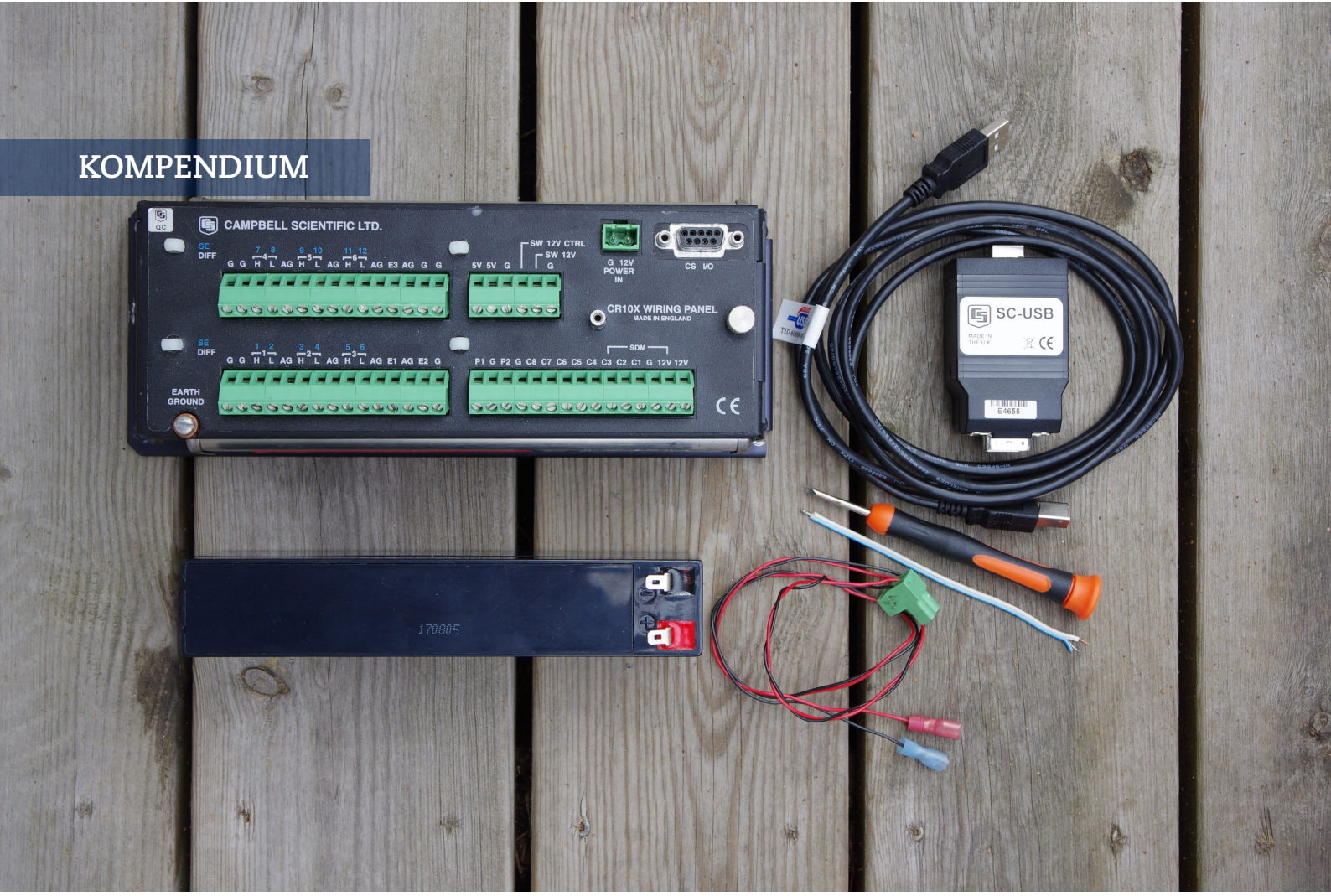
KOMPENDIUM



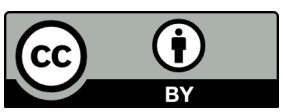
Using Campbell Scientific CR10X data loggers and associated sensors

Peter Jansson

Institutionen för naturgeografi May 6, 2019

Stockholms universitet

Contents



1 Introduction

Campbell Scientific (CS) data loggers have been the main data logging equipment in environmental research since the mid-1980s. The data loggers are very versatile and extremely powerful in what can be recorded. the data loggers contain a small computer, that can be programmed to perform tasks such as measuring different sensors but also to control other equipment. There are several data logger models available from CS but the primary type used by researchers have been the CR10X. The CR10X can use an external display for manipulating the data logger but using a laptop is the most convenient way to work with the logger.

The basic function of the data logger is to allow the user to attach sensors to the logger and program the logger to measure signals from sensors and if needed manipulate the recorded signal into something meaningful such as temperature in °C or wind speed in $\mathrm{m\,s^{-1}}$. To attach sensors, some understanding of elementary is useful, particularly the basics of Wheatstone bridge measurements (Appendix 1).

Working with data loggers involve several steps:

1. Identify what needs to be measured and how
2. Decide how often measurements have to be made
3. Writing a program that tells the logger to make the measurements and save data
4. Locating the appropriate location where measurements need to be made
5. On location, wiring the data logger with the intended sensors and placing sensors where measurements need to be made

This compendium describes how to work with the data loggers and some sensors to get you started and understand how the system works. You will always have to adjust your plan to the sensors and conditions you expect to meet.

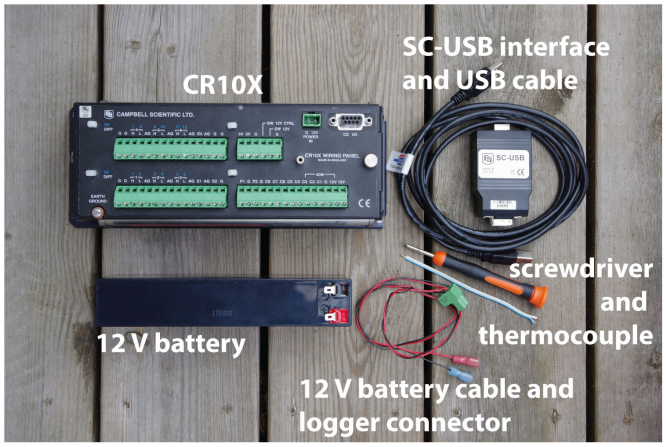
In this manual sequences keyed in using the CR10KD keyboard will be denoted by `typewriter` type face, the display messages are indicated by **italic** letters.

2 The data logger basic functions

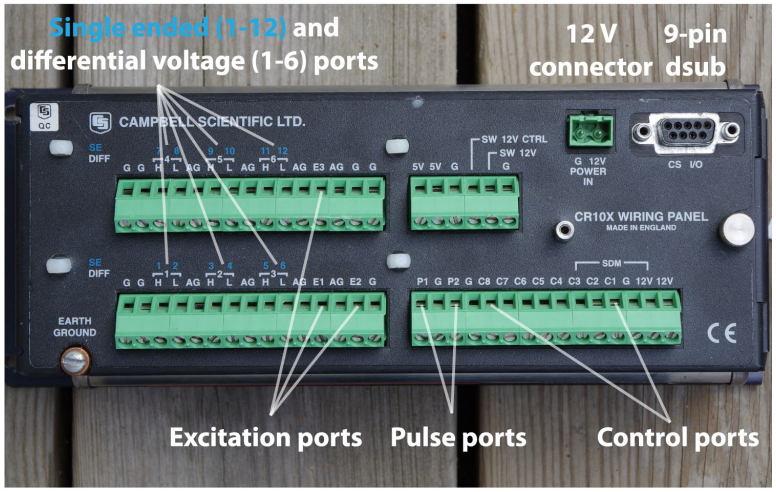
The CS data logger CR10X comes as one assembled unit with the operating part in the form of a panel where sensors, power source and interfaces (such as a laptop or the CR10KD keyboard) can be attached. Looking closer at the panel reveals a series of ports denoted by different abbreviations. We will now go through the meaning of all these.

2.1 The CR-10X logger and periferals

To set up a CS CR10X logger you need, of course, a logger, a battery cable, a 12 V battery, a computer, logger to USB interface and a USB cable. You will also need a small screwdriver to attach sensors to the logger.



The logger consists of a wiring panel where sensors are attached and the encapsulated logger electronics, a steel can beneath the wiring panel. The panel and the logger can can be separated although not unless necessary.



The wiring panel (above) has numerous ports with different functions. These will be discussed in detail below. Apart from the wiring connections there are three additional connection on the wiring panel. The 12 V socket requires a special connector that is delivered with each logger. Please be careful with these connector since the logger cannot be operated without 12 V power supply. Next to the power connector is a 9-pin dsub connector. This is where you connect both the laptop interface and the CR10KD display keyboard. Note that this is not a regular serial port. A special Campbell adapter is required to use the port with any non-CS peripheral.

The battery required is a regular 12 V battery. The logger typically does not draw much current but for long-term use a 6–12 A h battery with the addition of a solar panel is recommended. Please be careful when attaching the battery to ensure that positive and negative poles on the battery is connected to the correct positive (12 V) and negative (G) ports on the logger.

There is also an Earth/Ground connection on the wiring panel. This is intended to ground the logger by attaching a heavy copper wire to the logger and running it to a ground spike firmly embedded in the ground. If the logger is on a mast that in turn is grounded, then the ground wire can be attached to the mast.

In the following, please refer to the figure above of the wiring panel for details on locations of connections.

2.2 Ground

All measurements rely on a common ground or reference level to which measurements are carried out. The CS data loggers provide two types of ground and understanding their significance is very important to all measurements using the logger.

Analog ground (AG). Analog ground is a reference level within the data logger against which certain measurements are referenced. This ground level is arbitrary. The analog ground is also connected to the earth terminal, which is used for grounding the logger to the ground and prevent differences in potential between the data logger and the ground on which it sits. AG is used for referencing in so-called single ended voltage measurements.

Power ground (G). This grounding level is the same as the negative on the power source (battery). This can be important if, for example a sensor is also connected to the same power source. The ground in the data logger and in the sensor may then be the same. G is used as reference for pulse measurements and binary inputs.

2.3 Voltage measurements

Voltage measurements simply measure the difference in direct current (DC) voltage between two leads emanating from a sensor. The CS data logger provides two means of accomplishing this: the singe ended (SE) and the differential (DIFF) voltage measurement. The logger provides 12 possibilities for single ended measurements and 6 for differential measurements. The data logger can cope with measurements between $-2.5\,\mathrm{V}$ and $+2.5\,\mathrm{V}$. If a sensor provides a larger voltage range that $\pm 2.5\,\mathrm{V}$ you need to find a way to reduce the sensor's range to fit the data logger's limitations.

The *single ended voltage measurement* is carried out using any of the the ports labelled SE1 through SE12. A single ended measurement means you attach one lead to the SE port and the other to an AG port. The logger is then capable to measure the voltage difference between the two. Since there are 12 such SE channels up to 12 sensors can be used simultaneously when using SE measurements.

The *differential voltage measurement* is carried out by using any of the six differential ports labelled DIFF1 through DIFF6. Note that DIFF1 consists of SE1 and SE2. Thus only six differentially measured sensors can be used in parallel. A differential measurement simply measures the difference between two leads. This may sound trivial but since no ground is involved, is is possible (but not advisable) to measure the difference between $220\,\mathrm{V}$ and $220.1\,\mathrm{V}$. Although this signal really is $220\,\mathrm{V}$ from ground. Attaching such a measurement as single ended would be a disaster. Note that the differential voltage measurement has a high and low port. It is important to attach the signal wires correctly so that the higher voltage wire is attached to the high port and the other to the low.

2.4 Excitation

When measuring certain sensors it is necessary to first apply an excitation voltage to the sensor to allow it to 'power up' before measuring its status. This is accomplished by so-called Excitation ports labelled E1 through E3. The excitation ports can prove any voltage larger than $0\,\mathrm{V}$ and up to $2.5\,\mathrm{V}$ in steps of $1\,\mathrm{mV}$.

2.5 Pulse ports

The CS data logger can measure sensors that provide pulse as out put. Some common sensors of this type are tipping bucket rain gauges and wind sensors. The pulse ports

are labelled P1 through P2 on the CR10X. These ports register any pulsating signal up to +20 V and with a frequency of 250 kHz. Sensors are simply attached with one lead into the pulse port and the other to a G-ground port.

2.6 Control ports

The CS data logger provides 8 so called control ports, labelled C1 through C8. These ports are digital input/output ports. When configured for output the control ports can be used to switch external equipment on or off using a control voltage to trigger such events in the sensor. Ports 6 and 8 can be used to measure switch closure sensors and then act as pulse counters. This is suitable for tipping bucket rain gauges.

2.7 Switched 12V

Typical output from the logger is as already mentioned ±2.5 V, however, some sensors may require a much higher excitation voltage for operating. The CR10X provides a 12 V output that can be switched on and off by the logger through programming. This is accomplished by connecting the port 'Switched 12V control' to any of the C1–C8 control ports. setting one of these high, then turns the Switched 12 V on. The ports 'Switched 12V' and G can then be used for the external 12 V.

2.8 5V

The CS data logger provides a 5 V output as well. This is not switchable but provides an output with ±0.2% accuracy.

2.9 Concluding remarks on basic functions

The CS data logger has several means for attaching sensors and to the peripheral equipment. The data logger can basically measure any sensor, the crux lies in adapting the sensor so that it can be measured with the limitations imposed by the logger such as voltage limitations, type of measurement etc. To the uninitiated eye, this may seem like severe limitations but in reality it is not, but it requires certain know-how from the user.

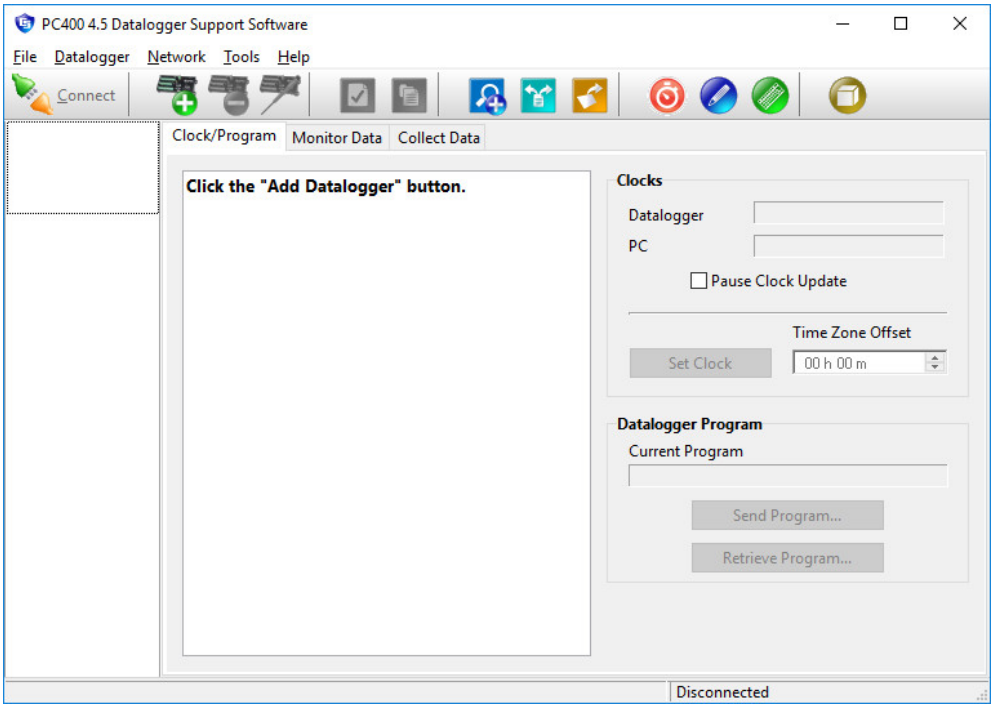
3 Working with the PC400 software

When working with loggers you need to interface with the logger to transfer the program you wish to run to the logger and later on to retrieve data from the logger. This is most conveniently accomplished with a computer (in the field a laptop). The laptop need to have a USB port and some specific software. First you need a software for communicating with the loggers which is what I will describe below. You also need to have drivers installed for the interface that is needed to communicate with the logger through a USB cable. In the following I will assume you have a computer with the necessary software installed.

CS provides several software with which you can work. First there is the free software called PC200W which has all the basic functions needed for working with CS loggers. Then there is the PC400 software which has additional feature compared to PC200W. This mainly concerns how you can write programs for the loggers. PC400 is licensed. Due to its additional functionality it is the software of choice and what this compendium is based. Finally there is the LoggerNet software which is a suite of software that can work with networks of loggers and act as a server for continuous

data retrieval from loggers. LoggerNet is also licensed but contains a large number of features any normal user would not need. So even if we focus on one of the three possible software for CS loggers it may be useful to know of the existence of the others.

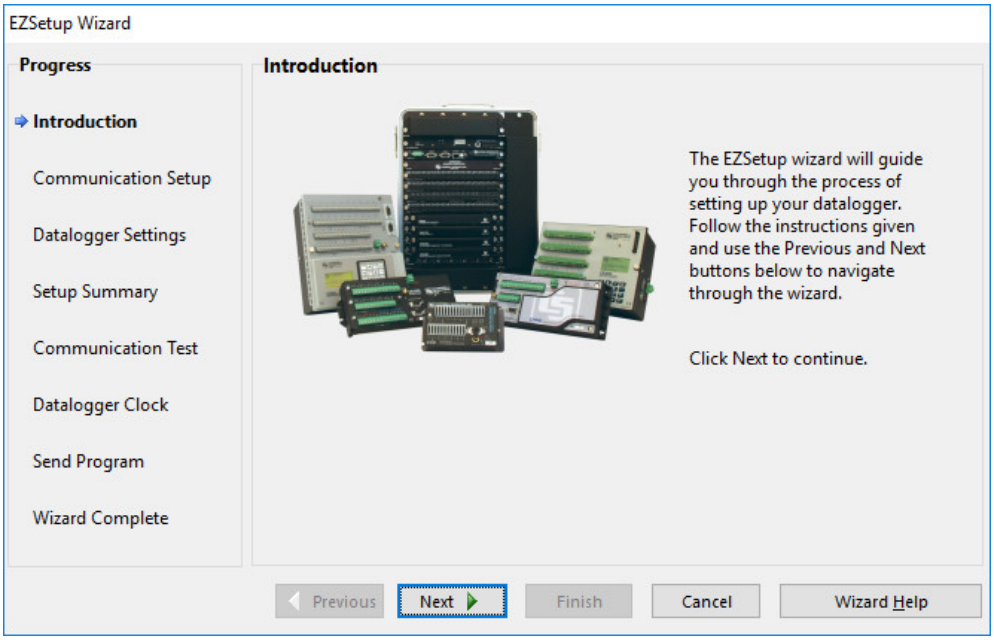
When you start PC400 you are met by the following screen



The row of buttons below the menu bar contains all the tools you will need. To the left there is a button called 'Connect'. Once you have a logger powered up and connected to a computer you can connect to the logger by pressing this button. Next there are three buttons starting with a small image of a logger and a green circle with a plus sign, 'Add Datalogger'. This button allows you to add a logger stations to your computer. We will run through this necessary procedure below. The following button is similar but has a minus sign 'Delete DataLogger'. This allows you to remove an existing logger station from the software. The third in the series shows a logger with a screwdriver superimposed, 'Edit Datalogger Setup'. This allows you to modify an existing logger station. The two grey button with be showing colour once they can be used, as we will see in illustrations below. The last button we will concern ourselves with here is the bright green button to the right. This allows you to open a program editor called Edlog which is where we will author logger programs. The red and blue buttons next to Edlog are two other program editors. The red is a more interactive editor called ShortCut and the blue is called CRbasic. The CRbasic editor is exclusively for another series of loggers than the CR10X and hence of no use here. ShortCut can be sued to set up simple logger programs for the CR10X.

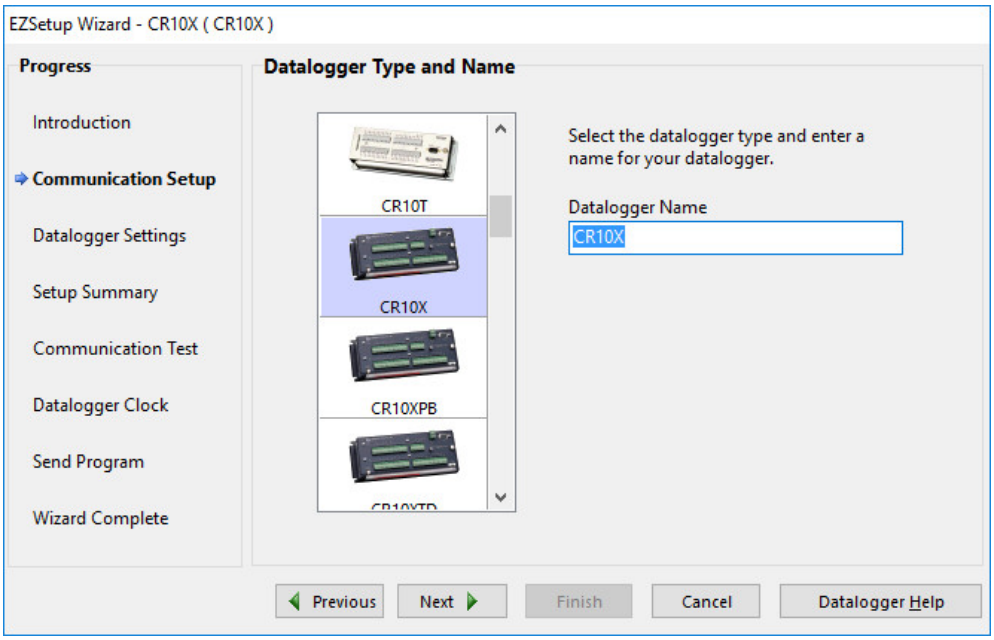
3.1 Setting up a new logger

The first step to connect the laptop to a logger is to press the 'Add Data Logger' button. This opens a new window called 'EZsetup Wizard'.



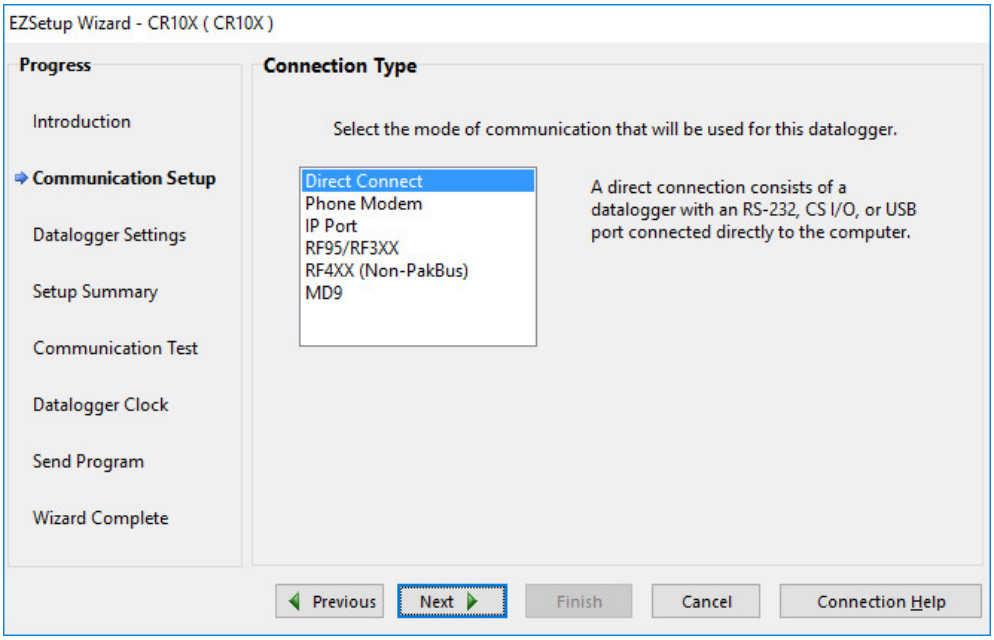
As you can see the window indicates several steps will follow. the process is started by simply pressing the 'Next' button at the bottom of the window. This takes us to the next screen.

We now enter the 'Communication Setup' stage. In this window we need to select what type of logger to use and name it.



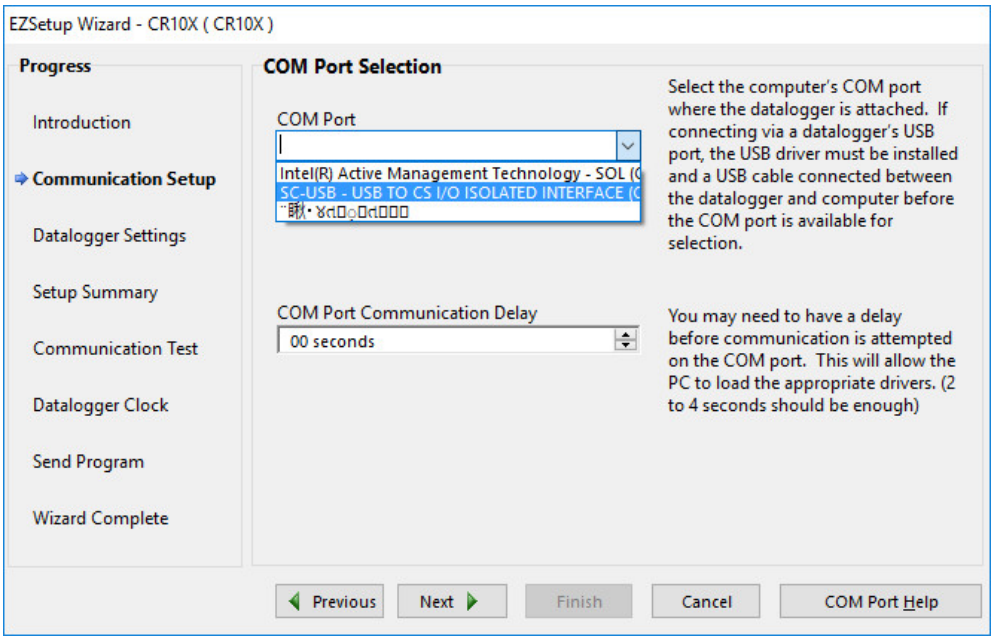
In this example I have chosen the CR10X logger and highlighted the default name 'CR10X'. I strongly recommend that you name your stations with a unique and understandable name. This will help you tremendously if you have many logger stations running in different places or running different tasks since everything associated to that station will be identifiable by the name. Once we have named the logger we an continue by pressing 'Next'.

In the new window you will see a series of options for the type of connection you are using to connect the computer to the logger.



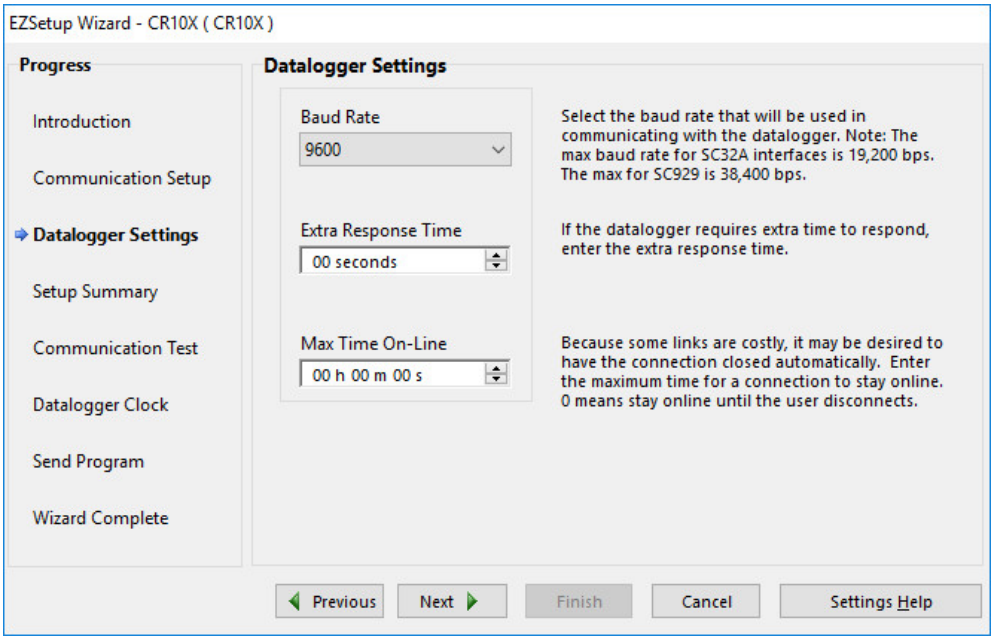
Only one option will be relevant here and it is the 'Direct Connect' which allows you to directly connect to the logger using an USB cable. Continue by pressing 'Next'.

The next step is to select which port is used by the computer for the communication.



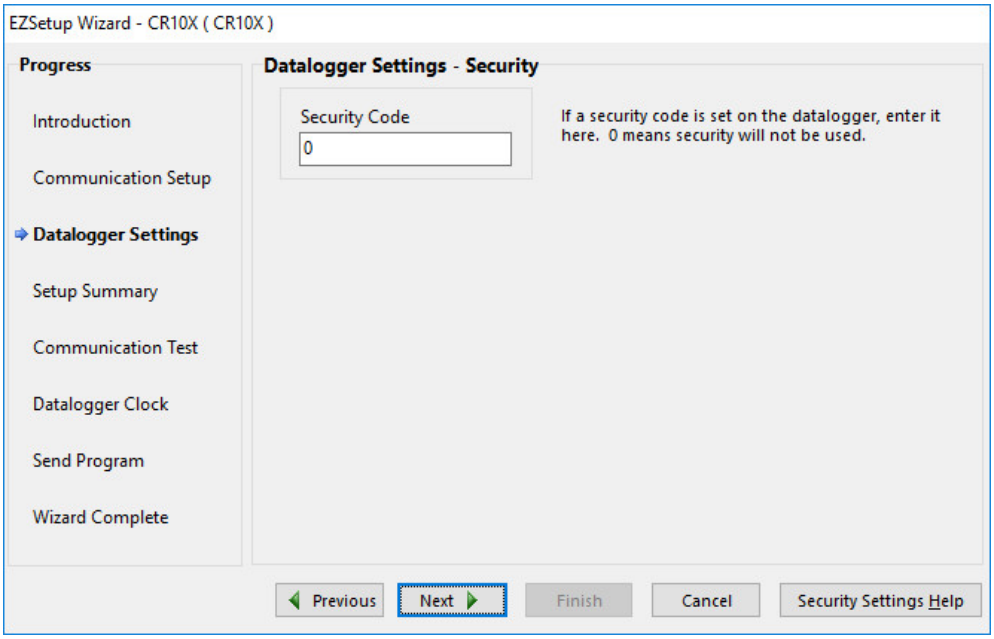
If you have installed the driver for the SC-USB interface and have connected the computer to a logger using this interface you should obtain one option looking like the example above. The SC-USB interface and its driver makes the computer believe one of the USB ports is a serial port, which is what is needed for the interface. Chose the SC-USB option and continue with 'Next'

In the next window you will see the data logger settings for communication.

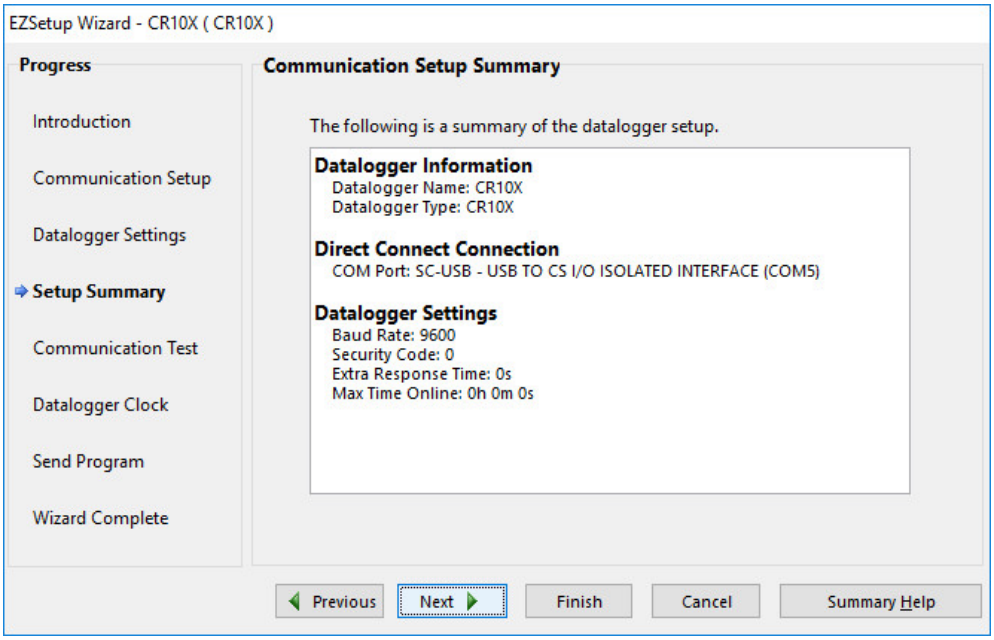


You should not change anything on this page unless the baud rate is higher than 9600, which is what the CR10X loggers use. Continue by pressing 'Next'

The next window concerns security setting. This is typically not used so press 'Next' to continue.

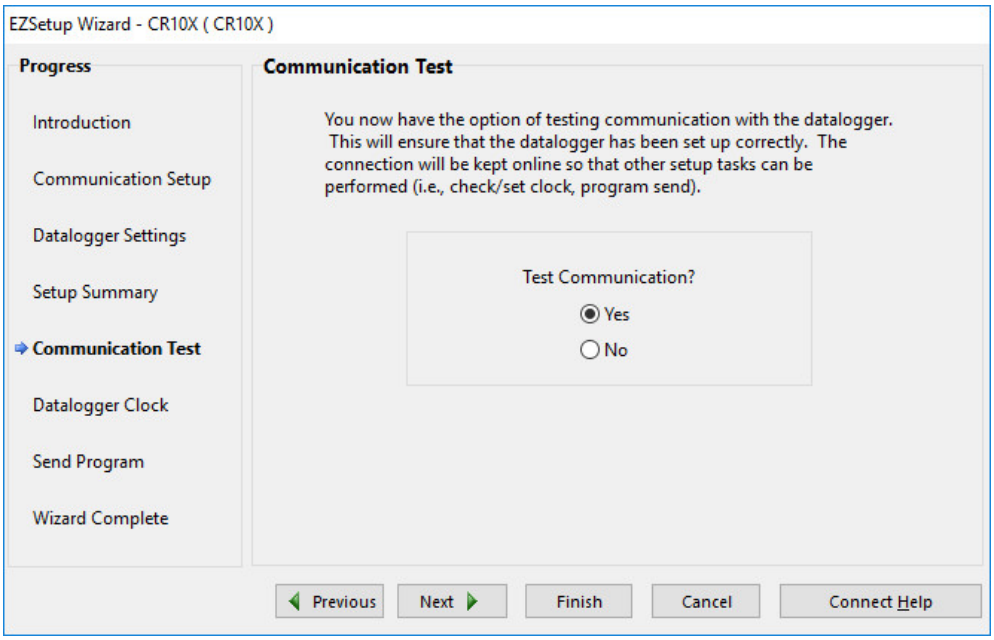


You now see a summary of all settings so far.



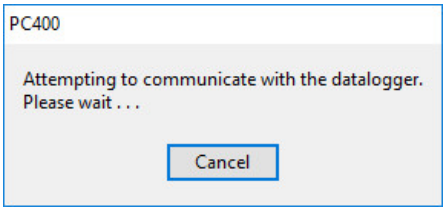
In this example you may note that the COM-port used by SC-USB has been named COM5. It is important to remember that this number is set by the software so it is not certain that the same port number will be used if you get back to the station later, particularly if you use a different USB port on the computer. You can always change the port settings if you find you cannot get a connection.

Once you have completed the setup you arrive at a window asking if you wish to text the communication.

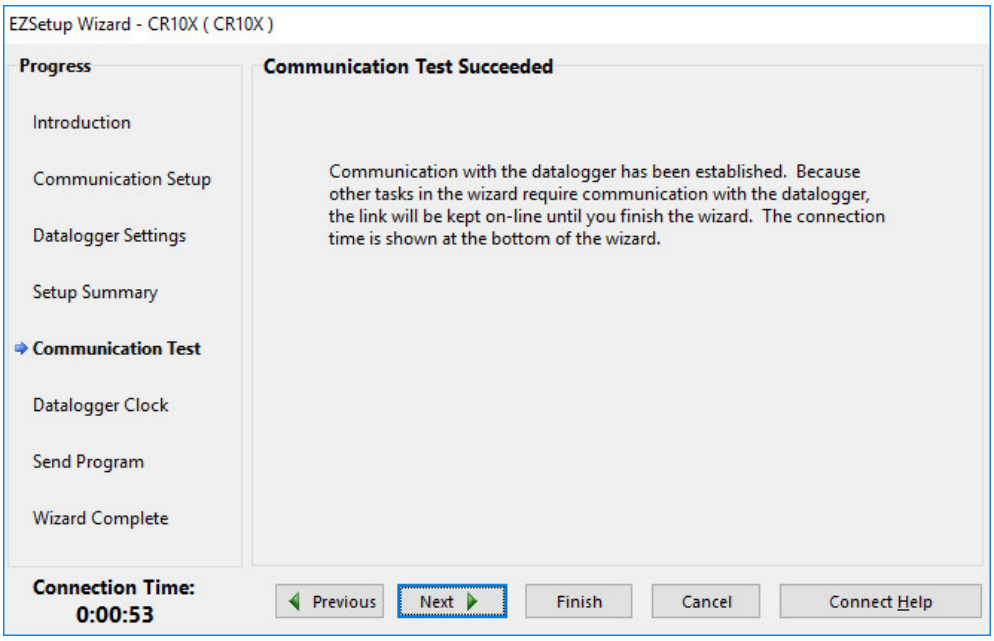


You can decide what you want to do but it is not a bad idea to test the communication. If so make sure 'Yes' is selected and press 'Next'

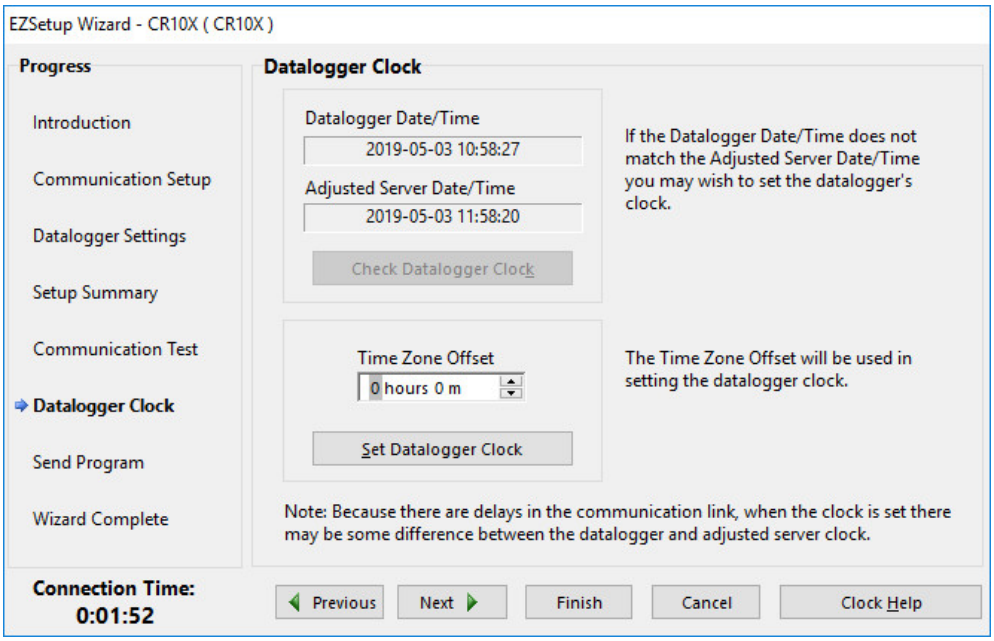
A small pane will open indicating that an attempt to communicate is occurring.



Once a connection has been established you will see the following screen indicating success. Note also that you now see a clock in the lower left corner indicating how long you have been connected. Press 'Next to continue.



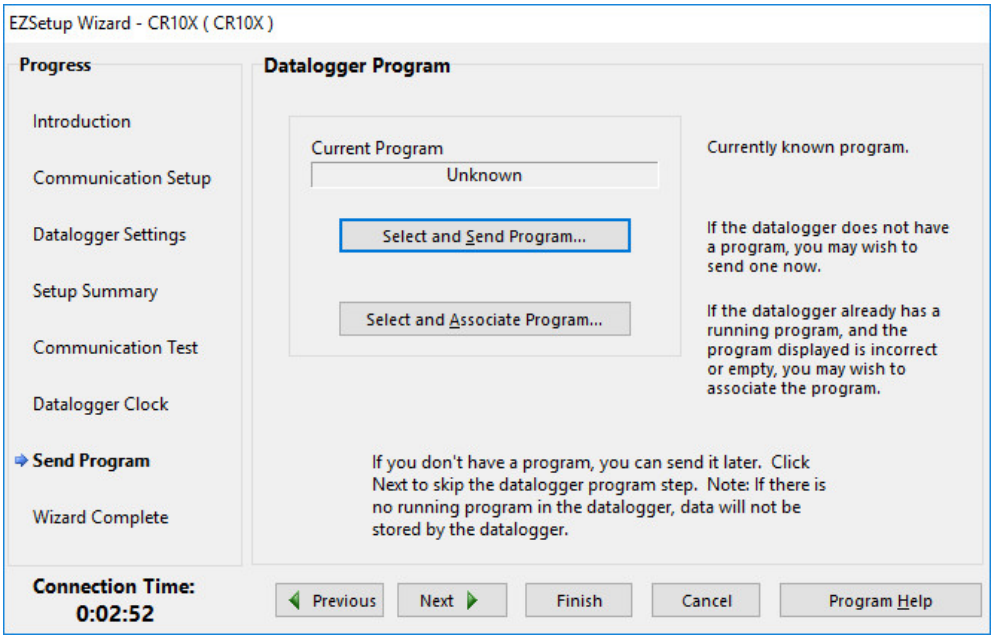
The Window that follows allows you to check the clock on the data logger.



You will see the clock on the data logger and the clock on the computer or 'server'. If the logger time is off you can adjust it to the computer time by pressing 'Set Datalogger Time'. The two fields should now show the same time.

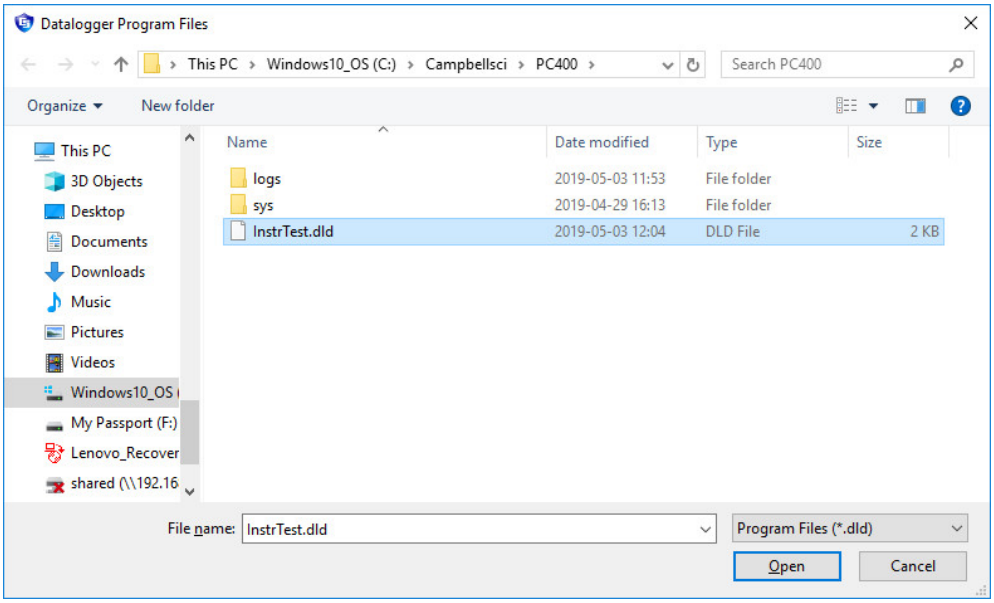
At this time it may be useful to make a comment about time. When logging data you want to use a standard time. Since we shift from standard time to daylight savings time and back, it is easy to make the mistake to set the logger time to daylight savings time if made during summer. All professional monitoring data runs on standard time throughout the year. this means you may inadvertently end up comparing data that is offset by one hour if you have set your logger to daylight savings time. Therefore always use standard time on the computer you use for working with loggers to make sure you always use the correct time in the logger.

Once the time has been set correctly you can continue to associate a logger program to the logger.



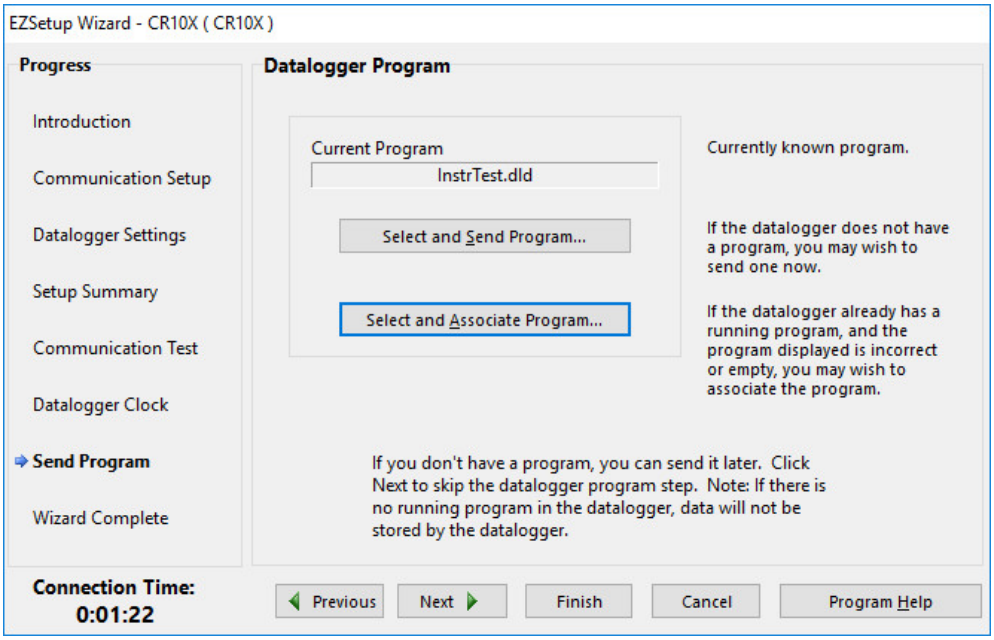
You have two options, either send a new program to the logger or just associate a program already on the logger to a file on the computer. This is necessary to allow variable name and other features in your logger program to be displayed correctly in the PC400 software.

When you press on the choice of action you will be able to select a program file that you have created for the logger station.

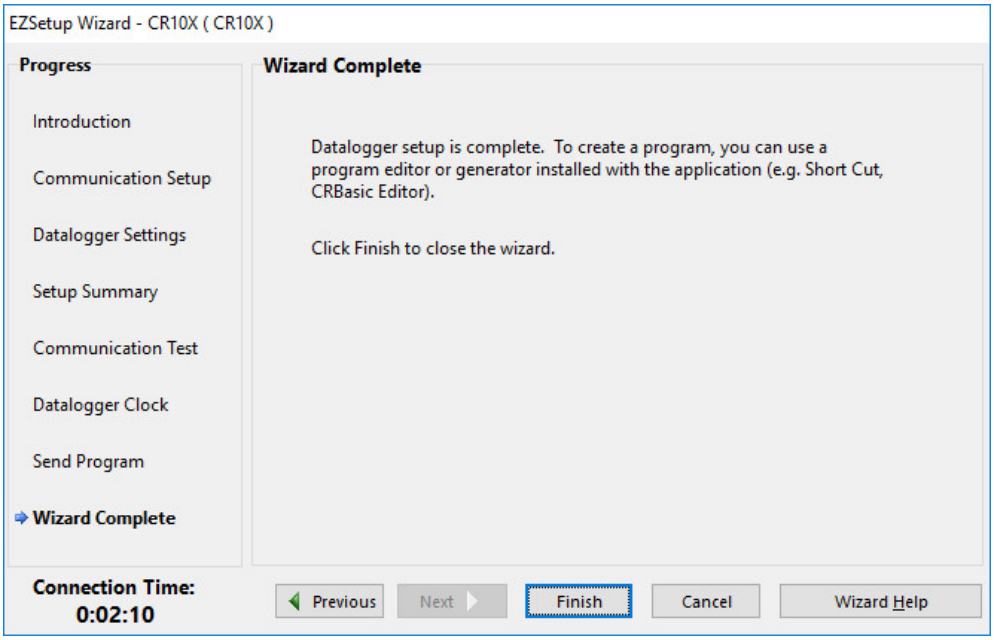


In this example, the program is called 'ÌnstrTest.dld'. The extension '.dld' indicates a program file containing a program that can be sent to a logger.

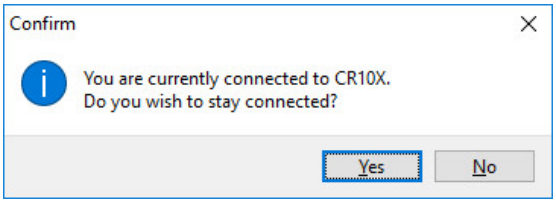
Once you have selected your program file, the file name will show in the 'Datalogger Program' window.



The final window shows that you have completed the setup and you can now press the 'Finish' button to end the setup.

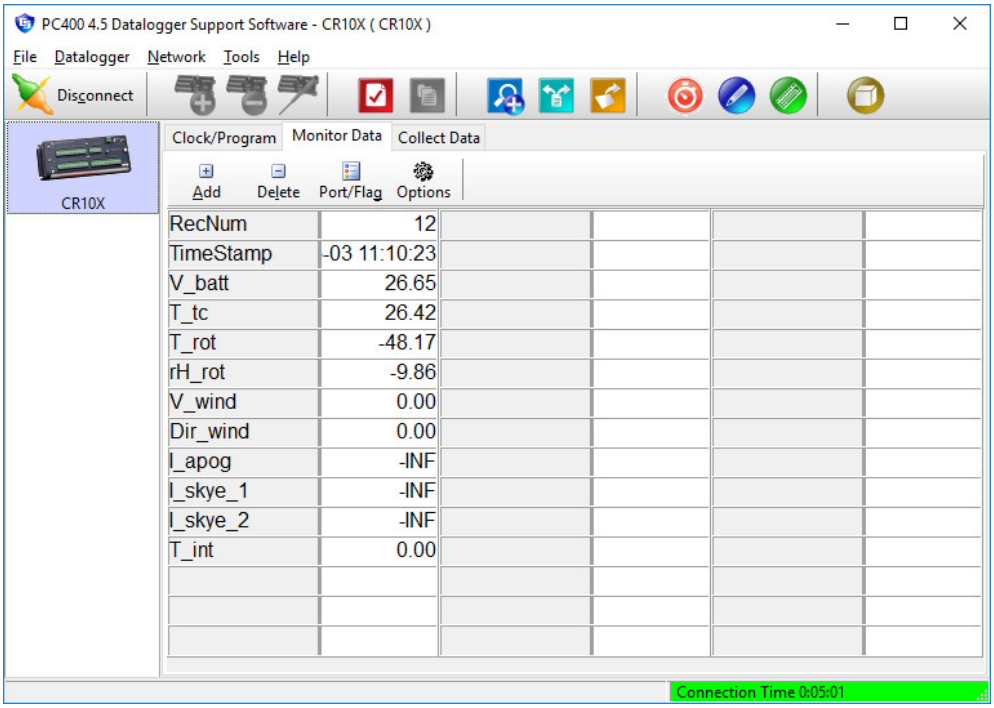


When you finish the procedure you will be asked whether or not you wish to remain connected to the logger. In our case we will continue working so we can press 'Yes'.



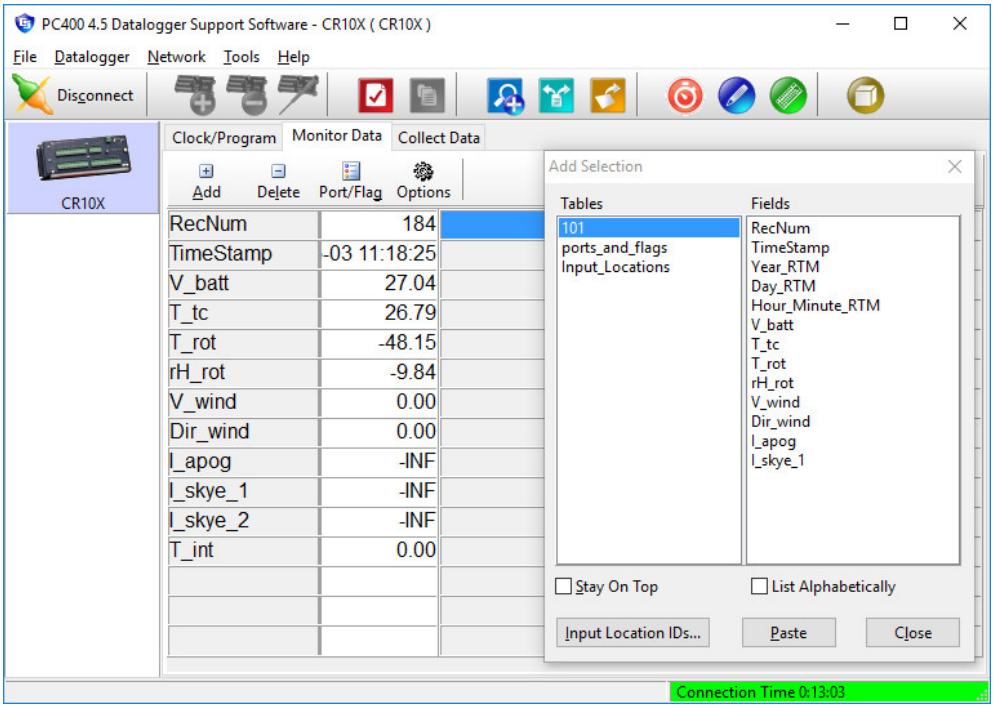
3.2 Monitoring data

PC400 allows us to monitor the data collected by a data logger while we are connected to it. In the main area of PC400 there is a tab called 'Monitor Data'.



This page will display the data recorded by the data logger. In the example we can see that a column of variable names are displayed and next to these are the values being collected by the logger. The variable names come from the logger program that we have associated with this logger and is an example why it is important to do so. If you look at the data in the example you see values of '-INF'. These are 'infinite' values caused by the fact that the sensors are not connected to the logger. Infinite values can also occur if a sensor fails and is hence a good indicator to check the sensor in question.

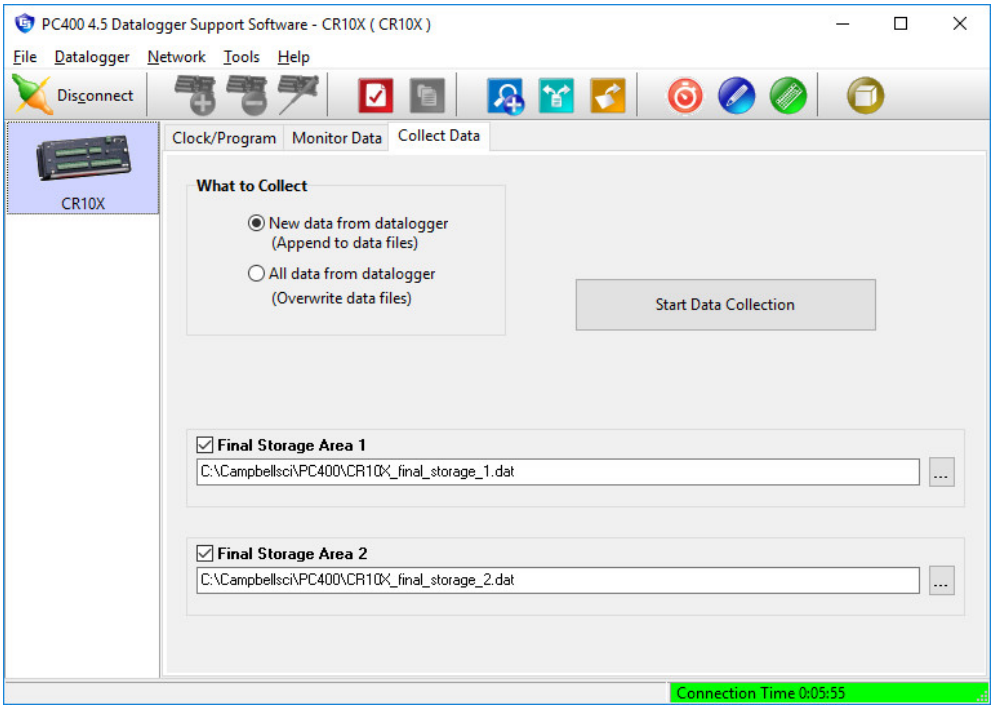
It is possible to edit the list of variables shown. This can be done by using the 'Add' and 'delete' buttons. To delete a field from view you simply mark it in the list and press the 'Delete' button. If you press the Àdd' button a new window will open showing what an be added.



In the left column we are given three choices. The first labelled '101' in the example is the data stored by the program. This will show the latest data the logger has stored to the data file you will retrieve from the logger. The second 'points_and_flags' will display flags that the logger uses for different purposes. This is not of major interest and will be useful only to advanced users. The third 'Input_locations' will show the data that is currently being measured by the logger. This is also what is shown in the example. The current measurements is usually what we want to see since it gives us quick indications of what is measured.

3.3 Collecting data

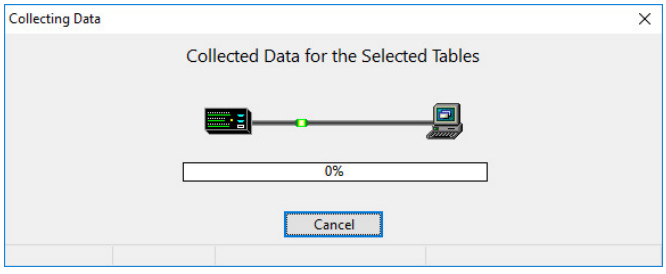
PC400 is used when we wish to retrieve data from a data logger. The tab 'Collect Data' in PC400 provides means to accomplish this.



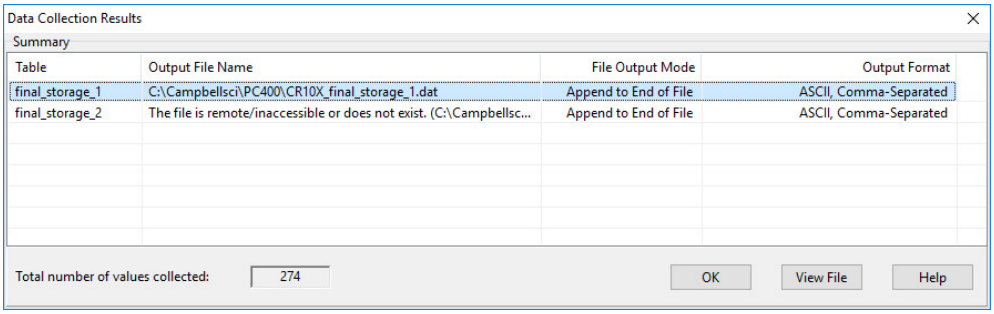
When you want to collect data you have to basic options. You can either collect all the data on the logger or collect only new data, i.e. data recorded by the logger since the previous time you collected data. Note that depending which you select will have an effect on the data files stored on the computer. By collecting only new data, the data will be appended to the file you may already have on the computer. If you collect all data, any file you may have will be overwritten and older content hence lost.

The data you collect will be stored in files given by the file names in the fields 'Final Storage Area 1' and 2. PC400 will assign a default name to the files but it is best to provide your own name, for example, reflecting the name of the logger station. Note also that by default PC400 will place the data files in its own working directory on the computer. It may also be useful to change this to a different location on the computer, perhaps a folder you have created in the 'Downloads' or 'Documents' folders.

When you want to collect data you simply press the 'Start Data Collection' button. This opens a new small window which indicates how data is being transferred.



Once the download is completed you will see a new window summarising the download and indicating how much data was downloaded. You can close this by pressing 'OK'.



Your data is now available in the data files specified above as comma separated values.

4 Programming

4.1 General programming

A data logger program contains instruction for measuring data and instructions for storing data for later data collection. In the following we will run through some and perhaps the most useful instructions used in regular logger programming. Note that it is by no means a comprehensive list. One basic issue when creating a program is the measurement strategy, answering questions such as

- what am I interested in knowing?
- how often should I measure?
- what data should I store?
- how often should I store data?

In terms of the logger programming, these questions mix and depend on each other. The most basic issue in the logger program is how often to measure the instruments. This can be done as many times as you wish (within practical limits). you can for example allow a simple program recording a temperature probe to measure every ten seconds. The first question, then, is: is it necessary to know the temperature that often? do you expect it to vary significantly over such a time period? The second question is how often do you save data? It is doubtful you wish to save every ten seconds (which is the shortest time interval you can use with the ten second run interval) since that will produce vast amounts of data. You may decide to save data every ten minutes and then chose to save the average over the last ten minutes. the logger will then collect all the 10 second values and automatically calculate the average for that period and save it for you. You could simultaneously save the current temperature (the single value just recorded) and save that at the same time. Making such judgement calls is the first issue facing the data logger programmer and it is necessary to have a strategy clear for how and why data should be collected. It may seem like a good idea to collect as much data as you can but that may soon change when you receive multi-megabyte data files and the logger runs out of battery very quickly (each run of the program consumes energy). In other words, there are trade offs. You need to optimise your program so that it collects enough data so that you can investigate what you need to without collecting redundant data that simply make the data difficult to handle. There are no right or wrong here, only a strategic way of thinking and adapting the logger to giving you what you need.

An issue often discussed among scientists is what you do inside a logger program. Some are proponents for leaving the data as near the raw measurement as possible and

do any calibration adjustments at a later stage in the office. The pros of this approach is that you get what you measure and no round off errors in the logger may interfere with your results. The cons include the fact that the numbers you can view are hard to interpret and spotting errors may be very hard. The other school of thought allows the logger to do most if not all of the adjustments inside the logger and saves data that is immediately interpretable. the pros and cons of this is of course the opposite of the first approach. An extra con for the first approach, and thereby pro for the second is that with only raw data being stored an uninitiated person will have no means of understanding what he or she is viewing in case you ask someone else to manage your loggers. Which school you adhere to is your business. It is perhaps clear, i belong to the second school.

Now let us go through some program instructions grouped according to their functionality.

4.2 Input/output instructions and associated instruments

There are several types of sensors that can become necessary to use but most sensors are built so that they change resistance with a change in forcing. this is true for thermistors, radiometers and some pressure transducers. Some sensors work by sending out pulses, examples are tipping bucket rain gauges and wind sensors. The data logger provides mean for measuring both a change in resistance (usually converted into a voltage through a Wheatstone bridge measurement, appendix 1) or by counting pulses over a specific time interval.

The data loggers provide a large number of so called *Input/output instructions* to allow for measurements of a wide range of instruments. The Input/output instructions are given program instruction numbers from 1 to 29 and 100 to 131. The latter range was added when the CR10s were upgraded to CR10X and hence not available on older models. Please refer to the CS data logger manual for details on all these instructions

With all sensors there may be a shield in the cable. This shield should be attached to AG to ground the sensor relative to the ground over which it is siting. This is not mentioned in the examples below but is a general rule. In the same spirit, you are advised to carefully read the sensor manual to intimately familiarise yourself to the sensor specific information. The manuals list cable colours and other specific details that may vary over time and therefore are best undocumented here. In all cases, **never, ever cut cables on sensors**. They can of course be re-calibrated by accredited institutes but at a cost to you.

4.2.1 Pt100 temperature probe

The Pt100 temperature probe is a standard for high accuracy temperature measurements. The probe changes its resistance with changing temperature. Thus, it is necessary to attach the sensor to the data logger in the form of a bridge measurement. this can be achieved in several ways. The simplest way is use a so called 3-wire half bridge measurement. The data loggers have a specific instruction for this, P7. the Pt 100 has four leads, two of each of two colours (usually white and red). connecting the sensor to the logger is straight forward and it does not matter which colour lead goes where as log as the same colours go to specific locations. We will therefore refer to leads as colour 1 and colour 2 rather than being specific. Attach one wire colour 1 to a single ended input corresponding to a Diff Low port and connect the other colour 1 wire to the corresponding Diff High port. Attach a 10 kΩ 0.1% precision resistor between an

excitation channel (Ex) port and the Diff High port. Then connect one 8or both) of the colour 2 wires to an AG port. This is now a 3-wire half bridge setup.

In the following example SE channels 1 and 2 (or Diff 1H and 1L) has been used together with the excitation channel 2. The programming uses the P7 instruction to measure the ratio of the voltage across the reference resistor Rf and the resistance across the Pt100, R0. The resulting ratio Rs/R0 is stored in location 22. The Pt100 typically comes calibrated from the factory. The calibration is valid for the intact sensor including its original length wires. Cutting the length of the sensor wire ruins the calibration and the temperature reading is no longer accurate. The multiplier constant of 100.1 is thus not given but can be obtained by running a program containing P7 with `mult = 1` in P7 and submerging the sensor in a ice-water bath at 0 °C. The resulting value in location 22 is the inverse of the constant mult.

```
; Measure R/R0 for Rt100
 1:  3W Half Bridge (P7)
 1: 1        Reps
 2: 33       25 mV 50 Hz Rejection Range
 3: 1        SE Channel
 4: 2        Excite all reps w/Exchan 2
 5: 2100     mV Excitation
 6: 22       Loc [ R_R0_T_1  ]
 7: 100.1    Mult
 8: 0        Offset

; Calculate T for Rt100
2:  Temperature RTD (P16)
 1: 1        Reps
 2: 22       R/R0 Loc [ R_R0_T_1  ]
 3: 1        Loc [ T_1       ]
 4: 1        Mult
 5: 0        Offset
```

4.2.2 Vaisala HMP45C temperature and humidity probe

The Vaisala temperature and humidity probe is measured as one Pt100 measurement, 3-wire half-bridge, and on differential measurement. The probe requires 8–30 V excitation which can be supplied either by attaching the power wires to the 12V and G ports or by using the switched 12V port. In the example below the sensor receives power from the Switched 12V. All constants are supplied by the manufacturer. In the example the temperature sensor is attached to single ended channel 3 (SE3) and AG ports and measured by P7 followed by the Temperature RTD instruction P16 resulting in the temperature value in °C located in input location 1. The sensor takes 2.1 V in excitation and outputs a result within –2.5–2.5 V. The humidity senor wires are connected to differential channel 3, DIFF 3H and 3L, and outputs the humidity in percent in input location 3

```
; Turn switxhed 12 V on
1:  Do (P86)
 1: 41       Set Port 1 High

; Measure Vaisala probe
2:  3W Half Bridge (P7)
```

```
 1: 1        Reps
 2: 33       25 mV 50 Hz Rejection Range
 3: 3        SE Channel
 4: 2        Excite all reps w/Exchan 2
 5: 2100     mV Excitation
 6: 12       Loc [ R_R0_2    ]
 7: 100      Mult
 8: 0        Offset

; Calculate temperature rom R/R0 values
3:  Temperature RTD (P16)
 1: 2        Reps
 2: 11       R/R0 Loc [ R_R0_1    ]
 3: 1        Loc [ Temp_1    ]
 4: 1        Mult
 5: 0        Offset

; Measure humidity
4:  Volt (Diff) (P2)
 1: 1        Reps
 2: 5        2500 mV Slow Range
 3: 3        DIFF Channel
 4: 3        Loc [ Rh        ]
 5: .1       Mult
 6: 0        Offset

; Turn of switched 12V
5:  Do (P86)
 1: 51       Set Port 1 Low
```

4.2.3 Rotronix HygroClip temperature and humidity probe

The Rotronix HygroClip sensor measure both temperature and humidity. The measurements are two differential voltage measurements. All constants are supplied by the manufacturer. The sensor requires 8–30 V excitation which can be supplied either by directly attaching the excitation wires to the 12V and G ports or by using the Switched 12V port. The following program then would be similar to the program for the Vaisala probe above in terms of switching the power on and off. In the example below the temperature sensor wires are attached to DIFF 3H and 3L and the humidity sensor to DIFF 4H and 4L.

```
; V E N T I L A T E D  T & Rh
; Measure temperature from ventilated
;  HygroClip sensor
1:  Volt (Diff) (P2)
 1: 1        Reps
 2: 35       2500 mV 50 Hz Rejection Range
 3: 3        DIFF Channel
 4: 3        Loc [ T_vent    ]
 5: .1       Mult
 6: -40      Offset
```

```
; Measure humidity from ventilated
;  HygroClip sensor
2:  Volt (Diff) (P2)
 1: 1        Reps
 2: 35       2500 mV 50 Hz Rejection Range
 3: 4        DIFF Channel
 4: 4        Loc [ rH_vent   ]
 5: .1       Mult
 6: 0.0      Offset
```

4.2.4 Young wind monitor

The young wind monitor is measured on the data logger using a combination of instructions, including a pulse count (P3) for the wind speed and a single ended voltage measurement for the wind direction. This latter measurement is of a type where the logger sends an excitation voltage, pauses for a brief moment and then makes the measurement, a so-called excite and delay measurement (P4). The constants are given by the manufacturer and are unique calibration constants. In the example below, the measured values of wind direction are corrected. The wind direction sensor has a ”blind spot” which should if possible be aimed at a direction where little wind is coming. this means that 0 may not point to north or any major direction and so corrections apply. These corrections can only be determined in the field by manually aiming the sensor to the north using a compass and then taking an offset reading with which to correct the original reading. In the example below, there is a correction of 6 degrees.

```
; Measure wind speed on Young Wind Monitor
1:  Pulse (P3)
 1: 1        Reps
 2: 1        Pulse Channel 1
 3: 21       Low Level AC, Output Hz
 4: 5        Loc [ Wind_spd  ]
 5: .098     Mult
 6: 0        Offset

; Measure wind direction on Young Wind Monitor
2:  Excite-Delay (SE) (P4)
 1: 1        Reps
 2: 5        2500 mV Slow Range
 3: 9        SE Channel
 4: 1        Excite all reps w/Exchan 1
 5: 2        Delay (0.01 sec units)
 6: 2500     mV Excitation
 7: 6        Loc [ Wind_dir  ]
 8: .142     Mult
 9: -135     Offset

; Make corrections to wind direction
3:  If (X<=>F) (P89)
 1: 6        X Loc [ Wind_dir  ]
 2: 4        <
```

```
 3: 0        F
 4: 30       Then Do

  4:  Z=X+F (P34)
   1: 6        X Loc [ Wind_dir  ]
   2: 360      F
   3: 6        Z Loc [ Wind_dir  ]

5:  End (P95)
```

4.2.5 SR-50 sonic ranger

The SR-50 Sonic ranger requires a little programming to produce an understandable output. Some of the programming in the example below can be performed as post-processing but then the number stored during measurement cannot immediately be checked against observation. The SR-50 measures the distance between the sensor and a surface, water, snow etc., by sending out an ultrasonic signal and measuring the time required for it to reflect against the surface and travel back to the sensor. The sensor is an advanced digital instrument which is measured using a specific instruction P105. The probe also needs a temperature taken from a separate temperature probe, in the example the temperature value obtained from a Rotronix HygroClip sensor. The general correction formula is $h = x\sqrt{T_K/273.15}$. the data logger does not have the capability to make this correction in a single step but must be programmed to calculate the correction in small steps. the first (program step 4) is to enter the difference between 0 °C and absolute zero temperature in °C (−273.15 °C). The next step (5) is to convert the measured temperature in °C to Kelvin. This is followed (6) by dividing the measured temperature by the reference temperature, that of freezing. Step 7 takes the square root of the result from step 6. This is followed by (8) multiplying the temperature correction with the measured value. In the last step the sensor value is subtracted from the elevation between the sensor and the ground at the start of measurements. This last step is necessary when measuring snow thickness because the sensor measures the distance between sensor and surface whereas the sought parameter is the distance between the ground and the snow surface or snow pack thickness. It is therefore necessary to have recorded the height between ground and sensor and have that entered in the program.

```
; S N O W  D E P T H,  S R - 5 0
1:  SDI-12 Recorder (P105)
 1: 0        SDI-12 Address
 2: 1        Start Measurement (aM1!)
 3: 1        Port
 4: 8        Loc [ SR_50     ]
 5: -1       Mult
 6: 0        Offset

; V E N T I L A T E D  T & Rh
; Measure temperature from ventilated
;  HygroClip sensor
2:  Volt (Diff) (P2)
 1: 1        Reps
 2: 35       2500 mV 50 Hz Rejection Range
```

```
 3: 3        DIFF Channel
 4: 3        Loc [ T_vent    ]
 5: .1       Mult
 6: -40      Offset

; Measure humidity from ventilated
;  HygroClip sensor
3:  Volt (Diff) (P2)
 1: 1        Reps
 2: 35       2500 mV 50 Hz Rejection Range
 3: 4        DIFF Channel
 4: 4        Loc [ rH_vent   ]
 5: .1       Mult
 6: 0.0      Offset

; Temperatue correcction to the SR-50 distance value
; dist = read * sqrt( T_Kelvin / 273.15 )
4:  Z=F x 10^n (P30)
 1: 273.15   F
 2: 00       n, Exponent of 10
 3: 9        Z Loc [ Ref_T     ]

; Convert measured air T from Celsius to Kelvin
5:  Z=X+F (P34)
 1: 4        X Loc [ T_HygClp  ]
 2: 273.15   F
 3: 10       Z Loc [ T_Kelvin  ]

; Divide the measured air T with the ref T
6:  Z=X/Y (P38)
 1: 10       X Loc [ T_Kelvin  ]
 2: 9        Y Loc [ Ref_T     ]
 3: 11       Z Loc [ Mult      ]

; Square multiplier value
7:  Z=SQRT(X) (P39)
 1: 11       X Loc [ Mult      ]
 2: 11       Z Loc [ Mult      ]

; Multiply depth value by multiplier
8:  Z=X*Y (P36)
 1: 12       X Loc [ SR_50     ]
 2: 11       Y Loc [ Mult      ]
 3: 12       Z Loc [ Snow_D    ]

; Add measured depth (neg) to sensor height to bare ground
9:  Z=X+F (P34)
 1: 12       X Loc [ Snow_D    ]
 2: 2        F
 3: 12       Z Loc [ Snow_D    ]
```

4.2.6 Tipping bucket rain gauge

The tipping bucket rain gauge is a simple sensor to measure. The gauge uses a hinged construction that tips whenever it fills with water, thus producing one pulse once a specific volume of water has accumulated. The sensor is measured using a P3 pulse measurement. The two wires from the gauge are connected to a pulse port and to G. In the example below the gauge has been connected to pulse port 2 (P2) and to a G port. A multiplier (0.016) converts each pulse to 0.016 mm precipitation. This multiplier is easily obtained by calibration in a lab. All that is needed is to slowly pour an exactly known volume of water into the gauge and record how many pulses are recorded to pass the entire volume of water. Each rain gauge type should have a characteristic catchment area that can be used to convert the volume of poured water into a specific precipitation value. By dividing the specific value by the number of pulses, the contribution by each pulse is obtained. This number can then be entered into the multiplier in P3.

```
; Measure tipping bucket rain gauge
1:  Pulse (P3)
 1: 1        Reps
 2: 2        Pulse Channel 2
 3: 2        Switch Closure, All Counts
 4: 8        Loc [ Precip    ]
 5: .16      Mult
 6: 0        Offset
```

4.2.7 Li200s Pyranometer

The Li200s pyranometer is measured using a straight forward single ended voltage measurement. The wires from the sensor is attached to the logger so that the signal wires make up a single ended configuration (one wire to a SE port and one to an AG port). The Pyranometer provides a voltage between 0 and 25 mV. The sensor comes with factory calibration and the multiplier 116.55 converts the output voltage into $\mathrm{W\,m^{-2}}$.

```
; Measure Li200s Pyranometer
1:  Volt (SE) (P1)
 1: 1        Reps
 2: 33       25 mV 50 Hz Rejection Range
 3: 10       SE Channel
 4: 7        Loc [ Li200S    ]
 5: 116.55   Mult
 6: 0        Offset
```

4.2.8 The CS100 Barometric pressure

The CS100 barometric pressure sensor is based on a straight forward single ended voltage measurement. the sensor is connected to the data logger in a single ended configuration. The sensor, however, must be turned on at least one minute before measurement in order to obtain a trustworthy value. The following program example is set up to accomplish this by running two if time instructions. the first one checks if the time is 1 minute before the hour, expressed as 59 minutes into a 60 minute interval (more on processing instructions below). If this is true then it sets port 8

high, otherwise does nothing. The next step checks if time is at the even hour when measurements should be done (expressed as 0 minutes into a 60 minute interval). If this is true then the loop containing instruction P1 (single ended measurement) is carried out followed by an instruction P86 turning port 8 low again. This ensures that the sensor is on one minute before measuring and that it is turned off immediately afterwards. Note that the execution interval of the logger program must be such that it allows the program to be scanned every minute (60 second interval set) for this program example to work. If program step 1 never becomes true, the sensor will never be measured accurately and the result will be garbage data.

```
; B A R O M E T R I C  P R E S S U R E
1:  If time is (P92)
 1: 59       Minutes (Seconds --) into a
 2: 60       Interval (same units as above)
 3: 48       Set Port 8 High

2:  If time is (P92)
 1: 0        Minutes (Seconds --) into a
 2: 60       Interval (same units as above)
 3: 30       Then Do


     3:  Volt (SE) (P1)
      1: 1        Reps
      2: 15       2500 mV Fast Range
      3: 11       SE Channel
      4: 11       Loc [ P_mb      ]
      5: 0.2      Mult
      6: 600      Offset

     4:  Do (P86)
      1: 58       Set Port 8 Low

5:  End (P95)
```

4.3 Processing instructions

Once we have the measurement section of a program done we may need to manipulate the data to convert the measured signal to something meaningful such as a temperature or to add a temperature correction to another sensor value. The data loggers provide a large number of so-called *Processing instructions* to accomplish this. You will find a range of such instruction in program instructions P30 through P68. There is no reason to run through their function here. Some have already occurred in the examples above. Please refer to the CS data logger manual for details on all these instructions.

4.4 Output processing instructions

The output processing instructions are the important instructions that take sensor output and saves data to final storage, where you can obtain it for analysis and plotting. The Output processing instruction are found as program instructions P69 through P82. We will briefly touch on some of the more common of these here Note that output

processing can be determined to occur at specific times. we will look at that further down.

4.4.1 P69 Wind vector

The P69 Wind vector takes the data gathered by the wind monitor instruction (see above). In the example below, it will take the data recorded by the wind speed pulse instruction and the wind direction voltage measurement and output mean horizontal wind speed, unit vector mean wind direction, and standard deviation of wind direction. There are other forms for saving the same data, so please refer to the data logger manual for the other options. executing this output processing command thus results in three numbers stored for posterity.

```
; Store wind speed, dir and std dev
27:  Wind Vector (P69)
 1: 1        Reps
 2: 1        Samples per Sub-Interval
 3: 0        S, theta(1), sigma(theta(1)) with polar sensor
 4: 5        Wind Speed/East Loc [ Wind_spd  ]
 5: 6        Wind Direction/North Loc [ Wind_dir  ]
```

4.4.2 P70 Sample

The P70 sample instruction takes the current measured (single) value and saves that to final storage. In the example below, 4 temperature sensors are stored in one step by using the repetitions (Reps) variable. It is possible to repeat the instruction four times without rewriting it by using this feature. The only caveat is that the values must be in four consecutive input locations, in this case locations 1–4.

```
; Store current unvent and vent T and Rh
25:  Sample (P70)
 1: 4        Reps
 2: 1        Loc [ T_1       ]
```

4.4.3 P7 Average

The P7 average instruction takes all measurements gathered since the last time the output processing command was issued and calculates the average of all values during that time period and saves that to final storage. In the example below, the average value for 1 temperature sensor is stored.

```
; Store current unvent and vent T and Rh
26:  Average(P71)
 1: 1        Reps
 2: 1        Loc [ T_1       ]
```

4.4.4 P72 Totalize

The P72 Totalize command sums up values gathered between the times output processing occurs. In this example we sum up all tipping bucket pulses to obtain the total volume of rain gathered over a time period.

```
; Store hourly precipitation
30:  Totalize (P72)
 1: 1        Reps
 2: 8        Loc [ Precip    ]
```

4.4.5 P73 Maximize and P74 Minimize

The P73 Maximize and P74 Minimize instructions are closely related and searches for the maximum and minimum values that have occurred in the period between output processing events. Both commands store the high or low value together with the hour-minute at which it occurred. Note that it is impossible to capture anything at a finer scale than the execution interval of the program itself. if it is important to find some very transient value, the program must be set to execute very often.

```
; Store max vent T
37:  Maximum (P73)
 1: 1        Reps
 2: 10       Value with Hr-Min
 3: 3        Loc [ T_vent    ]

; Store min vent T
38:  Minimum (P74)
 1: 1        Reps
 2: 10       Value with Hr-Min
 3: 3        Loc [ T_vent    ]
```

4.4.6 P77 record Real Time

The P77 record real time instruction is essential for all output processing since it allows you to save the year, day and time (even in seconds) when the data was saved to final storage. this is thus the time stamp for your data and MUST be included whenever you save data to final storage. The instruction is very simple with only one input, but one which requires some further explanation. As mentioned the instruction saves the time stamp. Since the time consists of four parts, the year, the Julian Day, the hour-minute value and seconds, the instruction has four digits to control which of these are saved. The first digit corresponds to year, the second with day and so on. If you set a digit to zero the information corresponding to the position of the zero is not saved. As an example 0110 saves day and hour-minute (ones) but not year and second (zeros). In addition you have the option to have midnight represented by 0 or by 2400 hours and also whether the Julian Day number changes at or after 0 (2400) hours. These settings can be signalled by entering a 2 in the day and hour-minute position. This is only a matter of convention and matters when you post-process your data. It is best to decide on one model and then stick to it to avoid having different time stamps for different stations.

```
23:  Real Time (P77)
 1: 1220     Year,Day,Hour/Minute (midnight = 2400)
```

4.5 Program control instructions

The program control instruction are necessary to apply logical structure to the program. examples are to make a choice whether to measure or not depending on what has been

measured by another sensor or simply to execute some instruction depending on time. You will find a range of such instruction in program instructions P83 through P98, P111 and P120 through P123. There is no reason to run through their function here. Some have already occurred in the examples above. Please refer to the CS data logger manual for details on all these instructions. we will focus on a few often used control instructions.

4.5.1 P86 Do

The P86 Do instruction simply forces the logger to execute a specific command. The set of commands are given in the CS data logger manual. Please refer to this for more information.

```
19:  Do (P86)
 1: 58       Set Port 8 Low
```

4.5.2 P92 If Time

The P92 If Time instruction is another essential instruction for saving data to final storage. This instruction checks if a desired time is true. In the example below the instruction checks if time is zero minutes into a 60 minute interval. If this is true then it sets the so-called output flag high, which signals to the logger to execute all instructions after this one. This is a roundabout way of saying that the time should be on the hour exactly for out put to occur. From this we learn that all output processing instructions should be positioned after a P92 that determines when they are executed. It is possible to have several P92 If time in a program. Please refer to the Tarfala meteorological station program reproduced in Appendix 2 for an example of multiple outputs.

```
21:  If time is (P92)
 1: 0        Minutes (Seconds --) into a
 2: 60       Interval (same units as above)
 3: 10       Set Output Flag High (Flag 0)
```

4.6 Concluding remarks on programming

The possibilities to program the CR10X are practically endless. Not only is it possible to make a wide variety of measurements, it is also possible to use the logger as a control unit to run other equipment. For most purposes the loggers are used as data collectors. A typical program consists of two basic parts. The first part should contain all the measurement instructions. Intermixed with these may be data manipulating instructions such as those described for the SR-50 above. This section of the program should collect all the data and have the data stored in input location in the form necessary for final storage. The second part of the program contains all the output instructions. These are most often but not necessarily preceded by an instruction checking if the time is right to save data. Such an instruction is then followed by instructions saving the collected data in the form and order desired. Understanding this structure is a good start for good programming.

5 Logger software

The editor for creating logger programs is called *EdLog* and allows you to program any sort of program for any of the CS loggers. To learn the EdLog program you need

to reference the PC400 manual. I will only briefly outline what you can expect from EdLog.

When you start Edlog to create a new program you are asked to pick the correct logger type. It is important to select the correct logger type since they differ in terms of the types of instructions available for programming and their measurement range. Edlog then opens a new program. you will see the following

```
;{CR10X}
;
*Table 1 Program
  01: 0.000      Execution Interval (seconds)

*Table 2 Program
  02: 0.000      Execution interval (seconds)

*Table 3 Subroutines

End Program
```

You start the programming by entering the desired execution interval in seconds at **01: 0.000**. The interval tells the logger how often the program should be run. You then put the cursor on the empty line after where you specified the execution interval and enter the program instruction number. As an example I have enter program instruction 1. EdLog responds by inserting all necessary parameters for P1 as follows:

```
;{CR10X}
;
*Table 1 Program
  01: 60        Execution Interval (seconds)

1:  Volt (SE) (P1)
 1: 1          Reps
 2: 00         Range Option
 3: 00         SE Channel
 4: 0000       Loc [ _________ ]
 5: 1.0        Mult
 6: 0.0        Offset

...
```

You now continue to enter each of the parameters, exactly what you need to enter depends on what you will be measuring. The rest of the program is added step by step by setting the cursor on the empty line after the last instruction, or, if you need to insert something in the middle of the program, on an empty line between two instructions where the new instruction should go. It is quick to build up a program this way. One nice feature of this software is that you can enter a name for the measured entity on the line shown as `Loc [ _________ ]`. Edlog automatically assigns an input location to the name (in chronological order). This is convenient since it means you only need to keep track of the name of the variable, not the exact location. Let us say we call the variable above `T_Air` for air temperature. later in the program when we need to save a sample using P70, we only need to type in `T_Air` and EdLog knows what input location to look for the value to be saved. The only disadvantage with this feature is

that measurements end up in the order in which they are measured. Edlog provides a means for reorganising their order called **InLoc Ed** (Input location editor). pressing the 'InLoc ed' button opens a window listing all variables and used input locations. You can move the names around so the the variables end up in the order you wish. The main point of this may be to organise the data so that you see the most important numbers first when viewing he data with 'star-six' mode in the field. please refer to the EdLog section of the LoggerNet manual for details.

6 Field maintenance of loggers

Collecting data is of course an important activity. CS provides several methods to do this. you can for example use a laptop running LoggerNet

6.1 Switching batteries on CS loggers

As with making any changes to a logger, download data and program before you switch batteries. The CR-10X model has an internal battery that keeps data, program and internal clock alive when disconnecting the battery. The CR-10X logger simply stops logging. The older CR-10 models do not have such an internal battery and will therefor lose everything that is in memory when disconnecting a battery. Therefore always take the precaution to download both data and program before risking anything with the logger.

Here is a suggested step by step instruction on how to proceed

1. Connect the laptop to the logger and start the Campbell software
2. In the software, identify the station that you are visiting. Or, if it does not exist, create a new station and connect to it
3. Move to the 'Collect data' tab and press the 'Start Data Collection' tab.
4. Once the data transfer is complete, disconnect the station in the software
5. Change the battery
6. You may want to connect to the logger again to check that it is running as it should

The same working procedure is necessary when changing the program or time on the logger. ALWAYS, collect data before making any changes to the logger time or program. If you do not, loss of data may occur.

6.2 Important note about time

Whenever visiting a remote station that is not frequently visited, ALWAYS take note of the time in the logger and compare it to your own clock (assuming it is showing correct time). Change the time if it differs but make a note of time before and after changing. Again, download any data before you change the time. Always keep loggers on standard time, NEVER on daylight savings time! Keeping logger on correct time is crucial when making data analysis. It is also extremely important to make clear notes on any changes made to the time on a logger. It should be possible to apply a correction to a logger that is losing its synch if we know that it was synched when it was started and if we know the size of the time shift at a given time later. So, ALWAYS CHECK AND ADJUST TIME ON LOGGERS AND NOTE THE TIME DIFFERENCE.

7 Working with the CR10KD keyboard

When starting work with a data logger there are a certain number of issues that must be familiar. Let us start with an unpowered logger with an attached CR10KD keyboard attached. The data logger requires a 12 V power source, commonly in the form of a 12 V sealed lead acid battery (SLAB). The logger has a connector to which +V (usually red on the battery) and –V (usually black on the battery) should be attached. If you prepare a battery cable for the logger yourself, it is convenient to stick to the red+/black– standard. When you connect the power, the data logger will respond by showing the word **HELLO** followed by a number indicating the number of kb of memory available. Press ⋆0 (zero) and the display will show the message **LOG**, this is the main display state of the logger (if the logger already contains some program, it may display **LOG 1**.

There are ten basic commands that are used for basic handling of the logger, these are collectively known as star (⋆ commands). We will now run through them in order, but first a brief over-view and summary (you will also find this on the logger cheat-sheet).

⋆0 By pressing 'star-zero' you will return to the main logging mode from any other command. If you for example are programming, 'star-zero' compiles the program and returns the logger to logging mode.

⋆1 'star-one' takes you into program table one, which is the main location where you store your logger program.

⋆2 'star-two' takes you into program table 2 where a second program can reside, performing entirely different tasks than program one. It is not likely you need to use program table two.

⋆3 'star-three' takes you into program table three, which is dedicated to subroutines only. These are routines that perform specific tasks that are callable from program tables one and two. It is even less certain that you will use program table three than two.

⋆4 'star-four' takes you into the Parameter entry table. Here you can manually enter or change certain parameters that your programs can access and use.

⋆5 With 'star-five' you enter the logger clock. you can then change the year, date and time on the logger to suit your needs. Note that an old 21X or CR10 starts with time 0 if power has been lost, whereas the CR10X maintains the time in memory.

⋆6 'star-six' allows you to view data as it is measured by the data logger. This data is in what is called *input location* and is stored by the logger program awaiting program instructions to save data to its final destination (*final storage*).

⋆7 The 'star-seven' allows you to browse data that has entered into what is called *final storage*, which is data that you can retrieve from the data logger.

⋆8 'star-eight' allows you to move data from final storage to an external storage module. It is a feature you are unlikely to need.

⋆9 'star-nine' allows you to enter commands for external storage modules. This is not a feature you are likely to use.

Besides these 'star'-commands, there are the ⋆A through ⋆D, these commands deals with logger memory allocation and security and should not be touched unless you really know what you are doing. Entering into these commands may cause data loss so be warned! We will now run through the most useful 'star'-commands except ⋆0 since its use is trivial. In most 'star'-modes, the logger will automatically revert back to **LOG**-mode if no keys are pressed for some time. This is true for all program table modes but not for 'star-six' mode, input location.

There are a number of important key-combinations we need to familiarise ourselves with before continuing with the 'star'-commands these involve entering and changing parameters in the logger (tab. 1). The letter A is used to ***a**dvance one step* in an instruction, B is used to ***b**ack up one step*. C is used to ***c**hange sign* on a number, when, for example, entering a negative constant in the program. D is used to to enter a ***d**ecimal point*. The #-sign acts as the backspace key on a computer and deletes the last digit entered (before advancing with A). #A allows you to move forward (advance) trough the program one program step at a time, #B allows you to back up one step. #D allows you to ***d**elete* an entire program step. These three are of use only in program tables 1–3. finally the digits 1–9 are used to enter numbers into the program tables.

Table 1. *Summary of editing functions.*

Command	description
A	Enter/Advance
B	Back up
C	Change sign
D	Decimal point
#	Clear the rightmost digit
#A	Advance one program instruction
#B	Back up one program instruction
#D	Delete one instruction
0--9	Enter numbers 0–9

7.1 ⋆5

It is important that loggers run on exact time when monitoring environmental parameters since once data set might need comparison with others. To be able to check and adjust time is therefore and important task which is accomplished by the ⋆5 command.

When you enter ⋆5, the logger responds by showing **:hh:mm:ss** (hh = hour; mm = minute; ss = second). This is the current time in the logger and you see the time display is active since it will be counting on. Press A to advance to the second step, to enter the year. The display shows **05:00**. You can now enter the year as 2008, followed by A to advance to the third step, to enter the Julian day. The display shows **05:0000**. You should now enter the Julian day corresponding to the current date. Campbell provides a table to simplify this. the Julian Day system starts with January 1 as Julian Day 1 and ends with December 31 as Julian Day 365 or 366 if the year is a leap year. Press A to continue to the fourth step, to enter the time of day. The display now shows **05:0000**. The time is given as a four digit number with the form *hhmm*, where *hh* is the hour 00–24 and *mm* is the minute 00–59. Note that you need to enter a four digit number so 5 minutes past 1 at night is *0105*. Press A to continue. The display will now show your new corrected time as **:hh:mm:ss**. Note that when you press A, the clock starts at exactly the *hhmm* your entered. This might be important if second accuracy matters in your measurements. Finish by pressing ⋆0 to accept all time changes and o back to the basic setting of the logger. You have now completed entering the correct time into the logger. This should be one of the first things you do when starting a logger project.

7.2 ⋆1

In order to use the logger you need to enter the program. we will look into the details of programming further down but will run through some basics here. Press ⋆1 to enter into program table where your main program should be located. The display shows **01:00** indicating that you have entered program table one. Press A and the display changes to show **01: 0.0000**. You should now enter the execution step or interval for the program. this is a value in seconds. If you want the program to be executed every ten minutes to take measurements, you enter 600 (seconds) and press A. The display now shows **01:P00** which indicates that you are ready to enter a program instruction into the first step of the program table. You can now enter a program code *Pnn*, where *nn* is a number you enter to enter a specific instruction into the program. These instructions are given in the manual and refer to specific measurements or other steps necessary to get a functioning program. For the sake of exercise we will enter a very simple program that records the battery voltage and saves this value together with information on what year, day and hour minute the measurement was done. at the display **01:P00** press 10 followed by *A* to enter program instruction number ten, which is *measure battery*. The display now shows **01:0000**. this field is an instruction telling the logger into which input location to put the value just recorded. Let us put it in location 3. Press 3 followed by A. The display now shows 02:P00 indicating that you can now enter the second instruction into the program. In this case we want the program to save the number. This is done by setting a so called *flag* 'high'. Output is made when flag 0 is set high which is done by program instruction 86. At the display **02:P00** press 86 followed by A. The display then shows **01:00**. Press 10 followed by A and the display shows **03:P00**. you have now set flag 0 high and is ready to enter the third program step. This is to save the year ,day and time to the file. This is accomplished by instruction 77 called 'Real time'. Press 77 followed by A and the display shows

01:0000. You can now enter a code that tells the logger what to save. To save the year, day and hour-minute, you enter `1110` followed by `A`. Next step is to actually save some data. The logger programming allows for several possibilities but here we want to save the value we just measured. This is accomplished with instruction 70 'sample'. At the display now showing **04:P00** press `70` followed by `A`. The display shows **01:0000**. We can now enter something called repetitions. We only have one value to save so we enter `1` followed by `A`. The display shows **02:0000**. We now have to specify the input location where we stored the measured value, which was location 3. Press `3` followed by `A`. The display now shows 05:P00, ready for another program step to be entered, but our program is finished so we finish by pressing `⋆0` to compile the program. The logger window may go blank for a short while but soon returns **LOG 1** indicating we are now executing program table 1. The program is now running and collecting a sample of the battery voltage every 10 minutes and saving this value together with year, day and hour.

7.3 ⋆6

Once a program is running it is interesting to view the data being collected. One reason is to double check that data is actually recorded in a reasonable way and that numbers make sense. This can be checked by entering the 'star-six' mode ,which allows you to view the values stored in input location. Press `⋆6`, the display responds by showing **06:0000**. Press `A` and the display shows **01: 0.0000**. this is input location 1. in our program we stored our battery voltage in input location 3 so we need to move on to the third input location. this is done by repeatedly pressing `A` until the display shows **03: xxxx**, where **xxxx** is the value measured. Since we use a 12 V for loggers we probably get a value of over 12 V. In my case the display shows **03: 12.432**. You can move around in input location using `A` for advance and `B` for back up as you wish.

7.4 Concluding remarks on basic handling

We have now established how to set the clock in the logger, how to enter a simple program that contains a measurements, instructions to the logger to make output and then save time and the value to final storage, the data you will later collect and use. These principles apply to all programs. Your programs will necessarily become more complicated as you need to add sensors and manipulate the data from the sensors in order to save useful numbers. We will look into some such aspects in he next section but this manual is in no way a replacement for the Campbell scientific manual and all details given there. Many programming examples are often accompanying sensors purchased for use with the logger. this means you usually only need to adapt an already given program to suit your exact needs. In the following chapter we will take a deeper look at what can be done in programming a logger for different tasks.

A Appendix: The Tarfala Research Station meteorological station logger program

This appendix contains the current (at the time of writing) logger program running the Tarfala Research Station meteorological station. The program includes instructions not covered by this manual. The basic structure of the program is to first make all measurements, then make output in three forms, hourly, daily and synoptic (3 hour) outputs.

```
;{CR10X}
*Table 1 Program
  01: 10.0000   Execution Interval (seconds)

;-----------------------------------------
; Check battery voltage
; and stop execution if lower than 9.7V
1:  Batt Voltage (P10)
 1: 10       Loc [ Battery   ]

2:  If (X<=>F) (P89)
 1: 10       X Loc [ Battery   ]
 2: 4        <
 3: 9.7      F
 4: 0        Go to end of Program Table

;-----------------------------------------
; A I R  T E M P E R A T U R E
; Measure R/R0 for old met cage Rt100
3:  3W Half Bridge (P7)
 1: 1        Reps
 2: 33       25 mV 50 Hz Rejection Range
 3: 1        SE Channel
 4: 2        Excite all reps w/Exchan 2
 5: 2100     mV Excitation
 6: 22       Loc [ R_R0_T_1  ]
 7: 100.1    Mult
 8: 0        Offset

; Meaasure R/R0 for Young screen Rt100
4:  3W Half Bridge (P7)
 1: 1        Reps
 2: 33       25 mV 50 Hz Rejection Range
 3: 3        SE Channel
 4: 2        Excite all reps w/Exchan 2
 5: 2100     mV Excitation
 6: 23       Loc [ R_R0_T_2  ]
 7: 100.2    Mult
 8: 0        Offset

; Calculate T for both Rt100
5:  Temperature RTD (P16)
 1: 2        Reps
 2: 22       R/R0 Loc [ R_R0_T_1  ]
 3: 1        Loc [ T_1       ]
 4: 1        Mult
 5: 0        Offset

;-----------------------------------------
; V E N T I L A T E D  T & Rh
; Measure temperature from ventilated
;  HygroClip sensor
6:  Volt (Diff) (P2)
 1: 1        Reps
 2: 35       2500 mV 50 Hz Rejection Range
 3: 3        DIFF Channel
 4: 3        Loc [ T_vent    ]
 5: .1       Mult
 6: -40      Offset

; Measure humidity from ventilated
```

```
;  HygroClip sensor
7:  Volt (Diff) (P2)
 1: 1        Reps
 2: 35       2500 mV 50 Hz Rejection Range
 3: 4        DIFF Channel
 4: 4        Loc [ rH_vent   ]
 5: .1       Mult
 6: 0.0      Offset

;-----------------------------------------
; W I N D
; Measure wind speed on Young Wind Monitor
8:  Pulse (P3)
 1: 1        Reps
 2: 1        Pulse Channel 1
 3: 21       Low Level AC, Output Hz
 4: 5        Loc [ Wind_spd  ]
 5: .098     Mult
 6: 0        Offset

; Measure wind direction on Young Wind Monitor
9:  Excite-Delay (SE) (P4)
 1: 1        Reps
 2: 5        2500 mV Slow Range
 3: 9        SE Channel
 4: 1        Excite all reps w/Exchan 1
 5: 2        Delay (0.01 sec units)
 6: 2500     mV Excitation
 7: 6        Loc [ Wind_dir  ]
 8: .142     Mult
 9: -135     Offset

; Make corrections to wind direction
10:  If (X<=>F) (P89)
 1: 6        X Loc [ Wind_dir  ]
 2: 4        <
 3: 0        F
 4: 30       Then Do

11:  Z=X+F (P34)
 1: 6        X Loc [ Wind_dir  ]
 2: 360      F
 3: 6        Z Loc [ Wind_dir  ]

12:  End (P95)

;-----------------------------------------
; G L O B A L  R A D I A T I O N
; Measure Li200s Pyranometer
13:  Volt (SE) (P1)
 1: 1        Reps
 2: 33       25 mV 50 Hz Rejection Range
 3: 10       SE Channel
 4: 7        Loc [ Li200S    ]
 5: 116.55   Mult
 6: 0        Offset

;-----------------------------------------
; P R E C I P I T A T I O N
; Measure tipping bucket rain gauge
14:  Pulse (P3)
 1: 1        Reps
 2: 2        Pulse Channel 2
 3: 2        Switch Closure, All Counts
 4: 8        Loc [ Precip    ]
 5: .16      Mult
 6: 0        Offset

;-----------------------------------------
; I N T E R N A L  T E M P E R A T U R E
15:  Internal Temperature (P17)
 1: 9        Loc [ T_int     ]
```

```
;-----------------------------------------
; B A R O M E T R I C  P R E S S U R E
16:  If time is (P92)
 1: 59       Minutes (Seconds --) into a
 2: 60       Interval (same units as above)
 3: 48       Set Port 8 High

17:  If time is (P92)
 1: 0        Minutes (Seconds --) into a
 2: 60       Interval (same units as above)
 3: 30       Then Do


     18:  Volt (SE) (P1)
      1: 1        Reps
      2: 15       2500 mV Fast Range
      3: 11       SE Channel
      4: 11       Loc [ P_mb      ]
      5: 0.2      Mult
      6: 600      Offset

     19:  Do (P86)
      1: 58       Set Port 8 Low

20:  End (P95)

;-----------------------------------------
; H O U R L Y  O U T P U T
21:  If time is (P92)
 1: 0        Minutes (Seconds --) into a
 2: 60       Interval (same units as above)
 3: 10       Set Output Flag High (Flag 0)

22:  Set Active Storage Area (P80)
 1: 1        Final Storage Area 1
 2: 101      Array ID

23:  Real Time (P77)
 1: 1220     Year,Day,Hour/Minute (midnight = 2400)

24:  Resolution (P78)
 1: 1        High Resolution

; Store average unvent and vent T and Rh
25:  Average (P71)
 1: 4        Reps
 2: 1        Loc [ T_1       ]

26:  Resolution (P78)
 1: 1        High Resolution

; Store wind speed, dir and std dev
27:  Wind Vector (P69)
 1: 1        Reps
 2: 1        Samples per Sub-Interval
 3: 0        S, theta(1), sigma(theta(1)) with polar sensor
 4: 5        Wind Speed/East Loc [ Wind_spd  ]
 5: 6        Wind Direction/North Loc [ Wind_dir  ]

28:  Resolution (P78)
 1: 1        High Resolution

; Store average global rad
29:  Average (P71)
 1: 1        Reps
 2: 7        Loc [ Li200S    ]

; Store hourly precipitation
30:  Totalize (P72)
 1: 1        Reps
 2: 8        Loc [ Precip    ]
```

```
; no data
31:  Average (P71)
 1: 2        Reps
 2: 12       Loc [ _________ ]

32:  Resolution (P78)
 1: 1        High Resolution

; Store maximum wind speed during last hour
33:  Maximum (P73)
 1: 1        Reps
 2: 10       Value with Hr-Min
 3: 5        Loc [ Wind_spd  ]

34:  Resolution (P78)
 1: 1        High Resolution

; Store transient unvent and vent T and Rh
35:  Sample (P70)
 1: 4        Reps
 2: 1        Loc [ T_1       ]

36:  Resolution (P78)
 1: 1        High Resolution

; Store max vent T
37:  Maximum (P73)
 1: 1        Reps
 2: 10       Value with Hr-Min
 3: 3        Loc [ T_vent    ]

38:  Resolution (P78)
 1: 1        High Resolution

; Store min vent T
39:  Minimum (P74)
 1: 1        Reps
 2: 10       Value with Hr-Min
 3: 3        Loc [ T_vent    ]

40:  Resolution (P78)
 1: 1        High Resolution

41:  Sample (P70)
 1: 1        Reps
 2: 11       Loc [ P_mb      ]

;-----------------------------------------
; D A I L Y  O U T P U T
42:  If time is (P92)
 1: 0        Minutes (Seconds --) into a
 2: 1440     Interval (same units as above)
 3: 10       Set Output Flag High (Flag 0)

43:  Set Active Storage Area (P80)
 1: 1        Final Storage Area 1
 2: 124      Array ID

44:  Real Time (P77)
 1: 1220     Year,Day,Hour/Minute (midnight = 2400)

45:  Resolution (P78)
 1: 1        High Resolution

; Store daily average unvent and vent T & Rh
46:  Average (P71)
 1: 4        Reps
 2: 1        Loc [ T_1       ]

47:  Resolution (P78)
 1: 1        High Resolution
```

```
; Store daily max unvent T
48:  Maximum (P73)
 1: 1        Reps
 2: 10       Value with Hr-Min
 3: 2        Loc [ T_2       ]

49:  Resolution (P78)
 1: 1        High Resolution

; Store daily min unvent T
50:  Minimum (P74)
 1: 1        Reps
 2: 10       Value with Hr-Min
 3: 2        Loc [ T_2       ]

51:  Resolution (P78)
 1: 1        High Resolution

; Store daily  max wind speed
52:  Maximum (P73)
 1: 1        Reps
 2: 10       Value with Hr-Min
 3: 5        Loc [ Wind_spd  ]

53:  Resolution (P78)
 1: 1        High Resolution

; Store average wind vector
54:  Wind Vector (P69)
 1: 1        Reps
 2: 1        Samples per Sub-Interval
 3: 1        S, theta(1) with polar sensor
 4: 5        Wind Speed/East Loc [ Wind_spd  ]
 5: 6        Wind Direction/North Loc [ Wind_dir  ]

55:  Resolution (P78)
 1: 1        High Resolution

; Store daily avg global radioation
56:  Average (P71)
 1: 1        Reps
 2: 7        Loc [ Li200S    ]

; Store daily precipitation
57:  Totalize (P72)
 1: 1        Reps
 2: 8        Loc [ Precip    ]

; Store sample of battery voltage
58:  Sample (P70)
 1: 1        Reps
 2: 10       Loc [ Battery   ]

; no data
59:  Average (P71)
 1: 2        Reps
 2: 12       Loc [ _________ ]

60:  Resolution (P78)
 1: 1        High Resolution

61:  Average (P71)
 1: 1        Reps
 2: 11       Loc [ P_mb      ]

;-----------------------------------------
; S Y N O P T I C  O U T P U T
; transient T data is stored every 3 hrs
;  according to synoptic standards.
62:  If time is (P92)
 1: 60       Minutes (Seconds --) into a
```

```
 2: 180      Interval (same units as above)
 3: 10       Set Output Flag High (Flag 0)

63:  Set Active Storage Area (P80)
 1: 1        Final Storage Area 1
 2: 103      Array ID

64:  Real Time (P77)
 1: 1220     Year,Day,Hour/Minute (midnight = 2400)

65:  Resolution (P78)
 1: 1        High Resolution

66:  Sample (P70)
 1: 1        Reps
 2: 2        Loc [ T_2       ]

*Table 2 Program
  01: 0.0000    Execution Interval (seconds)

*Table 3 Subroutines

End Program

1      [ T_1       ]  RW--   3       1      Start ------ ---
2      [ T_2       ]  RW--   6       1      ----- ------ End
3      [ T_vent    ]  RW--   5       1      ----- ------ ---
4      [ rH_vent   ]  RW--   3       1      ----- ------ ---
5      [ Wind_spd  ]  RW--   4       1      ----- ------ ---
6      [ Wind_dir  ]  RW--   4       2      ----- ------ ---
7      [ Li200S    ]  RW--   2       1      ----- ------ ---
8      [ Precip    ]  RW--   2       1      ----- ------ ---
9      [ T_int     ]  -W--   0       1      ----- ------ ---
10     [ Battery   ]  RW--   2       1      ----- ------ ---
11     [ P_mb      ]  RW--   2       1      ----- ------ ---
12     [ _________ ]  R---   2       0      ----- ------ ---
13     [ _________ ]  R---   2       0      ----- ------ ---
14     [ _________ ]  ----   0       0      ----- ------ ---
15     [ _________ ]  ----   0       0      ----- ------ ---
16     [ _________ ]  ----   0       0      ----- ------ ---
17     [ _________ ]  ----   0       0      ----- ------ ---
18     [ _________ ]  ----   0       0      ----- ------ ---
19     [ _________ ]  ----   0       0      ----- ------ ---
20     [ _________ ]  ----   0       0      ----- ------ ---
21     [ _________ ]  ----   0       0      ----- ------ ---
22     [ R_R0_T_1  ]  RW--   1       1      ----- ------ ---
23     [ R_R0_T_2  ]  RW--   1       1      ----- ------ ---
24     [ _________ ]  ----   0       0      ----- ------ ---
25     [ _________ ]  ----   0       0      ----- ------ ---
26     [ _________ ]  ----   0       0      ----- ------ ---
27     [ _________ ]  ----   0       0      ----- ------ ---
28     [ _________ ]  ----   0       0      ----- ------ ---
```

B Appendix: The Tarfala Research Station meteorological logger station sample data

The following is one full day of data from the Tarfala Research Station meteorological station. please note that hourly data starts with the number 101 and runs across two lines, synoptic data starts with 103 and is a short line of data and that the daily ouput occurs on a line starting with 124 (last line shown)

```
101,2008,105,100,-10.359,-9.982,-10.815,90.249,.97295,293.69,.03388,-.67007,0,0,0,3.3026,0,-10.613,-10.123,-10.928,
    90.14,-10.472,0,-11.013,55,875.95
103,2008,105,100,-10.123
101,2008,105,200,-10.717,-10.307,-11.154,89.928,1.131,296.41,.02958,-.68844,0,0,0,2.1266,128,-10.812,-10.404,-11.284,
    89.751,-10.878,104,-11.368,143,875.55
101,2008,105,300,-11.348,-10.962,-11.792,89.167,1.4753,302.15,.02821,-.70795,0,0,0,2.1756,256,-11.866,-11.542,-12.315,
    88.601,-11.267,200,-12.467,258,874.49
101,2008,105,400,-12.227,-11.857,-12.727,88.261,.93936,304.14,.02447,-.09822,0,0,0,1.9796,301,-12.575,-12.126,-12.97,
    87.821,-12.349,300,-13.038,347,873.83
103,2008,105,400,-12.126
101,2008,105,500,-12.454,-12.069,-12.952,88.037,1.3642,305.13,.02101,6.3198,0,0,0,2.5872,414,-12.555,-12.189,-13.118,
    87.835,-12.832,429,-13.118,459,873.3
101,2008,105,600,-12.486,-12.011,-13.033,87.816,1.5579,307.29,.01268,31.676,0,0,0,2.5872,553,-12.553,-12.106,-13.088,
    87.687,-12.864,519,-13.19,555,873.03
101,2008,105,700,-12.514,-11.962,-13.137,87.588,1.5625,303.36,.02331,59.897,0,0,0,2.7342,642,-12.588,-11.852,-13.11,
    87.435,-13.005,620,-13.295,638,872.5
103,2008,105,700,-11.852
101,2008,105,800,-12.288,-11.494,-13.141,87.339,1.0735,291.68,.02772,113.33,0,0,0,1.862,720,-12.014,-10.925,-13.063,
    87.389,-12.995,757,-13.28,743,872.1
101,2008,105,900,-11.358,-9.3691,-13.017,87.122,.61544,287.22,.0217,127.66,0,0,0,1.9894,858,-11.995,-10.803,-12.907,
    87.431,-12.839,859,-13.218,815,871.66
101,2008,105,1000,-11.846,-10.499,-13.004,87.021,1.3302,297.5,.0315,147.83,0,0,0,1.9992,910,-11.443,-9.756,-12.828,
    87.346,-12.76,959,-13.196,921,871.66
103,2008,105,1000,-9.756
101,2008,105,1100,-10.75,-8.3065,-12.601,87.08,.80921,286.34,.03313,186.21,0,0,0,1.6268,1052,-11.029,-9.0097,-12.583,
    87.116,-12.372,1023,-12.862,1002,871.5
101,2008,105,1200,-10.503,-7.8621,-12.415,87.226,.9408,286.56,.02452,195.31,0,0,0,2.4696,1157,-11.274,-9.4843,-12.334,
    87.271,-12.094,1133,-12.651,1101,871.23
101,2008,105,1300,-11.081,-8.9965,-12.425,87.384,1.1008,288.87,.02942,165.41,0,0,0,2.0972,1202,-10.984,-8.6758,-12.4,
    87.597,-12.216,1200,-12.589,1219,871.09
103,2008,105,1300,-8.6758
101,2008,105,1400,-10.38,-7.7242,-12.258,87.713,.63011,283.25,.03095,132.21,0,0,0,1.8424,1400,-11.016,-9.5962,-12.348,
    87.905,-12.044,1341,-12.484,1319,870.97
101,2008,105,1500,-11.038,-9.6917,-12.305,87.865,1.2839,291.06,.02815,112.93,0,0,0,2.0776,1400,-11.324,-9.7183,-12.261,
    88.047,-12.162,1416,-12.483,1438,870.85
101,2008,105,1600,-10.925,-9.7405,-12.344,87.844,1.1741,303.33,.03415,101.84,0,0,0,2.8224,1541,-11.387,-10.835,-12.528,
    87.822,-12.158,1539,-12.597,1557,870.59
103,2008,105,1600,-10.835
101,2008,105,1700,-11.539,-11.033,-12.468,88.118,2.0349,301.7,.03154,67.106,0,0,0,3.773,1654,-11.495,-10.654,-11.995,
    88.562,-11.912,1656,-12.865,1614,870.6
101,2008,105,1800,-11.328,-10.831,-11.995,88.897,2.1794,297.77,.03354,29.822,0,0,0,5.4292,1757,-9.4247,-9.6979,-10.719,
    91.579,-10.719,1800,-12.462,1740,870.64
101,2008,105,1900,-9.1491,-8.863,-9.9799,91.047,1.9179,277.59,.04224,12.584,0,0,0,6.027,1825,-9.0446,-8.3877,-9.4793,
    90.328,-8.5665,1835,-11.105,1811,871.18
103,2008,105,1900,-8.3877
101,2008,105,2000,-8.5986,-8.3367,-9.3684,91.441,2.3552,304.99,.03778,1.0391,0,0,0,6.2132,1936,-8.3607,-7.7241,-8.9671,
    91.864,-8.3079,1951,-10.24,1907,871.44
101,2008,105,2100,-8.1463,-7.8999,-8.8438,91.915,2.5721,289.5,.03889,-.61828,0,0,0,9.5648,2029,-8.1759,-7.8898,-8.885,
    92.06,-8.3751,2041,-9.2544,2006,871.97
101,2008,105,2200,-7.4936,-7.3195,-8.2989,91.944,2.579,270.09,.03997,-.66576,0,0,0,9.6432,2159,-6.1686,-6.4411,-7.2805,
    89.924,-7.1451,2158,-8.9527,2128,872.24
103,2008,105,2200,-6.4411
101,2008,105,2300,-5.65,-5.5125,-6.2838,76.769,6.1972,122.94,.03651,-.66679,0,0,0,16.66,2235,-5.402,-4.991,-5.9836,
    72.412,-5.7804,2255,-7.2298,2200,872.37
101,2008,105,2400,-5.6181,-5.2187,-6.1611,74.807,5.5665,124.19,.03096,-.63651,0,0,0,14.249,2317,-5.3036,-5.0363,-5.9341,
    72.663,-5.765,2353,-6.458,2325,873.16
124,2008,105,2400,-10.408,-9.4936,-11.355,87.608,-4.7622,2254,-12.23,501,16.66,2235,1.8068,295.22,61.939,0,13.902,0,0,872.38
```