第1章 µC/GUI的介绍

µC/GUI

µC/GUI 是一种用于嵌入式应用的图形支持软件。它被设计用于为任何使用一个图形 LCD 的应用提供一个有效的不依赖于处理器和 LCD 控制器的图形用户接口。它能工作于单任务或多任务的系统环境下。µC/GUI 适用于使用任何 LCD 控制和 CPU 的任何尺寸的物理和虚拟显示。它的设计是模块化的，由在不同的模块中的不同的层组成。一个层，称作 LCD 驱动程序，包含了对 LCD 的全部访问。µC/GUI 适用于所有的 CPU，因为它 100%由的 ANSI 的 C 语言编写的。

µC/GUI 很适合大多数的使用黑色/白色和彩色 LCD 的应用程序。它有一个很好的颜色管理器，允许它处理灰阶。µC/GUI 也提供一个可扩展的 2D 图形库和一个视窗管理器，在使用一个最小的 RAM 时能支持显示窗口。

本文档的目的

本指南描述如何安装，配置和在嵌入式应用中使用µC/GUI 图形用户界面。它也说明了软件的内部结构。

假设

本指南假定你对 C 编程语言已经具有一个扎实的认识。

如果你觉得你对 C 语言的认识不是很充分的话，我们推荐该由 Kernighan 和 Richie 编写的“C 语言编程语言”给你，它描述了程序设计标准，而在新版中，也包含了 ANSI 的 C 语言标准。汇编语言编程的知识不需要。

1.1 需求

在你使用μC/GUI 进行软件开发时，并不需要一个目标系统；只需要使用模拟器，大多数软件就能够进行开发。然而，最后的目的通常是能够在一个目标系统上运行该软件。

目标系统（硬件）

你的目标系统必须：

- 有一个 CPU（8/16/32/64 位）
- 有最少的 RAM 和 ROM
- 有一个完全的图形 LCD（任何类型和任何分辨率）

内存需求的变化取决于软件的哪些部分被使用以及你的目标编译程序的效率有多高。所以指定精确值是不可能的，但是下面的数值适合典型系统。

小的系统（没有视窗管理器）

- RAM：100 字节
- 堆栈：500 字节
- ROM：10～25KB（取决于使用的功能）

大的系统（包括视窗管理器和控件）

- RAM：2～6KB（取决于所需窗口的数量）
- 堆栈：1200 字节
- ROM：30～60KB（取决于使用的功能）

注意，如果你的应用程序使用许多字体的话，ROM 的需求将增加。以上所有的数值都是粗略的估计，不能得到保证。

开发环境（编译程序）

使用什么样的 CPU 并不重要；仅仅需要一个与 ANSI 兼容的 C 编译器。如果你的编辑器有一些限制，请告知我们，我们将通知你在编译软件时是否会带来问题。我们所知道的任何用于 16/32/64 位 CPU 或者 DSP 的编译器都可以使用；大多数的 8 位编译器也可以使用。

一个 C++编译器并不需要，不过可以使用。因此，如果想要的话，应用程序也可以用 C++

语言来编制。

1.2 µC/GUI 的特点

µC/GUI 被设计用于给使用一个图形 LCD 的任何应用程序提供一个高效率的，与处理器和 LCD 控制器无关的图形用户界面。它适合于单一任务和多任务环境，专用的操作系统或者任何商业的实时操作系统（RTOS）。µC/GUI 以 C 源代码形式提供。它可以适用于任何尺寸的物理和虚拟显示，任何 LCD 控制器和 CPU。其特点包括下列这些：

一般特点

- 任何 8/16/32 位 CPU；只需要一个与 ANSI 兼容的 C 编译器。
- 任何控制器支持（如果有合适的驱动程序）的任何（单色的，灰度级或者彩色）LCD。
- 在较小显示屏上，可以不要 LCD 控制器工作。
- 使用配置宏可以支持任何接口。
- 显示屏大小可配置。
- 字符和位图可能是写在 LCD 上的任一点，而不仅仅局限于偶数的字节的地址。
- 程序对大小和速度都进行了最优化。
- 允许编译时的切换以获得不同的优化。
- 对于较慢的 LCD 控制器，LCD 能够被存储到内存当中，减少访问的次数使其最小，从而得到非常高的速度。
- 清晰的结构。
- 支持虚拟显示；虚拟显示能够比实际的显示表现更大尺寸的内容。

图库

- 支持不同颜色深度的位图。
- 有效的位图转换器。
- 绝对没有使用浮点运算。
- 快速线/点绘制（没有使用浮点运算）。
- 非常快的圆/多边形的绘制。
- 不同的绘画模式。

字体

- 为基本软件提供了不同种类的字体：4*6，6*8，6*9，8*8，8*9，8*16，8*17，8*18，24*32，以及 8，10，13，16 等几种高度（以象素为单位）的均衡字体。更多的信息，参见第 25 章：“标准字体”。

- 可以定义和简便地链接新的字体。
- 只有用于应用程序的字体才实际上与执行结果链接，这样保证了最低的 ROM 占用。
- 字体可以分别在 X 轴和 Y 轴方向上充分地缩放。
- 提供有效的字体转换器，任何在你的主系统（即 Microsoft Windows）上的有效字体都可以转换。

字符串/数值输出程序

- 程序支持任何字体的十进制，二进制，十六进制的数值显示。
- 程序支持任何字体的十进制，二进制，十六进制的数值编辑。

视窗管理器（WM）

- 完全的窗口管理器包括剪切在内。一个窗口的外部区域的改写是不可能的。
- 窗口能够移动和缩放。
- 支持回调函数（可选择用法）。
- WM 使用极小的 RAM（大约每个窗口 20 字节）。

可选择用于PC外观的控件

- 控件（窗口对象）有效。它们一般自动运行，并且易于使用。

触摸屏和鼠标支持

- 对于比如按钮控件之类的窗口对象，μC/GUI 提供触摸屏和鼠标支持。

PC工具

- 模拟器及观察器。
- 位图转换器。
- 字体转换器。

范例和演示

关于μC/GUI 能做什么，为了给你一个更好的概念，我们准备有不同的演示作为可运行的仿真程序,在目录 sample\EXE 下。范例程序的源代码位于 Sample 目录。文件夹 Sample\GUIDemo 包括一个展示大部分μC/GUI 特点的应用程序。

1.3 估价板

一个完全的评估板包括一个带有 LCD 的演示板，一个 C 语言编译器和一个有效的范例工程。它已经设计好，主要测试和验证µC/GUI，并且它可用于熟悉这个软件。

评估板

评估板包括 Mitsubishi M30803 CPU 和 SED13705 LCD 控制器（包括原理图和技术资料）。

LCD（320×240 像素）或者单色的 LCD，1/4VGA 彩色显示 LCD 或者 TFT。

更详细的资料，请访问我们的网站：www.micrium.com。

1.4 如何使用本手册

该手册说明了如何安装，配置和使用µC/GUI。它描述了软件的内部结构和µC/GUI 提供的所有的功能（应用程序接口，或者 API）。

在实际上使用µC/GUI 之前，你应该阅读或者至少浏览本手册，对该软件做到耳濡目染。推荐下列步骤：

- 拷贝µC/GUI 文件到你的电脑。
- 仔细研究第 2 章：“入门指南”。
- 使用模拟器多熟悉一理这个软件能作什么（参考第 3 章：“仿真器”）。
- 使用本手册其余部分的参考资料扩展你的程序。

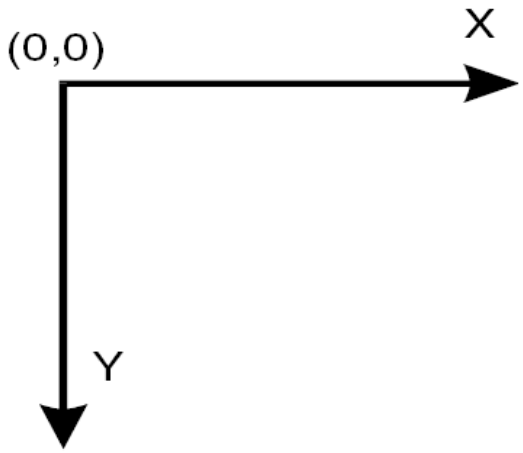
排版上的语法约定

本手册使用下列印刷惯例：

类型	用于
Body	正文文字。
Keyword	那些你在命令-提示中输入的文字，或者那些能在显示屏上看得见的文字（即系统函数，文件或者路径名）。
Parameter	API 函数中的参数。
Sample	在程序范例中的范例代码。
New Sample	那些已经被加到一个已存在有程序范例中的范例代码。

1.5 屏幕和坐标

屏幕由能够被单独控制的许多点组成。这些被称作像素。大部分µC/GUI 在它的 API 中向用户程序提供的文本和绘图函数能够在任何指定像素上写或绘制。



水平刻度被称作 X 轴，而垂直刻度被称作 Y 轴。一个二维坐标用 X 轴和 Y 轴坐标表示，即值(X,Y)。在程序中需要用到 X 和 Y 坐标时，X 坐标总在前面。显示屏（或者一个窗口）的左上角为一默认的坐标（0,0）。正的 X 值方向被总是向右；正的 Y 值方向总是向下。上图说明该坐标系和 X 轴和 Y 轴的方向。所有传递到一个 API 函数的坐标总是以像素为单位所指定。

1.6 如何连接LCD到微控制器

µC/GUI 处理所有的 LCD 访问。事实上任何 LCD 控制器都能够被支持，不取决于它是如何访问的。至于细节，请参阅第 20 章：“低层配置”。此外，如果你的 LCD 控制器不被支持的话，请与我们保持联系。我们目前为全部有销售 LCD 控制器编写驱动程序，对于你打算使用

的 LCD 控制器已经有一个经过验证的驱动程序提供。在你的应用程序中写这样的用于访问 LCD 的程序（或者宏）通常非常简单。如果你的目标硬件有需要的话，Micrium 公司可以提供定制的服务。

LCD 如何与系统连接并不真的重要，只要它通过软件以某种方式达到，可能是按多种方式完成的。大多数这些接口通过一个提供源代码方式的驱动程序来支持。这些驱动程序通常不需要修改，但是用于你的硬件，要通过修改文件 LCDConf.h 进行配置。有关根据需要如何定制一个驱动程序到你的硬件在第 22 章：“LCD 驱动程序”中说明。最通用的访问 LCD 的方式如下所描述。如果你只是想领会如何使用μC/GUI，你可以跳过本节。

带有存储映像LCD控制器的LCD

LCD 控制器直接连接到系统的数据总线，意思是能够如同访问一个 RAM 一样访问控制器。这是一个很有效的访问 LCD 控制器方法，最值得推荐。LCD 地址被定义为段 LCDSEG，为了能访问该 LCD，连接程序/定位器只需要告知这些段位于什么地方。该位置必须与物理地址空间中访问地址相吻合。驱动程序对于这类接口是有效的，并且能用于不同的 LCD 控制器。

带有LCD控制器的LCD连接到端口/缓冲区

对于在快速处理器上使用的较慢的 LCD 控制器，端口-连线的使用可能是唯一的方案。这个访问 LCD 的方法有稍微比直接总线接口慢一些的缺点，但是，特别是使用一个减少 LCD 访问次数的高速缓存的情况，LCD 刷新并不会有大的延迟。所有那些需要处理的是定义程序或者宏，设置或者读取与 LCD 连接的硬件端口/缓冲区。这类接口也被用于不同的 LCD 控制器的不同的驱动程序所支持。

特殊方案：没有LCD控制器的LCD

LCD 可以不需要 LCD 控制器而进行连接。在这种情况下，LCD 数据通常通过控制器经由一个 4 或 8 位移位寄存器直接提供。这些特殊的硬件方案有价格便宜的优点，但是使用上的缺点是占用了大部分有效的计算时间。根据不同的 CPU，这会占到 CPU 的开销的 20%到几乎 100%之间；对于较慢的 CPU，它根本是极不合理的。这类接口不需要一个特殊的 LCD 驱动器，因为 μC/GUI 简单地将所有显示数据放入 LCD 高速缓存中。你自己必须写硬件相关部分软件，周期性地将数据从高速缓存的内存传递到你的 LCD。

对于 M16C 和 M16C/80，传递图像到显示屏中的范例代码可以用“C”和优化的汇编程序实现。

1.7 数据类型

因为 C 语言并不提供与所有平台相吻合的固定长度的数据类型，大多数情况下，µC/GUI 使用它自己的数据类型，如下表所示：

数据类型	定义	说明
I8	signed char	8 位有符号值
U8	unsigned char	16 位无符号值
I16	signed short	16 位有符号值
U16	unsigned short	16 位无符号值
I32	signed long	32 位有符号值
U32	unsigned long	32 位无符号值
I16P	signed short	16 位（或更多）有符号值
U16P	unsigned short	16 位（或更多）无符号值

对于大多数 16/32 位控制器来说，该设置将工作正常。然而，如果你在你的程序的其它部分中有相似的定义，你可能想对它们进行修改或者重新配置。一个推荐的位置是置于配置文件 LCDConf.h 中。