

PS/2 设备数据手册



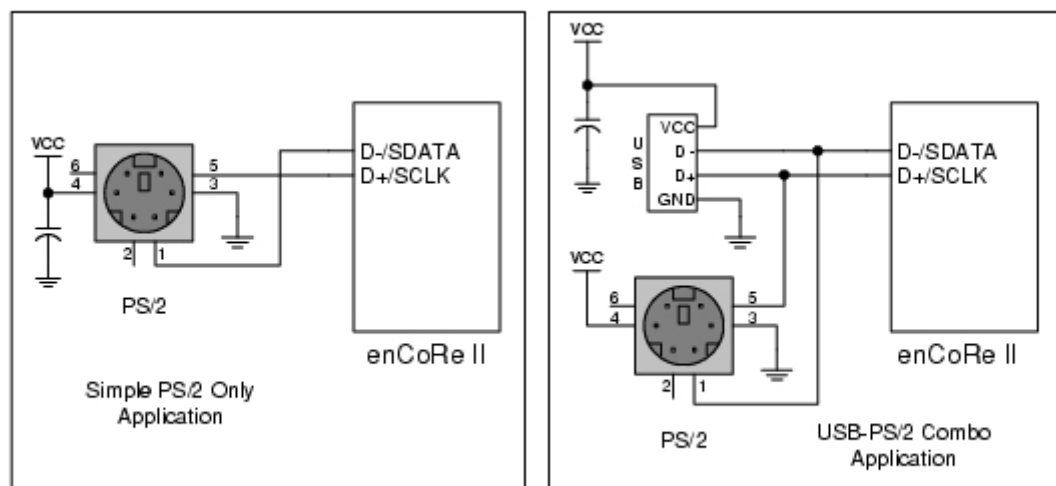
PS2D vX.Y

赛普拉斯半导体公司 2004-2007 年版权所有。所有权利均予保留。

资源	必需	选用
CY7C639/638/633/602/601xx		
鼠标支持		
内存	闪存: 935 个字节 RAM: 17 个字节, 包括 4 个字节的传输缓存器	
键盘支持		
内存	闪存: 810 个字节 RAM: 17 个字节, 包括 4 个字节的传输缓存器	

特征和概述

- PS/2 设备接口
- 鼠标或键盘应用程序可选择的命令集
- 客户特有解锁机制
- 集成了 USB 串行接口引擎 (SIE) 以实现兼容 USB-PS/2 应用程序



原理方框图

功能说明

PS/2 设备用户模块支持一个标准的 PS/2 键盘或一个标准的 PS/2 鼠标。PS/2 接头为 6 针的 mini-DIN 式接头。本模块可以用于将一个键盘或一个鼠标连接至兼容的计算机。用户必须选择用 PS/2 设备用户模块来支持一个键盘或一个鼠标。在用户选择本用户模块时，将要选择支持键盘命令集或鼠标命令集。如果用户希望在选择用户模块改换成另一种，则可以在用户模块图标上右键单击，然后选择“**用户模块选择选项**”。按照惯例，支持键盘的接头采用紫色代码色，而支持鼠标的接口采用绿色代码色。在加入 USB 设备用户模块后，用户可以创建一个支持一个 USB 和 PS/2 组合设备的单一设备。

数据格式

主机至设备数据格式（12 位）：

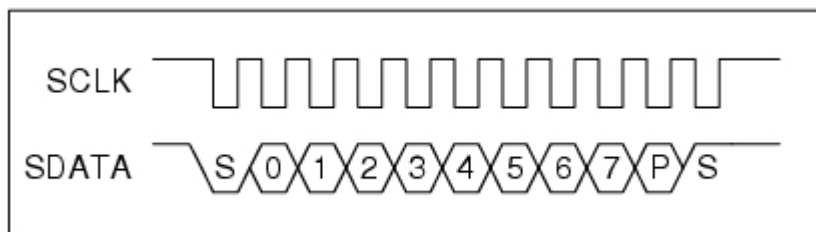
- 请求发送 (0)
- 8 个数据位。
- 奇偶校验位。（奇校验）
- 停止位。（1）
- 确认。（1）

设备至主机数据格式（11 位）：

- 停止位。（0）
- 8 个数据位。
- 奇偶校验位。（奇校验）
- 停止位。（1）

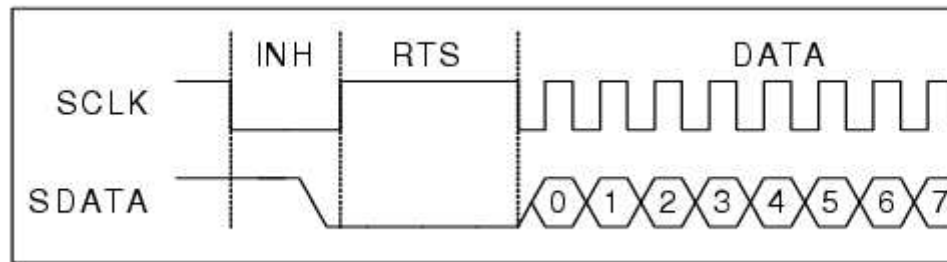
时序

以下时序图描述了为了正确实现 PS/2 数据发送和接收的 PS/2 时序要求。PS/2 设备负责在 SCLK 上生成同时用于主机至设备和设备至主机传输的全部时钟信号。



PS/2 设备至主机传输时序

在设备至主机传输期间，设备同时控制 SCLK 和 SDATA。设备保持着一个稳定在 10KHz 与 14.6KHz 之间的 SCLK 周期。本设备在 SCLK 为高时更新 SDATA，而且在 SCLK 为低时更新主机样本 SDATA。



PS/2 主机至设备传输时序

在主机至设备传输期间，存在 4 个界限分明的阶段。首先，主机将 **SCLK** 电平拉低，表示一种“主机禁止”（Host Inhibit）状况。在禁止期间，设备必须停止正在进行中的任何传输，并释放 **SCLK** 和 **SDATA**。下一步，主机将 **SDATA** 电平拉低，然后释放 **SCLK**，向设备发出一个“请求发送”信号。在数据传输阶段，设备循环操作 **SCLK** 电平。主机在 **SCLK** 为低电平时更新 **SDATA**。设备在 **SCLK** 为高电平时对 **SDATA** 取样。奇偶校验位的下降沿开始于“确认阶段”。主机释放 **SDATA**，构成停止位。在 **SDATA** 采样后，设备通过将 **SDATA** 拉到低电平来确认收到数据。主机在 **SCLK** 为低时对 **SDATA** 进行采样以进行确认。

参数和资源

TxBufferSize

此参数确定了发送缓存区内保留了多少个 **RAM** 字节。发送缓存区必须能够容纳应用程序所产生的最大的数据包。

应用程序编程接口

本节所说明的应用程序编程接口（API）子程序允许对 **PS/2** 用户模块进行编程控制。以下表格列出了基本的和设备特有的 API 函数。

备注：在使用本 **PS2D** 用户模块和来自端口 **n**（**Pn.2 – Pn.7**）的通用输入输出（**GPIO**）引脚时，应用程序应当使用 **Port_n_Data_SHADE** 影子寄存器来确保稳定一致的数据处理。在汇编语言中，可以直接访问 **Port_n_Data_SHADE** 的 **RAM** 位置。在 **C** 语言中，应当包括一个外部引用：

```
extern BYTE Port_n_Data_SHADE; // Where n is the port number
```

基本 PS/2 设备 API	
函数	说明
void PS2D_Start(void)	启用用户模块
void PS2D_Stop(void)	禁用用户模块。
BYTE PS2D_DoCommand(void)	如果主机请求发送 PS/2 主机命令，则执行接收和处理。此函数应当至少每隔 1-2

	毫秒调用一次。
void PS2D_SendNextByte(void)	发送处于发送缓存区内的下一个字节。
BYTE PS2D_TranserInProgress(void)	如果向 PS/2 主机的传输正在进行中，则返回 1，否则返回 0
void PS2D_Resend(void)	重新发送 PS/2 发送缓存区的内容。
void PS2D_AbortTransfer(void)	中止当前传输。

设备特有 API (鼠标)

函数	说明
BYTE PS2D_GetDeviceID(void);	返回设备 ID。PS2D 支持 2 个鼠标设备 ID。 设备 ID 0 作为默认鼠标设备 ID 返回。主机预计设备 ID 0 采用 1 个 3 字节的鼠标数据包。 针对滚轮型鼠标返回设备 ID 3。这项特色通过来自主机的一系列设置分辨率命令 (200, 100, 80) 来自动激活。主机预计设备 ID 3 采用 1 个 4 字节的鼠标数据包。
BYTE PS2D_IsEnabled(void)	如果主机用“启用数据报告”项启用了数据报告功能，则返回 1，否则返回 0。
BYTE PS2D_IsStreamMode(void);	如果接口处于流传输模式下，则返回 1，否则返回 0。
BYTE PS2D_GetSampleInterval(void);	返回采样间隔时间（毫秒）。

设备特有 API (键盘)

函数	说明
WORD PS2D_GetDeviceID(void);	返回设备 ID。
BYTE PS2D_IsEnabled(void)	如果主机已经启用了键盘，则返回 1，否则返回 0。
BYTE PS2D_GetScanCodeSet(void);	返回由 PS/2 主机所选定的当前扫描代码集。
BYTE PS2D_GetRepeatRate(void);	返回由 PS/2 主机所选定的当前击键重复率。
BYTE PS2D_GetDelay(void);	返回由 PS/2 主机所选定的当前击键延时。
BYTE PS2D_GetLED(void);	返回由 PS/2 主机所选定的当前 LED 设置。

PS2D_Start

说明：

执行 PS/2 设备用户模块所需的全部初始化。

C 语言原型：

```
void PS2D_Start(void)
```

汇编：

```
call PS2D_Start
```

参数:

无

返回值:

无

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_Stop

说明:

执行 PS/2 用户模块所需要的全部必要的关闭任务。此函数应当在实际接口决定将连接至 USB 主机时，在 PS/2-USB 组合应用程序中调用。

C 语言原型:

```
void PS2D_Stop(void)
```

汇编语言

```
call PS2D_Stop
```

参数:

无

返回值:

无

注意:

寄存器 A 和 X 有可能被此函数所改变。

PS2D_DoCommand

说明:

从 PS/2 主机接收和处理命令

C 语言原型:

```
BYTE PS2D_DoCommand(void)
```

汇编语言:

```
call PS2D_DoCommand
```

参数:

无

返回值:

如果 PS/2 命令已经接收并处理了，则返回一个非零数值。

注意:

寄存器 A 和 X 有可能被此函数所改变。

PS2D_SendNextByte

说明:

将发送缓存区内的下一个字节发送至 PS/2 主机。

C 语言原型:

```
void PS2D_SendNextByte(void)
```

汇编语言:

```
call PS2D_SendNextByte
```

参数:

无

返回值:

无

注意:

寄存器 A 和 X 有可能被此函数所改变。

PS2D_TransferInProgress

说明:

确定是否 PS/2 传输正在进行中。

C 语言原型:

```
BYTE PS2D_TranserInProgress(void)
```

汇编语言:

```
call PS2D_TranserInProgress
```

参数:

无

返回值:

如果 PS/2 传输正在进行中，则返回非零数值。

注意:

寄存器 A 和 X 有可能被此函数所改变。

PS2D_Resend

说明:

重新发送 PS/2 发送缓存区的内容。

C 语言原型:

```
void PS2D_RESEND(void)
```

汇编语言:

```
call PS2D_RESEND
```

参数:

无

返回值:

无

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_AbortTransfer

说明:

中止当前传输.

C 语言原型:

```
void PS2D_AbortTransfer(void)
```

汇编语言:

```
call PS2D_AbortTransfer
```

参数:

无

返回值:

无

注意:

寄存器 A 和 X 有可能被此函数所改变.

设备特有 API (鼠标)

PS2D_GetDeviceID

说明:

返回设备 ID.

C 语言原型:

```
BYTE PS2D_GetDeviceID(void)
```

汇编语言:

```
call PS2D_GetDeviceID
```

参数:

无

返回值:

设备 ID

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_IsEnabled

说明:

如果接口被启用则返回 1，否则返回 0.

C 语言原型:

BYTE PS2D_IsEnabled(void)

汇编语言:

call PS2D_IsEnabled

参数:

无

返回值:

无

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_IsStreamMode

说明:

如果接口处于流传输模式下，则返回 1，否则返回 0.

C 语言原型:

BYTE PS2D_IsStreamMode(void)

汇编语言:

call PS2D_IsStreamMode

参数:

无

返回值:

流模式

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_GetSampleInterval

说明:

返回由 PS/2 主机所选定的采样间隔时间（毫秒）.

C 语言原型:

BYTE PS2D_GetSampleInterval(void)

汇编语言:

call PS2D_GetSampleInterval

参数:

无

返回值:

采样间隔时间

注意:

寄存器 A 和 X 有可能被此函数所改变.

设备特有 API (键盘)

PS2D_GetDeviceID

说明:

返回键盘的 PS/2 ID.

C 语言原型:

```
WORD KB_GetDeviceID(void)
```

汇编语言:

```
call KB_GetDeviceID
```

参数:

无

返回值:

用于键盘的 PS/2 ID (0xAB83)

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_IsEnabled

说明:

如果接口被启用则返回 1, 否则返回 0.

C 语言原型:

```
BYTE PS2D_IsEnabled(void)
```

汇编语言:

```
call PS2D_IsEnabled
```

参数:

无

返回值:

无

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_GetScanCodeSet

说明:

返回由 PS/2 主机所选定的扫描代码集

C 语言原型:

BYTE KB_GetScanCodeSet(void)

汇编语言:

call KB_GetScanCodeSet

参数:

无

返回值:

扫描代码集

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_GetRepeatRate

说明:

返回由 PS/2 主机所选定的击键率.

C 语言原型:

BYTE KB_GetRepeatRate(void)

汇编语言:

call KB_GetRepeatRate

参数:

无

返回值:

重复率

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_GetDelay

说明:

返回由 PS/2 主机所选定的击键延时.

C 语言原型:

BYTE KB_GetDelay(void)

汇编语言:

call KB_GetDelay

参数:

无

返回值:

击键延时

注意:

寄存器 A 和 X 有可能被此函数所改变.

PS2D_GetLED

说明:

返回由 PS/2 主机所设定的当前 LED 设置。

C 语言原型:

```
BYTE KB_GetLED(void)
```

汇编语言:

```
call KB_GetLED
```

参数:

无

返回值:

LED Setting Bitmap.

Bit 0 Scroll Lock (0=Off, 1=On)

Bit 1 Num Lock (0=Off, 1=On)

Bit 2 Caps Lock (0=Off, 1=On)

注意:

寄存器 A 和 X 有可能被此函数所改变.

应用数据传输 API

出于效率方面的考虑, PS2D 用户模块没有提供一个用于载入传输缓存区的函数接口。而是由应用程序直接载入传输缓存区。这项应用负责确保在通过调用 PS2D_TranserInProgress 进程载入传输缓存区前没有正在执行的传输。应用程序必须使用 PS2D_TX_BUFFER_SIZE 来确保不会发生缓存区超出限度的现象。在缓存区载入后, 通过调用 PS2D_StartTransfer 宏指令并指定传输大小为唯一参数, 即可开始传输操作。

PS/2 低层驱动程序

PS2D 用户模块低层驱动程序能够处理应用程序与 PS/2 主机之间的所有数据传输。本驱动程序通过定时调用 PS2D_DoCommand 和 PS2D_SendNextByte 来调用。PS2D_DoCommand 必须至少每隔 5 毫秒调用一次。在 PS2D_TransferInProgress 指示正在执行传输操作时, PS2D_SendNextByte 应当在不超过 2 毫秒的时间间隔下进行调用。

为了确保对主机指令做出正确的响应, 除非前面对 PS2D_DoCommand 的调用表明没有得到处理的指令, 应用程序不得启用数据传输。某些指令要求针对某些特定的指令参数进行额外的数据交换。当然, 应用程序不得在当前传输正在进行时启动一次新的数据传输。

鼠标应用程序支持

下列表格显示了一个可能发送至鼠标的指令清单。在发送任何其它指令前，先检查鼠标的状态，如果鼠标处于流模式下，则主机应当禁用数据报告功能。命令 **0xF5** 可以用于禁用数据报告功能。

鼠标指令集		
指令	说明	S=支持 U=不支持
0xFF (复位)	鼠标用“确认”（ 0xFA ）来回应此指令，然后进入复位模式。	S
0xFE (重新发送)	在主机从鼠标收到无效数据时，主机发送此指令。鼠标以重新发送自己已发送给主机的最后一个数据包来做出响应。如果鼠标用另一个无效数据包来对“重新发送”指令做出响应，则主机既可以发送另一个“重新发送”指令，也可以发送一条“错误”指令，循环开关鼠标电源以复位鼠标，或可以禁止通讯（通过将时钟控制线变成低电平）。此项操作的采取取决于主机。	S
0xF6 (设置默认值)	如果鼠标以“确认”（ 0xFA ）做出响应，则随后载入下列数值：采样率 = 100，分辨率 = 4 次计数/毫米，比例 = 1.1，禁用数据报告。鼠标随后复位自己的移动计数器并进入流模式。	S
0xF5 (禁用数据报告)	鼠标用“确认”（ 0xFA ）做出响应，然后禁用数据报告功能并对自己的移动计数器复位。这样将只影响到流模式下的数据报告而不会禁用采样。禁用流模式与远程模式的功能相同。	S
0xF4 (启用数据报告)	鼠标用“确认”（ 0xFA ）做出响应，然后启用数据报告功能并对自己的移动计数器复位。此条指令可以在鼠标处于远程模式（或流模式）下发出，但将只影响流模式下的数据报告功能。	S
0xF3 (设置采样率)	鼠标以“确认”（ 0xFA ）做出确认，然后再从主机中多读取一个字节。鼠标保存此字节作为新采样率。在收到采样率后，鼠标再次用“确认”（ 0xFA ）做出响应，并复位自己的移动计数器。有效的采样率为 10、20、40、60、80、100 和 200 次采样/秒。	S
0xF2 (获取设备 ID)	鼠标以“确认”（ 0xFA ）做出响应，随后是自己的设备 ID（ 0x00 适用于标准的 PS/2 鼠标）鼠标还应当复位自己的移动计数器。	S
0xF0 (设置远程模式)	鼠标以“确认”（ 0xFA ）做出响应，然后复位自己的移动计数器，再进入远程模式。	S
0xEE (设置轮询模式)	鼠标以“确认”（ 0xFA ）做出响应，然后复位自己的移动计数器，再进入轮询模式。	S
0xEC (复位轮询模式)	鼠标用“确认”（ 0xFA ）做出响应，然后复位自己的移动计数器，然后进入它在轮询模式以前所处的模式（流模式或远程模式）。	S

询模式)		
0xEB (读取数据)	鼠标以“确认”(0xFA)做出响应, 然后发送一个移动数据包。这是从远程模式下读取数据的唯一方法。在数据包已经成功发送后, 鼠标将复位自己的移动计数器。	S
0xEA (设置流模式)	鼠标以“确认”做出响应, 然后复位自己的移动计数器, 再进入流模式。	S
0xE9 (状态请求)	鼠标以“确认”做出响应, 然后发送以下的 3 字节状态数据包 (然后复位自己的移动计数器)	S
0xE8 (设置分辨率)	鼠标以“确认”(0xFA)做出响应, 然后从主机读取 1 个字节, 并再次以“确认”(0xFA)做出响应, 然后复位自己的移动计数器。来自主机的数据决定了分辨率。	S
0xE7 (设置比例 2:1)	鼠标以“确认”(0xFA)做出响应, 然后启用 2:1 比例。	S
0xE6 (设置比例 1:1)	鼠标以“确认”(0xFA)做出响应, 然后启用 1:1 比例。	S

键盘应用程序支持

下列表格显示了一个主机可能发送给键盘的所有指令的清单:

键盘指令集		
指令	说明	S=支持 U=不支持
0xFF (复位)	键盘以“确认”(0xFA)做出响应, 然后进入“复位”模式。	S
0xFE (重新发送)	键盘以重新发送最后一次发送的字节做出响应。这种响应的例外情况是在最后一次发送的字节为“重新发出”(0xFE)。如果情况如此, 则键盘重新发送最后 1 个非 0xFE 字节。本指令被主机用于指示接收中的错误。	S
0xFD (设定键按下)	禁用指定按键的断码和击键重复功能。键盘以“确认”(0xFA)做出响应, 然后禁用扫描 (如果已启用), 然后从主机读取一个按键列表。这些按键将用其第 3 套通码来指定。键盘用“确认”对每一个通	U

通码)	码做出响应。主机通过发送 1 个无效的第 3 套通码（例如，1 个有效的指令）来结束这个清单。键盘随后重新启用扫描功能（如果以前禁用了）。	
0xFC (设置键按下通码/断码)	与“设置键按下通码”指令类似，例外是此指令只禁用击键重复功能。	U
0xFB (设置键按下击键参数)	与“设置键按下通码”和“设置键按下通码/断码”指令类似，例外是此指令只禁用断码。	U
0xFA (设置全部键按下参数/通码/断码)	键盘用“确认”(0xFA)做出响应。设置所有按键至自己正常设置（生成有关通、断和击键重复的扫描码）	U
0xF9 (设置所有按键通码)	键盘用“确认”(0xFA)做出响应。类似于 0xFD 指令，只是适用于全部按键。	U
0xF8 (设置所有按键通码/断码)	键盘用“确认”(0xFA)做出响应。类似于 0xFC 指令，只是适用于全部按键。	U
0xF7 (设置所有按键的击键参数)	键盘用“确认”(0xFA)做出响应。类似于 0xFB 指令，只是适用于全部按键。	U
0xF6 (设置默认值)	载入默认的击键率/延时（10.9cps/500 毫秒）、键类型（所有击键/通/断）以及扫描码集。	S
0xF5 (禁用)	键盘停止扫描，载入默认数值，然后等待下一步的指令。	S
0xF4 (启用)	在用前一条指令禁用后重新启用键盘。	S
0xF3 (设置击键率/延时)	主机用于一个单参数字节来跟在此指令后，此字节用于定义击键率和延时。	S
0xF2 (读取 ID)	键盘通过发送 1 个 2 字节的设备 ID 0xAB、0x83 来做出响应。(0xAB 首先发送，然后发送 0x83.)	S
0xF0 (设置扫描码集)	键盘用“确认”做出响应，然后从主机读取参数字节。此参数字节可以是 0x01、0x02 或 0x03，以分别选择扫描代码集 1、2 或 3。键盘用“确认”做出对此参数字节的响应。如果此参数字节为 0x00，则键盘用“确认”做出响应，后跟当前的扫描码集。	S

0xEE (回送)	键盘用“回送”（0xEE）做出响应.	S
0xED (置位/复位 LED)	主机用 1 个指定了键盘的 Num Lock、Caps Lock 和 Scroll Lock LED 的状态的单参数字节跟在此指令之后.	S

赛普拉斯半导体公司

公司地址: 198 Champion Court

San Jose, CA 95134-1709

电话: 408-943-2600

传真: 408-943-4730

应用支持热线: 425.787.4814

网址: <http://www.cypress.com>

©2007 赛普拉斯半导体公司, 版权所有。此处包含的信息如有更改, 恕不另行通知。赛普拉斯半导体公司对使用任何其他非赛普拉斯产品中所包含的电路不承担任何责任。也不对任何受专利或其他权利保护的许可作任何明示或暗示。除非与赛普拉斯公司签订明确的书面协议, 否则不得将赛普拉斯产品用于医疗、生命支持、临界控制或者安全应用, 赛普拉斯公司对此使用不作任何担保。此外, 赛普拉斯并未授权其产品作为关键的零部件用于生命支持系统, 因为其产品故障或失灵可能会给用户带来严重伤害。若将赛普拉斯产品用于生命支持系统应用之中, 则表示该系统的厂商应承担因使用该产品所带来的所有风险, 并赔偿赛普拉斯因此受到的任何损失。

PSoC Designer™、Programmable System-on-Chip™ 以及 PSoC Express™ 均为商标, 并且 PSoC® 是赛普拉斯半导体公司的注册商标。此处所提到的所有其他商标或注册商标均为其各自的公司所有。

所有源代码(软件和/或固件)均归赛普拉斯半导体公司所有, 并在世界范围内受专利保护(美国 and 国外)和美国版权法以及国际条约规定条款的保护。为了创建定制软件和/或创建仅用于和可适用协议中所指定的赛普拉斯集成电路的许可证产品固件, 赛普拉斯在此授予被授予方一个私人的、非专有并且不可转让的许可证以用于复制、使用、修改、创作其衍生作品和编译赛普拉斯源代码及其衍生作品。如果没有赛普拉斯半导体公司明确的书面协议, 除非上面另有规定, 否则禁止对源代码进行复制、修改、翻译、编译和展示。

免责声明: 赛普拉斯对本材料不提供任何担保, 无论是明示的或暗示的, 包括但不限于对适销性或适用某种特殊目的的默示担保。赛普拉斯保留对这里所描述的材料做出更改而不另行通知的权利。赛普拉斯对因具体应用、使用任何产品或此处所描述的电路而造成的任何后果不承担任何责任。赛普拉斯并未授权其产品作为关键的零部件用于生命支持系统, 因为其产品故障或失灵可能会给用户带来严重伤害。若将赛普拉斯产品用于生命支持系统应用之中, 则表示该系统的厂商应承担因使用该产品所带来的所有风险, 并赔偿赛普拉斯因此受到的任何损失。

具体使用应当依照可适用的赛普拉斯软件许可协议并受其限制。

© Cypress Semiconductor Corporation, 2007. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or

failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC Designer™, Programmable System-on-Chip™, and PSoC Express™ are trademarks and PSoC® is a registered trademark of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

Use may be limited by and subject to the applicable Cypress software license agreement.