

Data Sheet

H1PROT.PDF

14 Pages

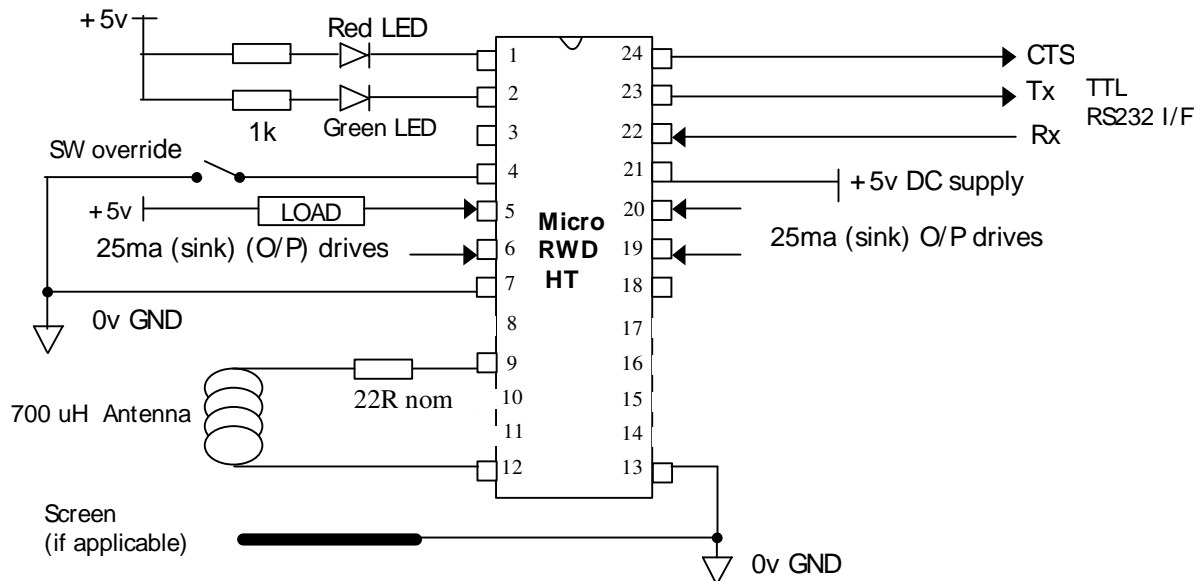
Last Revised 02/11/06

Micro RWD H1/S Protocol

The MicroRWD H1/S version is a complete reader and tag acceptance solution for Hitag 1 RF transponders (including Hitag S256 and S2048 in Plain Memory mode). The solution only needs a 700uH antenna coil connected and 5v DC supply to be a fully featured read/write system. The module provides internal EEPROM memory for holding lists of authorised identity codes, a manual override switch facility and has LED drives to give visual indication of acceptance.

The RWD also has a TTL level RS232 interface that allows a host system to communicate with the RWD if necessary, so that system features can be customised, configurations changed and tag read/write data handled by the host system.

Typical application configuration for Micro RWD module



The Hitag 1 transponders have significantly more memory than most other tags and provide 2048 bits (256 bytes) of EEPROM memory arranged as partitioned 32 bit pages. An area of 1536 bits (192 bytes) is open for general user data. (Hitag S256 / 2048 transponders in Plain Memory mode have 24/248 bytes available for user data). The communication protocol (handled entirely by the RWD) supports multiple tags in the RF field and data integrity and security is ensured using extensive CRC (Cyclic Redundancy Check) methods and highly encrypted data storage.

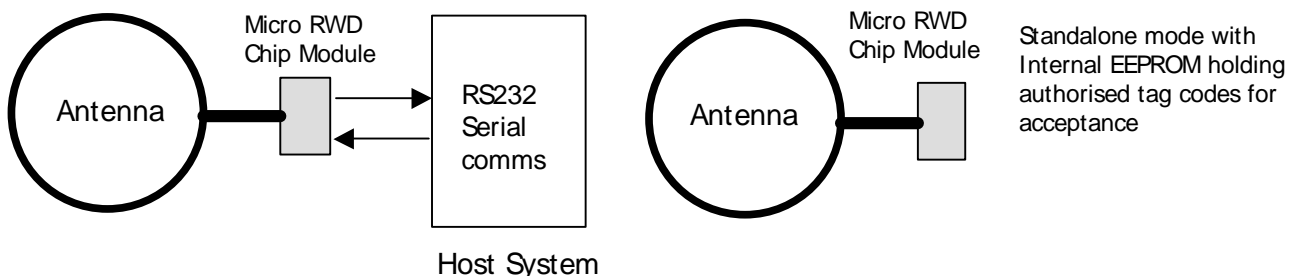
The MicroRWD is essentially a proximity system and a Read/Write range of up to 20cm can be achieved with the same level of reliable communication and EMC resilience. The unique AST (Adaptive Sampling) feature allows the RWD to continually adjust and re-tune the sampling to allow for inductive changes in the RF field, an essential feature for real-world reliability and robust operation. The communication protocol with the tags can achieve 4k bits/second of data transfer and the total time to read a 32 bit page, including reading of the serial number, selecting the tag and the read operation takes less than 100ms.

The MicroRWD can be easily integrated into almost any application; when power (5v DC) is first applied to the board the red and green LEDs flash once to indicate successful power-up. The device can also check for broken or shorted antenna and can even detect very badly tuned antennas, these problems are indicated by the red LED flashing continuously until the fault has been rectified.

The MicroRWD will normally have the red LED lit until a valid card or tag is brought into the RF field. If the tag is accepted as valid then the green LED is lit and the output drivers (OP0, OP1, OP2, OP3) are switched on. These outputs can be connected together to give up to 100ma of drive current for operating a relay etc. In addition, a switch input is provided for overriding the tag reading operation and switching the output drives directly.

(Hitag 1, Hitag S are trademarks of Philips Semiconductors NV)

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 can be empty, effectively authorising any HT1 transponder for subsequent read/write operations. A simple serial protocol allows a host system to communicate with the Micro RWD in order to program new authorised identity codes, change encryption seed code and perform read/write operations to the tag itself.
- 2) Standalone mode is where the HT1 tag 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.

Supported transponder types

The MicroRWD H1/S is designed to communicate with the following passive RF transponder types:-

- 1) Hitag 1 read/write transponders configured in R/W Public mode. Setting the HT1 to any other configuration will render them inoperable with this system. **Note: Only the HT1 ICS30 02x Hitag silicon is fully supported for WRITE/ READ operations. The earlier HT1 ICS30 01x silicon (made obsolete early 1997) is only supported for READ operations.**
- 2) Hitag S256 (32 byte) read/write transponders configured in PLAIN MEMORY mode (factory default).
- 3) Hitag S2048 (256 byte) read/write transponders configured in PLAIN MEMORY mode (factory default).

The operation of the Micro RWD and Hitag 1/S transponders is described in more detail at the end of this document.

The identification codes described in this text are regarded as the first four bytes (serial number or page 0) of the tag memory array.

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 tag communication. It is assumed that the host (such as a PC) can buffer received data.

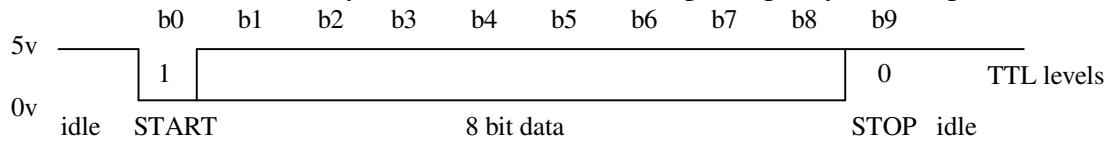
Tx, Rx and RTS signals from the Micro RWD are all TTL level and can be converted to +/- 10v RS232 levels using an inverting level converter device such as the MAX202 (note the inversion of the TTL levels).

The serial communication system and protocol allows for a 10ms 'window' every Tag 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.

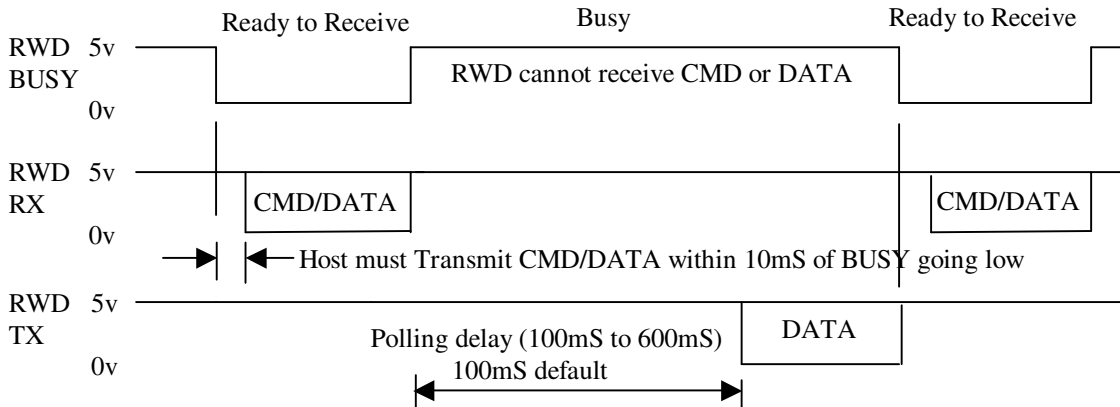
NOTE that only one command sequence is handled at a time.

ib technology

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



RWD tag polling cycle and serial communication BUSY protocol



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 PAGE 1 (0x52 0x01)

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

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

        GET_REPLY(); // Get Acknowledge byte + data
        // Response to READ command is 0xC0 (no tag) or 0xD6 + four bytes of DATA.
    }
}

```

Command Protocol

The following commands are supported. 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 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)
								Tag OK (Tag identity code matched to list)
								Rx OK (Tag communication and acknowledgement OK)
								RS232 error (Host serial communication error)
								RELAY Enabled flag
								HTRC (or Antenna fault) error flag

Note that bits 6 and 7 are fixed 1's so that an acknowledge code of D6 (Hex) would generally indicate no errors with a matched (or authorised) HT1 Tag present.

Note also that only the relevant flags are set after each command as indicated in the following specification.

Write Tag Page

Command to write 4 bytes of data to HT1 32 bit page. If the write was unsuccessful (invalid tag or out of field) then Status flags in acknowledge byte indicate error.

	B7							B0	
Command:	0	1	0	1	0	1	1	1	(0x57)
Argument1:	x	x	N	N	N	N	N	N	(N = HT1 page address 0-63)
Argument2:	D	D	D	D	D	D	D	D	(D = msb data to write to HT1)
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	(D = lsb data to write to HT1)
Acknowledge:	1	1	F	F	F	F	F	X	(F = Status flags)

Write Tag Block

Command to write up to 16 bytes of data to HT1 memory. A Block is made up of four pages (each page being 4 bytes of data). If the specified page lies on the block boundary then all 16 bytes (4 pages) can be written. If the specified page is on the block boundary + 1 then 12 bytes (3 pages) can be written.

In this way 16, 12, 8 or 4 bytes of data can be stored on the tag depending on the page number and it's position within the block. If the write was unsuccessful (invalid tag or out of field) then Status flags in acknowledge byte indicate error.

	B7		B0						
Command:	0	1	1	1	0	1	1	1	(0x77)
Argument1:	x	x	N	N	N	N	N	N	(N = HT1 page address 0-63)
Argument2:	D	D	D	D	D	D	D	D	(D = msb data to write to HT1)
Argument3:	D	D	D	D	D	D	D	D	(PAGE N DATA)
Argument4:	D	D	D	D	D	D	D	D	
Argument5:	D	D	D	D	D	D	D	D	(D = lsb data to write to HT1)
									Up to 16 bytes can be specified depending on page address N
									ie. Perform PAGE/4 => if remainder (mod) = 0 then full block (16 bytes)
									if remainder = 1 then 12 bytes sent
									if remainder = 2 then 8 bytes sent
									if remainder = 3 then 4 bytes sent
V									
Argument14:	D	D	D	D	D	D	D	D	(D = msb data to write to HT1)
Argument15:	D	D	D	D	D	D	D	D	(PAGE N+3 DATA)
Argument16:	D	D	D	D	D	D	D	D	
Argument17:	D	D	D	D	D	D	D	D	(D = lsb data to write to HT1)
Acknowledge:	1	1	F	F	F	F	F	X	(F = Status flags)

Read Tag Page

Command to read 4 bytes of data from HT1 32 bit page. If the read was successful, indicated by acknowledge status flags then four bytes of tag data follow.

	B7		B0						
Command:	0	1	0	1	0	0	1	0	(0x52)
Argument1:	x	x	N	N	N	N	N	N	(N = HT1 page address 0-63)
Acknowledge:	1	1	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 = msb data read from HT1)
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 = lsb data read from HT1)

Read Tag Block

Command to read up to 16 bytes of data from HT1 memory. A Block is made up of four pages (each page being 4 bytes of data). If the specified page lies on the block boundary then all 16 bytes (4 pages) can be read. If the specified page is on the block boundary + 1 then 12 bytes (3 pages) can be read. In this way 16, 12, 8 or 4 bytes of data can be retrieved from the tag depending on the page number specified and it's position within the block. If the read was successful, indicated by acknowledge status flags then up to 16 bytes of tag data follow.

	B7							B0	
Command:	0	1	1	1	0	0	1	0	(0x72)
Argument1:	x	x	N	N	N	N	N	N	(N = HT1 page address 0-63)
Acknowledge:	1	1	F	F	F	F	F	X	(F = Status flags)

+ up to 16 bytes

Data only follows if read was successful

Reply1:	D	D	D	D	D	D	D	D	(D = msb data read from HT1)
Reply2:	D	D	D	D	D	D	D	D	(PAGE N DATA)
Reply3:	D	D	D	D	D	D	D	D	
Reply4:	D	D	D	D	D	D	D	D	(D = lsb data read from HT1)

|
| Up to 16 bytes can be specified depending on page address N
| ie. Perform PAGE/4 => if remainder (mod) = 0 then full block (16 bytes)
| if remainder = 1 then 12 bytes read
| if remainder = 2 then 8 bytes read
| if remainder = 3 then 4 bytes read

V									
Reply13:	D	D	D	D	D	D	D	D	(D = msb data read from HT1)
Reply14:	D	D	D	D	D	D	D	D	(PAGE N+3 DATA)
Reply15:	D	D	D	D	D	D	D	D	
Reply16:	D	D	D	D	D	D	D	D	(D = lsb data read from HT1)

Tag STATUS

Command to return Tag status.

The acknowledge byte flags indicate general Tag status.

	B7							B0	
Command:	0	1	0	1	0	0	1	1	(0x53)
Acknowledge:	1	1	F	F	F	F	F	X	(F = Status flags)

Message

Command to return product and firmware identifier string to host.

	B7		B0						
Command:	0	1	1	1	1	0	1	0	(0x7A)
Reply:	"b IDE RWD H1 (SECx V1.xx) DD/MM/YY" 0x00								

Returned string identifies author, product descriptor, project name, firmware version no. and date of last software change. 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.

NOTE:

- 1) The serial communication uses hardware handshaking to inhibit the host from sending the Micro RWD commands while Tag interrogation is in progress.
- 2) Following the Read Tag command, if an error flag has been set in the Acknowledge code then there will be NO data.
- 3) The serial communication system and protocol allows for a 10ms 'window' every Tag 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.
- 4) NOTE that only one command sequence is handled at a time.

Program EEPROM

The Micro RWD has some internal EEPROM for storing system parameters such as passwords and authorised identity codes. This command sequence allows individual bytes of the EEPROM to be programmed with new data. 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	(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	1	X	F	X	X	X	F	(F = Status flags)

Internal EEPROM memory map

Byte 0: Tag Polling Rate (x 2.5ms)
Byte 1: RF ON/OFF lock byte (0x55 = RF ON, anything else = OFF, normally set to 0x55)
Byte 2: Reserved (Checksum)
Byte 3: Encryption ON/OFF control byte (0x00 = OFF, non-zero = ON)

Byte 4:) 32 bit Encryption seed (M.S byte)
Byte 5:)
Byte 6:)
Byte 7:) (L.S byte)

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

Start of authorised tag 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 20: (MSB) Tag identity code
Byte 21:
Byte 22:
Byte 23: (LSB)

-
-
-
-

Byte 255: Last Internal EEPROM location

Encryption Methodology

The Micro RWD H1/S has a data encryption system that allows data to be stored in an encoded form that cannot be read as sensible data by any other Hitag 1 reader system.

The format of the data stored in the transponder memory (apart from the serial number, configuration and other data in Pages 0 - 3) is controlled by the Encryption Control byte in the Micro RWD internal EEPROM. If Encryption Control ON is selected then all data stored in the transponder from page 4 upwards will be encrypted, and if OFF is selected then all data is stored in standard format.

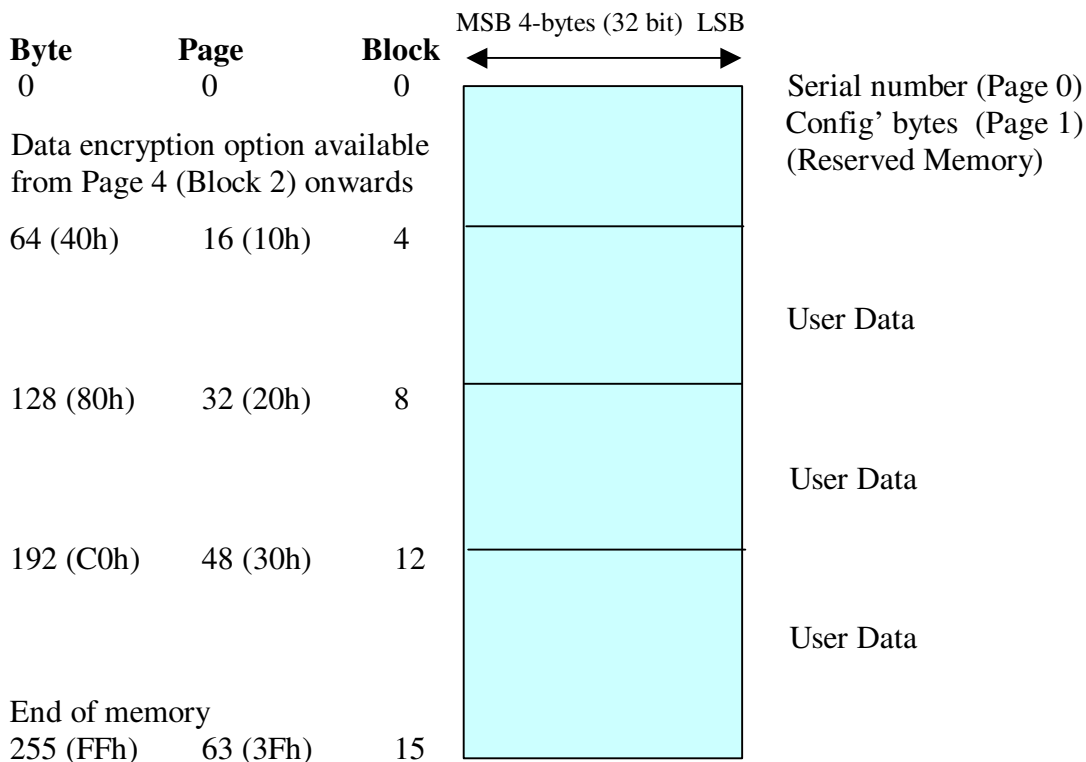
The method of encryption is unique and uses a “dynamic algorithm” which effectively makes the encoded data specific to a particular transponder and a set of encryption seed values stored in the Micro RWD internal EEPROM. This not only protects stored information but also prevents cloning of cards or copying of data. Information is encrypted when being stored and decrypted when being read, thereby making the process totally transparent to the user. Another Hitag 1 reader system would read encrypted data as random bytes with no meaning. Users should program their own encryption seed values to fully customise their system.

Method of Operation

The Micro RWD reader only allows full communication with Hitag 1 transponders if an initial level of security has been passed. The system works by firstly reading the tag serial number (identity code) which is the four bytes from page 0 (first page) of the tag memory. The Micro RWD internal EEPROM is then checked to see if this serial number is stored in the authorisation list located from byte 12 onwards. If the tag serial number is matched to a serial number stored in the Micro RWD or the list is empty then the tag has passed the validation test. If the Micro RWD has FF FF FF FF (hex) stored at EEPROM locations 12 to 15 then the list is treated as empty and all Hitag 1 tags are accepted through the validation test.

Full communication is only allowed if this initial security check has been passed (or the Micro RWD authorisation list is empty).

Hitag 1 Memory Map



Hitag 1 transponders have Pages 16 to 63 available for user data storage (192 bytes). It is advised not to use the memory locations below page 16 because these are used for configuration bytes and a “Reserved” memory area.

Hitag S transponders configured in PLAIN MEMORY mode have a similar memory map to Hitag 1 except they have available “user data ” memory from Page 2 onwards. Hitag S256 transponders therefore have Page 2 – 7 (24 bytes) for user data and Hitag S2048 types have Page 2 - 63 (248 bytes) for user data.

ib technology

Note that CON2 (Memory Lock bits) are OTP (One-Time-Programmable) if LCON = 1

Note also that the “Reserved” bits must not be altered. Page 1 must be read first and the bits that can be changed masked on/off before writing back.

No responsibility is taken for the method of integration or final use of Micro RWD

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