Software Manual

Rev. 0014A

NetMedia, Inc.
Tucson, Arizona

NetMedia, Inc.
10940 N. Stallard Pl.
Tucson, Arizona 85737
TEL: (520) 544-4567
FAX: (520) 544-0800

PURCHASE TERMS AND CONDITIONS

The laws of the State of Arizona shall govern PURCHASE TERMS AND CONDITIONS.

LIMITED WARRANTY: NETMEDIA MAKES NO WARRANTIES OTHER THAN THOSE CONTAINED HEREIN AND NETMEDIA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING ANY WARRANTY OF FITNESS FOR A PARTICULAR PURPOSE OR OF MERCHANTABILITY.

The foregoing limited warranty shall not apply unless Buyer has paid for in full the NetMedia products. Electronic updates to the NetMedia SitePlayer User's Manual and NetMedia SitePlayer software are available free to Registered Buyer upon request for a one (1) year period from the invoice date.

NOTICE

NetMedia, Inc. reserves the right to make improvements in the software product described in this manual as well as the manual itself at any time and without notice.

DISCLAIMER OF ALL WARRANTIES AND LIABILITY

NETMEDIA, INC. MAKES NO WARRANTIES, EITHER EXPRESSED OR IMPLIED, WITH RESPECT TO THIS MANUAL OR WITH RESPECT TO THE SOFTWARE DESCRIBED IN THIS MANUAL, ITS QUALITY, PERFORMANCE, MERCHANTABILITY, OR FITNESS FOR ANY PARTICULAR PURPOSE. NETMEDIA, INC. SOFTWARE IS SOLD OR LICENSED "AS IS". IN NO EVENT SHALL NETMEDIA, INC. BE LIABLE FOR INCIDENTAL OR CONSEQUENTIAL DAMAGES RESULTING FROM ANY DEFECT IN THE SOFTWARE.

Table of Contents

SitePlayer Overview

Welcome to SitePlayer™, the World's smallest Ethernet web server. The first product in a family of embedded web servers designed to enable any microprocessor-based device to become web enabled easily and inexpensively. In approximately one square inch, SitePlayer includes a web server, 10baseT Ethernet controller, flash web page memory, graphical object processor, and a serial device interface. SitePlayer is a plug in module that can also be used as a web enabled option, product upgrade, or to retrofit older products. Example applications include audio equipment, appliances, thermostats, home automation, industrial control, process control, test equipment, medical equipment, automobiles, machine control, remote monitoring, and cellular phones.

SitePlayer is a web server coprocessor that handles web protocols and Ethernet packets independently of the device processor. Web traffic does not effect the device processor, which also adds a measure of security. Communications between SitePlayer and the device is accomplished through objects sent through a standard two wire serial port. No TCP/IP or network code is required.

SitePlayer can also be used in some applications in a "stand alone" mode where simple I/O can be performed. SitePlayer has 8 configurable I/O pins which can also be used as four 8 bit PWMs or DACs, frequency generators, or event counters. These functions are available to a device processor also.

SitePlayer contains a powerful object system called SiteObjects™ which allow graphical images, text, music, links, radio buttons or checkboxes to change based on live data from the device processor without the need for CGI scripts or Java programming. A web page can contain a graphical knob rotated to a position, a switch can be toggled up or down, or a link can change based on a variable in the device processor.

Standard web authoring tools are used to make the pages for SitePlayer. Web pages are downloaded to SitePlayer's flash memory over the Internet. SitePlayer firmware updates are also downloadable keeping SitePlayer current. A library of graphical knobs, switches, LEDs and other user interface tools are provided for web page development.

SitePlayer Process for Creating a Project

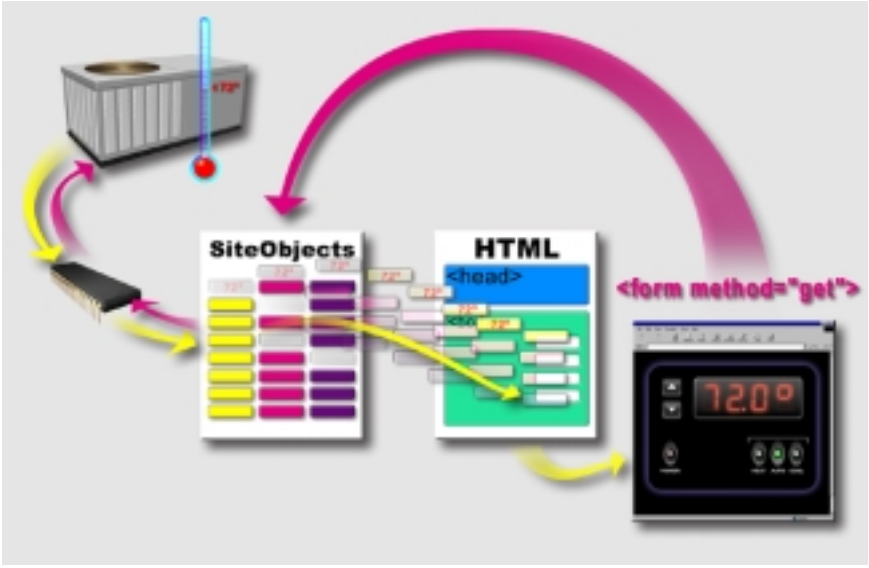
The code that tells SitePlayer how to perform and what web pages to serve must be defined by the SitePlayer Definition file and assembled into the SitePlayer Binary image using SiteLinker. This in turn, is downloaded to SitePlayer through your Ethernet connection. You can serve your web pages and interact with your device by using the SitePlayer module or SitePlayerPC. The chapters of this manual correspond to the steps involved in the creation of your SitePlayer project:

1. Define and create your objects using a text editor with a SitePlayer Definition File (.SPD)
2. Create your web pages using an HTML editor
3. Assemble and Download SitePlayer Binary file (.SPB) using SiteLinker program
4. Surf SitePlayer or SitePlayerPC using web browser

Documentation for Serial Transmitting and Receiving is located in Appendix A. Appendix B contains a list of the SitePlayer Definition File's Reserved Symbols.

SitePlayer Data Flow

In the illustrated example below, the picture of the integrated circuit represents your device processor. It acts as a smart thermostat which controls the Air Conditioning unit based on information it gathers such as current temperature and setpoint. This same information is written through the serial port to SitePlayer (see Serial Transmitting and Receiving) and stored in the memory locations defined by the Objects of the SitePlayer Definition file (see Object Section). When web pages are requested from SitePlayer, it substitutes the Object value for each SiteObject it encounters (see SiteObjects in HTML) and serves the information to your web browser via the Ethernet port.



Changes are submitted back to SitePlayer through the Ethernet port by using the web browser's HTML GET method (see SitePlayer Web Pages). There, the new values are written to the SitePlayer Objects as defined by the Definition file. Your device processor then requests the Object values from SitePlayer through the serial port and acts accordingly, in this case, by changing the mode or setpoint of the Air Conditioning unit.

If for instance, a change in setpoint then caused the cooling unit to turn on, this new information would be serially written to the SitePlayer Object, where it would in turn be dynamically substituted for the corresponding SiteObject in the web page HTML. In short, the web browser would show the cooling unit as now being on.

Your device processor's decision on when to ask SitePlayer for new data can be based on a timer that always gets all the data, or on a software status request that can then selectively retrieve data (see serial Status command), or on the hardware interupt of pin 11 (see $PostIRQ definition).

1) Design and Create Definition File

The SitePlayer Definition file (File Extension .SPD) contains the instructions for the SiteLinker program. The SiteLinker program creates the SitePlayer Binary image file (File Extension .SPB) which gets downloaded to SitePlayer. The binary file contains all the code for SitePlayer including its startup parameters, web pages, and SiteObjects. These parameters, file locations, and SiteObjects, are determined by the SitePlayer Definition file. In addition, the SitePlayer Definition file provides for exporting its information to other file formats such as HTML, C, and Visual Basic.

SitePlayer Definition File Example

The SitePlayer Definition File looks very much like an assembly language file for a microprocessor. Just without any assembly language. It has three sections, the Definition section, Object section, and Export section. Definitions determine SitePlayer's startup parameters. Objects consist of the items that will become SiteObjects. The Export section defines how and where any files will be exported during SitePlayer's SiteLinker process.

A typical file looks like:

```
;DEFINITION SECTION
$definition1 argument             ;comment
$definition2 argument
…
$definitionx argument

;OBJECT SECTION
     org 0h
object1  db 14                    ;defines a byte object with a default of 14
object2  dw 0                     ;defines a word object, default 0
object3  dd 0                     ;defines a long object
object4  ds 47                    ;defines a string object of 47 bytes
object5  db "String with some data"
object   dbit 1                   ;defines a bit with the value of 1

;EXPORT SECTION
$ExportFormatFile "filename"      ;defines export format
$ExportFile       "filename"      ;defines export file name
$Export                           ;export command
```

Comments are separated by semicolons like a typical assembly language file. All number values can be defined in decimal, hexadecimal (h), binary (b), or octal (o). If no letter designation follows, then SitePlayer defaults to decimal. SitePlayer stereo receiver example:

```
;
; Initial receiver DEFINITIONS
;

;$devicename sets the name or description of the device
$devicename "NetMedia Audio Receiver Demo"

;$DHCP on sets SitePlayer to find its IP address from a DHCP server
$DHCP on

;$DownloadPassword sets password for downloading web pages (and firmware)
$DownloadPassword ""
```

```
;$SitePassword sets password for browsing web pages
$SitePassword ""

;$InitialIP sets SitePlayer's IP address to use if no DHCP server is available
$InitialIP "192.168.1.250"

;$PostIRQ on set SitePlayer to generate a low level IRQ on pin 11
$PostIRQ off

;$Sitefile sets the binary image filename that will be created
$sitefile "C:\Program Files\SitePlayer\rcvr.spb"

;$Sitepath sets the root path of the web pages for this project
$sitepath "C:\Program Files\SitePlayer\SP_Root"

;$Include sets the name of a file to include during make process
$Include "C:\Program Files\SitePlayer\pcadef.inc"

;
; Receiver OBJECTS
;
        org 0h
select  db 2                    ;in this example select=2 is FM
volume  db 0                    ;default the volume low
channel dw 921                  ;a word with the default channel of 92.1

        org 0FF15h
power   db 1                    ;default off, address is IO4, the green LED

        org 248                 ;decimal numbers are assumed when no h, b, o
AM_page         dbit 0          ;set status bit with AM page submission
FM_page         dbit 0          ;set status bit with FM page submission
Tape_page       dbit 0          ;set status bit with Tape page submission
CD_page         dbit 0          ;set status bit with CD page submission
Aux_page        dbit 0          ;set status bit with Aux page submission

;
; Other Test OBJECTS
;
        org 30h
xytest_x db 0                   ;x coordinate of xytest graphic
xytest_y db 0                   ;y coordinate of xytest graphic

;
; Receiver EXPORT filenames
;

;$ExportFormatFile contains format definitions for $ExportFile
$ExportFormatFile "C:\Program Files\SitePlayer\makeHTML.def"

;$ExportHeaderFile contains header information for $ExportFile
$ExportHeaderFile "C:\Program Files\SitePlayer\htmlheader.htm"

;$ExportFooterFile contains footer information for $ExportFile
$ExportFooterFile "C:\Program Files\SitePlayer\htmlfooter.htm"

;$ExportFile sets name of export file created from these definitions
$ExportFile "C:\Program Files\SitePlayer\SP_Root\receiver_export.htm"

;$Export command creates HTML $ExportFile and clears parameters
$Export

;Export definitions for a Visual Basic formatted file
$ExportFormatFile "C:\Program Files\SitePlayer\makevb.def"
```

```
$ExportFile "C:\Program Files\SitePlayer\rcvrdefs.bas"
$Export

;Export definitions for a C header file
$ExportFormatFile "C:\Program Files\SitePlayer\makec.def"
$ExportFile "C:\Program Files\SitePlayer\rcvrdefs.h"
$Export

;Export definitions for an Assembly file
$ExportFormatFile "C:\Program Files\SitePlayer\makeasm.def"
$ExportFile "C:\Program Files\SitePlayer\rcvrdefs.asm"
$Export
```

Definition Section

The Definition section of the definition file determines SitePlayer's initial setup parameters. The definitions are preceded by a dollar sign ($) and are controls specific to SitePlayer. No spaces or tabs are allowed between the dollar sign and the body of the control.

From the Receiver example, the definition section looks like this:

```
;$devicename sets the name or description of the device
$devicename "NetMedia Audio Receiver Demo"

;$DHCP on sets SitePlayer to find its IP address from a DHCP server
$DHCP on

;$DownloadPassword sets password for downloading web pages (and firmware)
$DownloadPassword ""

;$SitePassword sets password for browsing web pages
$SitePassword ""

;$InitialIP sets SitePlayer's IP address to use if no DHCP server is available
$InitialIP "192.168.1.250"

;$PostIRQ on set SitePlayer to generate a low level IRQ on pin 11
$PostIRQ off

;$Sitefile sets the binary image filename that will be created
$sitefile "C:\Program Files\SitePlayer\rcvr.spb"

;$Sitepath sets the root path of the web pages for this project
$sitepath "C:\Program Files\SitePlayer\SP_Root"

;$Include sets the name of a file to include during make process
$Include "C:\Program Files\SitePlayer\pcadef.inc"
```

Descriptions for each of these variables are described in the following sections.

$DHCP [on, off] default on

This function defines whether the SitePlayer keeps a fixed IP address (off), or it obtains one from a DHCP server (on). DHCP stands for Dynamic Host Configuration Protocol. Many routers, cable modems, and DSL modems support this feature. For example if you carry a SitePlayer enabled device from one network to another, the DHCP server will allocate a local IP address for the SitePlayer. This makes it very convenient for portable devices.

```
$DHCP on
```

Forces SitePlayer to look for a DHCP server to obtain an address. If SitePlayer cannot find a server, it will go to a default IP address, see $InitialIP

$Devicename [name]

Each SitePlayer can have a 64 character name associated with it. This can be a product name, or some description that will signify your device on a network of many devices.

```
$Devicename “NetMedia Audio Receiver Demo”
```

$DownloadPassword [password]

You may not want everyone to be able to download new web sites into your SitePlayer product. The $DownloadPassword function allows you to specify a 16 character password that is required before someone can download a new web site into your product. Every time you download a new web site into SitePlayer, the password can be changed.

```
$DownloadPassword “mypassword”
```

For extra security, SitePlayer checks to make sure that the computer requesting the download is on the same local network. Meaning that only local requests will be honored for web site updates. This is not fool proof, since hackers can in some cases mimic local access, but it is typically not very easy.

$Include [filename]

You may have a library of objects, or want to use a Library of objects from NetMedia. You use the include function to add those files into your project as if you copied them in directly. An include file can also point to another include file, with up to 8 levels of nesting. The filename should be a fully qualified path.

```
$Include “C:\Program Files\SitePlayer\pcadef.inc”
```

This statement would add the text from pcadef.inc into the current set of object definitions.

$InitialIP [xxx.xxx.xxx.xxx]

For cases where you want SitePlayer to have a fixed IP address, you use this function to enter it. Even if you expect SitePlayer to get an IP address from a DHCP server (see $DHCP), you need to set an initial IP address.

```
$InitialIP 192.168.1.250
```

would set the initial address to 192.168.1.250. SitePlayers with a fixed internal address of 0.0.0.0

```
$InitialIP 0.0.0.0
```

will allow the setting of the IP address through a ping. First, make a static ARP table entry using your SitePlayer’s MAC address (see Serial Port Test Program for finding MAC address) and the intended IP address

```
ARP –S 192.168.1.240 00-03-75-00-05-3A
```

Then ping that address to set SitePlayer

```
PING 192.168.1.240
```

For security reasons, once the address is set with a ping it cannot be changed until a reset or a power up of SitePlayer.

$Parse [extention1]

The $Parse command allows you to choose which file types/extensions will be searched for objects.

Files that are always searched are htm, html, xml, and wml files. If you wish to add a file extension to the list of searched file types, you must use the $Parse command.

For example if you want all *.jar files searched, and *.wow then the commands:

```
$Parse jar
$Parse wow
```

would be placed in the SPD file for the SitePlayer project.

$PostIRQ [on, off] default off

For applications where a user sends some data to the SitePlayer through a form or by pressing special buttons, SitePlayer can transition a hardware pin when the data arrives. This allows applications where the device can sleep or perform other duties until this interrupt occurs.

Pin 11, or IO0 is used for this purpose. Pin 11 will be brought low (0 Volts) and then raised (+5 Volts) creating a pulse of 5 microseconds. To sense this interrupt, a device processor should be placed in a falling edge sensing mode for their interrupt instead of a level sensing mode.

If $PostIRQ is off, then the only way to know if data has arrived or not, is to use the serial port status command and check to see if data has arrived. Many times this is not even necessary since a product like a thermostat would typically just read the setpoint object every so often and use the latest value to make its decisions. See the serial port Status command in the *Device Transmitting and Receiving* section for more information.

SitePlayer will not interrupt the device if the data sent in is exactly the same as data that was already in the memory. This keeps people from flooding SitePlayer and the device with requests for changes which are already made. For example, setting a temperature on a thermostat repeatedly to 72 degrees would not cause an interrupt.

If $PostIRQ is off, then the only way to know if data has arrived or not, is to use the serial port status command and check to see if data has arrived. Many times this is not even necessary since a product like a thermostat will just read the setpoint every so often and use the latest value to make its decisions. See the serial port Status command in the *Serial Transmitting and Receiving* section for more information.

$Sitefile [filename]

When SiteLinker processes a directory (see $SitePath), to make a web site to be downloaded into SitePlayer, it first makes a binary image file. This binary image file, is then downloaded into the Flash memory of the SitePlayer device. The typical extension of these files ends in .SPB to be compatible with the SiteLinker program.

This image file is portable and can be sent to others. For example, if you have an image update to your product, you can send the customer an image file and they can download it into their SitePlayer without giving them all the details of the project.

```
$SiteFile "C:\Program Files\SitePlayer\image47.spb"
```

$SitePassword [password]

$SitePassword works in conjunction with the PasswordRequired file to protect all files that reside within are below a directory. If your root directory contains the file called PasswordRequired, then all the files within it would be protected by the password defined by $SitePassword.

```
$SitePassword "mypassword"
```

$Sitepath [path]

SitePath provides the SiteLinker program with the location of the data files which make up the SitePlayer Server. The path argument is the "root" path used for all file references. For example:

```
$Sitepath "C:\Program Files\SitePlayer\SP_Root"
```

processes all files in the demo subdirectory and makes them the root of the SitePlayer site. When the user requests the file http://xxx.xxx.xxx.xxx/index.htm they will be accessing the data obtained at C:\Program Files\SitePlayer\SP_Root\index.htm. Each project should have a different root directory; you will encounter errors when SiteLinker fails to match the SiteObjects within the HTML to the objects listed in the definition file. For convenience, you should create a file called INDEX.HTM since SitePlayerPC will attempt to load that if no filename is specified in your browser's address or location window.

Object Section

The Object section consists of simple variables that have a name, a type, a size, and an initial value. These will become SiteObjects you use to communicate between your device processor and the SitePlayer processor. Therefore, the object names you choose here must match the SiteObject names in your HTML pages, and the variable names in your device processor. They can be named using the alphanumeric characters A-Z, 0-9, and _. Object names can be up to 32 characters in length, and are not case sensitive. They must begin with a character A-Z and cannot conflict with the reserved symbols listed in the *Reserved Symbols* section.

To create an object you just need to make a command that looks like

```
MyObject db 5            ;comments
```

This creates an object called MyObject, which will be a byte with default value of 5.

If you are using graphical x/y coordinate submissions, the period is changed to an underscore:

`object.x` should be named `object_x` in SitePlayer.

The Object section of the Receiver example looks like this:

```
        org 0h
select  db 2                    ;in this example select=2 is FM
volume  db 0                    ;default the volume low
channel dw 921                  ;a word with the default channel of 92.1

        org 0FF15h
power   db 1                    ;default off, address is IO4, the green LED

       org 248                  ;decimal numbers are assumed when no h, b, o
AM_page         dbit 0          ;set status bit with AM page submission
FM_page         dbit 0          ;set status bit with FM page submission
Tape_page       dbit 0          ;set status bit with Tape page submission
CD_page         dbit 0          ;set status bit with CD page submission
Aux_page        dbit 0          ;set status bit with Aux page submission

        org 30h
xytest_x db 0                   ;x coordinate of xytest graphic
xytest_y db 0                   ;y coordinate of xytest graphic
```

Some of the types of objects you can use follow.

DBIT [value] – Define Bit

Defines a bit data type

```
Objectname       DBIT 1   ;defines a bit with an initial value of 1
```

DB [value] – Define Byte or String

Defines a byte or string data type. Use a number to define byte. Use quotes to define string. You may use hex codes (%HH) for special HTML characters: %20 = space

```
Objectname       DB 47h   ;defines a byte with an initial value of 47h
ObjectStr        DB “This is a string%0D%0A” ;defines an 18 character string
```

DHEX [value] – Define Byte

Defines a byte type. Objects values defined as DHEX will be displayed and received through web pages as hex values and not decimal values.

```
Objectname       DHEX 47h   ;defines a byte with an initial value of 47h
```

DW [value] – Define Word (16 bits)

Defines a word data type, integer, or 16 bits.

```
Objectname  DW 0FF12h ;defines a 16 bit word with an initial value of 0FF12 hex
```

DD [value] – Define Double (32 bits)

Defines a double word data type, long, or 32 bits.

```
Objectname  DD 2435A734h ;defines a long with an initial value of 2435A734 hex
```

DS [value] – Define Space (1 to 255 bytes)

Defines a data type depending on the space used. No initial value is specified. If 1 byte is defined, then the type is byte, 2 bytes then the type is integer, 4 bytes then the type is long, and more than 4 bytes, the type is string.

```
Objectname  DS 12   ;defines a 12 character string
Objectint   DS 2    ;defines an integer
```

ORG [value] – Define Location of Objects

Sets the memory location where the objects will be created.

```
        ORG 200h
Object1  db 0
Object2  db 0
```

Causes object 1 to be defined at location 200 hex in memory. Object2 will be created at location 201 hex.

Serial Port Output Object (defined in pcadef.inc)

The purpose of this object is to send data strings out SitePlayer's COM port. This is useful when you want SitePlayer to direct another processor or device, instead of being directed by another device. We will refer to this object as the COM object.

The COM object is like a string object, except instead of placing the data in object RAM, it sends the data out of the serial port's TX line. Using 8 bits, no parity, one stop bit. So now a link can send serial data out the COM port. Form input from a web page to the serial port output object will do the same. There is no handshaking. The serial data will be sent at the current baud rate. See the BAUD object for setting the baud rate in a form/link.

```
<a href="comtest.spi?com=Hello%20World">Hello!</a>
```

This HTML example sends out the string "Hello World" to the TX pin of the serial port when the "Hello!" link is pressed.

Any byte code can be sent from 0 to 255. If you are going to send non-printable characters you must to it this way.

```
<a href="comtest.spi?com=%01%03">Send Ctrl-A Ctrl-C</a>
```

Where the two digits after the percent sign are HEX characters. For a carriage return line feed combination, the following characters are used:

```
%0D%0A
```

If you use a multi-line text field in a form, carriage returns and line feeds are automatically generated and sent by the browser using "%xx".

The normal set of SitePlayer COM port commands (ReadObject, WriteObject…) will function. It is up to you to control any conflicts that might result from a readobject command getting data at the same time com port data is received from the com port object.

Baud Rate Object (defined in pcadef.inc)

The baud rate object consists of an integer that gets loaded into the timer register of the SitePlayer's UART to provide the correct count to generate a baud rate. The formula to calculate the number to send to the baud rate object is:

```
65536 - (1250000 / baudrate)
```

Some common numbers are:

Baud Rate	Number	Baud Rate	Number	Baud Rate	Number	Baud Rate	Number
110	54172	1200	64494	7200	65362	38400	65503
300	61369	2400	65015	9600	65406	57600	65514
600	63453	4800	65276	19200	65471	115200	65525

An example of a link that sets the baud rate to 9600 is:

```
<a href="comtest.spi?baud=65406">Click to set Baudrate to 9600</a>
```

Serial Peripheral Interface (SPI) Object

SPI is a powerful interface which allows high speed interaction with devices which include, DACs, ADCs, Output and Input Latches, motor controllers, and shift registers to name a few.

Hardware Pin Definition for SPI Object	
Pin	**Description**
IO0	SCK - Serial Clock
IO1	MOSI - Master Output Slave Input
IO2	MISO - Master Input Slave Output
IO3	Chip Select (optional)
IO4	Chip Select (optional)
IO5	Chip Select (optional)
IO6	Chip Select (optional)
IO7	Chip Select (optional)

With up to five SPI device chip selects, SitePlayer can provide interaction with many devices in a standalone mode. It is up to your link or form to lower the chip select and raise the chip select when you are done (see examples). Any data sent to the SPI object will cause an 8-bit SPI frame to be sent out MOSI and 8 bits to be received at the same time into MISO, MSB first.

The amount of data that is send/received is only dependent on the length of the data you send to the object. If you send 5 bytes you will receive 5 bytes. Whether the device uses all these bytes or sends you all these bytes will be dependent on the device, but SitePlayer will send them and receive them all for you. For example, the linear 24 bit A/D converter, LTC2415, (www.linear-tech.com) sends a 32 bit frame each time it is accessed. It requires no data from SitePlayer, but for SitePlayer to get the 4 bytes sent from the A/D converter, it must send out 4 bytes of zero.

Incoming data will be placed in an internal object that is pointed to by a single byte object called SPIaddr. This way you could gather data from 5 different devices into 5 different memory locations and send the whole thing back in one web page.

Here is what a web page link could look like for an SPI object:

```
<a href="spitest.spi?IO0=0&SPIaddr=%80&IO3=0&SPIBus=%47%98&IO3=1">SPI Test</a>
```

To break this link down into pieces:

`IO0=0` starts the SCK clock line at a zero state. SitePlayer will toggle the clock from this resting position up and then down for each bit shifted out. If the clock was started in a rising state (IO0=1) then SitePlayer would toggle the SCK pin down and then up for each bit. The choice of which way to start the clock, is dependent on your SPI device. The LTC2415 mentioned above, specifies that the clock to begin at a low state. Once the SCK pin has been set to a state, you do not have to keep setting it to the same state. It will return to this state. This way if you know you have 5 devices with all of them using SCK low, then you only need to set the IO0 pin a single time for all future accesses.

`SPIaddr=%80` means that the address that will be used to store the incoming SPI data will go into SitePlayer object memory at address 80h. Since SPIaddr is only one byte, you must send SPI data into the lower 256 bytes of object RAM inside of SitePlayer.

`IO3=0` means SitePlayer will take pin IO3 low - which becomes the Chip select for an SPI device.

`SPIBus=%47%98` means that two SPI 8 bit frames will be sent MSB first, and two 8 bit frames will be received into locations 80h and 81h (see SPIaddr above). 47 hex will go out first MSB first, and then 98 hex will go out MSB first.

`IO3=1` raises the chip select pin for the SPI device and the SPI transaction is done.

Since SPI devices typically send back a great deal of binary data which must be further processed, Visual Basic, C or Java can be used to take the data from the resulting objects and put it in a user readable form.

You may need residtor pullups or additional hardware on the various output pins of the SitePlayer depending on your application. Please refer to the Philips P89C51 document on the CD, SitePlayer or Philips web site for the current drive capability of SitePlayer's pins.

HalfSec object 0FF1Fh

This object is a byte that is decremented every 0.50135 seconds. This can be used as a timer by a device processor, or can be used to turn off display of web pages after a certain amount of time. See the ExitIF0 object modifier for more information.

```
^object:ExitIF0
```

ExitIf0 object modifier

If the byte object is zero, this command will stop the further display of a web page. If the object is non-zero, the page displays as normal. This command can be used for securing a web page. If a device processor determines that a person can look at a page, then the device processor can non-zero a location which will allow that page to be seen.

```
^object:ExitIf0
```

Used in conjunction with the HalfSec object, SitePlayer can automatically open access to pages for a certain amount of time without any further support from the device processor. To do

this you would set Halfsec to a number like 120 for one minute of access. On the pages you want to secure you would have this at the beginning of each page:

```
^HalfSec:ExitIf0
```

When the timer counts down to 0 then the pages cannot be displayed again until HalfSec is reset by the device processor.

UDPsend object 0FF1Eh

Writing any data to this object, results in SitePlayer formatting up a UDP packet and sending it as if the UDPsend command was received on the COM port. See *UDP Send and Receive* section for more UDP information.

$OutputOnly

If you place this command in front of a definition of an object, then the following objects become output only. This means that they cannot be used for entering data. In the case of a thermostat, you would want the locally generated temperature to be output only. You would not want anyone changing this data from a remote location. On the other hand, the setpoint you would specify as an input and an output variable.

```
$OutputOnly
temperature      db 72

$bidirectional
setpoint         db 72
```

These commands would make temperature an output only variable, and setpoint a bidirectional variable.

$Bidirectional

If you place this command in front of a definition of an object, then the following objects become both input and output.

```
$OutputOnly
temperature      db 72

$bidirectional
setpoint         db 72
```

These commands would make temperature an output only variable, and setpoint a bidirectional variable.

Export Section

The export section defines how and where any files will be exported during SitePlayer's SiteLinker process. This is useful for creating data files to be passed to assemblers, C compilers, Visual Basic, or to even make web pages based on the data from the SiteLinker.

Part of the export section of the receiver example looks like this:

```
;$ExportFormatFile contains format definitions for $ExportFile
$ExportFormatFile "C:\Program Files\SitePlayer\makeHTML.def"
```

```
;$ExportHeaderFile contains header information for $ExportFile
$ExportHeaderFile "C:\Program Files\SitePlayer\htmlheader.htm"

;$ExportFooterFile contains footer information for $ExportFile
$ExportFooterFile "C:\Program Files\SitePlayer\htmlfooter.htm"

;$ExportFile sets name of export file created from these definitions
$ExportFile "C:\Program Files\SitePlayer\SP_Root\receiver_export.htm"

;$Export command creates HTML $ExportFile and clears parameters
$Export

;Export definitions for a Visual Basic formatted file
$ExportFormatFile "C:\Program Files\SitePlayer\makevb.def"
$ExportFile "C:\Program Files\SitePlayer\rcvrdefs.bas"
$Export

;Export definitions for a C header file
$ExportFormatFile "C:\Program Files\SitePlayer\makec.def"
$ExportFile "C:\Program Files\SitePlayer\rcvrdefs.h"
$Export

;Export definitions for an Assembly file
$ExportFormatFile "C:\Program Files\SitePlayer\makeasm.def"
$ExportFile "C:\Program Files\SitePlayer\rcvrdefs.asm"
$Export
```

Multiple files may be exported. Each needs their own format file and $Export function called. The receiver example defines and creates HTML, Visual Basic, C, and Assembly files.

$ExportFormatFile [filename] none

Defines a file to be used by the $Export command.

The format file contains pointers to objects as the SiteLinker processes the objects. The following objects are defined for use during export:

^name – returns the name of the object
^address – returns the location of the object
^type – returns the type of the object
^direction – input, output or bidirectional
^default – returns the default value of the object
^secure – whether this variable is secure or not

For example if the following objects were defined,

```
      ORG 200h
Object1   db 0
Object2   dw 47h
```

a definition file with the following statements

```
#define ^name 0x^address /* ^type */
```

would result in a file containing this text

```
#define Object1 0x0200 /* byte */
#define Object2 0x0201 /* integer */
```

This could be used in a C program to locate the objects in SitePlayer without any typing. If you move objects around, the SiteLinker program will export a new file when you make the new flash binary file for SitePlayer.

The Receiver format file (makeHTML.def) creates HTML output with this definition:

```
^name = ^^^name<br>
```

Two uparrows, ^^, are necessary when you want one uparrow to be written into the export file.

$ExportFile [outputfilename]

Defines the file that will be used to output the objects created by the $ExportFormatFile. This file will only be created if there are no errors, and if the $Export function is executed.

You could create this HTML file from simple ExportHeaderFile, ExportFormatFile, and ExportFooterFile definitions:

```
<html>
<body>
power = ^power<br>
select = ^select<br>
volume = ^volume<br>
channel = ^channel<br>
</body>
</html>
```

$ExportHeaderFile [headerfilename]

Defines a file that will be copied to the beginning of the $ExportFile when the export process begins

```
$ExportHeaderFile “c:\definitions\header.htm”
```

Adds c:\definitions\header.htm to the beginning of the $ExportFile
Example contents of the header file might include the following:

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

$ExportFooterFile [footerfilename]

Defines a file that will be copied to the end of the $ExportFile when the export process completes. For example, this could be used to add an “</html>” at the end of a file.

```
$ExportFooterFile “c:\definitions\footer.htm”
```

Adds c:\definitions\footer.htm to the end of the $ExportFile
Example contents of the footer file might include the following:

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

$Export

Begins the process of creating the $ExportFile using the $ExportFormatFile file as a guide. The export variables will be cleared whenever the $Export command is encountered so that multiple formats can be exported from different export definitions.

```
$Export
```

2) Create Web Page Files

Web pages are the interface for your device through SitePlayer. SitePlayer does not require JAVA or Visual Basic Scripting to perform its live updates of data. You employ standard web authoring techniques to design the look and layout of your pages. Then, to make the pages active, you replace the static representations with the Object names you created in your SitePlayer Definition file. These are the SiteObjects that make SitePlayer so simple and your device interface so powerful and flexible. You may use SitePlayerPC to view and test your pages without connecting your device.

1. Build a standard web site using your favorite HTML authoring system; consult its documentation for web page development help.
2. Change the HTML code to make the data come alive by placing pointers to objects within the HTML data.

The SitePlayer Interface file (.SPI) and PasswordProtect files offer added control of your site.

SiteObjects in HTML

When you make an HTML file, you can make the data in it live by using the objects to modify the page. Objects are data variables that have a known data type, a size, and an initial value. They are the same objects that you defined earlier in your SitePlayer Definition file.

To use an object, you signal its use by an uparrow (“^”) followed by the name of the object and optional modifiers to the object. As an example, let us define an object called “flow” which is an integer, 2 bytes long, with an initial value of 0. If SitePlayer encounters the characters “^flow” when emitting a web page, it will substitute the value of flow at that instant for the pointer. So you could make a simple sentence in a web page:

```
The flow rate at this moment is ^flow
```

If the flow rate at that instant is 567 then the text returned by SitePlayer would be:

```
The flow rate at this moment is 567
```

SitePlayer converts the 2 bytes, to its decimal ASCII representation and emits it.

SiteObject Modifiers

Modifiers are available to allow certain types of manipulation of an object by the SitePlayer server as it emits the data. Legal modifier characters are 0-9, S, and P. These objects and modifiers only exist within the SitePlayer server and do not actually modify the HTML standard. When SitePlayer generates a page, it will act upon the them in the manner described in the following table.

Object Usage	Description of Action
^object	Displays the object
^object:n	Displays digit number n of the numeric object counting from the right towards the left, or character number n of a string object counting from the left to the right. Also S is allowed to return the sign of the object either space or "-" minus. A "P" returns "+" or "-" minus. An "M" returns "M" for minus or "P" for plus.
^object+n	Adding n to the numeric object and then displaying the result
^object-n	Subtracting n from the numeric object and then displaying the result
^object*n	Multiplies the numeric object by n and displays the result
^object/n	Divides the numeric object by n and displays the result
^object&n	Logically ANDs the numeric object with n and displays the result
^object|n	Logically ORs the numeric object with n and displays the result
^object~n	Logically XORs the numeric object with n and displays the result
^object#n	Logically ANDs the numeric object and n and displays CHECKED if the result is non-zero and nothing if zero
^object$n	If object = n then displays CHECKED otherwise nothing
^object'n	Obtain the nth bit of the object counting from the right 0 = the first bit

Selecting a Particular Digit of an Object

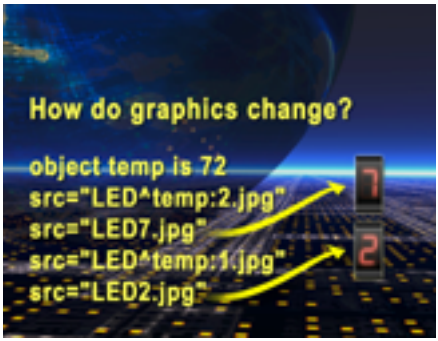
^flow:3 selects the third digit of flow, beginning from the right counting left. If flow is 567 then ^flow:3 becomes the ASCII character "5". Selecting digits is a powerful way to make simulated LED displays. Take the sample:

```
<img Src="/images/LED^flow:3.gif">
<img Src="/images/LED^flow:2.gif">
<img Src="/images/LED^flow:1.gif">
```

SitePlayer would return the strings

```
<img Src="/images/LED5.gif">
<img Src="/images/LED6.gif">
<img Src="/images/LED7.gif">
```

If you had LED graphics from 0 to 9, (which are included in your SitePlayer Developer's Kit), you can make live LED displays.



Performing Simple Math on an Object

^flow+5 adds 5 to the value of flow and emits it. In this example, ^flow+5 would be "572". These are the math operands: Add "+"; subtract "-"; multiply "*"; divide "/"; AND "&"; OR "|"; XOR "~".

Selecting a Particular Bit of an Object

^flow'13 selects bit 13 of the object counting bit zero as the least significant bit counting towards the left. ^flow'30 is not valid and will return a zero since flow is a 16 bit value. Bit numbers are valid from 0 to 31.

Avoiding Object Conflicts with Text

There are cases where you want to emit an object right next to some other text which could be confused with an object modifier. The uparrow ^ is like a double quote, used to stop parsing

whenever SitePlayer could get confused. Filenames are good examples where this occurs frequently. An example where ^flow equals 544:

`/^flow/32/file.xxx` returns `/17/file.xxx`

How does SitePlayer know that the /32 is a divisor for flow or a sub directory level? In fact you cannot tell by looking at the text. In this case SitePlayer assumes that you mean to divide flow by 32 unless you force the object to end with another uparrow.

`/^flow^/32/file.xxx` returns `/544/32/file.xxx`

The same problem occurs with digits running together.

`^flow+57856` returns `58400`
`^flow+5^7856` returns `5497856`

Two uparrows together, ^^, are necessary when you want to display a single uparrow, ^, in the browser display.

`SiteObject ^^flow = ^flow` returns `SiteObject ^flow = 544`

SitePlayer Web Pages

Create standard web pages to interact with your device. SitePlayer includes sample HTML files and graphic controls to help you create an attractive, active web site. The included samples also illustrate the SitePlayer Interface file, and SiteObject techniques. The main root of the SitePlayer device is determined from the $Sitepath definition when the site is assembled with SiteLinker. The entire web site structure must exist under the main root. Each project should have a different root directory; you will encounter errors when SiteLinker fails to match the SiteObjects within the .HTM, .HTML, .XML, and .SPI files, to the objects listed in the definition file. For convenience, you should create a file called INDEX.HTM since SitePlayer and SitePlayerPC will attempt to load that if no filename is specified in your browser's address or location window.

When designing your website, keep in mind that SitePlayer has a 48K limit. The size is displayed by SiteLinker after it makes the SPB binary image file. Every character, space, linefeed, carriage return, and filename will make a difference towards that total.

SitePlayer Interface File

The SitePlayer Interface file (.SPI) provides the method for sending data to SitePlayer from the web browser. It can be named any legal filename you choose. Use it in your web pages much like a Common Gateway Interface file (.CGI). Place it in hyperlinks with SitePlayer object names that you want to adjust or look up, or use it as the form action file name that will get submissions. It can also contain HTML instructions for redirecting the browser after it has linked or submitted to it. Password protect the SitePlayer Interface file's directory for access control security. Use SiteObjects in it for variable redirection.

Create SitePlayer Interface File

The sample.spi is a file that you can make which contains the next web page to display after the setpoint has been changed. Typically, this redirects the browser to right back to the page that they just came from with the new updated information. To cause a page to be redirected, the following command can be used in your sample.spi page:

```
HTTP/1.0 302 Found
Location: /URL
<cr><lf>
```

Where /URL is the location of the page you want to go to. It is critical that you have an extra blank line at the end of your SPI file or else this scheme will not work. You can refer to the definition of HTTP to find other possible commands for the server to send. If preferred, the SitePlayer Interface file will display by making it a web page. For variable redirection, try an entry like the one used in FI.SPI for the Form examples:

```
HTTP/1.0 302 Found
Location: /form^select.htm
<cr><lf>
```

Sending Data to SitePlayer from Links

Links can send data to SitePlayer with some careful formatting. A link with the destination:

```
<a href=”http://xxx.xxx.xxx.xxx/sample.spi?setpoint=73”> Sample</a>
```

would cause the SitePlayer at location xxx.xxx.xxx.xxx to change the setpoint to 73, assuming setpoint has been defined and it is a $bidirectional variable.

Here is an example text-only Receiver page. It demonstrates the link and SiteObject concepts with the following HTML:

```
<html>
<head>
<title>SitePlayer Text Audio Receiver</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
<!-- meta http-equiv="refresh" content="5" -->
</head>

<body bgcolor="#666666">
<h1 align="center"><font color="#FFFF00">SitePlayer<br>
  Audio Receiver Demo<br>
  Text Version</font><br>
  <br>
<font color="#00FF00">^channel:3^^channel:2.^channel:1 FM</font><br>
  The current volume is <font color="#00FF00">^volume</font><br>
  <br>
  Seek <a href="tr.spi?channel=^channel+2">Up</a>
              
  Seek <a href="tr.spi?channel=^channel-2">Down</a><br>
  Volume <a href="tr.spi?volume=0">0</a>
  <a href="tr.spi?volume=1">1</a>
  <a href="tr.spi?volume=2">2</a>
  <a href="tr.spi?volume=3">3</a>
  <a href="tr.spi?volume=4">4</a>
  <a href="tr.spi?volume=5">5</a>
  <a href="tr.spi?volume=6">6</a>
  <a href="tr.spi?volume=7">7</a><br>
</h1>
</body>
</html>
```

Sending Data to SitePlayer from Forms

SitePlayer uses the GET method of receiving form input. So you must tell your HTML editor that you are using GET instead of POST with the <FORM> tag.

```
<form method="get" action="sample.spi" name="">
```

When the form fields are submitted, they will be sent to SitePlayer with the sample.spi file. Form types such as Check Boxes, Radio Buttons, Select List/Menus, and Hidden fields are setup using the Object names defined in the SitePlayer Definition file.

```
<input type="hidden" name="power" value="0">
```

There are a few techniques to consider in dealing with the behaviors of the various form types. For instance, Check Boxes only send a data submission when they are checked, nothing gets sent when they are unchecked. Therefore, you should add a hidden field, as shown above, to make sure the 0 value gets submitted when you uncheck the following visible box.

```
<input type="checkbox" name="power" value="1">
```

That means one value (the hidden 0) will be submitted for the checkbox when unchecked and two values (the hidden 0 and the visible 1) will be submitted for it when checked. SitePlayer will accept the values in the order they are received so the default hidden field should occur first in your HTML. Additionally, you will want to know whether the box is currently checked or not. That is shown as “checked” or blank in the HTML:

```
<input type="checkbox" name="power" value="1" checked>
```

In order for the checkmark to become dynamic, you must replace “checked” with the SiteObject “^power#1” as further described in the SiteObject section:

```
<input type="checkbox" name="power" value="1" ^power#1>
```

If power is on, the word “checked” will be returned in place of “^power#1” and a checkmark will appear, otherwise, nothing will be returned and no checkmark will appear.

In the case of radio buttons, no hidden field is necessary but you still want to know if the button is checked or not. If the object has a binary value, the buttons will be returned as selected. Otherwise, you must place the “#” or “$” SiteObject within the radio button’s HTML tag:

```
<input type="radio" name="volume" value="0" ^volume$0>
```

Select List/Menus always submit something. If nothing is selected, the first option in the list will be sent. To ensure that your device object’s value is not changed unless you explicitly choose another selection, you should make the first option the current value by using SiteObjects:

```
<select name="channel" size="1">
    <option value="^channel">^channel:3^^channel:2.^channel:1</option>
    <option value="929">KWFM</option>
</select>
```

When the Select List/Menu is displayed, the first choice will show whatever the current channel is. Again, a later section explains SiteObjects in further detail.

When you define a form, you create a name for your data and particular style. The great thing about the web and SitePlayer is that SitePlayer does not care how the data gets to it as long as it is in the format that it likes. This process is better shown than to described since you must

feel the interaction. Please try out the different Audio Receiver demos that come with SitePlayer to see the various ways you can interact with a device. Here is a sample that demonstrates the demonstrates the form and SiteObject concepts:

```
<html>
<head>
<title>Untitled Document</title>
<meta http-equiv="Content-Type" content="text/html; charset=iso-8859-1">
</head>

<body bgcolor="#CCFFFF">
<form method="get" action="fi.spi" name="forminput">
  <h1>^channel:3^^channel:2.^channel:1</h1>
  <input type="hidden" name="power" value="0">
  <br>
  Power Off
  <input type="checkbox" name="power" value="1" ^power#1>
  <br>
  <br>
  Volume 0 through 7<br>
  <input type="radio" name="volume" value="0" ^volume$0>
  <input type="radio" name="volume" value="1" ^volume$1>
  <input type="radio" name="volume" value="2" ^volume$2>
  <input type="radio" name="volume" value="3" ^volume$3>
  <input type="radio" name="volume" value="4" ^volume$4>
  <input type="radio" name="volume" value="5" ^volume$5>
  <input type="radio" name="volume" value="6" ^volume$6>
  <input type="radio" name="volume" value="7" ^volume$7>
  <br>
  <br>
    Select your station:

  <select name="channel" size="1">
    <option value="^channel">^channel:3^^channel:2.^channel:1</option>
    <option value="929">KWFM</option>
    <option value="937">KRQQ</option>
    <option value="949">KMXZ</option>
    <option value="961">KLPX</option>
  </select>
    <br>
    <br>
    <input type="radio" name="select" value="1" ^select#1>
    AM<br>
    <input type="radio" name="select" value="2" ^select#2>
    FM<br>
    <input type="radio" name="select" value="4" ^select#4>
    Tape<br>
    <input type="radio" name="select" value="8" ^select#8>
    CD<br>
    <input type="radio" name="select" value="16" ^select#16>
    Aux<br>
    <br>
    <input type="submit" name="Submit" value="Submit">
    <br>
</form>
</body>
</html>
```

PasswordRequired File

Use the PasswordRequired file to provide security to any other files in the same folder. The file can be empty as long as it is named PasswordRequired. It also provides access control to your

device through the SitePlayer Interface file. That means no one can change your data without a password if the SitePlayer Interface file resides in a folder with a PasswordRequired file. *The PasswordRequired option is not currently available.*

3) Assemble and Download Binary File

The code that tells SitePlayer how to perform and what web pages to serve must be defined by the SitePlayer Definition file and assembled into the SitePlayer Binary image. This in turn, is downloaded to SitePlayer through your Ethernet connection. Once the binary image is in SitePlayer, it can serve your web pages and interact with your device. SiteLinker is the tool that assembles and downloads the binary image.

WARNING:
Do not perform serial port I/O to the SitePlayer during Ethernet downloading of new firmware, or web pages. If you must perform this action in the field, you need to have some customer interlock so that serial port activity stops during the updates. This could include resetting the device processor that talks to SitePlayer, or disconnecting the cable between the device processor and SitePlayer.

SiteLinker Operation

The purpose of SiteLinker is to create the SitePlayer binary file from SitePlayer Definition file and download it to SitePlayer over your Ethernet connection. You can also test the web site without connecting it to the device processor at this point using SitePlayerPC as discussed in the next chapter. Here are the main steps to using SiteLinker:

1. Start the SiteLinker program and use the File Menu to open either a SitePlayer Definition file (.SPD) or a previously assembled SitePlayer Binary file (.SPB).
2. Use the Configure Menu to set the SiteLinker Password and IP address to match either SitePlayer or SitePlayerPC.
3. Use the Download Menu to assemble and download the binary file. Definition files can be assembled with the Make Download F3, or Make and Download F5, commands. Binary files can be downloaded with the Download SitePlayer F4 command.

SiteLinker also performs other functions for your convenience. Use your Editor as defined in the Configure Menu to make changes to the SitePlayer Definition file before reassembling and downloading. Use Browser to open up your default web browser to your index.htm page. Use Calculator to display a calculator.

Commands that can be accessed with keyboard function keys such as F3, F4, F5, and F12, will display those keys next to their menu choices. All SiteLinker functions and displays can be accessed from the program window which consists of the following areas: Title Bar; Menu Bar; Assembly Progress Window; Status Bar.

Title Bar

The Title Bar spans the top of the SiteLinker program window. It includes the SiteLinker program name and the control boxes that minimize, maximize, and close the program.

Menu Bar

The Menu Bar displays SiteLinker's program control choices. The availability of the choices depends on whether a SitePlayer Definition file (.SPD), or Binary file (.SPB) is loaded. The Menu Bar lies just under the Title Bar and includes these items: File; Download; Configure; Editor; Browser; Calculator; About.

File Menu

The File Menu allows selection of the SitePlayer Definition file (.SPD), or Binary file (.SPB). You may use the Open command to navigate to a new file, or use one of the numbers 1, 2, 3, 4, of the previously loaded files. Keyboard key F12 is a shortcut to load file 1. The File Menu's Exit command stops and closes SiteLinker.

Download Menu

The Download Menu controls the creation and downloading of the SitePlayer Binary file. When a SitePlayer Definition file is loaded from the File Menu, the Make Download File F3 and Make and Download F5 choices become available. Make Download File F3 creates the SitePlayer Binary file from the SitePlayer Definition file. Make and Download F5 creates the SitePlayer Binary file and also downloads it to SitePlayer using the IP Address for Download as defined in the Configure Menu. When a SitePlayer Binary file is loaded from the File Menu, the Download SitePlayer F4 choice becomes available. Download SitePlayer F4 downloads the binary file using the IP Address for Download as defined in the Configure Menu. SitePlayerPC must be running if you are downloading to it.

Configure Menu

The Configure Menu sets the SiteLinker program parameters. IP Address for Download tells SiteLinker where to send the SitePlayer Binary file. This IP Address must match that of the SitePlayer device as set with the $InitialIP definition or by the DHCP server when $DHCP is on. When using SitePlayerPC, this address must match your PC's IP address or be configured as localhost at 127.0.0.1. Port Address for Download sets the SitePlayer port address so routers can be configured for multiple SitePlayers to be downloaded from a single IP address. Download Password sets the password that SiteLinker will send to SitePlayer when downloading. This password must match the one contained in the previously downloaded file as set with the $DownloadPassword definition. The Editor choice tells SiteLinker the filename of the editor you want to use on the Definition file when the Editor Menu button is pressed. Silent Errors suppresses error message boxes when it is checked. If an error then occurs during the Make or Download process, the Assembly Progress window background will still turn blue. The IP list displays the last several IP addresses used for easy access to multiple devices. Clear IP List removes all the entries from the list. Reset Default IP and Port to 192.168.1.250:26481.

Editor Menu

The Editor Menu does not have any choices beneath it. Once clicked, it displays the currently selected SitePlayer Definition file in your editor as defined in Configure Editor menu section. SitePlayer Binary files are not editable.

Browser Menu

The Browser Menu does not have any choices beneath it. Once clicked, it will open your default browser to the IP address defined under the Configure Menu to display your index.htm page. This IP address is also shown in the Ethernet Address window of the Status Bar. If your location is using a proxy server, you may have to add SitePlayer's address to your browser's proxy exception list.

Calculator Menu

The Calculator Menu does not have any choices beneath it. Once clicked, it will attempt to open the calculator. This will not function properly if it cannot find the CALC.EXE file.

About Menu

The About Menu shows the SiteLinker version and build number. This can be useful when troubleshooting.

Assembly Progress Window

The Assembly Progress window occupies the main body of the SiteLinker program. It provides feedback during the assembly process once the Make Download File F3, or Make and Download F5, commands are issued. It displays the start time and date of the process then lists which files are being used or exported. It tells how many objects entries there are, and how much flash memory is used before ending with the name of the SitePlayer Binary file created. If any errors are encountered, the background of the Assembly Progress Window will turn blue.

Progress Bar

The Progress Bar is activated when you begin a download. It indicates the percentage of download completion as the blue bars move from left to right.

Status Bar

The Status Bar goes across the bottom of the SiteLinker program window. From left to right, the five boxes within it display the following information: Definition Filename; Site Filename; Errors; Ethernet/Port Address; Download Progress. The Definition filename is the currently loaded SitePlayer Definition file. The Site filename is the currently loaded or generated SitePlayer Binary file. The Errors display if any are encountered during the assembly or download phase. The Ethernet and Port Addresses are shown as defined under the Configure Menu. This is where the binary file will be downloaded when the Download SitePlayer F4 or Make and Download F5 commands are issued. Download Progress is displayed in the last Status Bar window once the download process has been initiated.

4) Surf SitePlayer

Once the SitePlayer Binary image has been assembled and downloaded, it is time to surf SitePlayer with a web browser pointed to SitePlayer's IP address. The Browser command in both SitePlayerPC and SiteLinker offers a shortcut to your default web browser. The browser is the interface for your device. Since the look and performance of your web pages is so critical, you will probably need to make changes and recheck them often. SitePlayerPC may speed the process by using your computer to emulate SitePlayer. At some point though, you will use an Ethernet connection to communicate with the SitePlayer module.

When browsing either SitePlayer or SitePlayerPC, you may need to make some adjustments to your web browser's Internet Options or Preferences. You can adjust the cache so that it does not refresh on every visit to make pages with graphics reload faster and cleaner. If you are on a network with a proxy server, you will either want to disable the proxy, bypass local addresses, or add 'localhost' and SitePlayer's IP address to the exception list.

SitePlayerPC Operation

The purpose of SitePlayerPC is to emulate the SitePlayer chip on your PC. This allows faster testing of the SitePlayer web site. You can skip the step of downloading to the SitePlayer chip, and/or you can skip connecting SitePlayerPC to the device processor. That means you can modify the initial values of an object and try out a SitePlayer site without writing any device code. For large projects where the graphics and web layout is handled by a different team than the device code, this makes it invaluable for development. Once the teams agree on the objects, each can go about their tasks independently.

1. Start the SitePlayerPC program and use the File Menu to open a SitePlayer Binary file (.SPB). That will start the SitePlayerPC server.
2. Use the Comm Menu to select the computer COM Port your device is connected to. If no device is connected, make sure no COM Port is selected.
3. Use the Browser command to load your SitePlayer site into your web browser. Test your site's functionality, make changes to the site, and Reload the binary image.

Be sure that any other servers on your computer, such as Microsoft Personal Web Server, are disabled before running SitePlayerPC.

Commands that can be accessed with keyboard function keys such as F12, will display those keys next to their menu choices. SitePlayerPC program consists of the following areas: Title Bar; Menu Bar; Status Displays; Status Bar.

Title Bar

The Title Bar spans the top of the SitePlayerPC program window. It includes the SitePlayerPC program name and the control boxes that minimize, maximize, and close the program.

Menu Bar

The Menu Bar displays SitePlayerPC's program control choices. The availability of the choices depends on whether a SitePlayer Binary file (.SPB) is loaded. The Menu Bar lies just under the Title Bar and includes these items: File; Browser; Comm; Reload; About.

File Menu

The File Menu allows selection of the SitePlayer Binary file (.SPB). You may use the Open command to navigate to a new file, or one of the numbers 1, 2, 3, 4, of the previously loaded files. Keyboard key F12 is a shortcut to load file 1. The File Menu's Exit command stops and closes SitePlayerPC.

Browser Menu

The Browser Menu does not have any choices beneath it. Once clicked, it will open your default browser to http://localhost which will normally display your index.htm page. If your location is using a proxy server, you may have to add localhost to your browser's proxy exception list as shown. The Browser Menu choice is not available until a SitePlayer Binary file is loaded.

Comm Menu

The Comm Menu selects which of your computer's serial communications ports you will use between your device processor and SitePlayerPC. Only your unused existing COM Ports will be available from the possible COM1, COM2, COM3, and COM4 selections. Once selected, the port will appear with a checkmark beside it. Click it again to close the port and remove the checkmark. Do not select any ports if your testing does not include a device processor. SitePlayerPC can still serve the web pages.

Reload Menu

The Reload Menu has no choices beneath it. Once clicked, it reloads the SitePlayer Binary file so that any changes made to it will be reflected on the pages served.

About Menu

The About Menu shows the SitePlayerPC version and build number. This can be useful when troubleshooting.

Status Displays

The Status Displays occupy the main body of the SitePlayerPC program window. When a Binary file is loaded so that the SitePlayerPC server is started, the Status Displays become

active. The Status Displays include these three information windows: My Address; Last Visitor; Access Count.

My Address:

The My Address Window shows your IP address which is the number that can be typed into a web browser's *Address* or *Location* field to reach your SitePlayer. If multiple TCP/IP adapters are installed on your computer, this will list the first one, even though another one may be connected to your SitePlayer.

Last Visitor:

Whenever anyone accesses your SitePlayerPC, the Last Visitor status display changes to show their IP address.

Access Count:

The Access Count displays the number of file requests that SitePlayerPC has received. When someone accesses your site, the Access Count changes to show the time and new number of accesses to your site. Since the count is for every file request, a single page will change the access count by the number of different images or other elements that it calls.

Status Bar

The Status Bar goes across the bottom of the SitePlayerPC program window. The left side displays the SitePlayer Binary file that is loaded. The right side displays the End time when SitePlayerPC will stop serving files. SitePlayerPC must be closed and restarted once the End time has been reached.

Ethernet Connection

SitePlayer provides your device with a web interface through its Ethernet connection. Installing a common Ethernet network card into your computer will let it talk to SitePlayer. For development, you may even want two Ethernet cards, one for SitePlayer and one for your local network to minimize traffic. You will need to know SitePlayer's IP address in order to view its web pages. This may be determined by the SitePlayer Definition file's $InitialIP function or by a DHCP server when $DHCP in on. You can use the included SitePlayer Serial Port Tester if you need to find SitePlayer's IP address after a DHCP server has reassigned it. Both SitePlayer and your PC should be connected to an Ethernet hub. If you want to connect the PC directly to SitePlayer, you will have to use a crossover X cable designed for connecting two hubs to each other.

5) Serial Transmitting and Receiving

The process of transmitting an object to SitePlayer is relatively easy. A serial port is all you need to perform the transfer. If your device does not have a serial port, you can "bit bang" a stream of bits out a pin of your device. Code for your device typically requires less than 128 bytes in an 8051 or a processor that has a serial port - or typically no more than 256 bytes in a processor without a serial port. The baud rate is set by default whenever SitePlayer resets to 9600. Standard baud rates of 115,200 to 110 baud are available using the ComParams command. Custom baud rates are also available.

******* WARNING *****:**
Do not perform serial port I/O to the SitePlayer during Ethernet downloading of new firmware, or web pages. If you must perform this action in the field, you need to have some customer interlock so that serial port activity stops during the updates. This could include resetting the device processor that talks to SitePlayer, or disconnecting the cable between the device processor and SitePlayer.

Object Packets

An object packet looks like this:

Command Byte	Address 1 or 2 bytes	Data 0 to 16 bytes

A single byte command is used to tell SitePlayer what you want to do. The next one or two bytes provide the address of the object, and then from 0 to 16 bytes of data.

Serial Commands

The command byte is made up of two sections. The top four bits of the command byte select one of 16 functions to be performed. The bottom 4 bits provide a count of bytes that are going to be sent or received by the device.

Command Byte							
Command Value				**Number of bytes sent/requested**			
Up to 16 different commands				1 to 16 bytes (0=1, 1=2… 15=16)			
D7	D6	D5	D4	D3	D2	D1	D0

SitePlayer Serial Commands		
Command	**Command Byte**	**Description**
NOP	00h	Do Nothing, Recommended initializing command
Status	10h	Return Status of SitePlayer
Reset	20h	Perform a Watchdog Reset
ComParams	33h	Sets Baud Rate and UARTdelay
UDPSend	50h	Sends a UDP Packet
Read	0C0h	Read Object from SitePlayer
Write	80h	Write Object to SitePlayer
ReadX	0D0h	Read Using Extended Two Byte Addressing
WriteX	90h	Write Using Extended Two Byte Addressing
ReadBit	0E0h	Read a bit variable, One byte Address
WriteBit	0A0h	Write a bit variable, One byte Address
ToggleBit	0B0h	Toggles a bit variable, One byte Address

NOP

Performs no operation. It is a byte of all zeroes, or null. A device could come up without knowing whether the SitePlayer is ready or not for data. The device can send 20 bytes of nulls, and it can be assured that SitePlayer will have ended a maximum length message (command+2 byte address+16 byte write) and be ready for a new command. It is recommended that communications with SitePlayer be initialized with the NOP command.

Reset

Performs a warm boot of the SitePlayer device. Essentially it forces SitePlayer into a loop with interrupts turned off and the watchdog timer on. Eventually, SitePlayer will internally reset and start fresh. This assumes SitePlayer is alive enough to receive characters. There is always the reset pin which can be computer controlled.

Status

Returns status in a byte of data. To read SitePlayer's Status, send the command byte 10h and receive one byte. Once the status byte has been sent it is automatically cleared.

Status Byte Returned							
Submit	Bit 254	Bit 253	Bit 252	Bit 251	Bit 250	Bit 249	Bit 248
D7	D6	D5	D4	D3	D2	D1	D0

The top bit is set when form data has been submitted into the SitePlayer device regardless of whether an object has been modified. The other bits can be selectively set by your web pages to help streamline processing. First, create an object at one of the status bit addresses,

```
        org 248
AM_page         dbit 0          ;set status bit with AM page submission
```

Then, submit it in a hidden field from your web page.

```
<input type="hidden" name="AM_page" value="1">
```

Whenever you check the status, you can determine whether any AM data was sent and process accordingly.

ComParams

Sets the serial baud rate and a response delay. A response delay is needed when a device processor needs extra time to switch from a transmit mode to a receive mode in order to not miss characters. Whenever SitePlayer is reset, it will default to 9600 baud and 300 microseconds delay. The maximum response delay is 19.66 milliseconds. You must use the ComParams command whenever you want modify either of the parameters. The command requires four bytes. The first two bytes set the baud rate, low byte first, then high byte using this equation:

```
65536 - (1250000 / baudrate)
```

To set 115200 baud

```
65536 – (1250000 / 115200) = 65525 = 0FFF5h
```

The second two bytes for response delay are determined using this equation:

```
65536 – (timeinmicroseconds * 3.333333)
```

To set a 1000 microsecond delay

```
65536 – (1000 * 3.333333) = 62203 = 0F2F6h
```

This is the complete ComParams command for 115200 baud and 1000 microsecond delay:

```
33h OF5h 0FFh 0F6h 0F2h
```

Write or WriteX

Sends objects from the device to SitePlayer. Write has a single byte of address covering object addresses of 0 to 255 (00FFh), WriteX has an eXtended address of two bytes or 0 to 65535 (0FFFFh). Various models of SitePlayer have various object memory block sizes. The smallest SitePlayer has 768 bytes of objects covering addresses 0 to 2FFh. The Write or WriteX command also contains the count of bytes to send from 1 to 16 within the command byte in the bottom 4 bits. Zero meaning one byte and 15 meaning 16 bytes. As an example, to send two bytes of the number 1023 (3FFh) to object location 47h the Write command is used:

80h+2-1=81h	47h	0FFh	03h

Since the value of Write is 80h by adding the number 2 to it for the number of bytes you are sending then subtracting 1 which is the character offset, makes the command byte 81h. Lets look at it more closely:

Command Value	Byte Count (0=1, 1=2,…15=16)
8	1

Data is sent low order byte first. 1023 is 3FF hex, or 3 * 256 + 255. The data could also have been sent as a five byte extended command:

WriteX+2-1	47h	00h	0FFh	03h

Addresses are also sent low order byte first.

Read or ReadX

Receives objects from SitePlayer either from data input via form input, locally generated data, such as IP addresses, or previous objects sent to SitePlayer with Write or WriteX commands. Similar in format to the Write and WriteX commands, Read or ReadX have a 1 to 16 character count associated with them in the lower 4 bits of the command and a one byte or two byte. To read a two byte object at location 47h the following command could be used:

Read+2-1	47h

SitePlayer would respond with the two bytes from location 47h. Low order byte first. In other words, location 47h would be sent and then location 48h would be sent.

UDPsend

Instructs the device processor to send a UDP message to any MAC/IP address combination. This provides flexibility for a SitePlayer enabled device to send messages on a broadcast basis or to a specific computer system. Uses for this command include alarm conditions, data streaming, or periodic transmission of parameters without the need for polling by a remote computer system. See *UPD Send and Receive* section for more information.

Reading, Writing, and Toggling Bit Variables

SitePlayer also has bit variables, which are good for checkboxes, radio buttons, and the state of relays or switches. Instead of burdening the device with calculating where a bit lives within the object system, SitePlayer has 256 bit variables built into it. For example if you had to change the bit in the device without these variables, you would have to read the byte, modify the bit you wanted, and then send the byte back. What would happen if the user on a browser changed one of the bit variables in that byte before you had a chance to write it back? You would possibly have a major disaster. So SitePlayer performs the read, modify, write for you with interrupts off, so there can be no problems of this kind. Physically, bit variables live at byte addresses 02E0h to 02FFh in the object space. SiteLinker automatically allocates 32 bytes of memory in this area for bit objects. If bit objects are not used, then this memory is free to be used for regular objects. The bit commands are used the same way as Read and Write, except the single byte address is converted to a bit address with bit 0 living at the LSB of 02E0h, and bit 255 living at the MSB of address 02FFh. Read returns a single byte of 1 or 0 depending on the state of the bit at the bit location. Write takes an argument of non-zero or 0 and sets the bit accordingly. ToggleBit takes a single byte argument, and an address and inverts the bit at that address. The single byte argument in ToggleBit is just a place holder and can be any value, but it must be provided.

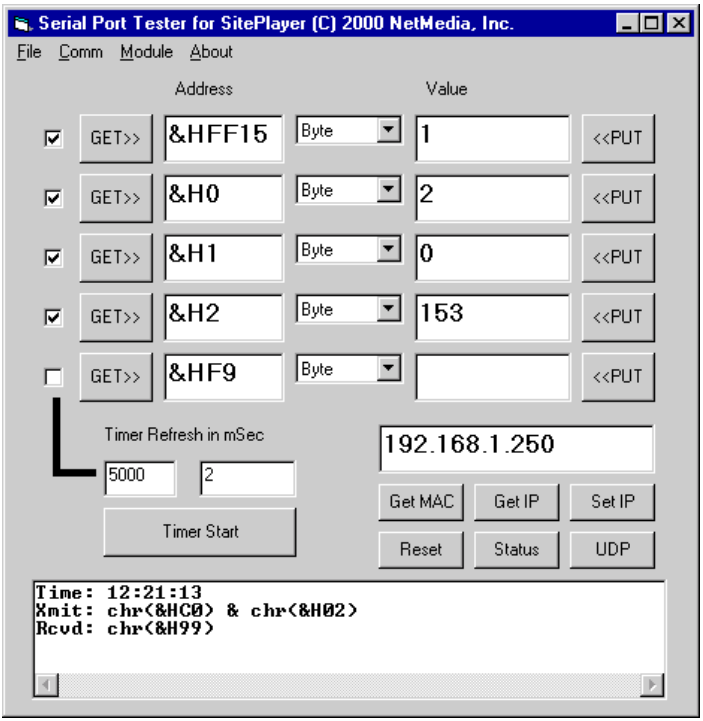
Sending More than One Object per Command

SitePlayer allows you to transmit 1 to 16 bytes per write command, or receive 1 to 16 bytes per read command. This means that you could transfer sixteen, one byte objects in a single command, or eight, two byte objects. This can cut down on the overhead of object transmissions, but can also lead to some possible complexity in programming. If you have a great number of single byte objects it can be as much as three to four times faster to send them with a multiple object command.

Serial Port Test Program

The SitePlayer Serial Port Test Program, SitePlayerSerialDemo.exe, is included with the SitePlayer software. It offers PC to SitePlayer serial communication for testing and development purposes. It also provides a means of getting or setting SitePlayer's IP address whenever it is unavailable through Ethernet. If necessary, change your PC COM port to match SitePlayer: 9600 baud; 8 data bits; No parity; 1 Stop bit.

1. Connect one end of the serial cable to the SitePlayer Development Board Serial Port J9.
2. Connect the other end of the serial cable to the PC's serial port.
3. Start the SitePlayer Serial Port Tester program.
4. Select the PC COM port under Comm Menu.
5. Choose functions such as Get IP Address or Set IP Address.
6. Enter hexadecimal memory address and data type to GET and PUT values.
7. Send UDP messages



Title Bar

The Title Bar spans the top of the the SitePlayer Serial Port Tester program window. It includes the SitePlayer Serial Port Tester program name and the control boxes that minimize, maximize, and close the program.

Menu Bar

The Menu Bar displays the SitePlayer Serial Port Tester program's control choices. The Menu Bar lies just under the Title Bar and includes these items: File; Comm; Module; About.

File Menu

The File Menu's Exit command stops and closes the program.

Comm Menu

The Comm Menu selects which of your computer's serial communications ports you will use to talk to SitePlayer. Only your unused existing COM Ports will be available from the possible COM1, COM2, COM3, and COM4 selections. Once selected, the port will appear with a checkmark beside it. Click it again to close the port and remove the checkmark.

Module Menu

The Module Menu has choices for Resetting or Rescuing the SitePlayer module. Reset is the same as the software and hardware Reset Buttons, it starts SitePlayer over again. Rescue will erase your website from SitePlayer. It is useful when SitePlayer has been setup in a way that makes it otherwise inaccessible. For instance, Rescue will let you download again after an unknown password is set.

About Menu

The About Menu shows the program's version and build number. This can be useful when troubleshooting.

Timer Functions

The Timer Functions allow you to poll SitePlayer at specific intervals.

Check Boxes

Place a checkmark next to any addresses you want polled. When the Timer is started, the program will get those values.

Refresh Timer

The Refresh Timer sets the polling interval in milliseconds. To set a 1 second interval, type 1000 into the Refresh Timer window.

Refresh Counter

The Refresh Counter tells how many times SitePlayer has been polled.

Timer Start/Stop Button

The Timer Start/Stop Button starts or stops the timer. Its function and text toggles according to the timer's current state. When the timer is already started, the text will say Timer Stop and pressing it will stop the timer.

Get/Put Buttons

The Get/Put Buttons are located beside the Address and Value windows. The Get Button retrieves the value at SitePlayer's hex address listed in the adjoining Address Window and places it in the corresponding Value Window. The Put Button sends the value from the adjoining Value Window to SitePlayer's hex address listed in the corresponding Address Window.

Address/Type/Value Windows

The Address/Type/Value Windows correspond to the Get and Put Buttons in the same row.

Address Window

The Address Window lists the hex memory address that you want to put or get values to or from.

Type Window

The Type Window selects what type of data is located in the memory address.

Value Window

The Value Window displays the value of the data located in the memory address when you press the Get Button. You may type in a new value and send it to SitePlayer using the Put Button.

Get MAC Address

The Get MAC button retrieves SitePlayer's MAC address and displays it in the Address/Status window. The MAC address is assigned by NetMedia and cannot be changed. You may need to use it when configuring your network to recognize a SitePlayer device.

Get/Set IP Address

You may retrieve SitePlayer's current IP address or send it a new one.

Get IP Button

The Get IP Button retrieves SitePlayer's current IP address and displays it in the Address/Status Window. This may be necessary after a DHCP server has reassigned it.

Set IP Button

The Set IP Button sets SitePlayer's IP address to the one displayed in the Address/Status Window. This may be necessary when SitePlayer is set to an IP address that is inaccessible to your Ethernet network's TCP/IP subnet mask.

Address/Status Window

The Address/Status Window displays the address retrieved by the Get MAC and Get IP Buttons or sent by the Set IP Button. It also displays the state of the Status byte when the Status Button is pressed.

Reset/Status Buttons

The Reset and Status Buttons are located below the Get/Set IP Buttons. The Reset Button is the same as the Reset menu item, it starts SitePlayer over again. The Status Button checks the state of the Status byte. When SitePlayer has received a submission, the byte will read 10000000b in the Address/Status Window. Once SitePlayer status has been polled, it will read 00000000b. The other bits, 248 – 254, can be individually set with SiteObjects to help streamline data processing.

UDP Button

The UDP Button is used to test SitePlayer's UDPsend function. When pressed, it instructs SitePlayer to send a UDP message. See *UDP Send and Receive* for more information.

Serial Command Window

The top row of the bottom window displays the time of the last serial transmission. The second row echos the serial command that the Visual Basic Serial Test program sent to SitePlayer. The third row shows the response received from SitePlayer in Visual Basic format. These serial commands can help illustrate how to communicate with SitePlayer.

SitePlayer Serial OCX for Visual Basic

This information is provided as a service for users to experiment with communications between SitePlayer and Visual Basic using SPControl.ocx which is located in the SitePlayer install directory. <u>These routines are unsupported.</u>

Settings

This sets the Com port parameters. Default is 9600,n,8,1 Please See Visual Basic's MSCOMM documentation for more information on serial port settings.

```
SP1.Settings = "9600,n,8,1"
```

PortOpen

PortOpen opens the Com port exclusively for SitePlayer communications. Please see PortOpen in Visual Basic's MSCOMM documention for more information.

```
SP1.PortOpen = True  'opens the port
SP1.PortOpen = False 'closes the port
```

CommPort

CommPort selects the serial communications port used for the control. Com1 = 1…Com4=4. This property must be set before the PortOpen command is executed. Please see CommPort in Visual Basic's MSCOMM documention for more information.

```
SP1.CommPort = 1
```

Init_Object

This command sends 20 bytes of nulls to the Comm port to initialize a SitePlayer.

```
Call SP1.InitObject
```

Status(ByRef b As Byte)

The Status method allows you to check on the status of a SitePlayer. The argument returns the status byte.

```
Dim b as byte
Call SP1.Status(b)
If b = 0 then
    ‘no form data has been sent
else
    ‘form data has been sent
end if
```

ReadObject(ByVal address As Long, ByRef data As Variant)

The ReadObject method allows you to receive data from your VB program to a SitePlayer object. Using the address you specify and the data, the method figures out how many bytes to receive automatically. If data is a string, then this command will only receive up to16 bytes of the string.

```
Dim I as integer
Dim B as byte
Dim L as long

Call SP1.ReadObject(27,I)
Call SP1.ReadObject(29,B)
Call SP1.ReadObject(30,L)
```

WriteObject(ByVal address As Long, ByVal data As Variant)

The WriteObject method allows you to send data from your VB program to a SitePlayer object. Using the address you specify and the data, the method figures out how many bytes to send automatically. If data is a string, then this command will only send the first 16 bytes of the string.

```
Dim I as integer
Dim B as byte
Call SP1.WriteObject(27,I)
Call SP1.WriteObject(29,B)
```

ReadBit(ByVal address As Byte, ByRef data As Byte)

WriteBit(ByVal address As Byte, ByRef data As Byte)

ToggleBit(ByVal address As Byte)

The Bit methods allow you to receive, send or toggle bit data from your VB program to a SitePlayer bit object. Addresses are from 0 to 255 only.

```
Dim B as byte

Call SP1.ReadBit(229,B)
Call SP1.WriteBit(229,0)
```

```
Call SP1.ToggleBit(230)
```

BaudSet(ByVal baud As Long, ByVal delay As Long)

The BaudSet command allows you to change the baud rate of a SitePlayer. Beware that immediately after the command, the baud rate is changed. You would then have to close the comm port with the PortOpen method and then issue a change in your settings and then reopen the port in order to talk to SitePlayer again.

Please refer to the SitePlayer Serial CommParams command, for the correct value calculations of the BaudSet parameters.

Regenerate

Essentially erases all web pages in SitePlayer and returns the device to a initial state.

Reset

Performs a watchdog reset within the SitePlayer device.

IP_to_Long(ByVal IPaddress As String) As Long

Performs a frendly conversion of an IP address string to a long number for sending to SitePlayer.

Long_to_IP(ByVal l As Long) As String

Performs a friendly conversion from a long number to an IP address string for displaying to a user.

6) UDP Send and Receive

SitePlayer can send and receive UDP packets through its Ethernet connection. It is instructed to broadcast with the UDPsend serial command and object. You may use the UDPsendtest program for verifying the UDP packets. The factory UDP settings can be viewed in the UDPsend_def.inc example include file.

UDPsend (serial command)

Instructs the device processor to send a UDP message to any MAC/IP address combination. This provides flexibility for a SitePlayer enabled device to send messages on a broadcast basis or to a specific computer system. Uses for this command include alarm conditions, data streaming, or periodic transmission of parameters without the need for polling by a remote computer system.

Special objects in fixed locations within the SitePlayer define the following parameters for the UDP packet:

```
        ORG 02D0h
DestinationMACaddress:    ds 6     ;6 bytes of MAC address
DestinationIPaddress:     ds 4     ;4 bytes of IP address
DestinationPort:          dw 0     ;2 bytes of destination port address
UDPstartAddress:          dw 0     ;Starting address of object(s) to send
UDPdataCount:             dw 0     ;Count of bytes to send 1-768
```

The filling of these objects can come from the device processor, or they can be set from a form. Using a form to set these objects allow a user to specify in the field where the UDP packets should be sent by using a web page to set up this information.

Once the objects are filled with the correct information, a single byte command is sent from the device processor over the serial port to generate the sending of a UDP packet. SitePlayer schedules in this UDP packet request with any other Ethernet traffic that it is currently working on, and then transmits the packet. The **UDPsend** command marks a flag within SitePlayer to send a UDP packet. It may be a few milliseconds later before SitePlayer has the time, or the free transmit space within the Ethernet controller to actually send the UDP packet.

NOTE: There is not necessarily a correspondence between the number of **UDPsend** commands and the number of UDP packets generated. Depending on the workload of SitePlayer at the time of reception of the UDPsend command. There may be a large transmit in progress and incoming requests for web pages at the same time. Since UDPsend, simply sets a flag for a UDP packet to be generated, the flag may still be set from a previous UDPsend command that has not completed.

Sending a UDP broadcast message

To send a UDP broadcast to multiple computers simultaneously on your local network, you would set the following objects to the following values:

```
DestinationMACaddress = FF-FF-FF-FF-FF-FF
DestinationIPaddress = xxx.xxx.xxx.255
```

Where xxx.xxx.xxx is the first 3 numbers of the local network that you wish to send the broadcast.

You can also send to all local networks by setting DestinationIPaddress to 255.255.255.255. Most software packages will accept and forward a broadcast to this address, into your applications software, but you should check your installation to make sure.

Sending UDP message to specific computer within local network

You must first get the MAC address of the Ethernet card of the computer you wish to communicate with. The easiest way to do this is to Ping the computer and then look at the ARP table to see what the MAC address is.

```
C>PING 192.100.100.164
C>ARP –A

Interface: 192.100.100.137 on Interface 0x1000002
  Internet Address      Physical Address      Type
  192.100.100.164       00-40-05-37-54-32     dynamic
```

You would then set the SitePlayer objects to the following numbers to send data to this computer:

```
DestinationMACaddress = 00-40-05-37-54-32
DestinationIPaddress = 192.100.100.164
```

Sending UDP message to specific computer outside local network

You must first get the MAC address of the Ethernet card of the gateway to the outside world. The easiest way to accomplish this is by Pinging the gateway computer. Then look at the ARP table to see what its MAC address is. In this example we will assume 192.100.100.1 is the local gateway computer (the computer SitePlayer is connected to) and 204.71.200.67 is the number of the remote computer we wish to UDP the packet.

```
C>PING 192.100.100.1
C>ARP –A

Interface: 192.100.100.137 on Interface 0x1000002
  Internet Address      Physical Address      Type
  192.100.100.1         00-40-05-37-94-32     dynamic
```

You would then set the SitePlayer objects to the following numbers to send data to this computer:

```
DestinationMACaddress = 00-40-05-37-94-32
DestinationIPaddress = 204.71.200.67
```

UDPsend object 0FF1Eh

Writing any data to this object, results in SitePlayer formatting up a UDP packet and sending it as if the UDPsend serial command was received on the COM port.

UDP receive function

SitePlayer can receive a packet on UDP port 26482 which will allow a remote processor or device, the ability to send information to SitePlayer's objects in real time.

The UDP packet sent is made up of one or more commands. Each command has the following format:

UDP Receive Command		
Name	**Size**	**Description**
Number of bytes	1 Byte	Number of bytes in the data
Compliment of number of bytes	1 Byte	Ones compliment of the number of bytes in the data
Data Address	2 Bytes	Pointer to object address in SitePlayer. Lower byte first
Data Value(s)	1…n	The data bytes

Commands can be repeated within a packet up to the limit of an Ethernet packet size. The special objects up at the 0FFxxh range can also be written to remotely. This allows you to do some useful things like perform an SPI object function, or serial character send, or manipulate the I/O port.

To end a UDP packet you MUST place two zero bytes at the end of the data.

UDP receive packet example:

Send 2 bytes (56h and 98h) to object address 0105h
```
02h,0FDh,05h,01h,56h,98h
```

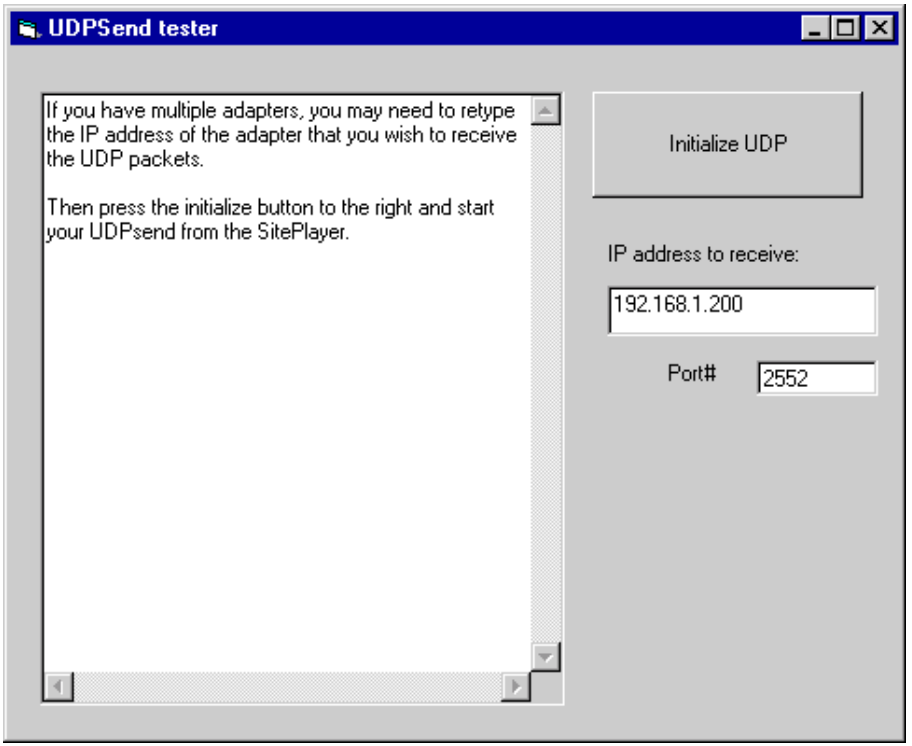
Send 1 byte (26h) to object address 0000h
```
01h,0FEh,00h,00h,26h
```

Send 1 byte (00h) to object address 0010h
```
01h,0FEh,10h,00h,00h
```

Total Packet:
num,comp,address,data...
```
02h,0FDh,05h,01h,56h,98h
01h,0FEh,00h,00h,26h
01h,0FEh,10h,00h,00h
00h,00h
```

UDPsendtest Program

You may use the UPDsendtest program located in the SitePlayer installation directory to test the UDP packets sent out by SitePlayer.



IP address to receive:

If necessary, change this to the IP address of the adapter in your computer that will receive the UPD messages.

Port#

Must be set to match the port object's value at 02DAh (UDPport).

Initialize UDP button

Press this to activate UDP reception in the test program.

UDP Message Window

Initially displays brief instructions on using the program. Once initialized, it will show the UPD packets received by the test program.

A) Memory Map / Serial Commands

SitePlayer Object Memory Map	
Address	**Description**
0000h	Regular objects addressable with one or two byte addresses 02D0h – 02DFh can also be UPDsend objects (see table)
02DFh	
02E0h	Bit objects addressed with one byte bit addresses, or regular objects with two byte addresses
02FFh	
0FF00h	Special SitePlayer Functions (see table)
0FFFFh	

SitePlayer UDPsend Memory Map		
Address	**Name**	**Description**
02D0h	UDPMAC	1st of 6 bytes for Destination MAC address
02D1h	UDPMAC2	2nd of 6 bytes for Destination MAC address
02D2h	UDPMAC3	3rd of 6 bytes for Destination MAC address
02D3h	UDPMAC4	4th of 6 bytes for Destination MAC address
02D4h	UDPMAC5	5th of 6 bytes for Destination MAC address
02D5h	UDPMAC6	6th of 6 bytes for Destination MAC address
02D6h	UDPIP	1st of 4 bytes for Destination IP address
02D7h	UDPIP2	2nd of 4 bytes for Destination IP address
02D8h	UDPIP3	3rd of 4 bytes for Destination IP address
02D9h	UDPIP4	4th of 4 bytes for Destination IP address
02DAh	UDPPORT	2 bytes for Destination Port address
02DCh	UPDADDR	2 bytes for Starting Memory Address of object(s) to send
02DEh	UDPCOUNT	2 bytes for Count of bytes to send 1-768

SitePlayer UDP Receive Structure		
Name	**Size**	**Description**
Number of bytes	1 Byte	Number of bytes in the data
Compliment of number of bytes	1 Byte	Ones compliment of the number of bytes in the data
Data Address	2 Bytes	Pointer to object address in SitePlayer. Lower byte first
Data Value(s)	1…n	The data bytes

SitePlayer Special Functions Memory Map

Address	Name	Description
0FF00h	P1	Port 1 – complete 8 bit port
0FF01h	CMOD	PCA Counter Mode
0FF02h	CCON	PCA Counter Control
0FF03h	CH	PCA Counter High
0FF04h	CL	PCA Counter Low
0FF05h	CCAPM0	Module 0 Mode
0FF06h	CCAPM1	Module 1 Mode
0FF07h	CCAPM2	Module 2 Mode
0FF08h	CCAPM3	Module 3 Mode
0FF09h	CCAP0H	Module 0 Capture High
0FF0Ah	CCAP1H	Module 1 Capture High
0FF0Bh	CCAP2H	Module 2 Capture High
0FF0Ch	CCAP3H	Module 3 Capture High
0FF0Dh	CCAP0L	Module 0 Capture Low
0FF0Eh	CCAP1L	Module 1 Capture Low
0FF0Fh	CCAP2L	Module 2 Capture Low
0FF10h	CCAP3L	Module 3 Capture Low
0FF11h	**IO0**	Port 1 Bit number 0
0FF12h	**IO1**	Port 1 Bit number 1
0FF13h	**IO2**	Port 1 Bit number 2
0FF14h	**IO3**	Port 1 Bit number 3
0FF15h	**IO4**	Port 1 Bit number 4
0FF16h	**IO5**	Port 1 Bit number 5
0FF17h	**IO6**	Port 1 Bit number 6
0FF18h	**IO7**	Port 1 Bit number 7
0FF19h	COM	Serial Port Output Object
0FF1Ah	Baud	Baud Rate Object
0FF1Ch	SPIbus	SPI Object
0FF1Dh	SPIaddr	Incoming SPI Data Memory Address Pointer
0FF1Eh	UDPsend	UDPsend Object
0FF1Fh	HalfSec	HalfSec Object
0FFE0h – 0FFE5h	MAC Address	6 byte Ethernet MAC Address
0FFE6h – 0FFE9h	Current IP Address	Current IP Address

For information on these special registers, please refer to the Philips Documentation on the P89C51RD2 processor.

Command Byte							
Command Value				**Number of bytes sent/requested**			
Up to 16 different commands				1 to 16 bytes (0=1, 1=2… 15=16)			
D7	D6	D5	D4	D3	D2	D1	D0

SitePlayer Serial Commands		
Command	**Command Byte**	**Description**
NOP	00h	Do Nothing, Recommended initializing command
Status	10h	Return Status of SitePlayer
Reset	20h	Perform a Watchdog Reset
ComParams	33h	Sets Baud Rate and UARTdelay
UDPsend	50h	Sends a UDP Packet
Read	0C0h	Read Object from SitePlayer
Write	80h	Write Object to SitePlayer
ReadX	0D0h	Read Using Extended Two Byte Addressing
WriteX	90h	Write Using Extended Two Byte Addressing
ReadBit	0E0h	Read a bit variable, One byte Address
WriteBit	0A0h	Write a bit variable, One byte Address
ToggleBit	0B0h	Toggles a bit variable, One byte Address