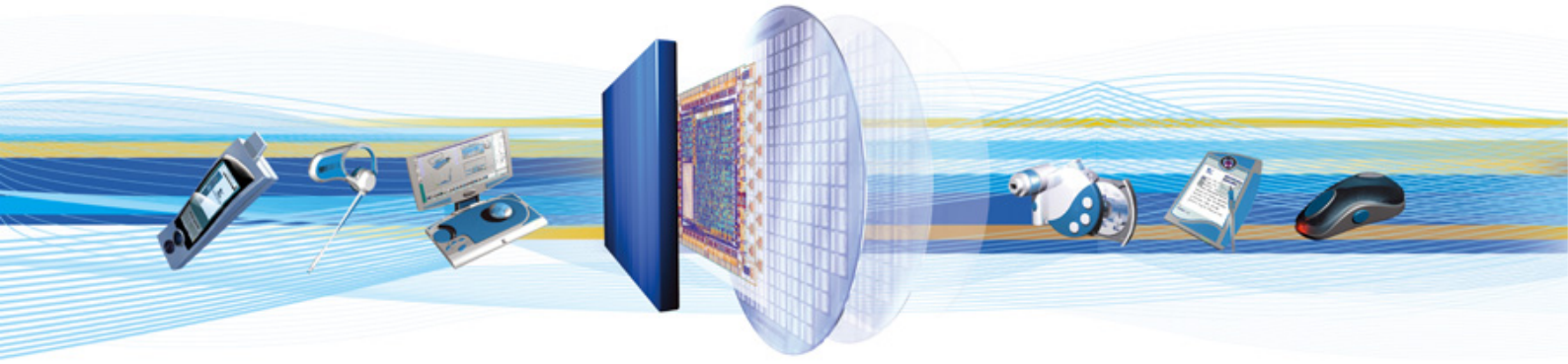
CSR Synergy Framework 3.1.2

FastPipe

API Description

January 2012



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

List of Figures

List of Tables

1 Introduction

1.1 Introduction and Scope

This document describes the Application Programming Interface (API) to the Fastpipe (FP) Host Software component.

The FastPipe interface provides functionality and message interface for fast data transfer to and from a BlueCore® chip via UART interface. FastPipe allows the host to send data to BlueCore using fast, flow controlled, channels that coexist with the HCI channels. The data is sent over ACL channels that FastPipe owns. The BlueCore has hardware acceleration for data RX and TX on ACL channels, which is why the data is sent this way. The FastPipe protocol is not part of the Bluetooth standard and its intended usage is related to other technologies than Bluetooth which might require fast flow controlled data transfers with the BlueCore.

2 Description

2.1 Reference model

Figure 1 show how FP fit into the general Host architecture.

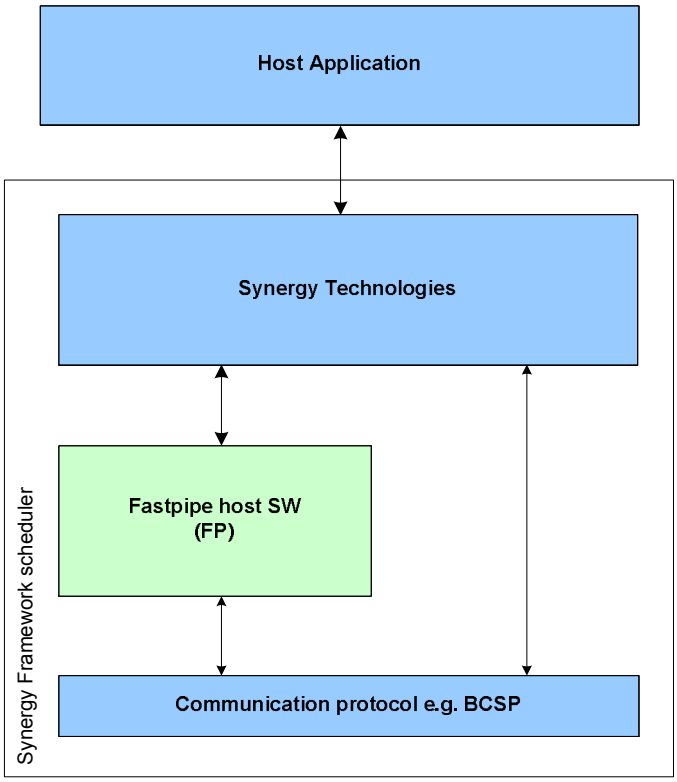


Figure 1: Host architecture

3 Interface Description

If CSR_BLUECORE_ONOFF is defined, any outstanding requests will be discarded when the BlueCore enters the deactivating state, and any requests that are received during the deactivating state will be discarded and the requesting tasks will not receive a confirm message. When this happens, all existing pipes will be lost, and the tasks that created these should no longer reference them. When the BlueCore is in the deactivated or activating state, CSR_FP_CREATE_REQ messages will be queued and processed when the BlueCore enters the active state.

3.1 CSR_FP_CREATE

The first message between the application and FP is the CSR_FP_CREATE message request. The confirm message result code and the FP handle. The FP handle are used for indentify the pipe and are used for all other operations. There need to be VM code (code running on the controller) to setup the mapping between the FP and the controllers target application.

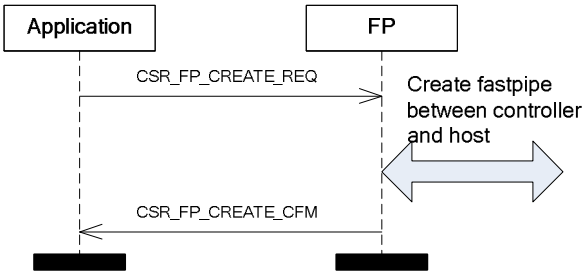


Figure 2: CSR_FP_CREATE

3.2 CSR_FP_WRITE

Write data from the host to the controller. The application is allowed to have one outstanding write confirm. So the application has to wait on the CSR_FP_WRITE_CFM before it call the next CSR_FP_WRITE_REQ.

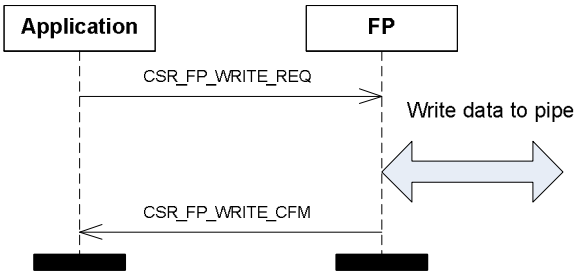


Figure 3: CSR_FP_WRITE

3.3 CSR_FP_READ

Read data from the controller. When there is data from the controller, FP will send a CSR_FP_READ_IND to the host application. You have to create a FP before you can receive data from the controller.

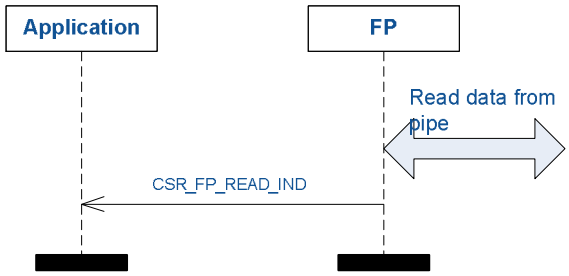


Figure 4: CSR_FP_READ

3.4 CSR_FP_CLEAR

Clear all data on host side. NB: A write data packet that has been partly transferred will not be cleared.

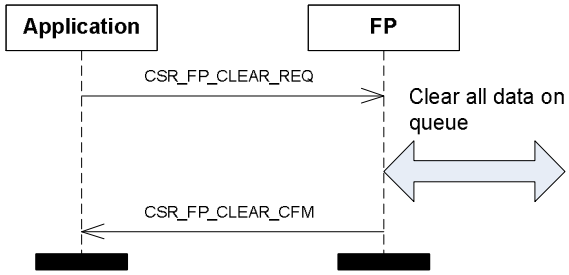


Figure 5: CSR_FP_CLEAR

3.5 CSR_FP_DESTROY

CSR_FP_DESTROY will clear and destroy a FP. A write packet, which is in progress of being sent through the pipe, will be send completely before the FP is cleared and destroyed.

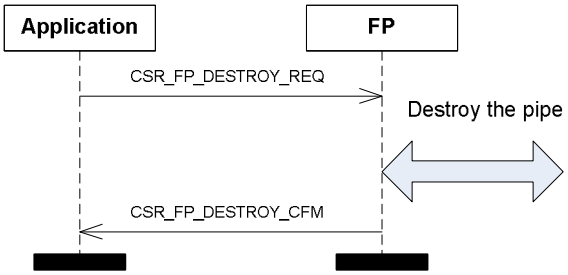


Figure 6: CSR_FP_DESTROY

4 Interface Description

The FP host software API uses a message based API, based on the CSR Synergy Framework scheduler (see [SYN-FRW-COAL-API] for more information). The scheduler splits the host software into tasks that each contains a message queue and a message handler. Several API functions allow messages to be placed in each task queue.

The following sections describe the available API between the applications on the host and the FP host software.

4.1 Primitives

Primitive	Reference
CSR_FP_CREATE	See section 4.2
CSR_FP_WRITE	See section 4.3
CSR_FP_CLEAR	See section 4.4
CSR_FP_DESTROY	See section 4.5
CSR_FP_READ	See section 4.6

4.2 CSR_FP_CREATE_REQ/CFM

Parameters / Primitives	type	appHandle	overheadHost	capacityRxHost	requiredTxController	desiredTxController	requiredRxController	desiredRxController	fpHandle	result	overheadController	capacityTxController	capacityRxController
CSR_FP_CREATE_REQ	✓	✓	✓	✓	✓	✓	✓	✓					
CSR_FP_CREATE_CFM	✓								✓	✓	✓	✓	✓

Table 1: CSR_FP_CREATE primitives

Description
This primitive is used for creating a data pipe.

Parameters

type	Signal identity, CSR_FP_CREATE_REQ/CFM
appHandle	Application handle – identifier for the application queue to which the message response should be returned.
overheadHost	Pipe overhead on the host
capacityRxHost	Capacity of receive buffer on the host
requiredTxController	Required capacity of tx buffer on controller
desiredTxController	Desired capacity of tx buffer on controller
requiredRxController	Required capacity of rx buffer on controller
desiredRxController	Desired capacity of rx buffer on controller
result	Zero indicates success and non-zero indicates an error code. See csr_fp_prim.h
fpHandle	Fastpipe handle in the range 1..15. 0 is used for credit channel values in the range 16..255 are reserved
overheadController	Pipe overhead on the controller
capacityTxController	Capacity of transmit buffer on controller
capacityRxController	Capacity of receive buffer on controller

Function prototype
The following function is used for constructing and sending this primitive:

```
CsrFpCreateReqSend (CsrFpAppHandleType appHandle,
                    CsrUint32 overheadHost,
                    CsrUint32 capacityRxHost,
                    CsrUint32 requiredTxController,
                    CsrUint32 desiredTxController,
                    CsrUint32 requiredRxController,
                    CsrUint32 desiredRxController);
```

4.3 CSR_FP_WRITE_REQ/CFM

Parameters / Primitives	type	fpHandle	payloadLength	payload	result
CSR_FP_WRITE_REQ	✓	✓	✓	✓	
CSR_FP_WRITE_CFM	✓	✓			✓

Table 2: CSR_FP_WRITE primitives

Description
This primitive is used for sending data to the controller.

Parameters

type — Signal identity, CSR_FP_WRITE_REQ/CFM

fpHandle — Fastpipe handle in the range 1..15.
0 is used for credit channel
values in the range 16..255 are reserved

payloadLength — Length of data in number of octets

payload — Pointer reference to the actual payload; this is of type mblk

result — Zero indicates success and non-zero indicates an error code. See csr_fp_prim.h

Function prototype
The following function is used for constructing and sending this primitive:

```
CsrFpWriteReqSend (CsrFpHandleType fpHandle,
                   CsrUint16 payloadLength,
                   CsrMblk *payload);
```

4.4 CSR_FP_CLEAR_REQ/CFM

Primitives \ Parameters	Type	fpHandle	result
CSR_FP_CLEAR_REQ	✓	✓	
CSR_FP_CLEAR_CFM	✓	✓	✓

Table 3: CSR_FP_CLEAR primitives

Description
This primitive is used for clearing the pipe for data.

Parameters

type — Signal identity, CSR_FP_CLEAR_REQ/CFM

fpHandle — Fastpipe handle in the range 1..15.
0 is used for credit channel
values in the range 16..255 are reserved

result — Zero indicates success and non-zero indicates an error code. See csr_fp_prim.h

Function prototype
The following function is used for constructing and sending this primitive:

```
CsrFpClearReqSend (CsrFpHandleType fpHandle);
```

4.5 CSR_FP_DESTROY_REQ/CFM

Primitives \ Parameters	Type	fpHandle	result
CSR_FP_DESTROY_REQ	✓	✓	
CSR_FP_DESTROY_CFM	✓	✓	✓

Table 4: CSR_FP_DESTROY primitives

Description
This primitive is used for clearing the pipe for data.

Parameters

type — Signal identity, CSR_FP_CLEAR_REQ/CFM

fpHandle — Fastpipe handle in the range 1..15.
0 is used for credit channel
values in the range 16..255 are reserved

result — Zero indicates success and non-zero indicates an error code. See csr_fp_prim.h.

Function prototype
The following function is used for constructing and sending this primitive:

```
CsrFpDestroyReqSend (CsrFpHandleType fpHandle);
```

4.6 CSR_FP_READ_IND

Parameters / Primitives	Type	fpHandle	payload
CSR_FP_READ_IND	✓	✓	✓

Table 5: CSR_FP_READ primitive

Description
This primitive is used for reading data from the controller.

Parameters

type	Signal identity, CSR_FP_READ_IND
fpHandle	Fastpipe handle in the range 1..15. 0 is used for credit channel values in the range 16..255 are reserved
payload	Pointer reference to the actual payload; this is of type mblk

Function prototype
-

5 Document References

Document	Reference
[SYN-FRW-COAL-API]	CSR Synergy Framework COAL API. Doc. api-0004-coal

Terms and Definitions

BlueCore®	Group term for CSR's range of Bluetooth® chips.
CSR	Cambridge Silicon Radio
API	Application Programming Interface
FP	Fastpipe

Document History

Revision	Date	History
1	13 JUL 09	Ready for release 1.1.0
2	30 NOV 09	Ready for release 2.0.0
3	20 APR 10	Ready for release 2.1.0
4	OCT 10	Ready for release 2.2.0
5	DEC 10	Ready for release 3.0.0
6	Aug 11	Ready for release 3.1.0

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.

No statements or representations in this document are to be construed as advertising, marketing, or offering for sale in the United States imported covered products subject to the Cease and Desist Order issued by the U.S. International Trade Commission in its Investigation No. 337-TA-602. Such products include SiRFstarIII™ chips that operate with SiRF software that supports SiRFInstantFix™, and/or SiRFLoc® servers, or contains SyncFreeNav functionality.

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.