CR510 DATALOGGER
OPERATOR'S MANUAL

REVISION: 2/03

WARRANTY AND ASSISTANCE

The **CR510 DATALOGGER** is warranted by CAMPBELL SCIENTIFIC, INC. to be free from defects in materials and workmanship under normal use and service for thirty-six (36) months from date of shipment unless specified otherwise. Batteries have no warranty. CAMPBELL SCIENTIFIC, INC.'s obligation under this warranty is limited to repairing or replacing (at CAMPBELL SCIENTIFIC, INC.'s option) defective products. The customer shall assume all costs of removing, reinstalling, and shipping defective products to CAMPBELL SCIENTIFIC, INC. CAMPBELL SCIENTIFIC, INC. will return such products by surface carrier prepaid. This warranty shall not apply to any CAMPBELL SCIENTIFIC, INC. products which have been subjected to modification, misuse, neglect, accidents of nature, or shipping damage. This warranty is in lieu of all other warranties, expressed or implied, including warranties of merchantability or fitness for a particular purpose. CAMPBELL SCIENTIFIC, INC. is not liable for special, indirect, incidental, or consequential damages.

Products may not be returned without prior authorization. To obtain a Returned Materials Authorization (RMA), contact CAMPBELL SCIENTIFIC, INC., phone (435) 753-2342. After an applications engineer determines the nature of the problem, an RMA number will be issued. Please write this number clearly on the outside of the shipping container. CAMPBELL SCIENTIFIC's shipping address is:

CAMPBELL SCIENTIFIC, INC.
RMA#_____
815 West 1800 North
Logan, Utah 84321-1784

CAMPBELL SCIENTIFIC, INC. does not accept collect calls.

Non-warranty products returned for repair should be accompanied by a purchase order to cover the repair.

CAMPBELL SCIENTIFIC, INC.

815 W. 1800 N.
Logan, UT 84321-1784
USA
Phone (435) 753-2342
FAX (435) 750-9540
www.campbellsci.com

Campbell Scientific Canada Corp.
11564 -149th Street
Edmonton, Alberta T5M 1W7
CANADA
Phone (780) 454-2505
FAX (780) 454-2655

Campbell Scientific Ltd.
Campbell Park
80 Hathern Road
Shepshed, Loughborough
LE12 9GX, U.K.
Phone +44 (0) 1509 601141
FAX +44 (0) 1509 601091

CR510 MEASUREMENT AND CONTROL MODULE
TABLE OF CONTENTS

2. INTERNAL DATA STORAGE

3. INSTRUCTION SET BASICS

DATA RETRIEVAL/COMMUNICATION

4. EXTERNAL STORAGE PERIPHERALS

5. TELECOMMUNICATIONS

6. 9-PIN SERIAL INPUT/OUTPUT

PROGRAM EXAMPLES

7. MEASUREMENT PROGRAMMING EXAMPLES

APPENDICES

FEATURES OF CR510

The CR510 is programmed in the same way as the CR500 and executes existing CR500 programs. The CR510 has a clock and memory backed by an internal battery. This keeps the time and data while the CR510 is not connected to external power.

GENERAL

POWER UP

When primary power is applied to the CR510, it tests the FLASH memory and loads the current program to RAM. After the program compiles successfully, the CR510 begins executing the program. If the ring line on the 9 pin connector is raised while the CR510 is testing memory, there will be a 128 second delay before compiling and running the program. This can be used to edit or change the program before it starts running. To raise the ring line, press any key on the CR10KD keyboard display or call the CR510 with the computer during the power up sequence (i.e., while "HELLO" is displayed on the CR10KD).

LITHIUM BATTERY

A lithium battery powers the clock and RAM when the primary 12 VDC is not connected. The clock is more accurate when connected to the primary 12 VDC power supply. The lithium battery has an expected life of four years of continuous use. That is, the primary 12 VDC can be disconnected for four years before the clock stops and data are lost. The voltage of the lithium battery is found in the 8th window of the ∗B mode. The voltage of a new battery is approximately 3 volts. The lithium battery must be replaced when its voltage falls below 2.4 VDC. (Section 14.11)

TWO FINAL STORAGE AREAS

Final Storage can be divided into two parts: Final Storage Area 1 and Final Storage Area 2. Final Storage Area 1 is the default storage area and the only one used if the operator does not specifically allocate memory to Area 2. Each Final Storage Area can be represented as ring memory, where the newest data writes over the oldest data.

INTERNAL FLASH PROGRAM STORAGE

Several programs can be stored in the CR510 Flash Memory and later recalled and run using the ∗D Mode. (Section 1.8)

LOW VOLTAGE INDICATOR

When primary power falls below 9.6 VDC, the CR510 stops executing its programs. The Low Voltage Counter (∗B window 9) is incremented by one each time the primary power drops below 9.6 VDC and E10 is displayed on the CR10KD. A double dash (--) in the 9th window of the ∗B mode indicates that the CR510 is currently in a low primary power mode. (Section 1.6)

CONTROL PORT COUNTERS AND INTERRUPTS

Control port 2 can be used to measure switch closures up to 40 Hz. Control port 2 can also be used to activate interrupt driven subroutine 98. (Sections 1.1.2, 9, Instruction 3)

TAPE

Cassette tape is not supported as a data retrieval method with the CR510.

NEW INSTRUCTIONS

P69 Wind Vector
P75 Histogram
P98 Send Character

SELECTED OPERATING DETAILS

1. **Storing Data** - Data are stored in Final Storage only by Output Processing Instructions and only when the Output Flag (Flag 0) is set. (Sections OV4.1.1 and 3.7.1)

2. **Storing Date and Time** - Date and time are stored with the data in Final Storage ONLY if the Real Time Instruction 77 is used. (Section 11)

3. **Data Transfer** - On-line data transfer from Final Storage to peripherals (printer, Storage Module, etc.) occurs only if enabled with Instruction 96 in the datalogger program. (Sections 4 and 12)

4. **Final Storage Resolution** - All Input Storage values are displayed (∗6 mode) as high resolution with a maximum value of 99999. However, the default resolution for data stored in Final Storage is low resolution, maximum value of 6999. Results exceeding 6999 are stored as 6999 unless Instruction 78 is used to store the values in Final Storage as high resolution values. (Sections 2.2.1 and 11)

5. **Floating Point Format** - The computations performed in the CR510 use floating point arithmetic. CSI's 4 byte floating point numbers contain a 23 bit binary mantissa and a 6 bit binary exponent. The largest and smallest numbers that can be stored and processed are 9×10^{18} and 1×10^{-19}, respectively. (Section 2.2.2)

6. **Erasing Final Storage** - Data in Final Storage can be erased without altering the program by using the ∗A Mode to repartition memory. (Section 1.5.2)

7. **ALL memory can be erased** and the CR510 completely reset by entering 98765 for the number of bytes allocated to Program Memory. (∗A Window 5, Section 1.5.2)

CAUTIONARY NOTES

1. Damage will occur to the analog input circuitry if voltages in excess of ±16 V are applied for a sustained period. Voltages in excess of ±5 V will cause errors and possible overranging on other analog input channels.

2. When using the CR510 with the PS12LA, remember that the sealed lead acid batteries are permanently damaged if deep discharged. The cells are rated at a 7 Ahr capacity but experience a slow discharge even in storage. It is advisable to maintain a continuous charge on the PS12LA battery pack, whether in operation or storage (Section 14).

3. When connecting power to the CR510, first connect the positive lead from the power source to the 12 V terminal. Then connect the negative lead to G. Connecting these leads in the reverse order makes it easier for the positive wire to accidentally touch a grounded component and short out the power supply (Section 14).

4. Voltages in excess of 5.6 volts applied to a control port can cause the CR510 to malfunction and damage the datalogger.

5. Voltage pulses can be counted by CR510 Pulse Counters configured for High Frequency Pulses. However, when the pulse is actually a low frequency signal (below about 10 Hz) AND the positive voltage excursion exceeds 5.6 VDC, the 5 VDC supply will start to rise, upsetting all analog measurements.

   Pulses whose positive voltage portion exceed 5.6 VDC with a duration longer than 100 milliseconds need external conditioning. See the description of the Pulse count instruction in Section 9 for details on the external conditioning.

6. The CR510 board is coated with a conformal coating to protect against excess humidity and corrosion. To protect the datalogger from corrosion, additional desiccant must be placed inside the enclosure. To reduce vapor transfer into the enclosure, plug the cable entry conduit with Duct Seal, a putty-type sealant shipped with Campbell Scientific enclosures and available at most electrical supply houses. DO NOT totally seal enclosures equipped with lead acid batteries. Hydrogen concentration may build up to explosive levels.

CR510 DATALOGGER OVERVIEW

The CR510 is a fully programmable datalogger/controller with non-volatile memory and a battery backed clock in a small, rugged module. The combination of reliability, versatility, and telecommunications support make it a favorite choice for networks and single logger applications.

Campbell Scientific Inc. provides four aids to operating the CR510:

1. *This Overview*
2. *The CR510 Operator's Manual*
3. *The CR510 Prompt Sheet*
4. *Short Cut*

This Overview introduces the concepts required to take advantage of the CR510's capabilities. Hands-on programming examples start in Section OV5. Working with a CR510 will help the learning process, so don't just read the examples, do them. If you want to start this minute, go ahead and try the examples, then come back and read the rest of the Overview.

The sections of the Operator's Manual which should be read to complete a basic understanding of the CR510 operation are the Programming Sections 1-3, the portions of the data retrieval Sections 4 and 5 appropriate to the method(s) you are using (see OV6), and Section 14 which covers installation and maintenance.

Section 6 covers details of serial communications. Sections 7 and 8 contain programming examples. Sections 9-12 have detailed descriptions of each programming instruction, and Section 13 goes into detail on the CR510 measurement procedures.

The Prompt Sheet is an abbreviated description of the programming instructions. Once familiar with the CR510, it is possible to program it using only the Prompt Sheet as a reference, consulting the manual if further detail is needed.

Short Cut is an easy-to-use DOS-based software program. It features point-and-click menus to guide you through the process of creating simple CR510 programs. In addition to the downloadable program file, Short Cut creates a table to simplify wiring sensors to the CR510.

Read the Selected Operating Details and Cautionary Notes at the front of the Manual before using the CR510.

OV1. PHYSICAL DESCRIPTION

The CR510 was designed to provide a rugged datalogger with a low per unit cost. Some of its distinguishing physical features are:

- The CR510 does not have an integral keyboard/display. The user accesses the CR510 with the portable CR10KD Keyboard Display or with a computer or terminal (Section OV2).
- The power supply is external to the CR510. This gives the user a wide range of options (Section 14) for powering the CR510.

OV1.1 ANALOG INPUTS

The terminals labeled 1H to 4L are analog inputs. These numbers refer to the high and low inputs to the differential channels 1 and 2. In a differential measurement, the voltage on the H input is measured with respect to the voltage on the L input. When making single-ended measurements, either the H or L input may be used as an independent channel to measure voltage with respect to the CR510 analog ground (AG). The single-ended channels are numbered sequentially starting with 1H; e.g., the H and L sides of differential channel 1 are single-ended channels 1 and 2; the H and L sides of differential channel 2 are single-ended channels 3 and 4, etc.

OV1.2 EXCITATION OUTPUTS

The terminals labeled E1, and E2 are precision, switched excitation outputs used to supply programmable excitation voltages for resistive bridge measurements. DC or AC excitation voltages between -2500 mV and +2500 mV are user programmable (Section 9).

OV1.3 PULSE INPUTS

The terminals labeled P1, P2, and P3 are the pulse counter inputs for the CR510. P1 and P2 are programmable for high frequency pulse, low level AC, or switch closure (Section 9, Instruction 3). C2/P3 can be configured to count switch closures up to 40 Hz.

OV1.4 DIGITAL I/O PORTS

Terminal C1 is a digital Input/Output port. On power-up it is configured as an input port, commonly used for reading the status of an external signal. High and low conditions are: 3V < high < 5.5V; -0.5V < low < 0.8V.

Configured as output the port allows on/off control of external devices. A port can be set high (5V ± 0.1V), set low (<0.1V), toggled or pulsed (Sections 3, 8.3, and 12).

Port C2/P3 can be configured as pulse counters for switch closures (Section 9, Instruction 3) or used to trigger subroutine execution (Section 1.1.2), or serial SDI-12 communication.

OV1.5 ANALOG GROUND (AG)

The AG terminals are analog grounds, used as the reference for single-ended measurements and excitation return.

OV1.6 12V, POWER GROUND (G), AND EARTH TERMINALS

The 12V and power ground (G) terminals are used to supply 12V DC power to the datalogger. The extra 12V and G terminals can be used to connect other devices requiring 12V power.

The G terminals are also used to tie cable shields to ground, and to provide a ground reference for pulse counters and binary inputs. The G terminals are directly connected to the Earth terminal. For protection against transient voltage spikes, Earth Ground should be connected to a good earth ground (Section 14.7.1).

OV1.7 5V OUTPUT

The 5V (±0.2%) output is commonly used to power peripherals such as the QD1 Incremental Encoder Interface and AVW1 Vibrating Wire Interface.

The 5V output is common with pin 1 on the 9 pin serial connector; 200 mA is the maximum combined current output.

OV1.8 SERIAL I/O

The 9 pin serial I/O port contains lines for serial communication between the CR510 and external devices such as computers, printers, Storage Modules, etc. **This port does NOT have the same configuration as the 9 pin serial ports currently used on many personal computers**. It has a 5VDC power line which is used to power peripherals such as the Storage Modules. The same 5VDC supply is used for the 5V output on the terminal strip. It also has a continuous 12 V power supply on pin 8 for external communication devices such as the COM200 and COM300. Section 6 contains technical details on serial communication.

OV1.9 CONNECTING POWER TO THE CR510

The CR510 can be powered by any 12VDC source. The green power connector is a plug in connector that allows the power supply to be easily disconnected without unscrewing the terminals. The Terminal Strip power connection is reverse polarity protected. See Section 14 for details on power supply connections.

> **CAUTION**: The metal surfaces of the CR510 Terminal Strip, and CR10KD Keyboard Display are at the same potential as power ground. To avoid shorting 12 volts to ground, connect the 12 volt lead first, then connect the ground lead.

When primary power falls below 9.6 VDC, the CR510 stops executing its programs. The Low Voltage Counter (*B window 9) is incremented by one each time the primary power falls below 9.6 VDC and E10 is displayed on the CR10KD. A double dash (--) in the 9th window of the *B mode indicates that the CR510 is currently in a low primary power mode. (Section 1.6)

The datalogger program and stored data remain in memory, and the clock continues to keep

time when power is disconnected. The clock and Static Random Access Memory (SRAM) are powered by an internal lithium battery.

OV2. MEMORY AND PROGRAMMING CONCEPTS

OV2.1 INTERNAL MEMORY

The standard CR510 has 128 K of Flash Electrically Erasable Programmable Read Only Memory (EEPROM) and 128 K Static Random Access Memory (SRAM). The Flash EEPROM stores the operating system and user programs. RAM is used for data and running the program. Data Storage can be expanded with an optional Flash EEPROM (Figure OV2.1-1). The use of the Input, Intermediate, and Final Storage in the measurement and data processing sequence is shown in Figure OV2.1-2. The five areas of SRAM are:

1. **System Memory** - used for overhead tasks such as compiling programs, transferring data, etc. The user cannot access this memory.

2. **Program Memory** - available for user entered programs.

3. **Input Storage** - Input Storage holds the results of measurements or calculations. The ∗6 Mode is used to view Input Storage locations for checking current sensor readings or calculated values. Input Storage defaults to 28 locations. Additional locations can be assigned using the ∗A Mode.

4. **Intermediate Storage** - Certain Processing Instructions and most of the Output Processing Instructions maintain intermediate results in Intermediate Storage. Intermediate storage is automatically accessed by the instructions and cannot be accessed by the user. The default allocation is 64 locations. The number of locations can be changed using the ∗A Mode.

5. **Final Storage** - Final processed values are stored here for transfer to printer, solid state Storage Module or for retrieval via telecommunication links. Values are stored in Final Storage only by the Output Processing Instructions and only when the Output Flag is set in the user's program. Approximately 62,000 locations are allocated to Final Storage on power up. This number is reduced if Input or Intermediate Storage is increased.

While the total size of these areas remains constant, memory may be reallocated between the areas to accommodate different measurement and processing needs (∗A Mode, Section 1.5).

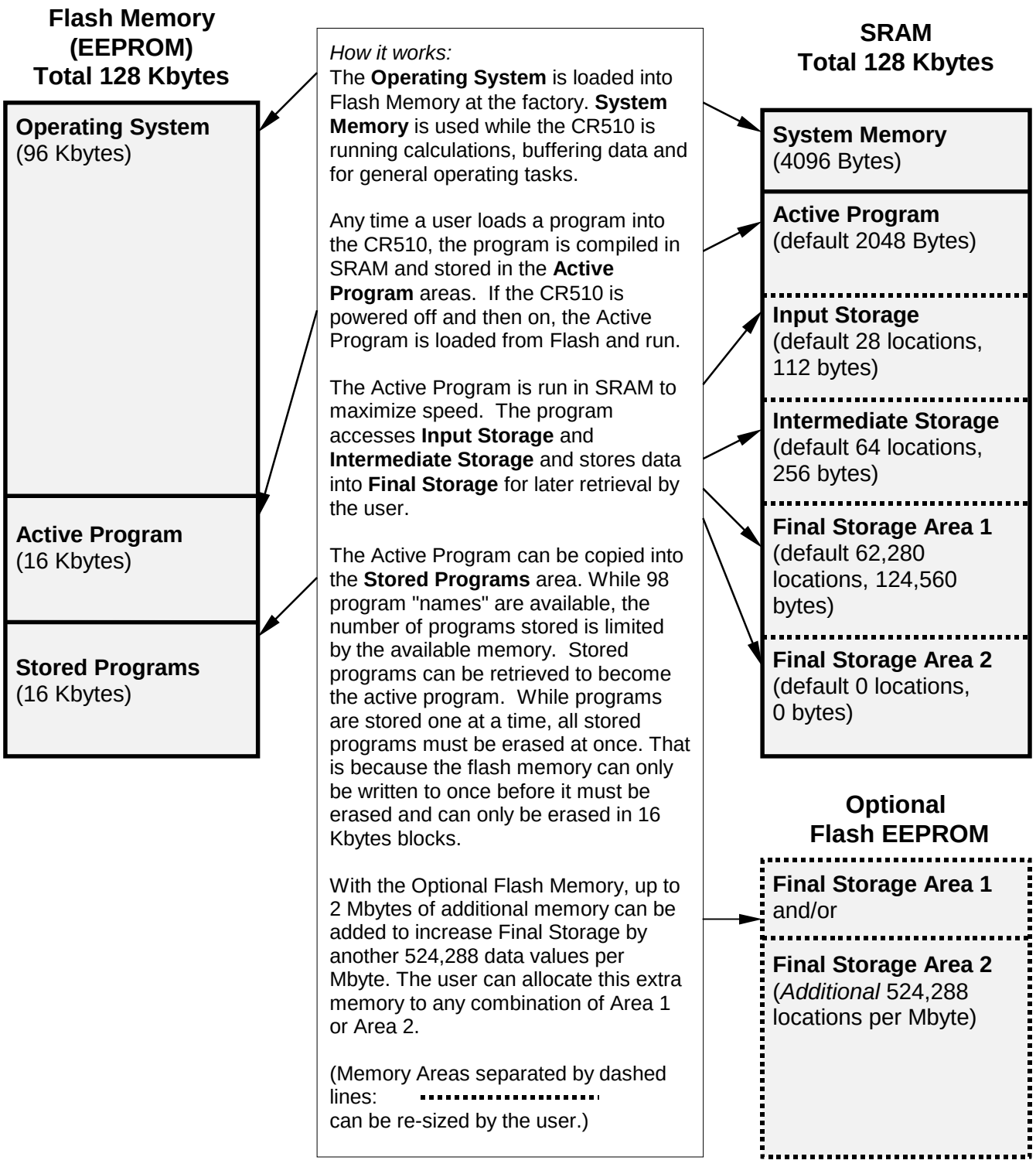


FIGURE OV2.1-1. CR510 Memory

OV2.2 PROGRAM TABLES, EXECUTION INTERVAL AND OUTPUT INTERVALS

The CR510 must be programmed before it will make any measurements. A program consists of a group of instructions entered into a **program table**. The program table is given an **execution interval** which determines how frequently that table is executed. When the table is executed, the instructions are executed in sequence from beginning to end. After executing the table, the CR510 waits the remainder of the execution interval and then executes the table again starting at the beginning.

The interval at which the table is executed generally determines the interval at which the sensors are measured. The interval at which data are stored is separate from how often the table is executed, and may range from samples every execution interval to processed summaries output hourly, daily, or on longer or irregular intervals.

Programs are entered in Tables 1 and 2. Subroutines, called from Tables 1 and 2, are entered in Subroutine Table 3. The size of program memory can be fixed or automatically allocated by the CR510 (Section 1.5).

Table 1 and Table 2 have independent execution intervals, entered in units of seconds with an allowable range of 1/8 to 8191 seconds. Subroutine Table 3 has no execution interval; subroutines are only executed when called from Table 1 or 2.

OV2.2.1 THE EXECUTION INTERVAL

The execution interval specifies how often the program in the table is executed, which is usually determined by how often the sensors are to be measured. *Unless two different measurement rates are needed, use only one table*. A program table is executed sequentially starting with the first instruction in the table and proceeding to the end of the table.

Table 1. Execute every x sec. $0.125 \leq x \leq 8191$
Instructions are executed sequentially in the order they are entered in the table. One complete pass through the table is made each execution interval unless program control instructions are used to loop or branch execution.
Normal Order: MEASURE PROCESS CHECK OUTPUT COND. OUTPUT PROCESSING

Table 2. Execute every y sec. $0.125 \leq y \leq 8191$
Table 2 is used if there is a need to measure and process data on a separate interval from that in Table 1.

Table 3. Subroutines
A subroutine is executed only when called from Table 1 or 2.
Subroutine Label Instructions End
Subroutine Label Instructions End
Subroutine Label Instructions End

FIGURE OV2.2-1. Program and Subroutine Tables

Each instruction in the table requires a finite time to execute. If the execution interval is less than the time required to process the table, an execution interval overrun occurs; the CR510 finishes processing the table and waits for the next execution interval before initiating the table. When an overrun occurs, decimal points are shown on either side of the G on the display in the LOG mode (*0). Overruns and table priority are discussed in Section 1.1.

OV2.2.2. THE OUTPUT INTERVAL

The interval at which output occurs must be an integer multiple of the execution interval (e.g., a table cannot have a 10 minute execution interval and output every 15 minutes).

A single program table can have many different output intervals and conditions, each with a unique data set (Output Array). Program Control Instructions are used to set the Output Flag. The Output Processing Instructions which follow the instruction setting the Output Flag determine the data output and its sequence. Each additional Output Array is created by another Program Control Instruction checking a output condition, followed by Output Processing Instructions defining the data set to output.

OV2.3 CR510 INSTRUCTION TYPES

Figure OV2.3-1 illustrates the use of three different instruction types which act on data. The fourth type, Program Control, is used to control output times and vary program execution. Instructions are identified by numbers.

1. **INPUT/OUTPUT INSTRUCTIONS** (1-12, 16-29, 105-106, 114, 117, 130, 131, Section 9) control the terminal strip inputs and outputs (Figure OV1.1-2), storing the results in Input Storage (destination). Multiplier and offset parameters allow conversion of linear signals into engineering units. The Digital I/O Ports are also addressed with I/O Instructions.

2. **PROCESSING INSTRUCTIONS** (30-68, Section 10) perform numerical operations on values located in Input Storage and store the results back in Input Storage. These instructions can be used to develop high level algorithms to process measurements prior to Output Processing.

3. **OUTPUT PROCESSING INSTRUCTIONS** (69-82, Section 11) are the only instructions which store data in Final Storage. Input Storage values are processed over time to obtain averages, maxima, minima, etc. There are two types of processing done by Output Instructions: **Intermediate** and **Final**.

   Intermediate processing normally takes place each time the instruction is executed. For example, when the Average Instruction is executed, it adds the values from the input locations being averaged to running totals in Intermediate Storage. It also keeps track of the number of samples.

   Final processing occurs only when the Output Flag is high (Section 3.7.1). The Output Processing Instructions check the Output Flag. If the flag is high, final values are calculated and output. With the Average, the totals are divided by the number of samples and the resulting averages sent to Final Storage. Intermediate locations are zeroed and the process starts over. *The Output Flag, Flag 0, is set high by a Program Control Instruction which must precede the Output Processing Instructions in the user entered program.*

4. **PROGRAM CONTROL INSTRUCTIONS** (83-98, 111, 120-121, Section 12) are used for logic decisions, conditional statements, and to send data to peripherals. They can set flags and ports, compare values or times, execute loops, call subroutines, conditionally execute portions of the program, etc.

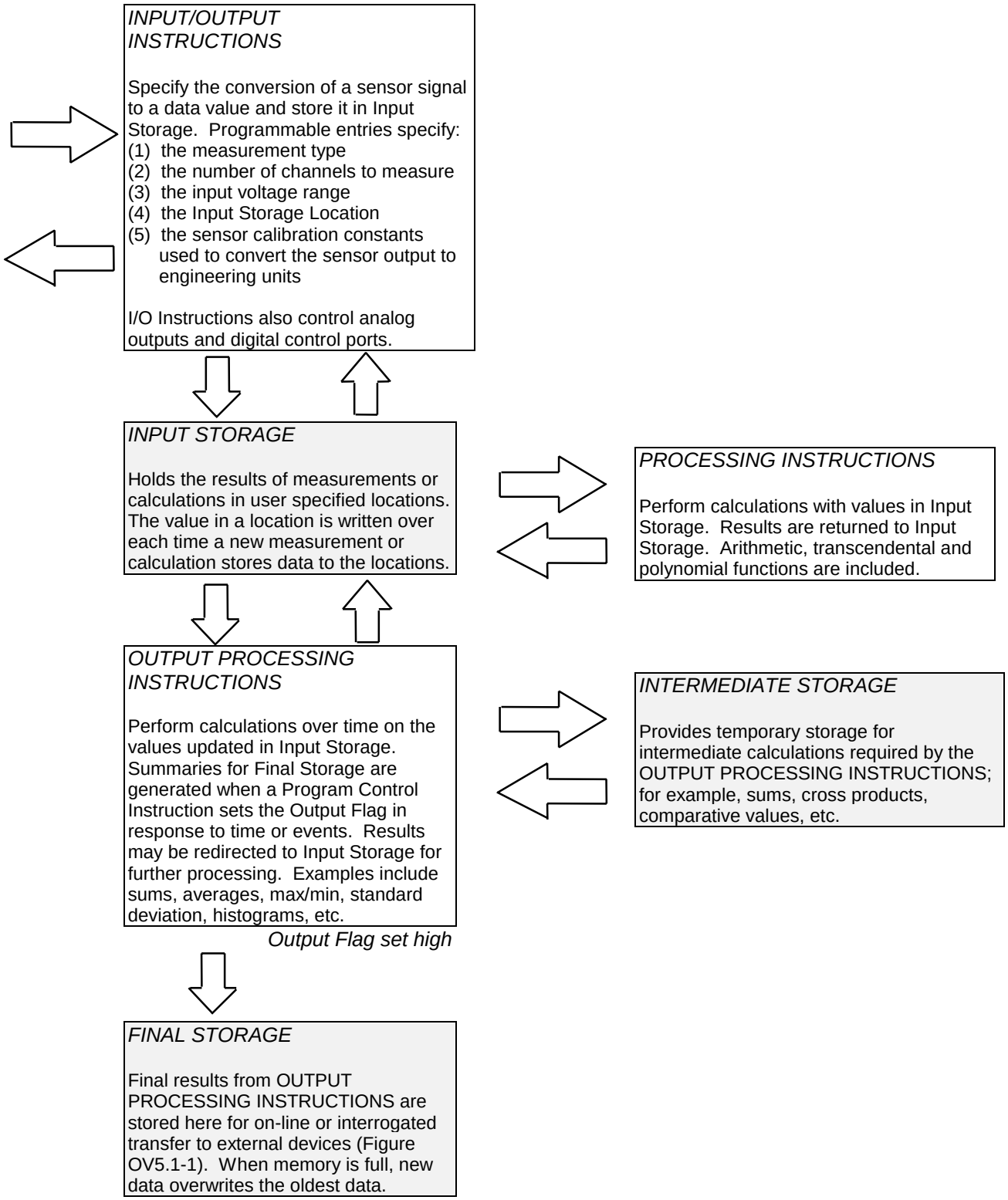


FIGURE OV2.3-1. Instruction Types and Storage Areas

OV3. COMMUNICATING WITH CR510

An external device must be connected to the CR510's Serial I/O port to communicate with the CR510. This may be either Campbell Scientific's CR10KD Keyboard Display or a computer/terminal.

The CR10KD is powered by the CR510 and connects directly to the serial port via the SC12 cable (supplied with the CR10KD). No interfacing software is required.

Computer communication and program editing is accomplished using Campbell Scientific's datalogger support software. This package contains a program editor (EDLOG), datalogger communications, automated telecommunications data retrieval, a data reduction program, and programs to retrieve data from Campbell Scientific Storage Modules.

To participate in the programming examples (Section OV5) you must communicate with the CR510. Read Section OV3.1 if the CR10KD is being used or Section OV3.2 if datalogger support software is being used.

OV3.1 KEYBOARD/DISPLAY

The SC12 cable (supplied with the CR10KD) is used to connect the Keyboard/Display to the 9 pin Serial I/O port on the CR510.

If the Keyboard/Display is connected to the CR510 prior to being powered up, the "HELLO" message is displayed while the CR510 checks memory. The total size of memory is then displayed (256 for 256 K bytes of memory). When the CR10KD is plugged in after the CR510 has powered up, the display is meaningless until "∗" is pressed to enter a mode.

This manual describes direct interaction with the CR510. If you have a CR10KD, work through the direct programming examples in this overview in addition to using EDLOG and you will have the basics of CR510 operation as well as an appreciation for the help provided by the software.

OV3.1.1 FUNCTIONAL MODES

CR510/User interaction is broken into different functional MODES (e.g., programming the measurements and output, setting time, manually initiating a block data transfer to Storage Module, etc.). The modes are referred to as Star (∗) Modes since they are accessed by first keying ∗, then the mode number or letter. Table OV3.1-1 lists the CR510 Modes.

TABLE OV3.1-1. ∗ Mode Summary

Key	Mode
∗ 0	LOG data and indicate active Tables
∗ 1	Program Table 1
∗ 2	Program Table 2
∗ 3	Program Table 3, subroutines only
∗ 4	Parameter Entry Table
∗ 5	Display/set real time clock
∗ 6	Display/alter Input Storage data, toggle flags or control ports.
∗ 7	Display Final Storage data
∗ 8	Final Storage data transfer to peripheral
∗ 9	Storage Module commands
∗ A	Memory allocation/reset
∗ B	Signature/status
∗ C	Security
∗ D	Save/load Program
∗ #	Used with TGT1 satellite transmitter

OV3.1.2 KEY DEFINITION

Keys and key sequences have specific functions when using the CR10KD keyboard or a computer/terminal in the remote keyboard state (Section 5). Table OV3.1-2 lists these functions. In some cases, the exact action of a key depends on the mode the CR510 is in and is described with the mode in the manual.

TABLE OV3.1-2 Key Description/Editing Functions

Key	Action
[0] - [9]	Key numeric entries into display
[*]	Enter Mode (followed by Mode Number)
[A]	Enter/Advance
[B]	Back up
[C]	Change the sign of a number or index an input location to loop counter
[D]	Enter the decimal point
[#]	Clear the rightmost digit keyed into the display
[#] [A]	Advance to next instruction in program table (∗1, ∗2, ∗3) or to next Output Array in Final Storage (∗7)
[#] [B]	Back up to previous instruction in program table or to previous Output Array in Final Storage
[#] [D]	Delete entire instruction
[#] [0]	(then A) Back up to the start of the current array.

When using a computer/terminal to communicate with the CR510 (Telecommunications remote keyboard state) there are some keys available in addition to those found on the CR10KD. Table OV3.1-3 lists these keys.

TABLE OV3.1-3. Additional Keys Allowed in Telecommunications

Key	Action
-	Change Sign, Index (same as C)
CR	Enter/advance (same as A)
:	Colon (used in setting time)
S or ^S	Stops transmission of data (10 second time-out; any character restarts)
C or ^C	Aborts transmission of Data

OV3.2 USING COMPUTER WITH DATALOGGER SUPPORT SOFTWARE

Direct datalogger communication programs in the datalogger support software provide menu selection of tools to perform the datalogger functions (e.g., set clock, send program, monitor measurements, and collect data). The user also has the option of directly entering keyboard commands via a built-in terminal emulator (Section OV3.3).

When using the support software, the computer's baud rate, port, and modem types are specified and stored in a file for future use.

The simplest and most common interface is the SC32A Optically Isolated RS232 Interface. The SC32A converts and optically isolates the voltages passing between the CR510 and the external terminal device.

The SC12 Two Peripheral cable which comes with the SC32A is used to connect the serial I/O port of the CR510 to the 9 pin port of the SC32A labeled "Datalogger". Connect the "Terminal/Printer" port of the SC32A to the serial port of the computer with a straight 25 pin cable or, if the computer has a 9 pin serial port, a standard 9 to 25 pin adapter cable.

OV3.3 ASCII TERMINAL OR COMPUTER WITH TERMINAL EMULATOR

Devices which can be used to communicate with the CR510 include standard ASCII terminals and computers programmed to function as a terminal emulator. See Section 6.7 for details.

To communicate with any device other than the CR10KD, the CR510 enters its Telecommunications Mode and responds only to valid telecommunications commands. Within the Telecommunications Mode, there are 2 "states"; the Telecommunications Command state and the Remote Keyboard state. Communication is established in the Telecommunications command state. One of the commands is to enter the Remote Keyboard state (Section 5).

The Remote Keyboard state allows the keyboard of the computer/terminal to act like the CR10KD keyboard. Various datalogger modes may be entered, including the mode in which programs may be keyed in to the CR510 from the computer/terminal.

OV4. PROGRAMMING THE CR510

A datalogger program is created on a computer using EDLOG or one of the programming aids such as Short Cut. A program can also be entered directly into the datalogger. Section OV4.3 describes options for loading the program into the CR510.

OV4.1 PROGRAMMING SEQUENCE

In routine applications, the CR510 measures sensor output signals, processes the measurements over some time interval and stores the processed results. A generalized programming sequence is:

1. Enter the execution interval. In most cases, the execution interval is determined by the desired sensor scan rate.

2. Enter the Input/Output instructions required to measure the sensors.

3. If processing in addition to that provided by the Output Processing Instructions (step 5) is required, enter the appropriate Processing Instructions.

4. Enter the Program Control Instruction to test the output condition and set the Output Flag when the condition is met. For example, use

   Instruction 92 to output based on time.

   Instruction 86 to output every execution interval.

   Instruction 88 or 89 to output based on a comparison of values in input locations.

   This instruction must precede the Output Processing Instructions which store data in Final Storage. Instructions are described in Sections 9 through 12.

5. Enter the Output Processing Instructions to store processed data in Final Storage. The order in which data are stored is determined by the order of the Output Processing Instructions in the table.

6. Repeat steps 4 and 5 for additional outputs on different intervals or conditions.

NOTE: The program must be executed for output to occur. Therefore, the interval at which the Output Flag is set must be evenly divisible by the execution interval. For example, with a 2 minute execution interval and a 5 minute output interval, the program will only be executed on the even multiples of the 5 minute intervals, not on the odd. Data will be output every 10 minutes instead of every 5 minutes.

Execution intervals and output intervals set with Instruction 92 are synchronized with datalogger time starting at midnight.

OV4.2 INSTRUCTION FORMAT

Instructions are identified by an instruction number. Each instruction has a number of parameters that give the CR510 the information it needs to execute the instruction.

The CR510 Prompt Sheet has the instruction numbers in red, with the parameters briefly listed in columns following the description. Some parameters are footnoted with further description under the "Instruction Option Codes" heading.

For example, Instruction 73 stores the maximum value that occurred in an Input Storage location over the output interval.

P73 Maximum
- 1: Reps
- 2: TimeOption
- 3: Loc

The instruction has three parameters (1) REPetitionS, the number of sequential Input Storage locations on which to find maxima, (2) TIME, an option of storing the time of occurrence with the maximum value, and (3) LOC the first Input Storage location operated on by the Maximum Instruction. The codes for the TIME parameter are listed in the "Instruction Option Codes".

The repetitions parameter specifies how many times an instruction's function is to be repeated. For example, four 107 thermistor probes may be measured with a single Instruction 11, Temp-107, with four repetitions. Parameter 2 specifies the input channel of the first thermistor (the probes must be connected to sequential channels). Parameter 4 specifies the Input Storage location in which to store measurements from the first thermistor. If location 5 were used and the first probe was on channel 1, the temperature of the thermistor on channel 1 would be stored in input location 5, the temperature from channel 2 in input location 6, etc.

Detailed descriptions of the instructions are given in Sections 9-12. Entering an instruction into a program table is described in OV5.

OV4.3 ENTERING A PROGRAM

Programs are entered into the CR510 in one of three ways:

1. Keyed in using the CR10KD keyboard.
2. Loaded from a pre-recorded listing using the ∗D Mode. There are 2 types of storage/input:
   a. Stored on disk/sent from computer.
   b. Stored/loaded from Storage Module.
3. Loaded from internal Flash Memory or Storage Module upon power-up.

A program is created by keying it directly into the datalogger as described in Section OV5, or on a PC using EDLOG or a programming aid such as Short Cut.

Program files (.DLD) can be downloaded directly to the CR510 using Campbell's datalogger support software. Communication via direct wire, telephone, or Radio Frequency (RF) is supported.

Programs can be copied to a Storage Module with the appropriate software. Using the ∗D Mode to save or load a program from a Storage Module is described in Section 1.8.

Once a program is loaded in the CR510, the program will be stored in flash memory and will automatically be loaded and run when the datalogger is powered-up.

The program on power up function can also be achieved by using a Storage Module. Up to 8 programs can be stored in the Storage Module, the programs may be assigned any of the numbers 1-8. If the Storage Module is connected when the CR510 is powered-up the CR510 will automatically load program number 8, provided that a program 8 is loaded in the Storage Module (Section 1.8). The program from the Storage Module will replace the active program in flash memory.

OV5. PROGRAMMING EXAMPLES

The following examples stress direct interaction with the CR510 using the CR10KD. At the beginning of each example is an EDLOG listing of the program. You can also participate in the example by entering the program in EDLOG and sending it to the CR510 and viewing measurements with Campbell's datalogger support software. If you have the CR10KD, work through the examples as well as using EDLOG. You will learn the basics of CR510 operation as well as an appreciation for the help provided by the software.

We will start with a simple programming example. There is a brief explanation of each step to help you follow the logic. When the example uses an instruction, find it on the Prompt Sheet and follow through the description of the parameters. Using the Prompt Sheet while going through these examples will help you become familiar with its format. Sections 9-12 have more detailed descriptions of the instructions.

Connect the CR510 to the CR10KD Keyboard/Display or a terminal (Section OV3). Hook up the power leads as described in Section OV1.2. The programming steps in the following examples use the keystrokes possible on the keyboard/display. With a terminal, some responses will be slightly different.

If the CR10KD is connected to the CR510 when it is powered up, the display will show:

Display	Explanation
HELLO	On power-up, the CR510 displays "HELLO" while it checks the memory (this display occurs only with the CR10KD).

after a few seconds delay

:0256	The size of the machine's total memory, 256 K (1280 if 1 meg option).

When primary power is applied to the CR510, it tests the FLASH memory and loads the current program to RAM. After the program compiles successfully, the CR510 begins executing the program. If the ring line on the 9 pin connector is raised while the CR510 is testing memory,

there will be a 128 second delay before compiling and running the program. This can be used to edit or change the program before it starts running. To raise the ring line, press any key on the CR10KD keyboard display or call the CR510 with the computer during the power up sequence (i.e., while "HELLO" is displayed on the CR10KD).

In order to ensure that there is no active program in the CR510, we will load an empty program using the *D Mode:

Key	Display Will Show: (ID:Data)	Explanation
[*]	00:00	Enter mode
[D]	13:00	Enter *D Mode
[7]	13:7	7 is command to load program from flash
[A]	07:00	Execute command 7, CR510 is ready for program number
[0]	07:0	Load Program 0 (empty program)
[A]		Execute program load, after a short wait, the display will show
	13:0000	Indicating that the command is complete.

OV5.1 SAMPLE PROGRAM 1

EDLOG Listing Program 1:

```
*Table 1 Program
01:         5.0         Execution Interval (seconds)

1: Internal Temperature (P17)
   1:       1           Loc [ CR510Temp ]

2: Do (P86)
   1:       10          Set Output Flag High

3: Sample (P70)
   1:       1           Reps
   2:       1           Loc [ CR510Temp ]
```

In this example the CR510 is programmed to read its own internal temperature (using a built in thermistor) every 5 seconds and to send the results to Final Storage.

Key	Display Will Show: (ID:Data)	Explanation
[*]	00:00	Enter mode.
[1]	01:0000	Enter Program Table 1.
[A]	01:0.0000	Advance to execution interval (In seconds)
[5]	01:5	Key in an execution interval of 5 seconds.
[A]	01:P00	Enter the 5 second execution interval and advance to the first program instruction location.
[1] [7]	01:P17	Key in Instruction 17 which directs the CR510 to measure the internal temperature in degrees C. This is an Input/Output Instruction.
[A]	01:0000	Enter Instruction 17 and advance to the first parameter.
[1]	01:1	The input location to store the measurement, location 1.
[A]	02:P00	Enter the location # and advance to the second program instruction.

The CR510 is now programmed to read the internal temperature every 5 seconds and place the reading in Input Storage Location 1. The program can be compiled and the temperature displayed.

Key	Display Will Show: (ID:Data)	Explanation
[*] [0]	LOG 1	Exit Table 1, enter *0 Mode, compile table and begin logging.
[*] [6]	06:0000	Enter *6 Mode (to view Input Storage).
[A]	01:21.234	Advance to first storage location. Internal datalogger temp. is 21.234°C (display shows actual temperature so exact value will vary).

Wait a few seconds:

	01:21.423	The CR510 has read the sensor and stored the result again. The internal temp is now 21.423 ^{O}C. The value is updated every 5 seconds when the table is executed. At this point the CR510 is measuring the temperature every 5 seconds and sending the value to Input Storage. No data are being saved. The next step is to have the CR510 send each reading to Final Storage. (Remember, the Output Flag must be set first.)
* 1	01:0000	Exit ∗6 Mode. Enter program table 1.
2 A	02:P00	Advance to 2nd instruction location (this is where we left off).
8 6	02:P86	This is the DO instruction (a Program Control Instruction).
A	01:00	Enter 86 and advance to the first parameter (which will specify the command to execute).
1 0	01:10	This command sets the Output Flag. (Flag 0)
A	03:P00	Enter 10 and advance to third program instruction.
7 0	03:P70	The SAMPLE instruction. It directs the CR510 to take a reading from an Input Storage location and send it to Final Storage (an Output Processing Instruction).
A	01:0000	Enter 70 and advance to the first parameter (repetitions).
1	01:1	There is only one input location to sample; repetitions = 1.
A	02:0000	Enter 1 and advance to second parameter (Input Storage location to sample).
1	02:1	Input Storage Location 1, where the temperature is stored.
A	04:P00	Enter 1 and advance to fourth program instruction.
*	00:00	Exit Table 1.
0	LOG 1	Enter ∗0 Mode, compile program, log data.

The CR510 is now programmed to measure the internal temperature every 5 seconds and send each reading to Final Storage. Values in Final Storage can be viewed using the ∗7 Mode.

Key	Display Will Show: (ID:Data)	Explanation
* 7	07: 13.000	Enter ∗7 Mode. The Data Storage Pointer (DSP) is at Location 13 (in this example).
A	01: 0102	Advance to the first value, the Output Array ID. 102 indicates the Output Flag was set by the second instruction in Program Table 1.
A	02: 21.23	Advance to the first stored temperature.
A	01: 0102	Advance to the next output array. Same Output Array ID.
A	02: 21.42	Advance to 2nd stored temp, 21.42 deg. C.

There are no date and time tags on the data. They must be put there with Output Instruction 77. Instruction 77 is used in the next example.

If a terminal is used to communicate with the CR510, Telecommunications Commands (Section 5) can be used to view entire Output Arrays (in this case the ID and temperature) at the same time.

OV5.2 EDITING AN EXISTING PROGRAM

When editing an existing program in the CR510, entering a new instruction inserts the instruction; entering a new parameter replaces the previous value.

To insert an instruction, enter the program table and advance to the position where the instruction is to be inserted (i.e., P in the data portion of the display) key in the instruction number, and then key A. The new instruction will be inserted at that point in the table, advance through and enter the parameters. The instruction that was at that point and all instructions following it will be pushed down to follow the inserted instruction.

An instruction is deleted by advancing to the instruction number (P in display) and keying #D (Table 4.2-1).

To change the value entered for a parameter, advance to the parameter and key in the correct value then press A. Note that the new value is not entered until A is keyed.

SAMPLE PROGRAM 2

Instruction # (Loc:Entry)	Parameter (Par#:Entry)	Description
∗1		Enter Program Table 1
01:60		60 second (1 minute) execution interval
Key # D until is displayed	01:P00	Erase previous Program before continuing.
01:P11		Measure reference temperature
	01:1	Store temp in Location 1
	02:5	
	03:3	
	04:1	
	05:1.0	
	06:0.0	
02:P92		If Time instruction
	01:0	0 minutes into the interval
	02:60	60 minute interval
	03:10	Set Output Flag 0

The CR510 is programmed to measure the datalogger internal temperature every sixty seconds. The If Time instruction sets the Output Flag high at the beginning of every hour. Next, the Output Instructions for time and average are added.

03:P77		Output Time instruction
	01:110	Store Julian day, hour, and minute
04:P71		Average instruction
	01:1	one repetition
	02:1	Location 1 - source of temps. to be averaged
05:P92		If Time instruction
	01:0	0 minutes into the interval
	02:1440	1440 minute interval (24 hrs.)
	03:10	Set Output Flag 0
06: P77		Output Time instruction
	01:100	Store Julian day

Instruction # (Loc.:Entry)	Parameter (Par.#:Entry)	Description
07: P73		Maximize instruction
	01:1	One repetition
	02:10	Output time of daily maximum in hours and minutes
	03:2	Data source is Input Storage Location 1.
08: P74		Minimize instruction
	01:1	One repetition
	02:10	Output the time of the daily minimum in hours and minutes
	03:1	Data source is Input Storage Location 1.

The program to make the measurements and to send the desired data to Final Storage has been entered. At this point, Instruction 96 is entered to enable data transfer from Final Storage to Storage Module.

09:P96		Activate Serial Data Output.
	1:71	Output Final Storage data to Storage Module.

OV5.3 SETTING THE DATALOGGER TIME

The next example shows how to set the datalogger date and time using the CR10KD. Here the example reverts back to the key-by-key format.

Key	Display	Explanation
* 5	00:21:32	Enter *5 Mode. Clock running but perhaps not set correctly.
A	05:0000	Advance to location for year.
1 9 9 8	05:1998	Key in year (1998).
A	05:0000	Enter and advance to location for Julian day.
1 9 7	05:197	Key in Julian day.
A	05:0021	Enter and advance to location for hours and minutes (24 hr. time).
1 3 2 4	05:1324	Key in hrs.:min. (1:24 PM in this example).
A	:13:24:01	Clock set and running.
* 0	LOG 1	Exit *5, compile Table 1, commence logging data.

OV6. DATA RETRIEVAL OPTIONS

There are several options for data storage and retrieval. These options are covered in detail in Sections 2, 4, and 5. Figure OV6.1-1 summarizes the various possible methods.

Regardless of the method used, there are three general approaches to retrieving data from a datalogger.

1) On-line output of Final Storage data to a peripheral storage device. On a regular schedule, that storage device is either "milked" of its data or is brought back to the office/lab where the data is transferred to the computer. In the latter case, a "fresh" storage device is usually left in the field when the full one is taken so that data collection can continue uninterrupted.

2) Bring a storage device to the datalogger and milk all the data that has accumulated in Final Storage since the last visit.

3) Retrieve the data over some form of telecommunications link, whether it be RF, telephone, short haul modem, or satellite. This can be performed under program control or by regularly scheduled polling of the dataloggers. Campbell Scientific's Datalogger Support Software automates this process.

Regardless of which method is used, the retrieval of data from the datalogger does NOT erase those data from Final Storage. The data remain in the ring memory until:

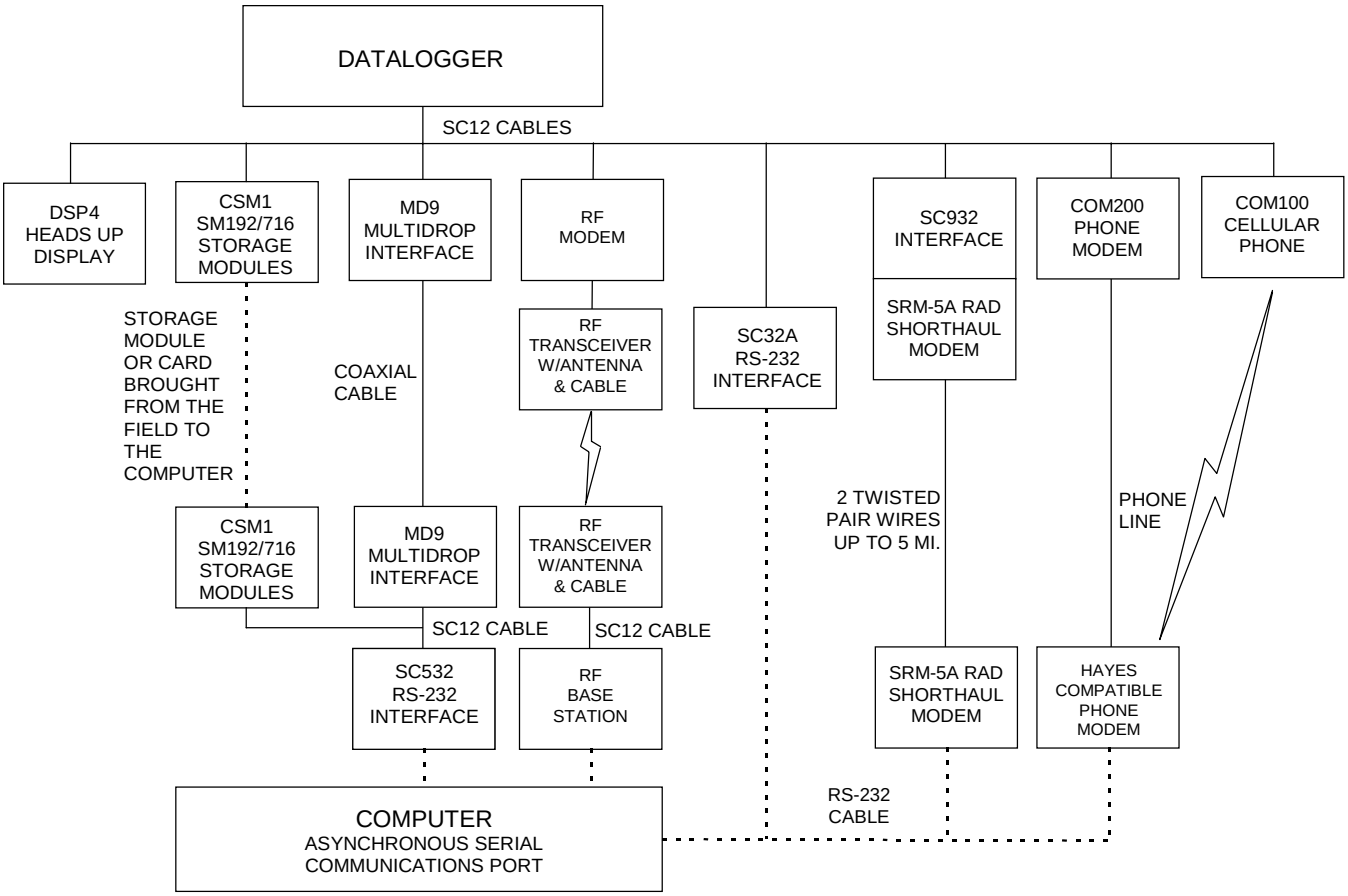
They are written over by new data (Section 2.1)

Memory is reallocated or the CR510 is reset (Section 1.5)

Table OV6.1-1 lists the instructions used with the various methods of data retrieval.

TABLE OV6.1-1. Data Retrieval Methods and Related Instructions

Method	Instruction/Mode	Section in Manual
Storage Module	Instruction 96	4.1, 12
	*8	4.2
	*9	4.5
Telecommunications	Telecommunications Commands	5
	Instruction 97	12
Printer or other Serial device	Instruction 96	4.1, 12
	*8	4.2



NOTES:

1. ADDITIONAL METHODS OF DATA RETRIEVAL ARE:
   A. SATELLITE TRANSMISSION
   B. DIRECT DUMP TO PRINTER
   C. VOICE PHONE MODEM TO VOICE PHONE OR PC WITH HAYES COMPATIBLE PHONE MODEM

2. THE DSP4 HEADS UP DISPLAY ALLOWS THE USER TO VIEW DATA IN INPUT STORAGE. ALSO BUFFERS FINAL STORAGE DATA AND WRITES IT TO PRINTER OR STORAGE MODULE.

3. ALL CAMPBELL SCIENTIFIC RS-232 INTERFACES HAVE A FEMALE 25 PIN RS-232 CONNECTOR.

FIGURE OV6.1-1. Data Retrieval Hardware Options

OV7. SPECIFICATIONS

Electrical specifications are valid over a -25° to +50°C range unless otherwise specified; non-condensing environment required. To maintain electrical specifications, yearly calibrations are recommended.

PROGRAM EXECUTION RATE

System tasks initiated in sync with real-time up to 64 Hz. One measurement with data transfer is possible at this rate without interruption.

ANALOG INPUTS

NUMBER OF CHANNELS: 2 differential or 4 single-ended, individually configured.

RANGE AND RESOLUTION:

Full Scale	*Resolution (µV)*	
Input Range (mV)	*Differential*	*Single-Ended*
±2500	333	666
±250	33.3	66.6
±25	3.33	6.66
±7.5	1.00	2.00
±2.5	0.33	0.66

INPUT SAMPLE RATES: Includes the measurement time and conversion to engineering units. The fast and slow measurements integrate the signal for 0.25 and 2.72 ms, respectively. Differential measurements incorporate two integrations with reversed input polarities to reduce thermal offset and common mode errors.

Fast differential voltage:	4.2 ms
Slow differential voltage:	9.2 ms
Differential with 60 Hz rejection:	25.9 ms

ACCURACY: ±0.1% of FSR (-25° to 50°C);
±0.05% of FSR (0° to 40°C);
e.g., ±0.1% FSR = ±5.0 mV for ±2500 mV range

INPUT NOISE VOLTAGE (for ±2.5 mV range):
Fast differential: 0.82 µV rms
Slow differential: 0.25 µV rms
Differential with 60 Hz rejection: 0.18 µV rms

COMMON MODE RANGE: ±2.5 V

DC COMMON MODE REJECTION: > 140 dB

NORMAL MODE REJECTION: 70 dB (60 Hz with slow differential measurement)

INPUT CURRENT: ±9 nA maximum

INPUT RESISTANCE: 20 Gohms typical

ANALOG OUTPUTS

DESCRIPTION: 2 switched excitations, active only during measurement, one at a time.

RANGE: ±2.5 V

RESOLUTION: 0.67 mV

ACCURACY: ±2.5 mV (0° to 40°C);
±5 mV (-25° to 50°C)

CURRENT SOURCING: 25 mA

CURRENT SINKING: 25 mA

FREQUENCY SWEEP FUNCTION: The switched outputs provide a programmable swept frequency, 0 to 2.5 V square wave for exciting vibrating wire transducers.

RESISTANCE MEASUREMENTS

MEASUREMENT TYPES: The CR510 provides ratiometric bridge measurements of 4- and 6-wire full bridge, and 2-, 3-, and 4-wire half bridges. Precise dual polarity excitation using any of the switched outputs eliminates dc errors. Conductivity measurements use a dual polarity 0.75 ms excitation to minimize polarization errors.

ACCURACY: ±0.02% of FSR plus bridge errors.

PERIOD AVERAGING MEASUREMENTS

DEFINITION: The average period for a single cycle is determined by measuring the duration of a specified number of cycles. Any of the 4 single-ended analog input channels can be used. Signal attenuation and ac coupling is typically required.

INPUT FREQUENCY RANGE:

Signal peak-to-peak[1]		*Min.*	*Max*
Min.	*Max.*	*Pulse w.*	*Freq.[2]*
500 mV	5.0 V	2.5 µs	200 kHz
10 mV	2.0 V	10 µs	50 kHz
5 mV	2.0 V	62 µs	8 kHz
2 mV	2.0 V	100 µs	5 kHz

RESOLUTION: 35 ns divided by the number of cycles measured

ACCURACY: ±0.03% of reading

TIME REQUIRED FOR MEASUREMENT: Signal period multiplied by the number of cycles measured plus 1.5 cycles + 2 ms.

PULSE COUNTERS

NUMBER OF CHANNELS: 2 eight-bit or 1 sixteen-bit; software selectable as switch closure, high frequency pulse, or low-level ac modes. An additional channel (C2/P3) can be software configured to read switch closures at rates up to 40 Hz.

MAXIMUM COUNT RATE: 16 kHz, eight-bit counter; 400 kHz, sixteen-bit counter. Channels are scanned at 8 or 64 Hz (software selectable).

SWITCH CLOSURE MODE:
Minimum Switch Closed Time: 5 ms
Minimum Switch Open Time: 6 ms
Maximum Bounce Time: 1 ms open without being counted

HIGH FREQUENCY PULSE MODE:
Minimum Pulse Width: 1.2 µs
Maximum Input Frequency: 400 kHz
Maximum Input Voltage: ±20 V
Voltage Thresholds: Count upon transition from below 1.5 V to above 3.5 V at low frequencies. Larger input transitions are required at high frequencies because of input filter with 1.2 µs time constant. Signals up to 400 kHz will be counted if centered around +2.5 V with deviations ‡ – 2.5 V for ‡ 1.2 µs.

LOW LEVEL AC MODE:
(Typical of magnetic pulse flow transducers or other low voltage, sine wave outputs.)

Input Hysteresis: 14 mV
Maximum ac Input Voltage: ±20 V
Minimum ac Input Voltage:

*(Sine wave mV rms)**	*Range (Hz)*
20	1 to 1000
200	0.5 to 10,000
1000	0.3 to 16,000

*16-bit config. or 64 Hz scan req'd for freq. > 2048 Hz

DIGITAL I/O PORTS

DESCRIPTION: Port C1 is software selectable as a binary input, control output, or as an SDI-12 port. Port C2/P3 is input only and can be software configured as an SDI-12 port, a binary input, or as a switch closure counter (40 Hz max).

OUTPUT VOLTAGES (no load): high 5.0 V ±0.1 V; low < 0.1 V

OUTPUT RESISTANCE: 500 ohms

INPUT STATE: high 3.0 to 5.5 V; low -0.5 to 0.8 V

INPUT RESISTANCE: 100 kohms

SDI-12 INTERFACE STANDARD

DESCRIPTION: Digital I/O Ports C1-C2 support SDI-12 asynchronous communication; up to ten SDI-12 sensors can be connected to each port. Meets SDI-12 standard Version 1.2 for datalogger and sensor modes.

EMI and ESD PROTECTION

The CR510 is encased in metal and incorporates EMI filtering on all inputs and outputs. Gas discharge tubes provide robust ESD protection on all terminal block inputs and outputs. The following European CE standards apply.

EMC tested and conforms to BS EN61326:1998.

Details of performance criteria applied are available upon request.

CPU AND INTERFACE

PROCESSOR: Hitachi 6303.

PROGRAM STORAGE: Up to 16 kbytes for active program; additional 16 kbytes for alternate programs. Operating system stored in 128 kbytes Flash memory.

DATA STORAGE: 128 kbytes SRAM standard (approximately 62,000 values). Additional 2 Mbytes Flash available as an option.

OPTIONAL KEYBOARD DISPLAY: 8 digit LCD (0.5" digits).

PERIPHERAL INTERFACE: 9 pin D-type connector for keyboard display, storage module, modem, printer, card storage module, and RS-232 adapter.

BAUD RATES: Selectable at 300, 1200, and 9600, 76,800 for certain synchronous devices. ASCII communication protocol is one start bit, one stop bit, eight data bits (no parity).

CLOCK ACCURACY: ±1 minute per month

SYSTEM POWER REQUIREMENTS

VOLTAGE: 9.6 to 16 Vdc

TYPICAL CURRENT DRAIN: 1.3 mA quiescent, 13 mA during processing, and 46 mA during analog measurement.

BATTERIES: Any 12 V battery can be connected as a primary power source. Several power supply options are available from Campbell Scientific. The model CR2430 lithium battery for clock and SRAM backup has a capacity of 270 mAhr.

PHYSICAL SPECIFICATIONS

SIZE: 8.4" x 1.5" x 3.9" (21.3 cm x 3.8 cm x 9.9 cm). Additional clearance required for serial cable and sensor leads.

WEIGHT: 15 oz. (425 g)

WARRANTY

Three years against defects in materials and workmanship.

We recommend that you confirm system configuration and critical specifications with Campbell Scientific before purchase.

CAMPBELL SCIENTIFIC, INC.
815 W. 1800 N. • Logan, Utah 84321-1784 • (435) 753-2342 • FAX (435) 750-9540
Offices also located in: Australia • Brazil • Canada • England • France • South Africa • Spain

SECTION 1. FUNCTIONAL MODES

1.1 DATALOGGER PROGRAMS - ∗1, ∗2, ∗3, AND ∗4 MODES

Data acquisition and processing functions are controlled by user-entered instructions contained in program tables. Programming can be separated into 2 tables, each having its own user-entered execution interval. A third table is available for programming subroutines which may be called by instructions in Tables 1 or 2 or by a special interrupt. The ∗1 and ∗2 Modes are used to access Tables 1 and 2. The ∗3 Mode is used to access Subroutine Table 3.

The ∗4 Mode Table is a table of values used in the program that someone can change while the rest of the program is protected. These values may be used for sensor calibrations or to select optional sensors. The ∗4 Table is only available when a special program created by EDLOG is loaded in the CR510.

When a program table is first entered, the display shows the table number in the ID field and 00 in the data field. Keying an "A" will advance the editor to the execution interval. If there is an existing program in the table, keying an instruction location number prior to "A" will advance directly to the instruction (e.g., 5 will advance to the fifth instruction in the table).

1.1.1 EXECUTION INTERVAL

The execution interval is entered in units of seconds as follows:

1/8 ....31.875 seconds, in multiples of 1/8 (0.125)

32 .....8191 seconds, in multiples of 1 second

Execution of the table is repeated at the rate determined by this entry. The table will not be executed if 0 is entered. Entries less than 32 seconds will be rounded to a valid interval if they are within 1/512 (0.00195) second of a valid interval, otherwise error E41 will be displayed. Entries greater than 32 seconds are rounded to the nearest second.

The sample rate for a CR510 measurement is the rate at which the measurement instruction can be executed (i.e., the measurement made, scaled with the instruction's multiplier and offset, and the result placed in Input Storage). Additional processing requires extra time. The throughput rate is the rate at which a measurement can be made and the resulting value stored in Final Storage. The maximum throughput rate for fast single-ended measurements with standard software is 192 measurements per second (12 measurements repeated 16 times per second).

If the specified execution interval for a table is less than the time required to process that table, the CR510 finishes processing the table and waits for the next occurrence of the execution interval before again initiating the table (i.e., when the execution interval has elapsed and the table is still executing, that execution is skipped). Since no advantage is gained in the rate of execution with this situation, it should be avoided by specifying an execution interval adequate for the table processing time.

> **NOTE:** Whenever the processing time of the user's program exceeds a table's execution interval, an error is logged in memory. The number of overrun errors can be displayed and reset in the ∗B mode (Section 1.6) or using the Telecommunications A command (Section 5.1). An overrun will also cause decimal points to appear on both sides of the sixth digit of the CR10KD. The decimal points will not appear around the G in LOG if the ∗0 Mode is entered before the overrun occurs.

In some cases, the processing time may exceed the execution interval only when the Output Flag is set and extra time is consumed by final Output Processing. This may be acceptable. For example, suppose it is desired to sample some phenomena every 0.125 seconds and output processed data every 10 minutes. The processing time of the table which does this is less than 0.125 seconds except when output occurs (every 10 minutes). With final output the processing time is 1 second. With the execution interval set at 0.125 seconds, and a one second lag between samples once every 10 minutes, 8 measurements out of 4800 (.17%) are missed: an acceptable statistical error for most populations.

1.1.2 SUBROUTINES

Table 3 is used to enter subroutines which may be called with Program Control Instructions in Tables 1 and 2 or other subroutines. The group of instructions which form a subroutine starts with Instruction 85, Label Subroutine, and ends with Instruction 95, End (Section 12).

Subroutine 98 has the unique capability of being executed when port 2 goes high. This subroutine will interrupt Tables 1 and 2 (Section 1.1.3) when port 2 goes high. When the port goes high, the processor awakes within a few microseconds. The port triggers on the rising edge (i.e., when it goes from low to high). If the port stays high, the subroutine is not called again.

1.1.3 TABLE PRIORITY/INTERRUPTS

Table 1 execution has priority over Table 2. If Table 2 is being executed when it is time to execute Table 1, Table 2 will be interrupted. After Table 1 processing is completed, Table 2 processing resumes at the interruption point. If the execution interval of Table 2 coincides with Table 1, Table 1 is executed first, then Table 2.

Interrupts by Table 1 are not allowed in the middle of an instruction or while output to Final Storage is in process (flag 0 is set high). The interrupt occurs as soon as the instruction is completed or flag 0 is set low.

Subroutine 98 can be initiated by port 2 going high (Section 1.1.2), can interrupt either Table 1 or 2 or can occur when neither is being executed. This subroutine can interrupt a table while the Output Flag is set. When the port goes high during the execution of a table, the instruction being executed is completed before the subroutine is run (i.e., as if the subroutine was called by the next instruction).

The priority is Subroutine 98, Table 1, Table 2. If the interrupt subroutine started when neither table was running, then neither table can interrupt it.

While subroutine 98 is being executed as a result of port 2 going high, that port interrupt is disabled (i.e., the subroutine must be completed before the port going high will have any effect).

1.1.4 ∗4 PARAMETER ENTRY TABLE

The ∗4 mode is a table with up to one hundred values. Each value corresponds to an instruction parameter in the datalogger program. When the datalogger compiles the program, values in the ∗4 table are transferred to the corresponding instruction parameter. The datalogger program must be created using EDLOG which allows instruction parameters to be assigned to the ∗4 table.

In a network of datalogger stations, the ∗4 table can be used to simplify site customization and the procedure of information entry. Once a generalized program is developed, application specific details, e.g., sensor calibration, can be entered without accessing the ∗1 and ∗2 program tables or the ∗3 subroutine table.

ASSIGNING PARAMETERS TO ∗4 - EDLOG

The only way to implement the ∗4 mode is through EDLOG. The datalogger program is generated in EDLOG in the normal way.

To assign a parameter to a ∗4 location, position the cursor on the desired parameter and press the "@" key. EDLOG then prompts for the location number in the ∗4 table to be assigned to the associated parameter. After a valid number is entered, EDLOG marks the parameter with "@@nn" to the right of the parameter description, where "nn" is the ∗4 location number.

Older versions of EDLOG (prior to DOS Version 6.0) may not support the ∗4 mode or may require that the support be enabled. To enable the ∗4 mode press the F5 key followed by the "@" key while in EDLOG's edit mode. "F5=∗4 List" is displayed at the top of the screen indicating that EDLOG's ∗4 feature is active. Subsequent use of the F5 key displays a list indicating which ∗4 locations are in use. If your copy of EDLOG is earlier than 6.0 and it does not display "F5=∗4 List", it is likely that that version of EDLOG does not support the ∗4 mode. Please contact Campbell Scientific for details of an upgrade.

Any program parameter or execution interval can be marked for inclusion in the table, as illustrated below.

```
                    PROGRAM

*       Table 1 Program
  01:       0.0       Execution Interval
                      (seconds) @@0

01:     Volts (SE) (P1)
        1:  1         Reps
        2:  1         ±2.5 mV Slow Range
        3:  1         SE Channel
        4:  1         Loc [ _________ ]
        5:  1         Mult @@1
        6:  0         Offset @@2
```

In the above example, ∗4 location 0 is assigned to the program table execution interval, and locations 1 and 2 to the multiplier and offset of the measurement instruction. Note that a default execution interval of zero means the program will not execute until an alternative interval is entered in location 00 of the ∗4 mode. A default multiplier and offset of 1 and 0 means that the measurement value is in units of millivolts. A different multiplier and offset can be entered in ∗4 locations 1 and 2, respectively.

A ∗4 location can be used in only one program parameter. For example, ∗4 locations 0, 1, and 2 used in the example cannot be reused in another instruction in the same program.

If the ∗4 feature is enabled in EDLOG when printing a program to a printer or disk file, the ∗4 list is printed at the end of the file.

Once the EDLOG created program has been sent to the CR510, it can be saved in the Flash memory program storage area using the ∗D Mode (Section 1.8).

CHANGING VALUES IN ∗4 TABLE

Enter the ∗4 Mode by keying "∗4"; "04:00" is then displayed. At this point it is possible to jump to any valid ∗4 location by keying the desired location number and pressing the A key. For example, when the display shows 04:00 and the desired location is 80, key in the number 80, press the A key and the display will show "80:XXXXX." where XXXXX. is the value stored in location 80. Pressing the "A" key advances to the next ∗4 location, and the "B" key backs up to the previous location. If a ∗4 location is not assigned in the datalogger program, it can not be displayed in the ∗4 mode.

To enter a value in a ∗4 location, advance to the desired location, key in the number and enter it by pressing the "A" key. The value is not entered if the "A" key is not pressed.

Entering a new value causes the datalogger to stop logging. Logging resumes when the program is compiled. Upon compiling, all current ∗4 values are incorporated into the program. For this reason, whenever changes are made in the ∗4 mode, make sure that all ∗4 values are correct before exiting the ∗4 mode.

Removing or adding an instruction to a program residing in the datalogger disables the ∗4 mode. An instruction parameter may be edited without any adverse affect. If the ∗4 mode is disabled, it may be reactivated by downloading the program to the datalogger or, if the program was saved to Flash storage, retrieving the program from the stored program area.

The ∗C mode (Section 1.7) may be used to secure the datalogger program and the ∗4 mode entries. The lowest level of security prevents access to the ∗1, ∗2, and ∗3 modes. Higher levels of security block ∗4.

The CR510 will not respond to the ∗4 command if any of the following conditions exist.

- the program that was downloaded does not contain any ∗4 assignments.
- a program that was downloaded has since been hand edited.
- Security is blocking access to ∗4.

1.1.5 COMPILING A PROGRAM

When a program is first loaded, or if any changes are made in the ∗1, ∗2, ∗3, ∗4, ∗A, or ∗C Modes, the program must be compiled before it starts running. The compile function checks for programming errors and optimizes program information for use during program execution. If errors are detected, the appropriate error codes are indicated on the display (Section 3.10). The compile function is executed when the ∗0 , ∗6, or ∗B Modes are entered and prior to saving a program listing in the ∗D Mode. The compile function is only executed after a program change has been made and any subsequent use of any of these

modes will return to the mode without recompiling.

When the ∗0 or ∗B Mode is used to compile, all output ports and flags are set low, the timer is reset, and data values contained in Input and Intermediate Storage are reset to zero.

When the ∗6 Mode is used to compile data values contained in Input Storage, the state of flags, control ports, and the timer (Instruction 26) are unaltered. Compiling always zeros Intermediate Storage.

1.2 SETTING AND DISPLAYING THE CLOCK - ∗5 MODE

The ∗5 Mode is used to display or set time. When "∗5" is entered, time is displayed. It is updated approximately once a second or longer depending on the rate and degree of data collection and processing taking place. The sequence of time parameters displayed in the ∗5 Mode is given in Table 1.2-1.

To set the year, day, or hours and minutes, enter the ∗5 Mode and advance to display the appropriate value. Key in the desired number and enter the value by keying "A". When a new value for hours and minutes is entered, the seconds are set to zero and current time is again displayed. To exit the ∗5 Mode, key "∗" and the mode you wish to enter.

When the time is changed, a partial recompile is done automatically to synchronize the program with real time.

Changing time affects the output and execution intervals in which time is changed. Because time can only be set with a 1 second resolution, execution intervals of 1 second or less remain constant. Averaged values will still be accurate, though the interval may have a different number of samples than normal. Totalized values will reflect the different number of samples. The pulse count instruction will use the previous interval's value if an option has been selected to discard odd intervals, otherwise it will use the count accumulated in the interval.

TABLE 1.2-1. Sequence of Time Parameters in ∗5 Mode

Key	Display ID:DATA	Description
[∗] [5]	:HH:MM:SS	Display current time
[A]	05:XXXX	Display/enter year
[A]	05:XXXX	Display/enter day of year 1-365(366)
[A]	05:HH:MM:	Display/enter hours:minutes

1.3 DISPLAYING/ALTERING INPUT MEMORY, FLAGS, AND PORTS - ∗6 MODE

The ∗6 Mode is used to display and/or change Input Storage values and to toggle and display user flags and ports. If the ∗6 Mode is entered immediately following any changes in program tables, the program will be compiled and run.

> **NOTE:** Input Storage data and the state of flags, control ports, and the timer (Instruction 26) are UNALTERED whenever program tables are altered and recompiled with the ∗6 Mode. Compiling always zeros Intermediate Storage.

TABLE 1.3-1. ∗6 Mode Commands

Key	Action
[A]	Advance to next input location or enter new value
[B]	Back-up to previous location
[C]	Change value in input location (followed by keyed in value, then "A")
[D]	Display/alter user flags
[0]	Display/alter ports
[#]	Display current location and allow a location number to be keyed in, followed by "A" to jump to that location

1.3.1 DISPLAYING AND ALTERING INPUT STORAGE WITH THE KEYBOARD DISPLAY

When "∗6" is entered, the keyboard/display will read "06:0000". One can advance to view the value stored in input location 1 by keying "A". To go directly to a specific location, key in the location number before keying "A". For example, to view the value contained in Input

Storage location 20, key in "*6 20 A". The ID portion of the display shows the last 2 digits of the location number. If the value stored in the location being monitored is the result of a program instruction, the value on the keyboard/display will be the result of the most recent scan and will be updated each time the instruction is executed. When using the ∗6 Mode from a remote terminal, a number (any number) must be sent before the value shown will be updated.

Input locations can be used to store parameters for use in computations. To store a value in a location, or change the current value, key "C" while monitoring the location, followed by the desired number and "A". This number will be saved once the program is recompiled.

If an algorithm requires parameters to be manually modified during execution of the Program without interruption of the Table execution process, the ∗6 Mode can be used. (If parameters will not need modification, it is better to load them from the program using Instruction 30.) If initial parameter values are required to be in place before program execution commences, use Instruction 91 at the beginning of the program table to prevent the execution until a flag is set (see the next section). Initial parameter values can be entered into input locations using the ∗6 Mode C command. The flag can then be set to enable the table(s).

If the program is altered and compiled with ∗0 Mode, all values previously entered via ∗6C will be set to zero. To preserve ∗6C entered values, compile in the ∗6 Mode after changing the program.

1.3.2 DISPLAYING AND TOGGLING USER FLAGS

If D is keyed while the CR510 is displaying a location value, the current status of the user flags will be displayed in the following format: "00:010010". The characters represent the flags, the left-most digit is Flag 1 and right most is Flag 8. A "0" indicates the flag is clear or “low” and a "1" indicates the flag is set or “high”. In the above example, Flags 4 and 7 are set. To toggle a flag, simply press the corresponding number. To return to displaying the input location, press "A".

Entering appropriate flag tests into the program allows manual control of program execution.

For example, to manually start the execution of Table 2: enter Instruction 91 as the first instruction in Table 2. The first parameter is 25 (do if Flag 5 is low), the second parameter is 0, go to end of program table. If Flag 5 is low, all subsequent instructions in Table 2 will be skipped. Flag 5 can be toggled from the ∗6 Mode, effectively starting and stopping the execution of Table 2.

1.3.3 DISPLAYING AND TOGGLING PORTS

The status of the CR510 ports can be displayed by hitting "0" while looking at an input location (e.g., ∗6 A0). Ports are displayed left to right as 0, 0, 0, 0, 0, 0, C2, C1 (exactly opposite to the flags). A port configured as output can be toggled by hitting its number while in the port display mode. There is no effect on C2 because it is configured as an input only, or on C1 when it is configured as input only.

On power up all ports are configured as inputs. Instruction 20 is used to configure C1 as an output. Port C1 can also be configured as an output by any program control commands which uses the port as an output (pulse, set high, set low, toggle).

1.4 COMPILING AND LOGGING DATA - ∗0 MODE

When the ∗0 Mode is entered after programming the CR510, the program is compiled and the display shows "LOG" followed by the active program table numbers. The display is not updated after entering ∗0.

> **NOTE:** All output ports are set low, the timer is reset, and data values in Input and Intermediate Storage are RESET TO ZERO whenever the program tables are altered and the Program is recompiled with the ∗0 Mode. The same is true when the programs are compiled with ∗B or ∗D.

To minimize current drain, the CR510 should be left in the ∗0 Mode when logging data.

1.5 MEMORY ALLOCATION - ∗A

1.5.1 INTERNAL MEMORY

When powered up with the keyboard display attached, the CR10KD displays HELLO while performing a self check. The total system

memory is then displayed in K bytes. The size of memory can be displayed in the ∗B mode.

Input Storage is used to store the results of Input/Output and Processing Instructions. The values stored in input locations may be displayed using the ∗6 Mode (Section 1.3).

Final Storage holds stored data for a permanent record. Output Instructions store data in Final Storage when the Output Flag is set (Section 3.7). The data in Final Storage can be monitored using the ∗7 Mode (Section 2.3).

Intermediate Storage is a scratch pad for Output Processing Instructions. It is used to store the results of intermediate calculations necessary for averages, standard deviations, histograms, etc. Intermediate Storage is not accessible by the user.

Each Input or Intermediate Storage location requires 4 bytes of memory. Each Final Storage location requires 2 bytes of memory. Low resolution data points require 1 Final Storage location, and high resolution data points require 2. Section 2 describes Final Storage and data retrieval in detail.

Figure 1.5-1 lists the basic memory functions and the amount of memory allotted to them.

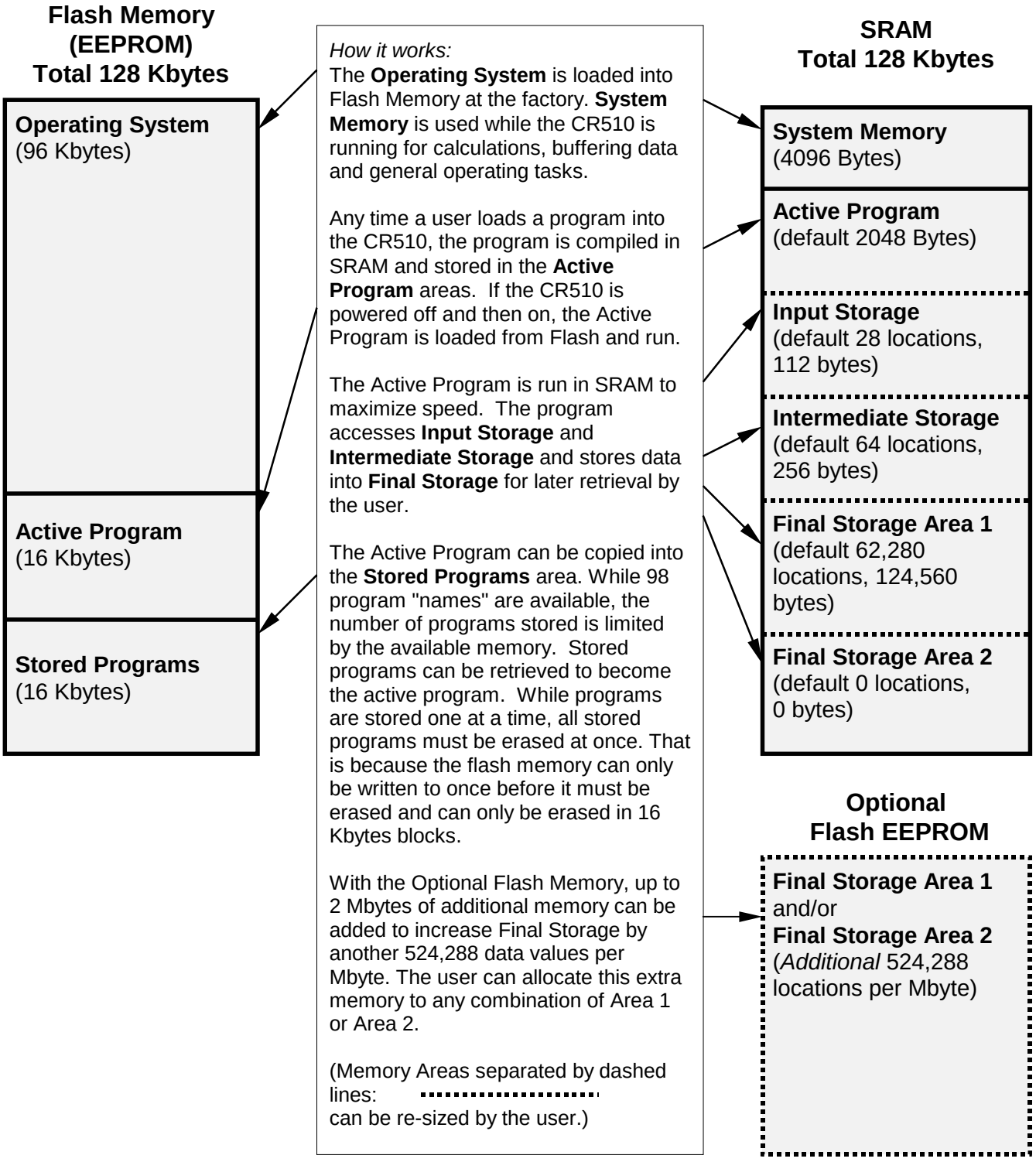


FIGURE 1.5-1. CR510 Memory

1.5.2 ∗A MODE

The ∗A Mode is used to 1) determine the number of locations allocated to Input Storage, Intermediate Storage, Final Storage Area 2, Final Storage Area 1, and Program Memory; 2) repartition this memory; 3) check the number of bytes remaining in Program memory; 4) erase Final Storage; and 5) to completely reset the datalogger.

A second Final Storage area (Storage Area 2) can be allocated in the ∗A Mode. On power up, the number of locations allocated for Storage Area 2 defaults to 0. Final Storage Area 1 is the source from which memory is taken when Input, Intermediate, Final Storage Area 2, or program memories are increased. When they are reduced, Final Storage Area 1 memory is increased. Allocation of Input and Intermediate Storage locations does NOT change Final Storage Area 2.

With the Flash EEPROM 1 or 2 meg. expanded memory options, the boundary between Area 1 and Area 2 must lie between 32 K location sectors. Entries for Area 2 greater than 32,769 locations will be rounded up to the next boundary.

When ∗A is entered, the first number displayed is the number of memory locations allocated to Input Storage. The "A" key is used to advance through the next 6 windows. Table 1.5-2 describes what the values in the ∗A Mode represent.

At the reset, the memory allocation defaults to the values in Figure 1.5-1. The size of Final Storage is determined by the size of memory installed.

The sizes of Input, Intermediate, Final Storage Area 2, and Program Memory may be altered by keying in the desired value and entering it by keying "A". One Input or Intermediate Storage location can be exchanged for two Final Storage locations. The size of Final Storage Area 1 will be adjusted automatically.

The maximum size of Input and Intermediate Storage and the minimum size of Final Storage are determined by the memory installed (Figure 1.5-1). A minimum 28 Input location and one Final Storage Area 1 location will ALWAYS be retained. The size of Intermediate Storage may be reduced to 0.

TABLE 1.5-2. Description of ∗A Mode Data

Keyboard Entry	Display ID: Data	Description of Data
∗ A	01: XXXX	**Input Storage Locations** (minimum of 28, maximum limited by available memory and constraints on Final Storage). This value can be changed by keying in the desired number.
A	02: XXXX	**Intermediate Storage Locations** (maximum limited by available memory and constraints on Input and Final Storage). This value can be changed by keying in the desired number. **Enter 0 and the CR510 will assign the exact number needed**. Entering 0 will also result in the CR510 erasing all data whenever the program is changed and compiled.
A	03: XXXXX	**Final Storage Area 2 Locations** (minimum of 0, maximum limited by available memory). Changing this number automatically reallocates Final Storage Area 1.
A	04: XXXXX	**Final Storage Area 1 Locations** (minimum of 1). This number is automatically altered when the memory allocation for Program, Input, Intermediate, or Final Storage Area 2 is changed.
A	05:	**Bytes allocated for user program.** The number of bytes to assign to program memory can be keyed in to change the size of program memory. Changing the size of program memory results in all data being erased. **Enter 0 and the CR510 will assign the exact number needed**. Entering 0 will also result in the CR510 erasing all data whenever the program is changed and compiled. **Key in 98765 to completely reset datalogger.**
A	06:	**Bytes free in program memory.** The user cannot change this window. It is a function of window 5 and the program.

After repartitioning memory, the program must be recompiled. Compiling erases Intermediate Storage. Compiling with ∗0 erases Input Storage; compiling with ∗6 leaves Input Storage unaltered.

If Intermediate Storage size is too small to accommodate the programs or instructions entered, the "E:04" ERROR CODE will be displayed in the ∗0, ∗6, and ∗B Modes. The user may remove this error code by entering a larger value for Intermediate Storage size. Intermediate Storage and Program Memory can be automatically allocated by entering 0 for their size. When automatic allocation is used, all data are erased any time the program is exchanged and recompiled. Final Storage size is maximized by limiting Intermediate Storage and Program Memory to the minimum necessary. The size of Final Storage and the rate at which data are stored determines how long it will take for Final Storage to fill, at which point new data will write over old.

Intermediate Storage and Final Storage are erased when memory is repartitioned. This feature may be used to clear memory without altering programming. The number of locations (Windows 1-4) does not actually need to be changed; the same value can be keyed in and entered.

ENTERING 98765 for the number of bytes to allocate for program memory (5th Window) COMPLETELY RESETS THE CR510. All memory is erased including any stored programs and memory is checked. Memory allocation returns to the default. The reset operation requires approximately 1 minute for a CR510, 5 minutes for a CR510-1M, and 10 minutes for a CR510-2M. Please be patient while the reset takes place; if the CR510 is turned off in the middle of a reset, it will perform the reset the next time it is powered up.

1.6 MEMORY TESTING AND SYSTEM STATUS - ∗B

The ∗B Mode is used to check the status of the program's operating system and lithium battery. Table 1.6-1 describes what the values seen in the ∗B Mode represent.

A signature is a number which is a function of the data and the sequence of data in memory. It is derived using an algorithm which assures a 99.998% probability that if either the data or its sequence changes, the signature changes. The signature of the program memory is used to determine if the program tables have been altered. During the self check on reset, the signature computed for the Operating Sytem (OS) is compared with a stored signature to determine if a failure has occurred. The algorithm used to calculate the signature is described in Appendix C.

> **NOTE:** Instruction 19 calculates one signature for the program *and* the Operating System. Because this is a combined signature, it is not the same as the signatures in Windows 1 or 2.

The contents of windows 6 and 7, Operating System (OS) version and version revision, are helpful in determining what OS is in the datalogger. As different versions are released, there may be operational differences. **When calling Campbell Scientific for datalogger assistance, please have these numbers available.**

TABLE 1.6-1. Description of ∗B Mode Data

Keyboard Entry	Display ID: Data	Description of Data
∗ B	01: XXXXX	Program memory Signature. The value is dependent upon the programming entered and memory allotment. If the program has not been previously compiled, it will be compiled and run.
A	02: XXXXX	Operating System (OS) Signature
A	03: XXXXX	Memory Size, Kbytes (Flash + SRAM)
A	04: XX	Number of E08 occurrences (Key in 88 to reset)
A	05: XX	Number of overrun occurrences (Key in 88 to reset)
A	06: X.XXXX	Operating System version number
A	07: XXXX.	Version revision number
A	08: X.XXXX	Lithium battery voltage (measured daily)
A	09: XX	Low 12 V battery detect counter (Key in 88 to reset)
A	10: XX	Extended memory error counter (Key in 88 to reset)
A	11: X.XXXX	Extended Memory time of erase, seconds

TABLE 1.7-1. ∗C Mode Entries

SECURITY DISABLED

Keyboard Entry	Display ID: Data	Description
∗ C	01:XXXX	Non-zero password blocks entry to ∗1, ∗2, ∗3, ∗A, and ∗D Modes.
A	02:XXXX	Non-zero password blocks ∗4, ∗5, and ∗6 except for display.
A	03:XXXX	Non-zero password blocks ∗5, ∗6, ∗7, ∗8, ∗9, ∗B, and all telecommunications commands except A, L, N, and E.

SECURITY ENABLED

Keyboard Entry	Display ID: Data	Description
∗ C	12:0000	Enter password. If correct, security is temporarily unlocked through that level.
A	01:XX	Level to which security has been disabled. 0 -- Password 1 entered (everything unlocked) 1 -- Password 2 entered 2 -- Password 3 entered

1.7 ∗C MODE -- SECURITY

The ∗C Mode is used to block access to the user's program information and certain CR510 functions. There are 3 levels of security, each with its own 4 digit password. Setting a password to a non-zero value "locks" the functions secured at that level. The password must subsequently be entered to temporarily unlock security through that level. Passwords are part of the program. If security is enabled in the active program, it is enabled as soon as the program is run when the CR510 is powered up.

When security is disabled, ∗C will advance directly to the window containing the first password. A non-zero password must be entered in order to advance to the next window. Leaving a password 0, or entering 0 for the password disables that and subsequent levels of security.

Security may be temporarily disabled by entering a password in the ∗C Mode or using the telecommunications L command (Section 5.1). The password entered determines what operations are unlocked (e.g., entering password 2 unlocks the functions secured by passwords 2 and 3). Password 1 (everything unlocked) must be entered before any passwords can be altered.

When security is temporarily disabled in the ∗C Mode, entering ∗0 will automatically re-enable security to the level determined by the passwords entered.

The telecommunications L command temporarily changes the security level. After hanging up, security is reset.

1.8 ∗D MODE -- SAVE OR LOAD PROGRAM

The ∗D Mode is used to save or load CR510 programs, to set the degree to which memory is cleared on powerup, to set the datalogger ID, and to set communication to full or half duplex.

Programs (∗1, ∗2, ∗3, ∗4, ∗A, ∗C, and ∗D Mode data) may be stored to and from computers, internal flash memory, and Storage Modules. Several programs can be stored in the CR510 Flash Memory and later recalled and run using the ∗D Mode or Instruction 111.

Campbell Scientific's datalogger support software automatically makes use of the ∗D Mode to upload and download programs from a computer. Appendix C gives some additional information on Commands 1 and 2 that are used for these operations.

When "∗D" is keyed in, the CR510 will display "13:00". A command (Table 1.8-1) is entered by keying the command number and "A".

TABLE 1.8-1. ∗D Mode Commands

Command	Description
1	Send (Print) ASCII Program
2	Load ASCII Program, ∗0 Compile
2--	Load ASCII Program, ∗6 Compile
6	Store Program in Flash
7	Load Program from Flash
7N	Save/Load/Clear Program from Storage Module N
8	Set Datalogger ID
9	Set Full/Half Duplex
10	Set Powerup Options

If the CR510 program has not been compiled when the command to save a program is entered, it will be compiled before the program is saved. When a program is loaded, it is immediately compiled and run. When a command is complete, "13:0000" is displayed; ∗D must be entered again before another command can be given.

TABLE 1.8-2. Program Load Error Codes

E 94	Program Storage Area full
E 95	Program does not exist in flash
E 96	Storage Module not connected or wrong address
E 97	Data not encountered within 30 sec.
E 98	Uncorrectable errors detected
E 99	Wrong type of file or Editor Error

1.8.1 INTERNAL FLASH PROGRAM STORAGE

Several programs can be stored in the CR510 Flash Memory and later recalled and run using the ∗D Mode. The Flash Electrically Erasable Programmable Read Only Memory is non-volatile memory that can only be erased in 16K blocks. The CR510 has 128K of Flash EEPROM memory, one 16K block is reserved for storing extra programs.

When a program is loaded and compiled, it is saved as the active program. The active

program will be automatically loaded and run when the CR510 is powered up. (If a Storage Module with a program 8 is connected when the CR510 powers-up, the Storage Module program 8 will be loaded into the CR510 and become the active program.)

The active program can be stored in internal flash memory program storage with *D command 6 (Table 1.8-3). Programs can be retrieved with *D command 7 (Table 1.8-4).

TABLE 1.8-3 Storing Program in Internal Flash

Key entry	Display
*D	13:00
6A	06:00

You may now enter one of the following options:

xxA	Save active program as number xx, xx may be 1-98.
A B	Scroll forward and backward through saved program numbers. The numbers are displayed in the order saved.
99A99A	Clear all saved programs.
0A	Display number of bytes free in saved program area.

TABLE 1.8-4 Retrieving a Program from Internal Flash

Key entry	Display
*D	13:00
7A	07:00

You may now enter one of the following options:

xxA	Retrieve program number xx (the most recent xx saved). To have the program compile like *6 (no resetting of input locations, flags, or ports) press C (xx--) before A.
0A	Erase active program (i.e., load a blank program; memory allocation and Final Storage are reset).
A B	Scroll forward and backward through saved program numbers.

Scrolling through the program names begins with the oldest program. "A" advances to the next newer program, "B" backs up to the next older program. While scrolling, at any time typing in a number (xxA) will cause a save or a retrieve operation.

Each program saved takes up the memory required for the program + 6 bytes.

Flash memory can only be written to once before being erased. Because it can only be erased in 16K blocks, if one stored program is to be erased, all must be erased. To allow revising a program and storing it with the same number (name) as an earlier version, the same number can be used by more than one saved program. When retrieving a program, the programs are searched beginning with the last program saved; the most recently saved version will be retrieved. An older program with a duplicate name cannot be retrieved. When the flash program memory is full, all programs must be erased before any more can be added (error 94 will be displayed).

1.8.2 PROGRAM TRANSFER WITH STORAGE MODULE

Storage Modules can store up to eight separate programs. The Storage Module and Keyboard/Display or Modem/Terminal must both be connected to the CR510. After keying *D, the command 7N, is entered (N is the Storage Module address 1-8, Section 4.4.1). Address 1 will work with any Storage Module address; the CR510 will search for the lowest address Storage Module that is connected. The command to save, load, or clear a program and the program number (Table 1.8-5) is entered. After the operation is finished "13:0000" is displayed. Error 96 indicates that the Storage Module is not connected or the wrong address was given.

TABLE 1.8-5 Transferring a Program using a Storage Module

Key entry	Display	
∗D	13:00	
7NA	7N:00	(N is Storage Module address 1-8)

You may now enter one of the following options:

1x	Save Program x to Storage Module (x = 1-8)
2x	Load Program x from Storage Module (x = 1-8)
3x	Erase Program x in Storage Module (x = 1-8)

The datalogger can be programmed on power-up using a Storage Module. If a program is stored as program number 8, and the Storage Module is connected to the datalogger I/O at power-up, program number 8 is automatically loaded into the active program area of the datalogger and run.

1.8.3 FULL/HALF DUPLEX

The *D Mode can also be used to set communications to full or half duplex. The default is full duplex, which works best in most situations.

TABLE 1.8-6. Setting Duplex

Key entry	Display
∗D	13:00
9A	09:0x

If x=0 the CR510 is set for full duplex.
If x=1 the CR510 is set for half duplex.

You may now change the option:

0A	Set full duplex
1A	Set half duplex

1.8.4 SET DATALOGGER ID

Command 8 is used to set the datalogger ID. The ID can be moved to an input location with Instruction 117 and can then be sampled as part of the data.

TABLE 1.8-7 Setting Datalogger ID

Key Entry	Display
∗D	13:00
8A	08:0XXX

Where XXX are 0s or the current ID. You may now key in the ID (1-254, excluding 13).

1.8.5 SETTING POWERUP OPTIONS

Setting options for the Program on Powerup allows the user to specify what information to retain from when the datalogger was last on. This allows Flag/Port status, the User Timer, and the Input/Intermediate Storage to be cleared or not cleared.

Table 1.8-8. Setting Powerup Options

Key entry	Display
∗D	13:00
10A	10:0X

Where X is the powerup option currently selected. You may now change the option:

0A	Clears input locations, ports, flags, user timer, and intermediate storage locations.
1A	Clears intermediate storage only (leaves Input Storage, Flags/Ports, and User Timer as is).
2A	Doesn't clear anything.
3A	Do not change power-up settings.

SECTION 2. INTERNAL DATA STORAGE

2.1 FINAL STORAGE AREAS, OUTPUT ARRAYS, AND MEMORY POINTERS

Final Storage is the memory where final processed data are stored. Final Storage data are transferred to your computer or external storage peripheral.

The size of Final Storage is expressed in terms of memory locations; one memory location is two bytes. A low resolution data point (4 decimal characters) occupies one memory location (2 bytes), whereas a high resolution data point (5 decimal characters) requires two memory locations (4 bytes). Table 1.5-1 shows the default allocation of memory locations to Program, Input, Intermediate, and the two Final Storage areas. The ∗A Mode is used to reallocate memory or erase Final Storage (Section 1.5).

The default size of Final Storage with standard memory is 62280 low resolution memory locations.

Final Storage can be divided into two parts: Final Storage Area 1 and Final Storage Area 2.

Final Storage Area 1 is the default storage area and the only one used if the operator does not specifically allocate memory to Area 2.

Two Final Storage Areas may be used to:

1. Output different data to different devices.

2. Separate archive data from real time display data. In other words, you can record a short time history of real time data and separately record long term, archive data.

3. Record both high speed data (fast recording interval) and slow data without having the high speed data write over the slow data.

Each Final Storage Area can be represented as ring memory (Figure 2.1-1) on which the newest data are written over the oldest data.

The Data Storage Pointer (DSP) is used to determine where to store each new data point in the Final Storage area. The DSP advances to the next available memory location after each new data point is stored.

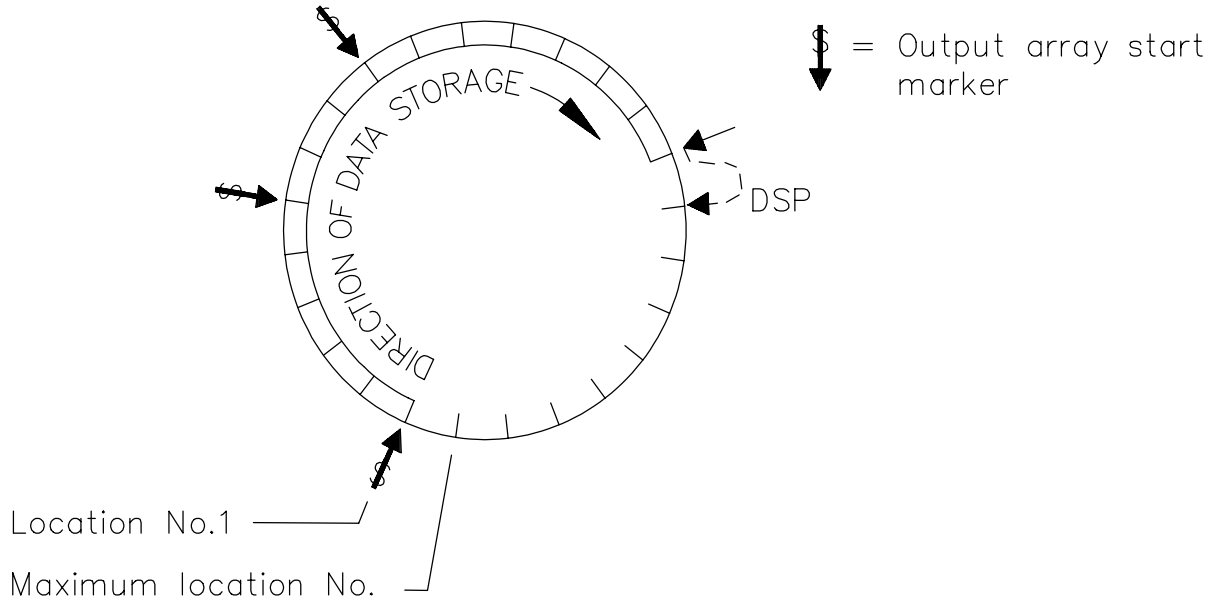


FIGURE 2.1-1. Ring Memory Representation of Final Data Storage

Output Processing Instructions store data into Final Storage only when the Output Flag is set. The string of data stored each time the Output Flag is set is called an **OUTPUT ARRAY**. The first data point in the output array is a 3 digit **OUTPUT ARRAY ID**. This ID number is set in one of two ways:

1. In the default condition, the ID consists of the program table number and the Instruction Location Number of the instruction which set the Output Flag for that particular array of data. For example, the ID of 118 in Figure 2.1-2 indicates that the 18th instruction in Table 1 set the Output Flag.
2. The output array ID can be set by the user with the second parameter of Instruction 80 (Section 11). The ID can be set to any positive integer up to 511. This option allows the user to make the output array ID independent of the programming. The program can be changed (instructions added or deleted) without changing the output array ID. This avoids confusion during data reduction, especially on long term projects where program changes or updates are likely.

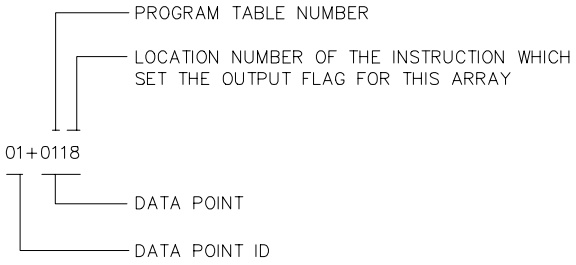


FIGURE 2.1-2. Output Array ID

> **NOTE:** If Instruction 80 is used to designate the active Final Storage Area and parameter 2 is 0, the output array ID is determined by the position of Instruction 80 or by the position of the instruction setting the Output Flag, whichever occurs last.

A start-of-array marker ($ in Figure 2.1-1) is written into Final Storage with the Output Array ID. This marker is used as a reference point from which to number the data points of the output array. The start of array marker occupies the same Final Storage location as the Array ID and is transparent for all user operations.

Data are stored in Final Storage before being transmitted to an external device. There are 4 pointers for each Final Storage Area which are used to keep track of data transmission. These pointers are:

1. Display Pointer (**DPTR**)
2. Printer Pointer (**PPTR**)
3. Telecommunications (Modem) Pointer (**MPTR**)
4. Storage Module Pointer (**SPTR**)

The **DPTR** is used to recall data to the keyboard/ display. The positioning of this pointer and data recall are controlled from the keyboard (∗7 Mode).

The **PPTR** is used to control data transmission to a printer or other serial device. Whenever on-line printer transfer is activated (Instruction 96), data between the PPTR and DSP are transmitted. The PPTR may also be positioned via the keyboard for manually initiated data transmission (∗8 Mode).

The **MPTR** is used in transmitting data over a telecommunications interface. When telecommunications is first entered, the MPTR is set to the same location as the DSP. Positioning of the MPTR is then controlled by commands from the external calling device (Section 5.1).

The **SPTR** is used to control data transmission to a Storage Module. When on-line transfer is activated by Instruction 96, data is transmitted each time an output array is stored in Final Storage IF THE STORAGE MODULE IS CONNECTED TO THE CR510. If the Storage Module is not connected, the CR510 does not transmit the data nor does it advance the SPTR to the new DSP location. It saves the data until the Storage Module is connected. Then, during the next execution of Instruction 96, the CR510 outputs all of the data between the SPTR and the DSP and updates the SPTR to the DSP location (Section 4.1)

The SPTR may also be positioned via the keyboard for manually initiated data transfer to the Storage Module (∗8 Mode, Section 4.2).

> **CAUTION:** All memory pointers are set to the DSP location when the datalogger compiles a program. ALWAYS RETRIEVE UNCOLLECTED DATA BEFORE MAKING PROGRAM CHANGES.

2.2 DATA OUTPUT FORMAT AND RANGE LIMITS

Data are stored internally in Campbell Scientific's Binary Final Storage Format (Appendix B.2). Data may be sent to Final Storage in either LOW RESOLUTION or HIGH RESOLUTION format.

2.2.1 RESOLUTION AND RANGE LIMITS

Low resolution data is a 2 byte format with 4 significant digits and a maximum magnitude of +6999. High resolution data is a 4 byte format with 5 significant digits and a maximum possible output value of +99999 (see Table 2.2-1 below).

TABLE 2.2-1. Resolution Range Limits of CR510 Data

Resolution	Zero	Minimum Magnitude	Maximum Magnitude
Low	0.000	+0.001	+6999.
High	0.0000	+ .00001	+99999.

The resolution of the low resolution format is reduced to 3 significant digits when the first (left most) digit is 7 or greater. Thus, it may be necessary to use high resolution output or an offset to maintain the desired resolution of a measurement. For example, if water level is to be measured and output to the nearest 0.01 ft., the level must be less than 70 ft. for low resolution output to display the 0.01 ft. increment. If the water level was expected to range from 50 to 80 ft. the data could either be output in high resolution or could be offset by 20 ft. (transforming the range to 30 to 50 ft.).

The default for Final Storage is low resolution. Program instruction 78 can be used to change this to high resolution.

2.2.2 INPUT AND INTERMEDIATE STORAGE DATA FORMAT

While output data have the limits described above, the computations performed in the CR510 are done in floating point arithmetic. In Input and Intermediate Storage, the numbers are stored and processed in a binary format with a 23 bit binary mantissa and a 6 bit binary exponent. The largest and smallest numbers that can be stored and processed are 9×10^{18} and 1×10^{-19}, respectively. The size of the number determines the resolution of the arithmetic. A rough approximation of the resolution is that it is better than 1 in the seventh digit. For example, the resolution of 97,386,924 is better than 10. The resolution of 0.008673192 4 is better than 0.000000001.

A precise calculation of the resolution of a number may be determined by representing the number as a mantissa between .5 and 1 multiplied by 2 raised to some integer power. The resolution is the product of that power of 2 and 2^{-24}. For example, representing 478 as .9336 $*$ 2^9, the resolution is $2^9 * 2^{-24} = 2^{-15} = 0.0000305$. A description of Campbell Scientific's floating point format may be found in the description of the J and K Telecommunications Commands in Appendix B.

2.3 DISPLAYING STORED DATA ON KEYBOARD/DISPLAY - *7 MODE

(Computer/terminal users refer to Section 5 for instructions on entering the Remote Keyboard State.)

Final Storage may be displayed by using the *7 Mode. Key *7.

If you have allocated memory to Final Storage Area 2, the display will show:

07:00

Select which Storage Area you wish to view:

00 or 01 = Final Storage Area 1
02 = Final Storage Area 2

If no memory has been allocated to Final Storage Area 2, this first window will be skipped.

The next window displays the current DSP location. Pressing A advances you to the Output array ID of the oldest Array in the Storage Area. To locate a specific Output Array, enter a location number that positions the Display Pointer (DPTR) behind the desired data and press the "A" key. If the location number entered is in the middle of an Output Array, the DPTR is automatically advanced to the first data point of the next Output Array. Repeated use of the "A" key advances through the Output Array, while use of the "B" key backs the DPTR through memory.

The memory location of the data point is displayed by pressing the "#" key. At this point,

another memory location may be entered, followed by the "A" key to jump to the start of the Output Array equal to or just ahead of the location entered. Whenever a location number is displayed by using the "#" key, the corresponding data point can be displayed by pressing the "C" key.

The same element in the next Output Array with the same ID can be displayed by hitting #A. The same element in the previous array can be displayed by hitting #B. If the element is 1 (Array ID), then #A advances to the next array and #B backs up to the previous array. #0A backs up to the start of the current array.

The keyboard commands used in the ∗7 Mode are summarized in Table 2.3-1.

Advancing the DPTR past the Data Storage Pointer (DSP) displays the oldest data point. Upon entering the ∗7 Mode, the oldest Output Array can be accessed by pressing the "A" key.

TABLE 2.3-1. ∗7 Mode Command Summary

Key	Action
A	Advance to next data point
B	Back-up to previous data point
#	Display location number of currently displayed data point value
C	Display value of current location
# A	Advance to same element in next Output Array with same ID
# B	Back-up to same element in previous Output Array with same ID
# 0 A	Back-up to the start of the current Final Data Storage Array
*	Exit ∗7 Mode

SECTION 3. INSTRUCTION SET BASICS

The instructions used to program the CR510 are divided into four types: Input/Output (I/O), Processing, Output Processing, and Program Control. I/O Instructions are used to make measurements and store the readings in input locations or to initiate analog or digital port output. Processing Instructions perform mathematical operations using data from Input Storage locations and place the results back into specified Input Storage locations. Output Processing Instructions provide a method for generating time or event dependent data summaries from processed sensor readings residing in specified Input Storage locations. Program Control Instructions are used to direct program execution based on time and or conditional tests on input data and to direct output to external devices.

Instructions are identified by a number. There are a fixed number of parameters associated with each instruction to give the CR510 the information required to execute the instruction. The set of instructions available in the CR510 is determined by the CR510 Operating System.

3.1 PARAMETER DATA TYPES

There are 3 different data types used for Instruction parameters: Floating Point (FP), 4 digit integers (4), and 2 digit integers (2). The parameter data type is identified in the listings of the instruction parameters in Sections 9-12. Different data types are used to allow the CR510 to make the most efficient use of its memory.

Floating Point parameters are used to enter numeric constants for calibrations or mathematical operations. While it is only possible to enter 5 digits (magnitude +.00001 to +99999.), the internal format has a much greater range ($1x10^{-19}$ to $9x10^{18}$, Section 2.2.1). Instruction 30 can be used to enter a number in scientific notation into an input location.

3.2 REPETITIONS

The repetitions parameter on many of the I/O, Processing, and Output Processing Instructions is used to repeat the instruction on a number of sequential Input Channels or Input Storage locations. For example, if you are making 2 differential voltage measurements on the same voltage range, wire the inputs to sequential channels and enter the Differential Voltage Measurement Instruction once with 2 repetitions, rather than entering 2 separate measurement instructions. The instruction will make 2 measurements starting on the specified channel number and continuing through the other differential channel. The results will be stored in the specified input location and the next succeeding input location. Averages for both measurements can be calculated by entering the Average Instruction with 2 repetitions.

When several of the same type of measurements will be made, but the calibrations of the sensors are different, it requires less time to make the measurements using one measurement with repetitions and then apply the calibrations with a scaling array (Inst. 53) than it does to enter the instruction several times in order to use a different multiplier and offset. This is due to set up and calibration time for each measurement instruction. However, if time is not a constraint, separate instructions may make the program easier to follow.

3.3 ENTERING NEGATIVE NUMBERS

After keying in a number, press C or "-" to change the number's sign. On floating point numbers a minus sign (-) will appear to the left of the number. Excitation voltages in millivolts for I/O Instructions are 4 digit integers; when C is keyed 2 minus signs (--) will appear to the right of the number indicating a negative excitation. Even though this display is the same as that indicating an indexed input location, (Section 3.4) there is no indexing effect on excitation voltage.

3.4 INDEXING INPUT LOCATIONS

When used within a loop, the parameters for input locations can be Indexed to the loop counter. The loop counter is added to the indexed value to determine the actual Input Location the instruction acts on. Normally the loop counter is incremented by 1 after each pass through the loop. Instruction 90, Step

Loop Index, allows the increment step to be changed. See Instructions 87 and 90, Section 12, for more details.

To index an input location (4 digit integer), C or "-" is pressed after keying the value but before entering the parameter. Two minus signs (--) will be displayed to the right of the parameter.

3.5 VOLTAGE RANGE AND OVERRANGE DETECTION

The voltage RANGE code parameter on Input/Output Instructions is used to specify the full scale range of the measurement and the integration period for the measurement (Table 3.5-1).

The full scale range selected should be the smallest that will accommodate the full scale output of the sensor being measured. Using the smallest possible range will result in the best resolution for the measurement.

Four different integration sequences are possible. The relative immunity of the integration sequences to random noise is: 60 Hz rej. = 50 Hz rej. > 2.72ms integ. > 272 µs integ. The 60 Hz rejection integration rejects noise from 60 Hz AC line power. The 50 Hz rejection is for countries whose electric utilities operate at 50 Hz (Section 13.1).

When a voltage input exceeds the range programmed, the value which is stored is set to the maximum negative number and displayed as -99999 in high resolution or -6999 in low resolution.

An input voltage greater than +5 volts on one of the analog inputs will result in errors and possible overranging on the other analog inputs. Voltages greater than 16 volts may permanently damage the CR510.

> **NOTE**: Voltages in excess of 5.5 volts applied to a control port can cause the CR510 to malfunction.

3.6 OUTPUT PROCESSING

Most Output Processing Instructions have both an Intermediate Data Processing operation and a Final Data Processing operation. For example, when the Average Instruction, 71, is initiated, the intermediate processing operation increments a sample count and adds each new Input Storage value to a cumulative total residing in Intermediate Storage. When the Output Flag is set, the final processing operation divides the cumulative total by the number of samples to find the average. The average is then stored in final storage and the cumulative total and number of samples are set to zero in Intermediate Storage.

Final Storage Area 1 (Sections 1.5, 2.1) is the default destination of data output by Output Processing Instructions. Instruction 80 may be used to direct output to either Final Storage Area 2 or to Input Storage.

Output Processing Instructions requiring intermediate processing sample the specified input location(s) each time the Output Instruction is executed, NOT each time the location value is updated by an I/O Instruction. For example: Suppose a temperature measurement is initiated by Table 1 which has an execution interval of 1 second.

TABLE 3.5-1. Input Voltage Ranges and Codes

Range Code				Full Scale Range	Resolution*
Slow 2.72ms Integ.	Fast 250 us Integ.	60 Hz Reject.	50 Hz Reject.		
1	11	21	31	±2.5 mV	0.33 µV
2	12	22	32	±7.5 mV	1.0 µV
3	13	23	33	±25 mV	3.33 µV
4	14	24	34	±250 mV	33.3 µV
5	15	25	35	±2500 mV	333. µV

***** *Differential measurement, resolution for single-ended measurement is twice value shown.*

The instructions to output the average temperature every 10 minutes are in Table 2 which has an execution interval of 10 seconds. The temperature will be measured 600 times in the 10 minute period, but the average will be the result of only 60 of those measurements because the instruction to average is executed only one tenth as often as the instruction to make the measurement.

Intermediate Processing can be disabled by setting Flag 9 which prevents Intermediate Processing without actually skipping over the Output Instruction.

All of the Output Processing Instructions store processed data values when and only when the Output Flag is set high (Section 3.7.1). The Output Flag (Flag 0) is set high at desired intervals or in response to certain conditions by using an appropriate Program Control Instruction (Section 12).

3.7 USE OF FLAGS: OUTPUT AND PROGRAM CONTROL

There are 10 flags which may be used in CR510 programs. Two of the flags are dedicated to specific functions: Flag 0 causes Output Processing Instructions to write to Final Storage, and Flag 9 disables intermediate processing. Flags 1-8 may be used as desired in programming the CR510. Flags 0 and 9 are automatically set low at the beginning of each execution of the program table. Flags 1-8 start out low when a program is compiled with ∗0 and remain unchanged until acted on by a Program Control Instruction or until manually toggled from the ∗6 Mode.

TABLE 3.7-1. Flag Description

Flag 0	-	Output Flag
Flag 1 to 8	-	User Flags
Flag 9	-	Intermediate Processing Disable Flag

Flags are set with Program Control Instructions. The Output Flag (Flag 0) and the intermediate programming disable flag (Flag 9) will always be set low if the set high condition fails. The status of flags 1-8 does not change when a conditional test is false.

3.7.1 THE OUTPUT FLAG

A group of processed data values is placed in Final Data Storage by Output Processing Instructions when the Output Flag (Flag 0) is set high. This group of data is called an Output Array. The Output Flag is set using Program Control Instructions according to time or event dependent intervals specified by the user. The Output Flag is set low at the beginning of each execution of the program table.

Output is most often desired at fixed intervals; this is accomplished with Instruction 92, If Time. Output is usually desired on the even interval, so Parameter 1, time into the interval, is 0. The time interval (Parameter 2), in minutes, is how often output will occur; i.e., the Output Interval. The command code (Parameter 3) is 10, causing Flag 0 to be set high. The time interval is synchronized to 24 hour time; output will occur on each integer multiple of the Output Interval starting from midnight (0 minutes). If the Output Interval is not an even divisor of 1440 minutes (24 hours), the last output interval of the day will be less than the specified time interval. Output will occur at midnight and will resume synchronized to the new day. Instruction 92 is followed in the program table by the Output Instructions which define the Output Array desired.

Each group of Output Processing Instructions creating an Output Array is preceded by a Program Control Instruction that sets the Output Flag high.

> **NOTE**: If the Output Flag is already set high and the test condition of a subsequent Program Control Instruction acting on Flag 0 fails, the flag is set low. This eliminates entering another instruction to specifically reset the Output Flag before proceeding to another group of Output Instructions with a different output interval.

3.7.2 THE INTERMEDIATE PROCESSING DISABLE FLAG

The Intermediate Processing Disable Flag (Flag 9) suspends intermediate processing when it is set high. This flag is used to restrict sampling for averages, totals, maxima, minima, etc., to times when certain criteria are met. The flag is automatically set low at the beginning of each execution of the program table.

TABLE 3.7-2. Example of the Use of Flag 9

1: If time is (P92)		
1:	0	Minutes (Seconds --) into a
2:	10	Interval (same units as above)
3:	10	Set Ouptut Flag High (Flag 0)
2: If (X⇔F) (P89)		
1:	14	X Loc [Wind_spd]
2:	4	<
3:	4.5	F
4:	19	Set Intermed. Proc. Disable Flag High (Flag 9)
3: Histogram (P75) ; See Section 11 for details of this intruction.		
4: Do (P86) ; Required when additional output processing follows		
1:	29	Set Intermed. Proc. Disable Flag Low (Flag 9)
5: Maximum (P73)		
1:	1	Reps
2:	00	Time Option
3:	14	Loc [Wind_spd]

As an example, suppose it is desired to obtain a wind speed rose incorporating only wind speeds greater than or equal to 4.5 m/s. The wind speed rose is computed using the Histogram Instruction 75, and wind speed is stored in input location 14, in m/s. Instruction 89 is placed just before Instruction 75 and is used to set Flag 9 high if the wind speed is less than 4.5 m/s:

> **NOTE**: Flag 9 is automatically reset the same as Flag 0. If the intermediate processing disable flag is already set high and the test condition of a subsequent Program Control Instruction acting on Flag 9 fails, the flag is set low. This feature eliminates having to enter another instruction to specifically reset Flag 9 low before proceeding to another group of test conditions.

3.7.3 USER FLAGS

Flags 1-8 are not dedicated to a specific purpose and are available to the user for general programming needs. The user flags can be manually toggled from the keyboard in the ∗6 Mode (Section 1.3). By inserting the flag test (Instruction 91) at appropriate points in the program, the user can use the ∗6 Mode to manually direct program execution.

3.8 PROGRAM CONTROL LOGICAL CONSTRUCTIONS

Most of the Program Control Instructions have a command code parameter which is used to specify the action to be taken if the condition tested in the instruction is true. Table 3.8-1 lists these codes.

TABLE 3.8-1. Command Codes

0	Go to end of program table[2]
1-9, 79-99	Call Subroutine 1-9, 79-99[1]
10-19	Set Flag 0-9 high
20-29	Set Flag 0-9 low
30	Then Do
31	Exit loop if true
32	Exit loop if false
41	Set Port 1 high
51	Set Port 1 low
61	Toggle Port 1
71	Pulse Port 1

[1] 98 is a special subroutine which can be called by Control port 2 going high; see Instruction 85 for details (Section 12).

[2] If this command is executed while in a subroutine, execution jumps directly to the end of the table that called the subroutine.

3.8.1 IF THEN/ELSE COMPARISONS

Program Control Instructions can be used for If then/else comparisons. When Command 30 (Then do) is used with Instructions 83 or 88-92, the If Instruction is followed immediately by instructions to execute if the comparison is true. The Else Instruction (94) is optional and is followed by the instructions to execute if the comparison is false. The End Instruction (95) ends the If then/else comparison and marks the beginning of the instructions that are executed regardless of the outcome of the comparison (see Figure 3.8-1).

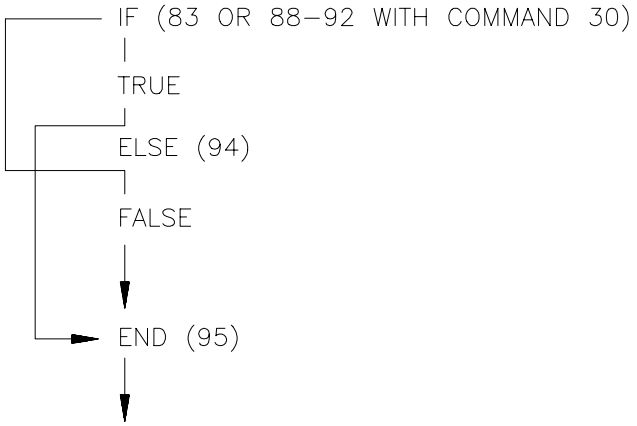


FIGURE 3.8-1. If Then/Else Execution Sequence

;Logical ELSE construction example:

```
;Check condition
1:  If (X⇔F) (P89)
   1:        1          X Loc [ DO_ppm  ]
   2:        4          <
   3:        3.5        F
   4        30          Then Do

;Instruction(s) to execute if above condition is true
2:  Do (P86)
   1:        41         Set Port 1 High

3:  Else (P94)

;Instruction(s) to execute if above condition is false
4:  Do (P86)
   1:        51         Set Port 1 Low

5:  End (P95)
```

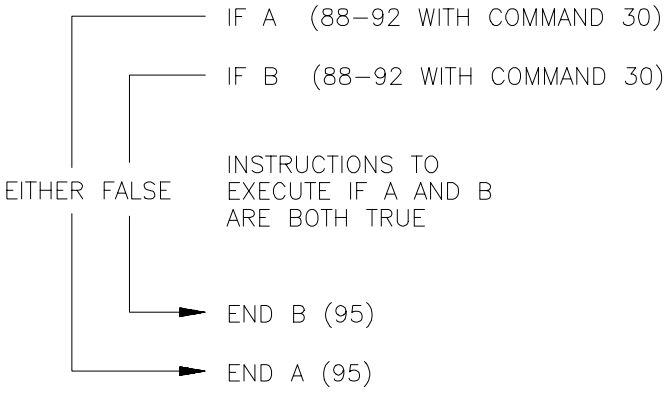


FIGURE 3.8-2. Logical AND Construction

;Logical AND construction example:

```
;Check first condition
6:  If (X⇔F) (P89)
   1:        1          X Loc [ DO_ppm ]
   2:        4          <
   3:        3.5        F
   4:       30          Then Do

;AND check second condition
7:  If (X⇔F) (P89)
   1:        2          X Loc [ Counter ]
   2:        3          >=
   3:       10          F
   4:       30          Then Do

;Instruction(s) to execute if both conditions are true
8:  Do (P86)
   1:       41  Set Port 1 High

9:  End (P95)

10:  End (P95)
```

If Then/Else comparisons may be nested to form logical AND or OR branching. Figure 3.8-2 illustrates an AND construction. If conditions A and B are true, the instructions included between IF B and the first End Instruction will be executed.

If either of the conditions is false, execution will jump to the corresponding End Instruction, skipping the instructions between.

A logical OR construction is also possible. Figure 3.8-3 illustrates the instruction sequence that will result in subroutine X being executed if either A or B is true.

;Logical OR construction example:

```
11:  If (X⇔F) (P89)
  1:        1          X Loc [ DO_ppm ]
  2:        4          <
  3:        3.5        F
  4:       30          Then Do

12:  Do (P86)
  1:       41          Set Port 1 High

13:  Else (P94)

14:  If (X⇔F) (P89)
  1:        2          X Loc [ counter ]
  2:        3          >=
  3:       10          F
  4:       30          Then Do

15:  Do (P86)
  1:       41          Set Port 1 High

16:  End (P95)

17:  End (P95)
```

A logical OR can also be constructed by setting a flag high if a comparison is true. (The flag is cleared or set low before making comparisons.) After all comparisons have been made, execute the desired instructions if the flag is set high.

The Begin Case Instruction 93 and If Case Instruction 83 allow a series of tests on the value in an input location. The case test is started with Instruction 93 which specifies the location to test. A series of Instruction 83s are then used to compare the value in the location with fixed values. When the value in the input location is less than the fixed value specified in Instruction 83, the command in that Instruction 83 is executed, and execution branches to the END Instruction 95 which closes the case test (see Instruction 93, Section 12).

;CASE Logic construction example:

```
18:  CASE (P93)
  1:        3          Case Loc [ Reading ]

19:  If Case Location < F (P83)
  1:        1.8        F
  2:       30          Then Do

;See Section 9 for details of this Instruction
20:  AC Half Bridge (P5)

21:  End (P95)

22:  If Case Location < F (P83)
  1:        9.25       F
  2:       30          Then do

;See Section 9 for details of this Instruction
23:  Full Bridge (P6)

24:  End (P95)

25:  If Case Location < F (P83)
  1:      280          F
  2:       30          Then do

;See Section 9 for details of this Instruction
26:  Full Bridge (P6)

27:  End (P95)

28:  End (P95)
```

3.8.2 NESTING

A branching or loop instruction which occurs before a previous branch or loop has been closed is nested. The maximum nesting level is 11 deep. Loop Instruction 87 and Begin Case Instruction 93 both count as 1 level. Instructions 83, 86, 88, 89, 91, and 92 each count as one level when used with the Command "30" which is the "Then Do" command. Use of Else, Instruction 94, also counts as one nesting level each time it is used. For example, the AND construction above is nested 2 deep while the OR construction is nested 3 deep.

Subroutine calls do not count as nesting with the above instructions, though they have their own nesting limit (maximum of 6, see Instruction 85, Section 12). Branching and loop nesting start at zero in each subroutine.

Any number of groups of nested instructions may be used in any of the three Programming Tables. The number of groups is only restricted by the program memory available.

3.9 INSTRUCTION MEMORY AND EXECUTION TIME

Each instruction requires program memory and uses varying numbers of Input, Intermediate, and Final Storage locations. Tables 3.9-1 to 3.9-4 list the memory used by each instruction and the approximate time required to execute it.

When attempting to make a series of measurements and calculations at a fast rate, it is important to examine the time required for the automatic calibration sequence and possibly make use of the program controlled calibration, Instruction 24. Section 13.7 describes the calibration process.

> **NOTE:** EDLOG generates a “trace” file with the extension .PTI which shows the estimated program execution time.

TABLE 3.9-1. Input/Output Instruction Memory and Execution Times

R = No. of Reps.

INSTRUCTION		MEMORY		EXECUTION TIME (ms) MEASUREMENT RANGE							
		INPUT LOC.	PROG. BYTES	1-4 or NA	5	11-14	15	21-24	25	31-34	35
1	VOLT (SE)	R	15	4.6 + 5.2R	2.0 + 2.8R	2.1 + 2.8R	1.8 + 2.6R	12.8 + 13.5R	11.6 + 12.3R	14.5 + 15.2R	12.1 + 12.8R
2	VOLT (DIFF)	R	15	0.6 + 9.1R	0.8 + 4.1R	0.5 + 4.2R	-0.6 + 4.1R	0.6 + 25.8R	0.0 + 21.0R	0.5 + 29.1R	-0.2 + 24.3R
3	PULSE	R	15	1.3 + 0.9R							
4	EX-DEL-SE	R	20	4.7 + 5.3R	2.3 + 2.9R	2.3 + 2.8R	2.1 + 2.7R	13.1 + 13.6R	10.7 + 11.2R	14.8 + 15.2R	12.3 + 12.9R
	above w/o delay; w/delay: F + R(Tr + delay), where F is the fixed time and Tr is the time/rep from above.										
5	AC HALF BR	R	18	6.9 + 9.6R	4.5 + 4.8R	4.5 + 4.7R	4.3 + 4.4R	15.2 + 26.3R	12.8 + 21.5R	17.0 + 29.6R	14.5 + 24.8R
6	FULL BR	R	18	3.3 + 17.4R	3.3 + 7.8R	3.3 + 7.5R	3.3 + 6.7R	0.6 + 52.7R	0.7 + 43.0R	1.9 + 59.2R	0.7 + 49.7R
7	3W HALF BR	R	18	6.6 + 21.3R	4.2 + 11.4R	4.1 + 11.6R	3.9 + 10.6R	13.6 + 56.4R	10.1 + 46.5R	15.3 + 63.1R	12.9 + 53.1R
8	EX-DEL-DIFF	R	20	4.7 + 5.3R	2.3 + 2.9R	2.4 + 2.9R	2.1 + 2.7R	13.1 + 13.6R	10.7 + 11.2R	14.8 + 15.3R	12.3 + 12.9R
	above w/o delay; w/delay: F + 2R(Tr + delay), where F is the fixed time and Tr is the time/rep from above.										
9	FULL BR-MEX	R	19	1.4 + 37.0R	1.5 + 17.5R	1.3 + 17.1R	1.6 + 15.3R	-2.3 + 107.4R	-2.2 + 87.9R	3.4 + 118.9R	-2.0 + 101.1R
	V1 on range 5, V2 on:			1.6 + 27.2R	1.5 + 17.5R	1.6 + 17.3R	1.5 + 16.4R	0.3 + 62.4R	-0.9 + 52.6R	-0.3 + 69.1R	0.2 + 59.3R
	V1 on range 15, V2 on:			1.5 + 26.2R	1.6 + 16.4R	1.6 + 16.2R	1.6 + 15.3R	0.3 + 61.3R	-0.9 + 51.6R	0.3 + 68.1R	0.3 + 58.2R
10	BATT. VOLT	1	4	8.3							
11	TEMP-(107)	R	15	2.3 + 7.6R				14 + 48.4R		14.7 + 53.4R	
12	RH (207)	R	17	2.3 + 9.6R							
16	TEMP-RTD	R	15	1.5 + 2.0R							
17	TEMP-INTERNL	1	4	6.2							
18	TIME	1 or 5	7	1.9 (options 0, 1, 2) 4.2 (option 3)							
19	SIGNATURE	1	4	3600							
20	PORT SET	0	6	11.3							
21	PULSE PORT	0	5	1.2 + 10H; where H is hundredths of a second							
22	EXCIT-DEL	0	9	0.5 + delay (ms)							
24	CALIBRATION	19	4	2614, 25.0 when only saving results of automatic calibration.							
25	READ PORTS	1	6	0.5							
26	TIMER	1 or 0	4	0.4 to reset, 0.9 to load into location							
27	PER.AVG.	R	19	period of signal * (No. cycles + 1.5), or time limit + 2							
28	VIB.WIRE	R	21	170 + period of signal * (No. cycles + 1.5), or time limit + 172							
29	INW PS9105	2	88	153							
105	SDI-12 REC	X	18	sensor dependent; where X depends on SDI-12 sensor							
106	SDI-12 SEN	0	10	ttt in parameter 2, this is sensor dependent							
114	SET TIME	3-5	5	994.6							
117	READ ID	0	4	0.8							

TABLE 3.9-2. Processing Instruction Memory and Execution Times R = No. of Reps.

INSTRUCTION	INPUT LOC.	MEMORY INTER. LOC.	PROG. BYTES	EXECUTION TIME (ms)
30 Z=F	1	0	9	1.0 + 0.2 ∗ exponent
31 Z=X	1	0	6	0.7
32 Z=Z+1	1	0	4	0.8
33 Z=X+Y	1	0	8	1.2
34 Z=X+F	1	0	10	1.1
35 Z=X-Y	1	0	8	1.2
36 Z=X∗Y	1	0	8	1.5
37 Z=X∗F	1	0	10	1.2
38 Z=X/Y	1	0	8	3.0
39 Z=SQRT(X)	1	0	6	9.0
40 Z=LN(X)	1	0	6	8.4
41 Z=EXP(X)	1	0	6	6.6
42 Z=1/X	1	0	6	2.9
43 Z=ABS(X)	1	0	6	1.0
44 Z=FRAC(X)	1	0	6	1.1
45 Z=INT(X)	1	0	6	1.4
46 Z=X MOD F	1	0	10	3.5
47 $Z=X^Y$	1	0	8	14.9
48 Z=SIN(X)	1	0	6	7.3
49 SPA. MAX	1 or 2	0	8	2.7 + 0.6 * swath
50 SPA. MIN	1 or 2	0	8	2.3 + 0.6 * swath
51 SPA. AVG	1	0	8	3.0 + 0.6 * swath
52 RUNNING AVG	1	(R ∗ par 4) + R + 1	11	2.1 + 3.7R
53 A∗X+B	4	0	36	3.5
54 BLOCK MOVE	R	0	10	0.3 + 0.2R
55 POLYNOMIAL	R	0	31	1.2 + (2.0 + 0.4 ∗ order)R
56 SAT. VP	1	0	6	4.5
58 LP FILTER	R	R + 1	13	1.0 + 2.2R
59 X/(1-X)	1	0	9	0.4 + 1.2R
61 INDIR. MOVE	1	0	6	0.6
63 PARA.EXTN.	0	0	10	0.2
65 BULK LOAD	8	0	36	4.5
66 ARC TAN	1	0	8	8.7
68 4 DIG PARA. EXTN.	0	0	8	0.7

TABLE 3.9-3. Output Instruction Memory and Execution Times R = No. of Reps.

INSTRUCTION	INTER. LOC.	MEM. FINAL VALUES[1]	PROG. BYTES	EXECUTION TIME (ms) FLAG 0 LOW	FLAG 0 HIGH
69 WIND VECTOR	2+9R	(2, 3, or 4)R	12		
		Options 00, 01, 02		4.0 + 17.4R	3.3 + 70.7R
		Options 10, 11, 12		3.0 + 16.3R	4.0 + 75R
70 SAMPLE	0	R	6	0.2	0.5+ 0.4R
71 AVERAGE	1+R	R	7	0.6+ 1.1R	1.5+ 4.4R
72 TOTALIZE	R	R	7	0.6+ 1.0R	1.0+ 1.7R
73 MAXIMIZE	(1 or 2)R	(1,2, or 3)R	8	1.0+ 1.2R	5.0+ 3.0R
74 MINIMIZE	(1 or 2)R	(1,2, or 3)R	8	0.8+ 1.2R	5.0+ 3.0R
75 HISTOGRAM	1+bins∗R	bins∗R	24	0.8+ 3.4R	1.6+ 5.2 + (1.4 ∗ bins)R
77 REAL TIME	0	1 to 4	4	0.2	3.8
78 RESOLUTION	0	0	3	0.4	0.4
79 SMPL ON MM	R	R	7	0.4	1.7 + 1.1R
80 STORE AREA[1]	0	0	7	0.3	0.3
82 STD. DEV.	1+3R	R	7	1.5+ 2.0R	2.9+ 2.1R

[1]Output values may be sent to either Final Storage area or Input Storage with Instruction 80.

TABLE 3.9-4. Program Control Instruction Memory and Execution Times

INSTRUCTION	MEMORY INTER. LOC.	PROG. BYTES	EXECUTION TIME (ms)				
83 IF CASE <F	0	10	0.5				
85 LABEL SUBR.	0	3	0				
86 DO	0	6	0.3				
87 LOOP	1	10	0.3				
88 IF X<=>Y	0	11	0.8				
89 IF X<=>F	0	13	0.6				
90 LOOP INDEX	0	3	0.7				
91 IF FLAG/PORT	0	7	0.4				
92 IF TIME	1	12	0.4				
93 BEGIN CASE	1	8	0.3				
94 ELSE	0	4	0.2				
95 END	0	4	0.2				
96 SERIAL OUT	0	3	Option:	0x	1x	2x	3x
			Time:	0.4	1.8	2.1	0.9
			Option:	4x	5x	6x	7x
			Time:	1.7	1.9	0.7	0.5
97 INIT.TELE.	7	17	2.3				
120 GOES SAT	0 or 2	5	8000				
121 ARGOS SAT	0	8	250				

3.10 ERROR CODES

There are four types of errors flagged by the CR510: Compile, Run Time, Editor, and ∗D Mode. Compile errors are errors in programming which are detected once the program is entered and compiled for the first time (∗0, ∗6, or ∗B Mode entered). If a programming error is detected during compilation, an E is displayed with the 2 digit error code. The Instruction Location Number of the Instruction which caused the error is displayed to the right of the error code (e.g., E23 105; 105 indicates that the fifth instruction in Table 1 caused error 23). Error 22, missing END, will indicate the location of the instruction which the compiler cannot match with an END instruction.

Run time errors are detected while the program is running. The number of the instruction being executed at the time the error is detected is displayed to the right of the error code (e.g., E09 06 indicates that an Instruction 6 in the program is attempting to store data in input locations beyond those allocated). Run time errors 9 and 31 are the result of programming errors. While E08 will display the number of the instruction that was being executed when the error occurred, it is unlikely that the instruction has anything to do with the error.

If there is a run time error in a table with a fast execution interval, the error may be written to the display so frequently that it seems the CR510 is not responding to the keyboard. Once the program is stopped, normal function will return. To stop the program some entry must be changed which requires recompiling (Section 1.1.4). For example, enter 0 for the execution interval of Table 1 (i.e., enter ∗1A0A as fast as possible). The program can easily be stopped by pressing any key while the CR10KD is displaying “HELLO” after applying power (turn the CR510 off and then on again). This delays program execution for about two minutes, allowing the program to be changed.

Error 8 is the result of a hardware and software "watchdog" that checks the processor state, software timers, and program related counters. The watchdog will attempt to reset the processor and program execution if it finds that the processor has bombed or is neglecting standard system updates, or if the counters are out of allowable limits. Error code 08 is flagged when the watchdog performs this reset. E08 is occasionally caused by voltage surges or transients. Frequent repetitions of E08 are indicative of a hardware problem or a software bug and should be reported to Campbell Scientific. The CR510 keeps track of the number of times (up to 99) that E08 has occurred. The number can be displayed and reset in the ∗B Mode (Section 1.6) or with the Telecommunications A command (Section 5.1).

Error 10 is displayed if the primary power drops below 9.6 volts. When this happens, the CR510 stops executing programs. The low voltage counter (∗B Window 9, Section 1.6) counts the number of times the voltage drops below 9.6 volts and displays a double dash (--) if the CR510 is currently in a low voltage shut down. Below approximately 8.5 volts the CR510 will not communicate with the CR10KD or modem, although there may be enough power to display characters on the CR10KD.

Editor errors are detected as soon as an incorrect value is entered and are displayed immediately. Only the error code is displayed. ∗D Mode errors indicate problems with saving or loading a program. Only the error code is displayed.

TABLE 3.10-1. Error Codes

Code	Type	Description
03	Editor	Program table full
04	Compile	Intermediate Storage full
05	Compile	Storage Area #2 not allocated
08	Run Time	CR510 reset by watchdog timer
09	Run Time	Insufficient Input Storage
10	Run Time	Low battery voltage
11	Editor	Attempt to allocate more Input or Intermediate Storage than is available
12	Compile	Duplicate ∗4 ID
20	Compile	SUBROUTINE encountered before END of previous subroutine
21	Compile	END without IF, LOOP or SUBROUTINE
22	Compile	Missing END
23	Compile	Nonexistent SUBROUTINE
24	Compile	ELSE in SUBROUTINE without IF
25	Compile	ELSE without IF

26	Compile	EXIT LOOP without LOOP
27	Compile	IF CASE without BEGIN CASE
30	Compile	IF and/or LOOP nested too deep
31	Run Time	SUBROUTINES nested too deep
32	Compile	Instruction 3 and interrupt subroutine use same port
40	Editor	Instruction does not exist
41	Editor	Incorrect execution interval
60	Compile	Insufficient Input Storage
92	Compile	Instruction 92, intervals in seconds: Time into Interval > 59 or Interval > 60
94	∗D MODE	Program Storage Area full
95	∗D MODE	Program does not exist in Flash memory
96	∗D MODE	Addressed device not connected or wrong address (see Table 1.8-2)
97	∗D MODE	Data not received within 30 seconds
98	∗D MODE	Uncorrectable errors detected
99	∗D MODE	Wrong file type or editor error

SECTION 4. EXTERNAL STORAGE PERIPHERALS

External data storage devices are used to provide a data transfer medium that the user can carry from the test site to the lab and to supplement the internal storage capacity of the CR510, allowing longer periods between visits to the site. The standard data storage peripheral for the CR510 is the Storage Module (Section 4.4). Output to a printer or related device is also possible (Section 4.3).

Data output to a peripheral device can take place ON-LINE (automatically, as part of the CR510's routine operation) or it can be MANUALLY INITIATED. On-line data transfer is accomplished with Instruction 96 (Section 4.1). Manual initiation is done in the ∗8 Mode (Section 4.2).

The CR510 can output data to multiple peripherals. The CR510 activates the peripheral it sends data to in one of two ways (Section 6.2):

1. *A specific pin in the 9-pin connector is dedicated to that peripheral; when that pin goes high, the peripheral is enabled. This is referred to as "PIN-ENABLED" or simply "ENABLED".*

2. *The peripheral is synchronously addressed by the CR510. This is referred to as "ADDRESSED".*

Modems are pin-enabled. Only one modem device may be connected to the CR510 at any one time. The CR510 considers the following devices to be pin-enabled modems: SC32A, SC932, short-haul, MD9, radio modems, and telephone modems except for voice modems.

The SM192, SM716, and CSM1 Storage Modules are addressed. The CR510 can tell when the addressed device is present. The CR510 will not send data meant for the Storage module if the Storage Module is not present (Section 4.4.2). Other addressed devices include the CR10KD and voice modems.

The ∗9 Mode (Section 4.5) allows the user to communicate directly with the Storage Module and to perform several functions, including review of data, battery test, review of Storage Module status, etc.

Cassette tape data storage is not supported by the CR510.

4.1 ON-LINE DATA TRANSFER - INSTRUCTION 96

All on-line data output to a peripheral device is accomplished with Instruction 96. (Instruction 96 can also be used to transfer data from one Final Storage Area to the other, Section 8.8, 12). This instruction must be included in the datalogger program for on-line data transfer to take place. Instruction 96 should follow the Output Processing Instructions, but only needs to be included once in the program table unless both Final Storage areas are in use. The suggested programming sequence is:

1. Set the Output Flag.
2. If both Final Storage Areas are in use or if you wish to set the Output Array ID, enter Instruction 80 (Section 11).
3. Enter the appropriate Output Processing Instructions.
4. Enter Instruction 96 to enable the on-line transfer of Final Storage data to the specified device. If outputting to more than one device, Instruction 96 must be entered separately for each device.
5. Repeat steps 2 through 4 if you wish to output data to the other Final Storage Area and the peripheral.

Instruction 96 has a single parameter which specifies the peripheral to send output to. Table 4.1-1 lists the output device codes.

TABLE 4.1-1. Output Device Codes for Instruction 96 and *8 Mode

Code	Device
	ADDRESSED PRINTER
1y	Printable ASCII
2y	Comma separated ASCII
3y	Binary
	PIN ENABLED PRINTER
4y	Printable ASCII
5y	Comma separated ASCII
6y	Binary
	y = BAUD RATE CODES 0 300 1 1200 2 9600 3 76,800
7N	Storage Module N (N=address, 1...8)
7N--	Output File Mark to Storage Module N
80	To the other Final Storage Area [Inst. 96 only], new data since last output
81	To the other Final Storage Area [Inst. 96 only], entire active Final Storage Area

The source of data for Instruction 96 is the currently active Final Storage Area as set by Instruction 80 (the default is Final Storage Area 1 at the beginning of each program table execution).

If the CR510 is using the 9-pin connector for other I/O tasks when Instruction 96 is executed, the output request is put in a queue and program execution continues. As the 9-pin connector becomes available, each device in the queue gets its turn.

An output request is not put in the queue if the same device is already in the queue. The data contained in the queue (and which determine a unique entry) are the device, baud rate (if applicable), and the Final Storage Area.

When an entry reaches the top of the queue, the CR510 sends all data accumulated since the last transfer to the device up to the location of the DSP at the time the device became active.

Printer output can be either pin-enabled or addressed. However, there is not a pin specifically dedicated to print enable. When a pin-enabled print output is specified, the SDE line, which is normally used in the addressing sequence, is used as a print enable. This allows some compatibility with the CR21, 21X, and CR7 dataloggers which have a Print Enable line. The pin-enabled print option will result in garbage being sent to the print peripheral if an addressed device is also connected to the CR510 (i.e., CR10KD, SM192 or SM716 etc.). The SDC99 Synchronous Device Interface can convert a print device to an Addressed peripheral (Section 6.2).

The STORAGE MODULE address is important only when using more than one Storage Module. One is a universal address which will find the Storage Module with lowest number address that is connected. If a Storage Module is not connected, the CR510 will not advance the SPTR (Section 2.1) and the Storage Module drops out of the queue until the next time Instruction 96 is executed. Section 4.4 contains specifics on the Storage Modules.

TABLE 4.2-1. *8 Mode Entries

Key	Display ID:DATA	Description
* 8	08:00	Key 1 or 2 for Storage Area. (This window is skipped if no memory has been allocated to Final Storage Area 2.)
A	01:XX	Key in Output Device Option. See Table 4.1-1.
A	02:XXXXX	Start of dump location. Initially the SPTR or PPTR location; a different location may be entered if desired.
A	03:XXXXX	End of dump location. Initially the DSP location; a different location may be keyed in if desired.
A	04:00	Ready to dump. To initiate dump, key any number, then A. While dumping, "04" will be displayed in the ID field and the location number in the Data field. The location number will stop incrementing when the dump is complete. (Any key aborts transmission after completion of the current data block.)

4.2 MANUALLY INITIATED DATA OUTPUT - ∗8 MODE

Data transfer to a peripheral device can be manually initiated in the ∗8 Mode. This process requires that the user have access to the CR510 through a terminal or the CR10KD. The ∗8 Mode allows the user to retrieve a specific block of data, on demand, regardless of whether or not the CR510 is programmed for on-line data output.

If external storage peripherals are not left on-line, the maximum time between collecting data must be calculated to ensure that data in Final Storage are not lost due to write-over. To calculate this time it is necessary to know: (1) the size of Final Storage, (2) the number of Output Arrays being generated, (3) the number of low and/or high resolution data points per Output Array, and (4) the rate at which Output Arrays are placed into Final Storage. When calculating the number of data points per Output Array, remember to add 1 data point per array for the Output Array ID.

For example, assume that 62,280 locations are assigned to Final Storage (∗A Mode), and that 1 Output Array, containing the Array ID (1 memory location), 9 low resolution data points (9 memory locations) and 5 high resolution data points (10 memory locations), is stored each hour. In addition, an Output Array with the Array ID and 5 high resolution data points (11 memory locations) is stored daily. This is a total of 491 memory locations per day ((20 x 24) + 11). 62,280 divided by 491 = 126.8 days. Therefore, the CR510 would have to be visited every 126 days to retrieve data, because write-over would begin on the 127th day. The site should be visited more frequently than this for routine maintenance. Thus data storage capacity would not be a factor in determining how frequently to visit the site.

The output device codes used with the ∗8 Mode are the same as those used with Instruction 96 (Table 4.1-1), with the exception of the option to transfer data from one Final Storage area to the other (80, 81). Table 4.2-1 lists the keystrokes required to initiate a ∗8 data dump.

4.3 PRINTER OUTPUT FORMATS

Printer output can be sent in binary Final Storage Format (Appendix C.2), Printable ASCII, or Comma Separated ASCII. These ASCII formats may also be used when data from the Storage Modules or Telecommunications are stored on disk with Campbell Scientific's datalogger support software.

4.3.1 PRINTABLE ASCII FORMAT

In the Printable ASCII format each data point is preceded by a 2 digit data point ID and a (+) or (–) sign. The ID and fixed spacing of the data points make particular points easy to find on a printed output. This format requires 10 bytes per data point to store on disk.

Figure 4.3-1 shows both high and low resolution data points in a 12 data point Output Array. The example data contains Day, Hour-Minute, and Seconds in the 2nd - 4th data points. REMEMBER! You must specifically program the CR510 to output the date and time values. The Output Array ID, Day, and Time are always 4 character numbers, even when high resolution output is specified. The seconds resolution is .125 seconds.

Each full line of data contains 8 data points (79 characters including spaces), plus a carriage return (CR) and line feed (LF). If the last data point in a full line is high resolution, it is followed immediately with a CR and LF. If it is low resolution, the line is terminated with a space, CR and LF. Lines of data containing less than 8 data points are terminated similarly after the last data point.

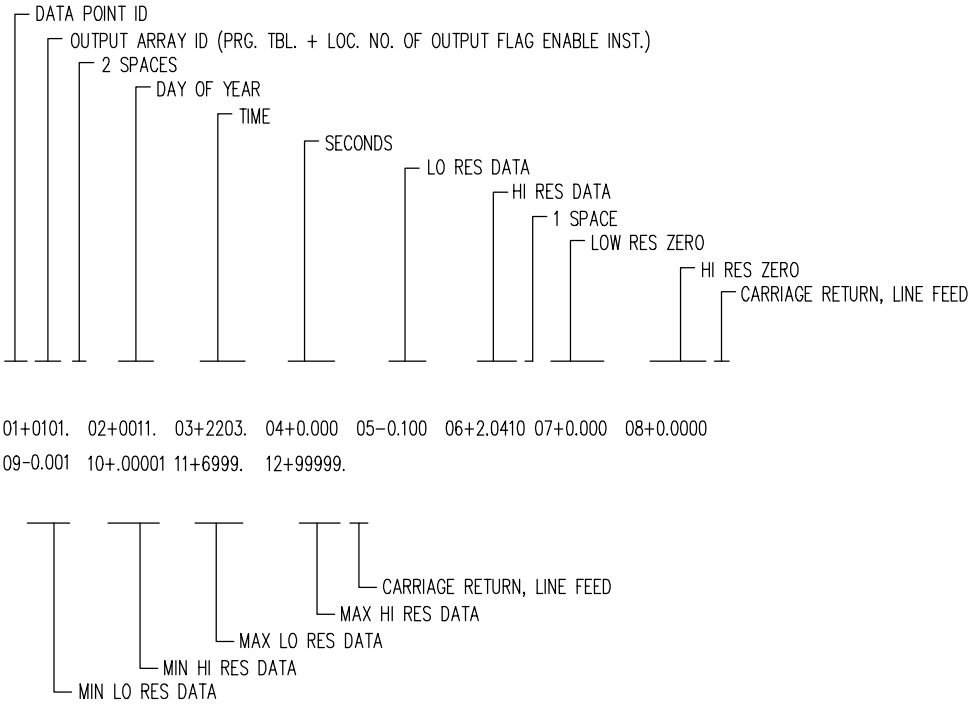


FIGURE 4.3-1. Example of CR510 Printable ASCII Output Format

4.3.2 COMMA SEPARATED ASCII

Comma Separated ASCII strips all IDs, leading zeros, unnecessary decimal points and trailing zeros, and plus signs. Data points are separated by commas. Arrays are separated by Carriage Return Line Feed. Comma Separated ASCII requires approximately 6 bytes per data point. Example:

1,234,1145,23.65,-12.26,625.9
1,234,1200,24.1,-10.98,650.3

4.4 STORAGE MODULE

The Storage Module stores data in battery backed RAM. Backup is provided by an internal lithium battery. The RAM is internal on the SM192/716 and on a PCMCIA card in the CSM1. Operating power is supplied by the CR510 over pin 1 of the 9-pin connector. Whenever power is applied to the 9-pin connector (after having been off), the Storage Module places a File Mark in the data (if a File Mark is not the last data point already in storage).

The File Mark separates data. For example, if you retrieve data from one CR510, disconnect the Storage Module and connect it to a second CR510, a File Mark is automatically placed in the data. This mark follows the data from the first CR510 but precedes the data from the second.

The SM192 has 192K bytes of RAM storage; the SM716 has 716K bytes. Both can be configured as either ring or fill and stop memory. The size of memory in the CSM1 depends on the PC Card used. The CSM1 is always fill and stop.

4.4.1 STORAGE MODULE ADDRESSING

The CSM1 does not support individual addresses. Use address 1 when sending data to the CSM1.

The SM192/716 Storage Modules can have individual addresses. Different addresses allow 1) up to 8 Storage Modules to be connected to the CR510 during on-line output, 2) different data to be output to different Modules, and 3) transfer of data from a Module that is left with the CR510 to a Module that is hand carried to the site for data transfer (∗9 Mode).

Storage Modules are assigned addresses (1-8) either through the ∗9 Mode or with the SMCOM or SMS software. 1 is the default address when the Storage Module is reset. Unless you are using one of the features which require different addresses, you need not assign any other address.

Address 1 is also a universal address when sending data or commands to a storage module with Instruction 96, ∗8, or ∗9. When address 1 is entered in the ∗9 Mode (default) or in the device code (71, Table 4.2-1) for Instruction 96 or the ∗8 Mode, The CR510 searches for the

Storage Module with the lowest address that is not full (fill and stop configuration only) and addresses it. In other words, if a single Storage Module is connected, and it is not full, address 1 will address that Storage Module regardless of the address that is assigned to the Module.

Address 1 would be used with Instruction 96 if several Storage Modules with different addresses were connected to the CR510 and were to be filled sequentially. The Storage modules would be configured as fill and stop. When the lowest addressed Module was full data would be written to the next lowest addressed Module, etc.

4.4.2 STORAGE MODULE USE WITH INSTRUCTION 96

When output to the Storage Module is enabled with Instruction 96, the Storage Module(s) may be either left with the CR510 for on-line data transfer and periodically exchanged, or brought to the site for data transfer.

USE OF STORAGE MODULE TO PICK UP DATA

The CR510 is capable of recognizing whether or not the Storage Module is connected. Each time Instruction 96 is executed and there is data to output, the CR510 checks for the presence of a Storage Module. If one is not present, the CR510 does not attempt to output data. Instead, the CR510 saves the data and continues its other operations without advancing the Storage Module Pointer (SPTR, Section 2.1).

When the user finally does connect the Storage Module to the CR510, two things happen:

1. Immediately upon connection, a File Mark is placed in the Storage Module Memory following the last data stored (if a File Mark wasn't the last data point already in storage).
2. During the next execution of Instruction 96, the CR510 recognizes that the Storage Module (SM) is present and outputs all data between the SPTR and the DSP location.

The File Mark allows the operator to distinguish blocks of data from different dataloggers or from different visits to the field.

To be certain that the Storage Module has been connected to the CR510 during an execution of P96, the user can:

- Leave the Storage Module connected for a time period longer than an execution interval *or*
- Use the SC90 9-Pin Serial Line Monitor. The SC90 contains an LED which lights up during data transmission. The user connects the SM to the CR510 with the SC90 on the line and waits for the LED to light. When the light goes off, data transfer is complete and the SM can be disconnected from the CR510.

4.4.3 ∗8 DUMP TO STORAGE MODULE

In addition to the on-line data output procedures described above, output to the Storage Module can be manually initiated in the ∗8 Mode. The procedure for setting up and transferring data is as follows:

1. Connect both the CR10KD Keyboard/Display (or terminal) and the Storage Module to the CR510 using the SC12 cable. (For terminals, an SC32A is needed. See Section 5 for interfacing details.)
2. Key in the appropriate commands as listed in Table 4.2-1.

4.5 ∗9 MODE -- SM192/716 STORAGE MODULE COMMANDS

The CSM1 does not support the ∗9 Mode Commands.

The ∗9 Mode is used to issue commands to the SM192/716 Storage Module, through the CR510, using the CR10KD or a terminal/computer. These commands are like ∗ Modes for the Storage Module and in some cases are directly analogous to the CR510 ∗ Modes. Command 7 enters a mode used to review stored data, and 8 is used to transfer data between two Storage Modules connected to the CR510. The operations with the Storage Module are not directly analogous as may be seen in Table 4.5-1 which lists the commands (e.g., when reviewing data, #A advances to the start of the next Output Array rather than to the same element in the next array with the same ID).

When ∗9 is keyed, the CR510 responds: 09:01. 1 is the default address for the Storage Module (Section 4.4.1). If you have more than 1 Storage Module connected, enter the address of the desired Storage Module. Address 1 will always work if only one Module is connected. Key A and the CR510 responds: 9N:00 Where N is the address which was entered.

You may now enter any of the commands in Table 4.5-1 (key in the command number and enter with A). Most commands have at least one response. Advance through the responses and return to the *9 command state by keying A.

TABLE 4.5-1. ∗9 Commands for Storage Module

COMMAND	DISPLAY	DESCRIPTION
1	01: 0000	RESET, enter 248 to erase all data and programs. While erasing, the SM checks memory. The number of good chips is then
	01: XX	displayed (6 for SM192, 22 SM716).
3	03: 01	INSERT FILE MARK, 1 indicates that the mark was inserted, 0 that it was not.
4	04: XX	DISPLAY/SET MEMORY CONFIGURATION enter the appropriate code to change configuration 0=ring, 1=fill & stop
5		DISPLAY STATUS (A to advance to each window)
	01: ABCD	Window 1:
	AB	Storage pointer location (chip no.)
	CD	Total good RAM chips (1-22)
	02: ABCD	Window 2:
	AB	Display pointer location (chip no.)
	C	Unloaded Batt. Chk. 0=low, 1=OK
	D	No. of Programs stored (Max=8)
	03: A0CD	Window 3:
	A	Errors logged (up to 9)
	0	Not Used
	C	Memory Config. (0=ring, 1=fill&stop)
	D	Memory Status (0=not full, 1=full)
	04: XXXXX	PROM signature (0 if bad PROM)
6	06: 0X	BATTERY CHECK UNDER LOAD (0=low, 1=OK)
7	07: 00	DISPLAY DATA, Select the Storage Module Area with these codes: 0 Dump pointer to SRP 1 File 1, current file 2 File 2, previous to file 1 3 File 3, previous to file 2 4 File 4, previous to file 3 5 File 5, previous to file 4 7 Display pointer to SRP 9 Oldest data to SRP 1-5 will loop within file boundaries, 0,7,9 allow display to cross boundaries
	07:XXXXXX	SM location at end of area selected. Key A to advance to first data. If another location is keyed in SM will jump to 1st start of array following that location. Review data with: A Advance and display next data point B Back-up one data point # Display location, C to return to data #A Advance to next start of Array #B Back-up to start of Array #D Return to ∗9 command mode
8		DUMP TO ANOTHER STORAGE MODULE
	08:00	Select Area as in 7 above
	01:XXXXXX	First Loc. in area selected/Enter Loc. to start dump
	02:XXXXXX	Final Loc. in area selected/Enter Loc. to end dump
	03:XX	Enter destination SM address
9		DISPLAY ADDRESSES OF CONNECTED SM
	XXXXXXXX	1 = occupied, 0 = unoccupied
	87654321	(Addresses 8-1 from left to right)
10		CHANGE ADDRESS
	10:0X	X is current address, enter address to change to (1-8)

SECTION 5. TELECOMMUNICATIONS

Telecommunications is used to retrieve data from Final Storage directly to a computer/terminal and to program the CR510. Any user communication with the CR510 that makes use of a computer or terminal instead of the CR10KD is through Telecommunications.

Telecommunications can take place over a variety of links including:

- *Telephone*
- *Cellular phone*
- *Radio frequency*
- *Short haul modem and twisted pair wire*
- *SC32A and ribbon cable*
- *Multi-drop interface and coax cable*

This section does not cover the technical interface details for any of these links. Those details are covered in Section 6 and in the individual manuals for the devices.

Data retrieval can take place in either ASCII or BINARY. The BINARY format is 5 times more compact than ASCII. The shorter transmission times for binary result in lower long distance costs if the link is telephone and lower power consumption with an RF link. On "noisy" links shorter blocks of data are more likely to get through without interruption.

For more efficient data transfer, binary data retrieval makes use of a signature for error detection. The signature algorithm assures a 99.998% probability that if either the data or its sequence changes, the signature changes.

Campbell Scientific has developed a software package which automates data retrieval and facilitates the programming of Campbell Scientific dataloggers and the handling of data files. This package has been designed to meet the most common needs in datalogger support and telecommunications. Therefore, this section does not furnish sufficient detail to write telecommunications software. Appendix B contains some details of binary data transfer and Campbell Scientific's binary data format.

The emphasis of this section is on the commands that a person would use when manually (i.e., keyed in by hand) interrogating or programming the CR510 via a computer/terminal. These commands and the responses to them are sent in the American Standard Code for Information Interchange (ASCII).

The telecommunications commands allow the user to perform several operations including:

- *monitor data in Input Storage and review data in Final Storage*
- *retrieve Final Storage data in either ASCII or BINARY*
- *open communications with the Storage Module*
- *remote keyboard programming*

The Remote Keyboard State (Section 5.2) allows the user with a computer/terminal to use the same commands as the CR10KD.

5.1 TELECOMMUNICATIONS COMMANDS

When a modem/terminal rings the CR510, the CR510 should answer almost immediately. Several carriage returns (CR) must be sent to the CR510 to allow it to set its baud rate to that of the modem/terminal (300, 1200, or 9600). Once the baud rate is set, the CR510 will send back the prompt, "**∗**", signaling that it is ready to receive a command.

GENERAL RULES governing the telecommunications commands are as follows:

1. **∗** from datalogger means "ready for command".
2. All commands are of the form: [no.]letter, where the number may or may not be optional.
3. Valid characters are the numbers **0-9**, the capital letters **A-M**, the colon (**:**), and the carriage return (**CR**).

4. An illegal character increments a counter and zeros the command buffer, returning a **∗**.
5. **CR** to datalogger means "execute".
6. **CRLF** from datalogger means "executing command".
7. ANY character besides a CR sent to the datalogger with a legal command in its buffer causes the datalogger to abort the command sequence with **CRLF∗** and to zero the command buffer.
8. All commands return a response code, usually at least a checksum.
9. The checksum includes all characters sent by the datalogger since the last **∗**, including the echoed command sequence, excluding only the checksum itself. The checksum is formed by summing the ASCII values, without parity, of the transmitted characters. The largest possible checksum value is 8191. Each time 8191 is exceeded, the CR510 starts the count over; e.g., if the sum of the ASCII values is 8192, the checksum is 0.
10. Commands that return Campbell Scientific binary format data (i.e., **F** and **K** commands) return a signature (see Appendix B.3).

The CR510 sends ASCII data with 8 bits, no parity, one start bit, and one stop bit.

After the CR510 answers a ring, or completes a command, it waits about 40 seconds (127 seconds in the Remote Keyboard State) for a valid character to arrive. It "hangs up" if it does not receive a valid character in this time interval. Some modems are quite noisy when not on line; it is possible for valid characters to appear in the noise pattern. To insure that this situation does not keep the CR510 in telecommunications, the CR510 counts all the invalid characters it receives from the time it answers a ring, and terminates communication after receiving 150 invalid characters.

The CR510 continues to execute its measurement and processing tasks while servicing the telecommunication requests. If the processing overhead is large (short Execution Interval), the processing tasks will slow the telecommunication functions. In a worst case situation, the CR510 interrupts the processing tasks to transmit a data point every 0.125 second.

The best way to become familiar with the Telecommunication Commands is to try them from a terminal connected to the CR510 via the SC32A (Section 6.7.1) or other interface. Commands used to interrogate the CR510 in the Telecommunications Mode are described in the following Table.

TABLE 5.1-1. Telecommunications Commands

Command	Description
[F.S. Area]**A**	SELECT AREA/STATUS - If 1 or 2 does not precede the A to select the Final Storage Area, the CR510 will default to Area 1. All subsequent commands other than A will address the area selected. Datalogger returns **R**eference, the DSP location; the number of filled **F**inal Storage locations; **V**ersion of datalogger; Final Storage **A**rea; **L**ocation of MPTR (the location number may be 1 to 7 characters long); **E**rrors #1, #2, and #3 where #1 is the number of E08's, #2 is the number of overrun errors, and #3 is the number of times the program stopped due to low voltage (all are cleared by entering 8888A; #2 is also cleared at time of program compilation); size of total **M**emory in CR510; the lithium **B**attery voltage; and **C**hecksum. All in the following format: R+xxxxx F+xxxxx Vxx Axx L+xxxxxxx Exx xx xx Mxxxx B+xxxxx Cxxxx If data is stored while in telecommunications, the A command must be issued to update the Reference to the new DSP.
[no. of arrays]**B**	BACK-UP - MPTR is backed-up the specified number of Output Arrays (no number defaults to 1) and advanced to the nearest start of array. CR510 sends the Area, MPTR Location, and Checksum: Ax Lxxxxxxx Cxxxx
[YR:DAY:HR:MM:SS]**C**	RESET/SEND TIME - If time is entered the time is reset. If only 2 colons are in the time string, HR:MM:SS is assumed; 3 colons means DAY:HR:MM:SS. If only the C is entered, time is unaltered. CR510 returns year, Julian day, hr:min:sec, and Checksum: Y:xx Dxxxx Txx:xx:xx Cxxxx
[no. of arrays]**D**	ASCII DUMP - If necessary, the MPTR is advanced to start of scan. CR510 sends the number of arrays specified (no number defaults to 1) or the number of arrays between MPTR and Reference, whichever is smaller, CRLF, Location, Checksum.
E	End call. Datalogger sends CRLF only.
[no. of loc.]**F**	BINARY DUMP - Used by CSI software for data retrieval. See Appendix C.
[F.S. loc. no.]**G**	MOVE MPTR - MPTR is moved to specified Final Storage location. The location number must be entered. CR510 sends Area, Location, and Checksum: Ax Lxxxxxxx Cxxxx
7H or **2718H**	REMOTE KEYBOARD - CR510 sends the prompt ">" and is ready to execute standard keyboard commands (Section OV3).
[loc. no.]**I**	Display/change value at Input Storage location. CR510 sends the value stored at the location. A new value and CR may then be sent. CR510 sends checksum. If no new value is sent (CR only), the location value will remain the same.

3142J TOGGLE FLAGS AND SET UP FOR K COMMAND - Used in the Monitor Mode and with the Heads Up Display. See Appendix C for details.

K CURRENT INFORMATION - In response to the K command, the CR510 sends datalogger time, user flag status, the data at the input locations requested in the J command, and Final Storage Data if requested by the J command. Used in the Monitor Mode and with Heads Up Display. See Appendix B.

[Password]**L** Unlocks security (if enabled) to the level determined by the password entered (See ∗C Mode, Section 1.7). CR510 sends security level (0-3) and checksum: Sxx Cxxxx

[X]**M** Connect to Storage Module with address 'X' and enter the Storage Module's Telecommunications Mode (see Storage Module manual). The Storage Module can also be accessed through the ∗9 Commands while in the Remote Keyboard (Section 4.5 and the Storage Module manual).

1N Connect phone modem to RF modem at phone to RF base station (requires 1200 baud communication).

5.2 REMOTE PROGRAMMING OF THE CR510

Remote programming of the CR510 can be accomplished with the datalogger support software or directly through the Remote Keyboard State.

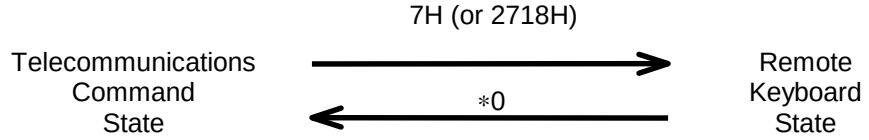
The datalogger support software was developed by Campbell Scientific for use with IBM or compatible PC's. Datalogger programs are developed on the computer using the program editor and downloaded to the datalogger with the terminal emulator program.

The CR510 is placed in the Remote Keyboard State by sending either "**7H**" or "**2718H**" and a carriage return (CR). The CR510 responds by sending a CR, line feed (LF), and the prompt '>'. The CR510 is then ready to receive the standard keyboard commands; it recognizes all the standard CR510 keyboard characters plus several additional characters, including the decimal point, the minus sign, and Enter (CR) (Section OV3.2). ENTERING ∗0 RETURNS THE CR510 TO THE TELECOMMUNICATIONS COMMAND STATE.

Remember that entering ∗0 will compile and run the CR510 program if program changes have been made. If the CR10KD is connected it will just display "LOG" when ∗0 is executed via telecommunications. It will not indicate active tables (keying "∗0" on the Keyboard/Display will show the tables).

The 7H Command is generally used with a terminal for direct entry since H makes use of a destructive backspace and does not send control Q between each entry. The 2718H Command functions the same as it does for other Campbell Scientific dataloggers (deleting an entry causes the entire entry to be sent, "control Q" is sent after each user entry).

It is important to remember that the Remote Keyboard State is still within Telecommunications. Entering ∗0 exits the Remote Keyboard State and returns the datalogger to the Telecommunications Command State, awaiting another command. So, the user can step back and forth between the Telecommunications Command State and the Remote Keyboard State.



SECTION 6. 9-PIN SERIAL INPUT/OUTPUT

6.1 PIN DESCRIPTION

All external communication peripherals connect to the CR510 through the 9-pin subminiature D-type socket connector located on the front of the Terminal Strip (Figure 6.1-1). Table 6.1-1 shows the I/O pin configuration, and gives a brief description of the function of each pin.

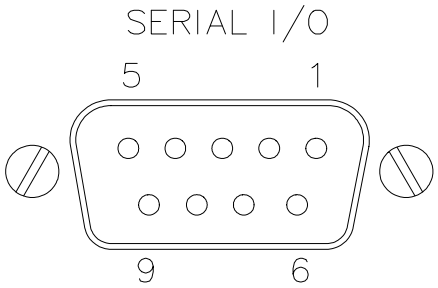


FIGURE 6.1-1. 9-pin Female Connector

TABLE 6.1-1. Pin Description

ABR = Abbreviation for the function name.
PIN = Pin number.
O = Signal Out of the CR510 to a peripheral.
I = Signal Into the CR510 from a peripheral.

PIN	ABR	I/O	Description
1	5 V	O	5V: Sources 5 VDC, used to power peripherals.
2	SG		Signal Ground: Provides a power return for pin 1 (5V), and is used as a reference for voltage levels.
3	RING	I	Ring: Raised by a peripheral to put the CR510 in the telecommunications mode.
4	RXD	I	Receive Data: Serial data transmitted by a peripheral are received on pin 4.
5	ME	O	Modem Enable: Raised when the CR510 determines that a modem raised the ring line.
6	SDE	O	Synchronous Device Enable: Used to address Synchronous Devices (SDs), and can be used as an enable line for printers.
7	CLK/HS	I/O	Clock/Handshake: Used with the SDE and TXD lines to address and transfer data to SDs. When not used as a clock, pin 7 can be used as a handshake line (during printer output, high enables, low disables).
8			12 V: Sources continuous 12 V, used to power telephone modems.
9	TXD	O	Transmit Data: Serial data are transmitted from the CR510 to peripherals on pin 9; logic low marking (0V) logic high spacing (5V) standard asynchronous ASCII, 8 data bits, no parity, 1 start bit, 1 stop bit, 300, 1200, 9600, 76,800 baud (user selectable).

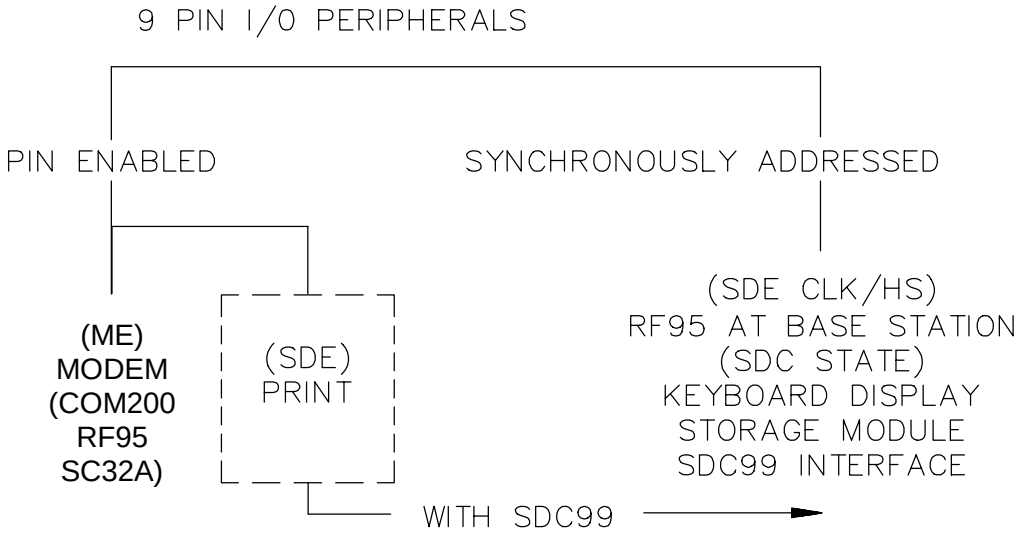


FIGURE 6.2-1. Hardware Enabled and Synchronously Addressed Peripherals

6.2 ENABLING AND ADDRESSING PERIPHERALS

While several peripherals may be connected in parallel to the 9-pin port, the CR510 has only one transmit line (pin 9) and one receive line (pin 4, Table 6.1-1). The CR510 selects a peripheral in one of two ways: 1) A specific pin is dedicated to that peripheral and the peripheral is enabled when the pin goes high; we will call this pin-enabled or simply enabled. 2) The peripheral is addressed; the address is sent on pin 9, each bit being synchronously clocked using pin 7. Pin 6 is set high while addressing.

6.2.1 PIN-ENABLED PERIPHERALS

Modem Enable (pin 5) is dedicated to a specific device. Synchronous Device Enable (pin 6) can either be used as a Print Enable or it can be used to address Synchronous Devices (Section 6.6).

Modem Enable (ME), pin 5, is raised to enable a modem that has raised the ring line. Modem/terminal peripherals include Campbell Scientific phone modems and computers or terminals using the SC32A RS232 interface. The CR510 interprets a ring interrupt (Section 6.3) to come from a modem if the device raises the CR510's Ring line, and holds it high until the CR510 raises the ME line. Only one modem/ terminal may be connected to the CR510.

Print Peripherals are defined as peripherals which have an asynchronous serial communications port used to RECEIVE data transferred by the CR510. In most cases the print peripheral is a printer, but could also be an on-line computer or other device.

Synchronous Device Enable (SDE), pin 6, may be used to enable a print peripheral only when no other addressable peripherals are connected to the 9-pin connector. Use of the SDE line as an enable line maintains CR510 compatibility with printer-type peripherals which require a line to be held high (Data Terminal Ready) in order to receive data.

If output to both a print peripheral and an addressable peripheral is necessary the SDC99 Synchronous Device Interface is required. With the SDC99 the print peripheral functions as an addressable peripheral. If the SDC99 is not used, the print peripheral receives the address and data sent to the addressed peripheral. Synchronous addressing appears as garbage characters on a print peripheral.

6.2.2 ADDRESSED PERIPHERALS

The CR510 has the ability to address Synchronous Devices (SDs). SDs differ from enabled peripherals in that they are not enabled solely by a hardware line (Section 6.2.1); an SD is enabled by an address synchronously clocked from the CR510 (Section 6.6).

Up to 16 SDs may be addressed by the CR510. Unlike an enabled peripheral, the CR510 establishes communication with an addressed peripheral before data are transferred. During data transfer an addressed peripheral uses pin 7 as a handshake line with the CR510.

Synchronously addressed peripherals include the CR10KD Keyboard Display, Storage Modules, SDC99 Synchronous Device Interface (SDC99), and RF95 RF Modem when configured as a synchronous device. The SDC99 interface is used to address peripherals which are normally pin enabled (Figure 6.2-1).

6.3 RING INTERRUPTS

There are three peripherals that can raise the CR510's ring line; modems, the CR10KD Keyboard Display, and the RF Modem configured for synchronous device for communication (RF-SDC). The RF-SDC is used when the CR510 is installed at a telephone to RF base station.

When the Ring line is raised, the processor is interrupted, and the CR510 determines which peripheral raised the Ring line through a process of elimination (Figure 6.3-1). The CR510 raises the CLK/HS line forcing all SDs to drop the ring line. If the ring line is still high the peripheral is a modem, and the ME line is raised. If the ring line is low the CR510 addresses the Keyboard Display and RF-SDC to determine which device to service. (Section 6.6)

After the CR510 has determined which peripheral raised the Ring line, the hierarchy is as follows:

A modem which raises the Ring line will interrupt and gain control of the CR510. A ring from a modem aborts data transfer to pin-enabled and addressed peripherals.

The CR10KD raises the ring line whenever a key is pressed. The CR10KD will not be serviced when the modem or RF-SDC is being serviced.

The ring from the CR10KD is blocked when the SDE line is high, preventing it from interrupting data transfer to a pin-enabled print device.

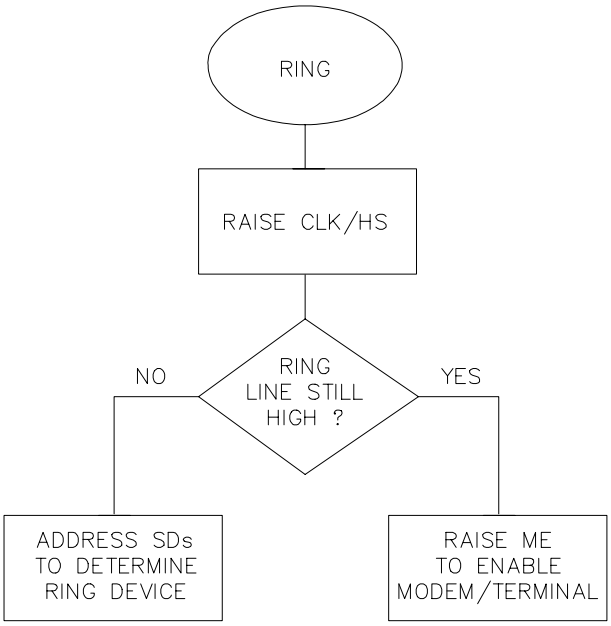


FIGURE 6.3-1. Servicing of Ring Interrupts

6.4 INTERRUPTS DURING DATA TRANSFER

Instruction 96 is used for on-line data transfer to peripherals (Section 4.1). Each peripheral connected to the CR510 requires an Instruction 96 with the appropriate parameter. If the CR510 is already communicating on the 9-pin connector when Instruction 96 is executed, the instruction puts the output request in a "queue" and program execution continues. As the 9-pin connector becomes available, each device in the queue will get its turn until the queue is empty.

Instruction 96 is aborted if a modem raises the Ring line. Data transfer to an addressed peripheral is aborted if the ring line is raised by a CR10KD or RF Modem configured as a synchronous device. Transfer of data is not resumed until the next time Instruction 96 is executed and the datalogger has exited telecommunications.

The ∗8 Mode is used to manually initiate data transfer from Final Storage to a peripheral. An abort flag is set if any key on the CR10KD or terminal is pressed during the transfer. Data transfer is stopped and the memory location displayed when the flag is set. During ∗8 data transfer the abort flag is checked as follows:

1. Comma separated ASCII - after every 32 characters.
2. Printable ASCII - after every line.
3. Binary - after every 256 Final Storage locations.

6.5 MODEM/TERMINAL PERIPHERALS

The CR510 considers any device with an asynchronous serial communications port which raises the Ring line (and holds it high until the ME line is raised) to be a modem peripheral. Modem/terminals include Campbell Scientific phone modems, and most computers, terminals, and modems using the SC32A Optically Isolated RS232 Interface or the SC932 RS232 DCE Interface.

When a modem raises the Ring line, the CR510 responds by raising the ME line. The CR510 must then receive carriage returns until it can establish baud rate. When the baud rate has been set, the CR510 sends a carriage return, line feed, "*".

The ME line is held high until the CR510 receives an "E" to exit telecommunications. The ME is also lowered if a character is not received after 40 seconds in the Telecommunications Command State (2 minutes in the Remote Keyboard State).

Some modems are quite noisy when not on line; it is possible for valid characters to appear in the noise pattern. For this reason, the CR510 counts all the invalid characters it receives from the time it answers a ring, and terminates communication (lowers the ME line and returns to the *0 Mode) after receiving 150 invalid characters.

6.6 SYNCHRONOUS DEVICE COMMUNICATION

The CR510 has the ability to address Synchronous Devices (SDs). SDs differ from enabled peripherals (Section 6.2.1) in that they are not enabled solely by a hardware line. An SD is enabled by an address synchronously clocked from the CR510. Up to 16 SDs may be addressed by the CR510, requiring only three pins of the 9-pin connector.

Synchronous Device Communication (SDC) discussed here is for those peripherals which connect to the 9-pin serial port. (This should not be confused with Synchronous Device for Measurement (SDM) peripherals. Although the communication protocol for SDMs is very similar, their addressing is independent of SDC addresses and they do not have a ring line.)

SD STATES

The CR510 and the SDs use a combination of the Ring, Clock Handshake (CLK/HS) and Synchronous Device Enable (SDE) lines to establish communication. The CR510 can put the SDs into one of six states.

STATE 1, the SD Reset State

The CR510 forces the SDs to the reset/request state by lowering the SDE and CLK/HS lines. The SD cannot drive the CLK/HS or RXD lines in State 1, however, it can raise the Ring line if service is needed. The SD can never pull the Ring low if a Modem/Terminal is holding it high. Data on TXD is ignored by the SD.

STATE 2, the SD Addressing State

The CR510 places the SDs in the addressing state by raising CLK/HS followed by or simultaneously raising SDE (Figure 6.6-1). TXD must be low while SDE and CLK/HS are changing to the high state.

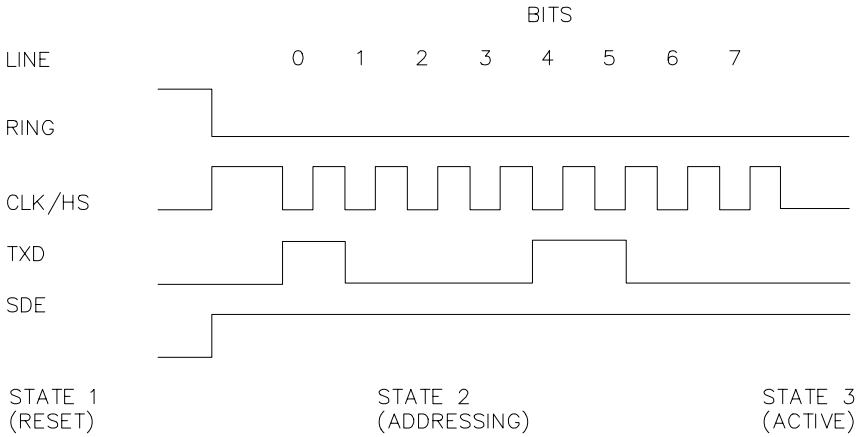


FIGURE 6.6-1. Addressing Sequence for the RF Modem

State 2 requires all SDs to drop the Ring line and prepare for addressing. The CR510 then synchronously clocks 8 bits onto TXD using CLK/HS as a clock. The least significant bit is transmitted first and is always logic high. Each bit transmitted is stable on the rising edge of CLK/HS. The SDs shift in bits from TXD on the rising edge of CLK/HS provided by the CR510. The CR510 can only address one device per State 2 cycle. More than one SD may respond to the address, however. State 2 ends when the 8th bit is received by the SD.

SDs implemented with shift registers decode the 4 most significant bits (bits 4, 5, 6, and 7) for an address. Bit 0 is always logic high. Bits 1, 2, and 3 are optional function selectors or commands. Addresses established to date are shown in Table 6.6-1 and are decoded with respect to the TXD line.

TABLE 6.6-1. SD Addresses

	B7	B6	B5	B4	B3	B2	B1	B0
SDC99 Printer	0	0	0	0	X	X	X	1
Storage Module	0	0	0	1	X	X	X	1
CR10KD Keyboard	0	0	1	0	0	X	X	1
CR10KD Display	0	0	1	0	1	X	X	1
RF Modem	0	0	1	1	X	X	X	1

STATE 3, the SD Active State

The SD addressed by State 2, enters State 3. All other SDs enter State 4. An active SD returns to State 1 by resetting itself, or by the CR510 forcing it to reset.

Active SDs have different acknowledgment and communication protocols. Once addressed, the SD is free to use the CLK/HS, TXD, and RXD lines according to its protocol with the CR510. The CR510 may also pulse the SDE line after addressing, as long as the CLK/HS and SDE are not low at the same time.

STATE 4, the SD Inactive State

The SDs not addressed by State 2 enter State 4, if not able to reset themselves (e.g., SM192 Storage Module). Inactive SDs ignore data on the TXD line and are not allowed to use the CLK/HS or RXD lines. Inactive SDs may raise the Ring line to request service.

STATE 5

State 5 is a branch from State 1 when the SDE line is high and the CLK/HS line is low. The SDs must drop the Ring line in this state. This state is not used by SDs. The CR510 must force the SDs back to the reset state from State 5 before addressing SDs.

STATE 6

State 6 is a branch from State 1, like State 5, except the SDE line is low and the CLK/HS line is high. The SDs must drop the Ring line in this state.

6.7 MODEM/TERMINAL AND COMPUTER REQUIREMENTS

The CR510 considers any device with an asynchronous serial communications port which raises the Ring line (and holds it high until the ME line is raised) to be a modem peripheral. Modems include Campbell Scientific phone modems, and most computers, terminals, and modems using the SC32A Optically Isolated RS232 Interface.

6.7.1 SC32A INTERFACE TO COMPUTER

Most computers require the SC32A Optically Isolated RS232 Interface. The SC32A raises the CR510's ring line when it receives characters from a modem, and converts the CR510's logic levels (0 V logic low, 5V logic high) to RS232 logic levels.

The SC32A 25-pin port is configured as Data Communications Equipment (DCE) (see Table 6.7-1) for direct connection to Data Terminal Equipment (DTE), which includes most PCs and printers.

When the SC32A receives a character from the terminal/computer (pin 2), 5 V is applied to the datalogger Ring line (pin 3) for one second or until the Modem Enable line (ME) goes high. The CR510 waits approximately 40 seconds to receive carriage returns, which it uses to establish baud rate. After the baud rate has been set the CR510 transmits a carriage return, line feed, "*", and enters the Telecommunications Command State (Section 5). If the carriage returns are not received within the 40 seconds, the CR510 "hangs up".

TABLE 6.7-1. SC32A Pin Description

PIN = Pin number
O = Signal Out of the SC32A to a peripheral
I = Signal Into the SC32A from peripheral

25-PIN FEMALE PORT:

PIN #	I/O	ABBREVIATION
1		GROUND
2	I	TX
3	O	RX
4	I	RTS (POWER)
5	O	CTS
6	O	DSR
7		GROUND
8	O	DCD
20	I	DTR (POWER)

9-PIN MALE PORT:

PIN #	ABBREVIATION
1	+5V INPUT
2	GROUND
3	RING
4	RX
5	ME
6	SDE
9	TX

NOTE: The SC32A has a jumper, which when used, passes data only when the ME line is high and the SDE line is low. The function of the jumper is to block data sent to SDs from being received by a computer/terminal used to initiate data transfer. Synchronous data will appear as garbage characters on a computer/terminal.

6.7.2 SC932 INTERFACE TO MODEMS

Most modems have an RS232 port configured as DCE. For connection to DCE devices such as modems and some computers, the SC932 9-pin to RS232 DCE Interface should be used.

6.7.3 COMPUTER/TERMINAL REQUIREMENTS

Computer/terminal peripherals are usually configured as Data Terminal Equipment (DTE). Pins 4 and 20 are used as handshake lines, which are set high when the serial port is enabled. Power for the SC32A RS232 section is taken from these pins. For equipment configured as DTE (see Table 6.7-2) a direct ribbon cable connects the computer/terminal to the SC32A. Clear to Send (CTS) pin 5, Data Set Ready (DSR) pin 6, and Data Carrier Detect (DCD) pin 8 are held high by the SC32A (when the RS232 section is powered) which should satisfy hardware handshake requirements of the computer/terminal.

Table 6.7-2 lists the most common RS232 configuration for Data Terminal Equipment.

TABLE 6.7-2. DTE Pin Configuration

PIN = 25-pin connector number
ABR = Abbreviation for the function name
O = Signal Out of terminal to another device
I = Signal Into terminal from another device

PIN	ABR	I/O	FUNCTION
2	TD	O	Transmitted Data: Data is transmitted from the terminal on this line.
3	RD	I	Received Data: Data is received by the terminal on this line.
4	RTS	O	Request to Send: The terminal raises this line to ask a receiving device if the terminal can transmit data.
5	CTS	I	Clear to Send: The receiving device raises this line to let the terminal know that the receiving device is ready to accept data.
20	DTR	O	Data Terminal Ready: The terminal raises this line to tell the modem to connect itself to the telephone line.
6	DSR	I	Data Set Ready: The modem raises this line to tell the terminal that the modem is connected to the phone line.
8	DCD	I	Data Carrier Detect: The modem raises this line to tell the terminal that the modem is receiving a valid carrier signal from the phone line.
22	RI	I	Ring Indicator: The modem raises this line to tell the terminal that the phone is ringing.
7	SG		Signal Ground: Voltages are measured relative to this point.

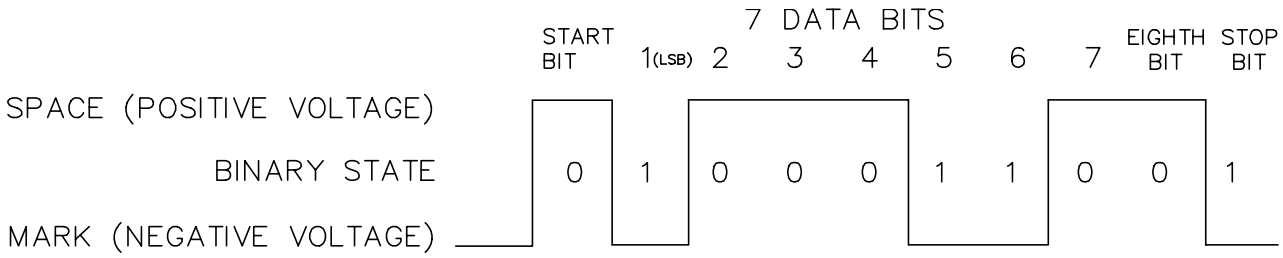


FIGURE 6.7-1. Transmitting the ASCII Character 1

6.7.3 COMMUNICATION PROTOCOL/TROUBLE SHOOTING

The ASCII standard defines an alphabet consisting of 128 different characters where each character corresponds to a number, letter, symbol, or control code.

An ASCII character is a binary digital code composed of a combination of seven "bits", each bit having a binary state of 1 (one) or 0 (zero). For example, the binary equivalent for the ASCII character "1" is 0110001 (decimal 49).

ASCII characters are transmitted one bit at a time, starting with the 1st (least significant) bit. During data transmission the marking condition is used to denote the binary state 1, and the spacing condition for the binary state 0. The signal is considered marking when the voltage is more negative than minus three volts with respect to ground, and spacing when the voltage is more positive than plus three volts.

Most computers use 8-bits (1 byte) for data communications. The 8th bit is sometimes used for a type of error checking called parity-checking. Even parity binary characters have an even number of 1's, odd-parity characters have an odd number of 1's. When parity checking is used, the 8th bit is set to either a 1 or a 0 to make the parity of the character correct. The CR510 ignores the 8th bit of a character that is received, and transmits the 8th bit as a binary 0. This method is generally described as "no parity".

To separate ASCII characters, a Start bit is sent before the 1st bit and a Stop bit is sent after the 8th bit. The start bit is always a space, and the stop bit is always a mark. Between characters the signal is in the marking condition.

Figure 6.7-1 shows how the ASCII character "1" is transmitted. When transmitted by the CR510 using the SC32A RS232 interface spacing and marking voltages are positive and negative, as shown. Signal voltages at the CR510 I/O port are 5V in the spacing condition, and 0V in the marking condition.

BAUD RATE

BAUD RATE is the number of bits transmitted per second. The CR510 can communicate at 300, 1200, 9600, and 76,800 baud. In the Telecommunications State, the CR510 will set its baud rate to match the baud rate of the computer/terminal.

Typically the baud rate of the modem/computer/terminal is set either with dip switches, or programmed from the keyboard. The instrument's instruction manual should explain how to set it.

DUPLEX

Full duplex means that two devices can communicate in both directions simultaneously. Half duplex means that the two devices must send and receive alternately. Full duplex should always be specified when communicating with Campbell Scientific peripherals and modems. However, communication between some Campbell Scientific modems (such as the RF95 RF modem) is carried out in a half duplex fashion. This can affect the way commands should be sent to and received from such a modem, especially when implemented by computer software.

To overcome the limitations of half duplex, some communications links expect a terminal sending data to also write the data to the screen. This saves the remote device having to echo that data back. If, when communicating with a Campbell Scientific device, characters are displayed twice (in pairs), it is likely that the terminal is set to half duplex rather than the correct setting of full duplex.

IF NOTHING HAPPENS

If the CR510 is connected to the SC32A RS232 interface and a modem/terminal, and an "*" is not received after sending carriage returns:

1. Verify that the CR510 has power AT THE 12V AND GROUND INPUTS, and that the cables connecting the devices are securely connected.

2. Verify that the port of the modem/terminal is an asynchronous serial communications port configured as DTE (see Table 6.7-2). The most common problems occur when the user tries to use a parallel port, or doesn't know the port assignments, i.e. COM1 or COM2. IBM, and most compatibles come with a Diagnostic disk which can be used to identify ports, and their assignments. If the serial port is standard equipment, then the operators manual should give you this information.

3. Verify that there is 5 volts between the CR510 5V and G terminals. Call Campbell Scientific technical support if the voltage is less than 4.8 volts.

Some serial ports, e.g., the Super Serial Card for Apple computers, can be configured as DTE or DCE with a jumper block. Pin functions must match Table 6.7-2.

If you are using a computer to communicate with the datalogger, communication software must be used to enable the serial port and to make the computer function as a terminal. The port should be enabled for 300, 1200, or 9600 baud, 8 data bits, 1 stop bit, and no parity. Campbell Scientific's GraphTerm, PC208E, PC208W, and TCOM provide this function.

If you are not sure that your computer/terminal is sending or receiving characters, there is a simple way to verify it. Make sure that the duplex is set to full. Next, take a paper clip and connect one end to pin 2, and the other end to pin 3 of the serial port. Each character typed on the keyboard will be displayed only if transmitted from the terminal on pin 2, and received on pin 3 (if duplex is set to half, the character will be displayed once if it is not transmitted, or twice if it is transmitted).

IF GARBAGE APPEARS

If garbage characters appear on the display, check that the baud rate is supported by the CR510. If the baud rate is correct, verify that the computer/terminal is set for 8 data bits, and no parity. Garbage will appear if 7 data bits and no parity are used. If the computer/terminal is set to 8 data bits and even or odd parity, communication cannot be established.

SECTION 7. MEASUREMENT PROGRAMMING EXAMPLES

This section gives some examples of Input Programming for common sensors used with the CR510. These examples detail only the connections, Input, Program Control, and Processing Instructions necessary to perform measurements and store the data in engineering units in Input Storage. Output Processing Instructions are omitted (see Section 8 for some processing and program control examples). It is left to the user to program the necessary instructions to obtain the final data in the form desired. No output to final storage will take place without additional programming.

The examples given in this section would likely be only fragments of larger programs. In general, the examples are written with the measurements made by the lowest numbered channels, the instructions at the beginning of the program table, and low number Input Storage Locations used to store the data. It is unlikely that an application and CR510 configuration exactly duplicates that assumed in an example.

These examples are not meant to be used verbatim; sensor calibration, input channels, and input locations must be adjusted for the actual circumstances. Unless otherwise noted, all excitation channels are switched analog output.

7.1 SINGLE ENDED VOLTAGE 107 TEMPERATURE PROBE

Instruction 11 excites Campbell Scientific's 107 Thermistor Probe with a 2 VAC excitation, makes a single ended measurement and calculates temperature (°C) with a fifth order polynomial. In this example, the temperatures are obtained from three 107 probes. The measurements are made on single-ended channels 1-3 and the temperatures are stored in Input Locations 1-3.

CONNECTIONS

The black leads from the probes go to excitation channel 1, the purple leads go to analog ground (AG), the clear leads go to ground (G), and the red leads go to single-ended channels 1, 2, and 3 (channel 1H, channel 1L, and channel 2H, respectively).

PROGRAM

```
01:   Temp (107) (P11)
        1:  3          Reps
        2:  1          SE Channel
        3:  1          Excite all reps w/E1
        4:  1          Loc [ 107_T_1   ]
        5:  1          Mult
        6:  0          Offset
```

7.2 DIFFERENTIAL VOLTAGE MEASUREMENT

Some sensors either contain or require active signal conditioning circuitry to provide an easily measured analog voltage output. Generally, the output is referenced to the sensor ground. The associated current drain usually requires a power source external to the CR510. A typical connection scheme where AC power is not available and both the CR510 and sensor are powered by an external battery is shown in Figure 7.2-1. Since a single-ended measurement is referenced to the CR510 ground, any voltage difference between the sensor ground and CR510 ground becomes a measurement error. A differential measurement avoids this error by measuring the signal between the 2 leads without reference to ground. This example analyzes the potential error on a differential CO_2 measurement using a LI-COR CO_2/H_2O analyzer, model LI-6262.

The wire used to supply power from the external battery is 18 AWG with an average resistance of 6.5 ohms/1000 ft. The power leads to the CR510 and LI-6262 are 2 ft and 10 ft, respectively. Typical current drain for the LI-6262 is 1000 mA. When making measurements, the CR510 draws about 35 mA. Since voltage is equal to current multiplied by resistance (V=IR), ground voltages at the LI-6262 and the CR510 relative to battery ground are:

LI-6262 ground =
1A ∗ 6.5 ohms/1000 ft ∗ 10 ft = +0.065 V

CR510 ground =
0.035A ∗ 6.5 ohms/1000 ft ∗ 2 ft = +0.0005 V

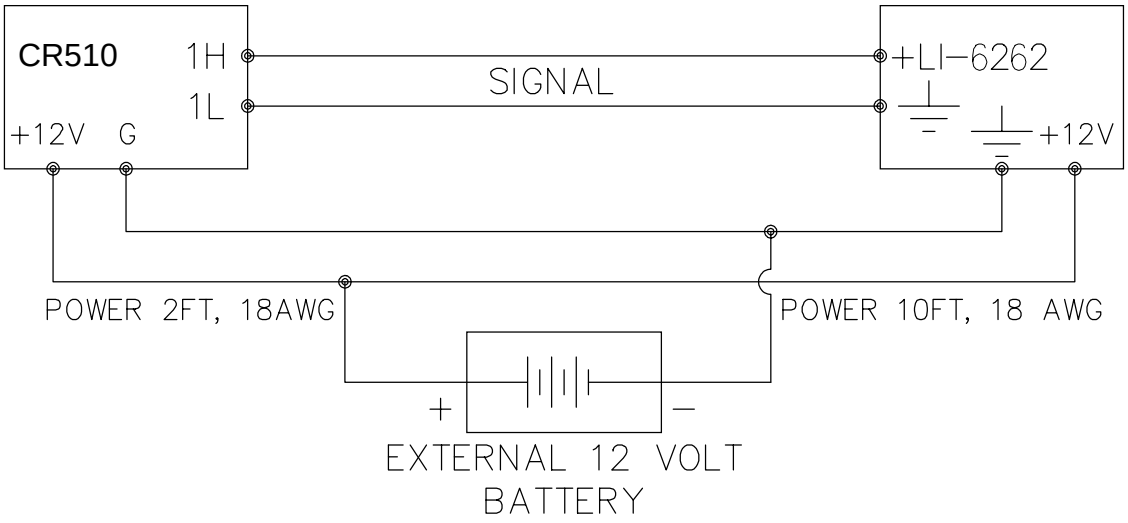


FIGURE 7.2-1. Typical Connection for Active Sensor with External Battery

Ground at the LI-6262 is 0.065 V higher than ground at the CR510. The LI-6262 can be programmed to output a linear voltage (0 to 100 mV) that is proportional to differential CO_2, 100 µmol/mol full scale, or 1 µmol/mol/mV. If the output is measured with a single-ended voltage measurement, it is 0.065 V or 65 µmol/mol high. If this offset remained constant, it could be corrected in programming. However, it is better to use a differential voltage measurement which does not rely on the current drain remaining constant. The program that follows illustrates the use of Instruction 2 to make the measurement. A multiplier of 1 is used to convert the millivolt output into µmol/mol.

PROGRAM

```
01:   Volt (Diff) (P2)
        1:   1        Reps
        2:   25       ±2500 mV 60 Hz Rejection
                      Range
        3:   1        DIFF Channel
        4:   1        Loc [ umol_mol   ]
        5:   1        Mult
        6:   0        Offset
```

7.3 HMP45C TEMPERATURE AND RH PROBE

Instruction 11 (107 Probe) is used to measure the temperature portion of the HMP45C Probe. It makes a single ended voltage measurement, and calculates temperature with a fifth order polynomial. A multiplier of 1.0 and offset of 0.0 yields temperature in degrees Celsius.

Instruction 4 is used to measure relative humidity. It provides an excitation voltage to power the RH sensor. A 150 millisecond delay is allowed for warm-up before the single-ended measurement is made.

The probe has an output of 0 to 100 millivolts for the 0 to 100% RH range, a multiplier of 0.1 and an offset of 0.0 provides relative humidity in percent.

This example uses Control Port 1 to power the RH sensor.

CONNECTIONS

The HMP45C probe is measured by two single-ended analog input channels. The green (RH) and the orange (temperature) leads are connected to either a HI or LO input. The black thermistor excitation lead connects to any excitation channel. The yellow lead powers the RH sensor via control port 1. The white and purple leads connect to Analog Ground (AG). The clear lead is the shield which connects to Ground (G) on the CR510.

```
01:   Temp (107) (P11)
        1:   1        Reps
        2:   1        SE Channel
        3:   1        Excite all reps w/E1
        4:   1        Loc [ air_temp   ]
        5:   1.0      Mult
        6:   0.0      Offset
```

02: Do (P86)
1: 41 Set Port 1 High

03: Excitation with Delay (P22)
1: 2 Ex Channel
2: 0 Delay w/Ex (units = 0.01 sec)
3: 15 Delay after Ex (units = 0.01 sec)
4: 0 mV Excitation

04: Volts (SE) (P1)
1: 1 Reps
2: 5 2500 mV Slow Range
3: 2 SE Channel
4: 2 Loc [RH]
5: .1 Mult
6: 0 Offset

05: Do (P86)
1: 51 Set Port 1 Low

7.4 ANEMOMETER WITH PHOTOCHOPPER OUTPUT

An anemometer with a photochopper transducer produces a pulse output which is measured by the CR510's Pulse Count Instruction. The Pulse Count Instruction with a Configuration Code of 20, measures "high frequency pulses", "discards data from excessive intervals", and "outputs the reading as a frequency" (Hz = pulses per second). The frequency output is the only output option that is independent of the scan rate.

The anemometer used in this example is the R. M. Young Model 12102D Cup Anemometer, with a 10 window chopper wheel. The photochopper circuitry is powered from the CR510 12 V supply; AC power or back-up batteries should be used to compensate for the increased current drain.

Wind speed is desired in meters per second (m/s). There is a pulse each time a window in the chopper wheel, which revolves with the cups, allows light to pass from the source to the photoreceptor. Because there are 10 windows in the chopper wheel, there are 10 pulses per revolution. Thus, 1 revolution per minute (rpm) is equal to 10 pulses per 60 seconds (1 minute) or 6 rpm = 1 pulse per second (Hz). The manufacturer's calibration for relating wind speed to rpm is:

$$\text{Wind(m/s)} = (0.01632 \text{ m/s})/\text{rpm} * \text{Xrpm} + 0.2 \text{ m/s}$$

The result of the Pulse Count Instruction (Configuration Code = 20) is X pulses per sec. (Hz). The multiplier and offset to convert XHz to meters per second are: Wind (m/s) = (0.01632 m/s)/rpm x (6 rpm/Hz) x XHz + 0.2 m/s

$$\text{Wind (m/s)} = (0.09792 \text{ m/s})/\text{Hz} \times \text{XHz} + 0.2 \text{ m/s}$$

PROGRAM

01: Pulse (P3)
1: 1 Reps
2: 1 Pulse Input Channel
3: 20 High Frequency, Output Hz
4: 10 Loc [WS_mph]
5: .09792 Mult
6: .2 Offset

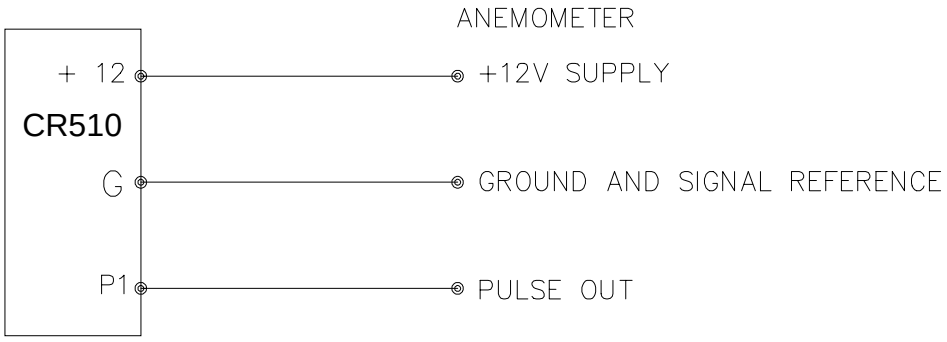


FIGURE 7.4-1. Wiring Diagram for Anemometer

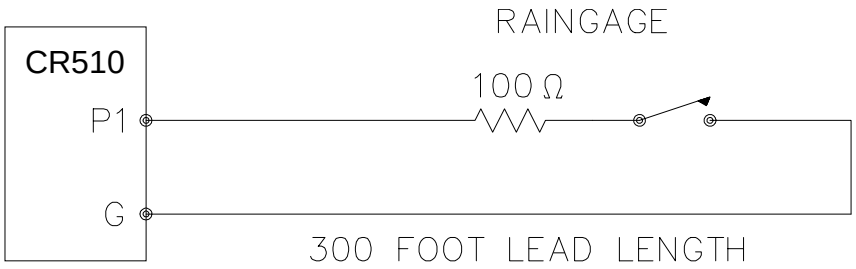


FIGURE 7.5-1. Wiring Diagram for Rain Gage with Long Leads

7.5 TIPPING BUCKET RAIN GAGE WITH LONG LEADS

A tipping bucket rain gage is measured with the Pulse Count Instruction configured for Switch Closure. Counts from long intervals will be used, as the final output desired is total rainfall (obtained with Instruction 72, Totalize). If counts from long intervals were discarded, less rainfall would be recorded than was actually measured by the gage (assuming there were counts in the long intervals). Output is desired in millimeters of precipitation. The gage is calibrated for a 0.01 inch tip, therefore, a multiplier of 0.254 is used.

In a long cable there is appreciable capacitance between the lines. The capacitance is discharged across the switch when it closes. In addition to shortening switch life, a transient may be induced in other wires packaged with the rain gage leads each time the switch closes. The 100 ohm resistor protects the switch from arcing and the associated transient from occurring, and should be included any time leads longer than 100 feet are used with a switch closure.

> **NOTE:** The TE525 and TE525MM raingages from CSI always have this resistor installed.

PROGRAM

```
01:   Pulse (P3)
        1:   1        Reps
        2:   1        Pulse Input Channel
        3:   2        Switch Closure, All Counts
        4:  11        Loc [ Precip_mm ]
        5:   .254     Mult
        6:   0        Offset
```

7.6 100 OHM PRT IN 4 WIRE HALF BRIDGE

Instruction 9 is the best choice for accuracy where the Platinum Resistance Thermometer (PRT) is separated from other bridge completion resistors by a lead length having more than a few thousandths of an ohm resistance. In this example, it is desired to measure a temperature in the range of -10 to 40°C. The length of the cable from the CR510 to the PRT is 500 feet.

Figure 7.6-1 shows the circuit used to measure the PRT. The 10 kohm resistor allows the use of a high excitation voltage and low voltage ranges on the measurements. This insures that noise in the excitation does not have an effect on signal noise. Because the fixed resistor (R_f) and the PRT (R_S) have approximately the same resistance, the differential measurement of the voltage drop across the PRT can be made on the same range as the differential measurement of the voltage drop across R_f.

If the voltage drop across the PRT (V_2) is kept under 50 mV, self heating of the PRT should be less than 0.001°C in still air. The best resolution is obtained when the excitation voltage is large enough to cause the signal voltage to fill the measurement voltage range. The resolution of this measurement on the 25mV range is +0.04°C. The voltage drop across the PRT is equal to V_x multiplied by the ratio of R_S to the total resistance, and is greatest when R_S is greatest (R_S=115.54 ohms at 40°C). To find the maximum excitation voltage that can be used, we assume V_2 equal to 25 mV and use Ohm's Law to solve for the resulting current, I.

$I = 25\ mV/R_S = 25\ mV/115.54\ ohms = 0.216\ mA$

Next solve for V_x:

$$V_x = I(R_1+R_s+R_f) = 2.21 \text{ V}$$

If the actual resistances were the nominal values, the CR510 would not over range with V_x = 2.2 V. To allow for the tolerances in the actual resistances, it is decided to set V_x equal to 2.1 volts (e.g., if the 10 kohms resistor is 5% low, then $R_s/(R_1+R_s+R_f)$=115.54/9715.54, and V_x must be 2.102 V to keep V_s less than 25 mV).

The result of Instruction 9 when the first differential measurement (V_1) is not made on the 2.5 V range is equivalent to R_s/R_f. Instruction 16 computes the temperature (°C) for a DIN 43760 standard PRT from the ratio of the PRT resistance at the temperature being measured to its resistance at 0°C (R_s/R_0). Thus, a multiplier of R_f/R_0 is used in Instruction 9 to obtain the desired intermediate, R_s/R_0 ($=R_s/R_f \times R_f/R_o$). If R_s and R_0 were each exactly 100 ohms, the multiplier would be 1. However, neither resistance is likely to be exact. The correct multiplier is found by connecting the PRT to the CR510 and entering Instruction 9 with a multiplier of 1. The PRT is then placed in an ice bath (@ 0°C; $R_s=R_0$), and the result of the bridge measurement is read using the ∗6 Mode. The reading is R_s/R_f, which is equal to R_o/R_f since $R_s=R_o$. The correct value of the multiplier, R_f/R_0, is the reciprocal of this reading. The initial reading assumed for this example was 0.9890. The correct multiplier is: R_f/R_0 = 1/0.9890 = 1.0111.

The fixed 100 ohm resistor must be thermally stable. Its precision is not important because the exact resistance is incorporated, along with that of the PRT, into the calibrated multiplier. The 10 ppm/°C temperature coefficient of the fixed resistor will limit the error due to its change in resistance with temperature to less than 0.15°C over the specified temperature range. Because the measurement is ratiometric (R_s/R_f), the properties of the 10 kohm resistor do not affect the result.

A terminal input module (Model 4WPB100) can be used to complete the circuit shown in Figure 7.8-1.

PROGRAM

```
01:    Full Bridge w/mv Excit (P9)
       1:    1          Reps
       2:    23         ±25 mV 60 Hz Rejection
                        Ex Range
       3:    23         ±25 mV 60 Hz Rejection
                        Br Range
       4:    1          DIFF Channel
       5:    1          Excite all reps w/Exchan 1
       6:2100           mV Excitation
       7:    1          Loc [ Rs_Ro    ]
       8:    1.0111     Mult
       9:    0          Offset

02:    Temperature RTD (P16)
       1:    1          Reps
       2:    1          R/Ro Loc [ Rs_Ro    ]
       3:    2          Loc [ Temp_C   ]
       4:    1          Mult
       5:    0          Offset
```

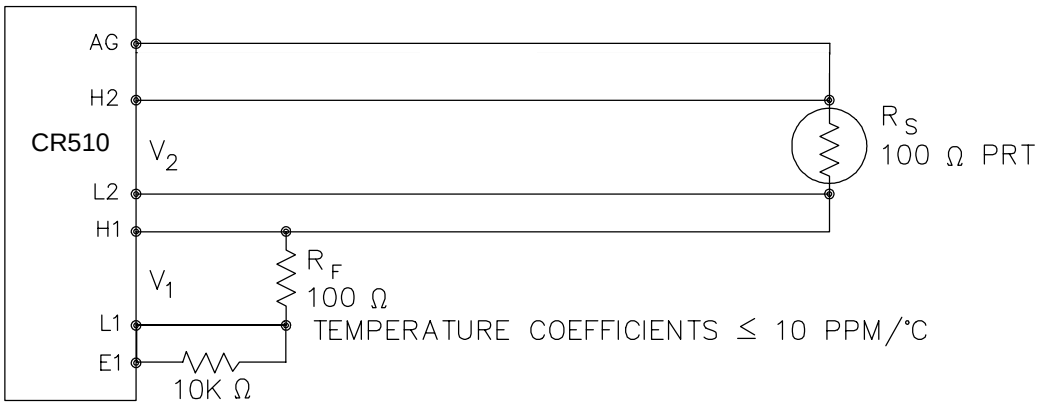


FIGURE 7.6-1. Wiring Diagram for PRT in 4 Wire Half Bridge

7.7 100 OHM PRT IN 3 WIRE HALF BRIDGE

The temperature measurement requirements in this example are the same as in Section 7.8. In this case, a three wire half bridge, Instruction 7, is used to measure the resistance of the PRT. The diagram of the PRT circuit is shown in Fig. 7.7-1.

As in the example in Section 7.8, the excitation voltage is calculated to be the maximum possible, yet allow the +25 mV measurement range. The 10 kohm resistor has a tolerance of ±1%; thus, the lowest resistance to expect from it is 9.9 kohms. We calculate the maximum excitation voltage (V_x) to keep the voltage drop across the PRT less than 25 mV:

$$0.025V > V_x\ 115.54/(9900+115.54);$$
$$V_x < 2.17\ V$$

The excitation voltage used is 2.1 V.

The multiplier used in Instruction 7 is determined in the same manner as in Section 7.8. In this example, the multiplier (R_f/R_0) is assumed to be 100.93.

The 3 wire half bridge compensates for lead wire resistance by assuming that the resistance of wire A is the same as the resistance of wire B. The maximum difference expected in wire resistance is 2%, but is more likely to be on the order of 1%. The resistance of R_s calculated with Instruction 7, is actually R_s plus the difference in resistance of wires A and B. The average resistance of 22 AWG wire is 16.5 ohms per 1000 feet, which would give each 500 foot lead wire a nominal resistance of 8.3 ohms. Two percent of 8.3 ohms is 0.17 ohms. Assuming that the greater resistance is in wire B, the resistance measured for the PRT (R_0 = 100 ohms) in the ice bath would be 100.17 ohms, and the resistance at 40°C would be 115.71. The measured ratio R_s/R_0 is 1.1551; the actual ratio is 115.54/100 = 1.1554. The temperature computed by Instruction 16 from the measured ratio would be about 0.1°C lower than the actual temperature of the PRT. This source of error does not exist in the example in Section 7.8, where a 4 wire half bridge is used to measure PRT resistance.

The advantages of the 3 wire half bridge are that it only requires 3 lead wires going to the sensor and takes 2 single-ended input channels, whereas the 4 wire half bridge requires 4 wires and 2 differential channels.

A terminal input module (Model 3WHB10K) can be used to complete the circuit in Figure 7.7-1.

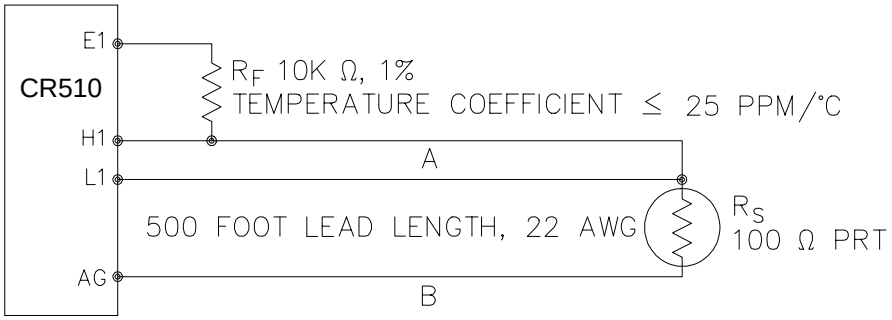


FIGURE 7.7-1. 3 Wire Half Bridge Used to Measure 100 ohm PRT

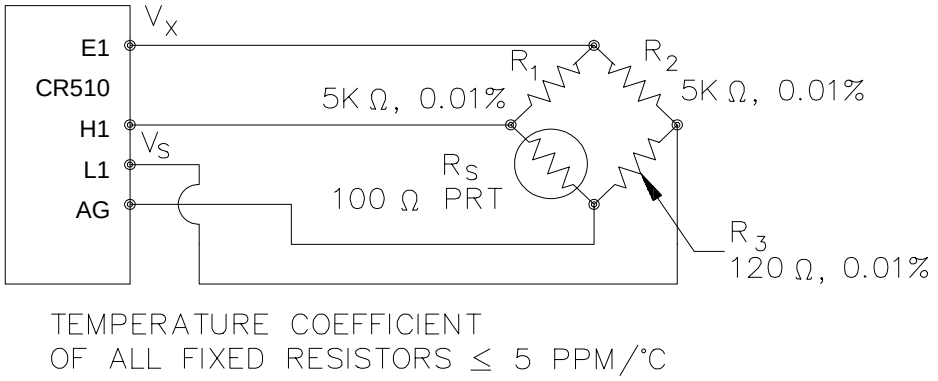


FIGURE 7.8-1. Full Bridge Schematic for 100 ohm PRT

PROGRAM

```
01:  3W Half Bridge (P7)
       1:    1        Reps
       2:   23        ±25 mV 60 Hz Rejection
                      Range
       3:    1        SE Channel
       4:    1        Excite all reps w/Exchan 1
       5: 2100        mV Excitation
       6:    1        Loc [ Rs_Ro    ]
       7:  100.93     Mult
       8:    0        Offset

02:  Temperature RTD (P16)
       1:    1        Reps
       2:    1        R/Ro Loc [ Rs_Ro    ]
       3:    2        Loc [ Temp_C   ]
       4:    1        Mult
       5:    0        Offset
```

7.8 100 OHM PRT IN 4 WIRE FULL BRIDGE

This example describes obtaining the temperature from a 100 ohm PRT in a 4 wire full bridge (Instruction 6). The temperature being measured is in a constant temperature bath and is to be used as the input for a control algorithm. The PRT in this case does not adhere to the DIN standard (alpha = 0.00385) used in the temperature calculating Instruction 16. Alpha is defined as $((R_{100}/R_0)-1)/100$, where R_{100} and R_0 are the resistances of the PRT at 100°C and 0°C, respectively. In this PRT alpha is equal to 0.00392.

The result given by Instruction 6 (X) is 1000 V_s/V_x (where V_s is the measured bridge output voltage, and V_x is the excitation voltage) which is:

$$X = 1000\ (R_s/(R_s+R_1)-R_3/(R_2+R_3))$$

The resistance of the PRT (R_s) is calculated with the Bridge Transform Instruction 59:

$$R_s = R_1\ X'/(1-X')$$

Where

$$X' = X/1000 + R_3/(R_2+R_3)$$

Thus, to obtain the value R_s/R_0, ($R_0 = R_s$ @ 0°C) for the temperature calculating Instruction 16, the multiplier and offset used in Instruction 6 are 0.001 and $R_3/(R_2+R_3)$, respectively. The multiplier used in Instruction 59 to obtain R_s/R_0 is R_1/R_0 (5000/100 = 50).

It is desired to control the temperature bath at 50°C with as little variation as possible. High resolution is needed so that the control algorithm will be able to respond to minute changes in temperature. The highest resolution is obtained when the temperature range results in an output voltage (V_s) range which fills the measurement range selected in Instruction 6. The full bridge configuration allows the bridge to be balanced (V_s = 0V) at or near the control temperature. Thus, the output voltage can go both positive and negative as the bath temperature changes, allowing the full use of the measurement range.

The resistance of the PRT is approximately 119.7 ohms at 50°C. The 120 ohm fixed resistor balances the bridge at approximately 51°C. The output voltage is:

$$V_s = V_x\ [R_s/(R_s+R_1) - R_3/(R_2+R_3)]$$

$$= Vx\ [R_s/(R_s+5000) - 0.023438]$$

The temperature range to be covered is ±50 ±10°C. At 40°C R_s is approximately 115.8 ohms, or:

$$V_s = -8.0224 \times 10^{-4}\ V_x$$

Even with an excitation voltage (V_X) equal to 2500 mV, V_S can be measured on the +2.5 mV scale (40°C = 115.8 ohms = -2.006 mV, 60°C = 123.6 ohms = 1.714 mV). There is a change of approximately 2 mV from the output at 40°C to the output at 51°C, or 181 µV/°C. With a resolution of 0.33 µV on the 2.5 mV range, this means that the temperature resolution is 0.0018°C.

The 5 ppm per °C temperature coefficient of the fixed resistors was chosen so that their 0.01% accuracy tolerance would hold over the desired temperature range.

The relationship between temperature and PRT resistance is slightly nonlinear one. Instruction 16 computes this relationship for a DIN standard PRT where the nominal temperature coefficient is 0.00385/°C. The change in nonlinearity of a PRT with the temperature coefficient of 0.00392/°C is minute compared with the slope change. Entering a slope correction factor of 0.00385/0.00392 = 0.98214 as the multiplier in Instruction 16 results in a calculated temperature which is well within the accuracy specifications of the PRT.

PROGRAM

```
01:  Full Bridge (P6)
     1:    1          Reps
     2:   21          ±2.5 mV 60 Hz Rejection
                      Range
     3:    1          DIFF Channel
     4:    1          Excite all reps w/Exchan 1
     5: 2500          mV Excitation
     6:   11          Loc [ Rs_Ro    ]
     7:      .001     Mult
     8:      .02344   Offset

02:  BR Transform Rf[X/(1-X)] (P59)
     1:    1          Reps
     2:   11          Loc [ Rs_Ro    ]
     3:   50          Multiplier (Rf)

03:  Temperature RTD (P16)
     1:    1          Reps
     2:   11          R/Ro Loc [ Rs_Ro    ]
     3:   12          Loc [ Temp_C   ]
     4:      .98214   Mult
     5:    0          Offset
```

7.9 PRESSURE TRANSDUCER - 4 WIRE FULL BRIDGE

This example describes a measurement made with a Druck PDCR 1230 depth measurement pressure transducer. The pressure transducer was ordered for use with 5 volt positive or negative excitation and has a range of 5 psi or about 3.5 meters of water. The transducer is used to measure the depth of water in a stilling well.

Instruction 6, 4 Wire Full Bridge, is used to measure the pressure transducer. The high output of the semiconductor strain gage necessitates the use of the 25 mV input range. The sensor is calibrated by connecting it to the CR510 and using Instruction 6, an excitation voltage of 2500 mV, a multiplier of 1 and an offset of 0, noting the readings (∗6 Mode) with 10 cm of water above the sensor and with 334.6 cm of water above the sensor. The output of Instruction 6 is 1000 V_S/V_X or millivolts per volt excitation. At 10 cm the reading is 0.19963 mV/V and at 334.6 cm the reading is 6.6485 mV/V. The multiplier to yield output in cm is:

(334.6 - 10)/(6.6485-.19963) =
50.334 cm/mV/V

The offset is determined after the pressure transducer is installed in the stilling well. The sensor is installed 65 cm below the water level at the time of installation. The depth of water at this time is determined to be 72.6 cm relative to the desired reference. When programmed with the multiplier determined above and an offset of 0, a reading of 65.12 is obtained. The offset for the actual measurements is thus determined to be 72.6 - 65.12 = 7.48 cm.

The lead length is approximately 10 feet, so there is no appreciable error due to lead wire resistance.

PROGRAM

```
01:  Full Bridge (P6)
     1:    1          Reps
     2:   23          ±25 mV 60 Hz Rejection
                      Range
     3:    1          DIFF Channel
     4:    1          Excite all reps w/Exchan 1
     5: 2500          mV Excitation
     6:    1          Loc [ HT_cm    ]
     7:   50.334      Mult
     8:    7.48       Offset
```

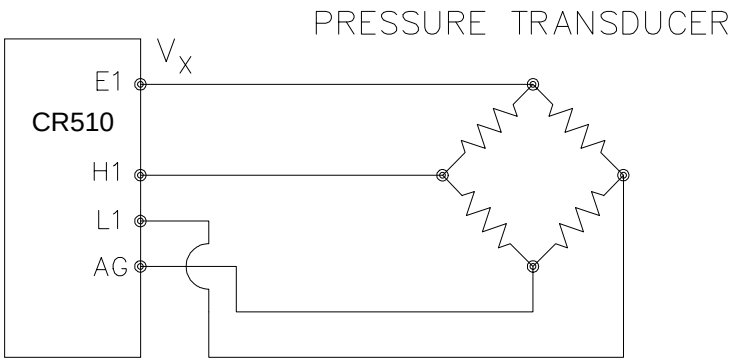


FIGURE 7.9-1. Wiring Diagram for Full Bridge Pressure Transducer

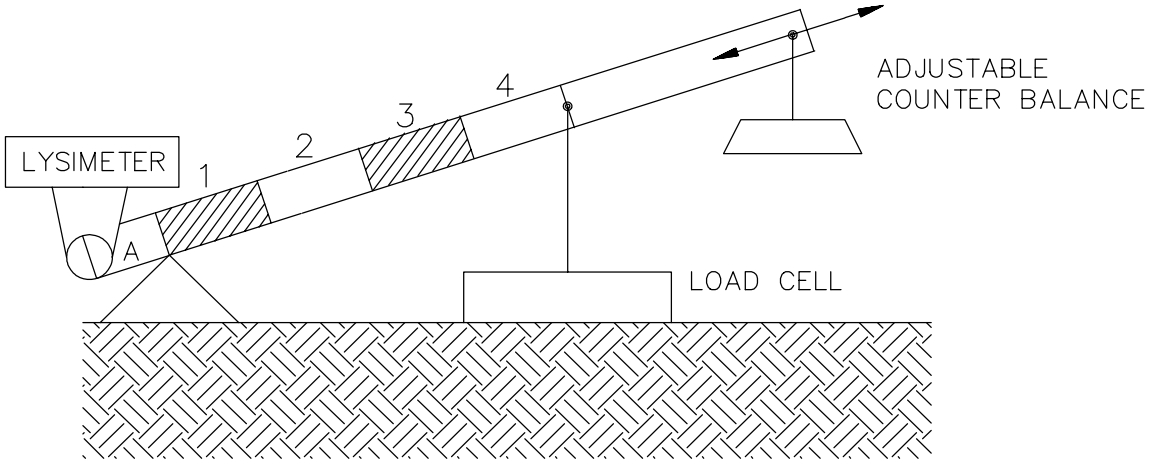


FIGURE 7.10-1. Lysimeter Weighing Mechanism

7.10 LYSIMETER - 6 WIRE FULL BRIDGE

When a long cable is required between a load cell and the CR510, the resistance of the wire can create a substantial error in the measurement if the 4 wire full bridge (Instruction 6) is used to excite and measure the load cell. This error arises because the excitation voltage is lower at the load cell than at the CR510 due to voltage drop in the cable. The 6 wire full bridge (Instruction 9) avoids this problem by measuring the excitation voltage at the load cell. This example shows the errors one would encounter if the actual excitation voltage was not measured and shows the use of a 6 wire full bridge to measure a load cell on a weighing lysimeter (a container buried in the ground, filled with plants and soil, used for measuring evapotranspiration).

The lysimeter is 2 meters in diameter and 1.5 meters deep. The total weight of the lysimeter with its container is approximately 8000 kg. The lysimeter has a mechanically adjustable counter-balance, and changes in weight are measured with a 250 pound (113.6 kg) capacity Sensotec Model 41 tension/compression load cell. The load cell has a 4:1 mechanical advantage on the lysimeter (i.e., a change of 4 kg in the mass of the lysimeter will change the force on the load cell by 1 kg-force or 980 N).

The surface area of the lysimeter is 3.1416 m^2 or 31,416 cm^2, so 1 cm of rainfall or evaporation results in a 31.416 kg change in mass. The load cell can measure ±113.6 kg, a 227 kg range. This represents a maximum change of 909 kg (28 cm of water) in the lysimeter before the counterbalance would have to be readjusted.

There is 1000 feet of 22 AWG cable between the CR510 and the load cell. The output of the load cell is directly proportional to the excitation voltage. When Instruction 6 (4 wire half bridge) is used, the assumption is that the voltage drop in the connecting cable is negligible. The average resistance of 22 AWG wire is 16.5 ohms per 1000 feet. Thus, the resistance in the excitation lead going out to the load cell added to that in the lead coming back to ground is 33 ohms. The resistance

of the bridge in the load cell is 350 ohms. The voltage drop across the load cell is equal to the voltage at the CR510 multiplied by the ratio of the load cell resistance, R_S, to the total resistance, R_T, of the circuit. If Instruction 6 were used to measure the load cell, the excitation voltage actually applied to the load cell, V_1, would be:

$$V_1 = V_x R_s/R_T = V_x 350/(350+33) = 0.91 V_x$$

Where V_x is the excitation voltage. This means that the voltage output by the load cell would only be 91% of that expected. If recording of the lysimeter data was initiated with the load cell output at 0 volts, and 100 mm of evapotranspiration had occurred, calculation of the change with Instruction 6 would indicate that only 91 mm of water had been lost. Because the error is a fixed percentage of the output, the actual magnitude of the error increases with the force applied to the load cell. If the resistance of the wire was constant, one could correct for the voltage drop with a fixed multiplier. However, the resistance of copper changes 0.4% per degree C change in temperature. Assume that the cable between the load cell and the CR510 lays on the soil surface and undergoes a 25°C diurnal temperature fluctuation. If the resistance is 33 ohms at the maximum temperature, then at the minimum temperature, the resistance is:

$$(1-25x0.004)33 \text{ ohms} = 29.7 \text{ ohms}$$

The actual excitation voltage at the load cell is:

$$V_1 = 350/(350+29.7) V_x = .92 V_x$$

The excitation voltage has increased by 1%, relative to the voltage applied at the CR510. In the case where we were recording a 91 mm change in water content, there would be a 1 mm diurnal change in the recorded water content that would actually be due to the change in temperature. Instruction 9 solves this problem by actually measuring the voltage drop across the load cell bridge. The drawbacks to using Instruction 9 are that it requires an extra differential channel and the added expense of a 6 wire cable. In this case, the benefits are worth the expense.

The load cell has a nominal full scale output of 3 millivolts per volt excitation. If the excitation is 2.5 volts, the full scale output is 7.5 millivolts; thus, the ±7.5 millivolt range is selected. The calibrated output of the load cell is 3.106 mV/V_1 at a load of 250 pounds. Output is desired in millimeters of water with respect to a fixed point. The "4" found in equation 7.12-1 is due to the mechanical advantage. The calibration in mV/V_1/mm is:

$$3.106 \text{ mV/V}_1\text{/250 lb x 2.2 lb/kg x } 3.1416 \text{ kg/mm/4} = 0.02147 \text{ mV/V}_1\text{/mm}$$

The reciprocal of this gives the multiplier to convert mV/V_1 into millimeters. (The result of Instruction 9 is the ratio of the output voltage to the actual excitation voltage multiplied by 1000, which is mV/V_1):

$$1/0.02147 \text{ mV/V}_1\text{/mm} = 46.583 \text{ mm/mV/V}_1$$

The output from the load cell is connected so that the voltage increases as the mass of the lysimeter increases. (If the actual mechanical linkage was as shown in Figure 7.10-1, the output voltage would be positive when the load cell was under tension.)

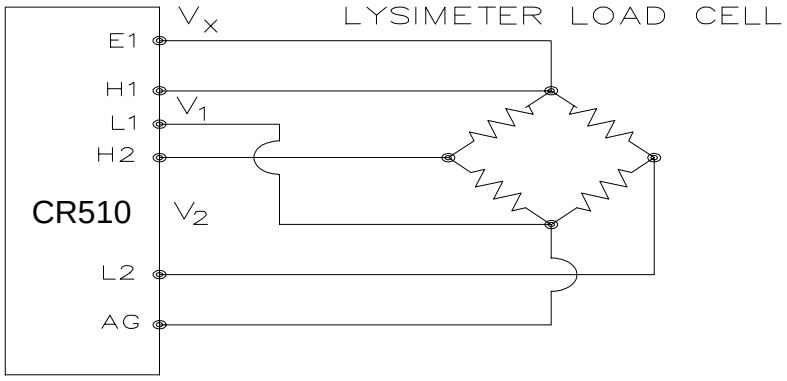


FIGURE 7.10-2. 6 Wire Full Bridge Connection for Load Cell

When the experiment is started, the water content of the soil in the lysimeter is approximately 25% on a volume basis. It is decided to use this as the reference (i.e., 0.25 x 1500 mm = 375 mm). The experiment is started at the beginning of what is expected to be a period during which evapotranspiration exceeds precipitation. Instruction 9 is programmed with the correct multiplier and no offset. After hooking everything up, the counterbalance is adjusted so that the load cell is near the top of its range; this will allow a longer period before readjustment is necessary. The result of Instruction 9 (monitored with the ∗6 Mode) is 109. The offset needed to give the desired initial value of 375 mm is 266. However, it is decided to add this offset in a separate instruction so the result of Instruction 9 can be used as a ready reminder of the strain on the load cell (range = ±140 mm). When the strain on the load cell nears its rated limits, the counterbalance is readjusted and the offset recalculated to provide a continuous record of the water budget.

The program table has an execution interval of 10 seconds. The average value in millimeters is output to Final Storage (not shown in Table) every hour. The average is used, instead of a sample, in order to cancel out effects of wind loading on the lysimeter.

PROGRAM

```
01:   Full Bridge w/mv Excit (P9)
        1:     1        Reps
        2:    25        ±2500 mV 60 Hz
                        Rejection Ex Range
        3:    22        ±7.5 mV 60 Hz Rejection
                        Br Range
        4:     1        DIFF Channel
        5:     1        Excite all reps w/Exchan 1
        6:  2500        mV Excitation
        7:     1        Loc [ Raw_mm    ]
        8:   46.583     Mult
        9:     0        Offset

02:   Z=X+F (P34)
        1:     1        X Loc [ Raw_mm    ]
        2:   266        F
        3:     2        Z Loc [ mm_H2O    ]
```

7.11 227 GYPSUM SOIL MOISTURE BLOCK

Soil moisture is measured with a gypsum block by relating the change in moisture to the change in resistance of the block. An AC Half Bridge (Instruction 5) is used to determine the resistance of the gypsum block. Rapid reversal of the excitation voltage inhibits polarization of the sensor. Polarization creates an error in the output so the fast integration option is used. The output of Instruction 5 is the ratio of the output voltage to the excitation voltage; this output is converted to gypsum block resistance with Instruction 59, Bridge Transform.

The Campbell Scientific 227 Soil Moisture Block uses a Delmhorst gypsum block with a 1 kohm bridge completion resistor. Using data supplied by Delmhorst, Campbell Scientific has computed coefficients for a 5th order polynomial to convert block resistance to water potential in bars. There are two polynomials: one to optimize the range from -0.1 to -2 bars, and one to cover the range from -0.1 to -10 bars (the minus sign is omitted in the output). The -0.1 to -2 bar polynomial requires a multiplier of 1 in the Bridge Transform Instruction (result in kohms) and the -0.1 to -10 bar polynomial requires a multiplier of 0.1 (result in 10,000s of ohms). The multiplier is a scaling factor to maintain the maximum number of significant digits in the polynomial coefficients.

In this example, we wish to make measurements on four gypsum blocks and output the final data in bars. The soil where the moisture measurements are to be made is quite wet at the time the data logging is initiated, but is expected to dry beyond the -2 bar limit of the wet range polynomial. The dry range polynomial is used, so a multiplier of 0.1 is entered in the bridge transform instruction.

When the water potential is computed, it is written over the resistance value. The potentials are stored in Input Locations 1-4 where they may be accessed for output to Final Storage. If it was desired to retain the resistance values, the potential measurements could be stored in Locations 5-8 by changing the value in Parameter 3 to 7 in Instruction 55.

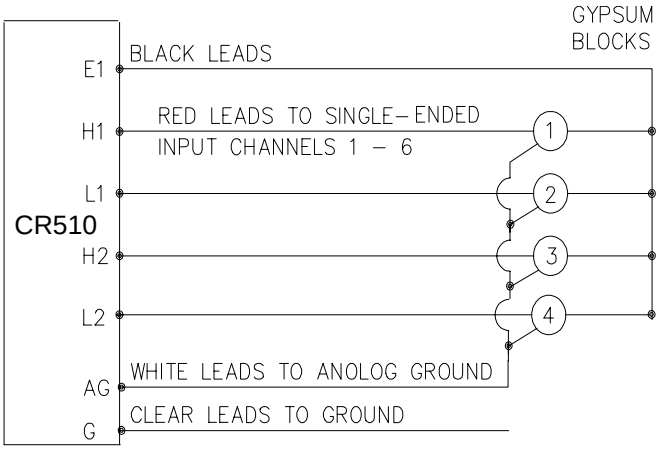


FIGURE 7.11-1. 6 227 Gypsum Blocks Connected to the CR510

PROGRAM

```
01:  AC Half Bridge (P5)
     1:    4          Reps
     2:   15          ±2500 mV Fast Range
     3:    1          SE Channel
     4:    1          Ex Channel Option
     5: 2500          mV Excitation
     6:    1          Loc [ H2O_bar_1 ]
     7:    1          Mult
     8:    0          Offset

02:  BR Transform Rf[X/(1-X)] (P59)
     1:    4          Reps
     2:    1          Loc [ H2O_bar_1 ]
     3:   .1          Multiplier (Rf)

03:  Polynomial (P55)
     1:    4          Reps
     2:    1          X Loc [ H2O_bar_1 ]
     3:    1          F(X) Loc [ H2O_bar_1 ]
     4:   .15836      C0
     5:  6.1445       C1
     6: -8.4189       C2
     7:  9.2493       C3
     8: -3.1685       C4
     9:   .33392      C5
```

7.12 NONLINEAR THERMISTOR IN HALF BRIDGE

Instruction 11, 107 Thermistor Probe, automatically linearizes the output of a nonlinear thermistor, 107 Probe, by transforming the millivolt reading with a 5th order polynomial. Instruction 55, Polynomial, can be used to calculate temperature of any nonlinear thermistor, provided the correlation between temperature and probe output is known, and an appropriate polynomial fit has been determined. In this example, the CR510 is used to measure the temperature of four 101 Probes (used with the CR21 but usually not the CR510). Instruction 4, Excite, Delay, and Measure, is used because the high source resistance of the probe requires a long input settling time (Section 10.3.1). The excitation voltage is 2000 mV, the same as used in the CR21. The signal voltage is then transformed to temperature using the Polynomial Instruction.

The manual for the 101 Probe gives the coefficients of the 5th order polynomial used to convert the output in millivolts to temperature (E denotes the power of 10 by which the mantissa is multiplied):

C0	-53.7842
C1	0.147974
C2	-2.18755E-4
C3	2.19046E-7
C4	-1.11341E-10
C5	2.33651E-14

The CR510 will only allow 5 significant digits to the right or left of the decimal point to be entered from the keyboard. The polynomial cannot be applied exactly as given in the 101 manual. The initial millivolt reading must be scaled if the coefficients of the higher order terms are to be entered with the maximum number of significant digits. If 0.001 is used as a multiplier on the millivolt output, the coefficients are divided by 0.001 raised to the appropriate power (i.e., C0=C0, C1=C1/0.001, C2=C2/.000001, etc.). With this adjustment, the coefficients entered in Parameters 4-9 of Instruction 55 become:

C0	-53.784
C1	147.97
C2	-218.76
C3	219.05
C4	-111.34
C5	23.365

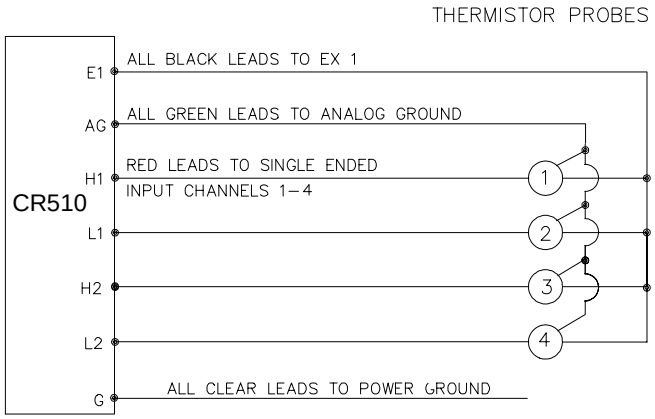


FIGURE 7.12-1. Nonlinear Thermistor Probes Connected to CR510

PROGRAM

```
01:   Excite-Delay (SE) (P4)
        1:     4          Reps
        2:    25          ±2500 mV 60 Hz
                          Rejection Range
        3:     1          SE Channel
        4:     1          Excite all reps w/Exchan 1
        5:    10          Delay (units 0.01 sec)
        6:  2000          mV Excitation
        7:     1          Loc [ Temp_C_1  ]
        8:     .001       Mult
        9:     0          Offset

02:   Polynomial (P55)
        1:     4          Reps
        2:     1          X Loc [ Temp_C_1  ]
        3:     1          F(X) Loc [ Temp_C_1  ]
        4:   -53.784      C0
        5:   147.97       C1
        6:  -218.76       C2
        7:   219.05       C3
        8:  -111.34       C4
        9:    23.365      C5
```

7.13 WATER LEVEL - GEOKON'S VIBRATING WIRE PRESSURE SENSOR

The vibrating wire sensor utilizes a change in the frequency of a vibrating wire to sense pressure. Figure 7.13-1 illustrates how an increase in pressure on the diaphragm decreases the tension on the wire attached to the diaphragm. A decrease in the wire tension decreases the resonant frequency in the same way that loosening a guitar string decreases its frequency.

Vibrating Wire Measurement Instruction 28 excites the "plucking" and "pickup" coils shown in Figure 7.13-1 with a "swept" frequency. A "swept" frequency is a group of different frequencies that are sent one right after another starting with the lowest frequency and ending with the highest. The lowest and highest frequencies are entered by the user in units of hundreds of Hz. This swept frequency causes the wire to vibrate at each of the individual frequencies. Ideally, all of the frequencies except the one matching the resonant frequency of the wire will die out in a very short time. The wire will vibrate with the resonant frequency for a relatively long period of time, cutting the lines of flux in the "plucking" and "pickup" coils and inducing the same frequency on the lines to the CR510. Instruction 28 then accurately measures how much time it takes to receive a user specified number of cycles.

The vibrating wire requires temperature compensation. A nonlinear thermistor built into the probe is measured using Instruction 4, a single-ended half bridge measurement with excitation, and calculated with Instruction 55, a fifth order polynomial instruction.

Campbell Scientific's AVW1 Vibrating Wire Sensor Interface is required between the sensor to the datalogger. The purpose is twofold:

- 5 or 12 volts can be used as the potential in the swept frequency excitation, thus plucking the wire harder than the maximum 2.5 volt switched excitation. The result is a larger magnitude signal for a longer time.
- A transformer strips off any DC noise on the signal, improving the ability to detect cycles.

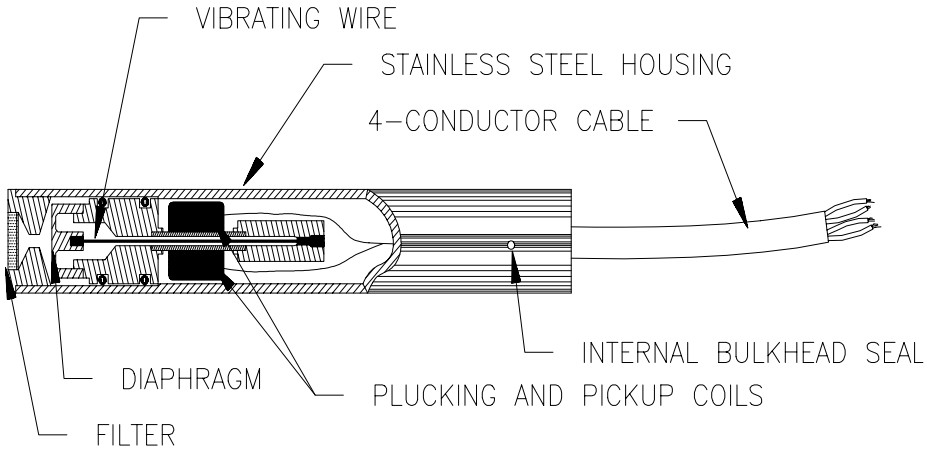


FIGURE 7.13-1. A Vibrating Wire Sensor

The following calculations are based on using a Geokon model 4500 Vibrating Wire sensor. An individual multiplier and offset must be calculated for each sensor used in a system.

MULTIPLIER

The fundamental equation relating frequency to pressure is

$$P = -F_x G + B \quad \text{where}$$

- P = pressure, PSI
- G = the Gage Factor obtained from the sensors calibration sheet in PSI/digit. The units of a digit are $Hz^2(10^{-3})$.
- B = offset
- $F_x = f^2Hz^2(10^{-3})$, where f is frequency.

Instruction 28 measures period, T, of the vibrating wire in milliseconds (ms) and returns a measured value, X, of

$$X = 1/(T^2(ms)^2) = f^2(10^{-6})Hz^2$$

A multiplier of -1000 in Instruction 28 converts the measurement to digits, as shown below.

$$-F_x = -X(-10^3) = -f^2(10^{-3})Hz^2$$

To calculate the multiplier, convert Geokon's gage factor, G, to the desired units (i.e., feet of water per digit) and multiply by -1000 digits/kHz^2.

TEMPERATURE CORRECTION

The temperature correction is applied as follows.

$$P_T = P + C * (t_1 - t_0), \text{ where}$$

- P_T = Pressure corrected for temperature, °C
- C = Temperature coefficient, PSI/°C (from Geokon calibration sheet)
- t_0 & t_1 = Initial and current temperatures, °C.

The temperature coefficient, C, must be converted to units compatible with the gage factor, G.

WELL MONITORING EXAMPLE

In this example the vibrating wire sensor is used to monitor water table height (Figure 7.13-2). The desired data is the distance from the lip of the well to the water surface. The sensor is vented to atmosphere to eliminate measurement errors due to changes in barometric pressure. The water level is expected to stay within 40 to 80 feet of the lip so the 50 psi pressure sensor is placed approximately 100 feet below the lip of the well. The calibration data from Geokon is provided in Table 7.13-1.

TABLE 7.13-1 Calibration Data for Sensor 3998

Gage Factor (psi/digit)	Temp. Coeff. (psi/°C)
0.0151	-0.0698

The multiplier, m, is calculated to convert the reading to feet of water.

m = 0.0151 (psi/digit) ∗ 2.3067 (ft of water/psi) ∗ -1000 digits/kHz^2 = -34.831 ft of water/kHz^2

After the probe reaches thermal equilibrium, the initial temperature, t_0, is measured to be 24°C.

The water column above the sensor is referred to as the "Reading". The Reading decreases with increasing "Distance" from lip of well to water surface so the Distance is computed by subtracting the Reading from the Offset as shown in Figure 7.13-2.

The "Initial Distance" to the water surface is measured with a chalked line to be 47.23 feet below the lip. The first time the program is executed, the program calculates the offset (Offset = Distance + Reading) required to obtain a reading of 47.23 feet. The offset is stored in Location 4 and applied to subsequent measurements.

NOTE: Following program compilation in the ∗0 Mode, all input locations are set to zero. This fact is utilized to detect the first execution following a program compilation.

The example assumes the sensor has been connected as shown here.

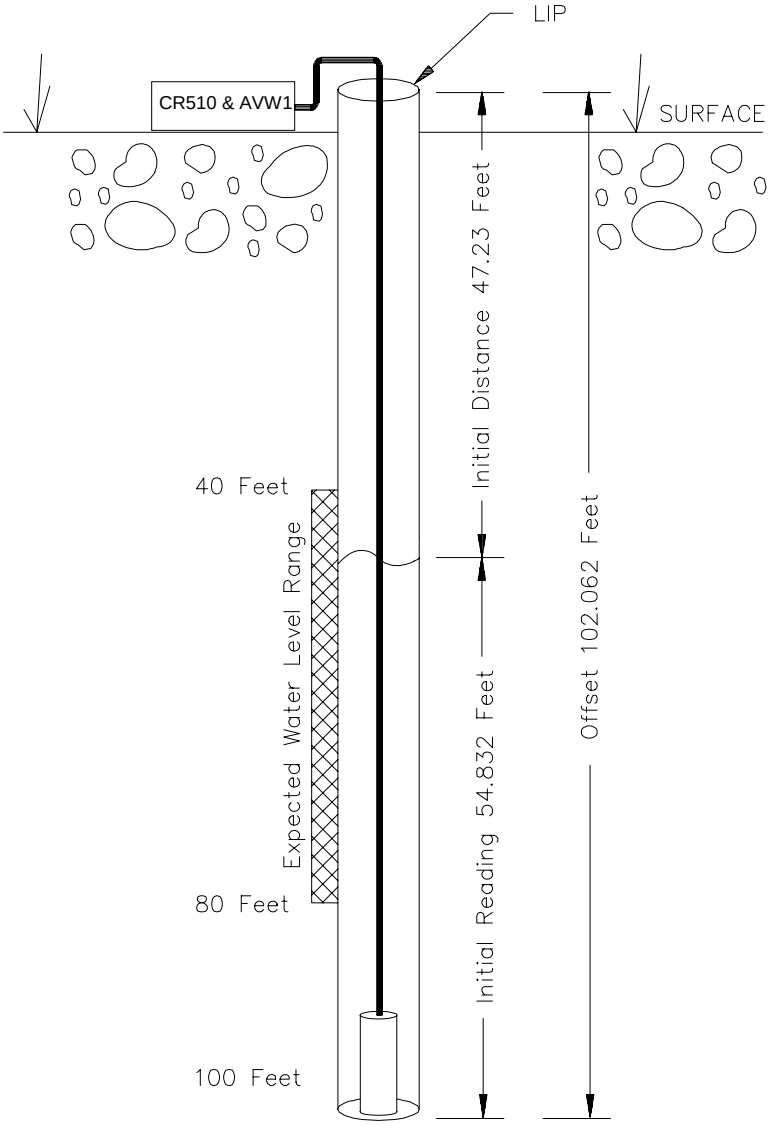


FIGURE 7.13-2. Well Monitoring Example

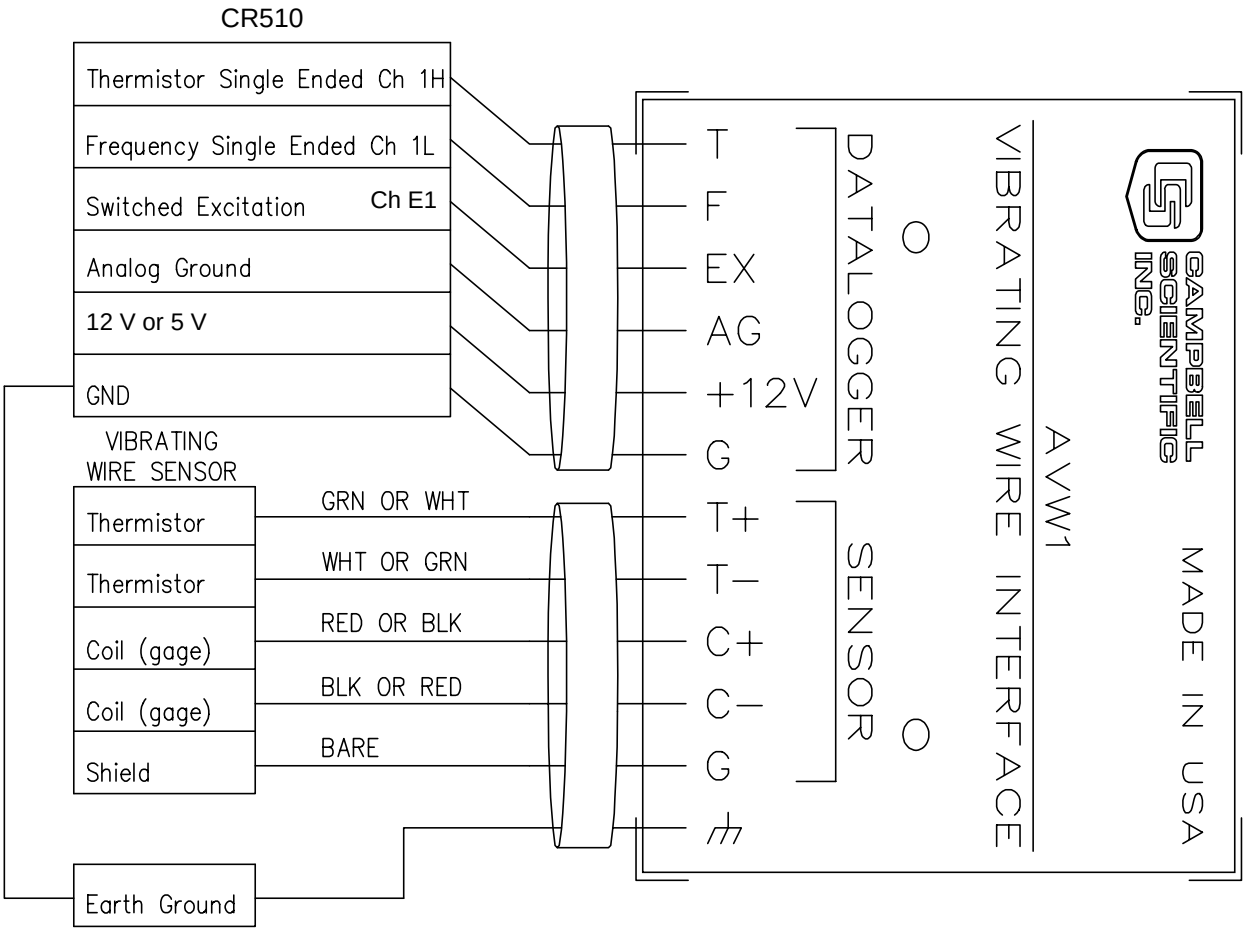


FIGURE 7.13-3. Hook up to AVW1

PROGRAM

AVW1 & CR510 USED TO MEASURE 1
GEOKON VIBRATING WIRE SENSOR.

* Table 1 Program
01: 60 Execution Interval (seconds)

01: Excite-Delay (SE) (P4)
1: 1 Reps
2: 15 ±2500 mV Fast Range
3: 1 SE Channel
4: 1 Excite all reps w/Exchan 1
5: 1 Delay (units 0.01 sec)
6: 2500 mV Excitation
7: 1 Loc [Temp]
8: .001 Mult
9: 0 Offset

02: Polynomial (P55)
1: 1 Reps
2: 1 X Loc [Temp]
3: 1 F(X) Loc [Temp]
4: -104.78 C0
5: 378.11 C1
6: -611.59 C2
7: 544.27 C3
8: -240.91 C4
9: 43.089 C5

03: Vibrating Wire (SE) (P28)
1: 1 Reps
2: 2 SE Channel
3: 1 Excite all reps w/Exchan 1
4: 24 Starting Freq. (units = 100 Hz)
5: 32 End Freq. (units = 100 Hz)
6: 500 No. of Cycles
7: 0 Rep Delay (units = 0.01 sec)
8: 2 Loc [Pressure]
9: -34.836 Mult
10: 0 Offset

```
04:   Z=X+F (P34)
        1:     1          X Loc [ Temp      ]
        2:   -24          F
        3:     3          Z Loc [ Temp_Comp ]

05:   Z=X*F (P37)
        1:     3          X Loc [ Temp_Comp ]
        2:    -.0698      F
        3:     3          Z Loc [ Temp_Comp ]

06:   Z=X+Y (P33)
        1:     3          X Loc [ Temp_Comp ]
        2:     2          Y Loc [ Pressure  ]
        3:     2          Z Loc [ Pressure  ]

07:   IF (X<=>F) (P89)
        1:     5          X Loc [ Cmpile_Ck ]
        2:     1          =
        3:     0          F
        4:    30          Then Do

08:   Z=X+F (P34)
        1:     2          X Loc [ Pressure  ]
        2:    47.23       F
        3:     4          Z Loc [ Offset    ]

09:   Z=F (P30)
        1:     1          F
        2:     0          Exponent of 10
        3:     5          Z Loc [ Cmpile_Ck ]

10:   End (P95)

11:   Z=X-Y (P35)
        1:     4          X Loc [ Offset    ]
        2:     2          Y Loc [ Pressure  ]
        3:     6          Z Loc [ Distance  ]
```

7.14 4 TO 20 MA SENSOR USING CURS100 TERMINAL INPUT MODULE

A dew point sensor has a 4 to 20 mA output over the dew point temperature range of -40° to +70°C. The dew point sensor output may be measured by the CR510 using the CUS100 Terminal Input Module (TIM). The CUS100 uses a 100 Ω, ± 0.01 % resistor to convert the 4 to 20 mA range to 400 to 2000 mV. The millivolt range was found using the relationship V = IR, where V is voltage, I is current, and R is resistance, e.g. the voltage at -40°C is given by V = 4 mA ∗ 100 Ω = 400 mV. The dew point sensor is measured with Instruction 2 (Volt Diff). The multiplier for dew point temperature is found with the following relationship [70°C - (-40°C)] / [2000 mV - 400 mV] = 0.06875°C/mV. The offset is found by taking the linear relationship °C = mV ∗ Mult + Offset and solving for the Offset. At -40°C the voltage is 400 mV, thus the Offset = -40 - [400 mV ∗ 0.06875°C/mV] = -67.5°C.

CONNECTIONS
The dew point sensor is measured with a differential voltage measurement on differential analog input 1. The CURS100 TIM and dew point sensor are wired to the CR510 terminal strip panel as shown in Figure 7.14-1.

PROGRAM

```
01:   Volt (Diff) (P2)
        1:     1          Reps
        2:    25          ±2500 mV 60 Hz Rejection
                          Range
        3:     1          DIFF Channel
        4:     1          Loc [ Dew_Pnt_C ]
        5:      .06875    Mult
        6:   -67.5        Offset
```

INPUT LOCATIONS

1 Dew_Pnt_C

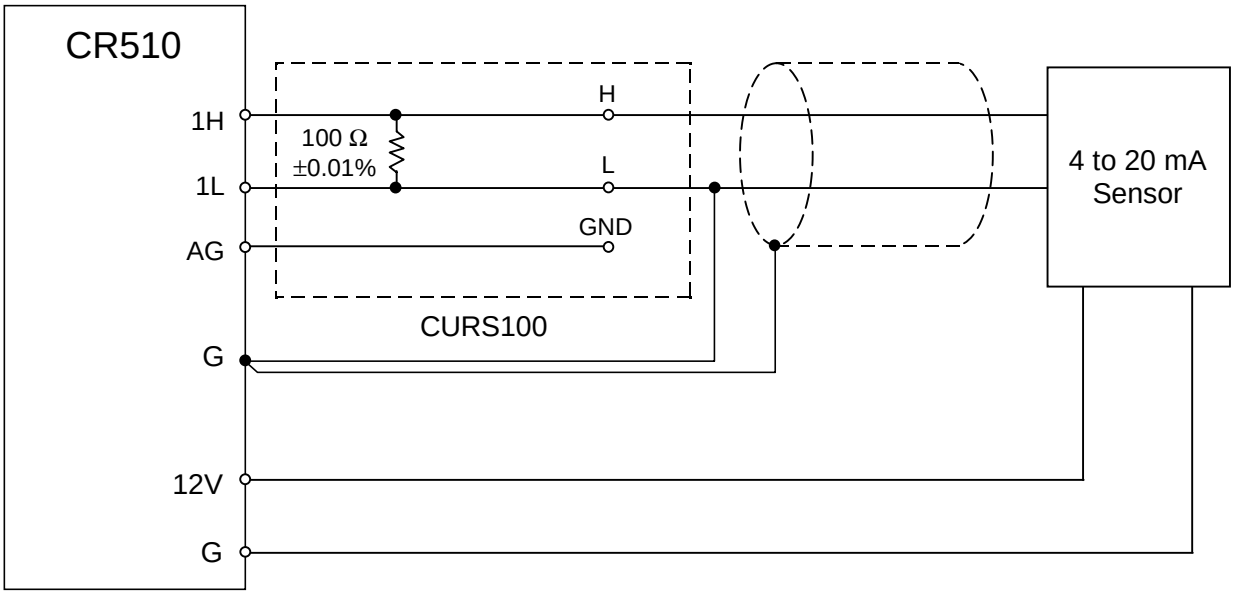


FIGURE 7.14-1 Wiring Diagram for CURS100 Terminal Input Module and 4 to 20 mA Sensor.

SECTION 8. PROCESSING AND PROGRAM CONTROL EXAMPLES

The following examples are intended to illustrate the use of Processing and Program Control Instructions, flags, dual Final Storage, and the capability to direct the results of Output Processing Instructions to Input Storage.

The specific examples may not be as important as some of the techniques employed, for example:

Directing Output Processing to Input Storage is used in the Running Average and Rainfall Intensity examples (8.1 and 8.2).

Flag tests are used in the Running Average, Interrupt Subroutine, Converting Wind Direction, and Saving Data Prior to Event examples (8.1, 8.5, 8.7 and 8.8).

An algorithm for a down counter is used in the Saving Data Prior to Event example (8.8).

As in Section 7 these examples are not complete programs to be taken verbatim. They need to be altered to fit specific needs.

8.1 COMPUTATION OF RUNNING AVERAGE

It is sometimes necessary to compute a running average (i.e., the average covers a fixed number of samples and is continuously updated as new samples are taken). Because the output interval is shorter than the averaging period, Instruction 71 cannot be used; the algorithm for computing this average must be programmed by the user. The following example demonstrates a program for computing a running average.

In this example, each time a new measurement is made (in this case the CR510 internal temperature) an average is computed for the 10 most recent samples. This is done by saving all 10 temperatures in contiguous input locations and using the Spatial Average Instruction (51) to compute the average. The temperatures are stored in locations 2 through 11. Each time the table is executed, the new measurement is stored in location 11 and the average is stored in location 1. The Block Move Instruction (54) is then used to move the temperatures from locations 3 through 11 down by 1 location; the oldest measurement (in location 3) is lost when the temperature from location 4 is written over it.

PROGRAM

```
*       Table 1 Program
  01:       10.0        Execution Interval (seconds)

01:     Internal Temperature (P17)
          1:  11        Loc [ Temp_i ]

02:     Spatial Average (P51)
          1:  10        Swath
          2:   2        First Loc [ Temp_i_9  ]
          3:   1        Avg Loc [ Av_10smpl ]

03:     Block Move (P54)
          1:   9        No. of Values
          2:   3        First Source Loc [ Temp_i_8  ]
          3:   1        Source Step
          4:   2        First Destination Loc [ Temp_i_9  ]
          5:   1        Destination Step

04:     Do (P86)
          1:  10        Set Output Flag High

05:     Sample (P70)
          1:   1        Reps
          2:   1        Loc [ Av_10smpl ]
```

INPUT LOCATIONS

```
1 Av_10smpl       7 Temp_i_4
2 Temp_i_#9       8 Temp_i_3
3 Temp_i_8        9 Temp_i_2
4 Temp_i_7       10 Temp_i_1
5 Temp_i_6       11 Temp_i
6 Temp_i_5
```

In the above example, all samples for the average are stored in input locations. This is necessary when an average must be output with each new sample. In most cases, averages are desired less frequently than sampling. For example, it may be necessary to sample some parameter every 5 seconds and output every hour an average of the previous three hours' readings. If all samples were saved, this would require 2160 input locations. The same value can be obtained by computing an hourly average and averaging the hourly averages for the past three hours. To do this requires that hourly averages be stored in input locations.

Instruction 80 is used to send the 1 hour average to Input Storage and again to send the 3 hour average to Final Storage.

PROGRAM

```
*       Table 1 Program
  01:       5.0         Execution Interval (seconds)

01:   Volt (Diff) (P2)
        1:      1       Reps
        2:     25       ±2500 mV 60 Hz
                        Rejection Range
        3:      1       DIFF Channel
        4:      5       Loc [ XX_mg_M3  ]
        5:     10       Mult
        6:      0       Offset

02:   If time is (P92)
        1:      0       Minutes (Seconds --) into a
        2:     60       Interval (same units as
                        above)
        3:     10       Set Output Flag High

03:   Set Active Storage Area (P80)
        1:      3       Input Storage Area
        2:      3       Array ID or Loc [ avg_i     ]

04:   Average (P71)
        1:      1       Reps
        2:      5       Loc [ XX_mg_M3  ]

05:   Spatial Average (P51)
        1:      3       Swath
        2:      1       First Loc [ avg_i_2   ]
        3:      4       Avg Loc [ 3_Hr_avg  ]

06:   Set Active Storage Area (P80)
        1:      1       Final Storage Area 1
        2:     25       Array ID or Loc [ _________ ]

07:   Real Time (P77)
        1:  0110        Day,Hour/Minute

08:   Sample (P70)
        1:      1       Reps
        2:      4       Loc [ 3_Hr_avg  ]

09:   If Flag/Port (P91)
        1:     10       Do if Output Flag is High
                        (Flag 0)
        2:     30       Then Do

10:   Block Move (P54)
        1:      2       No. of Values
        2:      2       First Source Loc
                        [ avg_i_1   ]
        3:      1       Source Step
        4:      1       First Destination Loc
                        [ avg_i_2   ]
        5:      1       Destination Step

11:   End (P95)
```

INPUT LOCATIONS

```
1 avg_i_2
2 avg_i_1
3 avg_i
4 3_Hr_avg
5 XX_mg_M3
```

8.2 RAINFALL INTENSITY

In this example, the total rain for the last 15 minutes is output only if any rain has occurred. The program makes use of the capability to direct the output of Output Processing Instructions to Input Storage.

Every 15 minutes, the total rain is sent to Input Storage. If the total is not equal to 0, output is redirected to Final Storage Area 1, the time is output and the total is sampled.

PROGRAM

```
*       Table 1 Program
  01:       60.0        Execution Interval (seconds)

01:   Pulse (P3)
        1:      1       Reps
        2:      1       Pulse Input Channel
        3:      2       Switch Closure
        4:      1       Loc [ Precip_mm ]
        5:      .254    Mult
        6:      0       Offset
```

```
02:   If time is (P92)
        1:     0          Minutes (Seconds --) into a
        2:    15          Interval (same units as above)
        3:    10          Set Output Flag High

03:   Set Active Storage Area (P80)
        1:     3          Input Storage Area
        2:     2          Array ID or Loc [ 15min_tot ]

04:   Totalize (P72)
        1:     1          Reps
        2:     1          Loc [ Precip_mm ]

05:   IF (X<=>F) (P89)
        1:     2          X Loc [ 15min_tot ]
        2:     2          <>
        3:     0          F
        4:    30          Then Do

06:   Set Active Storage Area (P80)
        1:     1          Final Storage Area 1
        2:    25          Array ID or Loc
                          [ _____ ]

07:   Real Time (P77)
        1:  0110          Day,Hour/Minute

08:   Sample (P70)
        1:     1          Reps
        2:     2          Loc [ 15min_tot ]

09:   End (P95)
```

INPUT LOCATIONS

1 Precip_mm
2 15min_tot

8.3 SUB 1 MINUTE OUTPUT INTERVAL SYNCHED TO REAL TIME

Output can be synchronized to seconds by pressing "-" or "C" while entering the first parameter in Instruction 92. If a counter, incremented within the program, was used to determine when to set the Output Flag, output would depend on the number of times the table was executed. The actual time of output would depend on when the program was actually compiled and started running. If the table overran its execution interval (Section 1.1.1), the output interval would not be the count multiplied by the execution interval, but some longer interval.

In this example a temperature (107 Temperature Probe) is measured every 0.5 seconds and the average output every 30 seconds.

PROGRAM

```
*     Table 1 Program
  01:     0.5       Execution Interval (seconds)

01:   Temp (107) (P11)
        1:   1        Reps
        2:   1        SE Channel
        3:   1        Excite all reps w/E1
        4:   1        Loc [ Temp   ]
        5:   1.0      Mult
        6:   0.0      Offset

02:   If time is (P92)
        1:   0--      Minutes (Seconds --) into a
        2:  30        Interval (same units as above)
        3:  10        Set Output Flag High

03:   Average (P71)
        1:   1        Reps
        2:   2        Loc [ TC_Temp   ]
```

INPUT LOCATIONS

1 Ref_Temp
2 TC_Temp

8.4 SWITCH CLOSURES ON CONTROL PORTS (RAIN GAGE)

Control port 2 can be used to measure switch closures up to 40 Hz. Instruction 3, pulse, is used to measure two rain gages on pulse input 1, and a rain gage with control port 2. This is done as a comparison. In a real application the pulse channel would be used for wind speed and a control port for a rain gage. The rain gage is connected as diagrammed below.

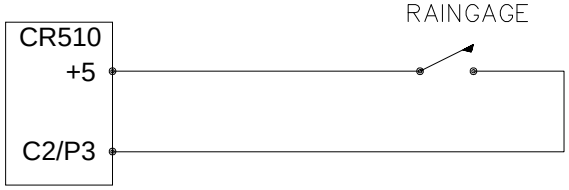


FIGURE 8.4-1. Connections for Rain Gage

PROGRAM

```
*       Table 1 Program
  01:       10.0        Execution Interval (seconds)

01:     Pulse (P3)
        1:      1       Reps
        2:      1       Pulse Input Channel
        3:      2       Switch Closure
        4:     10       Loc [ Precip_1  ]
        5:      .254    Mult
        6:      0       Offset

02:     Pulse (P3)
        1:      1       Reps
        2:      3       Pulse Channel 3
        3:      2       Switch Closure
        4:     11       Loc [ Precip_2  ]
        5:      .254    Mult
        6:      0       Offset

03:     If time is (P92)
        1:      0       Minutes (Seconds --) into a
        2:     60       Interval (same units as above)
        3:     10       Set Output Flag High

04:     Real Time (P77)
        1:  0110        Day,Hour/Minute

05:     Totalize (P72)
        1:      2       Reps
        2:     10       Loc [ Precip_1  ]
```

INPUT LOCATIONS

```
10 Precip_1
11 Precip_2
```

8.5 CONVERTING 0-360 WIND DIRECTION OUTPUT TO 0-540 FOR STRIP CHART

If 0-360 degree wind direction is output to a strip chart the discontinuity at 0/360 will cause the pen to jump back and forth full scale when the winds are varying from the north. In the days of strip charts this was solved with a 0-540 degree pot on the wind vane (direction changes from 540 to 180 and from 0 to 360 so the pen only jumps once when the wind is out of the north or south).

When faced with the necessity of strip chart output, the following algorithm can be used to change a 0-360 degree input to 0-540. (If you have a 0-540 pot, it can be used with the CR510 since the Wind Vector Instruction, 69, will work with this output.)

To change 0-360 degrees to the 0-540 degrees, 360 degrees must sometimes be added to the reading when it is in the range of 0 to 180. The following algorithm does this by assuming that if the previous reading was less than 270, the vane has shifted through 180 degrees and does not need to be altered. If the previous 0-540 reading was greater than 270, 360 degrees is added.

This example is written as a subroutine which is used to output an analog voltage to a strip chart.

```
*       Table 3 Subroutines

01:     Beginning of Subroutine (P85)
        1:    1         Subroutine 1

02:     IF (X<=>F) (P89)
        1:   10         X Loc [ 0_540_WD  ]
        2:    3         >=
        3: 270          F
        4:   30         Then Do

03:     Do (P86)
        1:   11         Set Flag 1 High

04:     Else (P94)

05:     Do (P86)
        1:   21         Set Flag 1 Low

06:     End (P95)

07:     Z=X (P31)
        1:    2         X Loc [ 0_360_WD  ]
        2:   10         Z Loc [ 0_540_WD  ]

08:     IF (X<=>F) (P89)
        1:   10         X Loc [ 0_540_WD  ]
        2:    4         <
        3: 180          F
        4:   30         Then Do

09:     If Flag/Port (P91)
        1:   11         Do if Flag 1 is High
        2:   30         Then Do

10:     Z=X+F (P34)
        1:   10         X Loc [ 0_540_WD  ]
        2: 360          F
        3:   10         Z Loc [ 0_540_WD  ]

11:     Z=X (P31)
        1:   10         X Loc [ 0_540_WD  ]
        2:    6         Z Loc [ 0_540_out ]
```

12: End (P95)

13: End (P95)

14: End (P95)

INPUT LOCATIONS

2 0_360_WD
6 0_540_out
10 0_540_WD

8.6 USE OF 2 FINAL STORAGE AREAS - SAVING DATA PRIOR TO EVENT

One of the uses of 2 Final Storage Areas is to save a fixed amount of data before and after some event.

In this example, a load cell is measured every second. It is assumed that at some random interval the load will exceed 25 pounds for less than 10 seconds. Exceeding 25 pounds is the event to be captured. The data from the 10 seconds before the event and 10 seconds after the event is to be saved (21 seconds including the scan in which the load first exceeds 25 pounds).

Every second the load cell is measured; hours-minutes, seconds, and the load are output to Final Storage Area 2 (4 values with the Array ID). 84 locations are allocated to Final Storage Area 2. Thus, Area 2 holds 21 seconds (4 values/second x 21 seconds = 84 locations).

When 25 pounds is exceeded, 10 is loaded into an input location and flag 1 is set high. The input location is used as a down counter. The flag indicates an event has occurred and prevents the input location from being reloaded until 11 seconds have passed.

The down counter is decremented by 1 each time the table is executed. When it equals 0 all the data in Final Storage Area 2 is transferred to Final Storage Area 1 (using Instruction 96) and Flag 1 is set low.

The down counter is set to 10 instead of 11 because it is decremented after checking to see if it is 0.

PROGRAM

```
*     Table 1 Program
  01:      1          Execution Interval (seconds)

01:   Full Bridge (P6)
      1:     1         Reps
      2:    22         ±7.5 mV 60 Hz Rejection Range
      3:     1         DIFF Channel
      4:     1         Excite all reps w/Exchan 1
      5:  2500         mV Excitation
      6:     1         Loc [ Force_kg  ]
      7:  15.120       Mult
      8:     0         Offset

02:   Do (P86)
      1:    10         Set Output Flag High

03:   Set Active Storage Area (P80)
      1:     2         Final Storage Area 2
      2:    10         Array ID or Loc
                       [ _________ ]

04:   Real Time (P77)
      1:    11         Hour/Minute,Seconds

05:   Sample (P70)
      1:     1         Reps
      2:     1         Loc [ Force_kg  ]

06:   IF (X<=>F) (P89)
      1:     1         X Loc [ Force_kg  ]
      2:     3         >=
      3:    25         F
      4:    30         Then Do

07:   If Flag/Port (P91)
      1:    21         Do if Flag 1 is Low
      2:    30         Then Do

08:   Do (P86)
      1:    11         Set Flag 1 High

09:   Z=F (P30)
      1:    10         F
      2:     0         Exponent of 10
      3:     2         Z Loc [ Down_cnt  ]

10:   End (P95)

11:   End (P95)

12:   IF (X<=>F) (P89)
      1:     2         X Loc [ Down_cnt  ]
      2:     1         =
      3:     0         F
      4:    30         Then Do
```

```
13:   If Flag/Port (P91)
        1:    11        Do if Flag 1 is High
        2:    30        Then Do

14:   Serial Out (P96)
        1:    81        All Data to other FS Area

15:   Do (P86)
        1:    21        Set Flag 1 Low

16:   End (P95)

17:   Else (P94)

18:   Z=X+F (P34)
        1:     2        X Loc [ Down_cnt  ]
        2:    -1        F
        3:     2        Z Loc [ Down_cnt  ]

19:   End (P95)

*     A                 Mode 10 Memory Allocation
  01:       28          Input Locations
  02:       64          Intermediate Locations
  03:       84          Final Storage Area 2
```

INPUT LOCATIONS

1 Force_kg
2 Down_cnt

8.7 LOGARITHMIC SAMPLING USING LOOPS

A ground water pump test requires that water level be measured and recorded according to the following schedule.

Time into Test Minutes	Output Interval	Output Interval Loop #
00 to 10	10 sec.	1
10 to 30	30 sec.	2
30 to 100	1 min.	3
100 to 300	2 min.	4
300 to 1000	5 min.	5
1000 and greater	10 min.	6

This is accomplished with a series of loops (Instruction 87), where the delay and count parameters are used to implement the frequency of measurement (and output) and the duration of that frequency. The unit of delay is the execution interval. A delay of 1 with a 10 second execution interval and a count of 60 means the instructions in the loop, in this case measure and output water level, are executed every 10 seconds for 10 minutes.

The drawdown portion of the test is completed at some time greater than 1000 minutes, at which time the operator sets Flag 1 low. At the next 10 minute pass through loop 6, the loop is exited and program execution returns to the top of the program table. To enter the recharge phase of the test, the operator enters the *6AD Mode and sets Flag 1 high and the measurement schedule starts over again.

The sensor is a 50 PSI Druck, model 930/ti with a calibration of 49.93 mV/10V of excitation or 4.993 mV/V. Your calibration will be different. An excitation voltage of 1500 mV yields a maximum signal of 7.489 mV at 50 PSI, fully utilizing the 7.5 mV Input Range to provide the best resolution.

The multiplier, m, is calculated to provide depth of water in feet:

m = (50 psi/4.993 mV/V) * (2.3067 ft/psi)

m = 23.099 ft/mV/V

The offset is calculated to provide a final value that represents the distance from the lip of the well to the water surface. Similar to Figure 7.16-2, the offset equals the initial distance of 47.23 feet plus the initial reading of 54.77, or 102 feet.

PROGRAM

```
*     Table 1 Program
  01:     10          Execution Interval (seconds)

;User must toggle Flag 1 to start measurements.
;
01:   If Flag/Port (P91)
        1:    21        Do if Flag 1 is Low
        2:     0        Go to end of Program Table

;Loop 1, Output every 10 seconds for 10 minutes.
;
02:   Beginning of Loop (P87)
        1:     1        Delay
        2:    60        Loop Count

03:   Do (P86)
        1:     1        Call Subroutine 1

04:   End (P95)
```

```
;Loop 2, Output every 30 seconds for 20 minutes.
;
05:   Beginning of Loop (P87)
        1:      3          Delay
        2:     40          Loop Count

06:   Do (P86)
        1:      1          Call Subroutine 1

07:   End (P95)

;Loop 3, Output every 1 minute for 70 minutes.
;
08:   Beginning of Loop (P87)
        1:      6          Delay
        2:     70          Loop Count

09:   Do (P86)
        1:      1          Call Subroutine 1

10:   End (P95)

;Loop 4, Output every 2 minutes for 200 minutes.
;
11:   Beginning of Loop (P87)
        1:     12          Delay
        2:    100          Loop Count

12:   Do (P86)
        1:      1          Call Subroutine 1

13:   End (P95)

;Loop 5, Output every 5 minutes for 700 minutes.
;
14:   Beginning of Loop (P87)
        1:     30          Delay
        2:    140          Loop Count

15:   Do (P86)
        1:      1          Call Subroutine 1

16:   End (P95)

;Loop 6, Output every 10 minutes until stopped
by user.
;
17:   Beginning of Loop (P87)
        1:     60          Delay
        2:      0          Loop Count

18:   Do (P86)
        1:      1          Call Subroutine 1

19:   If Flag/Port (P91)
        1:     21          Do if Flag 1 is Low
        2:     31          Exit Loop if True

20:   End (P95)

*     Table 3 Subroutines

01:   Beginning of Subroutine (P85)
        1:      1          Subroutine 1

02:   Full Bridge (P6)
        1:      1          Reps
        2:     22          ±7.5 mV 60 Hz Rejection Range
        3:      1          DIFF Channel
        4:      1          Excite all reps w/Exchan 1
        5:   1500          mV Excitation
        6:      1          Loc [ Level_Ft  ]
        7:     .46199      Mult
        8:    102          Offset

03:   Do (P86)
        1:     10          Set Output Flag High

04:   Real Time (P77)
        1:   0111          Day,Hour/Minute,Seconds

05:   Sample (P70)
        1:      1          Reps
        2:      1          Loc [ Level_Ft  ]

06:   End (P95)
```

INPUT LOCATIONS

1 Level_Ft

SECTION 9. INPUT/OUTPUT INSTRUCTIONS

TABLE 9-1. Input Voltage Ranges and Codes

Range Code					
Slow 2.72ms Integ.	Fast 250µs Integ.	60 Hz Reject	50 Hz Reject	Full Scale Range	Resolution*
1	11	21	31	±2.5mV	0.33 µV
2	12	22	32	±7.5mV	1. µV
3	13	23	33	±25 mV	3.33 µV
4	14	24	34	±250 mV	33.3 µV
5	15	25	35	±2500 mV	333. µV

* *Differential measurement; resolution for single-ended measurement is twice value shown.*

NOTE: When a voltage input exceeds the range programmed, the value which is stored is set to the maximum negative number and displayed as -99999 in high resolution or -6999 in low resolution.

*** 1 SINGLE-ENDED VOLTS ***

FUNCTION
This Instruction is used to measure voltage at a single-ended input with respect to ground. Output is in millivolts.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions*
02:	2	Range code (Table 9-1)
03:	2	Single-ended channel number for first measurement
04:	4	input location for first measurement
05:	FP	Multiplier
06:	FP	Offset

Input locations altered: 1

*Repetitions for the input/output instructions are limited to two differential or four single ended measurements.

*** 2 DIFFERENTIAL VOLTS ***

FUNCTION
This Instruction reads the voltage difference between the high and low inputs of a differential channel. Table 9-1 contains all valid voltage ranges and their codes. Both the high and low inputs must be within ±2.5 V of the CR510's ground (see Common Mode Range in Section 14.7.2). Pyranometer and thermopile sensors require a jumper between LO and GROUND to keep them in Common Mode Range. Output is in millivolts.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code (Table 9-1)
03:	2	Differential channel number for first measurement
04:	4	Input location for first measurement
05:	FP	Multiplier
06:	FP	Offset

Input locations altered: 1

*** 3 PULSE COUNT ***

There are three pulse input types which may be measured with the Pulse Count Instruction. The Pulse Count Instruction may also be used to measure switch closures (<40 Hz) with Control Port C2 see Section 8.5 for examples).

HIGH FREQUENCY PULSE

In this configuration, the minimum detectable pulse width is 2 microseconds, i.e. the maximum frequency is 250 kHz with a 50% duty cycle. The 8 bit counter has a maximum input frequency of 2000 Hz and the 16 bit option a maximum of 250 kHz. The count is incremented when input voltage changes from below 1.5

volts to above 3.5 volts. The maximum input voltage is +20 volts. A problem, however, arises when the pulse is actually a low frequency signal (below about 10 Hz) and the positive voltage excursion exceeds 5.6 VDC.

When this happens, the excess voltage is shunted to the CR510 5 VDC supply, with the current limited by an internal 10 Kohm resistor. When this extra current source exceeds the quiescent current needs of the CR510 (about 0.6 mA), the 5 VDC supply will start to rise, upsetting all analog measurements.

Thus, pulses whose positive voltage portion exceeds 5.6 VDC with a duration longer than 100 milliseconds need external conditioning. One method would be to use a 4 to 5.6 V zener diode from the signal to ground. The simplest method, however, is to add an external 20 Kohm resistor in series with the signal (Figure 9-1). This will limit the current for pulses to 20 VDC to the point that it will not upset the CR510 5 VDC supply.

LOW LEVEL AC

This configuration is used to count the frequency of AC signals from magnetic pulse flow transducers or other low voltage, sine wave inputs. The minimum input voltage is 6 millivolts RMS. Input hysteresis is 11 millivolts. The maximum AC input voltage is 20 volts RMS. The maximum input frequency ranges from 1000 Hz at 20 mV RMS to 16,000 Hz at 1000 mV or greater.

SWITCH CLOSURE (Pulse Inputs)

In this configuration, the minimum switch closed time is 5 milliseconds. The minimum switch open time is 6 milliseconds. The maximum bounce time is 1 millisecond open without being counted.

The switch is connected between the pulse input and ground (G). When the switch is open, the CR510 pulls the pulse input to 5 volts. When the switch is closed, the pulse input is at ground. The count is incremented when the switch opens.

SWITCH CLOSURE (Control Ports)

Control Ports C1, C2/P3 can be used to measure switch closures < 40 Hz. The minimum switch opened or closed time is 6 milliseconds.

The switch is connected between the control port and 5 V. When the switch is open, the control port is pulled to ground by an internal 100 Kohm resistor. When the switch is closed, the control port is at 5 V. The count is incremented when the switch closes.

> **NOTE:** If Control Port C2/P3 is used for pulse measurements or interrupt subroutines, the CR510 will not go into the quiescent power state (0.7 mA) if Control Port C2/P3 is high.

PULSE MEASUREMENT DETAILS

The 2 pulse count input channels each have eight bit counters. Input frequencies greater than 2000 Hz (the limit of the eight bit counter, 255 counts at the reset interval of 0.125 second) can be counted by combining two counters on one input channel. When this option is selected, channel 1 is used for the pulse input. Channel 2 is not used.

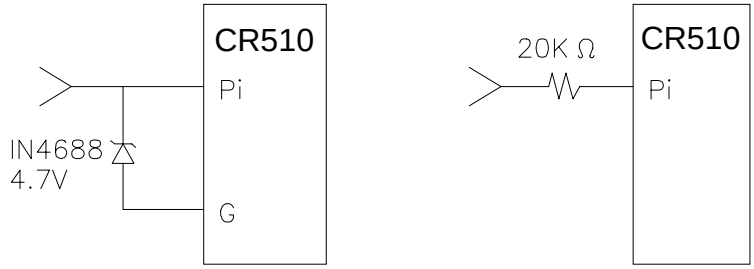


FIGURE 9-1. Conditioning for Long Duration Voltage Pulses

Every 0.125 seconds, the CR510 processor transfers the values from the 8 (or 16) bit pulse counters into 16 bit accumulators (max count is 65,535) and the counters are hardware reset to zero. The pulses accumulate in these 16 bit accumulators until the program table containing the Pulse Count Instruction is executed. At the beginning of the execution of the Table containing the Pulse Count Instruction, the total in the 16 bit accumulator is transferred to a temporary RAM buffer. The 16 bit accumulator is then zeroed. When the table execution reaches the Pulse Count Instruction, the value in the RAM buffer is multiplied by the multiplier and added to the offset and placed into the designated input location. A ramification of this is that the excitation interval of the table must be short enough that the 16 bit accumulator does not overflow.

CAUTION: The RAM buffer does NOT accumulate counts; it is zeroed each time the table is executed regardless of whether or not the pulse instruction is executed. If all counts are necessary, it is imperative that the Pulse Count Instruction be executed (not branched around) every time the table is executed.

If a table execution was skipped because the processor was executing the previous table (Section 1.1.1) or if the user resets the time, the value in the 16 bit accumulator is the result of a longer than normal interval. This value can either be used or it can be discarded. If pulse counts are being totalized, a missing count could be significant and the value from the erroneously long interval should NOT be discarded. If the pulse count is being processed in a way in which the resultant value is dependent upon the sampling interval (e.g., speed, RPM), the value from the excessive interval should be discarded. If the value is discarded the value in the RAM buffer from the previous measurement will be used.

There is also an option to output the count as a frequency (i.e., counts/execution interval in seconds = Hz) as well as discard the result from an excessive interval. This allows the use of a conversion factor that is independent of the execution interval.

The options of discarding counts from long intervals, pulse input type, and using a 16 bit counter are selected by the code entered for the 3rd parameter (Table 9-2).

NOTE: All pulse count instructions must be kept in the same table. If the Pulse Count Instruction is contained within a subroutine, that subroutine must be called from Table 2.

TABLE 9-2. Pulse Count Configuration Codes

Code	Configuration
0	High frequency pulse (Index (--) to change from 8 Hz to 64 Hz reset)
1	Low level AC (Index (--) to change from 8 Hz to 64 Hz reset)
2	Switch closure
3	High frequency pulse, sixteen bit counter
4	Low level AC, sixteen bit counter
1X	Long interval data discarded
2X	Long interval data discarded, frequency (Hz) output

where X is the configuration code

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Pulse channel or Control Port number for first measurement
03:	2	Configuration code (from above table)
04:	4	Input location for first measurement
05:	FP	Multiplier
06:	FP	Offset

Input locations altered: 1 per instruction

Intermediate storage locations altered: 1 per repetition

***** 4 EXCITE, DELAY, AND MEASURE *****

FUNCTION
This instruction is used to apply an excitation voltage, delay a specified time, and then make a single-ended voltage measurement. A 1 before the excitation channel number (1X) causes the channel to be incremented with each repetition.

The 50 and 60 Hz rejection ranges (Section 13.1) do not have enough time between integrations to allow a delay.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code (Table 9-1)
03:	2	Single-ended channel number for first measurement
04:	2	Excitation channel number
05:	4	Delay in hundredths of a second
06:	4	Excitation voltage (millivolts)
07:	4	Input location number for first measurement
08:	FP	Multiplier
09:	FP	Offset

Input locations altered: 1

*** 5 AC HALF BRIDGE ***

FUNCTION
This instruction is used to apply an excitation voltage to a half bridge (Figure 13.5-1), make a single-ended voltage measurement of the bridge output, reverse the excitation voltage, then repeat the measurement. The difference between the two measurements is used to calculate the resulting value which is the ratio of the measurement to the excitation voltage. A 1 before the excitation channel number (1X) causes the channel to be incremented with each repetition.

The excitation "on time" for each polarity is exactly the same to insure that ionic sensors do not polarize with repetitive measurements. The range should be selected to be a fast measurement (range 11-15), limiting the excitation on time to less than 800 microseconds at each polarity. A slow integration time should not be used with ionic sensors because of polarization error.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range Code (Table 9-1)
03:	2	Single-ended channel number
04:	2	Excitation channel number
05:	4	Excitation voltage (millivolts)
06:	4	Input location number for first measurement
07:	FP	Multiplier
08:	FP	Offset

Input locations altered: 1

*** 6 FULL BRIDGE WITH SINGLE *** DIFFERENTIAL MEASUREMENT

FUNCTION
This Instruction is used to apply an excitation voltage to a full bridge and make a differential voltage measurement of the bridge output. The measurement is made with the polarity of the excitation voltage both positive and negative (Figure 13.5-1). The result is 1000 times the ratio of the measurement to the excitation voltage. A 1 before the excitation channel number (1X) causes the channel to be incremented with each repetition.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code (Table 9-1)
03:	2	Differential channel number for first measurement
04:	2	Excitation channel number
05:	4	Excitation voltage (millivolts)
06:	4	Input location number for first measurement
07:	FP	Multiplier
08:	FP	Offset

Input locations altered: 1

*** 7 THREE WIRE HALF BRIDGE ***

FUNCTION
This Instruction is used to determine the ratio of the sensor resistance to a known resistance using a second voltage sensing wire from the sensor to compensate for lead wire resistance.

The measurement sequence is to apply an excitation voltage, make two voltage measurements on two adjacent single-ended channels, the first on the reference resistor and the second on the voltage sensing wire from the sensor (Figure 13.5-1), then reverse the excitation voltage and repeat the measurements. The two measurements are used to calculate the

resulting value, which is the ratio of the voltage across the sensor to the voltage across the reference resistor. A 1 before the excitation channel number (1X) causes the channel to be incremented with each repetition.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code for both measurements (Table 9-1)
03:	2	Single-ended channel number for first measurement
04:	2	Excitation channel
05:	4	Excitation voltage (millivolts)
06:	4	Input location number for first measurement
07:	FP	Multiplier
08:	FP	Offset

Input locations altered: 1

*** 8 DIFFERENTIAL VOLTAGE WITH EXCITATION AND DELAY ***

FUNCTION
This measurement consists of applying a single excitation voltage, delaying a specified time, and making a differential voltage measurement. The result stored is the voltage measured.

"Delay" (Parameter 5) refers to increasing the signal settling time by increasing the time between the start of excitation and the start of signal integration (Section 13.2). If a delay of 0 is specified, the inputs for the differential measurement are not switched for a second integration as is normally the case. With the 0 delay, Instruction 8 does not have as good resolution or common mode rejection as other differential measurements. It does provide a very rapid means of making bridge measurements. This instruction does not reverse excitation. A 1 before the excitation channel number (1X) causes the channel to be incremented with each repetition.

The 50 and 60 Hz rejection ranges (Section 13.1) do not have enough time between integrations to allow a delay.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code (Table 9-1)
03:	2	Differential channel number for first measurement
04:	2	Excitation channel number
05:	4	Delay (0.01s)
06:	4	Excitation voltage (millivolts)
07:	4	Input location number for first measurement
08:	FP	Multiplier
09:	FP	Offset

Input locations altered: 1

*** 9 FULL BRIDGE WITH EXCITATION COMPENSATION ***

FUNCTION
This instruction is used to apply an excitation voltage and make two differential voltage measurements. The measurements are made with both positive and negative excitation voltage. The measurements are made on sequential channels. The result is the voltage measured on the second channel (V_2) divided by the voltage measured on the first (V_1). If V_1 is measured on the 2.5 V range (code 5,15, 25 or 35 in Parameter 2), then the result is 1000 times V_2/V_1. A 1 before the excitation channel number (1X) causes the channel to be incremented with each repetition.

When used as a 6 wire full bridge (Figure 13.5-1), the connections are made so that V_1 is the measurement of the voltage drop across the full bridge, and V_2 is the measurement of the bridge output. Because the excitation voltage for a full bridge measurement is usually in the 2.5 V range, the output is usually 1000 V_2/V_1 or millivolts output per volt excitation.

When used to measure a 4 wire half bridge, the connections are made so that V_1 is the voltage drop across the fixed resistor (R_f), and V_2 is the drop across the sensor (R_s). As long as V_1 is not measured on the 2.5V range, the result is V_2/V_1 which equals R_s/R_f.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code for V_1 (Table 9-1)
03:	2	Range code for V_2
04:	2	Differential channel number for first measurement
05:	2	Excitation channel number
06:	4	Excitation voltage (millivolts)
07:	4	Input location number for first measurement
08:	FP	Multiplier
09:	FP	Offset

Input locations altered: 1

*** 10 BATTERY VOLTAGE ***

FUNCTION
This instruction reads the battery voltage and writes it to an input location. The units for battery voltage are volts. At approximately 9.6 V the CR510 suspends measurements.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location

Input locations altered: 1

*** 11 107 THERMISTOR PROBE ***

FUNCTION
This Instruction applies a 2 VAC excitation voltage to Campbell Scientific's Model 107 Thermistor Probe, makes a single-ended voltage measurement across a resistor in series with the thermistor, and calculates the temperature in °C with a polynomial. A multiplier of 1 and an offset of zero yields temperature in degrees Celsius. The maximum polynomial error from -40°C to +56°C is given here:

Curve Fit Error --

Range (°C)	Error (°C)
-40 to +56	<±1.0
-24 to +48	<±0.1

TABLE 9-3. Excitation/Integration Codes

Code	Result
0x	excite with channel x
1x	increment chan x with each rep
2x	excite with channel x, 60 Hz rejection, 10 ms delay
3x	excite with channel x, 50 Hz rejection, 10 ms delay
4x	increment chan x with each rep, 60 Hz rejection, 10 ms delay
5x	increment chan x with each rep, 50 Hz rejection, 10 ms delay

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Single-ended channel number of first measurement
03:	2	Excitation/Integration *Code (See Table)
04:	4	Input location for first measurement
05:	FP	Multiplier
06:	FP	Offset

Input locations altered: 1 per repetition

*** 12 207 RELATIVE HUMIDITY PROBE ***

FUNCTION
This instruction applies a 1.5 VAC excitation across Campbell Scientific's Model 207 Temperature and RH Probe, makes a fast single-ended measurement across a series resistor, calculates the result with a 5th order polynomial, and performs the required temperature compensation before outputting the result in % RH.

> **NOTE**: The temperature value used in compensating the RH value (Parameter 5) must be obtained (see Instruction 11) prior to executing Instruction 12 and must be in Celsius.

The RH results are placed sequentially into the input locations beginning with the first RH value.

In the 207 probe, the RH and temperature elements use a common excitation line.

CAUTION: Never excite the 207 probe with DC excitation because the RH chip will be damaged.

A 1 before the excitation channel number (1X) causes the channel to be incremented.

The maximum RH polynomial error is given here:

Curve Fit Error --

Range (%RH)	Error (%RH)
10 - 100	+4
15 - 94	+1

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	First single-ended channel for RH measurement
03:	2	Excitation channel number
04:	4	Input location for first compensating temperature measurement (°C)
05:	4	Input location for first R.H. measurement
06:	FP	Multiplier
07:	FP	Offset

Input locations altered: 1 per instruction

*** 16 TEMPERATURE FROM *** PLATINUM R.T.D.

FUNCTION
This instruction uses the result of a previous RTD bridge measurement to calculate the temperature according to the DIN 43760 specification adjusted (1980) to conform to the International Electrotechnical Commission standard. The range of linearization is -200° to 850°C. The error in the linearization is less than 0.001°C between -100° and +300°C, and is less than 0.003°C between -180° and +830°C. The error (T calculated - T standard) is +0.006° at -200°C and -0.006° at +850°C. The input must be the ratio R/Ro, where R is the RTD resistance and Ro the resistance of the RTD at 0°C (Sections 7.9 and 7.10). A multiplier of 1 and an offset of zero yields temperature in degrees Celsius.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	4	Input location of R/Ro
03:	4	Input location of result
04:	FP	Multiplier
05:	FP	Offset

Input locations altered: 1 per repetition

*** 17 INTERNAL TEMPERATURE ***

FUNCTION
This instruction measures the temperature (°C) of a thermistor on the CR510 analog board.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location number for temperature

Input locations altered: 1

*** 18 MOVE TIME TO INPUT LOCATION ***

FUNCTION
This instruction takes the current time in seconds into the minute, minutes into the day, or hours into the year and does a modulo divide (see Instruction 46) on the time value with the number specified in the second parameter. The result is stored in the specified input location. Entering 0 or a number which is greater than the maximum value of the time for the second parameter will result in the actual time value being stored.

PARAMETER 1 CODES

Code	Time Units
0	Seconds into minute (maximum 60)
1	Minutes into current day (maximum 1440)
2	Hours into current year (maximum 8784)
3	Store yr,day,hr,min,sec into 5 input locations (modulo divide not used)

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Time Code
02:	4	Number to modulo divide by
03:	4	Input location number

Input locations altered: 1 or 5

*** 19 MOVE SIGNATURE INTO INPUT LOCATION ***

FUNCTION
This instruction stores the signature of the Read Only Memory (ROM) and user program memory (SRAM) into an input location. The signature is a result of the CR510 PROM, the size of SRAM, and the entries in the ∗1, ∗2, ∗3, ∗4, ∗A, and ∗C Modes. This signature is not the same as the signatures given in the ∗B Mode. Recording the signature allows detection of any program change or ROM failure.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location number

Input locations altered: 1

*** 20 PORT SET ***

FUNCTION
This instruction sets or configures C1. On power-up, the port defaults to input configuration (i.e., is not driven high or low by the CR510, and can be used to read the status of an external signal using Instruction 25). When the port is set high, low, pulsed, or toggled by this instruction or a program control command, the port is automatically configured as an output.

> **NOTE**: Voltages in excess of 5.5 volts applied to a control port can cause the CR510 to malfunction.

C1 can also be set using the ∗6 Mode or the J and K telecommunications commands. However, C1 MUST be configured as output before these means of setting them will work. The option to configure the port as an output is used when it must be configured as an output without changing its state.

Pulse duration, initiated by a program control instruction, can be set for C1 (Table 12-2). Instruction 20 does not pulse the port, it only sets the duration. If Instruction 20 is not used to set the duration, the pulse command will result in a 10 ms pulse.

Instruction 20 has two 4 digit parameters. The code (0-9) entered as the digit determines what effect command 20 has on C1.

TABLE 9-4. Port Configuration Option Codes

Code	Function
0	Set port low
1	Set port high
2	Toggle port
3	Pulse duration 1 ms
4	Pulse duration 10 ms
5	Pulse duration 100 ms
6	Pulse duration 1 s
7	Configure as output
8	Configure as input
9	Leave unchanged

Duration of pulse on subsequent pulse port command in Program Control Instruction.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:		Not available
02:	2	C1 option codes

Input locations altered: 0

*** 21 PULSE PORT WITH DURATION ***

FUNCTION
Instruction 21 pulses control port 1 for a specified amount of time in hundredths of seconds (0.01 seconds).

The pulse works as a toggle; if the port is high before the instruction is executed, it will pulse low and vice versa. Any value less than 1, including 0, gives a pulse of 10 milliseconds. The maximum input value is limited to 65,000, which gives a pulse length of 650 sec.

Parameter 1 is 1 for C1. Parameter 2 is the input location containing the pulse length.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Control port
02:	4	Input location of pulse length in hundredths of a second

Input location altered: 0
Input locations read: 1

*** 22 EXCITATION WITH DELAY ***

FUNCTION
This instruction is used in conjunction with others for measuring a response to a timed excitation using the switched analog outputs. It sets the selected excitation output to a specific value, waits for the specified time, then turns off the excitation and waits an additional specified time before continuing execution of the following instruction. Analog power is turned off during delay after excitation to drop power to 3 mA.

If the excitation channel is indexed, parameter 4 becomes an input location. The excitation voltage must be loaded into the specified input location before Instruction 22 is executed.

If the only requirement is the delay of program execution, the excitation on time (parameter 2) can be set to zero and the off time delay (parameter 3) can be used.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Excitation channel number
02:	4	Delay time in hundredths of a second that excitation is on
03:	4	Delay time in hundredths of a second after excitation is turned off
04:	4	Excitation voltage in millivolts
	or	
		input location, when the excitation channel (Param. 1) is indexed

Input locations altered: 0
Input locations read: 0 or 1

*** 24 CALIBRATION ***

FUNCTION
Put the CR510's 19 calibration values into input locations. If C (--) is keyed before entering the input location, then the automatic calibrations are simply displayed, not measured. Otherwise, the calibration takes place only when Instruction 24 is executed; automatic calibration is disabled See Section 13.7 for details abut the CR510's calibration process.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location number, key C (--) for results of automatic calibration

Input locations altered: 19

*** 25 PORT READ ***

FUNCTION
The status of C1 is read and placed in an input location. The status is a base 2 representation of the port converted to base 10. For example, if port one is read and the port status is as follows:

PORT	C1
VALUE	1
STATUS (0=low, 1=high)	1

Base 10 equivalent: 1 = 1

1 will be stored in the input location.

The mask is also base 2 representation; 1 indicates the port is to be read, 0 results in a 0 for the port regardless of the status of the port (AND operation). For example, if 1 is entered, port 1 is read.

> **NOTE**: Voltages in excess of 5.5 volts applied to a control port can cause the CR510 to malfunction.

PARAM. NUM.	DATA TYPE	DESCRIPTION
01:	4	MASK (0-255)
02:	4	INPUT LOCATION TO STORE RESULT

Input locations altered: 1

*** 26 TIMER ***

FUNCTION
This instruction will reset a timer or store the elapsed time registered by the timer in seconds in an Input Storage location. Instruction 26 can be used with Program Control Instructions to measure the elapsed time between specific input conditions. There is only one timer and it is common to all tables (e.g., if the timer is reset in Table 1 and later

in Table 2, a subsequent instruction in Table 1 to read the timer will store the elapsed time since the timer was reset in Table 2).

Elapsed time is tracked in 0.125 second increments. The maximum interval that can be timed is 8191.875 seconds.

The timer is also reset in response to certain keyboard entries:

1. When tables are changed and compiled with the ∗0 Mode, the timer is reset automatically.
2. Entering "∗6" after changing the program compiles the programs, but does NOT reset the timer.

PARAM. NUM.	DATA TYPE	DESCRIPTION
01:	4	Input location no. of elapsed time in seconds (or enter 0 to reset)

Input locations altered: 1
(0 if timer is being reset)

*** 27 PERIOD MEASUREMENT ***

FUNCTION
Instruction 27 measures the period (microseconds) of a signal on a single ended input channel. As an option, the frequency of the signal in kHz may be output instead of the period.

The number of cycles to measure should be chosen so that at least one millisecond transpires while counting those cycles (e.g., if the maximum input frequency is 10 kHz, count more than 10 cycles). If the time for the number of cycles is less than 1 millisecond, an over range value (displayed as -99999.) will be stored for the measurement. The specified number of cycles are timed with a resolution of 35 nanoseconds, making the resolution on the period 35 nanoseconds divided by the number of cycles measured. Resolution is reduced by noise and signals with a slow transition through the zero voltage threshold.

The "Time out" parameter specifies the maximum length of time the instruction will wait on each repetition for the specified number of cycles. If the cycles have not been counted within this time, -99999 will be loaded into the input location.

TABLE 9-5. Input Frequency Codes

Range Code	Peak to Peak Volts Required @ Max. Freq.*	Maximum Frequency
1	2 mV	8 kHz
2	3 mV	20 kHz
3	12 mV	50 kHz
4	2 V	150 kHz**

0x Output period in microseconds
1x Output frequency in kHz

where x is range code

* AC voltage; must be centered around CR510 ground.

** Consult Campbell Scientific if higher frequencies are required.

PARAM. NUM.	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Gain/output option
03:	2	Single-ended input channel
04:	4	# Cycles to measure
05:	4	Time out (0.01 sec, at least the maximum duration of the number of cycles specified + 1 1/2 cycles.)
06:	4	Destination input location
07:	FP	Multiplier
08:	FP	Offset

Input locations altered: 1 per repetition

*** 28 VIBRATING WIRE MEASUREMENT ***

FUNCTION
Excites a vibrating wire sensor with a swept frequency (from low frequency to high), then measures the response period and calculates $1/T^2$, where T is the period in ms. Excitation is normally provided before each repetition. As an option, a single excitation can be made prior to all repetitions of the measurement. An AVW1 or AVW4 Vibrating Wire Interface is usually required for these sensors.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions Hit C (--) to skip repeat of excitation
02:	2	Single-ended channel for first measurement
03:	2	Excitation Channel
04:	2	Start frequency of sweep (100'S of Hz)
05:	2	End frequency of sweep (100'S of Hz)
06:	4	# Cycles to measure (0 means none)
07:	4	Delay before excitation applied (0.01 sec units)
08:	4	Input location ($1/T^2$), T in ms
09:		FP Multiplier
10:		FP Offset

Input locations altered: 1 per repetition

***** 29 INW PS9105E *****

FUNCTION
The Instrumentation Northwest PS9105 Enhanced Pressure Transducer is used to measure water level. This instruction excites the PS9105 with a single excitation channel and measures the sensor's output on two consecutive differential analog input channels. The pressure and sensor temperature are written into two consecutive input locations starting at the location specified in parameter 3.

Each PS9105 has 20 enhanced measurement parameters that are documented on the calibration sheet. Enter these parameters into Instruction 29.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Differential channel number for first measurement
02:	2	Excitation channel number
03:	4	Input location number
04:	FP	Pressure Range (psig) (enh. par. 1)
05:	FP	Enhanced Parameter 2
06:	FP	Enhanced Parameter 3
07:	FP	Enhanced Parameter 4
08:	FP	Enhanced Parameter 5
09:	FP	Enhanced Parameter 6
10:	FP	Enhanced Parameter 7
11:	FP	Enhanced Parameter 8
12:	FP	Enhanced Parameter 9
13:	FP	Enhanced Parameter 10
14:	FP	Enhanced Parameter 11
15:	FP	Enhanced Parameter 12
16:	FP	Enhanced Parameter 13
17:	FP	Enhanced Parameter 14
18:	FP	Enhanced Parameter 15
19:	FP	Enhanced Parameter 16
20:	FP	Enhanced Parameter 17
21:	FP	Enhanced Parameter 18
22:	FP	Enhanced Parameter 19
23:	FP	Enhanced Parameter 20

Input location altered: 2

***** 105 SDI-12 RECORDER *****

> **NOTE:** Version 1.2 of the SDI-12 specification has been implemented in the CR10X. Features added in this version are marked with a dagger (†) and may not be supported by older SDI-12 sensors.

Instruction 105 allows data to be collected from SDI-12 sensors. The sensor's SDI-12 data line is connected to a control port (C1 or C2). The SDI-12 ground should be connected to a "G" terminal on the CR10X wiring panel. SDI-12 power may be connected to 12V.

If multiple SDI-12 sensors are used, up to ten sensors may be connected to a single control port, but each sensor must have a unique address and requires a separate Instruction 105.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Address (0-9, †10-126)
02:	2	Command (Table 9-8)
03:	2	Control Port (C1-C8)
04:	4	Input Loc.
05:	FP	Mult
06:	FP	Offset

Input locations altered: 1-9 (†1-99), depending on the SDI-12 sensor

Intermediate locations required: 21

PARAMETER 1. ADDRESS

Enter the address of the SDI-12 sensor (0-9). †Extended addresses (A through Z and a through z) may be used by entering the decimal equivalent for the appropriate ASCII character (see Appendix C). For example, address 'A' would be entered as 65, and address 'z' would be entered as 122.

PARAMETER 2. COMMAND

Enter a number to select the command to be sent to the SDI-12 sensor. Usually 0 is entered to select the M command. The following Standard SDI-12 commands are supported by the CR510:

TABLE 9-6. SDI-12 Command Codes

ENTRY	COMMAND	DESCRIPTION
0	M	Initiate Measurement
0 --	C	†Initiate Concurrent Measurement
1..9	M1..M9	Additional Measurement commands specified by the SDI-12 sensor
10	V	Initiate Verify sequence
11	I	Send Identification

Command 0: the CR510 will issue the 'M' SDI-12 measurement command and wait for the sensor to complete its measurement before requesting the data and proceeding to the next instruction in the program table. If Instruction 105 is placed in Table 1, program execution will be suspended during this delay. If it is placed in Table 2, instructions in Table 1 may be executed during this delay.

†Command 0--: this command enables concurrent measurements with SDI-12 sensors that support this feature. With concurrent measurements, the CR510 can initiate measurements with multiple SDI-12 sensors without having to wait for each individual sensor to complete its sequence before proceeding to the next sensor. The CR510 will issue the 'C' SDI-12 concurrent measurement command and wait for the sensor response, which includes the amount of time in seconds it will take for the sensor to make the measurement. The CR510 will not wait for the data: rather, it will continue executing the table. The next time the instruction is executed, the CR510 will check the elapsed time. If the elapsed time is equal to or greater than that given by the sensor, the CR510 will get the data from the SDI-12 sensor. In the following execution of the instruction, the CR510 will again issue the 'C' command.

CAUTION: If you are using C2/P3 as a pulse input or interrupt subroutine, you must use the concurrent measurement option with the SDI-12 sensors on C1.

The results of an M, C, M1-M9, or V command sequence is numerical data, stored in input location(s). The response to the I command is text information, which is written directly to Final Storage regardless of the Output Flag's state. The ∗7 mode of the CR510 cannot be used to view text data.

†In addition to the Standard SDI-12 commands, the CR510 can issue 'Extended' SDI-12 commands. Instruction 68, Extended Parameters 4 Digit, is used to supply the characters and values to be transmitted. Multiple Instructions 68 can be used sequentially to extend the command string. The Command parameter 2 in Instruction 105 must be 0, not 0--. Parameter 4 in Instruction 105 should refer to the first Input Location, if any, to be sent as part of the command string. The parameters placed in P68 are the decimal ASCII equivalent of literal characters to be sent, or '128' if the value in an input location is to be sent. Enter a parameter of zero to end the string. The address prefix and '!' suffix are automatically sent in addition to the information listed in Instruction(s) 68. The CR510 keeps no data from the sensor response.

Example: To send the command 1A0+2.3456-87.654! where 1 is the SDI-12 sensor address, +2.3456 is the value in Input Location 5 and -87.654 is the value in Input Location 6, use the following instructions:

```
01:  SDI-12 Recorder (P105)
        1:      1          SDI-12 Address
        2:      0          Start Measurement (aM0!)
        3:      1          Port
        4:      5          Loc [ SendVal_1 ]
        5:      1          Mult
        6:      0          Offset

02:  Extended Parameters 4 Digit (P68)
        1:     65          Option  ;ASCII character A
        2:     48          Option  ;ASCII character 0
        3:    128          Option  ;Send first value
        4:    128          Option  ;Send second value
        5:      0          Option  ;End of command string
        6:      0          Option
        7:      0          Option
        8:      0          Option
```

PARAMETER 3. PORT

Enter the CR510 control port (C1, C2) connected to the SDI-12 sensor data line.

PARAMETER 4. INPUT LOCATION

Input location where the returned data is stored. If multiple values are returned from the SDI-12 sensor they are stored in sequential input locations beginning at the specified location.

ERRORS

If the CR510 receives either an incorrect response or no response from an SDI-12 sensor, the CR510 will retry the operation. If after retries a valid response has not been received, the CR510 will store a -99999 in the input location specified in Parameter 4. Only the first location will be altered. Sequential locations will contain values from previous measurements.

TRANSPARENT MODE

The SDI-12 transparent mode is used to communicate directly with an SDI-12 sensor. A common application of the transparent mode is to verify proper operation of the SDI-12 sensor.

A computer or terminal is required to use the transparent mode; the CR10KD (keyboard display) cannot be used. Transparent mode is entered while the computer is in telecommunications with the SDI-12 recorder CR510 (at the asterisk '*' prompt). Enter 'pX' at the asterisk prompt, where 'p' is the Control Port number (1, 2) attached to the SDI-12 data line. The CR510 responds with 'entering SDI-12.' Any SDI-12 command preceded with the sensor address and followed with an exclamation point '!' may then be entered. For example, entering '0I!' would request identification from an SDI-12 sensor addressed at 0.

The SDI-12 prompt will not appear until the CR510 finishes executing all program tables. While in transparent mode, scheduled tables in the CR510 will not execute. Transparent mode ends and the '*' prompt is returned when an invalid SDI-12 command (e.g., a blank line) is entered, if the SDI-12 sensor doesn't respond within the time-out period following a valid command (approximately 1/3 second), or if the user does not enter a command before the mode times out (approximately 35 seconds). Security must be unlocked to level 2 before the Transparent mode is enabled.

*** 106 SDI-12 SENSOR ***

Instruction 106 allows a CR510 to be used as an SDI-12 sensor. The CR510 can make measurements and transfer data using SDI-12 commands in response to another SDI-12 recorder.

Instruction 106 supports the Standard SDI-12 commands as listed in the Parameter 2 description for Instruction 105.

The SDI-12 data line is attached to Control Port 2 and Instruction 106 must be the first instruction in Subroutine 98 located in Table 3. An SDI-12 recorder addresses the SDI-12 sensor CR510 by sending a Break and the sensor's address. The sensor CR510 will call subroutine 98 whenever it detects activity on the

SDI-12 data line attached to Port 2, but if the Break and the specified address are not received by Instruction 106, the remainder of the subroutine is not executed.

Two programming techniques exist for obtaining measurement values to be transferred by the sensor Instruction 106. The first technique makes the requested measurements "on demand" in response to the recorders request. The measurement instructions are located in Subroutine 98 and are executed only when the SDI-12 recorder requests measurements. This technique is preferred when measurements are to be made at the recorder's command.

The second technique transfers measurement values previously obtained by instructions in Table 1 or Table 2. Subroutine 98 contains only Instructions 106 (SDI-12 sensor) and 95 (End). When the recorder requests measurements, values already in the specified input locations are used. The advantage of this technique is that the sensor CR510 can be making and storing measurements independent of the SDI-12 recorder. The data is also returned slightly faster since the sensor CR510 does not make measurements when the recorder requests data, but rather uses measurements made at the last regular table execution.

These two techniques can be combined allowing the sensor CR510 to function as an SDI-12 sensor and to make independent measurements. While Subroutine 98 is being executed, normal Table 1 or 2 execution scheduling may be altered or missed since Subroutine 98 is not interrupted. This is likely to occur if Subroutine 98 execution takes longer than the scan interval programmed for Table 1 or 2. It is also possible for instructions in Table 1 or 2 to prevent Subroutine 98 from being called in time for Instruction 106 to receive the address information from the recorder. This is likely to occur only if Table 1 or 2 is executed often and has instructions that take longer than 1/3 second to execute. For example, Instruction 4 (Excite-Delay-SE) with a 1/2 second delay could cause Subroutine 98 to miss the SDI-12 address information if it were executing when the SDI-12 data line became active. If this occurs the sensor CR510 will not respond to the SDI-12 recorder. Most instructions execute fast enough that when Instruction 106 misses the initial SDI-12 address, a subsequent retry by the recorder will work.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	ADDRESS (0-9, †10-126)
02:	4	TIME/VALUES ttnn: tt=time(sec) nn=no. values
03:	4	LOCATION starting loc. for values

Input locations altered: 0

Intermediate locations required: 182. To accommodate this number, increase the value in *A window 2, if keying program steps in by hand.

PARAMETER 1. ADDRESS

Enter the address for the CR510 acting as an SDI-12 sensor (0-9, †10-126 decimal value for ASCII character, see Appendix E). Each SDI-12 sensor connected to a control port must have a unique address.

PARAMETER 2. TIME/VALUES

Enter the time in seconds required for the sensor CR510 to complete subroutine 98 followed by the number of input locations to be returned to the SDI-12 recorder. The format is *ttnn* where *tt* specifies the time in seconds and *nn* (maximum 63) is the number of values from Input Locations to be sent.

Enter a time of 0 to transfer the values already stored in input locations. With a time of zero, the remaining instructions in Subroutine 98 are not executed.

The actual time to complete subroutine 98 is the time required to execute all instructions from Instruction 106 (SDI-12 sensor) to the final Instruction 95 (End).

For response to the ‘M’ command, the entered time may be longer than the actual time without slowing the data exchange because the sensor CR510 signals the SDI-12 data recorder when the data is ready for transfer.

For response to either the ‘M’ or the ‘C’ command, if the entered time is too short the sensor CR510 will not respond and the data values will not be transferred. Similarly, no response occurs if the SDI-12 recorder queries the sensor CR510 before the entered time has elapsed.

PARAMETER 3. LOCATION

This parameter determines the starting input location for the 'nn' values to be returned to the recorder. The 'M' or 'M1-M9' command issued by the SDI-12 recorder determines if the starting location is actually that specified in Parameter 3 or a multiple of 'nn' past Parameter 3.

Starting input location = Parameter 3 + (nn∗x), where nn is specified in Parameter 2, and, x is the number following the 'M' sent by the SDI-12 recorder (1-9).If the 'M' command is sent by the recorder x = 0.

Results of Instruction 106

The sensor CR510 will return a set of input locations in response to the M or M1..M9 command sequence. The set of Locations returned is determined by Parameters 2 and 3 of Instruction 106.

The three values, sent in response to a V command sequence, indicate the status of the sensor CR510. The first and second values are from the ∗B mode of the sensor CR510, giving the number of watchdog errors (E08) and the number of table overruns that have occurred. The third is a signature of the sensor CR510 memory. This signature is created by the same technique that the Instruction 19 (Signature) uses.

In response to an I command, the CR510 sends the string 'AVVCAMPBELLCR510 OS' where A is the sensor address, VV is the SDI-12 version number, OS is the CR510 operating system number.

***** 114 SET TIME *****

FUNCTION
Instruction 114 can be used to set the CR510 clock from values in input locations.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Option code: 0 set time with hr,min,sec with values in 3 input locations. 1 set time with day,hr,min,sec using 4 input locations. 2 set time with yr,day,hr,min,sec using 5 input locations.
02:	4	Input location number

***** 117 READ DATALOGGER ID *****

FUNCTION
Instruction 117 stores the datalogger ID into an input location. The datalogger ID is set in the ∗D mode with Command 8 (Section 1.8.4).

PARAM NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location number

***** 130 Error Monitor *****

FUNCTION
Stores an error count or status information from *B into an input location. Indexing the option code will cause the instruction to store the value into an input location and then clear the *B value. This is designed to assist with troubleshooting and to alert users of possible problems.

PARAM NUMBER	DATA TYPE	DESCRIPTION
01:	2	Option Code (Index (--) to reset error count after reading)
02:	4	Input Location

Option Codes:

0	Read Watchdog or E08s Errors
1	Read Table Overruns
2	Read Low 12V Detection
3	Read Lithium Battery Voltage Level (Indexing does nothing)
4	Read Flash (for CR510-1M or CR510-2M) Errors

Input Locations Altered: 1

*** 131 Enhanced Vibrating Wire Measurement ***

FUNCTION
Excites a vibrating wire sensor with a swept frequency (from low frequency to high), then measures the response period and calculates $1/T^2$, where T is the period in ms. Excitation is normally provided for each repetition. As an option, a single excitation can be made prior to all repetitions of the measurement. An AVW1 Vibrating Wire Interface is required for these sensors, but it may have trouble reading the low frequencies.

T - Amount of Time to sweep between the specified frequencies

N - Number of steps to use when sweeping from the Starting Frequency to the Ending Frequency

The $1/T^2$ value stored in the first Input Location specified in Parameter 11 is the result after the # of Cycles specified in Parameter 9. The $1/T^2$ value stored in the sequential Input Location is the result after 5 times the # of Cycles specified in Parameter 9. The value in the second Input Location is the correct value to use unless the reading falls into a certain range. This instruction is most often used with the Slope Indicator Vibrating wire. Slope Indicator has datalogger programming instructions they recommend using to decide which of the 2 readings to use.

PARAM NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions (Index (- -) to skip repeat of excitation)
02:	2	Single-ended channel for first measurement
03:	2	EX CHAN
04:	4	Starting Frequency (HZ)
05:	4	Ending Frequency (HZ)
06:	4	T (sweep, Units = 1 msec)
07:	4	N (Number of steps)
08:	4	Delay after excitation before measurement (Units = 1 msec)
09:	4	CYCLES to measure
10:	4	DELAY between reps (Units = 0.01 sec)
11:	4	First Input Location
12:	FP	Multiplier
13:	FP	Offset

Input Locations Altered: 2 per repetition

SECTION 10. PROCESSING INSTRUCTIONS

To facilitate cross referencing, parameter descriptions are keyed [ ] to the values given on the PROMPT SHEET. These values are defined as follows:

[Z] = Destination input location for result
[X] = Input location of X
[Y] = Input location of Y
[F] = Fixed Data (user specified floating point number)

***** 30 Z = F x 10^n *****

FUNCTION
Store a fixed value into an input location. The value is entered in scientific notation; the absolute value of the number may range from 1 x 10^{-19} to 9 x 10^{18}. A value smaller than the minimum is set to 0, while a larger value is set to the maximum.

PARAM. NUMBER	DATA TYPE	DESCRIPTION	
01:	FP	Mantissa	[F]
02:	2	Exponent of 10 (Press C to change sign)	
03:	4	Destination for input location	[Z]

Input locations altered: 1

***** 31 Z = X *****

FUNCTION
Copy data from one input location to another.

PARAM. NUMBER	DATA TYPE	DESCRIPTION	
01:	4	Source input location number	[X]
02:	4	Destination input location	[Z]

Input locations altered: 1

***** 32 Z = Z + 1 *****

FUNCTION
Add 1 to the current value in the specified input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION	
01:	4	Destination input location	[Z]

Input locations altered: 1

***** 33 X + Y *****

FUNCTION
Add X to Y and place result in a third input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Input location of Y	[Y]
03:	4	Dest. input location of X+Y	[Z]

Input locations altered: 1

***** 34 X + F *****

FUNCTION
Add F to X (where F is a fixed floating point number) and place the result in an input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	FP	Fixed value	[F]
03:	4	Dest. input location of X+F	[Z]

Input locations altered: 1

***** 35 X - Y *****

FUNCTION
Subtract Y from X and place the result in an input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Input location of Y	[Y]
03:	4	Dest. input location for X-Y	[Z]

Input locations altered: 1

*** 36 X * Y ***

FUNCTION
Multiply X by Y and place the result in an input location (Z).

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Input location of Y [Y]
03:	4	Dest. input location for X*Y [Z]

Input locations altered: 1

*** 37 X * F ***

FUNCTION
Multiply X by F (where F is a fixed multiplier) and place the result in an input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	FP	Fixed value [F]
03:	4	Dest. input location for X*F [Z]

Input locations altered: 1

*** 38 X / Y ***

FUNCTION
Divide X by Y and place the result in an input location. Division by 0 will cause the result to be set to the maximum CR510 number (99999).

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Input location of Y [Y]
03:	4	Dest. input location for X/Y [Z]

Input locations altered: 1

*** 39 SQUARE ROOT ***

FUNCTION
Take the square root of X and place the result in an input location. If X is negative, 0 will be stored as the result.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Dest. input location for $X^{1/2}$ [Z]

Input locations altered: 1

*** 40 LN(X) ***

FUNCTION
Take the natural logarithm of X and place the result in an input location. If X is 0 or negative, -99999 will be stored as the result.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Dest. input location for LN(X) [Z]

Input locations altered: 1

*** 41 EXP(X) ***

FUNCTION
Raise the exponential (EXP) base e to the X power and place it in an input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Dest. input for EXP(X) [Z]

Input locations altered: 1

*** 42 1/X ***

FUNCTION
Take the inverse of X and place the result in an input location. If X=0, 99999 will be given as the result.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Dest. input location for 1/X [Z]

Input locations altered: 1

*** 43 ABS(X) ***

FUNCTION
Take the absolute (ABS) value of X and place the result in an input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Dest. input location for ABS(X) [Z]

Input locations altered: 1

*** 44 FRACTIONAL VALUE OF X ***

FUNCTION
Take the fractional (FRAC) value (i.e., the non-integer portion) of X and place the result in an input location, e.g, FRAC (1.5) = 0.5.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Dest. input location for FRAC(X) [Z]

Input locations altered: 1

*** 45 INTEGER VALUE OF X ***

FUNCTION
Take the integer (INT) value of X and place the result in an input location, e.g., FRAC (1.5) = 1.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Dest. input location for INT(X) [Z]

Input locations altered: 1

*** 46 X MOD F ***

FUNCTION
Do a modulo divide of X by F and place the result in an input location. X MOD F is defined as the REMAINDER obtained when X is divided by F (e.g., 3 MOD 2 = 1). X MOD 0 returns X.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	FP	Fixed divisor [F]
03:	4	Dest. input loc. For X MOD F [Z]

Input locations altered: 1

*** 47 X^Y ***

FUNCTION
Raise X to the Y power and place the result in an input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Input location of Y [Y]
03:	4	Dest. input location for X^Y [Z]

Input locations altered: 1

*** 48 SIN(X) ***

FUNCTION
Calculate the sine of X (X is assumed to be in degrees) and place the result in an input location. The cosine of a number can be obtained by adding 90 to the number and taking the sine (COSX = SIN (X + 90)).

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Dest. input location for SIN(X) [Z]

Input locations altered: 1

*** 49 SPATIAL MAXIMUM ***

FUNCTION
Find the spatial maximum (SPA MAX) value of the given set or SWATH of input locations and place the result in an input location. To find the input location where the maximum value occurs, add 1000 to the input location number destination selected [Z] and enter this modified location number as Parameter 03. The input location ID of the maximum value observed will then be stored in destination [Z] plus 1.

Parameter 3 cannot be entered as an indexed location within a loop (Instruction 87). To use Instruction 49 within a loop, enter Parameter 3 as a fixed location and follow 49 with the Instruction 31 (Move Data). In Instruction 31, enter the location in which 49 stores its result as the source (fixed) and enter the destination as an indexed location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Swath [SWATH]
02:	4	Starting input location [1ST LOC]
03:	4	Dest. input location for maximum [MAX or Z]

Input locations altered: 1 or 2

***** 50 SPATIAL MINIMUM *****

FUNCTION
Find the spatial minimum (SPA MIN) value of the given set or SWATH of input locations and place the result in an input location. To find the input location where the minimum value occurs, follow the instructions given above for SPATIAL MAXIMUM.

Parameter 3 cannot be entered as an indexed location in a loop. Within a loop, Instruction 50 must be used in conjunction with Instruction 31 as described for Instruction 49.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Swath [SWATH]
02:	4	Starting input location [1ST LOC]
03:	4	Dest. input location for minimum [MIN or Z]

Input locations altered: 1 or 2

***** 51 SPATIAL AVERAGE *****

FUNCTION
Take the spatial average (SPA AVG) over the given set or SWATH of input locations and place the result in an input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Swath [SWATH]
02:	4	Starting input location [1ST LOC]
03:	4	Dest. input location of average [AVG or Z]

Input locations altered: 1

***** 52 RUNNING AVERAGE *****

FUNCTION
This instruction calculates the running average of a value in an input location. The most recent n values (where n is the number specified in parameter 4) are kept in intermediate storage. When Instruction 52 is executed, the current value is written over the oldest value and the average of the values is calculated and stored in the destination location. Out of range values (displayed as -99999.) are not included in the average.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Reps (REPS)
02:	4	Input location of source data
03:	4	Destination input location
04:	4	Number of values in running average window (NUMAVG)

Input locations altered: 1 per repetition
Intermediate locations required: Reps *(2+n)

***** 53 SCALING ARRAY WITH MULTIPLIER AND OFFSET *****

FUNCTION
Take 4 input location values, multiply each by a floating point constant, then add another floating point constant to the resulting products and place the final results back into each of the original 4 input locations.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	First input location [STRT LOC]
02:	FP	Multiplier 1 [A1]
03:	FP	Offset 1 [B1]
04:	FP	Multiplier 2 [A2]

05:	FP	Offset 2	[B2]
06:	FP	Multiplier 3	[A3]
07:	FP	Offset 3	[B3]
08:	FP	Multiplier 4	[A4]
09:	FP	Offset 4	[B4]

Input locations altered: 4

***** 54 BLOCK MOVE *****

FUNCTION
Executes a "block move" of data in input locations. Parameters specify the number of values to move, the source, source step, destination, and destination step. The "step" parameters designate the increment of the source and destination input locations for each value that is moved. For example, a "source step" of 2 and a "destination step" of 1 will move data from every other input location to a contiguous block of input locations.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Number of values to move
02:	4	1st source location
03:	2	Step of source
04:	4	1st destination location
05:	2	Step of destination

Input locations altered: Number of values of move

***** 55 5TH ORDER POLYNOMIAL *****

FUNCTION
Evaluate a 5th order polynomial of the form

$$F(X)=C0+C1X+C2X^2+C3X^3+C4X^4+C5X^5$$

where C0 through C5 are the coefficients for the argument X raised to the zero through fifth power, respectively. The magnitude of the user entered coefficient is limited to a range of ±.00001 to ±99999. Polynomials with coefficients outside this range can be modified by pre-scaling the X value by an appropriate factor to place the coefficients within the entry range. Pre-scaling can also be used to modify coefficients which are very close to 0 to increase the number of significant digits.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions [REPS]
02:	4	Starting input location for X [X]
03:	4	Dest. input location for F(X) [F(X) or Z]
04:	FP	C0 coefficient [C0]
05:	FP	C1 coefficient [C1]
06:	FP	C2 coefficient [C2]
07:	FP	C3 coefficient [C3]
08:	FP	C4 coefficient [C4]
09:	FP	C5 coefficient [C5]

Input locations altered: 1 per repetition

***** 56 SATURATION VAPOR PRESSURE *****

FUNCTION
Calculate saturation vapor pressure (over water SVPW) in kilopascals from the air temperature (°C) and place it in an input location. The algorithm for obtaining SVPW from air temperature (°C) is taken from: Lowe, Paul R.: 1977, “An approximating polynomial for computation of saturation vapor pressure,” *J. Appl. Meteor,* **16**, 100-103.

Saturation vapor pressure over ice (SVPI) in kilopascals for a 0°C to -50°C range can be obtained using Instruction 55 and the relationship

$$SVPI = -.00486 + .85471\ X + .2441\ X^2$$

where X is the SVPW derived by Instruction 56. This relationship was derived by Campbell Scientific from the equations for the SVPW and the SVPI given in Lowe's paper.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of air temperature °C [TEMP.]
02:	4	Destination input location for saturated vapor pressure [VP or Z]

Input locations altered: 1

*** 58 LOW PASS FILTER ***

FUNCTION
Apply a numerical approximation to an analog resistor capacitor (RC) low pass (LP) filter using the following algorithm.

$$F(Xi) = W*Xi + F(Xi-1) * (1-W)$$

Where X = input sample, W = user entered weighting function (O< W <1). If W=O, $F(Xi)=X_1$; if W=1, F(Xi)=X and F(Xi-1) = output calculated for previous sample

The equivalent RC time constant is given by T/W, where T is the sampling time in seconds. For values of W less than 0.25, the analogous "cut off" frequency (the frequency where the ratio of output to input is .707) is accurately represented by $W/(2\pi T)$. For larger values of W, this "analog" estimate of the cutoff frequency becomes less representative.

On the first execution after compiling, F(X) is set equal to X.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions [REPS]
02:	4	First input location for input data [X]
03:	4	Dest. input location for first filtered result[F(X) or Z]
04:	FP	Weighting function, W [W]

Input locations altered: 1 per repetition
Intermediate locations required: 1 per repetition

*** 59 BRIDGE TRANSFORM ***

FUNCTION
This instruction is used to aid in the conversion of a ratiometric Bridge measurement by obtaining the value for R_s which is equivalent to $R_f[X/(1-X)]$, where X is the value derived by the standard CR510 Bridge Measurement Instructions (with appropriate multiplier and offset, Section 13.5) and R_f represents the MULTIPLIER value. The result of Instruction 59 is stored in the same location that X was.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions [REPS]
02:	4	Starting input location & result destination [X]
03:	FP	Multiplier (Rf) [MULT]

Input locations altered: 1 per repetition

*** 61 INDIRECT INDEXED MOVE ***

FUNCTION
Moves input data from location X to location Y, where X and/or Y are indirectly addressed (X and Y are stored in the locations specified by Parameters 1 and 2). If a location parameter is specified as “indexed” (xxxx--), then the actual input location referenced is calculated by adding the current index counter to the value in the specified input location. When used outside a loop, the addressing is simply indirect because the index counter is zero.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location containing source location X
02:	4	Input location containing destination location Y

Input locations altered: 1

*** 63 PARAMETER EXTENSION ***

Instruction 63 is used immediately following Instructions 97 or 98 to allow the entry of a variable number of parameters. Instruction 63 can be entered several times in sequence if the number of parameters requires it. There are 8 two digit parameters. Refer to instruction being extended (97, 98) for specifics on the use of Instruction 63.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:-08:	2	Depends upon preceding instruction. Following Instruction 97 RF IDs and Phone No. 1 digit at a time, 32 Between RF IDs, 70 after the last RF ID, 32 and 84 Between RF and DC112 Phone,

	and 13 To END. Following Instruction 98 (255 character limit) Base 10 value of ASCII character (Appendix E) 00 TO END.

Input locations altered: 0

*** 65 BULK LOAD ***

FUNCTION
Instruction 65 inputs given values in up to eight Input Storage locations.

The Bulk Load instruction has 9 parameters. The first eight are the values to be entered in input storage locations. The ninth is the input location for the first data value; subsequent data values are placed in sequential input locations.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	FP	Param. to be entered
02:	FP	"
03:	FP	"
04:	FP	"
05:	FP	"
06:	FP	"
07:	FP	"
08:	FP	"
09:	4	Starting input location

Input locations altered: 8

*** 66 ARCTAN ***

FUNCTION
Calculate the angle in degrees whose tangent is X/Y. The polarity of X and Y must be known to determine the quadrant of the angle, as shown here. If 0 is entered for Parameter 2, the Arctangent of X is the result (limits of ARCTAN(X) are -90° < ARCTAN < 90°).

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location of X [X]
02:	4	Input location of Y [Y]
03:	4	Destination input location for ARCTAN(X/Y)

Input locations altered: 1

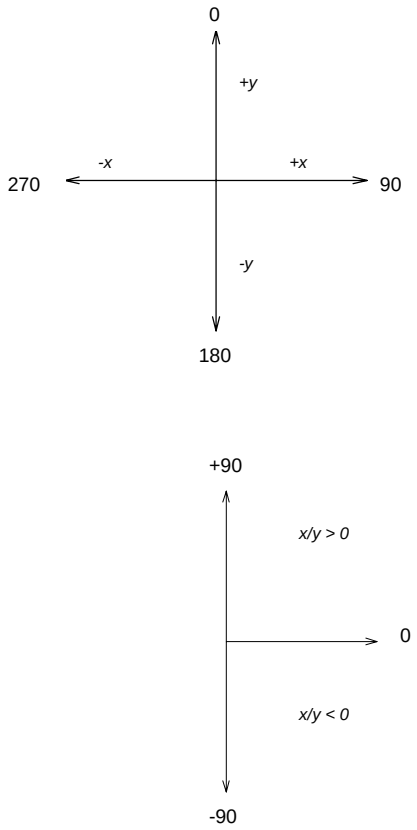


FIGURE 10-1. Quadrant that the Angle Falls in is Defined by the Sign of x and y.

*** 68 EXTENDED PARAMETERS 4 DIGIT ***

FUNCTION
This instruction is used to give other instructions additional parameters. Each of the eight parameters in Instruction 68 is defined by the instruction it follows. Refer to the specific instruction that uses extended parameters.

Input location altered: 0

SECTION 11. OUTPUT PROCESSING INSTRUCTIONS

*** 69 WIND VECTOR ***

FUNCTION
Instruction 69 processes the primary variables of wind speed and direction from either polar (wind speed and direction) or orthogonal (fixed East and North propellers) sensors. It uses the raw data to generate the mean wind speed, the mean wind vector magnitude, and the mean wind vector direction over an output interval. Two different calculations of wind vector direction (and standard deviation of wind vector direction) are available, one of which is weighted for wind speed.

When used with polar sensors, the instruction does a modulo divide by 360 on wind direction, which allows the wind direction (in degrees) to be 0 to 360, 0 to 540, less than 0, or greater than 540. The ability to handle a negative reading is useful in an example where a difficult to reach wind vane is improperly oriented and outputs 0 degrees at a true reading of 340 degrees. The simplest solution is to enter an offset of -20 in the instruction measuring the wind vane, which results in 0 to 360 degrees following the modulo divide.

When a wind speed sample is 0, the instruction uses 0 to process scalar or resultant vector wind speed and standard deviation, but the sample is not used in the computation of wind direction. The user may not want a sample less than the sensor threshold used in the standard deviation. If this is the case instruction 89 can be used to check wind speed, and if less than the threshold, Instruction 30 can set the input location equal to 0.

Standard deviation can be processed one of two ways: 1) using every sample taken during the output period (enter 0 for parameter 2), or, 2) by averaging standard deviations processed from shorter sub-intervals of the output period. Averaging sub-interval standard deviations minimizes the effects of meander under light wind conditions, and it provides more complete information for periods of transition[1].

[1] EPA On-site Meteorological Program Guidance for Regulatory Modeling Applications.

Standard deviation of horizontal wind fluctuations from sub-intervals is calculated as follows:

$$\sigma(\Theta)=[((\sigma\Theta_1)^2+(\sigma\Theta_2)^2 \ldots+(\sigma\Theta_M)^2)/M]^{1/2}$$

where $\sigma(\Theta)$ is the standard deviation over the output interval, and $\sigma\Theta_1 \ldots \sigma\Theta_M$ are sub-interval standard deviations.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	4	Samples per sub-interval (number of scans, enter 0 for no sub-interval)
03:	2	Sensor/Output 2 digits: AB A Sensor type: 0 = Speed and Direction 1 = East and North B Output option: 0 S, Θ1, σ(Θ1) 1 S, Θ1 2 S, U, Θu, σ(Θu)
04:	4	First wind speed input location no. (East wind speed)
05:	4	First wind direction input location no. (North wind speed)

Outputs Generated: 2-4 (depending on output option) per repetition

A sub-interval is specified as a number of scans. The number of scans for a sub-interval is given by:

Desired sub-interval (secs) / scan rate (secs)

In an example where the scan rate is 1 second and the Output Flag is set every 60 minutes, the standard deviation is calculated from all 3600 scans when the sub-interval is 0. With a sub-interval of 900 scans (15 minutes) the standard deviation is the average of the four sub-interval standard deviations. The last sub-interval is weighted if it does not contain the specified number of scans.

There are three Output Options that specify the values calculated.

Option 0:

Mean horizontal wind speed, S.
Unit vector mean wind direction, $\Theta 1$.
Standard deviation of wind direction, $\sigma(\Theta 1)$.

Standard deviation is calculated using the Yamartino algorithm. This option complies with EPA guidelines for use with straight-line Gaussian dispersion models to model plume transport.

Option 1:

Mean horizontal wind speed, S.
Unit vector mean wind direction, $\Theta 1$.

Option 2:

Mean horizontal wind speed, S.
Resultant mean wind speed, U.
Resultant mean wind direction, Θu.
Standard deviation of wind direction, $\sigma(\Theta u)$.

This standard deviation is calculated using Campbell Scientific's wind speed weighted algorithm.

Use of the Resultant mean horizontal wind direction is not recommended for straight-line Gaussian dispersion models, but may be used to model transport direction in a variable-trajectory model.

Measured raw data:

S_i = horizontal wind speed
Θ_i = horizontal wind direction
Ue_i = east-west component of wind
Un_i = north-south component of wind
N = number of samples

Calculations:

Scalar mean horizontal wind speed, S:

$$S=(\Sigma S_i)/N$$

where in the case of orthogonal sensors:

$$S_i=(Ue_i^2+Un_i^2)^{1/2}$$

Unit vector mean wind direction, $\Theta 1$:

$$\Theta 1=\text{Arctan } (Ux/Uy)$$

where

$$Ux=(\Sigma \sin \Theta_i)/N$$

$$Uy=(\Sigma \cos \Theta_i)/N$$

or, in the case of orthogonal sensors

$$Ux=(\Sigma(Ue_i/U_i))/N$$

$$Uy=(\Sigma(Un_i/U_i))/N$$

$$\text{where } U_i=(Ue_i^2+Un_i^2)^{1/2}$$

Standard deviation of wind direction, $\sigma(\Theta 1)$, using Yamartino algorithm:

$$\sigma(\Theta 1)=\text{arc sin}(\varepsilon)[1+0.1547\ \varepsilon^3]$$

where,

$$\varepsilon=[1-((Ux)^2+(Uy)^2)]^{1/2}$$

and Ux and Uy are as defined above.

Resultant mean horizontal wind speed, U:

$$U=(Ue^2+Un^2)^{1/2}$$

where for polar sensors:

$$Ue=(\Sigma S_i \text{ Sin } \Theta_i)/N$$

$$Un=(\Sigma S_i \text{ Cos } \Theta_i)/N$$

or, in the case of orthogonal sensors:

$$Ue=(\Sigma Ue_i)/N$$

$$Un=(\Sigma Un_i)/N$$

Resultant mean wind direction, Θu:

$$\Theta u=\text{Arctan } (Ue/Un)$$

Standard deviation of wind direction, $\sigma(\Theta u)$, using Campbell Scientific algorithm:

$$\sigma(\Theta u)=81(1-U/S)^{1/2}$$

***** 70 SAMPLE *****

FUNCTION
This instruction stores the value from each specified input location. The value(s) stored are those in the input location(s) when Instruction 70 is executed with the Output Flag set high.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Repetitions
02:	4	Starting input location no.

Outputs Generated: 1 for each repetition

*** 71 AVERAGE ***

FUNCTION
This instruction stores the average value over the given output interval for each input location specified.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	4	Starting input location no.

Outputs Generated: 1 per repetition

*** 72 TOTALIZE ***

FUNCTION
This instruction stores totalized value over the given output interval for each input location specified.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	4	Starting input location no.

Outputs Generated: 1 per repetition

*** 73 MAXIMIZE ***

FUNCTION
This instruction stores the MAXIMUM value taken (for each input location specified) over a given output interval. An internal FLAG is set whenever a new maximum value is seen. This FLAG may be tested by Instruction 79. Time of occurrence maximum value(s) is OPTIONAL output information, which is formatted and activated by entering one of the following CODES for Param. 2.

Code	**Options**
00	Output value ONLY
01	Output value with SECONDS
10	Output value with HOUR-MINUTE
11	Output value with HR-MIN,SEC

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Time of maximum (optional)
03:	4	Starting input location no.

Outputs Generated: 1 per repetition
(1 or 2 additional outputs per repetition with time option)

*** 74 MINIMIZE ***

FUNCTION
Operating in the same manner as Program 73, this instruction is used for storing the MINIMUM value (for each input location specified) over a given output interval.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Time of minimum (optional)
03:	4	Starting input location no.

Outputs Generated: 1 per repetition
(1 or 2 additional outputs per repetition with time option)

*** 75 STANDARD AND WEIGHTED *** VALUE HISTOGRAM

FUNCTION
Processes input data as either a standard histogram (frequency distribution) or a weighted value histogram.

The standard histogram outputs the fraction of the Output Interval that the value in a specified input location (defined as the bin select value) is within a particular sub-range of the total specified range. A counter in the bin associated with each sub-range is incremented whenever the value falls within that sub-range. The value which is output to Final Storage for each bin is computed by dividing the accumulated total in each bin by the total number of scans. This form of output is also referred to as a frequency distribution.

The weighted value histogram uses data from 2 input locations. One location contains the bin select value; the other contains the weighted value. Each time the instruction is executed, the weighted value is added to a bin. The sub-range that the bin select value is in determines the bin to which the weighted value is added. When the Output Flag is set, the value accumulated in each bin is divided by the TOTAL number of input scans to obtain the values that are output to Final Storage. These values are the contributions of the sub-ranges to the overall weighted value. To obtain the average of the weighted values that occurred

while the bin select value was within a particular sub-range, the value output to Final Storage must be divided by the fraction of time that the bin select value was within that particular sub-range (i.e., a standard histogram of the bin select value must also be output).

For either histogram, the user must specify: 1) the number of repetitions, 2) the number of bins, 3) a form code specifying whether a closed or open form histogram is desired (see below), 4) the bin select value input location, 5) the weighted value input location (see below), 6) the lower range limit, and 7) the upper range limit.

The standard histogram (frequency distribution) is specified by entering "0" in the weighted value input location parameter. Otherwise, this parameter specifies the input location of the weighted value. When more than one repetition is called for, the bin select value location will be incremented each repetition and the weighted value location will remain the same (same weighted value sorted on the basis of different bin select values). The weighted value location will be incremented if it is entered as an indexed location (key "C" at some point while keying in Parameter 5; two dashes, --, will appear on the right of the display). At the user's option, the histogram may be either closed or open. The open form includes all values below the lower range limit in the first bin and all values above the upper range limit in the last bin. The closed form excludes any values falling outside of the histogram range.

The difference between the closed and open form is shown in the following example for temperature values:

Lower range limit	10°C
Upper range limit	30°C
Number of bins	10

	Closed Form	Open Form
Range of first bin	10 to 11.99 deg.	<12 deg.
Range of last bin	28 to 29.99 deg.	>28 deg.

A common use of a closed form weighted value histogram is the wind speed rose. Wind speed values (the weighted value input) are accumulated into corresponding direction sectors (bin select input).

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	4	Number of bins
03:	2	Form code (0=open form, 1=closed form)
04:	4	Bin select value input location no.
05:	4	Weighted value input location no. (0 = frequency distribution option)
06:	FP	Lower limit of range
07:	FP	Upper limit of range

Outputs Generated: Number of Bins * Repetitions

***** 77 RECORD REAL TIME *****

FUNCTION
This Instruction stores the current time in Final Storage. At midnight the clock rolls over from 23:59 to 00:00. The day also changes.

If hourly or daily summary data is output, it may be desirable to have the previous day output, since that is when the measurements were made. Entering a 2 for the day code causes the previous day to be output if it is the first minute of the day. Similarly, entering 2 for the hour-minute code causes 2400 instead of 0000 to be output (the next minute is still 0001). When day and hour-minute are both output, a 2 for either code results in the previous day at 2400.

If year is output along with a 2 option in day or hour-minute, the previous year will be output during the first minute of the new year.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Enter appropriate TIME option code

Outputs Generated: 1 for each time parameter selected

Code	Result
xxx1	SECONDS (with resolution of 0.125 sec.)
xx1x	HOUR-MINUTE
xx2x	HOUR-MINUTE, 2400 instead of 0000
x1xx	JULIAN DAY
x2xx	JULIAN DAY, previous day during first minute of new day
1xxx	YEAR

Any combination of Year, Day, HR-MIN, and seconds is possible (e.g., 1011: YEAR, HR-MIN, SEC).

*** 78 SET HIGH OR LOW RESOLUTION *** DATA STORAGE FORMAT

FUNCTION
This instruction enables high resolution (5 character) or low resolution (4 character) final data storage format. Instruction 78 should be entered ahead of the output instructions for which the specified resolution is desired. The default format is low resolution. At the beginning of each program table execution, the low resolution format is automatically enabled.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	0 = low resolution; 1 = high resolution

Outputs Generated: 0

*** 79 SAMPLE ON MAXIMUM OR MINIMUM ***

FUNCTION

Instruction 79 samples specified input location values at the time a new maximum or minimum value is detected by a previous Maximize (73) or Minimize (74) Instruction. When the Output Flag is set, the values copied to Intermediate Storage are transferred to Final Storage.

Instruction 79 must directly follow the maximum or minimum Instruction to which it refers. If the previous Instruction 73 or 74 has more than 1 repetition, Instruction 79 samples whenever a new maximum or minimum is detected in any of the locations. If sampling is to occur only when a specific input location shows a new maximum or minimum, the previous Maximize or Minimize Instruction should have one rep referring to that input location.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions (number of sequential locations to sample)
02:	4	Starting input location no.

Outputs Generated: 1 per repetition

*** 80 SET ACTIVE STORAGE AREA ***

FUNCTION

Instruction 80 is used to redirect Output data to either of the Final Storage areas or to Input Storage and to set the array ID for Final Storage. The area set by Instruction 80 remains active until changed by another Instruction 80 or the table ends. At the beginning of each table the Active Output area is set to Final Storage Area 1.

When directed to Final Storage, the second parameter can be used to set the output array ID. Instruction 80 should follow the instruction setting flag 0. If parameter 2 is 0, the array ID is determined by the instruction location number of Instruction 80 or by the instruction that set the Output Flag, whichever comes last. When data are sent to Input Storage, no array ID is sent. Output array ID's can be indexed within a loop.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Storage area option 00 or 01 = Final Storage Area 1 02 = Final Storage Area 2 03 = Input Storage Area
02:	4	Starting input location destination if option 03 Output Array ID if options 0-2 (1-511 are valid IDs)

*** 82 STANDARD DEVIATION IN TIME ***

FUNCTION
Calculate the standard deviation (STD DEV) of a given input location. The standard deviation is calculated using the formula:

$$S = ((\Sigma X_i^2 - (\Sigma X_i)^2/N)/N)^{1/2}$$

where X_i is the ith measurement and N is the number of samples.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	4	Starting input location no.

Outputs Generated: 1 per repetition

SECTION 12. PROGRAM CONTROL INSTRUCTIONS

TABLE 12-1. Flag Description

Flag 0	Output Flag
Flag 1 to 8	User Flags
Flag 9	Intermediate Processing Disable Flag

TABLE 12-2. Command Codes

0	Go to end of program table[2]
1-9, 79-99	Call Subroutine 1-9, 79-99[1]
10-19	Set Flag 0-9 high
20-29	Set Flag 0-9 low
30	Then Do
31	Exit loop if true
32	Exit loop if false
41	Set Port 1 high
51	Set Port 1 low
61	Toggle Port 1
71	Pulse Port 1

1 98 is a special subroutine that can be called by Control port 2 going high; see Instruction 85 for details.

2 If this command is executed while in a subroutine, execution jumps directly to the end of the table that called the subroutine.

*** 83 IF CASE X < F ***

FUNCTION
If the value in the location specified in the Begin Case Instruction 93 is less than the fixed value entered as parameter 1 then execute the command in parameter 2 then go to the end of the case statement when the next Instruction 83 occurs. Else, continue to next instruction. See Instruction 93 for an example.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	FP	Fixed value
02:	2	Command

*** 85 LABEL SUBROUTINE ***

FUNCTION
This instruction marks the start of a subroutine. Subroutines are a series of instructions beginning with Instruction 85 and terminated with Instruction 95, END. All subroutines must be placed in Table 3 (Subroutine Table). When a subroutine is called by a command in a Program Control Instruction, the subroutine is executed, then program flow continues with the instruction following that which called the subroutine.

Subroutines may be called from within other subroutines (nested). The maximum nesting level for subroutines is 6 deep. Attempts to nest more than 6 deep will not be detected at compilation, but will result in a run time error. When the sixth subroutine attempts to call the seventh, error 31 will be displayed. Execution will not branch to the seventh subroutine; it will continue with the Instruction following that calling the subroutine.

98 PORT INTERRUPT SUBROUTINE. If subroutine 98 is included in Table 3 then Port 2 will cause an interrupt on the rising edge and the subroutine will be executed. This subroutine can also be called from any table.

This subroutine can interrupt Table 1 or 2 or can occur when neither Table is being executed. When port 2 activating 98 goes high during the execution of a table, the instruction being executed is completed before the subroutine is run (i.e. it is as if the subroutine was called by the next instruction).

The priority is 98, Table 1, Table 2.

While 98 is being executed as a result of the C2 going high, that port interrupt is disabled (i.e., the subroutine must be completed before the port going high will have any effect).

NOTE: If Control Port 2 is used for pulse measurements or interrupt subroutines, the CR510 will not go into the quiescent power state (0.7 mA), if Control Ports 2 is high.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Subroutine number (1-9, 79-99)

*** 86 DO ***

FUNCTION
This Instruction unconditionally executes the specified command.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Command (Table 12-2)

*** 87 LOOP ***

FUNCTION
Instructions included between the Loop Instruction and the End Instruction (95) are repeated the number of times specified by the iteration count (Parameter 2), or until an Exit Loop command (31,32) is executed by a Program Control Instruction within the Loop. If 0 is entered for the count, the loop is repeated until an Exit Loop command is executed.

The first parameter, delay, controls how frequently passes through the loop are made. Its units are multiples of the table execution interval. A delay of 0 means that there is no delay between passes through the loop. Each time the table is executed all iterations of the loop will be completed and execution will pass on to the following instructions. If the delay is 5, every fifth time that the execution interval comes up, one pass through the loop is made; only those instructions in the loop will be executed and other portions of the table are not executed in the interim. When a loop with delay is executed, the next execution starts at the loop, skipping over any previous and following instructions in the table.

When a fixed number of iterations are executed, the time spent in the loop is equal to the product of the execution interval, delay, and the number of iterations. For example, a loop with a delay of 1 and a count of 5 will take 5 seconds if the execution interval is 1 second. When the loop is first entered, one pass through the loop is made, then the CR510 delays until the next execution interval and makes the second pass through the loop. After making the fifth pass through the loop, there is the fifth delay, after which execution passes to the instruction following the END instruction which goes with the loop.

While in a loop with delay, the table will not be initiated at each execution interval. (However, the overrun decimals will not be displayed.) Some consequences of this are: The Output Flag will not be automatically cleared between passes through the loop. Because Table 2 cannot interrupt Table 1, Table 2 will not be executed while Table 1 is in a loop with delay. Table 1 will not interrupt Table 2 in the middle of an output array. Thus, if the Output Flag is set in Table 2 prior to entering the loop or within the loop, the flag must be specifically cleared before the end of the pass if Table 1 is to be executed.

Input locations for Processing Instructions within a loop can be entered as Indexed locations. An Indexed location causes the input location to be incremented by 1 with each pass through the loop. (The Index counter is added to the location number in the program table.) Input locations which are not indexed will remain constant.

To specify an Indexed location, depress the C key at some point while keying in the digits for the input location and before entering the location with the A key. Two dashes, --, appear in the two right most characters of the display, indicating the entry is Indexed.

When the same output processing is required on values in sequential input locations, it must be accomplished by using the repetitions parameter of the Output Instruction, not by indexing the input location within a loop.

An Output Instruction within a loop is allotted the same number of Intermediate Storage locations as it would receive if it were not in the loop. For example, the average instruction with a single repetition is allotted only two Intermediate locations: one for the number of samples and one for the running total. Each time through the loop the sample counter is incremented and the value in the referenced input location is added to the total. If the input location is indexed, the values from all input locations are added to the same total.

Note that if the Output Flag is set prior to entering the loop in the above example, 10 values will be output. The first will be the average of all the readings in locations 1-10 since the previous output. Because the Intermediate locations are zeroed each time an output occurs, the next nine values will be the

current values (samples at the time of output) of locations 2-10.

Loops can be nested. Indexed locations within nested loops are indexed to the inner most loop that they are within. The maximum nesting level in the CR510 is 11 deep. This applies to If Then/Else comparisons and Loops or any combination thereof. An If Then/Else comparison which uses the Else Instruction (94) counts as being nested 2 deep.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Delay
02:	4	Iteration count

*** 88 IF X COMPARED TO Y ***

FUNCTION
This Instruction compares two input locations and, if the result is true, executes the specified Command. The comparison codes are given in Table 12-3.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location for X
02:	2	Comparison code (Table 12-5)
03:	4	Input location for Y
04:	2	Command (Table 12-2)

TABLE 12-3. Comparison Codes

Parameter 1	Function
1	IF X = Y
2	IF X ≠Y
3	IF X ≥ Y
4	IF X < Y

*** 89 IF X COMPARED TO F ***

FUNCTION
This Instruction compares an input location to a fixed value and, if the result is true, performs the specified Command. The comparison codes are given in Table 12-3.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location for X
02:	2	Comparison code (Table 12-3)
03:	FP	Fixed value
04:	2	Command (Table 12-2)

*** 90 STEP LOOP INDEX ***

FUNCTION
When used within a Loop (Instruction 87), Instruction 90 will increment the index counter by a specified amount after the first time through the loop, thus affecting all indexed input location parameters in subsequent instructions. For example, if 4 is specified, the index counter will count up by 4 (0,4,8,12,...) inside the loop. Instruction 90 does not affect the loop counter which still counts by 1.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Increment for the loop index counter

*** 91 IF FLAG / PORT / MODEM ***

FUNCTION
This Instruction checks the status of one of the ten Flags, one of the two ports, or the serial I/O port modem enable and conditionally performs the specified Command. Ports can be indexed (--) with the C key.

The first Parameter specifies the condition to check:

1X	Execute command if Flag X is high
2X	Execute command if Flag X is low
40	Execute command if modem is on
4P	Execute command if port P is high
50	Execute command if modem is off
5P	Execute command if port P is low

where P = 1 or 2

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Flag or Port condition to check
02:	2	Command (Table 12-2)

*** 92 IF TIME ***

FUNCTION
The user specifies the number of minutes or seconds into an interval, the duration of the interval, and a command. The command is executed each time the real time is the specified time into the interval. The "If" condition will always be false if 0000 is entered as the time interval.

The time interval is synchronized with the datalogger's time; if a 60 minute time interval is specified with 0 minutes into the interval, the Command will be executed each hour on the hour. The time interval is synchronized internally by making a modulo divide (Instruction 46) of the number of minutes since midnight by the specified real time interval. If the result is 0, the interval is up. Thus, the first interval of the day always starts at midnight (0 minutes). The maximum interval is 1440 minutes.

The time into an interval is only true the first time Instruction 92 is executed within a given minute (or second). For example, if the command is to set the Output Flag at 0 minutes into a 10 minute interval, and the execution interval of the table is 10 seconds, every 10 minutes there will only be one output generated by this instruction, not five.

The time into interval and the interval may be entered in seconds for intervals less than 60 seconds.

To enter the times in seconds, press "C" after keying in the number of seconds into the interval for Parameter 1; two dashes will appear to the right of the number (XXXX--). When the time into interval is entered as seconds (XXXX--), the time interval will also be interpreted as seconds.

The maximum number of seconds that can be entered is 59 for Parameter 1 and 60 for Parameter 2.

The Output Flag (Flag 0) is a special case in that it will automatically be set low if it is not time to set it.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Time into interval (minutes or seconds if entered XXXX--)
02:	4	Time interval (units same as above)
03:	2	Command (Table 12-2)

*** 93 BEGIN CASE STATEMENT ***

The value in the specified input location is compared against parameters in following If Case instructions (83). When a comparison is true, the command in the If Case instruction is executed and the program flow goes to the End instruction (95) associated with the Begin Case instruction.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location for subsequent comparisons

EXAMPLE:

01:	P93	Case
01:	2	Case Loc
02:	P83	If Case Location < F
01:	69.4	F
02:	3	Call Subroutine 3
	else	
03:	P83	If Case Location < F
01:	72	F
02:	10	Set high Flag 0 (output)
	else	
04:	P83	If Case Location < F
01:	77.3	F
02:	30	Then Do
05:	P30	Z=F
01:	0	F
02:	0	Exponent of 10
03:	25	Z Loc :
06:	P95	End *Then Do*
07:	P95	End *of Case Statement*

*** 94 ELSE ***

FUNCTION
When Command 30 (Then/Else) is used with an If Instruction, the Else Instruction is used to mark the start of the instructions to execute if the test condition is false (Figure 3.8-1). The Else Instruction is optional; when it is omitted, a false comparison will result in execution branching directly to the End Instruction. Instruction 94 has no parameters.

*** 95 END ***

FUNCTION
Instruction 95 is used to indicate the end/return of a subroutine (Instruction 85), the end of a loop (Instruction 87), the end of an If Then/Else sequence (Instructions 88-92 when used with command 30), or the end of a Case sequence (Instruction 93). The End Instruction has no parameters.

*** 96 ACTIVATE SERIAL DATA OUTPUT ***

FUNCTION
Instruction 96 is used to activate Storage Module (SM192/SM716 or Card Storage Module) or serial data (printer) output. Normally Instruction 96 is placed in the program table after all Output Instructions have been entered and is executed each time the Table is executed. In this situation any data sent to Final Storage is output as soon as possible. However, by using Program Control Instructions to allow execution of Instruction 96 only at certain times, the user can control when the output device(s) are active. Instruction 96 allows a choice of serial data format and the selection of Addressed or Pin Enabled device for the serial print output.

A single parameter is used to select whether the instruction is to control the "printer", Storage Module, or Card Storage Module output, and if the printer is selected, the format and baud rate. The Instruction must be entered separately for each device that is to receive output.

If both Final Storage areas are in use, Instruction 96 will send data from the area which is currently active. Final Storage Area 1 is active at the start of each Table. Instruction 80 can be used to change the active area. The Area set by Instruction 80 remains active until changed by another Instruction 80 or the Table ends (at which time Area 1 becomes the active Area 1). Instruction 80 can also direct output to Input Storage, in which case Instruction 96 assumes Final Storage Area 1.

If the CR510 is already communicating on the 9-pin connector when Instruction 96 is executed, the output request is put in a queue and program execution continues. As the 9-pin connector becomes available, each device in the queue will get its turn.

The request is not put in the queue if the same device is already in the queue. The data contained in the queue (and which determine a unique entry) are baud rate (if applicable), and the Final Storage Area. Instruction 98 to send characters also uses this queue.

When an entry reaches the top of the queue, the CR510 sends all data accumulated since the last transfer to the device up to the location of the DSP at the time the device became active (this allows everything in the queue to get a turn even if data is being stored faster than it can be transferred to a particular device).

The "other Final Storage Area" device option (the non-active area) allows a "fast" Final Storage area to be transferred to the main area on some trigger condition so there is some history recorded prior to the trigger condition (Section 8.8). The source of data is the currently active Final Storage Area set by Instruction 80 (default = 0 or 1).

When the baud rate code specifying a checksum is used, the checksum of the data is sent as the last piece of data in the data array. This only works when sending out comma separated data. See Section 5.1 to learn about the checksum.

> **NOTE:** All memory pointers are positioned to the DSP location when the datalogger compiles a program. For this reason, Always retrieve uncollected data before making program changes.

TABLE 12-4. Baud Rate Codes

Code	Baud Rate
0	300
1	1200
2	9600
3	76800
4	300 with checksum
5	1200 with checksum
6	9600 with checksum
7	76800 with checksum

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Option Device

ADDRESSED PRINT DEVICE, y = Baud code
1y = Printable ASCII
2y = Comma Separated ASCII
3y = Binary Final Storage Format
7N = Storage Module N (N=1-8; Section 4.4.2) (Stored in Binary Format)
7N-- = Output File Mark to Storage Module N

SERIAL PRINTER, COMPUTER, OR PIN-ENABLED PRINT DEVICE, y = Baud code (SDE pulled high)
4y = Printable ASCII
5y = Comma Separated ASCII
6y = Binary Final Storage Format

TRANSFER DATA TO OTHER FINAL STORAGE AREA
80 = New data only
81 = All data

*** 97 INITIATE *** TELECOMMUNICATIONS

Instruction 97 is used to have the CR510 initiate telecommunications in response to certain conditions. This is commonly referred to as callback. This instruction can be used to initiate three different types of calls:

A) to expect a modem and computer to answer the call,

B) to program a voice modem to send a voice message to a phone, or

C) to call a remote datalogger and collect a specified number of input locations.

D) to call a pager

The section mostly discusses the computer communication method. See the Voice Modem manual for a more complete description of the voice callback. See Appendix H more complete information on the datalogger to datalogger communications.

Briefly, when the instruction is executed and the interrupt disable flag (Parameter 2) is low, the CR510 initiates a call to a computer. The CR510 starts communicating with the modem identified in Parameter 1 using the additional modem ID#s, phone numbers, or other commands given in the subsequent Instruction(s) 63 or 68. Once the computer answers the call, the CR510 sends the ID# (Parameter 8) to identify itself to the computer. When the computer sends back the same ID#, the CR510 will go into the normal telecommunications mode (Section 5) and the computer is now in charge of the call. The computer uses the ID# to know what station file/setup to use to control the calling datalogger. The CR510 will not send any data (or do anything else) without first receiving a command to do so. CSI's datalogger support software enables PCs to automatically answer calls, retrieve data, and instruct the datalogger on what to do now. See Appendix G for a phone modem callback example including the computer setup.

If the correct response is not received from the computer within the time allotted in Parameter 3 the datalogger hangs up and calls back later. The number of unsuccessful attempts is stored in the Failure Input Location (Parameter 7). Once a successful call is made, the Failure Input Location is reset to 0.

Be careful in calling P97 from a conditional statement or subroutine. Instruction 7 must be run at least twice for each call, once to initiate the call and another time to completely reset and clean up memory values used to make the call after the call was successful or has stopped trying to call.

Parameter 1:

The datalogger will call out using the modem specified in Parameter 1. If the call is to go from one type of modem to another, list the first modem to be used. For example, when calling through a RF modem to phone, you would use a code for an RF Modem. Table 12-5 shows the different modem/baud rate options valid for this Parameter.

When the phone modem is specified, the following commands are sent to the modem before the phone number: ATV0[Enter], ATS7=180[Enter], and ATDT. The first command causes the modem to respond with digits rather than words. The second command causes the modem to wait for 180 seconds after dialing for the carrier. The third command causes the modem to dial the phone number (in Instruction(s) 63 or 68) in "Touch Tones". Additional commands can be entered as part of the telephone number (e.g. "," for a delay or "P" for pulse dialing). Additional commands must be entered into the subsequent instructions as their ASCII equivalents (Appendix E).

Parameter 2:

When the instruction is executed and the interrupt disable flag (Parameter 2) is low, the CR510 initiates the call. The datalogger will continue to attempt communications until the interrupt disable flag has been set high. As soon as the flag is set high, the datalogger quits trying to initiate the call. After a successful call has ended, the flag is automatically set high. (After a voice callback, the flag is not automatically set high.)

Be careful to make sure the only times this flag gets set low or high are the times you want to initiate or disable the call.

Parameter 3:

Time limit in 1 sec. units. Limit on the call is timed from the start of the instruction until a valid ID# is received by the CR510. This time limit includes the dialing time.

If the complete ID# is not received by the CR510 within the time allotted in parameter 3, the datalogger hangs up and waits for the time for the next attempt or retry.

Parameter 4:

The CR510 will repeat the call at a fast interval specified by Parameter 4 (in 1 sec. units).

Parameter 5:

Number of times the CR510 will attempt retries at the fast interval (parameter 4), before attempting calls at the slow interval (parameter 6).

Parameter 6:

Delay between slow retries (in 1 min. units).

Note about retry rates, Parameter 4 & 6:

The actual delay between retries (both the fast and slow) has a random factor built in, which is added as an offset to the delay specified. The random factor prevents calls from different stations from occurring at the same time. This offset will range between 0 and ½ of the delay specified. The resolution of the timer for these delays is the execution interval of the table in which the Instruction 97 is initiated. The randomized retry time is divided by the execution interval to determine how many times Instruction 97 must be executed before it tries to call again. The Instruction 97 must be executed each time the table is.

Parameter 7:

The number of unsuccessful attempts is stored in the Failure Input Location (Parameter 7). Once a successful call is made, the Failure Input Location is reset to 0.

Parameter 8:

Once a connection is established, the datalogger will send (in ASCII at the specified baud rate) the ID# specified in Parameter 8 to identify itself. The computer must send the ID# back to the datalogger to finish establishing a valid link. The ID number will be sent every 4 seconds until the CR510 receives the same ID# back or the time specified in parameter 3 expires.

When the CR510 receives a correct character of the ID#, it restarts the 4 second timer. The CR510 must receive the correct ID# (each digit in order) with no more than 4 seconds between each digit. If an incorrect character is detected or the 4 second timer expires, the CR510 will immediately resend the correct ID#. If the complete ID# is not received by the CR510 within the time allotted in parameter 3, the datalogger hangs up and waits for the time for the next attempt or retry.

Additional programming requirements:

Instruction 97 must be followed by at least one Instruction 63 or 68 which specifies the call's modem path and/or phone number. Instruction 63 should be used unless one of the parameters is 3 or 4 digits, then Instruction 68 should be used.

If the first parameter of the first Instruction 63/68 is "X" (88), the second parameter represents a delay in seconds. This delay is the amount of time from when the call is answered and when the ID number is sent. This is useful when calling pagers.

The RF Station IDs or phone numbers are entered 1 digit at a time.

Decimal equivalents of specific ASCII characters (Appendix E) are used to identify breaks in the dialing string or changing the type of modem communicating with. Separate RF station IDs with a space (32). After the last RF ID (when no other modem is being used), put

an “F” (70). Indicate a switch from RF to phone with a space (32) followed by a “T” (84).

A Carriage Return (13) is used to end the series of characters to be used to initiate the call. Instruction 97 will **never** make a valid call if a 13 is not the last parameter of the last Instruction 63 or 68. Any unused parameters after the 13 command should be left as 0.

TABLE 12-5. Option Code for Modem Type and Baud Rate

	Baud Rate		
Modem	300	1200	9600
RF (RF95)	00	01	02
Direct (Short Haul)	10	11	12
Phone (DC112/COM200)	20	21	22
Voice Call-Back	N/A	31*	N/A
Data Call-Back	40	41	42

*See Voice Modem manual for the description of voice call-back.

CAUTION: Instruction 97 with the voice option and the SDI-12 instructions may not be in same table. Instruction 97 must be in Table 1 and the SDI-12 instructions must be in Table 2.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Modem option and baud rate code. Left digit specifies the modem being used and the right, the baud rate.
02:	2	Interrupt disable flag
03:	4	Time limit on call, 1 sec. units
04:	4	Delay between fast retries, 1 sec. units
05:	2	No. of retries at fast rate
06:	4	Delay between slow retries, 1 min. units
07:	4	Input location to store no. of tries
08:	4	ID to send

*** 98 SEND CHARACTER ***

Instruction 98 is used with Instruction 63 to send a character or string of characters (up to 15) to the printer. The printer may be either an addressed or a pin-enabled (Section 6.2). Instruction 63 must immediately follow 98. The character or characters to send are entered in Instruction 63 as the decimal equivalents (99 is the maximum number allowed) of the 7 bit ASCII character (sent as 8 bits, no parity). For example, to send the ASCII character control R, 18 would be entered. Enter a null (0) to terminate the string. Appendix E contains a listing of the ASCII characters.

If the 9 pin connector is already active when Instruction 98 is executed, the output request is put in a queue (see Instruction 96).

This instruction can be used to send a control character to activate some listing device. The specified character(s) is sent at the time Instruction 98 and 63 are executed.

*** 111 RUN PROGRAM FROM FLASH ***

FUNCTION
This instruction is used to load a program stored in FLASH into RAM. The program in RAM is replaced by the Flash program specified in Parameter 1. If Parameter 1 is indexed, the CR510 will compile the program like the ∗6 mode. Use this instruction with caution.

NOTE: The memory allocation (∗A) must be the same between the program in RAM and the program that is loaded from Flash. If the memory allocations are not the same, the CR510 will reallocate memory and the data in Final Storage will be lost. Do not use the auto program allocation (0 for Parameter 5 in the ∗A mode), because data will be lost.

PARAM. NUMBER	DATE TYPE	DESCRIPTION
01:	2	Flash program number (name)

*** 120 TGT1 TELONICS GOES ***

FUNCTION
This instruction is used to transmit data from CR510 Final Storage via a GOES satellite to a central ground station. See the TGT1 manual for information on Instruction 120.

*** 121 ARGOS ***

FUNCTION
This instruction is used to transmit data from CR510 Final Storage via an ARGOS satellite. See the ARGOS Interface Notes for information on Instruction 121.

*** 123 Automatic Programming of a TGT1 ***

FUNCTION
Instruction 123 performs Automatic Programming of TGT1 GOES Transmitter. See the TGT1 manual for information on Instruction 123.

SECTION 13. CR510 MEASUREMENTS

13.1 FAST AND SLOW MEASUREMENT SEQUENCE

The CR510 makes voltage measurements by integrating the input signal for a fixed time and then holding the integrated value for the analog to digital (A/D) conversion. The A/D conversion is made with a 13 bit successive approximation technique which resolves the signal voltage to approximately one part in 7500 of the full scale range on a differential measurement (e.g., 1/7500 x 2.5 V = 333 uV). The resolution of a single-ended measurement is one part in 3750.

Integrating the signal removes noise that could create an error if the signal were instantaneously sampled and held for the A/D conversion. There are two integration times which can be specified for voltage measurement instructions, the slow integration (2.72 ms), or the fast integration (250 us). The slow integration time provides a more noise-free reading than the fast integration time. Integration time is specified in the Range Code of the measurement instruction. Instructions 1 - 14 RANGE codes:

Slow (2.72 ms Integration time)	**Fast (250 us Integration time)**	**60 Hz rejection**	**50 Hz rejection**	**Full Scale range**
1	11	21	31	±2.5 mV
2	12	22	32	±7.5 mV
3	13	23	33	±25 mV
4	14	24	34	±250 mV
5	15	25	35	±2500 mV

In the United States one of the most common sources of noise is 60 Hz from AC power lines. Where 60 Hz noise is a problem, range codes 21 - 25 should be used. Two integrations are made spaced 1/2 cycle apart (Figure 13.2-2), which results in the AC noise integrating to 0. Integration time for the 2500 mV range is 1/10 the integration time for the other gain ranges (2.72 ms). For countries with 50 Hz power Range codes 31 - 35 are used for 50 Hz rejection.

There are several situations where the fast integration time is preferred. The fast integration time minimizes time skew between measurements and increases the throughput rate. The current drain on the CR510 batteries is lower when the fast integration time is used. The fast integration time should always be used with the AC half bridge (Instruction 5) when measuring AC resistance or the output of an LVDT. An AC resistive sensor will polarize if a DC voltage is applied, causing erroneous readings and sensor decay. 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 time would result in most of the integration taking place after the signal had disappeared.

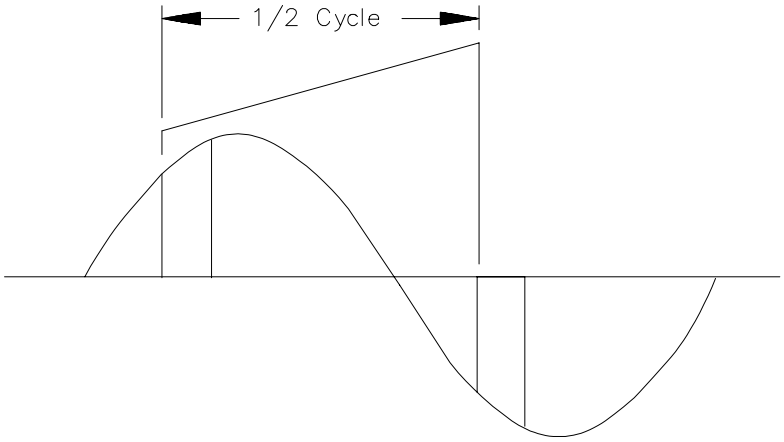


FIGURE 13.1-1. 50 and 60 Hz Noise Rejection

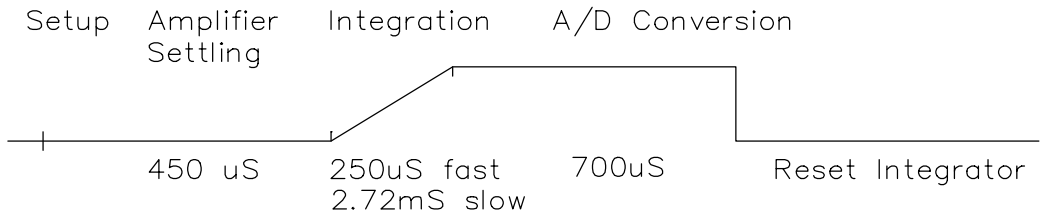


FIGURE 13.2-1. Timing of Single-Ended Measurement

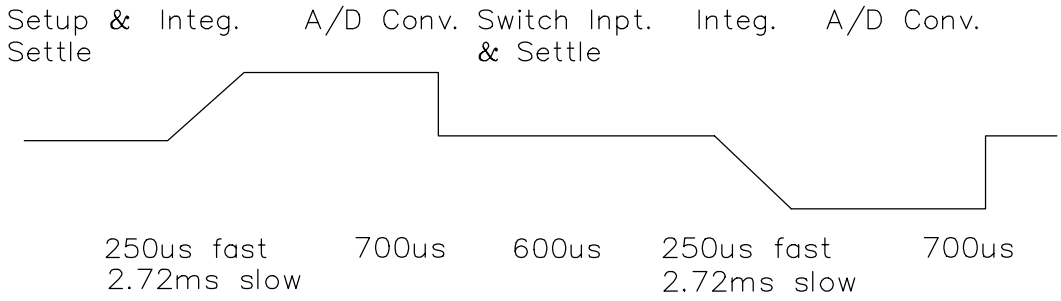


FIGURE 13.2-2. Differential Voltage Measurement Sequence

13.2 SINGLE-ENDED AND DIFFERENTIAL VOLTAGE MEASUREMENTS

> **NOTE:** Either the high or low side of a differential channel can be used for single-ended measurements. Each side must be counted when numbering single-ended channels; e.g., the high and low sides of differential channel 1 are single-ended channels 1 and 2, respectively.

The timing and sequence of a single-ended measurement is shown in Figure 13.2-1. A single-ended measurement is made on a single input which is referenced to ground. A single integration is performed for each measurement. A differential measurement measures the difference in voltage between two inputs. The measurement sequence on a differential measurement involves two integrations. First with the high input referenced to the low, then with the inputs reversed (Figure 13.2-2).

The CR510 computes the differential voltage by averaging the magnitude of the results from the two integrations and using the polarity from the first. An exception to this is the differential measurement in Instruction 8 which makes only one integration.

Because a single-ended measurement is referenced to CR510 ground, any difference in ground potential between the sensor and the CR510 will result in an error in the measurement. For example, if the measuring junction of a copper-constantan thermocouple, used to measure soil temperature, is not insulated and the potential of earth ground is 1 mV greater at the sensor than at the point where the CR510 is grounded, the measured voltage would be 1 mV greater than the thermocouple output or approximately 25°C high.

Another instance where a ground potential difference creates a problem is in a case such as described in Section 7.2, where external signal conditioning circuitry is powered from the same source as the CR510. Despite being tied to the same ground, differences in current drain and lead resistance result in different ground potential at the two instruments. For this reason a differential measurement should be made on an analog output from the external signal conditioner. Differential measurements MUST be used where the inputs are known to be different from ground, such as is the case with the output from a full bridge.

In order to make a differential measurement, the inputs must be within the CR510 common mode range of ±2.5 V. The common mode range is the voltage range, relative to CR510 ground, within

which both inputs of a differential measurement must lie in order for the differential measurement to be made.

For example, if the high side of a differential input is at 2 V and the low side is at 1 V relative to CR510 ground, there is no problem; a measurement made on the +2.5 V range would indicate a signal of 1 V. However, if the high input is at 2.8 V and the low input is at 2 V, the measurement cannot be made because the high input is outside of the common mode range. The CR510 will indicate the overrange with the maximum negative number (Section 3.5.)

Problems with exceeding common mode range may be encountered when the CR510 is used to read the output of external signal conditioning circuitry if a good ground connection does not exist between the external circuitry and the CR510. When operating where AC power is available, it is not always safe to assume that a good ground connection exists through the AC wiring. If a CR510 is used to measure the output from a laboratory instrument (both plugged into AC power and referencing ground to outlet ground), it is best to run a ground wire between the CR510 and the external circuitry. Even with this ground connection, the ground potential of the two instruments may not be at exactly the same level, which is why a differential measurement is desired (Section 7.2).

If a differential measurement is used on a sensor that is not referenced to CR510 ground through a separate connection (e.g., a net radiometer), a jumper wire should be connected between the low side of the differential input and analog ground to hold the sensor in common mode range.

A differential measurement has better noise rejection than a single-ended measurement. Integrating the signal in both directions also reduces input offset voltage due to thermal effects in the amplifier section of the CR510. Input offset voltage on a single-ended measurement is less than 5 microvolts; the input offset voltage on a differential measurement is less than 1 microvolt.

A single-ended measurement is quite satisfactory in cases where noise is not a problem and care is taken to avoid ground potential problems. Channels are available for twice as many single-ended measurements. A single-ended measurement takes about half the time of a differential measurement, which is valuable in cases where rapid sampling is a requirement.

NOTE: Sustained voltages in excess of +16 VDC applied to the analog inputs will damage the CR510 input circuitry.

13.3 THE EFFECT OF SENSOR LEAD LENGTH ON THE SIGNAL SETTLING TIME

Whenever an analog input is switched into the CR510 measurement circuitry prior to making a measurement, a finite amount of time is required for the signal to stabilize at its correct value. The rate at which the signal settles is determined by the input settling time constant which is a function of both the source resistance, and input capacitance (explained below). The CR510 allows a 450 µs settling time before initiating the measurement. In most applications this settling time is adequate, but the additional wire capacitance associated with long sensor leads can increase the settling time constant to the point that measurement errors may occur. There are three potential sources of error which must settle before the measurement is made:

1. The signal must rise to its correct value.
2. A small transient (~5 mV) caused by switching the analog input into the measurement circuitry must settle.
3. A larger transient, usually about 40 mV/V, caused by the switched, precision excitation voltage used in resistive bridge measurements must settle.

The purpose of this section is to bring attention to potential measurement errors caused when the input settling time constant gets too large and to discuss procedures whereby the effects of lead length on the measurement can be estimated. In addition, physical values are given for three types of wire used in CSI sensors, and error estimates for given lead lengths are provided. Finally, techniques are discussed for minimizing input settling error when long leads are mandatory.

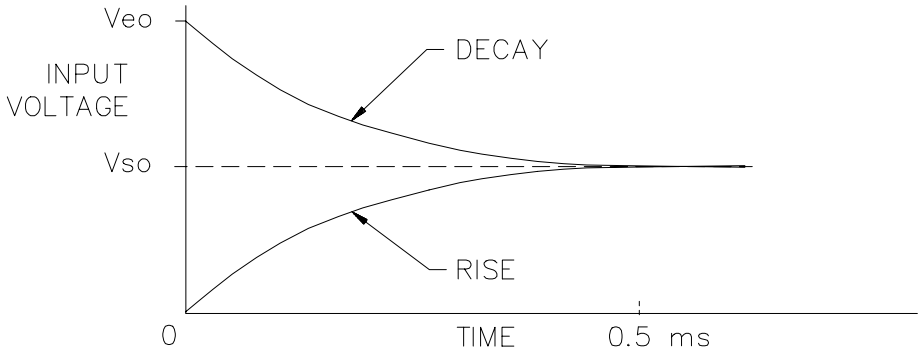


FIGURE 13.3-1. Input Voltage Rise and Transient Decay

13.3.1 THE INPUT SETTLING TIME CONSTANT

The rate at which an input voltage rises to its full value or that a transient decays to the correct input level are both determined by the input settling time constant. In both cases the waveform is an exponential. Figure 13.3-1 shows both a rising and decaying waveform settling to the signal level, Vso. The rising input voltage is described by Equation 13.3-1 and the decaying input voltage by Equation 13.3-2.

$$V_s = V_{so} (1-e^{-t/R_oC_T}), \text{ rise} \qquad [13.3\text{-}1]$$

$$V_s = V_{so} + (V_{eo}-V_{so}) e^{-t/R_oC_T}, \text{ decay} \qquad [13.3\text{-}2]$$

where V_s is the input voltage, V_{so} the true signal voltage, V_{eo} the peak transient voltage, t is time in seconds, R_o the source resistance in ohms, and C_T is the total capacitance between the signal lead and ground (or some other fixed reference value) in farads.

The settling time constant, τ in seconds, and the capacitance relationships are given in Equations 13.3-3 through 13.3-5,

$$\tau = R_oC_T \qquad [13.3\text{-}3]$$

$$C_T = C_f + C_wL \qquad [13.3\text{-}4]$$

$$C_f = 3.3 \text{ nfd} \qquad [13.3\text{-}5]$$

where C_f is the fixed CR510 input capacitance in farads, C_w is the wire capacitance in farads/foot, and L is the wire length in feet.

Equations 13.3-1 and 13.3-2 can be used to estimate the input settling error, V_e, directly. For the rising case, $V_s = V_{so}-V_e$, whereas for the decaying transient, $V_s = V_{so}+V_e$. Substituting these relationships for V_s in Equations 13.3-1 and 13.3-2, respectively, yields expressions in V_e, the input settling error:

$$V_e = V_{so} e^{-t/R_oC_T}, \text{ rise} \qquad [13.3\text{-}6]$$

$$V_e = V_{e'o} e^{-t/R_oC_T}, \text{ decay} \qquad [13.3\text{-}7]$$

Where $V_{e'o} = V_{eo}-V_{so}$, the difference between the peak transient voltage and the true signal voltage.

> **NOTE:** Since the peak transient, V_{eo}, causes significant error only if it is several times larger than the signal, V_{so}, error calculations made in this section approximate $V_{e'o}$ by V_{eo}; i.e., $V_{eo} = V_{eo}-V_{so}$.

If the input settling time constant, τ , is known, a quick estimation of the settling error as a percentage of the maximum error (V_{so} for rising, $V_{e'o}$ for decaying) is obtained by knowing how many time constants (t/τ) are contained in the 450 µs CR510 input settling interval (t). The familiar exponential decay relationship is given in Table 13.3-1 for reference.

TABLE 13.3-1. Exponential Decay, Percent of Maximum Error vs. Time in Units of τ

Time Constants	% Max. Error	Time Constants	% Max. Error
0	100.0	5	0.7
1	36.8	7	0.1
3	5.0	10	0.004

Before proceeding with examples of the effect of long lead lengths on the measurement, a discussion on obtaining the source resistance, R_o, and lead capacitance, C_wL, is necessary.

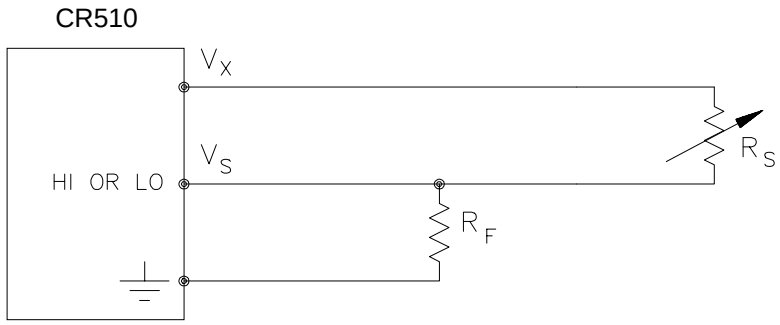


FIGURE 13.3-2. Typical Resistive Half Bridge

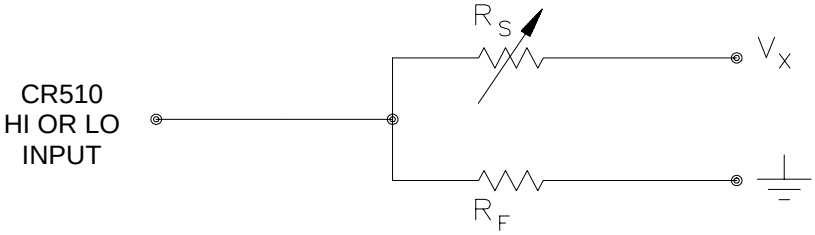


FIGURE 13.3-3. Source Resistance Model for Half Bridge Connected to the CR510

DETERMINING SOURCE RESISTANCE

The source resistance used to estimate the settling time constant is the resistance the CR510 input "sees" looking out at the sensor. For our purposes the source resistance can be defined as the resistance from the CR510 input through all external paths back to the CR510. Figure 13.3-2 shows a typical resistive sensor, (e.g., a thermistor) configured as a half bridge. Figure 13.3-3 shows Figure 13.3-2 re-drawn in terms of the resistive paths determining the source resistance Ro, is given by the parallel resistance of Rs and Rf, as shown in Equation 13.3-8.

$$R_o = R_s R_f/(R_s+R_f) \quad [13.3\text{-}8]$$

If R_f is much smaller, equal to or much greater than R_S, the source resistance can be approximated by Equations 13.3-9 through 13.3-11, respectively.

$$R_o \sim R_f,\ R_f << R_s \quad [13.3\text{-}9]$$

$$R_o = R_f/2,\ R_f = R_s \quad [13.3\text{-}10]$$

$$R_o \sim R_s,\ R_f >> R_s \quad [13.3\text{-}11]$$

The source resistance for several Campbell Scientific sensors are given in column 3 of Table 13.3-5.

DETERMINING LEAD CAPACITANCE

Wire manufacturers typically provide two capacitance specifications: 1) the capacitance between the two leads with the shield floating, and 2) the capacitance between the two leads with the shield tied to one lead. Since the input lead and the shield are tied to ground (often through a bridge resistor, R_f) in single-ended measurements such as Figure 13.3-2, the second specification is used in determining lead capacitance. Figure 13.3-4 is a representation of this capacitance, C_w, usually specified as pfd/ft. C_w is actually the sum of capacitance between the two conductors and the capacitance between the top conductor and the shield. Capacitance for 3 Belden lead wires used in Campbell Scientific sensors is shown in column 6 of Table 13.3-2.

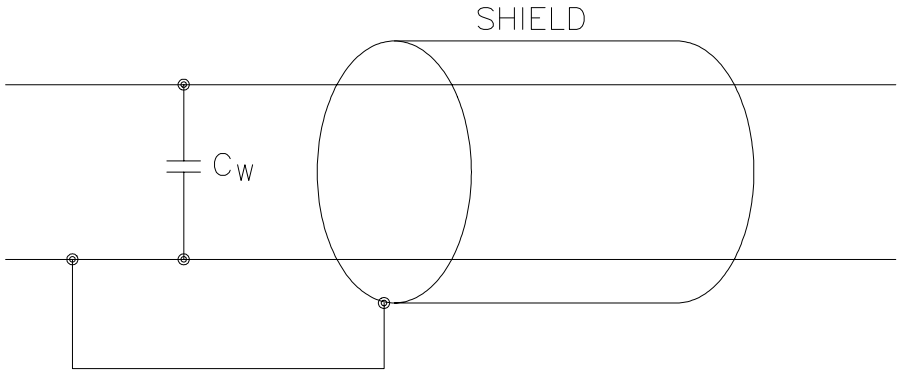


FIGURE 13.3-4. Wire Manufacturers Capacitance Specifications, C_w

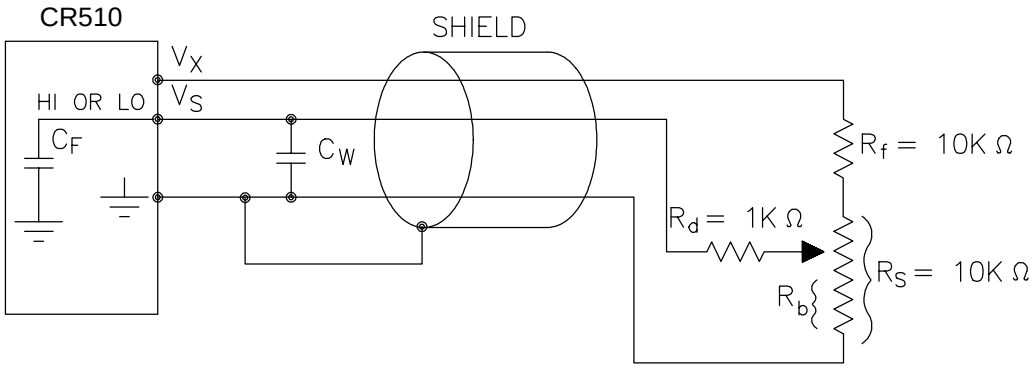


FIGURE 13.3-5. Model 024A Wind Direction Sensor

TABLE 13.3-2. Properties of Three Belden Lead Wires Used by Campbell Scientific

Belden Wire #	Conductors	Insulation	AWG	Rl (ohms/1000ft.)	C_w (pfd/ft.)
8641	1 shld. pair	polyethylene	24	23	42
8771	1 shld. 3 cond.	polyethylene	22	15	41
8723	2 shld. pair	polypropylene	22	15	62

DIELECTRIC ABSORPTION

The dielectric absorption of insulation surrounding individual conductors can seriously affect the settling waveform by increasing the time required to settle as compared to a simple exponential. Dielectric absorption is difficult to quantify, but it can have a serious effect on low level measurements (i.e., 50 mV or less). The primary rule to follow in minimizing dielectric absorption is: Avoid PVC insulation around conductors. PVC cable jackets are permissible since the jackets don't contribute to the lead capacitance because the jacket is outside the shield. Campbell Scientific uses only polyethylene and polypropylene insulated conductors in CR510 sensors (see Table 13.3-2) since these materials have negligible dielectric absorption. Teflon insulation is also very good but quite expensive.

13.3.2 EFFECT OF LEAD LENGTH ON SIGNAL RISE TIME

In the 024A Wind Vane, a potentiometer sensor, the peak transient voltage is much less than the true signal voltage (see Table 13.3-5). This means the signal rise time is the major source of error and the time constant is the same as if C_w were between the signal lead and ground as represented below.

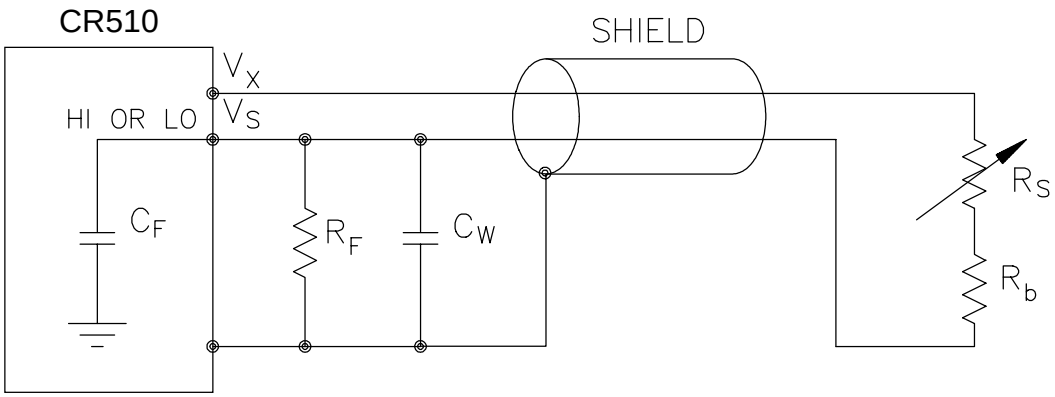


FIGURE 13.3-6. Resistive Half Bridge Connected to Single-Ended CR510 Input

R_o, the source resistance, is not constant because R_b varies from 0 to 10 kohms over the 0 to 360 degree wind direction range. The source resistance is given by:

$$R_o = R_d+(R_b(R_s-R_b+R_f)/(R_s+R_f)) =$$

$$R_d+(R_b(20k-R_b)/20k) \quad [13.3\text{-}12]$$

Note that at 360 degrees, R_o is at a maximum of 6k (R_b=10k) and at 0 degrees, R_o is 1k (R_b=0). It follows that settling errors are less at lower direction values.

The value of R_b for any direction D (degrees) is given by:

$$R_b(\text{kohms}) = (10k)(D)/360 \quad [13.3\text{-}13]$$

Equation 13.3-6 can be rewritten to yield the settling error of a rising signal directly in units of degrees.

$$\text{Error (degrees)} = De^{-t/(R_o(C_f+C_wL))} \quad [13.3\text{-}14]$$

Equation 13.3-12, -13 and -14 can be combined to estimate the error directly in degrees at various directions and lead lengths (Table 13.3-3). Constants used in the calculations are given below:

C_f = 3.3nfd

C_w = 41 pfd/ft., Belden #8771 wire

t = 450µs

TABLE 13.3-3. Settling Error, in Degrees, for 024A Wind Direction Sensor vs. Lead Length

Wind Direction	Error	
	L=1000 ft.	L=500 ft.
360°	66°	15°
270°	45°	9°
180°	21°	3°
90°	4°	0°

The values in Table 13.3-3 show that significant error occurs at large direction values for leads in excess of 500 feet. Instruction 4, Excite, Delay, and Measure, should be used to eliminate errors in these types of situations. Using a 10 ms delay, settling errors are eliminated up to lengths that exceed the drive capability of the excitation channel (~ 2000 ft.).

13.3.3 TRANSIENTS INDUCED BY SWITCHED EXCITATION

Figure 13.3-6 shows a typical half bridge resistive sensor, such as Campbell Scientific's Model 107 Temperature Probe, connected to the CR510. The lead wire is a single-shielded pair, used for conducting the excitation (V_x) and signal (V_s) voltages. When V_x is switched on, a transient is capacitively induced in V_s, the signal voltage. If the peak transient level, V_{eo}, is less than the true signal, V_{so}, the transient has no effect on the measurement. If V_{eo} is greater than V_{so}, it must settle to the correct signal voltage to avoid errors.

TABLE 13.3-4. Measured Peak Excitation Transients for 1000 Foot Lengths of Three Belden Lead Wires Used by Campbell Scientific

	----------V_{eo}(mV) ----------					
Vx(mV)	**R_f=1 kohm**			**R_f=10 kohm**		
	#	#	#	#	#	#
	8641	**8771**	**8723**	**8641**	**8771**	**8723**
2000	50	100	60	100	140	80
1000	25	65	40	60	90	40

NOTE: Excitation transients are eliminated if excitation leads are contained in a shield independent from the signal leads.

The size of the peak transient is linearly related to the excitation voltage and increases as the bridge resistor, R_f, increases. Table 13.3-4 shows measured levels of V_{eo} for 1000 foot lengths of three Belden wires used in Campbell Scientific sensors. Values are given for R_f equal to 1 kohm and 10 kohm. Table 13.3-4 is meant only to provide estimates of the size of excitation transients encountered; the exact level will depend upon the specific sensor configuration.

Equation 13.3-7 can be solved for the maximum lead length, L, permitted to maintain a specified error limit. Combining Equations 13.3-7 and 13.3-4 and solving for L gives:

$$L = -(R_o C_f + (t/\ln(V_e/V_{eo})))/R_o C_w \qquad [13.3\text{-}15]$$

where V_e is the measurement error limit.

EXAMPLE LEAD LENGTH CALCULATION FOR 107 TEMPERATURE SENSOR

Assume a limit of 0.05°C over a 0°C to +40°C range is established for the transient settling error. This limit is a reasonable choice since it approximates the linearization error over that range. The output signal from the thermistor bridge varies nonlinearly with temperature ranging from about 100 µV/°C at 0°C to 50 µV/°C at 40°C. Taking the most conservative figure yields an error limit of V_e = 2.5 µV. The other values needed to calculate the maximum lead length are summarized in Table 13.3-5 and listed below:

1) V_{eo} ~ 50 mV, peak transient at 2 V excitation
2) V_e ~ 2.5 µV, allowable measurement error
3) t = 450 µs, CR510 input settling time
4) R_o = 1 kohm, 107 probe source resistance
5) C_f = 3.3 nfd, CR510 input capacitance
6) C_w ~ 42 pfd/ft., lead wire capacitance

Solving Equation 13.3-15 gives a maximum lead length of:

L ~ 1003 ft., error ~ 0.05°C

Setting the allowable error at 0.1°C or approximately 5 µV, the maximum lead length increases to:

L ~ 1085 ft., error ~ 0.1°C

13.3.4 SUMMARY OF SETTLING ERRORS FOR CAMPBELL SCIENTIFIC RESISTIVE SENSORS

Table 13.3-5 summarizes the data required to estimate the effect of lead length on settling errors for Campbell Scientific's resistive sensors. Comparing the transient level, V_{eo}, to the input range, one suspects that transient errors are the most likely limitation for the 107 sensor. The sensors in the WVU-7 are the same as in the Model 107 (the lead wire is different), but the signal leads for the WVU-7 wet- and dry-bulbs are not subject to excitation transients because they are shielded independently from the excitation.

TABLE 13.3-5. Summary of Input Settling Data For Campbell Scientific Resistive Sensors

Sensor Model #	Belden Wire #	Ro (kohms)	Cw (pfd/ft.)	τ* (us)	Input Range(mV)	V_x(mV)	V_{eo}(mV)**
107	8641	1	42	45	7.5	2000	50
207(RH)	8771	1	41	44	250	1500	85
WVU-7	8723	1	62	65	7.5	2000	0
227	8641	0.1-1	42	5-45	250	250	0
237	8641	1	42	45	25	2500	65
024A	8771	1-6	41	1-222	250	500	0-90

* Estimated time constants are for 1000 foot lead lengths and include 3.3nfd CR510 input capacitance.

** Measured peak transients for 1000 foot lead lengths at corresponding excitation, V_x.

TABLE 13.3-6. Maximum Lead Length vs. Error for Campbell Scientific Resistive Sensors

Sensor Model #	Error	Range	V_e(µV)	Maximum Length(ft.)
107	0.05°C	0°C to 40°C	5	1000[1]
207(RH)	1%RH	20% to 90%	250	2000[3]
WVU-7	0.05°C	0°C to 40°C	5	852[2]
024A	3o	@ 360°	2083	380[2]
227	-	-	-	2000[3]
237	10 kohm	20k to 300k	1000	2000[3]

[1] based on transient settling
[2] based on signal rise time
[3] limit of excitation drive

The comparatively small transient yet large source resistance of the 024A sensor indicates that signal rise time may be the most important limitation. The analysis in Section 13.3.2 confirms this.

The Model 227 Soil Moisture Block has a relatively short time constant and essentially no transient. Lead lengths in excess of 2000 feet produce less than a 0.1 bar (0-10 bar range) input settling error. With this sensor, the drive capability of the excitation channel limits the lead length. If the capacitive load 0.1 µfd and the resistive load is negligible, V_x will oscillate about its control point. If the capacitive load is 0.1 or less, V_x will settle to within 0.1% of its correct value 150 µs. A lead length of 2000 feet is permitted for the Model 227 before approaching the drive limitation.

Table 13.3-6 summarizes maximum lead lengths for corresponding error limits in six Campbell Scientific sensors. Since the first three sensors are nonlinear, the voltage error, V_e, is the most conservative value corresponding to the error over the range shown.

MINIMIZING SETTLING ERRORS IN NON-CAMPBELL SCIENTIFIC SENSORS

When long lead lengths are mandatory in sensors configured by the user, the following general practices can be used to minimize or measure settling errors:

1. When measurement speed is not a prime consideration, Instruction 4, Excite, Delay, and Measure, can be used to insure ample settling time for half bridge, single-ended sensors.

2. An additional low value bridge resistor can be added to decrease the source resistance, R_o. For example, assume a YSI nonlinear thermistor such as the model 44032 is used with a 30 kohm bridge resistor, R'_f. A typical configuration is shown in Figure 13.3-7A. The disadvantage with this configuration is the high source resistance shown in column 3 of Table 13.3-7. Adding another 1 K resistor, R_f, as shown in Figure 13.3-7B, lowers the source resistance of the CR510 input. This offers no improvement over configuration A because R'_f still combines with the lead capacitance to slow the signal response at point P. The

source resistance at point P (column 5) is essentially the same as the input source resistance of configuration A. Moving R_f, out to the thermistor as shown in Figure 13.3-7C optimizes the signal settling time because it becomes a function of R_f and C_w only.

Columns 4 and 7 list the signal voltages as a function of temperature using a 2000 mV excitation for configurations A and C, respectively. Although configuration A has a higher output signal (2500 mV input range), it does not yield any higher resolution than configuration C which uses the ±250 mV input range.

NOTE: Since R_f' attenuates the signal in configuration B and C, one might consider eliminating it altogether. However, its inclusion "flattens" the non-linearity of the thermistor, allowing more accurate curve fitting over a broader temperature range.

3. Where possible, run excitation leads and signal leads in separate shields to minimize transients.

4. Avoid PVC-insulated conductors to minimize the effect of dielectric absorption on input settling time.

5. Use the CR510 to measure the input settling error associated with a given configuration. For example, assume long leads are required but the lead capacitance, Cw, is unknown. Configure Rf on a length of cable similar to the measurement. Leave the sensor end open as shown in Figure 13.3-8 and measure the result using the same instruction parameters to be used with the sensor. The measured deviation from 0V is the input settling error.

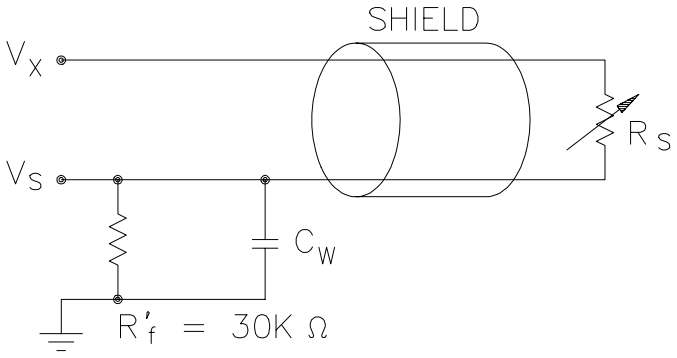
6. Most Campbell Scientific sensors are configured with a small bridge resistor, R_f, (typically 1 kohm) to minimize the source resistance. If the lead length of a Campbell Scientific sensor is extended by connecting to the pigtails directly, the effect of the lead resistance, R_l, on the signal must be considered. Figure 13.3-9 shows a Campbell Scientific Model 107 sensor with 500 feet of extension lead connected directly to the pigtails. Normally the signal voltage is proportional to $R_f/(R_s+R_b+R_f)$, but when the pigtails are extended, the signal is proportional to $(R_f+R_l)/(R_s+R_b+R_f+R_l)$. R_l is much smaller than the other terms in the denominator and can be discarded. The effect on the signal can be analyzed by taking the ratio of the signal with extended leads, V_{sl} to the normal signal, V_s:

$$V_{sl}/V_s = (R_f+R_l)/R_f$$

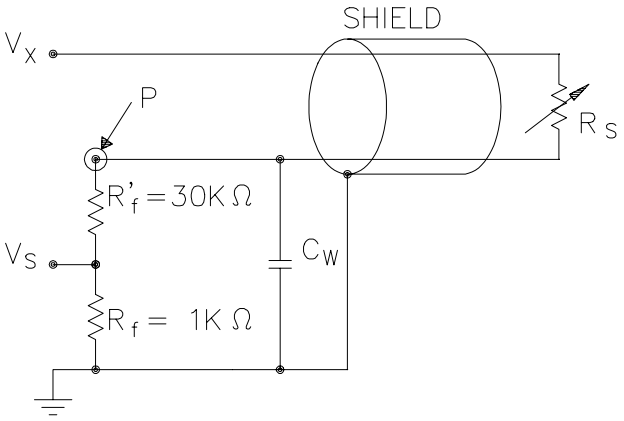
Plugging in values of R_f=1k and R_l=.012k (500' at 23 ohms/1000', Table 13.3-2) gives an approximate 1% error in the signal with extended leads. Converting the error to °C gives approximately a 0.33=°C error at 0°C, 0.53°C error at 20°C, and a 0.66°C error at 40°C. The error can be avoided by maintaining the pigtails on the CR510 end of the extended leads because R_l does not add to the bridge completion resistor, R_f, and its influence on the thermistor resistance is negligible.

TABLE 13.3-7. Source Resistances and Signal Levels for YSI #44032 Thermistor Configurations Shown in Figure 13.3-7 (2V Excitation)

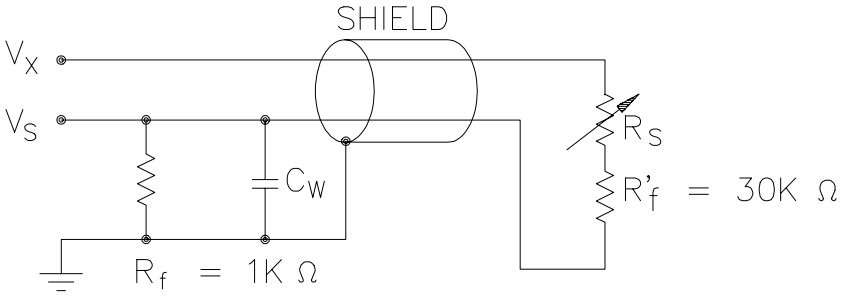
		--------A--------		-----B-----	-------C-------	
T	R_s (kohms)	R_o (kohms)	V_s(mV)	R_o@P (kohms)	R_o (kohms)	V_s(mV)
-40	884.6	29.0	66	30.0	1	2.2
-20	271.2	27	200	27.8	1	6.6
0	94.98	22.8	480	23.4	1	15.9
+25	30.00	15.0	1000	15.2	1	32.8
+40	16.15	10.5	1300	10.6	1	42.4
+60	7.60	6.1	1596	6.1	1	51.8



A) $R_O = R_S R'_f / (R_S + R'_f)$, $V_S = V_X R'_f / (R_S + R'_f)$



B) R_O @P = $R_S (R'_f + R_f) / (R_S + R'_f + R_f)$



C) $R_O = R_f$, $V_S = V_X R_f / (R_S + R'_f + R_f)$

FIGURE 13.3-7. Half Bridge Configuration for YSI #44032 Thermistor Connected to CR510 Showing: A) large source resistance, B) large source resistance at point P, and C) configuration optimized for input settling

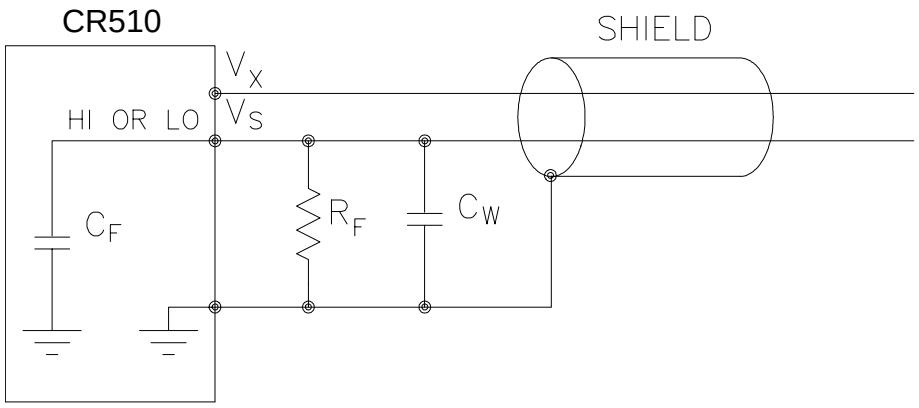


FIGURE 13.3-8. Measuring Input Settling Error with the CR510

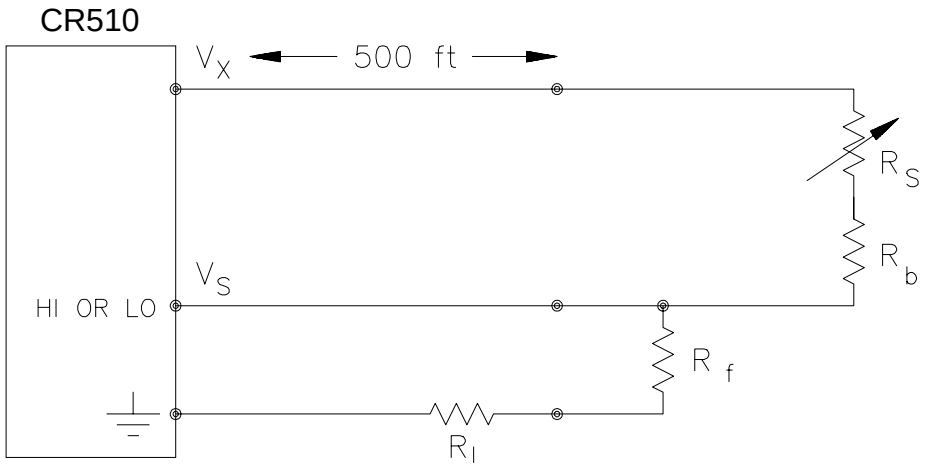


FIGURE 13.3-9. Incorrect Lead Wire Extension on Model 107 Temperature Sensor

13.4 BRIDGE RESISTANCE MEASUREMENTS

There are 6 bridge measurement instructions included in the CR510 software. Figure 13.4-1 shows the circuits that would typically be measured with these instructions. In the diagrams, the resistors labeled R_s would normally be the sensors and those labeled R_f would normally be fixed resistors. Circuits other than those diagrammed could be measured, provided the excitation and type of measurements were appropriate.

With the exception of Instructions 4 and 8, which apply an excitation voltage then wait a specified time before making a measurement, all of the bridge measurements make one set of measurements with the excitation as programmed and another set of measurements with the excitation polarity reversed. The error in the two measurements due to thermal emfs can then be accounted for in the processing of the measurement instruction. The excitation is switched on 450 µs before the integration portion of the measurement starts and is grounded as soon as the integration is completed. Figure 13.4-2 shows the excitation and measurement sequence for Instruction 6, a 4 wire full bridge. Excitation is applied separately for each phase of a bridge measurement. For example, in Instruction 6, as shown in Figure 13.4-2, excitation is switched on for the 4 integration periods and switched off between integrations.

Instruction 8 measurement sequence consists of applying a single excitation voltage, delaying a specified time, and making a differential voltage measurement. If a delay of 0 is specified, the inputs for the differential measurement are not switched for a second

integration as is normally the case (Section 13.2). The result stored is the voltage measured. Instruction 8 does not have as good resolution or common mode rejection as the ratiometric bridge measurement instructions. It does provide a very rapid means of making bridge measurements as well as supplying excitation to circuitry requiring differential measurements. This instruction does not reverse excitation. A 1 before the excitation channel number (1X) causes the channel to be incremented with each repetition.

The output of Instruction 8 is simply the voltage measurement. When 8 is used to measure a full bridge (same connections as Instruction 6 in Figure 13.5-1), the result is V_1 which equals V_x ($R_3/(R_3+R_4)$ - $R_2/(R_1+R_2)$). (In other words, to make the output the same as Instruction 6, use a factor of $1000/V_x$ in the multiplier.)

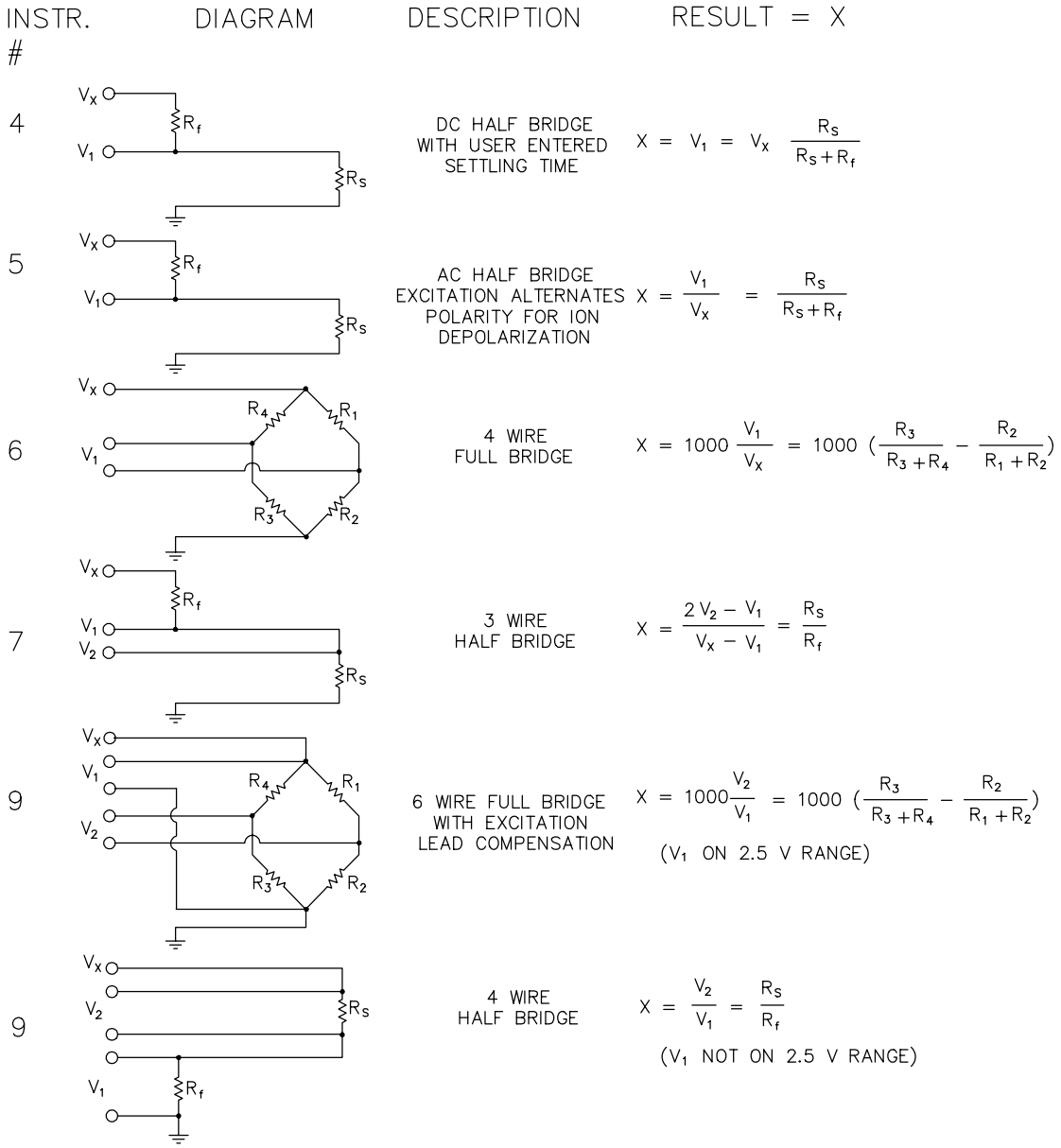


FIGURE 13.4-1. Circuits Used with Instructions 4-9

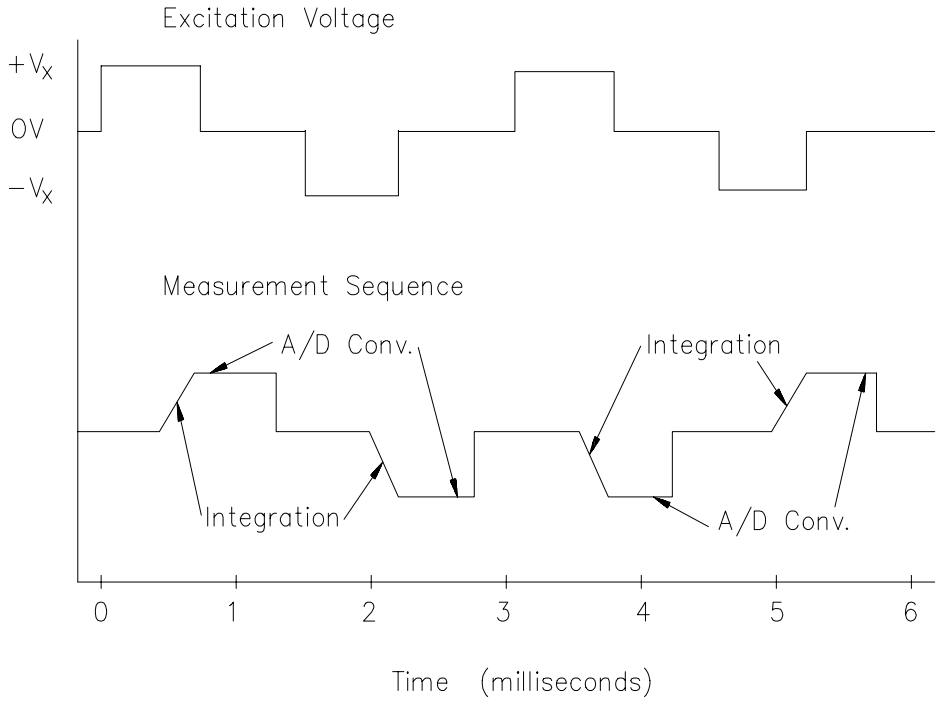


FIGURE 13.4-2. Excitation and Measurement Sequence for 4 Wire Full Bridge

TABLE 13.4-1. Comparison of Bridge Measurement Instructions

Instr. #	Circuit	Description
4	DC Half Bridge	The delay parameter allows the user entered settling time compensate for capacitance in long lead lengths. No polarity reversal. One single-ended measurement. Measured voltage output.
5	AC Half Bridge	Rapid reversal of excitation polarity for ion depolarization. One single-ended measurement at each excitation polarity. Ratiometric output.
6	4 Wire Full Bridge	One differential measurement at each excitation polarity. Ratiometric output.
7	3 Wire Half Bridge	Compensates for lead wire resistance, assuming resistance is same in both wires. Two single-ended measurements at each excitation polarity. Ratiometric output.
8	Differential Measurement with Excitation	Makes a differential measurement without reversing excitation polarity. Used for fast measurements on load cells, PRTs etc. Resolution and common mode rejection worse than 6 if used with delay = 0. Measured voltage output.
9	6 Wire Full Bridge or 4 Wire Half Bridge	Compensates for lead wire resistance. Two differential measurements at each excitation polarity. Ratiometric output.

Calculating the actual resistance of a sensor which is one of the legs of a resistive bridge usually requires the use of one or two Processing Instructions in addition to the bridge measurement instruction. Instruction 59 takes a value, X, in a specified input location and computes the value MX/(1-X), where M is the multiplier and stores the result in the original location. Instruction 42 computes the reciprocal of a value in an input location. Table 13.4-2 lists the instructions used to compute the resistance of any single resistor shown in the diagrams in Figure 13.4-1, provided the values of the other resistors in the bridge circuit are known.

TABLE 13.4-2. Calculating Resistance Values from Bridge Measurement

Instr.	Result	Instr.#	Multiplier;	Offset
4	$X = V_x(R_s/(R_s+R_f))$			
	$R_s = R_f \frac{X/V_x}{1-X/V_x}$	**4** **59**	$1/V_x$ R_f	0
	$R_f = \frac{1}{((X/V_x)/(1-X/V_x))/R_s}$	**4** **59** **42**	$1/V_x$ $1/R_s$	0
5	$X = R_s/(R_s+R_f)$			
	$R_s = R_f \frac{X}{1-X}$	**5** **59**	1 R_f	0
	$R_f = \frac{1}{(X/(1-X))/R_s}$	**5** **59** **42**	1 $1/R_s$	0
6,8,9*	$X = 1000\,[R_3/(R_3+R_4)-R_2/(R_1+R_2)]$	*used for full bridge		
	$R_1 = \frac{1}{(X_1/(1-X_1))/R_2}$	**6** or **9** **8** **6**,**8**, or **9** **59** **42**	-0.001; $1/V_x$ $1/R_2$	 $R_3/(R_3+R_4)$
	where $X_1 = -X/1000 + R_3/(R_3+R_4)$			
	$R_2 = R_1(X_2/(1-X_2))$ where $X_2 = X_1$	**6** or **9** **59**	-0.001 R_1	$R_3/(R_3+R_4)$
	$R_3 = R_4(X_3/(1-X_3))$ where $X_3 = X/1000 + R_2/(R_1+R_2)$	**6** or **9** **59**	0.001 R_4	$R_2/(R_1+R_2)$
	$R_4 = \frac{1}{(X_4/(1-X_4))/R_3}$	**6** or **9** **59** **42**	0.001 $1/R_3$	$R_2/(R_1+R_2)$
	where $X_4 = X_3$			
7&9*	$X = R_s/R_f$	* used as half bridge		
	$R_s = R_fX$	**7** or **9**	R_f	0
	$R_f = R_s/X$	**7** or **9** **42**	$1/R_s$	0

13.5 RESISTANCE MEASUREMENTS REQUIRING AC EXCITATION

Some resistive sensors require AC excitation. These include the 207 Relative Humidity Probe, soil moisture blocks, water conductivity sensors, and wetness sensing grids. The use of DC excitation with these sensors can result in polarization, which will cause an erroneous measurement, and may shift the calibration of the sensor and/or lead to its rapid decay.

The AC half bridge Instruction 5 (incorporated into the 247 conductivity probe) reverses excitation polarity to provide ion depolarization and, in order to minimize the time excitation is on, grounds the excitation as soon as the signal is integrated (Figure 13.5-1). The slow integration time should never be used with a sensor requiring AC excitation because it results in the excitation lasting about 1.5 times as long, allowing polarization to affect the measurement.

INFLUENCE OF GROUND LOOP ON MEASUREMENTS

When measuring soil moisture blocks or water conductivity, the potential exists for a ground loop which can adversely affect the measurement. This ground loop arises because the soil and water provide an alternate path for the excitation to return to CR510 ground, and can be represented by the model diagrammed in Figure 13.5-2.

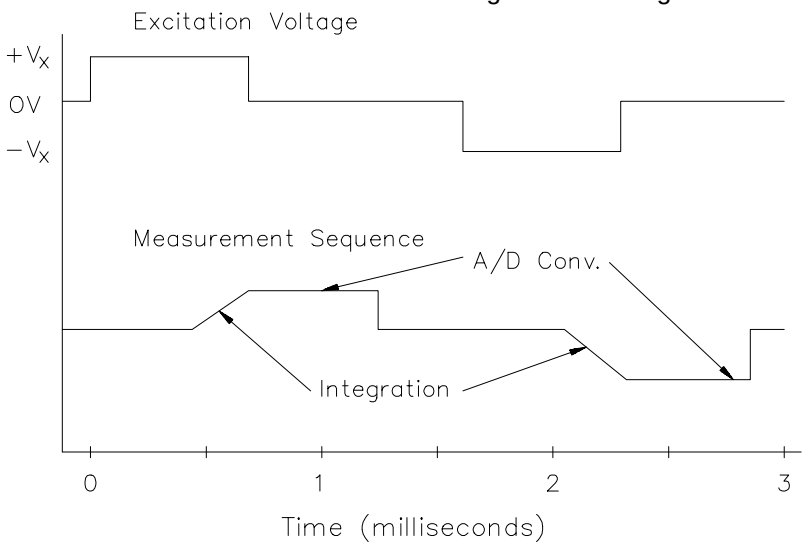


FIGURE 13.5-1. AC Excitation and Measurement Sequence for AC Half Bridge

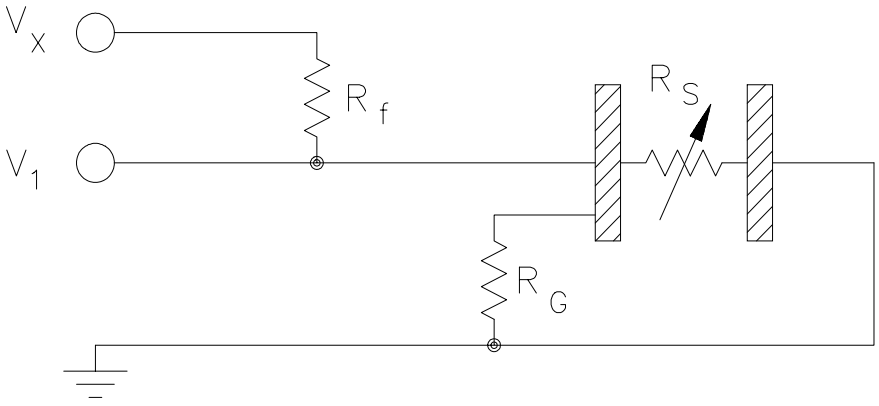


FIGURE 13.5-2. Model of Resistive Sensor with Ground Loop

In Figure 13.5-2, V_X is the excitation voltage, R_f is a fixed resistor, R_S is the sensor resistance, and R_G is the resistance between the excited electrode and CR510 earth ground. With R_G in the network, the measured signal is:

$$V_1 = V_x \frac{R_S}{(R_S+R_f) + R_SR_f/R_G} \quad [13.6\text{-}1]$$

R_SR_f/R_G is the source of error due to the ground loop. When R_G is large the equation reduces to the ideal. The geometry of the electrodes has a great effect on the magnitude of this error. The Delmhorst gypsum block used in the 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 CR510 earth ground form a galvanic cell, with the water/soil solution acting as the electrolyte. If current was allowed to flow, the resulting oxidation or reduction would soon damage the electrode, just as if DC excitation was used to make the measurement. Campbell Scientific probes are built with series capacitors in the leads to block this DC current. In addition to preventing sensor deterioration, the capacitors block any DC component from affecting the measurement.

13.6 CALIBRATION PROCESS

The CR510 makes voltage measurements by integrating the input signal for a fixed time and then holding the integrated value for the analog to digital (A/D) conversion. The A/D conversion is made by a 13 bit approximation using a digital to analog converter (DAC). The result from the approximation is DAC counts, which are multiplied by coefficients to obtain millivolts (mV). There are 10 calibration coefficients, one for each of the 5 gain ranges for the fast and slow integration times.

The CR510 has an internal calibration function that feeds positive and negative voltages through the amplifiers and integrator and calculates new calibration coefficients. By adjusting the calibration coefficients the accuracy of the voltage measurements is maintained over the -25 to +50°C operating range of the CR510. Calibration is executed under four conditions:

1. When the CR510 is powered up.
2. Automatically when Instruction 24 is not contained in a program table.
3. When the watchdog resets the processor.
4. When the calibration instruction, Instruction 24, is executed.

AUTOMATIC CALIBRATION SEQUENCE

The primary advantage of automatic calibration is that the CR510 is constantly calibrated without user programming. The CR510 defaults to automatic calibration when Instruction 24 is not contained in a program table.

Every 8 seconds one part of a 22 part calibration sequence is performed. Program execution is interrupted (5.4 - 21.4 ms), when necessary, for each part of the calibration. Every 2.9 minutes (8 seconds * 22) ten calibration coefficients are calculated. The calculated coefficients are multiplied by 1/5, and then added to 4/5 times the existing coefficients. Averaging is done as a safeguard against coefficients calculated from a noisy measurement.

The above weighting of the newly calculated coefficients results in a 15 minute time constant (see Instruction 58) in the response of the calibration to step changes affecting the calibration coefficients (primarily temperature). For most environmental applications a 15 minute time constant is acceptable. The automatic calibration may result in the calibration coefficients not being optimum for applications that subject the CR510 to extreme temperature gradients.

Automatic calibration extends the processing time 5.4 to 21.4 ms when it is executed (every 8 seconds). If the processing time exceeds the execution interval the CR510 finishes processing the table and awaits the next occurrence of the execution interval before initiating the table. At the fastest execution interval of 1/64 (0.0156) second the program

table WILL be overrun by the automatic calibration. If an overrun occurs every time calibration is executed, then 1 execution is skipped for every 512 times that the program table is executed. If the measurements are being averaged, the effect of the overrun is negligible. Program table overruns are indicated by the appearance of two decimals on either side of the sixth digit on the CR10KD and are also stored in memory (Section 1.7).

INSTRUCTION 24 CALIBRATION

The alternative to automatic calibration is the use of Instruction 24, the calibration instruction. Instruction 24 implements a complete calibration which occurs ONLY when EXECUTED by a program table. Instruction 24 calibration is the average of 10 calibrations, and takes approximately 2.8 seconds to complete. Automatic calibration is disabled when a program is compiled that contains Instruction 24.

Instruction 24 calibration, as opposed to automatic calibrations, may be advantageous in applications where: 1) the CR510 is exposed to extreme thermal gradients, or 2) automatic calibration would interfere with the desired sampling rate, and the ambient temperature is stable enough to allow calibration at specific points during program execution.

Calibration coefficients are replaced each time that Instruction 24 is executed. Unlike automatic calibration, there is no time constant for the coefficients to respond in changes to calibration. Instruction 24 calibration ensures that the coefficients are optimum at the time that the instruction is executed. For example, consider a CR510 mounted under the dash of an automobile, where temperature could easily change 50 degrees. Temperature changes affect the measurement circuitry which must be compensated for by calculating new coefficients. Each time Instruction 24 is executed a new set of calibration coefficients is calculated based on the measurements made at that time.

Calibration at a certain point during program execution may be advantageous for some applications. For example, suppose Table 2 has an execution time of 15.6 ms, but only executes when flag 1 is set. Table 1 has a 5 minute execution time which makes a temperature measurement, and sets flag 1 if the temperature exceeds a fixed value. To prevent overrun errors which would occur in Table 2 if the automatic calibration was used, Instruction 24 could be executed before the temperature measurement was made by Table 1.

Instruction 24 also has an option to store the results of the automatic calibration in Input Storage. This can be used to detect hardware problems. If -99999 appears in any of the 19 input locations, the CR510 has a hardware problem or needs factory calibration.

SECTION 14. INSTALLATION AND MAINTENANCE

14.1 PROTECTION FROM THE ENVIRONMENT

The normal environmental variables of concern are temperature and moisture. The standard CR510 is designed to operate reliably from -25° to +50°C (-55° to +85°C, optional). Internal moisture damage is reduced with a water resistant conformal coating on the current board. Extra desiccant should also be placed in the enclosure to prevent corrosion.

Campbell Scientific offers fiberglass enclosures for housing a CR510 and peripherals. The enclosures are classified as NEMA 4X (water-tight, dust-tight, corrosion-resistant, indoor and outdoor use). A 1.25" diameter entry/exit port is located at the bottom of the enclosure for routing cables and wires. The enclosure door can be fastened with the hasp for easy access, or with the two supplied screws for more permanent applications. The white plastic inserts at the corners of the enclosure must be removed to insert the screws. The enclosures are white for reflecting solar radiation, thus reducing the internal enclosure temperature.

The Model ENC 12/14 fiberglass enclosure houses the CR510, power supply, and one or more peripherals. Inside dimensions of the ENC 12/14 are 14" x 12" x 5.5", outside dimensions are 18" x 13.5" x 8.13" (with brackets); weight is 11.16 lbs.

The Model ENC 16/18 fiberglass enclosure houses the CR510, power supply, and several peripherals. Inside dimensions of the ENC 16/18 are 16" x 18" x 9", outside dimensions are 21.75" x 20.0" x 11.0" (with brackets); weight is 17.2 lbs.

14.2 POWER REQUIREMENTS

The CR510 operates at a nominal 12 VDC. Below 9.6 or above 18 volts the CR510 does not operate properly.

The CR510 is diode protected against accidental reversal of the positive and ground leads from the battery. Prolonged input voltages in excess of 18 V may damage the CR510 and/or power supply. A transzorb provides transient protection by limiting voltage at approximately 20 V.

CAUTION: The metal surfaces of the Terminal Strip and mounting bracket are at the same electrical ground as the power supply. Caution must be exercised when connecting power directly to the Terminal Strip's 12 V and ground terminals. Connect the plus (+) side of the supply first, keeping the minus (-) side away from the Terminal Strip. Once the plus side is secured, connect the power return.

TABLE 14.2-1. Typical Current Drain for Common CR510 Peripherals

	Typical Current Drain (mA)	
Peripheral	**Quiescent**	**Active**
RF100 VHF 5 Watt Radio	30	1730
RF200 UHF 4 Watt Radio	30	1530
RF95 RF Modem	1.4	30
SM192/SM716 Storage Module	0.25	3
CSM1 Card Storage Module	0.5	18
CR10KD Keyboard/Display	<1	0.3 (*6 Mode)
DC112 Phone Modem	<1	45
COM200 Phone Modem	0.0012	140
COM100 Cellular Phone	0.5	1.8
VS1 Voice Phone Modem	0.05	75 (Data)/110 (Voice)
MD9 Multidrop Interface	1.2	17 (80 if drives network)
SRM-6A RS232 DCE Interface	2.2	15

System operating time for the batteries can be determined by dividing the battery capacity (amp-hours) by the average system current drain. The CR510 draws <1 mA in the quiescent state, 13 mA while processing, and 46 mA during an analog measurement; the length of operating time for each datalogger instruction is listed in the programming section. Typical current requirements for common CR510 peripherals are given in Table 14.2-1.

14.3 CAMPBELL SCIENTIFIC POWER SUPPLIES

CR510 Power Supplies are available from Campbell Scientific with either alkaline or lead acid batteries, the BPALK and PS12LA, respectively. The PS512M is also a lead acid supply with two 9-pin null modem ports for communication modems, see Section 14.3.3. The BPALK has 8 D cell alkaline batteries, the PS12LA has a rechargeable lead acid battery. The alkaline batteries are discarded after use. The lead acid batteries should be float charged with either AC power or a solar panel. The lead acid battery supplies power during a power failure or in times of low charge with a solar panel.

The CH12R and CH512R contain the same circuitry as the PS12LA and PS512M, respectively. They are used to float charge an external 12 VDC Yuasa battery using AC or solar power. No internal batteries are contained in the CH12R and CH512R. Their operation, however, is identical to that of the PS12LA and PS512M. Other power supply options are connecting a 12 volt battery directly to the CR510, Section 14.5, or supplying power from a vehicle, Section 14.6.

The PS12LA Power Supply provides 12 volts, regulates incoming AC or DC power, limits current from the battery, and provides circuitry to connect an external 12 volt battery.

Each of the power supplies has a thermal fuse in the power circuit that limits source current. If excessive current is drawn, the fuse gets hot, increases in resistance, and limits current. When the problem is fixed, the fuse cools and the resistance decreases, eventually allowing current to pass. When excessive current is drawn due to shorting the power leads to the Terminal Strip, allow 10 to 15 seconds for the fuse to cool before connecting power.

14.3.1 BPALK ALKALINE POWER SUPPLY

The BPALK uses 8 alkaline D cells and comes with an AA cell battery pack to supply power while replacing the D cells. The 4 AA batteries are not included (see Figure 14.3-1).

To replace the batteries without stopping the datalogger program, 1) connect the external battery to the port labeled temporary, 2) remove the old batteries, 3) replace with new alkaline D cell batteries, and 4) remove the external battery.

A fresh set of eight alkaline D cells has 12.4 volts and a nominal rating of 7.5 amp-hours at 20°C. The amp-hour rating decreases with temperature as shown in Table 14.3-1. Datalogger Instruction 10 can be used to monitor battery voltage. Replace the alkaline cells before the CR510 battery voltage drops below 9.6 V.

FIGURE 14.3-1. BPALK Power Supply

TABLE 14.3-1. Typical Alkaline Battery Service and Temperature

Temperature (°C)	% of 20°C Service
20 - 50	100
15	98
10	94
5	90
0	86
-10	70
-20	50
-30	30

NOTE: This data is based on one "D" cell under conditions of 50 mA current drain with a 30 ohm load. As the current drain decreases, the percent service improves for a given temperature.

14.3.2 PS12LA LEAD ACID POWER SUPPLY

The PS12LA power supply includes a 12 V, 7.0 amp-hour lead acid battery, a AC transformer (18 V), and a temperature compensated charging circuit with a charge indicating diode. An AC transformer or solar panel should be connected to the PS12 at all times. The charging source powers the CR510 while float charging the lead acid batteries. The internal lead acid battery powers the datalogger if the charging source is interrupted. The PS12LA specifications are given in Table 14.3-2.

The two leads from the charging source can be inserted into either of the CHG ports, polarity doesn't matter. A transzorb provides transient protection to the charging circuit. A sustained input voltage in excess of 40 V will cause the transzorb to limit voltage.

Some solar panels are supplied with a connector, this connector must clipped off so the two wires can be inserted into the two terminal ports. It is recommended that these two leads be stripped and tinned.

The red light (LED) on the PS12LA is on when a charging source is connected to the PS12LA CHG ports. The switch turns power on and off to the 12 V ports, battery charging still occurs when the switch is off.

CAUTION: Switch the power to "off" before disconnecting or connecting the power leads to the Terminal Strip. The Terminal Strip and PS12LA are at power ground. If 12 V is shorted to either of these, excessive current will be drawn until the thermal fuse opens.

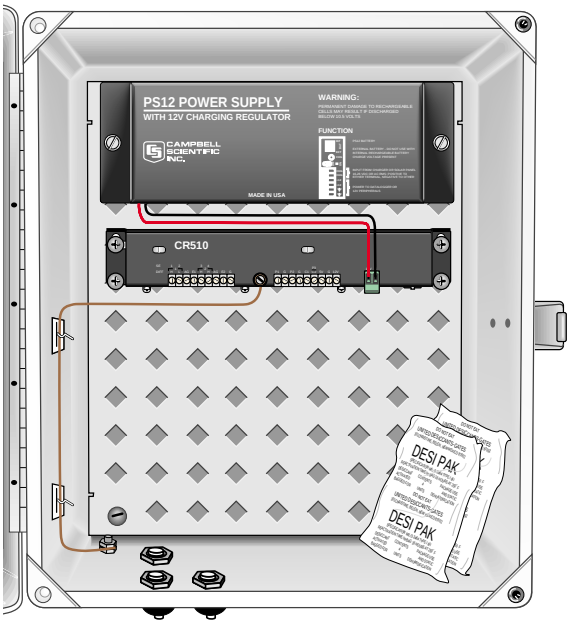


FIGURE 14.3-2 PS12LA

The external port, labeled EXT, is not meant to be used with the PS12LA unless the internal battery is removed. The primary power source is the charging source, and the secondary power source is the internal lead acid battery. Connecting a lead acid battery to the external source is the same as connecting two lead acid batteries in parallel, causing one battery to drop voltage and the other to raise voltage. Alkaline batteries connected to the external port must have a diode in series to block charging which would cause an explosion. (The PS12ALK battery pack has this diode.)

Monitor the power supply using datalogger Instruction 10. Users are strongly advised to incorporate this instruction into their data acquisition programs to keep track of the state of the power supply. If the system voltage level consistently decreases through time, some element(s) of the charging system has failed. Instruction 10 measures the voltage at the 12 V port, not the voltage of the lead acid battery. External power sources must be disconnected from the CR510 and charging circuit in order to measure the actual lead acid battery voltage.

TABLE 14.3-2. PS12LA, Battery, and AC Transformer Specifications

PS12LA	
Input Voltage	16 to 26 VDC or 16 to 26 VAC RMS
Output	Temperature compensated float charge
Maximum Battery Capacity	24 Ahr
Temperature range with NA 7-2 battery	-25 to +50°C
Lead Acid Battery	
Battery Type	Yuasa NA 7-12
Float Life @ 25°C	5 years typical
Capacity	7.0 amp-hour
Shelf Life, full charge	Check twice yearly
Charge Time (AC Source)	40 hr full charge, 20 hr 95% charge
AC Transformer	
Input:	120 VAC, 60 Hz
Isolated Output:	18 VAC @ 1.11 A max.

There are inherent hazards associated with the use of sealed lead acid batteries. Under normal operation, lead acid batteries generate a small amount of hydrogen gas. This gaseous by-product is generally insignificant because the hydrogen dissipates naturally before build-up to an explosive level (4%) occurs. However, if the batteries are shorted or overcharging takes place, hydrogen gas may be generated at a rate sufficient to create a hazard. Campbell Scientific makes the following recommendations:

1. A CR510 equipped with standard lead acid batteries should NEVER be used in applications requiring intrinsically safe equipment.
2. A lead acid battery should not be housed in a gas-tight enclosure.

14.3.3 PS512M VOLTAGE REGULATOR WITH NULL MODEM PORTS

The PS512M 12 Volt Lead Acid Power Supply with Charging Regulator and Null Modem Ports is used when 5 volts is needed to power external modems besides the capabilities of the PS12LA. The PS512M supplies 5 volts to pin 1 of the 9 pin null modem ports, otherwise the capabilities and functions are identical to the

PS12LA. A common use for the PS512M is in radiotelemetry networks. The PS12LA cannot be modified to a PS512M.

The maximum current drain on the 5 volt supply of the PS512M is 150 milliamps.

14.4 SOLAR PANELS

Auxiliary photovoltaic power sources may be used to maintain charge on lead acid batteries.

When selecting a solar panel, a rule-of-thumb is that on a stormy overcast day the panel should provide enough charge to meet the system current drain (assume 10% of average annual global radiation, kW/m^2). Specific site information, if available, could strongly influence the solar panel selection. For example, local effects such as mountain shadows, fog from valley inversion, snow, ice, leaves, birds, etc. shading the panel should be considered.

Guidelines are available from the Solarex Corporation for solar panel selection called "DESIGN AIDS FOR SMALL PV POWER SYSTEMS". It provides a method for calculating solar panel size based on general site location and system power requirements. If you need help in determining your system power requirements contact Campbell Scientific's Marketing Department.

14.5 DIRECT BATTERY CONNECTION TO THE CR510 TERMINAL STRIP

For some applications, size restrictions or other operational considerations may preclude the use of Campbell Scientific power supply options. In these cases the power supply may be connected directly to the terminal strip. Any 9.6 to 18 VDC supply may be connected to the 12 V and G terminals on the terminal strip. The metal surfaces of the terminal strip and mounting bracket are at power ground. Make connections to the terminal strip first and then to the power supply. If the power supply must be connected first, connect the positive to the terminal strip before the ground to avoid shorting to the terminal strip or mounting bracket.

14.6 VEHICLE POWER SUPPLY CONNECTIONS

If a CR510 is to be powered from the 12 Volts of a motor vehicle, a second 12 V supply is also required at the time of vehicle start-up. When the starting motor of a vehicle with a 12 V electrical system is engaged, the voltage drops considerably below the nominal 12 V, which would cause the CR510 to malfunction every time the vehicle is started. The second 12 V supply prevents this malfunction. Figure 14.6-1 shows the general case for connecting the two supplies to the Terminal Strip. The diode allows the vehicle to power the CR510 without the second supply attempting to power the vehicle. To reduce the potential for ground reference errors in measurements, the ground lead should be 16 AWG or larger.

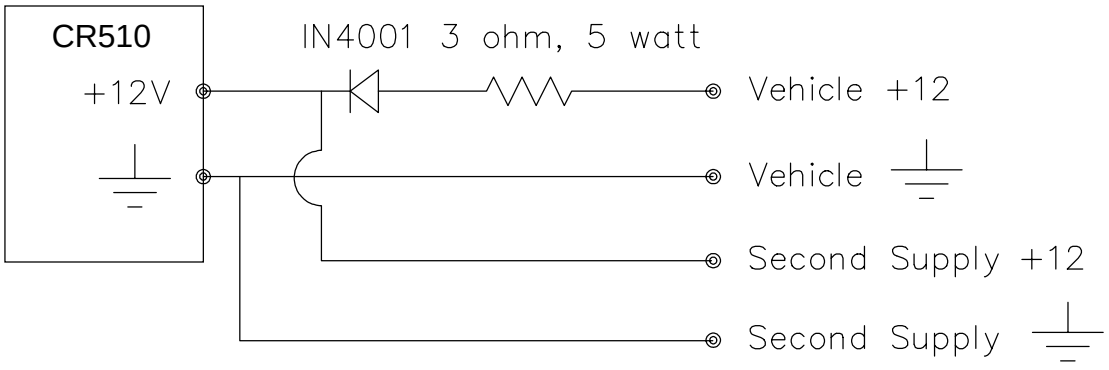


FIGURE 14.6-1. Connecting to Vehicle Power Supply

14.7 GROUNDING

14.7.1 PROTECTION FROM LIGHTNING

Primary lightning strikes are those where lightning hits the datalogger or sensors directly. Secondary strikes occur when the lightning strikes somewhere near the system and induces a voltage in the wires. The purpose of an earth ground is to minimize damage to the system by providing a low resistance path around the system to a point of low potential. Campbell Scientific recommends that all dataloggers in use in the field be earth grounded. All components of the system (datalogger, sensors, external power supplies, mounts, housings, etc.) should be referenced to one common earth ground.

Every terminal on the Terminal Strip, with the exception of ground (G) and analog ground (AG) terminals are spark gapped. The spark gaps will fire at 150 V and the current will be diverted to ground. As shown in Figure 14.7-1, the power ground and analog ground are independent lines until joined inside the CR510. The fuse shown in Figure 14.7-1 (located on the underside of the Terminal Strip) is a 30 AWG wire, equivalent to a conventional 5 Amp fuse. It will blow if a sufficient transient comes in on the G or AG lines, at which time the current is directed away from the CR510 through the diodes. The fuse may be replaced by soldering another 30 AWG wire to the soldering pads provided.

A modem/phone line connected to the Terminal Strip provides another path for transients to enter and damage the CR510. Campbell Scientific's COM200 phone modem has spark gaps on the phone lines. A 12 AWG wire should be run from the modem ground terminal to the earth ground. Additional protection is provided by the ground (Pin 2) of the 9 pin Serial I/O which is tied to power ground on the Terminal Strip.

The transient protection designed into Campbell Scientific's equipment is meaningless if a good system earth ground is not provided. It is the users responsibility to provide this earth ground.

In laboratory applications, locating a stable earth ground is not always obvious. In older buildings, new cover plates on old AC sockets may indicate that a safety ground exists when in fact the socket is not grounded. If a safety ground does exist, it is good practice to verify that it carries no current. If the integrity of the AC power ground is in doubt, also ground the system through the buildings, plumbing or another connection to earth ground.

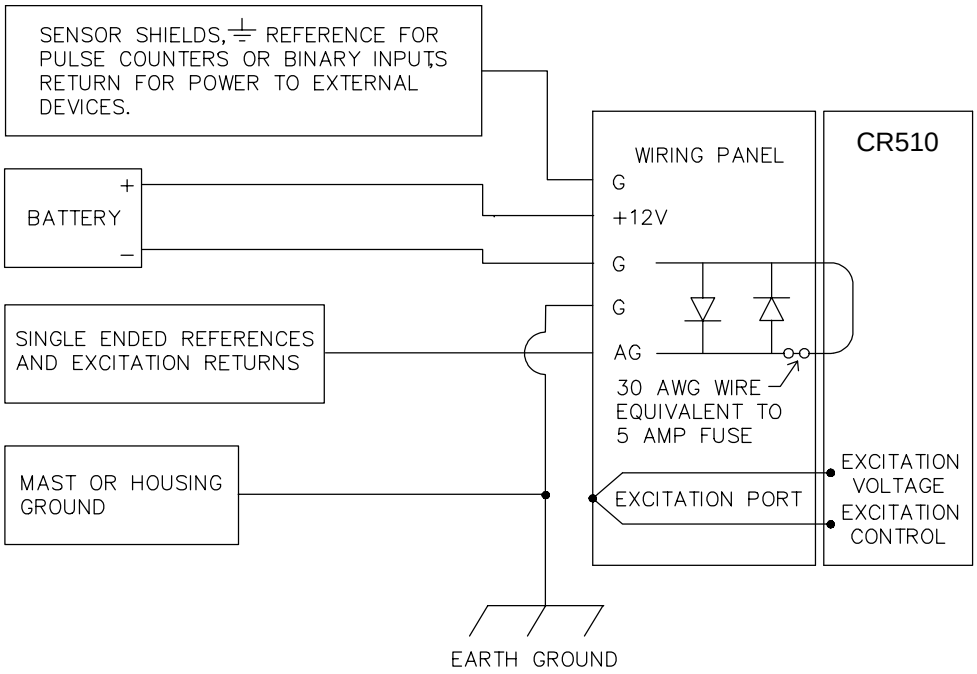


FIGURE 14.7-1. Terminal Strip Grounding Diagram and Excitation Control

In the field, an earth ground may be created through a grounding rod. A 12 AWG or larger wire should be run between a Terminal Strip power ground (G) terminal and the earth ground. Campbell Scientific's CM10 and CM6 Tripods and UT3 Tower come complete with ground and lightning rods, grounding wires, and appropriate ground wire clamps.

14.7.2 EFFECT OF GROUNDING ON MEASUREMENTS: COMMON MODE RANGE

The common mode range is the voltage range, relative to the CR510 ground, within which both inputs of a differential measurement must lie in order for the differential measurement to be made. Common mode range for the CR510 is ±2.5 V. For example, if the high side of a differential input is at 2 V and the low side is at 0.5 V relative to CR510 ground, a measurement made on the ±2.5 V range would indicate a signal of 1.5 V. However, if the high input changed to 3 V, the common mode range is exceeded and the measurement cannot be made.

Common mode range may be exceeded when the CR510 is measuring the output from a sensor which has its own grounded power supply and the low side of the signal is referenced to power ground. If the CR510 ground and the sensor ground are at sufficiently different potentials, the signal will exceed the common mode range. To solve this problem, the sensor power ground and the CR510 ground should be connected, creating one ground for the system.

In a laboratory application, where more than one AC socket may be used to power various sensors, it is not always safe to assume that the power grounds are at the same potential. To be safe, the ground of all the AC sockets in use should be tied together with a 12 AWG wire.

14.8 TERMINAL STRIP

The Terminal Strip provides transient protection, improves excitation voltage accuracy, and makes convenient, positive connections to power, sensors, and peripherals (refer to Figure 14.7-1). Terminal Strip transient protection is discussed in Section 14.7.

The Terminal Strip carries two lines between the CR510 and each excitation port. One line is for excitation voltage, the other is for feedback control of the voltage. The feedback line is required to compensate for line losses between the CR510 and the excitation port on the Terminal Strip (see Figure 14.7-1).

A 5 V output terminal is available on the Terminal Strip for powering 5 V peripherals. The 5 V ports can source up to 200 mA. An input protection transzorb will divert current to ground at approximately 10 V.

A functional description of the 37 pin connector located on the CR510 is provided in Appendix D.

14.9 USE OF DIGITAL I/O PORT (C1) FOR SWITCHING RELAYS

The I/O port (C1) can be configured as an output port and set low or high (0 V low, 5 V high) using I/O Instruction 20, Port Set, or commands 41 associated with 41, 51, 61 Program Control Instructions 83 through 93. A digital output port is normally used to operate an external relay driver circuit because the port itself has a limited drive capability (1.5 mA at 3.5 V).

Figure 14.10-1 shows a typical relay driver circuit in conjunction with a coil driven relay which may be used to switch external power to some device. In this example, when the control port is set high, 12 V from the datalogger passes through the relay coil, closing the relay which completes the power circuit to a fan, turning the fan on.

In other applications it may be desirable to simply switch power to a device without going through a relay. Figure 14.10-2 illustrates a circuit for switching external power to a device without going through a relay. If the peripheral to be powered draws in excess of 75 mA at room temperature (limit of the 2N2907A medium power transistor), the use of a relay (Figure 14.10-1) would be required.

Other control port activated circuits are possible for applications with greater current/voltage demands than shown in Figures 14.10-1 and 2. For more information contact Campbell Scientific's Marketing Department.

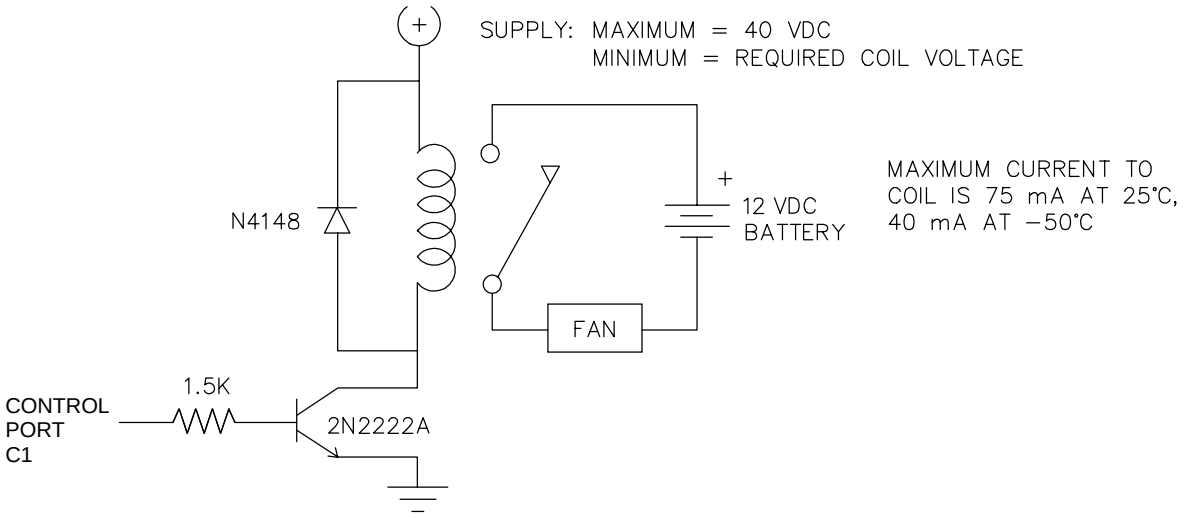


FIGURE 14.9-1. Relay Driver Circuit with Relay

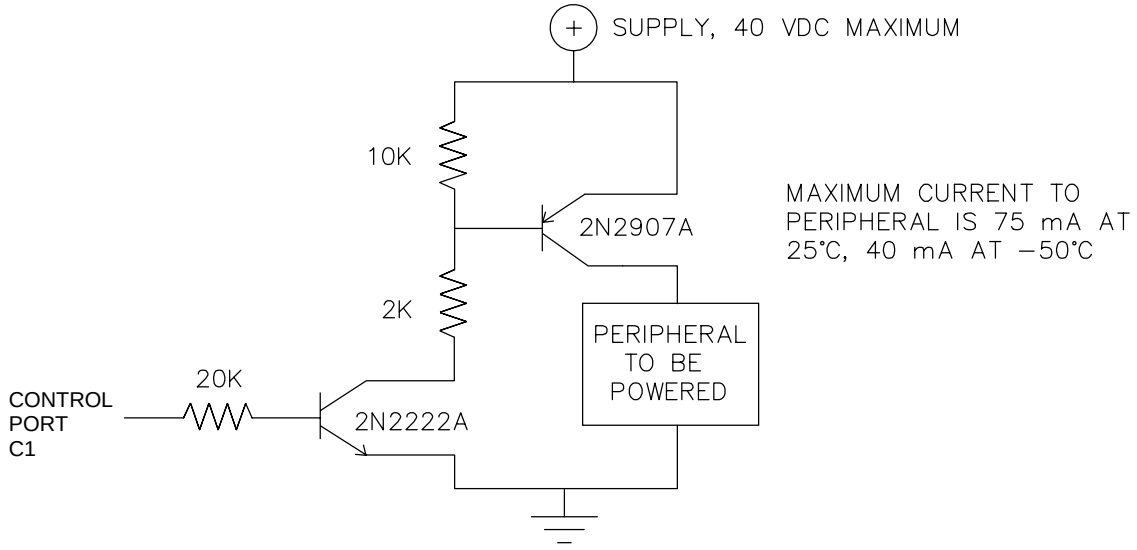


FIGURE 14.9-2. Power Switching without Relay

14.10 MAINTENANCE

The CR510 Terminal Strip and power supplies require a minimum of routine maintenance.

When not in use, the PS12LA should be stored in a cool, dry environment with the AC charging circuit activated.

The BPALK alkaline supply should not drop below 9.6 V before replacement. When not in use, remove the eight cells to eliminate potential corrosion of contact points and store in a cool dry place.

14.10.1 DESICCANT

To prevent corrosion, desiccant must be placed inside the enclosure. If only alkaline batteries are inside the enclosure, the sensor lead entrance may be plugged to inhibit vapor transfer into the enclosure. Do not plug the entrance if lead acid batteries are present. Hydrogen gas generated by the batteries may build up to an explosive concentration.

14.10.2 REPLACING THE INTERNAL BATTERY

The CR510 contains a lithium coin cell battery that operates the clock and SRAM when the CR510 is not connected to an external power source. The CR510 does not draw any power from the lithium battery while it is powered externally. In a CR510 stored at room temperature, the lithium battery should last approximately 4 years (less at temperature extremes). Where the CR510 is powered most or all of the time the lithium cell should last much longer.

While powered from an external source, the CR510 measures the voltage of the lithium battery daily. This voltage is displayed in the ∗B Mode window 8 (Section 1.6) and in response to the telecommunications status command (Section 5). A new battery will have approximately 3 volts. The battery should be replaced when the voltage drops below 2.4 volts. If the lithium cell is removed or allowed to discharge below the safe level, the CR510 will still operate correctly while powered. Without the lithium battery, the clock will reset and data will be lost when power is removed.

A replacement lithium battery can be purchased from Campbell Scientific or from an Electronics store (e.g., Radio Shack). Table 14.10-1 lists the specifications of the battery.

Table 14.10-1 CR510 Lithium Battery Specifications

Model	CR2430
Capacity	270 mAhr
self discharge rate	1% of capacity/yr.
Diameter	24.5 mm
Thickness	3.0 mm
Temperature range	
Operating	-20 to 60 °C
Storage	-40 to 60 °C

The CR510 module must be disassembled to replace the lithium cell. Disconnect the power and remove the back plate to expose the coin cell battery.

The coin cell is held in place by a spring clamp. It can be removed by grabbing the edge of the cell with your fingers or by inverting the circuit board and lifting the spring clip with a fingernail until the cell falls out.

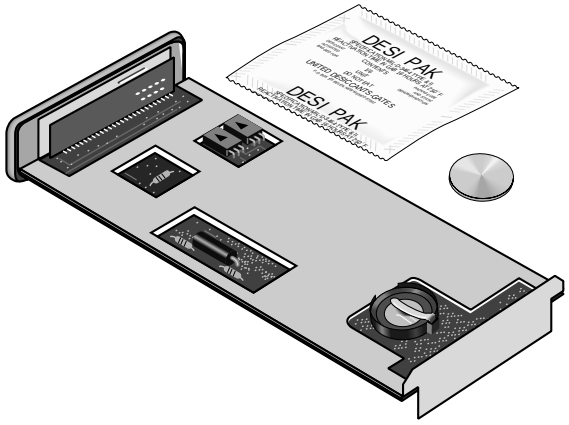


FIGURE 14.10-1. CR510 Lithium Battery Location

The new cell is slipped into place under the clip, the negative side toward the circuit board and the positive side touching the clip. Replace the desiccant taped to the circuit board holder with a fresh packet before reassembling the CR510.

APPENDIX A. GLOSSARY

ASCII: Abbreviation for American Standard Code for Information Interchange (pronounced "askee"). A specific binary code of 128 characters represented by 7 bit binary numbers.

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 (see Synchronous).

BAUD RATE: The speed of transmission of information across a serial interface, expressed in units of bits per second. For example, 9600 baud refers to bits being transmitted (or received) from one piece of equipment to another at a rate of 9600 bits per second. Thus, a 7 bit ASCII character plus parity bit plus 1 stop bit (total 9 bits) would be transmitted in 9/9600 sec. = .94 ms or about 1000 characters/sec. When communicating via a serial interface, the baud rate settings of two pieces of equipment must match each other.

DATA POINT: A data value which is sent to Final Storage as the result of an Output Instruction. Strings of data points output at the same time make up Output Arrays.

EXECUTION INTERVAL: The time interval between initiating each execution of a given Program Table. If the Execution Interval is evenly divisible into 24 hours (86,400 seconds), the Execution Interval will be synchronized with 24 hour time, so that the table is executed at midnight and every execution interval thereafter. The table will be executed for the first time at the first occurrence of the Execution Interval after compilation. If the Execution Interval does not divide evenly into 24 hours, execution will start on the first even second after compilation. See Section OV4.3.1 for information on the choice of an Execution Interval.

EXECUTION TIME: The time required to execute an instruction or group of instructions. If the execution time of a Program Table exceeds the table's Execution Interval, the Program Table will be executed less frequently than programmed (Section OV4.3.1 and 8.9).

FINAL STORAGE: That portion of memory allocated for storing Output Arrays. Final Storage may be viewed as a ring memory, with the newest data being written over the oldest. Data in Final Storage may be displayed using the *7 Mode or sent to various peripherals (Sections 2, 3, and OV4.1).

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

HANDSHAKE, HANDSHAKING: 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 CR510 is primarily used to detect the presence or absence of peripherals such as the Storage Module.

HIGH RESOLUTION: A high resolution data value has 5 significant digits and may range in magnitude from +.00001 to +99999. A high resolution data value requires 2 Final Storage locations (4 bytes). All Input and Intermediate Storage locations are high resolution. Output to Final Storage defaults to low resolution; high resolution output must be specified by Instruction 78.

INDEXED INPUT LOCATION: An Input location entered as an instruction parameter may be indexed by keying "C" before it is entered by keying "A"; two dashes (--) will appear at the right of the display. Within a loop (Instruction 87, Section 12), this will cause the location to be incremented with each pass through the loop. Indexing is also used with Instruction 75 to cause an Input location, which normally remains constant, to be incremented with each repetition.

INPUT STORAGE: That portion of memory allocated for the storage of results of Input and Processing Instructions. The values in Input Storage can be displayed and altered in the ∗6 Mode.

INPUT/OUTPUT INSTRUCTIONS: Used to initiate measurements and store the results in Input Storage or to set or read Control/Logic Ports.

INSTRUCTION LOCATION NUMBER: As instructions are entered in a Program Table, they are numbered sequentially. The instruction location number gives the location of that instruction in the program sequence. When programming a table, the instruction location number and a P (e.g., 04: P00) prompts the user when it is time to enter an instruction.

INTERMEDIATE STORAGE: That portion of memory allocated for the storage of results of intermediate calculations necessary for operations such as averages or standard deviations. Intermediate storage is not accessible to the user.

LOOP: In a program, a series of instructions which are repeated a prescribed number of times, followed by an "end" instruction which exists the program from the loop.

LOOP COUNTER: Increments by 1 with each pass through a loop.

LOW RESOLUTION: The default output resolution. A low resolution data value has 4 significant decimal digits and may range in magnitude from +0.001 to +6999. A low resolution data value requires 1 Final Storage location (2 bytes).

MANUALLY INITIATED: Initiated by the user, usually with a keyboard, as opposed to occurring under program control.

MODEM/TERMINAL: Any device which: 1) has the ability to raise the CR510's ring line or be used with the SC32A to raise the ring line and put the CR510 in the Telecommunications Command State and 2) has an asynchronous serial communication port which can be configured to communicate with the CR510.

ON-LINE DATA TRANSFER: Routine transfer of data to a peripheral left on-site. Transfer is controlled by the program entered in the datalogger.

OUTPUT ARRAY: A string of data points output to Final Storage. Output occurs only when the Output Flag (Flag 0) is set. The first point of an Output Array is the Output Array ID, which gives the Program Table Number and the Instruction Location Number of the Instruction which sets the Output Flag. The data points which complete the Array are the result of the Output Processing Instructions which are executed while the Output Flag is set. The Array ends when the Output Flag is reset at the end of the table or when another Instruction acts upon the Output Flag. Output occurs only when the output flag is set. (Section 2.1)

OUTPUT INTERVAL: The time interval between initiations of a particular Output Array. Output occurs only when the Output Flag is set. The flag may be set at fixed intervals or in response to certain conditions (Sections OV4 and 1.2.1).

OUTPUT PROCESSING INSTRUCTIONS: Process data values and generate Output Arrays. Examples of Output Processing Instructions include Totalize, Maximize, Minimize, Average, etc. The data sources for these Instructions are values in Input Storage. The results of intermediate calculations are stored in Intermediate Storage. The ultimate destination of data generated by Output Processing Instructions is usually Final Storage but may be Input Storage for further processing. The transfer of processed summaries to Final Storage takes place when the Output Flag has been set by a Program Control Instruction.

PARAMETER: Used in conjunction with CR510 Program Instructions, parameters are numbers or codes which are entered to specify exactly what a given instruction is to do. Once the instruction number has been entered in a Program Table, the CR510 will prompt for the parameters by displaying the parameter number in the ID Field of the display.

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: See Print Device.

PROCESSING INSTRUCTIONS: These Instructions allow the user to further process input data values and return the result to Input Storage where it can be accessed for output processing. Arithmetic and transcendental functions are included in these Instructions.

PROGRAM CONTROL INSTRUCTIONS: Used to modify the sequence of execution of Instructions contained in Program Tables; also used to set or clear flags.

PROGRAM TABLE: That portion of memory allocated for storing programs consisting of a sequence of user instructions which control data acquisition, processing, and output to Final Storage. Programming can be separated into 2 tables, each having its own user-entered Execution Interval. A third table is available for programming subroutines which may be called by instructions in Tables 1 or 2. The ∗1 and ∗2 Modes are used to access Tables 1 and 2. The ∗3 Mode is used to access Subroutine Table 3. The length of the tables is constrained only by the total memory available for programming (Section 1.5). Tables 1 and 2 have independent execution intervals. Table 1 execution has the higher priority; it may interrupt Table 2.

RING LINE (PIN 3): Line pulled high by an external device to "awaken" the CR510.

SAMPLE RATE: The rate at which measurements are made. The measurement sample rate is primarily of interest when considering the effect of time skew (i.e., how close in time are a series of measurements). The maximum sample rates are the rates at which measurements are made when initiated by a single instruction with multiple repetitions.

SIGNATURE: A number which is a function of the data and the sequence of data in memory. It is derived using an algorithm which assures a 99.998% probability that if either the data or its sequence changes, the signature changes.

SYNCHRONOUS: The transmission of data between a transmitting and 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 Asynchronous).

THROUGHPUT: The throughput rate is the rate at which a measurement can be made, scaled to engineering units, and the reading stored in Final Storage. The CR510 has the ability to scan sensors at a rate exceeding the throughput rate (see SAMPLE RATE). The primary factor affecting throughput rate is the amount of processing specified by the user. In normal operation, all processing called for by an instruction must be completed before moving on the next instruction. The maximum throughput rate for a fast single-ended measurement is approximately 192 measurements per second (12 measurements, repeated 16 times per second). This rate is possible if the CR510's self-calibration function is suspended (this is accomplished by entering Instruction 24 into Program Table 2 while leaving the Execution Interval 0 so Program Table 2 never executes).

When the self-calibration function is operating, the maximum throughput rate for a fast, single-ended measurement is 192 measurements per second (12 measurements, 16 times per second).

APPENDIX B. ADDITIONAL TELECOMMUNICATIONS INFORMATION

B.1 TELECOMMUNICATIONS COMMAND WITH BINARY RESPONSES

Command	Description
[no. of loc.]**F**	BINARY DUMP - CR510 sends, in Final Storage Format (binary, the number of Final Storage locations specified (from current MPTR locations), then Signature (no prompt).

DATALOGGER J AND K COMMANDS

3142J The 3142J command is used to toggle datalogger user flags, request final storage data, and to establish the input locations returned by the K command. The format of the command is as follows:

3142J<CR>abcd...nNULL

where
1) "3142J<CR>" is the command.

2) "a" is a 1 byte value representing the user flags to be toggled. The most significant bit (MSB), if set, will toggle datalogger user flag 8. Likewise, the 2nd most significant bit, if set, will toggle user flag 7, and so on to the least significant bit which, if set, toggles user flag 1. Toggle means that if a flag is set, it will be then reset, or if it is reset, it will be set.

3) "b" is a 1 byte value whose MSB will determine whether Final Storage Data is returned after the K command. If the bit is set, Final Storage Data, if any, will be returned after the next K command. The datalogger initially has this bit reset upon entering telecommunications, but once set by a J command, it will remain set until reset by another J command or telecommunications is terminated.

The 2nd MSB set means a port toggle byte will follow and port status is to be returned with the K command. Like the MSB, this bit is reset upon entering telecommunications, but remains set once set until reset by another J command or telecommunications is terminated. Currently only the CR510 datalogger recognizes this bit.

The remaining bits are reserved.

4) If the 2nd MSB in "b" was set then "c" is a port toggle byte, otherwise "c,d,...,n" are each 1 byte binary values each representing a datalogger input storage location. The data at those locations will be returned after the next K command. ASCII code 1 (0000001 binary) represents input location 1. ASCII codes 2 (00000010 binary) represents input location 2, and so on. The order of the location requests is not important. The list is limited, however, to 62 total location requests.

5) "Null" or ASCII code 0 (00000000 binary) terminates the J command. Alternately, 11111111 binary aborts the J command. If aborted, flags will not be toggled and location requests will not be saved.

User Enters	Datalogger Echo
3	3
1	1
4	4
2	2
J	J
CR	CR
	LF
	<
a	a
b	b
c	c
d	d
n	n
Null	Null

K The K command returns datalogger time, user flag status, port status if requested, the data at the input locations requested in the J command, and Final Storage Data if requested by the J command. The format of the command is K<CR> (K Return). The datalogger will echo the K and Return and send a Line Feed. The amount of data that follows depends on the J command previously executed; four time bytes, a user flags byte, four bytes for each input location requested in the J command, Final Storage data in CSI's binary format if requested by the J command, and terminating in 7F 00 HEX and two signature bytes.

User Enters	Datalogger Echo
K	K
CR	CR
	LF
	Time Minutes byte 1
	Time Minutes byte 2
	Time Tenths byte 1
	Time Tenths byte 2
	Flags byte
	Ports byte (if requested)
	Data1 byte 1
	Data1 byte 2
	Data1 byte 3
	Data1 byte 4
	Data2 byte 1
	Data2 byte 2
	Data2 byte 3
	Data2 byte 4
	DataN byte 1
	DataN byte 2
	DataN byte 3
	DataN byte 4
	Final Storage Data bytes
	01111111 binary byte
	00000000 binary byte
	Signature byte 1
	Signature byte 2

Time Minutes byte 1 is most significant. Convert from binary to decimal. Divide by 60 to get hours, the remainder is minutes. For example, 00000001 01011001 (01 59 HEX) is 345 decimal minutes or 5:45.

Time Tenths byte 1 is most significant. Convert from binary to decimal. Divide by 10 to get seconds and tenths of seconds. For example, 00000001 11000110 (01 C6 HEX) is 454 decimal or 45.4 seconds. Thus the datalogger time for 01 59 01 C6 HEX is 5:45:45.4.

The Flags byte expresses datalogger user flag status. The most significant bit represents Flag 8, and so on to the least significant bit which represents Flag 1. If a bit is set, the user flag is set in the datalogger.

The optional ports byte (currently on return if requested by a CR510 J command) expresses the datalogger port status. The most significant bit represents Port 8, and so on to the least significant bit which represents Port 1.

For each input location requested by the J command, four bytes of data are returned. The bytes are coded in Campbell Scientific, Inc. Floating Point Format. The format is decoded to the following:

$$\text{Sign}(\text{Mantissa}*2^{(\text{Exponent})})$$

Data byte 1 contains the Sign and the Exponent. The most significant bit represents the Sign; if zero, the Sign is positive, if one, the Sign is negative. The signed exponent is obtained by subtracting 40 HEX (or 64 decimal) from the 7 remaining least significant bits.

Data bytes 2 to 4 are a binary representation of the mantissa with byte 2 the most significant and 4 the least. The mantissa ranges in value from 80 00 00 hex (0.5 decimal) to FF FF FF HEX ($1-2^{-24}$ decimal). The one exception is for zero which is 00 00 00 00 HEX.

The Mantissa is calculated by converting Data bytes 2 to 4 into binary. Each bit represents some fractional value which is summed for all 24 bits. The bits are arranged from MSB to LSB with the most significant as bit_{23} and least significant as bit_0. The value that each bit represents = 2^{n-24}; where n=bit location. For example, if there was a 1 at bit_{20}, it's value would be $2^{(20-24)}$ or 2^{-4}.

A simple method for programming this is as follows:

Set Mantissa = 0.

Set Bit Value = 0.5.

For loop count = 1 to 24 do the following:

> If the MSB is one, then add Bit Value to the Mantissa.
>
> Shift the 24 bit binary value obtained from Data bytes 2 to 4 one bit to the left.
>
> Multiply Bit Value by 0.5.

End of loop.

Another method that can be used as an estimate is to convert Data bytes 2 to 4 from a long integer to floating point and dividing this value by 16777216.

As an example of a negative value, the datalogger returns BF 82 0C 49 HEX.

Data byte 1 = BF HEX.

Data byte 2 to 4 = 82 0C 49 HEX (or 8522825 decimal).

Data byte 1 is converted to binary to find the Sign. BF HEX = 10111111 BINARY.

The most significant bit is 1 so the Sign is NEGATIVE.

The exponent is found by subtracting 40 HEX from the remaining least significant bits. Converting the binary to hexadecimal, 111111 BINARY = 3F HEX (or 63 decimal).

3F - 40 HEX = FF FF FF FF FF HEX. Or in decimal: 63 - 64 = -1.

Exponent is -1 decimal.

The binary representation of Data bytes 2 to 4 is: 10000010 00001100 01001001.

Summing all the fractional values: $2^{-1} + 2^{-7} + 2^{-13} + 2^{-14} + 2^{-18} + 2^{-21} + 2^{-24} = 0.50800$.

Using the estimate method to find the Mantissa = 82 0C 49 HEX / 1 00 00 00 HEX (or 8522825 / 16777216) which is 0.50800 decimal.

The value is then $(-)0.508*2^{-1}$ which equals -0.254.

As an example of a positive value, the datalogger returns 44 D9 99 9A HEX.

Data byte 1 = 44 HEX.

Data byte 2 to 4 = D9 99 9A HEX (or 891290 decimal).

Data byte 1 is converted to binary to find the Sign. 44 HEX = 01000100 BINARY.

> **NOTE:** Don't lose the leading zero!

The most significant bit is 0 so the Sign is POSITIVE.

The exponent is found by subtracting 40 HEX from the remaining least significant bits.

Converting the binary to hexadecimal, 1000100 BINARY = 44 HEX (or 68 decimal).

44 - 40 HEX = 4 HEX. Or in decimal: 68 - 64 = 4.

Exponent is 4 decimal.

The binary equivalent of Data bytes 2 to 4 is: 11011001 10011001 10011010.

Summing all the fractional values:

$2^{-1} + 2^{-2} + 2^{-4} + 2^{-5} + 2^{-8} + 2^{-9} + 2^{-12} + 2^{-13} + 2^{-16} + 2^{-17} + 2^{-20} + 2^{-21} + 2^{-23} = 0.85000$.

Using the estimate method to find the Mantissa = D9 99 9A HEX / 1 00 00 00 HEX (or 14260634 / 16777216) which is 0.85000 decimal.

The value is then $(+)0.85*2^{4}$ which equals 13.60.

If appropriately requested by a J command, Final Storage data, if any, will immediately follow the input location data. Refer to the datalogger manual for a description of how to decode Final Storage data in CSI's binary data format. Final Storage data will be limited to not more than 1024 bytes per K command.

The K command data is terminated with 7F 00 HEX (a unique binary format code) followed by two signature bytes. Refer to the datalogger manual for the meaning and calculation of the signature bytes. The signature in this case is a function of the first time byte through the 7F 00 HEX bytes. Calculate the signature of the bytes received and compare with the signature received to determine the validity of the transmission.

B.2 FINAL STORAGE FORMAT

CR510 data is formatted as either 2 byte LOW Resolution or 4 byte HI Resolution values. The first two bytes of an Output Array contain a unique code (FC Hex) noting the start of the Output Array and the Output Array ID, followed by the 2 or 4 byte data values. At the end of the data sent in response to the telecommunications F command a 2 byte signature is sent (see below).

Representing the bits in the first byte of each two byte pair as ABCD EFGH (A is the most significant bit, MSB), the byte pairs are described here.

LO RESOLUTION FORMAT - D,E,F, NOT ALL ONES

Bits	Description
A	Polarity, 0 = +, 1 = -.
B, C	Decimal locators as defined below.
D-H plus second byte	13 bit binary value (D=MSB). Largest possible number without D, E, and F all 1 is 7167, but CSI defines the largest allowable range as 6999.

The decimal locators can be viewed as a negative base 10 exponent with decimal locations as follows:

B	C	Decimal Location
0	0	XXXX.
0	1	XXX.X
1	0	XX.XX
1	1	X.XXX

DATA TYPE WHEN D,E,F, ALL EQUAL ONE

If D, E, and F are all ones, the data type is determined by the other bits as shown below. X implies a "don't care" condition; i.e., the bit can be either 1 or 0 and is not used in the decode decision.

A	B	C	D	E	F	G	H	DATA TYPE AND SECOND BYTE FORMAT
1	1	1	1	1	1	X	X	A,B,C, = 1 - Start of Output Array, G & H are the most significant bits of the Output Array ID. All 8 bits of the 2nd byte are also included in the ID.
X	X	0	1	1	1	X	X	C = 0 - First byte of a 4 byte value.
0	0	1	1	1	1	X	X	A,B = 0; C = 1 - Third byte of a 4 byte value.
0	1	1	1	1	1	1	1	A = 0; remaining bits = 1 - First byte of a 2 byte "dummy" word. The CR510 always transmits a 0 for the 2nd byte, but the word can be decoded on the basis of the 1st byte only.

HI RESOLUTION FORMAT

Continuing to use the A-H bit representation, the four byte number is shown below as two two byte pairs.

AB0111GH XXXXXXXX 001111GH XXXXXXXX

BITS, 1ST BYTE, 1ST PAIR	DESCRIPTION
CDEF = 0111	Code designating 1st byte pair of four byte number.
B	Polarity , 0 = +, 1 = -.
G,H,A,	Decimal locator as defined below.
2nd byte	16th - 9th bit (left to right) of 17 bit binary value.
ABCDEF = 001111	Code designating 2nd byte pair of four byte number.
G	Unused bit.
H	17th and MSB of 17 bit binary value.
2nd byte	8th - 1st bit (left to right) of 17 bit binary value.

CSI defines the largest allowable range of a high resolution number to be 99999.

Interpretation of the decimal locator for a 4 byte data value is given below. The decimal equivalent of bits GH is the negative exponent to the base 10.

BITS G H A	DECIMAL FORMAT 5 digits
0 0 0	XXXXX.
0 0 1	XXXX.X
0 1 0	XXX.XX
0 1 1	XX.XXX
1 0 0	X.XXXX
1 0 1	.XXXXX

B.3 GENERATION OF SIGNATURE

At the end of a binary transmission, a signature is sent. The signature is a 2 byte integer value which is a function of the data and the sequence of data in the Output Array. It is derived with an algorithm that assures a 99.998% probability of detecting a change in the data or its sequence. The CR510 calculates the signature using each transmitted byte beginning with the Final Storage format data (for K command, echo and carriage return line feed are not included) until the 2 byte signature itself. By calculating the signature of the received data and comparing it to the transmitted signature, it can be determined whether the data was received correctly.

SIGNATURE ALGORITHM

- S1,S0 - represent the high and low bytes of the signature, respectively
- M - represents a transmitted data byte
- n - represents the existing byte
- n+1 - represents the new byte
- T - represents a temporary location
- C - represents the carry bit from a shift operation

1. The signature is initialized with both bytes set to hexadecimal AA.

$$S_1(n) = S_0(n) = AA$$

2. When a transmitted byte, M(n+1), is received, form a new highsignature byte by setting it equal to the existing low byte. Save the old high byte for later use.

$$T_1 = S_1(n)$$

$$S_1(n+1) = S_0(n)$$

3. Form a temporary byte by shifting the old low signature byte one bit to the left and adding any carry bit which results from the shift operation. A "shift left" is identical to a multiply by 2. Ignore any carry bit resulting from the add.

$$T_2 = \text{shift left } (S_0(n)) + \text{carry}$$

4. Form the new low signature byte by adding the results of operation 3 to the old high signature byte and the transmitted byte. Ignore any carry bits resulting from these add operations.

$$S_0(n+1) = T_2 + S_1(n) + M(n+1)$$

As each new transmitted byte is received, the procedure is repeated.

B.4 ∗D COMMANDS TO TRANSFER PROGRAM WITH COMPUTER

∗D Commands 1 and 2 (when entered from the Keyboard/Display) and 7 have an additional 2 digit option parameters (7 is entered with the Storage Module address, e.g., 71). The CR510 will display the command number and prompt for the option. If the keyboard display is not being used, the CR510 will have already set the baud rate to that of the device it is communicating with and will be ready to send or receive the file as soon as command 1 or 2 is entered.

TABLE B.4-1. ∗D Command 1 and 2 Options

Command		Option Code Description
1 & 2	1x	Synchronously addressed
	4x	Hardware enabled
		x = Baud Rate Codes 0 - 300 1 - 1200 2 - 9600 3 - 76,800

After the option code is keyed in, key "A" to execute the command. Command 2 will be aborted if no data is received within 40 seconds.

WHEN COMMAND 2 IS EXECUTED ALL DATA IN INPUT AND INTERMEDIATE STORAGE ARE ERASED. This section describes commands 1 and 2.

SENDING ASCII PROGRAM INFORMATION

Program listings are sent in ASCII. At the end of the listing, the CR510 sends control E (5 hex or decimal) twice.

Table 1.8-4 is an example of the program listing sent in response to command. Your numbers may be different. Note that the listing uses numbers for each mode: The numbers for ∗A, ∗B, and ∗C modes are 10, 11, and 12, respectively.

TABLE B.4-2. Example Program Listing From ∗D Command 1

```
MODE 1
SCAN RATE 5
1:P17
1:1

2:P86
1:10

3:P70
1:1
2:1

4:P0

MODE 2
SCAN RATE 0

MODE 3
1:P0

MODE 10
1:28
2:64
3:0
4:5332
5:1971

MODE 12
1:0
2:0

MODE 11
1:6597          6:
2:30351         7:
3:48            8:
4:0             9:
5:0             10:
^E ^E           11:
```

LOAD PROGRAM FROM ASCII FILE

Command 2 sets up the CR510 to load a program which is input as serial ASCII data in the same form as sent in response to command 1.

A download file need not follow exactly the same format that is used when listing a program (i.e., some of the characters sent in the listing are not really used when a program is loaded). Some rules which must be followed are:

1. "**M**" must be the first character other than a carriage return, line feed, semicolon, or 7D Hex. The "**M**" serves the same function as "∗" does from the keyboard. The order in which the Modes are sent in does not matter (i.e., the information for Mode 3 could be sent before that for Mode 1).
2. "**S**" is necessary prior to the **S**can Rate (execution interval).
3. The colons (**:**) are used to mark the start of actual data.
4. A semicolon (**;**) tells the CR510 to ignore the rest of the line and can be used after an entry so that a comment can be added.

There are 4 two-character control codes which may be used to verify that the CR510 receives a file correctly:

^B ^B (2hex, 2hex)--Discard current buffer and reset signature

^C ^C (3hex, 3hex)--Send signature for current buffer

^D ^D (4hex, 4hex)--Load current buffer and reset signature

^E ^E (5hex, 5hex)--Load current buffer, Exit and compile program

As a download file is received, the CR510 buffers the data in memory; the data is not loaded into the editor or compiled until the CR510 receives a command to do so. The maximum size of the buffer is 1.5K. The minimum file that could be sent is the program listing, then ^E ^E. ^C ^C tells the CR510 to send the signature (Appendix B.3) for the current buffer of data. If this signature does not match that calculated by the sending device, ^B ^B can be sent to discard the current buffer and reset the signature. If the signature is correct, ^D ^D can be sent to tell the CR510 to load the buffer into the editor and reset the signature. Once the complete file has been sent and verified, send ^E ^E to compile the program and exit the load command.

APPENDIX C. ASCII TABLE

American Standard Code for Information Interchange
Decimal Values and Characters

(X3.4-1968)

Dec.	Char.	Dec.	Char.	Dec.	Char.	Dec.	Char.
0	CONTROL @	32	SPACE	64	@	96	`
1	CONTROL A	33	!	65	A	97	a
2	CONTROL B	34	"	66	B	98	b
3	CONTROL C	35	#	67	C	99	c
4	CONTROL D	36	$	68	D	100	d
5	CONTROL E	37	%	69	E	101	e
6	CONTROL F	38	&	70	F	102	f
7	CONTROL G	39	'	71	G	103	g
8	CONTROL H	40	(	72	H	104	h
9	CONTROL I	41	)	73	I	105	i
10	CONTROL J	42	*	74	J	106	j
11	CONTROL K	43	+	75	K	107	k
12	CONTROL L	44	,	76	L	108	l
13	CONTROL M	45	-	77	M	109	m
14	CONTROL N	46	.	78	N	110	n
15	CONTROL O	47	/	79	O	111	o
16	CONTROL P	48	0	80	P	112	p
17	CONTROL Q	49	1	81	Q	113	q
18	CONTROL R	50	2	82	R	114	r
19	CONTROL S	51	3	83	S	115	s
20	CONTROL T	52	4	84	T	116	t
21	CONTROL U	53	5	85	U	117	u
22	CONTROL V	54	6	86	V	118	v
23	CONTROL W	55	7	87	W	119	w
24	CONTROL X	56	8	88	X	120	x
25	CONTROL Y	57	9	89	Y	121	y
26	CONTROL Z	58	:	90	Z	122	z
27	CONTROL [	59	;	91	[	123	{
28	CONTROL \	60	<	92	\	124	|
29	CONTROL]	61	=	93	]	125	}
30	CONTROL ^	62	>	94	^	126	~
31	CONTROL _	63	?	95	_	127	DEL

APPENDIX D. DATALOGGER INITIATED COMMUNICATIONS

Datalogger initiated communications, commonly referred to as "callback," is when the datalogger initiates a call back to a computer. A CR510 uses Instruction 97 to initiate a call. For complete information on Instruction 97 and its parameters, refer to section 12.

D.1 INTRODUCTION

In most applications, the datalogger initiates a call to the computer to notify the user that a specific condition has occurred. For example, a station monitoring conditions at a dam could be setup to call the computer to notify the user when the water level gets above a specified level. Another example would be a CR510 initiating a call to a PC when conditions in a control environment are not within acceptable ranges. Datalogger initiated calls can also be triggered based on time (Instruction 92).

This appendix gives an example of a CR510 calling a computer via a COM200 Phone Modem whenever the temperature exceeds 32° C. An example program, the necessary computer files, and setup are included. This is only an example, instruction parameters should be changed to match your system.

D.2 EXAMPLE PHONE CALLBACK PROGRAM BASED ON A CONDITION

The following program example measures a HMP35C Temperature/RH sensor every 30 seconds and stores hourly data. Whenever the temperature is above 32° C, the datalogger initiates a call to a dedicated computer.

Instruction 97 uses the status of a Flag to determine when to initiate the call. Any user flags (1-8) may be used as Instruction 97's Interrupt Flag. In this example when temperature is within acceptable limits (< 32° C), Flag 5 is set high, otherwise Flag 5 is set low. If Flag 5 is low when Instruction 97 executed, the datalogger will attempt to call the computer.

Once the computer answers the datalogger's call, the datalogger sends a 3 digit ID# (Parameter 8 of Instruction 97). This ID# is the "name" of the site. Each site that calls a computer needs to have a unique 3 digit ID#. The computer's software will use this 3 digit ID# to identify which datalogger is calling and what it should do as a result of the call. Section D.3 covers the DOS software setups for this example program. Section D.4 covers the Windows software setup of this example program.

When this CR510 tries to call, it will dial out the phone number given in Instruction(s) 63. If the datalogger is dialing 7 or less digits, the second Instruction 63 in not needed. If the datalogger needs to dial 8 digits, a second Instruction 63 is still required to dial the 13. **Do not forget to put a 13 for the last parameter in the last Instruction 63.** This is the only way the CR510 knows when to stop sending characters and to instruct the COM200 to make the call.

PROGRAM

```
*Table 1 Program
  01:      30          Execution Interval (seconds)

1:  Batt Voltage (P10)
  1:        3          Loc [ BATT_VOLT ]

2:  Temp (107) (P11)
  1:        1          Reps
  2:        1          SE Channel
  3:        2          Excite all reps w/Exchan 2
  4:        1          Loc [ AIR_TEMP  ]
  5:        1          Mult
  6:        0          Offset

3:  Do (P86)
  1:        4          Set Port 1 High

4:  Excitation with Delay (P22)
  1:        1          Ex Channel
  2:        0          Delay W/Ex (units = 0.01 sec)
  3:       15          Delay After Ex (units = 0.01 sec)
  4:        0          mV Excitation

5:  Volts (SE) (P1)
  1:        1          Reps
  2:       25          ± 2500 mV 60 Hz Rejection
                       Range
  3:        2          SE Channel
  4:        2          Loc [ RH        ]
  5:       .1          Mult
  6:        0          Offset
```

6: Do (P86)
1: 51 Set Port 1 Low

7: If time is (P92)
1: 0 Minutes (Seconds --) into a
2: 60 Interval (same units as above)
3: 10 Set Output Flag High

8: Real Time (P77)
1: 120 Day,Hour/Minute (2400 at midnight)

9: Average (P71)
1: 2 Reps
2: 1 Loc [AIR_TEMP]

10: Maximize (P73)
1: 2 Reps
2: 0 Value Only
3: 1 Loc [AIR_TEMP]

11: Sample (P70)
1: 2 Reps
2: 3 Loc [BATT_VOLT]

12: IF (X<=>F) (P89)
1: 1 X Loc [AIR_TEMP]
2: 4 <
3: 31 F
4: 15 Set Flag 5 High

13: IF (X<=>F) (P89)
1: 1 X Loc [AIR_TEMP]
2: 3 >=
3: 32 F
4: 25 Set Flag 5 Low

14: Initiate Telecommunications (P97)
1: 22 Phone Modem/9600 Baud
2: 5 Disabled when User Flag 5 is High
3: 75 Seconds Call Time Limit
4: 120 Seconds Before Fast Retry
5: 2 Fast Retries
6: 5 Minutes before Slow Retry
7: 4 Failures Loc [Failures]
8: 115 Data Logger ID

15: Extended Parameters (P63)
1: 1 Option
2: 8 Option
3: 0 Option
4: 1 Option
5: 7 Option
6: 5 Option
7: 2 Option
8: 3 Option

16: Extended Parameters (P63)
1: 2 Option
2: 6 Option
3: 8 Option
4: 13 Option
5: 00 Option
6: 00 Option
7: 00 Option
8: 00 Option

D.3 PC208 DOS COMPUTER SOFTWARE AND IT'S COMPUTER SETUP

The station file must be called ###.STN where ### is the 3 digit ID# (Parameter 8 of Instruction 97) from the datalogger program. After the computer answers the call, it searches for a station file (.STN) with the name of the 3 digit ID#. The computer will than collect data based on the data collection method in that station file. In this example, the 3 digit ID# is 115 so the station file is called 115.STN and the computer will store the data in a file called 115.DAT. After the computer has collected the data, it will look for a batch file (.BAT) with the same filename (in this example, 115.BAT). If it finds the correct batch file, it will execute the batch file before returning to the wait mode to wait for another call.

To setup a computer to answer datalogger initiated calls, do the following steps in order.

First, create the station file and use it to call the site to make sure it is correct (Figure D.3-1). Make sure to change the filename, COM Port, and phone number to match your system.

Telecommunication Parameters For Station:	115
Datalogger Type:	CR510
Security Code:	0
Use Asynchronous Communications Adapter:	COM2
Communications Baud Rate:	9600
Data File Format:	Comma separated ASCII
Final Storage Collection Area:	Area 1
Interface Device:	#1: Hayes Modem
Number:	18017509563

FIGURE D.3-1 Example Station File Settings for 115.STN

Next, create a schedule (for PC208E) or a script (for TELCOM) file. This file tells the computer to a) Wait for a call, b) Answer any incoming calls, & c) the COM port & baud rate to use. In this example, the file CALLME12.SCR is used. You may name the file anything you want, but it must have the extension of .SCR. The .SCR file is the same for TELCOM or PC208E (Figure D.3-2). Make sure to modify the file for the correct COM port.

```
/A COM2:9600
/W
```

FIGURE D.3-2 Example .SCR file for Datalogger Initiated Communications

Once the .SCR file is created, set the phone modem at the computer to Auto Answer. See the phone modem's manual or the PC208 manual for details.

The final step is to run the .SCR file. The computer will then start waiting for incoming calls. Once this is accomplished, the computer can't be used for anything else until you give it the commands to quit waiting for incoming calls. To run the .SCR using PC208E, select the schedule (File | Open | Schedule) and start the scheduled data collection (DataCollection | Scheduled). To run the .SCR from TELCOM, at the DOS prompt type TELCOM to run TELCOM. When TELCOM prompts you to "Enter station or script filename:", type in the filename.SCR and press <Enter>. For this example, you would enter CALLME12.SCR.

Once the computer is waiting for incoming calls, it will bring up a screen similar to the one shown in Figure D.3-3. The computer will now answer incoming calls.

```
Telecommunications Program ver. 7.3
Copyright (C) 1986,1991 Campbell Scientific, Inc.
Executing script file "C:\PC208\CALLME12"

Waiting for:
   - next wake up time (01/19/38 03:14:07)
   - PC203 on/off switch to be turned on
   - modem ring signal to become active
   - a ctrl-C or Esc to be pressed

Current Date and Time:  01/15/96 14:36:28
```

FIGURE D.3-3 Computer Screen in PC208E/TELCOM Waiting for a Call

APPENDIX E.
CALL ANOTHER DATALOGGER VIA PHONE OR RF

E.1 INTRODUCTION

Instructions 97, Initiate Telecommunications, and 63, Extended Parameters can be used to call another datalogger and collect data in input locations. This function can only be accomplished via phone or radio modems.

E.2 PROGRAMMING

Instruction 97 initiates the call and Instruction 63 specifies the dialing path and special options. More than one Instruction 63 may be required. The calling datalogger's program requires Instructions 97 and 63 for each datalogger it calls. Only one datalogger can be called with each set of instructions.

When the calling datalogger executes Instruction 97, it establishes communication and then toggles a flag in the remote datalogger. After a specified delay, the calling datalogger monitors that same flag (in the remote datalogger). When the calling datalogger detects the flag is low, it collects the specified input locations. The collected values are stored in the calling datalogger's input locations beginning at the location after the "failure location" (Parameter 7 of Instruction 97).

In the example below, the remote datalogger is programmed to make measurements when its flag 1 is high and then set flag 1 low. The calling datalogger toggles flag 1 and then waits until flag 1 is low to collect the data, insuring that the input location values are transferred after the desired measurements and processing are complete.

Section E.2 covers programming requirements for the calling CR510. Section E.3 shows the programming requirements for the remote datalogger.

E.3 PROGRAMMING FOR THE CALLING CR510

Programming Example 2.1 uses a DC112 phone modem while Programming Example 2.2 uses a RF modem.

E.3.1 INSTRUCTION 97

Many of the parameters in Instruction 97 don't apply when being used to call another datalogger. See Section 12 for more detailed information on Instruction 97's parameters.

PAR. NO.	DATA TYPE	DESCRIPTION
01:	2	Modem/Baud Option
02:	2	Flag No. to Disable Instruction 97
03:	4	Seconds Call Time Limit
04:	4	Seconds Before Fast Retry
05:	2	Fast Retries
06:	4	Minutes Before Slow Retry
07:	4	Failures Loc :
08:	4	Datalogger ID

Parameter 1
This specifies the first modem type to call out through and the appropriate baud rate. See Table E.3-1 for valid options. An RF modem is in the SDC state when the 9th dip switch is closed.

TABLE E.3-1. Option Code for Modem Type and Baud Rate

	Baud Rate		
Modem	300	1200	9600
RF, ME State	00	01	02
Phone (DC112/COM200)	20	21	22
Voice Data	40	41	42
RF, SDC State	50	51	52

Parameter 2
This specifies the flag lowered in the calling datalogger to initiate the call to the other datalogger. This should not be confused with the flag that the calling datalogger sets in the remote datalogger (specified in Instruction 63).

Parameter 3
This is the total length of time the calling datalogger will try to collect input locations from the remote datalogger. The timer begins when a call is initiated and includes the dialing time. This timer must be long enough for the remote datalogger to detect that a flag has been set (i.e., long enough to allow for the execution

interval of the remote datalogger), make the appropriate measurements, lower the flag, and allow for the input location to be transferred.

Parameters 4, 5, 6, and 8
These parameters don't apply when calling a datalogger. Leave these options as 0.

Parameter 7
This parameter specifies the location to store the number of times the call fails. The collected values are stored in the calling datalogger's input locations beginning at the location after the Failure Location.

E.3.2 INSTRUCTION 63

Instruction 63 is used to specify the dialing path and special options for Instruction 97. More than one Instruction 63 may be required. Instruction 68 may be used instead of instruction 63 when a 3 or 4 digit parameter needs to be entered instead of the 2 digit parameter allowed by Instruction 63.

The first set of the parameters are the phone number or RF path. Each digit of the phone number or an RF ID# goes into a separate parameter.

Separate each RF ID# by a 32 (Space). After the last RF ID#, enter a 70 ("F").

If the phone modem at the remote site is a VS1, after the last digit of the phone number you need several 44s (",") then a 42 ("∗"), and then a 9. Enough commas are needed so the VS1 has enough time to pickup the phone line and to start talking. Normally 2 or 3 commas is sufficient. These characters are required to tell the remote VS1 to go into the data/computer mode instead of voice.

After the last digit of the phone number (including any voice codes) or RF path (after the 70), enter a 68 ("D") as one parameter. This tells the calling datalogger that it is calling another datalogger.

The next two parameters indicate the number of locations to retrieve and the beginning input location to collect respectively. You can not retrieve from input locations 255 or higher. You are limited on the number of locations you may collect per call.

The next parameter is used to specify the flag to toggle and monitor in the remote datalogger.

The second to last parameter is the delay (in units of 0.1 seconds) the calling datalogger should wait before checking to see if the flag has been reset. Once the calling datalogger determines the flag has been reset, it collects the data.

The last parameter needs to be a 13 to terminate the call.

Programming Example 2.1: Calling CR510 Using Phone Modem

Program: This program fragment calls a datalogger at phone number '"539" every 2 minutes. The CR510 toggles Flag 1 in the remote datalogger to trigger measuring and data collection. It collects 3 input locations beginning at the remote datalogger's location 1. These values are stored in locations 2, 3, and 4.

Flag Usage: Flag 2 is used to control Instruction 97 (when to make the call). The program need only set the flag low; Instruction 97 will set it high after a successful transfer.

```
*Table 1 Program
  01:      1.0        Execution Interval (seconds)

1:  If time is (P92)
  1:         0        Minutes (Seconds --) into a
  2:         2        Interval (same units as above)
  3:        22        Set low Flag 2

2:  Initiate Telecommunications (P97)
  1:        21        Phone Modem/1200 Baud
  2:         2        Disabled when User Flag 2 is High
  3:        45        Seconds Call Time Limit
  4:         0        Seconds Before Fast Retry
  5:         0        Fast Retries
  6:         0        Minutes before Slow Retry
  7:         1        Failures Loc [ Failures  ]
  8:      0000        Data Logger ID

3:  Extended Parameters (P63)
  1:         5        Phone # = 539
  2:         3
  3:         9
  4:        68        "D" to call datalogger
  5:         3        # of Locs to collect
  6:         1        1st Loc to collect
  7:         1        Flag to toggle in Remote
                      Datalogger
  8:         0        Delay
```

```
4:  Extended Parameters (P63)
   1:      13          Terminate character
   2:      00
   3:      00
   4:      00
   5:      00
   6:      00
   7:      00
   8:      00
```

Programming Example 2.2: Calling CR510 using RF modems

Program: This program fragment calls a datalogger every 2 minutes at using the RF Path of "4 10F" (that is from the calling CR510 to RF ID# 4 to RF ID# 10 at the remote site). The CR510 toggles Flag 1 in the remote datalogger to trigger measuring and data collection. It collects 3 input locations beginning at the remote datalogger's location 1. These values are stored in locations 2, 3, and 4.

Flag Usage: Flag 2 is used to control Instruction 97 (when to make the call). The program need only set the flag low; Instruction 97 will set it high after a successful transfer.

```
*Table 1 Program
   01:      1.0         Execution Interval (seconds)

1:  If time is (P92)
   1:       0          Minutes (Seconds --) into a
   2:       2          Interval (same units as above)
   3:      22          Set low Flag 2

2:  Initiate Telecommunications (P97)
   1:      02          RF Modem/9600 Baud
   2:       2          Disabled when User Flag 2
                       is High
   3:      45          Seconds Call Time Limit
   4:       0          Seconds Before Fast Retry
   5:       0          Fast Retries
   6:       0          Minutes before Slow Retry
   7:       1          Failures Loc [ Failures  ]
   8:    0000          Data Logger ID

3:  Extended Parameters (P63)
   1:       4          RF ID# of repeater site = 4
   2:      32          Space
   3:       1          RF ID# of 2nd site = 10
   4:       0
   5:      70          "F"
   6:      68          "D" to call datalogger
   7:       3          # of Locs to Collect
   8:       1          1st Loc to Collect

4:  Extended Parameters (P63)
   1:       1          Flag to toggle in Remote
                       Datalogger
   2:       0          Delay
   3:      13          Terminate character
   4:       0
   5:       0
   6:       0
   7:       0
   8:       0
```

E.4 REMOTE DATALOGGER PROGRAMMING

The remote datalogger should be programmed to detect when the specified flag is set high. When the flag is set high, measurements and processing are done; then the flag is set low. Once the calling CR510 detects that the flag is low, it collects the specified input locations.

Since the calling CR510 sets the flag by toggling it, the remote datalogger should be programmed so the flag is always low except when the calling CR510 sets it. If the flag is already high when the call occurs, the flag will be toggled (setting it low), and the input locations will be transferred immediately.

The calling CR510 may have to wait an entire remote datalogger execution interval before the flag set is detected. The execution interval in the datalogger being called should be set to allow the call to be completed within the time limit set in Parameter 3 of Instruction 97 of the calling CR510.

Program example for a remote 21X

Program: Measures battery voltage, and internal and external temperatures in response to a call from the CR510.

Flag Usage: Flag 1 will be set high by the calling datalogger. The flag is lowered after the measurements are made.

```
*  1    Table 1 Programs
   01:      1          Sec. Execution Interval

01:       P91          If Flag
   01:     11          1 is set
   02:     30          Then Do

02:       P10          Battery Voltage
   01:      1          Loc :
```

```
03:     P17         Panel Temperature
  01:     2         Loc :

04:     P11         Temp 107 Probe
  01:     1         Rep
  02:     1         IN Chan
  03:     1         Excite all reps w/EXchan 1
  04:     3         Loc :
  05:     1         Mult
  06:     0         Offset

05:     P86         Do
  01:    21         Set low Flag 1

06:     P95         End

07:       P         End Table 1
```

APPENDIX F. MODBUS ON THE CR10(X) AND CR510

Modbus communication capability is available as a Library Special on the CR10(X) and CR510 dataloggers.

The implementation of MODBUS on the CR10(X) and CR510 allows input locations, ports, and flags to be read or to be set. Not supported are historical data retrieval, program downloads, setting the clock, and other functions of PC208.

Modbus on the CR10(X) and CR510 does not preclude interfacing with PC208 as long as the communications system (radios, modems, etc.) is compatible with PC208.

Applications Engineers at Campbell Scientific are not proficient in, and do not provide technical support for any of the commercially available software packages or PLCs that support Modbus. CR10(X) users have successfully communicated using the following software packages: Wonderware Intouch, Labview, Iconics Genesis for DOS, and Citect for Windows. LabTech's modbus driver does not support floating point, and thus is not compatible with CR10(X)s or CR510s.

F.1 TERMINOLOGY

The CR510s communicate in RTU mode (not ASCII) to other Modbus devices.

The SCADA and MMI software packages and Modbus use different terminology than does Campbell Scientific. For example in SCADA software using Modbus, to reference the CR510's input location number 3, specify "register 40005 F". The CR510's input locations are analogous to registers in Modbus. Because of floating point considerations, two registers of memory are required for each input location. Modbus registers are offset by 40000, therefore, the register equivalent of any input location X is: 40000+2X-1. The register is followed by a " F" in the Intouch software. Ports and flags are more straight forward. Modbus does not differentiate between ports and flags. They have "coils" originally meaning coil relays that are either on or off. Ports 1 on the CR510 correspond to coils 1, and flags 1..8 correspond to coils 9..16.

Campbell Scientific	Modbus
input location 1,2,3...X	register 40001,40003, 40005...(40000+2X-1)
port 1	coil 1
flag 1..8	coil 9..16

F.2 COMMUNICATIONS AND COMPATIBILITY

Direct RS232 connect and telephone are supported by PC208 and most SCADA software packages. Modbus communications are on the serial port of the CR510s. For a direct connect to a computer an SC32A is used just as it is for PC208 communications. Other devices such as spread spectrum radios may require an SC932. COM200 and VS1 telephone modems and some radio modems are compatible with modbus communications.

In the PC based software's Modbus driver or with a PLC the communications port, baud rate, data bits, stop bits, and parity are set. The CR10(X)s communicate at 9600, 1200, or 300 baud (they automatically sync to the baud rate of the computer), N,8,1.

The Modbus address can be set in *D mode allowing multiple dataloggers on the same serial port for direct communications. The address is set in *D window 8. Valid addresses are 1 to 254 with the exception of 13.

> **NOTE:** In earlier beta versions of the CR510 operating system the Modbus address is fixed at address 1.

This can be done from the CR10KD or a computer terminal with a terminal emulator. See section OV3 and section 5 of the CR510 manual for details.

F.2.1 RF COMMUNICATIONS

The Campbell Scientific UHF/VHF radio package is of course compatible with PC208. To also do Modbus on SCADA packages using the Campbell Scientific radio package, a special operation system PROM for the RF95 radio modem is needed. The RF95 PROMs will facilitate an auto route to the correct RF95. The RF95 address (dip switch) is set to the Modbus address of the CR510 at each specific site in this case. In the SCADA software the COM port settings and the Modbus address need to be set, but no dialing is necessary. The RF95s would connect through to the correct CR510. Repeater stations are not supported. ALL REMOTES MUST COMMUNICATE DIRECTLY TO THE RF232 BASE STATION.

Many of the popular radio systems for Modbus applications are not compatible with PC208 if there is more than one remote site.

F.2.2 CR510 TO CR510 OR CR10(X) COMMUNICATION

Library special software is available that allows one CR510 to communicate with another CR510 using Modbus in a P97 instruction. The standard implementation of Modbus enables the CR510 to respond to Modbus commands, not to issue them.

F.3 MORE ON MODBUS

Following is a brief explanation of the modbus functions supported and the strings that are transmitted. Most users will not need this information as the CR10(X) and PC based Modbus drivers handle this level of communication transparent to the user. If more information is needed, please refer to Modicon's publication "Modicon Modbus Protocol Reference Guide" (PI-MBUS-300 Rev. D). The Modicon phone number is 508-794-0800.

Functions supported by CR510	Description
1	read coil status
3	read holding registers
5	force single coil
15	force multiple coils
16	preset multiple registers

Example 1: Function 1 to read coils 1 to 16 from slave device 17

	Hex
Slave address	11
Function	01
Starting Address Hi	00
Starting Address Lo	00
No. of Points Hi	00
No. of points Lo	16
Error check (LRC or CRC)	--

The coil status is returned as one coil per bit of the data field. Coils 1 to 16 are addressed as 0 to 15. Response for example 1 follows:

	Hex	Binary 8765 4321
Slave address	11	
Function	01	
Byte Count	02	
Data (Coils 7-0)	CD	1100 1101
Data (Coils 15-8)	6B	0110 1011
Error check (LRC or CRC)	--	

The status of coils 7-0 is CD hex, or 1100 1101 binary. Coil 15 is the Most Significant Bit (MSB). The CR510's C1 is on, C2, C5, and C6 are off. Flags 1,2,4,6, and 7 are high.

Example 2: Function 3 to read registers 40009 to 40012 from slave device 17 to retrieve input locations 5 and 6 from the CR10(X)

	Hex
Slave address	x11
Function	03
Starting Address Hi	00
Starting Address Lo	09
No. of Points Hi	00
No. of points Lo	04
Error check (LRC or CRC)	--

The register data is returned as two bytes per register and two registers per input location. Response for example 2:

	Hex
Slave address	11
Function	03
Byte Count	08
Data Hi (Register 40009)	C0
Data Lo (Register 40009)	00
Data Hi (Register 40010)	44
Data Lo (Register 40010)	0A
Data Hi (Register 40011)	00
Data Lo (Register 40011)	00
Data Hi (Register 40012)	1F
Data Lo (Register 40012)	00
Error check (LRC or CRC)	--

The contents of 40008 and 40009 are C0 00 44 0A hex or a value of 555.000 in input location 5. Input location 6 = 0.

For complete documentation please refer to the Modicon publication referenced above.

APPENDIX G. TD OPERATING SYSTEM ADDENDUM FOR CR510, CR10X, AND CR23X MANUALS

TD OPERATING SYSTEM ADDENDUM FOR CR510, CR10X, AND CR23X MANUALS

REVISION: 1/03

TD and PakBus Operating System Addendum for CR510, CR10X, and CR23X Manuals

AD1 Major Differences

Table Data (TD) operating systems have two major differences from the standard operating systems: First - the namesake - in the way data are stored internally and second, in the options available for transferring that data to external devices. The standard operating systems support both on site external storage (i.e., storage modules) that may be manually retrieved and telecommunications. The TD operating systems have more advanced telecommunication and networking capabilities but do not support storage modules. There are two versions of the TD operating system: TD and PakBus. The PakBus operating system includes the PakBus communications protocol that allows some additional communications options (Section 12); other features are the same as the TD operating system.

The datalogger hardware and direct measurement capabilities are the same in either case.

Feature	Table Data OS	Standard OS
Internal Data Storage	Multiple Data Tables, at least one Table for each output interval.	One or two Final Storage Areas. Data Arrays output at different intervals may share the same area and are identified by ID.
Term for a set of values output together	Record (a row of the table)	Array
Term for individual values within the array or record	Field (a column of the table)	Element
Number of elements or fields in an array or record	A fixed number, determined by the program	Generally fixed but conditional elements possible. Determined by program
Conditional Output	Yes	Yes
TimeStamps	Automatic	Optional
At Site / Manual Data Transfer	**No**	**Yes**
Storage Module Support	**No**	**Yes**
Instruction 96 – Serial Output	**No**	**Yes**
Instruction 98 – Send Character	**No**	**Yes**
Telecommunications	Commands protocol with error checking on all commands and responses. Feedback to confirm commands have been accepted.	Simple commands with error checking on data sent from datalogger to computer
Error Checking for data	Yes	Yes
Confirmation for Commands	**Yes**	**No**
Data Advise – data automatically sent (speeds multi-hop RF)	**Yes**	**No**
PakBus packet switching.	**Yes – in PakBus**	**No**
Supports Wireless Sensor	**Yes – in PakBus**	**No**
Support for Satellite Transmitters	**No**	**Yes**
Analog Measurements	No Difference	No Difference
Pulse Measurements	No Difference	No Difference

AD2 Overview of Data Storage Tables

Within a data table, data is organized in records and fields. Each row in a table represents a record and each column represents a field. To understand the concept of tables it may be helpful to consider an example. A CR10-TD is to be used to monitor 3 thermocouples (TC). Each hour a temperature for each of the three TC is to be stored. The table has 4 fields : "DATE_TIME TEMP1 TEMP2 TEMP3". Each hour a new "record" would be added. The "hourly" table would then be organized as follows:

DATE TIME	TEMP1	TEMP2	TEMP3
01/27/91 10:00:00	23.5	24.6	28.2
01/27/91 11:00:00	24.2	22.4	23.4

Only the hourly data is stored in the hourly table, Each output interval has its own table. Data tables can also be "event driven" rather than interval driven, that is a new record is stored when a specified event occurs rather than based on time. Each table is completely independent of any other tables and all records in a given table have the same number of fields.

The TD operating system supports naming of tables and fields, so any data value can be referenced by the table and field names. For example, the temperature data for the first thermocouple is referenced as "HOURLY.TEMP1". Computer software also allows the station to be named. When multiple dataloggers are in use, this can be used to reference specific data in the network. If, in the previous example, the CR10T site was named DALLAS, the first thermocouple's data values would be referenced by "DALLAS.HOURLY.TEMP1".

AD3 Converting an existing program to Table Based OS

This section is intended for those familiar with programming an Array based datalogger.

AD3.1 Programming changes

- Remove all Record Real Time instructions (Instruction 77).
- Remove all Serial Data Output and Serial Print instructions (Instructions 96 and 98).
- Remove all Initiate Telecommunication (callback) instructions (Instruction 97).
- Check all instructions which set the Output Flag (Flag 0). These should be replaced with the Data Table Instruction (Instruction 84). If the Set Active Storage Area instruction (Instruction 80) is used, it should be removed as Instruction 84 provides this functionality.
- Check all If Time Instructions (Instruction 92) as the units may change from minutes to seconds. Any instruction 92 that sets the output flag (Flag 0) is replaced by Instruction 84.
- Check the Move Time To Input Location Instruction (Instruction 18) as some parameters have changed.

- Check the Maximum and Minimum Instructions (Instructions 73 and 74) as there is only one option to store time with the value.

- Edit Input Location labels removing all spaces and special characters. Only letters, numbers, and the "_" characters are allowed. Labels should start with a letter.

- Add labels for the Final Storage values. Use the same character as are allowed for Input Location labels. See Section 2.1

AD3.2 Making the Changes with Edlog

Programs for Array based logger can be converted to Table Based using EDLOG for most of the editing by doing the following:

1. Make a copy of the original program with the name you want the new program to have: Load the original into Edlog and "Save As" the new name.

2. Remove or comment out all Instructions 77, 96, 97, and 98. (first three points in AD3.1, these instructions are not in the Table OS)

3. Save the edited program and close it in Edlog.

4. Edit the CSI file with a text editor (e.g., "Notepad" - Edlog will not allow you to make and save this change) and add –TD to the datalogger type on the first line, for example, change:
   ;{CR10X}
   to:
   ;{CR10X-TD}.
   Save the CSI file and close the editor.

5. Open the file with Edlog. Edlog should now recognize that the program is for a table data OS.

6. Add Instructions 84 where necessary and make the other necessary changes.

AD4 Summary of Differences from the Datalogger Manual:

Section	Differences
Overview	**Figure OV2.1-2: See Figure 1.5-1 in Addendum.** **Table OV3.2-1: See Table OV4.1-1 in TD Addendum.** **OV4, OV5, OV6 :See TD Addendum.**
Section 1	**Section 1.5 A Mode** is replaced by addendum – the TD loggers allocate memory differently. **Section 1.8 - *D Mode** is replaced by TD Addendum – TD loggers do not support storeing multiple programs or Storage Modules. PakBus Settings are added to the *D Mode.
Section 2	**Replaced entirely** by TD Addendum.
Section 3	**Section 3.7.1** does not apply to the **TD operating system** which **does not use Output Flag 0.** **Table 3.8-1** Valid Flag Commands are 11 – 19 to set high and 21- 29 to set Low. Because the TD operating system **does not use Flag 0, Commands 10 and 20 are not valid with the TD operating system**. **Table 3.10-1** TD Addendum. has a corrected version
Section 4	**Does not apply:** The TD operating system does not support External Storage Peripherals.
Section 5	**Does not apply:** The communications commands and protocol of the TD operating system is different than that of the standard operating systems. Campbell Scientific provides software for communications; a description of the protocol is beyond the scope of this addendum.
Section 6	Many of the peripherals discussed in section 6 are not supported by the TD operating system.
Section 7	**No Change**
Section 8	**Replaced entirely** by TD Addendum.
Section 9	**Instruction 18** has some differences in the time options, see addendum.
Section 10	**No Change**
Section 11	**Instructions 73 and 74** have only one option for storing the time of max or min (time is output as a quoted string). **Instruction 80** – Set Active Storage Area, is not in the TD operating system. Its functions are included in Instruction 84 – Data Table. **Instruction 84** – Data Table, sets the conditions and destination for output data. This instruction is only in the TD operating system (see TD Addendum..)

Section 12	**The TD operating system does not use the output Flag 0**. Commands dealing with it are not valid. **Instruction 92** – There is no option for minutes, **time is in seconds only**. **Instructions Not In TD OS**: **Instruction 96 – Serial Output** **Instruction 98 – Send Character** **Instruction 111 – Load Program from Flash** **New Instructions for PakBus:** **Instruction 190 – Send or Get Input Locations** **Instruction 191 – One way Final Storage Data Transfer** **Instruction 192 – PakBus Message** **Instruction 193 – Wireless Network Master Control** **Instruction 194 – Time Until Transmit** **Instruction 195 – Set Clock from Address** **Instruction 196 – Wireless Remote** **Instruction 197 – Force Route Through Address**
Section 13	**No Change**
Section 14	**No Change**

MEASUREMENT AND CONTROL MODULE OVERVIEW

While this section of the addendum references the CR10X, everything but the measurement instructions in the example programs applies to the other dataloggers as well.

Table OV3.2-1 in the CR10X Manual is incorrect for the TD operating system. See Table OV4.1-1 below.

The following sections OV4, OV5, and OV6 replace those in the CR10X Manual.

OV4. PROGRAMMING THE CR10X

A program is created by entering it directly into the datalogger or into a computer using the LOGGERNET program EDLOG. This manual describes direct interaction with the CR10X. Work through the direct programming examples in this overview before using EDLOG and you will know the basics of CR10X operation as well as an appreciation for the help provided by the software. Section OV4.5 describes options for loading the program into the CR10X.

OV4.1 FUNCTIONAL MODES

CR10X/User interaction is broken into different functional MODES (e.g., programming the measurements and output, setting time, etc.). The modes are referred to as Star (*) Modes since they are accessed by first keying *, then the mode number or letter. Table OV4.1-1 lists the CR10X Modes.

TABLE OV4.1-1. * Mode Summary

Key	Mode
*0	LOG data and indicate active Tables
*1	Program Table 1
*2	Program Table 2
*3	Program Table 3, subroutines only
*5	Display/set real time clock
*6	Display/alter Input Storage data, toggle flags and ports
*7	Display Data Storage Table data
*9	Display Data Storage Table sizes
*A	Memory allocation/reset
*B	Signature/status
*C	Security

OV4.2 KEY DEFINITION

Keys and key sequences have specific functions when using the CR10KD keyboard or a computer in the remote keyboard state (Section 5). Table OV4-2 lists these functions. In some cases, the exact action of a key depends on the mode the CR10X is in and is described with the mode in the manual.

When using a computer/terminal to communicate with the CR10X (Telecommunications) there are some keys available in addition to those found on the CR10KD. Table OV4.2-2 lists these keys.

TABLE OV4.2-1. Key Description/Editing Functions

Key	Action
0-9	Key numeric entries into display
*	Enter Mode (followed by Mode Number)
A	Enter/Advance
B	Back up
C	Change the sign of a number or index an input location to loop counter
D	Enter the decimal point
#	Clear the rightmost digit keyed into the display
#A	Advance to next instruction in program table (*1, *2, *3)
#B	Back up to previous instruction in program table.
#D	Delete entire instruction

TABLE OV4.2-2. Additional Keys Allowed in Telecommunications

Key	Action
-	Change Sign, Index (same as C)
CR	Enter/advance (same as A)

OV4.3 PROGRAMMING SEQUENCE

In routine applications, the CR10X measures sensor output signals, processes the measurements over some time interval and stores the processed results. A generalized programming sequence is:

1. Enter the execution interval. In most cases, the execution interval is determined by the desired sensor scan rate.

2. Enter the Input/Output instructions required to measure the sensors.

3. If processing in addition to that provided by the Output Processing Instructions (step 5) is required, enter the appropriate Processing Instructions.

4. Enter the Data Table Instruction 84 to test the output condition and output when the condition is met. For example, use

   Instruction 84 to output based on time.

   Instruction 84 to output every execution interval.

   Instruction 84 to output based on a Program Flag.

   This instruction must precede the Output Processing Instructions which store data in a Data Storage Table. Instructions are described in Sections 9 through 12.

5. Enter the Output Processing Instructions to store processed data in the Data Storage Table. The order in which data are stored is determined by the order of the Output Processing Instructions in the table.

6. Repeat steps 4 through 6 for additional outputs on different intervals or conditions.

> **NOTE**: The program must be executed for output to occur. Therefore, the interval specified with the Data Table Instruction is set must be evenly divisible by the execution interval. For example, with a 2 minute execution interval and a 5 minute output interval, the program will only be executed on the even multiples of the 5 minute intervals, not on the odd. Data will be output every 10 minutes instead of every 5 minutes.

Execution intervals are synchronized with midnight. Output intervals set with Instruction 84 are synchronized with real time starting at midnight, January 1, 1990.

OV4.4 INSTRUCTION FORMAT

Instructions are identified by an instruction number. Each instruction has a number of parameters that give the CR10X the information it needs to execute the instruction.

The CR10X Prompt Sheet has the instruction numbers in red, with the parameters briefly listed in columns following the description. Some parameters are footnoted with further description under the "Instruction Option Codes" heading.

For example, Instruction 73 stores the maximum value that occurred in an Input Storage location over the output interval. The instruction has three parameters (1) REPetitionS, the number of sequential Input Storage locations on which to find maxima, (2) TIME, an option of storing the time of occurrence with the maximum value, and (3) LOC the first Input Storage location operated on by the Maximum Instruction. The codes for the TIME parameter are listed in the "Instruction Option Codes".

The repetitions parameter specifies how many times an instruction's function is to be repeated. For example, four 107 thermistor probes may be measured with a single Instruction 11, Temp-107, with four repetitions. Parameter 2 specifies the input channel of the first thermistor (the probes must be connected to sequential channels). Parameter 4 specifies the Input Storage location in which to store measurements from the first thermistor. If location 5 were used and the first probe was on channel 1, the temperature of the thermistor on channel 1 would be stored in input

location 5, the temperature from channel 2 in input location 6, etc.

Detailed descriptions of the instructions are given in Sections 9-12. Entering an instruction into a program table is described in OV5.

OV4.5 ENTERING A PROGRAM

Programs are entered into the CR10X in one of two ways:

1. Keyed in using the CR10X keyboard
2. Stored on disk/seat from computer

A program is created by keying it directly into the datalogger as described in Section OV5, or on a PC using EDLOG.

EDLOG is used to develop programs for Campbell Scientific CR10X dataloggers. EDLOG is a prompting editor for writing and documenting programs for Campbell Scientific CR10X dataloggers. Program files developed with EDLOG can be downloaded directly to the CR10X using NetAdmin. NetAdmin supports communication via direct wire, telephone, or Radio Frequency (RF).

OV5. PROGRAMMING EXAMPLES

We will start with a simple programming example. There is a brief explanation of each step to help you follow the logic. When the example uses an instruction, find it on the Prompt Sheet and follow through the description of the parameters. Using the Prompt Sheet while going through these examples will help you become familiar with its format. Sections 9-12 have more detailed descriptions of the instructions.

With the Wiring Panel connected to the CR10X, hook up the power leads as described in Section OV1.2. Next, connect the CR10X to either a CR10KD Keyboard/Display or the computer (Section OV3). The programming steps in the following examples use the keystrokes possible on the keyboard/display. With a terminal, some responses will be slightly different.

If the CR10KD is connected to the CR10X when it is powered up, the display will show:

Display	Explanation
HELLO	On power-up, the CR10X displays "HELLO" while it checks the memory (this display occurs only with the CR10KD).

after a few seconds delay

:96	The size of the machine's total memory (RAM plus 32 K of ROM), in this case 96K

OV5.1 SAMPLE PROGRAM 1

In this example the CR10X is programmed to read its own internal temperature (using a built in thermistor) every 5 seconds and to send the results to Final Storage.

Key	Display	Explanation
*	00:00	Enter mode.
1	01:00	Enter Program Table 1.
A	01:0.0000	Advance to execution interval (In seconds)
5	01:5	Key in an execution interval of 5 seconds.
A	01:P00	Enter the 5 second execution interval and advance to the first program instruction location.
17	01:P17	Key in Instruction 17 which directs the CR10X to measure the internal temperature in degrees C. This is an Input/Output Instruction.
A	01:0000	Enter Instruction 17 and advance to the first parameter.
1	01:1	The input location to store the measurement, location 1.
A	02:P00	Enter the location # and advance to the second program instruction.

The CR10X is now programmed to read the internal temperature every 5 seconds and place the reading in Input Storage Location 1. The program can be compiled and the temperature displayed.

Key	(ID:Data)	Explanation
*0	LOG 1	Exit Table 1, enter *0 Mode, compile table and begin logging.
*6	06:0000	Enter *6 Mode (to view Input Storage).
A	01:21.234	Advance to first storage location. Panel temperature is 21.234°C (the display will show the actual temperature).
Wait a few seconds:		
	01:21.423	The CR10X has read the sensor and stored the result again. The internal temp is now 21.423°C. The value is updated every 5 seconds when the table is executed. At this point the CR10X is measuring the temperature every 5 seconds and sending the value to Input Storage. No data are being saved. The next step is to have the CR10X send each reading to Final Storage.
*1	01:00	Exit *6 Mode. Enter program table 1.
2A	02:P00	Advance to 2nd instruction location (this is where we left off).
84	02:P84	This is the Data Table Instruction.
A	01:0.0000	Enter 84 and advance to the first parameter (which is the time into the interval).
0	01:0	This parameter determines when in the output internal data is stored. 0 stores data on the even interval.
A	02:0.000	Enter 0 and advance to the second parameter.
0	02:0	This parameter specifies the output interval. 0 stores data each execution.
A	03:0.0000	Enter 0 and advance to third parameter.
1000	03:1000.00	This parameter specifies how many records to store in the table before overwriting the oldest. For this example, keep 1000 records.
A	03:P00	Enter 1000 and advance to the third program instruction.
70	03:P70	The SAMPLE instruction. It directs the CR10X to take a reading from an Input Storage location and send it to Final Storage (an Output Processing Instruction).
A	01:0000	Enter 70 and advance to the first parameter (repetitions).
1	01:1	There is only one input location to sample; repetitions = 1.
A	02:0000	Enter 1 and advance to second parameter (Input Storage location to sample).
1	02:1	Input Storage Location 1, where the temperature is stored.
A	04:P00	Enter 1 and advance to fourth program instruction.
*	00:00	Exit Table 1.
0	LOG 1	Enter *0 Mode, compile program, log data.

OV5.2 SAMPLE PROGRAM 2

This second example is more representative of a real-life data collection situation. Once again the internal temperature is measured, but it is used as a reference temperature for the differential voltage measurement of a type T (copper-constantan) thermocouple; the CR10X should have arrived with a short type T thermocouple connected to differential channel 5.

When using a type T thermocouple, the copper lead (blue) is connected to the high input of the differential channel, and the constantan lead (red) is connected to the low input.

A thermocouple produces a voltage that is proportional to the difference in temperature between the measurement and the reference junctions.

To make a thermocouple (TC) temperature measurement, the temperature of the reference junction (in this example, the approximate panel temperature) must be measured. The CR10X takes the reference temperature, converts it to the equivalent TC voltage relative to 0°C, adds the measured TC voltage, and converts the sum to temperature through a polynomial fit to the TC output curve (Section 13.4).

The internal temperature of the CR10X is not a suitable reference temperature for precision thermocouple measurements. It is used here for the purpose of training only. To make thermocouple measurements with the CR10X, purchase the Campbell Scientific Thermocouple Reference, Model CR10XCR (Section 13.4) and make the reference temperature measurement with Instruction 11.

Instruction 14 directs the CR10X to make a differential TC temperature measurement. The first parameter in Instruction 14 is the number of times to repeat the measurement. Enter 1, because in this example there is only one thermocouple. If there were more than 1 TC, they could be wired to sequential channels, and the number of thermocouples entered for repetitions. The CR10X would automatically advance through the channels sequentially and measure all of the thermocouples.

Parameter 2 is the voltage range to use when making the measurement. The output of a type T thermocouple is approximately 40 microvolts per degree C difference in temperature between the two junctions. The ±2.5 mV scale will provide a range of ±2500/40 = ±62.5°C (i.e., this scale will not overrange as long as the measuring junction is within 62.5°C of the panel temperature). The resolution of the ±2.5 mV range is 0.33 µV or 0.008°C.

Parameter 3 is the analog input channel on which to make the first, and in this case only, measurement. Parameter 4 is the code for the type of thermocouple used. This information is located on the Prompt Sheet or in the description of Instruction 14 in Section 9. The code for a type T (copper-constantan) thermocouple is 1.

Parameter 5 is the Input Storage location in which the reference temperature is stored. Parameter 6 is the Input Storage location in which to store the measurement (or the first measurement; e.g., if there are 5 repetitions and the first measurement is stored in location 3, the final measurement will be stored in location 7). Parameters 7 and 8 are the multiplier and offset. A multiplier of 1 and an offset of 0 outputs the reading in degrees C. A multiplier of 1.8 and an offset of 32 converts the reading to degrees F.

In this example, the sensor is measured once a minute, and the average temperature is output every hour. Once a day the maximum and minimum temperatures and the times they occur will be output.

The first example described program entry one keystroke at a time. This example does not show the "A" key. Remember, "A" is used to enter and/or advance (i.e., between each line in the example below). This format is similar to the format used in EDLOG.

It's a good idea to have both the manual and the Prompt Sheet handy when going through this example. You can find the program instructions and parameters on the Prompt Sheet and can read their complete definitions in the manual.

To obtain daily output, the Data Table instruction is followed by the Output Instructions to store the daily maximum and minimum temperatures and the time each occurs.

SAMPLE PROGRAM 2

Instruction # (Loc:Entry)	Parameter (Par#:Entry)	Description
*1		Enter Program Table 1
01:60		60 second (1 minute) execution interval
Key "#D" repeatedly until is displayed	01:P00	Erase previous Program before continuing.
01:P17		Measure internal temperature
	01:1	Store temp in Location 1
02:P14		Measure thermocouple temperature (differential)
	01:1	1 repetition
	02:1	Range code (2.5 mV, slow)
	03:5	Input channel of TC
	04:1	TC type: copper-constantan
	05:1	Reference temp is stored in Location 1
	06:2	Store TC temp in Location 2
	07:1	Multiplier of 1
	08:0	No offset
03:P84		Data Table Instruction
	01:0	0 seconds into the interval
	02:3600	3600 second (60 min.) internal
	03: 0	Automatically allocate # of records

The CR10X is programmed to measure the thermocouple temperature every sixty seconds. The CR10X automatically allocates the number of records. Time information is automatically stored. Next the output instruction for the average is added.

Instruction # (Loc.:Entry)	Parameter (Par.#:Entry)	Description
04:P71		Average instruction
	01:1	One repetition
	02:2	Location 2 - source of TC temps. to be averaged
05:P84		Data Table Instruction
	01:0	0 seconds into the interval
	02:86400	86400 second interval (24 hrs.)
	03:0	Automatically allocate # of records
06: P73		Maximize instruction
	01:1	One repetition
	02:1	Output the time of the daily maximum
	03:2	Data source is Input Storage Location 2.
07: P74		Minimize instruction
	01:1	One repetition
	02:1	Output the time of the daily minimum
	03:2	Data source is Input Storage Location 2.

The program to make the measurements and send the desired data to Final Storage has been entered. The program is complete. The clock must now be set so that the date and time tags are correct. (Here the example reverts back to the key by key format.)

Key	Display	Explanation
*5	00:21:32	Enter *5 Mode. Clock running but not set correctly.
A	05:01.01	Advance to month-day (MMDD).
1004	05:1004	Key in MMDD (Oct 4 in this example).
A	05:1990	Enter and advance to location for year.
1994	05:1994	Key in year.
A	05:00:21	Enter and advance to location for hours and minutes (24 hr. time).
1324	05:1324	Key in hrs.:min. (1:24 PM in this example).
A	05:27.250	Key in seconds
30	05:30	
A	13:24:30	Clock set and running. Changes made when A was pressed.
*0	LOG 1	Exit *5, compile Table 1, commence logging data.

OV5.3 EDITING AN EXISTING PROGRAM

When editing an existing program in the CR10X, entering a new instruction inserts the instruction; entering a new parameter replaces the previous value.

To insert an instruction, enter the program table and advance to the position where the instruction is to be inserted (i.e., P in the data portion of the display) key in the instruction number, and then key A. The new instruction will be inserted at that point in the table, advance through and enter the parameters. The instruction that was at that point and all instructions following it will be pushed down to follow the inserted instruction.

An instruction is deleted by advancing to the instruction number (P in display) and keying #D (Table 4.2-1).

To change the value entered for a parameter, advance to the parameter and key in the correct value then press A. Note that the new value is not entered until A is keyed.

OV6. DATA RETRIEVAL OPTIONS

Data is retrieved over some form of telecommunications link, whether it be RF (radio), telephone, short haul modem, coaxial cable (mulitdrop) , or direct link. The table data operating system does not support on-line output to peripheral storage devices (see Figure OV 6.1-1).

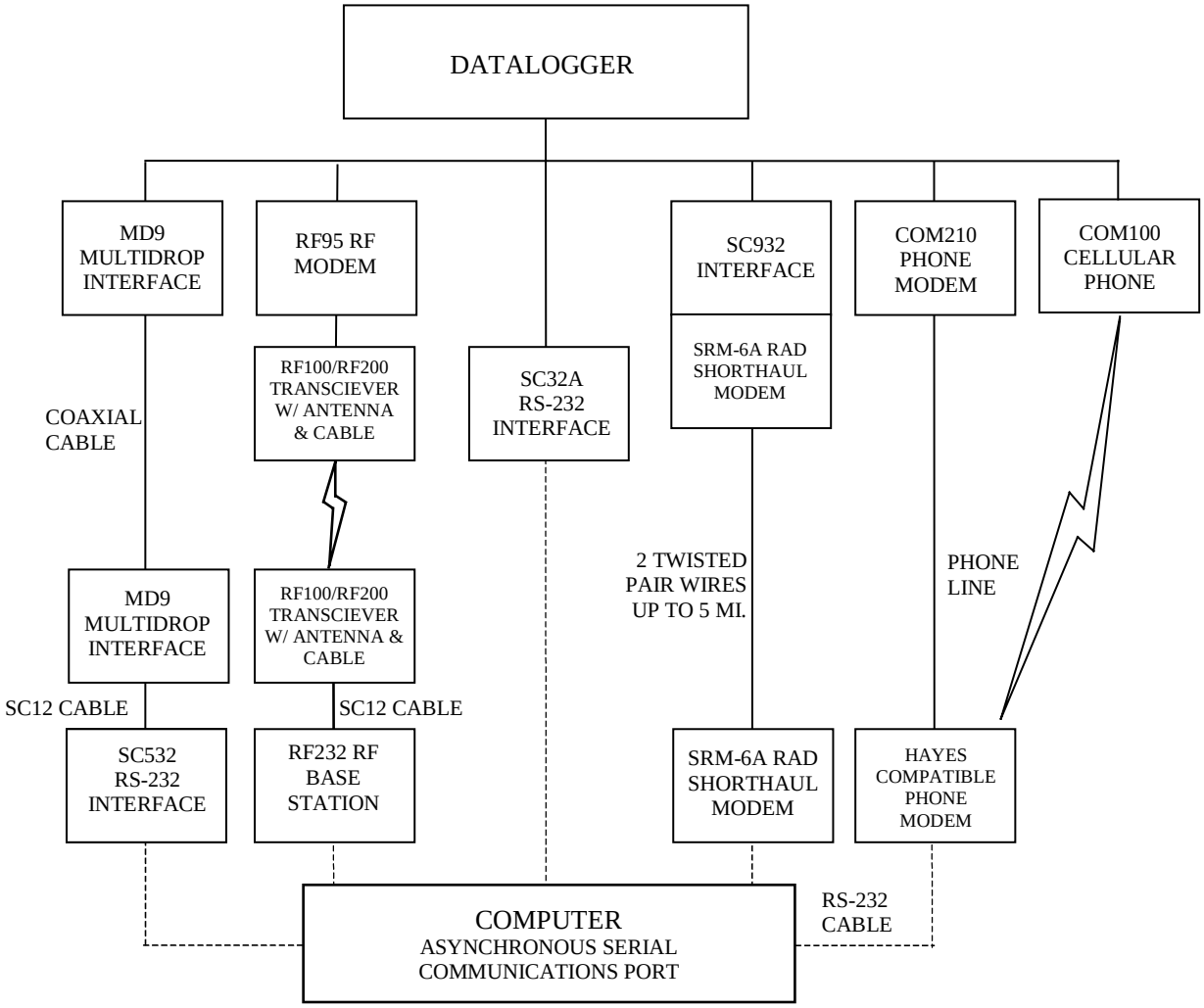
The retrieval of data does NOT erase those data from the Data Storage Tables in Final Storage. The data remains in the table ring memory until:

The are written over by new records of data. (Section 2.1)

Input Storage memory is reallocated (Section 1.5)

The datalogger program is changed and compiled.

Power to the datalogger is turned off.



NOTES:

1. ADDITIONAL METHODS OF DATA RETRIEVAL ARE:
   A. SATELLITE TRANSMISSION
   B. DIRECT DUMP TO PRINTER
   C. VOICE PHONE MODEM TO VOICE PHONE OR PC WITH HAYES COMPATIBLE PHONE MODEM

2. THE DSP4 HEADS UP DISPLAY ALLOWS THE USER TO VIEW DATA IN INPUT STORAGE. ALSO BUFFERS FINAL STORAGE DATA AND WRITES IT TO CASSETTE TAPE, PRINTER OR STORAGE MODULE.

3. ALL CAMPBELL SCIENTIFIC RS-232 INTERFACES HAVE A FEMALE 25 PIN RS-232 CONNECTOR.

FIGURE OV6.1-1. Data Retrieval Hardware Options

SECTION 1. FUNCTIONAL MODES

Sections 1.5 and 1.8 are replaced by the following sections.

1.5 MEMORY ALLOCATION - ∗A

1.5.1 INTERNAL MEMORY

When powered up with the keyboard display attached, the CR10KD displays HELLO while performing a self check. The total system memory is then displayed in K bytes. The size of memory can be displayed in the ∗B mode.

Input Storage is used to store the results of Input/Output and Processing Instructions. The values stored in input locations may be displayed using the ∗6 Mode (Section 1.3).

Final Storage holds stored data for a permanent record. Output Instructions store data in Final Storage Data Tables. The data in Final Storage can be monitored using the ∗7 Mode (Section 2.3).

Intermediate Storage is a scratch pad for Output Processing Instructions. It is used to store the results of intermediate calculations necessary for averages, standard deviations, histograms, etc. Intermediate Storage is not accessible by the user.

Each Input or Intermediate Storage location requires 4 bytes of memory. Each Final Storage location requires 2 bytes of memory. Low resolution data points require 1 Final Storage location and high resolution data points require 2. Section 2 describes Final Storage and data retrieval in detail.

Figure 1.5-1 lists the basic memory functions and the amount of memory allotted to them.

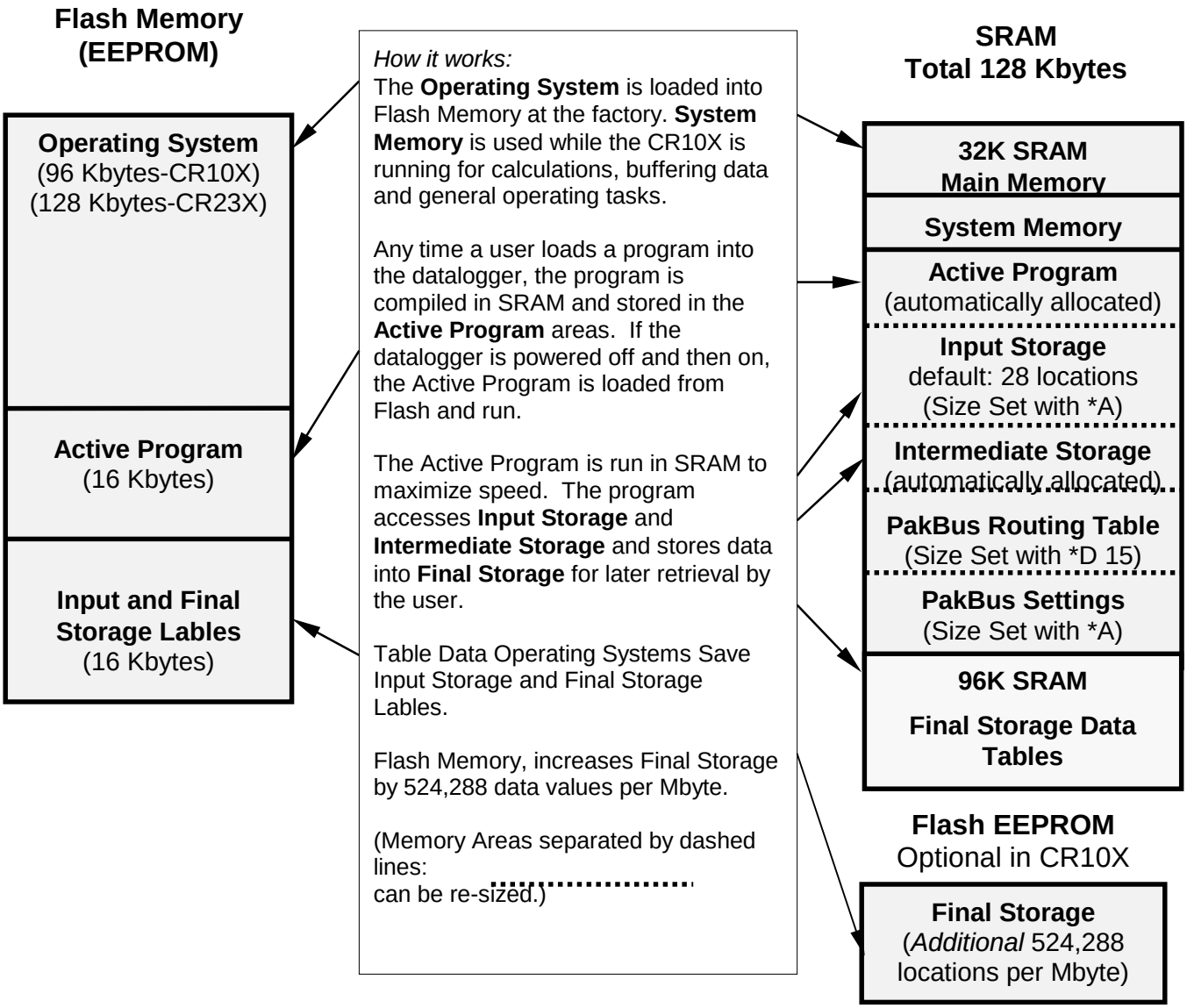


FIGURE 1.5-1. Datalogger Memory

1.5.2 ∗A MODE

The ∗A Mode is used to 1) check the size of Input Storage, Intermediate Storage, Final Storage, Program Memory; PakBus and user Settings memory 2) check the number of bytes remaining in Flash Program memory; Main Memory, and Label Memory 3) change the memory allotted to Input Locations and Settings; and 5) to completely reset the datalogger.

When ∗A is entered, the first number displayed is the number of memory locations allocated to Input Storage. The "A" key is used to advance through the next 6 windows. Table 1.5-2 describes what the values in the ∗A Mode represent.

The sizes of Input Storage and Settings Memory may be altered by keying in the desired value and entering it by keying "A".

TABLE 1.5-2. Description of ∗A Mode Data

Keyboard Entry	Display ID: Data	Description of Data
* A	01: XXXX	**Input Storage Locations** (minimum of 28, maximum of 6655, but the usable maximum is less than this because intermediate and program storage require some of this memory). This value can be changed by keying in the desired number.
A	02: XXXX	**Intermediate Storage Locations** (maximum limited by available memory and constraints on Input and Final Storage). The CR10X-TD will assign the exact number needed for the active program. The CR10X-TD erases all data whenever the program is changed and compiled.
A	03: XXXXX	**Final Storage Locations** (minimum of 0, maximum limited by available memory). Changing this number automatically reallocates Final Storage Area 1.
A	04: XXXXX	**Bytes allocated for user program**. The CR10X-TD will assign the exact number needed. The CR10X-TD erases all data whenever the program is changed and compiled. **Key in 98765 to completely reset datalogger.**
A	05:	**Bytes free in Flash Memory for active program.** The user cannot change this window. It is a function of window 5 and the program.
A	06:	**PakBus and user Settings memory**
A	07:	Main Memory Free
A	06:	Label Bytes Free

The maximum size of of Final Storage is determined by the memory installed (Table 1.5-1). The size of Final Storage and the rate at which data are stored determines how long it will take for Final Storage to fill, at which point new data will write over old.

Twenty-eight is the minimum number of Input locations allowed. Intermediate Storage and Final Storage are erased when the number of Input locations is changed. This feature may be used to clear memory without altering programming. The number of locations does not actually need to be changed; the same value can be keyed in and entered.

Intermediate Storage and Program Memory are automatically allocated. All data are erased any time the program is changed and compiled. If there is not enough memory available in the 32K Main Memory for the Intermediate Storage required by the current program, the "E:04" ERROR CODE will be displayed in the ∗0, ∗6, and ∗B Modes.

After repartitioning memory, the program must be recompiled. Compiling erases Intermediate Storage. Compiling with ∗0 erases Input Storage; compiling with ∗6 leaves Input Storage unaltered (If its size was unchanged).

ENTERING 98765 in the program memory window 6 COMPLETELY RESETS THE CR10X. All memory is erased including the program and memory is checked. Memory allocation returns to the default. The reset operation requires approximately 1 minute for a CR10X, 5 minutes for a CR10X-1M, and 10 minutes for a CR10X-2M. Please be patient while the reset takes place; if the CR10X is turned off in the middle of a reset, it will perform the reset the next time it is powered up.

1.6 MEMORY TESTING AND SYSTEM STATUS - ∗B

No changes from standard operating system, see datalogger manual.

1.7 ∗C MODE -- SECURITY

No changes from standard operating system, see datalogger manual.

1.8 ∗D MODE – TRANSFER PROGRAMS, GENERAL SETTINGS

The ∗D Mode is used to transfer datalogger programs between a datalogger and a computer, to erase a program, to set the degree

to which memory is cleared on powerup, to set the PakBus ID, and to set communication to full or half duplex.

CSI datalogger support software makes use of the ∗D Mode to upload and download programs from a computer. Appendix C gives some additional information on Commands 1 and 2 that are used for these operations.

When "∗D" is keyed in, the CR10X will display "13:00". A command (Table 1.8-1) is entered by keying the command number and "A".

TABLE 1.8-1. ∗D Mode Commands

Command	Description
1	Send (Print) ASCII Program
2	Load ASCII Program, ∗0 Compile
2--	Load ASCII Program, ∗6 Compile
3	# Rings Before Answering Phone
7	Erase Current Program
10	Set Powerup Options
12	Set Initial Baud
15	PakBus Address/Routing Table
16	Memory for General Purpose File
17	PakBus Routing Table
18	PakBus Beacon Interval
19	PakBus Neighbor List

If the CR10X program has not been compiled when the command to save a program is entered, it will be compiled before the program is saved. When a program is loaded, it is immediately compiled and run. When a command is complete, "13:0000" is displayed; ∗D must be entered again before another command can be given.

TABLE 1.8-2. Program Load Error Codes

E 94	Program Storage Area full
E 95	Program does not exist in flash
E 96	Storage Module not connected or wrong address
E 97	Data not encountered within 30 sec.
E 98	Uncorrectable errors detected
E 99	Wrong type of file or Editor Error

1.8.1 ERASING CURRENT PROGRAM

The 7 command may be used to delete the current program as show in Table 1.8-3.

TABLE 1.8-3 Deleting Current Datalogger Program

Key entry	Display
∗D	13:00
7A	07:00

You may now enter:

0A	Erase active program (i.e., load a blank program; memory allocation and Final Storage are reset).

1.8.2 PROGRAM TRANSFER WITH STORAGE MODULE

Not supported in Table Data Operating Systems.

1.8.3 FULL/HALF DUPLEX

Not supported in Table Data Operating Systems.

1.8.4 SET DATALOGGER ID

Command 8 not supported in Table Data Operating Systems.

1.8.5 SETTING POWERUP OPTIONS

Setting options for the Program on Powerup allows the user to specify what information to retain from when the datalogger was last on. This allows Flag/Port status, the User Timer, and the Input/Intermediate Storage to be cleared or not cleared.

Table 1.8-8. Setting Powerup Options

Key entry	Display
∗D	13:00
10A	10:0X

Where X is the powerup option currently selected. You may now change the option:

0A	Clears input locations, ports, flags, user timer, and intermediate storage locations.
1A	Clears intermediate storage only (leaves Input Storage, Flags/Ports, and User Timer as is).
2A	Doesn't clear anything.

1.8.6 SET INITIAL BAUD

Table 1.8-10 shows the option codes available for setting the initial baud rate. Setting the initial baud rate forces the CR10X to try the selected baud rate first when connecting with a device.

TABLE 1.8-9. Set Initial Baud Rate / Set RS232 Power

Key Entry	Display	Comments
*D	13:00	Enter Command
12A	12:00	Connect Baud Rate Enter Baud Rate Code X (Table 1.8-11).

TABLE 1.8-10. Baud Rate Codes

X = 0	300 Baud
X = 1	1200 Baud
X = 2	9600 Baud
X = 3	76.8 K Baud

1.8.7 SET PROGRAM COMPILE OPTION

Command 13 is not supported in Table Data operating systems.

1.8.8 SET PAKBUS ADDRESS

*D 15 allows the user to set the PakBus Address of the datalogger and to set the maximum size for its routing table.

TABLE 1.8-11. PakBus Address and Routing Table

Key Entry	Display	Comments
*D	13:00	Enter Command
15A	15:xxxx	PakBus Address , Enter zero if the datalogger is not to be used as a PakBus device (1..4094 is legal, the default is 1)
A	01:xxxx	If the datalogger is to be used a a router, enter the maximum number of nodes (PakBus Addresses) to allocate space for in the pakbus network. 0 = leafnode, <>0 = router
A	02:xxxx	Enter the maximum number of neighbors in the pakbus network to allocate space for. This parameter is used only if datalogger is used as a router (01: is non-zero).
A	03:xxxx	Enter maximum number of routers in the pakbus network to allocate space for. This parameter is used only if datalogger is used as a router (01: is non-zero).
A	04:xxxx	Enter the PakBus address for a default router (1..4094, 0 for no default router). The default router is used for a message if the destination PakBus address is not in the routing table. A router discovering new routes will not explore beyond its own default router.

The memory for a routing table comes out of the pool for program, input locations, and intermediate storage.

The total number of bytes used for the routing table =

Nodes x 12
+ Neighbors x 8
+ Routers x 6
+ (Routers x (Nodes – Routers) + (Routers x (Routers – 1))/2) x 4

The *D15 entries are sent when the program is retrieved. They can also be set like other *D settings via the DLD file.

1.8.9 ALLOCATE MEMORY FOR GENERAL PURPOSE FILES

*D16:xx ;allocate xx 64K byte chunks of memory for general purpose files. The area comes out of final storage space. Files are stored in a circular buffer (ring memory) in this space.

1.8.10 VIEW ROUTING TABLE

*D17 allows viewing the current routing table information. This is view only.

TABLE 1.8-12. Values in Routing Table

Key Entry	Display	Comments
*D	13:00	Enter Command
17A		Enter the view routing table command
	01:xxxx	;pakbus address of destination node
	02:xxxx	;via neighbor with xxxx pakbus address
	03:xxxx	;a worst case response time metric (seconds)
(Repeats for next destination node.)		

1.8.11 SET PAKBUS ROUTER BEACON INTERVAL

TABLE 1.8-13. Set Beacon interval

Key Entry	Display	Comments
*D	13:00	Enter Command
18A		Enter the beacon interval settings
A	01:xxxx	Enter the Interval (seconds) for SDC7
A	02:xxxx	Enter the Interval (seconds) for SDC8
A	03:xxxx	Enter the Interval (seconds) for CS I/O Pin Enabled, 9600 baud
A	04:xxxx	Enter the Interval (seconds) for RS232, 9600 baud (CR23X only)

1.8.12 PAKBUS NEIGHBOR FILTER

In some networks, sending beacons can be disruptive. Entering values in the *D19 mode disables the beacon. A PakBus datalogger with nonzero *D19 settings will not send beacons and will only respond to beacons from nodes with addresses in the neighbor list.

Instead of sending beacons, the datalogger will send “hello” messages to neighbors in the list to determine if it can communicate with them. Neighbors (or potential neighbors) should be nodes that communicate directly with the datalogger without going through a router.

Note that this list is not automatically cleared by compiling a new program. (It may be changed if the new program contains *D19 entries.) It can be edited by changing entries. Once 0 is entered for a neighbor address, all entries beyond the 0 entry are cleared.

TABLE 1.8-14. Set PakBus Neighbors

Key Entry	Display	Comments
*D	13:00	Enter Command
19A	19:00	Port (17- SDC7, 18 – SDC8, 02 – CSI/O, 02—CR23X RS232 port, 9600 baud
A	19:0000	Interval in seconds of the expected rate of communication. A neighbor is aged after 2.5 times this interval and the Hello attempt will be reinitiated.
A	01:xxxx	PakBus Address of neighbor
A	01:xx	Swath of neighbors with sequential addresses starting with above address.
A	02:xxxx	PakBus Address of neighbors
A	02:xx	Swath of neighbors with sequential addresses starting with above address.
... etc.		
A	nn:0000	Terminates the list.

1.9 *9 DATA TABLES SIZES.

The *9 Mode is used to view the sizes of the Data Storage Tables (section 2.1) created by the datalogger program (*1, *2, or *3). The *9 Mode will also display how long until any automatically allocated Data Storage Tables fill. All Data Storage Tables are in a ring configuration such that the oldest records are overwritten by new records once the table is full. The sizes are given as the number of records. A record can be thought of as a row of data where each field (i.e., column) is a data value associated with an Output Processing Instruction. The order and number of fields in a Data Table are determined by the Output Processing Instructions following the Data Table Instruction. The tables are numbered in the order the Data Table Instruction appear in the Program Tables, *1 first, *2 second, and *3 last.

TABLE 1.9-1. Description of * 9 Data

Keyboard Entry	Display ID: Data	Description
*9	09:xx	Number of tables/ Enter table number to jump to that table or 0 to see next parameter.
0A	00:x.xxxx	Days before any automatically allocated table fills.
A	01:xxxxx.	Number of records in table 01
A	02:xxxxx	Number of records in table 02
A	nn:xxxxx.	Number of records in table nn

THIS SECTION ENTIRELY REPLACES THE DATALOGGER MANUAL SECTION 2.

SECTION 2. INTERNAL DATA STORAGE

2.1 FINAL STORAGE AND DATA TABLES

Final Storage is that portion of memory where final processed data are stored. It is from Final Storage that data is transferred to your computer. With the TD datalogger, Final Storage is organized into Data Storage Tables. These data tables should not be confused with the program tables *1, *2, and *3 that contain the datalogger program.

Within each data table, data is organized in records and fields. Each row in a table represents a record and each column represents a field. To understand the concept of tables it may be helpful to consider an example. A CR10X is to be used to monitor 3 thermocouples (TC). Each hour a temperature for each of the three TC is to be stored. The table has 4 fields: "DATE_TIME TEMP1 TEMP2 TEMP3." Each hour a new "record" would be added. The "hourly" table would then be organized as follows:

DATE_TIME	TEMP1	TEMP2	TEMP3
01/27/91 10:00:00	23.5	24.6	28.2
01/27/91 11:00:00	24.2	22.4	23.4

Only the hourly data is stored in the hourly table, Each output interval has its own table. Data tables can also be "event driven" rather than interval driven, that is a new record is stored when a specified event occurs rather than based on time. Each table is completely independent of any other tables and all records in a given table have the same number of fields.

Each table is allocated (manually by the user or automatically by the datalogger) a number of records. Different tables have different numbers of records. Each data table is in a ring memory configuration such that when the allocated number of records has been stored, each subsequent new record will overwrite the oldest stored record. The *9 Mode may be used to view the size of the Data Storage Tables. (Section 1.9)

The TD datalogger supports naming of tables and fields, so any data value can be referenced by the table and field names. For example, the temperature data for the first thermocouple is referenced as "HOURLY.TEMP1." As Data Tables are allocated in the datalogger program, some Final Storage Memory is reallocated for the storage of these labels and other data table overhead.

> **NOTE:** All Data Storage Tables are reallocated and erased whenever the datalogger program is recompiled (*0, *6, *B), when Input Storage Memory is reallocated (*A), or when a new datalogger program is transferred from the computer to the datalogger. ALWAYS RETRIEVE UNCOLLECTED DATA BEFORE MAKING ANY CHANGES.

A time stamp and record number are automatically included with the each record in each table. These are used as part of the data collection protocol.

2.1.1 TIME AND TIMESTAMPS

Each record in a table has a time stamp associated with it. With Instruction 84 set for interval output (a interval in seconds is specified as the second parameter), time is not actually stored with each record. Using the timestamp of the last record stored and the table interval, the datalogger can calculate the timestamp for any previous record. When retrieved, each record in the data file will have a timestamp. This saves 6 bytes per record by not storing time with each record. A consequence of not storing time is that if output does not occur at a scheduled time, the datalogger must keep track of the discontinuity in the timestamps so it can correctly calculate timestamps for records older than the missing record. The datalogger will keep track of the 10 most recent discontinuities in each table. If more than ten discontinuities occur, records with timestamps older than the oldest discontinuity cannot be reliably timestamped when collected. For this reason interval tables should not be used if outputs will be routinely missed. Outputs can be missed (discontinuities can occur) when:

- The datalogger clock is changed such that is passes an output interval.

- The output interval is not an even multiple of the scan rate (table execution interval).
- Table execution is such that Instruction 84 is not executed each scan.
- Table overruns occur.
- Watchdog errors (E08) occur.

2.1.2 RECORD NUMBERS

In addition to a timestamp, each record has record number. The record numbers are unique within a data table. Record numbers are 4 byte unsigned numbers ranging from 0 to 4,294,967,296. When the datalogger program is compiled the next record number for each table is set to zero.

2.1.3 TABLES AND FIELDS.

There are four "built in" tables. These tables are present when the datalogger is powered on. These tables are named INLOCS, TIMESET, ERRORLOG, and STATUS. The fields within these tables are also already defined with the exception of the INLOCS. The field names for each of the fields is given below. TmStamp is the label for the timestamp and RecNbr is the label for the record number.

TimeSet
TmStamp, RecNbr, OldTime

ErrorLog
TmStamp, RecNbr, Code

Status
TmStamp, RecNbr, Battery, WatchDog, OverRuns, InLocs, PrgmFree, Storage, Tables, DaysFull, Holes, PrgmSig, PromSig, PromID, ObjSrlNo

Where the labels are defined as follows:

Battery – Indicates the datalogger battery voltage.

WatchDog – The number of Watchdog Errors (E08). (Maximum 99). Section 3.10

OverRuns – Program table overruns that have occurred (Maximum 99). Section 1.1.1.

InLocs – Number of Input Location that have been allocated. Section 1.5.2

PrgmFree – Amount (bytes) of program memory remaining. Section 1.5.2

Storage – Number of Final Storage Locations available for Data Storage Tables. Section 1.5.2

Tables – Number of user created Data Tables.

DaysFull – Size (in days) of the Data Storage Tables using automatic record allocation. See Instruction 84.

Holes – Number of missed records or holes in all Data Storage Tables. Section 2.1.1

PrgmSig – Signature of program memory program. Same as *B mode first window. Section 1.6.

PromSig – Signature of datalogger PROM. Same as *B mode second window. Section 1.6.

Prom ID – Item number of PROM. Same as *B mode seventh window. Section 1.6.

ObjSrlNo – Object code serial number of PROM. Same as *B mode eighth window. Section 1.6.

Like the InLocs table, a new Status record is always available when the datalogger is checked for data. For this reason the Status table is usually collected for display rather than archive.

InLocs
TmStamp, RecNbr, Flags, Ports, Loc1, Loc2, Loc3, Loc4, Loc5, Loc6, Loc7, Loc8, Loc9

Like the Status table, a new InLocs record is always available when the datalogger is checked for data. For this reason these tables are usually collected for display rather than archive.

By default only nine Input Location are labeled. The default InLocs field labels can be replaced with user created labels. These labels are created with EDLOG technique for Input Location labels. Press CTRL-L when editing a LOC field.

When additional data tables are created with Instruction 84, these tables and the fields they contain can also be named by the user. The datalogger will supply a default name for tables and fields if they are not named. The default names are T01, T02, for the tables and F01, F02, etc. for the fields, numbered in the order they appear. Edlog allows the programmer to name the tables and fields.

The Timestamp and record number labels are added automatically.

2.2 DATA OUTPUT FORMAT AND RANGE LIMITS

Data is stored internally in Campbell Scientific's Binary Final Storage Format (Appendix C.2). Data may be sent to Final Storage in either LOW RESOLUTION or HIGH RESOLUTION format.

2.2.1 RESOLUTION AND RANGE LIMITS

Low resolution data is a 2 byte format with 4 significant digits and a maximum magnitude of +7999. High resolution data is a 4 byte format (see Section 2.2.2).

TABLE 2.2-1. Resolution Range Limits of CR10 Data

Resolution	Zero	Minimum Magnitude	Maximum Magnitude
Low	0.000	+ 0.001	+7999.
High	0.0000	$1x10^{-19}$	$+9x10^{+18}$

The resolution of the low resolution format is reduced to 3 significant digits when the first (left most) digit is 8 or greater. Thus, it may be necessary to use high resolution output or an offset to maintain the desired resolution of a measurement. For example, if water level is to be measured and output to the nearest 0.01 ft., the level must be less than 80 ft. for low resolution output to display the 0.01 ft. increment. If the water level was expected to range from 50 to 90 feet the data could either be output in high resolution or could be offset by 20 ft. (transforming the range to 30 to 60 ft.).

2.2.2 HIGH RESOLUTION FINAL STORAGE DATA, INPUT, AND INTERMEDIATE STORAGE DATA FORMAT

While low resolution output data have the limits described above, computations are done in floating point arithmetic. In high resolution Final Storage Input and Intermediate Storage, the numbers are stored and processed in a binary format with a 23 bit binary mantissa and a 6 bit binary exponent. The largest and smallest numbers that can be stored and processed are 9×10^{18} and 1×10^{-19}, respectively. The size of the number determines the resolution of the arithmetic. A rough approximation of the resolution is that it is better than 1 in the seventh digit. For example, the resolution of 97,386,924 is better than 10. The resolution of 0.008673192$\underline{4}$ is better than 0.000000001.

A precise calculation of the resolution of a number may be determined by representing the number as a mantissa between .5 and 1 multiplied by 2 raised to some integer power. The resolution is the product of that power of 2 and 2^{-24}. For example, representing 478 as $.9336 * 2^9$, the resolution is $2^9 * 2^{-24} = 2^{-15} = 0.0000305$. A description of Campbell Scientific's floating point format may be found in Appendix C.

2.3 DISPLAYING STORED DATA ON KEYBOARD/DISPLAY *7 MODE.

The keyboard display (or the computer in Keyboard/Display mode) can be used to examine Data Storage Tables in Final Storage table data.

Key *7. The display will show: 07:nn where nn is the number of Data Storage Tables defined. Enter a table number (followed by the “A” key) to view that table. Tables are numbered in the order of the appearance of the Data Storage Table Instruction (84) in the datalogger program. Tables in the *1 program area are numbered first, followed by *2 and those in the *3 subroutines numbered last.

The display will then show the first field of the newest record in the table. If the display does not change, the select table has not had any data stored in it yet.

When a table is visualized the newest data record is at the bottom and the oldest is at the top. When the table is full and a new record is stored, the records shift up pushing the oldest record off the top and storing the new record at the bottom.

The display (or value displayed on the computer) can be thought of as a cursor, which can be moved up and down or right and left through the data. The display shows the field number to the left of the colon and the data value to the right. The keys used to move the display/cursor are summarized in the following table:

TABLE 2.3-1. *7 Mode Command Summary

KEY	ACTION
A	"Advances" along a record, when the end of the record is reached the 'cursor' advances to the first field in the next record.
B	"Backs" up along a record, wraps to the last element in the previous record
C	"Climbs" up the table, toward the oldest data, stops on oldest record.
D	"Drops" down the table, toward the newest data, stops on newest record.
#	Enter Time Mode to display timestamp. Enter new time values to jump to record.

Each record of data in a table has a time associated with it. Event data stores the time as part of the data, interval tables calculate the time based on the interval and a reference time. The time associated with each record consist of a year, month/day, hour/minute, and seconds value. Julian dates are not used. When viewing data in the *7 mode, event data records have time as part of the record. The field number on the keyboard/display does not change while viewing the four time values since time is considered a single field. Interval data does not contain time and it is not displayed as part of the record. To see the time values for a given record, press the # key while viewing any of the fields within the record.

The A and B key are used move through the four time values for the record. The C and D keys can be used to move to newer or older records and view the same time value of the new record. Pressing the # key again while viewing time or using the A or B keys to advance or back beyond the time values will return the display to the same field as was displayed when the time mode was entered.

The time field is displayed as: day.month, year, hr:min, seconds

While viewing time, entering new time values will allow the display to jump to the record with time values closest to those entered. The jump takes place when the time mode is left.

Interval tables do not store time as part of the record, but calculate the time. A table 10 records long is maintained inside Intermediate Storage to keep track of "holes" in the recorded data, due to watch dog errors or clock changes, so that the time of each record can be implicitly maintained.

This "hole" table rings around, and any recorded data that cannot be time stamped using this "hole" table cannot be displayed.

To view the “hole” table for a given table, Key *7. The display will show: 07:nn, where nn is the number of Data Storage Tables defined. Enter the number of table where you want to view the “holes." Then press the C key followed by the A key. The Hole table fields are as follows:

01:time of hole;
02:number of holes.

SECTION 3. INSTRUCTION SET BASICS

Section 3.7.1 does not apply to the TD operating system which does not use Output Flag 0.

Table 3.8-1 Valid Flag Commands are 11 – 19 to set high and 21- 29 to set low. Because the TD operating system does not use Flag 0, Commands 10 and 20 are not valid with the TD operating system.

The following table replaces Table 3.10-1 for the TD operating system.

TABLE 3.10-1. Error Codes

Code	Type	Description
03	Editor	Program table full
04	Compile	Intermediate Storage full
05	Compile	Storage Area #2 not allocated
08	Run Time	CR10X reset by watchdog timer
09	Run Time	Insufficient Input Storage
10	Run Time	Low Battery Voltage
11	Editor	Attempt to allocate more Input or Intermediate Storage than is available
20	Compile	SUBROUTINE encountered before END of previous subroutine
21	Compile	END without IF, LOOP or SUBROUTINE
22	Compile	Missing END
23	Compile	Nonexistent SUBROUTINE
24	Compile	ELSE in SUBROUTINE without IF
25	Compile	ELSE without IF
26	Compile	EXIT LOOP without LOOP
27	Compile	IF CASE without BEGIN CASE
28	Compile	At compile time, no output specified after P84 or unable to automatically allocate at least one record per P84
29	Compile	Output table requests more memory than available
30	Compile	IF and/or LOOP nested too deep
31	Run Time	SUBROUTINES nested too deep
32	Compile	Instruction 3 and interrupt subroutine use same port
33	Compile	Cannot use Control Port 6 as counter or interrupt subroutine with Instruction 15 or SDM or SDI-12 input/output
40	Editor	Instruction does not exist
41	Editor	Incorrect execution interval
43	Compile	Time in Instruction 84 or 92 is not a multiple of execution interval (note: time is in seconds)
44	Compile	Loop (P87) cannot contain Data Table (P84)
60	Compile	Insufficient Input Storage
61	Compile	Burst Measurement Scan Rate too short
62	Compile	N<2 in FFT
68	Compile	Instruction 118 without enough Instructions 68 or 63
80	Compile	Illegal Interval in P193
81	Compile	Illegal Node ID in P193
82	Compile	Illegal Reps in P193
92	Compile	Instruction 92, intervals in seconds: Time into Interval > 59 or Interval > 60

94	Program Transfer	Program Storage Area full
95	Program Transfer	Program does not exist in Flash memory
96	Program Transfer	Addressed device not connected
97	Program Transfer	Data not received within 30 seconds
98	Program Transfer	Uncorrectable errors detected
99	Program Transfer	Wrong file type or editor error

THIS SECTION ENTIRELY REPLACES THE CR10X MANUAL SECTION 8.

SECTION 8. PROCESSING AND PROGRAM CONTROL EXAMPLES

This section contains examples for the CR10X. The appropriate voltage range codes would have to be selected for the CR23X (see CR23X Manual Section 8 for the measurement instructions). The CR510-TD may not support all the examples.

The following examples are intended to illustrate the use of Processing and Program Control Instructions, flags, and the capability to direct the results of Output Processing Instructions to Input Storage.

The specific examples may not be as important as some of the techniques employed, for example:

Directing Output Processing to Input Storage is used in the Running Average and Rainfall Intensity examples (8.1 and 8.2).

Flag tests are used in the Running Average, Interrupt Subroutine, and Converting Wind Direction (8.1, 8.5, and 8.7).

Control ports and the Loop are illustrated in the AM32 example (8.3).

As in Section 7 these examples are not complete programs to be taken verbatim. They need to be altered to fit specific needs.

8.1 COMPUTATION OF RUNNING AVERAGE

It is sometimes necessary to compute a running average (i.e., the average covers a fixed number of samples and is continuously updated as new samples are taken). Because the output interval is shorter than the averaging period, Instruction 71 cannot be used; the algorithm for computing this average must be programmed by the user. The following example demonstrates a program for computing a running average.

In this example, each time a new measurement is made (in this case a thermocouple temperature) an average is computed for the 10 most recent samples. This is done by saving all 10 temperatures in contiguous input locations and using the Spatial Average Instruction (51) to compute the average. The temperatures are stored in locations 11 through 20. Each time the table is executed, the new measurement is stored in location 20 and the average is stored in location 2. The Block Move Instruction (54) is then used to move the temperatures from locations 12 through 20 down by 1 location; the oldest measurement (in location 11) is lost when the temperature from location 12 is written over it.

Input Location Labels:

1:Panl Temp	15:Temp_i5
2:smpl10av	16:Temp_i4
11:Temp_i9	17:Temp_i3
12:Temp_i8	18:Temp_i2
13:Temp_i7	19:Temp_i1
14:Temp_i6	20:Temp_i

Where i is current reading, i1 is previous reading, etc.

```
*          1    Table 1 Programs
  01:     10    Sec. Execution Interval

01:       P17   Module Temperature
  01:      1    Loc [:Panl_Temp]

02:       P14   Thermocouple Temp (DIFF)
  01:      1    Rep
  02:      1    2.5 mV slow Range
  03:      1    IN Chan
  04:      1    Type T (Copper-Constantan)
  05:      1    Ref Temp Loc Panl_Temp
  06:     20    Loc [:Tempi    ]
  07:      1    Mult
  08:      0    Offset

03:       P51   Spatial Average
  01:     10    Swath
  02:     11    First Loc Temp_i9
  03:      2    Avg Loc [:smpl10avg]

04:       P54   Block Move
  01:      9    No. of Values
  02:     12    First Source Loc Temp_i8
  03:      1    Source Step
  04:     11    First Destination Loc
                [:Temp_i9  ]
  05:      1    Destination Step
```

```
05:     P84     Data Table
  01:     0     Seconds into interval
  02:     0     Every time
  03:     0     Records (0=auto; -=redirect)

06:     P70     Sample
  01:     1     Reps
  02:     2     Loc smpl10avg

07:       P     End Table 1
```

In the above example, all samples for the average are stored in input locations. This is necessary when an average must be output with each new sample. In most cases, averages are desired less frequently than sampling. For example, it may be necessary to sample some parameter every 5 seconds and output every hour an average of the previous three hours' readings. If all samples were saved, this would require 2160 input locations. The same value can be obtained by computing an hourly average and averaging the hourly averages for the past three hours. To do this requires that hourly averages be stored in input locations.

Instruction 84 is used to send the 1 hour average to Input Storage and again to send the 3 hour average to Final Storage.

Input Location Labels:

```
1:AVG_i2
2:AVG_i1
3:AVG_i
4: AVG_3_HR
5:XX_mg_M3
```

```
*         1     Table 1 Programs
  01:     5     Sec. Execution Interval

01:      P2     Volt (DIFF)
  01:     1     Rep
  02:    25     2500 mV 60 Hz rejection Range
  03:     3     IN Chan
  04:     5     Loc [:XX_mg_M3 ]
  05:    10     Mult
  06:     0     Offset

02:     P84     Data Table
  01:     0     Seconds into interval
  02:  3600     Seconds interval
  03:    -3     Records (0=auto; -=redirect)

03:     P71     Average
  01:     1     Rep
  02:     5     Loc XX_mg_M3

04:     P51     Spatial Average
  01:     3     Swath
  02:     1     First Loc AVG_i2
  03:     4     Avg Loc [:Avg_3_HR ]

05:     P84     Data Table
  01:     0     Seconds into interval
  02:  3600     Seconds interval
  03:     0     Records (0=auto; -=redirect)

06:     P70     Sample
  01:     1     Reps
  02:     4     Loc Avg_3_HR

07:     P92     If time is
  01:     0     seconds into a
  02:  3600     second interval
  03:    30     Then Do

08:     P54     Block Move
  01:     2     No. of Values
  02:     2     First Source Loc AVG_i1
  03:     1     Source Step
  04:     1     First Destination Loc
                [:AVG_i2   ]
  05:     1     Destination Step

09:     P95     End

10:       P     End Table 1
```

8.2 RAINFALL INTENSITY

In this example, the total rain for the last 15 minutes is output only if any rain has occurred. The program makes use of the capability to direct the output of Output Processing Instructions to Input Storage.

Every 15 minutes, the total rain is sent to Input Storage. If the total is not equal to 0, output is redirected to Final Storage Area 1, the time is output and the total is sampled.

Input Location Labels:

```
1:Rain_mm
2: TOT_15mm
```

```
*         1     Table 1 Programs
  01:    60     Sec. Execution Interval

01:      P3     Pulse
  01:     1     Rep
  02:     1     Pulse Input Chan
  03:     2     Switch closure
  04:     1     Loc [:Rain_mm  ]
  05:  .254     Mult
  06:     0     Offset

02:     P86     Do
  01:    21     Set low Flag 1
```

```
03:     P92     If time is
  01:     0     seconds into a
  02:   900     second interval
  03:    11     Set high Flag 1

04:     P84     Data Table
  01:     0     Seconds into interval
  02:    -1     When flag 1 is high
  03:    -2     Records (0=auto; -=redirect)

05:     P72     Totalize
  01:     1     Rep
  02:     1     Loc Rain_mm

06:     P89     If X<=>F
  01:     2     X Loc TOT_15min
  02:     1     =
  03:     0     F
  04:    21     Set low Flag 1

07:     P84     Data Table
  01:     0     Seconds into interval
  02:    -1     When flag 1 is high
  03:     0     Records (0=auto;  -=redirect)

08:     P70     Sample
  01:     1     Reps
  02:     2     Loc TOT_15min

09:       P     End Table 1
```

8.3 USING CONTROL PORTS AND LOOP TO RUN AM416 MULTIPLEXER

This example uses an AM416 to measure 16 copper-constantan thermocouples and 16 Model 223 soil moisture blocks. The sensors are read every ten minutes and the average value output once an hour. The multiplexer is housed in an AM-ENCT enclosure to minimize thermocouple errors created by thermal gradients. A 107 Temperature Probe is centrally located on the multiplexer board and used as a thermocouple temp. reference.

The AM416 switches the 223 moisture block out of the circuit when it is not being measured. This eliminates the need for the blocking capacitors used in the model 227 Soil Moisture Block. The 223 blocks are about one fifth the cost of the 227 blocks.

Control ports are used to reset the AM416 and clock it through its channels. The program sequence is:

Measure the 107 probe located at the AM416 for TC temperature reference.

CR10 sets the port high which resets the AM416.

A loop is entered; within each pass:
The port clocking the AM416 is pulsed.
The connected TCs and moisture blocks are measured.

CR10 sets the port controlling AM416 reset low.

Soil moisture measurements are converted to block resistances.

The input location in which the temperature and soil moisture measurements are stored is indexed to the loop counter (Instruction 87, Section 12). An indexed location is incremented by one with each pass through the loop. For example, on the first pass temperature is stored in Location 2, and soil moisture in Location 18. On the second pass temperature is stored in Location 3, and soil moisture in Location 18. After 16 loop passes, temperature and soil moisture measurements occupy Locations 2 through 17 and 18 through 33, respectively. Connections are shown in Figure 8.3-1.

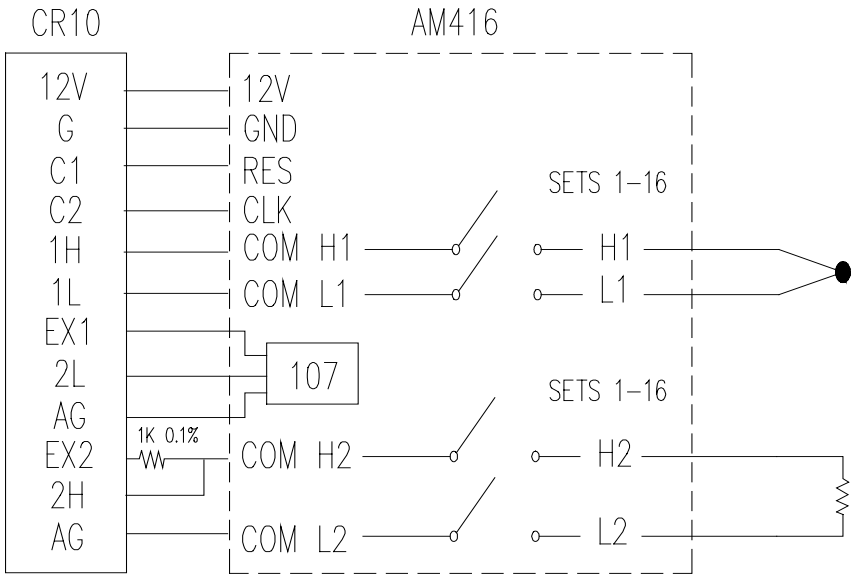


FIGURE 8.3-1. AM416 Wiring Diagram For Thermocouple and Soil Moisture Block Measurements

```
EXAMPLE PROGRAM  MULTIPLEXING
THERMOCOUPLES AND SOIL MOISTURE
BLOCK

*          1      Table 1 Programs
  01:    600      Sec. Execution Interval

01:        P11    Temp 107 Probe
  01:        1    Rep
  02:        4    IN Chan
  03:        1    Excite all reps w/EXchan 1
  04:        1    Loc [:REF_TEMP ]
  05:        1    Mult
  06:        0    Offset

02:        P86    Do
  01:       41    Set high Port 1

03:        P87    Beginning of Loop
  01:        0    Delay
  02:       16    Loop Count

04:        P86    Do
  01:       72    Pulse Port 2

05:        P14    Thermocouple Temp (DIFF)
  01:        1    Rep
  02:       21    2.5 mV 60 Hz rejection
                   Range
  03:        1    IN Chan
  04:        1    Type T (Copper-Constantan)
  05:        1    Ref Temp Loc REF_TEMP
  06:       2--   Loc [:TC_TEMP_1]
  07:        1    Mult
  08:        0    Offset

06:        P5     AC Half Bridge
  01:        1    Rep
  02:       14    250 mV fast Range
  03:        3    IN Chan
  04:        1    Excite all reps w/EXchan 1
  05:      250    mV Excitation
  06:      18--   Loc [:SOIL_M_1 ]
  07:        1    Mult
  08:        0    Offset

07:        P95    End

08:        P86    Do
  01:       51    Set low Port 1

09:        P59    BR Transform Rf[X/(1-X)]
  01:       16    Reps
  02:       18    Loc [:SOIL_M_1 ]
  03:       .1    Multiplier (Rf)

10:        P84    Data Table
  01:        0    Seconds into interval
  02:     3600    Seconds interval
  03:        0    Records (0=auto; -=redirect)

11:        P71    Average
  01:       33    Reps
  02:        1    Loc REF_TEMP

12:        P      End Table 1
```

8.4 INTERRUPT SUBROUTINE USED TO COUNT SWITCH CLOSURES (RAIN GAGE)

Subroutines given the label of 97 or 98 will be executed when control ports 7 or 8, respectively, go high (5 V, see Instruction 85, Section 12). In this example, Subroutine 98 and control port 8 are substituted for a pulse counting channel to count switch closures on a tipping bucket rain gage.

The subroutine adds 0.254 (mm, bucket calibrated for 0.01 inch tip) to an input location and uses Instruction 22 to delay 0.2 seconds.

The delay is to insure that any switch bouncing (when closing, the contacts actually bounce off each other, making and breaking the circuit several times) has died out before the subroutine is completed. (The pulse count inputs do this automatically.) Without the delay, the subroutine could be completed and called again by a bounce, causing false counts. The interrupt has no effect while the subroutine is still being executed.

Subroutine 98 is in effect keeping a running total in Input Storage. On the output interval, this total is sampled to Final Storage and zeroed by the program in Program Table 1.

To provide comparison, this example has the 2 pulse inputs also reading rain gages. (In a real situation, it is more likely that the pulse counters would be used for 2 wind speeds.) In Program Table 1, the 2 normal pulse inputs are read and the hourly totals output to Final Storage with Instruction 72.

The rain gage is connected as diagrammed below. When the switch closes, 5 volts is applied to port 8 which causes the subroutine to be executed.

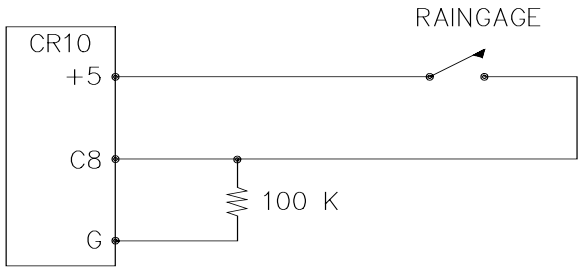


FIGURE 8.4-1. Connections for Rain Gage

Input location Assignments:

```
    10:Rain_1     (from Pulse count)
    11:Rain_2     (from Pulse count)
    12:Rain_3     (from subroutine 98 while
                  Output Flag is low)

*          1      Table 1 Programs
  01:     10      Sec. Execution Interval

01:       P3      Pulse
  01:      2      Reps
  02:      1      Pulse Input Chan
  03:      2      Switch closure
  04:     10      Loc [:Rain_1   ]
  05:    .254     Mult
  06:      0      Offset

02:      P84      Data Table
  01:      0      Seconds into interval
  02:   3600      Seconds interval
  03:      0      Records (0=auto; -=redirect)

03:      P72      Totalize
  01:      2      Reps
  02:     10      Loc Rain_1

04:      P70      Sample
  01:      1      Reps
  02:     12      Loc Rain_3

05:      P92      If time is
  01:      0      seconds into a
  02:   3600      second interval
  03:     30      Then Do

06:      P30      Z=F
  01:      0      F
  02:      0      Exponent of 10
  03:     12      Z Loc [:Rain_3   ]

07:      P95      End

08:        P      End Table 1


*          3      Table 3 Subroutines

01:      P85      Beginning of Subroutine
  01:     98      Subroutine Number

02:        P      34  Z=X+F
  01:     12      X Loc Rain_3
  02:    .254     F
  03:     12      Z Loc [:Rain_3   ]

03:      P22      Excitation with Delay
  01:      1      EX Chan
  02:      0      Delay w/EX (units=.01sec)
  03:     20      Delay after EX (units=.01sec)
  04:      0      mV Excitation

04:      P95      End

05:        P      End Table 3
```

8.5 SDM-A04 ANALOG OUTPUT MULTIPLEXER TO STRIP CHART

This example illustrates the use of the SDM-A04 4 Channel Analog Output Multiplexer to output 4 analog voltages to strip chart.

While of questionable value because of current requirements and strip chart reliability, some archaic regulations require strip chart backup on weather data. The SDM-A04 may be used with the CR10 to provide analog outputs to strip charts. The output values in this example are wind speed, wind direction, air temperature, and solar radiation.

Instruction 103 is used to activate the SDM-A04. The 4 millivolt values to output must be stored in adjacent Input Storage locations, the first of which is referenced in Instruction 103.

The following program measures the sensors every 5 seconds. The readings are moved to another 4 locations and scaled to a 0 to 1000 millivolt output for the SDM-A04. Wind direction is changed from a 0-360 degree input to output representing 0 to 540 degrees. This conversion is done in a subroutine which is described in the next example.

The example also includes instructions to output wind vector and average temperature and solar radiation every hour.

Input Location Labels:

```
1:WS
2: WD_360
3:TEMP_F
4:Solar_Rad
5:WS_output
6:WD540_out
7:TEMP_out
8:SR_out
10:WD_540
```

```
*          1        Table 1 Programs
  01:      5        Sec. Execution Interval

01:        P3       Pulse
  01:      1        Rep
  02:      1        Pulse Input Chan
  03:     22        Switch closure; Output Hz.
  04:      1        Loc [:WS      ]
  05:      1.789    Mult
  06:      1        Offset

02:        P4       Excite,Delay,Volt(SE)
  01:      1        Rep
  02:     14        250 mV fast Range
  03:      1        IN Chan
  04:      1        Excite all reps w/EXchan 1
  05:      5        Delay (units .01sec)
  06:   1000        mV Excitation
  07:      2        Loc [:WD_360   ]
  08:       .7273   Mult
  09:      0        Offset

03:       P11       Temp 107 Probe
  01:      1        Rep
  02:      2        IN Chan
  03:      2        Excite all reps w/EXchan 2
  04:      3        Loc [:Temp_F   ]
  05:      1.8      Mult
  06:     32        Offset

04:        P1       Volt (SE)
  01:      1        Rep
  02:      2        7.5 mV slow Range
  03:      3        IN Chan
  04:      4        Loc [:Solar_Rad]
  05:       .14493  Mult
  06:      0        Offset

05:       P54       Block Move
  01:      4        No. of Values
  02:      1        First Source Loc WS
  03:      1        Source Step
  04:      5        First Destination Loc
                    [:WS_output]
  05:      1        Destination Step

06:       P86       Do
  01:      1        Call Subroutine 1

07:       P53       Scaling Array (A*loc +B)
  01:      5        Start Loc [:WS_output]
  02:     10        A1 Scale WS, 0-100 MPH
                    = 0-1000 MV
  03:      0        B1
  04:      1.8519   A2 Scale WD, 0-540 DEG
                    = 0-1000 MV
  05:      0        B2
  06:      5.7143   A3 Scale TEMP, -25 - 100
                    F = 0-1000 MV
  07:     25        B3
  08:   1000        A4 Scale RADIATION, 0-
                    1KW/M^2 = 0-1000 MV
  09:      0        B4

08:      P103       SDM-A04
  01:      4        Reps
  02:     30        Address
  03:      5        Loc WS_output

09:       P84       Data Table
  01:      0        Seconds into interval
  02:   3600        Seconds interval
  03:      0        Records (0=auto;  -=redirect)
```

```
10:     P69     Wind Vector
  01:     1     Rep
  02:   180     Samples per sub-interval
  03:     0     Polar Sensor/(S, D1, SD1)
  04:     1     Wind Speed/East Loc WS
  05:     2     Wind Direction/North Loc WD_360
11:     P71     Average
  01:     2     Reps
  02:     3     Loc Temp_F
12:       P     End Table 1
```

8.6 CONVERTING 0-360 WIND DIRECTION OUTPUT TO 0-540 FOR STRIP CHART

If 0-360 degree wind direction is output to a strip chart the discontinuity at 0/360 will cause the pen to jump back and forth full scale when the winds are varying from the north. In the days of strip charts this was solved with a 0-540 degree pot on the wind vane (direction changes from 540 to 180 and from 0 to 360 so the pen only jumps once when the wind is out of the north or south).

When faced with the necessity of strip chart output (see previous example), the following algorithm can be used to change a 0-360 degree input to 0-540. (If you have a 0-540 pot, it can be used with the CR10 since the Wind Vector Instruction, 69, will work with this output.)

To change 0-360 degrees to the 0-540 degrees, 360 degrees must sometimes be added to the reading when it is in the range of 0 to 180. The following algorithm does this by assuming that if the previous reading was less than 270, the vane has shifted through 180 degrees and does not need to be altered. If the previous 0-540 reading was greater than 270, 360 degrees is added.

This example is written as a subroutine, used by the previous example to output an analog voltage to a strip chart.

Input Location Labels:

```
2:WD_360
6:WD540_out
10:WD_540
```

```
*       3     Table 3 Subroutines
*       3     Table 3 Subroutines

01:     P85     Beginning of Subroutine
  01:     1     Subroutine Number
02:     P89     If X<=>F
  01:    10     X Loc WD_540
  02:     3     >=
  03:   270     F
  04:    30     Then Do
03:     P86     Do
  01:    11     Set high Flag 1
04:     P94     Else
05:     P86     Do
  01:    21     Set low Flag 1
06:     P95     End
07:     P31     Z=X
  01:     2     X Loc WD_360
  02:    10     Z Loc [:WD_540   ]
08:     P89     If X<=>F
  01:    10     X Loc WD_540
  02:     4     <
  03:   180     F
  04:    30     Then Do
09:     P91     If Flag/Port
  01:    11     Do if flag 1 is high
  02:    30     Then Do
10:     P34     Z=X+F
  01:    10     X Loc WD_540
  02:   360     F
  03:    10     Z Loc [:WD_540   ]
11:     P31     Z=X
  01:    10     X Loc WD_540
  02:     6     Z Loc [:WD540_out]
12:     P95     End
13:     P95     End
14:     P95     End
15:       P     End Table 3
```

8.7 LOGARITHMIC SAMPLING USING LOOPS

A ground water pump test requires that water level be measured and recorded according to the following schedule.

Time into Test, min			Output Interval	Loop #
00	to	10	10 sec.	1
10	to	30	30 sec.	2
30	to	100	1 min.	3
100	to	300	2 min.	4
300	to	1000	5 min.	5
1000	and greater		10 min.	6

This is accomplished with a series of loops (Instruction 87), where the delay and count parameters are used to implement the frequency of measurement (and output) and the duration of the that frequency. The unit of delay is the execution interval. A delay of 1 with a 10 second execution interval and a count of 60 means the instructions in the loop, in this case measure and output water level, are executed every 10 seconds for 10 minutes.

The drawdown portion of the test is completed at some time greater than 1000 minutes. To enter the recharge phase of the test, the operator enters the *6AD Mode and sets Flag 1 high. At the next 10 minute pass through loop 6 the loop is exited. Program execution returns to the top of the program table and the measurement schedule starts over again for the recharge test.

The sensor is a 50 PSI Druck, model 930/ti with a calibration of 49.93 mV/10V of excitation or 4.993mV/V. Your calibration will be different. An excitation voltage of 1500 mV yields a maximum signal of 7.489 mV at 50 PSI, fully utilizing the 7.5 mV Input Range to provide the best resolution.

The multiplier, m, is calculated to provide depth of water in feet:

m = (50 psi/4.993 mV/V) * (2.3067 ft/psi)

m = 23.099 ft/mV/V

The offset is calculated to provide a final value that represents the distance from the lip of the well to the water surface. Similar to Figure 7.16-2, the offset equals the initial distance of 47.23 feet plus the initial reading of 54.77, or 102 feet.

```
*          1        Table 1 Programs
  01:     10        Sec. Execution Interval
```

User must toggle Flag 1 to start measurements

```
01:     P91        If Flag/Port
  01:    21        Do if flag 1 is low
  02:     0        Go to end of Program Table
```

Loop 1, Output every 10 seconds for 10 minutes

```
02:     P87        Beginning of Loop
  01:     1        Delay
  02:    60        Loop Count

03:     P86        Do
  01:     1        Call Subroutine 1

04:     P95        End
```

Loop2, Output every 30 seconds for 20 minutes

```
05:     P87        Beginning of Loop
  01:     3        Delay
  02:    40        Loop Count

06:     P86        Do
  01:     1        Call Subroutine 1

07:     P95        End
```

Loop 3, Output every 1 minute for 70 minutes

```
08:     P87        Beginning of Loop
  01:     6        Delay
  02:    70        Loop Count

09:     P86        Do
  01:     1        Call Subroutine 1

10:     P95        End
```

Loop 4, Output every 2 minutes for 200 minutes

```
11:     P87        Beginning of Loop
  01:    12        Delay
  02:   100        Loop Count

12:     P86        Do
  01:     1        Call Subroutine 1

13:     P95        End
```

Loop 5, Output every 5 minutes for 700 minutes

```
14:     P87        Beginning of Loop
  01:    30        Delay
  02:   140        Loop Count

15:     P86        Do
  01:     1        Call Subroutine 1

16:     P95        End
```

Loop 6, Output every 10 minutes until stopped by user

```
17:      P87          Beginning of Loop
  01:     60          Delay
  02:      0          Loop Count

18:      P86          Do
  01:      1          Call Subroutine 1

19:      P91          If Flag/Port
  01:     21          Do if flag 1 is low
  02:     31          Exit Loop if true

20:      P95          End

21:        P          End Table 1


*            3        Table 3 Subroutines

01:      P85          Beginning of Subroutine
  01:      1          Subroutine Number

02:       P6          Full Bridge
  01:      1          Rep
  02:     22          7.5 mV 60 Hz rejection Range

  03:      1          IN Chan
  04:      1          Excite all reps w/EXchan 1
  05:   1500          mV Excitation
  06:      1          Loc [:LEVEL_FT ]
  07:     .46199      Mult
  08:    102          Offset

03:      P84          Data Table
  01:      0          Seconds into interval
  02:      0          Every time
  03:      0          Records (0=auto; -=redirect)

04:      P70          Sample
  01:      1          Reps
  02:      1          Loc LEVEL_FT

05:      P95          End

06:        P          End Table 3
```

SECTION 9. INPUT/OUTPUT INSTRUCTIONS

***** 18 MOVE TIME TO INPUT LOCATION ******

FUNCTION

This instruction takes current time or date information and does a modulo divide (see Instruction 46) on the time/date value with the number specified in the second parameter. The result is stored in the specified Input Location.

Entering 0 or a number greater than the maximum value of the time/date for the modulo divide will result in the actual time/date value being stored.

PARAMETER 1 CODES

Code	**Time/Date Units**
00	Seconds into day (maximum 86400)
01	Minutes into day (maximum 1440)
02	Hours into year (maximum 8784)
03	Hours into day (maximum 24)
04	Day of month (maximum 31)
05	Month of year (maximum 12)
06	YR MO DAY HR MIN SEC

PARAM NUMBER	DATA TYPE	DESCRIPTION
01:	2	Time/Date Code
02:	4	Number to modulo divide by
03:	4	Input Location Number

Input Locations altered: 1

SECTION 11. OUTPUT PROCESSING INSTRUCTIONS

Instructions 73 – Maximum and 74 – Minimum have only one time option. (Time is output as a quoted string.) Instruction 80 – Set Active Storage Area, is not in the TD operating system. Its functions are included in Instruction 84 – Data Table. Instruction 84 is only in the TD operating system.

*** 73 MAXIMUM ***

FUNCTION
This instruction stores the MAXIMUM value taken (for each input location specified) over a given output interval. An internal FLAG is set whenever a new maximum value is seen. This FLAG may be tested by Instruction 79. Time of occurrence maximum value(s) is OPTIONAL output information, which is formatted and activated by entering one of the following CODES for Parameter no. 2.

CODE	**OPTIONS**
00	Output value ONLY
01	Output value with TIME

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Time of maximum (optional)
03:	4	Starting input location no.

Outputs Generated: 1 for each input location (plus 1 with time of max. option)

*** 74 MINIMUM ***

FUNCTION
Operating in the same manner as Program 73, this instruction is used for storing the MINIMUM value (for each input location specified) over a given output interval.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Time of minimum (optional)
03:	4	Starting input location no.

Outputs Generated: 1 for each input location (plus 1 with time of min. option)

*** 84 DATA TABLE ***

FUNCTION
Instruction 84 is used to define a table of final storage data. New records of data are stored in the table based on time (interval data) or when a user flag (CR10X flags 1-8) is set (event data). Time based output intervals are specified in seconds. Fractional values may be used following the same rules that apply for the table scan rate.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	FP	Time into interval (Seconds)
02:	FP	Time interval (Seconds) 0 = Store new record each time. -a = Store if user flag "a" is set (a=1..8)
03:	FP	Number of records in table 0 = automatically allocates number of records. If 0 and parameter 2 = -a (event driven) or if 0 and parameter 2 = 0 then allocates size as if a 1 second output table -x = redirect to input loc x

Parameter 1 specifies how many seconds into the interval, specified in parameter 2, data will be output to final storage. If the output is event driven, enter a 0 for parameter 1.

Parameter 2 specifies the output interval (seconds) for the table. If output is determined by a flag, -a should be used, ('a' is the flag to be tested). If Parameter 2 = 0, data will be output unconditionally each time Instruction 84 is executed.

Parameter 3 specifies how many records will be stored in the table. When the table is full, subsequent new records will overwrite the oldest

records. If 0 is entered, records will be automatically allocated such that all automatic tables will be filled at the same time. If some tables specify the number of records and some tables are automatically allocated, the specified records will be allocated first, and then the remaining memory will be divided among the automatically allocated tables such that they will be filled at the same time. The Star 9 mode gives the size of all tables and the period (in days) before any automatically allocated table fills. If -x is entered, the processed results are returned to input locations.

Section 12. Program Control Instructions

The TD operating system does not use the output Flag 0. Commands dealing with it are not valid.

Instructions 96 – Serial Output, 98 – Send Character, and 111 – Load Program from Flash, are NOT in the TD operating system.

The instructions described in this section are only in the PakBus operating system.

Wireless Networks

More recent CR10X, CR510, and CR23X dataloggers with the PakBus operating system use the PakBus communications protocol. In addition to providing a robust means of packet based communication, the protocol allows transfer of input location data from one datalogger to another, or from a Campbell Scientific wireless sensor/datalogger (CR200 series datalogger) to a "host" (or "master") TD datalogger.

The following is some general information on the requirements for successful communication in a PakBus network.

Datalogger Requirements

PakBus communication requires the current PakBus operating system in the CR10X, CR510, or CR23X datalogger. All CR200 Series dataloggers are capable of PakBus communication.

All dataloggers in the PakBus network require a unique address. The address for TD dataloggers is set in the *D15 mode. The address for the CR200 datalogger's is set using Pakcom software or LoggerNet version 2.1 or greater.

Communication Notes

PakBus dataloggers are also capable of Modbus communication. The Modbus packet can ride on top of the PakBus packet or can be send independently. Instruction 190 is used to set up a datalogger as a Modbus master device. The datalogger's Modbus address is set in the *D8 mode of the datalogger. The Modbus and Pakbus addresses for a datalogger cannot be the same.

In order to communicate to a datalogger via another datalogger from LoggerNet's Connect window (e.g., communicating with a CR205 via a CR10XTD-PB) the datalogger through which you will be communicating must be set as a router. This is done by entering the number of remote dataloggers that the router will be connecting to in the *D15 mode, parameter 2.

The PakBus instructions described in this section are often used in combination with a CR200 series datalogger and the RF400 series spread spectrum radio. Table 12-1 lists some of the advantages and disadvantages of different methods of transferring data.

TABLE 12-1. CR205/CR210/CR215 in PakBus Network

	Stand Alone Datalogger	SendGetData P190	Wireless Sensor P193
General Description	CR205 is programmed as a stand alone datalogger. Data are stored in datalogger and retrieved by computer running Loggernet	May be used as either a wireless sensor interface (data stored in "Master" Logger) or to transfer data to another logger while still independently storing data.	"Master" datalogger stores all data and sets transmission schedule for remote CR205 w/ sensors.
PRO	• Data are stored in Datalogger, loggernet will automatically retry if communication fails. • No special datalogger programming required for data transfer	• Most flexible datalogger to datalogger data transfer. • Can be set to automatically retry sending data.	• Potential for lowest power consumption by remote CR205. • Data are collected only from "Master" datalogger. • Automatic retries if adequate time is allocated
CON/limitations	• Separate data files for each logger.	• Both the "Remote Sensor" CR205 and the Master Datalogger need to be programmed to handle the data. • Data collection should be scheduled to avoid conflicts with datalogger to datalogger communication.	• Both the "Remote Sensor" CR205 and the Master Datalogger need to be programmed to handle the data. • Without storing data in the sensor, data are lost if transmission fails. • Master radio must be on continuously. • For simplicity, all wireless sensor CR205s should have the same sensors. (or at least no more than 4 different sensor configurations) • Data collection should be scheduled to avoid conflicts with Sensor to Master communication.
PakBus Instructions used in Datalogger Programming	Normal programming to measure sensors, process, and store data.	SetValue, GetValue, or P190 in datalogger initiating communication. Programming to deal with any sent data or codes in the other datalogger.	Wireless Network Master (P193) in master datalogger. TimeUntilTransmit (or P194) and SendGetData (or P196) in remote.

	Stand Alone Datalogger	SendGetData P190	Wireless Sensor P193
Radio Settings	*Radio address, net address, and hop sequence must be the same in all CR2xxs and RF400s in the network. Because only one header length can be set for a radio, only one power cycling interval should be used in network; i.e., do not mix 8 second and 1 second interval radio power cycling. RF on (no cycling) is specified where necessary below.*		
CR205 Power Mode and Header	Select power mode for appropriate response time and current drain.	Select power mode for appropriate response time and current drain. Header must match called station's cycle.	NO HEADER. RfpinEn (lowest power) or to allow communication with computer for checking station, downloading program: RF8_Sec or RF1_sec.
RF400 on CR10X or CR23X	Router, if used, must send appropriate header	Router, if used, must send appropriate header	RF on, *header to match CR205 setting to allow contacting CR205*
Beacon Interval	0	0	0
RF400 on computer	RF on, *header to match neighbors' settings*	RF on, *header to match neighbors' settings*	RF on, *header to match neighbors' settings*
LoggerNet Settings *When RF400 with direct access to network is connected to computer.*	*For each datalogger:* Com -PbusPort -logger	*For each datalogger with stored data:* Com -PbusPort -logger	*For each master datalogger:* Com -PbusPort -logger

PakBus Get/Send Locations (P190)

A program control instruction that sends data to or retrieves data from another datalogger in a PakBus network. This instruction can also be used to issue commands to Modbus devices, where the datalogger acts as a Modbus master.

```
1:  PakBus - Get/Send Locations (P190)
 1: 00       Port
 2: 0000     Address
 3: 00       Command
 4: 0000     Security
 5: 0000     Remote Loc/Coil/Register
 6: 0000     Swath
 7: 0000     Local Loc [ _____________ ]
 8: 0000     Result Code Loc [ _____________ ]
```

Notes:

Edlog allocates only one of the input locations used in parameters 5 and 7 of this instruction. The additional input locations must be inserted manually using the Input Location Editor.

If this instruction is used to retrieve a value or set a value in the remote datalogger's public (or input location) table (i.e., code 26 or 27 is used in parameter 3), Instruction 63 or 68 must follow this instruction to enter the variable name that will be accessed.

Index parameter 3 to delay execution of subsequent program instructions until the datalogger receives a valid response or error from the remote. Otherwise, the PakBus command is queued, the datalogger proceeds to the next instruction, and the communications are handled later when the remote replies.

Port

The communications port that will be used by the local PakBus datalogger during the execution of this instruction. Valid options are:

Code	Description
0	Modem Enabled Device, 300
1	Modem Enabled Device, 1200
2	Modem Enabled Device, 9600
3	Modem Enabled Device, 76800
4	Modem Enabled Device, 2400
5	Modem Enabled Device, 4800
6	Modem Enabled Device, 19200
7	Modem Enabled Device, 38400
16	SDC 6 (COM 310)
17	SDC 7
18	SDC 8

Note: The CR10X-TD and CR510-TD have limited communication rates and do not support options 4 through 7.

Address

When a PakBus command is being issued, this address refers to the PakBus address of the datalogger. When a Modbus command is being issued, this address refers to the Modbus address. The PakBus and Modbus addresses in the datalogger cannot be set to the same number.

If this instruction is used within a loop, index this parameter to automatically increment the address with each pass through the loop. When this parameter is indexed, program execution will be delayed until a response is received from the remote datalogger.

PakBus Communication

The unique address for the datalogger in the PakBus network that will be communicated with using this instruction.

The Pakbus address is set in the datalogger's *D15 mode.

Modbus Communication

The unique address for the datalogger in a Modbus network that will be communicated with using this instruction (the slave device).

The Modbus address is set in the datalogger's *D8 mode. The valid range of IDs for a Modbus slave device are 1 - 99. Setting the datalogger's Modbus address to 0 disables it as a Modbus slave.

Command

This parameter determines what type of communication should take place in the PakBus or Modbus network when the instruction is executed.

Command	Description
1	Read Coil Status (Modbus command)
2	Read Input Status (Modbus command)
3	Read Holding Registers (Modbus command)
4	Read Input Registers (Modbus command)
5	Force Single Coil (Modbus command)
15	Force Multiple Coils (Modbus command)
16	Preset Multiple Registers (Modbus command)
21	Receive input location data from another datalogger (Pakbus command)
22	Send input location data to another datalogger (Pakbus command)
26	Get Value
27	Set Value
61	Read Coil Status (Modbus command)
62	Read Input Status (Modbus command)
63	Read Holding Registers (Modbus command)
64	Read Input Registers (Modbus command)
65	Force Single Coil (Modbus command)
66	Force Multiple Coils (Modbus command)
67	Preset Multiple Registers (Modbus command)

Notes:

Codes 61 through 67 are used when the Modbus packet will ride on top of Pakbus as a datagram.

If Get Value or Set Value is used (26 or 27), parameter 5 is left blank and Instruction 63 or 68 is used following this instruction to enter the variable name in the datalogger's Public (or input locations) table that will be accessed.

If this parameter is indexed, the datalogger will proceed to the next instruction only after it receives a valid response or error from the remote. Otherwise, the PakBus command is queued, the datalogger proceeds to the next instruction, and the communications are handled later when the remote replies. It may be

desirable to delay execution of subsequent instructions if those instructions perform further processing on the response from the remote.

Security

Enter the level 2 security code for the remote datalogger in the PakBus network that will be communicated with using this instruction when Command 22 is used for parameter 3 (send input location data to another datalogger).

If the security code in this instruction does not match the security code of the remote datalogger, the remote datalogger will discard the message, and the failure will be indicated in the local datalogger by an incremental change in the Result Code Location (parameter 8).

If security is not set in the remote datalogger or if command 21 is used for parameter 3, this parameter can be left at 0.

For additional information on security codes, see Program Security.

Enter the level 2 security code for the remote datalogger in the PakBus network that will be communicated with using this instruction when Command 22 is used for parameter 3 (send input location data to another datalogger).

If the security code in this instruction does not match the security code of the remote datalogger, the remote datalogger will discard the message, and the failure will be indicated in the local datalogger by an incremental change in the Result Code Location (parameter 8).

If security is not set in the remote datalogger or if command 21 is used for parameter 3, this parameter can be left at 0.

Remote Location/Coil/Register

PakBus Communication

If data is being received from another datalogger in the PakBus network (Parameter 3 set to 21), this is the first input location in the remote datalogger from which to retrieve the data.

If data is being sent to another datalogger in the PakBus network (Parameter 3 set to 22), this is the first input location in the remote datalogger in which to store the first data value.

If a variable in the Public (or Input Locations) table is being accessed using Get Value or Set Value (Parameter 3 set to 26 or 27), this parameter is left blank and Instruction 63 or 68 is used following this instruction to enter the variable name.

Modbus Communication

This is the first coil or register to be acted upon when the instruction is executed.

For general information on input locations, see Input Locations.

Remote Location

PakBus Communication

If data is being received from another datalogger in the PakBus network (Parameter 3 set to 21), this is the first input location in the remote datalogger from which to retrieve the data.

If data is being sent to another datalogger in the PakBus network (Parameter 3 set to 22), this is the first input location in the remote datalogger in which to store the first data value.

Modbus Communication

This is the first coil or register to be acted upon when the instruction is executed.

Swath

PakBus Communication

The number of input locations that will be sent to or retrieved from the remote datalogger.

Modbus Communication

The number of subsequent coils or registers that will be acted upon when the instruction is executed.

Local Location

PakBus Communication

If data is being received from another datalogger in the PakBus network (Parameter 3 set to 21), this is the first input location in which to store the data.

If data is being sent to another datalogger in the PakBus network (Parameter 3 set to 22), this is the first input location for the swath of input locations that will be sent to the remote datalogger.

Notes:

If Command code 21 is chosen, the number of input locations required for the transferred data must be allocated manually.

If this instruction is used within a loop, index this parameter to automatically increment the input locations with each pass through the loop. The input locations for this parameter are calculated as the number of passes through the loop * the swath of locations, plus one location (for the response).

Modbus Communication

The input location that is the source or the destination of the information that will be transferred when this instruction is executed. Discrete values are packed

or unpacked with the least significant bit of the first byte, starting at this location. Incoming discrete values are set to -1.0 for ON and 0 for OFF. Outgoing discrete values are translated as 0.0 to OFF and non-zero to ON.

For general information on input locations, see Input Locations.

Result Code Location

The input location in which to store the results of the data transfer.

Result	Description
0	Successful
>0	Initial attempt failed (value indicates the number of retries)

Up to 2 retries will be attempted if data transfer fails. The retry interval is 1 second, plus 1/2 * number of hops for the node. The instruction runs in the background after initiated.

Note: If this instruction is used within a loop, index this parameter to automatically increment the input location in which the result is stored with each pass through the loop.

Send Final Storage Data (P191)

A program control instruction that transfers final storage data from one or more tables in a PakBus datalogger to a computer.

```
2:  PakBus - Send Final Storage Data (P191)
 1: 00       Port
 2: 0000     Address
 3: 0000     Table ID
 4: 00       Flag
```

Table ID

The ID for the data table that should be sent to the computer using this instruction. If the ID is set to 0, then all final storage tables will be transferred.

Flag

The user flag that will determine if the table definitions are transferred along with the data table(s). When the flag is high, the table definitions for the specified table(s) will be output as a separate data gram.

Send Message (P192)

A program control instruction that sends a message to another datalogger in the PakBus network.

This instruction can be used in a network with several Master dataloggers to synchronize all Master datalogger's clocks. One datalogger would use this instruction to periodically broadcast a clock report, and the remaining dataloggers would use instruction 195, Use Remote Clock Report, to set their clocks by the transmitted value.

This instruction is not necessary in networks with wireless sensors and only one Master datalogger, because the Wireless Network Master (P193) and Wireless Network Remote (P196) instructions perform these functions automatically.

This instruction can also be used to remove a datalogger from the PakBus network.

```
3:  PakBus - Send Message (P192)
 1: 00       Port
 2: 0000     Address
 3: 2        Clock Report
```

Message Type

Entry	Description
2	Clock report; sends the current time.
13	The datalogger that receives this message will remove the sending datalogger from its neighbor list (and therefore all links to the sending datalogger).

Wireless Network Master (P193)

A program control instruction that is used to prepare the local datalogger to send data to or receive data from one or more dataloggers/wireless sensors in a PakBus network. The instruction also assigns a transmission time to the remote dataloggers/wireless sensors. Instruction 193 does not actually initiate the transfer of data. Data transfer is initiated by the wireless sensor.

Multiple Instruction 193s can be used in a program to configure up to four different groups of dataloggers/wireless sensors. A "group" is determined by the First Remote Address and the Number of Remotes. A datalogger/wireless sensor can only belong to one group. An error message will occur (E81) if a datalogger/wireless sensor is assigned to more than one group.

```
4:  PakBus - Wireless Network Master (P193)
 1: 00       Number of Remotes
 2: 0000     First Remote Address
 3: 0000     Time Into Transmit Interval (sec)
 4: 0000     Transmit Interval (sec, 0 = use execution interval)
 5: 00       Transmit Delay Between Remotes (sec)
 6: 00       Swath to Receive
 7: 0000     First Loc for Data Received [ _____________ ]
 8: 00       Swath to Send
 9: 0000     First Loc to Send [ _____________ ]
10: 0000     Result Code Loc [ _____________ ]
```

Notes:

The wireless sensors will actually begin transmitting before the specified transmission time (based on Time Into Transmit Interval and Transmit Interval) so that transmission is complete when the specified transmission time occurs. The Transmit Delay Between Remotes is factored into to the transmit time assigned to each remote.

Edlog allocates only one of the input locations used in parameters 7, 9, and 10 of this instruction. The additional input locations must be inserted manually using the Input Location Editor. For information on manually inserting input locations, refer to Manually Inserting Input Locations Into Edlog.

Number of Remotes

The number of remote dataloggers/wireless sensors in the PakBus network that will be communicated with using this instruction.

First Remote Address

The unique address for the first remote datalogger/wireless sensor in the PakBus network that will be communicated with using this instruction. All of the remotes that will be communicated with using this P193 should have sequential addresses.

The address is set in the datalogger's *D15 mode (refer to the datalogger user's manual for additional information). It can be any number between 1 and 4095.

Time into Transmit Interval

An offset, in seconds, into the Transmit Interval. The valid range is 0 through 9998.

Example

To set up the remotes for an hourly transmission at 15 minutes past the hour, the Time into Transmit Interval would be set at 900 and the Transmit Interval would be set at 3600.

Transmit Interval

The transmission interval, in seconds, that will be assigned to the group of dataloggers/remote sensors being set up with this instruction. The valid range is 1 through 9999.

The wireless sensors will actually begin transmitting before the specified transmission time (based on Time Into Transmit Interval and Transmit Interval) so that transmission is complete when the specified transmission time occurs. The Transmit Delay Between Remotes, parameter 5, is factored into to the transmit time assigned to each remote.

The datalogger program can be written to execute P193 on every program scan, or within a P92 (If Time) instruction. The Transmit Interval must equal the interval on which Instruction 193 is being executed.

Note: The Transmit Interval must be sufficiently long so that all of the remotes have a chance to respond before the next transmit interval occurs. Therefore, Transmit Interval must be equal to or greater than Number of Remotes * Transmit Delay Between Remotes. You will have to estimate the Transmit Delay Between Remotes if that value (parameter 5) is set to 0 (which means use the estimated value from the datalogger's routing table).

Example

To set up the remotes for an hourly transmission at 15 minutes past the hour, the Time into Transmit Interval would be set at 900 and the Transmit Interval would be set at 3600.

Transmit Delay Between Remotes

The amount of delay, in seconds, between transmission from each remote. If this parameter is left at 0, the master datalogger will automatically assign the delay based on the routing table (usually about 3 seconds between remotes). Otherwise, a specific delay can be entered. A specific delay may be necessary for slow communication links.

Some communications links do not require a delay (such as when communicating over an NL100). In this instance, the parameter can be left at 0 and indexed.

Code	Description
0	Use the default, as determined by the routing table
>0	Use the value entered
--0	Use no delay (dashes are accomplished by indexing the parameter)

Note: The wireless sensors will actually begin transmitting before the specified transmission time (based on Time Into Transmit Interval and Transmit Interval) so that transmission is complete when the specified transmission time occurs. The Transmit Delay Between Remotes is factored into to the transmit time assigned to each remote.

Example

Assume 4 wireless remotes in a network, with the first having an address of 1 and the remainder of the remotes addressed consecutively. The transmission time is set at 900 seconds into a 3600 second interval (15 minutes past each hour). If Transmit Delay Between Remotes is set at 5, Remote 4 will transmit at about 15 seconds before the transmit time, Remote 3 at about 10 seconds before, Remote 2 at about 5 seconds, and Remote 1 at the transmit time.

Swath to Receive

The number of data values that will be received from each remote when data is transferred. If a remote sends less than the number of values indicated by the swath, the remaining locations will be filled with an overrange value (-99999). If a remote sends more than the number of data values indicated by the swath, the extra values will be discarded by the local datalogger.

First Location for Data Received

The first input location in which the first data value received from the first remote should be stored. Subsequent data values from the group of remotes will be stored in consecutive input locations.

Note: The number of input locations required for the transferred data (Number of Remotes * Swath to Receive) must be allocated manually.

For general information on input locations, see Input Locations.

Swath to Send

The number of data values that will be sent to each remote when data is transferred.

First Location to Send

The input location which holds the first value that should be sent to the dataloggers/wireless sensors in the group. The range of values sent to the remote(s) is determined by the Swath to Send parameter (parameter 8).

For general information on input locations, see Input Locations.

Result Code Location

The input location in which a code is stored to indicate the result of the data transfer. A 0 indicates the data transfer was successful; any number greater than 0 indicates a failure.

Note: One Result Code Location is required for each remote (specified by Number of Remotes, parameter 1). These input locations must be allocated manually.

When the datalogger receives a wireless message from a remote, the corresponding Result Code Location is set to -1. When Instruction 193 is executed, the Result Code Location is incremented by 1. Therefore, if communication is successful, the Result Code Location will be 0 after the execution of Instruction 193. If data transfer is unsuccessful, the Result Code Location for the remote that failed will be incremented, and will continue to increment with each failed attempt.

Seconds Until Transmit (P194)

A program control instruction that places in an input location the number of seconds until it is time to transmit data to the host datalogger. This instruction is used in conjunction with a conditional statement to determine when the Wireless Network Remote instruction (P196) is executed to initiate communication with the host. The communication schedule is determined by the host (or master) datalogger and is set in the remote when it first initiates communication. If no communication has taken place and the value has not been set, or if scheduled communication was not successful, Seconds Until Transmit will return a random offset into a one minute interval.

```
5:  PakBus - Seconds Until Transmit (P194)
 1: 0000     Loc with Seconds Until Transmit [ _____________ ]
```

Note: If the datalogger is not being used as a wireless sensor (i.e., Instruction 196 is not in the program), this instruction can be used to place a random number of seconds into a minute interval in the specified input location. The random seed is based on the datalogger's PakBus address.

Location with Seconds Until Transmit

The input location in which to store the number of seconds until it is time to transmit to the host datalogger.

Use Remote Clock Report (P195)

A program control instruction that sets a remote datalogger's clock based on the clock value transmitted from the host (or master) datalogger specified by the address provided in parameter 1.

```
6:  PakBus - Use Remote Clock Report (P195)
 1: 0000     Address
```

Note: This instruction is not used if the datalogger is configured as a wireless sensor using instruction 196 (Wireless Network Remote). When instruction 196 is used, the remote datalogger will automatically adjust its clock to match the host (master) datalogger's clock whenever communication is successful.

Address

The unique address for the host (master) datalogger in the PakBus network, whose clock value will be used to set the clock in the remote datalogger.

The address is set in the datalogger's *D15 mode (refer to the datalogger user's manual for additional information). It can be any number between 1 and 4095.

Wireless Network Remote (P196)

A program control instruction that is used to set up a remote datalogger to act as a wireless sensor/controller in a PakBus network.

Communication with the host (master) datalogger is dictated by the host datalogger. A communication time is assigned to the remote datalogger when communication is first accomplished with the host. The remote datalogger uses Instruction 194, Seconds Until Transmit, in conjunction with a conditional statement to determine when P196 is executed, and therefore, when data is transferred to the host.

When first installed or when communication is not successful, P194 will indicate a random interval of up to one second. The remote will try to contact the master on this interval until communication is successful and it is programmed with a transmit time.

The remote datalogger's clock is synchronized with the host datalogger's clock, each time communication between the two dataloggers is successful.

```
7:  PakBus - Wireless Network Remote (P196)
 1: 00       Port
 2: 0000     Master Address
 3: 0000     Security
 4: 00       Swath to Receive from Master
 5: 0000     First Loc for Data Received [ _____________ ]
 6: 00       Swath to Send to Master
 7: 0000     First Loc to Send [ _____________ ]
 8: 0000     Result Code Loc [ _____________ ]
```

Swath to Receive From Master

The number of data values that will be received from the host (master) datalogger when data is transferred. If the host sends less than the number of values indicated by the swath, the remaining locations will be filled with an overrange value (-99999). If the host sends more than the number of data values indicated by the swath, the extra values will be discarded by the local datalogger.

First Location for Data Received

The first input location in which the first data value received from the host (master) datalogger should be stored. Subsequent data values from the host will be stored in consecutive input locations.

Notes:

The number of input locations required for the transferred data must be allocated manually.

Security

Enter the level 2 security code for the master datalogger in the PakBus network that will be communicated with using this instruction .

If the security code in this instruction does not match the security code of the master datalogger, the master datalogger will discard the message, and the failure will be indicated in the local datalogger by an incremental change in the Result Code Location (parameter 8).

If security is not set in the master datalogger, this parameter can be left at 0.

For additional information on security codes, see Program Security.

Enter the level 2 security code for the master datalogger in the PakBus network that will be communicated with using this instruction .

If the security code in this instruction does not match the security code of the master datalogger, the master datalogger will discard the message, and the failure will be indicated in the local datalogger by an incremental change in the Result Code Location (parameter 8).

If security is not set in the master datalogger, this parameter can be left at 0.

Swath to Send

The number of data values that will be sent to the host (master) datalogger when data is transferred.

First Location to Send

The input location which holds the first value that should be sent to the host (master) datalogger. The range of values sent is determined by the Swath to Send parameter (parameter 6).

For general information on input locations, see Input Locations.

Result Code Location

The input location in which a code is stored to indicate the result of the data transfer. A 0 indicates the data transfer was successful; any number greater than 0 indicates a failure. A -2 indicates that communication was established with the datalogger at the specified address, but the datalogger was not programmed as a host (master) datalogger using Instruction 193. In this instance, a 0 is stored in parameter 5, First Location for Data Received.

For general information on input locations, see Input Locations.

Force Route (P197)

A program control instruction that is used to force the datalogger to use a specific route in the PakBus network to communicate with the destination datalogger. This information is set in the datalogger's routing table.

```
8:  PakBus - Force Route (P197)
 1: 00       Port
 2: 0000     Neighbor's Address
 3: 0000     Address
 4: 00       Hops
```

Note: For communications paths where there are multiple hops, this instruction fixes only the first hop.

Neighbor's Address

The address of the first hop (or repeater) in the PakBus network that the datalogger should use in communicating with the destination datalogger.

Hops

The number of hops (or repeaters) in the communications path to the destination datalogger.

Set Setting (P198)

A program control instruction that is used to set a setting in a PakBus datalogger. This instruction should be followed by instruction 63 or instruction 68 with the values for the setting that should be changed.

If the address in this instruction is set to the address of the datalogger executing the instruction, the datalogger will change its own setting.

```
9:  PakBus - Set Setting (P198)
 1: 00       Port
 2: 0000     Address
 3: 0000     Result Code Loc [ _____________ ]
```

Result Location

Result Code	Description
-1001	The attempted setting is a read-only setting
-1002	Out of space in the remote
-1003	Syntax error
0	Success
>1	Number of communication failures

Routing Table Information (P199)

A program control instruction that is used to store the datalogger's routing table information in a series of input locations. This instruction is used most often as a trouble-shooting tool.

```
10:  PakBus - Routing Table Information (P199)
 1: 0000     First Loc [ _____________ ]
```

Parameter 1 specifies the first input location in which to begin storing the information. For each route, there are 3 pieces of information returned:

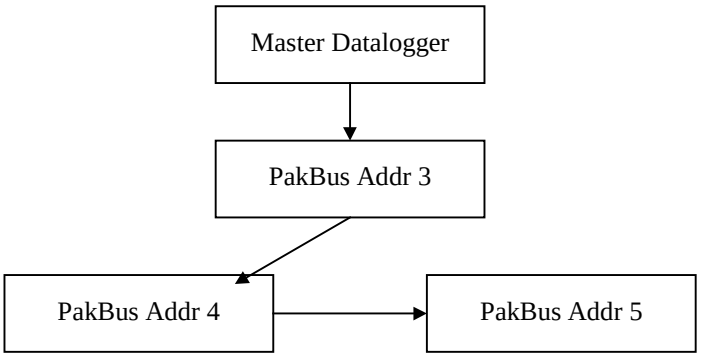
1. The PakBus Address of the Destination datalogger.
2. The PakBus address of any datalogger used as a hop to the Destination datalogger.
3. The Response Metric, in seconds (each hop takes 1 second).

A -1 in an input location indicates the end of the routing table information.

Note: The input locations required for this instruction must be allocated manually.

Example

Consider the following routing table information:



The information returned using this instruction would be similar to:

Input Location Used	Value Stored	Description
1	3	Address of destination datalogger
2	3	Address of repeater datalogger
3	1	Response metric, 1 second (1 hop)
4	4	Address of destination datalogger
5	3	Address of repeater datalogger
6	2	Response metric, 2 seconds (2 hops, 3 & 4)
7	5	Address of destination datalogger
8	3	Address of repeater datalogger (first hop only)
9	3	Response metric, 3 seconds (3 hops, 3, 4 & 5)
10	-1	End of string

PakBus Settings (Options | PakBus Settings)

This dialog box is used to configure some of the PakBus settings, that are normally set in the datalogger's *D mode, when a program is downloaded to the datalogger.

For all of the options below, if the check box Do Not Change Current Settings is enabled, then those settings will not be changed when the program is downloaded to the datalogger.

Network

The Network option is used to set the PakBus address in the datalogger and to configure the datalogger as a router if required. This option is the same as the datalogger's *D19 mode.

Address - Enter the PakBus address that should be assigned to the datalogger.

Maximum number of nodes - Enter the total number of dataloggers in the PakBus network.

Maximum number of neighbors - Enter the number of dataloggers in the PakBus network that the datalogger can communicate with directly (i.e., without going through another datalogger).

Maximum number of routers - Enter the number of neighbors to the datalogger that act as routers to one or more other dataloggers in the PakBus network.

Beacon Intervals

This option is used to set the interval on which the datalogger will transmit a beacon out a particular port to the PakBus network. Use the drop-down list box to select the port over which the beacon will be transmitted, and enter the

desired interval in the Communications Interval field. This option is the same as the datalogger's *D18 mode.

In some networks, a beacon interval might interfere with regular communication in the PakBus network (such as in an RF network), since the beacon is broadcast to all devices within range. In such cases, it may be more appropriate to use the Neighbor Filter instead, which broadcasts a beacon only to those dataloggers which it has not received communication from within a specified interval.

Neighbor Filter

This option allows you to list potential neighbors that are available to the datalogger in the PakBus network. The datalogger will attempt to issue a "hello" command to all the dataloggers listed in the neighbors filter list, and will transmit an expected communication interval. The communication interval is the interval on which the datalogger expects to receive communication from the neighbors. If communication is not received from a neighbor within 2.5 times this interval, then the datalogger will attempt to issue another "hello" command to that datalogger only (thus, creating less network traffic than the Beacon Interval).

The expected interval is entered into the Communication Interval field in seconds. The neighbors are defined by entering their addresses into the table. A range of addresses can be entered by using the Swath field. For example, entering 1 for the address and 5 for the swath will set up dataloggers with PakBus addresses 1, 2, 3, 4, and 5 as neighbors to the current datalogger.

This option is the same as the datalogger's *D19 mode.

Allocate General Purpose File Memory

PakBus dataloggers have the ability to store files transmitted from an NL100 in a general purpose memory area. This memory area is configured as ring memory. A value can be entered to specify the number of 64K blocks of memory that should be used for this purpose. Final storage memory will be reduce by the amount of memory specified in this option. This option is the same as the datalogger's *D16 mode.

LIST OF TABLES

PAGE

OVERVIEW

1. FUNCTIONAL MODES

2. INTERNAL DATA STORAGE

3. INSTRUCTION SET BASICS

4. EXTERNAL STORAGE PERIPHERALS

5. TELECOMMUNICATIONS

PAGE

6. 9 PIN SERIAL INPUT/OUTPUT

7. MEASUREMENT PROGRAMMING EXAMPLES

9. INPUT/OUTPUT INSTRUCTIONS

12. PROGRAM CONTROL INSTRUCTIONS

13. CR510 MEASUREMENTS

14. INSTALLATION AND MAINTENANCE

APPENDIX B. ADDITIONAL TELECOMMUNICATIONS INFORMATION

APPENDIX E. CALL ANOTHER DATALOGGER VIA PHONE OR RF

LIST OF FIGURES

PAGE

OVERVIEW

1. FUNCTIONAL MODES

2. INTERNAL DATA STORAGE

3. INSTRUCTION SET BASICS

4. EXTERNAL STORAGE PERIPHERALS

6. 9-PIN SERIAL INPUT/OUTPUT

7. MEASUREMENT PROGRAMMING EXAMPLES

8. PROCESSING AND PROGRAM CONTROL EXAMPLES

9. INPUT/OUTPUT INSTRUCTIONS

10. PROCESSING INSTRUCTIONS

13. CR510 MEASUREMENTS

14. INSTALLATION AND MAINTENANCE

APPENDIX D. DATALOGGER INITIATED COMMUNICATIONS