BARIX
THE VOICE OF SIMPLICITY

TECHNICAL DOCUMENTATION

ABCL

BASIC like programming language firmware for Barix Audio products

Firmware V1.21
Released 19th October 2017

Supports:

- EXSTREAMER 100, 105, 110, 120, 200, 205, 500, 1000
- Instreamer 100, Instreamer new
- ANNUNCICOM series
- IP Audio Module 100, 101, 102, 300, 301, 302, OEM362

Table of Contents

1 Introduction

This Technical Documentation describes the Non Volatile Databases, External Interfaces, Development Kit and Autonomous Operation of the Instreamer .

1.1 About the ABCL Software

The ABCL Software is a user programmable audio platform supporting a variety of functions and interfaces like filesystem, network access, serial interface, audio input/output or direct IO control. The platform is programmable in the Barix Control Language (BCL, see the language reference Error: Reference source not found) and runs on the standard Barix audio hardware (Instreamer, Exstreamer, Annuncicom series and the Barix IPAM).

The goal of the ABCL is to allow system integrators, distributors and users to develop custom specific applications running on Barix hardware.

1.2 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 BCL Development Kit (describing the demo programs and explaining how to use the provided tools to compile custom applications)
- 3 Application Remote Control Interface (explaining how to control the device using the command suite)
- 4 WEB User interface (explaining the User Interface functionality and how to customize it)
- 5 Memory Organization (explaining the use of the Flash memory and the EEPROM configuration memory)
- 6 Hardware Controls (explaining the Reset Button functionality and the status information provided by the LEDs)
- 7 Firmware and standard File Upload (explaining the standard Barix method for updating the units Software)

2 BCL Development Kit

2.1 Requirements

The ABCL rescuekit is distributed with necessary tools and scripts for compilation running on both Linux and Windows platforms.

However on Linux operating systems other external programs are required:

- the DOS emulator “dosemu” (see http://www.dosemu.org/)
- the Windows emulator “wine” (see http://www.winehq.org/)
- the Advanced TFTP client “atftp” (see http://freshmeat.net/projects/atftp/)

All the mentioned programs are expected to be in the system path. If they are not, modify the scripts accordingly.

2.2 ABCL Architecture

The ABCL platform consists of the BCL interpreter (firmware and extensions), the WEB UI and the BCL programs with sources.
Before loading into the device the BCL sources are converted into a binary form (tokenized) and packed (usually with sources) into a **`.cob`** file.
See Error: Reference source not found for more details about tokenization. Programs are identified by the filename in the **`.cob`** file.

More programs can be loaded into the device, but only one can be interpreted at a time.
The 15-character name of the BCL program to be interpreted is stored in the Setup memory at position **`S180`** (see section 5.3 for more details).
The application to be executed can be selected in the “Application” field in the “Settings” WEB page of the device.

2.3 Source files

Sample applications

The ABCL platform is delivered with set of a sample BCL programs (applications) including their sources (more applications may be added). The content of the set may vary between different versions of the rescuekit.

The compiled applications are stored in **`applications.cob`** file located in the “update_rescue” directory and loaded into the FLASH memory of the device starting from page **`WEB10`**, see section 5.2 for more details.

To recreate **`applications.cob`** call the **`applctns.bat`** script in the “bcldevkit” subfolder of the rescuekit; on Linux call **`applications.sh`**

Sample list of applications (version 1.14 of the WEB application) :

annunfdx	Full-duplex Annuncicom
custom1	Custom Application 1
custom2	Custom Application 2

Sources of the sample programs are available in the “bcldevkit” directory in subdirectories of the respective name.

Custom applications

There're two placeholders for user-modifiable applications called **`custom1`** and **`custom2`**. Currently only **`custom1`** is present and depending on the rescue-kit is either loaded with a dummy program or an OEM application. The custom application (**`custom1.cob`**) is loaded into the FLASH memory of the device starting from page **`WEB6`** (see the memory organization in chapter 5.2).

The sample programs can be modified by the user and using the provided tools compiled and loaded into the device (see 2.4). If a program with the same name already exists among the sample programs, the user program (which is at lower address in memory) takes precedence. After deleting or overwriting the user area with a program of a different name the original sample application is available again.

File Naming Conventions

Since the files in the FLASH memory of the device are not structured into directories (even though they might be in different **`.cob`** files), there are standard file naming conventions to prevent mixing of files between different applications. The “<basename>” in the following text refers to the application name as listed in the above table. The basic rule (with the exception of “readme” files) is that names of all files belonging to the same application start with the basename.

File	Name
program source	<basename>.bas
tokenized program	<basename>.tok
WEB UI frameset	<basename>.html
UI configuration	<basename>_config.html
UI help	<basename>_help.html
program description	README_<basename>

2.4 Compilation Tools

To compile and load a program into the FLASH memory of a device use the provided **`bcl.bat`** (**`bcl.sh`** on Linux) script in the “bcldevkit” directory. It calls the **`tokenizer.exe`**, creates a **`.cob`** file calling the **`web2cob.exe`** and optionally loads the **`.cob`** file into the device using the TFTP protocol. Syntax of the program is the following:

```
bcl.bat <basename> [ <address of the unit> ]
```

On Linux:

```
bcl.sh  <basename> [ <address of the unit> ]
```

The address is either the IP address or the DNS address. If it is omitted, only the **.cob** file is created.

Building a custom application

To build a custom program and load into the device, do the following:

1. Create a subdirectory of the “bcldevkit” directory with the basename of your program
2. Create the source **.bas** file, UI files and optionally other files belonging to the application using the naming convention mentioned above
3. Switch the device into the update mode (either over the WEB interface or using the reset button)
4. Run the bcl.bat script (bcl.sh on Linux) with the basename and address of the unit as described above
5. Reset the device

The program will be loaded into the FLASH memory starting from page **WEB6**. Please note that the above procedure will overwrite the original memory content! Therefore only ONE custom application can be loaded into memory at a time with this procedure.

Alternatively you can call **tokenizer.exe** and **web2cob.exe** manually to tokenize the source and create the **.cob** file, see Error: Reference source not found for further details.

If you're creating a new application don't forget to add a new entry to the list of applications in **uisettings.html** in the WEB UI as well(see 4.2).

3 Application Remote Control Interface

A command suite, based on AnnuncicomIC technology, is provided to manage the **Error! Unknown document property name.** via the WEB HTTP Interface.
For remote control via HTTP, the CGI WEB script rc.cgi is used. For example http://x.x.x.x/rc.cgi?c=99 (DEVICERESET- Hard reboot of device.)

3.1 Control Interface Description

1. 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).
- An answer can be selected by concatenating the L command. If no special answer is requested tempty page is returned.
- The answer files can be edited and changed to your needs (see 4**Error! Reference source not found.**).
- 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.

1. To concatenate control commands use **&** (Ampersand, ASCII:38).

- If password is set, it must be sent either with "a=" concatenated with the command, e.g. **`http://x.x.x.x/rc.cgi?c=99&a=password`** or within the HTTP request header ("WWW Authentication Basic").

3.2 List of commands

Element	Description	command
Reserved		c=0 … c=93, c=95 … c=98
DEFAULTS	Loads defaults into Setup rom : config.bin and appconfig.bin (if present) and stores in EEPROM	c=94
DEVICERESET	Hard reboot of device.	c=99
BOOTLOADER	Starts the bootloader. The application will be left. It isn't running until the next reboot.	c=100
PASSWORD	If password is set and not send in HTTP request header, a=must be used with command see 3.1	a=…
GETDYNFILE	The response is the dynamic file stored in a cob file (see 4.2) with that name. e.g.: L=index.html	L=…

3.3 Principles of the CGI WEB interface

- The browser should support frames.
- POST method should be used in configuration forms (setup.cgi).
- GET method should be used elsewhere.
- Respect the common character set for URLs.
- Example of CGI WEB commands: http://x.x.x.x/rc.cgi?c=99 (command for RESET on ABCL unit with IP address x.x.x.x)
- If "**L=**" is not used, blank page is sent as an answer.
- All strings and everything else are case sensitive.
- A CGI request should not exceed 1024 bytes.
- To concatenate control commands use **&** (Ampersand, ASCII:38).
- If a password is set, it must be sent either with "a=" concatenated with the command, e.g. **http://x.x.x.x/rc.cgi?c=99&a=password** or within the HTTP request header ("WWW Authentication Basic")

4 WEB User interface

4.1 The WEB Server

The Firmware runs two WEB server processes, which by default serve incoming HTTP requests on TCP port 80. The port number can be changed by setting the W196 parameter in Setup (see the WEB User interface parameter on page 9).

Mimetype.ini

In order the WEB server returns a proper MIME type, each file is processed by the **`web2cob`** tool and the proper HTTP header is prepended to the file content before it is stored in the COB file and in the device's FLASH. A database of known MIME types is held in the file **`mimetype.ini` in the package**. This text file contains a translation table from file extension to a MIME type. The MIME type database should be updated in case new file types are added to the WEB UI. If the file extension is not recognised, no extra HTTP header is added to the file content during file processing. Therefore also no MIME type description is returned to the browser and it is upon the browser to interpret the data correctly or to guess the file format.

The format of the MIME database is following:

- each file extension/MIME type pair is on a separate line
- lines are terminated by CR/LF (ASCII 0x0D 0x0A)
- the file content is case-sensitive
- a line starts with the file extension (without the leading dot and in the proper case), followed by a single space character (ASCII 0x20) and by the MIME type
- the line order is not significant

Note

Please note that during processing with **`web2cob`**, every file recognised in **`mimetype.ini`** will be stored in the COB file with an extra header. This needs to be possibly respected by the running BCL application to skip the header when reading the file. In order to store the file unmodified, remove the respective file extension from **`mimetype.ini.`**

Default content of the `mimetype.ini` file

```
txt text/plain
text text/plain
css text/css
xsl text/xml
xml text/xml
html text/html
htm text/html
shtml text/html
plg text/html
bmp image/bmp
```

Default content of the `mimetype.ini` file

```
dib image/bmp
gif image/gif
jpeg image/jpeg
jpg image/jpeg
jpe image/jpeg
jfif image/jpeg
pjpeg image/jpeg
pjp image/jpeg
tif image/tiff
tiff image/tiff
hta application/hta
js application/x-javascript
mocha application/x-javascript
class application/octet-stream
zip application/zip
rmi audio/mid
mid audio/mid
mp3 audio/mpeg
mp2 audio/x-mpeg
mpa audio/x-mpeg
abs audio/x-mpeg
mpega audio/x-mpeg
```

4.2 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 **`Development Kit`** 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`**, **`licence.cgi`**, **`basic.cgi`**, **`BAS.cgi`** or **`setup.cgi`**.

Web2cob tool

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

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 5.2) in upper case letters.
Select the cob file you want to upload and hit the “OK” button.
Click on the “Upload” button.

Rules:

1. If you upload a **`.cob`** file to pages already used, the current contents 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	System area factory default settings. The file is binary and it is an exact mirror of the system area settings for the EEPROM. See 5.3 for further details.
HTML	index.html	main page of the web server, included the five frames: info, menu, settings, empty. empty is a hidden frame that receives the answer of the CGI commands.
HTML	rebooting.html	displayed after the user settings are changed, redirects to the main page
HTML	status	shows the actual states of the device
HTML	uifapplication.html	frame page for BCL application configuration, links to the currently selected program in the setup (see 2.2) the actual configuration pages are stored in **`applications.cob`** file (see 2.3)
HTML	uifloader.html uifsettings.html uifreboot.html uifupdate.html	frameset for the corresponding pages
HTML	uifmenu.html	shows the Barix logo and the system version info
HTML	uihloader.html uihreboot.html uihsettings.html uihupdate.html	help for the corresponding pages

Type	Filename.extension	Description
HTML	uilogout.html	logout page
HTML	uimenuline.html	menu buttons for configuration, reboot and update
HTML	uireboot.html	reboot the device
HTML	uirloader.html	showed when the device is rebooting into the bootloader mode
HTML	uirreboot1.html	showed after the device is rebooted an has then successfully rebooted
HTML	uirreboot.html	Showed when the device is rebooting (“Reboot” button was pressed and confirmed)
HTML	uirupdate.html	forwarding page to hide the command for the update
HTML	uiupdate.html	update the device, appears after clicking “Update” in the menu
HTML	update.html	frameset for uirupdate.html
HTML	uisettings.html	main configuration page, contains the system settings
Image	4to0.gif	needed for apply the configuration for waiting for the reboot of the device
Image	barix.gif	used in uicfg.html
Image	menu.gif	pictures for the menu buttons in the configuration, used in uimenuline.html
Java Script	util.js	javascript functions for the HTML configuration pages (range checks)
text files	ABCLAPPVERSION	for the version number and the history
text files	ERRORS.HLP	table of textual error messages for the BCL interpreter
text files	SONICIPVERSION	for the version number of SonicIP implementation
Sound	baring.wav	Barix ringtone, used by BCL applications
Sound	n.mp3, dot.mp3	spoken “n” (where n:= 0-9) , spoken “dot”

4.3 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 4 is set the access will be exclusive (only one user at a time, tested by its IP address). The user has to 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).
- Bits 1-3 are used as password level (1-6) for the file corresponding to the password level parameters in the configuration.
  Example for level 5: (&L(0,"*",10);).
- 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, the “State” group for actual parameter states and the “BAS” group to interact with the BCL program (see Error: Reference source not found). 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, **`S`** for string, **`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. **`&LSetup(1,"%08lx",315,D);`** as hexadecimal value with 8 characters and leading zeros e.g. **`&LSetup(1,"%lu",311,D);`** 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 2. [par]: is always **`"%s"`** 3. [par]: Address (decimal) of the value in the setup 4. [par]: Type (see id 1 above) 5. [par]: value to compare. Can be integer or string. 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] Examples: **`&Lsetup(3,"%s",250,W,<10,"Parameter W250 is less than 10");`** **`&LSetup(3,"%s",419,S,"hello","Print this is hello");`**
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

ID	Type	Description
15	Byte (integer)	Filesystem Version Major
16	Byte (integer)	Filesystem Version Minor
17	Byte (integer)	sg.bin (Audio and Utility library) date string
18	Byte (integer)	reserved
19	Byte (integer)	reserved
20	Byte (integer)	Filesystem build date - year only decade
21	Byte (integer)	Filesystem build date - month
22	Byte (integer)	Filesystem build date - day

Dynamic Marks For Group &LState:

ID	Type	Description
1	Function	Print state variable 3. [par]: state variable index, see the table below eg. `&LState(1,"%s",12);` prints out device's MAC address
2	Function	Print string if condition is true 3. [par]: Index of the state variable to be compared, see the 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). When 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.

State Variables:

ID	Type	Description
0	Bool (Integer)	Filesystem present (1 means present)
1	Integer	Filesystem type: 0: unknown 1: FAT12 2: FAT16 4: VFAT

ID	Type	Description
		8: FAT32
2	Integer	Filesystem serial number
3-11		reserved
12	String	MAC address of the unit
13	String	Current IP address
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 mA
21	Bool (Integer)	USB device attached (1 means attached)
22	Integer	USB device capacity in kB
23-27		reserved
28	Integer	Hardware identification (hardware type)
29-34		reserved
35	Integer	IPAM identification (module type)
36-38		Reserved
39	Integer	System uptime in milliseconds
40	Integer	System uptime in seconds
43	Integer	Codec Type present in the device: • 0 = none • 1 = MAS3509 • 2 = MAS3587 • 4 = VS1003B • 5 = VS1023 • 6 = VS1033 • 7 = VS1053

ID	Type	Description
		• 8 = VS1063
44	Integer	Codec Info: 16-bit bitmask of DSP specific features. Bit set to 1 means feature supported. • bit 5 = DSP provides FIFO fill information, i.e. very low latency can be achieved with RTP buffering algorithm • bit 6 = DSP features 5-band equaliser • bit 7 = DSP features Acoustic Echo Cancellation
45	Integer	Supported Audio Modes: 32-bit bitmask of audio formats supported by the DSP. Bit set to 1 means format supported. • bit 0 = MP3 decoding • bit 1 =MP3 encoding • bit 2 = Windows Media decoding • bit 3 = uncompressed (PCM, G.711) full-duplex • bit 4 = uncompressed (PCM, G.711) decoding uni-directional • bit 5 = uncompressed (PCM, G.711) encoding uni-directional • bit 6 = AAC+ decoding • bit 7 = Ogg Vorbis decoding • bit 10 = MPEG layer 1 and 2 decoding • bit 11 = G.722 decoding only • bit 12 = G.722 encoding and duplex

4.4 Configuration via HTML Pages

The HTML pages for the device configuration use the functionality for dynamic web pages (see 4.3). All of the configuration parameters are placed in HTML forms and are transferred by the method POST. 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=POST action=setup.cgi target="answer"><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 'Flow control' 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=B82>
  <option value=0 &LSetup(3,"%s",82,B,0,"selected");>none</option>
  <option value=1 &LSetup(3,"%s",82,B,1,"selected");>Software (XON/XOFF)</option>
  <option value=2 &LSetup(3,"%s",82,B,2,"selected");>Hardware (RTS/CTS)</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=B198b7 value=0&LSetup(3,"%s",198,b7,0," checked");>Yes
<input type=radio name=B198b7 value=1&LSetup(3,"%s",198,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 stored in memory in hashed format (MD5) and set using the name **Px**, where **x** stands for the password level.

If the password is set already, the old password must also be supplied (with the name **Px**) together with the new password using the name **Px.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>
```

Px and **Px.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**.
- To set a password, use **P** as described above
-

The following decimal value in the name is the address of the configuration parameter (see chapter 5.3 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
- **B198b7** Sonic IP

4.5 WEB UI Configuration Logout

The logout is placed in an HTML form and is transferred by the method POST. 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=post 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.

5 Memory Organization

5.1 Serial Rescue Kit / Web Update

Two different procedures exist to upload the "**Error! Unknown document property name.**" 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.

5.2 Flash Memory usage

The "**Error! Unknown document property name.**" firmware is using the built-in Flash memory as described in the table below.

Flash memory usage table (Note: this may be subject to change if the applications.cob requires more than 3 pages)

Page / Target	File name	Content	Address for Rescuekit
8K (WEB0)	abclw.rom	Firmware	0xC00000
WEB1	fs.bin	USB Filesystem FW Extension	0xC10000
WEB2	sg.bin	Audio and Utility library - FW Extension	0xC20000
WEB3	sg.bin continued	Audio and Utility library - FW Extension	Continued (0xC30000)
WEB4	abclapp.cob	Web Application and Sonic IP Resources	0xC40000
WEB5	abclapp.cob continued	Web Application and Sonic IP Resources	Continued (0xC50000)
WEB6	custom1.cob	Custom Application Program	0xC60000
WEB7	Reserved	Reserved for continuation of custom1.cob and custom files	0xC70000
WEB8	Reserved	Reserved for continuation of custom1.cob and custom files	0xC80000
WEB9	bclio.bin	FW Extension – IO Driver	0xC90000
WEB10	applications.cob	BCL Application Programs	0xCA0000
WEB11	Reserved	Reserved for continuation of applications.cob	0xCB0000
WEB12	Reserved	Reserved for continuation of applications.cob	0xCC0000
WEB13	Reserved	Reserved for continuation of applications.cob	0xCD0000
WEB14	Reserved	Reserved for continuation of applications.cob	0xCE0000
WEB15 -WEB30	Unused	Free for user data	0xCF0000 - 0xDE0000
No symbol	Boot Loader	Boot Loader	0xDF0000

A page uses 64 kilobytes of flash memory. Flash memory of 2MB is assumed.

Please note: 0xC00000 is mirrored to 0xE00000 and 0xD00000 to 0xF00000.

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

The boot loader is always on the last page at absolute address 0xFF0000 (offset 0xFF00). The rest of the software is loaded at locations compatible with the symbolic locations: 8K, WEB1, WEB2 etc. 8K is always the first page (page 0), WEB1 page 1, etc. The target has to be in capital letters (i.e. WEB4).

5.3 Configuration storage (EEPROM)

The current configuration is stored in a non-volatile memory (EEPROM). To change the configuration use the web user interface and hit the “Apply” button to store it into the EEPROM. When the device starts (power on or reset) the stored configuration is copied from the EEPROM into a volatile data area known as “Setup”.

Setup Layout

Setup is logically split into two sequential areas, the system area (0-499) and the application area (500-1499). The first part contains the minimal set of parameters required by the BCL interpreter. The application area is used by BCL applications and does not have a fixed layout, nevertheless there is a recommended layout Error: Reference source not found for the demo applications delivered in the BCL rescue-kit.

There is no physical separation of these two areas and the BCL programs can access the whole setup range (0-1499).

Default settings

The system area defaults are stored in the **config.bin** file in the folder “webuidevkit/abclapp”, whereas the application area defaults are stored in the **appconfig.bin** file in the folder “bcldevkit”. The complete setup image is a concatenation of these two files plus the configuration for ethersound and is stored in binary file **config.bin** contained in the folder “update_rescue”.

Factory defaults using the Serial Rescue Kit

When applying the “Serial Rescue Kit” the EEPROM is overwritten with the “factory defaults configuration” held in the binary file **config.bin** (complete setup image) 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 the reset button

The “system defaults configuration” binary file **config.bin** (contained in abclapp.**cob)** and the “application defaults configuration” binary file **appconfig.bin** (contained in applications.**cob)** are loaded into the flash memory (not the EEPROM!) when performing the firmware update.

To apply the “factory defaults configuration” (both the system and the application factory defaults) the reset button has to be pushed for about 10 seconds, see chapter 6.1 for the usage of the reset button.

The file **appconfig.bin** as well as the **config.bin** can be edited with a hex editor. Consult Error: Reference source not found and the “configuration memory usage” table carefully before you make any changes.

Before uploading, folders containing the binary files have to be packed into the **abclapp.cob** and **applications.cob** files using the provided scripts.
For more details see chapters 4.2 and 2.3.

Configuration storage usage

The following table defines the configuration in the “Setup”. 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”.

General (Setup and 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.
- The Setup is stored in the EEPROM from offset 8
  (The first 8 bytes in the EEPROM holds the MAC address and check sum)

List of Configuration parameters

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	Subnetmask. The value is the count of the zero bits counted from the lowest byte. (ex. 8 for 255.255.255.0)
Lan Mode	9	B9b1-1, B9b2-1, B9b3-1 or B9	1	0	Ethernet Mode bit 0 - not used must be 0 bit 1 - auto-negotiate disable (set to 1 to disable) bit 2 - 100/10MBit (set to 1 for 10mbps) bit 3 - full/halfduplex (set to 1 for half duplex) Default 0 - auto-negotiate set
QoS	59	B59	1	0	Quality of Service/DSCP(differential Services Code Point) for UDP packets while streaming audio,

<table>
<tr><th>Parameter</th><th>Byte</th><th>Dynamic Name</th><th>Len</th><th>Default</th><th>Short Description</th></tr>
<tr><td>DNS 1</td><td>64</td><td>B64,B65, B66, B67</td><td>4</td><td>0.0.0.0</td><td>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.</td></tr>
<tr><td>DNS 2</td><td>68</td><td>B68, B69, B70, B71</td><td>4</td><td>0.0.0.0</td><td>Alternative DNS IP address. 0.0.0.0 here always disables secondary DNS</td></tr>
<tr><td>IFMODE0</td><td>80</td><td>B80b0-1, B80b2-3, B80b4-5, B80b6-7
or B80</td><td>1</td><td>0x4C</td><td>Mode parameters for serial port 0. Bit definitions:
<table>
<tr><td>Function</td><td>7</td><td>6</td><td>5</td><td>4</td><td>3</td><td>2</td><td>1</td><td>0</td></tr>
<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>
</table></td></tr>
<tr><td>BAUDRATE0</td><td>81</td><td>B81</td><td>1</td><td>2</td><td>Baudrate for the serial port 0:
0 = 38400 baud
1 = 19200 baud
2 = 9600 baud
3 = 4800 baud
4 = 2400 baud
5 = 1200 baud
6 = 600 baud
7 = 300 baud
8 = 115200 baud
9 = 57600 baud
10=230400 baud
11=76800 baud</td></tr>
<tr><td>FLOWCONTROL0</td><td>82</td><td>B82</td><td>1</td><td>0</td><td>Flow control for the serial port 0:
0 = no,
1= Software XON/XOFF,
2 = Hardware RTS/CTS,
8=RS485 Direction Control</td></tr>
<tr><td>Reserved</td><td>86</td><td>W86</td><td>2</td><td></td><td></td></tr>
<tr><td>Reserved</td><td>88</td><td>B88, B89, B90, B91</td><td>4</td><td></td><td></td></tr>
</table>

Parameter	Byte	Dynamic Name	Len	Default	Short Description
Reserved	92	W92	2		
Security Settings	97	B97	1	0	bit0: 0 - reset function enabled, 1- reset function disabled bit1: 0 - factory defaults enabled, 1 - factory defaults disabled bit2:0 – web update enabled, 1 – web update 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	4	Version Minor value (do not change)
Setupex Length	120	W120	2	894	Length of the extended setup (always 894)
Password Level 1	122	S122	8		Password stored as a MD5 hash (first 8 bytes) used for viewing and changing the configuration, all 0 means no password
Passwords Level 2-6 Reserved	130-179				
Application Name	180	S180	16	annunfdx	BCL application name (without the extension) including the terminating NULL character.
Web Server Port	196	W196	2	0	Port on which built-in webserver is running. Range: 1...65535
Media Configuration	198	B198b0, B198b1, B198b2, B198b3, B198b4, B198b5, B198b6, B198b7	1	0x00	Function is activated by setting the appropriate bit: 0x01: not used 0x02: not used 0x04: not used 0x08: not used 0x10: not used 0x20: not used 0x40: not used 0x80: 0 – SonicIP on, 1 – SonicIP off
Serial port usage	199	B199	1	1	Defines for what the serial port is used: 0 = serial GW 1 = VSC panel
Syslog Address	200	B200,B201, B202,B203	4	0	Destination address for syslog messages. If set to 0 (default), syslog is broadcasted.
LED Control	204	B204	1	0	On devices supporting LEDs on the front panel (Exstreamer 1000, Annuncicom

Parameter	Byte	Dynamic Name	Len	Default	Short Description
					1000) this parameter selects the source controlling the LEDs. On other devices this setting is ignored. 0 = LEDs display the status of the digital inputs 3 = LEDs display the status of the relays 4 = LEDs are mapped to the IO map (digital outputs) and can be controlled by the BCL application
System Name	205	S205	18	“”	SNMP system name.
System Location	223	S223	18	“”	SNMP system location.
System Contact	241	S241	18	“”	SNMP system contact.
IR Source	259	B259	1	0	IR receiver type: 0= Serial IR Dongle 1= Built-in IR receiver
VSC Font Size	260	B260	1	0	Font size selection for the new VSC panel (without rotary knob): 0 = large font (2 characters per line) 1 = small font (8 characters per line)
Reserved	261-499				
Application Config	500-1499		1000		Assigned to user applications (see Error: Reference source not found for more details)

6 Hardware Controls

This section describes the Reset Button functionality and the status information provided by the LEDs.

6.1 The Reset Button

A short press of the Reset Button resets the device. This can be disabled in the Web Configuration Interface.

In case the Reset Button is held for more than 10 seconds, factory default settings is restored. This is indicated by green LED being on and red LED blinking. Factory defaults feature can be disabled in the Web Configuration Interface.

Note: During power-up, the reset button has also another function. If reset button is held during power-up, it will bring the device to the Boot Menu mode. If that wasn't your intention, just reset the device again.

6.2 Device Availability with the Green and Red Status LEDs

During the initial boot up sequence the 2 status indicator LEDs are used as for standard Barix products.

No Application loaded (only bootloader) or reset button held during power up

Green LED	Red LED	Status
ON	BLINKING	No application loaded (only bootloader) or reset button held during power up

Application starts (Barix boot-up sequence):

First the red goes on and the green LED blinks once. Then during the startup the green and red LEDs blink.

During DHCP the red LED blinks with a continuous cycle . The green LED blinks five times and then pause four times.

If an error is detected the red led remains on and the device resets itself after the green LED has indicated the error as follows:

Green LED	Red LED	Status
5x BLINK	BLINKING	Corrupt application or IP address conflict
3x BLINK	OFF	The Network hardware could not be initialized or a Corrupt

Green LED	Red LED	Status
		MAC address
ON	ON	Reset Button is being pressed, the unit will reboot after releasing the button.
ON	BLINKING	Reset Button has been kept pressed and the unit is ready to set factory defaults after releasing the button.

Application running

After the boot-up when the application starts red and green LED indicate the following states:

Green LED	Red LED	State
OFF	BLINKING	Application is detecting USB devices or announcing the assigned IP address using SonicIP technology
ON	OFF	The application is operational.
ON	ON	Reset Button is being pressed, the unit will reboot after releasing the button.
ON	BLINKING	Reset Button has been kept pressed and the unit is ready to set factory defaults after releasing the button.

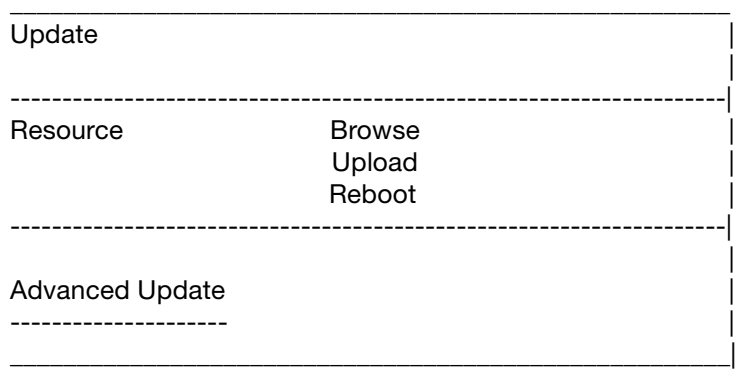
7 Firmware and standard File Upload

The standard Barix method is available to update the units with Firmware, the WEB UI, default configuration and language files. This method is summarized..

Update via a standard web browser

Please Note:
The update described below will NOT change the current configuration settings!
Reverting to Factory Defaults is not needed but recommended.

1. Open a browser and type the announced IP address into the URL field and hit the ENTER key.
2. Click on the CONFIGURATION button to enter the configuration pages.
3. Click on the UPDATE button to enter the update page.
4. Click on "Please click here to continue" to launch the update process. The device will restart in a special mode called Bootloader showing a number counting down. Upon start up the following screen appears ready for the update process. (The word Browse may differ depending on browser and language).

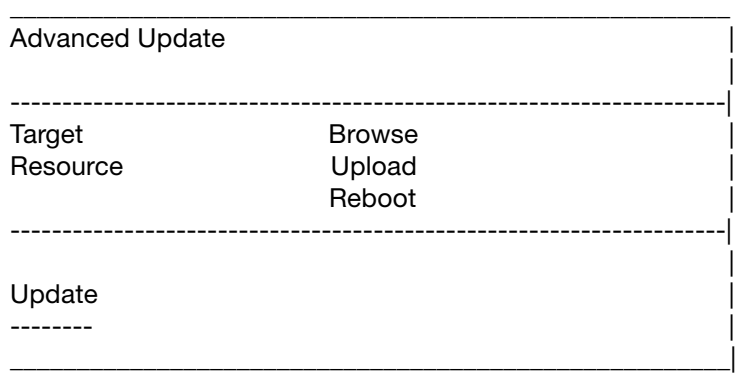


5. To upload an update click on "Browse" to locate the file you want to update. Browse to the folder "update_rescue" and choose the file compound.bin.
6. Once selected, click on "Upload". This process can take a few minutes. After a successful upload click on the "update" link and when the Update window reappears click the "Reboot" button or if there is no button, click on Browse and select the file "reboot". The device will reboot with the new firmware.

7.1 Advanced update

Individual files may be loaded using the Advanced Update.

1-4 Steps 1-4 above
5. Click on "Advanced Update" and the following screen appears ready for the advanced update process. (The word Browse may differ).



6. For advanced updates an additional parameter is required called the "Target". This specifies where in the Flash to load the file as follows:

Type	Resource	Target	
Firmware	abclw.rom	8K	
WEB UI and Sonic IP	abclapp.cob	WEB3	
FW Ext. 1	fs.bin	WEB1	
FW Ext. 2	sg.bin	WEB2	
FW Ext. 3	bclio.bin	WEB9	
Application programs	applications.cob		WEB10
Customcustom application	custom1.cob	WEB6	
Bootloader	UNIFULL.SPB	Loading the Boot Loader over the WEB is not normally recommended, please ask Barix for advice.	

7. To make an update type in the Target and then click on "Browse" to locate the corresponding file in the "update_rescue" folder.
8. Once selected, click on "Upload". This process can take up to one minute. After a successful upload click on the "update" link and when the Update window reappears click the "Reboot" button.

The device will reboot with the new resource file.

8 Appendix A: BCL I/O Address Map

ABCL provides access to peripherals of the hardware through I/O registers. They are numbered in ascending sequence starting from 1 and can hold an integer number (value range depends on the hardware and is defined below). I/O registers can be set using the **`IOCTL`** call and read using the **`IOSTATE`** call.

I/O registers are split into groups by their addresses, the groups are associated with particular inputs or outputs (e.g. relays, digital outputs, digital inputs etc.). Some of the registers are only virtual and are not mapped to any hardware, these can be used by the application to store an information which can be then retrieved e.g. via SNMP.

8.1 Hardware identification

Virtual registers starting from number 60000 describe the hardware ABCL runs on. These registers are read-only and have the following meaning:

Register	Description
60000	Hardware type identifier
60001	Number of serial ports
60002	Number of relays
60003	Number of digital outputs (excluding RTS/CTS)
60004	Number of digital inputs (excluding RTS/CTS)
60005	Number of analog outputs
60006	Number of analog inputs
60007	Number of keys if the hardware features a keyboard; 0 if no keyboard is available
60008	IPAM type identifier

Device overview

This section provides a list of hardware types and device capabilities supported by ABCL.

Device name	HW ID	Serial ports	Relays	Digital outp.	Digital inp.	Analog outp.	Analog inp.	Display	Keyboard
Exstreamer Red Box	1	1	0	0	0	0	0	none	none
Annuncicom legacy Annuncicom 100	7	1	1	0	2	0	0	none	2 keys (digital inputs)
Instreamer Instreamer 100	8	1	0	0	0	0	0	none	none
Exstreamer Digital	9	1	0	0	0	0	0	none	none
IPAM 100	13	2	0	8	8	0	0	none	none
Exstreamer 100 IPAM 200	14	1	0	4	4	0	0	None	none

Device name	HW ID	Serial ports	Relays	Digital outp.	Digital inp.	Analog outp.	Analog inp.	Display	Keyboard
IPAM 300									
IPAM 301	45	1	0	4	4	0	0	None	none
IPAM 302	46	1	0	4	4	0	0	None	none
OEM362	51	1	0	3	3	0	0	None	None
Exstreamer 200	15	1	0	0	0	0	0	none	none
IPAM 100 Carrier Board	16	2	0	8	8	0	0	none	none
Annuncicom 1000	17	1	9	0	9	0	2	none	8 keys (digital inputs)
Annuncicom 200	19	1	1	0	2	0	0	none	2 keys (digital inputs)
Exstreamer 110	20	1	1	0	0	0	0	2x16	none
Exstreamer 1000	21	1	4	0	4	0	1	none	4 keys
Annuncicom PS16	25	0	0	16	16	0	0	2x24	16 keys (keypad)
Annuncicom 155	32	0	1	1	2	0	3	none	2 keys (digital inputs)
Exstreamer 120	34	1	1	0	0	0	0	2x16	none
Exstreamer 500	35	1	4	0	4	0	1	none	4 keys
Exstreamer P5	36	1	0	0	0	0	0	none	none
Exstreamer 105	37	1	0	0	0	0	0	none	none
Exstreamer 205	38	1	0	0	0	0	0	none	none
IPAM 101	39	2	0	8	8	0	0	none	none
IPAM 102	40	2	0	8	8	0	0	none	none
Annuncicom 60	41	1	0	0	1	0	0	none	1 key (digital input)
Annuncicom PS1	42	0	1	0	2	0	0	none	2 keys (digital inputs)

IPAM overview

The below table lists IP Audio Module types supported by ABCL. The IPAM type can be obtained by reading register 60008.

The IPAM type provides additional information to the application to properly identify the hardware device and e.g. enable/disable the right set of features, audio formats, etc.

Please note that two devices with the same HW type might contain a different IPAM. For instance Annuncicom 100 can contain IPAM 100 or IPAM 102.

ID	Name	Note
0	Legacy	Interpret as “no information”. Can be IPAM 100, 200, 300, 101 or 102.

ID	Name	Note
1	IPAM 101	
2	IPAM 102	
3	IPAM 301	
4	IPAM 302	

8.2 Status LED control

Two write-only virtual registers 61000 and 61001 control the behaviour of device's red and green status LEDs. Each register holds a 16-bit pattern, that is rotated every 250ms and causes blinking of the respective LED. A bit value of 1 turns the LED on, a bit value of 0 turns the LED off.

Register	Description
61000	Red status LED control
61001	Green status LED control

Example

The pattern 0xffff turns the LED permanently on. The pattern 0x0000 turns the LED permanently off. The pattern 0x5555 causes fast flashing of the LED.

```
ioctl 61000,&H5555 ' red LED flashing
ioctl 61001,&Hffff ' green LED permanently on
```

8.3 I/O registers

This section describes the layout of the I/O hardware registers.

The input and output capabilities depend on the hardware the ABCL runs on and not all I/Os may be present on all hardware (e.g. there are no relays are available on

Exstreamer 100). Write to these registers has no effect and read always returns 0. List of available registers for each hardware is defined below.

I/Os are always numbered from the lowest number in ascending order up to the number of relays, inputs, etc.

E.g. a device with 4 digital inputs uses the following registers:

201 - 1st digital input
202 - 2nd digital input
203 - 3rd digital input
204 - 4th digital input

Exception:

RTS and CTS on serial ports are accessible through digital outputs 200, 199, 198, ... (RTS1, RTS2, RTS3, ...) and through inputs 300, 299, 298, ... (CTS1, CTS2, CTS3, ...)

1...100 Relay outputs (Read, Write)

I/O	Output
1	Relay 1
2	Relay 2
...	...
32	Relay 32
33..100	Reserved for future use

value	Function
0	Set output to inactive (relay contact open)
1	Set output to active (relay contact closed)
999	Toggle output
n	Pulse output for n*100ms,
	valid values 2-998

Read from a relay output returns the current state of the relay. The default state of relay outputs (after reset) is **inactive**.

If set with the register write command to "`999`", the output toggles. If set with the register write command to a value >`1`, output generates a pulse of duration n*100 milliseconds. The pulse appears on the output within maximum 20ms time.

The registers 1 to 32 map to the relays, the addresses 33 to 100 are reserved for future internal use. The below table shows availability of relays on Barix hardware.

Hardware	Relays
Legacy Exstreamer devices	N/A
Exstreamer 100/105/P5 IPAM 200/IPAM 300 IPAM 301/IPAM 302/ OEM362	N/A
Exstreamer 110/120	1
Exstreamer 200/205	N/A
Exstreamer 500/1000	1..4
Legacy Instreamer	N/A
Instreamer 100	N/A
Legacy Annuncicom	1
Annuncicom 60	N/A
Annuncicom 100	1
Annuncicom 155	1
Annuncicom 200	1
Annuncicom 1000	1..9
Annuncicom PS1	1
IPAM 100/101/102/carrier board	N/A
Annuncicom PS16	N/A

101...200 Digital outputs (Read, Write)

I/O	Output
101	Output 1
102	Output 2
...	...
132	Output 32
133..196	Reserved
197..198	Reserved for further serial ports
199	RTS out on 2. serial port
200	RTS out on 1. serial port

value	Function
0	Set output to inactive
1	Set output to active
999	Revert (toggle) output
n	Pulse output for n*100ms,
	valid values 2-998

Read from a digital output returns the current state of the output.

If set with the register write command to "**999**", the output toggles. If set with the register write command to a value >**1**, output generates a pulse of duration n*100 milliseconds. The pulse appears on the output within maximum 20ms time.

Active means "high" state on the output (on serial port RTS asserted). **Inactive** means "low" state on the output (on serial port RTS not asserted).

Important note: The RTS signal is by default (after reset) **active** as it is expected to be used by the flow-control. If used for other purposes than signaling (e.g. to control external hardware) this fact should be taken into account when connecting the external hardware (and use e.g. a signal invertor) to avoid problems like door being open after reset.

The addresses 101 to 132 are mapped to the digital outputs on the hardware, the addresses 133 to 196 are reserved for future use.

Addresses 199 and 200 are mapped to the RS-232 RTS output signal (on serial 2 and serial 1 respective) and should only be addressed if HW flow control is disabled. The addresses 197 and 198 are not used and reserved for the future for serial ports 4 and 3. Read from an RTS output is valid **only if used as I/O** and not when used for the flow-control.

The below table lists available digital outputs on Barix hardware.

Hardware	Digital Outputs	RTS Outputs
Legacy Exstreamer devices	N/A	200
Exstreamer 100/IPAM 200/IPAM 300 IPAM 301/IPAM 302	101..104	200
OEM362	101..103	200
Exstreamer 105/110/120	N/A	200
Exstreamer 200/205	N/A	200
Exstreamer 500/1000	If enabled 151..154	200
Exstreamer P5	N/A	N/A
Legacy Instreamer	N/A	200
Instreamer 100	N/A	200
Legacy Annuncicom	N/A	200
Annuncicom 60	N/A	200

Hardware	Digital Outputs	RTS Outputs
Annuncicom 100	N/A	200
Annuncicom 155	101	N/A
Annuncicom 200	N/A	200
Annuncicom 1000	If enabled 151..158	200
Annuncicom PS1	N/A	N/A
IPAM 100/101/102	101..108	199, 200
Annuncicom PS16	101..116 (extensible)	N/A

On the IPAM 100, 101, 102, 200 and 300 the digital outputs and inputs are mapped directly to PIOs of the board. Read from a register (**`IOSTATE`**) switches automatically the PIO to input mode, writing to a register (**`IOCTL`**) switches the PIO to output mode. The below tables list the assignment of PIOs to digital outputs on IPAM 100, 101, 102, 200 and 300. PIOs 8, 24 and 25 are used by the ABCL firmware (software reset, red and green LEDs) and can not be controlled by the BCL application.

By default all PIOs are initialised as inputs with weak pull-up.

IPAM100/101/102/ IPAM 100 Carrier board digital output assignment

On the IPAM 100 Carrier Board the PIO17 is shared with the onboard RTC. The I/O pin can be used only for one purpose: either for RTC or as a PIO.

Digital Output	PIO
101	PIO11
102	PIO16
103	PIO17
104	PIO20
105	PIO22
106	PIO23
107	PIO29
108	PIO30
Software reset	PIO8
Green LED	PIO24
Red LED	PIO25

Exstreamer 100/IPAM200/IPAM 300/IPAM 301/IPAM 302 digital output assignment

Please note that all three devices have the same hardware type. The direct access to the PIOs is not available on the Exstreamer 100.

Digital Output	PIO

Digital Output	PIO
101	PIO11
102	PIO17
103	PIO29
104	PIO30
Software reset	PIO8
Green LED	PIO24
Red LED	PIO25

OEM362 digital output assignment

Digital Output	PIO
101	PIO11
102	PIO29
103	PIO30
Software reset	PIO8
Green LED	PIO24
Red LED	PIO25

Exstreamer 500/1000 digital output assignment

If enabled in the firmware settings the four front panel LEDs are mapped to the IO space as digital outputs 151 to 154.

Digital Output	PIO
151	LED 1
152	LED 2
153	LED 3
154	LED 4

Annuncicom 1000 digital output assignment

If enabled in the firmware settings the eight front panel LEDs are mapped to the IO space as digital outputs 151 to 158.

Digital Output	PIO
151	LED 1
152	LED 2
153	LED 3
154	LED 4
155	LED 5
156	LED 6
157	LED 7

Digital Output	PIO
158	LED 8

201...300 Digital inputs (Read only)

I/O	Input
201	Input 1
202	Input 2
...	...
232	Input 32
233..295	Reserved
296	Reset button (*see below)
297..298	Reserved for further serial ports
299	CTS input on 2. serial port
300	CTS input on 1. serial port

value	Default function (low-active)
0	Input is active
1	Input is inactive
2	Input is short-cut
3	Input not connected

Digital inputs use low-active logic. Please note that digital inputs not available on the device return value 0.

Input values are hardware dependent: most devices use two-state logic (0 = active, 1 = inactive), some devices define additional values for input supervision (2 = short-cut, 3 = not connected). See the table below and the hardware specific sections at the end of this document.

For CTS inputs **value 1** means signal asserted, whereas **value 0** means signal not asserted.

If a CTS input is used for flow-control, read is valid and reflects the current state of CTS.

The reset button can be read by the BCL application as a digital input at address 296. The return value is: 0 – pressed, 1 – not pressed. To disable the standard function of the reset button (reboot the device and revert factory defaults) set the bits 0 and 1 of the Security settings parameter (B97b0 and B97b1) in the Setup to 1.

The addresses 201 to 232 are assigned to digital inputs, the addresses 233 to 295 are reserved for future use. The addresses 299 and 300 are assigned to CTS inputs on the second and the first serial port respectively. The addresses 297 and 298 are reserved for future use for serial ports 3 and 4.

The below table shows the assignment of digital inputs on Barix hardware.

Hardware	Digital Inputs	Value	CTS Inputs

Hardware	Digital Inputs	Value	CTS Inputs
Legacy Exstreamer devices	N/A	N/A	300
Exstreamer 100 IPAM 200/IPAM 300 IPAM 301/IPAM 302	201..204	Low active (0=active)	300
OEM362	201..203	Low active (0=active)	300
Exstreamer 105/110/120	N/A	N/A	300
Exstreamer 200/205	N/A	N/A	300
Exstreamer 500/1000	201..204	Low active (0=active)	300
Exstreamer P5	N/A	N/A	N/A
Legacy Instreamer	N/A	N/A	300
Instreamer 100	N/A	N/A	300
Legacy Annuncicom	201..202	Low active (0=active)	300
Annuncicom 60	201	Low active (0=active)	300
Annuncicom 100	201..202	Low active (0=active)	300
Annuncicom 155	201..202	0 = input activated 1 = input not activated 2 = short circuit 3 = input open (not connected)	N/A
Annuncicom 200	201..202	Low active (0=active)	300
Annuncicom 1000	201..208	0 = input activated 1 = input not	300

Hardware	Digital Inputs	Value	CTS Inputs
		activated 2 = short circuit 3 = input open (not connected)	
Annuncicom PS1	201..202	Low active (0=active)	N/A
IPAM 100/101/102	201..208	Low active (0=active)	299, 300
Annuncicom PS16	201..216 (extensible)	Low active (0=active)	N/A

On the IPAM 100, 101, 102, 200 and 300 the digital outputs and inputs are mapped directly to PIOs of the board. Read from a register (**`IOSTATE`**) switches automatically the PIO to input mode, writing to a register (**`IOCTL`**) switches the PIO to output mode. The below table lists the assignment of PIOs to digital inputs on IPAM. PIOs 8, 24 and 25 are used by the ABCL firmware (software reset, red and green LEDs) and can not be controlled by the BCL application.

By default all PIOs are initialised as inputs with weak pull-up.

IPAM100/101/102/IPAM 100 Carrier board digital input assignment

On the IPAM 100 Carrier Board the PIO17 is shared with the onboard RTC. The I/O pin can be used only for one purpose: either for RTC or as a PIO.

Digital Input	PIO
201	PIO11
202	PIO16
203	PIO17
204	PIO20
205	PIO22
206	PIO23
207	PIO29
208	PIO30
Software reset	PIO8
Green LED	PIO24
Red LED	PIO25

Exstreamer 100/IPAM200/IPAM 300/IPAM 301/IPAM 302 digital input assignment

Please note that all three devices have the same hardware type. The direct access to the PIOs is not available on the Exstreamer 100.

Digital Input	PIO
201	PIO11
202	PIO17
203	PIO29
204	PIO30
Software reset	PIO8
Green LED	PIO24
Red LED	PIO25

OEM362 digital input assignment

Digital Input	PIO
201	PIO11
202	PIO29
203	PIO30
Software reset	PIO8
Green LED	PIO24
Red LED	PIO25

301...400 Virtual I/O bits (Read, Write)

I/O	Function
301...400	Virtual bits

value	Function
0	Bit is inactive
1	Bit is active

The addresses 301 to 400 are mapped to virtual 1-bit registers (implemented in memory only). They can be used as a memory storage or to pass an information to external devices e.g. through SNMP.

401...500 Reserved

These registers are reserved for future use. Write to these registers has no effect and read always returns 0.

501...600 Analog inputs (Read only)

The analog inputs are hardware specific.

Hardware	Analog inputs	Values
Legacy Exstreamer devices	N/A	N/A
Exstreamer 100 IPAM 200/IPAM 300 IPAM 301/IPAM 302/ OEM362	N/A	N/A
Exstreamer 105/110/120	N/A	N/A
Exstreamer 200/205	N/A	N/A
Exstreamer 500/1000	501 – temperature sensor	In 0.01°C units; value 0 represents 0°C
Exstreamer P5	N/A	N/A
Legacy Instreamer	N/A	N/A
Instreamer 100	N/A	N/A
Legacy Annuncicom	N/A	N/A
Annuncicom 60	N/A	N/A
Annuncicom 100	N/A	N/A
Annuncicom 155	501 – internal temperature 502 – microphone bias current 503 – speaker supervision measurement	In 0.01°C units; value 0 represents 0°C In 0.01V units; value 0 represents 0V TBD
Annuncicom 200	N/A	N/A
Annuncicom 1000	501 – internal temperature 502 – battery voltage	In 0.01°C units; value 0 represents 0°C In 0.01V units; value 0 represents 0V
Annuncicom PS1	N/A	N/A
IPAM 100/101/102	N/A	N/A
Annuncicom PS16	N/A	N/A

601...700 Reserved

These registers are reserved for future use. Write to these registers has no effect and read always returns 0.

701...1000 Virtual 16bit registers (Read, Write)

I/O	Function
701...1000	Virtual 16bit registers

value	Function
0...65535	Register value

The addresses 701 to 1000 are mapped to virtual 16-bit registers (implemented in memory only). They can be used as a memory storage or to pass an information to external devices e.g. through SNMP.

9 Appendix B: Hardware Devices

9.1 Instreamer Legacy/100, New Instreamer

IO	Type	Name	Note
200	Digital output	RTS	
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	

9.2 Exstreamer Legacy/105/200

IO	Type	Name	Note
200	Digital output	RTS	
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	

9.3 Exstreamer 205

IO	Type	Name	Note
200	Digital output	RTS	
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	

Exstreamer 205 features a line input for passing audio from external device (e.g. a portable MP3 player) to speaker amplifier. By default the line input is deactivated. To activate it in your BCL program, do the following:

1. open audio interface for decoding (MP3 or uncompressed)

2. enable “recording input” in output mixer

Example:

```
' pass line-input only signal to output
open ”AUD:1,0,0,0” as 7              ' open audio for MP3 decoding
write 7,str$(&H0040),-13             ' configure output mixer to:
                                     ' recording input on, playback off
write 7,”20”,-12                     ' set volume to 100%  (20*5% steps)
close 7


' mix line-input only signal to output
open ”AUD:1,0,0,0” as 7              ' open audio for MP3 decoding
write 7,str$(&H4040),-13             ' configure output mixer to:
                                     ' recording input on, playback on
write 7,”20”,-12                     ' set volume to 100%  (20*5% steps)

'...
' write data to audio interface
'...

close 7
```

9.4 Exstreamer 100/IPAM 200/IPAM 300/IPAM 301/IPAM 302

On the IPAM 200, IPAM 300, IPAM 301 and IPAM 302 the digital outputs and inputs are mapped directly to PIOs of the board. Read from a register (**IOSTATE**) switches automatically the PIO to input mode, writing to a register (**IOCTL**) switches the PIO to output mode. By default all PIOs are initialised as inputs with weak pull-up.

Please note that all three devices have the same hardware type. The direct access to the PIOs is not available on the Exstreamer 100.

IO	Type	Name	Note
101	Digital output	PIO11	
102	Digital output	PIO17	

103	Digital output	PIO29	
104	Digital output	PIO30	
200	Digital output	RTS	
201	Digital input	PIO11	Low active
202	Digital input	PIO17	Low active
203	Digital input	PIO29	Low active
204	Digital input	PIO30	Low active
296	Digital input	Reset Button (PIO8)	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	

9.5 OEM362

The OEM362 hardware is similar to IPAM 302, however the PIO17 controls an external amplifier. The digital outputs and inputs are mapped directly to PIOs of the board. Read from a register (**`IOSTATE`**) switches automatically the PIO to input mode, writing to a register (**`IOCTL`**) switches the PIO to output mode. By default all PIOs are initialised as inputs with weak pull-up.

IO	Type	Name	Note
101	Digital output	PIO11	
102	Digital output	PIO29	
103	Digital output	PIO30	
200	Digital output	RTS	
201	Digital input	PIO11	Low active
202	Digital input	PIO29	Low active
203	Digital input	PIO30	Low active
296	Digital input	Reset Button (PIO8)	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	

9.6 Exstreamer 110/120

IO	Type	Name	Note
1	Relay output	Relay 1	
200	Digital output	RTS	
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard

			function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	

Exstreamer 110 and Exstreamer 120 feature a 2x16 character LCD display.

9.7 Annuncicom 60

IO	Type	Name	Note
200	Digital output	RTS	
201	Digital input	IN0	Low active
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	

Digital input 1 can be read as a key event.

9.8 Annuncicom Legacy/100

IO	Type	Name	Note
1	Relay output	Annuncicom relay	
200	Digital output	RTS	
201	Digital input	IN0	Low active
202	Digital input	IN1	Low active
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	

Digital inputs 1 and 2 can be read as key events.

9.9 Annuncicom 200

IO	Type	Name	Note
1	Relay output	Annuncicom relay	
200	Digital output	RTS	
201	Digital	IN0 / AI Phone	Low active

	input	button	
202	Digital input	IN1	Low active
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	

Digital inputs 1 and 2 can be read as key events.

Aihpone interface on the Annuncicom 200

Annuncicom 200 features a two-wire AIPhone interface allowing to connect a speaker, a microphone and a call button in one. This interface is intended to provide a simple attachment for a door station. Thanks to its hardware nature, this interface requires a special attention when using:

The call button is shared with the input 0. When pressed, it draws an extra current from the Annuncicom 200, which significantly lowers the audio output of the AIPhone speaker. Therefore the button can not be used as push-to-talk – it must be used as a ring button (to attract an attention on the called side).

The call **button must be ignored** by the application **if audio is being transmitted in any direction**. If audio is transmitted phantom button presses may occur.

The audio interface is purely half duplex and must be used so; operating the AIPhone interface in full-duplex causes an acoustic feedback. The audio interface is controlled by the audio parameter “AUX1 control” and by default is off. The following table contains the proper settings in encoding and decoding:

Mode	AUX1 control	Note
Off (default)	0	Aiphone interface off, no audio is transmitted. However, the Aiphone is still powered
Encoding	1	Mic full sensitivity, medium speaker volume
Decoding or “no connection”	2	Maximum speaker drive level, very low mic sensitivity The Aiphone power is lowered, can be also used if there is no connection (as reduced power mode)

Aiphone direction switching

The direction switching on the Aiphone interface requires special attention.

If the audio interface in BCL (the “AUD” handle) is open in half duplex the following sequence must be used:

2. stop audio (close)

3. reconfigure the Aiphone direction (AUX1 control parameter)
4. start (open) audio again.

Under certain conditions the underlying audio interface in BCL (the “AUD” handle) can be run in full-duplex to reduce the switch-over time. Please note that through an Aiphone the audio can not be transferred in both directions simultaneously. This would cause an acoustic feedback.

Since the Aiphone always feeds back the audio sent to it, if audio is open in full-duplex mode data can be sent only in one direction (the other direction must be ignored). In addition to that, the internal buffers must be flushed before switching over.

To achieve the minimum turnaround time with full-duplex follow the below sequence:

1. by default Aiphone is off
2. open audio in full-duplex
3. switch Aiphone to decoding (set AUX1 control to 2)
4. Decoding:
5. write to the audio interface but do not read anything from it
6. To switch over to encoding:
7. switch off the Aiphone (set AUX1 control to 0)
8. stop writing to the audio interface
9. flush the decoding audio buffer
10. wait a few milliseconds to make sure that the DSP's internal buffer is played out
11. flush the encoding audio buffer
12. switch Aiphone to encoding (set AUX1 control to 1)
13. Encoding:
14. read from the audio interface but do not write anything into it
15. To switch over to decoding:

16. stop reading from the audio interface
17. switch Aiphone to decoding (set AUX1 control to 2)

9.10 IPAM 100/101/102/IPAM 100 Carrier Board

On the IPAM board the digital outputs and inputs are mapped directly to PIOs of the board. Read from a register (**`IOSTATE`**) switches automatically the PIO to input mode, writing to a register (**`IOCTL`**) switches the PIO to output mode. By default all PIOs are initialised as inputs with weak pull-up.

On the IPAM Carrier Board the PIO17 is shared with the on-board RTC. Only one function of the PIO17 can be used at a time – either the RTC or the I/O function.

IO	Type	Name	Note
101	Digital output	PIO11	
102	Digital output	PIO16	
103	Digital output	PIO17	On IPAM 100 Carrier Board shared with RTC
104	Digital output	PIO20	
105	Digital output	PIO22	
106	Digital output	PIO23	
107	Digital output	PIO29	
108	Digital output	PIO30	
199	Digital output	RTS on serial 2	
200	Digital output	RTS on serial 1	
201	Digital input	PIO11	Low active
202	Digital input	PIO16	Low active
203	Digital input	PIO17	Low active On IPAM 100 Carrier Board shared with RTC
204	Digital input	PIO20	Low active
205	Digital input	PIO22	Low active
206	Digital input	PIO23	Low active
207	Digital input	PIO29	Low active
208	Digital input	PIO30	Low active
296	Digital input	Reset Button (PIO8)	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
299	Digital input	CTS on serial 2	
300	Digital input	CTS on serial 1	

Digital inputs 1-8 can be read as key events.

9.11 Exstreamer 500/1000

IO	Type	Name	Note
1	Relay output	Relay 1	
2	Relay output	Relay 2	
3	Relay output	Relay 3	
4	Relay output	Relay 4	
151	Digital output	LED 1	Access to the front panel LEDs This option must be enabled in the Firmware settings By default the LEDs are controlled by the driver and reflect the state of the digital inputs. 0 = LED is off 1 = LED is on
152	Digital output	LED 2	
153	Digital output	LED 3	
154	Digital output	LED 4	
200	Digital output	RTS	
201	Digital input	Input 1	Low active
202	Digital input	Input 2	Low active
203	Digital input	Input 3	Low active
204	Digital input	Input 4	Low active
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	
501	Analog input	Internal temperature	In 0.01°C units; value 0 represents 0°C

Digital inputs 1-4 can be read as key events.

9.12 Annuncicom 1000

IO	Type	Name	Note
1	Relay output	Relay 1	
2	Relay output	Relay 2	
3	Relay output	Relay 3	
4	Relay output	Relay 4	

5	Relay output	Relay 5	
6	Relay output	Relay 6	
7	Relay output	Relay 7	
8	Relay output	Relay 8	
9	Relay output	Relay 9	Default state is “closed” (value=1)
151	Digital output	LED 1	Access to the front panel LEDs This option must be enabled in the Firmware settings By default the LEDs are controlled by the driver and reflect the state of the digital inputs. 0 = LED is off 1 = LED is on
152	Digital output	LED 2	
153	Digital output	LED 3	
154	Digital output	LED 4	
155	Digital output	LED 5	
156	Digital output	LED 6	
157	Digital output	LED 7	
158	Digital output	LED 8	
200	Digital output	RTS	
201	Digital input	Input 1	0 = input activated 1 = input not activated 2 = short circuit 3 = input open (not connected)
202	Digital input	Input 2	
203	Digital input	Input 3	
204	Digital input	Input 4	
205	Digital input	Input 5	
206	Digital input	Input 6	
207	Digital input	Input 7	
208	Digital input	Input 8	
209	Digital input	Main input power status	Low – Main Supply >=17V High – Main Supply <17V
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1
300	Digital input	CTS	
501	Analog input	Internal temperature	In 0.01°C units; value 0 represents 0°C
502	Analog input	Battery input voltage	In 0.01V units; value 0 represents 0V

Digital inputs 1-8 can be read as key events. States 0 and 2 are treated as key pressed, states 1 and 3 as key released.

9.13 Annuncicom PS16

IO	Type	Name	Note
101	Digital output	LED 0	

102	Digital output	LED 1	
103	Digital output	LED 2	
104	Digital output	LED 3	
105	Digital output	LED 4	
106	Digital output	LED 5	
107	Digital output	LED 6	
108	Digital output	LED 7	
109	Digital output	LED 8	
110	Digital output	LED 9	
111	Digital output	LED 10	
112	Digital output	LED 11	
113	Digital output	LED 12	
114	Digital output	LED 13	
115	Digital output	LED 14	
116	Digital output	LED 15	
201	Digital input	Key 0 (lower left)	0 = key pressed 1 = key not pressed
202	Digital input	Key 1	
203	Digital input	Key 2	
204	Digital input	Key 3	
205	Digital input	Key 4	
206	Digital input	Key 5	
207	Digital input	Key 6	
208	Digital input	Key 7 (lower right)	
209	Digital input	Key 8 (upper left)	
210	Digital input	Key 9	
211	Digital input	Key 10	
212	Digital input	Key 11	
213	Digital input	Key 12	
214	Digital input	Key 13	
215	Digital input	Key 14	
216	Digital input	Key 15 (upper right)	

The Paging console features a 2x24 character LCD display and a 16-key keyboard.

The keyboard can be either polled directly through digital inputs or read as key events (key presses and releases). Every key on the keyboard has a LED, the LEDs are mapped to digital outputs corresponding to the respective digital input of the key.

The keyboard can be extended up to 48 keys, the number of digital inputs and outputs then increases.

PS16 Extensions

The PS16 Paging Station can be extended with up to two 48-key extension modules offering 112 keys in total. The extensions are detected at boot-up time.

The keys are numbered in the following way:

- keys 0..15 located on the PS16
- keys 16...63 located on the 1st extension (next to the PS16 unit)
- keys 64...111 located on the 2nd extension (next to the 1st extension)

The keyboard can be either polled directly through digital inputs or read as key events (key presses and releases). Every key on the keyboard has a LED, the LEDs are mapped to digital outputs.

The extended digital outputs are mapped to IO 1001..2000, the extended digital inputs are mapped to IO 2001..3000. For compatibility the first 100 outputs are mapped to the IO range 101..200; the first 100 inputs are mapped to the IO range 201..300.

IO	Type	Name	Note
1001	Digital output	LED 0	Mirrored to 101-200 0 = turn the light off 1 = turn the light on
1002	Digital output	LED 1	
...			
1100	Digital output	LED 99	
1101	Digital output	LED 100	0 = turn the light off 1 = turn the light on
...			
1112	Digital output	LED 111	
1113-2000		Reserved for future use	
2001	Digital input	Key 0	Mirrored to 201-300 0 = key pressed 1 = key not pressed
2002	Digital input	Key 1	
...			
2100	Digital input	Key 99	
2101	Digital input	Key 100	0 = key pressed 1 = key not pressed
...			
2112	Digital input	Key 111	
2113-3000		Reserved for future use	

9.14 Annuncicom 155

IO	Type	Name	Note

1	Relay	LED output	
101	Digital output	Enable speaker test 11Hz tone	0 = speaker test disabled (11Hz tone off) 1 = perform speaker test using 11Hz carrier tone
201	Digital input	Input 1	0 = input activated 1 = input not activated 2 = short circuit 3 = input open (not connected)
202	Digital input	Input 2	
210	Digital input	Status of last Speaker test	0 = speaker loop current within valid range 1 = wrong loop current (i.e. speaker broken/not connected) 2 = invalid (no measurement available)
211	Digital input	Microphone bias current range	0 = current is within valid range (0.2-2mA) 1 = current is outside valid range
501	Analog input	Internal temperature	In 0.01°C units; value 0 represents 0°C
502	Analog input	Microphone bias current	In 1uA units; value 0 represents 0uA
503	Analog input	Speaker test output value	-1 = invalid (no measurement available) 8- bit valueTBD

Digital inputs 1-2 can be read as key events. States 0 and 2 are treated as key pressed, states 1 and 3 as key released.

9.15 Exstreamer P5

IO	Type	Name	Note
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1

9.16 Annuncicom PS1

IO	Type	Name	Note

1	Relay output	Flashing front panel light – call indication	
201	Digital input	Button A	Low active
202	Digital input	Button B	Low active
296	Digital input	Reset Button	To use the reset button as a digital input, disable the standard function by setting Setup parameters B97b0 and B97b1 to 1

Digital inputs 1 and 2 can be read as key events.

10 Appendix C: Peripherals

10.1 VSC Panel – with rotary knob

To connect and set up the VSC panel:

- set the address selector rotary switch on the rear of the VSC panel to position “1”
- connect the VSC panel to the first serial port of the device using the VSC conversion plug and an RJ45 straight-cable
- select the “VSC Panel” in ABCL's serial settings (Setup parameter B199, see chapter 5.3 above)

The VSC panel features three interfaces: a 2-digit display, a rotary knob and an IR receiver.

The IR receiver is interfaced via handle -3 (see the BCL Programmers Manual).

Both the display and the rotary knob can be accessed from BCL via handle -2. The display is available only on devices without an internal display (not available on Exstreamer 110). See the BCL Programmers Documentation for more details how to drive the display.

The rotary knob generates key-events with event source set to 0x01:

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
							0x01	Pressed	Key number						

Where “Key number” is:

Key number	Description
0x00	Knob pressed (generates one event on press and another event on release)
0x01	Knob turned left (always generates two events: press and release)
0x02	Knob turned right (always generates two events: press and release)

10.2 VSC Panel new – with buttons

To connect and set up the VSC panel:

- connect the VSC panel to the first serial port of the device using the VSC conversion plug, the voltage adaptor and two RJ45 straight-cables.
  On the Exstreamer P5 attach the VSC to the “Control” port using two straight RJ45 cables and the voltage adaptor.
- select the “VSC Panel” in ABCL's serial settings (Setup parameter B199, see chapter 5.3 above)
- optionally select the small font size (IC Paging firmware) for 8 characters per line (Setup parameter B260, see chapter 5.3 above)

The VSC panel features three interfaces: a 2 or 8 character display (depending on the font size selection), a 4-button keypad and an IR receiver.

The IR receiver is interfaced via handle -3 (see the BCL Programmers Manual).

Both the display and keypad can be accessed from BCL via handle -2. The display is available only on devices without an internal display (not available on Exstreamer 110). See the BCL Programmers Documentation for more details how to drive the display.

The keypad generates key-events with event source set to 0x01:

The rotary knob generates key-events with event source set to 0x01:

b15	b14	b13	b12	b11	b10	b9	b8	b7	b6	b5	b4	b3	b2	b1	b0
							0x01	Pressed	Key number						

Where “Key number” is:

Key number	Description
0x00	The “src” button
0x01	The “-” button
0x02	The “+” button
0x03	The “power” button

On button press an even with bit7 set is generated, on button release another event with bit7 cleared is generated.

11 Document Management

11.1 Revision History

SW Version	Author	Description	Date
0.10	PK	Initial Working Draft	16. May 2007
0.11	PK	Added BCL Development Kit	7. August 2007
0.12	PK	BCL I/O Map	31. October 2007
0.13	PK	I/O Map for IPAM	9. November 2007
0.14	PK	Version A0.13: updated Setup record (syslog address)	25. April 2008
0.15	PK	Version A0.15: hardware identification added to IO map	10. July 2008
	PK	Corrected error in Syslog destination address (EEPROM layout), 4 bytes are used	21. July 2008
0.16	PK	Added hardware capabilities overview Added PS16 Paging Station	17. September 2008
	PK	Added hardware type 25 for the PS16 Paging Station	26. September 2008
0.17	PK	Dynamic mark example corrected	11. March 2008
0.18	PK	FLASH reorganisation for bigger applications.cob	6. November 2008
0.19	PK	Added esnd.rom into FLASH, updated FLASH layout	9. January 2009
	PK	Enabled PIO17 on the IPAM carrier board	4. February 2009
	PK	Enabled access to PIO11, PIO17, PIO29 and PIO30 on IPAM 200/Exstreamer 100	6. February 2009
	PK	Described Annuncicom 200 AIPhone interface	3. February 2009
	PK	Added serial port configuration B199 into Setup	20. March 2009
0.21	JP	Released	4. May 2009
	PK	Described PS16 extensions Added B204 into Setup LED control on Exstreamer 1000 and Annuncicom 1000	23. June 2009
0.24	JP	QoS/DSCP added.	28. July 2009

SW Version	Author	Description	Date
0.25	KS	Setup. Added: Ethernet mode, offset 9	22. September 2009
	KS	Added DEFAULTS (c=94) (available in FW V0.24)	25. September 2009
	PK	Added reset button as a digital input, IO address 296	17. December 2009
	PK	Speaker supervision for Annuncicom 155	30. July 2010
	JP	SNMP system parameters added in EEPROM	8. October 2010
	PK	New Setup parameter B259: IR source	6. September 2010
0.26	PK	Red/green status LED control via IO registers 61000 and 61001 &LState(...,28) – HW type	15. February 2011
0.37	PK	Removed obsolete section Network Interface LEDs	23. May 2011
0.38	PK	Updated to reflect the newest state of the package Described mimetype.ini Updated list of hardware devices Updated description of &LState(3	2. September 2011
1.01	PK	Codec type, features and supported modes exported as dynamic marks 43-45.	13. February 2012
1.02	PK	IPAM type added as register 60008	28. February 2012
1.03	PK	IPAM type overview Described new HW types: IPAM 101, 102, Annuncicom 50, Annuncicom PS1 Minor corrections in section Digital Inputs Line-in usage on Exstreamer 205 Dynamic mark for IPAM type	19. March 2012
1.05	PK	FLASH layout changed	19. April 2012
1.06	PK	Added DSP info about 5-band equaliser and AEC	31. May 2012
1.09	PK	Minor corrections: WEB page for custom application, Wine link	12. November 2012
1.14	PK	Added EEPROM field B260: font selection for new VSC panel	18. August 2014
	PK	Added section for new VSC panel in Appendix C: Peripherals Web method updated to use POST for setup.cgi	20. August 2014
1.15	PK	Added IO control for IPAM 301 and 302	26. January 2015

SW Version	Author	Description	Date
1.16	PK	Added support for OEM362 hardware type	21. May 2015
1.17	PK	Annuncicom 50 renamed to Annuncicom 60	1. September 2015
1.19	ASI	SONG-1: Fixed 16 and 32 kHz audio encoding issues	8. September 2017
1.20	ASI	Incremented the SONG module version to 10.17	12. September 2017
1.21	ASI	Incremented the SONG module version to 10.18	19. October 2017

11.2 References

Nr.	Doc Title	Author	Date
[1]	EEPROM Setup Record For “Barix ABCL Applications”	P. Kulhavý	12. March 2007
[2]	BCL Programmers Manual v1.18	P. Kulhavý J.J. Zvánovec J. Rietschel	05. March 2013

12 Legal Information

For information about our devices and the latest version of this manual please visit 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
email: sales@barix.com
support: support@barix.com
wiki: wiki.barix.com