CE209975 - Clock Buffer with Smart IO®

Objective

This example shows how to use a Smart IO® Component in PSoC® 4 to implement a clock buffer that can operate in chip low power modes. It can also be used to drive a heavier load than one GPIO is rated for by replicating the signal and driving two pins.

Overview

This code example demonstrates how an off-chip signal can be replicated using the Smart IO LUTs to drive a heavier load, while operating in chip deep-sleep mode. It also demonstrates how several signals may be logically operated on to generate a signal that triggers a wakeup event through the port interrupt.

Requirements

Tool: PSoC Creator 3.3 SP2

Programming Language: C (GCC 4.9, ARM MDK)

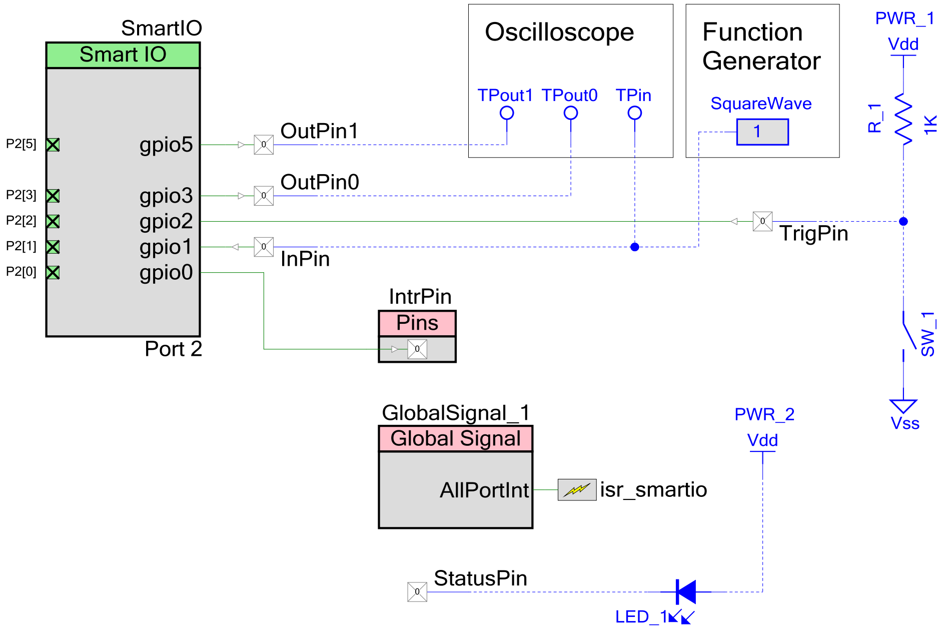
Associated Parts: PSoC 4000S, PSoC 4100S, PSoC 4200L

Related Hardware: CY8CKIT-041, CY8CKIT-046

Design

The design consists of a Smart IO Component with only the pins on the port as its inputs and outputs. It does not use any peripherals or internal chip signals. The Component performs a signal replication function by taking in an external signal (such as an external clock) and driving it out to two pins. This effectively implements a signal buffer functionality. The two pins can then be ganged external to the chip to drive a load that is higher than rated for a single GPIO pin. Figure 1 shows a schematic overview of the design.

Figure 1. Design Schematic



The Smart IO Component is operational during chip deep-sleep mode. This design shows how multiple signals can be used in the Smart IO Component to trigger an interrupt that is the result of a logical operation on those signals.

For the clock buffer, an external signal is input through gpio1, which in turn is connected to InPin. This signal is repeated and output to gpio3 and gpio5. These are connected to OutPin0 and OutPin1 respectively.

For interrupt generation, the Component accepts a digital input signal through gpio2, which is connected to TrigPin. This signal is logically ANDed with gpio1 by using LUT0. The result is output to gpio0, triggers the port interrupt on IntrPin. The IntrPin is configured to generate a port interrupt on a rising edge signal.

A Global Signal Reference Component is used to enable the Combined Port Interrupt (AllPortInt) and configure the ISR of the port interrupt. This is a single resource that triggers for all enabled port interrupt sources in the device. A digital output StatusPin is used to signal the blue LED via firmware for device wakeup indication.

The firmware is implemented in *main.c* and performs the following functions:

1. Starts the isr_smartio interrupt Component and sets up the ISR function.
2. Starts the Smart IO Component.
3. The device enters deep-sleep mode.
4. If the interrupt is triggered, the device wakes up and drives the StatusPin high for 1 second.
5. The StatusPin is driven low and the device re-enters deep-sleep.

Design Considerations

This code example is designed for the specified ports on the stated devices. The design is portable to other PSoC 4 devices with Smart IO, but it may require LUT reconfiguration due to the close relationship between the device port and the Component.

Hardware Setup

1. Connect InPin to a square wave (e.g. from a signal generator) that is under 1 MHz. Connect it also to an oscilloscope.
2. Connect OutPin0 and OutPin1 to an oscilloscope.
3. Connect TrigPin to GND (Alternatively use an external active high switch).

Software Setup

The Smart IO Component is a port-wide resource; you must define its port before it can be used. Follow these steps to configure your design.

CY8CKIT-041

Default port is port 2. No changes necessary.

CY8CKIT-046

1. Open the Smart IO configuration dialog in the design schematic, and define the Port parameter to be 11.
2. Click OK, and close. Rebuild the PSoC Creator project.

Components

Table 1 lists the PSoC Creator Components used in this example, as well as the hardware resources used.

Table 1. List of PSoC Creator Components

Component	Hardware Resources
SmartIO	1 I/O port (PRGIO)
OutPin0, OutPin1, IntrPin, InPin, TrigPin, StatusPin	6 pins

Global Signal Reference	-
isr_smartio	1 interrupt

Parameter Settings

The Smart IO Component is configured in Asynchronous mode, as Figure 2 shows. Figure 3 on page 4 shows the LUT configurations. Only combinatorial elements are used and the block is operational in chip deep-sleep mode.

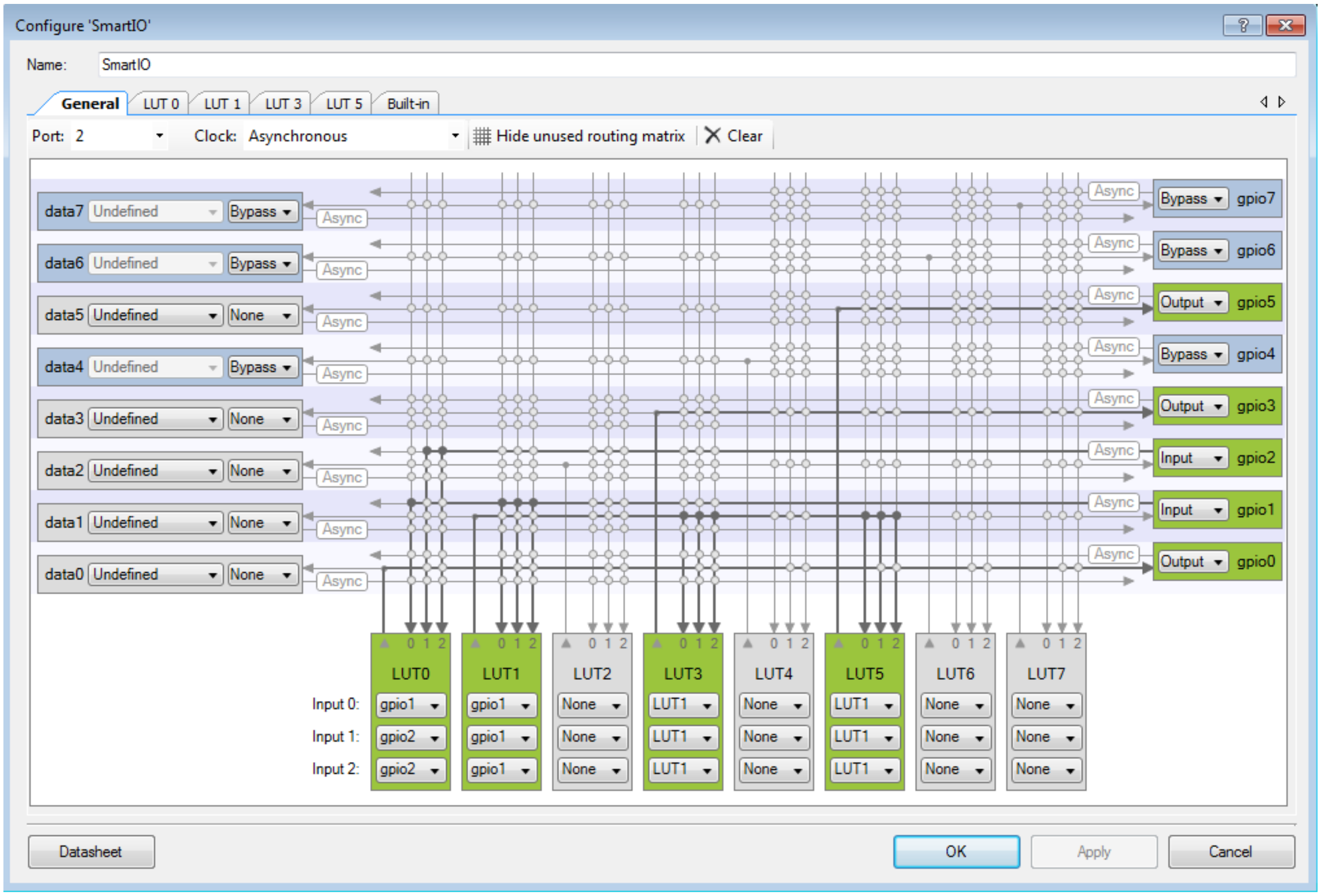
LUT0 accepts gpio1 and gpio2 as inputs. The LUT configuration performs a logical AND of these signals and outputs the result to gpio0. This is used to trigger the wakeup event on the port.

LUT1 is configured to repeat the gpio1 signal. Its output is fed to LUT3 and LUT5. Note that that there are two reasons why two LUTs are used per path (LUT1->LUT3 and LUT1->LUT5):

1. LUT4 to LUT7 cannot accept gpio[3:0] or data[3:0] as inputs. An intermediary LUT must be used.
2. LUT3 can directly accept gpio1 as input but LUT5 cannot. If the design requires that the signals appearing in gpio3 and gpio5 must be in sync, an intermediary LUT should be used to minimize path delay difference.

LUT3 and LUT5 are configured to repeat the LUT1 output signal. These are then output through gpio3 and gpio5 respectively.

Figure 2. Smart IO Routing Configuration



Name: SmartIO

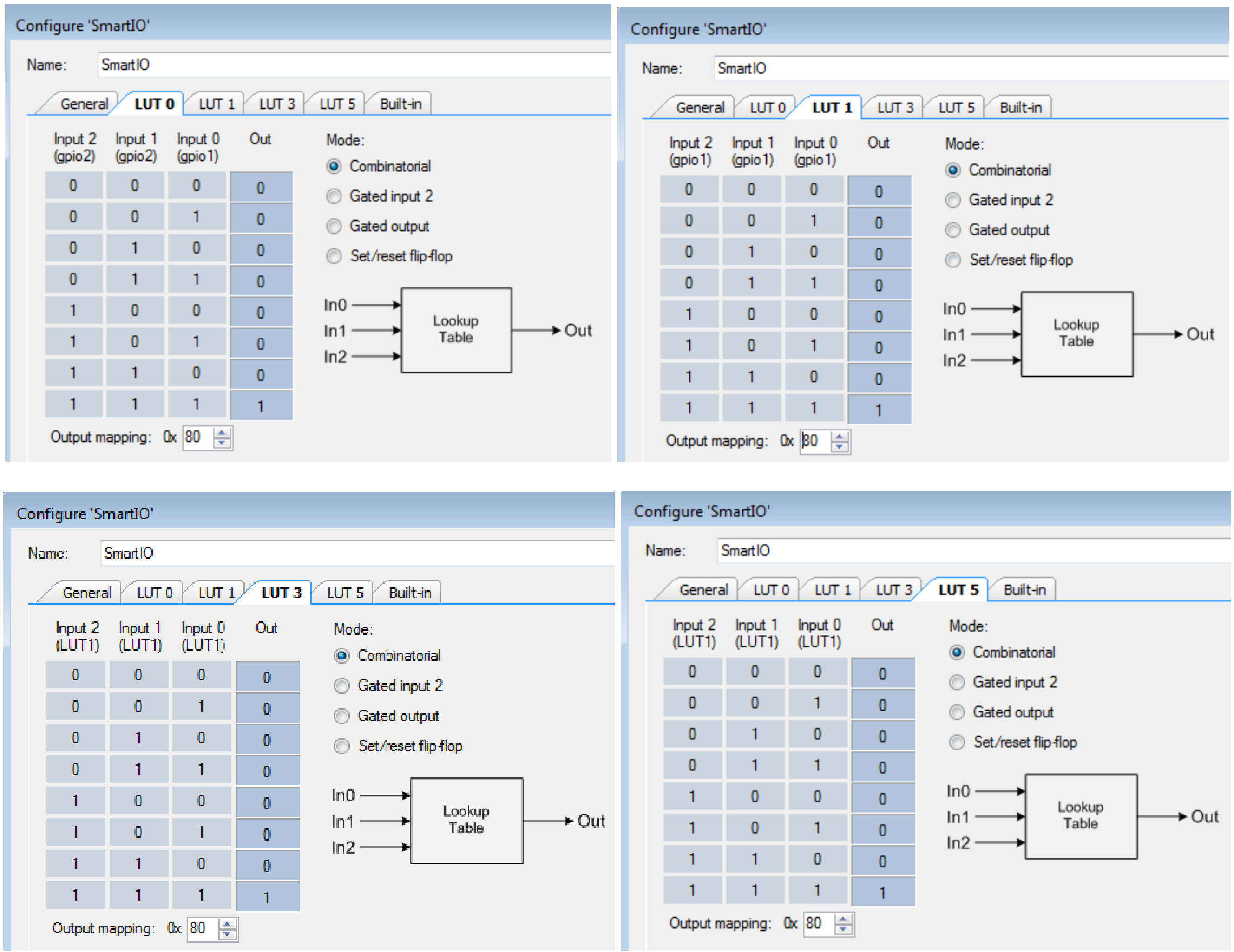
Port: 2
Clock: Asynchronous

data7	Undefined	Bypass
data6	Undefined	Bypass
data5	Undefined	None
data4	Undefined	Bypass
data3	Undefined	None
data2	Undefined	None
data1	Undefined	None
data0	Undefined	None

Bypass	gpio7
Bypass	gpio6
Output	gpio5
Bypass	gpio4
Output	gpio3
Input	gpio2
Input	gpio1
Output	gpio0

	LUT0	LUT1	LUT2	LUT3	LUT4	LUT5	LUT6	LUT7
Input 0:	gpio1	gpio1	None	LUT1	None	LUT1	None	None
Input 1:	gpio2	gpio1	None	LUT1	None	LUT1	None	None
Input 2:	gpio2	gpio1	None	LUT1	None	LUT1	None	None

Figure 3. Smart IO LUT Configuration



Configure 'SmartIO'

Name: SmartIO

General | **LUT 0** | LUT 1 | LUT 3 | LUT 5 | Built-in

Input 2 (gpio2)	Input 1 (gpio2)	Input 0 (gpio1)	Out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Mode:
● Combinatorial
○ Gated input 2
○ Gated output
○ Set/reset flip-flop

Output mapping: 0x 80

Configure 'SmartIO'

Name: SmartIO

General | LUT 0 | **LUT 1** | LUT 3 | LUT 5 | Built-in

Input 2 (gpio1)	Input 1 (gpio1)	Input 0 (gpio1)	Out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Mode:
● Combinatorial
○ Gated input 2
○ Gated output
○ Set/reset flip-flop

Output mapping: 0x 80

Configure 'SmartIO'

Name: SmartIO

General | LUT 0 | LUT 1 | **LUT 3** | LUT 5 | Built-in

Input 2 (LUT1)	Input 1 (LUT1)	Input 0 (LUT1)	Out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Mode:
● Combinatorial
○ Gated input 2
○ Gated output
○ Set/reset flip-flop

Output mapping: 0x 80

Configure 'SmartIO'

Name: SmartIO

General | LUT 0 | LUT 1 | LUT 3 | **LUT 5** | Built-in

Input 2 (LUT1)	Input 1 (LUT1)	Input 0 (LUT1)	Out
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	0
1	1	0	0
1	1	1	1

Mode:
● Combinatorial
○ Gated input 2
○ Gated output
○ Set/reset flip-flop

Output mapping: 0x 80

Operation

Perform the following steps:

1. Program the PSoC device and observe that the signal going into the InPin is replicated on OutPin0 and OutPin1. These are operational during deep-sleep mode, and the two signals can be externally ganged to drive a higher load.
2. Connect TrigPin to VDD and then connect it back to GND. Alternatively push the external active high switch connected to TrigPin. Observe that the blue LED connected to StatusPin lights up for approximately 1 second. Observe that the Output0 and Output1 continue to operate regardless of chip power mode.

Related Documents

Table 2 lists all relevant application notes, code examples, knowledge base articles, device datasheets, and Component datasheets.

Table 2. Related Documents

Code Examples		
CE209974	PSoC®4 Breathing LED with Smart IO®	Implements a breathing LED using a PWM and a Smart IO component.
CE209976	PSoC®4 SPI Slave Select Inversion with Smart IO®	Inverts the polarity of the SCB SPI slave select signal by using the Smart IO component.
PSoC Creator Component Datasheets		
Smart IO	Supports Smart IO peripheral	
Interrupt	Supports connection of interrupts to peripherals	
Global Signal Reference	Supports global interrupt signals	
Pins	Supports connection of hardware resources to physical pins	
Device Documentation		
PSoC 4 Datasheets		
PSoC 4 Technical Reference Manuals		
Development Kit (DVK) Documentation		
PSoC 4 Kits		

Document History

Document Title: CE209975 - Clock Buffer with Smart IO®

Document Number: 002-09975

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	5136361	RFMS	04/15/16	New code example.

Worldwide Sales and Design Support

Cypress maintains a worldwide network of offices, solution centers, manufacturer's representatives, and distributors. To find the office closest to you, visit us at Cypress Locations.

Products

Automotive	cypress.com/go/automotive
Clocks & Buffers	cypress.com/go/clocks
Interface	cypress.com/go/interface
Lighting & Power Control	cypress.com/go/powerpsoc cypress.com/go/plc
Memory	cypress.com/go/memory
PSoC	cypress.com/go/psoc
Touch Sensing	cypress.com/go/touch
USB Controllers	cypress.com/go/usb
Wireless/RF	cypress.com/go/wireless

PSoC® Solutions

psoc.cypress.com/solutions

PSoC 1 | PSoC 3 | PSoC 4 | PSoC 5LP

Cypress Developer Community

Community | Forums | Blogs | Video | Training

Technical Support

cypress.com/go/support

PSoC is a registered trademark and PSoC Creator is a trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are the property of their respective owners.

Cypress Semiconductor
198 Champion Court
San Jose, CA 95134-1709

Phone : 408-943-2600
Fax : 408-943-4730
Website : www.cypress.com

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, 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.