Aurora

working to advance road weather information systems technology

Off-the-Shelf Component RWIS

http://aurora-program.org

Aurora Project 2003-02

Final Report
May 2008

Technical Report Documentation Page

1. Report No. Aurora Project 2003-02	**2. Government Accession No.**	**3. Recipient's Catalog No.**
4. Title and Subtitle Off-the-Shelf Component RWIS		**5. Report Date** May 2008 **6. Performing Organization Code**
7. Author(s) Claude Lapointe, Max Perchanok, Jack Stickel		**8. Performing Organization Report No.**
9. Performing Organization Name and Address Center for Transportation Research and Education Iowa State University 2711 South Loop Drive, Suite 4700 Ames, IA 50010-8664		**10. Work Unit No. (TRAIS)** **11. Contract or Grant No.**
12. Sponsoring Organization Name and Address Iowa Highway Research Board Iowa Department of Transportation 800 Lincoln Way Ames, IA 50010		**13. Type of Report and Period Covered** Final Report **14. Sponsoring Agency Code**

15. Supplementary Notes

Visit www.ctre.iastate.edu for color PDF files of this and other research reports.

16. Abstract

Road Weather Information Systems (RWIS) are sold by most vendors as an integrated package. Vendors integrate a suite of sensors, a data acquisition system, and a communication and display system that is provided in a turn-key format. This restricts the ability of the highway agency to select competing components that may have preferred qualities, or to update the system as new hardware or software components are developed.

The purpose of this project was to develop a programmable, off-the-shelf RWIS that allows choice of initial sensors, and the ability to change sensors or other components over time without completely replacing the system. The methodology for this project involved review and selection of sensors and data acquisition systems to meet the following requirements:

- Sensors should be easily interfaced with a data acquisition system and should provide good functional performance and complete documentation to allow interfacing with alternative Remote Processing Units (RPUs).
- RPUs or data acquisition systems should have open programming and provide inputs to the desired sensors types.

17. Key Words data acquisition system—datalogger —remote processing units—road weather station—sensors		**18. Distribution Statement** No restrictions.	
19. Security Classification (of this report) Unclassified.	**20. Security Classification (of this page)** Unclassified.	**21. No. of Pages** 79	**22. Price** NA

Form DOT F 1700.7 (8-72) **Reproduction of completed page authorized**

OFF-THE-SHELF COMPONENT RWIS

Final Report
May 2008

Project Champion
Claude Lapointe
Quebec Ministry of Transportation

Co-Project Champions
Max Perchanok
Aurora Program Chair
Ontario Ministry of Transportation

Jack Stickel
Transportation Planner
Alaska Department of Transportation

Authors
Claude Lapointe, Max Perchanok, Jack Stickel

Preparation of this report was financed in part
through funds provided by the Iowa Department of Transportation
through its research management agreement with the
Center for Transportation Research and Education,
Aurora Project 2003-02.

A report from
Aurora Program
Iowa State University
2711 South Loop Drive, Suite 4700
Ames, IA 50010-8664
Phone: 515-294-7684
Fax: 515-294-0467
www.aurora-program.org

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

1. INTRODUCTION

Road Weather Information Systems (RWIS) are sold by most vendors as an integrated package. Vendors integrate a suite of sensors, a data acquisition system, and a communication and display system that is provided in a turn-key format. This restricts the ability of the highway agency to select competing components that may have preferred qualities, or to update the system as new hardware or software components are developed.

The purpose of this project was to develop a programmable, off-the-shelf RWIS that allows choice of initial sensors, and the ability to change sensors or other components over time without completely replacing the system.

2. METHODOLOGY

The methodology for this project involved review and selection of sensors and data acquisition systems to meet the following requirements:

- Sensors should be easily interfaced with a data acquisition system, and should provide good functional performance and complete documentation to allow interfacing with alternative Remote Processing Units (RPUs).
- RPUs or data acquisition systems should have open programming and provide inputs to the desired sensors types.

The first criteria to construct the system was to choose a programmable RPU. We chose Campbell Scientific CR5000 that has enough channels to connect many sensors at once and has a good real time operating system for programming (CRBasic). Also, the software for data acquisition Loggernet 1.2a is very powerfull and has a distributed architecture that includes a server for data acquisition, remote client for managment of the server over TCP/IP network, and remote server LoggerNet Socket data export to export data in real time to an external application to store data into a database.

We analyzed the specification of each sensor and selected one that was easy to interface to the RPU and offered good performance. For example, we selected a Lufft road sensor with RS-485 interface. Similarly, the Optical Scientific WIVIS 130 visibility and precipitation sensor has an RS232C interface and can be specified with either World Meterological Organization (WMO) precipitation code for Canada or National Weather Service (NWS) precipitation code for the U.S.

We did not compare all the sensors in the market for this proof of concept study. This is a possibility for a future project to analyse and rate all sensors with respect to interfacing capabilities with an open system RPU.

2.1 The Road Weather Station

The construction of the road weather station of Petite-Riviere-Saint-Francois took place in August 2003. It is located at kilometer 431.5 on road 138 west. It is equipped with sensors making it possible to obtain the following information:

- Speed and direction of the winds
- Temperature, relative humidity, and dew point
- Atmospheric pressure
- Temperature of the road surface with depth of -5, -10, -20, -30, -40, -60, -90, -150, -200, -300 cm
- Salt rate of the roadway
- Surface quality of the roadway
- Radiative value of the roadway

- Type of precipitation and visibility

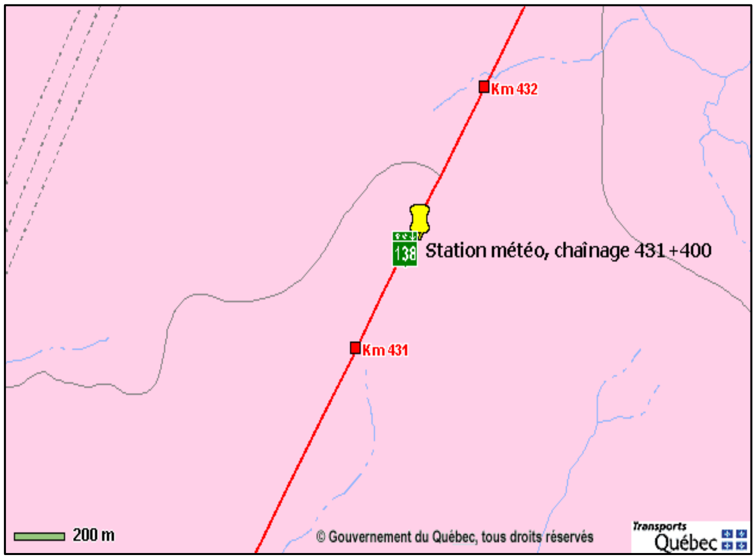


Figure 1. Location of the road weather station

Specifications for the road weather station include the following:

- Region : Charlevoix
- Municipality : Petite-Riviere-Saint-François
- Section : 000138-07-180-00C
- Road: 138
- G.P.S coordinates: N47°, 17’, 19.8’’; W070°, 39’, 20.7’’; Altitude: 743 M
- Magnetic variation: 18.4° West
- Station modem number: (418) 435-0199

2.2 Remote Access

In order to collect the data of the road weather station, we installed two telephone lines. The first telephone line is connected to the system of data acquisition of the road weather station. The second telephone line is located in a territorial direction or a center of service closer to the road weather station. A modem is connected on the second telephone line. In order to collect the remote data by the network of the ministry, an interface of network link interface (NL100) is connected with the network of the ministry. Thus, we can telephone the road weather station every five minutes to collect the stored data of the system of acquisition of the data (Datalloger CR5000).

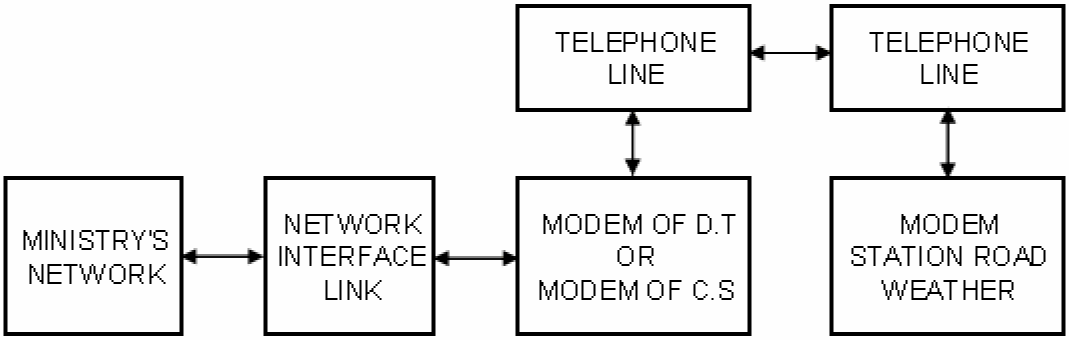


Figure 2. Collecting the information

2.3. Equipment and Specifications

2.3.1 Tower and Accessories

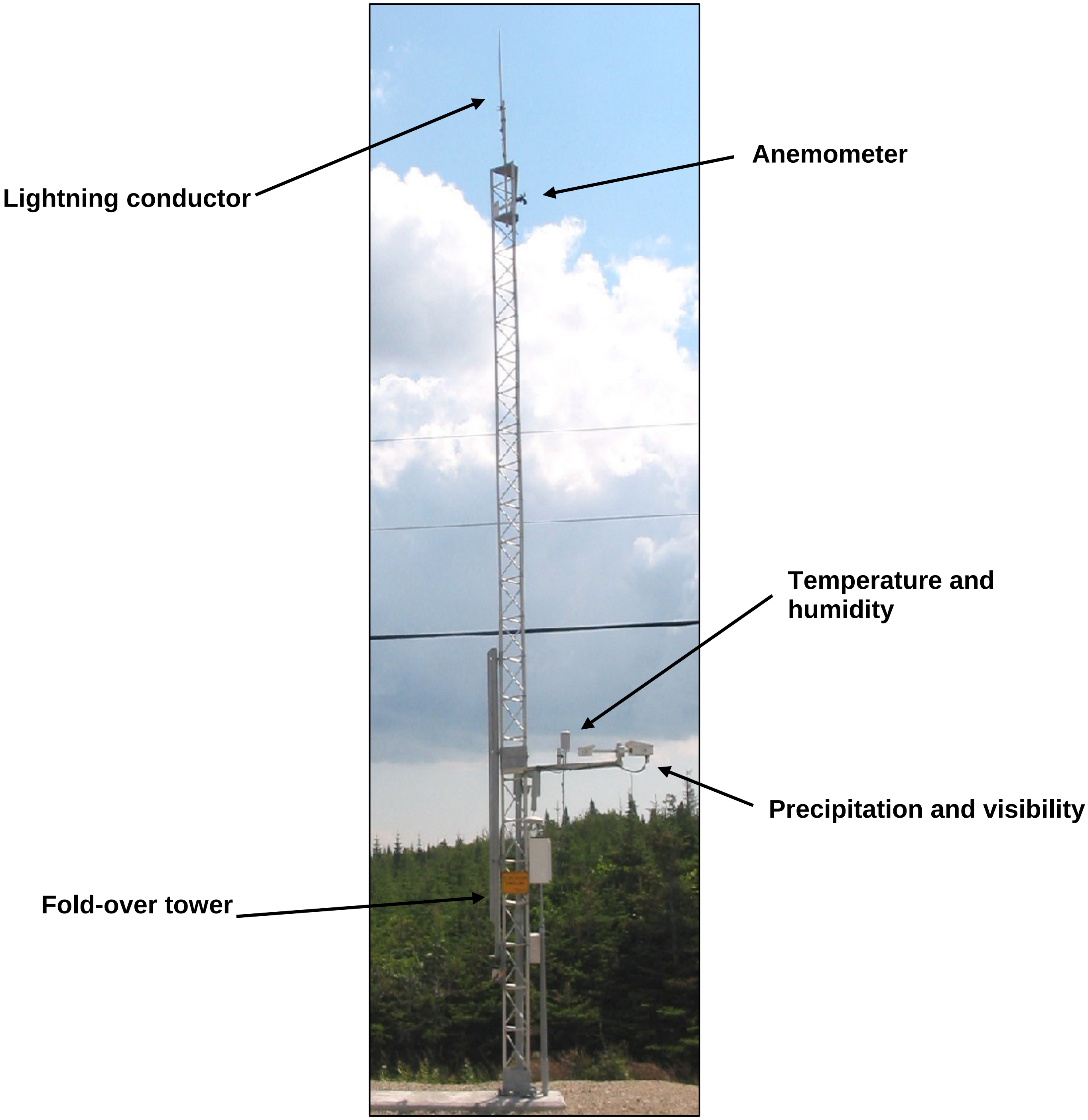


Figure 3. Tower

Figure 4. Temperature and humidity sensor

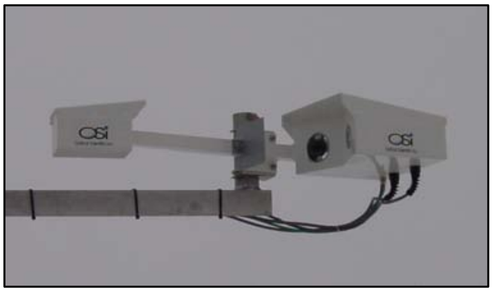


Figure 5. Precipitation and visibility sensor

Figure 6. Anemometer

2.3.2 Principal Cabinet

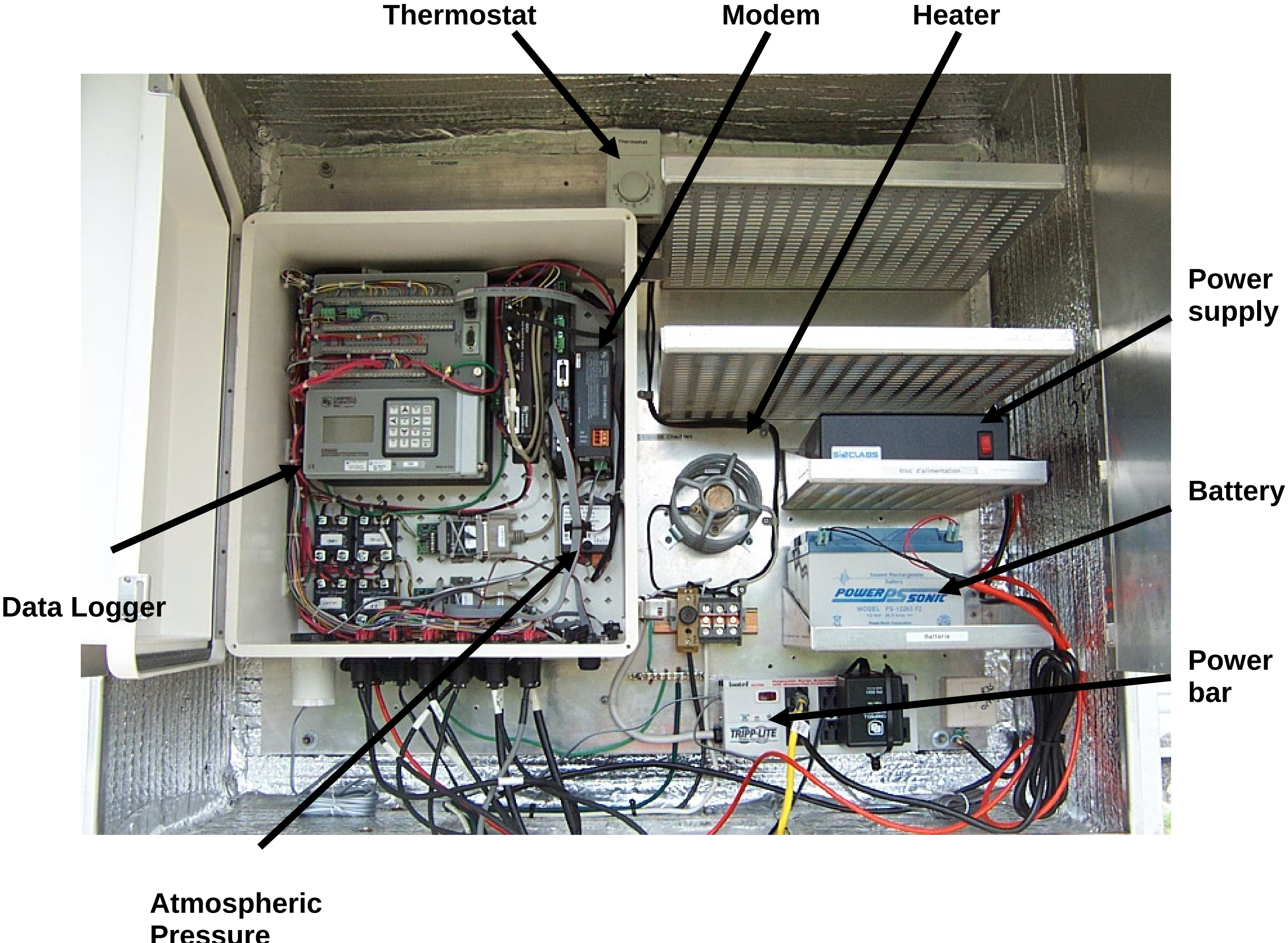


Figure 7. Principal cabinet

2.3.3 In-road Sensors

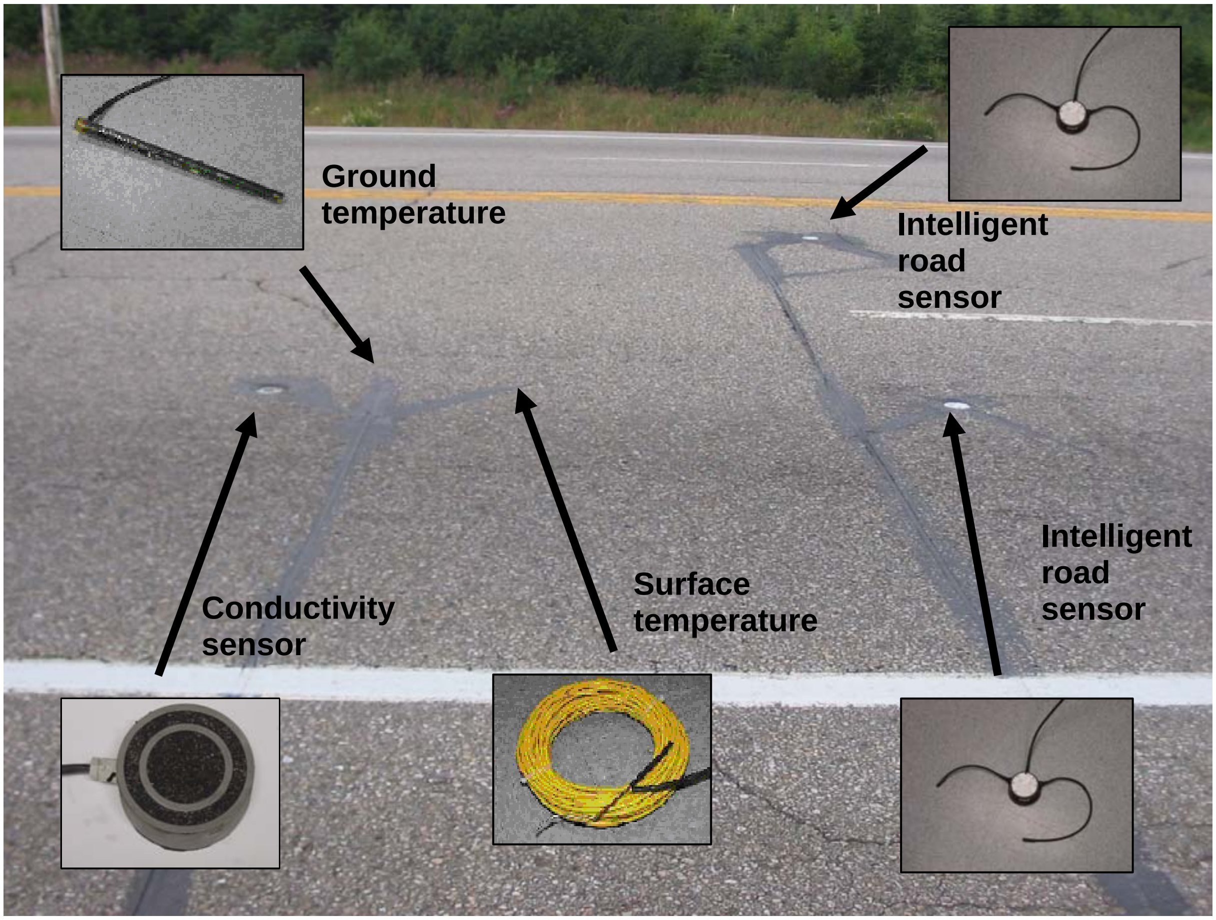


Figure 8. In-road sensors

2.3.4 Radiometer

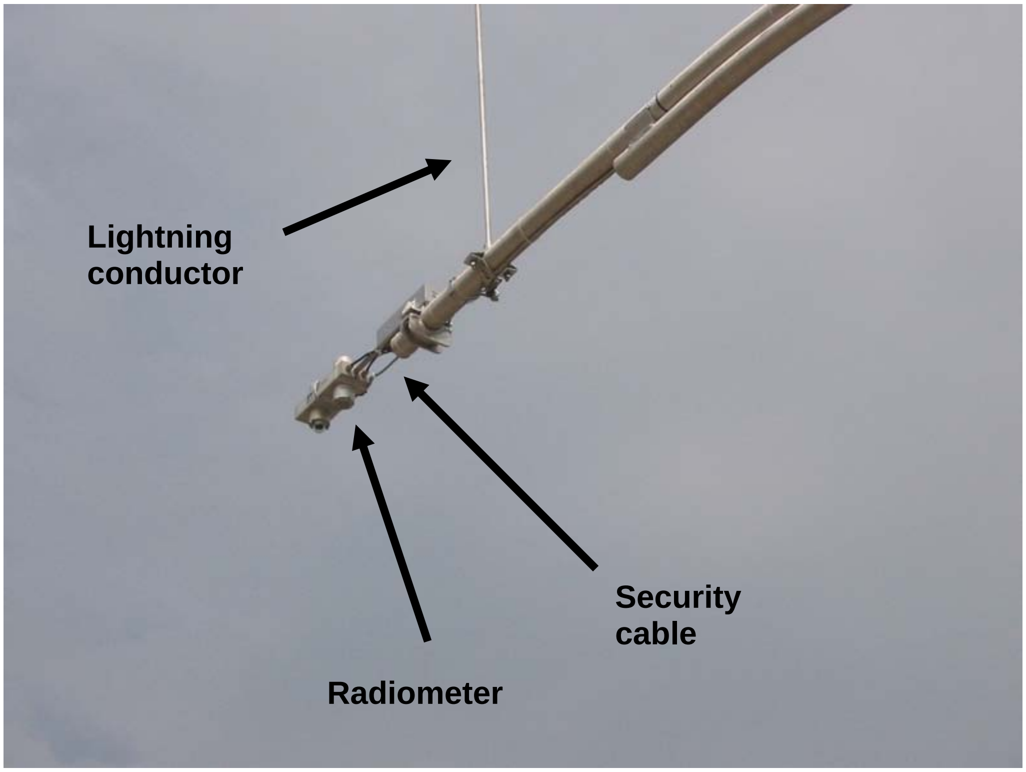


Figure 9. Radiometer

2.3.5 Datalogger CR5000

Model CR5000 (c) 2001 Campbell Scientific (Canada) Corp.

Figure 10. Datalogger CR5000

The CR5000 is a rugged, high performance, integrated data acquisition system with a built-in keyboard, graphics display, and PCMCIA card slot. It combines 16-bit resolution with a maximum throughput of 5000 measurements per second. The CR5000 is available with a built-in sealed rechargeable battery base or in a low profile version without batteries.

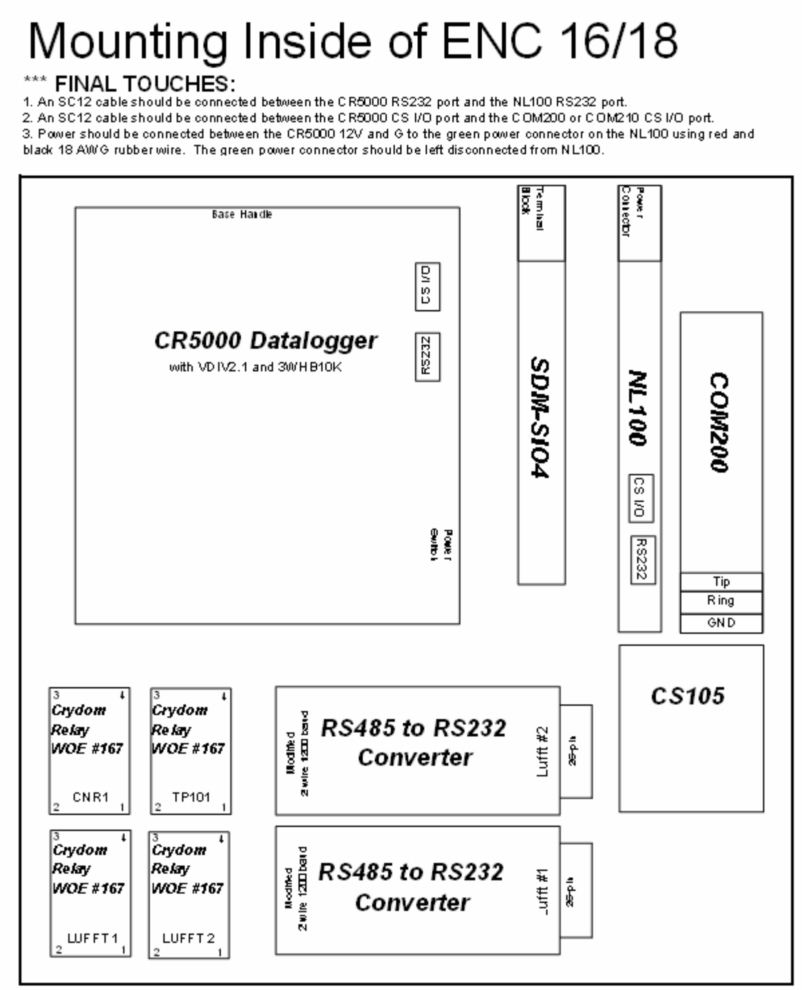


Figure 11. Inside of ENC 16/18

CR5000 Specifications

Electrical specifications are valid over a -25° to +50°C range unless otherwise specified; testing over -40° to +85°C is available as an option excluding batteries. Non-condensing environment is required. To maintain electrical specifications, Campbell Scientific recommends recalibrating dataloggers every two years.

PROGRAM EXECUTION RATE

The CR5000 can measure one channel and store the result in 500 μs; all 40 SE* channels can be measured in 8 ms (5 kHz aggregate rate).

ANALOG INPUTS

DESCRIPTION: 20 DF* or 40 SE, individually configured. Channel expansion provided through AM16/32, AM416, and AM25T Multiplexers.

RANGES, RESOLUTION, AND TYPICAL INPUT NOISE: Basic Resolution (Basic Res) is the A/D resolution of a single conversion. Resolution of DFM* with input reversal is half the Basic Res. Noise values are for DFM with input reversal; noise is greater with SEM.*

Input Rng (mV)	Basic Res (μV)	0 Int. (μV RMS)	250 μs Int. (μV RMS)	20/16.7 ms Int. (μV RMS)
±5000	167	70	60	30
±1000	33.3	30	12	6
±200	6.67	8	2.4	1.2
±50	1.67	3.0	0.8	0.3
±20	0.67	1.8	0.5	0.2

ACCURACY†:

±(0.05% of Reading + Offset) 0° to 40°C
±(0.075% of Reading + Offset) -25° to 50°C
±(0.10% of Reading + Offset) -40° to 85°C

Offset for DFM w/input reversal = Basic Res + 1 μV
Offset for DFM w/o input reversal = 2Basic Res + 2 μV
Offset for SEM = 2Basic Res + 10 μV

MINIMUM TIME BETWEEN MEASUREMENTS:

Zero Integration:	125 μs
250 μs Integration:	475 μs
16.7 ms Integration:	19.9 ms
20 ms Integration:	23.2 ms

COMMON MODE RANGE: ±5 V

DC COMMON MODE REJECTION: >100 dB with input reversal (>80 dB without input reversal)

NORMAL MODE REJECTION: 70 dB @ 60 Hz when using 60 Hz rejection

SUSTAINED INPUT VOLTAGE WITHOUT DAMAGE: ±16 Vdc

INPUT CURRENT: ±2 nA typ., ±10 nA max. @ 50°C

INPUT RESISTANCE: 20 G typical

ACCURACY OF INTERNAL THERMOCOUPLE REFERENCE JUNCTION:

±0.25°C, 0° to 40°C
±0.5°C, -25° to 50°C
±0.7°C, -40° to 85°C

ANALOG OUTPUTS

DESCRIPTION: 4 switched voltage; 4 switched current; 2 continuous voltage; switched outputs active only during measurements, one at a time.

RANGE: Voltage (current) outputs programmable between ±5 V (±2.5 mA)

RESOLUTION: 1.2 mV (0.6 μA) for voltage (current) outputs

ACCURACY: ±10 mV (±10 μA) for voltage (current) outputs

CURRENT SOURCING: 50 mA for switched voltage; 15 mA for continuous

CURRENT SINKING: 50 mA for switched voltage; 5 mA for continuous (15 mA w/selectable option)

COMPLIANCE VOLTAGE: ±5 V for switched current excitation

RESISTANCE MEASUREMENTS

Provides voltage ratio measurements of 4- and 6-wire full bridges, and 2-, 3-, 4-wire half bridges. Direct resistance measurements available with current excitation. Dual-polarity excitation is recommended.

VOLTAGE RATIO ACCURACY†: Assumes input and excitation reversal and an excitation voltage of at least 2000 mV.

±(0.04% Reading + Basic Res/4) 0° to 40°C
±(0.05% Reading + Basic Res/4) -25° to 50°C
±(0.06% Reading + Basic Res/4) -40° to 85°C

ACCURACY† WITH CURRENT EXCITATION: Assumes input and excitation reversal, and an excitation current, I_x, of at least 1 mA.

±(0.075% Reading + Basic Res/$2I_x$) 0° to 40°C
±(0.10% Reading + Basic Res/$2I_x$) -25° to 50°C
±(0.12% Reading + Basic Res/$2I_x$) -40° to 85°C

PERIOD AVERAGING MEASUREMENTS

DESCRIPTION: The average period for a single cycle is determined by measuring the duration of a specified number of cycles. Any of the 40 SE analog inputs can be used; signal attenuation and ac coupling may be required.

INPUT FREQUENCY RANGE:

Input Rng (mV)	Signal (peak to peak) Min.	Max.[1]	Min. Pulse W.	Max. Freq
±5000	600 mV	10 V	2.5 μs	200 kHz
±1000	100 mV	2.0 V	5.0 μs	100 kHz
±200	4 mV	2.0 V	25 μs	20 kHz

[1]Maximum signals must be centered around datalogger ground.

RESOLUTION: 70 ns/number of cycles measured

ACCURACY: ±(0.03% of Reading + Resolution)

PULSE COUNTERS

DESCRIPTION: Two 16-bit inputs selectable for switch closure, high frequency pulse, or low-level ac.

MAXIMUM COUNT: 4 x 10^9 counts per scan

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

HIGH FREQUENCY PULSE MODE:
Maximum Input Frequency: 400 kHz
Maximum Input Voltage: ±20 V
Voltage Thresholds: Count upon transition from below 1.5 V to above 3.5 V at low frequencies. Larger input transitions are required at high frequencies because of 1.2 μs time constant filter.

LOW LEVEL AC MODE:
Internal ac coupling removes dc offsets up to ±0.5 V.

Input Hysteresis: 15 mV
Maximum ac Input Voltage: ±20 V
Minimum ac Input Voltage (sine wave):

(mV RMS)	Range (Hz)
20	1.0 to 1000
200	0.5 to 10,000
1000	0.3 to 16,000

DIGITAL I/O PORTS

DESCRIPTION: 8 ports selectable as binary inputs or control outputs.

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

OUTPUT RESISTANCE: 330

INPUT STATE: high 3.0 to 5.3 V; low -0.3 to 0.8 V

INPUT RESISTANCE: 100 k

EMI and ESD PROTECTION

The CR5000 is encased in metal and incorporates EMI filtering on all inputs and outputs. Gas discharge tubes provide robust ESD protection on all terminal block inputs and outputs. The following European CE standards apply.

EMC tested and conforms to BS EN61326:1998.

Details of performance criteria applied are available upon request.

Warning: This is a Class A product. In a domestic environment this product may cause radio interference in which case the user may be required to correct the interference at the user's own expense.

CPU AND INTERFACE

PROCESSOR: Hitachi SH7034

MEMORY: Battery-backed SRAM provides 2 Mbytes for data and operating system use with 128 kbytes reserved for program storage. Expanded data storage with PCMCIA type I, type II, or type III card.

DISPLAY: 8-line-by-21 character alphanumeric or 128 x 64 pixel graphic LCD display w/backlight.

SERIAL INTERFACES: Optically isolated RS-232 9-pin interface for computer or modem. CS I/O 9-pin interface for peripherals such as CSI modems.

BAUD RATES: Selectable from 1,200 to 115,200 bps. ASCII protocol is eight data bits, one start bit, one stop bit, no parity.

CLOCK ACCURACY: ±1 minute per month, -25° to +50°C; ±2 minute per month, -40° to +85°C

SYSTEM POWER REQUIREMENTS

VOLTAGE: 11 to 16 Vdc

TYPICAL CURRENT DRAIN: 400 μA software power off; 1.5 mA sleep mode; 4.5 mA at 1 Hz (200 mA at 5 kHz) sample rate.

INTERNAL BATTERIES: 7 Ahr rechargeable base (optional); 1650 mAhr lithium battery for clock and SRAM backup, 10 years of service typical, less at high temperatures.

EXTERNAL BATTERIES: 11 to 16 Vdc; reverse polarity protected.

PHYSICAL SPECIFICATIONS

SIZE: 9.8" x 8.3" x 4.5" (24.7 cm x 21.0 cm x 11.4 cm) Terminal strips extend 0.4" (1.0 cm).

WEIGHT: 4.5 lbs (2.0 kg) with low-profile base; 12.2 lbs (5.5 kg) with rechargeable base

WARRANTY

Three years against defects in materials and workmanship.

*SE(M): Single-Ended (Measurement)
*DF(M): Differential (Measurement)
† *Sensor and measurement noise not included.*

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

CAMPBELL SCIENTIFIC
CANADA CORP.
11564 - 149 street - edmonton - alberta - T5M 1W7
tel 780.454.2505 fax 780.454.2655
www.campbellsci.ca

2.3.6 Serial Port Interface (SDM-SIO4)

Model SDM-SIO4 (c) 2001 Campbell Scientific (Canada) Corp.

Figure 12. Serial port interface

The SDM-SIO4 has four configurable serial RS-232 ports that communicate with intelligent serial sensors and other serial devices. Once programmed, the SDM-SIO4 communicates with devices connected in parallel with the datalogger's own program sequence, thus making the complete datalogging system faster and more efficient. A multi-tasking operating system allows concurrent transmission and receipt of data on all ports.

The SDM-SIO4’s program is as follows:

```
'_____________________SDM-SIO4 Serial Data Interface
Declarations_____________________________
'Global Variables used by all sensors measured through the SDM-SIO4
Const DelayProcessing = 1                'Delay type 1 delays processing
Const OneRep = 1                         'One repetition
Const NoValues = 0                 '0 values
Const OneValue = 1                 '1 value
Const UnityMultiplier = 1.0              'Multiplier is 1.0
Const NoOffset = 0.0               'Offset is 0.0
'SDM-SIO4 Configurations
Public ScanCounter                 'Location used for troubleshooting
Public CheckSigs                         'Location to control Sub VerSig
Public Sig(2)                      'Declare location for the signatures of the SDM-
SIO4
Public COMSetUp                          'Location to control Sub ComSU
Public ProgMing                          'Location to control Sub ProgSIO4
Const Sio4Address00 = 0                  'SDM-SIO4 address is 0
Const Port1 = 1                          'Port 1
Const Port2 = 2                          'Port 2
Const Port3 = 3                          'Port 3
Const Port4 = 4                          'Port 4
```

```
Const AllPorts = 5                              'All ports
Const PollForData = 0001                        'SDM-SIO4 command - Poll of available
data
Const GetSigs = 0002                    'SDM-SIO4 command - Signatures
Const FlushRxBuffers = 0003                     'SDM-SIO4 command - Flush of all receive
buffers
Const Status = 0005                     'SDM-SIO4 command - Status
Const ProgMingMode = 0321                       'SDM-SIO4 command - Execute command
line command
Const SendString = 1024                         'SDM-SIO4 command - Send String to
Device
Const ComSetupCmd = 2049                'SDM-SIO4 command - Set communication
parameters
Const StartRxFilter = 2054                      'SDM-SIO4 command - Setup receive filter
Const SendDataToLgr = 0004                      'SDM-SIO4 command - Send data to
datalogger
Const  SendByteToLgr   = 66             'SDM-SIO4 command - Send sigle-byte data to
datalogger
Const FlushCDBuffer = 0009              'SDM-SIO4 command - Flush converted data buffer
Const UnusedParameter = 0000                    'Unused constant
Dim NotUsed                             'Unused variable
Const ComCode = 9143                            'Communication code for Lufft sensors and
WIVIS
                                                '9=NoHandshaking; 1=1StopBitNoParity;
                                                '4=8DataBits; 3=1200Baud


'______________________General SDM-SIO4 Sub
Procedures___________________________________

  'SUBPROCEDURE VerSig: Check SDM-SIO4 to verify that it is functional.
  'SDM-Sio4 Figurations:
  'Check EPROM and String Area Signatures
   CheckSigs = False
   Scan(1,Sec,0,0)

SIO4(Sig(),OneRep,Sio4Address00,AllPorts,GetSigs,UnusedParameter,UnusedParameter,GetSi
gs,UnityMultiplier,NoOffSet)
   Delay(DelayProcessing,100,mSec)
   If Sig(1)>0 And Sig(2)>0 Then CheckSigs = True:ExitScan
   NextScan

  'Setup Sio4 Ports for 1200 baud communications. (Same for Lufft and WIVIS)
  'Program the Port communication Settings into the SDM-SIO4.
               COMSetUp = False
   ProgMing = False
   Scan(12,Sec,0,1) 'Setup Sio4 Ports for Lufft IRS-20 and WIVIS communications
```

```
SIO4(NotUsed,OneRep,Sio4Address00,AllPorts,ComSetupCmd,ComCode,UnusedParameter,N
oValues,UnityMultiplier,NoOffset)
        Delay(DelayProcessing,2,Sec)
        COMSetUp=True

  'Program Lufft and WIVIS Command Strings into the Sio4.
  'Program Command strings and Filter strings into the SDM-SIO4.
  'Lufft Strings
  'Program the request data command string: strst 1 "&&A^M^J"
   Data 115,116,114,115,116,32,49,32,34,38,38,65,94,77,94,74,34
   For j= 1 To strst1Bytes : Read strst1Byte(j) : Next j

SIO4(strst1Byte(),OneRep,Sio4Address00,AllPorts,ProgMingMode,UnusedParameter,UnusedPa
rameter,strst1Bytes,UnityMultiplier,NoOffset)
   Delay(DelayProcessing,2,Sec)
  'Program the defined receive filter string: fltst 2 "xfX"
   fltst2Byte(1)=102 : fltst2Byte(2)=108 : fltst2Byte(3)=116 : fltst2Byte(4)=115 :
fltst2Byte(5)=116
   fltst2Byte(6)=32 : fltst2Byte(7)=50 : fltst2Byte(8)=32
   fltst2Byte(9)=34 '"
   fltst2Byte(10)=120 'x
   fltst2Byte(11)=102 'f
   fltst2Byte(12)=88 'X
   fltst2Byte(13)=34 '"

SIO4(fltst2Byte(),OneRep,Sio4Address00,AllPorts,ProgMingMode,UnusedParameter,UnusedPa
rameter,fltst2Bytes,UnityMultiplier,NoOffset)
   Delay(DelayProcessing,2,Sec)
  'WIVIS Strings
  'Program the request data command string: strst 3 "D"
   Data 115,116,114,115,116,32,51,32,34,68,34
   For n= 1 To strst3Bytes : Read strst3Byte(n) : Next n

SIO4(strst3Byte(),OneRep,Sio4Address00,Port4,ProgMingMode,UnusedParameter,UnusedPara
meter,strst3Bytes,UnityMultiplier,NoOffset)
   Delay(DelayProcessing,2,Sec)
   'Program the defined receive filter string: fltst 4 "T[^BW]xN39zX"
   fltst4Byte(1)=102 : fltst4Byte(2)=108 : fltst4Byte(3)=116 : fltst4Byte(4)=115 :
fltst4Byte(5)=116
   fltst4Byte(6)=32 : fltst4Byte(7)=52 : fltst4Byte(8)=32
   fltst4Byte(9)=34 '"
   fltst4Byte(10)=84 'T
   fltst4Byte(11)=91 '[
   fltst4Byte(12)=2 '^B
   fltst4Byte(13)=87 'W
   fltst4Byte(14)=93 ']
   fltst4Byte(15)=120 'x
   fltst4Byte(16)=78 'N
```

```
   fltst4Byte(17)=51 '3
   fltst4Byte(18)=57 '9
   fltst4Byte(19)=122 'z
   fltst4Byte(20)=88 'X
   fltst4Byte(21)=34 '"

SIO4(fltst4Byte(),OneRep,Sio4Address00,Port4,ProgMingMode,UnusedParameter,UnusedPara
meter,fltst4Bytes,UnityMultiplier,NoOffset)
   Delay(DelayProcessing,2,Sec)
   CallTable(Sio4Prog)
   ProgMing=True
   NextScan
```

Table 1. The SDM-SIO4's specifications

Specifications

Valid for a temperatue range of -25° to +50°C, unless otherwise specified.

Compatible dataloggers: CR10(X), CR23X, CR5000, CR7, and CR9000. CR7s require PROM version OS7-0.2 or higher and a special SDM terminal provided on CR7s shipped after April, 1997. Contact Campbell Scientific if using an older CR7.

Serial Ports

Number of Ports:	4 (independently configurable for different serial data formats)
Baud Rate:	25 to 115,200 bps
Port Output:	0 to 5 V logic; ±5 V for RS-232 (switchable)
Port Configuration:	9-pin 'D' connectors
Data Flow Control:	By datalogger or SDM-SIO4, if required, using hardware or software protocols
Buffers (each port):	Receive (Rx), 981 bytes + 16-byte hardware buffer; transmit (Tx), 981 bytes + 16-byte hardware buffer; processed data storage, 891 bytes. (Suitable for storing 224 4-byte Campbell Scientific floating point values). Buffers are 'fill-and-stop' type. Once filled, any additional data received will be lost. There is an additional buffer, which is used only when the datalogger outputs floating point data via the SDM-SIO4. This buffer is 241 bytes — sufficient for 60 floating-point values.
On-board Diagnostics:	A built-in system watchdog resets the processor in the event of a crash caused by transients and a built-in LED gives an indication of SDM-SIO4 status on power-up.
Input Voltage Limits:	±25 V

SDM Port

Datalogger Communication rate:	The speed at which data is transferred is controlled by the datalogger and can vary with the microprocessor activity as well as the length of the SDM cables.
Typical Transfer Rate:	One byte per millisecond

Power

Power Supply:	Unregulated 12 V supply, 9 to 18 Vdc
Current Drain:	29 mA with all ports active; 0.7 mA quiescent (quiescent state entered if there is no SDM or port activity for approximately 30 ms)
Internal Battery:	Retains configuration information only (lithium battery with an estimated life of 10 years)

Environmental

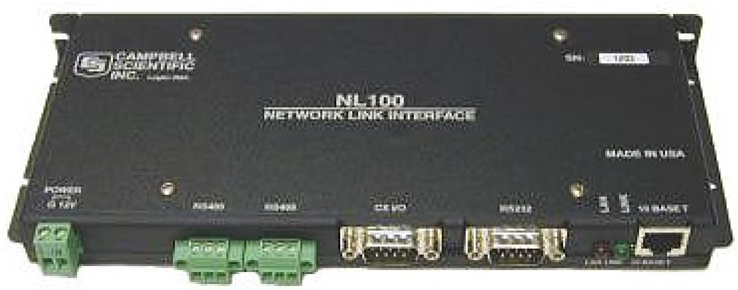
Temperature:	-25° to +50°C (contact Campbell Scientific for extended temperature operations)
Humidity:	0 to 95% RH (non-condensing)

Physical

Case:	Anodized aluminum
Dimensions:	7.3" x 3.5" x 1.4" (18.4 x 8.8 x 3.4 cm)
Weight:	15 oz. (0.4 kg)
Max. Cable Length:	20 ft (6 m)

CAMPBELLSCIENTIFIC
CANADA CORP.
11564 - 149 street - edmonton - alberta - T5M 1W7
tel 780.454.2505 fax 780.454.2655
www.campbellsci.ca

2.3.7 Network Link Interface (NL100)



Model NL100 (c) 2001 Campbell Scientific (Canada) Corp.

Figure 13. Network link interface (NL100)

The NL100 Network Link Interfaces are 10baseT ethernet communication peripherals. The devices allow any Campbell datalogger with an RS-232 or CS I/O port to communicate with a computer using TCP/IP, making it possible to communicate over a local area network or a dedicated Internet connection. Communication rate is dependent on the datalogger; for example, the CR5000 can communicate at rates up to 115,200 bps. The NL100 mounts directly to the backplate of our environmental enclosures. The NL100 may be located several miles from the datalogger by using short-haul modems, RF modems, MD485 multidrop interfaces, etc. The NL100 is configured by the user with an IP address via a computer's RS-232 port and terminal emulator software. Once the interface has a working IP address, subsequent configurations and setting changes are possible via TCP/IP using Telnet.

The Network link interface's specifications are as follows:

Standards

Ethernet Standard IEEE 802.3 (CSMA/CD Access Method)
TCP/IP Protocol

Case Dimensions

9 ¼" x 4 ¼" x 1"

Environment

Temperature: -25 to +50 °C

Power Supply Requirements

The NL100/105 is powered via the green G 12V connector (11 to 16 VDC at 140 mA average current) on the front panel of the unit.

Cable Specifications

The Ethernet 10 Base-T cable should be a Category 5 twisted pair cable. The two active pairs in an Ethernet 10 Base-T network are pins 1 & 2 and pins 3 & 6. Use only dedicated wire pairs (such as blue/white & white/blue, orange/white & white/orange) for the active pairs.

RJ-45 Pin-Outs: Pin 1 = TD+, Pin 2 = TD-, Pin 3 = RD+, Pin 6 = RD-

The RS-232 should be a standard straight through cable. It is recommended that the cable be kept at lengths of ≤ 6 feet to maintain high data throughput rates.

The CS I/O 9 Pin cable is a straight through cable with all 9 pins connected. Campbell Scientific's SC12 cable is recommended.

The cable for the RS-485 connection is a 2 twisted pair, 22 awg cable. CSI recommends the 9720 cable (with a Santoprene jacket) for use with the MD485s.

2.3.8 Modem (COM 210)

Model COM210 (c) 2002 Campbell Scientific (Canada) Corp.

Figure 14. Modem (COM 210)

The COM210 modem enables communications between a computer and a Campbell Scientific datalogger over a public switched telephone network. A Hayes-compatible modem is required at the computer base station. The COM210 connects to the datalogger at the field site. The COM210's wide operating temperature range and low power requirements make it ideal for use at remote sites.

LoggerNet software supports error-checking and manual or automated dialing. Loggernet can run unattended allowing the use of lower off-hour telephone rates.

The modem's specifications are as follows:

- Bell 212A, CCITT V.22, and CCITT V.32 compatible
- Full duplex at 9600 (default setting) and 1200 baud to datalogger
- V.42 LAPM and MNP2-4 error correction
- Hayes AT command set
- On-board speaker
- RJ-11C telephone jack
- FCC and IC (formerly known as DOC) approval
- Pulse or tone dialing
- Current drain: 120 µA quiescent, 160 mA active
- Direct connection to and powered by CSI dataloggers
- Supply requirements: 12 VDC regulated power supply
- Internally switches 12 VDC external power minimizing current drain
- Logic levels: below 1.5 V inputs a low state and above 3.5 V inputs a high state. A low voltage level on the TX data input (pin 9) and RX data output (pin 4) represents a mark
- Operational temperature: -25°C to +50°C standard (Extended operational temperature range of -55°C to +85°C is available at time of purchase.)
- Size: 5.2" x 1.7" x 3.6" // 13.1 x 4.3 x 9.2 cm
- Weight: 0.75 lbs // 0.34 kg

2.3.9 Pressure Transmitter (CS105)

Model CS105 (c) 2001 Campbell Scientific (Canada) Corp.

Figure 15. Pressure transmitter (CS105)

The CS105 Barometer uses Vaisala's silicon capacitive sensor to measure barometric pressure over a 600 to 1060 millibar range. The CS105 outputs a linear signal of 0 to 2.5 Vdc allowing it to be directly connected to Campbell Scientific dataloggers.

An integral circuit switches 12 volts from the datalogger to the barometer only during measurement, thereby reducing power requirements. Sensor warm-up and measurement time is one second minimum.

The pressure transmitter’s program is as follows:

```
'__________________CS105 Barometric Pressure Sensor
Declarations______________________________
Const PORT2HI = 1                    'Control Port #2 Hi
Const PORT2LO = 0                    'Control Port #2 Lo
Const CS105_OP = 0                   'Delay Analog Measurements only
Const CS105DLAY = 1                          'Actual Delay
Const CS105D_UNITS = 2               'Delay Units (sec)
Const CS105RNG = 0                   'CS105 Range (5000mV)
Const CS105REP = 1                   'CS105 Repetitions
Const CS105SETL = 200                'CS105 Settling Time (usecs)
Const CS105INT = 250                         'CS105 Integration Time (usecs)
Const CS105MULT = 0.184              'CS105 Multiplier
Const CS105OSET = 600                'CS105 Offset
Public CS105Blk(CS105REP)                    'Declare CS105 block
Units CS105Blk = mb                  'CS105 Barometric pressure units are millibars
Alias CS105Blk(1) = BaroPress                'Assign the alias BaroPress to CS105 block
Public CS105_1min(1)                         'Declare CS105 1 minute data
Alias CS105_1min(1) = BaroPress_1min         'Assign alias BaroPress_1min to
CS105_1min(1)
```

```
'____________________CS105 Barometric Pressure Sensor______________________________
'Turn on the CS105 and delay to allow the sensor to stabilize.
Portset (2,PORT2HI)
Delay (CS105_OP,CS105DLAY,CS105D_UNITS)
'Measure the CS105 barometric pressure in millibars.
VoltSe (CS105Blk(),CS105REP,CS105RNG,5,1,CS105SETL,CS105INT,CS105MULT,CS105OSET)
'Turn off the CS105.
Portset (2,PORT2LO)
If S_CS105 = True Then
CallTable CS105_1
GetRecord (CS105_1min(),CS105_1,1)
CallTable CS105_10
EndIf
```

The transmitter's specifications are as follows:

Operating Range

Pressure:	600 mb to 1060 mb
Temperature:	-40°C to +60°C
Humidity:	non-condensing

Accuracy

Total Accuracy***	±0.5 mb @ +20°C
	±2 mb @ 0°C to +40°C
	±4 mb @ -20°C to +45°C
	±6 mb @ -40°C to +60°C
Linearity*:	±0.45 mb @ 20°C
Hysteresis*:	±0.05 mb @ 20°C
Repeatability*:	±0.05 mb @ 20°C
Calibration uncertainty**:	±0.15 mb @ 20°C
Long-Term Stability:	±0.1 mb per year

* Defined as ±2 standard deviation limits of end-point non-linearity, hysteresis error, or repeatability error

** Defined as ±2 standard deviation limits of inaccuracy of the working standard at 1000 mb in comparison to international standards (NIST)

*** Defined as the root sum of the squares (RSS) of end-point non-linearity, hysteresis error, repeatability error and calibration uncertainty at room temperature

General

Dimensions: 9.7 cm x 5.9 cm x 2.1 cm (3.8" x 2.3" x 0.8")

Weight: 110 g (4 oz)

Housing material: anodized aluminum

Supply Voltage: 10 to 30 VDC

Supply Voltage Control: When enabled with an internal jumper, the CS105 is on continually. When disabled, the CS105 can be turned on/off with 5 VDC/0 VDC.

Supply voltage sensitivity: Less than 0.1 mb

Current Consumption: <4 mA (active); <1 μA (quiescent)

Output Voltage: 0 to 2.5 VDC

Warm Up Time: 1 second

Pressure fitting: barbed fitting for 1/8" I.D. tubing

Overpressure limit: 2000 mb

2.3.9 Anemometer (05103-10)

Figure 16. Anemometer (05103-10)

Model 05103-10 Wind Monitor is a sturdy instrument for measuring wind speed and direction in harsh environments. The Wind Monitor's design emphasizes simplicity and lightweight construction. Thermoplastic materials offer improved resistance to corrosion from sea air environments and atmospheric pollutants. The 05103-10 connects directly to Campbell Scientific dataloggers.

The wind speed sensor is a helicoid-shaped, four-blade propeller. Rotation of the propeller produces an AC sine wave; the frequency is directly proportional to the wind speed. The AC signal is induced in a transducer coil by a six-pole magnet mounted on the propeller shaft. The coil is located on the non-rotating central portion of the main mounting assembly, eliminating the need for slip rings and brushes.

Wind direction is sensed by a potentiometer. With the precision excitation voltage from the datalogger applied to the potentiometer element, the output signal is an analog voltage directly proportional to the azimuth angle.

The anemometer's program is as follows:

```
'_______________________05103-10 Wind Monitors
Declarations___________________________________
'Wind Speed for both 05103-10 #1 and #2
Const WSCONF = 1                          'Wind Speed Config (Lo Level)
Const WSOPT = 1                           'Wind Speed Option (Frequency)
Const WSREP = 1                           'Wind Speed Repetitions
Const WSMULT = 0.3528                             'Wind Speed Multiplier
Const WSOSET = 0                          'Wind Speed Offset
'Wind Direction for both 05103-10 #1 and #2
Const WDRNG = 0                           'Wind Direction Range (5000 mV)
Const WDREP = 1                           'Wind Direction Repetitions
Const WDEXCIT = 5000                              'Wind Direction Excitation mVolts
Const WDSETL = 200                                'Wind Direction Settling Time
(usecs)
Const WDINT = 250                         'Wind Direction Integration Time (usecs)
Const WDMULT = 355                                'Wind Direction Multiplier
Const WDOSET = 0                          'Wind Direction Offset
'05103-10 #1
Public WS1Blk(WSREP)                              'Declare Wind Speed #1 block
Units WS1Blk = m_s                        'Wind Speed #1 units are meters per second
Alias WS1Blk(1) = WindSpd1                        'Assign the alias WindSpd1 to Wind
Speed #1 block
Public WD1Blk(WDREP)                              'Declare Wind Direction #1 block
Units WD1Blk = deg                        'Wind Direction #1 units are degrees
Alias WD1Blk(1) = WindDir1                        'Assign the alias WindDir1 to Wind
Direction #1 block
'2 minute data for 05103-10 #1
Public Wind1_2min(4)                              'Declare locations for 05103-10 #1 2
minute averages
Alias Wind1_2min(1) = W1HSpd_2min                 'Assign alias W1HSpd_2min to
Wind1_2min(1)
                                          'Mean Horizontal Wind Speed
Alias Wind1_2min(2) = W1RSpd_2min                 'Assign alias W1RSpd_2min to
Wind1_2min(2)
                                          'Resultant Mean Wind Speed
Alias Wind1_2min(3) = W1RDir_2min                 'Assign alias W1RDir_2min to
Wind1_2min(3)
                                          'Resultant Mean WInd Direction
Alias Wind1_2min(4) = W1StdDir_2min               'Assign alias W1StdDir_2min to
Wind1_2min(4)
                                          'Standard Deviation of Wind Direction
'05103-10 #2
Public WS2Blk(WSREP)                              'Declare Wind Speed #2 block
Units WS2Blk = m_s                        'Wind Speed #2 units are meters per second
```

```
Alias WS2Blk(1) = WindSpd2                          'Assign the alias WindSpd2 to Wind
Speed #2 block
Public WD2Blk(WDREP)                                'Declare Wind Direction #2 block
Units WD2Blk = deg                          'Wind Direction #2 units are degrees
Alias WD2Blk(1) = WindDir2                          'Assign the alias WindDir2 to Wind
Direction #2 block
'2 minute data for 05103-10 #2
Public Wind2_2min(4)                                'Declare locations for 05103-10 #1 2
minute averages
Alias Wind2_2min(1) = W2HSpd_2min                   'Assign alias W2HSpd_2min to
Wind2_2min(1)
                                            'Mean Horizontal Wind Speed
Alias Wind2_2min(2) = W2RSpd_2min                   'Assign alias W2RSpd_2min to
Wind2_2min(2)
                                            'Resultant Mean Wind Speed
Alias Wind2_2min(3) = W2RDir_2min                   'Assign alias W2RDir_2min to
Wind2_2min(3)
                                            'Resultant Mean WInd Direction
Alias Wind2_2min(4) = W2StdDir_2min                 'Assign alias W2StdDir_2min to
Wind2_2min(4)
                                            'Standard Deviation of Wind Direction

  '____________________05103-10 Wind
Monitors________________________________________
  '_________________________05103-10
#1_____________________________________________
  'Measure the Wind Speed in km / heure.
  PulseCount (WS1Blk(),WSREP,1,WSCONF,WSOPT,WSMULT,WSOSET)
  'Measure the Wind Direction in degrees. (0-360 degrees)
  BrHalf (WD1Blk(),WDREP,WDRNG,1,Vx1,1,WDEXCIT,True
,WDSETL,WDINT,WDMULT,WDOSET)
  If  S_05103_1 = True Then
   CallTable Wind1_2
   GetRecord (Wind1_2min(),Wind1_2,1)
   CallTable Wind1_10
  EndIf
  '__________________________05103-10
#2____________________________________________
  'Measure the Wind Speed in km / heure.
  PulseCount (WS2Blk(),WSREP,2,WSCONF,WSOPT,WSMULT,WSOSET)
  'Measure the Wind Direction in degrees. (0-360 degrees)
  BrHalf (WD2Blk(),WDREP,WDRNG,2,Vx1,1,WDEXCIT,True
,WDSETL,WDINT,WDMULT,WDOSET)
  If  S_05103_2 = True Then
      CallTable Wind2_2
      GetRecord(Wind2_2min(),Wind2_2,1)
      CallTable Wind2_10
```

EndIf

Table 2. Anemometer specifications

Specifications

	05103 and 05106	05305
Wind Speed		
Range:	0-134 mph (0-60 m s^{-1})	0-90 mph (0-40 m s^{-1})
Accuracy:	±0.6 mph (±0.3 m s^{-1})	±0.4 mph (±0.2 m s^{-1})
Starting threshold:	2.2 mph (1.0 m s^{-1}) 05103; 2.4 mph (1.1 m s^{-1}) 05106	0.9 mph (0.4 m s^{-1})
Gust survival:	220 mph (100 m s^{-1})	100 mph (45 m s^{-1})
Distance constant (63% recovery):	8.9 ft (2.7 m)	6.9 ft (2.1 m)
Output:	ac voltage (3 pulses per revolution). 1800 rpm (90 Hz) = 19.7 mph (8.8 m s^{-1})	ac voltage (3 pulses per revolution) 1800 rpm (90 Hz) = 20.6 mph (9.2 m s^{-1})
Wind Direction		
Range:	0-360° mechanical, 355° electrical (5° open)	Same
Accuracy:	±3°	±3°
Starting threshold at 10° displacement:	2.2 mph (1.1 m s^{-1})	1.0 mph (0.5 m s^{-1})
Delay distance (50% recovery):	4.3 ft (1.3 m)	3.9 ft (1.2 m)
Damping ratio:	0.25	0.45
Damped natural wavelength:	24.3 ft (7.4 m)	16.1 ft (4.9 m)
Undamped natural wavelength:	23.6 ft (7.2 m)	14.4 ft (4.4 m)
Output:	Analog dc voltage from potentiometer - resistance 10 KΩ, linearity 0.25%, life expectancy 50 million revolutions.	Same
Power	Switched excitation voltage supplied by the datalogger.	Same

Specifications (continued)

	05103 and 05106	05305
Operating Temperature	-50° to +50°C, assuming non-riming conditions	-50° to +50°C, assuming non-riming conditions
Dimensions		
Overall:	14.6″ H x 21.7″ L (37 cm x 55 cm)	15.0″ H x 25.6“ L (38 cm x 65 cm)
Main housing Diameter:	2.0“ (5 cm)	Same
Propeller Diameter:	7.1″ (18 cm)	7.9″ (20 cm)
Mounting Pipe:	1.34″ (34 mm) OD; standard 1.0″ IPS schedule 40	Same
Weight (shipping approx.)	3.2 lbs (5.5 lbs); 1.5 kg (2.3 kg)	2.5 lbs (5.5 lbs); 1.1 kg (2.3 kg)

Manufactured by RM Young (Traverse City , MI) and cabled by Campbell Scientific for use with our dataloggers.

2.3.10 Temperature and Humidity (HMP45C)

Figure 17. HMP45C

The HMP45C is a rugged, accurate temperature/RH probe manufactured by Vaisala Inc.,that is ideal for long-term, unattended applications. The probe uses a capacitive polymer H chip to measure RH and a PRT to measure temperature. Power is only supplied to the HMP45C during measurement, which reduces the current drain. The probe connects directly to Campbell Scientific dataloggers that have a switched 12 V terminal such as the CR10X, CR23X, or CR5000. Dataloggers that do not have a switched 12 V terminal such as the CR510, CR10, or CR7 require an SW12V Switched 12 V device.

For optimum results, the HMP45C should be recalibrated annually. An optional removable sensor head, model HMP41, can replace the HMP45C's sensing element while the sensing element is getting recalibrated. The HMP41 is compatible with any HMP45C. A radiation shield (Model 41003-2 or UT12VA) should be used when the HMP45C is exposed to sunlight.

The HMP45C’s program is as follows:

```
'______________HMP45C Temperature & Relative Humidity Sensor
Declarations______________________
Const PORT8HI = 1                  'Control Port #1 Hi
Const PORT8LO = 0                  'Control Port #1 Lo
Const HMPD_OP = 0                  'Delay Analog Measurements only
Const HMPDLAY = 150                        'Actual Delay
Const HMPD_UNITS = 1               'Delay Units (msec)
Const HMPRNG_T = 1                         'Temp Range (1000mV)
Const HMPREP_T = 1                         'Temp Repetitions
Const HMPSETL_T = 200              'Temp Settling Time (usecs)
Const HMPINT_T = 250                       'Temp Integration Time (usecs)
Const HMPMULT_T = 0.1              'Temp Multiplier
Const HMPOSET_T = -40              'Temp Offset
Public HMPBlk_T(HMPREP_T)                  'Declare temp block
Units HMPBlk_T = degC              'Temp units are degree Celsius
Alias HMPBlk_T(1) = AirTemp                'Assign the alias AirTemp to Temp block
Const HMPRNG_R = 1                         'RH Range (1000mV)
Const HMPREP_R = 1                         'RH Repetitions
Const HMPSETL_R = 200              'RH Settling Time (usecs)
Const HMPINT_R = 250                       'RH Integration Time (usecs)
Const HMPMULT_R = 0.1              'RH Multiplier
Const HMPOSET_R = 0                        'RH Offset
Public HMPBlk_R(HMPREP_R)                  'Declare RH block
Units HMPBlk_R = percent           'RH units are percent
Alias HMPBlk_R(1) = RH             'Assign the alias RH to RH block
'DewPoint calculation declarations

Dim DPa, DPb, Es, Ws, W, E         'Declare locations used for calculation of Dew Point
Public DewPoint                    'Declare DewPoint
Units DewPoint = degC                      'DewPoint units are degrees Celsius
'1 minute data
Public HMP1min(3)                  'Declare locations for HMP45C 1 minute averages
Alias HMP1min(1) = AirTemp_1min            'Assign alias AirTemp_1min to HMP1min(1)
Alias HMP1min(2) = RH_1min                 'Assign alias HMPDate_1min to HMP1min(2)
Alias HMP1min(3) = DewPoint_1min           'Assign alias HMPDate_1min to HMP1min(3)
'Set up arrays for the Sample on Min/Max instruction
Dim DPoint(2)
Alias DPoint(1) = Air_Temp
Alias DPoint(2) = R_H
Dim Tmp(2)
Alias Tmp(1) = RH_
Alias Tmp(2) = Dew_Point
Dim RHum(2)
Alias RHum(1) = AirTemp_
Alias RHum(2) =DewPoint_
```

```
'_____________________HMP45C Temp & RH
Sensor______________________________________
'Turn on the HMP45C and delay to allow the probe to stabilize.
 Portset (8 ,PORT8HI)
 Delay (HMPD_OP,HMPDLAY,HMPD_UNITS)
 'Measure the Temperature in degrees Celcius.
      'VoltSe
(HMPBlk_T(),HMPREP_T,HMPRNG_T,3,True,HMPSETL_T,HMPINT_T,HMPMULT_T,HM
POSET_T)
 VoltDiff
(HMPBlk_T(),HMPREP_T,HMPRNG_T,14,True,HMPSETL_T,HMPINT_T,HMPMULT_T,H
MPOSET_T)
 'Measure the Relative Humidity in a percentage value. (0-100%)
 'VoltSe
(HMPBlk_R(),HMPREP_R,HMPRNG_R,4,True,HMPSETL_R,HMPINT_R,HMPMULT_R,H
MPOSET_R)
  VoltDiff
(HMPBlk_R(),HMPREP_R,HMPRNG_R,15,True,HMPSETL_R,HMPINT_R,HMPMULT_R,H
MPOSET_R)
 'Turn off the HMP45C.
  Portset (8 ,PORT8LO)
  If S_HMP45C = True Then
  If RH > 100 then
  RH = 100
  Endif
 'If CS105 barometric pressure sensor is being measured use the formula from the manual
  '"Principles of Meteorological Analysis" written by W.J. Saucier.
  If S_CS105 = False Then
  BaroPress = 1013 * Exp(-(0.03396796159 * Altitude)/ (AirTemp + 273.15))
  Endif
  If AirTemp <= 0 Then
     DPa = 21.875
     DPb = 265.5
     Else
     DPa = 17.2694
     DPb = 237.3
     EndIf
     Es = 6.1078 * (Exp( (DPa * AirTemp) / (DPb + AirTemp) ) )
     Ws = 0.62197 * (Es / (BaroPress - Es) )
     W = (RH / 100) * Ws
     E = BaroPress * (W / (W + 0.62197) )
     DewPoint = (DPb * LOG( (E / 6.1078) ) ) / (DPa - LOG( (E / 6.1078) ) )
     Air_Temp = AirTemp
     AirTemp_ = AirTemp
     R_H = RH
     RH_ = RH
     Dew_Point = DewPoint
     DewPoint_ = DewPoint
```

```
CallTable HMP45C1
GetRecord (HMP1min(),HMP45C1,1)
CallTable HMP45C10
EndIf
```

Table 3. HMP45C specifications

Specifications

Probe Dimensions: 10 inches (25.4 cm) length, 1 inch (2.5 cm) diameter

Filter: 0.2 µm Teflon® membrane

Filter Diameter: 0.75 inches (1.9 cm)

RELATIVE HUMIDITY

Sensor: HUMICAP® H-chip

Measurement Range: 0.8 to 100% RH, non-condensing

Output Signal Range: 0.008 to 1 Vdc

Accuracy at 20°C:
against factory reference: ±1% RH
field calibrated against references: ±2% RH (0-90% RH)
field calibrated against references: ±3% RH (90-100% RH)

Temperature Dependence: ±0.05% RH/°C

Typical Long-Term Stability: Better than 1% RH per year

Response Time (at 20°C, 90% response): 15 s with membrane filter

Settling Time: 500 ms

Supply Voltage: 12 Vdc Nominal

Current Consumption: 4 mA (Active)

Operating Temperature: -40° to +60°C

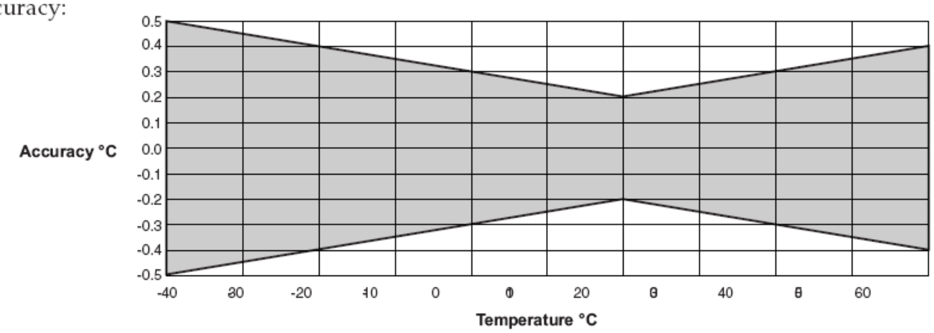
TEMPERATURE

Sensor: 1000 PRT

Measurement Range: -39.2° to +60°C

Output signal range: 0.008 to 1.0 V

Accuracy:



The HMP45C is manufactured by Vaisala, Inc. (Woburn, MA) but cabled and modified by Campbell Scientific for use with our dataloggers.

2.3.11 Visibility & Precipitation (WIVIS)

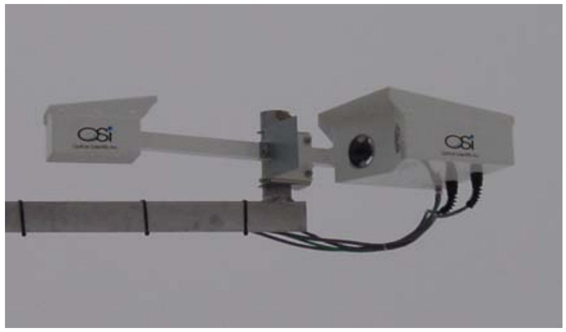


Figure 18. Optical Weather Identifier (OWI)

The Optical Weather Identifier (OWI) is an excellent optical instrument for precipitation rate and type identification. The sensor uses the same superior present weather capabilities as our top-of-the-line WIVIS but does not offer visibility detection. When accurate discrimination of rain and snow is required, the OWI is the best solution. This sensor works well in a wide variety of weather measurement applications, especially in road weather and aviation. OWI is a key component of many critical weather systems for users such as the USAF, FAA, and many transportation authorities worldwide.

OWI offers the user a present weather sensor with performance that rivals the LEDWI but at a lower cost. Based on the design concepts of the field-proven LEDWI and WIVIS, it has excellent performance but with slightly reduced EMI protection and is without hood heaters.

Many of the OWI features make the instrument excellent for efficient, real-time, remote weather sensing. The OWI provides present weather information such as precipitation rate and type in RS-232 format. You can link the sensor directly to your PC. The output comes in both NWS and WMO code formats. The OWI's small weight and size make it easy to handle, and U-bolt mounting gives the user a flexible method of installation. The OWI plays a crucial part in road weather applications.

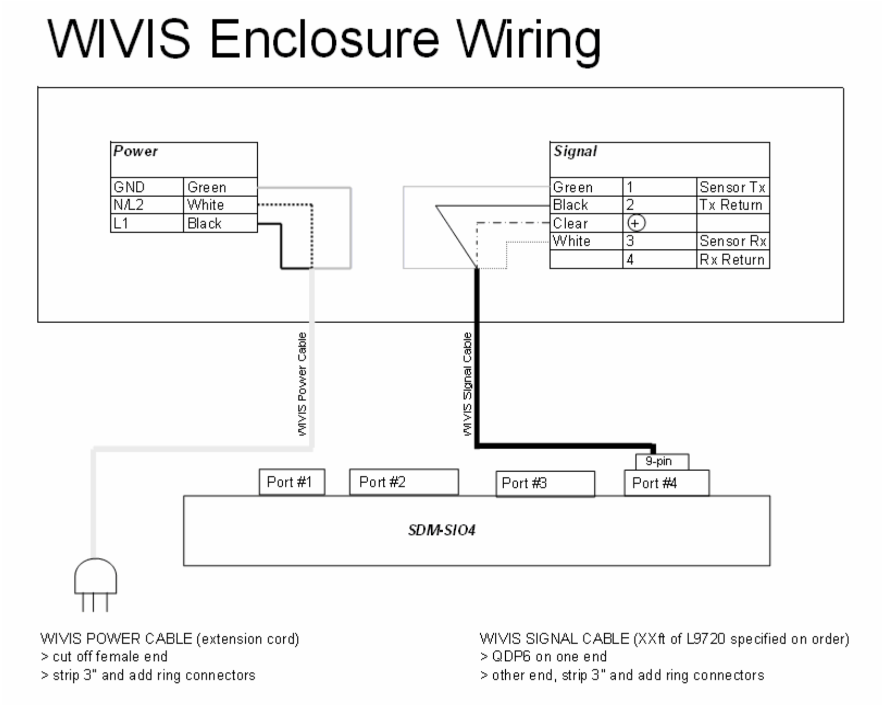


Figure 19. WIVIS Enclosure Wiring

The WIVIS' program is as follows:

```
'____________________WIVIS Intelligent Road Weather Sensor
Declarations_____________________
'The WIVIS Sensor requires an SDM-SIO4 and a Custom Cable (3 cond bare lead to 9 pin
female dSub)
'Data is acquired by sending an "D" to the sensor.  carriage Return and Line feed is not required.
'Sio4Setup:     WIVIS Fields:       Definitions:
'f1             Pres_Weather        Present Weather
'f2             PrecipRate          Precipitation Rate
'f3             Status              Status
'f4             Visibility                  Visibility
'f5             Checksum            Checksum
'f6             BackLight           Background Light
'f7             PrecipTotal         Precipitation Accumulation
'f8             15minW_Code                 15 minute past weather code
'f9             60minW_Code                 60 minute past weather code
'
'Sio4 programming:
'strst 3 "D" 'Message to send to sensor to request data.
'fltst 4 "T[^BW]xN39zX"  Receive filter string for WIVIS data from sensor to SDM-SI04
Public ScanCtrWivis                          'A counter of numbers of scan for WIVIS
Const  StringNumber3   = 3                   'String #3
Const FilterString4 = 9004                   'Filter string #4
Const  strst3Bytes=11                        '# bytes in string #3
```

```
Const  fltst4Bytes=21                          '# bytes in filter string #4
Dim strst3Byte(strst3Bytes)                    'Declare strst3byte
Dim fltst4Byte(fltst4Bytes)                    'Declare fltst4byte
Public DataPollPort4                           'If greater than 0, data is available on port 4
Public WPollingLoop                            'Counter for how many times the CR5000
                                               'tried to retrieve data from SDM-SIO4 for
WIVIS
Public WGotData                                'A counter which indicates the CR5000
received
                                               'data from WIVIS
Const WIVISBytes=39                                  'Number of bytes that will be sent
from WIVIS
Public RawWIVISData(WIVISBytes)                      'Declare RawWIVISData
Public WCode_Pres                              'Declare scaled present weather code
Public PrecipRate                              'Declare scaled precipitation rate
Units PrecipRate = mmph                        'PrecipRate units are millimeters per hour
Public Status1                           'Declare scaled Status1
Public Status2                           'Declare scaled Status2
Public Status3                           'Declare scaled Status3
Public Status4                           'Declare scaled Status4
Public Visibility                              'Declare scaled Visibility
Units Visibility = km                          'Visibility units are kilometers
Public Checksum1                               'Declare scaled Checksum1
Public Checksum2                               'Declare scaled Checksum2
Public BackLight                               'Declare Backlight
Public PrecipTotal                             'Declare precipitation accumulation
Units PrecipTotal = mm                         'PrecipTotal units are millimeters
Public WCode_15min                             'Declare scaled present weather code
Public WCode_60min                             'Declare scaled present weather code


' ____________________CR5000 Battery Voltage Declarations_________________
Public BattVolt                                'Declare variable for battery voltage
Units BattVolt = volts                   'BattVolt units are volts
Public Flag(8)


'________________________WIVIS Intelligent Road Weather
Sensor___________________________

 If S_WIVIS = True Then
 ScanCtrWivis = ScanCtrWivis + 1
 i=0
 DO
 'Flush Receive buffers and restart receive filters on Port 4.
 Call(FAndF4)
 'Command SDM-SI04 Port4 to transmit sensor polling command "D":
```

```
SIO4(NotUsed,OneRep,Sio4Address00,Port4,SendString,StringNumber3,UnusedParameter,No
Values,UnityMultiplier,NoOffset)
      Delay(DelayProcessing,3,sec)
     'Clear the RawWIVISData locations before retrieving the data.
      Move(RawWIVISData(),WIVISBytes,0,1)

     'Retrieve Data
    For WPollingLoop= 1 To 20
SIO4(DataPollPort4,OneRep,Sio4Address00,Port4,PollForData,UnusedParameter,UnusedParam
eter,1,UnityMultiplier,NoOffset)
        Delay(DelayProcessing,200,mSec)
If DataPollPort4>38 Then
SIO4(RawWIVISData(),OneRep,Sio4Address00,Port4,SendByteToLgr,UnusedParameter,Unuse
dParameter,WIVISBytes,UnityMultiplier,NoOffset)
        Delay(DelayProcessing,200,mSec)
        ExitFor
Else
Delay(DelayProcessing,1000,mSec)
        EndIf
Next WPollingLoop
        i=i+1
Loop Until (RawWIVISData(1) > 0) Or (i>2)

        'Increment a counter if the CR5000 has received data from the WIVIS sensor.
 IF RawWIVISData(1) > 0 Then
 WGotData=WGotData+1
 EndIf
        'Scale WIVIS data RawWIVISData into desired format: (Convert from decimal ascii to
char. ascii)
        'Convert Present Weather Code.
        'Check if wivis byte 3 is between 0 and 9
 If (RawWIVISData(3) > 47 and RawWIVISData(3) < 58) Then
 WCode_Pres = (((RawWIVISData(3) - 48) * 10) + (RawWIVISData(4)-48))

        'Check if wivis byte 3 is -
 ElseIf RawWIVISData(3) = 45 Then
 WCode_Pres = 97
        'Check if wivis byte 3 is E
 ElseIf RawWIVISData(3) = 69 Then
 WCode_Pres = 98
        'Check if wivis byte 3 is C
 ElseIf RawWIVISData(3) = 67 Then
 WCode_Pres = 99
 Else
 WCode_Pres = -999
 EndIf

        'Convert 15 minute Weather Code.
```

```
	'Check if wivis byte 35 is between 0 and 9
 If (RawWIVISData(35) > 47 and RawWIVISData(35) < 58) Then
 WCode_15min = (((RawWIVISData(35) - 48) * 10) + (RawWIVISData(36)-48))

	'Check if wivis byte 35 is -
 ElseIf RawWIVISData(35) = 45 Then
 WCode_15min = 97
	'Check if wivis byte 35 is E
 ElseIf RawWIVISData(35) = 69 Then
 WCode_15min = 98
	'Check if wivis byte 35 is C
 ElseIf RawWIVISData(35) = 67 Then
 WCode_15min = 99
 Else
 WCode_15min = -999
 EndIf

'Convert 60 minute Weather Code.
	'Check if wivis byte 37 is between 0 and 9
 If (RawWIVISData(37) > 47 and RawWIVISData(37) < 58) Then
 WCode_60min = (((RawWIVISData(37) - 48) * 10) + (RawWIVISData(38)-48))

	'Check if wivis byte 37 is -
 ElseIf RawWIVISData(37) = 45 Then
 WCode_60min = 97
'Check if wivis byte 37 is E
 ElseIf RawWIVISData(37) = 69 Then
 WCode_60min = 98
	'Check if wivis byte 37 is C
 ElseIf RawWIVISData(37) = 67 Then
 WCode_60min = 99
 Else
  WCode_60min = -999
  EndIf

 'Convert Precipitation Rate data.
 'Check if at least one of bytes 6 or 7 have a value of 0-9.
  If ( (RawWIVISData(6) > 47 and RawWIVISData(6) < 58) OR (RawWIVISData(7) > 47 and
RawWIVISData(7) < 58)) Then
 'If true than continue
 'Check if wivis byte 6 is .
  If RawWIVISData(6) = 46 Then
  PrecipRate = ( ((RawWIVISData(7) -48) / 10) + ((RawWIVISData(8) - 48) / 100) +
((RawWIVISData(9) - 48) /1000)  )
 'Check if wivis byte 7 is .
  ElseIf RawWIVISData(7) = 46 Then
  PrecipRate = ( ((RawWIVISData(6) -48)) + ((RawWIVISData(8) - 48) / 10) +
((RawWIVISData(9) - 48) /100)  )
```

```
 'Check if wivis byte 8 is .
  ElseIf RawWIVISData(8) = 46 Then
  PrecipRate = ( ((RawWIVISData(6) -48) * 10) + ((RawWIVISData(7) - 48)) +
((RawWIVISData(9) - 48) /10)  )
        'Check if wivis byte 9 is .
  lseIf RawWIVISData(9) = 46 Then
  PrecipRate = ( ((RawWIVISData(6) -48) * 100) + ((RawWIVISData(7) - 48) *10) +
((RawWIVISData(8) - 48))  )
 'Check if all bytes 6-9 are values between 0-9
  ElseIf ( (RawWIVISData(6) > 47 and RawWIVISData(6) < 58) AND (RawWIVISData(7) > 47
and RawWIVISData(7) < 58))       Then
  If ((RawWIVISData(8) > 47 and RawWIVISData(8) < 58) AND (RawWIVISData(9) > 47 and
RawWIVISData(9) < 58)) Then
  PrecipRate = ( ((RawWIVISData(6) -48) * 1000) + ((RawWIVISData(7) - 48) *100) +
((RawWIVISData(8) - 48) * 10) +   ((RawWIVISData(9) - 48)) )
  EndIf
  Else
  PrecipRate = -999
  EndIf
  Else
  PrecipRate = -999
  EndIf

 'Convert Visibility data.
 'Check if at least one of bytes 16-17 have a value of 0-9.
  If ( (RawWIVISData(16) > 47 and RawWIVISData(16) < 58) OR (RawWIVISData(17) > 47
and RawWIVISData(17) < 58)) Then
   'If true than continue
   'Check if wivis byte 16 is .
    If RawWIVISData(16) = 46 Then
    Visibility = ( ((RawWIVISData(17) -48) / 10) + ((RawWIVISData(18) - 48) / 100) +
((RawWIVISData(19) - 48) /1000)  )
   'Check if wivis byte 17 is .
    ElseIf RawWIVISData(17) = 46 Then
    Visibility = ( ((RawWIVISData(16) -48)) + ((RawWIVISData(18) - 48) / 10) +
((RawWIVISData(19) - 48) /100)  )
   'Check if wivis byte 18 is .
    ElseIf RawWIVISData(18) = 46 Then
    Visibility = ( ((RawWIVISData(16) -48) * 10) + ((RawWIVISData(17) - 48)) +
((RawWIVISData(19) - 48) /10)  )
   'Check if wivis byte 19 is .
    ElseIf RawWIVISData(19) = 46 Then
    Visibility = ( ((RawWIVISData(16) -48) * 100) + ((RawWIVISData(17) - 48) *10) +
((RawWIVISData(18) - 48))  )
   'Check if all bytes 16-19 are values between 0-9
ElseIf ( (RawWIVISData(16) > 47 and RawWIVISData(16) < 58) AND (RawWIVISData(17) >
47 and RawWIVISData(17) < 58))  Then
        If ( (RawWIVISData(18) > 47 and RawWIVISData(18) < 58) AND (RawWIVISData(19)
```

```
> 47 and RawWIVISData(19) < 58))     Then
    Visibility = ( ((RawWIVISData(16) -48) * 1000) + ((RawWIVISData(17) - 48) *100) +
((RawWIVISData(18) - 48) * 10) +    ((RawWIVISData(19) - 48)) )
    EndIf
    Else
    Visibility = -999
    EndIf
    Else
    Visibility = -999
    EndIf

   'Convert Status data.
   'Check if wivis byte 11 is between 0-9
    If RawWIVISData(11) > 47 and RawWIVISData(11) < 58 Then
    Status1 = (RawWIVISData(11) - 48)
   'Check if wivis byte 11 is between A-F
    ElseIf RawWIVISData(11) > 64 and RawWIVISData(11) < 71 Then
    Status1 = (RawWIVISData(11) - 55)
    Else
    Status1 = -999
    EndIf
   'Check if wivis byte 12 is between 0-9
    If RawWIVISData(12) > 47 and RawWIVISData(12) < 58 Then
    Status2 = (RawWIVISData(12) - 48)
   'Check if wivis byte 12 is between A-F
    ElseIf RawWIVISData(12) > 64 and RawWIVISData(12) < 71 Then
    Status2 = (RawWIVISData(12) - 55)
    Else
    Status2 = -999
    EndIf
   'Check if wivis byte 13 is between 0-9
    If RawWIVISData(13) > 47 and RawWIVISData(13) < 58 Then
    Status3 = (RawWIVISData(13) - 48)
   'Check if wivis byte 13 is between A-F
    ElseIf RawWIVISData(13) > 64 and RawWIVISData(13) < 71 Then
    Status3 = (RawWIVISData(13) - 55)
    Else
    Status3 = -999
    EndIf
   'Check if wivis byte 14 is between 0-9
    If RawWIVISData(14) > 47 and RawWIVISData(14) < 58 Then
    Status4 = (RawWIVISData(14) - 48)
   'Check if wivis byte 14 is between A-F
    ElseIf RawWIVISData(14) > 64 and RawWIVISData(14) < 71 Then
    Status4 = (RawWIVISData(14) - 55)
    Else
    Status4 = -999
    EndIf
```

```
  'Convert Checksum data.
  'Check if wivis byte 20 is between 0-9
   If RawWIVISData(20) > 47 and RawWIVISData(20) < 58 Then
   Checksum1 = (RawWIVISData(20) - 48)
  'Check if wivis byte 20 is between A-F
   ElseIf RawWIVISData(20) > 64 and RawWIVISData(20) < 71 Then
   Checksum1 = (RawWIVISData(20) - 55)
   Else
   Checksum1 = -999
   EndIf
  'Check if wivis byte 21 is between 0-9
   If RawWIVISData(21) > 47 and RawWIVISData(21) < 58 Then
  Checksum2 = (RawWIVISData(21) - 48)
  'Check if wivis byte 21 is between A-F
   ElseIf RawWIVISData(21) > 64 and RawWIVISData(21) < 71 Then
   Checksum2 = (RawWIVISData(21) - 55)
   Else
   Checksum2 = -999
   EndIf
  'Convert Background Light
  'Check if wivis byte 23 is a value between
   If RawWIVISData(23) > 47 and RawWIVISData(23) < 58 Then
   BackLight = ( ((RawWIVISData(23) - 48) * 100) + ((RawWIVISData(24) - 48) * 10) +
((RawWIVISData(25) -48)) )
   ElseIf RawWIVISData(23) = 45 Then
   BackLight = ( -1 * ( ((RawWIVISData(24) -48) * 10) + ((RawWIVISData(25) -48)) ) )
   Else
   BackLight = -999
   EndIf
  'Convert Precipitation Accumulation
  'Check if wivis byte 27 is between 0-9 and if byte 30 is .
   If (RawWIVISData(27) > 47 and RawWIVISData(27) < 58) Then
   If RawWIVISData(30) = 46 Then
  'Check if wivis byte 28 is between 0-9
   If RawWIVISData(28) > 47 and RawWIVISData(28) < 58 Then
         PrecipTotal = ( ((RawWIVISData(27) - 48) * 100) + ((RawWIVISData(28) - 48) * 10) +
((RawWIVISData(29) - 48)) +   ((RawWIVISData(31) - 48) / 10) + ((RawWIVISData(32) -48) /
100) + ((RawWIVISData(33) -48) / 1000) )
   Else
   PrecipTotal = -999
   EndIf
   EndIf
   Else
   PrecipTotal = -999
   EndIf
  'Call Tables to output the WIVIS Data
   CallTable WIVIS_10
   CallTable ChkWIVIS
```

EndIf

NextScan

Table 4. WIVIS specifications

Performance Specification	
Present Weather Codes	More than 50 NWS and WMO codes
Rain Dynamic Range	0.1 to 3000 mm/hr
Snow Dynamic Range	0.01 to 50 mm/hr (Water equivalent)
Electronic Specification	
Power Requirements	Factory selectable, 100/115/220/230 VAC, 50/60 Hz @ 80 VA-12 VDC optional, 20 VA nominal
Signal Output	RS-232 ASCII
Environmental	
Temperature	-40 to 50 C
Humidity	0-100%
Precipitation Dust	NEMA-4 type protection
Physical	
Head Size	737 x 102 x 254 mm (29 x 4 x 10 inches)
Head Weight	4.5 kg (10 lbs.)
Enclosure Size	457 x 305 x 230 mm (18 x 12 x 9 inches)
Enclosure Weight	11 kg (24.2 lbs)
Head & Enclosure Cable Length	5m (16.4 ft)

2.3.12 Radiometer (CNR1)

Figure 20. Radiometer

The CNR1 net radiometer is manufactured by Kipp & Zonen for applications requiring research-grade performance. The radiometer measures the energy balance between incoming short-wave and long-wave IR radiation versus surface-reflected short-wave and outgoing long-wave IR radiation. It consists of a pyranometer and pyrgeometer pair that faces upward and a complementary pair that faces downward. The pyranometers and pyrgeometers measure short-wave and far infrared radiation, respectively. All four sensors are calibrated to an identical sensitivity coefficient. The CNR1 also includes an RTD to measure the radiometer's internal temperature, a 4WPB100 module to interface the RTD with the datalogger, and a heater that can be used to prevent condensation.

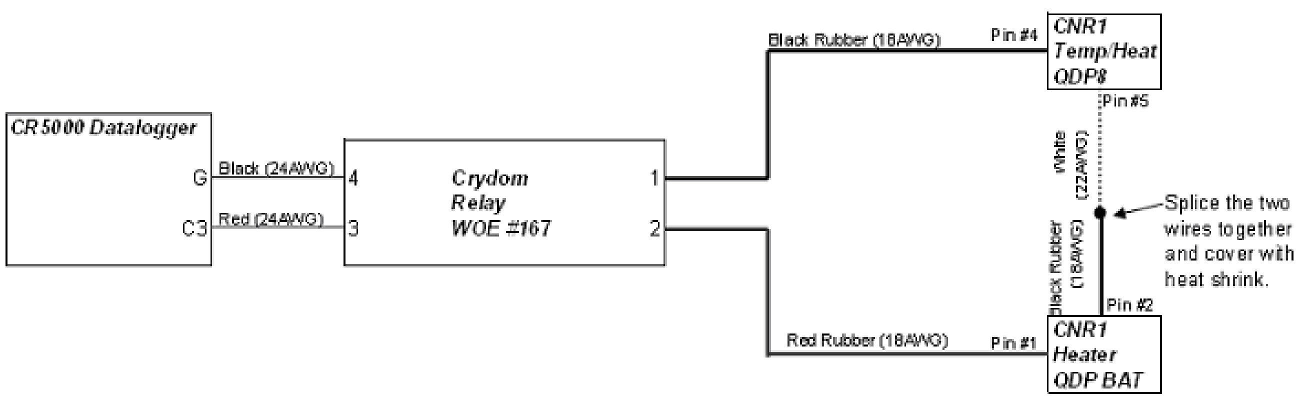


Figure 21. CNR1 Relay Wiring

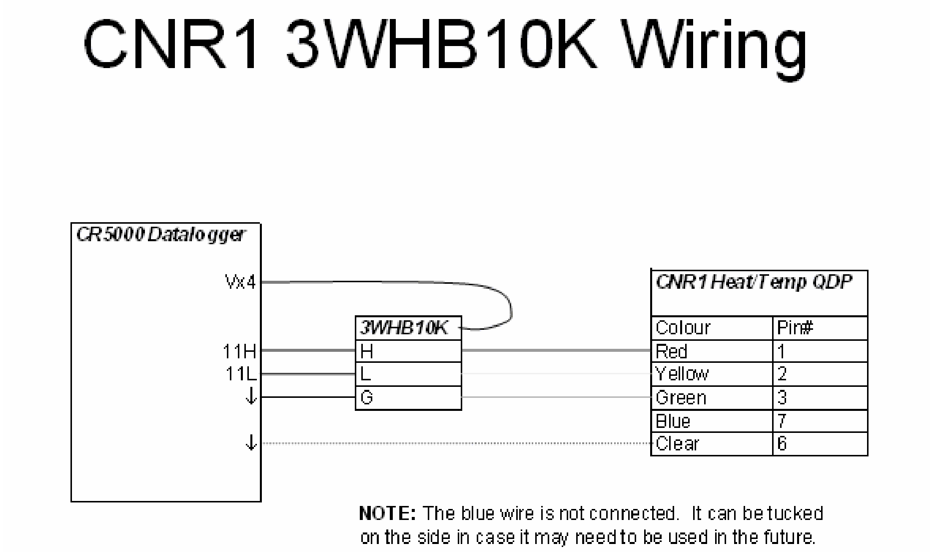


Figure 22. CNR1 3WHB10K Wiring

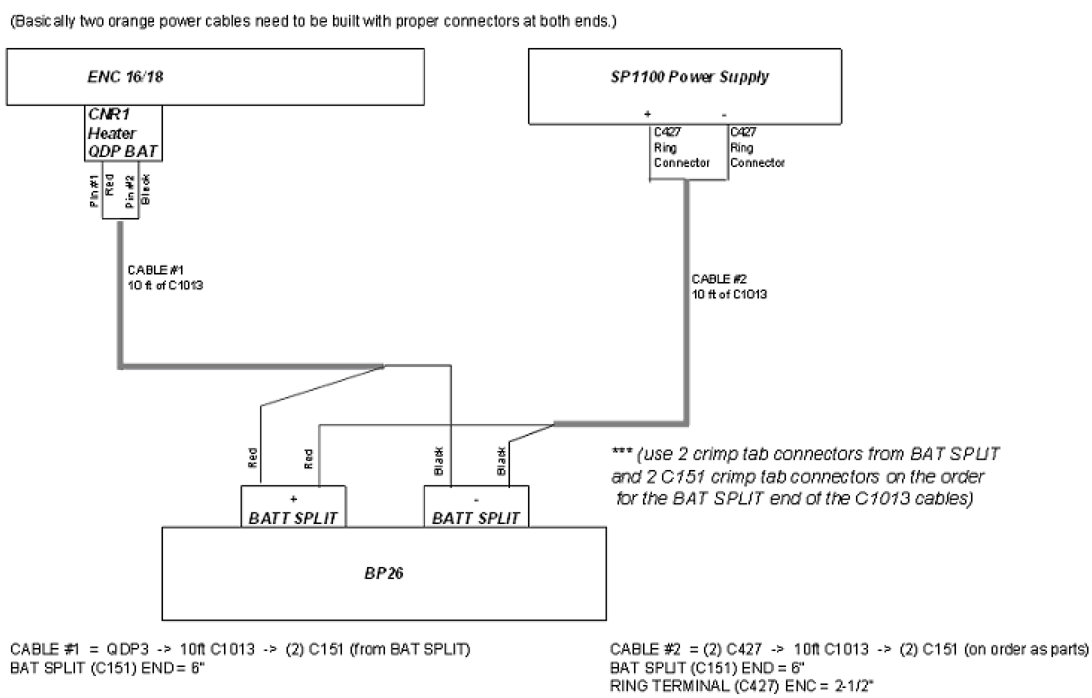


Figure 23. CNR1 heater cable connections

The CNR1's program is as follows:

'______________________CNR1 Net Radiometer Declarations__________________________________

```
Const CNRRNG = 5                    'CNR1 Range (Auto)
Const CNRREP = 4                    'CNR1 Repetitions
Const CNRSETL = 200                         'CNR1 Settling Time (usecs)
Const CNRINT = 250                  'CNR1 Integration Time (usecs)
Const CNRMULT = 1                           'CNR1 Multiplier
Const CNROSET = 0                   'CNR1 Offset
public CNRBlk(CNRREP)                       'Declare CNR1 Solar block
Dim CM3_UPmV                                'Delcare location used for calculations
Public CM3_UPmV                     'Declare CM3_UPmV
Dim CG3_UPmV                                'Delcare location used for calculations
Public Dim CG3_UPmV                         'Declare Dim CG3_UPmV
Dim CM3_DOWNmV                              'Delcare CM3_DOWNmV
Public Dim CM3_DOWNmV                       'Delcare location used for calculations
Dim CG3_DOWNmV                      'Delcare location used for calculations
Public Dim CG3_DOWNmV                       'Delcare Dim CG3_DOWNmV
Public ShortWaveUP                  'Declare upward short wave
Public ShortWaveDOWN                        'Declare downward short wave
Public LongWaveUP                   'Declare upward long wave
Public LongWaveDOWN                         'Declare downward long wave
Units ShortWaveUP = W_m2            'Upward short wave units are watts per meter
squared
Units ShortWaveDOWN = W_m2                  'Downward short wave units are watts per
meter squared
Units LongWaveUP = W_m2             'Upward long wave units are watts per meter
squared
Units LongWaveDOWN = W_m2                   'Downward long wave units are watts per
meter squared
'******************************************************************************
*************************************
'The sensitivity will be different for each CNR1 and therefore the program will need to be
modified for
'each station and when the sensor is exchanged.
'(9.61 mV/kW/m^2 refers to the CNR1 with S/N 010332.)
Const CNRSENS = 10.51                       'CNR1 Sensitivity (uV / W / m^2)
'******************************************************************************
*************************************
Const CNRTCON = 5.67E-08                    'CNR1 Temperature Factor
Const CNRTRNG = 2                   'CNR1 PT100 Range (200mV)
Const CNRTREP = 1                   'CNR1 PT100 Repetitions
Const CNRTEXCIT = 4300                      'CNR1 PT100 Excitation mVolts
Const CNRTSETL = 200                        'CNR1 PT100 Settling Time (usecs)
Const CNRTINT = 250                         'CNR1 PT100 Integration Time (usecs)
Const CNRTMULT = 100.93                     'CNR1 PT100 Multiplier
Const CNRTOSET = 0                          'CNR1 PT100 Offset
Dim CNRTBlk(CNRTREP)                        'Declare CNR1 Temp block
public Rs_Ro                        'Declare location used for calculations
Public CNRTEMP                              'Declare CNR1 Temperature
Units CNRTEMP = degK                        'CNR1 temperature units are degrees Kalvin
```

```
Public Albedo                       'Declare CNR1 Albedo
Public NetSolar                          'Declare CNR1 Net Solar Radiation
Public NetInfraRed                  'Declare CNR1 Net InfraRed Radiation
Public SkyTemp                           'Declare CNR1 Sky Temperature
Public GndTemp                           'Delcare CNR1 Ground Temperature
Public NetTotalRad                  'Declare CNR1 Net Total Radiation
Dim Port3OnOff                           'Control Port #3 Settings

'________________________CNR1 Net
Radiometer______________________________________
   'Measure the PT100 Temperature sensor of the CNR1 in degrees K.
   BrHalf3W
(CNRTBlk(),CNRTREP,CNRTRNG,21,Vx4,1,CNRTEXCIT,True,CNRTSETL,CNRTINT,CNR
TMULT,CNRTOSET)
   Rs_Ro = CNRTBlk(1)
   PRT (CNRTEMP,1,Rs_Ro,1,273.15)
   'Measure the voltages output by the CNR1's 4 separate components.
   VoltDiff
(CNRBlk(),CNRREP,CNRRNG,7,True,CNRSETL,CNRINT,CNRMULT,CNROSET)
   'Define the voltages better so that they're easier to work with.
   CM3_UPmV = CNRBlk(1)
   CG3_UPmV = CNRBlk(2)
   CM3_DOWNmV = CNRBlk(3)
   CG3_DOWNmV = CNRBlk(4)
   'Convert the shortwave mVolt readings from the CM3s to an energy in W/m^2.
   ShortWaveUP = (CM3_UPmV * 1000 / CNRSENS)
   ShortWaveDOWN = (CM3_DOWNmV * 1000 / CNRSENS)
   'Convert the longwave mVolt readings from the CG3s to an energy in W/m^2.
   LongWaveUP = ((CG3_UPmV * 1000 / CNRSENS) + (CNRTCON * (CNRTEMP^4)))
   LongWaveDOWN = ((CG3_DOWNmV * 1000 / CNRSENS) + (CNRTCON *
(CNRTEMP^4)))
   If ShortWaveUP = 0 Then
   Albedo = 0
   Else
   Albedo = (ShortWaveDOWN / ShortWaveUP)
   EndIf
   NetSolar = (ShortWaveUp - ShortWaveDOWN)
   NetInfraRed = (LongWaveUP - LongWaveDOWN)
   SkyTemp = ((LongWaveUP/CNRTCON)^0.25) - 273.15
   GndTemp = ((LongWaveDOWN/CNRTCON)^0.25) - 273.15
   NetTotalRad = (ShortWaveUP + LongWaveUP - ShortWaveDOWN - LongWaveDOWN)
   'Turn on heater at 5:00 pm and turn it off at 8:00 am.  This is preliminary and must be
optimised.
   If ShortWaveUP <= 50 Then Port3OnOff = 1
   If ShortWaveUP > 50 Then Port3OnOff = 0
   Portset (3,Port3OnOff)
   If S_CNR1 = True Then
```

```
CallTable CNR1
EndIf
```

Table 5. CNR1 specifications

Specifications

Sensor:	Kipp & Zonen's CM3 ISO-class, thermopile pyranometer, CG3 pyrgeometer, PT100 RTD
Spectral response:	305 to 2800 nm (pyranometer), 5000 to 50,000 nm (pyrgeometer)
Response time:	18 seconds
Typical sensitivity range:	7 to 15 $\mu V\ W^{-1}\ m^2$
Output range:	0 to 25 mV (pyranometer), ±5 mV (pyrgeometer)
Expected accuracy for daily totals:	±10%
Directional error:	< 25 W m^{-2} (pyranometer)
Heating resistor:	24 Ohms, 6 W at 12 Vdc
Operating temperature:	-40° to 70°C
Weight:	8.8 lbs (4 kg)
Dimensions:	9.1″ x 3.1″ x 6.1″ (23.2 cm x 8.0 cm x 15.6 cm)
Datalogger Requirements	Six differential or four single-ended and two differential analog channels
CE Compliance:	CE compliant under the European Union's EMC Directive

2.3.13 Intelligent Road Sensor (IRS21)

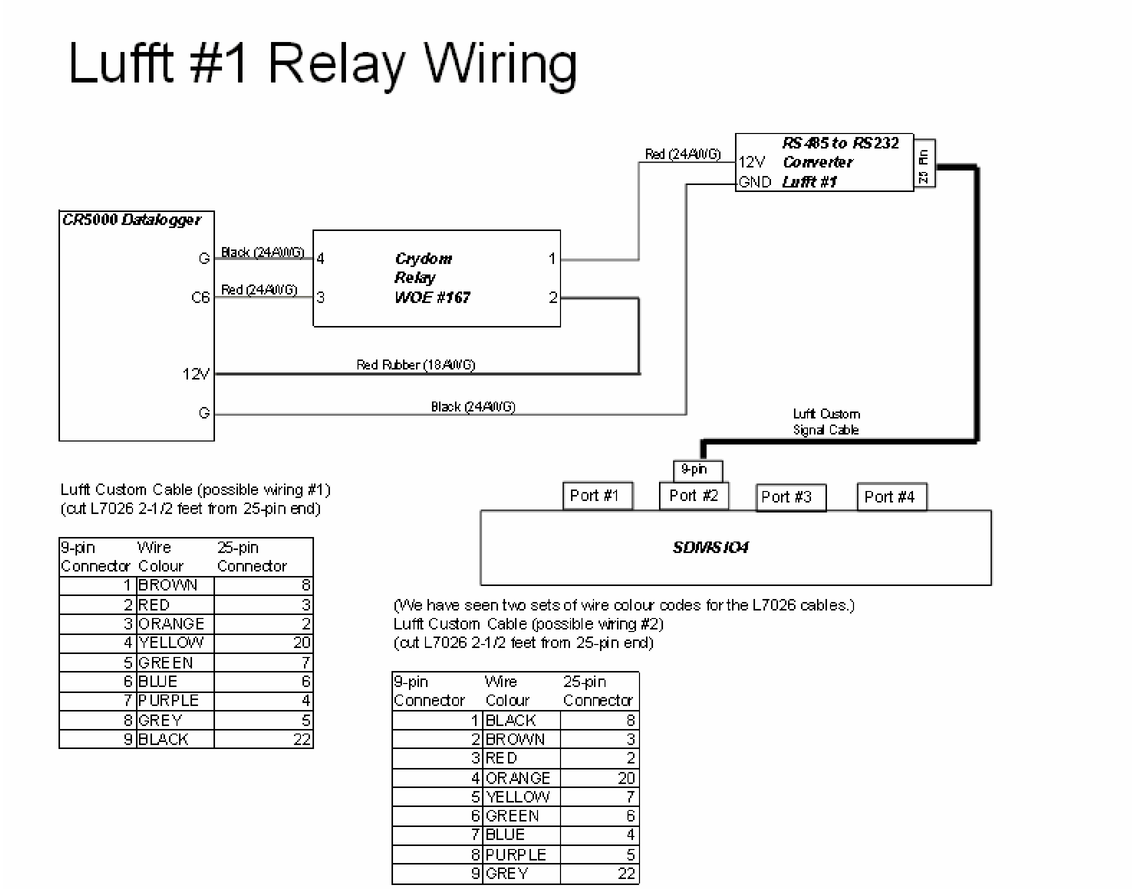
Figure 24. Intelligent road sensor

The IRS21 passive road sensor is built into the road, flush with the surface. The two-part housing design enables the combined sensor/electronics unit to be exchanged for maintenance or calibration purposes in a matter of minutes.

The following measurements are taken by the road sensor:

- Road surface temperature and 2 further below-ground temperatures (5 cm/30 cm)
- Water film level up to 4 mm
- Freezing temperature for NACL (others on enquiry)
- Road condition (dry/damp/wet/ice or snow/residual salt/freezing)

The sensors can be addressed and therefore networked. The measurement data is digitally transmitted via RS485-interface for further processing (logger, PLC etc.).



9-pin Connector	Wire Colour	25-pin Connector
1	BROWN	8
2	RED	3
3	ORANGE	2
4	YELLOW	20
5	GREEN	7
6	BLUE	6
7	PURPLE	4
8	GREY	5
9	BLACK	22

9-pin Connector	Wire Colour	25-pin Connector
1	BLACK	8
2	BROWN	3
3	RED	2
4	ORANGE	20
5	YELLOW	7
6	GREEN	6
7	BLUE	4
8	PURPLE	5
9	GREY	22

Figure 25. Lufft #1 Relay Wiring

The road sensor's program is as follows:

```
'_________________Lufft IRS-20 Intelligent Road Sensors
Declarations___________________________
'Notes:
'The Lufft Sensor requires an SDM-SIO4, a Fin6Cond, a B&B RS232/RS485 Converter and an
L7026 Cable
'Data is acquired by sending an "&A <CR LF>" to the sensor.
'Sio4Setup:    IRS20Fields:        Definitions:
'f1            Extt1Temp_C         Ext. 1 Temperature Deg C
'f2            Ext2Temp_C          Ext. 2 Temperature Deg C
'f3            SurfTemp_C          Sensor Surface Temperature Deg C
'f4            F1                  Frequency 1
'f5            F2                  Frequency 2
'f6            F3                  Frequency 3
'f7            F4                  Frequency 4
'f8            SaltConc%           Salt Concentration %
'f9            Freeze_C                    Freezing Temperature Deg C
'f10           Film_H                      Film Height mm
'f11           CorFactor           Correction Factor
```

```
'f12            RoadCondition           Road Condition
'f13            ErrorStat               Error Status
'Sio4 programming:
'strst 1 "&&A^M^J" 'Message to send to sensor to request data.
'fltst 2 "xfFX"  Receive filter string for Lufft data from sensor to SDM-SI04
'Lufft specific declarations
Const StringNumber1 = 1                         'String #1
Const FilterString2 = 9002                      'Filter String #2
Const strst1Bytes = 17                          '# bytes in string #1
Const fltst2Bytes = 13                  '# bytes in filter string #2
Dim strst1Byte(strst1Bytes)             'Declare strst1byte
Dim fltst2Byte(fltst2Bytes)                     'Declare fltst2byte
Const LufftFields = 13                          'Number of values to expect from Lufft
sensor
Public LufftCalc                                 'Location to control Sub LMultOff
Public LufftMult(LufftFields)           'Declare Lufft Multipliers
Public LufftOffset(LufftFields)                 'Declare Lufft Offsets
Dim i, j, k, l1, l2, m, n                'Declare variables used in loops
'Lufft #1
Public DataPollPort2                    'If greater than 0, data is available on port 2
Public PollingLoop1                     'Counter for how many times the CR5000
                                        'tried to retrieve data from SDM-SIO4 for Lufft #1
Public LufftGotData1                    'A counter which indicates the CR5000 received
                                        'data from Lufft #1
Dim   DataFlushPort2                    'Verify if the port 2 is flushed
Public RawLufftData1(LufftFields)               'Declare RawLufftData1 variables
Public LufftData1(LufftFields)                  'Declare scaled LufftData1 variables
Alias LufftData1(1) = Ext1Temp1                 'Assign alias Ext1Temp1 to LufftData1(1)
Units Ext1Temp1 = degC                          'Ext1Temp1 units are degrees Celsius
Alias LufftData1(2) = Ext2Temp1                 'Assign alias Ext2Temp1 to LufftData1(2)
Units Ext2Temp1 = degC                          'Ext2Temp1 units are degrees Celsius
Alias LufftData1(3) = SurfTemp1                 'Assign alias SurfTemp1 to LufftData1(3)
Units SurfTemp1 = degC                          'SurfTemp1 units are degrees Celsius
Alias LufftData1(4) = Freq1_1                   'Assign alias Freq1_1 to LufftData1(4)
Alias LufftData1(5) = Freq2_1                   'Assign alias Freq2_1 to LufftData1(5)
Alias LufftData1(6) = Freq3_1                   'Assign alias Freq3_1 to LufftData1(6)
Alias LufftData1(7) = Freq4_1                   'Assign alias Freq4_1 to LufftData1(7)
Alias LufftData1(8) = SaltConc1                 'Assign alias SaltConc1 to LufftData1(8)
Units SaltConc1 = percent                       'SaltConc1 units are percent
Alias LufftData1(9) = Freeze1                   'Assign aliasFreeze1 to LufftData1(9)
Units Freeze1 = degC                    'Freeze1 units are degrees Celsius
Alias LufftData1(10) = Film_H1                  'Assign alias Film_H1 to LufftData1(10)
Units Film_H1 = mm                      'Film_H1 units are millimeters
Alias LufftData1(11) =  CorFactor1              'Assign alias CorFactor1 to LufftData1(11)
Alias LufftData1(12) = RoadCondition1           'Assign alias RoadCondition1 to
LufftData1(12)
Alias LufftData1(13) = ErrorStat1               'Assign alias ErrorStat1 to LufftData1(13)
'Lufft #2
```

```
Public DataPollPort3                    'If greater than 0, data is available on port 3
Public PollingLoop2                     'Counter for how many times the CR5000
                                        'tried to retrieve data from SDM-SIO4 for Lufft #2
Public LufftGotData2                    'A counter which indicates the CR5000 received
                                        'data from Lufft #2
Dim DataFlushPort3                      'Verify if the port 3 is flushed
Public RawLufftData2(LufftFields)               'Declare RawLufftData2 variables
Public LufftData2(LufftFields)                  'Declare scaled LufftData2 variables
Alias LufftData2(1) = Ext1Temp2                 'Assign alias Ext1Temp2 to LufftData2(1)
Units Ext1Temp2 = degC                          'Ext1Temp2 units are degrees Celsius
Alias LufftData2(2) = Ext2Temp2                 'Assign alias Ext2Temp2 to LufftData2(2)
Units Ext2Temp2 = degC                  'Ext2Temp2 units are degrees Celsius
Alias LufftData2(3) = SurfTemp2                 'Assign alias SurfTemp1 to LufftData2(3)
Units SurfTemp2 = degC                          'SurfTemp2 units are degrees Celsius
Alias LufftData2(4) = Freq1_2                   'Assign alias Freq1_2 to LufftData2(4)
Alias LufftData2(5) = Freq2_2                   'Assign alias Freq2_2 to LufftData2(5)
Alias LufftData2(6) = Freq3_2                   'Assign alias Freq3_2 to LufftData2(6)
Alias LufftData2(7) = Freq4_2                   'Assign alias Freq4_2 to LufftData2(7)
Alias LufftData2(8) = SaltConc2                 'Assign alias SaltConc2 to LufftData2(8)
Units SaltConc2 = percent               'SaltConc2 units are percent
Alias LufftData2(9) = Freeze2                   'Assign aliasFreeze2 to LufftData2(9)
Units Freeze2 = degC                    'Freeze2 units are degrees Celsius
Alias LufftData2(10) = Film_H2                  'Assign alias Film_H2 to LufftData2(10)
Units Film_H2 = mm                      'Film_H2 units are millimeters
Alias LufftData2(11) =  CorFactor2              'Assign alias CorFactor2 to LufftData2(11)
Alias LufftData2(12) = RoadCondition2           'Assign alias RoadCondition2 to
LufftData2(12)
Alias LufftData2(13) = ErrorStat2               'Assign alias ErrorStat2 to LufftData2(13)

'________________________Lufft IRS-20 Intelligent Road
Sensors_____________________________
         'Programming for first Lufft sensor.

 'Program the multipliers and offsets for the Lufft fields.
 If LufftCalc=False Then
   LufftMult(1) = 0.1
   LufftMult(2) = 0.1
   LufftMult(3) = 0.1
   LufftMult(4) = 1.0
   LufftMult(5) = 1.0
   LufftMult(6) = 1.0
   LufftMult(7) = 1.0
   LufftMult(8) = 0.1
   LufftMult(9) = -0.1
   LufftMult(10) = 1.0
   LufftMult(11) = 1.0
   LufftMult(12) = 1.0
```

```
    LufftMult(13) = 1.0
    LufftOffset(1) = -50.0
    LufftOffset(2) = -50.0
    LufftOffset(3) = -50.0
    LufftOffset(4) = 0.0
    LufftOffset(5) = 0.0
    LufftOffset(6) = 0.0
    LufftOffset(7) = 0.0
    LufftOffset(8) = 0.0
    LufftOffset(9) = 0.0
    LufftOffset(10) = 0.0
    LufftOffset(11) = 0.0
    LufftOffset(12) = 0.0
    LufftOffset(13) = 0.0
    LufftCalc = True
  EndIf


  If S_Lufft_1 = True Then
  l1=0
  DO
 'Flush Receive buffers and restart receive filters on Port 2:
  Call(FAndF2)

 'Turn port 6 first relay of Lufft IRS-20 sensor "ON",  wait 1 second warm up:
  WriteIO (&B00100000,&B00100000)
  Delay(DelayProcessing,1,Sec)

 'Command SDM-SI04 Port2 to transmit sensor polling command "&A<CR LF>":

SIO4(NotUsed,OneRep,Sio4Address00,Port2,SendString,StringNumber1,UnusedParameter,NoValues,UnityMultiplier,NoOffset)
  Delay(DelayProcessing,800,mSec)

 'Clear the raw lufft data locations before retrieving the data.
  Move(RawLufftData1(),LufftFields,0,1)
 'Clear the lufft data locations before retrieving the data.
  Move(LufftData1(),LufftFields,0,1)
 'Retrieve data.
  For PollingLoop1 = 1 To 20

SIO4(DataPollPort2,OneRep,Sio4Address00,Port2,PollForData,UnusedParameter,UnusedParameter,1,UnityMultiplier,NoOffset)
      Delay(DelayProcessing,30,mSec)
      If DataPollPort2>51 Then

SIO4(RawLufftData1(),OneRep,Sio4Address00,Port2,SendDataToLgr,UnusedParameter,UnusedParameter,LufftFields,UnityMultiplier,NoOffset)
```

```
        Delay(DelayProcessing,200,mSec)
        ExitFor
        Else
Delay(DelayProcessing,1000,mSec)
        EndIf
        Next PollingLoop1
        l1 = l1 + 1
Loop Until (RawLufftData1(3) > 0) or (l1>2)

       'Turn port 6 first relay of Lufft IRS-20 sensor "OFF".
 WriteIO (&B00100000,&B00000000)

'Increment a counter if the CR5000 has received data from the first Lufft sensor.
 If RawLufftData1(3) > 0 Then
 LufftGotData1=LufftGotData1+1
 EndIf
'Scale Lufft data RawLufftData into desired format:

 For k = 1 To LufftFields
 If RawLufftData1(k) >= 999 Then
'If the raw data is greater than 999, store the data as 999 so we know there is an error.
 LufftData1(k) = 999
 ElseIf RawLufftData1(k) = 0 Then
 'If the raw data is 0, then store as zero, this means that the CR5000 did not receive data.
  LufftData1(k) = 0
  Else
 'Otherwise, convert the raw data to appropriate units.
  LufftData1(k) = (RawLufftData1(k) * LufftMult(k)) + LufftOffset(k)
  EndIf
  Next k
 'Convert film height to mm. (This is a more complicated equation that could not be calculated in
the
 'above loop.
  If LufftData1(10) > 0 Then
  If LufftData1(10) < 999 Then
  LufftData1(10) = (( 1560 -  ( 16.55 * LufftData1(10) ) ) +  ( 0.041 * LufftData1(10) *
LufftData1(10) ))  / 1000
  EndIf
  EndIf
 'Call Tables to output data for the first Lufft sensor.

  CallTable LufData1
  CallTable ChkLuf1
  EndIf

 'Programming for second Lufft sensor.
  If S_Lufft_2 = True Then
  l2=0
```

```
 DO
'Flush Receive buffers and restart receive filters on Port 3:
 Call(FAndF3)

'Turn port 7 second relay of Lufft IRS-20 sensor "ON",  wait 1 second warm up:
 WriteIO (&B01000000,&B01000000)
 Delay(DelayProcessing,1,Sec)

'Command SDM-SI04 Port3 to transmit sensor polling command "&A<CR LF>":

SIO4(NotUsed,OneRep,Sio4Address00,Port3,SendString,StringNumber1,UnusedParameter,No
Values,UnityMultiplier,NoOffset)
 Delay(DelayProcessing,800,mSec)

'Clear the raw lufft data locations before retrieving data.
 Move(RawLufftData2(),LufftFields,0,1)
'Clear the lufft data locations before retrieving data.
 Move(LufftData2(),LufftFields,0,1)

'Retrieve data.
 For PollingLoop2 = 1 To 20
SIO4(DataPollPort3,OneRep,Sio4Address00,Port3,PollForData,UnusedParameter,UnusedParam
eter,1,UnityMultiplier,NoOffset)
 Delay(DelayProcessing,30,mSec)
 If DataPollPort3>51 Then

SIO4(RawLufftData2(),OneRep,Sio4Address00,Port3,SendDataToLgr,UnusedParameter,Unuse
dParameter,LufftFields,UnityMultiplier,NoOffset)
 Delay(DelayProcessing,200,mSec)
 ExitFor
 Else
 Delay(DelayProcessing,1000,mSec)
 EndIf
 Next PollingLoop2
 l2=l2+1
 Loop Until (RawLufftData2(3) > 0) Or (l2>2)

'Turn port 7 second relay of Lufft IRS-20 sensor "OFF".
 WriteIO (&B01000000,&B00000000)

'Increment a counter if the CR5000 has received data from the second lufft sensor.
 If RawLufftData2(3) > 0 Then
 LufftGotData2=LufftGotData2+1
 EndIf
'Scale Lufft data RawLufftData into desired format:
 For m = 1 To LufftFields
 If RawLufftData2(m) >= 999 Then
'If the raw data is greater than 999, store the data as 999 so we know there is an error.
```

```
 LufftData2(m) = 999
 ElseIf RawLufftData2(m) = 0 Then
'If the raw data is 0, then store as zero, this means that the CR5000 did not receive data.
 LufftData2(m) = 0
 Else
'Otherwise, convert the raw data to appropriate units.
 LufftData2(m) = (RawLufftData2(m) * LufftMult(m)) + LufftOffset(m)
 EndIf
 Next m
'Convert film height to mm.  (This is a more complicated equation that could not be calculated
in the
'above loop.
 If LufftData2(10) > 0 Then
 If LufftData2(10) < 999 Then
 LufftData2(10) = (( 1560 -  ( 16.55 * LufftData2(10) ) ) +  ( 0.041 * LufftData2(10) *
LufftData2(10) ))  / 1000
 EndIf
 EndIf
'Call Tables to output data for the second Lufft sensor.
 CallTable LufData2
 CallTable ChkLuf2
 EndIf
 NextScan

 SlowSequence
'Output WIVIS 10 minute Data
 DataTable (WIVIS_10,True,-1)
 DataInterval (0,10,Min,10)
 Sample (1,WCode_Pres,FP2)
 Sample (1,PrecipRate,IEEE4)
 Sample (1,Status1,FP2)
 Sample (1,Status2,FP2)
 Sample (1,Status3,FP2)
 Sample (1,Status4,FP2)
 Sample (1,Visibility,IEEE4)
 Sample (1,PrecipTotal,IEEE4)
 Sample (1,WCode_15min, FP2)
 Sample (1,WCode_60min,FP2)
 EndTable
'Output Troubleshooting information from WIVIS.
 DataTable (ChkWIVIS,True,-1)
 DataInterval (0,10,Min,10)
 Sample (1,ScanCtrWivis,LONG)
 Sample (1,WGotData,LONG)
 Sample (1,DataPollPort4,FP2)
 Sample (1,i,FP2)
 Sample (1,WPollingLoop,FP2)
 Sample (1,Checksum1,FP2)
```

```
Sample (1,Checksum2,FP2)
Sample (1,BackLight,FP2)
EndTable
```

Table 6. The IRS21's specifications

Technical data	
Dimensions	Ø 120mm, height 50mm
Detectable road conditions	Dry/damp/wet/ice or snow/residual salt content/freezing wetness
Weight	Approx. 900g
Storage temperature	-30...70°C
Rated current	<200mA
Interface	RS485, baud rate: 2,400...38,400 bit/s (default: 19,200) cable length : 25m standard, up to 100m possible
Protection type	IP68
Power supply	9...14VDC, nominal 12V
Connector	CAGE CLAMP, WAGO (A <0,5mm)
Operating temperature	-30...70°C
Operating rel. humidity	0...100% RH
Road dampness	
Unit	Dry/damp/wet
Slippery road conditions	
Unit	No ice/snow, snow, freezing rain, ice
Temperature	
Principle	NTC
Measuring range	-30 ... 70 °C
Accuracy	±0.2°C (-10...10°C), or ±0.5°
Resolution	0,1 °C
Freezing point	
Measuring range	-20 ... 0 °C
Accuracy	±1°C (t>-10°C)
Resolution	0,1 °C
Water film height	
Principle	Radar
Measuring range	0 ... 4 mm
Accuracy	±(0.1mm + 20% of measurement)
Resolution	0,01 mm
Below-ground temperature (at 2 different depths)	
Principle	NTC
Measuring range	-30 ... 70 °C
Accuracy	±0.2°C (-10...10°C), or ±0.5°
Resolution	0,1 °C

2.3.14 Conductivity Sensor (MMRCS)

Figure 26. Conductivity sensor

Conductivity (water and salt contents) and asphalt temperature are measured on site in the asphalt, preferably in each lane. If necessary, deep soil temperature sensors can be used.

The Lufft road sensor IRS20 provides temperature of pavement, temperature subsurface at -5cm and -40cm (you can put external sensors at different levels), the state of the surface of the pavement, the depth of water or ice, and the temperature of freezing via conductivity. The cost for the intelligent sensor is about C$5000 to C$6000.

The Mierje Meteo sensor is a low cost conductivity sensor with limited accuracy because it does not account for depth of liquid. It does not provide a precise estimate of the freezing point. It was included because of the future possibility to develope an algorithm that would classify conductivity to determine the state of pavement (wet, dry, salt).

The MMRCS program consists of the following:

```
'______________________MMRCS Conductivity Sensor
Declarations_______________________________
Const MCSRNG = 0                'MMRCS Range (+-5000 mV)
Const MCSREP = 1                'MMRCS Repetitions
Const MCSEXCIT = 5000                   'MMRCS Excitation mVolts
Const MCSSETL = 200                     'MMRCS Settling Time (usecs)
Const MCSINT = 250              'MMRCS Integration Time (usecs)
Const MCSMULT = -0.001                  'MMRCS Multiplier
Const MCSOSET = 1               'MMRCS Offset
Public MCSBlk(MCSREP)                   'Declare MMRCS block
Alias MCSBlk(1) = MMCond                'Assign alias MMCond to MMRCS block
Const MCSCableLgth = 165        'MMRCS Cable Length (feet)
```

```
Const MCSCableRes = 16                        'MMRCS Cable Resistance (ohms / 1000
feet)
Const MCSCapRes = 0.005               'MMRCS Blocking Capacitor Resistance (kohms)
Dim MCSRTotal                                 'Declare location used for calculation
Public MMCondT                        'MMRCS Temperature Compensated Conductivity
Public MMCcode                        'Declare Conductivity code
Public MMCcodeT                       'Declare temperature conpensated conductivity code


  '________________________MMRCS Conductivity
Sensor_____________________________________
   'Measure the MMRCS Conductivity sensor.  Measure bridge voltage and
   'calculate sensor resistance.
   BrFull (MCSBlk(),MCSREP,MCSRNG,4,Vx2,1,MCSEXCIT,True ,True
,MCSSETL,MCSINT,MCSMULT,MCSOSET)
   If MMCond = 1 Then
   MMCond = 1.7E38
   else
  'valeur de la resistance variable R2 en Kohms
   MMCond = (1 * (MMCond / (1 - MMCond)))
   EndIf

  ' Calculate resistance errors (MCSRTotal) caused by the blocking capacitors
  ' (MCSCapRes) and the cable (MCSCableRes) length (MCSCableLgth).
  ' MCSRTotal = (MCSCapRes + (MCSCableLgth * MCSCableRes / 500000 ))
  ' Subtract the resistance errors.
  ' MMCond = (MMCond - MCSRTotal)

  'Convert the resistance value into a conductivity value in microsiemens.
   MMCond = (1000 / MMCond)

   If MMCond < 0 Then
   MMCcode = 2
   ElseIf MMCond < 10 Then
   MMCcode = 3
   ElseIf MMCond < 100 Then
   MMCcode = 4
   ElseIf MMCond < 1000 Then
   MMCcode = 5
   Else
   MMCcode = 6
   EndIf
  'Calculate the temperature compensated conductivity in microsiemens using the first
  'temperature from the MMRTS.
   MMCondT = (MMCond * (1 / (1 + (0.0191 * (MMTemp - 25)))))
   If MMCondT < 0 Then
   MMCcodeT = 2
   ElseIf MMCondT < 10 Then
```

```
MMCcodeT = 3
ElseIf MMCondT < 100 Then
MMCcodeT = 4
ElseIf MMCondT < 1000 Then
MMCcodeT = 5
Else
MMCcodeT = 6
EndIf
If S_MMRCS = True Then
CallTable MMRCS
EndIf
```

Table 7. MMRCS specifications

RCS – 290 Road conductivity sensor	
Operating range	0...999µS
Accuracy	1µS@0...100µS
	10µS@100...999µS
Dimensions	Ø 88,9 mm / height 30 mm
Material	Stainless steel 316
Cable length	15 m

2.3.15 Ground Temperature (TP101)

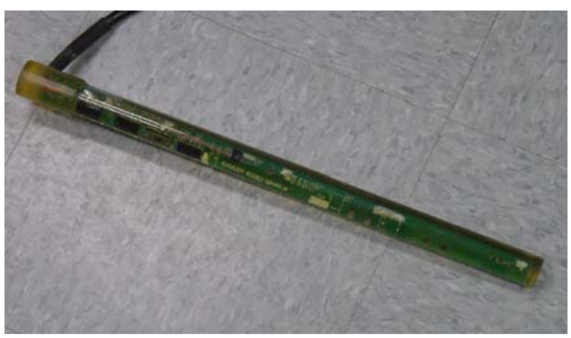


Figure 27. TP101

The ground temperature (TP101) is for reading the freeze-thaw cycle in roadway structures. Manufactured exclusively by Measurement Research Corporation, the MRC Temperature Probe provides the data to study the temperature of the ground.

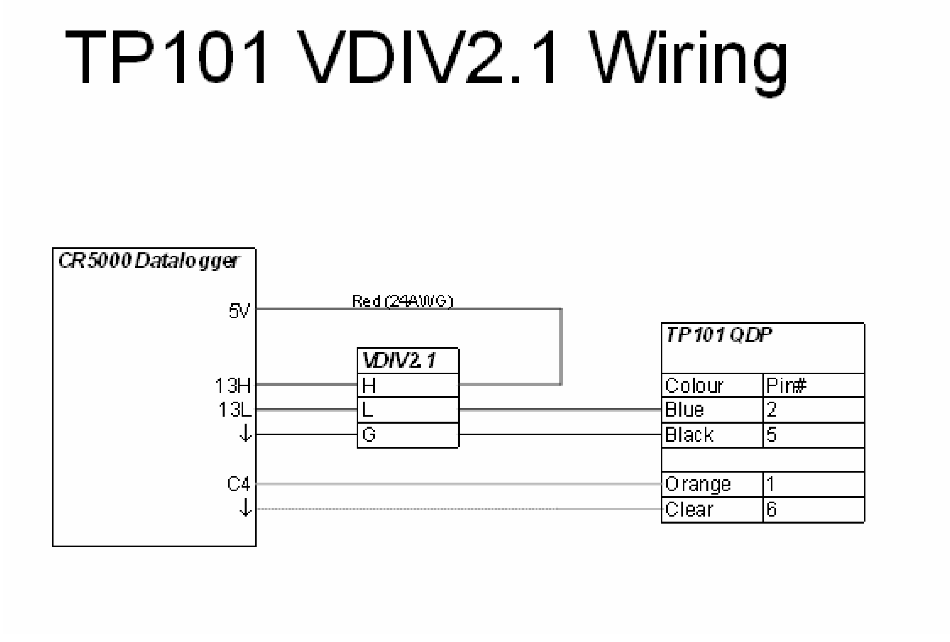


Figure 28. TP101 VDIV2.1 Wiring

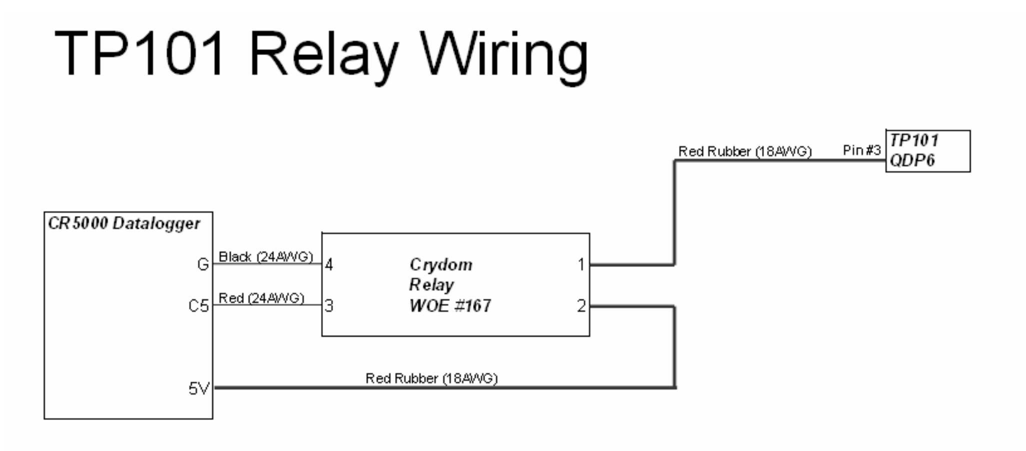


Figure 29. TP101 Relay Wiring

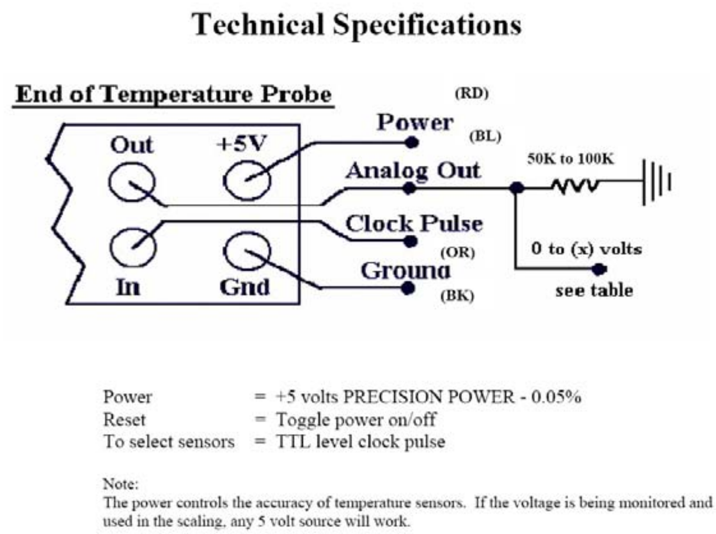


Figure 30. TP101 technical specifications

The TP101 program is as follows:

```
'__________________TP101 Road Temperature Sensor
Declarations_______________________________
Const PORT4HI = 1                   'Control Port #4 Hi
Const PORT4LO = 0                   'Control Port #4 Lo
Const PORT5HI = 1                   'Control Port #5 Hi
Const PORT5LO = 0                   'Control Port #5 Lo
Const TPD_OP = 0                    'Delay Analog Measurements only
Const TPDLAY1 = 10000                        'Actual Delay for length of the pulse
Const TPDLAY2 = 20                         'Actual Delay after the port is pulsed
Const TPD_UNITS = 1                        'Delay Units (msec)
Const TPRNG = 0                            'TP101 Range (5000mV)
Const TPREP = 2                            'TP101 Repetitions
Const TPSETL = 20500                       'TP101 Settling Time (usecs)
Const TPINT = _60Hz                        'TP101 Integration Time (usecs)
Const TPMULT = 1                    'TP101 Multiplier
Const TPOSET = 0                    'TP101 Offset
Dim TPBlk(TPREP)                    'Declare TP101 block
Dim Vx                                     'Declare location used for calculations
Dim Vin                                    'Declare location used for calculations
Dim TPRt                            'Declare location used for calculations
Dim TPTempC                                'Declare location used for calculations
Public TPTemp1                             'Declare first thermistor on TP101
Public TPTemp2                             'Declare second thermistor on TP101
Public TPTemp3                             'Declare third thermistor on TP101
Public TPTemp4                             'Declare fourth thermistor on TP101
```

```
Public TPTemp5                              'Declare fifth thermistor on TP101
Const TPA = 2.4886E-03                      'TP101 constant A for Temperature
Equation
Const TPB = 2.5079E-04                      'TP101 constant B for Temperature Equation
Const TPC = 3.1754E-07                      'TP101 constant C for Temperature Equation


'______________________TP101 Road Temperature
Sensor______________________________
   'Turn on the probe via a Crydom Relay.
   Portset (5,PORT5HI)
   '#1
   'Pulse the Clock line to measure the first thermistor
   Portset (4,PORT4HI)
   'Delay (TPD_OP,TPDLAY1,TPD_UNITS)
   'We simulate delay of 10 msec with the settle time
   VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPDLAY1,TPINT,TPMULT,TPOSET)
   Portset (4,PORT4LO)
   'Delay (TPD_OP,TPDLAY2,TPD_UNITS)
   'We simulate delay of 20.5 msec with the settle time
   'Measure the Excitation Voltage and the voltage across the Thermistor
   VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPSETL,TPINT,TPMULT,TPOSET)
   Vx = TPBlk(1)
   Vin = TPBlk(2)
   'Convert the voltage to degrees Celsius
   TPRt = LOG (((Vx-Vin) / Vin) * 20)
   TPTempC = (TPA + (TPB * TPRt) + (TPC * (TPRt^3)))
   TPTempC = ((1 / TPTempC) - 273.15)
   TPTemp1 = TPTempC
   '#2
   'Pulse the Clock line to measure the second thermistor
   Portset (4,PORT4HI)
   'Delay (TPD_OP,TPDLAY1,TPD_UNITS)
   'We simulate delay of 10 msec with the settle time
   VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPDLAY1,TPINT,TPMULT,TPOSET)
   Portset (4,PORT4LO)
   'Delay (TPD_OP,TPDLAY2,TPD_UNITS)
   'We simulate delay of 20.5 msec with the settle time
   'Measure the Excitation Voltage and the voltage across the Thermistor
   VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPSETL,TPINT,TPMULT,TPOSET)
   Vx = TPBlk(1)
   Vin = TPBlk(2)
   'Convert the voltage to degrees Celsius
   TPRt = LOG (((Vx-Vin) / Vin) * 20)
   TPTempC = (TPA + (TPB * TPRt) + (TPC * (TPRt^3)))
   TPTempC = ((1 / TPTempC) - 273.15)
   TPTemp2 = TPTempC
   '#3
```

```
'Pulse the Clock line to measure the third thermistor
Portset (4,PORT4HI)
'Delay (TPD_OP,TPDLAY1,TPD_UNITS)
'We simulate delay of 10 msec with the settle time
VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPDLAY1,TPINT,TPMULT,TPOSET)
Portset (4,PORT4LO)
'Delay (TPD_OP,TPDLAY2,TPD_UNITS)
'We simulate delay of 20.5 msec with the settle time
'Measure the Excitation Voltage and the voltage across the Thermistor
VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPSETL,TPINT,TPMULT,TPOSET)
Vx = TPBlk(1)
Vin = TPBlk(2)
'Convert the voltage to degrees Celsius
TPRt = LOG (((Vx-Vin) / Vin) * 20)
TPTempC = (TPA + (TPB * TPRt) + (TPC * (TPRt^3)))
TPTempC = ((1 / TPTempC) - 273.15)
TPTemp3 = TPTempC
'#4
'Pulse the Clock line to measure the fourth thermistor
Portset (4,PORT4HI)
'Delay (TPD_OP,TPDLAY1,TPD_UNITS)
'We simulate delay of 10 msec with the settle time
VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPDLAY1,TPINT,TPMULT,TPOSET)
Portset (4,PORT4LO)
'Delay (TPD_OP,TPDLAY2,TPD_UNITS)
'We simulate delay of 20.5 msec with the settle time
'Measure the Excitation Voltage and the voltage across the Thermistor
VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPSETL,TPINT,TPMULT,TPOSET)
Vx = TPBlk(1)
Vin = TPBlk(2)
'Convert the voltage to degrees Celsius
TPRt = LOG (((Vx-Vin) / Vin) * 20)
TPTempC = (TPA + (TPB * TPRt) + (TPC * (TPRt^3)))
TPTempC = ((1 / TPTempC) - 273.15)
TPTemp4 = TPTempC
'#5
'Pulse the Clock line to measure the fifth thermistor
Portset (4,PORT4HI)
'Delay (TPD_OP,TPDLAY1,TPD_UNITS)
'We simulate delay of 10 msec with the settle time
VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPDLAY1,TPINT,TPMULT,TPOSET)
Portset (4,PORT4LO)
'Delay (TPD_OP,TPDLAY2,TPD_UNITS)
'We simulate delay of 20.5 msec with the settle time
'Measure the Excitation Voltage and the voltage across the Thermistor
VoltSe (TPBlk(),TPREP,TPRNG,25,1,TPSETL,TPINT,TPMULT,TPOSET)
Vx = TPBlk(1)
Vin = TPBlk(2)
```

```
 'Convert the voltage to degrees Celsius
 TPRt = LOG (((Vx-Vin) / Vin) * 20)
 TPTempC = (TPA + (TPB * TPRt) + (TPC * (TPRt^3)))
 TPTempC = ((1 / TPTempC) - 273.15)
 TPTemp5 = TPTempC
 'Turn off the probe via a Crydom Relay
 Portset (5,PORT5LO)
 If  S_TP101 = True Then
 CallTable TP101
 EndIf
 NextScan

 SlowSequence
'Lufft 10 minute Data
'Output data from Lufft #1.
 DataTable(LufData1,True,-1)
 DataInterval (0,10,Min,10)
 Sample (1,Ext1Temp1,IEEE4)
 Sample (1,Ext2Temp1,IEEE4)
 Sample (1,SurfTemp1,IEEE4)
 Sample (1,SaltConc1,IEEE4)
 Sample (1,Freeze1,IEEE4)
 Sample (1,Film_H1,IEEE4)
 Sample (1,RoadCondition1,IEEE4)
 Sample (1,ErrorStat1,IEEE4)
 Sample (1,Freq1_1,IEEE4)
 Sample (1,Freq2_1,IEEE4)
 Sample (1,Freq3_1,IEEE4)
 Sample (1,Freq4_1,IEEE4)
 EndTable

'Output troubleshooting information from Lufft #1.
 DataTable (ChkLuf1,True,-1)
 DataInterval (0,10,Min,10)
 Sample(1,ScanCounter,LONG)
 Sample(1,LufftGotData1,LONG)
 Sample (1,DataPollPort2,FP2)
 Sample (1,l1,FP2)
 Sample (1,PollingLoop1,FP2)
 Sample (1,CorFactor1,FP2)
 EndTable

'Output data from Lufft #2.
 DataTable(LufData2,True,-1)
 DataInterval (0,10,Min,10)
 Sample (1,Ext1Temp2,IEEE4)
 Sample (1,Ext2Temp2,IEEE4)
 Sample (1,SurfTemp2,IEEE4)
```

```
 Sample (1,SaltConc2,IEEE4)
 Sample (1,Freeze2,IEEE4)
 Sample (1,Film_H2,IEEE4)
 Sample (1,RoadCondition2,IEEE4)
 Sample (1,ErrorStat2,IEEE4)
 Sample (1,Freq1_2,IEEE4)
 Sample (1,Freq2_2,IEEE4)
 Sample (1,Freq3_2,IEEE4)
 Sample (1,Freq4_2,IEEE4)
 EndTable

'Output troubleshooting information from Lufft #2.
 DataTable (ChkLuf2,True,-1)
 DataInterval (0,10,Min,10)
 Sample(1,ScanCounter,LONG)
 Sample(1,LufftGotData2,LONG)
 Sample (1,DataPollPort3,FP2)
 Sample (1,l2,FP2)
 Sample (1,PollingLoop2,FP2)
 Sample (1,CorFactor2,FP2)
 EndTable
```

TP101's specifications are as follows:

TP101 Temperature Probe

Std. Dimensions: ....1.1" diameter,
Accuracy: TP101 ....±0.1° C @ -30 to +75° C - YSI 44032
Operating Temp ....-30 to +75 Deg. C
Operating Humidity ....0 to 100%
Power Requirements ....15uA Stand-By - 380uA w/Clock Pulse High
Interconnection ....Open lead 4 conductor cable

Note 1: The electronic switches necessary for probe operation are guaranteed to operate from –40°F to +167°F but frequently function at lower temperatures. The SSI ESS software accepts and reports temperatures down to –60°F. Temperatures outside the range of –60 to +167°F are not allowed and cause blank display values.

2.3.16 Temperature Sensor (MMRTS)

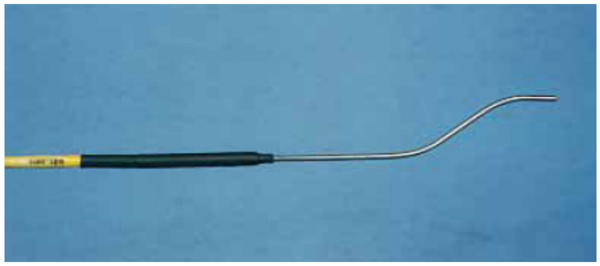


Figure 31. MMRTS temperature sensor

Slippery roads can occur under different conditions. These conditions determine if freezing rain, black ice, frost or snow make the roads hazardous. The road temperature sensor gives the temperature of the surface of the roadway.

The MMRTS' program is as follows:

```
'___________________MMRTS Temperature Sensor Declarations______________________________
Const MTSRNG = 5                          'MMRTS Range (Auto)
Const MTSREP = 1                          'MMRTS Repetitions
Const MTSEXCIT = 5000                     'MMRTS Excitation mVolts
Const MTSSETL = 200                       'MMRTS Settling Time (usecs)
Const MTSINT = 250                        'MMRTS Integration Time (usecs)
Const MTSMULT = -0.001                    'MMRTS Multiplier
Const MTSOSET = 1                         'MMRTS Offset
Dim MTSBlk(MTSREP)                        'Declare MMRTS block
Dim MM_T1                                 'Declare location used for calculations
Const MTSCableLgth = 165                  'MMRTS Cable Length (feet)
Const MTSCableRes = 16                    'MMRTS Cable Resistance (ohms / 1000
feet)
Dim MTSRTotal                             'Declare location used for calculations
Const MTSA = 1.4674E-03                   'MMRTS Constant A for Steinhart-Hart
Equation
Const MTSB = 2.3837E-04                   'MMRTS Constant B for Steinhart-Hart
Equation
Const MTSC = 1.0114E-07                   'MMRTS Constant C for Steinhart-Hart
Equation
Dim LnMM_T1                               'Declare location used for calculations
Dim Temp1K                                'Declare location used for calculations
Public MMTemp                             'Declare MMRTS Temperature
Units MMTemp = degC                       'MMRTS Temperature units are degrees
Celsius
```

'____________________MMRTS Temperature Sensor___________________________________

```
   'Measure the temperature sensor of the MMRTS.  Each MMRTS has two thermistors
   'in case one fails.  It is recommended that only one sensor be connected and measured.
   'If the sensor fails then the second thermistor should be connected.
   BrFull (MTSBlk(),MTSREP,MTSRNG,5,Vx3,1,MTSEXCIT,True ,True
,MTSSETL,MTSINT,MTSMULT,MTSOSET)
   MM_T1 = MTSBlk(1)
   MM_T1 = (1 * (MM_T1 / (1 - MM_T1)))
   'Calculate resistance errors (MTSRTotal) caused by the cable (MTSCableRes)
   'length (MTSCableLgth).
   MTSRTotal = (MTSCableLgth * MTSCableRes / 1000000 * 2)
   'Subtract the resistance errors.
   MM_T1 = (MM_T1 - MTSRTotal)
   'Convert the resistance values from kohms to ohms.
   MM_T1 = (MM_T1 * 1000)
   'Calculate the temperature using the Steinhart-Hart equation.
   'Temp(deg K) = 1/(A + B(LnR) +C(LnR)^3)
   'Where R is the resistance value that is currently in MM_Temp1
   LnMM_T1 = (LOG (MM_T1))
   'Convert the resistance of the thermistor to a temperature in degrees K.
   Temp1K = (1 / (MTSA + (MTSB * LnMM_T1) + (MTSC * (LnMM_T1^3))))
   'Convert the Temperature from degrees K to degrees C.
   MMTemp = (Temp1K - 273.15)
   If S_MMRTS = True Then
     CallTable MMRTS
   EndIf
```

Table 8. MMRTS specifications

RTS – 806T Road temperature sensor	
Sensor type	NTC 2252 Ω @ 25°C
Operating range	-20...+80°C
Accuracy	±0.1°C @ 0°C
Response time	< 0,5s
Dimensions	Ø 3,5 mm / length 225 mm
Material	Stainless steel 316
Cable length	15 m

3. FINDINGS

3.1 Costs

Table 9. Costs of building the station

PRINCIPAL CABINET AND ACCESSORIES				
Qty	Model	Description	$/Unit	Total
1	PCP-001	Principal cabinet, bottom, shelves and support	2,952.00	2,952.00
1	Isotel8Ultra	Block of distribution with protection of surcharge	104.00	104.00
1	Heater	Heating system in the cabinet (150 W)	~ 250.00	250.00
1	SP1110	Battery charger 12 vcc, 8 amps	285.00	285.00
1	PS-12260F	Battery 12 vcc with whole of connection	170.00	170.00
			Total:	**3,761.00**
TOWER AND ACCESSORIES				
Qty	Model	Description	$/Unit	Total
1	GLMF-1330-0	Collapsible tower of 10 m with base for anchoring	2,895.00	2,895.00
2	ARM-A1	Left arm for assembly of anemometer	78.00	156.00
1	ARM-W1	Left arm for temperature and precipitation	82.00	82.00
1	SUP-FW	Support of conduit and space for the cables	192.00	192.00
1	SUP-BWH	Higher support for assembly of the WIVIS cabinet	56.00	56.00
1	SUP-BWB	Lower support for assembly of the WIVIS cabinet	42.00	42.00
1	EXT-W2	Left arm and extension for the hail sensor	48.00	48.00
1	Lightning conductor	Complete lightning conductor system for the tower	400.00	400.00
			Total:	**3,871.00**
SYSTEM Of ACQUISITION OF The DATA AND SENSORS				
Qty	Model	Description	$/Unit	Total
1	Datalogger	Complete system of data acquisition	17,700.00	17,700.00
1	05103-10-L	Anemometer (speed and direction of the winds)	1,095.00	1,095.00
1	HMP45C-L	Temperature and humidity sensor	890.00	890.00
1	41002-2	Guard for Hmp45c-l (Solar radiation shield)	210.00	210.00
1	MMRTS-L	Temperature sensor of the roadway	960.00	960.00
1	MMRCS-L	Detector of the rate of salinity	1,325.00	1,325.00
1	TP101	Temperature sensor of the ground of 3 m	2,300.00	2,300.00
1	Wivis	Sensor of visibility and precipitation	16,100.00	16,100.00

2	IRS21	Sensor of Lufft roadway	7,070.00	14,140.00
1	CNR1	Radiometer	8,880.00	8,880.00
			Total:	**63,600.00**
OTHER ACCESSORIES				
Qty	Model	Description	$/Unit	Total
1	N.D	Foundation for the tower and the principal cabinet	4,200.00	4,200,00
1	N.D	Foundation for the radiometer	~225.00	225.00
1	N.D	Mast for radiometer 5.5 m	~888.03	888.03
1	N.D	Bracket for radiometer	1,010.25	1,010.25
1	N.D	Safety box	245.69	245.69
1	N.D	Support for the radiometer	276.00	276.00
1	N.D	System of lightning conductor for the radiometer	582.00	582.00
1	N.D	Mast for electric and telephone entry	~888.03	888.03
1	N.D	Foundation for the electrical entry	~225.00	225.00
1	N.D	Electrical entry 30 amps	300.00	300.00
1	N.D	Enclose with standard barbed wire	~2,700.00	2,700.00
5	N.D	Lock	19.79	98.85
1	N.D	Pulling box	575.00	575.00
1	N.D	Grout	325.00	325.00
1	N.D	Parts of conduit PVC for the interior of the foundation	359.69	359.69
1	N.D	Parts of conduit PVC (foundation until pulling box)	1,090.50	1,090.50
1	N.D	PVC for outside of the foundation with system WIVIS	122.66	122.66
			Total:	**14,111.70**
OTHERS				
A	Time rate of the vehicles of the ministry (170 hours)			4,500.00
B	Materials iron and cement			3,500.00
C	Hiring of machinery (30 hours)			3,000.00
D	Personnel (2 technicians and 3 workmen)			12,220.00
			Total:	**23,220.00**
Cost for the construction of a road weather station				
			TOTAL:	108,563.70

3.2 Problem Solving

Figure 32. The road weather station

We had some problems 1 year after construction. Two intelligent road sensors (Lufft) and two RS232/485 converters were damaged by lightning. The powerbar could not protect the system enough. The problem was that earthing of hydro-Quebec was at 8 ohms. We solved the problem by lowering the resistance of earthing to 6 ohms because we were at 9 ohms during breaking. Since this modification was made, all has been going very well.

4. CONCLUSIONS AND RECOMMENDATIONS

The open, programmable off-the-shelf system has three advantages:

1. The possibility to choose the sensor for each category that has the specification you need for your application and will eventually reduce cost.
2. The possibility to add new sensors that can be more powerful or new sensors of other types into the existing open system to measure other variables. For example, we can add into the system a traffic sensor to measure the traffic, we can add a sensor to detect ice on the structure of a bridge, we can add a sensor or change a road sensor to a more powerful sensor, etc. We can program the interface for any new sensors that have open specification.
3. An open system can be integrated with an Oracle database to store data. The data model of the relational database is very flexible with the acquisition system and can store data from any sensors without reprogramming. We add into the database a table of an RWIS station and for each table we define the list of columns corresponding to the variables measured into the Campbell Scientific CR5000 data logger. We presently have a plan to test a low cost data logger with fewer channels, the CR1000, and to build lower cost RWIS stations to only measure the temperature of air, the temperature of pavement, and the temperature of subsurface with freezing sensor of 3 or 4 m.