ib technology

Data Sheet

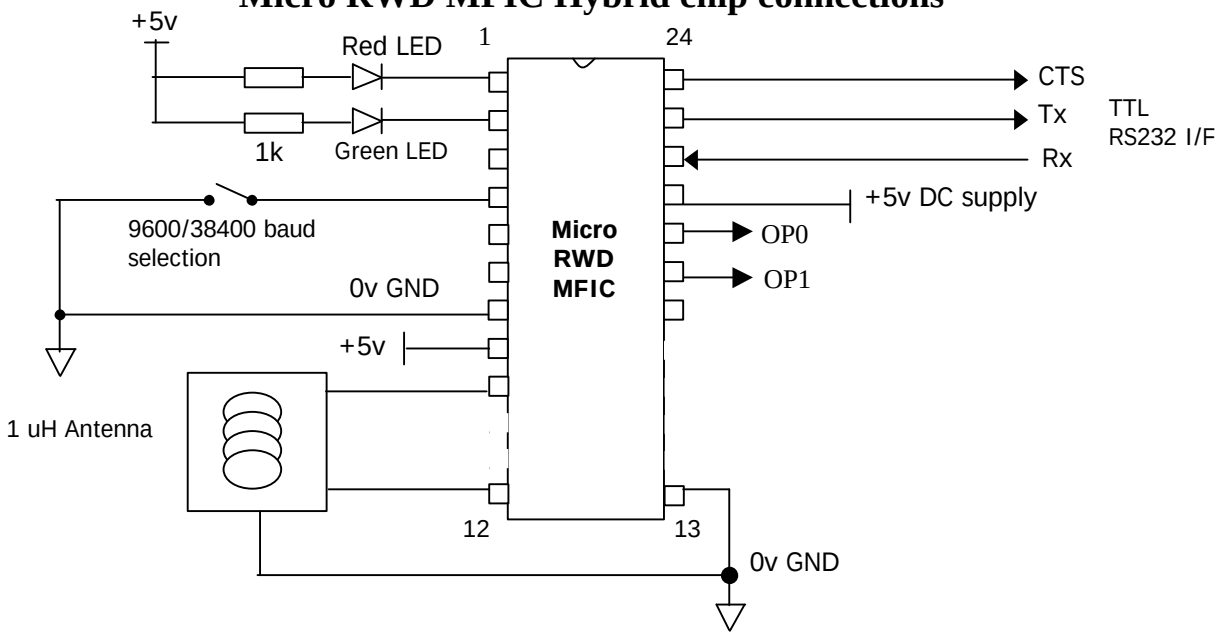
MF_ICprot.pdf (preliminary)
33 Pages
Last Revised 27/06/06

Micro RWD MFIC (Mifare and ICODE SLI) Reader

The MicroRWD MFIC module is a complete reader and transponder acceptance solution for 13.56 MHz Mifare (1k and 4k byte versions), Mifare Ultralight cards and ICODE SLI smart labels. In addition the module can acquire the single and double UID (serial numbers) and perform some limited read/write operations to DESFire, Mifare ProX, SmartMX and other ISO14443A card types. The solution is entirely housed within a 24 pin DIL package and only needs an antenna connected and 5v DC supply to be a fully featured ISO14443A Mifare and ISO15693 ICODE SLI read/write system. The module has internal EEPROM memory for holding Authentication keys, lists of authorised serial numbers and LED drives to give visual indication of acceptance. In addition, auxiliary outputs are provided for customised interfaces and for control purposes such as driving relays.

The RWD (**R**ead/**W**rite **D**evice) also has a TTL level RS232 interface that allows a host system to communicate with the RWD, so that system features can be customised, configurations changed and transponder read/write data can be handled by the host system. **The MicroRWD MFIC module is pin compatible and host command compatible with the 125kHz family of MicroRWD (Hitag and EM) modules to allow easy migration between these different technologies.**

Micro RWD MFIC Hybrid chip connections



Mifare or ICODE operational modes are selected by setting parameter byte 3 in the MicroRWD MFIC internal EEPROM memory (see page 7 for RWD EEPROM memory map).

Mifare Transponders

The Mifare transponder cards have significantly more memory than most other contactless cards and the 13.56 MHz carrier frequency provides fast transaction times of 106 kbaud.
The cards are available with 64 bytes (Mifare Ultralight), 1024 bytes (Mifare 1k) and 4096 bytes (Mifare 4k) of memory. For the 1k and 4k cards the memory is organised as 16 and 32 Sectors respectively, each Sector has 64 bytes arranged as 4 Blocks of memory (3 of which are available for general Read/Write use). Each Sector can be separately locked/unlocked for access using security keys.
Initial communication with the cards can only proceed after mutual authentication between the RWD and the card has succeeded (as defined by ISO 14443A standard). Combined with the "Security Key" access control for the memory sectors and encrypted data streams, the Mifare cards are ideally suited to Electronic-Purse applications such as ticketing and vending applications where each sector can hold entirely separate data for different applications. Outside of these markets, the large memory sizes, fast transaction times and high security access means that Mifare cards can be used for almost any application.

ICODE SLI Transponders

The ICODE SLI transponders support the ISO15693 smart label standard and have 128 bytes of memory organised as 4 byte blocks. Fast communication times of up to 52kbaud and the low cost of these transponders has allowed their use for asset tracking similar and applications where the transponder is packaged as a flexible label. ICODE SLI supports multiple transponder operation and anti-collision handling so several transponders can be identified in the RF field and subsequently communicated with.

The MicroRWD MFIC is a proximity system and a Read/Write range of up to 10cm can be achieved under ideal conditions using the appropriate antenna. For evaluation purposes the RWD is available on a base board with LEDs and 9-pin RS232 socket fitted. When power (5v DC) is first applied to the board the red and green LEDs flash once to indicate successful power-up (both LEDs stay on if initialisation fails). The RWD can also check for internal faults and error conditions, these problems are indicated by the red LED or both LEDs flashing continuously until the fault has been rectified.

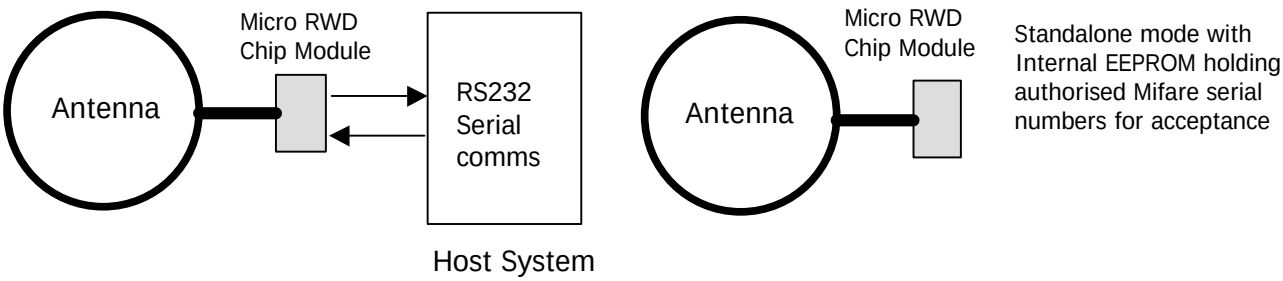
The RWD will normally have the red LED lit until a valid card is brought into the RF field. If the tag is accepted as valid then the green LED is lit and the output drivers (OP0 and OP1) are switched on. These outputs can be connected together to give up to 50ma of drive current for operating external devices etc. In addition, a switch input is provided for selecting high-speed (38400 baud) host communication; default host speed is 9600 baud.

Note: Some ISO14443A compliant cards have a SINGLE (4 byte) UID and others have a DOUBLE (7 byte) UID. These serial numbers are acquired as part of the initial anticollision/select procedure when a card is brought into the field. This UID information can be reported using the CARD UID command. For Mifare Classic 1k and 4k cards the SINGLE UID is acquired and can be reported BUT subsequent block read/write operations will not function if the Security KEY is incorrect for the particular sector and the authentication fails.

For many applications where only the UID (serial number) is required, the Security KEY information is therefore NOT required.

MicroRWD MFIC modes of operation

The Micro RWD has two basic modes of operation:-



Remote mode (connected to a host computer or microcontroller) and Standalone mode.

1) Remote mode involves connecting to a host serial interface. This is where the stored list of authorised identity codes (serial numbers) can be empty, effectively authorising any Mifare/ICODE card for subsequent read/write operations (depending on correct Security Key in Mifare case). A simple serial protocol allows a host system to communicate with the Micro RWD in order to program new authorised identity codes, change parameters, load Security Keys and perform Read/Write operations to the card itself.

2) Standalone mode is where the Mifare/ICODE card identity codes (serial numbers) are checked against a stored list of authorised codes. If an identity code is matched, the output drives and Green LED are enabled. Effectively standalone mode occurs when there is no host system communicating with the Micro RWD. Up to 60 serial numbers can be stored in the authorisation list so this mode of operation can be used to create a "mini access control" system.

Supported transponder types

The MicroRWD MFIC is designed to communicate with the following passive RF transponder types:-

Mifare Mode

Selected by RWD EEPROM parameter byte 3 set to 0x00.

Note that Mifare cards and transponder devices are fabricated by several companies under licence from Philips Semiconductors. They are all fully Mifare compliant and only differ in having different default Security Keys loaded in the factory:-

1) Mifare standard 1k card (MF1 IC S50 transponder) and Infineon equivalent.

2) Mifare standard 4k card (MF1 IC S70 transponder)

3) Mifare Ultralight card (MF0 IC U1 transponder).

4) DESFire, Mifare ProX, SmartMX contactless and dual-interface card types are supported to allow single or double UID to be acquired. Some types may allow some read/write operation depending on Operating System installed on card.

5) Any ISO 14443A compliant contactless card can be accessed for Serial Number. Full Read/Write access will only be possible if card fully supports Philips Semiconductors CRYPTO1 algorithm and encrypted data protocols.

The operation of the Micro RWD MF and the Mifare transponders is described in more detail at the end of this document.
The identification codes described in this text are regarded as the four byte (SINGLE) Mifare UID (Unique Identifier/serial number) or the least significant four bytes of the seven byte (DOUBLE) Ultralight UID.

ICODE SLI Mode

Selected by RWD EEPROM parameter byte 3 set to 0x01.

Note that ICODE SLI labels are designed to comply with the ISO15693 standard. Other manufacturers such as ST, EM Marin and TI also have ISO15693 smart labels but they often have different memory sizes and support sub-sets of the ISO15693 command protocol. The MicroRWD MFIC has been designed to work with the most common ICODE SLI type from Philips Semiconductors.

1) ICODE SLI (ISO15693) smart labels (Philips Semiconductors SL2 ICS20)

2) Any ISO15693 smart label that supports the core ISO15693 command set and has the same memory structure and configuration bytes as the ICODE SLI type.

The ICODE ident code is defined as the least significant four bytes of the 8 byte UID, effectively UID0 - UID3). Note that The UID used for the "Ident list" check is the first tag UID acquired when there are multiple tags in the field (first tag in INVENTORY list).

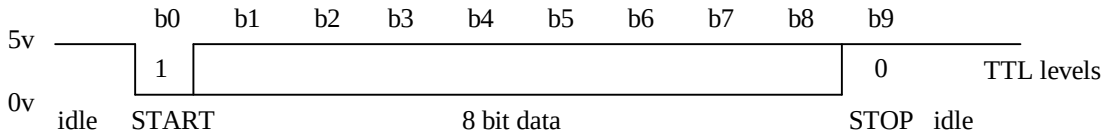
IMPORTANT NOTE: DUE TO SLIGHT DIFFERENCES IN THE RF CHARACTERISTICS OF MIFARE AND ICODE CARDS/LABELS, THE ANTENNA TUNING MAY NEED ADJUSTING WHEN SWITCHING BETWEEN MIFARE AND ICODE OPERATION.

Serial Interface

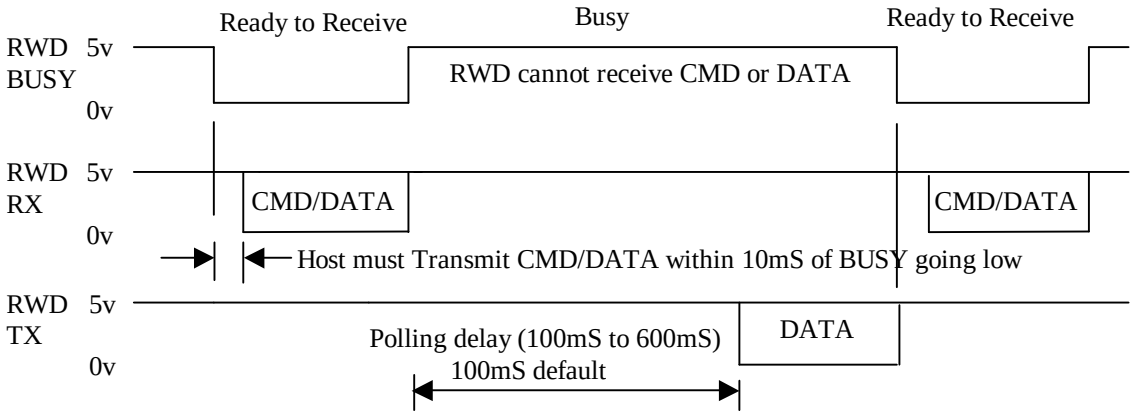
This is a basic implementation of RS232. The Micro RWD does not support buffered interrupt driven input so it must control a BUSY (CTS) line to inhibit communications from the host when it is fully occupied with card communication. It is assumed that the host (such as a PC) can buffer received data.
Tx, Rx and RTS signals from the Micro RWD are all TTL level and can be converted to +/- 10v RS232 levels using an inverting level converter device such as the MAX202 (note the inversion of the TTL levels).

The serial communication system and protocol allows for a 10ms 'window' every card polling cycle indicated by the BUSY line being low. During this 'window' the host must assert the first start bit and start transmitting data. The BUSY goes high again 10ms after the last stop bit is received. NOTE that only one command sequence is handled at a time.
The SW1 switch input allows for manual selection of a higher host communication baud rate (38400 baud); the default with SW1 unconnected is 9600 baud, 8 bits, 1 stop, no parity.

Transmitted or Received data byte, 9600/38400 baud, 8 bit, 1 stop, No parity (104 / 26 uS per bit)



RWD tag polling cycle and serial communication BUSY protocol



NOTE THAT THE FALLING-EDGE OF THE GREEN LED OUTPUT CAN BE USED AS AN INTERRUPT SIGNAL TO TRIGGER HOST COMMUNICATION RATHER THAN USING CONTINUOUS COMMAND POLLING.

Host Driver software

Communication with the MicroRWD module is via the TTL level RS232 interface (9600 baud, 8 bit, 1 stop bit, no parity) and uses the CTS line for hardware handshaking. The Windows applications (supplied with the Evaluation kit) can be used to communicate with the module or the user can write their own application on a PC or a microcontroller. The following basic communication algorithm should be used:-

Typical host computer "pseudo" driver code

```
if (Green LED ON (pin 2 = 0))                  // Optional check for valid tag in field
{
   if (CTS = 0)          // Wait for CTS = 0 (RWD ready to receive command / data)
  {
            // CTS times out after 10ms so command and all parameters must be sent with no-
            // gaps otherwise CTS times out and goes HIGH.
            // For example, send READ BLOCK 1 using KEY 0 as KEYA (0x52 0x01 0x00)

            SEND_CMD( );              // Sent command + parameters to RWD

            // RWD sets CTS = 1 after last parameter received. RWD module enters low-
            // power state during (programmable, default 100ms) polling delay then sends reply.

            // NOTE THAT THE REPLY IS AFTER THE POLLING DELAY

            GET_REPLY( );             // Get Acknowledge byte + data
            // Response to READ command is 0x80 (no tag) or 0x86 + sixteen bytes of DATA.
   }
}
```

Commands for both MIFARE and ICODE modes

These commands are common to both RWD Reader modes and have the same function, structure and arguments no matter which mode is selected.

Card / Label STATUS

Command to return card status.
The acknowledge byte flags indicate general Mifare/ICODE card status.

	B7						B0	
Command:	0	1	0	1	0	0	1	1	(Ascii "S", 0x53)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

MESSAGE Report

Command to return product and firmware identifier string to host.

	B7						B0	
Command:	0	1	1	1	1	0	1	0	(Ascii "z", 0x7A)

Reply (Mifare): "m IDE RWD Mifare (SECMFI_C V2.xx DD/MM/YY) IB Technology Ltd accepts no liability for the use of this product in any end application" 0x00

Reply (ICODE): "i IDE RWD ICODE (SECMFI_C V2.xx DD/MM/YY) IB Technology Ltd accepts no liability for the use of this product in any end application" 0x00

Returned string identifies product descriptor, project name, firmware version no. and date of last software change together with IB Technology disclaimer message. Note that the string is always NULL terminated. The string begins with a unique lower case character that can be used to identify a particular version of Micro RWD.

Program EEPROM

The Micro RWD has some internal EEPROM for storing system parameters such as polling rate and authorised identity codes (serial numbers). This command sequence allows individual bytes of the EEPROM to be programmed with new data. The data is internally read back after programming to verify successful operation. Note that due to the fundamental nature of these system parameters, incorrect data may render the system temporarily inoperable.

	B7							B0	
Command:	0	1	0	1	0	0	0	0	(Ascii "P", 0x50)
Argument1:	N	N	N	N	N	N	N	N	(N = EEPROM memory location 0 - 255)
Argument2:	D	D	D	D	D	D	D	D	(D = data to write to EEPROM)
Acknowledge:	1	X	X	X	F	X	X	F	(F = Status flags)

Internal EEPROM memory map

Byte 0: Tag Polling Rate (x 2.5ms) (Default = 0x32 = approx 100ms)
Byte 1: Reserved
Byte 2: Reserved (Checksum)
Byte 3: Mifare / ICODE Option byte, (Default = 0x01 = ICODE mode, 0x00 = Mifare mode)

Byte 4: Reserved
Byte 5: Reserved
Byte 6: Reserved
Byte 7: Reserved

Byte 8: Reserved
Byte 9: Reserved
Byte 10: Reserved
Byte 11: Reserved

Start of authorised card codes. List is terminated with FF FF FF FF sequence.
List is regarded as empty (all identity codes valid) if first code sequence in list is (FF FF FF FF).
List can hold up to 60 identity codes (serial numbers)

Byte 12: 0xFF Empty list
Byte 13: 0xFF
Byte 14: 0xFF
Byte 15: 0xFF

Byte 16: (MSB) Tag identity code
Byte 17:
Byte 18:
Byte 19: (LSB)

-
-
-
Byte 255: Last Internal EEPROM location

Command Protocol (Mifare Mode)

The following commands are supported in Mifare mode. The corresponding acknowledge code should be read back by the host and decoded to confirm that the command was received and handled correctly. The serial bit protocol is 9600/38400 baud, 8 bits, 1 stop, no parity (lsb transmitted first).

The status flags returned in the Acknowledge byte are as follows:

```
 b7  b6  b5  b4  b3 b2 b1 b0
 1   1   1   1   1  1  1  1
     |   |   |   |  |  |   EEPROM error (Internal EEPROM write error)
     |   |   |   |  |  Card OK (Card serial number matched to identity code list)
     |   |   |   |  Rx OK  (Card  communication and acknowledgement OK)
     |   |   |   RS232  error (Host serial communication error)
     |   |   MF type (0 = MF 1k byte card, 1 = MF 4k byte card)
     |   UL type (0 = MF standard 1k/4k card, SINGLE UID), 1 = MF Ultralight card, DOUBLE UID)
     MFRC error (Internal or antenna fault)
```

Note that bit 7 is fixed so that using a Mifare 1k card, the RWD acknowledge response to a valid host command would generally be 86 (Hex), indicating that a matched (or authorised) MF 1k card is present. The MF Ultralight card has a different memory structure to the standard 1k/4k MF cards so bits 4 and 5 have to be checked to determine which card type is present. Note also that only the relevant flags are set after each command as indicated in the following specification.

Store Keys

The Micro RWD has additional internal storage for 32 Security KEYs. Six byte Key codes are required to access individual card sectors for any Read or Write operations. This command sequence allows 6 byte Key codes to be stored at any one of the 32 key code locations. Factory defaults are Infineon/Philips specified transport key pairs (Hex FF FF FF FF FF FF / Hex FF FF FF FF FF FF) and (Hex A0 A1 A2 A3 A4 A5 / Hex B0 B1 B2 B3 B4 B5) and these are stored in the RWD non-volatile memory during manufacture. Note that due to the fundamental nature of these Key codes, incorrect values may render the system inoperable. Only one or two Security key codes are required to unlock a card sector so the provision of 32 storage locations allows for many possible applications and card uses.
IT IS STRONGLY ADVISED THAT THE KEY CODES IN THE RWD AND STORED ON THE MIFARE CARD ARE NOT CHANGED UNTIL THE OPERATION OF THE MIFARE CARD SECURITY IS FULLY UNDERSTOOD.

	B7						B0	
Command:	0	1	0	0	1	0	1	1	(Ascii “K”, 0x4B)
Argument1:	x	x	x	K	K	K	K	K	(K = Key code number, 0 - 31)
Argument2:	D	D	D	D	D	D	D	D	(D = data to write to EEPROM, LS byte)
Argument3:	D	D	D	D	D	D	D	D	
Argument4:	D	D	D	D	D	D	D	D	
Argument5:	D	D	D	D	D	D	D	D	
Argument6:	D	D	D	D	D	D	D	D	
Argument7:	D	D	D	D	D	D	D	D	(D = data to write to EEPROM, MS byte)
Acknowledge:	1	X	X	X	F	X	X	F	(F = Status flags)

Internal Key Storage memory map (default settings)

Location 0 (0x00): Key code 0 (Default 0xFF FF FF FF FF FF)
Location 1 (0x01): Key code 1 (Default 0xFF FF FF FF FF FF)

Location 2 (0x02): Key code 2 (Default 0xA0 A1 A2 A3 A4 A5)
Location 3 (0x03): Key code 3 (Default 0xB0 B1 B2 B3 B4 B5)
-
-
-
Location 28 (0x1C): Key code 28 (Default 0xFF FF FF FF FF FF)
Location 29 (0x1D): Key code 29 (Default 0xFF FF FF FF FF FF)

Location 30 (0x1E): Key code 30 (Default 0xA0 A1 A2 A3 A4 A5)
Location 31 (0x1F): Key code 31 (Default 0xB0 B1 B2 B3 B4 B5)

Note that Mifare cards manufactured by Infineon have default transport key codes of (0xFF FF FF FF FF FF) and Philips cards have (0xA0 A1 A2 A3 A4 A5 / 0xB0 B1 B2 B3 B4 B5) default transport keys. The MicroRWD MF has both pairs stored as factory settings to allow ease of use when the system is first used. (More information on the Mifare card memory maps, Security Keys and the KeyA and KeyB types can be found at the end of this document).

Write Card Block

Command to write 16 bytes of data to specified Mifare block. A Block is made up of 16 bytes and there are four blocks in each card sector (sixteen blocks per sector in upper half of Mifare 4k card). Note that blocks 3, 7, 11, 15 etc are sector trailer blocks that contain Security Key data and Access bits. Writing incorrect information to these blocks can permanently disable the sector concerned. The first argument is the block number to write data to, the second argument specifies which key code (0 - 31 from the internal storage area) to use for sector authentication/unlocking and if the Security Key is to be used as a KeyA or KeyB type code. If the write was unsuccessful (invalid card, authentication failed or card out of field) then Status flags in acknowledge byte indicate error.

	B7						B0	
Command:	0	1	0	1	0	1	1	1	(Ascii "W", 0x57)
Argument1:	N	N	N	N	N	N	N	N	(N = MF Card Block Address 0 – 255)
Argument2:	T	x	x	K	K	K	K	K	(T = Key Type, 0 = KeyA, 1= KeyB) (K = Key code number, 0 – 31)
Argument3:	D	D	D	D	D	D	D	D	(D = LS Byte of data to write to card)
Argument4:	D	D	D	D	D	D	D	D	
Argument5:	D	D	D	D	D	D	D	D	
Argument6:	D	D	D	D	D	D	D	D	
↓									16 Bytes of data
Argument15:	D	D	D	D	D	D	D	D	
Argument16:	D	D	D	D	D	D	D	D	
Argument17:	D	D	D	D	D	D	D	D	
Argument18:	D	D	D	D	D	D	D	D	(D = MS Byte of data to write to card)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Note that Mifare Ultralight cards DO NOT USE Security Keys or CRYPTO Authentication and the memory is organised differently as groups of 4 bytes (Pages). Only one Page of 4 bytes can be written at a time so to maintain compatibility and a simple RWD host command set, the same command as above is used to write data to Ultralight cards. The command and arguments have the same structure but different meanings. The "Block" address is treated as a "Page Address" and the KeyType/Key number parameter is a dummy 0x00 byte. In addition the 4 bytes of data are padded out to 16 bytes with dummy 0x00 bytes.

	B7						B0	
Command:	0	1	0	1	0	1	1	1	(Ascii "W", 0x57)
Argument1:	x	x	x	x	N	N	N	N	(N = UL Card Page Address 0 – 15)
Argument2:	0	0	0	0	0	0	0	0	(Dummy byte, 0x00)
Argument3:	D	D	D	D	D	D	D	D	(D = LS Byte of data to write to UL card)
Argument4:	D	D	D	D	D	D	D	D	
Argument5:	D	D	D	D	D	D	D	D	
Argument6:	D	D	D	D	D	D	D	D	(D = MS Byte of data to write to UL card)
Argument7 – Argument18									
	0	0	0	0	0	0	0	0	12 Dummy padding bytes, 0x00
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Read Card Block

Command to read 16 bytes of data from specified Mifare block. The first argument is the block number to read data from, the second argument specifies which key code (0 - 31 from the internal storage area) to use for sector authentication/unlocking and if the Security Key is to be used as a KeyA or KeyB type code. If the read was successful, indicated by acknowledge status flags then sixteen bytes of block data follow.

	B7						B0	
Command:	0	1	0	1	0	0	1	0	(Ascii "R", 0x52)
Argument1:	N	N	N	N	N	N	N	N	(N = MF Card Block Address 0 – 255)
Argument2:	T	x	x	K	K	K	K	K	(T = Key Type, 0 = KeyA, 1= KeyB) (K = Key code number, 0 – 31)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Data only follows if Read was successful

Reply1:	D	D	D	D	D	D	D	D	(D = LS Byte of data Read from card)
Reply2:	D	D	D	D	D	D	D	D	
Reply3:	D	D	D	D	D	D	D	D	
Reply4:	D	D	D	D	D	D	D	D	
↓									16 Bytes of data
Reply13:	D	D	D	D	D	D	D	D	
Reply14:	D	D	D	D	D	D	D	D	
Reply15:	D	D	D	D	D	D	D	D	
Reply16:	D	D	D	D	D	D	D	D	(D = MS Byte of data Read from card)

Note that as mentioned for the WRITE command, Mifare Ultralight cards DO NOT USE Security Keys or Authentication and the memory is organised differently as groups of 4 bytes (Pages).
However, unlike the Write command, 16 bytes (4 pages) can be read in a single operation The same Read command as above is used except the "Block" address is treated as a "Page Address" and the KeyType/Key number parameter is a dummy 0x00 byte. For page numbers greater than 12, the card data wraps around to page 0 etc.

	B7						B0	
Command:	0	1	0	1	0	0	1	0	(Ascii "R", 0x52)
Argument1:	x	x	x	x	N	N	N	N	(N = UL Card Page Address 0 – 15)
Argument2:	0	0	0	0	0	0	0	0	(Dummy byte, 0x00)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Data only follows if Read was successful

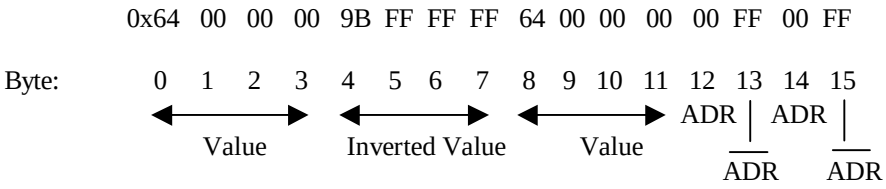
Reply1:	D	D	D	D	D	D	D	D	(D = LS Byte of data Read from UL card)
Reply2:	D	D	D	D	D	D	D	D	
Reply3:	D	D	D	D	D	D	D	D	
Reply4:	D	D	D	D	D	D	D	D	
↓									16 Bytes of data
Reply13:	D	D	D	D	D	D	D	D	
Reply14:	D	D	D	D	D	D	D	D	
Reply15:	D	D	D	D	D	D	D	D	
Reply16:	D	D	D	D	D	D	D	D	(D = MS Byte of data Read from UL card)

Inc Value (only operates on Value Data Structure)

Command to increment integer within a Value Data Structure. The command loads the value from the specified block address, adds the integer parameter and stores the result at the same or another block address. Note that the source block must have been formatted as a Value Block beforehand according to the data structure below, using the WRITE command. The INC Value command only operates on a "Value Block Structure" and will fail if the block configuration or the specified key type is incorrect.

Value Block Structure

Example format for value = 100 decimal (0x64), at block address 0.
(Value data stored LS byte first, ADR = block address, $\overline{ADR}$ = inverted block address)



The first argument is the source block address to load data from, the second argument specifies which key code and type to use for sector authentication (0-31 and if it is KeyA or KeyB type). The third argument specifies the destination block address where the incremented data is stored. Note that source and destination blocks must be within same authenticated sector. The four byte positive integer to add follows (least significant byte first).

	B7							B0	
Command:	0	1	0	0	1	0	0	1	(Ascii “I”, 0x49)
Argument1:	N	N	N	N	N	N	N	N	(N = MF source block address 0 – 255)
Argument2:	T	x	x	K	K	K	K	K	(T = Key Type, 0 = KeyA, 1= KeyB) (K = Key code number, 0 – 31)
Argument3:	N	N	N	N	N	N	N	N	(N = MF destination block address 0 – 255)
Argument4:	D	D	D	D	D	D	D	D	(D = LS byte of integer to add)
Argument5:	D	D	D	D	D	D	D	D	4 byte integer
Argument6:	D	D	D	D	D	D	D	D	
Argument7:	D	D	D	D	D	D	D	D	(D = MS byte of integer to add)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Dec Value (only operates on Value Data Structure)

Command to decrement integer within a Value Data Structure. The DEC Value command operates as the INC command except the integer parameter is subtracted from the loaded value. The first argument is the source block address to load data from, the second argument specifies which key code and type to use for sector authentication (0-31 and if it is KeyA or KeyB type). The third argument specifies the destination block address where the decremented data is stored. Note that source and destination blocks must be within same authenticated sector. The four byte positive integer to subtract follows (least significant byte first).

	B7							B0	
Command:	0	1	0	0	0	1	0	0	(Ascii “D”, 0x44)
Argument1:	N	N	N	N	N	N	N	N	(N = MF source block address 0 – 255)
Argument2:	T	x	x	K	K	K	K	K	(T = Key Type, 0 = KeyA, 1= KeyB) (K = Key code number, 0 – 31)
Argument3:	N	N	N	N	N	N	N	N	(N = MF destination block address 0 – 255)
Argument4:	D	D	D	D	D	D	D	D	(D = LS byte of integer to subtract)
Argument5:	D	D	D	D	D	D	D	D	4 byte integer
Argument6:	D	D	D	D	D	D	D	D	
Argument7:	D	D	D	D	D	D	D	D	(D = MS byte of integer to subtract)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Transfer Value (only operates on Value Data Structure)

Command to transfer (copy) Value Data Structure. The command loads the value from the specified block address and then stores the result at the same or another block address. As with INC and DEC commands the source block must have been formatted as a Value Block beforehand and the block addresses must be within same authenticated sector.

The first argument is the source block address to load data from, the second argument specifies which key code to use for sector authentication (0-31) and if it is a KeyA or KeyB code. The third argument specifies where the data is stored.

	B7							B0	
Command:	0	1	0	1	0	1	0	0	(Ascii “T”, 0x54)
Argument1:	N	N	N	N	N	N	N	N	(N = MF source block address 0 – 255)
Argument2:	T	x	x	K	K	K	K	K	(T = Key Type, 0 = KeyA, 1= KeyB) (K = Key code number, 0 – 31)
Argument3:	N	N	N	N	N	N	N	N	(N = MF destination block address 0 – 255)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

If the Inc, Dec or Transfer function was unsuccessful (invalid card, card out of field, authentication failed or data structures are incorrect) then Status flags in acknowledge byte indicate error. Note that the value manipulation commands operate internally on the Mifare card and no data is transferred back to the MicroRWD. Note also that Ultralight cards do not support Value Data Structures or the Inc, Dec, Transfer commands.

Card UID

Command to return card status and UID (Unique Identifier or Serial number).
The acknowledge byte flags indicate general Mifare card status.

	B7							B0	
Command:	0	1	0	1	0	1	0	1	(Ascii "U", 0x55)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Data only follows if card was selected OK with no errors detected.

Reply1:	D D D D D D D D	(D = LS Byte of UID/Serial number from card)
Reply2:	D D D D D D D D	
Reply3:	D D D D D D D D	
Reply4:	D D D D D D D D	
Reply5:	D D D D D D D D	Dummy bytes (0x00) for Mifare 1k/4k card types
Reply6:	D D D D D D D D	
Reply7:	D D D D D D D D	

Note that Mifare 1k and 4k cards have a four-byte serial number but Mifare Ultralight cards have a seven byte serial number. To accommodate all card types, the Card UID command returns a seven-byte field with the last three bytes padded out with 0x00 dummy bytes in the case of Mifare 1k/4k cards.

Type Identification

Command to return the **ATQA** (Answer to Request, Type A) two-byte codes and the **SAK** (Select Acknowledge) single-byte code after the complete UID has been acquired. As part of the initial communication with the Mifare card (as defined by ISO 14443A specification), the Mifare transponder responds to REQA (Request Command, Type A) with ATQA. The two-byte ATQA contains information that allows particular transponder types to be indentified. Following on from this the Mifare transponder responds to the SELECT (Select Command, Type A) with SAK (Select Acknowledge, Type A). The SAK code is a single byte value that contains further information about the type of transponder and the length of the UID. The SAK value reported is the final value after all "cascade levels" and the complete UID has been acquired.

NOTE THAT ALL THE COMMUNICATION PROTOCOL IS HANDLED INTERNALLY AND THIS COMMAND IS INCLUDED FOR DIAGNOSTIC PURPOSES TO ALLOW THE USER TO DETERMINE THE EXACT TYPE OF MIFARE CARD PRESENT IN THE FIELD, IF REQUIRED.

	B7							B0	
Command:	0	1	1	1	1	0	0	0	(Ascii "x", 0x78)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Data only follows if card was selected OK with no errors detected.

Reply1:	D D D D D D D D	ATQA - MSB
Reply2:	D D D D D D D D	ATQA - LSB
Reply3:	D D D D D D D D	SAK

	MF UL	MF 1K	MF 4K	MF DESFire	MF Prox	MF Prox	MF Prox	MF Prox	MF Prox	MF Prox
ATQA - MSB	0x00	0x00	0x00	0x03	0xXX	0xXX	0xXX	0xXX	0xXX	0xXX
ATQA - LSB	0x44	0x04	0x02	0x44	0x08	0x04	0x02	0x48	0x44	0x42

	Smart MX	Smart MX	Smart MX	Smart MX	Smart MX	Smart MX
ATQA - MSB	0xXX	0xXX	0xXX	0xXX	0xXX	0xXX
ATQA - LSB	0x08	0x04	0x02	0x48	0x44	0x42

	MF UL	MF 1K	INFINEON 1K	MF 4K	MF DESFire	MF ProX	MF ProX	MF ProX	MF ProX	MF ProX	MF ProX
SAK	0x00	0x08	0x88	0x18	0x20	0x20	0x08	0x28	0x00	0x20	0x08

	MF ProX	MF ProX	MF ProX	Smart MX	Smart MX	Smart MX	Smart MX	Smart MX	Smart MX
SAK	0x28	0x18	0x38	0x00	0x20	0x08	0x28	0x18	0x38

Note that many of the "extended" Mifare types are complex dual interface cards with embedded 8051 microcontrollers running operating systems. Depending on the card type the memory map and protocol of the "extended" Mifare card may be different to the standard card types. In these cases the MicroRWD MF will report the UID using the Card UID command but read/write operation WILL NOT work.

Command Protocol (ICODE SLI Mode)

The following commands are supported in ICODE mode. The command code followed by optional data/arguments is sent to the MicroRWD. The RWD replies with an acknowledge code (which is made up of various status flags) followed by optional data. After the command (+ data) has been sent, the acknowledge code should be read back by the host and decoded to confirm that the command was received and actioned correctly. The serial bit protocol is 9600 baud (or 38400 baud if SW1 pulled low), 8 bits, 1 stop, no parity (lsb transmitted first).

The status flags returned in the Acknowledge byte are as follows:

```
 b7  b6  b5  b4  b3 b2 b1 b0
 1   1   0   0   1  1  1  1
     |           |  |  |   EEPROM error (Internal EEPROM write error)
     |           |  |  Card OK (Label serial number matched to identity code list)
     |           |  Rx OK  (Label communication and acknowledgement OK)
     |           RS232  error (Host serial communication error)
     MFRC error (Internal or antenna fault)
```

Note that bit 7 is fixed so that the RWD acknowledge response to a valid host command would generally be 86 (Hex), indicating that a matched (or authorised) ICODE tag is present.

Write Label Block

Command to write 4 bytes of data to specified ICODE transponder block. The first argument is the block number to write data to (0 - 27), the next eight arguments specify the UID (Unique Identifier or serial number) of the tag to select (sent least significant byte first).
If the write was unsuccessful (invalid card, authentication failed or tag out of field) then Status flags in acknowledge byte indicate error.

	B7							B0	
Command:	0	1	0	1	0	1	1	1	(Ascii "W", 0x57)
Argument1:	x	x	x	N	N	N	N	N	(N = ICODE block address, 0 – 27)
Argument2:	U	U	U	U	U	U	U	U	(LSB, UID0)
Argument3:	U	U	U	U	U	U	U	U	(UID1)
Argument4:	U	U	U	U	U	U	U	U	(UID2)
Argument5:	U	U	U	U	U	U	U	U	(UID3)
Argument6:	U	U	U	U	U	U	U	U	(UID4)
Argument7:	U	U	U	U	U	U	U	U	(UID5)
Argument8:	U	U	U	U	U	U	U	U	(UID6)
Argument9:	U	U	U	U	U	U	U	U	(MSB, UID7)
Argument10:	D	D	D	D	D	D	D	D	(D = LS Byte of data to write to tag)
Argument11:	D	D	D	D	D	D	D	D	
Argument12:	D	D	D	D	D	D	D	D	
Argument13:	D	D	D	D	D	D	D	D	(D = MS Byte of data to write to tag)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Read Label Block

Command to read 4 bytes of data from specified ICODE transponder block. The first argument is the block number to read data from (0 - 27), the next eight arguments specify the UID (Unique Identifier or serial number) of the tag to select (sent least significant byte first).
If the write was unsuccessful (invalid card, authentication failed or tag out of field) then Status flags in acknowledge byte indicate error.

	B7							B0	
Command:	0	1	0	1	0	0	1	0	(Ascii "R", 0x52)
Argument1:	x	x	x	N	N	N	N	N	(N = ICODE block address, 0 – 27)
Argument2:	U	U	U	U	U	U	U	U	(LSB, UID0)
Argument3:	U	U	U	U	U	U	U	U	(UID1)
Argument4:	U	U	U	U	U	U	U	U	(UID2)
Argument5:	U	U	U	U	U	U	U	U	(UID3)
Argument6:	U	U	U	U	U	U	U	U	(UID4)
Argument7:	U	U	U	U	U	U	U	U	(UID5)
Argument8:	U	U	U	U	U	U	U	U	(UID6)
Argument9:	U	U	U	U	U	U	U	U	(MSB, UID7)
Acknowledge:	1	F	F	F	F	F	F	X	(F = Status flags)

Data only follows if Read was successful

Reply1:	D D D D D D D D	(D = LS Byte of data Read from ICODE tag)
Reply2:	D D D D D D D D	
Reply3:	D D D D D D D D	
Reply4:	D D D D D D D D	(D = MS Byte of data Read from ICODE tag)

Label UID

Command to return label status and UID (Unique Identifier or "serial number") of single (dominant) label. The acknowledge byte flags indicate general Tag status. NOTE that if multiple labels are expected in the RF field then LABEL INVENTORY command must be used to acquire full UID list.

	B7 B0	
Command:	0 1 0 1 0 1 0 1	(Ascii “U”, 0x55)
Acknowledge:	1 F F F F F F X	(F = Status flags)

Data only follows if label responded OK and UID is available.

Reply1:	U U U U U U U U	(LSB, UID0)
Reply2:	U U U U U U U U	(UID1)
Reply3:	U U U U U U U U	(UID2)
Reply4:	U U U U U U U U	(UID3)
Reply5:	U U U U U U U U	(UID4)
Reply6:	U U U U U U U U	(UID5)
Reply7:	U U U U U U U U	(UID6)
Reply8:	U U U U U U U U	(MSB, UID7)

Label INVENTORY

Command to return label status and list of UIDs (Unique Identifiers or "serial numbers"). Inventory command returns the number of tags present in the RF field (8 max) followed by UID data. Each UID is eight bytes long and is returned least significant byte first (UID0-UID7). The UID data sequence is contiguous groups of eight bytes, so host must use "Reply1" (number of UIDs) to ensure all data is received and stored (up to 66 bytes).

	B7 B0	
Command:	0 1 0 1 0 1 0 1	(Ascii “U”, 0x55)
Acknowledge:	1 F F F F F F X	(F = Status flags)

Data only follows if at least one label responds and UID is available.

Reply1:	x x x x N N N N	(N = number of UIDs following, maximum 8)
Reply2:	U U U U U U U U	(LSB, UID0)
Reply3:	U U U U U U U U	(UID1)
Reply4:	U U U U U U U U	(UID2)
Reply5:	U U U U U U U U	(UID3)
Reply6:	U U U U U U U U	(UID4)
Reply7:	U U U U U U U U	(UID5)
Reply8:	U U U U U U U U	(UID6)
Reply9:	U U U U U U U U	(MSB, UID7)

Reply58:	U U U U U U U U	(LSB, UID0)
Reply59:	U U U U U U U U	(UID1)
Reply60:	U U U U U U U U	(UID2)
Reply61:	U U U U U U U U	(UID3)
Reply62:	U U U U U U U U	(UID4)
Reply63:	U U U U U U U U	(UID5)
Reply64:	U U U U U U U U	(UID6)
Reply65:	U U U U U U U U	(MSB, UID7)

Notes for Commands (Mifare and ICODE)

NOTE also that for the “Read Card Block” or “Card UID” command (or INVENTORY command), if an error flag has been set in the Acknowledge code then there will be NO following data.

NOTE that the serial communication uses hardware handshaking to inhibit the host from sending the Micro RWD commands while Card interrogation is in progress. The serial communication system and protocol allows for a 10ms ‘window’ every Card polling cycle indicated by the BUSY(CTS) line being low. During this ‘window’ the host must assert the first start bit and start transmitting data. The BUSY goes high again 10ms after the last stop bit is received. The host must therefore send the command and all the arguments with no gaps otherwise timeout will occur and BUSYgoes high. NOTE that only one command sequence is handled at a time.

Method of Operation

The system works on a polling principle whereby the RF field is turned on for a short period to check if a card is present. Authentication and Read/Write operations can then be performed before the RF field is turned off again and the process repeats. The polling principle where the RF is off most of the time allows for low average power consumption.

When a Mifare card is detected in the field a multi-pass handshaking procedure takes place where card information and serial number data is exchanged and checked for integrity. Once this procedure has completed successfully an individual card has been selected and is available for other operations.

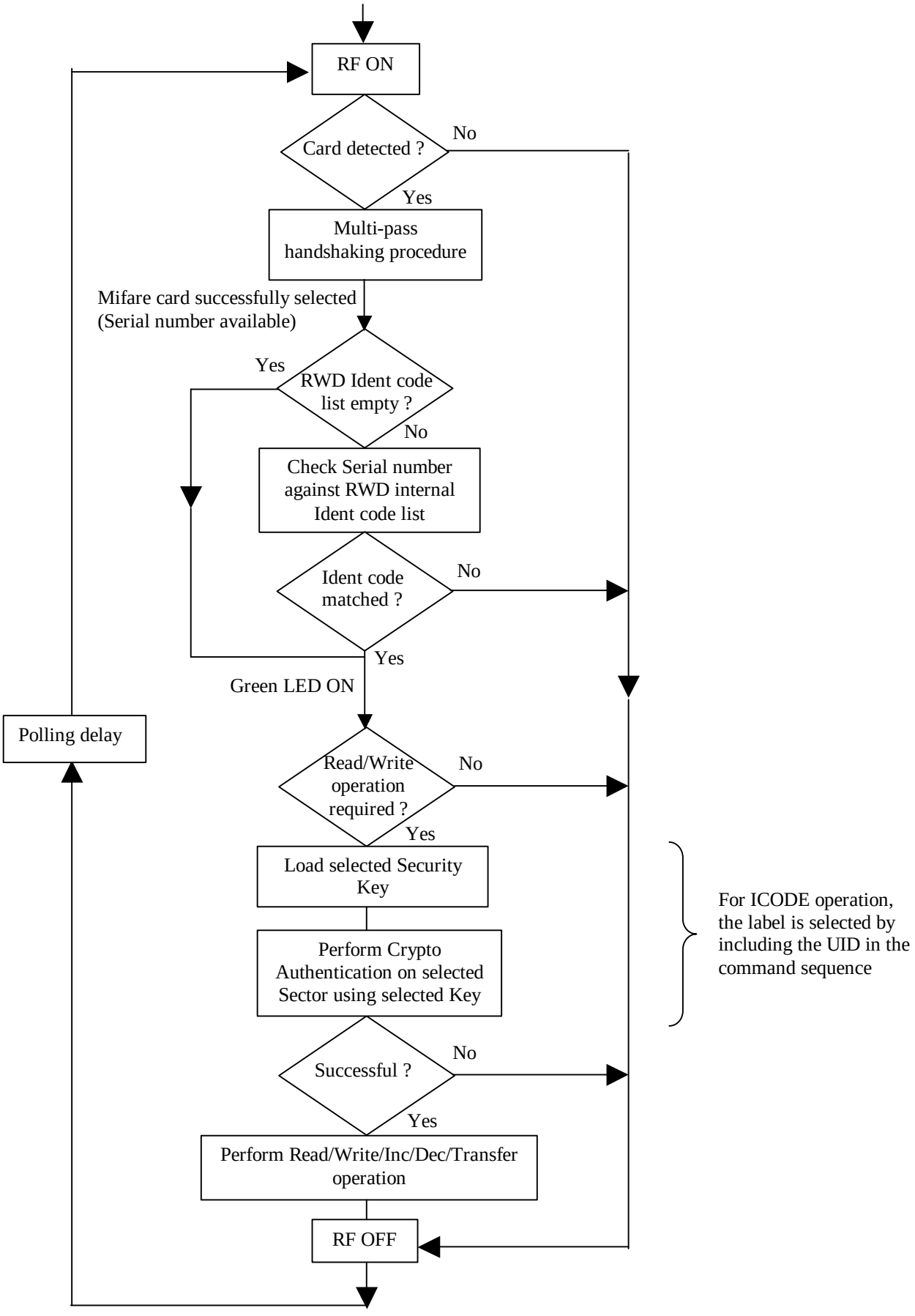
The RWD itself has the additional feature of then checking the four byte Mifare UID/serial number (or least significant four bytes of Ultralight UID) against an internal authorisation list. The RWD internal EEPROM contains a list of four byte Identity codes (up to 60 of them) located from byte 12 onwards. If the list has FF FF FF FF (hex) stored at the first location (EEPROM bytes 12 - 15) then the list is treated as empty so the Identity code check is skipped.
Otherwise the card serial number is checked against all the entries in the list (until the FF FF FF FF termination code is reached) and if matched then the RWD allows the card to be accessed for other operations. If not the Red LED remains on and the card is blocked for further access. This is an additional level of security that can be used as a "mini access control" system for simple applications that only involve the serial number or where the Security Keys are not known.

Once the RWD has selected the card and has matched the serial number against it's internal list (or the list is empty) then the Read/Write (or Inc/Dec/Transfer) operations can be performed. These require an internal high-security Authentication Crypto algorithm to take place that use the supplied Security Keys to gain access to a particular sector. If the Key selected does not match the Key stored in the Mifare card sector then the operation fails and the Red LED is turned on again

So in summary, a card can be successfully selected but can be blocked by the RWD authorisation list and fail Read/Write operations because the Keys are incorrect. Even if the Security Key is incorrect the Serial number can still be read using the "Card UID" command.

For ICODE operation, when a label is in the RF field a handshaking procedure occurs and the RWD acquires the labels UID (Unique Identifier/serial number). The least significant four bytes of the UID (UID0 - UID3) are used to check against the internal authorisation list. For subsequent read/write operations, the labels are individually selected by including their UID in the command sequence. The principle of the RF field being ON only for label communication means that the label is effectively turned off and reselected each polling cycle. The assumption is therefore made that after the label UID has been acquired (using Label UID command), the label is still present for the next polling cycle.

The exception to this principle is the INVENTORY command where the RF field is left ON while up to 8 label UIDs are determined. Thereafter the RF field is turned OFF and the polling cycle procedure continues and once again the assumption is made that the labels are still present.


RF ON
Card detected ?
No
Yes
Multi-pass handshaking procedure
Mifare card successfully selected (Serial number available)
Yes
RWD Ident code list empty ?
No
Check Serial number against RWD internal Ident code list
Ident code matched ?
No
Yes
Green LED ON
Polling delay
Read/Write operation required ?
No
Yes
Load selected Security Key
Perform Crypto Authentication on selected Sector using selected Key
For ICODE operation, the label is selected by including the UID in the command sequence
Successful ?
No
Yes
Perform Read/Write/Inc/Dec/Transfer operation
RF OFF

Basic RWD Communication

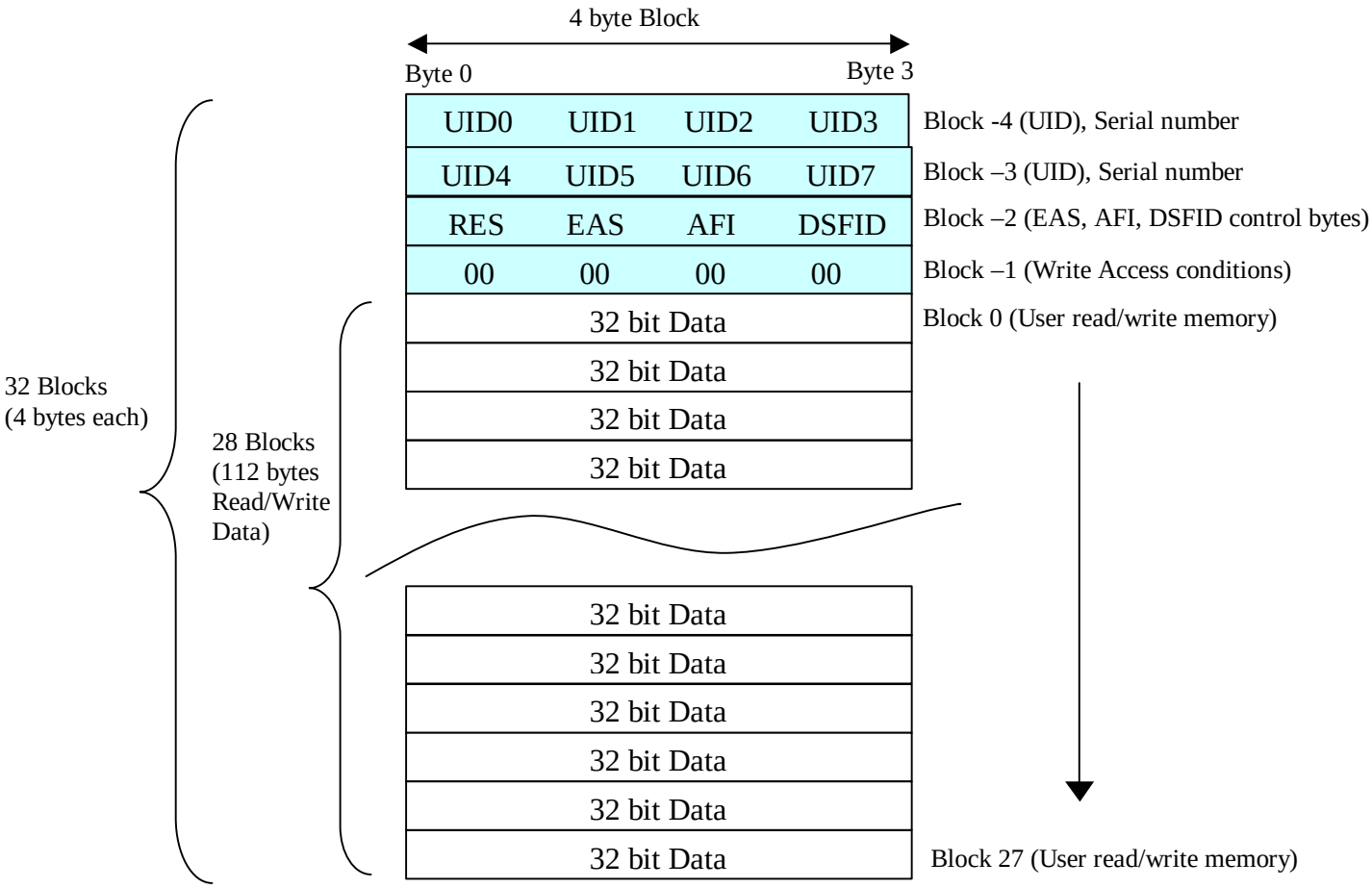
For basic operation of the MicroRWD connected to a host computer, the RWD can either be polled or communication can be triggered by an interrupt signal (Green LED output). In either case for Mifare operation, the host would generally send the “STORE KEYS” command to load a custom security key into the RWD memory or simply use the pre-loaded default key values.

For a polling technique, the host computer would then keep sending the “STATUS” or “CARD UID” command and would monitor the acknowledgement code until a valid Mifare card was detected.

For the interrupt technique, the Green LED output can be used as an interrupt signal connected to the host computer. The Green LED output is normally high and goes low only when a valid card has been detected. This falling-edge signal can trigger a host interrupt to then send the “STATUS” or “CARD UID” command to determine the card type and serial number.

In both cases once a valid card has been detected a “READ BLOCK” or “WRITE BLOCK” command can be sent and the acknowledge code monitored to establish that the operation was successful.

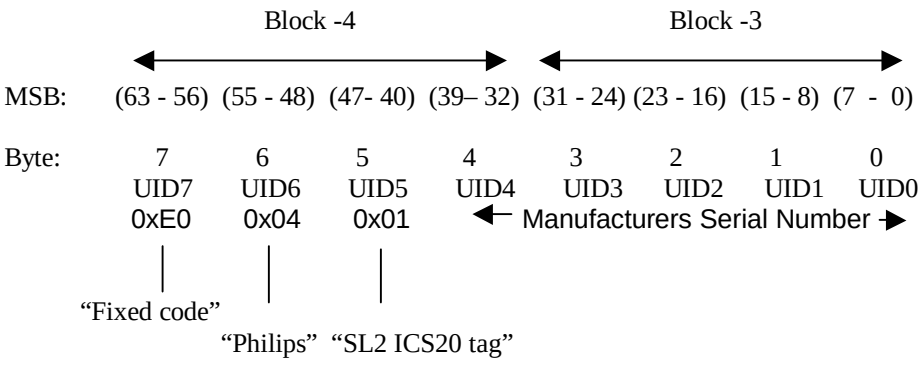
ICODE SLI (128 byte) Memory Map



Note: ICODE SLI cards/labels have 128 bytes of memory of which 16 bytes (first four blocks) are only accessible using special commands. The eight byte UID (Serial number) is acquired by the RWD when the card/label is first detected and is read using the Card UID command.

ICODE SLI UID (Unique Identifier/Serial number)

Note that the numbering convention for displaying bits according to ISO15693 is to start with LSB and end with MSB rather than usual convention of bit numbering within a byte.



ISO15693 defines that UID0 - UID5 (bits 0-47) contain the manufacturer serial number. (UID4 and UID5 can contain information on capacitor value, IC and customer identification).

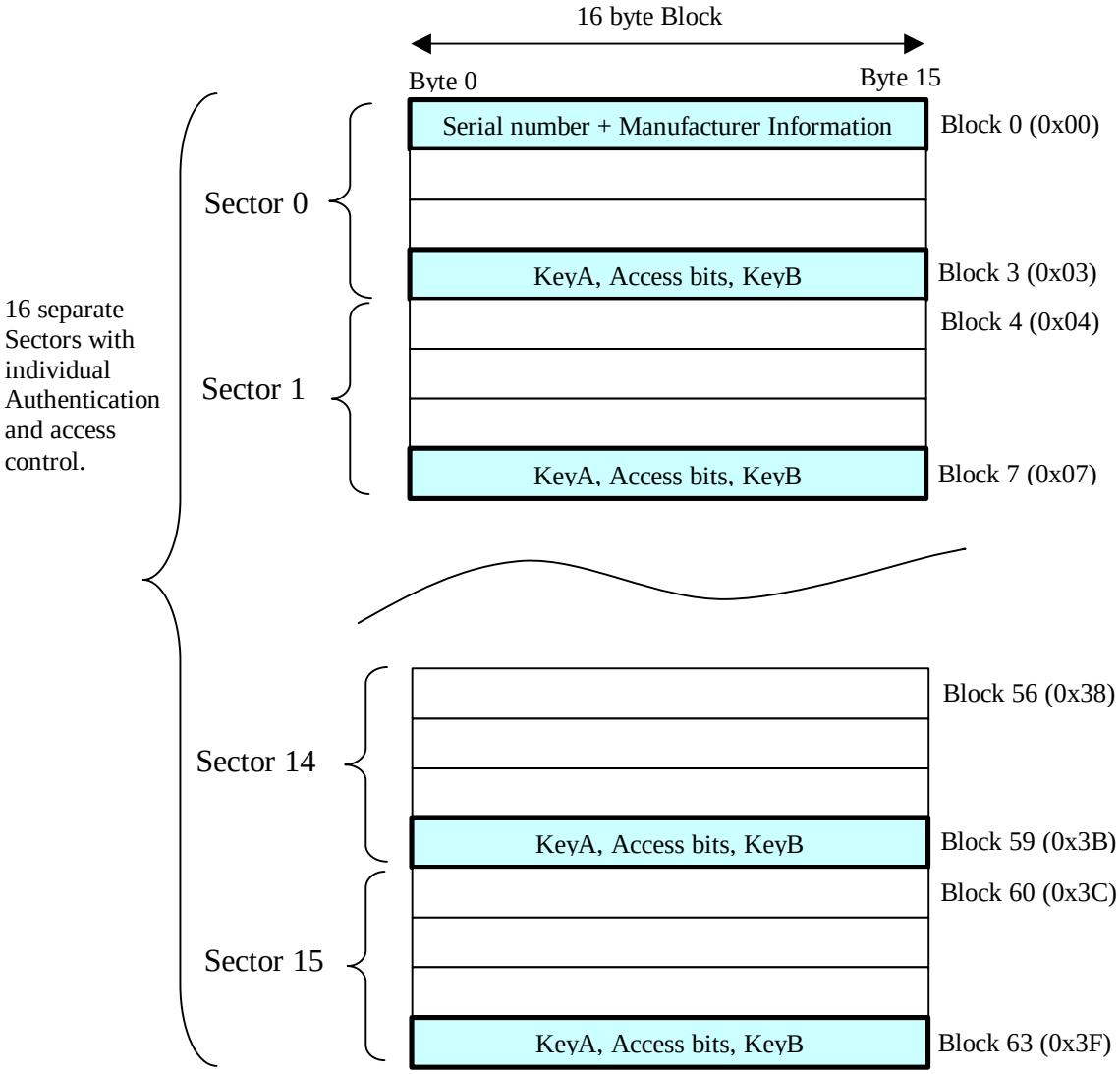
UID6 = IC manufacturers code

ST = 0x02
Philips = 0x04
Infineon = 0x05
TI = 0x07
EM Marin = 0x16

UID7 = Fixed code = 0xE0

(The ICODE ident code is defined as the least significant four bytes of the 8 byte UID, effectively UID0 - UID3). Note that The UID used for the "Ident list" check is the first tag UID acquired by internally generated Inventory command.

Mifare 1k (1024 byte) Memory Map



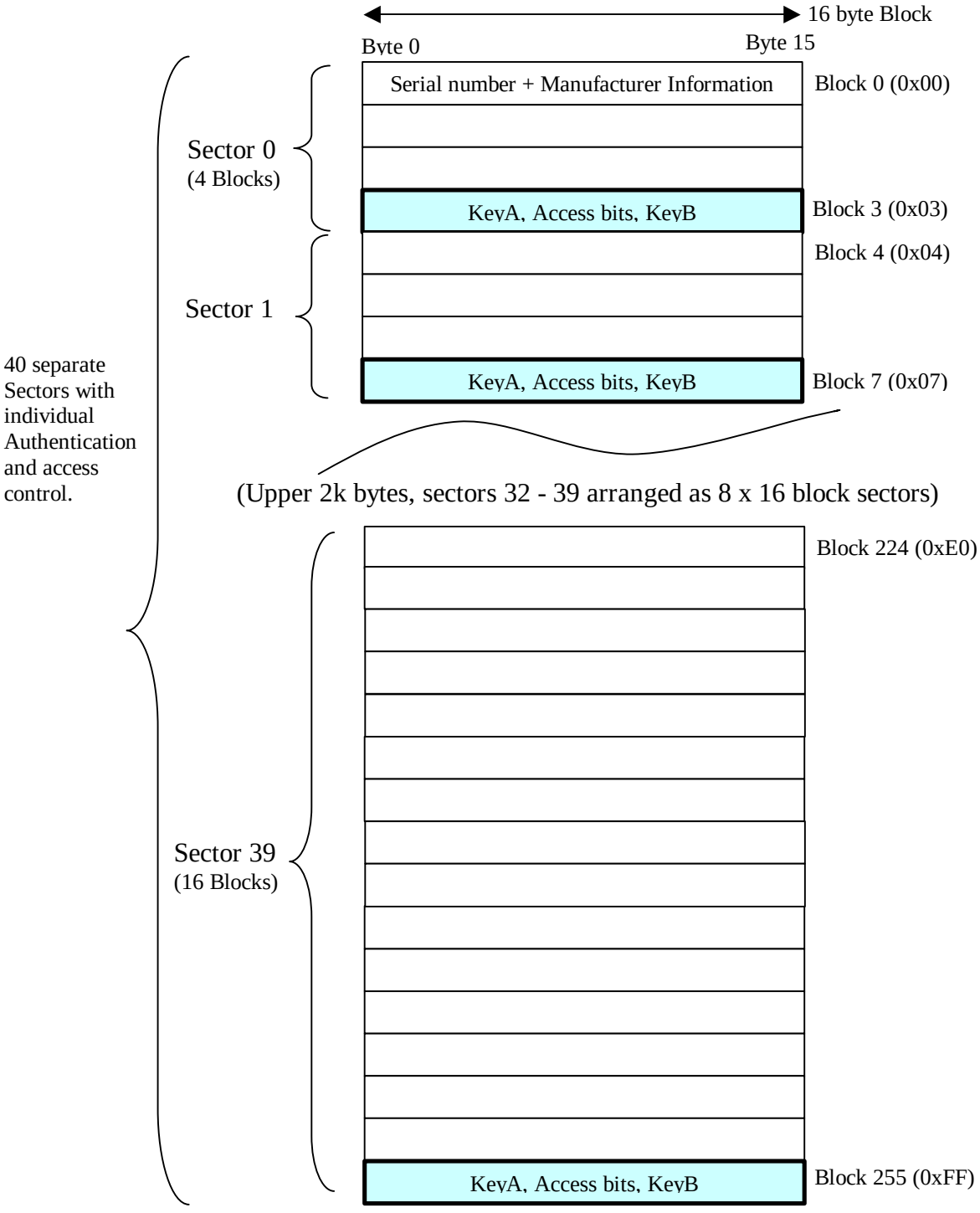
1024 byte memory is organised as sixteen sectors, each of which is made up of four blocks and each block is 16 bytes long. The first block in the memory (Block 0) is read-only and is set in the factory to contain the four-byte serial number (UID), check bytes and manufacturers data.

The last block of each sector (Blocks 3, 7, 11, 15……59, 63) is the Sector Trailer Block which contains the two security Key codes (KeyA and KeyB) and the Access bits that define how the sector can be accessed

Taking into account the Serial Number/Manufacturers Block and the Sector Trailer Blocks then there are 752 bytes of free memory for user storage. For all Read and Write operations the Mifare card memory is addressed by Block number (in hexadecimal format).

Mifare 4k (4096 byte) Memory Map

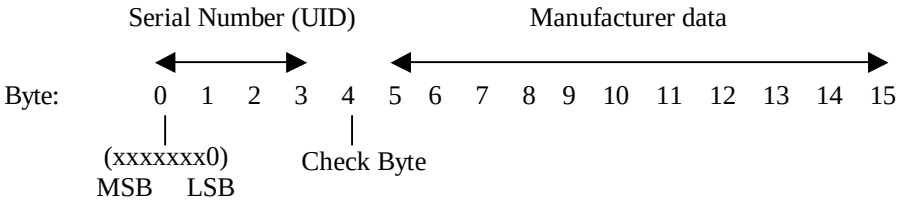
(Lower 2k bytes, sectors 0-31 arranged as 32 x 4 block sectors)



The lower 2048 bytes of the 4k card (sectors 0 – 31) are organised in the same way as the 1k card. However the upper 2048 bytes are organised as eight large sectors of 16 blocks each (sectors 32 – 39). Taking into account the Serial Number/Manufacturers Block and the Sector Trailer Blocks then there are 3440 bytes of free memory for user storage.

Manufacturer Block

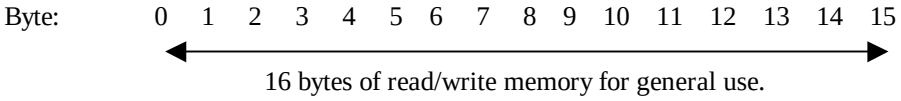
The Manufacturer block is the first data block in Sector 0 and it contains the read-only serial number (UID – Unique Identifier) and the IC manufacture information.



Data Blocks

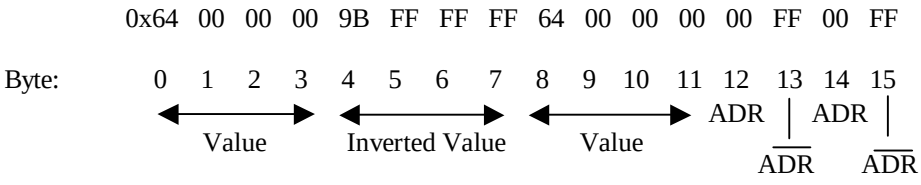
Mifare sectors contain 3 blocks (1k card) or 15 blocks (upper half of 4k card) of 16 data bytes (except Sector 0 that has Manufacturer Block and 2 data blocks). Data blocks can be configured as standard read/write memory or "Value Blocks" for special electronic-purse operations. "Value Blocks" can use additional commands such as Increment and Decrement for direct control of the data field, they have a fixed data format which permits error detection/correction and backup management. "VALUE" is a signed four byte integer (2's complement format) and is stored three times, twice non-inverted and once inverted. "ADR" signifies a one byte address that can be used to store the block number. "ADR" is stored four times, twice inverted and twice non-inverted.

Data Block



Value Block

Example format for value = 100 decimal (0x64), at block address 0.
(Value data stored LS byte first, ADR = block address, $\overline{\text{ADR}}$ = inverted block address)



Note that the ADR and inverted $\overline{\text{ADR}}$ block address is part of the required format for the Value Data Structure BUT it is not changed by the Inc, Dec or Transfer commands and exists to allow optional storage of block numbers. For general use, the 0x00 address and 0xFF inverted address can be used to satisfy the structure format. The Inc, Dec and Transfer commands first check the structure format before beginning the operations and the commands fail if the format is not correct.

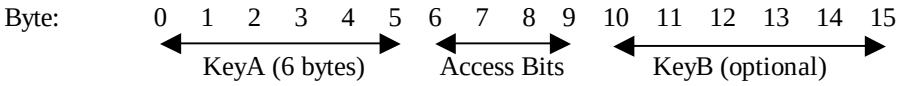
Value Data Structures must first be formatted as above using a WRITE Block command before INC, DEC or TRANSFER commands can be used.

Sector Trailer Block

The last block of each sector (Blocks 3, 7, 11, 15……59, 63 etc) is the Sector Trailer Block which contains the two security Key codes (KeyA and KeyB) and the Access bits that define how the data blocks can be accessed (Read/Write, Read or Write only, as data or Value blocks and using which key). If KeyB is not used then the last 6 bytes of the Sector Trailer Block can be used for general data storage. Byte 9 (last byte of Access bits) is not used and can also be used for general storage. Note that the KeyA (and KeyB) value read back as logical 0's to ensure system security.

IT IS STRONGLY RECOMMENDED THAT THE KEY CODES AND THE ACCESS BITS STORED ON THE MIFARE CARD ARE NOT CHANGED UNTIL THEIR OPERATION IS FULLY UNDERSTOOD.

Sector Trailer Block



KeyA and KeyB

The security Key codes are each six bytes long and for successful authentication and read/write communication the RWD device would have to have one or both keys stored in its internal memory as well (depending on Access Bit settings). To allow "out-of-the-box" operation with new Mifare transponders and RWD devices, Transport Key values are pre-loaded into the transponder memory and also into the RWD memory.

Transport Key values as defined by Infineon are:
KeyA: 0xFF FF FF FF FF FF **KeyB:** 0xFF FF FF FF FF FF

Transport Key values as defined by Philips Semiconductors are:
KeyA: 0xA0 A1 A2 A3 A4 A5 **KeyB**: 0xB0 B1 B2 B3 B4 B5

These KeyA and KeyB values are stored as repeated pairs in the MicroRWD MF device as a factory default.

If the user intends using other Mifare cards then they may have different transport keys loaded so the user would have to use the STORE KEY command to load the particular Key codes into the RWD memory first. Once correct communication is established then Keys and Access bits can be changed on the card and RWD to suit the users final system requirements.

Access Bits

The access conditions for every data block and sector trailer are defined by three bits (C1, C2, C3), which are stored in a specific non-inverted and inverted pattern in the sector trailer of the particular sector. The access bits control the memory access rights using the secret KeyA and KeyB codes. The access bits can be changed provided the relevant Key(s) is known and the current access conditions allow this operation.

Note that with each memory access the internal logic of the Mifare transponder verifies the format of the access conditions. If it detects a format error then the entire sector is irreversibly locked. Since the access bits themselves can be blocked for read/write operations, care must be taken when cards are being personalised for a particular application or service provider.

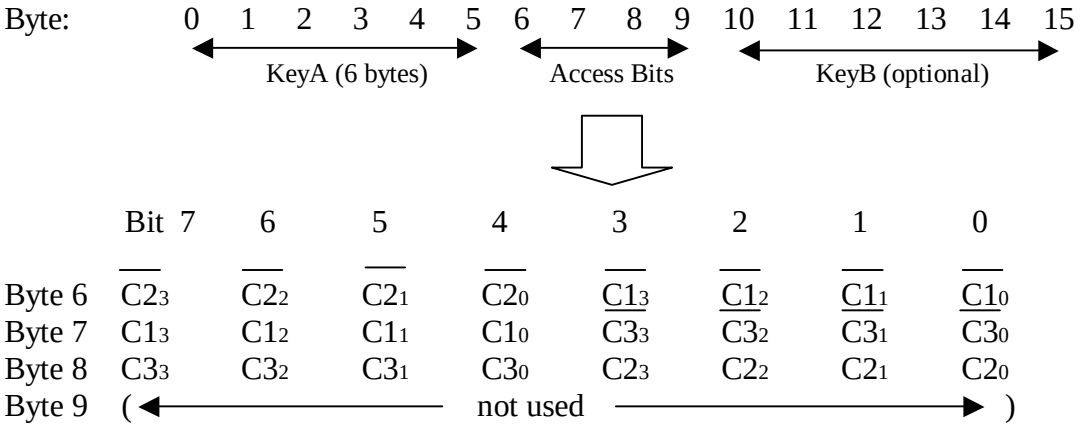
Access Bits	**Block Affected within a sector** (1k card and lower half of 4k card)		(Upper half of 4k card)	
$C1_0$ $C2_0$ $C3_0$	Block 0	(Data Block)	Blocks 0-4	(Data Blocks)
$C1_1$ $C2_1$ $C3_1$	Block 1	(Data Block)	Blocks 5-9	(Data Blocks)
$C1_2$ $C2_2$ $C3_2$	Block 2	(Data Block)	Blocks 10-14	(Data Blocks)
$C1_3$ $C2_3$ $C3_3$	Block 3	(Sector Trailer)	Block 15	(Sector Trailer)

Note that the Access Conditions for individual blocks can be set on 1k cards and for lower half of 4k card memory. However for the upper half of 4k cards the Access conditions are set for groups of five blocks (except for Sector Trailer Block which is still individually set).

Access Conditions

The C1, C2, C3 access bits are stored in a specific non-inverted and inverted pattern in the sector trailer of the particular sector. These bits control the Access Conditions for every data block and sector trailer block.

Sector Trailer Block



$\overline{Cxy}$ = Inverted bit

Transport settings (factory defaults)

Byte 6	(1	1	1	1	1	1	1	1) = 0xFF
Byte 7	0	0	0	0	0	1	1	1) = 0x07
Byte 8	1	0	0	0	0	0	0	0) = 0x80
Byte 9	(	◄	———	not used	———	►		)

Access Condition for Sector Trailer Block

The read/write access to the Keys and the Access Bits themselves is controlled by the access conditions for the Sector Trailer Block. The read/write access is specified as "Never", "KeyA", KeyB" or Key A|B (KeyA OR KeyB).

Access Bits			**Access condition for:**						
			KEY A		ACCESS BITS		KEY B		
C1	C2	C3	Read	Write	Read	Write	Read	Write	
0	0	0	never	keyA	keyA	never	keyA	keyA	(KeyB can be read)
0	0	1	never	keyA	keyA	keyA	keyA	keyA	**(Transport setting)**
0	1	0	never	never	keyA	never	keyA	never	(KeyB can be read)
0	1	1	never	keyB	keyA|B	keyB	never	keyB	
1	0	0	never	keyB	keyA|B	never	never	keyB	
1	0	1	never	never	keyA|B	keyB	never	never	
1	1	0	never	never	keyA|B	never	never	never	
1	1	1	never	never	keyA|B	never	never	never	

The new Mifare cards have the access conditions predefined as transport configuration:

C1 C2 C3 = (0 0 1) which means Sector Trailer Block can only be read or written to using KeyA and KeyA itself can never be read.

Because the Access Bits themselves can be locked great care must be taken when any of these settings are changed because they may be irreversible making the card unusable.

Access Condition for data areas

The read/write access to the data areas is also controlled by the access conditions defined in the Sector Trailer Block. The read/write access is specified as "Never", "KeyA", KeyB" or Key A|B (KeyA OR KeyB).

A data block can be a "read/write block" or a "value block". For a "read/write" block the basic read and write operations are allowed. For the "value block" the additional increment, decrement, transfer and restore operations can apply. In one case (001) only read and decrement are possible for a "non-rechargeable" card application and in another case (110) recharging is only possible using keyB.

The default transport configuration specifies that the data areas can only be accessed using KeyA|B, however the operation of the Mifare cards define that "IF KEYB CAN BE READ IN THE CORRESPONDING SECTOR TRAILER THEN IT CANNOT SERVE FOR AUTHENTICATION". This means that for the transport configuration (and 001 and 010 cases), KeyA must be used for access.

Note also that the read-only status of the Manufacturer Block is not affected by the access bits setting.

Access Bits			Access condition for:				Application
C1	C2	C3	Read	Write	Increment	Decrement, Transfer, Restore	
0	0	0	keyA|B	keyA|B	keyA|B	keyA|B	(Transport setting)
0	0	1	keyA|B	never	never	keyA|B	(value block)
0	1	0	keyA|B	never	never	never	(read/write block)
0	1	1	keyB	keyB	never	never	(read/write block)
1	0	0	keyA|B	keyB	never	never	(read/write block)
1	0	1	keyB	never	never	never	(read/write block)
1	1	0	keyA|B	keyB	keyB	keyA|B	(value block)
1	1	1	never	never	never	never	(read/write block)

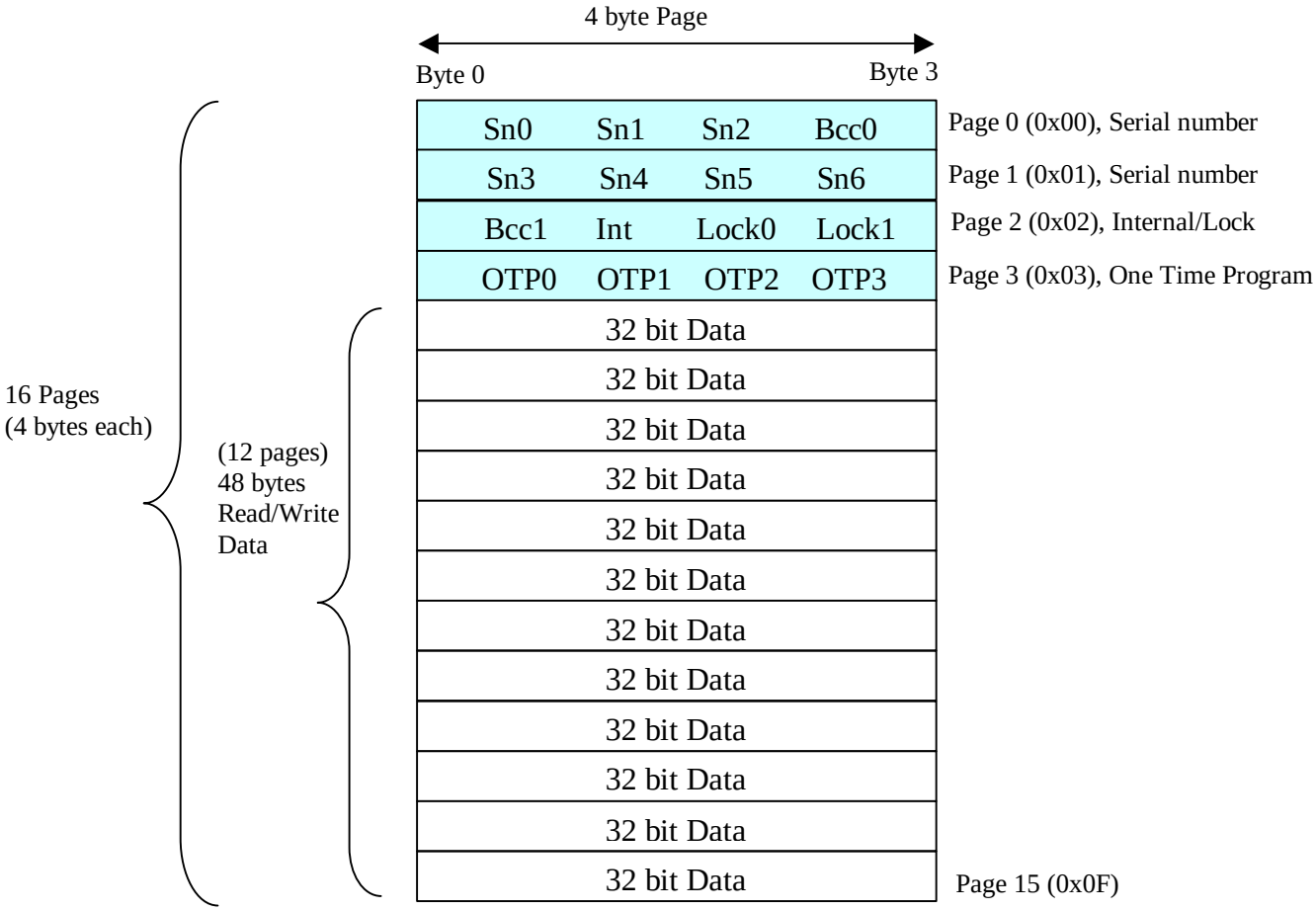
Combining the transport (default) access conditions for the data areas and the sector trailer block, the typical setting is:

Sector Trailer Block

Byte:	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	0xA0	A1	A2	A3	A4	A5	FF	07	80	69	B0	B1	B2	B3	B4	B5

The default access bits (0xFF 07 80 69) define that KeyA must used for all operations and KeyA can never be read, so in the previous example bytes 0 – 5 (keyA) would always read as zeros.

Mifare Ultralight (64 byte) Memory Map



Note: Ultralight card has 7 byte serial number (+ 2 check bytes)
Bcc0 = CT xor Sn0 xor Sn1 xor SN2 (CT = Cascade Tag = 0x88)
Bcc1 = Sn3 xor Sn4 xor Sn5 xor Sn6

The Mifare Ultralight transponder has a different structure to the standard 1k and 4k byte Mifare transponders. The 64 byte memory is organised as sixteen pages of four bytes each. The first two pages (and byte 0 of page 3) contain a seven byte serial number (UID) together with two check bytes. The other "system" bytes are used for locking card features and providing a number of OTP (One Time Programmable) data bytes. The remaining 12 pages (48 bytes) can be used for general read/write storage.

Despite having a different memory structure, the Ultralight uses the same Mifare communication protocol except that a cascaded selection procedure is used and the **Authentication procedure is not used for read/write operations**. The security Keys are therefore NOT required for any operations to an Ultralight card. Because of it's smaller memory, more basic operation and lower cost, the Ultralight card is typically used for low-value single applications.
The MicroRWD MF reader module fully supports the Ultralight cards and the read and write commands have the same format as for the standard cards except that dummy data bytes (0x00 values) are used for the Key number etc. In addition the "block number" argument becomes a "page number" parameter.

Mifare Applications and Security

The Mifare “classic” family is the pioneer and market leader in contactless smart card technology operating in the 13.56 MHz frequency range with read/write capability. The Mifare technology was originally designed for Electronic-Purse applications for public transport systems and was a benchmark for the ISO 14443A standard. The cards have up to 40 separate memory sectors or “purses” (on Mifare 4k card) that can be individually locked and unlocked for access. In addition, the high speed communication of the 13.56 MHz interface allows for quick increment/decrement operations that are required for rapid ticketing or e-purse applications. Typical transaction time is less than 100ms for read/modify/write operations. The multiple memory sectors allow for different service providers to use the same Mifare card with complete independence and security, the Ultralight card has only one sector and is designed for single use or disposable applications due to it’s low cost.

Because Mifare was designed originally for Electronic-Purse applications, a very high level of security was essential to prevent fraud. Each transaction is started with a mutual three pass authentication procedure according to the ISO 9798-2 standard. RF communication is protected from replay attack and data communication between RWD and card is encrypted according to the Philips triple-DES CRYPTO1 algorithm. Seperate sets of two security keys for each memory sector (or application) ensure that service providers have complete security control over their individual sector. The high level of data integrity for the 106 kbaud RF communication is achieved by combining a special patented modulation technique, 16 bit CRC, parity bit coding, bit counting, channel monitoring and an anticollision algorithm.

Finally, the RWD device itself stores up to 32 security keys that the service provider can change and use for their applications. These keys are non-volatile and cannot be read back further ensuring system security.

Even though the Mifare technology is ideally suited to e-purse uses, each card has a unique serial number at the beginning of it’s memory that makes the technology suitable for almost any application requiring large and secure read/write memory areas. Typical applications could include:-

Access Control
Automatic Fare Collection
Bus/Train/Airline ticketing
Electronic Purse
Vending
Loyalty/Membership Schemes
ID cards
Time and Attendance
Asset Tracking
Gambling
Electronic Keys
Logistics
Road Tolling
Payphones
Park and Ride Schemes
Pre-paid metering

IB Technology's design philosophy has created the "Universal RFID Socket" so that different technologies such as the 125 kHz MicroRWD (Hitag and EM Marin) modules, QT (Quad Tag) module and the 13.56 MHz MicroRWD Mifare and I.CODE modules are the same physical size, have the same pinout and share common serial commands. This concept allows users to select and migrate between different RFID technologies according to the specific features they require with the minimum of design changes.

More information on the Micro RWD and other products can be found at the Internet web site:

http://www.ibtechnology.co.uk

Or alternatively contact IB Technology by email at:

sales@ibtechnology.co.uk