Application Report

适用于工业应用的使用 MDIO 的以太网 PHY 配置

Garrett Ding, Pratheesh Gangadhar TK, David Zaucha

摘要

作为链路层器件介质访问控制器 (MAC) 和物理介质（如铜缆）之间的一种桥接器，以太网物理层器件 (PHY) 集成了在标准双绞线电缆上发送和接收数据所需的所有物理层功能。使用管理数据输入/输出 (MDIO) 进行正确的 PHY 配置是原型阶段的一项基本操作，也是满足工业以太网应用（如 EtherCAT®）中较低确定性延迟和较快链路检测要求的关键。本应用报告通过剖析 AMIC110 工业通信引擎 (ICE) 上 EtherCAT 的 PHY DP83822 配置，为针对工业应用使用 TI Sitara™ 器件中可编程实时单元工业通信子系统 (PRU-ICSS) 内 MDIO 模块的以太网 PHY 配置提供了指导。本应用报告旨在通过 PHY [1] [2] [3] 的迁移和故障排除指南，加快定制电路板上工业以太网应用的开发。

内容

插图清单

表格清单

商标

1 PHY 选择和连接

许多工业以太网应用要求 PHY 符合 IEEE 802.3 100BaseTX 或 100BaseFX，支持 100-Mbps 全双工链路，使用自动协商，并支持 100BaseTX 模式下的 MDI/MDI-X 自动交叉功能。此外，链路中断反应时间也是确保冗余操作链路灵敏度快于 15µs 的一项关键基准。尽管大多数 PHY 支持多个具有不同引脚数和数据速率的媒体独立接口 (MII) 以便与 MAC 进行通信，但推荐使用 MII，因为它减少了 RMII 中由 TX FIFO 造成的额外转发延迟。图 1-1 所示为一个具有 MAC 和物理介质的典型以太网 PHY 连接。

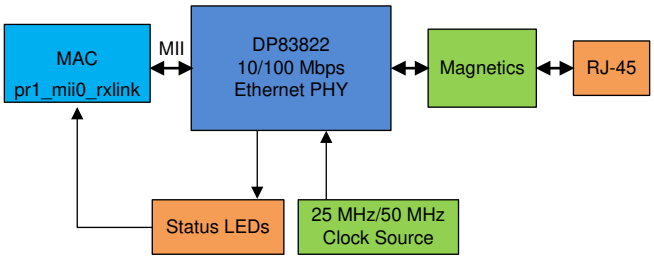

图 1-1. 具有 MAC 和物理介质的典型以太网 PHY 连接

AMIC110 工业通信引擎 (ICE) 是一款面向工业通信且尤其是工业以太网的开发平台。AMIC110 ICE 是 Sitara AMIC110 片上系统 (SoC)，搭载 ARM® Cortex™-A8 处理器和 PRU-ICSS，无需 ASIC 或 FPGA 即可实现对实时工业协议的集成。AMIC110 ICE 中选择了省电的 DP83822 器件（请参阅 图 1-2）。DP83822 不仅符合 IEEE 802.3u 的要求，还针对串话和外来噪声留有较高裕度。

DP83822 器件是一款全功能单端口物理层收发器，适用于 100BASE-Te、100BASE-TX 和 100BASE-FX 信令，其信号可分为以下几类：

- MAC 接口
- 串行管理接口
- 时钟接口
- GPIO 和 LED 接口
- 媒体独立接口
- 电源和接地引脚
- 其他引脚

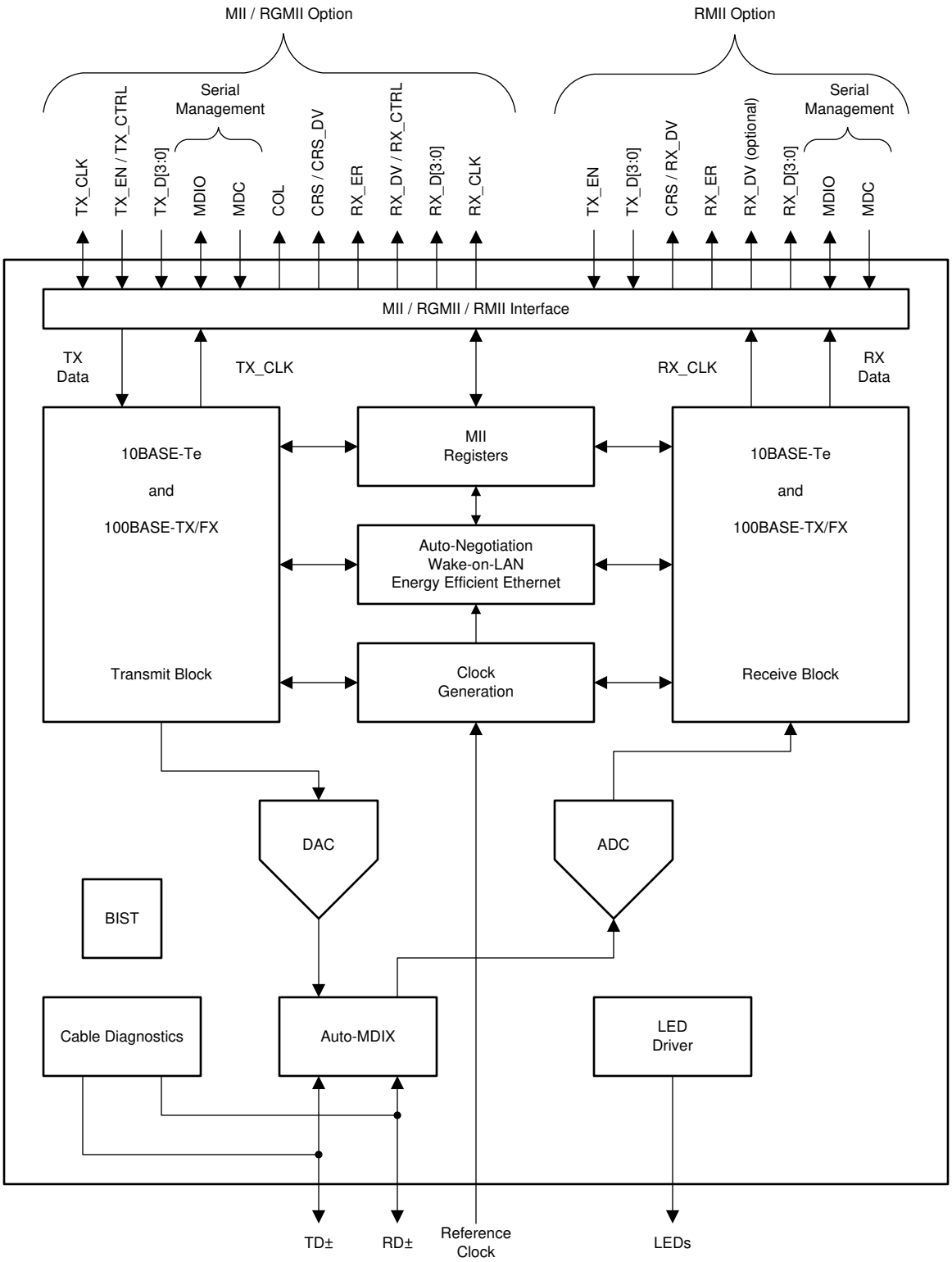

图 1-2. DP83822 功能方框图

Beckhoff 提供了一份全面的 PHY 选择指南，为 PHY 选择和配置[1]提供帮助。本文档列出了 TI 出品的各种 PHY，如 DP836x、DP838x、TLK10x 和 TLK11x。

2 PHY 复位和地址

在上电或硬件复位时进行 PHY 自举配置，将器件置于所需的工作模式下。对 DP83822 PHY 中的 RESET 引脚施加一个持续时间至少为 10µs (T1) 的低脉冲，以实现硬件复位。此脉冲将复位器件，以便将所有寄存器都重新初始化为默认值，并将硬件配置值关联到器件中（请参阅 图 2-1）。

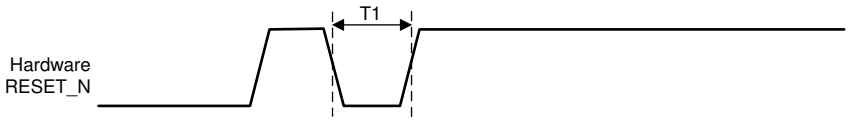

图 2-1. PHY 复位信号

自举引脚在解除置位后具有替代功能，因此，为了最大限度地减少自举引脚信号采样期间产生的干扰，PRU-ICSS MII 复用应发生在 AMIC110 ICE 上释放 PHY RESET 之后。如果未配置这些自举引脚，那么一旦释放复位，它们就会变成 GPIO 和高阻态。TI EtherCAT 中的硬件复位序列实现如下所示：

```
Board_init(BOARD_INIT_MODULE_CLOCK);
Board_phyReset(2);
/* 在 PHY 复位后复用 PRU MII，以防 PRU 驱动
* 引脚上的信号并干扰 PHY 自举配置
*/
#ifdef ENABLE_UART_PRINT
Board_init(BOARD_INIT_UART_STDIO|BOARD_INIT_ICSS_PINMUX);
#else
Board_init(BOARD_INIT_ICSS_PINMUX);
```

Board_init( ) API 来自 TI Processor SDK 电路板库。PRU-ICSS EtherCAT 是以 TI 的 Processor SDK 为基础构建的一款统一软件平台，适用于 TI 嵌入式处理器，其设置简单，提供开箱即用的基准测试和演示。TI PRU-ICSS 工业软件包支持 EtherCAT、PROFINET、PROFIBUS、EtherNet/IP、HSR/PRP 等。

Board_phyReset() 的实现方式如下：

```
/* 驱动 Phy1（和 Phy2）复位到高电平；两个 PHY 复位都来自一个 GPIO */
    GPIOModuleEnable(PhyResetInfo[1].baseAddr);
    GPIODirModeSet(PhyResetInfo[1].baseAddr, PhyResetInfo[1].pin, GPIO_DIR_OUTPUT);
    GPIOPinWrite(PhyResetInfo[1].baseAddr, PhyResetInfo[1].pin, GPIO_PIN_HIGH);
    delay_us(20);
    GPIOPinWrite(PhyResetInfo[1].baseAddr, PhyResetInfo[1].pin, GPIO_PIN_LOW);
/* T1 - RESET 脉冲宽度，最短时间为 10us，请参阅 DP83822 数据表 */
    delay_us(20);
    GPIOPinWrite(PhyResetInfo[1].baseAddr, PhyResetInfo[1].pin, GPIO_PIN_HIGH);
```

其中 PhyResetInfo[1] 配置了用于复位的 GPIO 引脚。

```
/* phy1 复位 - 驱动高电平 */
    PhyResetInfo[1].pin = 13;
    PhyResetInfo[1].baseAddr = SOC_GPIO_1_REGS;
```

NOTE

也可以通过 0x001F PHY 复位控制寄存器 (PHYRCR) 来复位 PHY，具体请参阅 表 2-1。

表 2-1. PHY 复位控制寄存器 (PHYRCR)

位	字段	类型	复位	说明
15	软件复位	RW。SC	0	软件复位： • 1 = 复位 PHY • 0 = 正常运行 此位可自行清除，具有与硬件复位引脚相同的效果。

表 2-1. PHY 复位控制寄存器 (PHYRCR) (continued)

位	字段	类型	复位	说明
14	数字重启	RW、SC	0	数字重启： • 1 = 复位 PHY • 0 = 正常运行 此位可自行清除，并复位除寄存器以外的所有 PHY 数字电路。
13:0	保留	RW	0	保留

在 DP83822 器件中，PHY 地址引脚 PHY_AD[4:1] 与 RX_D[3:0] 进行多路复用并被下拉。PHY_AD[0]（地址的 LSB）与引脚 29 上的 COL 进行多路复用并被上拉。如果不存在外部上拉或下拉，则默认 PHY 地址为 0x01。DP83822 器件可通过自举配置功能，配置为 32 个可能的 PHY 地址中的任一个。该 PHY 地址在器件上电或硬件复位之时锁存到器件中，并在软件中进行硬编码。

```
#define AM335X_ICSS1_PORT1_PHY_ADDR 1
#ifndef iceAMIC11x
#define AM335X_ICSS1_PORT2_PHY_ADDR 3
#else
#define AM335X_ICSS1_PORT2_PHY_ADDR 13
#endif
```

PHY 地址以如下定义的偏移量写入共享数据 RAM 中，以通知 PRU 固件。

```
#define ESC_ADDR_TI_PORT0_PHYADDR   0xE08
#define ESC_ADDR_TI_PORT1_PHYADDR   0xE09
    //通过供应商特定的寄存器向固件指示 PHY 地址
bsp_write_byte(pruIcssHandle, pmdio_params->addr0, ESC_ADDR_TI_PORT0_PHYADDR);
bsp_write_byte(pruIcssHandle, pmdio_params->addr1, ESC_ADDR_TI_PORT1_PHYADDR);
```

对于 AMIC110，共享数据 RAM 地址为 0x4A31_0000。

当 MDIO 无法使用主机 API（例如 Board_getPhyIdentifyStat()）访问 PHY_ID1_REG（寄存器 0x02）时，这通常意味着 PHY 未正确复位或 PHY 地址未正确配置。由于使用不同的 GPIO，TI 和客户电路板之间的复位方法可能有所不同。

3 PHY 速度、双工等

PHY 复位后，可以针对所需工作模式使用 MDIO 对其进行配置。AMIC110 中 PRU-ICSS 内的 MDIO 采用 802.3 串行管理接口 (SMI)，可通过使用共享双线总线同时询问和控制两个以太网 PHY。DP83822 器件中的 SMI 符合 IEEE 802.3 第 22 条和第 45 条的要求，可访问其内部寄存器空间，包括标准寄存器组 0 至 31、采用寄存器控制寄存器（REGCR，地址 0x000D）的扩展寄存器组以及数据寄存器（ADDAR，地址 0x000E），以获取状态信息和配置。

以太网自动协商功能提供了一种机制，可通过突发脉冲在链路段两端之间交换配置信息，如速度和双工。自动协商功能可确保根据链路伙伴和本地器件的广播能力选择最高性能协议。可在硬件中使用 AN_EN 自举启用或禁用自动协商功能，也可使用基本模式控制寄存器（BMCR，地址 0x0000）中的位 [12] 并通过寄存器配置来启用或禁用该功能。

在 PRU-ICSS EtherCAT 软件中，使用 MDIO 的 PHY 初始化的调用流程如下：

```
Phy_reset( )
|
Task_create(task1,&taskParams, NULL);
|
BIOS_start();
```

其中 MDIO 和 PHY 初始化是在 task1 中执行：

```
task1() -> Hw_init( ) --> bsp_init( ) -> bsp_pruss_mdio_init( ) -> bsp_ethphy_init( )
```

Hw_init( ) – EtherCAT 从接口初始化

bsp_init() – PRU 初始化

使用 CSL 函数 CSL_MDIO_init() 通过 bsp_pruss_mdio_init() 对 MDIO 进行初始化，其中输入和输出时钟用作参数。

```
/** ============================================================================
 *   @n@b CSL_MDIO_init
 *
 * \brief   此 API 初始化 MDIO 外设。这可启用 MDIO 状态
 *          机，使用标准前导码并设置时钟分频器值。
 *
 * \param   baseAddr       MDIO 模块的基地址。
 * \param   mdioInputFreq  MDIO 模块的时钟输入。
 * \param   mdioOutputFreq MDIO 总线上所需的时钟输出。
 * ============================================================================
 */
static inline void CSL_MDIO_init(uint32_t baseAddr,
              uint32_t mdioInputFreq,
              uint32_t mdioOutputFreq);
```

MDIO PRU 固件与 PHY 通信时需要对 PRU-ICSS 进行初始化，因此必须在 PRU-ICSS 域上电之后执行 PRU-ICSS 的初始化，然后在两个 PRU 上加载并执行 PRU 固件。

MDIO 完成初始化后，可以通过 CSL 函数 CSL_MDIO_phyRegRead() 和 CSL_MDIO_phyRegWrite() 访问 PHY，以配置以下内容：

- 连接速度、双工和自动协商
- 自动 MDIX，可确定是使用直通电缆还是交叉电缆来与链路伙伴相连。
- 扩展全双工模式。在扩展全双工模式下，当 PHY 设置为自动协商或 Force 100Base-TX，且链路伙伴在 Force 100Base-TX 模式下运行时，链路始终为全双工模式。禁用后，决定是在全双工还是半双工模式下工作时要遵循 IEEE 规范 – 半双工。
- 在奇半字节边界中检测到传输错误，该错误使 TX_EN 扩展一个额外的 TX_CLK 周期，并且其行为就好像在该额外周期中置位了 TX_ER 一样。

- 其他功能（如奇半字节插入和快速链路断开检测）如下所述：

```
/** ============================================================================
 *   @n@b CSL_MDIO_phyRegRead
 *
 * \param   baseAddr MDIO 模块的基地址。
 * \param   phyAddr  PHY 地址。
 * \param   regNum   要读取的寄存器编号。
 * \param   pData    写入读取值的指针。
 *
 * \retval  TRUE     读取成功。
 * \retval  FALSE    未正确确认读取。
 * ============================================================================
 */
static inline Uint32  CSL_MDIO_phyRegRead(uint32_t baseAddr,
                    Uint32 phyAddr,
                    Uint32 regNum,
                    Uint16 *pData)
/** ============================================================================
 *   @n@b CSL_MDIO_phyRegWrite
 *
 * \brief   此 API 使用 MDIO 写入一个 PHY 寄存器。
 *
 * \param   baseAddr      MDIO 模块的基地址。
 * \param   phyAddr       PHY 地址。
 * \param   regNum        要写入的寄存器编号。
 * \param   wrVal         要写入的值。
 * ============================================================================
 */
static inline void CSL_MDIO_phyRegWrite(uint32_t baseAddr,
                     uint32_t phyAddr,
                     uint32_t regNum,
                     uint16_t wrVal);
```

大多数 MDIO PHY 配置函数都集成在 Processor SDK 的电路板库中。

例如，若要启用 0x0019 PHY 控制寄存器 (PHYCR) 中的自动 MDIX 位，请参阅以下内容：

```
Board_enablePhyAutoMDIX((((PRUICSS_HwAttrs *)(pruIcssHandle->hwAttrs))->prussMiiMdioRegBase),
pmdio_params->addr0);
```

若要启用 0x000A 控制寄存器 2 (CR2) 中的扩展全双工模式位，请参阅以下内容：

```
Board_phyExtFDEnable((((PRUICSS_HwAttrs *)(pruIcssHandle->hwAttrs))->prussMiiMdioRegBase),
phy0addr);
```

要启用 0x000A 控制寄存器 2 (CR2) 中的奇半字节插入位，请参阅以下内容：

```
Board_phyODDNibbleDetEnable((((PRUICSS_HwAttrs *)(pruIcssHandle->hwAttrs))->prussMiiMdioRegBase),
phy0addr);
```

用户也可以直接调用 CSL 函数来管理 PHY，例如关闭 RMII 模式并选择 MII 模式，具体请参阅以下内容：

```
phyregval = 0;
CSL_MDIO_phyRegWrite((((PRUICSS_HwAttrs *)(pruIcssHandle->hwAttrs))->prussMiiMdioRegBase),
pmdio_params->addr0, TLKPHY_RCSR_REG, phyregval);
```

使用 Code Composer Studio™ 中的内存浏览器或寄存器窗口通过 USERACCESS0/1 寄存器读取 PHY 和写入 PHY，这是对 PHY 问题进行故障排除的一种有效现场调试功能（请参阅 表 3-1）。

表 3-1. MDIOUSERACCESS0 寄存器字段说明

位	字段	类型	复位	说明
31	GO	R/W/S	0	Go.向此位写入 1 会使 MDIO 状态机在其方便时执行 MDIO 访问（这不是一个瞬间过程）。将 0 写入此位无影响。仅当启用了 MDIO 状态机时，此位才可写。当请求的访问完成后，此位将自行清除。当 GO 位为 1 时，系统将阻止对 MDIOUSERACCESS0 寄存器的任何写入。如果使用的是字节访问，则应最后写入 GO 位。
30	WRITE	R/W	0	写入启用。将此位设为 1 会使 MDIO 事务成为一个寄存器写入，否则为一个寄存器读取。
29	ACK	R/W	0	确认。如果 PHY 确认了读取事务，则设置此位。
28:26	保留	R	0	保留
25:21	REGADR	R/W	0	寄存器地址。指定此事务要访问的 PHY 寄存器。
20:16	PHYADR	R/W	0	PHY 地址。指定此事务要访问的 PHY。
15:0	数据	R/W	0	用户数据。从指定 PHY 寄存器读取或写入该寄存器的数据值。

若要在内存浏览器或寄存器窗口中读取 PHY，请执行以下步骤：

1. 确保将 MDIO 用户访问寄存器 (MDIOUSERACCESSn) 中的 GO 位清除。
2. 在 MDIOUSERACCESSn 中写入与要读取的 PHY 和 PHY 寄存器相对应的 GO、REGADR 和 PHYADR 位。
3. 当模块在串行总线上完成读取操作后，读取的数据值在 MDIOUSERACCESSn 的 DATA 位中可用。可通过轮询 MDIOUSERACCESSn 中的 GO 和 ACK 位来确定读取操作的完成情况。当 GO 位执行清除操作时，系统将在成功读取后设置 ACK 位。

若要在内存浏览器或寄存器窗口中写入 PHY，请执行以下步骤：

1. 确保将 MDIO 用户访问寄存器 (MDIOUSERACCESSn) 中的 GO 位清除。
2. 写入 MDIOUSERACCESSn 中与要写入的 PHY 和 PHY 寄存器相对应的 GO、WRITE、REGADR、PHYADR 和 DATA 位。
3. 对 PHY 的写入操作由 MDIO 模块进行安排并完成。可以通过轮询 MDIOUSERACCESSn 中的 GO 位为 0 来确定写入操作的完成情况。

若要验证 PHY 是否成功连接，用户可以查看基地址偏移处的 MDIOALIVE 和 MDIOLINK 寄存器 0x04 和 0x08。

如果 PHY 确认了对 PHY 的最近一次访问，则设置 MDIOALIVE 寄存器的 32 位中的每一位，其地址对应于寄存器位数。如果 PHY 无法确认该次访问，则会将该位复位。对 PHY 的用户访问和轮询访问都会导致对相应活动位的更新。这些活动位仅用于用相应地址表示 PHY 的存在或缺失。对任何位写入 1 都会将该位清除；写入 0 则无影响。

读取 PHY 的状态寄存器后，MDIOLINK 寄存器将更新。如果 PHY 与相应地址的链路接通，并且 PHY 确认读取事务，则设置该位。如果 PHY 指示它没有链路或无法确认读取事务，则将该位复位。写入寄存器无影响。此外，可以使用 MLINK 输入引脚来确定 MDIOUSERPHYSELn 寄存器中指定的两个 PHY 的状态。此选择由 MDIOUSERPHYSELn 寄存器中的 LINKSEL 位确定。

图 3-1 显示了在 AMIC110 ICE 上捕获的 Code Composer Studio PRU 寄存器窗口的一个快照。此快照指示存在两个 PHY (MDIOALIVE = 0x0002002)，并且具有地址 0x1 链路的 PHY 已接通 (MDIOLINK = 0x00000002)。

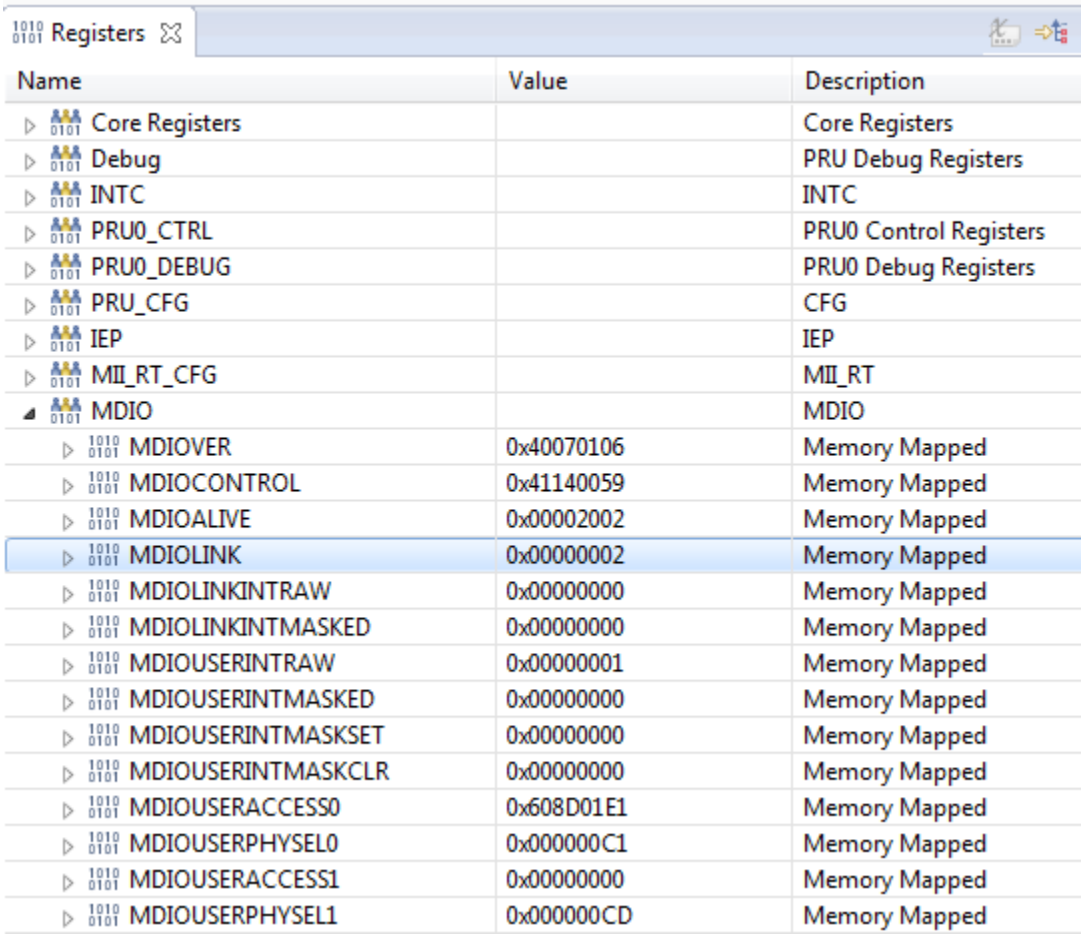

图 3-1. Code Composer Studio 中的 MDIO 寄存器

TI 的 EtherCAT 实现中提供了多个 PRU-ICSS MDIO 主机 API，这些 API 可用于读取/写入 PHY 寄存器以及查询 PHY 链路状态。例如，请参阅以下内容：

```
Int16 bsp_pruss_mdio_phy_read (Uint8 phyaddr, Uint8 regoffset, Uint16 *regval);
```

参数：

- phyaddr – 使用 PHY 地址选择要读取的 PHY
- regoffset – 要读取的 PHY 中的寄存器偏移
- regval – 指针，指向用于保存已读取寄存器值的变量
- 返回值
  - 0: 成功
  - – 1：MDIO 访问错误

```
Int16 bsp_pruss_mdio_phy_write (Uint8 phyaddr, Uint8 regoffset, Uint16 regval);
```

参数：

- phyaddr – 使用 PHY 地址选择要写入的 PHY
- regoffset – 要读取的 PHY 中的寄存器偏移
- regval – 要写入 PHY 寄存器的值
- 返回值
  - 0: 成功
  - – 1：MDIO 访问错误

```
Uint32 bsp_pruss_mdio_phy_link_state (Uint8 phyaddr);
```

参数：

- phyaddr – 选择 PHY 以获取链路状态
- 返回值
  - 0: 链路断开
  - 否则：链路接通

NOTE

bsp_pruss_mdio_phy_link_state( ) API 考虑 MII_LINK 信号极性差异，建议在启用 TIESC_MDIO_RX_LINK_ENABLE 的情况下使用该 API 来增强链路检测。

4 增强型链路检测

读取 PHY 状态寄存器后，可通过使用 MDIO 内部状态机来更新 MDIOLINK 寄存器中的链路状态。此外，也可以通过使用 MLINK 输入引脚来确定链路状态，具体取决于 MDIOUSERPHYSELn 寄存器中的 LINKSEL 位。基于 MDIO 状态机的检测速度很慢，因为 MDIO 控制器到 PHY 之间有一条串行链路用于传播讯息，通常需要的时间为 200 至 250µs。MLINK/mii_rxlink 检测发生的速度与 PHY 切换链路的速度一样快，通常在 10µs 内完成，这得益于 DP83822 器件先进的链接断开能力，该器件使用诸如 RX 误差计数、MLT3 误差计数、低 SNR 阈值以及信号和能量损失等标准来快速检测链路中断。

若要支持电缆冗余，必须快速进行链路中断检测，这需要 10 至 15µs 的链路中断检测时间。此持续时间为两到三个最小尺寸的 EtherCAT 帧，这是在链路故障或电缆断裂事件中可丢弃的最大值。

MDIOUSERPHYSELn 寄存器中的链路状态确定选择 (LINKSEL) 位通过使用 MLINK 引脚（将其设置为 1）来确定链路状态。默认值为 0，这说明链路状态由 MDIO 状态机确定。

PRU-ICSS 的 pr1_mii0_rxlink 和 pr1_mii1_rxlink 引脚（分别连接到 PHY 的 LED_LINK 或 LED_SPEED 引脚）作为 MLINK 信号连接到 MDIO。根据 PHY 自举设置，不同电路板上的链路极性不同，必须通过读取 MDIOLINK 寄存器为每个电路板进行调整。连接到 LED_LINK 引脚时，将 LED_LINK 控制模式设置为 *LINK_OK*（而不是 *RX/TX 活动*），以防止间歇性 RX/TX 流量导致链路检测失败（请参阅 表 4-1）。

表 4-1. LED 配置寄存器 1 (LEDCFG1)

位	字段	类型	复位	说明
15:12	保留	R/O	0	保留
11:8	LED_1 控制	R/W	0101	LED_1 控制：选择 LED_1 的源。 • 0000 =链路正常 • 0001 = RX/TX 活动 • 0010 = TX 活动 • 0011 = RX 活动 • 0100 = 碰撞 • 0101 = 速度，对于 100Base-TX 较高 • 0110 = 速度，对于 10Base-Te 较高 • 0111 = 全双工 • 1000 = 链路正常，针对 TX/RX 活动闪烁 • 1001 = 有源拉伸信号 • 1010 = MII 链路 (100BT+FD) • 1011 = LPI 模式 • 1100 = TX/RX MII 错误 • 1101 = 链路中断 • 1110 = 针对 PRBS 错误闪烁，LED 在清除 BMCR 寄存器（地址 0x0001）之前保持亮起状态。 • 1111 = 保留 链路中断，LED 在读取 BMCR 寄存器（地址 0x0001）之前保持亮起状态。 针对 PRBS 错误闪烁，LED 针对单个错误保持亮起状态，并且在清除 BICSR1 寄存器（地址 0x001B）之前保持亮起状态。
7 :4	LED_3 控制 (RX_D3)	R/W	0101	LED_3 控制：选择 RX_D3 的源。参考位 [11:8]，以获取源列表。
3:0	保留	R/W	0001	保留

如果增强型链路检测引脚连接到 AMIC110 ICE 上的 PHY_LED1，并且 PHY 配置设置为 LED1 针对活动闪烁，那么每当有活动时，在 MDIO 看来，链路正在打开和关闭，这会导致增强型链路检测失败。这不是针对增强型链路检测的正确 LED1 活动配置。此时应将 LED1 配置为指示 LINK OK（链路正常）。

若要更新源（LINK OK 而不是 RX/TX 活动）的 LED 配置，请使用 API：

```
Board_phyLED1Config((((PRUICSS_HwAttrs *)(pruIcssHandle->hwAttrs))->prussMiiMdioRegBase), phy0addr,
LED_CFG_MODE0);
Board_phyLED1Config((((PRUICSS_HwAttrs *)(pruIcssHandle->hwAttrs))->prussMiiMdioRegBase), phy1addr,
LED_CFG_MODE0);
```

其中 LED_CFG_MODE0 表示 0000 = LED_1 控制位中的 LINK OK。

若要启用增强型链路检测， 请使用 Board_phyFastLinkDownDetEnable( ) API 来配置 PHY 控制寄存器 3， 使用下列宏的任意组合作为其参数 val。

```
#define FAST_LINKDOWN_SIGENERGY        1u
#define FAST_LINKDOWN_LOWSNR           (1u<<1)
#define FAST_LINKDOWN_MLT3ERR          (1u<<2)
#define FAST_LINKDOWN_RXERR            (1u<<3)
void Board_phyFastLinkDownDetEnable(uint32_t mdioBaseAddress, uint32_t phyNum, uint8_t val);
```

当 PHY 没有 LED_LINK 或 LED_SPEED 信号，并且 pr1_mii0_rxlink 和 pr1_mii1_rxlink 信号可保持浮动时，还可在 MDIO 初始化期间通过使用 DISABLE 宏（而不是 ENABLE）来禁用增强型链路检测功能。

```
Int16 bsp_pruss_mdio_init (t_mdio_params *pmdio_params);
```

参数：

- pmdio_params – 指向 PRU-ICSS MDIO 初始化参数结构的指针
- pmdio_params->clkdiv – MDIO clkdiv
- pmdio_params->addr0 – 连接到 PRU-ICSS MII0 的 PHY 的地址
- pmdio_params->addr1 – 连接到 PRU-ICSS MII1 的 PHY 的地址
- pmdio_params->link0pol – 连接到 PRU-ICSS MII0 的 PHY 的 LINK_MII 信号极性
- pmdio_params->link1pol – 连接到 PRU-ICSS MII1 的 PHY 的 LINK_MII 信号极性
- mdio_params->enhancedlink_enable – 启用增强型链路检测功能

MDIO 初始化参数的结构：

```
typedef struct {
 Uint16 clkdiv;
 Uint8 addr0;
 Uint8 addr1;
 Uint8 link0pol; //1: 低电平有效 0: 高电平有效
 Uint8 link1pol; //1: 低电平有效 0: 高电平有效
 Uint8 enhancedlink_enable;
} t_mdio_params;
```

如果启用了增强型链路检测功能，则链路极性对 EtherCAT 非常重要。启用增强型链路检测功能后，链路极性参数 link0pol 和 link1pol 的设置 由 MDIOLINK 寄存器确定。若要检查链路极性：

1. 使用 MDIO 连接寄存器。
2. 将以太网线从 PC 插入 DUT 端口。
3. 观察任何变化，然后相应地调整 bsp_pruss_mdio_init( ) 中的链路极性字段。

例如，若要将两个 PHY 的极性均设置为 HIGH，请参阅以下内容：

```
    #define TIESC_LINK0_POL   TIESC_LINK_POL_ACTIVE_HIGH
    #define TIESC_LINK1_POL   TIESC_LINK_POL_ACTIVE_HIGH
    PRUICSS_V1_Object *object;
    object = (PRUICSS_V1_Object *)pruIcssHandle->object;
mdioParamsInit.addr0 = Board_getPhyAddress(object->instance, 1);
    mdioParamsInit.addr1 = Board_getPhyAddress(object->instance, 2);
    mdioParamsInit.enhancedlink_enable = TIESC_MDIO_RX_LINK_ENABLE;
    if(TIESC_MDIO_RX_LINK_ENABLE == mdioParamsInit.enhancedlink_enable)
    {
        //Enhanced link detection enabled
        mdioParamsInit.link0pol = TIESC_LINK0_POL;
        mdioParamsInit.link1pol = TIESC_LINK1_POL;
    }
    else
    {
        //Enhanced link detection disabled
        mdioParamsInit.link0pol = TIESC_LINK_POL_ACTIVE_HIGH;
        mdioParamsInit.link1pol = TIESC_LINK_POL_ACTIVE_HIGH;
    }
    bsp_pruss_mdio_init(pruIcssHandle, &mdioParamsInit);
```

TI EtherCAT 实现还提供了一些与 PHY 相关的寄存器，这些寄存器 在调试 PHY 配置时很有帮助，例如请参阅 表 4-2。

表 4-2. EtherCAT 中的 PHY 寄存器

位	字段	权限		寄存器偏移	复位	说明
		ECAT	PDI			
31:0	PRU MII RX LINK 极性	R/-	R/W	0x0E0C	0b	链路 LED 信号极性 PHY 地址 N (位 N)
7:0	Port0 PHY 地址	R/-	R/W	0x0E08	0	指定连接到 PRU 固件物理端口 0 的 PHY 的 PHY 地址
7:0	Port1 PHY 地址	R/-	R/W	0x0E09	0	指定连接到 PRU 固件物理端口 1 的 PHY 的 PHY 地址

PHY 链路状态和端口状态反映在标准 EtherCAT 数据链路 (DL) 状态寄存器中 (请参阅 表 4-3) 。

表 4-3. EtherCAT 中的数据链路状态寄存器

位	字段	权限		寄存器偏移	复位	说明
		ECAT	PDI			
15		R/-	R/-			端口 3 上的通信 – TI ESC 中不可用
14		R/-	R/-			环路端口 3 – TI ESC 中不可用
13		R/-	R/-			端口 2 上的通信 – TI ESC 中不可用
12		R/-	R/-			环路端口 2
11		R/-	R/-			端口 1 上的通信
10		R/-	R/-			环路端口 1
9		R/-	R/-			端口 0 上的通信 • 0 = 无稳定通信 • 1 = 已建立通信
8		R/-	R/-			环路端口 0 • 0 = 已打开 • 1 = 已关闭
7		R/-	R/-			端口 3 上的物理链路 – TI ESC 中不可用
6		R/-	R/-			端口 2 上的物理链路 – TI ESC 中不可用
5		R/-	R/-			端口 1 上的物理链路
4		R/-	R/-			端口 0 上的物理链路 • 0 = 无链路 • 1 = 检测到链路
3		R/-	R/-			保留
2		R/-	R/-			增强型链路检测 • 0 = 在所有端口上停用 • 1 = 在至少一个端口上激活
1		R/-	R/-			• 0 = PDI 看门狗已过期 • 1 = PDI 看门狗已重新加载
0		R/-	R/-			• 0 = EEPROM 未加载，PDI 不工作 (不访问 PD RAM) • 1 = EEPROM 已正确加载，PDI 工作 (访问 PD RAM)

NOTE

EtherCAT 对端口 OPEN/CLOSE 的定义是：如果没有链路，则端口不能为 OPEN（允许通信），必须为 CLOSE（无通信）。

5 在 Processor SDK 中添加 PHY

TI 的处理器 SDK 平台开发套件 (PDK) 中已经包含了若干 PHY 器件。PDK 是单个可扩展的软件驱动程序包，可提供跨不同处理器和平台的简化开发。PDK 软件包中包含器件抽象层库和外设/板级样片/演示示例，这些示例演示了外设在平台上开发、部署和执行应用的能力。

电路板库支持 PDK 中的 PHY 配置，如以下软件结构所示（请参阅 图 5-1）。

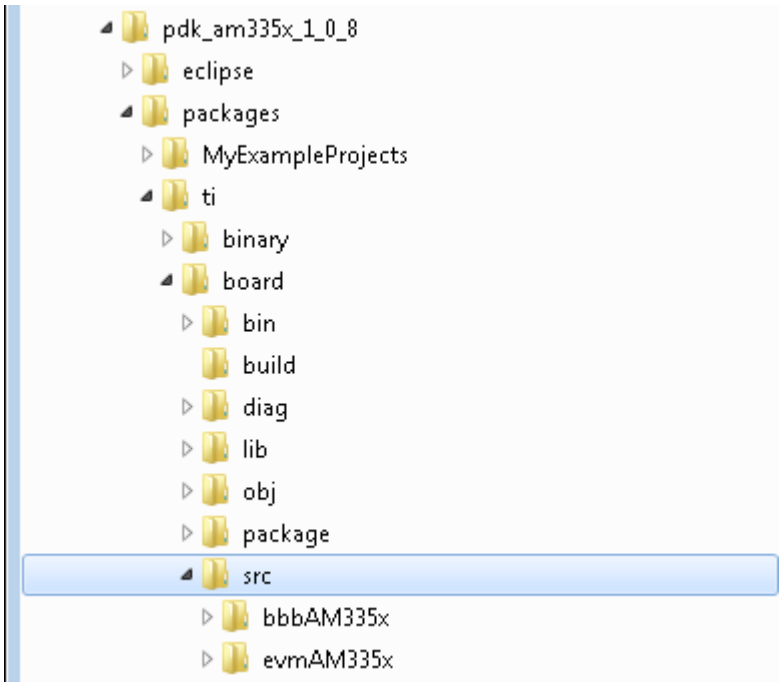

图 5-1. PDK 中的电路板库

电路板库为以下内容提供了高级抽象模式：

- Pinmux
- 时钟树
- 配置
- 电路板器件
- 存储器映射
- 板级多路复用器控制元件
- 电路板电源
- I/O 扩展器

电路板库 API 对电路板类型和定义执行自动检测，然后根据电路板配置和板载器件说明来开发抽象模式。

对于某些平台，PHY 的配置函数在 board\src\<BOARD>\device\enet_phy.c 中受到支持，而对于其他平台（如 AMIC110 ICE），其 PHY 函数在 src\<BOARD>\<PROCESSOR>_ethernet_config.c 中受到支持，如 图 5-2 中所示。

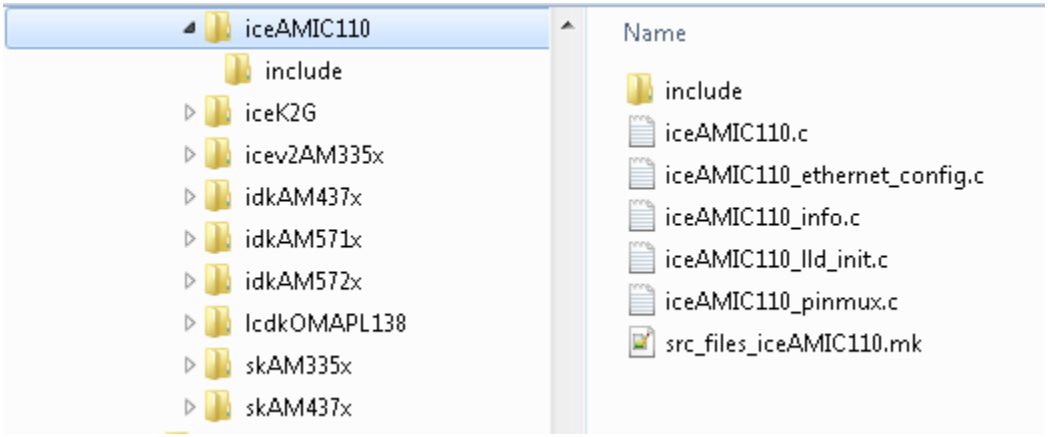

图 5-2. AMIC110 ICE 电路板库

要添加新电路板和 PHY（请参阅 图 5-3），请从一个平台迁移以下主要项目：

- Pinmux 设置
- 时钟和计时器配置更改。
- 存储器配置
- 中断更改
- I/O 更改（MII、MDIO、PHY、GPIO 和 UART）
- 电路板元件替代
- 电路板初始化（之前所有内容中的图）

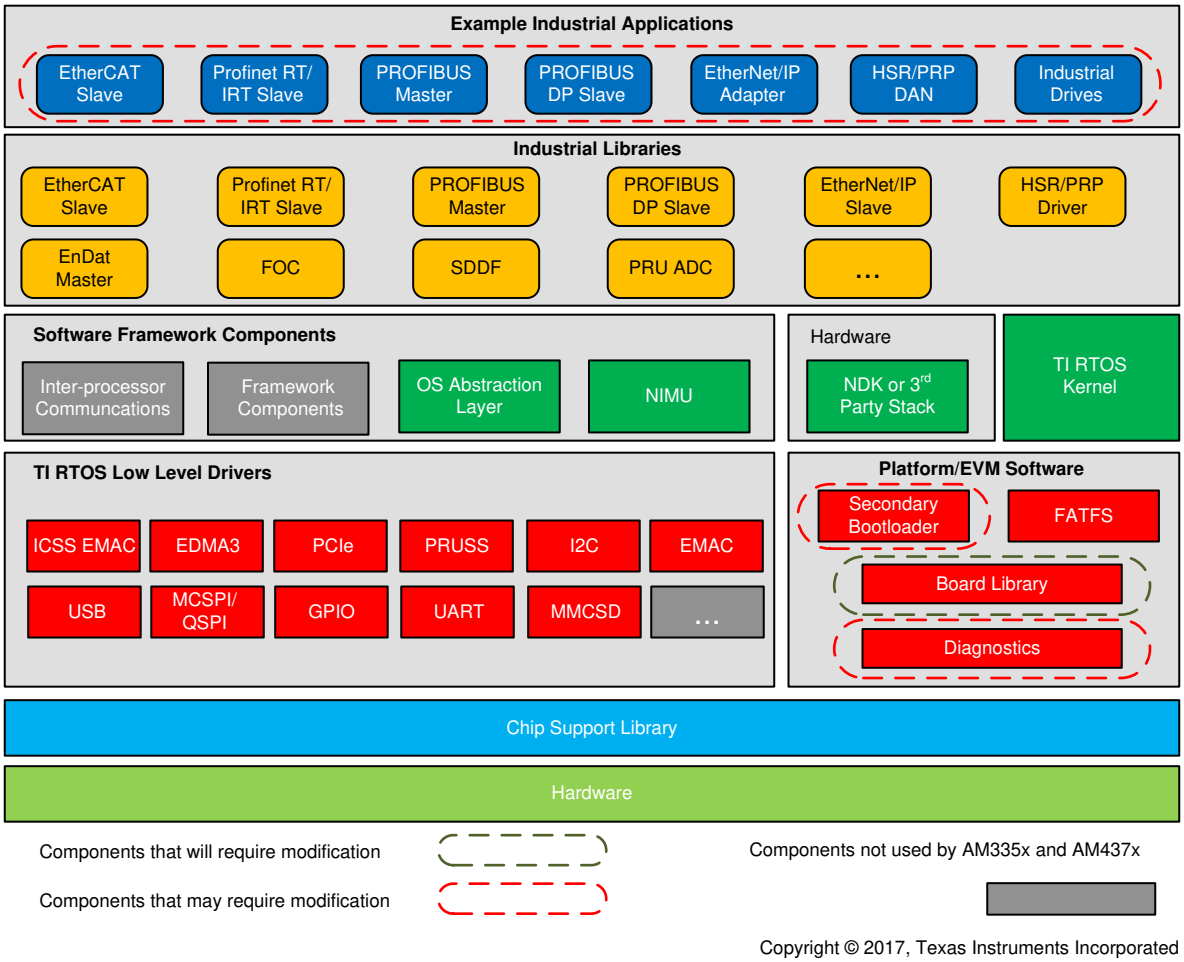

图 5-3. 添加新电路板和 PHY

6 结论

在定制电路板上调出以太网是一项挑战，因此使用 MDIO 进行正确的 PHY 配置是这一过程中必不可少的一步。本应用报告总结了使用 MDIO 配置以太网 PHY 的流程（包括 PHY 选择和连接、复位和地址、速度/双工/自动 MDIX），并解释了工业特定功能 – 增强型链路检测，最后讨论了如何在 TI 的统一软件 Processor SDK 中添加新 PHY。

7 参考文献

1. 德州仪器 (TI)，《DP83865 和 DP83864 千兆位物理层器件故障排除指南》，应用报告
2. 德州仪器 (TI)，《DP83867 故障排除指南》，应用报告
3. 德州仪器 (TI)，《DP83822 硬件翻转》，应用报告
4. Beckhoff，《PHY 选择指南》，应用手册
5. 德州仪器 (TI)，DP83822 低功耗耐用型 10/100Mbps 以太网物理层收发器，数据表
6. 德州仪器 (TI)，适用于 Sitara™ 处理器的 PRU-ICSS 工业软件
7. 德州仪器 (TI)，PRU ICSS EtherCAT 从控制器寄存器列表，TI Wiki
8. 德州仪器 (TI)，《工业 SDK EMAC 移植指南》，TI Wiki

8 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 * (December 2017) to Revision A (May 2021)	Page
• 更新了整个文档中的表格、图和交叉参考的编号格式。	2

重要声明和免责声明

TI 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安全、安保或其他要求。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 (https:www.ti.com/legal/termsofsale.html) 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。

邮寄地址：Texas Instruments, Post Office Box 655303, Dallas, Texas 75265

重要声明和免责声明

TI 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安全、安保或其他要求。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 (https:www.ti.com.cn/zh-cn/legal/termsofsale.html) 或 ti.com.cn 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。

邮寄地址：上海市浦东新区世纪大道 1568 号中建大厦 32 楼，邮政编码：200122