

引言

本设计实现了两种应用广泛的共享总线体系结构之间的协议转换：串行外设接口 (SPI) 和 I²C 总线。Altera® Max® II CPLD 起到了桥接的作用，使 SPI 接口主机能够与通过 I²C 总线连接的器件进行通信。

I²C 和 SPI

I²C 是串行 2 线窄带工业标准协议，用于嵌入式系统中各种低速外设之间的通信。而 SPI 是应用广泛的 4 线快速全双工串行通信接口。现在很多嵌入式系统都有 SPI 接口，和 I²C 外设连接有一定的困难。实现这种连接的方法之一是修改系统，但是性价比低。最好的方法是使用 CPLD 来桥接这两种接口。

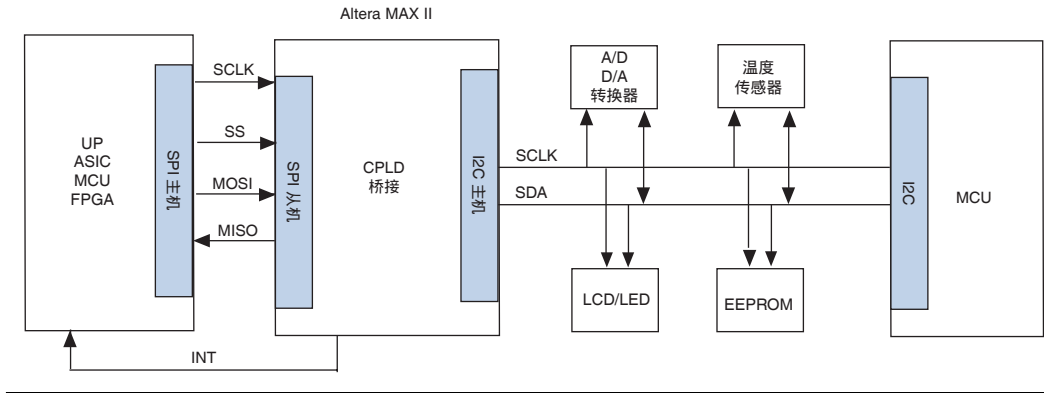
本设计使用 MAX II CPLD 来实现这一桥接功能，该器件非常灵活，功耗低，能够以较高的性价比集成到嵌入式系统中。对主机 (SPI 主机) 而言，CPLD 是 SPI 从机，而对于 I²C 总线，它是主机。

利用 MAX II CPLD 实现 SPI 至 I²C 的接口

本设计支持 SPI 接口主机对其他设备的数据流控制，这些设备包括 A/D 转换器、LED 控制器、音频处理器等，通过 I²C 接口来读取温度传感器、硬件监视器和诊断传感器等。

图 1 所示为采用 MAX II CPLD 来实现 SPI 至 I²C 的接口。它作为 SPI 从机和 SPI 主机进行接口，使用了四种信号线，控制用的 SS 和 SCLK 信号，数据用的 MISO 和 MOSI 信号。I²C 总线一侧接口有两条信号线，SCLK 和 SDA 信号。

图 1. 利用 MAX II CPLD 实现 SPI 至 I²C 的接口



SPI 接口

SPI 总线通常只有一个主机以及和它相连的多个从机。CPLD 作为从机之一和 SPI 主机设备相连，实现桥接功能。图 2 所示为 SPI 时序图。表 1 对 SPI 接口引脚进行了说明。

图 2. SPI 时序图

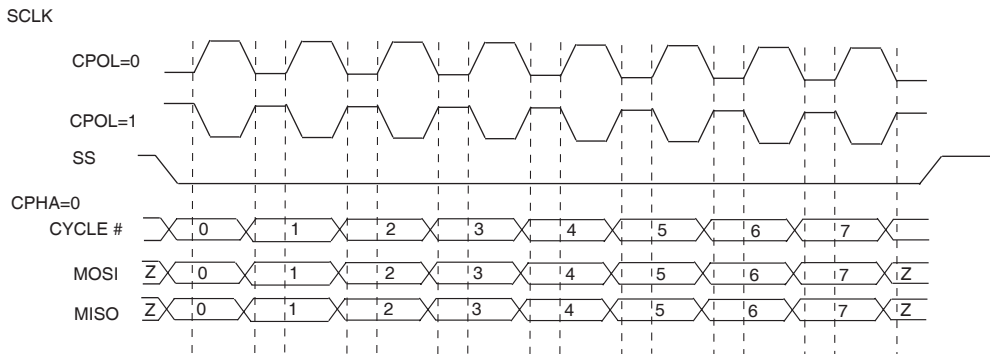


表 1. SPI 接口引脚说明

信号	目的	方向
SS	从机选择	输入 (低电平有效)
SCLK	SPI 时钟	输入
MISO	主机输入从机输出	输出
MOSI	主机输出从机输入	输出

SPI 主机发送:

- 命令寄存器 (8 位)
- 数据输入 (8 位)

SPI 主机接收:

- 状态寄存器 (8 位)
- 数据输出 (8 位)

SPI 字长度固定为 16 比特。如图 2 所示, CPOL = 0 和 CPHA = 1。在每一个 SPI 字中, 命令寄存器规定 I²C 总线上的功能, 数据输入保持 I²C 总线要发送的数据。类似的, 状态寄存器的最后一位是应答位, 数据输出是在前一 I²C 周期中通过 I²C 线接收到的数据。

在每一 SPI 总线的最后, 从机选择线变为高电平, 指示完成一个字的发送, 对于此时命令寄存器中的每个值, 执行一个 I²C 总线周期。经过固定时延后, 根据 I²C SCL 上的频率, 可以发送另一个 SPI 字。两个 SPI 字之间的最小时延是 I²C SCL 时钟频率。

I²C 接口

CPLD 作为 I²C 总线的主机，实现桥接功能。由于本设计的目的是提供 SPI 主机和 I²C 从机之间的接口，因此，I²C 总线不支持多个主机。表 2 对 I²C 接口引脚进行了说明。

信号	目的	方向
SCLK	I ² C 串行时钟	输出
SDA	I ² C 数据总线	双向

根据从 SPI 一侧接收到的命令寄存器值来执行 I²C 功能。表 3 列出了命令寄存器中存储的重要命令。

命令寄存器	I ² C 线的目的	寄存器中的数据
10000000	启动 / 重新启动	从机地址 + R/W
01000000	写入一个字节	要写入的数据
00100000	读取一个字节	不重要
00010000	停止	不重要
00000000	空，等待状态	不重要

某一 I²C 操作读入的数据存储在数据输出寄存器中，由 SPI 主机在下一 SPI 操作中读取。SPI 主机需要最后一个命令字 00000000 (b) 来读取状态值，而数据输出寄存器对 I²C 总线不作任何操作。I²C 命令格式如图 3 所示。

图 3. I²C 命令格式



□ 主机写操作 □ 从机写操作

设计实现

本设计可以采用 EPM240 或者其他 MAX II CPLD 来实现。实现过程涉及到使用设计源代码, 为 MAX II CPLD 的通用 I/O (GPIO) 线分配相应的信号和控制线等。在演示设计实现时, 还需要一个 SPI 主机和一个 I²C 从机。

以下步骤详细说明了这一设计演示, 演示用到了 MDN-B2 电路板, 以及采用了 Freescale SPIGen 软件 (以及对应的并口硬件) 的 PC 并口 SPI 环境和 I²C 从机电池量表模块 (由 MDN-B2 提供)。

表 4 列出了 EPM240G 的引脚分配。

表 4. EPM240G 引脚分配	
引脚分配	
信号	引脚
I2C_scl:	引脚 39
I2C_sda	引脚 40
SPI_cs	引脚 95
SPI_miso	引脚 91
SPI_mosi	引脚 92
SPI_clk	引脚 96



在 Quartus® II 软件的器件和引脚选项设置中, 将未使用的引脚分配为 **input tri-stated**。在引脚分配过程中, 在 Quartus II 软件的 I/O 标准栏中, 将 SPI_clk 分配为 2.5-V 施密特触发器输入。Quartus II 软件的分配编辑器被用于使能 I2C_scl 和 I2C_sda 引脚的 **Auto Open Drain**。打开相应的设置后, 对设计进行编译。

设计说明

请按照以下步骤, 在 MDN-B2 演示板上对设计进行演示:

1. 使用 PC 及其并口, 下载 SPI 仿真软件以建立 SPI 环境, 例如来自 Freescale 的 SPIGen。接受了 Freescale 的使用条款, 并免费注册后, 可免费下载该软件。



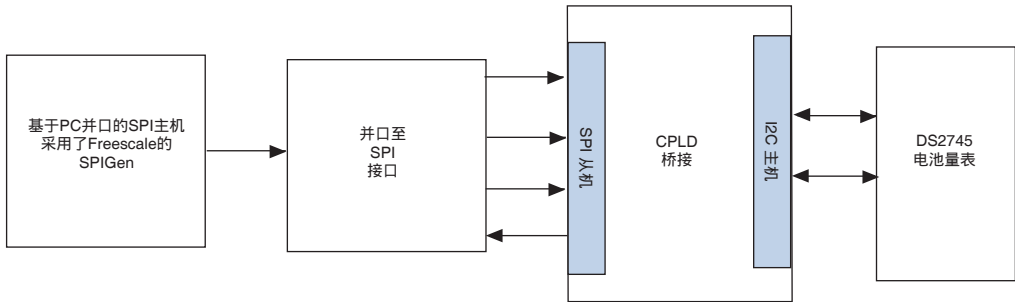
请访问下面的网址，下载 SPI 仿真软件：

http://www.freescale.com/files/soft_dev_tools/software/device_drivers/SPIGen.html

2. 完成注册后，下载并安装 SPIGen。您需要对它进行配置，以适应您的应用程序以及由 MDN-B2 演示板提供的并口至 SPI 软件狗。
3. 打开 SPIGen 软件，单击 **Configure** 菜单。选择 **Edit Configuration**。在 **General** 标签中，确定默认的 **Port Address** 是正确的 PC 并口地址。如果不是，进行正确的选择。您可以查看下面的设置，找到 PC 的并口地址：**Control Panel > System > Hardware > Device Manager > Ports > ECP Printer Port (LPT1) > Resources**。

图 4 所示为演示布局。

图 4. 演示布局



4. 选择16位格式，以十六进制(Intel格式)文件格式(.hex)来显示SPI值。
5. 按照以下步骤来配置 **SPI Pins** 标签：
 - a. 保持引脚分配的默认设置不变 (Chip Select: 2, Data In: 3, Data Out: 12, Clock: 4)。
 - b. 保持 **Bit Order** 的默认设置 **MSB is sent first** 不变。
 - c. 将 **Chip Select** 改为 **High when asserted**。
 - d. 将 **Data In** 和 **Data out** 改为 **Low = 1**。
 - e. 将 **SPI Type** 改为 **Type 4**。


- f. 单击 **Apply** 和 **OK**, 返回到 SPIGen 主屏幕。
6. 将并口至 SPI 软件狗和 PC 的并口相连。可以使用并口延长线以方便软件狗和演示板的连接。现在可以使用基于并口的 SPI 环境了。
 7. 打开演示板电源 (使用滑动开关 SW1)。通过演示板的 JTAG 插头 JP5 和普通编程电缆 (ByteBlaster™ II 或者 USB-Blaster™), 把设计下载到 MAX II CPLD 中。
 8. 在编程启动前以及启动过程中, 保持演示板上 SW4 的按下状态不变。完成后, 关断电源, 拔下 JTAG 连接器。
 9. 把 DS2745 I²C 电池量表模块 (由 MDN-B2 演示板提供) 安装到演示板的 JP3 上。确定模块的红色标记对准 JP3 的 pin#1。
 10. 将并口至 SPI 软件狗和演示板相连。
 11. 将软件狗尾端的 6 针连接器连接到演示板的 JP8 上。确定 6 针连接器的红色线对准 JP8 的 pin#2。
-  注意, JP8 是 10 x 2 引脚插头; 这一 6 针连接器只占用 JP8 的一个端口。
12. 使用滑动开关 SW1 打开演示板。
 13. 表 5 列出了在 SPIGen 软件中, 使用 **Word to Send (DI)** 域发送 .hex 数据的顺序。

表 5. 使用 SPIGen, Hex 格式的 SPI “Data to Send”

信号方向	命令 / 数据
发送 (DI)	80 90 (启动 + 从机地址)
发送 (DI)	40 0C (写入 + 存储器地址)
接收 (DO)	01 00 (ack)
发送 (DI)	80 91 (启动 + 从机地址)
接收 (DO)	01 00 (ack)
发送 (DI)	20 00 (读)
接收 (DO)	01 ?? (ack + MSB 数据)
发送 (DI)	00 00 (没有操作)
接收 (DO)	01 ?? (ack + LSB 数据)

14. 每次发送 SPI 数据时, 使用表 5 的 .hex 数据, 单击 **Send Once**。确定收到了应答数据。
15. 观察接收到的电池量表数据。收到的有两部分, 首先是 MSB 8 位数据, 然后是 LSB 3 位数据 (最后一行)。
16. 改变电池量表模块的黄色预置部分, 观察接收数据的变化。黄色预置部分改变了电池量表芯片 (Maxim DS2745) 的输入电压值, 该芯片通过其 I²C 接口, 以 11 位 2 进制兼容格式来提供电压读数。



如果需要了解详细信息, 请参考:

http://www.maxim-ic.com/quick_view2.cfm/qv_pk/4994

源代码

本文档所介绍的设计采用了 Verilog 来实现, 成功地运行在 MDN-B2 演示板上。下面的链接提供源代码、测试台文件以及完整的 Quartus II 工程:

http://www.altera.com/literature/an/an486_design_example.zip

结论

正如本设计所示, MAX II CPLD 是实现 SPI 和 I²C 等工业标准接口非常好的选择。其低功耗、易于上电以及内部振荡器等特性使其成为 SPI 至 I²C 等接口转换应用理想的可编程逻辑器件选择。

参考文档

本应用笔记参考了以下文档:

- http://www.altera.com/literature/an/an486_design_example.zip
- http://www.freescale.com/files/soft_dev_tools/software/device_drivers/SPIGen.html
- http://www.maxim-ic.com/quick_view2.cfm/qv_pk/4994

其他资源

以下是本应用笔记的其他资源:

- MAX II CPLD 主页:
<http://www.altera.com/products/devices/cpld/max2/mx2-index.jsp>
- MAX II 器件资料:
<http://www.altera.com/literature/lit-max2.jsp>

- MAX II 关断设计：
<http://www.altera.com/support/examples/max/exm-power-down.html>
- MAX II 应用笔记：
AN 428：MAX II CPLD 设计指南
AN 422：利用 MAX II CPLD 实现便携式系统的功耗管理

文档版本历史

表 6 列出了本应用笔记的版本历史。

表 6. 文档版本历史		
日期和文档版本	进行的改动	对改动的总结
2007 年 12 月, 1.0 版	初次发布	—



101 Innovation Drive
San Jose, CA 95134
www.altera.com
Technical Support:
www.altera.com/support
Literature Services:
literature@altera.com

版权 © 2007 Altera 公司。保留所有版权。Altera、可编程解决方案公司、程式化 Altera 标识、专用器件名称和其他所有其他专有商标或者服务标记，除非特别声明，均为 Altera 公司在美国和其他国家的商标和服务标记。所有其他产品或者服务名称的所有权属于其各自持有人。Altera 产品受美国和其他国家多种专利、未决应用、模板著作权和版权的保护。Altera 保证当前规范下的半导体产品性能与 Altera 标准质保一致，但是保留对产品和服务在没有事先通知时的升级变更权利。除非与 Altera 公司的书面条款完全一致，否则 Altera 不承担由此处所述信息、产品或者服务导致的责任。Altera 建议客户在决定购买产品或者服务，以及确信任何公开信息之前，阅读 Altera 最新版的器件规范说明。

