

什么是 HMI

HMI 是 Human Machine Interface 的缩写,“人机接口”,也叫人机界面。人机界面是系统和用户之间进行交互和信息交换的媒介,它实现信息的内部形式与人类可以接受形式之间的转换。凡参与人机信息交流的领域都存在着人机界面。

什么是串口 HMI

串口 HMI 就是设备封装好 HMI 的底层功能以后,通过串口(USART 232)与用户 MCU 进行交互,比如 MCU 可以随时通过 USART 发指令通知设备切换某个页面或者改变某个组件的属性。设备也可以随时通过 USART 通知用户 MCU 操作者目前触摸了页面上的某个组件或者设备当前进入了某个页面。

串口 HMI 和普通显示屏有何区别,我该如何选型。

对于产品研发者来说,产品研发初期可以选型的接口无非就 3 种类型:RGB 接口,MCU 总线接口,串口 HMI。

RGB 接口:

RGB 接口必须用在带有 RGB 驱动的 ARM 芯片上,一般的 ARM9 芯片有少许支持 RGB 的,ARM9 以上的芯片多数支持 RGB.但是此类接口的驱动是最复杂的,对硬件要求也是最高的。详细的驱动细节这里就不多讲了。

MCU 总线接口:

MCU 总线接口驱动比 RGB 简单一些,对硬件也基本没有任何要求,只要是个 MCU 都可以驱动。但是显示速度是个比较大的瓶颈,大多数用户的 MCU 都是 51 内核或者 stm32 这样的 ARM7 内核。驱动总线接口的屏速度不是很理想。当然 ARM7 内核也有超高速的。但是芯片成本也比较高,用的人也比较少。除了速度瓶颈之外,界面的显示驱动对于大多数用户来说也是很头疼的。总线型接口的屏只提供点阵的操作。图片,字符等任何显示内容都是通过取模数据,在屏幕上相应的位置把点阵一个一个的打出来。在此基础上再来实现人机界面的逻辑。工作量很大。通常情况下,比如做一个英文键盘可能开发者就得耗费几个星期的时间来制作。并且后期修改的时候也是相当的吃力。

串口 HMI

对于开发者来说,串口 HMI 是最简单的显示方案。首先他跟 MCU 总线屏一样对用户的硬件没有任何要求,其次。他没有速度瓶颈,因为界面的显示是设备内部自己实现的,用户 MCU 只是发

送指令，并不需要底层驱动。再次，针对显示的人机界面的布局和大多数的逻辑（比如界面背景，按钮效果，文本显示等）。全部都不需要用户的 MCU 参与，使用设备提供的上位软件，在电脑上点几下鼠标就完成了。制作好资源文件以后下载到屏幕即可自动运行，剩下的就是 USART 交互了（运行中用户 MCU 通过简单的对象操作指令来修改界面上的内容）。

串口 HMI 虽然是最简单的显示方案，那是不是意味着他是最高成本的显示方案。

如果单纯从硬件的生产成本来讲，串口 HMI 确实成本要高一点点。但是这里我们要考虑两个问题，第一：是否值得多花这个成本去使用这个先进的功能？第二：除了生产成本之外，研发成本和后期维护成本您考虑进去没？研发周期过长导致新产品延期上市造成的损失您考虑进去没？对于这两个问题。我们认为这里是没有绝对答案的，见仁见智，不同的人会得出不同的答案。

能否用简单的语言总体概括一下到底什么是串口 HMI,好让我可以快速了解他的特性。

不管是 RGB 接口屏还是 MCU 总线接口屏，开发者想要显示任何内容（注意是任何内容，不管图片，文字，还是刷色）归根到底，用户的 MCU 都是在对屏幕上的点进行底层绘制。任何图形都是用户在控制屏幕上的每一个点的状态。直观的说就是用户 MCU 控制的是屏幕上的点阵。

而串口 HMI 则颠覆了这个应用。对屏幕上点阵的控制现在交给了设备内部的主控芯片。面向用户的不再是点阵。那是什么呢？你猜对了！是控件。什么是控件？就是串口 HMI 封装好的一个功能模块。控件从哪里来的呢？配套的上位软件里面创建出来的,比如要在左上角显示一段文本，就在左上角创建一个文本控件，要在右下角显示一个按钮，就在右下角创建一个按钮控件，等等。引入控件操作以后，用户 MCU 无需理会一个内容的显示需要控制哪些点的显示状态。甚至是坐标，都不需要在用户 MCU 里操作。用户从此只需要关注的是屏幕上的这些控件的属性。在运行中用户 MCU 通过串口指令改变控件的属性，就可以改变屏幕上显示的内容。除此之外控件还有触摸事件功能。就是它被触摸以后可以主动通知用户，也可以自动执行一些指令。


文本操作

如果用户要改变此页面上的第一条文本内容，只需要串口发送指令：`t0.txt=" abc"`，当前显示的List1即可自动改变为abc.

t0为控件名称（上位软件编辑界面的时候创建的一个文本控件），txt为此控件的一个属性，List1是创建时为他指定的初始值，除此txt属性之外，文本控件还有字体颜色属性(pco)，横向对齐方式属性(xcen)，字库属性(font)等，都可以随时通过指令更改



按钮操作

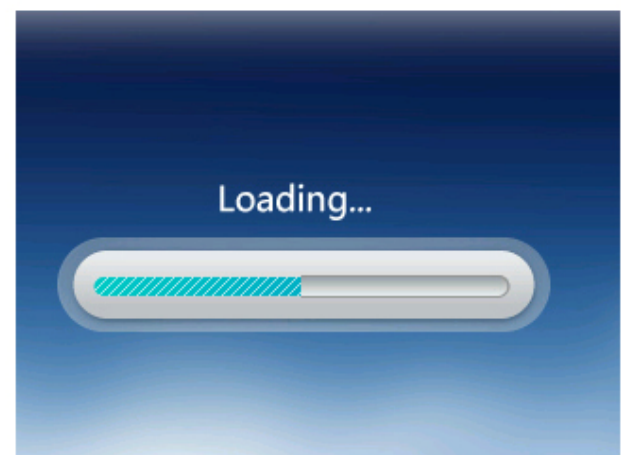
如果操作者触摸左边第一个按钮，屏幕将自动显示按下效果为： 然后可以在触摸事件里面设置 自动切换到别的页面，或者执行某些指令，或者通过串口主动通知用户MCU此按钮被按下，关于通知数据格式请参看”基本型串口HMI指令集”PDF文件中的”数据返回格式”表



进度条操作

如果想要此页面的进度条打到80位置，只需串口发送指令：`j0.val=80`

j0为进度条名称（上位软件编辑界面时创建），进度条的背景图片和前景图片都是在界面编辑时选择好的。下位MCU只需要更改val属性屏幕即可自动刷新进度



指针操作

如果想要此页面的第一个表盘指针转到90度位置，只需发送指令：`z0.val=90`
运行中如果要改变指针颜色只需发送指令
`z0.pco=RED`或者`z0.pco=1024`



除以上控件之外，还有图片控件，切图控件，触摸热区控件等等（并且我们会不定期的增加一些更新更实用的控件，请随时关注我们的最新上位软件和最新使用说明）。

基本型串口HMI独有的三大特色

- 屏幕接受纯字符串指令，方便阅读与调试
- 控件属性赋值支持简易运算，提高开发效率
- 屏幕固件自动升级，随时享受最新版本功能



为什么要使用字符串指令

有些人会说字符串指令解析速度没有十六进制指令解析速度快！对，确实是这个道理，但是指令解析速度会影响用户体验吗？假如屏幕接受十六进制指令解析是花 1us,解析字符串指令是花 2us,请问，当屏幕收到你要给某个文本控件的 txt 属性赋值的指令后，它在 1us 后更新文本显示和在 2us 后更新文本显示有区别吗？这个只是解析指令的速度，不代表内容的刷新速度，我们的串口屏全屏界面更新速度为：小于 4.3 寸的屏是 50ms 以下,大于等于 4.3 寸的屏是 18ms 以下。如果使用繁琐的 0XFF,0XFE,0X02 这样的 16 进制来做指令，那么你后期阅读单片机代码的时候将完全摸不着头脑。当然，为了方便用户 MCU 处理接受到的数据，我们的屏幕数据返回是用的 16 进制数据，不过这个不属于指令的范畴，只是数据返回格式罢了，呵呵！

控件属性赋值支持简易运算

进度条 j0 的 val 属性赋值正常情况下是这么写: j0.val=10, 不过,你还可以这么写: j0.val=j0.val+10 甚至你还可以这么写: j0.val=j0.val/2+10. 更不要命一点: j0.val=j1.val-j0.val*2+dim 含义不用我解释了吧？聪明的你肯定是已经看懂了。最后一句里面的 dim 是系统变量，表示背光当前亮度值，最小 0，最大 100，可以读取它，也可以对它赋值（“基本型串口 HMI 指令集”PDF 文件中有列出所有的系统变量说明）。不过这里要提前说明一点就是我们的加减乘除运算统一是从左到右的优先级，并且不支持括号。

屏幕固件自动升级

我们的上位软件编译出来的资源文件就包含了最新的屏幕固件，任何功能性的升级或者 bug 修复，你只需要使用我们最新的上位软件，编译出资源文件后下载到屏幕，屏幕固件立刻自动升级。

顺便悄悄的告诉你，我们的上位软件有自动更新功能，只要你的电脑网络是畅通的，你的电脑防火墙没有隔离我们的软件。每次启动软件的时候软件检测到新版本会提示您是否立即升级。

看到这里，可能完全不懂串口 HMI 是什么的朋友已经对我们的产品有了初步的了解了，具体的界面编辑步骤请观看我们资料中提供的”USART HMI 完整视频”。