Web Site: www.parallax.com
Forums: forums.parallax.com
Sales: sales@parallax.com
Technical: support@parallax.com

Office: (916) 624-8333
Fax: (916) 624-8003
Sales: (888) 512-1024
Tech Support: (888) 997-8267

PINK (Parallax Internet Netburner Kit - #30013)

General Description

The Parallax Internet Netburner Kit (PINK) provides an easy yet powerful solution for interfacing your BASIC Stamp® or SX microcontroller to an ethernet enabled network, such as the Internet.

Features

- Embedded web server with customizable web pages
- Easy to use web-based configuration interface
- FTP interface for drag-and-drop update of files
- Support for both DHCP and static IP support
- Telnet interface (can be used for debugging BASIC Stamp serial communications)
- Sending email
- Password protection of configuration and user selected web pages
- 10/100 base-T ethernet interface
- Flash memory for storing default variable values

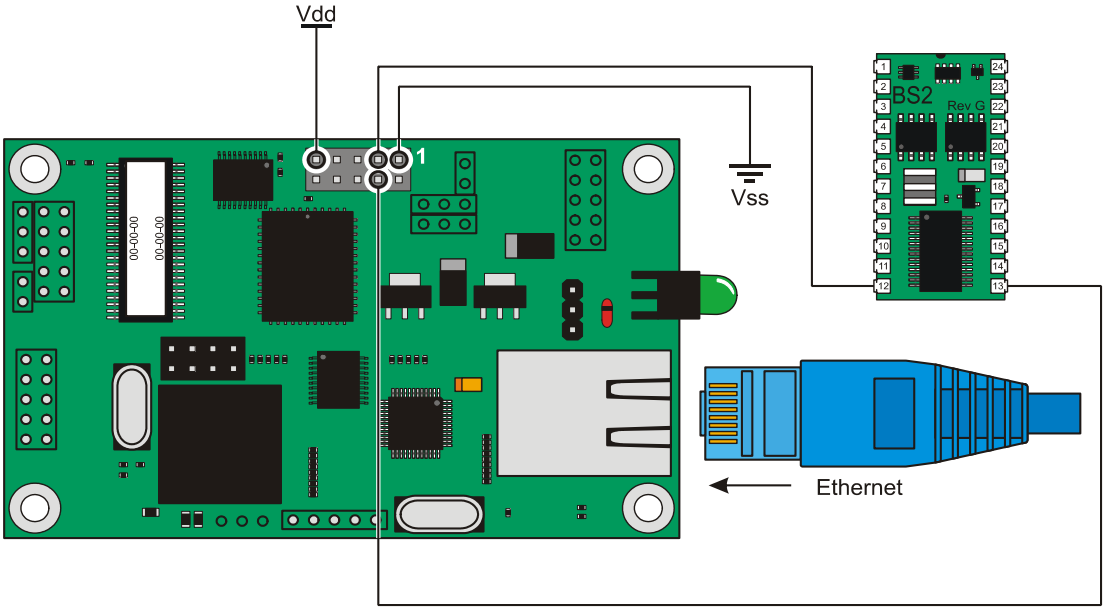
Application Ideas

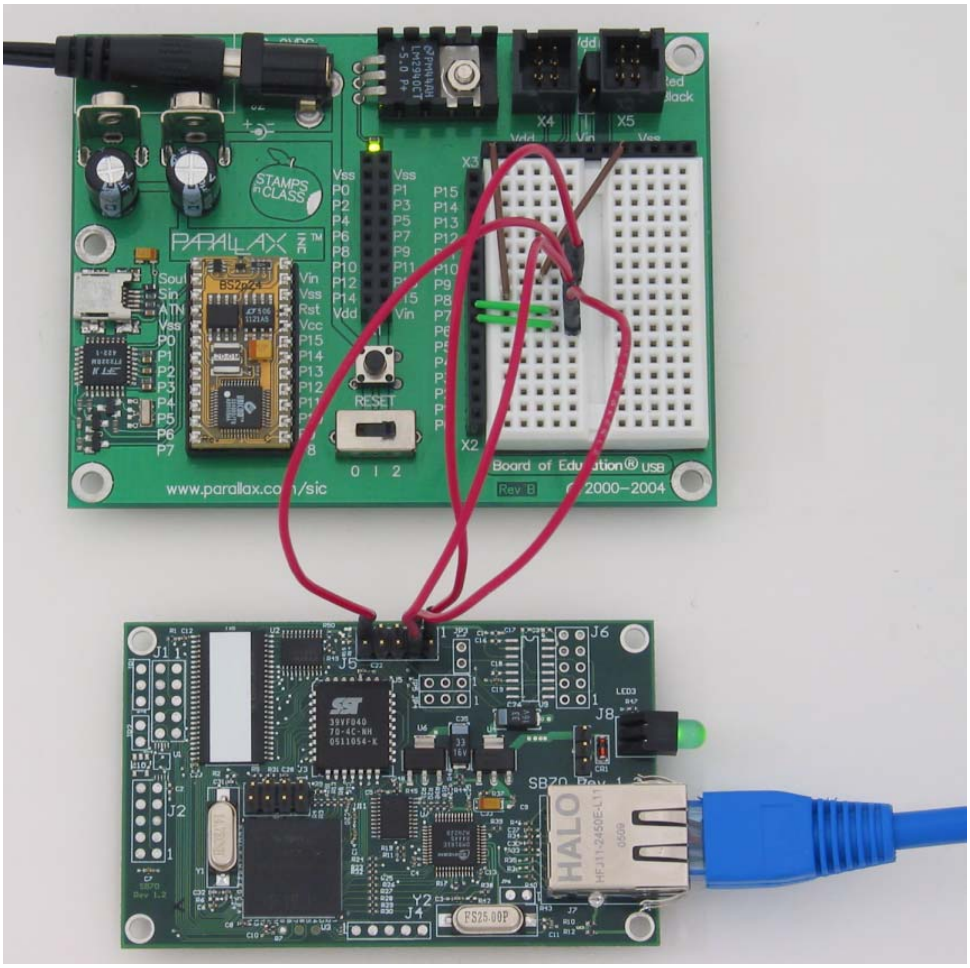
- Web enabling of a BASIC Stamp or SX microcontroller, allowing for the monitoring of data from anywhere in the world via the Internet
- Email alerts or notification sent from BASIC Stamp or SX microcontroller
- Control switches, LEDs, relays or motors over the Internet

Kit Contents

(1) PINK Ethernet Module anywhere in the world via the Internet
(1) Blue Cat5 Straight-through cable
(1) Red Cat5 Cross-over cable
(6) Hookup Wires
(1) Software CD
(1) Printed Documentation

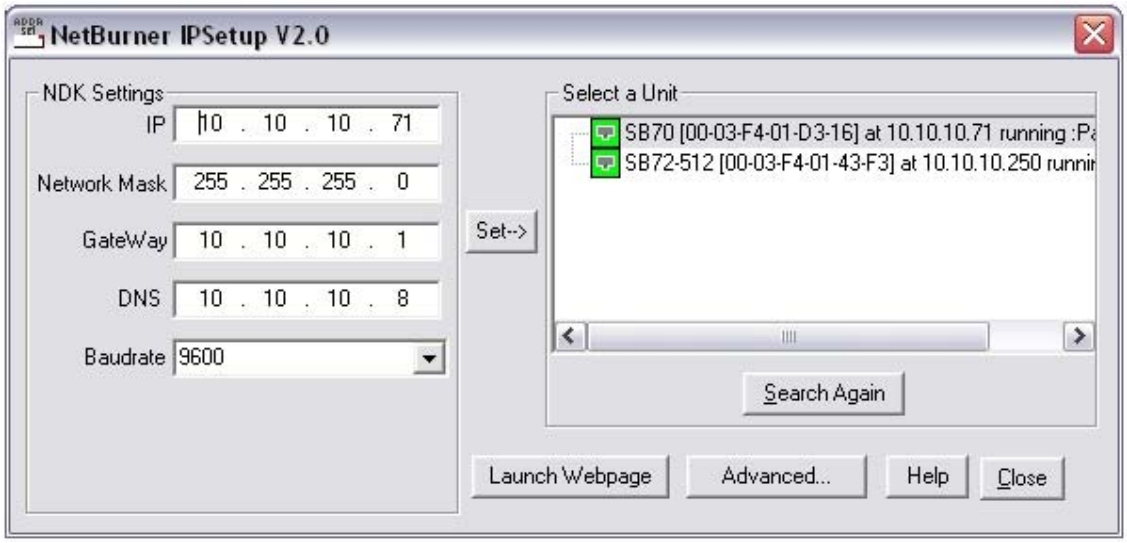
Quick Start Circuit





Assigning an IP Address

Every device on an ethernet network requires an IP (Internet Protocol) address. The PINK module supports both DHCP (Dynamic Host Configuration Protocol) and static (manually assigned) IP addresses. In order to configure the PINK module's IP settings, the NetBurner IPSetup software must be used. IPSetup can be used to simply view the PINK's network settings as well.

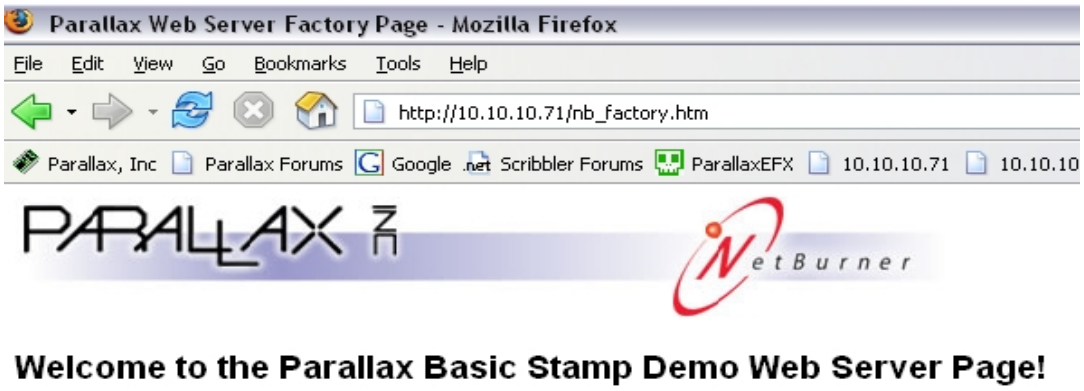


The NetBurner IPSetup software is included on CD with the PINK module; you may also check for a link to a free download of the latest version at the Parallax Internet Netburner Kit product page: http://www.parallax.com/detail.asp?product_id=30013.

Select the unit on the right, and fill in the IP number, Network Mask, Gateway and DNS server addresses on the right. Once these numbers are filled in, clicking Set will set up the module with the information provided. To use DHCP, set all four IP address fields to 0.0.0.0 and press the Set button.

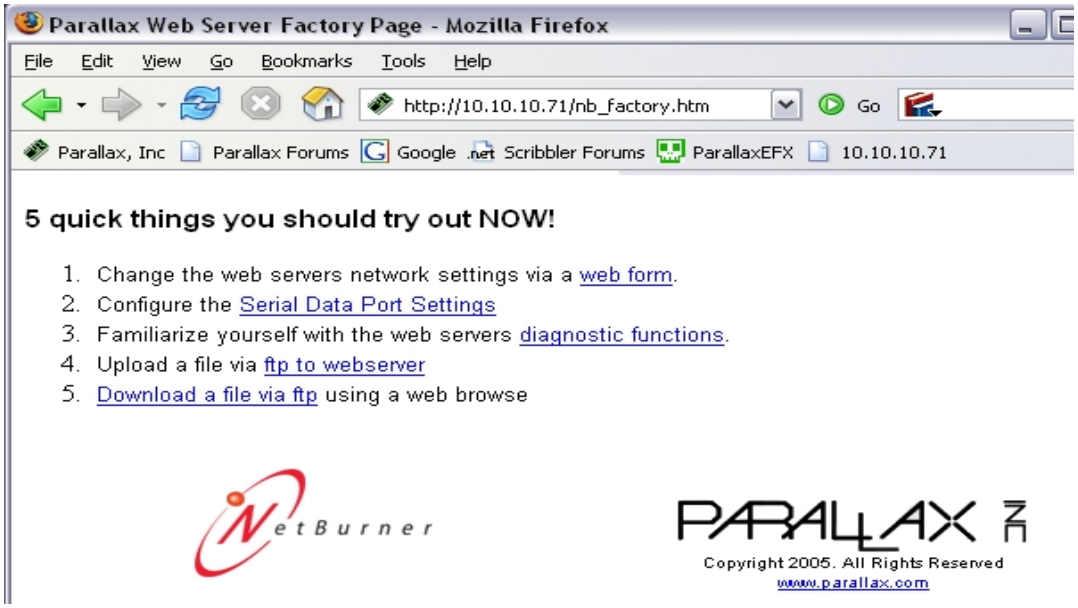
From the IPSetup utility you can launch the default web page on the PINK module by clicking the Launch Webpage button. If no files have been uploaded to the module previously, the default factory web page will be loaded. You can also just type the IP address of the PINK module into your web browser's URL field.

The Configuration Web Page

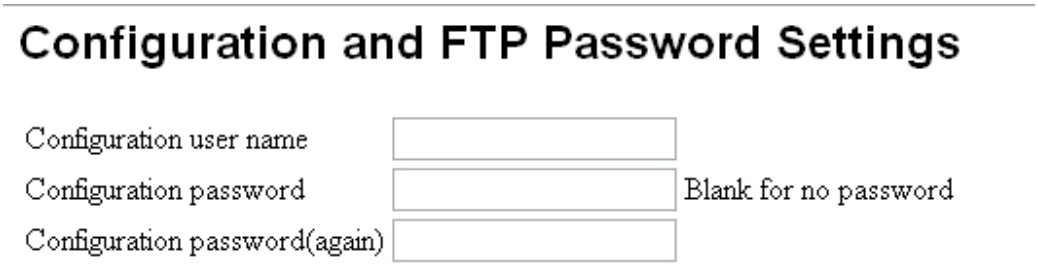


Browsing to the default IP of the PINK module *before* uploading custom web pages will bring up the default factory web page. The address of this page is **XX.XX.XX.XX/nb_factory.htm** where **XX.XX.XX.XX** is the IP address you assigned with the NetBurner IPSetup utility (or the IP address assigned to the module by DHCP, if you chose to use DHCP.)

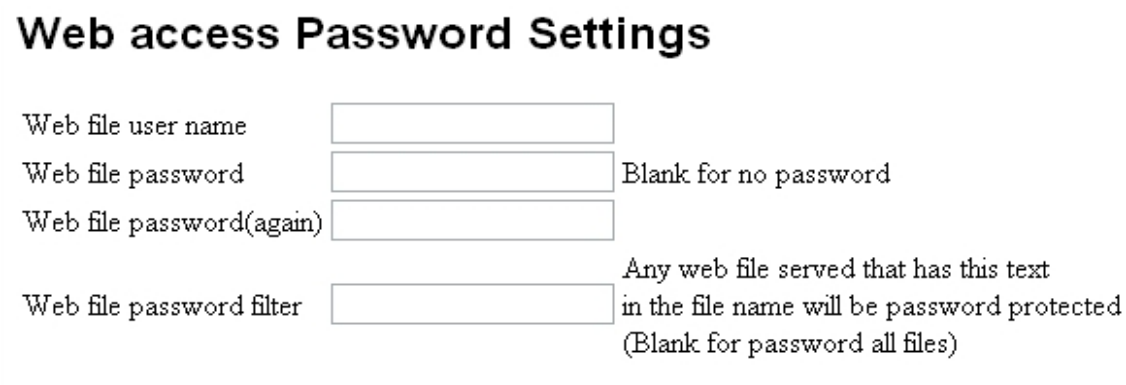
From the default factory web page you can access the network settings, serial data port settings, diagnostic functions, and FTP connections to the PINK module.



The network settings page consists of two parts: the network settings and the password settings. From the IP mode pull-down menu you can select either **Static settings** or **Dynamic settings**. (These settings need not be changed if already configured with the IPSetup program; the web interface is simply another method of changing the network IP settings.) If you are using DHCP, from this page you can see the addresses assigned by the DHCP server as well.



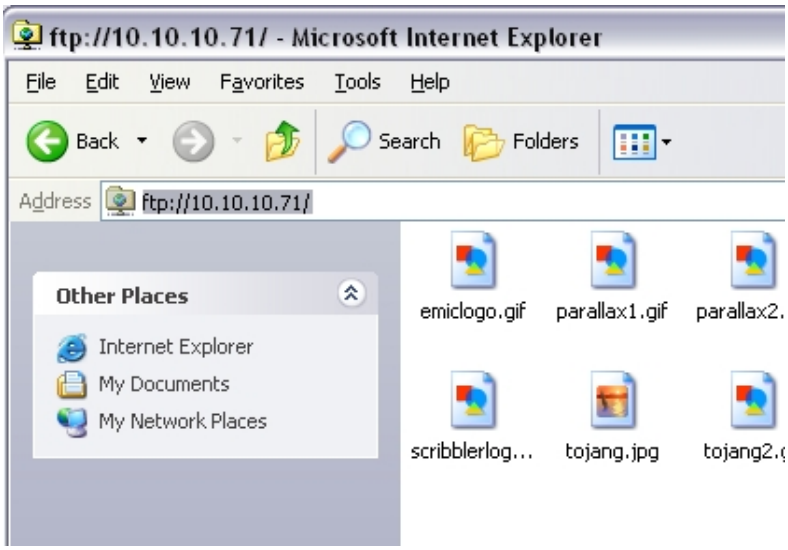
Setting the configuration and FTP password will prevent anonymous FTP connections, and password protect the configuration web pages on the PINK module. The password is repeated for verification.



You can enable password protection for web pages with the web access password settings dialog boxes. Filling in a web file user name and password (repeated for verification) yet leaving the web file password filter field blank will password protect *all* web files. Setting the password filter protects any file with the filter string in the filename. For example if the filter was set to "*pass*," then the files "*indexpass.htm*," "*passindex.htm*," or "*passtime.jpg*" would become password protected. Files not containing the password filter will be freely accessible. In order to access files filtered with the web file password setting, the correct user name and password must be entered.

Custom Web Pages

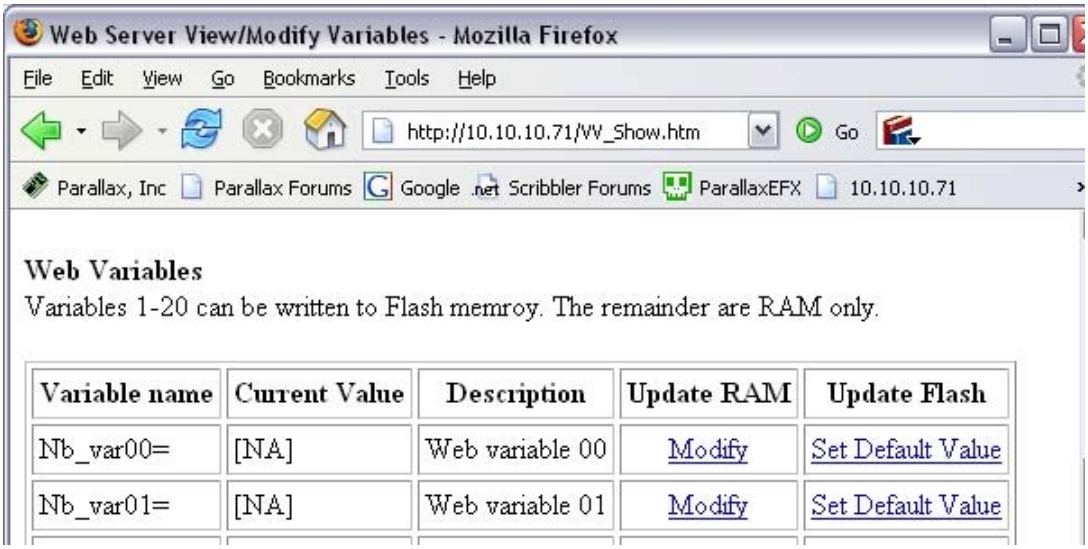
Custom web pages can be uploaded to the PINK module via FTP. Typing **FTP://XX.XX.XX.XX** into an Internet Explorer browser window will open a drag-and-drop FTP window where custom web pages can be moved, placing them in the PINK module's memory. (Where **XX.XX.XX.XX** is the address of the PINK module.)



If a specific web page is not indicated, the PINK will look for **index.html**. If no **index.html** is present, the default factory web page will be loaded.

Variables

The PINK module has 100 general purpose variables. The first 20 variables (0 through 19) can be written to flash memory. The remaining variables (20 through 99) are RAM variables only. These variables are all 64 bytes (maximum size).



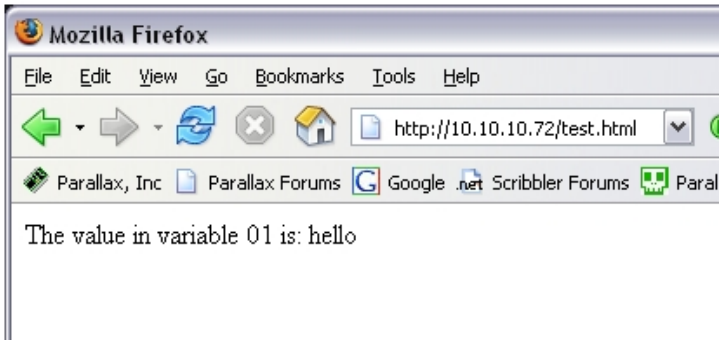
All of the variables can be accessed through a web interface by browsing to **XX.XX.XX.XX/VV_Show.htm** (where XX.XX.XX.XX is the assigned address of the module.) To change a variable via the web interface, click *Modify* in the variable location to be changed. Once a variable has been changed, clicking the *Set Default Value* option for that variable will write the current value to flash memory. Once a value has been written to flash, it will be the default value upon power on or reset of the module.

Web access of variables

Accessing the standard variables with a web page is as easy as using an HTML reference to the variable name. To create a webpage that displays the value of variable 01, open your favorite text editor and enter the following:

```
<html>
The value in variable 01 is: <Nb_var01>
</html>
```

Save the file, using an *htm* or *html* extension (for example *test.html*). Open an FTP connection (as explained previously) and drop the file into the PINK module. Browsing to **XX.XX.XX.XX/test.html** loads the page (where XX.XX.XX.XX is the address of the PINK module):

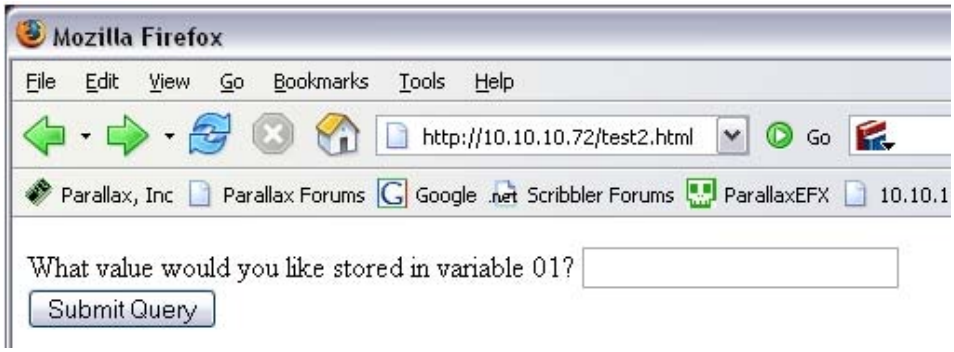


In this example the word “hello” was in the variable. If the value in the variable is changed, the resultant page output is changed as well. In this way dynamic data can be viewed via web pages that automatically update as the data does.

Writing to variables via web page can be accomplished with HTML POST methods. Assigning data the name of the variable that you wish to store it in will update the variable accordingly. To create a webpage that changes the value of variable 01, open a text editor, and enter the following text:

```
<html>
<FORM method="post" action="/test.html">
<P>
What value would you like stored in variable 01?
<INPUT name="Nb_var01"type="text"size="24"maxlength="63">
<INPUT type="submit">
</P>
</FORM>
</html>
```

Save the file with an *HTM* or *HTML* extension such as *test2.html*. Browsing to the page **XX.XX.XX.XX/test2.html** (where XX.XX.XX.XX is the address of the PINK module) opens a page that can be used to change the value stored in variable 01:



BASIC Stamp Microcontroller Access of Variables

To read the value of a variable with a BASIC Stamp, the following command is sent serially to the PINK module:

!NB0RXX

where **XX** is the number of the variable you wish to read (00 to 99). This command is sent serially, for example on a BS2:

```
'  PINK_01.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

NBVAR VAR Byte

SEROUT 8,396,["!NB0R06"]
SERIN  7,396,[NBVAR]

DEBUG DEC NBVAR

END
```

This will request from the PINK module the value in variable 06, which is then read into the BASIC Stamp variable NBVAR with the SERIN command. Since all data sent from the PINK module to the BASIC Stamp microcontroller will be ASCII text, the DEC formatter is used to get the decimal value of the variable.

A 2400 baud rate was set on the PINK module, so 396 is the baud mode used, as this example was written for a BASIC Stamp 2. The baud rate selected on the PINK module must be supported by the BASIC Stamp model you are using. *(For a BS2, BS2e or BS2pe, 2400 baud is suggested. Please see the BASIC Stamp Manual command reference for SERIN for more information. This information is also available in the BASIC Stamp Editor online help file.)*

To read a variable holding a string:

```
' PINK_02.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

NBVAR VAR Byte(16)

SEROUT 8,396,["!NB0R06"]
SERIN  7,396,[STR NBVAR\16\CLS]
DEBUG STR NBVAR

END
```

Writing to variables from a BASIC Stamp module is accomplished with commands formatted like this:

!NB0WXX:DD

where **XX** is the variable you wish to write to, and **DD** is the data you wish to send. The command must be followed by a **CLS**. For example:

```
' PINK_03.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

SEROUT 8,396,["!NB0W06:25",CLS]
END
```

This code will write a “25” to variable 06 on the PINK module. For longer strings, you can simply replace the “25” with the data you wish to put into the variable.

Special Purpose Registers

In addition to the 100 standard variables, the PINK module has nine special purpose registers:

- **Nb_varET** is the email TO: register. When using the PINK to send email, this register will hold the TO: address.

- **Nb_varEF** is the email FROM: register. When using the PINK to send email, this register will hold the FROM: address.

- **Nb_varES** is the email SUBJECT: register. When using the PINK to send email, this register will hold the SUBJECT of the email message.

- **Nb_varEC** is the email CONTENT: register. When using the PINK to send email, this register will hold the name of the file containing the content for the email message. The file must be on the PINK module prior to sending an email message. A *.txt file will be written out as the body text of an email message if indicated in the Nb_varEC variable. In the event that the file indicated cannot be found, the body of the email will read “Could not find file[DD]” where DD is the value that was in the Nb_varEC variable at the time of sending. This can be a quick way of sending an email without creating a *.txt or other type of file. You may also send a web page as the email body, allowing you to easily view variable values or other data.

- **Nb_varEV** is the email SMTP server register. This register must contain the address of a valid SMTP server for sending email.

- **Nb_varST** is the PINK module’s status register. This is a read-only variable. Every time the status variable is read, bits 1 and 3 are cleared. The bits are used for information on the status of the PINK module and the network as follows:

    - **Bit 0**: This is a network status register bit. If a network connection is established, this bit will be set. A cleared bit in this position indicates that no network connection has been established.

    - **Bit 1**: This is a variable update bit. If any of the variables have changed or been updated via web page since the last read of the status register, this bit will be set. A read of the status register clears this bit. This is useful for checking to see if data has been entered into the PINK module via a web page post since the last time the BASIC Stamp module polled for updated data.

    - **Bit 2**: If bit 2 is cleared the PINK module is ready to send an email. If the PINK module is busy sending an email, bit 2 will be set. In order to send an email, the Nb_varET, Nb_varEF, Nb_varES, Nb_varEC and Nb_varEV variables must contain correct information.

    - **Bit 3**: If bit 3 is set there was an error sending the last email transmission. A read of the status register clears this bit.

        - **Bit 4**: If bit 4 is set then the PINK module has received a UDP message. A read of the status register clears this bit.

- **Nb_varSV** will hold the number of the last variable updated from a web page POST. To read the value of this variable the command is:

    !NB0SV

    PBasic code to check the Nb_varSV variable to see what the last web updated variable was can be structured as follows:

```
' PINK_04.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

NBVAR VAR Byte

SEROUT 8,396,["!NB0SV"]
SERIN  7,396,[NBVAR]

DEBUG DEC NBVAR

END
```

- **Nb_varBI** is used for holding the IP destination address for UDP net messages.

- **Nb_varBM** is used for holding the content of a UDP net message. This is not the name of a file as is the case with the email variable, but the actual message content.

Sending Email

In order to send email via the PINK module, all of the email related variables must be set: Nb_varET, Nb_varEF, Nb_varES, Nb_varEC and Nb_varEV. If a text message named message.txt was already located on the PINK module, a BASIC Stamp microcontroller program for sending email may be structured like this:

```
' PINK_05.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

SEROUT 8,396,["!NB0WET:null@parallax.com",CLS]
SEROUT 8,396,["!NB0WEF:PINKmodule@parallax.com",CLS]
SEROUT 8,396,["!NB0WES:This is a test message from PINK.",CLS]
SEROUT 8,396,["!NB0WEC:message.txt",CLS]
SEROUT 8,396,["!NB0WEV:your.SMTP.server.address.here",CLS]

SEROUT 8,396,["!NB0SM"]

END
```

The first five serout commands set up the email, and the **!NB0SM** instruction actually tells the PINK module to send the email message.

Sending a UDP Net Message

PINK modules can send and receive UDP (User Datagram Protocol) messages. To send a UDP message, the Nb_varBI and Nb_varBM variables must be initialized. Nb_varBI must contain the destination IP address, and Nb_varBM must contain the UDP message to be sent. Once these variables are initialized, issuing the command

!NB0SB

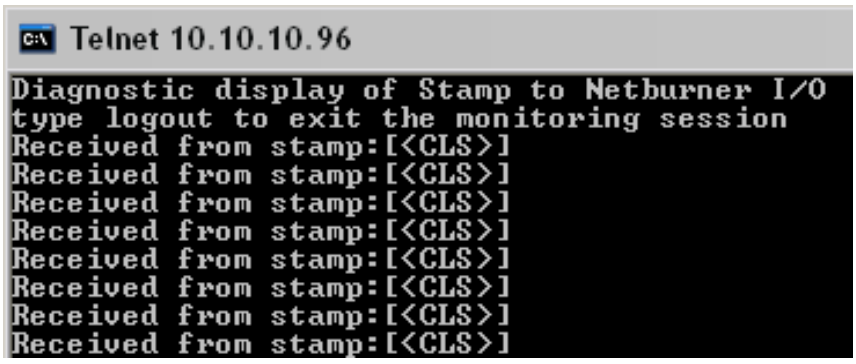
will send the UDP message. Here is sample code for sending a UDP message:

```
' PINK_06.bs2
' {$STAMP BS2}
' {$PBASIC 2.5}

SEROUT 8,396,["!NB0WBI:10.10.10.71",CLS]
SEROUT 8,396,["!NB0WBM:This is a UDP test message",CLS]

SEROUT 8,396,["!NB0SB"]

END
```

This will send the UDP message “This is a UDP test message” to the destination 10.10.10.71. If the address being sent to is another PINK module, that pink module will take the message it was just sent, and put it into the Nb_varBM variable on that module. If a UDP message has been received by a PINK module, Bit 4 of the status register (**Nb_varST, bit 4**) will be set. Reading the status register will clear this bit. Using UDP messaging it is therefore possible for BASIC Stamp modules to send messages back and forth via ethernet.

Telnet Debug Session

To activate the telnet debug window, open a command prompt and type **telnet xxx.xxx.xxx.xxx** (where **xxx.xxx.xxx.xxx** is the IP address of the module). This will open a window that shows all serial communication going between the Basic Stamp module and the PINK module.



BASIC Stamp 2p Code Example

The following code will interface a BASIC Stamp 2p module with a PINK module and the EMIC text-to-speech module. This will allow a user to enter ASCII text to a website, and have the text spoken aloud by the EMIC module which is being controlled by the BS2p.

```
' PINK_07.bsp
' {$STAMP BS2p}
' {$PBASIC 2.5}
' Parallax Inc. Tech Support: support@parallax.com

' Ethernet -> Web -> Speech interface for Netburner Module
' This code is for interfacing a Basic Stamp 2p microcontroller
' with a PINK module (#30013) and an EMIC text-to-speech module (#30006)
' In addition to this code, HTML for the web interface is required.
' Users can then browse to a web page, type in ASCII text, and have the text
' spoken by the EMIC module.
'

'-----------------------------------------------------------------------------------------
'VARIABLES
'-----------------------------------------------------------------------------------------
UPREG       VAR Byte
NBSTATUS    VAR Byte            'Will hold the status of the PINK module
NewPost     VAR NBSTATUS.BIT1   'Bit that indicates if the PINK has been updated
                                'via a POST (the PostSTATUS variable holds the actual
                                'variable location, NewPost only indicates if a POST
                                'update has been made or not.

Tx    PIN 3                     'Connect BS2P pin3 to TX on EMIC
Rx    PIN 4                     'Connect BS2P pin4 to RX on EMIC
Busy  PIN 5                     'Connect BS2P pin5 to BUSY on EMIC
Rst   PIN 6                     'Connect BS2P pin6 to /RESET on EMIC

EOM   CON $AA                   'End of Message indicator for EMIC strings
vol   CON 5                     'Used to set the volume of the EMIC
ptch  CON 1                     'Used to set the pitch of the EMIC
spd   CON 1                     'Used to set the speed of the EMIC

eePntr  VAR    Byte             'Pointer used for Scracth Pad RAM
char    VAR    Byte
x       VAR    Byte

'-----------------------------------------------------------------------------------------
'INITIALIZATION
'-----------------------------------------------------------------------------------------
PAUSE   200                     'Allow PINK time to get up & on the network

'-----------------------------------------------------------------------------------------
'MAIN
'-----------------------------------------------------------------------------------------
GOSUB Hard_Reset
PAUSE 2000
GOSUB Set_Voice

MAIN:                                       'The main program loop
  PAUSE 300                                 'pause for serial reads/writes (PINK)
  SEROUT 8,240,["!NB0ST"]                   'Poll for webpage update by sending a request
  SERIN  7,240,100,MAIN,[NBSTATUS]          'for the state of the status register (PINK)
  IF NewPost = 1 THEN speak                 'Check for update from web interface
```

```
GOTO MAIN                                       'Check everything again!
END

'-----------------------------------------------------------------------------------------
'Subroutines
'-----------------------------------------------------------------------------------------

speak:
  SEROUT 8,240,["!NB0R04"]
  SERIN 7, 240, 30, Say_String, [SPSTR 64]
  GOSUB Say_String
RETURN

Hard_Reset:
  LOW Rst
  PAUSE 2
  INPUT Rst
  GOSUB Wait_OK
RETURN

Wait_OK:
  SERIN Rx, 1021, 6000, TO_Error, [WAIT($55)]
RETURN

TO_Error:
  DEBUG CLS, "EMIC module did not initialize correctly."
  GOTO MAIN

Check_Busy:
  PAUSE 1
  DO WHILE (Busy = 1) : LOOP
RETURN

Say_String:
  eePntr = 0
  OUT0 = 1
  SEROUT Tx, 1021, [$00]
  DO
    GET eePntr, char
    SEROUT Tx, 1021, [char]
    eePntr = eePntr + 1
  LOOP UNTIL (char = CLS)
  SEROUT Tx, 1021, [$AA]
  GOSUB Check_Busy
RETURN

Set_voice:
  GOSUB Check_Busy
  SEROUT Tx, 1021, [$01, DEC vol, $AA]
  GOSUB Wait_OK
  SEROUT Tx, 1021, [$03, DEC ptch, $AA]
  GOSUB Wait_OK
  SEROUT Tx, 1021, [$02, DEC spd, $AA]
  GOSUB Wait_OK
RETURN
```

In addition to the BASIC Stamp module code, an HTML file is required on the PINK module:

```
<html>
<body>
```

```
Parallax Ethernet to Speech Interface<br>

<form method="POST" action="index.html" enctype="text/plain">

<pre>
<font size="6"><b> Speech</font></b>
Speak these words:<input name="Nb_var04" type="text" size="64" max length="64">
</pre>
<center>
<input type="submit" value="Speak">
</center>
</form>

</body>
</html>
```

The HTML file should be saved as *index.html* and loaded into the PINK module.