



## The Exstreamer Technical Description

Release 2.0

© Barix AG 3/2004, all rights reserved. All information is subject to change without notice. All mentioned trademarks are belonging to their respective owners and are used for reference only. Barix, Exstreamer, SonicIP and IPzator are trademarks or registered trademarks of Barix AG, Switzerland in certain legislations.



THINK FURTHER

## Table of Contents

<b>I</b>	<b>SOFTWARE APPLICATION INTERFACE .....</b>	<b>3</b>
1.1	CONTROL INTERFACE DESCRIPTION .....	3
1.2	CONCATENATE CONTROL COMMANDS .....	3
1.3	RESTRICTIONS.....	3
1.4	PRINCIPLES OF CGI WEB INTERFACE .....	4
1.5	PRINCIPLES OF SERIAL INTERFACE.....	4
1.6	PRINCIPLES OF UDP INTERFACE.....	4
1.7	PRINCIPLES OF TCP INTERFACE.....	4
1.8	PRINCIPLES OF PUSHED STREAMING .....	5
1.9	TCP TO SERIAL AND INVERSELY (SERIAL GATEWAY).....	5
1.10	CONTROL, SERIAL, UDP, TCP AND CGI WEB INTERFACE .....	6
1.11	CONTROL COMMANDS FOR SYNC.....	17
1.12	IR REMOTE CONTROL .....	18
1.13	SETUP.....	20
1.14	OWN SKINS AND WEB INTERFACE.....	23
1.15	GONG APPLICATION (FOR DOOR BELL) .....	28
1.16	IDLE APPLICATION (STREAM DURING IDLE).....	28
1.17	MEMORY PAGE USAGE .....	29
1.18	DYNAMIC WEB PAGE.....	31
1.19	THE DYNAMIC MARK .....	31
1.20	CONFIGURATION VIA HTML PAGES .....	32
1.21	CONFIGURATION LOGOUT .....	39
<b>2</b>	<b>ZSERVER .....</b>	<b>40</b>
2.1	ZSERVER DISCOVERY .....	40
2.2	ZSERVER.INI FILE.....	42
2.3	PLAYLIST DIRECTORY .....	43
2.4	INTERNET RADIO STATIONS.....	43
2.5	PLAYLIST GENERATOR.....	43
<b>3</b>	<b>SPECIFICATION PLAYLISTS .....</b>	<b>44</b>
3.1	LIST OF PLAYLISTS .....	44



- 3.2 PLAYLIST .....45
- 3.3 SONG.....45
- 4 HARDWARE AND CONNECTORS..... 46**
- 4.1 ETHERNET .....46
- 4.2 SERIAL PORT .....46
- 4.3 RTS STATE.....46
- 4.4 PIO3 (NOT FOR EXSTREAMER DIGITAL).....47
- 4.5 CODEC .....47
- 4.6 OTHER CONNECTORS .....48



THINK FURTHER

## I Software Application Interface

### I.1 Control Interface Description

- 0xnn means a hexadecimal number.
  - ↳ means 0x0D 0x0A 0x00 on answers. On requests ↳ could be one or more of the following codes/bytes: 0x0D, 0x0A, 0x00.
- The answers are only echoed to the origin source of the command (not to the other interfaces).
- The answer can be selected by concatenate the L command to the command. If no special answer is requested the file ack.ack will be sent back.
- The answer files can be edited and changed to your needs (see 1.14 Own skins and web interface).
- The standard answers are designed as XML.
- All strings and everything else are case sensitive.
- All commands are asynchronous to the stream.
- One command mustn't exceed 1024 bytes even it is concatenated.

### I.2 Concatenate Control Commands

- To concatenate control commands use &. The commands will be executed from left to right in sequence (not parallel). The ↳ must only be placed at the end of the whole command and not after each separate command.
- To start playing and set volume to 12 use: c=1&v=12↳
- This is useful in the init sequence, in UDP commands or the define the answer. The init sequence is based on the serial command interface.

### I.3 Restrictions

- Streams only MP3 files. MP3Pro files can be streamed but without the additional quality of MP3Pro.

#### 1.4 Principles of CGI WEB interface

- The browser should support frames.
- Use GET method in forms.
- Respect the common character set for URL's.
- Example for CGI WEB commands: `http://x.x.x.x/rc.cgi?c=I` (command for PLAY on Exstreamer x.x.x.x)

#### 1.5 Principles of SERIAL interface

- Default settings of the serial control interface: 9600 baud, 8 data bits, 1 stop bit, no parity
- Each command must be terminated with an ASCII code less than a space 0x20 (like carriage return or/and line feed).
- If the command is correct and could be executed the answer OK is sent back with attached carriage return (ASCII 0x0D) and line feed (ASCII 0x0A).
- ERROR with attached carriage return (ASCII 0x0D) and line feed (ASCII 0x0A) is sent back when:
  - a byte is gets lost
  - an invalid syntax is used
  - the time between two characters exceeds 10 seconds
  - the command is unknown or can't be executed
- There is no time to wait between two commands or characters.
- The serial connector pin out is described in chapter 4.2 Serial Port. It's the same as on a standard PC 9pol. DSub.
- If IR once is used, the serial command interface can't be used anymore.
- If the serial gateway functionality is used, the serial command interface can't be used anymore.

#### 1.6 Principles of UDP interface

- The standard UDP interface port for control commands is 12301.
- Each command must be terminated with a ASCII 0x00, ASCII 0x0D (carriage return) or ASCII 0x0A (line feed).

#### 1.7 Principles of TCP interface

- The standard TCP interface port for control commands is 12302.
- Each command must be terminated with a ASCII 0x00, ASCII 0x0D (carriage return) or ASCII 0x0A (line feed).
- The answers are the same as on the SERIAL interface.

## 1.8 Principles of pushed streaming

- The default TCP streaming port is 2020.
- The default UDP streaming port is 3030.
- In Exstreamer mode 0, 1 and 2 the device has to be stopped (initial state after startup) in order to receive a stream on the streaming ports. In Exstreamer mode 4 the device is always listening on the streaming port.
- Open a TCP connection to the TCP streaming port and send your MP3 file over this connection or send your file as raw UDP packets to the UDP streaming port.
- Send the raw MP3 file/stream in binary mode.
- The TCP streaming port will automatically go into listen mode after the TCP connection has been closed.
- Don't send another stream to the Exstreamer and don't let the Exstreamer stream from a source if a stream is sent to the UDP streaming port.
- Exstreamer mode 0: if no zServer is detected and a stream is received on the streaming port, the Exstreamer mode changes automatically to mode 4 until next startup. This means that an Exstreamer with can be used as streaming receiver after factory defaults.
- Exstreamer mode 0 or 1: if the responding server answers with SYS at the beginning of its version string, the Exstreamer mode changes automatically to mode 4 until next startup.
- The streaming buffer (receiving buffer) is 32 kByte. The parameter "Start Threshold" defines after how many received bytes the Exstreamer will start playing. In order to start the stream earlier, the PLAY command can be sent. If the stream is started too early, it could be that the Exstreamer is running out of data if the network connection is not reliable enough.
- If the streaming buffer runs out of data the Exstreamer stops playing. It will start again after the Start Threshold is reached.
- If another stream should immediately be sent even though a stream is playing already, don't close the TCP connection. Simply send the command FLUSHBUF.

## 1.9 TCP to Serial and inversely (Serial Gateway)

- Open a TCP connection to the local port (default 12302).
- The serial port parameters can be set in the configuration.
- Each byte transmitted to this TCP port is sent to the serial port.
- Each byte received on the serial port is sent to this TCP port.
- Only one TCP port at once can be used.
- As long as this TCP port is open the serial command interface is disabled.
- If IR functionality is once used (for receiving and/or transmitting), don't use the serial gateway without resetting the device .

### 1.10 Control, SERIAL, UDP, TCP and CGI WEB interface

Element	Description	CGI command	SERIAL, TCP or UDP command
ANSWERS	Standard answer file ack.ack will be sent if nothing else is specified with the L command. The file nosupport.ack will be sent on an unknown command. To change the answer concatenate the command GETDYNFILE and chose the needed answer file (that's to build up the same answers of the old Exstreamer versions below 6.00). ex. set volume: v=4&L=volume.ack ex. get state of CTS: L=getcts.ack (this is the replacement of all old get commands)		see the files in 1.14 Own skins and web interface
PLAY	If no playlist is selected, first song of first playlist will play. If state is pause, song will continue. If state is play, nothing happens. After startup the device is in STOP mode.	c=1	0x63 0x3D 0x31 0x00(c=1.)
STOP	If state is play or pause, playing will stop otherwise nothing happens.	c=2	0x63 0x3D 0x32 0x00 (c=2.)
PAUSE	If state is play, playing will stop. If state is pause, play will continue.	c=3	0x63 0x3D 0x33 0x00 (c=3.)
NEXTSONG	If no playlist is selected, first song of first playlist will play. If state is play, next song starts playing (in sequence or random order).	c=4	0x63 0x3D 0x34 0x00 (c=4.)
PREVSONG	If no playlist is selected, first song of last playlist will play. If state is play, current song restarts playing. If state is play and this command is sent two times in 2 seconds previous song starts playing (only if shuffle off).	c=5	0x63 0x3D 0x35 0x00 (c=5.)
SHUFFLEON	NEXTSONG selects song in actual playlist in random order. If shuffle is pressed during PLAY, the current song will be played to the end and the next song will be chosen in random order.	c=6	0x63 0x3D 0x36 0x00 (c=6.)

SHUFFLEOFF	NEXTSONG, PREVSONG selected in sequential order. If shuffle off is pressed during PLAY, the current song will be played to the end the next song will be played in order	c=7	0x63 0x3D 0x37 0x00 (c=7.↓)
MUTE	Toggle between mute and volume. MUTE stores the last volume and resets to this value on turn on volume. VOLUMEINC, VOLUMEDEC, VOLUME > 0 as well as FORCEMUTEOFF unmutes the device.	c=8	0x63 0x3D 0x38 0x00 (c=8.↓)
LOUDNESSON	Turn on loudness.	c=9	0x63 0x3D 0x39 0x00 (c=9.↓)
LOUDNESSOFF	Turn off loudness.	c=10	0x63 0x3D 0x31 0x30 0x00 (c=10.↓)
VOLUMELOCK	After this command the volume can't be changed until you unlock it.	c=11	0x63 0x3D 0x31 0x31 0x00 (c=11.↓)
VOLUMEUNLOCK	Unlock volume.	c=12	0x63 0x3D 0x31 0x32 0x00 (c=12.↓)
SETASDEFAULT	Store current values (volume, volume lock, mute, balance, bass, treble, loudness level, loudness on, shuffle, repeat) as default on startup.	c=13	0x63 0x3D 0x31 0x33 0x00 (c=13.↓)
FACTORYDEFAULTS	Set factory default values for the current runtime configuration.	c=14	0x63 0x3D 0x31 0x34 0x00 (c=14.↓)
NEXTPLAYLIST	Select next playlist. If currently no playlist is selected, first playlist will be chosen.	c=15	0x63 0x3D 0x31 0x35 0x00 (c=15.↓)
PREVPLAYLIST	Select previous playlist. If currently no playlist is selected, last playlist will be chosen.	c=16	0x63 0x3D 0x31 0x36 0x00 (c=16.↓)
FASTFORWARD	Fasten the stream approx. two times until a new song is played or the commands PLAY, STOP or an other time FASTFORWARD.	c=17	0x63 0x3D 0x31 0x37 0x00 (c=17.↓)
FASTREWIND	not yet implemented.	c=18	0x63 0x3D 0x31 0x38 0x00 (c=18.↓)
VOLUMEINC	Increment volume one step.	c=19	0x63 0x3D 0x31 0x39 0x00 (c=19.↓)
VOLUMEDEC	Decrement volume one step.	c=20	0x63 0x3D 0x32 0x30 0x00 (c=20.↓)
BALANCERIGHT	Set balance one step more to right.	c=21	0x63 0x3D 0x32 0x31 0x00 (c=21.↓)
BALANCELEFT	Set balance one step more to left.	c=22	0x63 0x3D 0x32 0x32 0x00 (c=22.↓)
BASSINC	Increment bass level one step.	c=23	0x63 0x3D 0x32 0x33 0x00 (c=23.↓)
BASSDEC	Decrement bass level one step.	c=24	0x63 0x3D 0x32 0x34 0x00 (c=24.↓)
TREBLEINC	Increment treble level one step.	c=25	0x63 0x3D 0x32 0x35 0x00 (c=25.↓)
TREBLEDEC	Decrement treble level one step.	c=26	0x63 0x3D 0x32 0x36 0x00 (c=26.↓)



SERIALBIN	Select Serial B as input source (digital MP3 data).	c=27	0x63 0x3D 0x32 0x37 0x00 (c=27_↓)
LINEIN	Select Line In as input.	c=28	0x63 0x3D 0x32 0x38 0x00 (c=28_↓)
MICIN	Select Mic In as input.	c=29	0x63 0x3D 0x32 0x39 0x00 (c=29_↓)
SHUFFLE	Toggle between SHUFFLEON and SHUFFLEOFF.	c=30	0x63 0x3D 0x33 0x30 0x00 (c=30_↓)
	The commands c=31 until c=38 aren't implemented. Use the files in 1.14 Own skins and web interface with the command GETDYNFILE as replacement of the command set of the Exstreamer versions below 6.00.		
FORCEPAUSE	playing will stop (no toggle)	c=39	0x63 0x3D 0x33 0x39 0x00 (c=39_↓)
FORCEMUTEON	mute on forced (no toggle), details see MUTE command	c=40	0x63 0x3D 0x34 0x30 0x00 (c=40_↓)
FORCEMUTEOFF	mute off forced (no toggle)	c=41	0x63 0x3D 0x34 0x31 0x01 (c=41_↓)
LOUDNESSINC	Increment loudness one step.	c=42	0x63 0x3D 0x34 0x32 0x00 (c=42_↓)
LOUDNESSDEC	Decrement loudness one step.	c=43	0x63 0x3D 0x34 0x32 0x00 (c=43_↓)
FLUSHBUF	Flushes the streaming buffer (without closing TCP connection)	c=44	0x63 0x3D 0x34 0x34 0x00 (c=44_↓)
PRGFEEDBACKON	Progress feedback on Sends back every second in the TCP stream connection (not on the TCP command interface) the actual streamed byte counter terminated by a 0x0D 0x0A. The decimal byte counter (unsigned long) starts at 0 with each new stream. This command turns off the second feedback in the TCP stream.	c=45	0x63 0x3D 0x34 0x35 0x00 (c=45_↓)
PRGFEEDBACKOFF	Progress feedback off (see PRGFEEDBACKON) Turns off all progress feedbacks and resets the seconds and byte counter.	c=46	0x63 0x3D 0x34 0x36 0x00 (c=46_↓)
STEREOOUT	Set output to stereo	c=47	0x63 0x3D 0x34 0x37 0x00 (c=47_↓)
MONOOUT	Set output to mono	c=48	0x63 0x3D 0x34 0x38 0x00 (c=48_↓)
BRIDGEOUT	Set output to bridge mode. This makes sense only for the line out.. This will be achieved by set the output to mono and invert the right channel.	c=49	0x63 0x3D 0x34 0x39 0x00 (c=49_↓)
REPEATOFF	Disables repeat.	c=50	0x63 0x3D 0x35 0x30 0x00 (c=50_↓)
REPEATSONG	Repeats the actual song forever.	c=51	0x63 0x3D 0x35 0x31 0x00 (c=51_↓)
REPEATPLAYLIST	Repeats the actual playlist forever.	c=52	0x63 0x3D 0x35 0x32 0x00 (c=52_↓)



PRGFEEDBUDPSEC	Progress feedback in seconds Every second a broadcast packet in the subnet is sent to the UDP command port as destination port with the elapsed seconds (decimal unsigned long) of the actual song (remains empty until first data received). Use PRGFEEDBACKOFF to stop the feedback. The periodical answer is the file fbsec.ack. If no UPD command port is set this feature will not work.	c=53	0x63 0x3D 0x35 0x33 0x00 (c=53.↓)
PRGFEEDBUDPBYTE	Progress feedback in bytes Every second a broadcast packet in the subnet is sent to the UDP command port as destination port. Use PRGFEEDBACKOFF to stop the feedback. The periodical answer is the same as for PRGFEEDBACKON. If no UPD command port is set this feature will not work.	c=54	0x63 0x3D 0x35 0x34 0x00 (c=54.↓)
PRGFEEDBSERSEC	Progress feedback in seconds Every second a packet is sent to the serial port (answer see PRGFEEDBUDPSEC).	c=55	0x63 0x3D 0x35 0x35 0x00 (c=55.↓)
PRGFEEDBSERBYTE	Progress feedback in bytes Every second a packet is sent to the serial port (answer see PRGFEEDBUDPBYTE).	c=56	0x63 0x3D 0x35 0x36 0x00 (c=56.↓)
PRGFEEDBTCPSEC	Progress feedback on Sends every second in the TCP stream (not on the TCP command interface) the elapsed seconds (decimal unsigned long) of the actual song terminated by a 0x0D 0x0A. This command turns off the byte feedback in the TCP stream.	c=57	0x63 0x3D 0x35 0x37 0x00 (c=57.↓)
PITCHINC	Increment pitch level one step.	c=58	0x63 0x3D 0x35 0x38 0x00 (c=58.↓)
PITCHDEC	Decrement pitch level one step.	c=59	0x63 0x3D 0x35 0x39 0x00 (c=59.↓)
SETRTS	Sets output RTS to logic 1 (-12V) Set the parameter RTS usage to off first.	c=60	0x63 0x3D 0x36 0x30 0x00 (c=60.↓)
RESETRTS	Sets output RTS to logic 0 (+12V) Set the parameter RTS usage to off first.	c=61	0x63 0x3D 0x36 0x31 0x00 (c=61.↓)

	The commands c=62 and c=63 aren't implemented. Use the files in 1.14 Own skins and web interface with the command GETDYNFILE as replacement of the command set of the Exstreamer versions below 6.00.		
RESETLASTCTS	Resets last state of CTS.	c=64	0x63 0x3D 0x36 0x34 0x00 (c=64.)
PRGFEEDBCTCPSEC	Progress feedback in seconds Every second a packet is sent to the TCP command interface. (answer see PRGFEEDBUDPSEC)	c=65	0x63 0x3D 0x36 0x35 0x00 (c=65.)
PRGFEEDBCTYPBYTE	Progress feedback in bytes Every second a packet is sent to the TCP command interface. (answer see PRGFEEDBUDBYTE)	c=66	0x63 0x3D 0x65 0x36 0x00 (c=66.)
	The commands c=67 and c=68 aren't implemented. Use the files in 1.14 Own skins and web interface with the command GETDYNFILE as replacement of the command set of the Exstreamer versions below 6.00.		
RESETLASTPIO	Resets last state of PIO (see 4.4 PIO3 (not for Exstreamer Digital)).	c=69	0x63 0x3D 0x36 0x39 0x00 (c=69.)
	The commands c=70 until c=72 aren't implemented. Use the files in 1.14 Own skins and web interface with the command GETDYNFILE as replacement of the command set of the Exstreamer versions below 6.00.		
STATEFEEDBKUDPON	Enables automatic state feedback (on change of song, shuffle, progress feedback, repeat, volume, volume lock, mute, loudness, balance, bass, treble) on the UDP command interface. The answer is the file getstate.ack.	c=73	0x63 0x3D 0x37 0x33 0x00 (c=73.)
STATEFEEDBKSERON	Enables automatic state feedback on the serial port (see STATEFEEDBKUDPON).	c=74	0x63 0x3D 0x37 0x34 0x00 (c=74.)
STATEFEEDBKTCPON	Enables automatic state feedback on the TCP command interface (see STATEFEEDBKUDPON).	c=75	0x63 0x3D 0x37 0x35 0x00 (c=75.)
STATEFEEDBKOFF	Disables automatic state feedback on all interfaces.	c=76	0x63 0x3D 0x37 0x36 0x00 (c=76.)
REPEAT	Toggles between REPEATSONG, REPEATPLAYLIST, REPEATOFF	c=77	0x63 0x3D 0x37 0x37 0x00 (c=77.)
VOLLOCKTOGGLE	Toggles between VOLUMELOCK and VOLUMEUNLOCK	c=78	0x63 0x3D 0x37 0x38 0x00 (c=78.)
LOUDNESSTOGGLE	Toggles between LOUDNESSON and LOUDNESSOFF	c=79	0x63 0x3D 0x37 0x39 0x00 (c=79.)



DEFAULTS	Set factory defaults without the network settings (without own IP, Gateway, Netmask, DNS, WLAN mode, WLAN name, WEP mode, WEP key and SonicIP) and reboots the device.	c=80	0x63 0x3D 0x38 0x30 0x00 (c=80↓)
	The commands c=81 until c=96 aren't implemented.		
GETNEWPLAYLISTS	Rereads the playlists from the zServer starting with playlist number 0 into the playlist buffer.	c=97	0x63 0x3D 0x39 0x37 0x00 (c=97↓)
DETECTZSERVER	Rediscover the zServer.	c=98	0x63 0x3D 0x39 0x38 0x00 (c=98↓)
DEVICERESET	Hard reboot of device.	c=99	0x63 0x3D 0x39 0x39 0x00 (c=99↓)
BOOTLOADER	Starts the bootloader. The application will be left. It isn't running until the next reboot.	c=100	0x63 0x3D 0x31 0x30 0x30 0x00 (c=100↓)
DISCOVER	If this command is received the device answers with the file discover.ack.	c=65535	0x63 0x3D 0x36 0x35 0x35 0x33 0x35 0x00 (c=65535↓)
PASSWORD	Concatenate this command to the rest of the command sequence if the command interface is password (level 3) protected. The password has to be added in plain text.	a=...	0x61 0x3D ... 0x00 (a=...↓)
BALANCEL10	Set balance to left (right off).	b=L10	0x62 0x3D 0x4C 0x31 0x30 0x00 (b=L10↓)
BALANCEL09	Set balance to level 9 on left side.	b=L9	0x62 0x3D 0x4C 0x39 0x00 (b=L9↓)
---		---	---
BALANCE00	Set balance to middle.	b=L0	0x62 0x3D 0x4C 0x30 0x00 (b=L0↓)
---		---	---
BALANCER09	Set balance to level 9 on right side.	b=R9	0x62 0x3D 0x52 0x39 0x00 (b=R9↓)
BALANCER10	Set balance to right (left off).	b=R10	0x62 0x3D 0x52 0x31 0x30 0x00 (b=R10↓)
	Corresponding values sent to the codec for 0..10: 0x00, 0x01, 0x02, 0x03, 0x04, 0x05, 0x06, 0x07, 0x08, 0x2E, 0x7E (corresponding neg. value for the other side)		

BASSM10	Set minimal bass level.	B=-10	0x42 0x3D 0x2D 0x31 0x30 0x00 (B=-10↓)
BASSM09	Set bass level to -9.	B=-9	0x42 0x3D 0x2D 0x39 0x00 (B=-9↓)
---		---	---
BASSP00	Set neutral bass level.	B=+0 or B=0	0x42 0x3D 0x2B 0x30 0x00 (B=+0↓)
---		---	---
BASSP09	Set bass level to 9.	B=+9 or B=9	0x42 0x3D 0x2B 0x39 0x00 (B=+9↓)
BASSP10	Set maximal bass level. Corresponding values sent to the codec for 0..10: 0x00, 0x09, 0x13, 0x1D, 0x26, 0x30, 0x3A, 0x43, 0x4D, 0x56, 0x5B, 0x60 (corresponding neg. value for neg. bass)	B=+10 or B=10	0x42 0x3D 0x2B 0x31 0x30 0x00 (B=+10↓)
PITCHP100	Set maximal pitch level (+10%).	i=+100 or i=100	0x69 0x3D 0x31 0x30 0x30 0x00 (i=100↓)
PITCHP099	Set pitch level to 9.9%.	i=+99 or i=99	0x69 0x3D 0x39 0x39 0x00 (i=-99↓)
---		---	---
PITCH000	Set neutral pitch level (0%).	i=+0 or i=0	0x69 0x3D 0x30 0x00 (i=0↓)
---		---	---
PITCHM69	Set pitch level to -9.9%.	i=-69	0x69 0x3D 0x2D 0x36 0x39 0x00 (i=-69↓)
PITCHM70	Set minimal pitch level (-10%). Value sent to the codec = $4800 + ((pitch/100) * pitch)$ ;	i=-70	0x69 0x3D 0x2D 0x37 0x30 0x00 (i=-70↓)

<p>SENDIR</p>	<p>Transmit an IR command.          This command is word organized (2 bytes, high byte first). The words can be separated by a comma (no spaces). The word is interpreted as a hexadecimal value.          The first word is the selection for the following IR command. It is organized in bits.          Low Byte:     0x01 = SONY IR modulation                            0x02 = RC 5 IR modulation                            0x04 = NEC IR modulation                            0x80 = RAW IR modulation          Bit 8:    1 for 37.5 kHz modulation frequency                    0 for base band          Bit 15:  0 for transmit on serial port 0 (Serial IR Dongle)                    1 for transmit on IR OUT (Exstreamer Digital only)          Example: 8102 for RC 5 IR modulation with 37.5 kHz on IR OUT.</p> <p>The following word are depended of the IR modulation:          SONY IR: Add the bits (0 or 1) to transmit starting with the first bit sent to IR as nibbles of the words in the IR command. If the count of bits is odd than add an f to have it even. The sync and end of transmission bits mustn't be added.          Example: j=0101,1010001001010f          RC 5: see SONY IR          Example: j=0102,10001001010f          NEC: Add the bytes to transmit starting with the first byte to sent. The bits inside the byte will be swapped inside this function for the transmission of the LSB first on IR.          Example: j=0104,00FE7887          RAW: The first word after the selection word, is the active 1 (IR on) time in 100 us. The second word is the active 0 (IR off) time in 100 us. The next is again for active 1 and so on until a word has the value 0000. This is to build every possible IR command.          Example: j=0180,0018,0004,000c,0008,0000</p>	<p>j=...</p>	<p>0x70 0x3D ... 0x00 (j=...)</p>
---------------	---	--------------	-----------------------------------



LOUDNESS00 LOUDNESS01 --- LOUDNESS20	Set minimal loudness level. Set loudness level l.  Set maximal loudness level. Corresponding values sent to the codec for 0..20: 0, 3, 6, 9, 12, 15, 18, 21, 24, 27, 30, 33, 36, 39, 42, 46, 50, 56, 60, 64, 68	l=0 l=1 --- l=20	0x6C 0x3D 0x30 0x00 (l=0.) 0x6C 0x3D 0x31 0x00 (l=1.) --- 0x6C 0x3D 0x32 0x30 0x00 (l=20.)
GETDYNFILE	The response is the dynamic file stored in a cob file (see 1.14 Own skins and web interface) with that name. Example: L=getstate.ack	L=...	0x4C 0x3D ... 0x00 (L=....)
PLAYROMSONG	Plays a song out of a flash page embedded in a web application.	m=...	0x6D 0x3D ... (m=....)
PLAYLISTSCOUNTTO BUF	Gets the buffer block of the playlists where the selected number is included from zServer and stores it to the playlists buffer. This counter is the global counter of all playlists available. The buffer can only store 4 kBytes of playlist names. If there are more playlists than there is space in the playlists buffer you can load the desired block by specifying a playlist number within the block with this command.	n=l --- n=65535	0x6E 0x3D 0x31 0x00 (n=l.) --- 0x6E 0x3D 0x36 0x35 0x35 0x33 0x35 0x00 (n=65535.)
PLAYLISTSCHARTO BUF	Gets the playlists starting with selected character from zServer and stores it to playlists buffer. If the character isn't available the next one will be taken.	N=A --- N=Z	0x4E 0x3D 0x41 0x00 (N=A.) --- 0x4E 0x3D 0x5A 0x00 (N=Z.)
VIEWPLAYLIST	Selection for viewing only a playlist (see PLAYPLAYLIST).	p=l --- p=65535	0x70 0x3D 0x31 0x00 (p=l.) --- 0x70 0x3D 0x36 0x35 0x35 0x33 0x35 0x00 (p=65535.)

PLAYPLAYLIST	<p>Selection for playing a playlist. First song of playlist will play immediate. Playlists are numbered in sequence from 1 to max. 65535, even if there are more playlists than space is in the playlists buffer (4 kBytes). (see PLAYLISTSCOUNTTOBUF)</p> <p>Direct playing of a playlist from any web server. Give a URL after P= to let the device get and play this playlist (m3u) from a web server with a HTTP GET command.</p>	<p>P=1 --- P=65535  P=http://...</p>	<p>0x50 0x3D 0x31 0x00 (P=1↵) --- 0x50 0x3D 0x36 0x35 0x35 0x33 0x35 0x00 (P=65535↵) 0x50 0x3D 0x68 0x74 0x74 0x70 0x3A 0x2F 0x2F ... (P=http://... ↵)</p>
SENDUDPSTRING	<p>Send a command string in an UDP packet Syntax: r=[&lt;prot&gt;://][[&lt;ip&gt;]:&lt;port&gt;]/&lt;cmd&gt; &lt;prot&gt; = udp &lt;cmd&gt; = [&lt;param&gt;]{&amp;&lt;cmd&gt;}</p> <p>If no port is defined then udp is used. If no ip is in the command then the IP address of the last received stream is used. If no port is defined then the UDP command port is used</p> <p>Example: r=udp://192.168.0.22:12301/v=2 is the same as r=192.168.0.22:12301/v=2 or r=192.168.0.22/v=2 or r=v=1 if the last received stream is from 192.168.0.22.</p>	r=...	0x72 0x3D (r=...↵)
PLAYSONG	<p>Direct selection of a song in a playlist. Songs are numbered in sequence started with 1.</p> <p>Direct playing of a song from any web server. Give a URL after S= to let the device get this song from a web server with a HTTP GET command.</p>	<p>S=1 --- S=999 S=http://...</p>	<p>0x53 0x3D 0x31 0x00 (S=1↵) --- 0x53 0x3D 0x39 0x39 0x39 0x00 (S=999↵) 0x53 0x3D 0x68 0x74 0x74 0x70 0x3A 0x2F 0x2F ... (S=http://... ↵)</p>

TREBLEM10 TREBLEM09 --- TREBLEP00 --- TREBLEP09 TREBLEP10	Set minimal treble level. Set treble level to -9.  Set neutral treble level.  Set treble level to 9. Set maximal treble level. Corresponding values sent to the codec for 0..10: 0x00, 0x09, 0x13, 0x1D, 0x26, 0x30, 0x3A, 0x43, 0x4D, 0x56, 0x5B, 0x60 (corresponding neg. value for neg. treble)	t=-10 t=-9 --- t=+0 or t=0 --- t=+9 or t=9 t=+10 or t=10	0x74 0x3D 0x2D 0x31 0x30 0x00 (t=-10.↓) 0x74 0x3D 0x2D 0x39 0x00 (t=-9.↓) --- 0x74 0x3D 0x2B 0x30 0x00 (t=+0.↓) --- 0x74 0x3D 0x2B 0x39 0x00 (t=+9.↓) 0x74 0x3D 0x2B 0x31 0x30 0x00 (t=+10.↓)
SENDTCPSTRING	Sends the attached string through the TCP interface. The answer will be the string itself.	T=...	0x53 0x3D (T=...↓)
VOLUME00 VOLUME01 --- VOLUME20	Set minimal volume level (volume off). Set volume level 1.  Set maximal volume level. Corresponding values sent to the codec for 0..20: 0, 76, 80, 84, 88, 92, 96, 99, 102, 106, 108, 110, 112, 114, 116, 118, 120, 124, 125, 126, 127 One step is showed as 5%. The level 0 equals the 0%.	v=0 v=1 --- v=20	0x76 0x3D 0x30 0x00 (v=0.↓) 0x76 0x3D 0x31 0x00 (v=1.↓) --- 0x76 0x3D 0x32 0x30 0x00 (v=20.↓)

## 1.11 Control Commands for Sync

This commands are licensed with the Exstreamer Digital or if the license key is stored in the device. Don't use these without the corresponding license.

Element	Description	CGI command	SERIAL, TCP or UDP command
SYNC	<p>This command synchronizes an Exstreamer as slave to another Exstreamer which then will be the master.</p> <p>IP: IP address of the master Exstreamer (ex. 192.168.2.50)</p> <p>PORT: Sync port of the master Exstreamer (ex. 20562). This port can be calculated but normally it should be taken from the answer of the DISCOVER command. The formula is:  <math>20000 + (\text{second to last byte of IP bit and } 0x0F) \times 256 + \text{last byte of IP}</math>  (ex. <math>20000 + (2 \times 256) + 50 = 20562</math>)</p> <p>NAME: Name of the master Exstreamer (ex. Master). It can be taken from the answer of the DISCOVER command.</p>	<code>s=&lt;IP&gt;&amp;port=&lt;PORT&gt;&amp;name=&lt;NAME&gt;</code>	<code>0x73 0x3D ... 0x00 (s=...)</code>
SYNCEND	<p>Finish the current synchronized connection to the master.</p>	<code>c=103</code>	<code>0x63 0x3D 0x31 0x30 0x33 0x00 (c=103)</code>

## 1.12 IR Remote Control

The commands on each button of the Remote Control can be user defined. In order to do this use the Exstreamer App Development Kit (see 1.14 Own skins and web interface). The remote.ini file in the development kit contains the commands for each button. Every line is for one button. The file format is comma-separated and the values are case-sensitive. Don't write spaces between the separation.

The 1st field is the IR remote control command (ex. 'NE: 00FE7887' for the play button) received from the remote control. To figure out the exact command for a button you can open the status web page on the Exstreamer with <http://x.x.x.x/status> where x.x.x.x is the IP address of the Exstreamer connected to the IR Remote Control Kit. The last received IR command is displayed right beside the text 'ir'.

A '\*' in the 2nd field tells the IR handler that this button is accepted for repetition.

An empty 3rd field tells the IR handler to simply execute the command (see 1.10 Control, SERIAL, UDP, TCP and CGI WEB interface) in the 4th field. It uses the same command set as TCP or SERIAL.

An 'L' in the 3rd field tells the IR handler that the received IR remote control command has to be sent as broadcast through the UDP command interface (if configured) following the local execution on the Exstreamer.

An 'G' in the 3rd field tells the IR handler that the received IR remote control command only has to be sent as broadcast through the UDP command interface (if configured) without the local execution on the Exstreamer.

If all unknown IR commands should be sent through the UDP command interface add the following line: `?: *, , ,`

An 'L' following by a decimal number tells the IR handler to select to numbered level for the next command. To select the command for the corresponding level simply add more fields at the end of the line for that button. The 5th field has the level 1, the 6th field has the level 2 and so on. The max. level is 255. The selected level is declined after 2 sec or after the next button will be pressed.

Every line that doesn't contain an IR remote control command is handled as a comment.

Examples (1st line executes PLAY by pushing the play button on the remote control, 2nd line increases volume, 3rd line sets level 1 (button \*), 4th line plays playlist 1 (button I), if level 1 play playlist 11):

```
NE: 00FE7887, , , c=1
NE: 00FEA857, *, , c=19
NE: 00FED02F, , , L1
NE: 00FE40BF, , L, P=1, P=11
```



Code table for Barix remote control buttons

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



THINK FURTHER

### 1.13 Setup

The factory default setup is contained in the binary file config.bin. This file can be edited with a hex editor. Be careful if you do changes. This file will be loaded to the EEPROM on factory default.

#### General Terms (EEPROM Organization)

- IP addresses are always stored with the highest byte at the lowest address.
- Strings are coded in ASCII and terminated with 0x00. The Length includes the termination.
- Values are stored in little endian format (Intel) (low byte first)
- All Values are integer.
- Signed values are stored in 2-complement.
- Unused bytes must be set to 0x00.

In the following table the column Byte shows the byte number in the 1016 bytes of configuration. The first byte has the number 0. If a password is set then the device only will answer to this commands if the password in the command is set correct.

Parameter	Byte [dec]	Length [Byte]	Default Value	Short Description
Own IP	0	4	0.0.0.0	Static IP address of the device. 0.0.0.0 for DHCP.
Gateway IP	4	4	0.0.0.0	Gateway IP address. 0.0.0.0 for no gateway
Netmask	8	1	0	Subnetmask. The value is the count of the zero bits counted from the lowest byte. (ex. 8 for 255.255.255.0)
WLAN Mode	9	1	1	1 for infrastructure mode, 0 for ad-hoc mode
WLAN Name	10	33		SSID of the wireless network. If it is empty then the device connects to the first wlan it will find.
WEP Mode	43	2	0	0 for encryption off, 1 for encryption on
WEP Key	45	14		To activate the 40 bit encryption use exactly 5 bytes (rest 0x00), for 128 bit encryption use exactly 13 bytes (last byte 0x00). The key always is stored binary.
DNS 1	64	4	0.0.0.0	Primary DNS IP address. 0.0.0.0 for no primary and no secondary DNS.
DNS 2	68	4	0.0.0.0	Alternative DNS IP address. 0.0.0.0 for no secondary DNS.
Password	72	8		Password as String. Used in the Telnet Setup.

IFMODE0	80	1	0x4C	Definition of the bits in that byte for the serial port 0:								
				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	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)								
FLOWCONTROLO	82	1	0	Flow control for the serial port 0. (0 = no, 1 = Software XON/XOFF, 2 = Hardware RTS/CTS)								
LOCALPORT	86	2	0	Port for the serial gateway (0 for disable).								
BOOTTARGET	94	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.								
Version Major	116	1	1	Version Major value (do not change)								
Version Minor	117	1	3	Version Minor value (do not change)								
Setupex Length	120	2	894	Length of the extended setup (always 894)								
Password Level 1	122	8		Password stored as MD5 hash (first 8 bytes) used for save configuration via web								
Password Level 2	130	8		Password stored as MD5 hash (first 8 bytes) used for view the configuration via web								
Password Level 3	138	8		Password stored as MD5 hash (first 8 bytes) used for control/commands								
Password Level 4	146	8		Password stored as MD5 hash (first 8 bytes)								
Password Level 5	154	8		Password stored as MD5 hash (first 8 bytes)								
Password Level 6	162	8		Password stored as MD5 hash (first 8 bytes)								
Server IP	240	4	0.0.0.0	IP address of the server (in Exstreamer mode 0 and 1 the zServer, in mode 2 the Web server, in mode 3 the IP address of the host)								
Volume	244	1	8	Volume 0..20								
Balance	245	1	0	Balance -10..10								

Bass	246	1	0	Bass -10..10
Treble	247	1	0	Treble -10..10
Loudness	248	1	20	Loudness 0..20
Repeat	249	1	0	0 for repeat of, 1 for repeat one song, 2 for repeat playlist
Output Mode	250	1	0	0 for stereo, 1 for mono, 2 for bridge (mono)
Stream Listen Port	251	2	2020	Streaming listen TCP port
RTS Mode	253	1	0	0 = RTS is used to show the receiving of an IR command and lets blink the LED on the Serial IR Dongle 1 = RTS is used to power on or off an AVR (see 4.3 RTS State) 2 = RTS is not used for something special
Server Port	254	2	0	Port of the server (in Exstreamer mode 0 and 1 the zServer's port, in mode 2 the Web server,'s port in mode 3 the host's port). 0 for default (8888 in mode 0 and 1, 80 in mode 3)
Device Name	256	21		Name of the device
Media Configuration	277	1	0x00	This values can be added (the function is activated by set the bit): 0x01: random on 0x02: not used 0x04: locks volume 0x08: loudness on (level see parameter loudness) 0x10: mute 0x20: walk through playlists 0x40: not used 0x80: no SonicIP
Playlist Path Match	286	33		If the path of a song in a playlist starts with this string, it will be replaced with the Playlist Substitution string
Playlist Substitution	319	33		see Playlist Path Match
Mode	352	1	0	0 for zServer autodetection, 1 for zServer fixed IP, 2 for Web Server, 3 for Streaming Puller, 4 for Streaming Receiver
Server Path	353	33	/	In mode 0 and 1 always /, in mode 2 the path part of the URL to the playlists, in mode 3 the path part of the URL to the MP3 file.
Playlist Path	386	33	/playlist.html	In mode 0 and 1 always /playlist.html, in mode 2 the path part of the URL to the list of playlists
Init Sequence	420	33		String of commands (like serial command interface) which is executed after each startup of the device.
IR Gateway IP	491	4	0.0.0.0	IP address for transmitting all received IR remote control commands via UDP j= commands to the other devices 'IR Out' output. Disabled if set to 0.0.0.0 and IR Port 0

IR Gateway Port	495	2	0	Port for transmitting all received IR remote control commands (see IR Gateway IP). If IR Gateway is set and this port is 0 the UDP command port is used. If the UDP command port is set to 0 the default UDP command port 12301 is used.
IR Source	497	1	0	0 = Serial IR Dongle, 1 = IR IN (Exstreamer Digital only)
Cfg Store Timeout	498	1	0	Timeout in seconds after the last change of shuffle, repeat, volume, volume lock, mute, balance, bass, treble and/or loudness the setup automatically will be stored into the EEPROM.
UDP RX Port	499	2	0	UDP streaming receiver port (0 for disable)
Start Threshold	509	2	25000	Amount of bytes received in the streaming buffer until the stream starts playing
UDP Command Port	511	2	12301	Receiving port for the UDP command interface (0 for disable)
TCP Command Port	513	2	12302	Listening port for the TCP command interface (0 for disable)
Preset	515	2	0	User-specific storage, this parameter has no functionality. It can be used by the user for the web interface.

### 1.14 Own skins and web interface

With the Exstreamer Digital App Development Kit or the Exstreamer App Development Kit (DOS) you can design your own web pages (skin) and modify the answers to your needs. This kit is available on [www.barix.com](http://www.barix.com) or [www.exstreamer.com](http://www.exstreamer.com).

The stuff and help folder hold the files you need for the web pages. You can simply edit these files and/or add new ones. The web interface (and the firmware) need at least the following files:

File	Dyn.	Description
Version file		
EXSTREAMERAPPVERSION		for the version number and the history

Answer files (see 1.20 Configuration via HTML Pages for the dynamic marks contained in the files and the exact description) Use these files in with the command GETDYNFILE as replacement of the GET commands of the Exstreamer versions below 6.00.

ack.ack		standard answer for commands
balance.ack	✓	current value of the balance
bass.ack	✓	current value of the bass
discover.ack	✓	answer for the DISCOVER command
fbbyte.ack	✓	answer for the progress feedback in seconds for the commands PRGFEEBUDPBYTE, PRGFEEBDBSERBYTE and PRGFEEBDBCTYPBYTE

fbsec.ack	✓	answer for the progress feedback in seconds for the commands PRGFEEDBUDPSEC, PRGFEEDBSERSEC and PRGFEEDBCTYPSEC
getactplaylist.ack	✓	content of the actual played playlist
getactplaylistname.ack	✓	name of the actual played playlist
getcts.ack	✓	actual state of CTS
getlastcts.ack	✓	Returns 0 if CTS has changed since the last RESETLASTCTS command. If CTS hasn't changed it returns 1.
getlastir.ack	✓	last received IR command
getlastpio.ack	✓	Returns 0 if PIO3 has changed since the last RESETLASTPIO command. If PIO3 hasn't changed it returns 1.
getpio.ack	✓	actual state of PIO (see 4.4 PIO3 (not for Exstreamer Digital))
getplaylists.ack	✓	outputs the list of playlists
getsong.ack	✓	name of the actual played song (without path)
getsongpath.ack	✓	path of the actual played song including the name
getstate.ack	✓	outputs the state of the device
getviewplaylist.ack	✓	content of the actual viewed playlist (selected with the command P)
loudness.ack	✓	current value of the loudness
nosupport.ack		answer for unknown and/or unsupported commands
pitch.ack	✓	current value of the pitch
random.ack	✓	current state of shuffle
repeat.ack	✓	current state of repeat
treble.ack	✓	current value of the treble
volume.ack	✓	current value of the volume

### Configuration file

config.bin		factory default settings. The file is binary and an exact mirror for the EEPROM. See 1.13 Setup for the organization. Edit this file with a hex editor if you need your own factory default settings.
------------	--	---

### Pictures

4to0.gif		needed for apply the configuration for waiting for the reboot of the device
active.gif		showed as back button in the songlist frame if a playlist is showed
activei.gif		showed as refresh button in the info frame
barix.gif		used in uimenu.html
exstreamer.gif		picture for the control interface, used in skin.html

menu.gif		picture for the menu buttons in the configuration, used in uicfg.html
Settings.gif		picture for the setting interface of the device, used in settings.html
view.gif		the little picture on the left side of each playlist name in the playlist frame to view the playlist without playing it.

HTML pages (see 1.20 Configuration via HTML Pages for the dynamic marks included in the files)

index.html		main page of the web server, included the five frames: skin, info, playlist, songlist, empty. empty is a hidden frame that receives the answer of the CGI commands.	
info.html	✓	page for the info frame of index.html	
notauthorized.html		showed if the user isn't authorized to view a page	
playlist.html	✓	page for the playlist frame of index.html	
settings.html		for the setting interface of the device, used in index.html	
skin.html		for the control interface of the device, used in index.html	
songlist.html	✓	for the songlist frame of index.html	
status	✓	shows the actual states of the device	
uiaudio.html	✓	configuration pages for the corresponding settings	
uicontrol.html	✓		
uinetwork.html	✓		
uisecurity.html	✓		
uipplaylist.html	✓		
uiserial.html	✓		
uistreaming.html	✓		
uiwlan.html	✓		not for Exstreamer Digital
uicfg.html	✓		shows the current loaded versions of the device
uiconfig.html	✓	main configuration, contains the main frames for the configuration	
uidefaults.html	✓	set factory defaults	



uiraudio.html		showed after pressing apply or a reboot of the device is needed until the device has rebooted
uircontrol.html		
uirdefaults.html		
uirloader.html		
uirnetwork.html	✓	
uirnetwork0.html	✓	
uirnetwork00.html		
uirsecurity.html		
uirplaylist.html		
uirreboot.html		
uirserial.html		
uirstreaming.html		
uirupdate.html	✓	
uirwlan.html		not for Exstreamer Digital
uirdefaults I.html		showed after the device is set to factory defaults an has successfully rebooted
uireboot.html		reboot the device
uirreboot I.html		showed after the device is rebooted an has then successfully rebooted
uiupdate.html	✓	update the device
update.html	✓	forwarding page to hide the command for the update

### Remote Control

remote.ini		contains the known IR commands and the corresponding commands therefore
------------	--	---

### Java Script

util.js		javascript functions for the HTML configuration pages (range checks)
---------	--	--

The filenames mustn't start with rc.cgi or setup.cgi.

Don't exceed 64 kByte of data per file. Note that a bigger .cob file needs per 64 kByte one flash page of 64 kByte.

To generate a .cob file start the batch exstreamerdigapp.bat. Upload the generated .cob file into the device to the web application page (overwrite).

For the upload go to the configuration page of the device and click on the button Update. Follow the instructions there. If the device has rebooted and the update page is showed type <http://x.x.x.x/updateex.html> in the address field of the browser where x.x.x.x is the IP address of the device.

Free targets can be found in 1.17 Memory Page Usage. The target field is case sensitive. If you upload a .cob file to used pages the current content will be overloaded by the new one.

Another way to upload is with TFTP. Use the name of the page as target.

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. Sometimes it's needed to close the browser to see the changes depending on the browser's cache strategy.

## 1.15 Gong Application (for door bell)

To use the gong functionality upload a .cob file built with the gong application development kit into the device as described in 1.14 Own skins and web interface. Download this kit from the download section on [www.exstreamer.com](http://www.exstreamer.com) and replace gong.MP3 (all lowercase) with your own MP3 file. The Exstreamer looks for this file after each reboot. If this file is available and the CTS input is activated, this file is played as long as CTS is active. The whole file will be played all the way to the end if CTS is deactivated and the Exstreamer will then go back into its operating stage.

## 1.16 Idle Application (stream during idle)

To use the idle functionality upload a .cob file built with the idle application development kit into the device as described in 1.14 Own skins and web interface. Download this kit from the download section on [www.exstreamer.com](http://www.exstreamer.com) and replace idle.MP3 (all lowercase) with your own MP3 file. The Exstreamer checks this file after each reboot. If this file is available and the Exstreamer is in idle mode (not pause) this file is played continuously. If the Exstreamer starts to play another stream the idle song stops playing.

## 1.17 Memory Page Usage

A page is 64 kByte of flash memory. Free pages can be used for additional resources.

**Exstreamer (1MB Flash)** (Note: 0xC00000 = 0xD00000 = 0xE00000 = 0xF00000)

Page / Target	Content	Address for Rescuekit
8K (WEB0)	exstreamdigware.ROM (Firmware)	0xC00000
WEB1	reserved	0xC10000
WEB2	XT05.BIN (BIOS)	0xC20000
WEB3	SG.BIN (Util library)	0xC30000
WEB4	sonicip.cob (Sonic IP Resources)	0xC40000
WEB5	exstreamerapp.cob (Web Application)	0xC50000
WEB6	exstreamerapp.cob continued (Web Application)	
WEB7	reserved	0xC70000
WEB8	temporary used for updates	0xC80000
...	free	
WEB11	free (see 1.14 Own skins and web interface)	0xFB0000
WEB12	free (recommended for Idle Application)	0xFC0000
WEB13	free (recommended for Gong Application)	0xFD0000
WEB14	free	0xFE0000

**Exstreamer Wireless or Exstreamer Digital (2MB Flash)** (Note: 0xC00000 = 0xE00000, 0xD00000 = 0xF00000)

Page / Target	Content	Address for Rescuekit
8K (WEB0)	exstreamdigware.ROM (Firmware)	0xC00000
WEB1	reserved	0xC10000
WEB2	XT05.BIN (BIOS)	0xC20000
WEB3	SG.BIN (Util library)	0xC30000
WEB4	sonicip.cob (Sonic IP Resources)	0xC40000
WEB5	exstreamerapp.cob or exstreamerdigapp.cob (Web Application)	0xC50000
WEB6	exstreamerapp.cob or exstreamerdigapp.cob continued (Web Application)	
WEB7	reserved	0xC70000
WEB8	temporary used for updates	0xC80000
...	free	
WEB28	free (see I.14 Own skins and web interface)	0xFB0000
WEB29	free (recommended for Idle Application)	0xFC0000
WEB30	free (recommended for Gong Application)	0xFD0000
WEB31	temporary used for updates	0xFE0000

### 1.18 Dynamic Web Page

Dynamic Web Pages are built in HTML or XML or each other text file format that doesn't contain the binary character 0x00. A dynamic page can be an HTML file. To indicate such a page it has to contain the initial mark `&L(0,"*")` in the first 500bytes before the first dynamic value is used. It's possible to use scripts or everything else allowed in the document's file format. The optional third parameter (decimal value) in the initial mark is for file options. If the bit 0 (`&L(0,"*",1)`;) in the third parameter is set the content length will not be added in the HTTP header. That's faster because the file will then only be parsed once. Normally the file is parsed once for the content length defined in the HTTP header. Since the length could change during execution a second time parsing is needed in order to send the page to the browser (ex. current uptime). If the content length is need (for special web servers) the format string can be used in order to define a fix length. The third parameter is used as a bitset. The bits 1 - 3 are used as password level (1-6) for the file corresponding to the password level parameters in the configuration (see 1.13 Setup). Example for level 5: (`&L(0,"*",10)`;) . If the password level is 7 then the password for the Telnet/Serial setup is used. If the bit 4 is set the access will be exclusive (only one user at a time, tested by its IP address). The user has to logout (see 1.21 Configuration Logout) or the software does an automatic logoff 20 min after the last access to such a page. Only one password level can have the exclusive feature (doesn't matter which one).

### 1.19 The Dynamic Mark

Definition: `&L<name>(<id>,<format>[,par]);`

A dynamic mark always starts with `&L`. The mark is case sensitive. Attached is a name which selects a group of dynamic values. Defined is the group 'State' for values used in the firmware during runtime. The group 'Setup' is defined for all configuration parameters. In the parentheses are the formal parameters like in a C function call. The mark ends always with a `“;”`. `<id>` defines the desired value. `<format>` is a C-style format string (refer to the ANSI documentation). `<par>` are additional parameters. If an additional parameter is needed it is mentioned in the related chapter in this document. No `'` (closing parenthesis) and `;` (semicolons) are allowed in the dynamic mark.

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

Example: `<td>&LSetup(1,"<b>%u</b>",0);` highest byte of the own IP address`</td>`  
is going to the following line if the current value of the highest byte of the own IP address will be 192:  
`<td><b>192</b>` highest byte of the own IP address`</td>`

## 1.20 Configuration via HTML Pages

The HTML pages for the device configuration use the functionality for dynamic web pages (see 1.18 Dynamic Web Page). All of the configuration parameters are placed in HTML forms and are transferred by the method GET. Some of the values are checked by java script to prevent wrong values. Not all of the configuration parameters have to be present in a 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="answer"><input type="hidden" type="text" name=L value=uirnetwork.html>
```

The target of the form could be changed.

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

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 (see more about this below). The type character B stands for an unsigned value (see table below). 0 is the address of the expected configuration parameter (see table in 1.13 Setup).

The value is a dynamic mark (see table below).

```
<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 'Exstreamer Name' in the setup (see table in 1.13 Setup).

```
<input name=S256 size=20 maxlength=20 value='&LSetup(4,"%s",256);'>
```

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 common setup (see table in 1.13 Setup). 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.

```
<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 'Output Mode' 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=B250>
  <option value=0 &LSetup(3,"%s",250,B,0,"selected");>Stereo</option>
  <option value=1 &LSetup(3,"%s",250,B,1,"selected");>Mono</option>
  <option value=2 &LSetup(3,"%s",250,B,2,"selected");>Bridge</option>
</select>
```

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.

Dynamic Marks for Group State:

<id>	Type	Description
1	Function	Prints volume in percent
2	Function	Print string if equal 3. [par]: state variable 1 shuffle (0 = off, 1 = on) 2 repeat (0 = off, 1 = song, 2 = playlist) 3 MPEG version (0 = MPEG 2.5, 2 = MPEG 2, 3 = MPEG 1) 4 MPEG Layer (0 = unknown 1 = Layer III, 2 = Layer II, 3 = Layer I)

		<ul style="list-style-type: none"> <li>5 volume lock (0 = no lock, 1 = locked)</li> <li>6 synchronisation state (0 = no synchronized, 1 = synchronized, 2 = start hook in, 3 = wait for data, 4 = wait for frame, 5 = wait for sync, 6 = master)</li> <li>7 more playlists (0 = not more playlists, 1 = no previous playlists, 2 = previous and next playlists, 3 = no next playlists, 4 = playlists for letter)</li> <li>8 playlist count (0 for no playlists)</li> <li>9 first letter of viewed playlist (0 for no playlist, else ASCII code of the first letter)</li> <li>10 1 if actual playlist number is equal to the viewed playlist number</li> <li>11 first letter of played playlist (0 for no playlist, else ASCII code of the first letter)</li> <li>12 loudness on (0 = off, 1 = on)</li> <li>13 volume mute (0 = off, else on)</li> <li>14 protection bit of the MP3 header of the current played MP3 stream (0 = off, 1 = on)</li> <li>15 padding bit of the MP3 header of the current played MP3 stream (0 = off, 1 = on)</li> <li>16 private bit of the MP3 header of the current played MP3 stream (0 = off, 1 = on)</li> <li>17 copyright bit of the MP3 header of the current played MP3 stream (0 = off, 1 = on)</li> <li>18 original bit of the MP3 header of the current played MP3 stream (0 = off, 1 = on)</li> <li>19 channel mode of the MP3 header of the current played MP3 stream (0 = stereo, 1 = joint stereo, 2 = dual channel, 3 = single channel)</li> <li>20 mode extension of the MP3 header of the current played MP3 stream</li> <li>21 emphasis of the MP3 header of the current played MP3 stream (0 = none, 1 = 50/15 ms, 2 = reserved)</li> <li>22 mode (0 = unknown, 1 = streaming, 2 = pull)</li> <li>23 status (0 = idle, 1 = play, 2 = pause)</li> <li>24 license (used as bitset (bit = 0 no license, 1 = license active): bit 0: synchronisation license)</li> </ul> <p>4. [par]: Type (see id 1 in 'Dynamic Marks for Group Setup' below)</p> <p>5. [par]: value to compare</p> <p>6. [par]: string for output if state value is equal to 5. [par]</p>
3	Fifo Function	<p>Outputs the content of a playlist. The format string is for only one song/line. The contained variables and the sequence in the format string are dependent of the source.</p> <p>3. [par]: source</p> <p style="padding-left: 20px;">1 for web: variables in the following sequence: song number (unsigned integer %u), song number (unsigned integer %u), song name (string %s)</p> <p style="padding-left: 20px;">else: only variable: song name (string %s)</p>

		<p>4. [par]: 0 for actual played playlist, else for viewed playlist</p> <p>5. [par]: unsigned decimal number of the desired playlist</p>
4	Fifo Function	<p>Outputs the file name of a path</p> <p>3. [par]: state variable</p> <ul style="list-style-type: none"> <li>1 song name</li> <li>2 viewed playlist name</li> </ul>
5	Function	<p>Outputs the bitrate of the current playing stream as an unsigned decimal number. If the bitrate is detected as bad it outputs the text 'bad'.</p>
6	Word Value	<p>sampling rate of the current playing stream</p>
7	Function	<p>Outputs the content of the current playlist (current played or the one selected for viewing). The contained variables and the sequence in the format string are in the following sequence: song number (unsigned integer %u), song number (unsigned integer %u), song name (string %s)</p> <p>3.[par]: fix set to 1 for song number</p> <p>4.[par]: fix set to 1 for song number</p> <p>5.[par]: fix set to 2 for song name</p>
8	Function	<p>Print Byte</p> <p>3. [par]: state variable</p> <ul style="list-style-type: none"> <li>1 current IP address</li> <li>2 LAN MAC address</li> <li>3 current netmask (stored as 4 bytes like the current IP address)</li> <li>4 current gateway IP address</li> <li>5 current DNS IP address</li> <li>6 current MAC address</li> <li>11 current server IP address</li> </ul> <p>4. [par]: offset in bytes for the state variable</p> <p>ex. &amp;LState(3,"%u",1,0); for the highest byte of the current IP address</p> <p>ex. &amp;LState(3,"%H",2,0); for the MAC address</p>
9	Fifo Function	<p>Outputs the list of playlists.</p> <p>The format string is for only one playlist/line. The contained variables and the sequence in the format string are: playlist number (unsigned integer %u), playlist number (unsigned integer %u), playlist name (string %s)</p>
10	Function	<p>Outputs the discovered Barix devices in the network.</p> <p>The format string is for only one device/line. The contained variables and the sequence in the format string are:</p>

		IP address of the discovered device (%A), name of the discovered device (string %s), IP address of the discovered device (%A), synchronisation port of the discovered device (unsigned integer %u, 0 for no synchronisation possible), name of the discovered device (string %s)
11	Function	<p>Print state value</p> <p>3. [par]: state variable</p> <ul style="list-style-type: none"> <li>1 number of the next playlist (for reload playlist buffer)</li> <li>2 number of the previous first playlist (for reload playlist buffer)</li> <li>3 number of the current viewed playlist</li> <li>4 number of the current played playlist</li> <li>5 version string of the server (zServer)</li> <li>6 hardware type (0 = unknown, 1 = Exstreamer, 2 = Exstreamer Wireless)</li> <li>7 codec type (0 = no codec, 1 = MAS3509 (decoder), 2 = MAS3587 (encoder))</li> <li>8 mode (0 = unknown, 1 = streaming, 2 = pull)</li> <li>9 status (0 = idle, 1 = play, 2 = pause)</li> <li>10 synchronisation state (0 = no synchronized, 1 = synchronized, 2 = start hook in, 3 = wait for data, 4 = wait for frame, 5 = wait for sync, 6 = master)</li> <li>11 count of playlists</li> <li>12 current playing playlist name</li> <li>13 count of entries in the current playlist</li> <li>14 current playing song name</li> <li>15 number of the current song</li> <li>16 TCP state admin (playlist) connection (0 closed, 1 listen, 2 established, 3 syn sent, 4 syn received, 5, fin wait 2, 6 fin wait 2, 7 fin wait 3, 8 closing, 9 last ack, 10 time wait, 11 drain)</li> <li>17 TCP state stream connection (0 closed, 1 listen, 2 established, 3 syn sent, 4 syn received, 5, fin wait 2, 6 fin wait 2, 7 fin wait 3, 8 closing, 9 last ack, 10 time wait, 11 drain)</li> <li>18 amount of bytes in the streaming buffer</li> <li>19 number of played bytes (double word)</li> <li>20 last received IR command (string)</li> <li>21 current volume</li> <li>22 current mute volume (if mute on equal to last current value, else 0)</li> <li>23 current balance</li> <li>24 current bass</li> </ul>

		<p>25 current treble                  26 current loudness                  27 shuffle (0 = off, 1 = on)                  28 repeat (0 = off, 1 = song, 2 = playlist)                  29 feedback (used as bitset, on if set to 1 (bit 0 = TCP stream byte, bit 1 = TCP stream sec, bit 2 = UDP byte, bit 3 = UDP sec, bit 4 = serial byte, bit 5 = serial sec bit 6 = TCP cmd byte, bit 7 = TCP cmd sec))                  30 seconds played of the current stream (double word)                  31 CTS (0 = off, 1 = on)                  32 CTS flag (see command RESETLASTCTS)                  33 PIO (0 = off, 1 = on)                  34 PIO flag (see command RESETLASTPIO)                  35 sync port (0 if slave, else own master sync port)                  36 current viewed playlist name                  37 mode extension of the MP3 header of the current played MP3 stream                  38 CRC of the MP3 header of the current played MP3 stream                  39 current uptime in milliseconds (double word)                  40 current uptime in seconds (double word)                  41 count of entries in the viewed playlist</p>
12	Fifo Function	<p>Outputs the list of playlists or one specific playlist.                  The format string is for only one playlist/line. The contained variable in the format string is: playlist name (string %s)                  3. [par]: 0 for all playlists else the decimal number of the desired playlist</p>

Dynamic Marks for Group Setup:

<id>	Type	Description
1	Function	<p>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 ex. b3 for the fourth bit). If this parameter isn't available the type will be B.                  ex. &amp;LSetup(1,"%08lx",315,D); content stamp as hexadecimal value with 8 characters an leading zeros                  ex. &amp;LSetup(1,"%lu",311,D); Phone home interval as unsigned long decimal value</p>
2	Function	<p>Print Netmask Byte                  3. [par]:Address (decimal) of the value in the setup</p>

		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 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

See chapter 1.13 Setup for the addresses of the configuration parameters.

Names for the form elements:

If the value is an integer the first character is a B.

If the value is a Netmask the first character is an N.

If the value is a string the first character is an S.

If the value is a long (4 byte) the first character is a D.

If the value is a signed byte the character is a c.

If the value is a word the character is a W.

The following decimal value in the name is the address of the configuration parameter (see 1.13 Setup).

To set a bit in a configuration parameter (ex. Mediaconfig) add the character b followed by the number of the bit (ex. 7 for the 8. bit in the byte) starting at 0.

To define the byte of the Netmask add the character B followed by the byte number (see <id> 2 in the table above).

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
S255	Exstreamer Name
B277b7	Sonic IP

## 1.21 Configuration Logout

The logout is placed in an HTML form and is transferred by the method GET. The form has to contain an element named L with the value for the answer page and a second element with the name D. This element is the indication for the logout.

```
<form action=setup.cgi method=get target=_top>  
  <input type=hidden name=L value=logout.html><input type=hidden name=D><input type=submit value=" Logout ">  
</form>
```

The target of the form could be changed.

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

## 2 zServer

### 2.1 zServer Discovery

After startup the Exstreamer sends every 5 second a UDP broadcast packet to the zServer's listening port 12321 until it receives the zServer's answer. If the zServer receives this packet it answers with its UDP packet to the Exstreamer's IP address and the Exstreamer's listening port 12320.

The Exstreamer's Broadcast UDP packet is an ASCII string 'Z11' (Z, one, one). The three bytes are (starting with byte number 0): 0x54 0x31 0x31.

The zServer's UDP answer:

Starting in byte number 0 is a null-terminated version string of the zServer. Directly appended to this string ,data structures in the following frame are concatenated without any gap: <type><length><info>

<type> is a byte and describes the info, <length> is a word (two bytes, low byte first) and includes the length of <info> in bytes and <info> contains the info data. All the values are starting with the low byte first (Intel byte order). The last frame is always an end of data.

<type>	<length>	<info>
0x00	0x0001	0x00 (end of data)
0x01	0x0008	This is the number of milliseconds since January 1, 1970, 00:00:00 GMT. of the latest changing of any playlist.
0x02	0x0004	This is the number of seconds since midnight local time.
0x03	0x0004	Byte 0: day of month (starting with 1), Byte 1:month (starting with 1), Byte 2/3: year
0x04	0x0010	This is a UUID of the running zServer. Note: only one zServer process per computer.
0x06	0x0002	HTTP serving port of the zServer (normally 8888)



An example zServer packet (length 60 bytes, all bytes in hexadecimal):

5a 31 2e 34 39 36 00 01 08 00 c7 73 7e a3 f2 00
00 00 02 04 00 e7 3a 00 00 03 04 00 09 01 d3 07
04 10 00 01 20 ea 01 00 00 00 00 00 00 00 00 00
00 00 00 06 02 00 b8 22 00 01 00 00

Explanation:

Table with 4 columns: data, type, length, info. It explains the fields of the zServer packet, such as version-string, timestamps, and UUID.



## 2.2 zServer.ini file

The zServer.ini file holds up the configuration of the zServer. Important for the user is the section services. Restart the zServer if you change something in this file.

```
#####  
# section services:  
# services are internally managed available to the launched classes  
# supported services are:  
# playlist  
# scans the given directory for playlists (*.m3u) adds this filenames avoid duplicate file names  
# adding a number +. previous to the name  
# shoutcast  
# creates a buffered stream service on demand. does one download from the internet, clones  
# internally saving bandwidth. talks shoutcast protocol for MP3 files. gives recording capabilities.  
  
services{  
    # playlist;root  
    # [r] playlist: indicates to add this directory to playlistdirectories  
    # [r] root: the root directory to scan for playlist files  
    playlist;c:/CD's/Playlists  
  
    # shoutcast;snam:url  
    # [r] snam: the name of the stream to appear in the playlist, you have to care for it being unique  
    # [r] url: the url to stream from  
    shoutcast;groovesalad;http://www.somafm.com/groovesalad.pls  
    shoutcast;jazzmasterzz;http://www.somafm.com/special.pls  
    shoutcast;electro;http://www.somafm.com/electro.pls  
    shoutcast;secretagent;http://www.somafm.com/secretagent.pls  
    shoutcast;dronezone;http://www.somafm.com/dronezone.pls  
    shoutcast;squiddown;http://www.somafm.com/squiddowntemp.pls  
    shoutcast;indiepop;http://www.somafm.com/indiepop.pls  
}
```

### 2.3 Playlist Directory

The entry of the playlist directory is done by the configuration tool of the zServer configz.bat. This tool registers the playlist directory in the first part of the service section. The format of this playlist directory is shown above.

If you have the playlists in more than one directory add a new line (playlists;...) for each directory in the zServer.ini under the section services. The zServer dynamically merges and sorts all playlists together. One exception are the internet radio stations which are always added at the end of the list.

If playlists are removed or added in this directories the zServer detects this and updates his list. Note: the Exstreamer takes over the changed playlists after one minute.

### 2.4 Internet Radio Stations

The zServer is also able to serve internet radio stations to the Exstreamer. This feature is undocumented yet, but can be used if you register the radio stations in the zServer.ini file. The pls playlist format as used from [www.shoutcast.com](http://www.shoutcast.com) or the direct URL to the radio station is supported. The playlist can be entered in the format as shown in the example above. You also have to give the radio station a name. This name appears in the playlist frame of the Exstreamer.

### 2.5 Playlist Generator

With the Exstreamer setup a playlist generator is included. With this playlist generator it's possible to generate playlists from one or more directories and subdirectories. Unpack the ExstreamerSetup.jar file which was installed in the setup path during execution of the setup with ex. WinZip into an own folder. To start this command line tool for help, type in the folder where you have extracted this utility: `java Dir2M3U`

The generator in the setup uses the call: `java Dir2M3U -sI -ps -bn0 <MP3 Path> <M3U Path>`

### 3 Specification Playlists

The following remarks describe how the Exstreamer handles lists of playlists, playlists and songs. The Exstreamer basically communicates with a web server. This could either be the Barix zServer (our own local java web server) or a standard web server. The Exstreamer finds its zServer automatically on the network. By using a standard web server, the server PC has to have a fix IP address, which has to be configured in the Exstreamer's system setup.

To support the Exstreamer's playlist functionality, the web server has to serve some certain system files, that give information about the servers playlists.

#### 3.1 List of Playlists

A list of playlists is a file including all names (without a path) of available playlists on the system with or without an extension ("m3u" is used if not defined). See section 3.2 for supported types. In this file, names are separated with one or more carriage return (cr ASCII 0dh), line feed (lf ASCII 0ah) or zero (ASCII 00h). Normally the separator is one ASCII 00h. There mustn't be any HTML tags inside.

The Exstreamer shows exactly these names in the playlist frame (without the extension).

The Exstreamer gets the playlist from the server by doing a file request. If the system is using a zServer i.e. on IP 192.168.1.71, the request would point to <http://192.168.2.71:8888/playlist.html>.

If the system is using a normal web server i.e. on IP 192.168.2.71, the Exstreamer gets the list of playlist file (playlist.html) on the standard HTTP port 80 (instead of port 8888 on zServer).

The path to the playlists, the path to playlist.html and other system files can be configured in the Exstreamer's setup. Also the IP port for the request of these files can be configured.

### 3.2 Playlist

The Exstreamer basically supports playlists with the extension "m3u", "asx", "pls" (WinAMP, Musicmatch, Windows Media Player, etc.).

To get the contents of a playlist i.e. "All Saints" from the zServer the Exstreamer adds ".m3u" to the name (if there is no extension stored). It then requests the file as follows: <http://192.168.2.71:8888/All%20Saints.m3u>.

If the system is using a common web server, the Exstreamer would request the playlist file on a standard HTTP port 80 (instead 8888). The playlist path in the server setup of the Exstreamer defines the path where this m3u playlists are relative to the web servers root (/playlistdirectory/). Don't forget the first slash (/) and the slash at the end. They are always needed.

If the zServer is used the path to the songs have to be absolute from the root.

The Exstreamer shows only the name of the file in the song list frame (no paths and no extensions). If the playlist is an "asx" or "pls" then the related title will be showed.

### 3.3 Song

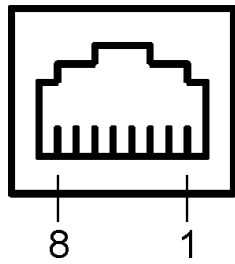
To get a song file the Exstreamer uses the complete path out of the playlist i.e. "C:\MP3\All Saints\All Saints - Never ever.mp3". The file is requested from zServer's port 8888 with the argument: [C:/MP3/All Saints/All%20Saints%20-%20Never%20Ever.mp3](http://192.168.2.71:8888/C:/MP3/All%20Saints/All%20Saints%20-%20Never%20Ever.mp3). Backslashes are generally replaced by slashes.

If the path is an URL i.e. [http://192.168.2.71/All Saints/All Saints - Never ever.mp3](http://192.168.2.71/All%20Saints/All%20Saints%20-%20Never%20Ever.mp3), the Exstreamer checks for a zServer on the requested IP address. If the zServer exists, the Exstreamer requests the file on port 8888. If there is no zServer (system uses standard web server), the Exstreamer requests the file on standard HTTP port 80 resp. on the configured port in the setup. If there is a port in the URL this port is used to connect to the file.

The MP3 bitstream is expected on a random TCP port between 30000 and 39999, which is generated by the Exstreamer and transmitted in the TCP layer. If the system uses a common web server the standard HTTP port resp. the configured port in the Exstreamer setup is used for the request. In this case the path until and with ':' will be removed from the path.

## 4 Hardware and Connectors

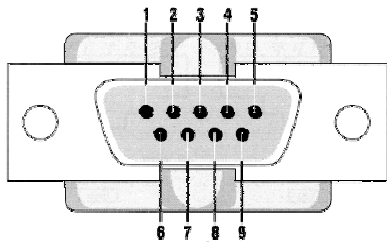
### 4.1 Ethernet



1	TX+		5	NC
2	TX-		6	RX-
3	RX+		7	NC
4	NC		8	NC

Yellow LED (right LED): Link ok if on  
 Green/Orange LED (left LED): green if 10Mbit, orange if 100Mbit

### 4.2 Serial Port



1	NC		6	NC
2	RxD		7	RTS
3	TxD		8	CTS
4	9VDC		9	NC
5	GND			

Transceiver: SIPEX SP3232E

$V_{RxD}$  +/- 15V TxD is short circuit protected.  
 Max. Input Logic Threshold LOW 0.8V Min. Input Logic Threshold HIGH 2.0V  
 $i_{max}$  (RTS): 10mA  
 $I_{max}$  (Exstreamer: 9VDC with supplied 9V power supply, Exstreamer Wireless and Exstreamer Digital: 12VDC with supplied 12V power supply): 100mA. This pin is connected directly to the power supply.

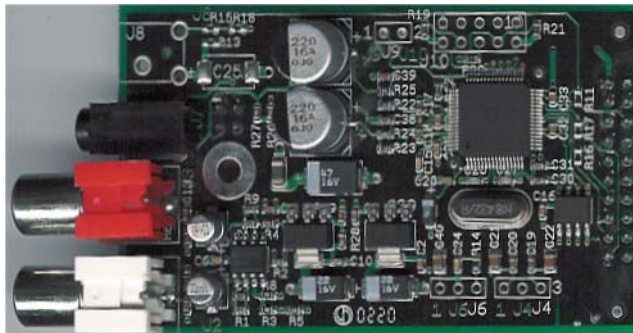
### 4.3 RTS State

If the parameter RTS usage is set to 'activate AVR' RTS is set to logic 1 (-12V) during streaming. One minute after the last stream is stopped (not paused) it will be reset to logic 0 (+12V).

This pin can be used to switch the source and/or power up your AVR for the Exstreamer. Be careful with this output and respect the RS232-C specifications.

## 4.4 PIO3 (not for Exstreamer Digital)

PIO3 is the pin 15 on the audio module connector. The GND pin can be found on the pin 2 or 20. Use this pin at your own risk because it's connected direct to a microprocessor port. The specification is a 5V-tolerant Input with  $V_{min} = 0V$ ,  $V_{max} = 5.6V$ ,  $I_{in,max} = 10\mu A$ . The pin is driven by a weak pullup.

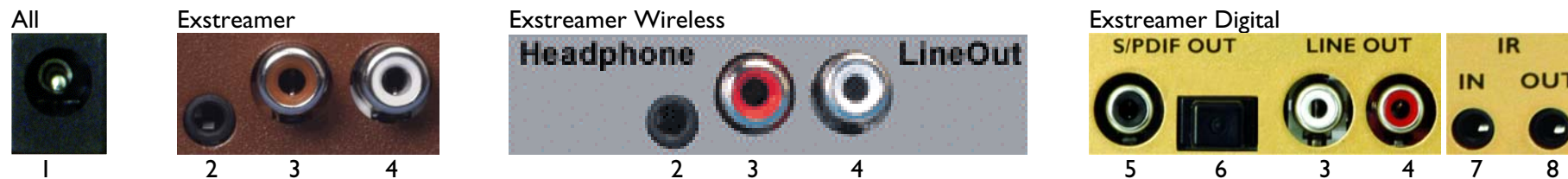


Pin Left	Pin Right
1	2
3	4
:	:
17	18
19	20

## 4.5 Codec

Built in is a Micronas MAS3509, MAS3519 or MAS3539 with the revision A2 (no S/PDIF support) or B4. For further information see Micronas' datasheets of the codecs.

## 4.6 Other Connectors



Name	Nr	Description
Power	1	Power in, Center +9..24 VDC, Ring Ground
Headphone	2	Headphone output, 3.5 mm Jack, 2Vpp max.
LINE OUT left	3	Line output left RCA, 4.2Vpp max. level (0 dbFs), SNR>85dbFs (Exstreamer Digital: SNR>90dbFs)
LINE OUT right	4	Line output right RCA, 4.2Vpp max. level (0 dbFs), SNR>85dbFs (Exstreamer Digital: SNR>90dbFs)
S/PDIF OUT coax.	5	coaxial S/PDIF output, 32 kHz, 44.1 kHz, 48 kHz
S/PDIF OUT optic.	6	optical S/PDIF output, 32 kHz, 44.1 kHz, 48 kHz
IR IN	7	3.5 mm Jack Infared input for IR receiver
IR OUT	8	3.5 mm Jack Infared output for IR transmitter