DATALOGIC REMOTE MANAGEMENT UTILITY

USER GUIDE



Windows® and Linux Standalone Utility

DATALOGIC

Datalogic USA, Inc.
959 Terry Street
Eugene, OR 97402
U.S.A.
Telephone: (541) 683-5700
Fax: (541) 345-7140

©2025 Datalogic S.p.A. and /or its affiliates

All rights reserved. Without limiting the rights under copyright, no part of this documentation may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means, or for any purpose, without the express written permission of Datalogic S.p.A. and/or its affiliates.

Owners of Datalogic products are hereby granted a non-exclusive, revocable license to reproduce and transmit this documentation for the purchaser's own internal business purposes. Purchaser shall not remove or alter any proprietary notices, including copyright notices, contained in this documentation and shall ensure that all notices appear on any reproductions of the documentation.

Electronic versions of this document may be downloaded from the Datalogic website (www.datalogic.com). If you visit our website and would like to make comments or suggestions about this or other Datalogic publications, please let us know via the "Contact" page.

Disclaimer

Datalogic has taken reasonable measures to provide information in this manual that is complete and accurate, however, Datalogic shall not be liable for technical or editorial errors or omissions contained herein, nor for incidental or consequential damages resulting from the use of this material. Datalogic reserves the right to change any specification at any time without prior notice.

Trademarks

Datalogic and the Datalogic logo are registered trademarks of Datalogic S.p.A. in many countries, including the U.S.A. and the E.U.

Patents

See www.patents.datalogic.com for patent list.

TABLE OF CONTENTS

DATALOGIC REMOTE MANAGEMENT UTILITY 1
- **INTRODUCTION 1**
- **DISTRIBUTION 1**
  - Windows 1
  - Linux 1
- **INSTALLATION 2**
  - File Organization 2
- **CONFIGURATION 3**
  - Miscellaneous parameters 3
  - Profile parameters 3
  - Model parameters 5
- **OPERATIONS 5**
  - Overview 5
  - Supported Operations 6
  - Command Line Options 6
  - Querying a scanner 8
  - Updating Scanner Firmware 9
- **Errors 11**
- **Legacy ERRORS 13**

APPENDIX A 16
- Command Line Parameter Details 16
- Example Batch file 17
- Linux Terminal Setup 18
- USB 2.0 Controller Troubleshooting 18

DATALOGIC REMOTE MANAGEMENT UTILITY

INTRODUCTION

DLRMUs is a utility whose purpose is to remotely update a scanner's firmware and/or its configuration by uploading a file (payload) to the device. DLRMUs exists as both Windows and Linux implementations and is intended to replace the older utilities **DLRMU** (Windows) and **cmdfw** (Linux).

The **s** in DLRMUs means 'standalone', indicating the application doesn't require either OPOS or JavaPOS to be installed, as in the case of its predecessors.

The latest version of the tool and this manual can be found on www.datalogic.com by searching for **DLRMUs**.

More detailed information about **DLRMUs** can be found on https://datalogic.github.io/opos/dlrmus/.

DISTRIBUTION

DLRMUs distributions in Windows and Linux environments are given below

Windows

Standalone: dlrmus.<version>.zip

OPOS-integrated: Datalogic_OPOS_<version>.msi

Linux

Standalone: dlrmus32.<version>.tar

Standalone: dlrmus64.<version>.tar

INSTALLATION

Standalone versions of the DLRMUs do not require an installer; you must simply unzip the distribution into a directory having read and write permissions. The OPOS-integrated version does require running an installer with administration privileges.

File Organization

Standalone and OPOS-integrated versions of DLRMUs have different requirements and thus the locations of installed files are different.

For standalone versions, the installation layout is:

```
dlrmus\
  Configuration\
    properties.json
  logs\
    <log files>
  dlrmus.exe
  DLRMUS.pdf
  EULA.pdf
```

For the OPOS-integrated version, the installation layout is:

```
C:\Program Files (x86)\
  DLSOPOS\
    Configuration\
      properties.json
    dlrmus\
      dlrmus.exe
      EULA.pdf
    Documentation\
      DLRMUS.pdf
C:\ProgramData\
  Datalogic\
    dlrmus\
      <log files>
```

In all versions, DLRMUs files includes a configuration file template titled properties.upd but uses properties.json for configuration. When DLRMUS is running, if there is no properties.json file present, the properties.upd template is copied into a properties.json file.

CONFIGURATION

Configuration is managed via the JSON file **properties.json**, and from here on it will be referred to as the **properties file**. You should become familiar with the JSON format because something as simple as a missing comma will cause you grief. To this end, there are many JSON validators available as online services. So, if you change the properties file, a validator will help you find syntax errors. The properties file has three basic sections 1) **Profiles**, 2) **Models** and 3) **miscellaneous parameters**. Viewing the properties file in an editor, the first two sections can be found by simply searching for either Profiles or Models. The miscellaneous parameters have no section name but are seen at the top of the file.

Miscellaneous parameters

The miscellaneous parameters are seen at the top of properties.json as:

```
"MaxLogFiles": 10,
“SummaryMaxSize”:100,
"ConsoleInfo": false,
"ConsoleHealth": false,
"ConsoleStatistics": false,
"LegacyErrorCodes": false
```

Of the above parameters, the **MaxLogFiles and SummaryMaxSize** will likely be the most important because they control the lifetime of the log files produced by DLRMUs.

Log files come in two forms 1) the **summary log** provides summary information about successive sessions (uses of DLRMUs) and 2) a **traffic log** provides detailed information about single session. MaxLogFiles controls the maximum number of traffic log files generated before the traffic log files begin to be recycled. SummaryMaxSize, whose units is kilobytes, controls the maximum size of the summary log file. The summary log file will be overwritten when its size exceeds SummaryMaxSize.

The remaining miscellaneous parameters simply control the display of various information to the command/terminal window. Note, the same (output) information is recorded in the log files with more context.

Profile parameters

A **Profile** is simply a name representing a family of scanners, related by their interface, and nearly all DLRMUs commands require you to reference a Profile.

The **Profiles** section is a json-object containing Profile definitions and each Profile is a json-object containing its defining parameters. The **Interface** parameter is significant because it defines the communication method between the host and the scanner. A Profile may contain additional communication properties related to its interface (e.g. COM port, baud rate, etc.)

In the following example, the Profile names are **RS232Scanner**, **SCRS232Scanner**, **RS232Imager** and **USBScanner**.

```
"Profiles": {
   "RS232Scanner": {
      "Interface": "RS-232",
      < more properties >
   },
   "SCRS232Scanner": {
      "Interface": "SC/RS-232",
      < more properties >
   },
   "RS232Imager": {
      "Interface": "USB/COM",
      < more properties >
   },
    "Usbscanner": {
       Interface": "USB-OEM",
       "Usage": "4A00",
        < more properties >
  },
   "ServicePort": {
      "Interface": "SERVICEPORT",
      < more properties >
   },
   "SC-COM": {
     "Interface": "SC-COM",
      < more properties >
   }
}
```

Nearly all DLRMUs commands require you to provide a Profile name with the **-a** command line argument. Example commands are provided in the OPERATIONS segment of this manual.



NOTE: if using the USBCOM interface, then the USB-com driver from Datalogic should be installed, which can be found on the Datalogic website

Model parameters

A **Model** is simply a name representing a specific make of scanner and can be used with some DLRMUs commands to restrict the scope of an operation.

The **Models** section is a json-object containing Model definitions and each Model is a json-object containing its defining parameters. The **Patterns** of a model is a set of character strings representing some number of expected characters returned in the scanner's **TopModelNumber**. It is unlikely you will ever need to make changes in this section.

OPERATIONS

Overview

DLRMUs is a command line application and must be executed with adminstrative privileges using a command/terminal window. A listing of command options can be obtained by executing the command **dlrmus --help**.

During command execution, DLRMUs performs an integrity check, ensuring the existence of the required configuration (JSON) files and a folder to which log files can be written. Failing the integrity check, DLRMUs will report the source of the error and exit immediately.

Excluding such cases of catastrophic failure, DLRMUs generates output in two log files:

1.dlslog.txt (a summary/overview log file)
2.dlslogX.txt (a traffic log file, where X is some value in the range [0..MaxLogFiles-1])

Entries in the summary log are made per session (i.e., per command), and each entry provides a reference to its associated traffic log. The lifetimes of the log files are determined by their respective SummaryMaxSize and MaxLogFiles parameters, defined under miscellaneous parameters.

The log files can (mostly) be ignored. However, if you encounter issues when using DLRMUs, the log files will help you and any support personnel to resolve your issue.

In a standalone installation, log files are recorded in the directory:

dlrmus\logs

In an OPOS-integrated installation, log files are recorded in the directory:

C:\ProgramData\Datalogic\dlrmus

This latter location was chosen because it is adjacent to the OPOS logs directory:

C:\ProgramData\Datalogic\logs

Upon completion of each command, a status code will be written to the summary log and to the command/terminal window.

Status Codes

View the section "Errors" on page 11

Upon successful completion of each firmware updating command, DLRMUs provides revision feedback in the form of:

ApplicationRevisionLevel: <*before updating*> ===> <*after updating*>

ApplicationROMID: <*before updating*>===> <*after updating*>

ConfigurationFileID: <*before updating*>===> <*after updating*>

When formulating a command line, one requiring interaction with a scanner, you must always provide the arguments **-a<Profile name>.** When interacting with a serial device, you may need to provide additional parameters to control communication; the decision to do this depends upon your goal and your programming style.

Supported Operations

DLRMUs supports the following operations:

1. Providing help about command line options
2. Querying a scanner to retrieve Information Health and Statistics directly, as well as in the form of a .prf file.
3. Updating scanner firmware and/or configuration

The details of these operations will be discussed in subsequent sections.

Command Line Options

You can obtain a summary of the available command line options by executing the command:

dlrmus --help

Referencing a standard Windows installation, this command should yield output like:

DLRMUs, Version=1.0.22.4, Datalogic 2024

--help Show this message

Information

-N Display a list of profile names (with associated interface).

-M Display a list of short model names.

Update a Scanner

-a [...] Select the scanner interface profile. See also **-N**.

e.g. **-a RS232Scanner**

-r Changes the meaning of **-F** from TopModelNumber to "short model name". See also **-M**.

e.g. **-r -F "9800i SS"**

-F [...] Select a scanner using its TopModelNumber (or beginning characters of).

e.g. **-F 981234567890** or **-F 981**

-f [...] Select the scanner updating payload. The file path should be both fully-qualified and surrounded by quotation marks.

e.g. **-f "C:\Users\Fred\update.S37"**

-e [...] Set the minimum allowed firmware revision. Prevents firmware updating if the payload revision is less-than-or-equal to the current firmware revision. The revision number is a 4-digit sequence. A value of 9999 allows all updates to proceed.

e.g. **-e 1234**

-l [...] Select a custom log file.

e.g. **-l "C:\Users\Fred\mylog.txt"**

-v [...] Set the delay (in seconds) between resetting the device and reconnecting to it. In the absence of the **-v** arguments, the default delay is 60 seconds.

e.g. **-v 30**

-E Leave the device enabled after an update.

Configure the Serial Port

-c [...] Select the Communicate port.

e.g. (Windows) **-c2** (linux) **-c/dev/ttyACM0**

-b [...] Select the starting baud rate.

e.g. **-b 115200**

-t [...] Select the ending baud rate. In the absence of this parameter the starting baud rate is used.

e.g. **-t 9600**

-d [...] Select the data bits. Must be one of **7** or **8**.

e.g. **-d 7**

-p [...] Select the parity. Must be one of **n**one, **e**ven or **o**dd.

e.g. **-p n**

-s [...] Select the stop bits. Must be one of **1** or **2**.

e.g. **-s 1**

-BCC Enable block check character over Single Cable RS232.

Properties are located at:

C:\Program Files (x86)\ Datalogic \dlrmus\properties.json

Log files are located at:

C:\ProgramData\Datalogic\dlrmus

Exiting with code 0

Querying a scanner

You can obtain the Information, Health and Statistics (IHS) from a scanner with a command as simple as:

dlrmus -a USBScanner

This works only because the USBScanner profile represents a scanner having a USB-OEM interface, so no other parameters are required to establish communication with the scanner.

Two more examples of querying IHS data are:

dlrmus -a RS232Scanner

dlrmus -a RS232Scanner -c 0 -b 9600 -d 8 -s 1 -p n

The first example illustrates using the default communication parameters associated with the RS232Scanner profile. The latter example illustrates using explicit communication parameters.

When you query a scanner for IHS data, the human-readable data is recorded in the summary log. IHS data is also twice recorded in the summary log every time you update the firmware of a scanner (prior to updating and after updating).

Additionally, IHS data (though in a slightly different form) is recorded in a .PRF file along with the log files. The name of the file is based upon the scanner's profile name (e.g. RS232Scanner.PRF) and its content is tailored to specific clients (that is, not everyone will find it useful).

Examples of querying a scanner (various interfaces)

dlrmus -a SCRS232Scanner -c 5	(Single-Cable RS232 Scanner, COM 5)
dlrmus -a RS232Imager -c 18	(USBCOM Scanner, COM 18)
dlrmus -a RS232Imager	(USBCOM Scanner, using a virtual COM port)
dlrmus -a RS232Scanner -c 1 -p n -d 8	(RS232 Scanner, COM 1, No Parity, 8 data bits)
dlrmus -c 5 -b 115200 -a SCRS232Scanner	(Single-Cable RS232 Scanner, COM 5, @115200)
dlrmus -a USBHHScanner	(USBOEM, Handheld Scanner)
dlrmus -a ServicePort	(Service Port connection)

The ServicePort profile is only available on DL scanners which utilize a composite host port. ie: ServicePort is available on a host connection. It supports all three file types .dat, .s37 or .swu.

Updating Scanner Firmware

Updating scanner firmware requires the most complex command lines but, again, the level of complexity depends upon your goal and programming style. For example, using a command line like:

dlrmus -a RS232Scanner -f "C:\Firmware\DR9401275.S37"

DLRMUs will obtain the supplementary parameters from the associated Profile in properties file. However, if you have reconfigured the scanner or want to ensure you are updating the correct scanner using the correct parameters then you might need a more complicated command line like:

dlrmus -a RS232Scanner -c 1 -p n -d 8 -b 115200 -t 9600 -r -F "9400i SO" -f "<filepath>" -v 30

The number of possible command lines is very large and cannot be reasonably discussed in this document. However, the following sections present categorized examples.

Example Scanner Firmware Update

The -f option is used to perform a firmware update. The name of the firmware file to be uploaded follows the -f parameter. Note, you must surround the filepath with quotation characters. For example, to update a Single-Cable RS232 scanner connected to COM2, the following command may be used:

dlrmus -a SCRS232Scanner -c 2 -f "<filepath>"

Example Scanner Firmware Update Using Model Number Validation.

The -F option validates the scanners model number by comparing it to the patterns in the properties file before updating the firmware. The argument associated with -F is typically the first few characters of the TopModelNumber (seen in the PRF file) returned by an IHS.

e.g. -F “984-8”.

The -f option is used to specify the payload to be uploaded. For example, to update any scanner whose model number begins with 86800 and connect to COM2, the following command may be used:

dlrmus -a SCRS232Scanner -c 2 -F "86800" -f "<filepath>"

Example Scanner Firmware Update Using Model Name Validation.

The **-r** option indicates the use of model names. The **-r** option modifies the meaning of **-F** and so it must be used in combination with the **-F** option. **The model name follows the -F parameter and it must be bounded by quotes**

e.g. -r -F “9400i SS”

If the application finds a matching string among the Models in the properties file then the firmware update will proceed. For example, to update a scanner with a model name of 9400i SO and connected to COM1, the following command may be used:

dlrmus -a RS232Scanner -c 1 -b 9600 -r -F "9400i SO" -f "<filepath>"

Example High Speed Scanner Firmware Update Sequence Using Model Name Validation

Please note, the following sequence of commands is better suited for use in a batch file or script. Here, we simply explain the sequence. See the appendix for an example batch file.

Validate model name (9400i SO), communicates at 9600 baud for scanner ID

dlrmus -c 1 -p n -d 8 -a RS232Scanner -b 9600 -r -F "9400i SO" -e 0390 -v 30

Communicates at 9600 for scanner ID, change to 115200 and switch @115200 after the update

dlrmus -c 1 -p n -d 8 -a RS232Scanner -b 9600 -t 115200 -r -F "9400i SO" -f "115K.S37" -v 30

Communicate at 115200 baud for scanner ID and file download.

Communicate at 115200 for scanner ID and to validate firmware update. It will reduce the time to upgrade the firmware (~21min)

Scanner will stay enabled after upgrade

dlrmus -c 1 -p n -d 8 -a RS232Scanner -b 115200 -t 115200 -r -F "9400i SO" -f "DR9401285.S37" –E

Communicate at 115200 baud for scanner ID and file download.

Communicate at 9600 for scanner ID and to validate baud change.

dlrmus -c 1 -p n -d 8 -a RS232Scanner -b 115200 -t 9600 -r -F "9400i SO" -f "9600.S37" -v 30

Example Service Port firmware update:

dlrmus -a ServicePort -f "C:\Firmware\DR9401634.DAT"



NOTE: Certain devices now support a faster host download protocol over USBOEM and DLRMUS will attempt to use this faster method whenever possible.

ERRORS

Legacy Exit Code	Exit Code	Exit Description
0	0	Success
0	0	Device successfully reconnected
1	101	Unknown device profile
1	102	See the summary log <file path> for details
1	103	The supplied file is not appropriate for the targeted device
1	104	See the summary log <file path> for details.) The Information, Health and Statistics query may fail for many reasons. Some things to verify are: 1. Is the device is turned on? 2. Are the cables are properly connected? 3. Are you using the correct device profile? 4. Do the connection parameters you're using match the device configuration?
1	105	See the summary log <file path> for details.) The Information, Health and Statistics query may fail after updating the device firmware for several reasons. Some things to verify are: 1. Did the uploaded file changed the communcation parameters? 2. Did the uploaded file changed the device from flatbed to handheld? 3. Is the pause duration (-v) adequately large?
1	106	The device TopModelNumber is associated with <model no.> and not the specified <model no.>
1	107	Payload does not contain a SoftwareID
1	108	Payload contains an invalid or unsupported SoftwareID.
1	109	Unable to parse Firmware Revision from HIS
1	110	Unable to parse Application ROM File ID from IHS
1	111	Exiting with code 111
1	112	The supplied Firmware Revision (-e) xxxx is the same as the device's yyyy
1	113	The supplied Firmware Revision (-e) xxxx is the older as the device's yyyy
1	114	Payload is not compatible with device.

Legacy Exit Code	Exit Code	Exit Description
1	115	Failed to load the payload.
1	116	Device failed to reconnect
1	117	Failed to enable device
1	118	Failed to open the summary/traffic log: <file path>
1	119	Failed to parse properties file for profile names
1	120	Failed to parse properties file for model names
1	121	Unrecognized model name (see \Models\)
1	122	Failed to update the device. See logs for details
1	123	Device received malformed data, update cancelled
1	124	Unable to write data to device
1	125	Failed to install
1	126	Update did not start
1	127	Did not receive entire message from device
1	128	Failed to send USB-OEM begin update command (or) Failed to send SC/RS-232 hard reset command. (or) Failed to send Serial hard reset command.
1	129	The device struct was initialized improperly
1	130	Get Device Error (or) Failed to open usb-oem device (or) Could not find a device whose usage-value is 0x<usage value> (or) CreateFile error:
1	131	Error getting comm state
1	132	Error setting comm timeouts
2	2	F/W update version == initial version Exiting with code 2 (Success) ApplicationRevisionLevel: <xxxx> ===> <yyyy> ApplicationROMID: <xx> ===> <yy> ConfigurationFileID: <xxx> ===> <yyy>

Legacy Exit Code	Exit Code	Exit Description
3	3	F/W update version **>** initial version Exiting with code 3 (Success) ApplicationRevisionLevel: <xxxx> ===> <yyyy> ApplicationROMID: <xx> ===> <yy> ConfigurationFileID: <xxx> ===> <yyy>
3	3	(Model numbers don't match)
4	4	F/W update version **<** initial version Exiting with code 4 (Success) ApplicationRevisionLevel: <xxxx> ===> <yyyy> ApplicationROMID: <xx> ===> <yy> ConfigurationFileID: <xxx> ===> <yyy>
5	5	Exiting with code 0
6	301	Invalid command line parameter(s)
6	302	Failed to parse <FilePath>
6	303	Failed to migrate properties from <FilePath>

LEGACY ERRORS

Exiting with code 1 (Error: Unknown device profile.)

The argument supplied with the **-a** parameter is not represented in Profiles section of the properties file. A typo in the argument is the most likely reason for this failure.

Exiting with code 1 (Error: Failed to load the payload.)

The likely reasons are 1) the payload cannot be found or 2) there is a file permissions issue.

Exiting with code 1 (Error: The supplied file is not appropriate for the targeted device.)

The payload file-type associated with the **-f** parameter is not correct for the targeted device. Devices using the USB-OEM interface must use .DAT files. Devices using any other interface must use .S37 files.

Exiting with code 1 (Error: Payload is not compatible with device.)

There is a mismatch between the software id of the payload and the software id expected by the device. In this case, contact Datalogic technical support.

Exiting with code 1 (Error: Payload does not contain a SoftwareID.)

Exiting with code 1 (Error: Payload contains an invalid or unsupported SoftwareID.)

There is an issue with payload. Contact Datalogic technical support.

Exiting with code 1 (Error: Failed to update the device. See logs for details.)

The log files may provide sufficient information to help your resolve the issue. If not, contact Datalogic technical support and be ready to provide your log files.

Exiting with code 1 (Error: Failed to query Information. See the summary log <file path> for details.)

Exiting with code 1 (Error: Failed to query Health. See the summary log <file path> for details.)

Exiting with code 1 (Error: Failed to query Statistics. See the summary log <file path> for details.)

The application failed to obtain the requested information from the device. There are many possible reasons for failure. The most common reason is the application did not have adequate time to properly reset before the query was made. The post-reset delay can be extended by using the **-v** option. Also, the summary log provides a reference to the traffic log associated with the action. In turn, the traffic log may provide some clues to the failure.

Exiting with code 1 (Error: Failed to parse json: <some path>/prf_parser.json)

The installed file prf_parser.json likely contains a syntax error.

Exiting with code 1 (Error: The device TopModelNumber is associated with "<model name> " and not the specified "<supplied model name>")

The supplied model name associated with the **-r -F** parameters does not match the device model, as identified by its TopModelNumber. Likely reasons for this failure are 1) the wrong model name was supplied or 2) the Pattern representing the device is associated with the wrong Model in the properties file.

Exiting with code 1 (Error: Failed to parse Firmware Revision from IHS after updating.)

Exiting with code 1 (Error: Failed to parse Application ROM File ID from IHS after updating.)

Contact Datalogic technical support.

Exiting with code 1 (Error: Unable to determine model from IHS data.)

The update was likely successful but the device could not be re-enabled. Contact Datalogic technical support.

Exiting with code 1 (Error: Failed to open the summary log: <file path>/dlslog.txt.)

Exiting with code 1 (Error: Failed to open the traffic log: <file path>/dlslog(X).txt.)

There is likely a file access conflict of permissions issue.

Exiting with code 1 (Error: Failed to parse properties file for profile names.)

Exiting with code 1 (Error: Failed to parse properties file for model names.)

The properties file contains a syntax error.

Exiting with code 1 (Error: Unrecognized model name (see "Models").

The model name associated with the **-r -F** parameters is not represented in Models section of the properties file.

Exiting with code 1 (Error: Unrecognized model number (see "Pattern").

The model number associated with the **-F** parameter is not represented by any pattern in the properties file.

Exiting with code 1 (Error: The supplied Firmware Revision (-e) *xxxx* is the same as the device's *yyyy*)

Exiting with code 1 (Error: The supplied Firmware Revision (-e) *xxxx* is the older as the device's *yyyy*)

These are not errors because the **-e** option was used. The message explains why the attempted updating action was rejected.

APPENDIX A

Command Line Parameter Details

-a	The argument associated with **-a** represents the name of the scanner profile you are going to reference to obtain communication parameters. You can obtain the list of available profile names using: **dlrmus -N.** Also, you can add you own profiles to the properties.json but you must take care to 1) honor the JSON file format and 2) backup your files.
-f [...]	The argument associated with **-f** represents the name of the file to be uploaded to the scanner. Said argument should be a fully-qualified file path surrounded by quotation marks (e.g. **-f "c:\tmp\myfiles\file.ext** "). Also note, USB-OEM scanners accept only .DAT files and all other scanners accept only S37 files. Service Port connections accept .S37, .DAT, and .SWU files.
-F [...]	The argument associated with **-F** is compared against the TopModelNumber returned by the scanner. All characters must match exactly (up to the limit provided by your argument). The scanner will not be updated if there is a mismatch.
-r -F [...]	The argument associated with **-F** is compared against the list of available Model names found in the Models section of properties.json. The Model name you provide must be bounded by quotes (e.g. **-r -F "9400i SO"**). You can obtain the list of available model names using: **dlrmus -M**
-e [...]	The argument associated with **-e** is four digits representing the Firmware Revision of the file you intend to upload to the scanner. The scanner will not be updated if its current EC Level is greater-than-or-equal-to the value you provide. A value of 9999 can be used to force an update.
-c [...]	The parameter following the -c indicates the port to be used for serial communications. You can rely on the default value in properties.json. However, if you insist on defining the parameter on the command line then the value you provide depends upon whether you are a Windows or Linux user. A Windows user need simply provide an integer value representing the COM port number. A Linux user must specify the device path (e.g. -c /dev/ttyS0). Finally, with respect to USB-COM devices, it is possible to use virtual COM port behavior. You can do this by setting "UseVirtualPort" to 1 in properties.json. Not supported with ServicePort Profile.

Example Batch file

The following batch file is an example of how DLRMUs can be used to target specific scanners and perform updates across an enterprise with different Datalogic Magellan Scanners (i.e. Mgl9400, Mgl9800i). This batch file determines the type of scanner, if there is need to update the firmware based on EC level and send the correct file if one of the two needs an update or not then loads the appropriate firmware for the scanner found. This batch file combines many of the DLRMU features to form a very powerful update solution.

- Queries the scanner and gathers Information, Health and Statistics.
- Validates the responding scanner is one of the two desired models.
  Uses “Model Names” and “Model Number Validation”.
- Validates the scanner needs updating. Uses the “EC Level Checking”.
- Checks command status to make decision on which firmware to load.
- Changes baud rate to 115200 to increase the update speed.
- Changes baud rate back to 9600 for customer’s application.(8500Xt only).

```
set comport=2
        dlrmus -a RS232Scanner -c %comport% -b 9600 -r -F "9400i SO" -e 0390
        if "%errorlevel%"=="0" goto download_9400i

        dlrmus -a SCRS232Scanner -c %comport% -b 9600 -r -F "9800i SS" -e 0340
        if "%errorlevel%"=="0" goto download_9800i
        goto exit


:download_9400i
        dlrmus -a RS232Scanner -c %comport% -b 9600 -t 115200 -r -F "9400i SO" –f “115K.S37”
        dlrmus -a RS232Scanner -c %comport% -b 115200 -t 115200 -r -F "9400i SO" –f
        “DR9401143.S37”
        dlrmus -a RS232Scanner -c %comport% -b 115200 -t 9600 -r -F "9400i SO" –f “9600.s37”
        goto exit


:download_9800i
        dlrmus -a SCRS232Scanner -c %comport% -b 9600 -t 115200 -r -F "9800i SS" –f
        “9800i_115200.s37” -v 10
        dlrmus -a SCRS232Scanner -c %comport% -b 115200 -t 9600 -r -F "9800i SS" –f
        “MR21_0494_KRSCO.s37” -v 10
        goto exit

        :exit
```

Linux Terminal Setup

To grant group permissions to access USB devices:

```
sudo usermod -a -G dialout <user>
```

To set the baud rate to 9600, 8 bits, 1 stop bit, no parity:

```
stty -F /dev/ttyS0 raw
stty -F /dev/ttyS0 9600 cs8 -cstopb -parenb
```

To set the baud rate to 9600, 7 bits, 1 stop bit, odd parity:

```
stty -F /dev/ttyS0 raw
stty -F /dev/ttyS0 9600 cs7 -cstopb parenb -parodd
```

USB 2.0 Controller Troubleshooting

Users that have only a USB 2.0 controller (often on older machines running CentOS 7), may experience connection issues with DLRMUS. To solve this modify the lrdms parameter found in the properties.json as follows:

In properties.json go to the specific model parameters and add in one for 'lrdms' (linux read delay milliseconds) with a value of 1000. There are some profiles that already use that parameter.

When connecting to device use both -r and -F flags to identify the model. An example command with the GD4500 is:

sudo ./dlrmus -a USBHHScanner -r -F "GD4500"

©2025 Datalogic S.p.A. and /or its affiliates • All rights reserved • Without limiting the rights under copyright, no part of this documentation may be reproduced, stored in or introduced into a retrieval system, or transmitted in any form or by any means, or for any purpose, without the express written permission of Datalogic S.p.A. and/or its affiliates • Datalogic and the Datalogic logo are registered trademarks of Datalogic S.p.A. in many countries, including the U.S. and the E.U.

DATALOGIC

www.datalogic.com

Datalogic S.r.l.
Via S. Vitalino, 13 | 40012 Calderara di Reno | Bologna - Italy
Tel. +39 051 3147011 | Fax +39 051 3147205

800000330 (Rev C) April 2025