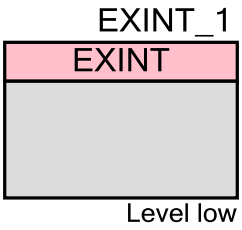
External Interrupt (PDL_EXINT) 1.0



Features

- Up to 32 external interrupt and one NMI input pin mounted
- Five types of detection condition
- Possible to use an EXINT component to return from standby mode
- Possible to enable/disable NMI interrupt in TYPE3-M0+ products

General Description

The Peripheral Driver Library (PDL) External Interrupt (PDL_EXINT) component control section outputs an external interrupt request to the interrupt controller in the following procedure:

1. The signal input to pin detects the edge or level specified in the External Interrupt Level Register. The edge or level to be detected can be selected from the following five types:
    - High level
    - Low level
    - Rising edge
    - Falling edge
    - Both rising and falling edge
2. The detected interrupt input is held in the interrupt factor F/F. It is read with the External Interrupt Factor Register.
3. If an external interrupt is enabled with the External Interrupt Enable Register, an external interrupt request to the interrupt controller is asserted.
4. The held interrupt factor is cleared with the External Interrupt Factor Clear Register, an external interrupt request to the interrupt controller is negated.

This component uses firmware drivers from the PDL_EXINT module, which is automatically added to your project after a successful build.

When to Use a PDL_EXINT Component

Use the PDL_EXINT component when you need to trigger the external signals in your firmware.

Quick Start

1. Drag a PDL_EXINT component from the Component Catalog FMx/Ports and Pins folder onto your schematic. The placed instance takes the name EXINT_1.
2. Double-click to open the component's Configure dialog.
3. On the **Basic** tab, set the following parameters:
   - select interrupt trigger condition
   - specify the callback function
4. You can use PDL_EXINT to configure NMI interrupt by setting NMI parameter in the component's Configure dialog.
5. Assign the pin in your device using the Pin Editor. If you are using a pin to read a SW button state on a development kit, refer the kit User Guide for suitable pin assignments.
6. Build the project to verify the correctness of your design. This will add the required PDL modules to the Workspace Explorer and generate configuration data for the EXINT_1 instance.
7. In the *main.c* file, initialize the peripheral and start the application.

```
stc_exint_config config;
config.abEnable[EINT_1_Index] = 1u;
config.aenLevel[EINT_1_Index] = EXINT_1_Level;
config.apfnExintCallback[EINT_1_Index] = EINT_1_ExintCallback;
config.bTouchNvic = 1;

Exint_Init(&config);
EXINT_1_SetPinFunc_INT();
```

8. Build and program the device.

Component Parameters

The PDL_EXINT component Configure dialog allows you to edit the configuration parameters for the component instance.

Basic Tab

This tab contains the component parameters used in the basic peripheral initialization settings.

Parameter Name	Data Structure Type	Description
enLevel	en_exint_level_t	Interrupt trigger condition.
NMI	stc_mfs_uart_config	Use NMI pin for EXINT.
pfnExintCallback	bool	Callback routine for external interrupt. Note: this generates a declaration only - USER must implement the function

Component Usage

After a successful build, firmware drivers from the PDL_EXINT module are added to your project in the pdl/drivers/exint folder. Pass the generated data structures to the associated PDL functions in your application initialization code to configure the peripheral.

Generated Data

The PDL_EXINT component populates the following peripheral initialization data structure(s). The generated code is placed in C source and header files that are named after the instance of the component (e.g. *EXINT_1_config.c*). Each variable is also prefixed with the instance name of the component.

Once the component is initialized, the application code should use the peripheral functions provided in the referenced PDL files. Refer to the PDL documentation for the list of provided API functions. To access this document, right-click on the component symbol on the schematic and choose "**Open API Documentation…**" option in the drop-down menu.

Preprocessor Macros

The PDL_EXINT component generates the following preprocessor macro(s). Note that each macro is prefixed with the instance name of the component (e.g. "EXINT_1").

Macro	Description
EXINT_1_SetPinFunc_NMIX()	Pin function macros. Visible when non masked interrupt mode is selected.
EXINT_1_SetPinFunc_INT()	Pin function macros. Visible when non masked interrupt mode isn't selected.
EXINT_1 _Index	Number of the EXINT line
EXINT_1 _Level	Indicate detection mode

Data in RAM

The generated data may be placed in flash memory (const) or RAM. The former is the more memory-efficient choice if you do not wish to modify the configuration data at run-time. Under the **Built-In** tab of the Configure dialog set the parameter CONST_CONFIG to make your selection. The default option is to place the data in flash.

Interrupt Support

If the PDL_EXINT component is specified to trigger interrupts. These can then be passed to the associated interrupt initialization functions along with your ISR callback function.

Code Examples and Application Notes

There are numerous code examples that include schematics and example code available online at the Cypress Code Examples web page.

Cypress also provides a number of application notes describing how FMx devices can be integrated into your design. You can access the Cypress Application Notes search web page at www.cypress.com/appnotes.

Resources

The PDL_EXINT component uses the EXINT (External Interrupt) peripheral block.

References

- FM0+ Family of 32-bit ARM® Cortex®-M0+ Microcontrollers Peripheral Manuals
- Cypress FM0+ Family of 32-bit ARM® Cortex®-M0+ Microcontrollers

Component Errata

This section lists known problems with the component.

Cypress ID	Component Version	Problem	Workaround
252934	1.0	EXINT is not supported on S6E1A devices. There is a defect in the EXINT interrupt handler that prevents its use on S6E1A devices. The defect does not impact S6E1B or S6E1C devices.	None. Contact Cypress technical support (http://www.cypress.com/mycases) for possible firmware updates and help with implementing external interrupts on S6E1A devices.

Component Changes

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

Version	Description of Changes	Reason for Changes / Impact
1.0	Initial Version	

© Cypress Semiconductor Corporation, 2016. 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.