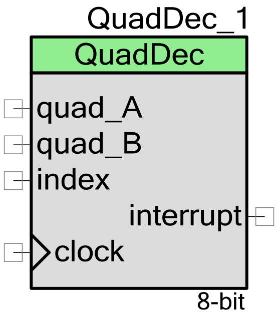
Quadrature Decoder (QuadDec)

1.20

Features



- Adjustable counter size: 8, 16, or 32-bits
- Counter resolution of 1x, 2x, or 4x the frequency of the A and B inputs, for more accurate determination of position or speed
- Optional Index input to determine absolute position
- Optional Glitch Filtering to reduce the impact of system-generated noise on the input(s)

General Description

The Quadrature Decoder (QuadDec) component provides the ability to count transitions on a pair of digital signals. The signals are typically provided by a speed / position feedback system mounted on a motor or trackball.

The signals – typically called A and B – are positioned 90° out-of-phase, which results in a "Gray" code output. A Gray code is a sequence where only one bit changes on each count. This is essential to avoid glitches, and it allows detection of direction and relative position. A third optional signal, named index, is used as a reference to establish an absolute position once per rotation.

When to use a Quadrature Decoder

A Quadrature Decoder is used to decode input that senses the current position, velocity and direction of an object (mouse, trackball, robotic axles). It can also be used for precision measurement of speed, acceleration and position of a motor's rotor and with rotary knobs, to determine user input.

Input/Output Connections

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

quad_A – Input (Required)

One of the outputs of the quadrature encoder.

quad_B – Input (Required)

One of the outputs of the quadrature encoder.

index – Input (Optional)

This detects a reference position for the quadrature encoder. If you use an index input, then if inputs A, B, and index are all zero the counter is also reset to zero. Additional logic is typically added to gate the index pulse. Index gating allows the counter to only be reset during one of many possible rotations. An example is a linear actuator that only resets the counter when the far limit of travel has been reached. This limit is signaled by a mechanical limit switch whose output is ANDed with the Index pulse.

clock – Input (Required)

Clock for sampling and glitch filtering the inputs. If glitch filtering is used then the filtered outputs will not change until three successive samples of the input are the same value. For effective glitch filtering, the sample clock period should be greater than the maximum time during which glitching is expected to take place. A counter can be incremented / decremented at a resolution of 1x, 2x, or 4x the frequency of the A and B inputs.

The clock input frequency should be greater than or equal to 10x the maximum A or B input frequency.

interrupt – Output (Required)

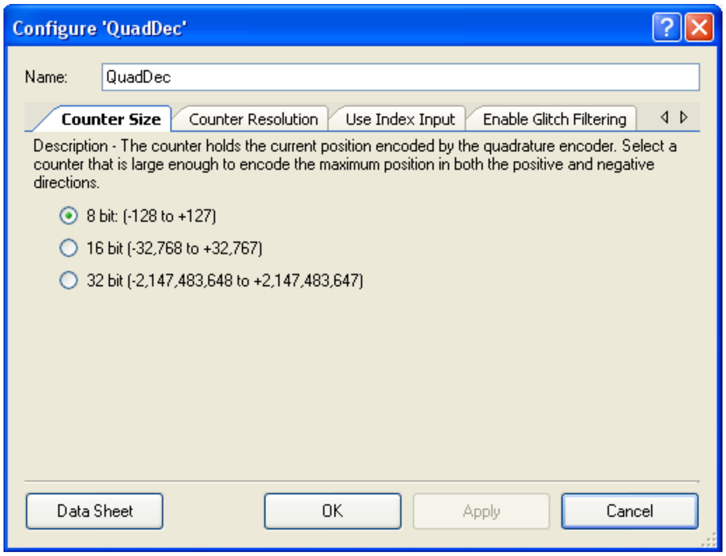
Interrupt on one or more of the following events:

- counter overflow and underflow
- counter reset due to index input (if index is used)
- invalid state transition on the A and B inputs

Parameters and Setup

Drag a Quadrature Decoder component onto your design and double-click it to open the Configure dialog. The dialog contains multiple tabs with categorized parameters.

Counter Size Tab



Configure 'QuadDec'

Name: QuadDec

Description - The counter holds the current position encoded by the quadrature encoder. Select a counter that is large enough to encode the maximum position in both the positive and negative directions.

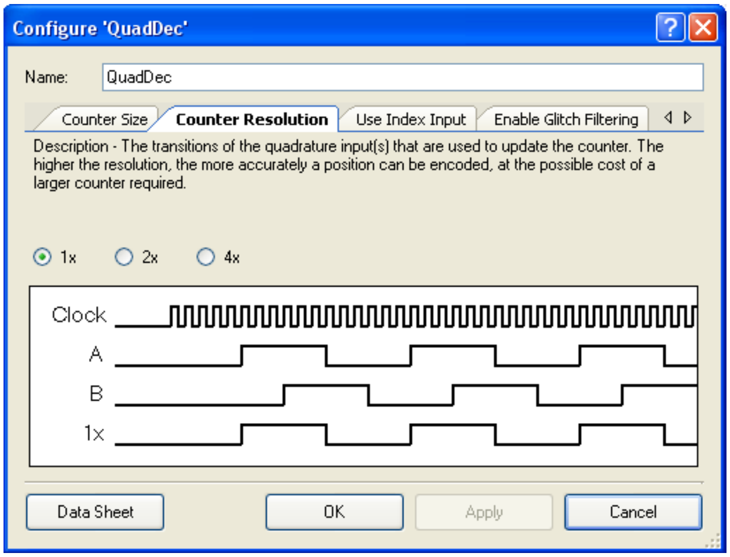
● 8 bit: (-128 to +127)

○ 16 bit (-32,768 to +32,767)

○ 32 bit (-2,147,483,648 to +2,147,483,647)

This defines the counter size, in bits. The counter holds the current position encoded by the quadrature encoder. Select a counter that is large enough to encode the maximum position in both the positive and negative directions. The 32-bit counter implements the lower 16 bits in the hardware counter and the upper 16 bits in software to reduce hardware resource usage. Settings include: 8, 16, or 32 bits.

Counter Resolution Tab



Configure 'QuadDec'

Name: QuadDec

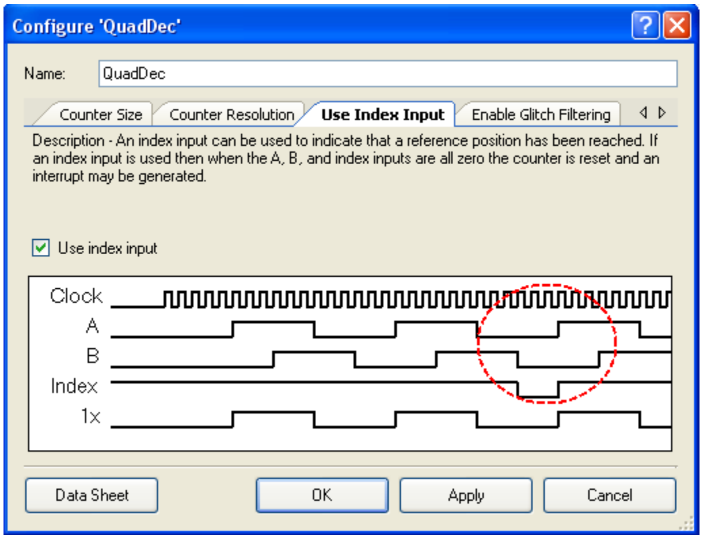
Description - The transitions of the quadrature input(s) that are used to update the counter. The higher the resolution, the more accurately a position can be encoded, at the possible cost of a larger counter required.

● 1x ○ 2x ○ 4x

This tab contains the number of counts recorded in one period of the A and B inputs.

It shows the transitions of the quadrature input signals that are used to update the counter. The higher the resolution, the more accurately the position can be resolved, at the possible cost of a larger counter. Settings include: 1x, 2x, or 4x.

Use Index Input Tab



Configure 'QuadDec'

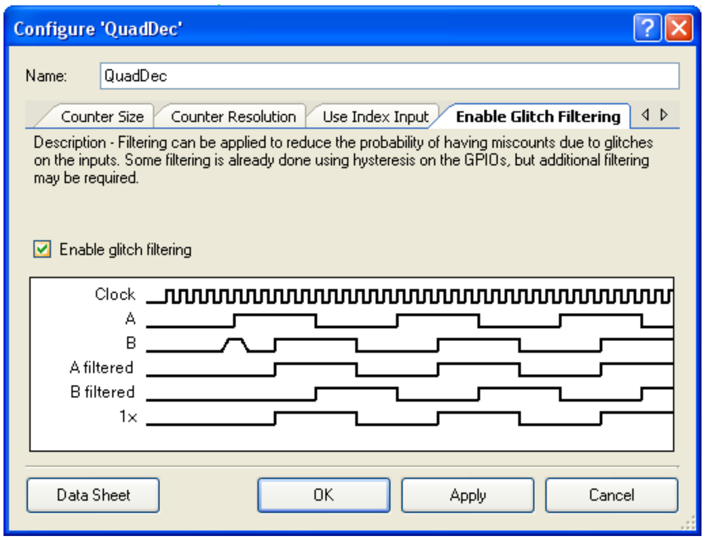
Name: QuadDec

Description - An index input can be used to indicate that a reference position has been reached. If an index input is used then when the A, B, and index inputs are all zero the counter is reset and an interrupt may be generated.

☑ Use index input

This contains a field to indicate whether or not the index input exists and will be used. An index input can be used to indicate that a reference position has been reached. If an index input is used then when the A, B, and index inputs are all zero the counter is reset and an interrupt may be generated. Settings: True or false.

Enable Glitch Filtering Tab



Configure 'QuadDec'

Name: QuadDec

Description - Filtering can be applied to reduce the probability of having miscounts due to glitches on the inputs. Some filtering is already done using hysteresis on the GPIOs, but additional filtering may be required.

☑ Enable glitch filtering

This contains a field to indicate whether or not to apply digital glitch filtering to all inputs. Filtering can be applied to reduce the probability of having miscounts due to glitches on the inputs. Some filtering is already done using hysteresis on the GPIOs, but additional filtering may be required.

If selected, filtering is applied to all inputs. The filtered outputs do not change until three successive samples of the input are the same value. For effective filtering, the period of the sample clock should be greater than the maximum time during which glitching is expected to take place.

Clock Selection

There is one internal clock in this component: the bus clock. It clocks the status register and generates interrupts. A clock source for clocking Quadrature Decoder component must be connected.

Placement

The Quadrature Decoder component is placed in the UDB array and all placement information is provided to the API through the *cyfitter.h* file.

Resources

	Digital Blocks					API Memory (Bytes)		
Resolution	**Datapaths**	**Macro cells**	**Status Registers**	**Control Registers**	**Counter7**	**Flash**	**RAM**	**Pins (per External I/O)**
8-Bits	1		1	0	0			
16-Bits	2		1	0	0			
32-Bits	2		1	0	0			

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 "QuadDec_1" to the first instance of a component in a given design. You can rename it 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.

Function	Description
void QuadDec_1_Start (void)	Initializes UDBs and other relevant hardware.
void QuadDec_1_Stop (void)	Turns off UDBs and other relevant hardware.
int32 QuadDec_1_GetCounter (void)	Reports the current value of the counter

Function	Description
void QuadDec_1_SetCounter (int32 value)	Sets the current value of the counter
uint8 QuadDec_1_GetEvents (void)	Reports the current status of events
void QuadDec_1_SetInterruptMask (uint8 mask)	Enables / disables interrupts due to the events.
uint8 QuadDec_1_GetInterruptMask (void)	Reports the current interrupt mask settings

void QuadDec_1_Start (void)

Description: Initializes UDBs and other relevant hardware. Resets counter to 0, enables or disables all relevant interrupts. Starts monitoring the inputs and counting.

Parameters: None

Return Value: None

Side Effects: None

void QuadDec_1_Stop (void)

Description: Turns off UDBs and other relevant hardware.

Parameters: None

Return Value: None

Side Effects: None

int32 QuadDec_1_GetCounter (void)

Description: Reports the current value of the counter.

Parameters: None

Return Value: int32: The counter value. Return type is signed and per the counter size setting. A positive value indicates clockwise movement (B before A).

Side Effects: None

void QuadDec_1_SetCounter (int32 value)

Description: Sets the current value of the counter.

Parameters: int32 value: The new value. Parameter type is signed and per the counter size setting.

Return Value: None

Side Effects: None

uint8 QuadDec_1_GetEvents (void)

Description: Reports the current status of events.

Parameters: None

Return Value: The events, as bits in an unsigned 8-bit value:

Bit	Description
QuadDec_1_COUNTER_OVERFLOW	Counter overflow.
QuadDec_1_COUNTER_UNDERFLOW	Counter underflow.
QuadDec_1_COUNTER_RESET	Counter reset due to index, if index input is used.
QuadDec_1_INVALID_IN	Invalid A, B inputs state transition.

Side Effects: None

void QuadDec_1_SetInterruptMask (uint8 mask)

Description: Enables / disables interrupts due to the events. For the 32-bit counter, the overflow and underflow interrupts cannot be disabled; these bits are ignored.

Parameters: uint8 mask: Enable / disable bits in an 8-bit value, where 1 enables the interrupt:

Bit	Description
QuadDec_1_COUNTER_OVERFLOW	Enable interrupt due to Counter overflow.
QuadDec_1_COUNTER_UNDERFLOW	Enable interrupt due to Counter underflow.
QuadDec_1_COUNTER_RESET	Enable interrupt due to Counter reset.
QuadDec_1_INVALID_IN	Enable interrupt due to invalid input state transition.

Return Value: None

Side Effects: None

uint8 QuadDec_1_GetInterruptMask (void)

Description: Reports the current interrupt mask settings.

Parameters: None

Return Value: Enable / disable bits in an 8-bit value, where 1 enables the interrupt.
For the 32-bit counter, the overflow and underflow enable bits are always set.

Bit	Description
QuadDec_1_COUNTER_OVERFLOW	Interrupt due to Counter overflow.
QuadDec_1_COUNTER_UNDERFLOW	Interrupt due to Counter underflow.
QuadDec_1_COUNTER_RESET	Interrupt due to Counter reset.
QuadDec_1_INVALID_IN	Interrupt due to invalid A, B inputs state transition.

Side Effects: None

Sample Firmware Source Code

The following is a C language example demonstrating the basic functionality of the QuadDec component. This example assumes the component has been placed in a design with the default name "QuadDec_1."

Note If you rename your component you must also edit the example code as appropriate to match the component name you specify.

```
#include <device.h>

void main()
{
    uint8 stat;
    uint16 count16;
    uint8 i;

    CYGlobalIntEnable;
    LCD_1_Start();
    QuadDec_1_Start();
    QuadDec_1_SetInterruptMask(QuadDec_1_COUNTER_RESET |
         QuadDec_1_INVALID_IN);

    stat = QuadDec_1_GetEvents();
    LCD_1_Position(1, 0);
    LCD_1_PrintInt16(stat);

    while(1)
    {
        CyDelay(150);
        count16 = QuadDec_GetCounter();
        LCD_1_Position(2, 0);
        LCD_1_PrintInt16(count16);
    }
}
```

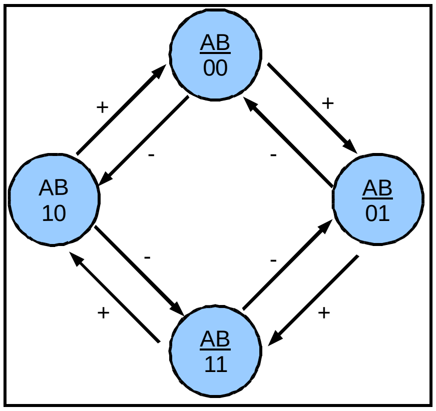
Functional Description

Default Configuration

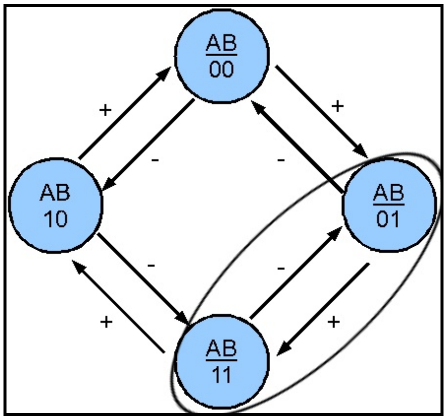
The default configuration for the Quadrature Decoder is 8 – bit Up&Down Counter with 1x resolution.

State Transition

Quadrature phase signals are typically decoded with a state machine and an up/down counter. A conventional decoder has four states, corresponding to all possible values of the A and B inputs. The state transition diagram is shown below (same-state transitions are not depicted). State transitions marked with a "+" and "-" indicate increment and decrement operations on the quadrature phase counter.



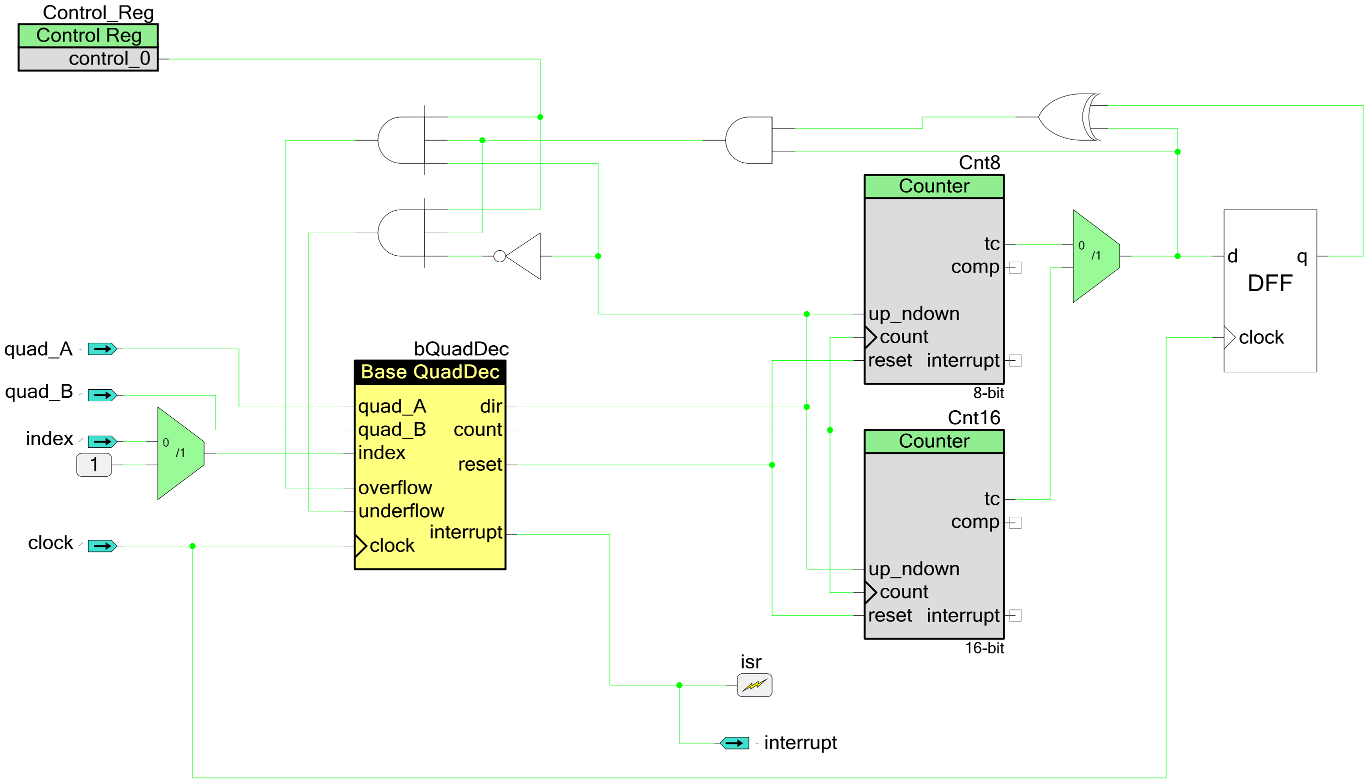
For each full cycle of the quadrature phase signal, the quadrature phase counter changes by four counts. Lower resolution counters can also be used by implementing up/down operations on only a subset of the state transitions. A quarter-resolution decoder is shown below.



All inputs are sampled using a clock signal derived internally within the PSoC3/5 part.

Block Diagram and Configuration

The Quadrature Decoder is only available as a UDB configuration of blocks. The API is described above and the registers are described here to define the overall implementation of the Quadrature Decoder.



Registers

Status

Bits	7	6	5	4	3	2	1	0
Value	reserved				invalid in	reset	underflow	overflow

The status register is a read only register which contains the various status bits defined for the Quadrature Decoder. The value of this register is available with the QuadDec_1_GetEvents() function call. The interrupt output signal is generated from an OR'ing of the masked bit-fields within the status register.

You can set the mask using the QuadDec_1_SetInterruptMask() function call and upon receiving an interrupt you can retrieve the interrupt source by reading the Status register with the QuadDec_1_GetEvents() function call. The Status register is clear on read so the interrupt source is held until the QuadDec_1_GetEvents() function is called. All operations on the status register must use the following defines for the bit-fields as these bit-fields may be moved around within the status register at build time.

There are several bit-fields masks defined for the status registers. Any of these bit-fields may be included as an interrupt source. All bit-fields configured as sticky bits in the status register. The defines are available in the generated header file (.h) as follows:

QuadDec_1_COUNTER_OVERFLOW

Defined as the bit-mask of the Status register bit "Counter overflow".

QuadDec_1_COUNTER_UNDERFLOW

Defined as the bit-mask of the Status register bit "Counter underflow".

QuadDec_1_RESET

Defined as the bit-mask of the Status register bit "Reset due index".

QuadDec_1_INVALID_IN

Defined as the bit-mask of the Status register bit "invalid state transition on the A and B inputs".

DC and AC Electrical Characteristics

The following values are indicative of expected performance and based on initial characterization data.

5.0V/3.3V DC and AC Electrical Characteristics

Parameter	Typical	Min	Max	Units	Conditions and Notes
Input					
Input Voltage Range	---		Vss to Vdd	V	
Input Capacitance	---		---	pF	
Input Impedance	---		---	Ω	
Maximum Clock Rate	---		67	MHz	

Component Changes

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

Version	Description of Changes
1.20	Updated the Configure dialog. Removed the *QuadDec_INT.c* file after compilation if the counter size is less than 32. Removed the checking condition in the *QuadDec_INT.c* file for counter size = 32 bit.

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.