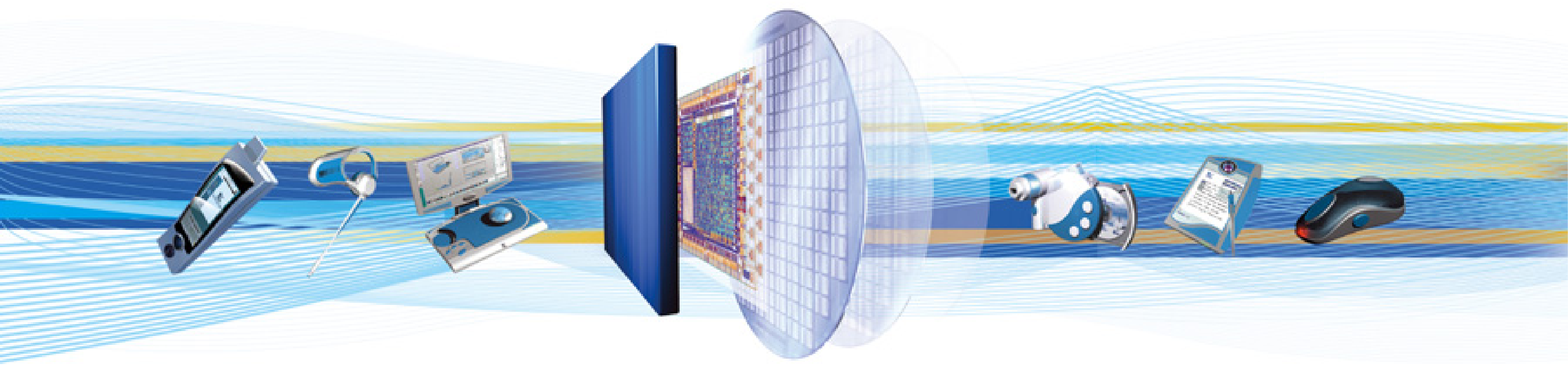
CSR Synergy Bluetooth 18.2.1

Users Guide

December 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

gu-0101

Contents

List of Figures

List of Tables

1 Introduction

CSR Synergy Bluetooth is a portable and scalable Bluetooth® embedded software stack. CSR Synergy Bluetooth is developed to operate with CSR's family of BlueCore ICs. CSR Synergy Bluetooth is intended for embedded products that have a host processor for running the CSR Synergy Bluetooth and the Bluetooth® application. The CSR Synergy Bluetooth stack together with the BlueCore IC is a complete Bluetooth® system solution complying with Bluetooth® specification 2.0, 2.1, 3.0 & 4.0 from RF to profiles.

CSR Synergy Bluetooth includes much of the Bluetooth intelligence and gives the user a simple API. This makes it possible to develop a Bluetooth product without in-depth Bluetooth knowledge.

Please read the license agreements before use. All documents and files are delivered in a file structure as described in 3.2.

CSR Synergy Bluetooth contains 3 elements:

- Bluetooth® Profile Managers
- Example applications
- Bluetooth® core stack

The profiles implemented in the CSR Synergy Bluetooth are focused on mobile phones and automotive devices. CSR Synergy Bluetooth can, however, be used in a number of different products, a more detailed description of the profiles in CSR Synergy Bluetooth can be found in chapter 2.2.

CSR Synergy Bluetooth is delivered with source code and documentation.

All user documentations are located in: `./doc`

This user guide gives an overview of the CSR Synergy Bluetooth architecture, but is also a guide to how to configure the software.

2 CSR Synergy Bluetooth Architecture

2.1 CSR Synergy Bluetooth Architecture

The CSR Synergy Bluetooth software must run in a host processor. CSR Synergy Bluetooth is designed so that the Bluetooth Profiles are running on the host while the lower part of the Bluetooth stack is running on-chip.

In Figure 1 this split is depicted.

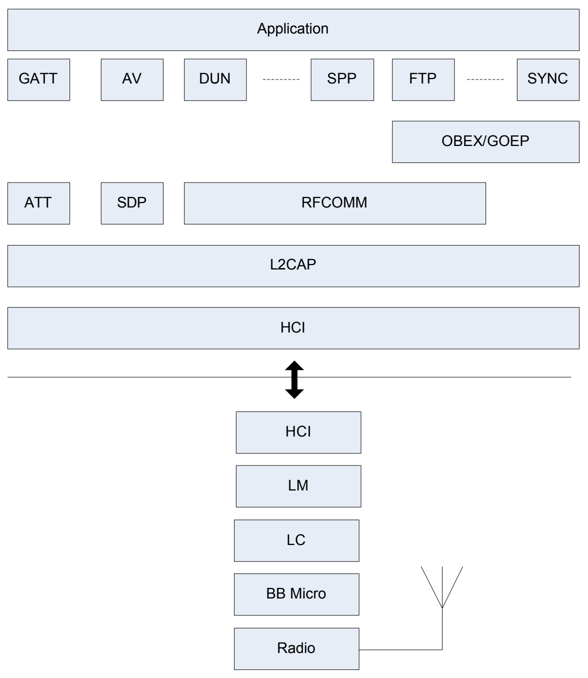


Figure 1: Split between host and Chip

The main component in CSR Synergy Bluetooth is the Profile Managers. The Profile Managers is an abstraction layer added on top of the Bluetooth protocol stack. The Profile Managers provide an interface for applications wanting to take advantage of the Bluetooth® features and give the user a simple API. The CSR Synergy Bluetooth API enables manufacturing of a Bluetooth® product without in depth Bluetooth® knowledge.

To run CSR Synergy Bluetooth the CSR Synergy Framework is needed as this makes the platform in which the different components within CSR Synergy Bluetooth run. The CSR Synergy Framework and CSR Synergy Bluetooth architecture is depicted in Figure 2. The architecture and the interfaces are the same for the two split of CSR Synergy Bluetooth. The only difference is one module. This means that it is easy to migrate from one version to the other.

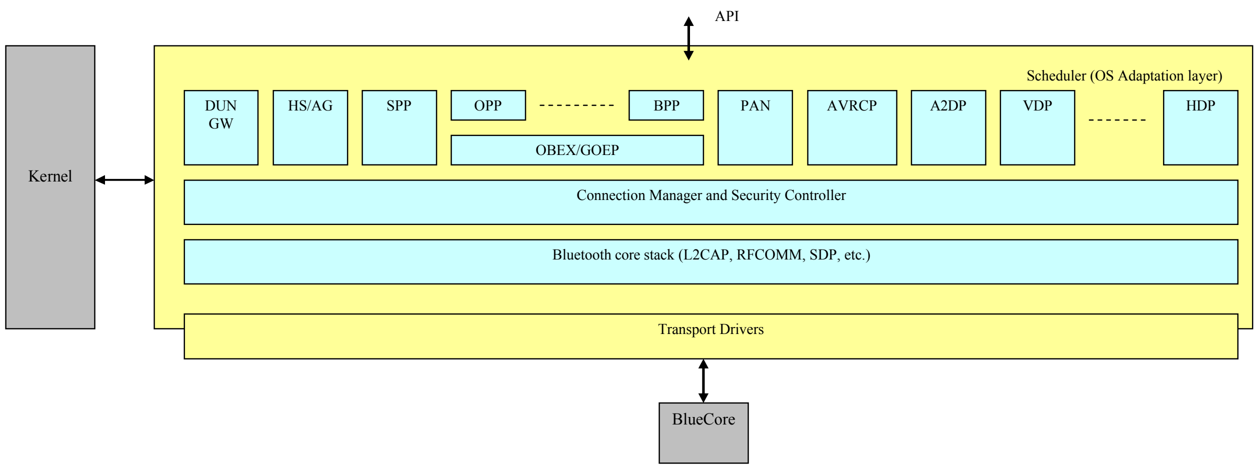

API
Scheduler (OS Adaptation layer)
Kernel
DUN GW
HS/AG
SPP
OPP
BPP
OBEX/GOEP
PAN
AVRCP
A2DP
VDP
HDP
Connection Manager and Security Controller
Bluetooth core stack (L2CAP, RFCOMM, SDP, etc.)
Transport Drivers
BlueCore

Figure 2: CSR Synergy Bluetooth architecture

Part of CSR Synergy Bluetooth is example applications. The example applications may be used as examples of how to utilise the CSR Synergy Bluetooth API and how to build an application on top of a certain profile. For most profiles interworking with existing applications must be provided as part of the integration process.

The architecture reflects the profiles supported by CSR Synergy Bluetooth at present. New profiles can be added alongside the existing profile managers. Further it is expected that new control components must be added for e.g. handover. A more detailed description of the individual components can be found in the following subsections.

2.2 Bluetooth® Profile Managers Components

2.2.1 Dial-Up Network

The Dial-up Networking profile defines the protocols and procedures that shall be used by devices implementing the usage model called ‘internet bridge’. The scenarios covered by this profile are the following:

- Usage of a cellular phone or modem by a computer as a wireless modem for connecting to a Dial-Up internet access server, or using other Dial-Up services
- Usage of a cellular phone or modem by a computer to receive data calls

The following roles are defined:

Gateway (GW): This is the device that provides access to the public network. Typical devices acting as gateways are cellular phones and modems.

Data Terminal (DT)*:* This is the device that uses the services of the gateway. Typical devices acting as data terminals are laptops and desktop PCs.

The Dial-Up Network (DUN) profile manager is the component for support of both the DUN Gateway and DUN Data Terminal features specified in [DUN].

The profiles make use of the Connection Manager (see 2.2.16) and provide functionality for:

- Establish and negotiate a serial port [SPP] emulation connection towards a remote device
- Send and receive data to and from the remote device

The DUN profile have requirements on dialing and control interoperability. This functionality is not included in the DUN GW, because it is already available in some existing applications. The dialling and control functionality is therefore implemented in a separate component (AT-Interpreter) that is put on top of the GW. The AT-Interpreter is described in the next section.

Typically the Point to Point Protocol (PPP) is used for Dial-Up connections. However, PPP is not part of the DUN profile and normally resides on the external server side. Thus, PPP is not needed in the terminal for DUN and hence not part of the DUN manager.

2.2.2 AT Dial-Up Network Gateway

The dialing and control functionality in DUN is implemented in a separate component called the AT-Interpreter. This functionality can be omitted if it e.g. already is implemented in an existing application.

The AT-Interpreter is put on top of the DUN GW and all communication with the application is going through the AT-Interpreter. The application will therefore see the AT-Interpreter as a Dial-Up Network gateway with AT-Interpreter functionality.

The AT-Interpreter interprets the AT-Commands being used for setting up the modem connection. The AT interpreter supports the AT-commands defined in the DUN profile and can handle a number of AT-commands (configurable) independently of the application on top of the AT-interpreter; this approach simplifies the interface between the AT-interpreter and the application and the application layer itself.

2.2.3 OBEX

The Object Exchange (OBEX) profile defines the protocols and procedures used for Synchronisation Profile [SP], Object Push Profile [OPP], File Transfer Profile [FTP], Basic Imaging Profile [BIP], SyncML over OBEX Profile [SMLP], Basic Printing Profile [BPP] and Phone Book Access Profile [PBAP]. The use cases typically involve a phone or a PC, but may also involve other devices such as PDAs.

The OBEX profile manager makes use of the connection manager (see 2.2.16) and is divided into a server part and client part. The OBEX profile manager has the following function:

- Establish a channel for exchanging objects between an OBEX server and OBEX client

Exchange of objects includes both push, pull and mutual push/pull, i.e. exchange.

It is assumed that the formatting of the OBEX objects are handled in the application layer, e.g. by an already existing application function.

Synchronization Profile

The synchronization profile defines the requirements for the protocols and procedures that shall be used by the applications providing the synchronization usage model. The most common devices using these usage models might be notebook PCs, PDAs and mobile phones.

The scenarios covered by this profile are:

- Usage of a mobile phone or PDA by a computer to exchange PIM (Personal Information Manager) data, including necessary log information to ensure that the data contained within their respective Object Stores is made identical. Type of the PIM data are, for example phonebook and calendar items
- Use of a computer by a mobile phone or PDA to initiate the previous scenario (Sync Command Feature)
- Use of a mobile phone or PDA by a computer to automatically start synchronization when a mobile phone or PDA enters the RF proximity of the computer

It is only the sync server that is implemented in CSR Synergy Bluetooth.

Object Push Profile

The Object Push profile defines the protocols and procedures that shall be used by users implementing the user model ‘Object Push’.

The profile makes use of Generic Object Exchange Profile (GOEP) and OBEX.

The Scenarios covered by this profile are the following:

- Usage of a Bluetooth® device e.g. a mobile phone to push an object to the inbox of another Bluetooth® device. The object can e.g. be a business card or an appointment
- Usage of a Bluetooth® device e.g. a mobile phone to pull a business card from another Bluetooth® device
- Usage of a Bluetooth® device e.g. a mobile phone to exchange business cards with another Bluetooth® device. An exchange defined as a push of a business card followed by a pull of a Business card

File Transfer Profile

The File Transfer profile [FTP] defines the protocols and procedures to be used when implementing the user model 'File Transfer'.

The profile makes use of Generic Object Exchange Profile (GOEP) and OBEX.

The scenarios covered by this profile are the following:

- Usage of a Bluetooth® device e.g. a mobile phone to push an object to another Bluetooth® device. The object can e.g. be a file or a folder
- Usage of a Bluetooth® device e.g. a mobile phone to pull an object from another Bluetooth® device. The object can e.g. be a file or a folder
- Usage of a Bluetooth® device e.g. a mobile phone to delete an object on another Bluetooth® device
- Usage of a Bluetooth® device e.g. a mobile phone to create or delete folders on another Bluetooth® device

Basic Imaging Profile

The Basic Imaging Profile [BIP] defines the protocols and procedures to be used when implementing the user model 'Image Push'. The profile makes use of Generic Object Exchange Profile (GOEP) and OBEX.

The scenarios covered by this profile are the following:

- Usage of a Bluetooth® device e.g. a camera to push one or more images to another Bluetooth® device. The object can e.g. be a mobile phone
- Usage of a Bluetooth® device to remotely control a Bluetooth® enabled camera to monitor the image, control the shutter, and pull one or more images. The object can e.g. be a mobile phone
- Usage of a Bluetooth® device to automatically archive the images stored on another Bluetooth® device.

The Basic Imaging Profile user model 'Image Push', 'Remote Camera' and 'Automatic Archive' is implemented as two modules, where BIPC is the Imaging Initiator and BIPS is the Imaging Responder.

SyncML over OBEX Profile

The SyncML over OBEX Profile [SMLP] defines protocols and procedures that shall be used by users implementing applications providing SyncML functionality e.g. a SyncML-client or a SyncML-server. The most common devices using this might be notebook PCs, PDAs and mobile phones.

The profile makes use of Generic Object Exchange Profile (GOEP) and OBEX.

The scenarios covered by this profile are the following:

- Use of a mobile phone or PDA having a SyncML client to make connection to a SyncML server placed on another device e.g. a notebook
- Usage of a Bluetooth® device e.g. a mobile phone having a SyncML client to push a SyncML message object to another Bluetooth® device having a SyncML server
- Usage of a Bluetooth® device e.g. a mobile phone having a SyncML client to pull a SyncML message object from another Bluetooth® device having a SyncML server

Basic Printing Profile

The Basic Printing Profile [BPP] defines the protocols and procedures to be used when implementing the user model 'Direct Printing'.

The scenarios covered by this profile are the following:

- Usage of a Bluetooth® device for printing emails, images, text messages and/or formatted text from a sender. The Sender can e.g. a mobile phone or a PDA

The Basic Printing Profile is implemented in two modules:

- BPPS – The printer server profile acts as a simple- or job controlled printer spool.
- BPP – The printer client is capable of printing both simple- and job based.

The profile makes use of Generic Object Exchange Profile (GOEP) and OBEX.

Phone Book Access Profile

The Phone Book Access Profile [PBAP] defines protocols and procedures that shall be used by users implementing the user model 'Phone Book Access'.

The profile makes use of Generic Object Exchange Profile (GOEP) an OBEX.

The Scenarios covered by this profile are the following:

- Usage of a Bluetooth® device e.g. a car kit access the list of phone book objects stored on a Mobile device e.g. a mobile phone
- Usage of a Bluetooth ® device down load one or several phone book entries from a Mobile device
- Usage of a Bluetooth ® device access the call histories stored on a Mobile device
- Usage of a Bluetooth ® device access the list of missed call from a Mobile device

2.2.4 Serial Port Profile

A Serial Port can be emulated. The Serial Port Profile (SPP) provides an interface for setting up a virtual serial link between two Bluetooth® devices. The SPP provides the following features:

- Establish virtual serial connection (client side)
- Accept virtual serial connection (server side)
- RS-232 control line exchange
- Sending/receiving data

A serial port emulation is typically used in legacy applications with Bluetooth® as a cable replacement.

2.2.5 Hands-Free Profile

Using a hands-free device in conjunction with a mobile terminal is supported by the hands-free profile. The hands-free use case typically involves a hands-free set in a car and a phone.

The hands-free manager utilises the connection manager and provides the following features:

- Establish a service level connection between the hands-free unit and a terminal
- Audio and AT-Command interface for control
- Transparent handling of low power modes if supported by the HF

The implementation of the HF device and HF gateway supports the headset and the device side of the headset profile respectively.

2.2.6 Cordless Telephony Profile and Intercom Profile

The Cordless Telephony profile [CTP] and Intercom Profile [ICP] define the protocols and procedures to be used when implementing cordless telephone systems.

The profiles make use of message formats and procedures defined in the TCS-BIN specification [TCS] part.

The scenarios covered by these profiles are the following:

- Usage of a Bluetooth® device e.g. a mobile phone to make and receive calls from a CTP server/gateway
- Usage of a Bluetooth® device as gateway between Bluetooth® devices making and receiving calls from an external network
- Usage of a Bluetooth® device e.g. a mobile phone to make and receive intercom calls from other cordless phones

2.2.7 Personal Area Networking Profile

The PAN profile is used for Ethernet packet encapsulation in conjunction with higher layer network protocols, e.g. TCP/IP. The CSR Synergy Bluetooth PAN profile supports the following:

- Establish and disconnect a BNEP connection between a pair of PAN devices
- Send data to a remote device. Receive data from a remote device
- Send multicast data to a number of connected devices
- Automatic low power link handling

Note: The PAN profile is only available in the CSR Synergy Bluetooth HCI version.

2.2.8 Audio Video Profile

The AV profile implements the protocols and procedures required for the advanced audio streaming usage model [A2DP] and the Video Distribution Profile [VDP]. The profile supports both the sink and source role. The most common devices using this profile are mobile phones, computers, audio adaptors, headphones and speakers.

The scenarios covered by this profile are the following:

- Establish an audio stream (retrieving end-points and capabilities, configuring and opening stream)
- Transport encoded audio from source to sink
- Transport encoded video from source to sink
- Stream suspension, re-configuration and close of stream

Note: No codec for video distribution is currently included.

2.2.9 Audio Video Remote Control Profile

The AV Remote Control profile [AVRCP] implements the protocols and procedures required for the Audio/Video Remote Control profiles usage model. The profile supports both the Control and the Target role. The most common devices using this profile are the devices that also use the AV profile, such as mobile phones, computers, audio adaptors, headphones and speakers.

The scenarios covered by this profile are the following:

- Connection management
- Exchange of control commands and responses between Control and Target

2.2.10 SIM Access Profile

The SIM Access Profile [SAP] defines how a remote access to and control of a remote SIM Card can be achieved using Bluetooth®. The implementation of the SAP includes both the client and server part of the SAP usage model. The most common devices using this profile include mobile phones and car phones.

The following scenarios are available through the SAP API:

- Remote access and control of a SIM Card
- SAP connection handling

2.2.11 Human Interface Device Profile

The Human Interface Device [HID] Profile defines how a HID device can communicate with a HID host over the Bluetooth air-interface. The implementation includes both the host side (HIDH) and the device side (HIDD). Typical devices using the host side of this profile are computers and gaming consoles. Typical devices using the device side of this profile are keyboards, joysticks, game pads and remote controls.

The following scenarios are available through the HIDH:

- Controlling a host unit (computer, gaming console, etc.) with devices like keyboard, mouse, joystick etc.
- Collecting metrics from sensors, bar code scanners, etc. on a host unit

Note: The HIDH profile is only available in the CSR Synergy Bluetooth HCI version.

The following scenarios are available through the HIDD:

- HIDD connection handling
- HIDD power-saving facilities
- Re-establishing connection if connection is dropped
- Support for multiple instances of the profile

Note: The HIDD profile is only available in the CSR Synergy Bluetooth HCI version.

2.2.12 JSR-82 Java

The JSR-82 layer provides an interface for a JSR-82 Java class library. The implementation handles signals corresponding to the class member functions defined in the JSR-82 specification.

The JSR-82 layer provides for:

- Device inquiry and service discovery
- Create, register, and unregister service records
- Access to properties describing the local device
- Create and close RFCOMM and L2CAP connections
- Send and receive data

2.2.13 Hardcopy Cable Replacement Profile

The HCRP profile facilitates a low-level data transfer interface to hardcopy machines such as faxes, printers and hardcopy machines.

The HCRP profile requires a low-level hardcopy/printer driver to be functional. The profile provides for:

- IEEE 1284 printer id and status indications
- Raw data transfer of printer data

2.2.14 Security Controller

The Security Controller handles all functions related to and involving security functions. The Security Controller in CSR Synergy Bluetooth relies on the Security Manager (SM) part of the device manager in the Bluetooth® Core Stack.

The Security Controller provides for:

- Creation of security bond between two devices
- Pairing as part of connection establishment
- Retrieving passkeys from the application/user and sending to the SM
- Storing and retrieval of link keys in persistent storage

The Security Controller handles functions on request from the application layer or from the profile managers. The security requirements for the individual profile managers are registered in the Security Controller and forwarded to the SM in DM.

2.2.15 Service Discovery Module and Device Identification Profile

The Service Discovery module (SD) provides a simple interface for applications and profiles, wanting to search for information about remote Bluetooth® devices. This includes information such as: “remote name”, “class of device”, “Bluetooth® addresses”, “peer radio(s) Bluetooth® and/or LE radio”, “Bluetooth® addresses type (LE only)” and available services.

The Service Discovery module provides for:

- Search for remote devices
- Search agent, which automatically searches for remote devices
- Service search
- Listing of remote devices paired in the SC
- Implementation of the Device Identification Profile.

Because more than one application and/or profile might want to use the SD at the same time, the SD is designed to handle multiple instances at the same time.

2.2.16 Connection Manager

The Connection Manager (CM) is a service layer for the different profile layers. The Connection Manager is the layer in between the profile components and the Bluetooth® Core Stack. The CM is responsible for handling RFCOMM connections and L2CAP connections between a pair of devices. Using the CM it is possible to:

- Establish a RFCOMM connection between a pair of devices
- Transfer data between the devices
- Add SCO capabilities to a connection
- Establish a L2CAP connection

Further the CM contains functionality for SDP service search. If the profile layers wish to make use of low power modes, the Connection Manager supports this.

As multiple profile managers may be active at the same time, the CM is designed to be able to handle several simultaneous connection requests.

2.2.17 Generic Attribute Profile

The Generic Attribute Profile (GATT) provides interface and procedures to implement profiles running via GATT. All Bluetooth low energy (LE) profiles run via GATT but the usage of GATT is not limited to LE. GATT provides API for:

- Peripheral/Central/Observe(scan)/Broadcast(advertise)
- Create connections over LE or BR/EDR.
- Database – add/remove
- Discover Services – primary/secondary/included
- Read/write – values/descriptors
- Notification/indication
- Whitelist – add/remove/clear

The GATT profile layer support multiple connections.

2.2.18 Bluetooth Low Energy Production Test Support

Synergy Bluetooth supports the Bluetooth 4.0 low energy HCI production/radio test commands. The Connection Manager (CM) exposes the interface needed to run the tests – e.g. CsrBtCmLeTransmitterTestReqSend(). See the CM API documentation for further details.

3 CSR Synergy Bluetooth

3.1 Documentation Overview

All user documentation is located in the directory: **./doc**

3.1.1 CSR Synergy Bluetooth

This is the general information in how to use CSR Synergy Bluetooth:

File-name	Description
gu-0101-users_guide.pdf	This document
gu-0102-porting_guide.pdf	Porting guide. Description of what and how to port CSR Synergy Bluetooth to a platform
gu-0103-av_implementation.pdf	Guidelines in how to implement the AV profile application layer using the A2DP profile layer
srn-0101-release_note.pdf	Supported features, test evidence, qualification status and known issues

3.1.2 Profile Managers

Each profile has a document describing the Application Programming Interface (API). The API document includes message definitions and simple use case sequences of the API.

Component	File name
Connection Manager	api-0101-cm.pdf
Security Controller	api-0102-sc.pdf
Service Discovery module	api-0103-sd.pdf
Dial-Up Gateway	api-0104-dg.pdf
AT-interpreter	api-0105-dg_at.pdf
Dial-up Networking Profile Data Terminal	api-0106-dt.pdf
OBEX Push Server	api-0107-opps.pdf
OBEX Push Client	api-0108-oppc.pdf
OBEX Sync Server	api-0109-syncs.pdf
OBEX Sync Client	api-0110-syncc.pdf
Serial Port	api-0111-spp.pdf
Hands-Free	api-0113-hf.pdf
Hands-Free Gateway	api-0114-hfg.pdf
Telephony Control Specification (Cordless Telephony Profile and Intercom Profile)	api-0115-tcs.pdf
OBEX File Transfer Server	api-0116-ftps.pdf
OBEX File Transfer Client	api-0117-ftpc.pdf
Personal Area Networking	api-0118-pan.pdf
OBEX BIP Responder	api-0119-bips.pdf
OBEX BIP Initiator	api-0120-bipc.pdf
Audio Video	api-0121-av.pdf
Audio Video Remote Control	api-0122-avrcp.pdf
OBEX Basic Printing Profile Server	api-0124-bpps.pdf
OBEX Basic Printing Profile Client	api-0125-bppc.pdf
SIM Access Profile Server	api-0126-saps.pdf

Component	File name
SIM Access Profile Client	api-0127-spac.pdf
Human Interface Device Host	api-0128-hidh.pdf
Human Interface Device	api-0129-hidd.pdf
OBEX PBAP Server	api-0130-pbaps.pdf
OBEX PBAP Client	api-0131-pbapc.pdf
Hardcopy Cable Replacement Server	api-0132-hcrps.pdf
OBEX SyncML transfer Server	api-0133-syncml_server.pdf
OBEX SyncML transfer Client	api-0134-syncml_client.pdf
Java APIs for Bluetooth Wireless Technology (JSR-82)	api-0135-jsr82.pdf
Device Identification Profile	api-0136-di.pdf
Sub-Band Codec	api-0138-sbc.pdf
vCard API	api-0139-vcard.pdf
Resampling API	api-0140-resampling.pdf
Health Device Profile	api-0142-hdp.pdf
GATT – Generic Attribute Profile	api-0149-gatt.pdf

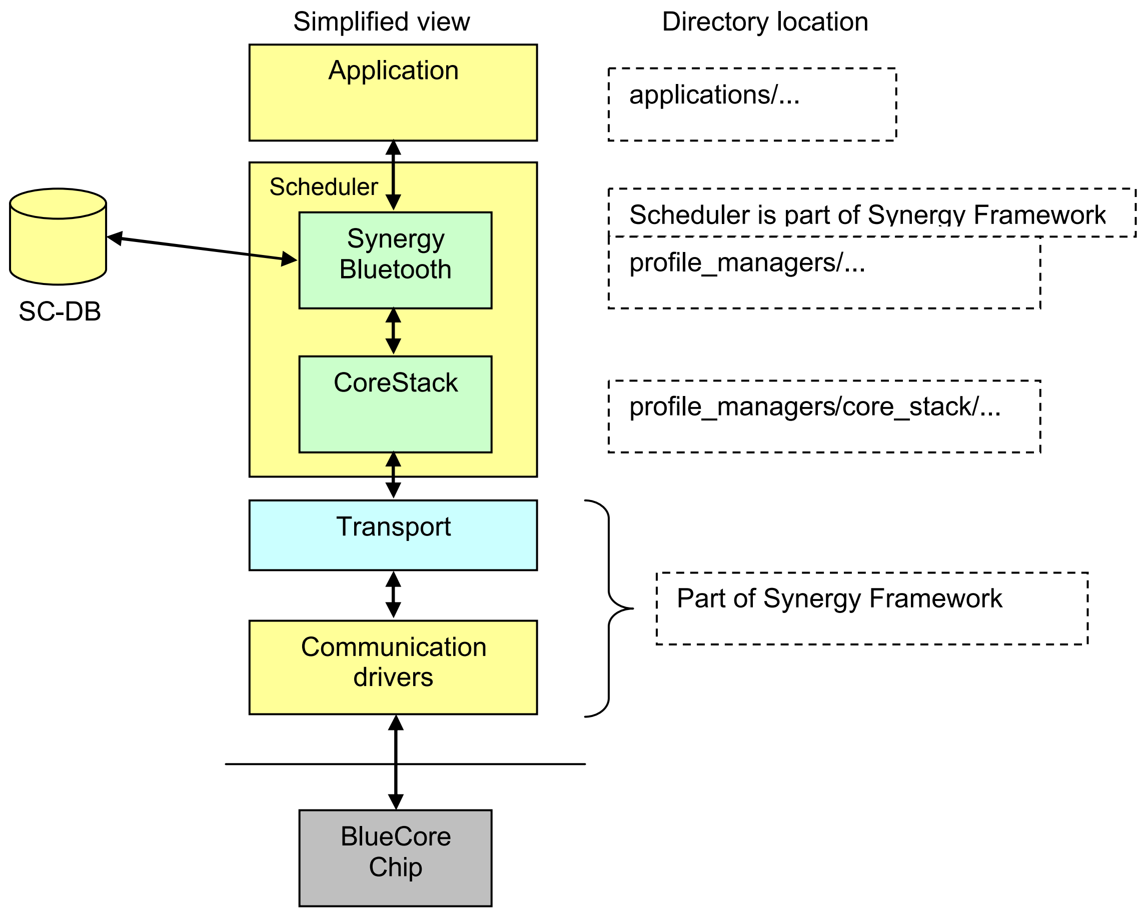
3.1.3 Test ICS

All relevant ICS documents are placed in the `./doc/ics` directory.

3.2 Directory Structure

The CSR Synergy Bluetooth directory structure is made with the following in mind:

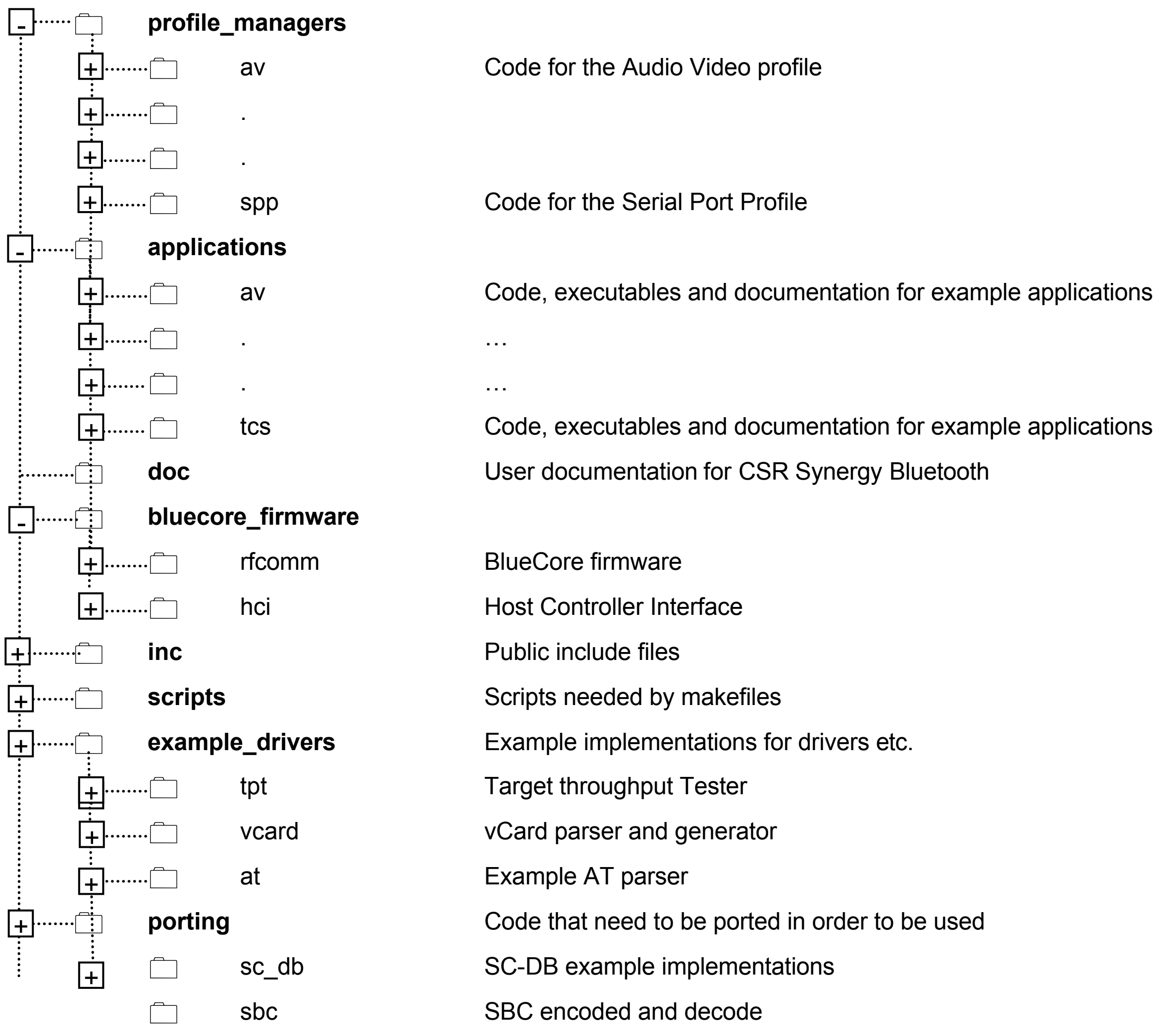
1. Resemble the CSR Synergy Bluetooth protocol stack layout and interaction with the outside world as much as possible (see Figure 3 for simplified figure of CSR Synergy Bluetooth and surrounding world).
2. "module" based for easy integration and use in customer applications.
3. Easy upgrade to new releases of CSR Synergy Bluetooth.



🟩	Part of synergy Bluetooth
🟦	Part of synergy Framework
🟨	Customer to port

Figure 3: Simplified view of CSR Synergy Bluetooth

The above three points as well as Figure 3 lead to the following directory structure:



profile_managers		
	av	Code for the Audio Video profile
	.	
	.	
	spp	Code for the Serial Port Profile
applications		
	av	Code, executables and documentation for example applications
	.	…
	.	…
	tcs	Code, executables and documentation for example applications
doc		User documentation for CSR Synergy Bluetooth
bluecore_firmware		
	rfcomm	BlueCore firmware
	hci	Host Controller Interface
inc		Public include files
scripts		Scripts needed by makefiles
example_drivers		Example implementations for drivers etc.
	tpt	Target throughput Tester
	vcard	vCard parser and generator
	at	Example AT parser
porting		Code that need to be ported in order to be used
	sc_db	SC-DB example implementations
	sbc	SBC encoded and decode

Figure 4: Directory structure

4 Building the CSR Synergy Bluetooth Libraries and Example Applications

This section describes the build system that is used for building the CSR Synergy Bluetooth libraries and the example applications.

The build system consists of a number of makefiles by which it is possible to build all the libraries that CSR Synergy Bluetooth has been divided into. The idea of using libraries instead of referring directly to C-files, is that it makes it easier to use and maintain.

4.1 Build System

The build system in CSR Synergy Bluetooth is based on the build system defined in the CSR Synergy Framework. Please refer to the CSR Synergy Framework Users [SYN-FWK-USERS-GUIDE] and CSR Synergy Framework Porting Guide [SYN-FWK-PORT-GUIDE] for more details about this.

CSR Synergy Bluetooth has been build and tested on the following architectures.

Build architecture	Target architecture	Compiler version
Cygwin-x86	winnt-x86	MSDEV compiler ver.: 12.00.8804
	arm	ADS ver. 1.2, 2.2, 3.0, 3.1 and 4.0
Linux-2.6-x86	Linux-2.6-x86	gcc version: 3.3.3 -> 4.0.0

Table 1: Build vs. target architecture and compiler versions

When using Cygwin as build environment and Microsoft Developer Studio it has been observed (depending on which packages having been installed in Cygwin and the setting of the PATH environment variable); that the Cygwin link-command may be used instead of the MSDEV linker, due to the fact that both are named “link.exe”.

This may be fixed by removing the Cygwin link.exe file or arranging the search path (PATH environment variable) so that the MSDEV link.exe is found first.

It may be checked by running the command:

```
~> which link
```

If something like “/cygdrive/c/Program Files/Microsoft Visual Studio/VC98/BIN/link” appears everything should be OK.

Another problem when using Cygwin as build environment is the find-command which in Cygwin is a “find files” and in Windows it is a “find in a file”. It is very important that the Cygwin binary directories are first in the search path. This may be checked by running the command:

```
~> echo $PATH
```

The libraries and binaries are built by the following command (stay in the CSR Synergy Bluetooth top directory, where the `makefile` is located):

```
make clean all TARGET=<ARCH> FW_ROOT=<path to CSR Synergy Framework>
```

where `<ARCH>` is one of the target architectures specified in the CSR Synergy Framework – upper- and lowercase letters are significant

If your target architecture is not defined by standard, please see CSR Synergy Framework for how to define new target architecture.

The following `make` target has been defined:

Target	Description
help	Print help
all	Make all libraries and demo applications
bin	Make all demo applications
lib	Make all libraries
clean	Remove all files generated by all, bin or lib
release	Creates a `setup.exe` file that is ready for installation. This is a binary release so some source directories will not be available. To build a binary release the following tools need to be installed: - NSIS version 2.34 or later (http://nsis.sourceforge.net) **The 'release' depends on the binaries to have been build prior to running the 'release' target, so please remember to run a 'make clean lib TARGET=<arch>' before running the 'make release'**

Table 2: `makefile` targets

4.1.1 How to handle Multiple Configurations of CSR Synergy Bluetooth at the same Time

In order to make it possible to use different configurations of CSR Synergy Bluetooth at the same time without having to duplicate the CSR Synergy Bluetooth source code, it is possible to compile CSR Synergy Bluetooth and give it a configuration file as one of the parameters.

The configuration is done in two files, namely `config-<config name>.mk` and `.h` file, both located in `./config`. The `config-<config name>.mk` file contain the compiler define settings and definition of the name of the config `.h` file, while the `.h` file contains the Bluetooth configuration.

The config format and values of the two files are described in the files.

To build with a specific configuration, do:

```
make clean all CONFIG=test
```

4.1.2 Building with Debug Information

If CSR Synergy Bluetooth is to be built with debug information the compiler define `DEBUG` is defines, as:

```
make clean all TARGET=<ARCH> FW_ROOT=<path to CSR Synergy Framework> DEBUG=1
```

This will make all libraries and binaries with debug information.

4.1.3 Building with Bluetooth low energy support

Bluetooth low energy (LE) support can be included by compiler define `CSR_BT_LE_ENABLE` as:

```
make clean all TARGET=<ARCH> FW_ROOT=<path to CSR Synergy Framework>
CSR_BT_LE_ENABLE=1
```

This will make all libraries and binaries with support for Bluetooth low energy for other configurations see section 5.3.2Bluetooth configuration.

4.1.4 Building with DSP manager support

In order to use the DSP manager component of the CSR Synergy Framework, the compilation of the CSR Synergy Bluetooth shall make use of the `CSR_DSPM_ENABLE` compiler definition as:

```
make clean all TARGET=<ARCH> FW_ROOT=<path to CSR Synergy Framework>
CSR_BT_DSPM_ENABLE=1
```

The framework version needed if this compiler definition is to be used shall be 3.1 or newer. The DSPM manager support is necessary for features like Wide-Band Speech in the Hands-free Profile. This will make all libraries and binaries with support for DSP manager for other configurations see section 5.3.2Bluetooth configuration. For more information about this compiler define and the Wide-Band Speech feature, please consult the Hands-Free profile API document.

Note: In order to use the DSP on-chip and make use of the Wide-Band Speech feature, the bootstrap needs tweaking. The MAP_SCO_PCM field of the bootstrap has to be set to 0. This field is set to 1 per default and needs to be 1 if the on-chip DSP is not going to be used. The field PSKEY_RESERVE_INIT_TOKENS (address 0x2171) has to be added with a value of 3.

4.2 CSR Synergy Bluetooth Libraries

A number of `makefiles` have been made in order to be able to build the libraries that constitute the CSR Synergy Bluetooth software.

As described in the [SYN-BT-PORT] some functions needs to be ported to the platform, like persistent memory etc. for which example implementations for windows has been included in the CSR Synergy Bluetooth release.

This mean that when CSR Synergy Bluetooth is build two different “types” of libraries will be build namely the libraries that include the Bluetooth functionality and the libraries that includes the functions that needs to be ported.

In Table 3 and Table 4 these two set of libraries are listed, with a description of the contents of the libraries.

CSR Synergy Bluetooth protocol stack library name	Description
csr_bt	CSR Synergy Bluetooth profile managers. If not all profile managers are needed it is possible to select individual profiles. These libraries are placed in the `extra` folder in the `output` directory.
csr_bt_lib	Interface functions to CSR Synergy Bluetooth profile managers for sending messages into the CSR Synergy Bluetooth protocol stack. If not all profile managers are needed it is possible to select individual profiles. These libraries are placed in the `extra` folder in the `output` directory.
csr_bt_corestack_hci	The CSR Synergy Bluetooth Corestack compiled for a HCI build.
csr_bt_hidparser	A HID parser for parsing HID events and HID descriptors.
csr_bt_msg_converter	Converter functions used for serializing messages. Used when messages are to be sent over an interface that does not support pointer passing.

Table 3: Libraries constituting CSR Synergy Bluetooth

CSR Synergy Bluetooth porting library name	Description
csr_bt_sc_db-file	Security Controller Database implementation based on the database being stored in a file
csr_bt_sc_db	Security Controller Database implementation based on the database being stored in memory. Database entries will be destroyed when the application exits
csr_sbc	Sub Band Coding implementation for use in the AV demo application. Different variants exists optimized for a specific platform. E.g. ARM9 and ARM11

Table 4: CSR Synergy Bluetooth protocol stack library name

In order to build an application one has to select which libraries to use for the applications and also select transport protocol to chip (serial or USB). The transport types are gotten from the CSR Synergy Framework.

As described in Table 3 there are two different ways to “select” which profiles to use, namely:

- Compile all profiles into separate libraries and link with the wanted libraries
- Compile one library with all profiles and ensure that only the needed profiles are part of this library

A number of compiler defines have been defined (see 5.3.1) to define which profiles that are used and which are not used.

If a whole profile is not to be used the way to exclude it is to not include the files constituting the profile in to build system, but it is still important to set the EXCLUDE_CSR_BT_XXX_MODULE defines as some common code may be removed if certain profiles are not to be used.

Please do not rely on the use of the EXCLUDE_CSR_BT_XXX_MODULE compiler defines to remove whole profiles

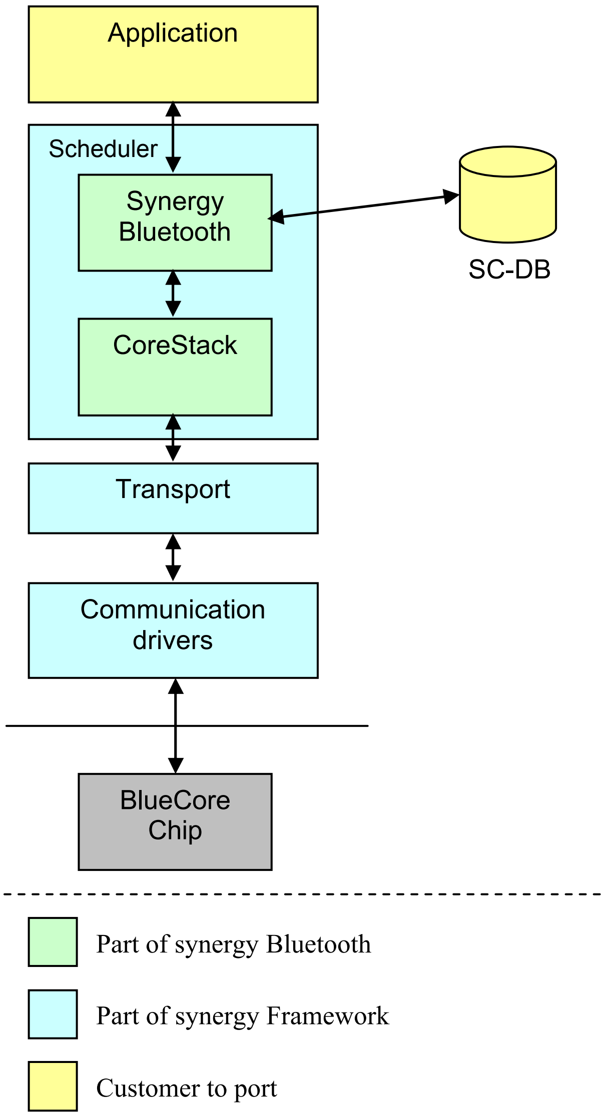


Figure 5: Simplified component view

4.3 CSR Synergy Bluetooth Example Applications

To show how to use the different profiles a number of example applications have been made and shipped with CSR Synergy Bluetooth.

All example applications are running on Windows and most has been ported to Linux also.

All example applications are located in `./applications`.

Note: The windows example application projects are setup to use the debug version of the libraries, see 0 for how to build applications.

4.3.1 Building Example Applications

The example applications may be built from the top CSR Synergy Bluetooth directory by doing:

```
make clean all FW_ROOT=<path to CSR Synergy Framework>
```

or in each example application directory, like:

```
      make clean all FW_ROOT=<path to CSR Synergy Framework>
```

This will build the example application with support for BCSP, H4DS (serial connection) and USB connection to a Casira.

Common to all example applications is that the executable file contains both the application and the CSR Synergy Bluetooth protocol stack.

For a description of the individual example applications, please see the documentation for each example application.

If the demo applications are to be built with debug information the following is done:

```
      make clean all FW_ROOT=<path to CSR Synergy Framework> DEBUG=1
```

5 Bringing it all together

In 4.1 it was described how the libraries that constitute CSR Synergy Bluetooth were built. This section describes how to use the libraries and what is needed in order to make an "application". For examples of this, please look in the `./applications` directory.

Below the different elements in setting up an application are listed:

1. CSR Synergy tasks to use
    - CSR Synergy Bluetooth
    - CSR Synergy Framework
2. Libraries to use
    - Interface libraries
    - Profile libraries
    - Ported code
        - CSR Synergy Framework
        - CSR Synergy Bluetooth ported interfaces
            - Persistent memory
3. Configuration
    - Build
    - Bluetooth

In the next sections theses element will be described in more details.

5.1 CSR Synergy Bluetooth Interface Functions

To make the interface to CSR Synergy Bluetooth as easy as possible all CSR Synergy Bluetooth profile managers have an API for sending messages into the CSR Synergy Bluetooth profile manager. That is, for every message defined in the API documentation in the direction towards CSR Synergy Bluetooth, there is a corresponding function that takes the message members as parameters, allocates and fills the message structure and sends the message to the appropriate CSR Synergy Bluetooth task. Any message member defined as a pointer is expected to be allocated as dynamic block using the CsrP`malloc()`. After a pointer has been given to the message function it is the responsibility of the receiving task to free (`CsrPfree()`) the allocated memory. This also applies if the application constructs the message itself. This scheme applies to all pointers passed in messages unless specified otherwise in the API documentation.

The message send functions use the following naming convention: The function name is the same as the message name without the underscores and with "Send" appended. For example, the CSR_BT_DG_ACTIVATE_REQ message send function is called *CsrBtDgActivateReqSend()*. The message send function declarations can be found in the `xxx_lib.h` files in the main "inc" directory.

5.2 CSR Synergy Tasks and Libraries to use

In order to get the wanted functionality of the CSR Synergy Bluetooth system it is necessary that the correct tasks and modules are used. Two different components are involved in setting up a CSR Synergy Bluetooth system, namely the CSR Synergy Framework and the CSR Synergy Bluetooth modules.

When defining a CSR Synergy Bluetooth system, the following tasks always need to be present:

CSR Synergy Component	Sub-component	Library	Lib functions
CSR Synergy Bluetooth	CM	csr_bt	csr_bt_libs
	SC	csr_bt	csr_bt_libs
	SD	csr_bt	csr_bt_libs
	DM	corestack	Not Applicable
	L2CAP	corestack	Not Applicable
	RFCOMM	corestack	Not Applicable
	DM_HCI	corestack	Not Applicable
	SDP_L2CAP	corestack	Not Applicable
	SDP	corestack	Not Applicable
CSR Synergy Bluetooth Porting	Persistent Memory	-	-
	Audio Interface	-	-
CSR Synergy Framework	BCCMD	csr_cmdif_bccmd	
	HQ	csr_cmdif_hq	
	Transport(USB,BCPS,H4DS,Type-A	csr_bcsp and csr_ser_com if BCSP serial communication is selected csr_h4ds and csr_ser_com_h4ds if H4DS serial communication is selected csr_usb_com if USB communication is selected csr_type_aif Type-A communication is selected	Not Applicable
	TM	csr_tm_bluecore	
	Scheduler	-	-

Table 5 CSR Synergy Bluetooth components and libraries to use

Besides the components listen in Table 5 the profiles needed in the final product is also needed to be included.

5.3 Configuration

Two levels of configuration may be done, namely:

1. CSR Synergy Bluetooth build configuration, like whether or not a specific module is include or code has been removed in some files to save space if a certain feature is not used.

2. CSR Synergy Bluetooth configuration, like maximum packet size etc.

Which configuration to use is selected as described in section 4.1.1.

5.3.1 Build Configuration

The following compiler defies are available when using the build system part of CSR Synergy Bluetooth.

The compiler defines marked with **bold** MUST always be defined in order to make the code work.

A set of compiler flags have been introduced to enable further control of the feature/code-size trade-off, these have been named *_OPTIONAL, and in general these flags will disable features considered optional, in the return of a decrease in code size. For the profiles, optional features cover some or all of the features marked with optional in the BT specification, in addition to utility functions not used by the synergy BT code, if such exist. For the common profiles CM, SD and SC the _OPTIONAL will reduce the API to what is needed by the profiles enabled, and finally a flag exist to remove utility functions not needed by the enabled profiles.

Compiler switch	Description
EXCLUDE_MBLK_BUFFER_SUPPORT	**Always needs to be defined when building the framework.**
AV_STREAM_INACTIVITY_LP_ENABLE	Enables detection of inactive AV streams (no stream data sent/received) and that low power mode (sniff) is requested for such streams upon detection. When defined it is possible to set inactive period (AV_LP_TIMEOUT) in usr_config.h. This should not be enabled if ENABLE_AV_ROUTER_SUPPORT is also enabled.
EDR_DISABLE	Disable support for EDR by modifying packet sizes. EDR requires at least a BT 2.0 controller. Default is that EDR is enabled
EXCLUDE_CSR_BT_AVRCP_MODULE	Exclude the Audio/Video Remote Control module
EXCLUDE_CSR_BT_AVRCP_CT_MODULE	Exclude the Controller side of AVRCP in order to save code space if this is not needed
EXCLUDE_CSR_BT_AVRCP_TG_MODULE	Exclude the Target side of AVRCP in order to save code space if this is not needed
EXCLUDE_CSR_BT_AVRCP_MODULE_OPTIONAL	Have been introduced to reduce code size, by disabling AVRCP 1.3 and 1.4 features, and corresponding API/LIB functions.
EXCLUDE_CSR_BT_AT_MODULE	Exclude the modem AT-Command module
EXCLUDE_CSR_BT_AV_MODULE	Exclude the Audio Video module
EXCLUDE_CSR_BT_AV_OPTIONAL	Exclude optional AV parameters[1]: • CsrBtAvSetQosIntervalReqSend()
EXCLUDE_CSR_BT_BPPS_MODULE	Exclude the Basic Printing server module

[1] Default values can be specified in csr_bt_usr_config.h

Compiler switch	Description
EXCLUDE_CSR_BT_BPPS_MODULE_OPTIONAL	Exclude optional BPPS features[2]: • Client side of the obex get operation • The obex description header • The obex authentication challenge/response headers
EXCLUDE_CSR_BT_BPPC_MODULE	Exclude the Basic Printing client module
EXCLUDE_CSR_BT_BPPC_MODULE_OPTIONAL	Exclude optional BPPC features[2]: • Client side of the obex get operation • The obex description header • The obex application parameters header
EXCLUDE_CSR_BT_BIPC_MODULE	Exclude the Basic Image Push Initiator module
EXCLUDE_CSR_BT_BIPS_MODULE	Exclude the Basic Image Push Responder module
EXCLUDE_CSR_BT_BSL_MODULE	Exclude the Personal Area Networking module
EXCLUDE_CSR_BT_DG_MODULE	Exclude the Dun Gateway module
EXCLUDE_CSR_BT_DUNC_MODULE	Exclude the DUN-DT profile module
EXCLUDE_CSR_BT_FTC_MODULE	Exclude the File Transfer Client module
EXCLUDE_CSR_BT_FTC_MODULE_OPTIONAL	Exclude optional FTC features[2]: • The obex description header
EXCLUDE_CSR_BT_FTS_MODULE	Exclude the File Transfer Server module
EXCLUDE_CSR_BT_FTS_MODULE_OPTIONAL	Exclude optional FTS features[2]: • The obex description header
EXCLUDE_CSR_BT_HCRP_MODULE	Exclude the Hardcopy Cable Replacement module
EXCLUDE_CSR_BT_HDP_MODULE	Exclude the Health Device Profile module
EXCLUDE_CSR_BT_HF_MODULE	Exclude the Hands-Free module
EXCLUDE_CSR_BT_HF_MODULE_OPTIONAL	Exclude optional HF features[2]: • CsrBtHfAudioConfigReq • Park mode
EXCLUDE_CSR_BT_HFG_MODULE	Exclude the Hands-Free Gateway module
EXCLUDE_CSR_BT_HFG_MODULE_OPTIONAL	Exclude optional HF features[2]: • CsrBtHfgAudioConfigReq • CSR to CSR features • CsrBtHfgConfigSingleAtcmdReq • CsrBtHfgConfigAtcmdHandlingReq • Park mode
EXCLUDE_CSR_BT_HID_PARSER_MODULE	Exclude the HID Parser module
EXCLUDE_CSR_BT_HIDD_MODULE	Exclude the HID Device Profile module
EXCLUDE_CSR_BT_HIDH_MODULE	Exclude the HID Host Profile module

[2] Unless they are needed by other enabled modules

Compiler switch	Description
EXCLUDE_CSR_BT_IWU_MODULE	Exclude the IWU application module (on top of DG)
EXCLUDE_CSR_BT_JSR82_MODULE	Exclude the JSR-82 module
EXCLUDE_CSR_BT_MAPC_MODULE	Exclude the Message Access Profile Client module
EXCLUDE_CSR_BT_MAPS_MODULE	Exclude the Message Access Profile Server module
EXCLUDE_CSR_BT_MCAP_MODULE	Exclude the Multi Channel Adaption Profile module. This module needs to be include when the HDP profile is used.
EXCLUDE_CSR_BT_OPC_MODULE	Exclude the OBEX push client profile module
EXCLUDE_CSR_BT_OPC_MODULE_OPTIONAL	Exclude optional OPC features[2]: • Client side of the obex get operation • The obex description header
EXCLUDE_CSR_BT_OPS_MODULE	Exclude the OBEX push server profile module
EXCLUDE_CSR_BT_OPS_MODULE_OPTIONAL	Exclude optional OPC features[2]: • Server side of the obex get operation, The obex get operation is mandatory to respond to, hence when setting this flag a NOT_FOUND response will be automatically generated. • The obex description header
EXCLUDE_CSR_BT_PAC_MODULE	Exclude the Phone Book Access Client module
EXCLUDE_CSR_BT_PAS_MODULE	Exclude the Phone Book Access Server module
EXCLUDE_CSR_BT_SAPC_MODULE	Exclude the SIM Access Profile Client module
EXCLUDE_CSR_BT_SAPS_MODULE	Exclude the SIM Access Profile Server module
EXCLUDE_CSR_BT_SCO_MODULE	Exclude all SCO related code. This flag will be set automatically if none of the enabled profiles needs the SCO-Module.
EXCLUDE_CSR_BT_SD_SERVICE_RECORD_MODULE	Excludes the Device Identification code embedded within the Service Discovery module. Exclude this to save code space if it is not need. *Note:* **HID and HDP needs the DI in order to be able to run** If EXCLUDE_CSR_BT_SD_MODULE_OPTIONAL is defined, this flag will automatically be set if neither HID nor HDP is enabled.
EXCLUDE_CSR_BT_SMLC_MODULE	Exclude the OBEX SyncML Transfer Client module
EXCLUDE_CSR_BT_SMLC_MODULE_OPTIONAL	Exclude optional SMLC features[2]: • The obex description header
EXCLUDE_CSR_BT_SMLS_MODULE	Exclude the OBEX SyncML Transfer Server module
EXCLUDE_CSR_BT_SMLS_MODULE_OPTIONAL	Exclude optional SMLS features[2]: • The obex description header
EXCLUDE_CSR_BT_SPP_MODULE	Exclude the Serial Port Profile module
EXCLUDE_CSR_BT_SPP_MODULE_OPTIONAL	Exclude optional SPP features: • 128 bit service search • SCO renegotiation
EXCLUDE_CSR_BT_SYNCC_MODULE	Exclude the Synchronization client module

Compiler switch	Description
EXCLUDE_CSR_BT_SYNCC_MODULE_OPTIONAL	Exclude optional SYNCC features[2]: • The obex description header
EXCLUDE_CSR_BT_SYNCS_MODULE	Exclude the Synchronization server module
EXCLUDE_CSR_BT_SYNCS_MODULE_OPTIONAL	Exclude optional SYNCS features[2]: • The obex description header
EXCLUDE_CSR_BT_PROX_SRV_MODULE	Exclude the LE Proximity Server. Only applicable if building with CSR_BT_LE_ENABLE=1
EXCLUDE_CSR_BT_THERM_SRV_MODULE	Exclude the LE Health Thermometer Server. Only applicable if building with CSR_BT_LE_ENABLE=1
EXCLUDE_CSR_BT_TEST_MODULE	Exclude the TEST module, also used for demo application tasks
EXCLUDE_CSR_EXCEPTION_HANDLER_MODULE	Enable handling of exception. What to do when an exception occur is depended of the port.
EXCLUDE_CSR_BT_SC_MODULE_OPTIONAL	Exclude optional SC features, not needed by any of the enabled profiles.
EXCLUDE_CSR_BT_SD_MODULE_OPTIONAL	Exclude optional SD features, not needed by any of the enabled profiles. Including the SD_SERVICE_RECORD_MODULE unless HID or HDP is enabled. To avoid excluding the SD_SERVICE_RECORD_MODULE simply add the compiler flag INSTALL_CSR_BT_SD_SERVICE_RECORD_MODULE
EXCLUDE_CSR_BT_CM_MODULE_OPTIONAL	Exclude the majority of CM API functions not used by the enabled profiles. If only a few of the functions are needed, they can be enabled individually by specifying the corresponding CSR_BT_INSTALL_CM_* flag. E.g. to enable the Read RSSI API functionality, use the flag CSR_BT_INSTALL_CM_READ_RSSI. Please refer to Appendix A.
EXCLUDE_CSR_BT_OPTIONAL_UTILS	Exclude Utility functions not used by the stack.

Table 6 Compiler defines to set in CFLAGS or EXTRA_CFLAGS

Build options	Description
DEBUG (default=1)	Defines whether or not to build with debug enabled. Usages: DEBUG=1 enables debug
CSR_BT_BLUE_STACK_DEBUG (default=1)	Defines whether or not to build with core stack debug features enabled. usages CSR_BT_BLUE_STACK_DEBUG
ENABLE_SHUTDOWN (default=1)	Include code (the deinit functions) for graceful shutdown in Profile Managers. This will ▪ ensure that no more messages are put on the task input queues ▪ ensure that no more timed functions are pending

Build options	Description
CSR_BT_LE_ENABLE (default=0)	Include code for Bluetooth low energy (LE) support ▪ Lib: corestack – Inclusion of task ATT and extensions to DM and L2CAP ▪ Lib: csr_bt – Inclusion of task GATT and extensions to SD, SC and CM
CSR_CHIP_MANAGER_ENABLE (default=0)	Enables chip manager support for recovery of chip resets. Upon a chip reset, the state of the chip will be automatically re-established except for any active connections. Applications with active connections are informed with disconnect indication and should handle it similar to an abnormal link loss.
CSR_BT_CONFIG_L2CAP_FCS (default=0)	If 0, then the L2CAP connection will not be reconfigured when a connection is moved from BT to WiFi. If 1, reconfiguration of L2CAP connections is enabled.
CSR_AMP_ENABLE (default=0)	Enable AMP support WIFI_ROOT=<path-to-wifi>

Table 7 Makefile options to which can be enabled. E.g. "make xxx CSR_AMP_ENABLE=1"

5.3.2 Bluetooth configuration

The configuration is done in two files, namely `config-<config name>.mk` and a `.h` file, both located in `./config`.

The `config-<config name>.mk` file contain the compiler define settings and definition of the name of the config `.h` file, while the `.h` file contains the Bluetooth configuration.

Please look in `./config/csr_bt_usr_config_default.h` for description of all the CSR Synergy Bluetooth configuration parameters.

Appendix A Optional CM API functions

This appendix lists the optional CM API functions removed when compiling with the EXCLUDE_CSR_BT_CM_MODULE_OPTIONAL compiler flag, unless they are used by the enabled profiles. The table below shows the flag used to protect the individual API function/feature, as well as the corresponding signal name/feature name.

Signal name	Compiler flag name
CSR_BT_CM_READ_LOCAL_NAME_REQ	CSR_BT_INSTALL_CM_READ_LOCAL_NAME
CSR_BT_CM_WRITE_LINK_SUPERV_TIMEOUT_REQ	CSR_BT_INSTALL_CM_WRITE_LINK_SUPERVISION_TIMEOUT
CSR_BT_CM_READ_TX_POWER_LEVEL_REQ	CSR_BT_INSTALL_CM_READ_TX_POWER_LEVEL
CSR_BT_CM_GET_LINK_QUALITY_REQ	CSR_BT_INSTALL_CM_GET_LINK_QUALITY
CSR_BT_CM_READ_RSSI_REQ	CSR_BT_INSTALL_CM_READ_RSSI
CSR_BT_CM_WRITE_COD_REQ	CSR_BT_INSTALL_CM_WRITE_COD
CSR_BT_CM_READ_COD_REQ	CSR_BT_INSTALL_CM_READ_COD
CSR_BT_CM_READ_SCAN_ENABLE_REQ	CSR_BT_INSTALL_CM_READ_SCAN_EANBLE
CSR_BT_CM_SDC_UUID128_SEARCH_REQ	CSR_BT_INSTALL_128_BIT_SERVICE_SEARCH
CSR_BT_CM_CONNECTABLE_REQ	CSR_BT_INSTALL_CM_CONNECTABLE
CSR_BT_CM_WRITE_PAGE_TO_REQ	CSR_BT_INSTALL_CM_WRITE_PAGE_TO
CSR_BT_CM_WRITE_PAGESCAN_SETTINGS_REQ CSR_BT_CM_WRITE_PAGESCAN_TYPE_REQ	CSR_BT_INSTALL_CM_WRITE_PAGE_SCAN
CSR_BT_CM_WRITE_INQUIRYSCAN_TYPE_REQ	CSR_BT_INSTALL_CM_WRITE_INQUIRY_SCAN_TYPE
CSR_BT_CM_READ_LOCAL_EXT_FEATURES_REQ	CSR_BT_INSTALL_CM_READ_LOCAL_EXT_FEATURES
CSR_BT_CM_SET_AFH_CHANNEL_CLASS_REQ CSR_BT_CM_READ_AFH_CHANNEL_ASSESSMENT_MODE_REQ CSR_BT_CM_WRITE_AFH_CHANNEL_ASSESSMENT_MODE_REQ CSR_BT_CM_READ_AFH_CHANNEL_MAP_REQ	CSR_BT_INSTALL_CM_AFH
CSR_BT_CM_READ_CLOCK_REQ	CSR_BT_INSTALL_CM_READ_CLOCK
CSR_BT_CM_WRITE_LINK_POLICY_REQ CSR_BT_CM_READ_LINK_POLICY_REQ	CSR_BT_INSTALL_CM_LINK_POLICY

CSR_BT_CM_EIR_FLAGS_REQ	CSR_BT_INSTALL_CM_EIR_FLAGS
CSR_BT_CM_ROLE_SWITCH_CONFIG_REQ	CSR_BT_INSTALL_CM_ROLE_SWITCH_CONFIG
CSR_BT_CM_READ_FAILED_CONTACT_COUNTER_REQ	CSR_BT_INSTALL_CM_READ_FAILED_CONTACT_COUNTER
CSR_BT_CM_SWITCH_ROLE_REQ	CSR_BT_INSTALL_CM_SWITCH_ROLE_PUBLIC
Mode change related functions: CsrBtCmSniffModeReqSend, CsrBtCmExitSniffModeReqSend and CsrBtCmModeChangeConfigReqSend	CSR_BT_INSTALL_CM_LOW_POWER_CONFIG_PUBLIC
Enable support for park mode	CSR_BT_INSTALL_CM_PARK_MODE
The following flags each enables an event type which can be subscribed to by a CSR_BT_CM_SET_EVENT_MASK_REQ: • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_BLUECORE_INITIALIZED • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_ACL_CONNECTION • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_SYNCHRONOUS_CONNECTION • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_ROLE_CHANGE • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_MODE_CHANGE • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_LSTO_CHANGE • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_CHANNEL_TYPE • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_EXT_SYNC_CONNECTION • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_REMOTE_FEATURES • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_REMOTE_VERSION • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_A2DP_BIT_RATE • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_INQUIRY_PAGE_STATE • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_LOW_ENERGY • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_ENCRYPT_CHANGE • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_LOCAL_NAME_CHANGE • CSR_BT_INSTALL_CM_EVENT_MASK_SUBSCRIBE_HIGH_PRIORITY_DATA	

6 Document References

Document	Reference
BC Command API, document number api-0003-bccmd	[BCCMD]
Bluetooth® Core Specification, profile section N/A, version 1.2, 2.0, 2.1, 3.0 and 4.0	[BT]
Device Identification Profile, 26. july 2007, version 1.3	[DI]
Hardcopy Cable Replacement Profile Specification, version 1.0a	[HCRP]
Java™ APIs for Bluetooth™ Wireless Technology (JSR-82), version 1.0a	[JSR-82]
The Bluetooth® Specification, Phone Book Access Profile, version 1.0	[PBAP]
The Bluetooth® Specification, the Advanced Audio Distribution Profile, version 1.0	[A2DP]
The Bluetooth® Specification, the Audio/Video Control Transport Protocol Specification, version 1.0	[AVCTP]
The Bluetooth® Specification, the Audio/Video Distribution Transport Protocol, version 1.0	[AVDTP]
The Bluetooth® Specification, the Audio/Video Remote Control Profile, version 1.0	[AVRCP]
The Bluetooth® Specification, the Basic Imaging Profile, version 1.0	[BIP]
The Bluetooth® Specification, the Basic Printing Profile	[BPP]
The Bluetooth® Specification, the Cordless Telephony Profile. Profile section K:3, version 1.1	[CTP]
The Bluetooth® Specification, the Dial-Up Network Profile. Profile section K:7, version 1.1	[DUN]
The Bluetooth® Specification, the File Transfer Profile. Profile section K:12, version 1.1	[FTP]
The Bluetooth® Specification, the Generic Audio/Video Distribution Profile, version 1.0	[GAVDP]
The Bluetooth® Specification, the Hands-free Gateway Profile	[HFG]
The Bluetooth® Specification, the Headset Profile, profile section K:6, version 1.1	[HSP]
The Bluetooth® Specification, the Human Interface Device Profile, version 1.0	[HID]
The Bluetooth® Specification, the Intercom Profile. Profile section K:4, version 1.1	[ICP]
The Bluetooth® Specification, the Object Push Profile. Profile section K:11, version 1.1	[OPP]
The Bluetooth® Specification, the Personal Area Networking Profile	[PAN]
The Bluetooth® Specification, the Serial Port Profile, profile section K:5, version 1.1	[SPP]
The Bluetooth® Specification, the SIM Access Profile	[SAP]
The Bluetooth® Specification, the Synchronisation Profile. Profile section K:13, version 1.1	[SP]

Document	Reference
The Bluetooth® Specification, the Telephony Control Protocol Profile. Profile section F:3, version 1.1	[TCS]
The Bluetooth® Specification, the Video Distribution Profile, version 1.0	[VDP]
The SyncML Specification, SyncML OBEX Binding, version 1.1	[SMLP]
SDIO Card Part E2, Type-A specification for Bluetooth, version 1.00, September 2002	[TYPE-A]
The Bluetooth® Specification, Generic Attribute Profile, Host Volumen Part G	[GATT]
The Bluetooth® Specification, Attribute Protocol, Host Volumen Part F	[ATT]
The Bluetooth® Specification, Security Manager Protocol, Host Volumen Part H	[SMP]

Terms and Definitions

Abbreviation	Explanation
A2D	Advanced Audio Distribution
API	Application Programming Interface
AT-Interpreter	Asynchronous Transfer Interpreter
ATT	Attribute Protocol
AV	Audio Video
AVRCP	Audio Video Remote Control Profile
BC01	BlueCore01, CSR's Bluetooth® chip
BC02	BlueCore02, CSR's Bluetooth® chip
BC2-ROM	BlueCore2-ROM, CSR's Bluetooth® chip
BCSP	BlueCore® Serial Protocol
BIP	Basic Imaging Profile
BIPC	OBEX Basic Imaging Profile (client)
BIPS	OBEX Basic Imaging Profile (server)
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
BNEP	Bluetooth Network Encapsulation Protocol
BPP	Basic Printing Profile
CM	Connection Manager
CSR	Cambridge Silicon Radio
CTP	Cordless Telephony Profile
DB	Data Base
DG	Dial-up network Gateway
DI	Device Identification Profile
DM	Device Manager
DT	Data Terminal
DUN	Dial-Up Network
DUNC	Dial-Up Network Client
FTC	File Transfer Client
FTP	File Transfer Protocol
FTS	File Transfer Server
GAP	Generic Access Profile
GATT	Generic Attribute Profile
GW	GateWay
H4DS	H4 Deep Sleep
HCI	Host Controller Interface
HCRP	Hardcopy Replacement Profile
HF	Hands-Free
HFG	Hands-Free Gateway
HIDD	Human Interface Device Device
HIDH	Human Interface Device Host
HQ	Host Query

Abbreviation	Explanation
HSP	HeadSet Profile
ICP	Intercom Profile
JSR-82	Java Specification Request
LC	Link Controller
LE	Bluetooth low energy
LM	Link Manager
MMI	Man Machine Interface
OBEX	OBject EXchange
OS	Operating System
PAC	Phone Book Access Profile Client
PAN	Personal Area Networking
PAS	Phone Book Access Profile Server
PBAP	Phone Book Access Profile
PDA	Personal Digital Assistant
PPP	Point to Point Protocol
RFCOMM	RF Communication Protocol
SAP	SIM Access Profile
SCO	Synchronous Connection Oriented
SDAP	Service Discovery Application Profile
SDP	Service Discovery Protocol
SIM	Subscriber Identity Module
SM	Security Manager
SMLC	OBEX SyncML transfer (client)
SMLP	SyncML OBEX binding profile
SMLS	OBEX SyncML transfer (server)
SMP	Security Manager Profile (LE)
SPP	Serial Port Profile
TCS	Telephony Control Specification
UART	Universal Asynchronous Receiver Transmitter
UniFi™	Group term for CSR's range of chips designed to meet IEEE 802.11 standards
USB	Universal Serial Bus
VDP	Video Distribution Profile

Document History

Revision	Date	History
1	26 SEP 11	Ready for release 18.2.0
2	12 DEC 11	Ready for release 18.2.1

TradeMarks, Patents and Licences

Life Support Policy and Use in Safety-critical Compliance

Performance and Conformance

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