

```

// - - - - -
// RFC-1/B script for Telix for DOS (SALT)
// - - - - -

// The constants below should be changed to reflect your RFC-1/B.  The original
// settings will dial the Sine Systems demo transmitter and log its readings.

// TelNum is the telephone number to dial to reach the RFC-1/B.

// MainCode is the main security code of the RFC-1/B.

// SiteID is the site ID that is programmed in the RFC-1/B.  This is not
// usually case sensitive but it must be exact spaces, etc.

// Setup is a modem init string in addition to the one that the com program
// already sends.  This must be changed to one that works with your modem.
// Use this to disable fax, compression, and automatic error correction.

// LogPath/LogFile are the path and filename of the file to log readings.

// ChStart/ChStop are the start and stop channels to log readings.

// LogPath/LogFile are the path and filename of the file to log readings.

// ToDisk/ToPrint will toggle disk and printer logging.  1 is on, 0 is off.

// - - - - -
// User Global Variables
// - - - - -

str TelNum[] = "1-615-777-7321" ;
str MainCode[] = "12345678";
str SiteID[] = "This is RFC1B";
str Setup[] = "AT\n";
str LogFile[] = "d:\temp\RFC1LOG.TXT";
int ChStart = 0;
int ChStop = 13;
int ToDisk = 1;
int ToPrint = 0;

// - - - - -
// Main Procedure
// - - - - -

main()
{
// _add_lf = 0;

// Local Variables
str ErrMsg[40];

// Make three attempts at logon
clear_scr();
ErrMsg = DoLogon();

if (not(carrier()))
{
hangup();
printS ("Making second attempt in 90 seconds");
delay (900);
ErrMsg = DoLogon();
}

if (not(carrier()))

```

```

    {
        hangup();
        printS ("Making third attempt in 90 seconds");
        delay (900);
        ErrMsg = DoLogon();
    }

// Clear before logging
clear_scr();

// Capture to disk enable
if (ToDisk == 1)
    capture (LogFile);

// Capture to printer enable
if (ToPrint == 1)
    printer (1);

// Display header
DoHeader();

if (carrier())
// Take readings
    TakeReadings();
else
// Log message
{
    printS ("^M^J");
    printS ("No readings available");
    printSC ("Error: ");
    printS (ErrMsg);
}

// Display footer
DoFooter();

// End capture
capture ("*CLOSE*");
printer (0);

// Logoff and hangup
printS ("Hanging up");
cPutS ("99");
hangup();
}

// - - - - -
// Login Procedure
// - - - - -

DoLogon()
{

// Local Variables
int Rcvd;

// Send modem setup str
printS ("Setting up modem");
cPutS (Setup);
cPutS ("^M");
Rcvd = (waitfor("OK", 5));
if (not Rcvd)
    return ("Setup str is invalid for this modem");
else
    delay (20);
}

```

```

// Send modem dial command and wait for connect message
cPutS ("ATDT");
cPutS (TelNum);
cPutS ("^M");
cPutS ("Waiting for connect");
Rcvd = (waitfor("CONNECT", "NO CONNECT", "NO CARRIER", "BUSY", "NO DIALTONE", 60));
if (Rcvd == 2)
    return "No connection";
else if (Rcvd == 3)
    return ("No carrier detected");
else if (Rcvd == 4)
    return ("Line busy");
else if (Rcvd == 5)
    return ("No dialtone");

// Wait for login prompt
Rcvd = (waitfor("ENTER", 10));
if (not Rcvd)
    return ("No logon prompt");
else
    delay (10);

// Send security code
cPutS (MainCode);
cPutS ("^M");
Rcvd = (waitfor(SiteID, 5));
if (not Rcvd)
    return ("Incorrect security code");

// Default Message
return "";
}

// -----
// Readings Procedure
// -----

TakeReadings()
{
// Local variables
int Tens;
int Ones;
int LoopNum;
int Rcvd;
str ChannelNum[3];

// Take readings
flushbuf();
delay(5);
for (LoopNum=ChStart; LoopNum <= ChStop; LoopNum = LoopNum + 1)
{
    Tens = (LoopNum / 10);
    Tens = Tens + 48;
    Ones = (LoopNum % 10);
    Ones = Ones + 48;
    cPutC (Tens);
    cPutC (Ones);
    Rcvd = (waitfor("^M", 3));
    if (Rcvd)
        delay (5);
    else
    {
        ItoS (LoopNum, ChannelNum);
        printSC ("Error on channel: ");
    }
}
}

```

```

        printS (ChannelNum);
    }
}

// -----
// Header Procedure
// -----

DoHeader()
{
// Local variables
str DateCap[9];
str TimeCap[9];

// Header routine
date(curtime(), DateCap);
time(curtime(), TimeCap);
printS ("-----");
printSC ("Site: ");
printS (SiteID);
printSC ("Date: ");
printS (DateCap);
printSC ("Time: ");
printS (TimeCap);
printS ("-----");
}

// -----
// Footer Procedure
// -----

DoFooter()
{
    printS ("End of set");
    printS ("^M^J");
}

// -----
// End of file
// -----

```