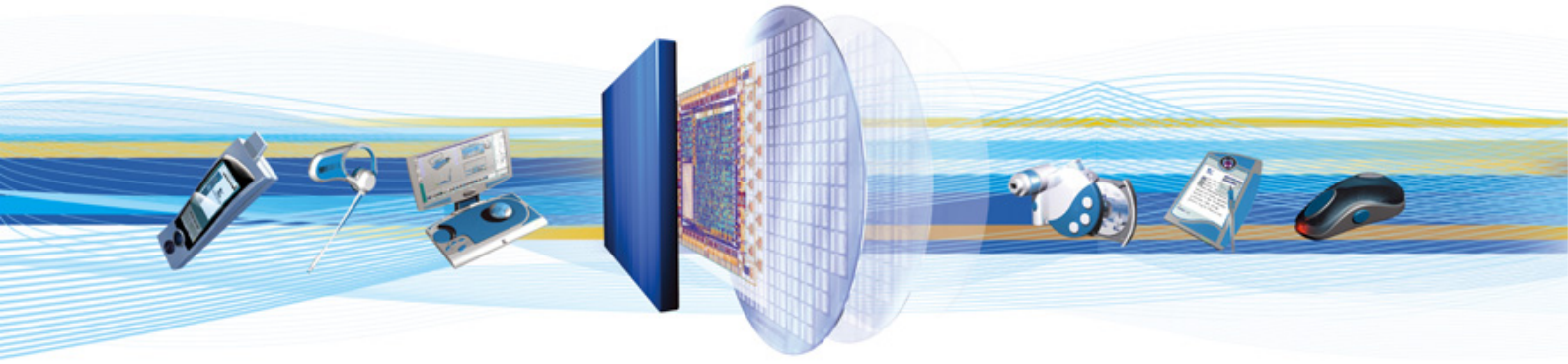
CSR Synergy Framework 3.1.0

Panic

API Description

August 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 Introduction

The panic handling API is used by e.g. transport drivers to signal that something unexpected has occurred. The abnormal behaviour is unrecoverable by any Framework module since the proper action to take may depend on the application or platform itself. A panic may result in a system restart – but how to do this properly is unknown to the Framework. The panic interface thus may either be implemented directly in the BSP code if the semantics does not rely on components outside the Framework. Or, it may e.g. be implemented by calling back to a registered panic handler. This external handler may e.g. be part of the application code.

The panic handling API must be provided in the file `csr_panic.h`.

2 Panic API

2.1 Basic Types

The panic interface contains defines that identify the Synergy Technology types that are able to cause panics. In addition, defines that identify the possible panic reason from Framework components. The defines must all be placed in the file `csr_panic.h`.

2.1.1 Panic Identifier Types

The panic defines are as follows.

Prototype

```
#include "csr_panic.h"

/* Synergy technology ID definitions */
#define CSR_TECH_FW     0
#define CSR_TECH_BT     1
#define CSR_TECH_WIFI   2
#define CSR_TECH_GPS    3
#define CSR_TECH_NFC    4

/* Panic type ID definitions for technology type CSR_TECH_FW */
#define CSR_PANIC_FW_UNEXPECTED_VALUE        0
#define CSR_PANIC_FW_HEAP_EXHAUSTION         1
#define CSR_PANIC_FW_INVALID_PFREE_POINTER   2
#define CSR_PANIC_FW_EXCEPTION               3
#define CSR_PANIC_FW_ASSERTION_FAIL          4
#define CSR_PANIC_FW_NULL_TASK_HANDLER       5
#define CSR_PANIC_FW_UNKNOWN_TASK            6
#define CSR_PANIC_FW_QUEUE_ACCESS_VIOLATION  7
#define CSR_PANIC_FW_TOO_MANY_MESSAGES       8
#define CSR_PANIC_FW_TOO_MANY_TIMED_EVENTS   9
#define CSR_PANIC_FW_ABCSP_SYNC_LOST        10
#define CSR_PANIC_FW_OVERSIZE_ABCSP_PRIM    11
#define CSR_PANIC_FW_H4_CORRUPTION          12
#define CSR_PANIC_FW_H4_SYNC_LOST           13
#define CSR_PANIC_FW_H4_RX_OVERRUN          14
#define CSR_PANIC_FW_H4_TX_OVERRUN          15
#define CSR_PANIC_FW_TM_BC_RESTART_FAIL     16
#define CSR_PANIC_FW_TM_BC_START_FAIL       17
#define CSR_PANIC_FW_TM_BC_BAD_STATE        18
#define CSR_PANIC_FW_TM_BC_TRANSPORT_LOST   19
```

2.2 CsrPanic

Prototype

```
#include "csr_panic.h"

void CsrPanic(CsrUint8 tech, CsrUint16 reason, const char *p);
```

Description

CsrPanic() is called when an uncorrectable condition is detected in which it is impossible to continue without potentially causing the entire system to crash. It signals a panic and halts execution in some way that **guarantees that the function does not return**. Doing this is system dependant, but it may involve disabling interrupts and entering an endless loop or signalling the operating system to terminate our process or thread.

The Synergy Technology raising the panic is identified by the `tech` parameter. The `reason` may be one of a predefined set of reasons. For the Framework the reason types are presented in Section 2.1.1. The `p` parameter is a string describing the panic cause.

Parameters

Type	Argument	Description
`CsrUint8`	`tech`	Synergy Technology identifier.
`CsrUint16`	`reason`	Panic reason type.
`const char*`	`p`	String describing panic cause.

Returns

None.

3 Panic Handling Reference

3.1 Introduction

The CSR Synergy Framework reference ports contain an interface for initialization of the implemented panic handling system. This interface is not to be used by any Synergy generic components, and thus is not required to be ported. This section presents this reference interface.

Note: The reference API is contained in file `csr_panic_init.h` in the special include directory `$(BSP_ROOT)/inc/platform`. Interfaces in this folder may be platform dependent and they are not necessarily available with any BSP. Thus, the CSR Synergy Framework does not mandate porting of these interfaces. Source code for the reference implementation is contained in file `csr_panic.c` in the `$(BSP_ROOT)/src/coal` directory.

3.2 Basic Types

The panic handler reference interface uses a panic handler function call back type. This type is described below.

3.2.1 Panic Handler Type

Typedef

```
typedef void (*CsrPanicHandler) (CsrUint8 tech, CsrUint16 reason, const char *p);
```

Description

Call back type matching exactly the `CsrPanic()` interface.

3.3 CsrPanicInit

Prototype

```
void CsrPanicInit(CsrPanicHandler cb);
```

Description

Initialize the panic handling reference implementation by registering an external panic handler call back. Applications using the reference ports must implement their own panic handler and register this with the `CsrPanicInit()` function.

Parameters

Type	Argument	Description
`CsrPanicHandler`	`cb`	Panic handler call back.

Returns

None.

4 Document References

Ref	Title

Terms and Definitions

Abbreviation	Explanation
CSR	Cambridge Silicon Radio

Document History

Revision	Date	History
1	19 Oct 09	First draft version
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.