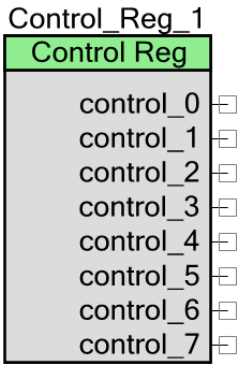
Control Register

1.70



Features

- Up to 8-bit Control Register

General Description

The Control Register allows the firmware to output digital signals.

When to Use a Control Register

Use a Control Register when the firmware needs to interact with a digital system. You can also use the Control Register as a configuration register, allowing the firmware to specify the desired behavior of the digital system.

Input/Output Connections

This section describes the input and output connections for the Control Register. An asterisk (*) indicates that the I/O may be hidden on the symbol under the conditions listed in the description of that I/O.

clock – Input *

This optional pin is present if the **Mode** parameter is set to **SyncMode** or **PulseMode**. Otherwise, the clock input does not show.

reset – Input *

This optional input is used to reset Control Register bits. This input is shown on the symbol when you enable the **External reset** parameter, and set the **Mode** parameter to **SyncMode** or **PulseMode**. The reset input may be left floating with no external connection. If nothing is connected to the reset line, the component will assign it a constant logic 0.

control_0 - control_7 – Output *

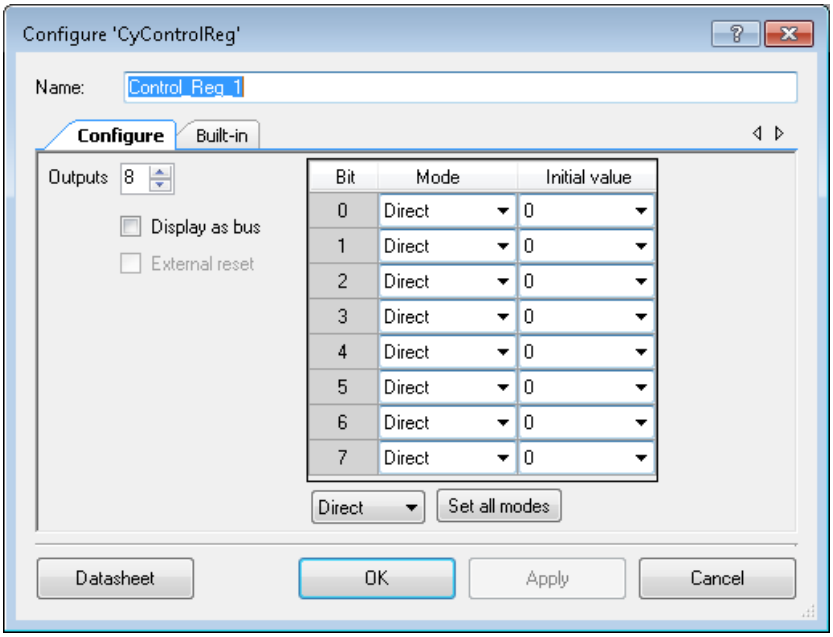
The Control Register contains up to eight outputs. The firmware sets the values of the output terminals by writing to the Control Register. The number of outputs depends on the setting for the **Outputs** parameter.

control[N:0] – Output *

This optional output sweeps the individual output terminals into a single bus terminal. This pin is visible when **Display as bus** parameter is enabled. N is the number of outputs - 1.

Component Parameters

Drag a control register onto your design and double-click it to open the Configure dialog.



Outputs

Number of output terminals (1 to 8). The default value is **8**. Bit0 is the LSB and corresponds to the control_0 terminal.

Display as bus

This parameter displays the outputs as a bus instead of individual terminals. This option is unchecked by default.

External reset

This check box is used to enable the reset input on the symbol. This option is unchecked by default. **External reset** is not valid when all the bits in the **Mode** parameter are configured as **DirectMode**. In this case this option is disabled.

Set all modes

This button sets all bits to **DirectMode**, **SyncMode** or **PulseMode**, depending on the mode selected in the combo box in the left hand side of this button.

Mode

These parameters are used to set specific bits of the Control Register to one of three settings:

- **DirectMode –** In this mode, when the control register is written with bus clock, the data is driven directly into the routing.
- **SyncMode –** Resamples (single-synched) the control bit input from the bus clock to the selected SC clock before it is driven into the routing.
- **PulseMode –** This mode is similar to SyncMode, in that the Control bit input is resampled from the bus clock to the selected SC clock and a single SC clock period pulse is generated. The output of the control bit into the routing is asserted for one full SC clock period. At the end of the pulse, the control bit is automatically reset.

Initial value

These parameters allow to set the default value of 0 or 1 for each bit in the Control Register. By default, the initial value is 0.

Low Power Mode Behavior

None of the Control Register content is retained during low power modes (sleep, deep sleep, and hibernate). Each bit of the Control Register component is initialized with a ‘0’ value when the device wakes up from low power mode.

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. The following table lists and describes the interface to each function. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name “Control_Reg_1” to the first instance of a control register in any given design. You can rename the component to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol. For readability, the instance names used in the following tables is “ControlReg.”

Function	Description
ControlReg_Write()	Writes a byte to a control register

Function	Description
ControlReg_Read()	Reads the current value assigned to a control register

void ControlReg_Write (uint8 control)

Description: Writes a byte to the control register

Parameters: control: The value to be assigned to the control register

Return Value: None

Side Effects: Sets the state of the control register's outputs

uint8 ControlReg_Read (void)

Description: Reads the current value assigned to the control register

Parameters: None

Return Value: Returns the current value assigned to the control register

Side Effects: None

DMA

The DMA component can be used to write data directly from RAM to the Control Register. The DMA Wizard can be used to configure DMA operations as follows:

Name of DMA Source/Destination in DMA Wizard	Direction	DMA Req Signal	DMA Req Type	Description
ControlReg_Control_PTR	Destination	N/A	N/A	Stores Control Register value.

MISRA Compliance

This section describes the MISRA-C:2004 compliance and deviations for the component. There are two types of deviations defined:

- project deviations – deviations that are applicable for all PSoC Creator components
- specific deviations – deviations that are applicable only for this component

This section provides information on component-specific deviations. Project deviations are described in the MISRA Compliance section of the *System Reference Guide* along with information on the MISRA compliance verification environment.

The Control Register component does not have any specific deviations.

Sample Firmware Source Code

PSoC Creator provides numerous example projects that include schematics and example code in the Find Example Project dialog. For component-specific examples, open the dialog from the Component Catalog or an instance of the component in a schematic. For general examples, open the dialog from the Start Page or **File** menu. As needed, use the **Filter Options** in the dialog to narrow the list of projects available to select.

Refer to the “Find Example Project” topic in the PSoC Creator Help for more information.

Resources

The Control Register component uses one control cell in the UDB array.

API Memory Usage

The component memory usage varies significantly, depending on the compiler, device, number of APIs used and component configuration. The following table provides the memory usage for all APIs available in the given component configuration.

The measurements have been done with the associated compiler configured in Release mode with optimization set for Size. For a specific design the map file generated by the compiler can be analyzed to determine the memory usage.

Configuration	PSoC 3 (Keil_PK51)		PSoC 4 (GCC)		PSoC 5LP (GCC)	
	Flash (Bytes)	RAM (Bytes)	Flash (Bytes)	RAM (Bytes)	Flash (Bytes)	RAM (Bytes)
Default	12	0	24	0	24	0

Component Changes

This section lists the major changes in the component from the previous version.

Version	Description of Changes	Reason for Changes / Impact
1.70.d	Minor datasheet edit.	
1.70.c	Updated the datasheet.	Added low power mode behavior section.
	Updated the table appearance in the Configure dialog.	Corrected a display problem at 120 dpi resolution.
1.70.b	Updated datasheet with memory usage for PSoC 4.	
1.70.a	Added MISRA Compliance section.	The component does not have any specific deviations.
1.70	Added support for PSoC 5LP silicon.	
	Updated the Configure dialog.	Added Display as bus option; Set all modes button and minor design changes.
	Added display as bus option for output terminals.	To have output terminals as bus
	Added the Sync Mode support for PSoC 5LP silicon and implemented DMA capabilities and Debug window support.	PSoC 5LP silicon supports Sync mode.
1.60	Updated the Configure dialog	Changed the Bit display and addressed minor Configure dialog issues
1.50.b	Minor datasheet edits and updates	
1.50.a	Minor datasheet edits and updates	
1.50	Updated the Configure dialog.	Created a customized interface. Added "Set All" and “Clear All” buttons and changed Number of Inputs field to allow keyboard entry. Updated the dialog to comply with corporate standards.
	Added reset input and ExternalReset parameter to control visibility of reset input	This was added for PSoC 3 Production silicon to control the reset behavior of the control register
	Added BitValue parameter	To set control register Initial Value.
	Added Bit mode parameter to pick different control register modes (Direct, Sync and Pulse Mode).	New modes (Sync, Pulse Mode) were added to give the possibility to select a mode that resamples necessary Control Register bits to the UDB clock. This new mode can be used for PSoC 3 Production or later.
	Added Clock pin	Clock pin was added to support Sync and Pulse Mode which is exposed only when these modes are selected.

© Cypress Semiconductor Corporation, 2010-2015. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.