Please note that Cypress is an Infineon Technologies Company.
The document following this cover page is marked as “Cypress” document as this is the company that originally developed the product. Please note that Infineon will continue to offer the product to new and existing customers as part of the Infineon product portfolio.

Continuity of document content
The fact that Infineon offers the following product as part of the Infineon product portfolio does not lead to any changes to this document. Future revisions will occur when appropriate, and any changes will be set out on the document history page.

Continuity of ordering part numbers
Infineon continues to support existing part numbers. Please continue to use the ordering part numbers listed in the datasheet for ordering.



THIS SPEC IS OBSOLETE

Spec No: 001-26199

Spec Title: AN2345 - GENERAL - SIMPLE METHOD TO GENERATE DIGITAL SIGNALS WITH VARIABLE PHASE SHIFT BETWEEN

Sunset Owner: Rajiv Vasanth Badiger (RJVB)

Replaced By: NONE

General - Simple Method to Generate Digital Signals with Variable Phase Shift Between

AN2345

Author: Victor Kremin, Ryshtun Andrij
Associated Project: Yes
Associated Part Family: CY8C29/27/24xxx
Software Version: PSoC® Designer™ 5.1
Associated Application Notes: No

OBSOLETE

Application Note Abstract

This Application Note describes a simple method to generate digital signals with variable phase shift between the signals in the range of 0 – 360 degrees. The described method is very simple because it requires only one digital block per generated signal. The associated project for two signals is also described.

Introduction

Often it is necessary to obtain several digital signals with some phase shift between them that can be changed during operation. These signals are usually used in quadrature receivers, impedance meters, full-bridge switch regulators, and motor control systems. There are several methods to obtain these signals and solve this task. Each of them has advantages and disadvantages. Let's consider the most common of these methods.

The first of these methods consists of generating needed signals entirely with software. This method does not require hardware and allows flexible control of signal parameters. The disadvantage is this method consumes all the processor's time, which complicates interrupt handling.

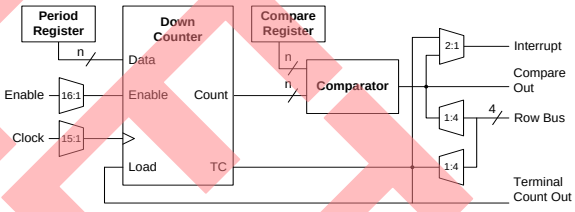
The second of these methods is DDS (Direct Digital Synthesis). This method uses a phase accumulator and DAC. The advantages of this method are high frequency support and phase resolution.

In this Application Note, a simple method to generate digital signals with variable phase shift is described. This method allows generation of up to 16 signals (depending to free PSoC blocks) with phase variation for each of them. The advantage of the described method is it is accomplished with internal hardware and no software intervention. The disadvantage of this method is that, the generator must be stopped for a short period of time when making a change to the phase.

Theory of Generator Operation

To better understand how the generator operates, let's consider the internal details of the counter (see Figure 1). For further details, see the Counter User Module Data Sheet in PSoC Designer™ or the *PSoC Technical Reference Manual (TRM)* on http://www.cypress.com.

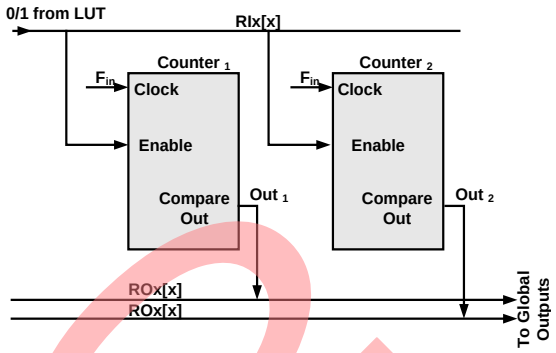
Figure 1. Counter Block Diagram



Once started, the counter operates continuously and reloads its internal value from the Period register upon reaching terminal count. During each clock cycle, the counter compares the current count to the value stored in the Compare register. When the counter is disabled and a period value is written into the Period register, the period value is also loaded into the down counter. When the counter is enabled, the counter counts down until terminal count (a count of 00h) is reached. On the next rising edge of the clock, the period is reloaded and, on subsequent clocks, counting continues.

The generator flowchart is shown in Figure 2.

Figure 2. Generator Flowchart



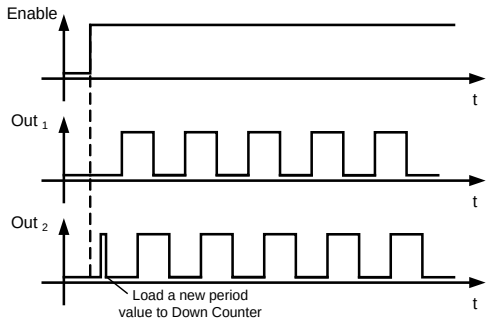
The generator includes two counters: Counter$_1$ and Counter$_2$. During operation, the signals are generated on compare outputs of these counters. The same clock frequency F_{in} is connected to the clock inputs of the counters. The LUT signal is connected to the enable inputs via digital interconnect (not shown in Figure 2). This signal is controlled by software.

The main idea behind the generator lies in setting the first period shorter than the following periods by using a unique feature of counter operation. When the counters are stopped and disabled by a low enable signal, we load two different period values into their Period registers. These values are automatically copied to the down counter upon start of the counter. But because the enable signal is set low, the counters do not operate.

We now write the same period value to both counters. These values are copied to the corresponding down counter at the next underflow cycle. We can enable both counters simultaneously by setting the enable signal high. The counters then start operation, but because they were started from different initial counts, the underflow occurred at different times. Because the Period register contains the same value at this time, the equal output frequency will be generated with constant phase shift.

The timing diagrams, which illustrate this principle, are shown in Figure 3.

Figure 3. Generator Timing Diagram



Initialization of the generator is described as follows:

1. Set the enable signal to 0 and disable the counters.
2. Load the period and compare values into Counter$_1$.
3. Load the phase shift value into the Period register of Counter$_2$.
4. Call Counter_Start function for both counters. The values from the Period register are transferred to the down counter. The counter cannot start to count because there is no enable signal on the enable input.
5. Load the same period and compare values for Counter$_1$ into Counter$_2$.
6. Set the enable signal to 1 and enable the counters.

Source Code and PSoC Internals

The source code for initialization of the generator is shown below:

Code 1.

```
void Gen_start(BYTE Phase_shift)
{
/* Stop counters */
Counter8_1_Stop();
Counter8_1_Stop();

/* LUT to false, Enable to low, counters
  are stopped */
RDI0LT1&=0x0F;

/* Write to first counter period and comp.
  value */
Counter8_1_WritePeriod(199);
Counter8_1_WriteCompareValue(100);

/* Write to second counter phase shift
value */
Counter8_2_WritePeriod(Phase_shift);

/* Start counters */
Counter8_1_Start();
Counter8_2_Start();

/* Write to second counter period and comp.
  value */
Counter8_2_WritePeriod(199);
Counter8_2_WriteCompareValue(100);

/* LUT true, Enable to high, counters are
  started */
RDI0LT1|=0xF0;
}
```

DRAFT

The period value P_{norm} can be calculated from Equation 1:

$$P_{norm} = [\frac{f_{clock}}{f_{des}}] - 1 \qquad \text{Equation 1}$$

f_{clock} is the PWM clock frequency. f_{des} is the PWM output frequency.

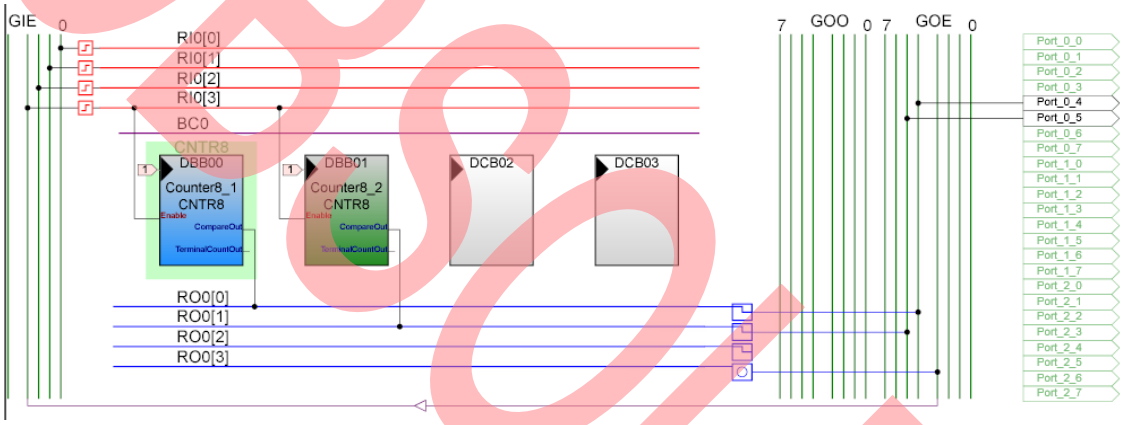
The phase shift value P_{start} can be defined by Equation 2:

$$P_{start} = (P_{norm} + 1)\frac{\Delta\varphi}{360^{\circ}} \qquad \text{Equation 2}$$

P_{norm} is the PWM period for the expected output frequency. $\Delta\varphi$ is the required phase shift, degrees.

The PSoC internal user module placement is shown in Figure 4.

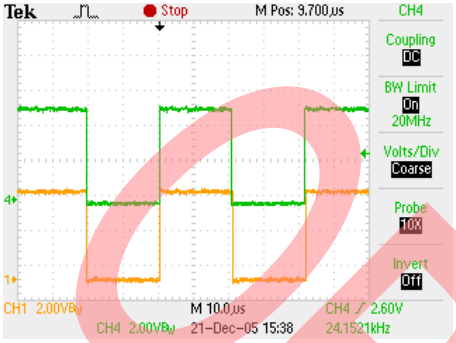
Figure 4. PSoC Internal User Module Configuration



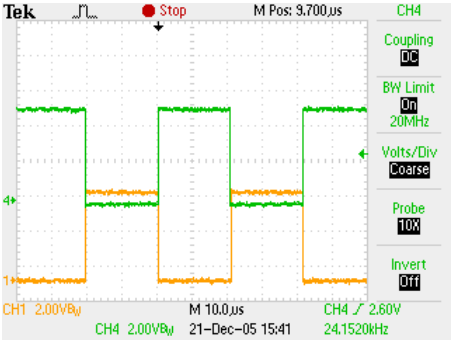
The signal timing diagrams with different phase shifts are shown in the figure below. The upper waveform is the Counter$_1$ output signal. The lower waveform is the Counter$_2$ output signal. P_{norm} = 199.

OBSOLETE

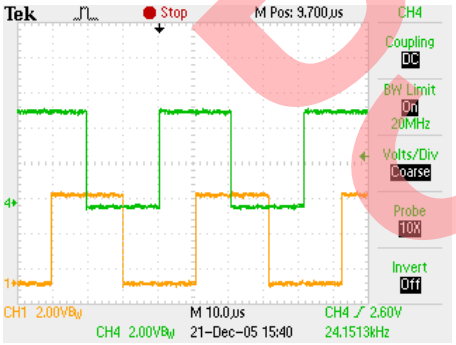
Figure 5. Scope Images



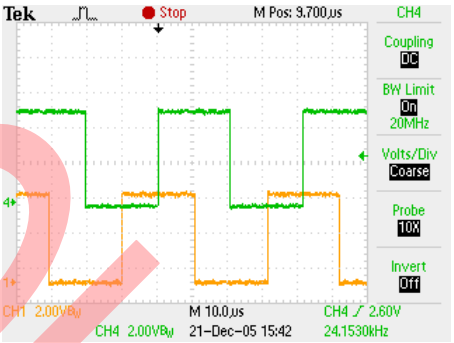
P_{start} =0, Phase Shift=0°



P_{start} =100, Phase Shift=180°



P_{start} =50, Phase Shift=90°



P_{start} =150, Phase Shift=270°

OBSOLETE

Summary

In this Application Note, the method to generate digital signals with variable phase shift between them is considered. This method allows generation of up to 16 signals with dynamic phase variation for each of them. Also, the project with two generated signals is demonstrated.

About the Authors

Name: Victor Kremin
Title: Associate Professor
Background: Victor earned his radiophysics diploma in 1996 from Ivan Franko National Lviv University, his Ph.D. in Computer Aided Design systems in 2000, and is presently working as Associate Professor at National University "Lvivska Polytechnika."
Contact: vkremin@lviv.farlep.net

Name: Ryshtun Andrij
Title: Undergraduate Student
Background: Ryshtun is a 5th-year student pursuing an MS at National University “Lvivs`ka Polytechnika.”
Contact: ryshtun@gmail.com

OBSOLETE

Document History

Document Title: AN2345 - General - Simple Method to Generate Digital Signals with Variable Phase Shift Between

Document Number: 001-26199

Revision	ECN	Orig. of Change	Submission Date	Description of Change
**	1445383	SSFTMP4	09/09/2007	New Spec.
*A	3056758	MEH	10/12/2010	Updated project to PSoC Designer 5.1 Added standard headers in main.c
*B	4169150	RJVB	10/22/2013	Obsolete document. Completing Sunset Review.

OBSOLETE

PSoC is a registered trademark of Cypress Semiconductor Corp. "Programmable System-on-Chip," PSoC Designer, and PSoC Express are trademarks 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
http://www.cypress.com/

© Cypress Semiconductor Corporation, 2007-2013. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, 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 products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

This Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

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.

Use may be limited by and subject to the applicable Cypress software license agreement.