Application Note

快速串行接口 (FSI) 在多芯片互连中的应用

Aki Li, Kevin Allen, and Aditya Dholakia, and Naveen Kothuri

摘要

在工业应用中，经常需要多个器件以快速、低延迟且同步的方式相互通信。比如，在去中心化/分布式控制系统架构中就存在此需求。快速串行接口 (FSI) 是为 C2000™ 实时控制微控制器 (MCU) 创建的新型通信外设，可将其可靠的高速通信功能扩展到系统中的多个器件。本应用报告演示了如何使用 FSI 实现菊花链或星型网络拓扑，提供了基于不同配置和模式下 FSI 高速通信的测试性能。您可以从 C2000WARE 下载这些源代码，在不同应用中进行 FSI 的快速设计与验证。

相应软件的目标处理器包括 TMS320F28002x、TMS320F28004x 和 TMS320F2838x。相应的实现方法和软件应用也可以用于并移植到任何包含 FSI 模块的 C2000 处理器上。本文档中讨论的示例代码，都可以在安装最新的 C2000WARE 版本后在以下本地目录地址找到：

C:\ti\c2000\C2000Ware_<version_number>\driverlib\f28xxxx\examples\fsi

可用的示例工程包括：

- fsi_ex_daisy_handshake_lead
- fsi_ex_daisy_handshake_node
- fsi_ex_star_broadcast

内容

插图清单

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

表格清单

商标

C2000™ and Code Composer Studio™ are trademarks of Texas Instruments.
所有商标均为其各自所有者的财产。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

1 FSI 模块简介

FSI 模块是一个串行通信外设，能够进行可靠而稳健的高速通信，速度高达 200Mbps。FSI 利用极少的单向信号，并在使用数字隔离器件时，提供了一种跨隔离栅的低成本通信方式。因此，FSI 在对实时处理能力和高速通信要求较高的工业应用场景中，采用了新的方法来分配 C2000 MCU 强大的传感、处理和驱动功能。

通常，可以在两种系统条件下实现 FSI：

- 在共享同一电压参考系 - 共地和接地层的情况下实现 MCU 间有线通信。
- 在不同电压参考系和接地层需要使用数字隔离器（例如 ISO77xx）实现跨越隔离屏障的有线通信，通常用于放置在热侧的 MCU，而这些 MCU 需要与冷侧的 MCU 进行通信。

许多实时系统可以从 FSI 外设中受益。可以用控制每个轴的 C2000 器件节点构建一个多轴伺服驱动器。将 FSI 作为通信链路，可以在器件之间快速传输和接收控制环路信息，以保持精确的运动控制。有关此系统的示例，请参阅*基于快速串行接口 (FSI) 的分布式多轴伺服驱动器参考设计*。

此外，随着全球电力消耗的增加、对更高效电源的需求，以及宽带隙 GaN 和 SiC 产品的推出，人们趋向使用更加复杂的配电架构。使用 C2000 MCU 的分布式电源控制解决方案可与 FSI 连接并灵活地满足这些要求。有关此类电源相关系统的讨论，请参阅*采用多个 MCU、基于 FSI 的分布式电源控制架构*。

FSI 外设提供许多功能，包括可编程数据长度、由硬件管理的 CRC、ECC 支持等。PING 看门狗和帧看门狗可以实现自动线路中断检测。FSI 接收模块中实现的独特延迟线控制功能可以针对布线长度不匹配、收发器或数字隔离 IC 引起的通道间的信号偏斜进行调整，从而使 FSI 保持高速且稳健的通信。

FSI 包含分别进行配置和运行的独立发送器 (FSITX) 和接收器 (FSIRX) 内核。因此，与某些其他同步通信协议不同，FSI 协议没有主器件和从器件的概念，并且允许在两个方向上同时进行全速通信。图 1-1 展示了每个 FSI 模块的 CPU 接口。每个模块最多拥有三条信号线：一个时钟信号和两个数据信号，其中第二条数据线 FSITXyD1 和 FSIRXyD1 是可选的，可以启用以进行多通道传输，从而使数据位的传输速度提高一倍。因此，需要至少四条信号线来创建双向点对点通信。考虑到 FSITX 的时序规格（请参阅节 8 中引用的器件特定数据表），使用两条数据线时，由于数据在时钟信号的两个边沿上进行传输，因此能够以 50MHz 最大时钟频率实现 200Mbps 的最大数据速率。有关 FSI 的完整概述（包括所有可用的特性和功能），请参阅特定于器件的技术参考手册 (TRM)。

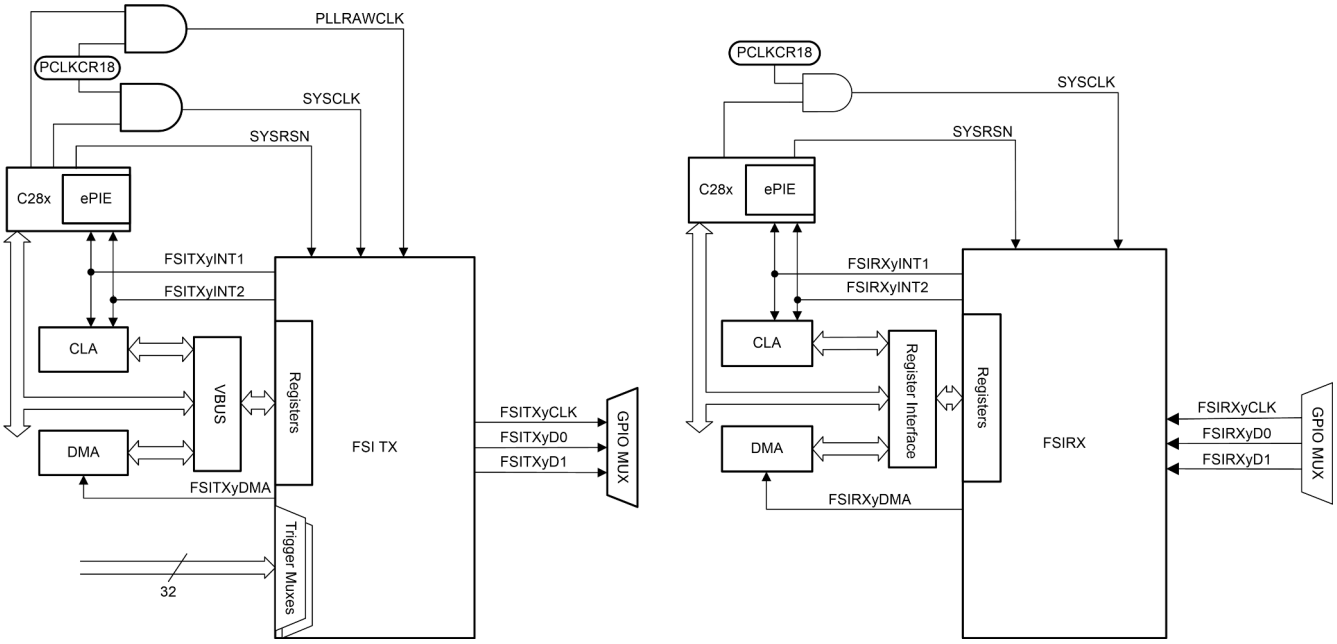

图 1-1. FSITX 和 FSIRX CPU 接口

2 FSI 应用

就电力电子应用的趋势而言，对更高功率水平的需求在不断增长，从而使多个并联的电源模块更受欢迎。此类应用的示例包括工业驱动器、通信电源整流器、服务器电源、车载充电器等。同时，为了实现具有高性能的复杂系

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

统，通常使用多个 MCU，并且这些 MCU 必须以同步方式运行。因此，关键数据（包括保护信号和采样参数，甚至是控制环路数据）需要在多个器件/模块之间以最快的速度和最小的延迟进行传输。与传统的控制器局域网 (CAN)、串行外围接口 (SPI) 或通用异步接收器/发送器 (UART) 相比，FSI 将更适合处理此问题。

有许多用于连接多个器件的通信网络拓扑，每种拓扑都有自己的优势。可以通过以菊花链方式连接多个设备的 FSI 来创建环形拓扑。环形拓扑的优势在于每个器件仅需要一个 FSI 发送器和接收器，并且从物理连接角度而言也很简单。图 2-1 显示了用于 N (N≥2) 个节点器件的菊花链连接系统，其中每个器件（索引为 i）与器件 i-1 的 FSITX 和器件 i+1 的 FSIRX 相连接。

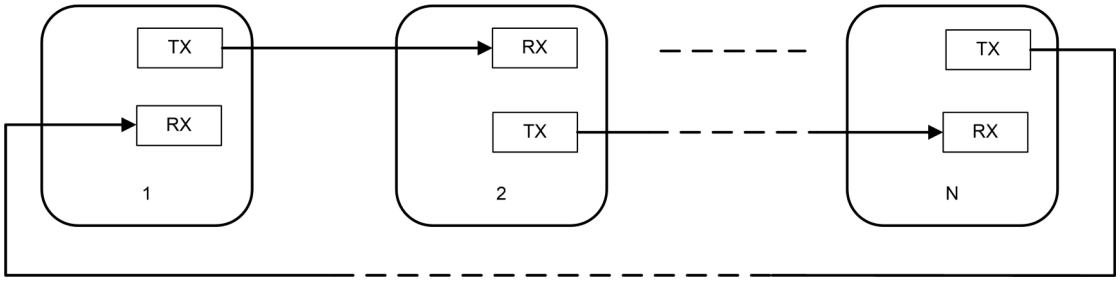

图 2-1. 菊花链连接示例

上述菊花链拓扑的一个缺点是，如果链中的一个器件发生故障，则整个通信链路都会中断。另一个缺点是，如果接收到的数据将用于后续器件，则器件必须将数据转发到链中的下一个器件。这可能会增加传输数据包以及链中的各个器件接收数据时的总延迟。

一种可以解决链路断开问题并减小器件间延迟的通信拓扑是星型拓扑，其中多个节点直接连接到一个中央主机器件。图 2-2 显示了具有 N (N≥2) 个节点器件的星型拓扑系统。

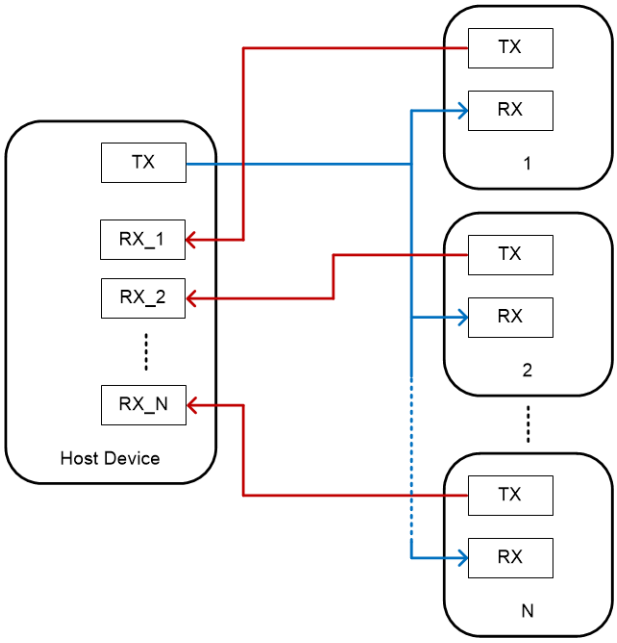

图 2-2. 星型拓扑示例

主机器件的 FSI 发送器连接到每个节点器件的 FSI 接收器，以便主机同时向所有节点广播数据包。另一方面，节点器件的发送器连接到主机器件的独立接收器，使其能够随时将数据直接发送回主机。主机需要 N 个独立的 FSI 接收器模块，因此星型连接方式会带来成本的增加。F2838x 系列 C2000 器件具有两个 FSI 发送器和八个 FSI 接收器，可安装到主机插槽中。

3 握手机制

可以实施握手机制，以配置器件并验证 FSI 通信拓扑中的链路。以下各节讨论了不同的握手序列。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

3.1 菊花链握手机制

配置每个器件的 FSITX 和 FSIRX 模块之后，应实现握手机制以在实际数据传输之前准备链中的每个器件，因为在实际情况下器件可能按任意顺序加电。

为了简化数据流，将一个器件指定为主控器件，充当握手序列的驱动器，同时将菊花链环路中的其他 N-1 个器件指定为节点。在图 2-1 中的示例中，器件 1 将是主控器件。应注意，其他 N-1 个节点器件将共享相同的握手配置。

握手过程可以描述为：

1. 对于所有器件，将 FSITX 的帧类型配置为 Ping 帧，并为 FSI INT1 向量上的“接收到 Ping 帧”事件启用接收器中断，以检测传入的信号。
2. 开始 Ping 循环 0：
   a. 主控器件将刷新序列发送至第二个器件，后跟一个带有 Tag0 (0000) 的 Ping 帧；等待一段时间。如果主控器件接收到有效的 Ping 帧标签 Tag0，则继续执行第二个循环；否则再次迭代 Ping 循环 0。
   b. 节点器件进入等待循环以等待接收器中断。如果从前一个器件接收到有效的 Ping 帧标签 Tag0，则继续执行循环 1；否则再次迭代 Ping 循环 0。

Copyright © 2023 Texas Instruments Incorporated

3. 开始 Ping 循环 1：
   a. 主控器件发送一个带有 Tag1 (0001) 的 Ping 帧；等待一段时间。如果主控器件接收到有效的 Ping 帧 Tag1，则握手序列完成，应用可以继续执行；否则再次迭代 Ping 循环 1。
   b. 节点器件发送刷新序列，后跟 Ping 帧 Tag0，然后等待接收器中断。如果接收到有效的 Ping 帧 Tag1，则发送 Ping 帧 Tag1，以表明握手序列已完成；否则再次迭代 Ping 循环 1。
4. 握手完成。

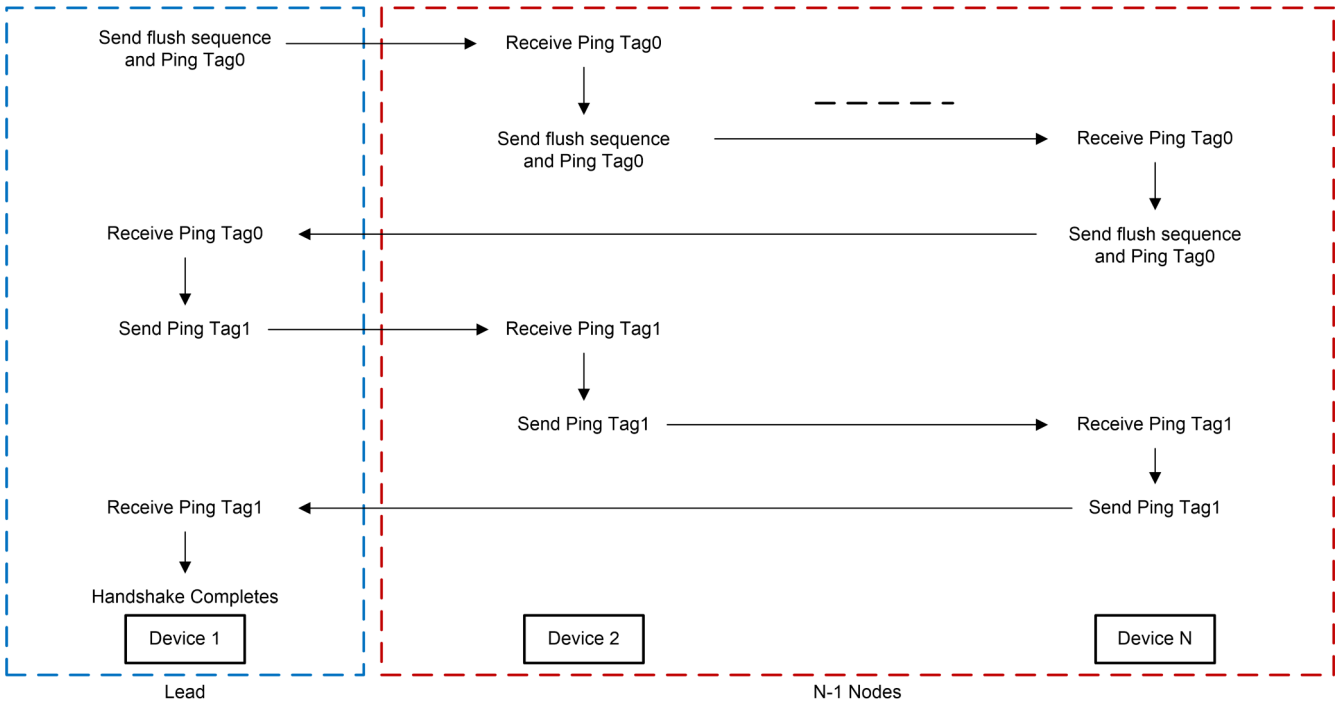

图 3-1. 菊花链握手序列

图 3-1 中显示了简化的数据流。菊花链连接握手机制涉及两个 Ping 循环。Ping 循环 0 的作用是沿着器件链建立通信路径，而 Ping 循环 1 用于向节点确认通信路径良好。在 Ping 循环 0 中，节点器件等待从链中的前一个器件接收 Ping Tag0。成功接收 Ping Tag0 之后，会将其转发至链中的下一个器件。如果链中的器件未加电或没有为接收做好准备，则 Ping 循环 0 将失败。Ping 循环 0 成功（其中 Ping tag0 返回到主控器件中）之后，启动 Ping 循环 1 来通知节点器件握手序列已完成，并开始期待实际数据。

可以在测试的项目中找到握手函数，菊花链环路中的主控器件使用 handshake_lead()，其他 N-1 个器件使用 handshake_node()。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

3.2 星型握手机制

与节 3.1 中描述的握手序列非常相似的握手序列可应用于星型拓扑用例。在该实现中，主机器件或主控器件将广播 Ping 发送至所有节点器件，每个节点器件都沿其独立的 TX 路径进行响应。主控器件会等待从所有节点接收 Ping 帧，然后再继续执行下一步操作。

图 3-2 中显示了该序列。

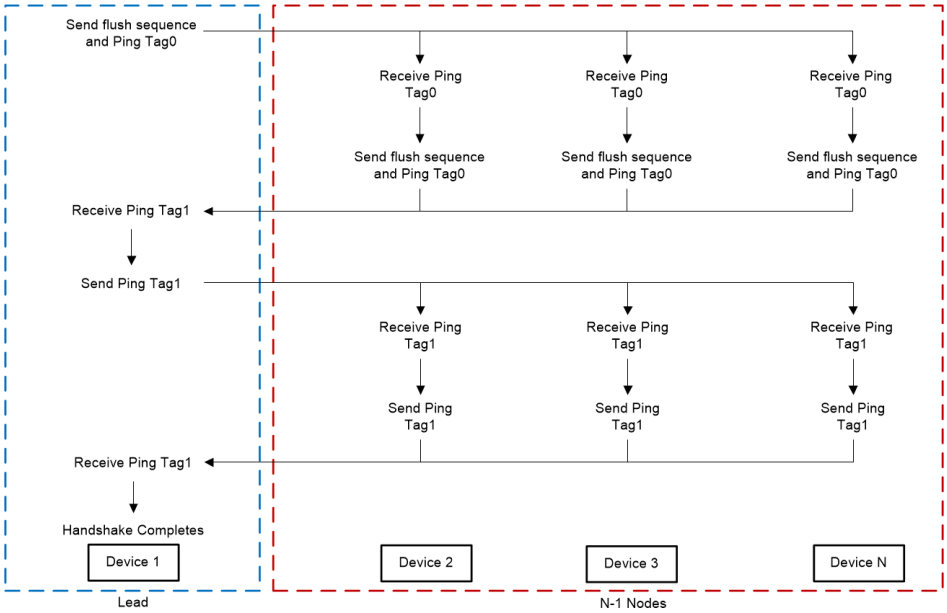

图 3-2. 星型握手序列

4 发送和接收 FSI 数据帧

4.1 FSI 数据帧配置 API

要正确发送和接收数据帧，需要进行多项配置，包括帧类型、帧标签、用户数据、字长、数据线条数以及对数据缓冲区的读写。配置示例代码针对 FSITX 和 FSIRX 使用 driverLib API 函数，C2000WARE 的 fsi.h driverLib 头文件中定义了这些函数，如下所示。请注意，帧标签和用户数据的内容完全可由用户配置，可用于区分从哪个器件发送了接收到的数据或将数据发送至哪个器件。

```
// TX 设置部分
   FSI_setTxFrameType(FSITXA_BASE, FSI_FRAME_TYPE_NWORD_DATA);
    FSI_setTxSoftwareFrameSize(FSITXA_BASE, nWords);
    FSI_setTxDataWidth(FSITXA_BASE, nLanes);
    FSI_setTxUserDefinedData(FSITXA_BASE, txUserData);
    FSI_setTxFrameTag(FSITXA_BASE, txDataFrameTag);
// RX 设置部分
    FSI_setRxSoftwareFrameSize(FSIRXA_BASE, nWords);
    FSI_setRxDataWidth(FSIRXA_BASE, nLanes);
```

4.2 开始传输数据帧

有四种触发数据传输的方法，包括软件触发、外部触发 (EPWMx-SOCA/B)、使用 DMA 和硬件直通特性。对于软件触发方法，将 1 写入到 TX_FRAME_CTRL.START 寄存器位中或使用 driverLib 函数“FSI_startTxTransmit()”将开始传输。如果使用外部触发（例如 EPWMx-SOCA），则一旦发生外部触发信号，就会发送数据。

由于可以在每次通过 FSITX 或 FSIRX 模块完成数据帧发送或接收时生成 DMA 触发，因此它提供了一种方便的方法来传输和存储数据，尤其是海量数据。此处提供了使用 DMA 进行 FSI 通信的配置示例。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

必须将 TX_OPER_CTRL_LO.START_MODE 设置为 0x2，这意味着向帧标签/用户数据字段写入数据可以触发传输，然后在 FSITX 上启用 DMA 事件：

```
FSI_setTxStartMode(FSITXA_BASE, FSI_TX_START_FRAME_CTRL_OR_UDATA_TAG);
FSI_enableTxDMAEvent(FSITXA_BASE);
```

需要使用两个连续的 DMA 通道来分别填充发送缓冲区和帧标签/用户数据字段。 依次使用两个通道，可以在设置帧标记和用户数据后，立即开始按照 TX_OPER_CTRL_LO.START_MODE 寄存器位中的配置进行传输。在示例代码中，使用了 DMA CH1 和 DMA CH2。此外，还务必注意，由于 FSI 发送缓冲区是一个 16 字的循环缓冲区，因此必须对超过 16 个字的数据启用饱和控制，否则超过 16 个字的数据会出现长度溢出。

```
DMA_configWrap(DMA_CH1_BASE, DMA_TRANSFER_SIZE_IN_BURSTS, 0, dest_WrapSize, 0);
```

此处，dest_WrapSize 表示在目标地址绕回之前要传输的脉冲数，因此 dest_WrapSize 应该为 16/nWords。这可以实现为使 DMA 连续向发送缓冲区馈送数据（由 FSITX 触发，DMA 连续模式启用）。

FSIRX 的配置与 FSITX 非常相似，不同之处在于，对于 RX 缓冲区和标签以及用户数据，DMA 通道没有顺序要求。在示例项目中，使用了 DMA CH3 和 DMA CH4。

利用硬件直通特性，可以在每个节点器件接收传入数据包的同时传输该数据包。这样一来，与其他触发模式不同，每个器件不必等到接收到数据包之后，就可以将其传输到菊花链中的下一个器件。有关此特性的更多详细信息，请参阅 F280039C 器件 TRM。要启用此特性，需要设置 TX_OPER_CTRL_LO.TDM_ENABLE 和 TX_OPER_CTRL_LO.SEL_TDM_IN 寄存器字段。或者，可以使用下面的 driverLib 函数。

```
FSI_enableTxTDMMode(FSITXA_BASE);
FSI_enableRxTDMMode(FSITXA_BASE);
```

5 菊花链拓扑测试

为了演示 FSI 的通信速度和不同的配置，测试并验证了两器件配置和三器件配置的菊花链连接。所使用的测试硬件由多个 F280025C ControlCARD 评估模块和 TMDSFSIADAPEVM 构成。

备注

1.使用其他支持 FSI 的 C2000 MCU LaunchPad 和具有相似硬件设置的 ControlCARD，可以执行这些相同的测试。

2.硬件直通特性通过 LAUNCHXL-F280039C 和 TMDSFSIADAPEVM 进行测试，因为仅该器件提供此特性。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

可以在下载的 C2000WARE 中找到测试的示例项目。所有测试结果是在 Code Composer Studio™ (CCS) 中使用优化等级 - 2 得到的。更改优化级别可能会产生不同的结果。表 5-1 中显示了测试项目的总体说明。为了便于理解，图 5-1 中显示了具有不同项目设置的通用软件流程图。

表 5-1. 示例项目说明 - 菊花链

工程	说明	Settings
fsi_ex_daisy_handshake_lead	菊花链环路中主控器件的项目。	① [#define FSI_DMA_ENABLE 0 && #define FSI_RX_TDM_ENABLE 0] 表示使用 CPU 控制的 FSI 通信。 ② [#define FSI_DMA_ENABLE 1 && #define TX_DMA_TRIGGER_ENABLE 0 && #define FSI_RX_TDM_ENABLE 0] 表示使用 DMA 控制并使用软件为传输的数据触发 DMA 的 FSI 通信（手动）。 ③ [#define FSI_DMA_ENABLE 1 && #define TX_DMA_TRIGGER_ENABLE 1 && #define FSI_RX_TDM_ENABLE 0] 表示使用 DMA 控制并启用 FSITX 来为传输的数据触发 DMA 的 FSI 通信。 ④ [#define FSI_DMA_ENABLE 0 && #define FSI_RX_TDM_ENABLE 1] 表示使用 HW RX TDM 控制的 FSI 通信
fsi_ex_daisy_handshake_node	菊花链环路中 N-1 (N>=2) 个其他器件的项目。	

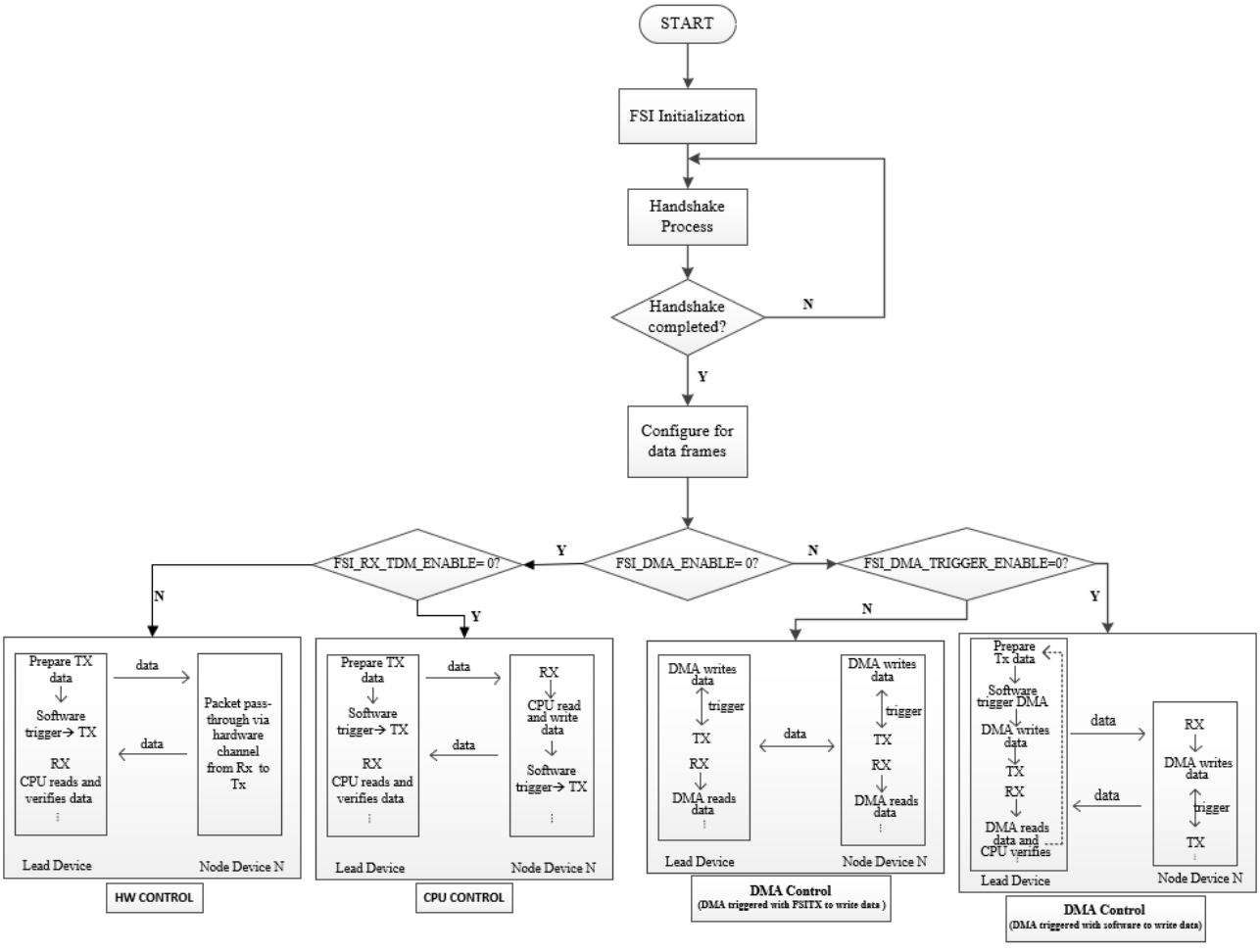

图 5-1. 具有不同项目设置的软件流程图

Copyright © 2023 Texas Instruments Incorporated

5.1 两器件 FSI 通信

在最小菊花链连接测试中，使用了一个包含两个 F280025C ControlCARD 评估模块和 TMDSFSIADAPEVM 的系统，如图 5-2 中所示。以下各小节对 CPU 控件和 DMA 控件的通信速度进行了比较。

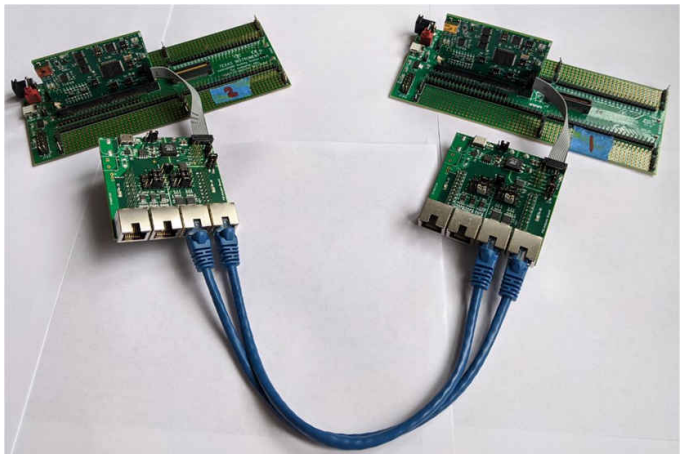

图 5-2. 针对两器件通信的测试平台

5.1.1 CPU 控制

- 测试条件

  器件 1 发送数据 -> 器件 2 接收数据 -> 器件 2 CPU 将 RX 数据移至 TX 缓冲区和寄存器 -> 器件 2 通过将接收到的数据转发回至器件 1 的软件触发 FSI TX -> 器件 1 接收返回的数据，CPU 验证这些数据是否与最初发送的 TX 数据相匹配。
- **测试案例**

  8 个字的数据长度，2 条数据线，TXCLK = 50MHz，启用设置 ①（表 5-1）。

在测试中，当通信期间发生特定事件时，会在软件内翻转通用输入/输出 (GPIO)，并使用示波器对其进行测量以获取相应的时序数据。在图 5-3 中，绿色信号表示器件 1（主控器件）的 GPIO 翻转，品红色信号表示器件 2（节点器件）的 GPIO 翻转。

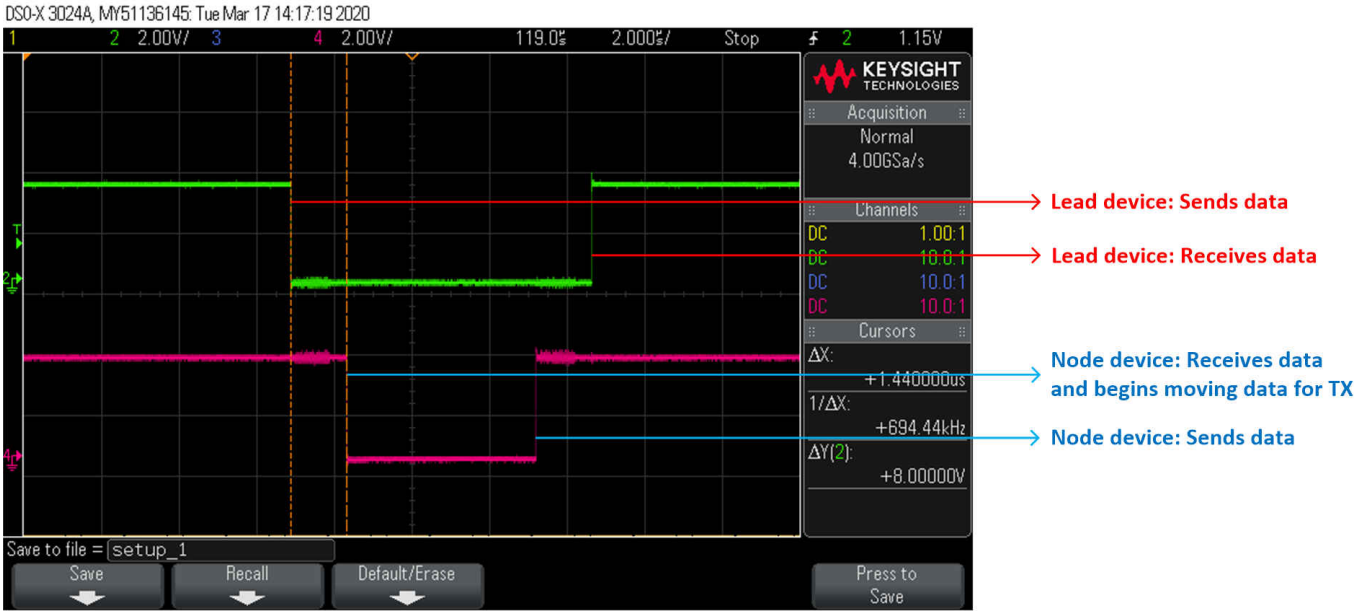

图 5-3. 采用 CPU 控件的数据传输测试

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

根据图 5-3 中所示的结果，获取的数据传输时间大约为 1.4µs。为了计算传输速度，应考虑总数据长度。表 5-2 显示了数据帧的通用结构，该结构可以分为两部分：有效数据位和开销位。

- **有效数据位：**包括 8 位用户数据、16 位数据字和 8 位 CRC 字段
- **开销位：**包括前同步码、SOF、帧类型、EOF 和后同步码字段

因此，理论上可以得出 8 个字的理想传输时间，如表 5-3 中所示。

应该注意的是，两条数据线仅适用于有效数据位，因此一个 FSITXCLK 周期提供 4 个有效数据位，而一个 FSITXCLK 周期仅提供 2 个开销位。因此，8 个数据字需要 48 个 FSITXCLK 周期，从而可以计算出传输时间，如方程式 1 中所示。

(FSITXCLK 周期数)/(FSITXCLK 频率) = 48/50MHz = 0.96 µ s (1)

因此，理论传输速度为 175Mbps（168/0.96µs），而测试速度为 120Mbps，传输时间为 1.4µs，这是由于测试的传输时间包括进入 ISR（用于翻转 IO 引脚）以及隔离器、收发器、电缆等引入的延迟。如果改为一条数据线，则理论传输速度为 100Mbps，而测试速度为 80Mbps，传输时间为 2.1µs。

图 5-3 的另一个发现是，使用 FSI driverLib 函数在节点器件中将数据从 FSIRX 缓冲区移至 FSITX 缓冲区需要一些时间，该时间大约为 4.9µs。这将是区分后续部分所示 DMA 控制和硬件控制的关键因素。

备注

可以通过直接对 FSI 寄存器进行读写（而不是使用提供的 driverLib 函数）来优化在 FSI 缓冲区和寄存器之间移动数据的时间。

表 5-2. 数据帧结构

IDLE	前导码	SOF	帧类型	用户数据	数据字	CRC	帧标签	EOF	后同步码	IDLE
	1111	1001	0011	8 位	N 个字	8 位	4 位	0110	1111	

表 5-3. 针对 8 个字计算的传输时间

有效数据位（位）	开销位（位）	总长度（位）	有效数据位的 FSITXCLK 周期（周期）	开销位的 FSITXCLK 周期（周期）	总数据传输时间 (us)
144	24	168	36	12	0.96

5.1.2 DMA 控件

- **测试条件**

  器件 1 发送数据 -> 器件 2 接收数据 -> 器件 2 DMA 将 RX 数据移至 TX 缓冲区和寄存器 -> 在对用于将接收到的数据转发回至器件 1 的 TX_FRAME_TAG_UDATA FSI 寄存器进行写入之后器件 2 触发 TX -> 器件 1 接收返回的数据 -> 器件 1 DMA 将 RX 数据移到内存中 -> CPU 验证内存中的数据是否与最初发送的 TX 数据相匹配。
- **测试案例**

  8 个字的数据长度（每个脉冲 8 个字，每次传输 1 个脉冲），2 条数据线，TXCLK = 50MHz，启用设置 ②（表 5-1）。

在该测试中，启用 CH2 和 CH4 的 DMA 中断以在主控器件中的传输结束时触发，这意味着每次将数据从内存复制到 FSITX 缓冲区 (CH2) 或将数据从 FSRX 缓冲区传输至内存中的某个位置 (CH4) 时都会发生中断。在节点器件中，DMA 通道配置为在接收到 FSI 数据帧时将接收到的数据从 RX 缓冲区和寄存器传输到 TX 缓冲区和寄存器，最终将数据转发回主控器件。因此，节点器件仅启用一个 DMA 中断，而主控器件启用了两个 DMA 中断。图 5-4 显示了在 DMA ISR 中进行 GPIO 翻转时，使用 DMA 控件的 FSI 通信所呈现的测试结果。

Copyright © 2023 Texas Instruments Incorporated

图 5-4. 使用 DMA 控件的 FSI 通信

应注意的是，所示的时间为 1.86µs，其中包含主控器件传输数据帧、节点器件将 RX 数据移至 TX 缓冲区/寄存器、ISR 进入以及翻转 GPIO 的时间。根据 DMA 管道时序要求（更多相关信息，请参阅节 8 中引用的特定于器件的 TRM），可以如方程式 2 中所示计算使用 2 个通道移动 9 个字的数据（8 个字的数据 + 1 个字的用户数据和帧标签）所需的时间。

$$(9 \times 3 \ \ cycles/Word + 2 \ \ cycles) \div (100MHz) = 0.29\mu s \tag{2}$$

因此，考虑到其他延迟时间，实际传输时间几乎与前面使用 CPU 控件的测试结果 (1.4µs) 一致。还应强调的是，在这种情况下利用 DMA 可以极大地节省传输接收到的数据所需的时间，尤其是在涉及大量数据传输的应用中。

5.1.3 硬件控制

- **测试条件**

  器件 1 发送数据 -> 器件 2 在接收数据时还使用硬件通道将数据传回器件 1 -> 器件 1 接收返回的数据，CPU 验证它是否与最初发送的 TX 数据匹配。
- **测试案例**

  8 个字的数据长度，2 条数据线，TXCLK = 30MHz，启用设置 ④（表 5-1）。

在此测试中，数据包在由器件 2 接收的同时，也会使用直通 Tx 通道发送出去。尽管器件 2 会验证数据包是否存在错误，但与其他传输模式不同，这些错误不会阻止数据包传递到器件 1。

Copyright © 2023 Texas Instruments Incorporated

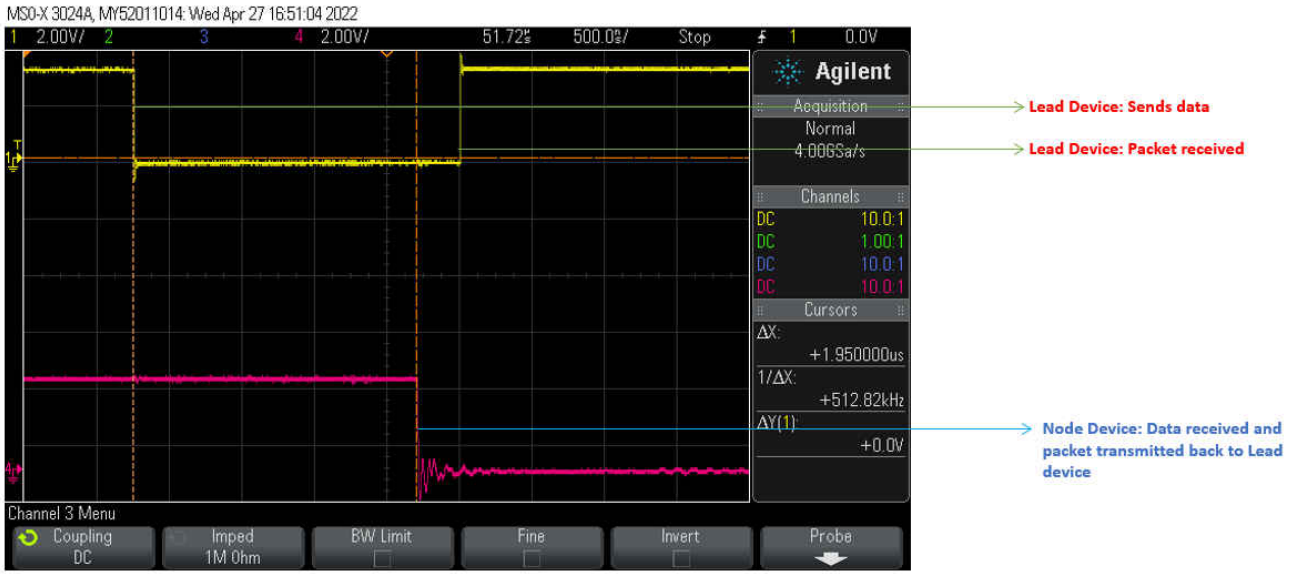

图 5-5. 使用硬件控制的 FSI 通信

在测试中，当通信期间发生特定事件时，会在软件内翻转 GPIO，并使用示波器对其进行测量以获取相应的时序数据。在图 5-5 中，黄色信号表示器件 1（主控器件）的 GPIO 翻转，品红色信号表示器件 2（节点器件）的 GPIO 翻转。根据方程式 1 中的计算，在两条数据线上以 30MHz 发送 8 个字所花费的理论传输时间为 1.6μs。因此，主控器件上发送和接收之间的理论间隔时间也等于 1.6 μ s（假设不存在有线传输延迟），这是因为硬件控制模式会运用 FSI 的直通 Rx TDM 特性，除了传输延迟之外没有任何延迟。但是，在测试中观察到的 1.95μs 值包括了翻转 GPIO 所花费的时间、隔离器引入的延迟、收发器、硬件和电缆中的信号传播延迟等。

表 5-4 中给出了更多测试结果，用于比较使用 CPU 控制、DMA 控制和硬件控制的 FSI。在 FSI 数据帧结构中固定了开销位的情况下，使用更大的数据长度来最大限度地提高有效数据吞吐量是有益的。

表 5-4. 在两个器件的 FSI 中使用 CPU 控件和 DMA 控件的比较

	FSITXCLK (MHz)	数据线路数	数据长度（16 位字）	传输时间 (μs) (1)	缓冲区数据移动时间 (μs) (1)	理论传输速度 (Mbps) (2)	测试传输速度 (Mbps)
CPU 控件	50	2	8	1.4	4.9	175	120
	50	2	16	2.1	8.3	185	141
	50	1	8	2.1	4.9	100	80
	10	1	8	8.9	4.9	20	18.9
DMA 控件	50	2	8	1.9		/	/
	50	2	16	3.0		/	/
	50	1	8	2.6		/	/
	10	1	8	9.3		/	/
硬件控制	30	2	8	1.95		/	/
	30	2	16	3.05		/	/
	12	1	8	7.3		/	/

(1) 测量时间四舍五入到最接近的 0.1μs。
(2) 在具有两条数据线的情况下，考虑了在两条线上传输的 FSI 帧开销位。

Copyright © 2023 Texas Instruments Incorporated

在某些情况下，FSI 通信可能需要一些额外的稳健性和抗噪性，因此，还测试了较低的时钟频率。FSI 协议旨在仅在发生数据交换时进行通信。这有助于降低系统中的功耗和总体 EMI。此外，较低的 FSI 时钟频率和半双工通信可以提高整体系统级 EMI 性能，同时在相同的工作频率下继续提供比通用串行端口更高的吞吐量。通常，对于板对板连接，最好为每个信号使用一条双绞线或屏蔽线，而板载 FSI 信号布线长度应匹配，并且在布局中要格外小心，以增强抗噪性能。

在执行的测试中，在 TMDSFSIADAPEVM 板上使用了隔离和差分收发器器件，这可能会导致通道间的信号偏斜。在利用这些相同或相似器件和/或不同的信号布线长度的实际应用中，FSI 接收器模块内的集成偏斜补偿块可用于管理时钟信号和数据信号之间可能发生的信号偏斜。有关该主题的更多信息，请参阅*快速串行接口 (FSI) 偏斜补偿*。

5.2 三器件 FSI 通信

还根据节 5.1 中提供的测试和比较结果，对三器件 FSI 通信进行了研究。图 5-6 中显示了所使用的测试平台。

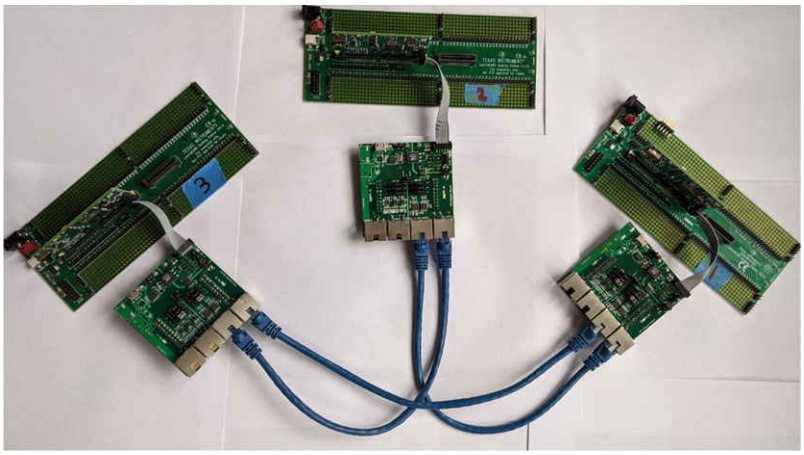

图 5-6. 针对三器件通信的测试平台

5.2.1 CPU/DMA 控制

- **测试条件**

  器件 1 发送数据 -> 器件 2 接收数据 -> 器件 2 将 RX 数据移至 TX 缓冲区并将数据发送至器件 3->…..-> 器件 3 将 RX 数据移至 TX 缓冲区并将数据发送至器件 1 -> 器件 1 接收数据并验证数据是否与最初发送的 TX 数据相匹配。
- **测试案例**

  8 个字的数据长度，1 条数据线，TXCLK = 50MHz，使用 CPU 控件时启用设置 ①，使用 DMA 控件时启用设置 ②（表 5-1）。

在测试中，当通信期间发生特定事件时，会在软件内翻转 GPIO，并使用示波器对其进行测量以获取相应的时序数据。在下图中，绿色信号表示器件 1（主控器件）的 GPIO 翻转，蓝色信号表示器件 2（节点器件）的 GPIO 翻转，品红色信号表示器件 3（节点器件）的 GPIO 翻转。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

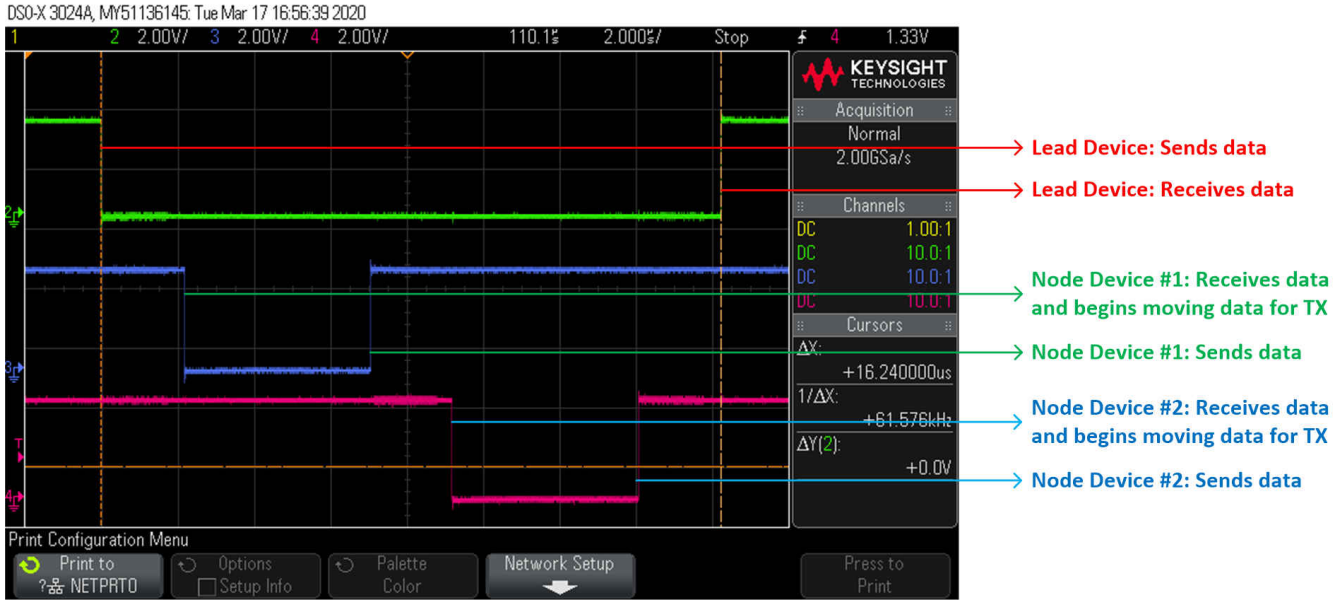

图 5-7. 在三个器件之间采用 CPU 控件的 FSI 通信

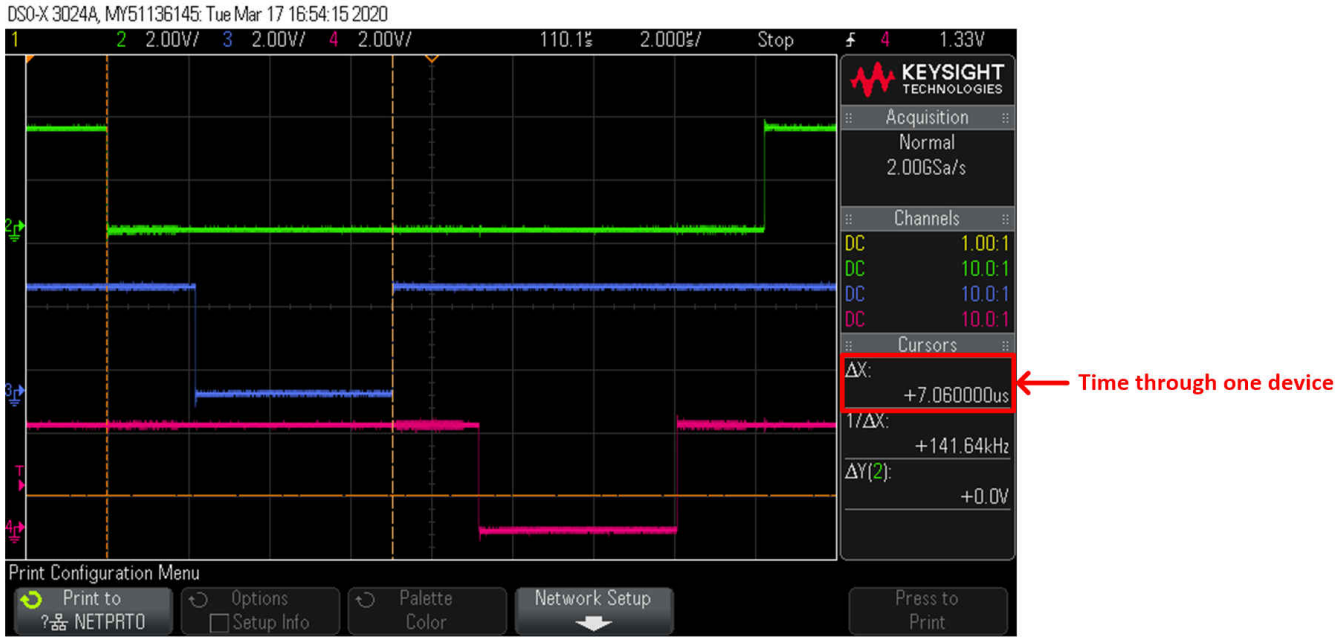

图 5-8. 数据通过一个器件的时间 - CPU 控件

对于使用 CPU 控件的情形，完成三器件菊花链环路所需的数据传输时间为 16.2µs，如图 5-7 所示。对于添加到菊花链连接系统中的每个器件，该时间将增加 7.1us，如图 5-8 所示。每个器件所增加的 7.1µs 时间包括传输时间以及将 RX 数据移到 TX 缓冲区和寄存器所需的时间。

Copyright © 2023 Texas Instruments Incorporated

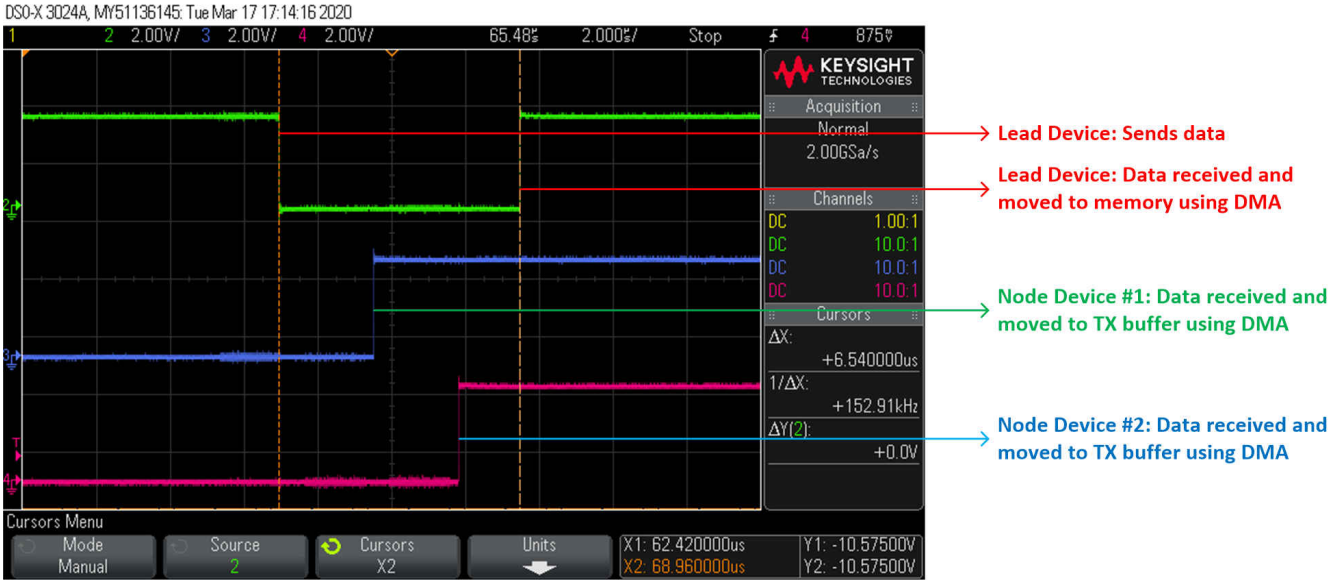

图 5-9. 三个器件之间采用 DMA 控制的 FSI 通信

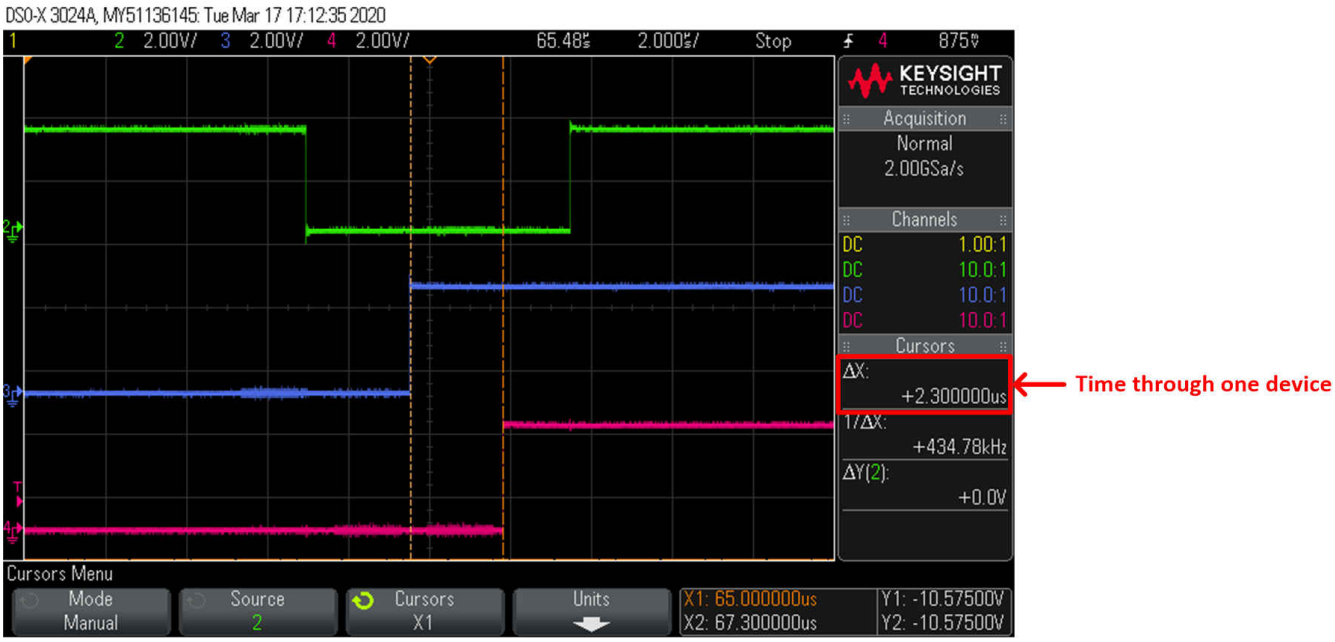

图 5-10. 数据通过一个器件的时间 - DMA 控件

对于使用 DMA 控件的情形，完成三器件菊花链环路所需的数据传输时间为 6.5μs，如图 5-9 所示。对于添加到菊花链连接系统中的每个器件，该时间将增加 2.3us，如图 5-10 所示。每个器件所增加的 2.3μs 时间包括传输时间以及将 RX 数据移到 TX 缓冲区和寄存器所需的时间。

5.2.2 硬件控制

- **测试条件**

  器件 1 发送数据 -> 器件 2 在接收数据时还将其传递到器件 3 -> 器件 3 在接收数据时还将其传递到器件 1 -> 器件 1 接收数据并验证它是否与最初发送的 Tx 数据匹配。
- **测试案例**

  8 个字的数据长度，1 条数据线，TXCLK = 30MHz，启用设置 ④（表 5-1）。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

在测试中，当通信期间发生特定事件时，会在软件内翻转 GPIO，并使用示波器对其进行测量以获取相应的时序数据。在下图中，黄色信号表示器件 1（主控器件）的 GPIO 翻转，蓝色信号表示器件 2（节点器件）的 GPIO 翻转，品红色信号表示器件 3（节点器件）的 GPIO 翻转。

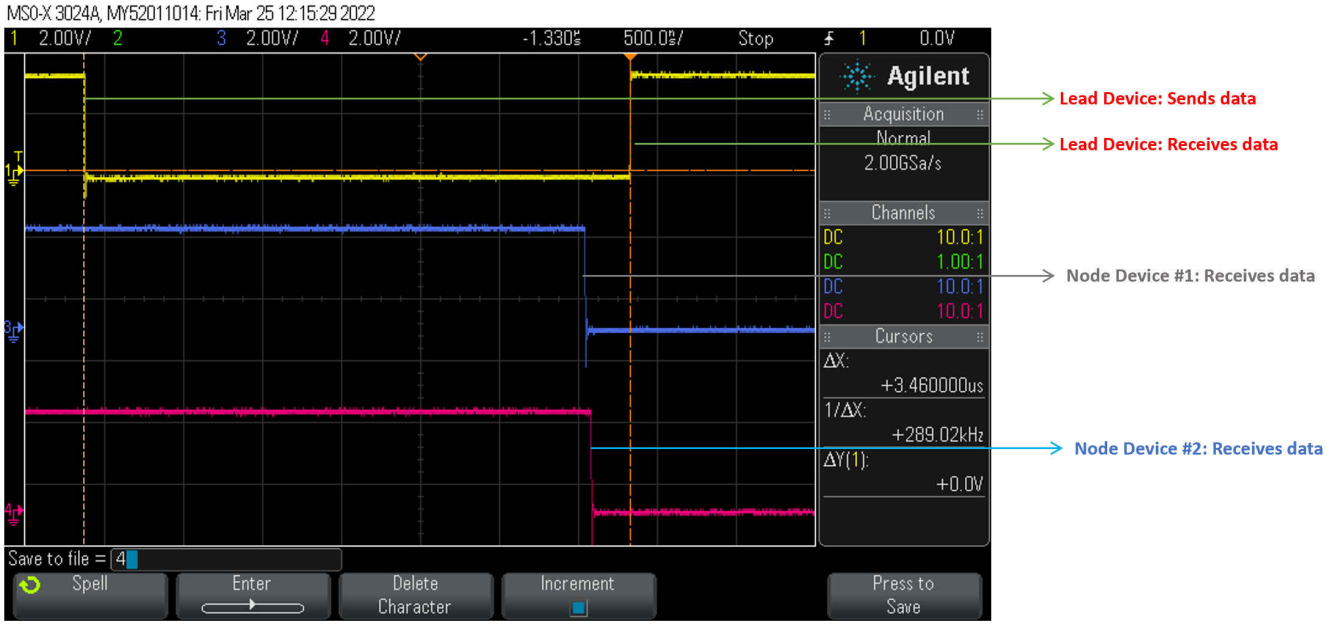

图 5-11. 三个器件之间采用硬件控制的 FSI 通信

如图 5-11 所示，对于这种情况，完成三器件菊花链环路所需的数据传输时间为 3.46µs。从图中可看出，器件 2 和器件 3 几乎同时接收数据。另请注意，器件 1 收到数据包时，甚至早于它对数据包传输后生成的 TX 帧中断完成服务，因此，即使接收数据包的时间要早很多，GPIO 翻转仍会延迟。

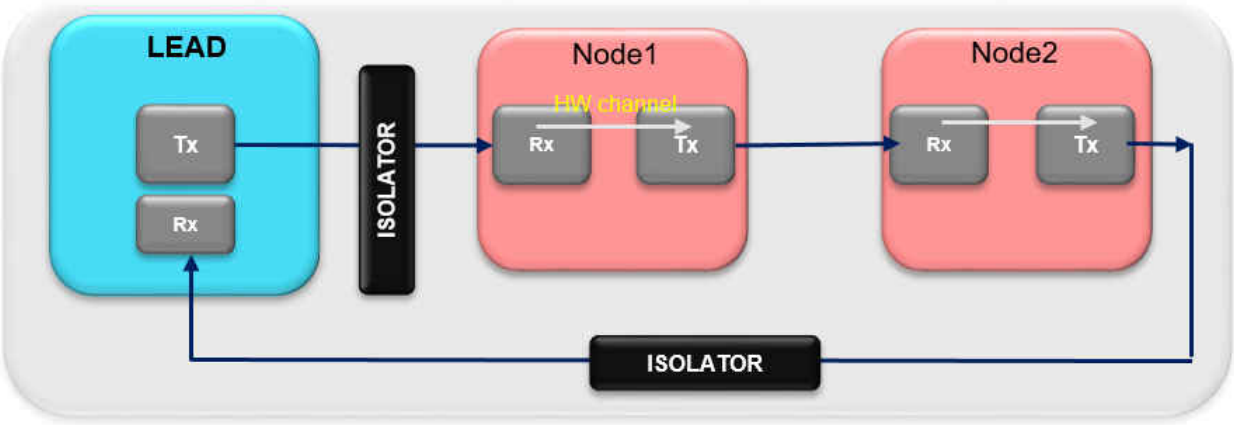

图 5-12. 用于生成硬件控制结果的三节点设置

请注意，这些结果是在假设器件 2 和器件 3 未相互隔离的情况下生成的。仅器件 1 保留隔离器。所使用的设置如图 5-12 所示。这是因为每个隔离器都会累积信号失真效应，而这会导致通信错误。

Copyright © 2023 Texas Instruments Incorporated

备注

1. 由于硬件控制使用硬件中的直通连接，因此网络中的所有器件实际上都连接在一起。这使得无法在硬件中对信号执行任何调节。
2. 在硬件控制模式下，隔离器和其他电路对 FSI 时钟和数据线的信号失真效应很明显，特别是，当三个或更多隔离器件在网络中以菊花链形式连接时，可能无法以高于 30MHz 的频率进行通信。为了在较高工作频率下尽可能提高吞吐量，建议采用 DMA 和硬件控制传输模式组合。

表 5-5 中给出了更多测试结果。

表 5-5. 三个器件之间在 FSI 中使用 CPU、DMA 和 HW 控制的比较

	FSITXCLK (MHz)	数据线路数	数据长度 (16 位字)	数据通过一个器件的时间 (µs)	实现完整连接环路的时间 - 3 个器件 (us)
CPU 控件	50	1	8	7.1	16.2
	50	1	16	11.8	26.8
	30	1	8	7.3	17.65
	30	1	16	12.2	29.66
	30	2	8	6.04	14.02
	30	2	16	9.95	22.587
DMA 控件	50	1	8	2.3	6.5
	50	1	16	4.0	11.8
	30	1	8	3.45	9.9
	30	1	16	5.9	17.3
	30	2	8	2.3	6.3
	30	2	16	3.75	10.65
硬件控制	30	1	8	~0.1	3.46
	30	1	16	~0.1	5.6
	30	2	8	~0.1	2.26
	30	2	16	~0.1	3.33

由于菊花链连接的性质，数据将需要通过许多器件才能从第一个器件传输到最后一个器件。因此，为了降低延迟，务必使每个器件中的数据处理和转发时间尽可能短，尤其是在连接环路中有多个器件时。根据节 5.1 中得出的结论，为了避免 CPU 使用过多的带宽来移动数据，建议使用 DMA 或硬件直通特性来满足 FSI 通信需求。

5.2.2.1 三器件菊花链系统的偏斜补偿

在利用数字隔离和差分收发器器件，或者信号布线长度不同的实际应用中，可以使用 FSI 接收器模块内的集成偏斜补偿块，来管理时钟信号和数据信号之间可能发生的信号偏斜。

菊花链示例嵌入了为三器件系统构建的偏斜补偿算法。此算法基于*快速串行接口 (FSI) 偏斜补偿*中提供的函数和示例。如果必须使用示例中包含的偏斜补偿算法，则 fsi_ex_daisy_handshake_node 中的 NODE_POS 设置必须如下所示：

对于器件 2

```
#define NODE_POS                    1
```

对于器件 3

```
#define NODE_POS                    2
```

出于校准目的，每个器件都会分配到一个 ID。虽然实际校准以用户定义的频率进行，但器件之间的握手调用以极低的频率完成，以确保未补偿偏斜的影响可以忽略不计。

Copyright © 2023 Texas Instruments Incorporated

每个器件的偏斜可能与其他器件不同，这是时钟和数据线上的传播延迟差异造成的。这也意味着，当传输源变化时，由一个传输源执行的偏斜补偿可能不起作用。例如，在上述 CPU/DMA 控制模式下，器件 3 的传输源是器件 2 的 Tx 模块。但在硬件控制模式下，源是偏斜补偿后的器件 2 的 Rx 通道。下图中描绘了这部分内容。 因此，分别解释了 CPU/DMA 控制模式和硬件控制的算法。

5.2.2.1.1 CPU/DMA 控制

器件 2 延迟线校准：器件 1 使用分配的 ID 调用器件 2，一旦器件 2 收到校准调用，就会使用器件 1 发送的数据包来校准其延迟线。在旁路模式下，器件 3 使用 CPU 控制将从器件 2 收到的数据包传输到器件 1。

器件 3 延迟线校准：器件 2 使用分配的 ID 调用器件 3，一旦器件 3 收到校准调用，就会使用器件 2 发送的数据包来校准其延迟线。在旁路模式下，器件 1 使用 CPU 控制将从器件 3 收到的数据包传输到器件 2。

器件 1 延迟线校准：器件 3 使用分配的 ID 调用器件 1，一旦器件 1 收到校准调用，就会使用器件 3 发送的数据包来校准其延迟线。在旁路模式下，器件 2 使用 CPU 控制将从器件 1 收到的数据包传输到器件 3。

因此，如图 图 5-13 所示，所有三个器件会分阶段进行校准，到校准结束时，菊花链中所有器件的 Rx 延迟线均会配置适当的值。

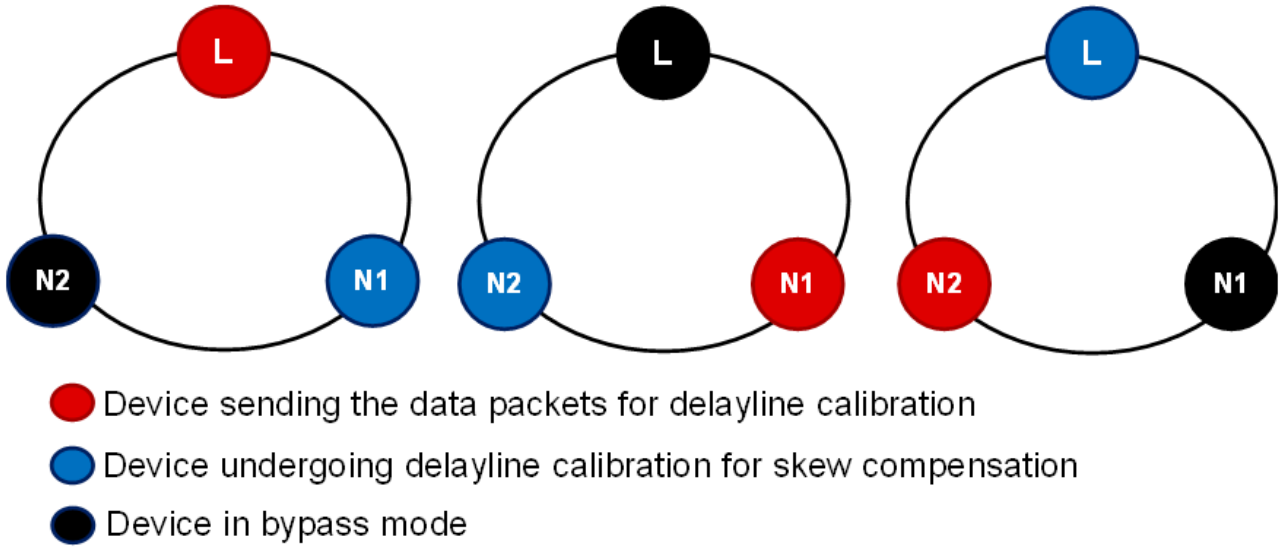

图 5-13. CPU、DMA 控制下偏斜补偿的延迟线校准图

5.2.2.1.2 硬件控制

就示例中考虑的硬件控制情况而言，器件 2 和器件 3 始终处于硬件直通模式，因此对于偏斜补偿算法，这些器件也必须处于直通模式。

器件 2 延迟线校准：器件 1 使用分配的 ID 调用器件 2，一旦器件 2 收到校准调用，就会使用器件 1 发送的数据包来校准其延迟线。在旁路模式下，器件 3 使用 CPU 控制将从器件 2 收到的数据包传输到器件 1。

器件 3 延迟线校准：器件 1 使用分配的 ID 调用器件 3，一旦器件 3 收到校准调用，就会使用器件 1 发送的数据包来校准其延迟线。在旁路模式下，器件 2 使用之前经过校准的硬件通道，将从器件 1 接收到的数据包传输到器件 3。

器件 1 延迟线校准：一旦器件 2 和器件 3 延迟线完成校准，就会进入旁路模式，此时接收到的数据包将通过每个器件的硬件通道进行传输。在这种配置下，器件 1 会通过发送数据包开始校准其延迟线，而发送的数据包会返回其 Rx。这样一来，器件 1 实际上是在外部环回网络中进行自我校准，此时环路由器件 2 和器件 3 的硬件通道构成。

因此，如图 图 5-14 所示，所有三个器件会分阶段进行校准，到校准结束时，菊花链中所有器件的 Rx 延迟线均会配置适当的值。

Copyright © 2023 Texas Instruments Incorporated

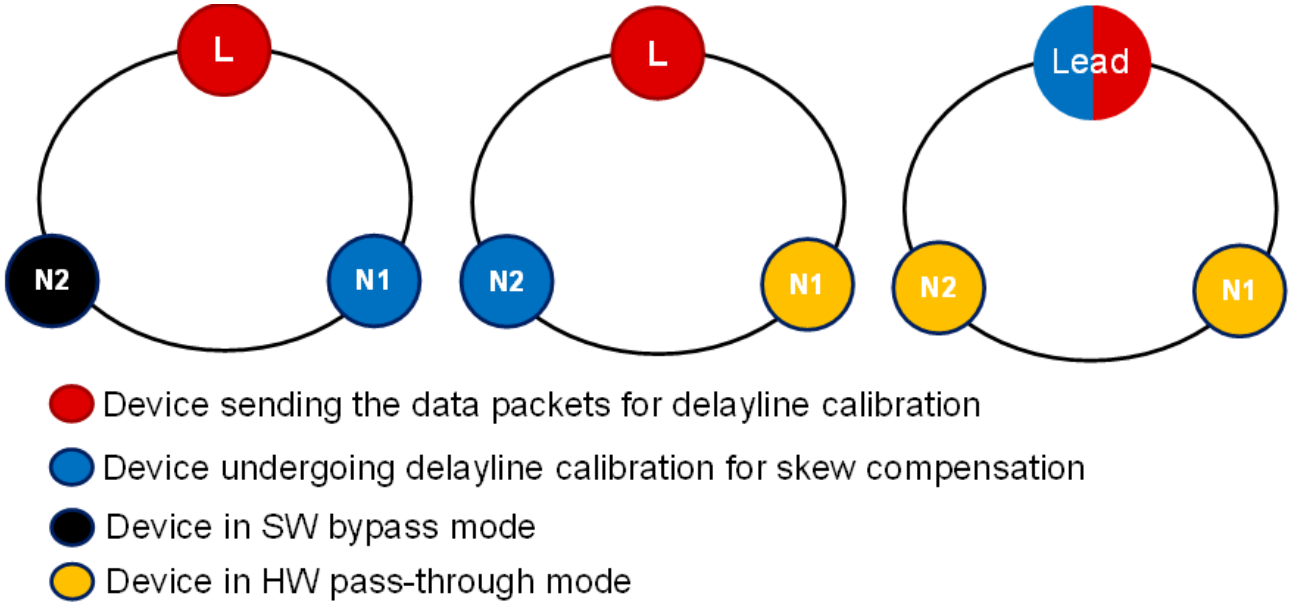

图 5-14. 硬件控制下偏斜补偿的延迟线校准图

下表列出了用于为三器件菊花链系统构建偏斜补偿算法的不同函数。如前所述，偏斜补偿算法是根据*快速串行接口 (FSI) 偏斜补偿*中的现有示例而构建的，表 5-6 仅提供为了在菊花链拓扑中进行偏斜补偿而创建的函数。

表 5-6. 偏斜补偿算法中使用的其他函数

功能	说明
`FSI_LpbkvalidatePing`	供主控器件用于传输和接收 ping 数据包，以验证为偏斜补偿提供的延迟线配置。此函数期望菊花链的其余器件处于环回模式。
`FSI_LpbkCalibrateExePoint`	此函数会为主控 Rx 三个延迟线的每种可能的配置分别调用 `FSI_LpbkvalidatePing` 函数，从中确定能够达到最优偏斜补偿的最佳配置。此函数期望菊花链的其余器件处于环回模式。

请注意，如果器件 2 和器件 3 也应进行传输，则必须扩展上述偏斜补偿算法，以便通过使校准后的系统 Rx 延迟线保持完好，校准 Tx 时钟上存在的延迟线和这些器件的数据线。

6 星型拓扑测试

基于 FSI 的星型拓扑应用示例演示了另一种通信拓扑，其中展示了中央主机器件如何同时从多个节点器件接收信息，而不是像在菊花链中那样等待数据包通过后续器件转发。节 2 中讨论了星型拓扑的优缺点。

所提供的星型实现与硬件有关，即主机器件 TX 需要具备针对每个节点的多点连接功能，此外还与 MCU 资源有关，即主机器件需要有 N 个 RX 实例。从软件角度而言，中央主机器件使用新的 *star_broadcast* 项目，而 N 个节点设备使用和菊花链测试中相同的软件。表 6-1 中提供了详细信息。

表 6-1. 软件示例项目 - 星型拓扑

工程	说明	支持的器件
fsi_ex_star_broadcast	星型网络中中央主机器件的项目	F2838x
fsi_ex_daisy_handshake_node	星型网络中 N 个节点器件的项目	F28002x、F28004x、F2838x

star_broadcast 项目的软件流与节 5 中讨论的主控器件 CPU 控制菊花链项目的软件流类似。如节 3.2 所示，握手机制会稍有不同。握手完成后，中央主机器件会向与其 FSITX 连接的所有节点器件发送广播数据帧。然后，主机等待从所有连接的节点器件接收返回的数据帧，然后验证每个接收到的帧是否与最初传输的帧匹配，接下来主机将进行准备并发送新的数据帧。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

默认情况下，*star_broadcast* 项目具有针对器件的 FSI RX 实例 A、B 和 C 的预制配置。可以通过将以下各条预处理器指令设置为“1”来配置每个实例。如果主机器件上提供了其他 FSI RX 实例，则可以添加这些实例。

```
//
// 启用 FSI RX 实例
//
#define FSI_RXA_ENABLE          1
#define FSI_RXB_ENABLE          0
#define FSI_RXC_ENABLE          1
```

星型拓扑的时序测量值将与先前菊花链测试中收集的时序测量值非常类似，甚至相同。 因此，可以将表 5-4 中提供的数据用于此目的。

7 通过 FSI 进行事件同步

本节介绍了使用快速串行接口模块实现多器件菊花链和星型配置的事件同步。

7.1 引言

7.1.1 分布式系统的事件同步需求

在任何实时控制系统（其中多个器件一同运行以执行一项任务）中，器件根据控制环路功能发送和接收关键信息。数据从一个器件传输到另一个器件所花费的时间可能因器件而异，这是多个因素造成的，包括器件之间的距离、因制造不确定性导致工作期间出现的器件时钟轻微偏差、热效应、老化等。如果网络中的任何器件上接收到的数据延迟或不正确，则时间关键型控制环路可能会失败。以固定的时间间隔同步器件的运行，以纠正由于不确定性造成的不同步，这对用户来说很重要。任何事件都可能需要同步，例如 PWM 信号、ADC 转换启动 (ADC SoC)、通过 GPIO 的外部触发器等。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

在实时应用中，可以采用带有主控器件和节点器件的多种配置，如节 2 中所述的星型网络或菊花链网络。从图 7-1 和图 7-2 中可以简单了解星型和菊花链配置。所有节点器件均可在不同的电压电平下运行，并且可与控制器保持不同的距离。因此，从主控器件发送的任何控制信号与在每个节点器件接收的信号在时间上可能有显著差异。每个节点器件上的信号将在不同的时间点到达，导致每个节点上所有运行器件异步运行。如果信号以不同的时间间隔馈送到各个器件，则可能会产生虚假操作，从而导致控制环路应用崩溃。为了避免系统故障，各器件同步运行就显得尤为重要。

其目的是以最小的触发延迟和事件抖动来同步网络拓扑中的所有器件，而且除了器件之间现有的通信通道外，不使用任何额外的线路。使用快速串行接口 (FSI) 通信协议即可实现此事件同步。

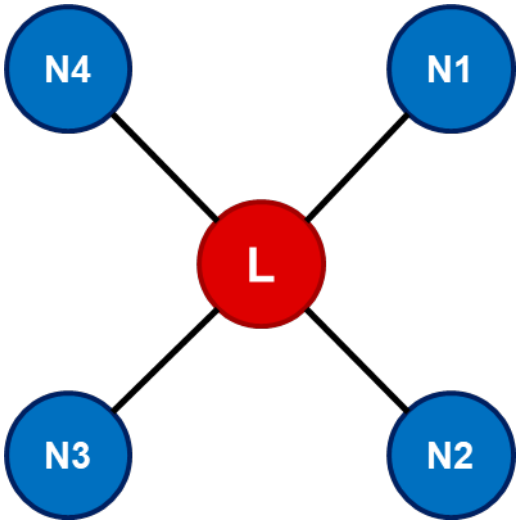

图 7-1. 星型网络配置

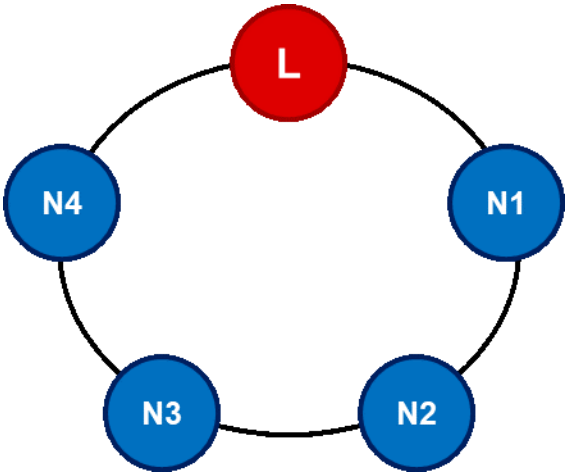

图 7-2. 菊花链网络配置

7.1.2 采用 FSI 事件同步机制的解决方案

快速串行接口 (FSI) 能够在不同的工作电压电平上通过隔离栅以高达 200Mbps 的速率发送数据。FSI 可以在多器件网络拓扑中实现不超过 100 纳秒的时间同步事件触发。这样就不需要在器件之间设置任何额外的同步信号，从而节省器件资源和额外的设计成本。

若要通过 FSI 实现事件同步机制，主控器件应使用类似于计时器的操作定期触发 FSITX 模块，以向网络中的所有器件发送同步请求包。收到来自主控器件的数据包后，每个节点器件可以启动本地同步机制。

同步数据包本身可能以不同的时间间隔到达每个节点器件。必须在本地调整对各节点器件上同步数据包的解释。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

利用 FSI 模块的 1 个时钟和 1 条数据线，以频繁的时间间隔发送同步包，确保所有节点器件保持同步。特定于应用的常规数据通过这些相同的线路发送，因此减少了确保所有节点器件之间保持同步所需的额外连接。

7.1.3 FSI 事件同步机制功能概述

为了展示事件同步机制，我们在所提供的示例中介绍了菊花链网络中 EPWM 同步的具体情况。任务是确保各节点器件的所有 EPWM 信号与主控器件的 EPWM 信号保持同步。一般来说，可以根据应用，使用 FSI 事件同步配置对任何事件进行同步。所有控制应用都使用 EPWM 时基，因此仅需保持所有节点器件的 EPWM 同步即可。通常，与 ADC、比较器或节点器件中的任何事件功能相关的活动会基于 EPWM 事件或触发器。

假设有 1 个主控器件和 2 个节点器件，如图 7-3 所示，便可理解如何通过 FSI 进行 PWM 同步。在菊花链网络中，首先使用节 3.1 中所述的握手机制建立 FSI 通信链路。

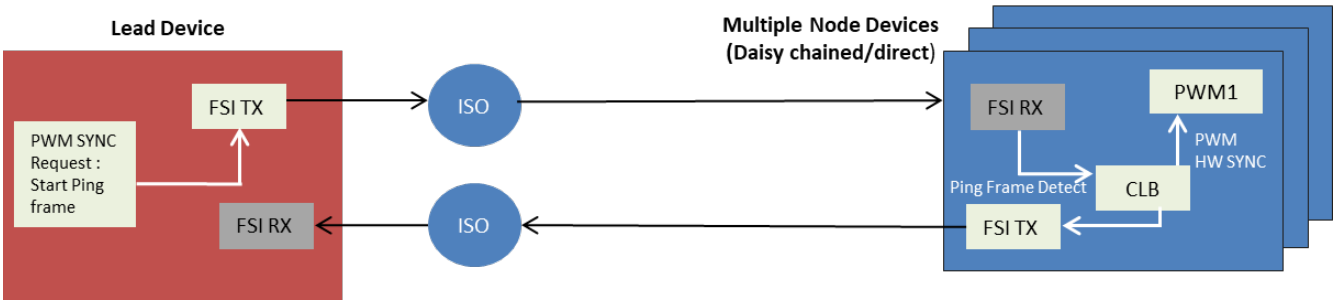

图 7-3. 菊花链通信方框图

主控器件以固定的时间间隔向各节点器件发送 Ping 数据包（就长度而言是最短的 FSI 数据包），以验证通信链路，并通知节点器件与主控器件同步。主控器件 FSITX 触发器由用户配置的本地 EPWM 比较事件进行控制。在配置的 EPWM 的上升沿或下降沿，触发 FSITX 以向节点器件发送 Ping 数据包。EPWM 信号的计数器比较值取决于链中的器件数量、器件之间的距离等，该值由用户根据具体应用进行配置。

在某个节点接收到主控器件发送的 Ping 数据包后，该节点处生成的 FSI Ping 帧接收信号 (RX_PING_FRAME) 便会在内部连接至同一节点的可配置逻辑块 (CLB) 模块，如图 7-3 所示。CLB 路由 RX_PING_FRAME 信号以立即触发 FSI TX Ping 转发到链中的下一器件。在 CLB 内部实现的可配置延迟计数器充当计时器，将周期作为“*match*”值馈送到 CLB，并在达到“*match*”值时重新开始计数。当 CLB 收到 RX_PING_FRAME 信号时，计数器开始计数，在达到“*match*”值时，它会为该节点的 EPWM 模块生成 EPWM 同步输入信号。换句话说，“*match*”值有助于计算出从节点收到 Ping 数据包信号开始到本地节点器件生成 PWM 同步输入信号为止的延迟。理想情况下，每个节点的 EPWM 同步输入信号将同时生成，并与主控器件 EPWM 计数器等于零事件对齐。图 7-4 展示了节点器件内部流程的功能表示。

Copyright © 2023 Texas Instruments Incorporated

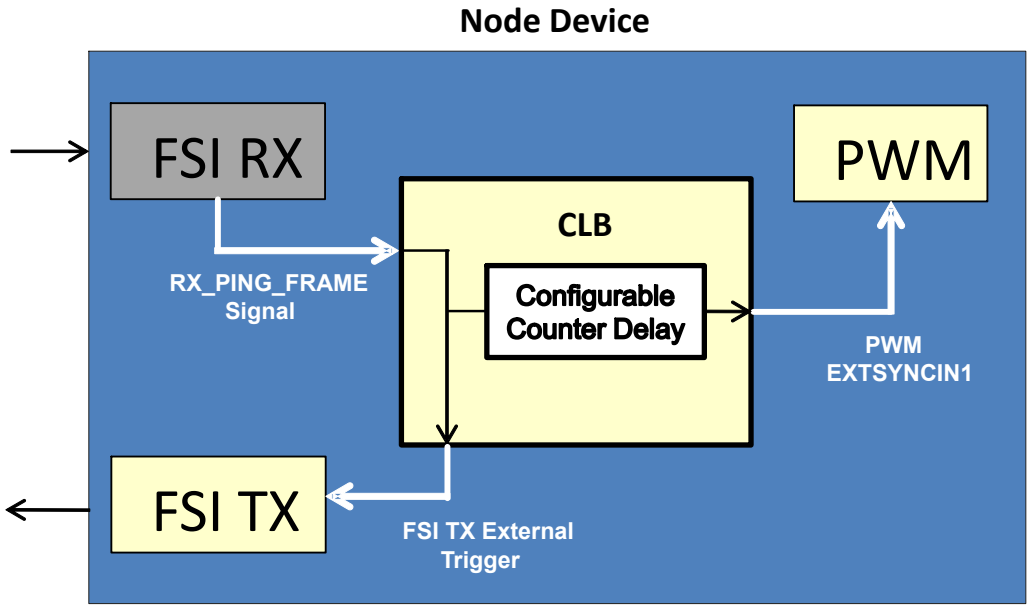

图 7-4. 节点器件实现

PWM 信号不会随着时间的推移而保持同步，因为每个器件的振荡器时钟不可能都是相同的。链中的节点器件会经历其各自 EPWM 信号中存在的抖动形式的不确定性，如图 7-5 所示。对于菊花链配置，链中的最后一个器件将经历最大的抖动量，因为每个器件的数据包转发和同步器偏差都会产生附加抖动，节 7.2.5 中对此进行了讨论。这种利用 Ping 数据包生成 EPWM 同步输入信号的方式将使链中所有器件的 EPWM 保持同步。

生成 EPWM 信号所需的延迟因菊花链网络中的节点而异。例如，对于节点 1，将在短时间内收到 Ping 数据包，而对于第 8 个节点，由于转发和传播延迟，Ping 数据包信号到达该节点所需的时间会更长。因此，节点 1 的 CLB 计数器计数（“*match*”值）必须更高，这样才能确保 EPWM 同步输入信号生成的延迟大于节点 8 的计数，从而根据传输延迟进行调整。要馈送到计数器的“*match*”值必须由用户设置并在 CLB 配置块中配置，这取决于所使用的隔离栅、节点之间的距离和运行中的器件。工程源文件的简介一节中提供了“*match*”计数开始时的近似值。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

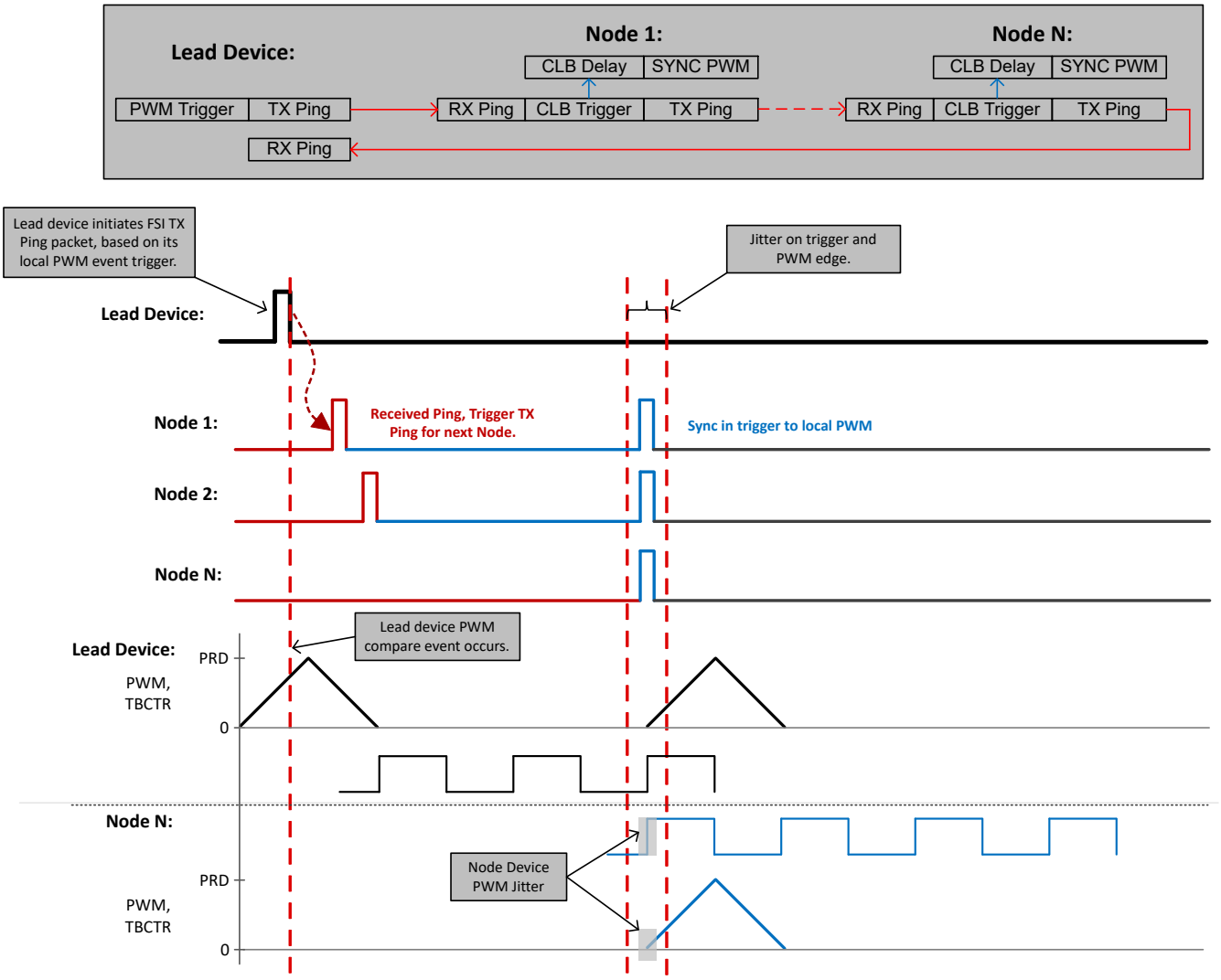

图 7-5. 菊花链同步时序图

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

7.2 C2000Ware FSI EPWM 同步示例

本节提供了有关 C2000Ware 中所述 FSI 事件同步示例的技术细节和一些其他细节。

7.2.1 C2000Ware 示例工程的位置

C2000Ware 3.04 及更高版本中展示了通过快速串行接口模块显示 PWM 同步功能的示例。相同的位置可在 <C2000Ware_ROOT_FOLDER\examples\demos> 文件夹中找到，名为 *f28004x_fsi_daisychain_epwmsync* 和 *f28002x_fsi_daisychain_epwmsync*。我们针对两个器件系列（f28004x 和 f28002x）展示了该功能。每个器件文件夹包含 2 个工程，一个用于主控器件，另一个用于节点器件。除了主控和节点器件文件夹外，“*common*”文件夹还包含必要的目标配置文件。节 7.3.2 中提到了有关配置目标配置文件的详细信息。这些工程可在 controlCARD 或 LaunchPad 上运行。

7.2.2 软件配置综述

7.2.2.1 主控器件配置

以固定时间间隔发送 Ping 数据包的主控器件帧传输触发过程是使用 EPWM1C 模块完成的。主控器件配置为根据 EPWM 比较事件发送 Ping 数据包。用户需要根据硬件配置，在代码中配置 EPWM 计数器比较 C 寄存器 EPWM_CMPC_VALUE。该值取决于外部因素，如两个节点之间的距离、器件振荡器时钟、隔离栅等。发送 ping 数据包的频率可以是 EPWM 时钟频率的分数，还可以通过配置 EPWM_CMPC_EVENT_COUNT 参数进一步调整。EPWM_CMPC_EVENT_COUNT 参数设置 EPWM 事件触发预分频器值，该值将帧传输触发配置为在每个第 N 次比较事件出现时发生。预分频器的最大值为 15。

可以在代码中通过更新 EPWM_CMPA_VALUE（EPWM1A 占空比）、EPWM_CMPB_VALUE（EPWM1B 占空比）和 EPWM_TIMER_TBPRD（频率）来调整 EPWM 的占空比和频率。默认值是在 20 kHz 的 EPWM 频率下占空比为 50%。

7.2.2.2 节点器件配置

节点器件在收到 Ping 数据包后，必须根据节点中器件的顺序，在一定的延迟后生成 EPWM 同步信号。在随工程一同提供的 *fsi_daisy_epwmsync_node.syscfg* 文件中完成延迟配置。如图 7-6 中所示，必须使用“Counter0”选项卡下可配置逻辑块 (CLB) 中的计数器值“*match*”来完成配置。FSM 块包含的逻辑是，一旦获得 Ping 数据包接收信号，在匹配计数延迟后生成 EPWM 同步输入信号。使用 CPU1_CLB 构建配置构建工程后，可在该工程的“syscfg”选项卡下的 clb.html 文件中看到使用 sysconfig 文件完成的 CLB 配置，如图 7-7 所示。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

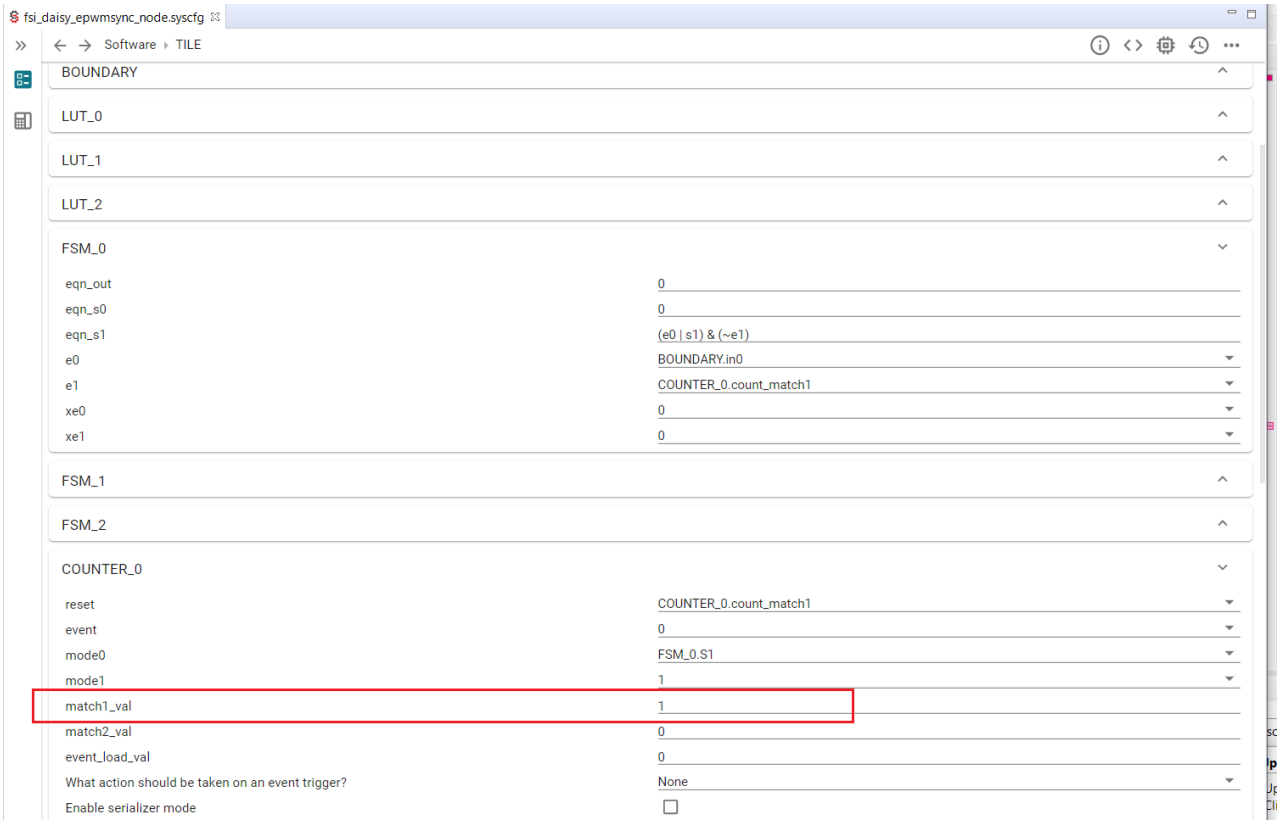

图 7-6. CLB 块配置

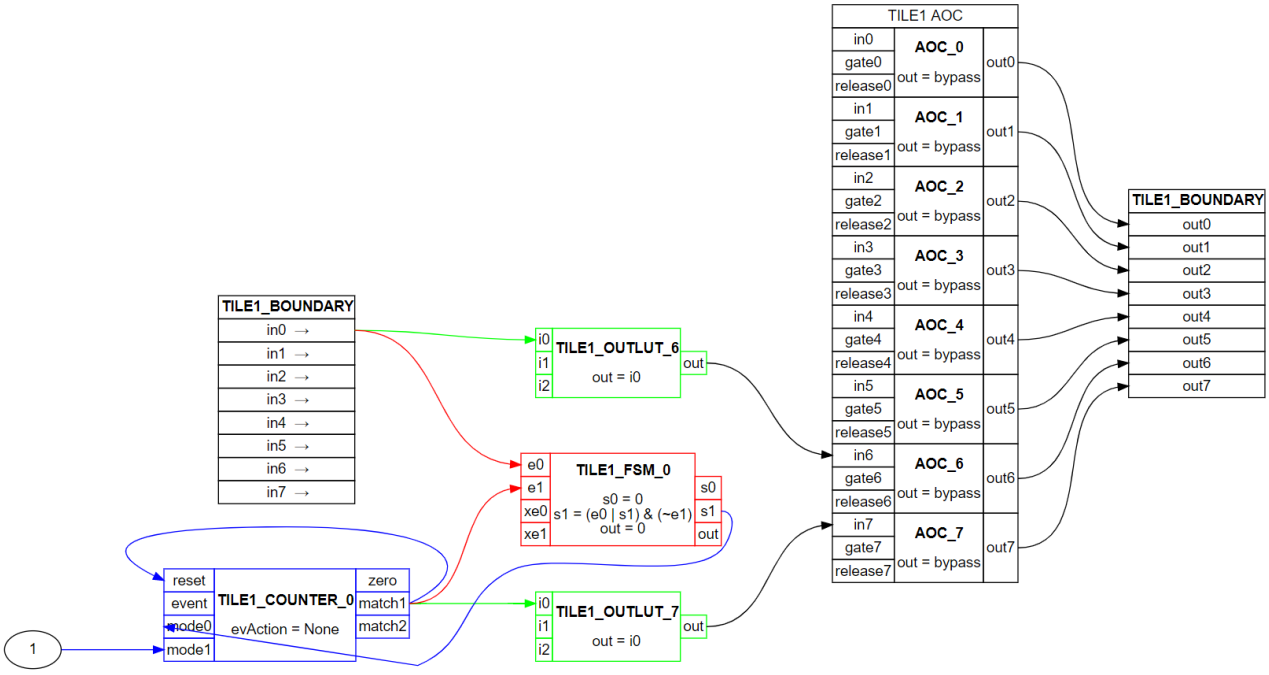

图 7-7. 配置的 CLB 块

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

与主控器件类似，用户可以使用 EPWM_TIMER_TBPRD 作为频率、使用 EPWM_CMPA_VALUE 作为 EPWM1A 占空比以及使用 EPWM_CMPB_VALUE 作为 EPWM1B 占空比，为相应节点配置 EPWM 频率和占空比。

F28004x 和 F28002x 的 CLB 配置略有不同。F28004x 器件必须通过 EPWM XBAR 将 CLBx_OUT 路由到 FSITX 外部触发器连接件，而 F28002x 器件已更新为将 FSITX 外部连接件直接连接到 CLBx_OUT。用户可以参阅器件参考手册，了解具体信息。

7.2.3 1 主控和 2 节点 F28002x 器件菊花链测试

7.2.3.1 硬件设置和配置

提供的菊花链示例在 1 个主控 LAUNCHXL-F280025C 器件和 2 个节点 LAUNCHXL-F280025C 器件以及每个主控器件和节点器件的 TMDSFSIADAPEVM 板上运行。这些器件使用长度为 1ft 的电缆进行连接。假定工作 PWM 频率为 20kHz，占空比为 50%。FSI 时钟配置为以 50MHz 频率运行。CLB 计数器的“*match*”值按照 43*(N-1) 计算，节点 1 的“*match*”值为 43，节点 2 的值为 1。

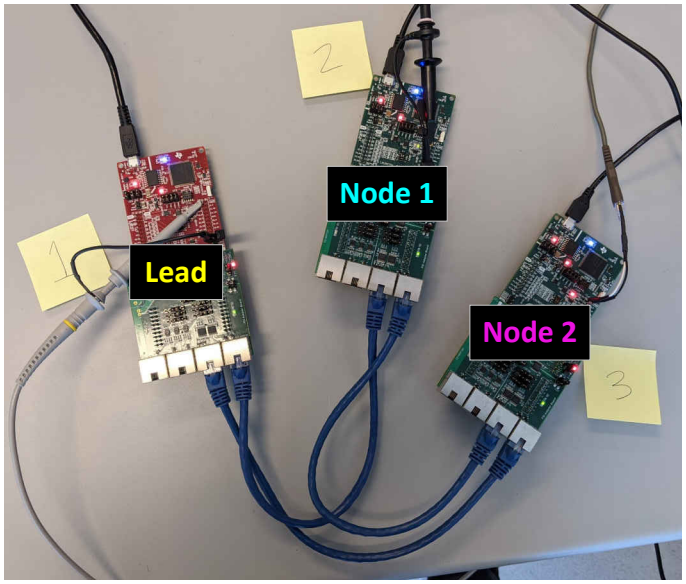

图 7-8. 1 主控 - 2 节点器件设置

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

7.2.3.2 试验结果

首先，在未启用通过 FSI 进行 EPWM 同步功能的情况下捕获 EPWM 信号，如图 7-9 所示。在未启用 FSI 事件同步解决方案的情况下，由于每个器件的振荡器时钟存在微小差异，节点器件 EPWM 信号会逐渐与主控器件的 EPWM 信号异相。

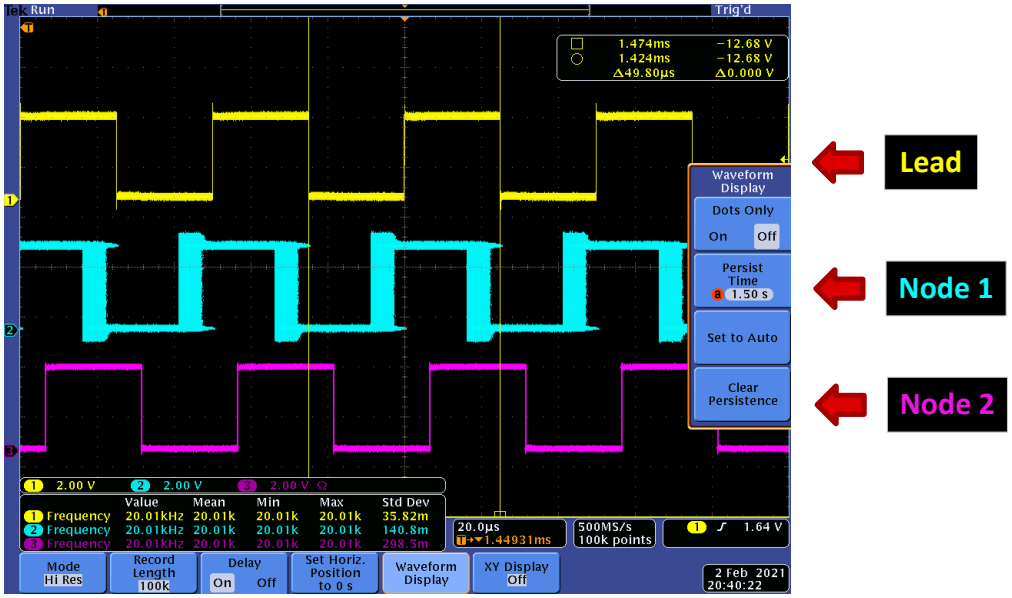

图 7-9. 1 主控 - 2 节点器件（未启用 EPWM 同步）

启用 EPWM 同步后捕获的 EPWM 信号，如图 7-10 所示。

运行该示例一周后，捕获主控器件和两个节点器件的 EPWM 上升沿抖动，从而获取所有可能的不确定性，如图 7-10 所示。理想情况下，节点器件上测量的抖动应尽可能小。它是由于器件、隔离栅、通信链路等之间的不一致而在链中出现的噪声。CLB 还会引入导致抖动的特定延迟，这称为节 7.2.5 中所述的同步器延迟。器件数量增加以及节点间距变大会导致 PWM 抖动加剧。

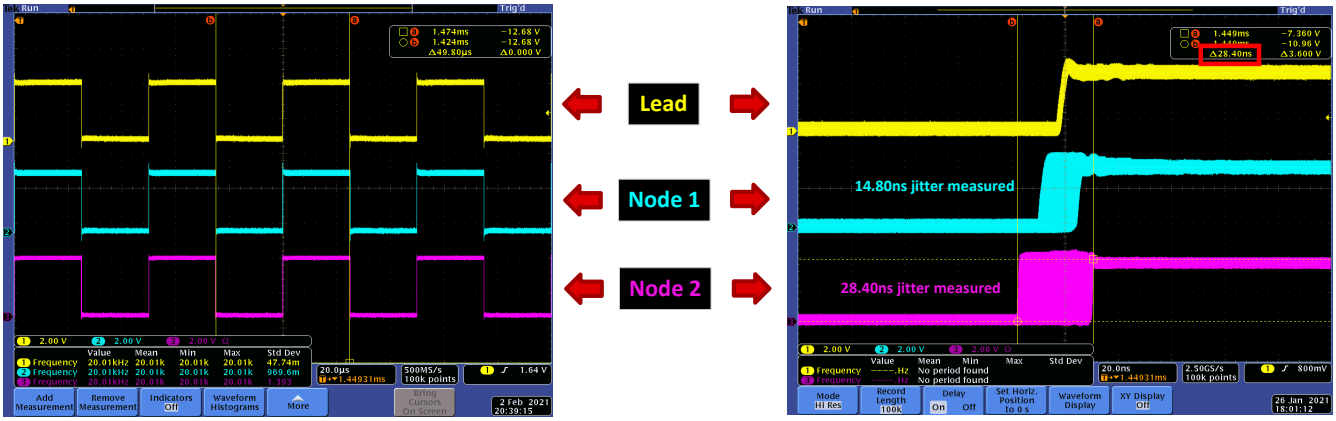

图 7-10. 1 主控 - 2 节点器件（同步 EPWM）

Copyright © 2023 Texas Instruments Incorporated

7.2.4 1 主控和 8 节点 F28002x 器件菊花链测试

7.2.4.1 硬件设置和配置

本示例展示了 1 个主控器件和 8 个节点器件的运行，每个节点之间的电缆长度为 1ft，但最后两个节点的电缆长度较长，为 5m，如图 7-11 所示。

每个节点上的 PWM 输出的占空比保持在 50%，频率为 20kHz。

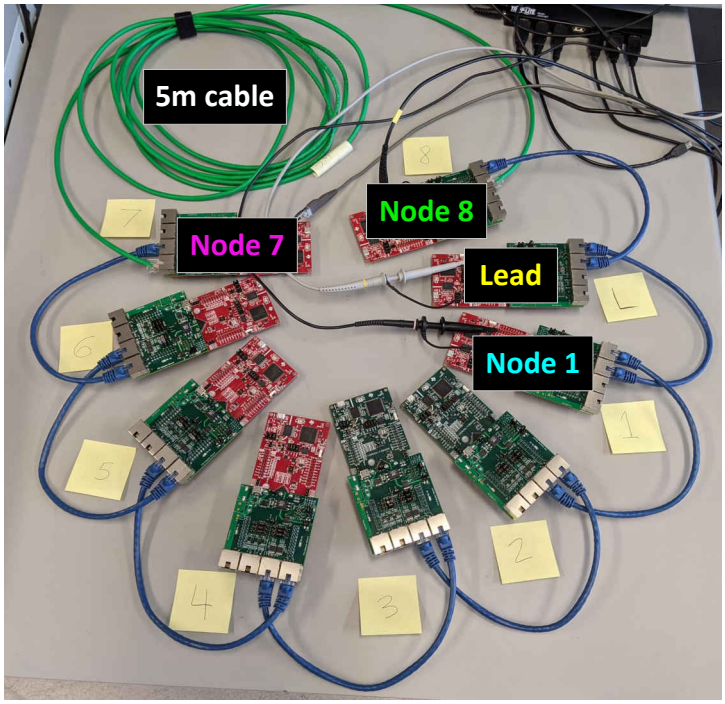

图 7-11. 1 主控 - 8 节点器件设置

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

7.2.4.2 试验结果

在示波器中捕获节点 1、节点 7 和节点 8 的 EPWM 波形以验证同步运行。对未启用 EPWM 同步情况下和启用 EPWM 同步情况下捕获的结果进行了比较，以展示通过 FSI 实现 EPWM 同步的能力。可以清楚地观察到与 2 节点配置的抖动差异。

在第 8 个节点观察到的最大抖动（经过一周的测试以捕获所有不确定性）在 100 纳秒以内，这对 EPWM 频率的边缘没有太大影响。

图 7-12. 1 主控 - 8 节点器件（未启用 EPWM 同步）

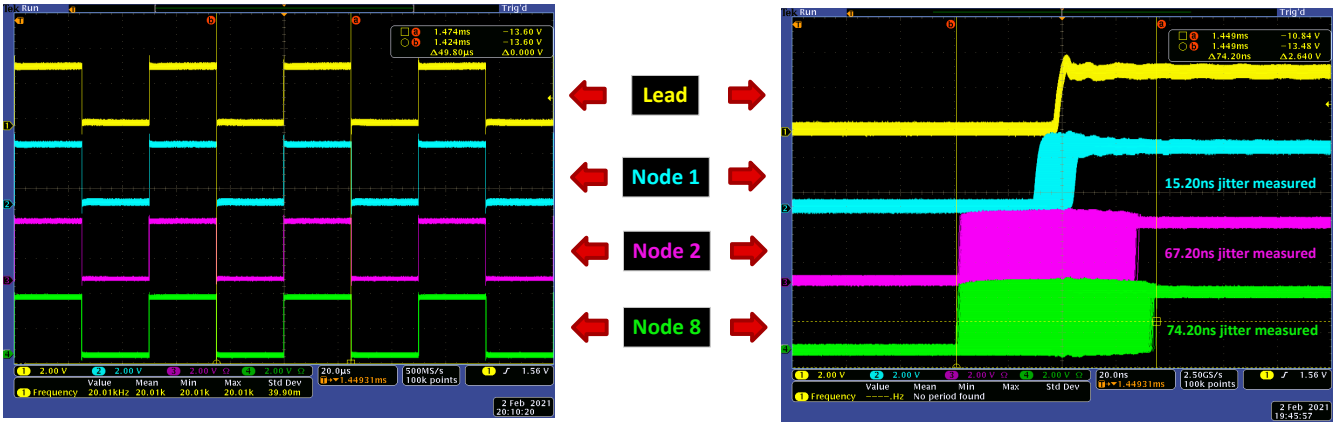

图 7-13. 1 主控 - 8 节点器件（同步 EPWM）

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

7.2.5 C2000 理论上的不确定性

前面几节所述的事件抖动是由器件与器件之间的振荡器时钟不确定性、器件之间的传输距离、数字隔离器和/或差分器件干扰等因素造成的。器件内部配置的从 FSIRX 到 CLB 模块再到 FSITX 的信号路径的同步也会进一步增加不确定性，如图 7-14 所示。

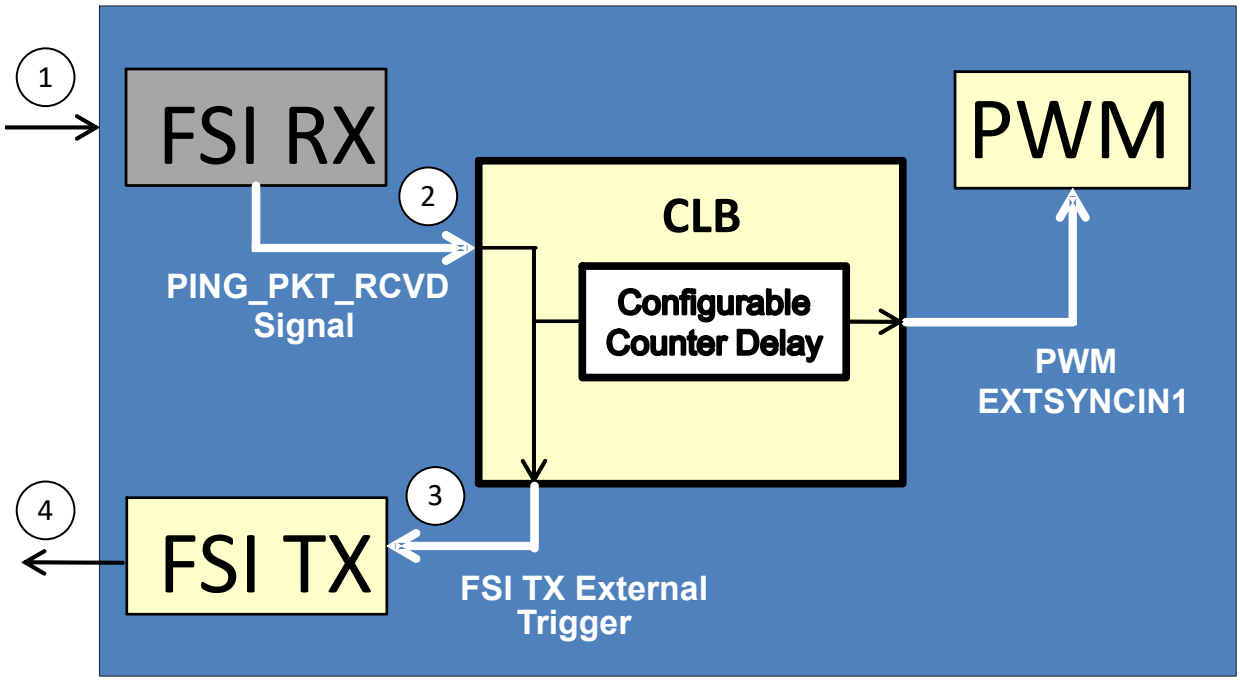

图 7-14. 内部 C2000 FSI 事件触发路径

时钟域沿着内部 C2000 FSI 事件触发路径，在 FSI RX 时钟、器件 SYSCLK（CLB 时钟）和 FSI TX 时钟之间变化。每次信号从一个时钟域移至另一时钟域时，都会遇到一个同步器，该同步器会增加一定量的可变周期延时（不是每次都是相同的数字）。下面描述了图 7-14 所示的第 1 至 4 段事件触发路径的理论循环计数不确定性。

路径 1-2：RX 模块接收到 FSI Ping 帧以生成 PING_PKT_RCVD 信号。时钟域从 FSI RX CLK 变为 SYSCLK 会导致 0-2 个 SYSCLK 周期的不确定性。

路径 2-3：Ping_PKT_RCVD 信号通过 CLB 模块外部触发 FSI TX 模块发送 Ping 帧。时钟域从 SYSCLK 变为 FSI TX CLK 会导致 0-2 个 SYSCLK 周期的不确定性。

路径 3-4：FSI TX 外部触发到 FSI TX ping 帧生成。时钟域从 SYSCLK（CLB 时钟）变为 FSI TX CLK 会导致 0-2 个 FSI TX CLK 周期的不确定性。

从上面可以计算出总的最坏情况理论周期不确定性：

备注

总的最坏情况理论周期不确定性 = 4 个 SYSCLK 周期 + 2 个 FSI TXCLK 周期。

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

对于以 100MHz 的 SYSCLK 频率运行的器件，假设 TXCLK 与 SYSCLK 相同，则理论上计算出的抖动量约为 40 纳秒。这是最坏情况下的理论考虑，因此所得的实际值将只是计算出的理论值的一小部分。对于网络拓扑中的任何器件而言，从统计学上看，得到这个最坏情况值的可能性很低。这可以从前面的实验结果中看出，测得的 8 个节点器件的实际抖动量为 75 纳秒，而假设链中每个器件产生的抖动为 40 纳秒，则理论上最坏的情况将是 320 纳秒。

7.3 FSI 事件同步的其他提示和用法

7.3.1 运行示例

若要运行示例，必须在菊花链配置中连接 1 个主控器件以及“n”个节点器件（F28004x 或 F28002x）。必须在 CCS 中导入各器件的主控和节点示例。成功导入工程后，打开“View”选项卡下可用的目标配置块。右键点击目标配置块内的任意空白区域，用户可以导入任何现有目标配置。必须从名为 *<fsi_daisychain_1lead_2node.ccxml>* 的主工程文件夹中可用的“*common*”文件夹导入目标配置文件。给定文件是为 1 个主控器件和 2 个节点器件配置的，用户可以根据硬件要求对其进行配置。下面的节 7.3.2 详细介绍了如何将新器件添加到现有目标配置文件。

使用目标配置文件成功添加并测试器件连接后，必须通过右键点击目标配置文件来启动目标配置。请注意，启动目标配置文件之前，必须成功构建主控和节点工程。启动目标配置文件后，必须使用顶部菜单栏中提供的连接和加载图标连接到器件并将工程加载到器件。

另一种运行工程的方法是在不同的工作区中打开多个 CCS 窗口（与器件数量相等），并在每个器件的专用 CCS 窗口分别构建和加载工程。

可在节点工程的 watch 变量中添加变量“*data_frame_cntr*”、“*error*”和“*FSIRxintCount*”，通过 CCS 窗口观察正在传输和接收的数据帧。用户也可能想使用“*epwm_cmpc_value_var*”和“*epwm_sync_event_count_var*”在无限循环的运行时环境中动态更改占空比和预分频器值。

7.3.2 目标配置文件

目标配置文件 (.ccxml) 有助于连接一个或多个目标硬件。对于给定示例中的多个目标硬件，适当地设置目标配置文件以确保成功连接到每个器件，这点非常重要。*TI 文档* 中提供了有关连接多个器件和更新器件序列号的详细文档。关于开发自定义目标配置文件和了解器件的调试探针的其他文档，可参阅链接 *[1]* 和 *[2]*。*自定义目标配置* 文档中提供了关于查找和更新现有目标配置的步骤。

备注

“*common*”文件夹中提供的目标配置文件已为器件配置了序列号，但对于您来说，情况并非如此。对于该用例，您可以在目标配置文件中对序列号重新编程，也可以通过所提供链接中提到的步骤更新调试探针的序列号。

Copyright © 2023 Texas Instruments Incorporated

7.3.3 星型配置事件同步的用法

与雏菊链配置相比，星型配置不存在器件间传输的延迟，因此预计会简单得多，并能够提供更好的性能。影响多个器件 EPWM 信号同步的因素是振荡器时钟的制造不确定性、器件之间的距离以及隔离器件的隔离器。菊花链示例除了星型连接相关因素外，还包含器件到器件通信的附加延迟，并且对于链中的最后一个器件，器件不确定性将会累加，导致情况更加复杂。

用户需要对现有代码稍作更改，以确保星型配置的 EPWM 同步。主控器件与节点器件之间的所有链路都直接相连。对于 C2000Ware 示例代码，星型配置将使所有 CLB 计数器“*match*”值处于相似范围内，具体取决于主控器件与节点器件之间的距离以及通信路径中的不确定性。“*match*”值不会像菊花链配置中那样从一个器件累加到另一器件。因此，星型事件同步将提供较低的事件抖动。选择的主控器件所包含的 FSIRX 模块数量须大于或等于节点数量，以便一个主控器件能够与多个节点器件连接。有关星型配置的详细信息，请参阅节 6。

8 参考文献

- 德州仪器 (TI)：*TMS320F28002x 微控制器数据表*
- 德州仪器 (TI)：*TMS320F28002x 微控制器技术参考手册*
- 德州仪器 (TI)：*TMS320F2838x 微控制器数据表*
- 德州仪器 (TI)：*TMS320F2838x 微控制器技术参考手册*
- 德州仪器 (TI)：*TMS320F28004x 微控制器数据表*
- 德州仪器 (TI)：*TMS320F28004x 微控制器技术参考手册*
- 德州仪器 (TI)：*TIDM-02006 基于快速串行接口 (FSI) 的分布式多轴伺服驱动器参考设计*
- 德州仪器 (TI)：*采用多个基于 FSI 的 MCU 的分布式电源控制架构*
- TMDSFSIADAPEVM：FSI 适配器板
- F280025C ControlCARD 评估模块
- LAUNCHXL-F280025C 评估模块

9 修订历史记录

注：以前版本的页码可能与当前版本的页码不同

Changes from Revision D (August 2021) to Revision E (January 2023)	Page
• 更新了整个文档中的表格、图和交叉参考的编号格式	3
• 更新了 节 5	8
• 添加了节 5.1.3	12
• 添加了节 5.2.2	16
• 添加了节 5.2.2.1	18

English Document: SPRACM3
Copyright © 2023 Texas Instruments Incorporated

重要声明和免责声明

TI“按原样”提供技术和可靠性数据（包括数据表）、设计资源（包括参考设计）、应用或其他设计建议、网络工具、安全信息和其他资源，不保证没有瑕疵且不做出任何明示或暗示的担保，包括但不限于对适销性、某特定用途方面的适用性或不侵犯任何第三方知识产权的暗示担保。

这些资源可供使用 TI 产品进行设计的熟练开发人员使用。您将自行承担以下全部责任：(1) 针对您的应用选择合适的 TI 产品，(2) 设计、验证并测试您的应用，(3) 确保您的应用满足相应标准以及任何其他功能安全、信息安全、监管或其他要求。

这些资源如有变更，恕不另行通知。TI 授权您仅可将这些资源用于研发本资源所述的 TI 产品的应用。严禁对这些资源进行其他复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。您应全额赔偿因在这些资源的使用中对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，TI 对此概不负责。

TI 提供的产品受 TI 的销售条款或 ti.com 上其他适用条款/TI 产品随附的其他适用条款的约束。TI 提供这些资源并不会扩展或以其他方式更改 TI 针对 TI 产品发布的适用的担保或担保免责声明。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265
Copyright © 2023，德州仪器 (TI) 公司