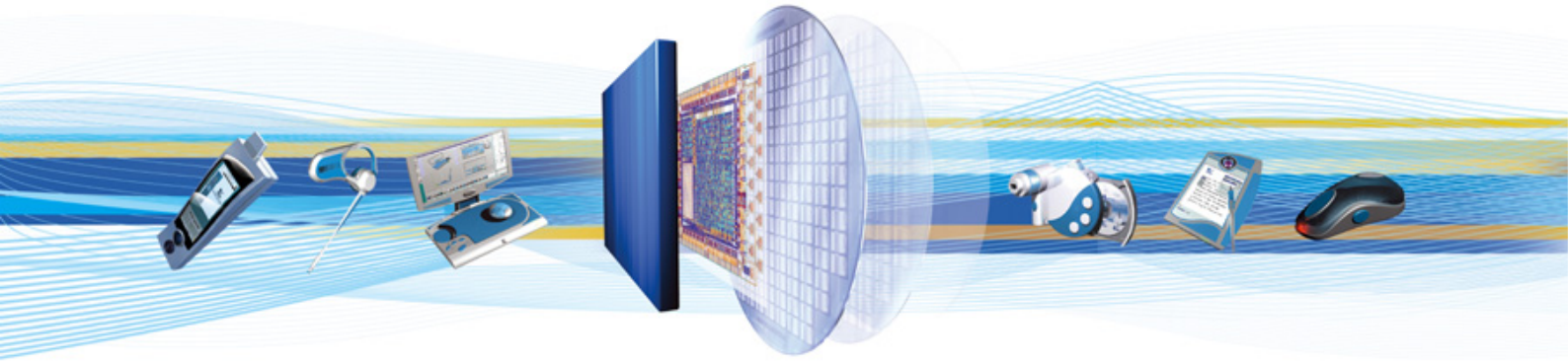
CSR Synergy Bluetooth 18.2.0

Bluetooth Bootstrap Guide

November 2011



Cambridge Silicon Radio Limited

Churchill House
Cambridge Business Park
Cowley Road
Cambridge CB4 0WZ
United Kingdom

Registered in England and Wales 3665875

Tel: +44 (0)1223 692000
Fax: +44 (0)1223 692001
www.csr.com

Contents

1 Bluetooth Bootstrap

1.1 Introduction

The CSR Synergy Bluetooth Bootstrap is an example on how the CSR BlueCore chips can be initialised and patched. This process is known as *bootstrapping*.

Patching is the process of uploading PS-Keys to the chip RAM, which can then fix known problems with e.g. ROM based chips.

The initialisation part of the bootstrap usually sets up platform and product specific settings. For example, a platform setting may want to use a higher UART bit-rate than what is available at power-on. A product specific setting is for example the local Bluetooth address, which must be unique amongst all Bluetooth devices in the world.

Because different platforms have vastly different requirements on how and when the bootstrap system must be applied, CSR has implemented several *examples* on how the bootstrap process can be implemented. These specific implementation examples can be found in the directory "*applications/bootstrap/*…". The examples implement the mandatory part of the BlueCore bootstrap process, as required by the CSR Synergy Framework.

The effort required by the application developer is minimal. Typically, a bootstrap requires only a few lines of code.

The example bootstrap consists of a *generic* part and a *specific* part. The generic part of the bootstrap should not be modified – it contains various helper functions that are part of almost every BlueCore bootstrap process. The *specific* part, on the other hand, usually needs modification by the application developer. It is the specific bootstrap process that tells the generic part which patch bundle and platform specific settings that need to be applied. The process is illustrated on Figure 1.

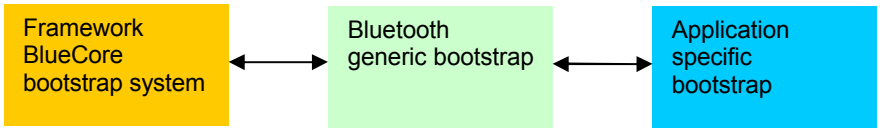


Figure 1: CSR Synergy Bluetooth Bootstrap Process

On Figure 1, the CSR Synergy Framework already implements the yellow box, and CSR Synergy Bluetooth has implemented the green box. Only the blue box – the specific bootstrap – needs to be tweaked by the application developer. The following sections describe the generic bootstrap and three examples of the specific bootstrap.

1.2 Generic Bootstrap Functions

The generic bootstrap functions are a complete set of helper-functions that the specific bootstrap can use to more quickly get the bootstrap up and running. The generic bootstrap implements the following functions that the specific bootstrap is free to use. In all functions, the *bootContext* member should be set to NULL.

```
void CsrBtBootstrapSetInterface(void *bootContext,
                                CsrUint16 hostInterface);
```

This sets up what host interface the BlueCore should use after the bootstrap.

```
void CsrBtBootstrapSetBitRate(void *bootContext,
                              CsrUint32 bitRate);
```

This sets the post-boostrap bit rate. For BC7 and newer chips, the bit rate is written directly as a PS-Key. For older chips, the bit rate needs to be converted to a baud-rate. The generic bootstrap procedure will do this internally.

```
void CsrBtBootstrapSetFriendlyName(void *bootContext,
                                   CsrCharString *name);
```

This sets the local device friendly name.

```
void CsrBtBootstrapSetLocalBtAddress(void *bootContext,
                                     CsrBtDeviceAddr *address);
```

This sets the local device Bluetooth address.

```
void CsrBtBootstrapSetFrequency(void *bootContext,
                                CsrUint16 frequency);
```

This sets the external crystal frequency.

```
void CsrBtBootstrapSetFreqTrim(void *bootContext,
                               CsrUint16 freqTrim);
```

This sets the external crystal frequency trim.

```
void CsrBtBootstrapParsePsrFile(void *bootContext,
                                const CsrCharString *filename);
```

This instructs the generic bootstrap to load, parse and apply a PSR file containing.

```
void CsrBtBootstrapParsePsrConstant(void *bootContext,
                                    const CsrCharString *string);
```

This instructs the generic bootstrap to parse and apply compiled-in constant C-string containing PSR entries.

1.3 Specific Bootstrap: File Based

The file *csr_bt_bootstrap_specific_file_based.c* is an example on how the bootstrap system can work with different chips, and use the on-chip firmware build-id to select the correct PSR file.

CSR has provided several PSR files for the most popular chip variants with CSR Synergy. These files can be found in the directory *"applications/bootstrap/psr/…"*.

1.4 Specific Bootstrap: Constant String Based

The file *csr_bt_bootstrap_specific_string_based.c* is an example on how the bootstrap system can select between multiple compiled-in PSR strings, and apply the correct string based on the on-chip firmware build-id.

This example is the default used by the CSR demo applications.

CSR has provided pre-generated C files with the PSR-strings. Each string has its own C-file, which has been generated based on the PSR-file. The C-string files can be found in the directory *"applications/bootstrap/psr/…"*.

1.5 Specific Bootstrap: Custom Platform

The file *csr_bt_bootstrap_specific_platform.c* is an example on how a minimal setup for a customer platform can be implemented.

Instead of manually loading the PSR files, the application developer can also use the compiler defines in *csr_bt_bootstrap_specific.h* to automatically load either a PSR-file or PSR-string.

1.6 PSR to C Converter Script

If the platform on which CSR Synergy runs does not have a file system, the file based PSR loading cannot be used. Instead, the bootstrap PSR files can be converted to constant C-style strings that can be compiled in with the CSR Synergy binary.

In order to ease the conversion from PSR-file to PSR-string, CSR has provided a very simple Perl script that can perform this task. The script is located in *"applications/bootstrap/psr/psr2c.pl"*. The syntax is:

```
perl psr2c.pl <psrfile>
```

The output file can then be compiled in with the rest of the bootstrap files, and the `CsrBtBootstrapParsePsrConstant()` function can be used to parse and apply the settings from the PSR string.

1.7 Custom Bootstrap Implementation

If the application developer chooses not to use the CSR CSR Synergy Bluetooth Bootstrap system described below, it is still necessary to implement the mandatory BlueCore bootstrap function:

```
CsrUint8 **CsrTmBlueCoreGetBootstrap(CsrUint16 buildId,
                                     CsrUint16 *entries)
```

The function is given the specific build-id for the BlueCore chip. The function must return an array of pointers of BCCMDs. The number of pointers must be stored in **entries*. The framework will ensure the pointers are written in order.

Terms and Definitions

BlueCore®	Group term for CSR's range of Bluetooth wireless technology chips
Bluetooth®	Set of technologies providing audio and data transfer over short-range radio connections
CSR	Cambridge Silicon Radio
SBC	Sub-Band Codec
UniFi™	Group term for CSR's range of chips designed to meet IEEE 802.11 standards

Document History

Revision	Date	History
1	26 SEP 11	Ready for release 18.2.0

TradeMarks, Patents and Licences

Unless otherwise stated, words and logos marked with ™ or ® are trademarks registered or owned by CSR plc or its affiliates. Bluetooth® and the Bluetooth logos are trademarks owned by Bluetooth SIG, Inc. and licensed to CSR. Other products, services and names used in this document may have been trademarked by their respective owners.

The publication of this information does not imply that any licence is granted under any patent or other rights owned by CSR plc.

CSR reserves the right to make technical changes to its products as part of its development programme.

While every care has been taken to ensure the accuracy of the contents of this document, CSR cannot accept responsibility for any errors.

Life Support Policy and Use in Safety-critical Compliance

CSR's products are not authorised for use in life-support or safety-critical applications. Use in such applications is done at the sole discretion of the customer. CSR will not warrant the use of its devices in such applications.

Performance and Conformance

Refer to www.csrsupport.com for compliance and conformance to standards information.