SMART ARM-based Microcontrollers

AT12198: SAM C Brown Out Detector (BOD) Driver

APPLICATION NOTE

Introduction

This driver for Atmel® | SMART ARM®-based microcontrollers provides an interface for the configuration and management of the device's Brown Out Detector (BOD) modules, to detect and respond to under-voltage events and take an appropriate action.

The following peripherals are used by this module:

- SUPC (Supply Controller)

The following devices can use this module:

- Atmel | SMART SAM C20/C21

The outline of this documentation is as follows:

Table of Contents

1. Software License

2. Prerequisites

There are no prerequisites for this module.

3. Module Overview

The SAM devices contain a number of Brown Out Detector (BOD) modules. Each BOD monitors the supply voltage for any dips that go below the set threshold for the module. In case of a BOD detection the BOD will either reset the system or raise a hardware interrupt so that a safe power-down sequence can be attempted.

4. Special Considerations

The time between a BOD interrupt being raised and a failure of the processor to continue executing (in the case of a core power failure) is system specific; care must be taken that all critical BOD detection events can complete within the amount of time available.

5. Extra Information

For extra information, see Extra Information for BOD Driver. This includes:

- Acronyms
- Dependencies
- Errata
- Module History

6. Examples

For a list of examples related to this driver, see Examples for BOD Driver.

7. API Overview

7.1. Structure Definitions

7.1.1. Struct bodvdd_config

Configuration structure for a BODVDD module.

Table 7-1. Members

Type	Name	Description
enum bodvdd_action	action	Action to perform when a low power detection is made
bool	hysteresis	If `true`, enables detection hysteresis
uint8_t	level	BODVDD level to trigger at when monitors VDD except in backup sleep mode
enum bodvdd_mode_in_active	mode_in_active	BODVDD configuration in active mode
enum bodvdd_mode_in_standby	mode_in_standby	BODVDD configuration in backup sleep mode
enum bodvdd_prescale	prescaler	Input sampler clock prescaler factor, to reduce the 1kHz clock from the ULP32K to lower the sampling rate of the BODVDD
bool	run_in_standby	If `true`, the BODVDD is kept enabled and sampled during standby

7.2. Function Definitions

7.2.1. Configuration and Initialization

7.2.1.1. Function bodvdd_get_config_defaults()

Get default BODVDD configuration.

```
void bodvdd_get_config_defaults(
        struct bodvdd_config *const conf)
```

The default BODVDD configuration is:

- Clock prescaler set to divide the input clock by two
- Continuous in active mode
- Continuous in standby mode
- Reset on BODVDD detect
- Hysteresis enabled
- BODVDD level 42 on V_{DD}
- BODVDD kept enabled during standby

Table 7-2. Parameters

Data direction	Parameter name	Description
[out]	conf	BODVDD configuration struct to set to default settings

7.2.1.2. Function bodvdd_set_config()

Configure a Brown Out Detector module.

```
enum status_code bodvdd_set_config(
        struct bodvdd_config *const conf)
```

Configures a given BOD module with the settings stored in the given configuration structure.

Table 7-3. Parameters

Data direction	Parameter name	Description
[in]	conf	Configuration settings to use for the specified BODVDD

Table 7-4. Return Values

Return value	Description
STATUS_OK	Operation completed successfully
STATUS_ERR_INVALID_ARG	An invalid BOD was supplied
STATUS_ERR_INVALID_OPTION	The requested BOD level was outside the acceptable range

7.2.1.3. Function bodvdd_enable()

Enables a configured BODVDD module.

```
enum status_code bodvdd_enable( void )
```

Enables the BODVDD module that has been previously configured.

Returns
Error code indicating the status of the enable operation.

Table 7-5. Return Values

Return value	Description
STATUS_OK	If the BODVDD was successfully enabled

7.2.1.4. Function bodvdd_disable()

Disables an enabled BODVDD module.

```
enum status_code bodvdd_disable( void )
```

Disables the BODVDD module that was previously enabled.

Returns
Error code indicating the status of the disable operation.

Table 7-6. Return Values

Return value	Description
STATUS_OK	If the BODVDD was successfully disabled

7.2.1.5. Function bodvdd_is_detected()

Checks if the BODVDD low voltage detection has occurred.

```
bool bodvdd_is_detected( void )
```

Determines if the BODVDD has detected a voltage lower than its configured threshold.

Returns
Detection status of the BODVDD.

Table 7-7. Return Values

Return value	Description
true	If the BODVDD has detected a low voltage condition
false	If the BODVDD has not detected a low voltage condition

7.2.1.6. Function bodvdd_clear_detected()

Clears the low voltage detection state of the BODVDD.

```
void bodvdd_clear_detected( void )
```

Clears the low voltage condition of the BODVDD module, so that new low voltage conditions can be detected.

7.3. Enumeration Definitions

7.3.1. Enum bodvdd_action

List of possible BODVDD actions when a BODVDD module detects a brown-out condition.

Table 7-8. Members

Enum value	Description
BODVDD_ACTION_NONE	A BODVDD detect will do nothing, and the BODVDD state must be polled
BODVDD_ACTION_RESET	A BODVDD detect will reset the device
BODVDD_ACTION_INTERRUPT	A BODVDD detect will fire an interrupt
BODVDD_ACTION_BACKUP	A BODVDD detect will put the device in backup sleep mode

7.3.2. Enum bodvdd_mode_in_active

List of possible BODVDD module voltage sampling modes in active sleep mode.

Table 7-9. Members

Enum value	Description
BODVDD_ACTCFG_CONTINUOUS	BODVDD will sample the supply line continuously
BODVDD_ACTCFG_SAMPLED	BODVDD will use the BODVDD sampling clock (1kHz) to sample the supply line

7.3.3. Enum bodvdd_mode_in_standby

List of possible BODVDD module voltage sampling modes in standby sleep mode.

Table 7-10. Members

Enum value	Description
BODVDD_STDBYCFG_CONTINUOUS	BODVDD will sample the supply line continuously
BODVDD_STDBYCFG_SAMPLED	BODVDD will use the BODVDD sampling clock (1kHz) to sample the supply line

7.3.4. Enum bodvdd_prescale

List of possible BODVDD controller prescaler values, to reduce the sampling speed of a BODVDD to lower the power consumption.

Table 7-11. Members

Enum value	Description
BODVDD_PRESCALE_DIV_2	Divide input prescaler clock by 2
BODVDD_PRESCALE_DIV_4	Divide input prescaler clock by 4
BODVDD_PRESCALE_DIV_8	Divide input prescaler clock by 8
BODVDD_PRESCALE_DIV_16	Divide input prescaler clock by 16
BODVDD_PRESCALE_DIV_32	Divide input prescaler clock by 32
BODVDD_PRESCALE_DIV_64	Divide input prescaler clock by 64
BODVDD_PRESCALE_DIV_128	Divide input prescaler clock by 128
BODVDD_PRESCALE_DIV_256	Divide input prescaler clock by 256
BODVDD_PRESCALE_DIV_512	Divide input prescaler clock by 512
BODVDD_PRESCALE_DIV_1024	Divide input prescaler clock by 1024
BODVDD_PRESCALE_DIV_2048	Divide input prescaler clock by 2048
BODVDD_PRESCALE_DIV_4096	Divide input prescaler clock by 4096
BODVDD_PRESCALE_DIV_8192	Divide input prescaler clock by 8192
BODVDD_PRESCALE_DIV_16384	Divide input prescaler clock by 16384

Enum value	Description
BODVDD_PRESCALE_DIV_32768	Divide input prescaler clock by 32768
BODVDD_PRESCALE_DIV_65536	Divide input prescaler clock by 65536

8. Extra Information for BOD Driver

8.1. Acronyms

Below is a table listing the acronyms used in this module, along with their intended meanings.

Acronym	Definition
BOD	Brown Out Detector

8.2. Dependencies

This driver has the following dependencies:

- None

8.3. Errata

There are no errata related to this driver.

8.4. Module History

An overview of the module history is presented in the table below, with details on the enhancements and fixes made to the module since its first release. The current version of this corresponds to the newest version in the table.

Changelog
Initial Release

9. Examples for BOD Driver

This is a list of the available Quick Start guides (QSGs) and example applications for SAM Brown Out Detector (BOD) Driver. QSGs are simple examples with step-by-step instructions to configure and use this driver in a selection of use cases. Note that a QSG can be compiled as a standalone application or be added to the user application.

- Quick Start Guide for BOD - Basic
- Application Use Case for BOD - Application

9.1. Quick Start Guide for BOD - Basic

In this use case, the BODVDD will be configured with the following settings:

- Continuous sampling mode
- Prescaler setting of 2
- Reset action on low voltage detect

9.1.1. Quick Start

9.1.1.1. Prerequisites

There are no special setup requirements for this use-case.

9.1.1.2. Code

Copy-paste the following setup code to your user application:

```
static void configure_bodvdd(void)
{
    struct bodvdd_config config_bodvdd;
    bodvdd_get_config_defaults(&config_bodvdd);

    bodvdd_set_config(&config_bodvdd);

    bodvdd_enable();
}
```

Add to user application initialization (typically the start of `main()`):

```
configure_bodvdd();
```

9.1.1.3. Workflow

1. Create a BODVDD module configuration struct, which can be filled out to adjust the configuration of a physical BOD peripheral.

   ```
   struct bodvdd_config config_bodvdd;
   ```

2. Initialize the BODVDD configuration struct with the module's default values.

   ```
   bodvdd_get_config_defaults(&config_bodvdd);
   ```

   Note: This should always be performed before using the configuration struct to ensure that all values are initialized to known default settings.

3. Configure the BODVDD module with the desired settings.

   ```
   bodvdd_set_config(&config_bodvdd);
   ```

4. Enable the BODVDD module so that it will monitor the power supply voltage.

```
bodvdd_enable();
```

9.1.2. Use Case

9.1.2.1. Code

Copy-paste the following code to your user application:

```
while (true) {
     /* Infinite loop */
}
```

9.1.2.2. Workflow

1. Enter an infinite loop so that the BOD can continue to monitor the supply voltage level.

```
while (true) {
     /* Infinite loop */
}
```

9.2. Application Use Case for BOD - Application

The preferred method of setting BODVDD levels and settings is trough the fuses. When it is desirable to set it in software, see the below use case.

In this use case, a new BODVDD level might be set in SW if the clock settings are adjusted after a battery has charged to a higher level. When the battery discharges, the chip will reset when the battery level is below the SW BODVDD level. Now the chip will run at a lower clock rate and the BODVDD level from fuse. The chip should always measure the voltage before adjusting the frequency up.

10. Document Revision History

Doc. Rev.	Date	Comments
42701A	08/2016	Initial document release

Atmel | Enabling Unlimited Possibilities®

Atmel Corporation 1600 Technology Drive, San Jose, CA 95110 USA **T:** (+1)(408) 441.0311 **F:** (+1)(408) 436.4200 | **www.atmel.com**