8 位移位寄存器数据表 SHIFTREG8 V 1.0

Copyright © 2009-2010 Cypress Semiconductor Corporation. All Rights Reserved.

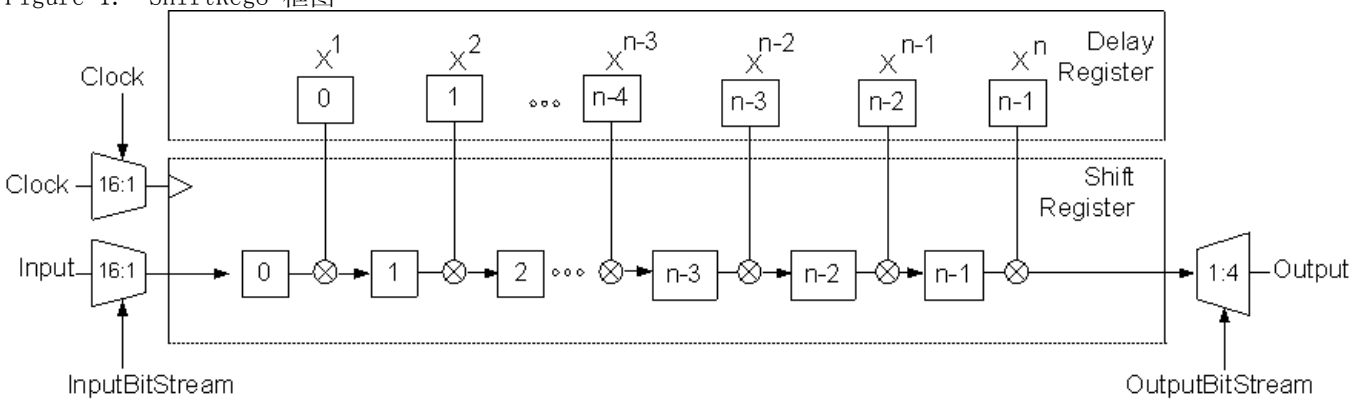
资源	PSoC® 模块			API 存储器 （字节）		引脚 （每个外部 I/O）
	数字	模拟 CT	模拟 SC	闪存	RAM	
CY8C29/27/24/22/21xxx、CY8C23x33、CY8CLED02/04/08/16、CY8CLED03D/04D、CY8CTST110、CY8CTMG110、CY8CTST120、CY8CTMG120、CY8CTMA120、CY8C21x45、CY8C22x45、CY8CPLC20、 CY8CLED16P01						
	0	0	0	25	0	1 到 4
CYWUSB6953	0	0	0	25	0	1 到 4
	0	0	0	25	0	1 到 4

功能和概述

- 8 位移位寄存器使用一个 PSoC 模块
- 源时钟频率最大为 48 MHz
- 时钟的数据输入最大为 24 MHz
- 可编程的延迟循环值
- 串行输出位流

ShiftReg8 用户模块是模块线性反馈移位寄存器 (LFSR) 的变体，可延迟输入位流。 可指定延迟循环值，将延迟的输出定义为最多 8 个 PSoC 模块时钟。

Figure 1. ShiftReg8 框图



功能说明

此用户模块将数字信号移入诸如频移键控 (FSK) 等应用程序。 数字信号在通过此用户模块时，可能会延迟特定 DB 时钟周期。

ShiftReg8 PSoC 模块的延迟寄存器和控制寄存器用于定义和控制生成输出位序列（相对输入有所延迟）。 延迟寄存器是只写的。 延迟寄存器中值为 1 的最高位的位置决定延迟循环的值。

数字 PSoC 模块的基础连接硬件的配置具有协调作用，可协调 PSoC 模块用作单独 ShiftReg8 用户模块时的操作。 延迟寄存器可设置 ShiftReg 的延迟循环，范围为 1 到 8 个 PSoC 模块时钟。

ShiftReg 用户模块要求启动用户模块之前必须先初始化延迟。 寄存器更新时，ShiftReg 用户模块将停止。

直流和交流电气特性

Table 1. CY8C22x45 器件系列的 ShiftReg 交流电电气特性

参数	典型值	限值	单位	条件和注释
最大时钟频率		48[1]	MHz	4.75V < Vdd < 5.25V
		24[1]	MHz	3.0V < Vdd < 4.75V
最大输入频率		24[1]	MHz	4.75V < Vdd < 5.25V
最大输出频率		24[1]	MHz	Vdd=5.0V，24 MHz 输入时钟

电气特性说明

1. 如果输入或输出通过全局总线路由，则频率限制为不超过 12 MHz。

放置

ShiftReg 每 8 位分辨率使用一个数字 PSoC 模块。 每个模块都有一个给定的符号名，器件编辑器会在放置模块的过程中以及放置之后显示该名称。 修饰 模块名称为 SHIFTREG8。

参数和资源

时钟

ShiftReg8 用户模块的时钟由 16 个可能的源之一提供。 全局 I/O 总线可用于将时钟输入连接至外部引脚或其他 PSoC 模块生成的时钟功能。 48 MHz 时钟、CPU_32 kHz 时钟、一个分频时钟 （24V1 或者 24V2）、另一 PSoC 模块输出都可指定为时钟输入。

输入位流

从可用数据路径中选择数据输入源。

输出位流

输出可路由至四个全局输出总线之一，也可被禁用。

长度

循环 循环 下表中提供了可能的。

[+] Feedback

参数	说明
1	输入数据延迟到 1 个 PSoC 模块时钟
2	输入数据延迟到 2 个 PSoC 模块时钟
3	输入数据延迟到 3 个 PSoC 模块时钟
4	输入数据延迟到 4 个 PSoC 模块时钟
5	输入数据延迟到 5 个 PSoC 模块时钟
6	输入数据延迟到 6 个 PSoC 模块时钟
7	输入数据延迟到 7 个 PSoC 模块时钟
8	输入数据延迟到 8 个 PSoC 模块时钟

时钟同步

在 PSoC 器件中，数字模块可以在系统时钟以外还提供其他时钟源。 数字时钟源甚至可以用连锁方式串联起来。 这样就导入了与系统时钟相关的时滞现象。 这些时滞对于 CY8C22x45、CY8C21345 PSoC 器件系列尤为重要，因为它们对数据路径 （尤其是系统总线）进行了各种优化。 此参数可用于控制时钟时滞，确保读取和写入 PSoC 模块寄存器的值时进行正确操作。 此参数的正确值应根据下表决定。

时钟同步值	说明
与 SysClk 同步	此设置值适用于任何由 24 MHz (SysClk) 经过二分频或更多分频所衍生出来的时钟源。 示例包括 VC1、VC2、VC3 （当 VC3 由 SysClk 驱动时）、32KHz 和采用 SysClk 时钟源的数字 PSoC 模块。 外部生成的时钟源还使用此值来确保执行正确的同步操作。
与 SysClk*2 同步	除非所导致的频率为 48 MHz （换句话说，在所有分频器的乘积为 1 时），此设置值可以适用于任何基于 48 MHz (SysClk*2) 的时钟。
直接使用 SysClk	在需要 24 MHz (SysClk/1) 时钟时使用。 虽然此选项并不实际执行同步，但提供了对系统时钟本身的低时滞访问方式。 如果选择此选项，上面的时钟参数设置将被覆盖。 净结果
不同步	在选定 48 MHz (SysClk*2) 输入时使用。 在需要不同步输入时使用。 一般来说，只有在生成中断是计数器的唯一应用时才推荐使用此选项。

应用程序编程接口

SHIFTREG8_Start

说明：

启用 ShiftReg8 用户模块进行操作。 循环

C 原型：

```
void  SHIFTREG8_Start(void)
```

汇编程序：

```
lcall  SHIFTREG8_Start
```


[+] Feedback

参数：

无

返回值：

无

副作用：

启动 ShiftReg8 时，写入种子寄存器的种子值将不会被锁存入移位寄存器。

请参阅 API 一节开头的注意 **。

SHIFTREG8_Stop

说明：

禁用 ShiftReg8 用户模块。

C 原型：

```
void  SHIFTREG8_Stop(void)
```

汇编程序：

```
lcall  SHIFTREG8_Stop
```

参数：

无

返回值：

无

副作用：

将种子值写入种子寄存器会把该种子值锁存入移位寄存器。

请参阅 API 一节开头的注意 **。

SHIFTREG8_SetLength

说明：

循环 寄存器更新时，ShiftReg 用户模块将停止。

C 原型：

```
void  SHIFTREG8_SetLength(BYTE bLength)
```

汇编程序：

```
mov   A, [bLength]
lcall  SHIFTREG8_SetLength
```

参数：

循环 循环 例如，如果设置 bit0，则长度将为 1。 如果设置 bit1，则长度为 2，依此类推。 此长度值通过参数表中所列的符号名传递。 符号名的格式为： ShiftReg8_LENGTH_x，其中 “x” 表示长度（1、2、3 等）

[+] Feedback

符号名	值	说明
SHIFTREG8_LENGTH_1	1	输入数据延迟到 1 个 PSoC 模块时钟
SHIFTREG8_LENGTH_2	2	输入数据延迟到 2 个 PSoC 模块时钟
SHIFTREG8_LENGTH_3	3	输入数据延迟到 3 个 PSoC 模块时钟
SHIFTREG8_LENGTH_4	4	输入数据延迟到 4 个 PSoC 模块时钟
SHIFTREG8_LENGTH_5	5	输入数据延迟到 5 个 PSoC 模块时钟
SHIFTREG8_LENGTH_6	6	输入数据延迟到 6 个 PSoC 模块时钟
SHIFTREG8_LENGTH_7	7	输入数据延迟到 7 个 PSoC 模块时钟
SHIFTREG8_LENGTH_8	8	输入数据延迟到 8 个 PSoC 模块时钟

返回值：

无

副作用：

寄存器更新时，ShiftReg 用户模块将停止。

请参阅 API 一节开头的注意 **。

SHIFTREG8_bReadRegister

说明：

此 API 读取 DR0 寄存器，该寄存器可将 LFSR 值转移到 DR2。 然后，此 API 会读取并返回 DR2 中的值。 调用此 API 之前，必须停止移位寄存器。

C 原型：

```
BYTE  ShiftReg8_bReadRegister (void)
```

汇编程序：

```
lcall  ShiftReg8_bReadRegister
```

参数：

无

返回值：

此函数将返回一个字节，该字节是移位寄存器中的内容。

副作用：

请参阅 API 一节开头的注意 **。

SHIFTREG8_WriteRegister

说明：

此 API 会把传入此 API 的值写入 DR2 寄存器（该值同时也会传输到 LFSR）。 调用此 API 之前，必须停止移位寄存器。

[+] Feedback

C 原型：

```
void  ShiftReg8_WriteRegister(bValue)
```

汇编程序：

```
mov A, [bValue]
lcall  ShiftReg8_WriteRegister
```

参数：

bValue： 这是写入移位寄存器的值。

返回值：

无

副作用：

请参阅 API 一节开头的注意 **。

固件源代码示例

在下例中，C 和汇编代码之间的对应关系非常简单和直接。 显示的周期值和比较值都与基本值相差一，因为寄存器是从零开始的，即零是递减计数循环的终端计数。 汇编程序和 C 语言编译器均通过将简单的单字节参数传输到 A 寄存器 （而不是堆栈）来优化用户模块 API 的性能。 当在 SHIFTREG8.h 文件中遇到 #pragma 快速调用声明时，C 语言编译器会对 “INT” 类型应用此机制，而不会将参数推入堆栈。

展示

```
;***************************************************************************
;  Setup the SHIFTREG8 to generate an 8-bit maximal sequence.
;
;***************************************************************************
include "SHIFTREG8.inc"
export  SetupSHIFTREG8

SetupSHIFTREG8:
    ; load the ShiftReg8 Delay cycle value
    mov   A, SHIFTREG8_LENGTH_8 ; delay 8 PSoC block clocks
    call  SHIFTREG8_SetLength
    ;start the ShiftReg8
    call  SHIFTREG8_Start

    ret
```

同一编码用 C 语言表示如下。

```
#include "SHIFTREG8.h"

void SetupSHIFTREG8(void)
{
// load the ShiftReg8 Delay cycle value
SHIFTREG8_SetLength(SHIFTREG8_LENGTH_8);
// start the ShiftReg8
SHIFTREG8_Start();
}
```


[+] Feedback

配置寄存器

通过寄存器对 ShiftReg8 用户模块进行个性化和参数化设置。 下文中的表格显示了常量寄存器值，以及已命名位域形式的参数。表格附带有简要描述。 寄存器符号名在用户模块实例的 C 语言和汇编语言接口文件 （后缀名为 “.h” 和 “.inc” 的文件）中定义。

Table 2. SHIFTREG8_FUNC_REG

位	7	6	5	4	3	2	1	0
值	0	0	1	0	0	0	1	0

Table 3. SHIFTREG8_INPUT_REG

位	7	6	5	4	3	2	1	0
值	InputBitStream[3:0]				Clock[3:0]			

“ 输入位流 ” 从 16 个源中选择数据输入。 “ 时钟 ” 从 16 个源中选择时钟输入。 上述两个参数均在器件编辑器中设定。

Table 4. SHIFTREG8_OUTPUT_REG

位	7	6	5	4	3	2	1	0
值	辅助时钟		0	0	0	0	输出位流	

器件编辑器中的用户模块 “ 时钟同步 ” 参数决定了 “ 辅助时钟 ” 位的值。 “ 输出位流 ” 从四个全局总线之一选择输出。 此参数在器件编辑器中设定。

Table 5. SHIFTREG8_DELAY_REG

位	7	6	5	4	3	2	1	0
值	循环							

循环 循环 延迟寄存器是只写的。 循环

Table 6. SHIFTREG8_CONTROL0_REG

位	7	6	5	4	3	2	1	0
值	0	0	0	0	0	0	0	启用

设定 “ 启用 ” 位表明启用了 ShiftReg8。 循环

[+] Feedback

版本历史记录

版本	创作者	说明
1.0	DHA	初始版本

Note PSoC Designer 5.1 在所有的用户模块数据表中提供版本历史记录。 本数据表详细介绍了当前和先前用户模块版本之间的区别。

Copyright © 2009-2010 Cypress Semiconductor Corporation. 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.

PSoC Designer™ and Programmable System-on-Chip™ are trademarks and PSoC® is a registered trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any 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.