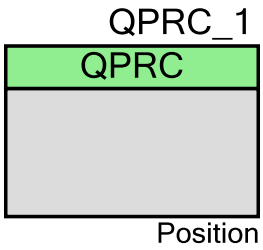
Quadrature Position Revolution Counter (PDL_QPRC)

1.0



Features

- 16-bit position counter
- 16-bit revolution counter
- Falling, rising, derived edge detection can be configured
- Noise filter for external pins
- CIN pin can be configured in Counter Clean function or Gate function
- Two 16-bit compare registers

General Description

The Peripheral Driver Library (PDL) Quadrature Position Revolution Counter (QPRC) component receives an input signal from AIN or BIN external pin as an event of count clock, and increments or decrements the counter. The counting conditions depend on the selected count mode.

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

When to Use a PDL_QPRC Component

The PDL_QPRC component is used to measure the position of the position encoder. In addition, it can be used as an up/down counter.

Quick Start

1. Drag a PDL_QPRC component from the Component Catalog FMx/System/Real-Time Clock folder onto your schematic. The placed instance takes the name QPRC_1.
2. Double-click to open the component's Configure dialog.
3. On the **Basic** tab set the following parameters:
    - set the compare mode;
    - select edge detection mode for external pins.
4. On the **Interrupts** tab, initialize needed interrupts and their callback functions.

5. 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 QPRC_1 instance.
6. In the *main.c* file initialize the peripheral and start the application.

```
QPRC_1_SetPinFunc_AIN();
QPRC_1_SetPinFunc_ZIN();
QPRC_1_SetPinFunc_BIN();
(void) Qprc_Init(QPRC_1_HW, (stc_qprc_config_t *)&QPRC_1_Config);
```

7. Build and program the device.

Component Parameters

The PDL_QPRC 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	Description
enAinEdge	Define AIN pin detection
enBinEdge	Define BIN pin detection
enZinEdge	Define ZIN pin detection
b8KValue	Out of range from 0x0 to 0x7FFF or 0x0 to 0xFFFF
bPhaseEdge	For 1-time frequency multiplication count on both edges or same edge
bSwapAinBin	Swap A and B inputs
enCompareMode	Compare with position or revolution counter
enPcResetMask	Position counter reset mask times

Filter Tab

This tab contains the Filter configuration settings.

Parameter Name	Description
bAinInputInvert	AIN pin invert input
bAinInputMask	AIN pin input mask
enAinWidth	Filter width for AIN pin
bBinInputInvert	BIN pin invert input

Parameter Name	Description
bBinInputMask	BIN pin input mask
enBinWidth	Filter width for BIN pin
bCinInputInvert	CIN pin invert input
bCinInputMask	CIN pin input mask
enCinWidth	Filter width for CIN pin

Interrupts Tab

This tab contains the Interrupts configuration settings.

Parameter Name	Description
bTouchNvic	Install interrupts in NVIC
bQprcPcMatchIrq	Enable interrupt on PC match of position counter
bQprcPcMatchRcMatchIrq	Enable interrupt on PC match and RC match of position counter
bQprcPcRcMatchIrq	Enable interrupt on PC and RC match of position counter
pfnPcMatchIrqCb	Callback function for PC match IRQ. Note: this generates a declaration only - USER must implement the function.
pfnPcMatchRcMatchIrqCb	Callback function for PC match and RC match IRQ. Note: this generates a declaration only - USER must implement the function.
pfnPcRcMatchIrqCb	Callback function for PC and RC match IRQ. Note: this generates a declaration only - USER must implement the function.
bQprcPcCountInvertIrq	Enable interrupt on PC invert
bQprcPcOfUfZeroIrq	Enable interrupt on overflow, underflow, or zero match of position counter
bQprcRcOutrangeIrq	Enable interrupt on RC out of range
pfnPcCountInvertIrqCb	Callback function for PC invert IRQ. Note: this generates a declaration only - USER must implement the function.
pfnPcOfUfZeroIrqCb	Callback function for overflow, underflow or zero match IRQ. Note: this generates a declaration only - USER must implement the function.
pfnRcOutrangeIrqCb	Callback function for RC out of range IRQ. Note: this generates a declaration only - USER must implement the function

Component Usage

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

Generated Data

The PDL_QPRC 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. *QPRC_1_config.c*). Each variable is also prefixed with the instance name of the component.

Data Structure Type	Name	Description
stc_qprc_irq_en_t	QPRC_1_IrqEn	Interrupt enable structure. This is automatically referenced in the QPRC_1_Config.
stc_qprc_irq_cb_t	QPRC _1_IrqCb	Interrupt callback functions structure. This is automatically referenced in the QPRC_1_Config.
stc_qprc_config_t	QPRC _1_Config	Configuration structure

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…**" in the drop-down menu.

Preprocessor Macros

The PDL_QPRC component generates the following preprocessor macro(s). Note that each macro is prefixed with the instance name of the component (e.g. "QPRC_1").

Macro	Description
QPRC_1_SetPinFunc_AIN	Macro to configure AIN output.
QPRC_1_SetPinFunc_BIN	Macro to configure BIN output.
QPRC_1_SetPinFunc_ZIN	Macro to configure ZIN output.
QPRC_1_HW	Hardware pointer to the block instance in the device. This should be used in all API calls to specify the block to access.

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_QPRC component is specified to trigger interrupts, it will generate the interrupt data structures. These can then be passed to the associated interrupt initialization functions along with your ISR callback function.

Function Callback	Description
QPRC_1_PcMatchIrqCb	PC match interrupt callback function of position counter. Note: this generates a declaration only - USER must implement the function.
QPRC_1_PcMatchRcMatchIrqCb	PC match and RC match interrupt callback function. Note: this generates a declaration only - USER must implement the function.
QPRC_1_PcRcMatchIrqCb	PC and RC match interrupt callback function. Note: this generates a declaration only - USER must implement the function.
QPRC_1_PcCountInvertIrqCb	PC invert interrupt callback function. Note: this generates a declaration only - USER must implement the function.
QPRC_1_PcOfUfZeroIrqCb	Overflow, underflow, zero match interrupt callback function of position counter. Note: this generates a declaration only - USER must implement the function.
QPRC_1_RcOutrangeIrqCb	RC outrange interrupt callback function. Note: this generates a declaration only - USER must implement the 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_QPRC component uses the Quadrature Position Revolution Counter (QPRC) 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 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.