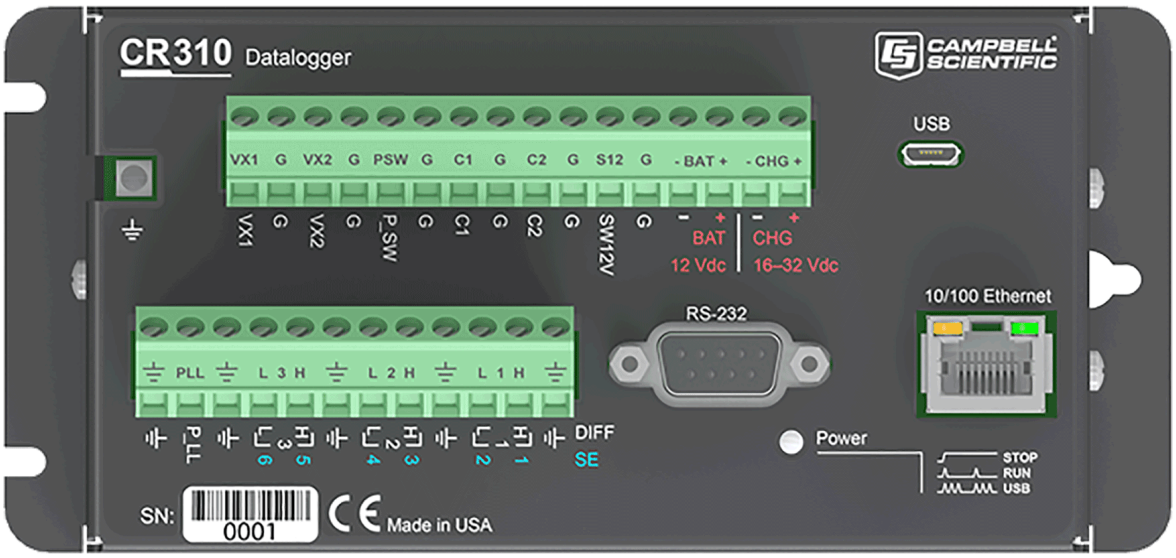
Product Manual

Datalogger

CR300 Series

Compact Datalogger



Revision: 04/30/18

RELIABLE SINCE 1974 MONITORING

CAMPBELL SCIENTIFIC

Warranty

The datalogger is warranted for three (3) years subject to this limited warranty: https://www.campbellsci.com/terms#warranty.

Acknowledgements

lwIP

Precautions

DANGER – MANY HAZARDS ARE ASSOCIATED WITH INSTALLING, USING, MAINTAINING, AND WORKING ON OR AROUND TRIPODS, TOWERS, AND ANY ATTACHMENTS TO TRIPODS AND TOWERS SUCH AS SENSORS, CROSSARMS, ENCLOSURES, ANTENNAS, ETC. FAILURE TO PROPERLY AND COMPLETELY ASSEMBLE, INSTALL, OPERATE, USE, AND MAINTAIN TRIPODS, TOWERS, AND ATTACHMENTS, AND FAILURE TO HEED WARNINGS, INCREASES THE RISK OF DEATH, ACCIDENT, SERIOUS INJURY, PROPERTY DAMAGE, AND PRODUCT FAILURE. TAKE ALL REASONABLE PRECAUTIONS TO AVOID THESE HAZARDS. CHECK WITH YOUR ORGANIZATION'S SAFETY COORDINATOR (OR POLICY) FOR PROCEDURES AND REQUIRED PROTECTIVE EQUIPMENT PRIOR TO PERFORMING ANY WORK.

Use tripods, towers, and attachments to tripods and towers only for purposes for which they are designed. Do not exceed design limits. Be familiar and comply with all instructions provided in product manuals. Manuals are available at www.campbellsci.ca or by telephoning 780.454.2505 (Canada). You are responsible for conformance with governing codes and regulations, including safety regulations, and the integrity and location of structures or land to which towers, tripods, and any attachments are attached. Installation sites should be evaluated and approved by a qualified engineer. If questions or concerns arise regarding installation, use, or maintenance of tripods, towers, attachments, or electrical connections, consult with a licensed and qualified engineer or electrician.

General

- Prior to performing site or installation work, obtain required approvals and permits.
- Use only qualified personnel for installation, use, and maintenance of tripods and towers, and any attachments to tripods and towers. The use of licensed and qualified contractors is highly recommended.
- Read all applicable instructions carefully and understand procedures thoroughly before beginning work.
- Wear a hardhat and eye protection, and take other appropriate safety precautions while working on or around tripods and towers.
- Do not climb tripods or towers at any time, and prohibit climbing by other persons. Take reasonable precautions to secure tripod and tower sites from trespassers.
- Use only manufacturer recommended parts, materials, and tools.

Utility and Electrical

- You can be killed or sustain serious bodily injury if the tripod, tower, or attachments you are installing, constructing, using, or maintaining, or a tool, stake, or anchor, come in contact with overhead or underground utility lines.
- Maintain a distance of at least one-and-one-half times structure height, or 20 feet, or the distance required by applicable law, whichever is greater, between overhead utility lines and the structure (tripod, tower, attachments, or tools).
- Prior to performing site or installation work, inform all utility companies and have all underground utilities marked.
- Comply with all electrical codes. Electrical equipment and related grounding devices should be installed by a licensed and qualified electrician.

Elevated Work and Weather

- Exercise extreme caution when performing elevated work.
- Use appropriate equipment and safety practices.
- During installation and maintenance, keep tower and tripod sites clear of un-trained or non-essential personnel. Take precautions to prevent elevated tools and objects from dropping.
- Do not perform any work in inclement weather, including wind, rain, snow, lightning, etc.

Maintenance

- Periodically (at least yearly) check for wear and damage, including corrosion, stress cracks, frayed cables, loose cable clamps, cable tightness, etc. and take necessary corrective actions.
- Periodically (at least yearly) check electrical ground connections.

DANGER: Fire, explosion, and severe-burn hazard. Misuse or improper installation of the internal lithium battery can cause severe injury. Do not recharge, disassemble, heat above 100 °C (212 °F), solder directly to the cell, incinerate, or expose contents to water. Dispose of spent lithium batteries properly.

WARNING:

- Protect from over-voltage.
- Protect from water (see "Datalogger Enclosures" on page 60).
- Protect from ESD (see "Electrostatic Discharge and Lightning Protection" on page 62).

IMPORTANT: Note the following about the internal battery:

- When primary power is continuously connected to the datalogger, the battery will last up to 10 years or more.
- When primary power is NOT connected to the datalogger, the battery will last about three years.
- See "Internal Battery" on page 60 for more information.

IMPORTANT: Maintain a level of calibration appropriate to the application. Campbell Scientific recommends factory recalibration of the datalogger every three years.

WHILE EVERY ATTEMPT IS MADE TO EMBODY THE HIGHEST DEGREE OF SAFETY IN ALL CAMPBELL SCIENTIFIC PRODUCTS, THE CUSTOMER ASSUMES ALL RISK FROM ANY INJURY RESULTING FROM IMPROPER INSTALLATION, USE, OR MAINTENANCE OF TRIPODS, TOWERS, OR ATTACHMENTS TO TRIPODS AND TOWERS SUCH AS SENSORS, CROSSARMS, ENCLOSURES, ANTENNAS, ETC.

Please Read First

About this manual

Please note that this manual was originally produced by Campbell Scientific Inc. (CSI) primarily for the US market. Some spellings, weights and measures may reflect this origin.

Some useful conversion factors:

Area:	1 in^2 (square inch) = 645 mm^2
Length:	1 in. (inch) = 25.4 mm
	1 ft (foot) = 304.8 mm
	1 yard = 0.914 m
	1 mile = 1.609 km
Mass:	1 oz. (ounce) = 28.35 g
	1 lb (pound weight) = 0.454 kg
Pressure:	1 psi (lb/in2) = 68.95 mb
Volume:	1 US gallon = 3.785 litres

In addition, part ordering numbers may vary. For example, the CABLE5CBL is a CSI part number and known as a FIN5COND at Campbell Scientific Canada (CSC). CSC Technical Support will be please to assist with any questions.

About sensor wiring

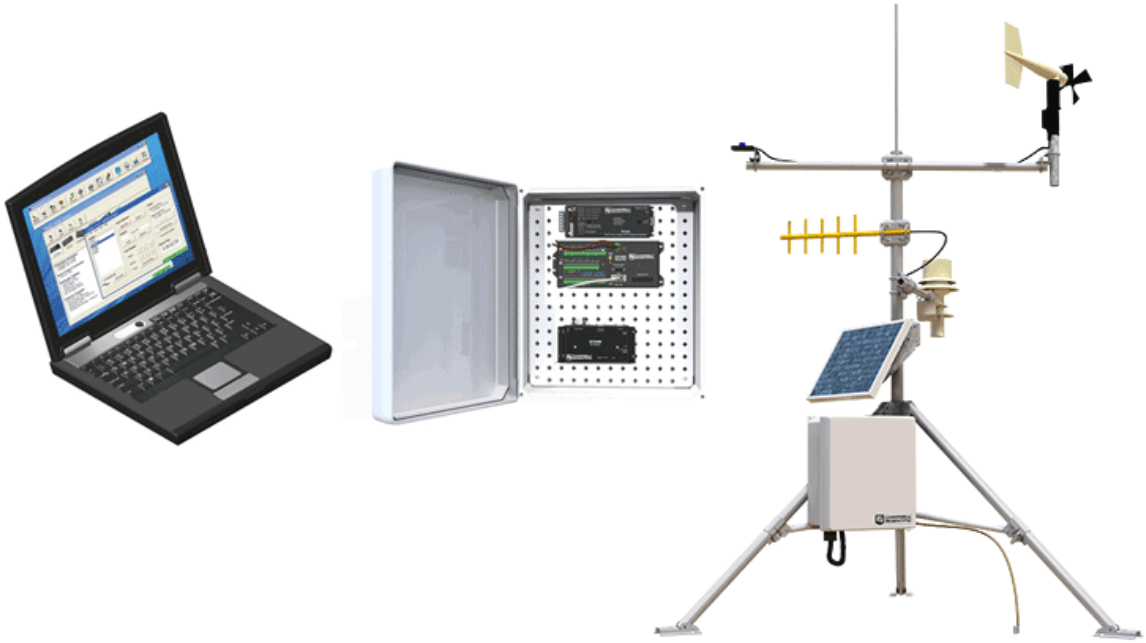
Please note that certain sensor configurations may require a user supplied jumper wire. It is recommended to review the sensor configuration requirements for your application and supply the jumper wire if necessary.

Table of Contents

Data Acquisition System Components

A basic data acquisition system consists of sensors, measurement hardware, and a computer with programmable software. The objective of a data acquisition system should be high accuracy, high precision, and resolution as high as appropriate for a given application.

The concept of a data acquisition system is illustrated in the following figure.



Following is a list of typical data acquisition system components:

- **Sensors** - Electronic sensors convert the state of a phenomenon to an electrical signal (see "Sensors" on page 2 for more information).
- **Datalogger** - The datalogger measures electrical signals or reads serial characters. It converts the measurement or reading to engineering units, performs calculations, and reduces data to statistical values. Data are stored in memory to await transfer to a computer by way of an external storage device or a communication link.
- **Data Retrieval and Communications** - Data are copied (not moved) from the datalogger, usually to a computer, by one or more methods using datalogger support software. Most communication options are bi-directional, which allows programs and settings to be sent to the datalogger. For more information, see "Sending a Program to the Datalogger" on page 29.
- **Datalogger Support Software** - Software retrieves data, sends programs, and sets settings. The software manages the communication link and has options for data display.
- **Programmable Logic Control** - Some data acquisition systems require the control of external devices to facilitate a measurement or to control a device based on measurements. This datalogger is adept at programmable logic control. See "Programmable Logic Control" on page 9 for more information.

Sensors

Sensors transduce phenomena into measurable electrical forms by modulating voltage, current, resistance, status, or pulse output signals. Suitable sensors do this with accuracy and precision. Smart sensors have internal measurement and processing components and simply output a digital value in binary, hexadecimal, or ASCII character form.

Most electronic sensors, regardless of manufacturer, will interface with the datalogger. Some sensors require external signal conditioning. The performance of some sensors is enhanced with specialized input modules. The datalogger, sometimes with the assistance of various peripheral devices, can measure or read nearly all electronic sensor output types.

The following list may not be comprehensive. A library of sensor manuals and application notes are available at www.campbellsci.com/support to assist in measuring many sensor types.

SUPPORTED SENSOR TYPES

- Analog
    - Voltage
    - Current
    - Strain
    - Thermocouples
    - Resistive bridges
- Pulse
    - High frequency
    - Switch-closure
    - Low-level ac
- Period average
- Vibrating wire (through interface modules)
- Smart sensors
    - SDI-12
    - RS-232
    - Modbus
    - DNP3
    - TCP/IP (CR310 only)

The CR300 Series Datalogger

CR300 series dataloggers are multi-purpose, compact, measurement and control dataloggers. These small, low-cost, high-value dataloggers offer fast communications, low power requirements, built-in USB, and excellent analog input accuracy and resolution. They can measure most hydrological, meteorological, environmental, and industrial sensors. They concentrate data, make it available over varied networks, and deliver it using your preferred protocol. They also perform automated on-site or

remote decision making for control and M2M communications. CR300 series dataloggers are ideal for small applications requiring long-term remote monitoring and control.

CR300 SERIES PRODUCT LINE

The CR300 series product line consists of the CR300 and the CR310. The primary differences between the CR300 and CR310 are that the CR310 offers removable terminals and a 10/100 Ethernet connection.

The CR300 series can include Wi-Fi, cellular, or the following radio options for different regions:

- RF407: US and Canada
- RF412: Australia and New Zealand
- RF422: Europe

COMPONENTS

CR300 series dataloggers are the main part of a data acquisition system (see "Data Acquisition System Components" on page 1 for more information). Each has a central-processing unit (CPU), analog and digital measurement inputs, digital outputs, and memory. An operating system (firmware) coordinates the functions of these parts in conjunction with the onboard clock and the CRBasic application program.

The CR300 series can simultaneously provide measurement and communication functions. Low power consumption allows the datalogger to operate for extended time on a battery recharged with a solar panel, eliminating the need for ac power.

OPERATIONS

CR300 series dataloggers measure almost any sensor with an electrical response, drive direct communications and telecommunications, reduce data to statistical values, perform calculations, and control external devices. After measurements are made, data are stored in onboard, nonvolatile memory awaiting transfer to the computer. Because most applications do not require that every measurement be recorded, the program usually combines several measurements into computational or statistical summaries, such as averages and standard deviations.

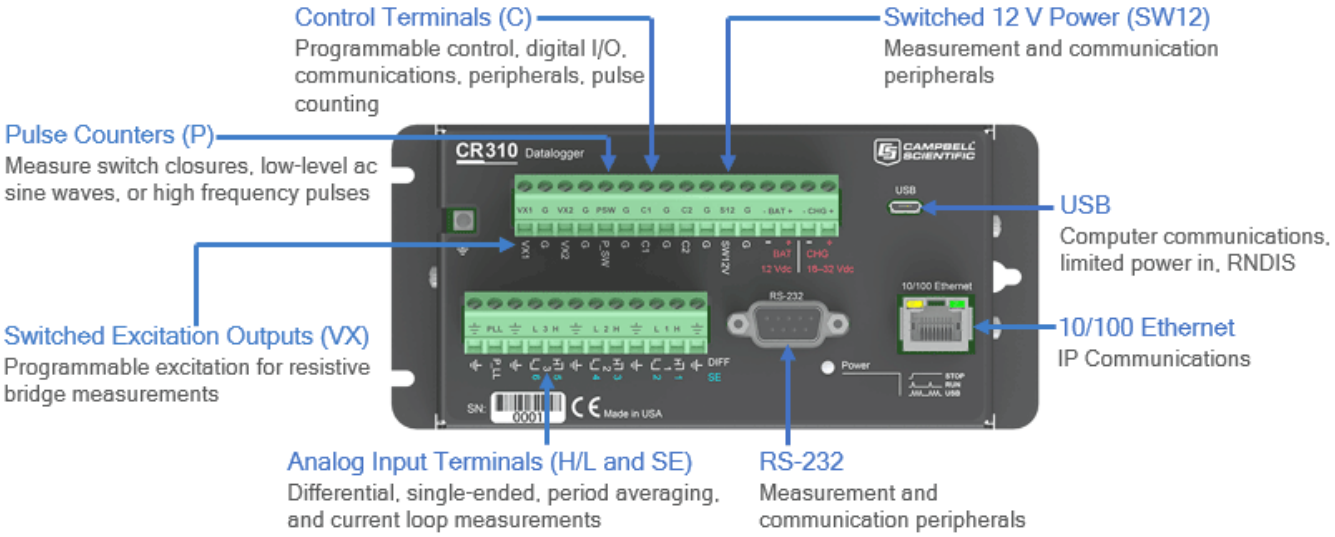
PROGRAMS

A program directs the datalogger on how and when sensors are to be measured, calculations made, and data stored. The application program for the CR300 series is written in CRBasic, which is a programming language that includes measurement, data processing, and analysis routines, as well as the standard BASIC instruction set. For simpler applications, Short Cut, a user-friendly program generator, can be used to write the program. For more demanding programs, use CRBasic Editor. If you are programming with CRBasic, you can utilize the extensive help available with the CRBasic Editor software (see https://help.campbellsci.com/CRBasic/CR300/ for searchable, CRBasic online help). Formal training is also available from Campbell Scientific.

Wiring Panel and Terminal Functions

The CR300 series wiring panel provides terminals and connectors for connecting sensors, power, and communication devices. Surge protection is incorporated internally in all terminals. The wiring panel is the interface to most CR300 series functions so studying it is a good way to get acquainted with the CR300 series. Functions of the terminals are broken down into the following categories.

- Analog input
- Pulse counting
- Analog output
- Communications
- Digital I/O
- Power input
- Power output
- Ground terminals



Analog Input						
SE DIFF	1 2 ⌐1¬ H	L	3 4 ⌐2¬ H	L	5 6 ⌐3¬ H	L
Single-Ended Voltage	✓	✓	✓	✓	✓	✓
Differential Voltage	H	L	H	L	H	L
Ratiometric/Bridge	✓	✓	✓	✓	✓	✓
Thermocouple	✓	✓	✓	✓	✓	✓
Current Loop	✓	✓				
Period Average	✓	✓	✓	✓		

Pulse Counting	C1	C2	P_SW	P_LL	SE1	SE2	SE3	SE4	SE5	SE6
Switch-Closure	✓	✓	✓							
High Frequency	✓	✓	✓	✓	✓	✓	✓	✓		
Low-level Ac				✓						
Quadrature	✓	✓			✓	✓				

Analog Output	VX1	VX2
Switched Voltage Excitation	✓	✓

Voltage Output[1]	C1	C2	SE1-4	VX1	VX2	P_SW	SW12V
3.3 Vdc			✓	✓	✓	✓	
5 Vdc	✓	✓		✓	✓		
12 Vdc							✓

[1]SE 1-4, P_SW, and C1-C2 have limited drive capacity

Communications	C1	C2	SE1-3	RS-232
SDI-12	✓	✓		
RS-232				✓
RS-232 0-5V	✓	✓		
GPS Time Sync	✓	✓	✓	
GPS NMEA Sentences	Rx	Rx		Rx

Communication Functions also include Ethernet (CR310 only) and USB

Digital I/O	C1	C2	P_SW	SE1	SE2	SE3	SE4	SE5	SE6
General I/O	✓	✓	✓	✓	✓	✓	✓		
Pulse-Width Modulation Output				✓	✓	✓	✓		
Interrupt	✓	✓		✓	✓	✓	✓		

Power Input

The datalogger requires a power supply. It can receive power from a variety of sources, operate for several months on non-rechargeable batteries, and supply voltage and control for many sensors and devices. The datalogger operates with external power connected to the green **BAT** and/or **CHG** terminals on the face of the wiring panel. The positive power lead connects to +. The negative lead connects to **-**. The power terminals are internally protected against polarity reversal and high voltage transients.

In the field, the datalogger can be powered in any of the following ways:

- 10 to 18 Vdc applied to the **BAT** + and – terminals
- 16 to 32 Vdc applied to the **CHG** + and – terminals

To establish an uninterruptible power supply (UPS), connect the primary power source (often a transformer, power converter, or solar panel) to the **CHG** terminals and connect a nominal 12 Vdc sealed rechargeable battery to the **BAT** terminals. See "Power Budgeting" on page 64 for more information.

Warning: Sustained input voltages in excess of 32 Vdc on **CHG** or **BAT** terminals can damage the transient voltage suppression.

Be sure that power supply components match the specifications of the device to which they are connected. When connecting power, first switch off the power supply, insert the positive lead, then insert the negative lead. Once you make the connection, turn the power supply on. See "Troubleshooting Power Supplies" on page 74 for more information.

Following is a list of CR300 series power input terminals and the respective power types supported.

- **BAT terminals**: Voltage input is 10 to 18 Vdc. This connection uses the least current since the internal datalogger charging circuit is bypassed. If the voltage on the **BAT** terminals exceeds 19 V, power is shut off to certain parts of the datalogger to prevent damaging connected sensors or peripherals.
- **CHG terminals**: Voltage input range is 16 to 32 Vdc. Connect a primary power source, such as a solar panel or Vac-to-Vdc transformer, to **CHG**. The voltage applied to **CHG** terminals must be at least 0.3 V higher than that needed to charge a connected battery. When within the 16 to 32 Vdc range, it will be regulated to the optimal charge voltage for a lead acid battery at the current datalogger temperature, with a maximum voltage of approximately 15 Vdc. A battery need not be connected to the **BAT** terminals to supply power to the datalogger through the **CHG** terminals. The onboard charging regulator is designed for efficiently charging lead-acid batteries. It will not charge lithium or alkaline batteries.
- **USB port**: 5 Vdc via USB connection. If power is also provided with **BAT** or **CHG**, power will be supplied by whichever has the highest voltage. If USB is the only power source, then the **SW12** terminal will not be operational. When powered by USB (no other power supplies connected) **Status** field **Battery** = **0**. Functions that will be active with a 5 Vdc source include sending programs, adjusting datalogger settings, and making some measurements. The excitation range of **VX1** and **VX2** is reduced to 150 to 2500 mV.

Note: The **Status** field **Battery** value and the destination variable from the **`Battery()`** instruction (often called `batt_volt` or `BattV`) in the **Public** table reference the external battery voltage. For detailed information about working with the internal battery, see "Internal Battery" on page 62.

POWER LED INDICATOR

When the datalogger is powered, the Power LED will turn on according to power and program states:

- **Off**: No power, no program running.
- **1 flash every 10 seconds**: Powered from **BAT**, program running.
- **2 flashes every 10 seconds**: Powered from **CHG**, program running.
- **3 flashes every 10 seconds**: Powered via USB, program running.
- **Always on**: Powered, no program running.

Power Output

The datalogger can be used as a power source for sensors and peripherals. Take precautions to prevent damage to sensors or peripherals from over- or under-voltage conditions, and to minimize errors. Exceeding current limits causes voltage output to become unstable. Voltage should stabilize once current is again reduced to within stated limits. The following are available:

- **Continuous 12 V**: **BAT +** and **–** provide a connection to the unregulated, nominal 12 V battery. It may rise above or drop below the power requirement of the sensor or peripheral.
- **SW12**: program-controlled, switched 12 Vdc terminal. Often used to power devices such as sensors that require 12 Vdc during measurement. Voltage on a **SW12** terminal will change with datalogger supply voltage. CRBasic instruction **`SW12()`** controls the **SW12** terminal (see the CRBasic instruction for **`SW12()`** for more information).

- **VX** terminals: supply precise output voltage used by analog sensors to generate high resolution and accurate signals. In this case, these terminals are regularly used with resistive-bridge measurements (see "Resistance Measurements" on page 41 for more information). Using the `SWVX()` instruction, **VX** terminals can also be used to supply a selectable, switched, regulated 3.3 or 5 Vdc power source to power digital sensors and toggle control lines (see the CRBasic instruction for `SWVX()` for more information).
- **C**, **SE 1-4**, and **P_SW** terminals: can be set low or high as output terminals (**SE 1-4** and **P_SW** to 3.3 V, and **C** to 5 V). With limited drive capacity, digital output terminals are normally used to operate external relay-driver circuits. Drive current and high-state voltage levels vary between terminals. See also "Digital Input/Output Specifications" on page 93.

See also "Power Requirements" on page 87.

Grounds

Proper grounding lends stability and protection to a data acquisition system. Grounding the datalogger with its peripheral devices and sensors is critical in all applications. Proper grounding will ensure maximum ESD protection and measurement accuracy. It is the easiest and least expensive insurance against data loss, and often the most neglected. The following terminals are provided for connection of sensor and datalogger grounds:

- **Signal Ground (⏚)** - reference for single-ended analog inputs, excitation returns, and as a ground for sensor shield wires.
- **Power Ground (G)** - return for 3.3 V, 5 V, 12 V, current loops, and digital sensors. Use of **G** grounds for these outputs minimizes potentially large current flow through the analog-voltage-measurement section of the wiring panel, which can cause single-ended voltage measurement errors.
- **Earth Ground Lug (⏚)** - connection point for heavy-gauge earth-ground wire. A good earth connection is necessary to secure the ground potential of the datalogger and shunt transients away from electronics. 14 AWG wire, minimum, is recommended.

Note: Several ground wires can be connected to the same ground terminal.

A good earth (chassis) ground will minimize damage to the datalogger and sensors by providing a low-resistance path around the system to a point of low potential. Campbell Scientific recommends that all dataloggers be earth (chassis) grounded. All components of the system (dataloggers, sensors, external power supplies, mounts, housings, etc.) should be referenced to one common earth (chassis) ground.

In the field, at a minimum, a proper earth ground will consist of a 5-foot copper-sheathed grounding rod driven into the earth and connected to the large brass ground lug on the wiring panel with a 14 AWG wire. In low-conductive substrates, such as sand, very dry soil, ice, or rock, a single ground rod will probably not provide an adequate earth ground. For these situations, search for published literature on lightning protection or contact a qualified lightning-protection consultant.

In laboratory applications, locating a stable earth ground is challenging, but still necessary. In older buildings, new Vac receptacles on older Vac wiring may indicate that a safety ground exists when, in fact, the socket is not grounded. If a safety ground does exist, good practice dictates the verification that it carries no current. If the integrity of the Vac power ground is in doubt, also ground the system through the building plumbing, or use another verified connection to earth ground.

See also:

- "Minimizing Ground Loop Errors" on page 74
- "Minimizing Ground Potential Differences" on page 76

Communication Ports

The datalogger is equipped with ports that allow communication with other devices and networks, such as:

- Computers
- Smart sensors
- Modbus and DNP3 networks
- Ethernet (CR310)
- Modems
- Campbell Scientific PakBus networks
- Other Campbell Scientific dataloggers

Campbell Scientific datalogger communication ports include:

- RS-232
- USB
- Ethernet
- **C** terminals

USB PORT

One micro-B USB port, labeled **USB**, for communicating with a computer through datalogger support software or through virtual Ethernet (RNDIS), and providing 5 Vdc power to the datalogger (powering through the USB port has limitations - details are available in the specifications). The datalogger USB port does not support USB flash or thumb drives. The USB connection supplies 5 V power as well as a communication link, which is adequate for s and making some measurements. but a 12V battery will be needed for field deployment.

ETHERNET PORT

CR310 models include one RJ45 **10/100 Ethernet** port used for IP communications.

C TERMINALS FOR COMMUNICATIONS

C terminals are configurable for the following communication types:

- SDI-12
- RS-232 (0 to 5 V)

Some communication types require more than one terminal, and some are only available on specific terminals. This is shown in the datalogger specifications.

SDI-12 Port

SDI-12 is a 1200 baud protocol that supports many smart sensors. **C1** and **C2** can each be configured as an **SDI-12** communication port. Maximum cable lengths depend on the number of sensors connected, the type of cable used, and the environment of the application. Refer to the sensor manual for guidance.

For more information, see "SDI-12 Communications" on page 54 and "Communications Specifications" on page 94.

RS-232 PORT

RS-232 represents a loose standard defining how two computing devices can communicate with each other. For instruction on setting up RS-232 communications with a computer, see "USB or RS-232 Communications" on page 11.

One nine-pin DCE port, labeled **RS-232**, normally used to communicate with a computer running datalogger support software, to connect a modem, or to read a smart sensor. The **RS-232** port functions as either a DCE or DTE device. The most common use of the **RS-232** port is as a connection to a computer DTE device (using a standard DB9-to-DB9 cable). Pins 1, 4, 6, and 9 function differently than a standard DCE device in order to accommodate a connection to a modem or other DCE device via a null modem. For the **RS-232** port to function as a DTE device, a null modem adapter is required.

RS-232 communications normally operate well up to a transmission cable capacitance of 2500 picofarads, or approximately 50 feet of commonly available serial cable.

RS-232 Power States

Under normal operation, the RS-232 port is powered down waiting for input. Upon receiving input, there is a 40-second software timeout before shutting down. The 40-second timeout is generally circumvented when communicating with datalogger support software because it sends information as part of the protocol that lets the datalogger know it can shut down the port.

When in sleep mode, hardware is configured to detect activity and wake up. Sleep mode has the penalty of losing the first character of the incoming data stream. PakBus takes this into consideration in the "ring packets" that are preceded with extra sync bytes at the start of the packet. **`SerialOpen()`** leaves the interface powered-up, so no incoming bytes are lost.

When the logger has data to send via RS-232, if the data are not a response to a received packet, such as sending a beacon, then it will power up the interface, send the data, and return to sleep mode with no 40 second timeout.

See also "Wiring Panel and Terminal Functions" on page 4.

Programmable Logic Control

The datalogger can control instruments and devices such as:

- Controlling wireless cellular modem or GPS receiver to conserve power.
- Triggering a water sampler to collect a sample.
- Triggering a camera to take a picture.
- Activating an audio or visual alarm.
- Moving a head gate to regulate water flows in a canal system.
- Controlling pH dosing and aeration for water quality purposes.
- Controlling a gas analyzer to stop operation when temperature is too low.
- Controlling irrigation scheduling.

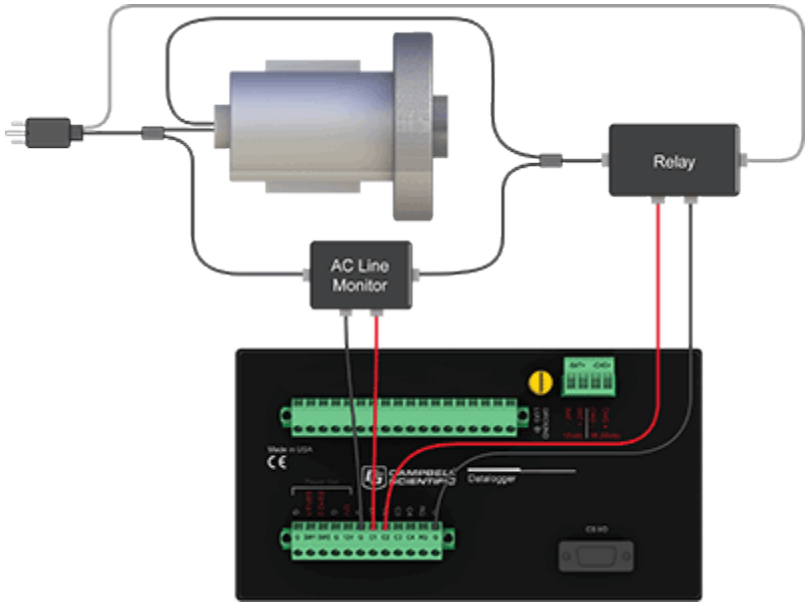
Control decisions can be based on time, an event, or a measured condition. Controlled devices can be physically connected to **C**, **VX**, **SE1** -**SE4**, **P_SW**, or **SW12** terminals. Short Cut has provisions for simple on/off control. Control modules and relay drivers are available to expand and augment datalogger control capacity.

- **C** terminals are selectable as binary inputs, control outputs, or communication ports. These terminals can be set low (0 Vdc) or high (5 Vdc) using the **`PortSet()`** or **`WriteIO()`** instructions (see **`PortSet()`** and **`WriteIO()`** in the CRBasic help for more information). Other functions include device-driven interrupts, asynchronous communications and SDI-12 communications. A

C terminal configured for digital I/O is normally used to operate an external relay-driver circuit because the terminal itself has limited drive capacity.

- VX terminals can be set low or high using the `PortSet()` or `SWVX()` instruction (see **`PortSet()`** and **`SWVX()`** in the CRBasic help for more information).
- SW12 terminals can be set low (0 V) or high (12 V) using the `SW12()` instruction (see **`SW12()`** in the CRBasic help for more information).

The following image illustrates a simple application wherein a C terminal configured for digital input, and another configured for control output are used to control a device (turn it on or off) and monitor the state of the device (whether the device is on or off).



EXAMPLE: TURN MODEM ON FOR 10 MINUTES HOURLY

In the case of a cell modem, control is based on time. The modem requires 12 Vdc power, so connect its power wire to a datalogger SW12 terminal. The following code snip turns the modem on for the first ten minutes of every hour using the `TimeIsBetween()` instruction embedded in an **If/Then** logic statement:

```
If TimeIsBetween (0,10,60,Min)Then
  SW12(1) 'Turn phone on.
Else
  SW12(0) 'Turn phone off.
EndIf
```

Setting Up the Datalogger

The basic steps to setting up your datalogger to measure data include:

1. Configuring your connection.
2. Testing communication (optional).
3. Connecting the datalogger to the computer.
4. Creating a program.
5. Sending that program to the datalogger.

Setting up Communications with the Datalogger

The first step in setting up and communicating with your datalogger is to configure your connection. Communication peripherals, dataloggers, and software must all be configured for communication. In addition to this instruction and manuals for your specific peripherals, refer to your datalogger support software manual and help for guidance.

The default settings for the datalogger allow it to communicate with a computer via USB, RS-232, or Ethernet (on CR310 models), and to accept and execute user application programs. For other communication methods or more complex applications, some settings may need adjustment. Settings can be changed through Device Configuration Utility or through datalogger support software.

You can configure your connection using any of the following options. The simplest is via USB. For detailed instruction, see:

- "USB or RS-232 Communications" on page 11
- "Virtual Ethernet over USB (RNDIS)" on page 13
- "Ethernet Communications" on page 13 (CR310 models only)
- "Wi-Fi Communications" on page 16 (WIFI models only)
- "Cellular Communications" on page 19 (CELL models only)
- "Radio Communications" on page 21 (RF models only)

For other configurations, see the LoggerNet EZSetup Wizard help. Context-specific help is given in each step of the wizard by clicking the **Help** button in the bottom right corner of the window. For complex datalogger networks, use Network Planner.

USB OR RS-232 COMMUNICATIONS

Setting up a USB or RS-232 connection is a good way to begin communicating with your datalogger. Because these connections do not require configuration (like an IP address), you need only set up the communication between your computer and the datalogger. Watch a video or use the following instructions.

Initial setup instruction follows. These settings can be revisited using the datalogger support software **Edit Datalogger Setup** option ( ).

1. Using datalogger support software, launch the EZSetup Wizard.
   - PC200W and PC400 users, click the **Add Datalogger** button ( ).
   - LoggerNet users, click the **Setup** ( ) option, click the **View** menu to ensure you are in the **EZ (Simplified)** view, then click the **Add Datalogger** button.
2. Click **Next**.
3. Select your datalogger from the list, type a name for your datalogger (for example, a site or project name), and click **Next**.
4. If prompted, select the **Direct Connect** connection type and click **Next**.
5. If this is the first time connecting this computer to a CR300 series via USB, click the **Install USB Driver** button, select your datalogger, click **Install**, and follow the prompts to install the USB drivers.
6. Plug the datalogger into your computer using a USB or RS-232 cable. The USB connection supplies 5 V power as well as a communication link, which is adequate for setup, but a 12V battery will be needed for field deployment. If using RS-232, external power must be provided to the datalogger.

   Note: The **Power** LED on the datalogger indicates the program and power state. Because the datalogger ships with a program set to run on power-up, the **Power** LED flashes 3 times every 10 seconds when powered over USB. When powered with a 12 V battery, it flashes 1 time every 10 seconds.
7. From the **COM Port** list, select the COM port used for your datalogger.
8. USB and RS-232 connections do not typically require a **COM Port Communication Delay** - this allows time for the hardware devices to "wake up" and negotiate a communication link. Accept the default value of **00 seconds** and click **Next**.
9. The baud rate and PakBus address must match the hardware settings for your datalogger. A USB connection does not require a baud rate selection, RS-232 connections default to 115200 baud, and the default PakBus address is **1**.
   - Set an **Extra Response Time** if you have a difficult or marginal connection and you want the datalogger support software to wait a certain amount of time before returning a communication failure error.
   - LoggerNet and PC400 users can set a **Max Time On-Line** to limit the amount of time the datalogger remains connected. When the datalogger is contacted, communication with it is terminated when this time limit is exceeded. A value of **0** in this field indicates that there is no time limit for maintaining a connection to the datalogger.
10. Click **Next**.
11. By default, the datalogger does not use a security code or a PakBus encryption key. Therefore, the **Security Code** can be set to **0** and the **PakBus Encryption Key** can be left blank. If either setting has been changed, enter the new code or key. See "Datalogger Security" on page 58 for more information.
12. Click **Next**.

13. Review the **Communication Setup Summary**. If you need to make changes, click the **Previous** button to return to a previous window and change the settings.

Setup is now complete, and the EZSetup Wizard allows to you click **Finish** or click **Next** to test your communication, set the datalogger clock, and send a program to the datalogger.

VIRTUAL ETHERNET OVER USB (RNDIS)

CR300 series dataloggers with OS version 6 or greater support RNDIS (virtual Ethernet over USB). This allows the datalogger to communicate via TCP/IP over USB. Watch a video or use the following instructions.

Connecting to Your Datalogger via RNDIS

1. Supply power to the datalogger. If connecting via USB for the first time, you must first install USB drivers by using Device Configuration Utility (select your datalogger, then on the main page, click **Install USB Driver**). Alternately, you can install the USB drivers using EZ Setup. A USB connection supplies 5 V power (as well as a communication link), which is adequate for setup, but a 12 V battery will be needed for field deployment.

   Note: Ensure the datalogger is connected directly to the computer USB port (not to a USB hub).

2. Physically connect your datalogger to your computer using a USB cable, then open Device Configuration Utility and select your datalogger.
3. Select the communication port used to communicate with the datalogger from the **COM Port** list.
4. Press the **Connect** button, click the **Settings Editor** tab, click the **Advanced** sub-tab, click the **USB Configuration** list and select **Virtual Ethernet (RNDIS)**.
5. Click **Apply**.
6. Retrieve your IP Address. On the bottom, left side of the screen, select **Use IP Connection**, then click the browse button next to the **Communication Port** box. Note the IP Address (default is **192.168.66.1**). If you have multiple dataloggers in your network, more than one datalogger may be returned. Ensure you select the correct datalogger by verifying the datalogger serial number or station name (if assigned).
7. A datalogger's Virtual IP address can be used to connect to it using Device Configuration Utility or other computer software, or to view the datalogger's internal web page in a browser. To view the web page, open a browser and type the IP address you retrieved in the previous step (for example, **192.168.66.1**) into the address bar.

To secure your datalogger from access by others who have access to your network, we recommend that you set security settings and establish access permissions using `.csipasswd`. For more information, see "Creating a .csipasswd File" on page 60.

ETHERNET COMMUNICATIONS

The CR310 offers a 10/100 Ethernet connection you can configure using Device Configuration Utility. You will use this utility to enter the datalogger IP Address, Subnet Mask, and IP Gateway address. After this, you can use the EZSetup Wizard to set up communications with the datalogger. If you already have the datalogger IP information, you can skip these steps and go directly to "Setting up Ethernet Communications between the Datalogger and the Computer" on page 15.

Configuring Datalogger Ethernet Settings

1. Supply power to the datalogger. If connecting via USB for the first time, you must first install USB drivers by using Device Configuration Utility (select your datalogger, then on the main page, click **Install USB Driver**). Alternately, you can install the USB drivers using EZ Setup. A USB connection supplies 5 V power (as well as a communication link), which is adequate for setup, but a 12 V battery will be needed for field deployment.
2. Connect an Ethernet cable to the **10/100 Ethernet** port on the datalogger. The yellow and green **Ethernet** port LEDs display activity approximately one minute after connecting. If you do not see activity, contact your network administrator. For more information, see "Ethernet LEDs" on page 14.
3. Using datalogger support software (LoggerNet, PC400, or PC200W), enter the Device Configuration Utility ( ).
4. Select the **CR300 Series** datalogger from the list
5. Select the port assigned to the datalogger from the **Communication Port** list. If connecting via Ethernet, check the **Use IP Connection** button.
6. By default, this datalogger does not use a PakBus encryption key, so the **PakBus Encryption Key** box can be left blank. If this setting has been changed, enter the new code or key. See "Datalogger Security" on page 58 for more information.
7. Click **Connect**.
8. On the **Deployment** tab, click the **Ethernet** subtab.
9. The **Ethernet Power** setting allows you to reduce the power consumption of the datalogger. If there is no Ethernet connection, the datalogger will turn off its Ethernet interface for the time specified before turning it back on to check for a connection. Select an option from the list.
10. Enter the **IP Address**, **Subnet Mask**, and **IP Gateway**. These values should be provided by your network administrator. A static IP address is recommended. If you are using DHCP, note the IP address assigned to the datalogger on the right side of the window. When the IP Address is set to 0.0.0.0, the default, the information displayed on the right side of the window updates with the information obtained from the DHCP server. Note, however, that this address is not static and may change. An IP address here of 169.254.###.### means the datalogger was not able to obtain an address from the DHCP server. Contact your network administrator for help.
11. Click **Apply** to save your changes.

Ethernet LEDs

When the datalogger is powered, the **10/100 Ethernet** port will turn on with Ethernet activity:

- **Solid Yellow**: Valid Ethernet link.
- **No Yellow**: Invalid Ethernet link.
- **Flashing Yellow**: Ethernet activity.
- **Solid Green**: 100 Mbps link.
- **No Green**: 10 Mbps link.

Setting up Ethernet Communications between the Datalogger and the Computer

Once you have configured the Ethernet settings or obtained the IP information for your datalogger, you can set up communication between your computer and the datalogger over Ethernet.

Initial setup instruction follows. These settings can be revisited using the datalogger support software **Edit Datalogger Setup** option ().

1. Using datalogger support software, launch the EZSetup Wizard.
   - PC400 users, click the **Add Datalogger** button ().
   - LoggerNet users, click the **Setup** () option, click the **View** menu to ensure you are in the **EZ (Simplified)** view, then click the **Add Datalogger** button.

   Note: PC200W does not allow IP connections.
2. Click **Next**.
3. Select the **CR300 Series** from the list, type a name for your datalogger (for example, a site or project name), and click **Next**.
4. Select the **IP Port** connection type and click **Next**.
5. Type the datalogger IP address followed by a colon, then the port number of the datalogger in the **Internet IP Address** field (these were set up when "Ethernet Communications" on page 13). They can be accessed in Device Configuration Utility on the **Ethernet** subtab. Leading 0s must be omitted. For example:
   - IPv4 addresses are entered as `192.168.1.2:6785`
   - IPv6 addresses must be enclosed in square brackets. They are entered as `[2001:db8::1234:5678]:6785`
6. The PakBus address must match the hardware settings for your datalogger. The default PakBus address is **1**.
   - Set an **Extra Response Time** if you want the datalogger support software to wait a certain amount of time before returning a communication failure error.
   - LoggerNet and PC400 users can set a **Max Time On-Line** to limit the amount of time the datalogger remains connected. When the datalogger is contacted, communication with it is terminated when this time limit is exceeded. A value of **0** in this field indicates that there is no time limit for maintaining a connection to the datalogger.
7. Click **Next**.
8. By default, the datalogger does not use a security code or a PakBus encryption key. Therefore the **Security Code** can be set to **0** and the **PakBus Encryption Key** can be left blank. If either setting has been changed, enter the new code or key. See "Datalogger Security" on page 58 for more information.
9. Click **Next**.
10. Review the **Communication Setup Summary**. If you need to make changes, click the **Previous** button to return to a previous window and change the settings.

Setup is now complete, and the EZSetup Wizard allows you click **Finish** or click **Next** to test your communication, set the datalogger clock, and send a program to the datalogger. See "Testing Communication and Completing EZ Setup" on page 26 for more information.

WI-FI COMMUNICATIONS

By default, CR300 series-WIFI dataloggers are configured to host a Wi-Fi network. The LoggerLink mobile app for iOS and Android can be used to connect with a CR300 series-WIFI. Up to eight devices can connect to a network created by a CR300 series. The setup follows the same steps shown in this video: CR6-WIFI Datalogger - Setting Up a Network.

To configure a CR300 series to communicate with a network of devices over Wi-Fi:

- Ensure your datalogger is connected to an antenna and power.
- Configure the datalogger to host a Wi-Fi network using Device Configuration Utility.
- Connect your computer to the datalogger over Wi-Fi.
- Set up communication between the CR300 series datalogger and the datalogger support software.
- Configure additional dataloggers to join the Wi-Fi network.

Note: The user is responsible for emissions if changing the antenna type or increasing the gain.

See also "Communications Specifications" on page 94.

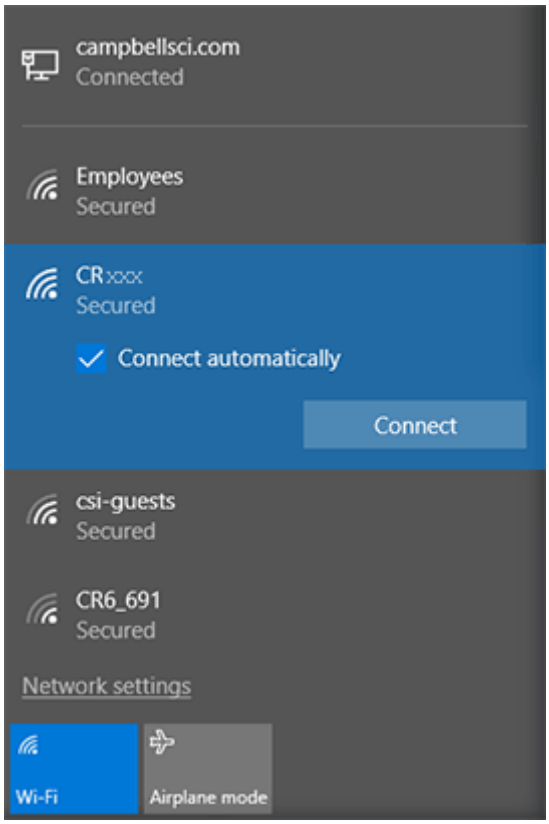
Configuring the Datalogger to Host a Wi-Fi Network

By default, CR300-WIFI dataloggers are configured to host a Wi-Fi network. If the settings have changed, you can follow these instructions to reconfigure the datalogger:

1. Ensure your CR300-WIFI is connected to an antenna and power.
2. Using Device Configuration Utility, connect to the datalogger.
3. On the **Deployment** tab, click the **Wi-Fi** sub-tab.
4. In the **Configuration** list, select the **Create a Network** option.
5. Optionally, set security on the network to prevent unauthorized access by typing a password in the **Password** box (recommended).
6. Apply your changes.

Connecting Your Computer to the Datalogger over Wi-Fi

1. Open the Wi-Fi network settings on your computer.



2. Select the Wi-Fi-network hosted by the datalogger. The default name is **CR300** followed by the serial number of the datalogger. In the previous image, the Wi-Fi network is **CRxxx**.
3. If you set a password, select the **Connect Using a Security Key** option (instead of a PIN) and type the password you chose.
4. Connect to this network.

Setting up Wi-Fi Communication between the Datalogger and the Datalogger Support Software

1. Using LoggerNet or PC400, click the **Add Datalogger** button ( ) to launch the EZSetup Wizard. For LoggerNet users, you must first click the **Setup** ( ) option, click the **View** menu to ensure you are in the **EZ (Simplified)** view, then click the **Add Datalogger** button.

   Note: PC200W does not allow IP connections.

2. Select the **IP Port** connection type and click **Next**.
3. In the **Internet IP Address** field, type `192.168.67.1`. This is the default datalogger IP address created when the CR300-WIFI creates a network.
4. Click **Next**.

5. The PakBus address must match the hardware settings for your datalogger. The default PakBus address is **1**.
    - Set an **Extra Response Time** if you want the datalogger support software to wait a certain amount of time before returning a communication failure error.
    - You can set a **Max Time On-Line** to limit the amount of time the datalogger remains connected. When the datalogger is contacted, communication with it is terminated when this time limit is exceeded. A value of **0** in this field indicates that there is no time limit for maintaining a connection to the datalogger.
6. Click **Next**.
7. By default, the datalogger does not use a security code or a PakBus encryption key. Therefore, the **Security Code** can be set to **0** and the **PakBus Encryption Key** can be left blank. If either setting has been changed, enter the new code or key. See "Datalogger Security" on page 58 for more information.
8. Click **Next**.
9. Review the **Communication Setup Summary**. If you need to make changes, click the **Previous** button to return to a previous window and change the settings.

Setup is now complete, and the EZSetup Wizard allows you click **Finish** or click **Next** to test your communication, set the datalogger clock, and send a program to the datalogger. See "Testing Communication and Completing EZ Setup" on page 26 for more information.

Configuring Dataloggers to Join a Wi-Fi Network

By default, the CR300-WIFI dataloggers are configured to host a Wi-Fi network. To set them up to join a network:

1. Ensure your CR300-WIFI is connected to an antenna and power.
2. Using Device Configuration Utility, connect to the datalogger.
3. On the **Deployment** tab, click the **Wi-Fi** sub-tab.
4. In the **Configuration** list, select the **Join a Network** option.
5. Next to the **Network Name (SSID)** box, click the **Browse** button () to search for and select a Wi-Fi network.
6. If the network is a secured network, you must enter the password in the **Password** box and add any additional security in the **Enterprise** section of the window.
7. Enter the **IP Address, Network Mask**, and **Gateway**. These values should be provided by your network administrator. A static IP address is recommended.
    - Alternatively, you can use an IP address assigned to the datalogger via DHCP. To do this, make sure the IP Address is set to `0.0.0.0`. Click **Apply** to save the configuration changes. Then reconnect. The IP information obtained through DHCP is updated and displayed in the **Status** section of the **Wi-Fi** subtab. Note, however, that this address is not static and may change. An IP address here of **169.254.###.###** means the datalogger was not able to obtain an address from the DHCP server. Contact your network administrator for help.
8. Apply your changes.
9. For each datalogger you want to connect to network, you must follow the instruction in "Setting up Wi-Fi Communication between the Datalogger and the Datalogger Support Software" on page 17, using the IP address used to configure that datalogger (step 7 in this instruction).

Wi-Fi LED Indicator

When the datalogger is powered, the Wi-Fi LED will turn on according to Wi-Fi communication states:

- **Off**: Insufficient power, Wi-Fi disabled, or datalogger failed to join or create a network (periodic retries will occur).
- **Solid for 2 seconds**: Attempting to join or create a network.
- **Flashing**: Successfully joined or created a network. Flashes with network activity and once every four seconds.

CELLULAR COMMUNICATIONS

In addition to the IP capabilities inherent in the cellular connection (CELL models only), you can use the **SMSSend()** and **SMSRecv()** instructions to send SMS messages and receive and store messages in a string variable. Note that the **SMSSend()** and **SMSRecv()** instructions require a cellular account that includes text messaging capabilities (many cellular data accounts do not include this option). Contact your cellular provider for more information on your plan. See the CRBasic help for further instruction on **SMSSend()** and **SMSRecv()**.

Note: The -CELL200 and -CELL205 options are not compatible with the Verizon cellular network. See "Cellular Option Specifications" on page 94 for more information.

1. Obtain an access point name (APN) from your cellular network provider. You will need this to configure the connection.
2. Insert a supported SIM card into the back of the unit.

3. Ensure your CR300-CELL is connected to an antenna and power.
4. Plug the datalogger into your computer using a USB or RS-232 cable. Supply power to the datalogger. Note that a USB connection supplies 5V power (as well as a communication link), which is adequate for setup but a 12V battery will be needed to connect to the network and obtain the IP address.
5. If this is the first time connecting this computer to your datalogger via USB, you need to update the USB drivers. Open Device Configuration Utility, select your datalogger, and on the main page, click **Install USB Driver**.

6. If you have a Fixed IP SIM Card and know the IP address of the card, proceed to step 7. Otherwise:
    - Connect to the datalogger using Device Configuration Utility.
    - Click the **Deployment** tab, then click the **Cellular** sub-tab. Type your **APN** into the box, then click **Apply**. This will disconnect the datalogger from Device Configuration Utility.
    - Reconnect the datalogger using Device Configuration Utility. From the same sub-tab, copy the **IP address** from the **Cellular Network Status** box, then click the **Disconnect** button.
7. Using datalogger support software, launch the EZSetup Wizard.
    - PC400 users, click the **Add Datalogger** button ().
    - LoggerNet users, click the **Setup** () option, click the **View** menu to ensure you are in the **EZ (Simplified)** view, then click the **Add Datalogger** button.

    Note: PC200W does not allow IP connections.
8. Click **Next**.
9. Select the datalogger from the list, type a name for your datalogger (for example, a site or project name), and click **Next**.
10. Select the **IP Port** connection type and click **Next**.
11. Type or paste the IP Address (see step 6) into the box, append it with a colon and port number 6785 (for example 192.168.67.1:6785), and click **Next**.
12. The PakBus address must match the hardware settings for your datalogger. The default PakBus address is **1**.
    - Set an **Extra Response Time** if you want the datalogger support software to wait a certain amount of time before returning a communication failure error.
    - You can set a **Max Time On-Line** to limit the amount of time the datalogger remains connected. When the datalogger is contacted, communication with it is terminated when this time limit is exceeded. A value of **0** in this field indicates that there is no time limit for maintaining a connection to the datalogger.
13. Click **Next**.
14. By default, the datalogger does not use a security code or a PakBus encryption key. Therefore, the **Security Code** can be set to **0** and the **PakBus Encryption Key** can be left blank. If either setting has been changed, enter the new code or key. See "Datalogger Security" on page 58 for more information.
15. Click **Next**.
16. Review the **Communication Setup Summary**. If you need to make changes, click the **Previous** button to return to a previous window and change the settings.
17. Click **Next** to test communications and set the clock (see "Testing Communication and Completing EZ Setup" on page 26 for more information).
18. Connect to the datalogger using datalogger support software (see "Connecting the Datalogger to a Computer" on page 27).

Cellular (TX/RX) LED Indicator

When the datalogger is powered, the cellular LED will turn on according to cellular modem communication states:

- **Off**: Cellular modem off, insufficient power, or failure to establish a connection with the provider (periodic retries will occur).
- **Solid**: Cellular modem is powering up and attempting to establish a connection with a provider.
- **Quick Flashing** (approximately 1 second duration): Indicates successful network registration.
- **Flashing**: Flashes with network activity.

See "Cellular Option Specifications" on page 94 for more information.

RADIO COMMUNICATIONS

CR300 series-RF dataloggers include radio options. The RF407-series frequency-hopping spread-spectrum (FHSS) radio options include the RF407, RF412, RF422, and RF427. RF407-series are designed for license-free use in several countries:

- The RF407 option has a 902 to 928 MHz operating-frequency range appropriate for use in the United States and Canada (FCC / IC compliant).
- The RF412 option has a 915 to 928 MHz operating-frequency range appropriate for use in Australia and New Zealand (ACMA compliant).
- The RF422 option has an 863 to 873 MHz operating-frequency range appropriate for use in most of Europe and some of Asia (ETSI compliant).
- The RF427 option has a 902 to 907.5 MHz/915 to 928 MHz operating-frequency range appropriate for use in Brazil.

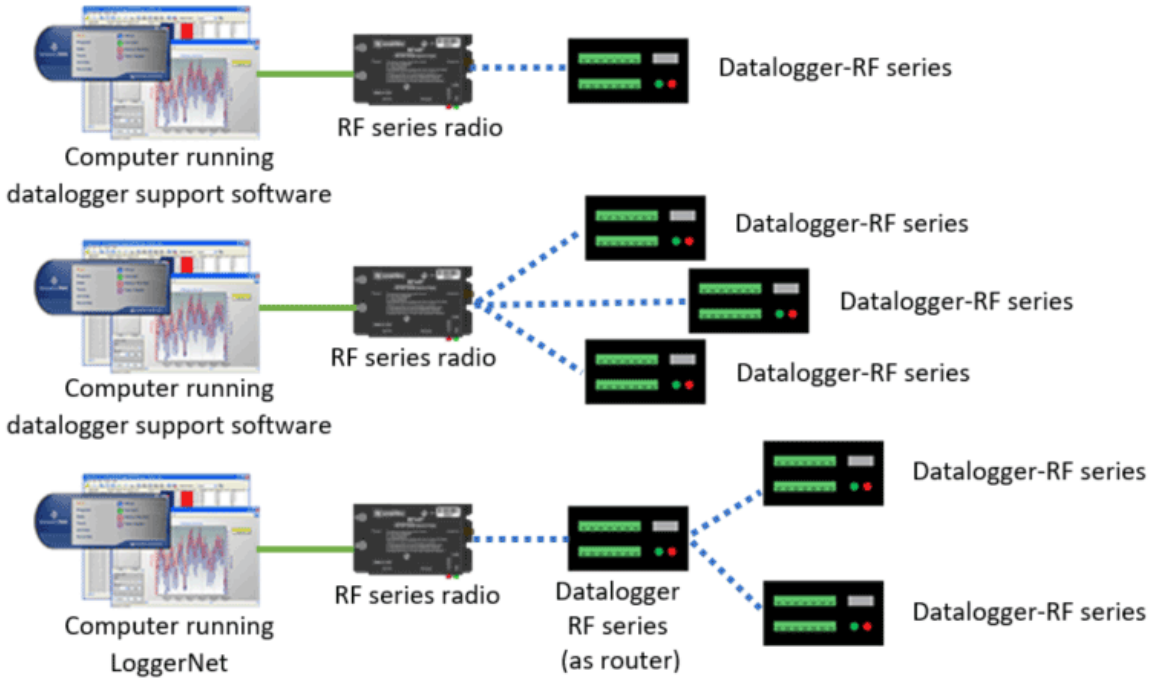
Note: This equipment has been tested and found to comply with the limits for a Class A digital device, pursuant to part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference when the equipment is operated in a commercial environment. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instruction manual, may cause harmful interference to radio communications. Operation of this equipment in a residential area is likely to cause harmful interference in which case the user will be required to correct the interference at his or her own expense.

Radio options cannot be mixed within a network. An RF407 can only be used with other RF407-type radios, an RF412 can only be used with other RF412-type radios, an RF422 can only be used with other RF422-type radios, and an RF427 can only be used with other RF427-type radios.

Throughout these instructions, RF407-series represents each of the RF407, RF412, RF422, and RF427 radio options, unless otherwise noted. Similarly, the RF407-series standalone, or independent radio represents each of the RF407, RF412, RF422, and RF427 models, unless otherwise noted.

Configuration Options

The most frequently used configurations with the RF-series datalogger and RF-series radio include the following:



For detailed instruction, see:

- "RF407-Series Radio Communications with One or More Dataloggers" on page 22
- "RF407-Series Radio Communications with Multiple Dataloggers Using One Datalogger as a Router" on page 24

See also "RF Radio Option Specifications" on page 95.

RF407-Series Radio Communications with One or More Dataloggers

To configure an RF407-series radio to communicate with the datalogger, you must complete the following steps (instruction follows):

- Ensure your datalogger and RF407-series radio are connected to an antenna and power.
- Configure the connection to the RF407-series device using Device Configuration Utility.
- If you are connecting to multiple dataloggers, you will have to assign unique PakBus addresses to each datalogger using Device Configuration Utility. (Connect to each datalogger, set the **PakBus Address** on the **Deployment** | **Datalogger** tab.)
- Use datalogger support software to set up communications between the RF407-series radio and the dataloggers.

Note: This procedure assumes the RF407 series devices are using factory default settings.

Configuring the RF407-Series Radio

Configure the RF407-Series radio connected to the computer (see image in "Configuration Options" on page 22 for reference).

1. Ensure your RF407-series radio is connected to an antenna and power.

2. If connecting via USB for the first time, you must first install USB drivers using Device Configuration Utility (select your radio, then on the main page, click **Install USB Driver**). Plug the RF407-series radio to your computer using a USB or RS-232 cable.
3. Using Device Configuration Utility, select the **Communication Port** used for your radio and connect to the RF407-series radio.
4. On the **Main** tab, set the **Active Interface** to **USB** or **RS-232** (depending on how your computer will be connected to the RF407-series radio).
5. Apply the changes.
6. Connect the RF407-Series radio to the computer communication port selected in the previous step.

Setting up Communication between the RF407-Series Datalogger and the Computer

These instructions provide an easy way to set up communication between the RF407-series datalogger and the computer connected to the RF407-series radio (as configured in previous instructions). Follow these instructions multiple times to set up multiple dataloggers. In this case, each datalogger must be given a unique PakBus address (see "PakBus Communications" on page 53 for more information). For more complicated networks, it is recommended that you use Network Planner.

1. Supply 12 Vdc power to the datalogger.
2. Ensure the datalogger antenna is connected.
3. Using datalogger support software, launch the EZSetup Wizard and add the datalogger.
   - PC200W and PC400 users, click the **Add Datalogger** button ().
   - LoggerNet users, click the **Setup** () option, click the **View** menu to ensure you are in the **EZ (Simplified)** view, then click the **Add Datalogger** button.
4. Click **Next**.
5. Select the **CR300Series** datalogger from the list, type a name for your datalogger (for example, a site or project name), and click **Next**.
6. If prompted, select the **Direct Connect** connection type and click **Next**.
7. Select the communication port used to communicate with the RF407-series radio from the **COM Port** list. (Note that the RF407-series radio to RF407-series datalogger link is not indicated in the LoggerNet Setup Standard View.)
8. Accept the default value of **00 seconds** in the **COM Port Communication Delay** - this box is used to allow time for hardware devices to "wake up" and negotiate a communication link. Click **Next**.
9. In the previous instruction "Configuring a Connection to an RF407-Series Radio," you were asked to select an active interface option of USB or RS-232. If you selected USB as the active interface for the radio, you do not need to select a baud rate. If you selected RS-232, set the baud rate to the one chosen during that step. The radio's default baud rate is 115200. The PakBus address must match the hardware settings for your datalogger. The default **PakBus Address** is **1**.
10. Click **Next**.
11. By default, the datalogger does not use a security code or a PakBus encryption key. Therefore, the **Security Code** can be set to **0** and the **PakBus Encryption Key** can be left blank. If either setting has been changed, enter the new code or key. See "Datalogger Security" on page 58 for more information.

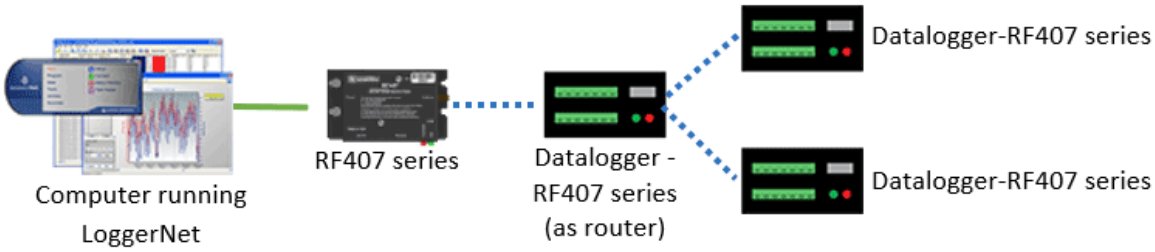
12. Click **Next**.
13. Review the **Communication Setup Summary**. If you need to make changes, click the **Previous** button to return to a previous window and change the settings.

Setup is now complete, and the EZSetup Wizard allows you to click **Finish** or click **Next** to test your communication, set the datalogger clock, and send a program to the datalogger. See "Testing Communication and Completing EZ Setup" on page 26 for more information.

If you experience network communication problems, see "Troubleshooting Radio Communication Problems" on page 72 for assistance.

RF407-Series Radio Communications with Multiple Dataloggers Using One Datalogger as a Router

This type of network configuration is useful for communicating around an obstacle, such as a hill or building, or to reach longer distances.



To configure an RF407-series radio to communicate with multiple dataloggers through a router, you must complete the following steps (instruction follows):

- Ensure your dataloggers and RF407-series radios are each connected to an antenna and power.
- Configure your connection to the RF407-series devices using Device Configuration Utility.
- Assign unique PakBus addresses to each datalogger using Device Configuration Utility. (Connect to each datalogger, and set the **PakBus Address** on the **Deployment** | **Datalogger** tab.)
- Configure the datalogger acting as a router.
- Use datalogger support software to set up communication between the computer and the dataloggers.

Configuring the RF407-Series Radio

Configure the RF407-Series radio connected to the computer (see previous image for reference).

1. Ensure your RF407-series radio is connected to an antenna and power.
2. If connecting via USB for the first time, you must first install USB drivers using Device Configuration Utility (select your radio, then on the main page, click **Install USB Driver**). Plug the RF407-series radio to your computer using a USB or RS-232 cable.
3. Using Device Configuration Utility, select the **Communication Port** used for your radio and connect to the RF407-series radio.
4. On the **Main** tab, set the **Active Interface** to **USB** or **RS-232** (depending on how your computer will be connected to the RF407-series radio).
5. Apply the changes.

6. Connect the RF407-Series radio to the computer communication port selected in the previous step.

Configuring the Datalogger Acting as a Router

1. Supply power to the datalogger. If connecting via USB for the first time, you must first install USB drivers by using Device Configuration Utility (select your datalogger, then on the main page, click **Install USB Driver**). Alternately, you can install the USB drivers using EZ Setup. A USB connection supplies 5 V power (as well as a communication link), which is adequate for setup, but a 12 V battery will be needed for field deployment. Ensure the datalogger antenna is connected.
2. Using Device Configuration Utility, connect to the RF407-series datalogger that will serve as a router.
3. On the **Deployment|Datlogger** tab, assign a unique PakBus Address (see "PakBus Communications" on page 53 for more information).
4. On the **Deployment** tab, click the **Com Ports Settings** sub-tab.
5. From the **Select the ComPort** list, select **RF**.
6. Set the **Beacon Interval** to **60** seconds (or the amount of time you are willing to wait for the leaf dataloggers in the network to be discovered).

Note: A beacon is a packet broadcast at a specified interval intended to discover neighbor devices.

7. Set the **Verify Interval** to something slightly greater than the expected communication interval between the router and the other (leaf) dataloggers in the network (for example, 90 seconds).
8. Click the **Advanced** sub-tab and set **Is Router** to **True**.
9. Apply your changes.

Adding Routing Datalogger to LoggerNet Network

1. Using LoggerNet, click the **Setup** () option and click the **View** menu to ensure you are in the **Standard** view.
2. Click the **Add Root** button ().
3. Click **ComPort**, then **PakBusPort (PakBus Loggers)**, then **CR300Series**.
4. Click **Close**.
5. In the Entire Network pane on the left side of the window, select the **ComPort**.
6. On the **Hardware** tab on the right, click the **ComPort Connection** list and select the communication port assigned to the RF407-series radio.
7. In the Entire Network pane on the left side of the window, select **PakBusPort**.
8. On the **Hardware** tab on the right, select the **PakBus Port Always Open** check box.
    - If you would like to prevent the possibility of LoggerNet communicating with any other dataloggers in the network without going through the router, set the **Beacon Interval** to **00 h 00 m 00s**.
9. In the Entire Network pane on the left side of the window, select the router datalogger (**CR300Series**) from the list.

10. On the **Hardware** tab on the right, type the **PakBus Address** you assigned to the router datalogger in Device Configuration Utility.
11. Optionally, click the **Rename** button ( ) to provide the datalogger a descriptive name.
12. Apply your changes.

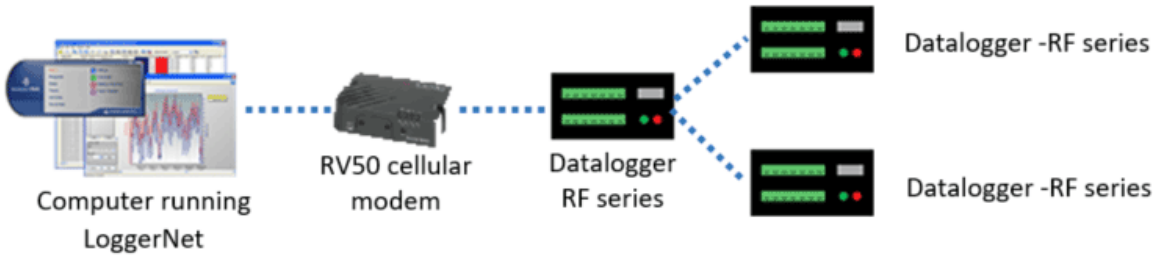
Adding Leaf Dataloggers to the Network

1. In the LoggerNet **Standard Setup** view (click the **Setup** ( ) option and click the **View** menu to ensure you are in the **Standard** view), right-click on the router datalogger in the Entire Network pane on the left side of the window and select **CR300Series**.
2. With the newly added datalogger selected in the **Entire Network** pane, set the **PakBus Address** to the address that was assigned to the leaf datalogger in Device Configuration Utility.
3. Click **Rename**. Enter a descriptive name for the datalogger.
4. Apply your changes.
5. Repeat these steps for each leaf datalogger in the network.

If you experience network communication problems, see "Troubleshooting Radio Communication Problems" on page 72 for assistance.

Using Additional Communication Methods

Using similar instructions, a RF407-series datalogger can be used in a system with additional communication methods. For example, in the following image, the router RF407-series datalogger communicates with LoggerNet through an RV50 cellular modem connected to RF407-series datalogger using the **RS-232** port. The router RF407-series datalogger communicates with the leaf RF407-series dataloggers over RF.



Testing Communication and Completing EZ Setup

1. Using datalogger support software EZ Setup, access the **Communication Test** window. This window is accessed during EZ Setup (see "USB or RS-232 Communications" on page 11 for more information). Alternatively, you can double-click a datalogger from the station list to open the EZ Setup Wizard and access the **Communication Test** step from the left side of the window.
2. Ensure the datalogger is connected to the computer, select **Yes** to test the communication, then click **Next** to initiate the test. To troubleshoot an unsuccessful test, see "Tips and Troubleshooting" on page 67.

3. With a successful connection, the **Datalogger Clock** window displays the time for both the datalogger and the computer.
    - The **Adjusted Server Date/Time** displays the current reading of the clock for the computer or server running your datalogger support software. If the **Datalogger Date/Time** and **Adjusted Server Date/Time** don't match, you can set the datalogger clock to the **Adjusted Server Date/Time** by clicking **Set Datalogger Clock**.
    - Use the **Time Zone Offset** to specify a positive or negative offset to apply to the computer time when setting the datalogger clock. This offset will allow you to set the clock for a datalogger that needs to be set to a different time zone than the time zone of the computer (or to accommodate for changes in daylight saving time).
4. Click **Next**.
5. The datalogger ships with a default **QuickStart** program. If the datalogger does not have a program, you can choose to send one by clicking the **Select and Send Program** button. Click **Next**.
6. LoggerNet only - Watch a video or use the following instructions:
    - The **Datalogger Table Output Files** window displays the data tables available to be collected from the datalogger and the output file name. To include a data table in scheduled collection, select the data table from the **Tables** list and check the **Table Collected During Data Collection** box. Select a **Data File Option**: **Append to End of File** adds new data to the end of the existing data file, **Overwrite Existing File** replaces the existing file with a newly created file, and **No Output File** results in no data file being written to disk. Make note of the **Output File Name** and location. Click **Next**.
    - Check **Scheduled Collection Enabled** to have LoggerNet collect data from the datalogger according to a schedule. Set the **Base Date** and **Time** to begin scheduled collections. Set a **Collection Interval**, then click **Next**.
7. Click **Finish**.

Connecting the Datalogger to a Computer

Once you have configured your connection (see "Setting up Communications with the Datalogger" on page 11 for more information), you can connect the datalogger to your computer.

- PC200W and PC400 users, select the datalogger from the list and click the **Connect** button ( ).
- LoggerNet users, select **Main** and click the **Connect** button ( ) on the LoggerNet toolbar, select the datalogger from the **Stations** list, then click the **Connect** button ( ).

To disconnect, click the **Disconnect** button ( ).

See also "Communication Protocols" on page 51.

Creating a Program in Short Cut

Use the Short Cut software to generate a program for your datalogger. Short Cut is included with your datalogger support software.

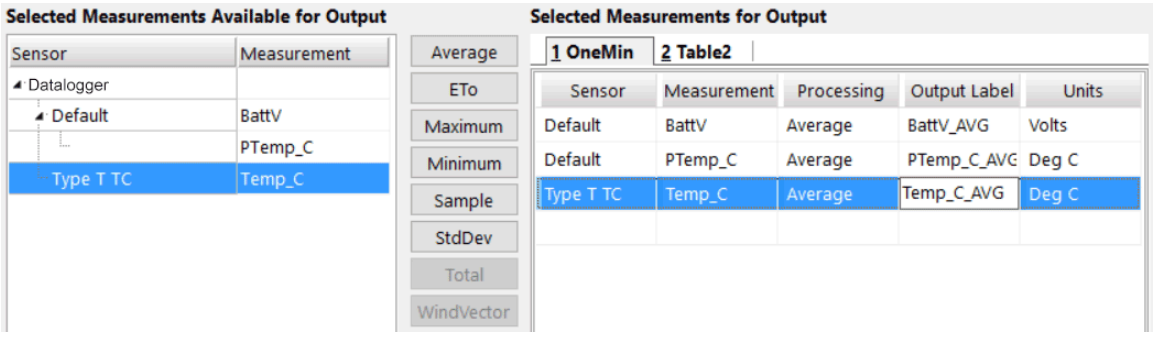
Watch a video or use the following instructions.

1. Using datalogger support software, launch Short Cut.
   - PC200W and PC400 users, click the **Short Cut** button ().
   - LoggerNet users, click **Program** then click the **Short Cut** button ().
2. Click **Create New Program**.
3. Select the **CR300 Series** datalogger and click **Next**.

> **Note**: The first time Short Cut is run, a prompt will appear asking for a choice of noise rejection. Select **60 Hz Noise Rejection** for North America and areas using 60 Hz ac voltage. **Select 50 Hz Noise Rejection** for most of the Eastern Hemisphere and areas that operate at 50 Hz.
>
> A second prompt lists sensor support options. **Campbell Scientific, Inc. (US)** is probably the best fit if you are outside Europe.
>
> To change the noise rejection or sensor support option for future programs, use the **Program** menu.

4. A list of **Available Sensors and Devices** and **Selected Measurements Available for Output** display. Battery voltage `BattV` and internal temperature `PTemp_C` are selected by default. During operation, battery and temperature should be recorded at least daily to assist in monitoring system status.
5. Use the Search feature or expand folders to locate your sensor or device. Double-click on a sensor or measurement in the **Available Sensors and Devices** list to configure the device (if needed) and add it to the **Selected** list.
6. If the sensor or device requires configuration, a window displays with configuration options. Click **Help** at the bottom of the window to learn more about any field or option.
7. Click **OK**.
8. Click **Wiring Diagram** on the left side of the window to see how to wire the sensor to the datalogger. With the power disconnected from the datalogger, insert the wires as directed in the diagram. Ensure you clamp the terminal on the conductor, not the wire insulation. Use the included flat-blade screwdriver to open/close the terminals.
9. Click **Sensors** on the left side of the window to return to the sensor selection window.
10. Use the **Output Setup** options to specify how often measurements are to be made and how often outputs are to be stored. Note that multiple output intervals can be specified, one for each output table (**Table1** and **Table2** tabs).
11. In the **Table Name** box, type a name for the table.
12. Select a **Data Output Storage Interval**.
13. Check the **Advanced Outputs** option if you want to specify the number of records and data events to store, set output intervals, specify measurements to evaluate, or set flags based on the value of a variable.

14. Click **Next**.
15. Select the measurement from the **Selected Measurements Available for Output** list, then click an output processing option to add the measurement to the **Selected Measurements for Output** list.



16. Click **Finish** to compile the program. Replace the **untitled.cr300** default name and click **Save**.
17. If LoggerNet or other datalogger support software is running on your computer, and the datalogger is connected to the computer (see "Connecting the Datalogger to a Computer" on page 27 for more information), you can choose to send the program.

> **Note**: A good practice is to always retrieve data from the datalogger before sending a program; otherwise, data may be lost. See "Collecting Data" on page 32 for detailed instruction.

If your data acquisition requirements are simple, you can probably create and maintain a datalogger program exclusively with Short Cut. If your data acquisition needs are more complex, the files that Short Cut creates are a great source for programming code to start a new program or add to an existing custom program using CRBasic. See the CRBasic Editor help for detailed information on program structure, syntax, and each instruction available to the datalogger.

> **Note**: Once a Short Cut generated program has been edited with CRBasic Editor, it can no longer be modified with Short Cut.

Sending a Program to the Datalogger

The datalogger requires a CRBasic program to direct measurement, processing, control, and data storage operations. The program file may use the extension `.CR300` or `.dld`.

A good practice is to always retrieve data from the datalogger before sending a program; otherwise, data may be lost. To collect data using LoggerNet, connect to your datalogger and click the **Collect Now** button (). Some methods of sending a program give the option to retain data when possible. Regardless of the program upload tool used, data will be erased when a new program is sent if any change occurs to one or more data table structure in the following list:

- Data table name(s)
- Data output interval or offset
- Number of fields per record
- Number of bytes per field
- Field type, size, name, or position
- Number of records in table

SENDING A PROGRAM USING DATALOGGER SUPPORT SOFTWARE

Watch a video or use the following instructions.

1. Connect the datalogger to your computer (see "Connecting the Datalogger to a Computer" on page 27 for more information).
2. Using your datalogger support software, click **Send New** or **Send Program** (located in the Current Program section on the right side of the window).
3. Navigate to the location of the program, select it, and click **Open**.
4. Confirm that you would like to proceed and erase all data tables saved on the datalogger. The program will send, compile, then display results.

After sending a program, it is a good idea to monitor the data to make sure it is measuring as you expect. See "Working with Data" on page 31 for more information.

Program Run Options

When sending a program via File Control, Short Cut, or CRBasic, you can choose to have programs **Run on Power-up** and **Run Now**. **Run Now** will run the program when it is sent to the datalogger. **Run on Power-up** runs the program when the datalogger is powered up. Selecting both **Run Now** and **Run on Power-up** will invoke both options.

Working with Data

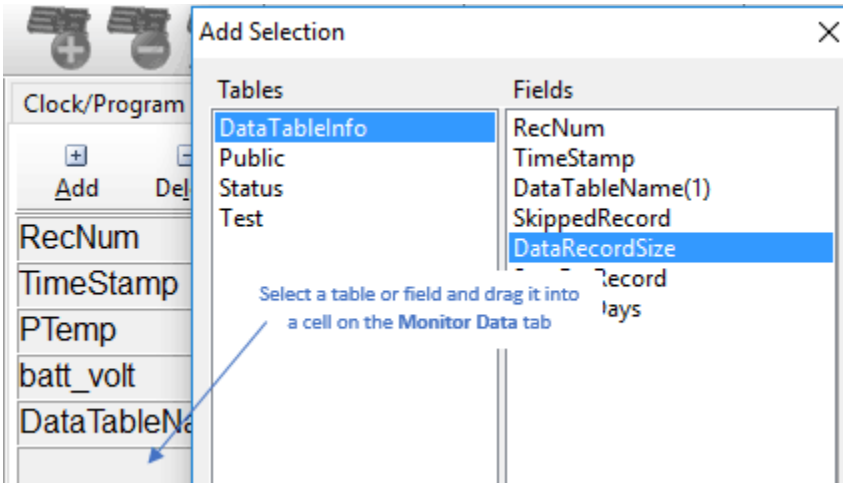
By default, the datalogger includes three tables: **Public**, **Status**, and **DataTableInfo**. Each of these tables only contains the most recent measurements and information.

- The **Public** table contains the measurements as they are made. It is updated at the scan interval set within the datalogger program.
- The **Status** table includes information on the health of the datalogger and is updated only when viewed.
- The **DataTableInfo** table reports statistics related to data tables. It also only updates when viewed.
- User-defined data tables update at the schedule set within the program.

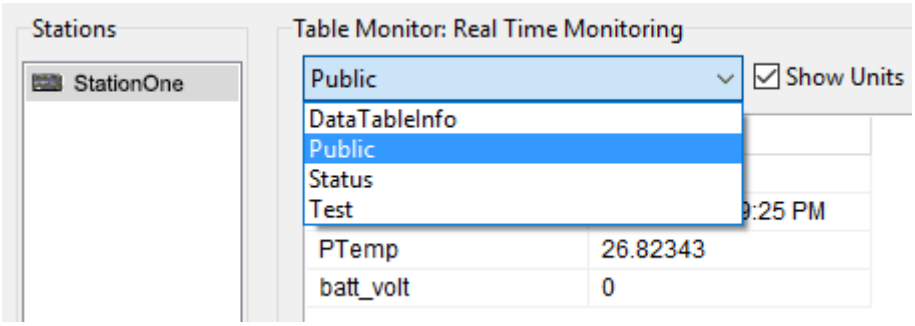
Monitoring Data

Follow a tutorial or use the following instructions:

PC200W and PC400 users, click the **Connect** button ( ), then click the **Monitor Data** tab. When this tab is first opened for a datalogger, values from the **Public** table are displayed. To view data from other tables, click the **Add** button ( ), select a table or field from the list, then drag it into a cell on the **Monitor Data** tab.



LoggerNet users, select **Main** and click the **Connect** button ( ) on the LoggerNet toolbar, select the datalogger from the **Stations** list, then click the **Connect** button ( ). Once connected, you can select a table to view using the **Table Monitor** list.



Collecting Data

The datalogger writes to data tables according to the schedule set within the CRBasic program (see "Creating Data Tables in a Program" on page 35 for more information). After the program has been running for enough time to generate data records, data may be collected and reviewed via your datalogger support software. Collections may be done manually, or automatically through scheduled collections set in LoggerNet **Setup**. Follow a tutorial or use the following instructions.

COLLECTING DATA USING LOGGERNET

1. LoggerNet users, select **Main** and click the **Connect** button () on the LoggerNet toolbar, select the datalogger from the **Stations** list, then click the **Connect** button ().
2. Click the **Collect Now** button ().
3. After the data is collected, the **Data Collection Results** window displays the tables collected and where they are stored on the computer.
4. Click **View File** to view the data.

COLLECTING DATA USING PC200W OR PC400W

1. PC200W and PC400 users, click the **Connect** button ().
2. Click the **Collect Data** tab.
3. Select an option for **What to Collect**. Either option creates a new file if one does not already exist.
   - **New data from datalogger (Append to data files)**: Collects only the data in the selected tables stored since the last data collection and appends this data to the end of the existing table files on the computer.
   - **All data from datalogger (Overwrite data files)**: Collects all of the data in the selected tables and replaces the existing table files on the computer.
4. Select the tables to collect from the list at the bottom of the window.
5. Click **Start Data Collection**.
6. The **Data Collection Results** window displays the tables collected and where they are stored on the computer.
7. Click **View File** to view the data.

Viewing Historic Data

View historic data in a spreadsheet format using View Pro. View Pro also contains tools for visualizing data in several graphical layouts. Follow a tutorial or use the following instructions:

Once the datalogger has had ample time to take multiple measurements, you can collect and review the data.

1. To view the most recent data, connect the datalogger to your computer and collect your data (see "Collecting Data" on page 32 for more information).

2. Open View Pro:
    - LoggerNet users select **Data** and click **View Pro** () on the LoggerNet toolbar.
    - PC200W and PC400 users click the **View Data Files via View Pro** toolbar button ().
3. Click the **Open** toolbar button (), navigate to the directory where you saved your tables (the default directory is **C:\Campbellsci**`[your datalogger software application]`).

About Data Tables

A data table is essentially a file that resides in datalogger memory (for information on data table storage, see "Memory and Data Storage" on page 37). The file consists of five or more rows. Each row consists of columns, or fields. The first four rows constitute the file header. Subsequent rows contain data records. Data tables may store individual measurements, individual calculated values, or summary data such as averages, maxima, or minima.

The file is written to each time data are directed to that file. The datalogger records data based on time or event. You can retrieve data based on a schedule or by manually choosing to collect data using datalogger support software (see "Collecting Data" on page 32). The number of data tables is limited to 20.

Example of a typical data table

TOA5, MyStation, CR300, 142, CRxxx.Std.01, CPU:MyTemperature.CR300, 1958, OneMin				
TIMESTAMP	RECORD	BattV_Avg	PTemp_C_Avg	Temp_C_Avg
TS	RN	Volts	Deg C	Deg C
		Avg	Avg	Avg
2016-03-08 14:24:00	0	13.68	21.84	20.71
2016-03-08 14:25:00	1	13.65	21.84	20.63
2016-03-08 14:26:00	2	13.66	21.84	20.63
2016-03-08 14:27:00	3	13.58	21.85	20.62
2016-03-08 14:28:00	4	13.64	21.85	20.52
2016-03-08 14:29:00	5	13.65	21.85	20.64

TABLE DEFINITIONS

Each data table is associated with overhead information, referred to as “table definitions,” that becomes part of the file header (first few lines of the file) when data are downloaded to a computer. These include the table format, datalogger type and OS, name of the CRBasic program running in the datalogger, name of the data table (limited to 20 characters), and the alphanumeric field names to attach at the head of data columns,

First Header Row

The first header row of the data table is the environment line, which consists of eight fields. The following list describes the fields using the previous table entries as an example:

- 1: **TOA5** - Table output format. Changed via LoggerNet **Setup** () **Standard View**, **Data Files** tab.
- 2: **MyStation** - Station name. Changed via LoggerNet **Setup**, Device Configuration Utility, or CRBasic program.
- 3: **CR**300 - Datalogger model.
- 4: **142** - Datalogger serial number.
- 5: **CRxxx.Std.01** - Datalogger OS version. Changed via installation of a new OS.
- 6: **CPU:MyTemperature.CR300** - Datalogger program name. Changed by sending a new program (see "Sending a Program to the Datalogger" on page 29 for more information).
- 7: **1958** - Datalogger program signature. Changed by revising a program or sending a new program (see "Sending a Program to the Datalogger" on page 29 for more information).
- 8: **OneMin** - Table name. Changed by revising a program (see "Creating Data Tables in a Program" on page 35 for more information).

Second Header Row

The second header row reports field names. Default field names are a combination of the variable names (or aliases) from which data are derived, and a three-letter suffix. The suffix is an abbreviation of the data process that outputs the data to storage. A list of these abbreviations follows in "Data Process Names and Abbreviations" on page 34.

If a field is an element of an array, the field name will be followed by a list of subscripts within parentheses that identifies the array index. For example, a variable named ***Values***, which is declared as a two-by-two array in the datalogger program, will be represented by four field names: ***Values(1,1)***, ***Values(1,2)***, ***Values(2,1)***, and ***Values(2,2)***. Scalar variables will not have array subscripts. There will be one value on this line for each scalar value defined by the table.

If the default field names are not acceptable to the programmer, the **`FieldNames()`** instruction can be used in the CRBasic program to customize the names. `TIMESTAMP`, `RECORD`, `BattV_Avg`, `PTemp_C_Avg`, and `Temp_C_Avg` are the default field names in the previous "Example of a typical data table" on page 33.

Third Header Row

The third header row identifies engineering units for that field of data. These units are declared at the beginning of a CRBasic program, as shown in "Creating Data Tables in a Program" on page . In Short Cut, units are chosen when sensors or measurements are added. Units are strictly for documentation. The datalogger does not make use of declared units, nor does it check their accuracy.

Fourth Header Row

The fourth header row reports abbreviations of the data process used to produce the field of data.

Data Process Names and Abbreviations

- Totalize **`Tot`**
- Average **`Avg`**

- Maximum **Max**
- Minimum **Min**
- Sample at Max or Min **SMM**
- Standard Deviation **Std**
- Moment **MMT**
- Sample **No abbreviation**
- Histogram[1] **Hst**
- Histogram4D **H4D**
- FFT **FFT**
- Covariance **Cov**
- Level Crossing **LCr**
- WindVector **WVc**
- Median **Med**
- ET **ETsz**
- Solar Radiation (from ET) **RSo**
- Time of Max **TMx**
- Time of Min **TMn**

[1]Hst is reported in the form **Hst,20,1.0000e+00,0.0000e+00,1.0000e+01** where **Hst** denotes a histogram, **20** = 20 bins, **1** = weighting factor, **0** = lower bound, **10** = upper bound.

Data Records

Subsequent rows are called data records. They include observed data and associated record keeping. The first field is a timestamp (**TS**), and the second field is the record number (**RN**).

The timestamp shown represents the time at which the data is written. Therefore, because the scan rate of the program `MyTemperature.CR300` is 1 second, `Temp_C_Avg` in record number 3 in the previous "Example of a typical data table" on page 33, shows the average of the measurements taken over the minute beginning at 14:26:01 and ending at 14:27:00. As another example, consider rainfall measured every second with a daily total rainfall recorded in a data table written at midnight. The record timestamped 2016-03-08 00:00:00 will contain the total rainfall beginning at 2016-03-07 00:00:01 and ending at 2016-03-08 00:00:00.

Creating Data Tables in a Program

Data are stored in tables as directed by the CRBasic program. In Short Cut, data tables are created in the **Output** steps (see "Creating a Program in Short Cut" on page 28). Data tables are created within the CRBasic datalogger program using the **DataTable()/EndTable** instructions. They are placed after variable declarations and before the **BeginProg** instruction. Between **DataTable()** and **EndTable()** are instructions that define what data to store and under what conditions data are stored. A data table

must be called by the CRBasic program for data storage processing to occur. Typically, data tables are called by the **CallTable** instruction once each Scan. These instructions include:

```
DataTable()
  'Output Trigger Condition(s)
  'Output Processing Instructions
EndTable
```

Use the **DataTable()** instruction to define the table size for your conditional data tables. You can set a specific number of records or allow your datalogger to auto-allocate table size. With auto-allocation, the datalogger balances the memory so the tables “fill up” (newest data starts to overwrite the oldest data) at about the same time. It is recommended you reserve the use of auto-allocation for data tables that store data based only on time (tables that store data based on the **DataInterval** instruction). Event or conditional tables are usually set to a fixed number of records. View data table fill times for your program on the **Station Status | Table Fill Times** tab (see "Checking Station Status" on page 68 for more information). For information on data table storage, see "Memory and Data Storage" on page 37.

For more information, see the CRBasic help.

Memory and Data Storage

The datalogger includes three types of memory: RAM, Flash, and Serial Flash. Memory storage is outlined in the following sections. For additional information on datalogger memory, visit the Campbell Scientific blog article, "How to Know when Your Datalogger Memory is Getting Full."

Flash Memory

The datalogger operating system is stored in a separate section of flash memory. To update the operating system, see "Updating the Operating System" on page 64.

Serial Flash Memory

Serial flash memory holds the CPU drive, the web page, and datalogger settings. Because flash memory has a limited number of erase/write cycles, care must be taken to avoid continuously writing to files on the CPU drive. For this reason, extended data storage is allocated in serial flash only when data tables are programmed to auto-allocate table size.

DATA STORAGE

Measurement data is primarily stored in data tables. Data are usually erased from this area when a program is sent to the datalogger. Final storage memory for the CR300 series are organized in 4 KB sectors of serial flash. Each sector is rated for 100,000 serial flash erases.

During data table initialization, memory sectors are assigned to each data table according to the parameters set in the program. Program options that affect the allocation of memory include the ***Size*** parameter of the **DataTable()** instruction, the ***Interval*** and ***Units*** parameters of the **DataInterval()** instruction, and the ***Interval*** and ***Units*** parameters of the **Scan()** instruction. The datalogger uses those parameters to assign sectors in a way that maximizes the life of its memory.

By default, data storage memory sectors are organized as ring memory. When the ring is full, oldest data are overwritten by newest data. Using the ***FillStop*** statement sets a program to stop writing to the data table when it is full, and no more data are stored until the table is reset. To see the total number of records that can be stored before the oldest data are overwritten, or to reset tables, go to **Station Status | Table Fill Times** in your datalogger support software.

Data concerning the datalogger memory are posted in the **Status** and **DataTableInfo** tables.

CPU DRIVE

The serial flash memory CPU drive contains datalogger programs and other files. This memory is managed in File Control.

Caution: When writing to files under program control, take care to write infrequently to prevent premature failure of serial flash memory. Internal chip manufacturers specify the flash technology used in Campbell Scientific CPU: drives at about 100,000 write/erase cycles. While Campbell Scientific's in-house testing has found the manufacturers' specifications to be very conservative, it is prudent to note the risk associated with repeated file writes via program control. See also SerialFlashErrors.SerialFlashErrors.

Also, see "System Specifications" on page 86 for information on datalogger memory.

Measurements

Voltage Measurements

Voltage measurements are made using an ADC. A high-impedance programmable-gain amplifier amplifies the signal. Internal multiplexers route individual terminals within the amplifier. The CRBasic measurement instruction controls the ADC gain and configuration – either single-ended or differential input. Information on the differences between single-ended and differential measurements can be found here: "Deciding Between Single-Ended or Differential Measurements" on page 76.

A voltage measurement proceeds as follows:

1. Set PGIA gain for the voltage range selected with the CRBasic measurement instruction parameter ***Range***. Set the ADC for the first notch frequency selected with ***fN1***.
2. If used, turn on excitation to the level selected with ***ExmV***.
3. Multiplex selected terminals (***SEChan*** or ***DiffChan***).
4. Delay for the entered settling time (***SettlingTime***).
5. Perform the analog-to-digital conversion.
6. Repeat for input reversal as determined by parameter ***RevDiff***.
7. Apply multiplier (***Mult***) and offset (***Offset***) to measured result.

Conceptually, analog voltage sensors output two signals: high and low. For example, a sensor that outputs 1000 mV on the high lead and 0 mV on the low has an overall output of 1000 mV. A sensor that outputs 2000 mV on the high lead and 1000 mV on the low also has an overall output of 1000 mV. Sometimes, the low signal is simply sensor ground (0 mV). A single-ended measurement measures the high signal with reference to ground, with the low signal tied to ground. A differential measurement measures the high signal with reference to the low signal. Each configuration has a purpose, but the differential configuration is usually preferred.

In general, use the smallest input range that accommodates the full-scale output of the sensor. This results in the best measurement accuracy and resolution (see "Power Output Specifications" on page 89 for more information).

A set overhead reduces the chance of overrange. Overrange limits are available in the specifications. The datalogger indicates a measurement overrange by returning a **NAN** for the measurement.

Warning: Sustained voltages in excess of ±-6 V/+9 V (SE1, SE2), ±17 V (SE3 to SE6) applied to terminals configured for analog input will damage CR300 series circuitry.

SINGLE-ENDED MEASUREMENTS

A single-ended measurement measures the difference in voltage between the terminal configured for single-ended input and the reference ground. For example, single-ended channel 1 is comprised of terminals **SE 1** and ⏚. Single-ended terminals are labeled in blue. For more information, see "Wiring Panel and Terminal Functions" on page 4. The single-ended configuration is used with the following CRBasic instructions:

- **VoltSE()**
- **BrHalf()**
- **BrHalf3W()**
- **TCSE()**
- **Therm107()**
- **Therm108()**
- **Therm109()**

DIFFERENTIAL MEASUREMENTS

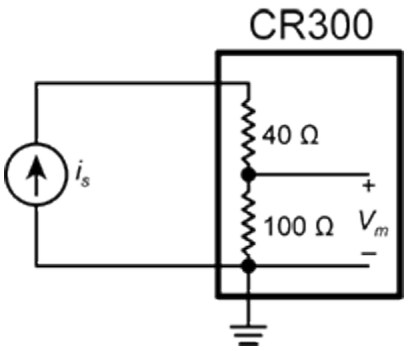
A differential measurement measures the difference in voltage between two input terminals. For example, **DIFF** channel **1** is comprised of terminals **1H** and **1L**, with **1H** as high and **1L** as low. For more information, see "Wiring Panel and Terminal Functions" on page 4. The differential configuration is used with the following CRBasic instructions:

- **VoltDiff()**
- **BrFull()**
- **BrFull6W()**
- **BrHalf4W()**
- **TCDiff()**

For more information, see "Analog Specifications" on page 89 and "Power Output Specifications" on page 89.

Current-Loop Measurements

Terminals **SE1** and **SE2** can be configured to make analog current measurements using the **CurrentSE()** instruction. Current is measured across the 100 Ω resistor with 140 Ω total resistance to ground. The following image shows a simplified schematic of a current measurement.



Use a CURS100 terminal input module when an application needs more than 2 current inputs or measurements. For detailed instructions, see http://www.campbellsci.com/curs100.

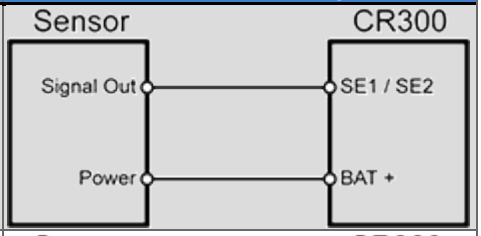
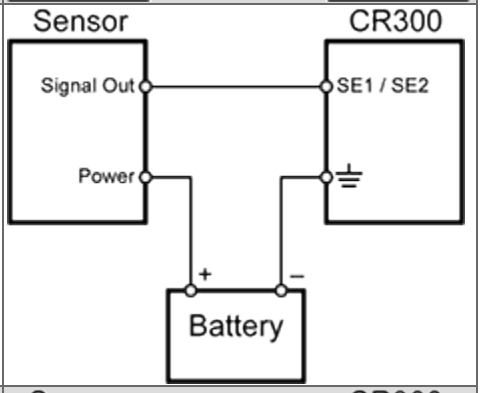
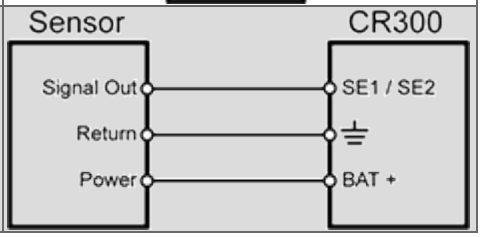
VOLTAGE RANGES FOR CURRENT MEASUREMENTS

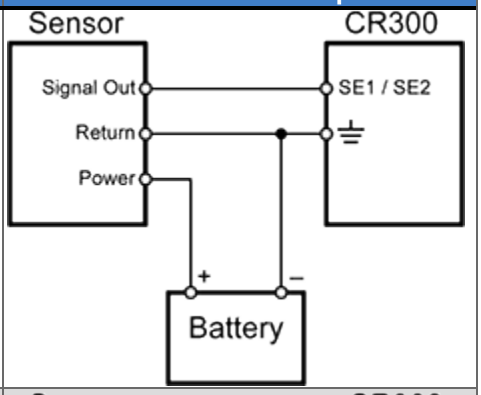
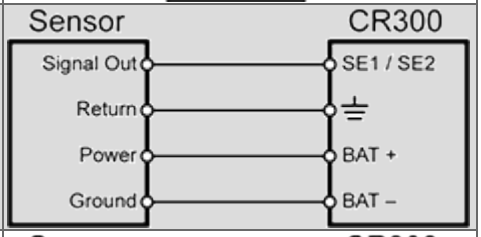
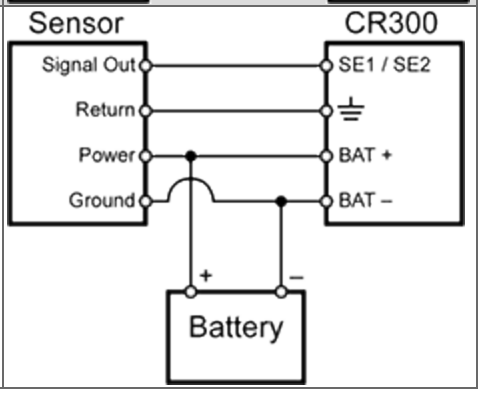
The datalogger measures the current through the use of a 100 Ω resistor. Thus, like a single-ended voltage instruction, it requires a voltage range option. In general, use the smallest fixed-input range that accommodates the full-scale output of the transmitter. This results in the best measurement accuracy and resolution.

To select the appropriate voltage range, the expected current output range must be known. Using Ohm's Law, multiply the maximum expected current by 100 Ω to find the maximum voltage to be measured. Because the maximum voltage input is 2500 mV, the maximum current input must be 25 mA or less.

EXAMPLE CURRENT-LOOP MEASUREMENT CONNECTIONS

The following table shows example schematics for connecting typical current sensors and devices.

Sensor Type	Connection Example
2-wire transmitter using datalogger power	
2-wire transmitter using external power	
3-wire transmitter using datalogger power	

Sensor Type	Connection Example
3-wire transmitter using external power	
4-wire transmitter using datalogger power	
4-wire transmitter using external power	

See also "Current-Loop Measurements Specifications" on page 91.

Resistance Measurements

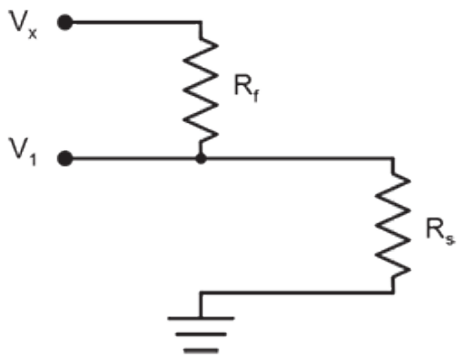
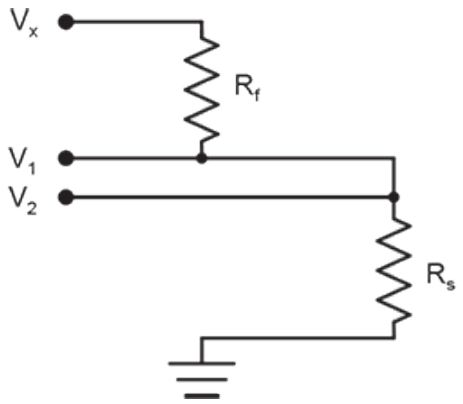
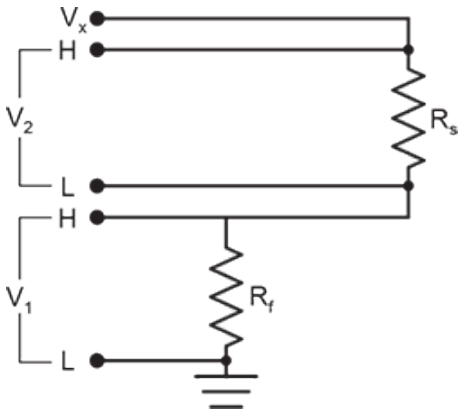
Bridge resistance is determined by measuring the difference between a known voltage applied to the excitation (input) of a resistor bridge and the voltage measured on the output arm. The datalogger supplies a precise voltage excitation via **VX** terminals. Return voltage is measured on analog input terminals configured for single-ended (**SE**) or differential (**DIFF**) input. The result of the measurement is a ratio of measured voltages.

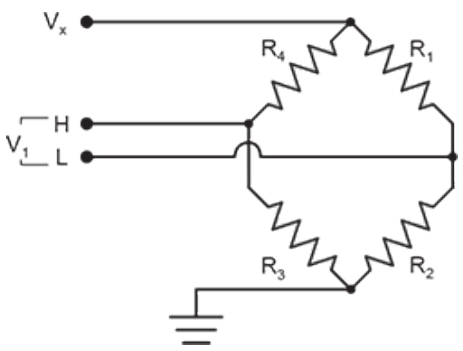
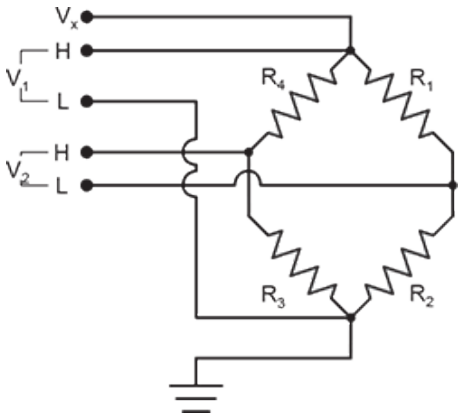
See also "Resistance Measurements Specifications" on page 91.

RESISTANCE MEASUREMENTS WITH VOLTAGE EXCITATION

CRBasic instructions for measuring resistance with voltage excitation include:

- **`BrHalf()`** - half bridge
- **`BrHalf3W()`** - three-wire half bridge
- **`BrHalf4W()`** - four-wire half bridge
- **`BrFull()`** - four-wire full bridge
- **`BrFull6W()`** - six-wire full bridge

Resistive-Bridge Type and Circuit Diagram	CRBasic Instruction and Fundamental Relationship	Relational Formulas
Half Bridge[1]  	CRBasic Instruction: **`BrHalf()`** Fundamental Relationship: X = result w/mult = 1, offset = 0 $X = 1000 \frac{V_2}{V_1} = 1000 \left(\frac{R_3}{R_3+R_4} - \frac{R_2}{R_1+R_2}\right)$	$R_s = R_f \frac{X}{1-X}$ $R_f = \frac{R_s(1-X)}{X}$
Three Wire Half Bridge[1,2]  	CRBasic Instruction: **`BrHalf3W()`** Fundamental Relationship: X = result w/mult = 1, offset = 0 $X = \frac{2V_2 - V_1}{V_x - V_1} = \frac{R_s}{R_f}$	$R_s = R_f X$ $R_f = \frac{R_s}{X}$
Four Wire Half Bridge[1,2]  	CRBasic Instruction: **`BrHalf4W()`** Fundamental Relationship: X = result w/mult = 1, offset = 0 $X = \frac{V_2}{V_1} = \frac{R_s}{R_f}$	$R_f = \frac{R_s}{X}$ $R_s = R_f X$

<table>
<tr><th>Resistive-Bridge Type and Circuit Diagram</th><th>CRBasic Instruction and Fundamental Relationship</th><th>Relational Formulas</th></tr>
<tr><td>Full Bridge[1,2]
 
</td><td>CRBasic Instruction: BrFull()
Fundamental Relationship:
X = result w/mult = 1, offset = 0
$X = 1000 \frac{V_1}{V_x} = 1000 \left(\frac{R_3}{R_3+R_4} - \frac{R_2}{R_1+R_2}\right)$</td><td rowspan="2">These relationships apply to BrFull() and BrFull6W()
$R_1 = \frac{R_2(1-X_1)}{X_1}$
$R_2 = \frac{R_1 X_1}{1-X_1}$
where $X_1 = \frac{-X}{1000} + \frac{R_3}{R_3+R_4}$
$R_3 = \frac{R_4 X_2}{1-X_2}$
$R_4 = \frac{R_3(1-X_2)}{X_2}$
where $X_2 = \frac{-X}{1000} + \frac{R_2}{R_1+R_2}$</td></tr>
<tr><td>Six Wire Full Bridge[1]
 
</td><td>CRBasic Instruction: BrFull6W()
Fundamental Relationship:
X = result w/mult = 1, offset = 0
$X = 1000 \frac{V_2}{V_1} = 1000 \left(\frac{R_3}{R_3+R_4} - \frac{R_2}{R_1+R_2}\right)$</td></tr>
</table>

[1]Key: V_x = excitation voltage; V_1, V_2 = sensor return voltages; R_f = fixed, bridge or completion resistor; R_s = variable or sensing resistor.

[2]Campbell Scientific offers a list of available terminal input modules to facilitate this measurement.

Offset voltage compensation applies to bridge measurements. ***RevDiff*** and ***MeasOff*** parameters are discussed in "Minimizing Offset Voltages" on page 83. Much of the offset error inherent in bridge measurements is canceled out by setting ***RevDiff*** and ***MeasOff*** to **True**.

CRBasic Example: Four-Wire Full Bridge Measurement and Processing

```
'This program example demonstrates the measurement and
'processing of a four-wire resistive full bridge.
'In this example, the default measurement stored
'in variable X is deconstructed to determine the
'resistance of the R1 resistor, which is the variable
'resistor in most sensors that have a four-wire
'full-bridge as the active element.

'Declare Variables

Public X
Public X_1
Public R_1
Public R_2 = 1000 'Resistance of fixed resistor R2
Public R_3 = 1000 'Resistance of fixed resistor R2
Public R_4 = 1000 'Resistance of fixed resistor R4
```

```
'Main Program
BeginProg
  Scan(500,mSec,1,0)

    'Full Bridge Measurement:
    BrFull(X,1,mV2500,1,Vx1,1,2500,False,True,0,60,1.0,0)
    X_1 = ((-1 * X) / 1000) + (R_3 / (R_3 + R_4))
    R_1 = (R_2 * (1 - X_1)) / X_1

  NextScan

EndProg
```

STRAIN MEASUREMENTS

A principal use of the four-wire full bridge is the measurement of strain gages in structural stress analysis. **`StrainCalc()`** calculates microstrain (με) from the formula for the particular strain bridge configuration used. All strain gages supported by **`StrainCalc()`** use the full-bridge schematic. In strain-gage parlance, 'quarter-bridge', 'half-bridge' and 'full-bridge' refer to the number of active elements in the full-bridge schematic. In other words, a quarter-bridge strain gage has one active element, a half-bridge has two, and a full-bridge has four.

`StrainCalc()` requires a bridge-configuration code. The following table shows the equation used by each configuration code. Each code can be preceded by a dash (-). Use a code without the dash when the bridge is configured so the output decreases with increasing strain. Use a dashed code when the bridge is configured so the output increases with increasing strain. A dashed code sets the polarity of V_r to negative.

StrainCalc() BrConfig Code	Configuration
1	Quarter-bridge strain gage[1]: $$\mu\varepsilon = \frac{-4 * 10^6 V_r}{GF(1 + 2V_r)}$$
2	Half-bridge strain gage[1]. One gage parallel to strain, the other at 90° to strain: $$\mu\varepsilon = \frac{-4 * 10^6 V_r}{GF[(1 + \nu) - 2V_r(\nu - 1)]}$$
3	Half-bridge strain gage. One gage parallel to +ε, the other parallel to -ε[1]: $$\mu\varepsilon = \frac{-2 * 10^6 V_r}{GF}$$

StrainCalc() BrConfig Code	Configuration
4	Full-bridge strain gage. Two gages parallel to +ε, the other two parallel to -ε[1]: $$\mu\varepsilon = \frac{-10^6 V_r}{GF}$$
5	Full-bridge strain gage. Half the bridge has two gages parallel to +ε and -ε, and the other half to +νε and -νε 1: $$\mu\varepsilon = \frac{-2 * 10^6 V_r}{GF(\nu + 1)}$$
6	Full-bridge strain gage. Half the bridge has two gages parallel to +ε and -νε , and the other half to -νε and +ε[1]: $$\mu\varepsilon = \frac{-2 * 10^6 V_r}{GF[(\nu + 1) - V_r(\nu - 1)]}$$

[1]Where

- ν : Poisson's Ratio (0 if not applicable).
- GF: Gage Factor.
- V_r: 0.001 (Source-Zero) if **BRConfig** code is positive (+).
- V_r: -0.001 (Source-Zero) if **BRConfig** code is negative (-).

and where:

- "source": the result of the full-bridge measurement (X = 1000 • V1 / Vx) when multiplier = 1 and offset = 0.
- "zero": gage offset to establish an arbitrary zero.

ACCURACY FOR RESISTANCE MEASUREMENTS

Consult the following technical papers for in-depth treatments of several topics addressing voltage measurement quality:

- Preventing and Attacking Measurement Noise Problems
- Benefits of Input Reversal and Excitation Reversal for Voltage Measurements
- Voltage Measurement Accuracy, Self- Calibration, and Ratiometric Measurements

Note: Error discussed in this section and error-related specifications of the CR300 series do not include error introduced by the sensor or by the transmission of the sensor signal to the datalogger.

For accuracy specifications for ratiometric resistance measurements, see "Resistance Measurements Specifications" on page 91. Voltage measurement is variable V_1 or V_2 in "Resistance Measurements" on page 41. Offset is the same as that for simple analog voltage measurements.

Assumptions that support the ratiometric-accuracy specification include:

- Datalogger is within factory calibration specification.
- Effects due to the following are not included in the specification:
  - Bridge-resistor errors
  - Sensor noise
  - Measurement noise

Period-Averaging Measurements

Period-averaging measurements are used to measure the period or frequency of a signal. For these measurements, the datalogger uses a high-frequency digital clock to measure time differences between signal transitions, whereas pulse-count measurements simply accumulate the number of counts. As a result, period-average measurements offer much better frequency resolution per measurement interval than pulse-count measurements. See also "Pulse Measurements" on page 46.

SE 1-4 terminals on the datalogger are configurable for measuring the period of a signal.

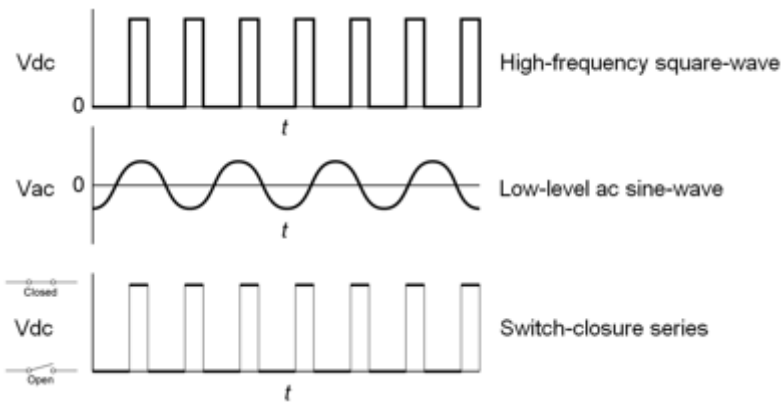
See also "Period Averaging Specifications" on page 91.

Tip: Both pulse count and period-average measurements are used to measure frequency output sensors. However, their measurement methods are different. Pulse count measurements use dedicated hardware - pulse count accumulators, which are always monitoring the input signal, even when the datalogger is between program scans. In contrast, period-average measurements use program instructions that only monitor the input signal during a program scan. Consequently, pulse count scans can occur less frequently than period-average scans. Pulse counters may be more susceptible to low-frequency noise because they are always "listening", whereas period-averaging measurements may filter the noise by reason of being "asleep" most of the time. Pulse count measurements are not appropriate for sensors that are powered off between scans, whereas period-average measurements work well since they can be placed in the scan to execute only when the sensor is powered and transmitting the signal.

Pulse Measurements

The output signal generated by a pulse sensor is a series of voltage waves. The sensor couples its output signal to the measured phenomenon by modulating wave frequency. The datalogger detects the state transition as each wave varies between voltage extremes (high-to-low or low-to-high). Measurements are processed and presented as counts, frequency, or timing data. Both pulse count and period-average measurements are used to measure frequency-output sensors. For more information, see "Period-Averaging Measurements" on page 46.

The datalogger includes terminals that are configurable for pulse input to measure counts or frequency as shown in the following image.



Pulse input terminals and the input types they can measure

Using the **PulseCount()** instruction, **P_LL**, **P_SW**, **SE 1-4**, and **C** terminals are configurable for pulse input to measure counts or frequency. Maximum input frequency is dependent on input voltage. If pulse input voltages exceed the maximum voltage, third-party external-signal conditioners should be employed.

Terminals configured for pulse input have internal filters that reduce electronic noise, and thus reduce false counts. Internal ac coupling is used to eliminate dc offset voltages. For tips on working with pulse measurements, see "Pulse Measurement Tips" on page 49.

See also "Pulse Counting Specifications" on page 92.

LOW-LEVEL AC MEASUREMENTS

Low-level ac (sine-wave) signals can be measured on **P_LL** terminals. Ac generator anemometers typically output low-level ac.

Measurements include the following:

- Counts
- Frequency (Hz)
- Running average

Rotating magnetic-pickup sensors commonly generate ac voltage ranging from thousandths of volts at low-rotational speeds to several volts at high-rotational speeds.

CRBasic instruction: **PulseCount()**

HIGH FREQUENCY MEASUREMENTS

High-frequency (square-wave) signals can be measured on **P_LL**, **P_SW**, **SE 1-4**, or **C** terminals. Common sensors that output high-frequency include:

- Photo-chopper anemometers
- Flow meters

Measurements include counts, frequency in hertz, and running average.

- CRBasic instructions: **PulseCount()**

SWITCH-CLOSURE AND OPEN-COLLECTOR MEASUREMENTS

Switch-closure and open-collector signals can be measured on **P_SW** or **C** terminals. Mechanical switch-closures have a tendency to bounce before solidly closing. Unless filtered, bounces can cause multiple counts per event. The datalogger automatically filters bounce. Because of the filtering, the maximum switch-closure frequency is less than the maximum high-frequency measurement frequency. Sensors that commonly output a switch-closure or open-collector signal include:

- Tipping-bucket rain gages
- Switch-closure anemometers
- Flow meters

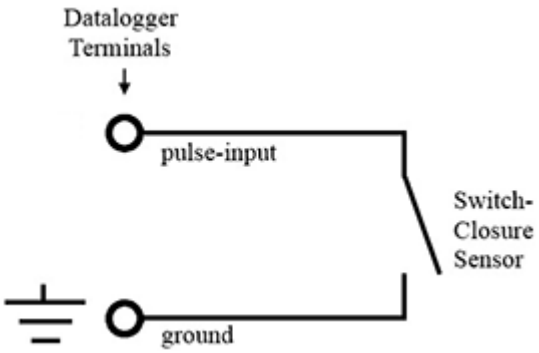
Data output options include counts, frequency (Hz), and running average.

P_SW Terminal

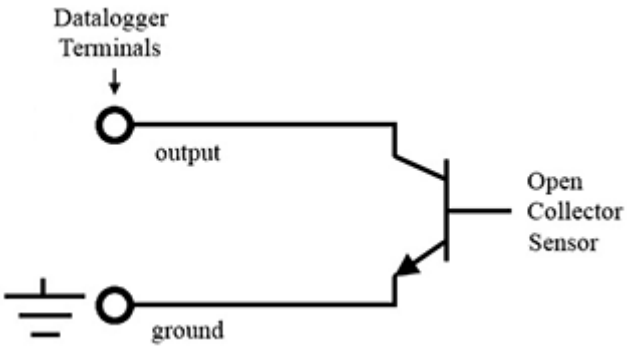
An internal 100 kΩ pull-up resistor pulls an input to 3.3 Vdc with the switch open, whereas a switch-closure to ground pulls the input to 0 V.

- CRBasic instruction: **PulseCount()**

Switch Closure on **P** Terminal



Open Collector on **P** Terminal



C Terminals

Switch-closure measurements on **C** terminals require a 100 kΩ pull-up resistor to 12 V. Switch-closure mode is a special case edge-count function that measures dry-contact switch-closures or open collectors. The operating system filters bounces.

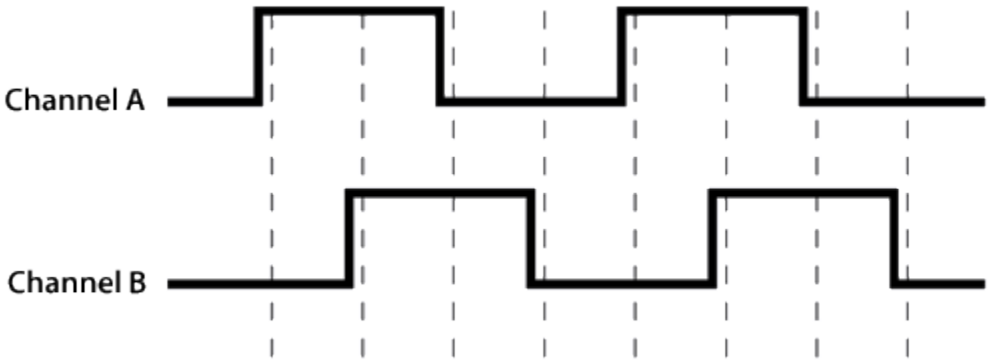
- CRBasic instruction: **PulseCount()**

See also "Power Output Specifications" on page 89.

QUADRATURE MEASUREMENTS

The **Quadrature()** instruction is used to measure shaft or rotary encoders. A shaft encoder is an electro-mechanical device that outputs a signal to represent the annular position or motion of the shaft. Each encoder will have two output signals an A line and a B line. As the shaft rotates the A and B lines will generate digital pulses that can be read/measured by the datalogger.

In the following example, channel A leads channel B, therefore the encoder is determined to be moving in a clockwise direction. If channel B led channel A, it would be determined that the encoder was moving in a counterclockwise direction.



Terminals SE1 and SE2 or C1 and C2 can be configured as digital port pairs to monitor the two sensing channels of an encoder. The `Quadrature()` instruction can return:

- The accumulated number of counts from channel A and channel B. Count will increase if channel A leads channel B. Count will decrease if channel B leads channel A.
- The net direction.
- Number of counts in the A leading B direction.
- Number of counts in the B leading A direction.

Counting modes:

- Counting the increase on rising edge of channel A when channel A leads channel B. Counting the decrease on falling edge of channel A when channel B leads channel A.
- Counting the increase at each rising and falling edge of channel A when channel A leads channel B. Counting the decrease at each rising and falling edge of channel A when channel A leads channel B.
- Counting the increase at each rising and falling edge of both channels when channel A leads channel B. Counting the decrease at each rising and falling edge of both channels when channel B leads channel A.

For more information, see "Pulse Counting Specifications" on page 92.

PULSE MEASUREMENT TIPS

The **`PulseCount()`** instruction uses dedicated 32-bit counters to accumulate all counts over the programmed scan interval. The resolution of pulse counters is one count or 1 Hz. Counters are read at the beginning of each scan and then cleared. Counters will overflow if accumulated counts exceed 4,294,967,296 (2^{32}), resulting in erroneous measurements.

Counts are the preferred `PulseCount()` output option when measuring the number of tips from a tipping-bucket rain gage or the number of times a door opens. Many pulse-output sensors, such as anemometers and flow meters, are calibrated in terms of frequency (Hz) so are usually measured using the `PulseCount()` frequency-output option.

Understanding the signal to be measured and compatible input terminals and CRBasic instructions is helpful. See "Pulse input terminals and the input types they can measure" on page 47.

Input Filters and Signal Attenuation

Terminals configured for pulse input have internal filters that reduce electronic noise. The electronic noise can result in false counts. However, input filters attenuate (reduce) the amplitude (voltage) of the signal. Attenuation is a function of the frequency of the signal. Higher-frequency signals are attenuated more. If a signal is attenuated enough, it may not pass the detection thresholds required by the pulse count circuitry.See "Pulse Counting Specifications" on page 92 for more information. The listed pulse measurement specifications account for attenuation due to input filtering.

Pulse Count Resolution

Longer scan intervals result in better resolution. `PulseCount()` resolution is 1 pulse per scan. On a 1 second scan the resolution is 1 pulse per second. The resolution on a 10 second scan interval is 1 pulse per 10 seconds, which is 0.1 pulses per second. The resolution on a 100 millisecond interval is 10 pulses per second.

For example, if a flow sensor outputs 4.5 pulses per second and you use a 1 second scan, one scan will have 4 pulses and the next 5 pulses. Scan to scan, the flow number will bounce back and forth. If you did a 10 second scan (or saved an average to a 10 second table), you would get 45 pulses. The average is 45 pulses for every 10 seconds. The average will correctly show 4.5 pulses per second. You wouldn't see the reading bounce on the longer time interval.

Vibrating Wire Measurements

The datalogger can measure vibrating wire sensors through vibrating-wire interface modules. Vibrating wire sensors are the sensor of choice in many environmental and industrial applications that need sensor stability over very long periods, such as years or even decades. A thermistor included in most sensors can be measured to compensate for temperature errors.

VSPECT

Measuring the resonant frequency by means of period averaging is the classic technique, but Campbell Scientific has developed static and dynamic spectral-analysis techniques (VSPECT) that produce superior noise rejection, higher resolution, diagnostic data, and, in the case of dynamic VSPECT, measurements up to 333.3 Hz. For detailed information on VSPECT, see Vibrating Wire Spectral Analysis Technology.

Communication Protocols

Dataloggers communicate with datalogger support software and other Campbell Scientific dataloggers using a number of protocols including PakBus, Modbus, DNP3, and TCP/IP. Several industry-specific protocols are also supported. See also "Communications Specifications" on page 94.

Some communication services, such as satellite networks, can be expensive to send and receive information. Best practices for reducing expense include:

- Declare **Public** only those variables that need to be public. Other variables should be declared as **Dim**.
- Be conservative with use of string variables and string variable sizes. Make string variables as big as they need to be and no more. The default size, if not specified, is 24 bytes, but the minimum is 4 bytes. Specify the size if 24 bytes of possibly unused space is not desired. Declare string variables **Public** and sample string variables into data tables only as needed.
- When using **`GetVariables()`** / **`SendVariables()`** to send values between dataloggers, put the data in an array and use one command to get the multiple values. Using one command to get 10 values from an array and swath of 10 is much more efficient (requires only 1 transaction) than using 10 commands to get 10 single values (requires 10 transactions).
- Set the datalogger to be a PakBus router only as needed. When the datalogger is a router, and it connects to another router like LoggerNet, it exchanges routing information with that router and, possibly (depending on your settings), with other routers in the network. Network Planner set this appropriately when it is used. This is also set through the **IsRouter** setting in the Settings Editor.
- Set PakBus beacons and verify intervals properly. For example, there is no need to verify routes every five minutes if communications are expected only every 6 hours. Network Planner will set this appropriately when it is used. This is also set through the **Beacon** and **Verify** settings in the Settings Editor.

For information on Designing a PakBus network using the Network Planner tool in LoggerNet, watch a video.

General Serial Communications

The datalogger supports two-way serial communication. These communication ports can be used with smart sensors that deliver measurement data through serial data protocols or with devices, such as modems, that communicate using serial data protocols. See "Communication Ports" on page 8 for information on port configuration options.

CRBasic instructions for general serial communications include:

- **`SerialOpen()`**
- **`SerialClose()`**

- **SerialIn()**
- **SerialInRecord()**
- **SerialInBlock()**
- **SerialOut()**
- **SerialOutBlock()**

To communicate over a serial port, it is important to be familiar with the protocol of the sensor or device. Refer to the manual of the sensor or device to find its protocol and then select the appropriate options for each CRBasic parameter. See the application note Interfacing Serial Sensors with Campbell Scientific Dataloggers for more programming details and examples.

NOTES ON COM1

- Though **Com1** uses RS-232 logic levels, it is limited to outputting 0 V (logic high) to 5 V (logic low). This may make **Com1** incompatible with some serial devices.
- **Com1** is not capable of TTL logic levels and so is not compatible with TTL-to-RS-232 converters for the purpose of presenting a true RS-232 interface.
- **Com1** also has a low input resistance that may make it incompatible with some serial devices with the addition of in-line resistance.

Modbus Communications

The datalogger supports Modbus RTU, Modbus ASCII, and Modbus TCP protocols and can be programmed as a Modbus master or Modbus slave. These protocols are often used in Modbus SCADA networks. Dataloggers can communicate with Modbus on all available communication ports. The datalogger communicates with Modbus over RS-232 using a RS-232-to RS-485 adapter and over TCP using an Ethernet or Wireless connection.

CRBasic Modbus instructions include (see CRBasic Editor help for the most recent information on each of these instructions and for program examples):

- **ModbusMaster()**
- **ModbusSlave()**
- **MoveBytes()**

For additional information on Modbus, see:

- Why Modbus Matters: An Introduction
- How to Access Live Measurement Data Using Modbus
- Using Campbell Scientific Dataloggers as Modbus Slave Devices in a SCADA Network

Because Modbus has a set command structure, programming the datalogger to get data from field instruments is much simpler than from serial sensors. Because Modbus uses a common bus and addresses each node, field instruments are effectively multiplexed to a datalogger without additional hardware.

A datalogger goes into sleep mode after 40 seconds of communication inactivity. Once asleep, two packets are required before it will respond. The first packet awakens the datalogger; the second packet is received as data. This would make a Modbus master fail to poll the datalogger, if not using retries.

The datalogger, through Device Configuration Utility or the **Status** table, can be set to keep communication ports open and awake, but at higher power usage.

Further information on Modbus can be found at:

- www.simplyModbus.ca/FAQ.htm
- www.Modbus.org/tech.php
- www.lammertbies.nl/comm/info/modbus.html

Internet Communications

The internet protocols listed in "Communications Specifications" on page 94, are supported by the CR310 or when using a cell modem with the CR300 series. The most up-to-date information on implementing these protocols is contained in CRBasic Editor help.

CRBasic instructions for internet communications include:

- **EmailRelay()**
- **EmailSend()**
- **EmailRcv()**
- **FTPClient()**
- **HTTPGet()**
- **HTTPOut()**
- **HTTPPost()**
- **HTTPPut()**
- **IPInfo()**
- **PPPOpen()**
- **PPPClose()**
- **TCPOpen()**
- **TCPClose()**

DNP3 Communications

DNP3 is designed to optimize transmission of data and control commands from a master computer to one or more remote devices or outstations. The datalogger allows DNP3 communication on all available communication ports. CRBasic DNP3 instructions include:

- **DNP()**
- **DNPUpdate()**
- **DNPVariable()**

See the CRBasic help for detailed information and program examples.

For additional information on DNP3 see:

- DNP3 with Campbell Scientific Dataloggers
- Getting to Know DNP3
- How to Access Your Measurement Data Using DNP3

PakBus Communications

PakBus is a protocol similar in concept to IP (Internet Protocol). By using signatured data packets, PakBus increases the number of communication and networking options available to the datalogger.

The datalogger allows PakBus communication on all available communication ports. For additional information, see The Many Possibilities of PakBus Networking.

Advantages of PakBus are as follows:

- Simultaneous communication between the datalogger and other devices.
- Peer-to-peer communication - no computer required. Special CRBasic instructions simplify transferring data between dataloggers for distributed decision making or control.
- Data consolidation - other PakBus dataloggers can be used as "sensors" to consolidate all data into one datalogger.
- Routing - the datalogger can act as a router, passing on messages intended for another Campbell Scientific datalogger. PakBus supports automatic route detection and selection.
- Short distance networks - a datalogger can talk to another datalogger over distances up to 30 feet by connecting transmit, receive, and ground wires between the dataloggers.

In a PakBus network, each datalogger is set to a unique address. The default PakBus address in most devices is 1. To communicate with the datalogger, the datalogger support software must know the datalogger PakBus address. The PakBus address is changed using Device Configuration Utility, datalogger **Status** table, Settings Editor, or PakBus Graph software.

CRBasic PakBus instructions include:

- **GetDataRecord()**
- **GetVariables()**
- **SendData()**
- **SendGetVariables()**
- **SendVariables()**

SDI-12 Communications

SDI-12 is a 1200 baud protocol that supports many smart sensors. The datalogger supports SDI-12 communication through two modes – transparent mode and programmed mode (see "SDI-12 Port" on page 8 for wiring terminal information).

- Transparent mode facilitates sensor setup and troubleshooting. It allows commands to be manually issued and the full sensor response viewed. Transparent mode does not record data. See "SDI-12 Transparent Mode" on page 55 for more information.
- Programmed mode automates much of the SDI-12 protocol and provides for data recording. See "SDI-12 Programmed Mode/Recorder Mode" on page 56 for more information.

CRBasic SDI-12 instructions include:

- **SDI12Recorder()**
- **SDI12SensorSetup()**
- **SDI12SensorResponse()**
- **SDI12Watch()**

The CR300 series use SDI-12 version 1.3.

SDI-12 TRANSPARENT MODE

System operators can manually interrogate and enter settings in probes using transparent mode. Transparent mode is useful in troubleshooting SDI-12 systems because it allows direct communication with probes.

Transparent mode may need to wait for commands issued by the programmed mode to finish before sending responses. While in transparent mode, the datalogger programs may not execute. Datalogger security may need to be unlocked before transparent mode can be activated.

Transparent mode is entered while the computer is in communication with the datalogger through a terminal emulator program. It is easily accessed through a terminal emulator available with Device Configuration Utility and other datalogger support software. Keyboard displays cannot be used. For how-to instructions for communicating directly with an SDI-sensor using a terminal emulator, watch this video.

To enter the SDI-12 transparent mode, enter the datalogger support software terminal emulator:

1. Press **Enter** until the datalogger responds with the prompt `CR300 series>`.
2. Type `SDI12` at the prompt and press **Enter**.
3. In response, the query `Enter Cx Port` is presented with a list of available ports. Enter the port number assigned to the terminal to which the SDI-12 sensor is connected, and press **Enter**. For example, `1` is entered for terminal **C1**.
4. An `Entering SDI12 Terminal` response indicates that SDI-12 transparent mode is active and ready to transmit SDI-12 commands and display responses.

The terminal-mode utility allows monitoring of SDI-12 traffic by using the watch command (sniffer mode). Watch an instructional this video or use the following instructions.

1. Enter the terminal mode as described previously.
2. Press **Enter** until a `CR300 series>` prompt appears.
3. Type `W` and then press **Enter**.
4. In response, the query `Select:` is presented with a list of available ports. Enter the port number assigned to the terminal to which the SDI-12 sensor is connected, and press **Enter**.
5. In answer to `Enter timeout (secs):` type `100` and press **Enter**.
6. In response to the query `ASCII (Y)?`, type `Y` and press **Enter**.
7. SDI-12 communications are then opened for viewing.

SDI-12 Transparent Mode Commands

SDI-12 commands and responses are defined by the SDI-12 Support Group (www.sdi-12.org) and are available in the SDI-12 Specification. Sensor manufacturers determine which commands to support. Commands have three components:

- Sensor address (**a**): A single character and the first character of the command. Sensors are usually assigned a default address of zero by the manufacturer. The wildcard address (**?**) is used in the `Address Query` command. Some manufacturers may allow it to be used in other commands. SDI-12 sensors accept addresses 0 through 9, a through z, and A through Z.
- Command body (for example, **M1**): An upper case letter (the "command") followed by alphanumeric qualifiers.
- Command termination (**!**): An exclamation mark.

An active sensor responds to each command. Responses have several standard forms and terminate with `<CR><LF>` (carriage return-line feed).

SDI-12 PROGRAMMED MODE/RECORDER MODE

The datalogger can be programmed to act as an SDI-12 data recorder or as an SDI-12 sensor using a terminal configured for SDI-12. The **`SDI12Recorder()`** instruction automates sending commands and recording responses. With this instruction, the commands to poll sensors and retrieve data are done automatically with proper elapsed time between the two. The datalogger automatically issues retries. See CRBasic Editor help for more information on this instruction.

Commands entered into the **`SDIRecorder()`** instruction differ slightly in function from similar commands entered in transparent mode. In transparent mode, for example, the operator manually enters `aM!` and `aD0!` to initiate a measurement and get data, with the operator providing the proper time delay between the request for measurement and the request for data. In programmed mode, the datalogger provides command and timing services within a single line of code. For example, when the **`SDI12Recorder()`** instruction is programmed with the `M!` command (note that the SDI-12 address is a separate instruction parameter), the datalogger issues the `aM!` and `aD0!` commands with proper elapsed time between the two. The datalogger automatically issues retries and performs other services that make the SDI-12 measurement work as trouble free as possible.

For troubleshooting purposes, responses to SDI-12 commands can be captured in programmed mode by placing a variable declared `As String` in the variable parameter. Variables not declared `As String` will capture only numeric data.

PROGRAMMING THE DATALOGGER TO ACT AS AN SDI-12 SENSOR

The **`SDI12SensorSetup()`** / **`SDI12SensorResponse()`** instruction pair programs the datalogger to behave as an SDI-12 sensor. A common use of this feature is the transfer of data from the datalogger to other Campbell Scientific dataloggers over a single-wire interface (terminal configured for SDI-12 to terminal configured for SDI-12), or to transfer data to a third-party SDI-12 recorder.

Details of using the **`SDI12SensorSetup()`** / **`SDI12SensorResponse()`** instruction pair can be found in the CRBasic Editor help.

When programmed as an SDI-12 sensor, the datalogger will respond to SDI-12 commands `M`, `MC`, `C`, `CC`, `R`, `RC`, `V`, `?`, and `I`. The following rules apply:

A datalogger can be assigned only one SDI-12 address per SDI-12 port. For example, a datalogger will not respond to both `0M!` AND `1M!` on SDI-12 port **C1**. However, different SDI-12 ports can have unique SDI-12 addresses.

SDI-12 POWER CONSIDERATIONS

When a command is sent by the datalogger to an SDI-12 probe, all probes on the same SDI-12 port will wake up. However, only the probe addressed by the datalogger will respond. All other probes will remain active until the timeout period expires.

Example:

Probe: Water Content

Power Usage:

- Quiescent: 0.25 mA
- Active: 66 mA
- Measurement: 120 mA

Measurement time: 15 s

Timeout: 15 s

Probes 1, 2, 3, and 4 are connected to SDI-12 port **C1**.

The time line in the following table shows a 35-second power-usage profile example.

For most applications, total power usage of 318 mA for 15 seconds is not excessive, but if 16 probes were wired to the same SDI-12 port, the resulting power draw would be excessive. Spreading sensors over several SDI-12 terminals helps reduce power consumption.

Example Power Usage Profile for a Network of SDI-12 Probes:

Time into Measurement Processes	Command	All Probes Awake	Time Out Expires	Probe 1 (mA)	Probe 2 (mA)	Probe 3 (mA)	Probe 4 (mA)	Total (mA)
Sleep				0.25	0.25	0.25	0.25	1
1	1M!	Yes		120	66	66	66	318
2-14				120	66	66	66	318
15			Yes	120	66	66	66	318
16	1D0!	Yes		66	66	66	66	264
17-29				66	66	66	66	264
30			Yes	66	66	66	66	264
Sleep				0.25	0.25	0.25	0.25	1

Maintaining Your Datalogger

Protect the datalogger from humidity and moisture. When humidity levels reach the dewpoint, condensation occurs, and damage to datalogger electronics can result. Adequate desiccant should be placed in the instrumentation enclosure to provide protection and control humidity, and it should be changed periodically.

If sending the datalogger to Campbell Scientific for calibration or repair, consult first with Campbell Scientific. If the datalogger is malfunctioning, be prepared to perform some troubleshooting procedures (see "Tips and Troubleshooting" on page 67). Many problems can be resolved with a telephone conversation. If calibration or repair is needed, the procedure shown on: https://www.campbellsci.com/repair should be followed when sending the product.

Datalogger Calibration

Campbell Scientific recommends factory recalibration every three years. During calibration, all the input terminals, peripheral and communications terminals, operating system, and memory areas are checked; and the internal battery is replaced. The datalogger is checked to ensure that all hardware operates within published specifications before it is returned. To request recalibration for a product, see www.campbellsci.com/repair.

It is recommended that you maintain a level of calibration appropriate to the application of the datalogger. Consider the following factors when setting a calibration schedule:

- The importance of the measurements.
- How long the datalogger will be used.
- The operating environment.
- How the datalogger will be handled.

You can download and print calibration certificates for many products you have purchased by logging in to the Campbell Scientific website and going to: https://www.campbellsci.com/calcerts. Note, you will need your product's serial number to access its certificate. Watch an instructional video.

Datalogger Security

Datalogger security concerns include:

- Collection of sensitive data
- Operation of critical systems
- Networks accessible by many individuals

Some options to secure your datalogger from mistakes or tampering include:

- Sending the latest operating system to the datalogger. See "Updating the Operating System" on page 64 for more information.
- Disabling unused services and securing those that are used. This includes disabling HTTP, HTTPS, FTP, Telnet, and Ping network services (**Device Configuration Utility | Settings Editor | Network Services** tab). These services can be used to discover your datalogger on an IP network.
- Setting security codes (see following information under "Security Codes").
- Setting a PakBus/TCP password. The PakBus TCP password controls access to PakBus communication over a TCP/IP link. PakBusTCP passwords can be set in Device Configuration Utility.
- Disabling FTP or setting an FTP username and password in Device Configuration Utility.
- Setting a PakBus encryption (AES-128) key in Device Configuration Utility. This forces PakBus data to be encrypted during transmission.
- Disabling HTTP/HTTPS or creating a `.csipasswd` file to secure HTTP/HTTPS (see "Creating a .csipasswd File" on page 60 for more information).
- Enabling HTTPS and disabling HTTP. To prevent data collection via the web interface, both HTTP and HTTPS must be disabled.
- Tracking Operating System, Run, and Program signatures.
- Encrypting program files if they contain sensitive information (see CRBasic help **`FileEncrypt()`** instruction or use the CRBasic Editor **File** menu, **Save and Encrypt** option).
- Hiding program files for extra protection (see CRBasic help **`FileManage()`** instruction).
- Securing the physical datalogger and power supply under lock and key.
- Monitoring your datalogger for changes by tracking program and operating system signatures, as well as CPU file contents.

Warning: All security features can be subverted through physical access to the datalogger. If absolute security is a requirement, the physical datalogger must be kept in a secure location.

SECURITY CODES

The datalogger employs a security scheme that includes three levels of security. Security codes can effectively lock out innocent tinkering and discourage wannabe hackers on all communication links. However, any serious hacker with physical access to the datalogger or to the communications hardware can, with only minimal trouble, overcome the five-digit security codes.

Methods of enabling security codes include the following:

- Device Configuration Utility: Security codes are set on the **Deployment | Datalogger** tab.
- **`SetSecurity()`** instruction in CRBasic: **`SetSecurity()`** is only executed at program compile time.

Note: Deleting **`SetSecurity()`** from a CRBasic program is not equivalent to **`SetSecurity(0,0,0)`**. Settings persist when a new program is downloaded that has no **`SetSecurity()`** instruction.

Up to three levels of security codes can be set. Valid security codes are **1** through **65535** (**0** confers no security). **Security 1** must be set before **Security 2**. **Security 2** must be set before **Security 3**. If any one of the codes is set to **0**, any security code level greater than it will be set to **0**. For example, if **Security 2** is **0** then **Security 3** is automatically set to **0**. Security codes are unlocked in reverse order: **Security 3** before **Security 2**, **Security 2** before **Security 1**.

Functions affected by each level of security are:

- **Security 1**: Collecting data, setting the clock, and setting variables in the **Public** table are unrestricted, requiring no security code. If **Security 1** code is entered, read/write values in the **Status**, **Settings**, and **DataTableInfo** tables can be changed; fields can be edited in the **ConstTable**; and the datalogger program can be changed or retrieved.
- **Security 2**: Data collection is unrestricted, requiring no security code. If the user enters the **Security 2** code, the datalogger clock can be changed, values in the **Public** table can be changed, and read/write **DataTable Sample** fields can be edited.
- **Security 3**: When this level is set, all communication with the datalogger is prohibited if no security code is entered. If **Security 3** code is entered, data can be viewed and collected from the datalogger (except data suppressed by the **`TableHide()`** instruction).

For additional information on datalogger security, see:

- 4 Ways to Make your Data More Secure
- Available Security Measures for Internet-Connected Dataloggers
- How to Use Datalogger Security Codes
- How Can Data be Made More Secure on a CRBasic PakBus Datalogger

CREATING A .CSIPASSWD FILE

The datalogger employs a security scheme that includes three levels of security (see "Datalogger Security" on page 58 for more information). This scheme can be used to limit access to a datalogger that is publicly available. However, the security code is visible in Device Configuration Utility. In addition, the range of codes is relatively small. To provide a more robust means of security, Basic Access Authentication was implemented with the HTTP API interface in the form of an encrypted password file named `.csipasswd`. In order to provide read/write access to the web interface, you must create a `.csipasswd` file. This file is also used to access the embedded web interface, which provides access to real-time and stored datalogger data. For more information on the web interface, watch an instructional video.

When a file named `.csipasswd` is stored on the datalogger's CPU drive, basic access authentication is enabled in the datalogger and read/write access to the web interface can be defined. Multiple user accounts/levels of access can be defined for one datalogger. Four levels of access are available:

- **None**: Disable a user account.
- **Read Only**: Data collection is unrestricted and available. Clock and writable variables cannot be changed; programs cannot be viewed, stopped, deleted, or retrieved.
- **Read/Write**: Data collection is unrestricted and available, and clock and writable variables can be changed. Programs cannot be viewed, stopped, deleted, or retrieved.
- **All**: Data collection is unrestricted and available, clock and writable variables can be changed; and programs can be viewed, stopped, deleted and retrieved.

Note: All levels of access allow data collection.

Create an encrypted password file or modify an existing password file using Device Configuration Utility:

1. Connect to your device in Device Configuration Utility.
2. Click the **Network Services** tab, then click the **Edit .csipasswd File** button.
3. Define your user accounts and access levels.
4. Click **Apply**. The `.csipasswd` file is automatically saved to the datalogger's CPU drive.

When a `.csipasswd` file enables basic access authentication, the datalogger's PakBus/TCP Password security setting is not used when accessing the datalogger via HTTP. If the `.csipasswd` file is blank or does not exist, the default user name is "anonymous" with no password and a user level of read only.

When access to the datalogger web server is attempted without the appropriate security level, the datalogger will prompt the web client to display a user name/password request dialog. If an invalid username or password is entered, the datalogger web server will default to the level of access assigned to "anonymous". As noted above, anonymous is assigned a user level of read-only, though this can be changed using Device Configuration Utility.

If the numeric security code has been enabled using Device Configuration Utility, the **`SetSecurity()`** instruction, or the **Security() Status** table settings, and no `.csipasswd` file is on the datalogger, then that numeric security code must be entered to access the datalogger. If a `.csipasswd` file is on the datalogger, the User Name and Password employed by the Basic Access Authentication will eliminate the need for entering the numeric security code.

Command Syntax

Syntax for the commands sent to the web server generally follows the form of:

```
URL?command=CommandName&uri=DataSource&arguments
```

Arguments are appended to the command string using an ampersand (&). Some commands have optional arguments, where omitting the argument results in a default being used. When applicable, optional arguments and their defaults are noted and examples are provided in the CRBasic help (search Web Server/API Commands).

Datalogger Enclosures

The datalogger and most of its peripherals must be protected from moisture and humidity. Moisture in the electronics will seriously damage the datalogger. In most cases, protection from moisture is easily accomplished by placing the datalogger in a weather-tight enclosure with desiccant and elevating the enclosure above the ground. Desiccant in enclosures should be changed periodically.

Warning: Do not completely seal the enclosure if lead-acid batteries are present; hydrogen gas generated by the batteries may build up to an explosive concentration.

The following details a typical installation using a Campbell Scientific enclosure. The datalogger has mounting holes through which small screws are inserted into nylon anchors in the backplate.

1. Insert the three included nylon anchors into the backplate. Position them to align with the three mounting holes on the base of the datalogger.

2. Holding the datalogger to the backplate, screw the three included screws into the nylon anchors.

See also "Physical Specifications" on page 87.

Internal Battery

The lithium battery powers the internal clock when the datalogger is not powered. The internal lithium battery has a six-year life when no external power source is applied. Its life is extended when the datalogger is installed with an external power source. If the datalogger is used in a high-temperature application, the battery life is shortened.

> **Note**: The **Status** field **Battery** value and the destination variable from the **`Battery()`** instruction (often called `batt_volt`) in the **Public** table reference the external battery voltage.

To prevent clock and memory issues, it is recommended you proactively replace the battery every 5 years, or more frequently when operating continuously in high temperatures. For example, replace at least every 2 years if operating continuously at 80 °C.

> **Note**: The battery is replaced during regular factory recalibration, which is recommended every 3 years. For more information, see "Datalogger Calibration" on page 58.

When the lithium battery is removed (or is depleted and primary power to the datalogger is removed), the CRBasic program and most settings are maintained, but the following are lost:

- Temporary program memory (current values for variables).
- Routing and communication logs (relearned without user intervention).
- Time. Clock will need resetting when the battery is replaced.

A replacement lithium battery can be purchased from Campbell Scientific or another supplier. See "Power Requirements" on page 87 for more information.

> **Warning**: Misuse or improper installation of the internal lithium battery can cause severe injury. Fire, explosion, and severe burns can result. Do not recharge, disassemble, heat above 100 °C (212 °F), solder directly to the cell, incinerate, or expose contents to water. Dispose of spent lithium batteries properly.

REPLACING THE INTERNAL BATTERY

It is recommended that you send the datalogger in for scheduled calibration, which includes internal battery replacement (see "Datalogger Calibration" on page 58).

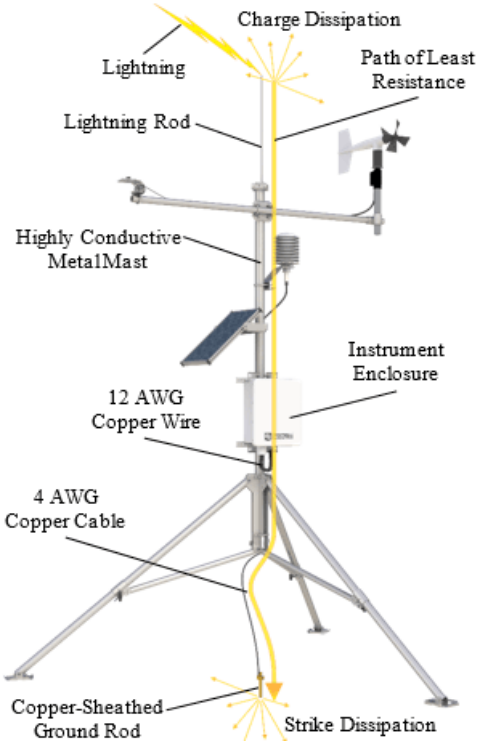
Warning: Any damage made to the datalogger during replacement of the internal battery is not covered under warranty.

1. Loosen or remove the screws from the sides of the datalogger.
2. Pull the bottom of the enclosure away from the top of the enclosure.
3. Carefully pinch each of the four plastic pins and pull the top circuit board away from the mounts and connector. A wire connects the board, do not pull the board too far and damage the wire connection.
4. Remove the lithium battery by gently prying it out with a small flat point screwdriver. Reassemble the datalogger. Take particular care to ensure the board is reseated tightly onto the plastic pins and connector.

Electrostatic Discharge and Lightning Protection

Warning: Lightning strikes may damage or destroy the datalogger and associated sensors and power supplies.

Electrostatic discharge (ESD) can originate from several sources, the most common and destructive being primary and secondary lightning strikes. Primary lightning strikes hit instrumentation directly. Secondary strikes induce voltage in power lines or wires connected to instrumentation. While elaborate, expensive, and nearly infallible lightning protection systems are on the market, Campbell Scientific, for many years, has employed a simple and inexpensive design that protects most systems in most circumstances. The system employs a lightning rod, metal mast, heavy-gauge ground wire, and ground rod to direct damaging current away from the datalogger. This system, however, is not infallible. The following image displays a typical application of the system:



All critical inputs and outputs on the datalogger are ESD protected to 75 V. To be effective, the earth ground lug must be properly connected to earth (chassis) ground.

Communication ports are another path for transients. You should provide communication paths, such as telephone or short-haul modem lines, with spark-gap protection. Spark-gap protection is usually an option with these products, so request it when ordering. Spark gaps must be connected to either the earth ground lug, the enclosure ground, or to the earth (chassis) ground.

For detailed information on grounding, see "Grounds" on page 7.

Power Budgeting

In low-power situations, the datalogger can operate for several months on non-rechargeable batteries. Power systems for longer-term remote applications typically consist of a charging source, a charge controller, and a rechargeable battery. When ac line power is available, a Vac-to-Vdc wall adapter, the onboard charging regulator, and a rechargeable battery can be used to construct an uninterruptible power supply (UPS).

When designing a power supply, consider worst-case power requirements and environmental extremes. For example, the power requirement of a weather station may be substantially higher during extreme cold, while at the same time, the extreme cold constricts the power available from the power supply. System operating time for batteries can be determined by dividing the battery capacity (ampere hours) by the average system current drain (amperes).

For more information see:

- **Application Note** - Power Supplies
- Power Budget Spreadsheet
- **Video Tutorial** - Power Budgeting

See also:

- "Power Input" on page 5
- "Power Output" on page 6

Updating the Operating System

Campbell Scientific posts operating system (OS) updates at www.campbellsci.com/downloads when they become available. It is recommended that before deploying instruments, you check operating system versions and update them as needed. The datalogger operating system version is shown in the **Status** table, **Station Status Summary**, and Device Configuration Utility **Deployment | Datalogger**. An operating system may be sent through Device Configuration Utility or through program-send procedures.

Warning: Because sending an OS resets datalogger memory and resets all settings on the datalogger to factory defaults, data loss will certainly occur. Depending on several factors, the datalogger may also become incapacitated for a time. It is recommended that you retrieve data from the datalogger and back up your programs before updating your OS; otherwise, data may be lost. To collect data using LoggerNet, connect to your datalogger and click the **Collect Now** button (). To backup your datalogger, connect to it in Device Configuration Utility, click the **Backup** menu and select **Backup Datalogger**.

SENDING AN OPERATING SYSTEM TO A LOCAL DATALOGGER

Send an OS using Device Configuration Utility. This method requires a direct connection between your datalogger and computer.

1. Download the latest Operating System at www.campbellsci.com/downloads.
2. Locate the .exe download and double-click to run the file. This will extract the .obj OS file to the **C:\Campbellsci\Lib\OperatingSystems** folder.
3. Supply power to the datalogger. If connecting via USB for the first time, you must first install USB drivers by using Device Configuration Utility (select your datalogger, then on the main page, click **Install USB Driver**). Alternately, you can install the USB drivers using EZ Setup. A USB connection supplies 5 V power (as well as a communication link), which is adequate for setup, but a 12 V battery will be needed for field deployment.
4. Physically connect your datalogger to your computer using a USB cable, then open Device Configuration Utility and select your datalogger.
5. Select the communication port used to communicate with the datalogger from the **COM Port** list (you do not need to click **Connect**).
6. Click the **Send OS** tab. At the bottom of the window, click **Start**.
7. On the **Avoid Conflicts with the Local Server** window, click **OK**.
8. Navigate to the **C:\Campbellsci\Lib\OperatingSystems** folder.
9. Ensure **Datalogger Operating System Files (*.obj)** is selected in the **Files of type** list, select the new OS .obj file, and click **Open** to update the OS on the datalogger.

Watch a video: Sending an OS to a Local Datalogger.

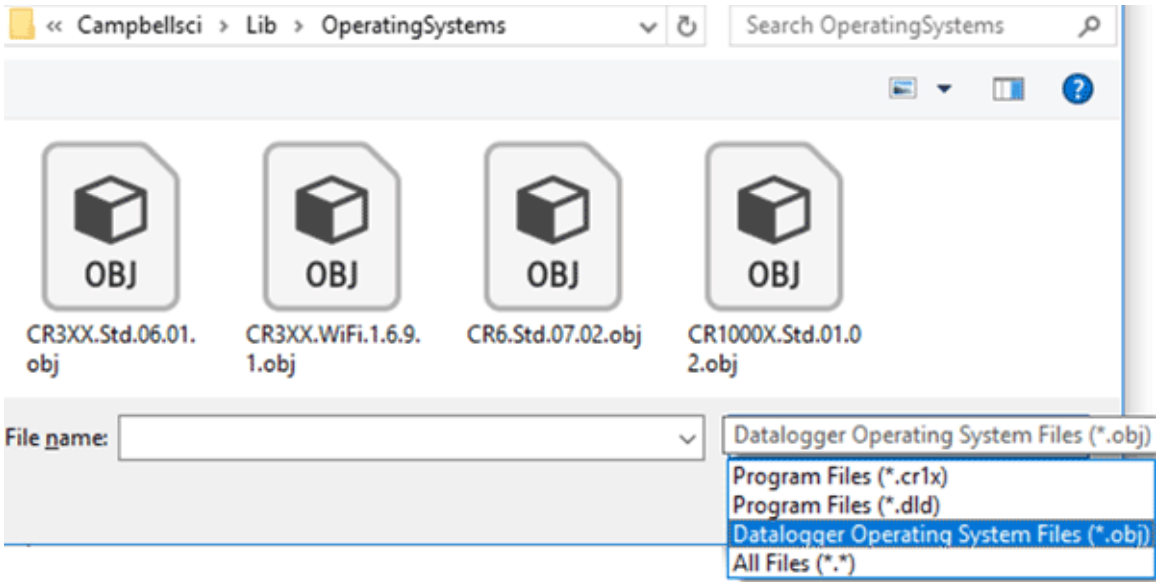
SENDING AN OPERATING SYSTEM TO A REMOTE DATALOGGER

Caution: Sending an OS remotely to a datalogger with Operating System 4 or earlier is not recommended. These dataloggers should be updated using the previous instruction ("Sending an Operating System to a Local Datalogger" on page 65).

If you have a datalogger that is already deployed, you can update the OS over a telecommunications link by sending the OS to the datalogger as a program. In most instances, sending an OS as a program preserves settings. This allows for sending supported operating systems remotely (check the release notes). However, this should be done with great caution as updating the OS may reset the datalogger settings, even settings critical to supporting the telecommunication link.

1. Download the latest Operating System at www.campbellsci.com/downloads.
2. Locate the .exe download and double-click to run the file. This will extract the .obj OS file to the **C:\Campbellsci\Lib\OperatingSystems** folder.
3. Using datalogger support software, connect to your datalogger.
   - LoggerNet users, select **Main** and click the **Connect** button () on the LoggerNet toolbar, select the datalogger from the **Stations** list, then click the **Connect** button ().
   - PC200W and PC400 users, select the datalogger from the list and click the **Connect** button ().
4. Click **File Control** () at the top of the Connect window.

5. Click **Send** ( ) at the top of the File Control window.
6. Navigate to the **C:\Campbellsci\Lib\OperatingSystems** folder.
7. Ensure **Datalogger Operating System Files (*.obj)** is selected in the files of type list, select the new OS .obj file, and click **Open** to update the OS on the datalogger.



Note the following precautions when sending as a program:

- Any peripherals being powered through the **SW12** terminal will be turned off until the program logic turns them on again.
- Operating systems are very large files. Be cautious of data charges. Sending over a direct serial or USB connection is recommended, when possible.

Tips and Troubleshooting

The following basic procedure can be used if a system is not operating properly.

1. Using a voltmeter, check the voltage of the primary power source at the **CHG** and **BAT** terminals on the face of the datalogger. If connecting to a power source via the **BAT** terminal, it should be 10 to 18 Vdc. If connecting to a power source via the **CHG** terminals, voltage measured should be 16 to 32 Vdc.
2. Check wires and cables for the following:
    - Incorrect wiring connections. Make sure each sensor and device are wired to the channels assigned in the program. If the program was written in Short Cut, check wiring against the generated wiring diagram. If written in CRBasic Editor, check wiring against each measurement and control instruction.
    - Loose connection points.
    - Faulty connectors.
    - Cut wires.
    - Damaged insulation, which allows water to migrate into the cable. Water, whether or not it comes in contact with wire, can cause system failure. Water may increase the dielectric constant of the cable sufficiently to impede sensor signals, or it may migrate into the sensor, which will damage sensor electronics.
3. Check the CRBasic program. If the program was written solely with Short Cut, the program is probably not the source of the problem. If the program was written or edited with CRBasic Editor, logic and syntax errors could easily have crept into the code. To troubleshoot, create a stripped-down version of the program, or break it up into multiple smaller units to test individually. For example, if a sensor signal-to-data conversion is faulty, create a program that only measures that sensor and stores the data, absent from all other inputs and data.
4. Reset the datalogger. Sometimes the easiest way to resolve a problem is by resetting the datalogger (see "Resetting the Datalogger" on page 72 for more information).

For additional troubleshooting options, see:

You can also find a number of troubleshooting articles at: www.campbellsci.com/blog/troubleshooting-topics.

Checking Station Status

View the condition of the datalogger using **Station Status**. Here you can see the operating system version of the datalogger, the name of the current program, program compile results, and other key indicators. Items that may need your attention appear in red or blue. The following information describes the significance of some entries in the station status window. Watch a video or use the following instructions.

VIEWING STATION STATUS

Using your datalogger support software, access the **Station Status** to view the condition of the datalogger.

- From LoggerNet: Click **Connect** (), then click the **Station Status** button to view the **Summary** tab.
- From PC200W and PC400: Click the **Datalogger** menu and select **Station Status** to view the **Summary** tab.

WATCHDOG ERRORS

Watchdog errors indicate that the datalogger has crashed and reset itself. Experiencing a few watchdog errors is normal. You can reset the Watchdog error counter (noting the date you did so) in the **Station Status|Status Table**. Watchdog errors could be due to:

- Transient voltage.
- Miswired or malfunctioning sensor.
- Poor ground connection on the power supply.
- Running the CRBasic program very fast.
- Numerous `PortSet()` instructions back-to-back with no delay.
- High-speed serial data on multiple ports with very large data packets or bursts of data.

The error "Results for Last Program Compiled: Warning: Watchdog Timer IpTask Triggered" can result from:

- The IP communication on the datalogger got stuck, and the datalogger had to reboot itself to recover. Or communications failures may cause the datalogger to reopen the IP connections more than usual. Check your datalogger operating system version; recent operating system versions have improved stability of IP communications.

If any of these are not the apparent cause, contact Campbell Scientific for assistance (see https://www.campbellsci.com/support). Causes that may require assistance include:

- Memory corruption
- Operating System problem
- Hardware problem
- IP communications problem

RESULTS FOR LAST PROGRAM COMPILED

Messages generated by the datalogger at program upload and as the program runs are reported here. Warnings indicate that an expected feature may not work, but the program will still operate. Errors indicate that the program cannot run. For more information, see "CRBasic Program Errors" on page 71.

SKIPPED SCANS

Skipped scans are caused when a program takes longer to process than the scan rate allows. An occasional skipped scan can be caused by memory formatting as discussed in as discussed in "Memory and Data Storage" on page 37. If any scan skips repeatedly, the datalogger program may need to be optimized or reduced. For more information, see: How to Prevent Skipped Scans and a Sunburn.

SKIPPED RECORDS

Skipped records usually occur because a scan is skipped. They indicate that a record was not stored to the data table when it should have been.

VARIABLE OUT OF BOUNDS

Variable-out-of-bounds errors happen when an array is not sized to the demands of the program. The datalogger attempts to catch all out-of-bounds errors at compile time and document the error. However, it is not always possible, so these errors may occur during runtime.

BATTERY VOLTAGE

If powering through USB, reported battery voltage should be 0 V. If connecting to an external power source, battery voltage should be reported at or near 12 V. See also:

- "Power Input" on page 5
- "Power Requirements" on page 87

Understanding NAN and INF Occurrences

NAN (not a number) and INF (infinite) are data words indicating an exceptional occurrence in datalogger function or processing. **INF** indicates that the program has encountered an arithmetic expression that is undefined. **NAN** indicates an invalid measurement. For more information, see Tips and Tricks: Who's NAN?

NANs are expected in the following conditions:

- Input signals exceed the voltage range chosen for the measurement.
- An invalid SDI-12 command is sent
- An SDI-12 sensor does not respond or aborts without sending data
- Undefined arithmetic expressions, such as 0 ÷ 0.

NAN is a constant that can be used in expressions. This is shown in the following code snip that sets a CRBasic control feature (a flag) if the wind direction is **NAN**:

```
If WindDir = NAN Then
  WDFlag = False
Else
  WDFlag = True
EndIf
```

If a NAN is included in the values being processed, NAN will be stored. Note that since there is no such thing as NAN for integers, values that are converted from float to integer will be expressed in data tables as the most negative number for a given data type. For example, the most negative number of data type FP2 is –7999, so NAN for FP2 data will appear in a data table as –7999. If the data type is Long, NAN will appear in the data table as –2,147,483,648.

Because NAN is a constant, it can be used in conjunction with the disable variable parameter (*DisableVar*) in output processing instructions. Use the *DisableVar* parameter to call the output table conditionally (for example, do not call the table if a variable = NAN) to keep NANs from affecting the other good values.

Time Keeping

Measurement of time is an essential function of the datalogger. Time measurement with the onboard clock enables the datalogger to attach time stamps to data, measure the interval between events, and time the initiation of control functions. Details on clock accuracy and resolution are available in the "System Specifications" on page 86. An internal lithium battery backs the clock when the datalogger is not externally powered (see "Internal Battery" on page 62 for more information).

CLOCK BEST PRACTICES

If you are going to set the clock with LoggerNet, initiate it manually during a maintenance period when you are already disrupting site data (clock settings can be found on the Loggernet Standard View **Clock** tab).

If you are going to use automated clock check with LoggerNet, it is recommended that you do this on the order of days (not hours). Set an allowed clock deviation that is appropriate for the expected jitter in the network, and use the initial time setting to offset the clock check away from storage and measurement intervals.

Take into account the amount of time that is required for the **Clock Check** command to reach the datalogger, be processed, and for it to send its response (round-trip time, or time-of-flight). LoggerNet maintains a history of the round trip times for up to the ten previous successful clock check transactions in order to calculate an estimate of this time of flight. It adds this average to the time values received from the datalogger and subtracts it from any adjustment that it might make.

The **ClockChange** instruction can be used to:

- Detect, track, and log externally triggered clock change events.
- Characterize the jitter in your clock-syncing method.

TIME STAMPS

A measurement without an accurate time reference has little meaning. Data on the datalogger are stored with time stamps. How closely a time stamp corresponds to the actual time a measurement is taken depends on several factors.

The time stamp in common CRBasic programs matches the time at the beginning of the current scan as measured by the real-time clock in the datalogger. If a scan starts at 15:00:00, data output during that scan will have a time stamp of **15:00:00** regardless of the length of the scan or when in the scan a measurement is made. The possibility exists that a scan will run for some time before a measurement is made. For instance, a scan may start at 15:00:00, execute time-consuming code, then make a measurement at 15:00:00.51. The time stamp attached to the measurement, if the **CallTable** instruction is called from within the **Scan() / NextScan** construct, will be **15:00:00**, resulting in a time-stamp skew of 510 ms.

AVOIDING TIME SKEW

Time skew between consecutive measurements is a function of settling and integration times, ADC, and the number entered into the ***Reps*** parameter of CRBasic instructions. A close approximation is:

time skew = reps * (settling time + integration time + ADC time) + instruction setup time
where ADC time equals 170 µs, and instruction setup time is 15 µs.
If reps (repetitions) > 1 (multiple measurements by a single instruction), no setup time is required.
If reps = 1 for consecutive voltage instructions, include the setup time for each instruction.

Time-stamp skew is not a problem with most applications because:

- Program execution times are usually short, so time stamp skew is only a few milliseconds. Most measurement requirements allow for a few milliseconds of skew.
- Data processed into averages, maxima, minima, and so forth are composites of several measurements. Associated time stamps only reflect the time the last measurement was made and processing calculations were completed, so the significance of the exact time a specific sample was measured diminishes.

Applications measuring and storing sample data wherein exact time stamps are required can be adversely affected by time-stamp skew. Skew can be avoided by:

- Making measurements in the scan before time-consuming code.
- Programming the datalogger such that the time stamp reflects the system time rather than the scan time using the **DataTime()** instruction. See topics concerning data table declarations in CRBasic Editor help for more information.

CRBasic Program Errors

Analyze data soon after deployment to ensure the datalogger is measuring and storing data as intended. Most measurement and data-storage problems are a result of one or more CRBasic program bugs. Watch a video: CRBasic | Common Errors - Identifying and fixing common errors in the CRBasic programming language.

PROGRAM DOES NOT COMPILE

Although the CRBasic Editor compiler states that a program compiles OK, the program may not run or even compile in the datalogger. This is rare, but reasons may include:

- The datalogger has a different operating system that is not fully compatible with the computer compiler. Check the two versions if in doubt. The computer compiler version is shown on the first line of the compile results. Update the computer compiler by first downloading the executable OS file from www.campbellsci.com. When run, the executable file updates the computer compiler. To update the datalogger operating system, see "Updating the Operating System" on page 64.
- The program has large memory requirements for data tables or variables and the datalogger does not have adequate memory. This normally is flagged at compile time in the compile results. If this type of error occurs, check:
    - Copies of old programs on the CPU drive. The datalogger keeps copies of all program files unless they are deleted, the drive is formatted, or a new operating system is loaded with Device Configuration Utility.

PROGRAM COMPILES BUT DOES NOT RUN CORRECTLY

If the program compiles but does not run correctly, timing discrepancies are often the cause. If a program is tight on time, look further at the execution times. Check the measurement and processing times in the **Status** table (**MeasureTime**, **ProcessTime**, **MaxProcTime**) for all scans, then try experimenting with the **`InstructionTimes()`** instruction in the program. Analyzing `InstructionTimes()` results can be difficult due to the multitasking nature of the logger, but it can be a useful tool for fine-tuning a program.

Troubleshooting Radio Communication Problems

If there are intermittent communication problems when connecting via radio, there may be another network in the area causing interference. To help remove the interference, use Device Configuration Utility to change the **Network ID** and **RF Hop Sequence** in all RF407, RF412, and RF422 radios within a network (standalone or included in a datalogger) to another value. Each of these settings must have the same value in all radios and dataloggers within a network. For example, the **Network ID** in all devices could be set to **1726**, and the **RF Hop Sequence** in all devices could be set to **1**. The **Network ID** can be any number between 0 and 32767. The **RF Hop Sequence** can be any number between 0 and 7 in an RF407 or RF412 network; it can be any number between 0 and 9 in an RF422 network.

See also "Radio Communications" on page 21. For specifications information, see "RF Radio Option Specifications" on page 95.

Reducing Out of Memory Errors

When the datalogger gives an “Out of memory” error upon program compile, it means that there is no unused continuous section of memory large enough to fit an element of the program. The most common cause of this error is a large variable array. In that case, split the large array into smaller arrays until the datalogger compiler accepts the program.

See also "Memory and Data Storage" on page 37.

Resetting the Datalogger

A reset is referred to as a "memory reset." Be sure to back up the current datalogger configuration before a reset in case you need to revert to the old settings. To back up the datalogger configuration, connect to the datalogger using Device Configuration Utility, and click **Backup** | **Back Up Datalogger**. To restore a configuration after the datalogger has been reset, connect and click **Backup** | **Restore Datalogger**.

The following features are available for complete or selective reset of datalogger memory:

- Processor reset.
- Program send reset.
- Manual data table reset.
- Formatting memory drives.
- Full memory reset.

PROCESSOR RESET

To reset the processor, simply power cycle the datalogger. This resets its short-term memory, restarts the current program, sets variables to their starting values, and clears communication buffers. This does

not clear data tables but may result in a skipped record. If the datalogger is remote, a power cycle can be mimicked in the **Terminal Emulator** program (type REBOOT <Enter>) or by using the `Restart` CRBasic instruction.

PROGRAM SEND RESET

Final-storage data are erased when user programs are uploaded, unless preserve / erase data options are used and the program was not altered. Preserve / erase data options are presented when sending programs using File Control **Send** command and CRBasic Editor **Compile, Save and Send**.

When a program compiles, all variables are initialized. A program is recompiled after a power failure or a manual stop. For instances that require variables to be preserved through a program recompile, the CR300 series have the `PreserveVariables()` instruction and User Settings. Entries in User Settings are not erased when a program is recompiled.

User Settings can be entered manually in Device Configuration Utility in the **Settings Editor | User Settings** tab, or they can be written through a CRBasic program using the `SetSetting()` instruction.

Note: User settings are stored in flash memory. To avoid corrupting the flash memory, edit these settings infrequently.

MANUAL DATA TABLE RESET

Data table memory is selectively reset from:

- Datalogger support software: Station Status **Table Fill Times** tab, **Reset Tables**.
- Device Configuration Utility: **Data Monitor** tab, **Reset Table** button.

FORMATTING DRIVES

CPU, CRD (memory card required), and USB (module required) drives can be formatted individually. Formatting a drive erases all files on that drive. If the currently running user program is on the drive to be formatted, the program will cease running. Drive formatting is performed through the datalogger support software **File Control | Format** command.

FULL MEMORY RESET

Full memory reset occurs when an operating system is sent to the datalogger using Device Configuration Utility or when entering 98765 in the **Status** table field **FullMemReset**. A full memory reset does the following:

- Clears and formats CPU drive (all program files erased).
- Clears **Status** table elements.
- Restores settings to default.
- Initializes system variables.
- Clears communication memory.

Full memory reset does not affect the CRD drive directly. Subsequent user program uploads, however, can erase CRD. See "Updating the Operating System" on page 64 for more information.

Troubleshooting Power Supplies

Power supply systems may include batteries, charging regulators, and a primary power source such as solar panels or ac/ac or ac/dc transformers attached to mains power. All components may need to be checked if the power supply is not functioning properly. Check connections and check polarity of connections.

Base diagnostic: connect the datalogger to a new 12 V battery. (A small 12 V battery carrying a full charge would be a good thing to carry in your troubleshooting tool kit.) Ensure correct polarity of the connection. If the datalogger powers up and works, troubleshoot the datalogger power supply.

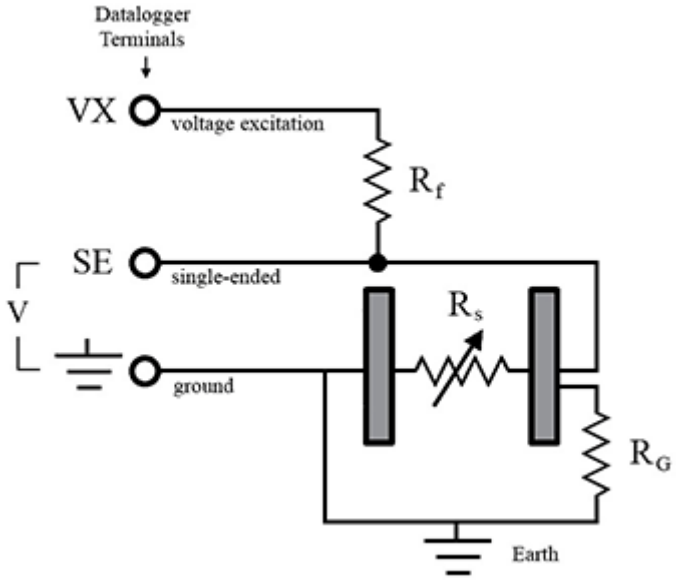
When diagnosing or adjusting power equipment supplied by Campbell Scientific, it is recommended you consider:

- Battery-voltage test.
- Charging-circuit test (when using an unregulated solar panel).
- Charging-circuit test (when using a transformer).
- Adjusting charging circuit.

If power supply components are working properly and the system has peripherals with high current drain, such as a satellite transmitter, verify that the power supply is designed to provide adequate power. For additional information, see "Power Budgeting" on page 64.

Minimizing Ground Loop Errors

When measuring soil moisture with a resistance block, or water conductivity with a resistance cell, the potential exists for a ground loop error. In the case of an ionic soil matric potential (soil moisture) sensor, a ground loop arises because soil and water provide an alternate path for the excitation to return to datalogger ground. This example is modeled in the following image:



With R_g in the resistor network, the signal measured from the sensor is described by the following equation:

$$V_1 = V_x \frac{R_s}{(R_s + R_f) + R_s R_f / R_g}$$

where

- V_x is the excitation voltage
- R_f is a fixed resistor
- R_s is the sensor resistance
- R_g is the resistance between the excited electrode and datalogger earth ground.

$R_s R_f / R_g$ is the source of error due to the ground loop. When R_g is large, the error is negligible. Note that the geometry of the electrodes has a great effect on the magnitude of this error. The Delmhorst gypsum block used in the Campbell Scientific 227 probe has two concentric cylindrical electrodes. The center electrode is used for excitation; because it is encircled by the ground electrode, the path for a ground loop through the soil is greatly reduced. Moisture blocks which consist of two parallel plate electrodes are particularly susceptible to ground loop problems. Similar considerations apply to the geometry of the electrodes in water conductivity sensors.

The ground electrode of the conductivity or soil moisture probe and the datalogger earth ground form a galvanic cell, with the water/soil solution acting as the electrolyte. If current is allowed to flow, the resulting oxidation or reduction will soon damage the electrode, just as if dc excitation was used to make the measurement. Campbell Scientific resistive soil probes and conductivity probes are built with series capacitors to block this dc current. In addition to preventing sensor deterioration, the capacitors block any dc component from affecting the measurement.

See also "Grounds" on page 7.

Improving Voltage Measurement Quality

The following topics discuss methods of generally improving voltage measurements:

Read More: Consult the following technical papers at www.campbellsci.com/app-notes for in-depth treatments of several topics addressing voltage measurement quality:

- Preventing and Attacking Measurement Noise Problems
- Benefits of Input Reversal and Excitation Reversal for Voltage Measurements
- Voltage Accuracy, Self-Calibration, and Ratiometric Measurements

DECIDING BETWEEN SINGLE-ENDED OR DIFFERENTIAL MEASUREMENTS

Deciding whether a differential or single-ended measurement is appropriate is usually, by far, the most important consideration when addressing voltage measurement quality. The decision requires trade-offs of accuracy and precision, noise cancellation, measurement speed, available measurement hardware, and fiscal constraints.

In broad terms, analog voltage is best measured differentially because these measurements include noise reduction features, listed below, that are not included in single-ended measurements.

- Passive Noise Rejection
    - No voltage reference offset
    - Common-mode noise rejection, which filters capacitively coupled noise
- Active Noise Rejection

Reasons for using single-ended measurements, however, include:

- Not enough differential terminals are available. Differential measurements use twice as many analog input terminals as do single-ended measurements.
- Rapid sampling is required. Single-ended measurement time is about half that of differential measurement time.
- Sensor is not designed for differential measurements. Many Campbell Scientific sensors are not designed for differential measurement, but the drawbacks of a single-ended measurement are usually mitigated by large programmed excitation and/or sensor output voltages.

Sensors with a high signal-to-noise ratio, such as a relative-humidity sensor with a full-scale output of 0 to 1000 mV, can normally be measured as single-ended without a significant reduction in accuracy or precision.

Sensors with a low signal-to-noise ratio, such as thermocouples, should normally be measured differentially. However, if the measurement to be made does not require high accuracy or precision, such as thermocouples measuring brush-fire temperatures, which can exceed 2500 °C, a single-ended measurement may be appropriate. If sensors require differential measurement, but adequate input terminals are not available, an analog multiplexer should be acquired to expand differential input capacity.

Because a single-ended measurement is referenced to datalogger ground, any difference in ground potential between the sensor and the datalogger will result in an error in the measurement. For more information on grounds, see "Grounds" on page 7 and "Minimizing Ground Potential Differences" on page 76.

MINIMIZING GROUND POTENTIAL DIFFERENCES

Low-level, single-ended voltage measurements (<200 mV) are sensitive to ground potential fluctuation due to changing return currents from SW12 and C terminals. The datalogger grounding scheme is designed to minimize these fluctuations by separating signal grounds (⏚) from power grounds (G). For more information on datalogger grounds, see "Grounds" on page 7. To take advantage of this design, observe the following rules:

- Connect grounds associated with SW12 and C terminals to G terminals.
- Connect excitation grounds to the nearest ⏚ terminal on the same terminal block.

- Connect the low side of single-ended sensors to the nearest ⏚ terminal on the same terminal block.
- Connect shield wires to the ⏚ terminal nearest the terminals to which the sensor signal wires are connected.

If offset problems occur because of shield or ground leads with large current flow, tying the problem leads into terminals next to terminals configured for excitation and pulse-count should help. Problem leads can also be tied directly to the ground lug to minimize induced single-ended offset voltages.

Ground Potential Differences

Because a single-ended measurement is referenced to datalogger ground, any difference in ground potential between the sensor and the datalogger will result in a measurement error. Differential measurements MUST be used when the input ground is known to be at a different ground potential from datalogger ground.

Ground potential differences are a common problem when measuring full-bridge sensors (strain gages, pressure transducers, etc), and when measuring thermocouples in soil.

- **Soil Temperature Thermocouple**: If the measuring junction of a thermocouple is not insulated when in soil or water, and the potential of earth ground is, for example, 1 mV greater at the sensor than at the point where the datalogger is grounded, the measured voltage is 1 mV greater than the thermocouple output. With a copper-constantan thermocouple, 1 mV equates to approximately 25 °C measurement error.
- **External Signal Conditioner**: External instruments with integrated signal conditioners, such as an infrared gas analyzer (IRGA), are frequently used to make measurements and send analog information to the datalogger. These instruments are often powered by the same Vac-line source as the datalogger. Despite being tied to the same ground, differences in current drain and lead resistance result in different ground potentials at the two instruments. For this reason, a differential measurement should be made on the analog output from the external signal conditioner.

MINIMIZING POWER-RELATED ARTIFACTS

Some Vac-to-Vdc power converters produce switching noise or ac ripple as an artifact of the ac-to-dc rectification process. Excessive switching noise on the output side of a power supply can increase measurement noise, and so increase measurement error. Noise from grid or mains power also may be transmitted through the transformer, or induced electromagnetically from nearby motors, heaters, or power lines.

High-quality power regulators typically reduce noise due to power regulation. Using the 50 Hz or 60 Hz `fN1` option for CRBasic analog input measurement instructions often improves rejection of noise sourced from power mains.

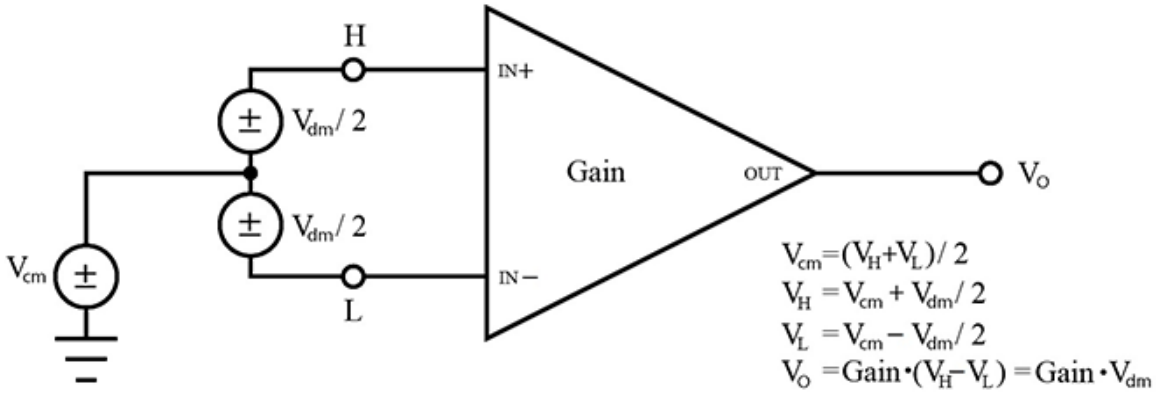
The datalogger includes adjustable digital filtering, which serves two purposes:

- Arrive as close as possible to the true input signal
- Filter out measurement noise at specific frequencies, the most common being noise at 50 Hz or 60 Hz, which originate from mains-power lines.

Filtering time is inversely proportional to the frequency being filtered.

Minimizing Electronic Noise

Electronic noise can cause significant error in a voltage measurement, especially when measuring voltages less than 200 mV. So long as input limitations are observed, the PGIA ignores voltages, including noise, that are common to each side of a differential-input pair. This is the common-mode voltage. Ignoring (rejecting or canceling) the common-mode voltage is an essential feature of the differential input configuration that improves voltage measurements. The following image illustrates the common-mode component (V_{cm}) and the differential-mode component (V_{dm}) of a voltage signal. V_{cm} is the average of the voltages on the V+ and V- inputs. So, V_{cm} = (V+ + V-)/2 or the voltage remaining on the inputs when V_{dm} = 0. The total voltage on the V+ and V- inputs is given as $V_H = V_{cm} + V_{dm}/2$, and $V_L = V_{cm} - V_{dm}/2$, respectively.



FILTERING TO REDUCE MEASUREMENT NOISE

The datalogger applies an adjustable filter to analog measurements, reducing signal components at selected frequencies. The following figures show the frequency response of the filters applied when the first notch frequency (***fN1***) is set to 4000, 400, or 50/60 Hz, respectively. Note that the same filter is applied when ***fN1*** is set to either 50 or 60 Hz, simultaneously filtering both 50 and 60 Hz signal components. Commonly, ***fN1*** is set at 50 or 60 Hz in order to filter 50 or 60 Hz signal components, reducing noise from ac power mains.

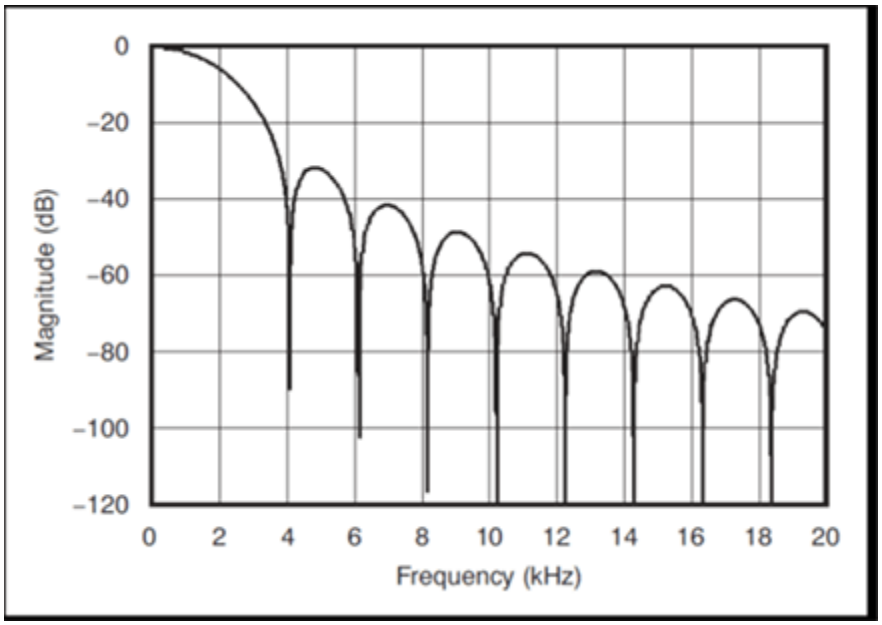
Filtering comes at the expense of measurement time. The time required for filtering is 0.5 ms when ***fN1*** is set to 4000 Hz, 6.226 ms when ***fN1*** is 400 Hz, and 49.812 ms when ***fN1*** is set to either 50 or 60 Hz. Random noise in the measurement results decreases, while measurement time increases, as ***fN1*** is set to smaller values. The total time required for a single result includes settling + filtering + overhead.

A faster filter may be preferred to achieve the following objectives:

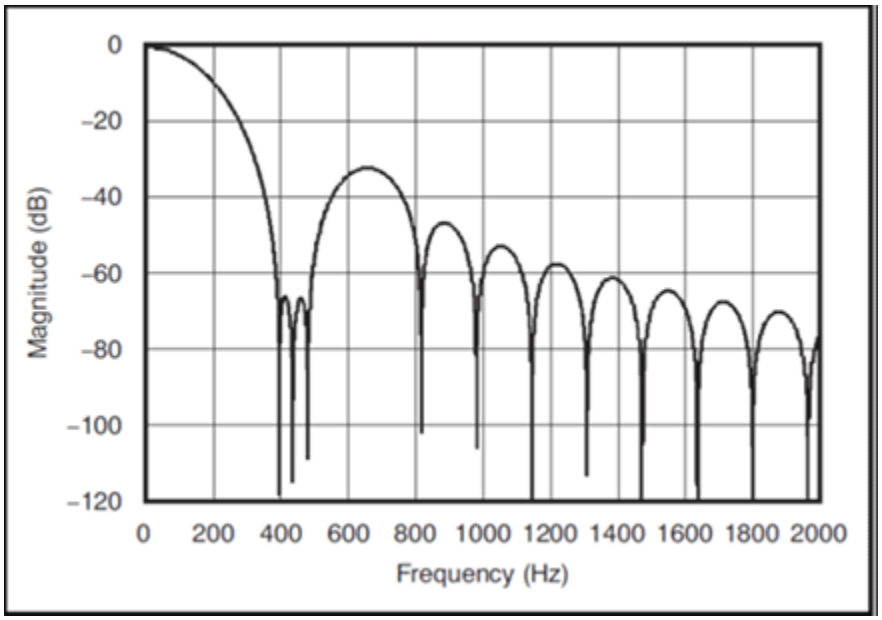
- Minimize time skew between successive measurements (see "Avoiding Time Skew" on page 71 for more information).
- Maximize throughput rate.
- Maximize life of the datalogger power supply.
- Minimize polarization of polar sensors such as those for measuring conductivity, soil moisture, or leaf wetness. Polarization may cause measurement errors or sensor degradation.
- Improve accuracy of an LVDT measurement. The induced voltage in an LVDT decays with time as current in the primary coil shifts from the inductor to the series resistance; a long integration may result in most of signal decaying before the measurement is complete.

Consult the following technical paper at www.campbellsci.com/app-notes for in-depth treatment of measurement noise: Preventing and Attacking Measurement Noise Problems.

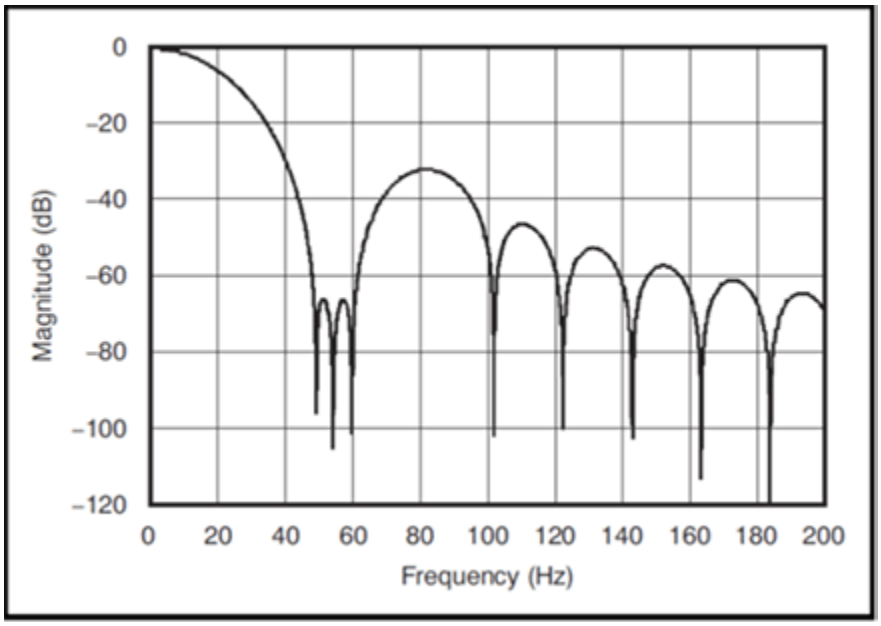
fN1 set to 4000 Hz:



fN1 set to 400 Hz:



fN1 set to 50 or 60 Hz:



MINIMIZING SETTLING ERRORS

Settling time allows an analog voltage signal to rise or fall closer to its true magnitude prior to measurement. Default settling times (those resulting when *SettlingTime* = 0) provide sufficient settling in most cases. Additional settling time is often programmed when measuring high-resistance (high-impedance) sensors or when sensors connect to the input terminals by long cables. The time to complete a measurement increases with increasing settling time. For example, a 1 ms increase in settling time for a bridge instruction with input reversal and excitation reversal results in a 4 ms increase in time to perform the instruction.

When sensors require long lead lengths, use the following general practices to minimize settling errors:

- Do not use wire with PVC-insulated conductors. PVC has a high dielectric constant, which extends input settling time.
- Where possible, run excitation leads and signal leads in separate shields to minimize transients.
- When measurement speed is not a prime consideration, additional time can be used to ensure ample settling time.
- In difficult cases where measurement speed is a consideration, an appropriate settling time can be determined through testing.

Measuring Settling Time

Settling time for a particular sensor and cable can be measured with the CR300 series. Programming a series of measurements with increasing settling times will yield data that indicate at what settling time a further increase results in negligible change in the measured voltage. The programmed settling time at this point indicates the settling time needed for the sensor / cable combination.

The following **CRBasic Example: Measuring Settling Time** presents CRBasic code to help determine settling time for a pressure transducer using a high-capacitance semiconductor. The code consists of a series of full-bridge measurements (`BrFull()`) with increasing settling times. The pressure transducer is placed in steady-state conditions so changes in measured voltage are attributable to settling time rather than changes in pressure.

CRBasic Example: Measuring Settling Time

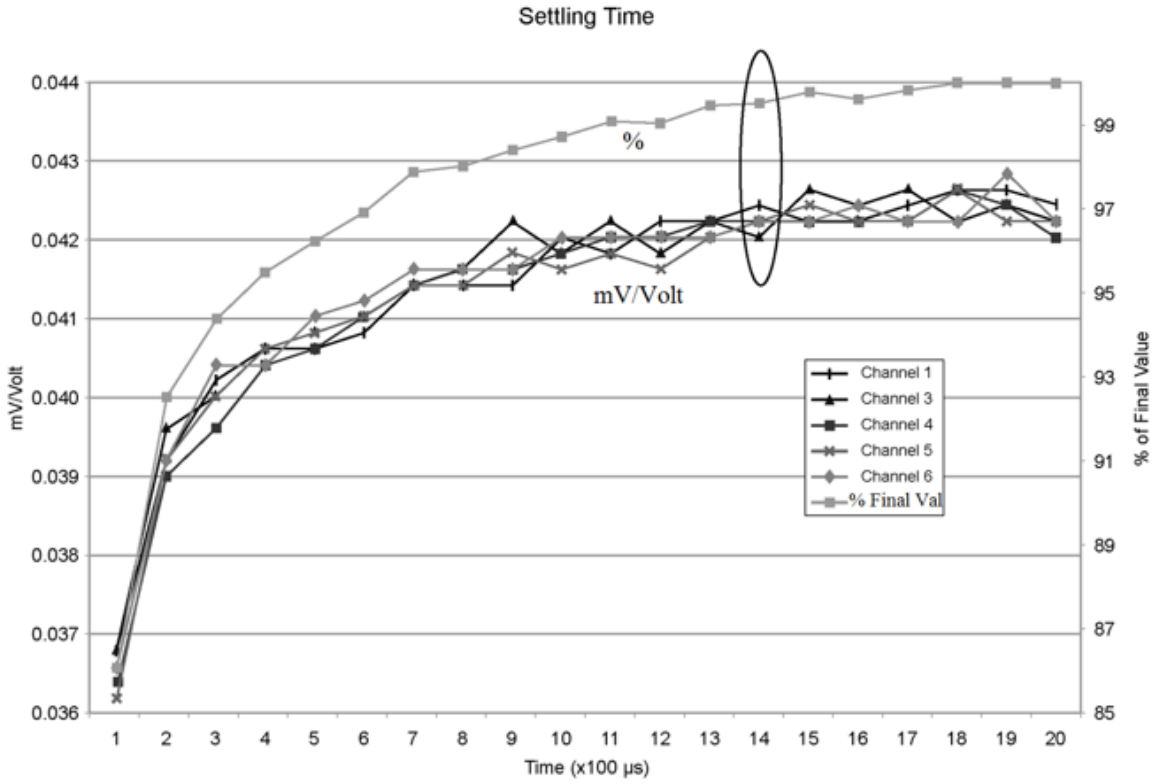
```
'This program example demonstrates the measurement of settling time
'using a single measurement instruction multiple times in succession.

Public PT(20)   'Variable to hold the measurements

DataTable(Settle,True,100)
  Sample(20,PT(),IEEE4)
EndTable

BeginProg
  Scan(1,Sec,3,0)

    BrFull(PT(1),1,mV2500,1,Vx1,1,2500,False,True ,100,60,1.0,0)
    BrFull(PT(2),1,mV2500,1,Vx1,1,2500,False,True ,200,60,1.0,0)
    BrFull(PT(3),1,mV2500,1,Vx1,1,2500,False,True ,300,60,1.0,0)
    BrFull(PT(4),1,mV2500,1,Vx1,1,2500,False,True ,400,60,1.0,0)
    BrFull(PT(5),1,mV2500,1,Vx1,1,2500,False,True ,500,60,1.0,0)
    BrFull(PT(6),1,mV2500,1,Vx1,1,2500,False,True ,600,60,1.0,0)
    BrFull(PT(7),1,mV2500,1,Vx1,1,2500,False,True ,700,60,1.0,0)
    BrFull(PT(8),1,mV2500,1,Vx1,1,2500,False,True ,800,60,1.0,0)
    BrFull(PT(9),1,mV2500,1,Vx1,1,2500,False,True ,900,60,1.0,0)
    BrFull(PT(10),1,mV2500,1,Vx1,1,2500,False,True ,1000,60,1.0,0)
    BrFull(PT(11),1,mV2500,1,Vx1,1,2500,False,True ,1100,60,1.0,0)
    BrFull(PT(12),1,mV2500,1,Vx1,1,2500,False,True ,1200,60,1.0,0)
    BrFull(PT(13),1,mV2500,1,Vx1,1,2500,False,True ,1300,60,1.0,0)
    BrFull(PT(14),1,mV2500,1,Vx1,1,2500,False,True ,1400,60,1.0,0)
    BrFull(PT(15),1,mV2500,1,Vx1,1,2500,False,True ,1500,60,1.0,0)
    BrFull(PT(16),1,mV2500,1,Vx1,1,2500,False,True ,1600,60,1.0,0)
    BrFull(PT(17),1,mV2500,1,Vx1,1,2500,False,True ,1700,60,1.0,0)
    BrFull(PT(18),1,mV2500,1,Vx1,1,2500,False,True ,1800,60,1.0,0)
    BrFull(PT(19),1,mV2500,1,Vx1,1,2500,False,True ,1900,60,1.0,0)
    BrFull(PT(20),1,mV2500,1,Vx1,1,2500,False,True ,2000,60,1.0,0)

    CallTable Settle

  NextScan

EndProg
```

The first six measurements are shown in the following table:

Timestamp	Rec	PT(1) Smp	PT(2) Smp	PT(3) Smp	PT(4) Smp	PT(5) Smp	PT(6) Smp
8/3/2017 23:34	0	0.03638599	0.03901386	0.04022673	0.04042887	0.04103531	0.04123745
8/3/2017 23:34	1	0.03658813	0.03921601	0.04002459	0.04042887	0.04103531	0.0414396
8/3/2017 23:34	2	0.03638599	0.03941815	0.04002459	0.04063102	0.04042887	0.04123745
8/3/2017 23:34	3	0.03658813	0.03941815	0.03982244	0.04042887	0.04103531	0.04103531
8/3/2017 23:34	4	0.03679027	0.03921601	0.04022673	0.04063102	0.04063102	0.04083316

Each trace in the following image contains all twenty `PT()` mV/V values (left axis) for a given record number, along with an average value showing the measurements as percent of final reading (right axis). The reading has settled to 99.5% of the final value by the fourteenth measurement, which is contained

in variable `PT(14)`. This is suitable accuracy for the application, so a settling time of 1400 µs is determined to be adequate.



FACTORS AFFECTING ACCURACY

Accuracy describes the difference between a measurement and the true value. Many factors affect accuracy. This topic discusses the effect percent-of-reading, offset, and resolution have on the accuracy of the measurement of an analog voltage sensor signal. Accuracy is defined as follows:

accuracy = percent-of-reading + offset

where percents-of-reading and offsets are displayed in the "Analog Specifications" on page 89.

Note: Error discussed in this section and error-related specifications of the datalogger do not include error introduced by the sensor or by the transmission of the sensor signal to the datalogger.

Measurement Accuracy Example

The following example illustrates the effect percent-of-reading and offset have on measurement accuracy. The effect of offset is usually negligible on large signals.

Example:

- Sensor-signal voltage: ≈2500 mV
- CRBasic measurement instruction: **`VoltDiff()`**
- Programmed input-voltage range (***`Range`***): **mV2500** (±-100 to 2500 mV)
- Input measurement reversal (***`RevDiff`***): **True**
- Datalogger circuitry temperature: 10° C

Accuracy of the measurement is calculated as follows:

accuracy = percent-of-reading + offset

where

percent-of-reading = 2500 mV • ±0.04%

= ±1 mV

and

offset = ±20 µV

Therefore,

accuracy = ±(1 mV + 20 µV) = ±1.02 mV

MINIMIZING OFFSET VOLTAGES

Voltage offset can be the source of significant error. For example, an offset of 3 µV on a 2500 mV signal causes an error of only 0.00012%, but the same offset on a 0.25 mV signal causes an error of 1.2%. Measurement offset voltages are unavoidable, but can be minimized. Offset voltages originate with:

- Ground currents (see "Minimizing Ground Potential Differences" on page 76 for more information).
- Seebeck effect.
- Residual voltage from a previous measurement.

Remedies include:

- Connecting power grounds to power ground terminals (**G**).
- Automatic offset compensation for single-ended measurements when ***MeasOff = False***.
- Using ***MeasOff = True*** for better offset compensation.
- Programming longer settling times.

Single-ended measurements are susceptible to voltage drop at the ground terminal caused by return currents from another device that is powered from the datalogger wiring panel, such as another manufacturer's communication modem, or a sensor that requires a lot of power. Currents >5 mA are usually undesirable. The error can be avoided by routing power grounds from these other devices to a power ground **G** terminal, rather than using a signal ground (⏚) terminal. Ground currents can be caused by the excitation of resistive-bridge sensors, but these do not usually cause offset error. These currents typically only flow when a voltage excitation is applied. Return currents associated with voltage excitation cannot influence other single-ended measurements because the excitation is usually turned off before the datalogger moves to the next measurement. However, if the CRBasic program is written in such a way that an excitation terminal is enabled during an unrelated measurement of a small voltage, an offset error may occur.

The Seebeck effect results in small thermally induced voltages across junctions of dissimilar metals as are common in electronic devices. Differential measurements are more immune to these than are single-ended measurements because of passive voltage cancellation occurring between matched high and low pairs such as **1H/1L**. So, use differential measurements when measuring critical low-level voltages, especially those below 200 mV, such as are output from pyranometers and thermocouples.

When analog voltage signals are measured in series by a single measurement instruction, such as occurs when **VoltSE()** is programmed with ***Reps = 2*** or more, measurements on subsequent terminals may be affected by an offset, the magnitude of which is a function of the voltage from the previous measurement. While this offset is usually small and negligible when measuring large signals, significant error, or NAN, can occur when measuring very small signals. This effect is caused by dielectric absorption of the integrator capacitor and cannot be overcome by circuit design. Remedies include the following:

- Programing longer settling times.
- Using an individual instruction for each input terminal, the effect of which is to reset the integrator circuit prior to filtering.
- Avoiding preceding a very small voltage input with a very large voltage input in a measurement sequence if a single measurement instruction must be used.

Field Calibration

Calibration increases accuracy of a measurement device by adjusting its output, or the measurement of its output, to match independently verified quantities. Adjusting sensor output directly is preferred, but not always possible or practical. By adding the **FieldCal()** or **FieldCalStrain()** instruction to a CRBasic program, measurements of a linear sensor can be adjusted by modifying the programmed multiplier and offset applied to the measurement, without modifying or recompiling the CRBasic program. For more information, see the CRBasic help.

File System Error Codes

Errors can occur when attempting to access files on any of the available drives. All occurrences are rare. Often, formatting the drive will resolve the error. The errors display in the **File Control** messages box or in the **Status** table.

1 Invalid format
2 Device capabilities error
3 Unable to allocate memory for file operation
4 Max number of available files exceeded
5 No file entry exists in directory
6 Disk change occurred
7 Part of the path (subdirectory) was not found
8 File at EOF
9 Bad cluster encountered
10 No file buffer available
11 Filename too long or has bad chars
12 File in path is not a directory
13 Access permission, opening DIR or LABEL as file, or trying to open file as DIR or mkdir existing file
14 Opening read-only file for write
15 Disk full (can't allocate new cluster)
16 Root directory is full
17 Bad file ptr (pointer) or device not initialized
18 Device does not support this operation
19 Bad function argument supplied
20 Seek out-of-file bounds
21 Trying to mkdir an existing dir
22 Bad partition sector signature
23 Unexpected system ID byte in partition entry

24 Path already open
25 Access to uninitialized ram drive
26 Attempted rename across devices
27 Subdirectory is not empty
31 Attempted write to Write Protected disk
32 No response from drive (Door possibly open)
33 Address mark or sector not found
34 Bad sector encountered
35 DMA memory boundary crossing error
36 Miscellaneous I/O error
37 Pipe size of 0 requested
38 Memory-release error (relmem)
39 FAT sectors unreadable (all copies)
40 Bad BPB sector
41 Time-out waiting for filesystem available
42 Controller failure error
43 Pathname exceeds _MAX_PATHNAME

File Name and Resource Errors

The maximum size of the file name that can be stored, run as a program, or FTP transferred in the datalogger is 59 characters. If the name + file extension is longer than 59 characters, an **Invalid Filename** error is displayed. If several files are stored, each with a long file name, memory allocated to the root directory can be exceeded before the actual memory of storing files is exceeded. When this occurs, an **Insufficient resources or memory full** error is displayed.

Specifications

Electrical specifications are valid over a -40 to +70 °C, non-condensing environment, unless otherwise specified. Recalibration is recommended every three years. Critical specifications and system configuration should be confirmed with Campbell Scientific before purchase.

System Specifications

Processor: ARM Cortex M4 running at 144 MHz

Memory:

- CPU Drive: 80 MB serial flash
- Data Storage: 30 MB serial flash
- Operating System: 2 MB flash
- Settings, Calibration, TLS Certificates and Key, System Information: 3 MB serial flash
- Background Tasks, Buffers, System Memory, Program Memory, Table Memory, Program Variables: 756 KB RAM

> **Note**: CR300 series dataloggers with serial numbers 2812 and older have a 5 MB CPU drive and 10 MB serial flash storage. CR300 series dataloggers with serial numbers 2813 and newer, and all CR310 dataloggers have an 80 MB CPU drive and 30 MB serial flash storage.

Program Execution: 100 ms to 1 day

Real-Time Clock:

- Battery backed while external power is disconnected
- **Resolution**: 1 ms
- **Accuracy**: ±1 minute per month

Wiring Panel Temperature: A thermistor, located on the processing board, is measured when reporting wiring panel temperature.

Physical Specifications

Dimensions (additional clearance required for cables and leads):

- **CR300**: 13.97 x 7.62 x 4.56 cm (5.5 x 3.0 x 1.8 in). For CAD files, see CR300 Images and CAD 2D Drawings.
- **CR310**: 16.26 x 7.62 x 5.68 cm (6.4 x 3.0 x 2.2 in). For CAD files, see CR310 Images and CAD 2D Drawings.

Weight/Mass

- **CR300**: 242 g (0.53 lb)
- **CR310**: 288 g (0.64 lb)
- **CR300-WIFI/RF407/RF412/RF422**: 249.5 g (0.55 lb)
- **CR300-WIFI/RF407/RF412/RF422**: 306 g (0.68 lb)

Case Material: Powder-coated aluminum

Power Requirements

Power specifications for a communication option are shown within the specifications section for that option.

Protection: Power inputs are protected against surge, over-voltage, over-current, and reverse power.

Charger Input (**CHG+** and **CHG-** terminals):

- 16 – 32 Vdc
- Current limited to 0.9 A maximum
- Power converter or solar panel input

External Batteries (**BAT+** and **BAT-** terminals):

- 10 – 18 Vdc input
- 12 Vdc, lead-acid 7 Ah battery, typical

Internal Lithium Battery (see "Internal Battery" on page 62 for more information): 3 V coin cell CR2016 for battery-backed clock. 6-year life with no external power source.

Average Current Drain (assumes 12 Vdc on **BAT** terminals – add 2 mA if using **CHG** terminals):

- **Idle**: 1.5 mA
- **Active 1 Hz Scan with One Analog Measurement**: 5 mA
- **Serial** (RS-232): Active + 25 mA
- **Active** (Processor Always On): 23 mA
- **Ethernet Power Requirements** (CR310 Only):
  - **Ethernet Idle**: 32 mA
  - **Ethernet Link**: Active + 51 mA

USB Power: Functions that will be active with USB 5 Vdc include sending programs, adjusting datalogger settings, and making some measurements. If **USB** is the only power source, then the **VX1** – **VX2** range is reduced to 150 to 2500 mV, the **SW12V** terminal will not be operational, voltage output for the control ports (**C1**, **C2**) is limited to 4.75 V, and current output for the control ports (**C1**, **C2**) is limited to 8 mA.

Wi-Fi Additional Current Contribution at 12 Vdc:

- **Client mode communicating**: 70 mA typical
- **Client mode idle**: 7 mA typical
- **Access point mode communicating**: 65 mA
- **Access point mode idle**: 62 mA typical
- **Sleep** (sleep forced with a setting or CRBasic program): <0.1 mA

RF Average Additional Current Contribution at 12 Vdc

- Transmit
  - **RF407**, **RF412**, and **RF427**: < 80 mA
  - **RF422**: 20 mA
- Idle On
  - **RF407**, **RF412**, and **RF427**:12 mA
  - **RF422**: 9.5 mA
- Idle 0.5 sec Power Mode
  - **RF407**, **RF412**, and **RF427**:4 mA
  - **RF422**: 3.5 mA
- Idle 1 sec Power Mode
  - **RF407**, **RF412**, and **RF427**:3 mA
  - **RF422**: 2.5 mA
- Idle 4 sec Power Mode
  - **RF407**, **RF412**, and **RF427**:1.5 mA
  - **RF422**: 1.5 mA

See also "Power Output" on page 6.

Ground Specifications

Signal Ground (⏚) - reference for single-ended analog inputs, excitation returns, and a ground for sensor shield wires.

- 5 common terminals

Power Ground (G) - return for 12 V and digital sensors.

- 6 common terminals

Earth Ground Lug (⏚) - connection point for heavy-gauge earth-ground wire. 14 AWG wire, minimum, is recommended.

Power Output Specifications

VX: Two independently configurable voltage terminals (**VX1-VX2**). When providing voltage excitation, a single 12-bit DAC shared by all **VX** outputs produces a user-specified voltage during measurement only. In this case, these terminals are regularly used with resistive-bridge measurements (see "Resistance Measurements" on page 41 for more information). **VX** terminals can also be used to supply a switched, regulated 5 Vdc power source to power digital sensors and toggle control lines.

Note: CR300 series dataloggers are not capable of applying reverse excitation. Set the ***RevEx*** parameter of all bridge measurement instructions to **`False`**.

- **Range**: 150 to 5000 mV
- **Resolution**: 4.5 mV
- **Maximum Source or Sink Current**: 50 mA total, concurrently or independently.

See also "Voltage Measurements" on page 38.

SW12V: Provides unregulated 12 Vdc power with voltage equal to the Power Input supply voltage. **SW12V** is disabled when operating on USB power only. A thermal fuse regulates current sourcing.

- **Thermal Fuse Hold Current** (Overload causes voltage drop. Disconnect and let cool to reset. Operate at limit if the application can tolerate some fluctuation.):
  - 1200 mA @ -40 °C
  - 1100 mA @ 20 °C
  - 830 mA @ 70 °C

See also "Power Output" on page 6.

Digital Output: Up to seven terminals may be configured for digital output. See also "Digital Input/Output Specifications" on page 93.

- **SE1-SE4, P_SW**: 3.3 V Logic Level Drive Capacity = 100 µA at 3.0 V
- **C1-C2**: 3.3 V Logic Level Drive Capacity = 10 mA at 3.5 V

Analog Specifications

6 single-ended (SE) or 3 differential (DIFF) inputs individually configurable for voltage, thermocouple, current loop, ratiometric, and period average measurements, using a 24-bit ADC. One channel at a time is measured in numeric succession.

VOLTAGE MEASUREMENTS

Terminals:

- Differential Configuration: **DIFF 1H/1L - 3H/3L**
- Single-Ended Configuration: **SE 1 - 6**

Input Resistance:

- 5 GΩ typical (f_{N1} = 50/60 Hz)
- 300 MΩ typical (f_{N1} = 4000 Hz)

Input Limits: -100 to +2500 mV

Sustained Input Voltage without Damage:

- **SE 1-2**: -6 V, +9 V
- **SE 3-6**: ±17 V

DC Common-Mode Rejection:

- >120 dB with input reversal
- ≥90 dB without input reversal

Normal-Mode Rejection:

- >71 dB at 50 Hz
- >74 dB at 60 Hz

Input Current @ 25 °C:

- ±.08 nA typical (f_{N1} = 50/60 Hz)
- ±13 nA typical (f_{N1} = 4000 Hz)

Analog Range and Resolution:

		Differential with Input Reversal		Single-Ended and Differential without Input Reversal	
Notch Frequency (f_{N1}) (Hz)	Range[1] (mV)	RMS (µV)	Bits[2]	RMS (µV)	Bits[2]
4000	-100 to +2500 -34 to +34	23 3.0	16.8 14.5	33 4.2	16.3 14.0
400	-100 to +2500 -34 to +34	3.8 0.58	19.4 16.8	5.4 0.82	18.9 16.3
50/60[3]	-100 to +2500 -34 to +34	1.6 0.23	20.6 18.2	2.3 0.33	20.1 17.7

[1] Range overhead of ~10% on all ranges guarantees that full-scale values will not cause overrange.
[2] Typical effective resolution (ER) in bits; computed from ratio of full-scale range to RMS resolution.
[3] 50/60 correspond to rejection of 50 and 60 Hz ac power mains noise.

Accuracy (does not include sensor error or measurement noise):

- 0 to 40 °C: ±(0.04% of measurement+ Offset)
- -40 to 70 °C: ±(0.1% of measurement+ Offset)

Voltage Measurement Accuracy Offsets:

Range (mV)	Typical Offset (µV RMS)		
	Differential with Input Reversal	Differential without Input Reversal	Single-Ended
-100 to +2500	±20	±40	±60
-34 to +34	±6	±14	±20

Multiplexed Measurement Time: ((Multiplexed Measurement Time + Settling time) • reps +0.8 ms) (Default settling time of 500 µs. These are not maximum speeds. Multiplexed denotes circuitry inside the datalogger that switches signals into the ADC.)

Multiplexed Measurement Time with 500 µs Setting Time:

	Differential with Input Reversal	Single-Ended or Differential without Input Reversal
Example fN1[1] (Hz)	Time (ms)	Time (ms)
4000	2.9	1.4
400	14.6	7.3
50/60	103	51.5

[1]Notch frequency (1/integration time).

See also "Voltage Measurements" on page 38.

RESISTANCE MEASUREMENTS SPECIFICATIONS

The datalogger makes ratiometric-resistance measurements for four- and six-wire full bridge circuits and two-, three-, and four-wire half bridge circuits using voltage excitation. Typically, at least one terminal is configured for excitation output. Multiple sensors can usually use a common excitation terminal.

Terminals:

- **SE** terminals **1-6**
- **DIFF** terminals **1H/1L - 3H/3L**
- Excitation terminals **VX1-VX2**

Accuracy: Assumes input reversal for differential measurements (***RevDiff***). Does not include bridge resistor errors or sensor and measurement noise.

- **0 to 40 °C**: ±(0.05% of voltage measurement + offset)
- **-40 to 70 °C**: ±(0.06% of voltage measurement + offset)

See also "Resistance Measurements" on page 41.

PERIOD AVERAGING SPECIFICATIONS

Up to four analog inputs may be configured for period averaging.

Terminals: **SE** terminals **1-4**

Accuracy: ±(0.01% of reading + resolution), where resolution is (0.13 µs / number of cycles to be measured)

Frequency Range: 5 Hz to 200 kHz

Voltage Range: 0 to 3.3 V

Minimum Pulse Width: 2.5 µs

Voltage Threshold: Counts cycles on transition from <0.9 Vdc to >2.1 Vdc

See also "Period-Averaging Measurements" on page 46.

CURRENT-LOOP MEASUREMENTS SPECIFICATIONS

Two independent analog inputs terminals may be configured as independent, non-isolated 0-20 mA or 4-to-20 mA current-loop inputs. One channel at a time is measured in numeric succession. Current is measured using a 24-bit ADC.

Terminals: **SE** terminals **1-2**

Range: 0 to 25 mA

Accuracy

- **0 to 40 °C**: ±0.14% of reading
- **-40 to 70 °C**: ±0.26% of reading

See also "Current-Loop Measurements" on page 39.

Pulse Counting Specifications

Terminals are individually configurable for switch closure, high-frequency pulse, or low-level AC measurements. See also "Pulse-Width Modulation Specifications" on page 93.

Maximum Input Voltage: ±20 Vdc

SWITCH-CLOSURE INPUT

Terminals:

- **P_SW**
- **C1-C2** (Requires an external 100 kΩ resistor connected from the terminal to **BAT+**.)

Minimum Switch Closed Time: 3 ms

Minimum Switch Open Time: 3 ms

Maximum Bounce Time: 1 ms open without being counted

Maximum Input Frequency: 150 Hz

Maximum Input Voltage: ±17 Vdc

HIGH-FREQUENCY INPUT

Terminals:

- **SE** terminals **1-4**
- **P_LL**
- **P_SW**
- **C1-C2**

Maximum Input Frequency:

- **SE 1-4**: 35 kHz
- **P_LL**: 20 kHz
- **P_SW**: 35 kHz
- **C1-C2**: 3 kHz

LOW-LEVEL AC INPUT

Terminal: **P_LL**

Dc-offset Rejection: Internal ac coupling eliminates dc-offset voltages up to ±0.05 Vdc

Input Hysteresis: 12 mV at 1 Hz

Low-Level Ac Pulse Input Ranges:

- Sine Wave Input 20 mv RMS, Input Frequency Range 1.0 to 20 Hz
- Sine Wave Input 200 mv RMS, Input Frequency Range 0.5 to 200 Hz
- Sine Wave Input 2000 mv RMS, Input Frequency Range 0.3 to 10,000 Hz
- Sine Wave Input 5000 mv RMS, Input Frequency Range 0.3 to 20,000 Hz

QUADRATURE OUTPUT

Terminals: **SE1** and **SE2** or **C1** and **C2** can be configured as digital port pairs to monitor the two sensing channels of an encoder.

Maximum Frequency: 2.5 kHz

See "Digital Input/Output Specifications" on page 93 and "Pulse Measurements" on page 46 for additional information.

Digital Input/Output Specifications

Up to seven terminals may be configured for digital input or output (I/O).

Terminals:

- **SE** terminals **1-4**
- **P_SW**
- **C1-C2**

Digital I/O Voltage Levels:

Terminal	High State	Low State	Current Source	Maximum Input Voltage
C1 **C2**	5.0 V output 3.3V input	0V	10 mA at 3.5 V	-10 V, +15 V
SE1 **SE2**	3.3 V	0 V	100 µA at 3.0 V	-6 V, +9 V
SE3 **SE4** **P_SW**	3.3 V	0 V	100 µA at 3.0 V	±17 V

See also "Power Output" on page 6 and "Pulse Counting Specifications" on page 92.

PULSE-WIDTH MODULATION SPECIFICATIONS

Terminals: **SE** terminals **1-4**

Period Maximum: 2047 ms

Resolution

- 0 - 5 ms: 83.33 ns or 12 MHz
- 5 - 325 ms: 5.00 µs or 200 kHz
- > 325 ms: 31.25 µs or 32 kHz

See "Digital Input/Output Specifications" on page 93 for more information.

Communications Specifications

A datalogger is normally part of a two-way conversation started by a computer. In applications with some types of interfaces, the datalogger can also initiate the call (callback) when needed. In satellite applications, the datalogger may simply send bursts of data at programmed times without waiting for a response.

Ethernet Port (CR310 Only): RJ45/ jack, 10/100Base Mbps, full and half duplex, Auto-MDIX, magnetic isolation, and TVS surge protection. See also "Ethernet Communications" on page 13.

Internet Protocols: Ethernet, PPP, CS I/O IP, RNDIS, ICMP/Ping, Auto-IP(APIPA), IPv4, IPv6, UDP, TCP, TLS, DNS, DHCP, SLAAC, NTP, Telnet, HTTP(S), FTP(S), SMTP/TLS, POP3/TLS

Additional Protocols: PakBus, PakBus Encryption, SDM, SDI-12, Modbus RTU / ASCII / TCP, DNP3, NMEA 0183, I2C, SPI, custom user definable over serial, UDP

Data File Formats: CSV, XML, JSON, binary, encrypted

USB: Micro-B device for computer connectivity

RS-232: Female RS-232, 9-pin interface, 1200 to 115.2 kbps

WI-FI OPTION SPECIFICATIONS

WLAN (Wi-Fi) (CR300-WIFI only)

Maximum Possible Over-the-Air Data Rates: <11 Mbps over 802.11b, <54 Mbps over 802.11g, <72 Mbps over 802.11n

Operating Frequency: 2.4 GHz, 20 MHz bandwidth

Antenna Connector: Reverse Polarity SMA (RPSMA)

Antenna (shipped with datalogger): Unity gain (0 dBd), 1/2 wave whip, omnidirectional. Features an articulating knuckle joint that can be oriented vertically or at right angles

Supported Technologies: 802.11 a/b/g/n, WPA/WPA2-Personal, WPA/WPA2-Enterprise Security, WEP

Client Mode: WPA/WPA2-Personal and Enterprise, WEP

Access Point Mode: WPA2-Personal

WiFi Average Additional Current Contribution @ 12 Vdc:

- **Client Mode**: 7 mA idle, 70 mA communicating
- **Access Point Mode**: 62 mA idle, 70 mA communicating
- **Sleep (using IPNetPower or DevConfig setting to disable)**: 4 mA

Receive Sensitivity: -97 dBm

CELLULAR OPTION SPECIFICATIONS

Cell Technology:

Option	Cellular Protocol	Market	Verizon	AT&T	T-Mobile	International (1a, 1b)
-CELL200	3G, 2G	International *				✓
-CELL205	4G LTE with automatic 3G fallback	North America		✓	✓	
-CELL210	4G LTE CAT-1	United States	✓			

**Confirm modem compliance for country/carrier where services are needed.*

Antenna Terminal: SMA

Average Additional Current Contribution at 12 Vdc:

- **Idle**: Connected to network, no data transfer.
  - -CELL200 minimum = 2 mA, average = 10 mA
  - -CELL205 minimum = 2 mA, average = 14 mA
  - -CELL210 minimum = 2 mA, average = 28 mA
- **Transfer/Receive**:
  - -CELL200 minimum = 20 mA. Average = 105 mA
  - -CELL205 minimum = 20 mA. Average = 75 mA
  - -CELL210 minimum = 20 mA. Average = 90 mA

SIM Slot: Industry standard 3FF micro-SIM

-CELL200 (International)

The -CELL200 option is not compatible with a Verizon cellular network.

Technology	Frequency Bands (MHz)	Maximum Data Rate Downlink	Maximum Data Rate Uplink
UMTS/HSPA+ (3G)	800, 850, 900, 1900, 2100	7.2 Mbps	5.7 Mbps
GSM/GPRS/EDGE (2G)	850, 900, 1800, 1900	236.8 Kbps	236.8 Kbps

-CELL205 (North America)

The -CELL205 option is not compatible with a Verizon cellular network.

Technology	Frequency Bands (MHz)	Maximum Data Rate Downlink	Maximum Data Rate Uplink
LTE CAT-1 (4G)	700, 850, 1700/2100 (AWS-1), 1900	10.2 Mbps	5.2 Mbps
UMTS/HSPA+ (3G)	850, 1700/2100 (AWS), 1900	7.2 Mbps	5.7 Mbps

-CELL210 (United States)

Technology	Frequency Bands (MHz)	Maximum Data Rate Downlink	Maximum Data Rate Uplink
LTE CAT-1 (4G)	700, 850, 1700, 1900, 2100	10.2 Mbps	5.2 Mbps

RF RADIO OPTION SPECIFICATIONS

Antenna Terminal: Reverse Polarity SMA (RPSMA)

Radio Type

- **RF407, RF412**, and **RF427**: Frequency Hopping Spread Spectrum (FHSS)
- **RF422**: SRD860 Radio with Listen before Talk (LBT) and Automatic Frequency Agility (AFA)

Frequency

- **RF407**: 902 to 928 MHz (US, Canada)
- **RF412**: 915 to 928 MHz (Australia, New Zealand)
- **RF422**: 863 to 870 MHz (European Union)
- **RF427**: 902 to 907.5 MHz/915 to 928 MHz (Brazil)

Transmit Power Output (software selectable)

- **RF407** and **RF412**: 5 to 250 mW
- **RF422**: 2 to 25 mW
- **RF427**: 5 to 250 mW

Channel Capacity

- **RF407**: Eight 25-channel hop sequences sharing 64 available channels.
- **RF412**: Eight 25-channel hop sequences sharing 31 available channels.
- **RF422**: Ten 30-channel hop sequences (default), software configurable to meet local regulations; 10 sequences for reducing interference through channel hop.
- **RF427**: Eight 25-channel hop sequences sharing 43 available channels.

Receive Sensitivity

- **RF407**, **RF412**, and **RF427**: -101 dBm
- **RF422**: -106 dBm

RF Data Rate

- **RF407**, **RF412**, and **RF427**: 200 kbps
- **RF422**: 10 kbps

For RF additional current contribution specifications, see "Power Requirements" on page 87.

See also "Radio Communications" on page 21.

Standards Compliance Specifications

View EU Declarations of Conformity at www.campbellsci.com/cr300 and www.campbellsci.com/cr310.

Shock and Vibration: ASTM D4169-09

Protection: IP30

RF407 Option

- United States FCC Part 15.247: MCQ-XB900HP
- Industry Canada (IC): 1846A-XB900HP
- Mexico IF: RCPDIXB15-0672-A1

RF412 Option

- ACMA RCM
- United States FCC Part 15.247:
- MCQ-XB900HP
- Industry Canada (IC): 1846A-XB900HP

RF422 Option: View the CR300 series RF422 radio option EU Declaration of Conformity at www.campbellsci.com/cr300 and www.campbellsci.com/cr310.

WIFI Option

- United States FCC ID: XF6-RS9113SB
- Industry Canada (IC): 8407A-RS9113SB
- View the CR300 series Wi-Fi radio option EU Declaration of Conformity at www.campbellsci.com/cr300 and www.campbellsci.com/cr310.

Note: The user is responsible for emissions if changing the antenna type or increasing the gain.

Glossary

A

ac
Alternating current (see Vac).

accuracy
The degree to which the result of a measurement, calculation, or specification conforms to the correct value or a standard.

ADC
Analog to digital conversion. The process that translates analog voltage levels to digital values.

allowed neighbor list
In PakBus networking, an allowed neighbor list is a list of neighbors with which a device will communicate. If a device address is entered in an allowed neighbor list, a hello exchange will be initiated with that device. Any device with an address between 1 and 3999 that is not entered in the allowed neighbor list will be filtered from communicating with the device using the list.

amperes (A)
Base unit for electric current. Used to quantify the capacity of a power source or the requirements of a power-consuming device.

analog
Data presented as continuously variable electrical signals.

APN
Cellular Access Point Name (obtained from your cellular network provider)

argument
Part of a procedure call (or command execution).

ASCII/ANSI
Abbreviation for American Standard Code for Information Interchange / American National Standards Institute. An encoding scheme in which numbers from 0-127 (ASCII) or 0-255 (ANSI) are used to represent pre-defined alphanumeric characters. Each number is usually stored and transmitted as 8 binary digits (8 bits), resulting in 1 byte of storage per character of text.

asynchronous
The transmission of data between a transmitting and a receiving device occurs as a series of zeros and ones. For the data to be "read" correctly, the receiving device must begin reading at the proper point in the series. In asynchronous communication, this coordination is accomplished by having each character surrounded by one or more start and stop bits which designate the beginning and ending points of the information. Also indicates the sending and receiving devices are not synchronized using a clock signal.

AWG
AWG ("gauge") is the accepted unit when identifying wire diameters. Larger AWG values indicate smaller cross-sectional diameter wires. Smaller AWG values indicate large-diameter wires. For example, a 14 AWG wire is often used for grounding because it can carry large currents. 22 AWG wire is often used as sensor leads since only small currents are carried when measurements are made.

B

baud rate
The rate at which data is transmitted.

beacon
A signal broadcasted to other devices in a PakBus network to identify "neighbor" devices. A beacon in a PakBus network ensures that all devices in the network are aware of other devices that are viable. If configured to do so, a clock-set command may be transmitted with the beacon. This function can be used to synchronize the clocks of devices within the PakBus network.

binary
Describes data represented by a series of zeros and ones. Also describes the state of a switch, either being on or off.

BOOL8
A one-byte data type that holds eight bits (0 or 1) of information. BOOL8 uses less space than the 32 bit BOOLEAN data type.

boolean
Name given a function, the result of which is either true or false.

boolean data type
Typically used for flags and to represent conditions or hardware that have only two states (true or false) such as flags and control ports.

burst
Refers to a burst of measurements. Analogous to a burst of light, a burst of measurements is intense, such that it features a series of measurements in rapid succession, and is not continuous.

C

calibration wizard
The calibration wizard facilitates the use of the CRBasic field calibration instructions FieldCal() and FieldCalStrain(). It is found in LoggerNet (4.0 or higher) or RTDAQ.

callback
A name given to the process by which the datalogger initiates communications with a computer running appropriate Campbell Scientific datalogger support software. Also known as "Initiate Comms."

CardConvert software
A utility to retrieve binary final-storage data from memory cards and convert the data to ASCII or other formats.

CD100
An optional enclosure mounted keyboard/display for use with dataloggers.

CDM/CPI
CPI is a proprietary interface for communications between Campbell Scientific dataloggers and Campbell Scientific CDM peripheral devices. It consists of a physical layer definition and a data protocol. CDM devices are similar to Campbell Scientific SDM devices in concept, but the use of the CPI bus enables higher data-throughput rates and use of longer cables. CDM devices require more power to operate in general than do SDM devices.

CF
CompactFlash®

code
A CRBasic program, or a portion of a program.

Collect button
Button or command in datalogger support software that facilitates collection-on-demand of final-data memory. This feature is found in PC200W, PC400, LoggerNet, and RTDAQ software.

Collect Now button
Button or command in datalogger support software that facilitates collection-on-demand of final-data memory. This feature is found in PC200W, PC400, LoggerNet, and RTDAQ software.

COM port
COM is a generic name given to physical and virtual serial communication ports.

COM1
When configured as a communication port, terminals C1 and C2 act as a pair to form Com1.

command
An instruction or signal that causes a computer to perform one of its basic functions (usually in CRBasic).

command line
One line in a CRBasic program. Maximum length, even with the line continuation characters <space> <underscore> (_), is 512 characters. A command line usually consists of one program statement, but it may consist of multiple program statements separated by a <colon> (:).

CompactFlash
CompactFlash® (CF) is a memory-card technology used in some Campbell Scientific card-storage modules.

compile
The software process of converting human-readable program code to binary machine code. Datalogger user programs are compiled internally by the datalogger operating system.

conditioned output
The output of a sensor after scaling factors are applied.

connector
A connector is a device that allows one or more electron conduits (wires, traces, leads, etc) to be connected or disconnected as a group. A connector consists of two parts – male and female. For example, a common household ac power receptacle is the female portion of a connector. The plug at the end of a lamp power cord is the male portion of the connector.

constant
A packet of memory given an alpha-numeric name and assigned a fixed number.

control I/O
C terminals configured for controlling or monitoring a device.

CoraScript
CoraScript is a command-line interpreter associated with LoggerNet datalogger support software.

CPU
Central processing unit. The brains of the datalogger.

cr
Carriage return.

CRBasic
Campbell Scientific's BASIC-like programming language that supports analog and digital measurements, data processing and analysis routines, hardware control, and many communication protocols.

CRBasic Editor
The CRBasic programming editor; supplied as part of LoggerNet, PC400, and RTDAQ software.

CRC
Cyclic Redundancy Check

CRD
An optional memory drive that resides on a memory card.

CS I/O
Campbell Scientific proprietary input/output port. Also, the proprietary serial communication protocol that occurs over the CS I/O port.

CVI

Communication verification interval. The interval at which a PakBus® device verifies the accessibility of neighbors in its neighbor list. If a neighbor does not communicate for a period of time equal to 2.5 times the CVI, the device will send up to four Hellos. If no response is received, the neighbor is removed from the neighbor list.

D

DAC

Digital to analog conversion. The process that translates digital voltage levels to analog values.

data bits

Number of bits used to describe the data and fit between the start and stop bit. Sensors typically use 7 or 8 data bits.

data cache

The data cache is a set of binary files kept on the hard disk of the computer running the datalogger support software. A binary file is created for each table in each datalogger. These files mimic the storage areas in datalogger memory, and by default are two times the size of the datalogger storage area. When the software collects data from a datalogger, the data are stored in the binary file for that datalogger. Various software functions retrieve data from the data cache instead of the datalogger directly. This allows the simultaneous sharing of data among software functions.

data output interval

The interval between each write of a record to a final-storage memory data table.

data output processing instructions

CRBasic instructions that process data values for eventual output to final-data memory. Examples of output-processing instructions include Totalize(), Maximize(), Minimize(), and Average(). Data sources for these instructions are values or strings in variable memory. The results of intermediate calculations are stored in data output processing memory to await the output trigger. The ultimate destination of data generated by data output processing instructions is usually final-storage memory, but the CRBasic program can be written to divert to variable memory by the CRBasic program for further processing. The transfer of processed summaries to final-data memory takes place when the Trigger argument in the DataTable() instruction is set to True.

data output processing memory

SRAM memory automatically allocated for intermediate calculations performed by CRBasic data output processing instructions. Data output processing memory cannot be monitored.

data point

A data value which is sent to final-storage memory as the result of a data-output processing instruction. Strings of data points output at the same time make up a record in a data table.

data table

A concept that describes how data are organized in memory, or in files that result from collecting data in memory. The fundamental data table is created by the CRBasic program as a result of the DataTable() instruction and resides in binary form in main-memory SRAM. The data table structure also resides in the data cache, in discrete data files on the data logger drives, and in binary or ASCII files that result from collecting final-storage memory with datalogger support software.

datalogger support software

LoggerNet, PC400, and PC200W - these Campbell Scientific software applications includes at least the following functions: datalogger communications, downloading programs, clock setting, and retrieval of measurement data.

DC

Direct current.

DCE
Data Communication Equipment. While the term has much wider meaning, in the limited context of practical use with the datalogger, it denotes the pin configuration, gender, and function of an RS-232 port. The RS-232 port on the datalogger is DCE. Interfacing a DCE device to a DCE device requires a null-modem cable.

desiccant
A hygroscopic material that absorbs water vapor from the surrounding air. When placed in a sealed enclosure, such as a datalogger enclosure, it prevents condensation.

Device Configuration Utility
Configuration tool used to set up dataloggers and peripherals, and to configure PakBus settings before those devices are deployed in the field and/or added to networks.

DHCP
Dynamic Host Configuration Protocol. A TCP/IP application protocol.

differential
A sensor or measurement terminal wherein the analog voltage signal is carried on two leads. The phenomenon measured is proportional to the difference in voltage between the two leads.

Dim
A CRBasic command for declaring and dimensioning variables. Variables declared with Dim remain hidden during datalogger operations.

dimension
To code a CRBasic program for a variable array as shown in the following examples: DIM example(3) creates the three variables example(1), example(2), and example(3); DIM example(3,3) creates nine variables; DIM example(3,3,3) creates 27 variables.

DNP3
Distributed Network Protocol is a set of communications protocols used between components in process automation systems. Its main use is in utilities such as electric and water companies.

DNS
Domain name server. A TCP/IP application protocol.

DTE
Data Terminal Equipment. While the term has much wider meaning, in the limited context of practical use with the datalogger, it denotes the pin configuration, gender, and function of an RS-232 port. The RS-232 port on the datalogger is DCE. Attachment of a null-modem cable to a DCE device effectively converts it to a DTE device.

duplex
A serial communication protocol. Serial communications can be simplex, half-duplex, or full-duplex.

duty cycle
The percentage of available time a feature is in an active state. For example, if the datalogger is programmed with 1 second scan interval, but the program completes after only 100 milliseconds, the program can be said to have a 10% duty cycle.

E

earth ground
A grounding rod or other suitable device that electrically ties a system or device to the earth. Earth ground is a sink for electrical transients and possibly damaging potentials, such as those produced by a nearby lightning strike. Earth ground is the preferred reference potential for analog voltage measurements. Note that most objects have a "an electrical potential" and the potential at different places on the earth - even a few meters away - may be different.

endian
The sequential order in which bytes are arranged into larger numerical values when stored in memory.

engineering units
Units that explicitly describe phenomena, as opposed to, for example, the datalogger base analog-measurement unit of millivolts.

ESD
Electrostatic discharge.

ESS
Environmental sensor station.

excitation
Application of a precise voltage, usually to a resistive bridge circuit.

execution interval
The time interval between initiating each execution of a given Scan() of a CRBasic program. If the Scan() Interval is evenly divisible into 24 hours (86,400 seconds), it is synchronized with the 24 hour clock, so that the program is executed at midnight and every Scan() Interval thereafter. The program is executed for the first time at the first occurrence of the Scan() Interval after compilation. If the Scan() Interval does not divide evenly into 24 hours, execution will start on the first even second after compilation.

execution time
Time required to execute an instruction or group of instructions. If the execution time of a program exceeds the Scan() Interval, the program is executed less frequently than programmed and the Status table SkippedScan field will increment.

expression
A series of words, operators, or numbers that produce a value or result.

F

FAT
File Allocation Table - a computer file system architecture and a family of industry-standard file systems utilizing it.

FFT
Fast Fourier Transform. A technique for analyzing frequency-spectrum data.

field
Final-storage data tables are made up of records and fields. Each row in a table represents a record and each column represents a field. The number of fields in a record is determined by the number and configuration of output processing instructions that are included as part of the DataTable() declaration.

File Control
File Control is a feature of LoggerNet, PC400, PC200W, Device Configuration Utility, and RTDAQ datalogger support software. It provides a view of the datalogger file system and a menu of file management commands.

fill and stop memory
A memory configuration for data tables forcing a data table to stop accepting data when full.

final-storage data
Data that resides in final-data memory.

final-storage memory
The portion of SRAM memory allocated for storing data tables with output arrays. Once data are written to final-data memory, they cannot be changed but only overwritten when they become the oldest data. Final-data memory is configured as ring memory by default, with new data overwriting the oldest data.

Flash
A type of memory media that does not require battery backup. Flash memory, however, has a lifetime based on the number of writes to it. The more frequently data are written, the shorter the life expectancy.

FLOAT
Four-byte floating-point data type. Default datalogger data type for Public or Dim variables. Same format as IEEE4.

FP2
Two-byte floating-point data type. Default datalogger data type for stored data. While IEEE four-byte floating point is used for variables and internal calculations, FP2 is adequate for most stored data. FP2 provides three or four significant digits of resolution, and requires half the memory as IEEE4.

frequency domain
Frequency domain describes data graphed on an X-Y plot with frequency as the X axis. VSPECT vibrating wire data are in the frequency domain.

frequency response
Sample rate is how often an instrument reports a result at its output; frequency response is how well an instrument responds to fast fluctuations on its input. By way of example, sampling a large gage thermocouple at 1 kHz will give a high sample rate but does not ensure the measurement has a high frequency response. A fine-wire thermocouple, which changes output quickly with changes in temperature, is more likely to have a high frequency response.

FTP
File Transfer Protocol. A TCP/IP application protocol.

full-duplex
A serial communication protocol. Simultaneous bi-directional communications. Communications between a serial port and a computer is typically full duplex.

G

garbage
The refuse of the data communication world. When data are sent or received incorrectly (there are numerous reasons why this happens), a string of invalid, meaningless characters (garbage) often results. Two common causes are: 1) a baud-rate mismatch and 2) synchronous data being sent to an asynchronous device and vice versa.

global variable
A variable available for use throughout a CRBasic program. The term is usually used in connection with subroutines, differentiating global variables (those declared using Public or Dim) from local variables, which are declared in the Sub() and Function() instructions.

ground
Being or related to an electrical potential of 0 volts.

ground currents
Pulling power from the datalogger wiring panel, as is done when using some communication devices from other manufacturers, or a sensor that requires a lot of power, can cause voltage potential differences between points in datalogger circuitry that are supposed to be at ground or 0 Volts. This difference in potentials can cause errors when measuring single-ended analog voltages.

H

half-duplex
A serial communication protocol. Bi-directional, but not simultaneous, communications. SDI-12 is a half-duplex protocol.

handshake
The exchange of predetermined information between two devices to assure each that it is connected to the other. When not used as a clock line, the CLK/HS (pin 7) line in the datalogger CS I/O port is primarily used to detect the presence or absence of peripherals.

hello exchange
In a PakBus network, this is the process of verifying a node as a neighbor.

hertz
SI unit of frequency. Cycles or pulses per second.

HTML
Hypertext Markup Language. Programming language used for the creation of web pages.

HTTP
Hypertext Transfer Protocol. A TCP/IP application protocol.

hysteresis
The dependence of the state of the system on its history.

Hz
SI unit of frequency. Cycles or pulses per second.

I

I2C
Inter-Integrated Circuit is a multi-master, multi-slave, packet switched, single-ended, serial computer bus.

IEEE4
Four-byte, floating-point data type. IEEE Standard 754. Same format as Float.

Include file
A file containing CRBasic code to be included at the end of the current CRBasic program, or it can be run as the default program.

INF
A data word indicating the result of a function is infinite or undefined.

initiate comms
A name given to a processes by which the datalogger initiates communications with a computer running LoggerNet. Also known as Callback.

input/output instructions
Used to initiate measurements and store the results in input storage or to set or read control/logic ports.

instruction
Usually refers to a CRBasic command.

integer
A number written without a fractional or decimal component. 15 and 7956 are integers; 1.5 and 79.56 are not.

intermediate memory
SRAM memory automatically allocated for intermediate calculations performed by CRBasic data output processing instructions. Data output processing memory cannot be monitored.

IP
Internet Protocol. A TCP/IP internet protocol.

IP address
A unique address for a device on the internet.

IP trace
Function associated with IP data transmissions. IP trace information was originally accessed through the CRBasic instruction IPTrace() and stored in a string variable. Files Manager setting is now modified to allow for creation of a file on a datalogger memory drive, such as USR:, to store information in ring memory.

isolation

Hardwire communication devices and cables can serve as alternate paths to earth ground and entry points into the datalogger for electromagnetic noise. Alternate paths to ground and electromagnetic noise can cause measurement errors. Using opto-couplers in a connecting device allows communication signals to pass, but breaks alternate ground paths and may filter some electromagnetic noise. Campbell Scientific offers optically isolated RS-232 to CS I/O interfaces as an accessory for use on the CS I/O port.

J

JSON

Java Script Object Notation. A data file format available through the datalogger or LoggerNet.

K

keep memory

keep memory is non-volatile memory that preserves some settings during a power-up or program start up reset. Examples include PakBus address, station name, beacon intervals, neighbor lists, routing table, and communication timeouts.

keyboard/display

The datalogger has an optional external keyboard/display.

L

leaf node

A PakBus node at the end of a branch. When in this mode, the datalogger is not able to forward packets from one of its communication ports to another. It will not maintain a list of neighbors, but it still communicates with other PakBus dataloggers and wireless sensors. It cannot be used as a means of reaching (routing to) other dataloggers.

lf

Line feed. Often associated with carriage return (<cr>). <cr><lf>.

linearity

The quality of delivering identical sensitivity throughout the measurement.

local variable

A variable available for use only by the subroutine in which it is declared. The term differentiates local variables, which are declared in the Sub() and Function() instructions, from global variables, which are declared using Public or Dim.

LoggerLink

Mobile applications that allow a mobile device to communicate with IP, wi-fi, or Bluetooth enabled dataloggers.

LoggerNet

Campbell Scientific's datalogger support software for programming, communications, and data retrieval between dataloggers and a computer.

LONG

Data type used when declaring integers.

loop

A series of instructions in a CRBasic program that are repeated for a programmed number of times. The loop ends with an End instruction.

loop counter

Increments by one with each pass through a loop.

LSB
Least significant bit (the trailing bit).

LVDT
The linear variable differential transformer (LVDT) is a type of electrical transformer used for measuring linear displacement (position).

M

mains power
The national power grid.

manually initiated
Initiated by the user, usually with a Keyboard/Display, as opposed to occurring under program control.

mass storage device
A mass storage device may also be referred to as an auxiliary storage device. The term is commonly used to describe USB mass storage devices.

MD5 digest
16 byte checksum of the TCP/IP VTP configuration.

micro SD
A removable memory-card technology used in CR6 and CR1000X dataloggers.

milli
The SI prefix denoting 1/1000 of a base SI unit.

Modbus
Communication protocol published by Modicon in 1979 for use in programmable logic controllers (PLCs).

modem/terminal
Any device that has the following: ability to raise the ring line or be used with an optically isolated interface to raise the ring line and put the datalogger in the communication command state, or an asynchronous serial communication port that can be configured to communicate with the datalogger.

modulo divide
A math operation. Result equals the remainder after a division.

MSB
Most significant bit (the leading bit).

multimeter
An inexpensive and readily available device useful in troubleshooting data acquisition system faults.

multiplier
A term, often a parameter in a CRBasic measurement instruction, that designates the slope (aka, scaling factor or gain) in a linear function. For example, when converting °C to °F, the equation is °F = °C*1.8 + 32. The factor 1.8 is the multiplier.

mV
The SI abbreviation for millivolts.

N

NAN
Not a number. A data word indicating a measurement or processing error. Voltage overrange, SDI-12 sensor error, and undefined mathematical results can produce NAN.

neighbor device
Device in a PakBus network that communicates directly with a device without being routed through an intermediate device.

Network Planner
Campbell Scientific software designed to help set up dataloggers in PakBus networks so that they can communicate with each other and the LoggerNet server. For more information, see https://www.campbellsci.com/loggernet.

NIST
National Institute of Standards and Technology.

node
Devices in a network – usually a PakBus network. The communications server dials through, or communicates with, a node. Nodes are organized as a hierarchy with all nodes accessed by the same device (parent node) entered as child nodes. A node can be both a parent and a child.

NSEC
Eight-byte data type divided up as four bytes of seconds since 1990 and four bytes of nanoseconds into the second.

null modem
A device, usually a multi-conductor cable, which converts an RS-232 port from DCE to DTE or from DTE to DCE.

Numeric Monitor
A digital monitor in datalogger support software or in a keyboard/display.

O

offset
A term, often a parameter in a CRBasic measurement instruction, that designates the y-intercept (aka, shifting factor or zeroing factor) in a linear function. For example, when converting °C to °F, the equation is °F = °C*1.8 + 32. The factor 32 is the offset.

ohm
The unit of resistance. Symbol is the Greek letter Omega (Ω). 1.0 Ω equals the ratio of 1.0 volt divided by 1.0 ampere.

Ohm's Law
Describes the relationship of current and resistance to voltage. Voltage equals the product of current and resistance (V = I • R).

on-line data transfer
Routine transfer of data to a peripheral left on-site. Transfer is controlled by the program entered in the datalogger.

operating system
The operating system (also known as "firmware") is a set of instructions that controls the basic functions of the datalogger and enables the use of user written CRBasic programs. The operating system is preloaded into the datalogger at the factory but can be re-loaded or upgraded by you using Device Configuration Utility software. The most recent datalogger operating system .obj file is available at www.campbellsci.com/downloads.

output
A loosely applied term. Denotes a) the information carrier generated by an electronic sensor, b) the transfer of data from variable memory to final-data memory, or c) the transfer of electric power from the datalogger or a peripheral to another device.

output array
A string of data values output to final-data memory. Output occurs when the data table output trigger is True.

output interval
The interval between each write of a record to a final-storage memory data table.

output processing instructions
CRBasic instructions that process data values for eventual output to final-data memory. Examples of output-processing instructions include Totalize(), Maximize(), Minimize(), and Average(). Data sources for these instructions are values or strings in variable memory. The results of intermediate calculations are stored in data output processing memory to await the output trigger. The ultimate destination of data generated by data output processing instructions is usually final-storage memory, but the CRBasic program can be written to divert to variable memory by the CRBasic program for further processing. The transfer of processed summaries to final-data memory takes place when the Trigger argument in the DataTable() instruction is set to True.

output processing memory
SRAM memory automatically allocated for intermediate calculations performed by CRBasic data output processing instructions. Data output processing memory cannot be monitored.

P

PakBus
® A proprietary communication protocol developed by Campbell Scientific to facilitate communications between Campbell Scientific devices. Similar in concept to IP (Internet Protocol), PakBus is a packet-switched network protocol with routing capabilities. A registered trademark of Campbell Scientific, Inc.

PakBus Graph
Software that shows the relationship of various nodes in a PakBus network and allows for monitoring and adjustment of some registers in each node.

parameter
Part of a procedure (or command) definition.

PC200W
Basic datalogger support software for direct connect. It supports a connection between computer and datalogger and includes Short Cut for creating datalogger programs. Tools for setting the datalogger clock, sending programs, monitoring sensors, and on-site viewing and collection of data are also included.

PC400
Datalogger support software that supports a variety of communication options, manual data collection, and data monitoring displays. Short Cut and CRBasic Editor are included for creating datalogger programs. PC400 does not support complex communication options, such as phone-to-RF, PakBus® routing, or scheduled data collection.

PDP
Packet Data Protocol

period average
A measurement technique using a high-frequency digital clock to measure time differences between signal transitions. Sensors commonly measured with period average include water-content reflectometers.

peripheral
Any device designed for use with the datalogger. A peripheral requires the datalogger to operate. Peripherals include measurement, control, and data retrieval and communication modules.

PGIA
Programmable Gain Input Amplifier

ping
A software utility that attempts to contact another device in a network.

pipeline mode
A CRBasic program execution mode wherein instructions are evaluated in groups of like instructions, with a set group prioritization.

PLC
Programmable Logic Controllers

Poisson ratio
A ratio used in strain measurements.

ppm
Parts per million.

precision
The amount of agreement between repeated measurements of the same quantity (AKA repeatability).

PreserveVariables
CRBasic instruction that protects Public variables from being erased when a program is recompiled.

print device
Any device capable of receiving output over pin 6 (the PE line) in a receive-only mode. Printers, "dumb" terminals, and computers in a terminal mode fall in this category.

print peripheral
Any device capable of receiving output over pin 6 (the PE line) in a receive-only mode. Printers, "dumb" terminals, and computers in a terminal mode fall in this category.

processing instructions
CRBasic instructions used to further process input-data values and return the result to a variable where it can be accessed for output processing. Arithmetic and transcendental functions are included.

program control instructions
Modify the execution sequence of CRBasic instructions. Also used to set or clear flags.

Program Send command
Program Send is a feature of datalogger support software.

program statement
A complete program command construct confined to one command line or to multiple command lines merged with the line continuation characters <space><underscore> (_). A command line, even with line continuation, cannot exceed 512 characters.

public
A CRBasic command for declaring and dimensioning variables. Variables declared with Public can be monitored during datalogger operation.

pulse
An electrical signal characterized by a rapid increase in voltage follow by a short plateau and a rapid voltage decrease.

R

ratiometric
Describes a type of measurement or a type of math. Ratiometric usually refers to an aspect of resistive-bridge measurements - either the measurement or the math used to process it. Measuring ratios and using ratio math eliminates several sources of error from the end result.

record
A record is a complete line of data in a data table or data file. All data in a record share a common time stamp. Final-storage data tables are made up of records and fields. Each row in a table represents a record and each column represents a field. The number of fields in a record is determined by the number and configuration of output processing instructions that are included as part of the DataTable() declaration.

regulator
A setting, a Status table element, or a DataTableInformation table element. Also a device for conditioning an electrical power source. Campbell Scientific regulators typically condition ac or dc voltages greater than 16 Vdc to about 14 Vdc.

resistance
A feature of an electronic circuit that impedes or redirects the flow of electrons through the circuit.

resistor
A device that provides a known quantity of resistance.

resolution
The smallest interval measurable.

ring line
Ring line is pulled high by an external device to notify the datalogger to commence RS-232 communications. Ring line is pin 3 of a DCE RS-232 port.

ring memory
A memory configuration that allows the oldest data to be overwritten with the newest data. This is the default setting for final-storage data tables.

ringing
Oscillation of sensor output (voltage or current) that occurs when sensor excitation causes parasitic capacitances and inductances to resonate.

RMS
Root-mean square, or quadratic mean. A measure of the magnitude of wave or other varying quantities around zero.

RNDIS
Remote Network Driver Interface Specification - a Microsoft protocol that provides a virtual Ethernet link via USB.

router
A device configured as a router is able to forward PakBus packets from one port to another. To perform its routing duties, a datalogger configured as a router maintains its own list of neighbors and sends this list to other routers in the PakBus network. It also obtains and receives neighbor lists from other routers. Routers maintain a routing table, which is a list of known nodes and routes. A router will only accept and forward packets that are destined for known devices. Routers pass their lists of known neighbors to other routers to build the network routing system.

RS-232
Recommended Standard 232. A loose standard defining how two computing devices can communicate with each other. The implementation of RS-232 in Campbell Scientific dataloggers to computer communications is quite rigid, but transparent to most users. Features in the datalogger that implement RS-232 communication with smart sensors are flexible.

RS-485
Recommended Standard 485. A standard defining how two computing devices can communicate with each other.

RTDAQ
Datalogger support software for industrial and real-time applications.

RTU
Remote Telemetry Units

Rx
Receive

S

sample rate
The rate at which measurements are made by the datalogger. The measurement sample rate is of interest when considering the effect of time skew, or how close in time are a series of measurements, or how close a time stamp on a measurement is to the true time the phenomenon being measured occurred. A 'maximum sample rate' is the rate at which a measurement can repeatedly be made by a single CRBasic instruction. Sample rate is how often an instrument reports a result at its output; frequency response is how well an instrument responds to fast fluctuations on its input. By way of example, sampling a large gage thermocouple at 1 kHz will give a high sample rate but does not ensure the measurement has a high frequency response. A fine-wire thermocouple, which changes output quickly with changes in temperature, is more likely to have a high frequency response.

scan interval
The time interval between initiating each execution of a given Scan() of a CRBasic program. If the Scan() Interval is evenly divisible into 24 hours (86,400 seconds), it is synchronized with the 24 hour clock, so that the program is executed at midnight and every Scan() Interval thereafter. The program is executed for the first time at the first occurrence of the Scan() Interval after compilation. If the Scan() Interval does not divide evenly into 24 hours, execution will start on the first even second after compilation.

scan time
When time functions are run inside the Scan() / NextScan construct, time stamps are based on when the scan was started according to the datalogger clock. Resolution of scan time is equal to the length of the scan.

SDI-12
Serial Data Interface at 1200 baud. Communication protocol for transferring data between the datalogger and SDI-12 compatible smart sensors.

SDK
Software Development Kit

SDM
Synchronous Device for Measurement. A processor-based peripheral device or sensor that communicates with the datalogger via hardwire over a short distance using a protocol proprietary to Campbell Scientific.

Seebeck effect
Induces microvolt level thermal electromotive forces (EMF) across junctions of dissimilar metals in the presence of temperature gradients. This is the principle behind thermocouple temperature measurement. It also causes small, correctable voltage offsets in datalogger measurement circuitry.

semaphore
(Measurement semaphore.) In sequential mode, when the main scan executes, it locks the resources associated with measurements. In other words, it acquires the measurement semaphore. This is at the scan level, so all subscans within the scan (whether they make measurements or not), will lock out measurements from slow sequences (including the auto self-calibration). Locking measurement resources at the scan level gives non-interrupted measurement execution of the main scan.

send button
Send button in datalogger support software. Sends a CRBasic program or operating system to a datalogger.

sequential mode
A CRBasic program execution mode wherein each statement is evaluated in the order it is listed in the program.

serial
A loose term denoting output of a series of ASCII, HEX, or binary characters or numbers in electronic form.

Settings Editor
An editor for observing and adjusting settings. Settings Editor is a feature of LoggerNet|Connect, PakBus Graph, and Device Configuration Utility.

Short Cut
A CRBasic programming wizard suitable for many datalogger applications. Knowledge of CRBasic is not required to use Short Cut.

SI
Système Internationale. The uniform international system of metric units. Specifies accepted units of measure.

signature
A number which is a function of the data and the sequence of data in memory. It is derived using an algorithm that assures a 99.998% probability that if either the data or the data sequence changes, the signature changes.

simplex
A serial communication protocol. One-direction data only. Serial communications between a serial sensor and the datalogger may be simplex.

single-ended
Denotes a sensor or measurement terminal wherein the analog voltage signal is carried on a single lead and measured with respect to ground (0 V).

skipped scans
Occur when the CRBasic program is too long for the scan interval. Skipped scans can cause errors in pulse measurements.

slow sequence
A usually slower secondary scan in the CRBasic program. The main scan has priority over a slow sequence.

SMS
Short message service. A text messaging service for web and mobile device systems.

SMTP
Simple Mail Transfer Protocol. A TCP/IP application protocol.

SNP
Snapshot file.

SP
Space.

SPI
Serial Peripheral Interface - a clocked synchronous interface, used for short distance communications, generally between embedded devices.

SRAM
Static Random-Access Memory

start bit
The bit used to indicate the beginning of data.

state
Whether a device is on or off.

Station Status command
A command available in most datalogger support software.

stop bit
The end of the data bits. The stop bit can be 1, 1.5, or 2.

string
A datum or variable consisting of alphanumeric characters.

support software
Campbell Scientific software that includes at least the following functions: datalogger communications, downloading programs, clock setting, and retrieval of measurement data.

synchronous

The transmission of data between a transmitting and a receiving device occurs as a series of zeros and ones. For the data to be "read" correctly, the receiving device must begin reading at the proper point in the series. In synchronous communication, this coordination is accomplished by synchronizing the transmitting and receiving devices to a common clock signal (see also asynchronous).

system time

When time functions are run outside the Scan() / NextScan construct, the time registered by the instruction will be based on the system clock, which has a 10 ms resolution.

T

τ

Time constant

table

Final-storage data tables are made up of records and fields. Each row in a table represents a record and each column represents a field. The number of fields in a record is determined by the number and configuration of output processing instructions that are included as part of the DataTable() declaration.

task

Grouping of CRBasic program instructions automatically by the datalogger compiler. Tasks include measurement, SDM or digital, and processing. Tasks are prioritized when the CRBasic program runs in pipeline mode. Also, a user-customized function defined through LoggerNet Task Master.

TCP/IP

Transmission Control Protocol / Internet Protocol.

TCR

Temperature Coefficient of Resistance. TCR tells how much the resistance of a resistor changes as the temperature of the resistor changes. The unit of TCR is ppm/°C (parts-per-million per degree Celsius). A positive TCR means that resistance increases as temperature increases. For example, a resistor with a specification of 10 ppm/°C will not increase in resistance by more than 0.000010 Ω per ohm over a 1 °C increase of the resistor temperature or by more than .00010 Ω per ohm over a 10 °C increase.

Telnet

A software utility that attempts to contact and interrogate another specific device in a network. Telnet is resident in Windows OS.

terminal

Point at which a wire (or wires) connects to a wiring panel or connector. Wires are usually secured in terminals by screw- or lever-and-spring actuated gates with small screw- or spring-loaded clamps.

terminal emulator

A command-line shell that facilitates the issuance of low-level commands to a datalogger or some other compatible device. A terminal emulator is available in most datalogger support software available from Campbell Scientific.

thermistor

A thermistor is a temperature measurement device with a resistive element that changes in resistance with temperature. The change is wide, stable, and well characterized. The output of a thermistor is usually non-linear, so measurement requires linearization by means of a Steinhart-Hart or polynomial equation. CRBasic instructions Therm107(), Therm108(), and Therm109() use Steinhart-Hart equations.

throughput rate

Rate that a measurement can be taken, scaled to engineering units, and the stored in a final-memory data table. The datalogger has the ability to scan sensors at a rate exceeding the throughput rate. The primary factor determining throughput rate is the processing programmed into the CRBasic program. In sequential-mode operation, all processing called for by an instruction must be completed before moving on to the next instruction.

time domain
Time domain describes data graphed on an X-Y plot with time on the X axis. Time series data are in the time domain.

TLS
Transport Layer Security. An Internet communication security protocol.

toggle
To reverse the current power state.

TTL
Transistor-to-Transistor Logic. A serial protocol using 0 Vdc and 5 Vdc as logic signal levels.

Tx
Transmit

U

UART
Universal Asynchronous Receiver/Transmitter for asynchronous serial communication.

UINT2
Data type used for efficient storage of totalized pulse counts, port status (status of 16 ports stored in one variable, for example) or integer values that store binary flags.

unconditioned output
The fundamental output of a sensor, or the output of a sensor before scaling factors are applied.

UPS
Uninterruptible Power Supply. A UPS can be constructed for most datalogger applications using ac line power, a solar panel, an ac/ac or ac/dc wall adapter, a charge controller, and a rechargeable battery.

URI
Uniform Resource Identifier

URL
Uniform Resource Locator

user program
The CRBasic program written by you in Short Cut program wizard.

USR drive
A portion of memory dedicated to the storage of image or other files.

V

Vac
Volts alternating current.

variable
A packet of SRAM given an alphanumeric name. Variables reside in variable memory.

Vdc
Volts direct current.

VisualWeather
Datalogger support software specialized for weather and agricultural applications. The software allows you to initialize the setup, interrogate the station, display data, and generate reports from one or more weather stations.

volt meter
An inexpensive and readily available device useful in troubleshooting data acquisition system faults.

voltage divider
A circuit of resistors that ratiometrically divides voltage. For example, a simple two-resistor voltage divider can be used to divide a voltage in half. So, when fed through the voltage divider, 1 mV becomes 500 µV, 10 mV becomes 5 mV, and so forth. Resistive-bridge circuits are voltage dividers.

volts
SI unit for electrical potential.

VSPECT®
® A registered trademark for Campbell Scientific's proprietary spectral-analysis, frequency domain, vibrating wire measurement technique.

W

watchdog timer
An error-checking system that examines the processor state, software timers, and program-related counters when the CRBasic program is running. The following will cause watchdog timer resets, which reset the processor and CRBasic program execution: processor bombed, processor neglecting standard system updates, counters are outside the limits, voltage surges, and voltage transients. When a reset occurs, a counter is incremented in the WatchdogTimer entry of the Status table. A low number (1 to 10) of watchdog timer resets is of concern, but normally indicates that the situation should just be monitored. A large number of errors (>10) accumulating over a short period indicates a hardware or software problem. Consult with a Campbell Scientific support engineer.

weather-tight
Describes an instrumentation enclosure impenetrable by common environmental conditions. During extraordinary weather events, however, seals on the enclosure may be breached.

web API
Application Programming Interface

wild card
A character or expression that substitutes for any other character or expression.

X

XML
Extensible markup language.

Index