Application Note

采用 MSPM0 MCU 的 USB 设计

摘要

本指南向希望使用 MSPM0 微控制器 (MCU) 设计基于 USB 的设备的用户提供概要性介绍。本文档重点介绍 TI MSPM0 提供的硬件和软件资源，说明 MSPM0 器件的内置 USB 功能，并概述可用于简化开发的工具。

本指南涵盖：

- USB 系统概览
- MSPM0 USB 硬件模块和可用的 MCU 系列
- USB 硬件设计建议
- MSPM0 USB 软件开发人员套件和支持系统概述
- 基于 MSPM0 MCU 的 USB 设备折硬件参考设计

English Document: SDAA280

1 USB 让复杂的系统看起来更简单

1.1 为什么 USB 如此成功？

USB 是当今使用最广泛的行业标准之一。简单易用和流畅的操作体验使其成为许多用户必不可少的工具。还有其他几项优势，包括：

- 经济实用：低成本使得 USB 对用户很有吸引力，促使其广泛被采用
- 广泛的兼容性：USB 的普及使其成为许多产品的标准功能。
- 扩展用例：除了主要功能外，USB 还支持创新用例，例如：
  - 电力输送：为小型装置和设备充电。
  - 外设连接：实现各种外设 (如键盘、鼠标等) 的连接。
  - 数据存储：提供从闪存驱动程序到外部硬盘驱动器的各种存储选项。
- 可靠性：在 USB 设备之间建立无错误、无故障的稳定连接。

尽管对最终用户来说非常简单，但 USB 内部设计可能很复杂。这种复杂性源于需要提供快速可靠的数据总线，该总线可自动执行常见行为并承受热插拔，那需要多层协议。

1.2 为什么 USB 看起来很简单？

USB 向用户隐藏了复杂性，但开发者往往能看到其底层的运作机制。与 UART、SPI 或 I2C 等其他协议相比，USB 需要更多的数据传输处理能力。USB 发送数据比写入寄存器更费事。

片上 USB 模块有助于部分地降低其复杂性，但这些模块的复杂性无法消除。USB 堆栈的很大一部分仍然需要使用软件来管理复杂的 USB 通信。精心设计的软件可以使应用程序开发人员免受其中许多复杂问题的影响，但它们仍然面临着如下关键挑战：

- 处理设备连接和断开事件
- 即使在总线繁忙或不可靠之类的严苛条件下，也能保持稳定的通信和可靠的数据传输，以防止数据丢失。
- 制定策略来管理和支持多个主机操作系统。

这些注意事项对于开发人员来说至关重要，有助于验证基于 USB 的应用是否可靠、高效且对用户友好。

2 MSPM0 USB 器件

MSPM0 微控制器系列包含集成式 USB 模块。该模块与最新的 MSPM0-SDK 兼容，后者包含称为 Tiny USB 的开发包。

尽管通过位拆裂实现 USB 通信是可行的，但这种方法不是全速运行的首选，并且可能会消耗处理器的大部分容量。大多数 USB 应用都依赖于片上 USB 模块，该模块提供了一种更高效、更可靠的方式来管理 USB 通信。

2.1 MSPM0 器件的文档说明体系

对于任何给定的 MSPM0 器件系列成员，MSPM0 器件文档分属三个位置：

- 技术参考手册：包含 MSPM0 MCU 系列的所有架构信息。所有集成 USB 功能的器件都可以在 *MSPM0 G 系列 80MHz 微控制器技术参考手册*中找到。
- 数据表：该数据表包含特定于该器件系列成员的所有参数和详细信息，有助于详细了解其特性和功能。
- SDK 用户指南：SDK 用户指南概述了不同的驱动程序和示例程序，可将这些程序用作为新工程创建软件的起点。

结合这些文档，可以全面地了解 MSPM0 器件系列。

2.2 MSPM0 USB 模块

MSPM0 USB 模块具备以下特性：

- 主机和设备模式下可实现 USB 2.0 全速 (12Mbps) 运行，主机模式下可实现低速 (1.5Mbps) 运行
- 符合 USB-IF 认证标准
- 四种传输类型：控制、中断、批量和等时
- 十六个端点
  - 一个专用的控制输入端点和一个专用的控制输出端点
  - 七个可配置输入端点和七个可配置输出端点
- 2kB 专用端点存储器
- 用于支持 DMA 的专用硬件触发器
- 通过引导加载程序 (BSL) 支持基于 USB 的器件固件更新 (DFU)
- USBFS 模块包含集成式全速 PHY
- 支持挂起与恢复信令功能

3 MSPM0 USB 硬件设计

3.1 方框图

在图 3-1 展示了 USB 结构图。

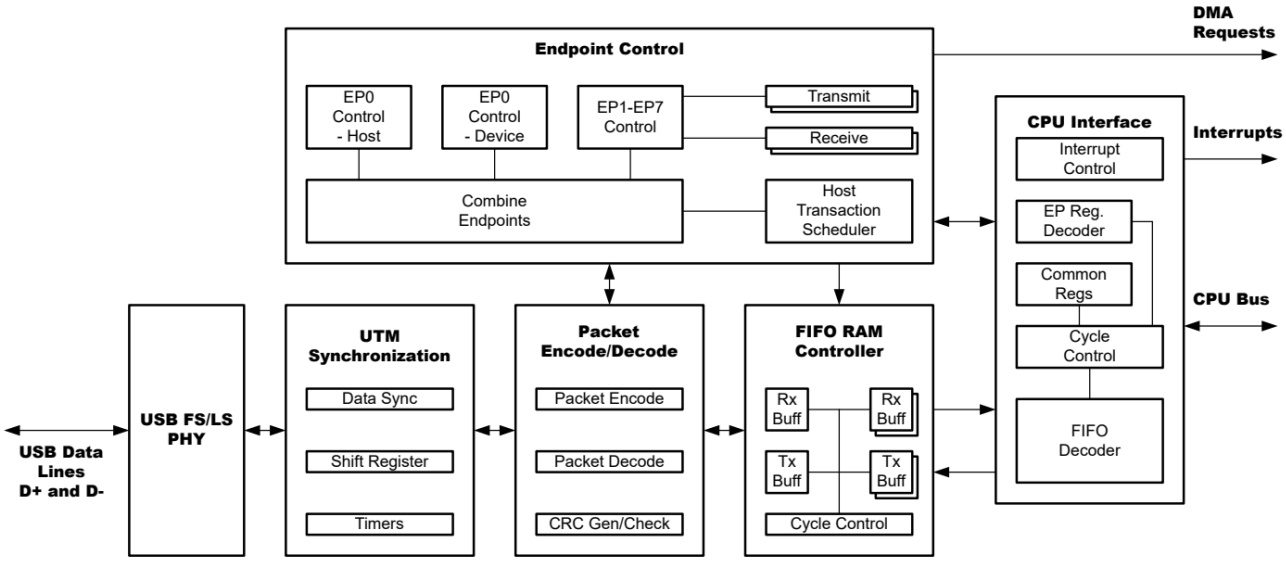

图 3-1. MSPM0 中 USB 外设的方框图

3.2 USB 运行模式

USB IP 在静态配置中支持器件模式和主机模式。

对于 USB-C 型连接器，需要通过 PCB 系统 CC1/CC2 线路上的上拉/下拉电阻来指示器件和主机模式，这会限制模式动态配置的 USB IP。因此，USB IP 不支持在器件和主机模式之间进行 On-The-Go (OTG) 切换。

3.2.1 USB 设备模式：总线供电

在总线供电的 USB 应用中，设备通过 USB 连接器从 USB 主机获得电源。此供电方法限于仅在连接时运行的 USB 设备应用，因为需要 USB 主机为总线供电。总线供电的 USB 应用的典型示例是 USB 鼠标或键盘。这些设备仅在设备连接到计算机且设备从 USB 主机获得电源时才工作。

下图展示了总线供电应用的系统组件。由于 VBUS 上的电压为 5V，因此该系统需要一个外部 3.3V LDO 来为额外的板载元件供电。在此模式下，无需 USB 连接检测电路，因为 SoC 仅在通电后运行。

3.2.2 USB 设备模式：自供电

另一个用于开启 USB 设备的选项是*自供电*。这意味着除了 USB 电源外，设备上还有一些外部电源，在必要时为芯片和连接供电。

音频或存储设备可作为此类模式的典型示例，即使在断开连接的情况仍然需要供电。与总线供电模式相比，这些设计需要额外的电源管理，根据电源树的复杂性使用某种形式的主电源稳压器和时序控制。

通常，自供电设备也可以被视为*混合型*，这意味着它们要么在连接时使用总线电源，要么在连接时为电池充电。因此，通常需要使用 VBUS 监控电路来检测连接并切换到总线提供的电源路径。这可以通过连接到 MSPM0G5187 上 ADC 引脚的电阻分压器来实现，后者可切换电源路径上的开关。

3.2.3 USB 主机模式电源注意事项

与 MSP430 不同，MSPM0Gx 能够在 USB 设计中作为主机运行。虽然大多数复杂性来自这些应用的软件设计，但在设计主机时会有额外的硬件要求。

电源是设计主机器件时的主要需求之一，因为主机*必须为连接的器件提供 5V VBUS* 电源，USB 2.0 全速模式需要高达 500mA 的电流。这通常意味着应将器件与专用电源（例如，单独的开关电源）配对。由于 USB 设备通常由总线供电，因此确保主机能够满足任何器件的电源要求对于保持一致运行至关重要。

必须考虑保护措施，因为无法保证所连接的器件设计正确，并可能会引入过流或过压情况。在电源线上集成保护电路和电流感应电阻器有助于确保器件不会暴露于此类过功率情况。

3.2.4 ESD 注意事项

对于数据线和 VBUS 的 ESD 保护，主机需要更高程度的稳健性，因为多个、未知且潜在可疑的器件可能会导致下游发生 ESD 事件。通常，建议使用更高的 ESD 额定值，接触放电约为 8kV，空气放电约为 15kV。

3.2.5 布局布线注意事项

与 USB 设备相似，匹配数据线上的走线长度以验证两条线路是否同步至关重要。

设计人员必须小心地保持 USB 数据走线干净，并使用极少的过孔（如果必须使用则成对），使其布线远离高速信号。DP/DM 信号必须具有干净完整的接地平面，以在信号后面或上方的层中作为参考。这些信号上的差分电阻必须尽可能接近 90Ω，这样可以使用大多数 PCBCAD 软件计算出该值。

3.3 USB 时钟实现

USB 对时序有严格的要求，但由于 USB 主机和 USB 设备在总线时序和同步中的角色不同，这些要求存在显著差异。概括来说，主机是总线上的时序机构，而设备则自行与主机的时序同步。

USB 主机负责为全速设备以精确的 1kHz 速率生成帧起始 (SOF) 数据包。SOF 数据包定义了 USB 总线的全局时基，因此主机时钟必须独立满足 USB 精度要求，而不依赖于总线的任何外部基准。主机具有严格的时序要求，因此需要高精度外部振荡器。该稳定时钟可用作 MSPM0 FLL 或 PLL 的基准输入，从而合成 USB 模块所需的更高频率的时钟。内部 LFOSC 不足以充当此角色，因为频率误差和温度漂移超过长期 SOF 时序精度可接受的值。

另一方面，USB 设备不会生成自己的 SOF 数据包，因此不是时序主器件。这使得该设备能够从主机生成的 SOF 数据包中获取其时序。MSPM0 将 SOF 数据包用作 USBFLL 的输入基准，该基准将用作时钟源。设备通常不需要精度极高的外部振荡器，因为由主机确定其时序，允许使用精度较低的内部时钟。MSPM0 上的 USB 外设包含一个以 60MHz 运行的外设 *USBCLK* 所特有的时钟。USBCLK 需要一个精确的基准时钟，该时钟可由 SYSPLL 块或 USB 专用 USBFLL 提供，与 PLL 相比，它通过 FLL 协议达到更低的功耗。有关时钟树的更多信息，请参阅技术参考手册的第 2.3 节。

3.3.1 选择时钟源

选择时钟源时最大的考虑因素是精度。USB 规范要求时钟的容差达到 2500ppm。超出此规范的源可能会导致一致性能降低，并且无法验证 USB 合规性。

有三个选项可为基准时钟提供时钟源：

表 3-1. 时钟源差异

源	频率范围	使用场合
外部时钟源	4-48MHz 晶体振荡器，PLL 至更高频率	晶体振荡器具有出色的精度和 USB 合规性，让用户能够满足 USB 规范的要求。
内部时钟源	4-32MHz，PLL 至更高频率	通常不推荐，因为内部时钟经常超出 2500ppm，可用于原型设计。
USBFLL	48 至 60 MHz	没有外部晶体空间，因为该器件利用 SOF 数据包进行同步。但是，空间和成本受限的设计需要有源 USB 连接来实现时钟同步。

如果使用外部晶体，请确保遵循晶体振荡器指南以获得出色性能。

3.3.2 选择时钟频率

由于 MSPM0 的 USBFLL 时钟源利用 SOF 数据包进行同步，因此可以获得很宽的频率范围。每个 MSPM0 头文件都包含 SYSPLL 或 USBFLL 源的预定义常量，SYSPLLCLK1 上还提供额外配置，支持高达 32 的扩展数。

虽然 USB 功能需要 48MHz 的频率，但有多种方法可以实现此目标。通常，使用 48MHz 晶体可能不在设计范围之内，尽管具有信号完整性和时钟简单性的优势，但它们往往更昂贵、EMI 更高且设计要求更严格。

通常，设计人员在 PLL 中使用多个 HFXT 晶体（从而产生 SYSPLLCLK），让用户可以采用较低频率晶体的组合，例如在一个 PLL 中使用六个 8MHz 晶体或在一个 PLL 中使用三个 16MHz 晶体。它们的优点是成本更低，EMI 更低。但是这会增加时钟树的复杂性和布局复杂性。

在许多情况下，利用 USBFLL 时钟源是器件的可靠选择。由于没有额外的晶体（节省空间和金钱），它非常适合空间和成本受限的器件。但是，必须考虑到延迟更高，因为 FLL 需要多个 SOF 帧来锁定并与主机同步。这对于仅在连接到 USB 时运行的总线供电设备来说是非常好的选择。

3.4 实现示例

图 3-2 是基于评估套件 LP-MSPM0G5187 的伪原理图：

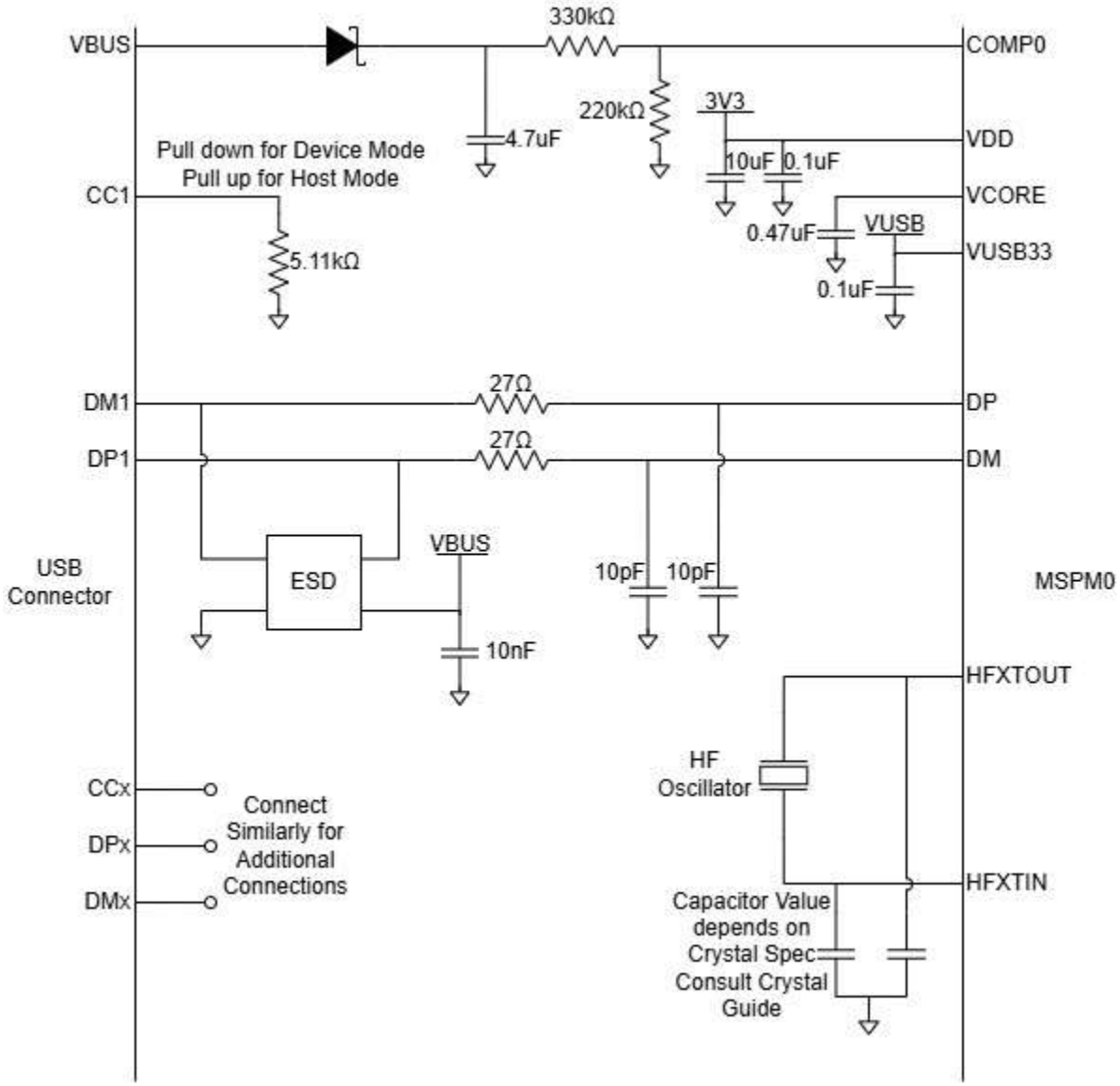

图 3-2. 器件配置参考设计示例

图 3-2 虽经简化，但仍有多种重要特性，需要在电路设计的实现中关注和验证。

- 在上一个示例中，VDD、VCORE 和 VUSB33 保持断开状态。按照器件的数据表验证所使用的去耦电容值是否正确。
- D+ 必须在器件侧的 DP 中上拉，因为这是在建立新连接时通知主机的信号。
- ESD 在 USB 设备中至关重要，尤其是鼠标、键盘和耳机等通常由人处理的设备。图 3-2 中的器件是 TDP4E05U06。
- 连接到 HFXTOUT 和 HFXTIN 的外部时钟源可以是晶体、谐振器或外部时钟源。如果使用晶体，请按照 节 3.3.1 中的晶体指南操作。

4 软件概述

4.1 USB 堆栈：特性

TinyUSB 库是使用 MSPM0SDK 开发 USB 设备的基础。此 API 支持四种最常见的 USB 设备类：

- 通信设备类 (CDC)：用于串行通信设备的 USB 设备类。CDC 允许 USB 设备模拟传统的串行端口 (COM)，从而实现主机和设备之间的数据交换。
- 人机接口设备类 (HID)：专为键盘、鼠标和游戏控制器等用户输入设备而设计的 USB 设备类。HID 设备采用标准化协议，允许在大多数操作系统上实现无驱动程序安装。
- 音频设备类 (UAC)：用于在主机和设备之间发送和接收数字音频数据的 USB 设备类。UAC 允许 USB 设备在主机上显示为标准音频 IO 设备。
- 大容量存储类 (MSC)：此 USB 设备类允许设备显示为主机操作系统的外部存储驱动器，从而实现标准文件系统操作。

这些类为通用用途提供了良好的选择。有关如何选择接口的讨论，请参阅*选择设备类*部分。

API 的特性包括：

- 跨平台支持
- 主机和设备堆栈
- 多种设备类
- 干净且可读的代码库
- 已获 MIT 许可
- 极低的资源要求
- 支持多配置

MSPM0SDK 中提供了多个示例，在撰写本文档时包含的示例如下：

表 4-1. MSPM0SDK 中提供的示例

CDC 示例	HID 示例	MSC 示例	UAC 示例	通用示例
cdc_acm_uart	hid_cdc_composite_ti	msc_dual_lun	uac_microphone_i2s	device_billboard
cdc_dual_ports	device_hid_composite	sd_card_bridge	uac2_headset	
hid_cdc_composite_ti	device_hid_generic_inout	msc_file_explorer	uac2_speaker_fb	
billboard_cdc	hid_keyboard_ti		audio_test	
	hid_mouse_ti			
	hid_multiple_interface			

这些示例可使用 MSPM0G5187 LaunchPad 用于快速测试和开发。有关开始原型设计的更多信息，请参阅节 5。

4.2 SysConfig 描述符工具

TI 提供使用 *SysConfig* 进行简单描述符配置的功能。利用该工具，可以在 MSPM0 器件上配置许多外设，对于支持 USB 的 MSPM0 器件，可从标题为 *TinyUSB* 的章节中了解描述符的配置和生成，而无需手动对其进行编码。

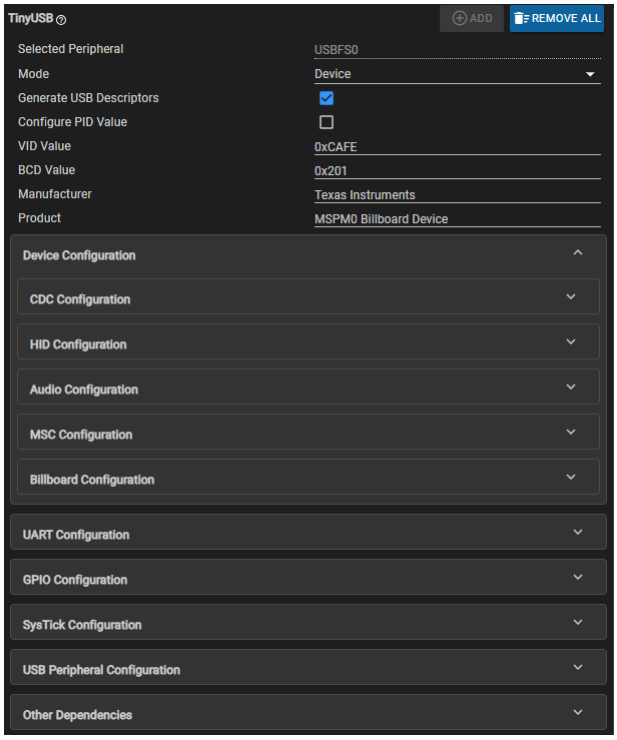

图 4-1. 通过 Code Composer Studio 提供的 SysConfig 编辑器屏幕截图

利用该工具可以配置以下功能：

- 模式：设备/主机
- VID 值
- BCD 值
- 产品和制造商代码
- 设备类配置
  - RX FIFO 大小（仅限 CDC）
  - TX FIFO 大小（仅限 CDC）
  - 端点传输缓冲区大小
  - 启用扬声器反馈、编码和解码（仅限 UAC）
- 默认和替代告示板标识符
- 配置 UART 实例和引脚多路复用选择
  - 时钟源
  - 时钟分频器
  - 波特率
  - 字长
  - 奇偶校验
  - 停止位
  - 硬件流控制
  - DMA 配置
  - 中断配置
- 配置 GPIO 依赖项
  - 方向
  - 初始值

  - Pinmux
  - 中断
  - 内部电阻器
- 配置 Systick 依赖项
- 配置 USB 时钟源

利用 SysConfig 生成 USB 描述符会很有帮助，因为撰写描述符可能很繁琐，而且容易导致其他错误。错误描述符导致的故障往往会被混淆，而跟踪这些错误可能会导致调试时间显著增加。

SysConfig 一上手就能为 CDC、HID、UAC 和 MSC 接口的任意组合生成可靠的描述符。

将该工具视为构建与应用交互的 USB 接口。这是开发 MSPM0 TinyUSB 工程的第一步。

4.3 选择设备类

创建 USB 设计的第一步是选择正确的设备类，因为这种选择将设备限制在 表 4-2 中列出的特定特性集范围内。

表 4-2. 支持的四个设备类之间的比较

特性	CDC（通信器件类）	HID（人机接口设备类）	MSC（大容量存储类）	UAC（音频设备类）
在主机上生成的接口	虚拟 COM 端口	人机接口设备	存储卷	音频设备
此接口的行业专业知识	COM 端口在行业中很常见；得到广泛支持和充分了解	与 COM 端口或存储卷不同，HID 接口在一定程度上为 USB 专用，在业界不太知名	存储卷在行业中很常见；得到广泛支持和充分了解	UAC 设备已使用近二十年；得到广泛支持和充分了解
在主机安装	Windows PC 必须经过需要最终用户交互的设备安装过程。(1) 需要此 Windows PC 的管理员权限 尽管 Windows 中实际上已经有二进制文件，但用户还必须提供一个 INF 文件	在大多数操作系统中以静默方式加载 – 只需开始工作。 无需驱动程序文件	在大多数操作系统中以静默方式加载 – 只需开始工作。 无需驱动程序文件	在大多数主机上支持符合类标准的加载，无需安装
最终用户如何与之交互	主机上连接 COM 端口的应用程序 该应用程序可以是定制应用程序，也可以是任何使用 COM 端口的现有应用程序	主机上与 HID 器件连接的定制应用程序	设备将存储卷挂载到系统；应用程序读取和写入卷上的文件。 该应用程序可以是定制应用程序，也可以是读取或写入文件的任何应用程序	音频通过 UAC 接口流式传入或传出设备
驱动程序认证需求	除非 INF 文件经过 WHQL 认证（已签名），否则 Windows 会报告驱动程序“未经认证”	未生成*未经认证*消息	• 未生成*未经认证*消息	• 未生成*未经认证*消息
代码占用空间和复杂性	代码占用空间小（4K 至 6K） 架构简单	代码占用空间小（4K 至 6K） 架构简单	代码占用空间更大（8K 至 15K） 需要文件系统，这会增加成本、大小和复杂性。	占用空间更大 需要接口等时传输，这会增加复杂性
吞吐量	较快（数百 KB/秒） 使用批量 USB 传输。	较慢（64 KB/秒） 使用中断 USB 传输。	较快（数百 KB/秒） 使用批量 USB 传输。	取决于采样率和位深度 通常为数百 KB/秒
适用于主机与设备之间的点对点通信	是	是	否	是
适合批量数据传输	是	否	是	是（音频）

4.3.1 决定 USB 设备类的示例过程

有时，设备类的选择很明确。在其他情况下，应用可被视为通用，让开发人员有多种选择。尽管有许多方法可以做出此决定，但用户可以使用 图 4-2 来选择设备类。

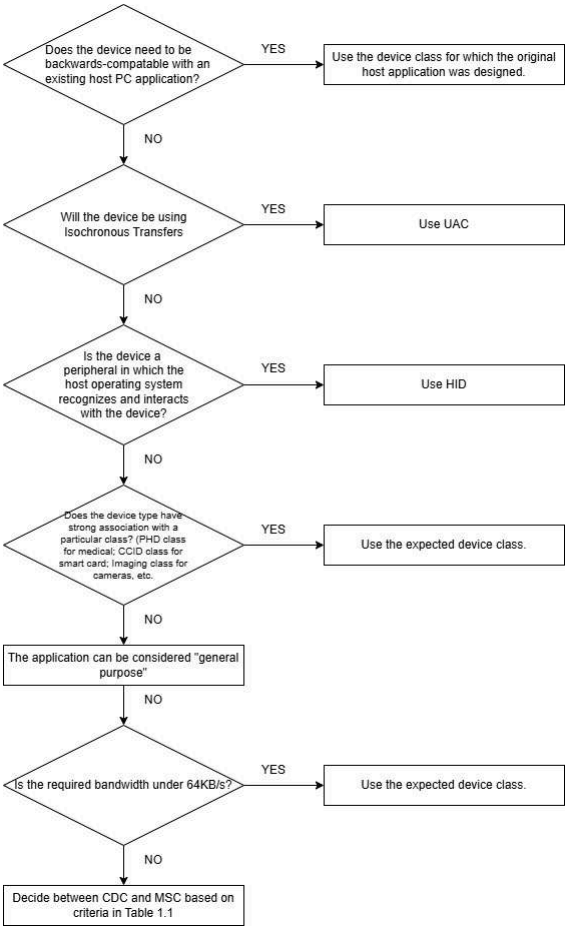

图 4-2. 选择正确设备类的决策树示例

4.4 如何选择供应商 ID (VID) 和产品 ID (PID)

关于 USB 的一个常见问题是如何选择 VID 和 PID。当 USB 设备连接到主机时，主机会请求询问 USB 描述符。这些描述符向主机报告设备的性质和设备的功能。

描述符中包含 16 位 VID 和 PID 值。VID 与特定供应商和 OEM 关联，PID 与该供应商销售的产品关联。

例如，如果供应商 *Vendor1* 销售第一个 USB 产品 (*Product1*)，则供应商获得一个 VID，它现在与该公司关联；然后供应商需要选择一个 PID 与 *Product1* 关联。当供应商稍后发布 *Product2* 时，供应商使用相同的 VID，但现在应使用新的 PID。供应商有责任确保自己不会重复使用 PID，那可能会导致现场冲突。

因此，VID 和 PID 的独特组合使 USB 主机能够区分不同的 USB 产品类型。如果 *Product1* 和 *Product2* 的 VID 和 PID 相同，并且现场的主机遇到这两个产品，则可能会导致冲突，因为主机会混淆这两个产品并加载不适当的驱动程序。一般而言，如果设备的 USB 描述符存在任何差异，则设备必定具有不同的 PID。

4.4.1 选择并获取 VID 和 PID

VID 由 USB Implementers Forum (USB-IF) 分配，该论坛是负责监督 USB 的标准机构。供应商可以选择通过加入 USB-IF 来获取 VID，或者在不加入的情况下授予 VID 许可。在撰写本文时，前者的费用为每年 5000 美元，后者的两年期许可证费用为 3500 美元。（更多详情，请访问 http://www.usb.org/developers/vendor/。）

与前代 MSP430 器件不同，MSP 不再运行 VID 共享计划，在该计划中客户可以在线申请其唯一 PID 并在 TI 的 VID 下使用。如上所述，客户可以自由向 USB-IF 注册 VID，并将该 VID 和相关 PID 用于我们的器件。

4.4.2 在开发过程中使用 VID 和 PID

在 USB 设备上具有唯一 VID 和 PID 对对于防止冲突非常重要。在首次与给定的 USB 和 PID 接触后，给定的 VID 主机存储有关 USB 设备驱动程序要求的信息。它必须能够假设任何具有相同 VID 和 PID 的后续设备都需要相同的主机驱动程序设置。因此，一旦产品上市，就不得更改产品的 VID 和 PID。

但在开发过程中，随着开发人员到达最终的 USB 描述符集，有时可能需要更改 VID 和 PID。开发人员必须防止所使用的主机发生冲突。这可以通过在 USB 描述符更改时随时使用新 PID 值来实现，也可以使用原始 PID，但必须从系统中卸载并重新安装设备。有关更多信息，请参阅 USB 开发包中的《USB API 编程人员指南》。

4.5 TinyUSB API 编程人员指南和示例

SDK 中的 TinyUSB 用户指南包含有关使用 TinyUSB 开发软件以及对 表 4-1 中提到的许多示例进行扩展和对 图 4-1 中的 USB 描述符工具进行分解的详细信息。

此外，通过 TinyUSB 网站还可获得许多在线示例以及对 API 更深入的探索。其中包括许多其他定义、软件概念以及与应用设计相关的资源。

5 入门：评估 MSPM0 USB

LP-MSPM0G5187 评估套件包含开始评估以及开发主机和设备设计所需的所有分线选项。

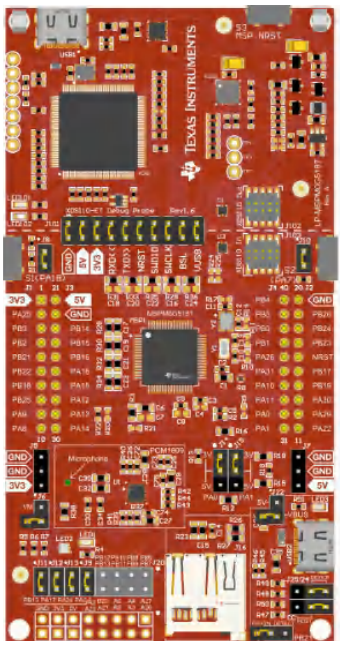

图 5-1. LP-MSPM0G5187 评估套件

该评估套件包含多个跳线，可用于调试许多硬件和软件功能：

- 适用于 UAC 设备的麦克风和基于 I2S 的音频 ADC
- 适用于 MSC 应用的 microSD 插槽
- 两个 LED (一个支持 RGB 的 LED)
- 三个按钮
- 40 多个可用于测试通用器件的引脚。
- 两个 USB-C 连接器，一个用于调试，一个用于 MSPM0 连接
- 用于编程、调试和 EnergyTrace™ 技术的板载调试探针。

得益于 LaunchPad 开发套件上的 40 引脚扩展接头，可以连接市面上的各种 BoosterPack™ 插件模块，因此快速原型设计变得非常简单。您可以快速添加无线、显示、传感器等功能。还可以设计 BoosterPack 插件模块，或者从 TI 和其他方已有的众多插件模块中进行选择。该 40 针接口兼容所有符合标准的 20 引脚 BoosterPack 插件模块。

TI 的 Code Composer Studio IDE (CCS) 是使用 LaunchPad 进行设计和测试的主要开发环境。USB API 示例通过 MSPM0SDK 提供，下载最新版本通常是确保获得最新示例的理想选择。

6 总结

MSPM0 系列支持 USB 的器件能够支持多种类型的应用，包括音频、人机接口和通信设备，以及大容量存储主机。USB 2.0 全速模块具有最多 16 个端点，以及专用端点 RAM、DMA 触发器、集成式全速 PHY，并支持 DFU/BSL。有关每个模块的更多信息，请参阅与 USB 器件相关的技术参考手册和数据表。MSPM0 的 USB 外设与 CCS 开发环境和 TinyUSB 架构相结合，当与本应用手册中的硬件指南配合使用时，可以简化开发，从而加快 USB 应用的开发速度。这些设计中有许多重要的注意事项，包括 USB-C 模式指示、自供电和总线供电电源架构注意事项以及布线和布局规则。虽然 MSPM0 中的时钟模块能够处理器件应用，但需要仔细考虑以确保符合 USB 协议。

7 参考资料

1. 德州仪器 (TI)，使用 MSP430 MCU 开始 USB 设计，应用手册。
2. 德州仪器 (TI)，*LP-MSPM0G5187 LaunchPad*，开发套件。
3. 德州仪器 (TI)，*MSPM0 G 系列 80MHz 微控制器*，技术参考手册。
4. 德州仪器 (TI)，*MSPM0 Academy*，资源浏览器。
5. TinyUSB，*TinyUSB 概念*，网页。

8 USB 术语表

1. 批量传输：USB 总线上的四种数据传输类型之一。批量传输专为移动大量数据而设计。它们能够在总线上使用任何可用带宽（即其他传输类型尚未使用带宽）。这使得它们能够实现最高的数据速率，但无保留带宽，因此在繁忙的总线上，批量传输可能获得小带宽或出现高延迟。传输类型由 USB 接口类型的选择决定；例如，CDC 和 MSC 接口使用批量传输。
2. 复合 USB 设备：包含多个 USB 接口（例如，两个 CDC 接口或 CDC+HID）的物理 USB 设备（一个 USB 连接器）。主机将每个接口枚举为单独的逻辑实体。
3. 控制传输：USB 总线上的四种数据传输类型之一。控制传输处理设置连接的管理任务，如报告 USB 描述符。主机还发送其他 USB 设备请求，然后该设备使用控制传输进行响应。有一个专用于这些传输的 USB 端点：端点 0 (EP0)。
4. 设备类：针对一类设备定义的 USB 协议。常见的设备类包括通信设备类 (CDC)、人机接口设备 (HID) 类、USB 音频类 (UAC) 和大容量存储类 (MSC)。
   a. CDC：用于串行通信设备的 USB 设备类。CDC 允许 USB 设备模拟传统的串行端口 (COM)，从而实现主机和设备之间的数据交换。
   b. HID：专为键盘、鼠标和游戏控制器等用户输入设备而设计的 USB 设备类。HID 设备采用标准化协议，允许在大多数操作系统上实现无驱动程序安装。
   c. UAC：用于在主机和设备之间发送和接收数字音频数据的 USB 设备类。UAC 允许 USB 设备在主机上显示为标准音频 IO 设备。
   d. MSC：此 USB 设备类允许设备显示为主机操作系统的外部存储驱动器，从而实现标准文件系统操作。
5. 设备安装：首次枚举 USB 设备时，主机可以执行一次性功能来安装该设备。例如，Windows 使用设备的 VID 和 PID 作为索引，在系统注册表中记录有关设备的信息。在随后的枚举中，主机从注册表中获取有关设备的大部分信息。设备安装可能静默进行（大多数情况下对最终用户不可见），或者对于 Windows 中的 CDC，可能需要用户操作。
6. 端点：管道的末端。它充当该管道的 USB 设备上的*邮箱*。设备通常具有多个活动端点。当主机在总线上通信时，它首先标识物理 USB 设备，然后标识该设备内要与之通信的端点编号。根据创建的 USB 接口为端点分配特定的功能。HID/MSC 各使用一个输入端点和一个输出端点，而 CDC 使用两个输入端点和一个输出端点。在 MSP430 API 堆栈中，端点管理由描述符工具完全自动化。
7. 枚举：主机查询物理 USB 设备以确定它是什么并加载适当驱动程序以使主机应用程序能够与之对接的过程。每次连接设备时都会发生枚举。
8. 中断传输：四种 USB 数据传输类型之一。中断传输专为延迟、带宽和传输而设计。但是，带宽限制为每帧 (1ms) 仅一个 USB 数据包（全速 USB 为 64 字节）。传输类型由 USB 接口类型的选择决定；例如，HID 接口使用中断传输。
9. 等时传输：四种 USB 数据传输类型之一。等时传输可保证延迟和带宽，但不保证传输。也就是说，如果错误检查显示数据受损，则不会重试尝试。这种类型适用于音频和视频流式传输 - 在此类应用中，重试会导致中断，因此与丢失数据包相比，对用户来说更为明显。
10. INF (*.inf) 文件：在 Windows 上安装任何 USB 设备期间所需的基于文本的文件，允许 Windows 将设备与特定驱动程序相关联。对于某些设备类，Windows 内部包含 INF，允许静默设备安装。对于 CDC，Windows 会提示最终用户提供 INF 文件。
11. 管道：主机和设备之间的单线通信线路。管道要么是输入（进入主机），要么是输出（离开主机）。它们具有特定传输类型（例如，批量或中断）的特征。
12. 描述符：USB 设备在枚举期间提供给主机的数据结构，描述设备的功能、配置、接口和端点。通用描述符包括：设备、配置、接口和端点描述符。
13. 全速 USB：USB 以 12Mbps 的速度运行，这是 USB 1.1 的标准速度和 USB 2.0 设备的可选速度。
14. 主机：负责管理总线并启动所有数据传输的 USB 控制器。通常是 PC，但也可以是具有 USB 主机功能的嵌入式系统。

重要通知和免责声明

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出任何明示或暗示的担保，包括但不限于对适销性、与某特定用途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、安保法规或其他要求。

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 产品的相关应用。严禁以其他方式对这些资源进行复制或展示。您无权使用任何其他 TI 知识产权或任何第三方知识产权。对于因您对这些资源的使用而对 TI 及其代表造成的任何索赔、损害、成本、损失和债务，您将全额赔偿，TI 对此概不负责。

TI 提供的产品受 TI 销售条款)、TI 通用质量指南 或 ti.com 上其他适用条款或 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。 除非德州仪器 (TI) 明确将某产品指定为定制产品或客户特定产品，否则其产品均为按确定价格收入目录的标准通用器件。

TI 反对并拒绝您可能提出的任何其他或不同的条款。

最后更新日期：2025 年 10 月