Application Note

使用 8b-10b 线路编码和可编程实时单元的驱动器内通信

Chen Gao, Thomas Leyer

摘要

在工业电机驱动系统中，通常需要多个器件或芯片以快速、低抖动、低延迟和同步的方式相互通信。典型的示例应用是使用标准接口（如串行外设接口）或物理层中的自定义协议（如 8b-10b 线路编码）进行驱动器内通信。

可编程实时单元 (PRU) 的独特之处在于，它可以使用 960 位宽的数据总线执行单周期功能，从而使用户在实时执行通信和控制应用时不会出现抖动。TI Sitara™ 处理器提供两种类型的 PRU 子系统：PRU-ICSS 和 PRU_ICSSG。PRU-ICSS 可用于 AM335x、AM437x 和 AM57x 系列。AM243x、AM65x 和 AM64x 上具有 PRU_ICSSG。

本应用手册介绍了如何使用 PRU 实现 8b-10b 线路编码，以实现 100Mbps 数据速率的驱动器内通信。本文档还介绍了用于高速信号传输的低电压差分信令 (LVDS) 接口和多点低电压差分信令 (M-LVDS) 接口。

内容

插图清单

表格清单

商标

1 8b-10b 线路编码简介

顾名思义，8b-10b 表示将 8 位数据作为 10 位符号发送，以实现直流平衡和有界差异。数据的五个低位被编码为一个 6 位组（5b 和 6b 段），三个高位被编码为一个 4 位组（3b 和 4b 段）。这些代码组连接在一起形成 10 位符号，该符号通过线路发送和接收，然后反向解码。该算法由 IBM 公司发明并获得专利。

为了平衡高频数据接收时的直流信号，插入的两个控制位有助于使至少 20 位的串中的 0 和 1 的计数差不超过 2，并且一行中不超过 5 个 1 或 0。此方案有助于在高频数据接收期间平衡直流信号，并减少对传输信号所需的通道带宽下限的需求。然而，使用 8b-10b 编码算法会给每个字符增加 25% 的开销。这种字符开销不是唯一的开销，但却是最重要的因素。图 1-1 显示了 8b-10b 编码示例。

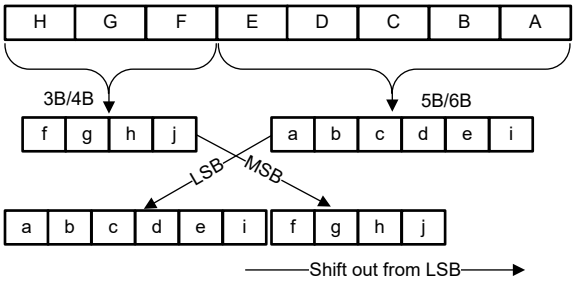

图 1-1. 8b-10b 编码

2 用于数据发送和接收的 PRU 实现

2.1 编码和解码数据

通常，有四个表预先写入存储器，包括编码 5b-6b 表、编码 3b-4b 表、解码 6b-5b 表和解码 4b-3b 表。数据必须在发送之前使用查找表 (LUT) 进行编码，并在接收之后使用 LUT 进行解码。为了演示编码和解码数据，选择 Sitara™ AM243x LaunchPad™ 开发套件作为验证器件。HW_WR_REG8 函数可用于使用 Arm® 内核项目的 C 代码将四个表放入具有不同偏移地址的 PRU 动态随机存取存储器 (DRAM) 中，另请参阅以下代码。

```
uint32_t enc_5b6b = CSL_PRU_ICSSG0_DRAM1_SLV_RAM_BASE + ENC_5B6B_OFFS;
uint32_t enc_3b4b = CSL_PRU_ICSSG0_DRAM1_SLV_RAM_BASE + ENC_3B4B_OFFS;
uint32_t dec_5b6b = CSL_PRU_ICSSG0_DRAM0_SLV_RAM_BASE + DEC_5B6B_OFFS;
uint32_t dec_3b4b = CSL_PRU_ICSSG0_DRAM0_SLV_RAM_BASE + DEC_3B4B_OFFS;
    //Encoding LUTs (input MSB first, output LSB first)
    //LUT 5b/6b encoding
HW_WR_REG8(enc_5b6b + 0x00, 0x18);
...
    //LUT 3b/4b encoding
HW_WR_REG8(enc_3b4b + 0x00, 0x04);
...
    //Decoding LUTs (input LSB first, output MSB first)
    //LUT 6b/5b decoding
HW_WR_REG8(dec_5b6b + 0x00, INVAL);
...
    //LUT 4b/3b decoding
HW_WR_REG8(dec_3b4b + 0x00, INVAL);
...
```

数据可以通过使用加载字节突发 (LBBO) 指令的 PRU 固件项目中的 LUT 进行编码。LBBO 指令用于将内存中的数据块读入寄存器文件。REG_TMP11 寄存器存储 LUT 头地址，REG_ENC 寄存器存储原始数据和编码数据。请参阅以下对 8 位数据 (0x34) 进行编码的代码：

```
Ldi REG_ENC.b0, 0x14                        ;raw data 5b LSB
ldi REG_TMP11, (PDMEM00+LUT_5b6b_ENC)       ; TEMP11 for 5b/6b LUT header
lbbo &REG_ENC.b3, REG_TMP11, REG_ENC.b0, 1 ; FNC.b0 for original 5 bit -data, ENC.b3 for encoded 6
bit
ldi REG_ENC.b1, 0x01                        ; raw data 3b MSB
ldi REG_TMP11, (PDMEM00+LUT_3b4b_ENC)       ; TEMP11 for 3b/4b LUT header
lbbo &REG_ENC.b2, REG_TMP11, REG_ENC.b1, 1 ; FNC.b1 for original 3 bit -data, ENC.b2 for encoded 4
bit
```

0x34 的五个低位为 0x14，这些位 会立即加载到 REG_ENC 的字节 0 中。在 LUT 之后，将六个编码位写入 REG_ENC 的字节 3 中。0x34 的三个高位为 0x01，这些位会直接加载 到 REG_ENC 的字节 1 中。在 LUT 之后，将四个编码位写入 REG_ENC 的字节 2 中。

图 2-1 显示了原始数据和编码数据的 REG_ENC 寄存器分布。

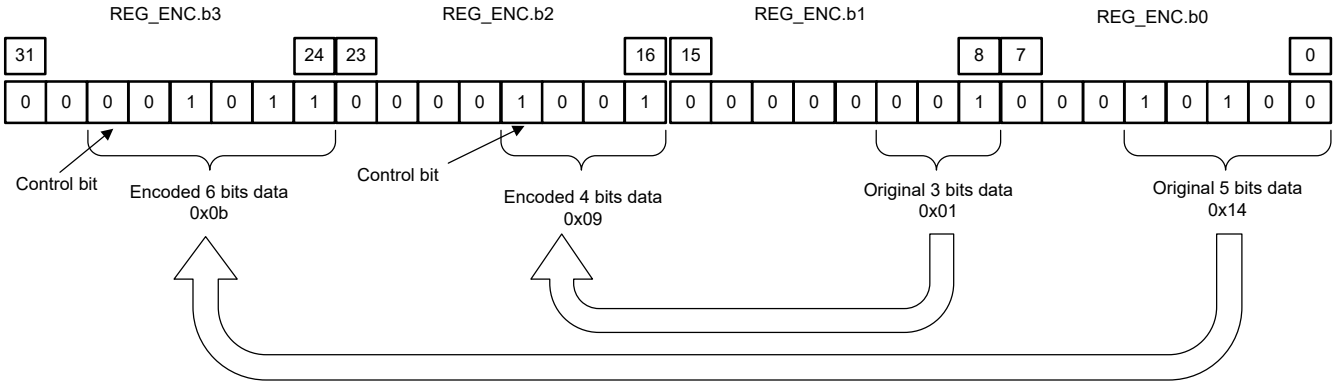

图 2-1. REG_ENC 编码寄存器分布

解码过程与编码过程几乎相同。解码过程还将 LBBO 指令与 LUT 一起，用于在接收到编码数据之后进行解码。REG_TMP11 寄存器存储 LUT 头地址，REG_DEC 寄存器存储编码数据和解码数据。

将接收六个编码位，并将其移至 REG_DEC 的字节 1 中。在 LUT 之后，将这五个解码位写入 REG_DEC 的字节 2 中。将接收四个高位，并将其移入 REG_DEC 的字节 0 中。在 LUT 之后，将这三个解码位写入 REG_DEC 的字节 3 中。要将寄存器的两个字节中的 10 位数据合并为 8 位数据，REG_DEC 的字节 3 需要左移 5 位，并与 REG_DEC 的字节 2 逻辑相加。最终解码的 8 位数据存储在 REG_DEC 的字节 0 中。以下代码显示了解码过程。

```
ldi REG_TMP11, (PDMEM00+LUT_5b6b_DEC)       ; TEMP11 for 5b/6b LUT header
lbbo &REG_DEC.b2, REG_TMP11, REG_DEC.b1, 1 ; decode 6b
ldi REG_TMP11, (PDMEM00+LUT_3b4b_DEC)       ; TEMP11 for 3b/4b LUT header
lbbo &REG_DEC.b3, REG_TMP11, REG_DEC.b0, 1 ; decode 4b
lsl REG_DEC.b3, REG_DEC.b3, 5               ; shift left 5 bit
add REG_DEC, REG_DEC.b2, REG_DEC.b3         ; combine to 8 bit data
```

图 2-2 显示了编码数据和解码数据的 REG_DEC 寄存器分布。

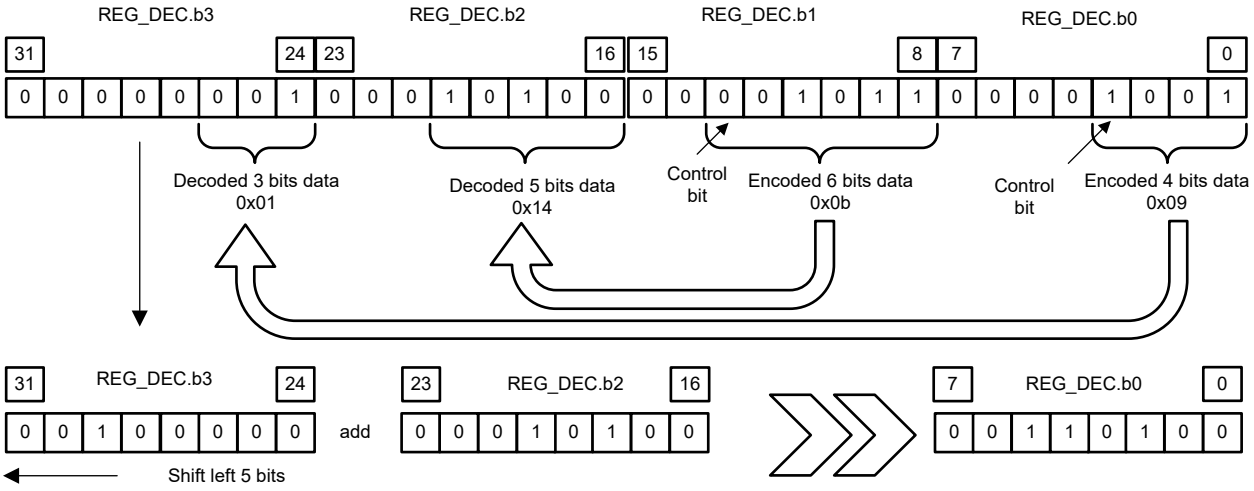

图 2-2. REG_DEC 解码寄存器分布

2.2 PRU 模块接口和 GPIO 模式

PRU 模块接口由 PRU 内部寄存器 30 和 31 (R30 和 R31) 组成。图 2-3 显示了 PRU 模块接口，以及 R30 和 R31 的功能。寄存器 R31 用作专用 PRU 通用输入 (GPI) 引脚与 PRU 中断控制器 (INTC) 之间的接口。读取 R31 可使用 PRU 实时状态接口返回来自 GPI 引脚和 PRU INTC 的状态信息。写入 R31 会通过 PRU 事件接口生成 PRU 系统事件。寄存器 R30 用作与专用 PRU 通用输出 (GPO) 引脚的接口。

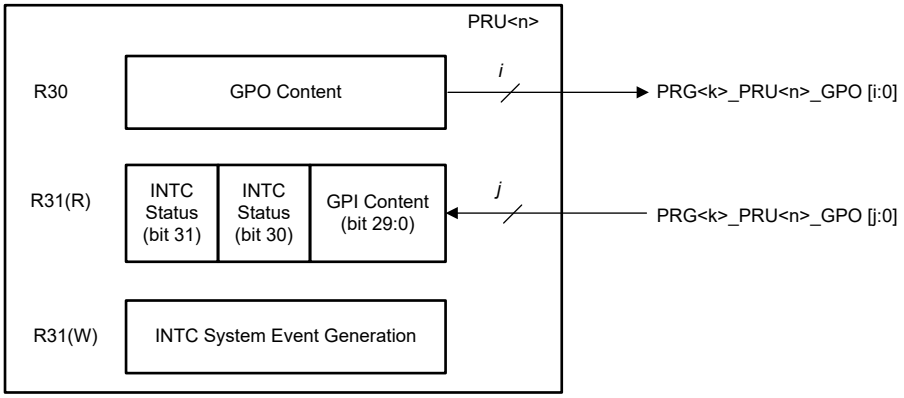

图 2-3. PRU 模块接口

PRU 事件接口直接从 PRU 的内部算术和逻辑单元 (ALU) 发送脉冲事件信息。这些事件从 PRU 中导出，需要连接到片上系统 (SoC) 级别的系统中断控制器。固件可使用事件接口创建从 PRU 到 Arm® 内核 (主机处理器) 的软件中断。例如，当通信帧封装完成向 Arm® 发送中断信号时，可能会生成该事件。

PRU 实现了一个支持以下通用输入模式的增强型通用输入或输出 (GPIO) 模块：直接输入、16 位并行采集、28 位串行移入和 MII_RT (以太网 MAC 接口) 。寄存器 R31 用作通用输入的接口。R31 还支持两种通用输出模式：直接输出和移出。寄存器 R30 用作通用输出的接口。

2.3 用于通信的 PRU GPIO 移出和移入模式

PRU 增强型通用输出 (EGPO) 可配置为以移出模式进行数据传输，数据在 PRU GPO1 (CLOCKOUT) 的每个上升沿移出 PRU GPO0 (DATAOUT)。移位速率由应用于 PRU 内核时钟的两个级联分频器的有效除数控制。为了通过 8b-10b 编码实现 100Mbit 数据速率，必须将 125MHz 时钟与除以 2 的 250MHz 内核时钟结合使用。移出模式支持两种时钟子模式：自由运行时钟模式 (默认) 和固定时钟计数模式。对于固定模式，数据包长度最多可配置为 255 位，并且在 ICSSG_GPECFG<n>_REG 寄存器的位 17 中设置了一个额外的 PRU_GPO_SHIFT_CLK_DONE 标志。

两个 16 位影子寄存器（GPO_SH0 和 GPO_SH1）用于支持乒乓缓冲器。每个影子寄存器都有独立的负载控制，这些负载控制可通过 PRU R30 寄存器的位 29 和 30 进行编程。请注意，GP_SH0 寄存器需要加载 0x8000 作为起始帧，因为移入 GPI 检测到起始位为第一个 1 或 0，而该起始位不包含在数据帧中。加载起始帧后，可以将更新的数据不受任何限制地加载到 GP_SH0 中。图 2-4 显示了 PRU 移出模式的方框图。

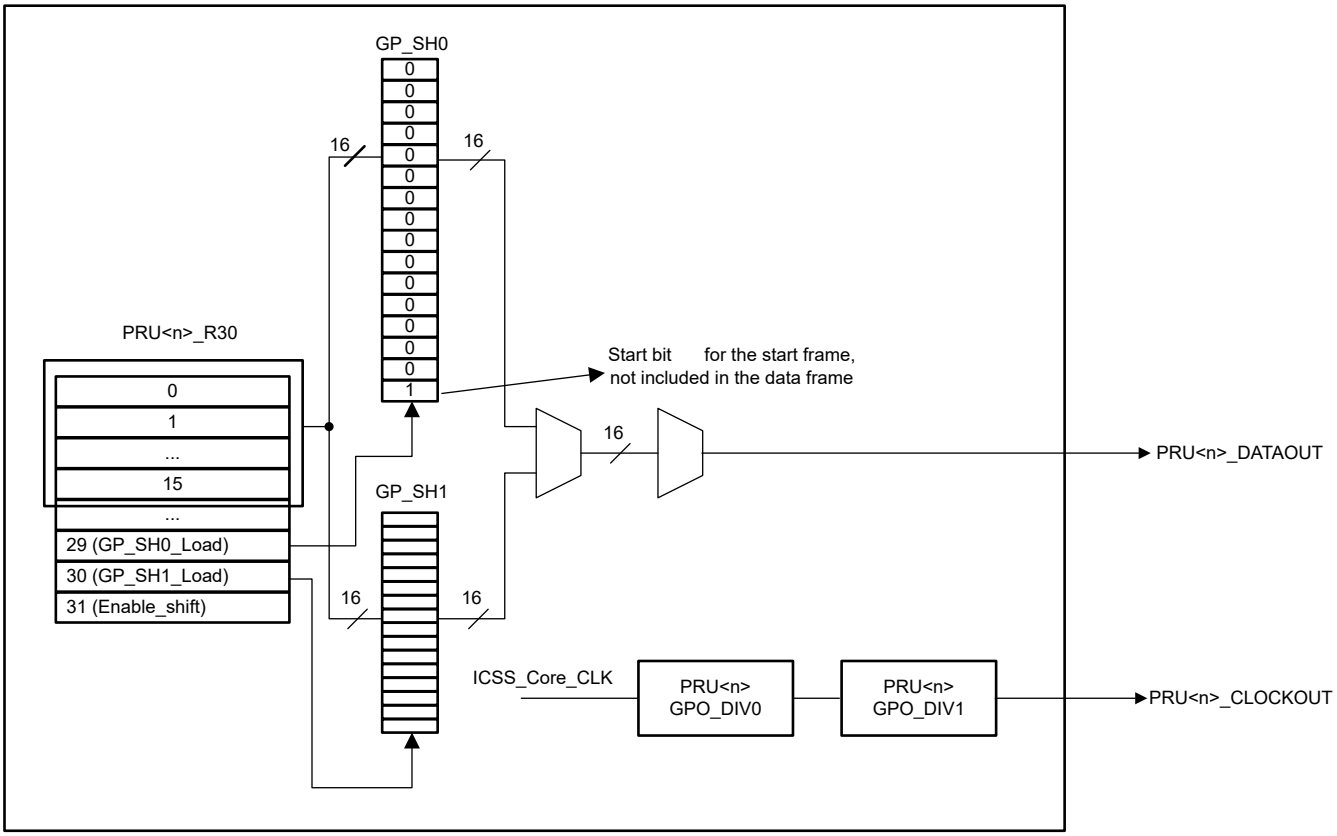

图 2-4. PRU 移出模式方框图

图 2-5 显示了 PRU GPO 移出模式的编程模型。

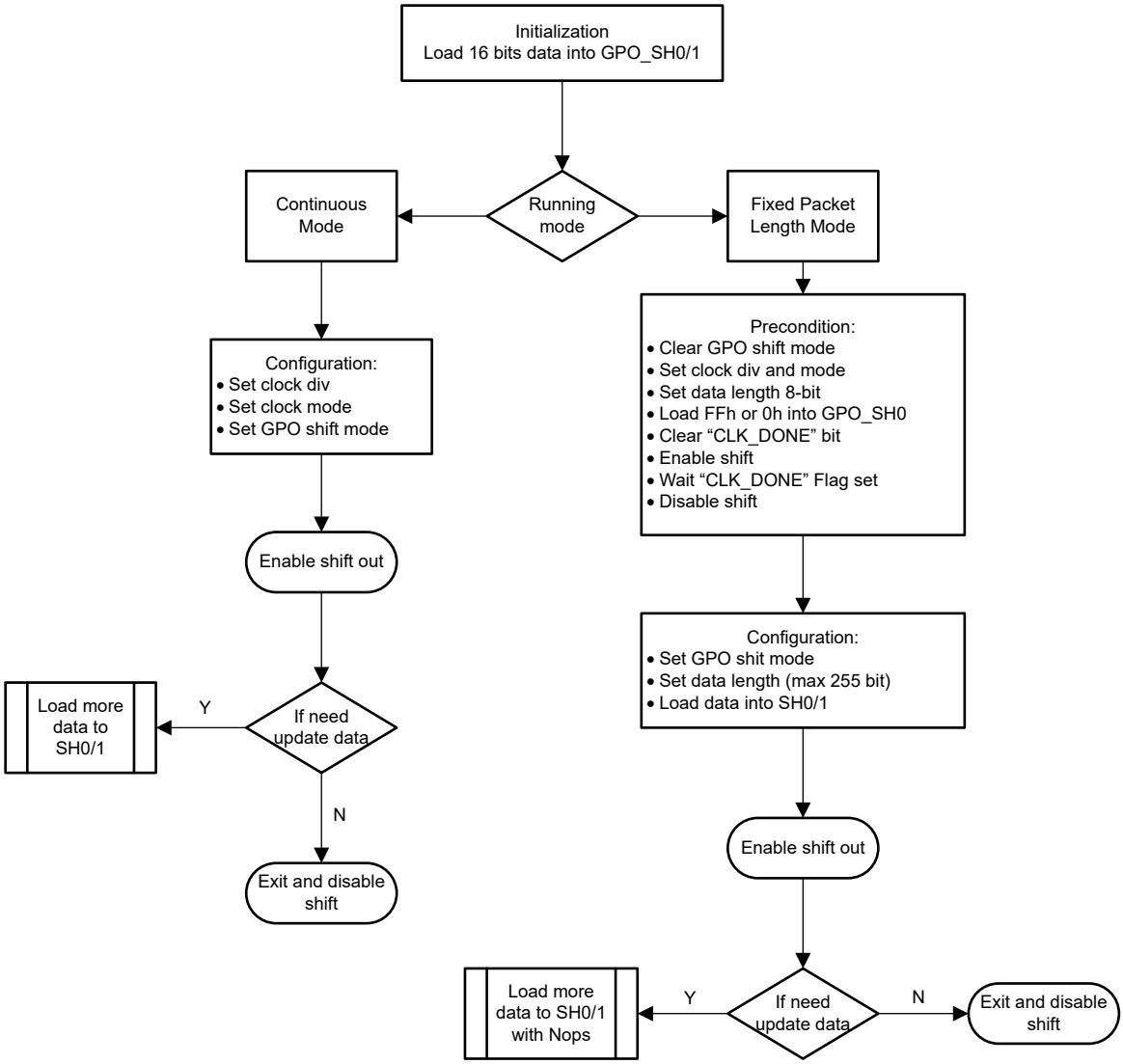

图 2-5. PRU 移出模式的编程模型

可为数据接收配置 GPI 移入模式。在 28 位移入模式下，对通用输入引脚 PRU<n>_DATAIN 进行采样，并根据内部时钟脉冲移入一个 28 位移位寄存器。该寄存器填充最低有效位 (LSB) 顺序（从位 0 到位 27），然后溢出到一个位桶中。28 位寄存器映射到 pru<n>_r31_status [0:27]。移位速率由应用于 PRU 内核时钟的两个级联分频器的有效除数控制。与移出模式类似，选择具有 250MHz 内核时钟的 125MHz 采样时钟。由于移位计数器，CNT_16 在接收到起始位后每 16 个移位时钟样片设置并自清零一次，因此只能将 28 位移位寄存器中的 16 位大小用作接收缓冲器。

图 2-6 显示了 PRU GPI 移入模式的方框图，图 2-7 显示了编程模型。

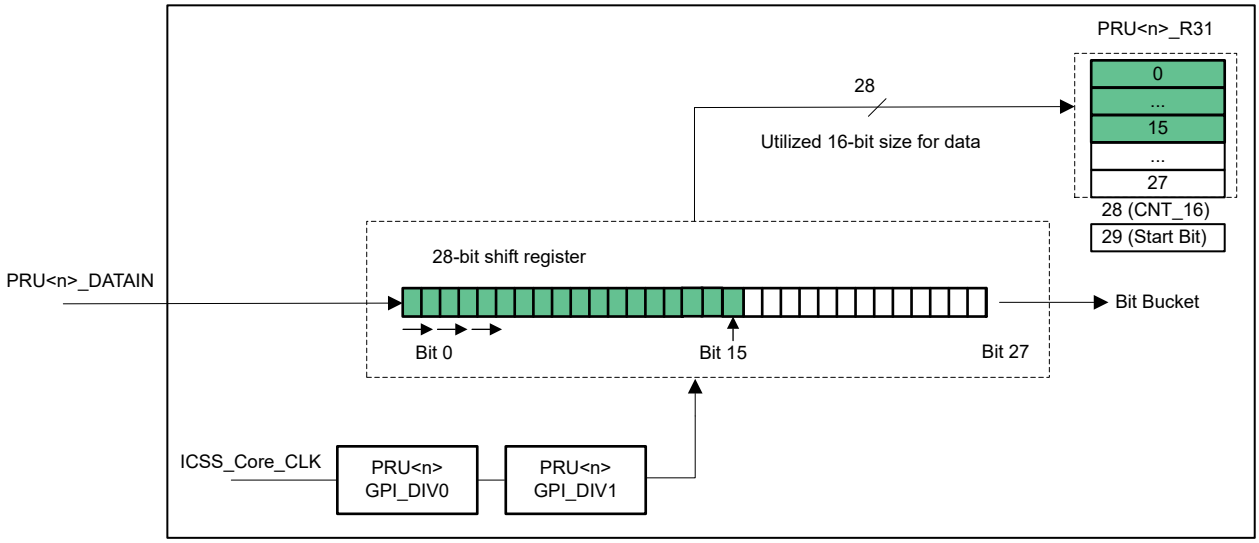

图 2-6. PRU GPI 移入模式方框图

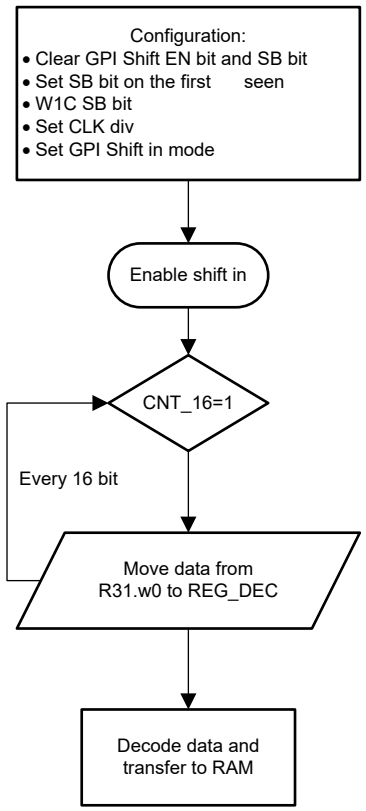

图 2-7. PRU 移入模式的编程模型

为了验证使用 PRU_GPIO 移位模式发送数据的准确性，由 PRU1 内核的移出 GPO 使用前 16 位 0x8000（数据帧在第一个 1 之后）发送 64 位宽编码数据 (0x8000 0012 0034 0056)，并由 PRU0 内核的移入 GPI 接收。然后，数据从 PRU0 寄存器文件移动到从地址 0x00000000 开始的 PRU0 RAM。移出和移入时钟都是 125MHz。图 2-8 显示 PRU1 内核收到正确数据，没有错误。

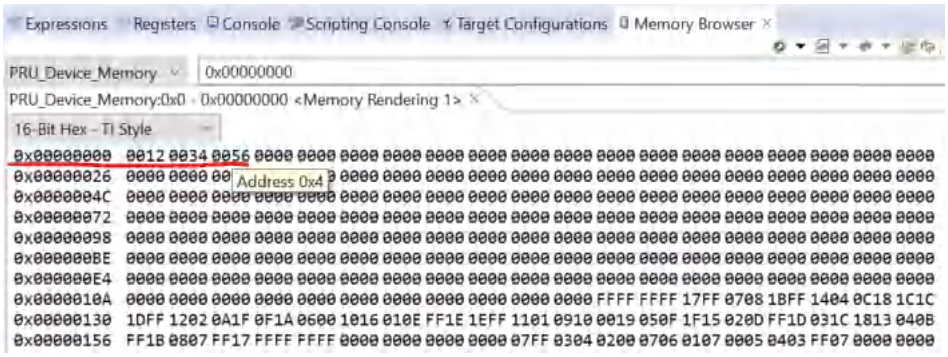

图 2-8. 使用 PRU GPIO 移位模式验证数据传输

2.4 用于通信的三通道外设接口

3 通道外设接口还可以支持类似于 GPIO 移位模式的数据传输。由于本应用手册重点介绍 GPIO 移位模式，因此本节仅简要介绍 3 通道外设接口。可以在 AM62x 处理器器件版本 1.0 德州仪器 (TI) 产品系列技术参考手册中找到有关 3 通道外设模式的详细介绍。

PRU 使用 R30 和 R31 寄存器与外设接口 (I/F) 连接。发送 (TX) 先入先出 (FIFO) 缓冲器是 32 位，并且可以连续模式操作，同时接收具有最大 8 倍过采样的 4 位大小 (RX) FIFO 缓冲器。TX 和 RX 时钟均可由 ICSS_UART_CLK 或 ICSS_CORE_CLK 提供。TX 和 RX 时钟有两个独立的时钟分频器，每个时钟分频器可通过两个级联分频器进行配置。图 2-9 显示了单通道外设 I/F 的方框图，其中其他 2 个通道是相同的。

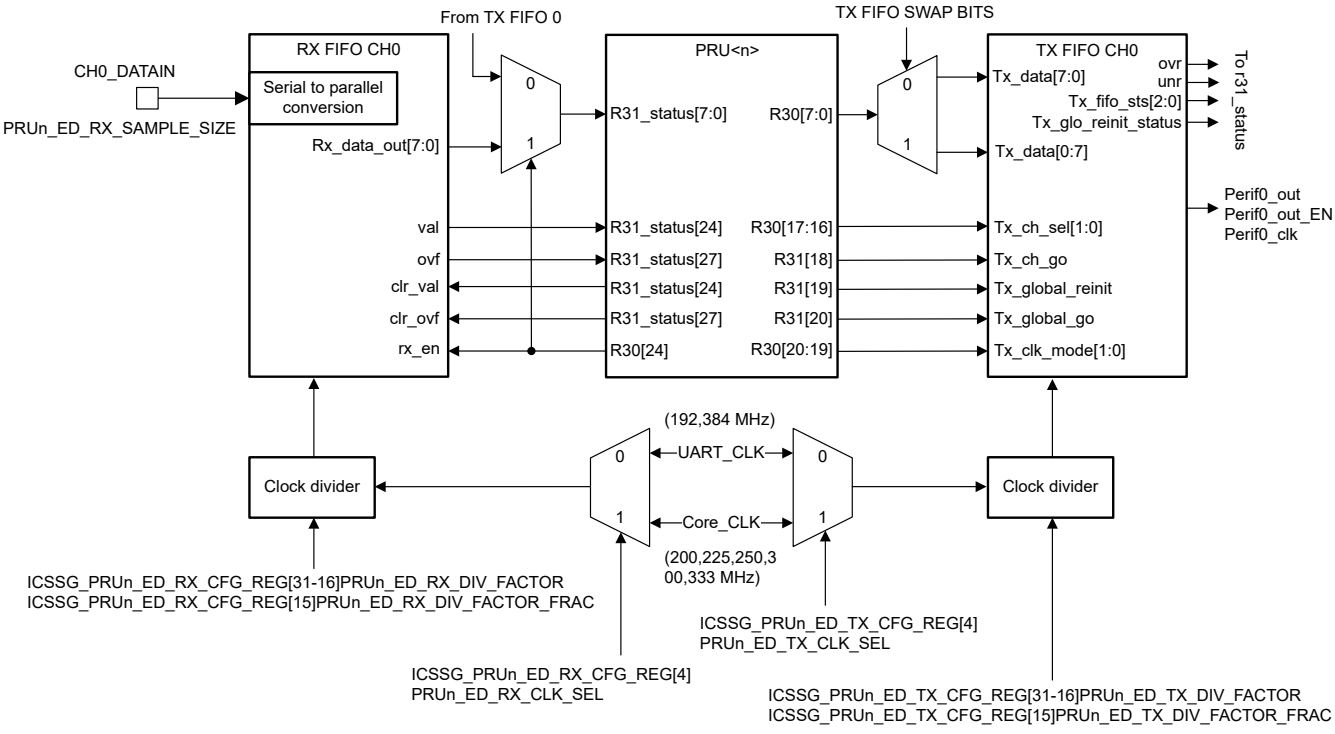

图 2-9. 单通道外设 I/F 方框图

可在 AM62x 处理器器件版本 1.0 德州仪器 (TI) 产品系列技术参考手册中找到三外设模型的基本编程模型。

2.5 LVDS 和 M-LVDS 接口

低电压差分信令 (LVDS) 器件通常符合美国国家标准学会 (ANSI) 标准 TIA/EIA-644。多点低电压差分信令 (M-LVDS) 符合 ANSI TIA/EIA-899。表 2-1 重点介绍了 LVDS 和 M-LVDS 之间的驱动器和接收器的一些重要规格。

表 2-1. LVDS 规格比较

参数	TIA/EIA-644-A (LVDS REV A)	TIA/EIA-899 (M-LVDS)	单位
驱动器特性			
失调电压：Vos（最大值）	1375	2100	mV
失调电压：Vos（最小值）	1125	300	mV
差分输出电压：Vod（最大值）	454 (100Ω)	650 (50Ω)	mV
差分输出电压：Vod（最小值）	247 (100Ω)	480 (50Ω)	mV
失调电压变化：Vospp	150	150	mV
短路电流：Ios	12/24	43	mA
差分电压变化：ΔVod	50	50	mV
失调电压变化：ΔVos	50	50	mV
转换时间：tr/tf（最小值）	260	1000	ps
接收器特性			
接地电位差：Vgpd	±1	±1	V
输入漏电流：输入电流	20	20	μA
差分输入漏电流：Iid	6	4	μA
输入电压范围：Vin	0 至 2.4	-1.4 至 3.8	V
输入阈值：Vith	100	50	mV

驱动器内通信设计需要高速、低功耗和电磁兼容性。因此，LVDS 器件为从点对点到多点数据传输的驱动器内通信设计提供了广泛的解决方案。数据传输速率可达 1500Mbps，电缆长度为 100 米，功耗仅为 1.2mW。差分信号还有助于提高抗噪声能力。德州仪器 (TI) 用于 LVDS 的 DS90x 系列和用于 M-LVDS 的 SN65MLVDSx 系列为设计人员提供了各种单通道和多通道器件。此外，TI 的 ISO7821LLx 系列在驱动器内设计中需要时，特别是在冷热侧控制应用中，通过 LVDS 接口提供了增强的隔离。图 2-10 显示了具有 LVDS 接口和 PRU 的驱动器内通信方框图。

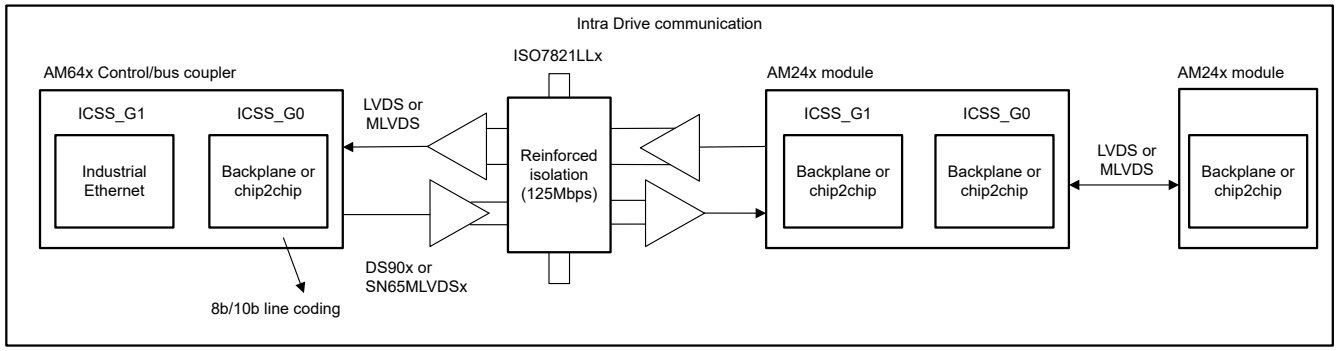

图 2-10. 具有 LVDS 接口和 PRU 的驱动器内通信方框图

为了验证 LVDS 收发器和隔离器的延迟，PRU_GPO 使用 DS90LV049+ISO7821LL 为 LVDS 接口发送 100MHz 时钟信号，使用 SM65MLVDS203+ISO7840 为 M-LVDS 接口发送 100MHz 时钟信号。图 2-11 和图 2-12 显示 LVDS 接口的延迟为 18ns，M-LVDS 接口的延迟为 22ns。

图 2-11. DS90LV049 接口和 ISO7821LL LVDS 接口的 18ns 延迟

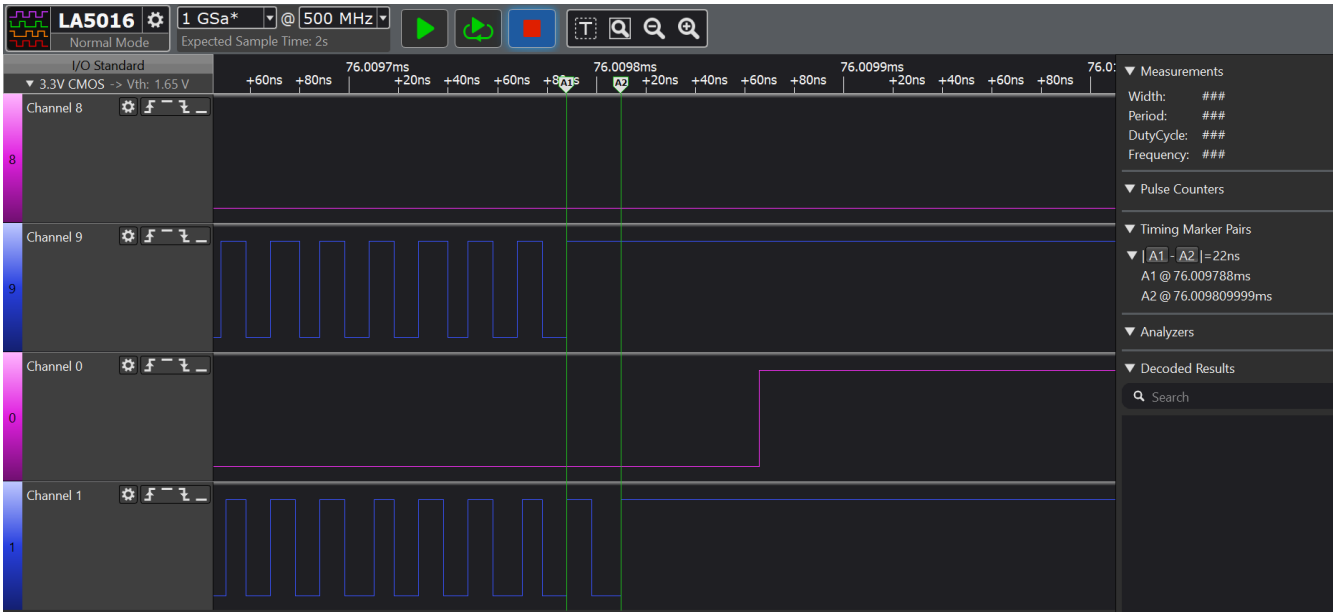

图 2-12. SN65MLLVDS203 接口和 ISO7840 M-LVDS 接口的 22ns 延迟

3 具有 CRC 模块和开销优化的系统解决方案

3.1 PRU CRC16/32 模块

通信帧封装中需要进行循环冗余校验 (CRC)，以确保数据传输过程中的数据准确性。每个 PRU 内核都有一个指定的外设 CRC16/32 模块，为通信系统提供错误检测功能。CRC16/32 模块支持具有不同多项式的 CRC32、CRC16 和 CRC16-CCITT 功能。该模块还通过使用 PRU 宽边接口与 PRU 内部寄存器 R25 至 R29 连接。R29 映射到 CRC_DATA 寄存器，支持最大 32 位数据宽度，并根据配置以 16 位或 32 位读回检查数据。可以生成具有 16 位数据字的 16 位 CRC 数据，并将其放入每个帧中。图 3-1 显示了 PRU CRC16/32 模块的方框图，以下代码显示了用于 16 位数据的 CRC16 函数：

```
zero &r25, 4                ; config CRC type
xout CRC_XID, &25, 4        ; enable CRC module, CRC_XID = 0x1
mov  r29, r20               ; load CRC data, ASCII equivalent for "21"
xout CRC_XID, &29.w0, 2     ; push CRC data to CRC16 module
nop
xin  CRC_XID, &r28, 4       ; load the accumulated CRC result into PRU
```

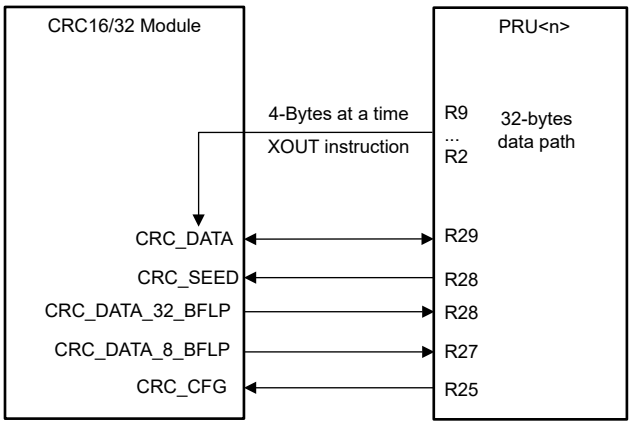

图 3-1. PRU CRC16/32 模块方框图

CRC 输入数据必须交换字节，因为 PRU 基于最低有效字节 (LSByte) 优先架构。例如，必须将 0x3412 作为输入数据给出，才能对 0x1234 执行 CRC16。图 3-2 显示了 CRC16/32 模块的验证结果。PRU0 从 PRU1 GPO 每 16 位接收一次数据，并存储在 PRU DRAM 中从 0x00000000 开始的地址中。

基地址存储 PRU1 GPO 发送的原始数据 (0x12)。接收到的数据被发送到 PRU0 CRC 模块，并将累积的 CRC 结果加载到 PRU0 R28 寄存器和偏移量为 0x24 的存储器地址中。偏移量为 0x02 的存储器地址存储来自 PRU1 的数据 0x12 的 CRC16 结果 (0xf30c)。地址偏移 0x04 存储 PRU1 GPO 发送的原始数据 (0x34)。接收到的数据被发送到 PRU0 CRC 模块，并将累积的 CRC 结果加载到 PRU0 R28 寄存器和偏移量为 0x26 的存储器地址中。偏移量为 0x06 的存储器地址存储来自 PRU1 的数据 0x34 的 CRC16 结果 (0x36c7)。图 3-2 表明，从存储器值来看，原始数据和 CRC 数据都是正确的。

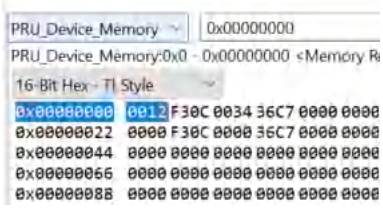

图 3-2. PRU CRC16 模块验证

3.2 编码和解码开销优化

正如前面章节中描述的 8b-10b 编码方法所述，8 位数据生成 10 位编码数据，并存储在 REG_ENC 寄存器的 2 个字节中，这在字节 2 中留下了 6 个未使用的位。发送缓冲区为 16 位宽度，以便数据可以组合并连续发送以减少开销。图 3-3 显示了用于发送编码数据的优化方法：

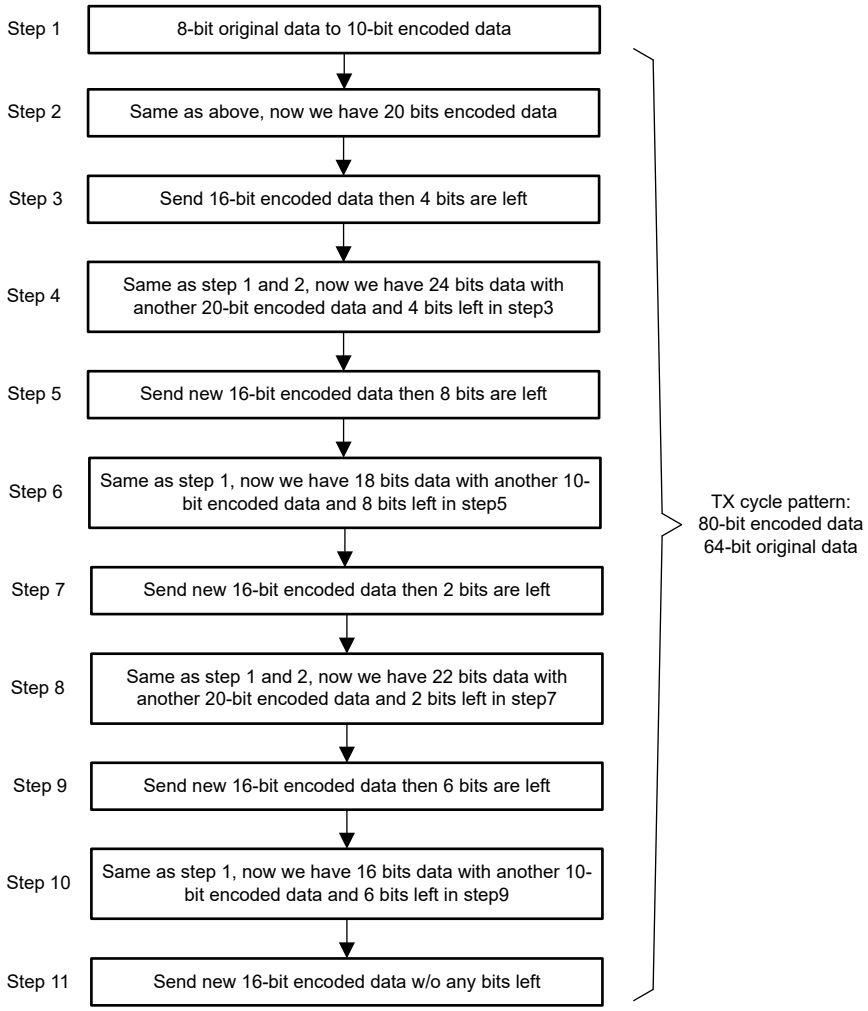

图 3-3. 用于发送编码数据的优化方法

一种发送周期模式是具有 8 次编码处理的 80 位编码数据宽度（64 位原始数据宽度）。

图 3-4 显示了接收解码数据的优化过程，这个过程与发送编码数据的方法类似：

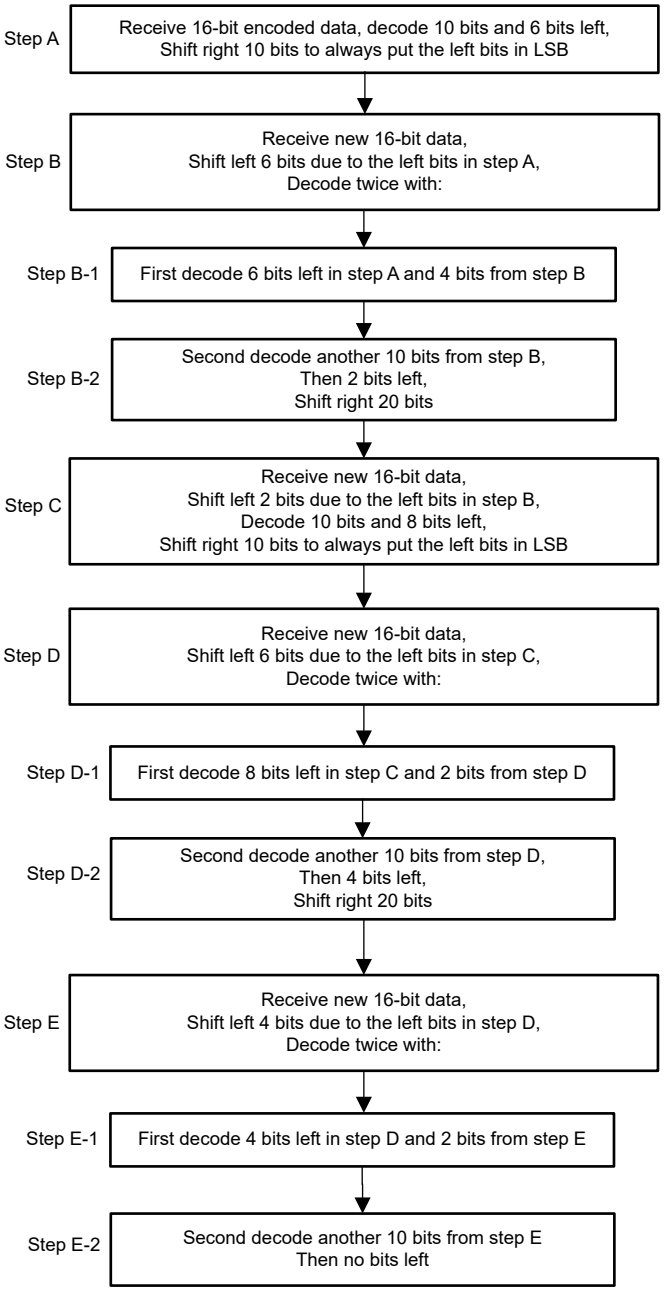

图 3-4. 接收解码数据的优化方法

一个接收周期模式是具有 8 次解码处理的 80 位解码数据宽度。

原始帧模式可以是具有 16 位前导码、32 位数据字和 16 位 CRC 数据的 64 位宽度。为了提高效率，RTU_PRU0 和 RTU_PRU1 辅助内核可用于并行编码、解码和 CRC16 工作负载。图 3-5 显示了一个同时包含 PRU 和 RTU 内核的方框图。

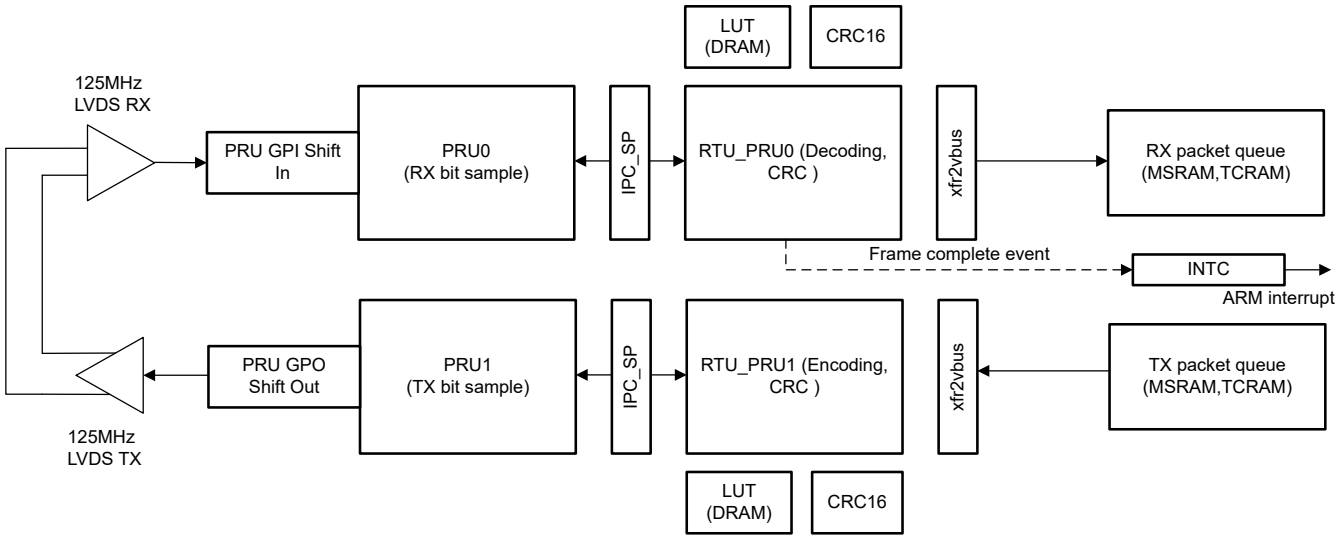

图 3-5. 具有 PRU 和 RTU 内核的系统方框图

4 验证

节 2.3 和节 2.4 说明了如何验证数据传输准确性和 LVDS 接口延迟。本节介绍如何测试 PRU GPIO 侧的抖动，以及发送和接收的处理时间。图 4-1 显示了 125MHz 时的发送数据速率。图 4-2 和图 4-3 显示了在上升沿和下降沿期间 GPO 侧的抖动仅为 60ps。

图 4-1. 发送数据速率

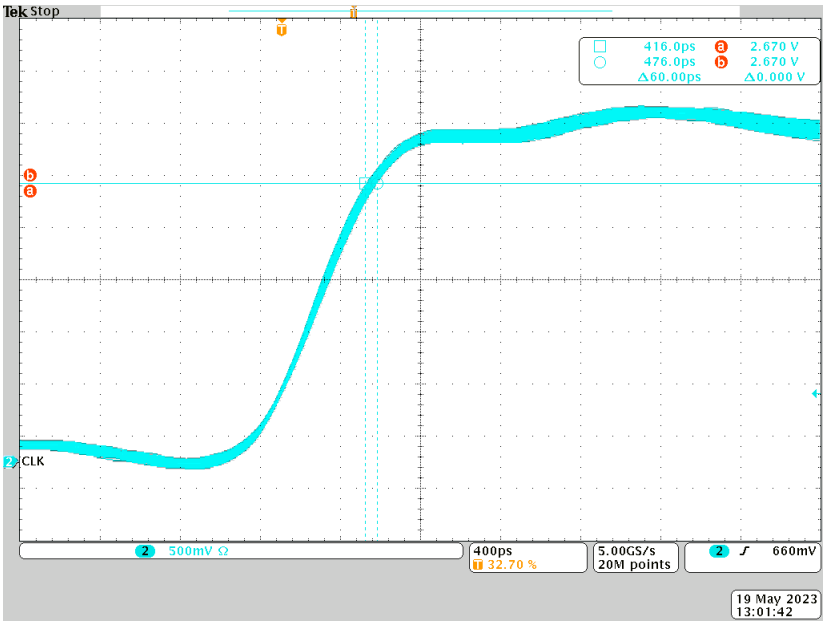

图 4-2. 上升沿上的 PRU GPO 抖动

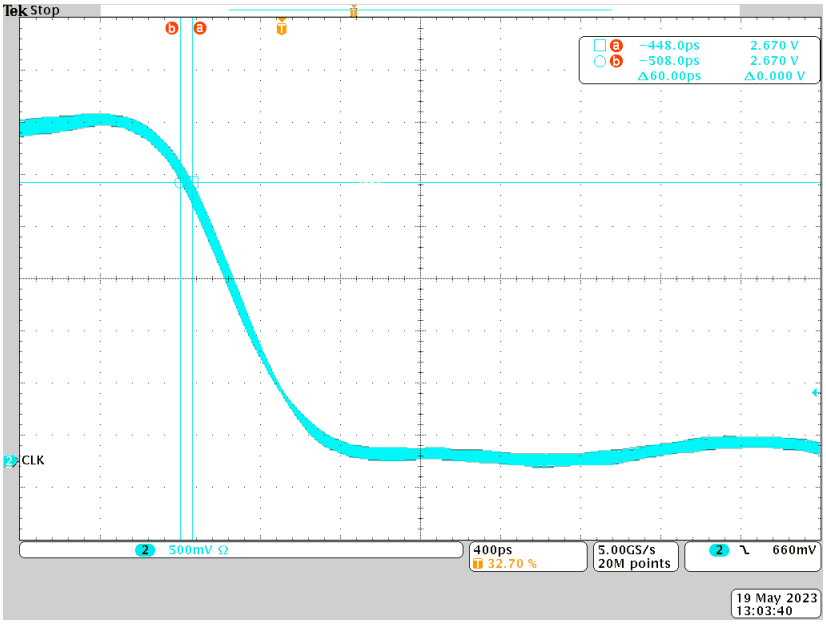

图 4-3. 下降沿上的 PRU GPO 抖动

发送数据的处理时间包括对数据进行编码、数据 CRC16 校验和 IPC 暂存区存储器共享（可选）。在最大 250MHz PRU 内核时钟下，对 8 位数据进行编码的处理时间为 48ns，处理 32 位 CRC16 校验的时间为 20ns。64 位宽度的发送周期模式需要 404ns，其中编码时间为 384ns，CRC 时间为 20ns。接收处理时间包括对数据进行解码、数据 CRC16 校验和 IPC 暂存区存储器共享（可选）。在最大 250MHz PRU 内核时钟下，解码 10 位数据的处理时间为 60ns，处理 32 位 CRC16 校验的时间为 20ns。64 位宽度的接收周期模式需要 500ns，其中编码时间为 480ns，CRC 时间为 20ns。图 4-4 至图 4-6 显示了使用 PRU_GPO 切换处理时间的测试结果。由于 PRU_GPIO 一次只能配置为一种模式，因此将测试条件与通信周期分开，仅测试相关代码指令的时间。

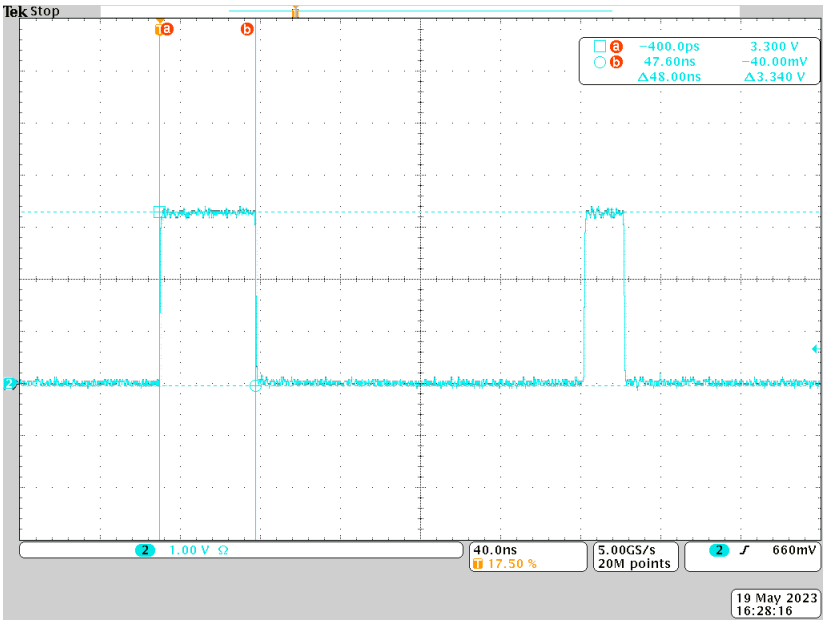

图 4-4. 8 位数据编码的处理时间

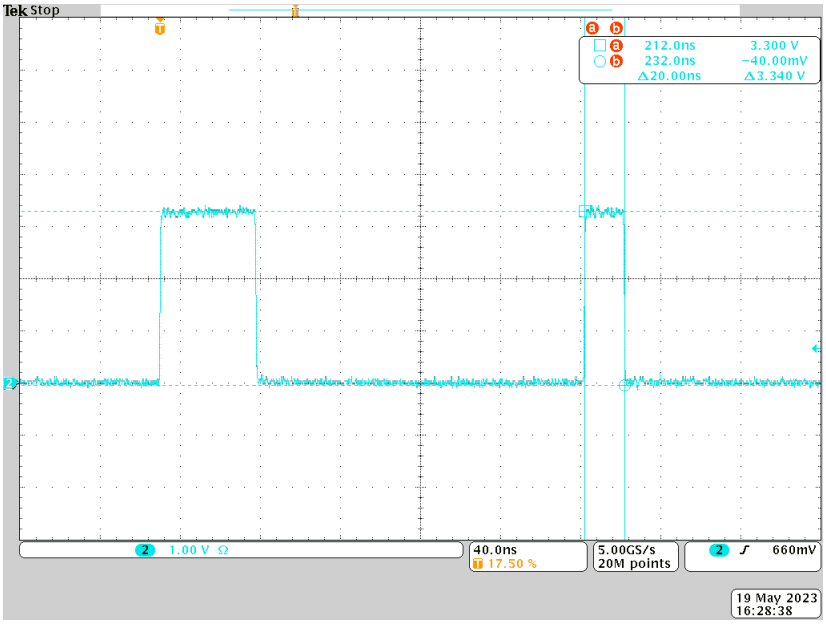

图 4-5. CRC16 的处理时间

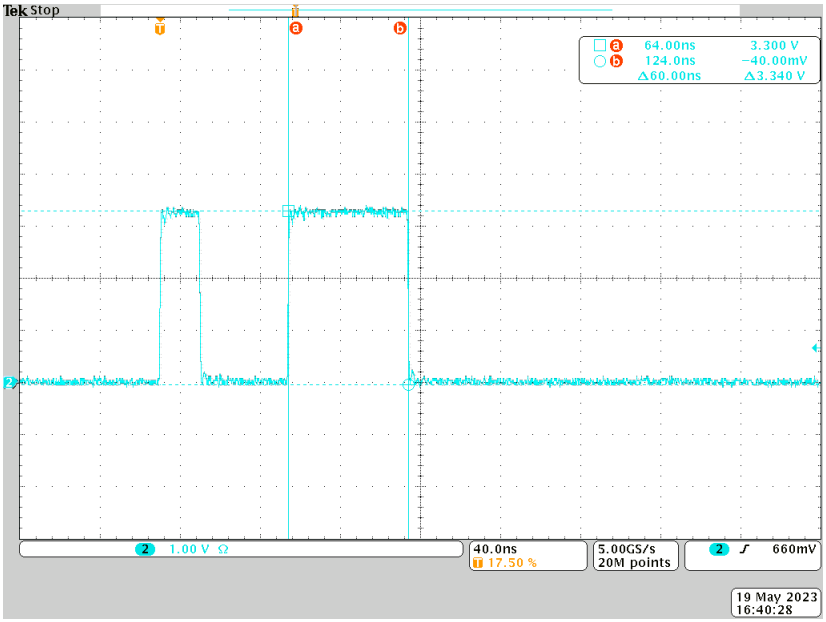

图 4-6. 10 位数据解码的处理时间

在 RTU_PRU0 和 RTU_PRU1 辅助内核的支持下，可以并行实现数据传输和处理。IPC 模块可用于通过 XFR 指令在 PRU 和 RTU 之间交换数据。暂存区（寄存器 R2:R9）为 32 字节宽，将一个片内的 PRU 和 RTU_PRU 内核连接在一起。XFR 指令定义操作的开始、大小、方向和器件 ID。图 4-7 显示了 PRU 和 IPC 暂存区的集成。

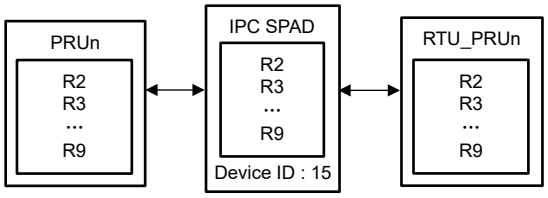

图 4-7. PRU 和 IPC 暂存区的集成

XFR2VBUS 硬件加速器可以支持在磁阻随机存取存储器 (MRAM) 或紧密耦合随机存取存储器 (TCRAM) 之间来回移动数据。XFR2VBUS TX 和 RX 缓冲器均为 64 字节深，可通过单个寄存器传入 (XIN) 或寄存器传出 (XOUT) 指令传输 64 字节数据。以下代码示例显示如何使用 XFR2VBUS 小工具将数据移入和移出 TCRAM。

```
; Read wait
wait_till_read_busy_0?:
  xin XFR2VBUSP_RD0_XID, &r18, 1
  qbbs wait_till_read_busy_0?, r18, 3    ; R18.3 :RD_MST_REQ Wait until Last Data has been latched

; TCM to XFR2VBUS RX buffer
  ldi   r18, 6                           ; 64 bytes
  ldi32 r19, CSL_R5FSS0_CORE0_ATCM_BASE  ; TCM Address
  ldi   r20, 0
  xout  XFR2VBUSP_RD0_XID, &r18, 10      ; transfer address
  xin   XFR2VBUSP_RD0_XID, &r2, 65       ; transfer data

; Write wait
wait_till_write_done_0?:
  xin   XFR2VBUSP_WR0_XID, &r20, 1
  qbbs  wait_till_write_done_0?, r20, 0  ; R20.0 : WR_BUSY Wait until Idle

; XFR2VBUS TX buffer to TCM
  ldi32 r18, CSL_R5FSS0_CORE0_ATCM_BASE  ; TCM Address
  ldi   r19, 0
  xout  XFR2VBUSP_RD0_XID, &r18, 10      ; transfer address
  xin   XFR2VBUSP_RD0_XID, &r2, 65       ; transfer data
```

PRUn 可在 RTU_PRUn 对前 16 位原始数据进行编码之后开始发送数据。类似地，接收路径中的 RTU_PRUn 可以在 PRUn 接收到前 20 位编码数据之后开始解码数据。64 位宽周期模式的发送处理时间可以从 404ns 减少到 96ns（16 位数据编码），64 位宽周期模式的接收处理时间可以从 500ns 减少到 120ns（20 位数据解码）。表 4-1 总结了在有和没有 RTU_PRUn 内核支持情况下的处理时间。

表 4-1. 使用和不使用 RTU_PRUn 内核的处理时间

PRU 内核	采样时钟 (MHz)	数据长度（位）	TX 模块数据处理时间 (ns)	数据传输时间 (ns)	RX 模块数据处理时间 (ns)	总时间 (ns)
PRU	125	64	404	640(1)	500	1544
PRU + RTU	125	64	96	640(1)	120	856

(1) 用于数据传输的 80 位编码数据宽度

5 总结

本应用手册提供了一种使用 PRU 实现数据速率为 100Mbps 的 8b-10b 线路编码，并使用 LVDS 接口发送数据的方法。PRU 中使用 LVDS 接口的线路编码可以灵活地定义驱动器内通信的自定义协议，从而实现高速数据速率、低抖动和低延迟。

6 参考文献

1. 德州仪器 (TI)，*AM243x Sitara™ 微处理器* 数据表
2. 德州仪器 (TI)，*AM64x/AM243x 处理器器件* 技术参考手册
3. 德州仪器 (TI)，*PRU 汇编指令用户指南*
4. 德州仪器 (TI)，*M-LVDS (TIA/EIA-899) 简介* 应用手册
5. IBM Research，*适用于高速应用的 8B/10B 编码和解码* 技术白皮书

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