Introduction

This driver for Atmel® | SMART ARM®-based microcontrollers provides an interface for the configuration and management of the device's system relation functionality, necessary for the basic device operation. This is not limited to a single peripheral, but extends across multiple hardware peripherals.

The following peripherals are used by this module:
- PM (Power Manager)
- RSTC (Reset Controller)
- SUPC (Supply Controller)

The following devices can use this module:
- Atmel | SMART SAM C20/C21

The outline of this documentation is as follows:
- Prerequisites
- Module Overview
- Special Considerations
- Extra Information
- Examples
- API Overview
# Table of Contents

Introduction ...................................................................................................................... 1

1. Software License....................................................................................................... 3

2. Prerequisites.............................................................................................................. 4

3. Module Overview....................................................................................................... 5
   3.1. Voltage Regulator ............................................................................................... 5
   3.2. Voltage References ............................................................................................. 5
   3.3. System Reset Cause ........................................................................................... 5
   3.4. Sleep Modes ....................................................................................................... 5

4. Special Considerations.............................................................................................. 7

5. Extra Information....................................................................................................... 8

6. Examples................................................................................................................ 9

7. API Overview........................................................................................................... 10
   7.1. Structure Definitions........................................................................................... 10
       7.1.1. Struct system_standby_config ..................................................................... 10
       7.1.2. Struct system_voltage_references_config .................................................. 10
       7.1.3. Struct system_voltage_regulator_config ..................................................... 10
   7.2. Function Definitions............................................................................................ 10
       7.2.1. Voltage Regulator ....................................................................................... 10
       7.2.2. Voltage References ..................................................................................... 11
       7.2.3. Device Sleep Control .................................................................................. 12
       7.2.4. Standby Configuration ................................................................................ 13
       7.2.5. Reset Control ............................................................................................... 13
       7.2.6. System Debugger ......................................................................................... 14
       7.2.7. System Identification ................................................................................... 14
       7.2.8. System Initialization ...................................................................................... 14
   7.3. Enumeration Definitions...................................................................................... 15
       7.3.1. Enum system_reset_cause .......................................................................... 15
       7.3.2. Enum system_sleepmode ............................................................................ 15
       7.3.3. Enum system_voltage_reference ................................................................. 15
       7.3.4. Enum system_voltage_references_sel ............................................................. 16
       7.3.5. Enum system_vreg_switch_mode ................................................................. 16

8. Extra Information for SYSTEM Driver ...................................................................... 17
   8.1. Acronyms ........................................................................................................... 17
   8.2. Dependencies ..................................................................................................... 17
   8.3. Errata .................................................................................................................. 17
   8.4. Module History ................................................................................................. 17

9. Document Revision History ..................................................................................... 18
1. **Software License**

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

3. The name of Atmel may not be used to endorse or promote products derived from this software without specific prior written permission.

4. This software may only be redistributed and used in connection with an Atmel microcontroller product.

THIS SOFTWARE IS PROVIDED BY ATMEL "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NON-INFRINGEMENT ARE EXPRESSLY AND SPECIFICALLY DISCLAIMED. IN NO EVENT SHALL ATMEL BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.
2. Prerequisites

There are no prerequisites for this module.
3. **Module Overview**

The System driver provides a collection of interfaces between the user application logic, and the core device functionality (such as clocks, reset cause determination, etc.) that is required for all applications. It contains a number of sub-modules that control one specific aspect of the device:

- System Core (this module)
- System Clock Control (sub-module)
- System Interrupt Control (sub-module)
- System Pin Multiplexer Control (sub-module)

3.1. **Voltage Regulator**

The SAM device controls the voltage regulators for the core (VDDCORE). It sets the voltage regulators according to the sleep modes.

There are a selectable reference voltage and voltage dependent on the temperature which can be used by analog modules like the ADC.

3.2. **Voltage References**

The various analog modules within the SAM devices (such as AC, ADC, and DAC) require a voltage reference to be configured to act as a reference point for comparisons and conversions.

The SAM devices contain multiple references, including an internal temperature sensor and a fixed band-gap voltage source. When enabled, the associated voltage reference can be selected within the desired peripheral where applicable.

3.3. **System Reset Cause**

In some applications there may be a need to execute a different program flow based on how the device was reset. For example, if the cause of reset was the Watchdog timer (WDT), this might indicate an error in the application, and a form of error handling or error logging might be needed.

For this reason, an API is provided to retrieve the cause of the last system reset, so that appropriate action can be taken.

3.4. **Sleep Modes**

The SAM devices have several sleep modes. The sleep mode controls which clock systems on the device will remain enabled or disabled when the device enters a low power sleep mode. **Table 3-1 SAM Device Sleep Modes** on page 6 lists the clock settings of the different sleep modes.
### Table 3-1  SAM Device Sleep Modes

<table>
<thead>
<tr>
<th>Sleep mode</th>
<th>CPU clock</th>
<th>AHB clock</th>
<th>APB clocks</th>
<th>Clock sources</th>
<th>System clock</th>
<th>32KHz Reg mode</th>
<th>RAM mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>Idle 0</td>
<td>Stop</td>
<td>Run</td>
<td>Run</td>
<td>Run</td>
<td>Run</td>
<td>Run</td>
<td>Normal</td>
</tr>
<tr>
<td>Idle 1</td>
<td>Stop</td>
<td>Stop</td>
<td>Run</td>
<td>Run</td>
<td>Run</td>
<td>Run</td>
<td>Normal</td>
</tr>
<tr>
<td>Idle 2</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
<td>Run</td>
<td>Run</td>
<td>Run</td>
<td>Normal</td>
</tr>
<tr>
<td>Standby</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
<td>Stop</td>
<td>Low Power</td>
</tr>
</tbody>
</table>

Before entering device sleep, one of the available sleep modes must be set. The device will automatically wake up in response to an interrupt being generated or upon any other sleep mode exit condition.

Some peripheral clocks will remain enabled during sleep, depending on their configuration. If desired, the modules can remain clocked during sleep to allow them continue to operate while other parts of the system are powered down to save power.
4. **Special Considerations**

Most of the functions in this driver have device specific restrictions and caveats; refer to your device datasheet.
5. Extra Information

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

• Acronyms
• Dependencies
• Errata
• Module History
6. Examples

For SYSTEM module related examples, refer to the sub-modules listed in the Module Overview.
7. **API Overview**

7.1. **Structure Definitions**

7.1.1. **Struct system_standby_config**

Configuration structure for standby mode.

Table 7-1 Members

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>hmcramchs_back_bias</td>
<td>Back bias for HMCRAMCHS (false: no, true: standby)</td>
</tr>
<tr>
<td>enum system_vreg_switch_mode</td>
<td>vreg_switch_mode</td>
<td>Regulator switch mode in standby</td>
</tr>
</tbody>
</table>

7.1.2. **Struct system_voltage_references_config**

Configuration structure for VREF.

Table 7-2 Members

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>on_demand</td>
<td>On demand control</td>
</tr>
<tr>
<td>bool</td>
<td>run_in_standby</td>
<td>Run in standby</td>
</tr>
<tr>
<td>enum system_voltage_references_sel</td>
<td>sel</td>
<td>Voltage references selection</td>
</tr>
</tbody>
</table>

7.1.3. **Struct system_voltage_regulator_config**

Configuration structure for VREG.

Table 7-3 Members

<table>
<thead>
<tr>
<th>Type</th>
<th>Name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>bool</td>
<td>run_in_standby</td>
<td>Run in standby in standby sleep mode</td>
</tr>
</tbody>
</table>

7.2. **Function Definitions**

7.2.1. **Voltage Regulator**

7.2.1.1. **Function system_voltage_regulator_get_config_defaults()**

Retrieve the default configuration for voltage regulator.

```c
void system_voltage_regulator_get_config_defaults(
    struct system_voltage_regulator_config *const config)
```

Fills a configuration structure with the default configuration:
- The voltage regulator is in low power mode in Standby sleep mode

Table 7-4 Parameters

<table>
<thead>
<tr>
<th>Data direction</th>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[out]</td>
<td>config</td>
<td>Configuration structure to fill with default values</td>
</tr>
</tbody>
</table>

7.2.1.2. Function system_voltage_regulator_set_config()

Configure voltage regulator.

```c
void system_voltage_regulator_set_config(
    struct system_voltage_regulator_config *const config)
```

Configures voltage regulator with the given configuration.

Table 7-5 Parameters

<table>
<thead>
<tr>
<th>Data direction</th>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[in]</td>
<td>config</td>
<td>Voltage regulator configuration structure containing the new config</td>
</tr>
</tbody>
</table>

7.2.1.3. Function system_voltage_regulator_enable()

Enable the selected voltage regulator.

```c
void system_voltage_regulator_enable( void )
```

Enables the selected voltage regulator source.

7.2.1.4. Function system_voltage_regulator_disable()

Disable the selected voltage regulator.

```c
void system_voltage_regulator_disable( void )
```

Disables the selected voltage regulator.

7.2.2. Voltage References

7.2.2.1. Function system_voltage_reference_get_config_defaults()

Retrieve the default configuration for voltage reference.

```c
void system_voltage_reference_get_config_defaults(
    struct system_voltage_references_config *const config)
```

Fill a configuration structure with the default configuration:

- 1.024V voltage reference typical value
- On demand control: disabled
- The voltage reference and the temperature sensor are halted during standby sleep mode

Table 7-6 Parameters

<table>
<thead>
<tr>
<th>Data direction</th>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[out]</td>
<td>config</td>
<td>Configuration structure to fill with default values</td>
</tr>
</tbody>
</table>
7.2.2.2. Function system_voltage_reference_set_config()

Configure voltage reference.

```c
void system_voltage_reference_set_config(
    struct system_voltage_reference_config *const config)
```

Configures voltage reference with the given configuration.

<table>
<thead>
<tr>
<th>Data direction</th>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[in]</td>
<td>config</td>
<td>Voltage reference configuration structure containing the new config</td>
</tr>
</tbody>
</table>

7.2.2.3. Function system_voltage_reference_enable()

Enable the selected voltage reference.

```c
void system_voltage_reference_enable(
    const enum system_voltage_reference vref)
```

Enables the selected voltage reference source, making the voltage reference available on a pin as well as an input source to the analog peripherals.

<table>
<thead>
<tr>
<th>Data direction</th>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[in]</td>
<td>vref</td>
<td>Voltage reference to enable</td>
</tr>
</tbody>
</table>

7.2.2.4. Function system_voltage_reference_disable()

Disable the selected voltage reference.

```c
void system_voltage_reference_disable(
    const enum system_voltage_reference vref)
```

Disables the selected voltage reference source.

<table>
<thead>
<tr>
<th>Data direction</th>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[in]</td>
<td>vref</td>
<td>Voltage reference to disable</td>
</tr>
</tbody>
</table>

7.2.3. Device Sleep Control

7.2.3.1. Function system_set_sleepmode()

Set the sleep mode of the device.

```c
void system_set_sleepmode(
    const enum system_sleepmode sleep_mode)
```

Sets the sleep mode of the device; the configured sleep mode will be entered upon the next call of the `system_sleep()` function.

For an overview of which systems are disabled in sleep for the different sleep modes, see Sleep Modes.
Table 7-10 Parameters

<table>
<thead>
<tr>
<th>Data direction</th>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[in]</td>
<td>sleep_mode</td>
<td>Sleep mode to configure for the next sleep operation</td>
</tr>
</tbody>
</table>

7.2.3.2. Function system_sleep()

Put the system to sleep waiting for interrupt.

```c
void system_sleep( void )
```

Executes a device DSB (Data Synchronization Barrier) instruction to ensure all ongoing memory accesses have completed, then a WFI (Wait For Interrupt) instruction to place the device into the sleep mode specified by `system_set_sleepmode` until woken by an interrupt.

7.2.4. Standby Configuration

7.2.4.1. Function system_standby_get_config_defaults()

Retrieve the default configuration for standby.

```c
void system_standby_get_config_defaults(
  struct system_standby_config *const config)
```

Fills a configuration structure with the default configuration for standby:
- Automatic VREG switching is used
- Retention back biasing mode for HMCRAMCHS

Table 7-11 Parameters

<table>
<thead>
<tr>
<th>Data direction</th>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[out]</td>
<td>config</td>
<td>Configuration structure to fill with default values</td>
</tr>
</tbody>
</table>

7.2.4.2. Function system_standby_set_config()

Configure standby mode.

```c
void system_standby_set_config(
  struct system_standby_config *const config)
```

Configures standby with the given configuration.

Table 7-12 Parameters

<table>
<thead>
<tr>
<th>Data direction</th>
<th>Parameter name</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>[in]</td>
<td>config</td>
<td>Standby configuration structure containing the new config</td>
</tr>
</tbody>
</table>

7.2.5. Reset Control

7.2.5.1. Function system_reset()

Reset the MCU.

```c
void system_reset( void )
```
Resets the MCU and all associated peripherals and registers, except RTC, OSC32KCTRL, RSTC, GCLK(if WRTLOCK is set) and I/O retention state of PM.

7.2.5.2. Function system_get_reset_cause()

Get the reset cause.

```c
enum system_reset_cause system_get_reset_cause( void )
```

Retrieves the cause of the last system reset.

**Returns**

An enum value indicating the cause of the last system reset.

7.2.6. System Debugger

7.2.6.1. Function system_is_debugger_present()

Check if debugger is present.

```c
bool system_is_debugger_present( void )
```

Check if debugger is connected to the onboard debug system (DAP).

**Returns**

A bool identifying if a debugger is present.

**Table 7-13 Return Values**

<table>
<thead>
<tr>
<th>Return value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>true</td>
<td>Debugger is connected to the system</td>
</tr>
<tr>
<td>false</td>
<td>Debugger is not connected to the system</td>
</tr>
</tbody>
</table>

7.2.7. System Identification

7.2.7.1. Function system_get_device_id()

Retrieve the device identification signature.

```c
uint32_t system_get_device_id( void )
```

Retrieves the signature of the current device.

**Returns**

Device ID signature as a 32-bit integer.

7.2.8. System Initialization

7.2.8.1. Function system_init()

Initialize system.

```c
void system_init( void )
```

This function will call the various initialization functions within the system namespace. If a given optional system module is not available, the associated call will effectively be a NOP (No Operation).
Currently the following initialization functions are supported:
- System clock initialization (via the SYSTEM CLOCK sub-module)
- Board hardware initialization (via the Board module)
- Event system driver initialization (via the EVSYS module)
- External Interrupt driver initialization (via the EXTINT module)

7.3. Enumeration Definitions

7.3.1. Enum system_reset_cause

List of possible reset causes of the system.

Table 7-14 Members

<table>
<thead>
<tr>
<th>Enum value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM_RESET_CAUSE_SOFTWARE</td>
<td>The system was last reset by a software reset</td>
</tr>
<tr>
<td>SYSTEM_RESET_CAUSE_WDT</td>
<td>The system was last reset by the watchdog timer</td>
</tr>
<tr>
<td>SYSTEM_RESET_CAUSE_EXTERNAL_RESET</td>
<td>The system was last reset because the external reset line was pulled low</td>
</tr>
<tr>
<td>SYSTEM_RESET_CAUSE_BODVDD</td>
<td>The system was last reset by VDD brown out detector</td>
</tr>
<tr>
<td>SYSTEM_RESET_CAUSE_BODCORE</td>
<td>The system was last reset by VDDCORE brown out detector</td>
</tr>
<tr>
<td>SYSTEM_RESET_CAUSE_POR</td>
<td>The system was last reset by the POR (Power on reset)</td>
</tr>
</tbody>
</table>

7.3.2. Enum system_sleepmode

List of available sleep modes in the device. A table of clocks available in different sleep modes can be found in Sleep Modes.

Table 7-15 Members

<table>
<thead>
<tr>
<th>Enum value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM_SLEEPMODE_IDLE_0</td>
<td>IDLE 0 sleep mode</td>
</tr>
<tr>
<td>SYSTEM_SLEEPMODE_IDLE_1</td>
<td>IDLE 1 sleep mode</td>
</tr>
<tr>
<td>SYSTEM_SLEEPMODE_IDLE_2</td>
<td>IDLE 2 sleep mode</td>
</tr>
<tr>
<td>SYSTEM_SLEEPMODE_STANDBY</td>
<td>Standby sleep mode</td>
</tr>
</tbody>
</table>

7.3.3. Enum system_voltage_reference

List of available voltage references (VREF) that may be used within the device.
Table 7-16 Members

<table>
<thead>
<tr>
<th>Enum value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM_VOLTAGE_REFERENCE_TEMPSENSE</td>
<td>Temperature sensor voltage reference</td>
</tr>
<tr>
<td>SYSTEM_VOLTAGE_REFERENCE_OUTPUT</td>
<td>Voltage reference output for ADC/DAC</td>
</tr>
</tbody>
</table>

7.3.4. Enum system_voltage_references_sel

Voltage references selection for ADC/DAC.

Table 7-17 Members

<table>
<thead>
<tr>
<th>Enum value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM_VOLTAGE_REFERENCE_1V024</td>
<td>1.024V voltage reference typical value</td>
</tr>
<tr>
<td>SYSTEM_VOLTAGE_REFERENCE_2V048</td>
<td>2.048V voltage reference typical value</td>
</tr>
<tr>
<td>SYSTEM_VOLTAGE_REFERENCE_4V096</td>
<td>4.096V voltage reference typical value</td>
</tr>
</tbody>
</table>

7.3.5. Enum system_vreg_switch_mode

Table 7-18 Members

<table>
<thead>
<tr>
<th>Enum value</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>SYSTEM_VREG_SWITCH_AUTO</td>
<td>Automatic mode</td>
</tr>
<tr>
<td>SYSTEM_VREG_SWITCH_PERFORMANCE</td>
<td>Performance oriented</td>
</tr>
<tr>
<td>SYSTEM_VREG_SWITCH_LP</td>
<td>Low Power consumption oriented</td>
</tr>
</tbody>
</table>
8. Extra Information for SYSTEM Driver

8.1. Acronyms

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

<table>
<thead>
<tr>
<th>Acronym</th>
<th>Definition</th>
</tr>
</thead>
<tbody>
<tr>
<td>PM</td>
<td>Power Manager</td>
</tr>
<tr>
<td>SUPC</td>
<td>Supply Controller</td>
</tr>
<tr>
<td>RSTC</td>
<td>Reset Controller</td>
</tr>
</tbody>
</table>

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.

<table>
<thead>
<tr>
<th>Changelog</th>
</tr>
</thead>
<tbody>
<tr>
<td>Initial Release</td>
</tr>
</tbody>
</table>
## 9. Document Revision History

<table>
<thead>
<tr>
<th>Doc. Rev.</th>
<th>Date</th>
<th>Comments</th>
</tr>
</thead>
<tbody>
<tr>
<td>42484A</td>
<td>12/2015</td>
<td>Initial document release.</td>
</tr>
</tbody>
</table>