

硬核处理器系统 (HPS) 提供两个串行外设接口 (SPI) 主器件和两个 SPI 从器件。SPI 主器件和从器件是 Synopsys® DesignWare® 同步串行接口 (SSI) 控制器 (DW\_apb\_ssi) 的实例。

## SPI 控制器的功能

SPI 控制器具有以下功能：

- 串行主和串行从控制器 - 通过串行-主或串行-从外设器件使能串行通信。
- 串行接口操作 - 从可编程的以下协议中进行选择：
  - Motorola SPI 协议
  - Texas Instrument 同步串行协议
  - National Semiconductor Microwire
- 集成了 HPS DMA 控制器的 DMA 控制器接口
- SPI 主器件支持 rxd 采样延迟
- 发送和接收 FIFO 缓冲器是 256 个字深
- SPI 主器件支持高达 4 个从器件选择
- 可编程的主器件串行位速率
- 可编程的 4 到 16 位的数据项容量

## SPI 结构图和系统集成

SPI 支持 32 位的数据总线宽度。

© 2012 Altera Corporation. All rights reserved. ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS and STRATIX words and logos are trademarks of Altera Corporation and registered in the U.S. Patent and Trademark Office and in other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders as described at [www.altera.com/common/legal.html](http://www.altera.com/common/legal.html). Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

Portions © 2011 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

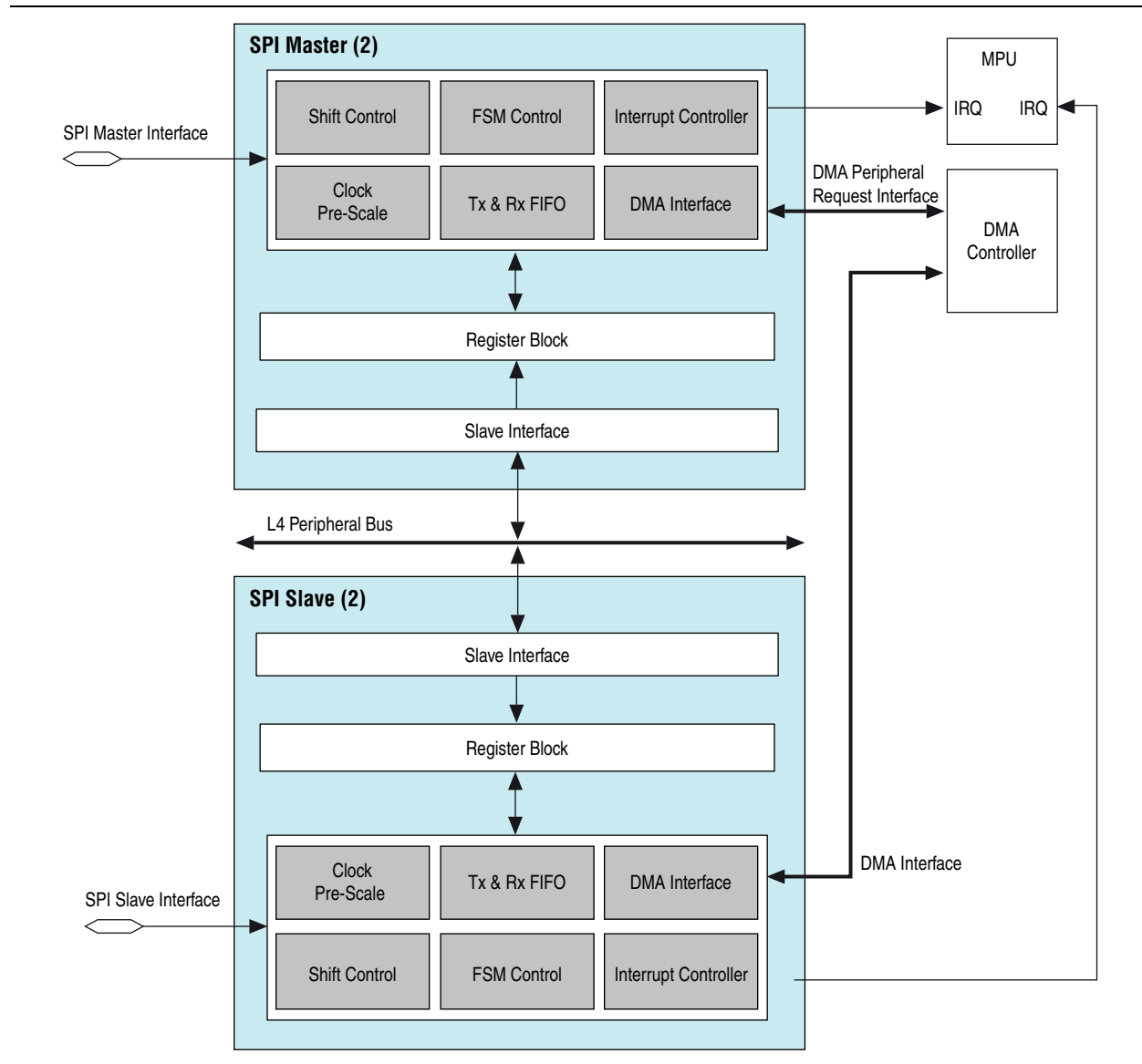
†Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.



## SPI 结构图

图 19 - 1 显示了 SPI 模块的主要接口的功能组合。

图 19 - 1. SPI 结构图



SPI 模块的主要接口的功能组合如下：

- 系统总线接口
- DMA 外设请求接口
- 中断接口
- SPI 接口

## SPI 控制器的功能说明

### 协议详细信息和标准兼容

该章节介绍 SPI 控制器的功能操作。

主机处理器通过系统总线接口访问关于 SPI 控制器的数据、控制和状态信息。SPI 也与 DMA 控制器连接：

HPS 包含两个通用 SPI 主控制器和两个通用 SPI 从控制器。

SPI 控制器可以使用以下任何协议连接其它任何 SPI 器件：

- Motorola SPI 协议
- Texas Instrument 串行协议 (SSP)
- National Semiconductor Microwire 协议

### SPI 控制器概述

为了使 SPI 控制器连接到串行-主或串行-从外设器件，外设必须至少具有以下其中一个接口：

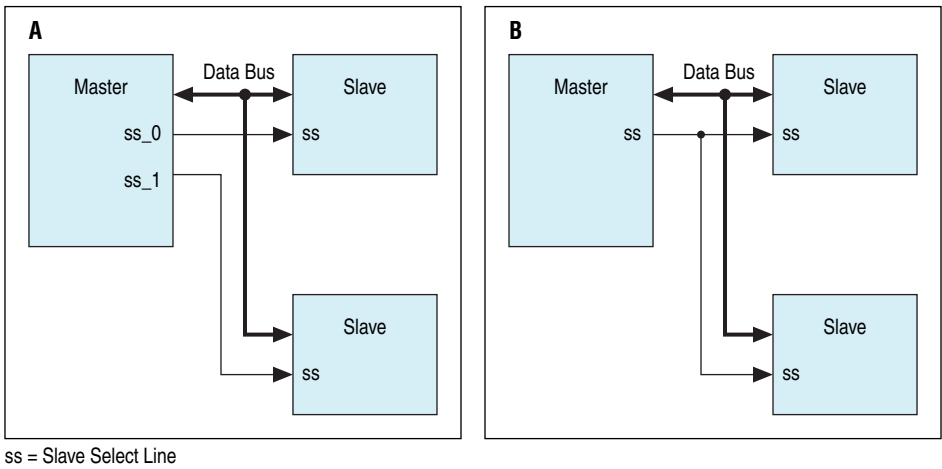
- Motorola SPI 协议 - 来自 Motorola 的四线，全双工串行协议。当 SPI 控制器是闲置或禁用时，从器件选择线保持高电平。要了解更多信息，请参考第 19 - 11 页的“[Motorola SPI 协议](#)”。
- Texas Instrument 串行协议 (SSP) - 一个四线、双工串行协议。用于 SPI 和 Microwire 协议的从器件选择线也可用作 SSP 协议的帧指示标识。要了解更多信息，请参考第 19 - 12 页的“[Texas Instrument 同步串行协议 \(SSP\)](#)”。
- National Semiconductor Microwire - 一个半双工串行协议，使用从串行主器件发送到目标串行从器件的控制字。要了解更多信息，请参考第 19 - 13 页的“[National Semiconductor Microwire 协议](#)”。您可以编程控制寄存器 0 (CTRLR0) 中的 FRF (帧格式) 位域来选择使用哪个协议。

SPI 控制器支持的串行协议可实现通过使用硬件对串行从器件选择或寻址。串行从器件在专用硬件选择线的控制下被选择。串行主器件生成的选择线数等于总线上出现的串行从器件数。数据传输开始之前，串行-主器件置位目标串行从器件的选择线。该体系结构显示在图 19 - 2 的 A 部分。

当在软件中实现时，所有串行从器件的输入选择线应该产生于串行主器件的单一从器件选择输出。该模式中假设串行主器件只有一个单一从器件选择输出。如果系统中有多多个串行主器件，那么所有主器件的从器件选择输出可以逻辑与以生成所有串行从器件的一个单一从器件选择输入。

软件域中的主编程控制目标从器件的选择；该体系结构显示在图 19 - 2 的 B 部分。软件会控制哪一个从器件响应主器件的串行传输请求。

图 19 - 2. 硬件 / 软件从器件选择

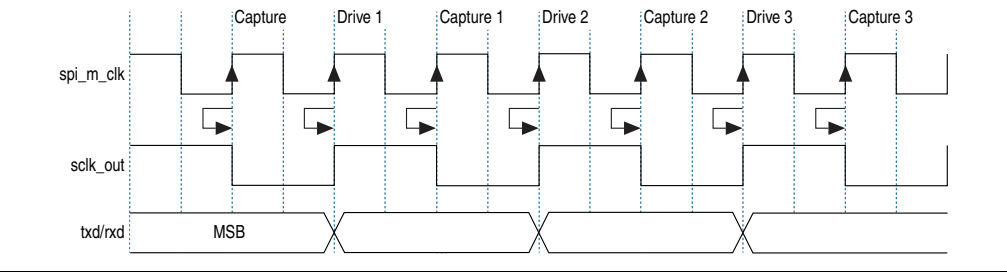


串行比特率时钟

SPI 主器件比特率时钟

SPI 主器件比特率时钟 (sclk\_out) 的最高频率是 SPI 主器件时钟 (spi\_m\_clk) 频率的一半。这使得移位控制逻辑能够采集 sclk\_out 的时钟沿的数据并且传播相反沿的数据，如图 19 - 3 所示。只有当激活的传输在进行时，sclk\_out 线才触发。其它所有时间，它处于未激活状态 (如其进行操作的串行协议中所定义)。

图 19 - 3. 最高 sclk\_out/spi\_m\_clk Ratio



sclk\_out 频率可从公式 19 - 1 得出，其中  $\langle SPI\ clock \rangle$  是主 SPI 模块的 spi\_m\_clk 以及从 SPI 模块的 l4\_main\_clk。

公式 19 - 1.

$$F_{sclk\_out} = F_{\langle SPI\ clock \rangle} / SCKDV$$

SCKDV 是寄存器 BAUDR 中的一个比特域，持有从 2 到 65,534 的任何偶数值。如果 SCKDV 是 0，那么 sclk\_out 被禁用。

公式 19-2 介绍比特率时钟  $sclk\_out$  和 SPI 主器件外设时钟之间的频率比率限制。SPI 主器件外设时钟的频率必须至少是片外主器件时钟的两倍。

#### 公式 19-2. SPI 主器件外设时钟

$$F_{spi\_m\_clk} \geq 2 \times (\text{最高 } F_{sclk\_out}) +$$

#### SPI 从器件比特率时钟

$l4\_main\_clk$  的最低频率取决于从器件外设的操作。如果从器件是 *receive only*，那么  $l4\_main\_clk$  的最低频率是主器件 ( $sclk\_in$ ) 比特率时钟的最高预期频率的 6 倍。 $sclk\_in$  信号双同步到  $l4\_main\_clk$  域，然后被边沿检测；该同步需要 3 个  $l4\_main\_clk$  周期。

如果从器件是 *transmit 和 receive*，那么  $l4\_main\_clk$  的最低频率是主器件 ( $sclk\_in$ ) 比特率时钟的最高预期频率的 8 倍。这确保主器件 rxd 线的数据在主器件移位控制逻辑采集数据之前处于稳定。

比特率时钟  $sclk\_in$  和 SPI 从器件外设时钟之间的频率比率限制如下所示：

- 从器件（只接收）： $F_{l4\_main\_clk} \geq 6 \times (\text{最高 } F_{sclk\_in})$
- 从器件： $F_{l4\_main\_clk} \geq 8 \times (\text{最高 } F_{sclk\_in})$

#### 发送和接收 FIFO 缓冲器

存在两个 16-bit FIFO 缓冲器，一个发送 FIFO 缓冲器和一个接收 FIFO 缓冲器，具有 256 的深度。容量上少于 16 bit 的数据帧写入发送 FIFO 缓冲器时必须右对齐。移位控制逻辑自动右对齐接收 FIFO 缓冲器中的接收数据。

FIFO 缓冲器中的每个数据入口包含一个单数据帧。单 FIFO 缓冲器位置中存储多个数据帧是不可能的；例如，不可以在一个单 FIFO 缓冲器位置中存储两个 8-bit 数据帧。如果需要一个 8-bit 数据帧，那么当串行移位器发送数据时，FIFO 缓冲器入口的较高 8-bit 被忽略或不被使用。



当 SPI 控制器被禁用 (SSIENR=0) 或复位时，发送和接收 FIFO 缓冲器被清零。要了解关于复位信号的更多信息，请参考 *Cyclone® V 器件手册* 第 3 卷的 *Reset Manager* 章节。

发送 FIFO 缓冲器通过写命令被加载到 SPI 数据寄存器 (DR)。数据通过移位控制逻辑从发送 FIFO 缓冲器移到发送移位寄存器。当 FIFO 缓冲器的入口数少于或等于 FIFO 缓冲器阈值时，发送 FIFO 缓冲器生成一个发送 FIFO 空中断请求。通过寄存器 TXFTLR 设置的阈值决定中断被生成的 FIFO 缓冲器入口的程度。阈值使您能够提前指示处理器发送 FIFO 缓冲器几乎为空。如果您尝试将数据写入已满的发送 FIFO 缓冲器，那么发送 FIFO 上溢中断被生成。

数据通过读命令从接收 FIFO 缓冲器移到 SPI 数据寄存器 (DR)。接收 FIFO 缓冲器通过移位控制逻辑从接收移位寄存器加载。当 FIFO 缓冲器的入口数大于或等于 FIFO 缓冲器阈值加 1 时，接收 FIFO 缓冲器生成接收 FIFO 满中断请求。通过寄存器 RXFTLR 设置的阈值决定中断被生成的 FIFO 缓冲器入口的程度。

阈值使您能够提前指示处理器接收 FIFO 缓冲器几乎为满。当接收移位逻辑尝试将数据加载到全满接收 FIFO 缓冲器时，接收 FIFO 溢出中断被生成。然而，最新接收到的数据丢失。如果您尝试从空接收 FIFO 缓冲器读取，那么接收 FIFO 下溢中断被生成。这会提醒处理器读数据无效。

## SPI 中断

SPI 控制器支持可以被屏蔽的合并中断请求。合并中断请求是屏蔽后的所有其它 SPI 中断的 ORed 结果。所有 SPI 中断都有有效高极性电平。SPI 中断的描述如下：

- 发送FIFO空中断 - 发送FIFO缓冲器等于或低于其阈值并且需要执行以防止下溢时设置。通过软件可编程寄存器设置的阈值决定中断被生成的发送 FIFO 缓冲器入口的水平。当数据写入发送 FIFO 缓冲器时该中断由硬件清除，从而使该缓冲器高于阈值水平。
- 发送FIFO上溢中断 - 主器件尝试将数据写入完全填充后的发送FIFO缓冲器时设置。设置时，新数据写操作被丢弃。该中断保持设置状态直到您读取发送 FIFO 上溢中断清除寄存器 (TXOICR)。
- 接收FIFO满中断 - 当接收FIFO缓冲器等于或高于其阈值加1并且需要执行以防止上溢时设置。通过软件可编程寄存器设置的阈值决定中断被生成的接收 FIFO 缓冲器入口的水平。当数据从接收 FIFO 缓冲器读取时该中断由硬件清除，从而使该缓冲器低于阈值水平。
- 接收 FIFO 上溢中断 - 当接收逻辑尝试将数据放入完全填充后的接收 FIFO 缓冲器设置。设置时，新接收的数据被丢弃。该中断保持设置状态直到您读取接收 FIFO 上溢中断清除寄存器 (RXOICR)。
- 接收FIFO下溢中断 - 当系统总线访问尝试从空的接收FIFO缓冲器读取时设置。设置时，零从接收 FIFO 缓冲器读回。该中断保持设置状态直到您读取接收 FIFO 下溢中断清除寄存器 (RXUICR)。
- 合并中断请求 - 屏蔽以上所有中断请求后导致 ORed。要屏蔽该中断信号，您必须屏蔽所有其它 SPI 中断请求。

发送 FIFO 上溢、发送 FIFO 空、接收 FIFO 满、接收 FIFO 下溢和接收 FIFO 上溢中断均可通过使用中断屏蔽寄存器 (IMR) 被单独地屏蔽。

## 传输模式

当传输串行总线上的数据时，SPI 控制器在该部分讨论的模式中操作。

传输模式 (TMOD) 通过写入控制寄存器 0 (CTRLR0) 而设置。



传输模式设置不影响串行传输的双工。微总线传输忽略 TMOD，它由 MWCR 寄存器控制。

## 发送和接收

当 TMOD = 0 时，发送和接收逻辑均有效。根据所选的帧格式（串行协议），数据传输正常进行。发送数据从发送 FIFO 缓冲器移除并且通过 txd 线发送到目标器件，该目标器件会返回 rxd 线的数据。目标器件的接收数据在每个数据帧结束时从接收移位寄存器移到接收 FIFO 缓冲器。

## 只发送

当 TMOD = 1 时，所有接收数据均被忽略。根据所选的帧格式（串行协议），数据传输正常进行。发送数据从发送 FIFO 缓冲器移除并且通过 txd 线发送到目标器件，该目标器件会返回 rxd 线的数据。数据帧结束时，接收移位寄存器不将最新接收的数据加载到接收 FIFO 缓冲器。接收移位寄存器中的数据由下一个传输所覆盖。当进入该模式时，应该屏蔽来自接收逻辑的中断。

## 只接收

当 TMOD = 2 时，发送数据无效。SPI 从器件的情况中，发送 FIFO 缓冲器在 Receive Only 模式中从不被移除。传输期间，txd 输出保持在恒定逻辑电平。根据所选帧格式（串行协议），数据传输正常进行。每个数据帧结束时，目标器件的接收数据从接收移位寄存器移到接收 FIFO 缓冲器。当进入该模式时，应该屏蔽来自发送逻辑的中断。

## EEPROM 读



该传输模式只对串行主器件有效。

当 TMOD = 3 时，发送数据用于发送一个 opcode 和 / 或一个地址到 EEPROM 器件。通常该行为使用 3 个数据帧（8-bit 高位地址和 8-bit 低位地址紧跟 8-bit opcode）。opcode 和地址的传输期间，接收逻辑没有采集数据（只要 SPI 主器件发送 txd 线的数据，rxn 线的数据就会被忽略）。SPI 主器件继续发送数据直到发送 FIFO 缓冲器为空。因此，发送 FIFO 缓冲器中只能有足够的数据帧以对 EEPROM 提供 opcode 和地址。如果发送 FIFO 缓冲器中的数据帧多于所需的，那么读数据丢失。

当发送 FIFO 缓冲器为空时（所有控制信息已经被发送），那么接收线（rxn）的数据有效并且被存储在接收 FIFO 缓冲器中；txd 输出保持在恒定逻辑电平。串行传输继续进行直到 SPI 主器件接收的数据帧数匹配 CTRL1 寄存器中 NDF 域的值加 1。



当 SPI 控制器被控制为 SSP 模式时，EEPROM 读模式不被支持。

## SPI 主器件

SPI 主器件通过串行-从外设器件启动和控制所有串行传输。第 19-2 页的图 19-1 显示了一个 SPI 主器件。

SPI 控制器生成和控制的串行比特率时钟在 scln\_out 线上驱动。当 SPI 控制器被禁用时，串行传输不会进行并且 scln\_out 保持在“未激活”状态（如其进行操作的串行协议所定义）。

## RXD 采样延迟

SPI 主器件能够延迟 rxn 信号的默认采样时间，以便提高串行总线上的最高可实现频率。

来自主器件的 scln\_out 信号和来自从器件的 rxn 信号的双向布线延迟意味着 rxn 信号的时序（如主器件观测）**已经离开正常采样时间**。

没有 RXD 采样延迟的情况下，必须提高传输的 baud 速率，以便确保 rxn 信号的建立时间在范围内。这样会降低串行接口的频率。

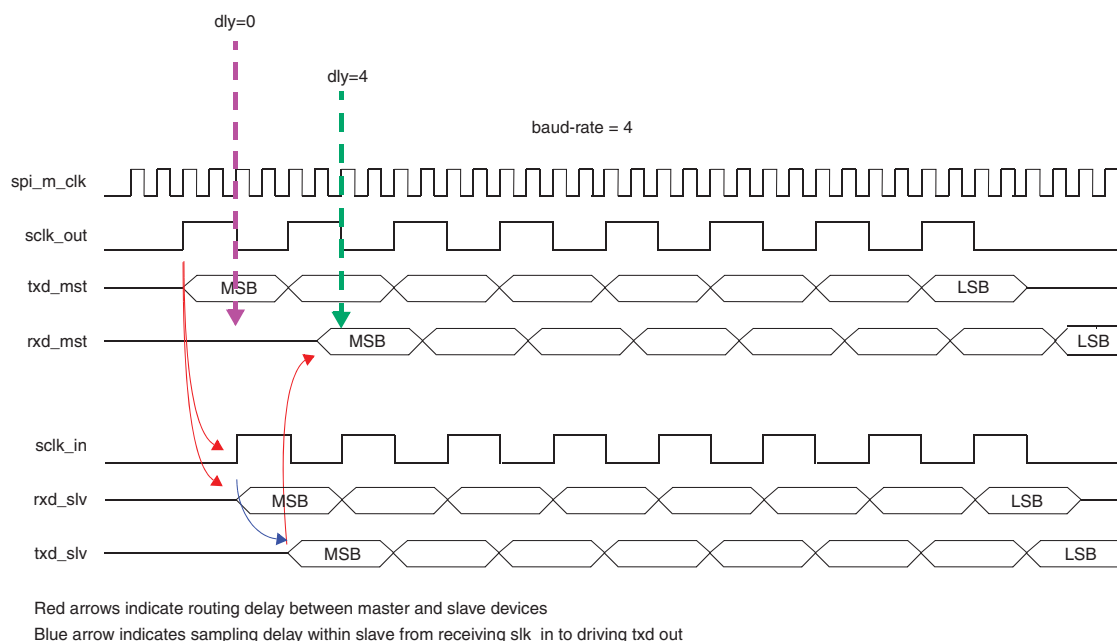
额外的逻辑包含在 SPI 主器件中以便延迟 rxn 信号的默认采样时间。该额外逻辑有助于提高串行总线上的最高可实现频率。

通过写入 RX 采样延迟寄存器（rx\_sample\_dly）的 rsd 域，可以使用高达 64 个周期的 spi\_m\_clk 时钟周期数指定应用到 rxn 采样的额外延迟量。如果 rsd 域使用超过 64 的值被编程，那么零延迟应用于 rxn 采样。



来自主器件的 `sclk_out` 信号和来自从器件的 `rx_d` 信号的双向布线延迟意味着 `rx_d` 信号的时序 (如主器件观测), **已经离开正常采样时间**。图 19-4 显示了该情况。红箭头表示主器件和从器件之间的布线延迟。蓝箭头表示从接收 `sclk_in` 到驱动 `tx_d` 的从器件内的采样延迟。

图 19-4. `sclk_out` 信号的往返布线延迟的影响



## 数据传输

当符合以下所有条件时, SPI 主器件开始数据传输:

- SPI 主器件被使能
- 发送 FIFO 缓冲器中至少有一个有效入口
- 一个从器件被选择

当有效地传输数据时, 状态寄存器 (SR) 中的忙碌标记 (BUSY) 被设置。尝试新的串行传输之前, 必须等待直到忙碌标记被清零。



当数据被写入发送 FIFO 缓冲器时, BUSY 状态不被设置。只有当目标从器件已经被选择并且传输在进行时, 该位才被设置。将数据写入发送 FIFO 缓冲器后, 直到 `sclk_out` 信号的正边沿出现移位逻辑才开始串行传输。等待该正边沿的延迟取决于串行传输的 baud 速率。轮询 BUSY 状态之前, 应该首先轮询 Transit FIFO Empty (TFE) 状态 (等待 1) 或等待 (BAUDR \* SPI clock) 时钟周期。

## 主器件 SPI 和 SSP 串行传输

第 19-11 页的 “Motorola SPI 协议” 和第 19-12 页的 “Texas Instrument 同步串行协议 (SSP)” 分别介绍 SPI 和 SSP 串行协议。

当传输模式是 “发送和接收” 或 “只发送” (分别为 `TMOD = 0` 或 `TMOD = 1`) 时, 当发送 FIFO 缓冲器为空时, 传输由移位控制逻辑中止。要进行连续地数据传输, 必须确保所有数据被发送之前发送 FIFO 缓冲器不为空。**发送 FIFO 阈值水平** (TXFTLR) 可用于提前中断 (发送 FIFO 空中断) 处理器, 从而表明发送 FIFO 缓冲器几乎为空。



当同时使用 DMA 和 SPI 主器件时，**发送数据水平** (DMATDLR) 可用于提前请求 DMA 控制器，表明发送 FIFO 缓冲器几乎为空。然后 FIFO 缓冲器可被重新填充数据来继续串行传输。使能串行从器件之前，用户可能将一个数据块（至少两个 FIFO 缓冲器入口）写入发送 FIFO 缓冲器。这确保直到组成连续传输的数据帧数出现在发送 FIFO 缓冲器中，串行传输才开始。

当传输模式是“只接收” (TMOD = 2) 时，选择了串行从器件时，串行传输通过将一个“哑”数据字写入发送 FIFO 缓冲器而开始。串行传输期间，SPI 控制器的 txd 输出保持在持续逻辑电平。串行传输期间，发送 FIFO 缓冲器仅在开始时被移除并且会保持为空。串行传输结束时由控制寄存器 1 (CTRLR1) 中“数据帧数” (NDF) 域控制。

例如，如果想要从串行-从器件外设接收 24 个数据帧，那么应该使用值 23 编程 NDF 域；当接收的帧数等于 NDF 值加 1 时，接收逻辑中止串行传输。因为发送 FIFO 缓冲器在传输期间从不需要被执行，所以该传输模式提高系统总线的带宽。每次接收 FIFO 缓冲器生成一个 FIFO 满中断请求时接收 FIFO 缓冲器应该被读取以防止上溢。

当传输模式是“eeprom\_read” (TMOD = 3) 时，当一个串行从器件 (EEPROM) 被选择时，串行传输通过将 opcode 和 / 或地址写入发送 FIFO 缓冲器而开始。opcode 和地址被发送到 EEPROM 器件，之后读数据从 EEPROM 器件接收并且存储到接收 FIFO 缓冲器。串行传输结束时由控制寄存器 1 (CTRLR1) 中的 NDF 域控制。



当 SPI 控制器配置为 SSP 模式时，EEPROM 读模式不被支持。

**接收 FIFO 阈值水平** (RXFTLR) 可用于提前指示接收 FIFO 缓冲器几乎为满。当使用 DMA 时，**接收数据水平** (DMARDLR) 可用于提前请求 DMA 控制器，从而表明接收 FIFO 缓冲器几乎为满。

## 主器件 Microwire 串行传输

第 19 - 13 页的“National Semiconductor Microwire 协议”详细介绍 Microwire 串行协议。

SPI 串行主器件的 Microwire 串行传输由 Microwire 控制寄存器 (MWCR) 控制。MHS 位域使能和禁用 Microwire 握手接口。MDD 位域控制数据帧的方向（控制帧总是由主器件发送以及由从器件接收）。MWMOD 位域定义传输是有序的还是无序的。

当发送 FIFO 缓冲器中至少有 1 个控制字并且从器件使能时，所有 Microwire 传输通过 SPI 串行主器件而开始。当 SPI 主器件发送数据帧 (MDD = 1) 时，发送 FIFO 缓冲器为空时，传输由移位逻辑中止。当 SPI 主器件接收数据帧 (MDD = 1) 时，传输的中止取决于 MWMOD 位域的设置。如果传输是无序的 (MWMOD = 0)，从从器件移入数据帧后，当发送 FIFO 变空时，传输会被终止。当传输有序时 (MWMOD = 1)，接收的数据帧数等于 CTRLR1 寄存器中的值加 1 时，它被移位逻辑中止。

当 SPI 主器件的握手接口使能 (MHS = 1) 时，目标从器件的状态在传输后被轮询。只有当从器件报告一个 *ready status* 时，SPI 主器件才完成传输并且清除 BUSY 状态。如果传输是连续的，那么下一个控制 / 数据帧直到从器件返回一个 *ready status* 才被发送。

## SPI 从器件

SPI 从器件处理串行主外设器件启动和控制的传输的串行通信。

图 19 - 5 显示了单-主器件总线系统中的一个 SPI 从器件实例，包括以下信号：

- sclk\_in—SPI 从器件的串行时钟
- ss\_in\_n—SPI 从器件的从器件选择输入

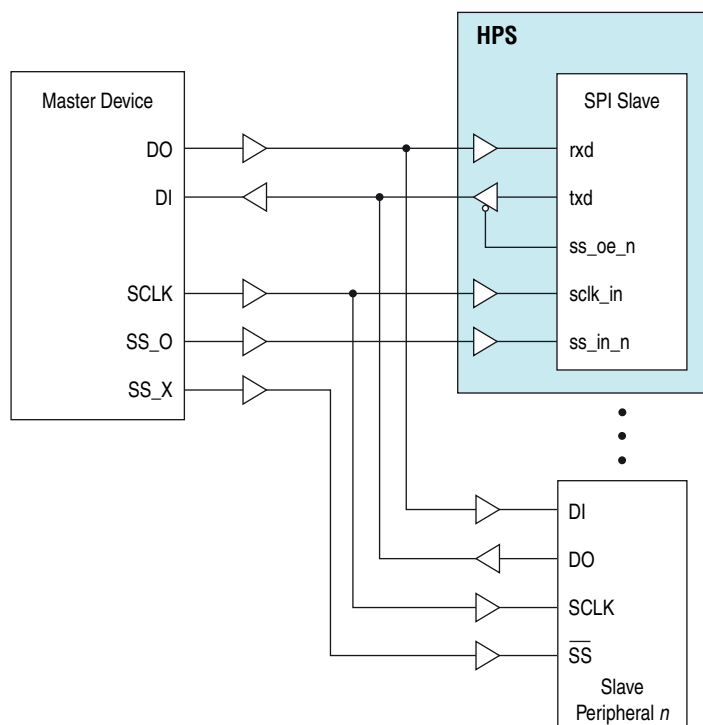
- `ss_oe_n`—SPI 主器件或从器件的输出使能
- `txd`—SPI 主器件或从器件的发送数据线
- `rxn`—SPI 主器件或从器件的接收数据线

当 SPI 串行从器件被选择时，它使其 `txd` 数据传输到串行总线。所有传输到和来自串行从器件的数据均在串行时钟线 (`sclk_in`) 上调节，从 SPI 主器件驱动。数据从串行从器件的串行时钟线的一个边沿传播并且在相反的边沿被采样。

当 SPI 串行从器件不被选择时，它不可以干扰串行-主和其它串行-从器件之间的数据传输。当串行从器件不被选择时，其 `txd` 输出被缓冲，从而导致 SPI 主器件 `rxn` 线的高阻抗驱动。图 19-5 所示的缓冲器在 SPI 控制器的外部。`spi_oe_n` 是 SPI 从器件输出使能信号。

调节数据传输的串行时钟由串行-主器件和 SPI 从器件的 `sclk_in` 的输入生成。从器件保持闲置状态直到被总线主器件所选。当没有有效地发送数据时，从器件必须保持其 `txd` 线处于高阻抗状态以避免干扰到其它从器件的串行传输。SPI 从器件输出使能 (`ss_oe_n`) 信号可用于控制 `txd` 输出缓冲器。只要从器件被选择，它就会继续将数据传输到主器件以及传输来自主器件的数据。如果主器件发送到所有串行从器件，SPI 控制寄存器 0 (CTRLR0) 中的一个控制位 (SLV\_OE) 会被编程以便通知从器件它是否应该响应 `txd` 线的数据。

图 19-5. SPI 从器件



## 从器件 SPI 和 SSP 串行传输

第 19-11 页的“Motorola SPI 协议”和第 19-12 页的“Texas Instrument 同步串行协议 (SSP)”分别包含 SPI 和 SSP 串行协议的说明。

如果 SPI 从器件是 *receive only* (TMOD=2)，那么发送 FIFO 缓冲器不需要包含有效数据，因为每次从器件被选择时，发送移位寄存器中的当前数据会被重新发送。TMOD=2 时，状态寄存器 (SR) 中的 TXE 错误标记不被设置。当使用该模式时，您应该屏蔽发送 FIFO 空中断。

如果 SPI 从器件发送数据到主器件，那么必须确保传输由串行-主器件发起前数据出现在发送 FIFO 缓冲器。当没有数据出现在发送 FIFO 缓冲器时，如果主器件发起到 SPI 从器件的传输，那么错误标记 (TXE) 在 SPI 状态寄存器中设置，并且之前发送的数据帧在 txd 上重新发送。为了实现连续地数据传输，必须确保发送 FIFO 缓冲器在所有数据被发送前不为空。**发送 FIFO 阈值水平寄存器** (TXFTLR) 可用于提前中断（发送 FIFO 空中断）处理器，从而表明发送 FIFO 缓冲器几乎为空。当使用 DMA 控制器时，**DMA 发送数据水平寄存器** (DMATDLR) 可用于提前请求 DMA 控制器，从而表明发送 FIFO 缓冲器几乎为空。然后 FIFO 缓冲器可被重新填充数据以便继续进行串行传输。

每次接收 FIFO 缓冲器生成一个 FIFO 满中断请求时，接收 FIFO 缓冲器应该被读取以防止上溢。接收 FIFO 阈值水平寄存器 (RXFTLR) 可用于提前指示接收 FIFO 缓冲器几乎为满。当使用 DMA 控制器时，**DMA 接收数据水平寄存器** (DMARDLR) 可用于提前请求 DMA 控制器，从而表明接收 FIFO 缓冲器几乎为满。

## 串行传输

第 19 - 13 页的 “National Semiconductor Microwire 协议” 详细介绍微总线串行协议，包括时序图以及串行传输之前和之后在发送和接收 FIFO 缓冲器中如何组织数据的信息。微总线协议的操作和 SPI 协议大致相同。**不存在由 SPI 从器件解码的控制帧。**

## 伙伴连接接口

通过使用以下部分中讨论的其中一个接口，SPI 可以连接到所有串行-主或串行-从外设器件。

### Motorola SPI 协议

串行时钟的未激活状态为低电平。数据帧的长度可以为 4 到 16 bit。

数据传输在从器件选择信号的下降沿开始。第一个数据位由主和从外设设在串行时钟的第一个沿上采集；因此，有效数据必须出现在第一个串行时钟沿之前的 txd 和 rxd 线。



只有当用作从器件 SPI 时，从器件选择信号才生效。对于主器件 SPI，输出使能信号一旦被置低，数据传输就开始。

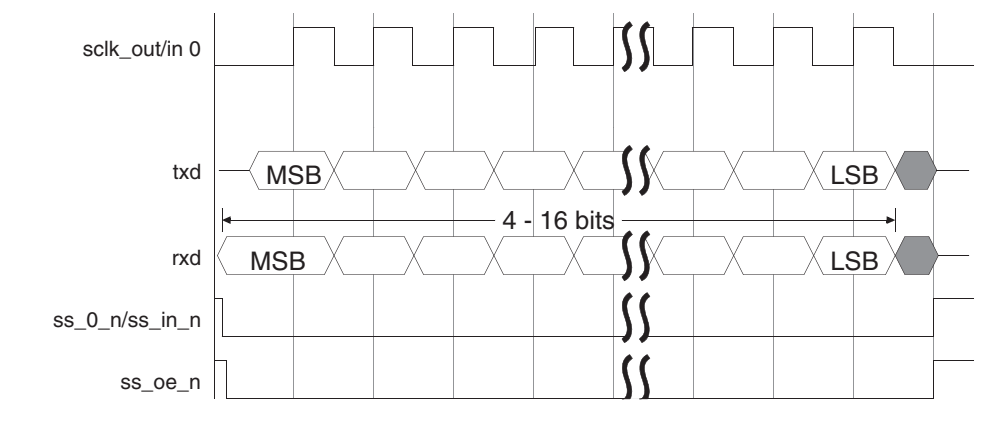
图 19 - 6 显示了单 SPI 数据传输的时序图。

以下信号在该部分的时序图中显示：

- sclk\_out—SPI 主器件的串行时钟
- sclk\_in—SPI 从器件的串行时钟
- ss\_0\_n—SPI 主器件的从器件选择信号
- ss\_oe\_n—SPI 主器件或从器件的输出使能

- txd—SPI 主器件或从器件的发送数据线
- rxd—SPI 主器件或从器件的接收数据线

图 19-6. SPI 串行格式



对于执行 SPI 串行传输，SPI 控制器上有 4 个可能的传输模式，请参考第 19-6 页的“传输模式”。对于发送和接收传输（控制寄存器 0 = 0 的传输模式域 (9:8)），从 SPI 控制器发送到外部串行器件的数据被写入发送 FIFO 缓冲器。从外部串行器件接收的到 SPI 控制器的数据被送入接收 FIFO 缓冲器。

对于 *transmit only* 传输（控制寄存器 0 = 1 的传输模式域 (9:8)），从 SPI 控制器发送到外部串行器件的数据被写入发送 FIFO 缓冲器。因为从外部串行器件接收的数据被视为无效，所以它不被存储在 SPI 接收 FIFO 缓冲器中。

对于 *receive only* 传输（控制寄存器 0 = 2 的传输模式域 (9:8)），从 SPI 控制器发送到外部串行器件的数据无效，所以一个单哑字被写入发送 FIFO 缓冲器以开始串行传输。串行传输期间，SPI 控制器的 txd 输出被保持在恒定逻辑电平。从外部串行器件接收的到 SPI 控制器的数据被送到接收 FIFO 缓冲器。

对于 *eeprom\_read* 传输（控制寄存器 0 = 3 的传输模式域 [9:8]），opcode 和 / 或 EEPROM 地址被写入发送 FIFO 缓冲器。这些控制帧的传输期间，接收的数据不被 SPI 主器件采集。控制帧被发送后，EEPROM 的接收数据被存储在接收 FIFO 缓冲器中。

### Texas Instrument 同步串行协议 (SSP)

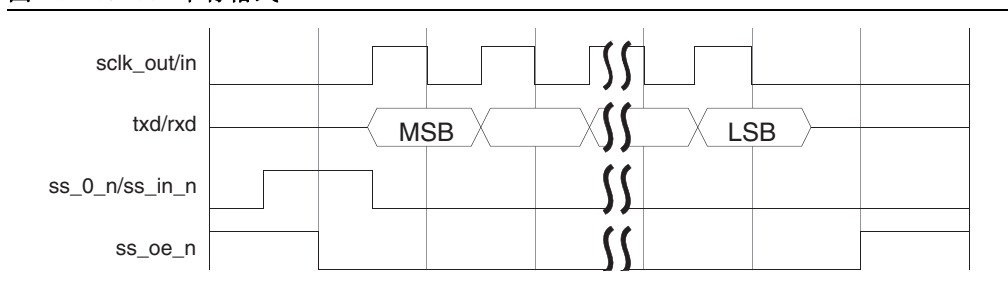
通过置位帧指示标识线 (`ss_0_n`) 一个串行时钟周期，数据传输开始。要发送的数据在一个串行时钟周期后被驱动到 txd 线；从器件的相似数据也被驱动到 rxd 线。数据在串行时钟的 (`sclk_out/sclk_in`) 的上升沿被传播并且在下降沿被采集。数据帧的长度范围是 4 到 16 位。



只有当用作从器件 SPI 时，从器件选择信号 (`ss_0_n`) 才生效。对于主器件 SPI，输出使能信号一被置低，数据传输就会开始。

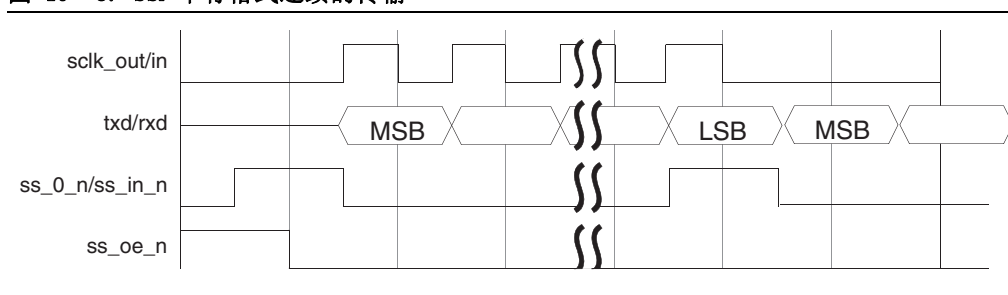
图 19-7 显示了单 SSP 串行传输的时序图。

图 19-7. SSP 串行格式



连续的数据帧的传输方式和单数据帧的相同。帧指示器在和当前传输的 LSB 的相同周期期间置位一个时钟周期，从而指示另一个数据帧紧跟其后。图 19-8 显示了连续的 SSP 传输的时序。

图 19-8. SSP 串行格式连续的传输



## National Semiconductor Microwire 协议

对于主器件 SPI，输出使能信号一被置低，数据传输就会开始。一半串行时钟 (sclk\_out) 周期后，控制的第一个位在 txd 线被发送出。控制字的长度范围可以是 1 到 16 bit 并且通过写入 CTRLR0 中的位域 CFS (bit 15:12) 而设置。控制字的其余部分通过 SPI 串行主器件而发送 (在 sclk\_out 的下降沿传播)。该传输期间，没有数据出现在 (高阻抗) 串行主器件的 rxd 线。

数据字的方向由 Microwire Control Register (MWCR) 中 MDD 位域 (bit 1) 控制。当 MDD=0 时，这表示 SPI 串行主器件从外部串行从器件接收数据。控制字的 LSB 被发送后的一个时钟周期，从外设使用 0 bit，以及紧跟其后的长度可以为 4 到 16 bit 的数据帧进行响应。数据在串行时钟的下降沿被传播并且在上升沿被采集。

来自 Microwire 协议的连续的传输可以是有序或无序的，并且由 MWCR 中的 MWMOD 位域 (bit 0) 控制。

通过下一个传输的控制字紧跟在当前数据字的 LSB 之后，无序的连续传输发生。

执行连续的无序传输所需的唯一修改是将更多控制字写入发送 FIFO 缓冲器。

有序连续传输期间，只有一个控制字从 SPI 主器件发送。该传输的开始方式和从无序读操作开始的方式相同，但是周期继续进行以进一步读取数据。从器件自动地对下一个位置递增其地址指针并且继续从该位置提供数据。所有的位置数都可以使用该方式读取；当接收的字数等于 CTRLR1 寄存器的值加 1 时，SPI 主器件中止传输。

当 MDD = 1 时，这表示 SPI 串行主器件发送数据到外部串行从器件。控制字的 LSB 被发送后，SPI 主器件立即开始发送数据帧到从外设。

 SPI 控制器不支持连续的有序 Microwire 写操作，其中 MDD = 1 以及 MWMOD = 1。

通过下一个传输的控制字紧跟在当前数据字的 LSB 之后，连续的传输发生。

Microwire 握手接口可以被使能以实现 SPI 主器件写操作传输到外部串行从器件。要使能握手接口，必须将 1 写入 MWCR 寄存器的 MHS 位域 (bit 2)。当 MHS 被设置为 1 时，完成传输，或发送下一个控制字进行继续传输之前，SPI 串行主器件检查从器件的就绪状态。

**第一个数据字发送到串行-从器件后，SPI 主器件轮询 rxd 输入从而等待从器件的就绪状态。一接收到就绪状态，SPI 主器件就开始下一个控制字的传输。最后数据帧的传输完成后，SPI 主器件在完成传输前发送一个起始位以清除从器件的就绪状态。**

在 SPI 从器件中，数据传输从器件选择信号 (ss\_in\_0) 的下降沿开始。一半的串行时钟 (sclk\_in) 周期后，控制的第一个位出现在 rxd 线。控制字的长度范围可以是 1 到 16 bit 并且通过在 CTRLR0 寄存器中写入位域 CFS 而设置。CFS 位域必须设置为预期的串行主器件的控制字的容量。其余的控制字由 SPI 串行从器件接收（在 sclk\_in 的上升沿采集）。该接收期间，没有数据驱动（高阻抗）在串行从器件的 txd 线。

数据字的方向由 MDD 位域 (bit 1) MWCR 寄存器控制。当 MDD=0 时，这表示 SPI 串行从器件要接收外部串行主器件的数据。控制字被发送后，串行主器件就立即开始将数据帧驱动到 SPI 从器件 rxd 线。数据在串行时钟的下降沿被传播并且在上升沿被采集。从器件-选择信号在传输期间保持有效低电平并且在数据传输后的一半的时钟周期后被置低。SPI 从器件输出使能信号在传输期间保持未激活状态。

当 MDD=1 时，这表示 SPI 串行从器件发送数据到外部串行主器件。控制字的 LSB 被发送后，SPI 从器件立即在 txd 线上发送一个亚 0 bit，以及紧跟着的 4- 到 16-bit 数据帧。

SPI 从器件的连续传输和对 SPI 主器件指定的连续传输的进行方式相同。SPI 从器件不支持握手接口，因为不存在忙碌周期。

图 19-9 显示了单 SPI 串行主器件读取外部串行从器件的时序图。

图 19-9. 单 SPI 串行主器件 Microwire 串行传输 (MDD=0)

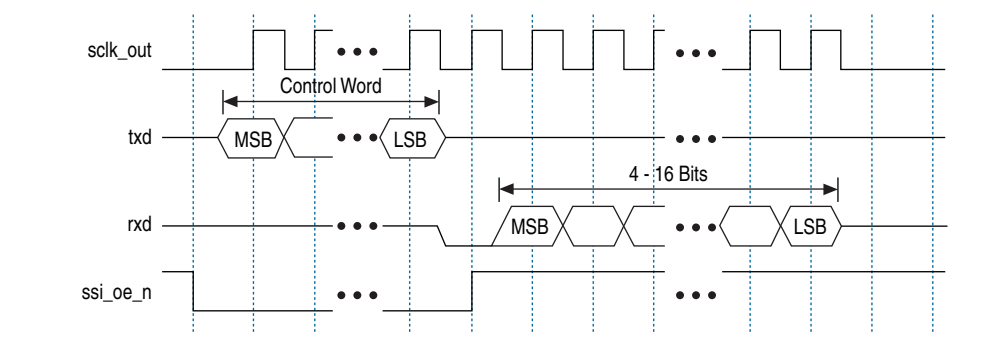
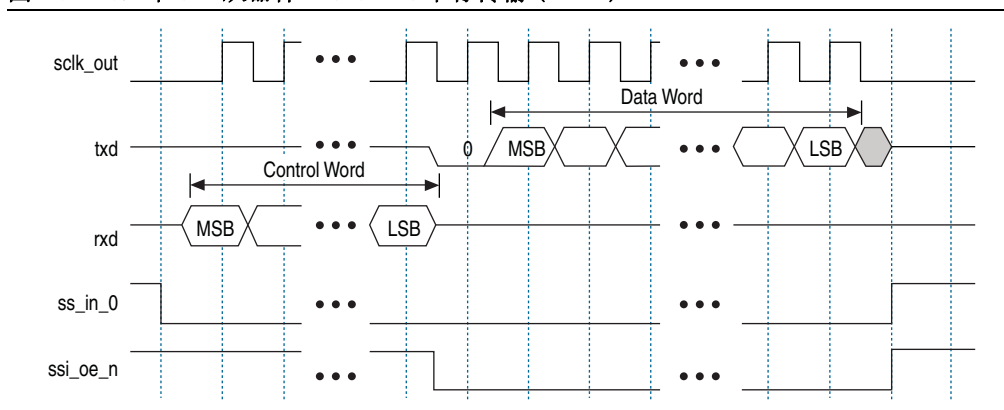




图 19-10 显示了单 SPI 串行从器件写入外部串行主器件的时序图。

图 19-10. 单 SPI 从器件 Microwire 串行传输 (MDD=1)



## DMA 控制器接口

SPI 控制器支持 DMA 发信号以指示接收 FIFO 缓冲器的数据什么时候可读或发送 FIFO 缓冲器什么时候需要数据。它需要两个 DMA 通道，发送数据通道和接收数据通道。SPI 控制器可以发出单或突发 DMA 传输并且接受 DMA 的突发确认信号。通过将相应值编程到阈值寄存器，系统软件可以触发 DMA 突发模式。阈值寄存器值的典型设置是半满。

要使能 SPI 控制器的 DMA 控制器接口，必须写入 DMA 控制寄存器 (DMACR)。写入 1 到 DMACR 寄存器的 TDMAE 位域使能 SPI 发送握手接口。写入 1 到 DMACR 寄存器的 RDMAE 位域使能 SPI 接收握手接口。

## 从器件接口

主机处理器通过从器件接口访问 SPI 控制器的数据、控制和状态信息。SPI 支持 32 bit 的数据总线宽度。访问 SPI 外设以下小节中介绍。

### 控制和状态寄存器访问

SPI 控制器内的控制和状态寄存器是按字节寻址的。SPI 控制器的控制或状态寄存器的最高宽度是 16 bit。因此，SPI 控制和状态寄存器的所有读和写操作仅需要一个访问。

### 数据寄存器访问

SPI 控制器内的数据寄存器 (DR) 是 16 bit 宽以便和最高串行传输容量（数据帧）保持一致。对 DR 的写操作将数据从从器件写数据总线移到发送 FIFO 缓冲器。对 DR 的读操作将数据从接收 FIFO 缓冲器移到从器件读回数据总线。



**SPI 控制器中的 DR 寄存器占用存储器映射的 64 个 32-bit 位置来促进突发传输。**系统总线本身不存在突发传输，但是 SPI 支持系统互联的突发。写入这些地址位置的任何一个和将数据从从器件写数据总线送入发送 FIFO 缓冲器具有相同的效果。从这些位置的任何一个读取和将数据从接收 FIFO 缓冲器移入从器件读回数据总线具有相同的效果。SPI 控制器上的 FIFO 缓冲器是不可寻址的。



## 时钟和复位

SPI 控制器使用表 19 - 1 所示的时钟和复位信号。

表 19 - 1. SPI 控制器时钟和复位

	主器件	从器件
SPI clock	spi_m_clk	l4_main_clk
SPI bit-rate clock	sclk_out	sclk_in
Reset	spim_rst_n	spis_rst_n

## SPI 编程模型

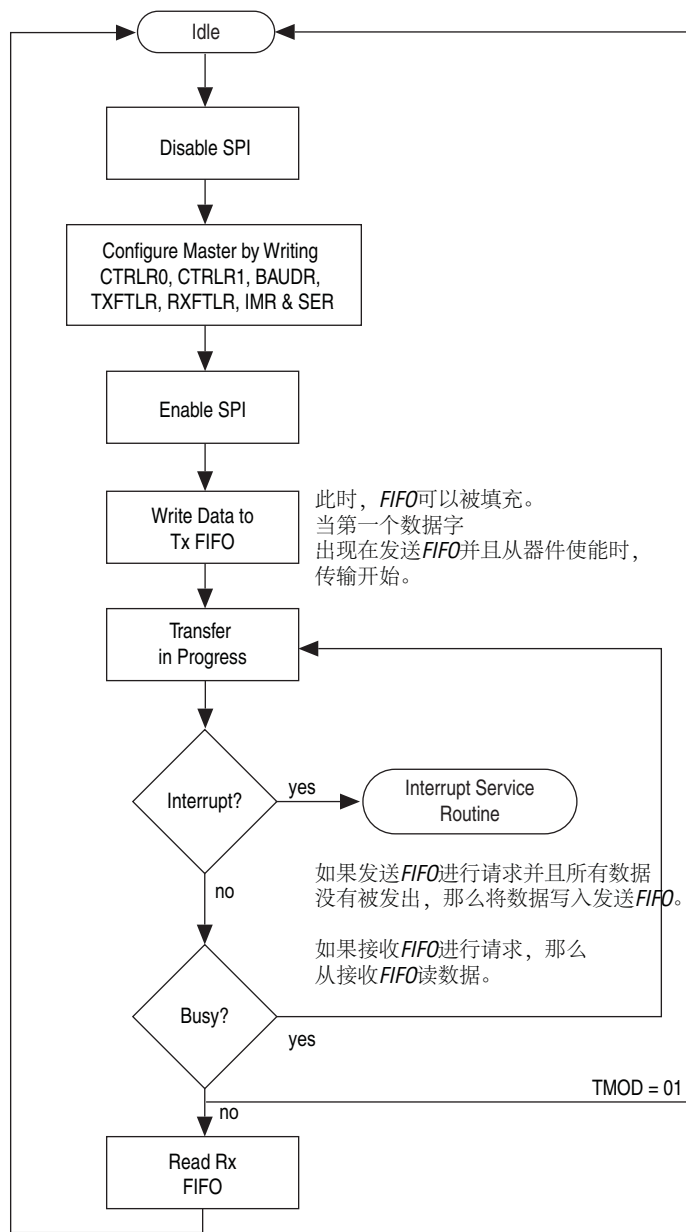
该部分介绍基于以下主器件和从器件传输的 SPI 控制器的编程模型：

- 主器件 SPI 和 SSP 串行传输
- 主器件 Microwire 串行传输
- 从器件 SPI 和 SSP 串行传输
- 从器件 Microwire 串行传输
- 从器件选择的软件控制

## 主器件 SPI 和 SSP 串行传输

图 19 - 11 显示了主器件 SPI 或 SSP 串行传输的软件流程。

图 19 - 11. 主器件 SPI 或 SSP 串行传输软件流程



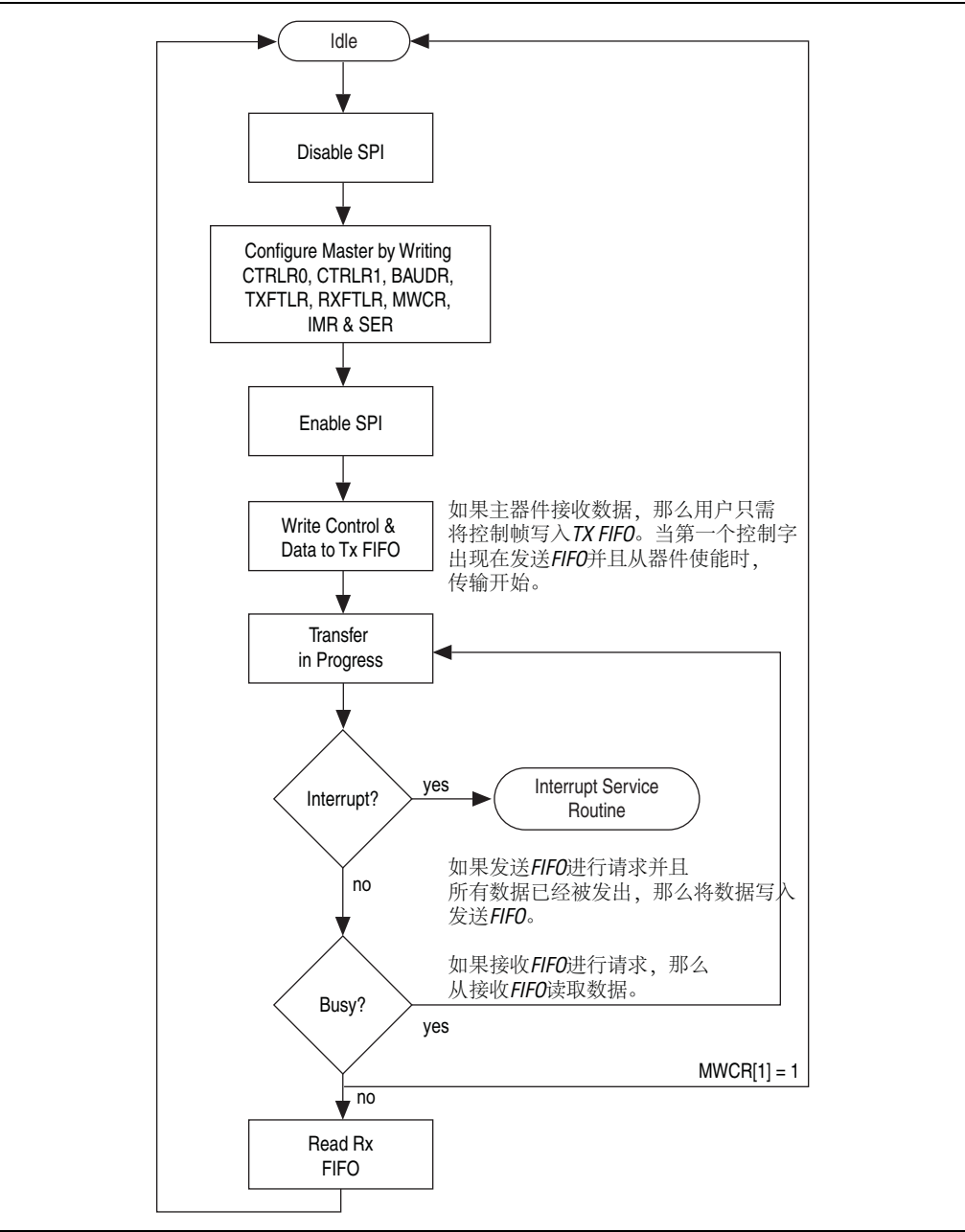
要完成 SPI 主器件的 SPI 或 SSP 串行传输，请遵循以下步骤：

1. 如果 SPI 主器件被使能，那么通过将 0 写入 SSI 使能寄存器 (SSIENR) 将其禁用。
2. 对传输设置 SPI 主器件控制寄存器；您可以用任何顺序设置这些寄存器。
  - 写入控制寄存器 0 (CTRLR0)。对于 SPI 传输，必须设置串行时钟极性和串行时钟相位参数等同于目标从器件。
  - 如果传输模式是只接收，那么使用传输中的帧数减 1 写入控制寄存器 1 (CTRLR1)。例如，如果想要接收 4 个数据帧，那么将 3 写入该寄存器。
  - 对于传输，写入 Baud 速率选择寄存器 (BAUDR) 以设置 baud 速率。
  - 写入发送和接收 FIFO 阈值水平寄存器 (TXFTLR 和 RXFTLR) 以设置 FIFO 缓冲器阈值水平。
  - 写入 IMR 寄存器以设置中断屏蔽。
  - 写入从器件使能寄存器 (SER) 寄存器以便选择性地使能目标从器件。如果从器件使能，那么一个有效数据输入一出现在发送 FIFO 缓冲器，传输就会开始。如果写入数据寄存器 (DR) 之前没有使能从器件，那么传输直到从器件使能才开始。
3. 通过将 1 写入 SSIENR 寄存器使能 SPI 主器件。
4. 将传输到目标从器件的数据写入发送 FIFO 缓冲器（写 DR）。如果此时没有从器件在 SER 寄存器使能，那么将其使能以开始传输。
5. 轮询 BUSY 状态以等待传输完成。如果进行了发送 FIFO 空中断请求，那么写入发送 FIFO 缓冲器（写 DR）。如果进行了接收 FIFO 满中断请求，那么读取接收 FIFO 缓冲器（读 DR）。
6. 当发送 FIFO 缓冲器为空时，移位控制逻辑停止传输。如果传输模式是只接收 (TMOD = 2'b10)，那么当接收到指定的帧数时，移位控制逻辑停止传输。当传输完成时，BUSY 状态被复位至 0。
7. 如果传输模式不是只发送 (TMOD != 01)，那么读取接收 FIFO 缓冲器直到它为空。
8. 通过将 0 写入 SSIENR 禁用 SPI 主器件。

## 主器件 Microwire 串行传输

图 19 - 12 显示了 Microwire 串行传输的软件流程。

图 19 - 12. Microwire 串行传输软件流程



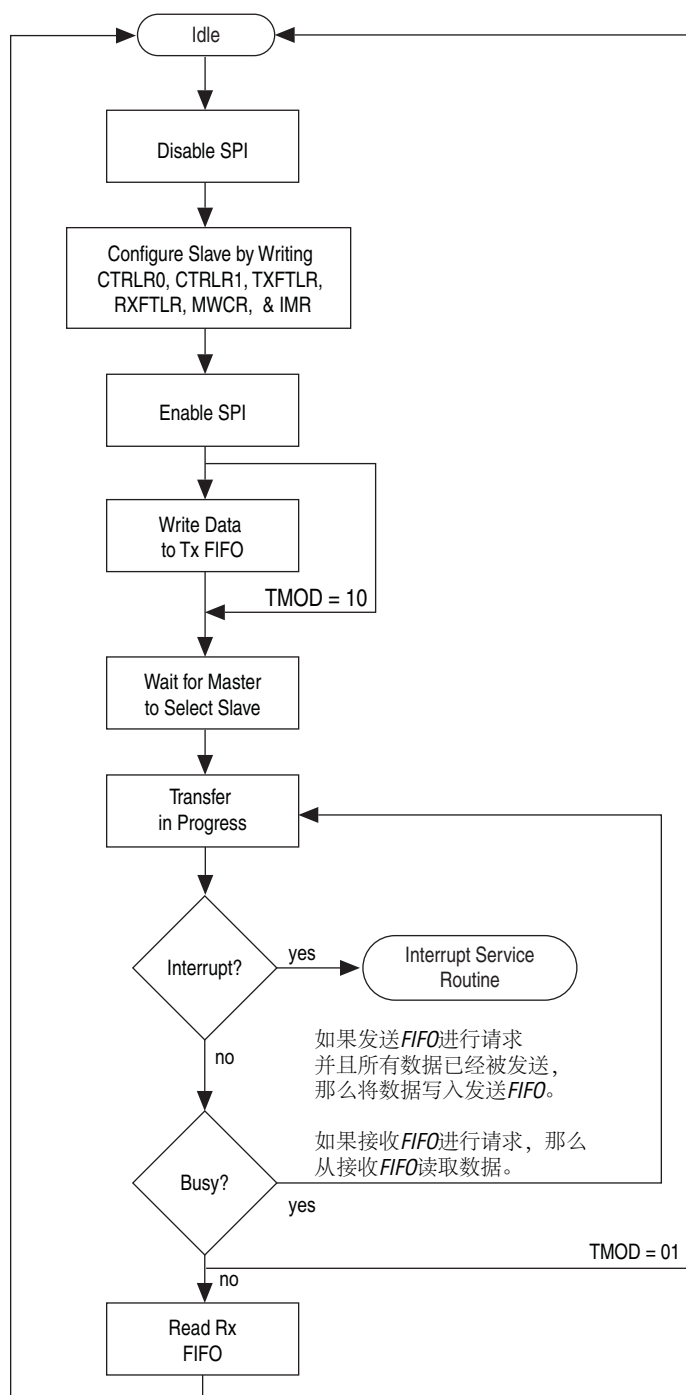
要完成 SPI 主器件的 Microwire 串行传输，请遵循以下步骤：

1. 如果 SPI 主器件使能，那么通过将 0 写入 SSIENR 而禁用它。
  2. 对传输设置 SPI 控制寄存器。您可以用任何顺序设置这些寄存器。
    - 写入 CTRLR0 以设置传输参数。如果传输是有序的并且 SPI 主器件接收到数据，那么使用传输中的帧数减 1 写入 CTRLR1。例如，如果想要接收 4 个数据帧，那么将 3 写入该寄存器。
    - 写入 BAUDR 以设置该传输的 baud 速率。
    - 写入 TXFTLR 和 RXFTLR 以设置 FIFO 缓冲器阈值水平。
    - 写入 IMR 寄存器以设置中断屏蔽。
- 您可以写入 SER 寄存器以选择性地使能目标从器件。如果从器件使能，那么一个有效数据输入一出现在发送 FIFO 缓冲器，那么传输就开始。如果写入 DR 寄存器之前没有使能从器件，那么传输直到从器件使能才开始。
3. 通过将 1 写入 SSIENR 寄存器使能 SPI 主器件。
  4. 如果 SPI 主器件发送数据，那么将控制和数据字写入发送 FIFO 缓冲器（写 DR）。如果 SPI 主器件接收到数据，那么将控制字或字写入发送 FIFO 缓冲器。如果此时没有从器件在 SER 寄存器中使能，那么使能它以开始传输。
  5. 轮询 BUSY 状态以等待传输完成。如果进行了发送 FIFO 空中断请求，那么写入发送 FIFO 缓冲器（写 DR）。如果进行了接收 FIFO 满中断请求，那么读取接收 FIFO 缓冲器（读 DR）。
  6. 当发送 FIFO 缓冲器为空时，移位控制逻辑停止传输。如果传输模式是有序的并且 SPI 主器件接收到数据，那么当接收到指定的数据帧数时，移位控制逻辑停止传输。当传输完成时，BUSY 状态被复位至 0。
  7. 如果 SPI 主器件接收到数据，那么读取接收 FIFO 缓冲器直到它为空。
  8. 通过将 0 写入 SSIENR 禁用 SPI 主器件。

## 从器件 SPI 和 SSP 串行传输

图 19 - 13 显示了从器件 SPI 或 SSP 串行传输的软件流程。

图 19 - 13. 从器件 SPI 或 SSP 串行传输软件流程



要完成从串行主器件到 SPI 从器件的连续的串行传输，请遵循以下步骤：

1. 如果使能 SPI 从器件，那么通过将 0 写入 SSIENR 禁用它。
2. 对于传输设置 SPI 控制寄存器。您可以用任何顺序设置这些寄存器。
  - 写入 CTRLR0（对于 SPI 传输，设置 SCPH 和 SCPOL 等同于主器件。
  - 写入 TXFTLR 和 RXFTLR 以设置 FIFO 缓冲器阈值水平。
  - 写入 IMR 寄存器以设置中断屏蔽。
3. 通过将 1 写入 SSIENR 寄存器，使能 SPI 从器件。
4. 如果传输模式是发送和接收 (TMOD=2'b00) 或是只发送 (TMOD=2'b01)，那么将传输到主器件的数据写入发送 FIFO 缓冲器（写 DR）。如果传输模式是只接收 (TMOD=2'b10)，那么不需要将数据写入发送 FIFO 缓冲器。发送移位寄存器中的当前值被重新发送。
5. SPI 从器件现在可以开始串行传输。当串行-主器件选择 SPI 从器件时传输开始。
6. 当传输进行时，BUSY 状态可以被轮询以返回传输状态。如果进行了发送 FIFO 空中断请求，那么写入发送 FIFO 缓冲器（写 DR）。如果进行了接收 FIFO 满中断请求，那么读取接收 FIFO 缓冲器（读 DR）。
7. 当串行主器件移除 SPI 从器件的选择输入时，传输结束。当完成传输时，BUSY 状态被复位至 0。
8. 如果传输模式不是只发送 (TMOD!= 01)，那么读取接收 FIFO 缓冲器直到它为空。
9. 通过将 0 写入 SSIENR，禁用 SPI 从器件。

## 从器件 Microwire 串行传输

对于 SPI 从器件，Microwire 协议的操作方式和 SPI 协议基本相同。**不存在 SPI 从器件解码的控制帧。**

## 从器件选择的软件控制

当使用软件选择从器件时，串行从器件的输入选择线被连接到 SPI 主器件的单从器件选择输出。以下实例显示了从器件选择的软件流程：

- 对于 SPI 主器件：
  1. 如果 SPI 主器件被使能，那么通过将 0 写入 SSIENR 将其禁用。
  2. 写入 CTRLR0 以匹配所需传输。
  3. 如果传输是只接收，那么将帧数写入 CTRLR1。
  4. 写入 BAUDR 以设置传输 baud 速率。
  5. 写入 TXFTLR 和 RXFTLR 以设置 FIFO 缓冲器阈值水平。
  6. 写入 IMR 寄存器以设置中断屏蔽。
  7. 将 SER 寄存器 bit[0] 写入逻辑 '1' 以选择该实例的从器件 1。
  8. 将 SSIENR 寄存器 bit[0] 写入逻辑 '1' 以使能 SPI 主器件。
- 对于 SPI 从器件：
  9. 如果 SPI 从器件被使能，那么通过将 0 写入 SSIENR 将其禁用。



10. 写入 CTRLR0 以匹配所需传输。
  11. 写入 TXFTLR 和 RXFTLR 以设置 FIFO 缓冲器阈值水平。
  12. 写入 IMR 寄存器以设置中断屏蔽。
  13. 将 SSIENR 寄存器 bit[0] 写入逻辑 '1' 以使能 SPI 从器件。
  14. 如果 SPI 从器件发送数据，那么将数据写入 TX FIFO 缓冲器。请注意所有其它 SPI 从器件都被禁用 (SSIENR = 0)，因此不会在其 ss\_in\_n 端口上响应为一个有效电平。
- SPI 控制器中 RX 和 TX 缓冲器的 FIFO 缓冲器深度 (FIFO\_DEPTH) 是 256 个入口。

## DMA 控制器操作

要启用 SPI 控制器的 DMA 控制器接口，必须写入 DMA 控制寄存器 (DMACR)。将 1 写入 DMACR 寄存器的 TDMAE 位域使能 SPI 控制器发送握手接口。将 1 写入 DMACR 寄存器的 RDMAE 位域使能 SPI 控制器接收握手。



要了解关于 DMA 控制器的详细信息，请参考 *Cyclone V 器件手册* 第 3 卷的 *DMA Controller* 章节。

### DMA 操作

要了解关于 DMA 操作的更多信息，请参考 ARM DMA 章节。

### 发送 FIFO 缓冲器下溢

SPI 串行传输期间，无论发送 FIFO 缓冲器中的入口数少于还是等于 DMA 发送数据水平寄存器 (DMATDLR) 中的值 (称为水印层)，都会对 DMA 控制器进行发送 FIFO 缓冲器请求。DMA 控制器通过将一个突发的数据写入发送 FIFO 缓冲器做出响应，突发长度被指定为 DMA 突发长度。



要了解关于 DMA 突发长度微代码设置的详细信息，请参考 *Cyclone V 器件手册* 第 3 卷的 *DMA Controller* 章节。

应该从 DMA 经常获取数据，以便发送 FIFO 缓冲器可以连续地执行串行传输，也就是，当 FIFO 缓冲器开始为空时，另一个 DMA 请求应该被触发。否则，FIFO 缓冲器会运行完数据 (下溢)。要防止该情况发生，必须正确地设置水印层。

### 发送水印层

请考虑以下假设的实例：

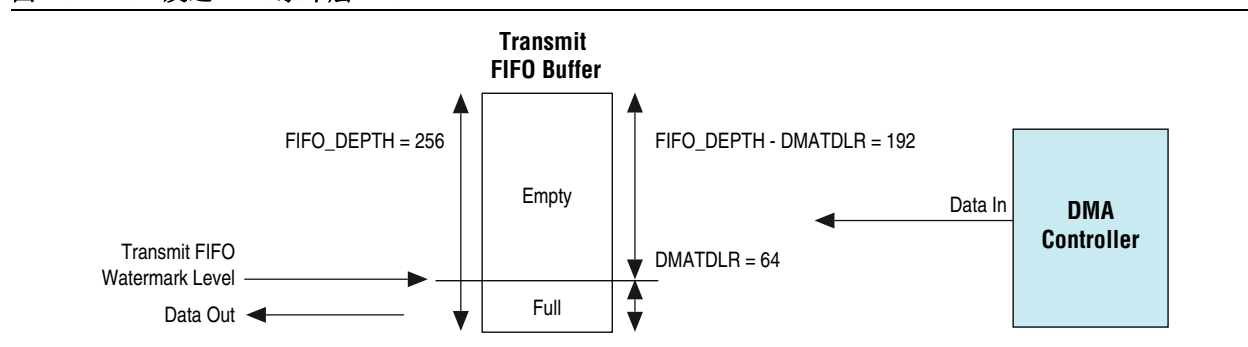
DMA 突发长度 = FIFO\_DEPTH - DMATDLR

此实例中，DMA 突发中传输的数据项数等于发送 FIFO 缓冲器中的空区域。请考虑下列两个不同的水印层设置：

- 用例 1: DMATDLR = 64。
  - 发送 FIFO 水印层 = DMATDLR = 64
  - DMA 突发长度 = FIFO\_DEPTH - DMATDLR = 192
  - SPI 发送 FIFO\_DEPTH = 256
  - 模块传输容量 = 960

图 19-14 显示了水印层等于 64 时的发送 FIFO 缓冲器。

图 19-14. 发送 FIFO 水印层 = 64



所需的突发传输数等于模块容量除以每突发的数据项数：

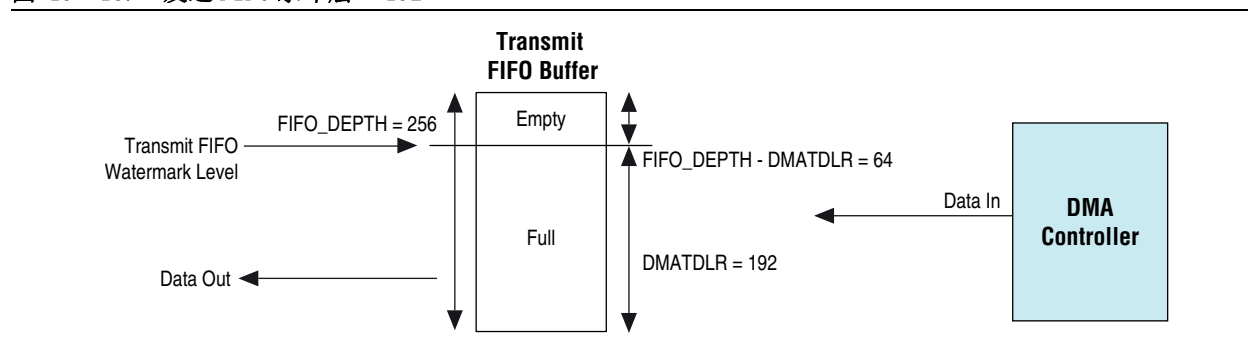
模块传输容量 / DMA 突发长度 =  $960/192 = 5$

DMA 模块传输中的突发传输数是 5。但是水印层 DMATDLR 很低。因此，SPI 串行发送线需要发送数据的发送下溢的可能性很高，但是在发送 FIFO 缓冲器中不会留下数据。这种情况发生是因为 DMA 在 FIFO 缓冲器变空之前没有时间执行 DMA 请求。

- 用例 2: DMATDLR = 192
  - 发送 FIFO 水印层 = DMATDLR = 192
  - DMA 突发长度 = FIFO\_DEPTH - DMATDLR = 64
  - SPI 发送 FIFO\_DEPTH = 256
  - 模块传输容量 = 960

图 19-15 显示了水印层等于 192 时的发送 FIFO 缓冲器。

图 19-15. 发送 FIFO 水印层 = 192



模块的突发传输数：

模块传输容量 / DMA 突发长度 =  $960/64 = 15$

对于该模块传输中，在 DMA 模块传输中具有 15 个目的突发传输。但是水印层 DMATDLR 很高。因此，SPI 发送下溢的可能性很低，因为在 SPI 发送 FIFO 缓冲器变空之前，DMA 控制器有足够的时间执行目的突发传输请求。

因此，通过每模块的较多突发传输，第 2 个用例具有较低的下溢可能性。跟第 1 个用例相比，这提供了一个潜在地每模块的较高量突发和较差总线利用率。

因此，选择水印层的目标是最小化每模块的传输数，同时保持下溢情况的可能性到可接受的水平。*在实践中，这是一个速率比函数，在这个速率比上，SPI 将数据发送为 DMA 可以响应的目的突发请求速率。*

## 发送 FIFO 缓冲器上溢

当发送 FIFO 缓冲器中没有足够的空间执行目的突发请求时，设置 DMA 传输突发长度为一个大于水印层（触发 DMA 请求）的值会导致上溢。因此，必须遵循以下的公式来避免上溢：

$$\text{DMA 突发长度} \leq \text{FIFO\_DEPTH} - \text{DMATDLR}$$

用例 2 中：DMATDLR = 192，突发请求进行时的发送 FIFO 缓冲器中的空间量等于 DMA 突发长度。因此，突发传输完成时，发送 FIFO 缓冲器可能为满，但是不上溢。

因此，要实现最佳操作，DMA 突发长度应设置在触发一个发送 DMA 请求的 FIFO 缓冲器水平；也就是：

$$\text{DMA 突发长度} = \text{FIFO\_DEPTH} - \text{DMATDLR}$$

遵循这一公式会减少模块传输所需的 DMA 突发数，反而这会提高总线利用率。



传输期间，如果 SPI 控制器在串行发送线上已经成功地发送了一个或多个数据项，那么发送 FIFO 缓冲器在 DMA 突发传输结束时不会满。

## 接收 FIFO 缓冲器上溢

*SPI 串行传输期间，无论接收 FIFO 缓冲器中的入口数是在 DMA 接收数据水平寄存器还是在其之上 (DMATDLR + 1)，都会对 DMA 进行接收 FIFO 缓冲器请求。*这被称为水印层。DMA 通过从接收 FIFO 缓冲器获取一个突发的数据做出响应。

数据应该由 DMA 经常获取，以便接收 FIFO 缓冲器连续地接受串行传输，也就是，当 FIFO 开始填充时，另一个 DMA 传输被请求。否则，FIFO 将会被填充数据（上溢）。要防止这一情况，用户必须正确地设置水印层。

## 选择接收水印层

与选择发送水印层相似，接收水印层，DMATDLR + 1，应被设置以最小化上溢的可能性，如图 8 所示。*它是每模块所需的 DMA 突发传输数和上溢发生的可能性之间的权衡。*

## 接收 FIFO 缓冲器下溢

将源传输突发长度设置为大于水印层会导致下溢，其中没有足够数据执行源突发请求。因此，必须遵循以下公式来避免下溢：

$$\text{DMA 突发长度} = \text{DMATDLR} + 1$$

进行了突发请求时，如果接收 FIFO 缓冲器中的数据项数等于源突发长度，那么突发传输完成时，接收 FIFO 可能为空，但是不下溢。为了实现最佳操作，DMA 突发长度应该设置在水印层，DMATDLR + 1。

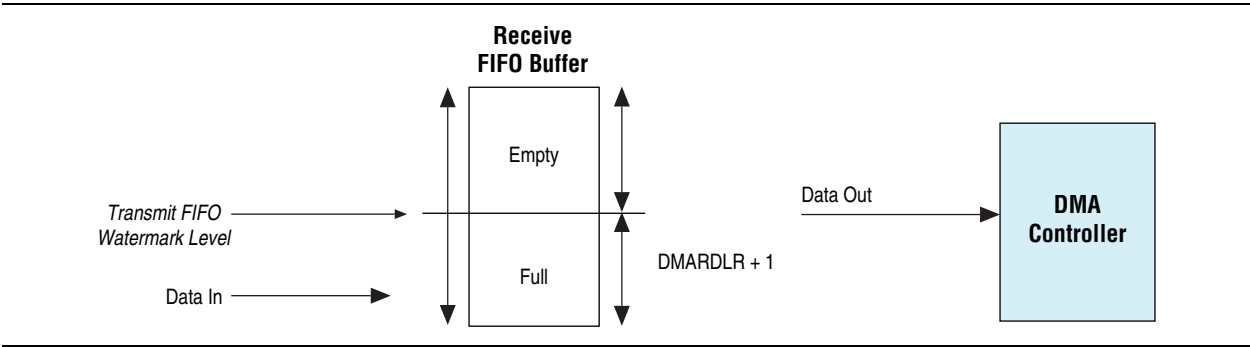
遵循该公式会减少模块传输中 DMA 突发数，从而可以提高总线利用率。




突发期间，如果 SPI 控制器在串行接收线上已经成功地接收了一个或多个数据项，那么源突发传输结束时接收 FIFO 缓冲器不会为空。

图 19 - 16 显示了接收 FIFO 缓冲器。

图 19 - 16. 接收 FIFO 缓冲器




## SPI 控制器地址映射和寄存器定义

 地址映射和寄存器定义位于该手册卷附带的 [hps.html](#) 文件中。点击链接以打开文件。

要查看模块说明和基地址，找到并且点击以下任何模块实例的链接：

- `spis0`
- `spis1`
- `spim0`
- `spim1`

然后要查看寄存器和域说明，找到并且点击寄存器名称。寄存器地址是相对于每个模块实例的基地址的偏移。

 所有模块的基地址也在 *Cyclone V 器件手册* 第 3 卷的 *Introduction to the Hard Processor System* 章节中列出。

## 文档修订历史

表 19 - 2 显示了该文档的修订历史。

表 19 - 2. 文档修订历史

日期	版本	修订内容
2012 年 11 月	1.2	少量文本编辑。
2012 年 5 月	1.1	添加了编程模型、地址映射、寄存器定义、时钟和复位部分。
2012 年 1 月	1.0	首次发布。