uAPP223 (v1.0) July 12,2005

Reading Time from the MICROCOM DESIGN GTX Satellite Transmitter into 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 like the Campbell Scientific Programmable Data Logger via its RS-232 port.

Microcom GTX Satellite Transmitter

Campbell Scientific Programmable Data Logger

The GTX internal time keeping is +-0.25 PPM (+-0.65 seconds per month or +-0.02 seconds per day maximum). With GPS this is improved to +-0.1 milliseconds at the GPS update time and drifts at the max rate of +_0.25 PPM until the next GPS update. This is more accurate than the CR1000's internal clock, and so it makes sense to set the CR1000 clock to the GTX clock's more accurate time.

INTRODUCTION

This Application note goes shows how to read the time in the GTX Transmitter and then set the internal time of the CR1000 to the potentially more accurate time of the GTX. Some potential problems will be discussed, with strategies developed to insure the best system operation.

A subroutine is introduced which can be used in CRBASIC programs to bring in the system time from the GTX Transmitter.

The basic setup and connections for the CR1000 and GTX is referenced in the Microcom Application Note uAPP222.

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

The timing on systems operating on platforms like the GOES DCP is critical. Because of the use of timed transmissions with time slot assignments in these systems the transmission clock can be/are required to be accurate to within 0.1 seconds of the actual UTC time. This requirement falls on the transmitters like the GTX which get certified by NOAA/NESDIS. This requirement is often satisfied with the use of a GPS

Reading Time from the MICROCOM DESIGN GTX Satellite Transmitter into the Campbell Scientific CR1000 Data Logger

Author: Richard Schwarz

PRELIMINARY

receiver. The GTX transmitter which is a certified GOES transmitter handles most of the timing requirements automatically, once it is set up for GOES operation. For example, until the GTX clock (time) is set, the GTX will not transmit in GOES mode.

Table 1 shows the maximum drift of the GTX transmitter's internal clock. With GPS this is improved to +-0.1 milliseconds at the GPS update time. We will call this parameter the Transmitter's Update Time Inaccuracy (TUTI)(note that the GTX TUTI is 1000 times better than the specification requirement right after the update).

UNIT	Drift per day (+-) In seconds	Drift per week (+-) In seconds	Drift per month (+-) In seconds	Drift Per year (+-) In seconds
Max Transmitter Clock Drift (MTCD)	0.02166	0.1516	0.65	8

With a GTX clock drift rate of 0.02166 seconds per day and a NESDIS specification for timed transmissions of being within 0.1 seconds of actual UTC, the GTX could, after an initial update run for up to four and a half days before needing another GPS update.

The GTX's Post updated Instantaneous time inaccuracy after a given time (if not updated) is given in table 2 and is simply the TUTI + MTCD.

UNIT	After 1 day (+-) In seconds	After 1 week (+-) In seconds	After 1 month (+-) In seconds	After 1 year (+-) In seconds
GTX's Post Updated Instantaneous Timing Inaccuracy (TPUITI)	0.02176	0.1517	0.6501	8.001

We already mentioned that the GTX clock is more accurate than the logger's clock. The GTX's clock will always drift less then the logger's internal clock. As shown in the following table.

UNIT	Drift per day (+-) In seconds	Drift per week (+-) In seconds	Drift per month (+-) In seconds	Drift Per year (+-) In seconds
GTX Maximum Transmitter Drift	0.02166	0.1516	0.65	8
CR1000 Logger Maximum Drift	0.5	3.45	14.8	180
Max Logger to Transmitter Clock Differential (MLTCD)	0.52166	3.6016	15.45	188

The logger is usually set manually initially. Setting the time manually will most likely have an initial update time inaccuracy which we will call Logger Update Time

Inaccuracy (LUTI) which represent the difference between the logger time set and the actual UTC time.

The Maximum Logger to Tx Clock Differential (MLTCD) for our system in a given time period can be estimated given the following formula:

MLTCD = MaxLoggerDrift + MaxGTXDrift

The MLTCD inaccuracy for a given time period would be:

MLTCDI = LUTI + MLTCD

Lastly we can define a third value called the Maximum System to UTC time Inaccuracy (MSUI) which is simply:

MSUI = MLTCDI + TUTI

This would represent the maximum timing inaccuracy from UTC time that the Logger sees. It DOES NOT represent the GTX transmission time error which is determined only by the transmitter clock drift and the time of the last GPS update and is much lower.

One might ask how much of a problem is having an inaccurate drifting Logger clock really is, since the GTX's transmission time requirement is met, with a superior GTX clock.

Let's do an example.

In our example scenario we will assume we have a 2 minute Logger Update Time Inaccuracy. This represents the difference between the time manually set in the logger and the time in the GTX clock which we will assume got a GPS lock and is essentially UTC time.

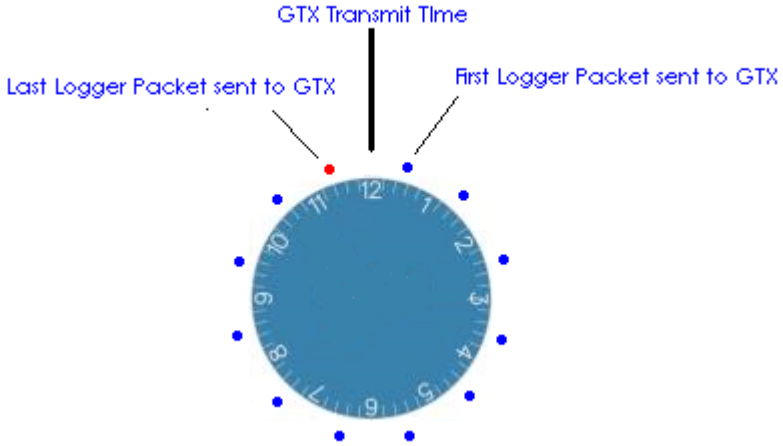
Well lets say that in our scenario we send data to the GTX from the logger every five minutes and we start the first logger send at 2.5 minutes after the top of the hour.

uAPP223 (v1.0) July 12,2005

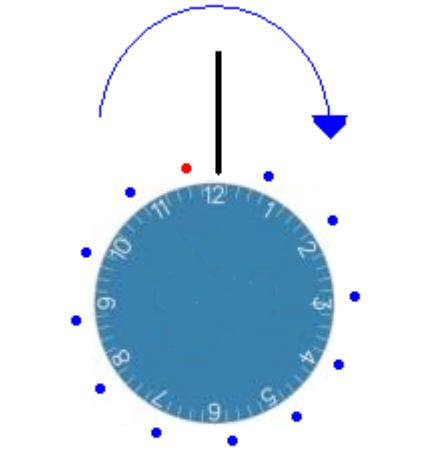
Reading Time from the MICROCOM DESIGN GTX Satellite Transmitter into the Campbell Scientific CR1000 Data Logger

Author: Richard Schwarz

PRELIMINARY



Also assume that we transmitted from the GTX at the top of the hour every 4 hours. The added 2.5 minutes will insure that we are well away from the top of the hour (when the GTX transmits) when we send the first GTX packet from the Logger. So each GTX transmission will contain 12 data packets form the logger. Lets subtract the logger update time inaccuracy from our offset of 2.5 minutes which only leaves us a timing margin of 30 seconds. If the drift error is in the same direction as the offset error, --and lets say the logger clock is running faster than the GTX clock--, and that the offset error places the initial logger time at 2 minutes fast.



Logger Clock drift (logger clock fast) causes the actual packet send times to rotate relative to actual UTC time.

Then from the MLTCD values shown, if the Logger times are allowed to drift, the last message in the GTX packet will become the first value in the GTX packet (ie the

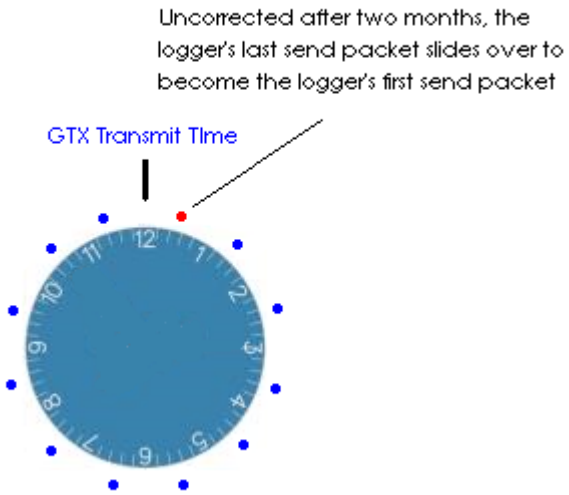
uAPP223 (v1.0) July 12,2005

Reading Time from the MICROCOM DESIGN GTX Satellite Transmitter into the Campbell Scientific CR1000 Data Logger

Author: Richard Schwarz

PRELIMINARY

Logger clock drift error is greater than 30 seconds) after 2 months.



This can be a real problem if you are expecting the data to be in a certain positions in the GTX transmission. Also if any time information is embedded in the logger send packets they will obviously be off id uncorrected.

In this case, to correct this ,we probably would be safe updating the logger time from the more accurate GTX time every few weeks. If the logger update time inaccuracy was greater than 2 minutes, of course the problem gets worse or can show up right away.

By syncing the Logger time to the more accurate GTX time the entire system operation becomes more accurate.

In fact if we recently updated the logger clock from the GTX clock, and we lost power/time on the GTX clock we could actually update he GTX clock from the Logger clock if done so within 4.5 hours of the last logger/GTX update. This could also be done if the logger was attached directly to t own source of a more accurate time (GPS, Time Standard, etc). It should be noted that the GTX will assume that any time set (even externally set) is a valid accurate time (as if it were a GPS time), so if you are doing timed transmissions, and unless you are absolutely sure the time being set is within the 0.1 seconds of actual UTC, you shouldn't set the GTX time externally.

For random GOES transmissions where transmissions are not tied to timeslots the timing is not as critical.

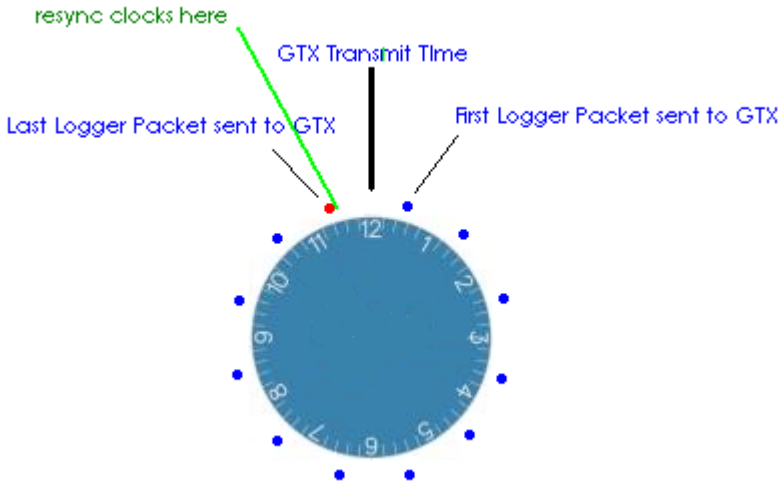
uAPP223 (v1.0) July 12,2005

Reading Time from the MICROCOM DESIGN GTX Satellite Transmitter into the Campbell Scientific CR1000 Data Logger

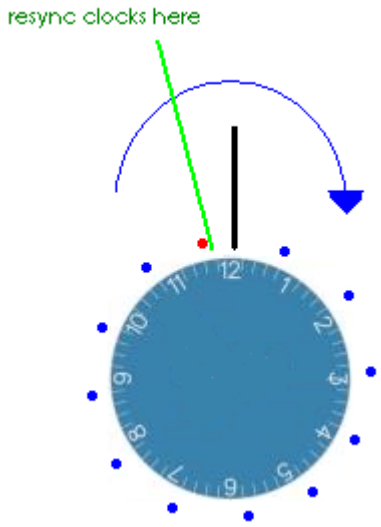
Author: Richard Schwarz

PRELIMINARY

OK, so we have decided to update the logger clock with the more accurate GTX clock. We are feeling very good about our system now. We have to be careful, however not to introduce even more problems. Lets examine our last example and lets go ahead and resync our clocks after sending our last packet to the GTX, since we are talking to the GTX at that time anyway.

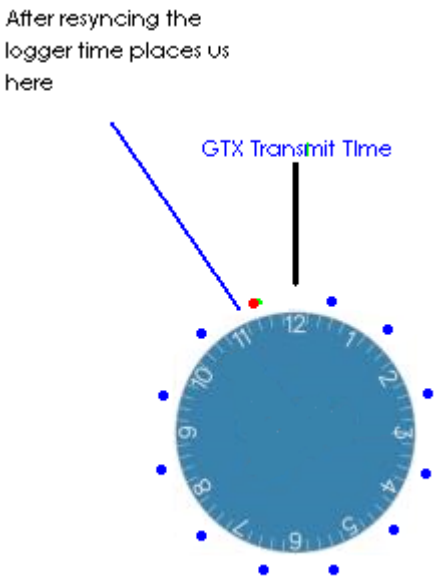


Ok this looks good at first glance, but lets remember that our clock has drifted and we are adjusting it. Further lets say we do the correction at a fairly slow update rate such that some amount of drift is allowed. A more accurate picture would be this one (with some drift introduced):



Now lets look at what the picture looks like after we readjust the clock, As can be

seen the readjusted time places us in a position where we actually resend our last packet again



The same situation could occur if we resync before sending the packet and the drift was in the opposite direction. The best solution in this instance is to split the difference and resync the clock between Logger packet sends, and insure that the maximum drift allowed (by your resync update rate) does not allow these packet send times to be missed.

GetGTXTime Subroutine Code:

The text of a subroutine is shown below, which can get the time from the GTX transmitter and set the current time of the CR1000. Variables should be declared at the top of the CrBasic file, and the COM ports being used also need to be opened before calling this routine. Also a DebugOnOff variable is also used which if set to a 1 will output debug messages to a com port which can be viewed on a standard terminal.

```
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
```

In our GetGTXTime routine It is assumed that the GTX time is already set. If it isn't the GTX will respond with the message "Not Set" . A conditional statement can be implemented to handle this case, when the GTX time is read. We can develop a subroutine which first checks the GTX time. The following routine does this:

```
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'S u b r o u t i n e''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'
''''''''''''''''''''''''''''''''''''''''
Sub CheckGtxTime

' Send CRs until getting Com4 > " >"
                SerialOut (Com4,CR,">",100,100)

'  Request Time from GTX
                SerialOut(Com4,"Time?","",0,100)
                SerialOut (Com4, CRLF,"",0,100)
'  Echo stuff to Debug Terminal
                If Debug = 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)
'__________________________________
                If (InString = "Not Set") then
                  GtxTimeSet =0
                Else
                 GtxTimeSet=1
                 endif
'_________________________________

endsub
```

Then, you could insure that the GetGtxTime isn't called unless the GtxTime is set:

```
CheckGtxTime  'check GTX time to see if it is set

If GtxTimeSet then  GetGTXTime  ' if it is then set to GTX time
```

Sample Application File

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. The GTX's serial port responds to commands and data.

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.

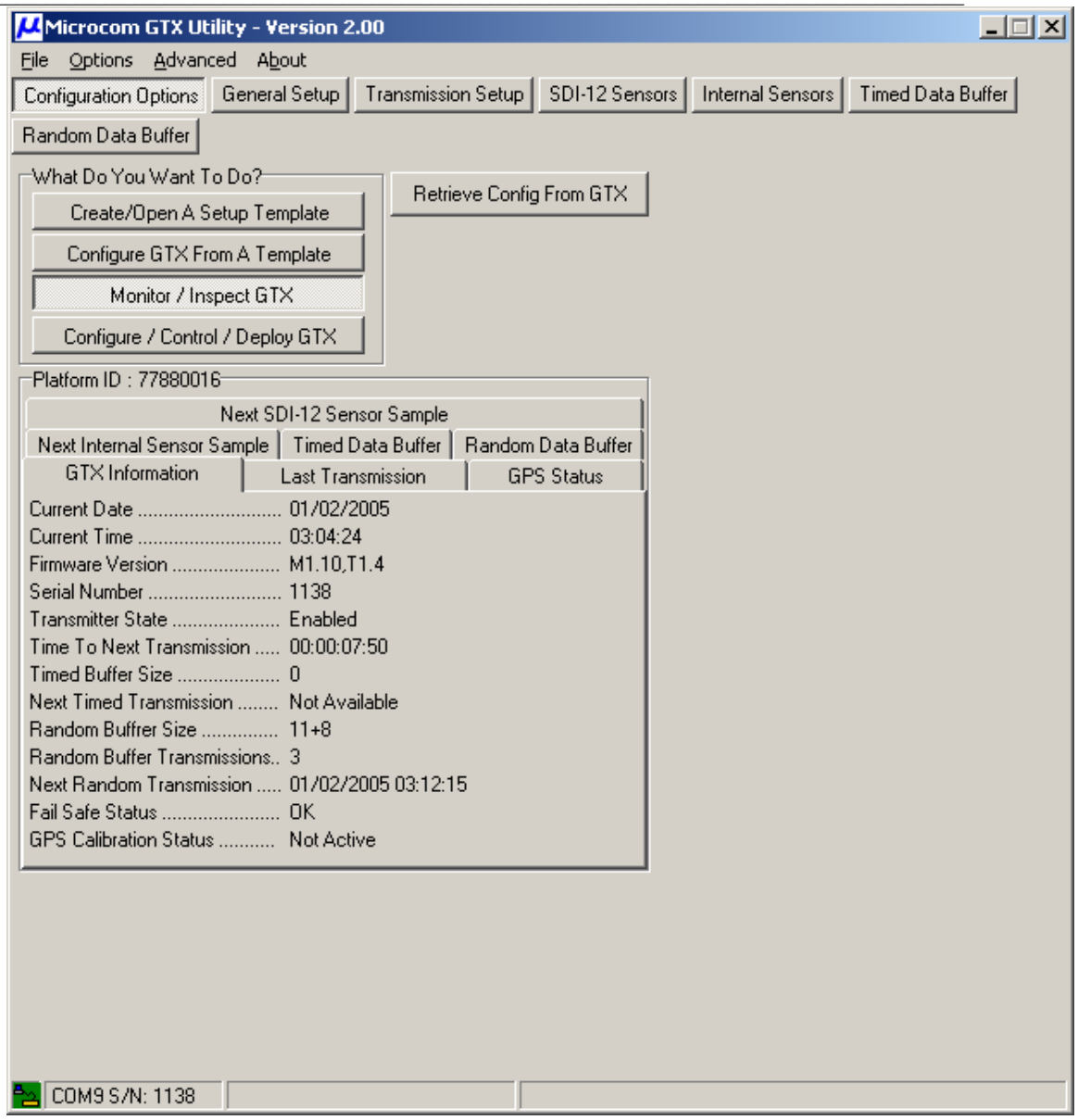
NOTE

Check uapp222 for CR1000 to GTX interfacing Instructions

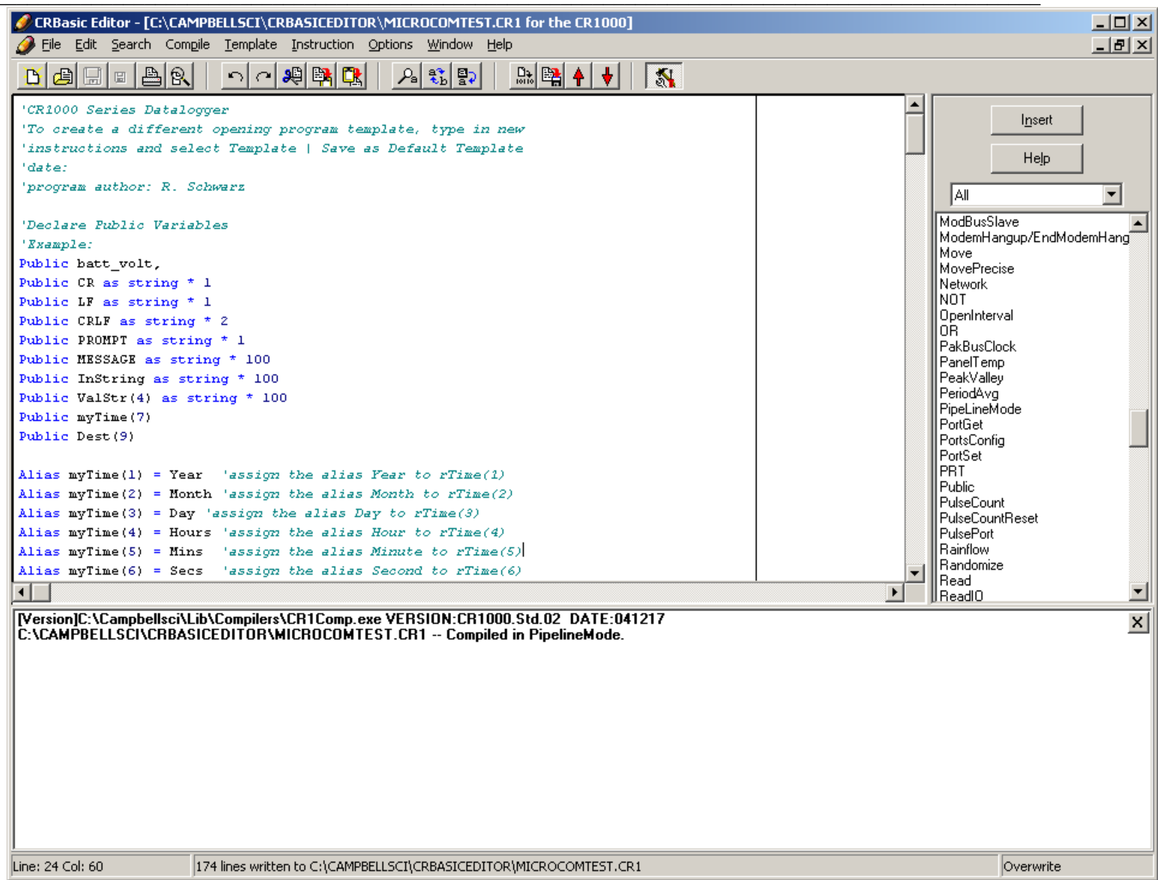
First we must set up the GTX. Using the Microcom GTX Utility software we can set the GTX up for Random transmission. The Microcom GTX utility runs on a PC and needs for a PC serial port to connect up to the GTX serial port. We do this (for now) by disconnecting the PC connection from the CR1000 instead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



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 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
'
```

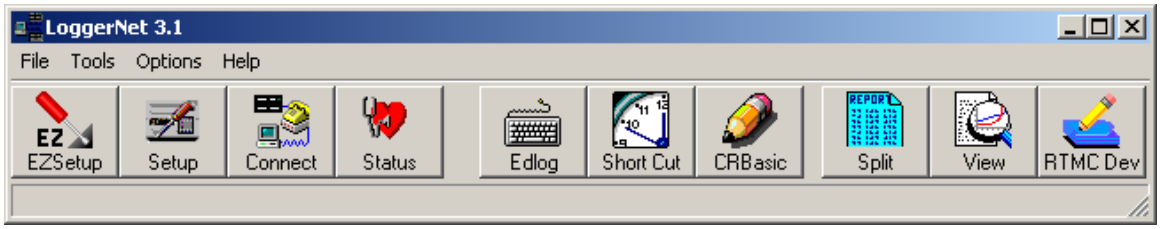
```
Units Batt_Volt=Volts
'
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
'Constant Declarations''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''
''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''''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

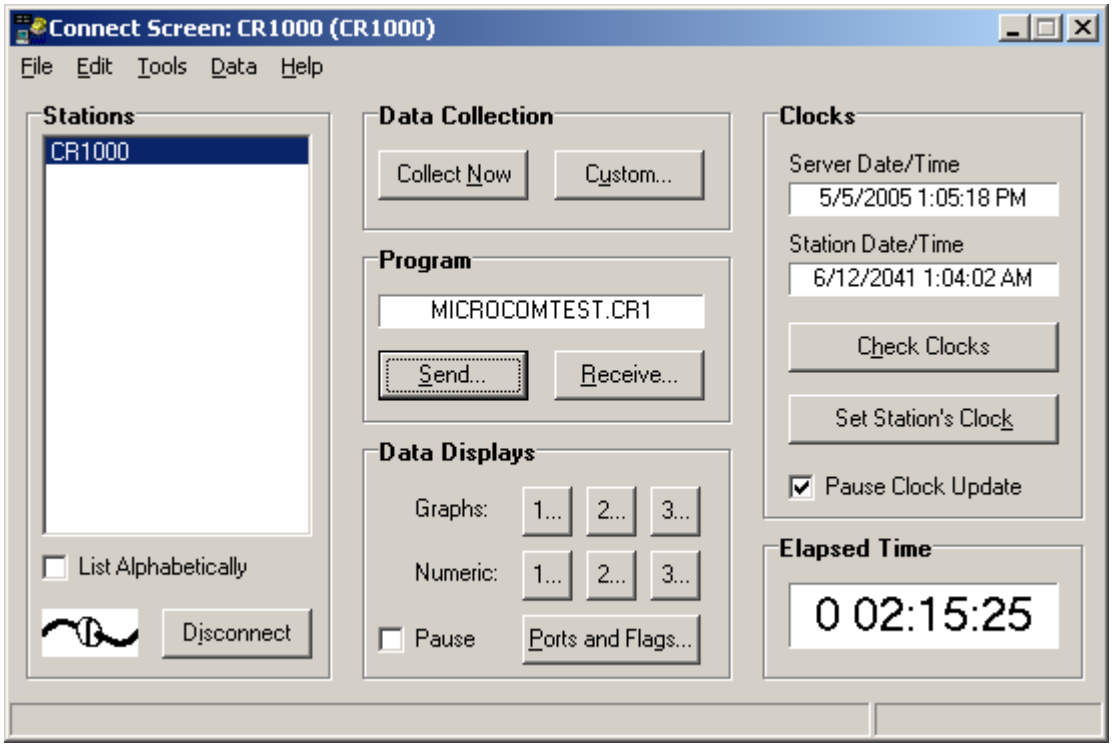
```
                    SerialOut(Com1,Mess2,"",0,100)
                    SerialOut (Com1,InString,"",0,100)
                EndIf

                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
```

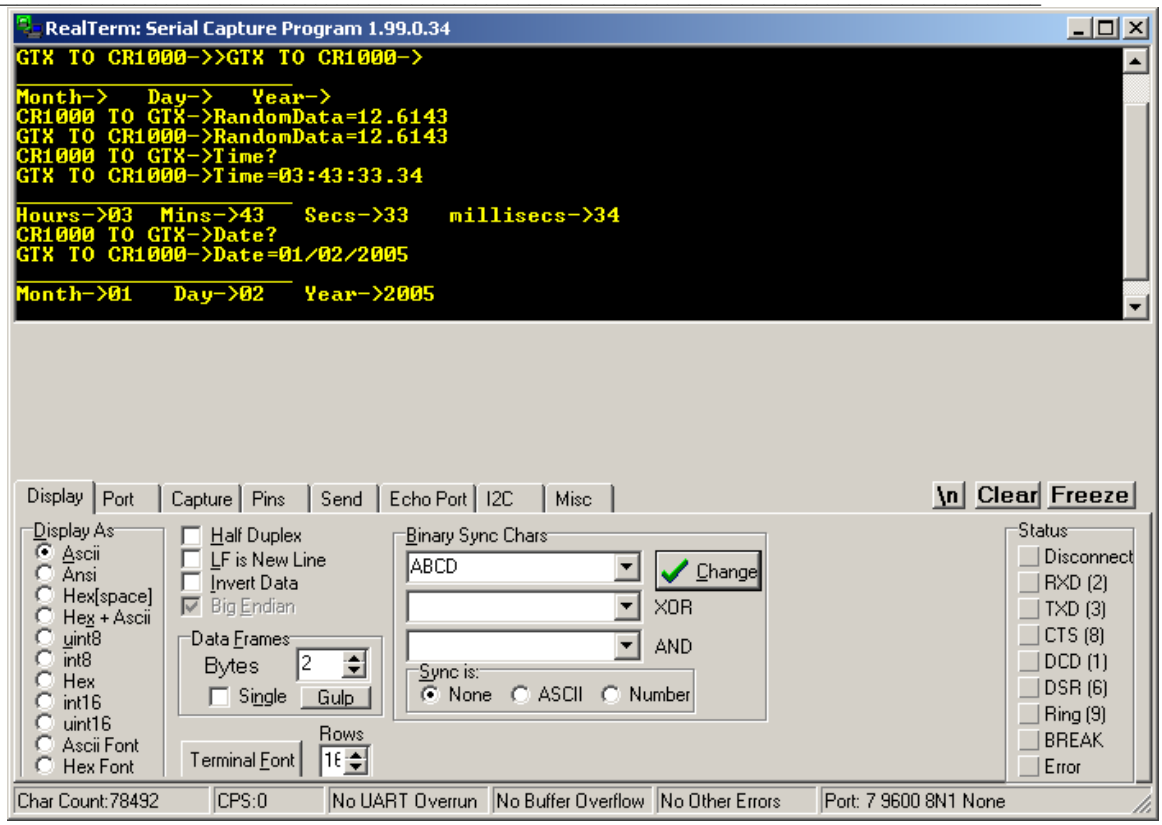
Once the program is written we need to download it to the CR1000 and allow the CR1000 to compile the program. For this we select the CONNECT button on the Logger Net software



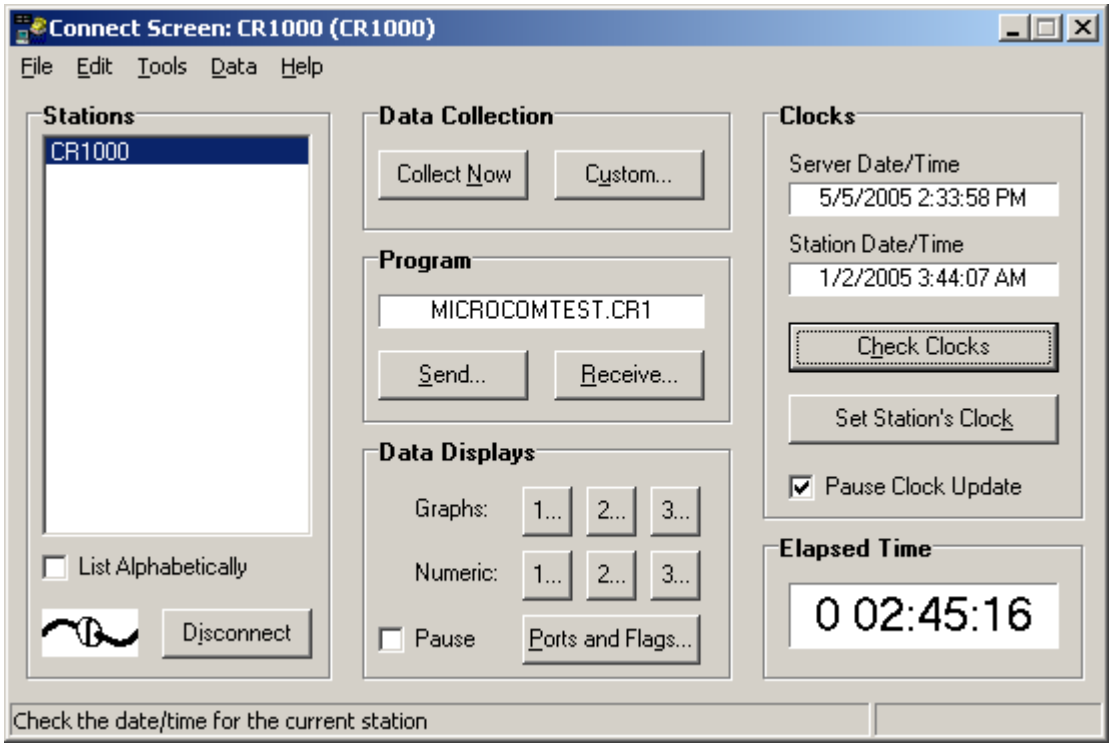


As can be seen on the connect screen we will also be able to read the CR1000 clock from this screen, to verify the time change our program will make.

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.



Conclusion

The MICROCOM Model GTX Satellite Transmitter has access to a more accurate clock via it's GPS capability than the Campbell CR1000 Data logger. A subroutine was developed which can be used in CRBASIC for the CR1000 to get the GTX time and set the CR1000 to the more accurate time.

Revision History

Date	Version	Revision
July 12, 2005	1.0	Initial Microcom Preliminary Release
September 20, 2005	1.01	Added a more detailed timing analysis example