

# STREAMING CLIENT



## Streaming Client

Network audio decoder firmware for  
WMA and MP3 streaming (HTTP,  
UDP, RTP) with automatic failover  
and USB playback



## Technical Documentation

Firmware V1.31

Released 2. Feb. 2007

Supports:

- EXSTREAMER (legacy)
- EXSTREAMER 100
- EXSTREAMER 200
- IP Audio Module
- IP Audio Module 200



# Table of Contents

<b>I Introduction.....</b>	<b>4</b>
1.1 About the “Streaming Client” firmware.....	4
1.2 Features.....	4
1.3 Installing the device.....	5
1.4 Additional documents.....	5
1.5 Preloaded Firmware.....	5
1.6 About this Technical Documentation.....	5
Links to chapters.....	5
Bookmarks pane in Adobe Acrobat.....	5
Chapter overview.....	6
<b>2 Memory organization.....</b>	<b>7</b>
2.1 Serial Rescue Kit / Web Update.....	7
2.2 Flash Memory usage.....	7
Flash memory usage table.....	7
2.3 Configuration storage (EEPROM).....	8
Factory defaults using Serial Rescue Kit.....	8
Factory defaults using Web Update.....	8
Configuration storage usage.....	8
<b>3 Application Programming Interface (API).....</b>	<b>12</b>
3.1 CGI WEB command interface.....	12
Principles of CGI WEB interface.....	12
List of CGI WEB commands.....	12
<b>4 WEB User interface.....</b>	<b>14</b>
4.1 User Interface Development Kit .....	14
Web2cob tool.....	14
Original UI Files.....	15
4.2 Dynamic Web Pages.....	17
Initial Dynamic Mark.....	17
Syntax of Dynamic Marks.....	17
List of Dynamic Mark IDs for &LSetup.....	18
List of Dynamic Mark IDs for &LState.....	19
List of Dynamic Mark Parameters for &LState.....	19
4.3 Configuration via HTML Pages.....	21
Examples.....	21
Form element names.....	23
<b>5 Advanced Streaming Settings.....</b>	<b>24</b>
5.1 URL Variable Substitution.....	24
<b>6 IR control interface.....</b>	<b>25</b>
IR Buttons.....	25
Channel Selection.....	25
File “remote.ini”.....	25
Barix IR Remote Control Button Assignment.....	27
<b>7 Remote Configuration and Update interface.....</b>	<b>28</b>
7.1 Configuration parameters.....	28
Update URL.....	28
Remote Update Period.....	28
7.2 Configuration Meta File.....	28
Keywords.....	28
Control Commands.....	28
Config values.....	28
Execution procedure.....	29
File “update.ini”.....	29
Configuration Meta File Grammar.....	30
7.3 How to update the firmware remotely.....	31
7.4 How to configure the device remotely.....	32
Device dependent update files.....	32
<b>8 Remote monitoring interface.....</b>	<b>34</b>
8.1 MTELL Remote Monitoring.....	34
Configuration Parameters for MTELL periodic report.....	34
Requesting MTELL report.....	35

8.2 Own Monitoring Server using MTELL protocol.....	35
Configuration Parameters for MTELL periodic report.....	35
Example “submit.php” .....	36
8.3 SNMP Remote Monitoring.....	36
Configuration for SNMP trap sending.....	36
SNMP querying.....	36
File “BARIXAUDIOSNMP.MIB”.....	36
<b>9 Legal Information.....</b>	<b>43</b>

# I Introduction

---

## I.1 About the “Streaming Client” firmware

The “Streaming Client” firmware was designed for the professional field: audio bridging, audio distribution, in store and standalone applications.

It is capable of playing MP3 and WMA files using various protocols. Up to three sources can be defined (both streaming over network and playing from a local USB storage) for streaming with automatic failover.

Thanks to easy remote control and monitoring the “Streaming Client” firmware can be used on Barix devices to build a manageable distributed audio network.

The standalone capability (playing from external USB or internal flash memory, without network connection) allows the use of the Barix Exstreamer 100 or the Barix Exstreamer 200 as a simple MP3/WMA player with automatic start on power up.

## I.2 Features

- Plays MP3 streams from network (HTTP, BRTP, RTP) and M3U playlists (HTTP)
- Plays WMA streams from network (MMS) and WMA files (HTTP) \*
- Plays MP3 files, M3U playlists and WMA files from external USB memory \*
- Supports authentication (HTTP, Shoutcast, Icecast)
- Supports up to 3 sources with automatic failover
- Control and configuration using a standard web browser
- Supports automatic remote update of settings, configuration and firmware
- Monitoring using SNMP and MTELL (HTTP, UDP)
- Supports the Barix IR Remote Control
- Automatic network configuration (BOOTP, DHCP, AutoIP and IPzator) as well as manual static IP configuration
- Features SonicIP® announcing the IP address on power up over the audio outputs
- Supports proxy server (HTTP proxy support)
- Autoplay functions plays all audio files without playlist (standalone mode)

\* These features are not available for legacy devices (Exstreamer, Exstreamer Wireless, Exstreamer Digital and Exstreamer Gold).

### 1.3 Installing the device

For the installation of the Barix Exstreamer 100 or the Barix Exstreamer 200 please refer to the corresponding “Quick Install Guide”. A printed version is included in the box and can also be downloaded from our site [www.barix.com](http://www.barix.com).

For the installation of the Barix IP Audio Module or the Barix IP Audio Module 200 please refer to the corresponding “Development Specification” which can be downloaded from our site [www.barix.com](http://www.barix.com).

### 1.4 Additional documents

Technical specifications can be found in the corresponding product sheet which can be downloaded from our site [www.barix.com](http://www.barix.com).

For configuration information please download the “Streaming Client Manual” from our website.

### 1.5 Preloaded Firmware

Barix preloads all Exstreamer family devices with the “Standard” firmware version, which suits most home and consumer applications.

Before continuing with this technical documentation the firmware has to be changed from “Standard” to “Streaming Client” firmware. Please follow the steps in chapter “Updating the Firmware” of the “Streaming Client Manual” in order to change the firmware.

### 1.6 About this Technical Documentation

#### Links to chapters

References to chapters (e.g. [X Chapter name](#)) are red and underlined and serve as direct links when viewed in Adobe Acrobat Viewer. Click on the link to jump to the referenced chapter, click on the left arrow icon to jump back to where you came from.

#### Bookmarks pane in Adobe Acrobat

The complete “Table of Contents” is available in Adobe Acrobat Viewer. Click on the “Bookmarks” pane tab on the left side of Adobe Acrobat Viewer to open it. Click on any bookmark to directly jump to the corresponding part of the manual.

## Chapter overview

This technical documentation is divided into the following chapters:

- [2 Memory organization](#) (explaining the use of the Flash memory and the EEPROM configuration memory)
- [3 Application Programming Interface \(API\)](#) (explaining how to control the device using CGI web commands )
- [4 WEB User interface](#) (explaining the User Interface functionality and how to customize it)
- [5 Advanced Streaming Settings](#) (explaining the functionality of URL Variable substitution)
- [6 IR control interface](#) (explaining the functionality IR Remote control interface)
- [7 Remote Configuration and Update interface](#) (explaining configuration and firmware update via a remote webserver)
- [8 Remote monitoring interface](#) (explaining the remote monitoring capabilities using a MTELL or own monitoring server and explaining the SNMP interface capabilities and the required MIB file)

## 2 Memory organization

### 2.1 Serial Rescue Kit / Web Update

Two different procedures exist to upload the “Streaming Client” firmware into the device:

The “Serial Rescue Kit” using the serial cable will upload the firmware files, the boot loader and the “factory defaults configuration” which will erase the current configuration. The “Web update” using a browser will upload the firmware files and the “factory defaults configuration” but will not alter the current configuration. For factory defaults and memory usage details see the following two sections.

### 2.2 Flash Memory usage

The “Streaming Client” firmware is using the built-in Flash memory as described in the table below.

Flash memory usage table

Page / Target	File name	Content	Address (Rescue Kit)
8K (WEB0)	stream.rom	Firmware	0xC00000
WEB1	fs.bin	USB file system	0xC10000
WEB2	sg.bin	Audio and Utility library	0xC20000
WEB3		reserved	0xC30000
WEB4	streamapp.cob	Web Application and SonicIP Resources	0xC40000
WEB5	streamapp.cob continued	Web Application and SonicIP Resources	continued (0xC50000)
WEB6	streamapp.cob continued	Web Application and SonicIP Resources	continued (0xC60000)
WEB7...WEB14	reserved for remote firmware update		0xC70000...0xCE0000

A page uses 64 kilobytes of flash memory. (Note: 0xC00000 = 0xD00000 = 0xE00000 = 0xF00000)

Both update procedures (Web update & Serial Rescue Kit) respect the above memory usage.

The above memory usage table must be used accordingly when loading single files using advanced web update. The target has to be in capital letters (i.e. WEB4).

### 2.3 Configuration storage (EEPROM)

The current configuration is stored in a non-volatile memory (EEPROM). To change the current configuration use the web user interface and hit the “Apply” button to store it into the EEPROM as described in the “Streaming Client Manual” in chapter “Device Configuration”.

#### Factory defaults using Serial Rescue Kit

The EEPROM is overwritten by the “factory defaults configuration” when applying the “Serial Rescue Kit” using the binary file `config.bin` which is stored in the folder “update\_rescue”. This file can be edited with a hex editor. Consult the “configuration memory usage” table carefully before you make any changes.

#### Factory defaults using Web Update

The “factory defaults configuration” binary file `config.bin` is contained in the file `streamapp.cob` which is loaded into the flash memory (not the EEPROM!) when applying the “Web Update”. To apply the “factory defaults configuration” the reset button has to be pushed for about 10 seconds.

The file `config.bin` can be edited with a hex editor. Consult the “configuration memory usage” table carefully before you make any changes. Before uploading the folder `streamapp` (residing in folder `webuidevkit`) has to be packed into the file `streamapp.cob` using the tool `web2cob.exe`. The file is loaded to the EEPROM as factory default when the reset button is pushed for about 10 seconds. For more details see chapter [4 WEB User interface](#).

#### Configuration storage usage

The following table shows where the configuration is stored in the EEPROM. The column “Byte” shows the offset as a decimal number. The column “Len” shows the length in Bytes. The column “Default” shows the default value as stored in the original “factory defaults configuration”.

Parameter	Byte	Dynamic Name	Len	Default	Short Description
Own IP	0	B0,B1, B2,B3	4	0.0.0.0	Static IP address of the device. 0.0.0.0 for automatic assignment 0.0.1.0 to disable AutoIP 0.0.2.0 to disable BOOTP 0.0.4.0 to disable DHCP 0.0.8.0 to disable IPzator add these special IP addresses to disable multiple protocols
Gateway IP	4	B4,B5, B6,B7	4	0.0.0.0	Gateway IP address. 0.0.0.0 for no gateway
Netmask	8	N8B0, N8B1, N8B2, N8B3	1	0	Subnet mask. The value is the count of the zero bits counted from the lowest byte. (eg. 8 for 255.255.255.0)
DNS 1	64	B64,B65, B66, B67	4	0.0.0.0	Primary DNS IP address. Set to 0.0.0.0 to get primary DNS from DHCP, if DHCP is configured, or to disable DNS, if DHCP is not configured.
DNS 2	68	B68, B69, B70, B71	4	0.0.0.0	Alternative DNS IP address. 0.0.0.0 here always disables secondary DNS

Parameter	Byte	Dynamic Name	Len	Default	Short Description																																																																																	
IFMODE0	80	B80b0-1, B80b2-3, B80b4-5, B80b6-7 or B80	1	0x4C	Serial port 0 settings Definition of the bits in that byte for the serial port 0: <table border="1"> <thead> <tr> <th>Function</th> <th>7</th> <th>6</th> <th>5</th> <th>4</th> <th>3</th> <th>2</th> <th>1</th> <th>0</th> </tr> </thead> <tbody> <tr> <td>RS232-C</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> <td>0</td> <td>0</td> </tr> <tr> <td>7 Bit</td> <td></td> <td></td> <td></td> <td></td> <td>1</td> <td>0</td> <td></td> <td></td> </tr> <tr> <td>8 Bit</td> <td></td> <td></td> <td></td> <td></td> <td>1</td> <td>1</td> <td></td> <td></td> </tr> <tr> <td>no parity</td> <td></td> <td></td> <td>0</td> <td>0</td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>even parity</td> <td></td> <td></td> <td>1</td> <td>1</td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>odd parity</td> <td></td> <td></td> <td>0</td> <td>1</td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>1 Stopbit</td> <td>0</td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> <tr> <td>2 Stopbit</td> <td>1</td> <td>1</td> <td></td> <td></td> <td></td> <td></td> <td></td> <td></td> </tr> </tbody> </table>	Function	7	6	5	4	3	2	1	0	RS232-C							0	0	7 Bit					1	0			8 Bit					1	1			no parity			0	0					even parity			1	1					odd parity			0	1					1 Stopbit	0	1							2 Stopbit	1	1						
Function	7	6	5	4	3	2	1	0																																																																														
RS232-C							0	0																																																																														
7 Bit					1	0																																																																																
8 Bit					1	1																																																																																
no parity			0	0																																																																																		
even parity			1	1																																																																																		
odd parity			0	1																																																																																		
1 Stopbit	0	1																																																																																				
2 Stopbit	1	1																																																																																				
BAUDRATE0	81	B81	1	2	Baudrate for the serial port 0. (7 = 300, 6 = 600, 5 = 1200, 4 = 2400, 3 = 4800, 2 = 9600, 1 = 19200, 0 = 38400, 9 = 57600, 8 = 115200)																																																																																	
FLOWCONTROL0	82	B82	1	0	Flow control for the serial port 0. (0 = no, 1= Software XON/XOFF, 2 = Hardware RTS/CTS)																																																																																	
BOOTTARGET	94	W94	2	0x0000	If this parameter is set to 0x0000 the firmware with the highest version will be started. If this parameter is set to a value highest version of the firmware with this target will be started. If the target is 3Q (like for the standard Exstreamer firmware) the value will be the ASCII code of this two characters 0x5133. The first character is the high byte.																																																																																	
Reset button settings	97	B97	1	0	bit0: 0 - reset function enabled, 1- reset function disabled bit1: 0 - factory defaults enabled, 1 - factory defaults disabled																																																																																	
DHCP Host Name	98	S98	16		Name of the device sent in DHCP request. If not set, automatically generated name based on device's MAC address is sent. The string includes terminating zero.																																																																																	
Version Major	116	B116	1	1	Version Major value (do not change)																																																																																	
Version Minor	117	B117	1	3	Version Minor value (do not change)																																																																																	
Setupex Length	120	W120	2	894	Length of the extended setup (always 894)																																																																																	
Password Level 0	122	S122	8		Password reserved for future use, stored as a MD5 hash (first 8 bytes), all 0 means no password																																																																																	
Password Level 1	130	S130	8		Password used for viewing and changing the configuration, stored as a MD5 hash (first 8 bytes) all 0 means no password																																																																																	
Password Level 2	138	S138	8		Password reserved for future use, stored as a MD5 hash (first 8 bytes), all 0 means no password																																																																																	
Password Level 3	146	S146	8		Password reserved for future use, stored as a MD5 hash (first 8 bytes), all 0 means no password																																																																																	
Password Level 4	154	S154	8		Password reserved for future use, stored as a MD5 hash (first 8 bytes), all 0 means no password																																																																																	
Password Level 5	162	S162	8		Password reserved for future use, stored as a MD5 hash (first 8 bytes), all 0 means no password																																																																																	

Parameter	Byte	Dynamic Name	Len	Default	Short Description
Volume	244	B244	1	8	Volume Range: 0..20, corresponds to 0%,5%,...,100%
Reserved	245		32	0	
Media Configuration	277	B277b0, B277b1, B277b2, B277b3, B277b4, B277b5, B277b6, B277b7	1	0x02	This values can be added (the function is activated by setting the bit): 0x01: 0 – shuffle off, 1 – shuffle on 0x02: 0 – USB Autoplay off, 1 – USB Autoplay on 0x04: not used 0x08: not used 0x10: not used 0x20: not used 0x40: not used 0x80: 0 – SonicIP on, 1 – SonicIP off
Remote Update File Version	278	W278	2	0	Version of the last update-meta file processed. Internally used by the firmware. For further details see chapter <a href="#">7 Remote Configuration and Update interface</a> .
Web Server Port	283	W283	2	0	Port on which built-in web server is running. Range: 0...65535 (0 stands for standard port 80)
USB Serial Number	285	D285	4		Used for playlist position memory
RTP Priority Port	289	W289	2	0	RTP port for receiving priority messages. Range: 1...65535, 0 means disabled (default)
Priority Buffer length	291	W291	2	100	Buffer data for given number of milliseconds before starting playback priority message. Range: 0...65535
IR Source	497	B497	1	0	IR receiver type: 0= Serial IR Dongle 1= Built-in IR receiver
Buffer length	509	W509	2	65535	Buffer data for given number of milliseconds before starting playback. Range: 0...65535
Preset	515	W515	2	0	User-specific storage, this parameter has no functionality. It can be used by the user for the web interface.
User-Agent	517	S517	32		HTTP/Shoutcast/Icecast User-Agent string If empty, default Streaming Client identification is used.
Reserved	677		1	0	
UDP MTELL port	684	W684	2	0	UDP port where the device reports its MTELL status. 0 means disabled.
SNMP Target Trap IP address	686	B686, B687, B688, B689	4	0.0.0.0	SNMP Target IP (0.0.0.0 for disable SNMP)

Parameter	Byte	Dynamic Name	Len	Default	Short Description
Update Period	690	W690	2	720	Period in minutes how often to poll update information from a remote server. Range 1...1000. For further details see chapter <a href="#">7 Remote Configuration and Update interface</a> .
MTELL Period	692	W692	2	5	Period in minutes how often to send device's status to the MTELL server. Allowed values are in range 1...1000. For further details see chapter <a href="#">8.1 MTELL Remote Monitoring</a> .
Stream Check Period	694	W694	2	30	Period in seconds how often the stream sources are checked for availability. Allowed values are in range 1...65535
Stream Max Check Period	696	W696	2	1800	Maximal time period (in seconds) the stream sources are checked. Sources are periodically checked and the period is dynamically changed. This is the maximum value the period can reach. Default is 1800 seconds (30 minutes).
URL1	700	S700	100		URL of first streaming source
URL2	800	S800	100		URL of second streaming source
URL3	900	S900	100		URL of third streaming source
HTTP proxy URL	1000	S1000	100		URL of HTTP proxy server. Note: The proxy is not used for the MMS protocols.
Update URL	1100	S1100	100		Remote update URL. For further details see chapter <a href="#">7 Remote Configuration and Update interface</a> .
MTELL URL	1200	S1200	100		URL of MTELL server. For further details see chapter <a href="#">8.1 MTELL Remote Monitoring</a> .

## 3 Application Programming Interface (API)

---

### 3.1 CGI WEB command interface

#### Principles of CGI WEB interface

- The browser should support frames.
- GET method should be used in forms.
- Respect the common character set for URLs.
- Example for CGI WEB commands: `http://x.x.x.x/rc.cgi?c=99` (command for RESET on Streaming Client with IP address x.x.x.x)
- All strings and everything else are case sensitive.
- All commands are asynchronous to the stream.
- A CGI request should not exceed 1024 bytes.
- To concatenate control commands use `&` (Ampersand, ASCII:38).  
For example, to move to next song and set volume to 12 use: `c=4&v=12`.  
The commands will be executed from left to right in sequence (not parallel).
- If `"L="` is not used, blank page is sent as an answer.
- If password is set, it must be sent with `"a="` e.g. `http://x.x.x.x/rc.cgi?c=99&a=password`

#### List of CGI WEB commands

Element	Description	CGI command
PLAY	Restarts current stream	c=1
NEXTSONG	If current source is playlist, next song starts playing.	c=4
PREVSONG	If current source is playlist, previous song starts playing.	c=5
SHUFFLEON	Shuffle on.	c=6
SHUFFLEOFF	Shuffle off.	c=7
VOLUMEINC	Increment volume (one step).	c=19
VOLUMEDEC	Decrement volume (one step).	c=20

Element	Description	CGI command
TOGGLESHUFFLE	Toggle shuffle.	c=30
DEVICERESET	Hard reboot of the device.	c=99
BOOTLOADER	Starts the bootloader. The application will be left. It isn't running until the next reboot.	c=100
GETDYNFILE	The response is the dynamic file stored in a cob file with given name. Example: L=index.html	L=...
PASSWORD	Concatenate this command with the rest of the command sequence if the command interface is password protected. The password has to be added in plain text.	a=...
PUSHDIGIT0 PUSHDIGIT1 --- PUSHDIGIT9	Push digit 0. Push digit 1.  Push digit 9.	r=0 r=1 --- r=9 see chapter <a href="#">6 IR control interface</a>
VOLUME00 VOLUME01 --- VOLUME20	Set minimal volume level (volume off). Set volume level 1.  Set maximal volume level. One step is showed as 5%. The level 0 equals the 0%.	v=0 v=1 --- v=20

## 4 WEB User interface

---

### 4.1 User Interface Development Kit

With the "User Interface Development Kit" you can design your own web pages (skin) and modify the answers to your needs. The "UI Development Kit" is included in the "Streaming Client Update Kit" which is available on [www.barix.com](http://www.barix.com).

Change to the contained folder `uidevkit`.

The folder `streamapp` holds the original HTML files you need for the web pages, the answer text files, lookup files (ini), graphics and sounds as well as the default configuration file `config.bin`.

You can simply edit these files and/or add new ones.

**Note:** Filenames must not start with `rc.cgi` or `setup.cgi`.

#### Web2cob tool

To generate the `streamapp.cob` file start the batch `streamapp.bat` which uses the packaging tool `web2cob.exe`.

Only .cob files up to 192 kilobytes are supported by the Streaming Client.

For the upload of the .cob file to the device, go to the configuration page of the device and click on the button "Update".

After the device has rebooted and the update page appears, click on "Advanced Update".

Enter the correct Target (check the flash memory usage table) in upper case letters.

Select the cob file you want to upload and hit the "OK" button.

Click on the "Upload" button.

#### Rules:

- If you upload a .cob file to already used pages the current content will be overwritten
- The web server in the device sees all the targets (.cob files) as one directory
- If two files in different .cob files have the same name then the one from the lower page is chosen.

After the upload reboot the device and reload the modified page in the browser to see the changes.

Depending on the browser's cache strategy, sometimes it's needed to close and reopen the browser to see the changes.

### Original UI Files

The web interface (and the firmware) need at least the following files (more example files might be included):

Type	Filename.extension	Description
Binary	config.bin	Factory default settings. The file is binary and it is an exact mirror for the EEPROM
HTML	index.html	Main page of the web server, frameset including the frames: uifmenu, uimenuline, uifsettings, empty. “empty” is a hidden frame that receives the answer of the CGI commands
HTML	rebooting.html	Page shown while device is rebooting
HTML	status	Status page showing all configuration and useful run time parameters
HTML	uicfg.html uifloader.html uifmenu.html uifreboot.html uifsettings.html uifupdate.html	specific configuration pages, containing the frames for the corresponding configuration pages
HTML	uihadvanced.html uihapply.html uihloader.html uihnetwork.html uihplay.html uihpriority.html uihreboot.html uihremote.html uihsecurity.html uihsettings.html uihstreaming.html uihupdate.html	help for the corresponding configuration pages
HTML	uilogout.html	logout page
HTML	uimenuline.html	Page with menu icons/buttons
HTML	uireboot.html	Page for reboot of the device
HTML	uirloader.html uirreboot.html uirupdate.html	Shown after pressing “apply” or during reboot of the device
HTML	uirrebootI.html	Shown after the device is updated and successfully rebooted
HTML	uisettings.html	Page for configuration

Type	Filename.extension	Description
HTML	uiupdate.html	update the device
HTML	update.html	forwarding page to hide the command for the update
Image	4to0.gif	needed for waiting for the reboot of the device
Image	activei.gif	Info reload icon/button
Image	barix.gif	Barix logo and “Streaming Client” title
Image	exstreamer.gif	Exstreamer logo including volume buttons
Image	h.gif	Help icon/button
Image	menu.gif	Menu icons/buttons
Sound	0.mp3	spoken “0”
Sound	1.mp3	spoken “1”
Sound	2.mp3	spoken “2”
Sound	3.mp3	spoken “3”
Sound	4.mp3	spoken “4”
Sound	5.mp3	spoken “5”
Sound	6.mp3	spoken “6”
Sound	7.mp3	spoken “7”
Sound	8.mp3	spoken “8”
Sound	9.mp3	spoken “9”
Sound	dot.mp3	spoken “dot”
Text	remote.ini	lookup file for IR commands, see section <a href="#">File “remote.ini”</a>
Text	SONICIPVERSION	for the version number of SonicIP implementation
Text	STREAMAPPVERSION	for the version number and the history of Streaming Client
Text	update.ini	lookup file for names used in remote controlling, see section <a href="#">File “update.ini”</a>

## 4.2 Dynamic Web Pages

Web pages can include dynamic values. Dynamic Web Pages are built in HTML or XML or in an other text file format that exclude the binary character 0x00, i.e. the dynamic page can be an HTML file. It's possible to use scripts or everything else allowed in the given document's file format.

### Initial Dynamic Mark

In order to indicate that Web page is dynamic, it has to contain the special initial dynamic mark `&L(0,"*")`; in the first 500 Bytes and before any other dynamic value is used. The initial mark can also have decimal number as its optional third parameter. Example of such initial mark is `&L(0,"*",1)`;

The third parameter is parsed bitwise and has the following meaning:

- If bit 7 is set then the code page IBM437 will be used instead of the standard HTML code page.
- If bit 0 is set, then the content length will not be included in the HTTP header. Page is sent faster by saving the time needed to calculate the content length.

### Syntax of Dynamic Marks

Dynamic marks can be used to put dynamic values in Web pages. All dynamic marks have the following syntax: `&L<name>(<id>,<format>[,<par>])`; A dynamic mark always starts with &L and it is always case sensitive.

- `<name>` selects a group of dynamic values. Defined is the "Setup" group for all configuration parameters and the "State" group for actual parameter states. Remaining parameters are included in parentheses, with the right parenthesis followed by a semicolon.
- `<id>` determines the desired function.
- `<format>` is a C-style format string (refer to the ANSI documentation).
- `<par>` are optional additional parameters. If additional parameters are needed, it is mentioned in the function lists below.

**Note:** The string `) ;` is not allowed inside a dynamic mark.

To have this construct inside the format string, use `) \;` (in an unknown escape sequence, only the `'\'` will be removed).

To have a `"%"` sign (percent sign) inside the format string, use `"%%"` (two signs without space).

The whole mark is replaced by the dynamic value formatted with the `<format>` string. Only one value is allowed per dynamic mark. The length of the dynamic mark mustn't exceed 500 characters. The resulting string from the dynamic mark must not exceed 500 characters.

A dynamic mark can be contained in an another dynamic mark. Only one recursion step is allowed and correct "escaping" has to be applied. Example:

```
&LSetup(3,"%s",419,B,!0,"<meta http-equiv=refresh content=\"&LSetup(1,\"%u\",419)\"; url=info.html\">");
```

Note the special `"\"` before the semicolon of the dynamic mark inside. This is because the escape sequence is interpreted as only a semicolon and is needed in order to include the prohibited sequence `) ;` inside a dynamic mark.

**List of Dynamic Mark IDs for &LSetup**

ID	Type	Description
1	Function	Print setup value 3. [par]: Address (decimal) of the value in the setup 4. [par]: Type of the value (B for unsigned byte, W for word, D for double word, c for char/signed byte, b for bit numbered from 0 to 7, e.g. b3 for the fourth bit). If this parameter isn't available the type will be B. e.g. <code>&amp;LSetup (1, "%08lx", 315, D)</code> ; as hexadecimal value with 8 characters and leading zeros e.g. <code>&amp;LSetup (1, "%lu", 311, D)</code> ; as unsigned long decimal value
2	Function	Print Netmask Byte 3. [par]:Address (decimal) of the value in the setup 4. [par]: Byte number of the Netmask IP address byte starting with 0 for the first left byte and incremented by one for the next bytes
3	Function	Print string if equal 3. [par]: Address (decimal) of the value in the setup 4. [par]: Type (see id 1 above) 5. [par]: value to compare. The prefixes !, > or < are allowed to change the comparison (no spaces between) 6. [par]: string for output if value at address is equal to 5. [par]
4	Function	Print string 3. [par]: Address (decimal) of the value in the setup
5	Byte (integer)	Firmware Version Major
6	Byte (integer)	Firmware Version Minor
7	Byte (integer)	Bootloader Version Major
8	Byte (integer)	Bootloader Version Minor
9	Function	Prints the version out of a standard version file in a *.cob application 3. [par]: name of the version file 4. [par]: 1 for major version number (byte), 0 for minor version number (byte)
10	Byte (integer)	year of the firmware build (only decade)
11	Byte (integer)	month of the firmware build
12	Byte (integer)	day of the firmware build
13	Byte (integer)	sg.bin (Audio and Utility library) Version Major
14	Byte (integer)	sg.bin (Audio and Utility library) Version Minor
15	Byte (integer)	fs.bin (USB file system) Version Major
16	Byte (integer)	fs.bin (USB file system) Version Minor
17	Byte (integer)	sg.bin (Audio and Utility library) year of the build (only decade)

ID	Type	Description
18	Byte (integer)	sg.bin (Audio and Utility library) month of the build
19	Byte (integer)	sg.bin (Audio and Utility library) day of the build
20	Byte (integer)	fs.bin (USB file system) year of the build (only decade)
21	Byte (integer)	fs.bin (USB file system) month of the build
22	Byte (integer)	fs.bin (USB file system) day of the build

**List of Dynamic Mark IDs for &LState**

ID	Type	Description
1	Function	Print status variable 3. [par]: Variable index, see the parameters table below, e.g. <code>&amp;LState(1, "%s", 12)</code> ; prints out device's MAC address
2	Function	Print string if condition is true 3. [par]: Index of the variable to be compared, see the parameters table below 4. [par]: value to compare. Variable is compared with the value "if equals", the prefixes !, > or < can be used to change the comparison (no spaces between allowed). If comparing variable with a string, the string has to be quoted ( e.g. "string") 5. [par]: string to output output if condition is true. The string has to be quoted.

**List of Dynamic Mark Parameters for &LState**

Par	Type	Description
0	Boolean (Int.)	File system present (1 if present)
1	Integer	File system type (0,1,2,4,8) 0=unknown, 1=FAT12, 2=FAT16, 4=VFAT, 8=FAT32
2	Integer	File system serial number
3	Integer	Audio volume in percent (0..100)
4	Integer	Current stream number (or 99 for priority stream)
5	Integer	Last error (number)
6	Integer	Audio buffer level

Par	Type	Description
7	Integer	Lost frames counter
8	Integer	Soft error counter
9	String	Current URL ("PRIORITY" when receiving priority stream)
10	Integer	Stream bit rate in kilobits per second
11	Integer	Reconnection counter
12	String	Device's MAC address (each byte separated by a colon e.g. 00:08:E1:00:3D:90)
13	String	Current IP address (four numbers, dot separated, without leading zeroes)
14	Integer	USB device vendor ID
15	Integer	USB device product ID
16	Integer	USB device class
17	Integer	USB device subclass
18	Integer	USB interface class
19	Integer	USB interface subclass
20	Integer	USB device's max. power consumption in milliamperes
21	Boolean (Int.)	USB device attached
22	Integer	USB device capacity in kilobytes
39	Integer	System uptime in milliseconds
40	Integer	System uptime in seconds

### 4.3 Configuration via HTML Pages

The HTML pages for the device configuration use the "dynamic web page" functionality. All of the configuration parameters are placed in HTML forms and are transferred by the "GET" method. Input values can be checked by Javascript to prevent wrong values (see example below). Not all of the configuration parameters have to be present in the form. It is possible to have only a part of the configuration on a web page. The form has to start with the following two tags:

```
<form method=GET action=setup.cgi target="empty"><input type="hidden" type="text" name=L value=rebooting.html>
```

The target of the form could be changed.

The answer after transmitting the form will be the HTML page `rebooting.html`. For another HTML page change this value. If this value isn't available only the HTTP status 200 OK will be sent back.

#### Examples

The following example shows how to implement a form field for the configuration value of the highest byte in the 'own IP address'. The input element name is a defined string, which has to be handled with care. The type character **B** stands for an unsigned value. 0 is the address of the expected configuration parameter. The value is a dynamic mark. The string `onChange=IPCheck(this)` will call the Javascript `util.js` to check if the value entered is in the range of 0 to 255.

```
<input name=B0 size=3 maxlength=3 value=&LSetup(1,"%u",0) ; onChange=IPCheck(this)>
```

In the next example the name selects the configuration parameter "DHCP Host Name".

```
<input name=S98 size=15 maxlength=15 value="&LSetup(4,"%s",98) ;">
```

This example shows how to implement a form field for the configuration of the Netmask. The names for the bytes of the Netmask are **N8B0**, **N8B1**, **N8B2** and **N8B3**. 8 is the address of the Netmask in the configuration memory. The value after the **B** is the byte number of the byte in the Netmask starting with 0 for the first byte at the left. This special handling for Netmask is needed because the Netmask is stored in one byte and not like the IP address in 4 bytes. The string `onChange=netMaskCheck(this)` will call the Javascript `util.js` to check if the value entered is in the correct range.

```
<input name=N8B0 size=3 maxlength=3 value=&LSetup(2,"%u",8,0) ; onChange=netMaskCheck(this)>
```

The next example shows how to implement a form field for the configuration of the parameter "Volume" as a selection. If the value of the configuration parameter is equal to the second last parameter in the dynamic mark it will be replaced by the last parameter of the dynamic mark.

```
<select size=1 name=B244>
  <option value=0 &Lsetup(3,"%s",244,B,0,"selected") ;>0</option>
  <option value=1 &Lsetup(3,"%s",244,B,1,"selected") ;>5</option>
  .....
  <option value=19 &Lsetup(3,"%s",244,B,19,"selected") ;>95</option>
```

```

    <option value=20 &Lsetup(3,"%s",244,B,20,"selected");>100</option>
</select><font size=2>

```

This example shows how to implement radio buttons for the configuration parameter 'Sonic IP'.  
The functions of the dynamic marks are equal to the example above.

```

<input type=radio name=B277b7 value=0&LSetup(3,"%s",277,b7,0," checked");>Yes
<input type=radio name=B277b7 value=1&LSetup(3,"%s",277,b7,1," checked");>No

```

To transmit the new configuration data to the device the submit input type of the form is used.

```

<input type=submit value=" Apply ">

```

By pressing the Apply button the new configuration data will be transferred to the device. It will store the new data to its configuration memory (EEPROM).  
After this it sends the answer (see above) to the browser and reboots itself to apply the new configuration.

Passwords are hashed (MD5) and stored in memory and set using the name P $x$ , where  $x$  stands for the password level.

If the password is set already, the old password must also be supplied (with the name P $x$ ) together with the new password using the name P $x$ .1 (P level dot one).

```

<tr>
    &Lsetup(3,"%s",130,D,0,"
    <td><b><font size=2>Set Password</font></b></td>
    <td><input name=P1 size=18 maxlength=25 type=password value=></td>
    ");
    &Lsetup(3,"%s",130,D,!0,"
    <td><b><font size=2>Old Password</font></b></td>
    <td><input name=P1 size=18 maxlength=25 type=password value=></td>
</tr>
<tr>
    <td><b><font size=2>New Password</font></b></td>
    <td><input name=P1.1 size=18 maxlength=25 type=password value=></td>
    ");
</tr>

```

P $x$  and P $x$ .1 can also be used for remote configuration.

## Form element names

- If the value is an integer (1 byte) the first character is a **B**.
- If the value is an IP address the first character is an **I**, the complete IP address can be set as a string at once e.g.:  
**I0=192.168.1.2** (same as **B0=192 B1=168 B2=1 B3=2**) for IP address  
**I4=192.168.1.1** (same as **B4=192 B5=168 B6=1 B7=1**) for Gateway IP address
- If the value is a Netmask the first character is an **N**, e.g.:  
**N8=255.255.255.0** (same as **N8B0=255 N8B1=255 N8B2=255 N8B3=0**)
- If the value is a string the first character is an **S**.
- If the value is a word (2 bytes) the character is a **W**.
- If the value is a double word (4 bytes) the first character is a **D**.

The following decimal value in the name is the address of the configuration parameter (see chapter [2.3 Configuration storage \(EEPROM\)](#)).

To set a bit in a configuration parameter (e.g. Media Configuration) add the character **b** followed by the number of the bit (starting at 0), e.g.: **b7** for the 8. bit in the byte.

Examples of names:

- **B0** first (left) byte of the configuration parameter 'own IP address'
- **B1** second byte of the configuration parameter 'own IP address'
- **N8B0** first (left) byte of the Netmask
- **N8B1** name of the second byte of the Netmask
- **N8** Netmask
- **S98** DHCP Host Name
- **B277b7** Sonic IP

## 5 Advanced Streaming Settings

---

### 5.1 URL Variable Substitution

URLs may contain variables which are processed and substituted by their values. Variables have syntax **\$NAME\$**, where name consists of printable upper case characters other than '\$'. Following variable names are defined:

Name	Description	Example
<b>MAC</b>	Device MAC address (12 digits in hexadecimal notation with no separators, A..F digits in capital letters)	<b>0008E1002B0E</b>
<b>IP</b>	Device IP address (four numbers, dot separated, without leading zeroes)	<b>192.168.2.202</b>
<b>NAME</b>	DHCP Host name (ASCII string configured in network settings)	<b>XSTREAM1</b>
<b>NUM</b>	channel number (three digits)	<b>003</b>

When the device is fetching a stream then the above variables will be substituted. This allows for specific tasks like identification, zoning and logging of the device. The variable **\$NAME\$** for an example could be used to group several devices to a zone which will be supplied with different streams by the server analyzing the name of the fetching device.

Example:

- URL: `http://myserver.com:4567/device.cgi?zone=$NAME&id=$MAC&logip=$IP`
- URL: `http://myserver.com:4567/device.cgi?channel=$NUM&id=$MAC&logip=$IP`

The variable **\$NUM\$** can also be used to select numbered playlists stored on a USB memory stick using the IR remote control.

## 6 IR control interface

---

When using IR Remote Control, make sure there is line of sight between the IR Serial receiver and the IR Remote control.

### IR Buttons

With the default factory configuration, following buttons can be used:

- +VOL/-VOL – Volume up/Volume down
- +SONG/-SONG – Next song/Previous song (for use with playlists)
- SHUFFLE – Toggles the Shuffle Play Mode
- The digit buttons (0..9) and the Play button (▶) can be used to select channel number.

### Channel Selection

The channel number can be used as a part of the source URL (see chapter [5.1 URL Variable Substitution](#) for details).

Last three digits pressed within five seconds before pressing ▶ are set (from left to right) as the channel number. If less than three digits were pressed within five seconds before ▶, remaining positions (from left) are filled with zeroes.

Examples:

- After pressing 2,3,▶ the channel number will be 023
- After pressing 1,4,5,6,7,▶ the channel number will be 567

### File “remote.ini”

The commands triggered by each button of the IR Remote Control can be user defined. To upload the altered `remote.ini` file follow the procedure described in chapter [4.1 User Interface Development Kit](#). The `remote.ini` file in the sub folder `webuidevkit/streamapp` contains the commands for each button in a separate line. The file format is comma-separated and the values are case-sensitive. Don't write spaces between the separation, the only space needed is the one after the IR coding token (“NE: “ stands for NEC coding). Every line that doesn't contain an IR remote control command is handled as a comment.

- The first field is the IR remote control code sequence received from the remote control (e.g. “NE: 00FE7887“ for the play button).
- A “\*” in the second field tells the IR handler that this button is accepted for repetition (as defined for Volume + and Volume -).
- The third field is not used and should be empty.
- In fourth field, an “L” followed by a decimal number tells to the IR handler to switch to numbered level for the next command. To select the command for the corresponding level, simply add more fields to the end of the line for that button. The fifth field is for the level 1, the sixth field is for the level 2 and so on. The maximal level is 255. The selected level is declined after 2 seconds or when the next button is pressed.

Example: Execute NEXTSONG command upon pushing the +SONG button on the remote control when in level 0: “NE: 00FE48B7,,,c=4“

"Channel number selection can be implemented using IR Remote Control. There is a three digit buffer in the firmware, that can be used with "r=x" and "c=1" commands. The "r=0"... "r=9" commands push digits to the buffer and the "c=1" command sets the value of the NUM variable to the current value of the buffer. The buffer is cleared to zeroes if no digit have been pressed for 5 seconds.

Excerpt of the `remote.ini` file contained in the "Streaming Client Update Kit"

**Exstreamer Remote Control NEC 00FE**

```
NE: 00FE7887,,,c=1
NE: 00FE08F7,,,c=2
NE: 00FED827,,,c=3
NE: 00FE48B7,,,c=4
NE: 00FE6897,,,c=5
NE: 00FEF807,,,c=8
NE: 00FE00FF,,,c=15
NE: 00FEB04F,,,c=16
NE: 00FEA857*,,,c=19
NE: 00FEC837*,,,c=20
NE: 00FEB847,,,c=30
NE: 00FE38C7,,,c=77
NE: 00FE30CF,,,r=0
NE: 00FE40BF,,,r=1
NE: 00FEC03F,,,r=2
NE: 00FE20DF,,,r=3
NE: 00FEA05F,,,r=4
NE: 00FE609F,,,r=5
NE: 00FEE01F,,,r=6
NE: 00FE10EF,,,r=7
NE: 00FE906F,,,r=8
NE: 00FE50AF,,,r=9
NE: 00FED02F,,,s=
NE: 00FE8877,,,c=97
```

**Barix IR Remote Control Button Assignment**

Button	IR code sequence	Button	IR code sequence
0	NE: 00FE30CF	*	NE: 00FED02F
1	NE: 00FE40BF	#	NE: 00FE8877
2	NE: 00FEC03F	+ VOL	NE: 00FEA857
3	NE: 00FE20DF	- VOL	NE: 00FEC837
4	NE: 00FEA05F	+ SONG	NE: 00FE48B7
5	NE: 00FE609F	- SONG	NE: 00FE6897
6	NE: 00FEE01F	MUTE	NE: 00FEF807
7	NE: 00FE10EF	ON/OFF	NE: 00FE807F
8	NE: 00FE906F	PLIST +	NE: 00FE00FF
9	NE: 00FE50AF	PLIST -	NE: 00FEB04F
Play ▶	NE: 00FE7887	SHUFFLE	NE: 00FEB847
Stop ■	NE: 00FE08F7	REPEAT	NE: 00FE38C7
Pause II	NE: 00FED827	WAKEUP	NE: 00FE38C7

## 7 Remote Configuration and Update interface

### 7.1 Configuration parameters

#### Update URL

For remote configuration and update the configuration field "Update URL" can be used to point to the web server (http) containing the "Configuration Meta File". Only the HTTP protocol is supported, including all its options and the possibility of using HTTP Proxy.

#### Remote Update Period

The URL provided is checked and processed periodically. The frequency of checking the configuration meta file can be set in the configuration field "Remote Update Period" in minutes.

### 7.2 Configuration Meta File

When the "Configuration Meta File" is loaded, its version of is checked against the "Remote Update File Version" stored in the EEPROM. The "Configuration Meta File" is only parsed (executed) if the version is higher.

The "Configuration Meta File" can contain three different types of assignments: keywords, control commands and config values.

#### Keywords

ID	Type	Description
<b>VERSION</b>	16bit unsigned decimal number	Meta file version
<b>FW_VERSION</b>	16bit unsigned hexadecimal number	Version of the firmware file
<b>FW_URL</b>	URL string with maximum length of 99 characters	URL of the firmware .bin file

#### Control Commands

Control commands are described in chapter [3.1 CGI WEB command interface](#).

There must be only one command per line, no command concatenation is supported. Commands are executed in the same order as they appear in the file.

#### Config values

Config values are textual descriptors of places in the configuration memory. Config value names are looked up in the `update.ini` file (described in the next section, see chapter [4.1 User Interface Development Kit](#) on how to upload this file).

## Execution procedure

Once the version is checked to be higher the “Configuration Meta File” is processed in the following four steps:

- control commands are executed and config values are stored in the configuration memory
- the firmware is updated if the value of the keyword **FW\_URL** is pointing to a valid firmware file (**compound.bin**) and the value of the keyword **FW\_VERSION** differs (smaller or bigger) from the currently running firmware version
- the version of the executed “Configuration Meta File” is stored in the EEPROM field “Remote Update File Version”
- device restarts if necessary (configuration has been altered, firmware has been changed or the **c=99** command has been issued)

**Exception:** If the firmware update fails the version of the executed “Configuration Meta File” is obviously NOT stored.

For a detailed specification of the configuration meta file grammar see section [Configuration Meta File Grammar](#) further below.

## File “update.ini”

The file `update.ini` is a text file containing lines with the following syntax: `<descriptor>,<address>[,<size>]`

Where:

- `<descriptor>` is a textual descriptor of a configuration value
- `<address>` is a dynamic name of a configuration value (see chapter [2.3 Configuration storage \(EEPROM\)](#) for details).  
For passwords, use `Px` and `Px.1` (see chapter [4.3 Configuration via HTML Pages](#))
- `<size>` is an optional parameter used only for strings. It defines the length of the string in the setup memory.

Content of the `update.ini` file contained in the “Streaming Client Update Kit”

```

volume,B244
url1,S700,100
url2,S800,100
url3,S900,100
proxy_url,S1000,100
update_url,S1100,100
mtell_url,S1200,100
buffer_msec,W509
web_server_port,W283
shuffle,B277b0
usb_autoplay,B277b1
sonic_ip,B277b7
udp_mtell_port,W684
dhcp_host_name,S98,16

```

```

Content of the update.ini file contained in the “Streaming Client Update Kit”

remote_update_period,W690
mtell_period,W692
stream_check_period,W694
stream_max_check_period,W696
rtp_priority_port,W289
priority_buffer_msec,W291
ip_addr,I0
netmask,N8
gateway,I4
dns1,I64
dns2,I68
snmp_trap_ip,I686
password,P1.1
ir_input,B497
user_agent,S517,32
    
```

**Configuration Meta File Grammar**

Type	Description
OCTET	<any 8-bit sequence of data>
CHAR	<any US-ASCII character (octets 0-127) >
CTL	<any US-ASCII control character (octets 0 - 31) and DEL (127)>
TEXT	<any OCTET except CTL, but including HT>
LF	<US-ASCII LF, linefeed (10)>
CR	<US-ASCII CR, carriage return (13)>
HT	<US-ASCII HT, horizontal-tab (9)>
UPALPHA	<any US-ASCII upper-case letter “A” .. “Z”>
LOALPHA	<any US-ASCII lower-case letter “a” .. “z”>
DIGIT	<any US-ASCII digit “0” .. “9”>
ALPHA	UPALPHA   LOALPHA
ALPHADIGIT	ALPHA   DIGIT
CRLF	CR LF
EOL	LF   CRLF

Type	Description
comment	“#” *(TEXT)
rvalue	*(TEXT)
control	ALPHA
keyword	UPALPHA   *[ ALPHADIGIT   “_” ]
config	LOALPHA   *[ ALPHADIGIT   “_” ]
lvalue	keyword   control   config
assignment	lvalue “=” rvalue
content	comment   assignment
line	[ content ] EOL
file	*(line)

### 7.3 How to update the firmware remotely

Let's assume you have an HTTP server `http://www.myserver.net` and want to update your device with firmware version 01.31. Here is an example how to do it:

- create a directory `http://www.myserver.net/streamingclient/update/` on the server
- Upload the `compound.bin` file from the “Streaming Client Update Kit” folder `update_rescue` into your HTTP directory, the URL will be `http://www.myserver.net/streamingclient/update/compound.bin`
- create new text file `http://www.myserver.net/streamingclient/update/update.txt` containing :

Content of the `update.txt`

```
VERSION=1
FW_VERSION =0131
FW_URL=http://www.myserver.net/streamingclient/update/compound.bin
```

- configure your devices “Update URL” field with `http://www.myserver.net/streamingclient/update/update.txt`
- push the “Apply” button on the WEB interface. The device will reboot and automatically update the firmware. If you want the device to check the update file every 30 minutes set the configuration field “Remote Update Period” to 30.

## 7.4 How to configure the device remotely

In previous section we configured the HTTP server and the device for remote update and updated the firmware. In this section we will use the same server and paths but will alter the file `update.txt`.

In this example we will change the streaming URLs and set volume to 25% (Volume range is 0 to 20). Change the `update.txt` as follows:

Content of the `update.txt`

```
VERSION=2
url2=rtp://85.124.188.115:4000
url1=http://vruk.sc.llnwd.net:12265
volume=5
```

The device will change the three configuration fields and then reboot.

In situations where we want to change temporarily (without rebooting) CGI commands can be used instead of configuration change directives. The following example shows how to change the volume without rebooting:

Content of the `update.txt`

```
VERSION=3
v=10
```

This way we can issue any CGI command. See chapter [3.1 CGI WEB command interface](#) for available commands.

### Device dependent update files

Sometimes we need to update devices with different configurations. This can be easily done using the [URL Variable Substitution](#) in the Update URL.

Let us imagine we have two devices and want to load them with different settings (e.g. to play two different radio stations). Let us assume the devices have IP addresses `192.168.2.100` and `192.168.2.101`. Here is an example how to do it:

- Configure both devices with following Update URL: `http://www.myserver.net/streamingclient/update/update-$IP$.txt`
- Create the file `http://www.myserver.net/streamingclient/update/update-192.168.2.100.txt` with following content:

Content of the `update-192.168.2.100.txt`

```
VERSION=1
url1=mms://dwar.wm.llnwd.net/dwar_kcdxfm
url2=file://backup.m3u
```

- Create the file `http://www.myserver.net/streamingclient/update/update-192.168.2.101.txt` with following content:

Content of the `update-192.168.2.101.txt`

```
VERSION=1
url1=http://vruk.sc.llnwd.net:12265
url2=file://backup.m3u
```

If the devices use dynamic addressing (IP might change) use MAC addresses (`$MAC$`) or DHCP names (`$NAME$`) to identify the right configuration file on the server. See [URL Variable Substitution](#) for more details.

## 8 Remote monitoring interface

The “Streaming Client” firmware supports two different ways of remote monitoring: SNMP and MTELL.

SNMP can send a trap on start-up and when switching the stream and can be requested at any time . MTELL sends periodic reporting as well as information on request. The features of these protocols are described in the following chapters.

### 8.1 MTELL Remote Monitoring

The device can be monitored using MTELL technology. Please visit <http://www.mtell.de> for detailed information and to create your own free MTELL project.

#### Configuration Parameters for MTELL periodic report

The MTELL server has to be specified in the configuration field “MTELL URL”. Only the HTTP protocol is supported, including all its options and possibility of using the HTTP Proxy. The “MTELL URL” syntax is:

`http:// [<name>: <password>@] <address>/` (name and password can be omitted, e.g. `http://www.mtell.de/`)

The frequency of MTELL reporting can be set in the configuration field “MTELL Report Period” in minutes. The complete report will be sent in this defined time interval.

The following table show the content of the periodic report.

Value	Type	Description
<b>mac</b>	String (e.g.: 0008E1003D90)	Devices MAC address used for MTELL “sensor” identification
<b>alarm</b>	String	Alarm trigger (not yet used: currently always “false”)
<b>BufferLevel</b>	8bit unsigned decimal number	Amount of bytes in the buffer
<b>FrameLoss</b>	16bit unsigned decimal number	Amount of lost frames
<b>SoftErrorCount</b>	16bit unsigned decimal number	Amount of stream drop-outs (missed more than 5 frames in a row)
<b>StreamNumber</b>	String (0,1,2,3 or “prio” for priority)	Number of played stream
<b>URL</b>	String	URL of the currently played stream
<b>Bitrate</b>	8bit unsigned decimal number	Bitrate of the played stream in kilobits per second
<b>Reconnects</b>	16bit unsigned decimal number	Amount of reconnects due to loss of the stream source
<b>Error</b>	8bit unsigned decimal number	Number of last error (see the definition in the <a href="#">File “BARIXAUDIOSNMP.MIB”</a> )
<b>Volume</b>	8bit unsigned decimal number	Current volume level in percent
<b>UpTime</b>	16bit unsigned decimal number	Up time of the device in seconds (since last reboot)

## Requesting MTELL report

Furthermore, the actual status of the device can be requested using the MTELL protocol over UDP. The Port number used must be the same as specified in the configuration field "MTELL UDP Port". Setting the Port to 0 disables this function.

Sending a UDP datagram with a payload "**MTELL\r\n**", i.e. bytes **0x4D, 0x54, 0x45, 0x4C, 0x4C, 0x0D 0x0A**, will result in a UDP reply sent on same port to the IP address the request originated from. The reply is comma separated and contains no spaces and no line feeds (the table below is word wrapped).

Content of the UDP reply
<b>BufferLevel=34065,FrameLoss=0,SoftErrorCount=0,StreamNumber=2,URL=mms://dms-cl-017.skypro.tv/virus,Bitrate=48,Reconnects=1,Error=404,Volume=50,UpTime=113</b>

To test this we recommend the free PC software called "UDP Test Tool" from <http://www.simplecomtools.com>.

## 8.2 Own Monitoring Server using MTELL protocol

To run an own monitoring server you will need to write your own scripts depending on the server architecture and OS (PHP, ASP...).

The script has to be named "submit" and should be available in the folder "/sensors/data" as the "Streaming Client" firmware sends an HTTP GET request for "sensors/data/submit?..." to that server. This path is fixed and can not be changed. The information is included after the questions mark.

```
GET sensors/data/submit?mac=<mac address>&alarm=false&info=<info> HTTP/1.0
```

<mac address> is 12 hex character string without any delimiters: XXXXXXXXXXXXX e.g.: 0008E1003D90

<info> is a string in the format: **BufferLevel=<int>,FrameLoss=<long int>,SoftErrorCount=<long int>,StreamNumber=<string>,URL=<string>,Bitrate=<int>,Reconnects=<long int>,Error=<int>,Volume=<int>,UpTime=<long int>**

<int> is an 8bit integer decimal number, <long int> is an 16bit integer decimal number, <string> is a string of characters

### Configuration Parameters for MTELL periodic report

The MTELL server has to be specified in the configuration field "MTELL URL". Only the HTTP protocol is supported, including all its options and possibility of using the HTTP Proxy. The "MTELL URL" syntax is:

```
http:// [<name>:<password>@] <address>/ (name and password can be omitted e.g. http://www.myserver.com/)
```

The frequency of MTELL reporting can be set in the configuration field "MTELL Report Period" in minutes. The complete report will be sent in this defined time interval.

**Example "submit.php"**

The submit PHP script can read the variables from the \$\_GET array e.g.:

```
$mac=$_GET["mac"];           // here you can check if the MAC address is registered in your database and decide to accept/ignore this request
$alarm=$_GET["alarm"];      // alarm is always "false"
$info=$_GET["info"];        // comma separated list of "measured values"
```

The \$info variable will contain complete device info which is the string as described in the section above.

The GET variable handling is all standard, there's nothing "MTELL specific", you can access the variables as in any other web CGI script.

**8.3 SNMP Remote Monitoring**

The "Streaming Client" firmware supports SNMPv1 (Simple Network Management Protocol Version 1) which uses UDP for the transfer of information.

**Configuration for SNMP trap sending**

The IP address of the receiver of SNMP traps has to be specified in the configuration field "SNMP Trap Receiver". If set to 0.0.0.0 then no traps are sent. Traps are sent on UDP port 162 to the specified receiver.

**SNMP querying**

The device can be queried using the SNMPv1 protocol on UDP port 161, the MIB (Management Information Base) version supported is 2. The MIB file **BARIXAUDIOSNMP.MIB** is included in the "Streaming Client Update Kit" and can be found in the folder **update\_rescue**. See the following print out of the MIB file for capabilities.

**File "BARIXAUDIOSNMP.MIB"**

Content of the **BARIXAUDIO.MIB** file contained in the "Streaming Client Update Kit"

```
-- The Barix Audio MIB leaf
-- The Barix MIB Registration Authority is barix.mib
-- Version: 2.2
-- Date:    07 March, 2006
-- Copyright (c) 2004-2006 Barix AG

-- Changes:
-- 20050503 KPS    Updated according to Barix MIB registration authority
-- 20060116 KS/PK  Added streaming variables
```

## Content of the BARIXAUDIO.MIB file contained in the "Streaming Client Update Kit"

```
-- 20060307 KS unit net, hostname added
-- 20060307 KS instreaming levels added

BARIXAUDIOSNMP-MIB DEFINITIONS ::= BEGIN

IMPORTS
    enterprises, IPAddress, Counter, TimeTicks, Gauge
        FROM RFC1155-SMI
    OBJECT-TYPE
        FROM RFC-1212
    DisplayString
        FROM RFC-1213;

barix          OBJECT IDENTIFIER ::= { enterprises 17491 }
products      OBJECT IDENTIFIER ::= { barix 1 }
systems       OBJECT IDENTIFIER ::= { barix 2 }
unit          OBJECT IDENTIFIER ::= { barix 3 }
-- 4-9 Spare
oem           OBJECT IDENTIFIER ::= { barix 10 }

-- Audio Section
-- states for dynamic audio states that don't fit into any streaming category
-- streaming for general streaming information
-- exstreaming for specific out to audio information
-- instreaming for specific in from audio information
audio         OBJECT IDENTIFIER ::= { systems 1 }
states        OBJECT IDENTIFIER ::= { audio 1 }
streaming     OBJECT IDENTIFIER ::= { audio 2 }
exstreaming   OBJECT IDENTIFIER ::= { audio 3 }
instreaming   OBJECT IDENTIFIER ::= { audio 4 }

-- unit Group
-- contains information common to all Barix units
--
net           OBJECT IDENTIFIER ::= { unit 1 }
netHostName   OBJECT-TYPE
```

Content of the `BARIXAUDIO.MIB` file contained in the "Streaming Client Update Kit"

```

        SYNTAX      DisplayString (SIZE (0..15))
    MAX-ACCESS      read-only
    STATUS          current
    DESCRIPTION
        "The bootP and DHCP host name"
    ::= { net 1 }

```

```
-- Barix Audio MIB
```

```

audioStateLeft OBJECT-TYPE
    SYNTAX      INTEGER(0..65535)
    ACCESS      read-only
    STATUS      mandatory
    DESCRIPTION "Audio State Left Channel
0 = silence
1 = running
2 = high"
    ::= { states 1 }

```

```

audioStateRight OBJECT-TYPE
    SYNTAX      INTEGER(0..65535)
    ACCESS      read-only
    STATUS      mandatory
    DESCRIPTION "Audio State Right Channel
0 = silence
1 = running
2 = high"
    ::= { states 2 }

```

```

-- streaming
-- Buffer level
streamingBufferLevel OBJECT-TYPE
    SYNTAX      Gauge
    ACCESS      read-only
    STATUS      mandatory
    DESCRIPTION "Streaming buffer level"
    ::= { streaming 1 }

```

## Content of the BARIXAUDIO.MIB file contained in the "Streaming Client Update Kit"

```

-- Frame drop out count
streamingFrameLoss OBJECT-TYPE
    SYNTAX      Counter
    ACCESS      read-only
    STATUS      current
    DESCRIPTION "Lost frames counter"
 ::= { streaming 2 }

-- Stream drift correction counter
streamingSoftErrorCount OBJECT-TYPE
    SYNTAX      Counter
    ACCESS      read-only
    STATUS      current
    DESCRIPTION "Number of soft errors since stream start.
Soft error is:
RTP: lost more frames than could be corrected
TCP, UDP: buffer empty (sampled every 100ms)
"
 ::= { streaming 3 }

-- exstreaming
-- Current Stream number
exstreamingStreamNumber OBJECT-TYPE
    SYNTAX      INTEGER(0..65535)
    ACCESS      read-only
    STATUS      mandatory
    DESCRIPTION "Current stream number
0 - inactive
1 and more - stream number
"
 ::= { exstreaming 1 }

-- Current URL
exstreamingURL OBJECT-TYPE
    SYNTAX      OCTET STRING
    ACCESS      read-only
    STATUS      mandatory

```

## Content of the BARIXAUDIO.MIB file contained in the "Streaming Client Update Kit"

```

        DESCRIPTION "Current URL"
 ::= { exstreaming 2 }

-- Stream bitrate
exstreamingBitrate OBJECT-TYPE
    SYNTAX      INTEGER(0..65535)
    ACCESS      read-only
    STATUS      mandatory
    DESCRIPTION "Stream bitrate in Kbits/sec"
 ::= { exstreaming 3 }

-- Number of reconnects
exstreamingReconnects OBJECT-TYPE
    SYNTAX      Counter
    ACCESS      read-only
    STATUS      mandatory
    DESCRIPTION "Number of reconnects/stream-switches since device startup"
 ::= { exstreaming 4 }

-- Time of last reconnect
exstreamingReconnectTime OBJECT-TYPE
    SYNTAX      TimeTicks
    ACCESS      read-only
    STATUS      mandatory
    DESCRIPTION "Time of last reconnect"
 ::= { exstreaming 5 }

-- Last streaming error
exstreamingError OBJECT-TYPE
    SYNTAX      INTEGER{
        No-Error(0),
        DNS-Problem(1),
        No-TCP-Reply(2),
        TCP-Closed(3),
        No-HTTP-Response(4),
        Invalid-HTTP-Response(5),
        Missing-Path(6),
        Missing-Port-Number(7),
        Missing-Hostname(8),
        Invalid-Filetype(9),
        Filesystem-Error(10),
    }

```

## Content of the BARIXAUDIO.MIB file contained in the "Streaming Client Update Kit"

```

    Connection-Timed-Out (11) ,
    Invalid-Protocol (12) ,
    Too-Many-Dropouts (13) ,
    Invalid-Port (14) ,
    Wrong-Filename (15) ,
    Playlist-Error (16) ,
    Epty-URL (17) ,
    Internal-Error (99) ,
    HTTP-Bad-Request (400) ,
    HTTP-Unauthorized (401) ,
    HTTP-Payment-Required (402) ,
    HTTP-Forbidden (403) ,
    HTTP-Not-Found (404) ,
    HTTP-Method-Not-Allowed (405) ,
    HTTP-Not-Acceptable (406) ,
    HTTP-Proxy-Authentication-Required (407) ,
    HTTP-Request-Time-Out (408) ,
    HTTP-Conflict (409) ,
    HTTP-Gone (410) ,
    HTTP-Length-Required (411) ,
    HTTP-Precondition-Failed (412) ,
    HTTP-Request-Entity-Too-Large (413) ,
    HTTP-Request-URL-Too-Large (414) ,
    HTTP-Unsupported-Media-Type (415) ,
    HTTP-Server-Error (500) ,
    HTTP-Not-Implemented (501) ,
    HTTP-Bad-Gateway (502) ,
    HTTP-Out-of-Resources (503) ,
    HTTP-Gateway-Time-Out (504) ,
    HTTP-Version-not-supported (505)
}
ACCESS          read-only
STATUS          mandatory
DESCRIPTION "Last streaming error
0-90          = connection/configuration errors
90-99         = internal errors
400-599      = HTTP errors"
 ::= { exstreaming 6 }

-- Time of last streaming error
exstreamingErrorTime OBJECT-TYPE

```

## Content of the BARIXAUDIO.MIB file contained in the "Streaming Client Update Kit"

```

        SYNTAX      TimeTicks
        ACCESS      read-only
        STATUS      mandatory
        DESCRIPTION "Time of last error"
 ::= { exstreaming 7 }

-- --- Trap
-- name NOTIFICATION-TYPE
--         OBJECTS      {
--                 Object
--         }
--         STATUS      current
--         DESCRIPTION
--                 ""
--         ::= { OID in MIB tree }
--
--
-- instreaming Levels
--
levels OBJECT IDENTIFIER ::= { instreaming 1 }
audioInputLevelLeft OBJECT-TYPE
        SYNTAX      INTEGER(0..65535)
        ACCESS      read-only
        STATUS      mandatory
        DESCRIPTION "Audio Level Left Channel"
 ::= { levels 1 }

audioInputLevelRight OBJECT-TYPE
        SYNTAX      INTEGER(0..65535)
        ACCESS      read-only
        STATUS      mandatory
        DESCRIPTION "Audio Level Right Channel"
 ::= { levels 2 }

END

```

## 9 Legal Information

---

© 2007 Barix AG, Zurich, Switzerland.

All rights reserved.

All information is subject to change without notice.

All mentioned trademarks belong to their respective owners and are used for reference only.

Barix, Annunicom, Exstreamer, Instreamer, SonicIP and IPzator are trademarks of Barix AG, Switzerland and are registered in certain countries.

For information about our devices and the latest version of this manual please visit [www.barix.com](http://www.barix.com).



Barix AG  
Seefeldstrasse 303  
8008 Zurich

SWITZERLAND

Phone: +41 43 433 22 11  
Fax: +41 44 274 28 49

Internet

web: [www.barix.com](http://www.barix.com)

email: [sales@barix.com](mailto:sales@barix.com)

support: [support@barix.com](mailto:support@barix.com)