ROGUE robotics

uMMC Serial Data Module
Data Sheet

Version 1.02
Mar. 5th, 2004

Version Information

Hardware Version

Current Version: **UMMC-100-A2**

Revision History:

Jan. 03/2004 – UDL-100-A1/UMMC-100-A1
- Initial Release

Feb. 23/2004 – UMMC-100-A2
- Slight dimension change.
- Centered card connector.
- Various hardware changes.

Firmware Version

Current Version: **101.42**

Jan. 03/2004 – 100.22
- Initial Release

Feb. 23/2004 – 101.42
- Added time-out conditions for Write and Append commands.
- Added support for multiple serial bit rates.
- Added Settings command.
- Added Make Directory command.
- Added Erase File command.
- Added Status command.
- Added Query Volume command.
- Added Version/Serial Number command.

Document Version

Current Version: **1.01**

Jan. 03/2004 – 1.00 Preliminary
- Initial Release

Feb. 23/2004 – 1.01

- First Official Release
- Reflects Hardware/Firmware changes

Mar. 05/2004 – 1.02

- Removed ***Append*** command (no separate command existed – uses ***Write*** command)

General Description

The uMMC™ is a small footprint, serial data storage adapter for reading and writing data to Multimedia Cards (MMC) or Secure Digital (SD) Cards in MMC mode formatted in either FAT16 or FAT32.

Using the uMMC, you can create files and store data to MMC/SD cards up to their respective capacities and then remove the card and read it from any standard MMC/SD card reader on a PC.

The uMMC is perfect for use in low cost, removable data logging, for integration into field devices for inexpensive software update delivery or data extraction, and for anywhere memory expansion, portability and ease of use are important.

Specifications and Standards

Electrical

Input voltage must be regulated +5VDC, at 200mA maximum.

The processor and MMC/SD cards run at 3.3v regulated from +5v with the onboard 3.3v regulator.

The TTL serial interface is +5v maximum.

Ensure that all grounds are tied together for correct, noise-limited operation.

Temperature

This version of the uMMC is rated for commercial temperature ranges.
Commercial Temperature Operating Range: 0ºC to +70ºC

Industrial temperature ranges are special order devices.
Industrial Temperature Operating Range: -40ºC to +85ºC

Communications

TTL Serial Port:

- 9600, 19200, 38400, 57600, or 115200 bps (9600 bps default)
- 8 bits
- no parity
- 1 stop bit
- no echo

MMC/SD interface:

- SPI Mode operation

File System

FAT16 and FAT32 only. (FAT12 is not supported)

File system MUST be formatted on a PC initially. The device does not format cards.

SD cards can be locked and the uMMC detects locking. Encryption features of SD cards are not implemented.

The number of files that can be created is limited by the file system (FAT16, FAT32).

Maximum of 4 files can be open at any time using file handles.

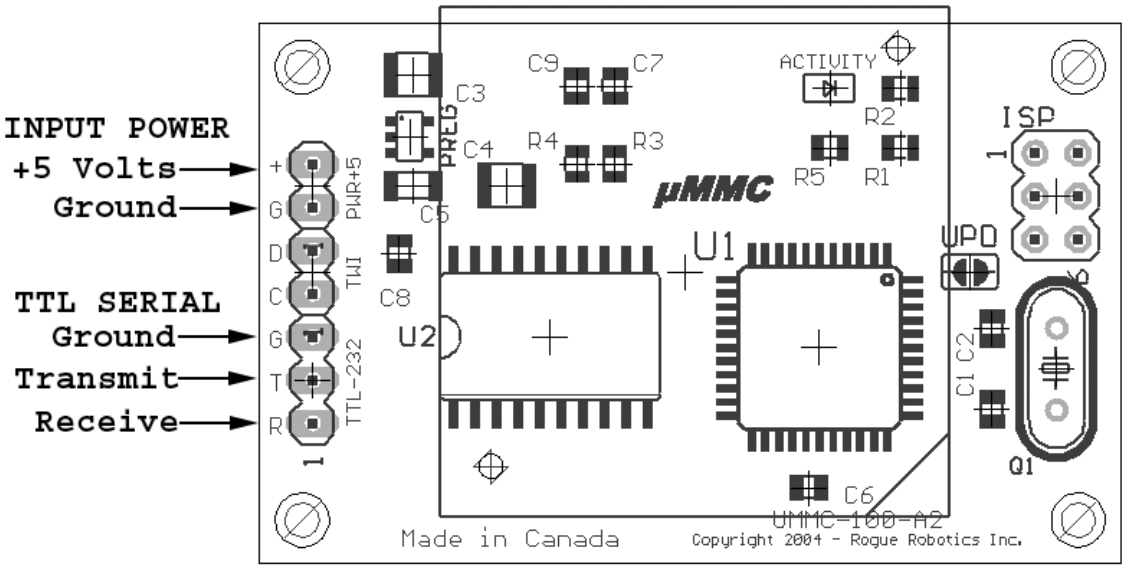
Maximum of 512 bytes per read/write command.

Files must be closed before removing cards for data reliability.

If the Activity LED is on, you should not remove a card. When the LED is off, and you have closed all files, you can safely remove the MMC or SD card.

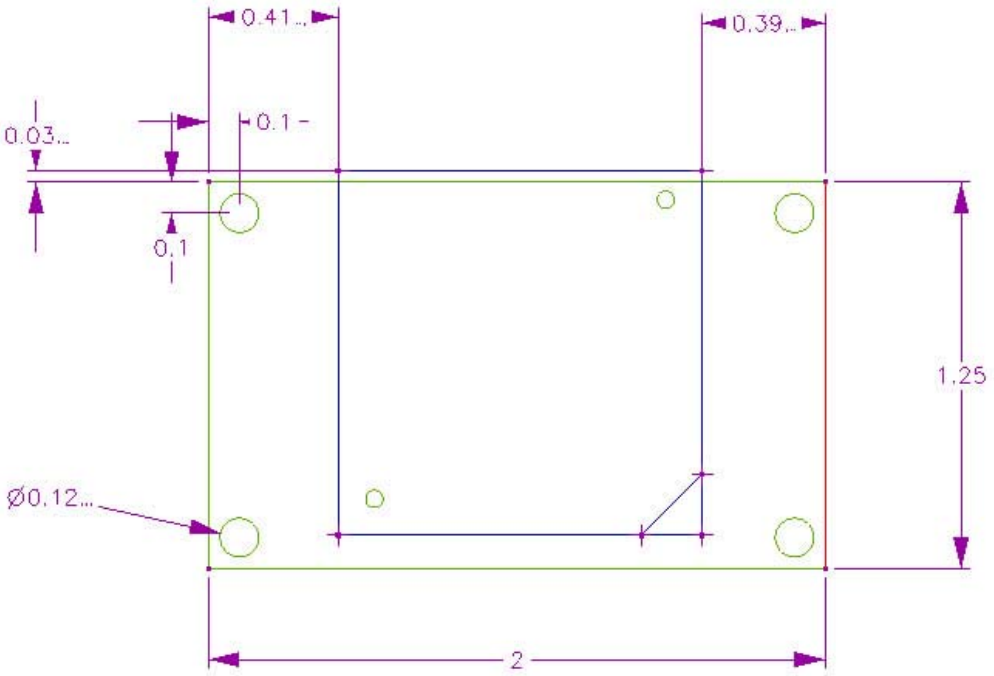
Electrical Connections

Figure 1. – uMMC Electrical Connections



Mechanical Drawing

Units in inches. Holes are clearance holes for 4-40 hardware. The MMC/SD connector hangs off one side and is mounted on the back. DXF files or IGES files available on request.



Updating the Firmware

If there is a new update to the firmware for the uMMC, it may be downloaded from http://www.roguerobotics.com/. Once downloaded, they can be sent to the uMMC to update its firmware. You will need the new firmware and the "update" program. They will be packaged together in a Zip compressed file.

You will need to connect the uMMC to a PC serial port through a TTL Level converter. ***THE uMMC WILL BE DAMAGED IF CONNECTED DIRECTLY TO A PC SERIAL PORT!***

To put the uMMC into bootloader mode and download the firmware:

- Disconnect power to the uMMC, and remove any MMC/SD card from the uMMC.
- Use a small flat-blade screwdriver to bridge the "UPD" jumper, and continue to hold the screwdriver in place.
- Connect the power to uMMC. The Activity LED will stay illuminated (longer than 2 seconds)
- Start the update program from the command line (Start -> Run -> "cmd.exe" or "command.exe"):
    - `"update umm1-10142.rfw"` (if your serial port is something other than COM1, use `"update umm1-10142.rfw -com2"`)

The update program will show the progress and you will see the uMMC Activity LED blink as the firmware is updated. Once complete, the uMMC will reset and start normally.

When the uMMC is put into bootloader mode, all settings for the ***Settings*** command are reset to default values (this is so that you can reset the uMMC if an unknown value is put in any of the settings). You don not have to download anything to reset the values. Just simply put the uMMC into bootloader mode (explained above), then remove the power to the uMMC. The values will be reset.

Communications Protocol

Description

The Protocol for the uMMC employs a simple but robust asynchronous serial control protocol. A command prompt "**>**" ("greater than" symbol, ASCII 62, HEX 0x3E) indicates that the uMMC is ready to accept a command. A command can be sent, a response will be returned, and the command prompt will be sent again.

Example

F*{cr}*
1>

If an error occurs while processing a command, an error is returned in the format Enn, listed in Table 1.

Example:
>O 1 R /LOGS/2004/FEBRUARY/FEB30.LOG*{cr}*
EF2>

Important

After a card is inserted, the card must be scanned the first time for file system information. For FAT16, this can take up to 10 seconds. On FAT32 cards, this never takes more than 2 seconds (although, if the card has just been formatted, the very first time it is used on the uMMC, it can take up to 20 seconds to initialize the file system). Always wait for the command prompt ("**>**") before sending any commands.

Command Format

C{sp}Parameter1{sp}Parameter2{sp}…{cr}

Where:
"C" is a single command character (command listed below)
{sp} is a single space character (ASCII 32, HEX 0x20) [this space is necessary]
Parameter1, Parameter2, … are parameters associated with the command
{cr} is a carriage return character (ASCII 13, HEX 0x0d)

Command Listing Format

C Parameter1 [Parameter2 [Parameter3]]…

"***C***" is the command character. A single character.
"***Parameter1***" is the first parameter for the command.
"***Parameter2***" is the second parameter for the command. If it is listed inside of square brackets [ ] then the parameter is optional.

Any parameters listed inside of square brackets [ ] are optional. Most commands that have optional parameters will require the previous parameter to be given.

Commands

Close File

C *fh*

- *fh* is a file handle (1 - 4)

Description
This closes an open file.

Example
```
>C 1{cr}
>
```

Free Handle

F

Description
Returns the next available free handle (1 through 4). If no handles are available, 0 is returned. *For simple applications, this is not necessary to implement.*

Example
```
>F{cr}
1>
```

Open File

O *fh mode path*

- *fh* is a file handle (1 - 4)
- *mode* is the open mode for the file.
  - 'R' – Open in read mode. Data can be read non-sequentially (randomly). **The filename in the path must exist.**
  - 'W' – Open in write mode. This opens a new file for writing. Data is written to the file sequentially. **The filename in the path must NOT exist.**
  - 'A' – Open in append mode. This opens a new or existing file for writing. Data is written to the file sequentially. **If the filename in the path does not exist, it will be created.**
- *path* is the absolute path to the file. A properly formatted path must begin with a "/" and be absolute (that is, it must begin at the root directory). Sub-directories are separated with "/" (forward slash).
  - Eg. "/LOGS/2004/JANUARY/JAN3.LOG"

Description
This will open a file on the card in one of three modes.

Example
```
>O 1 R /LOGS/2004/JANUARY/JAN03.LOG{cr}
>
```

Important
All files created with the uMMC will have their modification and creation dates set to 01/01/2004 00:00:00. The date is not updated.

Read File

R *fh* [*bytes* [*address*]]

- *fh* is a file handle (1 - 4)
- *bytes* is the number of bytes to read. This parameter is optional. If it is not specified, up to 512 bytes will be returned.
- *address* is the address at which to start reading. The *bytes* parameter MUST be given for the address parameter to be used.

Description
You can read up to 512 bytes at a time using the ***Read*** command. If the *bytes* parameter is larger than the number of bytes remaining in the file, then only the remaining bytes are returned. Use the ***Info*** command to find the current position in and the size of the file. If the ***Read*** command is successful, a single *{sp}* character is sent, followed by the data. If an error occurs, the first character returned is "E", followed by an error code (see Table 2). Data is sent verbatim (no escape characters) from the card.

Example
(the data file contains only two lines of information)

```
>R 1{cr}
 13:22:02 ADC1=4.9V
13:22:32 ADC1=4.9V
>R 1 18 0{cr}
 13:22:02 ADC1=4.9V>
```

Write File

W *fh bytes*

- *fh* is a file handle (1 - 4)
- *bytes* is the number of bytes to be written. You must send this number of bytes to return to the command prompt.

Description
You can write up to 512 bytes at a time with the ***Write*** command. If the bytes parameter is omitted, then 512 bytes will be expected on the incoming serial stream. Data is accepted and written to the card directly (there is no escape sequence).
By default, there is no time-out for how long it takes to send the bytes to the uMMC. This means that the uMMC will wait indefinitely for all the bytes to be sent (unless the power is removed, or a time-out value has been set using the ***Settings*** command).
If you assign a value to the time-out setting using the ***Settings*** command, then the ***Write*** command will terminate, write the accepted bytes to the file, and return to the command prompt; no error will be returned.
If the file has been opened for append, the ***Write*** command will append all bytes to the end of the file.

Example
```
>W 1 18{cr}
13:22:02 ADC1=4.9V>
```

Information

I *fh*

- *fh* is a file handle (1 - 4)

Description
The ***Information*** command returns the current file position and the current file size for a given file handle. The format is **`position/filesize`**. The two values are given in decimal format.

Example
>`I 1`*{cr}*
`19/37>`

Settings

S n [newvalue]

- *n* is a setting number listed in the table below.
- *newvalue* is the new value to assign to the setting. If *newvalue* is not provided, then the current value for the setting is returned.

Description
The Setting command will display or assign a value to a setting. There are several different settings on the uMMC and are listed in the table below.

Table 1 – Setting Numbers and Values

Setting Number	Name	Value	Description (**default**)
0	Serial Bit Rate	**0**	**9600 bps**
		1	19200 bps
		2	38400 bps
		3	57600 bps
		4	115200 bps
1	Write/Append Time-out	**0** to 254	Time in 10ms increments (eg. 20 = 200ms) **0 = No time-out (waits indefinitely)**
2+	Reserved	-	-

Example
```
>S 1{cr}
0>S 1 22{cr}
> S 1{cr}
22>
```

Make Directory

M path

- *path* is the absolute path to the directory to be created. A properly formatted path must begin with a "/" and be absolute (that is, it must begin at the root directory). Sub-directories are separated with "/" (forward slash).

Description
The ***Make Directory*** command will create a directory.

Example
To create a directory named "JANUARY" under the path "/LOGS/2004":

```
>M /LOGS/2004/JANUARY{cr}
>
```

Erase File

E path

- *path* is the absolute path to the file to be erased. A properly formatted path must begin with a "/" and be absolute (that is, it must begin at the root directory). Sub-directories are separated with "/" (forward slash).

Description
The ***Erase File*** command will erase a file.

Example
To erase a file named "JAN03.LOG" under the path "/LOGS/2004/JANUARY":

```
>E /LOGS/2004/JANUARY/JAN03.LOG{cr}
>
```

Status

Z

Description
The ***Status*** command will return the current status of the file system. If the file system is initialized, then a single *{sp}* is returned. Otherwise, an error code from the Error Response Code list is returned.

Example
>Z*{cr}*
>

Query Volume

Q

Description
The ***Query Volume*** command will return the free space and the total volume of the current memory card. The format is **`freespace/totalspace`**. The two values are given in decimal format and it kilobytes (i.e. 1024 bytes per kilobyte).

Example
>Q*{cr}*
51245/61525>

Version/Serial Number

V

Description
The ***Version/Serial Number*** command will return the current firmware version and the uMMC serial number. The format is VVV.MM SN:UMM1-NNNN-NNNN.

Example
>V*{cr}*
101.42 SN:UMM1-0408-0001>

Example Session

This example session opens a log file, writes data, and then retrieves it.

```
>O 1 W /LOGS/2004/JANUARY/JAN03.LOG{cr}
>W 1 18{cr}
13:22:02 ADC1=4.9V>C 1{cr}
>O 1 R /LOGS/2004/JANUARY/JAN03.LOG{cr}
>R 1 18{cr}
13:22:02 ADC1=4.9V>C 1{cr}
>
```

Error Responses

When a command produces an error, the response is **Enn**. The table of responses is given below.

Below is an example error response from an open command that attempts to open a file that does not exist.

Example:
>`O 1 R /LOGS/2004/FEBRUARY/FEB30.LOG`*{cr}*
EF2>

Table 2 – Table of Error Response Codes

Error Code	Description
E02	Buffer Overrun – Too many bytes were sent in the command. All command can be a maximum of 256 bytes (including the path).
E03	No Free Files – This is a response from the ***Free File*** command. There are no more open handles. You must close an open file handle before a new one can be opened.
E04	Unrecognized command.
E06	Command formatting error – this occurs if parameters are missing or invalid.
E07	End of file
E08	Card not inserted
E09	MMC/SD Reset failure
E0A	Card write protected
EE6	Read-only file – a Read-Only file (file attributes) is trying to be opened for write or append.
EE7	Not a file – an invalid path.
EE8	Write Failure – There could be many reasons for this (damaged card, card removed WHILE writing, etc…)
EEA	No free space – There is no free space on the card.
EEB	File not open – The file handle specified has not been opened with the ***Open*** command.
EEC	Improper mode – A ***Read*** command was attempted while the file has been opened for writing, or vice-versa.
EED	Invalid ***Open*** mode – only ‘R’, ‘W’, and ‘A’ are acceptable open modes.
EF1	Handle in use – The specified handle is already being used.
EF2	File does not exist – The file in the path specified does not exist.

EF4	File already exists – A ***Write*** command was issued, and the file in the path already exists.
EF5	Path invalid – The path specified does not exist. Ensure that all directory names in the path exist.
EF6	Invalid handle – The handle specified is not valid.
EFB	Bad FSINFO Sector (FAT32 only)
EFC	Unsupported FAT version. Ensure the card is inserted correctly and that the card has been formatted to FAT16 or FAT32.
EFD	Unsupported Partition type
EFE	Bad Partition information
EFF	Unknown Error

Notices

This document may contain technical inaccuracies or typographical errors. The information in this document may change without notice. Changes and improvements to the products may occur at any time.

This document is provided “as-is” and without warranty of any kind, either express or implied.

Copyright Information

Trademarks

uMMC is a trademark of Rogue Robotics Corporation Inc.

Life Support Policy

Rogue Robotics Corporation Inc. products are not intended or authorized for use in any life support situation. These systems may include devices for sustaining life, surgical implant into or onto the body, or any other system whose failure to perform correctly could result in life support failure.

Contact and Support

Contact:
Rogue Robotics Corporation
103 Sarah Ashbridge Ave.
Toronto, ON Canada M4L 3Y1

Support:
support@roguerobotics.com
(416) 707-3745
(647) 439-1577 Fax