CR10 Measurement and Control Module

Instruction Manual

Issued 2.9.94

Guarantee

This equipment is guaranteed against defects in materials and workmanship. This guarantee applies for thirty-six months from date of delivery. We will repair or replace products which prove to be defective during the guarantee period provided they are returned to us prepaid. The guarantee will not apply to:

- Equipment which has been modified or altered in any way without the written permission of Campbell Scientific
- Batteries
- Any product which has been subjected to misuse, neglect, acts of God or damage in transit.

Campbell Scientific will return guaranteed equipment by surface carrier prepaid. Campbell Scientific will not reimburse the claimant for costs incurred in removing and/or reinstalling equipment. This guarantee and the Company's obligation thereunder is in lieu of all other guarantees, expressed or implied, including those of suitability and fitness for a particular purpose. Campbell Scientific is not liable for consequential damage.

Please inform us before returning equipment and obtain a Repair Reference Number whether the repair is under guarantee or not. Please state the faults as clearly as possible, and if the product is out of the guarantee period it should be accompanied by a purchase order. Quotations for repairs can be given on request.

When returning equipment, the Repair Reference Number must be clearly marked on the outside of the package.

Note that goods sent air freight are subject to Customs clearance fees which Campbell Scientific will charge to customers. In many cases, these charges are greater than the cost of the repair.

Campbell Scientific Ltd,
Campbell Park, 80 Hathern Road,
Shepshed, Leicestershire, LE12 9GX UK
Tel: +44 (0) 1509 601141
Fax: +44 (0) 1509 601091
Email: support@campbellsci.co.uk
http://www.campbellsci.co.uk

Contents

Part III Reference Manual

Selected Operating Details

1. **Storing Data** — Data is stored in Final Storage only by Output Processing Instructions and only when the Output Flag is set (see Overview).

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 (see Section 11).

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

4. **Final Storage Resolution** — All Input Storage values are displayed as high resolution with a maximum value of 99999. However, the default resolution for data stored in Final Storage is low resolution, with a 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 (see sections 2 and 11).

5. **Floating Point Format** — The computations performed in the CR10 use floating point arithmetic. Campbell Scientific'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 (see Section 2).

6. **Erasing Final Storage** — Data in Final Storage can be erased without altering the program by using the *A Mode to repartition memory (see Section 1).

7. **ALL memory can be erased** and the CR10 completely reset by entering 1986 for the number of bytes left in Program Memory (see Section 1).

8. **The set of instructions** available in the CR10 is determined by the PROM (Programmable Read Only Memory) that it is equipped with. Standard and optional software are identified in Appendix B. If you have ordered optional software that is not covered in the standard manual, the documentation is in Appendix G.

9. **Radiotelemetry Users** — From February 1990, CR10 PROMs no longer contain RF interface software. That function is now contained in the RF95 Modem. To make measurements at a phone-to-RF base station using a radio and the RF95 Modem, current CR10 software is required. A CR10 with old software can be used with the new RF95 in the 'RF95-ME' state, but the datalogger loses the 'callback' capability as well as the SDC function.

10. **Switched 12V Supply** — The wiring panel now includes a switched 12V output (not available on old-style silver wiring panels). This can be used to power sensors or devices requiring an unregulated 12 volts. The output is limited to 600mA current. A control port is used to operate the switch. Connect a wire from the control port to the switched 12V control. When the port is set high, the 12V is turned on; when the port is low, the switched 12V is off. If necessary an excitation channel (programmed to give 2500mV) can be used to control the switched 12V output. (Do not use the same excitation channel to provide AC excitation to a sensor because the excitation voltage will not be able to change quickly enough.)

Cautionary Notes

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

2. When using the CR10 with the 'Cyclon' lead-acid batteries previously supplied by Campbell Scientific remember that they are permanently damaged if discharged below 11.76V. The cells experience a slow discharge even in storage. It is therefore advisable to maintain a continuous charge on them, whether in operation or storage (see Section 14).

3. When connecting power to the CR10, *first connect the positive lead from the power source to the 12V terminal,* then connect the negative lead to G. Connecting these leads in the reverse order creates the possibility of a short circuit (see Section 14).

4. Voltages in excess of 5.5V applied to a control port can cause the CR10 to malfunction.

5. Voltage pulses can be counted by the CR10 pulse counters configured for high frequency pulses. However, when the pulse is actually a low frequency signal (below about 10Hz) *and* the positive voltage excursion exceeds 5.6V DC, the 5V DC supply will start to rise, upsetting all analogue measurements.

   Pulses whose positive voltage portion exceeds 5.6V DC with a duration longer than 100ms need external conditioning. See the description of the Pulse Count instruction in Section 9 for details on the external conditioning.

6. The CR10 module is sealed and contains desiccant to protect against the effects of moisture. However, the Wiring Panel and the connections between the Wiring Panel and the CR10 are still susceptible to moisture. To prevent corrosion at these points, additional desiccant must be placed inside the enclosure. If alkaline batteries only are inside the enclosure, the sensor cable entry may be sealed to inhibit vapour transfer into the enclosure.

WARNING **Do not seal the cable entry completely if lead-acid batteries are present, since hydrogen gas generated by the batteries could build up to an explosive concentration.**

CR10 Measurement and Control Module Overview

The CR10 Instruction Manual is divided into three parts:

1. *This Overview*
2. *The CR10 User Guide (Sections 1–8)*
3. *The CR10 Reference (Sections 9–14 and Appendices A–G)*

This Overview introduces the concepts required to take advantage of the CR10's capabilities. Hands-on programming examples start in Section OV5. Working with a CR10 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. You may also find the PCTOUR demonstration disk useful — please call if you do not have a copy.

The sections of the Instruction Manual which should be read to complete a basic understanding of the CR10 operation are Sections 1–3 (programming), Sections 4 and 5 (data retrieval) and Section 14 (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 describes CR10 measurement procedures in detail.

The Prompt Sheet is an abbreviated description of the programming instructions. Once familiar with the CR10, it is possible to program it using only the Prompt Sheet as a reference, consulting the manual if further detail is needed.

NOTE Read the Selected Operating Details and Cautionary Notes at the front of the Manual before using the CR10.

OV1. Physical Description

The CR10 is a fully programmable datalogger and controller in a small, rugged, sealed module. Programming is very similar to Campbell Scientific's 21X and CR7 dataloggers. Some fundamental physical differences are listed below.

- The CR10 does not have an integral keyboard/display. You communicate with the CR10 with the portable CR10KD Keyboard/Display or with a computer or terminal (see Section OV3).
- The CR10 does not have an integral terminal strip. A removable wiring panel, the CR10WP (see Figure OV-1), performs this function and attaches to the two D-type connectors located at the end of the module.
- The power supply is external to the CR10. This gives you a wide range of options for powering the CR10 (see Section 14).

OV1.1 Wiring Panel — Model CR10WP

The CR10WP Wiring Panel and CR10 datalogger make electrical contact through the two D-type connectors at the (left) end of the CR10.

The Wiring Panel has a 9-pin serial I/O port used when communicating with the datalogger and provides terminals for connecting sensor, control and power leads to the CR10. It also provides transient protection and reverse polarity protection. Figure OV-1 shows the panel and Figure OV-2 shows the instructions used to access the various terminals.

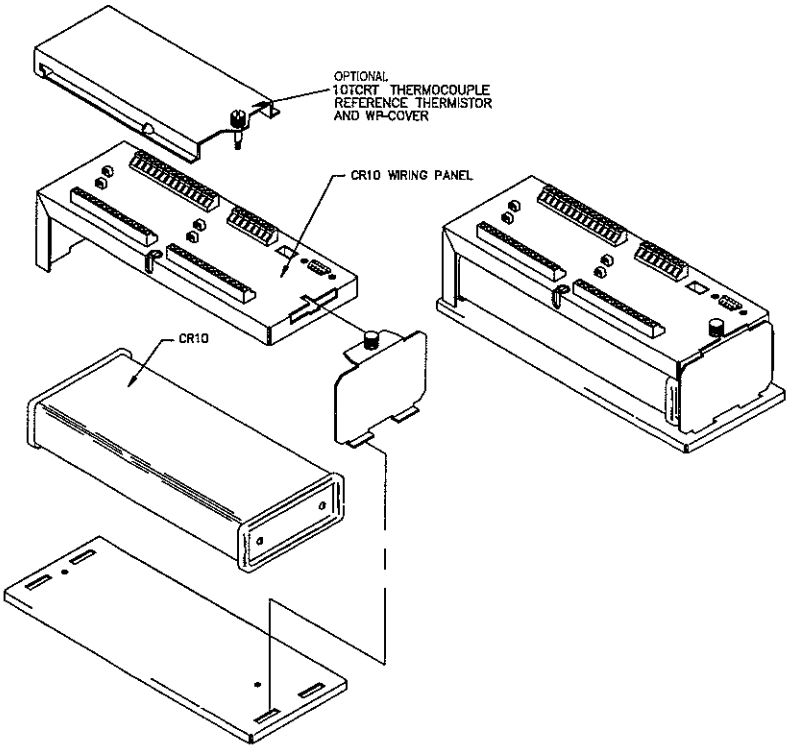


Figure OV-1 CR10 and Wiring Panel

Analogue Inputs

The terminals labelled 1H to 6L are analogue inputs. These numbers refer to the high and low inputs to the differential channels 1 to 6. 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 CR10 analogue 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, and so on.

NOTE The single-ended channel numbers do *not* appear on older (silver) Wiring Panels.

Switched Excitation Outputs

The terminals labelled E1, E2, and E3 are precision, switched excitation outputs used to supply programmable excitation voltages for resistive bridge measurements. DC or AC excitation at voltages between -2500mV and +2500mV are user programmable (see Section 9).

Pulse Inputs

The terminals labelled P1 and P2 are the pulse counter inputs. They are programmable for switch closure, high frequency pulse or low level AC (see description of Instruction 3 in Section 9).

Digital I/O Ports

Terminals C1 to C8 are digital input/output ports. On power-up they are configured as input ports, 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 outputs the ports allow on/off control of external devices. A port can be set high (5V ± 0.1V), set low (<0.1V), toggled or pulsed (see Sections 3, 8 and 12).

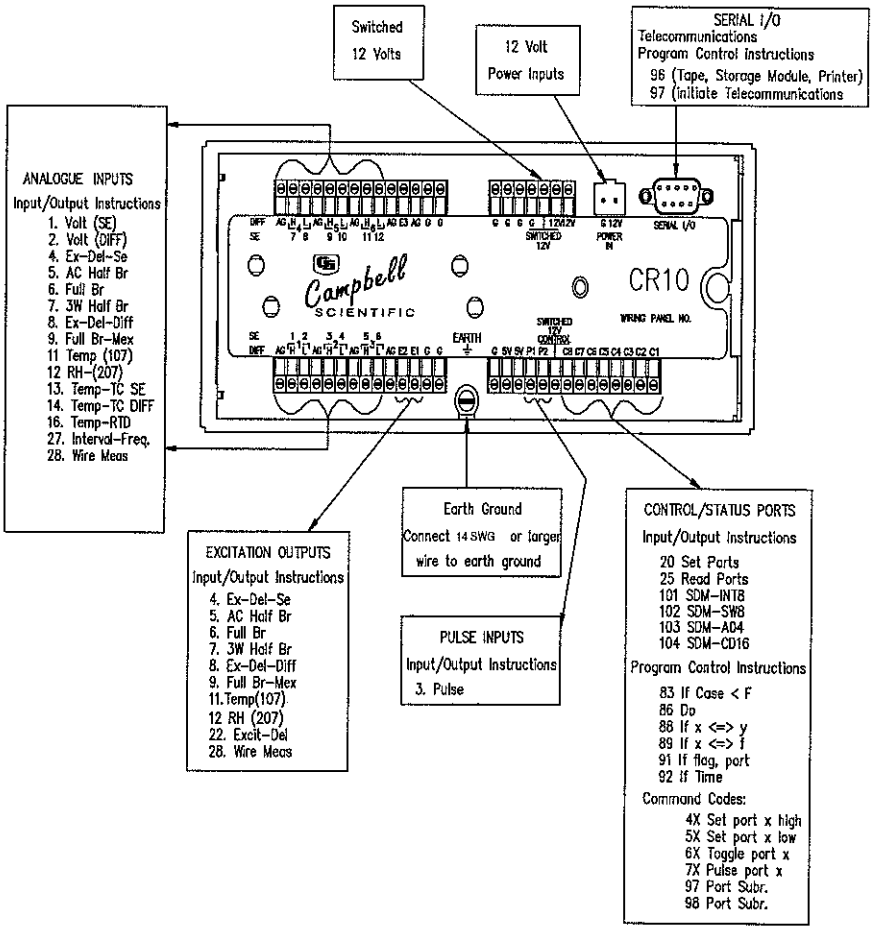


Figure OV-2 CR10 Wiring Panel/Instruction Access

Analogue Ground (AG)

The AG terminals are analogue grounds, used as the reference for single-ended measurements and to provide a return path for excitation current.

12V and Power Ground (G) 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. For protection against transient voltage spikes, power ground should be connected to a good earth ground (see Section 14).

5V Outputs

The two 5V outputs (accurate to ±0.2%) are commonly used to power peripherals such as the AVW1/AVW4 Vibrating Wire Interfaces, or the QD1 Incremental Encoder Interface. The 5V outputs are common with pin 1 on the 9-pin serial connector; 200mA is the maximum combined output.

Serial I/O

The 9-pin serial I/O port contains lines for serial communication between the CR10 and external devices such as computers, printers, Storage Modules, etc. The I/O port has a 5V DC power line which is used to power peripherals such as Storage Modules or telephone modems. The same 5V DC supply is used for the 5V outputs on the lower terminal strip. Section 6 contains technical details on serial communication.

NOTE This port does *not* have the same configuration as the 9-pin serial ports currently used on many personal computers.

Switched 12V Supply

Please refer to 'Selected Operating Details' at the front of the manual.

OV1.2 Connecting Power to the CR10

The CR10 can be powered by any 12V DC source. First connect the positive lead from the power supply to one of the 12V terminals and then connect the negative lead to one of the power ground (G) terminals. The Wiring Panel power connection is reverse polarity protected. See Section 14 for details on power supply connections.

CAUTION The metal surfaces of the CR10, CR10WP Wiring Panel, 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.

OV2. Memory and Programming Concepts

The CR10 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 CR10 waits for the remainder of the execution interval and then executes the table again starting at the beginning.

The execution interval generally determines the interval at which measurements are made. The interval at which data is 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 at longer or irregular intervals.

Figure OV-3 represents the measurement, processing and data storage sequence, and the types of instructions used to accomplish these tasks.

OV2.1 Internal Memory

The CR10 has 64K bytes of random access memory (RAM), divided into five areas. The use of the Input, Intermediate, and Final Storage memory areas in the measurement and data processing sequence is shown in Figure OV-3. While the total size of these three areas remains constant, memory can be reallocated between the areas to accommodate different measurement and processing needs (see description of *A Mode in Section 1). The sizes of the two additional memory areas (system memory and program memory) are fixed. The five areas of RAM are:

1. **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.

2. **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 program 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.

3. **Final Storage** — Final processed values are stored here for transfer to a Storage Module or printer 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 your program.* Approximately 29,900 locations are allocated to Final Storage on power-up. This number is reduced if Input or Intermediate Storage is increased.

4. **System Memory** — This is used for overhead tasks such as compiling programs and transferring data. You cannot access this memory.

5. **Program Memory** — This is the memory available for your programs entered in program tables 1 and 2, and subroutine table 3.

OV2.2 CR10 Instruction Types

Figure OV-3 illustrates the use of three different instruction types which act on data. The fourth type, Program Control, is used to control output times and to vary program execution by using techniques such as loops or conditional tests. Instructions are identified by numbers.

1. **INPUT/OUTPUT INSTRUCTIONS** control the terminal strip inputs and outputs (the sensor is the source — see Figure OV-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. (See description of Instructions 1-28 and 101-104 in Section 9.)

2. **PROCESSING INSTRUCTIONS** perform numerical operations on values located in Input Storage (source) and store the results back in Input Storage (destination). These instructions can be used to develop complex algorithms to process measurements before Output Processing. (See description of Instructions 30-66 in Section 10.)

3. **OUTPUT PROCESSING INSTRUCTIONS** are the only instructions which store data in Final Storage (destination). Input Storage (source) values are processed over time to obtain averages. (See description of Instructions 69-82 in Section 11.)

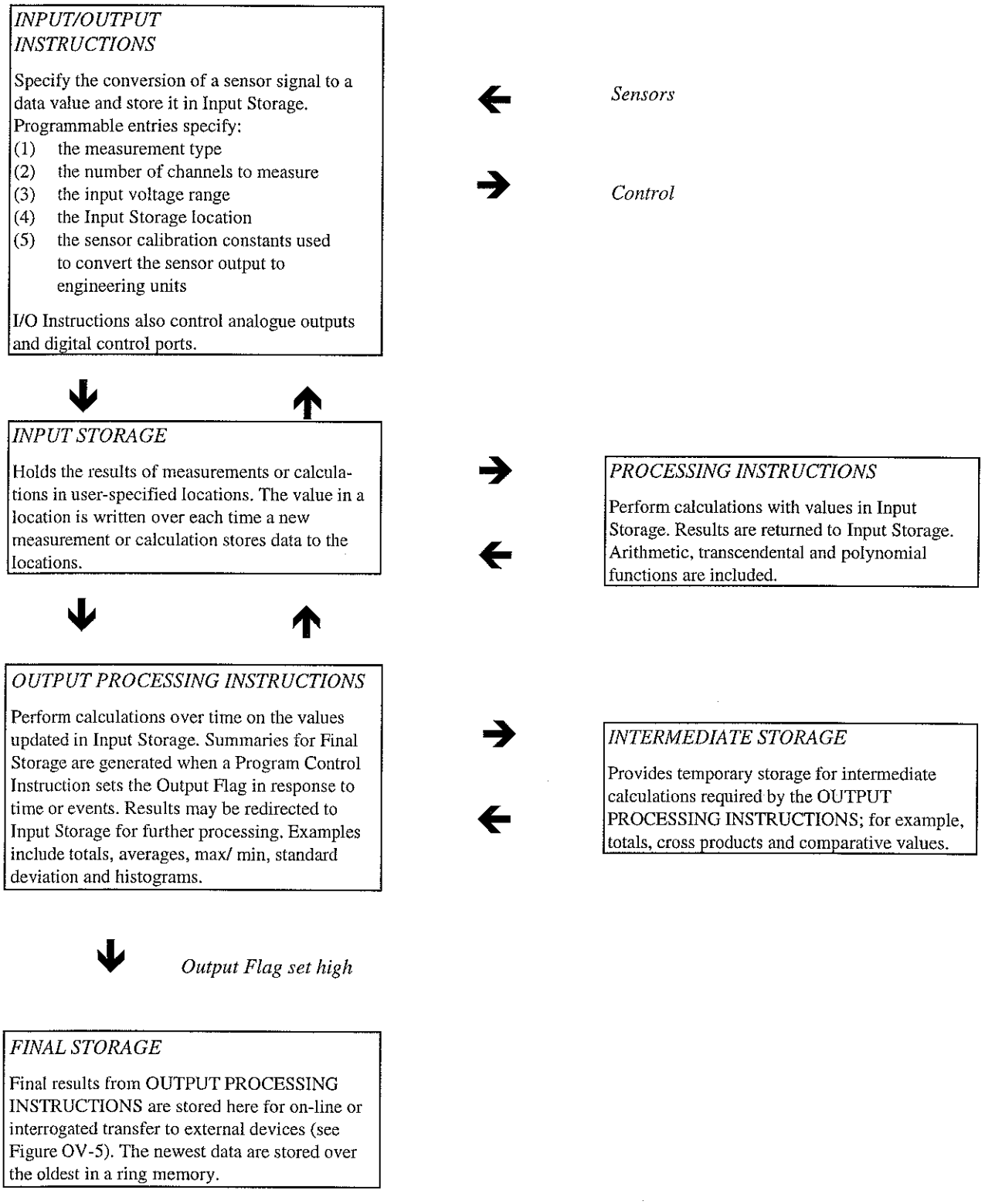


Figure OV-3 Instruction Types and Storage Areas

maxima, minima and so on. 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. The Output Processing Instructions check the Output Flag. If the flag is high, final values are calculated and output. With the Average instruction, for example, 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 again.

NOTE The Output Flag, Flag 0, is set high by a Program Control Instruction which must precede the Output Processing Instructions in your program.

4. **PROGRAM CONTROL INSTRUCTIONS** are used for logic decisions and conditional statements. They can set flags, compare values or times, execute loops, call subroutines and conditionally execute portions of the program. (See description of Instructions 83-98 in Section 12.)

OV2.3 Program Tables, Execution Interval and Output 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 each table is flexible, limited only by the total amount of program memory. If Table 1 is the only table programmed, the entire program memory is available for Table 1.

Table 1 and Table 2 have independent execution intervals, entered in units of seconds with an allowable range of 1/64 to 8191 seconds. Subroutine Table 3 has no execution interval; subroutines are only executed when called from Table 1 or Table 2 (see Figure OV-4).

The Execution Interval

The execution interval specifies how often the program in the table is executed, which is usually determined by how often the measurements are to be made. *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.

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 CR10 finishes processing the table and waits for the next execution interval before initiating the table. When an overrun occurs, decimal points may be shown on either side of the G on the display in the LOG mode (*0). (Overruns and table priority are discussed in Section 1.)

The Output Interval

The interval at which output occurs is independent of the execution interval, other than the fact that it must occur when the table is executed (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

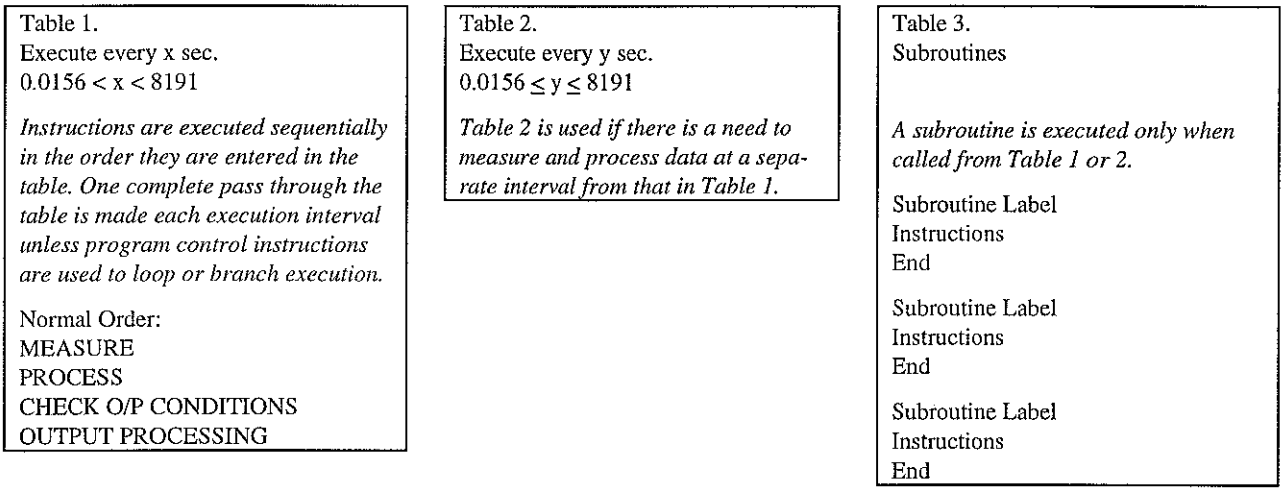


Figure OV-4 Program and Subroutine Tables

instruction setting the Output Flag determine the data output and its sequence. Each additional Output Array is created by another Program Control Instruction checking an output condition, followed by Output Processing Instructions defining the data set to write to Final Storage.

OV3. Communicating with the CR10

An external device must be connected to the CR10's Serial I/O port to communicate with the CR10. This may be either Campbell Scientific's portable CR10KD Keyboard/Display or a computer/terminal. The CR10KD is powered by the CR10 and connects directly to the serial port via the SC12 cable (supplied with the CR10KD). No interfacing software is required.

To communicate with any device other than the CR10KD, the CR10 enters its Telecommunications Mode and responds only to valid telecommunications commands. Within the Telecommunications Mode, there are two '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. 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 can be keyed in to the CR10 from the computer/terminal.

Campbell Scientific's PC208 Datalogger Support Software enables you to use IBM PC/XT/AT/PS-2's and compatibles for communicating with the CR10. This package contains a program editor (Edlog), a terminal emulator (GraphTerm), telecommunications (Telcom), a data reduction program (Split), and two programs to retrieve data from Storage Modules (SMCOM and CSMCOM).

NOTE The PC200 Starter Software includes Edlog and Term (a non-graphical version of GraphTerm). These two programs are sufficient to do the programming examples in this Overview.

To do the programming examples in Section OV5 you need to communicate with the CR10.

If you are using:	Then read:
A CR10KD	Section 0V3.1
The PC208 or PC200 Software	Section 0V3.2
A computer or terminal with telecommunications software	Section OV3.3 and Section 5

OV3.1 CR10 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 CR10.

If the Keyboard/Display is connected to the CR10 before the CR10 is powered up, the 'HELLO' message is displayed while the CR10 checks memory. The size of the usable system memory is then displayed (96 for 96K bytes of memory). When the CR10KD is connected after the CR10 has powered up, the display shows random characters until '*' is pressed to enter a mode.

OV3.2 Using the PC208 Terminal Emulator (GraphTerm)

For IBM PC-compatible computers, the PC208 software contains a terminal emulator program called GraphTerm. When using GraphTerm, the baud rate, port and modem types are specified and stored in a file for future use.

NOTE The Term program included with the PC200 Starter Software is a non-graphical version of GraphTerm. The Terminal Emulator option is identical to that of GraphTerm and is all you need for the programming examples.

The simplest and most common interface is the SC32A Optically Isolated RS232 Interface. The SC32A converts and optically isolates the voltages passing between the CR10 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 CR10 to the 9-pin port of the SC32A labelled 'Datalogger'. Connect the 'Terminal/Printer' port of the SC32A to the serial port of the computer with an SC25 cable or, if the computer has a 9-pin serial port, an SC25AT.

To establish the communication link between the computer and the CR10, you can either select the T option and send carriage returns or select the 'C' option to 'Call' the station (see PC208 User Guide). Once the link is active, issue the '7H' command to enter the Remote Keyboard State.

OV3.3 ASCII Terminal or Computer with Terminal Emulator

Devices which can be used to communicate with the CR10 include standard ASCII terminals and computers programmed to function as a terminal emulator.

Computer / Terminal Requirements

The basic requirements are:

1. There must be an asynchronous serial port to transmit and receive characters.
2. Communication protocol must be matched for the two devices.
3. The proper cable/interface must be used between the serial ports.
4. A computer must be programmed to function as a terminal.

While the connection between the computer/terminal and the CR10 may be via modem (phone, RF, or short haul), the most frequently used device for a short connection is the SC32A Optically Isolated RS232 Interface.

Most computer/terminal devices need RS232 input logic levels of -5V for logic low and +5V for logic high. Logic levels from the CR10's serial I/O port are 0V for logic low and +5V for logic high. The SC32A converts and optically isolates the voltages passing between the CR10 and the external terminal device. The SC32A is configured as Data Communications Equipment (DCE) for direct connection to Data Terminal Equipment (DTE), which includes most computers and terminals.

The SC12 Two-Peripheral Cable which comes with the SC32A is used to connect the serial I/O port of the CR10 to the 9-pin port of the SC32A labelled 'Datalogger'. Connect the 'Terminal/Printer' port of the SC32A to the serial port of the terminal with an appropriate cable (Campbell Scientific SC25 or equivalent for a 25-pin serial port configured as DTE).

Establishing Communication with the CR10

Communication software is available for most computers having a serial port. The software must be capable of the following communication protocol:

1. Configuring an asynchronous serial port for 8 data bits, 1 stop bit, no parity, and full duplex at baud rates of 300, 1200 or 9600 baud.
2. Transmitting characters typed on the keyboard out through the serial port.
3. Displaying characters/data received through the computer's serial port.

Once the computer is functioning as a terminal, initiate communications by sending the CR10 several carriage returns for the CR10 to match the baud rate and respond with '*'. Enter the 7H command to enter the Remote Keyboard State. At this point, the CR10 can be controlled using the keyboard commands described in Section OV4. For additional information on communications, see Sections 5 and 6.

OV4. Programming the CR10

A program is created by entering it directly into the datalogger or on a computer using the Campbell Scientific program Edlog. This Overview describes direct interaction with the CR10. Work through the direct programming examples in this Overview before using Edlog; this will teach you the basics of CR10 operation. Section OV4.5 describes options for loading the program into the CR10.

OV4.1 Functional Modes

Interaction between you and the CR10 is broken into different *functional modes* (e.g. programming the measurements and output, setting the time, manually initiating a block data transfer to Storage Module, etc.). The modes are also referred to as Star (*) Modes since they are accessed by first keying *, then the mode number or letter. Table OV-1 lists the CR10 Modes.

Table OV-1 * Mode Summary

Key	Mode
*0	Log data and indicate active program 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 Final Storage data
*8	Final Storage data transfer to Storage Module, printer or cassette tape
*9	Storage Module commands
*A	Memory allocation/reset
*B	Signature/status
*C	Security
*D	Save/load program

OV4.2 Key Definition

Keys and key sequences have specific functions when using the CR10KD keyboard or a computer/terminal in the Remote Keyboard State. Table OV-2 lists these functions. In some cases, the exact action of a key depends on the mode the CR10 is in and is described with the mode in the manual.

Table OV-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 a 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 (*1, *2, *3) or to previous Output Array in Final Storage (*7)
#D	Delete entire instruction
#0	(then A or RETURN) Back up to the start of current array

When using a computer/terminal to communicate with the CR10 there are some keys available in addition to those found on the CR10KD. Table OV-3 lists these keys.

Table OV-3 Additional Keys Allowed in Telecommunications

Key	Action
-	Change sign, index (same as C)
RETURN	Enter/advance (same as A)
:	Colon (used in setting time)
S or CTRL-S	Stops transmission of data (10 second time-out; any character restarts)
C or CTRL-C	Aborts transmission of data

OV4.3 Programming Sequence

In routine applications, the CR10 measures sensor output signals, processes the measurements over some time interval and stores the processed results. A generalised 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 (see 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 produce output based on time.
   - Instruction 86 to produce output every execution interval.
   - Instruction 88 or 89 to produce output based on a comparison of values in input locations.

   This instruction must precede the Output Processing Instructions which store data in Final Storage.

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

6. Repeat steps 4 to 6 for additional outputs at 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 two-minute execution interval and a five-minute output interval, the program will only be executed on the even multiples of the five minute intervals, not on the odd. Data will thus be output every ten minutes instead of every five minutes.

Execution intervals and output intervals set with Instruction 92 are synchronised with real time, starting at midnight.

OV4.4 Instruction Format

Instructions are identified by an instruction number. Each instruction has a number of parameters that give the CR10 the information it needs to execute the instruction.

The CR10 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 '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 '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 monitored 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 was 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, and so on.

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

OV4.5 Loading a Program

Programs are loaded into the CR10 in one of the following ways:

1. Keyed in using the CR10KD Keyboard/Display.
2. Loaded from a pre-recorded listing. The program listing may come from one of two sources:
   a. A PC (transferred using Term or GraphTerm)
   b. A Campbell Scientific Storage Module
3. Loaded from internal PROM (special software required) or Storage Module on power-up.

A program is created either by entering it directly into the datalogger as described in Section OV5, or on a PC using the Edlog program.

Programs on disk can be copied to a Storage Module with SMCOM or CSMCOM. Using the *D Mode to save or load a program from a Storage Module is described in Section 1.

It is possible (with special software) to create a PROM (Programmable Read Only Memory) that contains a datalogger program. With this PROM installed in the datalogger, the program will automatically be loaded and run when the datalogger is powered up, requiring only that the clock be set.

The program on power-up function can also be achieved by using a Campbell Scientific Storage Module. Up to eight programs can be stored in the Storage Module. If the Storage Module is connected when the CR10 is powered up the CR10 automatically loads program number 8, provided that a program 8 is loaded in the Storage Module.

OV5. Programming Examples

This section guides you through some simple programming examples. There is a brief explanation of each step to help you follow the logic. When the examples use 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 CR10, connect the power leads as described in Section OV1.2. Next, connect the CR10 to either a CR10KD Keyboard/Display or a terminal (see Section OV-3). If using a terminal, use the 7H command to get into the Remote Keyboard State. 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 Keyboard/Display is connected to the CR10 before the CR10 is powered up, the 'HELLO' message is displayed while the CR10 checks memory. The size of the usable system memory is then displayed (96 for 96K bytes of memory). When the CR10KD is connected after the CR10 has powered up, the display shows random characters until '*' is pressed to enter a mode.

OV5.1 Sample Program 1

In this example the CR10 is programmed to read its own internal temperature (using a built-in thermistor) every five seconds and to send the results to Final Storage.

Press these keys:	**Display will show (ID: Data):**	**Explanation of this step:**
*	00:00	*Enter a functional 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 five seconds.*
A	01:P00	*Enter the five second execution interval and advance to the first program instruction location.*
17	01:P17	*Key in Instruction 17, which directs the CR10 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 number and advance to the second program instruction.*

The CR10 is now programmed to read the internal temperature every five seconds and place the reading in Input Storage location 1. The program can be compiled and the temperature displayed.

Press these keys:	Display will show (ID: Data):	Explanation of this step:
*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 CR10 has measured the sensor output and stored the result again. The internal temperature is now 21.423°C. The value is updated every five seconds when the table is executed. At this point the CR10 is measuring the temperature every five seconds and sending the value to Input Storage. No data is being saved. The next step is to have the CR10 send each reading to Final Storage (remember, the Output Flag must be set first).*
*1	01:00	*Exit *6 Mode. Enter program table 1.*
2A	02:P00	*Advance to 2nd instruction location (this is where we left off).*
86	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).*
10	01:10	*This command sets the Output Flag. (Flag 0)*
A	03:P00	*Enter 10 and advance to third program instruction.*
70	03:P70	*The SAMPLE instruction. This directs the CR10 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.*
	02:0000	*Enter 1 and advance to second parameter (Input Storage location to sample).*
	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 CR10 is now programmed to measure the internal temperature every five seconds and send each reading to Final Storage. Values in Final Storage can be viewed using the *7 Mode.*

Press these keys:	Display will show (ID: Data):	Explanation of this step:
*7	07: 13.000	*Enter *7 Mode. The Display 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.*

Notice that there are no date and time tags on the data. These can be obtained by using Instruction 77, as shown in the next example.

If a terminal is used to communicate with the CR10, telecommunications commands (see Section 5) can be used to view all the elements of an Output Array (in this case the ID and temperature) at the same time.

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 this time it is used as a reference temperature for the differential voltage measurement of a type T (copper-constantan) thermocouple (new CR10s are delivered with a short type T thermocouple connected to differential channel 5).

When using a type T thermocouple, the copper lead (white) is connected to the high input of the differential channel, and the constantan lead (blue) 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 CR10 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 (see Section 13 for more information on thermocouple measurements).

NOTE *The internal temperature of the CR10 is not a suitable reference temperature for precision thermocouple measurements.* It is used here for training purposes only. To make thermocouple measurements with the CR10, you should use the Campbell Scientific Thermocouple Reference, Model 10TCRT, and make the reference temperature measurement with Instruction 11.

Instruction 14 directs the CR10 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 one TC, they could be wired to sequential channels, and the number of thermocouples entered as the number of repetitions. The CR10 would automatically advance through the channels sequentially and measure the output of all the thermocouples.

Parameter 2 is the voltage range to use when making the measurement. The output of a type T thermocouple is approximately 40μV per degree C difference in temperature between the two junctions. The ±2.5mV scale will therefore 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 ±2.5mV range is 0.33μV or 0.008°C.

Parameter 3 is the analogue 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 were five repetitions and the first measurement was stored in location 3, the final measurement would 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 thermocouple temperature is measured once a minute and the day, time and average temperature are output every hour. Once a day the day, time, maximum and minimum temperatures and the times they occur are output.

The example then uses Instruction 96 to send Final Storage data to a Storage Module (see Sections 4 and 12).

The first example program described program entry one keystroke at a time. This second example does not specifically tell you to press the 'A' key. Remember, 'A' is used to enter and/or advance (i.e. between each line in the example below). The format used to describe this example 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 read their complete definitions in the manual.

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: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 (2.5mV, 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:P92		*If Time instruction*
	01:0	*0 minutes into the interval*
	02:60	*60 minute interval*
	03:10	*Set Output Flag 0*

The CR10 is programmed to measure the thermocouple temperature every sixty seconds. The If Time instruction sets the Output Flag at the beginning of every hour. Next, the Output Instructions for time and average are added.

Instruction # (Loc:Entry)	Parameter (Par#:Entry)	Description
04:P77		*Output Time instruction*
	01:110	*Store Julian day, hour, and minute*
05:P71		*Average instruction*
	01:1	*one repetition*
	02:2	*Location 2 — source of TC temperatures to be averaged*

To obtain daily output, the If Time instruction is again used to set the Output Flag and is followed by the Output Instructions to store time and the daily maximum and minimum temperatures, along with the times at which they occur.

Any Program Control Instruction which is used to set the Output Flag high will set it low if the conditions are not met for setting it high. Instruction 92 above sets the Output Flag high every hour. The additional Output Instructions which will now be entered do not produce output every hour because they are preceded by another Instruction 92 which sets the Output Flag high at midnight (and sets it low at any other time). This is a special feature of Flag 0. The Output Flag is set low at the start of each table (see Section 3 for more information on the use of flags).

Instruction # (Loc.:Entry)	Parameter (Par.#:Entry)	Description
06:P92		*If Time instruction*
	01:0	*0 minutes into the interval*
	02:1440	*1440 minute interval (24 hrs.)*
	03:10	*Set Output Flag 0*
07: P77		*Output Time instruction*
	01:100	*Store Julian day*
08: P73		*Maximize instruction*
	01:1	*One repetition*
	02:10	*Output the time of the daily maximum in hours and minutes*
	03:2	*Data source is Input Storage location 2.*
09: P74		*Minimize instruction*
	01:1	*One repetition*
	02:10	*Output the time of the daily minimum in hours and minutes*
	03:2	*Data source is Input Storage location 2.*

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.

Instruction # (Loc:Entry)	Parameter (Par#:Entry)	Description
10:P96		*Activate Serial Data Output.*
	1:71	*Output Final Storage data to Storage Module.*

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	05:00	*Advance to location for year.*
93	05:93	*Key in year (1993).*
A	05:0000	*Enter and advance to location for Julian day.*
197	05:197	*Key in Julian day.*
A	05:0021	*Enter and advance to location for hours and minutes (24hr time)*
1324	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 Mode, compile Table 1, start logging data.*

OV5.3 Editing an Existing Program

When you edit an existing program in the CR10, entering a new instruction inserts the instruction; entering a new parameter value 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. Pxx is displayed in the data portion of the display, where xx is the instruction number). Key in the instruction number, and then press A. The new instruction is 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 are pushed down to follow the inserted instruction.

Delete an instruction by advancing to the instruction number (Pxx in display) and keying #D.

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 pressed.

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 OV-5 summarises 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 such as a Storage Module. On a regular schedule, either the data is transferred from the storage device to a portable computer in the field, or the storage device is brought back to the office/lab where the data is transferred to the computer. In the latter case, another 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 transfer all the data that has accumulated in Final Storage since the last visit.
3. Retrieve the data over a telecommunications link such as RF, telephone, short haul modem or satellite. This can be done under program control or by regularly scheduled polling of the dataloggers. Campbell Scientific's Telcom program automates this process for IBM PC/XT/AT/PS-2's and compatibles.

Regardless of which method is used, the retrieval of data from the datalogger does *not* erase that data from Final Storage. The data remains in the ring memory until:

- it is written over by new data (see Section 2) *or*
- memory is reallocated (see Section 1) *or*
- the power to the datalogger is turned off.

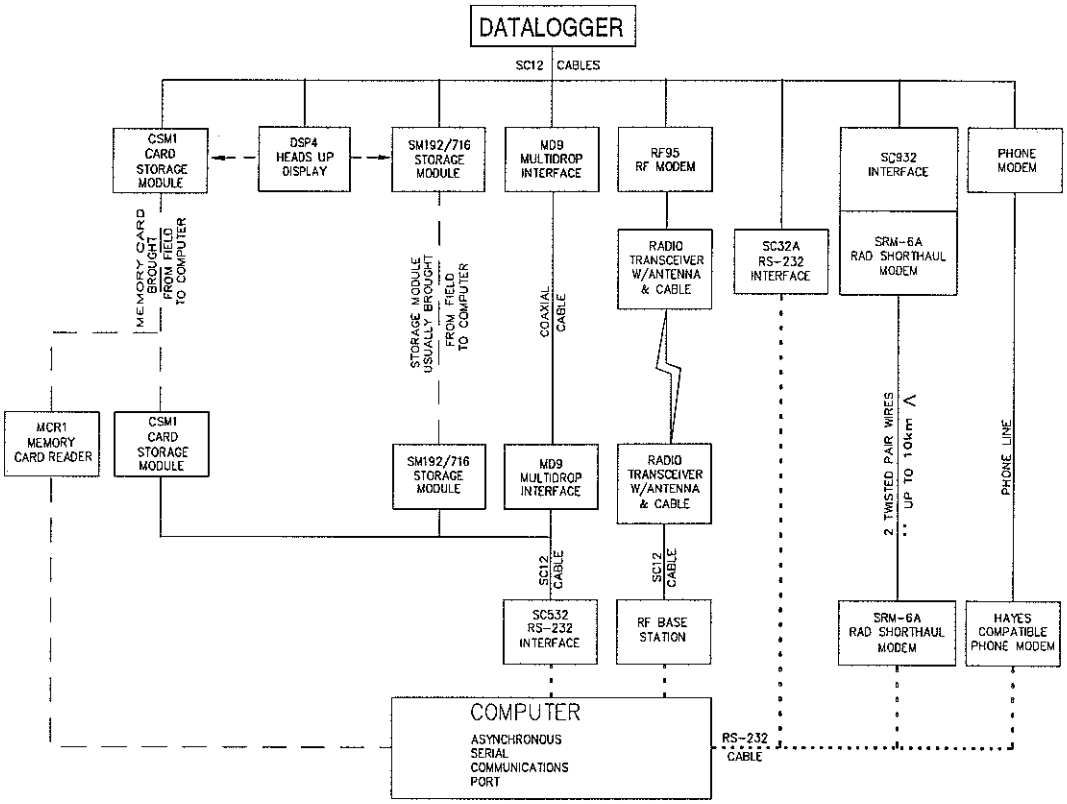
Table OV-4 lists the instructions used with the various methods of data retrieval.

Table OV-4 Data Retrieval Methods and Related Instructions

Cassette Tape	Storage Module	Printer, other Serial Device	Telecommunications (RF, Phone, Short Haul, SC32A)
Inst. 96	Inst. 96	Inst. 96	Inst. 97
*8	*8*8		
	*9Inst. 98	(Telecommunications Commands)	
	*D	*D	

Table OV-5 Data Retrieval Sections in Manual

To learn about this Instruction or Mode:	Read this section:
Instr. 96	4.1, 12
Instr. 97	12
Instr. 98	12
*8	4.2
*9	4.4
*D	1.8
Telecommunications	5



NOTES:

1. Additional methods of data retrieval are:
   - Cassette Tape
   - Satellite transmission
   - Direct dump to printer
2. The DSP4 Heads up Display allows you 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 RS232 interfaces have a female 25-pin RS232 connector.

Figure OV-5 Data Retrieval Hardware Options

OV7. Specifications

Valid for an ambient temperature range of -25°C to +50°C unless otherwise specified

ANALOGUE INPUTS

NUMBER OF CHANNELS: 12 single-ended or 6 differential with any combination, software selectable.

CHANNEL EXPANSION: The AM416 Analogue Multiplexer allows an additional 64 single-ended channels to multiplex into four CR10 single-ended channels.

ACCURACY OF VOLTAGE MEASUREMENTS AND ANALOGUE OUTPUT VOLTAGES: 0.2% of FSR, 0.1% of FSR (0 to 40°C)

RANGE AND RESOLUTION: Ranges are software selectable for any channel.

Resolution for single-ended measurements is twice the value shown.

Full Scale Range	Resolution
±2500mv	333µV
±250mv	33.3µV
±25mv	3.33µV
±7.5mv	1.00µV
±2.5mv	0.33µV

INPUT SAMPLE RATES: The fast or slow A/D conversion on the four lowest input ranges uses a 250µs or 2.72ms signal integration time, respectively. Two integrations, separated in time by half of an AC line cycle, are used with the 60Hz or 50Hz noise rejection options.

Differential measurements include a second sampling with reversed input polarity to reduce thermal offset and common mode errors. Input sample rates are governed by the time required to measure and convert the result to engineering units.

Fast single-ended voltage:	2.6ms
Fast differential voltage:	4.2ms
Slow single-ended voltage:	5.1ms
Slow differential voltage:	9.2ms
Diff. w/50 Hz rejection:	29.4ms
Fast diff. thermocouple:	8.6ms

INPUT NOISE VOLTAGE:

Fast differential	-	0.82µV RMS
Slow differential	-	0.25µV RMS
Diff. w/50 Hz rejection	-	0.18µV RMS

COMMON MODE RANGE: ±2.5V

DC COMMON MODE REJECTION: >140dB

NORMAL MODE REJECTION: 70dB at 50Hz with 50Hz rejection option

INPUT CURRENT: 3nA max.

INPUT RESISTANCE: 200GΩ

EXCITATION OUTPUTS

DESCRIPTION: The CR10 has 3 switched excitations, active only during measurement, with only one output active at any time. The off state is high impedance.

RANGE: ±2.5V

RESOLUTION: 0.67mV

ACCURACY: Same as voltage input

OUTPUT CURRENT: 20mA @ ±2.5V, 35mA @ ±2.0V, 50mA @ 1.5V

FREQUENCY SWEEP FUNCTION: A swept frequency square wave output between 0 and 2.5V is provided for vibrating wire transducers. Timing and frequency range are specified by the instruction.

PERIOD AVERAGING MEASUREMENTS

DEFINITION: The time period for a specified number of cycles of an input frequency is measured, then divided by the number of cycles to obtain the average period of a single cycle.

INPUTS: Any single-ended analogue channel; signal dividing or AC coupling is normally required.

INPUT FREQUENCY RANGE:

Range Code	Peak to Peak Volts Required @ Max. Freq.*	Maximum Frequency
1	2mv	8kHz
2	3mv	20kHz
3	12mv	50kHz
4	2000mv	200kHz

*AC voltage: must be centred around CR10 ground.

REFERENCE ACCURACY:
-25°C to 0°C ±80 ppm, 0° to +50°C ±30 ppm

RESOLUTION: ±100ns divided by the number of cycles measured. Resolution is reduced by signal noise and for signals with a slow transition through the zero voltage threshold.

TIME REQUIRED FOR MEASUREMENT: Signal period times the number of cycles measured plus 1.5 cycles; minimum measurement time is 2ms.

RESISTANCE & CONDUCTIVITY MEASUREMENTS

ACCURACY: 0.015% of full scale bridge output, limited by the matching bridge resistors. The excitation voltage should be programmed so the bridge output matches the full scale input voltage range.

MEASUREMENT TYPES: 6-wire and 4-wire full bridge, 4-wire, 3-wire and 2-wire half bridge. Bridge measurements are ratiometric and dual polarity to eliminate thermal emfs. AC resistance measurements use a dual polarity 750µs excitation pulse for ionic depolarisation, with the signal integration occurring over the last 250µs.

PULSE COUNTERS

NUMBER OF PULSE COUNTER CHANNELS: selectable as 1 fast and 2 slow 16-bit channels.

MAXIMUM COUNT RATE: 2000Hz, slow counters: 250kHz, fast counters. Pulse counter channels scanned at 8Hz.

MODES: Switch closure, high frequency pulse, and low level AC.

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

HIGH FREQUENCY PULSE MODE
Minimum Pulse Width: 2µs
Maximum Input Frequency: 250kHz
Voltage Thresholds: Count on transition from below 1.5V to above 3.5V
Maximum Input Voltage: 0-5V (up to ±20V with external resistor)

LOW LEVEL AC MODE
(Typical of magnetic pulse flow sensors, selected anemometers etc.)

Min. AC Input Voltage:	6mV RMS
Input Hysteresis:	11mV
Max. AC Input Voltage:	20V RMS

AC Input (RMS)	Range
20mV	1Hz to 100Hz
50mV	0.5Hz to 400Hz
150mV to 20V	0.3Hz to 1000Hz

(Consult Campbell Scientific if higher frequencies are desired.)

DIGITAL I/O PORTS

Eight ports, software selectable as binary inputs or control outputs.

OUTPUT VOLTAGES (no load): high: 5 ± 0.1V low: 0.1V

OUTPUT RESISTANCE: 500 ohms

INPUT STATE: high: >3V: low: <0.8V

INPUT RESISTANCE: 100kΩ

SDI-12 INTERFACE STANDARD

This communication protocol, developed for microprocessor-based hydrological and environmental sensors is available as a software option in the CR10.

SENSOR CONNECTIONS: Digital I/O Port 8 (for asynchronous communication), 12V power and ground. Up to ten SDI-12 sensors can be connected to a CR10.

TRANSIENT PROTECTION

All input and output connections to the CR10 module are protected using RC filters or transzorbs connected to a heavy copper bar between the circuit card and the case. The CR10WP Wiring Panel includes additional spark gap and transzorb protection.

CPU AND INTERFACE

PROCESSOR: Hitachi 6303

MEMORY: 32K ROM, 64K RAM

STORAGE CAPACITY: The CR10 stores 29,908 data values in Final Storage memory. The datalogger has a capacity of 1986 bytes available for programming.

CR10 DISPLAY: 8-digit LCD 12.7mm digits

PERIPHERAL INTERFACE: 9-pin D-type connector for keyboard/display, storage module, cassette, modem, printer and RS232 adapter. Baud rates selectable at 300, 1200, 9600 and 76,800.

CLOCK ACCURACY: ±1 minute per month

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

POWER REQUIREMENTS

VOLTAGE: 9.6 to 16 volts

TYPICAL CURRENT DRAIN: Quiescent: 0.7mA; During processing: 13mA; During analogue measurement: 35mA average, 50mA peak.

BATTERIES: Any 12V battery can be connected as a primary power source. Enclosures with power supply options are available.

DIMENSIONS

SIZE: 198 x 89 x 38mm: 229 x 89 x 74mm with CR10WP Wiring Panel. Input connectors extend length by 3.8mm.

WEIGHT: 0.91Kg

GUARANTEE

Three years against defects in materials and workmanship.

July 1994

Section 1. Functional Modes

You can control different aspects of the CR10's operation by switching between functional modes. This section shows you how to use the modes associated with programming. Other modes (such as those associated with data retrieval) are described in other sections of the User Guide.

1.1 Program Tables — *1, *2 and *3 Modes

The CR10's data acquisition and processing functions are controlled by user-entered instructions contained in program tables. Programming can be separated into two tables, each having its own user-entered execution interval. A third table is available for programming subroutines, which can 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.

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

1.1.1 Execution Interval

The execution interval is entered in units of seconds as shown in Table 1-1.

Table 1-1 Valid Execution Intervals (All times in seconds)

For execution intervals in the range:	You must enter a value which is a multiple of:
0.015625 to 1	0.015625 (1/64)
1 to 31.875	0.125 (1/8)
32 to 8191	1

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

The *sample rate* for a CR10 measurement is the rate at which the measurement instruction can be executed (i.e. make the measurement, scale the result with the instruction's multiplier and offset, and place the result 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 execution interval for a table is less than the time required to process that table, the CR10 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 your 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 (see Section 1.6) or using the telecommunications 'A' command (see Section 5). An overrun also causes decimal points to appear on both sides of the sixth digit of the CR10KD display. (The decimal points do 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 you want to sample some phenomena every 0.125 seconds and output processed data every ten minutes. The processing time of the table which does this is less than 0.125 seconds except when output occurs (every ten minutes). With final output the processing time is one second. With the execution interval set at 0.125 seconds, and a one second lag between samples once every ten minutes, eight measurements out of 4800 (.17%) are missed: this is an acceptable statistical error for most populations.

1.1.2 Subroutines

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

Subroutines labelled 97 or 98 have the extra feature of being executed when a digital control port goes high (port 7 for subroutine 97 and port 8 for subroutine 98). Either subroutine will interrupt Tables 1 and 2 (see Section 1.1.3) when the appropriate port goes high. Port 7 cannot wake the processor; subroutine 97 will be executed at the next 1/8 second interval after the port goes high. Port 8, however, will wake the processor 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 and 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 is interrupted. After Table 1 processing is completed, Table 2 processing resumes at the point of interruption. If the execution interval of Table 2 coincides with Table 1, Table 1 is executed first, followed by Table 2.

NOTE 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.

Special subroutines 97 and 98, initiated by a port going high (see above), can interrupt either Table 1 or 2, or can occur when neither Table is being executed. These subroutines *can* interrupt a table while the Output Flag (flag 0) is set. When the port activating subroutine 97 or 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 order is subroutine 98, subroutine 97, Table 1, Table 2. If subroutines 97 and 98 are both pending (ports go high at the same time or both go high during the execution of the same instruction in one of the tables), subroutine 98 will be executed first. Subroutines 97 and 98 cannot interrupt each other, and cannot be interrupted by a program table.

While subroutine 97 or 98 is being executed as a result of the appropriate port 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 Compiling a Program

When a program is first loaded into the CR10, or if any changes are made in the *1, *2, *3, *A or *C Modes, the program must be compiled before it will run. The compile function checks for programming errors and optimises program information for use during program execution. If errors are detected, the appropriate error codes are indicated on the display (see Section 3 for a description of the error codes). The compile function is executed when the *0 , *6 or *B Modes are entered and before saving a program listing in the *D Mode. The compile function is only executed after a program change has been made, so that subsequent use of any of these Modes simply returns to the Mode without recompiling.

NOTE When the *0, *B or *D 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 are *unaltered*, as are the states of flags, control ports and the timer (Instruction 26). Compiling always zeros Intermediate Storage.

1.2 Setting and Displaying the Clock — *5 Mode

The *5 Mode is used to display the time or change the year, day or time. When *5 is entered, the time is displayed and 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.

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

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

Changing the time also affects the output and execution intervals during which the time is changed. Because the time can only be set with a one second resolution, execution intervals of one second or less are unaffected by resetting the time, whereas a longer execution interval may be shortened or lengthened for one execution of the program. 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 Sequence of Time Parameters in *5 Mode

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

NOTE When entering a new value for hours and minutes, enter the figures only; no colon is required.

1.3 Displaying and Altering Input Storage, 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 digital control ports. If the *6 Mode is entered immediately following any new entries or changes in program tables, the compile function is executed and program execution begins.

NOTE Data values contained in Input Storage 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.

1.3.1 Displaying and Altering Input Storage

When you enter ***6**, the keyboard/display will read '**06:0000**'. You can advance to view the value stored in input location 1 by pressing A. To go directly to a specific location, key in the location number before pressing 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 two digits of the location number. If the value stored in the location being monitored is the result of a program instruction, the value displayed 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.

Table 1-3 *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
O	Display/alter ports
#	Display current location and allow a location no. to be keyed in, followed by A to jump to that location

Input locations can be used to store parameters for use in computations. You can store a value, or change an existing value, by pressing C while monitoring the location, followed by the desired number and A.

If an algorithm requires parameters to be modified manually during execution of the program *without interruption of the Table execution process*, the *6 Mode can be used to change parameters stored in input locations. (If parameters do not need modification, it is better to load them from the program using Instruction 30.) If initial parameter values need to be in place before program execution starts, use Instruction 91 at the beginning of the program table to prevent execution until a flag is set (see 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).

NOTE If any program tables (*1, *2 or *3) are altered and compiled in the *0 Mode after values have been entered into input locations using the *6C function, all values entered using *6C are set to zero. To preserve these values, always compile in the *6 Mode after altering the program tables.

1.3.2 Displaying and Toggling User Flags

If you press D while the CR10 is displaying a location value, the current status of the user flags is displayed in the following format: '00:010010'. The digits represent the flags; the left-most digit is flag 1 and right most is flag 8. A '0' indicates the flag is clear and a '1' indicates the flag is set. 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 start the execution of Table 2 manually: enter Instruction 91 as the first instruction in Table 2. The first parameter is 25 (do if flag 5 is low) and the second parameter is 0 (go to end of program table). If flag 5 is low , all subsequent instructions in Table 2 are 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 current status of the control ports can be displayed by keying 0 (zero) while looking at an input location (e.g. ***6A0**). Ports are displayed left to right as C8, C7, ... , C1 (exactly opposite to the flags). A port configured as an output can be toggled by keying its number while in the port display mode. There is no effect on ports configured as inputs.

On power up all ports are configured as inputs. Instruction 20 is used to configure a port as an output. Ports are also automatically configured as outputs by any program control commands which use the port as an output (i.e. pulse, set high, set low, toggle).

1.4 Compiling and Logging Data — *0 Mode

When the *0 Mode is entered after programming the CR10, a program compile function is executed and the display shows 'LOG' followed by the program table numbers that were enabled at compilation time. 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 minimise current drain, the CR10 should be left in the *0 Mode when logging data.

1.5 Memory Allocation — *A Mode

1.5.1 Internal Memory

The CR10 has two sockets for Random Access Memory (RAM) and one for PROM (programmable read only memory). The standard CR10 has 64K of RAM: a 32K RAM chip in each socket. Earlier versions had an 8K RAM chip in each socket. Appendix F describes how to change RAM and PROM chips.

When the CR10 is powered up with the CR10KD attached, the CR10KD displays **HELLO** while performing a self check. The total system memory (RAM and ROM) is then displayed in kilobytes. The size of RAM can be displayed in the *A Mode.

There are 1986 bytes allocated to program memory. This memory can be used for one table or shared among all tables. Tables 3-5 to 3-7 list the amount of memory used by program instructions.

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

The results of Output Instructions (data used for a permanent record) are stored in Final Storage when the Output Flag is set (see Section 3). The data in Final Storage can be monitored using the *7 Mode (see Section 2).

Table 1-4 Memory Allocation in the CR10 (32K ROM, 64K RAM)

Default Allocation

	Program Memory	System Memory	Input Storage	Intermediate Storage	Final Storage Area 1	Final Storage Area 2
Bytes	1986	3302	112	256	59,816	0
Locations			28	64	29,908	0

Maximum Reallocation From Final Storage

Maximum No. of Input + Intermediate Storage Locations	Minimum No. of Final Storage Locations Area 1 + Area 2
6,862	16,368

Notes: 1. 28 is the minimum number of Input Storage locations.
2. 768 is the minimum number of Final Storage Area 1 locations.
3. 64 bytes of RAM are not used (32 in each chip).

Intermediate Storage acts as a scratch pad for Output Processing Instructions. It is used to store the results of intermediate calculations necessary for averages, standard deviations, histograms, etc.. You cannot access Intermediate Storage.

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

Table 1-4 lists the basic memory functions and the amount of memory allocated to them.

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, and Final Storage Area 1.
2. Repartition this memory.
3. Check the number of bytes remaining in program memory.
4. Erase Final Storage.
5. Completely reset the datalogger (just as if power were turned off and then on again).

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, and Final Storage Area 2 memories are increased. When they are reduced, Final Storage Area 1 memory is increased.

NOTE Allocation of extra Input and Intermediate Storage locations does *not* change Final Storage Area 2 and therefore, the data in this area is preserved.

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 four windows. Table 1-5 describes what the values in the *A Mode represent.

The number of memory locations allocated to Input, Intermediate and Final Storage Area 2 default at power-up to the values in Table 1-3. The size of Final Storage is determined by the size of RAM.

You can change the sizes of Input, Intermediate and Final Storage Area 2 by keying in the desired value and entering it by pressing A. One Input or Intermediate Storage location can be exchanged for two Final Storage locations. The size of Final Storage Area 1 is adjusted automatically.

The maximum size of Input and Intermediate Storage and the minimum size of Final Storage are determined by the size of RAM chips installed (see Table 1-4). Input and Intermediate Storage are confined to the same RAM chip as system and program memory; they cannot be expanded onto the second chip, which is always entirely dedicated to Final Storage. A minimum 28 Input and 768 Final Storage Area 1 locations are *always* retained. The size of Intermediate Storage can be reduced to zero.

Table 1-5 Description of *A Mode Data

Keyboard Entry	Display ID: Data	Description of Data
*A	01: XXXX	Input Storage Locations. This value can be changed by keying in the desired number (minimum of 28, maximum limited by available memory and constraints on Final Storage).
A	02: XXXX	Intermediate Storage Locations. This value can be changed by keying in the desired number (minimum of 0, maximum limited by available memory and constraints on Input and Final Storage).
A	03: XXXXX	Final Storage Area 2 Locations. Changing this number automatically reallocates Final Storage Area 1 (minimum of 0, maximum limited by available memory.)
A	04: XXXXX	Final Storage Area 1 Locations. This number is automatically altered when the number of memory locations in Input, Intermediate or Final Storage Area 2 are changed (minimum of 768).
A	05: XXXXX	Bytes free in program memory. **Key in '1986' to completely reset the CR10.**

Intermediate Storage and Final Storage Area 1 are erased when memory is repartitioned. This feature can be used to clear the memory without altering the CR10 program. The number of locations does not actually need to be changed; the same value can be keyed in and entered. Final Storage Area 2 is protected when Input and/or Intermediate Storage is reallocated, but cleared if Final Storage Area 2 itself is reallocated.

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

If Intermediate Storage is too small to accommodate the programs or instructions entered, the 'E:04' error code is displayed in the *0, *6 and *B Modes. You can remove this error code either by altering the programs or by entering a larger value for the size of Intermediate Storage. The size of Final Storage can be maximised by limiting Intermediate Storage to the minimum number of locations necessary to accommodate the programs entered. The size of Final Storage and the rate at which data is stored determines how long it will take for Final Storage to fill, at which point new data will write over old.

CAUTION

The number of bytes remaining in program memory is displayed in the fifth window. *Entering 1986* (the total bytes available) *completely resets the CR10.* All memory is erased and the power-up memory check and initialisation is repeated as if the power were switched off and on again.

1.6 Memory Testing and System Status — *B Mode

The *B Mode is used to:

1. Read the signature of the program memory and the software PROM.
2. Display the total size of RAM and PROM.

3. Display the number of E08 occurrences (see Section 3 for a description of error codes).
4. Display the number of overrun occurrences (see Section 1).
5. Display the PROM version number.

Table 1-6 describes what the values seen in the *B Mode represent. The correct signature of the standard CR10 PROM is given in Appendix B.

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 ensures 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 power-up, the signature computed for a PROM is compared with a signature stored in the PROM to determine if a failure has occurred. The algorithm used to calculate the signature is described in Appendix C.

The contents of windows 6 and 7, PROM version and version revision, are helpful in determining what PROM is in the datalogger. When calling Campbell Scientific for datalogger assistance, please have these two numbers available.

Table 1-6 Description of *B Mode Data

Keyboard Entry	Display ID: Data	Description of Data
*B	01: XXXXX	Program memory Signature. The value depends on the program entered and memory allocation. If the Tables have not been previously compiled, they are compiled and run.
A	02: XXXXX	PROM Signature
A	03: XXXXX	Memory Size (32K ROM + 64K RAM)
A	04: XXXXX	No. of E08 occurrences (enter **88** to reset)
A	05: XXXXX	No. of overrun occurrences (enter **88** to reset)
A	06: X.XXXX	PROM version number
A	07: XXXX.	Version revision number

1.7 *C Mode — Security

The *C Mode (see Table 1-7) is used to block access to your program information and certain CR10 functions. There are three levels of security, each with its own 4-digit password. All passwords are set to 0000 on power-up, which disables security (unless the CR10 has a custom PROM with the password built in). 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 stored in write-protected memory and affect the program signature.

Table 1-7 *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 *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

When security is disabled, entering *C will advance you 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 as 0000, or entering 0000 for the password, disables that and subsequent levels of security.

Security can be temporarily disabled by entering a password in the *C Mode or using the telecommunications L command (see Section 5). 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** automatically re-enables security to the level determined by the passwords entered.

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

1.8 *D Mode — Save or Load Program

The *D Mode is used with a Storage Module, a computer/printer or a cassette recorder to save or load your program information (i.e. the information for the *1, *2, *3, *A, *C and *B Modes).

NOTE The *D Mode functions for cassette recorder are available only as a special software option (see Appendix B.)

The PC208 software includes the program GraphTerm, which automatically makes use of the *D Mode to upload and download programs from a computer.

When you enter ***D**, the CR10 displays '`13:00`'. A command is entered by keying the command number and A (see Table 1-8).

Table 1-8 *D Mode Commands

Command	Description
1	Send ASCII Program
2	Load ASCII Program
7N	Save/Load/Clear Program from Storage Module N

Commands 1 and 2 (when entered from the Keyboard/Display) and 7 have an additional 2-digit option parameter (command 7 is entered with the Storage Module address, e.g. 71). The CR10 will display the command number and prompt for the option. If the Keyboard/Display is not being used, the CR10 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.

After entering the option code press A to execute the command. Command 2 is aborted if no data is received within 30 seconds.

CAUTION

When command 2 (Load ASCII Program) is executed all data in Input and Intermediate Storage is erased.

If the CR10 program has not been compiled when the command to save a program (i.e. command 1, 2 or 7) is entered, it will be compiled before the program is saved. After a command is executed, '`13:0000`' is displayed; ***D** must be entered again before another command can be given.

Table 1-9 ASCII and Storage Module Command Options

Command		Option Code Description
1 & 2	1x	Synchronously addressed
	4x	Hardware enabled
		x = Baud Rate Codes 0 - 300 1 - 1200 2 - 9600 3 - 76800
7N:00		(N is Storage Module address)
	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)

Table 1-10 Program Load Error Codes

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 (see Section 3)

1.8.1 Program Transfer With Computer / Printer

This section describes commands 1 and 2 of the *D Mode. GraphTerm (PC208 Software) uses these commands automatically for uploading and downloading programs.

Send ASCII Program Information from CR10

Program listings are sent in ASCII. At the end of the listing, the CR10 sends control-E (5 hex or decimal) twice.

Table 1-11 is an example of the program listing sent in response to command 1. Note that the listing uses numbers for each mode: The numbers for *A, *B and *C modes are 10, 11 and 12, respectively.

Table 1-11 Example Program Listing From *D Mode 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
2:30351
3:48
4:0
5:0
^E ^E
```

Load Program from ASCII File to CR10

Command 2 sets up the CR10 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 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, or semicolon, or 7D Hex. The 'M' serves the same function as '*' does from the keyboard. The order in which the Modes are sent 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 Scan Rate (execution interval).
3. The colons (:) are used to mark the start of actual data.
4. A semicolon (;) tells the CR10 to ignore the rest of the line and can be used after an entry so that a comment can be added.

There are four two-character control codes which may be used to verify that the CR10 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 CR10 buffers the data in memory; the data is not loaded into the editor or compiled until the CR10 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 CR10 to send the signature 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 CR10 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.

1.8.2 Program Transfer with Storage Module

The Storage Module and Keyboard/Display or computer must both be connected to the CR10. Enter ***D7N**, where N is the Storage Module address. Address 1 will work with any Storage Module address, because the CR10 searches for the Storage Module with the lowest address that is connected. Next enter the command to save, load or clear a program and the program number (see Table 1-9). 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.

The CR10 can be programmed on power-up using a Storage Module. Storage Modules can store up to eight separate programs. If a program is stored as program number 8, and if the Storage Module is connected to the CR10 at power-up, program number 8 is downloaded and compiled.

1.9 Other Functional Modes

*4 Mode:	not included in standard CR10
*7 Mode:	see section 2
*8 Mode:	see section 4
*9 Mode:	see section 4

Section 2. Internal Data Storage

2.1 Final Storage Areas, Output Arrays and Memory Pointers

Final Storage is the portion of memory where the CR10 stores final processed data. It is from Final Storage that data is transferred to your computer or external storage peripheral.

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

The default size of Final Storage is 64K bytes or 29908 memory locations. One RAM chip is dedicated to Final Storage. This chip has 32K bytes. A minimum of 32K bytes (16K memory locations) is *always* retained in Final Storage.

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 you do not specifically allocate memory to Area 2. A minimum of 768 memory locations is *always* retained in Final Storage Area 1.

Two Final Storage Areas can 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* (see Figure 2-1) in which the newest data is written over the oldest data.

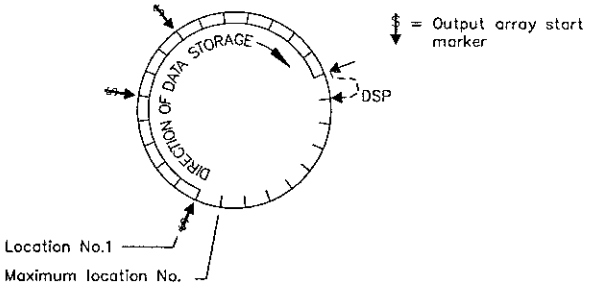


Figure 2-1 Ring Memory Representation of Final Data Storage

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.

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 4-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 2-2 indicates that the 18th instruction in Table 1 set the Output Flag.

2. You can set the output array ID with the second parameter of Instruction 80. The ID can be set to any positive integer up to 511. This option allows you to make the output array ID independent of the programming because 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.

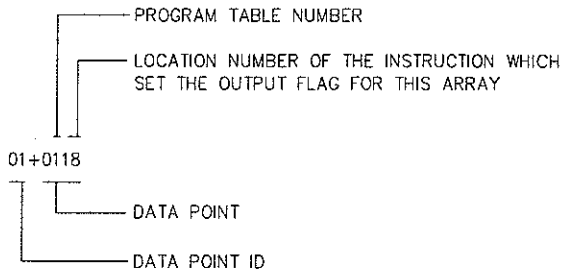


Figure 2-2 Output Array ID

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) 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 cannot actually be seen.

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

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

The **DPTR** is used to recall data to the Keyboard/Display. The positioning of this pointer and data recall are controlled from the keyboard using the *7 Mode.

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 CR10.* If the Storage Module is not connected, the CR10 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 CR10 outputs all of the data between the SPTR and the DSP and updates the SPTR to the DSP location (see Section 4.)

The SPTR can also be positioned via the keyboard for manually initiated data transfer to the Storage Module (*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 (see Section 5).

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

The **TPTR** is used to control data transmission to a cassette tape recorder. When on-line tape transfer is activated (using Instruction 96, option 00), data is transmitted to tape whenever the DSP is a minimum of 512 memory locations ahead of the TPTR. The TPTR can also be positioned via the keyboard for manually initiated data transfer to tape (*8 Mode).

NOTE

All memory pointers are set to the DSP location when the CR10 compiles a program. For this reason, *always retrieve uncollected data before making program changes.* For example, assume the TPTR lags the DSP by less than 512 data points when the datalogger program is altered. On compiling, the TPTR is positioned with the DSP, losing reference to the data that was intended to be transferred to tape. The data is not automatically transferred and appears as a discontinuity in the data file. Until the ring memory wraps around and data overwrite occurs, the data may be recovered using the *8 Mode. This scenario is also true for the SPTR and data intended for a Storage Module.

2.2 Data Output Format and Range Limits

Data is stored internally in Campbell Scientific's Binary Final Storage Format (see Appendix C). Data can 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 four significant digits and a maximum magnitude of +6999. High resolution data is a 4-byte format with five significant digits and a maximum possible output value of +99999 (see Table 2-1).

Table 2-1 Resolution Range Limits of CR10 Data

Resolution	Zero	Minimum Magnitude	Maximum Magnitude
Low	0.000	+0.001	+6999.
High	0.0000	+0.00001	+99999.

The resolution of the low resolution format is reduced to three significant digits when the first (left most) digit is seven 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 0.01cm, the level must be less than 70cm for low resolution output to display the 0.01cm increment. If the water level was expected to range from 50 to 80cm the data could either be output in high resolution or could be offset by 20cm (transforming the range to 30 to 50cm).

2.2.2 Input and Intermediate Storage Data Format

While output data has the limits described above, the computations performed in the CR10 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 one 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 is given in Appendix C.

2.3 Displaying Stored Data on the CR10KD — *7 Mode

NOTE If you are using a computer refer to Section 5 for instructions on entering the Remote Keyboard State. You can then follow the instructions below.

You can view the contents of Final Storage by using the *7 Mode. Enter *7 to enter this Mode.

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

```
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 is 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 A. 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 entering **#A**. The same element in the previous array can be displayed by entering **#B**. If the element is element1 (the 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 summarised in Table 2-2.

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

Table 2-2 *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
#0A	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 CR10 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 analogue or digital control port output. Processing Instructions perform numerical 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 in specified Input Storage locations. Program Control Instructions are used to control 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 is a fixed number of parameters associated with each instruction to give the CR10 the information required to execute the instruction.

The set of instructions available in the CR10 is determined by the PROM (Programmable Read Only Memory) inside the CR10. Appendix B lists the PROM options available.

3.1 Parameter Data Types

There are three 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 CR10 to make the most efficient use of its memory.

Floating Point parameters are used to enter numeric constants for calibrations or arithmetic operations. While it is only possible to enter five digits (magnitude +.00001 to +99999.), the internal format has a much greater range ($1x10^{-19}$ to $9x10^{18}$ – see Section 2). Instruction 30 can be used to enter a number in scientific notation to be loaded into an input location.

3.2 Repetitions

The repetitions parameter in 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 have four differential voltage measurements to make on the same voltage range, wire the inputs to sequential channels and enter the Differential Voltage Measurement instruction once with four repetitions, rather than entering four separate measurement instructions. The instruction will then make four measurements, starting on the specified channel number and continuing through the three succeeding differential channels, with the results being stored in the specified input location and the three succeeding input locations. Averages for all four measurements can be calculated by entering the Average instruction with four repetitions.

When several of the same type of measurements are to be made, but the calibrations of the sensors are different, time can be saved by using one measurement instruction with repetitions. The calibrations can then be applied with a scaling array (Instruction 53). This is quicker than entering the instruction several times in order to use a different multiplier and offset because of the setup 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 entering a number, press '-' (if using a computer) or C 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 pressed two 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, there is no indexing effect on the excitation voltage.

3.4 Indexing Input Locations and Ports

When used within a loop, the parameter for input locations and the commands to set, toggle or pulse a port can be indexed to the loop counter. The loop counter is added to the indexed value to determine the actual input location or port 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) or Set Port command (2-digit integer) parameter, press C or '-' after entering the value but before pressing A to enter 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 in Input/Output Instructions is used to specify the full scale range of the measurement and the integration period for the measurement (see Table 3-1).

Table 3-1 Input Voltage Ranges and Codes

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

[1] *Differential measurement; resolution for single-ended measurement is twice value shown.*
[2] *272µs on ±2500mV range.*
[3] *25µs on ±2500mV range.*

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

Four different integration sequences are possible. The relative immunity of the integration sequences to random noise is:

1. 50Hz rejection = 60Hz rejection
2. 2.72ms integration
3. 250µs integration.

The 50Hz rejection integration rejects noise from 50Hz AC line power.

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

CAUTION

An input voltage greater than +5V on one of the analogue inputs will result in errors and possible overranging on the other analogue inputs. Voltages greater than 16V may permanently damage the CR10.

Voltages in excess of 5.5V applied to a control port can cause the CR10 to malfunction.

3.6 Output Processing

Most Output Processing Instructions require both an Intermediate Data Processing operation and a Final Data Processing operation. For example, when Instruction 71, Average, is initiated, the intermediate processing operation increments a sample count and adds each new Input Storage value to a cumulative total in Intermediate Storage. When the Output Flag is set, the final processing operations divide the total by the sample count, store the resulting average in Final Storage and zero the value in Intermediate Storage.

Final Storage Area 1 is the default destination for data produced by Output Processing Instructions. Instruction 80 can 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. 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 Output Processing Instructions store processed data values when and only when the Output Flag is set. The Output Flag (flag 0) is set at desired intervals or in response to certain conditions by using an appropriate Program Control Instruction (see Section 11).

3.7 Use Of Flags: Output and Program Control

There are 10 flags which may be used in CR10 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 (see 'Output Processing'). Flags 1-8 can be used as needed in programming the CR10.

NOTE

Flags 0 and 9 are automatically set low at the beginning of the program table. Flags 1-8 remain unchanged until acted on by a Program Control Instruction or until manually toggled from the *6 Mode.

Table 3-2 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) are always 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 Storage by Output Processing Instructions when the Output Flag (flag 0) is set high. This group of data values is called an Output Array. The Output Flag is set using Program Control Instructions according to time- or event-dependent intervals specified in your program. The Output Flag is set low at the beginning of each table.

Output is most often desired at fixed intervals; this is done with Instruction 92, If Time. Output is usually desired at the start of the 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 synchronised to 24-hour time; output occurs 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 is less than the specified time interval. Output occurs at midnight and resumes synchronised 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 must be preceded by a Program Control Instruction that sets the Output Flag.

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 feature eliminates having to enter 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 the program table.

As an example, suppose it is desired to obtain a wind speed rose incorporating only wind speeds greater than or equal to 4.5ms^{-1}. The wind speed rose is computed using the Histogram instruction (75), and wind speed is stored in input location 14, in ms^{-1}. Instruction 89 is placed just before Instruction 75 and is used to set flag 9 high if the wind speed is less than 4.5ms^{-1} (see Table 3-3).

Table 3-3 Example of the Use of Flag 9

Inst. Loc.	Param. No.	Entry	Description
X	P	89	If wind speed < 4.5m/s
	1	14	Wind speed location
	2	4	Comparison: <
	3	4.5	Minimum wind speed for histogram
	4	19	Set Flag 9 high
X+1	P	75	Histogram
X+2	P	86	Do
	1	29	Set Flag 9 Low

NOTE Flag 9 is automatically reset in the same way as flag 0. If flag 9 is already set high and the test condition of a subsequent Program Control Instruction acting on it fails, it is set low. This feature eliminates having to enter another instruction to specifically reset flag 9 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 for general programming needs. These user flags can be manually toggled from the keyboard in the *6 Mode (see Section 1). By inserting the flag test (Instruction 91) at appropriate points in the program, you can use the *6 Mode to direct program execution manually.

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 3-4 lists these codes.

Table 3-4 Command Codes

0	-	Go to end of program table
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
41-48	-	Set port 1-8 high[2]
51-58	-	Set port 1-8 low[2]
61-68	-	Toggle port 1-8[2]
71-78	-	Pulse port 1-8[2]

[1] 97 and 98 are special subroutines which can be called by control ports 7 and 8 going high; see Instruction 85 for details.

[2] Ports can be indexed to the loop counter (see Section 3.4).

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 (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 which are to be executed regardless of the outcome of the comparison (see Figure 3-1).

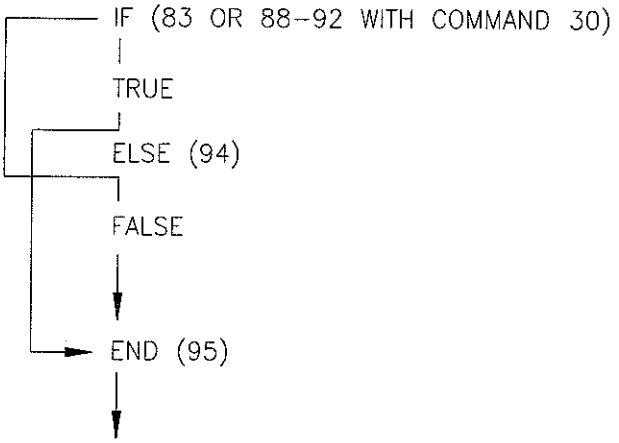


Figure 3-1 If Then/Else Execution Sequence

'If Then/Else' comparisons may be nested to form logical AND or OR branching. Figure 3-2 illustrates an AND construction. If conditions A and B are true, the instructions included between IF B and the first End instruction are executed. If either of the conditions is false, execution jumps to the corresponding End instruction, skipping the instructions between.

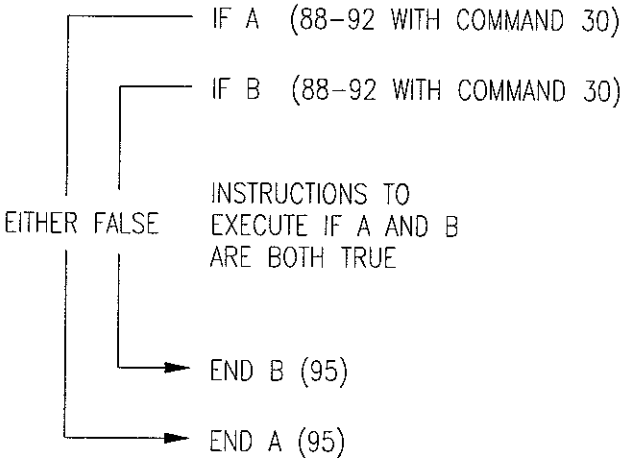


Figure 3-2 Logical AND Construction

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

```
IF A (88-92 with command 30)
Call subroutine X (86, command=X)
ELSE (94)
IF B (88-92 with command 30)
Call subroutine X (86, command=X)
END B (95)
END A (95)
```

Figure 3-3 Logical OR Construction

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

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. Instruction 83 is then used several times 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 description of Instruction 93 in Section 12).

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 nine deep. Loop (Instruction 87) and Begin Case (Instruction 93) both count as one level. Instructions 83, 86, 88, 89, 91 and 92 each count as one level when used with command 30 ('Then Do'). Use of Else (Instruction 94) also counts as one nesting level each time it is used. For example, the AND construction above is nested two deep while the OR construction is nested three deep.

Subroutine calls do not count as nesting with the above instructions although they have their own nesting limit (seven; see description of Instruction 85). Branching and loop nesting starts at zero within each subroutine.

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

3.9 Instruction Memory and Execution Time

There are 1986 bytes of program memory available for the programs entered in the *1, *2 and *3 Program Tables. Each instruction also makes use of varying numbers of Input, Intermediate and Final Storage locations. The following tables list the memory used by each instruction and the approximate time required to execute the instruction.

Table 3-5 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	4.5 + 5.1R	2.1 + 2.7R	2.1 + 2.6R	1.9 + 2.5R	12.9 + 13.4R	10.5 + 11.1R	14.8 + 15.1R	12.1 + 12.7R
2	VOLT (DIFF)	R	15	0.5 + 9.2R	0.5 + 4.4R	0.5 + 4.2R	0.5 + 3.8R	0.5 + 25.9R	0.5 + 21.3R	0.5 + 29.4R	0.5 + 24.7R
3	PULSE	R	15	0.9 + 1.0R							
4	EX-DEL-SE	R	20	4.7 + 5.3R	2.3 + 2.8R	2.3 + 2.8R	2.1 + 2.6R	13.1 + 13.6R	10.7 + 11.2R	15.0 + 15.2R	12.3 + 12.8R
5	AC HALF BR	R	18	6.9 + 9.6R	4.4 + 4.8R	4.4 + 4.7R	4.3 + 4.4R	15.0 + 26.3R	11.3 + 21.7R	16.8 + 29.6R	14.5 + 24.8R
6	FULL BR	R	18	3.3 + 17.7R	3.3 + 8.0R	3.3 + 7.8R	3.2 + 7.0R	3.2 + 51.1R	3.3 + 41.7R	3.3 + 57.8R	3.3 + 48.5R
7	3W HALF BR	R	18	4.5 + 21.6R	2.8 + 11.8R	3.6 + 11.6R	3.4 + 10.9R	14.9 + 54.6R	10.4 + 45.3R	15.9 + 61.4R	14.3 + 51.6R
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	10.6 + 11.2R	15.0 + 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	1.3 + 37.3R	1.3 + 18.0R	1.3 + 17.6R	1.3 + 15.9R	1.3 + 104.3R	1.3 + 85.0R	1.3 + 117.9R	1.3 + 98.3R
	V1 on range 5, V2 on:			1.3 + 27.8R	1.3 + 18.0R	1.3 + 17.7R	1.3 + 16.9R	1.3 + 61.2R	1.3 + 51.8R	1.3 + 67.8R	1.3 + 58.2R
	V1 on range 15, V2 on:			1.3 + 26.7R	1.3 + 16.9R	1.3 + 16.7R	1.3 + 15.9R	1.3 + 59.9R	1.3 + 50.6R	1.3 + 66.6R	1.3 + 57.4R
10	BATT. VOLT	1	4	4.8							
11	TEMP-(107)	R	15	4.5 + 7.2R							
12	RH (207)	R	17	2.1 + 9.1R							
13	TEMP-TC SE	R	18	2.4 + 10.2R	6.7 + 6.4R	4.9 + 6.6R	2.5 + 6.5R	15.7 + 17.3R	12.1 + 15.2R	17.0 + 19.2R	18.9 + 15.6R
14	TEMP-TC DIF	R	18	3.7 + 13.5R	3.7 + 8.2R	3.7 + 8.6R	3.7 + 7.7R	3.7 + 30.4R	3.7 + 25.4R	3.7 + 33.5R	3.7 + 28.1R
16	TEMP-RTD	R	15	0.4 + 3.0R							
17	TEMP-INTERNL	1	4	6.2							
18	TIME	1	7	1.7							
19	SIGNATURE	1	4	750							
20	PORT SET	0	6	9.5							
22	EXCIT-DEL	0	9	0.5							
24	CALIBRATION	19	4	2664, 21.2 when only saving results of automatic calibration.							
25	READ PORTS	1	6	0.4							
26	TIMER	1 or 0	4	0.3 to reset, 0.8 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							
101	SDM-INT8	1-8	27	2.3 + 1.65 * No.Values+averaging interval if used							
102	SDM-SW8A	R	16	4 + F * R; F=function time = 1 for state, 2 for counts or cycle, 0 for signature							
103	SDM-AO4	0	6	1.7 + 0.8R							
104	SDM-CD16	0	6	2R							

When attempting to make a fast series of measurements and calculations, 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 describes the calibration process.

Table 3-6 Processing Instruction Memory and Execution Times (R = No. of Repetitions)

Instruction	Memory			Execution Time (ms)
	Input Loc.	Inter. Loc.	Prog. Bytes	
30 Z=F	1	0	9	0.2 + 0.6 * exponent
31 Z=X	1	0	6	0.5
32 Z=Z+1	1	0	4	0.6
33 Z=X+Y	1	0	8	1.1
34 Z=X+F	1	0	10	0.9
35 Z=X-Y	1	0	8	1.1
36 Z=X*Y	1	0	8	1.2
37 Z=X*F	1	0	10	0.9
38 Z=X/Y	1	0	8	2.7
39 Z=SQRT(X)	1	0	6	12.0
40 Z=LN(X)	1	0	6	7.4
41 Z=EXP(X)	1	0	6	5.9
42 Z=1/X	1	0	6	2.6
43 Z=ABS(X)	1	0	6	0.7
44 Z=FRAC(X)	1	0	6	0.3
45 Z=INT(X)	1	0	6	1.0
46 Z=X MOD F	1	0	10	3.2
47 Z=XY	1	0	8	13.3
48 Z=SIN(X)	1	0	6	6.5
49 SPA. MAX	1 or 2	0	8	1.5 + 0.9 (swath-1)
50 SPA. MIN	1 or 2	0	8	1.7 + 0.9 (swath-1)
51 SPA. AVG	1	0	8	3.3 + 0.6 (swath-1)
53 A*X+B	4	0	36	4.1
54 BLOCK MOVE	R	0	10	0.2 + 0.2R
55 POLYNOMIAL	R	0	31	1.2 + (2.0 + 0.4 * order)R
56 SAT. VP	1	0	6	4.2
57 WDT-VP	1	0	10	8.1
58 LP FILTER	R	R	13	0.5 + 2.2R
59 X/(1-X)	1	0	9	0.4 + 3.0R
61 INDIR. MOVE	1	0	6	0.4 neither indexed 0.5 1 location indexed 0.7 both locations indexed
63 PARA.EXTN.	0	0	10	0.1
66 ARC TAN	1	0	8	6.7

Table 3-7 Output Instruction Memory and Execution Times (R = No. of Repetitions)

Instruction	Memory			Execution Time (ms)	
	Inter. Loc.	Final Values[1]	Prog. Bytes	Flag 0 Low	Flag 0 High
69 WIND VECTOR	2+9R	(2, 3 or 4)R	12		
		Options 00, 01, 02		3.5 + 17.5R	3.5 + 75R
		Options 10, 11, 12		3.5 + 16R	3.5 + 30R
70 SAMPLE	0	R	6	0.1	0.4+ 0.6R
71 AVERAGE	1+R	R	7	0.9+ 0.5R	2.1+ 3.0R
72 TOTALIZE	R	R	7	0.6+ 0.5R	1.1+ 1.0R
73 MAXIMIZE	(1 or 2)R	(1,2 or 3)R	8	0.9+ 1.7R	1.3+ 2.8R
74 MINIMIZE	(1 or 2)R	(1,2 or 3)R	6	0.9+ 1.7R	1.3+ 2.8R
75 HISTOGRAM	1+bins*R	bins*R	24	0.4+ 3.1R	0.9+ (3.3+2.8*bins)R
77 REAL TIME	0	1 to 4	4	0.1	1.0
78 RESOLUTION	0	0	3	0.4	0.4
79 SMPL ON MM	R	R	7	0.3	1.1
80 STORE AREA[1]	0	0	7	0.2	0.2
82 STD. DEV.	1+3R	R	7	1.0+ 1.4R	1.8+ 2.2R

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

Table 3-8 Program Control Instruction Memory and Execution Times

Instruction	Memory		Execution Time (ms)
	Inter. Loc.	Prog. Bytes	
83 IF CASE <F	0	9	0.5
85 LABEL SUBR.	0	3	0
86 DO	0	5	0.1
87 LOOP	1	7	0.2
88 IF X<=>Y	0	10	0.6
89 IF X<=>F	0	12	0.4
90 LOOP INDEX	0	3	0.5
91 IF FLAG/PORT	0	6	0.3
92 IF TIME	1	11	0.3
93 BEGIN CASE	1	8	0.2
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
98 SEND CHAR.	0	3	0.7+0.05 * No. char.

3.10 Error Codes

There are four types of errors flagged by the CR10: Compile, Run Time, Editor and *D Mode.

Compile errors are errors in programming which are detected when 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 two-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. 105 indicates that the fifth instruction in Table 1 caused the error). Error 22, missing End, indicates the location of the instruction which the compiler cannot match with the 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 error 8 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 (see below).

If there is a run time error in a table with a short execution interval, the error may be written to the display so frequently that it seems the CR10 is not responding to the keyboard. Once the program is stopped, normal function will return. To stop the program an entry must be changed which requires recompiling (see Section 1). For example, enter 0 for the execution interval of Table 1 and then recompile. (This should be done quickly since error messages are queued and it may take some time to communicate with the CR10 if many such messages have accumulated.)

Error 8 is the result of a hardware and software 'watchdog' that checks the processor state, software timers and program-related counters. The watchdog attempts to reset the processor and program execution if it finds that the processor has crashed or is neglecting standard system updates, or if the counters are out of allowable limits. Error code 08 is displayed 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 CR10 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 (see Section 1) or with the Telecommunications A command (see Section 5).

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 3-9 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	CR10 reset by watchdog timer
09	Run Time	Insufficient Input Storage
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 (P95)
23	Compile	Non-existent 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
40	Editor	Instruction does not exist
41	Editor	Incorrect execution interval
60	Compile	Inadequate Input Storage for FFT
61	Compile	Burst Measurement Scan Rate too short
96	*D MODE	Addressed device not connected
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 you can carry from the test site to the lab and to supplement the internal storage capacity of the CR10, allowing longer periods between visits to the site. The standard data storage peripheral for the CR10 is the Storage Module. Output to a printer or cassette tape is also possible.

Data output to a peripheral device can take place ***on-line*** *(automatically, as part of the CR10's routine operation) or it can be* ***manually initiated****. On-line data transfer is accomplished with Instruction 96. Manual initiation is done in the *8 Mode.*

The CR10 can output data to multiple peripherals. The CR10 activates the peripheral it sends data to in one of two ways (see also Section 6):

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 CR10. This is referred to as* ***'addressed'****.*

Modems and cassette recorders are pin-enabled. Only one cassette recorder and only one modem/terminal device may be connected to the CR10 at any one time.

The SM192 and SM716 Storage Modules and the CSM1 Card Storage Module are addressed. The CR10 can detect when the addressed device is present. The CR10 will not send data meant for the Storage Module if the Storage Module is not present.

*The *9 Mode allows you to communicate directly with an SM192/716 Storage Module and to perform several functions, including review of data, battery test and review of Storage Module status.*

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 — see Section 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 tables 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 (see Section 10).
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 to 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 lists the output device codes.

NOTE 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).

Table 4-1 Output Device Codes for Instruction 96 and *8 Mode

Code	Device
00	Tape. Data transferred in blocks of 512 Final Storage locations
09	Tape. All data since last output. [Inst. 96 only]
	Addressed Printer
1x	Printable ASCII
2x	Comma delineated ASCII
3x	Binary
	Pin-Enabled Printer
4x	Printable ASCII
5x	Comma delineated ASCII
6x	Binary

x = Baud Rate Codes

0	300
1	1200
2	9600
3	76800

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

If the CR10 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 information in the queue (which determines a unique entry) consists of the device, the baud rate (if applicable) and the Final Storage Area.

When an entry reaches the top of the queue, the CR10 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.

4.1.1 Output to Storage Module

The Storage Module address is important only when using more than one Storage Module[1]. 1 is a universal address which will find the Storage Module with the lowest number address that is connected. If a Storage Module is not connected, the CR10 will not advance the SPTR and the Storage Module drops out of the queue until the next time Instruction 96 is executed.

4.1.2 Output to Printer

Printer output can be either pin-enabled or addressed. However, there is no pin specifically dedicated to enabling a printer. 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 line. This allows some compatibility with the 21X and CR7 dataloggers, which have a Print Enable line.

CAUTION The pin-enabled print option results in garbage being sent to the print peripheral if an addressed device is also connected to the CR10 (i.e. CR10KD, SM192 or SM716, etc.). The SDC99 Synchronous Device Interface can convert a print device to an addressed peripheral (see Section 6).

4.1.3 Output to Cassette Tape

The most efficient use of cassette tape and power is made with option 00, which transfers data in blocks of 512 Final Storage locations. This is because information is always stored on the tape in a fixed block size equivalent to 512 locations. If the CR10 is forced to write less than 512 data values to tape (by specifying option 09), then the block is filled up with null characters. This wastes space on the tape. It also means that the cassette recorder is powered up more frequently. When option 00 is specified, the cassette recorder is not activated until there are 512 values to transfer.

Option 09, transfer any new data, should therefore be used with care. It may be appropriate if it is desired to run the tape only at particular times or under certain conditions (the CR10 program would then be written so that Instruction 96 was only executed when these conditions were met). When instruction 96 is finally executed, all data between the TPTR and the DSP, including a final block less than 512 locations, is written to tape.

Section 4.6 contains specifics on the cassette recorder. Note that tape operation is for above 0°C only.

4.2 Manually-Initiated Data Output — *8 Mode

Data transfer to a peripheral device can be manually initiated in the *8 Mode. To do this, you need to have access to the CR10 through a computer or the CR10KD Keyboard/Display. The *8 Mode allows you to retrieve a specific block of data, on demand, regardless of whether or not the CR10 is programmed for on-line data output.

[1]Up to eight SM192 or SM716 Storage Modules can be connected to a CR10. The address of a CSM1 Card Storage Module is always 1, and only one CSM1 can be connected.

If external storage peripherals (such as a Storage Module) are not left on-line, the maximum time between site visits and data retrieval must be calculated to ensure that data placed in Final Storage is not lost due to being overwritten.
In order to make this calculation, you need 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 included per Output Array
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 one overhead data point (two bytes) per array for the Output Array ID.)

For example, assume that 29900 locations are assigned to Final Storage and that one Output Array, containing the Array ID (one memory location), nine low resolution data points (nine memory locations) and five high resolution data points (10 memory locations), is stored each hour. In addition, an Output Array with the Array ID and five high resolution data points (11 memory locations) is stored daily. This makes a total of 491 memory locations per day ((20 x 24) + 11). 29900 divided by 491 = 60.9 days. Therefore, the CR10 would have to be visited every 60 days to retrieve data, because write-over would begin on the 61st day.

The output device codes used with the *8 Mode are the same as those used with Instruction 96 (see Table 4-1), with the exception of 'all data to tape' (i.e. option code 09: in the *8 Mode all data between the start and stop locations is always written) and the options to transfer data from one Final Storage area to the other (i.e. option codes 80 and 81). Table 4-2 lists the keystrokes required to initiate a *8 Mode data dump.

Table 4-2 *8 Mode Entries

Key	Display ID:DATA	Description
*8	08:00	Enter 1 or 2 for Storage Area. (This window is skipped if no memory has been allocated to Final Storage Area 2.)
A	01:XX	Enter output device option (see Table 4-1)
A	02:XXXXX	Start of dump location. Initially the TPTR, 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 entered if desired.
A	04:00	Ready to dump. To initiate dump, enter any number, then press A. While dumping, '04' is displayed in the ID field and the location number in the Data field. The location number stops incrementing when the dump is complete. (Any key aborts transmission after completion of the current data block.)

4.3 Use of Storage Modules (SM192 / 716 and CSM1)

The Storage Module stores data in battery-backed RAM. Backup is provided by an internal lithium battery. Operating power is supplied by the CR10 over pin 1 of the 9-pin connector. When power is applied to the Storage Module, a File Mark is placed in the data (if a File Mark is not the last data point already in storage).

The File Mark separates blocks of data. For example, if you retrieve data from one CR10, disconnect the Storage Module and connect it to a second CR10, a File

Mark is automatically placed in the data. This mark follows the data from the first CR10 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 capacity of the CSM1 depends on the memory card used. The memory structure is always 'fill and stop'.

4.3.1 Storage Module Addressing

NOTE The address of a CSM1 is always 1, and only one CSM1 can be connected to the CR10.

Assigning different addresses to Storage Modules allows:

1. Multiple (up to eight) Storage Modules to be connected to the CR10 during on-line output (Instruction 96),
2. Different data to be output to different Modules, and
3. Transfer of data from a Module that is left with the CR10 to a Module that is hand carried to the site for data transfer (*9 Mode).

Storage Modules are assigned addresses (in the range 1-8) using either the *9 Mode or telecommunications commands (see SM192/SM716 Manual). The default address when the Storage Module is reset is 1, and unless you are using one of the features which needs 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, or with the *8 or *9 Modes.* When address 1 is entered in the *9 Mode (default) or in the device code for Instruction 96 or the *8 Mode, the CR10 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 actually assigned to the Module.

Address 1 would be used with Instruction 96 if several Storage Modules with different addresses were connected to the CR10 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, and so on.

4.3.2 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 CR10 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 CR10 detects whether or not the Storage Module is connected. Each time Instruction 96 is executed and there is data to output, the CR10 checks for the presence of a Storage Module. If one is not present, the CR10 does not attempt to output data to it. Instead, the CR10 saves the data and continues its other operations without advancing the Storage Module Pointer (see Section 2 for information on pointers).

When you connect the Storage Module to the CR10, two things happen:

1. Immediately on 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 stored).
2. During the next execution of Instruction 96, the CR10 recognises that the Storage Module (SM) is present and outputs all data between the SPTR and the DSP location.

To be certain that the SM has been connected to the CR10 during an execution of Instruction 96, you can:

- leave the SM connected for a time 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 CR10 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 CR10.

4.3.3 *8 Mode Data Dump to Storage Module

In addition to the on-line data output procedures described above, output from CR10 Final Storage to a Storage Module can be manually initiated in the *8 Mode. The procedure for setting up and transferring data is as follows:

1. Connect the CR10KD Keyboard/Display (or computer) and the Storage Module in parallel to the CR10 using the SC12 cable. If you are using a computer, you will need an SC32Aor SC929. See Section 5 for interfacing details.
2. Enter the appropriate commands as listed in Table 4-2.

4.4 *9 Mode — Commands to SM192/716 Storage Module

The *9 Mode is used to issue commands to an SM192/716 Storage Module (from the CR10) using the CR10KD or a terminal/computer. These commands are like Functional Modes for the Storage Module and in some cases are directly analogous to the CR10 Functional Modes. For example, command 7 enters a mode used to review stored data, and command 8 is used to transfer data between two Storage Modules connected to the CR10. The operations with the Storage Module are not directly analogous, as shown in Table 4-3 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 entered, the CR10 responds:

```
09:01
```

'1' is the default address for the Storage Module. If you have more than one Storage Module connected, enter the address of the Storage Module you want to communicate with. Address 1 will always work if only one Module is connected. Press A and the CR10 responds:

```
9N:00
```

Where N is the address which was entered. You can now enter any of the commands in Table 4-3 (key in the command number and enter it with A). Most commands have at least one response; advance through these and return to the *9 command state by pressing A.

Table 4-3 *9 Mode Commands for SM192/716 Storage Module

Command	Display	Description
1	01: 0000	RESET: enter 248 to erase all data and programs. While erasing, the SM checks memory.
	01: XX	The number of good chips is then displayed (six for SM192, 22 for SM716). The time required to reset an SM192 is about 60 seconds, and for an SM716 about 4 minutes.
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 and 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: A0 CD	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 the following 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 (Codes 1-5 will loop within file boundaries, codes 0, 7 and 9 allow display to cross boundaries.)

Table 4-3 (cont.)

	07:XXXXXX	SM location at end of area selected. Press A to advance to first data. If another location is entered, SM jumps to first 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 command 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)

4.5 Printer Output Formats

Printer output can be sent in Final Storage Format (see Appendix C), printable ASCII or comma delineated ASCII. These ASCII formats can also be used when data from Storage Modules, cassette tape or telecommunications is stored on disk with Campbell Scientific's PC208 software.

4.5.1 Printable ASCII Format

In the printable ASCII format each data point is preceded by a two-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 ten bytes per data point to store on disk.

Figure 4-1 shows both high and low resolution data points in an Output Array with 12 data points. The example data contains Day, Hour-Minute and Seconds in the second, third and fourth data points respectively.

NOTE You must specifically program the CR10 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.

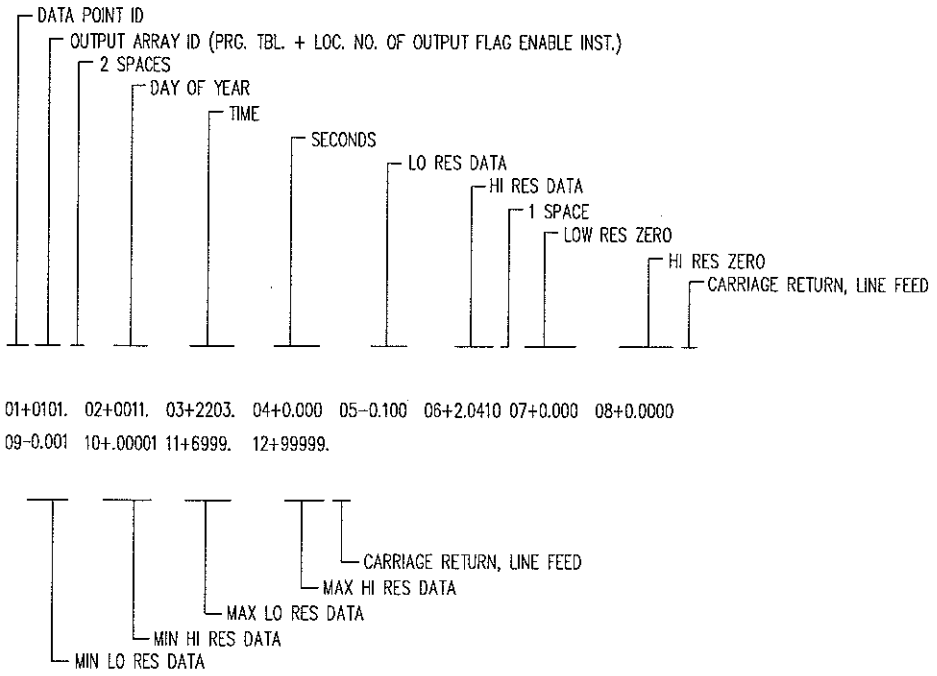


Figure 4-1 Example of CR10 Printable ASCII Output Format

Each full line of data contains eight 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 eight data points are terminated similarly after the last data point.

4.5.2 Comma Delineated ASCII

Comma delineated 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 and line feed. Comma delineated ASCII requires, on average, approximately six bytes per data point. Example:

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

4.6 Cassette Tape Option

The RC35 Cassette Tape Recorder or equivalent can be left attached to the CR10 for continuous on-line data recording or it can be periodically taken to the CR10 site for the manually-initiated retrieval of data accumulated in Final Storage. The *8 Mode is used to initiate tape transfer manually.

4.6.1 Cassette Recorder

The RC35 Cassette Recorder offered by Campbell Scientific is an inexpensive recorder for use with the CR10 (also compatible with the 21X and CR7 dataloggers). The record/playback function of each RC35 is tested along with a head alignment procedure before shipment. CR10/RC35 connections are made with the SC92A Cassette Write Only Interface or the SC93A Cassette Read/ Write Interface. The CR10 controls the on/off state of the RC35 by switching power through the DC power line of the SC92A/SC93A.

Table 4-4 Cassette Recorder Specifications	
Power:	6V DC (provided by CR10 through SC92A or SC93A); 4 AA size batteries; 240V AC / 6V DC adapter
Current Drain while Recording:	200mA typ., 300mA max.
Tape Length:	C-60 recommended
Tape Quality	Normal bias, high quality (e.g. TDK, Maxell)
External Inputs:	Mic., DC In, Monitor and Remote
Operating Temperature:	0° to +40°C

Power Supply

The CR10's power supply powers the recorder during periods of data transfer, but is *not* available to play, advance or rewind tapes. In order to perform these functions during setup and check-out operations, the recorder needs four alkaline AA batteries or the 240V AC adapter.

Operating Temperature Limitations

The cassette recorder is recommended for use in an environmental operating temperature range of 0°C to +40°C. Temperatures below 0°C may cause tape speed variation in excess of that which can be tolerated during playback. If the RC35 is outside the 0°C to 40°C range, data transferred may be unreadable.

Volume Control

When recording data, the RC35's volume setting does not matter. The recorder is equipped with an automatic gain control which controls the recorded signal level. For playback, a mid-range volume setting is normally required.

Cassette Tapes

Normal bias, high-quality cassette tapes are recommended for use with the recorder. The more expensive high bias chromium oxide tapes will *not* perform satisfactorily. Although the use of C-90 tapes is generally successful, Campbell Scientific recommends the use of C-60 (30 minutes per side) cassettes. TDK, Maxell and equivalent quality cassette tapes perform well and are readily available. Bargain-priced tapes have often performed poorly and are not recommended.

New tapes are often tightly wound, creating enough drag or pressure to cause the tape recorder to drop out of the record mode. This potential loss of data may be overcome by fast-forward/rewinding the entire tape before placing it in service.

4.6.2 Cassette Connector Interface Cables

A cassette interface cable is required to connect the cassette recorder to the CR10. Two models are available. The SC92A is a *write only* interface. The SC93A is a *read/write* interface that allows the CR10 to load datalogger programs from tape in addition to writing data and programs. The SC93A is required only if special software exists in the datalogger PROM for transferring programs via tape (refer to Appendix B).

CAUTION The SC92/SC93 interfaces previously supplied with the 21X and CR7 dataloggers are not compatible with the CR10. The SC235 CR21 Cassette Connector Interface supplied with the CR21 datalogger is not compatible with the CR10. If the SC92, SC93, or SC235 interfaces are used with the CR10, the data on tape cannot be recovered.

The SC92A and SC93A have a combination backshell circuit card and subminiature 9-pin D-type connector which attaches to the socket connector on the CR10WP. The other end of the SC92A has two plugs which are plugged into the POWER and MIC jacks on the recorder. The SC93A has three plugs which are plugged into the POWER, MIC and EAR (or MONITOR) jacks on the recorder. Both cables transform 12V from the CR10 to 6V for powering the recorder during periods of data transfer. Additional circuitry shapes the data signal waveform.

4.6.3 Tape Format

Data is transferred to cassette tape in the high speed/high density Format 2. Data tapes generated by the CR10 are read by the PC201 Tape Read Card for the IBM PC or by the C20 Cassette Interface. The C20 decodes the tape and transmits the data in ASCII to any external device equipped with a standard RS232 interface.

Table 4-5 Format 2 Specifications

Data:	Binary
Low Resolution:	2 bytes/data point
High Resolution:	4 bytes/data point
C-60 Capacity (Lo Res.):	180,000 data points (one side only)
Data Transfer Rate (Lo Res.):	100 data points/sec.
Block Size:	512 Final Storage locations

4.6.4 Connecting the Cassette Recorder to the CR10

The procedure for setting up the CR10 and cassette recorder for transfer to tape is as follows:

1. Load a cassette in the recorder and advance the tape forward until the tape leader is past the recording head. (Internal batteries or AC power required.)
2. Connect the SC92A or SC93A to the 9-pin D-type connector in the upper right-hand corner of the CR10WP. (This connection should be made via the SC12 ribbon cable if using the *8 Mode with the CR10KD or a modem/ terminal.)
3. Connect the plugs on the free end of the SC92A or SC93A into the DC-IN and MIC (and EAR if using the SC93A) jacks on the recorder.
4. Simultaneously press the RECORD and PLAY buttons on the recorder to set it for recording. With the DC-IN Jack plugged in, the tape will not move until the dump occurs.
5. To test the connections, initiate transfer manually by keying in the *8 commands as listed in Table 4-2. The tape should advance as data is trans-

ferred. If the Start of dump location is equal to the End of dump location, the CR10 writes a 'dummy' block of data to tape.

If you are leaving the recorder with the CR10 (on-line output to tape enabled with Instruction 96) it is a good idea to write a dummy block of data to tape (step 5 above) to ensure that the recorder is correctly connected. Leave the CR10 in the *0 Mode.

When on-line, the CR10 dumps data to tape in blocks of 512 locations (unless the option to dump any new data is selected in Instruction 96). When picking up a data tape from a field site, dump the residual data (data which has accumulated since the last full block) before removing the tape. Dump the residual data by entering the *8 Mode, advancing through windows 2 and 3 and initiating a dump. (The start and stop locations should be less than 512 locations apart.) After removing the old tape, insert a new tape and go through the set up steps above.

Section 5. Telecommunications

Telecommunications is used to retrieve data from Final Storage directly to a computer and to program the CR10. Whenever you use a computer or terminal instead of the CR10KD to communicate with the CR10 you are using telecommunications.

Telecommunications can take place over a variety of links including:

- *telephone*
- *radio frequency*
- *short-haul modem and twisted pair wires*
- *SC32A and ribbon cable ('direct' connection)*
- *multi-drop interface and coax cable (network)*

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 ASCII format or binary format. The binary format is five 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.

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

The PC208 Datalogger Support Software automates data retrieval and helps you to program the CR10. PC208 meets the most common needs in datalogger support and telecommunications. Therefore, this section does not give enough detail to write telecommunications software. Appendix C contains some details of binary data transfer and Campbell Scientific's binary data format.

The emphasis of this section is on the commands to use when manually (i.e. keyed in by hand) interrogating or programming the CR10 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 you 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 an SM192/716 Storage Module*
- *remote keyboard programming*

The Remote Keyboard State allows you to use the same commands as the CR10KD if you have a computer or terminal.

5.1 Telecommunications Commands

When a modem/terminal calls the CR10, the CR10 should answer almost immediately. Several carriage returns (CR) must be sent to the CR10 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 CR10 sends back the prompt '*', signalling 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 CR10 restarts the count; 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. the **F** and **K** commands) return a signature (see Appendix C).

The CR10 sends ASCII data with eight bits, no parity, one start bit and one stop bit.

After the CR10 answers a call or completes a command, it waits about 40 seconds (127 seconds in the Remote Keyboard State) for a valid character to arrive. It drops out of communications if it does not receive a valid character in this time interval. Some modems are quite noisy when not on line, and it is possible for valid characters to appear in the noise pattern. To ensure that this situation does not keep the CR10 in telecommunications, the CR10 counts all the invalid characters it receives from the time it answers a ring, and terminates communication after receiving 150 invalid characters.

The CR10 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 CR10 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 CR10 via the SC32A or other interface (see Section 6). Commands used to interrogate the CR10 in the telecommunications mode are described in the following table.

[F.S. Area]**A**
Select area/Status

If 1 or 2 does not precede the A to select the Final Storage Area, the CR10 defaults to Area 1. All subsequent commands other than A will address the area selected. Datalogger returns Reference (the DSP location); the number of filled Final Storage locations; version of datalogger software; Errors #1 and #2 where #1 is the number of E08's and #2 is the number of overrun errors (both are cleared by entering 8888A; #2 is also cleared at time of program compilation); size of total memory in CR10; Final Storage Area; Location of MPTR; and Checksum. All in the following format:

R+xxxxx F+xxxxx Vxx Exx xx Mxx Ax L+xxxxx Cxxxx

If data is stored while in telecommunications, the A command must be re-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. CR10 sends the Area, MPTR Location, and Checksum: Ax L+xxxxx Cxxxx

[YR:DAY:HR:MM:SS]**C**
Reset/send time

If time is entered the time is reset. If only colons are in the time string, HR:MM:SS is assumed; three colons means DAY:HR:MM:SS. If only the C is entered, time is unaltered. CR10 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 the next start of array. CR10 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 in Telcom (PC208). See Appendix C.

[F.S. loc. no.]**G**
Move MPTR

MPTR is moved to specified Final Storage location. The location number must be entered. CR10 sends Area, Location, and Checksum: Ax L+xxxxx Cxxxx

7H or 2718**H**
Remote Keyboard State

CR10 sends the prompt '>' and is ready to execute standard keyboard commands.

[loc. no.]**I**
Display/change Input Storage

CR10 sends the value stored at the location. A new value and CR may then be sent. CR10 sends checksum. If no new value is sent (CR only) the location value remains 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 CR10 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 C.

[Password]**L**
Security

Unlocks security (if enabled) to the level determined by the password entered (See *C Mode, Section 1). CR10 sends security level (0-3) and checksum: Sxx Cxxxx

[X]M
Connect to Storage Module

Connect to Storage Module with address 'X' and enter the Storage Module's Telecommunications Mode (see Storage Module manual). The SM192/716 Storage Module can also be accessed with the *9 Mode commands while in the Remote Keyboard State (see Section 4 and the Storage Module manual).

1N
Phone to RF

Connect phone modem to RF modem at phone to RF base station.

5.2 Remote Programming of the CR10

Remote programming of the CR10 can be accomplished with Campbell Scientific software or directly through the Remote Keyboard State.

The PC208 Datalogger Support Software runs on IBM or compatible PCs. Datalogger programs are developed on the computer using the program editor (Edlog) and downloaded to the datalogger with the terminal emulator program (GraphTerm in PC208, Term in PC200).

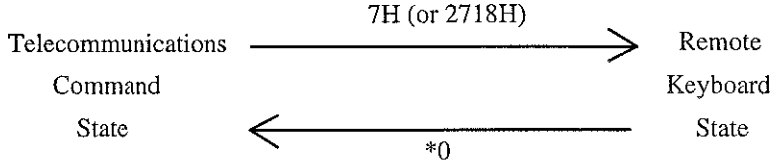
The CR10 is placed in the Remote Keyboard State by sending either **'7H'** or **'2718H'** and a carriage return (CR). The CR10 responds by sending a CR, line feed (LF), and the prompt '>'. The CR10 is then ready to receive the standard keyboard commands; it recognises all the standard CR10 keyboard characters plus several additional characters such as the decimal point and the minus sign (see Section OV3). Entering ***0** returns the CR10 to the telecommunications command state.

NOTE

Entering ***0** compiles and runs the CR10 program if program changes have been made. If the CR10KD is connected it displays 'LOG' when *0 is executed via telecommunications, i.e. it does not indicate the active tables (entering **'*0'** on the Keyboard/ Display itself will show the tables).

The **'7H'** command is generally used with a terminal for direct entry since 7H makes use of a destructive backspace and does not send ctrl-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, ctrl-Q is sent after each user entry) and its use will be familiar to those already working with a 21X or CR7.

It is important to remember that the Remote Keyboard State is a mode within telecommunications. Entering *0 exits the Remote Keyboard State and returns the datalogger to the Telecommunications Command State, awaiting another command. In this way you can switch between the Telecommunications Command State and the Remote Keyboard State.



Section 6. 9-Pin Serial Input / Output

All external communication peripherals connect to the CR10 through the 9-pin subminiature D-type socket connector located on the front of the Wiring Panel (see Figure 6-1). Table 6-1 shows the I/O pin configuration and gives a brief description of the function of each pin.

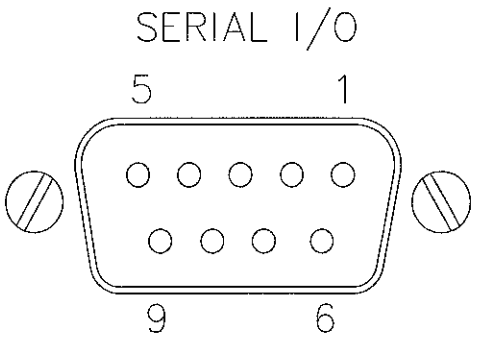


Figure 6-1 9-Pin Connector

Table 6-1 Pin Description

PIN = Pin number
ABR = Abbreviation for function name
O = Signal Out of CR10 to peripheral
I = Signal Into CR10 from peripheral

PIN	ABR	I/O	Description
1	5V	O	5V: Sources 5V DC, 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 CR10 in the telecommunications mode.
4	RXD	I	Receive Data: Serial data transmitted by a peripheral is received on pin 4.
5	ME	O	Modem Enable: Raised when the CR10 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	TE	O	Tape Enable: Powers the cassette recorder during tape transfer.
9	TXD	O	Transmit Data: Serial data is transmitted from the CR10 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, 76800 baud (user selectable).

6.1 Enabling and Addressing Peripherals

Although several peripherals can be connected in parallel to the 9-pin port, the CR10 has only one transmit line (pin 9) and one receive line (pin 4). The CR10 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; this is referred to as 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.

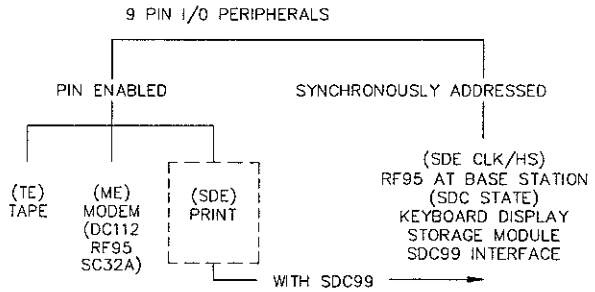


Figure 6-2 Pin-Enabled and Synchronously Addressed Peripherals

6.1.1 Pin-Enabled Peripherals

Two pins are dedicated to specific devices: Tape Enable (pin 8) and Modem Enable (pin 5). Pin 6 (Synchronous Device Enable) can be used as a Print Enable *or* it can be used to address synchronous devices (see Section 6.5).

Modem Enable (ME), pin 5, is raised to enable a modem that has raised the Ring line. Modem/terminal peripherals include telephone modems and computers or terminals using the SC32A RS232 interface. The CR10 interprets a Ring interrupt to come from a modem if the device raises the CR10's Ring line, and holds it high until the CR10 raises the ME line (see 'Ring Interrupts' below). *Only one modem/terminal can be connected to the CR10.*

Tape Enable (TE), pin 8, is raised to enable data transfer to tape. The SC92A Cassette Interface regulates 12 volts from the CR10 to 6V DC to power the RC35 recorder and also provides signal conditioning. *Only one tape interface and recorder can be connected to the CR10.*

Print Peripherals are defined as peripherals which have an asynchronous serial communications port used to *receive* data transferred by the CR10. 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, can 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 CR10 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 behaves 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.

6.1.2 Addressed Peripherals

The CR10 is different to other Campbell Scientific dataloggers because it can address Synchronous Devices (SDs). SDs differ from enabled peripherals in that they are not enabled solely by a hardware line; an SD is enabled by an address synchronously clocked from the CR10 (see Section 6.5).

The CR10 can address up to 16 SDs. Unlike an enabled peripheral, the CR10 establishes communication with an addressed peripheral before data is transferred. During data transfer an addressed peripheral uses pin 7 as a handshake line with the CR10.

Synchronously addressed peripherals include the CR10KD Keyboard/Display, SM716 and SM192 Storage Modules, SDC99 Synchronous Device Interface (SDC99), and RF95 RF Modem when configured as a synchronous device. The SDC99 interface is used to synchronously address peripherals which are normally pin-enabled (see Figure 6-2).

6.2 Ring Interrupts

There are three peripherals which can raise the CR10's Ring line: modems, the CR10KD Keyboard/Display and the RF Modem configured for synchronous device communication (RF-SDC). The RF-SDC is used when the CR10 is installed at a telephone-to-RF base station.

When the Ring line is raised, the processor is interrupted and the CR10 determines which peripheral raised the Ring line through a process of elimination (see Figure 6-3). The CR10 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 CR10 addresses the Keyboard/Display and RF-SDC to determine which device to service (see 'Synchronous Device Communication').

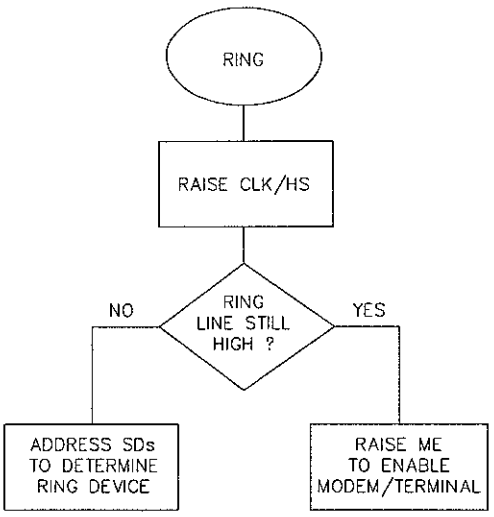


Figure 6-3 Servicing of Ring Interrupts

After the CR10 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 CR10. The one exception is that a modem cannot interrupt an active RF-SDC. 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 and RF-SDC is blocked when the SDE line is high, preventing it from interrupting data transfer to a pin-enabled print device.

6.3 Interrupts During Data Transfer

Instruction 96 is used for on-line data transfer to peripherals (see Section 4). Each peripheral connected to the CR10 requires an Instruction 96 with the appropriate parameter. If the CR10 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 is serviced in 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 an 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 a *8 Mode data transfer the abort flag is checked as follows:

1. Comma delineated ASCII — after every 32 characters.
2. Printable ASCII — after every line.
3. Binary — after every 256 Final Storage locations.
4. Tape — after every block (512 Final Storage locations).

6.4 Modem / Terminal Peripherals

The CR10 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/terminal peripherals include telephone modems and computers or terminals using the SC32A RS232 Interface.

When a modem raises the Ring line, the CR10 responds by raising the ME line. The CR10 must then receive carriage returns until it can establish the baud rate. When the baud rate has been set, the CR10 sends a carriage return, line feed, and '*'.

The ME line is held high until the CR10 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 (two minutes in the Remote Keyboard State).

Some modems are quite noisy when not on line, and it is possible for valid characters to appear in the noise pattern. For this reason, the CR10 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.

6.5 Synchronous Device Communication

The CR10 is different to other Campbell Scientific dataloggers because it can address Synchronous Devices (SDs). SDs differ from enabled peripherals in that they are not enabled solely by a hardware line. An SD is enabled by an address synchronously clocked from the CR10. Up to 16 SDs may be addressed by the CR10, requiring only three pins of the 9-pin connector.

The 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 connected to control ports 1, 2 and 3. (Although the communication protocol for SDMs is very similar, their addressing is independent of SDC addresses and they do not have a Ring line.)

6.5.1 SD States

The CR10 and the SDs use a combination of the Ring, Clock Handshake (CLK/HS) and Synchronous Device Enable (SDE) lines to establish communication. The CR10 can put the SDs into one of six states.

State 1, The SD Reset State

The CR10 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 line low if a modem/terminal is holding it high. Data on TXD is ignored by the SD.

State 2, The SD Addressing State

The CR10 places the SDs in the addressing state by raising CLK/HS followed by or simultaneously raising SDE (see Figure 6-4). TXD must be low while SDE and CLK/HS are changing to the high state.

State 2 requires all SDs to drop the Ring line and prepare for addressing. The CR10 then synchronously clocks eight 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 CR10. The CR10 can only address one device per State 2 cycle. More than one SD may respond to the address, however. State 2 ends when the eighth bit is received by the SD.

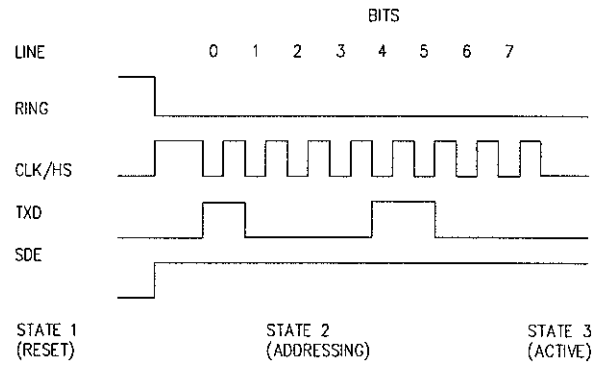


Figure 6-4 Addressing Sequence for the RF Modem

SDs implemented with shift registers decode the four 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 6-2 and are decoded with respect to the TXD line.

Table 6-2 Synchronous Device 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
CR10 Keyboard	0	0	1	0	0	X	X	1
CR10 Display	0	0	1	0	1	X	X	1
CR10 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 CR10 forcing it to reset.

Active SDs have different acknowledgement and communication protocols. Once addressed, the SD is free to use the CLK/HS, TXD and RXD lines according to its protocol with the CR10. The CR10 may also pulse the SDE line after addressing, as long as the CLK/HS and SDE lines 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 CR10 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.

6.6 Modem / Terminal and Computer Requirements

6.6.1 SC32A Interface

Most modem and print peripherals require the SC32A Optically Isolated RS232 Interface. The SC32A raises the CR10's Ring line when it receives characters from a modem, and converts the CR10's logic levels (0V logic low, 5V logic high) to RS232 logic levels.

When the SC32A receives a character from the terminal/computer (pin 2), 5V is applied to the datalogger Ring line (pin 3) for one second or until the Modem Enable line (ME) goes high. The CR10 waits approximately 40 seconds to receive carriage returns, which it uses to establish baud rate. After the baud rate has been set the CR10 transmits a carriage return, line feed, and '*', and enters the Telecommunications Command State (see Section 5). If the carriage returns are not received within the 40 seconds, the CR10 drops out of telecommunications.

NOTE

1. 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.

2. Please refer to the SC32A manual for pin descriptions and other technical details.

3. The SC929 RS232 Interface Cable is suitable for interfacing between the CR10 and battery-powered (laptop) PCs.

6.6.2 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 (or SC929) RS232 section is taken from these pins. For equipment configured as DTE (see Table 6-3) a direct ribbon cable connects the computer/terminal to the SC32A. Clear to Send (CTS) pin 5, Data Set Ready (DSR) pin 6, and Received Line Signal Detect (RLSD) pin 8 are held high by the SC32A (when the RS232 section is powered) which should satisfy the hardware handshake requirements of the computer/terminal.

Table 6-3 lists the most common RS232 configuration for Data Terminal Equipment.

Table 6-3 DTE Pin Configuration

PIN = 25-pin connector number
ABR = Abbreviation for the function name
O = Signal out of the terminal to another device
I = Signal into the 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.

If the computer/terminal is configured as DCE equipment (pin 2 is an input for RD), a null modem cable is required. Refer to the SC32A manual for details.

6.6.3 Communication Protocol and 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 first (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 -3V with respect to ground, and spacing when the voltage is more positive than +3V.

Most computers use eight bits (one byte) for data communications. The eighth 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 eighth bit is set to either a 1 or a 0 to make the parity of the character correct. The CR10 ignores the eighth bit of a character that is received, and transmits the eighth bit as a binary 0. This method is generally described as 'no parity'.

To separate ASCII characters a start bit is sent before the first bit, and a stop bit is sent after the eighth 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 6-5 shows how the ASCII character '1' is transmitted. When transmitted by the CR10 using the SC32A RS232 interface, spacing and marking voltages are positive and negative, as shown. Signal voltages at the CR10 I/O port are 5V in the spacing condition and 0V in the marking condition.

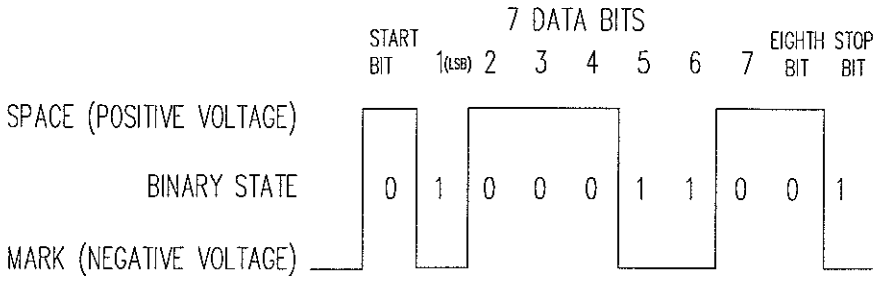


Figure 6-5 Transmitting the ASCII Character 1

Baud Rate

The *baud rate* is the number of bits transmitted per second. The CR10 can communicate at 300, 1200 and 9600 baud. In the Telecommunications State, the CR10 sets its baud rate to match the baud rate of the computer/terminal.

Typically the baud rate of the modem/ computer/terminal is either set with 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. Please contact Campbell Scientific for further details.

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 CR10 is connected to the SC32A RS232 interface and a modem/terminal, and '*' is not received after sending carriage returns:

1. Verify that the CR10 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 6-3). 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 PCs, 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 operator's manual should give you this information.

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 with Table 6-3.

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. Campbell Scientific's GraphTerm and Term programs provide this function for IBM PC/XT/AT/PS-2's and compatibles. The port should be enabled for 300, 1200 or 9600 baud, eight data bits, one stop bit and no parity.

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 CR10. If the baud rate is correct, verify that the computer/terminal is set for eight data bits, and no parity. Garbage will appear if seven data bits and no parity are used. If the computer/terminal is set to eight 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 CR10. These examples detail only the connections, Input Instructions, Program Control Instructions and Processing Instructions needed to make measurements and store the data in engineering units in Input Storage. Output Processing Instructions are omitted (Section 8 contains some processing and program control examples). If you use these examples, remember that ***no output to Final Storage will take place without additional programming****.*

The examples given in this section would probably only be 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 CR10 configuration would exactly duplicate that assumed in an example.

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

7.1 Single-Ended Voltage – LI200SZ Silicon Pyranometer

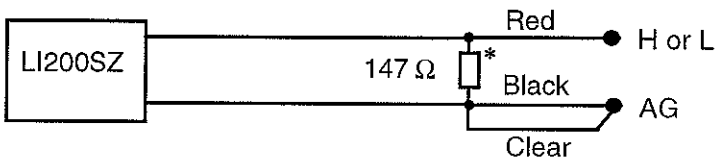
A silicon pyranometer outputs a current which depends on the solar radiation incident on the sensor. The current is measured as the voltage drop across a fixed resistor. The LI200SZ uses a 147Ω resistor. The calibration supplied by LI-COR, the manufacturers of the pyranometer, is given in $\mu A/kWm^{-2}$. The calibration in terms of volts is determined by multiplying the μA calibration by the resistance of the fixed resistor.

The calibration of the pyranometer used in this example is assumed to be $76.9\mu A/kWm^{-2}$, which when multiplied by 147Ω equals $11.3mV/kWm^{-2}$. The multiplier used to convert the voltage reading to kWm^{-2} is therefore $1/11.3mV/kWm^{-2} = 0.0885\ kWm^{-2}/mV$.

Most LI-COR calibrations are between 60 and $90\mu A/kWm^{-2}$, which correspond to calibrations of 8.8 to $13.2mV/kWm^{-2}$ respectively. Above the earth's atmosphere, the flux density through a surface normal to the solar beam is $1.36kWm^{-2}$; radiation at the surface of the earth will be less than this. Thus, the 25mV scale provides an adequate range (since $13.2mV/kWm^{-2} \times 1.36\ kWm^{-2} < 25mV$).

Connections

The pyranometer output is measured with a single-ended voltage measurement on channel 5. There are twice as many single-ended channels as differential channels and they are numbered accordingly: single-ended channel 5 is the high side of differential channel 3 (3H); the low side (3L) is single-ended channel 6.



* Fitted by Campbell Scientific

Figure 7-1 Wiring Diagram for LI200SZ

Program

```
01:   P 1        Volt (SE)
  01:   1        Rep
  02:   33       25mV 50Hz rejection Range
  03:   5        IN Chan
  04:   1        Loc [:RAD kW/m2]
  05:   .0885    Mult
  06:   0        Offset
```

7.2 Differential Voltage Measurement

Some sensors either contain or require active signal conditioning circuitry to provide an easily measured analogue voltage output. Generally, the output is referenced to the sensor ground. The associated current drain usually requires a power source external to the CR10. A typical connection scheme where AC power is not available and both the CR10 and sensor are powered by an external battery is shown in Figure 7-2. Since a single-ended measurement is referenced to the CR10 ground, any voltage difference between the sensor ground and CR10 ground becomes a measurement error. A differential measurement avoids this error by measuring the signal between the two leads without reference to ground. This example analyses the potential error on a water pH measurement using a Martek Mark V water quality analyser.

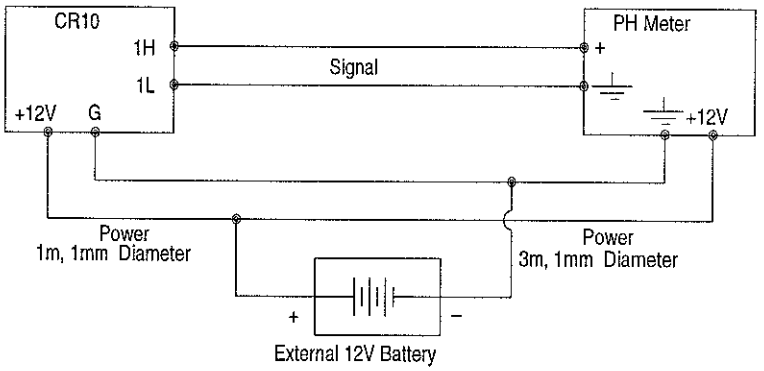


Figure 7-2 Typical Connection for Active Sensor with External Battery

The wire used to supply power from the external battery is 1mm diameter with an average resistance of 2.2 ohms/100m. The power leads to the CR10 and pH meter are 1m and 3m, respectively. Typical current drain for the pH meter is 300mA. When making measurements, the CR10 draws about 35mA. Since

voltage is equal to current multiplied by resistance (V=IR), ground voltages at the pH meter and the CR10 relative to battery ground are:

pH meter ground = 0.3 x 2.2/100 x 3 = +0.0198V

CR10 ground = 0.035 x 2.2/100 x 1 = +0.0008V

Ground at the pH meter is 0.0190V higher than ground at the CR10. The meter output is 0-1 volt referenced to meter ground, for the full range of 14 pH units, or 0.0714V/pH. Thus, if the output is measured with a single-ended voltage measurement, it is 0.0190V or 0.266pH units too 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 0.014 is used to convert the millivolt output into pH units.

Program

```
01:   P  2        Volt (DIFF)
  01:    1        Rep
  02:   35        2500mV 50Hz rejection
  03:    1        IN Chan
  04:    1        Loc [:pH  ]
  05:    0.014    Mult
  06:    0        Offset
```

7.3 Thermocouple Temperatures using the Optional 10TCRT to Measure the Reference Temperature

The 10TCRT Thermocouple Reference is a temperature reference for thermocouples monitored with the CR10. When installed, the 10TCRT lies between the two analogue input terminal strips of the CR10 Wiring Panel (see Figure 7-3). The 10TCRT circuitry, measurement and specifications are equivalent to Campbell Scientific's 107 Temperature Probe.

The 10TCRT is connected to single-ended channel 1 (1H), excitation channel 3 (E3) and analogue ground (AG). The temperature is measured with Instruction 11, which excites the probe with a 2.5V AC excitation, makes a single-ended measurement and calculates the temperature (°C). In this example, five differential thermocouples are measured with Instruction 14.

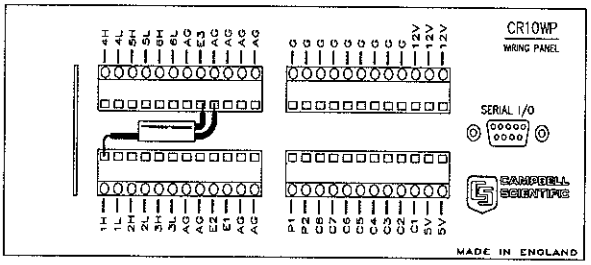


Figure 7-3 10TCRT Mounted on the CR10 Wiring Panel

The temperature (°C) of the 10TCRT is stored in input location 1 and the thermocouple temperatures (°C) in locations 2-6.

Program

```
01:    P  11     Temp 107 Probe
  01:      1     Rep
  02:      1     IN Chan
  03:      3     Excite all reps w/EXchan 3
  04:      1     Loc [:REF TEMP ]
  05:      1     Mult
  06:      0     Offset

02:    P  14     Thermocouple Temp (DIFF)
  01:      5     Reps
  02:     32     7.5mV 50Hz rejection Range
  03:      2     IN Chan
  04:      1     Type T (Copper-Constantan)
  05:      1     Ref Temp Loc REF TEMP
  06:      2     Loc [:TC #1 ]
  07:      1     Mult
  08:      0     Offset
```

7.4 Thermocouple Temperatures using an External Reference Junction

When a number of thermocouple measurements is made at some distance from the CR10, it is often better to use a reference junction box located at the site rather than use the 10TCRT Thermocouple Reference. Use of the external reference junction reduces the required length of expensive thermocouple wire as ordinary copper wire can be used between the junction box (J-box) and CR10. In addition, if the temperature gradient between the J-box and the thermocouple measurement junction is smaller than the gradient between the CR10 and the measurement junction, thermocouple accuracy is improved.

In the following example, an external temperature measurement is used as the reference for five thermocouple measurements. A Campbell Scientific 107 Temperature Probe is used to measure the reference temperature. The connection scheme is shown in Figure 7-4.

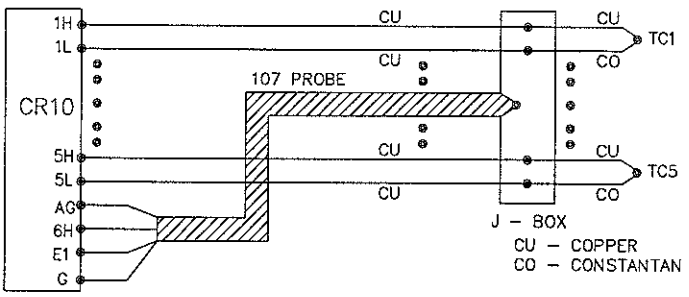


Figure 7-4 Thermocouples with External Reference Junction

The temperature (in °C) of the 107 Probe is stored in input location 1 and the thermocouple temperatures (°C) in locations 2-6.

Program

```
01:    P11    Temp 107 Probe
  01:   1     Rep
  02:   11    IN Chan
  03:   1     Excite all reps w/EXchan 1
  04:   1     Loc [:REF TEMP ]
  05:   1     Mult
  06:   0     Offset

02:    P14    Thermocouple Temp (DIFF)
  01:   5     Reps
  02:   32    7.5mV 50Hz rejection Range
  03:   1     IN Chan
  04:   1     Type T (Copper-Constantan)
  05:   1     Ref Temp Loc REF TEMP
  06:   2     Loc [:TC #1 ]
  07:   1     Mult
  08:   0     Offset
```

7.5 107 Temperature Probe

Instruction 11 excites Campbell Scientific's 107 Thermistor Probe (or the thermistor portion of the 207 temperature and relative humidity probe) with a 2V AC excitation, makes a single ended measurement and calculates the 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 white leads go to analogue 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:  P 11    Temp 107 Probe
  01:   3     Reps
  02:   1     IN Chan
  03:   1     Excite all reps w/EXchan 1
  04:   1     Loc [:107 T #1 ]
  05:   1     Mult
  06:   0     Offset
```

7.6 207 Temperature and RH Probe

Instruction 12 excites and measures the output of the RH portion of the Campbell Scientific 207 Temperature and Relative Humidity probe. This instruction relies on a previously measured temperature (in °C) to compute RH from the probe resistance. In this example, Instruction 11 is used to obtain the temperatures of the three probes which are stored in input locations 1-3; the RH values are stored in input locations 4-6. The temperature measurements are made on single-ended input channels 1-3. The program listed below is a continuation of the program given in the previous example.

Connections

The black leads from the probes are connected to excitation channel 1, the purple leads are connected to analogue ground (AG) and the clear leads are connected to ground (G). The red leads are from the thermistor circuit and are connected to single-ended channels 1-3 (1H, 1L, 2H). The white leads are from the RH circuit and are connected to single-ended channels 4-6 (2L, 3H, and 3L). The correct order must be maintained when connecting the red and white leads; i.e. the red lead from the first probe is connected to single-ended channel 1H and the white lead from that probe is connected to single-ended channel 2L, and so on.

Program *(continuation of previous example)*

```
02:    P12    RH 207 Probe
  01:    3    Reps
  02:    4    IN Chan
  03:    1    Excite all reps w/EXchan 1
  04:    1    Temperature Loc 107 T #1
  05:    4    Loc [:RH #1 ]
  06:    1    Mult
  07:    0    Offset
```

7.7 Anemometer with Photochopper Output

An anemometer with a photochopper transducer produces a pulse output which is measured by the CR10'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 CR10 12V supply; AC power or back-up batteries should be used to compensate for the increased current drain.

Wind speed is desired in metres per second (ms^{-1}). 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:

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

The result of the Pulse Count Instruction (configuration code = 20) is X pulses per sec. (Hz). The multiplier and offset to convert XHz to metres per second are:

$$\text{Wind Speed (m/s)} = (0.01632 \text{ m/s})/\text{rpm} \times (6 \text{ rpm/Hz}) \times \text{XHz} + 0.2 \text{ m/s}$$

and thus:

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

Connections

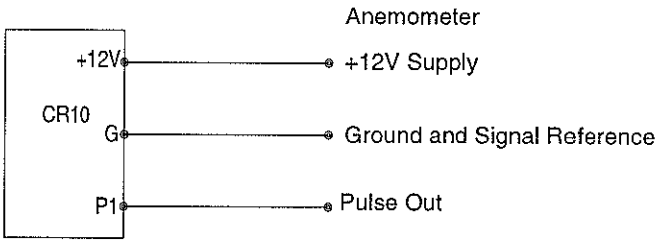


Figure 7-5 Wiring Diagram for Anemometer

Program

```
01:    P 3          Pulse
  01:    1          Rep
  02:    1          Pulse Input Chan
  03:   20          High frequency; Output Hz.
  04:   10          Loc [:WS MPH ]
  05:     .09792    Mult
  06:    0.2        Offset
```

7.8 Tipping Bucket Raingauge with Long Leads

A tipping bucket raingauge is monitored with the Pulse Count instruction configured for Switch Closure. Counts from long intervals are 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 gauge (assuming there were counts in the long intervals). Output is desired in millimetres of precipitation. The gauge is calibrated for a 0.2mm tip, so a multiplier of 0.2 is used.

In a long cable there is appreciable capacitance between the conductors. 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 gauge 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 whenever leads longer than 30m are used.

Connections

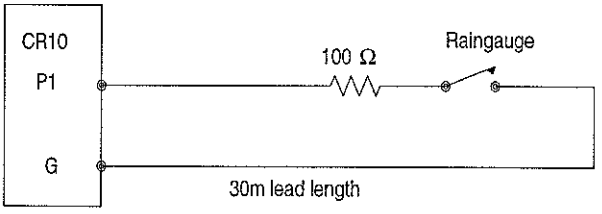


Figure 7-6 Wiring Diagram for Raingauge with Long Leads

Program

```
01:    P 3        Pulse
  01:    1        Rep
  02:    1        Pulse Input Chan
  03:    2        Switch closure
  04:   11        Loc [:RAIN mm ]
  05:    0.2      Mult
  06:    0        Offset
```

7.9 100Ω 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°C to +40°C. The length of the cable from the CR10 to the PRT is 170m.

Figure 7-7 shows the circuit used to measure the PRT. The 10kΩ resistor allows the use of a high excitation voltage and low voltage ranges on the measurements. This ensures 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.

Connections

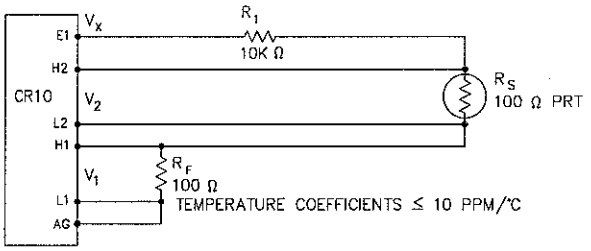


Figure 7-7 Wiring Diagram for PRT in 4-Wire Half Bridge

If the voltage drop across the PRT (V_2) is kept under 50mV, 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=115.54Ω at 40°C). To find the maximum excitation voltage that can be used, we assume V_2 equal to 25mV and use Ohm's Law to solve for the resulting current, I.

$$I = 25/R_S = 25/115.54 = 0.216mA$$

Next solve for V_X:

$$V_X = I(R_1+R_S+R_f) = 2.21V$$

If the actual resistances were the nominal values, the CR10 would not overrange with V_X = 2.2V. To allow for the tolerances in the actual resistances, it is decided

to set V_X equal to 2.1V (e.g. if the 10kΩ resistor is 5% low, then $R_S/(R_1+R_S+R_f)$=115.54/9715.54, and V_X must be 2.102V to keep V_S less than 25mV).

The result of Instruction 9 when the first differential measurement (V_1) is not made on the 2.5V 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 x R_f/R_o). If R_S and R_0 were each exactly 100Ω, the multiplier would be 1. However, neither resistance is likely to be exact. The correct multiplier is found by connecting the PRT to the CR10 and entering Instruction 9 with a multiplier of 1. The PRT is then placed in an ice bath (at 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_0/R_f since R_S=R_0. 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 100Ω resistor must be thermally stable. Its precision is not important if the sensor is to be calibrated because the exact resistance can be incorporated, along with that of the PRT, into the calibrated multiplier. The 10ppm/°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 10kΩ resistor do not affect the result.

CAUTION *Do not* simply copy this program without reading the description associated with it. The multiplier for your application will probably be different.

Program

```
01:    P  9          Full BR w/Compensation
  01:     1          Rep
  02:    33          25mV 50Hz rejection EX Range
  03:    33          25mV 50Hz rejection BR Range
  04:     1          IN Chan
  05:     1          Excite all reps w/EXchan 1
  06:  2100          mV Excitation
  07:     1          Loc [:Rs/Ro ]
  08:     1.0111     Mult (Rf/Ro)
  09:     0          Offset

02:    P 16          Temperature RTD
  01:     1          Rep
  02:     1          R/Ro Loc Rs/Ro
  03:     2          Loc [:TEMP C ]
  04:     1          Mult
  05:     0          Offset
```

7.10 100Ω PRT in 3-Wire Half Bridge

The temperature measurement requirements in this example are the same as for the 4-wire half bridge. In this case, a 3-wire half bridge, Instruction 7, is used to measure the resistance of the PRT. The diagram of the PRT circuit is shown in Fig. 7-8

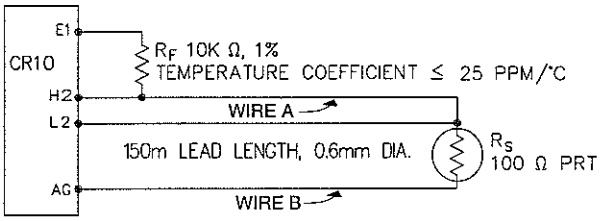


Figure 7-8 3-Wire Half Bridge Used to Measure 100Ω PRT

As in the example in Section 7.9, the excitation voltage is calculated to be the maximum possible, yet allow the 25mV measurement range. The 10kΩ resistor has a tolerance of 1%; thus, the lowest resistance to expect from it is 9.9kΩ. We calculate the maximum excitation voltage (V_X) to keep the voltage drop across the PRT less than 25mV:

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

The excitation voltage used is therefore 2.1V.

The multiplier used in Instruction 7 is determined in the same manner as in Section 7.9. In this example, the multiplier (R_f/R_0) is assumed to be 100.93.

The 3-wire half bridge compensates for wire resistance by assuming that the resistance of wire A is the same as the resistance of wire B (see Figure 7-8). The maximum difference expected in wire resistance is 2%, but is more likely to be of the order of 1%. The resistance of R_S calculated with Instruction 7 is actually R_S plus the difference in resistances of wires A and B. The average resistance of 0.6mm diameter wire is 5.53Ω per 100m, which would give each 150m lead wire a nominal resistance of 8.3Ω. Two percent of 8.3Ω is 0.17Ω. Assuming that the greater resistance is in wire B, the resistance measured for the PRT (R_0 = 100Ω) in the ice bath would be 100.17Ω, 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 7.9, where a 4-wire half bridge is used to measure the PRT resistance.

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

CAUTION

Do not simply copy this program without reading the description associated with it. The multiplier for your application will probably be different.

Program

```
01:      P  7     3 Wire Half Bridge
    01:     1     Rep
    02:    33     25mV 50Hz rejection Range
    03:     3     IN Chan (single-ended channel number)
    04:     1     Excite all reps w/EXchan 1
    05:  2100     mV Excitation
    06:     1     Loc [:Rs/Ro ]
    07:   100.93  Mult
    08:     0     Offset
```

```
02:    P 16   Temperature RTD
  01:     1   Rep
  02:     1   R/aRo Loc Rs/Ro
  03:     2   Loc [:TEMP C ]
  04:     1   Mult
  05:     0   Offset
```

7.11 100Ω PRT in 4-Wire Full Bridge

This example describes obtaining the temperature from a 100Ω 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.

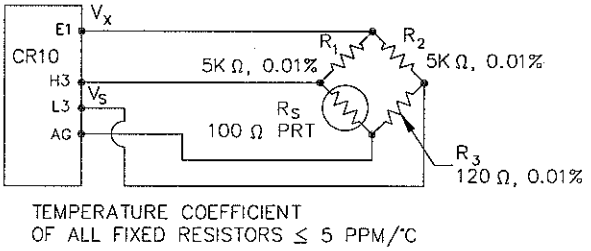


Figure 7-9 Full Bridge Schematic For 100Ω PRT

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 Instruction 59, Bridge Transform:

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

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 desired so that the control algorithm will be able to respond to very small 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 119.7Ω at 50°C. The 120Ω 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 115.8Ω, or:

$$V_S = -802.24x10^{-6} V_X$$

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

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

The relationship between temperature and PRT resistance is a slightly non-linear one. Instruction 16 computes this relationship for a DIN-standard PRT where the nominal temperature coefficient is 0.00385/°C. The change in non-linearity of a PRT with the temperature coefficient of 0.00392/°C is negligible 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.

CAUTION

Do not simply copy this program without reading the description associated with it. The multiplier for your application will probably be different.

Program

```
01:    P     6          Full Bridge
 01:         1          Rep
 02:        31          2.5mV 50Hz rejection Range
 03:         3          IN Chan
 04:         1          Excite all reps w/EXchan 1
 05:      2500          mV Excitation
 06:        11          Loc [:Rs/Ro ]
 07:         0.001      Mult
 08:          .02344    Offset

02:  P     59          BR Transform Rf[X/(1-X)]
 01:         1          Rep
 02:        11          Loc [:Rs/Ro ]
 03:        50          Multiplier (Rf) =R1/R0

03:  P     16          Temperature RTD
 01:         1          Rep
 02:        11          R/Ro Loc Rs/Ro
 03:        12          Loc [:TEMP C ]
 04:          .98214    Mult
 05:         0          Offset
```

7.12 Pressure Transducer – 4-Wire Full Bridge

This example describes a measurement made with a Druck PDCR830 depth measurement pressure transducer. The pressure transducer was ordered for use with 5V positive or negative excitation and has a range of 350mbar or about 3.5m 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 output of the pressure transducer. The high output of the semiconductor strain gauge necessitates the use of the 25mV input range. The sensor is calibrated by connecting it to the CR10 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 10cm of water above the sensor and with 334.6cm of water above the sensor. The output of Instruction 6 is 1000 V_S/V_X or millivolts per volt excitation. At 10cm the reading is 0.19963mV/V and at 334.6 cm the reading is 6.6485mV/V. The multiplier to yield output in cm is therefore:

$$(334.6 - 10)/(6.6485-.19963) = 50.334 \text{ cm/mV/V}$$

The offset is determined after the pressure transducer is installed in the stilling well. The sensor is installed 65cm below the water level at the time of installation. The depth of water at this time is determined to be 72.6cm 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 = 7.48cm.

The lead length is approximately 3m, so there is no appreciable error due to lead wire resistance.

Connections

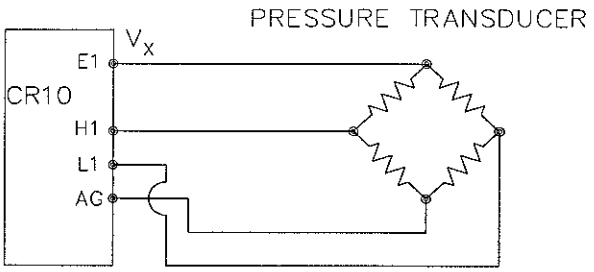


Figure 7-10 Wiring Diagram for Full Bridge Pressure Transducer

Program

```
01:      P  6       Full Bridge
     01:    1       Rep
     02:   33       25mV 50Hz rejection Range
     03:    1       IN Chan
     04:    1       Excite all reps w/EXchan 1
     05: 2500       mV Excitation
     06:    1       Loc [:HTcm ]
     07:   50.334   Mult
     08:    7.48    Offset
```

7.13 Lysimeter — 6-Wire Full Bridge

When a long cable is required between a load cell and the CR10, 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 CR10 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 you would encounter if the actual excitation voltage was not measured and shows the use of a 6-wire full bridge to measure the output of 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 2m in diameter and 1.5m deep. The total weight of the lysimeter with its container is approximately 8000kg. The lysimeter has a mechanically adjustable counterbalance, and changes in weight are measured with a 113.6kg 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 4kg in the mass of the lysimeter will change the force on the load cell by 1kg-force or 980N).

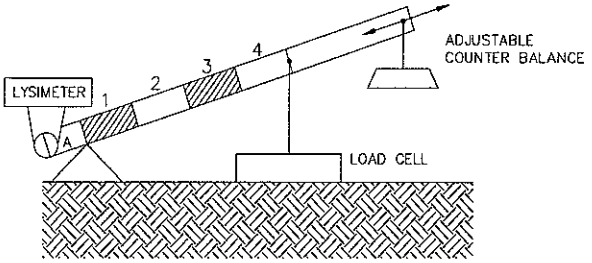


Figure 7-11 Lysimeter Weighing Mechanism

The surface area of the lysimeter is $3.1416m^2$ or $31416cm^2$, so 1cm of rainfall or evaporation results in a 31.416kg change in mass. The load cell can measure ±113.6kg, a 227kg range. This represents a maximum change of 909kg (28cm of water) in the lysimeter before the counterbalance would have to be readjusted.

The cable between the CR10 and the load cell is 0.6mm diameter and 300m long. 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 0.6mm wire is 5.53Ω per 100m. Thus, the resistance in the excitation lead going out to the load cell added to that in the lead coming back to ground is 33Ω. The resistance of the bridge in the load cell is 350Ω. The voltage drop across the load cell is equal to the voltage at the CR10 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 output of 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 100mm of evapotranspiration had occurred, calculation of the change with Instruction 6 would indicate that only 91mm 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, the voltage drop could be corrected with a fixed multiplier. However, the resistance of copper changes by 0.4% per degree C change in temperature. Assume that the cable between the load

cell and the CR10 lies on the soil surface and undergoes a 25°C diurnal temperature fluctuation. If the resistance is 33Ω at the maximum temperature, then at the minimum temperature, the resistance is:

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

The actual excitation voltage at the load cell is then:

$$V_1 = 350/(350+29.7)\ V_X = .92\ V_X$$

The excitation voltage has increased by 1%, relative to the voltage applied at the CR10. In the case where we were recording a 91mm change in water content, there would be a 1mm 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 3mV per volt excitation. If the excitation is 2.5V, the full scale output is 7.5mV; thus, the ±7.5mV range is selected. The calibrated output of the load cell is 2.734mV/V_1 at a load of 100kg. Output is desired in millimetres of water with respect to a fixed point. The mechanical advantage is 4. The calibration in mV/V_1/mm is thus:

$$2.734/V_1/100 \times 3.1416/4 = 0.02147 mV/V_1/mm$$

The reciprocal of this gives the multiplier to convert mV/V_1 into millimetres. (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):

$$1/0.02147 = 46.583 mm/mV/V_1$$

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 7-11, the output voltage would be positive when the load cell was under tension.)

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 1500mm = 375mm). 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 connecting everything up, the counterbalance is adjusted so that the load cell is near the top of its range; this allows 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 375mm is therefore 266. However, it is decided to add this offset in a separate instruction so that the result of Instruction 9 can be used as a ready reminder of the strain on the load cell (range = ±140mm). 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 millimetres is output to Final Storage every hour (the instructions to do this are not shown). The average is used, instead of a sample, in order to cancel out the effects of wind loading on the lysimeter.

Program

```
01:    P  9      Full BR w/Compensation
  01:     1      Rep
  02:    35      2500mV 50Hz rejection EX Range
  03:    32      7.5mV 50Hz rejection BR Range
  04:     1      IN Chan
  05:     1      Excite all reps w/EXchan 1
  06:  2500      mV Excitation
  07:     1      Loc [:RAW mm ]
  08:   46.583   Mult
  09:     0      Offset

02:    P 34      Z=X+F
  01:     1      X Loc RAW mm
  02:   266      F
  03:     2      Z Loc [:mm H2O ]
```

Connections

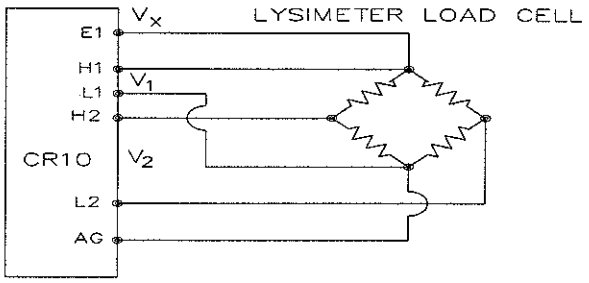


Figure 7-12 6-Wire Full Bridge Connection for Load Cell

7.14 227 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 polarisation of the sensor. (The fast integration time is used because polarisation creates an error in the output.) 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 1kΩ bridge completion resistor. Using data supplied by Delmhorst, Campbell Scientific has computed coefficients for a fifth order polynomial to convert block resistance to water potential in bars. There are two polynomials: one to optimise 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 kΩ) and the -0.1 to -10 bar polynomial requires a multiplier of 0.1 (result in 10000s of ohms). The multiplier is a scaling factor to maintain the maximum number of significant digits in the coefficients of the polynomial.

In this example, we wish to make measurements on six 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 1-6 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 7-12 by changing the value of parameter 3 in Instruction 55 to 7.

Section 8 gives an example using the AM416 Multiplexer to measure the resistances of 16 Soil Moisture Blocks.

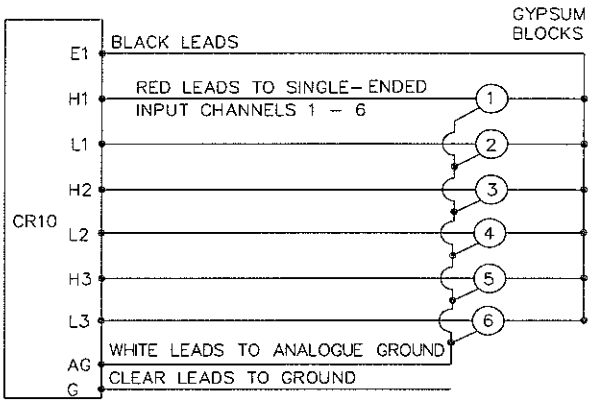


Figure 7-13 Six 227 Gypsum Blocks Connected to the CR10

Program

```
01:      P     5        AC Half Bridge
     01:       6        Reps
     02:      15        2500mV fast Range
     03:       1        IN Chan
     04:       1        Excite all reps w/EXchan 1
     05:    2500        mV Excitation
     06:       1        Loc [:H2O BARS ]
     07:       1        Mult
     08:       0        Offset

02:      P    59        BR Transform Rf[X/(1-X)]
     01:       6        Reps
     02:       1        Loc [:H2O BARS ]
     03:      .1        Multiplier (Rf)

03:      P    55        Polynomial
     01:       6        Reps
     02:       1        X Loc H2O BARS
     03:       1        F(X) Loc [:H2O BARS ]
     04:     .15836     C0
     05:    6.1445      C1
     06:   -8.4189      C2
     07:    9.2493      C3
     08:   -3.1685      C4
     09:     .33392     C5
```

7.15 Non-linear Thermistor in Half Bridge

Virtually any thermistor can be used with the CR10, providing care is taken to set up the bridge correctly. Please contact Campbell Scientific for further details or consult Technical Note 15-87AS.

7.16 Water Level – Geokon's Vibrating Wire Pressure Sensor

Strain gauge sensors of the vibrating wire type have a good reputation for long term stability. These sensors use a change in the frequency of a vibrating wire to sense strain. Two measurements are usually made; the first is the measurement of the frequency of the vibrating wire. The second is an optional measurement of the sensor temperature to allow temperature compensation in the frequency calculation.

Please refer to the AVW1 and AVW4 Vibrating Wire Interfaces manual for further information on vibrating wire sensors and a programming example showing the use of Instruction 28, Vibrating Wire Measurement.

7.17 Paroscientific 'T' Series Pressure Transducer

The Paroscientific 'T' series transducer has two resonating quartz crystals which output frequency signals for temperature and pressure. The pressure output requires temperature compensation. Instruction 27 (Period Measurement) measures the outputs and returns period in microsecond.

Please refer to Technical Note 3-93MP for a programming example.

7.18 SDM Peripherals

The SDM peripherals are measurement and control modules which are controlled by the CR10 through control ports 1, 2 and 3. The instructions for these peripherals are:

101 SDM-INT8 8-channel interval timer

102 SDM-SW8 8-channel switch closure multiplexer

103 SDM-A04 4-channel analogue output multiplexer

104 SDM-CD16 16-channel control port expansion module

Please consult the SDM peripheral manual for programming examples.

Section 8. Processing and Program Control Examples

The examples in this section illustrate the use of Processing and Program Control Instructions, flags, dual Final Storage and the CR10's ability to direct the results of Output Processing Instructions to Input Storage.

The specific examples are not as important as some of the techniques employed. For example:

- *Directing the results of Output Processing to Input Storage is used in the Running Average and Rainfall Intensity examples.*
- *Flag tests are used in the Running Average, Interrupt Subroutine, Converting Wind Direction and Saving Data Prior to Event examples.*
- *Control ports and the use of loops are illustrated in the AM416 example.*
- *An algorithm for a down counter is used in the Saving Data Prior to Event example.*

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 (Instruction 51) to compute the average. The temperatures are stored in locations 11 to 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 (Instruction 54) is then used to move the temperatures from locations 12 to 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 i-5
2:10smpl av	16:Temp i-4
11:Temp i-9	17:Temp i-3
12:Temp i-8	18:Temp i-2
13:Temp i-7	19:Temp i-1
14:Temp i-6	20:Temp i

Where i is current reading, i-1 is previous reading, etc.

```
*          1        Table 1 Programs
   01:    10        Sec. Execution Interval

01:       P17       Panel Temperature
   01:     1        Loc [:Panl Temp]

02:       P14       Thermocouple Temp (DIFF)
   01:     1        Rep
   02:     1        2.5mV slow Range
   03:     1        IN Chan
   04:     1        Type T (Copper-Constantan)
   05:     1        Ref Temp Loc Panl Temp
   06:    20        Loc [:Temp i    ]
   07:     1        Mult
   08:     0        Offset

03:       P51       Spatial Average
   01:    10        Swath
   02:    11        First Loc Temp i-9
   03:     2        Avg Loc [:10smpl av]

04:       P54       Block Move
   01:     9        No. of Values
   02:    12        First Source Loc Temp i-8
   03:     1        Source Step
   04:    11        First Dest. Loc [:Temp i-9 ]
   05:     1        Destination Step

05:       P86       Do
   01:    10        Set high Flag 0 (output)

06:       P70       Sample
   01:     1        Reps
   02:     2        Loc 10smpl av

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 80 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 i-2
2:AVG i-1
3:AVG i
4:3 HR AVG
5:XX mg/M3

```
*          1     Table 1 Programs
  01:      5     Sec. Execution Interval

01:    P  2     Volt (DIFF)
  01:      1     Rep
  02:     35     2500mV 50Hz rejection
  03:      3     IN Chan
  04:      5     Loc [:XX mg/M3 ]
  05:     10     Mult
  06:      0     Offset

02:    P 92     If time is
  01:      0     minutes into a
  02:     60     minute interval
  03:     10     Set high Flag 0 (output)

03:    P 80     Set Active Storage Area
  01:      3     Input Storage Area
  02:      3     Array ID or location

04:    P 71     Average
  01:      1     Rep
  02:      5     Loc XX mg/M3

05:    P 51     Spatial Average
  01:      3     Swath
  02:      1     First Loc AVG i-2
  03:      4     Avg Loc [:3 HR AVG ]

06:    P 80     Set Active Storage Area
  01:      1     Final Storage Area 1
  02:     25     Array ID or location

07:    P 77     Real Time
  01:    110     Day, Hour-Minute

08:    P 70     Sample
  01:      1     Reps
  02:      4     Loc 3 HR AVG

09:    P 91     If Flag/Port
  01:     10     Do if flag 0 (output) is high
  02:     30     Then Do

10:    P 54     Block Move
  01:      2     No. of Values
  02:      2     First Source Loc AVG i-1
  03:      1     Source Step
  04:      1     First Dest. Loc [:AVG i-2  ]
  05:      1     Destination Step

11:    P 95     End

12:    P        End Table 1
```

8.2 Rainfall Intensity

In this example, the total rainfall for the last 15 minutes is output only if any rain has occurred. The program makes use of the CR10's ability to direct the output of Output Processing Instructions to Input Storage.

Every 15 minutes, the total rainfall 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)
2:15min tot

```
*          1      Table 1 Programs
  01:     60      Sec. Execution Interval

01:     P  3      Pulse
  01:      1      Rep
  02:      1      Pulse Input Chan
  03:      2      Switch Closure
  04:      1      Loc [:Rain (mm)]
  05:     .2      Mult
  06:      0      Offset

02:     P 92      If time is
  01:      0      minutes into a
  02:     15      minute interval
  03:     10      Set high Flag 0 (output)

03:     P 80      Set Active Storage Area
  01:      3      Input Storage Area
  02:      2      Array ID or location

04:     P 72      Totalize
  01:      1      Rep
  02:      1      Loc Rain (mm)

05:     P 89      If X<=>F
  01:      2      X Loc 15min tot
  02:      2      <>
  03:      0      F
  04:     30      Then Do

06:     P 80      Set Active Storage Area
  01:      1      Final Storage Area 1
  02:     25      Array ID or location

07:     P 77      Real Time
  01:    110      Day, Hour-Minute
08:     P 70      Sample
  01:      1      Reps
  02:      2      Loc 15min tot

09:     P 95      End

10:     P         End Table 1
```

8.3 Using Control Ports and Loop to Run an AM416 Multiplexer

This example uses an AM416 Analogue Multiplexer to measure the output of 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 minimise thermocouple errors created by thermal gradients. A Campbell Scientific 107 Temperature Probe is centrally located on the multiplexer board and used as a thermocouple temperature reference.

NOTE 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 quarter the cost of the 227 blocks.

Control ports are used to reset the AM416 and clock it through its channels. The sequence of the program is:

1. Temperature of the 107 probe located at the AM416 is measured to provide a TC reference.
2. CR10 sets the port high which resets the AM416.
3. A loop is entered; within each pass:
   The port clocking the AM416 is pulsed.
   The connected TCs and moisture blocks are measured.
4. CR10 sets the port controlling AM416 reset low.
5. Soil moisture measurements are converted to block resistances.

The input locations in which the temperature and soil moisture measurements are stored are indexed to the loop counter (see description of Instruction 87 in Section 12). An indexed location is incremented by one with each pass through the loop. For example, on the first pass the 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 19. After 16 loop passes, temperature and soil moisture measurements occupy locations 2 to 17 and 18 to 33 respectively. Connections are shown in Figure 8-1.

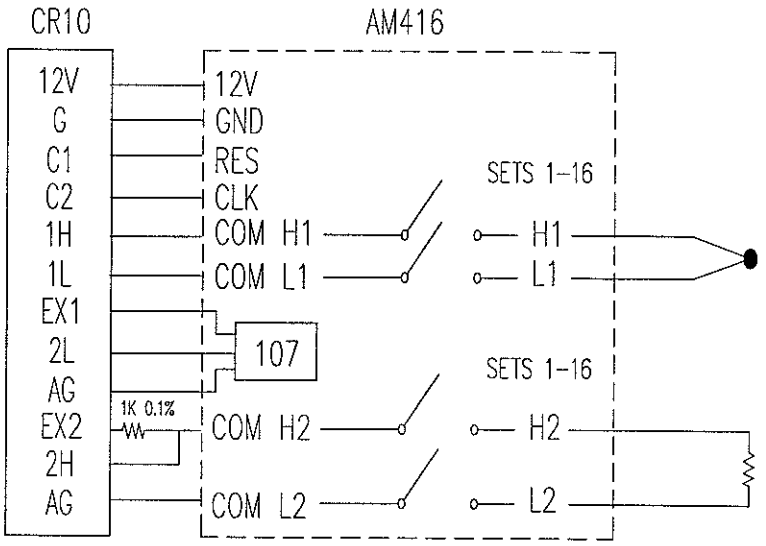


Figure 8-1 AM416 Wiring Diagram For Thermocouple and Soil Moisture Block Measurements

Example Program Multiplexing Thermocouples and Soil Moisture Blocks

```
*          1      Table 1 Programs
 01:     600      Sec. Execution Interval

01:   P   11      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:   P   86      Do
 01:      41      Set high Port 1

03:   P   87      Beginning of Loop
 01:       0      Delay
 02:      16      Loop Count

04:   P   86      Do
 01:      72      Pulse Port 2

05:   P   14      Thermocouple Temp (DIFF)
 01:       1      Rep
 02:      31      2.5mV 50Hz 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:   P    5      AC Half Bridge
 01:       1      Rep
 02:      14      250mV 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:   P   95      End

08:   P   86      Do
 01:      51      Set low Port 1

09:   P   59      BR Transform Rf[X/(1-X)]
 01:      16      Reps
 02:      18      Loc [:SOIL M#1 ]
 03:      .1      Multiplier (Rf)

10:   P   92      If time is
 01:       0      minutes into a
 02:      60      minute interval
 03:      10      Set high Flag 0 (output)
```

```
11:     P77     Real Time
   01: 110      Day,Hour-Minute

12:     P71     Average
   01:  33      Reps
   02:   1      Loc REF TEMP

13:     P       End Table 1
```

8.4 Output Interval not a Multiple of 1 Second Synchronised to Real Time

Instruction 92 has a resolution of 1 second. If processed output is required at an interval which is not a multiple of 1 second, Instructions 18 and 89 can be used to set the Output Flag at intervals which are a multiple of 0.125s.

Instruction 18 takes the time (seconds into minute, minutes into day, or hours into year), performs a modulo divide by a user-specified value and loads it into an input location.

When the modulo divisor divides evenly into the interval (seconds into the minute, etc.) the result is zero. Thus the use of Instruction 18 results in a counter in an input location which periodically goes to zero. In this example, seconds into the minute is modulo divided by 30. Thus the counter counts up to 29.5 then goes to 0 (i.e. every 30 seconds). Seconds into the minute has a resolution of 0.125 seconds.

Instruction 89 is used to set the Output Flag when the seconds counter is less than 0.5 (the execution interval). With this short program the Output Flag could be set when the seconds counter equalled 0. However, if Instruction 18 followed a series of instructions that took longer than 0.125 seconds to execute or was in Table 2 (executed at the same interval as an extensive Table 1) the time at which Instruction 18 was executed might be 0.125 seconds or more beyond the modulo divisor. The value output would then not equal 0. Setting the Output Flag when the seconds counter is less than the execution interval avoids this problem.

Using Instruction 18 keeps the output interval synchronised with real time. 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 compiled and started running. If the table overran its execution interval, the output interval would not be the count multiplied by the execution interval, but some longer interval.

In this example a temperature is measured every 0.5 seconds and the average is output every 30 seconds.

Instruction 89 is used to determine when the result of Instruction 18 is less than the execution interval (0.5s in this case). This is done in preference to comparing the result of Instruction 18 to zero because in a more complex program, Instruction 18 could be executed more than 0.125s after table execution began. In such a case, the result of the modulo divide would not be zero but would still be less than the table execution interval.

Input Location Labels:

```
1:TEMP DEG C
10:30 SEC 0

*          1      Table 1 Programs
   01:     .5     Sec. Execution Interval

01:       P18     Time
   01:     0      Seconds into current minute
   02:    30      Mod/by
   03:    10      Loc [:30 SEC 0 ]

02:       P17     Module Temperature
   01:     1      Loc [:REF TEMP ]

03:       P14     Thermocouple Temp (DIFF)
   01:     1      Rep
   02:    11      2.5mV fast Range
   03:     2      IN Chan
   04:     2      Type E (Chromel-Constantan)
   05:     1      Ref Temp Loc REF TEMP
   06:     2      Loc [:TC TEMP  ]
   07:     1      Mult
   08:     0      Offset

04:       P89     If X<=>F
   01:    10      X Loc 30 SEC 0
   02:     4      <
   03:     .5     F
   04:    10      Set high Flag 0 (output)

05:       P71     Average
   01:     1      Rep
   02:     2      Loc TC TEMP

06:       P       End Table 1
```

8.5 Interrupt Subroutine Used to Count Switch Closures (Raingauge)

A subroutine given the label of 98 will be executed when control port 8 goes high (see Section 1). In this example, Subroutine 98 and control port 8 are substituted for a pulse counting channel to count switch closures on a tipping bucket raingauge.

The subroutine adds 0.2 (the bucket is calibrated for 0.2mm per tip) to an input location and uses Instruction 22 to delay 0.2 seconds.

The delay is to ensure that any switch bouncing 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. When it is time to store data, this total is sampled to Final Storage and zeroed by the program in Program Table 1.

NOTE An interrupt-driven subroutine can interrupt a table while the Output Flag is set. The CR10 will complete the instruction it is executing, execute the subroutine, and then resume executing the table.

If the subroutine always added the count to the same location and a tip occurred while the total rainfall was being sampled, the subroutine would add the count to the input location before the location was zeroed, causing the count to be missed.

To overcome this in the example, the subroutine adds the count to another input location (location 13) when the Output Flag is high. Program Table 1 sets the Output Flag low after zeroing the location where the normal total is kept (location 12). The value in location 13 is then added to the value in location 12 and location 13 is zeroed.

To test the accuracy of this method of counting switch closures, this example has the two pulse inputs also reading raingauges. In Program Table 1, the two normal pulse inputs are read and the hourly totals output to Final Storage with Instruction 72.

The raingauge is connected as shown below. When the switch closes, 5V is applied to port 8 which causes the subroutine to be executed.

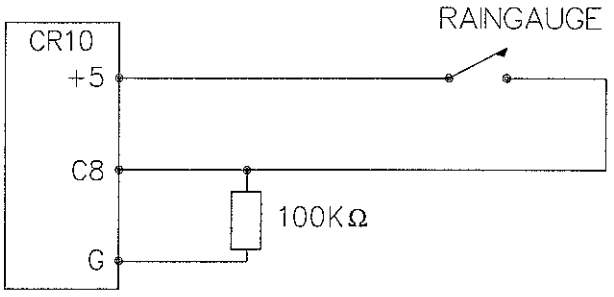


Figure 8-2 Connections for Raingauge

Input location Labels:

10:Rain #1	(from pulse count)
11:Rain #2	(from pulse count)
12:Rain #3	(from subroutine 98 while Output Flag is low)
13:Rain alt	(from subroutine 98 while Output Flag is high)

```
*          1     Table 1 Programs
   01:    10     Sec. Execution Interval

01:       P 3    Pulse
   01:     2     Reps
   02:     1     Pulse Input Chan
   03:     2     Switch Closure
   04:    10     Loc [:Rain #1  ]
   05:    .2     Mult
   06:     0     Offset
02:      P 92    If time is
  01:      0     minutes into a
  02:     60     minute interval
  03:     10     Set high Flag 0 (output)
```

```
03:    P 77      Real Time
  01:   110      Day,Hour-Minute

04:    P 72      Totalize
  01:     2      Reps
  02:    10      Loc Rain #1

05:    P 91      If Flag/Port
  01:    10      Do if flag 0 (output) is high
  02:    30      Then Do

06:    P 70      Sample
  01:     1      Reps
  02:    12      Loc Rain #3

07:    P 30      Z=F
  01:     0      F
  02:     0      Exponent of 10
  03:    12      Z Loc [:Rain #3  ]

08:    P 86      Do
  01:    20      Set low Flag 0 (output)

09:    P 33      Z=X+Y
  01:    13      X Loc Rain alt
  02:    12      Y Loc Rain #3
  03:    12      Z Loc [:Rain #3  ]

10:    P 30      Z=F
  01:     0      F
  02:     0      Exponent of 10
  03:    13      Z Loc [:Rain alt ]

11:    P 95      End

12:    P         End Table 1

*         3      Table 3 Subroutines

01:    P 85      Beginning of Subroutine
  01:    98      Subroutine Number

02:    P 91      If Flag/Port
  01:    10      Do if flag 0 (output) is high
  02:    30      Then Do

03:    P 34      Z=X+F
  01:    13      X Loc Rain alt
  02:   0.2      F
  03:    13      Z Loc [:Rain alt ]

04:    P 94      Else

05:    P 34      Z=X+F
  01:    12      X Loc Rain #3
  02:    .2      F
  03:    12      Z Loc [:Rain #3  ]
```

```
06:    P 95      End

07:    P 22      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

08:    P 95      End

09:    P         End Table 3
```

8.6 SDM-AO4 Analogue Output Module to Strip Chart

This example illustrates the use of the SDM-AO4 4-Channel Analogue Output Module to output four analogue voltages to a strip chart.

While of questionable value because of current requirements and strip chart reliability, some regulations require strip chart backup of weather data. The SDM-AO4 can be used with the CR10 to provide analogue 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-AO4. The four voltage values to be output must be stored (in mV) in adjacent Input Storage locations, the first of which is referenced in Instruction 103.

The following program measures the output of the sensors every five seconds. The readings are moved to another four locations and scaled to a 0 to 1000mV output for the SDM-AO4. Wind direction is changed from a 0-360 degree input to an output representing 0-540 degrees. This conversion is done in a subroutine which is described in the next example.

The example also includes instructions to output wind vector, average temperature and solar radiation every hour.

Input Location Labels:

1:WS
2:0-360 WD
3:Ta
4:SR
5:WS output
6:0-540 out
7:Ta output
8:SR output
10:0-540 WD

```
*       1       Table 1 Programs
 01:    5       Sec. Execution Interval

01:  P  3       Pulse
 01:    1       Rep
 02:    1       Pulse Input Chan
 03:   22       Switch Closure
 04:    1       Loc [:WS       ]
 05:    1.789   Mult
 06:    1       Offset

02:  P  4       Excite,Delay,Volt(SE)
 01:    1       Rep
 02:   14       250mV 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 [:0-360 WD ]
 08:     .7273  Mult
 09:    0       Offset

03:  P 11       Temp 107 Probe
 01:    1       Rep
 02:    2       IN Chan
 03:    2       Excite all reps w/EXchan 2
 04:    3       Loc [:Ta       ]
 05:    1.8     Mult
 06:   32       Offset

04:  P  1       Volt (SE)
 01:    1       Rep
 02:    2       7.5mV slow Range
 03:    3       IN Chan
 04:    4       Loc [:SR       ]
 05:     .14493 Mult
 06:    0       Offset

05:  P 92       If time is
 01:    0       minutes into a
 02:   60       minute interval
 03:   10       Set high Flag 0 (output)

06:  P 54       Block Move
 01:    4       No. of Values
 02:    1       First Source Loc WS
 03:    1       Source Step
 04:    5       First Dest. Loc [:WS output]
 05:    1       Destination Step

07:  P 86       Do
 01:    1       Call Subroutine 1
```

```
08:   P 53          Scaling Array (A*loc +B)
 01:     5          Start Loc [:WS output]
 02:    10          A1 SCALE WS, 0-100 M/S = 0-1000mV
 03:     0          B1
 04:     1.8519     A2 SCALE WD, 0-540 DEG = 0-1000mV
 05:     0          B2
 06:     5.7143     A3 SCALE TEMP, -25 - 100 F =0-1000mV
 07:    25          B3
 08:  1000          A4 SCALE RADIATION, 0-1KW/M^2 = 0-1000mV
 09:     0          B4

09:   P103          SDM-AO4
 01:     4          Reps
 02:    30          Address
 03:     5          Loc WS output

10:   P 92          If time is
 01:     0          minutes into a
 02:    60          minute interval
 03:    10          Set high Flag 0 (output)

11:   P 69          Wind Vector
 01:     1          Rep
 02:   180          Samples per sub-interval
 03:    00          US, DV, SD (Polar Sensor)
 04:     1          Wind Speed/East Loc WS
 05:     2          Wind Direction/North Loc 0-360 WD

12:   P 71          Average
 01:     2          Reps
 02:     3          Loc Ta

13:   P            End Table 1
```

8.7 Converting 0-360 Wind Direction Output to 0-540 For Strip Chart

If 0-360 degree wind direction is sent to a strip chart the discontinuity at 0/360 causes 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 potentiometer on the windvane (direction changes from 540-180 and from 0-360 so the pen only jumps once when the wind is from the north or south).

When faced with the necessity for 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 potentiometer, it can be used with the CR10 since Instruction 69, Wind Vector, will work with this output.)

To change 0-360 degrees to 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 the reading 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 by the previous example to output an analogue voltage to a strip chart.

Input Location Labels:

2:0-360 WD
6:0-540 out
10:0-540 WD

```
*          3       Table 3 Subroutines

01:     P 85       Beginning of Subroutine
  01:      1       Subroutine Number

02:     P 89       If X<=>F
  01:     10       X Loc 0-540 WD
  02:      3       >=
  03:    270       F
  04:     30       Then Do

03:     P 86       Do
  01:     11       Set high Flag 1

04:     P 94       Else

05:     P 86       Do
  01:     21       Set low Flag 1

06:     P 95       End

07:     P 31       Z=X
  01:      2       X Loc 0-360 WD
  02:     10       Z Loc [:0-540 WD]

08:     P 89       If X<=>F
  01:     10       X Loc 0-540 WD
  02:      4       <
  03:    180       F
  04:     30       Then Do

09:     P 91       If Flag/Port
  01:     11       Do if flag 1 is high
  02:     30       Then Do

10:     P 34       Z=X+F
  01:     10       X Loc 0-540 WD
  02:    360       F
  03:     10       Z Loc [:0-540 WD]

11:     P 31       Z=X
  01:     10       X Loc 0-540 WD
  02:      6       Z Loc [0-540 out]

12:     P 95       End

13:     P 95       End

14:     P 95       End

15:     P          End Table 3
```

8.8 Use of Two Final Storage Areas – Saving Data Prior to Event

One of the uses of two Final Storage Areas is to save a fixed amount of data before and after some event.

In this example, the output of a load cell is measured every second. It is assumed that at some random interval the load will exceed 11.4kg for less than 10 seconds. Exceeding 11.4kg 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 11.4kg).

Every second the output of the load cell is measured; hours-minutes, seconds and the load are output to Final Storage Area 2 (four values with the Array ID). 84 locations are allocated to Final Storage Area 2. Thus, Area 2 holds 21 seconds (four values/second x 21 seconds = 84 locations).

When 11.4kg is exceeded, the value 10 is loaded into an input location and flag 1 is set high. The input location is then used as a down counter. The flag indicates that 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 and flag 1 is set low. Instruction 96 is used with output code 81, which transfers the entire contents of Final Storage Area 2 to Final Storage Area 1 in chronological order, oldest to newest.

The down counter is set to 10 instead of 11 because it is decremented after checking to see if it is 0.

Input Location Labels:

1:FORCE KG
2:DOWN CNT

```
*          1        Table 1 Programs
  01:      1        Sec. Execution Interval

01:     P  6        Full Bridge
  01:      1        Rep
  02:     32        7.5mV 50Hz rejection Range
  03:      1        IN Chan
  04:      1        Excite all reps w/EXchan 1
  05:   2500        mV Excitation
  06:      1        Loc [:FORCE KG]
  07:   15.152      Mult
  08:      0        Offset

02:     P 86        Do
  01:     10        Set high Flag 0 (output)

03:     P 80        Set Active Storage Area
  01:      2        Final Storage Area 2
  02:     10        Array ID or location

04:     P 77        Real Time
  01:     11        Hour-Minute,Seconds
```

```
05:    P 70        Sample
  01:     1        Reps
  02:     1        Loc FORCE KG

06:    P 89        If X<=>F
  01:     1        X Loc FORCE KG
  02:     3        >=
  03:  11.4        F
  04:    30        Then Do

07:    P 91        If Flag/Port
  01:    21        Do if flag 1 is low
  02:    30        Then Do

08:    P 86        Do
  01:    11        Set high Flag 1

09:    P 30        Z=F
  01:    10        F
  02:     0        Exponent of 10
  03:     2        Z Loc [:DOWN CNT ]

10:    P 95        End

11:    P 95        End

12:    P 89        If X<=>F
  01:     2        X Loc DOWN CNT
  02:     1        =
  03:     0        F
  04:    30        Then Do

13:    P 91        If Flag/Port
  01:    11        Do if flag 1 is high
  02:    30        Then Do

14:    P 96        Serial Output
  01:    81        All data to other FS Area

15:    P 86        Do
  01:    21        Set low Flag 1

16:    P 95        End

17:    P 94        Else

18:    P 34        Z=X+F
  01:     2        X Loc DOWN CNT
  02:    -1        F
  03:     2        Z Loc [:DOWN CNT ]

19:    P 95        End

20:    P           End Table 1

*         A        Mode 10 Memory Allocation
  01:    28        Input Locations
  02:    64        Intermediate Locations
  03:    84        Final Storage Area 2
```

8.9 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 #
0	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. To enter the recharge phase of the test, the operator enters the *6D 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 50psi Druck 930/ti with a calibration of 49.93mV/10V of excitation or 4.993mV/V (at 50psi). Your calibration will be different. An excitation voltage of 1500mV yields a maximum signal of 7.489mV at 50psi, fully utilising the 7.5mV Input Range to provide the best resolution.

The multiplier, m, is calculated to provide depth of water in metres:

m = (50psi/4.993 mV/V) * (0.703m/psi)

m = 7.04m/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. The offset equals the initial distance of 14.4m (47.23 feet) plus the initial reading of 16.69m (54.77 feet), or 31.09m (102 feet).

```
*          1        Table 1 Programs
  01:     10        Sec. Execution Interval
```

User must toggle flag 1 to start measurements

```
01:     P 91        If Flag/Port
  01:     11        Do if flag 1 is high
  02:      0        Go to end of program table
```

Loop 1, Output every 10 seconds for 10 minutes

```
02:     P 87        Beginning of Loop
  01:      1        Delay
  02:     60        Loop Count
```

```
03:    P 86        Do
  01:     1        Call Subroutine 1

04:    P 95        End
```

Loop 2, Output every 30 seconds for 20 minutes

```
05:    P 87        Beginning of Loop
  01:     3        Delay
  02:    40        Loop Count

06:    P 86        Do
  01:     1        Call Subroutine 1

07:    P 95        End
```

Loop 3, Output every 1 minute for 70 minutes

```
08:    P 87        Beginning of Loop
  01:     6        Delay
  02:    70        Loop Count

09:    P 86        Do
  01:     1        Call Subroutine 1

10:    P 95        End
```

Loop 4, Output every 2 minutes for 200 minutes

```
11:    P 87        Beginning of Loop
  01:    12        Delay
  02:   100        Loop Count

12:    P 86        Do
  01:     1        Call Subroutine 1

13:    P 95        End
```

Loop 5, Output every 5 minutes for 700 minutes

```
14:    P 87        Beginning of Loop
  01:    30        Delay
  02:   140        Loop Count

15:    P 86        Do
  01:     1        Call Subroutine 1

16:    P 95        End
```

Loop 6, Output every 10 minutes until stopped by user

```
17:    P 87        Beginning of Loop
  01:    60        Delay
  02:     0        Loop Count
```

```
18:    P 86        Do
  01:     1        Call Subroutine 1

19:    P 91        If Flag/Port
  01:    21        Do if flag 1 is low
  02:    31        Exit Loop if true

20:    P 95        End

21:    P           End Table 1

*         3        Table 3 Subroutines

01:    P 85        Beginning of Subroutine
  01:     1        Subroutine Number

02:    P  6        Full Bridge
  01:     1        Rep
  02:    32        7.5mV 50Hz rejection Range
  03:     1        IN Chan
  04:     1        Excite all reps w/EXchan 1
  05:  1500        mV Excitation
  06:     1        Loc [:LEVEL M]
  07:    7.04      Mult
  08:   31.09      Offset

03:    P 86        Do
  01:    10        Set high Flag 0 (output)

04:    P 77        Real Time
  01:   111        Day,Hour-Minute,Seconds

05:    P 70        Sample
  01:     1        Reps
  02:     1        Loc LEVEL M

06:    P 95        End

07:    P           End Table 3
```

Section 9. Input / Output Instructions

Table 9-1 Input Voltage Ranges and Codes

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

[1] *Differential measurement; resolution for single-ended measurement is twice value shown.*

[2] *272µs on ±2500mV range.*

[3] *25µs on ±2500mV range.*

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.

Instruction 1: Single-Ended Volts

This instruction is used to measure the voltage at a single-ended input with respect to ground. The output is in millivolts.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code (Table 9-1)
03:	2	Channel number for first measurement
04:	4	input location for first measurement
05:	FP	Multiplier
06:	FP	Offset

Input locations altered: 1 per measurement

Instruction 2: Differential Volts

This instruction measures 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 ±2.5V of the CR10's ground (see description of Common Mode Range in Section 14). Pyranometer and thermopile sensors require a jumper between LO and power ground (G) to keep the signal inside the common mode range. The output is in millivolts.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code (Table 9-1)
03:	2	Channel number for first measurement
04:	4	Input location for first measurement
05:	FP	Multiplier
06:	FP	Offset

Input locations altered: 1 per measurement

Instruction 3: Pulse Count

There are three types of pulse input which may be measured with the Pulse Count Instruction:

High Frequency Pulse

In this configuration, the minimum detectable pulse width is 2µs, i.e. the maximum frequency is 250kHz with a 50% duty cycle. The 8-bit counter has a maximum input frequency of 2000Hz and the 16-bit option a maximum of 250kHz. The count is incremented when the input voltage changes from below 1.5V to above 3.5V. The maximum input voltage is +20V. A problem, however, arises when the pulse is actually a low frequency signal (below about 10Hz) *and* the positive voltage excursion exceeds 5.6V DC.

When this happens, the excess voltage is shunted to the CR10 5V DC supply, with the current limited by an internal 10kΩ resistor. When this extra current source exceeds the quiescent current needs of the CR10 (about 0.7mA), the 5V DC supply starts to rise, upsetting all analogue measurements.

Thus, pulses whose positive voltage portion exceeds 5.6V DC with a duration longer than 100ms need external conditioning. One method would be to use a 4 to 5.6V zener diode from the signal to ground. The simplest method, however, is to add an external 20kΩ resistor in series with the signal (see Figure 9-1). This limits the current for pulses up to 20V DC such that it will not upset the CR10 5V DC supply.

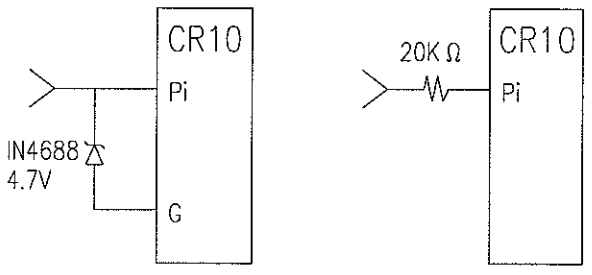


Figure 9-1 Conditioning for Long Duration Voltage Pulses

Low Level AC

This configuration is used for measuring the frequency of AC signals from magnetic pulse flow transducers or other low voltage, sine wave inputs. The minimum input voltage is 6mV RMS. Input hysteresis is 11mV. The maximum AC input voltage is 20V RMS. The maximum input frequency ranges from 100Hz at 20mV

RMS to 1000Hz at 150mV or greater. If you need to measure higher frequency AC signals, please refer to Technical Note 26-88AS.

Switch Closure

In this configuration, the minimum switch closed time is 5ms, the minimum switch open time is 6ms and the maximum bounce time is 1ms open without being counted.

The two pulse count input channels each have 8-bit counters. Input frequencies greater than 2000Hz (the limit of the 8-bit counter; 255 counts at the reset interval of 0.125 second) can be measured 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.

Every 0.125 seconds, the CR10 processor transfers the values from the 8-bit pulse counters into 16-bit accumulators (max count is 65535) and the 8-bit 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.

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, the Pulse Count instruction must 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 (see Section 1) or if you reset 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 discarded. If pulse counts are being totalised, 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 depends on the sampling interval (e.g. speed or 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 is 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 parameter parameter 4 (see Table 9-2).

NOTE All pulse count instructions must be kept in the same table. If the pulse count instruction is contained in a subroutine, that subroutine must be called from Table 2.

NOTE To prevent the storage of spurious pulse counts a table containing an Instruction 3 will skip execution at the first execution interval after the program is compiled. If Instruction 3 is in table 2 then the first execution of table 1 is also skipped.

Table 9-2 Pulse Count Configuration Codes

Code	Configuration
0	High frequency pulse
1	Low level AC
2	Switch closure
3	High frequency pulse, 16-bit counter
4	Low level AC, 16-bit counter
1X	Long interval data discarded
2X	Long interval data discarded, frequency (Hz) output

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Channel number for first measurement
03:	2	Configuration code (from Table 9-2)
04:	4	Input location for first measurement
05:	FP	Multiplier
06:	FP	Offset

Input locations altered: 1 per measurement

Intermediate storage locations altered: 1 for each repetition

Instruction 4: Excite, Delay and Measure

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 (i.e. 1x, where x is the initial excitation channel number) 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	Input 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 per measurement

Instruction 5: AC Half Bridge

This instruction is used to apply an excitation voltage to a half bridge (see Section 13), 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 (i.e. 1x, where x is the initial excitation channel number) causes the channel to be incremented with each repetition.

The excitation is turned on for exactly the same time for each polarity to ensure that ionic sensors do not polarise with repetitive measurements. The range should be selected to be a fast measurement (ranges 11-15), limiting the excitation on-time to less than 800µs at each polarity. A slow integration time should not be used with ionic sensors because of polarisation error.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range Code (Table 9-1)
03:	2	Input Channel
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 per measurement

Instruction 6: Full Bridge with Single Differential Measurement

This instruction is used to apply an excitation voltage to a full bridge (see Section 13) 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. The result is 1000 times the ratio of the measurement to the excitation voltage. A 1 before the excitation channel number (i.e. 1x, where x is the initial excitation channel number) 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	Input 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 per measurement

Instruction 7: Three-Wire Bridge with Single Differential Measurement

This instruction is used to determine the ratio of the sensor resistance to a known resistance using a separate 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) 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 (i.e. 1x, where x is the initial excitation channel number) 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 input 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 per measurement

Instruction 8: Differential Voltage with Excitation and Delay

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 (see Section 13). 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 a delay of 0 the resolution and common mode rejection of Instruction 8 are not as good as the ratiometric bridge measurement instructions. It does, however, provide a very rapid means of making bridge measurements. This instruction does not reverse excitation. A 1 before the excitation channel number (i.e. 1x, where x is the initial excitation channel number) 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	Input channel number for first measurement
04:	2	Excitation channel number
05:	4	Delay (multiples of 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 per measurement

Instruction 9: Full Bridge with Excitation Compensation

This instruction is used to apply an excitation voltage and make two differential voltage measurements. The measurements are made with the excitation voltage both positive and negative. 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 2.5V 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 (i.e. 1x, where x is the initial excitation channel number) causes the channel to be incremented with each repetition.

When used as a 6-wire full bridge (see Section 13), 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 2.5V range, the output is usually 1000 V_2/V_1 or millivolts output per volt of excitation.

When Instruction 9 is used to measure the output of 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, V_1 (Table 9-1)
03:	2	Range code for V_2
04:	2	Input 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 per measurement

Instruction 10: Battery Voltage

This instruction reads the CR10 battery voltage and writes it to an input location. The units for battery voltage are volts. When the batteries are around 8V, false battery readings of 9 to 10V will result, and the quiescent current drain increases to 7mA. At 9.2 to 9.3V, false analogue measurements are possible (Example: 2000mV input is measured as 2010 to 2050mV).

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location

Input locations altered: 1

Instruction 11: 107 Thermistor Probe

This instruction applies a 2V AC excitation voltage to a Campbell Scientific 107 Thermistor Probe, makes a fast, single-ended voltage measurement across a resistor in series with the thermistor and calculates the temperature in °C with a fifth order polynomial. A 1 before the excitation channel number (i.e. 1x, where x is the initial excitation channel number) causes the channel to be incremented with each repetition.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Input channel number of first measurement
03:	2	Excitation channel number
04:	4	Input location for first measurement
05:	FP	Multiplier
06:	FP	Offset

Input locations altered: 1 for each repetition

Instruction 12: 207 Relative Humidity Probe

This instruction applies a 1.5V AC excitation to a Campbell Scientific 207 Temperature and RH Probe, makes a fast single-ended measurement across a series resistor, calculates the result with a fifth order polynomial, and performs the required temperature compensation before outputting the result in % RH.

When measuring the output of several probes, all the RH elements should be connected sequentially. The temperature values used to correct the RH measurements should also be stored sequentially to make use of the repetitions parameter in Instruction 11.

NOTE The temperature value used in compensating the RH value (parameter 5) must be obtained (see Instruction 11) before 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. *Never excite the 207 probe with DC excitation* as the RH chip will be damaged. A 1 before the excitation channel number (i.e. 1x, where x is the initial excitation channel number) causes the channel to be incremented with each repetition.

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 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 for each RH measurement

Instruction 13: Thermocouple Temperature, Single-Ended

This instruction first uses the selected thermocouple calibration to calculate the thermocouple output voltage at the reference temperature. It then makes a single-ended voltage measurement on the thermocouple, adds the measured voltage to the calculated reference voltage and converts the result to temperature in °C.

Table 9-3 Thermocouple Type Codes

Code	Thermocouple Type
1	T (copper - constantan)
2	E (chromel - constantan)
3	K (chromel - alumel)
4	J (iron - constantan)

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code (Table 9-1)
03:	2	First TC channel
04:	2	Thermocouple type code
05:	4	Reference temperature location
06:	4	Destination input location
07:	FP	Multiplier
08:	FP	Offset

Input locations altered: 1 for each thermocouple channel

Instruction 14: Thermocouple Temperature, Differential Measurement

This instruction calculates the thermocouple temperature for the thermocouple type selected. The instruction makes a differential voltage measurement on the thermocouple, adds the measured voltage to the voltage calculated for the reference temperature relative to 0°C, and converts the combined voltage to temperature in °C. The differential inputs are briefly shorted to ground before making the voltage measurement to ensure that they are within the common mode range. Table 9-3 gives the thermocouple type codes for parameter 4. Entering a 9 in front of the type code causes the CR10 to make an additional check on common mode range; -99999 is output for temperature if the common mode range is exceeded.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Range code (Table 9-1)
03:	2	First TC channel
04:	2	Thermocouple type code
05:	4	Reference temperature location
06:	4	Destination input location
07:	FP	Multiplier
08:	FP	Offset

Input locations altered: 1 for each thermocouple channel

Instruction 16: Temperature From Platinum RTD

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 pending International Electrotechnical Commission standard. The range of linearisation is -200°C to 850°C. The error in the linearisation is less than 0.001°C between -100°C and +300°C, and is less than 0.003°C between -180°C and +830°C. The error ($T_{calculated}$ - $T_{standard}$) is +0.006°C at -200°C and -0.006°C at +850°C. The input must be the ratio R_S/R_0, where R_S is the RTD resistance and R_0 the resistance of the RTD at 0°C (see examples in Section 7).

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	4	Input location of R_S/R_0
03:	4	Input location of result
04:	FP	Multiplier
05:	FP	Offset

Input locations altered: 1 for each RTD

Instruction 17: Internal Temperature

This instruction measures the temperature (°C) of a thermistor on the CR10 analogue board inside the CR10 module.

NOTE This temperature is generally not adequate for use as a thermocouple reference temperature.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location number for temperature

Input locations altered: 1

Instruction 18: Move Time to Input Location

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 modulo divide, results 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)

PARAM.DATA NUMBER	TYPE	DESCRIPTION
01:	2	Time Code
02:	4	Number to modulo divide by
03:	4	Input location number

Input locations altered: 1

Instruction 19: Move Signature into Input Location

This instruction stores the signature of the CR10 PROM and user program memory (RAM) in an input location. This signature is a result of the contents of the PROM, the size of RAM, and the entries in the *1, *2, *3, *A and *C Modes. This signature is not the same as any of the signatures given in the *B Mode. Recording the signature allows detection of any program change or PROM failure.

PARAM.DATA NUMBER	TYPE	DESCRIPTION
01:	4	Input location number

Input locations altered: 1

Instruction 20: Port Set

This instruction sets or configures specified control ports (C1-C8). On power-up, the ports default to input configuration (i.e. they are not driven high or low by the CR10, and can be used to read the status of an external signal using Instruction 25). When a port is set high or low, pulsed, or toggled by this instruction or a program control command, it is automatically configured as an output.

CAUTION Voltages in excess of 5.5V applied to a control port can cause the CR10 to malfunction.

Ports can also be set using the *6 Mode or the J and K telecommunications commands. However, the ports *must* be configured as outputs before these means of setting them will work. The option to configure the port as an output is used when a port must be configured as an output without changing the state of the port.

Another option that can be selected for a port is the duration of a pulse initiated by a program control instruction giving the pulse command (see Table 12-2).

NOTE Instruction 20 does *not* pulse the port, it only sets the duration. If Instruction 20 is not used to set this duration, the pulse command results in a 10ms pulse.

Instruction 20 has two 4-digit parameters. Each digit represents one control port. The code (0-9) entered as the digit determines what effect command 20 has on the corresponding port.

Table 9-4 Port Configuration Option Codes

Code	Function
0	Set port low
1	Set port high
2	Toggle port
3	Pulse duration 1ms
4	Pulse duration 10ms
5	Pulse duration 100ms
6	Pulse duration 1s
7	Configure as output
8	Configure as input
9	Leave unchanged

('Pulse duration' refers to the duration of the pulse resulting from a subsequent 'pulse port' command in a Program Control Instruction.)

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	C8, C7, C6, C5 option codes
02:	4	C4, C3, C2, C1 option codes

Input locations altered: 0

Instruction 22: Excitation with Delay

This instruction is used in conjunction with others for measuring a response to a timed excitation using the switched analogue 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 allowing the CR10 to execute the next instruction. The analogue circuitry is not powered up during the delay after excitation. This reduces the current consumption of the CR10 to about 3mA.

If the only requirement is the delay of program execution, the off-time delay should be used, with the excitation on-time set to zero.

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:	FP	Excitation voltage in millivolts

Input locations altered: 0

Instruction 23: Burst Measurement

Instruction 23 makes repeated voltage measurements on a series of single-ended or differential channels, applying excitation if desired. The measurement units are millivolts before scaling. The measurements saved can be those made immediately on execution of the instruction or grouped around a specified trigger condition. The results of the measurements can be stored in Input Storage or the raw A/D data can be transmitted from the CR10's I/O port. The minimum sample time per channel is 1.333ms (i.e. one channel can be sampled at a maximum rate of 750Hz).

Measurement

The voltage measurement must use the fast integration time of 250µs (25µs on the 5V range). Differential measurements are made with a single integration. The noise level on a fast single-ended measurement or a differential measurement with one integration is 3µV RMS. Thus, only the 25mV and greater voltage ranges are practical (range codes 13-15).

Excitation is always supplied from excitation channel 1. The excitation voltage in millivolts is entered in parameter 9. If excitation is not desired, enter 0 for parameter 9.

Three options are available for the first digit in parameter 4:

0 - Trigger on first measurement channel.
1 - Trigger on control port 1.
2 - Trigger on first measurement channel, set control port high when trigger is met and low when measurements have finished.

When triggering on options 0 or 2, the measurement on the first specified channel (parameter 3) is compared to the limit specified in parameter 8. The multiplier and offset you specify are *not* applied before the comparison: the limit must be entered in units of millivolts. If a digital trigger (where logic low is less than 1.5V and logic high is between 3.5V and 5V) is used, it must be connected to control port 1. Option 2 is useful when two or more CR10s are required to start burst measurements at the same time. For example, control port 1 of all the CR10s involved are connected in parallel. The master CR10 sets its control port 1 high when the trigger condition is met. The remaining dataloggers, program-med with option 1, trigger on the digital signal. The master CR10 sets control port 1 low when its measurements are completed.

Five trigger options are available for the second digit in parameter 4:

0 - Trigger immediately, start saving or sending data with the first measurement made.
1 - Trigger if the measurement is greater than the limit entered or if the digital trigger is high.
2 - Trigger if the measurement is less than the limit or if the digital trigger is low.
3 - Trigger on rising edge; i.e. when the measurement goes from below to above the limit or when the digital trigger goes from low to high.
4 - Trigger on falling edge; i.e. trigger when measurement goes from above the limit to below it or when the digital trigger goes from high to low.

When triggering on the rising or falling edge, the input must make the specified transition to trigger. For example, when triggering on the rising edge, if the input starts out high, it must go low and then high again to satisfy the trigger condition.

Data Sent to Input Storage

When the measurements are sent to Input Storage, parameter 6 is used to specify the number of scans made on the channels being measured (the CR10 multiplies the number entered by 1000). The measurements from each channel are stored contiguously. For example: parameter 1 specifies that four channels are to be measured, parameter 6 specifies 250 scans (0.250 entered), and parameter 10 specifies 1 as the first input location in which to store data. The measurements from the first channel are stored in input locations 1-250, those from the second channel in locations 251-500, and so on. If insufficient locations are allocated to Input Storage (using the *A Mode) to accommodate the number of locations called for by parameter 6 multiplied by parameter 1 (e.g. 250 x 4 = 1000), an error code, E60, is displayed when the program is compiled.

The number of scans determines how many samples are saved, This in turn determines when Instruction 23 is completed to allow execution to pass on to the next instruction. Measurements before and/or after the trigger can be saved. Parameter 7 determines how many scans that occurred before the trigger are saved. For example, if 250 scans are specified, and an offset of 20 is entered for parameter 7, then the trigger measurement is stored in Location 21. Locations 1-20 contain the measurements that preceded the trigger and locations 22-250 contain the measurements following the trigger. If only 10 measurements are made before the trigger, then they are stored in locations 11-20 while the value -99999. is stored in locations 1-10 (for which no measurements were made).

Data Sent to Serial Port

When the raw A/D data is transmitted from the CR10's I/O port, the measurement data is not buffered and hence only the trigger and subsequent measurements can be sent. The number of measurements is determined by parameters 1 and 6. Because the total number of measurements is limited only by the storage of the receiving device, this can be a very large number. Parameter 6 is the number of measurements per channel to send in units of 1000 (e.g. 0.001 represents 1 measurement). If 0 is entered for parameter 6, the CR10 sends data until the instruction is aborted by pressing any key on the CR10KD Keyboard/Display.

Raw A/D data can be sent to a maximum of eight SM192 or SM716 Storage Modules (SMs) at 76800 baud (parameter 4: Destination Option C = 3). All SMs connected should be set to 'fill and stop' and have consecutive addresses. The Burst Mode data is sent to the first available (lowest addressed) Storage Module, followed by the next lowest addressed, and so on. If the data is retrieved from the SMs using SMCOM, a Storage Module communications program contained in the PC208 Datalogger Support Software, the data collection format must be 'as stored 8-bit'. This will transfer the raw A/D data to a computer file. The raw A/D data can be converted to ASCII using Split, a general purpose data reduction program also contained in PC208.

NOTE Burst Mode data can also be sent to a single CSM1 Card Storage Module and the data recovered using the CSMCOM program.

If Split is not available for converting the raw A/D data, the following A/D format information is provided for decoding purposes: at the start of the series of measurements, the CR10 makes a self-calibration measurement. The calibration

data is sent at the start of the measurement data. The serial data is sent as a series of signed 2-byte integers (most significant byte sent first; i.e. Integer = 256 * byte 1 + byte 2): $I_1 ... I_n$. The first integer, I_1, is a start of output identifier, FCxx (hex), where the first byte is always FC (never seen in the data), and the second byte is a number less than 100 (decimal, 64 hex), which is the Instruction Location Number of Instruction 23 in the program table. I_2 divided by I_3 is the multiplier and I_4 the offset (to the raw data) determined by the first calibration. I_2 is a fixed value determined by the input range selected. I_5 to I_n are the raw measurement values. Thus, the value of the first measurement sent (M_1) in millivolts is:

$$M_1 = I_2/I_3 (I_5 - I_4)$$

The measurement data is sent in the order that the measurements are made (i.e. the first measurement for each channel, then the second measurement for each channel, etc.).

NOTE When the raw serial data option is selected, the calibration values are for conversion to millivolts only. Parameters 11 and 12 are ignored.

Scan Interval

Instruction 23 has its own scan interval which is independent of the execution interval of the program table. The resolution of the clock timing the execution interval is 813 nanoseconds. This scan interval, entered in parameter 5 (in milli-seconds), is the time between each scan of the specified channels (i.e. if three channels are specified in parameter 1, and the scan interval is 5ms, then the three measurements are repeated every 5ms). The minimum time that is allowed per measurement is 1.333ms. The maximum time that is allowed per measurement is 50ms. If the scan interval entered does not allow this much time per measure-ment (e.g. if with four repetitions, an interval less than 5.332ms is entered), an error code, E61, is displayed when the program is compiled. When sending data to the I/O port, the rate at which the data can be transferred may limit the scan interval (e.g. at 9600 baud the minimum time per measurement is 2.2ms).

Burst/Telecommunications Considerations

If a Burst Mode measurement sequence is in progress, raising the datalogger's ring line aborts the Burst Mode sequence. Peripherals which raise the ring line are modems (i.e. RF, Telephone, Short Haul, MD9, SC32A) or the CR10KD.

If the Burst Measurement instruction is encountered while telecommunications is in progress, the destination of the data determines whether or not the instruction is executed:

Burst data sent to input locations – If a CR10 already in Telecommunications Mode executes a Burst Measurement instruction specifying that Burst Mode data be sent to input locations, all telecommunication activity is suspended. After the Burst Measurement trigger condition is met and all the measurements have been made, telecommunications activity can resume.

Burst data sent to Serial I/O Port – If the Burst Measurement instruction specifies that Burst data be sent to the I/O port (i.e. to a Storage Module), CR10 program execution pauses until the Telecommunications Mode is exited. During this pause telecommunications (i.e. view input locations, Monitor Mode with GraphTerm, etc.) can continue. No Burst measurements are made while in telecommunications and no Burst data is sent to the serial port. After telecom-

munications has ended, datalogger program execution resumes as if the Burst Measurement instruction had just been executed.

NOTE Instruction 23 can be aborted by pressing any key on the CR10KD Keyboard/Display.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	No. of channels per scan
02:	2	Range code (13-15)
03:	2	First channel for analogue measurements
04:	4	Option, 4-digit code: ABCD
	A	Trigger: 0 - Trigger on first analogue channel 1 - Digital trigger on control port 1 2 - Same as option 0, but set control port 1 high when trigger occurs, low when done measuring
	B	Trigger option: 0 - Trigger immediately 1 - Trigger if above limit (high) 2 - Trigger if below limit (low) 3 - Trigger on rising edge 4 - Trigger on falling edge
	C	Destination: 0 - Input Storage 1 - Serial port 9600 baud 2 - Serial port 76800 baud 3 - Serial port 76800 baud to SM192/716/CSM1
	D	Measurement: 0 - Differential measurement 1 - Single-ended measurement
05:	FP	Scan interval (ms, minimum 1.333 x reps, limited to 1.333-50ms)
06:	FP	Number of scans (units of 1000)
07:	4	Number of samples saved before trigger (not used with serial output)
08:	FP	Trigger limit (mV, unscaled measurement)
09:	4	Excitation voltage (mV)
10:	4	First input location in which to store data
11:	FP	Multiplier (not used with serial output)
12:	FP	Offset (not used with serial output)

Instruction 24: Calibration

This instruction writes the 19 CR10 calibration values into input locations. If C (--) is pressed before entering the input location, then the values are the results of the last automatic calibration. Otherwise, the calibration takes place only when Instruction 24 is executed; automatic calibration is disabled. (See also Section 13.)

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location number; press C (--) for results of automatic calibration

Input locations altered: 19

Instruction 25: Port Read

Instruction 25 reads the status of a group of ports selected by a mask and places the result in an input location. The status is a base 2 representation of the ports converted to base 10. Port 1 is the least significant bit. For example, if all ports are read, and the port status is as follows:

PORT	C8	C7	C6	C5	C4	C3	C2	C1
VALUE	128	64	32	16	8	4	2	1
STATUS (0=low, 1=high)	0	0	1	1	0	0	1	0

Base 10 equivalent: 32 + 16 + 2 = 50; 50 will be stored in the input location.

The mask is also in 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. For example, if 50 (see above example) is entered for the mask, ports 2, 5, and 6 are read. If only ports 4 and 5 are high, the status will be 16 (port 4 is not read).

CAUTION Voltages in excess of 5.5V applied to a control port can cause the CR10 to malfunction.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Mask (0-255)
02:	4	Input location to store result

Input locations altered: 1

Instruction 26: Timer

This instruction resets a timer or stores 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.

When program tables are changed and compiled with the *0 Mode, the timer is reset automatically.

Entering '***6**' after changing the program compiles the programs, but does *not* reset the timer.

PARAM. NUMBER	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)

Instruction 27: Period Measurement

Instruction 27 measures the period (in 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 specified number of cycles is timed with a resolution of 60 nanoseconds, making the resolution of the period 60 nanoseconds divided by the number of cycles measured. The 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 waits during each repetition for the specified number of cycles. If the cycles have not been counted within this time, -99999 is loaded into the input location.

Input Frequency Gain 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	200 kHz

0x Output period in microseconds
1x Output frequency in kHz where x is range code

* AC voltage; must be centred around CR10 ground.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	2	Gain/output option
03:	2	Single-Ended Input Channel
04:	4	No. of cycles to measure
05:	4	Time out (in multiples of 0.01 sec, at least the maximum duration of the no. of cycles specified + 1.5 cycles.)
06:	4	Destination input location
07:	FP	Multiplier
08:	FP	Offset

Input locations altered: 1* Repetitions

Instruction 28: Vibrating Wire Measurement

This instruction excites a vibrating wire sensor with a swept frequency (from low to high), then measures the period of the response and calculates $1/T^2$, where T is the period in milliseconds. The excitation is normally provided for each repetition. As an option, a single excitation can be made before 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 (Press C (--) to skip repeat of excitation)
02:	2	Input channel (Single-ended)
03:	2	Excitation Channel
04:	2	Starting frequency of sweep (Hz x 100)
05:	2	End frequency of sweep (Hz x 100)
06:	4	No. of cycles to measure (0 means none)
07:	4	Delay before excitation applied (multiples of 10ms)
08:	4	Input location ($1/T^2$), T in ms
09:	FP	Multiplier
10:	FP	Offset

Instruction 101: SDM-INT8

The 8-channel Interval Timer (SDM-INT8) is a measurement module which sends processed timing information to the datalogger. Each of the eight input channels can be independently configured to detect either rising or falling edges of either a low level AC signal or a 5V logic signal. Each channel can be pro-grammed independently . See the SDM-INT8 manual for detailed instructions and examples.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Address of INT8
02:	4	*Input configuration; channels 8,7,6,5
03:	4	*Input configuration; channels 4,3,2,1
04:	4	**Function; channels 8,7,6,5
05:	4	**Function; channels 4,3,2,1
06:	4	***Averaging option
07:	4	Loc
08:	FP	Mult
09:	FP	Offset

Execution Time: 2.3ms + 1.65ms/value +averaging interval if used (See SDM-INT8 manual for possible extra processing time for higher frequency signals.)

Intermediate Storage: 1 location

* Input configurations:
0 = 5V logic level, rising edge
1 = 5V logic level, falling edge
2 = low level AC, rising edge
3 = low level AC, falling edge

** Function:
0 = none
1 = period in ms
2 = frequency in kHz
3 = time since previous channel's edge in ms
4 = time since channel 1 in ms
5 = counts on channel 2 since channel 1, linear interpolation
6 = frequency in kHz (low resolution)
7 = counts
8 = counts on channel 2 since channel 1, no interpolation

*** Averaging option:

0	Average over execution interval
0-	Continuous averaging
XXXX	Averaging interval in ms, XXXX>0
XXXX-	Capture all events until XXXX edges of channel 1 (0<XXXX<9999)
9999-	Test memory

Instruction 102: SDM-SW8A

The 8-channel SDM-SW8A Switch Closure Input Module is a peripheral for measuring up to eight channels of switch closure or voltage pulse inputs. Each channel can be configured to read single-pole double-throw (SPDT) switch closure, single-pole single-throw (SPST) switch closure or voltage pulse. Output options include counts, duty cycle and state.

The SW8A is addressed by the datalogger, allowing multiple SW8As to be connected to one datalogger. 16 addresses are available.

If more channels are requested than exist in one module, the datalogger automatically increments the address and continues to the next SW8A. The address settings for multiple SW8As must increase sequentially. For example, assume that two SW8As with addresses of 22 and 23 are connected, and 12 repetitions are requested. Eight channels from the first SW8A and the first four channels from the next will be read.

Only one Function Option (parameter 3) may be specified per Instruction102 . If all four functions are desired, four separate Instruction 102s are required.

Function Option 0 provides the state of the signal at the time Instruction 102 is executed. A 1 or 0 corresponds to high or low states, respectively.

Function Option 1 provides signal duty cycle. The result is the percentage of time the signal is high during the sample interval.

Function Option 2 provides a count of the number of positive transitions of the signal.

Function Option 3 provides the signature of the SW8A PROM. A positive number (signature) indicates the PROM and RAM are good, a zero (0) indicates bad PROM, and a negative number indicates bad RAM. Function Option 3 is not used routinely, but is helpful in debugging. Only one repetition is required for Option 3.

Parameter 4 specifies the first SW8A channel to be read (1..8). One or more sequential channels are read depending on the repetitions. To optimise program efficiency, the sensors should be wired sequentially.

Data is stored in sequential datalogger input locations, starting at the location specified in parameter 5. The number of input locations consumed is equal to the number of repetitions.

The scaling multiplier and offset (parameters 6 and 7) are applied to all readings. If a multiplier is not entered, all readings are set to 0.

If the SW8A does not respond, -99999 is loaded into the input locations. Modules which do not respond when addressed by the datalogger are possibly wired or addressed incorrectly. Verify that the address specified in parameter 2

corresponds to the jumper setting and that all connections are correct and secure. See the SDM-SW8A manual for examples.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Number of Channels
02:	2	Module Address (Base 4: 00..33)
03:	2	Function Option (0=State, 1=Duty 2=Counts, 3=Signature)
04:	2	SDM-SW8A Starting Channel (1..8)
05:	4	Starting input location
06:	FP	Mult
07:	FP	Offset

Instruction 103: SDM-AO4

Instruction 103 is used to activate an SDM-AO4 connected to ports C1, C2 and C3.

There are four analogue voltage outputs per SDM-AO4. The output voltages in millivolts must be stored in four adjacent input locations starting with the location entered in parameter 3. Four repetitions are required for each SDM-AO4. For every four repetitions another device at the next higher address is selected.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01	2	Reps (no. of outputs)
02	2	Address (Base 4)
03	4	Starting input location

C1 is Data line
C2 is Clk/Handshake line
C3 is SDE (Enable) line

Instruction 104: SDM-CD16

The SDM-CD16 Control Port Expansion Module has 16 digital control ports with drivers. Each port can be controlled by a datalogger or controlled manually with an override toggle switch. Each port can be thought of as a switch to ground which is closed when active and open when inactive. The primary function is to activate DC-powered external relays, solenoids or resistive loads under datalogger control.

The SDM-CD16 is a synchronously addressed datalogger peripheral. Datalogger control ports 1, 2 and 3 are used to address the SDM-CD16 then clock out the desired state of each of the 16 control ports. Up to 16 SDM-CD16s may be addressed, making it possible to control a maximum of 256 ports from the first three datalogger control ports.

For each repetition, the states of the 16 ports of the addressed SDM-CD16 are sent according to 16 sequential input locations starting at the input location specified in parameter 3. Any non-zero value stored in an input location activates (connects to ground) the associated SDM-CD16 port. A value of zero (0) deactivates the port (open circuit). For example, assuming two repetitions and a starting input

location of 33, outputs 1 to 16 of the first SDM-CD16 are set according to input locations 33 to 48, and outputs 1 to 16 of the second SDM-CD16 are set according to input locations 49 to 64. See the SDM-CD16 manual for detailed instructions and examples.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
1	2	Reps (no. of CD16 modules sequentially addressed)
2	2	Starting Address (base 4: 00..33)
3	4	Starting input location

Section 10. Processing Instructions

To help you when programming, the parameter descriptions given in this section are related 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)

Instruction 30: Load Fixed Data

This instruction stores a fixed value into an input location. The value is entered in scientific notation; the absolute value of the number can range from 1×10^{-19} to 9×10^{18}. A value smaller than the minimum is set to 0, while a value larger than the maximum is set to the maximum.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	FP	Mantissa	[F]
02:	2	Exponent of 10 (Press C to change sign)	
03:	4	Destination input location	[Z]

Input locations altered: 1

Instruction 31: Move Input Data

This instruction moves data from one input location to another.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location no. of source	[X]
02:	4	Destination input location	[Z]

Input locations altered: 1

Instruction 32: Increment Input Location

This instruction adds 1 to the current value in the specified input location.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Destination input location	[Z]

Input locations altered: 1

Instruction 33: X + Y

This instruction adds X to Y and places the result in a third input location.

PAR. No.	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

Instruction 34: X + F

This instruction adds F to X (where F is a fixed floating point number) and places the result in an input location.

PAR. No.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	FP	Fixed number	[F]
03:	4	Dest. input location of X+F	[Z]

Input locations altered: 1

Instruction 35: X - Y

This instruction subtracts Y from X and places the result in an input location.

PAR. NO.	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

Instruction 36: X * Y

This instruction multiplies X by Y and places the result in an input location.

PAR. NO.	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

Instruction 37: X * F

This instruction multiplies X by F (where F is a fixed multiplier) and places the result in an input location.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	FP	Fixed multiplier	[F]
03:	4	Dest. input location for X*F	[Z]

Input locations altered: 1

Instruction 38: X / Y

This instruction divides X by Y and places the result in an input location. Division by 0 causes the result to be set to 99999.

PAR. NO.	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

*Instruction 39: Square Root

This instruction takes the square root of X and places the result in an input location. If X is negative, 0 is stored as the result.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Dest. input location for $X^{1/2}$	[Z]

Input locations altered: 1

Instruction 40: LN(X)

This instruction takes the natural logarithm of X and places the result in an input location. If X is 0 or negative, -99999 is stored as the result.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Dest. input location for LN(X)	[Z]

Input locations altered: 1

Instruction 41: EXP(X)

This instruction raises the exponential (EXP) base e to the power of X and places the result in an input location.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Dest. input location for EXP(X)	[Z]

Input locations altered: 1

Instruction 42: 1/X

This instruction takes the reciprocal of X and places the result in an input location. If X=0, 99999 is stored as the result.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Dest. input location for 1/X	[Z]

Input locations altered: 1

Instruction 43: ABS(X)

This instruction takes the absolute (ABS) value of X (i.e. the value without any sign) and places the result in an input location.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Dest. input location for ABS(X)	[Z]

Input locations altered: 1

Instruction 44: Fractional Value

This instruction takes the fractional (FRAC) value (i.e. the non-integer portion) of X and places the result in an input location.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Dest. input loc. for FRAC(X)	[Z]

Input locations altered: 1

Instruction 45: Integer Value

This instruction takes the integer (INT) portion of X and places the result in an input location.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Dest. input location for INT(X)	[Z]

Input locations altered: 1

Instruction 46: X Mod F

This instruction does a modulo divide of X by F and places 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.

PAR. NO.	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

Instruction 47: X^Y

This instruction raises X to the power of Y and places the result in an input location.

PAR. NO.	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

Instruction 48: Sin(X)

This instruction calculates the sine of X (X is assumed to be in degrees) and places the result in an input location. The cosine of a number can be obtained by adding 90 to the number and taking the sine (COS(X) = SIN (X + 90)).

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of X	[X]
02:	4	Dest. input location for SIN(X)	[Z]

Input locations altered: 1

Instruction 49: Spatial Maximum

This instruction finds the spatial maximum (SPA MAX) value of the given set or *swath* of input locations and places 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 3. 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 Instruction 49 with Instruction 31 (Move Data). In Instruction 31, enter the location in which Instruction 49 stores its result as the source (fixed) and enter the destination as an indexed location.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Swath	[SWATH]
02:	4	Starting input location	[1ST LOC]
03:	4	Dest. input location for maximum	[Z]

Input locations altered: 1 or 2

Instruction 50: Spatial Minimum

This instruction finds the spatial minimum (SPA MIN) value of the given set or *swath* of input locations and places 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.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Swath	[SWATH]
02:	4	Starting input location	[1ST LOC]
03:	4	Destination input location for minimum	[MIN or Z]

Input locations altered: 1 or 2

Instruction 51: Spatial Average

This instruction takes the spatial average (SPA AVG) over the given set or *swath* of input locations and places the result in an input location.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Swath	[SWATH]
02:	4	Starting input location	[1ST LOC]
03:	4	Destination input location of average	[AVG or Z]

Input locations altered: 1

Instruction 53: Scaling Array with Multiplier and Offset

This instruction takes 4 input location values, multiplies each by a floating point constant, then adds another floating point constant to the resulting products and places the final results back into each of the original four input locations.

PAR. NO.	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

Instruction 54: Block Move

This instruction 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 two and a 'destination step' of 1 will move data from every other input location to a contiguous block of input locations.

PAR. NO.	DATA TYPE	DESCRIPTION
01:	4	Number of values to move
02:	4	First source location
03:	2	Step of source
04:	4	First destination location
05:	2	Step of destination

Intermediate storage: 0

*** Instruction 55: Fifth Order Polynomial

This instruction evaluates a fifth order polynomial of the form.

$$F(X)=C0+C1X+C2X^2+C3X^3+C4X^4+C5X^5$$

where C0 to C5 are the coefficients for the argument X raised to the zero to fifth power, respectively. The magnitude of the user-entered coefficient is limited to a range of ±0.00001 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 in order to increase the number of significant digits.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	2	Repetitions	[REPS]
02:	4	Starting input location for X	[X]
03:	4	Destination 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* Reps

Instruction 56: Saturation Vapour Pressure

This instruction calculates saturation vapour pressure over water (SVP_W) in kilopascals from the air temperature (°C) and places it in an input location. The algorithm for obtaining SVP_W from air temperature (°C) is taken from: Lowe, Paul R., 1977: *An approximating polynomial for computation of saturation vapour pressure.* J. Appl. Meteor 16, 100-103.

Saturation vapour pressure over ice (SVP_i) in kilopascals for a -50°C to 0°C range can be obtained using Instruction 55 and the relationship:

$$SVP_i = -.00486 + .85471\ X + .2441\ X^2$$

where X is the SVP_W derived by Instruction 56. This relationship was derived by Campbell Scientific from the equations for the SVP_W and the SVP_i given in Lowe's paper.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of air temperature °C	[TEMP.]
02:	4	Destination input location for saturated vapour pressure	[VP or Z]

Input locations altered: 1

Instruction 57: Vapour Pressure from Wet- /Dry-Bulb Temperatures

This instruction calculates vapour pressure in kilopascals from wet and dry-bulb temperatures in °C. The algorithm is of the type used by the US National Weather Service:

$VP = VP_W - A(1 + B*T_W)(T_a - T_W)\ P$

VP = ambient vapour pressure in kilopascals
VP_W = saturation vapour pressure at the wet-bulb temperature in kilopascals
T_W = wet-bulb temperature, °C
T_a = ambient air temperature, °C
P = air pressure in kilopascals
A = 0.000660
B = 0.00115

Although the algorithm requires an air pressure entry, the daily fluctuations are small enough that for most applications a fixed entry of the standard pressure at the site elevation will suffice. If a pressure sensor is employed, the current pressure can be used.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	4	Input location of atmospheric pressure in kilopascals	[PRESSURE]
02:	4	Input location of dry-bulb temp.	[DB TEMP.]
03:	4	Input location of wet-bulb temp.	[WB TEMP.]
04:	4	Destination input location for vapour pressure	[VP or Z]

Input locations altered: 1

Instruction 58: Low Pass Filter

This instruction applies a numerical approximation to an analogue resistor-capacitor (RC) low pass (LP) filter using the following algorithm:

$$F(X_i) = W*X_i + F(X_{i-1}) * (1-W)$$

Where, X = input sample
W = user-entered weighting function, $O< W <1$
$F(X_{i-1})$ = output calculated for previous sample

If W=O, $F(X_i)=F(X_1)$; if W=1, $F(X_i)=X_i$

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 'analogue' estimate of the cut-off frequency becomes less representative.

On the first execution after compiling, F(X) is set equal to X.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	2	Repetitions	[REPS]
02:	4	First input location for input data	[X]
03:	4	Destination input location for first filtered result	[F(X) or Z]
04:	FP	Weighting function, W	[W]

Input locations altered: 1 for each repetition

Instruction 59: Bridge Transform

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 CR10 bridge measurement instructions (with appropriate multiplier and offset; see Section 13) and R_f represents the multiplier value. The result of Instruction 59 is stored in the same location that X was.

PAR. NO.	DATA TYPE	DESCRIPTION	
01:	2	Repetitions	[REPS]
02:	4	Starting input location and destination	[X]
03:	FP	Multiplier (R_f)	[MULT.]

Input locations altered: 1 for each repetition

Instruction 61: Indirect Indexed Move

This instruction 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.

PAR. NO.	DATA TYPE	DESCRIPTION
01:	4	Input location in which Source input location is stored
02:	4	Input location in which Destination input location is stored

Input locations altered: 1

Instruction 63: Parameter Extension

This instruction is used immediately following Instruction 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 eight two-digit parameters. Refer to the instruction being extended (97 or 98) for further details on the use of Instruction 63.

PAR. NO.	DATA TYPE	DESCRIPTION
01:-08:	2	Depends on the preceding instruction. **Following Instruction 97:** RF IDs and Phone No., one digit at a time '32' Between RF IDs '72' Between RF and DC112/Hayes Modem Phone No. '13' To end **Following Instruction 98** (255 character limit): Base 10 value of ASCII character (see Appendix E) '00' To end

Input locations altered: 0

Instruction 66: Arctan

This instruction calculates 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 $-90^{o} < ARCTAN < 90^{o}$).

Quadrant	Sign of X	Sign of Y
I	+	+
II	+	-
III	-	-
IV	-	+

PAR. NO.	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

Section 11. Output Processing Instructions

Instruction 69: Wind Vector

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, for example. The simplest solution is to enter an offset of -20 in the instruction monitoring the wind vane, which results in 0 to 360 degrees following the modulo divide.

When a wind speed sample is 0, the sample is not used in the computation of wind direction. At low wind speeds, when the wind direction readings may be erroneous, you may not want a sample less than the sensor threshold used in the calculation of mean unit vector direction, mean resultant wind direction or resultant wind speed. If this is the case, Instruction 89 can be used to check the wind speed, and if less than the threshold, Instruction 30 can set the wind speed location(s) equal to 0.

Standard deviation can be processed in one of two ways: 1) using every sample taken during the output period, or, 2) by averaging standard deviations processed from shorter sub-intervals of the output period. Averaging sub-interval standard deviations minimises the effects of meander under light wind conditions and provides more complete information for periods of transition[1].

The standard deviation of horizontal wind fluctuations from sub-intervals is calculated as follows:

$$\sigma(\Theta)=[(N_1(\sigma\Theta_1)^2+N_2(\sigma\Theta_2)^2 \ldots+N_m(\sigma\Theta_m)^2)/N]^{1/2}$$

where $\sigma(\Theta)$ is the standard deviation over the output interval, $\sigma\Theta_1 \ldots \sigma\Theta_m$ are sub-interval standard deviations, $N_1 \ldots N_m$ are the numbers of samples (with windspeed >0) in each sub-interval, and N is the total number of samples.

NOTE The algorithm treats periods of zero wind speed as periods of zero variation in wind direction.

[1] EPA On-site Meteorological Program Guidance for Regulatory Modeling Applications.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions
02:	4	Samples per sub-interval (number of scans)
03:	2	Sensor/Output (two 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) for each 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, which specify the values calculated.

Option 0:

Mean horizontal wind speed, **S**.
Unit vector mean wind direction, **Θ1**.
Standard deviation of wind direction, **σ(Θ1)**.

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, **Θ1**.

Option 2:

Mean horizontal wind speed, **S**.
Resultant mean wind speed, **U**.
Resultant mean wind direction, **Θu**.
Standard deviation of wind direction, **σ(Θu)**.

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

$$Ux=(\Sigma(Ue_i/U_i))/N$$

$$Uy=(\Sigma(Un_i/U_i))/N$$

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:

$$Ue=(\Sigma S_i \sin \Theta_i)/N$$

$$Un=(\Sigma S_i \cos \Theta_i)/N$$

or, in the case of orthogonal sensors:

$$Ue=(\Sigma Ue_i)/N$$

$$Un=(\Sigma Un_i)/N$$

Resultant mean wind direction, Θu:

Θu=Arctan (Ue/Un)

Standard deviation of wind direction, σ(Θu), using Campbell Scientific algorithm:

$\sigma(\Theta u)=81(1-U/S)^{1/2}$

Instruction 70: Sample

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 sample

Instruction 71: Average

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 for each input location

Instruction 72: Totalize

This instruction stores the totalised 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 for each input location

Instruction 73: Maximize

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 can be tested by Instruction 79.) The time of occurrance of the maximum value(s) is optional output information, which is formatted and activated by entering one of the following codes for parameter 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 for each input location (plus 1 or 2 with time of maximum)

Instruction 74: Minimize

Operating in the same manner as Instruction 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 or 2 with time of minimum)

Instruction 75: Standard and Weighted Value Histogram

This instruction processes input data as either a standard histogram (frequency distribution) or a weighted value histogram.

The *standard histogram* shows you the fraction of the output interval that the value in a specified input location (called the *bin select value*) fell within a particular sub-range of the total specified range.

Example

You are measuring a voltage which varies from 0 to 20mV. You choose to divide the total range into four sub-ranges or bins. The bins therefore cover these ranges:

Bin no.	Range
1	0 - 4.99mV
2	5 - 9.99mV
3	10 - 14.99mV
4	15 - 20mV

The output interval is one hour. During one particular hour, the voltage stays around 7mV for 30 minutes and around 13mV for 30 minutes, i.e. it spends 50% of the time in bin 2 and 50% in bin 3. The result of Instruction 75 in this case is therefore 0, 0.5, 0.5, 0.

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 two input locations. One location contains the bin select value (e.g. wind direction); the other contains the *weighted value* (e.g. wind speed). 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 average of the weighted value.

NOTE

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. you must also generate a standard histogram of the bin select value).

For either histogram, you 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 required (see below)
4. the bin select value input location
5. the weighted value input location (see below)
6. the lower range limit
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 is incremented with each repetition and the weighted value location remains the same (same weighted value sorted on the basis of different bin select values). The weighted value location *is* incremented if it is entered as an indexed location (press 'C' at some point while keying in parameter 5; two dashes, --, will appear on the right of the display).

The histogram can 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 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°C	<12°C
Range of last bin	28 to 29.99°C	>28°C

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

Instruction 77: Record Real Time

This instruction stores the current time in Final Storage. At midnight the time changes 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 '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.

The year is output as 19xx if xx is greater than 85, otherwise it will be output as 20xx. The CR10 will require a PROM update in the year 2085. If the 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 a 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).

Instruction 78: Set High or Low Resolution Data Storage Format

This instruction enables high resolution (5-character) or low resolution (4-character) final data storage format. Instruction 78 must be entered before 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

Instruction 79: Sample on Maximum or Minimum

Used in conjunction with Instructions 73 or 74, this instruction copies specified input location values into Intermediate Storage whenever a previous Maximize or Minimize instruction senses a new maximum or minimum value. When the Output Flag is set, the values copied to Intermediate Storage are transferred to Final Storage.

Instruction 79 looks for a flag that is set by Instruction 73 or 74 when a new maximum or minimum is detected. This flag is cleared at the start of a Maximize or Minimize instruction or at the beginning of the program table. If Instruction 73 or 74 has more than one repetition, there is no way for Instruction 79 to know which input location caused the maximum or minimum flag to be set. Thus, Instruction 79 should directly follow the maximum or minimum instruction to which it refers. If sampling is to occur only when one specific input location shows a new maximum or minimum value, the previous Maximize or Minimize instruction should only refer to that input location (i.e. there should only be one repetition).

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Repetitions (number of sequential locations to sample)
02:	4	Starting input location no.

Outputs Generated: 1 for each repetition

Instruction 80: Set Active Storage Area

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. At the beginning of each program table the active output area defaults to Final Storage Area 1.

When output is directed to Final Storage, the second parameter can be used to set the output array ID. In this case Instruction 80 should follow the instruction setting the Output Flag (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 is sent to Input Storage, no array ID is sent.

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)

Instruction 82: Standard Deviation in Time

This instruction calculates 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 for each repetition

Section 12. Program Control Instructions

Table 12-1 Flag Description

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

Table 12-2 Command Codes

0	Go to end of program table[3]
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
41-48	Set Port 1-8 high[2]
51-58	Set Port 1-8 low[2]
61-68	Toggle Port 1-8[2]
71-78	Pulse Port 1-8[2]

[1] 97 and 98 are special subroutines which can be called by control ports 7 and 8 going high; see Instruction 85 for details.

[2] The ports can be indexed to the loop counter.

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

Instruction 83: If Case X < F

If the value in the location specified in the Begin Case instruction (Instruction 93) is less than the fixed value entered as parameter 1 of Instruction 83 then the CR10 executes the command in parameter 2 and goes to the end of the case statement when the next Instruction 83 occurs. Otherwise, it continues to the next instruction. See Instruction 93 for an example.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	FP	Fixed value
02:	2	Command

Instruction 85: Label Subroutine

This instruction marks the start of a subroutine. A subroutine is 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 can be called from within other subroutines (nested). The maximum nesting level for subroutines is seven deep. Attempts to nest more than seven deep are not detected at compilation, but result in a run time error: when the seventh subroutine attempts to call the eighth, error 31 is displayed. Execution does not branch to the eighth subroutine, but continues with the instruction following that calling the subroutine.

Port Interrupt Subroutines (97 and 98)

If subroutine 97 or 98 is included in Table 3 then a rising edge on control port 7 (subroutine 97) or 8 (subroutine 98) causes an interrupt which calls the subroutine. These subroutines can also be called as normal from any table.

Special subroutines 97 and 98, initiated by a port going high (see above), can interrupt either Table 1 or 2, or can occur when neither Table is being executed. These subroutines *can* interrupt a table while the Output Flag (flag 0) is set. When the port activating subroutine 97 or 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 order is subroutine 98, subroutine 97, Table 1, Table 2. If subroutines 97 and 98 are both pending (ports go high at the same time or both go high during the execution of the same instruction in one of the tables), subroutine 98 will be executed first. Subroutines 97 and 98 cannot interrupt each other, and cannot be interrupted by a program table.

While subroutine 97 or 98 is being executed as a result of the appropriate port going high, that port interrupt is disabled (i.e. the subroutine must be completed before the port going high will have any effect).

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Subroutine number (1-9, 79-99)

Instruction 86: Do

This instruction unconditionally executes the specified command.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Command (Table 12-2)

Instruction 87: Loop

Instructions included between the Loop instruction and the End instruction (Instruction 95) are repeated the number of times specified by the iteration count (parameter 2), or until an Exit Loop command (command 31 or 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 are completed and execution passes 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 are executed and other portions of the table are not executed in the interim. When a loop with delay is executed, execution starts at the loop, skipping over any previous and following instructions in the table.

When a fixed number of iterations is executed, the time spent in the loop is equal to the product of the execution interval, the 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 CR10 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 is *not* initiated at each execution interval. (However, the overrun decimals are not displayed.) As a consequence of this, the Output Flag is not cleared automatically between passes through the loop. Also, because Table 2 cannot interrupt Table 1, Table 2 will not be executed while Table 1 is in a loop with delay. Note also that 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 this is not done, Table 1 will never be executed (i.e. will not interrupt Table 2) because it will detect that the Output Flag is still high in Table 2.

Input locations for Processing Instructions within a loop can be entered as indexed locations. Indexing causes the input location number 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 remain constant.

To index a location, press 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. The reason for this is that an Output Instruction within a loop is allocated 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 allocated 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.

If the Average Instruction with one repetition and location 1 indexed is placed within a loop of 10 and the Output Flag is set high before entering the loop, 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 innermost loop that they are within. The maximum nesting level in the CR10 is nine deep. This applies to 'If Then/Else' comparisons and Loops or any

combination thereof. An 'If Then/Else' comparison which uses the Else instruction (Instruction 94) counts as being nested two deep.

PARAM.DATA NUMBER	TYPE	DESCRIPTION
01:	4	Delay
02:	4	Iteration count

Examples

The following example involves the use of the Loop instruction without a delay to perform a block data transformation:

The user wants 1-hour averages of the vapour pressure calculated from the wet- and dry-bulb temperatures of five psychrometers. One pressure transducer measurement is also available for use in the vapour pressure calculation.

1. The input locations are assigned as follows:
   a) pressure: Location 10
   b) dry-bulb temperatures: Locations 11-15
   c) wet-bulb temperatures: Locations 16-20
   d) calculated vapour pressure: Locations 16-20 (Vapour pressure is written over the wet-bulb temperatures.)

2. The program flow is as follows:
   a) Enter the Loop instruction (87) with Delay=0 and iteration count=5.
   b) Calculate the vapour pressure with Instruction 57 using a normal location entry of 10 for atmospheric pressure and indexed locations of 11, 16 and 16 for the dry-bulb, wet-bulb, and calculated vapour pressure, respectively.
   c) End the loop with Instruction 95.
   d) Use the If Time instruction (92) to set the Output Flag every hour.
   e) Use the Average instruction (71) with five repetitions starting at input location 16 to average the vapour pressure over the hour.

The actual keyboard entries for the examples are shown below with the first example instruction location equal to 10. The Input Instructions to make the pressure and temperature measurements are not shown.

Table 12-3 Loop Example: Block Data Transform

```
10:        P  87   Beginning of Loop
   01:        0   Delay
   02:        5   Loop Count

11:        P  57   Wet/Dry Bulb Temp to VP
   01:       10   Pressure Loc
   02:       11-- Dry Bulb Temp Loc DRY BLB#1
   03:       16-- Wet Bulb Temp Loc VP #1
   04:       16-- Loc [:VP #1  ]

12:        P  95   End

13:        P  92   If time is
   01:        0   minutes into a
   02:       60   minute interval
   03:       10   Set high Flag 0 (output)

14:        P  71   Average
   01:        5   Reps
   02:       16   Loc VP #1
```

The Loop with a delay can be used so that only the instructions within the Loop are executed while certain conditions are met. As a simple example, suppose it is desired to execute one set of instructions from midnight until 6 AM, another set between 6 AM and 4 PM, and a third set between 4 PM and midnight. Between 6 AM and 4 PM, samples are desired every 10 seconds; the rest of the time one minute between samples is sufficient. The execution interval is set to 10 seconds; when a one minute sample rate is desired, a delay of 6 (6 x 10s = 60s) is used in the loop.

NOTE The example does not show the actual measurement instructions.

Table 12-4 Example: Loop with Delay

```
*             1   Table 1 Programs
   01:       10   Sec. Execution Interval

01:        P  87   Beginning of Loop
   01:        6   Delay
   02:        0   Loop Count

11:        P  86   Do
   01:        1   Call Subroutine 1

12:        P  89   If X<=>F
   01:       25   X Loc DAY
   02:        3   >=
   03:        6   F
   04:       31   Exit Loop if true

13:        P  95   End
```

Table 12-4 (cont.)

14:	P	87	Beginning of Loop
01:		1	Delay
02:		0	Loop Count
27:	P	86	Do
01:		1	Call Subroutine 1
28:	P	89	If X<=>F
01:		25	X Loc DAY
02:		3	>=
03:		16	F
04:		31	Exit Loop if true
29:	P	95	End
30:	P	87	Beginning of Loop
01:		6	Delay
02:		0	Loop Count
36:	P	86	Do
01:		1	Call Subroutine 1
37:	P	89	If X<=>F
01:		25	X Loc DAY
02:		3	>=
03:		5	F
04:		32	Exit Loop if false
38:	P	95	End
39:	P		End Table 1
*		3	Table 3 Subroutines
01:	P	85	Beginning of Subroutine
01:		1	Subroutine Number
02:	P	18	Time
01:		2	Hours into current year (maximum 8784)
02:		24	Mod/by
03:		25	Loc [:DAY]
03:	P	95	End

Error! Bookmark not defined.Instruction 88: If X Compared to Y

This instruction compares two input locations and, if the result is true, executes the specified command. The comparison codes are given in Table 12-5.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location for X
02:	2	Comparison code (Table 12-4)
03:	4	Input location for Y
04:	2	Command

Table 12-5 Comparison Codes

Parameter 1	Function
1	IF X = Y
2	IF X ≠ Y
3	IF X ≥ Y
4	IF X < Y

Instruction 89: If X Compared to F

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 12-5.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location for X
02:	2	Comparison code (Table 12-5)
03:	FP	Fixed value
04:	2	Command

Instruction 90: Step Loop Index

When used within a loop (Instruction 87), Instruction 90 increments 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 counts 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

Instruction 91: If Flag / Port

This instruction checks the status of one of the ten flags or one of the eight ports and conditionally performs the specified command.

The first parameter specifies the condition to check:

1X	Execute command if flag X is high
2X	Execute command if flag X is low
4X	Execute command if port X is high
5X	Execute command if port X is low

PARAM.DATA NUMBER	TYPE	DESCRIPTION
01:	2	Flag or port condition to check
02:	2	Command

Instruction 92: If Time

This instruction allows you to specify 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 synchronised with real 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 synchronised internally by making a modulo divide (see Instruction 46) of the number of minutes since midnight by the specified real time interval. If the result is 0, the interval has elapsed. Thus, the first interval of the day always starts at midnight (0 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 only 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 Output Flag (flag 0) is a special case in that it is automatically cleared if it is not time to set it.

Note that older versions of the Edlog program do not document this feature — it can, however, still be used.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Time into interval (minutes, or seconds if entered as XXXX--)
02:	4	Time interval (minutes, or seconds if time into interval is entered as XXXX--)
03:	2	Command

Instruction 93: Begin Case Statement

Instruction 93 specifies an input location for comparison with fixed values in subsequent If Case instructions (i.e. Instruction 83). When a comparison is true, the command in the If Case instruction is executed and the program flow goes to the End instruction associated with the Begin Case instruction.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	4	Input location for subsequent comparisons

Example

```
01:     P 93        Case
 01:      2         Case Loc
02:     P 83        If Case Location < F
 01:     69.4       F
 02:      3         Call Subroutine 3
         else
03:     P 83        If Case Location < F
 01:     72         F
 02:     10         Set high Flag 0 (output)
         else
04:     P 83        If Case Location < F
 01:     77.3       F
 02:     30         Then Do
```

```
05:     P 30      Z=F
 01:       0      F
 02:       0      Exponent of 10
 03:      25      Z Loc :
06:     P 95      End  Then Do
07:     P 95      End  of Case Statement
```

Instruction 94: Else

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. The Else instruction is optional; when it is omitted, a false comparison results in execution branching directly to the End instruction. Instruction 94 has no parameters.

Instruction 95: End

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.

Instruction 96: Activate Serial Data Output

Instruction 96 is used to activate Storage Module, printer or tape 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, you can control when the output device(s) are active. Instruction 96 allows a choice of serial data formats and the selection of Addressed or Pin-Enabled devices for the serial print output.

A single parameter is used to select whether the instruction is to control the tape, printer or 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 sends 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). Instruction 80 can also direct output to Input Storage, in which case Instruction 96 assumes that Final Storage Area 1 is the active area.

If the CR10 is already communicating via its serial I/O port when Instruction 96 is executed, the output request is put in a queue and program execution continues. As the I/O port becomes available, each device in the queue is serviced in turn. The request is not put in the queue if the same device is already in the queue. The information contained in the queue (and which determines a unique entry) is the baud rate (if applicable) and the Final Storage Area. Instruction 98 (Send Character) also uses this queue.

When an entry reaches the top of the queue, the CR10 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).

Tape output (code 00) will not activate the tape until at least 512 Final Storage locations have been stored. Code 09 will cause *any* data between the TPTR and

the DSP (location at time Instruction 96 is executed) to be written to tape. Code 00 is used for most tape applications (see Section 4).

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 before the trigger condition (see example in Section 8).

CAUTION

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.* For example, assume the TPTR lags the DSP by less than 512 data points when the datalogger program is altered. On compiling, the TPTR is positioned with the DSP, losing reference to the data that was intended to be transferred to tape. The data is not automatically transferred and appears as a discontinuity in the data file. However, until the ring memory wraps around and data overwrite occurs, the data can still be recovered using the *8 Mode. This scenario is also true for the SPTR and data intended for a Storage Module.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Device Option 00 = Tape 09 = Tape: all data to current DSP 80 = To the other Final Storage Area (New data since the last time) 81 = The other Final Storage Area (The entire active Final Storage Area)

(x) BAUD RATE CODES

0	300 baud
1	1200 baud
2	9600 baud
3	76800 baud

ADDRESSED PRINT DEVICE, x = Baud rate code
1x = Printable ASCII
2x = Comma Delineated ASCII
3x = Binary Final Storage Format
7N = Storage Module N (N=1-8)
7N--= Output File Mark to Storage Module N

PIN-ENABLED PRINT DEVICE, x = Baud rate code
(SDE pulled high)
4x = Printable ASCII
5x = Comma Delineated ASCII
6x = Binary Final Storage Format

Instruction 97: Initiate Telecommunications

Instruction 97 is used to have the CR10 initiate telecommunications in response to certain conditions. When the instruction is executed with the interrupt disable flag set low (see below), the CR10 makes a call and sends the ID number specified in parameter 8 (in ASCII at the specified baud rate). The ID number is sent every 4

seconds until the CR10 receives a response or the time specified in parameter 3 expires. The expected response is to have the ID sent back to the CR10, at which time the CR10 enters the normal telecommunications mode and the time limit on the call becomes inactive. In the normal telecommunications mode, the CR10 waits for commands from the device it called. The CR10 will not send any data without first receiving a command to do so. Campbell Scientific's Telcom program (part of the PC208 Datalogger Support Software) enables IBM PC/XT/AT/PS-2's or compatibles to automatically answer calls and retrieve data.

When the CR10 receives a correct character, it restarts the 4-second timer used to determine when the ID is sent. There is then 4 seconds in which the CR10 waits to receive the next digit of the ID before it again sends the ID. The CR10 must receive the ID in the correct order without mistakes. If an incorrect character is detected, the CR10 immediately sends the correct ID. If a correct response is not received within the time allotted in parameter 3, the CR10 exits telecommunications.

When either the RF Modem or Hayes Modem (DC112) options are specified, the time limit on the call (without a correct response) specified in parameter 3 is timed from the start of the instruction and must include the dialling time.

If the correct response is not received, the CR10 continues making calls. Calls are repeated at the fast interval specified by parameter 4 for the number of retries specified in parameter 5, and then at the slow interval specified in param-eter 6. The actual delay between retries for both the fast and slow attempts 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 ranges between 0 and one half of the delay specified. The resolution of the timer for these delays is the execution interval of the table in which the alarm call is initiated. The randomised retry time is divided by the execution interval to determine how many times instruction 97 must be executed before it calls again. The instruction must be executed each time the table is executed.

Parameter 2 specifies which user flag (1-8) is to be used as the interrupt disable flag. If this flag is set, Instruction 97 does not initiate an alarm call. If the CR10 is in the process of trying to get through with an alarm call, setting the interrupt disable flag aborts further attempts. Instruction 97 sets this flag when an alarm call has received the correct response. Instruction 97 does *not* clear this flag; the flag remains set until cleared by the program or external command. When the flag is cleared, Instruction 97 is reinitialised.

The RF path and/or telephone number is entered by following Instruction 97 with one or more entries of Instruction 63. The RF station IDs and/or phone numbers are entered one digit at a time. Decimal equivalents of certain ASCII characters (see Appendix E) are used to identify breaks. RF stations should be separated with a space (ASCII 32). Indicate the switch from RF stations to a telephone number with a space (ASCII 32) followed by a 'T' (ASCII 84). Carriage Return (ASCII 13) is used to end the series of numbers.

If the call is to go through an RF link to a phone, then the RF Modem is specified in parameter 1. See the PC208 and RF manuals for additional interfacing notes.

When the DC112 Modem (Hayes compatible commands) is specified, the datalogger automatically sends the following commands to the modem before the phone number: ATV0, ATS7=180, and ATDT. The first command causes the modem to respond with digits rather than words. The second command causes the modem to wait for the carrier 180 seconds after calling or answering. The third command causes the modem to dial the number that follows the command in

'Touch Tones'. Additional commands can be entered as part of the telephone number (e.g. ',' for delay or 'P' for pulse dialling). The CR10 does not accept the line feed found in some Hayes 'compatible' modems.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	Modem option and baud rate code. The left digit specifies the modem being used and the right, the baud rate.
		Modem Baud Rate
		0 - RF 0 - 300
		1 - Short-haul 1 - 1200
		2 - DC112 2 - 9600
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	Number 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

Instruction 98: Send Character

Instruction 98 is used with Instruction 63 to send a character or string of characters (up to 255) to the printer. The printer may be either addressed or pin-enabled (see Section 6). Instruction 63 must immediately follow Instruction 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 eight 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 I/O port 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 a listing device. The specified character(s) is sent at the time Instructions 98 and 63 are executed.

PARAM. NUMBER	DATA TYPE	DESCRIPTION
01:	2	1x Addressed Print Device 4x Pin-enabled Print Device x is baud rate code 0 300 baud 1 1200 baud 2 9600 baud 3 76800 baud

Section 13. CR10 Measurements

13.1 Fast and Slow Measurement Sequence

The CR10 makes voltage measurements by integrating the input signal for a fixed time and then holding the integrated value for the analogue 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 2.5V = 333μV). 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 (2.72ms) or the fast integration (250μs). 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 (2.72ms Integration time)	**Fast (250μs Integration time)**	**60Hz rejection**	**50Hz rejection**	**Full Scale range**
1	11	21	31	± 2.5mV
2	12	22	32	± 7.5mV
3	13	23	33	± 25mV
4	14	24	34	± 250mV
5	15	25	35	± 2500mV

One of the most common sources of noise is 50Hz from AC power lines. Where 50Hz noise is a problem, range codes 31 - 35 should be used. Two integrations are made spaced half a cycle apart (see Figure 13-1), which results in the AC noise integrating to 0.

NOTE The integration time for the 2500mV range is 1/10 of the integration time for the other gain ranges (i.e. 272μs slow integration, 25μs fast integration).

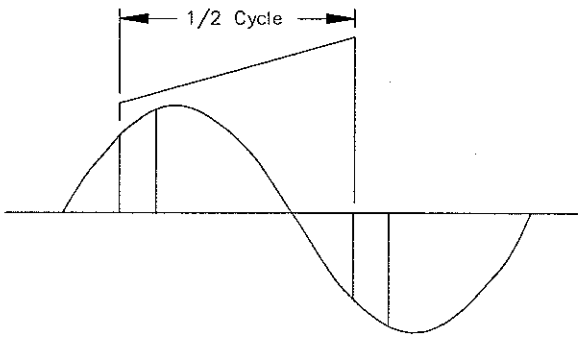


Figure 13-1 50Hz Noise Rejection

There are several situations where the fast integration time is preferred. The fast integration time minimises time skew between measurements and increases the throughput rate. The current drain on the CR10 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 polarise if a DC voltage is applied, causing erroneous readings and sensor decay. The induced voltage in an LVDT decays with time as the 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.

13.2 Single-Ended and Differential Voltage Measurements

NOTE The channel numbering on the CR10 wiring panel refers to differential channels. 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 4 are single-ended channels 7 and 8 respectively.

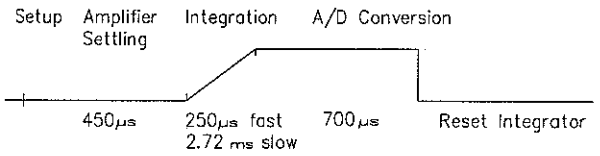


Figure 13-2 Timing of Single-Ended Measurement

The timing and sequence of a single-ended measurement is shown in Figure 13-2. 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 (see Figure 13-3). The CR10 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.

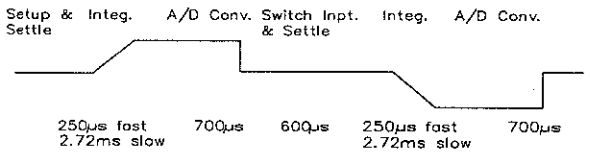


Figure 13-3 Differential Voltage Measurement Sequence

Because a single-ended measurement is referenced to CR10 ground, any difference in ground potential between the sensor and the CR10 will result in an error in the measurement. For example, if the measuring junction of a copper-constantan thermocouple, being used to measure soil temperature, is not insulated and the

potential of earth ground is 1mV greater at the sensor than at the point where the CR10 is grounded, the measured voltage would be 1mV 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, where external signal conditioning circuitry is powered from the same source as the CR10. Despite being tied to the same ground, differences in current drain and lead resistance result in different ground potentials at the two instruments. For this reason a differential measurement should be made on an analogue 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 CR10 common mode range of ±2.5V. The common mode range is the voltage range, relative to CR10 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 2V and the low side is at 1V relative to CR10 ground, there is no problem; a measurement made on the +2.5V range would indicate a signal of 1V. However, if the high input is at 2.8V and the low input is at 2V, the measurement cannot be made because the high input is outside the common mode range. The CR10 indicates the overrange by displaying -9999.

Problems with exceeding common mode range may be encountered when the CR10 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 CR10. 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 CR10 is used to measure the output from a laboratory instrument (both plugged into AC power and referencing ground to the AC outlet ground), it is best to run a ground wire between the CR10 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 preferable.

If a differential measurement is used on a sensor that is not referenced to CR10 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 ground to hold the sensor signal inside the 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 CR10. The input offset voltage on a single-ended measurement is less than 5µV; the input offset voltage on a differential measurement, however, is less than 1µV.

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.

CAUTION Sustained voltages in excess of +16V DC applied to the analogue inputs will damage the CR10 input circuitry.

13.3 The Effect of Sensor Lead Length on Signal Settling Time

Whenever an analogue input is switched into the CR10 measurement circuitry before making a measurement, a finite amount of time is required for the signal to stabilise 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 the input capacitance (explained below). The CR10 allows a 450µs 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 (≅5mV) caused by switching the analogue input into the measurement circuitry must settle.
3. A larger transient, usually about 40mV/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 Campbell Scientific sensors, and error estimates for given lead lengths are provided. Finally, techniques are discussed for minimising input settling error when long leads are unavoidable.

13.3.1 The Input Settling Time Constant

The rate at which an input voltage rises to its full value or at which 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 13-4 shows both a rising and decaying waveform settling to the signal level, V_{so}. The rising input voltage is described by Equation 1 and the decaying input voltage by Equation 2.

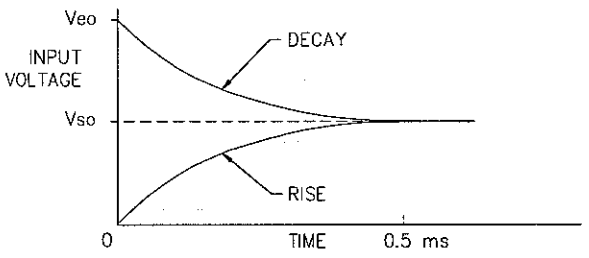


Figure 13-4 Input Voltage Rise and Transient Decay

$$V_S = V_{so}\,(1-e^{-t/R_oC_T}),\ \text{rise} \qquad [1]$$

$$V_S = V_{so} + (V_{eo}-V_{so})\,e^{-t/R_oC_T},\ \text{decay} \qquad [2]$$

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 3 to 5.

$$\tau = R_O C_T \quad [3]$$

$$C_T = C_f + C_w L \quad [4]$$

$$C_f = 3.3nF \quad [5]$$

where C_f is the fixed CR10 input capacitance in farads, C_w is the wire capacitance in farads/metre, and L is the wire length in metres.

Equations 1 and 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 1 and 2, respectively, yields expressions in V_e, the input settling error:

$$V_e = V_{so}\, e^{-t/R_O C_T}, \text{ rise} \quad [6]$$

$$V_e = V_{e'o}\, e^{-t/R_O C_T}, \text{ decay} \quad [7]$$

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 450µs CR10 input settling interval (t). The familiar exponential decay relationship is given in Table 13-1 for reference.

Table 13-1 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.

Determining Source Resistance

The source resistance used to estimate the settling time constant is the resistance the CR10 input 'sees' looking out at the sensor. For our purposes the source resistance can be defined as the resistance from the CR10 input through all external paths back to the CR10. Figure 13-5 shows a typical resistive sensor, (e.g.

a thermistor) configured as a half bridge. Figure 13-6 shows Figure 13-5 re-drawn in terms of the resistive paths determining the source resistance R_o, which is given by the parallel resistance of R_s and R_f, as shown in Equation 8.

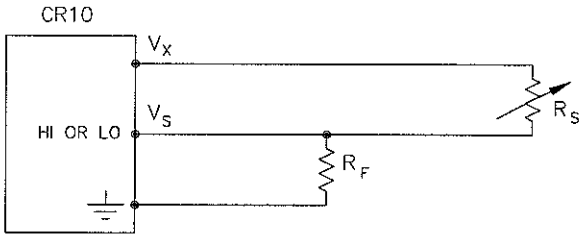


Figure 13-5 Typical Resistive Half Bridge

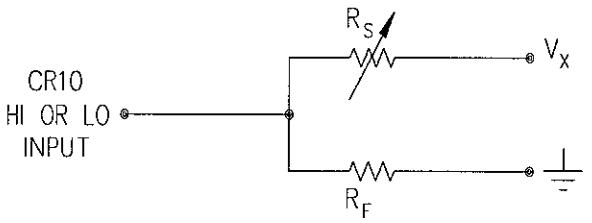


Figure 13-6 Source Resistance Model for Half Bridge Connected to the CR10

$$R_o = R_s R_f/(R_s+R_f) \quad [8]$$

If R_f is much smaller, equal to or much greater than R_s, the source resistance can be approximated by Equations 9 to 11 respectively.

$$R_o \cong R_f,\ R_f << R_s \quad [9]$$

$$R_o = R_f/2,\ R_f = R_s \quad [10]$$

$$R_o \cong R_s,\ R_f >> R_s \quad [11]$$

The source resistance for several Campbell Scientific sensors is given in column 3 of Table 13-5.

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 13-5, the second specification is used in determining lead capacitance. Figure 13-7 is a representation of this capacitance, C_w, usually specified as pF/m. C_w is actually the sum of the capacitance between the two conductors and the capacitance between the top conductor and the shield. The capacitance for three Belden lead wires used in Campbell Scientific sensors is shown in column 6 of Table 13-2.

Table 13-2 Properties of Three Belden Lead Wires Used by Campbell Scientific

Belden Wire #	Conductors	Insulation	AWG	R_o (ohms/100m)	C_w (pF/m)
8641	1 shld. pair	polyethylene	24	8	142
8771	1 shld. 3 cond.	polyethylene	22	5	134
8723	2 shld. pair	polypropylene	22	5	170

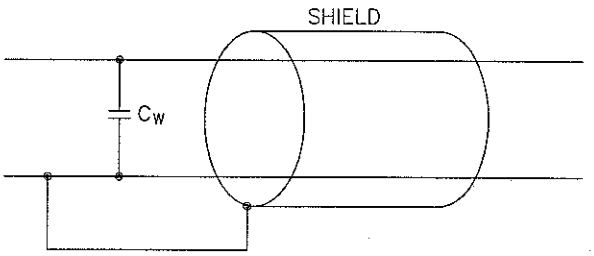


Figure 13-7 Wire Manufacturers' Capacitance Specifications, C_w

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. 50mV or less). The primary rule to follow in minimising dielectric absorption is: *avoid PVC insulation* around conductors. PVC cable jackets are permissible since the jackets don't contribute to the lead capacitance, being outside the shield. Campbell Scientific uses only polyethylene and polypropylene insulated conductors in CR10 sensors (see Table 13-2) 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 13-5). 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.

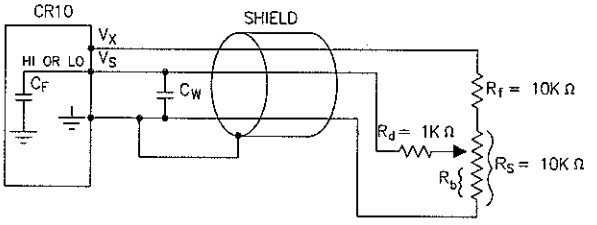


Figure 13-8 Model 024A Wind Direction Sensor

R_o, the source resistance, is not constant because R_b varies from 0 to 10kΩ over the 0 to 360 degree wind direction range. The source resistance is given by:

$$R_o = R_d+R_b(R_s-R_b+R_f)/(R_s+R_f)$$
$$= R_d+R_b(20k-R_b)/20k \quad [12]$$

Note that at 360°, R_o is at a maximum of 6k (R_b=10k) and at 0°, 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:

$$R_b(kohms) = (10k)(D)/360 \quad [13]$$

Equation 6 can be rewritten to yield the settling error of a rising signal directly in units of degrees.

$$Error\ (degrees) = De^{-t/(R_o(C_f+C_wL))} \quad [14]$$

Equations 12, 13 and 14 can be combined to estimate the error directly in degrees at various directions and lead lengths (see Table 13-3). Constants used in the calculations are given below:

C_f = 3.3nF

C_w = 41pF/m Belden #8771 wire

t = 450µs

Table 13-3 Settling Error, in Degrees, for 024A Wind Direction Sensor vs. Lead Length

Wind Direction	----- Error -----	
	L=300m	L=155m
360°	47°	8°
270°	31°	5°
180°	12°	1°
90°	1°	0°

The values in Table 13-3 show that significant error occurs at large direction values for leads in excess of 155m. Instruction 4 (Excite, Delay and Measure) should be used to eliminate errors in these types of situations. Using a 10ms delay, settling errors are eliminated up to lengths that exceed the drive capability of the excitation channel (≅ 700m).

NOTE Other measurement errors are possible when using a potentiometer windvane with a long cable. Please contact Campbell Scientific for further information.

13.3.3 Transients Induced by Switched Excitation

Figure 13-9 shows a typical half bridge resistive sensor, such as Campbell Scientific's 107 Temperature Probe, connected to the CR10. 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.

NOTE Excitation transients are eliminated if excitation leads are contained in a shield independent from the signal leads.

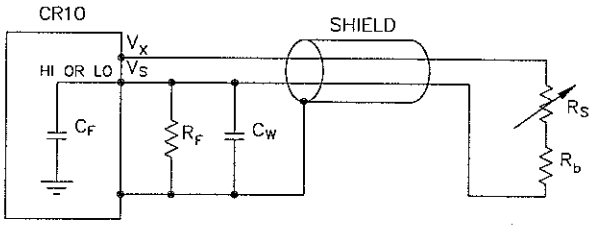


Figure 13-9 Resistive Half Bridge Connected to Single-Ended CR10 Input

The size of the peak transient is linearly related to the excitation voltage and increases as the bridge resistor, R_f, increases. Table 13-4 shows measured levels of V_{eo} for 300m lengths of three Belden wires used in Campbell Scientific sensors. Values are given for R_f equal to 1kΩ and 10kΩ. Table 13-4 only provides estimates of the size of excitation transients encountered; the exact level depends on the specific sensor configuration.

Equation 7 can be solved for the maximum lead length, L, permitted to maintain a specified error limit. Combining Equations 7 and 4 and solving for L gives:

$$L = -(R_o C_f + (t/\ln(V_e/V_{eo})))/R_o C_w \qquad [15]$$

where V_e is the measurement error limit.

Table 13-4 Measured Peak Excitation Transients for 300m Lengths of Three Belden Lead Wires Used by Campbell Scientific

	V_{eo}(mV)					
V_x(mV)	R_f=1kΩ			R_f=10kΩ		
	# 8641	# 8771	# 8723	# 8641	# 8771	# 8723
2000	50	100	60	100	140	80
1000	25	65	40	60	90	40

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 linearisation error over that range. The output signal from the thermistor bridge varies non-linearly with temperature ranging from about 100µV/°C at 0°C to 50µV/°C at 40°C. Taking the most conservative figure yields an error limit of V_e = 2.5µV. The other values needed to calculate the maximum lead length are summarised in Table 13-5 and listed below:

1. $V_{eo} \cong$ 50mV, peak transient at 2V excitation
2. $V_e \cong$ 2.5µV, allowable measurement error
3. t = 450µs, CR10 input settling time
4. R_o = 1kΩ, 107 probe source resistance
5. C_f = 3.3nF, CR10 input capacitance
6. $C_w \cong$ 142pF/m, lead wire capacitance

Solving Equation 15 gives a maximum lead length of:

$L \cong 281m$, error $\cong 0.05°C$

Setting the allowable error at 0.1°C or approximately 5μV, the maximum lead length increases to:

$L \cong 303m$, error $\cong 0.1°C$

13.3.4 Summary of Settling Errors for Campbell Scientific Resistive Sensors

Table 13-5 summarises 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 shows that transient errors are the most likely limitation for the 107 sensor.

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 600m 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 exceeds 0.1μF and the resistive load is negligible, V_X oscillates about its control point. If the capacitive load is 0.1μF or less, V_X settles to within 0.1% of its correct value in 150μs. A lead length of 600m is permitted for the Model 227 before approaching the drive limitation.

Table 13-5 Summary of Input Settling Data For Campbell Scientific Resistive Sensors

Sensor Model #	Belden Wire #	R_o (kΩ)	C_w (pF/m)	τ* (μs)	Input Range(mV)	V_X(mV)	V_{eo}(mV)**
107	8641	1	142	45	7.5	2000	50
207(RH)	8771	1	134	44	250	1500	85
227	8641	0.1-1	142	5-45	250	250	0
237	8641	1	142	45	25	2500	65
024A	8771	1-6	139	1-222	250	500	0-90

* Estimated time constants are for 300m (1000 foot) lead lengths and include 3.3nF CR10 input capacitance.

** Measured peak transients for 300m (1000 foot) lead lengths at corresponding excitation, V_X.

Table 13-6 Maximum Lead Length vs. Error for Campbell Scientific Resistive Sensors

Sensor Model #	Error	Range	V_e(μV)	Maximum Length(m)
107	0.05°C	0°C to 40°C	5	308[1]
207(RH)	1%RH	20% to 90%	250	582[3]
024A	3°	@ 360°	2083	112[2]
227	-	-	-	615[3]
237	10 kohm	20k to 300k	1000	572[3]

1	based on transient settling
2	based on signal rise time
3	limit of excitation drive

Table 13-6 summarises maximum lead lengths for corresponding error limits in six Campbell Scientific sensors. Since the first two sensors are non-linear, the voltage error, V_e, is the most conservative value corresponding to the error over the range shown.

Minimising Settling Errors in Non-Campbell Scientific Sensors

When long lead lengths are unavoidable in sensors configured by the user, the following general practices can be used to minimise or measure settling errors:

1. When measurement speed is not a prime consideration, Instruction 4, Excite, Delay, and Measure, can be used to ensure 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 non-linear thermistor such as the model 44032 is used with a 30kΩ bridge resistor, R_f'. A typical configuration is shown in Figure 13-10A. The disadvantage with this configuration is the high source resistance shown in column 3 of Table 13-7. Adding another 1kΩ resistor, R_f, as shown in Figure 13-10B, lowers the source resistance of the CR10 input. However, 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 of Table 13-7) is essentially the same as the input source resistance of configuration A. Moving R_f' out to the thermistor as shown in Figure 13-10C optimises the signal settling time because it becomes a function of R_f and C_w only.

   Columns 4 and 7 of Table 13-7 list the signal voltages as a function of temperature using a 2000mV excitation for configurations A and C, respectively. Although configuration A has a higher output signal (2500mV input range), it does not yield any higher resolution than configuration C which uses the 250mV input range.

NOTE Since R_f' attenuates the signal in configuration B and C, it could be eliminated altogether. However, its inclusion 'flattens' the non-linearity of the thermistor, allowing more accurate curve fitting over a broader temperature range.

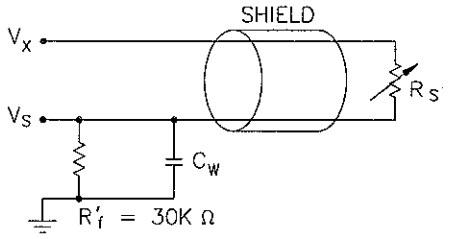
3. Where possible, run excitation leads and signal leads in separate shields to minimise transients.

4. *Avoid PVC insulated conductors* to minimise the effect of dielectric absorption on input settling time.

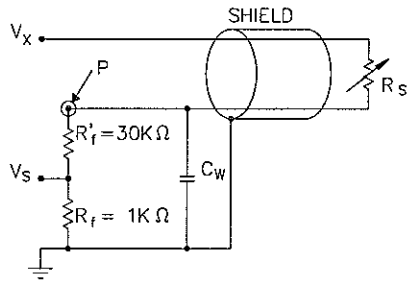
5. Use the CR10 to measure the input settling error associated with a given configuration. For example, assume long leads are required but the lead capacitance, C_w, is unknown. Configure R_f on a length of cable similar to the measurement. Leave the sensor end open as shown in Figure 13-11 and measure the result using the same instruction parameters to be used with the sensor. The measured deviation from 0V is the input settling error.

Table 13-7 Source Resistances and Signal Levels for YSI #44032 Thermistor Configurations Shown in Figure 13-10 (2V Excitation)

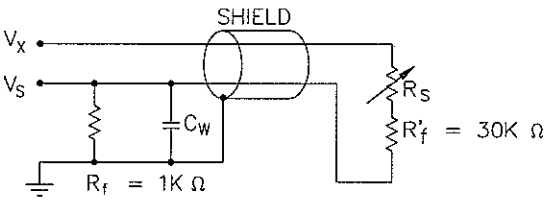
		- - - A - - -		- B -	- - - C - - -	
T	**R_S (kΩ)**	**R_O (kΩ)**	**V_S(mV)**	**R_O@P (kΩ)**	**R_O (kΩ)**	**V_S(mV)**
-40	884.6	29.0	66	30.0	1	2.2
-20	271.2	27	200	27.8	1	6.6
0	94.98	22.8	480	23.4	1	15.9
+25	30.00	15.0	1000	15.2	1	32.8
+40	16.15	10.5	1300	10.6	1	42.4
+60	7.60	6.1	1596	6.1	1	51.8



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 13-10 Half Bridge Configuration for YSI #44032 Thermistor Connected to CR10 Showing:
A) large source resistance,
B) large source resistance at point P, and
C) configuration optimised for input settling

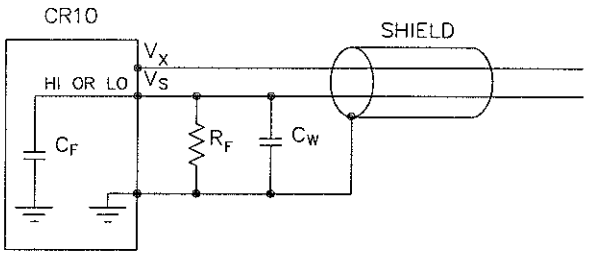


Figure 13-11 Measuring Input Settling Error With the CR10

6. Most Campbell Scientific sensors are configured with a small bridge resistor, R_f, (typically 1kΩ) to minimise the source resistance. If the lead length of a Campbell Scientific sensor is extended simply by joining extension lead on to the end of the existing cable, the effect of the lead resistance, R_l, on the signal must be considered. Figure 13-12 shows a Campbell Scientific Model 107 sensor with 150m of extension lead connected in this way. Normally the signal voltage is proportional to $R_f/(R_s+R_b+R_f)$, but when the cable is joined in this way, 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 analysed by taking the ratio of the signal with extended leads, V_{sl} to the normal signal, V_s:

   $V_{sl}/V_s = (R_f+R_l)/R_f$

   Substituting values of R_f=1kΩ and R_l=.012kΩ (see Table 13-2) gives an approximate 1% error in the signal with extended leads. Converting the error to °C gives approximately a 0.3°C error at 0°C, 0.6°C error at 20°C, and a 1.5°C error at 40°C. The error can be avoided by ensuring that the R_f is moved to the CR10 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.

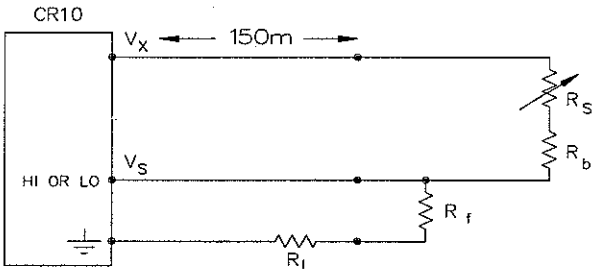


Figure 13-12 Incorrect Lead Wire Extension on 107 Temperature Sensor

13.4 Thermocouple Measurements

A thermocouple consists of two wires, each of a different metal or alloy, which are joined together at each end. If the two junctions are at different temperatures, a voltage proportional to the difference in temperatures is induced in the wires. When a thermocouple is used for temperature measurement, the wires are soldered or welded together at the measuring junction. The second junction, which becomes the reference junction, is formed where the other ends of the wires are connected to the measuring device. (With the connectors at the same

temperature, the chemical dissimilarity between the thermocouple wire and the connector does not induce any voltage.) When the temperature of the reference junction is known, the temperature of the measuring junction can be determined by measuring the thermocouple voltage and adding the corresponding temperature difference to the reference temperature.

The CR10 determines thermocouple temperatures using the following sequence. First, the temperature of the reference junction is measured. The reference junction temperature in °C is stored in an input location which is accessed by the thermocouple measurement instruction (Instruction 13 or 14). The CR10 calculates the voltage that a thermocouple of the type specified would output at the reference junction temperature if its reference junction were at 0°C, and adds this voltage to the measured thermocouple voltage. The temperature of the measuring junction is then calculated from a polynomial approximation of the National Bureau of Standards (NBS) TC calibrations.

13.4.1 Error Analysis

The error in the measurement of a thermocouple temperature is the sum of the errors in the reference junction temperature, the thermocouple output (deviation from standards published in NBS Monograph 125), the thermocouple voltage measurement and the polynomial error (difference between NBS standard and CR10 polynomial approximations). The discussion of errors which follows is limited to these errors in calibration and measurement and does not include errors in installation or matching the sensor to the environment being measured.

Reference Junction Temperature

The junction which is created when a thermocouple is wired to the wiring panel is referred to as the reference junction. The temperature of the reference junction must be known in order to calculate the absolute temperature of the measuring junction. The 10TCRT Thermocouple Reference Temperature is used to measure the temperature of the reference junction (terminal strips).

The 10TCRT uses a precision thermistor to measure the relative temperature of the terminal strips. The accuracy of this measurement is a combination of the thermistor's interchangeability specification, the precision of the bridge resistors, and the polynomial error. In a 'worst case' example, all errors add to yield a ±0.4°C error over the range of -20°C to +48°C. It is emphasised that this is the worst case. Campbell Scientific's experience shows that the overall accuracy is typically better than ±0.2°C. The major error component in the -20°C to +60°C. range is the ±0.32°C thermistor specification. When a CR10 is outside this temperature range, the polynomial error becomes much worse (see Figure 13-13), and may necessitate the use of an external reference junction to improve the accuracy.

If the terminal that the thermocouple is wired into is at a different temperature than the 10TCRT thermistor, this difference in temperature becomes an error in the thermocouple temperature measurement. With the CR10 in an enclosure (see Section 14) this error will generally be less than 0.3°C.

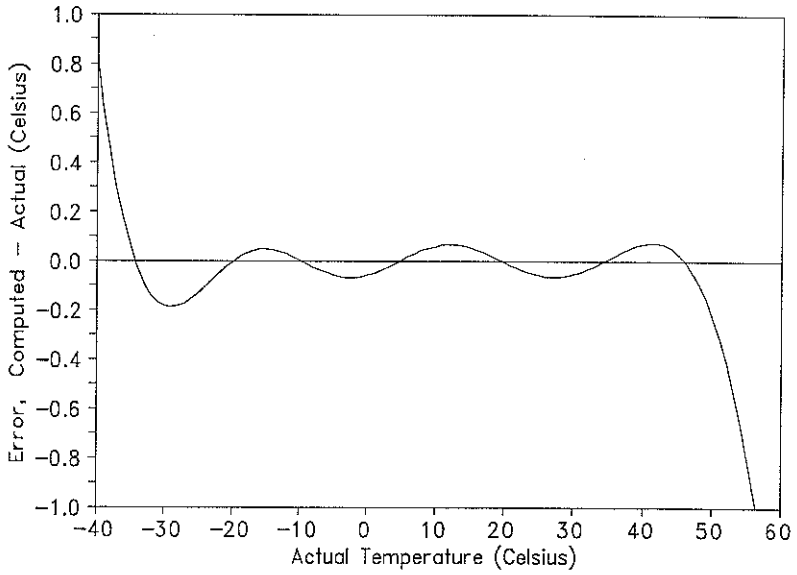


Figure 13-13 Thermistor Polynomial Error

Thermocouple Limits of Error

The standard reference which lists thermocouple output voltage as a function of temperature (reference junction at 0°C) is the National Bureau of Standards Monograph 125 (1974). The American National Standards Institute has established limits of error on thermocouple wire which are accepted as an industry standard (ANSI MC 96.1, 1975). Table 13-8 gives the ANSI limits of error for standard and special grade thermocouple wire of the types accommodated by the CR10.

Table 13-8 Limits of Error for Thermocouple Wire (Reference Junction at 0°C)

Thermocouple Type	Temperature Range °C	Limits of Error (Whichever is greater) Standard	Special
T	-200 to 0	± 1.0°C or 1.5%	
	0 to 350	± 1.0°C or 0.75%	± 0.5°C or 0.4%
J	0 to 750	± 2.2°C or 0.75%	± 1.1°C or 0.4%
E	-200 to 0	± 1.7°C or 1.0%	
	0 to 900	± 1.7°C or 0.5%	± 1.0°C or 0.4%
K	-200 to 0	± 2.2°C or 2.0%	
	0 to 1250	± 2.2°C or 0.75%	± 1.1°C or 0.4%

When both junctions of a thermocouple are at the same temperature, there is no voltage produced (law of intermediate metals). A consequence of this is that a thermocouple cannot have an offset error; any deviation from a standard (assuming the wires are each homogeneous and no secondary junctions exist) is due to a deviation in slope. In view of this, the fixed temperature limits of error (e.g. +1.0°C for type T as opposed to the slope error of 0.75% of the temperature) in the table above are probably greater than you would experience when considering temperatures in the environmental range. In other words, the reference junction, at 0°C, is relatively close to the temperature being measured, so the absolute error (the product of the temperature difference and the slope error) should be closer to the percentage error than the fixed error. Likewise, because

thermocouple calibration error is a slope error, accuracy can be increased when the reference junction temperature is close to the measurement temperature. For the same reason, differential temperature measurements, over a small temperature gradient, can be extremely accurate.

In order to quantitatively evaluate thermocouple error when the reference junction is not fixed at 0°C, the limits of error for the Seebeck coefficient (slope of thermocouple voltage vs. temperature curve) for the various thermocouples are required. Lacking this information, a reasonable approach is to apply the percentage errors, with perhaps 0.25% added on, to the difference in temperature being measured by the thermocouple.

Accuracy of the Thermocouple Voltage Measurement

The accuracy of a CR10 voltage measurement is specified as 0.2% (0.1% from 0 to 40°C) of the full scale range being used to make the measurement. The actual accuracy may be better than this as it involves a slope error (the error is proportional to the measurement being made, although this is limited by the resolution). The error in the temperature due to inaccuracy in the measurement of the thermocouple voltage is worst at temperature extremes, where a relatively large scale is necessary to read the thermocouple output. For example, assume type K (chromel-alumel) thermocouples are used to measure temperatures at 600°C. The TC output is of the order of 24.9mV, requiring the 25mV input range. The accuracy specification of 0.1% FSR is 25μV which is a temperature error of about 0.60°C. However, in the environmental temperature range, with the voltage measured on an appropriate scale, the error in temperature due to the voltage measurements is a few hundredths of a degree.

Thermocouple Polynomials – Voltage to Temperature Conversion

NBS Monograph 125 gives high order polynomials for computing the output voltage of a given thermocouple type over a broad range of temperatures. In order to speed processing and accommodate the CR10's mathematics and storage capabilities, four separate sixth order polynomials are used to convert from voltage to temperature over the range covered by each thermocouple type. Table 13-9 gives error limits for the thermocouple linearisation functions.

Table 13-9 Limits of Error in CR10 Thermocouple Output Linearisation (Relative to NBS Standards)

TC Type	Range °C	Limits of Error °C
T	**-270 to 400**	
	-270 to -200	±18 @ -270
	-200 to -100	± 0.08
	-100 to 100	± 0.001
	100 to 400	± 0.015
J	**-150 to 760**	± 0.008
	-100 to 300	± 0.002
E	**-240 to 1000**	
	-240 to -130	± 0.4
	-130 to 200	± 0.005
	200 to 1000	± 0.02
K	**-50 to 1372**	
	-50 to 950	± 0.01
	950 to 1372	± 0.04

Reference Junction Compensation – Temperature to Voltage Conversion

The polynomials used for reference junction compensation (converting reference temperature to equivalent TC output voltage) do not cover the entire thermocouple range. Substantial errors will result if the reference junction temperature is outside of the calibrated range. The ranges covered by these calibrations include the CR10 environmental operating range, so there is no problem when the CR10 is used as the reference junction. External reference junction boxes, however, must also be within these temperature ranges. Temperature difference measurements made outside of the reference temperature range should be made by obtaining the actual temperatures referenced to a junction within the reference temperature range and subtracting. Table 13-10 gives the reference temperature ranges covered and the limits of error in the linearisations within these ranges.

Two sources of error arise when the reference temperature is out of range. The most significant error is in the calculated compensation voltage; however, error is also created in the temperature difference calculated from the thermocouple output. For example, suppose the reference temperature for a measurement on a type T thermocouple is 300°C. The compensation voltage calculated by the CR10 corresponds to a temperature of 272.6°C, a -27.4°C error. The type T thermocouple with the measuring junction at 290°C and reference at 300°C would output -578.7μV; using the reference temperature of 272.6°C, the CR10 calculates a temperature difference of -10.2°C, a -0.2°C error. The temperature calculated by the CR10 would be 262.4°C, 27.6°C low.

Table 13-10 Reference Temperature Compensation Range and Linearisation Error Relative to NBS Standards

TC Type	Range °C	Limits of Error °C
T	-100 to 100	± 0.001
J	-150 to 296	± 0.005
E	-150 to 206	± 0.005
K	-50 to 100	± 0.01

Error Summary

The magnitude of the errors described in the previous sections illustrate that the greatest sources of error in a thermocouple temperature measurement are likely to be the limits of error on the thermocouple wire and in the reference temperature determined with the 10TCRT thermistor. Errors in the thermocouple and reference temperature polynomials are extremely small, and error in the voltage measurement is negligible.

To illustrate the relative magnitude of these errors in the environmental range, we will take a worst case situation where all errors are maximum and additive. A temperature of 45°C is measured with a type T (copper-constantan) thermocouple, using the ±2.5mV range. The nominal accuracy on this range is 2.5μV (0.1% of 2.5mV), which at 45°C changes the temperature by 0.06°C. The reference temperature device (RTD) is at 25°C but is indicating 25.3°C, and the terminal that the thermocouple is connected to is 0.3°C cooler than the RTD.

Table 13-11 Example of Errors in Thermocouple Temperature

Source	Error °C	% of Total Error	
		1°C Error	1% Slope Error
Reference Temp.	0.6	36.1	69.6
TC Output			
ANSI	1.0	60.1	
0.01 x 20°C	0.2		23.2
Voltage Measurement	0.06	3.6	7.0
Reference Linearisation	0.001	0.1	0.1
Output Linearisation	0.001	0.1	0.1
Total Error			
With ANSI error	1.662	100	
Assuming 1% slope error	0.862		100

13.4.2 Use of External Reference Junction or Junction Box

An external junction box is often used to make connections simpler and to reduce the expense of thermocouple wire when the temperature measurements are to be made at a distance from the CR10. In most situations, it is preferable to make the box the reference junction, in which case its temperature is measured and used as the reference for the thermocouples; copper wires are run from the box to the CR10. Alternatively, the junction box can be used to couple extension grade thermocouple wire to the thermocouples being used for measurement, and the CR10 panel used as the reference junction. Extension grade thermocouple wire has a smaller temperature range than standard thermocouple wire, but meets the same limits of error within that range. The only situation where it would be necessary to use extension grade wire instead of an external measuring junction is where the junction box temperature is outside the range of reference junction compensation provided by the CR10. This is only a factor when using type K thermocouples, where the upper limit of the reference compensation linearisation is 100°C and the upper limit of the extension grade wire is 200°C. With the other types of thermocouples, the reference compensation range is to equal to or greater than the extension wire range. In any case, errors can arise if temperature gradients exist within the junction box.

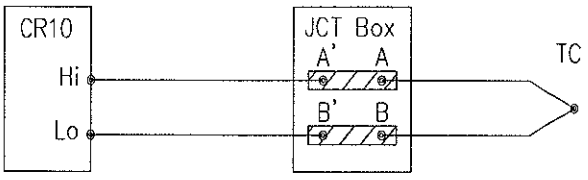


Figure 13-14 Diagram of Junction Box

Figure 13-14 illustrates a typical junction box. Terminal strips will be a different metal than the thermocouple wire. Thus, if a temperature gradient exists between A and A' or B and B', the junction box acts as another thermocouple in series, creating an error in the voltage measured by the CR10. This thermoelectric offset voltage is a factor whether or not the junction box is used for the reference. It can be minimised by making the thermal conduction between the two points large and the distance small. The best solution in the case where extension grade wire is being connected to thermocouple wire would be to use connectors which clamped the two wires in contact with each other.

An external reference junction box must be constructed so that the entire terminal area is very close to the same temperature. This is necessary so that a valid reference temperature can be measured, and to avoid a thermoelectric offset voltage which will be induced if the terminals at which the thermocouple leads are connected (points A and B in Figure 13-14) are at different temperatures. The box should contain elements of high thermal conductivity, which will act to rapidly remove any thermal gradients to which the box is subjected. It is not necessary to design a constant temperature box; it is desirable that the box respond slowly to external temperature fluctuations.

Radiation shielding must be provided when a junction box is installed in the field. Care must also be taken that a thermal gradient is not induced by conduction through the incoming wires. The CR10 can be used to measure the temperature gradients within the junction box.

13.5 Bridge Resistance Measurements

There are six bridge measurement instructions included in the standard CR10 software. Figure 13-15 shows the circuits that would typically be used with these instructions. In the diagrams, the resistors labelled R_S would normally be the sensors and those labelled R_f would normally be fixed resistors. Circuits other than those shown could be used, 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 voltages can then be accounted for in the processing of the measurement instruction. The excitation is switched on 450µs before the integration portion of the measurement starts and is grounded as soon as the integration is completed. Figure 13-16 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 13-16, excitation is switched on for the four integration periods and switched off between integrations.

The measurement sequence for Instruction 8 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 (see Section 13.2). The result stored is the voltage measured. The resolution and common mode rejection of Instruction 8 are not as good as the ratiometric bridge measurement instructions. It does, however, 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 (i.e. 1x, where x is the initial excitation channel number) causes the channel to be incremented with each repetition.

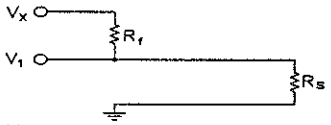
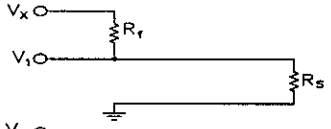
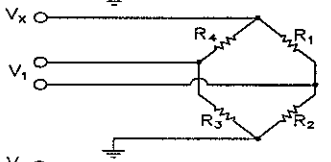
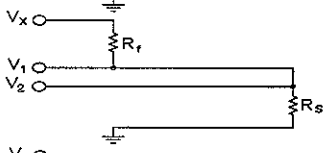
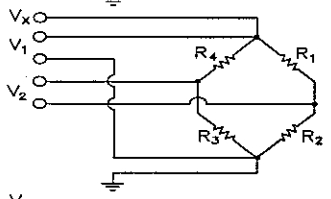
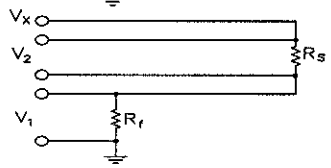
INSTR. #	DIAGRAM	DESCRIPTION	RESULT = X
4		DC HALF BRIDGE WITH USER ENTERED SETTLING TIME	$X = V_1 = V_x \frac{R_s}{R_s+R_f}$
5		AC HALF BRIDGE EXCITATION ALTERNATES POLARITY FOR ION DEPOLARIZATION	$X = \frac{V_1}{V_x} = \frac{R_s}{R_s+R_f}$
6		4 WIRE FULL BRIDGE	$X = 1000 \frac{V_1}{V_x} = 1000 \left(\frac{R_3}{R_3+R_4} - \frac{R_2}{R_1+R_2}\right)$
7		3 WIRE HALF BRIDGE	$X = \frac{2V_2 - V_1}{V_x - V_1} = \frac{R_s}{R_f}$
9		6 WIRE FULL BRIDGE WITH EXCITATION LEAD COMPENSATION	$X = 1000 \frac{V_2}{V_1} = 1000 \left(\frac{R_3}{R_3+R_4} - \frac{R_2}{R_1+R_2}\right)$ (V_1 NOT ON 2.5 V RANGE)
9		4 WIRE HALF BRIDGE	$X = \frac{V_2}{V_1} = \frac{R_s}{R_f}$ (V_1 NOT ON 2.5 V RANGE)

Figure 13-15 Circuits Used with Instructions 4 to 9

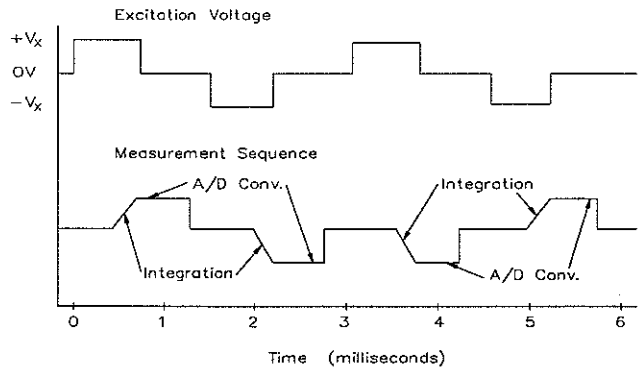


Figure 13-16 Excitation and Measurement Sequence for 4-Wire Full Bridge

Table 13-12 Comparison of Bridge Measurement Instructions

Instruction No.	Circuit	Description
4	DC Half Bridge	The delay parameter allows the user-entered settling time to 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 depolarisation. 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 Instruction 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.

The output of Instruction 8 is simply the voltage measurement. When it is used to measure a full bridge (same connections as Instruction 6 in Figure 13-15), 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.)

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 13-13 lists the instructions used to compute the resistance of any single resistor shown in the diagrams in Figure 13-15, provided the values of the other resistors in the bridge circuit are known.

Table 13-13 Calculating Resistance Values from Bridge Measurement

Instr.	Result	Instr.	Multiplier and Offset
4	$X = V_x(R_s/(R_s+R_f))$		
	$R_s = R_f \frac{X/V_x}{1-X/V_x}$	**4.**	Mult. = $1/V_x$; offset = 0
		59.	Mult. = R_f
	$R_f = \frac{1}{((X/V_x)/(1-X/V_x))/R_s}$	**4.**	Mult. = $1/V_x$; offset = 0
		59.	Mult. = $1/R_s$
		42.	
5	$X = R_s/(R_s+R_f)$		
	$R_s = R_f \frac{X}{1-X}$	**5.**	Mult. = 1; offset = 0
		59.	Mult. = R_f
	$R_f = \frac{1}{(X/(1-X))/R_s}$	**5.**	Mult. = 1; offset = 0
		59.	Mult. = $1/R_s$; offset = 0
		42.	
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.**	Mult. = -0.001; offset = $R_3/(R_3+R_4)$
		8.	Mult. = $1/V_x$; offset = $R_3/(R_3+R_4)$
		59.	Mult. = $1/R_2$
		42.	
	where $X_1 = -X/1000 + R_3/(R_3+R_4)$		
	$R_2 = R_1(X_2/(1-X_2))$	**6 or 9.**	Mult. = -0.001; offset = $R_3/(R_3+R_4)$
		59.	Mult. = R_1
	where $X_2 = X_1$		
	$R_3 = R_4(X_3/(1-X_3))$	**6 or 9.**	Mult. = 0.001; offset = $R_2/(R_1+R_2)$
		59.	Mult. = R_4
	where $X_3 = X/1000 + R_2/(R_1+R_2)$		
	$R_4 = \frac{1}{(X_4/(1-X_4))/R_3}$	**6 or 9.**	Mult. = 0.001; offset = $R_2/(R_1+R_2)$
		59.	Mult. = $1/R_3$
		42.	
	where $X_4 = X_3$		
7&9*	$X = R_s/R_f$		**used as half bridge*
	$R_s = R_fX$	**7 or 9.**	Mult. = R_f; offset = 0
	$R_f = R_s/X$	**7 or 9.**	Mult. = $1/R_s$; offset = 0
		42.	

13.6 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 polarisation, 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 207 relative humidity measurement Instruction 12), reverses excitation polarity to provide ion depolarisation. In order to minimise the time excitation is on, it also grounds the excitation as soon as the signal is integrated (see Figure 13-17).

CAUTION 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 polarisation to affect the measurement.

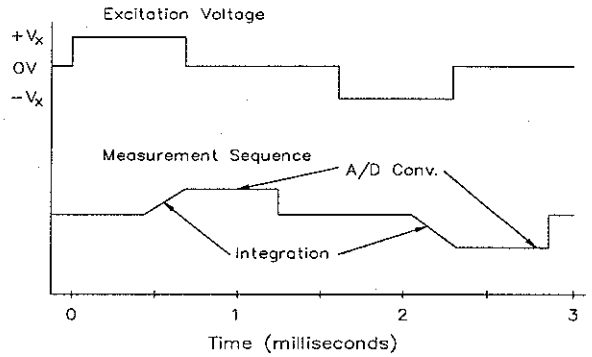


Figure 13-17 AC Excitation and Measurement Sequence for AC Half Bridge

13.6.1 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 alternative path for the excitation to return to CR10 ground, and can be represented as shown in Figure 13-18.

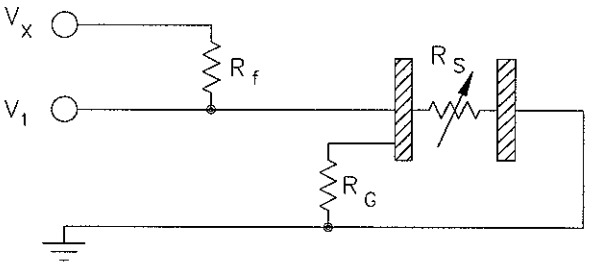


Figure 13-18 Model of Resistive Sensor with Ground Loop

In Figure 13-18, 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 CR10 earth ground. With R_G in the network, the measured signal is:

$$V_1 = V_X \frac{R_S}{(R_S+R_f) + R_SR_f/R_G}$$

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 centre 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 CR10 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.

13.7 Calibration Process

The CR10 makes voltage measurements by integrating the input signal for a fixed time and then holding the integrated value for the analogue to digital (A/D) conversion. The A/D conversion is made by a 13-bit approximation using a digital to analogue 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 five gain ranges for the fast and slow integration times.

The CR10 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°C to +50°C operating range of the CR10. Calibration is executed under four conditions:

1. When the CR10 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.

13.7.1 Automatic Calibration Sequence

The advantage of automatic calibration is that the CR10 is constantly calibrated without user programming. The CR10 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 (for a time which ranges from 5.4 to 21.4ms), 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 CR10 to extreme temperature gradients.

Automatic calibration extends the processing time by an amount which ranges from 5.4 to 21.4ms when it is executed (every 8 seconds). If the processing time of the program then exceeds the execution interval the CR10 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 automatic calibration always causes an overrun. 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 counted in memory.

13.7.2 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 CR10 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.

The calibration coefficients are replaced each time Instruction 24 is executed. Unlike automatic calibration, there is no time constant for the coefficients to respond to changes in calibration. Instruction 24 calibration ensures that the coefficients are optimum at the time that the instruction is executed. For example, consider a CR10 mounted under the dashboard of a vehicle, where the temperature could easily change by 50°. 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 15.6ms, 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 CR10 has a hardware problem or needs factory calibration.

Section 14. Installation and Maintenance

14.1 Protection from the Environment

The standard CR10 operates reliably from -25°C to +50°C (-55°C to +85°C, optional). Internal moisture is eliminated by sealing the module at the factory with three packets of silica gel (0.75g each) inside. The desiccant is replaced whenever the CR10 is repaired at Campbell Scientific. You should not open the module except for the infrequent purpose of PROM replacement (refer to Appendix F). Repeated disassembly/assembly of the CR10 will degrade the seal, leading to potential moisture problems. Extra desiccant could also be placed in the enclosure to prevent corrosion on the Wiring Panel terminals and CR10/ CR10WP Panel connections. This preventative measure is critical in environments with a continuously high relative humidity.

Campbell Scientific offers two enclosures for housing a CR10 Module, wiring panel and peripherals. Each enclosure is constructed of glass fibre reinforced polyester with a hinged, lockable door and the units are water-tight, dust proof and corrosion resistant. (As supplied to Campbell Scientific, each enclosure is classified as IP68 but this rating cannot be applied to enclosures which are sub-sequently fitted with a cable gland. It is then your responsibility to ensure that adequate sealing exists around cables entering via the gland).

Please refer to the *ENC 10/12, ENC 12/14 and AM-ENC Enclosures Installation Manual* for further information on the ENC 10/12 and ENC 12/14.

NOTE If you have bought a PROM upgrade and have one of the older 022 or 028 series enclosures, please refer to your old CR10 manual for information on the enclosure.

14.2 Power Requirements

The CR10 operates at a nominal 12V DC. Below 9.6V or above 16V the CR10 will not operate correctly.

The CR10 is diode-protected against accidental reversal of the positive and ground leads from the battery. Input voltages in excess of 18V may damage the CR10 and/or power supply. A transzorb provides transient protection by limiting the voltage at approximately 20V.

CAUTION The metal surfaces of the CR10WP Wiring Panel and mounting bracket are at the same electrical ground as the power supply. Caution must be exercised when connecting power directly to the Wiring Panel's 12V and ground terminals. Connect the plus (+) side of the supply first, keeping the minus (-) side away from the Wiring Panel. Once the plus side is secured, connect the power return.

You can estimate the battery life by dividing the battery capacity (amp-hours) by the average system current drain. The CR10 draws 0.7mA in the quiescent state, 13mA while processing, and 35mA during an analogue measurement; the length of operating time for each datalogger instruction is given in Section 3. Typical current requirements for common CR10 peripherals are given in Table 14-1.

Table 14-1 Typical Current Drain for Common CR10 Peripherals

Peripheral	Typical Current Drain (mA) Quiescent	Active
CSM1 Card Storage Module	0.3	4 (active, not busy) 17 (when processing)
SM192/SM716 Storage Module	0.25	4.5 (active, not busy) 18 (when processing)
CR10KD Keyboard/Display	0.085	0.3 (*6 Mode)
SDM-A04	10.5	--
SDM-CD16	6.0	11/LED lit
SDM-INT8	0.4	6.5
SDM-SW8A	3	6
RAD Short Haul Modem (with SC932)	<2	11
DC112/DCSCOM Phone Modem	0.002	48/30
RF95 RF Modem	1.4	30
TCV450 UHF Radio	0.2	15 (receive) 80 (transmit)
RC35 Cassette Recorder	0	300

14.3 Campbell Scientific Power Supplies

The power supplies normally sold for use with the CR10 are the PS12E-ALK Alkaline Power Supply and the PS12E-LA Lead-Acid Power Supply. For details on these power supplies please refer to the *PS12E and PS512-M Power Supplies User Guide.*

NOTE If you have bought a PROM upgrade and have one of the older PS38 or PS40 power supplies, please refer to your old CR10 manual for information on the power supply.

14.3.1 Use of Rechargeable Batteries

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:

WARNING

1. **A CR10 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 totally gas-tight enclosure.**

A rechargeable battery pack and charger is ideal for 'float charge' applications where the charger is permanently connected to the mains and the lead-acid batteries provide a reliable back-up in the event of a mains failure. The rechargeable battery option is also ideal in situations where a solar panel is used to provide power on remote sites.

If continuous float charging is not available and the application requires rapid logging, short execution intervals, or the significant use of excitation outputs, the alkaline battery option may be more suitable.

NOTE The alkaline battery pack is not suitable for use in RF applications.

14.4 Solar Panels

Auxiliary photovoltaic power sources such as the SOP5/X, SOP10/X and SOP18 Solar Panels may be used to maintain the charge on lead-acid batteries.

Table 14-2 Solar Panel Specifications

	SOP5/X	SOP10/X	SOP18
Typical Peak Power (Watts)	4.5	10	18.5
Current @ Peak (Amps)	.27	.58	1.1
Amp Hrs/week	6.4	14.4	26.4

NOTE These specifications assume 1kW/m^2 illumination at a panel cell temperature of 25°C. Individual panel performance may vary as much as 10%.

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 infor-mation, 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. Please refer to Campbell Scientific Technical Note 12-93PG for a more detailed analysis of how to estimate power consumption and solar panel size.

The SOP5/X and SOP10/X are most commonly used with the PS12E-LA power supply. The SOP18 solar panel, as normally supplied by Campbell Scientific, is *not* suitable for connecting to the charger input. This is because this panel is normally fitted with a shunt regulator which is designed for direct connection to the battery; this prevents it from working properly with the charger in the PS12E.

If an 18W solar panel is required please either order an SOP18 without shunt regulator, or if you already have an SOP18, remove the shunt regulator (normally mounted in the junction box on the back of the panel).

If other solar panels are to be used, please be aware that a panel should be chosen which gives its optimum power output at a voltage greater than 16V. Some cheaper 12V panels show a sharp decline in efficiency above 14V DC.

14.5 Direct Battery Connection to the CR10WP Wiring Panel

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 CR10WP Wiring Panel. Any 9.6 to 18V DC supply may be connected to the 12V and G terminals on the Wiring Panel. The metal surfaces of the Wiring Panel and mounting bracket are at power ground. Make connections to the Wiring Panel first and then to the power supply. If the power supply must be connected first, connect the positive to the Wiring

Panel before the ground to avoid shorting to the Wiring Panel or mounting bracket.

14.6 Vehicle Power Supply Connections

If a CR10 is to be powered from the 12V of a motor vehicle, a second 12V supply is also required at the time of vehicle start-up. When the starting motor of a vehicle with a 12V electrical system is engaged, the voltage drops considerably below the nominal 12V, which would cause the CR10 to malfunction every time the vehicle was started. The second 12V supply prevents this malfunction. Figure 14-1 shows the general case for connecting the two supplies to the Wiring Panel. The diode allows the vehicle to power the CR10 without the second supply attempting to power the vehicle. To reduce the potential for ground reference errors in measurements, the ground lead should be 1.3mm diameter or larger.

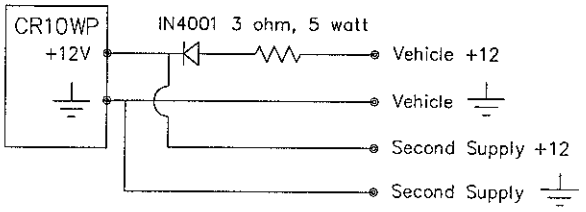


Figure 14-1 Connecting to Vehicle Power Supply

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 minimise 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 Wiring Panel, with the exception of ground (G) and analogue ground (AG) terminals is fitted with a spark gap. The spark gaps are gas discharge tubes which divert current to ground at about 150V.

NOTE In older Wiring Panels, the spark gaps consist of a thin air gap between the conductor and a heavy copper bar tied to ground. Depending on the conductivity of the air, current will be diverted to ground at roughly 400 to 600V.

As shown in Figure 14-2, the power ground and analogue ground are independent lines until joined inside the CR10. The fuse shown in Figure 14-2 (located on the underside of the Wiring Panel) is a 35 SWG (0.21mm diameter) wire, equivalent to a conventional 5A fuse. It will blow if a sufficient transient appears on the G or AG lines, at which time the current is directed away from the CR10 through the diodes. The fuse may be replaced by soldering another 35 SWG wire to the soldering pads provided.

A modem/telephone line connected to the Wiring Panel provides another path for transients to enter and damage the CR10. Campbell Scientific's DC112 telephone modem has spark gaps on the phone lines; other modems may require a separate protection device. A 14 SWG (2mm diameter) 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 Wiring Panel.

The transient protection designed into Campbell Scientific equipment is meaningless if a good system earth ground is not provided. It is your responsibility to provide this earth ground.

In laboratory applications, locating a stable earth ground is not always simple. 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 cannot be verified, it is better to ground the system to a massive metal object such as a steel water pipe.

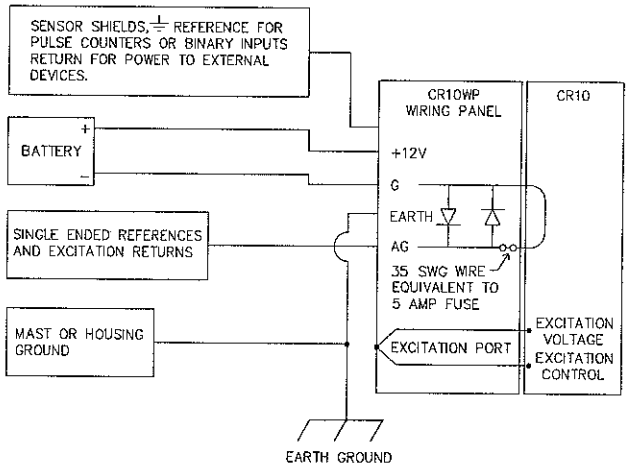


Figure 14-2 CR10WP Wiring Panel Grounding Diagram and Excitation Control

In the field, an earth ground may be created through a grounding rod. A 14 SWG or larger wire should be run between a Wiring Panel power ground (G) terminal and the earth ground. Campbell Scientific's CM10 Tripod is supplied 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 CR10 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 CR10 is ±2.5V. For example, if the high side of a differential input is at 2V and the low side is at 0.5V relative to CR10 ground, a measurement made on the ±2.5V range would indicate a signal of 1.5V. However, if the high input changes to 3V, the common mode range is exceeded and the measurement cannot be made.

Common mode range may be exceeded when the CR10 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 CR10 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 CR10 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 14 SWG wire.

14.8 CR10WP Wiring Panel

The purpose of the CR10WP Wiring Panel is to provide transient protection, improve excitation voltage accuracy, and make convenient, positive connections of power, sensors, and peripherals to the CR10 (refer to Figure 14-2). Wiring Panel transient protection is discussed in Section 14.7.

The Wiring Panel carries two lines between the CR10 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 CR10 and the excitation port on the Wiring Panel (see Figure 14-2).

Two 5V output terminals are available on the Wiring Panel for powering 5V peripherals. The most common use of these terminals is to switch the 5V to a relay coil through a relay driver circuit which is enabled by one of the eight digital control ports, C1 to C8 (see Section 14.9 for relay driver circuits). The 5V ports can source up to 200mA. An input protection transzorb will divert current to ground at approximately 10V.

The switched 12V output (see section OV1.1) can source a maximum of 600mA. Do not allow the switched 12V output to be shorted to ground, as this is likely to damage the CR10.

A functional description of the 37-pin connector located on the CR10 is provided in Appendix D.

14.9 Use of Digital Control Ports for Switching Relays

Each of the eight digital control ports can be configured as an output port and be set low or high (0V low, 5V high) using I/O Instruction 20, Port Set, or commands 41 - 68 associated with Program Control Instructions 83 to 93. A digital output port is normally used to operate an external relay driver circuit because the port itself has a limited drive capability (1.5mA at 3.5V). Figure 14-3 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, 12V from the datalogger energises the relay coil. This closes the relay contacts which completes the power circuit to a fan, turning the fan on. Campbell Scientific offers the A21REL-12 Four Channel Relay Driver (12V coil) and the A6REL-12 Six Channel Relay Driver with manual override (12V coil) for use with the CR10.

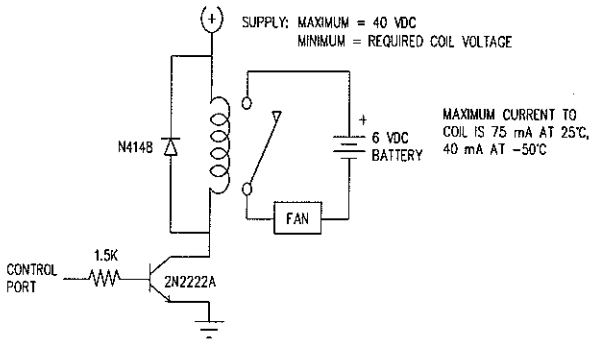


Figure 14-3 Relay Driver Circuit with Relay

In other applications it may be desirable to simply switch power to a device without going through a relay. Figure 14-4 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 75mA at room temperature (the current limit of the 2N2907A medium power transistor), the use of a relay (Figure 14-3) would be required.

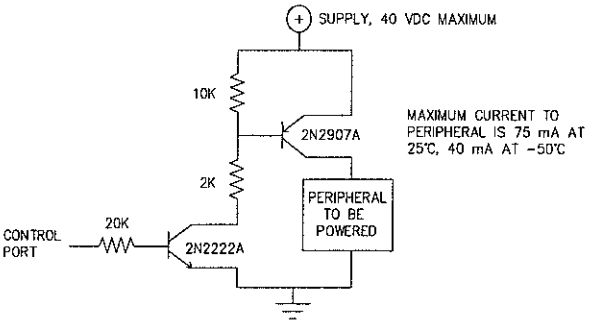


Figure 14-4 Power Switching without Relay

Other circuits activated by a control port are possible for applications with greater current/voltage demands than shown in Figures 14-3 and 14-4. For more information please contact Campbell Scientific.

14.10 Maintenance

The CR10, CR10WP Wiring Panel and power supplies require a minimum of routine maintenance.

When not in use, lead-acid batteries should be stored in a cool, dry environment with the AC charging circuit activated.

Alkaline batteries should not be allowed to drop below 9.6V before replacement. When not in use, remove the eight cells to eliminate potential corrosion of contact points and store in a cool dry place.

The CR10 module is sealed and contains desiccant to protect against the effects of moisture. However, the Wiring Panel and the connections between the Wiring Panel and the CR10 are still susceptible to moisture. To prevent corrosion at these points, additional desiccant must be placed inside the enclosure. If alkaline

batteries only are inside the enclosure, the sensor lead entrance may be sealed to inhibit vapour transfer into the enclosure.

WARNING

Do not seal the entrance completely if lead-acid batteries are present, since hydrogen gas generated by the batteries could build up to an explosive concentration.

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 co-ordination 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 also 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. = .94ms 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 which are output at the same time make up Output Arrays.

Execution Interval

The time between initiating each execution of a given Program Table. If the execution interval is evenly divisible into 24 hours (86400 seconds), the execution interval is synchronised 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 starts on the first even second after compilation.

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.

Final Storage

That portion of memory allocated for storing Output Arrays. Final Storage is configured as a ring memory, with the newest data being written over the oldest. Data in Final Storage can be displayed using the *7 Mode or sent to various peripherals.

Garbage

When data is sent or received incorrectly (and there are numerous reasons why 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 which serves to indicate that they are correctly connected to each other. When not used as a clock line, the CLK/HS (pin 7) line in the CR10 is primarily used to detect the presence or absence of peripherals such as the Storage Module.

High Resolution

A high resolution data value has five significant digits and may range in magnitude from ±.00001 to ±99999. A high resolution data value requires two Final Storage locations (four 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 pressing C before it is entered by pressing A; two dashes (--) will appear at the right of the display. Within a loop (Instruction 87), 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 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 you when a new an instruction is expected.

Intermediate Storage

That portion of memory allocated for the storage of results of intermediate calculations necessary for operations such as averages or standard deviations. You cannot access Intermediate Storage.

Loop

In a program, a series of instructions which are repeated a prescribed number of times, followed by an End instruction which exits the program from the loop.

Loop Counter

A counter which increments by 1 with each pass through a loop.

Low Resolution

The default output resolution. A low resolution data value has four significant decimal digits and may range in magnitude from ±0.001 to ±6999. A low resolution data value requires one Final Storage location (two bytes).

Manually Initiated

An action 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 CR10's ring line or be used with the SC32A to raise the ring line and put the CR10 in the Telecommunications Command State and 2) has an asynchronous serial communication port which can be configured to communicate with the CR10.

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 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.

Output Interval

The time 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.

Output Processing Instructions

These instructions process data values and generate Output Arrays. Examples of Output Processing Instructions include Totalize, Maximize, Minimize and Average. 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 CR10 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 CR10 prompts for the parameters by displaying the parameter number in the ID Field of the display.

Print Device

Any device capable of receiving data when enabled by pin 6 (the PE line) in a receive-only mode. Printers, 'dumb' terminals, and computers in a terminal mode are in this category.

Print Peripheral

See Print Device.

Processing Instructions

These instructions allow you 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

These instructions are 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 two 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. Tables 1 and 2 have independent execution intervals. Table 1 execution has the higher priority; it may interrupt Table 2.

Ring Line (Pin 3)

A line pulled high by an external device to alert the CR10.

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 a series of measurements are). 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 ensures 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 co-ordination is accomplished by synchronising the transmitting and receiving devices to a common clock signal (see also 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 CR10 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 to the next instruction. The absolute maximum throughput rate for a fast single-ended measurement is approximately 256 measurements per second (12 measurements, repeated about 21 times per second). This theoretical maximum is only possible if the CR10's self-calibration function is suspended (this is accomplished by entering Instruction 24 into Program Table 2 while leaving the Execution Interval at zero so that 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).

Window

Each line of information in a functional mode is referred to as a window. For example, the first window in *5 Mode shows the current time, the second window allows you to view and alter the year, and so on.

Appendix B. CR10 PROM Signature and Optional Software

B.1 PROM Signature and Version

The CR10 PROM signature is viewed by entering the *B Mode and advancing to window 2 (see Section 1). The version number is in window 6 and the revision number in window 7. The signatures for the current standard OS10 PROMs are given below. If the CR10 has a library option PROM or the OS10-1.1 PROM, instructions for using the special features can be found in Appendix G* .

Table B-1 CR10 PROM Signature

Software Description	PROM Number	Signature (*B, Window 2)	PROM Version (*B, Window 6)	Revision Number (*B, Window 7)
OS10-0.1	6174-01	51903	.10000	0001
OS10-1.1	6690-01	55030	1.1000	0001
OS10-0.1-4K	8278-00	19464	90.100	0000

Table B-2 CR10 Library Options

Inst. No.	Description	Bytes
	Core files must be included — leaves 5570	FREE
13,14	Add R, S, & B Thermocouple Linearisations	456
15	Control Port Serial I/O	1211
23+	Burst Measurement	1588
27,28+	Period, Vibrating Wire Measurement	932
60	FFT	2271
62	Covariance/Correlation	796
64	Paroscientific 'T' Series Processing169	
81	Rainflow histogram	1159
97+	Initiate telecommunications	880
98+	Send Character	73
100	TDR Soil Moisture Measurement	3308
101+	SDM-INT8	631
102+	SDM-SW8A	268
103,104+	SDM-AO4, SDM-CD16	239
	SDM Utilities	216
105,106	SDI12 Recorder/Sensor	1930
108	UDG01	339
	64Hz Pulse counters reset	-3
*4	Parameter Entry Table	630
*D	Add Tape up/down load	1196
*D	Program on power-up	2100
*D	Prog. on power-up without other *D functions	781
*D	EEPROM program storage	308
*D	EEPROM program and data storage	419

+ These instructions are in OS10-0.1 and are described in the standard manual

* Instruction 108 is described in the UDG01 User Guide. Instruction 100 is described in the TDR Soil Moisture Measurement System manual.

B.2 Available PROMs / Library Options

The set of instructions available in the CR10 is determined by the PROM (Programmable Read Only Memory) that it is equipped with. If the standard PROM, OS10-0.1, does not suit your application it is possible to create a library option PROM with the instructions and features desired, provided the memory capacity of the PROM is not exceeded.

Library Option PROMs are assembled from a base set of instructions plus selected functions such as those listed in Table B-2.

NOTE The Figures in Table B-2 are for guidance only; please consult Campbell Scientific before ordering a library option PROM.

B.3 Description of Library Options Not in Standard Manual

The following is a brief description of library options not found in the standard manual. If the CR10 PROM contains one of the following options or any other non-standard software then detailed information on the special option(s) will be found in Appendix G.

Instructions 13 and 14 — Add R, S and B Thermocouple Linearisations

In addition to the linearisations for the T, E, J and K thermocouples, Instructions 13 and 14 have the R, S and B thermocouple linearisations.

Instruction 15 — Control Port Serial I/O (for smart sensor)

This instruction enables the CR10 to send and receive data through the control ports at 300 or 1200 baud.

Instruction 60 — Fast Fourier Transform

Instruction 60 performs a Fast Fourier Transform (FFT) on a set of data contained in contiguous locations in Input Storage. The FFT is used to obtain information on the relative magnitudes and phases of the various frequency components in a time varying signal.

The results of the FFT can be expressed as: 1) the real and imaginary components, 2) the magnitude and phase components, or 3) the power spectra.

If desired, the original time varying signal can be reconstructed by taking the Inverse Fourier Transform of either the real and imaginary or the magnitude and phase results.

Instruction 62 — Covariance/Correlation

This instruction calculates means, variances, standard deviations, covariances and correlations for a set of input values, and stores the results in Input Storage. Subintervals are allowed for convenient high pass filtering.

Instruction 64 — Paroscientific 'T' Series Processing

Instruction 64 processes measurements made on a Paroscientific 'T' Series pressure transducer. The transducer outputs a temperature frequency and a pressure frequency. The frequencies are measured using the Period Averaging Instruction (Instruction 27), which returns the period of the temperature and pressure signals in microseconds. Instruction 64 converts the periods to temperature and pressure according to relationships provided by Paroscientific.

Instruction 81 — Rainflow Histogram

The Rainflow counting algorithm is a means of estimating cumulative damage fatigue by processing strain measurements to produce a histogram in which closed stress/strain hysteresis loops are counted by amplitude ranges. The histogram can be two dimensional, with each amplitude range sorted on the basis of average strain during the cycle. (The name 'rainflow' comes from the idea that a strain/time plot with strain as the y axis looks like a pagoda roof. The algorithm then is loosely analogous to the way in which rain flows down this roof.)

The instruction can either process a swath of data from measurements made with the Burst Measurement instruction, or it can be used 'on line', processing a new reading with each execution interval.

Instruction 100 — TDR Soil Moisture Measurement

This instruction enables the CR10 to analyse signals from the Campbell Scientific TDR system.

Instructions 105 and 106 — SDI-12 Recorder / Sensor

Instruction 105 allows the CR10 to use its control ports to collect data from an intelligent sensor which uses the SDI-12 communication protocol.

Instruction 106 allows the CR10 to behave as an SDI-12 sensor.

Instruction 108 — UDG01

This instruction simplifies the use of the UDG01 for measuring snow depth or distance. If your CR10 PROM contains Instruction 108, please refer to the description in the UDG01 Manual.

64Hz Reset On Pulse Counters

The reset interval on the pulse counters is changed from 1/8 second to 1/64 second. This allows reading the pulse counters at intervals shorter than 0.125 second and allows measurement of frequencies up to 16kHz with the 8-bit option. The quiescent current drain of the CR10 is increased to 5mA when the pulse count instruction is used. A hardware modification to the CR10 is also required.

*4 Parameter Entry Table

This option allows instruction parameters to be flagged. The values to use for the parameters are then entered into a table in the *4 Mode. This feature is of use when the same program is used for a number of stations but individual changes need to be made for each station (e.g. identical measurements are made at each

station but certain sensors do not have interchangeable calibrations). It is easier and safer to train someone to enter the correct values in the table format than it would be to have them attempt to alter the program.

*D Tape Upload and Download

This library option contains additional *D Mode commands that allow datalogger programs to be saved to and loaded from cassette tape.

*D Program on Power-Up

This option allows your program to be stored in the main system PROM. The program runs automatically when the CR10 is powered up. To use this option you need to be able to program a PROM chip.

*D Program on Power-Up Without Other *D Functions

As above but without certain *D Mode functions (e.g. load program from Storage Module) to save system memory.

*D EEPROM Program Storage

This option allows your program to be stored in 8K EEPROM. The program runs automatically when the CR10 is powered up. This option reduces Final Storage by over 50%. The advantage of the EEPROM option over the Program on Power-Up option described above is that you can easily change your program and load it into the EEPROM without the need for special programming equipment.

*D EEPROM Data and Program Storage

As above but uses 32K EEPROM. Up to 15,000 Final Storage locations can be allocated to be stored in the non-volatile memory of the EEPROM as well as a copy of the program.

Appendix C. Binary Telecommunications

The following commands provide the most efficient extraction of data from a datalogger. They are designed to be used by computer programs which communicate with the datalogger automatically. To work efficiently these instructions use binary data (including non-printable characters); they are therefore not suitable for manual interrogation of the datalogger using a terminal emulator.

The protocols used are not complicated but do require a reasonable knowledge of computer programming to be able to decode the binary data. These instructions are used by Campbell Scientific's PC programs for communications with the dataloggers; there are no other undocumented instructions which are used. Specifically, data collection by Term, GraphTerm and Telcom use the F command, and the monitor mode of Term and GraphTerm use the 3142J and K commands.

An inexperienced user may find the D command for data collection and the I command for reading Input Storage easier to use in the first instance.

C.1 Telecommunications Commands with Binary Responses

C.1.1 The F Command

This requests a block of binary data from the datalogger. The data is sent in Final Storage format and can either be decoded 'on the fly' (using user-written code to decode the data as described in Section C.2), or decoded using the Split program.

After the data the last two bytes sent are a binary checksum. This is a much more effective checksum than that used for ASCII data transfer. For further details of this checksum see Section C.3.

Command	Description
[no. of loc.]F	BINARY DUMP — CR10 sends, in Final Storage Format (binary), the number of Final Storage locations specified (from current MPTR locations), then Signature (no prompt).

C.1.2 Datalogger J and K Commands

The 3142J and K commands are primarily used for reading data from the datalogger's Input Storage. They also allow:

- Reading and writing of the datalogger's real-time clock
- Toggling of ports and flags
- Collection of newly-stored Final Storage data.

For reading Input or Final Storage data these commands are used as a pair, where the J command is sent first to configure the datalogger, followed by one or more K commands which request the data previously specified using the 3142J command. If the datalogger time is to be changed, or the ports or flags toggled, then just a single 3142J command is used, often followed by another 3142J command to set the datalogger up again for subsequent K commands.

3142J

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, toggles datalogger user flag 8. Likewise, the second most significant bit, if set, toggles 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 is reset, or if it is reset, it is set.
3. 'b' is a 1-byte value whose MSB determines whether Final Storage data is returned after the K command. If the MSB is set, Final Storage data, if any, is returned after the next K command. The datalogger initially has this bit reset on entering telecommunications, but once set by a J command, it remains set until reset by another J command or telecommunications is terminated.

   If the second MSB is set, it means that a port toggle byte will follow and port status is to be returned with the K command. Like the MSB, this bit is reset on entering telecommunications, but remains set until reset by another J command or telecommunications is terminated.

   The remaining bits are reserved.
4. If the second MSB in 'b' was set then 'c' is a port toggle byte, otherwise 'c,d,...,n' are each 1-byte binary values 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 code 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. Alternatively, 11111111 binary aborts the J command. If aborted, flags are not toggled and location requests are not 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 Campbell Scientific'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 the most significant byte. To obtain seconds, convert from binary to decimal and 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 the most significant byte. To obtain seconds, convert from binary to decimal and 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 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 Floating Point Format. The format is decoded to the following:

$$Sign(Mantissa*2^{(Exponent)})$$

The Data byte 1 contains the Sign and the Exponent. The most significant bit represents the Sign; if reset the Sign is positive. Subtract 40 hex from the seven least significant bits to obtain the signed exponent.

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 from80 00 00 HEX (.5 decimal) to FF FF FF HEX (1 bit less than 1 decimal, $1-2^{-24}$).

As an example, 41 80 00 00 HEX = $+(.5*2^{(+1)}) = .5*2 = 1$ decimal.

Note these exceptions:

00 00 00 00 HEX = 0 decimal
FF FF FF FF HEX = -99999 decimal

If appropriately requested by a J command, Final Storage data, if any, will immediately follow the input location data. See section C.2 for a description of how to decode Final Storage data in Campbell Scientific's binary data format. Final Storage data is 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. See section C.3 for the meaning and calculation of the signature bytes. The signature in this case is a function of all bytes from the first time byte to 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.

C.2 Final Storage Format

CR10 data is formatted as either 2-byte Low Resolution or 4-byte High Resolution values. The first two bytes of an output array contain a code 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 in the next section.

Table C-1 Low Resolution Data Types

A	B	C	D	E	F	G	H	Data Type And Second Byte Format
1	1	1	1	1	1	0	X	A,B,C = 1: Start of output array, G=0. H is the most significant bit of the output array ID. All eight bits of the second 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 CR10 always transmits a 0 for the second byte, but the word can be decoded on the basis of the first byte only.

C.2.1 Low 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 Campbell Scientific defines the largest allowable range as 6999.

The decimal locators can be viewed as a negative base 10 exponent with decimal locations as follows:

Bits		Decimal Location
B	C	(4 digits)
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 in Table C-1. 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.

C.2.2 High Resolution Format

Continuing to use the A-H bit representation, the 4-byte number is shown in Table C-2 as two 2-byte pairs.

AB0111GH	XXXXXXXX
001111GH	XXXXXXXX

Campbell Scientific 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 A, G and H is the negative exponent to the base 10.

Bits			Decimal Location
G	H	A	(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

Table C-2 Description of High Resolution Format

Bits, first Byte, 1st Pair	Description
CDEF = 0111	Code designating first byte pair of 4-byte number.
B	Polarity, 0 = +, 1 = -.
G,H,A,	Decimal locator as defined below.
Second byte	16th - 9th bit (left to right) of 17-bit binary value.
ABCDEF = 001111	Code designating second byte pair of 4-byte number.
G	Unused bit.
H	17th and MSB of 17-bit binary value.
Second byte	8th - 1st bit (left to right) of 17-bit binary value.

C.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 CR10 calculates the signature using each transmitted byte except 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 initialised 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 high signature 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.

$$S_0(n+1) = T_2 + T_1 + M(n+1)$$

As each new transmitted byte is received, the procedure is repeated.

Appendix D. CR10 37-Pin Port Description

PIN #	DESCRIPTION
1	12V
2	6L
3	AG
4	5H
5	4L
6	AG
7	3H
8	2L
9	AG
10	1H
11	EX CTRL 3*
12	EX CTRL 2*
13	EX CTRL 1*
14	AG
15	P1
16	C7
17	C5
18	C3
19	C1
20	G
21	6H
22	5L
23	AG
24	4H
25	3L
26	AG
27	2H
28	1L
29	AG
30	E3
31	E2
32	E1
33	P2
34	C8
35	C6
36	C4
37	C2

* The EX CTRL lines must be connected to the excitation lines (E1, E2, E3) at the point where excitation is provided (e.g. the excitation terminals on the CR10WP).

Appendix E. 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 F. Changing RAM or PROM Chips

The CR10 has two sockets for Random Access Memory (RAM) and one socket for Programmable Read Only Memory (PROM). The standard CR10 has 64K of RAM, (a 32K RAM chip in each socket). Earlier CR10s had 16K of RAM (an 8K RAM chip in each socket).

It may be necessary to change the EPROM in the CR10 module as a result of the release of a PROM upgrade or the supply of a custom PROM.

Whenever possible, we recommend that you return the CR10 to Campbell Scientific for the new chip to be installed. When the replacement is carried out by Campbell Scientific, a temperature cycling test is performed on the CR10 to ensure guaranteed operation.

Both the new chips and the circuitry of the CR10 are static sensitive. Precautions should be taken to prevent build-up of static on the person installing the chip and on the tools used. Also, avoid touching the pins of the chip. If possible, use a proper IC extractor/insertion tool.

CAUTION The guarantee does not cover damage caused by an unskilled attempt at a RAM or PROM change.

F.1 Disassembling The CR10

The sockets provided for RAM and PROM are located on the CR10 CPU circuit board inside the CR10 can. To expose the RAM and PROM sockets, remove the two phillips head screws from the end opposite the connectors. Remove the end cap. The ends of two circuit boards and the RF shield will be visible (see Figure F-1). Now lay the CR10 on a flat surface and push on the RF shield with your thumbs while grasping the can with your hands. Remove the circuit boards from the can. Orient the boards with the connector on the left and with the board that matches Figure F-2 upwards. The Central Processing Unit (CPU) is found at location H9 and the three slots for RAM and PROM are directly beneath it.

F.2 Installing New RAM Chips in CR10s with 16k RAM

The two 8K RAM chips are found at locations C11 and C14. With a small flat screwdriver gently pry out the two 8K RAM chips at these locations and replace them with the 32K RAM chips provided in the memory upgrade. The new chips should be installed so the notched end is towards the nearest board edge. Before pushing the chips into the socket make certain that all the pins are correctly seated. After installing the 32K chips check for pins that may be bent or not firmly seated in the socket. If you notice a bent pin, remove the chip, carefully straighten it and repeat the installation procedure.

F.2.1 Changing Jumpers

In older CR10s there are six jumpers used to configure hardware for different RAM sizes. Figure F-2 shows the jumper settings for different memory configurations. A pin or small screwdriver tip will work best for pulling these jumpers out and relocating them as shown in Figure F-2.

F.2.2 RAM Test

Attach the CR10KD Keyboard/Display and apply power to the CR10. After the CR10 executes the RAM/PROM self test, the number 96 should be displayed in the window. The number is the sum of kilobytes in RAM (64) plus the number of kilobytes in ROM (32).

F.3 Installing New PROM

The PROM chip is found at location C8 on the CR10 CPU board, (see Figure F-2). With a small flat screwdriver, gently pry out the PROM chip and replace it with the new one. The new chip should be installed so that the notched end is towards the nearest card edge. Before pushing the chip into the socket make certain that all the pins are seating correctly. After installing the chip check for pins that may be bent or not making contact. If you notice a bent pin, remove the chip, carefully straighten it and repeat the installation procedure.

To make certain that the new chip is installed correctly enter the CR10 *B Mode and advance to the second window. This window displays the PROM signature. The five-digit number in the window should match the PROM signature given with the new PROM documentation. If the numbers are different disassemble the CR10 and look for pins that are bent or not firmly seated.

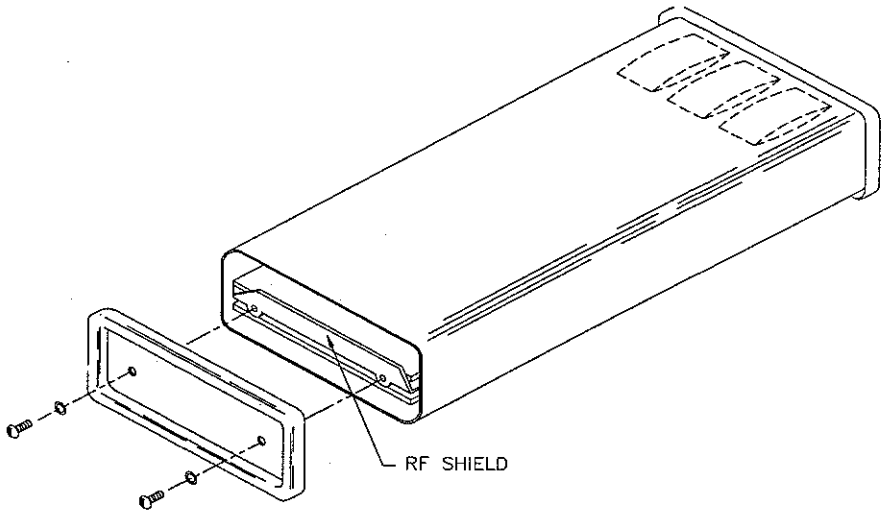


Figure F-1 Disassembling CR10

F.4 Installing 4K Program Memory PROM

Newer CR10s (shipped after April 1994) can be converted from the standard 2K program memory to 4K program memory by installing the correct PROM and moving a jumper. Figure F-3 shows the location and the settings for the jumper. Install the PROM as described above.

Older CR10s do not have this jumper and must be returned to Campbell Scientific for a hardware modification in order to use the 4K program memory PROM.

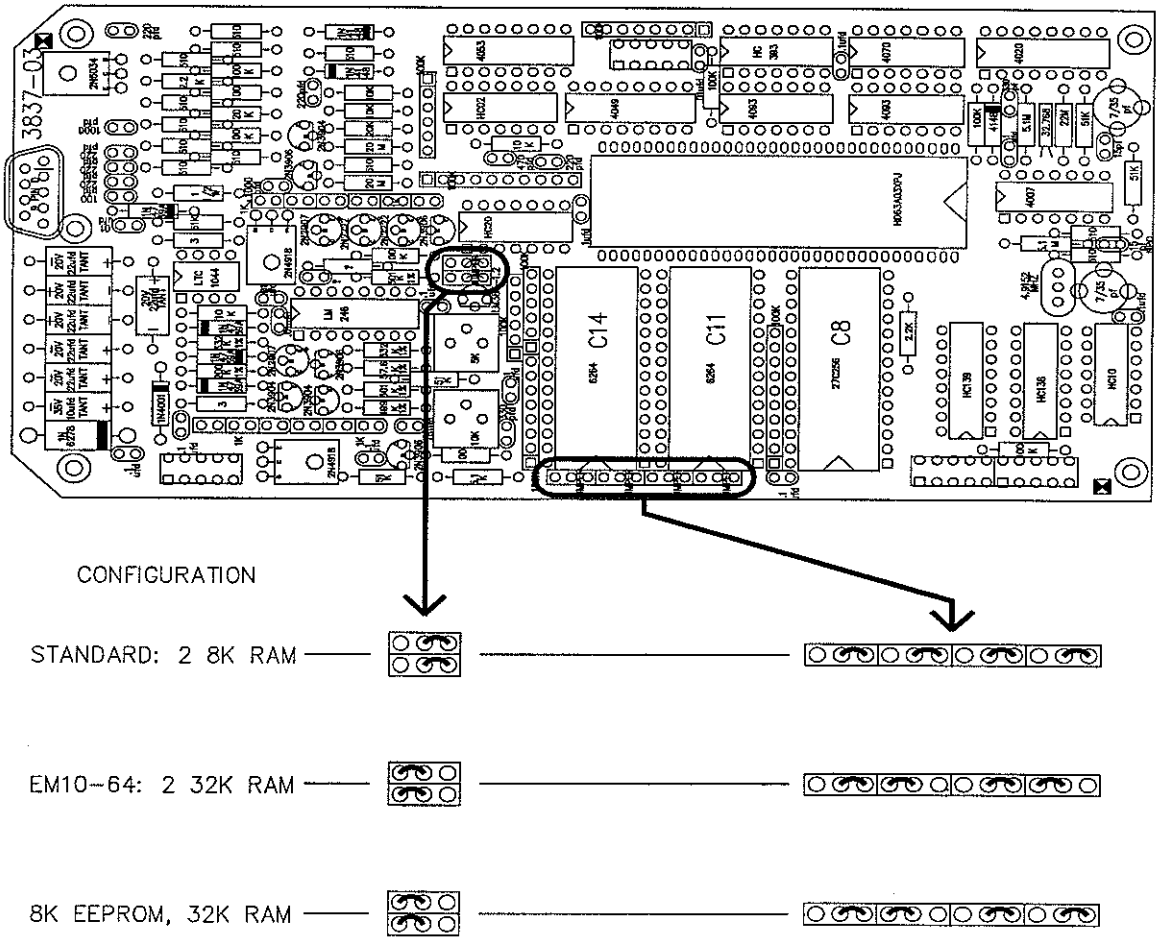


Figure F-2 Jumper Settings for Different RAM Configurations (older CR10s)

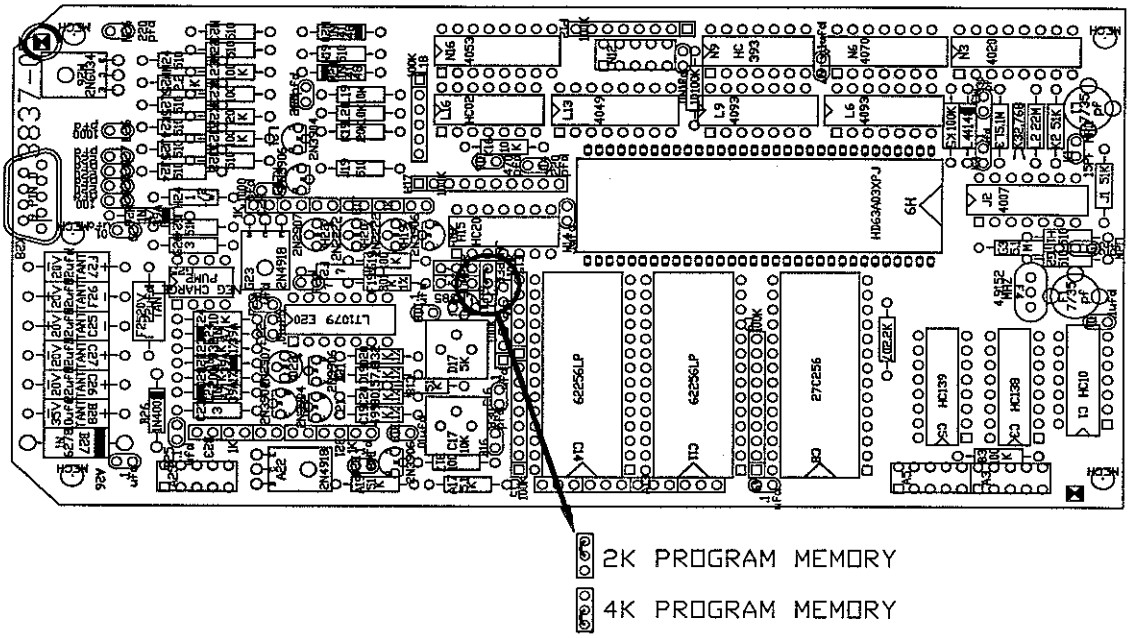


Figure F-3 Jumper Location and Settings for 2K/4K Program Memory

Appendix G. Documentation for Special Software

Figures

Tables

CR10 Index

A

B

C

D

E

F

G

H

I

K

L

M

N

O

P

R

S

T

U

V

W

X

Z