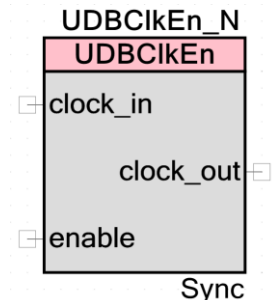


UDB Clock Enable (UDBClkEn)

1.0

Features

- Clock enable support
- Addition of synchronization on a clock when needed



General Description

The universal digital block (UDB) Clock Enable (UDBClkEn) component supports precise control over clocking behavior.

When to Use a UDBClkEn

The UDBClkEn component can be used to:

- Apply a level-sensitive enable to any clock signal.
- Force the clock signal to be synchronous to BUS_CLK.
 If the incoming clock is already synchronous, no changes are made. If the incoming clock is asynchronous, logic is inserted to synchronize to BUS_CLK.
- Indicate that a clock signal is allowed to be asynchronous.
 Several UDB elements (control register, status register, and datapath) must normally be clocked with a synchronous clock. PSoC Creator only allows these to be clocked with an asynchronous clock if that clock comes from the UDBClkEn component in Async mode. This feature should only be used after analyzing the potential clock crossing issues associated with communicating with the CPU operating on the BUS_CLK clock domain.

Input/Output Connections

This section describes the various input and output connections for the UDBClkEn component.

enable – Input

Level-sensitive clock enable signal applied to the input clock to form the output clock. If the enable signal is not synchronous to the clock_in signal, a synchronizer is automatically implemented. In addition, if the enable signal is not synchronous to the clock_in signal, then the pulse width of the enable signal must be at least the period of clock_in plus 2ns.

clock_in – Input

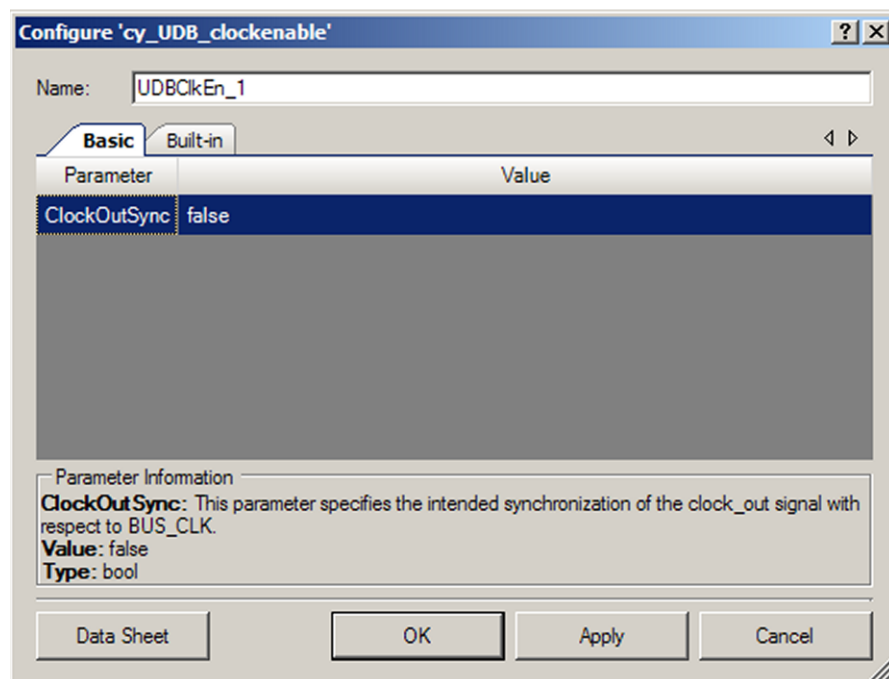
The incoming clock signal.

clock_out – Output

Resulting clock derived from enable and clock_in signals that meet the synchronization setting characteristics. This signal can only be connected to the clock input of a registered element.

Component Parameters

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



The UDBClkEn component provides the following parameter.

ClockOutSync

This parameter specifies the intended synchronization of the clock_out signal with respect to BUS_CLK. Setting this to true forces the clock to be synchronized if the clock_in signal is not already synchronous to BUS_CLK. Setting it to false leaves the synchronization unchanged from the synchronization of clock_in.

Functional Description

Each UDB has four clock-control blocks. These are used for each of the two PLDs, the datapath, and the status/control registers. The clock-control block selects the clock signal and, optionally, a clock enable signal. The enable capability of the UDBClkEn component is directly supported by this clock enable signal.

Synchronization of the clock_in input is required if it is asynchronous to BUS_CLK and the **ClockOutSync** parameter has been set to true. In this case, the double synchronizer mode of a status register (aka, sync cell) is used to synchronize the clock signal before using the clock as the input to the clock control block.

Resources

UDBClkEn component will consume one sync cell when clock_in is asynchronous to the BUS_CLK (SYS_CLK for PSoC 4) and the **ClockOutSync** parameter has been set to true. It will consume an additional macrocell if an enable signal is used in such a scenario.

If the enable signal is asynchronous to clock_in, then it will consume a sync block regardless of ClockOutSync or clock_in configuration. For more information on the advanced usage, refer to the *PSoC Creator Component Author Guide*.

Component Changes

Version	Description of Changes	Reason for Changes / Impact
1.0.e	Minor datasheet edits.	
1.0.d	Expanded the Functional Description and added resource information in the Resources section. Minor datasheet edits.	Clarified the actual resources used.
1.0.c	Added timing information to the enable input description	
1.0.b	Minor updates	
1.0.a	Removed Placement and API sections	Irrelevant sections for this component
1.0	First release of the UDBClkEn component	

© Cypress Semiconductor Corporation, 2011-2017. This document is the property of Cypress Semiconductor Corporation and its subsidiaries, including Spansion LLC ("Cypress"). This document, including any software or firmware included or referenced in this document ("Software"), is owned by Cypress under the intellectual property laws and treaties of the United States and other countries worldwide. Cypress reserves all rights under such laws and treaties and does not, except as specifically stated in this paragraph, grant any license under its patents, copyrights, trademarks, or other intellectual property rights. If the Software is not accompanied by a license agreement and you do not otherwise have a written agreement with Cypress governing the use of the Software, then Cypress hereby grants you a personal, non-exclusive, nontransferable license (without the right to sublicense) (1) under its copyright rights in the Software (a) for Software provided in source code form, to modify and reproduce the Software solely for use with Cypress hardware products, only internally within your organization, and (b) to distribute the Software in binary code form externally to end users (either directly or indirectly through resellers and distributors), solely for use on Cypress hardware product units, and (2) under those claims of Cypress's patents that are infringed by the Software (as provided by Cypress, unmodified) to make, use, distribute, and import the Software solely for use with Cypress hardware products. Any other use, reproduction, modification, translation, or compilation of the Software is prohibited.

TO THE EXTENT PERMITTED BY APPLICABLE LAW, CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS DOCUMENT OR ANY SOFTWARE OR ACCOMPANYING HARDWARE, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. To the extent permitted by applicable law, Cypress reserves the right to make changes to this document without further notice. Cypress does not assume any liability arising out of the application or use of any product or circuit described in this document. Any information provided in this document, including any sample design information or programming code, is provided only for reference purposes. It is the responsibility of the user of this document to properly design, program, and test the functionality and safety of any application made of this information and any resulting product. Cypress products are not designed, intended, or authorized for use as critical components in systems designed or intended for the operation of weapons, weapons systems, nuclear installations, life-support devices or systems, other medical devices or systems (including resuscitation equipment and surgical implants), pollution control or hazardous substances management, or other uses where the failure of the device or system could cause personal injury, death, or property damage ("Unintended Uses"). A critical component is any component of a device or system whose failure to perform can be reasonably expected to cause the failure of the device or system, or to affect its safety or effectiveness. Cypress is not liable, in whole or in part, and you shall and hereby do release Cypress from any claim, damage, or other liability arising from or related to all Unintended Uses of Cypress products. You shall indemnify and hold Cypress harmless from and against all claims, costs, damages, and other liabilities, including claims for personal injury or death, arising from or related to any Unintended Uses of Cypress products.

Cypress, the Cypress logo, Spansion, the Spansion logo, and combinations thereof, WICED, PSoC, CapSense, EZ-USB, F-RAM, and Traveo are trademarks or registered trademarks of Cypress in the United States and other countries. For a more complete list of Cypress trademarks, visit cypress.com. Other names and brands may be claimed as property of their respective owners.

