commands to MaxReader to prevent buffer over-run.

4.1.7 SCRIPTOFF: Disable Scripting

Syntax	`SCRIPTOFF`
Function	Disables scripting
Parameters	None
Cmd Reply	None
Description	When scripting is disabled, the reader will transmit characters to the terminal as they are received. This is used when the user is manually entering CLI commands on a keyboard. MaxReader will execute CLI commands regardless if scripting is enabled or not.

4.1.8 SCRIPTDELAY: Delaying Scripting Output

Syntax	`SCRIPTDELAY {val}`
Function	CLI command scripting delay insertion
Parameters	

Parameter	Value (hex)	Definition
val	0 – FF	The script delay parameter defines the number of milliseconds before the nCTS signal is re-asserted, allowing a subsequent command to be processed.

Cmd Reply	None
Description	This command inserts a delay after a CLI command has been received. The delay is added to the nCTS signal preventing a subsequent CLI command from being transmitted.

4.1.9 RFCHAN: RF Channel Select

Syntax `RFCHAN {chan}`

Function Define RF channel

Parameters

Parameter	Value (hex)	Definition
chan	1 – 32	The channel identifier defines a channel number between 1 and 50. The parameter passed in the instruction is presented as a hexadecimal value.

Cmd Reply None

Description This command sets the reader's RF frequency channel to 1 of 50 available channels between 902-928Mhz. The channel spacing is 500kHz with the first RF channel starting at 902.5MHz.

4.1.10 ATTEN: Transmit Power Attenuation

Syntax `ATTEN {val}`

Function Transmit RF attenuation – AS3992.

Parameters

Parameter	Value (hex)	Definition
val	0-E	The output attenuation is set through this parameter. The AS3992 output power attenuation defaults to 10dB, or *val*=xA (in hexadecimal).

Cmd Reply None

Description This command applies additional attenuation to the transmit output power. At startup, 10dB of attenuation is applied to the AS3992 output. The transmitted output power is not a linear function – it has the general output from shown in Figure 1 below. The *AS3992 output* may be attenuated from 0dB to 14dB. If maximum output power is desired, the attenuation must be set to 0.

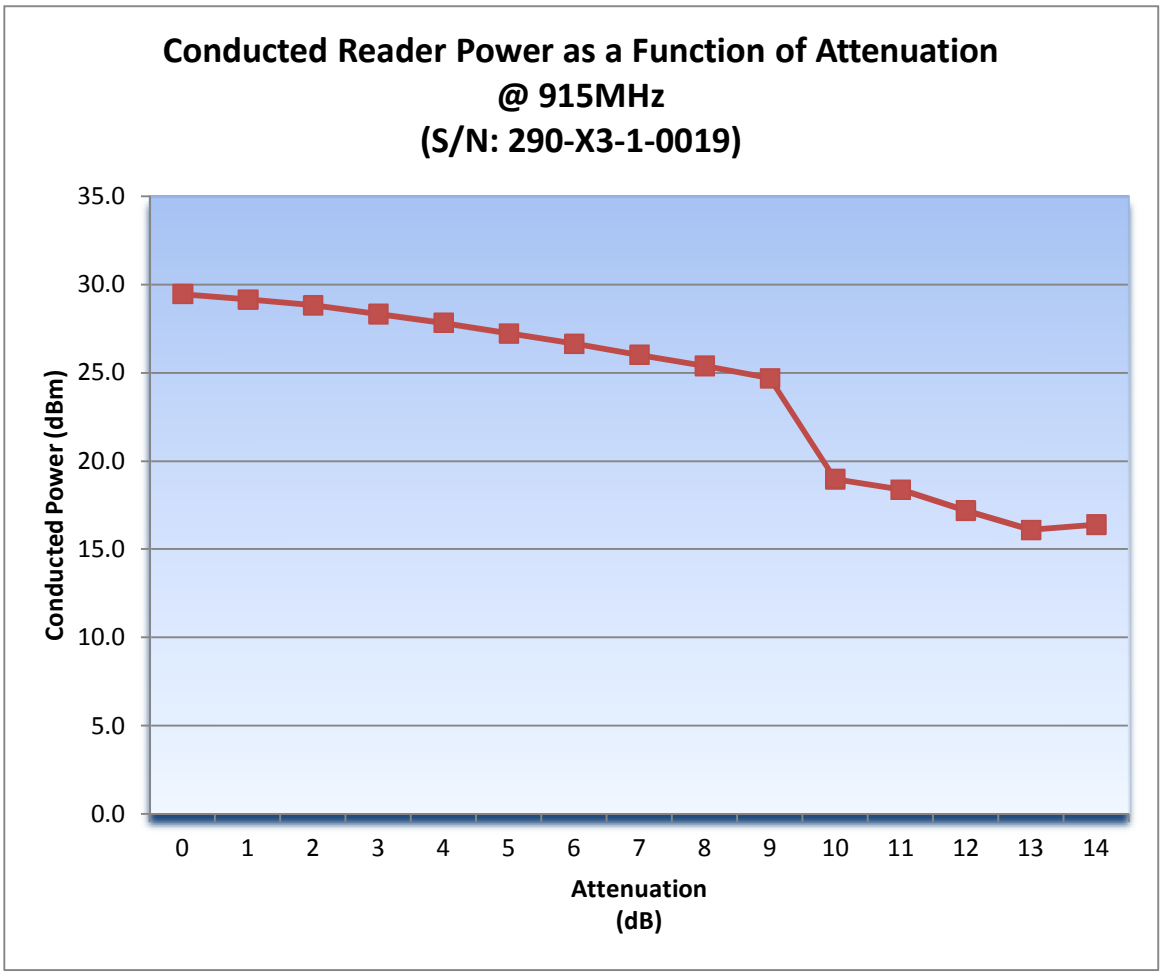


Figure 1 Relationship Between Nominal Output Power and 'Attenuation Setting'

4.1.11 LF40K: Link Frequency 40kHz

Syntax	`LF40K`
Function	Set link frequency to 40kHz
Parameters	None
Cmd Reply	None
Description	Command sets the link frequency (tag-to-reader data rate) to 40kHz. The forward link (reader-to-tag) TARI is set to 25.0µs.

4.1.12 LF160K: Link Frequency 160kHz

Syntax	`LF160K`
Function	Set link frequency to 160kHz
Parameters	None
Cmd Reply	None
Description	Command sets the link frequency (tag-to-reader data rate) to 160kHz. The forward link (reader-to-tag) TARI is set to 6.25µs.

4.1.13 M0: Terminal Display Mode - Raw

Syntax	`M0`
Function	Set display mode to raw
Parameters	None
Cmd Reply	None
Description	No display formatting is performed for Gen2 or DSPI read commands on the computer's terminal.

4.1.14 M1: Terminal Display Mode – Single-Column

Syntax	`M1`
Function	Set display mode to single-column
Parameters	None
Cmd Reply	None
Description	Gen2 and DSPI read data is presented in a single column and presented in the following format: {*address*}:{*data word*}. For Gen2 read commands, the address parameter is the address offset to the base address defined in the read instruction. For DSPI read commands, the address is the physical DSPI memory address being read. The number of data words displayed is defined by the *len* parameter in the respective read instruction.

4.1.15 M2: Terminal Display Mode – 8-Column

Syntax	`M2`
Function	Set display mode to 8-column
Parameters	None
Cmd Reply	None
Description	Gen2 and DSPI read data is presented in 8 columns in the following format: {*addr*}:{*data*}| {*addr*}:{*data*}| … For Gen2 read commands, the address parameter is the address offset to the base address defined in the read instruction. For DSPI read commands, the address is the physical DSPI memory address being read. The number of data words displayed is defined by the *len* parameter in the respective read instruction.

4.1.16 M3: Terminal Display Mode – Enable Line Feeds

Syntax	`M3`
Function	Enable line feed transmission
Parameters	None
Cmd Reply	None
Description	The reader shall transmit a line feed after every carriage return character. Line feeds are enabled by default.

4.1.17 M4: Terminal Display Mode – Disable Line Feeds

Syntax	`M4`
Function	Disable line feed transmission
Parameters	None
Cmd Reply	None
Description	The reader shall terminate a line transmission by transmitting a carriage return only. Line feeds are enabled by default.

4.1.18 M5: Terminal Display Mode – Enable Terminal Prompt

Syntax	`M5`
Function	Enables CLI prompt transmission
Parameters	None
Cmd Reply	None
Description	This mode enables the " CLI:" prompt on the terminal. The terminal prompt is enabled by default.

4.1.19 M6: Terminal Display Mode – Disable Terminal Prompt

Syntax	`M6`
Function	Disables CLI prompt transmission
Parameters	None
Cmd Reply	None
Description	This mode disables the " CLI:" prompt on the terminal. The terminal prompt is enabled by default.

4.1.20 T: Gen2 Protocol Elapsed Time

Syntax	`T`
Function	Gen2 protocol time
Parameters	None
Cmd Reply	`#mmm.uuu ms`
Description	Displays the elapsed time of the last Gen2 commands sequence with a granularity of 1µs. The timer starts at the assertion of the first symbol of the first command from the interrogator and stops when the last tag message has been received. In the event that the last message is from the reader (no tag response), the timer stops upon transmission of the last symbol of the last message. The value remains valid until a subsequent Gen2 CLI command has been issued.

4.1.21 TESTPAG: Test Output Port Page Select

Syntax `TESTPAG {data}`

Function Set test port page select

Parameters

Parameter	Value (hex)	Definition
data	0 - F	The channel identifier defines a channel number between 1 and 50. The parameter passed in the instruction is presented as a hexadecimal value.

Cmd Reply None

Description Outputs internal signals on the test port. Refer to APPENDIX C.

4.1.22 EBVCONV: EBV Conversion

Syntax `EBVCONV {addr}`

Function Gen2 protocol time

Parameters

Parameter	Value (hex)	Definition
addr	0 – 7FF	The *addr* parameter represents the physical USER memory address to be written or read. The last USER memory location with the largest USER memory block sizes is 0x3E7.

Cmd Reply `#addr_ebv`

Description This command provides an quick mechanism between converting from a physical address to a EBV-formatted address. EBV-formatted addresses are required when using the G2WRITEX or G2READX commands to address memory above 0x7F. EBV-formatted data will be presented as a 16-bit (2-byte) value *addr_ebv*.

Example: Converting the address 0x123 to an EBV-formatted address will result in:
addr_ebv = x8223.

4.1.23 Reader Reset

Syntax `W 0 FF`

Function Generate reader reset

Parameters

Cmd Reply

Description This command will generate a reset which will reset the reader's FPGA registers to an initial state

4.2 Query Instructions

CLI query instructions perform a Gen2 command sequence to transfer a tag from the *arbitration* state to an *open* state. A tag must continue to be powered by the RF field and must be in the *open* state to perform additional Gen2 access commands, such as read, write, lock, etc. The CLI query commands defined in this section include single tag queries as well as continuous looping query command used for tag and RF channel detection.

CLI Command	Description
G2FASTSCAN	Continuous RF channel scan
G2AUTOSCAN	Continuous RF channel scan
G2SCAN	Continuous RF channel scan
G2CQRY	Continuous tag query
G2QRY	Single tag query
G2RQRY	Single tag re-query
G2QRYPASS	Query pass count
G2QRYFAIL	Query fail count

4.2.1 G2SCAN: Gen2 Continuous Tag Query – RF Channel Polling

Syntax `G2SCAN`

Function Perform a continuous tag query over all RF channels

Parameters None

Cmd Reply `:aaaa_bbbbccccddddeeeeffffgggg_hhhh`

Parameter	Definition
aaaa	EPC Protocol Control word (PC)
bbbb	EPC word 0
cccc	EPC word 1
dddd	EPC word 2
eeee	EPC word 3
ffff	EPC word 4
gggg	EPC word 5
hhhh	EPC CRC

NOTES:
1. The actual number of EPC words returned is defined in the Protocol Control word. The number of words can be any value between 0 and 6. The shown configuration (number of words = 6) is typical. Refer to the G2WRITEEPC command (section 4.4.3) for detail on the defined number of EPC data words.
2. The PC and CRC words are delimited from the EPC identifier with underscore characters on the computer's terminal.
3. No response is returned if a tag query fails.

Description This mode performs a continuous tag query function over all 50 RF channels in a cyclical manner. For each RF channel, a single tag query is performed after which the RF channel is incremented (RF channel 50 rolls over to channel 1). When a tag is successfully queried, the EPC identifier is displayed on the terminal and the buzzer will output a short high-pitched chirp (if enabled).

The continuous query scanning mode is terminated by either pressing the escape <ESC> key, space bar, or ENTER key a single time. This mode must be terminated prior to executing other Gen2 commands. Depending on the state of the reader when the escape key is hit, the RF may be on or off. In the event that the RF is off and additional tag access commands are required, the following must be performed:

1. The reader will require the RF be re-enabled: CLI command = RFON.
2. Perform a single query on the tag: CLI command = G2QRY.
3. Perform any additional access commands.

4.2.2 G2FASTSCAN: Gen2 Continuous Tag Query – Fast RF Channel Polling

Syntax `G2FASTSCAN`

Function Perform a continuous tag query over all RF channels

Parameters None

Cmd Reply `:aaaa_bbbbccccddddeeeeffffgggg_hhhh`

Parameter	Definition
aaaa	EPC Protocol Control word (PC)
bbbb	EPC word 0
cccc	EPC word 1
dddd	EPC word 2
eeee	EPC word 3
ffff	EPC word 4
gggg	EPC word 5
hhhh	EPC CRC

NOTES:
1. The actual number of EPC words returned is defined in the Protocol Control word. The number of words can be any value between 0 and 6. The shown configuration (number of words = 6) is typical. Refer to the G2WRITEEPC command (section 4.4.3) for detail on the defined number of EPC data words.
2. The PC and CRC words are delimited from the EPC identifier with underscore characters on the computer's terminal.
3. No response is returned if a tag query fails.

Description This mode performs a continuous tag query function over all 50 RF channels in a similar manner as the G2SCAN command, however failing channels are quickly skipped over. For each RF channel, a single tag query is performed after which the RF channel is incremented (RF channel 50 rolls over to channel 1). When a tag is successfully queried, the EPC identifier is displayed on the terminal and the buzzer will output a short high-pitched chirp (if enabled).

The continuous query scanning mode is terminated by either pressing the escape <ESC> key, space bar, or ENTER key a single time. This mode must be terminated prior to executing other Gen2 commands. Depending on the state of the reader when the escape key is hit, the RF may be on or off. In the event that the RF is off and additional tag access commands are required, the following must be performed:

4. The reader will require the RF be re-enabled: CLI command = RFON.
5. Perform a single query on the tag: CLI command = G2QRY.
6. Perform any additional access commands.

4.2.3 G2AUTOSCAN: Gen2 Continuous Tag Query – Automatic Channel Removal

Syntax	`G2AUTOSCAN`
Function	Perform a continuous tag query over all RF channels
Parameters	None
Cmd Reply	`:aaaa_bbbbccccddddeeeeffffgggg_hhhh`

Parameter	Definition
aaaa	EPC Protocol Control word (PC)
bbbb	EPC word 0
cccc	EPC word 1
dddd	EPC word 2
eeee	EPC word 3
ffff	EPC word 4
gggg	EPC word 5
hhhh	EPC CRC

NOTES:
1. The actual number of EPC words returned is defined in the Protocol Control word. The number of words can be any value between 0 and 6. The shown configuration (number of words = 6) is typical. Refer to the G2WRITEEPC command (section 4.4.3) for detail on the defined number of EPC data words.
2. The PC and CRC words are delimited from the EPC identifier with underscore characters.
3. No response is returned if a tag query fails.

Description

This mode performs a continuous tag query function over all 50 RF channels in a cyclical manner while systematically removing failed RF channels. For each enabled RF channel, a single tag query is performed after which the RF channel is incremented (RF channel 50 rolls over to channel 1). When a tag is successfully queried, the EPC identifier is displayed on the terminal and the buzzer will output a short high-pitched chirp (if enabled).

A failed RF channel is defined herein as a channel where three consecutive tag query cycles have been unsuccessful on the same channel. Once a channel has been deemed as 'bad', it is removed from the RF channel lineup. The initial number of RF channels (num_chan_0) is set to 50. Upon execution of the G2AUTOSCAN command, a failed RF channel is removed resulting in the number of channels being decremented: $num\_chan_n = num\_chan_{n+1} - 1$. In the event that $num\_chan_n = 0$, the entire RF channel lineup is re-enabled, allowing the procedure to re-start.

The continuous query scanning mode is terminated by either pressing the escape <ESC> key, space bar, or ENTER key a single time This mode must be terminated prior to executing other Gen2 commands. Depending on the state of the reader when the escape key is hit, the RF may be on or off. In the event that the RF is off and additional tag access commands are required, the following must be performed:

1. The reader will require the RF be re-enabled: CLI command = RFON.
2. Perform a single query on the tag: CLI command = G2QRY.
3. Perform any additional access commands.

4.2.4 G2CQRY: Gen2 Continuous Tag Query – Single RF Channel

Syntax `G2CQRY {Q} {sel} {session} {target} {M}`

Function Perform a continuous tag query over a single RF channel

Parameters

Parameter	Value (hex)	Definition
Q	0-F	Slot count parameter (default *Q*=0)
sel	0-4	Gen2 Query command *sel* parameter (default *sel*=0)
session	0-3	Gen2 Query command *session* parameter (default *session*=2)
target	0-1	Gen2 Query command *target* parameter (default *target*=0)
M	0-3	Gen2 Query command *M* parameter (default *M*=0)

Cmd Reply `:aaaa_bbbbccccddddeeeeffffgggg_hhhh`

Parameter	Definition
aaaa	EPC Protocol Control word (PC)
bbbb	EPC word 0
cccc	EPC word 1
dddd	EPC word 2
eeee	EPC word 3
ffff	EPC word 4
gggg	EPC word 5
hhhh	EPC CRC

NOTES:
1. The actual number of EPC words returned is defined in the Protocol Control word. The number of words can be any value between 0 and 6. The shown configuration (number of words = 6) is typical. Refer to the G2WRITEEPC command (section 4.4.3) for detail on the defined number of EPC data words.
2. The PC and CRC words are delimited from the EPC identifier with underscore characters.
3. No response is returned if a tag query fails.

Description This mode performs a continuous tag query function on a single RF channel. When a tag is successfully queried, the EPC identifier is displayed on the terminal and the buzzer will output a short high-pitched chirp (if enabled).

The current RF channel is either the last retained RF channel from one of the two RF channel scanning query modes or manually set through the RFCHAN command. The current RF channel may be read through the following direct debug command:

```
R 7A
```

The returned value will be a value between 1 (0x01) and 50 (0x32).

The continuous query scanning mode is terminated by either pressing the escape <ESC> key, space bar, or ENTER key a single time. This mode must be terminated prior to executing other Gen2 commands. When the continuous query mode is terminated, the RF will remain enabled.

4.2.5 G2QRY: Single Gen2 Tag Query

Syntax `G2QRY {Q} {sel} {session} {target} {M}`

Function Perform a single tag query

Parameters

Parameter	Value (hex)	Definition
Q	0-F	Slot count parameter (default *Q*=0)
sel	0-4	Gen2 Query command *sel* parameter (default *sel*=0)
session	0-3	Gen2 Query command *session* parameter (default *session*=2)
target	0-1	Gen2 Query command *target* parameter (default *target*=0)
M	0-3	Gen2 Query command *M* parameter (default *M*=0)

Cmd Reply `:aaaa_bbbbccccddddeeeeffffgggg_hhhh`

Parameter	Definition
aaaa	EPC Protocol Control word (PC)
bbbb	EPC word 0
cccc	EPC word 1
dddd	EPC word 2
eeee	EPC word 3
ffff	EPC word 4
gggg	EPC word 5
hhhh	EPC CRC

NOTES:
1. The actual number of EPC words returned is defined in the Protocol Control word. The number of words can be any value between 0 and 6. The shown configuration (number of words = 6) is typical. Refer to the G2WRITEEPC command (section 4.4.3) for detail on the defined number of EPC data words.
2. The PC and CRC words are delimited from the EPC identifier with underscore characters.
3. No response is returned if a tag query fails.

Description This mode performs a single tag query function on a single RF channel. When a tag is successfully queried, the EPC identifier is displayed on the terminal and the buzzer will output a short high-pitched chirp (if enabled).

The current RF channel is either the last retained RF channel from one of the three continuous query modes or manually set through the RFCHAN command. The current RF channel may be read through the following direct debug command:

```
R 7A
```

The returned value will be a value between 1 (0x01) and 50 (0x32).

Regardless of the success of the command, the RF will remain enabled.

4.2.6 G2RQRY: Single Gen2 Tag Re-Query

Syntax `G2RQRY {Q} {sel} {session} {target} {M}`

Function Perform a single tag re-query

Parameters

Parameter	Value (hex)	Definition
Q	0-F	Slot count parameter (default *Q*=0)
sel	0-4	Gen2 Query command *sel* parameter (default *sel*=0)
session	0-3	Gen2 Query command *session* parameter (default *session*=2)
target	0-1	Gen2 Query command *target* parameter (default *target*=0)
M	0-3	Gen2 Query command *M* parameter (default *M*=0)

Cmd Reply `:aaaa_bbbbccccddddeeeeffffgggg_hhhh`

Parameter	Definition
aaaa	EPC Protocol Control word (PC)
bbbb	EPC word 0
cccc	EPC word 1
dddd	EPC word 2
eeee	EPC word 3
ffff	EPC word 4
gggg	EPC word 5
hhhh	EPC CRC

NOTES:
1. The actual number of EPC words returned is defined in the Protocol Control word. The number of words can be any value between 0 and 6. The shown configuration (number of words = 6) is typical. Refer to the G2WRITEEPC command (section 4.4.3) for detail on the defined number of EPC data words.
2. The PC and CRC words are delimited from the EPC identifier with underscore characters.
3. No response is returned if a tag query fails.

Description This mode performs a single tag query function on a single RF channel with the exception that no Gen2 Select command is included in the tag query command sequence. When a tag is successfully queried, the EPC identifier is displayed on the terminal and the buzzer will output a short high-pitched chirp (if enabled).

The omission of the Select Gen2 command for this CLI command effectively allows additional tags in the field to be queried by not *selecting* all tags to participate in the subsequent query round – i.e. a tag that has been queried in a previous query round will not respond to the CLI G2RQRY command.

The current RF channel is either the last retained RF channel from one of the three continuous query modes or manually set through the RFCHAN command. The current RF channel may be read through the following direct debug command:

```
R 7A
```

The returned value will be a value between 1 (0x01) and 50 (0x32).

Regardless of the success of the command, the RF will remain enabled.

4.2.7 G2QRYPASS: Gen2 Continuous-Query Pass Counter

Syntax `G2QRYPASS`

Function Counts the number of successful tag queries

Parameters None

Cmd Reply `:pppp`

Parameter	Definition
pppp	16-bit pass count

Description Counts the number of successful tag queries when any one of the three continuous tag query modes is initiated. The counter is initialized upon entry into a continuous query mode.

NOTE: At the current time, the fail counter can only be read through the following direct debug mode command:
`R 58`

4.2.8 G2QRYFAIL: Gen2 Continuous-Query Fail Counter

Syntax `G2QRYFAIL`

Function Counts the number of failed tag queries

Parameters None

Cmd Reply `:ffff`

Parameter	Definition
ffff	16-bit fail count

Description Counts the number of failed tag queries when any one of the three continuous tag query modes is initiated. The counter is initialized upon entry into a continuous query mode.

NOTE: At the current time, the fail counter can only be read through the following direct debug mode command:
`R 59`

4.3 Read Instructions

Gen2 read instructions are performed through the CLI by using the target memory bank (or register) in the name of the CLI command. It is inferred that commands that do not include the Gen2 memory bank name are accessing the USER memory bank. In addition, read commands may specify single or multiple words, MaxArias registers, or use extended EBV-formatted addressing.

CLI Command	Description
G2READ	Gen2 read – single-byte addressing
G2READX	Gen2 read – EBV-formatted addressing
G2READRSVD	Gen2 RESERVED memory bank read
G2READEPC	Gen2 EPC memory bank read
G2READTID	Gen2 TID memory bank read
G2READUSER	Gen2 USER memory bank read
G2READCNTL	Gen2 MaxArias Control/Status read
G2READSTRDADDR	Gen2 MaxArias Working Stored Address read

4.3.1 G2READ: Gen2 Read

Syntax `G2READ {addr} {len}`

Function Gen2 USER memory bank read command

Parameters

Parameter	Value (hex)	Definition
addr	00-7F	
len	00-FF	

Cmd Reply $data_0$, $data_1$, …, $data_{len-2}$, $data_{len-1}$, *Handle*, *CRC16*

Description Performs a standard Gen2 read command on the USER memory bank using a single 8-bit address parameter (note: Gen2 read instructions of the other memory banks are performed with other CLI commands described later in this section). The G2READ command has 2 parameters: address (*addr*) and length (*len*). The address parameter defines the read instruction starting address and may not be larger than x7F; the length parameter defines the number of words, with the first word being addressed in the command's address parameter – the length parameter can take on any value between x0 and xFF.

NOTE:

1. Once the address and length parameters have been entered in an instruction, subsequent read instructions may be repeated without re-entering the parameters – entering the command on its own will produce the same result. In addition, a subsequent read instruction may only define the address – the previously defined length parameter will continue to be used.
2. As defined in the Gen2 standard, if the length parameter is set to null (0x00), the tag shall respond with read data starting at the address defined in the read instruction, and continue to the end of the memory bank.

4.3.2 G2READX: Gen2 Read – Extended Addressing

Syntax `G2READX {addr} {len}`

Function Gen2 USER memory bank read command – extended EBV addressing

Parameters

Parameter	Value (hex)	Definition
addr	8000-FF7F	
len	00-FF	

Cmd Reply $data_0$, $data_1$, …, $data_{len-2}$, $data_{len-1}$, *Handle*, *CRC16*

Description Performs a standard Gen2 read command on the USER memory bank using a 2-byte EBV-formatted address parameter (note: Gen2 read instructions of the other memory banks are performed with other CLI commands described in this section). The G2READX command has 2 parameters: address (*addr*) and length (*len*). The address parameter defines the read instruction starting address presented as a 2-byte EBV-formatted address (refer to APPENDIX B to convert standard addressing to EBV-formatted addressing); the length parameter defines the number of words, with the first word being addressed in the command's address parameter – the length parameter can take on any value between x0 and xFF. This command can be used to address any USER memory location (the G2READ command only addresses USER memory space x00 – x7F).

NOTE:

1. Once the address and length parameters have been entered in an instruction, subsequent read instructions may be repeated without re-entering the parameters – entering the command on its own will produce the same result. In addition, a subsequent read instruction may only define the address – the previously defined length parameter will continue to be used.
2. As defined in the Gen2 standard, if the length parameter is set to null (0x00), the tag shall respond with read data starting at the address defined in the read instruction, and continue to the end of the memory bank.

4.3.3 G2READRSVD: Gen2 Read RESERVED Memory Bank

Syntax `G2READRSVD`

Function Gen2 RESERVED memory bank read command

Parameters None

Cmd Reply *kill_pwd0*, *kill_pwd1*, *access_pwd0*, *access_pwd1*, *Handle*, *CRC16*

Description Performs a standard Gen2 read command of the entire RESERVED memory bank. The RESERVED memory bank stored the kill and access passwords for the tag. In the event that the RESERVED memory bank is locked, no command reply will be displayed as specified in the Gen2 standard.

4.3.4 G2READEPC: Gen2 Read EPC Memory Bank

Syntax `G2READEPC`

Function Gen2 EPC memory bank read command

Parameters None

Cmd Reply *EPC_CRC16*, *PC*, *epc0*, *epc1*, *epc2*, *epc3*, *epc4*, *epc5*, *epc6*, *epc7*, *Handle*, *CRC16*

Description Performs a standard Gen2 read command of the entire EPC memory bank. The command reply will display 8 EPC data words for all current MaxArias tags.

4.3.5 G2READTID: Gen2 Read TID Memory Bank

Syntax	**G2READTID**
Function	Gen2 TID memory bank read command
Parameters	None
Cmd Reply	*tid0, tid1, tid2, tid3, Handle, CRC16*
Description	Performs a standard Gen2 read command of the entire TID memory bank. The first two TID words of a MaxArias WM72016 tag shall be: xE201_6216. This shall indicate: 1. Ramtron's "Mask Designer Identification" = x016. 2. The encoded WM72016 part number = x216. 3. Ramtron's serial number is encoded in *tid2* and *tid3*.

4.3.6 G2READUSER: Gen2 Read USER Memory Bank

Syntax	**G2READUSER**
Function	Gen2 USER memory bank read command
Parameters	None
Cmd Reply	$data_0$, $data_1$, …, $data_{n-2}$, $data_{n-1}$, *Handle, CRC16*
Description	Performs a standard Gen2 read command of the entire USER memory bank. This command functions identically to a standard G2READ command with both the address and length parameters set to zero. The number of data words *n* returned for a MaxArias tag is dependent on the USER memory bank block size setting (set by the Control/Status register – refer to the MaxArias data sheet for additional detail). For example, *n*=931 (block size = 1 word/block), *n*=963 (block size = 2 words/block), etc.

4.3.7 G2READCNTL: Gen2 Read MaxArias Control/Status

Syntax	**G2READCNTL**
Function	MaxArias Control/Status read command
Parameters	None
Cmd Reply	*data, Handle, CRC16*
Description	Performs a standard Gen2 read command of the MaxArias Control/Status register. The parsing of the Control/Status data word is shown in the MaxArias datasheet. Key control register bits are high-lighted below. Particular attention should be given to the AINC (auto-increment) control bit – notably its effect on WM72016 addressing shifting when using unaddressed writes or block-writes. These two types of write instructions use the address contents of the working stored address register to write data to WM72016 memory.

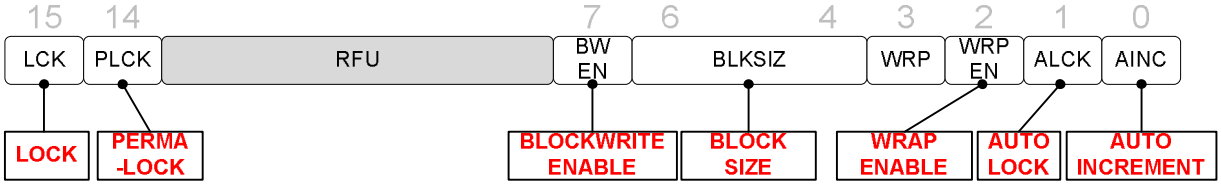


Figure 2 WM72016 Control/Status Register

4.3.8 G2READSTRDADDR: Gen2 Read MaxArias Working Stored Address

Syntax	`G2READSTRDADDR`
Function	MaxArias Working Stored Address read command
Parameters	None
Cmd Reply	*data*, *Handle*, *CRC16*
Description	Performs a standard Gen2 read command of the MaxArias Working Stored Address register. The parsing of the Working Stored Address data word is shown in the MaxArias datasheet.

4.4 Write Instructions

Gen2 write instructions are performed through the CLI by using the target memory bank (or register) in the name of the CLI command. It is inferred that commands that do not include the Gen2 memory bank name are accessing the USER memory bank.

CLI Command	Description
G2WRITE	Gen2 single address byte write
G2WRITEX	Gen2 EBV-address write
G2WRITEEPC	Gen2 EPC memory bank write
G2WRITEACC	Gen2 access password write
G2WRITEKILL	Gen2 kill password write
G2WRITECNTL	Gen2 MaxArias Control/Status write
G2WRITESTRDADDR	Gen2 MaxArias Working Stored Address write
G2UWRITE	Gen2 MaxArias unaddressed write

4.4.1 G2WRITE: Gen2 Write

Syntax `G2WRITE {addr} {data}`

Function Gen2 USER memory bank write command

Parameters

Parameter	Value (hex)	Definition
addr	00-7F	USER memory bank single-byte address
data	0-FFFF	Data to be written to tag's memory

Cmd Reply `:WR_OK:handle`

Description Performs a standard Gen2 write command on the USER memory bank using a single 8-bit address parameter (note: Gen2 write instructions on the other memory banks are performed with other CLI commands described later in this section). The G2WRITE command has 2 parameters: address (*addr*) and data (*data*). The address parameter defines the write instruction starting address and may not be larger than x7F; the data parameter defines the 16-bit data word written to USER memory. A successful Gen2 write cycle is indicated by an audible chirp from the buzzer on the reader as well as a "write OK" status message and the tag's handle displayed on the terminal.

NOTE:

1. Once the address and data parameters have been entered in an instruction, subsequent write instructions may be repeated without re-entering the parameters – entering the command on its own will produce the same result. In addition, a subsequent write instruction may only define the address – the previously defined data parameter will continue to be used.
2. The data parameter is expressed as a hexadecimal value up to 4 characters in length. For example, the value *x000B* may be expressed as *xB*, *x0B*, *x00B*, or *x000B*.

4.4.2 G2WRITEX: Gen2 Write – Extended Addressing

Syntax `G2WRITEX {addr} {data}`

Function Gen2 USER memory bank write command – extended addressing

Parameters

Parameter	Value (hex)	Definition
addr	8000-FF7F	USER memory bank EBV address
data	0-FFFF	Data to be written to tag's memory

Cmd Reply `:WR_OK:handle`

Description Performs a standard Gen2 write command on the USER memory bank using a 2-byte EBV-formatted address parameter (note: Gen2 write instructions on the other memory banks are performed with other CLI commands described later in this section). The G2WRITEX command has 2 parameters: address (*addr*) and data (*data*). The address parameter defines the write instruction starting address and must be presented as a 4-character, 2 byte EBV-formatted address (refer to APPENDIX B to convert standard addressing to EBV-formatted addressing); the data parameter defines the 16-bit data word written to USER memory. The G2WRITEX command may be used to write any unlocked USER memory location. A successful Gen2 write cycle is indicated by an audible chirp from the buzzer on the reader as well as a "write OK" status message and the tag's handle displayed on the terminal.

NOTE:

1. Once the address and data parameters have been entered in an instruction, subsequent write instructions may be repeated without re-entering the parameters – entering the command on its own will produce the same result. In addition, a subsequent write instruction may only define the address – the previously defined data parameter will continue to be used.
2. The data parameter is expressed as a hexadecimal value up to 4 characters in length. For example, the value *x000B* may be expressed as *xB*, *x0B*, *x00B*, or *x000B*.

4.4.3 G2WRITEEPC: Gen2 Write EPC Identifier

Syntax `G2WRITEEPC {pc} {epc1} {epc2} {epc3} {epc4} {epc5} {epc6}`

Function Gen2 EPC memory bank write command

Parameters

Parameter	Value	Definition
pc	See next	The Protocol Control (PC) word in part defines the number of words in the EPC identifier – the first 5 bits of the PC word are used for this purpose and may not exceed a value of 6. **EPC Size** / **# words** / **PC** b00000 / 0 / x0000 b00001 / 1 / x0800 b00010 / 2 / x1000 b00011 / 3 / x1800 b00100 / 4 / x2000 b00101 / 5 / x2800 b00110 / 6 / x3000
epc1	user-defined	EPC identifier – word 1
epc2	user-defined	EPC identifier – word 2
epc3	user-defined	EPC identifier – word 3
epc4	user-defined	EPC identifier – word 4
epc5	user-defined	EPC identifier – word 5
epc6	user-defined	EPC identifier – word 6

Cmd Reply None

Description Performs a standard Gen2 write command on the EPC memory bank. The G2WRITEEPC command may be executed with one or more parameters (up to the *epc6* parameter). As with other CLI, commands, parameters are stored in the reader – subsequent G2WRITEEPC commands that do not include one or more parameters will have the respective parameters written with the previously set value(s).

4.4.4 G2WRITEACC: Gen2 Write Access Passwords

Syntax `G2WRITEACC {pwd1} {pwd2}`

Function Gen2 RESERVED memory bank write command of the ACCESS passwords.

Parameters

Parameter	Value	Definition
pwd1	user-defined	Access password – most significant word
pwd2	user-defined	Access password – least significant word

Cmd Reply None

Description Performs a standard Gen2 write command on the RESERVED memory bank to write the two 16-bit ACCESS passwords.

4.4.5 G2WRITEKILL: Gen2 Write Kill Passwords

Syntax `G2WRITEKILL {pwd1} {pwd2}`

Function Gen2 RESERVED memory bank write command of the KILL passwords.

Parameters

Parameter	Value	Definition
pwd1	user-defined	Kill password – most significant word
pwd2	user-defined	Kill password – least significant word

Cmd Reply None

Description Performs a standard Gen2 write command on the RESERVED memory bank to write the two 16-bit KILL passwords.

4.4.6 G2WRITECNTL: Gen2 Write MaxArias Control/Status

Syntax `G2WRITECNTL {data}`

Function MaxArias Control/Status write command

Parameters

Parameter	Value	Definition
data	user-defined	

Cmd Reply None

Description Performs a standard Gen2 write command to the MaxArias Control/Status register. The MaxArias Control/Status register is used for the following purpose:

1. Enables Gen2 blockwrite commands
2. Sets the USER memory bank block size
3. Displays the memory wrap status
4. Sets wrap enable control
5. Sets auto-lock enable control
6. Sets auto-increment enable

For additional detail on this functionality, please refer to the MaxArias datasheet.

4.4.7 G2WRITESTRDADDR: Gen2 Write MaxArias Working Stored Address

Syntax `G2WRITESTRDADDR {addr}`

Function MaxArias Working Stored Address write command

Parameters

Parameter	Value	Definition
addr	user-defined	The address parameter must not be less than x0005 and not greater than the USER memory size.

Cmd Reply None

Description Performs a standard Gen2 write command to the MaxArias Working Stored Address register. The MaxArias Working Stored Address register is used for unaddressed write cycles and Gen2 blockwrite instructions. In addition, the *Initial Stored Address* can be set using this command (refer to the MaxArias datasheet).

4.4.8

4.4.9 G2UWRITE: Gen2 Write MaxArias Unaddressed Write

Syntax `G2UWRITE {data}`

Function MaxArias Unaddressed write command

Parameters

Parameter	Value	Definition
data	user-defined	

Cmd Reply None

Description Performs a MaxArias unaddressed write command using the address pointer stored in the MaxArias Working Stored Address register.

4.4.10 G2BLKWRITE: Gen2 MaxArias Blockwrite Command

The MaxArias blockwrite instruction allows multiple data words (up to 127) to be written to the USER memory bank with a single instruction. The MaxArias block-write instructions defined here are designed to support the block-write syntax specifically for MaxArias tags.

Syntax `G2BLKWRITE {addr} {data0} {data1} {…} {data_n-2} {data_n-1}`

Function Gen2 MaxArias *n*-word blockwrite

Parameters

Parameter	Value (hex)	Definition
addr	5-3E7	The maximum address is defined by the USER memory bank block size (the maximum address defined is given for a block size = 128 words/block).
data	user-defined	Up to 127 data words may be defined, with a space delimiting each word.

Cmd Reply None

Description Performs a MaxArias Gen2 blockwrite command with a 1 to 127 data word payload. A MaxArias blockwrite command shall write data to the memory address pointer defined by the Working Stored Address register (refer to the MaxArias data specifically regarding the affect of the auto-increment bit on the blockwrite starting address using the Working Stored Address). The CLI G2BLKWRITE command provides a convenient method with a single command to write multiple data words without the requirement of manually updating the Working Stored Address register.

NOTE: When using the G2BLKWRITE command to write to the WM72016, an address shift of the data will occur if the tag's AUTOINCR bit in the CNTL/STAT register is set. To avoid the shift, do one of the following:

1. *Clear the tag's AUTOINCR control bit, or*
2. *Account for the address shift in the G2BLKWRITE 'addr' parameter by subtracting 1.*

4.5 Memory Bank Lock Instructions

The Gen2 standard defines two methodologies of locking tag memory:

1. Entire memory bank locking, and
2. USER memory bank block locking

For each memory bank (with the exception of the TID memory bank that is already permanently locked by the manufacturer), three commands are available to manipulate the lock status of the respective memory bank. These commands are:

G2LOCK*

G2PLOCK*

G2ULOCK*

Each memory bank has 2 lock control bits: a *lock* control bit, and a *permalock* control bit. The *lock* control bit is manipulated by through the use of the G2LOCK* and G2ULOCK* commands while the permalock control bit may only be permanently set through the G2PLOCK* command. Locked tags that are not permanently set can be unlocked from the tag's *secured* state.

4.5.1 G2LOCKKILL: Set Kill Lock

Syntax	`G2LOCKKILL`
Function	Sets the RESERVED memory bank kill password lock bit
Parameters	None
Description	Performs the lock function on the RESERVED memory bank kill password. Prior to executing the G2LOCKKILL command, the tag must be in the *secured* state. A tag whose access password is null will automatically advance from the *open* state to the *secured* state during arbitration; a tag whose access password is non-zero remains in the *open* state until the tag has successfully transitioned to the *secured* state – this is done through the G2ACC command accompanied with the valid respective passwords. Once the kill password has been locked, it will no longer be readable by the interrogator. The kill password may be unlocked while the tag is in the *secured* state through the G2ULOCKKILL command.

4.5.2 G2PLOCKKILL: Set Kill Permalock

Syntax	`G2PLOCKKILL`
Function	Sets the RESERVED memory bank kill password permalock bit
Parameters	None
Description	Sets the permalock bit of the RESERVED memory bank kill password. Once the permalock bit has been asserted, the state of the kill password lock bit may no longer be altered, whether a logic 0 or a logic 1.

4.5.3 G2ULOCKKILL: Clear Kill Lock

Syntax	`G2ULOCKKILL`
Function	Clears the RESERVED memory bank kill password lock bit
Parameters	None
Description	Clears the lock bit of the RESERVED memory bank kill password. Prior to executing the G2ULOCKKILL command, the tag must be in the *secured* state. A tag whose access password is null will automatically advance from the *open* state to the *secured* state during arbitration; a tag whose access password is non-zero remains in the *open* state until the tag has successfully transitioned to the *secured* state – this is done through the G2ACC command and the valid respective passwords. Once the kill password has been unlocked, it will be readable by the interrogator in the *open* or *secured* states. The kill password may be locked while the tag is in the *secured* state through the G2LOCKKILL command.

4.5.4 G2LOCKACC: Set Access Lock

Syntax	`G2LOCKACC`
Function	Sets the RESERVED memory bank access password lock bit
Parameters	None
Description	Performs the lock function on the RESERVED memory bank access password. Prior to executing the G2LOCKACC command, the tag must be in the *secured* state. A tag whose access password is null will automatically advance from the *open* state to the *secured* state during arbitration; a tag whose access password is non-zero remains in the *open* state until the tag has successfully transitioned to the *secured* state – this is done through the G2ACC command and the valid respective passwords. Once the access password has been locked, it will no longer be readable by the interrogator. The access password may be unlocked while the tag is in the *secured* state through the G2ULOCKACC command.

4.5.5 G2PLOCKACC: Set Access Permalock

Syntax	`G2PLOCKACC`
Function	Sets the RESERVED memory bank access password permalock bit
Parameters	None
Description	Sets the permalock bit of the RESERVED memory bank access password. Once the permalock bit has been asserted, the state of the access password lock bit may no longer be altered, whether a logic 0 or a logic 1.

4.5.6 G2ULOCKACC: Clear Access Lock

Syntax	`G2ULOCKACC`
Function	Clears the RESERVED memory bank access password lock bit
Parameters	None
Description	Clears the lock bit of the RESERVED memory bank access password. Prior to executing the G2ULOCKACC command, the tag must be in the *secured* state. A tag whose access password is null will automatically advance from the *open* state to the *secured* state during arbitration; a tag whose access password is non-zero remains in the *open* state until the tag has successfully transitioned to the *secured* state – this is done through the G2ACC command and the valid respective passwords. Once the access password has been unlocked, it will be readable by the interrogator in the *open* or *secured* states. The access password may be locked while the tag is in the *secured* state through the G2LOCKACC command.

4.5.7 G2LOCKEPC: Set EPC Lock

Syntax	`G2LOCKEPC`
Function	Sets the EPC memory bank lock bit
Parameters	None
Description	Performs the lock function on the EPC memory bank. Prior to executing the G2LOCKEPC command, the tag must be in the *secured* state. A tag whose access password is null will automatically advance from the *open* state to the *secured* state during arbitration; a tag whose access password is non-zero remains in the *open* state until the tag has successfully transitioned to the *secured* state – this is done through the G2ACC command and the valid respective passwords. Once locked, the EPC memory bank will still be readable by the interrogator. The EPC memory bank may be unlocked while the tag is in the *secured* state through the G2ULOCKEPC command.

4.5.8 G2PLOCKEPC: Set EPC Permalock

Syntax	`G2PLOCKEPC`
Function	Sets the EPC memory bank permalock bit
Parameters	None
Description	Sets the permalock bit of the EPC memory bank. Once the permalock bit has been asserted, the state of the EPC memory bank lock bit may no longer be altered, whether a logic 0 or a logic 1.

4.5.9 G2ULOCKEPC: Clear EPC Lock

Syntax	`G2ULOCKEPC`
Function	Clears the EPC memory bank lock bit
Parameters	None
Description	Clears the lock bit of the EPC memory bank. Prior to executing the G2ULOCKEPC command, the tag must be in the *secured* state. A tag whose access password is null will automatically advance from the *open* state to the *secured* state during arbitration; a tag whose access password is non-zero remains in the *open* state until the tag has successfully transitioned to the *secured* state – this is done through the G2ACC command and the valid respective passwords.

4.5.10 G2LOCKUSER: Set USER Lock

Syntax	`G2LOCKUSER`
Function	Sets the USER memory bank lock bit
Parameters	None
Description	Performs the lock function on the entire USER memory bank. Prior to executing the G2LOCKUSER command, the tag must be in the *secured* state. A tag whose access password is null will automatically advance from the *open* state to the *secured* state during arbitration; a tag whose access password is non-zero remains in the *open* state until the tag has successfully transitioned to the *secured* state – this is done through the G2ACC command and the valid respective passwords. Once locked, the USER memory bank will still be readable by the interrogator. The USER memory bank may be unlocked while the tag is in the *secured* state through the G2ULOCKUSER command.

4.5.11 G2PLOCKUSER: Set USER Permalock

Syntax	`G2PLOCKUSER`
Function	Sets the USER memory bank permalock bit
Parameters	None
Description	Sets the permalock bit of the USER memory bank. Once the permalock bit has been asserted, the state of the USER memory bank lock bit may no longer be altered, whether a logic 0 or a logic 1.

4.5.12 G2ULOCKUSER: Clear USER Lock

Syntax	`G2ULOCKUSER`
Function	Clears the USER memory bank lock bit
Parameters	None
Description	Clears the lock bit of the USER memory bank. Prior to executing the G2ULOCKUSER command, the tag must be in the *secured* state. A tag whose access password is null will automatically advance from the *open* state to the *secured* state during arbitration; a tag whose access password is non-zero remains in the *open* state until the tag has successfully transitioned to the *secured* state – this is done through the G2ACC command and the valid respective passwords.

4.6 Block Permalock

The Gen2 block permalock function allows discrete USER memory blocks of data to be permanently locked. This functionality allows part of the USER memory to be locked while allowing other USER memory blocks of data to be written. This is useful in applications where a portion of the USER bank memory bank is reserved for application settings and/or definitions creating static and dynamic memory regions within the USER memory bank. This becomes especially useful considering the large size of the MaxArias USER memory bank. The alternative to block locking (available only for the USER memory bank) is to permanently lock the entire USER memory bank which in some cases may not be desirable. The block lock status can be set and read through the following commands:

G2BLKPERMALOCK_LK

G2BLKPERMALOCK_RD

USER memory blocks that have been block-locked are permanently locked and may not be unlocked.

4.6.1 G2BLKPERMALOCK_RD: Read USER Block Lock Status

Syntax `G2BLKPERMALOCK_RD {blkpntr} {range}`

Function Reads the block permalock status

Parameters

Parameter	Value (hex)	Definition
blkpntr	0-7F	
range	1-FF	

Description Performs a Gen2 blockpermalock read instruction. The USER memory block lock status is read with a granularity of 16 blocks encoded into a 16-bit data word. The *blkpntr* parameter defines the starting block in units of 16 blocks – *blkpntr*=0 maps to block 0; *blkpntr*=1 maps to block 16; *blkpntr*=2 maps to block 32, etc. The *range* parameter defines the number of 16-block permalock status data words to return. Thus, the block permalock read status is further defined by:

blocklock_start_block = *blkpntr*

blocklock_end_block = *blkpntr* + (*range*×16)-1

4.6.2 G2BLKPERMALOCK_LK: Set USER Block Lock

Syntax `G2BLKPERMALOCK_LK {blkpntr} {mask}`

Function Performs the locking function of the block permalock Gen2 command

Parameters

Parameter	Value (hex)	Definition
blkpntr	0-7F	
mask	0-FFFF	

Description Performs a Gen2 blockpermalock lock instruction. Once a USER memory block has been permalocked, it may not be unlocked and is permanent – the respective USER memory block may no longer be written to. The *blkpntr* parameter defines the starting block in units of 16 blocks – *blkpntr*=0 maps to block 0; *blkpntr*=1 maps to block 16; *blkpntr*=2 maps to block 32, etc. The *mask* defines one or more discrete blocks are to be locked. The G2BLKPERMALOCK_LK command supports only a single set of 16 consecutive blocks – as such, the *range* parameter is fixed at x01 (not included in the CLI command). USER memory blocks that are to be locked are defined within the *mask* parameter by asserting the respective bit to a logic one. For example, say the following blocks are to be block-permalocked:

0,2,3,14,32,40

Two G2BLKPERMALOCK_LK commands would be required. The parameters for the first command would be:

blkpntr = 0

mask = b1011_0000_0000_0010 = xB002

The parameters for the second command would be:

blkpntr = 2

mask = b1000_0000_1000_0000 = x8080

4.7 Other ACCESS Instructions

This section details additional Gen2 instructions required for various functions, such as accessing a password-protected tag by advancing the tag's state from the *open* state to the *secured* state, or generating a MaxArias interrupt.

CLI Command	Description
G2ACKO	Gen2 acknowledge (open state)
G2ACC	Gen2 access command sequence
G2KILL	Gen2 kill command sequence
G2INTGEN	MaxArias Gen2 interrupt generation

4.7.1 G2ACKO: Gen2 Acknowledge – Open State

Syntax `G2ACKO`

Function Performs a Gen2 acknowledge command

Parameters None

Description Performs a single Gen2 acknowledge command. This command is intended to be used in the tag's *open* or *secured* state. The tag shall respond with it's EPC identifier in the same manner as done during the tag query sequence.

4.7.2 G2ACC: Gen2 Access Command Sequence

Syntax `G2ACC {pwd1} {pwd2}`

Function Performs a Gen2 access command sequence

Parameters

Parameter	Value	Definition
pwd1	user-defined	
pwd2	user-defined	

Cmd Reply NOTE: Currently, a successful access command sequence is done through an audible chirp from the buzzer on the reader.

Description Performs a Gen2 access command to advance the tag to the *secured* state. Certain Gen2 access commands, such as lock and kill require the tag to be in the *secured* state prior to successfully implementing the respective Gen2 instruction. A tag whose complete access password is non-zero must utilize the G2ACC command to advance the tag to the *secured* state. The parameters *pwd1* and *pwd2* must match the 2 16-bit passwords stored in the tag's RESERVED access memory locations.

4.7.3 G2KILL: Gen2 Kill Command Sequence

Syntax `G2KILL {pwd1} {pwd2}`

Function Performs a Gen2 kill command sequence

Parameters

Parameter	Value	Definition
pwd1	user-defined	
pwd2	user-defined	

Cmd Reply None

Description Performs a Gen2 kill command to permanently disable the Gen2 interface from responding to any future interrogator communications. Prior to executing the G2KILL command, the tag must be in the *secured* state and must have a non-zero kill password – tags with kill passwords set to zero cannot be killed. Applications that never want the tag killed shall set the kill password to zero and permanently lock the kill password in the RESERVED memory bank. The parameters *pwd1* and *pwd2* must match the 2 16-bit passwords stored in the tag's RESERVED kill memory locations.

4.7.4 G2INTGEN: MaxArias Gen2 Interrupt Generation

Syntax `G2INTGEN`

Function Performs a MaxArias DSPI interrupt command generation

Parameters None

Cmd Reply None

Description Performs a MaxArias DSPI interrupt command sequence by transmitting two Gen2 write commands to USER memory address 0x04 and 0x05. Prior to executing the G2INTGEN command, the tag must be in the *open* or *secured* state. The G2INTGEN command will consecutively write USER memory address 0x04 with a data value of 0x1200 and USER memory address 0x05 with a data value of 0x0034. If the MaxArias tag is connected to the reader's DSPI port, the assertion of the tag's DSPI CS ping will cause all 4 status LEDs to flash. The interrupt can be cleared by using the command DSINTEND.

4.8 DSPI Instructions

The CLI supports basic instructions to read/write MaxArias memory through the DSPI port. Additional functions have been incorporated to further assist use and implementation.

CLI Command	Description
DSREAD	DSPI read
DSWRITE	DSPI write
DSINTEND	DSPI interrupt end opcode
DSPINGON	DSPI ping enable
DSPINGOFF	DSPI ping disable

4.8.1 DSREAD: DSPI Read

Syntax `DSREAD {addr} {len}`

Function Performs a serial port DSPI read cycle

Cmd Reply *len* data words

Parameters

Parameter	Value (hex)	Definition
addr	0-3FF	DSPI starting address
len	0-400	Number of data words to read starting at *addr*

Description Performs a DSPI read cycle of *len* data words starting at address *addr*.

4.8.2 DSWRITE: DSPI Write

Syntax `DSWRITE {addr} {data}`

Function Performs a serial port DSPI write cycle

Parameters

Parameter	Value (hex)	Definition
addr	0-3FF	
data	0-FFFF	

Cmd Reply None

Description Performs a DSPI write cycle at address *addr* with a single data word *data*.

4.8.3 DSINTEND: DSPI Interrupt End Opcode

Syntax	**DSINTEND**
Function	Transmits a serial port DSPI interrupt end opcode
Parameters	None
Cmd Reply	None
Description	Performs a MaxArias DSPI interrupt end instruction by transmitting the DSPI INTEND opcode. Refer to the MaxArias datasheet for additional detail.

4.8.4 DSPINGON: DSPI Ping Enable

Syntax	**DSPINGON**
Function	Enables DSPI ping
Parameters	None
Cmd Reply	None
Description	Enables the DSPI ping control bit. DSPI ping is a periodic reader poll of the DSPI port. It provides a mechanism for the reader to auto-detect the presence of a connected MaxArias device. When connected correctly, the LED directly next to the DSPI header (D3) will light to indicate that the MaxArias tag has been auto-detected. When either disconnected or incorrectly connected, the DSPI connect LED (D3) will not be lit. DSPI ping may be disabled through the CLI command DSPINGOFF.

4.8.5 DSPINGOFF: DSPI Ping Disable

Syntax	**DSPINGOFF**
Function	Disables DSPI ping
Parameters	None
Description	Disables the DSPI ping control bit. DSPI ping is a periodic reader poll of the DSPI port. It provides a mechanism for the reader to auto-detect the presence of a connected MaxArias device. When the DSPI ping control bit has been cleared, the DSPI connect D3 status LED is undefined. Disabling the DSPI ping allows the user to trigger off the any of the DSPI interface signals (available either directly on the DSPI port itself or through the test port with the correct test page selected) without interference from the reader's ping polling sequence. DSPI ping may be enabled through the CLI command DSPINGON.

4.9 Direct-Mode Instructions

Direct-mode instructions give access to the reader's register map. Caution should be exercised when the direct mode instructions – they should only be used as indicated within this documentation or through advice from Ramtron's technical support.

4.9.1 W: Direct-Mode (Debug) Write Command

Syntax `W {addr} {data}`

Function Direct debug mode register write instruction

Parameters

Parameter	Value (hex)	Definition
addr	0-FFFF	
data	0-FFFF	

Cmd Reply None

Description Performs a reader register write command. The write command requires the parameters *addr* and *data* defining the register address and data to be written.

4.9.2 R: Direct-Mode (Debug) Read Command

Syntax `R {addr} {len*}`

Function Direct debug mode register read instruction

Parameters

Parameter	Value (hex)	Definition
addr	0-FFFF	
len	0-FFF	

Cmd Reply `:dddd`
where *dddd* is a 16-bit data word

Description Performs a reader register read command. The read command requires an address parameter; in some cases, the read command may additionally require an optional length parameter.

5 GEN2 COMMAND EXAMPLES

To perform a single tag query:

```
ASINIT
RFON
G2QRY
```

NOTE: ASINIT is only executed a single time after reader power-on.

To perform a continuous-query of the RFID tag:

```
G2CQRY
```

NOTE: Continuous-query mode terminated by hitting the ESC or ENTER key.

Examples from this point forward assume that the RF is ON and that the tag has been queried into the tag's *open* state.

To read the tag's TID memory:

```
G2READTID
```

To read the entire tag's USER memory:

```
M2
G2READUSER
```

To read 18 (x12) words of USER memory starting at address x020:

```
G2READ 20 12
```

To read 50 (x32) words of USER memory starting at address x123:

```
G2READX 8223 32
```

NOTE: Reading memory above address 0x7F requires the use of the G2READX command and EBV-formatted addressing.

To write the value x1234 to address x054:

```
G2WRITE 54 1234
```

To write the value x2323 to address x101:

```
G2WRITEX 8201 2323
```

NOTE: Writing memory above address 0x7F requires the use of the G2WRITEX command and EBV-formatted addressing.

To blockwrite 8 words of data (x1111, x222, …) to address 0x010:

```
G2BLKWRITE 0F 1111 2222 3333
4444 5555 6666 7777 8888
```

NOTE: There is a single space between all commands parameters. The address parameter is one less than the desired value in the event the MaxArias auto-increment bit is set in its Control/Status register.

To maximize the reader's output power:

```
ATTEN 0
```

To convert an address to EBV formatting used for commands such as G2READX and G2WRITEX:

```
EBVCONV 123
```

NOTE: The EBVCONV command given above generates the EBV address 8223 used in the G2READX example earlier.

APPENDIX A Command Line Interpretter (CLI) Shortform Command Set

CLI Opcode		Parameter 1	Parameter 2	Parameter 3	Parameter 4	Parameter 5	Description
ASINIT		n/a					AS3992 Initialization enable.
RFOFF		n/a					RF disable
RFON		n/a					RF enable
SPKRON		n/a					Buzzer enable
SPKROFF		n/a					Buzzer disable
RFCHAN		data					RF channel select
LF40K		n/a					Set link frequency = 40kHz; TARI = 25.0µs
LF160K		n/a					Set link frequency = 160kHz; TARI = 6.25µs
M0		n/a					Set display mode to raw
M1		n/a					Set display mode to single-column
M2		n/a					Set display mode to 8-column
M3		n/a					Enable line feeds
M4		n/a					Disable line feeds
M5		n/a					Enable CLI prompt
M6		n/a					Disable CLI prompt
T		n/a					Gen2 command sequence timer
TESTPAG		n/a					Test port page select 0x0 - 0xF
G2SCAN		n/a					Gen2 Continuous Query over all 50 RF channels
G2FASTSCAN		n/a					Gen2 Continuous Query over 50 RF channels – stick on working channel.
G2AUTOSCAN		n/a					Gen2 Continuous Query over functioning RF channels only.
G2CQRY		Q	Sel	Session	Target	M	Gen2 Continuous Query
G2QRY		Q	Sel	Session	Target	M	Gen2 Single Query
G2RQRY		Q	Sel	Session	Target	M	Gen2 Re-Query
G2QRYPASS		n/a					Gen2 Query auto-mode tag pass count
G2QRYFAIL		n/a					Gen2 Query auto-mode tag fail count
G2ACKO		n/a					Gen2 acknowledge - OPEN state
G2ACC		pwd1	pwd2				Gen2 ACCESS
G2KILL		pwd1	pwd2				Gen2 KILL
G2INTGEN		n/a					Gen2 MaxArias DSPI interrupt generate

CLI Opcode		Parameter 1	Parameter 2	Parameter 3	Parameter 4	Parameter 5	Description
G2READ		addr					Gen2 Read
G2READX		addr					Gen2 Extended Read
G2READRSVD		n/a					Gen2 Read Reserved Memory Bank
G2READEPC		n/a					Gen2 Read EPC Memory Bank
G2READTID		n/a					Gen2 Read TID Memory Bank
G2READUSER		n/a					Gen2 Read USER Memory Bank
G2READCNTL		n/a					Gen2 MaxArias Read Control/Status Register
G2READSTRDADDR		n/a					Gen2 MaxArias Read Working Stored Address Register
G2WRITE		addr	data				Gen2 Write
G2WRITEX		addr	data				Gen2 Extended Write
G2WRITEEPC		data x7					Gen2 Write EPC identifier
G2WRITEACC		data x2					Gen2 Write access password
G2WRITEKILL		data x2					Gen2 Write kill password
G2WRITECNTL		data					Gen2 MaxArias Write Control/Status register
G2WRITESTRDADDR		data					Gen2 MaxArias Write Working Stored Address register
G2BLKWRITE8W		***					Gen2 8-word blockwrite
G2BLKWRITE32W							Gen2 32-word blockwrite - data updated through direct-mode register writes
G2BLKWRITE127W							Gen2 127-word blockwrite - data updated through direct-mode register writes
G2BLKWRITE994W							Gen2 994-word blockwrite. For demonstration purposes only.
G2UWRITE		data x2					Gen2 MaxArias Unaddressed-Write
G2LOCKKILL							Gen2 lock KILL password.
G2PLOCKKILL							Gen2 permalock KILL password
G2ULOCKKILL							Gen2 unlock KILL password
G2LOCKACC							Gen2 lock ACCESS password
G2PLOCKACC							Gen2 permalock ACCESS password
G2ULOCKACC							Gen2 unlock ACCESS password
G2LOCKEPC							Gen2 lock EPC memory bank
G2PLOCKEPC							Gen2 permalock EPC memory bank
G2ULOCKEPC							Gen2 unlock EPC memory bank
G2LOCKUSER							Gen2 lock USER memory bank
G2PLOCKUSER							Gen2 permalock USER memory bank

CLI Opcode		Parameter 1	Parameter 2	Parameter 3	Parameter 4	Parameter 5	Description
G2ULOCKUSER							Gen2 unlock USER memory bank
G2BLKPERMALOCK_RD		blkpntr	len				Gen2 block-permalock read USER blocks lock status bits
G2BLKPERMALOCK_LK		blkpntr	mask				Gen2 block-permalock lock USER blocks.
DSREAD		addr	len				DSPI read command
DSWRITE		addr	data				DSPI write command
DSINTEND		n/a					DSPI INTEND opcode.
DSPINGON		n/a					
DSPINGOFF		n/a					
W		addr	data				Direct-mode write cycle
R		addr	len				Direct-mode read cycle

APPENDIX B Direct Register Memory Map

Register Mnemonic	Address (hex)	Init. State (hex)	Data Granularity (bits)	Description
MAIN_CNTL	00		8	Main control register (generate hardware reset)
DSPI_FREQ	23	000F	16	DSPI clock frequency register.
QRY_PASS_CNT	58	0000	16	Continuous query mode pass count
QRY_FAIL_CNT	59	0000	16	Continuous query mode fail count
RF_CHAN	7A	1A	8	Current RF channel
FPGAVER	80		8	FPGA version register.
TEST_REG	FF	00	8	FPGA test register. The 4 least significant bits of this register appear as outputs on the test port when the TESTPAG register is written with a value of xE.

APPENDIX C Test Port Pages

Page	Test 0	Test 1	Test 2	Test3
0	Gen2 Tx BB	Gen2 Rx BB	Gen2 Rx Enable	Gen2 Rx Done
1	DSPI Clk	DSPI Data1	DSPI Data0	DSPI CS
2	UART TxD	UART RxD	UART RTS	UART CTS
4	Rx decode enable	Rx decode data	Rx decode strobe	Gen2 Rx BB
A	Gen2 Tx BB	Gen2 Rx BB	Gen2 Rx Enable	DSPI CS
E	test_reg(0)	test_reg(1)	test_reg(2)	test_reg(3)

APPENDIX D EBV-Formatting

EBV-formatted addressing is best understood by first considering the conversion from an EBV-formatted address to a standard-formatted address using the following simple rules:

1. Remove the most significant bits from all bytes.
2. Concatenate the remaining bits.
3. The result is a standard formatted address.

For MaxArias tags up to 16k-bit, 2 bytes are sufficient to address the entire memory. The most significant bit of the most significant byte is always set to a logic one, while the most significant bit of the least significant byte is always set to a logic zero.

Given the above addressing conversion, consider the conversion of a standard-formatted address to an EBV-formatted address:

1. Present the standard address as a binary value.
2. Divide the binary address value using a delimiter every 7 bits.
3. Add a flag bit to the most significant bit location for each of the group of 7 bits. The value of the flag bit is defined as a logic zero for the least significant set of 7 bits and a logic one for all other set(s) of 7 bits.

Several EBV conversion examples are shown below. It should be noted that for Gen2 extended addressing commands, a 16-bit EBV-formatted address must be used. Address values in the range 0x00 to 0x7F simply have a 0x80 prepended to the address to generate a 16-bit EBV-formatted address. For example, the following addresses are equivalent:

0x00 = 0x8000

0x01 = 0x8001

0x7F = 0x807F

Address (hex)	Address (binary)	Address (formatted to 7 bit characters)	EBV8-formatted		EBV8 (hex)
x00	b0000_0000	0_0000000	0_0000000		x00
0x01	b0000_0001	0_0000001	0_0000001		x00
0x7F	b0111_1111	0_1111111	0_1111111		0x7F
0x80	b1000_0000	1_0000000	1_0000001	0_0000000	x81_00
0xFF	b1111_1111	1_1111111	1_0000001	0_1111111	x81_7F
0x100	b1_0000_0000	10_0000000	1_0000010	0_0000000	x82_00
0x300	b11_0000_0000	110_0000000	1_0000110	0_0000000	x86_00
0x3FF	b11_1111_1111	111_1111111	1_0000111	0_1111111	x87_7F

Revision History

Rev'n	Date	Author	Comments
1.00	05.09.11	Kirk Greefkes	Initial creation.
1.01	05.21.11	Kirk Greefkes	Add elapsed timer information. Add section 5. Add two appendices. Add various detail.
1.03	09.02.11	Kirk Greefkes	Command updates.
1.04	10.20.11	Kirk Greefkes	1. Additional information regarding G2BLKWRITE data shifting. 2. Add G2FASTSCAN to Appendix A. 3. Add output power graph to ATTEN documentation. 4. Buzzer functionality when the USB port is connected/disconnected. 5. Add key register bit definitions for the WM72016 control/status register. 6. Include 115,200/460,800 baud sensing documentation. 7. Update test port page definitions. 8. Update reader startup and USB driver installation description. 9. Update Gen2 command examples (section 5).
1.05	10.31.11	Kirk Greefkes	Update reader reset description. Update ATTEN graph details. Other minor text updates.