Brutus Usage Guide

STB_Brutus-SWUM300-R

5300 California Avenue • Irvine, CA 92617 • Phone: 949-926-5000 • Fax: 949-926-5203

03/13/08

Revision History

Revision	*Date*	*Change Description*
STB_Brutus-SWUM300-R	03/13/08	Initial release.

Broadcom Corporation
5300 California Avenue
Irvine, CA 92617

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

Section 1: Introduction

BRUTUS OVERVIEW

Brutus is a demonstration application that is delivered as part of the Broadcom Set-Top Reference Software release. It offers a wide variety of functionality in an integrated architecture and provides an example of how the Set-Top Reference Software can be used to create a final product. It also provides a means to test and use Broadcom reference boards, with actual functionality depending on the platform. Brutus is not intended to be a production-quality application, but its source code can be used as the basis for real applications.

To get going quickly, refer to the *Brutus Installation Guide* (document number STB_Brutus-SWUM200-R) for instructions on how to configure, build, and run Brutus.

Some of the features supported in Brutus are:

- Front-end Interfaces:
  - QAM64, QAM256, QAM1024
  - QPSK DVB and DIRECTV, 8PSK
  - VSB
  - Analog RF and A/V line input
- S-Video, Composite, Component, DVI/HDMI, RF modulator, L/R audio, S/PDIF outputs supported
- Single or dual decode
- Single or dual output
- Picture-in-picture (PIP)
- Channel change
  - PSI scanning
  - Programmable channel map
- PVR
  - Single or dual record and playback
  - Time-shifting (simultaneous record and playback)
  - Single or dual channel encoder
  - Wide range of host, decoder, and STC trick modes (including Broadcom proprietary trick modes like 1x rewind)
  - DES/3DES PVR encryption and decryption
- GUI
  - Broadcom embedded windowing system and widget set
  - Anti-aliased TrueType fonts supported through FreeType
  - Image rendering (including PNG, JPEG, and BMP support)
  - Wide range of user input devices supported
  - Single- or dual-instance GUI

- On-screen status
  - Constellations
  - Front-end and back-end Status
  - Playback and decode FIFO monitors

Broadcom's Reference Software team also develops numerous example applications, utilities, and test applications. They can be found in the source code tree in the following directories:

- BSEAV/api/examples
- BSEAV/api/utils

THE REFERENCE API SOFTWARE STACK

Brutus is the topmost layer of the Reference Software Release, a complete top-to-bottom set-top box software stack. The Reference API, shown in Figure 1, consists of four main layers: Brutus and example/utility applications, the Settop API, the Porting Interface (PI), and the Operating System.

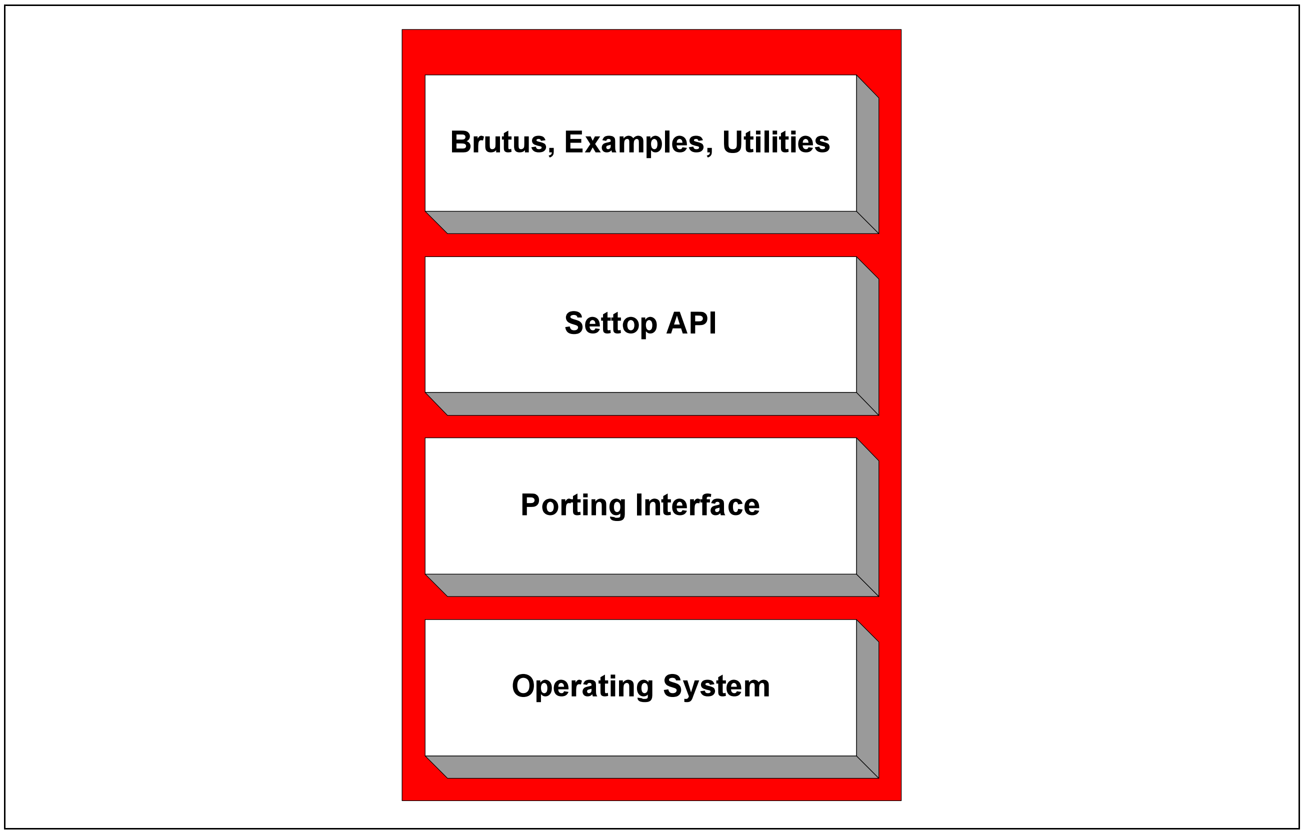

Figure 1: Reference API Software Stack

THE SETTOP API

The Settop API is an application-level C-language API that offers cross-platform, cross-OS portability. It implements a single usage model that is easily understood and extensible to support the features of Broadcom reference set-top boxes.

The Settop API handles a wide range of system programming tasks internally. These include:

- Interrupt handling
- Memory management
- Multi-threading and synchronization
- Optimal file I/O for PVR
- Multiple OS support

THE PORTING INTERFACE

The porting interface is a low-level C-language API that offers full control of Broadcom set-top chips. This API:

- is modularized for each block on the chip.
- has a consistent API and design methodology.
- supports interrupt safety through clearly marked ISR code.
- is portable across multiple operating systems like Linux and VxWorks.
- supports single- and multi-threading systems.
- is fully ANSI C-compliant.

On BCM7038-based platforms, the porting interface is known as “Magnum.” For previous platforms, it was referred to simply as the “Porting Interface.” This layer also includes other elements like base modules and system libraries.

THE OPERATING SYSTEM

Broadcom supports a variety of operating systems, including Linux and VxWorks. While each layer in the architecture requires some operating-system-specific code, a large majority of the code is platform-independent. The operating system consists of the following components:

- The bootloader
- The BSP (Board Support Package)
- The kernel
- The root file system (Linux only)
- The tool chain

Source code for all Linux modules can be obtained upon request.

Section 2: Brutus Functional Description

High-Level Design Goals

Brutus is designed to:

- integrate the many features of the set-top reference platforms into a cohesive application.
- be configurable to support many of the usage modes of the set-top reference platforms.
- have a well-designed graphical user interface.
- provide intuitive use of input devices like IR remotes.
- support all Broadcom set-top reference platforms, including both satellite and cable systems.
- run on multiple operating systems, such as Linux and VxWorks.
- run on both disk-based and diskless systems.

Brutus Architecture

The Brutus application consists of several modules, shown in Figure 2. The only interface to any chip is through the Settop API. These modules are described below.

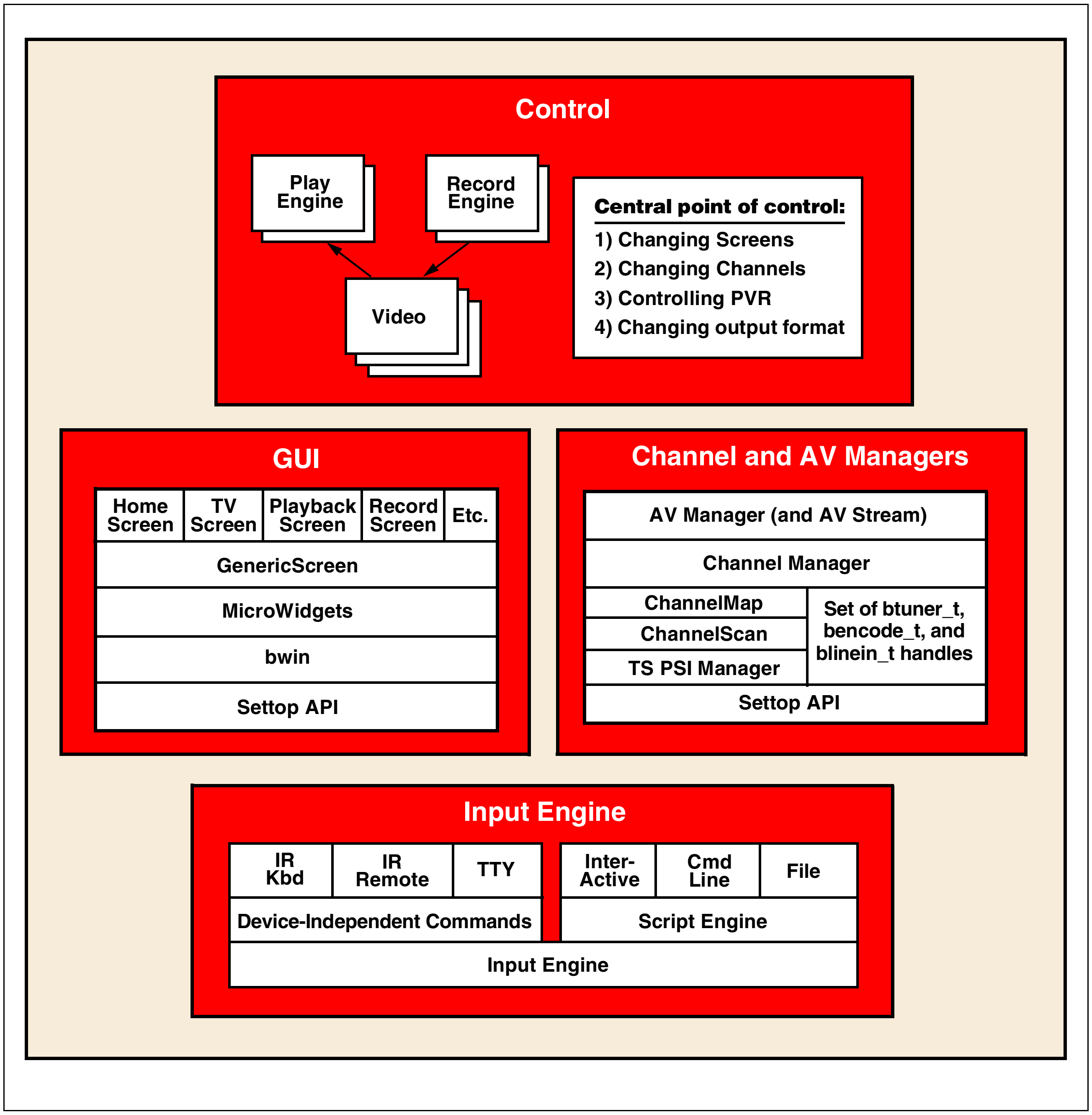

Figure 2: Brutus Component Modules

The Control Module

The Control class coordinates the various components of Brutus. It keeps a list of screens and facilitates changing screens and resizing decode windows. It manages a set of Play and Record engines for doing PVR. It manages channel changes, including incremental channel digit entry, and deferral of the start of decode in order to speed up repeated channel changes. It also manages the change of display formats, which involves rescaling graphics and possibly restarting the decode.

Some of the engines have callbacks directly into the user interface. For instance, PlayEngine can notify the PvrTimeline widget that its state or position has changed. The Control module is responsible for setting up these connections.

The Graphical User Interface Module

Brutus' user interface is built on top of a series of software layers. These layers include:

- Set-Top API Graphics, the part of the Settop API that provides frame buffer functionality
- bwin, a minimal windowing system with font and image rendering support
  - bwin is written in C.
  - The source code is found in BSEAV/lib/bwin.
- Microwidgets
  - This is a TV-friendly widget library
  - It is written in C++.
  - It is modeled after Trolltech's Qt library.
  - The source code is found in BSEAV/lib/mlibs.

The UI follows a single-threaded, cooperative, multitasking model. This means that most application work is performed during "idle time," when the UI is done painting. Idle-time processing must usually be done in short intervals (no blocking); otherwise the UI response time is slow.

Channel Manager and AV Manager Module

The Channel Manager takes a channel map as an input. The channel map is a file that lists frequencies, symbol rate and other information needed to obtain channels. By default, this file is channels.txt.

The Channel Manager scans the listed frequencies for PSI information. It uses the tspsimgr library to parse the PSI information and build a channel list. It makes this channel list available to Brutus. The Channel Manager also keeps the set of tuners which can be checked out for exclusive use.

The AV Manager wraps the Channel Manager and adds the concept of an AV Stream. The AV Stream encapsulates everything needed to decode or record a stream, including bstream_t (analog and/or digital), btuner_t, bencode_t, blinein_t, and the channel number currently on the stream.

THE INPUT ENGINE MODULE

Brutus manages a variety of input devices. See Section 7: “Brutus Input Devices” on page 20 for details. The devices are plugged into bwin’s event loop so that Brutus can multiplex the I/O and need not create a thread per input device. Each input device’s commands must be translated to a set of device-independent commands.

The input engine also manages Brutus scripting. There are three scripting modes available:

- **Interactive mode**: Use the `./brutus --exec` syntax, and type in scripting commands at a prompt. Type “help” to see the available commands.
- **Script file**: Load and execute script files stored on the file system. You can specify the files on the command line or select from an on-screen list through the Admin screen.
- **Command line text**: Specify script commands to run on the command line. This is helpful for testing or if you want to come up in a specific initial state.

OTHER OPTIONAL FEATURES

The following features have special uses and do not appear in Figure 2 on page 5:

- **Cheops**: a Broadcom-proprietary media protocol used for streaming media and Metadata over Ethernet and wireless connections
- **XML and HTTP libraries**: Small, efficient, Broadcom-proprietary libraries used by the Cheops protocol

Section 3: Startup Options

The Brutus application can be started with optional parameters that alter the way it runs. These should be supplied on the command line when starting the application. For example:

```
$ settop brutus --help
```

The following table lists the startup options.

Table 1: Brutus Start-up Options

Option	*Description*
`--help`	Print this help screen.
`--help-cfg`	Print the configuration file options.
`-cfg FILE`	Specify a configuration file (default is brutus.cfg).
`-ch FILE`	Specify a channel map file (default is channels.txt).
`-pg FILE`	Specify a program guide file (default is programguide.txt).
`-pal`	576i (PAL) display mode.
`-admin`	Set admin sticky mode by default.
`-tty`	Use tty instead of Sejin IR keyboard.
`-dbg {err|wrn|msg}`	Set the debug level.
`--exec`	Start in interactive mode.
`-scr SCRIPTFILE`	Run a script file.
`-scrtext SCRIPT`	Run a script with semicolon delimited commands.
`-hd`	Use 1080i display mode.
`-skin SKIN`	Use graphical skin to display user interface (see Section 12: “Brutus Skinning” on page 41 for more details)

Note: These startup options depend on the options used to build the application. This means that not all the options are available for a particular build. Use the `--help` option for a list of the available options for a particular build.

Section 4: Brutus Application Modes

Brutus runs in four main modes. Application modes cannot be changed after starting Brutus, so the mode must be selected before starting.

Single Display Mode

Single Display mode, shown in Figure 3, consists of a single application running with a single user interface. It uses only one compositor, which means it supports only one output format at a time (such as 480i or 1080i, but not both). If an HD mode is selected, the SD video outputs are turned off.

For dual-decode platforms, picture-in-picture (PIP) is available.

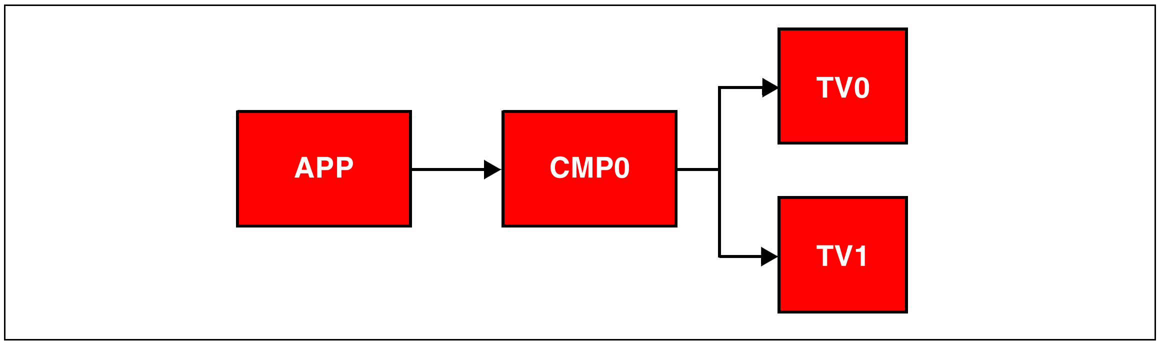

Figure 3: Single Display Mode

In Figure 3, the APP block is one instance of the Brutus application, CMP0 is the first compositor, TV0 is the first TV (also referred to as A), and TV1 is the second TV (also referred to as B).

HD/SD Single Display

Like the previous mode, HD/SD Single Display mode consists of single application running with a single user interface. Internally, however, both the video and graphics that make up the user interface are duplicated on two display pipelines. This means that when the first display is in an HD mode, the second display can still be driving the SD outputs. HD/SD Single Display mode is shown in Figure 4.

To enable HD/SD Single Display mode, add “`HDSD_SINGLE=1`” to the brutus.cfg file.

This mode is supported on the BCM7038, BCM7401, BCM7400, BCM7405, BCM7325, and BCM7335 platforms.

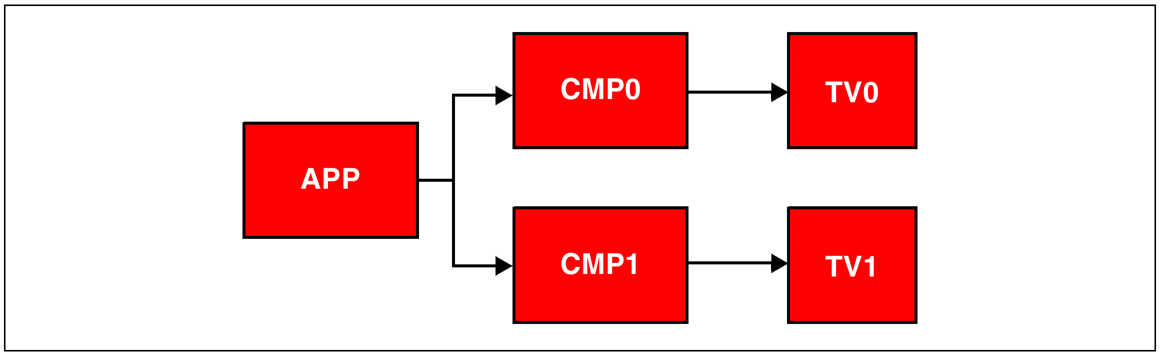

Figure 4: HD/SD Single Display Mode

Dual Display Mode

In Dual Display Mode mode, shown in Figure 5, a single set-top drives two displays as if it were two independent set-top boxes. They share the same front-end resources and hard drive, but have distinct user interfaces and application states.

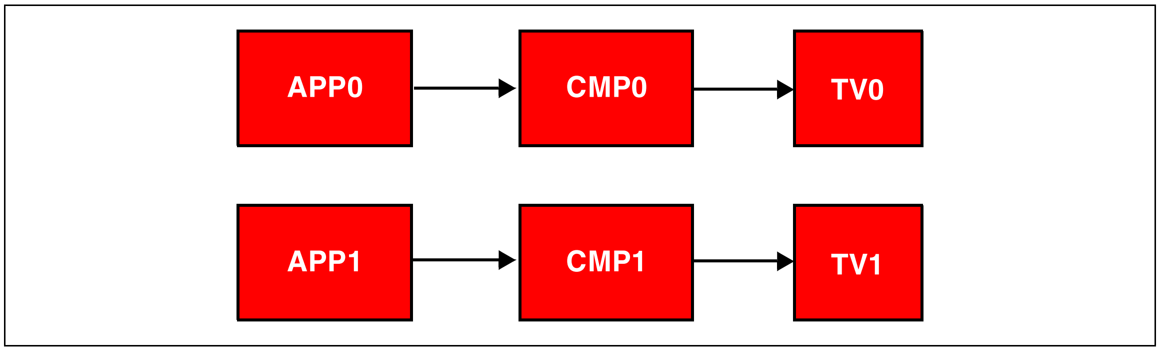

Figure 5: Dual Display Mode

Brutus can accomplish dual display in two ways, depending on how many instances of the API are called. In this mode, there are two instances or Brutus and the Settop API. In Single Application, Dual Display mode, described below, there is only one.

Single Application, Dual Display

In Single Application, Dual Display mode, one copy of Brutus is started, which starts one instance of the Settop API. Then Brutus creates two instances of the main Control module and it runs two instances of the application logic. Minimal communication between instances is accomplished with synchronization between threads.

To enable Single Application, Dual Display mode, add “DUAL_OUTPUT=1” to the brutus.cfg file.

This is the only way to perform dual output with the user-mode driver architecture. This mode also supports the kernel-mode driver architecture. It is the only way to run dual output on a Magnum platform, such as the BCM7038 and BCM7400 platforms.

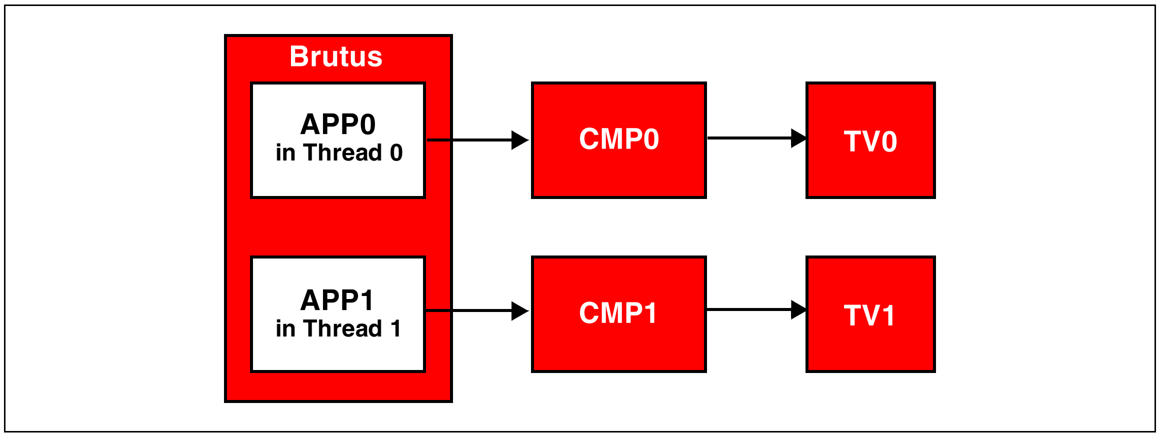

Figure 6: Single Application, Dual Display Mode

Dual Display Input Devices

When running in dual display mode (either single or dual application), you have to assign the inputs to the application instances. By default, the following assignments are made:

Table 2: Inputs to the Two Single Application/Dual Display Mode Instances

APP0	*APP1*
GI remote (**CBL** button on One-For-All remote)	SA remote (**SAT** button on One-For-All remote)
Front panel	No front panel
In single application, dual display mode, `--exec` defaults to app0. Use “`app(#)`” to switch between applications.	In dual application, dual display mode, both applications can have the `--exec` interface.
In single application, dual display mode, `-tty` defaults to app0. Use “`a`” to switch between applications.	In dual application, dual display mode, both applications can have the `-tty` interface.

Triple Display (HD/SD Single + Dual Output)

In Triple Display mode, a single set-top drives three displays as if it were two independent set-top boxes. They share the same front-end resources and hard drive, but have distinct user interfaces and application states. This mode is only supported on BCM7400 platforms.

To run this mode, add both “`HDSD_SINGLE=1`” and “`DUAL_OUTPUT=1`” to brutus.cfg.

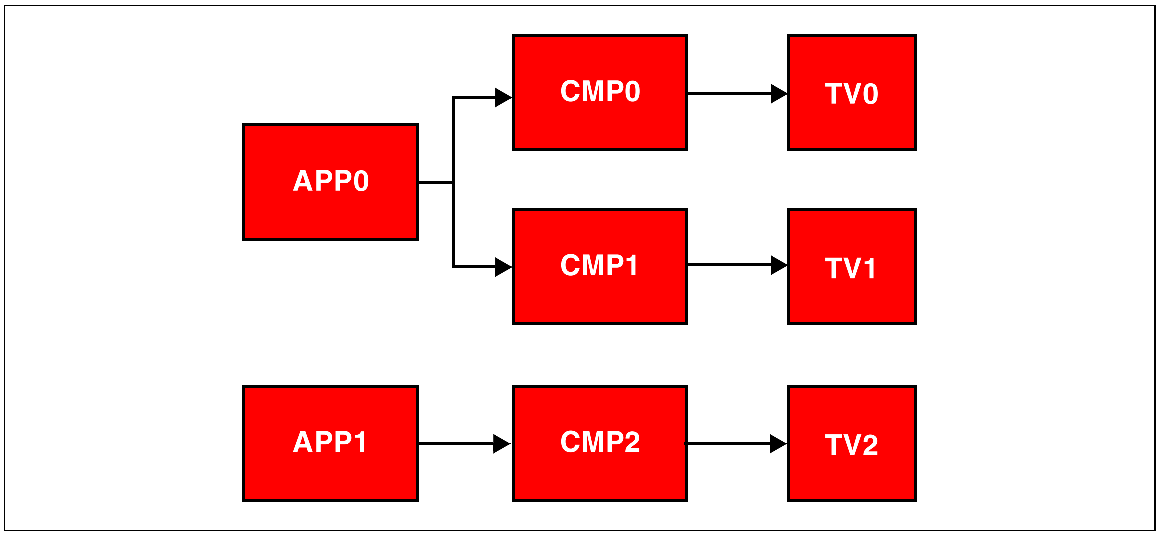

Figure 7: Triple Display (HD/SD Single + Dual Output) Mode

Switching between PIP and Dual Output modes

For dual-output systems, you can toggle between single display and dual display modes using the Admin screen and an external script. The Brutus application does not actually switch modes. The application must exit and restart using the script.

On the Admin screen, there is a **PIP Mode** or **Dual Mode** button. The text indicates the mode you will enter if you click the button. Pressing this button causes Brutus to exit with a special return code. A special shell script must capture this return code and then restart Brutus in the other mode.

A sample bash script can be found in the BSEAV/app/brutus/samples/brutus_7320_wrapper directory. It uses brutus.cfg and creates a pip.cfg file. This is only sample code and may have to be modified for specific environments.

Section 5: Brutus Front-End Architecture

The Channel Map and Channel List

Brutus uses a text file called the channel map (channels.txt) to specify frequencies and symbol rates. Each entry in the channel map is a channel number. For digital channels, the PSI data is scanned and if programs are found, they are included in the channel list as subchannels (i.e., 10.1, 10.2, 10.3, etc.).

There is no initial scan. Channels are scanned when they are tuned.

Tuners

Terminology Note: From a hardware or porting interface point of view, the "tuner" is an RF tuning device and the demodulator is treated separately. From a Settop API or Brutus point of view, the "tuner" is both the RF tuner and its associated demodulator.

Brutus collects all tuners into a global Channel Manager. Tuners are checked out for exclusive use whenever a channel is decoded and/or recorded. When you are done decoding and recording, the tuner is checked back in. When you are in playback mode, there is no tuner needed because a live channel is not being decoded. When playback is stopped and live decoding is resumed again, Brutus tries to acquire another tuner.

This dynamic tuner assignment allows single-display and dual-display systems to make full use of the tuner resources without hard-coding which tuner must be used for a certain display.

If the platform has a non-orthogonal set of tuners, however (that is, dual display system with only one SDS tuner), you can scan SDS channels and have them in the channel list, but if you are decoding or recording an SDS channel on one display, the other display will have no tuner available. In this case, the GUI reports a "(no tuner)" message alongside the current channel number. If you stop record or decode on the first display, the tuner will become available. The other display can start decoding or recording the channel.

Background Recording

If there are available tuners and record devices, you can easily start multiple background record sessions. A "background" record session is a record session that is not currently being decoded. Currently, you can only start recording a channel which you are currently decoding (because the GUI does not allow selecting a non-current channel to record). However, after you start recording, if you change the channel and if another tuner is available, the record session will not be stopped. It will simply be moved into the background. Another tuner will be checked out to start the live decode on the new channel.

You can move the record session back into the foreground by simply channel-changing back to the channel you are recording. This works because Brutus always prefers to decode something that is already being recorded, as opposed to using two tuners to record and decode the same content.

On platforms like the BCM97038 and BCM97400, this technique can be performed twice, which starts triple record. You can see the number of record sessions by looking at the red circles on the TV screen or on the Playback screen.

Section 6: Brutus On-Screen User Interface

Navigating the GUI

Please refer to Section 7: “Brutus Input Devices” on page 20 for options on controlling the graphical user interface (GUI). This manual assumes that you have a remote control.

Watching Live TV

Before starting Brutus, you must set your channel map to match what your head-end is broadcasting. See Section 8: “Brutus Channel Map” on page 26 for details. When Brutus is started, it performs a channel scan and builds a channel list. Analog channels are not scanned and will always appear in the channel list (whether they are there or not). Digital channels generally must be scanned.

Press the **Exit** button to go to full-screen TV. Use the channel +/– keys to change channels.

Press the **Info** button to see the front-end and back-end status. When viewing these status panels, you can also use the Right and Left arrow buttons to see addition status. Press the **Info** button again to toggle the panel off.

Press the **Menu** button to go to the Brutus home page. Press the arrow buttons to move to the TV window and press the **Select** button to go back to full-screen TV.

PVR Recording

Tune to the desired TV channel. Press the **Record** button once to go to a record screen. This allows you to set various options including:

- Recording name that will appear on the Playback screen[1]
- Time-shifting checkbox
- DES/3DES/AES encryption
- Encoding quality for analog channels

Press the **Record** button again (or navigate to the **Record** button on the lower-right corner of the screen) to start recording.

On the TV screen, you'll see a red circle in the lower left corner for each active recording session.

1. Recording name changes can be made with an IR keyboard or with the `--exec` interface.

For platforms that support dual- or triple-record, you see two or three red circles, depending on the number of record sessions that are active. Note that pressing **Stop** on a full-screen TV does the following in this order:

- If currently in playback, it will stop the current playback. If in PIP mode, it will not stop the other playback.
- If not in playback but recording, it will stop one of the recordings.

You can also stop a specific record by pressing the **Guide** button, which will take you to the playback screen. The current recording should be at the top of the list with a red circle by it. You can press the **Stop** button here to stop that recording.

TIME-SHIFTING AND CONTINUOUS RECORD

Any playback of a file which is currently being recorded is called "time-shifting." The user is delaying the viewing of live content. Brutus supports two modes of time-shifting: continuous record and linear.

Continuous record means that Brutus records the live channel into a circular buffer. This is enabled by adding the following option to brutus.cfg:

```
CONTINUOUS_RECORD_ENABLED=y
```

The duration of this file can be set in terms of time:

```
CONTINUOUS_RECORD_DURATION=<seconds>
```

If the continuous record option is not enabled, a **Timeshifting** checkbox is presented on the **Record** screen. This is the simplistic "linear" form of time-shifting. It simply starts a playback of whatever you are recording. The recorded file is not a circular, fixed-size buffer; it simply continues until the disk is filled.

PVR PLAYBACK

Press the **Guide** button on the remote to go to the **Playback** screen. You see a list of videos that have been recorded. If the list is empty, you will need to record something first.[2]

Navigate to the video you want to play and press the **Select** button. Now you'll see detailed information about that video. Press the **Play** button to start playback. You then see the video being played full-screen with a timeline below.

As a shortcut, you can press the **Play** button while on the Playback screen to play the video without seeing the details screen.

On the TV Screen, press **Stop** to stop playback and return to live TV.

2. It is also possible to import a file into Brutus. You must create a `*.info file`. See Section 10: "Brutus PVR *.info files" on page 34 for more information.

TIMELINE GUI

During playback, you see a PVR timeline when viewing full-screen TV. If you are playing a static file (that is, not time-shifting) the timeline will be green. If you are time-shifting, either continuous or linear, the timeline will be red.

For static playback, the width of the timeline represents the entire duration of the file.

For linear time-shifting, the width of the timeline can be set with `RECORD_TIME_WINDOW=seconds`.

For continuous time-shifting, the width of the timeline represents the CONTINUOUS_RECORD_DURATION and the live decode is always represented at the rightmost edge of the timeline. The distance between the cursor and the rightmost edge is the amount of time that playback is behind live decode.

BEGINNING/END OF FILE CONTROL

You can control what action Brutus takes when playback hits the end or beginning of a file with the following options:

```
PLAYBACK_AT_BOF={play|loop|pause}
PLAYBACK_AT_EOF={play|loop|pause}
PLAYBACK_AT_BOF_WITH_RECORD={play|loop|pause}
PLAYBACK_AT_EOF_WITH_RECORD={play|loop|pause}
```

The `_WITH_RECORD` variants apply when you are doing time-shifting, either with a liner or continuous record.

The options are:

- `play`: Switch to normal play in the current location (an exception is slow motion will remain slow motion).
- `loop`: Loop to the opposite end of the file and continue the current mode.
- `pause`: Pause playback wherever you are.

The `loop` option when recording is not naturally expected, but can be useful for testing.

PVR TRICK MODES

After playback has been started, trick modes can be performed using the VCR buttons on your remote. The trick mode buttons will increment or decrement the speed. For instance, if you press the rewind button twice to see the second rewind speed, and then press the fast-forward button, you will return to the first rewind speed.

Other controls include:

- **Play** will resume normal playback from either pause or a trick mode.
- **Pause** will pause either normal playback or a trick mode.
- Pressing **Pause** again when in a paused state will resume normal play.
- Right arrow will jump forward 30 seconds
- Left arrow will jump backward 5 seconds
- When paused, **Fast Forward** functions as “frame advance” and **Rewind** functions as “frame reverse.”

Using Picture-in-Picture

For dual decode systems running in single display mode, Brutus supports picture-in-picture (PIP) mode. See Section 4: "Brutus Application Modes" on page 9 for more details.

Navigate to full-screen TV by pressing the **Exit** button. Then press the **PIP** button. If PIP is supported, you see a box appear in the upper right corner surrounded by a yellow outline. The yellow outline means that PIP has the control. All commands (like channel +/– or PVR) will apply to PIP and not to Main when the yellow box is visible.

Press the **Move** button and the yellow box will disappear. This means that Main has the control, but PIP is still visible.

Press the **Swap** button and the content of Main and PIP will be swapped. Press the **Swap** button again to restore the original Main and PIP.

Playing MP3s

If Brutus was compiled with `AUDIO_SUPPORT=y` and `MP3_ENABLED=1` is in the configuration file, then you should see an **MP3s** button on the Brutus home page. This is the default for most platforms.

Press **Menu** to see the home page. Press the **MP3s** button, and you see a list of files in the "audio" subdirectory that can be played. You can refresh the list from the admin page if you copy more files after starting Brutus.

Click on a song title (or press the VCR **Play** button) to start playing. Press the VCR **Stop** button to stop playing.

Note: BCM7411 and BCM740x platforms support hardware MP3 decode. The Brutus MP3 screen still requires a software MP3 decoder, however. This is used to test PCM playback and to add CPU stress. You can do a hardware decode of an MP3 using the playback test application. It is included in the default installation binary tarball. Run it as follows:

```
settop playback -mpeg_type ES -audio 1 -audio_type 0x1 file.mp3
```

Viewing Pictures

If Brutus was compiled with `PICTURES_SUPPORT=y`, then you should see a **My Pictures** button on the Brutus home page.

Press the **My Pictures** button, and you see a list of JPEGs, PNGs and BMPs that can be viewed. If the view is empty, you'll need to copy some images into the pictures/ subdirectory and restart Brutus.

Some Broadcom chips support still-picture decode in hardware. This has not been integrated with this Brutus feature. You can use the feature with the Settop API test application BSEAV/api/tests/decode_still_picture.

Running a Script

You can control all Brutus functions using a simple scripting language. The easiest way to learn about it is to start Brutus with the `--exec` command line option, type **help** to see your choices, and then enter script commands interactively. Please see Section 11: “Brutus Scripting” on page 37 for more details.

Skinning the Graphical User Interface

You can modify the look and feel of the Brutus graphical user interface using “skins.” Start Brutus with the `-skin blue` command-line option to change its visual presentation based on the sample “blue” skin. Please see Section 12: “Brutus Skinning” on page 41 for more details.

Changing Video Outputs

You can control your display settings from the Info panel. Press the **Info** button on the remote, or press the **Info** button on the main menu. Navigate to the **Display** button, and then arrow over to the control panel.

The controls that you see reflect the options available for the current video format. Use the right and left arrows to change the arrow bars. Press the **Select** button on the remote to enable the change. You see an immediate change. Also notice that the list of available controls may change as the video formats are switched.

Editing the Channel Map Onscreen

The default channel map comes from the channels.txt file. You may find it easier to edit using a text editor. You can also edit the channel map onscreen using a remote and then overwrite that channel map.

From the main menu, click on **Admin**. If `ADMIN=y` has not been set in the configuration file, the password is 8822. Click on **Edit Channels**. You can then perform the following operations:

- To add a channel, click on **Add**, enter your settings, then click **OK**.
- To remove a channel, click on the channel in the list, then click **Delete**.
- To edit a channel, click on the channel in the list, make your changes, then click **OK**.
- To save your changes, click on **Save**.
- To abandon your changes, click **Reset**.

RENAMING VIDEO STREAMS

When a stream is recorded, Brutus will use the entry in programguide.txt. If there is no entry, Brutus will use a text string that describes the source (QAM, ANALOG, etc.)

After a stream has been recorded, it is often helpful to rename it. From the GUI, go to the Playback list, select the stream, and you will find a **Rename** button. Select the button. Here is a list of methods to change the name.

If you are using a Sejin IR keyboard, you can type in directly.

- If you are using `--exec`, you can use the entertext(XXXX) command.
- If you are using `-tty`, you can use the ";" command to enter text.
- You can exit Brutus and edit the *.info file and change the name field.
- If you are using a remote, your only option is to enter numbers; otherwise you need to use one of the above methods.

Section 7: Brutus Input Devices

There are several ways of controlling Brutus. Here is an overview:

1. Using a front panel
   a. IR remote
   b. IR keyboard
   c. Keypad
2. Text control from a console
   a. Script control
      `-scr` (from file)
      `-scrtext` (from command line)
   b. Interactive
      `--exec` (interactive script commands)
      `-tty` (single key command)

The easiest is to use is the remote control. But in case you do not have a front panel, the other methods provide full control, only not as intuitively.

This section presents is a detailed overview of each input device

ONE FOR ALL REMOTE

The reference platform should include a “One For All” remote that has been programmed with GI and SA remote codes. If you need a remote, please request one from Broadcom’s System Operations department. These remotes are programmed using a computer application with a wired connection to each remote. It is very difficult to manually program every button correctly on these remotes.

Press the **CBL** button located at the top of the remote to use GI codes. These work for all platforms. On dual-output systems, this will control the first output.

Press the **SAT** button located at the top of the remote to use SA codes. For dual-output systems, this will control the second output.

If your remote does not work, do the following:

1. Press the **CBL** button at the top of the remote (or **SAT** if using second output).
2. Change the batteries.
3. Make sure the front panel is connected; check whether the LEDs are lit up.
4. Try the remote with another good set-top box, and/or try another good remote with this set-top box.
5. If nothing works, contact Broadcom’s System Operations department for a new remote.

The remote control key map is given in Table 3:

Table 3: Remote Control Key Map

Key	*Function*
PIP	Toggle pip on and off (lower left)
MOVE	Toggle pip control (lower)
SWAP	Toggle video between main and pip (lower)
INFO	Toggle the info panel on the full-screen TV screen (upper right)
MENU	Toggle the main menu (upper left)
GUIDE	Toggle the playback screen (upper left)
POWER	Enter Standby mode (if supported)
MUTE	Toggle volume mute
0…9	Change the channel
REW	Fast rewind
FWD	Fast forward
F.REW	Slow rewind
F.FWD	Slow forward
REC	Press once to bring up record screen, press again to start recording.
PAUSE	Pause PVR. Only works on full-screen TV.
PLAY	Resume PVR play. Only works on full-screen TV.
STOP	Stop PVR. Only works on full-screen TV.
VOL +/-	Control volume (this will un-mute)
CH +/-	Change channel (this will stop PVR)

SEJIN IR KEYBOARD

This wireless keyboard can be requested from Broadcom's System Operations department. It is not included by default. The only reason a keyboard may be needed for Brutus is to enter PVR recording names (otherwise you must edit programguide.txt. See Section 8: "Brutus Channel Map" on page 26 for details.). The Sejin IR keyboard key mappings are given in Table 4.

Table 4: Sejin IR Keyboard Mapping

Key	***Function***
F1	Stop
F2	Pause
F3	Play
F4	Slow motion
F5	Rewind
F6	Fast forward
F7	Record
F9	Show tv
F10	Show menu
F11	Show guide
F12	Show info
PrtSc/SysRq	Toggle video
ScrollLock	Toggle PIP control
-	Channel down
+	Channel up
[	Volume down
]	Volume up
\	Mute
Popup Menu	Toggle PIP

FRONT PANEL KEYPAD

In order to make the front panel as fully functional as possible, the meaning of some buttons on the keypad is context-sensitive.

FULL-SCREEN TV SCREEN

When you are on full-screen TV, the buttons will behave differently if PVR playback or record is active, and if the Info panel is visible. The best way to learn these is to try them. Table 5 and Table 6 provide some documentation of these key mappings.

Table 5: Full-Screen TV Screen Key Mappings

Key	*Function*
Power/Reset	Toggle SD/HD output (BCM937XX) or NTSC/PAL (all other platforms)
Left Arrow	If the info panel is visible, navigate left. If PVR playback is active, rewind.
Right Arrow	If the info panel is visible, navigate right. If PVR playback is active, fast forward.
Up Arrow	If PVR playback is active, pause; otherwise, channel up.
Down Arrow	If PVR playback or record is active, stop; otherwise, channel down.
Sel	If PVR playback is active and you are in a trick mode, then resume normal play. Otherwise toggle the info panel.
Guide	Toggle Playback screen and TV screen.
Menu	Toggle Home screen and TV screen. On the BCM97320, the **Info** button functions as the **Menu** button. To show the info panel, go to the TV screen and press **Sel**.
Ch +	Channel up
Ch –	Channel down
Vol +	Volume up
Vol –	Volume down

All Other Screens

Table 6: All Other TV Screen Key Mappings

Key	*Function*
Power	Toggle SD/HD output (937XX) or NTSC/PAL (97xxx)
Left Arrow	Navigate Left
Right Arrow	Navigate Right
Up Arrow	Navigate Up
Down Arrow	Navigate Down
Sel	Enter
Guide	Toggle Playback screen and TV screen
Menu	Toggle Home screen and TV screen
Ch +	Channel up
Ch –	Channel down
Vol +	Volume up
Vol –	Volume down

Text Control: Script Control

This uses the `settop brutus--exec` command and provides a `Brutus>` prompt from which commands can be entered. The commands are identical with the scripting language commands, which are documented in Section 11: "Brutus Scripting" on page 37.

TEXT CONTROL: SINGLE KEY COMMAND

This mode allows you to control Brutus with single key presses. You can only use this option if you start Brutus from a command line, using the serial console or telnet. Start Brutus with the `settop brutus -tty` option. Press **?** to see the key mapping given in Table 7.

Brutus will open the console in raw mode, which allows it to process individual keystrokes. This can be hard to learn at first, but is very fast.

Table 7: Text Control Key Mapping

Key	***Function***
w	Go to TV Screen (Exit)
e	Go to Playback Screen
t	Go to Home screen
z	Stop PVR
x	Pause PVR
c	Play PVR
r	Go to Record Screen
n	Fast forward, frame adv when paused
b	Fast rewind, frame rev when paused
.	Slow forward
,	Slow rewind
- +	Channel down/up
p	Toggle pip
m	Catch up time-shifting PVR to record
o	Swap main and PIP
i	Toggle focus between main and PIP
[ ]	Volume down/up
\	Mute audio
hjkl	Arrow left, down, up, right (some contexts)
tab	Tab to next control
`	Reverse tab to previous control
a	Switch between application instances in DUAL_OUTPUT modes.

Section 8: Brutus Channel Map

The channel map tells Brutus which frequencies and modulation types to include in its list of channels. It supports multiple line inputs, analog RF, QAM, QPSK, VSB, OFDM, and IP channels, depending on the platform.

For line-in and analog, each channel file entry corresponds to one channel in the user interface. For digital channels (either QAM or QPSK), the channel may have zero or more programs. These are displayed as a single channel with subchannels.

The default channel file is channels.txt. You can specify your own channel file using the `-ch` command line parameter.

There is also a primitive way to specify program guide information. Edit programguide.txt and add channel names. These will be used when displaying the current live channel and in the recorded stream name. Each entry in the program guide corresponds to one main channel. Subchannels all share the same name.

CHANNEL MAP SYNTAX

Table 8: Channel Map Syntax

Type	*Parameter*	*Parameter*	*Parameter*	*Parameter*
LINEINPUT	index	PAL or NTSC	–	–
ANALOG	Disabled?	Frequency	PAL or NTSC	–
QAM64	Disabled?	Frequency	Symbol rate	annex
QAM256	Disabled?	Frequency	Symbol rate	annex
QAM1024	Disabled?	Frequency	Symbol rate	annex
QPSK_DVB	Disabled?	Frequency	Symbol rate	SDS Options
QPSK_DSS	Disabled?	Frequency	Symbol rate	SDS Options
QPSK_DSS_PES	Disabled?	Frequency	Symbol rate	SDS Options
QPSK_TURBO	Disabled?	Frequency	Symbol rate	SDS Options
QPSK_LDPC	Disabled?	Frequency	Symbol rate	SDS Options
8PSK	Disabled?	Frequency	Symbol rate	SDS Options
STREAMER	Disabled?	Streamer #	Transport type	–
VSB8	Disabled?	Frequency	Symbol rate	–
OFDM	Disabled?	Frequency	Bandwidth	OFDM Options
VSB16	Disabled?	Frequency	Symbol rate	–
IP_UDP	Disabled?	IP Address	UDP port	–
IP_RTP[1]	Disabled?	IP Address	UDP port	–
IP_RTP	Disabled?	URL[2]	–	–

1 See Section : “Support for RTP and RTCP Protocols” on page 31 on support for RTP protocol
2 See Section : “Support for RTP and RTCP Protocols” on page 31 on support for RTSP protocol

Be aware that for the BCM4500, 8PSK is always used as “Turbo 8PSK.” For the BCM4501, 8PSK is always used as “LDPC 8PSK.”

If a channel is marked as “Disabled” (set to 1), then channel up and down will skip over it. You can still select the channel using the digits on the remote.

SDS OPTIONS

For the SDS (satellite downstream) modulation types, there are a variety of options which can be listed after the symbol rate. They are not positional. These are given in Table 9:

Table 9: Satellite Downstream Modulation Channel Map Options

SDS Option	*Description*
13v	Set DISEQC to 13 volts
18v	Set DISEQC to 18 volts
#/#	Select code rate #/#. Otherwise “scan” mode is used.
LDPC_PILOT	For LDPC modulation types, this enables LDPC_PILOT. Otherwise it defaults off.
LDPC_PILOT_PLL	If LDPC_PILOT is enabled, this also enables the LDPC PILOT PLL.
ToneOn	Enable DISEQC test tone. Otherwise it is off.

OFDM OPTIONS

For the OFDM (DVB-T/Terrestrial Handheld) modulation types, there are a variety of options which can be listed after the Bandwidth. They are given inTable 10.

Table 10: OFDM Channel Map Options

SDS Option	*Description*
off	CCI Off (default)
auto	CCI Auto select
high	Tune to High Priority Streams (default)
low	Tune to Low Priority streams

The default options for OFDM are 8 MHz bandwidth, CCI off, and high priority streams.

SAMPLE CHANNEL MAP ENTRIES

```
# Line inputs
LINEINPUT       0   # 1st lineinput, e.g. composite
LINEINPUT       1   # 2nd lineinput, e.g. svideo
LINEINPUT       2
LINEINPUT       3

# Analog RF
ANALOG          0       63.25
ANALOG          0       409.25  PAL
ANALOG          0       411.25  NTSC

# Digital Cable
QAM64           0       765.0
QAM64           0       500.0   DEFAULT  AnnexB
QAM64           0       500.0   DEFAULT  AnnexA
QAM256          0       777.0   5.3069
QAM1024         0       789.0

# OFDM/Tererstarial Hendheld
OFDM            0       327.0  DEFAULT
OFDM            0       527.0   8        off       high
OFDM            0       527.0   5        auto      low

# Satellite
QPSK_DVB        0       1119.0  20.0    13V 3/4
QPSK_LDPC       0       1119.0  30.0    2/3 LDPC_PILOT

# Direct TV ES stream (must specify PROGRAM PIDS)
QPSK_DSS        0       1382.0  20.0    18V   ToneOff
PROGRAM PIDS 0x64 0x64 0x65 0x3

# Direct TV PES stream (must specify PROGRAM PIDS)
QPSK_DSS_PES    0       1382.0  20.0    18V   ToneOff
PROGRAM PIDS 0x64 0x64 0x65 0x3

# Streamer input (board muxes will be set appropriately)
STREAMER        0       0   TS      # TS1_IN with MPEG2 Transport
STREAMER        0       1   TS      # TS2_IN with MPEG2 Transport
STREAMER        0       0   DSS_ES  # TS1_IN with Direct TV ES stream
STREAMER        0       1   DSS_PES # TS2_IN with Direct TV PES stream

# IP channels
IP_UDP          0  224.1.1.10  1234
IP_UDP          0  224.1.1.11  1234
# optionally specify PIDs
IP_UDP          0  224.1.1.12  1234
PROGRAM PIDS 0x1023 0x1023 0x1022 0x3 video_type=0x2
# an RTSP “channel”
IP_RTP          0 rtsp://192.168.0.38:554/Video/MyVTR/mystream.ts
# an RTP channel
IP_RTP          0  224.1.1.16  1234
```

CHANNEL NUMBERS

Each line in the channel map corresponds to one channel in the Brutus UI. Comment lines and blank lines are not counted. The first Brutus channel is 0. LINEINPUTS are given channel numbers.

BYPASSING CHANNEL SCAN BY SPECIFYING PIDS

If scanning PSI information is not desired, you can specify the PIDs immediately after each channel map entry. Here are a few examples:

```
QAM64           0       500.0
PROGRAM PIDS video=0x11 audio=0x14 audio_type=AC3
PROGRAM PIDS video=0x21 pcr=0x22

QAM64           0       765.0
PROGRAM PIDS video=0x11 audio=0x14 audio_type=MPEG2

STREAMER 0 0
PROGRAM PIDS video=0x11 video_type=AVC
```

The syntax of PROGRAM PIDS is:

```
PROGRAM PIDS pcr_pid video_pid audio_pid audio_type video_type
```

If only `pcr_pid` is specified, it is used for `video_pid` and `pcr_pid`.

`video_type` values are: `MPEG2`, `AVC`, `VC1`, etc. (see Table 13 on page 36). The default is `MPEG2`. In addition, `MPEG` is accepted as `MPEG2`.

`audio_type` values are: `MPEG`, `AC3`, `AAC`, etc. (see Table 12 on page 36). These values are the same as `audtype` in the *.info file.

You can also use the numeric Settop API values as well. If `pcr` is not specified, it defaults to video.

Using Streamers With Dual-Decode Platforms

Brutus does not allow two decodes from the same band, which was a design decision. It could be done if the application monitored PID usage, but Broadcom elected to enforce this at a band level.

For a single-decode system, it is relatively simple. STREAMER 0 in the channel map is TS1_IN, STREAMER 1 is TS2_IN. It all works with no problems.

For a dual decode system (either main/pip or dual output), it is more complicated and there are options.

- Connect a single streamer to TS1_IN and set `export duplicate_streamer=yes` before running Brutus. This causes the FPGA to duplicate the TS1_IN data to two bands, as if you have an identical streamer connected to TS2_IN. Only add STREAMER 0 into the channel map. App0 will use Settop API streamer id 0, App1 will use Settop API streamer id 1.

  Note: This only works in Linux user mode because `env` variables are not supported in kernel mode. For VxWorks, you can call `bsettop_set_config(name, value)`.

- Connect two streamers to TS1_IN and TS2_IN and do not set `duplicate_streamer`. Only add STREAMER 0 into the channel map. APP0 will use Settop API streamer id 0, APP1 will use Settop API streamer id 1. Be aware that if you use PROGRAM PIDS to specify PIDs in the channel map, you will need identical PIDs on both streamers.
- Under no circumstances should you specify STREAMER 1 in the channel map when running dual decode; otherwise problems will occur.

BT.656/I^{2}S Input Support

On some platforms, BT.656 video and I^{2}S audio external analog inputs are supported in both the Settop API and Brutus.

In the Settop API, it varies by platform.

- Settop API BT.656 input support is defaulted on for the BCM7038.
- Settop API BT.656 and I^{2}S input support defaults to off for BCM7401 platforms. To enable it, uncomment the `#define B_HAS_EXTERNAL_ANALOG` line found in BSEAV/api/src/magnum/board/bsettop_bsp_7401.h.
- Settop API BT.656 and I^{2}S input support defaults to off for BCM7400 platforms. To enable, uncomment the `#define B_HAS_EXTERNAL_ANALOG` line found in BSEAV/api/src/magnum/board/bsettop_bsp_7400.h.

In all cases, external analog input is available as `btuner_open(B_ID(7))`. See BSEAV/api/examples/tune656input.c for an example application.

This feature can be tested using the `decode` utility (from BSEAV/api/utils/decode) as follows:

```
settop decode –freq 0 –decode 0 –tuner 7
```

In Brutus, two changes are needed for external analog input support:

- Add `B_HAS_EXTERNAL_ANALOG=yes` to brutus.cfg
- Add an ANALOG line to your channel map. Frequency is ignored. PAL/NTSC is used.

For example:

```
ANALOG 0 0 NTSC    # NTSC 656 input support
ANALOG 0 0 PAL     # PAL 656 input support
```

Note: BCM7038 BT.656 input support has not been added to Brutus.

SUPPORT OF VARIOUS IP NETWORK PROTOCOLS

Rudimentary IP STB functionality is part of the standard Brutus Reference Software build. The channel map tells Brutus which IP addresses (typically multicast IP addresses) and which UDP port numbers have programs that it can tune to. Edit the Brutus channels.txt file appropriately to match the IP head-end/video server configuration.

More advanced features require that the Reference Software be built with Live Media support. Building with Live Media support provides additional capabilities, including support for the RTP, RTCP, RTSP and SAP protocols.

SUPPORT FOR RTP AND RTCP PROTOCOLS

This release may provide basic support for the Real-time Transport Protocol (check the release notes, both for current and previous versions). When configured correctly, the STB is capable of accepting and decoding MPEG-2 SPTS carried in RTP.

The application defaults to a Basic RTP support, which provides the minimum set of functionality required to receive an RTP stream and decode it. As such, Basic RTP support does not support RTCP, and is similar to ProMPEG Code of Practice #3 (Forward Error Correction), but omits the FEC element of that specification. Packet reordering is supported.

The more complete “Standard RTP” support provides processing of the RTP timestamp, along with the SSRC and CSRC (x CC), for use with RTCP. Standard RTP conforms fully to the IETF RFC 3550 and is implemented via the Live Media open-source library (under LGPL). In order to utilize this capability, users must build for Live Media support.

Support for RTSP Protocol

This release of Brutus may provide support for the Real-Time Streaming Protocol (check the release notes, current and previous). Also known as the “Internet VCR” protocol, RTSP is used for the retrieval of media from a Media Server (such as Video on Demand) and provides for setup, start, pause, trick-mode, and stop control of IP streams.

When configured correctly, the STB is capable of “tuning” to a URL via the Brutus channel map. Some restrictions apply:

- URL tuning is limited to RTSP URLs (for now)
- For RTSP, the server offers transport options via SDP, and the client then chooses between them (such as UDP or RTP). Your Media Server may only support RTP or UDP, or may support both.
- Choice of IP_UDP or IP_RTP must match the client/server capabilities (currently the STB only supports RTP when using RTSP)
- You must build with `LIVEMEDIA_SUPPORT=y` to use RTSP.

This means that to begin receiving and decoding an RTSP stream, the user places an entry in channels.txt, starts Brutus and then channel changes (if not the first entry) into the RTSP stream. The URL typically takes a form similar to:

```
rtsp://192.168.0.38:554/Video/MyVTR/mystream.ts
```

A more complete channel map example is contained in Section 8: “Brutus Channel Map” on page 26.

Support for SAP/SDP Protocol

This release provides support for dynamic IP channel acquisition using Session Announcement Protocols (SAP) - RFC 2974 and Session Description Protocols (SDP) – RFC4566.

SAP and SDP protocols are implementing by extending the basic SDP parsing support provided by the Live Media Library. A new library called libblive_ext.a provides SAP/SDP support. Brutus has been modified to dynamically update the channel map as SAP requests are received to add or delete an IP session. Further details on the SAP/SDP design can be found in the SAP-SDP-Design document.

In addition, libblive_ext.a also provides thread-safe access to the Live Media library by implementing a scheduling thread.

Section 9: The Brutus Configuration File

The configuration file contains `NAME=VALUE` pairs that change the default operation of Brutus. The default configuration file is brutus.cfg. You can specify another configuration file using the `-cfg` command line option.

Basic types supported are:

- boolean (Strings beginning with “t,” “T,” “y,” “Y” or numeric non-zero are regarded as true.)
- int
- float
- string (with some special types defined, see below)

Learning the Options

To learn the standard options, run `settop brutus --help-cfg`, which will print each standard option, its default value (which can be platform-specific), its type, and a description. There are a large number of these options.

In this print-out, you will also see some special types, which are just variations on the string format:

- filename
- font (a bwin pre-rendered font file)
- color (ARGB8888 hex value, which must use 0x prefix)

Be aware that the configuration file can support custom options that are not documented, but these are not used often.

Custom Options

Some configuration options are not listed with `settop brutus –help-cfg` because they are custom and/or involve software parsing. These are described in this section.

Specifying Multiple Video Directories for PVR

By default, Brutus scans the videos subdir for *.info files. You can modify this behavior and specify a list of video directories to scan. The syntax is:

```
VIDEODIRx=subdir
VIDEODIRx_LABEL=Name
```

Some examples:

```
VIDEODIR0=avc_files
VIDEODIR0_LABEL=My AVC Files
VIDEODIR1=mpeg_files
VIDEODIR1_LABEL=My MPEG Files
```

If you want the labels visible in the Playback screen, add this option to brutus.cfg:

```
SHOW_VIDEODIR_LABELS=y
```

Section 10: Brutus PVR *.info files

The Playback screen shows a list of PVR files that can be played. This list is populated by scanning the videos/ subdirectory for *.info files. The standard installation comes with sample.info as an example.

The easiest way to generate *.info files is simply to record your own streams. Sometimes, however, you will want to import a stream you recorded elsewhere and make it available for Brutus. The minimum information needed is name, mediafile and the PIDs (vidpid, pcrpid, audpid, audtype). The mediafile and indexfile file names can be absolute; otherwise they are relative to the directory where the *.info file is (not where the Brutus binary is).

MINIMAL EXAMPLE

```
name=The Mummy Returns
mediafile=brutus1.mpg
vidpid=0x11
pcrpid=0x11
audpid=0x14
audtype=AC3
```

FULL EXAMPLE

```
################################
#
# Sample .info file
#

# name shows up in the Playback list
name=The Mummy Returns
description=Recorded on the BCM93740.
mediafile=brutus1.mpg

# indexfile is optional. enables trick modes.
# .nav or .bcm extensions signify a Broadcom-proprietary index.
* Otherwise it is read as an SCT-format indx.
indexfile=brutus1.nav

vidpid=0x11
pcrpid=0x11
audpid=0x14
audtype=MPEG

# sortindex determines sort order on the screen.
# Highest number is on top.
sortindex=0

recordtime=1031651588
resumeindex=141
quality={Digital|HD|HITS|Best|High|Medium|Basic} #default is Digital
format=TS#default is TS. See table below
enc_type={NONE|DES|3DES} #default is none
```

PES STREAM EXAMPLE

```
name=PES Video Stream
mediafile=file.pes
# vidpid is the Stream ID for PES
vidpid=0xE0
format=PES
```

ES STREAM EXAMPLE

```
name=ES Video Stream
mediafile=file.es
# Setting the vidpid to non-zero means that this is a video stream
vidpid=1
format=ES
```

VALUES

Table 11: Values for the format Field

Value for the format field	*Meaning*
TS	MPEG-2 Transport with MPEG-2 Video
PES	MPEG-2 PES with MPEG-2 Video
ES	MPEG-2 Video Elementary Stream
AVC_TS, AVC_PES, AVC_ES	See above, but with AVC/H264/MPEG-4 Part 10 video
VC1_TS, VC1_PES, VC1_ES, VC1_ASF	See above, but with VC-1 video
DIVX_TS, DIVX_PES, DIVX_ES, DIVX_ASF	See above, but with DivX (MPEG-4 Part 2)
DIRECT_TV_ES	Direct TV Transport (DSS) with ES MPEG-2 Video Payload
DIRECT_TV_PES	Direct TV Transport (DSS) with PES Payload with MPEG-2 Video Payload

Table 12: Values for the audtype field

Values for the audtype field	*Value*	*Meaning*
MPEG, MPEG1, MPEG1L2	0x03	MPEG1/2 layer 1/2
MP3, MPEG1L3	0x01	MPEG1/2 layer 3
AC3	0x81	AC3 (aka Dolby Digital, DD)
AC3+, DDP	0x06	AC3 Plus (aka Dolby Digital Plus, DDP)
AAC	0x0F	AAC
AAC+	0x11	AAC Plus (aka AAC-HE, AAC plus SBR) for ADTS streams.
AAC+_LOAS	0x12	AAC Plus for LOAS streams.
DTS	0x82	Digital Theater Systems, Surround sound
LPCM_HDDVD	0x83	LPCM for HD-DVD
LCPM_BLURAY	0x84	LPCM for Blu-ray™
DTSHD	0x85	Digital Theater Systems, Surround sound HD
WMA, WMASTD	0x86	Windows Media Audio, standard format
WMAPRO	0x87	Windows Media Pro Audio, standard format
LPCM_DVD	0x88	LPCM, DVD mode

Table 13: Values for the vidtype field

Values for the vidtype field	*Value*	*Meaning*
MPEG2, MPEG	0x01	MPEG-2 Video (ISO/IEC 13818-2)
MPEG1	0x02	MPEG-1 Video (ISO/IEC 11172-2)
AVC, H264, H.264	0x1B	H.264 (ITU-T) or ISO/IEC 14496-10/MPEG-4 AVC
H263, H.263	0x1A	H.263 Video
VC1, VC-1	0xEA	VC-1 Advanced Profile
VC1SM, VC1-SM, VC-1SM,VC-1-SM	0xEB	VC-1 Simple & Main Profile
Divx3, DivX-3	0x311	DivX 3.11 coded video
MPEG4, MPEG4-2	0x10	MPEG-4 Part 2 Video

Section 11: Brutus Scripting

Brutus has a very simple scripting interface. The easiest way to learn scripting is to use the interactive mode. Start Brutus with the `--exec` command line option. You see a `Brutus>` prompt. Type `help` and press **Enter**.

```
$ settop brutus --exec
Brutus>help
```

Type `quit` to exit Brutus cleanly. You can also press **Ctrl-C** to exit immediately, but it might leave the hardware in a bad state.

BRUTUS SCRIPTING OVERVIEW

A script is a series of input commands and execution commands. Anything you can do with a remote, you can do with a script. There are also some general functions, like play(#), that perform multiple commands.

Multiple commands can be specified on a single input line when separated by semi-colons. A # on a command line indicates the start of a comment (the remaining text is discarded).

The script system does not support conditionals and error-checking. By creating multiple scripts that all end with the `quit` command, you can then create scripts (like bash or perl scripts) that can control Brutus in a variety of configurations.

RUNNING A SCRIPT FILE

You can start a script when Brutus starts using a command line parameter. Here is an example:

```
$ settop brutus -scr scripts/changechannel.txt
```

This causes the scripts/changechannel.txt file to be executed after Brutus comes up. You can also start a script using the GUI. From the main menu, click **Admin**. You may need to know the admin password. Click the **Run Scripts** button. Select a script and click on it. It will start executing.

STOPPING A SCRIPT FILE

Pressing any key from another input device, like a remote control or front panel, stops a script file. The key you press will only be used to stop the script and will not result in its normal action.

COMMAND LINE SCRIPT

You can also specify a short script on the command line. This is helpful for testing without having to pick up remote control all the time. Here's a sample:

```
$ settop brutus -scrtext "play(1);r;r;rew;sleep 2;pause"
```

Interactive and Scripting Commands

Table 14: Interactive and Scripting Commands

Command	*Description*
play(x)	Play the xth PVR file on the playback screen. If x is negative, play the n-xth PVR file where n is the total files available on playback screen.
ch(x)	Do live decode of channel specified by x.
vol(x)	Set the volume level to the value x.
goto (<screen>)	Go to screen specified by <screen>, and set the focus to the upper left corner for consistent navigation. Screens: TV, playback, menu, record, admin, mp3s, pictures
pause	Pause playback.
stop	Stop playback and record.
play	Press the play button. Also, resume normal play mode
record	Show the Record screen.
ff	PVR Fast Forward.
rew	PVR Fast Rewind.
slow	PVR Slow forward.
slowrew	PVR Slow rewind.
chanup	Increment the current channel number by one.
chandown	Decrement the current channel number by one.
pip	Toggle Picture-In-Picture visible.
0-9	Enter digits for channel change.
move	Move focus/control between main and PIP screens.
swap	Swap (audio/video) between main and PIP screens.
volup	Increase the volume level by one volume-scale unit (1-10).
voldown	Decrease the volume level by one volume-scale unit (1-10).
up	Move cursor (focus) up one control.
down	Move cursor (focus) down one control.
left	Move cursor (focus) left one control.
right	Move cursor (focus) right one control.
select	Select control/item with current focus. For buttons, this acts as a button press.
guide	Toggle between the Playback guide screen and TV screen.
menu	Toggle between the Brutus main menu screen and TV screen.
mute	Invoke the mute option (mute-audio) during live or recorded playback
exit	Display full-screen TV. This is the same as “goto(tv).”
info	Toggle the info panel. This option is only available in full-screen operation
pageup	Move focus up one page length.
pagedown	Move focus down one page length.
music	Go to the MP3's screen.
lock	Force playback to catch up to current record.

Table 14: Interactive and Scripting Commands (Cont.)

Command	***Description***
entertext('text')	Enter 'text' into the text box with current focus.
# (pound character)	Comment. All text following the "#" character is ignored by Brutus (up to next new line).
sleep <#>	Sleep "#" of seconds. Note there is a space between the command and the number. The number can also be enclosed in parenthesis.
loop	Designates the start of a matching repeat/loop sequence.
repeat <#>	Repeat previous set of statements "#" more times. Repeat will start at matching "loop" command or beginning of file (if none). Note that there is a space between the command and the number. The number can also be enclosed in parenthesis.
quit	Graceful exit of the Brutus test application
app(#)	Switch between application instances on DUAL_OUTPUT=1 systems.

THE LOOP/REPEAT PAIR

Sets of script commands can be repeated by using a `loop/repeat` pair, comprised of a `loop` command followed by a set of other commands, followed by a `repeat` command. The `loop/repeat` pair can be nested (another `loop`, commands, and `repeat` inside a `loop/repeat` pair) to create complex scripts.

Each time a `loop` command is processed, it adds to a nested count. When a `repeat` command is processed, its repeat count is decremented and, if not zero, control skips back to the point where the last `loop` command was processed. Once the repeat count is exhausted, the command processing continues past the `repeat`. When another `repeat` command is processed, control skips back to the `loop` command that marked the matching `loop/repeat` pair.

If a `repeat` command is encountered without a matching `loop` command (dangling repeat), then control is transferred to the beginning of the script. If a repeat is encountered without a repeat count then that loop will repeat forever.

SAMPLE SCRIPTS

Here are a few samples to get started. Once again, use the `--exec` option, which puts you into interactive mode, to learn about scripting.

Channel change script:

```
# Channel change
1
sleep 5
chanup
sleep 5
chanup
sleep 5
chanup
sleep 5
repeat 10
quit
```

PVR script:

```
# PVR
play(1); sleep 4
ff; sleep 2
ff; sleep 2
play
rew; sleep 2
rew; sleep 2
quit
```

Fast forward/reverse script:

```
# loop example
loop
play(1); sleep 5
loop; ff; sleep 4; repeat 3
loop; play; sleep 5; pause; sleep 5; repeat 5
loop; rew; sleep 2; repeat 3
loop; play; sleep 5; pause; sleep 5; repeat 5
stop
repeat 10
```

Section 12: Brutus Skinning

BRUTUS SKINNING OVERVIEW

A skin is a set of commands and media files that determine the overall look and feel of the Brutus user interface. This allows users to modify the user experience without having to recompile the application itself. Users have the freedom to use the predefined “blue” skin or can easily create new skins of their own.

Brutus skins are comprised of an XML-based command file (skin.xml) and associated image files. On startup, Brutus will use the instructions in the skin’s skin.xml file to generate Brutus’ graphical user interface.

STARTING BRUTUS WITH SKIN

Skin type can only be specified on startup and cannot be modified after Brutus has been started. The following example shows how to start Brutus with the “blue” sample skin.

```
$ settop brutus –skin blue
```

By default, Brutus will assume the “blue” skin directory resides in a directory named “skins” (alongside the brutus executable). It will also assume the skin command file (under “blue” directory) will be named: “skin.xml”.

Brutus configuration file options exist to modify these defaults:

Table 15: Brutus Configuration File Skin Options

Brutus Config File Options	***Description***
SKINS_PATH	Path where Brutus will attempt to load the specified skin.
SKIN	Default skin to use when running Brutus (“-skin” command line option will always override this if specified).
SKIN_FILE	Name of skin command file. (default: “skin.xml”)

XML Command File Syntax

Brutus' XML-based grammar, like HTML, uses markup tags and plain text. A *tag* is a keyword enclosed by the angle bracket (< and >) characters. A tag may also have *attributes* inside the angle brackets which specify associated features and settings. Each attribute consists of a *name* and a *value*, separated by an equal (=) sign.

Tags occur in pairs consisting of the start tag <*keyword*> and the end tag </*keyword*>. Between the start and end tag, other tags and text may appear. Everything from the start tag to the end tag is called an *element*. If one element contains another, the containing element is called the *parent* element of the contained element. The contained element is called a *child* element of its containing element. The parent element may also be called a *container*.

`<!--` and `-->` mark the start and end of an XML comment and can be placed almost anywhere within the command file.

Here is an example illustrating a "skin" topmost parent element and a "screen" child element, each with a corresponding name attribute:

```
<!—- My skin -->
<skin name=blue>
    <screen name=menu>
    </screen>
</skin>
```

Similar to how most web browsers work, malformed XML elements and attributes will be ignored by Brutus.

XML Tags

Unspecified element and attributes will default to that of the standard Brutus user interface look and feel.

The <skin> Tag

The <skin> element is the topmost element in the Brutus XML command file. Its attributes are given in Table 16.

Table 16: <skin> Tag Attributes

Attribute	***Description***
name	Name of the skin. Must match the name specified after the -skin command line option when running Brutus. Must also match the directory name where the skin.xml file resides.
pixelformat	argb8888 | argb1555 | rgb565 | palette8
bordercolor	Color of the border around the entire user interface. (see "Color Attribute Values" on page 50)

THE <SCREEN> TAG

The `<screen>` element and its child elements describe all the widgets for the named screen. Each screen may contain a set of *mandatory* widgets that are not explicitly described as child elements but can still be modified using attributes of the `<screen>` tag. For example, the playback screen contains a list control that always exists, and will not be described as an independent XML element. Changing the look of this list control can be accomplished using attributes of the playback screen tag. Additional buttons and labels can be added to any of the screens using the `<button>` and `<label>` tags. The mandatory widgets provide a general framework for screen functionality, while the optional `<button>` and `<label>` tags allow skin designers the freedom to change the look and feel of the user interface.

Note that screens that do not have an associated named `<screen>` tag are displayed using the standard Brutus user interface style.

Common <Screen> Attributes

These attributes, given in Table 17, are common to all `<screen>` tags:

Table 17: Common <Screen> Attributes

Attribute	***Description***
name	menu | playback | video | record | pictures | tv | audio Indicates the name of the associated user interface screen.
color	Screen background color (see “Color Attribute Values” on page 50)
image	Name of background image
pixmap	Name of background pixmap
height	Screen height
width	Screen width
x	X coordinate of upper left corner of screen
y	Y coordinate of upper left corner of screen
tvh	Height of decimated video window
tvw	Width of decimated video window
tvx	X coordinate of upper left corner of decimated video window
tvy	Y coordinate of upper left corner of decimated video window
dropshadow	1 for drop shadowed text, 0 otherwise
bevelwidth	Width of the bevel around the decimated video window
bevelcolor	Color of the bevel around the decimated video window (see“Color Attribute Values” on page 50)

Playback <Screen> Attributes

These attributes, given in Table 18, are only valid for the screen tag named “playback.”

Table 18: Playback <Screen> Attributes

Attribute	*Description*
listx	X coordinate of the upper left corner of the list of playback streams
listy	Y coordinate of the upper left corner of the list of playback streams
listw	Width of the list of playback streams
listh	Height of the list of playback streams
listbevelwidth	Width of the bevel around the list of playback streams
listbevelstyle	raised | sunken | updown Style of bevel around the list of playback streams
listdropshadow	1 for drop shadowed text, 0 otherwise
listrows	Number of visible rows in list of playback streams
listtextcolor	Color of unfocused text in list of playback streams (see “Color Attribute Values” on page 50)
listtextcolorfocus	Color of focused text in list of playback streams (see “Color Attribute Values” on page 50)
listfocusimage	Image used to display focus in list of playback streams. Image is scaled to required size.
listscrollbarimage	Image used to display the background of the scroll bar in list of playback streams
listscrollbarindimage	Image used to display the scroll bar thumb in list of playback streams

VIDEO <SCREEN> ATTRIBUTE

This attribute, given in Table 19, is only valid for the screen tag named “video.”

Table 19: Video <Screen> Attribute

Attribute	*Description*
focus	Name of the control that is to receive initial focus.

PICTURES <SCREEN> ATTRIBUTES

These attributes, given in Table 20, are only valid for the screen tag named “pictures.”

Table 20: Pictures <Screen> Attributes

Attribute	*Description*
listnothumbimage	Name of placeholder image used when a thumbnail does not exist and the original image is too large to downscale.
iconwidth	Width of each thumbnail image
iconborder	Empty space allotted around each thumbnail image

TV <Screen> Attributes

These attributes, given in Table 21, are only valid for the screen tag named “tv”.

Table 21: TV <Screen> Attributes

Attribute	***Description***
infobevelwidth	Width of bevel on floating information windows
infobevelstyle	raised | sunken | updown Style of bevel on floating information windows
infocolor	Information windows background color (see “Color Attribute Values” on page 50)
infobannercolor	Background color of embedded status banner (see “Color Attribute Values” on page 50)
infobannerbevelstyle	raised | sunken | updown Style of bevel on embedded status banner
infobannerbevelwidth	Width of bevel on embedded status banner
halign	left | right | center Horizontal alignment of the primary information window menu button text
infotextcolor	Color of the unfocused text in the information windows (see “Color Attribute Values” on page 50s)
infotextcolorfocus	Color of the focused text in the information windows (see “Color Attribute Values” on page 50)
infoimagefocus	Image used to display a focused button in the primary information window
infobuttonimageOn	Image used to display a focused button in the secondary information windows
infobuttonimageOff	Image used to display an unfocused button in the secondary information windows
infobuttonimageDown	Image used to display a pressed button in the secondary information windows
pvrbarcolor	Color of the PVR progress bar. (see “Color Attribute Values” on page 50)
pipx	X coordinate of the upper left corner of the PIP window
pipy	Y coordinate of the upper left corner of the PIP window
pipw	Width of the PIP window
piph	Height of the PIP window

Audio <Screen> Attributes

These attributes, given in Table 22, are only valid for the screen tag named “audio.”

Table 22: Audio <Screen> Attributes

Attribute	***Description***
listx	X coordinate of the upper left corner of the list of audio files
listy	Y coordinate of the upper left corner of the list of audio files
listw	Width of the list of audio files
listh	Height of the list of audio files
listbevelwidth	Width of the bevel around the list of audio files
listbevelstyle	raised | sunken | updown Style of bevel around the list of audio files
listdropshadow	1 for drop shadowed text, 0 otherwise
listrows	Number of visible rows in list of audio files
listtextcolor	Color of unfocused text in list of audio files (see “Color Attribute Values” on page 50)
listtextcolorfocus	Color of focused text in list of audio files (see “Color Attribute Values” on page 50)
listfocusimage	Image used to display focus in list of audio files. Image is scaled to required size.
listscrollbarimage	Image used to display the background of the scroll bar in list of audio files
listscrollbarindimage	Image used to display the scroll bar thumb in list of audio files

The <IMG> Tag

The `<img>` element declares an image resource for use throughout the skin.xml file. All `<img>` elements are global, regardless of element nesting, and can be referenced from anywhere within the skin.xml file. `<img>` elements must include the filename of the associated image between the start and end tags as follows:

```
<img name=testimage>test.png</img>
```

All `<img>` tags must also contain a name attribute used to reference it by other elements in the skin.xml file. By themselves, `<img>` tags will not display anything on screen; they are only a means to declare shared image resources.

COMMON <LABEL> AND <BUTTON> ATTRIBUTES

These attributes, given in Table 22, are common to all `<label>` and `<button>` tags.

Table 23: Common <label> and <button> Attributes

Attribute	***Description***
name	Name used to identify widget
color	Background color (see "Color Attribute Values" on page 50)
image	Name of background image
pixmap	Name of background pixmap
height	Widget height
width	Widget width
x	X coordinate of upper left corner of widget
y	Y coordinate of upper left corner of widget
textcolor	Color of unfocused text (see "Color Attribute Values" on page 50)
textcolorfocus	Color of focused text (see "Color Attribute Values" on page 50)

font	Format: *name*;*size*[;AA][;PK]
	If pre-rendered font, size = –1
	AA = anti-aliasing true
	PK = kerning true

dropshadow	1 for drop shadowed text, 0 otherwise
action	Assign an action to the widget. Actions may be generic (usable on any screen) or screen specific. (see "Action Attributes" on page 49)
halign	left | center | right Horizontal text alignment
valign	top | center | bottom Vertical text alignment
bevel	raised | sunken | updown 3D widget border type
bevelwidth	Width of widget border

The <label> Tag

The `<label>` element creates a widget that can be used to display text or an image. Action attributes, described in "Action Attributes" on page 49, can be assigned to dynamically set label text.

Figure 8: <label> Tag Attributes

Attribute	***Description***
scale	single | stretch | max | down Scaling mode to use for label image. Has no effect if image not assigned to label. • single: no scaling • stretch: scale to fit label dimensions (non-proportionally) • max: scale to fit label dimensions (proportionally) • down: scale down to max only (proportionally)
image	Image name to be displayed in label
wrap	word | newline | none Text wrap mode.

THE <BUTTON> TAG

The `<button>` element, whose attributes are given in Table 24, creates a widget that can be used to display images and/or text. When selected by user, it will perform some action (see "Action Attributes" on page 49).

Table 24: <button> Tag Attributes

Attribute	*Description*
imageOn	Image used for button in focused state.
imageOff	Image used for button in unfocused state.
imageDown	Image used for button in selected state.
imageIcon	Icon displayed to left of button text.

ACTION ATTRIBUTES

Action attributes, given in Table 25, can be applied to both buttons and labels. An action can be applied to a button which will perform some function when pressed by the user. When applied to a label, an action will update its appearance dynamically. A particular widget can only have one associated action.

Table 25: Action Attributes

Attribute	*Description*	*Screen*
showAdminWin	Display the admin screen.	All
showAudioWin	Display the MP3 player screen.	All
showInfoWin	Go to full screen video and display the Info screen.	All
showMenuWin	Display the main menu screen.	All
showPhotoWin	Display the picture viewer screen.	All
showVideoWin	Display the video details screen.	All
showPlaybackWin	Display available playback streams screen.	All
showRecordWin	Display record options screen.	All
playVideo	Start video playback of selected stream.	Video
resumeVideo	Continue video playback of paused stream.	Video
renameVideo	Display video rename popup.	Video
deleteVideo	Display video delete confirmation popup.	Video
platformName	Sets widget text attribute to the platform name	All
platformNameLong	Sets widget text attribute to a detailed version of the platform name.	All
platformDesc	Sets widget text attribute to a short description of the platform's capabilities.	All
record	Starts a recording.	Record
save	Saves program guide.	Record
videoTitle	Sets widget text attribute to the currently selected video's title.	Video
videoDate	Sets widget text attribute to the currently selected video's date.	Video
videoFilename	Sets widget text attribute to the currently selected video's filename.	Video

Table 25: Action Attributes (Cont.)

Attribute	Description	Screen
videoSize	Sets widget text attribute to the currently selected video's size.	Video
videoLength	Sets widget text attribute to the currently selected video's length.	Video
videoIndex	Sets widget text attribute to the currently selected video's index.	Video
videoVFormat	Sets widget text attribute to the currently selected video's format.	Video
videoAFormat	Sets widget text attribute to the currently selected video's audio format.	Video
videoVPid	Sets widget text attribute to the currently selected video's video PID.	Video
videoAPid	Sets widget text attribute to the currently selected video's audio PID.	Video
videoEncryption	Sets widget text attribute to the currently selected video's encryption.	Video
playAudio	Plays selected audio file.	Audio
playAllAudio	Plays all audio files sequentially.	Audio
stopAudio	Stops audio playback.	Audio

COLOR ATTRIBUTE VALUES

Color attribute values can be specified in two ways:

1. ARGB value: #AARRGGBB where AA = Alpha value 00-FF, RR = Red value 00-FF, GG = Green value 00-FF, and BB = Blue value 00-FF
2. Natural language color names. The following color names are supported: red, green, blue, magenta, cyan, yellow, black and white.

EXAMPLE "BLUE" SKIN

The example "blue" skin can be found at this location: BSEAV/app/brutus/samples/skins/blue.

Section 13: Brutus MP3 Playback

Compiling Audio UI Support

There are compile-time and run-time controls for MP3 decode. If you see the **MP3s** button on the Main Menu, then you have correctly enabled both.

The compile-time control is `export AUDIO_SUPPORT=y`, which is the default for IDE-enabled chips.

The run-time control is `MP3_ENABLED=y` in the brutus.cfg file. This is also set in the default brutus.cfg file.

Installing MP3 Decoders

Open source software decoders can be used to play MP3 files. Brutus has support to call two common decoders:

- mpg123: A floating-point decoder for FPU-enabled chips like the BCM7038 and BCM7400.
- madplay: A fixed-point decoder for no-FPU chips like the BCM7401 and BCM7118.

These programs are open source and GPL. The Settop API Reference Software is in no way a derivative work of these programs. It can call these programs through a shell interface and they are optional.

Broadcom does not include mpg123 or madplay in the reference release. They are available on the Internet and are easily cross-compiled for MIPS.

Section 14: Display Formats

By default, Brutus starts in NTSC output mode. There are three ways to change display formats in Brutus:

1. Use Info panel GUI.
2. Set `OUTPUT_FORMAT` in brutus.cfg.
3. If an HDMI display is connected, allow Brutus to switch to the HDMI display's preferred format.

The Info panel GUI is available by pressing the **Info** button on the remote or the **Info** button on the home screen.

The `OUTPUT_FORMAT` brutus.cfg option can be used to change the default format from NTSC to something else. The format you want must be spelled exactly, including spaces for the 50-Hz modes. The options are given in Table 26.

Table 26: OUTPUT_FORMAT String Options

OUTPUT_FORMAT strings	***Description***
480i, NTSC	NTSC synonyms
576i, PAL	PAL synonyms
720p, 1080i, 1080p 24Hz, 1080p 30Hz	ATSC/60Hz HD modes
720p 50Hz, 1080i 50Hz, 1080p 25Hz	50Hz HD modes
1366x768p, 1024x768p, 800x600p, 640x480p, vga, 1280x768p, 1280x720p	VESA modes
NTSC J, PAL M, PAL N, PAL NC, SECAM	Other NTSC, PAL and SECAM modes

When running the BCM7038, BCM7401 and BCM7400 in a dual-compositor mode, you can only change the format of the primary display. The secondary display is always NTSC, PAL or SECAM. It defaults to NTSC unless `OUTPUT_FORMAT` is set to a PAL/50Hz format.

When HDMI is connected (on platforms where HDMI output is supported), Brutus will read the EDID information and set the display to that preferred format. You can turn off this behavior by setting `DVI_USE_PREFERRED_FORMAT=n` in brutus.cfg. This does not take precedence over any `OUTPUT_FORMAT` setting.

1080p display formats are available with a Settop API compile-time configuration option. See BSEAV/api/src/magnum/board/bsettop_bsp_CHIP.h for B_HAS_1080P.

`AUTO_OUTPUT_FORMAT=true` instructs Brutus to set the display video format to match the video source format, as much as possible. If HDMI is connected, it will check with the supported HDMI formats. This auto-mode will not switch between 50 and 60 Hz modes.

Section 15: Settop API and Brutus Run-Time Options

To facilitate debugging, the Settop API and Brutus use environment variables to adjust internal state. The internal function is called bsettop_get_config. Magnum does not have any concept of environment variables.

The way the variables are set varies by system:

- In Linux usermode (include the proxy layer), these are environment variables.
- In Linux kernelmode, use the config variable to pass these in (see above).
- In VxWorks, there is a public function calls bsettop_set_config which you can call to set variables.

These run-time options are not part of the public API and are subject to change. Ultimately, you should grep the code for “bsettop_get_config” to find out what the options are and how to use them. The options can vary by platform.

The following is a partial list:

export msg_modules=XXXX,XXXX

Set the BDBG interface modules which should run at MSG level. Grep the api/src code for BDBG_MODULE() statements to find the module names.

export sync_disabled=yes

Disable the bsync module. TSM will still be enabled, but compositor sync and precision lipsync will not be active.

export force_vsync=yes

Do not enable TSM (time stamp managed) mode for audio or video decode. “No TSM” is also called vsync mode (even when applied to audio which has no vsync per se).

export not_realtime_isr=yes

By default, ISR processing thread in user mode runes at the highest priority. This option makes it run at normal priority which maximizes our chances of catching isr race conditions. This should be used for testing only.

export no_watchdog =yes

Disable audio and video decoder watchdog processing. The only reason to disable it is to catch decoder bugs during development. It’s also useful on the BCM74xx if you are using the ARC prompt, which halts the decoder and generates watchdogs.

export no_brcm_trick_modes=yes

If set, only use host trick modes.

export jail_avd=yes

Use memory controller's address range checkers to see if any AVD (BCM740x video decoder) client is accessing memory outside of its allocations.

export jail_xpt=yes

Use memory controller's address range checkers to see if any XPT client is accessing memory outside of its allocations.

export avoffset=OFFSET

Apply an AV offset (a.k.a. PTS offset) to both audio and video decode. OFFSET is a decimal value in PTS units (45 KHz for MPEG-2 TS, 27 MHz for DSS).

export sw_destripe=yes

For still-picture decode on BCM740x platforms, do not use the M2MC blitter. Use a software blitter instead.

export cont_count_ignore=yes

Do not enable transport continuity counter checking in XPT or RAVE. All packets will reach the decoders.

export bypass_7411=yes

On BCM97398 systems, do not use the BCM7411. Instead use the BCM7038's MVD and AUD decoders.

export simple_bcmplayer=yes

Disable Broadcom trick mode performance optimizations

export BCM3520_TUNER=XXX

See bsettop_tuner_3520.c for different tuner types which are supported.

export hdmi_bypass_edid=yes

HDMI will not read or parse the EDID.

export use_rap_trick_modes =yes

BCM740x platforms now default to STC trick modes for audio and video pause and slow motion. You can switch back to decoder trick modes with this.

export indep_spdif_delay=XXX

On BCM740x platforms, set the independent S/PDIF delay feature to XXX milliseconds. By default, the feature is off.

export max_hd_display_format=1080p
export max_hd_display_format=1080p_30Hz

On BCM740x platforms, enable 1080p display formats. These do not default on because they require significantly more memory. The 1080p_30Hz setting allows for 24, 25, and 30. The 1080p setting also allows for 60 Hz, which is not supported by BCM740x HW.

See the *Brutus Installation Guide* (document number STB_Brutus-SWUM200-R) for compile-time options.

Broadcom Corporation

5300 California Avenue
Irvine, CA 92617
Phone: 949-926-5000
Fax: 949-926-5203