uAPP222 (v1.0) July 12,2005

Interfacing the MICROCOM DESIGN GTX Satellite Transmitter to the Campbell Scientific CR1000 Data Logger

Author: Richard Schwarz

PRELIMINARY

SUMMARY

The Microcom Model GTX Satellite Transmitter and Data Collector works on GOES, GMS, ARGOS, SCD & METEOSAT systems. The GTX has some data logger functions built into it, including an SDI-12 and counter input. The GTX can interface to external data acquisition systems via its RS-232 port.

Microcom GTX Satellite Transmitter

Campbell Scientific Programmable Data Logger

For example, the CR1000 from Campbell Scientific is an external Data logger / Data Acquisition system which can be interfaced to various sensors to log measurements. The CR1000 has its own programming language which allows programs to be written for various application scenarios. By connecting the CR1000 to the Microcom Design GTX Modulator, programs can be written on the CR1000 and logged measurements can then be forwarded via the GTX serial port and then sent via GOES, GMS, ARGOS, SCD and METEOSAT satellite systems.

INTRODUCTION

This Application note goes over basic interfacing of the Microcom GTX Satellite Transmitter and Campbell Scientific Data Logger. This including hardware connections and sample software applications, and code snippets. Issues covered include:

1) Interfacing the serial port of the GTX to the CR1000.
2) Programming the CR1000 to handle the serial methodology which the Micrcom GTX uses for going into and out of sleep (low power) mode. The GTX has a wake up mode which takes a time period to respond back to a wake up command (Carriage Return) being sent to the GTX. The GTX responds back with a '>' prompt which should be used by the CR1000 to send data to the GTX.
3) Setting the internal time of the CR1000 to the more accurate time of the GTX.
4) A possible CR1000 "pass through mode" implementation , such that one serial connection (from PC) to the CR1000 could be used to talk to both the CR1000 and the GTX through the CR1000.

User's of this application note should be familiar with, or have access to basic coding techniques, user's manuals and software associated with using the GTX and CR1000, including:

1) Microcom GTX User's Manual
2) Microcom GTX GUI Software
3) Campbel Scientific CR1000 User's Manual
4) Campbell Scientific LoggerNet Software

Throughout this application note, one of the CR1000 serial ports is used to send debug messages to a terminal screen for verification and testing.

GTX to CR1000 Connections:

Figure 1 is a block diagram which shows the connections used for this Application note. Our test PC uses three separate serial ports which are used to interface and program the GTX and the CR1000 data logger.

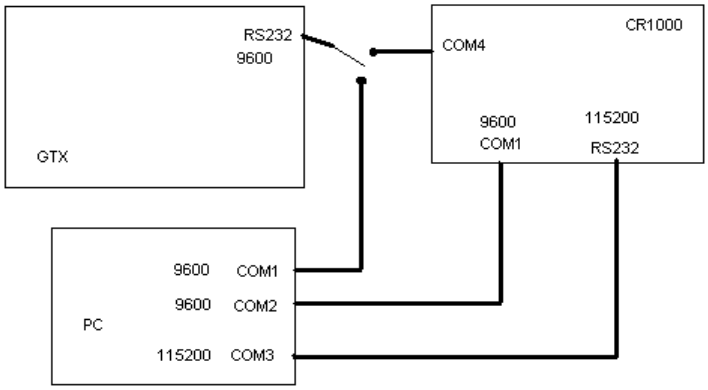


Figure 1 GTX to CR1000 EXAMPLE SERIAL CONNECTIONS

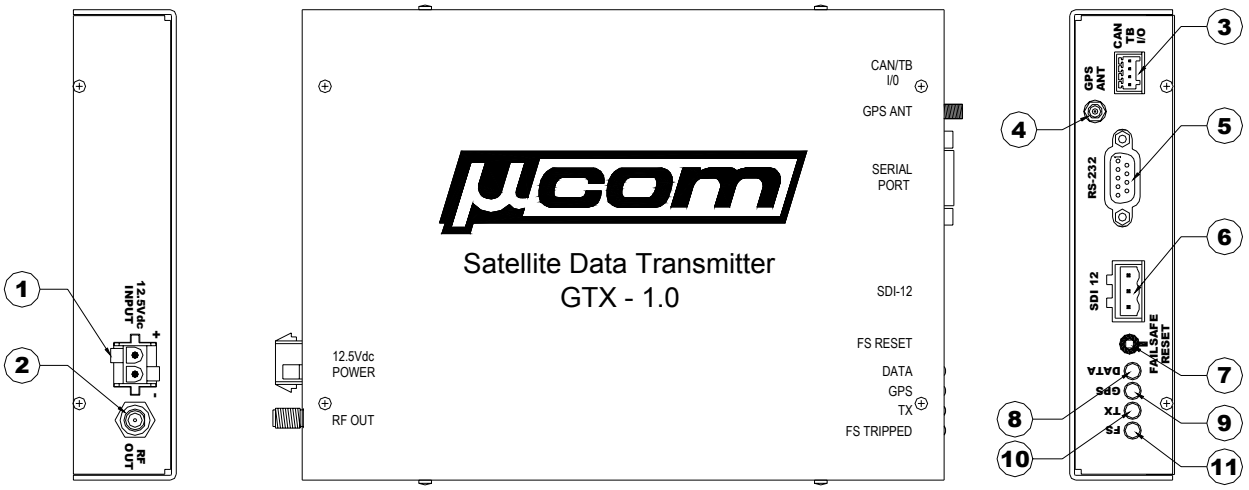


Figure 2 Microcom GTX Satellite Transmitter

1. Main Power Input Connector - +12.5 VDC
2. RF Output Connector
3. CAN (future) and Tipping Bucket Connector
4. GPS Antenna Connector
5. RS-232 Serial Port Connector
6. SDI-12 Interface Connector
7. Failsafe Reset Push-Button
8. LED4: Data in Transmit Buffer LED (Green)
9. LED3: GPS Receive Active LED (Green)
10. LED2: RF Transmit Active LED (Green)
11. LED1: Failsafe Tripped LED (Red)

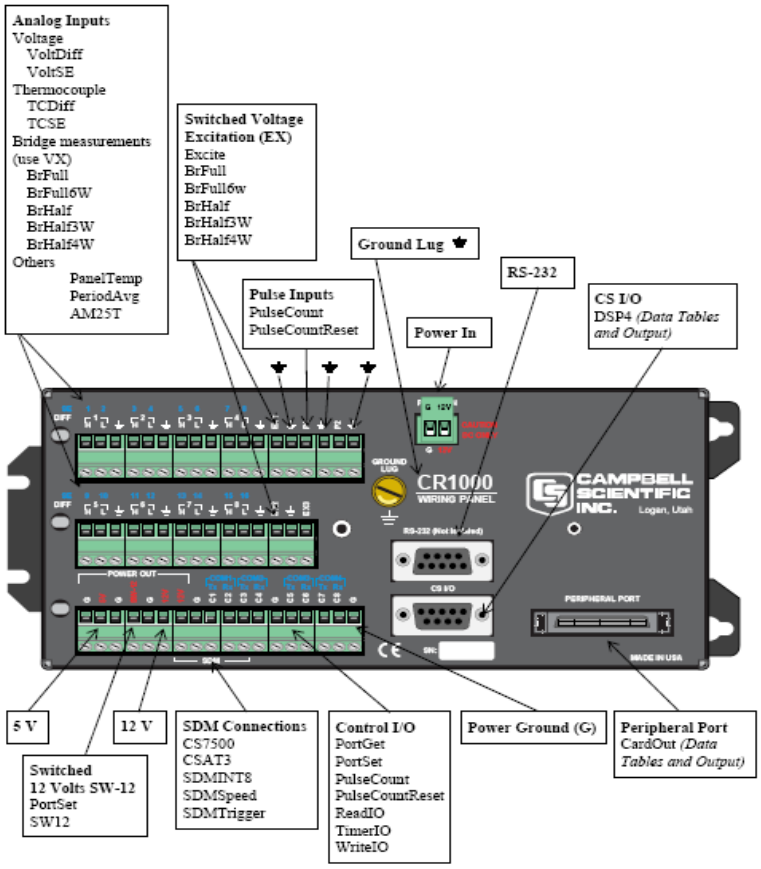


Figure 3 CR1000

Figure 2 and 3 are diagrams of the GTX and CR1000. As can be seen there are two D9 serial ports shown on the CR1000 device. One is a proprietary serial port (TTL and non standard pin out), while the other is a standard RS232 port. It is certainly feasible to connect the GTX to the RS232 D9, however, upon further examination it can be seen that on the lower terminal block, that 4 each 2 wire com ports are available. These ports were TTL level on older Campbell devices like the CRX10. However on the CR1000 these ports can be programmed for RS232 levels.

The main connections to the CR1000 are shown below. A direct connection from the PC through a standard 232 cable is made to communicate and program the CR1000. Power (+12v) is connected to the POWER IN port of the CR1000.

For our setup we will use COM4 to connect to the RS232 serial line of the GTX. A three wire cable is connected as shown.

Figure 3 CR1000 Connections

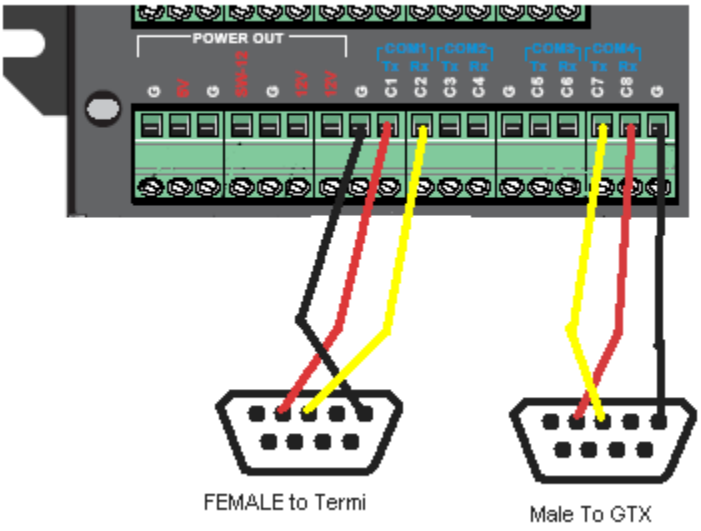


Figure 3 CR1000 Serial Connections

	9 pin
Receive Data	2
Transmit Data	3
System Ground	5

D9 Signals

CR1000 Programming

The first test program we will write will test the ability of the CR1000 to wake up the GTX. Using the LOGGERNET software we will write a CRBASIC program which responds to the GTX '>' prompt. A second serial cable was added from the CR1000 COM4 to be used as a debug port.

The program listing is shown below:

```
'CR1000 Series Datalogger
'To create a different opening program template, type in new
'instructions and select Template | Save as Default Template
'date:
'program author: R. Schwarz

'Declare Public Variables
'Example:
Public PTemp, batt_volt,
Public CR as string * 1
Public LF as string * 1
Public CRLF as string * 2
Public PROMPT as string * 1
Public MESSAGE as string * 100

'Declare Other Variables
'Example:
'Dim Counter

'Declare Constants
'Example:
'CONST PI = 3.141592654

'Define Data Tables
DataTable (Test,1,-1)
        DataInterval (0,15,Sec,10)
        Minimum (1,batt_volt,FP2,0,False)
        Sample (1,PTemp,FP2)
EndTable

'Define Subroutines
'Sub
        'EnterSub instructions here
'EndSub

'Main Program
BeginProg
  CR=CHR(13)
  LF=CHR(10)
  CRLF=CHR(13)+CHR(10)
  PROMPT= ">"
  MESSAGE = "MICROCOM DESIGN TEST MESSAGE TEMP "
  SerialOpen (Com1,9600,0,0,10000)
  SerialOpen (Com4,9600,0,0,10000)

        Scan (1,Sec,0,0)
                PanelTemp (PTemp,250)
                Battery (Batt_volt)
                'Enter other measurement instructions
                'Call Output Tables
                'Example:
                CallTable Test

                'SerialOutBlock (Com1,CR,1)
                SerialOut (Com4,CR,PROMPT,100,100)  'wake upGTX
```

```
                SerialOut (Com1,CR,"",0,0)
                SerialOut (Com4,MESSAGE,"",0,100)
                SerialOut (Com1,MESSAGE,"",0,100)
                SerialOut (Com4,batt_volt,"",0,100)
                SerialOut (Com4, CRLF,"",0,100)
                SerialOut (Com1,batt_volt,"",0,100)
                SerialOut (Com1, CRLF,"",0,100)
        NextScan
EndProg
```

In the program, internal temperature and voltage measurements are taken in the CR1000 and sent out in an ASCII message header. We haven't covered the GTX command structure/format and so this program will simply allow the GTX to go into sleep mode, and then wake it up with carriage returns until we get the '**>**' character returned from the GTX signifying that it is awake and ready for communication. Later, we will format the data into the actual GTX format for a random or timed message.

Test Program #2 Random Data Message

In this next test program we will format the data sent by the CR1000 to enable the GTX to include the data messages in its buffer for a random transmission.

First we will send a Random Data message from the CR1000 to the GTX. In our case we will use the CR1000 Battery Voltage. We will also read the GTX's Time and Date and set the CR1000 local clock to the GTX time.

First we must set up the GTX. Using the Microcom GTX Utility software we can set the GTX up for Random transmission. The Microcom GTX utility runs on a PC and requires a PC serial port to connect to the GTX serial port. For now, we do this by disconnecting the PC connection from the CR1000 so that the PC is connected to the GTX serial port (instead of the CR1000). Once the GTX is set up and enabled, then the GTX serial connection should be reattached to the CR1000 COM4.

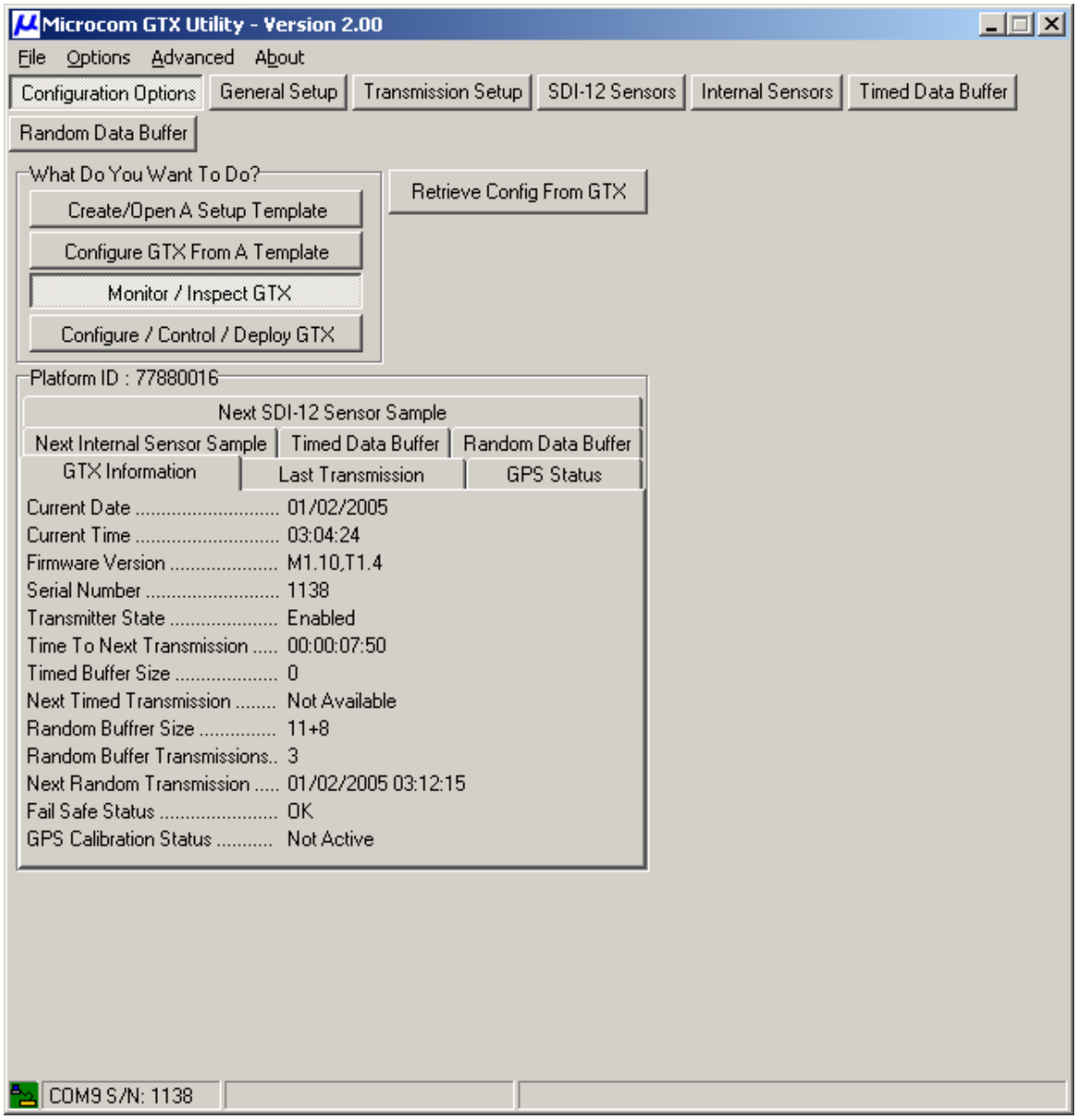
The full use of the GTX utility is beyond the scope of this application note and is not going to be covered in this application note. The GTX needs to be set to accept Random Data Tx and it needs to be enabled. A configuration file was saved called **ricktestConfigFile.txt** which does this. After opening up the file you also need to ENABLE the GTX.

NOTE

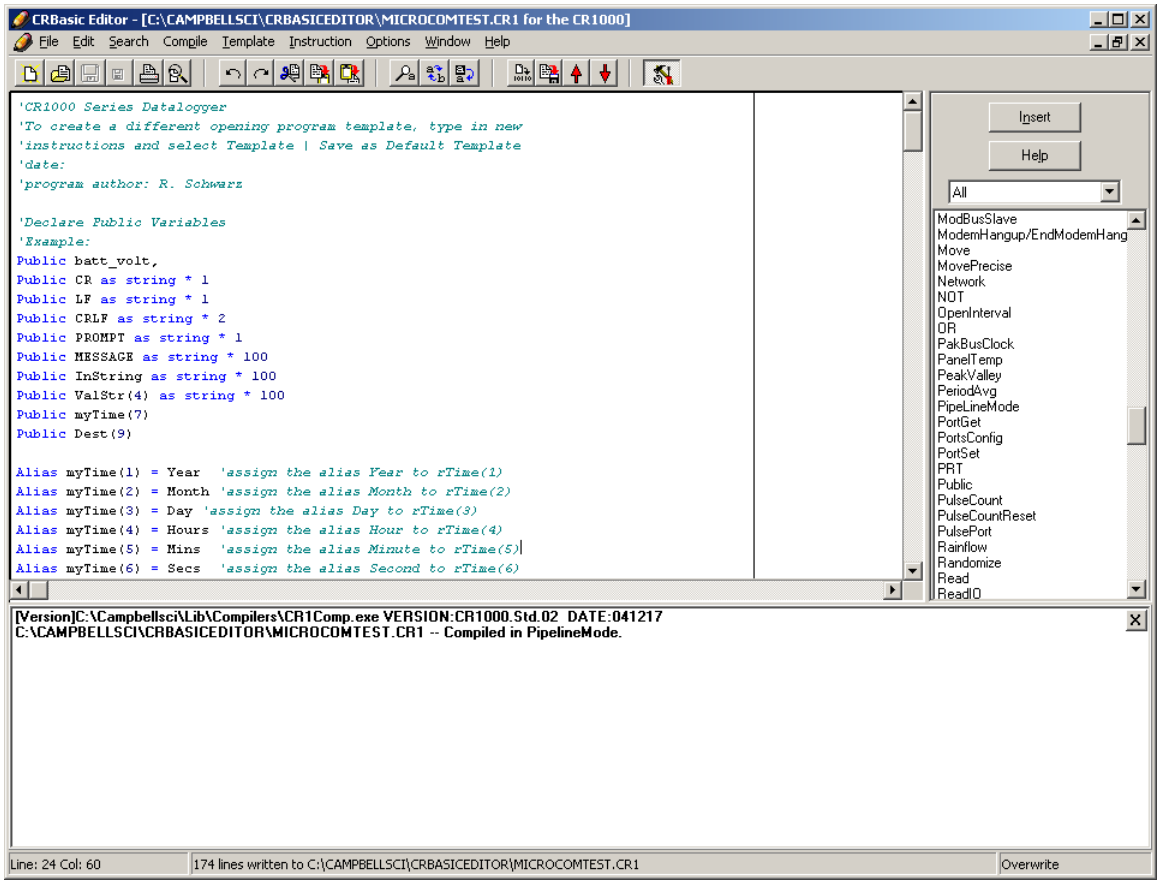
CR1000 programs can be written with commands which include the setup from for the GTX. For example the rickstestConfigFile.txt contains commands like:

NID=77880016

Which can be sent in the CR1000 by sending "NID=77880016" out the com port connected to the GTX. A good strategy for code development of the CR1000 GTX interface is to set up the GTX using the GTX GUI, and then saving off a configuration you know works, and then writing a setup subroutine for the GTX in the CR1000 CRBASIC code based on the setup text file.



Once all this is done we use the LOGGERNET software's CRBASIC editor to actually write the program. Once again the Loggernet Software's capabilities are beyond the scope of this application note, and it is assumed that the user is somewhat familiar with the Loggernet software.



The code for our program is shown below called MICROCOMDATA.CR1:

```
' Program: GTXRand.cr1
' Description: GTX Random Data Transmission Program
' CR1000 Series Datalogger forwa"RandomData=" Battery Voltage and Temperature
' to the GTX in Random Mode. Time and Date is read from the GTX using
' Subroutine GetGTXTime
'
'program author: R. Schwarz  Microcom Design Incorporated
'program date: Original Code May 1, 2005
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Declare Public Variables''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Public PTemp
Public CR as string * 1
Public LF as string * 1
Public CRLF as string * 2
Public PROMPT as string * 1
Public myTime(7)
Public Dest(9)
Public InString as string * 100
Public ValStr(4) as string * 100
Public LINE as string * 100
Public Mess1 as string * 100
Public Mess2 as string * 100
Public Batt_volt
'
```

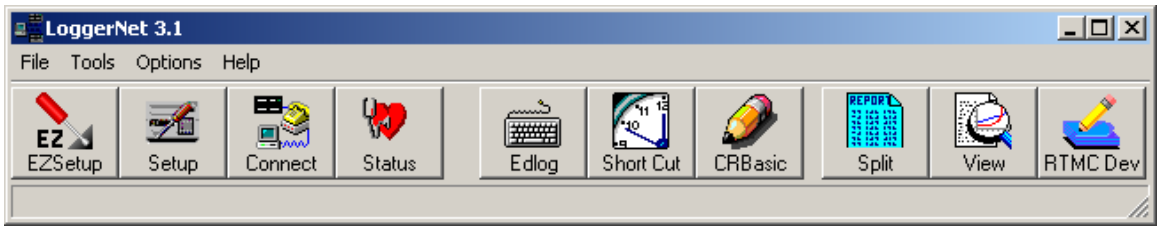
```
Units Batt_Volt=Volts
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Constant Declarations'''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Const DebugOnOff = 1



''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Alias Declarations''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Alias myTime(1) = Year        'assign the alias Year to rTime(1)
Alias myTime(2) = Month       'assign the alias Month to rTime(2)
Alias myTime(3) = Day         'assign the alias Day to rTime(3)
Alias myTime(4) = Hours       'assign the alias Hour to rTime(4)
Alias myTime(5) = Mins        'assign the alias Minute to rTime(5)
Alias myTime(6) = Secs        'assign the alias Second to rTime(6)
Alias myTime(7) = uSecs       'assign the alias uSecond to rTime(7)

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Define Data Tables''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

'Test Data Table
DataTable (TempTbl,1,-1)
        DataInterval (0,10,Sec,10)
        Minimum (1,batt_volt,FP2,0,False)
        Sample (1,PTemp,FP2)
EndTable


''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Define Subroutines''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Sub Initialize()
' Use Com1 as an ECHO output port to viwe on Com1
  SerialOpen (Com1,9600,0,0,10000)
' Setup Com4 serial port to GTX
  SerialOpen (Com4,9600,0,0,10000)
   CR=CHR(13)
   LF=CHR(10)
   CRLF=CHR(13)+CHR(10)
   PROMPT= ">"
   LINE="_____________________"
   Mess1="CR1000 TO Com4->"
   Mess2="Com4 TO CR1000->"
EndSub
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'  Get the Microcom GTX Time
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Sub GetGTXTime()
'   Request Time from GTX
                SerialOut(Com4,"Time?","",0,100)
                SerialOut (Com4, CRLF,"",0,100)
'   Echo stuff to Debug Terminal
                If DebugOnOff = 1 Then
                        SerialOut(Com1,Mess1,"",0,100)
                        SerialOut(Com1,"Time?","",0,100)
                        SerialOut (Com1, CRLF,"",0,100)
                EndIf
'   Get GTX Responce
                SerialIn (InString,Com4,100,10,100)

'   Echo stuff to Debug Terminal
                If DebugOnOff = 1 Then
```

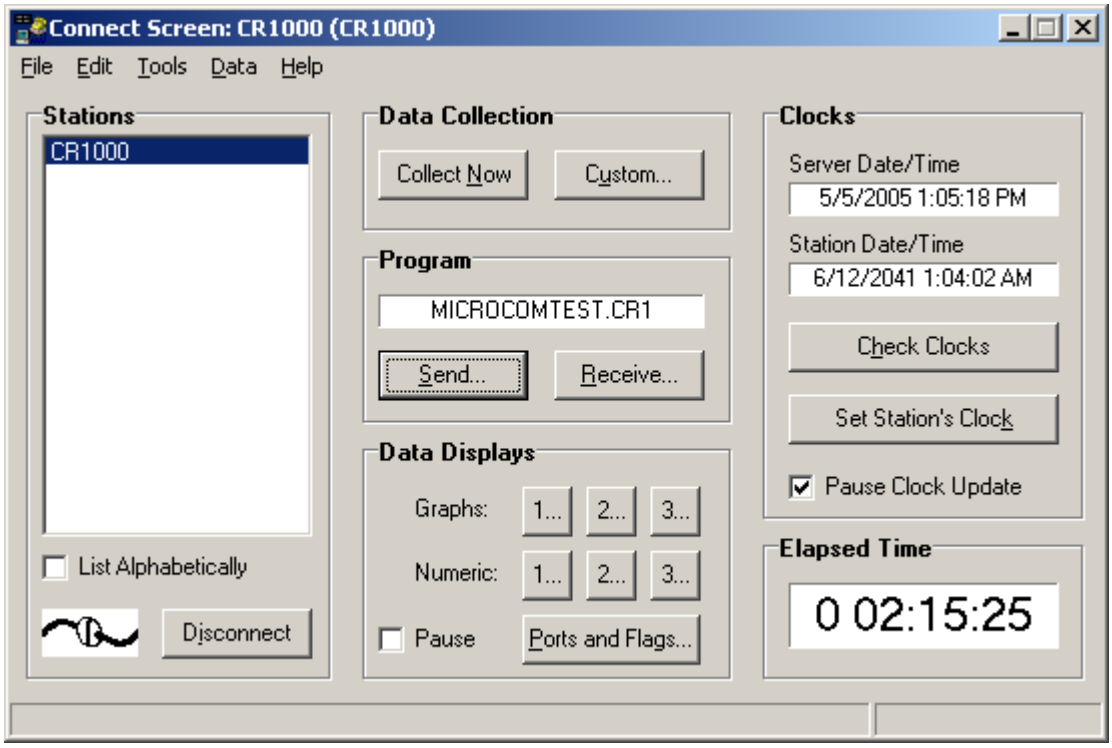
```
                    SerialOut(Com1,Mess2,"",0,100)
                    SerialOut (Com1,InString,"",0,100)
                EndIf

                SerialIn (InString,Com4,100,10,100)
'   Echo stuff to Debug Terminal
                If DebugOnOff = 1 Then
                    SerialOut(Com1,Mess2,"",0,100)
                    SerialOut (Com1,InString,"",0,100)
                    SerialOut (Com1, CRLF,"",0,100)
                    SerialOut (Com1,LINE,"",0,100)
                    SerialOut (Com1, CRLF,"",0,100)
                EndIf

                'Breakup Time String
                SplitStr (ValStr(1),InString,":",1,0)
                SplitStr (ValStr(2),InString,":",2,4)
                SplitStr (ValStr(4),ValStr(3),".",1,4)
                SplitStr (ValStr(3),ValStr(3),".",1,5)

            'Assign Hours
                Hours = ValStr(1)

                If DebugOnOff = 1 Then
                    SerialOut(Com1,"Hours->","",0,100)
                    SerialOut(Com1,ValStr(1),"",0,100)
                    SerialOut (Com1, "  ","",0,100)
                EndIf

                'Assign Mins
                Mins = ValStr(2)

                If DebugOnOff = 1 Then
                    SerialOut(Com1,"Mins->","",0,100)
                    SerialOut(Com1,ValStr(2),"",0,100)
                    SerialOut (Com1, "   ","",0,100)
                EndIf

                'Assign Seconds
                Secs = ValStr(3)

                If DebugOnOff = 1 Then
                    SerialOut(Com1,"Secs->","",0,100)
                    SerialOut(Com1,ValStr(3),"",0,100)
                    SerialOut (Com1, "   ","",0,100)
                EndIf

                'Assign Microsecs
                uSecs = ValStr(4)

                If DebugOnOff = 1 Then
                    SerialOut(Com1,"millisecs->","",0,100)
                    SerialOut(Com1,ValStr(4),"",0,100)
                    SerialOut (Com1, CRLF,"",0,100)
                EndIf

                'Request Date from Com4
                SerialOut(Com4,"Date?","",0,100)
                SerialOut (Com4, CRLF,"",0,100)
                SerialIn (InString,Com4,100,10,100)
                SerialIn (InString,Com4,100,10,100)

                If DebugOnOff = 1 Then
                    SerialOut(Com1,Mess1,"",0,100)
                    SerialOut(Com1,"Date?","",0,100)
```

```
                SerialOut (Com1, CRLF,"",0,100)
        SerialOut(Com1,Mess2,"",0,100)
         SerialOut (Com1,InString,"",0,100)
         SerialOut(Com1,Mess2,"",0,100)
         SerialOut (Com1,InString,"",0,100)
         SerialOut (Com1, CRLF,"",0,100)
         SerialOut (Com1,LINE,"",0,100)
         SerialOut (Com1, CRLF,"",0,100)
        EndIf

        'Break up Date String
        SplitStr (ValStr(1),InString,"/",1,0)
        SplitStr (ValStr(2),InString,"/",2,4)

        'Assign Month
        Month = ValStr(1)

        If DebugOnOff = 1 Then
                SerialOut(Com1,"Month->","",0,100)
                SerialOut(Com1,ValStr(1),"",0,100)
                SerialOut (Com1, "   ","",0,100)
        EndIf

        'Assign Day
        Day = ValStr(2)

        If DebugOnOff = 1 Then
                SerialOut(Com1,"Day->","",0,100)
                SerialOut(Com1,ValStr(2),"",0,100)
                SerialOut (Com1, "   ","",0,100)
        EndIf

        'Assign Year
        Year = ValStr(3)

        If DebugOnOff = 1 Then
                SerialOut(Com1,"Year->","",0,100)
                SerialOut(Com1,ValStr(3),"",0,100)
                SerialOut (Com1, CRLF,"",0,100)
        EndIf

EndSub



''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Main Program''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
BeginProg

   CALL Initialize()

' Infinite loop to be executed every ten seconds
        Scan (10,Sec,0,0)
' take panel temp
                PanelTemp (PTemp,250)
'  take internal cr1000 baterry voltage
                Battery(batt_volt)
'  Call Temperature Table

                CallTable TempTbl

'  Send CRs until getting Com4 > " >"
                SerialOut (Com4,CR,">",100,100)
' Send Baterry Voltage to the Com4 in a Random Data Command
```

```
                SerialOut (Com4,"RandomData=","",0,100)
                SerialOut (Com4,batt_volt,"",0,100)
'   Echo stuff to Debug Terminal
                If DebugOnOff = 1 Then
                        SerialOut(Com1,Mess1,"",0,0)
                        SerialOut (Com1,"RandomData=","",0,100)
                        SerialOut (Com4, CRLF,"",0,100)
                        SerialOut (Com1,batt_volt,"",0,100)
                        SerialOut (Com1, CRLF,"",0,100)
                EndIf
'   Get responce from GTX
                SerialIn (InString,Com4,100,10,100)
'   Echo stuff to Debug Terminal
                If DebugOnOff = 1 Then
                  SerialOut(Com1,Mess2,"",0,0)
                        SerialOut (Com1,InString,"",0,100)
                EndIf
'   Get responce from GTX
                SerialIn (InString,Com4,100,10,100)
'   Echo stuff to Debug Terminal
                If DebugOnOff = 1 Then
                        SerialOut (Com1, CRLF,"",0,100)
                        SerialOut(Com1,Mess2,"",0,100)
                        SerialOut (Com1,InString,"",0,100)
                EndIf
'  Send CRs until getting Com4 > " >"
                SerialOut (Com4,CR,">",100,100)
' Send Temperature to the Com4 in a Random Data Command
                SerialOut (Com4,"RandomData=","",0,100)
                SerialOut (Com4,"Temp:"+PTemp,"",0,100)
'   Echo stuff to Debug Terminal
                If DebugOnOff = 1 Then
                        SerialOut(Com1,Mess1,"",0,0)
                        SerialOut (Com1,"RandomData=","",0,100)
                        SerialOut (Com4, CRLF,"",0,100)
                        SerialOut (Com1,"Temp: "+PTemp,"",0,100)
                        SerialOut (Com1, CRLF,"",0,100)
                EndIf
'   Get responce from GTX
                SerialIn (InString,Com4,100,10,100)
'   Echo stuff to Debug Terminal
                If DebugOnOff = 1 Then
                  SerialOut(Com1,Mess2,"",0,0)
                        SerialOut (Com1,InString,"",0,100)
                EndIf
'   Get responce from GTX
                SerialIn (InString,Com4,100,10,100)
'   Echo stuff to Debug Terminal
                If DebugOnOff = 1 Then
                        SerialOut (Com1, CRLF,"",0,100)
                        SerialOut(Com1,Mess2,"",0,100)
                        SerialOut (Com1,InString,"",0,100)
                EndIf

                ' Get Time from GTX
                Call GetGTXTime()
                ' Set CR1000 Station CLock to the new values
                ClockSet (myTime())

        NextScan
EndProg
```

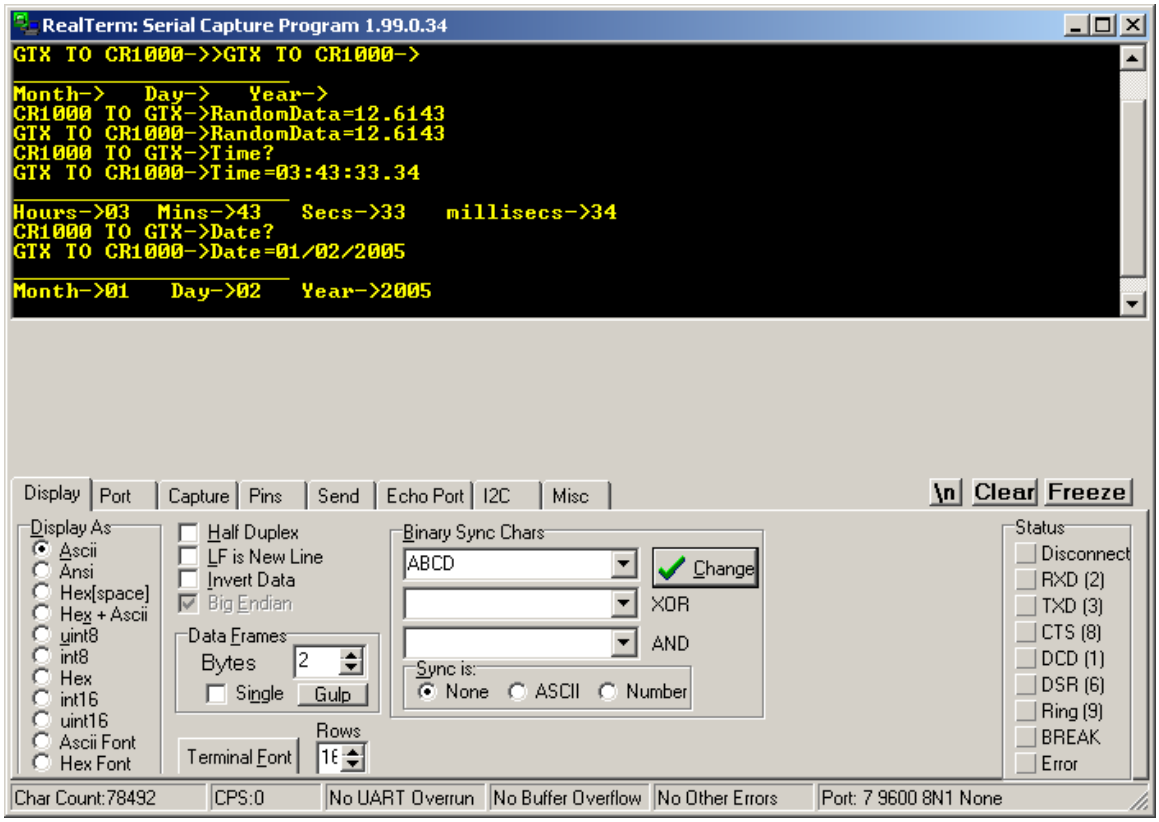
Once the program is written we need to download it to the CR1000 and allow the CR1000 to compile the program. For this we select the CONNECT button on the Logger Net software



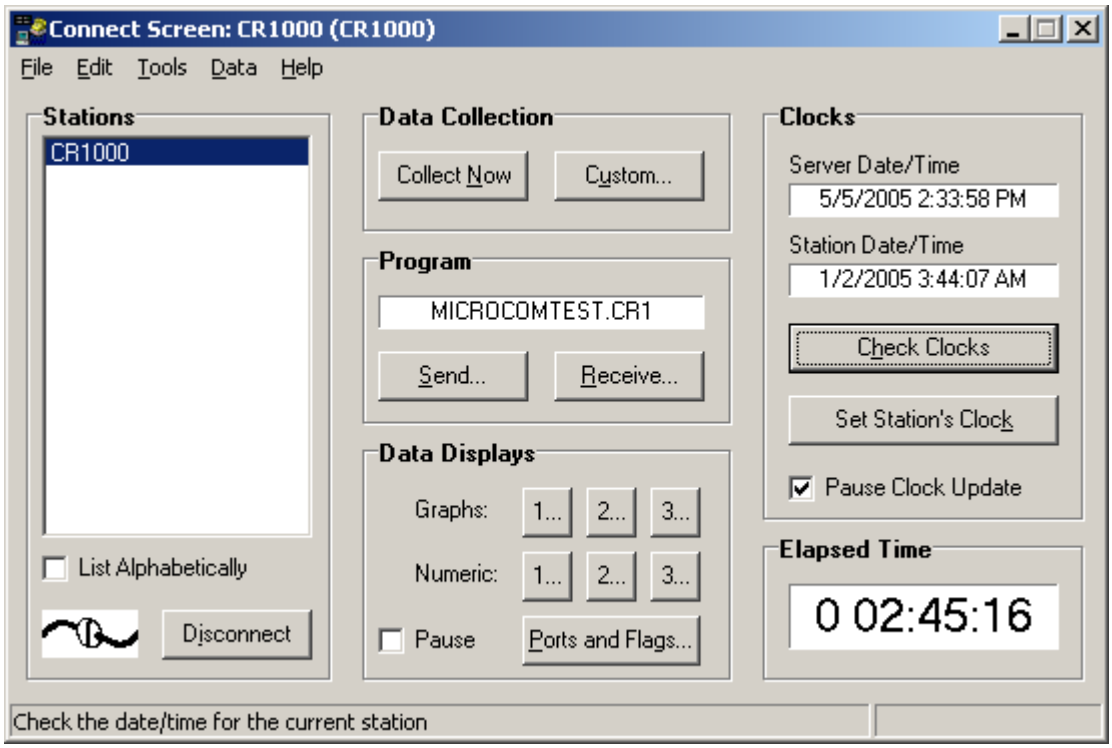


As can be seen on the connect screen we will also be able to read the CR1000 clock from this screen, to verify the time change our program will make.

We can hook a standard terminal to the COM1 port of the CR1000 and view our debug echo outputs as the program runs.



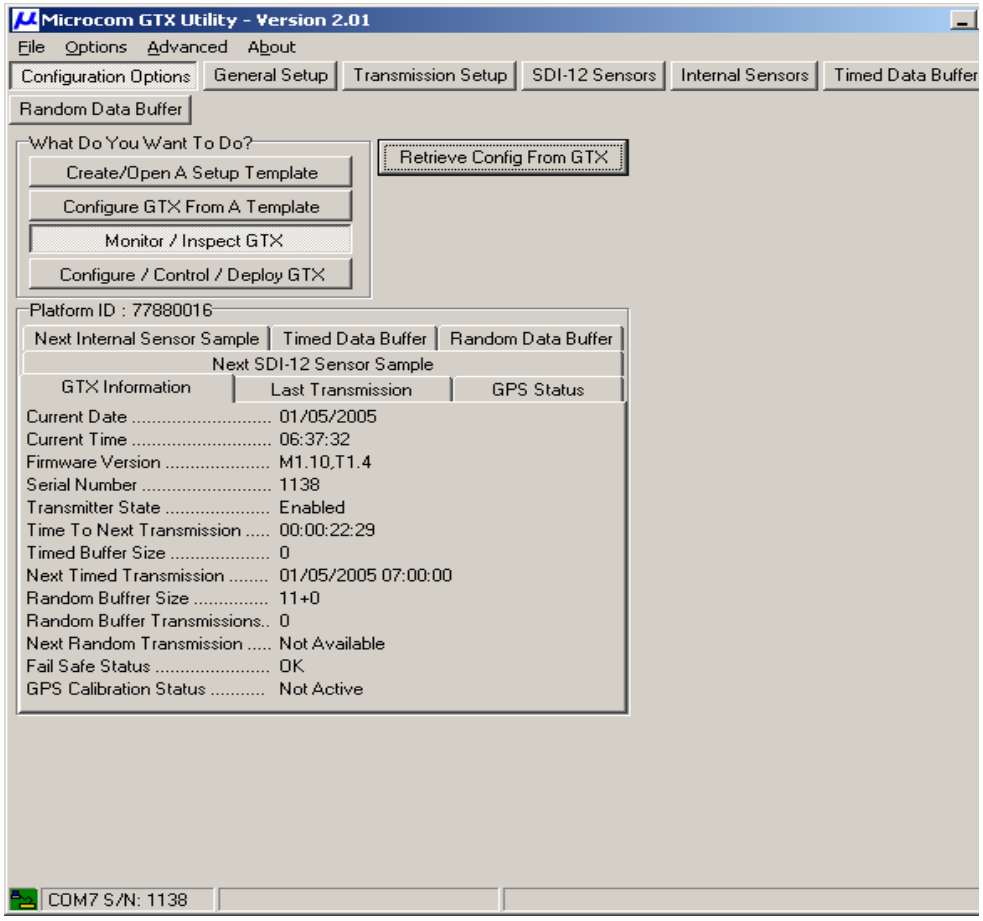
We can then go back and see our updated clock on the Connect Screen.



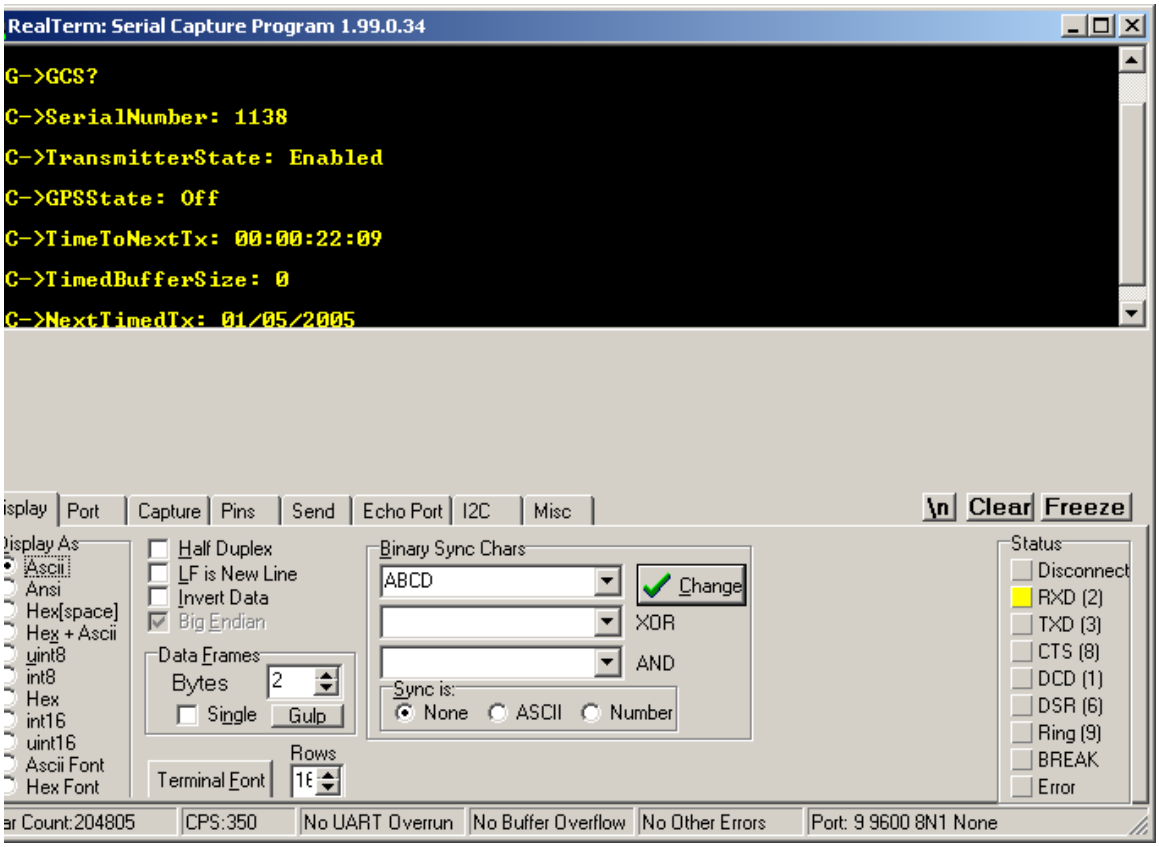
PASS THROUGH MODE

The second serial port (COM4) on the CR1000 can be connected to the PC on another COM port and to a terminal emulator to observe communications of, and talk to, the GTX. The following code was written to implement the pass through operation.

```
' Program: GTXRand.cr1
' Description: GTX Passthrough Mode Attempt
'program author: R. Schwarz  Microcom Design Incorporated
'program date: Original Code May 1, 2005
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Declare Public Variables''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''

Public GTXInString as string * 1000
Public GTXOutString as string * 1000
Public DummyString as string * 10
Public CR as string * 1
Public LF as string * 1
Public CRLF as string * 2
Public SizeRead,Fred

''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Main Program''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
BeginProg

' Use Com1 as an ECHO output port to view on Com1
  SerialOpen (Com1,9600,0,0,1000)
' Setup Com4 serial port to GTX
  SerialOpen (Com4,9600,0,0,1000)
' Setup Com4 serial port to GTX
  SerialOpen (ComRS232,9600,0,0,1000)

   CR=CHR(13)
   LF=CHR(10)
   CRLF=CHR(13)+CHR(10)

Do
                GTXInString=""
                SerialIn (GTXInString,Com4,1,10,100) 'read in from GTX
                'SerialIn (DummyString,Com4,1,10,1) 'read in from GTX
                        if (GTXInString <> "") then
                                SerialOut (ComRS232, GTXInString+LF,"",0,0) 'send to PC Util
            SerialOut (Com1, "G2C->"+GTXInString+CRLF,"",0,0)
            SerialOut (Com1, CRLF,"",0,100)
                        endif

                        GTXOutString =""
                        SerialIn (GTXOutString,ComRs232,1,10,100) 'read in from PC util
                        if (GTXOutString <> "") then
                                SerialOut (Com4, GTXOutString,"",0,0)' send to GTX
            SerialOut (Com1, "C2G->"+GTXOutString+CRLF,"",0,0)
            SerialOut (Com1, CRLF,"",0,100)
    endif

Loop

' Infinite loop to be executed every ten seconds
Scan (10,msec,0,0)

NextScan

EndProg
```



Output can be seen in the following terminal screen output.



A subroutine can be developed which can be called periodically to implement the pass through mode calls in the CRBASIC source code.

```
Sub SerialThru()

        ' handle feed through of comport 1 In
          GTXInString=""
           SerialIn (GTXInString,Com4,1,10,100) 'read in from GTX
           'SerialIn (DummyString,Com4,1,10,1) 'read in from GTX
           if (GTXInString <> "") then
                  SerialOut (ComRS232, GTXInString+LF,"",0,0) 'send to PC Util
                   SerialOut (Com1, "G2C->"+GTXInString+CRLF,"",0,0)
                   SerialOut (Com1, CRLF,"",0,100)
            Endif
      ' handle feed through of comport 1 In
        GTXOutString =""
       SerialIn (GTXOutString,ComRs232,1,10,100) 'read in from PC util
       if (GTXOutString <> "") then
             SerialOut (Com4, GTXOutString,"",0,0)' send to GTX
             SerialOut (Com1, "C2G->"+GTXOutString+CRLF,"",0,0)
            SerialOut (Com1, CRLF,"",0,100)
      endif

EndSub
```

The code can be called periodically in a program:

```
BeginProg

'Initialize
        Call Initialize()

'Execution loop
        Scan(10,msec,1,0)

            SerialThru() ' call pass through code

          'Take other measurements as needed
                'Default Datalogger Battery Voltage measurement Batt_Volt:
                Battery(Batt_Volt)
                '
'
'
'

        NextScan


EndProg
```

Conclusion

The MICROCOM Model GTX Satellite Transmitter and Data collector has some internal data logger capability. For more advanced data acquisition, The GTX can be interfaced to external data loggers like the CR1000 from Campbell Scientific. This application note showed various ways of connecting up and debugging the interface between the GTX and the CR1000.

Revision History

Date	Version	Revision
July 12, 2005	1.0	Initial Microcom Preliminary Release