# IPICO Reader

## Serial Protocol

Version 1.00

21 November 2007

# NOTICE

IPICO shall not be liable for any errors or for incidental or consequential damages in connection with the furnishing, performance or use of this document, hardware and/or software.

All information in this document including the design and specification are subject to change without notice for the purpose of product improvement

For further information contact +27 12 345-9520.

# Table of Contents

# FIGURES

# TABLES

# HISTORY

| Version | Date | Person | Reason |
|---|---|---|---|
| 1.00 | 2004-01-19 | WHH | Separated from ShowTags User Manual |
| | 2004-02-10 | whh | added recent configuration additions |
| | 2004-03-16 | whh | minor changes - fix some typo’s |
| | 2004-03-18 | whh | removed reference to slave PIC in HH tag config cmd (7.16) |
| | 2004-03-19 | whh | Added additional parameter to x4 Set Timeouts |
| | 2004-06-17 | whh | Added X4 options command and fixed some other command details. |
| | 2004-07-21 | whh | Added parameter 2 description to Set IO Configuration command and added “get” variant there. |
| | 2004-08-27 | whh | Event mode implemented. |
| | 2004-09-09 | whh | Changed 7.10 get statistics, 7.19 X4 Data and 7.36 GetX4 Status |
| | 2004-10-14 | whh | Changed references to X4 to RW. Changes RW status reply. Changed RW config. |
| | 2004-10-27 | whh | Added “Sleep” command |
| | 2004-12-14 | whh | Data Page read “success flags” changed (§ 3) |
| | 2004-12-24 | whh | Added a description of the startup banner and the modifiers (§ 2). |
| | 2005-01-05 | whh | Added set/get test options (§ 7.43). |
| | 2005-01-13 | whh | Added set part of Config3 command, changed command summary table. Print banner command added. |
| | 2005-01-27 | whh | added Want Wiegand to startup modifier flags |
| | 2005-02-09 | whh | added some HW codes to § 7.10 |
| | 2005-04-13 | whh | formatting changes, changed: § 7.19, 7.20, 7.30, |
| | 2005-04-20 | whh | Description of IO Configuration words. |
| | 2005-05-17 | whh | Added filter reject count to status message. |
| | 2005-07-26 | whh | Removed IPICO UK from addresses |
| | 2005-09-21 | whh | Add some info on TTO |
| | 2005-10-05 | whh | Added TTO options (§ 7.39) |
| | 2005-10-27 | whh | Updated Message mode and Wiegand sections (§§ 5.3, 7.9 & 1.2) |
| | 2005-12-12 | whh | Updated Message mode (§ 7.9 Set Config 3 Options) and Message format (§ 7.15) – the TTO tamper field description. |
| | 2006-01-04 | whh | Added optional TTO bytes to § 1.1.2 Binary delimited records |
| | 2006-03-22 | whh | Added TTO data page formats. |
| | 2006-04-07 | whh | Reduced TTO format only has 2 bytes of TTO info (no tamper) |
| | 2007-02-06 | whh | Added Set Expected Number of TTO Pages, Get Frequency, modulate and commands |
| | 2007-03-06 | whh | Fixed contact details |
| | 2007-07-24 | whh | |
| | 2007-09-07 | whh | Command 0x3b added and 0x12 renamed as set tab baudrate. Decoder configuration word description added. Reader types added. |
| | 2007-11-20 | ra | New branding, removed reference to DIMI |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |
| | | | |

# GLOSSARY

| | |
|---|---|
| CW | Continuous Wave |
| dB | Decibels |
| dBd | Antenna gain in dB relative to dipole antenna |
| dBi | Antenna gain in dB relative to isotropic antenna |
| dBil | Antenna gain in dB relative to linearly polarized isotropic antenna |
| EIRP | Effective Isotropic Radiated Power (measured in dBi or dBil) |
| ERP | Effective Radiated Power (referred to a dipole) (measured in dBd) |
| I and Q | Quadrature RF signals (90 deg out of phase) |
| NV | Non-Volatile |
| NYI | Not Yet Implemented |
| RFID | Radio Frequency Identification |
| RFU | Radio Frequency Unit |
| RW | Read Write – referring to IP-X X4, X5 and X7 tags |
| TTO | Tag Talk Only – a tag can be configured to transmit data pages as well as its unique ID without being queried by a reader. |

# REFERENCED DOCUMENTS

1. "Low cost Read/Write UHF Identification Device", van Eeden, 2003 - p40x4data_v5.doc

2. "ShowTags User Manual", W Hofmeyr, 2004.

# 1 Summary

This manual describes the command structure and sets in order for customers to define their own User interface. Multiple data formats for serial port spooling are permitted and provided for. To accommodate ASCII based flow control such as XON/XOFF an ASCII transfer mode is supported. For higher speed throughput a binary mode is supported.

## 1.1 Definitions

### 1.1.1 Hexadecimal ASCII delimited records

Instantaneous ID hits and IDs in transit are spooled in small packets, one for each ID. These can be considered as stand alone data-grams similar to the UDP protocol. **Table 1** depicts the content of these data-grams for the hexadecimal ASCII format. This format converts the binary data to ASCII but with a hexadecimal base (e.g. 0b00101111 is converted to “2f”). All ASCII characters are lower case. In addition, the ID CRC bytes are stripped off. A simple 8-bit LRC is appended to the packet and is a modulo $2^8$ addition of all the data in the packet but not the Frame header bytes.

| Byte | Description | Info |
|---|---|---|
| 0 | Header character 1 | Frame header, ‘a’ |
| 1 | Header character 2 | Frame header, ‘a’ |
| 2-3 | Reader ID | 0-255 in ASCII hex |
| 4-15 | Tag ID | MS digit first |
| 16-19 | I and Q channel counter | Binary counters 0-255 in ASCII hex |
| 20 –33 | Date/Time | Date and time with 10ms resolution. 390ms/10 = 39 = “27” (27 = 0x32 + 0x37) and the month 12 is 0x31+ 0x32. |
|  | TTO information (optional) | 3 bytes of TTO information (see the comments section of §7.15). |
| 34-35 | LRC | Checksum on bytes 2 to 33 |
| 36-37 | End of packet (CR, LF) | 0x0d, 0x0a |

**Table 1 Hexadecimal ASCII delimited records**

Examples

***ASCII spooled ID***,

“aa400000000123450a2a01123018455927a7<CR><LF>” (38 bytes)

Reader ID 0x40 (64d), tag ID 0x000000012345, I counter 0x0a (10d), Q counter 0x2a (42d), date (20)01-12-30, 18:45:59.39, checksum 0xa7. (Note the millennium and century value is discarded in date stamp)

***ASCII LRC checksum calculation:***

‘4’+‘0’ + ….. ‘2’ + ‘7’

Calculations are done in hexadecimal on ASCII data, (i.e ASCII character ‘a’ is 0x61)

0x34 + 0x30 + 0x30 + 0x30 + 0x30 + 0x30 + 0x30 + 0x30 + 0x30 + 0x31 + 0x32 + 0x33 + 0x34 + 0x35 + 0x30 + 0x61 + 0x32 + 0x61 + 0x30 + 0x31 + 0x31 + 0x32 + 0x33 + 0x30 + 0x31 + 0x38 + 0x34 + 0x35 + 0x35 + 0x39 + 0x32 + 0x37 = 0x06a7,

discard carry thus LRC = 0xa7 or ASCII 0x61, 0x37

### 1.1.2 Binary delimited records

The sequence of data is identical to the ASCII hex format except that the data is not converted to ASCII.

| Byte | Description | Info |
|---|---|---|
| 0 | Header byte | Frame header, ‘0xaa’ |
| 1 | Reader ID | 0-255, binary |
| 2-7 | Tag ID | MS digit first |
| 8-9 | I and Q channel counter | Binary counters 0-255 |
| 10 –16 | Date/Time | Date and time with 10ms resolution, All are BCD except the 10ms counter which is binary e.g. 390ms= 390/10 = 39d = “0x27” or “27” binary and the month 12 is 0x12 or “12” BCD. |
|  | TTO information (optional) | 3 bytes of TTO information (see the comments section of §7.15). |
| 17 | LRC | Checksum on bytes 1 to 16 |
| 18-19 | End of packet (CR, LF) | 0x0d, 0x0a |

**Table 2 Binary packet format**

Examples

***Binary spooled ID***

“0xaa400000000123450a2a01123018455927fd<CR><LF>” (20 bytes)

Reader ID 0x40 (64d), tag ID 0x000000012345, I counter 0x0a (10d), Q counter 0x2a (42d), date (20)01-12-30, 18:45:59.39, checksum 0xfd. (Note the millennium and century value is discarded in date stamp)

***Binary LRC checksum calculation:***

Calculations are done in hexadecimal on binary data,

0x40 + 0x00 + 0x00 + 0x00 + 0x01 + 0x23 + 0x45 + 0x0a + 0x2a + 0x01 + 0x12 + 0x30 + 0x18 + 0x45 + 0x59 + 0x27 = 0x2fd.

discard carry thus LRC = 0xfd

### 1.1.3 TTO data page messages

TTO data pages have been supported in FPGA based readers since version 10.4

TTO data pages are only reported if the TTO information is included in the fields selected for tag ID reporting and then in one of two formats:

1. The standard tag ID format: In this format the same fields and data bytes are included in the message as are transmitted for a normal tag ID. An ID will have a page number of zero (0), while TTO data pages will have page numbers greater than zero.

2. A short TTO data page format: In this format the following rules are used to build the message:

   a. A header of "ae" is used (instead of the standard "aa" used for ID messages).

   b. The reader ID is always included.

   c. All eight (8) bytes of the data page are transmitted (even if, for example, the CRC bytes are not sent as part of an ID message).

   d. The date and time information is not included in the message even if it is selected for ID messages.

### 1.1.4 Short-Format TTO Pages

The short format TTO data page message is recognised by a header "ae" instead of the tag ID message header " aa". It is enabled by using the TTO Options Command as described in §7.39.

The transmission of optional fields is controlled by the normal ID message format options (see §7.15)

#### 1.1.4.1 ASCII Short-Format TTO message format

| Byte | Description | Info |
|---|---|---|
| 0 | Header character 1 | Frame header, 'a' |
| 1 | Header character 2 | TTO Frame header, 'e' |
| 2-3 | Reader ID | 0-255 in ASCII hex |
| 4-19 | TTO data page | MS digit first (all 8 bytes are transmitted even if not all ID bytes are sent in an ID record). |
| 20-23 | I and Q channel counter | Binary counters 0-255 in ASCII hex (optional) |
| 24-29 | TTO information | 2 bytes of TTO information (without the tamper byte) (see the comments section of §7.15). |
| 30-31 | LRC | Checksum on bytes 2 to 29 |
| 32-33 | End of packet (CR, LF) | 0x0d, 0x0a |

**Table 3 Short format ASCII TTO data**

### 1.1.4.2 Binary Short-Format TTO message format

| Byte | Description | Info |
|---|---|---|
| 0 | Header byte | Frame header, ‘0xae’ |
| 1 | Reader ID | 0-255, binary |
| 2-9 | TTO data page | MS digit first |
| 10-11 | I and Q channel counter | Binary counters 0-255 (optional) |
| 12-14 | TTO information | 2 bytes of TTO information (without the tamper byte) (see the comments section of §7.15 ). |
| 15 | LRC | Checksum on bytes 1 to 14 (optional) |
| 16-17 | End of packet (CR, LF) | 0x0d, 0x0a |

**Table 4 Short format binary TTO data**

## 1.2 *System modes, Instruction set and states*

### *1.2.1 Configuration and Maintenance mode*

When the unit is switched on for the first time by the user the default settings in par 1.2.2 take preference.

**NOTE:** The decoder unit has a number of configurable items that the user can configure via the serial port. All configurations are saved on the reader in EEPROM thus, changes made will not get lost when the reader power is switched off.

The configuration follows a framed instruction structure as depicted in **Table 6**. **Table 8** lists the valid Instructions and response codes/error codes. Frames written to the unit are acknowledged with an ACK frame which is a frame containing zero data and a copy of the Instruction that was sent to it. Only single packet Instructions are written. Instructions are in ASCII ***only*** as described in par 1.1.1 above. A terminal mode is available that ignores LRC checking in the controller if the header bytes are 'ac' instead of 'ab'. This allows configuration from a terminal without the need to calculate the LRC. In this case, the replies are still sent as per non-terminal mode with LRC's and 'ab' headers. In general, if the Reader ID in the command is zero or matches the current Reader ID, the Instruction has effect, otherwise the Instruction is rejected by the reader. Thus a broadcast message can be send to all readers connected by making the "Reader ID" bytes in the Instruction equals to x00.

### *1.2.2 Default configuration (factory settings)*

| Item | Value | Description |
|---|---|---|
| Reader ID | 0 | |
| Baud rate | 9600 | Different rates can be set via software. The new selected baud rate will then become the default value in EEPROM. |
| Transfer mode | | ASCII |
| RS 232 Protocol | 1,8,1,N | 1 starts bit, 8 Data bits, 1 stop bit, and No parity. This setting is fixed and not user definable. |
| Synthesizer | 00 | Factory Hardwired for specified frequency range |
| RF TX | 0 | Off (Only initially. If RF TX = 1, that value is stored in EEPROM until such time that RF TX is reset to "0") |
| Auxiliary output | 0 | Open collector output. (Needs pull-up device. Thus, if Aux = 0 then NO current will flow through pull-up device) |
| Flow control | None | See par 7.3 |

**Table 5 Default Factory settings**

### 1.2.3 Configuration / Command frame format

| Byte position | Field | Description |
|---|---|---|
| 0 | Header synch 1 | ‘a’ |
| 1 | Header synch 2 | ‘b’ or ‘c’ (if ‘c’, no LRC checking is done) |
| 2,3 | Reader ID | Reader ID 0-255, 0 is broadcast |
| 4,5 | Length | 0-10 |
| 6,7 | Instruction/Ack/Error | See Instruction table |
| 8 –> n | Data | Up to 10 bytes |
| n + 1 | Checksum | 8-bit LRC on all data in this table except headers, checksum and CR+LF. This is performed on the ASCII data (omit for ‘ac’ mode).<br>See the detailed description of the Set Date command (§7.1) for an example of the LRC calculation. |
| n + 2 | End of packet (CR, LF) | 0x0d, 0x0a |

**Table 6 Configuration frame format**

### 1.2.4 Reader reply frame format

| Byte position | Field | Description |
|---|---|---|
| 0 | Header synch 1 | ‘a’ |
| 1 | Header synch 2 | ‘b’ (All readers with firmware up to ver 5.1 must still be ‘a’) |
| 2,3 | Reader ID | Reader ID 0-255, 0 is broadcast |
| 4,5 | Length | 0-15 |
| 6,7 | Instruction/Ack/Error | See Instruction table |
| 8 –> n | Data | Up to 15 bytes |
| n + 1 | Checksum | 8-bit LRC on all data in this table except headers, checksum and CR+LF. This is performed on the ASCII data |
| n + 2 | End of packet (CR, LF) | 0x0d, 0x0a |

**Table 7 Reply frame**

### *1.2.5 Normal mode*

When power is applied to the reader, the reader will send a Boot-up message.

***Controller 6.1<CR><LF>***

After this message is read correctly the user can now send instructions to the reader that switch the RF transmitter on in order to read tags or send/receive other instructions as described in this document.

### *1.2.6 Diagnostic mode*

The reader has built-in test capabilities that allow the user to do periodic diagnostics on the unit.

Refer to par 7.9 and Table 8 for details.

The following parameters can be retrieved from the reader to evaluate the status of the unit

- Firmware version – represented by a decimal.hex pair of nibbles, where the first nibble is the major version number as a decimal value between 0 and 15, while the second nibble is the minor version number as a hex digit between 0 and f. Prior to version 4.9 it was translated as a binary value multiplied by 10. (i.e. version 5.3 = 53d or 0x35)
- Reader ID
- Configuration Word
- CRC count * – Indicates the number of CRC errors due to a noisy external environment or multiple tags in the beam simultaneously causing collisions.
- Power up count *– Indicates the number of power-ups the reader has had since the last read
- Activity count * - Indicates the edges received. This can be valid edges (Tag ID) or spurious noise triggers.

**** Cleared on read – after replying to the "Get Status" command the reader clears these fields.***

### *1.2.7 Offline mode*

On the standard reader, there is no additional memory available in the reader to go into an offline state. (This is where the communication between the reader and Controller (PC or other) is interrupted or suspended)

### 1.2.8 Instruction set summary

| Instruction byte | Description | Params |
|---|---|---|
| 0x01 | Set Date/Time | 0-6 = \|yy\|mm\|dd\|dw\|hh\|mn\|ss\|<br>dw = day of week 0-6 |
| 0x02 | Get Date/Time | 0-7 = \|yy \|mm\|dd\|dw\|hh\|mn\|ss\|ds<br>dw = day of week 0-6<br>7 = config |
| 0x03 | Set CONFIG1 | 0 = Byte |
| 0x04 | Set Reader ID | 0 = ID |
| 0x05 | Set RF synth configuration | P0 0 – 3 for corresponding S0, S1 values. |
| 0x06 | RF TX on/off | Set RF TX pin state in param 0, bit 0, 0 = logic '0' |
| 0x07 | Aux. Output | Open collector Output |
| 0x08 | CRC seed | Sets the seed to be used for CRC calculation<br>0-1 the seed MSB first |
| 0x09 | Set CONFIG3 | P0 = Byte (message mode)<br>Bit 2-0 mode<br>000 = normal<br>001 = trigger 1<br>010 = trigger 2<br>011 = event<br>Bit 3 – Send status as tag<br>Bit 4 – Sleep on startup<br>Bit 5 – Sleep active low<br>Bit 6 – Sleep tracks RF Tx<br>Bit 2 – Send tags even when RF Tx is off<br>P1 = event mode timeout<br>P2 = mask for changing only part of CONFIG3 |
| 0x0a | Get stats | N parameters each one byte in length<br>Parm 0 FW version,<br>Parm 1 Reader ID<br>Parm 2 Config Word<br>Parm 3 CRC count *<br>Parm 4 Power up count *<br>Parm 5 Activity count (reserve for future use) *<br>Parm 6: Decoder I channel FW version<br>Parm 7: Decoder Q channel FW version<br>Parm 8: Second Configuration word.<br>Parm 9: Wiegand Configuration word.<br>Parm 10: Timer value for test Wiegand.<br>Parm 11: Third Configuration word.<br>* Cleared on read |
| 0x0b | Self test | Parm 0, bits<br>Ls bit 0 – RTC clock 0 = fail<br>Bit 1 – RTC mem 0 = fail<br>Bit 2 - Config checksum 0 = fail |
| 0x0c | Boot load - Initiate the boot loader | 0: 0 or 1 – if 1 then the bootloader is invoked by executing a "Reset" command, otherwise a branch is executed. |
| 0x0d | set CRC checking options | 0 = Byte |
| 0x0e | set Beep options | 0 = Byte |
| 0x0f | set Wiegand options | 0 – options<br>1 – interval (optional)<br>2 – dummy tag interval (optional) |
| | | |

| Instruction byte | Description | Params |
|---|---|---|
| 0x10 | Bootload a slave (decoder) PIC | 0: slave ID – 0 = I, 1 = Q |
| 0x11 | set tag ID message format options | 0 – info options<br>1 – which ID bytes to send (optional)<br>2 – ASCII header byte 1 (optional)<br>3 – ASCII header byte 2 (optional)<br>4 – Binary header byte 1 (optional)<br>5 – binary header byte 2 (optional)<br>6 – trailer byte 1 (optional)<br>7 – trailer byte 2 (optional) |
| 0x12 | set tag configuration | P0<br>7-3 unused<br>2-0 tag baudrate<br>0 32kbps<br>1 64kbps<br>2 128kbps<br>3 256kbps (only 64k and 256k supported to D040) |
| 0x13 | set “nurse” tag | P0 – P7 tag ID |
| 0x14 | get “nurse” tag | R0 – R7 tag ID |
| 0x15 | set RW command/data | P0 = 0 for command<br>1 for data<br>P1 – P8 = command/data |
| 0x16 | transmit RW command (plus data) | none |
| 0x17 | set RW transmission rate | P0 = period (150 == 4kbps) |
| 0x18 | reset factory defaults | none |
| 0x19 | configure IO pins | Po:0 In1 is Tx Switch<br>P0:1 In 1 is active low<br>P0:2 Out 1 is TxStat<br>P0:3 Out 1 is Wiegand D1<br>P0:4 Out 1 is active low<br>P0:5 Out 2 is Aux1<br>P0:6 Out 2 is Wiegand D0<br>P0:7 Out 2 is active low<br>P1:0 Out 3 is “Health” (heartbeat)<br>P1:1 Out 3 is valid tag<br>P1:2 Out 3 is active low<br>P1:3 unused<br>P1:4 Send message on In 1 0 – 1 change<br>P1:5 Send message on In 1 1 – 0 change<br>P1:6 unused<br>P1:7 unused |
| 0x1a | get IO | R0:0 Input 1 value<br>R0:1 RF Tx state |
| 0x1b | get IO settings | R0, R1 = P0, P1 of set IO cofig |
| 0x1c | set Output 1 | P0:0 value for Output 1 |
| 0x1d | Fast Multiplex times | P0: delay<br>P1: mark<br>P2: space |
| 0x1e | Fast Mux start (sync) | none |
| 0x1f | Set Sleep | P0: 0 = sleep off (awake), 1 = sleep |
| 0x20 | RW Action command | P0: seq no<br>P1:7-4 start page<br>P1:3-0 num pages<br>P2:7-5 X4 command<br>P:4 extended<br>P:3 auto suspend<br>P:2 continuous<br>1-0 target – 0 = immediate, 1 = broadcast, 2 = addressed, 3 = specific.<br>P3-P10 (optional) Tag ID |

| Instruction byte | Description | Params |
|---|---|---|
| 0x21 | RW data | P0-P7 data to write |
| 0x22 | RW tag match mask | P0-P7 mask |
| 0x23 | RW command stop | none |
| 0x24 | RW set timeouts | P0: command timeout (NYI)<br>P1: read timeout<br>P2: redo time (recommand same tag)<br>P3: read delay (after write)<br>P4: retry limit (count). |
| 0x25 | RW immediate action | P0: X4 command (suspend or resume) shifted into place for air protocol ("a0" – suspend, "e0" – resume). |
| 0x26 | RW get status | none<br>15 bytes in reply (see 7.36) |
| 0x27 | Save RW settings to EEPROM | none |
| 0x28 | Set RW Options | P0: see §7.38<br>P1: |
| 0x29 | Set TTO Options | P0: see §7.39<br>P1: |
| 0x2a | Set TTO number of pages expected & report only N | P0: N |
| 0x2b | Get FPGA statistics | None |
| | | |
| 0x30 | Tag filtering – set select pattern | P0-P7: the select pattern |
| 0x31 | Tag filtering – set select mask | P0-P7: the select mask |
| 0x32 | Tag filtering – set reject pattern | P0-P7: the reject pattern |
| 0x33 | Tag filtering – set reject mask | P0-P7: the reject mask |
| 0x34 | tag filtering – save patterns to EEPROM | none |
| 0x35 | Set/Get test options | P0 – the options |
| 0x36 | Modify part of CONFIG3 | P0 – mask<br>P1 – bits<br>P2 – event mode timeout.(optional) |
| 0x37 | Print the start-up banner without resetting the controller | none |
| 0x38 | Get frequency | Note: only on test board FW |
| 0x39 | Set Hopping timing | P0: - dwell time (10ms units)<br>P1: - modulate time (microfortnights) |
| 0x3a | Set modulate | P0:0 – 0 – not modulate, 1 - modulate |
| | | |
| | | |
| 0x40 | Dump memory | ARM |
| 0x41 | Dump SEEPROM memory | ARM7 |
| 0x42 | Write memory | ARM |
| 0x43 | Set IMASK | ARM7 |
| | | |
| 0x80 | Send last I2C read contents. The address, status and data of the last I2C read operation are returned. – used for debugging. | The offset into the read buffer. |
| 0x81 | Send an I2C message. A read or write message can be sent on the I2C bus.<br>– used for debugging | addr\|R_W\|n\|data....<br>0 – the unshifted I2C address<br>1 – Read = 1, write = 0 |

| Instruction byte | Description | Params |
|---|---|---|
|  |  | 2 – the number of bytes to read/write<br>3… the data for a write operation |
|  |  |  |

Table 8 Instructions Set Summary

| Error byte | Description | Comments |
|---|---|---|
| 0xf0 | Bad length ( >10 ) | |
| 0xf1 | Bad LRC | |
| 0xf2 | Bad Instruction (unknown) | |
| 0xf3 | Reserved | |
| 0xf4 | Unsupported command | The command is not supported by this reader configuration. |
| 0xf5 | Unsupported sub-command | The sub-command is not supported by this reader configuration (other sub-commands associated with the command may be supported). |

**Table 9 Errors Codes**

# 2 Start-up Banner

On start-up the controller PIC transmits two messages on the serial port. The first is the Bootloader version string and then, after the bootloader has timed-out, the controller transmits its own version string.

Both these messages are transmitted at 9600 baud.

## 2.1 *Bootloader Start-up message*

The bootloader first transmits its version string that will be “Bootloader v1.5” for newer devices or “Bootloader v1.4” for older devices. This is followed by a carriage-return + line-feed pair.

After waiting for possible loader data from the host, the bootloader sends another carriage-return + line-feed pair and returns control to the controller firmware.

## 2.2 *Controller Start-up message*

The controller version message has many forms, the most common, historically, being of the form:

“Controller 08.0 2004/11/16 (RWB)”

Where the version (here version 8.0) and the build date (here 16 November 2004) are displayed. Both values are important to determine the capabilities of the reader and when requesting support.

The version string is followed by a set of modifier codes enclosed in parentheses – these codes are also important when requesting support. Their meanings are described in the next paragraph.

This form was used on all readers equipped with PIC decoder front-ends. Newer readers that use an FPGA decoder will have one of the following forms.

Controller for RW Actel FPGA v10.1 2004/12/22 (RWBF)

Some debug builds of the firmware are flagged with additional messages which serve as a warning that IO lines and even tag IDs may be used for debug purposes. These

### 2.2.1 *Version Modifier Codes*

| | |
|---|---|
| DT | "Take-on" or desktop reader using HH decoder |
| RW | Read/Write firmware |
| P | The reader configuration is stored in EEPROM (not in Dallas NV RAM) |
| S | used "bit-blasting" I2C (not PIC device) |
| 2A | ancient hardware is being used |
| 1A | antique hardware |
| X | PULSE_ON_SEE_OR_SEND – The Aux1 (Out2) pin is pulsed when a tag is seen or when the ID is sent to the host. |
| L | Aux1 (Out2) is active low |
| w | periodically send a test ID on Wiegand |
| W | HH Wiegand or Wiegand output is required on a standard decoder. |
| 0 (zero) | default CRC is P4022 (else iPX) |
| B | both CRC flavours are tested/accepted |
| I | ignore CRCs |
| E | the controller tries testing for the EmM bad CRC syndrome (early X4, X6, X7) |
| C | an old 'C' type PIC is being used (not field programmable) |
| U | unprogrammed tags are 'good' |

| | |
|---|---|
| Y | TX cycles 50% duty-cycle (DF HH) |
| A | antique RFU (TX_Off signal polarity different) |
| R | an old C type PIC has been reprogrammed |
| RR | an old C type PIC has been reprogrammed twice |
| WW | Want Wiegand output (pre 2005-01-27 only "W") |
| F | FPGA decoder is being used |
| A | early versions of FPGA used the Aux1 (Out2) pin for FPGA control and were thus not available for other purposes. |
| S | early versions of FPGA used the freq select pins for FPGA control and were thus not available for other purposes. |

### *2.2.2 Additional Start-up messages/warnings*

Some debug builds of the firmware use IO lines to provide debug information for development and some embed status information in the Tag IDs. These builds are flagged by displaying the following warning messages as required:

"IO pins are used for debug signals"

"Tag data contains debug information"

## 3 RW page data Reporting Format

The Advanced RW commanding functions introduced with controller versions 7.7 and higher (Handheld V3.a and higher), result in asynchronous reporting of page data.

The format of these report packets depends on the currently selected tag reporting format

These report packets have the following format:

| Byte(1) | Byte(2) | Byte(3) | Description | Info |
|---|---|---|---|---|
| 0 | 0 | 0 | Header character 1 | Frame header, 'a |
| 1 | 1 | 1 | Header character 2 | Frame header, 'd |
| 2-3 | | 2-3 | Reader ID | 0-255 in ASCII hex |
| 4-15 | 2-17 | 4-19 | Tag ID | MS digit first |
| 16-17 | 18-19 | 20-21 | Page number | the page which returned the included data and success flags |
| 18-19 | 20-21 | 22-23 | Sequence Number | The sequence number (from the host) that caused this data report.<br>The retries remaining counts can optionally be returned here – see §0. |
| 20-35 | 22-37 | 24-39 | Data | the data read |
| 36-37 | | 40-41 | LRC | Checksum on bytes 2 to 35/39 |
| 38-39 | 38-39 | 42-43 | End of packet (CR, LF) | 0x0d, 0x0a |
| | | | | |

**Table 10 RW Page Data Report Format**

The byte positions listed above are for: (1) standard reporting format, (2) reduced reporting format and (3) standard reporting format but with CRC included.

What it boils down to is that the data report has the same fields as a tag ID report:

- there is a reader ID in the data report if there is one in the tag report
- the CRC of the tag is excluded if it is excluded in the tag report
- the LRC is included if it is included in the tag report

Example data messages:

**ad45800000192f798b0108aaaaaaaaaaaaaaaa** when reduced mode is being used

```
hdr      tag ID       pg seq    data
ad   45800000192f798b 01 08   aaaaaaaaaaaaaaaa
```

**ad0045800000191b002aa5555555555555555d** when in standard mode

```
hdr  RID   tag ID          pg  seq   data                   LRC
ad   00   45800000191b  00  2a   a555555555555555  5d
```

**ad0045800000191bf9d4002ca55555555555555596** when in standard mode with CRC

```
hdr  RID   tag ID  (inc CRC)   pg  seq   data                   LRC
ad   00   45800000191bf9d4  00  2c   a555555555555555  96
```

Versions 7.b to 7.f of the controller used the top nibble of the page number byte to signal success or otherwise of both read and write operations.

| Bit(s) | Meaning | Comment |
|---|---|---|
| 7 (version < 8) | Seen on I | The data was decoded on the I channel of the dual channel reader |
| 6 | Seen on Q | same as above but Q channel. |
| 5-4 | count | The total number of times this data was seen. A successful read will have a count of two (2) if the data was seen only on a single channel else the count will be three (3).<br>A write operation is considered to be successful if a single read matches the written data but the reader sets the count to three (3) on detection of a match. |

From version 8.0 the top nibble of the page number byte used a different mechanism to signal success or otherwise of both read and write operations.

| Bit(s) | Meaning | Comment |
|---|---|---|
| 7-6 | I count | The number of times this data was seen on the “I” channel. |
| 5-4 | Q count | The number of times this data was seen on the “Q” channel. |

A count of >= 2 on either channel is taken as an indication of a “good” read. (This can be tested by ANDing with 0xa0 and accepting a non-zero result as a good read.)

# 4 Troubleshooting

**Visual indicator guide**

- Steady RED light indicates Power ON but no processor/communication activity.
- Slow AMBER (green and red mixed) Flash at 1/6Hz indicates Internal processor working.
- Fast AMBER (green and red mixed) Flash at 1/2Hz indicates communication activity between PC and reader. A command must be received by the reader to activate the fast heartbeat.
- A random AMBER fast Flash indicates a valid Tag ID.

| Symptom | Possible causes |
|---|---|
| Slow Amber flash with no communication possible between PC and reader | Communication Cable not connected or faulty wiring. |
| | Application software not running |
| | Baud rate incorrect |
| Cannot read a Tag although PC communicates with reader (fast heartbeat) | Transmitter not switched ON |
| | Faulty Tag |
| | Tag not orientated correctly. |
| | Faulty Reader front end |

**Table 11 Troubleshooting guide**

# 5 Descriptions of Configuration words

## 5.1 *Configuration word 1*

The first configuration word controls the serial communication parameters and is modified using the "Set Configuration" command described in par. 7.3 below. It also contains two status bits which are

| Bit | Item | Description |
|---|---|---|
| 0 | ASCII mode | 0 = OFF (binary mode), 1 = ON, controller to host traffic only, commands to controller are always in ASCII. |
| 2-1 | Flow control | 00 – none<br>11 - RTS/CTS<br>10 - XON/XOFF |
| 5-3 | Baud rate | 000 – Unused<br>001 – 4800<br>010 – 9600 (Default)<br>011 – 19200<br>100 – 38400<br>101 – 57600<br>110 – 115200 |
| 6 | RT Clock operating OK<br>Disable Synth I2C error | Bit. '1' = pass (Read only) NYI<br>When set the error message associated with the failure of an I2C message to the Synth is discarded. |
| 7 | Config checksum | Bit. '1' = pass (Read only) NYI |

## 5.2 Configuration word 2

Configuration word 2 contains configuration information controlling the RF frequency, the presence of RF carrier, and some digital IO options, and is modified using several commands:

- "Set synth", described in par. 7.5 below.
- "Set RF Tx", described in par. 7.6 below.
- "Set Aux1", described in par. 7.7 below.
- "Set Aux1 pulse options", described in par. 0 below.
- "Set Output1", described in par. 0 below.

| **Bit** | **Item** | **Description** |
|---|---|---|
| 1 - 0 | Synth configuration | Selects RF carrier frequency and mode (CW or frequency hopping). |
| 2 | Output 1 | This bit represents the logical level to be output to the Output 1 pin when this is pin is not being used to reflect the RF Tx status (see par. |
| 3 | RF Tx | This bit represents the logical presence of the RF carrier |
| 4 | Aux 1 | This bit represents the logical level to be output to the Aux 1 (Output 2) pin when this is pin is not being used as a pulsed output to report tag activity (see par. xx). |
| 5 | Aux1 Pulse on Send | The Aux1 (output 2) pin is pulsed when a tag ID is transmitted to the host (see § xx). |
| 6 | Aux1 Pulse on See | The Aux1 (output 2) pin is pulsed each time a tag ID is detected (see § xx). |
| 7 | Aux1 Pulse re-trigger | Normally a pulse, once started, will be allowed to complete (for the period as set by the "Pulse configuration command" as described in § xx), before a tag event will start a new pulse. When this bit is set the pulse will be re-triggered, that is the pulse width will be extended by the nominal pulse width whenever a tag event is encountered. This can cause a continuous high level on the output pin while a tag is present in the reader beam. |
| NOTE: If all three bits (7 – 5) are set, then the reader sets AUX1 whenever a tag is seen. AUX1 is reset when the "Nurse" tag is seen. This is known as "Hospital" mode. AUX1 is connected to a buzzer, which sounds when any tag (a baby) is seen by the reader. A nurse has to reset this alarm state using a special tag (Nurse tag). | | |

## 5.3 Configuration word 3

Configuration word 3 contains the desired tag transmission behaviour and is controlled using the "Set Configuration word 3" command described in par, 7.9 below.

| Bit | Item | Description |
|---|---|---|
| 2 – 0 | message mode | 000 = normal<br>001 = event 2<br>010 = trigger 2<br>011 = event |
| 3 | send status as tag | If set, any status message from the decoders will be handled as if it were a special tag and transmitted to the host. |
| 4 | Sleep | If set then the controller will assert the "Sleep" pin on startup. |
| 5 | Sleep active low | If set then the sleep pin must be pulled low to cause the RFU to sleep |
| 6 | Sleep tracks RF Tx | If set then the sleep state tracks RF Tx Off – the RFU will be put into sleep mode whenever the RF power is off. |
| 7 | Listen while RF off | if set then tags can be read even when the RFU is not transmitting.  If not set then tags can |

## 5.4 IO Configuration word 1

IO Configuration word 1 defines how the Output-1 and Output-2 pins are used. It is controlled using the "Configure the IO pins on the Reader" command, described in par 7.23 below.

| Bit | Item | Description |
|---|---|---|
| 0 | In1 is Tx Switch | If 1, then the Input 1 line is used to perform the RF TX switch function. Whether the RF carrier is transmitted or not is an "OR" function between this input and the state as set by the host using the RF Tx On/Off command described in 7.6 below.<br>If 0, then the state of this input is not used to control the RF carrier.<br>In either case the level of this input can be determined by the host using the Get IO command described in 7.24 below. |
| 1 | In 1 is active low | This bit controls the polarity of the "Input 1" signal.<br>If this bit is set then a low input on the pin will be interpreted as "on" and, for example, turn on the RF carrier when it is being used for this purpose. |
| 2 | Out 1 is TxStat | This bit controls whether the state of the RF carrier is reflected on the Output 1 pin.<br>If set, then the "Output 1" pin will be set when the RF carrier is being transmitted.<br>If the bit is clear then the pin will not track the RF carrier state.<br>**NOTE: Under normal operating conditions, bits 2 and 3 should not both be set.** |
| 3 | Out 1 is W1 | This bit controls whether the "Output 1" bit is used as the Wiegand D1 data line.<br>**NOTE: Under normal operating conditions, bits 2 and 3 should not both be set.** |
| 4 | Out 1 is active low | The sense of this "Output 1" pin is controlled by this bit.<br>If this bit is set then a logical "1" will be represented by a low level on the Output 1 pin. |
| 5 | Out 2 is Aux1 | This bit controls whether the Output 2 pin is used for the Aux1 signal, used on the Dual frequency unit as a buzzer output. |
| 6 | Out 2 is W0 | This bit controls whether the Output 2 pin is used for the Wiegand D0 signal. |
| 7 | Out 2 is active low | This bit controls the sense of the Output 2 pin. If set then a logical "1" will be represented by a low signal on the Output 2 pin. |

### 5.5 IO Configuration word 2

IO Configuration word 2 defines how the Output-3 pin is used, as well as whether unsolicited IO status messages are sent when Input-1 changes. It is controlled using the "Configure the IO pins on the Reader" command, described in par 7.23 below.

| Bit | Item | Description |
|---|---|---|
| 0 | Out 3 is health | This bit controls whether the Output 3 pin is used to display the "Heartbeat" on the reader.<br>If set then the normal health state of the reader is displayed on the Output 3 pin in the form of a heartbeat which flashes an LED.<br>If this bit is not set then the heartbeat will not be change the Output 3 pin. |
| 1 | Out 3 is valid ID | This bit controls whether the Output 3 pin is flashed whenever a message is transmitted on the reader's serial output port.<br>If it is set then the transmission of a valid tag ID will cause the Output 3 pin to be flashed. Replies to host commands and now unsolicited status and IO messages will also flash the pin.<br>If clear then communication messages will not be signalled on the Output 3 pin. |
| 2 | Out 3 is active low | This bit controls the sense of the Output 3 pin.<br>If set then a logical "1" will be represented by a low signal on the output pin. |
| 3 | unused | |
| 4 | Send In1 0-1 message | This bit controls whether a transition from 0 to 1 on the Input 1 pin will cause an unsolicited "Get IO" message to be sent to the host.<br>If set then a 0-1 transition on the Input 1 pin will cause the reader to send the same message as it would have sent in reply to a "Get IO" command described in 7.24 below. |
| 5 | Send In1 1-0 message | This bit controls whether a transition from 1 to 0 on the Input 1 pin will cause an unsolicited "Get IO" message to be sent to the host.<br>If set then a 1-0 transition on the Input 1 pin will cause the reader to send the same message as it would have sent in reply to a "Get IO" command described in 7.24 below. |
| 6 | UNSOL_STATUS_REQ | Used internally by the firmware to indicate that an unsolicited status message (0x0a) is to be sent. |
| 7 | UNSOL_IO_STATUS | Used internally by the firmware to indicate that an unsolicited IO status message (0x1a) is to be sent. |

### 5.6 IO Configuration word 3

IO Configuration word 3 defines how the Output-1 and Output-2 pins are used. It is controlled using the “Configure the IO pins on the Reader” command, described in par 7.23 below.

| Bit | Item | Description |
|---|---|---|
| 0 | HH Wiegand signals active low | In a Handheld controller with Wiegand firmware, both Wiegand signals are active low. |
| 1 | Imput2 is active low | The Input-2 signal is active low. |
| 2 | Beep Override | If set then user control of Output 2 will override its use to generate a beep pulse on detecting a tag. |
| 3 | Output1 tracks Input2 | When set the Output1 signal will track the logic level of the Input2 signal. |
| 4 | Output2 tracks Input2 | When set the Output2 signal will track the logic level of the Input2 signal. |
| 5 | Ouput3 tracks Input2 | When set the Output3 signal will track the logic level of the Input2 signal. |
| 6 | not used | |
| 7 | INPUT2_LOGIC | Not user modifiable – holds the logic value of the Input2 signal. |

## 5.7 RW Configuration word 1

RW Configuration word 1 contains the desired RW behaviour of the reader and is controlled using the "Set RW Options" command described in par, 7.38 below.

| Bit | Item | Description |
|---|---|---|
| 0 | Auto read after write | A confirming read is performed after any write action to verify the successful writing of the data. |
| 1 | Auto suspend/resume | Auto suspend/resume |
| 2 | Specific tag | The reader waits until a specific tag is seen and then sends the command to that tag. Using the select mask can address a range of tags. |
| 3 | Addressed | If set then the ID of the tag is used as the target, else the broadcast address is used. |
| 4 | Continuous | Repeat the command until stop is issued |
| 5 | Voting | Use "voting" and multiple reads |
| 6 | Want Sequence # | If set then the command sequence number is returned in the page data message (the default), else the remaining retry and read retry counts are packed into a byte in the message. |
| 7 | Use S-list | Use any data records available in the S-list (Pages already sent to the host) when "voting". |

## 5.8 RW Configuration word 2

RW Configuration word 3 contains the desired data page CRC checking options and is set using the "Set RW Options" command described in par, 7.38 below.

| Bit | Item | Description |
|---|---|---|
| 0 | CRC result | Set if the last data page tested passed the CRC check |
| 1 | Generate data CRC | If set – the controller will generate the CRC for the data (NYI) |
| 3 – 2 | Data CRC options | CRC checking to be performed on data<br>00 – none<br>01 – 4022<br>10 – IPX<br>11 – CCITT |
| 4 | Discard bad reads | If set – unconfirmed reads are not transmitted to host (NYI) |
| 5 | Pass P15 CRC | If set – CRC checking is not performed on page 15 |
| 6 | not used | |
| 7 | not used | |

## 5.9 Decoder Configuration word

Decoder Configuration word contains the tag baud rate and the configuration parameters used by the FPGA decoders.

.

| Bit | Item | Description |
|---|---|---|
| 1-0 | Baud rate | 0 – 32kb/s (Not yet supported)<br>1 – 64kb/s<br>10 – 128kb/s (Not yet supported)<br>11 – 256kb/s |
| 2 | DSP control | 0 = normal, 1 = DSP filter bypass |
| 3 | Input invert | 0 = no invert, 1 = invert input signal |
| 4 | Modulate control | 0 = OFF, 1 = ON |
| 5 | Duty cycle control | 0 = 50%, 1 = 100% |
| 6 | Synthesizer control | 0 = No carrier, 1 = 125 kHz carrier |
| 7 | TXOFF polarity control | 0 = Normal, 1 = Inverted |

# 6 Description of Digital IO Functions

The Reader has a single digital input and three digital outputs available.

Under normal operating conditions these IO lines are assigned to the following standard functions:

- INPUT1, the digital input, is used to control the RF carrier (TX_SW – now INPUT1)
- OUTPUT1 reflects the state of the RF carrier (TX_STAT – now OUTPUT1)
- OUTPUT2 is used as a general purpose output under host control (AUX1 – now OUTPUT2)
- OUTPUT3 is used to indicate tag activity (actually it indicates message transmission to the host) (VALID_ID – now OUTPUT3)

These IO lines can be re-configured to perform other activities:

- The OUTPUT1 (TX_STAT) and OUTPUT2 (AUX1) lines can be used to transmit Wiegand protocol.
- The controller can be programmed to generate a pulse on the OUTPUT2 (AUX1) line whenever tag activity is detected.
- OUTPUT1 (TX_STAT) can be used as a general-purpose output under host control.
- The OUTPUT3 (VALID_ID) will be available as a general-purpose output under host control (as of 2003-07-08 no reader functionality is available to control this line).

The following table lists the function of the IO lines in the standard IPICO readers:

| | | **I/O** | **Default** | **Wiegand** |
|---|---|---|---|---|
| **UHF/uW/OEM RFU** | | | | |
| | Input 1 | Non isolated input | Tx Switch | Isolated input (LED ) |
| | Output 1 | TTL output | Tx Status | Isolated output (D1) |
| | Output 2 | TTL output (OD) | Aux output | Isolated output (D0) |
| **DF** | | | | |
| | Input 1 | Isolated input (Push button) | Tx Switch | Isolated input (LED ) |
| | Output 1 | (Relay output) Buzzer | Tx Status | Isolated output (D1) |
| | Output 2 | (Relay output) Strobe | Built-in buzzer | Isolated output (D0) |
| **Handheld** | | | | |
| | GPIO_2 | TTL input (“Read”) | | |
| | Input 1 | TTL input NC | | |
| | Output 1 | TTL output NC | | |
| | Output 2 | TTL output NC | | |

The current handheld unit does not use the standard digital I/O lines, but uses an input line to control RF power and filtering of tag records (a tag code is transmitted only once during a read event).

# 7 Detail Instruction Set Descriptions

## 7.1 Set Date

| | | |
|---|---|---|
| **Function:** | Instruction to set Real Time Clock (RTC) date and time. | |
| **Function Code** | 0x01 | |
| **Parameters:** | Parameters 0-6: | The date and time in the format yymmddDDhhMMss where yy is the year (excluding the century), mm is the month (Jan = 01), dd is the day-of-month, DD is the day-of-week (Mon = 1), hh is the hours (24hr clock), MM is the minutes and ss the seconds. |
| **Description:** | Instruction forces RTC on and sets RTC interrupt at 1-second interval.<br>The clock is Y2K compliant, handles leap year adjustments, and is a full 100-year calendar. Accuracy is 2 minutes per year. Resolution is 1 second. The time is set in 24 hour format e.g. 10:00 pm is 22H00. | |
| **Example:** | ***Set date***<br>Thursday (4th day of week), 10 Jan 2002, 10:15:23 p.m.<br>To controller:<br>Hdr ID len cmd date day time LRC<br>“ab” “00” ”07” “01” “020110” “04” “221523” “df’ <CR><LF><br>***Example of LRC checksum calculation:***<br>‘0’ + ‘0’ + ‘0’ + ‘7’… …‘2’ + ‘3’<br>Calculations on ASCII data are performed in hexadecimal, (i.e ASCII character ‘a’ is 0x61)<br>0x30 + 0x30 + 0x30 + 0x37 + 0x30 + 0x31 + 0x30 + 0x32 + 0x30 + 0x31 + 0x31 + 0x30 + 0x30 + 0x34 + 0x32 + x32 + 0x31 + 0x35 + 0x32 + 0x33 = 0x03df<br>Discard the carry (0x03) thus LRC = 0xdf or ASCII 0x64, 0x66<br>The same Instruction can be send without LRC checking i.e.<br>Hdr ID Len cmd date day time<br>“ac” “00” “07” “01” “010210” “04” “221523” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “01” “21” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. | |
| **Bytes:** | “ac” command = 22 bytes, “ab” command = 24 bytes | |
| **Encoding:** | “ac000701…” | |
| **Operations:** | Controller → reader | |

## 7.2 Get Date

| | |
|---|---|
| **Function:** | Instruction to retrieve current RTC date and time. |
| **Function Code** | 0x02 |
| **Parameters:** | none |
| **Results:** | 9 bytes in same format as ***Set Date*** Instruction with the CONFIG parameter appended. |
| **Example:** | ***Get date***<br>Replies with Tuesday, 23 Apr 2002, 14:32:23 p.m. and CONFIG<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “02” “22” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“ac” “00” “00” “02” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd date day time config Lrc<br>“ab” “00” “09” “02” “020423” “02” “14235117” “11" "b2" <CR><LF><br>***Error reply***<br>Set date with bad LRC<br>Hdr ID Len cmd date day time Lrc<br>“ab” “00” “07” “01” “010210” “04” “221523” “**a3**” <CR><LF><br>Reply from controller (error):<br>Hdr ID Len Err Lrc<br>“ab” “00” “00” “f1” “57” <CR><LF> |
| **Bytes:** | 8 or 10 bytes |
| **Encoding:** | “ac000002” |
| **Operations:** | Controller → reader |

## 7.3 Set configuration

| | | | |
|---|---|---|---|
| **Function:** | Instruction to set configurable items. | | |
| **Function Code** | 0x03 | | |
| **Parameters:** | Parameter 0: the configuration bits | | |
| | **Bit** | **Item** | **Description** |
| | 0 | ASCII mode | 0 = OFF (binary mode), 1 = ON, controller to host traffic only, commands to controller are always in ASCII. |
| | 2-1 | Flow control | 00 – none<br>11 - RTS/CTS<br>10 - XON/XOFF |
| | 5-3 | Baud rate | 000 – Unused<br>001 – 4800<br>010 – 9600 (Default)<br>011 – 19200<br>100 – 38400<br>101 – 57600<br>110 – 115200 |
| | 6 | RT Clock operating OK<br>Synth error suppression | Bit. ‘1’ = pass (Read only)<br>When set the error message associated with the failure of an I2C message to the Synth is discarded. |
| | 7 | Config checksum | Bit. ‘1’ = pass (Read only) |
| **Results:** | No results are returned – only ACK message | | |
| **Example:** | ***Set Config***<br>To set ASCII mode without flow control and 9600 baud<br>To controller:<br>Hdr ID len cmd cfg LRC<br>“ab” “00” “01” “03” “11” “56” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd cfg<br>“a**c**” “00” “01” “03” “11” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “03” “23” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. | | |
| **Bytes:** | 10 or 12 | | |
| **Encoding:** | “ac000103…” (use ac00e203bf40 (and or option) to set bit 6) | | |
| **Operations:** | Controller → reader | | |

## 7.4 Set reader ID

| | |
|---|---|
| **Function:** | Instruction to set the reader ID. |
| **Function Code** | 0x04 |
| **Parameters** | Parameter 0: the reader ID to use |
| **Results:** | No results are returned – only ACK message |
| **Description:** | The reader ID can be set to a value between 0 and 255 (0xff hex). An ID of zero is used as the broadcast address, so zero is not useful when assigning a specific address to a reader. |
| **Example:** | ***Set reader ID***<br>To set the reader ID to 100 (0x64)<br>To controller:<br>Hdr ID len cmd ID LRC<br>“ab” “00” “01” “04” “64” “8f” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd ID<br>“a**c**” “00” “01” “04” “64” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “04” “24” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 10 or 12 |
| **Encoding:** | “ac000104…” |
| **Operations:** | Controller → reader |

## 7.5 Set Synth Control lines

<table>
<tr><td><b>Function:</b></td><td colspan="2">Instruction to configure the “Synth” control bits.</td></tr>
<tr><td><b>Function Code</b></td><td colspan="2">0x05</td></tr>
<tr><td rowspan="6"><b>Parameters</b></td><td colspan="2">Parameter 0: 0 – 3 to set the desired frequency and mode</td></tr>
<tr><td><b>Value</b></td><td><b>Frequency and mode</b></td></tr>
<tr><td>0</td><td>915.3 – 921.6MHz frequency hopping</td></tr>
<tr><td>1</td><td>869.4MHz fixed frequency</td></tr>
<tr><td>2</td><td>915.3MHz fixed frequency</td></tr>
<tr><td>3</td><td>undefined – used for testing during manufacture (DO NOT USE).</td></tr>
<tr><td><b>Results:</b></td><td colspan="2">No results are returned – only ACK message</td></tr>
<tr><td><b>Description:</b></td><td colspan="2">This Instruction sets the Synth0 and Synth1 pins. It also logs the setting to the NV configuration area.</td></tr>
<tr><td><b>Example:</b></td><td colspan="2">Set Synth0 to 0 and Synth1 to 1 (915.3MHz)<br>To controller:<br>Hdr ID len cmd val LRC<br>“ab” “00” “01” “05” “02” “LL” &lt;CR&gt;&lt;LF&gt;<br>or for terminal mode:<br>Hdr ID len cmd val<br>“ac” “00” “01” “05” “02” &lt;CR&gt;&lt;LF&gt;<br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “0c” “LL” &lt;CR&gt;&lt;LF&gt;<br>In the case of a terminal Instruction with “ac”, we have the same reply.</td></tr>
<tr><td><b>Bytes:</b></td><td colspan="2">10 or 12</td></tr>
<tr><td><b>Encoding:</b></td><td colspan="2">“ac000106…”</td></tr>
<tr><td><b>Operations:</b></td><td colspan="2">Controller → reader</td></tr>
<tr><td><b>Comments:</b></td><td colspan="2"><b>This command will set the control lines to the specified value, but exported RFUs will not change the frequency or mode being used.</b><br>The two bits representing the selected synth control are saved in bits 1-0 of the second configuration word.</td></tr>
</table>

### 7.6 RF TX ON/OFF

| | |
|---|---|
| **Function:** | Instruction to switch the RF transmitter On/Off. |
| **Function Code** | 0x06 |
| **Parameters** | Parameter 0: 0 or 1 to switch the RF transmitter off or on |
| **Results:** | No results are returned – only ACK message |
| **Description:** | This Instruction sets the RF TX pin. It also logs the setting to the NV configuration area. |
| **Example:** | Set RF TX On/Off<br>To set the RF TX On<br>To controller:<br>Hdr ID len cmd val LRC<br>“ab” “00” “01” “06” “01” “88” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd val<br>“ac” “00” “01” “06” “01” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “06” “26” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 10 or 12 |
| **Encoding:** | “ac000106…” |
| **Operations:** | Controller → reader |

## 7.7 Auxiliary output ON/OFF

| | |
|---|---|
| **Function:** | Instruction to switch the auxiliary output On/Off. |
| **Function Code** | 0x07 |
| **Parameters** | Parameter 0: 0 or 1 to set the output level of the Aux pin |
| **Results:** | No results are returned – only ACK message |
| **Description:** | This Instruction sets the auxiliary output to the state specified in the Instruction. It also logs the setting to the non-volatile configuration area. Since the output is an "open drain" driver, the effect of a "1" or "0" depends on the external circuitry. |
| **Example:** | ***Set Aux On/Off***<br>To set the Aux On<br>To controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "01" "07" "01" "89" <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>"ac" "00" "01" "07" "01" <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "00" "07" "27" <CR><LF><br>In the case of a terminal Instruction with "ac", we have the same reply. |
| **Bytes:** | 10 or 12 |
| **Encoding:** | "ac000107…" |
| **Operations:** | Controller → reader |

## 7.8 Set CRC seed

| | | |
|---|---|---|
| **Function:** | A function to set the seed used for CRC calculations. | |
| **Function Code** | 0x08 | |
| **Parameters** | Parameter 0: CRC high byte<br>Parameter 1: CRC low byte | |
| **Results:** | No results are returned – only ACK message | |
| **Example:** | ***Set CRC seed for P4022***<br>To controller:<br>Hdr ID len cmd hi low LRC<br>“ab” “00” “02” “08” “00” “00” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd hi low<br>“ac” “00” “02” “08” “00” “00” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “08” “28” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. | |
| **Bytes:** | Command: 12 or 14 | Reply: 10 |
| **Encoding:** | “ac0002080000”<br>reply - “ab00000828” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware version of 5.a and higher. | |
| **Comments:** | This command is used to initiate the bool load process.<br>The input parameters:<br>Byte 0: the CRC seed high byte.<br>Byte 1: the CRC seed low byte.<br>For the Xn chips (P4222, P4322 etc) both seed bytes are “ff”, while for the P4022 chips the seed bytes are both “00”. | |

## 7.9 Set Config 3 Options

| | | | |
|---|---|---|---|
| **Function:** | Instruction to set the configuration 3 options. | | |
| **Function Code** | 0x09 | | |
| **Parameters** | Parameter 0: | Bit-mask of configuration options (ID message mode) | |
| | **Bit** | **Item** | **Description** |
| | 2 – 0 | message mode | 000 = normal<br>001 = trigger<br>010 = trigger 2<br>011 = event<br>100 = event 2<br>101 = First/Last seen |
| | 3 | send status as tag | If set, any status message from the decoders will be handled as if it were a special tag and transmitted to the host. |
| | 4 | Sleep | If set then the controller will assert the “Sleep” pin on startup. |
| | 5 | Sleep active low | If set then the sleep pin must be pulled low to cause the RFU to sleep |
| | 6 | Sleep tracks RF Tx | If set then the sleep state tracks RF Tx Off – the RFU will be put into sleep mode whenever the RF power is off. |
| | 7 | Listen while RF off | if set then tags can be read even when the RFU is not transmitting. If not set then tags read while RF Tx is Off will be ignored. |
| | Parameter 1: | Event timeout value (optional) | |
| | Parameter 2: | (Optional) Mask to change only part of CONFIG3 (cf cmd 0x36 §7.44) | |
| **Results:** | No results are returned – only ACK message | | |
| **Example:** | ***Set Normal message mode***<br>To controller:<br>Hdr ID len cmd cfg LRC<br>“ab” “00” “01” “09” “00” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd cfg<br>“ac” “00” “01” “09” “00” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “09” “29” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. | | |
| **Bytes:** | 10 or 12 | | |
| **Encoding:** | “ac000109xx”<br>reply = “ab00000929” | | |

| | |
|---|---|
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware version of 6.3 and higher. The “Sleep” related options are only applicable to handheld devices from version HH 3.a.<br>“Listen while RF Off” options are available from v7.8.<br>The third parameter (the mask) is only available after 8.1, FPGA 10.2 and HH 4.0.<br>Event 2 and First/Last seen modes are available from FPGA version 10.5 and HH 4.1. |
| **Comments:** | In “Normal” message mode, tags Ids are transmitted as fast as the serial communications port will allow. Tag counts will be accumulated in a stack and then messages are sent whenever possible.<br>In Trigger mode, a specified condition “triggers” the reading and transmission of tag codes. Each tag code is transmitted only once during the presence of the trigger condition. When the trigger condition disappears, the system is reset and a subsequent manifestation of the trigger condition will cause the tag code to be transmitted again if it is detected.<br>The single transmission of each tag code is dependant on a stack of IDs which has space for 64 IDs, if more than 64 tags are present during the trigger event then tags codes will be retransmitted as the stack is overwritten. (Some versions of the firmware restrict the stack to 32 tags)<br>Currently the only trigger supported is the logical OR of the software RF Tx state and the TX-switch (an active low input).<br>Trigger mode 1 is the identical to the operation of the Bancolini Handheld reader; Tags are counted and added to the stack whenever they are seen, irrespective of the state of the trigger condition. In the case of the Bancolini Handheld reader this causes a continuous stream of IDs to be sent if the trigger is absent without the RF power being actually switched off.<br>Trigger mode 2 ignores all tags seen while the trigger condition is not present. Tags are only added to the stack while the trigger condition is present.<br>Event mode transmits a tag ID only once each time it enters the beam. After the tag has left the beam for the specified time its ID will again be transmitted to the host when it re-enters the beam. The time is specified in seconds, but because of the testing rate (once per second), this will not be very accurate.<br>Event mode 2 is similar to event mode 1 except that the ID of a tag which remains in the beam is sent every N seconds where N is the timeout period.<br>First/Last seen mode: sends an ID message when the tag is first detected and then does not send any messages until the tag has not been detected for the specified time, at which point, a “last seen” record is sent. The First seen record is flagged by setting the MSB in the TTO “tamper” byte (see § ). The Last seen record is flagged by setting the second most significant bit in the tamper byte. The sending of the TTO information must be enabled for the host to get the F/L seen bits.<br>Readers with firmware v7.e and higher have event mode implemented, but, as with the trigger mode limitation, a tag which remains in the beam permanently will have its ID transmitted whenever the stack wraps (after 32 or 64 IDs have been transmitted to the host).<br>Status messages as tag:<br>This option causes the controller to convert status messages received from the decoders into tag records. The tag ID generated contains the status information:<br>Decoder FW_VERSION, noise count, stateAddr, “00” fields as received from the decoder plus the header byte used on the status message and then padded with “ac”.<br>Sleep<br>This facility was introduced specifically for handheld devices and causes the RFU to enter a power-down mode when “Sleep” is asserted. This function had to be separated from the RF Tx On/Off functionality that is also used for modulation of the carrier for R/W tag |

| | |
|---|---|
| | commanding.<br>Listen when RF Off<br>When enabled it is still possible for a reader to “hear” a tag even when the reader is not transmitting the RF carrier. This ability must be disabled (by default) to prevent a reader that is off from trying to command a R/W tag (in a multiplexed portal scenario). |

**Note 1. The “Trigger”, “Event” and “First/Last seen” modes use a buffer of limited size to record the fact that a tag has been seen and its record sent to the host: this means that if more than the allowed number of tags is seen while the first tag remains in the beam, then the buffer will overflow and the fact that the first tag had been seen will be overwritten and the tag’s ID will be resent.**

**Only the “head” of this buffer is checked for expiry so another anomaly is possible: If tag 1 stays in the beam forever then other tags that are seen and then leave the beam will never expire and so their IDs will never be resent (until the tag which is blocking the queue expires).**

**A tag blocking the queue will also cause Last-seen records of other tags to be delayed – giving incorrect times for them being in the beam ( remember that the time is not updated in the ‘S’ list, so the time reported is the first-seen time). Also if a blocked tag returns to the beam its first/last seen records will be reported in the incorrect order. It has been suggested that this mode be referred to as “Single tag First-seen Last-seen” mode.**

**Because of the resolution of the countdown timer and the asynchronous arrival of tag IDs, there is another anomaly where a timeout of one (1s) results in the tag’s ID being sent each second even in Event mode and F/L seen mode.**

## 7.10 Get Statistics

| | |
|---|---|
| **Function:** | A function to retrieve various reader hardware and software diagnostics. |
| **Function Code** | 0x0a |
| **Parameters:** | No parameters are required |
| **Results:** | Parameter 0: Firmware version (see notes 1 & 2)<br>Parameter 1: Reader ID<br>Parameter 2: Configuration word as described in par. 7.3 above.<br>Parameter 3: CRC count<br>Parameter 4: Power up count<br>Parameter 5: Activity count (Measure of noise on decoder input)<br>Parameter 6: Decoder I channel FW version (see notes 1 & 2)<br>Parameter 7: Decoder Q channel FW version (see notes 1 & 2)<br>Parameter 8: Second Configuration word as described in par. 5.2 above.<br>Parameter 9: Wiegand Configuration word as described in par. 7.14 below.<br>Parameter 10: Timer value for test Wiegand transmissions as described in § 7.14 below.<br>Parameter 11: Third Configuration word as described in par. 7.9 above.<br>Parameter 12: Hardware code (after v8.0PIC, v4.0HH, V10.0 FPGA)<br>Parameter 13: Number of IDs rejected by selection filter (since 2005-05-17) |
| **Example:** | ***Get Statistics***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “0a” “51” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “00” “0a” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “06” “0a” “05” “00” “d1” “3f” “00” “00” “0a” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8 or 10 |
| **Encoding:** | “ac00010a” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in all versions of the controller, but the returned data message varies between versions. |
| **Comments:** | The payload of the reply is augmented when more parameters are needed to configure the controller firmware as its functionality is enhanced. The status reflects this configuration and so, of necessity, will change as enhancements are added to the controller firmware. Parameters will only be added to the end of the message, making the message backwards compatible.<br>Note 1: The version number is encoded as a decimal.hex pair of nibbles – where the first nibble is the major version number as a decimal number (0 – 15) and the second nibble is the minor version number as a hex digit (0 – f). |

In versions prior to version 4.9, the version number was a decimal value with an implied decimal point before the last digit (allowing for values 0.0 – 25.5)

Note 2: A controller version number of 0xff (15.f) is used as a flag to signal that the reader is a single device with controller and decoder combined. In this case, the controller equivalent version is returned in parameter 6 (I channel version) and the decoder equivalent version is returned in parameter 7 (Q channel version).

Note 3: The Hardware code is a hex value assigned sequentially to each generation of hardware in each major hardware type. The first hex digit is used to specify the type, while the second hex digit specifies the version. When implemented (2004-09-09), space was left for historic hardware versions still in service. Currently allocated types are:

00-0f: Handheld (single PIC) devices

10-1f: PIC decoder based readers

20-2f: FPGA decoder based readers.

Currently allocated codes are:

| | |
|---|---|
| 05 | HH 1E hardware |
| 12 | rev 2A of board with dual 16F628 decoder PICs. 18C452 in PLCC socket |
| 13 | rev 2A of board with dual 16F628 decoder PICs. 18F452 in PLCC socket |
| 18 | Rev 1C of “256k” board with dual 18F452 decoder PICs. |
| 22 | Altera FPGA decoder (FPGADEC MHC0104) |
| 23 | Actel FPGA decoder (ACTELDEC1A MHC0404) |
| 24 | Actel FPGA decoder with FPGA RtoT (ACTELDEC1B MHC 0406) with piggy-back components. |
| 25 | Actel FPGA rev 1C |
| 26 | Actel FPGA rev 1D |

Note 4: The reject count includes tags rejected because of either matching the reject pattern or not matching the select pattern. The count is limited to 255 and is cleared when this message is transmitted.

## 7.11 Run Selftest (Future implementation)

| | |
|---|---|
| **Function:** | A function to force the controller to run internal diagnostics and return the test results |
| **Function Code** | 0x0b |
| **Parameters:** | No parameters are required |
| **Results:** | Parm 0, bits<br>Ls bit 0 – RTC clock 0 = fail<br>Bit 1 – RTC mem 0 = fail<br>Bit 2 - Config checksum 0 = fail |
| **Example:** | ***Run Selftest routine***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “0b” “52” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“ac” “00” “00” “0b” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “01” “0b” “rr” “xx” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8 or 10 |
| **Encoding:** | “ac00000b” |
| **Operations:** | Controller → reader |
| **Comments:** | This command is not yet implemented |

## 7.12 Set CRC Checking Options

| | | | |
|---|---|---|---|
| **Function:** | Instruction to set the CRC checking options. | | |
| **Function Code** | 0x0d | | |
| **Parameters** | Parameter 0: Bit-mask of CRC checking options | | |
| | **Bit** | **Item** | **Description** |
| | 0 | Ignore CRC | 0 = OFF (do not ignore CRC failures), 1 = ON (check and count CRC errors, but count tag as valid). |
| | 1 | Check the CRC using both standard seeds | 0 – Off – only use the selected seed, 1- On – if the normal seed fails then try its complement (works for seeds 0x0000 and 0xffff) |
| | 2 | Replace the top 32 bits of the code with expected EmM values and check CRC | 0 – Off – don't try fancy stuff, 1- On – replace top 32 bits with the expected value (0x05800000) |
| | 3 | All 1s | Pass tags where all bits of the code and CRC are set (an unprogrammed tag) |
| | 4 | mask first byte | zero the first byte of the |
| | 5 | All 0s | Normally a tag ID consisting of only zeros will be discarded even though the CRC (P4022) passes. If this option is set then these (all zero) IDs are transmitted to the host. |
| | 6 | CC ITT6 | Accept a CCITT 16 CRC (only FPGA decoder) |
| | 7 | X6X7 | Kludge to accept incorrectly programmed X6, X6 and X7 tags where the ID must be shifted right by one bit before testing the (unshifted) CRC (v7.6 returns the shifted ID while >= v7.7 returns the correct ID. |
| **Results:** | No results are returned – only ACK message | | |
| **Example:** | ***Set CRC checking to try both CRC flavours and, if both fail to try setting the high word to 0x05800000***<br>To controller:<br>Hdr ID len cmd cfg LRC<br>"ab" "00" "01" "0d" "06" "xx" <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd cfg<br>"ac" "00" "01" "0d" "06" <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "00" "0d" "xx" <CR><LF><br>In the case of a terminal Instruction with "ac", we have the same reply. | | |
| **Bytes:** | 10 or 12 | | |

<table>
<tr><td><b>Encoding:</b></td><td>“ac00010dxx”<br>reply = “ab00000dxx”</td></tr>
<tr><td><b>Operations:</b></td><td>Controller → reader</td></tr>
<tr><td><b>Availability:</b></td><td>This command is available in controllers with firmware version of 5.a and higher. The bit 2 option is only available on version 5.a (a limited release version).</td></tr>
<tr><td><b>Comments:</b></td><td>

This command is used to enable the reading of P4022 tags as well as X1 to X3 tags (P4122, P4222, P4322).
The input parameters:

Byte 0: is a bitmask where each of the low order bits enables one of the CRC handling options on the reader.
Bit 0 (lowest order bit) – if set, causes incorrect CRCs to be ignored and handled a is the CRC check succeeded (except that the CRC fail count is incremented).
Bit 1 – if set, causes the alternative CRC seed (1’s complement of the current seed) to be tried if the CRC fails using the current seed. This allows both tag types to be read by the same reader – note that since the reader strips off the CRC when transmitting tag IDs, regeneration of the CRC is no longer straightforward.
Bit 2 – if set, causes a last resort attempt at decoding to be attempted. The high order word is assumed to be correct and contain the binary value 0x05800000. If the CRC now succeeds (only using the current seed) then the tag is recorded as good. This option should only be used on MPW sample devices. From version 7.0 of the controller, both seeds will be tried if that option is enabled.

The sequence of tests is as follows:

```
If bit 5 is set and the tag ID consists of all 0s – PASS
If bit 3 is set and the tag ID consists of all 1s – PASS
Then the currently selected seed is used to test the CRC
– if it passes no further processing is required – PASS
– if it fails then
  - if bit 1 is set
    - check using complement of current seed
    - if the CRC now passes no further processing – PASS
    - if it fails then
      - if bit 2 is set
        - preset high-order word and test CRC using selected seed
          - if it passes – done
          - if it fails then
            - if bit 1 is set
              - check using complement of the current seed
              - if it passes - done
              - if it fails
                if bit 4 is set
                - clear the leading byte of the ID and try both
                  seeds
                - if it passes – done
                - if bit 0 is set
                  - PASS
    else FAIL
```

</td></tr>
<tr><td></td><td></td></tr>
</table>

## 7.13 Set Aux 1 Pulse Options

<table>
<tr><td><b>Function:</b></td><td colspan="3">Instruction to set the Aux 1 pulse options.</td></tr>
<tr><td><b>Function Code</b></td><td colspan="3">0x0e</td></tr>
<tr><td><b>Parameters</b></td><td colspan="3">Parameter 0: Bit-mask of Aux 1 pulse options (forms part of config word 2)</td></tr>
<tr><td></td><td><b>Bit</b></td><td><b>Item</b></td><td><b>Description</b></td></tr>
<tr><td></td><td>7</td><td>Retrigger</td><td>0 = OFF (do not retrigger the pulse), 1 = ON (retrigger the pulse whenever the conditions specified in bits 6 and 5 are met).</td></tr>
<tr><td></td><td>6</td><td>Pulse when a tag is seen</td><td>0 – Off – don’t pulse when a tag is seen,<br>1 – On – pulse when a tag is detected.</td></tr>
<tr><td></td><td>5</td><td>Pulse when tag ID is sent to host</td><td>0 – Off – don’t pulse when a tag ID is sent to the host,<br>1 – On – pulse whenever a tag ID is transmitted to the host.</td></tr>
<tr><td></td><td colspan="3">NOTE: If all three bits are set, then the reader sets AUX1 whenever a tag is seen. AUX1 is reset when the “Nurse” tag is seen. This is known as “Hospital” mode. AUX1 is connected to a buzzer which sounds when any tag (a baby) is seen by the reader. A nurse has to reset this alarm state using a special tag (Nurse tag).</td></tr>
<tr><td></td><td colspan="3">Parameter 1: The pulse width (in 10ms units) – ignored in hospital mode.</td></tr>
<tr><td><b>Results:</b></td><td colspan="3">No results are returned – only ACK message</td></tr>
<tr><td><b>Example:</b></td><td colspan="3"><b><i>Set the AUX1 pin to be pulsed for 30ms whenever a tag is detected</i></b><br>To controller:<br>Hdr ID len cmd cfg width LRC<br>“ab” “00” “02” “0e” “40” “03” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd cfg width<br>“ac” “00” “02” “0e” “40” “03” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “0e” “55” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.</td></tr>
<tr><td><b>Bytes:</b></td><td colspan="3">12 or 14</td></tr>
<tr><td><b>Encoding:</b></td><td colspan="3">“ac00020exxyy”<br>reply = “ab00000e55”</td></tr>
<tr><td><b>Operations:</b></td><td colspan="3">Controller → reader</td></tr>
<tr><td><b>Availability:</b></td><td colspan="3">This command is available in controllers with firmware versions of 6.1 and higher.</td></tr>
<tr><td><b>Comments:</b></td><td colspan="3">The IO configuration settings will also influence the behaviour of the reader and may</td></tr>
</table>

| | |
|---|---|
| | disable the control of the Aux1 pin (referred to as “Output 2” from 2003-06-13).<br>**THE PULSE-WIDTH IS NOT SAVED IN NON-VOLATILE (NV) MEMORY.**<br>**Controller versions since 2006/12/24 do save the pulse width in NV memory.** |

### 7.14 Set Wiegand Output Options

<table>
<tr><td><b>Function:</b></td><td colspan="3">Instruction to set the Wiegand Output options.</td></tr>
<tr><td><b>Function Code</b></td><td colspan="3">0x0f</td></tr>
<tr><td><b>Parameters</b></td><td colspan="3">Parameter 0: Bit-mask of Wiegand options</td></tr>
<tr><td></td><td><b>Bit</b></td><td><b>Item</b></td><td><b>Description</b></td></tr>
<tr><td></td><td>7</td><td>Enable Wiegand</td><td>0 = OFF (do not send wiegand format data), 1 = ON (send wiegand format data on port B<5> = D0, B<6> = D1).</td></tr>
<tr><td></td><td>6</td><td>Send a dummy tag ID</td><td>0 – Off – donothing,<br>1 – On – send a dummy tag ID (cleared after sending).</td></tr>
<tr><td></td><td>5-3</td><td>Pulse Interval time</td><td>000 =0.3ms, 001 = 0.6ms, 010 = 0.9ms,<br>011 =1.2ms, 100 = 1.5ms, 101 = 1.8ms,<br>110 =2.1ms, 111 =2.4ms (nominal)</td></tr>
<tr><td></td><td>2-1</td><td>Pulse width</td><td>00 = 50µs, 01 = 75µs, 10 = 100µs, 11 = 125µs (nominal)</td></tr>
<tr><td></td><td>0</td><td>26/34 bit code</td><td>1 = 26 bit code (24 bit ID + 2 parity bits)<br>0 = 34 bit code (32 bit ID + 2 parity bits)</td></tr>
<tr><td></td><td colspan="3">Parameter 1: Transmit interval (in 10ms units) (this parameter is optional).</td></tr>
<tr><td></td><td colspan="3">Parameter 2: A repeat interval (in seconds) for transmitting the dummy tag ID (this parameter is optional).</td></tr>
<tr><td><b>Results:</b></td><td colspan="3">No results are returned – only ACK message</td></tr>
<tr><td><b>Example:</b></td><td colspan="3"><b><i>Set the reader to transmit 34bit Wiegand using a 25µs pulse width and a 700µs pulse interval</i></b><br>To controller:<br>Hdr ID len cmd cfg LRC<br>“ab” “00” “01” “0f” “a8” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd cfg<br>“a<b>c</b>” “00” “01” “0f” “a8” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “0f” “” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.</td></tr>
<tr><td><b>Bytes:</b></td><td colspan="3">12 / 14, 13 / 15 or 14 / 16</td></tr>
<tr><td><b>Encoding:</b></td><td colspan="3">“ac00010fxx” or “ac00020fxxnn”<br>reply = “ab00000fxx”</td></tr>
<tr><td><b>Operations:</b></td><td colspan="3">Controller → reader</td></tr>
</table>

| | |
|---|---|
| **Availability:** | This command is available in controllers with firmware versions of 6.2 and higher. Normally the ICSP header is used for the Wiegand signals, but versions of the firmware may be built that use AUX1 and TxStatus (pins which are available on the RFU connector), in this case a “W” will be included in the startup message transmitted on booting. |
| **Comments:** | This is the first command to introduce an optional parameter – the transmit interval and dummy tag repeat interval (parameters 1 and 2). The decoder uses the length (characters 5 & 6) to determine if a transmit interval and a repeat interval are specified, else the existing value of these parameters is not changed.<br>The pulse width and interval parameters need no further explanation, nor does the enable bit (bit 7).<br>The 26/34-bit selection bit (bit 0) determines whether 24 or 32 bits of ID are transmitted.<br>For a 24 bit ID, bits 24 to 47 of the IPICO Tag ID are used (least significant 24 bits excluding the CRC).<br>For a 32 bit ID, bits 16 to 47 of the IPICO Tag ID are used (least significant 32 bits excluding the CRC).<br>The transmit-interval is the shortest interval between wiegand messages.<br>The transmission of a dummy tag ID is controlled by bit 6 of the configuration byte and also by the optional repeat interval.<br>If the repeat interval is zero (0) the setting bit 6 will cause the dummy tag ID to be transmitted once and then the bit will be cleared.<br>If the repeat interval is non-zero then the firmware will set bit 6 at the specified rate causing the dummy tag ID to be transmitted.<br>The dummy tag ID used is A1B2C3D4<br>**Query current settings**<br>Versions of the controller firmware since 8.1 (FPGA 10.2 and HH 4.0) implement a query of the current Wiegand settings by specifying a length of 0xff (without any parameters). The message returned by the controller has the same format as the set Wiegand options command. |

## 7.15 Set Tag ID Message Contents

| | |
|---|---|
| **Function:** | Instruction to set the message format. |
| **Function Code** | 0x11 |
| **Description** | The items to be sent in a tag ID message can be specified, but not the order in which they are transmitted. |
| **Parameters** | Up to nine (9) parameters can be specified. All parameters except the first are optional, but parameters may not be skipped i.e. if parameter 3 is to be specified then parameter 2 must also be specified with the correct desired value. |
| | Parameter 0: Bit-mask of information fields to be transmitted. |

| Bit | Item | Description |
|---|---|---|
| 0 (LSB) | Reader ID | If set, the reader ID will be transmitted before the tag ID (field 1). |
| 1 | I count | If set, the count of times the tag was seen on the I-channel is transmitted after the tag ID (field 3). |
| 2 | Q count | If set, the count of times the tag was seen on the Q-channel is transmitted after the I-count (field 4). |
| 3 | Date | If set, the date, in the format “yymmdd”, will be transmitted after the count values if any (field 5). |
| 4 | Time | If set, the time, in the format “hhmmss”, will be transmitted after the date (field 6). |
| 5 | 10ms count | If set, the 10ms counter (part of the complete time information), in the hexadecimal format “xx”, will be transmitted after the time (field 7) |
| 6 | LRC | If set, the LRC, as two hexadecimal digits “xx”, will be transmitted after the 10ms count (field 8) |
| 7 | TTO | If set, then TTO index, page and tamper bytes are included in the ID message (field before LRC) |

Parameter 1: Bit-mask of the bytes of the ID to be transmitted. Field 2 of the message is generated from selected bytes of the tag ID. The standard format includes bytes 0-5 of the tag ID (all except the CRC bytes).

| Bit | Item | Description |
|---|---|---|
| 7 (MSB) | Byte 0 (MS) of ID | If set, the first (most significant) byte of the tag ID will be transmitted. |
| 6 – 2 | corresponding byte of ID | each byte can be included or excluded from the message a desired |
| 1 – 0 | CRC bytes | The CRC bytes are treated as normal ID bytes and can also be included or excluded as desired. |

| | |
|---|---|
| Parameter 2: | Header byte 1 for ASCII messages. The byte specified here, as a two character hexadecimal number, is used as the first character of the ASCII format tag ID message. The standard message uses the character ‘a’, so this parameter would be specified as “61” (the hexadecimal ASCII value for ‘a’). If zero “00” is specified then no character is transmitted. |

| | | |
|---|---|---|
| | Parameter 3: | Header byte 2 for ASCII messages. The second character of the optional two byte header used on ASCII tag ID messages. The value is specified as with header byte 1. The default value is ‘a’ (“61”). |
| | Parameter 4: | Header byte 1 for binary messages. The standard binary message uses the header byte 0xaa. This can be changed by specifying another hexadecimal value (except zero), or omitted by specifying a value of zero. |
| | Parameter 5: | Header byte 2 for binary messages. The standard binary message does not contain a second header byte, so by default this parameter is zero. |
| | Parameter 6: | Trailer byte 1 for both binary and ASCII messages. The same trailing sequence is used for both message types, with the default being the carriage-return – line-feed pair (<CR><LF> 0x0d 0x0a). |
| | Parameter 7: | Trailer byte 2 |
| | Parameter 8: | Separator byte. If not zero, this character will be inserted into the message stream after any included field (except the LRC). This option is available in controller versions C6.4 and higher. |
| **Results:** | No results are returned – only ACK message | |
| **Example 1:** | ***Set the standard ASCII and binary message formats***<br>To controller:<br>Hdr ID len cmd fld byte h1a h2a h1b h2b t1 t2 LRC<br>“ab” “00” “08” “11” “7f” “fc” “61” “61” “aa” “00” “0d” “0a” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd fld byte h1a h2a h1b h2b t1 t2<br>“ac” “00” “08” “11” “7f” “fc” “61” “61” “aa” “00” “0d” “0a” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “11” “xx” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. | |
| **Example 2:** | ***Set the format to that used by the Bancolini Handheld reader***<br>No header, no trailer and no additional data – only tag ID, but the complete tag ID is transmitted (including CRC bytes). The trailer bytes are used to delimit the tag IDs.<br>Hdr ID len cmd fld byte h1a h2a h1b h2b t1 t2<br>“ac” “00” “08” “11” “00” “fc” “00” “00” “00” “00” “0d” “0a” <CR><LF> | |
| **Example 3:** | ***Send the standard fields but include the tag’s CRC as part of the ID – leave other options unchanged.***<br>Hdr ID len cmd fld byte<br>“ac” “00” “02” “11” “7f” “ff” <CR><LF> | |
| **Bytes:** | From 12 / 14 to 28 / 30 | |
| **Encoding:** | “ac00081100fc000000000d0a”<br>reply = “ab000011xx” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 6.3 and higher. The separator option was included in version 6.4<br>The TTO fields are only available on FPGA decoders with versions 10.3 and higher | |

| | |
|---|---|
| **Comments:** | The TTO fields are:<br>byte 1: index – a value used to link a tag's ID to its data pages (a data page belongs to the most recent tag whose ID message had the same index).<br>byte 2: page number – the page number (0 = tag ID)<br>byte 3: tamper flag – 0 = no tamper, ff = tag has been tampered (pre 2005-12-07).<br>Since 2005-12-07 (FPGA 10.5, HH 4.1) the tamper byte has been used for flagging First-seen and Last-seen records. From these versions the tamper flag is only the LSB (bit-0) of the byte, the first-seen flag is the MSB (bit-7) and the last-seen flag is bit-6 of the tamper byte. User software should now not only test for 0xff as the tamper flag, it is suggested that, to maintain compatibility with readers already in the field, one tests for a value of 0xff and if a match is found signal tamper only, else each bit should be tested and the corresponding condition signalled if set.<br>Separator:<br>The separator is inserted after a field has been transmitted and is not included if the field is omitted. If the LRC is not included in the message then the message will have a trailing separator.<br>**THE SEPARATOR IS NOT ACCUMULATED INTO THE LRC.**<br>**Query current format**<br>Versions of the controller firmware since 8.1 (FPGA 10.2 and HH 4.0) implement a query of the current message format by specifying a length of 0xff (without any parameters). The message returned by the controller has the same format as the set message command. |

### 7.16 Set tag baud rate – for use with single PIC readers (Handheld and FPGA based decoders)

<table>
<tr><td><b>Function:</b></td><td colspan="3">Sets the baud rate of tags to be decoded.</td></tr>
<tr><td><b>Function Code</b></td><td colspan="3">0x12</td></tr>
<tr><td><b>Description:</b></td><td colspan="3">The tag baud rate is selected.<br>From version 1.3 of the handheld (HH) firmware, the baud rate of tags to be decoded is no longer hard coded into the firmware, but can be changed during operation.<br>See EEPROM life warning</td></tr>
<tr><td><b>Parameters</b></td><td colspan="3">Parameter 0: Bit-mask configuration options</td></tr>
<tr><td></td><td><b>Bit</b></td><td><b>Item</b></td><td><b>Description</b></td></tr>
<tr><td></td><td>7-2</td><td>Decoder parameters</td><td>ignored (see §7.49)</td></tr>
<tr><td></td><td>1-0</td><td>Baud rate</td><td>0 – 32kb/s (Not yet supported)<br>1 – 64kb/s<br>10 – 128kb/s (Not yet supported)<br>11 – 256kb/s</td></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><td><b>Results:</b></td><td colspan="3">No results are returned – only ACK message</td></tr>
<tr><td><b>Example:</b></td><td colspan="3"><b><i>setting the tag baudrate to 256k</i></b><br>To controller:<br>Hdr ID len cmd val lrc<br>“ab” “00” ”01” “12” “03” “87” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd lrc<br>“ab” “00” “00” “12” “</td></tr>
<tr><td><b>Bytes:</b></td><td colspan="3">0</td></tr>
<tr><td><b>Encoding:</b></td><td colspan="3">“ab0001120387”<br>reply – “ab000012xx”</td></tr>
<tr><td><b>Operations:</b></td><td colspan="3">Controller → reader</td></tr>
<tr><td><b>Availability:</b></td><td colspan="3">This command is available in handheld readers with firmware version of 1.3 and higher and FPGA based decoders.</td></tr>
<tr><td><b>Comments:</b></td><td colspan="3">This command is required by HH and FPGA based readers.<br>Legacy (3 PIC) readers will issue a “Bad command” error message).<br>Sending a length of 0xff (but without and data bytes) will cause the reader to reply with the current value of the decoder configuration word.<br>Firmware versions before 2007/09/07 will affect bits 7-2 when this command is sent (i.e. have the same effect as command 0x3b (see §7.49).</td></tr>
</table>

### *7.17 Set “Nurse” tag code*

| | |
|---|---|
| **Function:** | Instruction to set the “Nurse” tag ID. |
| **Function Code** | 0x13 |
| **Parameters** | Parameter 0-7: The tag ID (including CRC) of the “Nurse” tag used to stop the beeping when in “Hospital” mode (see command 0x0e) above |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Set Nurse tag to 0000000012cc6e70***<br>To controller:<br>Hdr ID len cmd … tab ID … LRC<br>“ab” “00” “08” “13” “00” “00” “00” “00” “00” “00” “12” “cc” “6e” “70” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd cfg width<br>“a**c**” “00” “08” “13” “00” “00” “00” “00” “00” “00” “12” “cc” “6e” “70” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “13” “24” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 28 or 30 |
| **Encoding:** | “ac00081300000000012cc6E70”<br>reply = “ab00001324” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.0 and higher. |
| **Comments:** | |

### 7.18 Get “Nurse” tag code

| | |
|---|---|
| **Function:** | Instruction query the current “Nurse” tag ID. |
| **Function Code** | 0x14 |
| **Parameters** | none |
| **Results:** | Parameter 0-7: The tag ID (including CRC) of the current “Nurse” tag used to stop the beeping when in “Hospital” mode (see command 0x0e) above |
| **Example:** | ***Get Nurse tag***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “14” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “00” “14 <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len … tag ID … LRC<br>“ab” “00” “08” “14” “00” “00” “00” “00” “12” “cc” “6e” “70” “d8” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 28 or 30 |
| **Encoding:** | “ac000014”<br>reply = “ab000814000000012cc6e70d8” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.0 and higher. |
| **Comments:** | |

## 7.19 Set RW data

| | |
|---|---|
| **Function:** | Sets the command or data word that will be transmitted to a read-write tag when the next “Send RW” command is issued. |
| **Function Code** | 0x15 |
| **Parameters:** | Parameter 0: selector between command and data word.<br>A value of 0 indicates that the payload is the command to be issued next.<br>A value of 1 indicates that the payload is the data to be transmitted after the command (as for a write operation).<br>The protocol makes provision for multiple data words to be downloaded (by using values greater than 1 here), but since the current tag protocol does not allow multi-page writes, this feature is not implemented in the reader.<br>The format of an RW command is detailed in the P40x4 datasheet “p40x4data_v5.doc” (1)<br>Parameters 1-8: the data payload to be used as specified by parameter 0. |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Set RW Command***<br>To controller:<br>Hdr ID len cmd page RW cmd - target tag - LRC<br>“ab” “00” “09” “15” “00” “a0” “0b” “ff” “ff” “ff” “ff” “ff” “ff” “7a” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd page RW cmd - target tag -<br>“ac” “00” “09” “15” “00” “a0” “0b” “ff” “ff” “ff” “ff” “ff” “ff” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “15” “26” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 26 or 28 |
| **Encoding:** | “ab00091500a00bffffffffffff7a” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.1 and higher. |
| **Comments:** | This command is part of the “first generation” R/W command set, and is used to load the command or data to be sent by the next “Transmit” command. It is used by ShowTags for “manual” testing (where all checking of success or failure is done by the host application). |

## 7.20 Transmit preloaded RW Command

| | |
|---|---|
| **Function:** | Causes the reader to transmit the preloaded command (and data in the case of a write command). |
| **Function Code** | 0x16 |
| **Parameters:** | No parameters are required |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Transmit To RW tag***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “16” “27” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “00” “16” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “16” “27” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8 or 10 |
| **Encoding:** | “ab00001627” |
| **Operations:** | Controller → reader → tag |
| **Availability:** | This command is available in controllers with firmware versions of 7.1 and higher. |
| **Comments:** | This command is part of the “first generation” R/W command set, and is used to transmit the previously loaded command or data. It is used by ShowTags for “manual” testing (where all checking of success or failure is done by the host application). |

## *7.21 Set RW Transmission Bit-rate (R->T) (modulation depth in future?)*

| | |
|---|---|
| **Function:** | Sets the bit-rate to be used by the reader for RF communication with the tag(s). |
| **Function Code** | 0x17 |
| **Parameters:** | Parameter 0:<br>A value that is used to determine the period of the modulated signal. A value of 150 translates to 4kbps on the current readers with a 40MHz clock frequency. |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Set R->T bit-rate to 4kbps***<br>To controller:<br>Hdr ID len cmd per’d LRC<br>“ab” “00” “01” “17” “96” “98” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd per’d<br>“a**c**” “00” “01” “17” “96” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “17” “LL <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 10 or 12 |
| **Encoding:** | “ab0001179698” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.1 and higher. |
| **Comments:** | |

### *7.22 Reset factory defaults*

| | |
|---|---|
| **Function:** | Reset the reader configuration to the factory default settings. Settings modified by the user, and stored in NV memory, are overwritten by the factory default settings (those reflected by the power-up message). |
| **Function Code** | 0x18 |
| **Parameters:** | An optional parameter can be specified, which, if non-zero, will force the reader to save the factory defaults to NV memory. |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Get Statistics***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “18” “LL” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “00” “18” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “18” “LL” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8/10 or 10/12 |
| **Encoding:** | “ac000018” or for NV reset “ac00011801” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.2 and higher. |
| **Comments:** | |

## 7.23 Configure the IO pins on the Reader

<table>
<tr><td><b>Function:</b></td><td colspan="3">Configure the IO pins on the reader.  The single input pin and three output pins can be programmed to perform specific functions.<br>Using a length of 0xff without any parameters will cause the controller to return the current configuration (duplicating the functionality of command 0x1b – see 7.25 below).</td></tr>
<tr><td><b>Function Code</b></td><td colspan="3">0x19</td></tr>
<tr><td rowspan="12"><b>Parameters:</b></td><td colspan="3">Parameter 0:  A bit-mask for controlling some of the reader IO pins.</td></tr>
<tr><td>Bit</td><td>Function</td><td>Description</td></tr>
<tr><td>0</td><td>In1 is Tx Switch</td><td>If 1, then the Input 1 line is used to perform the RF TX switch function.  Whether the RF carrier is transmitted or not is an “OR” function between this input and the state as set by the host using the RF Tx On/Off command described in 7.6 above.<br>If 0, then the state of this input is not used to control the RF carrier.<br>In either case the level of this input can be determined by the host using the Get IO command described in 0 above.</td></tr>
<tr><td>1</td><td>In 1 is active low</td><td>This bit controls the polarity of the “Input 1” signal.<br>If this bit is set then a low input on the pin will be interpreted as “on” and, for example, turn on the RF carrier when it is being used for this purpose.</td></tr>
<tr><td>2</td><td>Out 1 is TxStat</td><td>This bit controls whether the state of the RF carrier is reflected on the Output 1 pin.<br>If set, then the “Output 1” pin will be set when the RF carrier is being transmitted.<br>If the bit is clear then the pin will not track the RF carrier state.<br><b>NOTE: Under normal operating conditions, bits 2 and 3 should not both be set.</b></td></tr>
<tr><td>3</td><td>Out 1 is W1</td><td>This bit controls whether the “Output 1” bit is used as the Wiegand D1 data line.<br><b>NOTE: Under normal operating conditions, bits 2 and 3 should not both be set.</b></td></tr>
<tr><td>4</td><td>Out 1 is active low</td><td>The sense of this “Output 1” pin is controlled by this bit.<br>If this bit is set then a logical “1” will be represented by a low level on the Output 1 pin.</td></tr>
<tr><td>5</td><td>Out 2 is Aux1</td><td>This bit controls whether the Output 2 pin is used for the Aux1 signal, used on the Dual frequency unit as a buzzer output.</td></tr>
<tr><td>6</td><td>Out 2 is W0</td><td>This bit controls whether the Output 2 pin is used for the Wiegand D0 signal.</td></tr>
<tr><td>7</td><td>Out 2 is active low</td><td>This bit controls the sense of the Output 2 pin.  If set then a logical “1” will be represented by a low signal on the Output 2 pin.</td></tr>
<tr><td colspan="3">Parameter 1: A bit-mask for controlling some of the reader IO pins</td></tr>
<tr><td>Bit</td><td>Function</td><td>Description</td></tr>
</table>

| | | | |
|---|---|---|---|
| | 0 | Out 3 is health | This bit controls whether the Output 3 pin is used to display the “Heartbeat” on the reader.<br>If set then the normal health state of the reader is displayed on the Output 3 pin in the form of a heartbeat which flashes an LED.<br>If this bit is not set then the heartbeat will not be change the Output 3 pin. |
| | 1 | Out 3 is valid ID | This bit controls whether the Output 3 pin is flashed whenever a message is transmitted on the reader’s serial output port.<br>If it is set then the transmission of a valid tag ID will cause the Output 3 pin to be flashed. Replies to host commands and now unsolicited status and IO messages will also flash the pin.<br>If clear then communication messages will not be signalled on the Output 3 pin. |
| | 2 | Out 3 is active low | This bit controls the sense of the Output 3 pin.<br>If set then a logical “1” will be represented by a low signal on the output pin. |
| | 3 | unused | |
| | 4 | Send In1 0-1 message | This bit controls whether a transition from 0 to 1 on the Input 1 pin will cause an unsolicited “Get IO” message to be sent to the host.<br>If set then a 0-1 transition on the Input 1 pin will cause the reader to send the same message as it would have sent in reply to a “Get IO” command described in 7.24 below. |
| | 5 | Send In1 1-0 message | This bit controls whether a transition from 1 to 0 on the Input 1 pin will cause an unsolicited “Get IO” message to be sent to the host.<br>If set then a 1-0 transition on the Input 1 pin will cause the reader to send the same message as it would have sent in reply to a “Get IO” command described in 7.24 below. |
| | 6 | unused | NOTE: in version 7.3 of the reader these bits are used internally by the reader and a change to the protocol must be accompanied by a corresponding change to the reader FW. |
| | 7 | unused | |
| | Parameter 2: A bit-mask for controlling more of the reader IO pins | | |
| | 0 | HH Wiegand signals active low | In a Handheld controller with Wiegand firmware, both Wiegand signals are active low. |
| | 1 | Imput2 is active low | The Input2 signal is active low. |
| | 2 | Beep Override | If set then user control of Output 2 will override its use to generate a beep pulse on detecting a tag. |
| | 3 | Output1 tracks Input2 | When set the Output1 signal will track the logic level of the Input2 signal. |
| | 4 | Output2 tracks Input2 | When set the Output2 signal will track the logic level of the Input2 signal. |
| | 5 | Ouput3 tracks Input2 | When set the Output3 signal will track the logic level of the Input2 signal. |
| | 6 | unused | |
| | 7 | INPUT2_LOGIC | Not user modifiable – holds the logic value of the Input2 signal. |
| **Results:** | No results are returned – only ACK message.<br>When a “get” operation is performed (by using a length of 0xff) the returned message has | | |

| | |
|---|---|
| | the same format as the “set” command. Also see 7.25 below. |
| **Example:** | ***Set IO Configuration to Factory defaults***<br>The command detailed here will set the following behaviour (which is the factory default):<br>Input 1 is Tx Switch<br>Input 1 is active low<br>Output 1 is Tx state<br>Output 1 is not active low<br>Output 2 is used for the Aux1 function<br>Output 2 is not active low<br>Output 3 is health indicator<br>Output 3 is valid ID indicator<br>Output 3 is not active low<br>No unsolicited messages are sent on IO state change.<br>To controller:<br>Hdr ID len cmd P0 P1 LRC<br>“ab” “00” “02” “19” “27” “03” “f8” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd P0 P1<br>“a**c**” “00” “02” “19” “27” “03” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “19” “05” “LL” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 12 or 14 |
| **Encoding:** | “ab0002192703f8” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.2 and higher.<br>The “get” variant is available on versions 7.e and higher. |
| **Comments:** | |

### 7.24 Get IO

<table>
<tr><td><b>Function:</b></td><td colspan="3">Gets the state of the IO pins.</td></tr>
<tr><td><b>Function Code</b></td><td colspan="3">0x1a</td></tr>
<tr><td><b>Parameters:</b></td><td colspan="3">No parameters are required</td></tr>
<tr><td rowspan="5"><b>Results:</b></td><td colspan="3">Parameter 0: The state of some IO pins</td></tr>
<tr><td>Bit</td><td>Pin</td><td>Description</td></tr>
<tr><td>0</td><td>Input 1</td><td>This bit reflects the logical state of the Input 1 pin</td></tr>
<tr><td>1</td><td>RF Tx is on</td><td>This bit reflects the logical state of the RF carrier, if set the RF carrier is being transmitted.</td></tr>
<tr><td>2-7</td><td>unused</td><td></td></tr>
<tr><td><b>Example:</b></td><td colspan="3"><b><i>Get IO</i></b><br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “1a” “LL” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“ac” “00” “00” “1a” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “01” “1a” “02” “LL” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.</td></tr>
<tr><td><b>Bytes:</b></td><td colspan="3">8 or 10</td></tr>
<tr><td><b>Encoding:</b></td><td colspan="3">“ac00001a”</td></tr>
<tr><td><b>Operations:</b></td><td colspan="3">Controller → reader</td></tr>
<tr><td><b>Availability:</b></td><td colspan="3">This command is available in controllers with firmware versions of 7.2 and higher.</td></tr>
<tr><td><b>Comments:</b></td><td colspan="3"></td></tr>
</table>

## 7.25 Get IO settings

| | |
|---|---|
| **Function:** | Get the current IO configuration settings. |
| **Function Code** | 0x1b |
| **Parameters:** | No parameters are required |
| **Results:** | Parameter 0: The value of the first IO configuration byte<br>Parameter 1: The value of the second IO configuration byte<br>Parameter 2: The value of the third IO configuration byte<br>See paragraph 7.23 above for a description of the bit-fields |
| **Example:** | ***Get Statistics***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “1b” “LL” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “00” “1b” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “03” “1b” “26” “03” “80” “89” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8 or 10 |
| **Encoding:** | “ac00001b” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.2 and higher. |
| **Comments:** | |

### *7.26 Set Output 1*

| | |
|---|---|
| **Function:** | Set the output on the “Output 1” pin (previously TsStat).. |
| **Function Code** | 0x1c |
| **Parameters:** | Parameter 0: the logical level to be presented on the pin |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Set Output 1 to “1”***<br>To controller:<br>Hdr ID len cmd val LRC<br>“ab” “00” “01” “1c” “01” “b6” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “01” “1c” “01” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “1c” “LL” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 10 or 12 |
| **Encoding:** | “ab00011c01b6” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.2 and higher. |
| **Comments:** | |

## 7.27 Set Automatic RF Multiplex times

| | |
|---|---|
| **Function:** | Sets the three periods for the automatic RF multiplexing, namely, the delay period, the "mark" period and the "space" period. The RF transmitter will be off for the space and on for the mark period. This cycle will begin after the delay period.<br>Sending this command does not start the pattern (in fact any current pattern will be stopped), a multiplex synchronisation command (0x1e) must be used to start the pattern.<br>The same command without any parameters is used to stop the current pattern. |
| **Function Code** | 0x1d |
| **Parameters:** | Parameter 0: The delay period (in 10ms units). Allowed values 1 to 255 |
| | Parameter 1: The "mark" period (in 10ms units). Allowed values 1 to 255 |
| | Parameter 3: The "space" period (in 10ms units). Allowed values 1 to 255 |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Set a 1:1 mark/space pattern, 180°out of phase with 20ms period***<br>To controller:<br>Hdr ID len cmd delay mark space LRC<br>"ab" "00" "03" "1d" "01" "01" "01" "98" <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd delay mark space<br>"ac" "00" "03" "1d" "01" "01" "01" <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "00" "1d" "LL" <CR><LF><br>In the case of a terminal Instruction with "ac", we have the same reply.<br>***Stop any current pattern without changing the current settings***<br>To controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "00" "1d" "LL" <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>"ac" "00" "00" "1d" <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "00" "1d" "LL" <CR><LF><br>In the case of a terminal Instruction with "ac", we have the same reply. |
| **Bytes:** | 10 or 12 |
| **Encoding:** | "ab00031e" |
| **Operations:** | Controller → reader |

| | |
|---|---|
| **Availability:** | This command is available in controllers with firmware versions of 7.5 and higher. |
| **Comments:** | **NOTE: A Mark period of 0 is not allowed (== 256).**<br>RF Tx is switched off on receipt of this command, but the setting stored in NV memory is not changed.<br>Sending a RFTx On or Off command during multiplexing can cause the pattern to be disturbed until the end of the current mark or space period, but will not terminate the multiplexing action. The value of RFTx in NV memory will be updated to reflect the command. |

## 7.28 *Synchronise (start) Multiplexing*

| | |
|---|---|
| **Function:** | Starts the multiplexing pattern. |
| **Function Code** | 0x1e |
| **Parameters:** | None. |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Start multiplexing pattern***<br>To controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "00" "1e" "LL" <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>"a**c**" "00" "00" "1e" <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "00" "1e" "LL" <CR><LF><br>In the case of a terminal Instruction with "ac", we have the same reply. |
| **Bytes:** | 8 or 10 |
| **Encoding:** | "ab00001eLL" |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.5 and higher. |
| **Comments:** | A "short-form" of this command is available and consists of the "SYN" character (0x16) only and is not acknowledged by the reader. |

## 7.29 Set Sleep mode

| | |
|---|---|
| **Function:** | Sets “Sleep mode” on or off. |
| **Function Code** | 0x1f |
| **Parameters:** | Parameter 0: 0 or 1 to switch the RFU out of or into sleep mode. |
| **Results:** | If the length parameter is 0xff then the current state will be returned. |
| **Example:** | ***Set sleep mode ON***<br>To controller:<br>Hdr ID len cmd On/Off LRC<br>“ab” “00” “01” “1f” “01” “LL” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“ac” “00” “01” “1e” “01” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “1f” “LL” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 10 or 12 |
| **Encoding:** | “ab00011f01LL” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in handheld controllers with firmware versions of 3.x and higher. |
| **Comments:** | In sleep mode the power to the RF section is cut resulting in much lower power drain on the handheld computer’s battery. |

### 7.30 RW Action Commands

| | | | |
|---|---|---|---|
| **Function:** | The same protocol command is used to prime the RW Commander to perform all the "Over-the-air" commanding of tags. The specific command (read, write, lock etc) is coded into bits 7-5 of parameter 2 passed to the reader. | | |
| **Function Code** | 0x20 | | |
| **Parameters:** | Parameter 0: | The command sequence No. To facilitate the linking of a status message or page-read data packet to the host command that triggered it, a sequence number is included in the command and the response (not the ACK). | |
| | Parameter 1: | Bits 7-4:<br>Bits 3-0: | Start Page (0-15)<br>Number of pages |
| | Parameter 2: | Bits 7-5:<br>Bit 4:<br>Bit 3:<br>Bit 2:<br>Bits 1-0: | The command value[2]<br>extended flag: 1 = extended command [2]<br>Auto suspend, 1 = issue suspend command before read, and then resume after successful read<br>Continuous, 1 = repeat the command until a stop command is issued.<br>The target[1], 0 = immediate, 1 = broadcast, 2 = any tag but use its ID, 3 = specific tag(s) and use tag's ID. |
| | Parameters 3-10: (optional) the target tag's ID | | |
| **Results:** | No results are returned – only ACK message | | |
| **Example:** | ***See specific command examples*** | | |
| **Bytes:** | 8/10, 14/16 or 30/32 | | |
| **Encoding:** | | | |
| **Operations:** | | | |
| **Availability:** | Only R/W readers (Controller versions > 7.7) and HH versions > 3.a | | |
| **Comments:** | If a length value of 0xff is used (no parameters are passed) the reader will respond by returning the current target tag ID (as set in the most recent action command). | | |

**Notes:**

1. Target
   The target bits specify which tags to command and whether a broadcast command is issued or if a specific tag is addressed.
   There are four possible targets:
   0 = immediate – issue the command on receipt of the command from the host (do not wait until a tag has been detected). NOTE – if RF Tx is off then this command is ignored (even if the absence of RF power is due to fast multiplexing).
   1 = broadcast – the command is issued in broadcast mode if any tag is detected.
   2 = any tag, but use its ID – the command is issued to the tag which was seen.
   3 = specific tag(s) and use tag's ID – only a tag that matches the target specified in the protocol command is addressed (matching is subject to the match mask).

2. Commands
   The following table lists the commands and extended command codes which can be used:

| Command | Code | Ext* | Comment |
|---|---|---|---|
| Reserved | 0 | 0 | Reserved – factory use only |
| Reserved | 1 | 0 | Reserved – factory use only |
| Lock | 2 | 0 | Specified pages can be write-protected (cannot be reversed). |
| Write | 3 | 0 | A single page of data can be written to the tag. |
| Read | 4 | 0 | One or more (sequential) pages can be read. |
| Suspend | 5 | 0 | Commanded tags will not transmit their IDs until a resume command is received or they reset (loss of power). |
| Spare | 6 | 0 | Reserved for future use |
| Resume | 7 | 0 | Tags that were previously suspended will resume their ID transmissions. |
| Write Incremental | 0 | 1 | The contents of the data buffer is incremented by one (1) each time it is written to a tag. (A unique value is written to each tag) Note that values may be skipped due to write failure. |
| Write Date/Time | 1 | 1 | The current date/time is written to the tag. Note that duplicate values can be written to multiple tags. |
| Raw write | 2 | 1 | The same write command is repeated RetryLimit (see §7.34) times without any attempted confirming read. |
| Confirm | 3 | 1 | A single page of data is read from a tag and compared to the contents of the data buffer. A successful read is only signalled if the two are the same. |

**Table 12 RW command codes**

* Extended commands are implemented in the reader using the standard iPX command set as described in [1].

### *7.30.1 The RW Read Command*

| | | | |
|---|---|---|---|
| **Function:** | Prime the reader to perform a Read of specified pages when certain conditions are met. The read command closely mirrors the tag command itself, where both the starting page and the number of pages can be specified. Specifying the target and other common options extends the functionality. | | |
| **Function Code** | 0x20 | | |
| **Parameters:** | Parameter 0: | The command sequence No. | |
| | Parameter 1: | Bits 7-4:<br>Bits 3-0: | Start Page (0-15)<br>Number of pages |
| | Parameter 2: | Bits 7-5:<br>Bit 4:<br>Bit 3:<br><br>Bit 2:<br><br>Bits 1-0: | The read command value (4)<br>0 (not extended)<br>Auto suspend, 1 = issue suspend command before read, and then resume after successful read<br>Continuous, 1 = repeat the command until a stop command is issued.<br>The target, 0 = immediate, 1 = broadcast, 2 = any tag but use its ID, 3 = specific tag(s) and use tag's ID. |
| | Parameters 3-10: (optional) the target tag's ID | | |
| **Results:** | No results are returned – only ACK message | | |
| **Example:** | ***Prime the reader to wait for tag ID 458000001b207f9 and then to read page 1 note that the select mask must consist of all 1s to match only this tag.***<br>To controller:<br>Hdr ID len cmd Seq St/N CMD tag ID LRC<br>"ab" "00" "0b" "20" "0F" "11" "8B" "458000001b207f99" "3b" <CR><LF><br>St/N = Start and number of pages = page 1, 1 page.<br>CMD (0x8D) = Read, not ext, Auto Suspend, once off, specific tag.<br>or for terminal mode:<br>Hdr ID len cmd Seq St/N CMD tag ID<br>"a**c**" "00" "0b" "20" "0F" "11" "8B" "458000001b207f99" <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "00" "20" "22 <CR><LF><br>In the case of a terminal Instruction with "ac", we have the same reply.<br>***Read page 1 from any tag (broadcast)***<br>To controller:<br>Hdr ID len cmd Seq St/N CMD LRC<br>"ab" "00" "03" "20" "10" "11" "89" "59" <CR><LF><br>St/N = Start and number of pages = page 1, 1 page.<br>CMD (0x8D) = Read, not ext, Auto Suspend, once off, any tag.<br>Note that no tag ID need be sent. | | |

|  |  |
|---|---|
|  | Reply from controller: same as above. |
| **Bytes:** | 30/32 or 14/16 |
| **Encoding:** | “ab000b200F118B458000001b207f993b” or “ab00032010118959” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. |
| **Comments:** |  |

### 7.30.2 The RW Write Incremental

| | | | |
|---|---|---|---|
| **Function:** | Prime the reader to perform a Write of a specified page when certain conditions are met. This command causes the reader to generate new data each time a write is attempted. The contents of the data buffer is incremented (as a single 64-bit number) and the result is written to the next matching tag. Specifying the target and other common options extends the functionality. | | |
| **Function Code** | 0x20 | | |
| **Parameters:** | Parameter 0: | The command sequence No. | |
| | Parameter 1: | Bits 7-4:<br>Bits 3-0: | Start Page (0-15)<br>Number of pages. NOTE: only a single page can be written |
| | Parameter 2: | Bits 7-5:<br>Bit 4:<br>Bit 3:<br>Bit 2:<br>Bits 1-0: | The write incremental command value (0)<br>1 (extended)<br>Auto suspend, 1 = issue suspend command before read, and then resume after successful read<br>Continuous, 1 = repeat the command until a stop command is issued.<br>The target, 0 = immediate, 1 = broadcast, 2 = any tag but use its ID, 3 = specific tag(s) and use tag's ID. |
| | Parameters 3-10: (optional) the target tag's ID | | |
| **Results:** | No results are returned – only ACK message | | |
| **Example:** | ***Prime the reader to wait for tag ID 4580000015906c49 and then to write the current contents of the data buffer incremented by one to page 1 of the tag. This should be repeated at the current ReDo interval as long as (or whenever) the tag is in the read beam.***<br>***note that the select mask must consist of all 1s to match only this tag.***<br>To controller:<br>Hdr ID len cmd Seq St/N CMD tag ID LRC<br>“ab” “00” “0b” “20” “cb” “11” “1f” “4580000015906c49” “78” <CR><LF><br>St/N = Start and number of pages = page 1, 1 page.<br>CMD (0x8D) = Write incr, ext, Auto Suspend, continuous, specific tag.<br>or for terminal mode:<br>Hdr ID len cmd Seq St/N CMD tag ID<br>“a**c**” “00” “0b” “20” “cb” “11” “1f” “4580000015906c49” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “20” “22 <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. | | |
| **Bytes:** | 30/32 or 14/16 | | |
| **Encoding:** | “ab000b20cb111f4580000015906c4978” | | |
| **Operations:** | Controller → reader | | |

|  |  |
|---|---|
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. |
| **Comments:** |  |

## 7.31 The RW Set Data Command

| | | |
|---|---|---|
| **Function:** | Set the data to be written to a tag | |
| **Function Code** | 0x21 | |
| **Parameters:** | Parameters 0-7: | The data bytes. |
| **Results:** | No results are returned – only ACK message (* see Comment for exception). | |
| **Example:** | ***Send aaaaaaaaaaaaaaaa to the reader as the data to next be written.***<br>To controller:<br>Hdr ID len cmd the data LRC<br>“ab” “00” “08” “21” “aaaaaaaaaaaaaaaa” “3b” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd the data<br>“a**c**” “00” “08” “21” “aaaaaaaaaaaaaaaa” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “21” “23” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.<br>***Get the current data***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “ff” “21” “8f”<br>Reply from controller:<br>Hdr ID len cmd data LRC<br>“ab” “00” “08” “21” “aaaaaaaaaaaaaaaa” “3b” <CR><LF>. | |
| **Bytes:** | 8/10 or 24/26 | |
| **Encoding:** | “ab000821aaaaaaaaaaaaaaaa3b” or “ab00ff218f” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. | |
| **Comments:** | **Setting the data to be written stops any pending RW commands (after v8.0).**<br>Using a length parameter of “ff” will cause the reader to respond with the current data, as in the second example above. | |

## 7.32 The RW Set Tag Mask Command

| | | |
|---|---|---|
| **Function:** | Set the tag selection mask | |
| **Function Code** | 0x22 | |
| **Parameters:** | Parameters 0-7: | The mask bytes. |
| **Results:** | No results are returned – only ACK message (* see Comment for exception). | |
| **Example:** | ***Set the mask to fff0000000000000 – this together with a target ID of 4580000000000000 will cause any iPX4 tag to be commanded.***<br>To controller:<br>Hdr ID len cmd the mask LRC<br>“ab” “00” “08” “22” “fff0000000000000” “ce” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd the mask<br>“ac” “00” “08” “22” “fff0000000000000” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “22” “24” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.<br>***Get the current mask***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “ff” “22” “90”<br>Reply from controller:<br>Hdr ID len cmd mask LRC<br>“ab” “00” “08” “22” “fff0000000000000” “ce” <CR><LF>. | |
| **Bytes:** | 8/10 or 24/26 | |
| **Encoding:** | “ab000822fff0000000000000ce” or “ab00ff2290” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. | |
| **Comments:** | Using a length parameter of “ff” will cause the reader to respond with the current mask, as in the second example above.<br>The ID of an observed tag is first exclusive-ORed with the target and then masked by ANDing with the mask. The result must be all zeros for a match to be registered.<br>This means that in bit positions where the mask is a 1 the corresponding bit of the observed tag must be the same (0 or 1) as the corresponding bit of the target. | |

## 7.33 The RW Stop Command

| | |
|---|---|
| **Function:** | This command stops any continuous or pending command |
| **Function Code** | 0x23 |
| **Parameters:** | none |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Stop a pending write to a tag.***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “23” “25” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “00” “23” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “23” “25” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8/10 |
| **Encoding:** | “ab00002325” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. |
| **Comments:** | Any command sent to the reader by means of command “20” will wait until a matching tag is seen (or the timeout expires – when that has been implemented). Also if the “continuous” bit was set then the command will be repeated whenever a tag matches. To interrupt this behaviour the stop command must be sent. |

## 7.34 The RW Set Timeouts Command

| | | |
|---|---|---|
| **Function:** | This command sets the various timeouts that govern the operation of the Commander engine. | |
| **Function Code** | 0x24 | |
| **Parameters:** | Parameter 0: | CommandTimeout – the period (in seconds) within which a matching tag must be seen or an error is reported (NYI). |
| | Parameter 1: | ReadTimeout – the period, after a read command is issued over the air, within which an ID will be interpreted as data. If nothing is seen in this period then a failure is reported. The value is specified as 256 – N where N is the required time is 25.6µs units. For 256k tags the currently used value is 220. |
| | Parameter 2: | RedoTO – the period, in seconds, to wait before re-commanding a tag. |
| | Parameter 3: | ReadDelay – the period to delay after issuing a write command before the automatic read command is issued. The units and specified value are the same as for the read timeout.<br>This time is also used as the delay between successive retries. |
| | Parameter 4: | RetryLimit – the number of times the commander must try any command that fails. |
| | Parameter 5: | ReadRetryLimit – the number of times the commander must try read commands when a write confirmation fails or when voting is used on reads. |
| **Results:** | No results are returned – only ACK message (* a length value of “ff” will cause the current values to be sent by the reader). | |
| **Example:** | ***Set the command timeout to 0 (NYI), the read timeout to 220, the redo timeout to 5s, the read delay to 109 and the retry limit to 5.***<br>To controller:<br>Hdr ID len cmd CmdTO RdTO ReTO RdDel Retry LRC<br>“ab” “00” “05” “24” “00” “dc” “05” “6d” “05” “b6” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd CmdTO RdTO ReTO RdDel Retry<br>“a**c**” “00” “05” “24” “00” “dc” “05” “6d” “05” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “24” “26” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. | |
| **Bytes:** | 18/20 | |
| **Encoding:** | “ab00052400dc056d05b6” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. | |

|  |  |
|---|---|
| **Comments:** | The stack of tags already commanded is only 32 entries deep; if more tags than this are in the beam within the ReDo timeout, tags will be re-commanded more than desired. |

### 7.35 The RW Immediate action Commands

Certain iPX RW commands can usefully be issued without first priming the commander, waiting for a tag and then doing error checking. Commands such as suspend and resume can usefully be broadcast “asynchronously”.

| | | |
|---|---|---|
| **Function:** | Immediate action Suspend/Resume | |
| **Function Code** | 0x25 | |
| **Parameters:** | Parameters 0: | The IP-X RW command (suspend or resume) shifted into place as required by the air protocol (“a0” – suspend, “e0” – resume) (see [1]). |
| **Results:** | No results are returned – only ACK message. | |
| **Example:** | ***Send a suspend command..***<br>To controller:<br>Hdr ID len cmd susp LRC<br>“ab” “00” “01” “25” “a0” “LL” &lt;CR&gt;&lt;LF&gt;<br>or for terminal mode:<br>Hdr ID len cmd susp<br>“ac” “00” “01” “25” “a0” &lt;CR&gt;&lt;LF&gt;<br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “25” “27” &lt;CR&gt;&lt;LF&gt;<br>In the case of a terminal Instruction with “ac”, we have the same reply. | |
| **Bytes:** | 10/12 | |
| **Encoding:** | “” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. | |
| **Comments:** | These commands are not “reliable”; that is, the commander does not check that they have had the desired effect and retry in the event of failure. | |

### 7.36 Get RW Command Status

| | | | |
|---|---|---|---|
| **Function:** | Gets the status of the RW Command Engine. | | |
| **Function Code** | 0x26 | | |
| **Parameters:** | No parameters are required | | |
| **Results:** | Parameter 0: | x4State | The current state of the Command Engine |
| | | bit 0 | a read was issued - a page should pitch |
| | | bit 1 | a command must be issued when a tag matches |
| | | bit 2 | a read has confirmed the success of a write |
| | | bit 3 | a read is busy - don't process spool buffer |
| | | bit 4 | reading more than one page |
| | | bit 5 | a read following a write is pending |
| | | bit 6 | a command or retries is busy |
| | | bit 7 | a retry is pending |
| | Parameter 1: | x4Status | The current state of the Command Engine |
| | | bit 0 | a timeout occurred on page read |
| | | bit 1 | timeout timer must be started after command |
| | | bit 2 | the tag has not changed since the last command |
| | | bit 3 | the tag in lastTagSeen has been commanded |
| | | bit 4 | a write command is being performed - write data |
| | | bit 5 | a data page was detected |
| | | bit 6 | a timeout message must be sent to host |
| | | bit 7 | too many retries |
| | Parameter 2: | x4Options | The RW options selected |
| | | bit 0 | auto read after write |
| | | bit 1 | auto suspend/resume |
| | | bit 2 | a specific tag, not just any |
| | | bit 3 | if set then use the tags ID else broadcast |
| | | bit 4 | repeat the command until stop is issued |
| | | bit 5 | use "voting" and multiple reads |
| | | bit 6 | If set then the command sequence number is returned in the page data message (the default), else the remaining retry and read retry counts are packed into a byte in the message. |
| | | bit 7 | not used |
| | Parameter 3: | x4Options2 | More RW options |
| | | bit 0 | Set if the last data page tested passed the CRC check |
| | | bit 1 | not used |
| | | bit 3-2 | Data page CRC checking options<br>00 – none<br>01 – 4022<br>10 – IPX<br>11 – CCITT |
| | | bit 4 | not used |

| | | | |
|---|---|---|---|
| | | bit 5 | not used |
| | | bit 6 | not used |
| | | bit 7 | not used |
| | Parameter 4: | x4ToDo | The command to issue (see [1]) |
| | Parameter 5: | bits <7-4> | the start page |
| | | bits <3-0> | the number of pages |
| | Parameter 6: | bits <7-4> | the current page |
| | | bits <3-0> | the number of pages still to be processed |
| | Parameter 7: | retry count | the number of retries still to process |
| | Parameter 8&9: | | the command as transmitted to the tag including CRC5 (see [1]) |
| | Parameter 10-15: | ID | The ID attached to the last command transmitted. This is either all zeros for broadcast, or the bottom six bytes of the addressed tag's ID (see [1]).. |

| | |
|---|---|
| **Example:** | ***Get Commander Status***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “26” “LL” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“ac” “00” “00” “26” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd ste stu opt CMD 1/n n/l retry xmitted cmd + ID<br>“ab” “00” “0f” “26” “02” “00” “4e” “03” “11” “20” “00” “e00800001719ebd2”<br>LRC<br>“2c” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8 or 10 |
| **Encoding:** | “ac000026” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. |
| **Comments:** | The response to this command is also used as an unsolicited message when an operation (tag command) fails. |

## 7.37 The Save RW Settings to EEPROM Command

| | |
|---|---|
| **Function:** | Save settings to EEPROM |
| **Function Code** | 0x27 |
| **Parameters:** | none |
| **Results:** | No results are returned – only ACK message. |
| **Example:** | ***Record all settings.***<br>To controller:<br>Hdr   ID   len   cmd  LRC<br>“ab”  “00”  “00”  “27”  “29”  <CR><LF><br>or for terminal mode:<br>Hdr   ID   len   cmd<br>“ac”  “00” “00”  “27”  <CR><LF><br>Reply from controller:<br>Hdr   ID    len  cmd  LRC<br>“ab”  “00”  “00”  “27”  “29” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8/10 |
| **Encoding:** | “ab00002729” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. |
| **Comments:** | The iPX RW Commander settings are not saved to NV memory whenever a value is changed.  Instead all values are written to EEPROM (even if other NV data is stored to the NV RAM in the Dallas clock device) only on demand. |

## 7.38 The Set RW Options Command

| | | |
|---|---|---|
| **Function:** | Set the RW options | |
| **Function Code** | 0x28 | |
| **Parameters:** | Parameters 0: | The option bits |
| | Bit 0: | If set then a write operation will automatically be followed by a read to confirm the success of the write. |
| | Bit 1: | If set then any read or write command will be preceded by a suspend command. After completion of the operation (including retries) a resume command will be issued. (In future a Tx break may be used as a more reliable resume).<br>NOTE: other second generation RW commands set or reset this bit. |
| | Bit 2: | Only a specific tag (which matched the masked target) will be commanded.<br>NOTE: other second generation RW commands set or reset this bit. |
| | Bit 3: | The ID of an observed tag will be used to address the command.<br>NOTE: other second generation RW commands set or reset this bit. |
| | Bit 4: | The command will be issued to all tags until a stop command (or another RW command) is received.<br>NOTE: other second generation RW commands set or reset this bit. |
| | Bit 5: | If set then a “voting” mechanism is used for read confirmation |
| | Bit 6: | If set then the command sequence number is returned in the page data message (the default), else the remaining retry and read retry counts are packed into a byte in the message. |
| | Bit 7: | |
| | Parameter 1: | More options (optional) |
| | Bit 0: | Result of last data page CRC check performed – 1 = CRC passed |
| | Bit 1: | If set – the controller will generate the CRC for the data (NYI) |
| | Bit 3-2: | CRC checking to be performed on data<br>00 – none<br>01 – 4022<br>10 – IPX<br>11 – CCITT |
| | Bit 4: | If set – unconfirmed reads are not transmitted to host (NYI) |
| | Bit 5: | If set – CRC checking is not performed on page 15 |
| | bit 7-6 | not used |
| **Results:** | No results are returned – only ACK message (* see Comment for exception). | |
| **Example:** | ***Set the options to their default values.***<br>To controller:<br>Hdr ID len cmd the options LRC<br>“ab” “00” “02” “28” “61” “00” “LL” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd options<br>“a**c**” “00” “02” “28” “61” “00” <CR><LF><br>Reply from controller: | |

| | |
|---|---|
| | Hdr ID len cmd LRC<br>“ab” “00” “00” “28” “24” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.<br>***Get the current options***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “ff” “28” “LL”<br>Reply from controller:<br>Hdr ID len cmd options LRC<br>“ab” “00” “01” “28” “61” “LL” <CR><LF>. |
| **Bytes:** | 10/12 |
| **Encoding:** | “ab000128xx” or “ab00ff2890” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.e and higher. |
| **Comments:** | Using a length parameter of “ff” will cause the reader to respond with the current options, as in the second example above. |

## 7.39 The Set TTO Options Command

| | | |
|---|---|---|
| **Function:** | Set the TTO options | |
| **Function Code** | 0x29 | |
| **Parameters:** | Parameters 0: | The option bits |
| | Bits 3-0: | Number of TTO pages expected. |
| | Bits 6-4: | Threshold - only report a page if it is seen at least this number of times. NYI. |
| | Bit 7: | if set then only expected pages are reported . |
| | Parameter 1: | More options (optional) |
| | Bit 0: | use short-format TTO page report |
| | Bit 1: | switch TTO processing off treat all as ID |
| | Bit 2: | discard unconfirmed TTO data pages (NYI – a simple test would discard pages if the ring-buffer was scanned before enough counts were accumulated – would need some delay like first-seen/last-seen) |
| | bit 7-3 | not used |
| **Results:** | No results are returned – only ACK message (* see Comment for exception). | |
| **Example:** | ***Set the TTO options to their default values.***<br>To controller:<br>Hdr ID len cmd the options LRC<br>“ab” “00” “02” “29” “00” “01” “LL” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “29” “2b” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.<br>***Get the current options***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “ff” “29” “97”<br>Reply from controller:<br>Hdr ID len cmd options LRC<br>“ab” “00” “02” “29” “00” “01” “LL” <CR><LF>. | |
| **Bytes:** | 10/12 | |
| **Encoding:** | “ab000129xxyy” or “ab00ff2997” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 10.4 and higher. | |
| **Comments:** | Using a length parameter of “ff” will cause the reader to respond with the current options, as in the second example above. | |

## 7.40 The Set Expected Number of TTO Pages Command

| | |
|---|---|
| **Function:** | Set the Expected Number of TTO Pages |
| **Function Code** | 0x2a |
| **Parameters:** | Parameter 0: The expected number of TTO pages.<br>Saved in bits 3-0 of the TTO options word. The controller is also set to report only this number of pages (bit 7 of TTO options word is set). |
| **Results:** | No results are returned – only ACK message (* see Comment for exception). |
| **Example:** | ***Set the expected number of TTO pages to 2.***<br>To controller:<br>Hdr ID len cmd N LRC<br>“ab” “00” “01” “2a” “02” “b6” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “2a” “53” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.<br>***Get the current options***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “ff” “2a” “bf”<br>Reply from controller:<br>Hdr ID len cmd N LRC<br>“ab” “00” “01” “2a” “01” “LL” <CR><LF>. |
| **Bytes:** | 10/12 |
| **Encoding:** | “ab00012a02b6” or “ab00ff2abf” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 10.4 and higher. |
| **Comments:** | Using a length parameter of “ff” will cause the reader to respond with the current number of pages expected, as in the second example above. |

## 7.41 The Tag Filter Commands

Controllers with versions 7.7 or greater can have a filtering capability where tags must match a certain pattern and/or not match a reject pattern. Whether this feature is included in a firmware build depends on the PIC used and whether the largest possible buffer is being used for the tag counting/reporting function.

When included, a target pattern and its corresponding mask can be set, as well as a “reject” pattern and its mask.

If a tag, masked by the reject mask, matches the reject pattern then the tag is rejected and not recorded (or commanded by the RW commander).

If the tag does not match the reject pattern then it is compared against the select pattern and is recorded if it matches.

The ID of an observed tag is first exclusive-ORed with the reject pattern and then masked by ANDing with the reject mask.  If the result is all zeros, the tag is rejected.

To not reject any tags use a reject target of all zeros and a reject mask of all ones (this will still reject a tag of all zeros, but this is a special case which has been rejected since IPICO decoders began – there is an option in the CRC options to accept “all zero” tags which will bypass this test).

This means that in bit positions where the mask is a 1, the corresponding bit of the observed tag must be the same (0 or 1) as the corresponding bit of the select pattern.

The ID of the observed tag is then exclusive-ORed with the select pattern and then masked by ANDing with the select mask.  The result must be all zeros for a match to be registered.

To select all tags use a selection mask of all zeros.

There are commands for setting both targets and both masks:

| **Operation** | **Command** | **Section** |
|---|---|---|
| Set select pattern | 0x30 | |
| Set select mask | 0x31 | |
| Set rejest pattern | 0x32 | |
| Set reject mask | 0x33 | |

There is a command to save these patterns and masks to EEPROM.  These values are not saved to NV memory each time they are changed (as are most IPICO reader operating parameters and states), but must be written to EEPROM when required.

NOTE that the PIC EEPROM can only be written a limited number of times (in the order of 10000 to 100000 times).

### 7.41.1 Set select Pattern

| | | |
|---|---|---|
| **Function:** | Set the tag selection ID | |
| **Function Code** | 0x30 | |
| **Parameters:** | Parameters 0-7: | The ID bytes. |
| **Results:** | No results are returned – only ACK message (* see Comment for exception). | |
| **Example:** | ***Set the target ID to 4580000000000000 (iPX4 tags).***<br>To controller:<br>Hdr ID len cmd the pattern LRC<br>“ab” “00” “08” “30” “4580000000000000” “3c” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd the pattern<br>“a**c**” “00” “08” “30” “4580000000000000” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “30” “23” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.<br>***Get the current pattern***<br>To controller:<br>Hdr ID len cmd<br>“ac” “00” “ff” “30”<br>Reply from controller:<br>Hdr ID len cmd pattern LRC<br>“ab” “00” “08” “30” “4580000000000000” “3c” <CR><LF>. | |
| **Bytes:** | 8/10 or 24/26 | |
| **Encoding:** | “ab00083045800000000000003c” or “ab00ff30” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. This feature may be excluded from a firmware build if the bigger tag-seen buffer is required. | |
| **Comments:** | Using a length parameter of “ff” will cause the reader to respond with the current mask, as in the second example above.<br>The ID of an observed tag is first exclusive-ORed with the target and then masked by ANDing with the mask. The result must be all zeros for a match to be registered.<br>This means that in bit positions where the mask is a 1 the corresponding bit of the observed tag must be the same (0 or 1) as the corresponding bit of the target. | |

### 7.41.2 The Set Tag Select Mask Command

| | |
|---|---|
| **Function:** | Set the tag selection mask |
| **Function Code** | 0x31 |
| **Parameters:** | Parameters 0-7: The mask bytes. |
| **Results:** | No results are returned – only ACK message (* see Comment for exception). |
| **Example:** | ***Set the mask to fff0000000000000 – this together with a target pattern of 4580000000000000 will cause any iPX4 tag to be seen.***<br>To controller:<br>Hdr ID len cmd the mask LRC<br>"ab" "00" "08" "31" "fff0000000000000" "LL" <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd the mask<br>"ac" "00" "08" "31" "fff0000000000000" <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>"ab" "00" "00" "31" "24" <CR><LF><br>In the case of a terminal Instruction with "ac", we have the same reply.<br>***Get the current mask***<br>To controller:<br>Hdr ID len cmd<br>"ac" "00" "ff" "31"<br>Reply from controller:<br>Hdr ID len cmd mask LRC<br>"ab" "00" "08" "31" "0000000000000000" "2c" <CR><LF>. |
| **Bytes:** | 8/10 or 24/26 |
| **Encoding:** | "ab000822fff0000000000000ce" or "ab00ff2290" |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. |
| **Comments:** | Using a length parameter of "ff" will cause the reader to respond with the current mask, as in the second example above.<br>The ID of an observed tag is first exclusive-ORed with the target and then masked by ANDing with the mask. The result must be all zeros for a match to be registered.<br>This means that in bit positions where the mask is a 1 the corresponding bit of the observed tag must be the same (0 or 1) as the corresponding bit of the target. |

### 7.41.3 Set The Reject Pattern

| | | |
|---|---|---|
| **Function:** | Set the tag rejection ID | |
| **Function Code** | 0x32 | |
| **Parameters:** | Parameters 0-7: | The ID bytes. |
| **Results:** | No results are returned – only ACK message (* see Comment for exception). | |
| **Example:** | ***Set the reject pattern to 4580000000000000 (iPX4 tags).***<br>To controller:<br>Hdr ID len cmd the pattern LRC<br>“ab” “00” “08” “32” “4580000000000000” “3e” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd the pattern<br>“a**c**” “00” “08” “32” “4580000000000000” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “32” “25” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.<br>***Get the current pattern***<br>To controller:<br>Hdr ID len cmd<br>“ac” “00” “ff” “32”<br>Reply from controller:<br>Hdr ID len cmd pattern LRC<br>“ab” “00” “08” “32” “4580000000000000” “3e” <CR><LF>. | |
| **Bytes:** | 8/10 or 24/26 | |
| **Encoding:** | “ab000832458000000000000003e” or “ab00ff32” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. This feature may be excluded from a firmware build if the bigger tag-seen buffer is required. | |
| **Comments:** | Using a length parameter of “ff” will cause the reader to respond with the current mask, as in the second example above. | |

### 7.41.4 Set Tag Reject Mask

| | | |
|---|---|---|
| **Function:** | Set the tag rejection mask | |
| **Function Code** | 0x33 | |
| **Parameters:** | Parameters 0-7: | The mask bytes. |
| **Results:** | No results are returned – only ACK message (* see Comment for exception). | |
| **Example:** | ***Set the reject mask to fff0000000000000 – this together with a target pattern of 4580000000000000 will cause any iPX4 tag to be rejected.***<br>To controller:<br>Hdr ID len cmd the mask LRC<br>“ab” “00” “08” “33” “fff0000000000000” “LL” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd the mask<br>“a**c**” “00” “08” “33” “fff0000000000000” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “33” “LL” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply.<br>***Get the current mask***<br>To controller:<br>Hdr ID len cmd<br>“ac” “00” “ff” “33”<br>Reply from controller:<br>Hdr ID len cmd mask LRC<br>“ab” “00” “08” “33” “ffffffffffffff8e” “8e” <CR><LF>. | |
| **Bytes:** | 8/10 or 24/26 | |
| **Encoding:** | “ab000822fff0000000000000ce” or “ab00ff2290” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. | |
| **Comments:** | Using a length parameter of “ff” will cause the reader to respond with the current mask, as in the second example above.<br>The ID of an observed tag is first exclusive-ORed with the target and then masked by ANDing with the mask. The result must be all zeros for a match to be registered.<br>This means that in bit positions where the mask is a 1 the corresponding bit of the observed tag must be the same (0 or 1) as the corresponding bit of the target. | |

## 7.42 The Save Tag Selection Settings to EEPROM Command

| | |
|---|---|
| **Function:** | Save tag selection settings to EEPROM |
| **Function Code** | 0x34 |
| **Parameters:** | none |
| **Results:** | No results are returned – only ACK message. |
| **Example:** | ***Record all tag selection settings.***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “34” “27” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “00” “34” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “34” “27” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8/10 |
| **Encoding:** | “ab00003427” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 7.7 and higher. |
| **Comments:** | The tag selection settings are not saved to NV memory whenever a value is changed. Instead all values are written to EEPROM (even if other NV data is stored to the NV RAM in the Dallas clock device) only on demand. |

## 7.43 The Set/Get Test Options Command

| | | | |
|---|---|---|---|
| **Function:** | Sets or gets the test options byte. | | |
| **Function Code** | 0x35 | | |
| **Parameters:** | Parameter 0: | The test options | |
| | Bit | Description | |
| | 0 | Send a test ID. | The “ID” stored in the “select” pattern (as described in section 7.41.1 above) is repeatedly added to the tags seen stack and processed according to the current message mode (normal, event etc). |
| | 1-7 | Unused | |
| **Results:** | No parameters returned – only Ack<br>If a length of 0xff is specified without any parameter, then the current settings will be returned. The format of the returned value is as described above. | | |
| **Example:** | ***Set Test options to 0x00 (All test options disabled)***<br>To controller:<br>Hdr ID len cmd TOpt LRC<br>“ab” “00” “01” “35” “00” “LL” &lt;CR&gt;&lt;LF&gt;<br>or for terminal mode:<br>Hdr ID len cmd TOpt<br>“ac” “00” “00” “35” “00” &lt;CR&gt;&lt;LF&gt;<br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “35” “28” &lt;CR&gt;&lt;LF&gt;<br>In the case of a terminal Instruction with “ac”, we have the same reply.<br>To query the current value of the test options use:<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “ff” “35” “LL” &lt;CR&gt;&lt;LF&gt;<br>Reply from controller:<br>Hdr ID len cmd TOpt LRC<br>“ab” “00” “01” “35” “00” “89” &lt;CR&gt;&lt;LF&gt; | | |
| **Bytes:** | 10 or 12 (or 8 or 10 for query) | | |
| **Encoding:** | “ac00013500” | | |
| **Operations:** | Controller → reader | | |
| **Availability:** | This command is available in controllers with firmware versions of 8.0 and higher. | | |

## 7.44 *Modify part of Config 3*

| | |
|---|---|
| **Function:** | Instruction to set part of the configuration 3 options. This command is used to change the message mode, for example, without changing the other settings |
| **Function Code** | 0x36 |
| **Parameters** | Parameter 0: Mask of bits to change (where a bit is a ‘1’ the corresponding value from parameter 1 will be written to Config 3.<br>Parameter 1: Bit-mask of configuration options (must be in correct bit position)<br>Parameter 2: (optional) The timeout period for “event” message mode. |

| **Bit** | **Item** | **Mask** | **Description** |
|---|---|---|---|
| 2 – 0 | message mode | 0x07 | 000 = normal<br>001 = event 2<br>010 = trigger 2<br>011 = event |
| 3 | send status as tag | 0x08 | If set, any status message from the decoders will be handled as if it were a special tag and transmitted to the host. |
| 4 | Sleep | 0x10 | If set then the controller will assert the “Sleep” pin on startup. |
| 5 | Sleep active low | 0x20 | If set then the sleep pin must be pulled low to cause the RFU to sleep |
| 6 | Sleep tracks RF Tx | 0x40 | If set then the sleep state tracks RF Tx Off – the RFU will be put into sleep mode whenever the RF power is off. |
| 7 | Listen while RF off | 0x80 | if set then tags can be read even when the RFU is not transmitting. If not set then tags read while RF Tx is Off will be ignored. |

Parameter 1: Event timeout value (optional)

| | |
|---|---|
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Set sleep on at start-up***<br>To controller:<br>Hdr ID len cmd mask cfg LRC<br>“ab” “00” “02” “36” “10” “10” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd cfg<br>“ac” “00” “02” “36” “10” “10” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “36” “LL” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 12/14 or 14/16 |
| **Encoding:** | “ac0002360700xx”<br>reply = “ab000036LL” |

| | |
|---|---|
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware version of 8.1, FPGA 10.2 & HH4.0. |
| **Comments:** | See §7.9 above for a description of the options.<br>More one option can be changed at once, for example, to set the start-up Sleep mode on and to set the sleep signal to be active low use a mask of 0x30 and a value of 0x30. |

### 7.45 Print the start-up banner

| | |
|---|---|
| **Function:** | Print the start-up banner without resetting the controller |
| **Function Code** | 0x37 |
| **Parameters:** | none |
| **Results:** | The start-up banner is sent followed by an ACK message. |
| **Example:** | ***Record all tag selection settings.***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “37” “LL” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “00” “37” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “37” “LL” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8/10 |
| **Encoding:** | “ab000037LL” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 8.1 and higher. |
| **Comments:** | |

## 7.46 Get Frequency

| | |
|---|---|
| **Function:** | Get the frequency counters from the 3 PIC test board. |
| **Function Code** | 0x38 |
| **Parameters:** | none |
| **Results:** | The frequency counters collected by the test board are returned.<br>Byte 0: Edge count high byte<br>Byte 1: Edge count low byte<br>Byte 2: Tag time high byte<br>Byte 3: Tag time low byte |
| **Example:** | ***Query the frequency counters.***<br>To controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “38” “2b” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd<br>“a**c**” “00” “00” “38” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd EH EL TH TL LRC<br>“ab” “00” “04” “38” “77” “02” “10” “36” “c9” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 8/10 |
| **Encoding:** | “ab000038LL” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is only available in test board controllers with firmware versions of x.x and higher. |
| **Comments:** | |

## 7.47 Set Frequency Hopping timing

| | | |
|---|---|---|
| **Function:** | Set the frequency hopping timing parameters | |
| **Function Code** | 0x39 | |
| **Parameters:** | Parameter 0: | The dwell time at each frequency in 10ms units. |
| | Parameter 1: | The time for which the RF Tx is modulated (in micro fortnights). |
| **Results:** | None - an ACK message (* see Comment for exception). | |
| **Example:** | ***Set the dwell time to 360ms and the modulate duration to 6.***<br>To controller:<br>Hdr ID len cmd dwell mod LRC<br>“ab” “00” “02” “39” “24” “06” “fa” &lt;CR&gt;&lt;LF&gt;<br>or for terminal mode:<br>Hdr ID len cmd dwell mod<br>“ac” “00” “02” “39” “24” “06” &lt;CR&gt;&lt;LF&gt;<br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “37” “LL” &lt;CR&gt;&lt;LF&gt;<br>In the case of a terminal Instruction with “ac”, we have the same reply. | |
| **Bytes:** | 12/14 | |
| **Encoding:** | “ab0002392406fa” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware versions of 11.0 and higher. | |
| **Comments:** | Using a length parameter of “ff” will cause the reader to respond with the current times, as in the second example above. | |

## 7.48 Modulate RF Tx

| | |
|---|---|
| **Function:** | Instruction to modulate the RF transmitter. |
| **Function Code** | 0x3a |
| **Parameters** | Parameter 0: 1 or 0 to modulate the RF transmitter or not |
| **Results:** | No results are returned – only ACK message |
| **Description:** | This Instruction causes the FPGA to assert/de-assert the modulate line. |
| **Example:** | Set Modulate On/Off<br>Modulate<br>To controller:<br>Hdr ID len cmd val LRC<br>“ab” “00” “01” “3a” “01” “b6” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd val<br>“ac” “00” “01” “3a” “01” <CR><LF><br>Acknowledgement from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “3a” “54” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | 10 or 12 |
| **Encoding:** | “ac00013a01b6” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware versions of 11.0 and higher. |
| **Comments:** | This state is not saved to NV memory. |

## 7.49 Set decoder Configuration

This command is used with single PIC readers (Handheld) and FPGA based decoders.

<table>
<tr><td><b>Function:</b></td><td colspan="3">Sets the decoder configuration, specifically the baud rate of tags to be decoded and configuration parameters for Dual Frequency (DF) readers and DUAL DSP decoders.</td></tr>
<tr><td><b>Function Code</b></td><td colspan="3">0x3b</td></tr>
<tr><td><b>Description:</b></td><td colspan="3">The tag baud rate is selected, as are settings required by the FPGA to control DF RFUs.<br>In the single PIC HH version the configuration word is saved in EEPROM on the PIC itself. Users of this feature must be aware of the write “endurance” which can be as low as 1000 cycles at temperatures above 85°C.</td></tr>
<tr><td><b>Parameters</b></td><td colspan="3">Parameter 0: Bit-mask configuration options</td></tr>
<tr><td></td><td><b>Bit</b></td><td><b>Item</b></td><td><b>Description</b></td></tr>
<tr><td></td><td>7</td><td>TXOFF polarity control</td><td>0 = Normal, 1 = Inverted</td></tr>
<tr><td></td><td>6</td><td>Synthesizer control</td><td>0 = No carrier, 1 = 125 kHz carrier</td></tr>
<tr><td></td><td>5</td><td>Duty cycle control</td><td>0 = 50%, 1 = 100%</td></tr>
<tr><td></td><td>4</td><td>Modulate control</td><td>0 = OFF, 1 = ON</td></tr>
<tr><td></td><td>3</td><td>Input invert</td><td>0 = no invert, 1 = invert input signal</td></tr>
<tr><td></td><td>2</td><td>DSP control</td><td>0 = normal, 1 = DSP filter bypass</td></tr>
<tr><td></td><td>1-0</td><td>Baud rate</td><td>0 – 32kb/s (Not yet supported)<br>1 – 64kb/s<br>10 – 128kb/s (Not yet supported)<br>11 – 256kb/s</td></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><td><b>Results:</b></td><td colspan="3">No results are returned – only ACK message</td></tr>
<tr><td><b>Example:</b></td><td colspan="3"><b><i>setting the tag baudrate to 256k</i></b><br>To controller:<br>Hdr ID len cmd val lrc<br>“ab” “00” ”01” “12” “03” “87” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd lrc<br>“ab” “00” “00” “12” “</td></tr>
<tr><td><b>Bytes:</b></td><td colspan="3">0</td></tr>
<tr><td><b>Encoding:</b></td><td colspan="3">“ab0001120387”<br>reply – “ab000012xx”</td></tr>
<tr><td><b>Operations:</b></td><td colspan="3">Controller → reader</td></tr>
<tr><td><b>Availability:</b></td><td colspan="3">This command is available in readers with build dates after 2007-09-10.</td></tr>
</table>

|  |  |
|---|---|
| **Comments:** | This command is required by HH and FPGA based readers.<br>Legacy (3 PIC) readers will issue a “Bad command” error message).<br>DF FPGA readers can generate a 125kHz signal from which the carrier is derived. and also control the duty cycle of the carrier.<br>Bits 4 and 3 are used to control the "Dual DSP" FPGA decoder used in the ARM based controller.<br>Sending a length of 0xff (but without and data bytes) will cause the reader to reply with the current value of the decoder configuration word.<br>A command with length 0xc2 (with two parameters) performs and AND-OR operation – allowing one to change only selected bits:<br>For example to change only the TXOFF polarity to "normal" without changing the tag baud rate (or any of the other settings use the command ac00c23b7f00. |

## 7.50 Get Last I2C read results

| | | |
|---|---|---|
| **Function:** | A function to retrieve the status and data from the most recently executed I2C read operation. | |
| **Function Code** | 0x80 | |
| **Parameters** | Parameter 0: The offset into the read buffer where data transfer is to begin | |
| **Results:** | Byte 0: Error code<br>Byte 1: Fail state<br>Byte 2: I2C address<br>Byte 3: read count<br>Byte 4-16: The read buffer contents from the specified offset on | |
| **Example:** | ***Get Last I2C read results***<br>To controller:<br>Hdr ID len cmd offset LRC<br>“ab” “00” “01” “80” “00” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd offset<br>“a**c**” “00” “01” “80” “00” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd err state addr count … data …<br>“ab” “00” “10” “80” “30” “01” “68” “0f” “34” “39” “10” “03” “27” “11” “02”<br>… LRC<br>“10” “00” “11” “1f” “ff” “3b” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. | |
| **Bytes:** | Command: 10 or 12 | Reply: 42 |
| **Encoding:** | “ac00018000” | |
| **Operations:** | Controller → reader | |
| **Availability:** | This command is available in controllers with firmware version of 6.1 and higher. | |
| **Comments:** | This command is used for debugging purposes.<br>The input parameters:<br>Byte 0: Offset – since the protocol, or rather the internal buffers, can only handle 16 bytes of data, while the I2C read may involve up to 32 bytes, the user can specify the offset at which to begin the transfer. The status bytes (0-3) of the reply also restrict the data that can be seen at once.<br>The returned parameters have the following significance:<br>Byte 0: Error code – this is the inclusive OR of the error codes detected since last reset. This command resets this parameter<br>Byte 1: Fail state – the state of the I2C state-machine when the last error was detected<br>Byte 2: I2C address – the “unshifted” I2C address used for the read<br>Byte 3: read count – the number of bytes read<br>Byte 4-16: The read buffer contents from the specified offset onwards | |

## 7.51 I2C read or write operation

| | |
|---|---|
| **Function:** | A function to execute an I2C read operation. |
| **Function Code** | 0x81 |
| **Parameters** | Parameter 0: The I2C address<br>Parameter 1: R_W – the read/write flag, 1 = read, 0 = write<br>Parameter 2: The number of bytes to transfer<br>Parameter 3… : The data in the case of a write operation. |
| **Results:** | For Write operation:<br>Byte 0: I2C address<br>Byte 1: The result code from the write operation<br>For Read operation:<br>Byte 0: I2C address<br>Byte 1: The result code from the read operation<br>Byte 2: Error code - reset<br>Byte 3…: The bytes read |
| **Example:** | ***I2C read operation***<br>To controller:<br>Hdr ID len cmd addr read n LRC<br>“ab” “00” “03” “81” “68” “01” “0f” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd addr read n<br>“ac” “00” “03” “81” “68” “01” “0f” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd addr reslt err … data …<br>“ab” “00” “12” “81” “68” “00” “00” “7e” “00” “00” “00” “80” “00” “00”<br>… LRC<br>“21” “00” “00” “00” “00” “04” “10” “0a” “77” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | Command: 14 or 16 for read<br>14+n or 16+2n for write<br>Reply:16+2n for read<br>14 for write |
| **Encoding:** | “ac00038168010f”<br>reply - “ab0012816800007e000000800000210000000004100a77” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware version of 6.1 and higher. |
| **Comments:** | This command is used for debugging purposes.<br>The input parameters:<br>Byte 0: I2C address – this is the “unshifted” I2C address of the target device (“68” for the DS1308 clock device, “02” for slave PIC I, “03” foe slave PIC Q)<br>Byte 1: R_W – the read/write flag, 1 = read, 0 = write<br>Byte 2: The number of bytes to transfer<br>Byte 3…: The data in the case of a write operation<br>The returned parameters have the following significance:<br>In the case of a “read” operation:<br>Byte 0: I2C address – the “unshifted” I2C address used for the read. |

| | |
|---|---|
| | Byte 1: the result code for this read operation.<br>Byte 2: Error code – this is the inclusive OR of the error codes detected since last reset. This command resets this parameter<br>Byte 3…: The data read<br>In the case of a write operation:<br>only the first two bytes (I2C address and result code) are returned. |

### 7.52 Bootload

| | |
|---|---|
| **Function:** | A function to invoke the boot loader facility. |
| **Function Code** | 0x0c |
| **Parameters** | Parameter 0: method – 0 = “manual” reset, 1 = use reset instruction |
| **Results:** | No results are returned – only ACK message |
| **Example:** | ***Bootload using “manual” reset***<br>To controller:<br>Hdr ID len cmd reset LRC<br>“ab” “00” “01” “0c” “00” “xx” <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd offset<br>“ac” “00” “01” “0c” “00” <CR><LF><br>Reply from controller:<br>Hdr ID len cmd LRC<br>“ab” “00” “00” “oc” “53” <CR><LF><br>In the case of a terminal Instruction with “ac”, we have the same reply. |
| **Bytes:** | Command: 10 or 12<br>Reply: 10 |
| **Encoding:** | “ac00010c00”<br>reply - “ab00000c53” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware version of 5.9 and higher. |
| **Comments:** | This command is used to initiate the boot load process.<br>The input parameters:<br>Byte 0: the method flag – The bootloader occupies high memory in the PIC and is called first when the PIC is reset or on startup. It can thus be invoked either by causing a reset instruction to be executed, or by branching to its start address directly. If the parameter sent with the command is a 1 then the reset method is used otherwise the program resets all registers to their reset states and branches to the bootloader. |

### 7.53 Bootload a slave (decoder) PIC

| | |
|---|---|
| **Function:** | A function to begin the bootloading process of a slave (decoder) PIC. |
| **Function Code** | 0x10 |
| **Parameters** | Parameter 0: The slave ID (not its I2C address) – 0 = I, 1 = Q<br>Bit 7: if set, a failure of the initial command to the slave is ignored |
| **Results:** | If the process is successfully started then only an ACK message is received<br>In the event of the I2C command to the slave failing then the following status information is returned:<br>Byte 0: I2C address ("02" for I decoder, "03" for Q decoder).<br>Byte 1: the result code for this read operation.<br>Byte 2: Error code – this is the inclusive OR of the error codes detected since last reset. Byte 3: Fail state<br>Byte 4: retries remaining |
| **Example:** | ***Bootload "I" decoder PIC***<br>To controller:<br>Hdr ID len cmd offset LRC<br>"ab" "00" "01" "10" "00" "xx" <CR><LF><br>or for terminal mode:<br>Hdr ID len cmd offset<br>"a**c**" "00" "01" "10" "00" <CR><LF><br>Reply from controller (on error):<br>Hdr ID len cmd addr status err state count LRC<br>"ab" "00" "05" "10" "02" "30" "30" "01" "00" "0f" <CR><LF><br>In the case of a terminal Instruction with "ac", we have the same reply. |
| **Bytes:** | Command: 10 or 12<br>Reply: 10 (20 on fail) |
| **Encoding:** | "ac00011000"<br>reply – "ab00001021<br>error reply – "ab00051002303001000f" |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware version of 6.1 and higher. |
| **Comments:** | This command is used for debugging purposes.<br>The input parameters:<br>Byte 0: the slave PIC ID. "00" for the "I" channel decoder, "01" for the "Q" channel decoder.<br>Bit 7 of Byte 0: is an "ignore error" flag. If it is set then failure of the initial command to the slave PIC does not abort the process and the controller PIC stays in the "download slave" mode.<br>In the event of the command failing the returned parameters have the following significance:<br>Byte 0: I2C address – the "unshifted" I2C address to which the command was sent.<br>Byte 1: the result code for this write operation.<br>Byte 2: Error code – this is the inclusive OR of the error codes detected since last reset. This command resets this parameter. |

| | |
|---|---|
| | Byte 3: Fail state – the state of the I2C state-machine when the last error was detected.<br>Byte 4: Retries left – the number of retries remaining when operation was aborted.<br>The controller PIC exits this mode through a reset when either an EOF Hex record has been processed or when a dummy special Hex record with record-type “FF” is received. The special record will be:<br>`:000000FF01` |

# 8 Decoder Instruction Set Descriptions

The commands described in this section are issued over the I2C bus, by the controller PIC to the decoder (slave) PICs. The instruction coders and parameters are of most use to developers of firmware for the two PICs. Access to these commands will either be built into ShowTags and the OEM SDK or can be accessed using the I2C write command to the controller.

Currently (2002-12-12, F/W versions C6.2 and D3.2) there is no formal protocol used between the controller and slave PICs. Future releases of the firmware will incorporate message length and LRC type parameters.

## *8.1 Load Status Info into I2C transmit buffer*

| | |
|---|---|
| **Function:** | Load status info onto I2C transmit buffer. |
| **Function Code** | 0x00 |
| **Description:** | Obtaining the status from one of the decoder PICs is a two step operation: first the slave PIC must be instructed to load this information into its transmit buffer, then the controller must issue a read, of the correct number of bytes, to collect the information.<br>This instruction initiates the first part – loading the data into the transmit buffer. |
| **Parameters** | none |
| **Results:** | byte 0: I2C error status on slave<br>byte 1: header byte used by slave when transmitting a tag ID (I/Q identifier).<br>byte 2: F/W version<br>byte 3: configuration byte<br>byte 4: power-up counter<br>byte 5: noise count<br>byte 6: STKPTR as saved at last restart<br>byte 7: RCON as saved at last restart<br>byte 8: TOSH as saved at last hang-trap<br>byte 9: TOSL as saved at last hang-trap<br><br>These results must be fetched using the I2C read function. |
| **Example:** | ***using the Controller “Write I2C” function to retrieve status information***<br>First issue command to load the slave’s transmit-buffer.<br>To controller:<br>Hdr ID len cmd addr wr cnt cmd<br>“ac” “00” ”04” “81” “02” “00” “01” “00” <CR><LF><br>Notes: 1 2<br>***Notes:***<br>1. the parameters for the I2C write command are described in the Protocol manual, included as Appendix B above (see section 7.51 above).<br>Briefly, they are:<br>02 – the I2C address of slave PIC 0 (decoder “I” channel).<br>00 – write command.<br>01 – number of bytes to write.<br>00 – the slave command (load status to tx buffer – what this section is all about).<br>2. The actual I2C command is the payload of the write I2C command we are using to |

indirectly access the slave PIC.

Reply from controller:

```
Hdr   ID   len  cmd  addr  stat
“ab”  “00”  “02”  “81”  “02”  “00”  “ed”
```

The reply is associated with the write I2C command and does not relate to a result code from the slave PIC.

To actually see the data loaded into the I2C tx buffer, one must issue the read I2C command.

```
Hdr   ID   len  cmd  addr  read  cnt
“ac”  “00”  “03”  “81”  “02”  “01”  “08”
```

Reply from controller:

```
Hdr   ID   len  cmd  addr  stat  err   s-err  hdr  vers  conf  pwr  noise
“ab”  “00”  “0b”  “81”  “02”  “00”  “00”  “00”  “6e”  “32”  “01”  “03”  “00”
NU  NU  LRC
“00”  “00”  “c1”
```

This reply contains the following information:

| byte | parameter | value |
|---|---|---|
| 0 | I2C error | 00 – no error |
| 1 | header | 6e – decoder “I” channel |
| 2 | F/W version | 32 – version 3.2 |
| 3: | config | 01 – 64kb/s tags |
| 4: | power-up counter | 03 – power-up has occurred three times |
| 5: | noise count | 00 – noise count is zero |
| 6: | dummy | not used |
| 7: | dummy | not used |

| | |
|---|---|
| **Bytes:** | 0 |
| **Encoding:** | “ac00048102000100”<br>reply – “ab0002810200ed”<br>“ac000381020108”<br>reply – “ab000b81020000006e320103000000c1” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in controllers with firmware version of 3.1 and higher. |
| **Comments:** | |

## 8.2 Set tag Configuration

<table>
<tr><td><b>Function:</b></td><td colspan="3">Sets the decoder configuration, including the baud rate of tags to be decoded.</td></tr>
<tr><td><b>Function Code</b></td><td colspan="3">0x06</td></tr>
<tr><td><b>Description:</b></td><td colspan="3">From version 3.2 of the decoder firmware, the baud rate of tags to be decoded is no longer hard coded into the firmware, but can be changed during operation.<br>The configuration word is saved in EEPROM on the PIC itself. Users of this feature must be aware of the write “endurance” which can be as low as 1000 cycles at temperatures above 85°C.</td></tr>
<tr><td><b>Parameters</b></td><td colspan="3">Parameter 0: Bit-mask configuration options</td></tr>
<tr><td></td><td><b>Bit</b></td><td><b>Item</b></td><td><b>Description</b></td></tr>
<tr><td></td><td>7-2</td><td>reserved</td><td>ignored</td></tr>
<tr><td></td><td>1-0</td><td>Baud rate</td><td>00 – 32kb/s (Not yet supported)<br>01 – 64kb/s<br>10 – 128kb/s (Not yet supported)<br>11 – 256kb/s</td></tr>
<tr><td></td><td colspan="3"></td></tr>
<tr><td><b>Results:</b></td><td colspan="3">byte 0: I2C error status on slave<br>byte 1: header byte used by slave when transmitting a tag ID (I/Q identifier).<br>byte 2: F/W version<br>byte 3: spare</td></tr>
<tr><td><b>Example:</b></td><td colspan="3"><b><i>using the Controller “Write I2C” function to retrieve status information</i></b><br>First issue command to load the slave’s transmit-buffer.<br>To controller:<br>Hdr ID len cmd addr wr cnt cmd<br>“ab” “00” ”05” “81” “02” “00” “02” “06” “01” “19” &lt;CR&gt;&lt;LF&gt;<br>Notes: 1 2<br><b><i>Notes:</i></b><br>1. the parameters for the I2C write command are described in the Protocol manual, included as Appendix B above (see section 7.51 above).<br>Briefly, they are:<br>02 – the I2C address of slave PIC 0 (decoder “I” channel).<br>00 – write command.<br>01 – number of bytes to write.<br>06 – the slave command (set tag configuration – what this section is all about).<br>2. The actual I2C command is the payload of the write I2C command we are using to indirectly access the slave PIC.<br>Reply from controller:<br>Hdr ID len cmd addr stat<br>“ab” “00” “02” “81” “02” “00” “ed”<br>The reply is associated with the write I2C command and does not relate to a result code from the slave PIC.</td></tr>
<tr><td><b>Bytes:</b></td><td colspan="3">0</td></tr>
</table>

| | |
|---|---|
| **Encoding:** | "ab0005810200020601 19"<br>reply – "ab0002810200ed" |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in decoders with firmware version of 3.2 and higher. |
| **Comments:** | |

### 8.3 Initiate Bootload sequence

| | |
|---|---|
| **Function:** | Bootload. |
| **Function Code** | 0x08 |
| **Description:** | . |
| **Parameters** | none |
| **Results:** | byte 0: I2C error status on slave<br>byte 1: header byte used by slave when transmitting a tag ID (I/Q identifier).<br>byte 2: F/W version<br>byte 3: spare |
| **Example:** | ***using the Controller “Write I2C” function to retrieve status information***<br>First issue command to load the slave’s transmit-buffer.<br>To controller:<br>Hdr ID len cmd addr wr cnt cmd<br>“ac” “00” ”04” “81” “02” “00” “01” “00” <CR><LF><br>Notes: 1 2<br>***Notes:***<br>1. the parameters for the I2C write command are described in the Protocol manual, included as Appendix B above (see section 7.51 above).<br>Briefly, they are:<br>02 – the I2C address of slave PIC 0 (decoder “I” channel).<br>00 – write command.<br>01 – number of bytes to write.<br>00 – the slave command (load status to tx buffer – what this section is all about).<br>2. The actual I2C command is the payload of the write I2C command we are using to indirectly access the slave PIC.<br>Reply from controller:<br>Hdr ID len cmd addr stat<br>“ab” “00” “02” “81” “02” “00” “ed”<br>The reply is associated with the write I2C command and does not relate to a result code from the slave PIC.<br>To actually see the data loaded into the I2C tx buffer, one must issue the read I2C command.<br>Hdr ID len cmd addr read cnt<br>“ac” “00” “03” “81” “02” “01” “04”<br>Reply from controller:<br>Hdr ID len cmd addr stat err s-err hdr vers NU LRC<br>“ab” “00” “07” “81” “02” “00” “00” “00” “6e” “32” “00” “12”<br>This reply contains the following information:<br>byte parameter value<br>0 I2C error 0 – no error<br>1 header 6e – decoder “I” channel<br>2 F/W version 32 – version 3.2<br>3 spare not used |

| | |
|---|---|
| **Bytes:** | 0 |
| **Encoding:** | “ac00048102000100”<br>reply – “ab0002810200ed” |
| **Operations:** | Controller → reader |
| **Availability:** | This command is available in decoders with firmware version of 3.2 and higher. |
| **Comments:** | |

# 9 Reader Types

## 9.1 3 PIC reader (Legacy device)

Two PICs are used to perform the decoder function (one per channel (I/Q)) and a third is used as the controller.
A PIC 18F452 is used as the controller with limited program and data memory.
The decoder PICs are significantly underutilised while the controller PIC is reaching its limit as far as functions requiring data storage.

## 9.2 Handheld (HH) reader

This reader uses a single PIC to do both the decoding function as well as the controller functions.
A PIC 18F452 device is used and has limited program and data memory. Since both decoder and controller functions share the same device there is an even greater pressure on memory.

**WARNING**

**In the single PIC HH and "Prox" versions the configuration is saved in EEPROM on the PIC itself.**
**Users must be aware of the write “endurance” which can be as low as 1000 cycles at temperatures above 85°C.**

## 9.3 FPGA reader

Uses a PIC as the controller and an FPGA decoder.

## 9.4 Dual DSP reader

Uses an ARM as the controller and an FPGA decoder/DSP.

# 10 Support

## Technical Assistance

| | | |
|---|---|---|
| Willie Hofmeyr: Software design | +27 12 345-9520 x 219 | willie.hofmeyr@ipico.com |
| Marius van Dyk: System Design | +27 12 345-9520 x 212 | marius.van.dyk@ipico.com |

IPICO online http://www.ipico.com/

## Address

IPICO South Africa
Unit 5B Prospect Close
311 Regency Drive
Route 21 Corporate Park
Irene
0157
South Africa

PO Box 73180
Lynnwood Ridge
0040
South Africa

Tel: +27 12 345 9520
Fax: +27 12 345 5834