APPLICATION NOTE

App. Note Code: 3MI-F

Interfacing the Irrometer 950R1-O/950T Radio Network with Campbell Scientific's CR1000 Datalogger

CAMPBELL® SCIENTIFIC
WHEN MEASUREMENTS MATTER

*Interfacing the Irrometer 950R1-O/950T Radio Network with a CR1000**

This application note assumes that the user knows how to configure their Irrometer 950R1 network. The purpose of this application note is to explain how to communicate with the Irrometer 950R1-O as well as show programming tips and tricks. A full CR1000 program can be found at the end of this write up that measures a single 950R1 that's recording information from 16 950Ts in the field. Each 950T has four WaterMark blocks attached to it.

1. Communication Interface Equipment Needed

Either the SC110 RS-232 Cable or item #18663 Null Modem Male-to-Male Cable is required for datalogger-to-datalogger communication.

The SC110 cable terminates in male 9-pin connector and is used if connecting to a CR1000 COM port (COM1 to COM4); see Table 1. The #18663 Null modem cable is used if connecting to the CR1000's RS-232 port.

Table 1: SC110 Cable Wiring

Wire Color	Function	CR1000 Channel			
		COM1	COM2	COM3	COM4
Brown	Tx	C1	C3	C5	C7
White	Rx	C2	C4	C6	C8
Yellow	Ground	G	G	G	G

All other wires from the SC110 must be carefully folded back and wire tied to the main SC110 cable or in some way protected from shorting to each other or to metal.

The 950R1 must have an accessible 9-pin RS-232 serial port. Model 950R1-O includes a terminal strip that allows external power and data cables to be connected. Irrometer can supply a cable that will connect to this terminal strip and provide a 9-pin serial connection for connecting to the SC110 or null modem cable.

2. 950R1/950T Communication Basics

The 950T only transmits data if there has been a 2% change on any sensor at an interval not to exceed once every 37 minutes. Soil moisture changes fairly slowly so there is a good chance that you won't get a transmission every 37 minutes if the 950T has been configured with four WaterMark blocks. To increase the number of transmissions you might consider configuring the 950T with one soil temperature sensor and three WaterMark blocks. Soil temperature will change at a higher frequency than soil moisture and cause more communication events to occur. The program in this application note uses four WaterMark blocks but a similar method of communication can be used for a mixture of one soil temperature sensor and three WaterMark blocks.

*Thanks to Jeremy Sullivan with Irrometer for loaning equipment to test the system, checking this document for accuracy, and answering questions.

815 W. 1800 N., Logan, UT 84321-1784 (435) 227-9000

CR1000 RS-232 serial configuration settings to communicate with the 950R1 are (9600, 8, N, 1):

- 9600 baud
- 8 bits
- No parity
- One stop bit

2.1 Types of Data Requests

There are two basic types of 950R1 serial data requests. One type of request causes the 950R1 to send all of its stored data for all of the 950Ts and the other type of request causes the 950R1 to send the most recently logged information for each of the 950Ts.

This write up explains how to work with the latter type of request that brings in the most recently logged information. The advantage to this method is that a datalogger can poll the 950R1 at rates that exceed the normal 950T response time of 37 minutes so there isn't a need to pull out all the data and processing the data is simpler.

2.2 950R1 General Communication Process Overview

1. Wake up the 950R1
2. Send a request to get the most recently logged arrays of data
3. Bring the data in and parse it into individual arrays
4. Match the data up with the correct 950T and do something with it. For this write up the data are saved to a table. The data could also be used to control irrigation or implement other control/measurement functions.

NOTE

Data arrives from the 950R1 in sequential form based on time. The data is composed of the most recently received information from each 950T in the field. Structure of the 950R1 is from newest to oldest. For example if data from 950T #13 has a time stamp of 4/05/12 @ 13:53 and 950T #2 has a time stamp of 4/05/12 @ 8:23 then 950T #13 data will precede the data from 950T #2 in the data sent by the 950R1.

2.3 Wake Up the 950R1

The 950R1 is not expecting communication through the RS-232 serial port. To conserve power, it shuts the port down and ignores it unless it is woken up and put into command mode. To do this, the serial port on the 950R1 must be sent a serial break signal followed by an ASCII escape character [27_d or $1B_h$].

The 950R1 acknowledges that it is awake and ready to receive commands by transmitting a carriage return [13_d or $0D_h$], a line feed [10_d or $0A_h$], and a greater-than symbol (>) [62_d or $3E_h$].

The monitor only remains awake for 15 seconds after replying back. Anytime the 950R1 is not outputting data or receiving commands, it restarts the 15 second timer. The 950R1 drops communication if there is no activity after 15 seconds.

The CR1000 program tries three times to get a response back from the 950R1. If it doesn't get a response back it loads all the arrays with "NAN" to indicate an unsuccessful communication attempt.

2.4 Send a Request to Get the Most Recently Logged Arrays of Data

To retrieve the most recently stored arrays a capital 'L' [76_d or $4C_h$] and a carriage return [13_d or $0D_h$] is sent to the 950R1.

The 950R1 responds back with the following minus the commas:

Capital 'L' [76_d or $4C_h$], 20_h, carriage return [13_d or $0D_h$], line feed [10_d or $0A_h$], *data stream for all 16 wireless 950Ts, EOT (End Of Transmission) [4_d or 4_h], carriage return [13_d or $0D_h$], line feed [10_d or $0A_h$]. All the data for all the various 950Ts will come streaming in and must be captured in one big string.

* data stream is structured as follows for each 950T (the commas ARE included in the data set):

<*STX> <Line Number> , Month, Day, Year, Hour, Minute, Address, Mode, Sensor1 (Hex), Sensor2 (Hex), Sensor3(Hex), Sensor4 (Hex), Battery Voltage (Hex + some math), <**ETX> <LRC Checksum>

Where:

1. *STX: Start of Text character represented by 2_d or 02_h.
2. Line Number: line number for this data as stored in the 950R1.
3. Month: two digit month (1 – 12)
4. Day: two digit day (1 – 31)
5. Year: two digit year (0 – 99)
6. Hour: two digit hour (1 – 24)
7. Minute: two digit minute (0 – 59)

NOTE

Values 8 through 14 require conversion from hexadecimal to decimal.

8. Address: two digits (BR) that consist of two parts.
   a. B = 950R1 base address (0-8). For this program the base address is ignored.
   b. R = 950T address in hexadecimal (0-F)
9. Mode: sensor configuration of 950T radio. See 950T manual. The attached program ignores this value.
10. Sensor 1: hexadecimal value.
11. Sensor 2: hexadecimal value.

12. Sensor 3: hexadecimal value.
13. Sensor 4: hexadecimal value.
14. Battery Voltage: hexadecimal value that must be converted to decimal and then multiplied by 0.038 to get the correct decimal voltage.
15. **ETX: End Of Text character represented by 3_d or 03_h.
16. LRC Checksum: I ignore this value. I'm assuming the datalogger is situated right next to the 950R1 so noise shouldn't be an issue.

2.5 Bring the Data In and Parse It into Individual Arrays

This is a two-step process performed by a subroutine within the CR1000 program. The reason for using a subroutine is to make it easier to add additional 950R1 dataloggers to one CR1000.

1. Break the one big string of all the 950T data into sixteen separate string arrays. Each individual 950T string is composed of one large string of characters.
    a. Strings are broken up using the STX character. Each string starts with the characters following the STX character and run up to, but not including, the next STX character.
    b. Count the number of lines that are split out. Use this counter to determine how many strings need to be processed in the next step. If 16 950Ts are being used in the field then this number will stick at 16 and never change. Initially there might not be 16 active 950Ts in the field so only process what has been received.
2. Break the 16 950T strings into 16 950T arrays composed of 12 elements.
    a. Strings are broken up using the commas in the string. Elements following the commas are kept. This means that the line number is automatically shaved off at the beginning. The values that are kept are: (1) Month, (2) Day, (3) Year, (4) Hour, (5) Minute, (6) Address, (7) Mode, (8) Sensor1, (9) Sensor2, (10) Sensor3, (11) Sensor4, and (12) Battery Voltage.
    b. An additional step is done to compress the data further by combining the two separate hour and minute values into one value – Hour/Minute (HHMM).

    HHMM = Hour *100 + Minute

2.6 Match the Data up with the Correct 950T and Do Something With It

The data coming in from the 950R1 is structured based on time stamps from the newest received record to the oldest. The sorted data arrays from the previous step must now be sorted again and matched up with the correct 950T public table variables based on the 950T address.

The attached program only keeps the following information from each of the 950Ts.

1. Hour/Minute

2. Address
3. Sensor1
4. Sensor2
5. Sensor3
6. Sensor4
7. Battery Voltage

The program could be modified to keep more information if required. The 950R1 is read once an hour and samples of these seven values are stored once an hour.

The 950R1 can be read manually at a different interval by setting Boolean variable "Read950R1" to true.

```
'CR1000 Series Datalogger
'Program: Irrometer_950R.CR1
'Programmer: Bart Nef - CSI
'Date: 18.April.2012
'Ver: A
'Notes: Program reads the information from one Irrometer 950R unit.
'Each 950R is in turn receiving data from 16 remote 950T units.
'Each 950T module has a soil temperature and three WaterMark blocks attached to it.
'
'I'm using the 'L' command to read the most recent data from the 950R. This is the
'least destructive way to get at the data and does NOT clear out any data
'stored in the 950R.
'
'If I can't get the proper response from the 950R presently being polled I 'will load all
associated 950T variables with -6999 to indicate communication failure.
'
'The 950Ts are addressed as 0 - 15. I'm taking that number and giving it
'a zone number starting at 1.


SequentialMode
'NOTE: I can't start a name for a constant or a variable with a number!
'So...950R = R950 & 950T = T950
Const ESC=CHR(27)  'Escape character. Used to get the 950R's attention.
Const CR=CHR(13)  'Carriage return character
Const GT=CHR(62)  'Greater than symbol (>). Valid response from 950R.
Const STX=CHR(2)  'Start of Text character. Indicates beginning of line of 950T data.
Const Comma=CHR(44)  'Comma character (,). Used to sort elements within 950T line.
Const R950_1=Com1  'Set 950R #1 as being connected on CR1000 COM1.
Const T950_Nmbr = 16  'total number of 950Ts talking to this 950R.
Const On = True  'I like using "On" and "Off" vs. True & False.
Const Off = False
```


815 W. 1800 N., Logan, UT 84321-1784 (435) 227-9000

```
Public PTemp, batt_volt  'Datalogger temperature and battery voltage.
Public sub950R  'Subroutine variable that contains the 950R com port to work with.
Public addr  'Contains the 950T address from the retrieved 950R data.
Public raw950r As String * 1500  'Contains the complete data dump from the 950R.
Public RawBlkSens(T950_Nmbr) As String * 100 'Contains individual 950T arrays from the 950R data dump.
Public dummystr(12) As String 'Contains the individual string elements from a single 950T array.
'Elements for this string are as follows:
'
'NOTE: I don't keep all of these elements!
'
'1. Month : MM
'2. Day : DD
'3. Year : YY
'4. Hour: HH
'5. Minute: MM
'6. Address : Hexadecimal 0 - F
'7. Mode : See Irrometer manual
'8. WaterMark Block 'A' : Hexadecimal
'9. WaterMark Block 'B' : Hexadecimal
'10. WaterMark Block 'C' : Hexadecimal
'11. WaterMark Block 'D' : Hexadecimal
'12. Battery Voltage : Hexadecimal & math!
'
Public BlockVal(T950_Nmbr,12) 'Contains the fully converted float values from the 950R data dump.
Public R905_1S(T950_Nmbr,7)  'Elements kept from the 950R #1 data dump.
'
'Alias names for the individual stored elements. Have to key this all in
'manaully to match the number the 950Ts used in the field.
'
Alias R905_1S(1,1) = Z1_HM : Alias R905_1S(1,2) = Z1_Addr
Alias R905_1S(1,3) = Zone1A : Alias R905_1S(1,4) = Zone1B
Alias R905_1S(1,5) = Zone1C : Alias R905_1S(1,6) = Zone1D
Alias R905_1S(1,7) = Z1_BattV

Alias R905_1S(2,1) = Z2_HM : Alias R905_1S(2,2) = Z2_Addr
Alias R905_1S(2,3) = Zone2A : Alias R905_1S(2,4) = Zone2B
Alias R905_1S(2,5) = Zone2C : Alias R905_1S(2,6) = Zone2D
Alias R905_1S(2,7) = Z2_BattV

Alias R905_1S(3,1) = Z3_HM : Alias R905_1S(3,2) = Z3_Addr
Alias R905_1S(3,3) = Zone3A : Alias R905_1S(3,4) = Zone3B
Alias R905_1S(3,5) = Zone3C : Alias R905_1S(3,6) = Zone3D
Alias R905_1S(3,7) = Z3_BattV

Alias R905_1S(4,1) = Z4_HM : Alias R905_1S(4,2) = Z4_Addr
Alias R905_1S(4,3) = Zone4A : Alias R905_1S(4,4) = Zone4B
Alias R905_1S(4,5) = Zone4C : Alias R905_1S(4,6) = Zone4D
```


815 W. 1800 N., Logan, UT 84321-1784 (435) 227-9000

```
Alias R9O5_1S(4,7) = Z4_BattV
Alias R9O5_1S(5,1) = Z5_HM : Alias R9O5_1S(5,2) = Z5_Addr
Alias R9O5_1S(5,3) = Zone5A : Alias R9O5_1S(5,4) = Zone5B
Alias R9O5_1S(5,5) = Zone5C : Alias R9O5_1S(5,6) = Zone5D
Alias R9O5_1S(5,7) = Z5_BattV

Alias R9O5_1S(6,1) = Z6_HM : Alias R9O5_1S(6,2) = Z6_Addr
Alias R9O5_1S(6,3) = Zone6A : Alias R9O5_1S(6,4) = Zone6B
Alias R9O5_1S(6,5) = Zone6C : Alias R9O5_1S(6,6) = Zone6D
Alias R9O5_1S(6,7) = Z6_BattV

Alias R9O5_1S(7,1) = Z7_HM : Alias R9O5_1S(7,2) = Z7_Addr
Alias R9O5_1S(7,3) = Zone7A : Alias R9O5_1S(7,4) = Zone7B
Alias R9O5_1S(7,5) = Zone7C : Alias R9O5_1S(7,6) = Zone7D
Alias R9O5_1S(7,7) = Z7_BattV

Alias R9O5_1S(8,1) = Z8_HM : Alias R9O5_1S(8,2) = Z8_Addr
Alias R9O5_1S(8,3) = Zone8A : Alias R9O5_1S(8,4) = Zone8B
Alias R9O5_1S(8,5) = Zone8C : Alias R9O5_1S(8,6) = Zone8D
Alias R9O5_1S(8,7) = Z8_BattV

Alias R9O5_1S(9,1) = Z9_HM : Alias R9O5_1S(9,2) = Z9_Addr
Alias R9O5_1S(9,3) = Zone9A : Alias R9O5_1S(9,4) = Zone9B
Alias R9O5_1S(9,5) = Zone9C : Alias R9O5_1S(9,6) = Zone9D
Alias R9O5_1S(9,7) = Z9_BattV

Alias R9O5_1S(10,1) = Z10_HM : Alias R9O5_1S(10,2) = Z10_Addr
Alias R9O5_1S(10,3) = Zone10A : Alias R9O5_1S(10,4) = Zone10B
Alias R9O5_1S(10,5) = Zone10C : Alias R9O5_1S(10,6) = Zone10D
Alias R9O5_1S(10,7) = Z10_BattV

Alias R9O5_1S(11,1) = Z11_HM : Alias R9O5_1S(11,2) = Z11_Addr
Alias R9O5_1S(11,3) = Zone11A : Alias R9O5_1S(11,4) = Zone11B
Alias R9O5_1S(11,5) = Zone11C : Alias R9O5_1S(11,6) = Zone11D
Alias R9O5_1S(11,7) = Z11_BattV

Alias R9O5_1S(12,1) = Z12_HM : Alias R9O5_1S(12,2) = Z12_Addr
Alias R9O5_1S(12,3) = Zone12A : Alias R9O5_1S(12,4) = Zone12B
Alias R9O5_1S(12,5) = Zone12C : Alias R9O5_1S(12,6) = Zone12D
Alias R9O5_1S(12,7) = Z12_BattV

Alias R9O5_1S(13,1) = Z13_HM : Alias R9O5_1S(13,2) = Z13_Addr
Alias R9O5_1S(13,3) = Zone13A : Alias R9O5_1S(13,4) = Zone13B
Alias R9O5_1S(13,5) = Zone13C : Alias R9O5_1S(13,6) = Zone13D
Alias R9O5_1S(13,7) = Z13_BattV
```


815 W. 1800 N., Logan, UT 84321-1784 (435) 227-9000

```
Alias R9O5_1S(14,1) = Z14_HM : Alias R9O5_1S(14,2) = Z14_Addr
Alias R9O5_1S(14,3) = Zone14A : Alias R9O5_1S(14,4) = Zone14B
Alias R9O5_1S(14,5) = Zone14C : Alias R9O5_1S(14,6) = Zone14D
Alias R9O5_1S(14,7) = Z14_BattV

Alias R9O5_1S(15,1) = Z15_HM : Alias R9O5_1S(15,2) = Z15_Addr
Alias R9O5_1S(15,3) = Zone15A : Alias R9O5_1S(15,4) = Zone15B
Alias R9O5_1S(15,5) = Zone15C : Alias R9O5_1S(15,6) = Zone15D
Alias R9O5_1S(15,7) = Z15_BattV

Alias R9O5_1S(16,1) = Z16_HM : Alias R9O5_1S(16,2) = Z16_Addr
Alias R9O5_1S(16,3) = Zone16A : Alias R9O5_1S(16,4) = Zone16B
Alias R9O5_1S(16,5) = Zone16C : Alias R9O5_1S(16,6) = Zone16D
Alias R9O5_1S(16,7) = Z16_BattV

Public BinString As String  'Stores the first block of information when the 95OR wakes up.
Public Sendstr As String  'Characters to send to 95OR.
Public Read95Or1 As Boolean  'Flag that controls data collection from 95OR #1.
Public Success As Boolean  'Flag indicates successful connection to 95OR.
Dim linecntr  'Value holds the number lines returned from the 95OR.
Dim n, i  'Values used as counters in loops.

'Subroutine used to wake up a 95OR and then pull in and parse all the data into
'a generic array.
Sub Wakeup
  'Clear out arrays.
  raw95Or = ""
  BinString = ""
  'Wake up the 95OR by sending an "Escape" character.
  For n = 1 To 3
    SerialBrk (sub95OR,5O)  'MUST send a serial break to wake up the 95OR!
    SerialFlush (sub95OR)
    SerialOut (sub95OR,ESC,"",0,0)
    Delay (1,1OO,mSec)
    Success = False
    'Test for a response back from the 95OR by looking for a '>' symbol.
    'If I don't get the '>' then communication failed and I'm done.
    'Use the flag "Success" to indicate whether communication worked.
    For i = 1 To 3
      SerialInBlock (sub95OR,BinString,1)
      If BinString = GT Then
        Success = True
        ExitFor
      EndIf
    Next i
   If Success Then ExitFor
```

```
Next n
If Success Then
  'Load big array with NANs to indicate no data received.
  For n = 1 To T95O_Nmbr
    For i = 1 To 12
      BlockVal(n,i) = NAN
      If n = 1 Then
        dummystr(i) = ""
      EndIf
    Next i
    'Clear out other arrays.
    RawBlkSens(n) = ""
  Next n
  'Send command to bring in all the data from the 95OR
  Sendstr = "L" + CR
  SerialOut (sub95OR,Sendstr,"",0,0)
  Delay (1,2,Sec)
  SerialInBlock (sub95OR,raw95Or,15OO)
  'Split out individual lines of data.
  SplitStr (RawBlkSens(),raw95Or,STX,T95O_Nmbr,4)
  linecntr = 0
  'Count up the number lines actually received for parsing control.
  'This value should be fixed at 16 but I'm testing it anyway.
  For n = 1 To T95O_Nmbr
    If RawBlkSens(n) <> "" Then linecntr = linecntr + 1
  Next n
  'Split out individual elements for each line of data.
  'Convert hexadecimal values to decimal where required.
  For n = 1 To linecntr
    SplitStr (dummystr(),RawBlkSens(n),Comma,12,4)
    For i = 1 To 12
      If i >= 1 AND i < 6 Then
        BlockVal(n,i) = dummystr(i)
      Else
        'Pull out the address of the 95OT. Ignore the base address.
        If i = 6 Then
          BlockVal(n,i) = Right (dummystr(i),1)
        EndIf
        'Convert all hexadecimal characters to decimal values.
        BlockVal(n,i) = HexToDec (dummystr(i))
        'Convert the battery voltage to a full number.
        If i = 12 Then  BlockVal(n,i) = BlockVal(n,i) * 0.038
      EndIf
    Next i
  Next n
Else
```

```
  EndIf
EndSub
'Define Data Tables
DataTable (R95O,1,-1)
  DataInterval (0,1,Hr,10)
  Sample (5,R9O5_1S(1,3),FP2)
  Sample (5,R9O5_1S(2,3),FP2)
  Sample (5,R9O5_1S(3,3),FP2)
  Sample (5,R9O5_1S(4,3),FP2)
  Sample (5,R9O5_1S(5,3),FP2)
  Sample (5,R9O5_1S(6,3),FP2)
  Sample (5,R9O5_1S(7,3),FP2)
  Sample (5,R9O5_1S(8,3),FP2)
  Sample (5,R9O5_1S(9,3),FP2)
  Sample (5,R9O5_1S(10,3),FP2)
  Sample (5,R9O5_1S(11,3),FP2)
  Sample (5,R9O5_1S(12,3),FP2)
  Sample (5,R9O5_1S(13,3),FP2)
  Sample (5,R9O5_1S(14,3),FP2)
  Sample (5,R9O5_1S(15,3),FP2)
  Sample (5,R9O5_1S(16,3),FP2)
EndTable

'Main Program
BeginProg
  'Open serial port used with the 95Or
  SerialOpen (R95O_1,96OO,3,0,15OO)

  'Load -6999 in all locations first time program runs to catch no
  'information coming from a unit.
  For n = 1 To T95O_Nmbr
    For i = 1 To 6
      R9O5_1S(n,i) = -6999
    Next i
  Next n

  Scan (6O,Sec,0,0)
    PanelTemp (PTemp,25O)
    Battery (batt_volt)
    'At five minutes before the top of the hour read all three 95ORs.
    If TimeIntoInterval (55,6O,Min) Then Read95Or1 = On

    'If the control for 95OR #1 is set then read it and process the data.
    'The trick here is that the data does not arrive in any particular order
    'so the address must be identified and associated with the correct array.
```

```
    If Read950r1 Then
      Read950r1 = Off
      sub950R = R950_1
      Call Wakeup
      If Success Then
        For n = 1 To linecntr
          addr = BlockVal(n,6)
          'Read in the seven elements I want to keep.
          For i = 1 To 7
            'Convert the separate HOUR and Minute values to one value - HHMM.
            If i = 1 Then
              R905_1S(addr+1,1) = BlockVal(n,4)*100+BlockVal(n,5)
            ElseIf i = 2 Then
              'Convert the address which starts at 0 to something that
              'starts at one.
              R905_1S(addr+1,2) = BlockVal(n,6) + 1
            Else
              'Bring in four WaterMark block values and battery voltage.
              R905_1S(addr+1,i) = BlockVal(n,i+5)
            EndIf
          Next i
        Next n
      Else
        'If "Success" is false then load all the arrays with -6999.
        For n = 1 To T950_Nmbr
          For i = 1 To 7
            R905_1S(n,i) = -6999
          Next i
        Next n
      EndIf
    EndIf

    CallTable R950
  NextScan
EndProg
```


815 W. 1800 N., Logan, UT 84321-1784 (435) 227-9000