SMART ARM-based Microcontrollers

AT11512: SAM L 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 peripheral is used by this module:

- SUPC (Supply Controller)

The following devices can use this module:

- Atmel | SMART SAM L21/L22

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 bod12_config

Configuration structure for a BOD12 module.

Table 7-1. Members

Type	Name	Description
enum bod12_action	action	Action to perform when a low power detection is made
bool	hysteresis	If `true`, enables detection hysteresis
uint8_t	level	BOD12 level to trigger at (see electrical section of device datasheet)
enum bod12_mode_in_active	mode_in_active	BOD12 configuration in active mode
enum bod12_mode_in_standby	mode_in_standby	BOD12 configuration in backup sleep mode
enum bod12_prescale	prescaler	Input sampler clock prescaler factor, to reduce the 1KHz clock from the ULP32K to lower the sampling rate of the BOD12
bool	run_in_standby	If `true`, the BOD12 is kept enabled and sampled during device sleep

7.1.2. Struct bod33_config

Configuration structure for a BOD33 module.

Table 7-2. Members

Type	Name	Description
enum bod33_action	action	Action to perform when a low power detection is made
uint8_t	backuplevel	BOD33 level to trigger at when monitors VBAT or in backup sleep mode
bool	hysteresis	If `true`, enables detection hysteresis
uint8_t	level	BOD33 level to trigger at when monitors VDD excpt in backup sleep mode
enum bod33_mode_in_active	mode_in_active	BOD33 configuration in active mode
enum bod33_mode_in_standby	mode_in_standby	BOD33 configuration in backup sleep mode

Type	Name	Description
enum bod33_vol_monitor	monitor	Voltage monitored in active and standby mode
enum bod33_prescale	prescaler	Input sampler clock prescaler factor, to reduce the 1KHz clock from the ULP32K to lower the sampling rate of the BOD33
bool	run_in_backup	If `true`, the BOD33 is kept enabled and sampled during device sleep
bool	run_in_standby	If `true`, the BOD33 is kept enabled and sampled during standby

7.2. Function Definitions

7.2.1. Configuration and Initialization

7.2.1.1. Function bod33_get_config_defaults()

Get default BOD33 configuration.

```
void bod33_get_config_defaults(
        struct bod33_config *const conf)
```

The default BOD33 configuration is:

- Clock prescaler set to divide the input clock by two
- Continuous in active mode
- Continuous in standby mode
- Monitor the V_{DD} power pin
- No action on BOD33 detect
- Hysteresis enabled
- BOD33 level 0x7 on V_{DD}
- BOD33 level 0x7 on V_{BAT}
- BOD33 kept enabled during device sleep
- BOD33 kept enabled during standby

Table 7-3. Parameters

Data direction	Parameter name	Description
[out]	conf	BOD33 configuration struct to set to default settings

7.2.1.2. Function bod33_set_config()

Configure a Brown Out Detector module.

```
enum status_code bod33_set_config(
        struct bod33_config *const conf)
```

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

Table 7-4. Parameters

Data direction	Parameter name	Description
[in]	conf	Configuration settings to use for the specified BOD33

Table 7-5. 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 bod33_enable()

Enables a configured BOD33 module.

```
enum status_code bod33_enable( void )
```

Enables the BOD33 module that has been previously configured.

Returns
Error code indicating the status of the enable operation.

Table 7-6. Return Values

Return value	Description
STATUS_OK	If the BOD33 was successfully enabled

7.2.1.4. Function bod33_disable()

Disables an enabled BOD33 module.

```
enum status_code bod33_disable( void )
```

Disables the BOD33 module that was previously enabled.

Returns
Error code indicating the status of the disable operation.

Table 7-7. Return Values

Return value	Description
STATUS_OK	If the BOD33 was successfully disabled

7.2.1.5. Function bod33_is_detected()

Checks if the BOD33 low voltage detection has occurred.

```
bool bod33_is_detected( void )
```

Determines if the BOD33 has detected a voltage lower than its configured threshold.

Returns
Detection status of the BOD33.

Table 7-8. Return Values

Return value	Description
true	If the BOD33 has detected a low voltage condition
false	If the BOD33 has not detected a low voltage condition

7.2.1.6. Function bod33_clear_detected()

Clears the low voltage detection state of the BOD33.

```
void bod33_clear_detected( void )
```

Clears the low voltage condition of BOD33 module, so that new low voltage conditions can be detected.

7.2.1.7. Function bod12_get_config_defaults()

Get default BOD12 configuration.

```
void bod12_get_config_defaults(
        struct bod12_config *const conf)
```

The default BOD12 configuration is:

- Clock prescaler set to divide the input clock by two
- Continuous in active mode
- Continuous in standby mode
- Reset on BOD12 detect
- Hysteresis enabled
- BOD12 level 0x12
- BOD12 kept enabled during device sleep

Table 7-9. Parameters

Data direction	Parameter name	Description
[out]	conf	BOD12 configuration struct to set to default settings

7.2.1.8. Function bod12_set_config()

Configure a Brown Out Detector module.

```
enum status_code bod12_set_config(
        struct bod12_config *const conf)
```

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

Table 7-10. Parameters

Data direction	Parameter name	Description
[in]	conf	Configuration settings to use for the specified BOD12

Table 7-11. 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

7.2.1.9. Function bod12_enable()

Enables a configured BOD12 module.

```
enum status_code bod12_enable( void )
```

Enables the BOD12 module that has been previously configured.

Returns
Error code indicating the status of the enable operation.

Table 7-12. Return Values

Return value	Description
STATUS_OK	If the BOD12 was successfully enabled

7.2.1.10. Function bod12_disable()

Disables an enabled BOD12 module.

```
enum status_code bod12_disable( void )
```

Disables the BOD12 module that was previously enabled.

Returns
Error code indicating the status of the disable operation.

Table 7-13. Return Values

Return value	Description
STATUS_OK	If the BOD12 was successfully disabled

7.2.1.11. Function bod12_is_detected()

Checks if the BOD12 low voltage detection has occurred.

```
bool bod12_is_detected( void )
```

Determines if the BOD12 has detected a voltage lower than its configured threshold.

Returns
Detection status of the BOD12.

Table 7-14. Return Values

Return value	Description
true	If the BOD12 has detected a low voltage condition
false	If the BOD12 has not detected a low voltage condition

7.2.1.12. Function bod12_clear_detected()

Clears the low voltage detection state of the BOD12.

```
void bod12_clear_detected( void )
```

Clears the low voltage condition of BOD12 module, so that new low voltage conditions can be detected.

7.3. Enumeration Definitions

7.3.1. Enum bod12_action

List of possible BOD12 actions when a BOD12 module detects a brown-out condition.

Table 7-15. Members

Enum value	Description
BOD12_ACTION_NONE	A BOD12 detect will do nothing, and the BOD12 state must be polled
BOD12_ACTION_RESET	A BOD12 detect will reset the device
BOD12_ACTION_INTERRUPT	A BOD12 detect will fire an interrupt

7.3.2. Enum bod12_mode_in_active

List of possible BOD12 module voltage sampling modes in active sleep mode.

Table 7-16. Members

Enum value	Description
BOD12_ACTCFG_CONTINUOUS	BOD12 will sample the supply line continuously
BOD12_ACTCFG_SAMPLED	BOD12 will use the BOD12 sampling clock (1KHz) to sample the supply line

7.3.3. Enum bod12_mode_in_standby

List of possible BOD12 module voltage sampling modes in standby sleep mode.

Table 7-17. Members

Enum value	Description
BOD12_STDBYCFG_CONTINUOUS	BOD12 will sample the supply line continuously
BOD12_STDBYCFG_SAMPLED	BOD12 will use the BOD12 sampling clock (1KHz) to sample the supply line

7.3.4. Enum bod12_prescale

List of possible BOD12 controller prescaler values, to reduce the sampling speed of a BOD12 to lower the power consumption.

Table 7-18. Members

Enum value	Description
BOD12_PRESCALE_DIV_2	Divide input prescaler clock by 2
BOD12_PRESCALE_DIV_4	Divide input prescaler clock by 4
BOD12_PRESCALE_DIV_8	Divide input prescaler clock by 8
BOD12_PRESCALE_DIV_16	Divide input prescaler clock by 16
BOD12_PRESCALE_DIV_32	Divide input prescaler clock by 32
BOD12_PRESCALE_DIV_64	Divide input prescaler clock by 64
BOD12_PRESCALE_DIV_128	Divide input prescaler clock by 128
BOD12_PRESCALE_DIV_256	Divide input prescaler clock by 256
BOD12_PRESCALE_DIV_512	Divide input prescaler clock by 512
BOD12_PRESCALE_DIV_1024	Divide input prescaler clock by 1024
BOD12_PRESCALE_DIV_2048	Divide input prescaler clock by 2048
BOD12_PRESCALE_DIV_4096	Divide input prescaler clock by 4096
BOD12_PRESCALE_DIV_8192	Divide input prescaler clock by 8192
BOD12_PRESCALE_DIV_16384	Divide input prescaler clock by 16384
BOD12_PRESCALE_DIV_32768	Divide input prescaler clock by 32768
BOD12_PRESCALE_DIV_65536	Divide input prescaler clock by 65536

7.3.5. Enum bod33_action

List of possible BOD33 actions when a BOD33 module detects a brown-out condition.

Table 7-19. Members

Enum value	Description
BOD33_ACTION_NONE	A BOD33 detect will do nothing, and the BOD33 state must be polled
BOD33_ACTION_RESET	A BOD33 detect will reset the device
BOD33_ACTION_INTERRUPT	A BOD33 detect will fire an interrupt
BOD33_ACTION_BACKUP	A BOD33 detect will put the device in backup sleep mode

7.3.6. Enum bod33_mode_in_active

List of possible BOD33 module voltage sampling modes in active sleep mode.

Table 7-20. Members

Enum value	Description
BOD33_ACTCFG_CONTINUOUS	BOD33 will sample the supply line continuously
BOD33_ACTCFG_SAMPLED	BOD33 will use the BOD33 sampling clock (1KHz) to sample the supply line

7.3.7. Enum bod33_mode_in_standby

List of possible BOD33 module voltage sampling modes in standby sleep mode.

Table 7-21. Members

Enum value	Description
BOD33_STDBYCFG_CONTINUOUS	BOD33 will sample the supply line continuously
BOD33_STDBYCFG_SAMPLED	BOD33 will use the BOD33 sampling clock (1KHz) to sample the supply line

7.3.8. Enum bod33_prescale

List of possible BOD33 controller prescaler values, to reduce the sampling speed of a BOD33 to lower the power consumption.

Table 7-22. Members

Enum value	Description
BOD33_PRESCALE_DIV_2	Divide input prescaler clock by 2
BOD33_PRESCALE_DIV_4	Divide input prescaler clock by 4
BOD33_PRESCALE_DIV_8	Divide input prescaler clock by 8
BOD33_PRESCALE_DIV_16	Divide input prescaler clock by 16
BOD33_PRESCALE_DIV_32	Divide input prescaler clock by 32
BOD33_PRESCALE_DIV_64	Divide input prescaler clock by 64
BOD33_PRESCALE_DIV_128	Divide input prescaler clock by 128
BOD33_PRESCALE_DIV_256	Divide input prescaler clock by 256
BOD33_PRESCALE_DIV_512	Divide input prescaler clock by 512
BOD33_PRESCALE_DIV_1024	Divide input prescaler clock by 1024
BOD33_PRESCALE_DIV_2048	Divide input prescaler clock by 2048
BOD33_PRESCALE_DIV_4096	Divide input prescaler clock by 4096

Enum value	Description
BOD33_PRESCALE_DIV_8192	Divide input prescaler clock by 8192
BOD33_PRESCALE_DIV_16384	Divide input prescaler clock by 16384
BOD33_PRESCALE_DIV_32768	Divide input prescaler clock by 32768
BOD33_PRESCALE_DIV_65536	Divide input prescaler clock by 65536

7.3.9. Enum bod33_vol_monitor

List of possible BOD33 module voltage monitored in active and standby mode.

Table 7-23. Members

Enum value	Description
BOD33_VMON_VDD	The BOD33 monitors the VDD power pin in active and standby mode
BOD33_VMON_VBAT	The BOD33 monitors the VBAT power pin in active and standby mode

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 BOD33 and BOD12 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_bod33(void)
{
    struct bod33_config config_bod33;
    bod33_get_config_defaults(&config_bod33);

    bod33_set_config(&config_bod33);

    bod33_enable();
}

static void configure_bod12(void)
{
    struct bod12_config config_bod12;
    bod12_get_config_defaults(&config_bod12);

    bod12_set_config(&config_bod12);

    bod12_enable();
}
```

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

```
configure_bod33();
```

```
configure_bod12();
```

9.1.1.3. Workflow

1. Create a BOD33 module configuration struct, which can be filled out to adjust the configuration of a physical BOD peripheral.

   ```
   struct bod33_config config_bod33;
   ```

2. Initialize the BOD33 configuration struct with the module's default values.

   ```
   bod33_get_config_defaults(&config_bod33);
   ```

   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 BOD33 module with the desired settings.

   ```
   bod33_set_config(&config_bod33);
   ```

4. Enable the BOD33 module so that it will monitor the power supply voltage.

   ```
   bod33_enable();
   ```

The workflow of the BOD12 is the same as for the BOD33.

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 BOD33 levels and settings is through the fuses. When it is desirable to set it in software, see the below use case.

In this use case, a new BOD33 level might be set in SW if the clock settings are adjusted up after a battery has charged to a higher level. When the battery discharges, the chip will reset when the battery level is below SW BOD33 level. Now the chip will run at a lower clock rate and the BOD33 level from fuse. The chip should always measure the voltage before adjusting the frequency up.

10. Document Revision History

Doc. Rev.	Date	Comments
42453B	12/2015	Added support for SAM L22
42453A	06/2015	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**