



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D SATELLITE MODEM & XLITE 9210 DATA LOGGER)

1

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER APPLICATION NOTE



November 18, 2005

Prepared by:

Integrated Systems Division
Sutron Corporation
21300 Ridgetop Circle
Sterling, VA 20166

Copyright© 2005 Sutron Corporation

SUTRON

22400 DAVIS DRIVE
(703)406-2800

STERLING, VA 20164
(703)406-2801 FAX

WWW.SUTRON.COM
SALES@SUTRON.COM



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

TABLE OF CONTENTS

1. Overview 1

2. Equipment 1

3. Device Connections and Specifications.....2

3.1 Equipment Specifications..... 3

 3.1.1 SATELLITE MODEM FEATURES 3

4. Cellular Modem Activation.....4

5. IRIDIUM SATELLITE Modem Setup4

5.1 AT Commands 4

5.2 EXAMPLES..... 5

6. Testing and Troubleshooting.....5

APPENDIX6



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

1. OVERVIEW

Iridium Modem (Model A3LA-D) is a satellite modem designed to operate with the Iridium network. Similar to a standard landline modem, the A3LA-D can be controlled by any DTE (data terminal equipment) capable of sending standard AT commands via a serial port. A DTE can be a desktop computer, a laptop computer, a PDA, or even a micro-controller.

This modem has been tested for telemetry applications (remote monitoring and configuration) between a 9210 XLite/XPert Sutron dataloggers, a location provided with a telephonic line and e-mail for receiving data.

The following sections present:

- ▶ The required equipment and options
- ▶ Equipment cable connections and equipment specifications
- ▶ Basic instructions to set up the modem using AT commands
- ▶ Basic code used to format data for Short Burst Data (SBD) transmissions

The instructions for programming the 9210 XLITE/XPERT to use the IRIDIUM SATELLITE modem are achieved using BASIC program. A copy of the manual for the 9210 XLite/XPert is available at:

www.sutron.com/DownloadsUpdates/Xpert%20User%20Manual.pdf

Important Note: Most of the manufacturers use the AT commands to set up their devices, for this reason the commands and steps of configuration the present IRIDIUM SATELLITE modem are valid for a variety of typical modems. However, this application note is completely accurate for the *NAL Research* Model A3LA-D Iridium Modem.

2. EQUIPMENT

For a telemetric application (for monitoring and remotely configure the datalogger) it is required the following equipment:

	Part Number	Description
1	9210-0000-2A	9210 XLITE Data Logger
2	Iridium Modem (Model A3LA-D)	IRIDIUM SATELLITE modem with power and communication cables
3	Rod Antenna Model SAF5350-C	External Rod Antenna with TNC Female Connector (Option: TNC, SMA, N, N Bulkhead, MCX, MMCX or Cable)
4	DC-DC Converter Model SYN-DC-936	Power supply



3. DEVICE CONNECTIONS AND SPECIFICATIONS

The configuration requires attaching the IRIDIUM SATELLITE modem to COM3 of the Sutron datalogger. A serial cable (DB25 to DB9) is provided from the manufacturer of the modem DC/DC converter. The modem is attached to the *Rod* antenna. When the connections are done, the modem needs to be powered up (the power for the IRIDIUM SATELLITE modem can be provided from the 12VDC battery or 9210 since the modem uses 5V to 32V).



Figure 1 9210 connected through the IRIDIUM SATELLITE Modem.



3.1 EQUIPMENT SPECIFICATIONS

3.1.1 SATELLITE MODEM FEATURES & SPECIFICATIONS

MECHANICAL	
Dimensions	7.73" L x 3.25" W x 1.54" D (19.6 cm x 8.3 cm x 4.0 cm)
Weight	~1.44 pounds (659 g)
I/O Interface	25-Pin D-Sub, SIM Reader
Antenna	TNC Female
Cooling	Convection
Enclosure	Aluminum/EMI shielding
ELECTRICAL	
Input Voltage Range	4.0VDC to 4.8VDC
Input Nominal Voltage	4.4VDC
Input Ripple Voltage	40mV pp
Avg. Standby Current	130mA @4.4VDC
Avg. Transmit Current	1.0A @ 4.4VDC
Avg. Data Call Current	500mA @ 4.4VDC
Peak Power-up Current	~2.2A @ 4.4VDC
Operating Frequency:	1616 to 1626.5 MHz
Duplexing Method	Time Division Duplex
Multiplexing Method	TDMA/FDMA
Link Margin	13.1 Db
DATA I/O	
Dial-up Data, RUDICS	2.4 Kbits/sec
Direct Internet	~10.0 Kbits/sec
Short-Burst Data	1960 Bytes/message
Short Messaging	160 characters
Hardware Interface	RS232
Software Interface	AT Commands
ENVIRONMENTAL	
Operating Temperature	-40°F to +140°F (-20°C to +60°C)
Operating Humidity	< 75% RH
Storage Temperature	-40°F to +185°F (-40°C to +85°C)
Storage Humidity	< 93% RH



4. CELLULAR MODEM ACTIVATION

NAL Research is a Tier I Iridium Value-Added Reseller (VAR) providing airtime for data only services including dial-up data, direct Internet connection, router-based unrestricted digital internetworking connectivity solution (RUDICS), short-burst data (SBD) and short-messaging service (SMS).

For most application a DATA SERVICE PLAN needs to be added to the IRIDIUM SATELLITE modem. The providers usually offer voice services to the IRIDIUM SATELLITE subscribers but data is available upon request as an extra feature. For these reason, it is necessary to set up the device as a DATA modem (although it is able to have voice communication also).

A SIM card will be provided with each device, which it holds the information of the IRIDIUM SATELLITE modem and its phone number. The SIM card is the guarantee of a service, which is going to allow the device to enter into the IRIDIUM SATELLITE network. The IRIDIUM SATELLITE provider will provides the phone number, band, and coverage.

NOTE: When calling the modem it will be charged as an international call.

Before start using the modem, although it has been activated by a local provider, it needs to be configured and tested. For this reason, the modem should be powered up and hooked up to the antenna.

5. IRIDIUM SATELLITE MODEM SETUP

Since most if not all on the configuration of the modem is done by the BAASIC program, all that is required to setup the modem is to configure the modem for the amount of ring before it answer and to turn the echo of the modem.

5.1 AT Commands

The IRIDIUM SATELLITE modem accepts a wide variety of AT commands to change its operation, however, only a few of them are necessary to put it to work with the 9210 Sutron datalogger. Among others, the essential AT commands are:

COMMAND	VALUE	DESCRIPTION
ATEn	0 or 1	Characters are not echoed to the DTE
ATS0	1	Enable auto-answer after three rings.

After issuing each AT command, it is suggested to save it using the command AT&Wn. Additionally, to check out the configuration or for maintenance purposes the command AT+V will display the basic setting of the device.

To issue the AT commands or to start talking with the modem, a serial link needs to be made between the PC and the IRIDIUM SATELLITE modem. To do that, first connect the serial cable DB25 to DB9 DC/DC Converter between the PC and the Iridium modem

Then, create a new connection using the *HYPERTERMINAL* of Windows. Usually, it is located in *START-----ALL PROGRAMS-----ACCESSORIES-----COMMUNICATIONS*.



To see the commands we are issuing make sure that the echo character is enabled locally. To do that, inside the *HyperTerminal session* go to *FILE/Properties/Settings*, and then select *ASCII Setup*. For *ASCII Sending* it is necessary to check the box beside: ECHO TYPED CHARACTERS LOCALLY. Once you are done configuring the modem turn the echo off in the modem using the command ATE0

5.2 EXAMPLES

The following are examples of the commands issued to the Iridium modem using HyperTerminal. Some of them are only information commands that don't affect or change the configuration. Comments are placed in red letters.

- ▶ Checking the operation device settings:
AT&V (AT commands and their actual value)
Q:0 V:1 S0:000 S2:043 S3:013 S4:010 S5:008
+CR:0 +CRC:0 +CMEE:0 +CBST:0,0,1
+SPEAKER:0 +ECHO:0,1 &C:1 &D:2 %C:0
+IPR:9600 +ICF:3,4 +IFC:2,2
OK
- ▶ To check auto-answer mode setting:
ATS0?
001 (Auto-answer after one ring)
OK

6. TESTING AND TROUBLESHOOTING

After the entire configuration has been done, some problems can be found in the link. Hence, it is recommendable to test the IRIDIUM SATELLITE modem with the *HyperTerminal* before it is connected to the datalogger or DTE.

To test if the modem is answering correctly in a DATA link, it is necessary to dial the IRIDIUM SATELLITE modem number and look at the answer in the screen. Since the modem has been configured as an auto-answer device, automatically will display the mode of the incoming call. An example of a correct response from the modem to a data call is:

```
CARRIER 4800  
PROTOCOL: LAP-M  
COMPRESSION: NONE  
CONNECT: 4800
```



APPENDIX

AT Commands Used

En - Echo

Echo command characters.

0 Characters are not echoed to the DTE.

1 Characters are echoed to the DTE (default).

S0=n - Auto-Answer (Initial implementation; revised in Phase III)

Auto-answer. This causes the ISU to auto-answer the incoming data call.

0 Disable auto-answer.

n>0 Enable auto-answer.

&V - View Active and Stored Configuration

View the current active configuration and stored profiles.

&Wn - Store Active Configuration

Store the active profile in non-volatile memory. This is used to store user configurations for later use.

0 Store current (active) configuration as profile 0.

1 Store current (active) configuration as profile 1.

BASIC PROGRAMS

Iridium_sbd.bas

Const True = -1

Const False = 0

Const LoggingIn=2

Const LoggedIn=4

Const LF = Chr(10) ' Line feed character

Const CR = Chr(13) ' Carriage return character

Declare Function ReadLog(LogName, Sensor, TimeStamp)

Public Declare Sub FlushPort(Handle)

Public Declare Function OpenPort(ComPort, RejectWhen)

Public Declare Sub ClosePort(Handle)

Public Declare Function ReadPort(Handle, BytesToRead, BytesRead, BytesRemain, TimeoutSec)

Public Declare Sub WritePort(Handle, Data)

Public Declare Function HayesCommand(Handle, Command, TimeoutSec)

Public Declare Sub SetDTRPort(Handle)

Public Declare Sub ClrDTRPort(Handle)



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

```
Public Declare Sub SetRTSPort(Handle)
Public Declare Sub ClrRTSPort(Handle)
Public Declare Sub SetBreakPort(Handle)
Public Declare Sub ClrBreakPort(Handle)
Public Declare Function CDPort(Handle)
```

```
' Variables used to store results of ReadLog function.
```

```
RLData = 0.0
```

```
RLQuality = "U"
```

```
RLUnits = ""
```

```
Public Sub Sched_ModemInit
```

```
' the iridium modem "looses" its baud rate whenever it is powered up
' or when DTR is dropped. The solution is to send an AT command on power up
' and to not use DTR. This routine sends the AT command and must be scheduled
' to run a few seconds after the modem is powered up.
```

```
ModemInit = True
```

```
Handle = OpenPort("COM3:", LoggingIn Or LoggedIn)
```

```
' Handle = OpenPort("COM3:", 0) ' use this to open even if in use
```

```
If Handle Then
```

```
    Call SetDTRPort(Handle) 'note: DTR should already be high
```

```
    Sleep(1) ' Allow the modem time to wake up
```

```
    Txt = HayesCommand(Handle, "AT", 2)
```

```
    If Txt="" Then
```

```
        ErrorMsg "Hayes Command AT failed"
```

```
    Else
```

```
        StatusMsg "Hayes Command AT returned:" + txt
```

```
    End If
```

```
' Call ClrDTRPort(Handle)
```

```
    Call ClosePort(Handle)
```

```
Else
```

```
    ModemInit = False
```

```
    ErrorMsg "ModemInit could not access the Modem"
```

```
End If
```

```
End Sub
```

```
'Public Declare Function FormatData
```

```
Public Sub FormatData(DataSet1,DataSet2)
```

```
,
```

```
' Sensors to tx:
```

```
' 15minute interval- S1, S2, B
```

```
'make tx look like this
```



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

```
'mm/dd/yyyy,hh:mm:ss,S1,S2,B
'
'INITIALIZE local variables
StatusMsg "Starting format routine"
Const LogName = "SSP.LOG" 'where to get the data from
Const MinTx = 96 '15min values to tx ALWAYS MAKE THIS AN EVEN NUMBER
Const DataInt = 900
TimeNow = now ' What time are we starting
'

'set up sensors array
'Sensors(2) = "B"
'Sensors(1) = "S2"
'Sensors(0) = "S1"
'

'Initialize array at 0 to hold data for transmission,start with :sensor
NumSensors = Ubound(Sensors)
'

' Loop to get data from log
'Get all defined sensors and build their string
'Get recent timestamp based on sensor interval and add time offset and interval to
sensor message
TSens = TimeNow - (TimeNow Mod DataInt)

DataToTx = ""
For T = 1 to (MinTx/2)
'Get the number of values specified (recent data first),
' Add good data to sensor tx string, place an M in bad or missing data
'mm/dd/yyyy,hh:mm:ss,S1,S2,B
DateStr = Month(Tsens) & "/" & Day(Tsens) & "/" & Year(Tsens)
TimeStr = Format("%02d",Hour(Tsens)) & ":" & Format("%02d",Minute(Tsens)) & ":" &
Format("%02d",Second(Tsens))
DateTimeStr = DateStr & "," & TimeStr

For I = 0 To NumSensors
If (ReadLog(LogName, Sensors(I), TSens) = 1) Then
Select Case I
Case 0
DataToTx = DataToTx + DateTimeStr & "," + Format("%.2f",RLData) ' format 2
rights digits
Case 1
DataToTx = DataToTx + DateTimeStr & "," + Format("%.2f",RLData) ' format 2
rights digits
```



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

Case 2

DataToTx = DataToTx + DateTimeStr & "," + Format("%.1f",RLData) ' format 1
rights digits

End Select

Else

DataToTx = DataToTx + DateTimeStr & ",M"

End If

DataToTx = DataToTx + CR+LF

TSens = TSens - DataInt

Next I

Next T

'Grab First Set of data

DataSet1 = DataToTX

'Loop to get data from log

'Get all defined sensors and build their string

,

'Get recent timestamp based on sensor interval and add time offset and interval to
sensor message

DataToTx = ""

For T = ((MinTx/2) + 1) to MinTx

'Get the number of values specified (recent data first),

' Add good data to sensor tx string, place an M in bad or missing data

'mm/dd/yyyy,hh:mm:ss,S1,S2,B

DateStr = Month(Tsens) & "/" & Day(Tsens) & "/" & Year(Tsens)

TimeStr = Format("%02d",Hour(Tsens)) & ":" & Format("%02d",Minute(Tsens)) & ":" &
Format("%02d",Second(Tsens))

DateTimeStr = DateStr & "," & TimeStr

For I = 0 To NumSensors

If (ReadLog(LogName, Sensors(I), TSens) = 1) Then

Select Case I

Case 0



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

```
DataToTx = DataToTx + DateTimeStr & "," + Format("%.2f",RLData) ' format 2
rights digits
```

```
Case 1
```

```
DataToTx = DataToTx + DateTimeStr & "," + Format("%.2f",RLData) ' format 2
rights digits
```

```
Case 2
```

```
DataToTx = DataToTx + DateTimeStr & "," + Format("%.1f",RLData) ' format 1
rights digits
```

```
End Select
```

```
Else
```

```
DataToTx = DataToTx + DateTimeStr & ",M"
```

```
End If
```

```
DataToTx = DataToTx + CR+LF
```

```
TSens = TSens - DataInt
```

```
Next I
```

```
Next T
```

```
'Grab Second Set of data
```

```
DataSet2 = DataToTX
```

```
Statusmsg DataSet1 & DataSet2
```

```
End Sub
```

```
,
```

```
,
```

```
*****
```

```
,
```

```
Function ReadLog(LogName, Sensor, TimeStamp)
```

```
' LogName, Sensor, and TimeStamp are inputs. RLData, RLQuality, and RLUnits
```

```
' are global variables that receive this function's outputs.
```

```
' If Sensor at Timestamp is found, 1 is returned. Otherwise, 0.
```

```
ReadLog = 0
```

```
Type = 0
```

```
TimeFound = 0
```

```
SensorFound = ""
```

```
FileNum = FreeFile
```

```
Open LogName for Log as FileNum
```



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

```
Seek FileNum, TimeStamp
If Not Eof(FileNum) Then
    Input FileNum, Type, TimeFound, SensorFound, RLData, RLQuality, RLUnits
    Do While TimeFound = TimeStamp And Not EOF(FileNum)
        If SensorFound = Sensor Then
            If RLQuality = "G" Then
                ReadLog = 1
            End If
        Exit Do
    Else
        ' Log may contain multiple entries for this time-slot so keep looking.
        ' Original Seek finds last entry for specified time, so move to previous.
        Seek FileNum, Next
        Input FileNum, Type, TimeFound, SensorFound, RLData, RLQuality, RLUnits
    End If
End Loop
End If
Close FileNum
End Function
*****
Public Sub Sched_SendSBD
    Const RegTimeout = 300
    Const SendTimeout = 300

    DataSet1 = ""
    DataSet2 = ""
    '
    Handle = OpenPort("COM3:", LoggingIn Or LoggedIn)
    ' Handle = OpenPort("COM3:", 0) 'use this to openport even if in use.

    If Handle Then
        Call SetDTRPort(Handle)
        Sleep(5) ' Allow the modem time to wake up
```



APPLICATION NOTE

12

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

```
' wait for the modem to be registered
StartTime = Time
ModemRegistered = false
Do
  StatusMsg "AT+CREG?"
  txt = HayesCommand(Handle, "AT+CREG?", 10)
  StatusMsg "CREG=" &txt
  If Right(txt,5) = "001OK" then
    ModemRegistered = true
  End If
  If CDPort(Handle) <> 0 then
    StatusMsg "CD active on modem ... abort"
    goto errorhandler
  End If
Loop Until (ModemRegistered OR ((Time - StartTime)> RegTimeout))
If ((Time - StartTime) > RegTimeout) then
  ErrorMessage "Timeout waiting for Registration -- CREG: " &txt
  goto ErrorHandler
End If
  'clear SBD Message Buffer
StatusMsg "AT+SBDD0"
txt = HayesCommand(Handle, "AT+SBDD0", 2)
If txt="" OR txt="1" Then
  ErrorMessage "Command AT+SBDD failed " &txt
  goto ErrorHandler
End If
If 0 = 1 Then
  'send some text to the message buffer
  StatusMsg "AT+SDBWT"
  txt = HayesCommand(Handle, "AT+SBDWT=SBDWT Text Test", 2)
  If txt="" OR txt <>"OK" Then
    ErrorMessage "Command AT+SBDWT failed " &txt
    goto ErrorHandler
```



```
End If
Else
Call FormatData(DataSet1,DataSet2)
BinString = DataSet1
LenData = Format("%d", Len(BinString))
Result = ComputeCRC(0,0,BinString) Mod 65536'
StatusMsg "CRC " &(Result>>8) &" " &(Result Mod 256)
    BinString = BinString & Chr(Result>>8) & Chr(Result Mod 256)
Cmd = "AT+SBDWB="&LenData
Txt = HayesCommand(Handle, Cmd, 2)
If Left(Txt, 5)="READY" then
    Txt = HayesCommand(Handle, BinString, 2)
    StatusMsg "SBDWB:" &txt
    If Left(Txt,1) <> "0" then
        ErrorMsg "Command SBDWB failed " &Txt
        goto ErrorHandler
    End If
End If
End If
End If
'Initiate SBD Session to send the data
StartTime = Time
DataSent = false
Do
    statusmsg "AT+SBDI"
    txt = HayesCommand(Handle, "AT+SBDI", 15)
    StatusMsg "SBDI=" &txt
    If Left(txt, 8) = "+SBDI: 1" then
        DataSent = true
    End If
    If CDPort(Handle) <> 0 then
        StatusMsg "CD active on modem ... abort"
        goto errorhandler
    End If
```



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

```
Loop Until (DataSent OR ((Time - StartTime) > SendTimeout))
If ((Time - StartTime) > SendTimeout) then
    ErrorMessage "SBDI timeout " &txt
    goto ErrorHandler
End If

    'StatusMsg "first done"
    *****

    BinString = DataSet2
LenData = Format("%d", Len(BinString))
Result = ComputeCRC(0,0,BinString) Mod 65536'
StatusMsg "CRC " &(Result>>8) &" " &(Result Mod 256)
    BinString = BinString & Chr(Result>>8) & Chr(Result Mod 256)
Cmd = "AT+SBDWB="&LenData
Txt = HayesCommand(Handle, Cmd, 2)
If Left(Txt, 5)="READY" then
    Txt = HayesCommand(Handle, BinString, 2)
    StatusMsg "SBDWB: " &txt
    If Left(Txt,1) <> "0" then
        ErrorMessage "Command SBDWB failed " &Txt
        goto ErrorHandler
    End If
End If

'Initiate SBD Session to send the data
StartTime = Time
DataSent = false
Do
    statusmsg "AT+SBDI"
    txt = HayesCommand(Handle, "AT+SBDI", 15)
    StatusMsg "SBDI=" &txt
    If Left(txt, 8) = "+SBDI: 1" then
        DataSent = true
    End If
    If CDPort(Handle) <> 0 then
```



```
StatusMsg "CD active on modem ... abort"  
Goto errorHandler  
End If  
Loop until (DataSent OR ((Time - StartTime)> SendTimeout))  
If ((Time - StartTime) > SendTimeout) then  
    ErrorMsg "SBDI timeout " &txt  
    goto ErrorHandler  
End If  
    'StatusMsg "Second done"  
ErrorHandler:  
' Call ClrDTRPort(Handle)  
    Call ClosePort(Handle)  
Else  
    ErrorMsg "Could not access the Iridium Modem"  
End If  
End Sub
```

SERIALLIB.BAS

REM SerialLib.bas (C) 2005 Sutron Corporation

REM

REM This is a library of functions that may be used to communicate with
REM a serial port which is under the control of the Coms Manager and/or Remote.
REM For ports which are not in use, the standard file I/O api built-in to Xpert Basic
REM should be used.

REM

REM The library uses TCP/IP socket communications to take control of the port from
REM Remote. Strings may be read or written, control lines can be toggled, and
REM there is a command to issue Hayes commands and get back the result.

REM

REM This code may be freely modified or adapted as long as it is used on Sutron
REM equipment and this notice is not removed.

REM Constants that may be reused by other Basic Programs using this Library



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

Const True = -1
Const False = 0
Const LoggingIn=2
Const LoggedIn=4

REM Subroutines and Function declarations that may be reused by other Basic Programs using this Library

Public Declare Function OpenPort(ComPort, RejectWhen)

Public Declare Sub ClosePort(Handle)

Public Declare Sub FlushPort(Handle)

Public Declare Function ReadPort(Handle, BytesToRead, BytesRead, BytesRemain, TimeoutSec)

Public Declare Sub WritePort(Handle, Data)

Public Declare Function HayesCommand(Handle, Command, TimeoutSec)

Public Declare Sub SetDTRPort(Handle)

Public Declare Sub ClrDTRPort(Handle)

Public Declare Sub SetRTSPort(Handle)

Public Declare Sub ClrRTSPort(Handle)

Public Declare Sub SetBreakPort(Handle)

Public Declare Sub ClrBreakPort(Handle)

Public Declare Function CDPort(Handle)

Public Declare Function DSRPort(Handle)

Public Declare Function CTSPort(Handle)

' Documentation:

,

' There is a sample called SerialLibTest at the end of this file that may be
' commented in, and used to test the library. There is also a separate program
' called SerialTest.Bas that demonstrates how these functions can be called from
' a separate program.

,

' Do not expect fast performance when using this library. A lot of indirection is
involved

' and reading data specifically requires a lot of overhead. Sending data to a port will



' be efficient if the data can be built up in to strings.

,

' OpenPort: Call this to gain access to a port. 0 is returned if access could not be obtained, otherwise an I/O handle is returned. The RejectWhen parameter determines whether the port should be grabbed only when the the port is disconnected (6), only when someone isn't logged in (4), or without restriction (0).

,

' ClosePort: Call this when you are done with the port.

,

' WritePort: Writes bytes to a port.

,

' ReadPort: Reads bytes from a port. BytesToRead specifies how many bytes are desired.

' BytesRead indicates how many were read, BytesRemain indicates how many can be read, and TimeoutSec is how long to wait.

,

' HayesCommand: Appends a CR to a command and sends it to a port. It then waits up to

' TimeoutSec for a reply and returns any data. The returned data is trimmed of control-codes. No retries are performed.

,

' SetDtrPort: Enables DTR on the port

,

' ClrDtrPort: Disables DTR on the port

,

' SetRtsPort: Enables RTS on the port

,

' ClrRtsPort: Disables RTS on the port

,

' SetBreakPort: Creates a break condition on the port.

,

' ClrBreakPort: Clears a break condition on the port

,



APPLICATION NOTE

18

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

'CDPort: Returns the status of CD, any incoming data is lost.

,

'DSRPort: Returns the status of DSR, any incoming data is lost.

,

'CTSPort: Returns the status of CTS, any incoming data is lost.

REM Private constants

Const OpAck = 0

Const OpEOT = 16

Const OpComOpen = 82

Const OpComClose = 83

Const OpComLock = 84

Const OpComUnLock = 85

Const OpComData = 86

Const OpComControl = 90

Const OpComGetStatus = 91

Const OpComStatus = 92

Const OpComCapture = 93

Const OpComGetOptions = 97

Const OpComOptions = 98

Const OpComSetCommState = 102

Const ThreeNulls = Chr(0)&Chr(0)&Chr(0)

REM Private variables

Dim DataBuffer

Sub SendOp(F, Opcode, Data)

L = Len(Data)

Print F, Chr(Opcode); Chr(L); Chr(L >> 8); Data;

End Sub

Function GetOp(F, Opcode, L, Data)

L = 0



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

```
Opcode = OpEOT
LL = ""
LH = ""
OP = ""
GetOp = False
If (ReadB(F, OP, 1) = 1) Then
  Opcode = Asc(OP)
  If (ReadB(F, LL, 1) = 1) Then
    If (ReadB(F, LH, 1) = 1) Then
      L = Asc(LH)*256 + Asc(LL)
      If (ReadB(F, Data, L) = L) Then
        GetOp = True
      End If
    End If
  End If
End If
End Function
```

```
Function Send(F, Opcode, Data)
  Call SendOp(F, Opcode, Data)
  Send = False
  O = 0
  D = ""
  L = 0
  If GetOp(F, O, L, D) Then
    If O=OpAck And Opcode=Asc(D) Then
      Send = True
    End If
  End If
End Function
```

```
' Return true if someone is logged in or in the process of logging in to the port
Function IsLoggedIn(F, RejectWhen)
```



```
Call SendOp(F, OpComGetOptions, "")
IsLoggedIn = False
O = 0
D = ""
L = 0
If GetOp(F, O, L, D) Then
  If (O=OpComOptions) And (Asc(D) And RejectWhen) Then
    IsLoggedIn = True
  End If
End If
End Function

' Return true if the port is not inuse and we can get exclusive access
Function MakeConnect(F, RejectWhen)
  MakeConnect = False
  If Not IsLoggedIn(F, RejectWhen) Then
    If Send(F, OpComLock, "") Then
      MakeConnect = True
    End If
  End If
End Function

' Erase the input buffer
Public Sub FlushPort(Handle)
  DataBuffer(Handle) = ""
  FlushInput Handle
End Sub

' Open's a a COM port in the specified mode
' RejectWhen is used to prevent accessing a port which is already inuse and
' can take the value: 0, LoggingIn, LoggedIn, or Logging Or LoggedIn
Public Function OpenPort(ComPort, RejectWhen)
  OpenPort = 0
```



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

```
PortIndex = Asc(Mid(ComPort, 4, 1)) - Asc("0")
Handle = FreeFile
' 52733 is the socket server in Remote
Open "localhost:52733" As Handle
SetTimeout Handle, 5.0
If Send(Handle, OpComOpen, Chr(PortIndex)) Then
    StartTick = GetTickCount
    ' Try for up to 5 seconds to get exclusive access to the port
    Do While Not MakeConnect(Handle, RejectWhen)
        Sleep(0.1)
        If (Now - StartTick) >= 5000 Then
            Close Handle
            Exit Function
        End If
    End Loop
    Call SendOp(Handle, OpComCapture, Chr(1))
    OpenPort = Handle
    Call FlushPort(Handle)
Else
    Close Handle
End If
End Function

Public Sub ClosePort(Handle)
    If Handle <> 0 Then Close Handle
End Sub

Public Function ReadPort(Handle, BytesToRead, BytesRead, BytesRemain, TimeoutSec)
    Result = ""
    O = 0
    D = ""
    L = 0
    StartTick = GetTickCount
```



```
TimeoutMS = TimeoutSec * 1000
Do While Len(DataBuffer(Handle)) < BytesToRead
  If Loc(Handle) >= 3 Then
    If GetOp(Handle, O, L, D) Then
      If O=OpComData Then
        DataBuffer(Handle) = DataBuffer(Handle) + D
      End If
    Else
      Exit Do
    End If
  ElseIf (GetTickCount-StartTick) > TimeoutMS Then
    Exit Do
  Else
    Sleep(1)
  End If
End Loop
BytesRead = Len(DataBuffer(Handle))
If BytesRead > BytesToRead Then
  BytesRead = BytesToRead
End If
ReadPort = Left(DataBuffer(Handle), BytesRead)
DataBuffer(Handle) = Mid(DataBuffer(Handle), BytesRead+1)
BytesRemain = Len(DataBuffer(Handle))
End Function

Public Sub WritePort(Handle, Data)
  Call SendOp(Handle, OpComData, Data)
End Sub

Public Sub SetDTRPort(Handle)
  Call SendOp(Handle, OpComControl, Chr(5)+ThreeNulls)
End Sub
```



```
Public Sub ClrDTRPort(Handle)
    Call SendOp(Handle, OpComControl, Chr(6)+ThreeNulls)
End Sub
```

```
Public Sub SetRTSPort(Handle)
    Call SendOp(Handle, OpComControl, Chr(3)+ThreeNulls)
End Sub
```

```
Public Sub ClrRTSPort(Handle)
    Call SendOp(Handle, OpComControl, Chr(4)+ThreeNulls)
End Sub
```

```
Public Sub SetBreakPort(Handle)
    Call SendOp(Handle, OpComControl, Chr(8)+ThreeNulls)
End Sub
```

```
Public Sub ClrBreakPort(Handle)
    Call SendOp(Handle, OpComControl, Chr(9)+ThreeNulls)
End Sub
```

```
Function GetStatus(F)
    Result = 0
    Call SendOp(F, OpComGetStatus, "")
    IsLoggedIn = False
    O = 0
    D = ""
    L = 0
    Do While GetOp(F, O, L, D)
        If O=OpComData Then ' Process a data packet that might slip in
            DataBuffer(F) = DataBuffer(F) + D
        ElseIf O=OpComStatus Then
            Result = Asc(D)
        Exit Do
```



```
End If
End Loop
GetStatus = Result
End Function
```

```
Public Function CDPort(Handle)
Const MS_RLSD_ON = &h80
CDPort = (GetStatus(Handle) And MS_RLSD_ON) <> 0
End Function
```

```
Public Function DSRPort(Handle)
Const MS_DSR_ON = &h20
DSRPort = (GetStatus(Handle) And MS_DSR_ON) <> 0
End Function
```

```
Public Function CTSPort(Handle)
Const MS_CTS_ON = &h10
CTSPort = (GetStatus(Handle) And MS_CTS_ON) <> 0
End Function
```

' Eliminate control characters from a string

```
Function Trim(S)
Result = ""
For i = 1 To Len(S)
Ch = Mid(S, i, 1)
If (" " <= Ch) And (Ch <= "~") Then
Result = Result + Ch
End If
Next i
Trim = Result
End Function
```

' Sends a Hayes commands to the modem and returns the reply (empty if a timeout occurred)



```
Public Function HayesCommand(Handle, Command, TimeoutSec)
    Result = ""
    Call FlushPort(Handle)
    ' Tack on a CR and send the command to the modem
    Call WritePort(Handle, Command + Chr(13))
    BytesRead = 0
    BytesRemain = 0
    ' Wait up to the timeout for 1 character to come in
    Result = ReadPort(Handle, 1, BytesRead, BytesRemain, TimeoutSec)
    ' Now read in any additional characters but with a very short timeout
    If Result <> "" Then
        Result = Result + ReadPort(Handle, 1000, BytesRead, BytesRemain, 0.01)
    End If
    HayesCommand = Trim(Result)
End Function
```

```
' Sample subroutine that may be commented out and scheduled.
' It will open COM1: and prompt for some characters and echo them back
' It requires that noone is logged in to COM1:
'
'Public Sub Sched_SerialLibTest
' Handle = OpenPort("COM1:", LoggedIn)
' If Handle = 0 Then
'     ErrorMsg "SerialLib could not open COM1:"
'     Exit Sub
' End If
' StatusMsg "SerialLib DSR = " & DSRPort(Handle) & ", CD = " & CDPort(Handle) & ", DSR
= " & DSRPort(Handle)
' CRLF = Chr(13) + Chr(10)
' Call WritePort(Handle, CRLF+CRLF+"Hello there. This is a test of
SerialLib.bas"+CRLF+"Please enter a string: ")
' BytesRead = 0
' BytesRemain = 0
' Reply = ReadPort(Handle, 32, BytesRead, BytesRemain, 10.0)
```



APPLICATION NOTE

IRIDIUM (MODEL A3LA-D) SATELLITE MODEM & XLITE 9210 DATA LOGGER

```
' Call WritePort(Handle, CRLF + "You typed: " + Reply + CRLF)
' Call WritePort(Handle, "BytesRead is "+BytesRead+CRLF)
' Call ClosePort(Handle)
'End Sub
```