



## IRCore MKII Protocol Specification

**Revision #1.0**

April 23, 2004

### Ravens Ridge Design (R2DI) Offices

2575 NE Kathryn St., Suite 31  
Hillsboro, OR 97014  
503-615-8215

1008 Paseo Del Pueblo Sur #79  
Taos, NM 87571  
505-758-8314

Website: [www.r2di.com](http://www.r2di.com)

Contact: [sales@r2di.com](mailto:sales@r2di.com) or 503-615-8215

**- Confidential -**

**IMPORTANT!** Contents of this document are considered proprietary and confidential information and are the property of R2DI, LLC and is protected under the copyright laws of the United States of America. Reproduction and/or distribution of this document without consent of R2DI, LLC is prohibited.

## **IMPORTANT NOTICE**

*The contents of this proposal are confidential and contain proprietary information including trade secrets of R2DI, LLC ("The Company") and may only be used for the purpose of evaluating the acceptance of this statement of terms and conditions. Neither this document, nor any parts contained herein, may be reproduced or disclosed to any person under any circumstances without the express written permission of R2DI, LLC.*

### **R2DI, LLC**

*Corporate Offices:  
2575 NE Kathryn Street, Suite 31  
Hillsboro, Oregon 97124  
(503) 615-8215*

*Engineering Offices:  
1008 Paseo Del Pueblo Sur #279  
Taos, NM 87571-6412  
505-758-8314*

Ravens-Ridge Design, Inc. and R2DI are trademarks of R2DI, LLC All other names are trademarks of their respective manufacturers.

### **Document Control**

<b>Date</b>	<b>Revision</b>	<b>Description</b>
April 23, 2004	Revision 1.0	First Release

## 1. Introduction

The R2DI IRCore is a microprocessor based module for the management if IR data. It is used to learn, analyze, store, and playback IR data. It is available as an embedded module, or in stand-alone form with several standard interfaces.

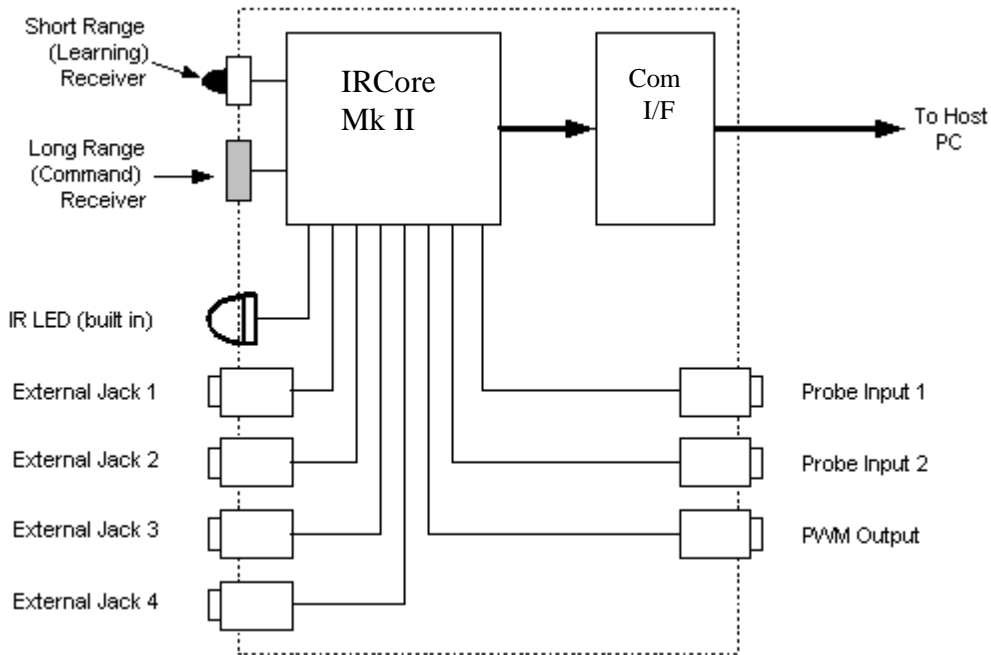


Figure 1: Typical IRCore Block Diagram

### 1.1 Features

- Wide Band IR detector captures modulated data from 10kHz to 700 kHz
- Narrow Band IR detector for command decoding
- ASCII command strings with multiple line termination options
- 1Mbit Non-Volatile memory on-board command storage
- Standard versions available for RS-232, USB, PCI and Ethernet

## **1.2 Interface Details**

### IR Command Capture Support

- Broadband IR Receiver for universal IR command capture
- Modulation timing resolution of 1uSec
- Supports "Ultra-Monochromatic" pin diodes that prevent interference from compact fluorescent lighting or plasma TV's

### IR Command Decoding Support

- Narrowband IR Receiver for protocol specific decoding
- Supports long range reception of specific remote control
- Remote commands are decoded to specific protocol codes

### IR Command Storage

- IRCore supports storing IR data in 1Mbit, on-board non-volatile memory
- The IR data is organized into a database format, for use by the protocol.

IR commands are referenced by Device ID and Command ID. Devices can hold multiple commands. Commands **MUST** be associated with a Device. This is the industry-standard method of organizing IR data for use by applications, and how the IRCore supports these features.

### IR Command Transmission

- Frequency Agile IR Emitter ports
- Multiple output ports are provided
- Port functions can be software-selectable between output port and probe input

### Device Status Probes

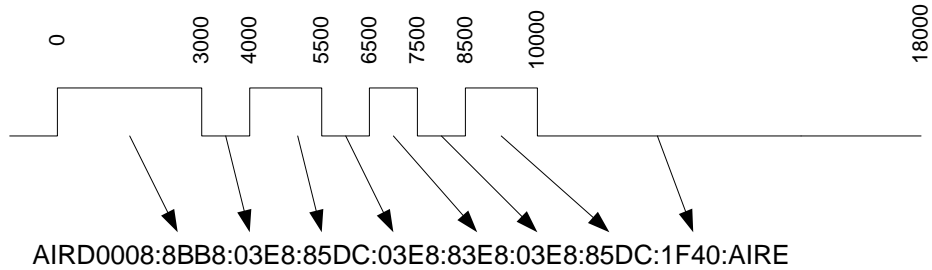
- Probe inputs support standard sensing devices
- Probe status reports and polling supported by command protocol
- Port functions can be software-selectable between output port and probe input

Command Protocol

- Captured IR Commands can be reported to the host communications port
- Captured IR Commands can be stored in on-board, non-volatile memory, referenced by Device ID and Command ID
- Protocol supports transmitting from IR data sent to the communications port by the host
- Protocol supports transmitting IR data stored in on-board, non-volatile memory
- IR Logic Analyzer Mode captures all IR data over an approximately 14 second period and reports it in a raw format to the serial port
- Protocol can return Button ID's for known IR protocols
- Enabled Probe State Changes are reported asynchronously to the host

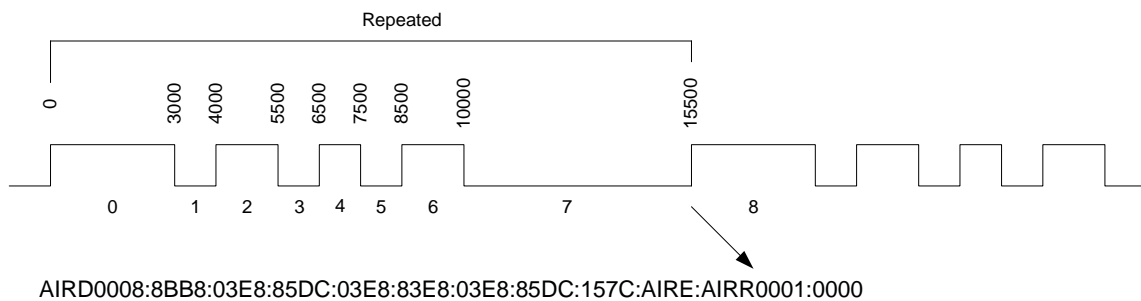
## 2. IR Data Description Protocol – AIR Format

R2DI uses a description protocol to describe an IR event. The core protocol consists of a data starting command, timing event tokens, and a data ending command. There are several modifiers that can be added to the base protocol to describe complex events. The protocol can best be explained with an example:



AIRD is the opening token of the description. The next Hex word value indicates that there are 8 transitions in the description. The first event token is 8BB8. The top bit is a 1 so it's an on event. The lower 15 bits indicate 0xBB8 which is 3,000 uSec. The next event is 03E8. The top bit is a 0 so it's an off event of 1,000 uSec duration. Events of the same type, on or off, may be combined to make event times longer than 32,767 uSecs. After all the transition events are done, the closing token is AIRE which tells the parser the description is complete.

One common method of error avoidance in Infrared communications is to repeat the commands. Most IR descriptions just capture a number of these repeats to support this method. R2DI's protocol allows the repeating of a command by appending the AIRR command to the AIR description string.

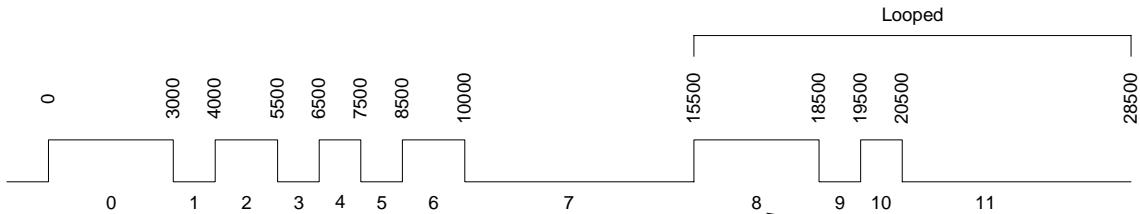


The first hex word value of the repeat command is the repeat count. The delay between repeats is given by the final event of the repeated section. In this case the token is 157C which is a delay of 5,500 uSecs. This means the repeated section will repeat after 5,500 uSecs, one time.

The second hex word of the repeat command is the loop index. This indicates the zero based index of the event to begin looping. The end of looping is the last event in the description. The count of the looping is given in the first hex word value of the AIR

command. In this example, the repeat count of 0 will mean that the loop section will get played once.

If the repeat value is set to 0xFFFF the repeating or looping will continue until interrupted by another command.



AIRD000C:8BB8:03E8:85DC:03E8:83E8:03E8:85DC:157C:8BB8:03E8:83E8:1F40:AIRE:AIRR0000:0008

The reduction of the IR data to these descriptions is managed by the control processor of the IRCore. Many rules are run against the captured signal to give the most data reduction possible.

When the IRCore is in learning mode, the Learn LED will flash rapidly to indicate readiness to learn. The light goes out when learning and simplification is complete.

### 3. IRCore Commands

The commands for the IRCore fall into three categories:

1. The first controls the Learning, Storing, Reporting, and Playback of IR information.
2. The second set of commands are used to adapt the IRCore to behave in a particular way. Most are never used, but some users will want to change the line termination or suppress the insertion of colons to save space in their string tables.
3. The final group of commands are used to describe infrared data to the IRCore.

#### 3.1 Learning, Storing, and Reporting Commands

Description	Prefix	Cmd	Param0	Param1	Param2
Learn IR <sup>1</sup>	AIR	T	<CR> <sup>2</sup>		
Learn Then Store IR <sup>3</sup>	AIR	T	wDevice <sup>4</sup>	wCmd	<CR>
Display Active Devices and Commands <sup>5</sup>	AIR	T	?	<CR>	
Xmit a stored value <sup>67</sup>	AIR	J	wDevice	wCmd	<CR>
Read Data <sup>5</sup>	AIR	N	wDevice	wCmd	<CR>
Delete Device <sup>8</sup>	AIR	K	wDevice	<CR>	
Delete Command <sup>9</sup>	AIR	K	wDevice	wCmd	<CR>
Query Emitter enabled <sup>10</sup>	AIR	O	?	<CR>	
Set Emitters Enabled	AIR	O	wEmits	(CR) <sup>2</sup>	
Logic Analyzer Start <sup>11</sup>	AIR	C	<CR>		
Set Probes Enabled <sup>12</sup>	AIR	P	wProbes	<CR>	
Query Probe State <sup>13</sup>	AIR	P	?	<CR>	

<sup>1</sup> AIRT<CR> will respond with the learned IR data in AIR format.

<sup>2</sup> The reference <CR> is a required line termination; (CR) shows an optional line termination.

<sup>3</sup> AIRTwDevice:xCmd<CR> will respond with ACK when learning is complete.

<sup>4</sup> A lower case w at the start of a variable name indicates it is a 16 bit HEX value.

<sup>5</sup> AIRT?<CR> will list the devices and commands loaded in memory. The format is the first value on the line is the device ID in 16 bit HEX, followed by the command Ids in 16 bit HEX. Each device gets its own line. An example would be: 0000 – 0000, 0001, 0002, 0005

This would mean that device 0 has 4 commands stored with it. The command numbers are 0, 1, 2, and 5. When IR data is stored over an existing IR command the old data is lost.

<sup>6</sup> If a read or transmit command references a non-existent device-command you will receive a RANGE ERROR response.

<sup>7</sup> When you request a Xmit a Stored Value it will be sent out on whatever the default emitters are at that moment. It is better to add a specific emitter enable command in front of this command. The form would be AIROwEmit:AIRJwDevice:wCmd<CR>.

<sup>8</sup> If a device is deleted all commands attached to it are also deleted.

<sup>9</sup> Specific commands may be deleted using the Delete Command command.

<sup>10</sup> These two commands control which emitters are active for an IR transmission.

<sup>11</sup> The IRCore will collect all IR events that occur until its buffer is full. No modulation frequency information is calculated in logic analyzer mode.

<sup>12</sup> These two commands support the enabling and querying of probe states. The wProbes parameter is a bit mapped word where each bit represents a probe. A bit set to a 1 enables the probe reporting. The probes are debounced with a 250 mSec period.

<sup>13</sup> When an enabled probe changes state a report is sent asynchronously to the host. The report takes the form AIRPwProbe:wState. The probe number is a zero based counting of the supported probes. The state indicates the new state. When a probe is enabled it will always send a state change message indicating the present state of the probe.

### 3.2 Device Operation Control Commands

Description	Prefix	Cmd	Param0	Param1	Param2
Identify <sup>1</sup>	AIR	G	<CR>		
Query Version <sup>2</sup>	AIR	V	?	<CR>	
Control Rule Reporting <sup>3</sup>	AIR	U	B	<CR>	
Query System Variable <sup>4</sup>	AIR	W	B	<CR>	
Set System Variable <sup>4</sup>	AIR	W	B	w	<CR>
Set Termination <sup>5</sup>	AIR	L	B	<CR>	
Control ':' Insertion <sup>6</sup>	AIR	I	B	<CR>	
Jump to Bootloader <sup>7</sup>	AIR	S	<CR>		
Set Everything to Factory Defaults <sup>8</sup>	AIR	Q	<CR>		

### 3.3 Data Commands

Description	Prefix	Cmd	Param0	Param1	Param2
Query Emitter enabled	AIR	O	?	<CR>	
Set Emitters Enabled	AIR	O	wEmits	(CR)	
Set Modulation Frequency <sup>9</sup>	AIR	X	wFreq	(CR)	
Start Data Field <sup>10</sup>	AIR	D	wCount		
End Data Field	AIR	E	(CR)		
Repeat Data	AIR	R	wCount	wLoop	(CR)
Store Data <sup>11</sup>	AIR	M	wDevice	wCmd	<CR>

<sup>1</sup> This command is used to verify the presence of an IRCore. The IRCore will respond with the product id. In the case of the IR500 it will respond with "IR500".

<sup>2</sup> This command shows the version of the firmware.

<sup>3</sup> This turns on and off the reporting of the rule that has kept the learn from completing.

<sup>4</sup> These two commands support the reading and modifying of system variables. These values control the internal operations of the IRCore. A table of the values is included in this document.

<sup>5</sup> This allows a user to match the line terminations sent by the IRCore to the system software. The values to use are:

- CR\_THEN\_LF            0
- CR\_ONLY                1
- LF\_ONLY                2
- LF\_THEN\_CR            3

<sup>6</sup> To minimize the size of AIR description strings returned by the learning process the ':' character used to separate values may be dropped. This makes the strings very difficult for humans to read so the default is on.

<sup>7</sup> This command turns off the firmware signature and causes the IRCore to run only the bootloader. The IRCore firmware will not boot again until the firmware has been once more digitally signed using the "BIRS" command of the bootloader. This behavior protects against interrupted firmware loads causing the IRCore to be rendered useless.

<sup>8</sup> If all else fails and you need to get back to factory defaults this command will do just that. The command takes a while to complete since the IR storage is cleared as well.

<sup>9</sup> This sets or reports the frequency of modulation for the IR signal. The value is in hundreds of hertz. A value of 386 would give a modulation frequency of 38,600 hertz.

<sup>10</sup> These three commands make up a base IR description.

<sup>11</sup> When this command is appended to an AIR description it is stored into the Device, Command database instead of transmitting the value as IR. The form this would take is:

```
AIRX016A:AIRD0008:8BB8:03E8:85DC:03E8:83E8:03E8:85DC:157C:AIRE:AIRR0000:0000:AIRM0000:0000<CR>
```

>  
This would cause the described IR data to be stored into Device 0, Command 0 of the non-volatile memory.

## 4. IRCore System Variables

Name of the variable	Num	Hex	Default	Definition
DEBUG_STATE	0	// 00 //		Diagnostic report mode - Do not modify
MIN_MODULATION	1	// 01 //	10	Lowest allowable modulation frequency
MAX_MODULATION	2	// 02 //	500	Highest allowable modulation frequency
IR_IDLE_AT_INIT	3	// 03 //	16000	Wait this long to be sure receiver is Idle
LONGEST_PULSE_AT_INIT	4	// 04 //	6000	Pulses longer than this cause re-initialization
LONGEST_START_PULSE	5	// 05 //	24000	Pulses longer than this cause re-initialization
LONGEST_DATA_IDLE	6	// 06 //	8000	Pause longer than this cause end of data collection
LONGEST_DATA_ACTIVE	7	// 07 //	8000	Pulses longer than this cause re-initialization
LONGEST_BETWEEN_PACKETS	8	// 08 //	53000	Maximum time to wait for a repeat is LONGEST_BETWEEN_PACKETS + LONGEST_DATA_IDLE
IR_IDLE_AT_END	9	// 09 //	10000	10 mSec to be sure its Idle
BIT_CHANGE_TIMEOUT	10	// 0A //	1000	Loop to escape stuck in frequency detect due to lack of pulses
MIN_START_PULSE	11	// 0B //	100	Smallest allowable start pulse in uSec
MIN_TRANSITIONS	12	// 0C //	7	There must be at least this many transitions in the first buffer
MIN_FOLLOWUP_TRANSITIONS	13	// 0D //	4	There must be at least this many transitions in the second buffer, ala Sharp TV
MOD_CAPTURES	14	// 0E //	2	How many captures for modulation detection averaging
MAX_MOD_TIME	15	// 0F //	150	Maximum number of uSec in a mod clock duration.
MAX_REPEAT	16	// 10 //	10	This is on learn only, playback may repeat up to 255
DEFAULT_REPEAT	17	// 11 //	0	We default to zero
COMPARE_THRESHOLD	18	// 12 //	75	Durations must be within this many uSec to be the same
STUCK_MAX_COUNT	19	// 13 //	50	Number of 10 mSec intervals to remain stuck
STUCK_CMD_MAX_COUNT	20	// 14 //	5000	Number of mSecs to allow it to be stuck before resetting
RAW_PREAMBLE	21	// 15 //	1	If a one, display '.' while waiting in AIRC
ACK_AFTER_XMIT	22	// 16 //	1	If zero, send ACK as soon as xmit data received, else only when sending complete

To access these variables:

AIRW06? - Reads LONGEST\_DATA\_IDLE value

AIRW06:134F - Sets the value of LONGEST\_DATA\_IDLE to 0x134F

## **5. IR500e Notes:**

1. The IR300's IP address is assigned via DHCP
2. Once the IP address is assigned, the IR300 communicates using a standard telnet port (TCP port 23)
3. The IR300 can be controlled via any telnet client (such as PuTTY (<http://www.chiark.greenend.org.uk/~sgtatham/putty/download.html>))
4. A driver that supplies a C and a C++ API is completing development to allow the creation of user application under Windows.
5. Updated documentation is also in progress.
6. The RS232 port is currently only used to display the IP address assigned by DHCP. Its settings are 115.2Kbaud, No parity, 8 data bits, 1 stop bit, no handshaking.

---

### **Ravens Ridge Design (R2DI) Offices**

2575 NE Kathryn St., Suite 31  
Hillsboro, OR 97014  
503-615-8215

1008 Paseo Del Pueblo Sur #79  
Taos, NM 87571  
505-758-8314

Website: [www.r2di.com](http://www.r2di.com)

Contact: [sales@r2di.com](mailto:sales@r2di.com) or 503-615-8215