

# 使用 C2000 EtherCAT 从站控制器的 SMI 进行以太网 PHY 配置



Bruce Liu, Chen Gao

Systems Engineering and Marketing

## 摘要

以太网物理层 (PHY) 是用于发送和接收以太网帧数据的收发器元件，而 PHY 器件在开放系统互连 (OSI) 模型中实现了物理层。PHY 器件用作介质访问控制器 (MAC - OSI 模型中的数据链路层) 与物理介质 (例如铜缆或光缆) 之间的桥梁。

串行管理接口 (SMI) 支持访问 PHY 器件内部寄存器空间，从而获得状态信息和配置。使用 SMI 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。SMI 符合 IEEE 802.3 第 22 条和第 45 条条款。所实现的寄存器组包括 IEEE 802.3 所需寄存器和其他几个寄存器，能够提高 PHY 器件的可见性和可控性。

本应用手册为针对工业应用在 C2000™ 器件中使用 EtherCAT 从站控制器 (ESC) SMI 进行以太网 PHY 配置提供了指导。

## 内容

1 SMI 简介.....	2
2 EtherCAT 的 PHY 选择和配置.....	3
3 如何使用 ESC 的 SMI 读写 PHY 寄存器.....	6
3.1 EtherCAT 的 PHY 寄存器配置.....	6
3.2 在 C2000 ESC 中读取或写入 PHY 寄存器的步骤.....	6
3.3 在 CCS 中使用脚本调试以太网 PHY 寄存器.....	8
4 总结.....	10
5 参考文献.....	10

## 插图清单

图 1-1. PHY 管理接口连接.....	2
图 2-1. 硬件自动加载 (bootstrap).....	3
图 2-2. DP83822 PHY 硬件复位信号.....	4
图 2-3. ESC PHY MII 接口图.....	5
图 3-1. 脚本接口.....	9
图 3-2. 带有 DP83822 寄存器转储的 F28388 控制卡.....	10

## 商标

C2000™ and Code Composer Studio™ are trademarks of Texas Instruments.

EtherCAT® and Beckhoff® are registered trademarks of Beckhoff Automation GmbH.

所有商标均为其各自所有者的财产。

## 1 SMI 简介

用于控制自动化技术的以太网 (EtherCAT®) 是一种基于以太网的现场总线系统，由 Beckhoff® Automation 发明，并在 IEC 61158 中进行了标准化。连接到总线的所有从节点均动态解释、处理和修改地址数据 (根据需要)，而无需在节点内缓冲帧。这种实时行为、帧处理和转发要求由 EtherCAT 从站控制器硬件实现。EtherCAT 不需要软件交互即可在从器件内部进行数据传输。EtherCAT 仅定义 MAC 层，而更高层协议和堆栈在连接到 ESC 的微控制器上的软件中实现。C2000 ESC 中的 SMI 被称为 PHY 管理接口，用于与以太网 PHY 通信。如需了解 SMI 的连接，请参阅图 1-1。

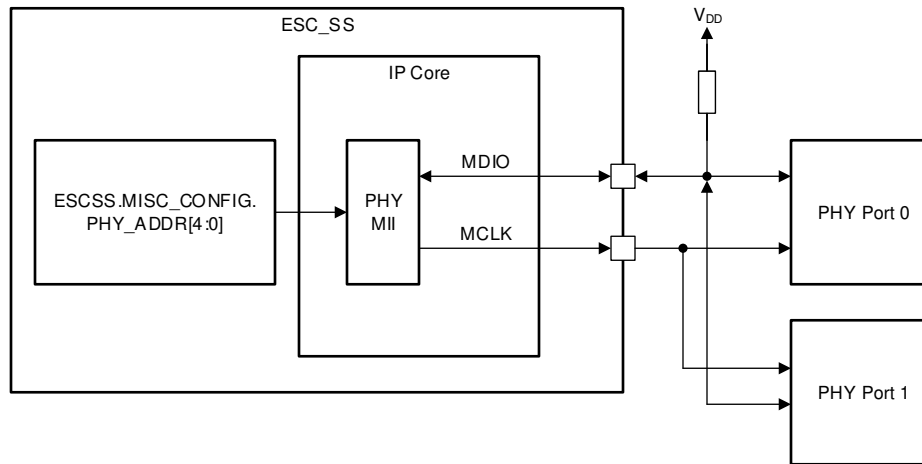


图 1-1. PHY 管理接口连接

MDIO 必须外接上拉电阻 (建议使用  $4.7k\Omega$ )。MCLK 为轨到轨驱动，空闲值为高电平。

SMI 包括管理时钟 (MDC) 和管理输入/输出数据引脚 (MDIO)。MDC 由 ESC 提供，可在 25MHz 的最大时钟速率下运行。MDC 不应持续运行，在总线空闲时可由 ESC 关闭。

MDIO 由 ESC 和 PHY 提供。MDIO 引脚上的数据在 MDC 的上升沿锁存。MDIO 引脚需要上拉电阻器，可在 IDLE 和转换期间将 MDIO 拉高。

## 2 EtherCAT 的 PHY 选择和配置

为 EtherCAT 选择合适的 PHY 时，首先查看 EtherCAT 主页上的[应用手册 - PHY 选择指南](#)。本应用手册从 EtherCAT 的角度介绍了 PHY 的性能规格和建议。本文档还列出了 TI 出品的各种 PHY，如 DP836x、DP838x、TLK10x 和 TLK11x。

正确配置 PHY 以符合 IEEE 802.3 100BaseTX 或 100BaseFX，包括：

- 支持 100Mbps 全双工链路
- 提供 MII ( 或 RMII、RGMII ) 接口
- 100Base TX 模式下的自动协商
- 支持 MII 管理接口
- 100BaseTX 模式下的 MDI、MDI-X 自动交叉
- 接收和发送延迟必须符合标准 ( RX 延迟目标低于约 320ns，TX 延迟低于约 140ns )
- 必须提供 RX\_ER 信号 ( MII、RMII ) 或 RX\_ER 作为 RX\_CTL 信号的一部分 (RGMII)
- 链路中断反应时间 ( 链路中断到链路信号或 LED 输出变化的时间 ) 必须小于 15  $\mu$ s，才能实现冗余操作

为了将 PHY 设置为正确的模式以使其能够在 EtherCAT 环境中工作，可使用串行管理接口 (SMI) 对要在特定模式下设置的 PHY 进行编程。

最多 32 个 PHY 可共用一条公共 SMI 总线。为区分 PHY，采用了 x 位地址。在上电或硬件复位期间，PHY 器件锁存 PHY\_AD[x:0] 配置引脚以确定地址。可以通过添加 PHY 器件数据表的“自动加载 (bootstrap)”部分中定义的所需上拉或下拉电阻器来更改地址。自动加载 (bootstrap) 引脚也可用于 PHY 配置。本应用手册仅重点介绍了如何使用 SMI。有关硬件自动加载 (bootstrap) 图，请参阅图 2-1。

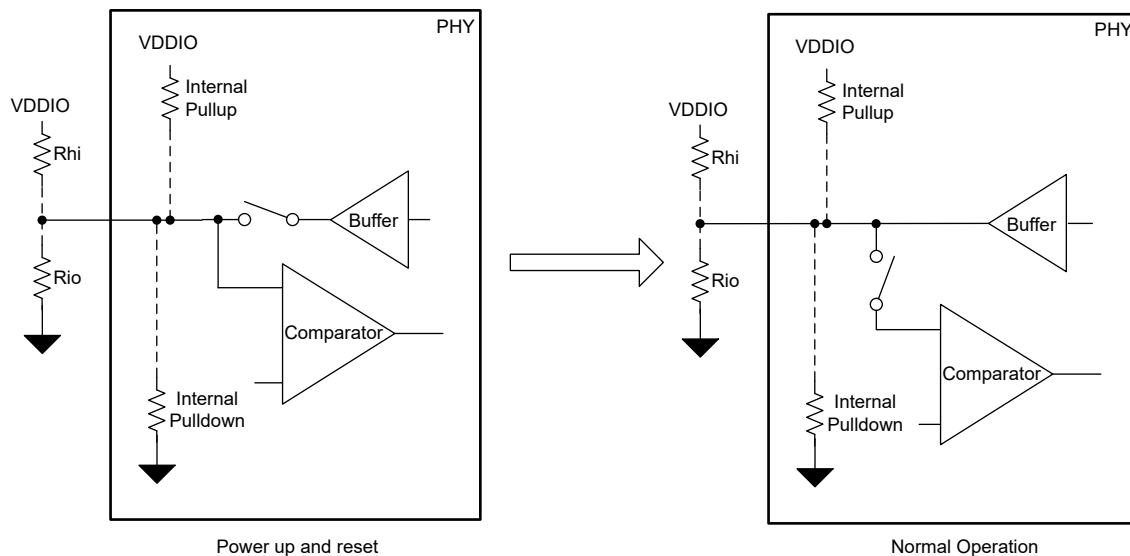


图 2-1. 硬件自动加载 (bootstrap)

C2000 ESC 通常使用逻辑端口号加上 PHY 地址偏移来寻址以太网 PHY。在理想情况下，以太网 PHY 地址与逻辑端口号相对应，因此使用 PHY 地址 0 和 1。可以应用 0 到 31 的 PHY 地址偏移，将 PHY 地址移动到任何连续的地址范围。ESC 模块期望逻辑端口 0 具有 PHY 地址 0 加上 PHY 地址偏移量。PHY 地址偏移可在寄存器 ESCSS\_MISC\_CONFIG.PHY\_ADDR[4:0] 中选择。

在进入所需的运行模式之前，PHY 器件必须通过向 RESET 引脚施加高电平来脱离 RESET 条件。此 RESET 信号由 ESC 模块生成。由于在复位期间和复位后，MCU 上没有激活的上拉器件，因此必须在电路板级的这个信号上添加一个下拉电阻器。在某些情况下，可在释放 ESC 模块后从复位状态释放 PHY。为了生成延迟，nPHY\_RESET 的引脚可被用作一个 I/O 并在晚些时候被切换替代输出功能。此外，通过对 RESET 引脚（以 DP83822 PHY 为例，请参阅图 2-2）施加一个持续时间至少为 10 $\mu$ s (T1) 的低脉冲，硬件复位可以将所有 PHY 寄存器重新初始化为默认值。

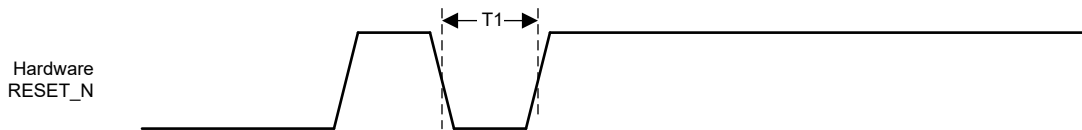


图 2-2. DP83822 PHY 硬件复位信号

ESC 和 PHY 设备之间的接口图如图 2-3 所示。如果需要，可以使用 ESCSS\_PHY\_CLK 信号为 PHY 计时，否则为 PHY 和 ESC 提供外部 25MHz 源（两者都必须由同一个源计时）。

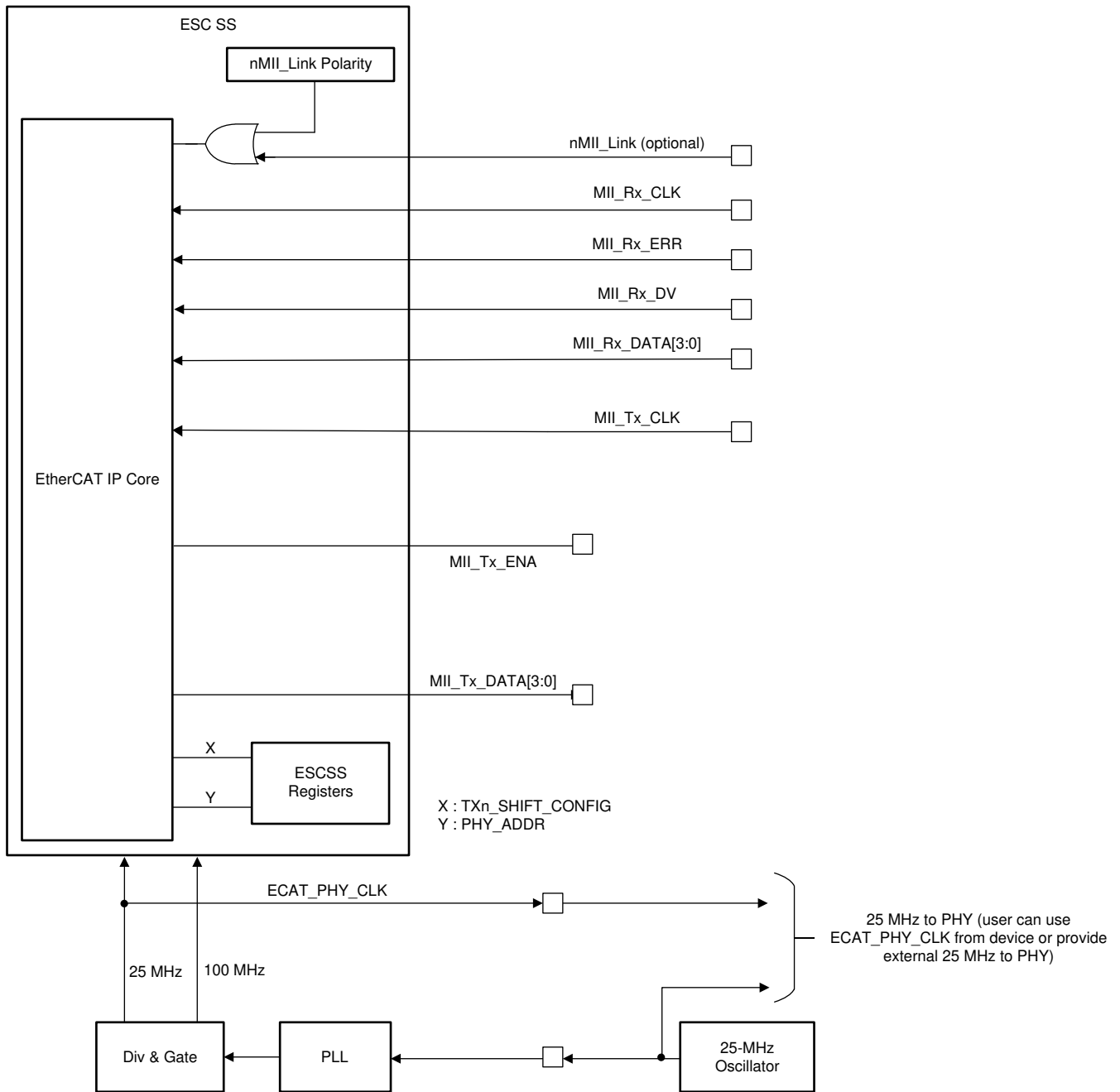


图 2-3. ESC PHY MII 接口图

## 3 如何使用 ESC 的 SMI 读写 PHY 寄存器

### 3.1 EtherCAT 的 PHY 寄存器配置

以 F28388D controlCARD (TMDSCNCD28388D) 为例，控制卡上有两个分别具有 PHY 地址 0x00 和 0x01 的 EtherCAT 端口。使用 SMI 对 DP83822 PHY 寄存器进行编程以用于 EtherCAT 100BASE-TX，如下所示。

- **LED 0 :**  
PHY 寄存器 0x19 值 0x8020 ( 启用自动 MDIX 并启用 LED0 配置 )  
PHY 寄存器 0x18 值 0x0080 ( 高电平有效极性 )
- **LED 1 :**  
PHY 寄存器 0x469 值 0x0440 ( 高电平有效极性 )  
*自动协商使能配置 :*  
确保 PHY 寄存器 0x04 值默认为 0x01E1 ( 通告 PHY 支持哪些模式 )  
PHY 寄存器 0x09 值 0x0020 ( 启用强大的自动 MDIX )  
PHY 寄存器 0x00 值 0x3300 ( 启用自动协商并重新启动过程 )
- **奇半字节检测禁用配置 :**  
PHY 寄存器 0x0A 值 0x0002 ( 禁用奇半字节检测 )
- **启用快速链路丢弃功能 :**  
PHY 寄存器 0x0B 值 0x0008 ( 使用正确的 FLD 功能 RX 错误计数启用 FLD )

### 3.2 在 C2000 ESC 中读取或写入 PHY 寄存器的步骤

#### 1. 授予 PDI 访问 MII 管理权限 :

根据 [ESC 硬件](#) 数据表，默认情况下 MII 管理控制仅设置为 ECAT 主机。PDI 必须通过 EtherCAT IP 寄存器 **MIU 管理 PDI 访问状态 (0x0517)** 声明对 MII 管理的访问权限。这样才可以开启 PDI 对 MII 的管理访问权限。启用后，PDI 可以通过 **MIU 管理控制** 和 **PHY 地址** 寄存器访问 PHY 寄存器。以下代码中显示了一个示例：

```
#define ESC_MII_PDI_ACCESS_OFFSET 0x0517 //0x28B - High for C28x, 0x0517 for CM
ESC_writeWordISR(0x0100, ESC_MII_ECAC_ACCESS_OFFSET); //0x0100 for C28x, 0x01 for CM
```

由于可寻址寄存器长度 ( C28 为 16 位，CM 为 8 位 )，C28 内核和 CM (M4) 内核的 ESC 基地址和偏移量是不同的。由于 ESC 寄存器是 8 位的，因此使用 C28 内核控制的偏移量和地址需要减半。

#### 2. 将 PHY 地址设置为读取或写入 :

ESC 寄存器 0x0512 定义了要读取或写入的 PHY 地址。请参阅以下代码：

```
#define ESC_PHY_ADDRESS_OFFSET 0x0512 //0x289 - low for C28x, 0x0512 for CM
ESC_writeWordISR(PHY_address, ESC_PHY_ADDRESS_OFFSET);
```

#### 3. 设置要写入 PHY 寄存器 ( 写入 ) 的值 :

ESC 寄存器 0x0514:0x0515 定义要读取或写入 PHY 寄存器的 PHY 数据。请参阅以下代码：

```
#define ESC_PHY_DATA_OFFSET 0x0514 //0x28A - low for C28x, 0x0514 for CM
ESC_writeWordISR(PHY_data, ESC_PHY_DATA_OFFSET);
```

#### 4. 启动读取或写入命令：

ESC 寄存器 0x0510:0x0511 定义了 MII 管理控制和状态。位 9:8 定义了读取或写入 PHY 寄存器的命令。

位 9:8 命令：

00：无命令/MI 空闲 (清除错误位)

01：读取

10：写入

11：保留/无效命令 (不发出)

请参阅以下代码：

```
#define ESC_MII_CTRL_STATUS_1_OFFSET 0x0510 //0x288 - low for C28x, 0x0510 for CM
#define ESC_MII_CTRL_STATUS_2_OFFSET 0x0511 //0x288 - high for C28x, 0x0511 for CM
ESC_writeWordISR(0x0200, ESC_MII_CTRL_STATUS_1_OFFSET); //Write command for C28x
ESC_writeWordISR(0x0200, ESC_MII_CTRL_STATUS_2_OFFSET); // Write command for CM
ESC_writeWord(0x0100, ESC_MII_CTRL_STATUS_1_OFFSET); //Read command for C28x
ESC_writeWord(0x0100, ESC_MII_CTRL_STATUS_2_OFFSET); //Read command for CM
```

前面介绍的方法是访问 IEEE 802.3 中定义的标准寄存器 0 至 31。要访问第 45 条扩展寄存器集，需要设置寄存器控制寄存器 (REGCR，地址 0x000D) 和数据寄存器 (ADDAR，地址 0x000E)。寄存器 REGCR [4:0] 为器件地址 DEVAD，可将 ADDAR 寄存器的任何访问引向适宜 MDIO 管理器件。以下示例演示了根据 [DP83822 低功耗耐用型 10/100Mbps 以太网物理层收发器](#) 数据表中的示例写入操作部分，无后增量的写入操作。在本例中，使用 IO MUX GPIO 控制寄存器 (IOCTRL，地址 0x0461) 将 MAC 阻抗调整为 99.25 Ω。

- 将值 0x001F 写入寄存器 0x000D
- 将值 0x0461 写入寄存器 0x000E (将所需寄存器设置为 IOCTRL)
- 将值 0x401F 写入寄存器 0x000D
- 将值 0x0400 写入寄存器 0x000E (将 MAC 阻抗设置为 99.25 Ω)

请参阅 CCS 中的代码。

```
#define ESC_PHY_REG_ADDRESS_OFFSET 0x0513 //0x289 - High for C28x, 0x0513 for CM
#define ESC_PHY_DATA_OFFSET 0x0514 //0x28A - low for C28x, 0x0514 for CM
#define ESC_MII_CTRL_STATUS_1_OFFSET 0x0510 //0x288 - low for C28x, 0x0510 for CM
#define ESC_MII_CTRL_STATUS_2_OFFSET 0x0511 //0x288 - high for C28x, 0x0511 for CM
ESC_writeWordISR(0x0D00, ESC_PHY_REG_ADDRESS_OFFSET); //0x0D for CM, set extended PHY register control
ESC_writeWordISR(0x001F, ESC_PHY_DATA_OFFSET); // DEVAD for MMD
ESC_writeWord(0x0200, ESC_MII_CTRL_STATUS_1_OFFSET); //write command for C28x, status_2_offset register for CM
ESC_writeWordISR(0x0E00, ESC_PHY_REG_ADDRESS_OFFSET); //0x0E for CM, set extended PHY Data register
ESC_writeWordISR(0x0461, ESC_PHY_DATA_OFFSET); // PHY extended register address
ESC_writeWord(0x0200, ESC_MII_CTRL_STATUS_1_OFFSET); //write command for C28x, status_2_offset register for CM
ESC_writeWordISR(0x0D00, ESC_PHY_REG_ADDRESS_OFFSET); //0x0D for CM, set extended PHY register control
ESC_writeWordISR(0x401F, ESC_PHY_DATA_OFFSET); // change to Data in REGCR Bit 15:14
ESC_writeWord(0x0200, ESC_MII_CTRL_STATUS_1_OFFSET); //write command for C28x, status_2_offset register for CM
ESC_writeWordISR(0x0E00, ESC_PHY_REG_ADDRESS_OFFSET); //0x0E for CM, set extended PHY Data register
ESC_writeWordISR(0x0400, ESC_PHY_DATA_OFFSET); // PHY extended register value to be written
ESC_writeWord(0x0200, ESC_MII_CTRL_STATUS_1_OFFSET); //write command for C28x, status_2_offset register for CM
```

读取扩展 PHY 寄存器，下面的示例演示了读操作。在此示例中，MMD7 节能以太网链路伙伴能力寄存器 (MMD7\_EEE\_LP\_ABILITY，地址 0x703D) 被读取。

- 将值 0x0007 写入寄存器 0x000D
- 将值 0x003D 写入寄存器 0x000E (将所需寄存器设置为 MMD7\_EEE\_LP\_ABILITY)
- 将值 0x4007 写入寄存器 0x000D
- 读取寄存器 0x000E 的值 (读取的数据是 MMD7\_EEE\_LP\_ABILITY 中包含的值)

请参阅 CCS 中的代码。

```
#define ESC_PHY_REG_ADDRESS_OFFSET 0x0513 //0x289 - High for C28x, 0x0513 for CM
#define ESC_PHY_DATA_OFFSET 0x0514 //0x28A - low for C28x, 0x0514 for CM
#define ESC_MII_CTRL_STATUS_1_OFFSET 0x0510 //0x288 - low for C28x, 0x0510 for CM
#define ESC_MII_CTRL_STATUS_2_OFFSET 0x0511 //0x288 - high for C28x, 0x0511 for CM
ESC_writeWordISR(0x0D00, ESC_PHY_REG_ADDRESS_OFFSET); //0x0D for CM, set extended PHY register
control
ESC_writeWordISR(0x0007, ESC_PHY_DATA_OFFSET); // DEVAD for MMD7
ESC_writeWord(0x0200, ESC_MII_CTRL_STATUS_1_OFFSET); //write command for C28x, status_2_offset
register for CM
ESC_writeWordISR(0x0E00, ESC_PHY_REG_ADDRESS_OFFSET); //0x0E for CM, set extended PHY Data register
ESC_writeWordISR(0x003D, ESC_PHY_DATA_OFFSET); // PHY extended register address
ESC_writeWord(0x0200, ESC_MII_CTRL_STATUS_1_OFFSET); //write command for C28x, status_2_offset
register for CM
ESC_writeWordISR(0x0D00, ESC_PHY_REG_ADDRESS_OFFSET); //0x0D for CM, set extended PHY register
control
ESC_writeWordISR(0x4007, ESC_PHY_DATA_OFFSET); // change to Data in REGCR Bit 15:14
ESC_writeWord(0x0200, ESC_MII_CTRL_STATUS_1_OFFSET); //write command for C28x, status_2_offset
register for CM
ESC_writeWordISR(0x0E00, ESC_PHY_REG_ADDRESS_OFFSET); //0x0E for CM, set extended PHY Data register
ESC_writeWord(0x0100, ESC_MII_CTRL_STATUS_1_OFFSET); //Read command for C28x, status_2_offset
register for CM
```

### 3.3 在 CCS 中使用脚本调试以太网 PHY 寄存器

节 3.2 介绍了使用 C2000 ESC 读取和写入 PHY 寄存器的方法。为了简化该方法，可在调试阶段使用脚本读取或写入 PHY 寄存器转储并在 CCS 的控制台窗口中打印。

通用扩展语言 (GEL) 可用于配置 Code Composer Studio™ 开发环境和初始化目标 CPU。GEL 是一种解释语言，语法类似于 C 语言。提供了一组丰富的内置 GEL 函数，也可以编写自定义 GEL 函数。本文档介绍了创建编写良好的 GEL 启动文件的方法。有关建议的摘要，请参阅 [GEL 指南](#)。

以下列表详细介绍了分步指南：

1. 配置可立即在 C2000\_ESC\_PHY\_READ.gel 中使用的正确 PHY 地址和内核 (c28x 或 cm)
2. 使用 “Tools” > “GEL Files” > “Load GEL” 选项添加 GEL 文件
3. 初始化 PHY 和 ESC SMI (例如；运行 f2838x\_cpu1\_pdi\_hal\_test\_app)，然后使用 “Scripts” > “ESC SMI configuration” > “C2000\_ESC\_PHY\_Reg\_Dump” 运行脚本



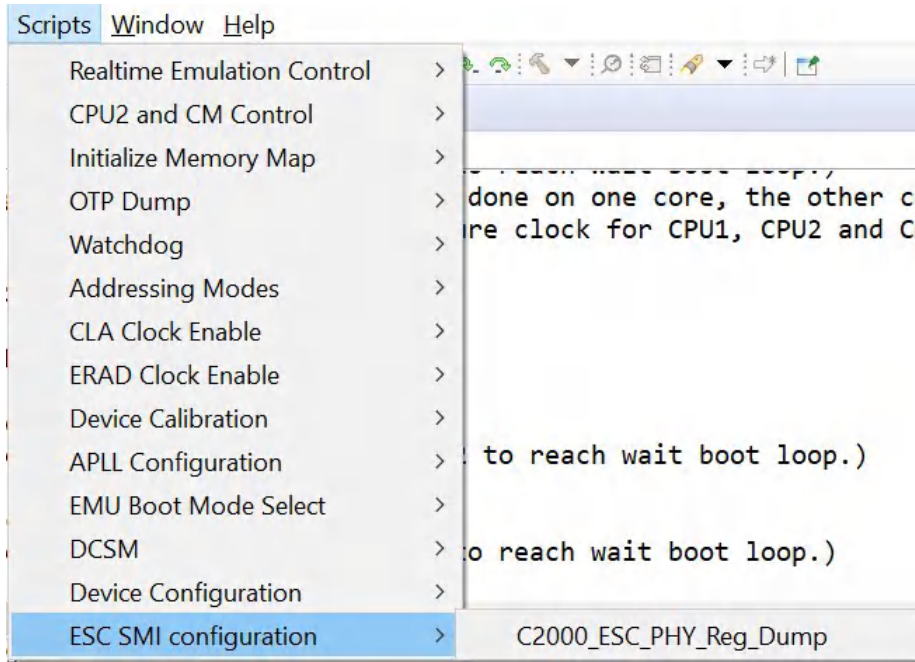


图 3-1. 脚本接口

您可以将经常访问的 GEL 函数添加到 Code Composer Studio 菜单栏的 GEL 菜单中，如图 3-1 所示。为此，请使用“menuitem”关键字在 GEL 菜单下创建菜单项的新下拉列表。然后，使用关键字“hotmenu”、“dialog”或“slider”在最新的下拉列表中添加新菜单项。选择用户定义的菜单项（在 GEL 菜单下）后，会出现一个对话框或滑块对象。示例代码如下：

```

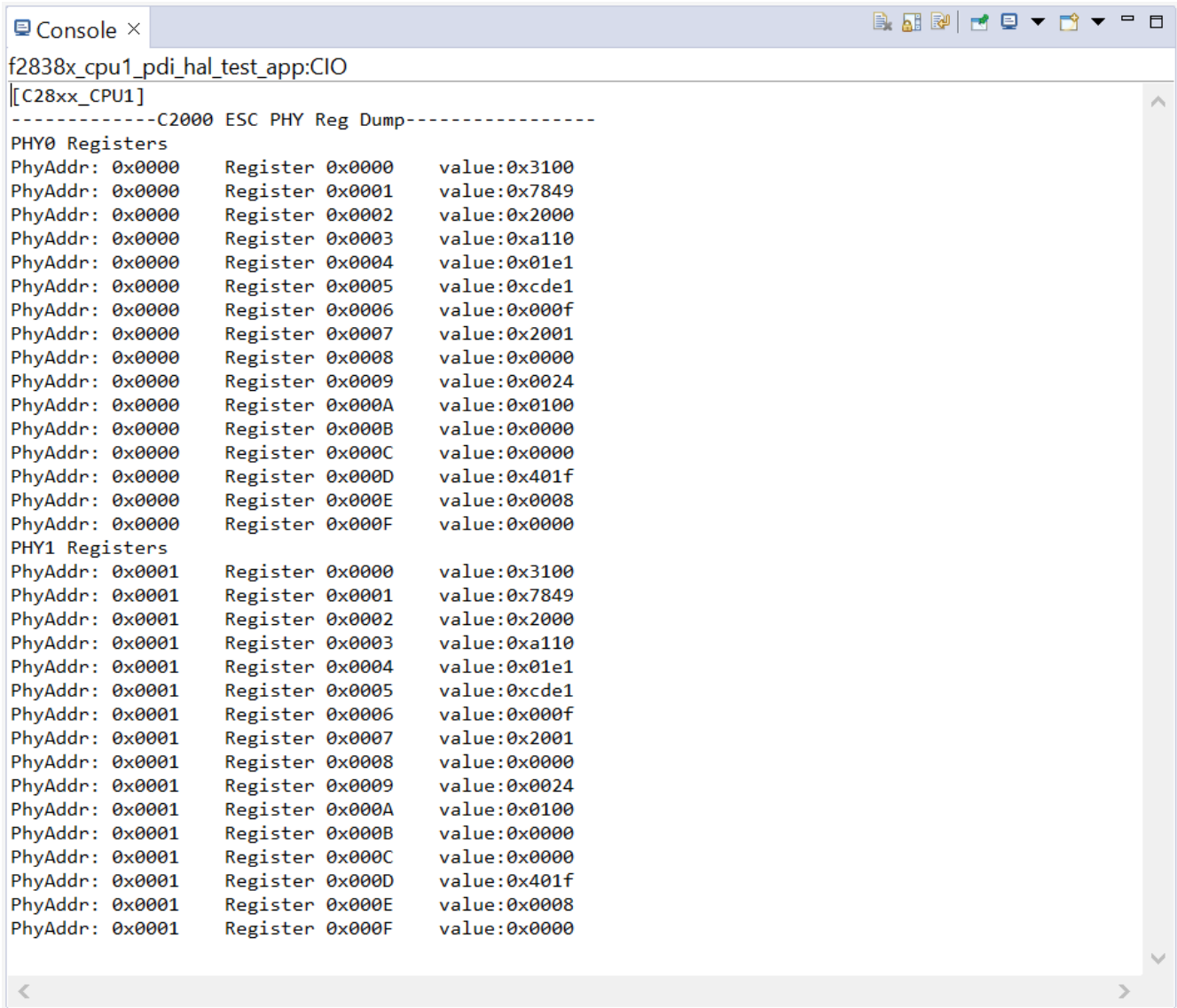
menuitem "ESC SMI configuration";
hotmenu C2000_ESC_PHY_Reg_Dump()
{}
  
```

在 gel 文件中读取和写入寄存器的方法需要重写，因为没有库函数的干预。示例代码如下：

```

ESC_readWordISR(unsigned short wordValue, unsigned short address)
{
  unsigned short *p_address;
  p_address = (unsigned short *) address;
  wordValue = *p_address;
}
ESC_writeWordISR(unsigned short wordValue, unsigned short address)
{
  unsigned short *p_address;
  p_address = (unsigned short *) address;
  wordValue = *p_address;
}
  
```

最后，在控制台窗口中找到 PHY 寄存器转储，如图 3-2 所示。



```

Console ×
f2838x_cpu1_pdi_hal_test_app:CIO
[[C28xx_CPU1]]
-----C2000 ESC PHY Reg Dump-----
PHY0 Registers
PhyAddr: 0x0000   Register 0x0000   value:0x3100
PhyAddr: 0x0000   Register 0x0001   value:0x7849
PhyAddr: 0x0000   Register 0x0002   value:0x2000
PhyAddr: 0x0000   Register 0x0003   value:0xa110
PhyAddr: 0x0000   Register 0x0004   value:0x01e1
PhyAddr: 0x0000   Register 0x0005   value:0xcde1
PhyAddr: 0x0000   Register 0x0006   value:0x000f
PhyAddr: 0x0000   Register 0x0007   value:0x2001
PhyAddr: 0x0000   Register 0x0008   value:0x0000
PhyAddr: 0x0000   Register 0x0009   value:0x0024
PhyAddr: 0x0000   Register 0x000A   value:0x0100
PhyAddr: 0x0000   Register 0x000B   value:0x0000
PhyAddr: 0x0000   Register 0x000C   value:0x0000
PhyAddr: 0x0000   Register 0x000D   value:0x401f
PhyAddr: 0x0000   Register 0x000E   value:0x0008
PhyAddr: 0x0000   Register 0x000F   value:0x0000
PHY1 Registers
PhyAddr: 0x0001   Register 0x0000   value:0x3100
PhyAddr: 0x0001   Register 0x0001   value:0x7849
PhyAddr: 0x0001   Register 0x0002   value:0x2000
PhyAddr: 0x0001   Register 0x0003   value:0xa110
PhyAddr: 0x0001   Register 0x0004   value:0x01e1
PhyAddr: 0x0001   Register 0x0005   value:0xcde1
PhyAddr: 0x0001   Register 0x0006   value:0x000f
PhyAddr: 0x0001   Register 0x0007   value:0x2001
PhyAddr: 0x0001   Register 0x0008   value:0x0000
PhyAddr: 0x0001   Register 0x0009   value:0x0024
PhyAddr: 0x0001   Register 0x000A   value:0x0100
PhyAddr: 0x0001   Register 0x000B   value:0x0000
PhyAddr: 0x0001   Register 0x000C   value:0x0000
PhyAddr: 0x0001   Register 0x000D   value:0x401f
PhyAddr: 0x0001   Register 0x000E   value:0x0008
PhyAddr: 0x0001   Register 0x000F   value:0x0000
    
```

图 3-2. 带有 DP83822 寄存器转储的 F28388 控制卡

## 4 总结

本应用手册为针对工业应用在 C2000 器件中使用 EtherCAT 从站控制器 SMI 进行以太网 PHY 配置提供了指导。还在 Code Composer 中提供了一个简单的脚本工具，用于读取/写入 PHY 寄存器。

## 5 参考文献

1. 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
2. 德州仪器 (TI), [具有连接管理器的 TMS320F2838x 实时微控制器](#) 技术参考手册
3. 德州仪器 (TI), [TMS320F28388D controlCARD 信息指南](#) 用户指南
4. Beckhoff, [EtherCAT 从站控制器](#) 硬件数据表第 I 部分
5. Beckhoff, [EtherCAT 从站控制器 - PHY 选择指南](#) 应用手册

## 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