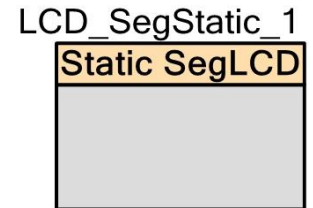


静态段 LCD (LCD_SegStatic)

2.30

特性

- 1 到 61 像素或符号
- 10 至 150 Hz 的刷新率
- 用户定义的像素或符号图，可使用 7 段、14 段、16 段以及条形图计算子程序
- 直接驱动静态（一条共用线路）LCD



概述

静态段 LCD (LCD_SegStat) 组件可直接驱动 3.3 V 和 5.0 V 的 LCD 显示屏。此组件为您的自定义或标准显示屏提供了一个简单的 PSoC 器件配置方法。

每个 LCD 像素/符号都可以打开或者关闭。静态段 LCD 组件也提供高级支持，以简化显示屏中的以下显示结构类型：

- 7 段数字
- 14 段字母数字
- 16 段字母数字
- 1 到 255 个元素的条形图

何时使用静态段 LCD

在静态模式下（对于仅有一条共用线路的显示屏配置），如果需要直接驱动 3.3 V 或 5.0 V 的 LCD 显示屏，请使用静态段驱动 LCD 组件。静态段 LCD 组件不需要目标 PSoC 器件提供 LCD 驱动硬件资源。组件可用于任何有足够引脚支持所需共用线路和段线路数的目标芯片。

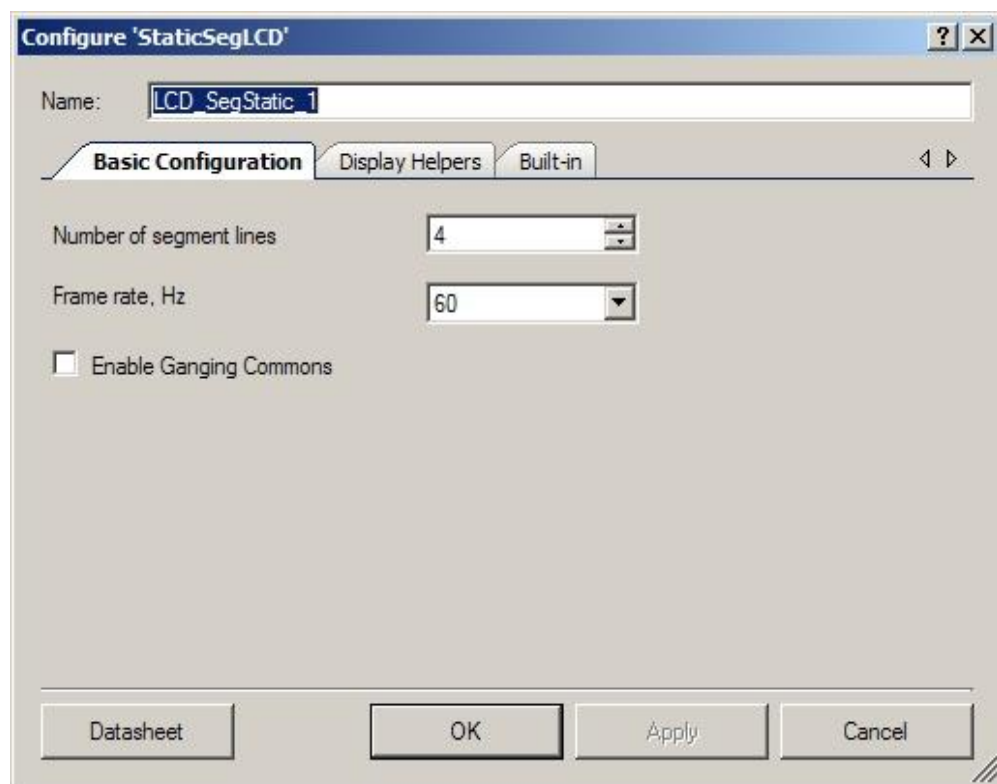
输入/输出连接

在原理图画布上没有可见的组件连接；然而，使用“设计范围资源”中的引脚编辑器，可以将各种信号连接至引脚。

组件参数

将一个静态段 LCD 组件拖放到您的设计上，并双击它以打开配置对话框。

基本配置选项卡



Number of segment lines (段线路数)

对用户定义的显示屏中的段线路数进行定义。默认值为 4。

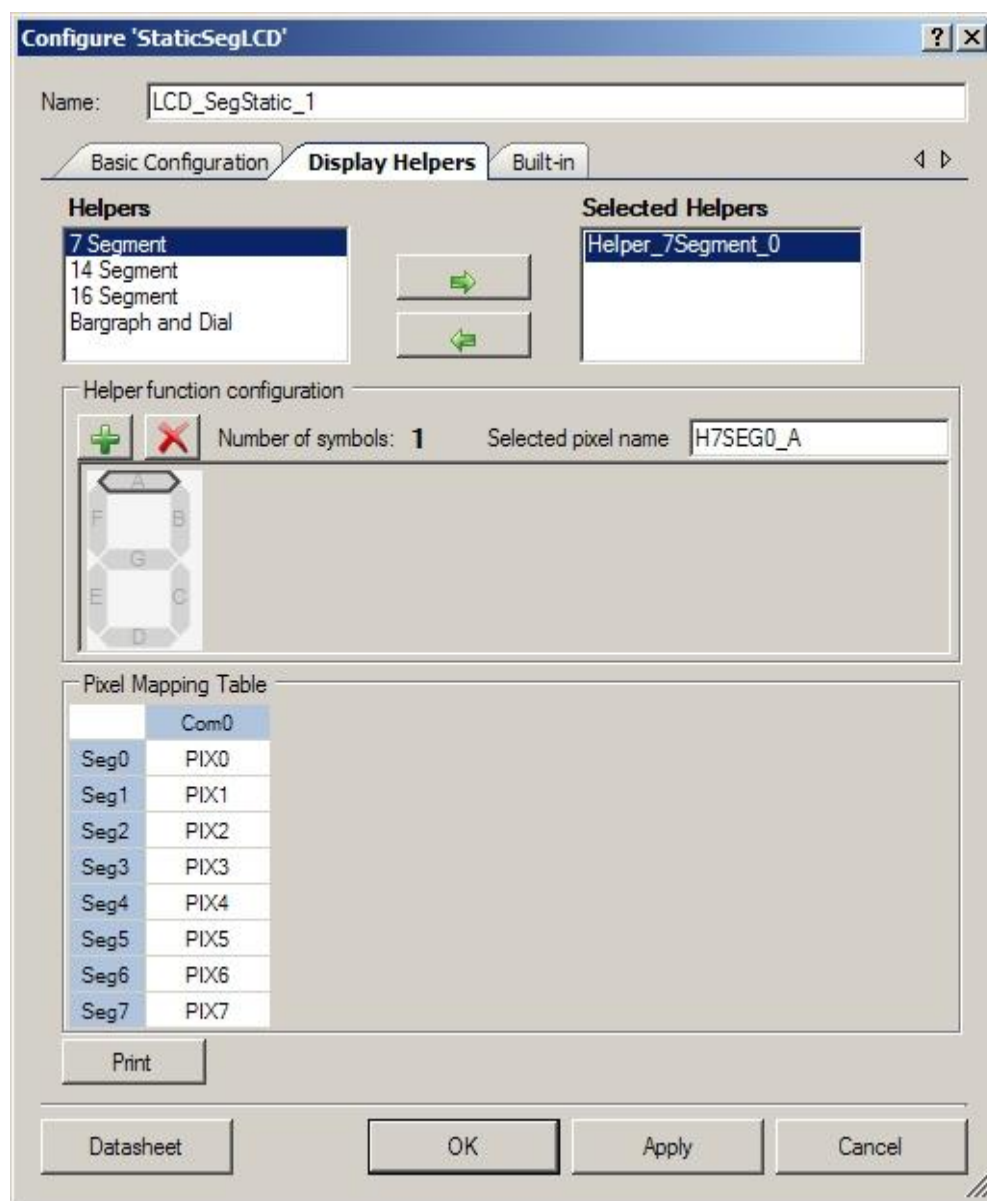
Frame rate, Hz (帧率, Hz)

此参数确定显示屏的刷新率。可能值为 10 Hz 到 150 Hz。默认值为 60 Hz。

Enable Ganging Commons (使能组合共用连接)

选择此复选框以连接 PSoC 引脚，以驱动共用信号。使能该项后，将分配两个 PSoC 引脚用于共用信号。它用来驱动较大显示器。

显示助手选项卡



Display Helpers（显示助手）允许您配置一组共同使用显示屏段，作为多个预定义显示元素类型之一：

- 7、14 或 16 段显示
- 线或圆条形图显示器

基于字符的显示助手可用于组合多个显示符号，从而创建多字符显示器元件。

Helpers / Selected Helpers (助手/选定助手)

可以将一个或多个助手添加到 **Selected Helpers** (选定助手) 列表中，其方法是在 **Helpers** (助手) 列表中选择所需助手类型，然后单击右箭头按钮。如果没有足够的引脚来支持新助手，则不会添加该助手。要删除某个助手，在 **Selected Helpers** 列表中选中它，然后单击左箭头按钮。

选定助手在列表中显示的顺序很重要。默认情况下，添加到 **Selected Helpers** 列表的给定类型的第一个助手的名称带后缀 **0**，同类型的下一个助手名称带后缀 **1**，以此类推。如果已从列表中移除了选定助手，将不会重新命名剩下的助手。添加助手时，名称将使用最低有效的后缀。

为每个助手提供了 API。有关详细信息，请参见[应用编程接口](#)一节中的内容。

- **7 Segment Helper (7 段助手)** — 该助手长度可能为 **1** 到 **5** 位，可显示十六进制位 **0** 到 **F**，或十进制 **16** 位无符号整数 (uint16) 值。助手功能不支持小数点。



- **14 Segment Helper (14 段助手)** — 该助手长度可高达 **20** 个字符。它会显示单个 ASCII 字符或空结尾的字符串。可能值是标准的 ASCII 可打印字符 (编码范围为 **0** 到 **127**)。



- **16 Segment Helper (16 段助手)** — 此助手长度可高达 **20** 个字符。它可以显示为单一的 ASCII 字符或空结尾的全字符串。可能值是标准的 ASCII 字符和扩展编码表 (编码范围为 **0** 到 **255**)。不提供扩展编码表。



- **Bargraph and Dial Helper (条形图和拨号助手)** — 这些助手用于 **1** 到 **255** 段条形图和拨号指示符。条形图是单个选择的像素或选定像素及指定像素左边或右边的所有像素



Helper function configuration (助手功能配置)

对话框的此部分允许您配置助手，包括向/从助手中添加或移除符号，以及命名像素。

从 **Selected Helpers** 列表选定助手。

单击[+]或[x]按键，为选定助手添加或移除符号。可添加的最大符号数取决于助手类型以及组件支持的像素总数。如果可用的引脚数量不足以支持新符号，则不再添加新符号。

要重新命名作为助手功能组成部分的像素，请在 **Helper function configuration** 显示屏中，选择符号图像上的像素。当前名称将显示在选定像素名称字段中，并能够按需要进行修改。

Pixel Naming (像素命名)

默认像素名称的格式为“PIX#”，其中“#”是 **Pixel Mapping Table** (像素映射表) 右上角以递增顺序排列的像素数。

像素（与助手符号相关）的默认命名方式具有不同格式。默认名称由前缀部分、所有像素的通用的符号部分和唯一段标识符组成。默认前缀表示助手类型和符号实例。例如，7 段显示助手中某符号中的像素默认名称可能是“H7SEG4_A”，其中：

H7 指示 7 段助手的像素部分

SEG4 指示指定为 7 段符号项目第 4 个符号的部分像素

A 指示 7 段符号中的唯一段

对于默认的像素名称，只有像素名称的唯一部分显示在符号图像中。如果修改了像素名称，则即使修改前和修改后的名称都使用了共同的前缀，符号图像上也将显示整个名称。

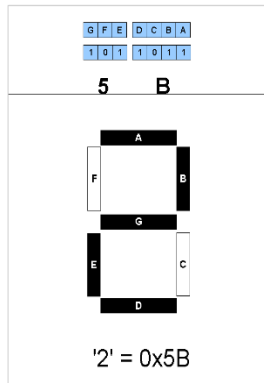
注意：所有像素名称必须是唯一的。

当将助手函数符号元素分配给 **Pixel Mapping Table** (如下所述) 中的一个像素时，此像素使用助手符号元素的名称。助手符号元素名称取代默认像素名称，但不会替换该名称。您不可重新使用与助手函数相关联的像素的默认名称。

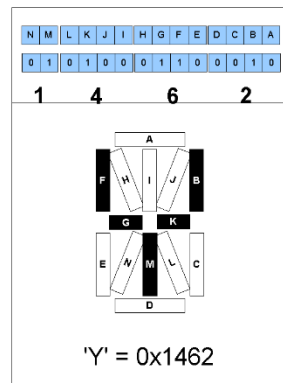
字符编码

所有高级助手 APIs 都有自己的查找表。该表包含一组编码像素状态，这构成特定的字符映像。下列示例说明了如何对特定字符进行编码（段名称与自定义程序中显示的名称不同）。

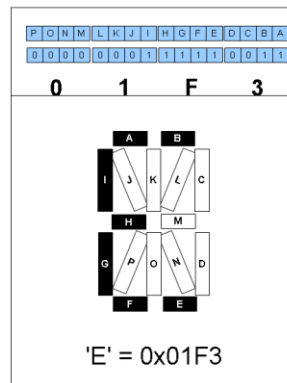
7段编码



14段编码



16段编码



Pixel Mapping Table (像素映射表)

该 **Pixel Mapping Table** (像素映射表) 是帧缓冲区的展示。为了使 **API** 函数正确地工作, **Helper function configuration** 的每一个像素必须分配给 **Pixel Mapping Table** 中的像素位置。有关进行正确分配所需的信息, 请参见你的 **LCD** 显示屏的数据手册。

要分配像素, 在 **Helper function configuration** 面板中选择所需像素, 然后将其拖入 **Pixel Mapping Table** 中的正确位置。

通过在表显示屏上双击像素, 然后输入所需名称, 可以重新命名**像素映射表**中的像素。可以使用此方法命名与其中某个可用助手类型关联的像素。

打印按钮用于打印像素映射表。

时钟选择

静态段 **LCD** 组件使用了一个内部时钟, 因此无需外部时钟。一旦安装了组件, 时钟将自动专用于 **LCD** 组件。时钟生成帧率频率。

应用编程接口

通过应用编程接口 (API) 子程序，您可以使用软件对该器件进行配置。下表列出并说明了每个函数的接口。以下各节将更详细地介绍每个函数。

默认情况下，PSoC Creator 将实例名称 “LCD_SegStat_1” 分配给指定设计中组件的第一个实例。您可以将该实例重新命名为符合标识符语法规则的唯一一个任意值。实例名称会成为每个全局函数名称、变量和常量符号的前缀。出于可读性考虑，下表中使用的实例名称为 “LCD_SegStat”。

函数	说明
LCD_SegStatic_Start()	启动LCD组件和DMA通道。初始化帧缓冲器。如果之前预定义了帧缓冲器RAM，并不会清除它。
LCD_SegStatic_Stop()	禁用LCD组件和关联的中断和DMA通道。不会清除帧缓冲器。
LCD_SegStatic_EnableInt()	使能LCD中断。如果调用了LCD_SegStat_Start()，则无需此函数。
LCD_SegStatic_DisableInt()	禁用LCD中断。如果调用了LCD_SegStat_Stop()，则无需此函数。
LCD_SegStatic_ClearDisplay()	清除帧缓冲器的显示RAM的数据。
LCD_SegStatic_WritePixel()	基于PixelState设置或清除一个像素。通过一个封装好的数据包寻址像素。
LCD_SegStatic_ReadPixel()	读取帧缓冲器中像素的状态。通过一个封装好的数据包寻址像素。
LCD_SegStatic_WriteInvertState()	根据输入参数反转显示屏。
LCD_SegStatic_ReadInvertState()	返回显示反向状态的当前值：正常或反向。
LCD_SegStatic_Sleep()	停止LCD，然后保存用户配置。
LCD_SegStatic_Wakeup()	恢复并使能用户配置。
LCD_SegStatic_Init()	配置各个帧中断，并初始化帧缓冲器。
LCD_SegStatic_Enable()	启用组件的时钟生成。
LCD_SegStatic_SaveConfig()	保存LCD配置。
LCD_SegStatic_RestoreConfig()	恢复LCD配置。

注意：包含后缀 “n” 的函数名称表示在组件自定义程序中创建了多个相同符号类型的显示助手。特定显示助手元素由 API 函数控制，函数名称各自都带有 “n” 索引值。

全局变量

变量	说明
LCD_SegStatic_initVar	指示是否已对LCD_SegStat进行了初始化。变量将初始化为0，并在第一次调用LCD_SegStat_Start()时设置为1。这样，第一次调用LCD_SegStat_Start()子程序后，组件不用重新初始化即可重启。 如果需要重新对组件进行初始化，可在调用 LCD_SegStatic_Start() 或 LCD_SegStatic_Enable()函数之前先调用LCD_SegStatic_Init()函数。

uint8 LCD_SegStatic_Start(void)

说明： 启动LCD组件、DMA通道、帧缓冲器和硬件。不清除帧缓冲器RAM。

参数： 无

返回值： uint8 cystatus: 标准API返回值。

返回值	说明
CYRET_LOCKED	某些TD或通道已经被占用。
CYRET_SUCCESS	函数已成功完成

其他影响： 无

void LCD_SegStatic_Stop(void)

说明： 禁用LCD组件和关联的中断和DMA通道。自动清空显示屏，以避免因直流偏移而产生的损害。不清除帧缓冲器。

参数： 无

返回值： 无

其他影响： 无

void LCD_SegStatic_EnableInt(void)

说明： 启用LCD中断。如果调用了LCD_SegStat_Start()，则无需此函数。每次更新LCD后（TD完成）都会发生中断。

参数： 无

返回值： 无

其他影响： 无

void LCD_SegStatic_DisableInt(void)

说明: 禁用LCD中断。如果调用了LCD_SegStat_Stop(), 则无需此函数。

参数: 无

返回值: 无

其他影响: 无

void LCD_SegStatic_ClearDisplay(void)

说明: 此函数会清除页面缓冲器的显示屏RAM。

参数: 无

返回值: 无

其他影响: 无

uint8 LCD_SegStatic_WritePixel(uint16 pixelNumber, uint8 pixelState)

说明: 此函数基于“PixelState”参数设置或清除帧缓冲器中的像素。通过一个封装好的数组对此像素进行寻址。

参数: **uint16 pixelNumber:** 指的是帧缓冲器中指向像素位置的封装数组。LSB低效半字节中3个最低有效位是字节中的数位位置, LSB高效半字节(4位)是复用行中的字节地址, 而MSB低效半字节(4位)则是复用行编号。

uint8 pixelState: 指定的pixelNumber设为此像素状态。提供了像素状态的符号名称, 其相关值如下所示:

值	说明
LCD_SegStatic_PIXEL_STATE_OFF	将像素设置为关闭。
LCD_SegStatic_PIXEL_STATE_ON	将像素设置为打开。
LCD_SegStatic_PIXEL_STATE_INVERT	反转像素的当前值。

返回值: **uint8 status:** 根据字节地址和复用行编号范围检查通过或失败。未对数位位置执行检查。

返回值	说明
CYRET_BAD_PARAM	封装字节地址或行值无效
CYRET_SUCCESS	函数已成功完成

其他影响: 无



uint8 LCD_SegStatic_ReadPixel(uint16 pixelNumber)

说明: 此函数在帧缓冲器中读取像素的状态。通过一个封装好的数据包寻址像素。

参数: uint16: pixelNumber: 帧缓冲器中指向像素位置的封装数组。LSB低效半字节中3个最低有效位是字节中的数位位置，LSB高效半字节（4位）是复用行中的字节地址，而MSB低效半字节（4位）则是复用行编号。

返回值: uint8 status: 返回指定PixelNumber的当前状态。

值	说明
0	像素关闭。
1	像素打开。

其他影响: 无

uint8 LCD_Seg_WriteInvertState(uint8 invertState)

说明: 此函数根据输入参数进行反转显示屏。

参数: uint8 invertState: 设置显示的反转状态。

值	说明
LCD_SegStatic_NORMAL_STATE	正常非反相显示。
LCD_SegStatic_INVERTED_STATE	反向显示。

返回值: uint8 cstatus: 标准API返回值。

值	说明
CYRET_SUCCESS	函数已成功完成。
CYRET_BAD_PARAM	InvertState参数赋值失败

其他影响: 无

uint8 LCD_Seg_ReadInvertState(void)

说明: 此函数用于返回显示器反转状态的当前值：正常或反转。

参数: 无

返回值: (uint8) invertState: 设置显示的反转状态。

值	说明
LCD_SegStatic_NORMAL_STATE	正常非反相显示。
LCD_SegStatic_INVERTED_STATE	反向显示。

其他影响: 无

void LCD_SegStatic_Init(void)

说明: 根据自定义程序“Configure”对话框设置，初始化或恢复组件。无需调用LCD_SegStatic_Init()，因为LCD_SegStatic_Start()子程序会调用该函数，并且它是开始组件操作的首选方法。配置各个帧中断，并初始化帧缓冲器。

参数: 无

返回值: 无

其他影响: 无

void LCD_SegStatic_Enable(void)

说明: 启用组件的时钟生成。

参数: 无

返回值: 无

其他影响: 无

void LCD_SegStatic_Sleep(void)

说明: 这是准备组件睡眠的首选子程序。LCD_SegStatic_Sleep()子程序保存当前组件的状态。然后，它会调用LCD_SegStatic_Stop()函数，并调用LCD_SegStatic_SaveConfig()以保存硬件配置。

在调用CyPmSleep()或CyPmHibernate()函数之前调用LCD_SegStatic_Sleep()函数。有关功耗管理函数的详细信息，请参考PSoC Creator《系统参考指南》。

参数: 无

返回值: 无

其他影响: 不更改组件引脚的驱动模式。



void LCD_SegStatic_Wakeup(void)

说明: 该函数是将组件恢复到调用 LCD_SegStatic_Sleep() 状态的首选子程序。LCD_SegStatic_Wakeup()函数调用LCD_SegStatic_RestoreConfig()函数，以恢复配置。如果组件在调用LCD_SegStatic_Sleep()函数前已启用，则 LCD_SegStatic_Wakeup()函数也将重新启用组件。

参数: 无

返回值: 无

其他影响: 如果调用 LCD_SegStatic_Wakeup() 函数前未调用 LCD_SegStatic_Sleep() 或 LCD_SegStatic_SaveConfig()函数，则可能产生意外行为。

void LCD_SegStatic_SaveConfig(void)

说明: 此函数会保存组件配置以及非保留寄存器。它还会保存“Configure”对话框中定义的或通过相应API修改的当前组件参数值。该函数由LCD_SegStatic_Sleep()函数调用。

参数: 无

返回值: 无

其他影响: 无

void LCD_SegStatic_RestoreConfig(void)

说明: 此函数会恢复组件配置以及非保留寄存器。它还将组件参数值恢复为在调用 LCD_SegStatic_Sleep()函数之前的值。

参数: 无

返回值: 无

其他影响: 调用 LCD_SegStatic_RestoreConfig() 函数前未调用 LCD_SegStatic_Sleep() 或 LCD_SegStatic_SaveConfig()函数，则可能会产生意外行为。

可选助手 APIs

只有在 Configure 对话框中选中个别助手时，才会提供下列 API。

功能	说明
LCD_SegStatic_Write7SegDigit_n	显示7段显示元素阵列中的十六进制数字。
LCD_SegStatic_Write7SegNumber_n	显示7段显示元素1-5位数字的整型值。
LCD_SegStatic_WriteBargraph_n	显示线性或圆形条形图上的整数位置
LCD_SegStatic_PutChar14Seg_n	显示14段字母数字字符显示元素阵列中的字符。
LCD_SegStatic_WriteString14Seg_n	显示14段字母数字字符显示元素阵列中的空字符结束的字符串。
LCD_SegStatic_PutChar16Seg_n	显示16段字母数字字符显示元素阵列中的字符。
LCD_SegStatic_WriteString16Seg_n	显示16段字母数字字符显示元素阵列中的空字符结束的字符串。

void LCD_SegStatic_Write7SegDigit_n(uint8 digit, uint8 position)

说明： 此函数用来显示7段显示元素阵列中的十六进制数字。数字可以为0到9以及A到F范围内的十六进制值。必须使用定制器显示助手工具定义与7段显示元素相关联的像素集。可在帧缓冲器中定义多个7段显示元素，并可通过函数名称中的后缀（n）对这些元素进行寻址。只有在组件自定义程序中定义了7段显示元素，才包含该函数。

参数： **uint8 digit:** 要显示为十六进制数字的无符号整数值（其范围为0到15）。
uint8 position: 从右侧的0开始自右向左计算的数字位置。如果此位置不在定义的显示区域内，将不显示此字符。

返回值： 无

其他影响： 无

void LCD_SegStatic Write7SegNumber_n(uint16 value, uint8 position, uint8 mode)

说明: 此函数显示7段显示元素中1位到5位阵列上的16位整数。自定义程序显示助手工具必须用于定义与7段显示元素相关联的像素集。在帧缓冲区中，可以定义多个7段显示元素组，并通过函数名称中的后缀（n）寻址它们。符号转换、符号显示、小数点和其他自定义特性必须由应用特定的用户代码来进行处理。只有在器件自定义程序中定义了7段显示元素，才包含此函数。

参数: uint16 value: 显示无符号的整数值。

uint8位置: 从右侧的0开始自右向左计算的最低有效位位置。如果定义的显示区域包含的位数少于值需要的位数，则不会显示最高有效位或多个位

uint8模式: 设置显示模式。可能为0或1。

值	说明
LCD_SegStatic_MODE_0	不显示前导零。
LCD_SegStatic_MODE_1	显示前导零

返回值: 无

其他影响: 无

void LCD_SegStatic_WriteBargraph_n(uint16 location, int8 Mode)

- 说明：

此函数显示1到255段条形图（自左向右编号）上的8位整数位置。条形图可以是任意用户定义的大小（1到255段）。此外，条形图还可以创建为圆形，用来显示旋转位置。必须使用定制器显示助手工具定义与条形图显示元素相关联的像素集。可在帧缓冲区中定义多个条形图显示，并可通过函数名称中的后缀（n）对这些显示进行寻址。只有在组件自定义程序中定义了条形图显示元素，才可以包含该函数
- 参数：

uint16位置：要显示的无符号整数位置。有效值为从0到条形图的段数。0值关闭所有条形图元素。大于条形图段数的值将会导致所有元素为打开状态。

int8模式：设置条形图显示模式。

值	说明
0	打开指定位置段
1	打开位置段及其左侧的所有段
-1	位置段和右侧所有段均为打开状态。
2-10	显示位置段和右侧2-10段。此模式可用于创建较宽的指示符。

- 返回值：

无
- 其他影响：

无

void LCD_SegStatic_PutChar14Seg_n(uint8 character, uint8 position)

- 说明：

此函数用来显示14段字母数字字符显示元素阵列中的一个8位字符。自定义程序显示助手工具必须用于定义与14段显示元素相关联的像素集。在帧缓冲器中，可以定义多个14段字母数字显示元素组，并通过函数名称中的后缀（n）来寻址。只有在组件自定义程序中定义了14段显示元素，才包含该函数。
- 参数：

uint8字符：要显示字符的ASCII值（ASCII 值为0-127的可打印字符）

uint8位置：从左侧的0开始自左向右计算的字符位置。如果此位置不在定义的显示区域内，将不显示此字符。
- 返回值：

无
- 其他影响：

无



void LCD_SegStatic_WriteString14Seg_n(uint8 const character[], uint8 position)

说明: 此函数用来显示14段字母数字字符显示元素阵列中的空字符结束的字符串。必须使用定制器显示助手工具定义与14段字母数字显示元素相关联的像素集。可在帧缓冲器中定义多个14段字母数字显示元素组，并可通过函数名称中的后缀（n）对这些元素组进行寻址。只有在组件自定义程序中定义了14段显示元素，才包含此函数

参数: uint8 const character[]: 指针指向空字符结束的字符串。

uint8位置: 从左侧的0开始自左向右计算的第一个字符的位置。如果字符串的长度超出已定义显示区域的大小，将不显示额外的字符。

返回值: 无

其他影响: 在数据输出之前不清除显示。未受影响的所有位置将仍处于其先前的像素状态。

void LCD_SegStatic_PutChar16Seg_n(uint8 character, uint8 position)

说明: 此函数用来显示16段字母数字字符显示元素阵列中的一个8位字符。自定义显示助手工具必须用于定义与16段显示元素相关联的像素集。在帧缓冲器中，可以定义多个16段字母数字显示元素组，并通过函数名称中的后缀（n）来寻址。只有在组件自定义程序中定义16段显示元素，才包含该函数。

参数: uint8字符: 要显示的字符的ASCII 值（值为0到255的可打印 ASCII和表扩展字符）

uint8位置: 从左侧的0开始自左向右计算的字符位置。如果此位置不在定义的显示区域内，将不显示此字符。

返回值: 无

其他影响: 无

(void) LCD_SegStatic_WriteString16Seg_n(uint8 const character[], uint8 position)

- 说明：

此函数显示16段字母数字字符显示元素阵列上以空字符作为结尾的各字符串。自定义显示助手工具必须用于定义与16段显示元素相关联的像素集。在帧缓冲器中，可以定义多个16段字母数字显示元素组，并通过函数名称中的后缀（n）来寻址。只有在组件自定义程序中定义了16段显示元素时，此函数才可用。
- 参数：

uint8 const character[]： 指针指向空字符结束的字符串。

uint8位置： 从左侧的 0 开始自左向右计算的第一个字符的位置。如果字符串的长度超出已定义显示区域的大小，将不显示额外的字符。
- 返回值：

无
- 其他影响：

在数据输出之前不清除显示。未受影响的所有位置将仍处于其先前的像素状态。

引脚 API

这些 API 函数用于更改静态段 LCD 组件所用引脚的驱动模式。这些 API 大多用于将静态段 LCD 组件引脚置于 HI-Z 模式，以便在器件处于低功耗模式时将漏电流降到最低。

函数	说明
LCD_SegStatic_ComPort_SetDriveMode	针对静态段LCD组件的通用线路使用的引脚设置驱动模式。
LCD_SegStatic_SegPort_SetDriveMode	针对静态段LCD组件的段线路使用的所有引脚设置驱动模式。

void LCD_SegStatic_ComPort_SetDriveMode(uint8 mode)

- 说明：

针对静态段LCD组件的通用线路使用的引脚设置驱动模式。
- 参数：

uint8 mode: 所需的驱动模式有关驱动模式的信息，请参考引脚组件数据手册。
- 返回值：

无
- 其他影响：

无



LCD_SegStatic_SegPort_SetDriveMode(uint8 mode)

说明:	针对静态段LCD组件的段线路使用的所有引脚设置驱动模式。
参数:	uint8 mode: 所需的驱动模式有关驱动模式的信息，请参考引脚组件数据手册。
返回值:	无
其他影响:	无

定义**LCD_SegStatic_SEG_NUM**

此宏针对组件当前的用户定义显示屏配置定义段线路数。

LCD_SegStatic_FRAME_RATE

此宏针对组件当前的用户定义显示屏配置定义刷新率。

LCD_SegStatic_WRITE_PIXEL

这是返回 void 的 LCD_SegStatic_WritePixel()函数的宏定义。

LCD_SegStatic_READ_PIXEL

这是 LCD_SegStatic_ReadPixel()函数的宏定义。

LCD_SegStatic_FIND_PIXEL

此宏用来计算帧缓冲器中的像素位置。它使用定制器像素表中的信息和专用于 LCD 的物理引脚的相关信息。此宏是像素映射机制的基础。像素表中的每个像素名称都是用帧缓冲器中计算出的像素位置定义的。API 使用像素名称访问各自的像素。

MISRA 合规性

本节介绍了 MISRA-C:2004 合规性和本器件的偏差情况。定义了下面两种类型的偏差：

- 项目偏差 — 适用于所有 PSoC Creator 组件的偏差
- 特定偏差 — 仅适用于该组件的偏差

本节提供了有关组件特定偏差的信息。在 *系统参考指南* 的“MISRA 合规性”章节中介绍了项目偏差以及有关 MISRA 合规性验证环境的信息。

尚未根据 MISRA-C: 2004 编码准则合规性，验证静态段 LCD 组件源代码。

固件源代码示例

在“Find Example Project”对话框中，PSoC Creator 提供了大量的示例项目，包括原理图和示例代码。要获取组件特定的示例，请打开器件目录中的对话框或原理图中的器件实例。要查看通用示例，请打开“Start Page”或“File”菜单中的对话框。根据要求，可以通过使用对话框中的 **Filter Options** 选项来限定可选的项目列表。

更多有关信息，请参考《PSoC Creator 帮助》部分中主题为“查找示例项目”的内容。

功能说明

静态段 LCD 组件提供强大而灵活的机制，以驱动不同类型的 LCD 显示屏。使用配置对话框，可访问可用于定制组件的参数。可使用一组标准的 API 子程序控制显示屏和特定像素。其他显示屏 API 是基于定义的显示助手的类型和数量而生成的。

默认配置

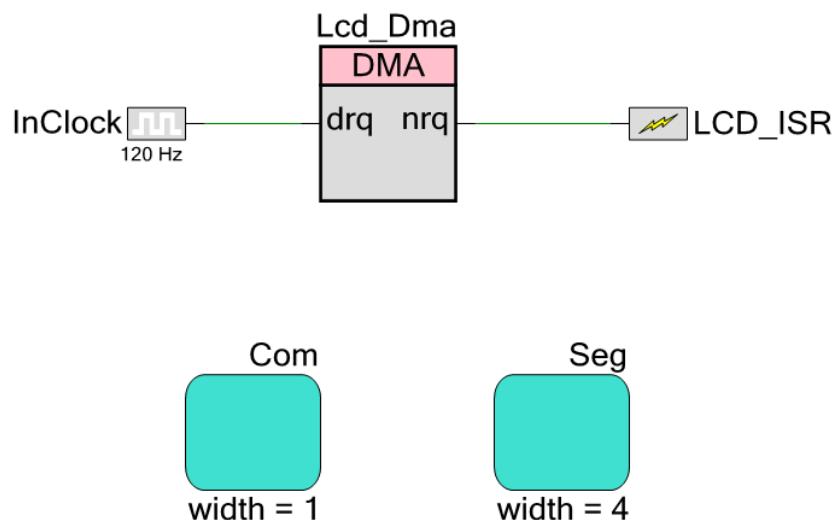
LCD_SegStatic 组件的默认配置提供普通的 LCD 直接段驱动控制器。默认 LCD_SegStatic 配置为：

- 四条段线路
- 30 Hz 刷新率
- 未定义显示助手。默认 API 生成不包括任何支持的显示元素的函数。



框图和配置

下图显示的是静态段 LCD 组件的原理图。组件不需要专用的 LCD 驱动硬件。组件利用 DMA 和标准数字端口 I/O。



此图显示的是静态段 LCD 组件的内部原理图。它包含一个 DMA 组件、ISR 组件、时钟组件和两个 LCD 端口。

- DMA 组件用于将数据通过伪信号存储器区域从帧缓冲器传输到 LCD 数据寄存器。
- LCD 端口（Com 和 Seg）用于将逻辑信号映射到物理引脚上。有两个 LCD 端口实例：一个是通用线路实例，一个是段线路实例。
- ISR 组件可用于用户中断。

顶层架构

组件的架构非常简单。它仅使用几个模块。组件以包含 LCD 数据的帧缓冲器为基础。可使用 `LCD_SegStatic_WritePixel()` 函数或引用此参数的高级函数修改帧缓冲器。然后，DMA 使用伪存储器将 LCD 数据传输到端口寄存器。DMA 数据操作由内部时钟触发，此时钟设置预计算的值，用以为 LCD 提供正确的刷新率。

资源

静态段 LCD 组件使用一个 DMA 组件和一个 ISR。另外，该组件还为共模占用一个 I/O 引脚，并且为每个段线占用一个引脚。

API 存储器大小

根据编译器、器件、所使用的API数量以及组件的配置不同，组件对存储资源的占用也不一样。下表提供了在某种器件配置中所有API占用存储器的大小。

数据是在将编译器设置为Release模式并将优化等级设置为Size的情况下测得的。对于特定设计，可以分析编译器生成的映射文件，从而确定存储器的使用大小。

基本：低级 API 函数集，无任何高级助手 API

基本，7 段助手：低级 API 函数集 + 7 段助手 API

基本，14 段助手：低级 API 函数集 + 14 段助手 API

基本，16 段助手：低级 API 函数集 + 16 段助手 API

基本，条形图助手：低级 API 函数集 + 条形图助手高级 API

配置	PSoC 3 (Keil_PK51)		PSoC 5LP (GCC)	
	闪存 字节	SRAM 字节	闪存 字节	SRAM 字节
基本	1652	40	750	45
基本，7段助手	1963	40	884	45
基本，14段助手	2131	40	1118	45
基本，16段助手	2135	40	1122	45
基本的条形图助手	2195	40	986	45

直流和交流电的电气特性

N/A

组件勘误表

本节列出了组件的已知问题。

赛普拉斯ID	组件版本	问题	解决方案
191257	v2.30	在没有修正PSoC Creator 3.0 SP1中的版本编号时进行更改这组件。更多有关信息，请参见基础知识文章 KBA94159（网页地址： www.cypress.com/go/kba94159 ）。	解决方案是不必要的。不会对设计产生影响。



组件更改

本节列出了该组件各版本中的主要更改内容。

版本	更新内容	更改原因/影响
2.30.a	编辑数据手册并将其添加到组件勘误章节。	文档的组件被更改，但设计不受任何影响。
2.30	此组件未进行MISRA合规性验证。	更新了“MISRA合规性”章节。
	LCD_SegStaticic_dispN[] 声明被移转到所使用它的函数内。	MISRA相关的更改。为组件中的每个助手声明一个 LCD_SegStatic_dispN[] (N=0,...,7) 实例，该实例仅用于一个特定于助手的函数。
	常量 LCD_SegStatic digitNum_n (n=0,...,7) 由 #define LCD_SegStatic_DIGIT_NUM_n代替。	MISRA相关更改。因为LCD_SegStatic_digitNum_n值在代码内不变，所以最好将它声明为#define。
	下面的变量为“静态”： • LCD_SegStatic_7SegDigits[]; • LCD_SegStatic_14SegChars[]; • LCD_SegStatic_16SegChars[].	MISRA合规性的相关更改。这些变量特意用于助手API函数，所以应声明为“静态”。
	LCD_Seg_WriteString14Seg_n() 和 LCD_Seg_WriteString16Seg_n() API 中的“character”参数的声明是由“uint8 * character”更改为“uint8 const character[]”的。	MISRA相关更改。将变量声明为指针时，MISRA不再允许将变量作为阵列使用。
	修改了在 LCD_SegStatic_WritePixel() 和 LCD_SegStatic_ReadPixel() 中 pixelNumber 参数的参数类型，从uint8改为uint16。	这些函数的执行不匹配于数据手册。
	更正了LCD_SegStatic_WriteBargraph_n()的“Mode”参数的说明。	数据手册中指定“Mode”参数类型为“unit8”，而不是“int8”。
	第二页上的帧率默认值从30改为60 Hz，以匹配实现。	
	更正了LCD_SegStatic_Start()函数中返回值的说明。	该说明指定了 CYRET_BAD_PARAM 而不是 CYRET_LOCKED。
	删除PSoC 5支持。	
2.20	已添加了MISRA合规性章节。	此组件未进行MISRA合规性验证。
	删除了表示增加助手函数后不能修改共模线数的备注。	目前组件已允许在助手增加的情况下进行修改共模信号数。
	更新了API存储器使用表。	
	修改了 LCD_SegStatic_Write7SegNumber_n ()说	

版本	更新内容	更改原因/影响
	明中的问题。参数“Value”的类型是uint16，不过数据手册中表示为uint8。	
	修改了LCD_SegStatic_WriteBargraph_n()说明中的问题。参数“位置”是uint16的一种类型，不过数据手册中表示成uint8。	
	解决了LCD_SegStatic_WriteStringDotMatrix_n()和Seg_Display_WriteString16Seg_n()中的问题。 API。	之前的执行预期使用空指针作为字符串的结尾，而不是一个单一的零字节。这样会导致显示一个不正确的字符串。
	解决了LCD_SegStatic_WriteBargraph_n() API的相关问题。	此前，该API不清除其先前所显示的输出。这样可能导致LCD上显示意外结果。
	解决了LCD_SegStatic_Write7SegNumber_n() API的相关问题	此前，该API不会清除其先前所显示的输出。这样可能导致LCD上显示意外结果。
2.10	将默认实例名称由LCD_SegStat改为LCD_SegStatic。	
	向数据手册中添加了特性数据	
	增加了两个新的API： LCD_SegStatic_ReadInvertState(); LCD_SegStatic_WriteInvertState();	LCD_SegStatic_WriteInvertState()允许反转整个LCD显示屏。该功能可用于测试目的。 LCD_SegStatic_ReadInvertState()用于读取显示状态是正常还是反转状态。
	解决了LCD_SegStatic_Write7SegNumber() API中的问题。	此API有一个错误，即LCD上所显示的最后一位数字被0覆盖。仅在使用“前导零”输出模式时才会发生这种情况。
	向“.cyre”文件中包括的所有组件API中添加了CYREENTRANT关键词。	添加此功能，以便使客户能够指定各个生成函数可重入。
	向组件定制器内添加了“Enable ganging commons”（使能组合共用信号）复选框。	这是一个新的功能，有助于驱动大型的显示。
2.0	已从PSoC 5芯片中移除低功耗API支持。	将以下APIs添加到条件编译中： LCD_SegStat_SaveConfig(); LCD_SegStat_RestoreConfig(); LCD_SegStat_Sleep(); LCD_SegStat_Wakeup(). 如果工程在PSoC 5器件上运行，则该工程不再包括这些API。
	解决了LCD_SegStat_Stop() API问题。	LCD_SegStat_Stop() 函数未释放DMA配置。这将导致组件在数个LCD_SegStat_Start() -> LCD_SegStat_Stop()序

版本	更新内容	更改原因/影响
		列之后会停止运行。出现此情况是因为组件用完了在 LCD_SegStat_Start() 函数中所分配的DMA资源，且未在 LCD_SegStat_Stop() API 中释放这些资源。在 LCD_SegStat_Stop() API中实施和使用正确的DMA释放程序。
	已将WRITE_PIXEL()宏定义从 <pre>#define LCD_SegStat_WRITE_PIXEL(pixelNumber, pixelState) LCD_SegStat_WritePixel(pixelNumber, pixelState)</pre> 改为 <pre>#define LCD_SegStat_WRITE_PIXEL(pixelNumber, pixelState) (void) LCD_SegStat_WritePixel(pixelNumber, pixelState)</pre>	无需在组件实现过程中分析LCD_SegStat_WritePixel() API的返回值，因此更新了宏以处理这种情况。
	已从头文件中删除了以下废弃的名称： LCD_SegStat_Buffer LCD_SegStat_Channel LCD_SegStat_TermOut LCD_SegStat_TD LCD_SegStat_GCommons LCD_SegStat_Commons	已在之前的组件版本中将名称标记为废弃。
	已优化函数 LCD_SegStat_WriteBargraph()。	代码重构为此组件的API节省了一些闪存空间。
	优化了函数 LCD_SegStat_WriteString14Seg() 和 LCD_SegStat_WriteString16Seg()。	已将阵列索引更改为指针索引，从而使组件节省了RAM空间，并从这些函数中获得更多的执行时间。
	解决了整个源代码上一行中的多个变量声明。	这种情况违反了内部代码标准规范。
	优化了 LCD_SegStat_Write7SegNumber() API实现。	LCD_SegStat_Write7SegNumber() API中有一些在此API的较长执行时间内出现的无效代码。
	解决了LCD_SegStat_Wakeup() API。	LCD_SegStat_Wakeup() API函数即使正确执行，也始终会返回 CYRET_LOCKED。已实施正确返回状态处理的程序。
1.50.a	已在数据表中将手动调用解决方案添加到 LCD_SegStat_Stop() 说明内	丢失LCD_SegStat_Stop()子程序中对 LCD_SegStat_DmaDispose()的调用。
	已将引脚APIs部分添加到数据表中。	

版本	更新内容	更改原因/影响
	数据规格书的微小编辑和更新。	
1.50	已添加睡眠API和唤醒API。	用于支持低功耗模式。
	已添加存储组件状态的备份结构。	在进入睡眠模式之前，备份结构存储组件状态（已启用/已禁用）。当调用了LCD_SegStat_Wakeup() API以将组件从睡眠状态中唤醒时，此结构使组件返回进入睡眠模式之前的状态。
	已在“Helpers”（助手）选项卡中添加了工具提示。	执行此操作的原因是“Configure”（配置）对话框的默认大小不足以显示像素映射表。
	已为设置为启用（之前为禁用）的“段线路数”添加数字上下控制和编辑字段，即使用户添加了助手函数。	之前，如要在添加助手之后更改段线路数，您必须删除助手并丢失所有配置信息。将一个助手添加到段LCD组件之后，可返回并更改段线路数。如果段数增加了，将会清除所有信息。如果段数减少了，您将看到一则警告，提醒您可能丢失配置信息。 如果要移除的段线路路上有已分配的像素，将取消这些像素的分配。
	已将错误提供商添加到选定的像素名称文本框内。如果在文本框中输入错误的值，将显示一个错误图标，同时您无法离开此文本框，直至输入正确的值为止。错误图标提供带问题说明的工具提示。	错误提示器可更好的处理错误的值。使用错误提示器，而非显示信息框。
	以下全局变量用camel小写名称进行了调整： #define ` \$LCD_SegStat`_Buffer #define ` \$LCD_SegStat`_Channel #define ` \$LCD_SegStat`_TermOut #define ` \$LCD_SegStat`_TD #define ` \$\$LCD_SegStat`_GCommons #define ` \$\$LCD_SegStat`_Commons	编码标准要求变量符合camel式小写的命名风格。这些变量被废弃，并将在之后移除。
1.30.b	向组件中添加了信息，以说明它与芯片修订版的兼容性。	如果组件在不兼容的芯片上使用，该工具将报告错误/警告。如果发生此情况，请更新到支持您的目标组件的修订版。
	进行了v1.30数据表日志累积更改	
1.30.a	已将本地参数移动到形式参数列表。	为了解决PSoC Creator v1.0 Beta 4.1和更早版本中存在的缺陷，更新了组件，以使它可以继续在该工具的较新版本中使用。此组件使用了本地参数（这些参数不向用户公开），以在用户输入时执行后台计算。这些参数已更改为可见的形式参数，但是不可编辑。组件没有功能上的更改，但是现在可在自定义程序对话框的“expression view”（表达式视图）中看到受影响的参数。

版本	更新内容	更改原因/影响
1.30	已更新生成的源代码。	已将StaticSegLCD_INT.c文件添加到组件的库中。 已更改组件的源代码，以针对内部中断配置使用CyLib.h函数。 已添加DMA补丁，以在32位地址空间中正常工作。 已将一个等式添加到组件源文件中，以从DMA中选择正确的终止信号。

赛普拉斯半导体公司，2013-2016 年。本文件是赛普拉斯半导体公司及其子公司，包括 Spansion LLC（“赛普拉斯”）的财产。本文件，包括其包含或引用的任何软件或固件（“软件”），根据全球范围内的知识产权法律以及美国与其他国家签署条约由赛普拉斯所有。除非在本款中另有明确规定，赛普拉斯保留在该等法律和条约下的所有权利，且未就其专利、版权、商标或其他知识产权授予任何许可。如果软件并不附随有一份许可协议且贵方未以其他方式与赛普拉斯签署关于使用软件的书面协议，赛普拉斯特此授予贵方属人性质的、非独家且不可转让的如下许可（无再许可）（1）在赛普拉斯特软件著作权项下的下列许可权（一）对以源代码形式提供的软件，仅出于在赛普拉斯硬件产品上使用之目的且仅在贵方集团内部修改和复制软件，和（二）仅限于在有关赛普拉斯硬件产品上使用之目的将软件以二进制代码形式的向外部最终用户提供（无论直接提供或通过经销商和分销商间接提供），和（2）在被软件（由赛普拉斯公司提供，且未经修改）侵犯的赛普拉斯专利的权利主张项下，仅出于在赛普拉斯硬件产品上使用之目的制造、使用、提供和进口软件的许可。禁止对软件的任何其他使用、复制、修改、翻译或汇编。

在适用法律允许的限度内，赛普拉斯未对本文件或任何软件作出任何明示或暗示的担保，包括但不限于关于适销性和特定用途的默示保证。赛普拉斯保留更改本文件的权利，届时将不另行通知。在适用法律允许的限度内，赛普拉斯不对因应用或使用本文件所述任何产品或电路引起的任何后果负责。本文件，包括任何样本设计信息或程序代码信息，仅为供参考之目的提供。文件使用人应负责正确设计、计划和测试信息应用和由此生产的任何产品的功能和安全性。赛普拉斯产品不应被设计为、设定为或授权用作武器操作、武器系统、核设施、生命支持设备或系统、其他医疗设备或系统（包括急救设备和手术植入物）、污染控制或有害物质管理系统中的关键部件，或产品植入之设备或系统故障可能导致人身伤害、死亡或财产损失其他用途（“非预期用途”）。关键部件指，若该部件发生故障，经合理预期会导致设备或系统故障或会影响设备或系统安全性和有效性的部件。针对由赛普拉斯产品非预期用途产生或相关的任何主张、费用、损失和其他责任，赛普拉斯不承担全部或部分责任且贵方不应追究赛普拉斯之责任。贵方应赔偿赛普拉斯因赛普拉斯产品任何非预期用途产生或相关的所有索赔、费用、损失和其他责任，包括因人身伤害或死亡引起的主张，并使之免受损失。

赛普拉斯、赛普拉斯徽标、Spansion、Spansion 徽标，及上述项目的组合，WICED，及 PSoC、CapSense、EZ-USB、F-RAM 和 Traveo 应视为赛普拉斯在美国和其他国家的商标或注册商标。请访问 cypress.com 获取赛普拉斯商标的完整列表。其他名称和品牌可能由其各自所有者主张为该方财产。