

716-52 Multi Sector MIFARE® Reader/Writer

Data Sheet

Overview

The 716-52 multi sector MIFARE® Reader/Writer is an intelligent 13.56MHz contactless reader/writer designed to operate with the Mifare® Std 1k and 4k cards.

The reader/writer uses a PHILIPS MIFARE® reader IC and complies to ISO 14443A.

The multi sector MIFARE® Reader/Writer can read and write¹ to any sector and block on the card and is MAD compliant. All the encryption needed to provide a secure and reliable transaction is handled automatically with a few simple commands.

An RS232 interface is used to communicate between the unit and a PC or application hardware.

Three independent LEDs and a beeper are available for user tasks.

The product consists of three parts, a potted unit containing the electronics and antenna, a front cover, and an optional spacer plate. A fixed 10 way colour-coded cable protrudes from the back of the potted unit.



Features

- Reads MIFARE® cards unique ID number, UID
- Reader can store 32 keys for use as A or B keys
- Reads data from any sector/block on a MIFARE® Std 1k or 4k card
- Writes¹ data to any sector/block on a MIFARE® Std 1k or 4k card
- Supports MAD1 and MAD2
- Commands to support transactions using the MIFARE® VALUE² format
- Format checking on access bytes to prevent inadvertent damage to card
- Uses on-card MIFARE® security keys to prevent card copying and tampering

Applications

- cashless vending machines
- squash court access and lights control
- meal voucher control
- laundry machines
- loyalty programs
- cashless payments
- biometric data storage
- access control

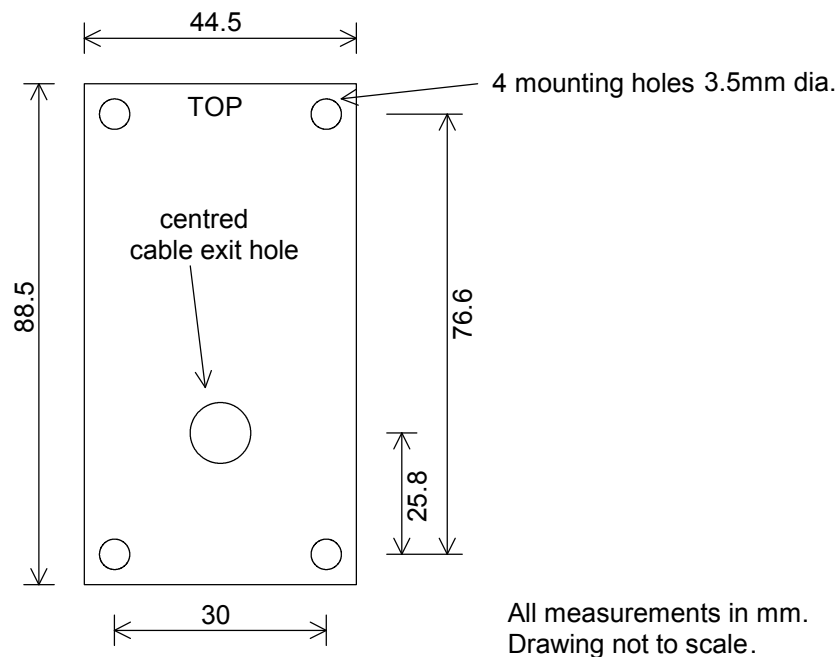
Note 1. The manufacturer's data, sector 0 block 0, is read only.

Note 2. The MIFARE^a VALUE format is a fixed data format for a sector block that permits error detection, correction and backup management. The size of the data is a signed 4 byte value, for reasons of data integrity and security a value is stored 3 times. A 1 byte address, stored 4 times is also stored in the sector block and is used when implementing backup management.

Specifications

- Power requirements: 5.0 to 13.8V dc
- Current consumption: 100 mA (typical)
- RF Frequency: 13.56MHz
- Typical reading range - 20mm to 50mm, dependant upon card manufacturer
- 3 LEDs (GREEN, RED, YELLOW).
- Beeper emits a 4kHz tone
- Operating temperature range: -20°C to +60°C.
- 10 way cable: 1m long
- Weight: 90 grams.
- Dimensions: reader 89 x 45 x 16 mm, optional spacer plate 89 x 45 x 7 mm

Physical Dimensions and Mounting Details



If the spacer plate is used the reader cable may be brought out of one of four exit points on the spacer: top, bottom, left or right. This enables the cable to be run on the surface of the wall. If no spacer plate is used a minimum hole size of 6.5mm must be drilled in the wall at the cable exit position as shown above to allow the cable to exit perpendicular to the reader.

The optional spacer plate may also be used when mounting the reader on a metal surface to reduce the negative effects of metal on the read range.

Connections

The table below details the function of each wire:

Colour	Name	Function
BROWN	TTL ³ RX	TTL RX comms
WHITE	TTL ³ TX	TTL TX comms
BLUE	COMMS GND	TTL/RS232 comms ground
GREEN	RS232 RX	RS232 RX comms
YELLOW	RS232 TX	RS232 TX comms
RED	+VDC ⁴	Connect +5V +13.8V to power supply
BLACK	0V	Connect 0V to power supply

Note 3. TTL comms bypasses the RS232 level shifting and inverting IC and offers the user the ability to connect the application microprocessor directly to the Reader Writer's microprocessor.

Note 4. The Reader Writer has an internal low dropout 5V regulator. For supply voltages less than 5.5V the user must provide a regulated DC with RMS ripple and noise levels below 1mV.

RS232 Communication

There are two alternate full duplex serial communication connections to the Reader Writer, RS232 or TTL. The RS232 conforms to the EIA standard and can be connected directly to the serial COM port of a PC. The TTL connections are non-inverted 0V and 5V levels and are intended to be connected directly to the USART of the application microprocessor.

The baud rate is 19200. Data format is 8 bits, no parity, and 1 stop bit.

Command format

Commands and replies are sent in ASCII and are case sensitive, only keyboard characters are used. Data is sent as ascii chars. A terminal program (HyperTerminal) can be used to send commands and receive replies.

Two COMMAND FORMATS are provided for, one with an appended checksum and one without, either may be used

With checksum the format is:

`<$><A>,<C>,<d1>,<d2>,...,<CHK><CR><LF>`

Without checksum the format is:

`<!><A>,<C>,<d1>,<d2>,...<CR><LF>`

where,

- `<$>` command header that indicates start of the command with checksum
- `<!>` command header that indicates start of the command without checksum
- `<A>` is a 1 digit target address code in decimal, this is always 1 for the Reader Writer
- `<C>` is one or more command identification characters
- `<d1> <d2> ...` are optional command parameters in decimal (or hex if preceded by 0x)
- `<CHK>` is the checksum, where sent,
checksum formatted as 'c-style' hex number e.g. 0xA3
- `<CR>` is a carriage return \r <enter>
- `<LF>` is a line feed \n - optional and is ignored

the ',' is used as a delimiter between characters

The checksum is calculated by performing an 8 bit addition of the ascii value of each char making up the command and includes the start char and comma delimiters but not the checksum itself.

REPLY FORMAT

The reply always includes a checksum, the format is:

<\$><A>,<REPLY>,<y..y>.....,<CHK><CR><LF>

where,

- <\$> reply header that indicates start of the reply
- <A> is a 1 digit target address code in decimal, this is always 0 for the application hardware
- <REPLY> is the reply, may be, OK for a successfully implemented command, ERROR xx for any failure in deciphering or executing the command (where xx is the error code) or a command identification character
- <y..y> any parameters used by the reply
- <CHK> is the checksum,
checksum formatted as 'c-style' hex number e.g. 0x0B
- <CR> is a carriage return \r <enter>
- <LF> is a line feed \n - optional and is ignored

the ',' is used as a delimiter between characters

The checksum is calculated by performing an 8 bit addition of the ascii value of each char making up the command and includes the start char and comma delimiters but not the checksum itself.

Note. Where a hex value is given or requested, two chars always follow the 0x
Example - for hex value 1, 0x1 is illegal, it is required to be 0x01

The reader/writer has a buffer which is cleared and starts filling after the header has been received, the command is processed after a <CR> is received. All commands are replied to.

Command Set

Note. For ease of reading the commands and replies are shown here without the start character, address code, checksum and <CR><LF>

Query Commands

Return Version Information

I

Function. Outputs both hardware and firmware version information

- Reply list: 1. Up to 20 ascii chars of version information, example 716-52 rev1 v1.00
2. ERROR xx

Control Commands

Bootloader Command

L

- Reply List: 1. OK
2. ERROR xx

To prevent unintentionally entering the bootloader mode and subsequently erasing the current firmware in flash this command will only be accepted by the reader/writer if sent with the checksum.

Note. Once this command has been accepted the current firmware in flash has been erased and no communication is now possible. The reader/writer will remain in this mode until new firmware has been downloaded by using proprietary bootloader software.

Software Reset (Clear)

C

Function. Initiates a microprocessor software reset

Reply list: 1. OK (before resetting)
2. ERROR xx

Control Beeper

B,tttt

Function: Turns on the beeper for tttt ms

where: tttt is the beeper on time in milliseconds entered as decimal ascii chars, will accept any time from 0ms to 9999ms

example B,1 gives 1ms beep time

B,0 turns beeper off

Reply list: 1. OK
2. ERROR xx

Control RF Field

F,x

Function. Turns RF field either on or off

where: x is RF field state

0=OFF

1=ON

Reply list: 1. OK
2. ERROR xx

Control Green LED

G,x

Function. Turns Green LED either on or off

where: x is Green LED state

0=OFF

1=ON

Reply list: 1. OK
2. ERROR xx

Control Red(Scarlet) LED

S,x

Function. Turns Red LED either on or off

where: x is Red LED state

0=OFF

1=ON

Reply list: 1. OK
2. ERROR xx

Control Yellow LED

Y,x

Function. Turns Yellow LED either on or off

where: x is Yellow LED state

0=OFF

1=ON

Reply list: 1. OK
2. ERROR xx

where bb is the block number in decimal
range $00 \leq bb \leq 03$ for MIFARE[®] Std 1k and 4k cards
range $04 \leq bb \leq 15$ for MIFARE[®] Std 4k cards

where k is the key type
range k=A or k=B

where ii is the key index in decimal
range $00 \leq ii \leq 31$

Data is in hexadecimal

Reply List: 1.OK
2. ERROR xx

Note. Any block, except the manufacturer's block, sector 0 block 0, can be written to. The sector trailer's will only be written if access bytes are in the correct format.

Read VALUE² Sector Block

V,ss,bb,k,ii

Function. Reads 4 bytes of data in the MIFARE^a VALUE² format from the addressed sector block ss, bb of the MIFARE^a card, uses the given key type k found at the key index ii

where ss is the sector number in decimal
range $00 \leq ss \leq 15$ for MIFARE[®] Std 1k and 4k cards
range $16 \leq ss \leq 39$ for MIFARE[®] Std 4k cards

where bb is the block number in decimal
range $00 \leq bb \leq 02$ for MIFARE[®] Std 1k and 4k cards
range $04 \leq bb \leq 14$ for MIFARE[®] Std 4k cards

where k is the key type
range k=A or k=B

where ii is the key index in decimal
range $00 \leq ii \leq 31$

Reply List:
1. V,ss,bb,0xhhhhhhhh(a 4 byte signed hex number)
2. ERROR xx

Read VALUE² AID Sector Block

MV,0xaaaa,bb,k,ii

Function. Reads 4 bytes of data in the MIFARE^a VALUE² format from the sector block addressed by the AID 0xaaaa and the block bb, follows MAD rules (MAD1 and MAD2 are supported), uses the given key type k found at the key index ii

where 0xaaaa is the 2 byte AID in hexadecimal

where bb is the block number in decimal
range $00 \leq bb \leq 03$ for MIFARE[®] Std 1k and 4k cards
range $04 \leq bb \leq 15$ for MIFARE[®] Std 4k cards

where bb is the block number in decimal
range $00 \leq bb \leq 02$ for MIFARE[®] Std 1k and 4k cards
range $04 \leq bb \leq 14$ for MIFARE[®] Std 4k cards

where k is the key type
range k=A or k=B

where ii is the key index in decimal
range $00 \leq ii \leq 31$

Reply List:

1. V,ss,bb,0xhhhhhhh(a 4 byte signed hex number)
2. ERROR xx

where ss is the sector number in decimal
range $00 \leq ss \leq 15$ for MIFARE[®] Std 1k and 4k cards
range $16 \leq ss \leq 39$ for MIFARE[®] Std 4k cards

Write VALUE² Sector Block

X,ss,bb,k,ii,0xhhhhhhh

Function. Writes 4 bytes of data in the MIFARE^a VALUE² format to the addressed sector block, ss,bb, of the MIFARE^a card using the given key type k found at the key index ii.

where ss is the sector number in decimal
range $00 \leq ss \leq 15$ for MIFARE[®] Std 1k and 4k cards
range $16 \leq ss \leq 39$ for MIFARE[®] Std 4k cards

where bb is the block number in decimal
range $00 \leq bb \leq 02$ for MIFARE[®] Std 1k and 4k cards
range $04 \leq bb \leq 14$ for MIFARE[®] Std 4k cards

where k is the key type
range k=A or k=B

where ii is the key index in decimal
range $00 \leq ii \leq 31$

where 0xhhhhhhh is a 4 byte signed hexadecimal number, negative numbers are represented in 2s complement

- Reply List: 1.OK
2. ERROR xx

Write VALUE² AID Sector Block

MX,0xaaaa,bb,k,ii,0xhhhhhhh

Function. Writes 4 bytes of data in the MIFARE^a VALUE² format to the sector block addressed by the AID 0xaaaa and the block bb, follows MAD rules (MAD1 and MAD2 are supported), uses the given key type k found at the key index ii

where 0xaaaa is the 2 byte AID in hexadecimal

where bb is the block number in decimal
range $00 \leq bb \leq 02$ for MIFARE[®] Std 1k and 4k cards
range $04 \leq bb \leq 14$ for MIFARE[®] Std 4k cards

where k is the key type
range k=A or k=B

where ii is the key index in decimal
range $00 \leq ii \leq 31$

where 0xhhhhhhh is a 4 byte signed hex number, negative numbers are represented in 2s complement

Reply List: 1.OK
2. ERROR xx

Decrement VALUE² Sector Block

D,ss,bb,k,ii,0xhhhhhhhh

Function. The 4 bytes of hex data will be subtracted from the VALUE² found at the addressed sector block, ss,bb, of the MIFARE^a card using the given key type k found at the key index ii.

where ss is the sector number in decimal
range $00 \leq ss \leq 15$ for MIFARE[®] Std 1k and 4k cards
range $16 \leq ss \leq 39$ for MIFARE[®] Std 4k cards

where bb is the block number in decimal
range $00 \leq bb \leq 02$ for MIFARE[®] Std 1k and 4k cards
range $04 \leq bb \leq 14$ for MIFARE[®] Std 4k cards

where k is the key type
range k=A or k=B

where ii is the key index in decimal
range $00 \leq ii \leq 31$

where 0xhhhhhhhh is a 4 byte signed hex number, negative numbers are not permitted (this effectively limits numbers to 31 bits).

Reply List: 1.OK
2. ERROR xx

Decrement VALUE² AID Sector Block

MD,0xaaaa,bb,k,ii,0xhhhhhhhh

Function. The 4 bytes of data will be subtracted from the VALUE² found at the sector block addressed by the AID 0xaaaa and the block bb, follows MAD rules (MAD1 and MAD2 are supported), uses the given key type k found at the key index ii

where 0xaaaa is the 2 byte AID in hexadecimal

where bb is the block number in decimal
range $00 \leq bb \leq 02$ for MIFARE[®] Std 1k and 4k cards
range $04 \leq bb \leq 14$ for MIFARE[®] Std 4k cards

where k is the key type
range k=A or k=B

where ii is the key index in decimal
range $00 \leq ii \leq 31$

where 0xhhhhhhhh is a 4 byte signed hex number, negative numbers are not permitted (this effectively limits numbers to 31 bits).

Reply List: 1.OK
2. ERROR xx

Increment VALUE² Sector Block

A,ss,bb,k,ii,0xhhhhhhhh

Function. The 4 bytes of hex data will be added to the VALUE² found at the addressed sector block, ss,bb, of the MIFARE^a card using the given key type k found at the key index ii.

where ss is the sector number in decimal
range $00 \leq ss \leq 15$ for MIFARE[®] Std 1k and 4k cards

range $16 \leq ss \leq 39$ for MIFARE[®] Std 4k cards

where bb is the block number in decimal

range $00 \leq bb \leq 03$ for MIFARE[®] Std 1k and 4k cards

range $04 \leq bb \leq 15$ for MIFARE[®] Std 4k cards

where k is the key type

range k=A or k=B

where ii is the key index in decimal

range $00 \leq ii \leq 31$

where 0xhhhhhhhh is a 4 byte signed hex number, negative numbers are not permitted (this effectively limits numbers to 31 bits).

Reply List: 1.OK

2. ERROR xx

Increment VALUE² AID Sector Block

MA,0xaaaa,bb,k,ii,0xhhhhhhhh

Function. The 4 bytes of data will be added to the VALUE² found at the sector block addressed by the AID 0xaaaa and the block bb, follows MAD rules (MAD1 and MAD2 are supported), uses the given key type k found at the key index ii

where 0xaaaa is the 2 byte AID in hexadecimal

where bb is the block number in decimal

range $00 \leq bb \leq 03$ for MIFARE[®] Std 1k and 4k cards

range $04 \leq bb \leq 15$ for MIFARE[®] Std 4k cards

where k is the key type

range k=A or k=B

where ii is the key index in decimal

range $00 \leq ii \leq 31$

where 0xhhhhhhhh is a 4 byte signed hex number, negative numbers are not permitted (this effectively limits numbers to 31 bits).

Reply List: 1.OK

2. ERROR xx

Write VALUE² Sector Block

X,ss,bb,k,ii,0xhhhhhhhh

Function. Writes 4 bytes of hex data in the MIFARE^a VALUE² format to the addressed sector block, ss,bb, of the MIFARE^a card using the given key type k found at the key index ii.

where ss is the sector number in decimal

range $00 \leq ss \leq 15$ for MIFARE[®] Std 1k and 4k cards

range $16 \leq ss \leq 39$ for MIFARE[®] Std 4k cards

where bb is the block number in decimal

range $00 \leq bb \leq 03$ for MIFARE[®] Std 1k and 4k cards

range $04 \leq bb \leq 15$ for MIFARE[®] Std 4k cards

where k is the key type

range k=A or k=B

where ii is the key index in decimal

range $00 \leq ii \leq 31$

where 0xhhhhhhh is a 4 byte signed hex number, negative numbers are not permitted (this effectively limits numbers to 31 bits).

- Reply List: 1.OK
2. ERROR xx

Note: The manufacturer's block, sector 0 block 0 and all the sector trailer's, block 3, cannot be used in the MIFARE^a VALUE² format.

Write VALUE² AID Sector Block

MX,0xaaaa,bb,k,ii,0xhhhhhhh

Function. Writes 4 bytes of data in the MIFARE^a VALUE² format to the sector block addressed by the AID 0xaaaa and the block bb, follows MAD rules (MAD1 and MAD2 are supported), uses the given key type k found at the key index ii

where 0xaaaa is the 2 byte AID in hexadecimal

where bb is the block number in decimal
range $00 \leq bb \leq 03$ for MIFARE[®] Std 1k and 4k cards
range $04 \leq bb \leq 15$ for MIFARE[®] Std 4k cards

where k is the key type
range k=A or k=B

where ii is the key index in decimal
range $00 \leq ii \leq 31$

where 0xhhhhhhh is a 4 byte signed hex number, negative numbers are not permitted (this effectively limits numbers to 31 bits).

- Reply List: 1.OK
2. ERROR xx

Note: The manufacturer's block, sector 0 block 0 and all the sector trailer's, block 3, cannot be used in the MIFARE^a VALUE² format.

Note. An OK reply does not necessarily guarantee that the transaction took place successfully, it is up to the user application to validate this.

Error Codes

The table shows the error codes returned when a transaction with a MIFARE[®] card fails

Error Code	Error Message
1	No card in the field
2	Communication error
3	Security error
4	Corrupt Value
5	Transaction value negative
6	Transaction failed
7	Command format error
8	MAD transaction error