

MercuryOS Advanced User Guide

**For: Mercury4, Mercury5 (v2.3.23 and later)
Astra (v4.1.17 and later)**

Government Limited Rights Notice: All documentation and manuals were developed at private expense and no part of it was developed using Government funds.

The U.S. Government's rights to use, modify, reproduce, release, perform, display, or disclose the technical data contained herein are restricted by paragraph (b)(3) of the Rights in Technical Data-Noncommercial Items clause (DFARS 252.227-7013(b)(3)), as amended from time-to-time. Any reproduction of technical data or portions thereof marked with this legend must also reproduce the markings. Any person, other than the U.S. Government, who has been provided access to such data must promptly notify ThingMagic, Inc.

ThingMagic, Mercury, and the ThingMagic logo are trademarks or registered trademarks of ThingMagic, Inc.

Other product names mentioned herein may be trademarks or registered trademarks of ThingMagic, Inc. or other companies.

© Copyright 2000–2008 ThingMagic, Inc. All Rights Reserved

ThingMagic, Inc.
One Broadway, 5th floor
Cambridge, MA 02142
866-833-4069

Revision B December 2008

Contents

MercuryOS Advanced User Guide	5
Introduction	5
Methods of Setting Parameters	6
Permanently Setting Parameters	7
Tm.conf File	9
Web Interface	10
RQL	11
Example	11
Temporarily Setting Parameters	12
RQL	12
API	12
Reader Configuration Parameters	13
Network Parameters	14
Miscellaneous Configuration Parameters	15
SNMP Parameters	16
Gen2 Parameters	17
EPC0/EPC1/ISO18000-6b Parameters	19
Quick Search Parameters [M4, M5 Only]	21
Regulatory Parameters	22
Performance Tuning	23
Modes of Operation: GEN2	24
Sessions	24
Session Example	25
Targets	25
Changing the Gen2 Session and Target Settings.	27
Tm.conf	27
RQL	27

Gen2 Target Parameter Values	27
Example 1:	28
Example 2:	28
Searching with Session and Target Parameters	28
Example: Session=1 and Target=2 (Default)	29
Example: Session=0 and Target=1	30
Example: Session=2 and Target=3	31
SNMP	33
Introduction	33
SNMP Overview	34
SNMP Enabled (Yes/No):	34
SNMP Write Enabled (Yes/No):	34
SNMP Read Community (string):	34
SNMP Write Community (string):	35
EPCglobal Reader MIB	35
Levels of Abstraction	35
Notifications	36
SNMP Interfaces	37
MercuryOS Web Interface	37
C API Interface	37
Automatic Upgrade Mechanism	38
Activating AutoUpdate	38
“Jitter” Time	39
DHCP Server Required	39
Creating tmfw AutoUpdate Packages	39

MercuryOS Advanced User Guide

Introduction

This guide is appropriate for those users who have an advanced understanding of how the Mercury4, Mercury5 and Astra readers operate and want to configure the readers for optimum performance.

Within this guide you will find information on:

- ◆ [Methods of Setting Parameters](#) using the web interface, and the C Application Programming Interface (API) or the Reader Query Language (RQL) programs.
- ◆ A list of [Reader Configuration Parameters](#) which can be used for configuring the readers.
- ◆ Improving the performance of your reader using the methods detailed in the [Performance Tuning](#) section.
- ◆ Monitoring your system using [SNMP Interfaces](#)
- ◆ Upgrade the firmware automatically. This process is explained in the section [Automatic Upgrade Mechanism](#).

Note

Not all functionality is available on all the products, exceptions are noted.

Methods of Setting Parameters

There are four methods for accessing and setting reader parameters. These methods are the following:

- ◆ Web interface
- ◆ *tm.conf* file
- ◆ RQL commands
- ◆ API commands

The web interface and the *tm.conf* file permanently store any parameters and any configuration data that you set. The web interface lets you change any parameters stored in the *tm.conf* file. When the reader is started up, the parameters are pulled from the *tm.conf* file. For information on storing parameters and data permanently, see the next section “Permanently Setting Parameters.”

Parameters set through RQL or API commands are valid until the reader is rebooted or the parameters are changed through the web interface or *tm.conf* file. For information on the API and RQL commands, see [Temporarily Setting Parameters](#).

Permanently Setting Parameters

The `tm.conf` file and the web interface allow you to permanently store parameters and configuration data. The following table compares the `tm.conf` file and the web interface.

Note

These settings are the same as those found in the factory default `tm.conf` shipped with your reader.

Tm.conf and Web Interface Setting

Tm.conf Setting	Web Interface Settings Page	Platform
hostname	Hostname	Astra, M4, M5
iface_ixp1	Use DHCP? (No: 'static')	Astra, M4, M5
dhcpcd_ixp1	-t 15 : Timeout (secs) for DHCP offer command -H : Use DHCP Server-supplied Hostname	Astra, M4, M5
ipaddr_ixp1	LAN IP Address	Astra, M4, M5
netmask_ixp1	LAN Netmask	Astra, M4, M5
gateway_ixp	LAN Gateway	Astra, M4, M5
ntp_servers	NTP Server	Astra, M4, M5
uhf_power_centidbm	UHF Power (dBm)	Astra, M4, M5
cc915_id_length	Class 1 96-bit Support	M4, M5
primary_dns	Primary DNS Server	Astra, M4, M5
gen2EnableDRM	Generation 2 Settings	M4, M5
secondary_dns	Secondary DNS Server	Astra, M4, M5
domain_name	Domain Name	Astra, M4, M5
dhcpcd_vendor	Vendor Class ID	Astra, M4, M5
hostname_automatic	Automatic Host Name	Astra, M4, M5
boot_config_opt	Boot Config Options	Astra, M4, M5
boot_config	Boot Config (URI of Config File) Note: local:default prevents downloading of new config file at boot time	Astra, M4, M5
boot_firmware	Boot Firmware (URI of Firmware File)	Astra, M4, M5
boot_firmware_opt	Boot Firmware Options	Astra, M4, M5
gen2InitQ	<i>[tm.conf only]</i>	M4, M5
epc0_search_depth	EPC0 Search Depth	M4, M5
default_rql_query	Default RQL Query	Astra, M4, M5
syslog_host	Syslog Host	Astra, M4, M5
failsafe_ipaddr_ixp1	Fallback IP Address	Astra, M4, M5

failsafe_netmask_ixp1	Fallback Netmask	Astra, M4, M5
failsafe_gateway_ixp1	Fallback Gateway	Astra, M4, M5
reader_description	Reader Description	Astra, M4, M5
reader_role	Reader Role	Astra, M4, M5
ant#_readpoint_descr	Ant# Description	Astra, M4, M5
secure_shell_only	Secure Shell Only: 'yes' indicates telnet server is disabled (ssh access only).	Astra, M4, M5
secure_web_only	Secure Web Only: 'yes' indicates reader will respond only to HTTPS (secure) URLs	Astra, M4, M5
secure_rql_only	Secure RQL Only: 'yes' indicates RQL is not listing on port 8080 (still OK to tunnel to reader via ssh).	Astra, M4, M5
snmp_enabled	SNMP enabled	Astra, M4, M5
epc0_degghost	EPC0 Tag Deghosting Setting: turns on/off the deghosting threshold level setting for EPC0 tags. <i>[tm.conf only]</i>	M4, M5
epc1_degghost	EPC1 Tag Deghosting Setting: turns on/off the deghosting threshold level setting for EPC1 tags. <i>[tm.conf only]</i>	M4, M5

Tm.conf File

All parameters are pulled from the *tm.conf* file when the reader is booted. Parameters set in the *tm.conf* file are permanent and remain when the reader is rebooted.

All configuration data (except the root password) is stored in *tm.conf*.

Note

The *tm.conf* file structure must be a shell environment file, structured in the standard key=value format.

At boot time, the reader loads its configuration settings from one of two possible places:

- ◆ /tm/etc/tm.conf on the reader itself
- ◆ from a URI referenced in the Boot config setting, such as ftp://server/tm.conf.tmfw

In the firmware file (factory default), there is a *tm.conf* file containing the settings, which the reader uses in various phases of its boot cycle.

To access the *tm.conf* file currently running on your reader:

1. From your terminal program type:

```
telnet <reader IP Address>
```

For example: telnet 10.0.0.101 by default if no DHCP server is active.

2. Log on as superuser:

- a. <hostname> login: root

- b. password: secure

Note

This is the factory-installed and default password for all readers. It is found in the password file: */etc/passwd* on the reader.

3. To display the contents of the *tm.conf* file, type the following:

```
cat /tm/etc/tm.conf
```

Note

Upgrading or downgrading the firmware has no effect on the contents of the *tm.conf* file. Unless the *Reset to Factory Defaults* option is selected during a firmware install.

Web Interface

Setting parameters through the web interface modifies those set in the *tm.conf* file. These parameters are permanent, even when the reader has been rebooted. All parameters are pulled from the *tm.conf* file when the reader is booted.

When you access the reader via a browser, the Settings Page has 5 distinct sections. To check which version of MercuryOS software is running on your reader, go to the Status Page or Diagnostics Page, and check the MercuryOS version. See the *User Guide* for your product for more details.

RQL

Some tm.conf settings can also be updated using the **Saved_Settings** RQL table. When updated using RQL the same rules apply. That is, the change takes effect after a reboot. Cases where a parameter is available for persistent setting via RQL are noted in the description for the parameter in the [Reader Configuration Parameters](#) section.

Example

To update the hostname setting on the reader persistently to take effect on the next boot send the following RQL command:

```
UPDATE saved_settings SET hostname='newname';
```

Temporarily Setting Parameters

The Reader Query Language (RQL) and C Application Programming Interface (API) let you set parameters transiently, until the reader is rebooted or the parameter is changed.

RQL

RQL is a communication protocol that runs between the client software located on a remote computer and the RFID readers.

[Reader Configuration Parameters](#), most of which are in the *params* RQL table, can be set through RQL using the following command syntax:

- ◆ Get the value of a variable as a string.

```
SELECT <variable_name> FROM params;
```

- ◆ Set a variable to a new value.

```
UPDATE params SET <variable_name>=<new_value>;
```

Note

Most parameter values in RQL statements must appear within single quotes.

Example: Setting Gen2InitQ to 3:

```
UPDATE params SET gen2initq='3';
```

See the *RQL Setup and Reference Guide* for further details on using RQL.

API

The MercuryOS APIs provide access to the reader at a very low level.

[Reader Configuration Parameters](#) can be set through the following API calls:

```
char* reader_params_get(reader_handle_t h, char* key);  
int reader_params_set(reader_handle_t h, char* key, char* value);
```

See the *MercuryOS API Reference Guide* for further details.

Reader Configuration Parameters

Reader configuration parameters are used to change the RFID protocol, network, reader security, etc. settings. The following sections provide information on available configurations settings.

- ◆ [Network Parameters](#)
- ◆ [Miscellaneous Configuration Parameters](#)
- ◆ [SNMP Parameters](#)
- ◆ [Gen2 Parameters](#)
- ◆ [EPC0/EPC1/ISO18000-6b Parameters](#)
- ◆ [Quick Search Parameters \[M4, M5 Only\]](#)
- ◆ [Regulatory Parameters](#)

Note

When setting these parameters using RQL or the C API the change takes effect immediately and only transiently. If the reader is rebooted the setting will revert to the default setting or the value specified in Web Interface I Settings / tm.conf file (if available there).

Network Parameters

Network Parameters

Read/ Write	Parameter	Description	Values	Platform
R/W	hostname	The name of the device is dependent on the DNS server configuration. If you are not using DHCP, you must configure the DNS server manually. The default value is 'm4', 'm5' or 'astra'.	['<hostname>']	Astra, M4, M5

Miscellaneous Configuration Parameters

Miscellaneous Parameters

Read/Write	Parameter	Description	Values	Platform
R/W	uhf_power_centidbm	Reader power setting. Value in centi dBm. Default is 3000 centidBm = 30.0 dBm.	[500 - 3000] (int)	Astra
R/W	tx_power	Reader power setting. Value in centi dBm. Default is 3250 centidBm = 32.5 dBm. <i>Note:</i> Get/set in the <i>saved_settings</i> table, not the <i>params</i> table. Still a transient setting.	[1000 -3250] (int)	M4, M5
R/W	antMode or antenna_mode	Enables/Disables bistatic antenna mode. When set to '0' each antenna will be used as an individual monostatic	['0', '1']	Astra,
R	reader_available_antennas	Returns the number of antenna ports on the reader. Does not provide information on which are connected.		Astra, M4, M5
R/W	antennasMakeReadsUnique	When set to 'yes' the antenna becomes a distinguishing feature of a unique tag read. The same tag will be reported on each antenna that reads in during a query.	['yes', 'no']	Astra, M4, M5
R/W	synch_regulatory_timeout	This turns on and off the feature of synchronizing the regulatory timeouts. This uses an algorithm similar to the SYNCH_NTP searches. The default is 0.		M4, M5
R/W	tagOpRetryTimeoutMs	This defines how long in milliseconds tag modification commands retry before giving up. The default is 1000.		M4, M5

SNMP Parameters

SNMP Parameters

Read/Write	Parameter	Description	Values	Platform
R/W	snmp_v1_enabled	Enable/Disable SNMP <i>Note:</i> Only settable via tm.conf	['yes', 'no']	Astra, M4, M5
R/W	snmp_write_enabled	Enables SNMP writes. <i>Note:</i> Only settable via tm.conf	['yes', 'no']	Astra, M4, M5
R/W	snmp_read_community	The community string required for read access to SNMP MIB objects. <i>Note:</i> Only settable via tm.conf	['public']	Astra, M4, M5
R/W	snmp_write_community	The community string required for write access to SNMP MIB objects. For SNMP v1 and v2c (versions current used) the only secure configuration is to disable SNMP writes. <i>Note:</i> Only settable via tm.conf	[<write community>]	Astra, M4, M5
R/W	counter_reset_timestamp			Astra, M4, M5
R/W	ant[#]_readpoint_descr	This is a user-defined identifier string to describe the default Antenna [#] that gets echoed back verbatim via the web interface, SNMP, or RQL. <i>Note:</i> Can be set persistently using <i>saved_settings</i> RQL table	null	Astra, M4, M5
R/W	reader_description	This is a user-defined identifier string to describe the Reader that gets echoed back verbatim via the web interface, SNMP, or RQL. <i>Note:</i> Can be set persistently using <i>saved_settings</i> RQL table	null	Astra, M4, M5
R/W	reader_role	This is a user-defined identifier string to describe the Reader's role in a system that gets echoed back verbatim via the web interface, SNMP, or RQL.? Can be set persistently using <i>saved_settings</i> RQL table	null	Astra, M4, M5

Gen2 Parameters

Gen2 Parameters

Read/Write	Parameter	Description	Values	Platform
R/W	gen2EnableDRM	This sets the reader into and out of dense reader mode when performing Gen2 searches. Default: 'no' in NA, and 'yes' in EU. Note: In EU version of firmware, the reader always operates in DRM Mode.	['yes', 'no']	M4, M5
R/W	gen2InitQ	This is initial value of Q when a search on either GEN2 target begins for the first time. Default is '3'	['0' - '15']	M4, M5
R/W	gen2InitQSmall	When query is resubmitted during a GEN2 search session, this value is used instead of the initQ. Default is 3.	['0' - '15']	M4, M5
R/W	gen2MaxQ	This is the largest value Q is allowed to be during a GEN2 search. Default is 6.	['0' - '15']	M4, M5
R/W	gen2MinQ	This is the smallest value Q is allowed to be during a GEN2 search. Default is 2.	['0' - '15']	M4, M5
R/W	gen2QIncLarge	The GEN2 search algorithm increments by large and small values depending on where in the algorithm it is. Default is 8	[0-8]	M4, M5
R/W	gen2QIncSmall	The GEN2 search algorithm increments Q by large and small values depending on where in the algorithm it is. Default is 1.	[0-8]	M4, M5
R/W	gen2Session	This is the session used by the GEN2 tags when they perform queries. For more information, see Defining Sessions. The default is 1.	['0', '1', '2', '3']	Astra, M4, M5
R/W	gen2Target	The Target type used during GEN2 searches. For more information, see Target(s). The default is 2.	['0', '1', '2', '3']	Astra, M4, M5

R/W	defaultProtocolTimeout	This defines the default timeout, in milliseconds, tag commands attempt to perform their operation before giving up. The default is 500.	[<ms>]	Astra
-----	------------------------	--	--------	-------

EPC0/EPC1/ISO18000-6b Parameters

EPC0/EPC1/ISO18000-6b Parameters

Read/Write	Parameter	Description	Values	Platform
R/W	epc0_deghost	Extra bits in the EPC0 tag are checked to augment checksum. Default is ON. The default value is 1.	['0', '1']	M4, M5
R/W	epc0_deghost_len	This is an unsigned 32-bit number. The number of bits past the end of an EPC0 tag id which can be read to ensure a valid tag and not a ghost tag. See EPC0 de-ghosting techniques for further details. The default value is 8.		M4, M5
R/W	epc0Repeats	This is an unsigned 16-bit number. The RQL equivalent used to specify EPC0 Search Depth. The default value is 80.		M4, M5
R/W	epc1_check_v0_crc	This value controls whether the reader accepts IDs as defined by EPC1 Specification prior to v. 1.0. In these early specifications, a tag's CRCs are calculated differently from those of version 1 EPC1 tags. The epc1_check_v0_crc parameter toggles to accept or reject tags whose CRCs are calculated based on the earlier specification. Default value is 0.	[0 1]	M4, M5
R/W	epc1_deghost	Extra bits in the EPC1 tag are checked to augment checksum. The default value is 1, on.	[0 1]	M4, M5
R/W	epc1_id_length	The radio button that enables 96-bit tag support or 64-bit tag support or both. To optimize the reader for 96-bit tags, select the 96-bit radio button. Same for 64-bit optimization. The default value is 64.	[64 96]	M4, M5

R/W	iso18kTranscoreCompatibility	Enables the reading of Transcore ISO18000-6B tags. The default is off, 0.	['0', '1']	M4, M5
R/W	isoEpcBitCount	This command sets the ISO tag id bit length. By default, the isoEPCBitCount parameter sets reading ISO tag ids to 64-bit length. Bit count changes take effect only in UCode mode. For example, useUCodeEPC parameter must be active (such as set to '1') to enable 96-bit ISO tag id reads, else setting isoEPCBitCount to '96' has no effect and the reader reads ISO tag ids as 64-bit length. Default value is 64.	[64 96]	M4, M5
R/W	useUCodeEpc	This command inactivates or activates reading of ISO18000-6B tag information in UCode format. By default, this is off. Default value is 0.	[1 0]	M4, M5

Quick Search Parameters [M4, M5 Only]

Quick Search is a set of features that utilize a tag-detection procedure in order to quickly determine whether any tags of a particular type (or any tags at all) exist in a reader's read-zone. The purpose of quick search is to turn on the reader (emit RF) only when tags are in the field to be read.

Quick Search uses the information derived from the tag detection to optimize search characteristics. What can be done with the information provided by the tag-detection procedure? During a traditional search, the reader is always emitting RF and the percentage of time spent searching for tags never changes. Quick search uses tag-detection to allow the reader to minimize RF emission and adjust read time as appropriate.

Quick Search Parameters

Read/Write	Parameter	Description	Values	Platform
R/W	quick_search_interval	Controls the time interval between tag detection phases. For example: 50: Use this value for fast moving applications, such as conveyor belts. 100: Use this value for slow moving applications such as portals. Only used when quick_search_mode=0. The default value is 100.	['0' - '100']	M4, M5
R/W	quick_search_mode	0: MINIMUM_RF_OUTPUT - Optimizes Quick Search to minimize RF output. If no tags are found then the reader will not transmit for [quick_search_interval] milliseconds. 1: MAXIMUM_READTIME - In a multi-protocol search, optimizes Quick Search such that if a tag of a particular protocol is not found, time from its statically allocated time-slot will be reallocated fro reading of ther protocols. Default is 0.	[0 1]	M4, M5
R/W	quick_search_on	Sets the default mode of any search performed on the reader from standard to Quick Search Mode. The default value is '0' (off).	['0', '1']	M4, M5
R/W	quick_search_rf_power	RF power level used when quick search is determining whether tags are present. This does not affect the tag read power level. A value of 0 results in the default reader power to be used. Default is 0	[0-32.5]	M4, M5

Regulatory Parameters

Regulatory Parameters

Read/Write	Parameter	Description	Values	Platform	
R/W	lbtOn	This parameter controls whether Listen before Talk (LBT) is used during the frequency hopping period in the EU region. Set to [110] default is 1 for EU readers only.	['0', '1']	Astra, M4, M5	Yes
R/W	regulatoryTimeout	Specifies how long in milliseconds the reader will stay on a frequency channel before hopping to the next channel.		Astra, M4, M5	Yes
R/W	regionversion	Specifies which of the modes of ETSI compliance to use. [EU-region readers only]	eu' : 10 channel, full-power 'eu3' : 4-channel, full-power	Astra	yes
R/W	channel_list	Specifies which of the four high-power channels will be used when set to 'eu3'. Comma seperated list of frequencies (kHz). '0' indicates use all channels. [EU-regions readers only]	'#####, #####, #####, #####'	Astra	Yes

Performance Tuning

The following sections describe how to enhance or tailor the reader's settings to fit your unique RFID environment.

- ◆ [Quick Search Parameters \[M4, M5 Only\]](#)
- ◆ [Modes of Operation: GEN2](#)

Modes of Operation: GEN2

One weakness of the Generation 1 protocol was the possibility that one reader would interfere with another reader's ongoing counting of a group of tags. Generation 2 (Gen2) addresses this problem with the creation of sessions and targets.

Sessions refer to the timeframe during which tags are read. Target parameters determine which flags the reader acknowledges and the order in which they are acknowledged.

- ◆ [Sessions](#)
- ◆ [Targets](#)
- ◆ [Changing the Gen2 Session and Target Settings](#)
- ◆ [Searching with Session and Target Parameters](#)
 - [Example: Session=1 and Target=2 \(Default\)](#)
 - [Example: Session=0 and Target=1](#)
 - [Example: Session=2 and Target=3](#)

Sessions

Sessions, along with inventory flags, categorize tag populations, so that readers cannot mix up which tags were read and which were not read in any given inventory round.

When you click **Query**, an RQL query is initiated and the reader performs one or more inventory rounds depending on the timeout period (longer timeouts result in more inventory rounds being executed). During an inventory round, the reader attempts to read all tags within the field. The reader operates in only one session for the duration of an inventory round.

Unless the session was changed, the default session is Session 1.

The following table displays all session states:

Session Flag States With and Without Power

Session	State of Flag When Tag is Powered On	State of Flag When Tag is Powered Off
0	Keeps new flag state indefinitely	Loses new flag state and reverts back to original state instantly
1	Keeps new flag for between .5 and 5 seconds	Keeps new flag for between .5 and 5 seconds
2	Keeps new flag state indefinitely	Keeps new flag for between 5 seconds and Infinity.
3		

The exact length of time a tag keeps its flag can vary by Tag Manufacturer.

By searching in a session, the reader is “telling” the tags to follow the rules of that session. The rules of each session are stipulated in the Gen2 spec.

Session Example

A retailer or manufacturer can set up their system so that all fixed readers read tags in Session1, and all hand-held readers use Session 2. So, if the fixed reader inventories a tag in Session1, the hand-held reader can still see a tag in Session 2 while not interfering with the ongoing query operation of the fixed reader in Session 1.

Different sessions are assigned to different types of readers. For instance, a company might have dock door readers use Session 1, forklift readers use Session 2 and hand-held readers use Session 3.

The major new features in the Gen2 protocol were driven by the needs of early adopters of EPC technologies to enable systems to perform better and give companies more flexibility in how they use EPC systems.

Targets

As discussed in the section on [Sessions](#), the reader can search for tags and affect them according to their session (0, 1, 2, or 3).

The reader can also search for and affect tags based on their target.

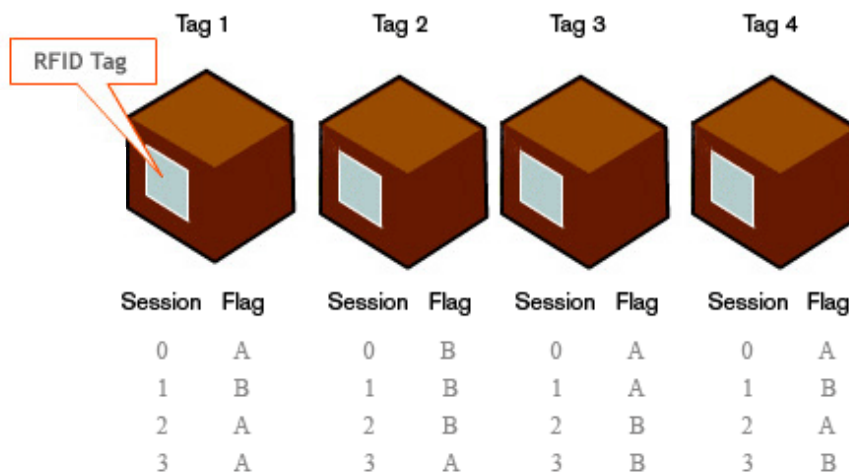
Note

The Gen2 Target setting cannot currently be changed on the Astra. For Astra readers the current static setting is 2, A only.

Target values are contained in the **gen2target** parameter that determines which flags the reader looks for and the order in which it looks for them.

A flag can be set to either 'A' or 'B' and be of session 0 or 1 or 2 or 3. See the figure.

Tags and Associated Sessions and Flags



The target values as shown in the following table can be either 0, 1, 2, or 3. Each value corresponds to which tags the reader looks for and the order in which it looks for them.

Target Values

Target Value	Flag Searched For in Round 1	Flag Searched For in Round 2
0	A	B
1	B	A
2	only A	
3	only B	

Changing the Gen2 Session and Target Settings

The Gen2 session that the reader searches for and affects can be changed. The default value resides in the `tm.conf` file on the reader: `/tm/etc/tm.conf`.

Tm.conf

`tm.conf` changes the default Gen2 Session start up setting by adding the following values to the `tm.conf` file (located in the directory <tr/tm/etc/tm.conf> on the reader):

- ◆ `gen2Session=<0,1,2, or 3>`
- ◆ `gen2Target=<0,1,2, or 3>`

RQL

RQL changes the Gen2 Session setting for the period until the reader is restarted. Use the following steps to change the Gen2 Session setting.

To change the session and target values:

1. Login to the reader.
2. Execute the following commands (choose between to 0, 1, 2, or 3):
 - a. `UPDATE params SET Gen2Session='<0,1,2, or 3>';`
 - b. `UPDATE params SET Gen2Target='<0,1,2, or 3>';`

Gen2 Target Parameter Values

In addition to the **gen2session** parameter on the reader that indicates the session in which the inventory round operates, there is also a parameter called **gen2target**.

The **gen2target** parameter indicates which value(s) of the tags' inventoried flag to query.

The values are:

- ◆ 0=query A then B
- ◆ 1=query B then A
- ◆ 2=query A only
- ◆ 3=query B only
- ◆ The default is `gen2target=2`

Example 1:

Set the session=2 and target=2, and the following actions occur:

All tags with inventoried flag set to A are inventoried and their flag set to B. In Session S2, the inventoried flag maintains its state indefinitely as long as its energized, it does not revert to B unless it is de-energized for longer than its persistent time. At that point, it reverts to A and can be inventoried in a session=2. Repeat target=2 inventory round again. A second query is executed with target=3 to set the flag back to A after it is inventoried.

Example 2:

Set gen2session=1 and gen2target=2, and the following is true:

All tags with inventoried flag set to A are inventoried. When the Gen2 session is set to 2, the tag maintains the last inventoried flag value unless the flag was set longer ago than its persistence time. If that is so, the reader powers up set to A. When Gen2 target is set to 2, it queries for target A only.

After inventorying a tag, its inventory flag should be set to B. Subsequent inventory rounds don't see that tag again until its inventoried flag persistence time has elapsed at which time the tag should revert back to A and is read again.

Searching with Session and Target Parameters

When performing a search, the reader uses the parameters `gen2session` and `gen2target`. To set these parameters, see the section [Changing the Gen2 Session and Target Settings](#).

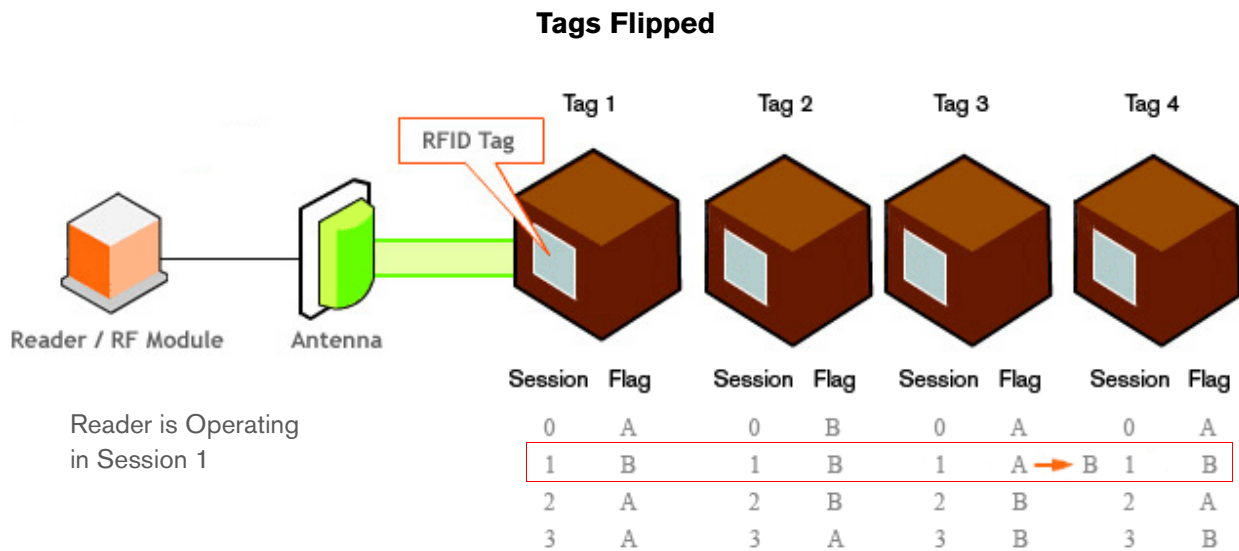
The section provides the following examples:

- ◆ [Example: Session=1 and Target=2 \(Default\)](#)
- ◆ [Example: Session=0 and Target=1](#)
- ◆ [Example: Session=2 and Target=3](#)

Example: Session=1 and Target=2 (Default)

In the following example, the reader uses the parameters **gen2session=1** and **gen2target=2**. The reader can search for and read tags only with a Session 1 Flag of 'A' in all rounds because a Target value of 2 means only 'A' Flags are attenuated between 'A' and 'B'.

- ◆ In the first round, tag 3 flips its flag from 'A' to 'B.' All session 1 tags are now flagged 'B' (see the illustration). All other flags remain unchanged.
- ◆ In all succeeding rounds, all 4 tags, now flagged 'B' are not read since the reader is searching for session 1 tags flagged A only.



How long a flag stays "flipped" is determined by the following criteria:

- ◆ The session the tag is in
- ◆ The Gen2 specification
- ◆ the Tag type

For instance, session 0 tags maintain their new flag indefinitely as long as the tag remains powered on. If the tag is powered off, it flips its flag back to its original state.

Note

When tags revert to their default states, 'B' Flags revert to 'A', 'A' Flags do not revert to 'B'. The 'A' state is the default tag state.

Example: Session=0 and Target=1

In the following example, the reader is using the parameters **gen2session=0** and **gen2target=1**. This means the reader searches for and reads tags with a Session 0 flag of 'B' in round 1 and 'A' in round 2 (because a Target value of 1 means 'B' and then 'A' flags are attenuated between 'B' and 'A').

1. In the first round, tag 2 flips its flag from 'B' to 'A.' All session 0 tags in the example are now flagged 'A.'
2. In the second round, all 4 tags, now flagged 'A' will be read and flipped back to B.
3. In the third round, all 4 tags, now flagged 'B' will be seen again and flipped back to A.

Inventory Round Example



During a given inventory round, only one kind of flag can be flipped. Only 'B' flags can be flipped to 'A,' or 'A' flags flipped to 'B.' Flags of both 'A' and 'B' type cannot be flipped during the same inventory round.

⚠ CAUTION ! ⚠

Seeing the same tags repeatedly by using Target 0 or 1 may cause excessive collisions in large tag populations and is not recommended.

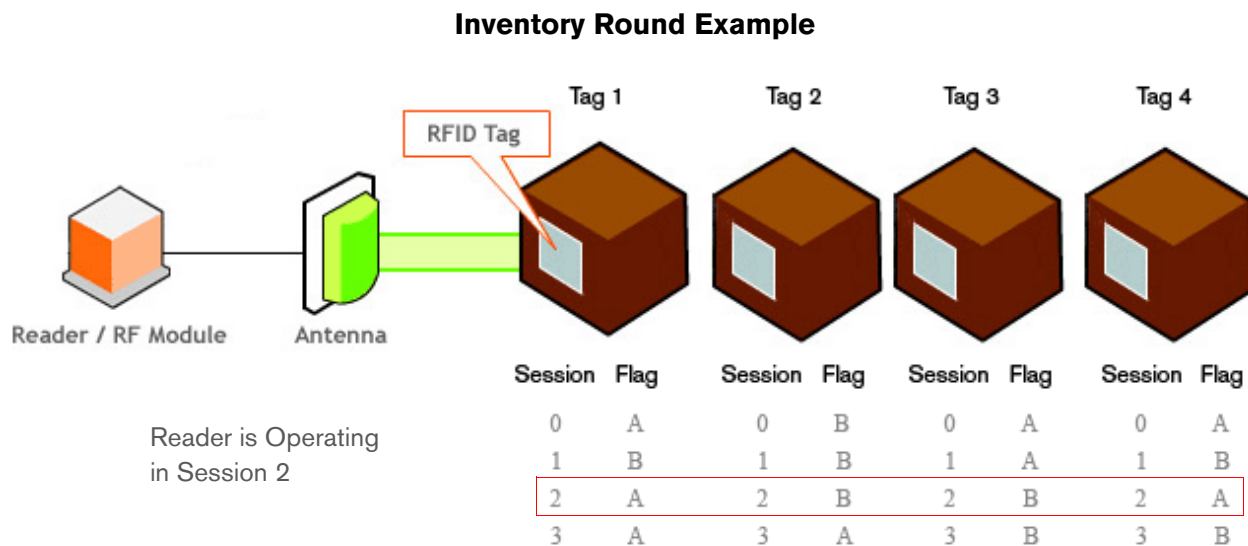
Example: Session=2 and Target=3

For instance, assuming **gen2session=2** and **gen2target=3**, the reader searches for and reads only tags with a Session 2 Flag of 'B' in a given inventory round (because a Target value of 3 means the reader searches for only 'B' Flags).

Once read, the tags flip their flag from 'B' to 'A'. See the illustration. The tags remain flipped to 'A' indefinitely as long as the tag is powered up. From the moment the tag loses power, the tag remains flipped for at least 5 seconds (or more) depending on the tag manufacturer.

As shown in the illustration, if another inventory round is performed with the same parameters, no tags are seen because all 'B' tags were flipped to 'A' and remain flipped for at least 5 seconds. The reader is only searching for tags with a Session 2 Flag of B, therefore no tags appear in an inventory round until at least 5 seconds after they lose power, at which time their Flags revert back to 'A' and can be read again.

See the table [Target Values](#) for more information.



SNMP

Introduction

The MercuryOS facilitates Network Management of the Mercury4, Mercury5 and Astra readers through support for SNMPv1/v2c, including MIB-2 and the EPCglobal Reader MIB as defined by the EPCglobal Reader Management Standard v1.0.1:
(http://www.epcglobalinc.org/standards/rm/rm_1_0_1-standard-20070531.pdf).

SNMP Overview

The MercuryOS SNMP and MIB-II implementations are based on NetSNMP, a widely used, GPL-ed SNMP implementation for Linux. The structure and contents of the underlying NetSNMP configuration files have not been modified.

For customers who are not familiar with the underlying NetSNMP configuration files or who do not prefer to use them, a small set of configuration options will be made available via the web interface, RQL and the C API. These options will include:

SNMP Enabled (Yes/No):

Key: snmp_enabled

Default: Yes

This option determines whether SNMP is enabled on the reader. At start-up, MercuryOS will read the value of this option to determine whether or not to start the SNMP task. If this option is changed to “yes” on a running reader, the SNMP task will be started immediately, and if it is changed to “no” on a running reader, the SNMP task will be killed.

SNMP Write Enabled (Yes/No):

Key: snmp_write_enabled

Default: No

This option controls whether SNMP writes are supported. If this option is set to “yes”, then SNMP writes that use the appropriate SNMP community string will succeed. If it is set to “no”, then no SNMP writes will be allowed using any community string.

SNMP Read Community (string):

Key: snmp_read_community

Default: “public”

This is the community string that must be used for read access to SNMP MIB objects.

SNMP Write Community (string):

Key: snmp_write_community

Default: NULL string

This is the community string that must be used for write access to the SNMP MIB objects. The web interface will not allow users to set the SNMP Write Community from a non-secure (non-HTTPS) connection.

For SNMP version 1 and 2c, the only secure configuration is to disable SNMP writes, using the configuration variable supplied for that purpose. SNMP version 3 (SNMPv3) provides a secure mechanism for SNMP reads and writes, but it is not planned for this release.

EPCglobal Reader MIB

The MercuryOS Reader MIB implementation is fully compatible with the mandatory features of the EPCglobal Reader MIB and the EPCglobal Reader Management Object Model. It is also be fully conformant with the MIB-related portions of the Reader Management Conformance Requirements.

Levels of Abstraction

The Reader MIB is intended to reflect the EPCglobal Reader Object Model in a form that is accessible via SNMP. Therefore, it is impossible to understand the full semantics of the Reader MIB without referencing the Reader Object Model.

In the Reader Object Model, readers can be monitored and managed at four levels of abstraction: Antenna Read Points, Read Points, Reader Devices and Logical Sources. The Reader MIB reflects these four levels with separate objects available at each level for management and monitoring. The MercuryOS architecture does not provide levels of abstraction at each of those levels. Instead, a degenerate version of the MIB, accurately reflecting the levels of abstraction that we do have (Antenna Read Points and Reader Device), and collapsing the levels of abstraction that we do not provide (Read Point and Logical Source) onto the ones that we do provide.

The following table summarizes how each level of Reader MIB abstraction will be represented on the MercuryOS reader:

MIB Reader Representation

Reader MIB Abstraction	Object Model Intent	MercuryOS Representation
Antenna Read Point (epcgAntennaReadPoints)	One physical antenna	One per Antenna
Read Point (epcgReadPoints)	Logical construct intended to group Antennas	One per Antenna
Reader Device (epcgReaderDevice)	One physical reader	One per Reader
Logical Source (epcgSources)	Logical construct intended to cross reader boundaries	One per Reader

Notifications

Not currently supported

SNMP Interfaces

SNMP is typically accessed from a Network Management Station or a health monitoring process running on a remote system. Many such tools exist and use of those tools is recommended.

In addition to the tools commonly available for SNMP based management, the MercuryOS provides several SNMP configuration options along with the ability to get reader statistics through the various MercuryOS control interfaces. These methods include:

MercuryOS Web Interface

As defined in [Permanently Setting Parameters](#) the Web Interface Settings page, and the entries in [Tm.conf File](#), allows for persistent configuration changes including enabling SNMP and setting reader metadata values.

In addition to setting SNMP configuration options the Web Interface can also be used to get reader statistics.

The corresponding product User Guide also provides details on SNMP settings and getting reader statistics through the web interface.

C API Interface

For details on using the C API to configure SNMP and get statistics see the MercuryOS API Reference Guide available at <http://www.thingmagic.com/manuals-firmware>

Automatic Upgrade Mechanism

Automatic Reader Upgrade (ARU) is a mechanism where the readers automatically and periodically discover new firmware updates and update themselves.

A high-level description of the mechanism is as follows:

- ◆ On boot, a reader determines if it needs to update its firmware.
- ◆ If a firmware update is required, the reader does this automatically.
- ◆ If the new firmware package requires a reboot, the reader reboots automatically.
- ◆ If rebooting can be avoided, the reader does not reboot.
- ◆ While running, a reader periodically checks to see if a new firmware update is available
- ◆ Readers download and install the new firmware without any intervention:
- ◆ If the firmware requires a reboot, the readers reboot after installing the firmware
- ◆ If none of these features is implemented, the current behavior is maintained, readers only update firmware on boot

Activating AutoUpdate

By default, AutoUpdate is present but disabled. Each network has its own update policies and rules, so rather than providing one update schedule, we allow the end-user full

control over the update procedure. The following describes how to turn on the AutoUpdate function for your particular application.

Reader updates are controlled by the file `/etc/crontab` which resides on the reader. The default crontab file is as follows:

```
# Update firmware every day at 4AM (600s jitter)
```

```
##* 4 * * * root /bin/autoupdate.sh -j 600
```

Lines beginning with a hash mark (`#`) are comments, so both lines in this file are ignored by cron. If the hash mark is removed from the beginning of the second line it becomes a valid crontab entry.

"Jitter" Time

You can also customize the update process by changing the "jitter" time. This is a random time the reader waits before actually performing the update. This is intended so that in large reader distributions, not every reader performs the update at the same time, clogging the network. If you don't want the jitter, you can eliminate that term:

```
* 4 * * * root /bin/autoupdate.sh
```

Or you can adjust it, for two minutes use a value of 120:

```
* 4 * * * root /bin/autoupdate.sh -j 120
```

DHCP Server Required

The AutoUpdate procedure assumes a DHCP-based network in which the DHCP server supplies a configuration file to the readers as they boot. When autoupdate is being used, the package supplied is an "autoupdate tmfw" package, containing a `tm.conf` file that contains not only the basic reader settings (search depth, TCP/IP settings) but also a minimum firmware version.

Creating tmfw AutoUpdate Packages

AutoUpdate tmfw packages are produced by the "make_autoupdate_tmfw.sh" shell script. When the reader fetches this AutoUpdate package it looks at the minimum

firmware version required and compares it against the currently installed firmware version. If it discovers that the current version is out of date, it fetches the firmware package listed in the "boot_firmware" field of the *tm.conf* file and installs it.

Once the AutoUpdate tmfw package is prepared, it replaces the current tmfw package supplied by the DHCP server. The readers download this new file and use it as described above.