AN1368

采用集成图形控制器的 PIC® 单片机开发嵌入式图形应用

作者： *Pradeep Budagutta*
Microchip Technology Inc.

简介

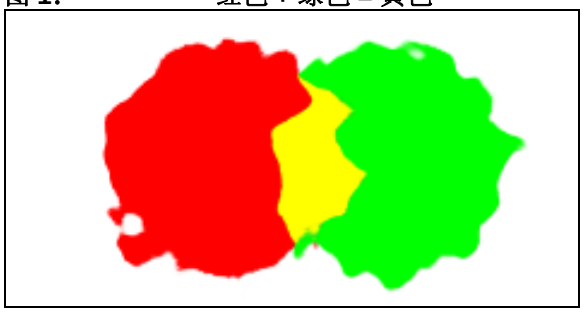
支持图形的设备在日常生活中得到广泛应用。这些设备随处可见，包括电话、计算器、传呼机、 MP3 播放器、数字电表、智能遥控和 UPS 显示屏等室内产品。它们还用于户外产品，如交通信号灯、计程器、公共汽车显示屏、广告牌等。这类的应用几乎举不胜举。目前的趋势是，因为其价格经济、易于使用且是最新技术，许多现有设备都将支持图形。

本应用笔记旨在帮助工程师设计自己的第一个图形应用。它描述了图形应用的基本定义和术语，并帮助工程师了解设计原理、必要的决定因素、硬件注意事项，以及可用的单片机和开发工具。可从Microchip获得软件库和支持，并为高级用户提供更深入的参考文献。

色彩学的基础知识

在人类的可见光范围内（波长范围从约 400 nm（紫光）到 700 nm（红光），以黄光（大约 575 nm）为中心），单纯的颜色与光的波长相关。这意味着，波长为 575 nm 的光进入人眼时，被感知为黄光。我们还知道，色彩源自三种基本颜色：红色、蓝色和绿色。例如，黄光可通过混合红光和绿光得到。这是真的吗？回答是否定的，也是肯定的。否定是因为红光和绿光混合将构成波长为 700 nm 和 560 nm 的光，这不是表示黄色的波长。肯定的回答是因为人眼将该混合光感知为黄色光。因此，我们将红光和绿光的混合看成单一的黄光，如图 1 所示。这是由人眼的颜色辨别特性造成的。

图 1：　　红色 + 绿色 = 黄色



人眼将该光感知为黄色光，而不是单独的红色光和绿色光。人眼的此颜色辨别特性是 RGB （红色、绿色和蓝色）模型的基础。该模型表示，人眼可将相应比例（饱和度）的红色、蓝色和绿色混合来感知不同的颜色。因此，可通过混合不同比例的红色光、绿色光和蓝色光形成彩色光。

- 混合相同比例的三种 RGB 颜色形成灰色
- 混合零量的所有 RGB 颜色形成黑色
- 混合最大量的所有 RGB 颜色形成白色

在保持相同比例的 RGB 的同时改变光的亮度，将形成不同深浅的灰色，也称为灰度。整个应用使用一种颜色（固定比例的 RGB）即为单色应用，意味着使用单一颜色。

由于数字系统中的一切均以位和字节表示，因此如何将实际的颜色表示为位或字节形式的数字呢？这三种基本颜色 （RGB）中的每一种都代表一个字节 （0 到 255 范围内的一个数）。因此，通过 3 个字节，我们可表示 1600 万种颜色 （2^{24}），这被称为真彩色。通常还使用 16 位来表示颜色。通过 16 位，我们可表示 64K 种颜色（2^{16}），这对于许多图形应用已足够。

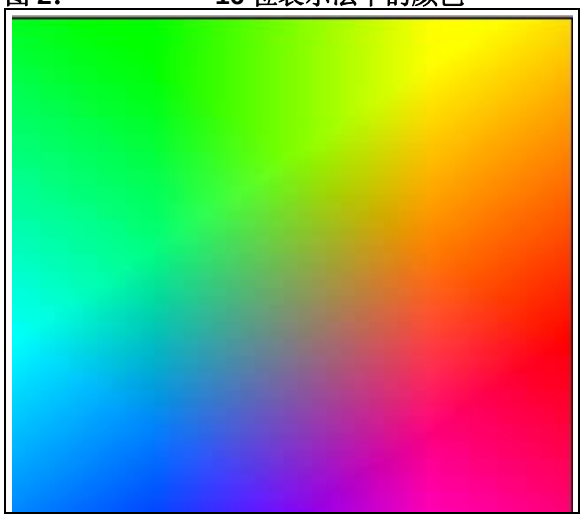
© 2012 Microchip Technology Inc.

通常，使用以下两种方案在红色、绿色和蓝色之间分配 16 位：

- **方案 1（R<5> G<6> B<5>）：**在该方案中，红色为 5 位，紧接着是 6 位绿色和 5 位蓝色。由于人眼的特性，相对红色和蓝色，它可分辨出更多色调的绿色，所以分给绿色更多的位。图 2 给出了这 64K 种颜色。
- **方案 2（T<1> R<5> G<5> B<5>）：**在该方案中，有一个透明位，紧接着是红色、绿色和蓝色各 5 位。透明位用于指示该颜色是否可用。

目前， Microchip 图形库 （版本 2.11）仅支持方案 1。

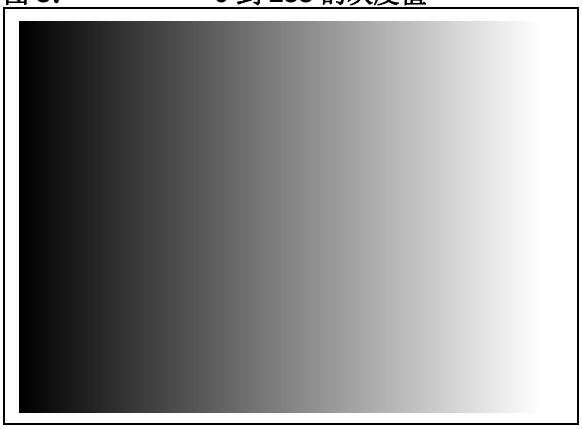
图 2：　　16 位表示法中的颜色



灰度通常用一个字节表示， 0 表示黑色， 1-254 表示不同色调的黑色，随着数字的增大亮度也越高，而 255 表示白色，如图 3 所示。有时，只使用 4 位或 2 位来分别表示 16 种或 4 种色调的黑色。如果只使用一个位来表示一种颜色的有无，则称为单色。

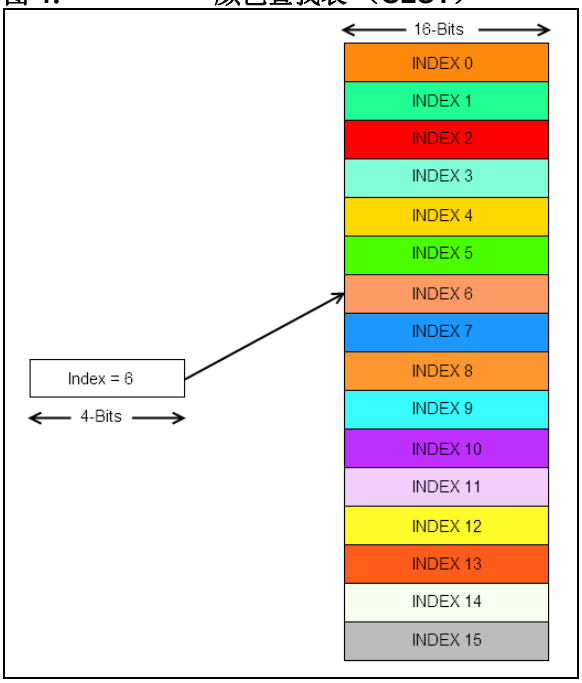
表示一种颜色所需的位数称为颜色深度。例如，16 位的颜色深度意味着需要 16 位来表示一个颜色，因此我们可以表示 2^{16} 种不同颜色。

图 3：　　0 到 255 的灰度值



此外，可使用颜色查找表 （Color Look-up Table，CLUT）来表示颜色，该表也称为调色板表，其中颜色通过表索引指定，如图 4 所示。根据表大小的不同，用于表示索引的位会有所不同，可为 256 个 RGB 条目（8 位索引）、16 个 RGB 条目（4 位索引）、4 个 RGB 条目 （2 位索引）和 2 个 RGB 条目 （1 位索引）。该方案主要用于节省存储器。有关该方案的更多信息，请参见**附录 A：“ 颜色查找表 （CLUT） ”**。

图 4：　　颜色查找表 （CLUT）



© 2012 Microchip Technology Inc.

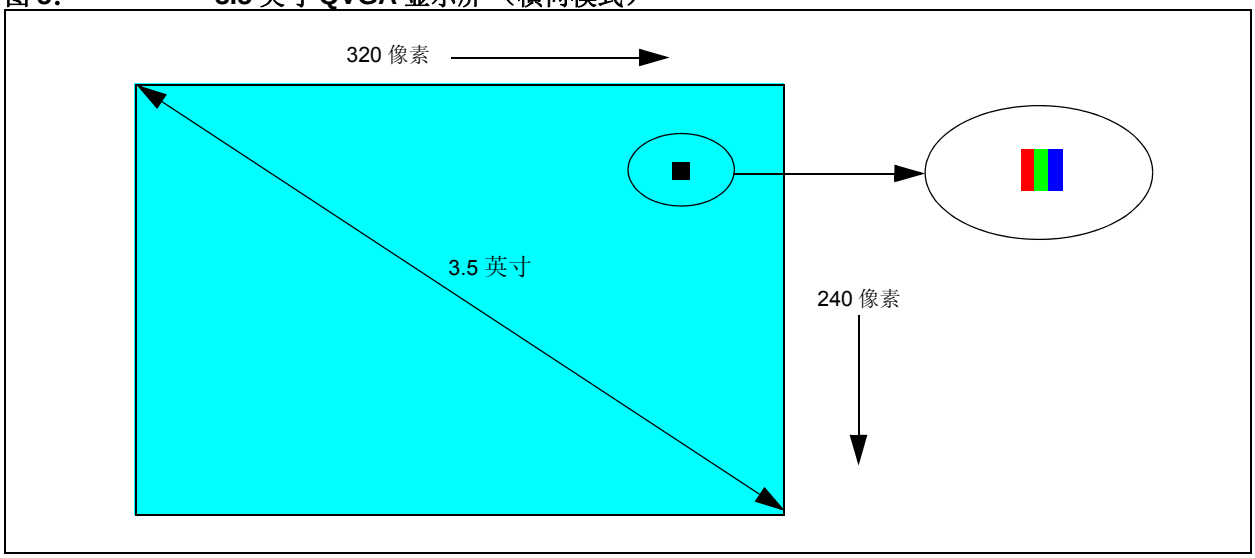
基本显示术语

屏幕由离散的元素组成，这些元素称为像素。每个像素可以显示一个颜色点，并且每个像素由三个点组成：红色、绿色和蓝色。这些颜色在彩色屏幕上彼此相邻排列，为灰度屏幕上的一个亮度点，或为单色屏幕上一个设置为亮 / 暗的点。此类像素在水平和垂直方向的数量称为屏幕分辨率。例如，分辨率为 320x240 表示水平方向像素为 320（列数），垂直方向像素为 240（行数）。标准分辨率具有指定名称，例如 QCIF （176x144） 、CIF （352x288） 、 QVGA （320x240） 、 WQVGA （480x272）、VGA（640x480）和 WVGA（800x480）等。提及分辨率时，通常最好给出像素数，而不是这些名称。

屏幕可为横向模式 （宽 > 高）或纵向模式 （高 > 宽）。显示屏的可见宽度与可见高度的比例称为宽高比。最常用的宽高比为 4:3。显示屏的对角线长度称为显示屏的长度。

例如，显示屏为 3.5 英寸表示显示屏的对角线长度为 3.5 英寸，如图 5 所示。

图 5： 3.5 英寸 QVGA 显示屏 （横向模式）



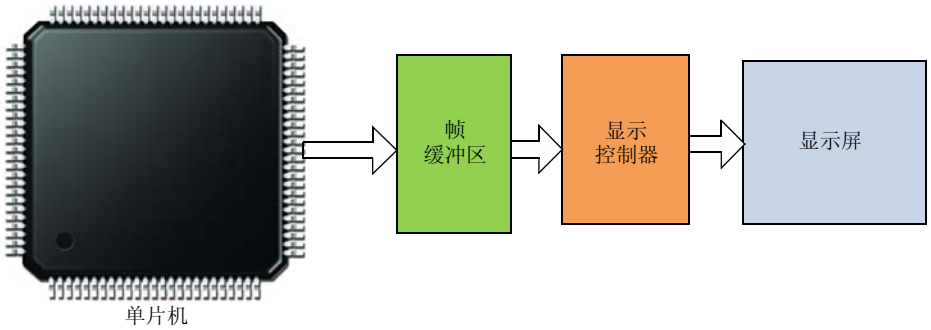
图形子系统硬件

将在以下小节中描述图形应用所需的硬件组件以及这些组件的互连和设计决策。

图形系统的组件

所有嵌入式图形系统中都有4个基本组件，如图6所示。它们由显示屏、显示控制器、帧缓冲区和单片机组成。

图 6：　　图形系统的 4 个基本组件



显示屏

显示屏是具有下列功能的器件，以像素显示一系列颜色，并且还将数字表示的颜色转化为实际颜色。在术语中，提到颜色就包含灰度。通常，所使用的显示屏类型为 TFT LCD、 CSTN/MSTN LCD 或 OLED/AMOLED。

所有LCD模块都包括栅极和源极驱动器来驱动电压和电流以显示所有像素。表 1 给出了不同显示技术的简要比较。

表 1：　　比较不同的显示技术(1)

属性	TFT LCD	STN （CSTN/MSTN）	AMOLED
帧速率	高	低	高
重影	无	有	无
HBLANK 和 VBLANK	有	无	有
背光	需要	需要	不需要
成本	中等	低	高
QVGA （320x240）分辨率的典型尺寸	1.5 英寸到 5.7 英寸	1.5 英寸到 5.7 英寸	最多 2.8 英寸
对比度	中等	低	高
功耗	高	中等	低
视角	中等	中等	高

注　1： 本表中提到的数据可能因显示技术的不断变化或改进而发生更改。

© 2012 Microchip Technology Inc.

下面将诠释显示技术的某些重要属性：

- **帧速率（Frame Rate）：**显示屏一秒内刷新的次数（此参数不反映单片机的刷新能力，仅反映显示屏的刷新能力）。
- **重影（Ghosting）：**当屏幕变化时，前一帧以较低的亮度显示一小会，看起来就像当前屏幕的影子。
- **HBLANK 和 VBLANK：**水平和垂直消隐周期，此时不更新显示内容。更多信息，请参见《PIC24F 系列参考手册》中的**第 43 章“ 图形控制器模块（GFX）”**（DS39731）。
- **背光（Backlight）：**对于 TFT/STN LCD，必需有背光才能查看显示内容。可采用 CCFL 或 LED 提供背光。
- **对比度（Contrast）：**定义为显示屏上白色与黑色的亮度比。对比度越高，显示质量越好。
- **视角（Viewing Angle）：**是可正常查看显示内容的水平或垂直角度。

显示屏没有内置存储器，必须不断更新每行和每列的像素数据，如果更新失败，将显示空白（类似于 DRAM 刷新）。显示数据的刷新由显示控制器处理。

显示控制器

显示屏必须通过反复地从帧缓冲区获取水平和垂直像素数据来不断刷新。该任务由显示控制器执行。它从帧缓冲区获取数据，将数据解码为所需的位格式，然后将其与适当的控制信号一起提供给显示屏。显示控制器必须符合显示屏的时序要求。

帧缓冲区

帧缓冲区为一块存储区（通常是 RAM），保存要显示在显示屏上的数据并充当显示控制器的数据源。所需的帧缓冲区大小取决于分辨率和颜色深度。最低要求是，应能够保存显示一个完整帧所需的数据，且必须可支持显示控制器的扫描速率（根据显示屏的数据手册选择的首选刷新速率）。

公式 1：

所需的帧缓冲区大小（字节）= 像素数 x 颜色深度（位）/8

例 1：

对于使用 16 bpp 颜色深度的 QVGA（320x240）显示屏：

帧缓冲区 = 320 x 240 x 16/8
= 153,600 字节
= 150 KB

例 2：

对于使用 16 bpp 颜色深度的 WQVGA（480x272）显示屏：

帧缓冲区 = 480 x 272 x 16/8
= 261,120 字节
= 255 KB

例 3：

对于使用8 bpp颜色深度的QCIF（176x144）显示屏:

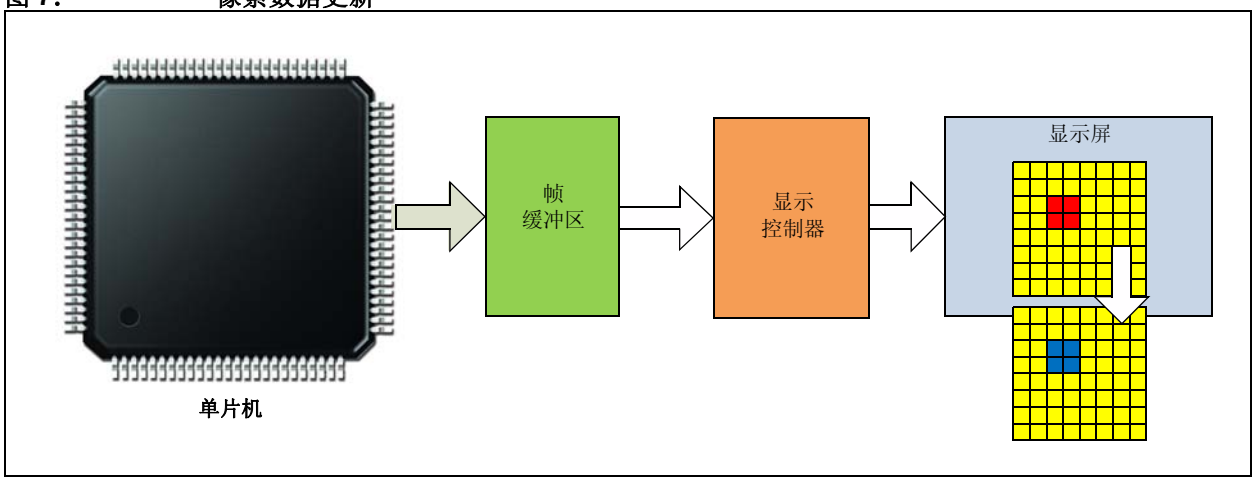
帧缓冲区 = 176 x 144 x 8/8
= 25,344 字节
= 24.75 KB

单片机

单片机内部运行的应用代码决定哪些数据应保存在帧缓冲区中，并且当帧缓冲区的内容变化时，显示内容也会随之变化。计算每个像素的颜色并存储在帧缓冲区中。就所使用的颜色深度和帧缓冲区的存储范围来说，单片机和显示控制器必须具有相同的设置。单片机必须具有足够的处理能力（通常以 MIPS 来衡量）以将所需形状所对应的数据及时送入帧缓冲区，这样，便不会看似形状是缓慢的绘制在显示屏上。这是因为在单片机将像素数据传送到帧缓冲区的同时，显示控制器不断从帧缓冲区读取数据。然而，如果显示屏没有任何变化，那么表示单片机没有将任何新的形状数据传送到帧缓冲区中。如果屏幕发生变化，那么只有更改的像素需要发送到帧缓冲区，从而最大限度地减少到帧缓冲区的数据传输。

在图 7 中，只有 4 个像素将发生变化，颜色深度为 16 位时， 4 * 16/8 = 8 字节需要发送到帧缓冲区。

图 7：　　像素数据更新



© 2012 Microchip Technology Inc.

基本组件的整合

如何整合 4 个基本组件是设计图形应用中最重要的步骤。要作出选择，设计人员需要了解这些可能是 IC 形式的基本组件的组合类型。有四种可能的组合类型，如图 8 所示。

表 2 列出了这 4 种组合各自的优缺点。

图 8：　　整合基本图形组件的不同方式

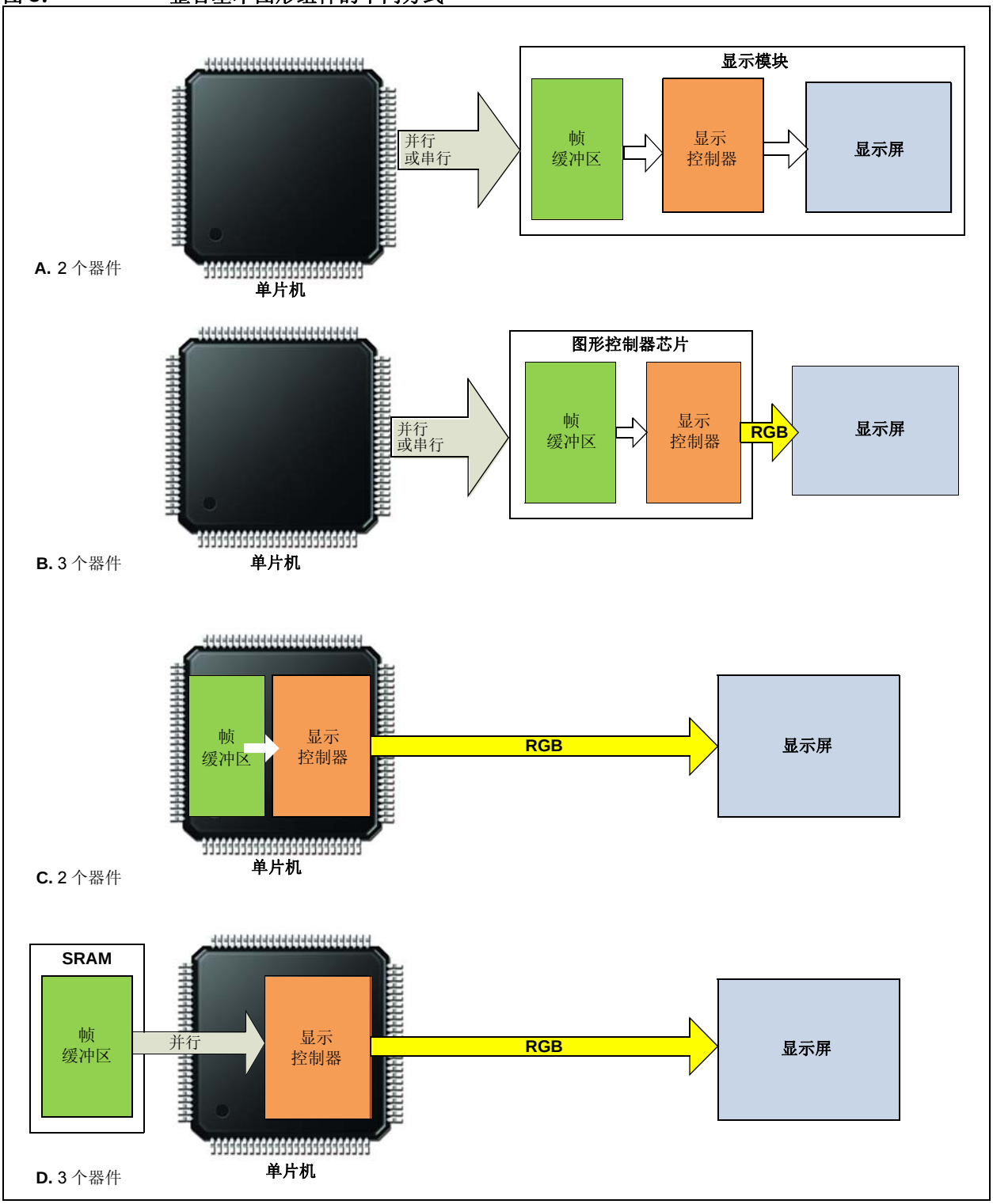


表 2： 基本组件

选项	优点	缺点
A. 帧缓冲区和显示控制器集成在一个模块中，称为显示模块。单片机通过串行或并行接口与显示模块接口。 (1)	• 不需要特定 IC 来实现图形功能 • 较少的系统元件和较小的 PCB 空间	• 通常成本较高 • 通常，显示模块更改时，会迫使软件驱动程序也发生更改。 • 可能缺少双缓冲、动画等所需的额外存储区
B. 帧缓冲区和显示控制器集成在一起。单片机和图形控制器通过串行或并行接口进行通信，而图形控制器通过 RGB 接口与显示屏接口。 (1)	• 显示屏更改时，软件驱动程序不需要更改。可能只需更改编译时配置。 • 成本可能比选项 A 低	• 需要较多的系统元件并占用较大的 PCB 空间 • 显示尺寸受限于显示控制器内的帧缓冲区
C. 帧缓冲区和显示控制器集成在单片机中。单片机通过 RGB 接口与显示屏接口。 可以用一个并行接口与 DMA 引擎的组合来取代显示控制器。	• 只需要一个 IC 来实现图形功能 • 外形小巧 • 通常是最便宜的选择 • 由于存储器在单片机中，成像速度更快 • 显示模块更改时，不需要更改软件驱动程序。可能只需更改编译时配置。	• 显示尺寸受限于单片机内的帧缓冲区
D. 显示控制器集成在单片机中。将单独的 RAM 用作帧缓冲区。单片机通过 RGB 接口与显示屏 （显示面板）接口，通过并行接口与帧缓冲区接口。 可以用一个并行接口与 DMA 引擎的组合来取代显示控制器。	• 由于帧缓冲区大小可由用户选择，因此单片机可支持最大显示尺寸 • 成本通常比选项 A 和 B 低	• 需要额外的 IC 芯片用于帧缓冲区

注 1： 串行连接可用于具有低颜色深度的低分辨率显示屏 （例如， 1 bpp， 120x64）。分辨率很高时，速度可能成为瓶颈。

对于选项 A 和 B， Microchip 图形库当前支持带 PMP 或 EPMP 的 PIC24、 PIC32 单片机及 dsPIC® DSC。 PIC24FJXXXDAXXX 系列器件可支持选项 C 和 D。 PIC24FJ256DA210 包含 96 KB 内部存储器并且内置一个图形控制器。它还支持通过 EPMP 模块将可选外部 RAM 用作带并行接口的帧缓冲区。有关该器件的更多信息，请参见对应器件的数据手册以及《PIC24F 系列参考手册》中的**第 43 章 “ 图形控制器模块（GFX）”**（DS39731）。

© 2012 Microchip Technology Inc.

显示面板的上电时序

不同显示面板将有不同的电源要求和时序以支持显示面板的各种电源信号。在一些显示面板中，可找到以下电源信号：

- 数字电源信号
- 模拟电源信号
- LCD 电源信号
- 背光电源信号

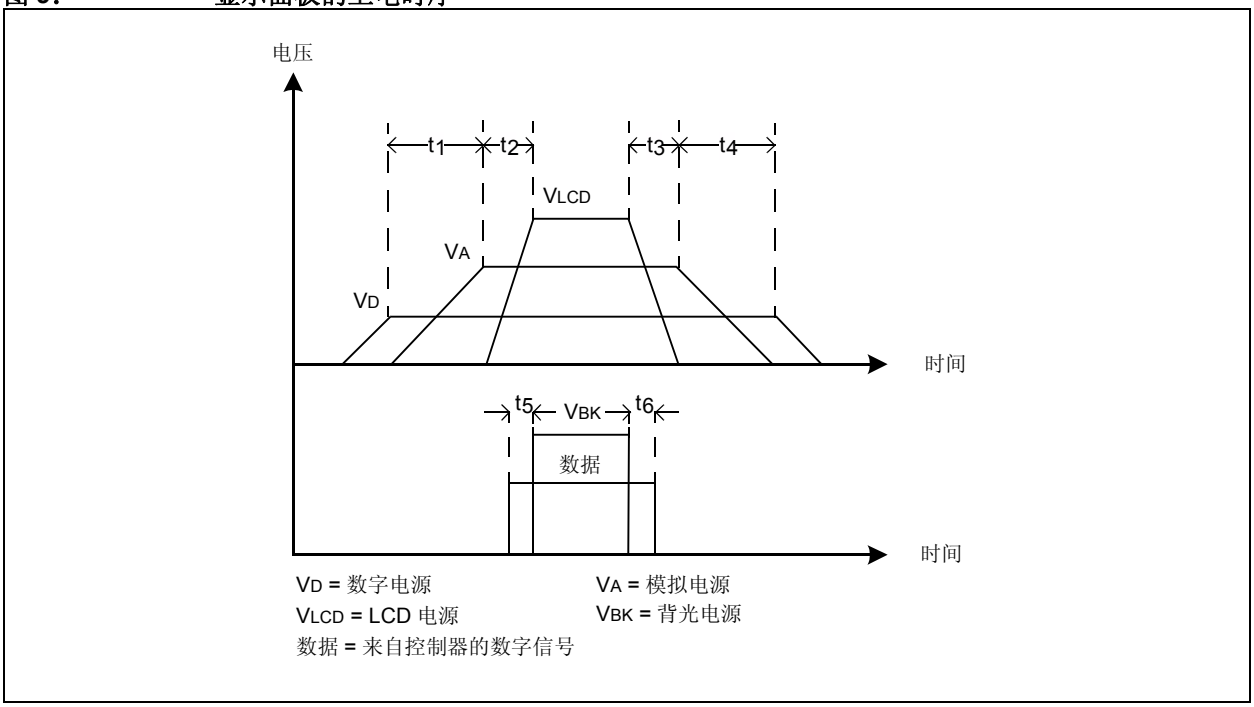
数字电源信号用于为面板上的数字逻辑供电。模拟电源信号用于为面板的模拟部分供电。LCD 电源信号也称为栅极电压，与数字数据信号一起主要用于控制像素亮度。在某些情况下，LCD 电源信号由两个电源信号组成：LCD 正电源信号和 LCD 负电源信号。在某些情况下，只有一个信号可用。背光亮度由背光电源信号控制。

根据显示面板的设计，可在显示板数据手册中找到所有四种类型的电源信号。在某些情况下，只显示数字电源和背光信号。这意味着显示面板已集成一个内部电路来生成模拟和 LCD 电源信号。这通常适用于小型显示屏（2 英寸到 4 英寸）。对于较大的显示屏，模拟和 LCD 电源信号的电压要求往往比较高。在这种情况下，显示面板集成此类电路不太实际；因此，这两个电源信号（指定为输入）必须从外部提供。

显示面板的不同电源信号必须遵循制造商建议的上电顺序。如果没有遵循正确的顺序，显示面板的使用寿命可能会明显缩短。图 9 给出了显示面板的典型上电时序。

时序要求由 t1、t2、t3、t4、t5 和 t6 表示。大部分情况下，这些要求表示为 t1 = t4、 t2 = t3 以及 t5 = t6。

图 9：　　显示面板的上电时序



触摸屏

某些应用需要支持触摸屏显示。这可通过在显示屏上使用单独的触摸屏或选择具有触摸屏的显示模块实现。在这两种情况下，必须通过单片机或单独的触摸屏控制器（例如，Microchip 的 AR1000 系列触摸屏控制器）处理触摸信号。这些触摸信号为模拟和数字信号，必须进行解码以感应触摸坐标。透明的触摸屏通常是电阻式或电容式。电阻式触摸屏最常用，并且通常为 4 线或 5 线配置。可通过测量触摸屏的电阻变化检测触摸点。在此，只对 4 线触摸屏进行说明。

4 线电阻式触摸屏

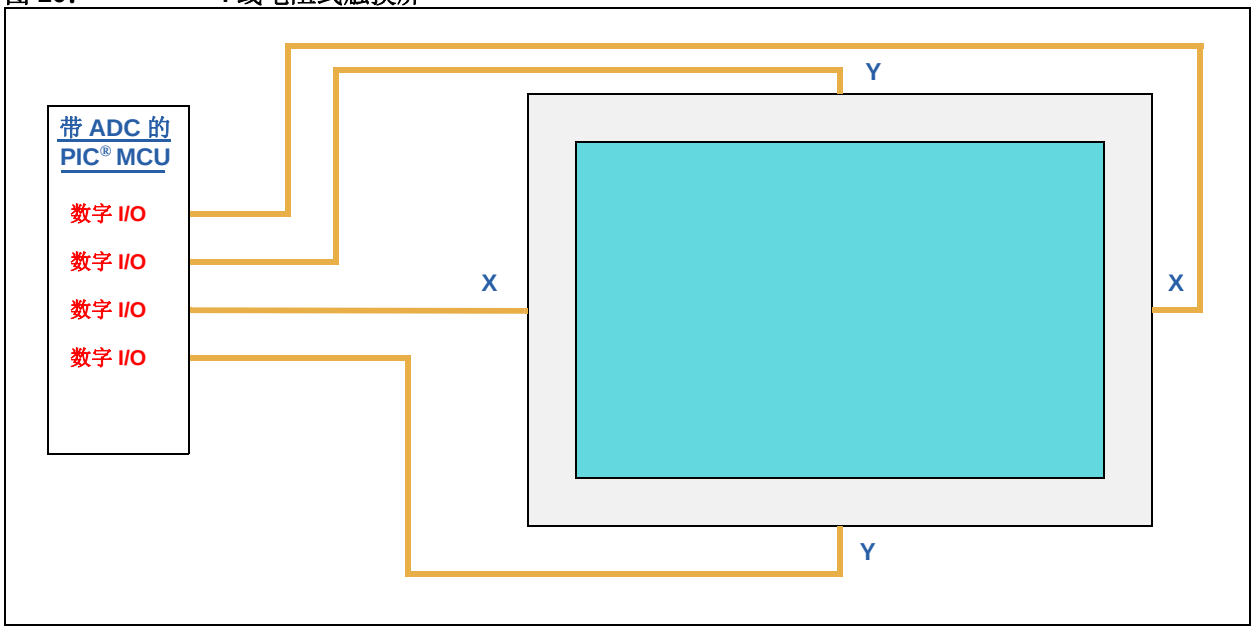
这种触摸屏有 4 个信号，其中两个为纯粹的数字信号。其他两个信号可交替配置为数字和模拟信号。

这 4 个信号可直接连接到单片机 I/O 引脚（两个为数字输入，两个为数字输出或模拟引脚）。图 10 给出了该模式的连接。

当用户触摸屏幕时，屏幕的电阻发生变化。通过测量水平和垂直方向的电阻，并将这两个值与校准值比较，即可获得触摸点的 (x,y) 坐标。

当触摸屏幕的某个点时，通过对 y 信号施加电压并测量 x 信号上的模拟电压获得 x 坐标电压，如图 11 所示。通过对 x 信号施加电压并测量 y 信号上的模拟电压获得 y 坐标电压，如图 12 所示。

图 10：　　4 线电阻式触摸屏



© 2012 Microchip Technology Inc.

图 11：　　测量 X 轴电压

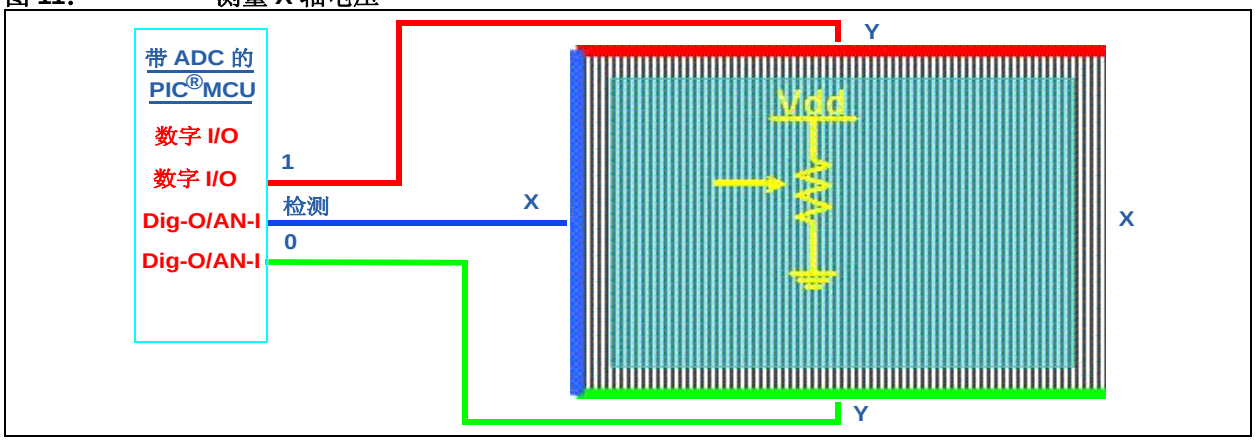
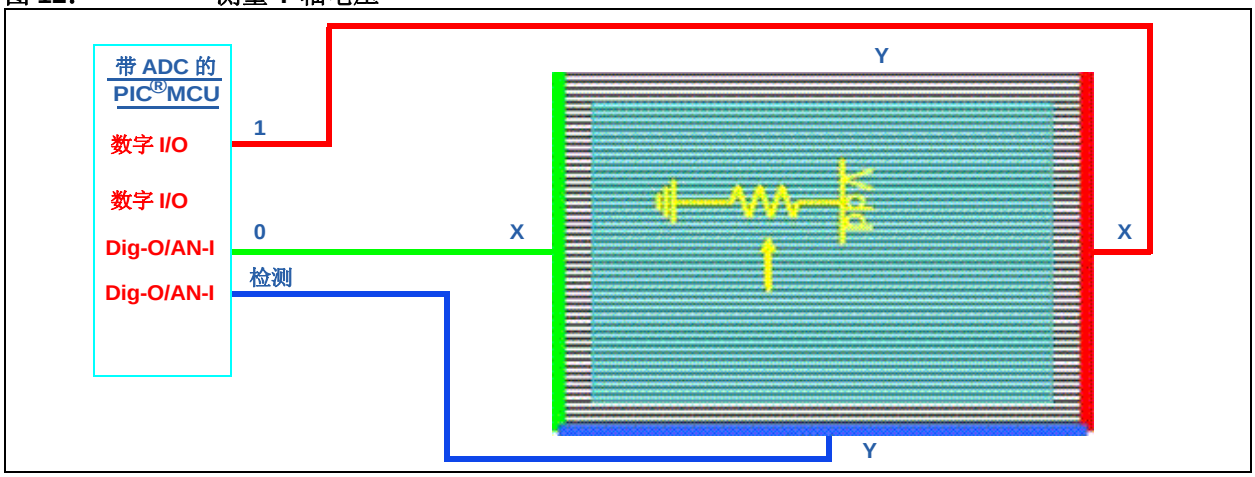


图 12：　　测量 Y 轴电压



决定因素

了解了图形子系统的基本定义和组件后，下一步是要决定应用规范。决定规范时需要考虑的一些重要因素如下：

- 显示屏分辨率和尺寸
- 显示屏方向 —— 横向模式或纵向模式
- 颜色深度 （bpp）
- 帧缓冲区大小
- 单片机处理能力
- 图形组件的配置
- 帧速率与 MIPS 的关系
- 与数量不匹配的显示 RGB 线接口

将在以下部分对这些决定因素进行描述。

显示屏分辨率和尺寸

一个特定的分辨率可通过不同显示尺寸获得。例如，QVGA （320x240）显示可由 1.5 英寸到 5.7 英寸尺寸范围内的显示屏提供。随着尺寸的增加，要保持分辨率不变，像素将看起来很粗糙，即屏幕原本显示弧线的地方会出现块状。

在某个应用中，当用户需要近距离查看显示屏 （例如，手持设备）时，较大的显示屏采用较高的分辨率是个比较好的选择。当用户远距离查看显示屏 （例如，银行的号牌显示）时，可使用具有较低分辨率的较大尺寸显示屏。如果要显示图片，最好是使用较高的分辨率。图 13 给出了 “A” 如何分别在较小尺寸的低分辨率显示屏、较大尺寸的较低分辨率显示屏和较大尺寸的较高分辨率显示屏上显示。

图 13：　　不同分辨率下 “A” 的显示



显示屏方向

显示屏可以是横向模式（例如，320x240）或纵向模式（例如，240x320）。也可通过在图形库中设置 90° 旋转函数或由显示控制器将横向模式显示屏用于纵向模式。同样，纵向模式显示屏也可用于横向模式。如果通过图形控制器中的特殊硬件实现像素旋转，那么性能上不会有损失。但是，如果由软件执行旋转（例如，使用图形库），那么软件性能方面有所损失。这是因为，对于每个 (x, y) 点，必须计算旋转后的新点 (x', y')，这将耗费一定的处理能力。

显示屏用于旋转方式时，请注意 RGB 色带对齐的差异，如图 14 所示。

颜色深度选择

除显示屏分辨率以外，正确选择颜色深度是另一个决定因素，因为这决定了帧缓冲区的大小（RAM 用量）。如果要显示自然照片，最好使用 16 bpp 或更高的颜色深度。如果 256 种不同颜色对于应用已足够，那么选择 8 bpp 的颜色深度即可（使用显示控制器提供的标准 256 色或使用 CLUT 的定制 256 色（见**附录 A：“颜色查找表（CLUT）”**））。与 16 bpp 相比，这将减少 50% 的 RAM 需求。如果只要 16 种或 4 种不同的颜色就已足够，那么可使用 4 bpp 或 2 bpp，与 16 bpp 相比，分别节省了 75% 和 87.5% 的 RAM。表 3 列出了不同颜色深度的 RAM 要求。

图 14：　　用于横向模式的横向和纵向显示屏

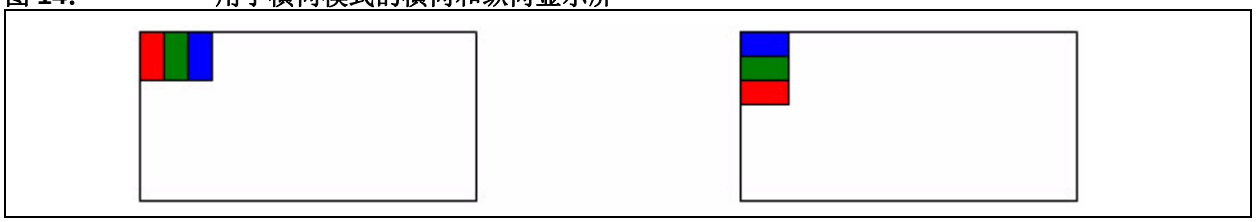


表 3：　　不同颜色深度的 RAM 大小要求

QVGA （320x240）的 bpp	16 bpp	8 bpp	4 bpp	2 bpp
颜色数	65,536	256	16	4
RAM 大小 （字节）	153,600	76,800	38,400	19,200

© 2012 Microchip Technology Inc.

帧缓冲区大小

按如下方式计算帧缓冲区的大小：

公式 2：

帧缓冲区（字节）= 总像素数 x 颜色深度（以 bpp 为单位）/8

表 3 给出了 QVGA（320x240）显示屏的示例。使用双缓冲技术时，帧缓冲区要求将加倍 （见**附录 B：“ 双缓冲 ”** 获取更多信息）。

如果帧缓冲区内置于显示控制器或智能显示模块中，并且 RAM 大小固定，那么可支持的最大分辨率有限。

处理能力 （MIPS）

所需的处理能力特定于应用。它取决于显示在屏幕上的图形元素的数量以及图形元素的复杂性。绘制诸如圆、斜角、文本之类的复杂图形比绘制线和矩形需要更强的处理能力。处理能力要求还取决于是否提供并使用了硬件图形加速器。处理能力要求还取决于屏幕元素的更新速率。对于许多嵌入式图形应用，≥ 16 MIPS 的处理能力可能已足够。检查处理能力要求的最佳方式是使用标准图形开发工具进行评估。（关于开发工具的更多信息，请参见 **“ 开发工具 ”** 章节。）

图形组件的配置

在表 2 中，每个配置都有各自的优点和缺点。

大多数情况下，决定使用一种或另一种配置不受技术优势或劣势的影响，而受组件的供应链优势或供货情况的影响。设计人员必须在优化设计满足要求和管理供应链之间进行权衡。

帧速率与 MIPS 的关系

帧速率是指一秒内可显示的不同帧数。这是个良好的视频显示性能指标，但不适用于嵌入式 GUI 应用。通常，嵌入式应用并不总是改变整个屏幕，而只是改变屏幕的一部分，如一个按钮或复选框。更改量取决于发生更改的控件的大小。更新时间也取决于多个因素，如是更改控件还是图像。规划应用时很有必要考虑最坏的情况。要了解系统的初步要求，有必要对帧速率和 MIPS 进行初步计算。除了这些计算，还建议使用开发工具（如评估工具包）来评估系统。

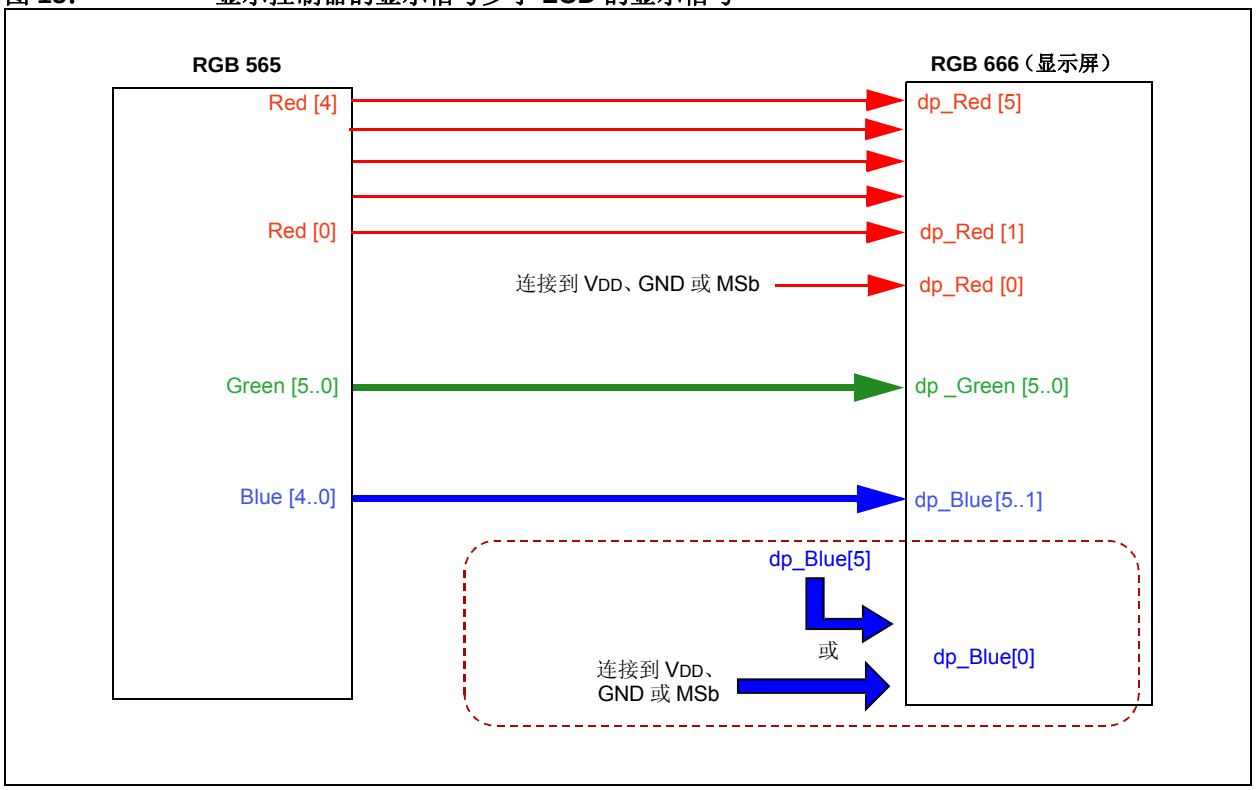
© 2012 Microchip Technology Inc.

与数量不匹配的显示 RGB 线接口

有可能显示控制器的 RGB 输出线数与显示屏的 RGB 输入线数不同，但仍可以将它们接口。如果显示屏的 RGB 输入线数等于显示控制器的 RGB 输出线数，那么将不会有任何色降。如果不相等，可能会有轻微的色降，因为显示面板将无法呈现显示控制器产生的所有颜色。通常，前者比后者更常见。

在图 15 中，显示面板比显示控制器具有更多的 RGB 信号线。此处，显示控制器的每个颜色的所有 RGB 线均连接到显示屏的显示信号线的 MSb。未连接的 LSb 可接地或 VDD，或者相同颜色的 MSb。将 LSb 连接到 MSb 是广泛使用的方法，因为这使显示屏有一个更大范围的颜色值。

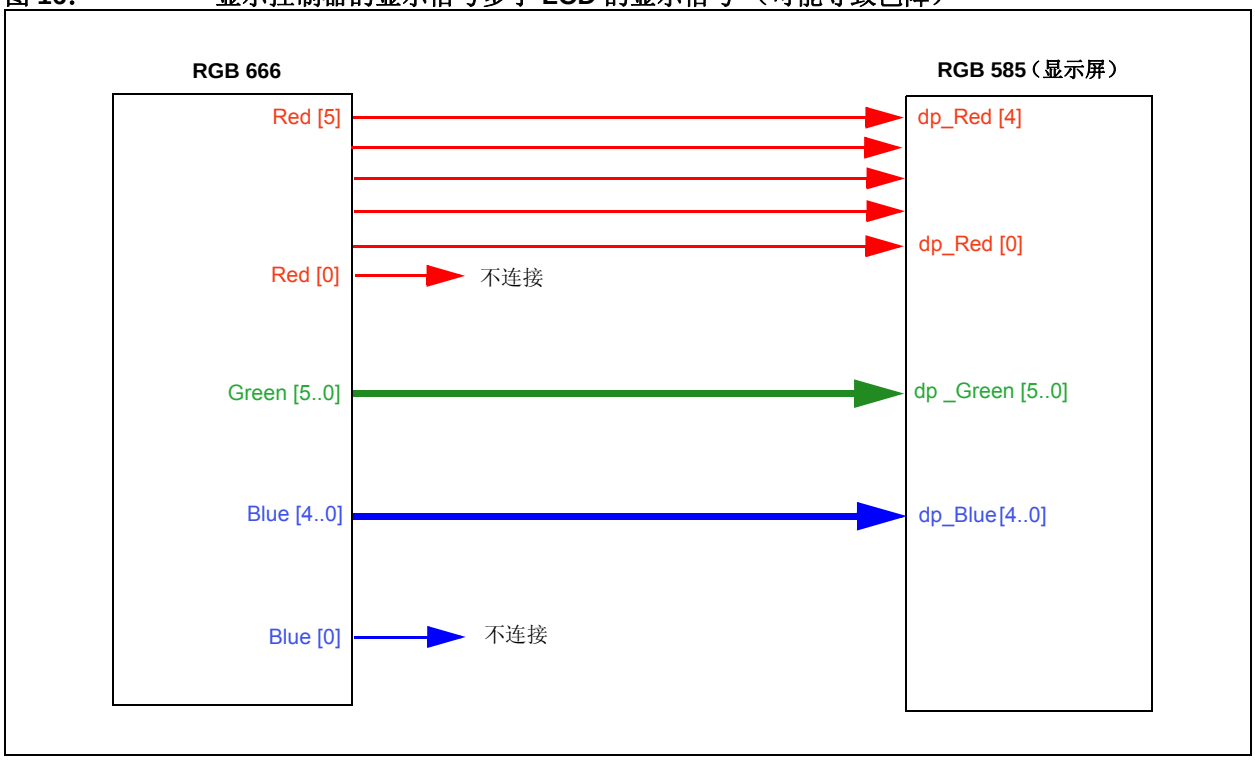
图 15：　　显示控制器的显示信号少于 LCD 的显示信号



© 2012 Microchip Technology Inc.

在图 16 中，LCD 显示屏的显示信号线少于显示控制器。显示控制器的每个颜色的显示线的 MSb 连接到 LCD 的所有显示信号线。未连接的 LSb 可保留不连。

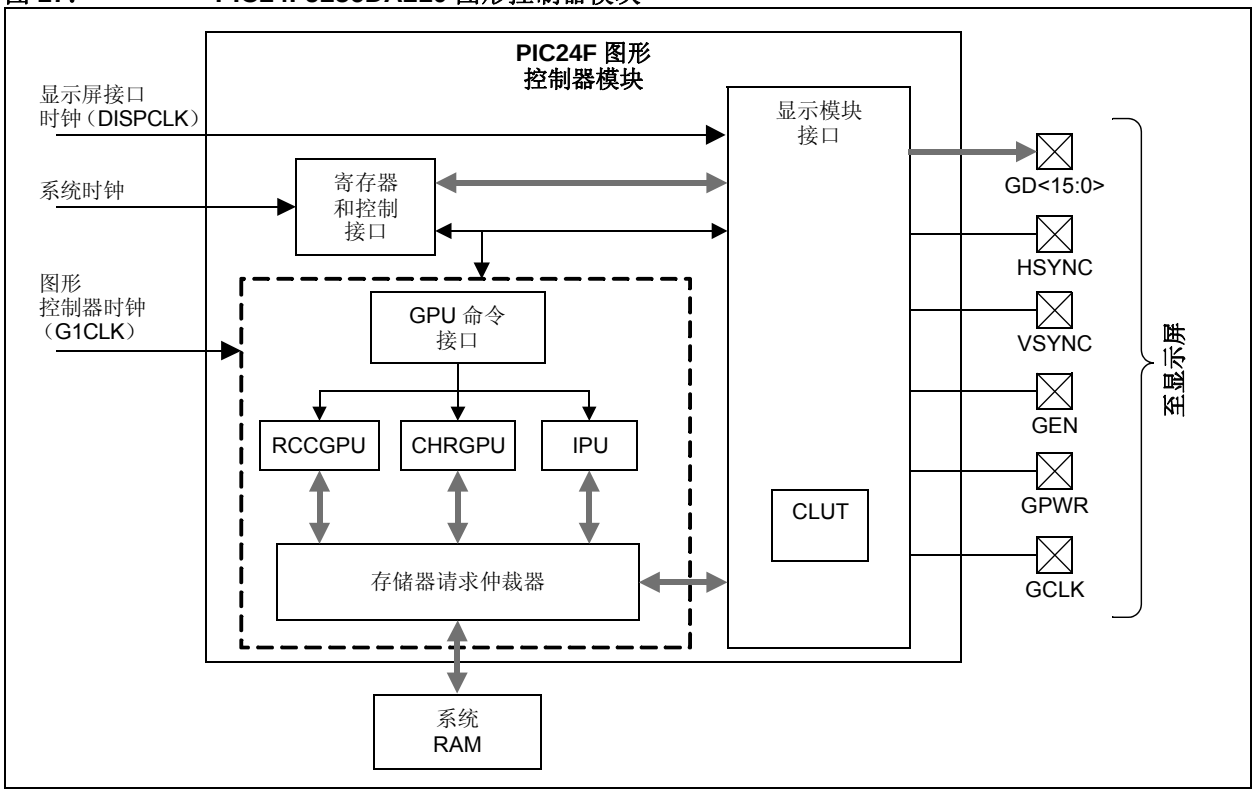
图 16：　显示控制器的显示信号多于 LCD 的显示信号 （可能导致色降）



PIC24FJ256DA210 单片机

PIC24FJ256DA210 器件为 16 位单片机，支持高达 16 MIPS 的处理速率。该单片机包含 96 KB 的内部 RAM 和一个带图形处理单元（Graphics Processing Unit，GPU）的内置显示控制器以加速绘制普通的 2D 形状。它还可通过增强型 PMP 模块与可选的外部并行 RAM 接口，以增大帧缓冲区的大小。PIC24FJ256DA210 图形控制器模块如图 17 所示。

图 17：　　PIC24FJ256DA210 图形控制器模块



© 2012 Microchip Technology Inc.

- DISPCLK 为驱动显示屏的时钟。
- 系统时钟为程序访问命令 / 控制 / 状态寄存器的时钟速度。
- G1CLK 是驱动 GPU 绘制线、矩形、渲染字符和解码压缩数据而无需处理器干预的时钟。
- 外部 RAM 最高可达 16 MB，可通过 EPMP 模块使用并行接口连接。图形模块可独立使用外部 RAM，而无需处理器参与。允许的接口局限于 8 位或 16 位并行连接。有关更多选择和信息，请参见《PIC24F 系列参考手册》中的**第 42 章 “ 增强型并行主端口 （EPMP） ”**（DS39730）。
- HSYNC 和 VSYNC 是到显示屏的垂直和水平同步信号。
- GCLK 是像素时钟。
- GEN 的功能会针对是 TFT 还是 STN 显示屏而有所不同。对于 TFT，该信号指示数据线有效。对于 STN，该信号在线翻转模式下每条线翻转一次，在帧翻转模式下每帧翻转一次。有关更多信息，请参见《PIC24F 系列参考手册》中的**第 43 章 “ 图形控制器模块 （GFX） ”**（DS39731）。
- GD<15:0> 根据图形模块设置存储显示屏 RGB 或灰度值。根据显示屏的接口要求，仅使能所需的线数。（例如，对于 TFT LCD 的 RGB565 输入，使能 16 根线，或对于 MSTN 的灰度输入，使能 4 根线）。
- GPWR 是显示屏的电源控制信号。对于一些大显示屏，可能需要外部电路。使用该信号使能或禁止外部电源电路。在包含内部电源电路的显示屏中，该信号可连接到该显示屏的电源使能引脚。该信号不应用作显示屏的电源线。

表 4 列出了不同显示屏和 RAM 配置所需的单片机引脚数。

图形处理单元（GPU），如字符图形处理单元（Character Graphical Processing Unit， CHRGPU）、矩形复制图形处理单元（Rectangle Copy Graphics Processing Unit， RCCGPU）和解压缩处理单元（Inflate Processing Unit，IPU）均为图形加速器。这些加速器分别用于渲染字符、矩形以及解压缩数据。

这些 GPU 有助于释放单片机的处理能力，从而用于应用目的。应用只需发出命令将原始的渲染函数 （如线、条形和字符）绘制到屏幕，而不是由 CPU 渲染像素。发出命令后，CPU 可释放出来执行其他应用任务。应用程序代码与 RCCGPU 并行运行，同时绘制线条。但是，应小心，因为函数 （例如， `Line()`）返回时不需要指示该线条已完成绘制。这就是所谓的非阻塞绘制。也可通过在 `GraphicsConfig.h` 文件中设置适当的编译器开关来阻塞绘制，如以后章节所述。

表 4： 单片机引脚

配置	显示数据引脚（RGB）	EPMP 引脚	其他（时钟和同步）引脚	总数
无外部 RAM 的 TFT LCD （使用 CLUT 和 16 位颜色）	16	0	5	21
无外部 RAM 的 256 色 CSTN	8	0	5	13
具有 256 KB 的外部 16 位宽 RAM 的 TFT LCD （使用 16 位颜色）	16	37	5	58
无外部 RAM 的 16 色 MSTN	4	0	5	9

GPU 简要说明如下：

- **CHRGPU：**在显示屏上显示字符。字体表必须装入 RAM 并且 CHRGPU 必须指向该字体表。还必须在相应的 CHRGPU 寄存器中初始化 (x, y) 坐标。给定一个字符代码和绘制命令时，字符将被填充至所配置的 RAM 区域，该区域也可以是帧缓冲区。由于 CHRGPU 不能动态旋转字符，用户必须注意显示屏方向。 CHRGPU 不支持平滑字体，字符上的所有像素均具有相同的颜色。使用 CHRGPU 命令设置前景色和背景色。使能透明状效果时，仅绘制前景色，而禁止透明状效果时，还会在字符周围绘制背景色。要默认使用 CHRGPU，请取消注释 `MicrochipGraphicsModule.h` 中的该行：`#define USE_DRV_OUTCHAR` 。
- **RCCGPU：**用于绘制水平或垂直线、矩形、填充矩形以及复制矩形区域。 RCCGPU 可执行以下三种操作：
  - 复制 —— 将存储器块从 RAM 的一个区域复制到另一区域。根据命令参数的不同，存储器块可以是连续块或矩形块。
  - 带纯色填充的复制 —— 用特定颜色填充矩形区域。
  - 带透明状效果的复制 —— 与复制选项相同，但是设置为透明状效果的颜色，不会复制到目标，保持目标的该部分不变。

每种操作均可使用 16 种称为光栅运算（Raster Operation， ROP）的可用逻辑运算中的一种，这在复制时运用。

例如，可按原样复制源区域或源区域可与目标区域进行或运算，或者源区域可与单独的区域进行与运算并复制到目标区域。有关图形控制器模块（GFX）和支持的 ROP 的更多信息，请参阅《PIC24F 系列参考手册》中的**第 43 章“ 图形控制器模块（GFX）”**（DS39731）。

RCCGPU 可用于实现特殊效果，如滚动、剥离等屏幕动画。关于 RCCGPU 高级用法的更多信息，请参见**附录 C：“RCCGPU 的高级用法 ”**。

- **IPU：**用于对通过采用固定霍夫曼代码的 DEFLATE 算法压缩的数据进行解压缩。例如，可将图像压缩并保存在内部闪存或外部存储器中，并可在运行时解压缩到 RAM。同样，在运行时也可使用 IPU 解压缩用户特定的压缩数据并进行使用。注意，解压缩只能从压缩块的开头开始，而不能从中间开始。例如，存储多个图像时，将每个图像压缩到其自己的压缩块。通过指定所需压缩块的位置，可使用 IPU 解压缩任何图像。Microchip 图形库将处理该情况，使此过程对客户透明。

注：	转换图像时，可通过选择图形资源转换器（Graphics Resource Converter，GRC）中的“IPU”选项压缩位图。GRC 是一个包含在图形库安装中的工具。`PutImage()` API 在运行时使用 IPU 自动解压缩这些压缩图像。用户需要分配所需的RAM量进行IPU操作，方法是使用 Microchip 图形库帮助文件中所述的编译时选项。

关于这些GPU及其寄存器的更多信息，请参见《PIC24F 系列参考手册》中的**第 43 章“ 图形控制器模块（GFX）”**（DS39731）。

© 2012 Microchip Technology Inc.

开发工具

一种评估应用系统规范的快速有效的方法是使用现有的开发工具。Microchip 有几款支持图形设计的开发工具。可用于图形开发的 2 个重要工具为：

- LCD 图形控制器 PICtail™ Plus SSD1926 电路板（AC164127-5），该电路板是 Microchip 的 16 位和 32 位单片机的通用开发电路板（如 Explorer 16 电路板和 PIC32 入门工具包）的附加板。
- PIC24FJ256DA210 开发板（DM240312），是一个独立的电路板。

这两个电路板都需要附加显示模块，可提供带各种尺寸显示屏的显示模块。借助于显示屏实验布线板可使用用户定制的显示面板。图 18、图 19、图 20 和图 21 给出了这些开发工具。关于最新的工具套件，请访问 http://www.microchip.com/graphics。

图 18：　　带 3.2 英寸显示屏的图形 PICtail™ PLUS 子板工具包（AC164127-3）

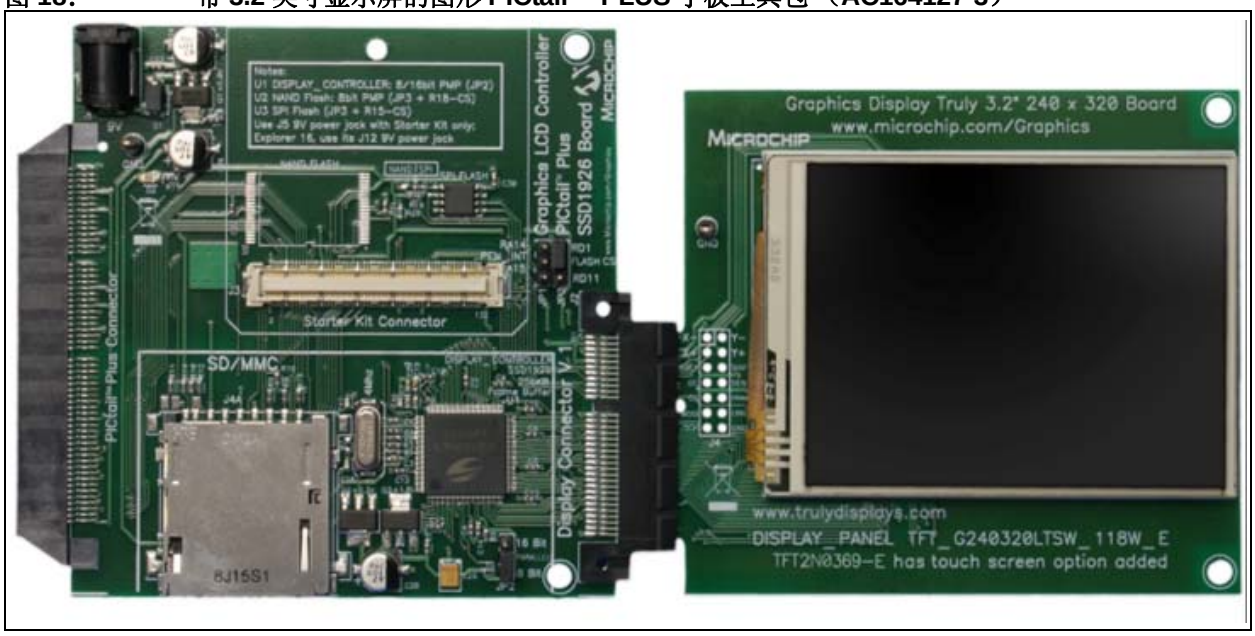


图 19：　　随 PIC24FJ256DA210 开发工具包提供的开发板（DV164039）

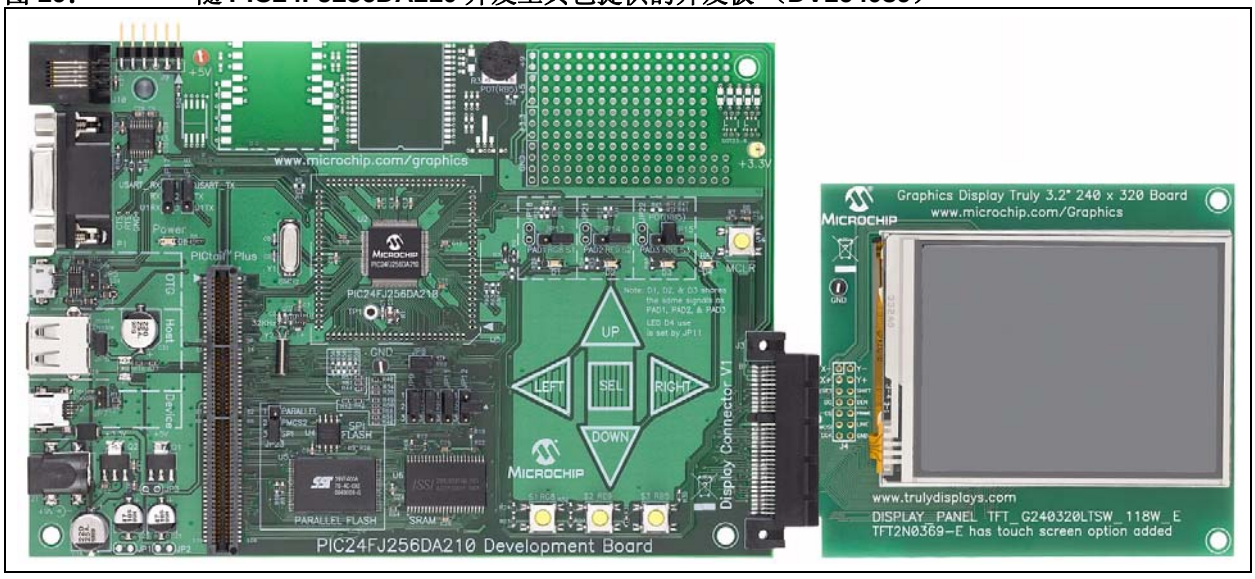


图 20：　　4.3 英寸 WQVGA POWERTIP TFT 显示屏板 （AC164127-6）

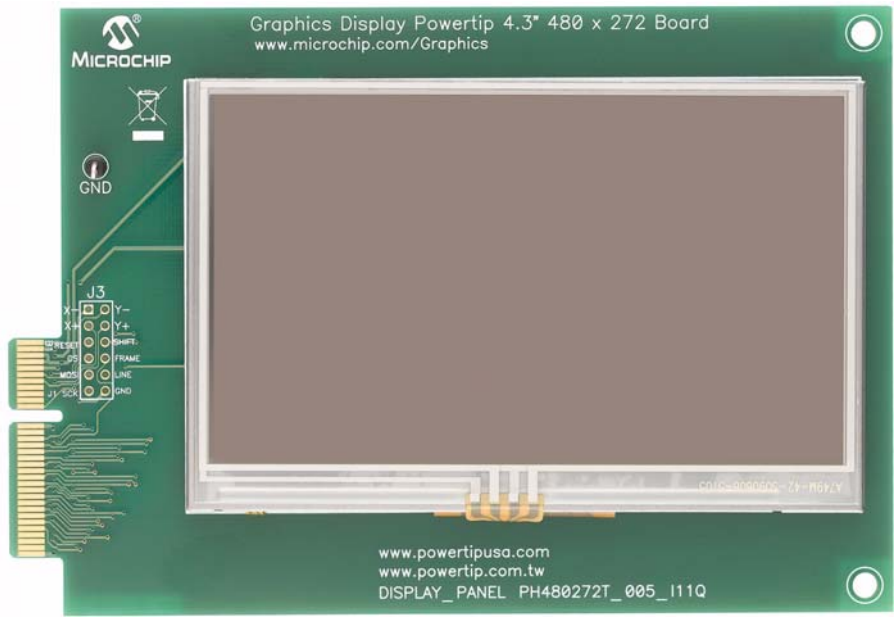


图 21：　　图形显示实验布线板 （AC164139）



© 2012 Microchip Technology Inc.

软件

任何图形应用所需的基本软体组件是一个软件显示驱动程序，它提供一个基本的操作（即，设置像素的颜色）。驱动程序还可实现 API 以绘制基本形状，例如，线、矩形、条形、圆形、文本和图像等。对于使用的每一种图形驱动器，必须单独编写对应的软件显示驱动程序。更复杂的图形元素，如标签、按钮、复选框、滑动条和进度条，通过调用更高层，然后使用对应软件显示驱动程序实现。

Microchip 提供了一个免费用于 PIC® MCU 的软件库，即 “Microchip 图形库 ”，它包含上文讨论的驱动程序和更高层协议。该图形库配有几个演示程序，用户可利用对应的开发工具直接运行这些程序。

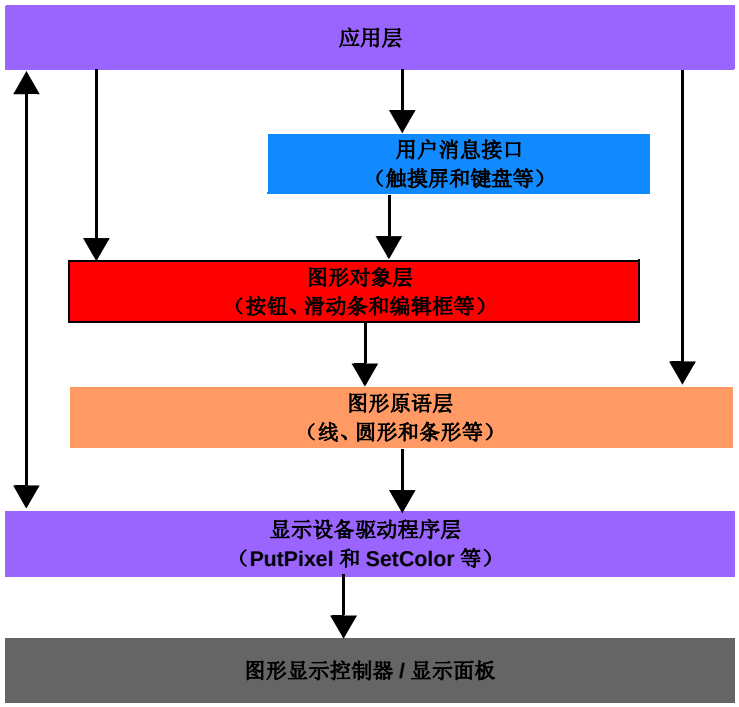
Microchip 图形库具有以下特性：

- 用于 16 位和 32 位 PIC® MCU 以及 dsPIC® DSC
- 模块化设计，仅编译需要的内容
- 支持多个用户接口
- 不依赖于显示屏尺寸或分辨率
- 成本低且功能齐全的开发工具
- 可导入字体和图像的实用程序
- 免费提供给 Microchip 的客户
- 包含多个底层驱动程序

图 22 显示了 Microchip 图形库的结构。

Microchip 图形库 v2.11 位于 Microchip 应用程序库中，可从 www.microchip.com/MAL 下载获得。

图 22： MICROCHIP 图形库的结构



注： 此处说明的是Microchip图形库V2.11。将来，可能会修改软件结构。请参见Microchip图形库的帮助文件获取最新信息。

图形显示控制器是由帧缓冲区和显示控制器组成的硬件模块。剩下的层为软件层。 Microchip 图形库以文件夹 “Microchip Solutions” 下的一组文件夹组成。C 文件位于 *Microchip Solutions/Microchip/Graphics* 文件夹中，而头文件在 *Microchip Solutions/Microchip/Include/Graphics* 文件夹中。特定于开发板的文件在 *Microchip Solutions/Board Support Package* 文件夹中。项目路径必须设置为指向这些文件夹。

从最底的软件层开始到最上面的层，每层的功能和负责这些层的文件将在以下部分进行深入阐述。有关更多信息，请参见 Microchip 图形库的帮助文件。

显示设备驱动程序层

每个硬件显示控制器都有自己的命令集和状态信息。因此，每个支持的显示控制器都需要单独的软件驱动程序，这符合图形库定义的显示驱动程序和标准 API 的要求。支持的驱动程序列表可在图形库帮助文件中找到。要知道是否支持某个显示模块，只需查看上述文件夹中是否提供了该显示模块中的显示驱动程序。

该层提供的主函数使用API `ResetDevice()`初始化驱动程序，使用 API `SetColor(color)` 和 `PitPixel(x,y)` 绘制像素并通过 API `GetPixel(x, y)` 获知像素的颜色。提供的其他功能为：设立剪切区域，得出显示屏的最大 x 和 y 值等。可仅使用该层（而不使用任何更高层）编写应用程序。在该情况下，所有形状必须由用户绘制。要包含该层，必须将以下文件添加到项目中：

头文件

`Graphics.h`

`DisplayDriver.h` 或特定 `<Driver.h>`（如 `SSD1926.h`）

配置文件

`HardwareProfile.h`

`GraphicsConfig.h`

这些文件均与项目相关且必须在项目文件夹中。

源文件

特定 `<Driver.c>`（如 `SSD1926.c`）

图形原语层

这是显示驱动程序层上的层并通过 API 提供最常用服务，这些 API 用于绘制基本形状，例如，线（正常、粗线和虚线）、条形、矩形、圆形、多边形、斜角和圆弧。它还提供用于绘制文本和图像的 API。这些都是通用 API，可与任何给定显示驱动程序配合使用。但是，这些 API 中的一部分可通过驱动程序层实现以达到最佳性能，尤其在驱动程序支持 2D 加速（例如，Microchip 图形模块和 SSD1926）时。可以仅使用原语和驱动程序层而不使用更高层开发简单的应用程序。要包含该层，必须将以下文件添加到项目中：

头文件

`Primitive.h`

源文件

`Primitive.c`

图形对象层（GOL）

GOL 由许多称为控件的对象组成，例如按钮、文本框、复选框、滚动条、进度条、图片、列表框、组合框、仪表、数字仪表、刻度盘、图表和网格，这些构成了复杂图形应用的基本元素。这些控件中的每一个都用 C 实现，但基于面向对象的基本原则，且可用作模块。该层的使用必须在编译时在 `GraphicsConfig.h` 文件中使能。各个控件的使用均可在编译时在 `GraphicsConfig.h` 文件中使能或禁止，从而节省 RAM 和 ROM。有关更多信息，请参见 **“配置”** 一节。

每种控件都有缺省的样式方案，该方案定义用于控件的各个部分和状态的字体和颜色。例如，处于按下状态的按钮的颜色与处于释放时的按钮不同。每个控件的样式方案在 Microchip 图形库的帮助文件中进行了详细说明。对于每种样式方案，需要一些堆存储区（动态分配的存储区）来存储样式方案值。需要堆来存储每个使能的控件的状态信息。总的堆大小必须大于所用控件的所有实例的堆总和。除了堆要求，每种控件类型还需要使用 RAM 空间存储渲染和管理控件时使用的变量。该额外 RAM 要求是每类控件的固定开销。两者的差异在于：每个控件实例都有堆要求，而 RAM 要求是每类控件需要的。RAM 要求是不变的，不依赖于一种控件类型的实例数。

© 2012 Microchip Technology Inc.

例如，如果版本说明记录：

模块	按钮	GOL
用于 PIC24F MCU 的堆	28（每个实例）	20（每个样式方案）
用于 PIC32 MCU 的堆	44（每个实例）	24（每个样式方案）
用于 PIC24F MCU 的 RAM	8	32
用于 PIC32 MCU 的 RAM	12	28
用于 PIC24F MCU 的 ROM	1002	2076
用于 PIC32 MCU 的 ROM	2748	5400

注： PIC24F 和 PIC32 器件的 RAM 和 ROM 要求可能不同，因为两者的单片机架构和编译器不同。

对于仅使用两个按钮的 PIC24F 应用，RAM 为 8 字节，ROM 为 1002 字节，而所需的堆存储区将为 2 x 28 = 56 字节。

如果只使用一个样式方案，那么将需要 20 字节的堆存储区。

公式 3：

堆总量（最低堆要求）= 20
（针对每个样式方案）+ (2 x 28) = 76 字节

RAM 总量（针对图形）= 32（针对 GOL）+ 8 = 40 字节

注： 此例仅用于说明。建议查看 Microchip 图形库的版本说明以获得特定版本的对应值。

GOL 取决于原语和显示驱动程序层。必须在循环中连续调用 `GOLDraw()` 函数以简化控件绘制。另外，必须在应用代码中执行 `GOLDrawCallback()` 函数。该函数用于定制绘制，在帮助文件中有所说明。通常，该函数只返回：TRUE。

要包含 GOL，必须将以下文件与原语层和显示驱动程序层所需的文件，一起添加到项目中。请参见帮助文件，以获取最新文件列表。如果使用 GOL，那么必须在 `GraphicsConfig.h` 文件中定义 `#define USE_GOL` 宏。还必须定义所用控件的各个宏，例如 `#define USE_BUTTON`。如果未定义这些宏，将不会编译这些控件，即使它们已经包含在项目中。

用户还可创建自己的控件并添加到图形库中。请参见 **“参考资料”** 以获取更多详细信息。

文件类别	按钮
头文件	• GOL.h • Button.h • Chart.h • CheckBox.h • DigitalMeter.h • EditBox.h • Grid.h • GroupBox.h • ListBox.h • Meter.h • Picture.h • ProgressBar.h • RadioButton.h • RoundDial.h • Slider.h • StaticText.h • TextEntry.h • Window.h • <CustomWidget.h>
配置文件	• GraphicsConfig.h（用于选择 GOL 及其各个控件的使用）
源文件	• GOL.c • GOLFontDefault.c • Button.c • Chart.c • CheckBox.c • DigitalMeter.c • EditBox.c • Grid.c • GroupBox.c • ListBox.c • Meter.c • Picture.c • ProgressBar.c • RadioButton.c • RoundDial.c • Slider.c • StaticText.c • TextEntry.c • Window.c • <CustomWidget.c>

注： 该列表仅用于说明。请参见 Microchip 图形库帮助文件获取最新文件列表。

用户消息接口

用户消息接口是 GOL 的一个子层，当使用 GOL 时会使能该子层。该子层用于方便在控件和用户输入之间进行消息传递。例如，当用户按某个按钮时，消息被发送到回调函数，称为 `GOLMsgCallback()`，通过此函数检查指示按钮被按下的消息并执行操作。当使用 GOL 时，应用代码中必须有该回调函数，无论是否使用消息传递。如果未使用消息传递，函数主体必定返回 `1`。

与 `GOLDraw()` 类似，必须在应用代码的循环中连续调用 `GOLMsg()` 以方便消息收集和传递。

`GOLDraw()`、`GOLDrawCallback()`、`GOLMsg()` 和 `GOLMsgCallback()` 的使用将在例 4、例 5 和例 6 进行说明。

应用层

在该层中，用户可完全控制应用。最初，用户必须初始化 Microchip 图形库。如果要使用所有层，则通过调用 `GOLInit()` 完成初始化，如果不使用 GOL 但使用原语和显示驱动程序层，则调用 `InitGraph()`，或如果仅使用显示驱动程序层，则调用 `ResetDevice()`。初始化程序后，可调用原语和驱动程序层的 API 来实现所需的绘制功能。要使用 GOL 对象（如按钮），必须通过调用控件的创建函数（例如，`BtnCreate()`）逐个创建控件，直到创建完所有控件。这一步不会显示这些控件。当在 while 循环中重复调用 `GOLDraw()` 函数时，创建的控件才会绘制在屏幕上。通过在同一循环中调用 `GOLMsg()` 处理消息，如例 5 所示。

`GOLDraw()` 完成后，从触摸屏驱动程序和硬按钮驱动程序中接收消息。获得的消息传递到 `GOLMsg()` 以进行处理并输出特定于控件的消息。例如，它将 “USER TOUCHED POSITION 100, 100（用户触摸了位置 100, 100）” 消息转换为 `BUTTON1_PRESSED`。

此外，应用必须具有 `GOLDrawCallback()` 和 `GOLMsgCallback()` 函数。

如果定制绘制未完成，那么使用绘制回调，如例 4 所示。

例 4：

```
WORD GOLDrawCallback(void)
{
    return (1);
}
```

消息回调处理控件发出的处理过的消息，如例 6 所示。

如果应用已使用一个主循环，可在循环中调用 `GOLDraw()` 和 `GOLMsg()`（见例 5）。

注 1： 请参见应用笔记 *AN1136*，《如何使用 Microchip 图形库中的控件》来创建简单的应用。

2： 请参见 Microchip 图形库帮助文件获取相关应用笔记的列表。

© 2012 Microchip Technology Inc.

例 5：

```
while(1)
{
        if(GOLDraw())
        {                                   // 绘制 GOL 对象
                                            // 在此完成绘制，并处理消息
                TouchGetMsg(&msg);          // 从触摸屏获取消息
                GOLMsg(&msg);               // 处理消息
                SideButtonsMsg(&msg);       // 从侧按钮获取消息
                GOLMsg(&msg);               // 处理消息
        }
}
```

例 6：

```
WORD GOLMsgCallback(WORD objMsg, OBJ_HEADER *pObj, GOL_MSG *pMsg)
{
    // 按下按钮时发出蜂鸣声
    if(objMsg == BTN_MSG_PRESSED)
    {
        Beep();
    }
}
```

配置

通过以下两个文件实现对 Microchip 图形库的配置：

- GraphicsConfig.h
- HardwareProfile.h

GraphicsConfig.h

在 GraphicsConfig.h 中完成软件库相关的配置。可用选项为：

1. **USE_NONBLOCKING_CONFIG**：如果定义了该选项，那么将使用 API 调用的非阻塞模式，即，带返回状态值的 API 可返回值，而无需完成其任务。该任务可在返回后的某一时间完成。这在显示驱动程序支持 2D 加速时特别有用。在该情况下，应用可在并行处理渲染时重新获得控制权。这减少了单片机等待原语命令渲染（例如 line() 和 bar()）的空闲时间。如果禁止（注释掉）该定义，那么所有 API 调用只有在其任务完成后才返回。对于非阻塞配置，用户必须检查 API 的返回值以了解它的执行操作是否完成。
2. **USE_DOUBLE_BUFFERING**：如果定义了该选项，将使用两个缓冲区。一个缓冲区将用作绘制缓冲区，用于渲染下一个屏幕，而另一个为帧缓冲区，用于保存屏幕的可视像素。该模式用于避免人眼观察到对象缓慢绘制到屏幕上的情况，如渲染较大区域或解码和显示图像。该模式使用两倍的帧缓冲区量并且图形库当前只针对特定图形控制器支持该功能（请参见 Microchip 图形库文档）。如果该选项被注释掉，那么只有一个缓冲区用作帧缓冲区并且屏幕的更改将在同一时间可见。
3. **USE_PALETTE**：如果定义了该选项，将使能调色板模式并且从调色板表获取颜色。该选项仅对具有内置的可编程颜色查找表的控制器（例如，PIC24FJ256DA210 具有一个带颜色查找表的图形模块）可用。在屏幕上显示任何内容之前，需要该表来初始化调色板引擎和设置有效的调色板表。该图形库仅针对具有内置显示控制器的 PIC 单片机支持此功能。 COLOR_DEPTH 在使能调色板时可设置为 1、 2、 4 或 8。如果禁止该选项，将使用正常颜色模式。
4. **USE_FOCUS**：如果定义了该选项，将在选中控件上显示虚线轮廓（焦点线）。将按钮用作用户输入时，该焦点线对控件的导航和选择特别有用。
5. **USE_TOUCHSCREEN**：该选项通过使能 GOL 的触摸消息处理部分使应用支持触摸屏。
6. **USE_KEYBOARD**：该选项通过使能 GOL 的硬键消息处理部分使应用支持物理键。
7. **USE_GOL**：该选项使能图形对象层（GOL）。最初，必须调用 GOLInit()。必须在循环中反复调用 GOLDraw() 和 GOLMsg()，而 GOLDrawCallback()和GOLMsgCallback()函数必须通过应用执行。
8. **USE_BUTTON、USE_CHECKBOX、USE_WINDOW... USE_CUSTOM**：这些选项仅在定义了 USE_GOL 时有效。它们使能了对应用中每个控件的支持，并且每个使能的控件将使用其 RAM、 ROM 和堆份额。
9. **USE_MULTIBYTECHAR**：这将使 XCHAR 为 2 字节长，从而支持 Unicode（UTF16）。使用 Unicode 可支持除英语以外的其他字符。如果应用中需要支持多个语言，那么使能该选项；如果仅使用 ASCII，那么禁止（注释掉）该选项。如果该选项被禁止，那么 XCHAR 的每个字符只占用一个字节。
10. **USE_FONT_FLASH**：支持在应用中使用单片机内部闪存中存储的字体。
11. **USE_FONT_RAM**：支持在应用中使用单片机的 RAM中存储的字体。该选项在带内置显示控制器的 PIC 单片机中用作字体加速。
12. **USE_FONT_EXTERNAL**：支持在应用中使用外部存储器（串行或并行闪存）中存储的字体。应用要从外部存储器获取数据，需要执行函数 WORD ExternalMemoryCallback(EXTDATA *memory, LONG offset, WORD nCount, void *buffer)。
13. **COLOR_DEPTH**：指定演示板中使用的颜色深度，单位为每像素位数，值可为 1、2、4、8、16 或 24（受硬件功能限制）。

© 2012 Microchip Technology Inc.

14. **USE_BITMAP_FLASH**：支持在应用中使用单片机内部闪存中存储的位图图像。
15. **USE_BITMAP_EXTERNAL**：支持在应用中使用外部存储器（串行或并行闪存）中存储的位图图像。应用要从外部存储器获取数据，需要执行函数 `WORD ExternalMemoryCallback (EXTDATA *memory, LONG offset, WORD nCount, void *buffer)`。

注：	请参见Microchip图形库帮助文件获取最新配置选项集。

HardwareProfile.h

该文件用于配置应用的硬件，且用法与其他Microchip软件库中的类似。如果 HardwareProfile.h 已存在于应用程序中，可在该文件中添加以下定义，如果不存在，必须创建一个具有以下定义的新文件。该文件主要包含以下部分：

1. **GetSystemClock()**：该宏必须返回系统时钟的频率（单位为赫兹）。
2. **GetPeripheralClock()**：该宏必须返回外设时钟的频率（单位为赫兹）。在 PIC24 单片机中，这是系统时钟的一半。有关 PIC32 器件的更多信息，请参见相应的器件数据手册。
3. **GetInstructionClock()**：该宏必须返回指令时钟的频率（单位为赫兹）。在 PIC24 单片机中，这是系统时钟的一半，而在 PIC32 单片机中，这与系统时钟相同。
4. **显示相关设置**：这些设置包含一组 #define 语句，它们定义各个显示屏相关参数，如显示屏类型（TFT、CSTN 或 MSTN）、屏幕的分辨率、颜色深度、显示屏时钟速度、各种显示屏时序参数以及绘制缓冲区和帧缓冲区的地址等等。
5. **键的 I/O 端口**：提供了应用所用的 I/O 引脚的定义。这些引脚可用作开关的输入或点亮 LED 的输出，或者使能特定硬件的应用控制引脚。
6. **外部存储器定义**：这些 #define 语句提供了端口引脚的定义及 / 或模块的初始化，比如通过 SPI、I^2C™ 或 PMP 使单片机与外部存储器（EEPROM、闪存或 SD 卡）接口。
7. **触摸屏定义**：这些 #define 语句提供了用于检测支持触摸屏的应用中是否发生触摸的 ADC 通道的定义和设置信息。
8. **RTCC 定义**：这些 #define 语句提供了当在应用中使用内部实时时钟（Real-Time Clock，RTC）时 RTCC 模块的设置信息。
9. **通信定义**：这些 #define 语句提供了通信模块的定义和设置信息，如用于在应用中进行通信的 UART/USB 通道。
10. **应用相关定义**：任何其他应用相关硬件定义均可包含在此。

请参见 Microchip 图形库随附的任何图形演示程序了解更具体的相关信息或作为一个示例。

注：	请参见Microchip图形库帮助文件获取最新配置定义集。

PIC 单片机的其他图形库

除了 Microchip，还有多个第三方为 PIC MCU 提供图形库。例如：

- Segger 的 emWin 图形库
- Micrium 的 µC/GUI

结论

支持嵌入式图形在如今的用户界面应用中变得越来越重要。 Microchip 通过提供开发工具、图形库、支持图形的单片机（PIC24FJ256DA210），以及广泛的文档、演示代码和支持，为客户提供多方位支持。 Microchip 的图形解决方案可成功用于许多嵌入式图形应用，进而缩短上市时间并降低成本。

附录 A： 颜色查找表 （CLUT）

基本概念回顾

在了解 CLUT 的概念之前，应先了解图像表示及其数据类型。

数字图像由像素（也称为像元）组成。这是一个二进制图像（0 或 1）并通过简单的像素亮 / 暗表示。它还称为单色图像，对于 640x480 屏幕大小，图像大小为（640x480/8） 37.5 KB。

图像分辨率是指图像的像素数。纵横比为列 / 行的比。在上述 640x480 的例子中，纵横比为 4:3。已验证上述纵横比可显示为自然图像。

8 位灰度级图像

8位图像是其中每个像素都有通过一个字节表示的8位值（0-255）（也称为灰度）的图像。因此，该图像可以表示为一个二维数组 （值的范围在 0-255 之间），这种图像也称为位图。

例如：8 位灰度的 VGA 分辨率将是 300 KB。

彩色图像的图像数据类型

图形和图像文件格式的常见数据类型为24位彩色或8位彩色。在 24 位彩色图像数据类型中，每个像素由三个字节表示，一个用于每个主颜色的 RGB。其他颜色表示为 RGB 值的组合。由于每个值都在 0-255 的范围内，故而提供了总共 16,777,216 种可能的颜色，但是，这需要一个很大的存储器 （16 MB）。

例如，对于 640x480 的分辨率， 24 位彩色图像将需要 900 KB 的存储器 （未进行任何压缩）。

如果存储空间是个问题 （这种情况很常见），通过量化 24 位颜色信息，可实现合理且精确的 8 位颜色信息。这还意味着我们只有 256 种可能的颜色。

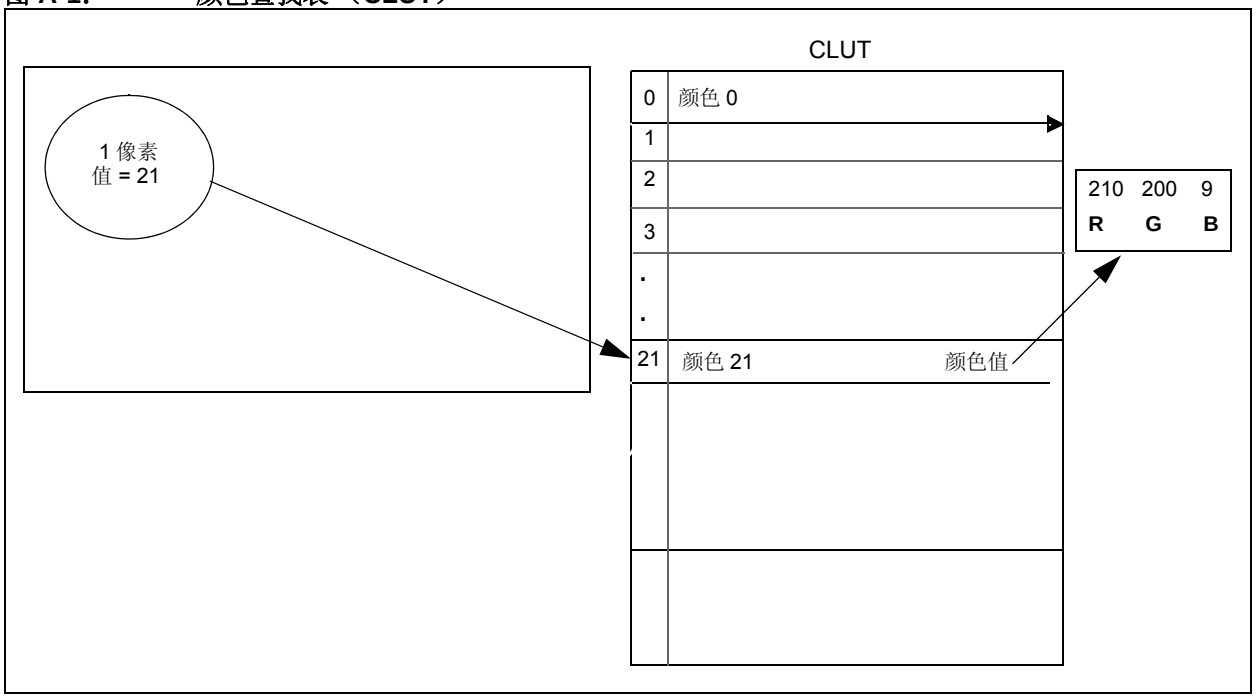
8 位彩色图像文件

图像文件使用一种特殊的概念将颜色信息存储在 CLUT 中。图像由一组字节而非颜色表示。这些字节构成表的索引，索引具有 3 字节值，用来为像素指定颜色。这表示，用户必须通过选择可最佳表示图像的颜色来表示图像且不超过 256 种颜色组合，这是因为它们通过 8 位值进行索引。

8 位表示相对 24 位表示的一个最重要优势是节省了存储空间，具体为 300 KB 对比 900 KB （未应用压缩）。

图 A-1 可帮助了解 CLUT，也称为调色板表。

图 A-1： 颜色查找表 （CLUT）



© 2012 Microchip Technology Inc.

图 A-1 给出了一个具有 RGB 值的像素，且它由值 21 表示。查找表中的值 21 指向该像素的 24 位 RGB 值。

表 A-1 使用调色板表的索引（如表 A-2 所示）而不是 RGB 值表示图像。

表 A-1： 使用索引而不是 RGB 值的示例图像的内容

0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
0	0	0	0	1	1	1	1
2	2	2	2	3	3	3	3
2	2	2	2	3	3	3	3
2	2	2	2	3	3	3	3
2	2	2	2	3	3	3	3

表 A-2： 调色板表 （24 BPP）

颜色索引	R	G	B
0	0	255	0
1	255	0	0
2	0	0	255
3	0	255	255
. .	. .	. .	. .
255	219	200	198

注： 一个颜色索引指代一行 CLUT。

一个简单的颜色更改类动画制作可通过上述方法更改 CLUT 完成。这还称为调色板动画。

在上述说明中，我们看到一种聚类图像和设置 CLUT 的简单方法，但聚类是个很慢的过程。

另一种设置 CLUT 的方法是缩放 RGB 范围以生成 16 位代码。但是，由于人眼对绿色比较敏感，我们可以给红色和蓝色分别分配 5 位，而为绿色分配 6 位，且图像中的每个元将被其索引取代。这可能导致图像产生边缘效应。因此，这不适合于自然图像，但适合于人工创建的图像。

PIC24FJ256DA210 支持最多 256 个 CLUT 条目，每个条目为 16 bpp。

平均分割算法

它是一种自适应方法，尝试将大部分位放置在颜色最聚类的位置，从而可将它们区别对待。这是将 RGB 图像转换为基于 CLUT 的颜色图像时最常用的算法，如 PC 应用 （例如， GIMP）。

附录 B： 双缓冲

屏幕上的像素处理需要直接写入帧缓冲区。当执行这些更改时，还将刷新屏幕。这表示，更改将在帧缓冲区更新后立即显示。这不适应于更改缓慢显示的案例，例如，解码图像或屏幕上的控件数较多。在该情况下，显示可能会很慢，并导致用户不满。

解决该问题的一个方案是使用Microchip图形库支持的双缓冲方案。在该方案中，更改不直接写入帧缓冲区，而是写入单独的缓冲区，称为绘制缓冲区。所有更改完成后，会交换绘制缓冲区和帧缓冲区的内容。现在，绘制缓冲区变成帧缓冲区，而且由于所有更改这时已绘制完成，更改将自然地显示给用户。当然，可能会有些延时，因为显示前所有更改必须先写入绘制缓冲区。相较于很慢显示更改，这样的延时通常更能被用户接受。

交换这两个缓冲区后，最新的缓冲区 （现在为帧缓冲区）的内容被复制到后台中的新绘制缓冲区，使新绘制缓冲区与要显示的内容同步。然后将新更改写入绘制缓冲区并周而复始地执行这一过程。

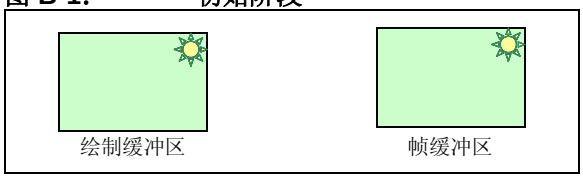
由于双缓冲方案使用两个缓冲区，因此 RAM 要求将加倍。

在 Microchip 图形库中，如果使能双缓冲，屏幕上所有控件更改完成（即在整个屏幕更新完成而非更新单个控件后显示新屏幕）后，帧缓冲区和绘制缓冲区将交换。

双缓冲的工作流程如下：

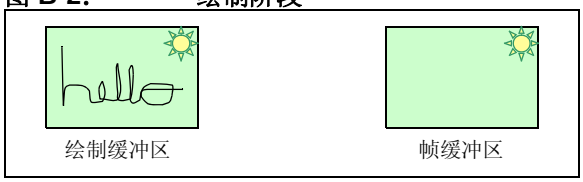
1. 在初始阶段，绘制缓冲区和帧缓冲区具有相同的内容，如图 B-1 所示。

图 B-1： 初始阶段



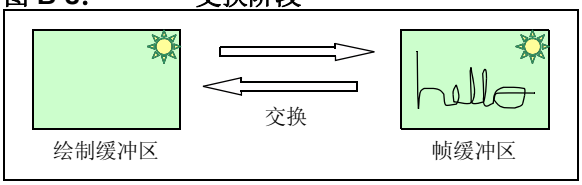
2. 在绘制阶段，只修改绘制缓冲区，因此帧缓冲区对用户不可见，如图 B-2 所示。

图 B-2： 绘制阶段



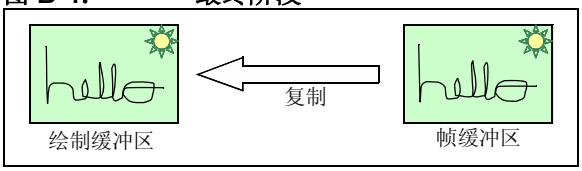
3. 在交换阶段，绘制缓冲区与帧缓冲区交换（通过交换指针），如图 B-3 所示。

图 B-3： 交换阶段



4. 在最终阶段，帧缓冲区的内容被复制到绘制缓冲区，因此，两个缓冲区具有相同的内容，如图 B-4 所示。

图 B-4： 最终阶段



5. 工作流程周而复始地进行。

何时使用双缓冲？

在下列情况下使用双缓冲：

- 屏幕上有大量控件
- 需要解码和显示图像
- 显示分辨率大于 WQVGA （即使上述几点均不成立）

何时不使用双缓冲？

在下列情况下不使用双缓冲：

- 需要即时显示更改。例如，绘制图表。
- 可用于显示的 RAM 有限。
- 显示屏分辨率小于 WQVGA （这不是一个硬规则）。

如何使用双缓冲？

可如下使用双缓冲：

- 在 `GraphicsConfig.h` 中定义 `USE_DOUBLE_BUFFERING` 使能双缓冲。
- 请参见 Microchip 图形库帮助文件中的双缓冲部分获取代码示例。

© 2012 Microchip Technology Inc.

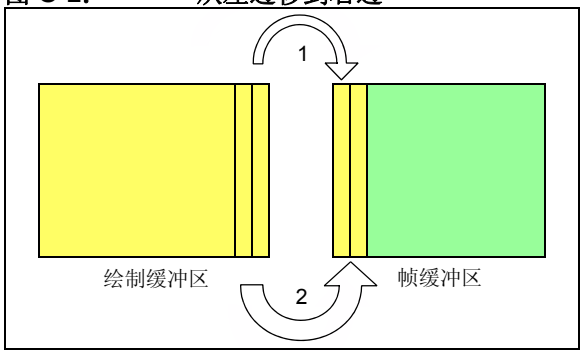
附录 C： RCCGPU 的高级用法

RCCGPU 可用于实现不同种类的图形效果，例如动画和滚动屏幕。使用存储器中与帧缓冲区分开的一个缓冲区实现这些效果，该缓冲区包含要显示在屏幕上的最终图像。然后，该缓冲区的图像以特殊方式传输到帧缓冲区，从而构成屏幕上的各种视觉效果。例如，以垂直方式逐行传输图像到帧缓冲区将呈现剥离效果，而逐行移动现有帧缓冲区并用最终图像填充移动部分呈现滚动屏幕的效果。

部分效果及其算法如下：

1. 从左边移到右边：该效果显示新屏幕从左滑到右。

图 C-1： 从左边移到右边



算法：

第 i 步：在绘制缓冲区中完整创建新屏幕。

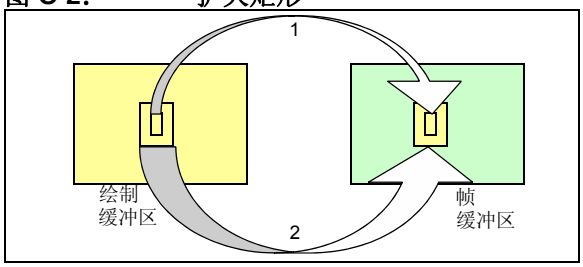
第 ii 步：使用 RCCGPU 矩形复制命令将绘制缓冲区的右边一行（高度 = 屏幕高度，宽度 = 一个像素）移到帧缓冲区的左边。

第 iii 步：使用 RCCGPU 矩形复制命令将绘制缓冲区的右边两行（高度 = 屏幕高度，宽度 = 两个像素）移到帧缓冲区的左边。

重复上述步骤直到整个屏幕传输到帧缓冲区，如图 C-1 所示。

2. 扩大矩形：这种效果显示将矩形从中间扩大到屏幕的边缘。

图 C-2： 扩大矩形



算法：

第 i 步：在绘制缓冲区中完整创建新屏幕。

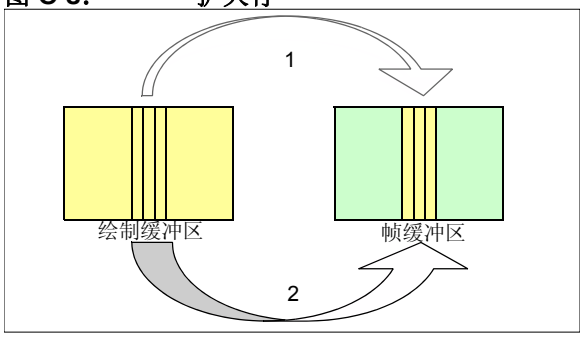
第 ii 步：使用 RCCGPU 矩形复制命令将 2x2 的矩形从绘制缓冲区的中间移到帧缓冲区的中间。

第 iii 步：使用 RCCGPU 矩形复制命令将 3x3 的矩形从绘制缓冲区的中间移到帧缓冲区的中间。

重复上述步骤直到整个屏幕传输到帧缓冲区，如图 C-2 所示。

3. 扩大行：该效果显示将一条垂直线在水平方向从中间扩展到两端。

图 C-3： 扩大行



算法：

第 i 步：在绘制缓冲区中完整创建要显示的新屏幕。

第 ii 步：使用 RCCGPU 矩形复制命令将绘制缓冲区的中间两行（高度 = 屏幕高度，宽度 = 两个像素）移到帧缓冲区的中间。

第 iii 步：使用 RCCGPU 矩形复制命令将绘制缓冲区的中间 4 行（高度 = 屏幕高度，宽度 = 4 个像素）移到帧缓冲区的中间。

重复上述步骤直到整个屏幕传输到帧缓冲区，如图 C-3 所示。同样，很多屏幕过渡效果也可轻松实现。

© 2012 Microchip Technology Inc.

附录 D： 缩写

表 D-1 列出了本文档中使用的缩写。

表 D-1： 缩写

缩写	定义
1 KB	1024 字节
AMOLED	主动矩阵 有机发光二极管
CCFL	冷阴极灯管
CHRGPU	字符图形处理单元
CIF	通用中间格式
CLUT	颜色查找表
CSTN	彩色超扭曲向列
DRAM	动态随机访问存储器
EPMP	增强型并行主端口
GPU	图形处理单元
GUI	图形用户界面
HBLANK	水平消隐
IC	集成电路
IPU	解压缩处理单元
LCD	液晶显示器
LED	发光二极管
MIPS	每秒百万条指令
MSTN	单色超扭曲向列
OLED	有机发光二极管
PCB	印刷电路板
PMP	并行主端口
QCIF	四分之一通用中间格式
QVGA	四分之一视频图形阵列
RCCGPU	矩形复制图形处理单元
RGB	红绿蓝
RTCC	实时时钟和日历
STN	超扭曲向列
TFT-LCD	薄膜晶体管液晶显示屏
UART	通用异步收发器
USB	通用串行总线
VBLANK	垂直消隐
VGA	视频图形阵列
WQVGA	四分之一宽屏视频图形阵列

参考资料

有关使用 Microchip 图形库的更多信息，请参见以下应用笔记：

- AN1136 《如何使用 Microchip 图形库中的控件》（DS01136A_CN）， Paolo Tamayo， Microchip Technology Inc.， 2007。
- AN1246 《如何在 Microchip 图形库中创建控件》（DS01246A_CN）， Paolo Tamayo 和 Harold Serrano， Microchip Technology Inc.， 2009。
- AN1227 《Microchip 图形库与键盘的配合使用》（DS01227A_CN）， Anton Alkhimenok， Microchip Technology Inc.， 2008。
- AN1182 《Microchip 图形库中的字体》（DS01182B_CN）， Paolo Tamayo， Microchip Technology Inc.， 2010。
- DEFLATE Compressed Data Format Specification (RFC1951) —— http://www.ietf.org/rfc/rfc1951.txt

关于 Microchip 产品或图形应用的更多信息，请访问以下链接：

http://www.microchip.com/graphics

© 2012 Microchip Technology Inc.

请注意以下有关 Microchip 器件代码保护功能的要点：

- Microchip 的产品均达到 Microchip 数据手册中所述的技术指标。
- Microchip 确信：在正常使用的情况下， Microchip 系列产品是当今市场上同类产品中最安全的产品之一。
- 目前，仍存在着恶意、甚至是非法破坏代码保护功能的行为。就我们所知，所有这些行为都不是以 Microchip 数据手册中规定的操作规范来使用 Microchip 产品的。这样做的人极可能侵犯了知识产权。
- Microchip 愿与那些注重代码完整性的客户合作。
- Microchip 或任何其他半导体厂商均无法保证其代码的安全性。代码保护并不意味着我们保证产品是 “ 牢不可破 ” 的。

代码保护功能处于持续发展中。 Microchip 承诺将不断改进产品的代码保护功能。任何试图破坏 Microchip 代码保护功能的行为均可视为违反了 《数字器件千年版权法案 （Digital Millennium Copyright Act）》。如果这种行为导致他人在未经授权的情况下，能访问您的软件或其他受版权保护的成果，您有权依据该法案提起诉讼，从而制止这种行为。

提供本文档的中文版本仅为了便于理解。请勿忽视文档中包含的英文部分，因为其中提供了有关 Microchip 产品性能和使用情况的有用信息。Microchip Technology Inc. 及其分公司和相关公司、各级主管与员工及事务代理机构对译文中可能存在的任何差错不承担任何责任。建议参考 Microchip Technology Inc. 的英文原版文档。

本出版物中所述的器件应用信息及其他类似内容仅为您提供便利，它们可能由更新之信息所替代。确保应用符合技术规范，是您自身应负的责任。**Microchip 对这些信息不作任何明示或暗示、书面或口头、法定或其他形式的声明或担保，包括但不限于针对其使用情况、质量、性能、适销性或特定用途的适用性的声明或担保。** Microchip 对因这些信息及使用这些信息而引起的后果不承担任何责任。如果将 Microchip 器件用于生命维持和 / 或生命安全应用，一切风险由买方自负。买方同意在由此引发任何一切伤害、索赔、诉讼或费用时，会维护和保障 Microchip 免于承担法律责任，并加以赔偿。在 Microchip 知识产权保护下，不得暗中或以其他方式转让任何许可证。

商标

Microchip 的名称和徽标组合、 Microchip 徽标、 dsPIC、 KEELOQ、 KEELOQ 徽标、 MPLAB、 PIC、 PICmicro、 PICSTART、 PIC32 徽标、 rfPIC 和 UNI/O 均为 Microchip Technology Inc. 在美国和其他国家或地区的注册商标。

FilterLab、 Hampshire、 HI-TECH C、 Linear Active Thermistor、MXDEV、MXLAB、SEEVAL 和 The Embedded Control Solutions Company 均为 Microchip Technology Inc. 在美国的注册商标。

Analog-for-the-Digital Age、 Application Maestro、 chipKIT、 chipKIT 徽标、 CodeGuard、 dsPICDEM、 dsPICDEM.net、 dsPICworks、 dsSPEAK、 ECAN、 ECONOMONITOR、 FanSense、 HI-TIDE、 In-Circuit Serial Programming、 ICSP、 Mindi、 MiWi、 MPASM、 MPLAB Certified 徽标、 MPLIB、 MPLINK、 mTouch、 Omniscient Code Generation、 PICC、 PICC-18、 PICDEM、 PICDEM.net、 PICkit、 PICtail、 REAL ICE、 rfLAB、 Select Mode、 Total Endurance、 TSHARC、 UniWinDriver、 WiperLock 和 ZENA 均为 Microchip Technology Inc. 在美国和其他国家或地区的商标。

SQTP 是 Microchip Technology Inc. 在美国的服务标记。

在此提及的所有其他商标均为各持有公司所有。

© 2012, Microchip Technology Inc. 版权所有。

ISBN：978-1-62076-079-6

QUALITY MANAGEMENT SYSTEM
CERTIFIED BY DNV
═ ISO/TS 16949 ═

Microchip 位于美国亚利桑那州 Chandler 和 Tempe 与位于俄勒冈州 Gresham 的全球总部、设计和晶圆生产厂及位于美国加利福尼亚州和印度的设计中心均通过了 ISO/TS-16949:2009 认证。 Microchip 的 PIC® MCU 与 dsPIC® DSC、KEELOQ® 跳码器件、串行 EEPROM、单片机外设、非易失性存储器和模拟产品严格遵守公司的质量体系流程。此外， Microchip 在开发系统的设计和生产方面的质量体系也已通过了 ISO 9001:2000 认证。

全球销售及服务网点

美洲

公司总部 Corporate Office
2355 West Chandler Blvd.
Chandler, AZ 85224-6199
Tel: 1-480-792-7200
Fax: 1-480-792-7277
技术支持：
http://www.microchip.com/support
网址：www.microchip.com

亚特兰大 Atlanta
Duluth, GA
Tel: 1-678-957-9614
Fax:1-678-957-1455

波士顿 Boston
Westborough, MA
Tel: 1-774-760-0087
Fax: 1-774-760-0088

芝加哥 Chicago
Itasca, IL
Tel: 1-630-285-0071
Fax: 1-630-285-0075

克里夫兰 Cleveland
Independence, OH
Tel: 1-216-447-0464
Fax: 1-216-447-0643

达拉斯 Dallas
Addison, TX
Tel: 1-972-818-7423
Fax: 1-972-818-2924

底特律 Detroit
Farmington Hills, MI
Tel: 1-248-538-2250
Fax: 1-248-538-2260

印第安纳波利斯 Indianapolis
Noblesville, IN
Tel: 1-317-773-8323
Fax: 1-317-773-5453

洛杉矶 Los Angeles
Mission Viejo, CA
Tel: 1-949-462-9523
Fax: 1-949-462-9608

圣克拉拉 Santa Clara
Santa Clara, CA
Tel: 1-408-961-6444
Fax: 1-408-961-6445

加拿大多伦多 Toronto
Mississauga, Ontario, Canada
Tel: 1-905-673-0699
Fax: 1-905-673-6509

亚太地区

亚太总部 Asia Pacific Office
Suites 3707-14, 37th Floor
Tower 6, The Gateway
Harbour City, Kowloon
Hong Kong
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 北京
Tel: 86-10-8569-7000
Fax: 86-10-8528-2104

中国 – 成都
Tel: 86-28-8665-5511
Fax: 86-28-8665-7889

中国 – 重庆
Tel: 86-23-8980-9588
Fax: 86-23-8980-9500

中国 – 杭州
Tel: 86-571-2819-3187
Fax: 86-571-2819-3189

中国 – 香港特别行政区
Tel: 852-2401-1200
Fax: 852-2401-3431

中国 – 南京
Tel: 86-25-8473-2460
Fax: 86-25-8473-2470

中国 – 青岛
Tel: 86-532-8502-7355
Fax: 86-532-8502-7205

中国 – 上海
Tel: 86-21-5407-5533
Fax: 86-21-5407-5066

中国 – 沈阳
Tel: 86-24-2334-2829
Fax: 86-24-2334-2393

中国 – 深圳
Tel: 86-755-8203-2660
Fax: 86-755-8203-1760

中国 – 武汉
Tel: 86-27-5980-5300
Fax: 86-27-5980-5118

中国 – 西安
Tel: 86-29-8833-7252
Fax: 86-29-8833-7256

中国 – 厦门
Tel: 86-592-238-8138
Fax: 86-592-238-8130

中国 – 珠海
Tel: 86-756-321-0040
Fax: 86-756-321-0049

亚太地区

台湾地区 – 高雄
Tel: 886-7-536-4818
Fax: 886-7-330-9305

台湾地区 – 台北
Tel: 886-2-2500-6610
Fax: 886-2-2508-0102

台湾地区 – 新竹
Tel: 886-3-5778-366
Fax: 886-3-5770-955

澳大利亚 Australia - Sydney
Tel: 61-2-9868-6733
Fax: 61-2-9868-6755

印度 India - Bangalore
Tel: 91-80-3090-4444
Fax: 91-80-3090-4123

印度 India - New Delhi
Tel: 91-11-4160-8631
Fax: 91-11-4160-8632

印度 India - Pune
Tel: 91-20-2566-1512
Fax: 91-20-2566-1513

日本 Japan - Osaka
Tel: 81-66-152-7160
Fax: 81-66-152-9310

日本 Japan - Yokohama
Tel: 81-45-471- 6166
Fax: 81-45-471-6122

韩国 Korea - Daegu
Tel: 82-53-744-4301
Fax: 82-53-744-4302

韩国 Korea - Seoul
Tel: 82-2-554-7200
Fax: 82-2-558-5932 或
82-2-558-5934

马来西亚 Malaysia - Kuala Lumpur
Tel: 60-3-6201-9857
Fax: 60-3-6201-9859

马来西亚 Malaysia - Penang
Tel: 60-4-227-8870
Fax: 60-4-227-4068

菲律宾 Philippines - Manila
Tel: 63-2-634-9065
Fax: 63-2-634-9069

新加坡 Singapore
Tel: 65-6334-8870
Fax: 65-6334-8850

泰国 Thailand - Bangkok
Tel: 66-2-694-1351
Fax: 66-2-694-1350

欧洲

奥地利 Austria - Wels
Tel: 43-7242-2244-39
Fax: 43-7242-2244-393

丹麦 Denmark-Copenhagen
Tel: 45-4450-2828
Fax: 45-4485-2829

法国 France - Paris
Tel: 33-1-69-53-63-20
Fax: 33-1-69-30-90-79

德国 Germany - Munich
Tel: 49-89-627-144-0
Fax: 49-89-627-144-44

意大利 Italy - Milan
Tel: 39-0331-742611
Fax: 39-0331-466781

荷兰 Netherlands - Drunen
Tel: 31-416-690399
Fax: 31-416-690340

西班牙 Spain - Madrid
Tel: 34-91-708-08-90
Fax: 34-91-708-08-91

英国 UK - Wokingham
Tel: 44-118-921-5869
Fax: 44-118-921-5820

11/29/11

© 2012 Microchip Technology Inc.