TORREY PINES SCIENTIFIC

OPERATING MANUAL

EchoTHerm™
DIGITAL, ELECTRONIC CHILLING/HEATING DRY BATH
MODEL RIC20, RICXR and RIC20XT
version 2.3 and later
(Backward compatible with v1.0 and later)

DOCUMENT NUMBER RIC20-100
Revised November 7,2024

TORREY PINES SCIENTIFIC, INC.
2713 Loker Ave. West
Carlsbad, CA 92010

TELEPHONE: (760)-930-9400
TOLL FREE: (866)-573-9104
FAX: (760)-930-9480
E-Mail: info@torreypinesscientific.com

Web site: www.torreypinesscientific.com

Table of Contents

I. INTRODUCTION

Congratulations on your purchase of a *EchoTherm*™ Remote Controlled Chilling/Heating Dry Bath Model RIC20, RIC20XT or RIC20XR Please read the instructions carefully to insure that you receive the maximum benefit from it. Also, be sure to go to our web site and register your unit for warranty coverage. When users in the USA and Canada do they will receive a Torrey Pines Scientific, Inc. T-Shirt free.

II. WARRANTY

Torrey Pines Scientific warrants this product to be free from defects in material and workmanship for a period of one year from the date of purchase. If repair or adjustment is necessary and has not been the result of abuse or misuse within the one year period, please return---freight prepaid---and correction of the defect will be made without charge. Out of warranty products will be repaired on a charge basis.

III. RETURN OF ITEMS

Authorization must be obtained from our Customer Service Department before returning items for any reason. When applying for authorization, please include data regarding the reason the items are to be returned. For your protection, items must be **carefully packaged** to prevent damage in shipment and **insured** against possible damage or loss. Torrey Pines Scientific will not be responsible for damage resulting from careless or insufficient packing. A 15% restocking charge will be made on all unauthorized returns.

Note: Torrey Pines Scientific reserves the right to make improvements in design, construction, and appearance without notice.

IV. LABELS

There are various labels on the body of this unit. Listed below are the labels and their meanings.

This symbol means: ATTENTION. The INSTRUCTION MANUAL IS TO BE CONSULTED FOR FURTHER INFORMATION

This symbol means: WARNING. HOT SURFACE.

This symbol means: GROUND OR EARTH CONNECTIONS

V. CAUTIONS

HEATER PLATE SURFACE

RIC20 units are capable of chilling and heating the plate/block surface form -10ºC to 100ºC on the RIC20, from -20ºC to 100ºC on the RIC20XT, and from -10ºC to 110ºC on the RIC20XR. The upper temperature of 100ºC to 110ºC (212ºF to 230ºF) is hot enough to burn the skin if touched. **Use extreme caution at all times.** Never leave your unit accessible to others when it is hot. Never touch the plate surface unless you are sure it is cold.

ELECTRICAL

The RIC20 and RIC20XR cooling/heating modules run off 12 volts dc 5.0 amp for the RIC20 and 10.0 amp for the RIC20XR and RIC20 XT These instruments are supplied with a universal power supply that can take inputs from 100 to 260 volts AC "10%”. The units are supplied with an AC input cord for the power supply. Be certain to use a line cord with the same rating and of the same type as the one supplied by themanufacturer. Use the normal care and precaution one would use with any electricalappliance. **Be certain to use a line cord of the same type and rating as the one supplied with this unit. Note: all fuses are fast blow.**

Bien vérifier que le cordon utilisé est du même type que celui livré avec lunité. Note: tous les fusibles sont à action rapide.

VI. GENERAL DESCRIPTION

The Torrey Pines Scientific Models RIC20 and RIC20XR and RIC20XT are Peltier driven chilling/heating dry baths. They come with a universal power supply and the chilling/heating module. These units have only one moving part, the DC fan that cools the unit. Everything else is solid state and should last years without problem. All functions of the unit are accessed via the RS232 I/O port. The Programming Manual is provided at the end of this manual and has all instructions necessary for running the unit. There is no display or keypad on the units.

! IMPORTANT POWER CONSIDERATIONS

The RIC20, RIC20XT & RIC20XR cooling/heating modules can be used as a standalone benchtop instrument but are designed primarily for integration into automated processing cells. In such systems, managing energy consumption during startup is often a concern, and the RIC20 units can contribute significantly to the initial load. For instance, when a RIC20 unit is powered on with a target temperature above or below ambient, it quickly reaches its peak current draw (over 4 amps for the RIC20 and over 8 amps for the RIC20XR or RIC20XT). However, as the unit approaches its target temperature, the current draw decreases significantly unless the target is near the maximum or minimum temperature limit.

When multiple RIC20 units are installed and powered on simultaneously, the initial combined current draw can place a substantial load on the cell's power supply. To address this, a staggered power-up sequence can help distribute the energy demand more evenly. Since the highest current draw occurs as each unit begins reaching its target temperature, staggering the activation allows each unit's current draw to drop before the next unit powers up. Alternatively, setting all units to Idle Mode or to a target temperature equal to ambient can significantly minimize initial current draw. For further load management, target temperatures can be set sequentially—adjusting the target for each unit only after the previous one nears or reaches its goal.

To ensure RIC20 units draw minimal current during startup, consider the following approaches:

1. **Idle Mode**: This mode effectively disables heating and chilling by removing the target temperature, keeping current draw minimal regardless of ambient conditions. Units can be set to start up in Idle Mode by sending a command to put the unit into Idle Mode prior to power down or by sending a command to configure the unit to always start up in Idle Mode regardless of the power down target temperature or mode. For more details see command i/I and x/X in the RIC20 Program Guide.

2. **Ambient Target**: Theoretically, setting target temperatures to match ambient can also minimize current draw if the plate and ambient temperatures are close, which is often the case in cold startups. However, in scenarios where the unit is restarted after a cell shutdown the plates will likely not be at ambient so start up in Idle Mode will be much more reliable and is highly recommended as the initial condition for sequential startups of multiple units.

HEATER/CHILLER PLATE

The plate surface is a very flat aluminum plate designed for good contact with any flat surfaced item placed on it. It is supplied with magnets that match magnets in the sample blocks that hold the blocks firmly in place for use with robotic systems. Units are available on special order with threaded holes in the plate that will match holes in the blocks for screwing down the sample blocks. The plate size of the RIC20, RIC20XT and RIC20XR is 2.875" (7.3 cm) x 4.375" (11.1 cm). It is designed to accommodate all the aluminum blocks supplied by Torrey Pines Scientific. The temperature of the plate is sensed by a platinum RTD mounted under the plate. The computer in the unit compares the plate temperature with the target temperature and instructs the Peltier module to heat or chill the plate as required to reach and hold the target temperature.

TEMPERATURE STATE LED

There is a green LED provided which flashes when the unit is powered and continues to flash even when a target temperature has been set. The LED will stop flashing and be constantly illuminated when the target temperature is reached and is stable to within +/- 0.2ºC for more than 60 seconds.

TIMER

All units have a countdown timer reading in hours, minutes, and seconds. It can be set to a maximum of 99:59:59. When the timer reaches 00:00:00 it will count up to show how much time has elapsed since the timer timed-out

CALIBRATION

The units have been calibrated in the factory to NIST standard prior to shipment. In the event that the user wants to calibrate to a different standard, the ability to recalibrate each of the two calibration points is provided. Additionally, the ability to easily restore the factory calibration values is included. See section “IX Temperature Calibration” and the attached Programming Manual at the end of this document for more details.

NON-VOLATILE MEMORY

The units are equipped with EEPROM memory that will store user settings such as the last set point, MENU options, user changed calibration points, and program steps. These stored values will be recalled each time the unit powers up to maintain the configuration of the previous session.

VII. RIC20 POWER AND RS232 CONNECTOR

Figure 5: RIC20 RS232 and Power Input Connectors

The RIC20 power is supplied using the 12VDC, 5A (10A for RIC20XR) power supply provided via the 2.1mm barrel connector on the input power cable as shown in Figure 5.

Serial communication is wired by plugging the RS232 serial cable provided into the DB9 connector on the second cable shown in Figure 5. Com connection to the PC or equivalent is by either direct connection to the DB9 connector on the PC or by connecting to a DB9 to USB dongle

The RS232 cable provided is straight through wiring (pin 1 to pin1, 2 to 2, etc.). Pins 2, 3 and 5 on the DB9 connector are the only pins used. It operates at 9600 baud, 1 stop bit, no parity. No handshake hardware or software is necessary. All communications settings and queries are done using ASCII characters with carriage return as the terminating character. **See instructions at the end of this manual for the Serial Command Set.**

Note: To comply with CE and to avoid possible EMI radiation from the RS232 cable, use a shielded cable.

VIII. SET UP PARAMETERS

<u>SET UP PARAMETERS</u>

1. Ambient operating room temperature range is from 5ºC to 40ºC.

2. Maximum altitude of operation should not exceed 2000 meters.

3. Maximum ambient operating relative humidity should not exceed 80% at 31ºC decreasing linearly to 50% relative humidity at 40ºC.

<u>SET UP INSTRUCTIONS</u>

1. Place the unit securely on the surface.
2. Plug the power supply into a properly grounded, 3-wire outlet of proper voltage.
3. Plug the power supply cable into the jack on the chilling/heating module.
4. Place the sample block on the plate surface.
5. Connect the RS232 cable to the serial port on the computer. The LED on the inside of the unit will illuminate and be blinking. The LED will stop blinking when the plate temperature is within 1C of the Set Point temperature for 60secs.

Note: Do not use this equipment in any manner not specified by the manufacturer.

<u>ENVIRONMENTAL INFORMATION</u>

1. This unit is for installation category II.

2. This unit is rated pollution degree 2.

IX. TEMPERATURE CALIBRATION

The temperature calibration set into the units is stable and will hold without drifting. However, our standards for temperature measurement may not be the same as the users. Therefore, the RIC20 & RIC20XR are designed to be calibrated in the field by the user. Calibration instructions are provided in Programming Manual included at the end of this document.

Note: The calibration is two-point for optimum accuracy. The unit is calibrated at the factory at 5.0°C and 70.0°C.

To calibrate the units at a particular temperature, set the unit to go to that temperature. Give the unit time to equilibrate, usually 20 minutes. Now measure the plate temperature (or the block or the samples to be controlled) using an electronic thermometer with a good surface temperature probe (a good immersion probe if calibrating a solution in a sample container).

*NOTE: PLATE SURFACE MEASUREMENTS ARE DIFFICULT TEMPERATURE MEASUREMENTS TO MAKE ACCURATELY. Use temperature probes specifically designed for surface temperature measurement and a digital meter that can accurately measure to 0.1C. Ensure that the unit and the measurement device have had sufficient time to stabilize at the new calibration temperature prior to entering in Measured Temperature Value. If the temperatures are not stable when the measured temperature value is entered, significant display and unit performance errors may result. Check with the factory if help is needed. Also, a calibration kit is available.

X. TEMPERATURE MEASUREMENT ERROR CODES

If an issue occurs with the temperature measurement circuitry or with a calibration value, the unit will return the error code in place of the returned plate temperature value when the “p” command is sent. If an error is detected, the unit will go into Heater OFF mode (power to the heater plate will be turned off). Table 2 below lists the error codes, the likely cause, and recommended action to address the error. Calibration errors (cal0-cal4) are easily addressed by performing the action to reset the appropriate calibration points to the factory settings. The unit can then be re-calibrated if necessary.

Error Code	Possible Cause	Recommended Action
RTDo	The RTD Sensor is not connected or has failed	Contact Torrey Pines Scientific torreypinesscientific.com/contact/
RTDs	The RTD Sensor has shorted or has failed	
cal1	Low Cal Point out of range	Reset Low Cal Point (h<CR>)
cal2	High Cal Point out of range	Reset High Cal Point (H<CR>)
cal3	High Point Measured Cal Value is Lower than Low Point Measured Value (or reverse)	Reset High(H<CR>), Low(h<CR>), or Both Cal Points
cal4	High Point Temperature Value is Lower than Low Point Temperature Value (or reverse)	

Table 3: Temperature Measurement Error Codes

Detailed Explanation of Temperature Measurement Error Codes

RTDo and RTDs: The first two error codes in Table 3 convey information on the state of the actual temperature sensor inside the RIC20. The sensor inside is a platinum RTD (Resistance Temperature Detector). “RTDo” is an abbreviation for “RTD is open” meaning that the temperature sensor appears to be electrically open and “RTDs” means that the sensor appears to be shorted. The sensor is not a user replaceable component so practically speaking, if this error is ever displayed then the unit must be returned to Torrey Pines Scientific for repairs. Visit https://www.torreypinesscientific.com/contact/.

cal1: An “open” circuit could be due to a disconnected sensor wire to the PCB, a broken sensor wire, a damaged or malfunctioning RTD sensor, or malfunctioning circuitry in the internal PCB. “RTDs” is an abbreviation for “RTD is shorted” meaning that the temperature sensor appears to be electrically closed. sencircuitry or with a calibration value, the unit will return the error code in place of the returned plate temperature value when the “p” command is sent. If an error is detected, the unit will go into Heater OFF mode (power to the heater plate will be turned off). Table 2 below lists the error codes, the likely cause, and recommended action to address the error. Calibration errors (cal0-cal4) are easily addressed by performing the action to reset the appropriate calibration points to the factory settings. The unit can then be re-calibrated if necessary.

XI. CLEANING, MAINTENANCE, AND CONSUMABLE PARTS

CLEANING

These units are subject to splashes and spills during normal use. Be sure to wipe up all spills with a soft cloth or paper towel as soon as they occur. If a cleaning solution is necessary, use a mild soap or detergent solution and a soft cloth. Do not use solvents.

Caution: Do not attempt to clean the plate surface when hot. Burns might occur.

MAINTENANCE

There is no ongoing maintenance program needed with these units other than the normal care and cleaning as instructed above, and a simple inspection done whenever the unit is to be used. This simple inspection should include:

1. Checking that the Power cord and the serial cable to and from the heater module are not frayed or burned.

2. Checking that the unit is not dirty to a point where proper performance is impaired. This is especially important relative to the membrane switch and LCD window.

3. Being certain to store the unit properly, when not in use, in an area that will not have items placed on top of the unit, and covering the unit in a way that will keep dirt and other foreign bodies out of the unit.

SPARE PARTS AND CONSUMABLES

There are very few spare or consumable parts. A simple list is below. For more information contact the factory.

Part Number	Description
730-0001	Power Cord, US
730-0006	Power Cord, German (European)
730-0008	Power Cord, UK
730-0004	Power Cord, Italian
730-0005	Power Cord, Australian

XII. ADDITIONAL SYMBOLS

The following are additional symbols found on labels on the instrument

Symbol	Description
V	Voltage
~	Alternating Current
A	Current
Hz	Frequency
W	Power

Contact Info:
For assistance, contact the factory at:
Torrey Pines Scientific, Inc.
Email: techsupport@torreypinesscientific.com
Phone: (760)930-9400

RIC20 Programming Manual

November 7, 2024

Applicable to RIC20. RIC20XR,and RIC20XT firmware v2.3
(Backward compatible with v1.0 and above)

TORREY PINES SCIENTIFIC, INC.
2713 Loker Ave. West
Carlsbad, CA 92010

TELEPHONE: (760)-930-9400
TOLL FREE: (866)-573-9104
FAX: (760)-930-9480
E-Mail: info@torreypinesscientific.com
Web site: www.torreypinesscientific.com

Features

Identification

Unit Model and version
Unit Serial Number
User Defined String (10 chars max)

Set Point (C)

Tenth Degree Precision
Settable Range: -10.0 to +100.0
“off” (Idle Mode—temp controller turned off)
Option to configure to always start in Idle Mode

Plate Temperature (C)

Tenth Degree Precision

Calibration

Calibrated at Factory
2 Point User Settable

Timer

User Settable Start Time (format -- hh:mm:ss)
Range: 00:00:00 to 24:59:59
Count Down (00:00:00 min)
Count Up (24:59:59 max)

LED Event Notification

LED Blinking → Plate Temp Changing
LED Steady On →Plate Temp Within 0.2C of Set Point for 60 seconds

Serial Event Notification

Plate Temp (User settable period 0-99:59)
TEMP_STEADY (User settable enable/disable)
TIMER=0 (User settable enable/disable)

Non-Volatile Memory

On Power Down, the following values are stored:

- set point
- high and low calibration values
- event broadcast settings

Manual Conventions

<CR>
The notation "<CR> " refers to the ASCII character for "carriage return" which is decimal 13 or hex D. Every valid RIC20 command must be terminated by this character. In HyperTerminal the character is sent when the "enter" key is pressed. Programs written in C for example, typically send this character when "\r" is appended to the transmitted command string.

<LF>
The notation "<LF> " refers to the ASCII character for "line feed" or "new line" which is decimal 10 or hex A. Every string that is returned from the RIC20 will be terminated with this character. Actually, every string that is returned from the RIC20 will be terminated with a <CR> then a <LF>.

(string)
Characters within parenthesis are strings consisting of 7-bit ASCII characters and the string itself is the argument for a command. The length of the string is clarified in the discussion of the command and the two parenthesis are not included in the transmission of the command string.

IMPORTANT POWER CONSIDERATIONS

The RIC20, RIC20XT & RIC20XR cooling/heating modules can be used as a standalone benchtop instrument but are designed primarily for integration into automated processing cells. In such systems, managing energy consumption during startup is often a concern, and the RIC20 units can contribute significantly to the initial load. For instance, when a RIC20 unit is powered on with a target temperature above or below ambient, it quickly reaches its peak current draw (over 4 amps for the RIC20 and over 8 amps for the RIC20XR or RIC20XT). However, as the unit approaches its target temperature, the current draw decreases significantly unless the target is near the maximum or minimum temperature limit.

When multiple RIC20 units are installed and powered on simultaneously, the initial combined current draw can place a substantial load on the cell's power supply. To address this, a staggered power-up sequence can help distribute the energy demand more evenly. Since the highest current draw occurs as each unit begins reaching its target temperature, staggering the activation allows each unit's current draw to drop before the next unit powers up. Alternatively, setting all units to Idle Mode or to a target temperature equal to ambient can significantly minimize initial current draw. For further load management, target temperatures can be set sequentially—adjusting the target for each unit only after the previous one nears or reaches its goal.

To ensure RIC20 units draw minimal current during startup, consider the following approaches:

1. **Idle Mode**: This mode effectively disables heating and chilling by removing the target temperature, keeping current draw minimal regardless of ambient conditions. Units can be set to start up in Idle Mode by sending a command to put the unit into Idle Mode prior to power down or by sending a command to configure the unit to always start up in Idle Mode regardless of the power down target temperature or mode. For more details see command i/I and x/X in the RIC20 Program Guide.

2. **Ambient Target**: Theoretically, setting target temperatures to match ambient can also minimize current draw if the plate and ambient temperatures are close, which is often the case in cold startups. However, in scenarios where the unit is restarted after a cell shutdown the plates will likely not be at ambient so start up in Idle Mode will be much more reliable and is highly recommended as the initial condition for sequential startups of multiple units.

Serial Interface

The RIC20 line of products are controlled using a simple serial RS232 interface. Each command sent to the RIC20 must be terminated with an ASCII carriage return character and every response string from the RIC20 will be terminated with a character pair consisting of an ASCII carriage return and ASCII linefeed character. All commands described within this manual are case sensitive. When a command is successfully received, an appropriate string will be returned. If a command is received with a syntax error, the character “e” will be returned to indicate the error.

The command set was developed to provide functionality that will enable multiple interface and programming scenarios. For example, if a user wishes to use a simple application such as HyperTerminal to control a single unit, linefeed and carriage control characters included in the command and return strings enable clean formatting within the HyperTerminal User Interface. Additionally, commands are included to enable different programming methodologies such as polled response or event driven responses. For example, a program could be written to set a new set point then read the plate temperature at some time interval (polling) and take specific actions when the plate reaches various values along the way. Or a program could be written to set a new set point then the program could do something else until the set point is reached and the “TEMP_STEADY” message is received from the unit (event driven). **NOTE: If the user written controller is event driven and it “misses” the TEMP_STEADY event message, it may hang waiting for the event that it missed. Asynchronous Serial communication “misses” are not that uncommon so it is recommended that the controller algorithm have provisions like a wait-for-event timeout and if a timeout occurs, code to check the plate temperature and compare that to the set point temperature then take the appropriate action. As a rule of thumb, the most robust coding approach is to set the set point, read and verify the new set point, then continuously poll the plate temperature for routines that do things like determine the user established steady condition and for routines that continuously monitor the temperature. Code written using set and verify with polling will recover easily if an occasional corruption of the serial communication occurs.**

When the Serial COM settings are configured correctly on the external control computer, every properly formatted command sent to the RIC20 will return either the requested data string or “ok” to acknowledge the successful reception of the command. The character “e” is returned when the RIC20 does not recognize the received command string. The code running in the external computer can use the returned strings for handshaking and/or to verify that the command was properly executed.

COM Settings

9600 baud
1 stop bit
no parity
no hardware handshake
50ms delay after each line sent (after each “;”)

Unit Identification Commands

Command: **v**
Function: **Return RIC20 Model and Version**
Description: When the command v<CR> is received by the RIC20 unit, the model number and the firmware version will be returned in a text string terminated by <CR><LF>. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example:
send: v<CR>
returned: RIC20 v2.30 <CR><LF>

Command: **V**
Function: **Return Serial Number**
Description: When the command V<CR> is received by the RIC20 unit, the 8 character serial number will be returned in a text string terminated by <CR><LF>. The serial number for every RIC20 unit is unique. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Return the 8 char serial number:
send: V<CR>
returned:12345678 <CR><LF>

Command: **>(user defined string)**
Function: **Store User ID String**
Description: Each unit has a unique serial number (ref cmd V) but it may be helpful to assign a custom "name" to identify a unit. This command enables a string of up to 10 characters to be stored with the unit. When the command >(user defined string)<CR> is received by the RIC20 unit, the string is stored and the unit returns ok<CR><LF>. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example—assign RIC20 the name "Unit 1"
send: Unit 1<CR>
returned: ok <CR><LF>

Example—verify that the name is "Unit 1"
send: ><CR>
returned: Unit 1<CR><LF>

Command: **>**
Function: **Return User ID String**
Description: When the command **>**<CR> is received by the RIC20 unit, the User ID String will be returned in a text string terminated by <CR><LF>. If no string has been stored, 10 space characters will be returned followed by <CR><LF>. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example—Return the string that was stored using the >(user defined string) command:
send: ><CR>
returned: Unit 1<CR><LF> if "Unit 1" was previously stored
returned: <CR><LF> if no user string has been stored

Start Up Commands

Command: **x**
Function: **Configure Unit to Always Power Up in Idle Mode**
Description: When the command x<CR> is received by the RIC20 unit, the unit will always power up in Idle Mode. See "Command: i" for further description of Idle Mode. When this command is sent, the setting is stored in memory of the unit and does not need to be sent again. See "Command: X" to return the unit to the default power up behavior.

See **"Important Power Considerations"** section of this manual for info related to Idle Mode

Example—Configure the unit to always power up in Idle Mode :
send: x<CR>
returned: ok<CR><LF>

Command: **X**
Function: **Configure Unit to Up in Default Mode**
Description: When the command X<CR> is received by the RIC20 unit, the unit will return to the default power up behavior. The default behavior is at power up, the unit will be set to the target temperature that was set when the unit powered down. If the unit powered down in Idle Mode, it will power up in Idle mode. When this command is sent, the setting is stored in memory of the unit and does not need to be sent again.

See **"Important Power Considerations"** section of this manual for info related to Idle Mode

Example—Configure the unit to always power up in Default Mode :
send: X<CR>
returned: ok<CR><LF>

Temperature Commands

Command: **s**
Function: **Return Set Point Temperature**
Description: When the command s<CR> is received by the RIC20 unit, the current set point temperature will be returned in a text string terminated by <CR><LF>. The setpoint format length is variable depending on the number of digits in the setpoint temperature and whether the value is negative. The value will always include a decimal point and one character representing the fractional tenth value. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Note: Putting the RIC20 into Idle Mode (see "Command: i") will turn off the temperature controller and the set point will be read as "off".

Example—Return the current set point:
send: s<CR>
returned example 1: -10.0 <CR><LF>
returned example 2: 9.3 <CR><LF>
returned example 3: 100.0 <CR><LF>
returned example 4: off<CR><LF> if unit is in Idle Mode (see "Command: i")

Command: **n**
Function: **Set and Store New Set Point Temperature**
Description: When the command n(new_temperature)<CR> is received by the RIC20 unit, the set point will be changed to new_temperature and the text string "ok" will be returned terminated by <CR><LF>. The format for new_temperature is variable depending on the number of digits in the desired new set point temperature and whether the value is negative. The value must always include a decimal point and one character representing the fractional tenth value. The settable range is -10.0C to 100.0C.

If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example1— Change set point to -10.0C:
send: n-10.0<CR>
returned: ok <CR><LF>

Example2 – Change set point to 9.3C :
send: n9.3<CR>
returned: ok <CR><LF>

Example3 – Change set point to 100.0C :
send: n100.0<CR>
returned: ok <CR><LF>

Command: **i**
Function: **Set RIC20 Unit to Idle Mode**
Description: When the command i<CR> is received by the RIC20 unit, the temperature controller will be switched "off", meaning that the plate will no longer heat or cool. In Idle Mode, the unit will report a set point of "off". To exit Idle Mode, simply set the set point to a new value using cmd n. If the command is not received in the proper syntax, e<CR><LF> will be returned.

See "Start Up Commands" to configure unit to always power up in Idle Mode and "**Important Power Considerations**" section of this manual for more info related to Idle Mode

Example – Set Idle Mode:
send: i<CR>
returned: ok <CR><LF>

Example – verify Idle Mode:
send: s<CR>
returned: off<CR><LF>

Example – exit Idle Mode by setting new set point to 25.0C
send: n25.0<CR>
returned: ok <CR><LF>

Example – verify no longer in Idle Mode and set point is 25.0C:
send: s<CR>
returned: 25.0<CR><LF>

Timer Commands

Command: **a**
Function: **Return Current Timer Value**
Description: When the command a<CR> is received by the RIC20 unit, the current timer value will be returned in a text string terminated by <CR><LF>. The timer string is a fixed width of 8 characters in the format hh:mm:ss. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – the current timer value is 1 hour, 32 minutes, 15 seconds:
send: a<CR>
returned: 01:32:15 <CR><LF>

Example – verify that the current timer has been cleared (see Command: ac):
send: a<CR>
returned: 00:00:00 <CR><LF>

Command: **a(hh:mm:ss)**
Function: **Set Current Timer Value**
Description: When the command a(hh:mm:ss)<CR> is received by the RIC20 unit, the current timer value will be set to the new value in the 8 character format hh:mm:ss. The settable timer range is 00:00:00 to 24:59:59. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Set the current timer value to 1 hour, 32 minutes, 15 seconds:
send: a01:32:15<CR>
returned: ok <CR><LF>

Command: **au**
Function: **Start Timer—Count Up**
Description: When the command au<CR> is received by the RIC20 unit, the current timer value will increment every second. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Verify current timer value is 00:00:00
send: a<CR>
returned: 00:00:00 <CR><LF>

Example – Start incrementing timer
send: au<CR>
returned: ok<CR><LF>

Example – Check timer after 5 seconds
send: a<CR>
returned: 00:00:05 <CR><LF>

Command: **ad**
Function: **Start Timer—Count Down**
Description: When the command au<CR> is received by the RIC20 unit, the current timer value will decrement every second. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Verify current timer value is 00:30:00

send: a<CR>
returned: 00:30:00 <CR><LF>

Example – Start decrementing timer
send: ad<CR>
returned: ok<CR><LF>

Example – Check timer after 5 seconds
send: a<CR>
returned: 00:29:55 <CR><LF>

Command: **ap**
Function: **Pause or Stop Timer**
Description: When the command ap<CR> is received by the RIC20 unit, the current timer value will stop decrementing or incrementing. To restart the timer from the current timer value, send either an au or ad command.   If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Stop Timer
send: ad<CR>
returned: ok<CR><LF>

Example – Read Current Timer Value
send: a<CR>
returned: 00:29:55 <CR><LF>

Command: **ac**
Function: **Clear the Current Timer Value**
Description: When the command ac<CR> is received by the RIC20 unit, the current timer value will be set to 00:00:00. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Clear the Current Timer Value
send: ac<CR>
returned: ok<CR><LF>

Example – Read Current Timer Value
send: a<CR>
returned: 00:00:00 <CR><LF>

Calibration Commands

Command: **R**
Function: **Return the High Calibration Point Temperature**
Description: When the command R<CR> is received by the RIC20 unit, the temperature at which the high calibration point was established will be returned in a text string terminated by <CR><LF>. The high calibration point temperature format length is variable depending on the number of digits in the calibration temperature and whether the value is negative. The value will always include a decimal point and one character representing the fractional tenth value. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Determine the high temperature calibration point:
send: R<CR>
returned: 75.0 <CR><LF> (the RIC20 was calibrated at 75.0C)
returned: 100.0 <CR><LF> (the RIC20 has the default High Calibration Temp)

Command: **r**
Function: **Return the Low Calibration Point Temperature**
Description: When the command r<CR> is received by the RIC20 unit, the temperature at which the low calibration point was established will be returned in a text string terminated by <CR><LF>. The low calibration point temperature format length is variable depending on the number of digits in the calibration temperature and whether the value is negative. The value will always include a decimal point and one character representing the fractional tenth value. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Determine the low temperature calibration point:
send: r<CR>
returned: 10.0 <CR><LF> (the RIC20 was calibrated at 10.0C)
returned: -10.0 <CR><LF> (the RIC20 has the default Low Calibration Temp)

Command: **T**
Function: **Return the Measured Temperature at the High Calibration Point**
Description: When the command T<CR> is received by the RIC20 unit, the measured temperature when the RIC20 was at the high calibration point temperature will be returned in a text string terminated by <CR><LF>. The measured temperature format length is variable depending on the number of digits in the calibration temperature and whether the value is negative. The value will always include a decimal point and one character representing the fractional tenth value. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Determine the measured temperature at the high calibration point of 75.0C:
send: T<CR>
returned: 73.2 <CR><LF> (the plate was measured to be 73.2C when the RIC20 was set to 75.0C)

Command: **T(measured_temp)**
Function: **Set the Measured Temperature at the High Calibration Point**
Description: When the command T(measured_temp)<CR> is received by the RIC20 unit, measured_temp will be stored and used with the High Calibration Point temperature to calculate calibration offsets. The measured_temp format length is variable depending on the number of digits in the calibration temperature and whether the value is negative. The value will always include a decimal point and one character representing the fractional tenth value. If the command is not received in the proper syntax, e<CR><LF> will be returned.

**IMPORTANT NOTE: Both the RIC20 temperature and the calibration measurement temperature must be steady at the High Calibration Point for at least 10 minutes before the measured temperature is entered. Additional settling time may be required depending on the thermal conduction of the material at the calibration measurement point. For example, if a liquid sample is the measurement point for the calibration, more time may be required to reach a stable measurement temperature. If the unit or the measured value are not steady at the High Calibration Point temperature when the measured value is entered, significant calibration error may result.

Example – The RIC20 High Calibration Point is 75.0C, the measured temp when the unit is steady at 75.0C is 73.2C. Set the measured High Calibration Point temperature:

send: T73.2<CR>
returned: ok <CR><LF>

Example – Verify that the Measured Temperature at the High Cal Point is 73.2
send: T<CR>
returned: 73.2<CR><LF>

Command: **t**
Function: **Return the Measured Temperature at the Low Calibration Point**
Description: When the command t<CR> is received by the RIC20 unit, the measured temperature when the RIC20 was at the low calibration point temperature will be returned in a text string terminated by <CR><LF>. The measured temperature format length is variable depending on the number of digits in the calibration temperature and whether the value is negative. The value will always include a decimal point and one character representing the fractional tenth value. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – Determine the measured temperature at the low calibration point of 10.0C:
send: t<CR>
returned: 11.3 <CR><LF> (the plate was measured to be 11.3C when the RIC20 was set to 10.0C)

Command: **t(measured_temp)**
Function: **Set the Measured Temperature at the Low Calibration Point**
Description: When the command t(measured_temp)<CR> is received by the RIC20 unit, measured_temp will be stored and used with the Low Calibration Point temperature to calculate calibration offsets. The measured_temp format length is variable depending on the number of digits in the calibration temperature and whether the value is negative. The value will always include a decimal point and one character representing the fractional tenth value. If the command is not received in the proper syntax, e<CR><LF> will be returned.

**IMPORTANT NOTE: Both the RIC20 temperature and the calibration measurement temperature must be steady at the Low Calibration Point for at least 10 minutes before the measured temperature is entered. Additional settling time may be required depending on the thermal conduction of the material at the calibration measurement point. For example, if a liquid sample is the measurement point for the calibration, more time may be required to reach a stable measurement temperature. If the unit or the measured value are not steady at the Low Calibration Point temperature when the measured value is entered, significant calibration error may result.

Example – The RIC20 Low Calibration Point is 10.0C, the measured temp when the unit is steady at 10.0C is 11.3C. Set the measured Low Calibration Point temperature:

send: t11.3<CR>
returned: ok <CR><LF>

Example – Verify that the Measured Temperature at the Low Cal Point is 11.1
send: t<CR>
returned: 11.3<CR><LF>

Command: **H**
Function: **Reset the High Temperature Calibration Points to Default**
Description: When the command H<CR> is received by the RIC20 unit, the High Temperature Calibration Point and the Measured Temperature at the High Calibration Point will be set to the default value of 100.0C. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – set the High Calibration Point to Default (100.0C)
send: H<CR>
returned: ok <CR><LF>

Command: **h**
Function: **Reset the Low Temperature Calibration Points to Default**
Description: When the command h<CR> is received by the RIC20 unit, the Low Temperature Calibration Point and the Measured Temperature at the Low Calibration Point will be set to the default value of -10.0C. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – set the Low Calibration Point to Default (-10.0C)
send: h<CR>
returned: ok <CR><LF>

Macro Commands

Command: **m**
Function: **Macro to return all 4 calibration values**
Description: When the command m<CR> is received by the RIC20 unit, the Low Temperature Calibration Point, the Measured Temperature at the Low Calibration Point, the High Temperature Calibration Point, and the Measured Temperature at the High Calibration Point will be returned in that order in a string delimited by commas and terminated by <CR><LF>. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – return all cal points from the previous examples
send: m<CR>
returned: 10.0,11.3,75.0,73.2 <CR><LF>

Command: **M**
Function: **Macro to return all Status, SP, PT, Timer**
Description: When the command M<CR> is received by the RIC20 unit, a string containing the Status String, Set Point Temperature, Plate Temperature, and Current Timer Value will be returned delimited by commas and terminated by <CR><LF>. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – return Status, SP, PT, Timer
send: M<CR>
returned: StbLH,-10.0,-10.0,00:04:13<CR><LF> (temp is steady, timer not running, unit not broadcasting, low temp cal done, high temp cal done, set point is -10.0, plate temp is -10.0, the timer value is 4 mins 13 sec)

Event Notification Commands

Command: **b(mm:ss)**
Function: **Broadcast the Plate Temperature**
Description: When the command b(mm:ss)<CR> is received by the RIC20 unit, the unit will return the plate temperature every time interval defined by mm:ss. To disable the Plate Temperature, set the time interval to 00:00. Sending b<CR> will return the current settings. If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – return the plate temperature every 5 seconds
send: b00:05<CR>
returned: ok <CR><LF>

Example – stop broadcasting the plate temperature
send: b00:00<CR>
returned: ok <CR><LF>

Example – return the current Broadcast Plate Temperature settings
send: b<CR>
returned: 00:00 <CR><LF>

Command: **B(sz)**
Function: **Broadcast Temperature or Timer Events**
Description: When the command B(ab)<CR> is received by the RIC20 unit, the unit will enable or disable Temperature or Timer Events depending on the case of the variables s and z as shown in the table below:

Return string “TEMP_STEADY<CR><LF>” when plate temperature is steady
S = enable
s = disable

Return string “TIMER=0<CR><LF>” when timer reaches zero
Z = enable
z = disable

Sending B<CR> will return the current settings.
If the command is not received in the proper syntax, e<CR><LF> will be returned.

Example – set to broadcast when the temperature is steady at the set point but do not broadcast when the timer has reached zero.
send: BSz<CR>
returned: ok <CR><LF>

Example – return the current Broadcast Event settings
send: B<CR>
returned: Sz <CR><LF>

Utility Commands

Command: **S**
Function: **Return Status String**
Description: When the command S<CR> is received by the RIC20 unit, the unit will return a string of 5 characters in the order stblh<CR><LF> that indicate the current status of the unit as described in the table below:

s = temperature is not steady
S = temperature is steady
t = timer is not running
T = timer is running
b = unit is not broadcasting
B = unit is broadcasting
l = low temp cal has not been done (default values stored)
L = low temp cal has been done
h = high temp cal has not been done (default values stored)
H = high temp cal has been done

Example – return the Status String
send: S<CR>
returned: StbLH <CR><LF> (temp is steady, timer not running, unit not broadcasting, low temp cal done, high temp cal done)

Serial Command Quick Reference Table

			Example	
	Command	**Function**	**sent**	**returned**
ID	**v**	return model and version	v <CR>	RIC20 v1.0 <CR LF>
	V	return serial number	V <CR>	12345678 <CR LF>
	>(abc123...)	set user string (10 chars max)	>UNIT 10 <CR>	ok <CR LF>
	>	return user string	> <CR>	UNIT 10 <CR LF>
Start Up	**x**	Configure unit to always power up in Idle Mode	x<CR>	ok <CR LF>
		Configure unit to power up in Defaut Mode (Target or Idle Mode same as power down value)	X<CR.	ok <CR LF>
Temperature and Timer	**s**	return set point temperature	s <CR>	-10.0 <CR LF> off <CR LF> if idle mode
	n(xxx.x)	set new set point temperature (x.x min)	n25.0 <CR>	ok <CR LF>
	p	return plate temperature	p <CR>	-10.0 <CR LF>
	i	set idle mode (plate is off)	i <CR>	ok <CR LF>
	a	return current timer value	a <CR>	00:04:13 <CR LF>
	a(hh:mm:ss)	set timer value (24:59:59 max)	a00:05:00 <CR>	ok <CR LF>
	au	start timer--count up	au <CR>	ok <CR LF>
	ad	start timer--count down	ad <CR>	ok <CR LF>
	ap	pause/stop timer	ap <CR>	ok <CR LF>
	ac	clear timer (00:00:00)	ac <CR>	ok <CR LF>
	M	macro to return status,sp, pt, timer	M <CR>	StbLH,-10.0,-10.0,00:04:13 <CR LF>
Calibration	**R**	return HIGH cal point temperature	R <CR>	75.0 <CR LF>
	r	return LOW cal point temperature	r <CR>	10.0 <CR LF>
	T(xxx.x)	set measured temperature at HIGH cal point (x.x min)	T73.2 <CR>	ok <CR LF>
	T	return measured RTD temp at HIGH cal point	T <CR>	73.2 <CR LF>
	t(xxx.x)	set measured temperature at LOW cal point (x.x min)	t11.3 <CR>	ok <CR LF>
	t	return measured RTD temp at LOW cal point	t <CR>	11.3 <CR LF>
	H	reset HIGH temp cal points to default (100.0C)	H <CR>	ok <CR LF> (R and T now 100.0)
	h	reset LOW temp cal points to default (-10.0C)	h <CR>	ok <CR LF> (r and t now -10.0)
	m	macro to return all 4 cal values (r,t,R,T<CR>)	m <CR>	10.0,11.3,75.0,73.2 <CR LF>
Event Notification	**b**(mm:ss)	Broadcast Plate Temperature returns Plate Temp every mm:ss, 99:59 max	b00:10 <CR>	plate temp <CR LF> returned every 10 seconds
	B(sz)	Broadcast Temperature or Timer Events Return "TEMP_STEADY" when temp is steady S = enable s = disable Return "TIMER=0" when timer reaches zero Z = enable z = disable	BSz <CR>	TEMP_STEADY <CR LF> returned at event TIMER=0 <CR LF> not returned at event
Utility	**S**	return status (stblh<CR>) s = temperature is not steady S = temperature is steady t = timer is not running T = timer is running b = unit is not broadcasting B = unit is broadcasting l = low temp cal not done (default values stored) L = low temp cal has been done h = high temp cal not done (default values stored) H = high temp cal has been done	S <CR>	StbLH <CR LF> --temp is steady --timer not running --unit not broadcasting --unit has been calibrated at low temp --unit has been calibrated at high temp

<CR> is return char (for example: “enter” keyboard press for HyperTerminal, “\r” for C pgms, ASCII hex char “0D”)
<LF> is new line ASCII char (hex ‘0A’)