32*8LED 点阵电子钟的制作说明

作者：杜洋

2005 年 10 月 22 日

近日完成了 32*8 点阵电子钟的部分制作，现在就在我的身边正常的行动着。在此写一下制作及注意事项，便于大家共同进步。

我的这款 32*8LED 点阵电子钟是用多层夹板组成的，所以其非常像三明治，我就把它叫三明治电子钟了。这个电子钟的主要元件有 4 块 8*8 的单色行列式点阵屏单片机控制器，时钟发生芯片，温度检测芯片和一个 BCD 转 16 线的转换芯片。大家也可以根据自己的需要加装蜂鸣器和红外遥控接收头之类的外围接口。

这台电子钟最关键的地方是 LED 的流动显示和对接口芯片数据的读取上。以下就原理方面就不像大家细说了。旨在详细的阐述一下制作与程序的设计。

电路的设计：

为了有效的利用 I/O 资源，我们利用外扩的芯片的方法解决。74HC154 是 4 线 BCD 码转换为 16 线的译码器，在电路中用作列的扫描。而行则是由单片机的 P0、P2 口直接驱动的，显示方式应该是逐列扫描。为了增加驱动能力，提高 LED 的显示亮度，我用了三极管 8050 和 8550 作显示驱动。

时钟的发生我采用了目前较流行的 DS1302 实时时钟芯片，而没有用单片机直接产生，因为单片机产生日期星期计算是比较复杂而且不精准的。而 DS1302 具有涡流电池备电，2100 年以前的日期、时间计算、SPI 总线通信。并在内部集成了 31 个静态 RAM。使用这款芯片可以减小 CPU 的工作量，使 CPU 主攻 LED 的扫描。

温度检测我采用了 DS18B20 温度转感器，它是普通的三极管封装，内部直接将温度变成数字信号，并用先进的单总线输出。抗扰力强，占用资料少。更在内置的 EEPROM。没用 AD 转换，使系统更简单。

在制作时一定请注意 P0 口和 DS18B20 的上拉电阻。LED 屏的引线应先用万用表测好，行和列标出。我是引用一个 16*16 屏的电路图的原理，只是将它的拼成一行而已，再在软件上加以修理就行了。所以会作网上流行的 16*16 的屏再作我的电子钟就简单了许多。电源最好是用 9V 的稳压电压通过 7805 稳压得到，这样可以得到大一点的驱动力。

程序的设计：

硬件部份对于有一定的电子制作基础的朋友并不是很难，甚至可以说是简单。这个东东的关键是程序的设计，思路、实现方法、技巧是我们要了解的重点之所在。

首先我来解决一下 LED 屏的显示问题吧！我们的时间显示是可以向左流动的

数据显示，了解 16*16LED 点阵屏的朋友会知道他是用查表的方法来实现的，可是查表的方法只配合不变的或是很少改变的数据，而我们所要显示的时间是随时改动的。对此，我们引用了显示缓冲区的方法，也就是在单片机的 RAM 中开辟一定的显示单元（视显示的字节数而定，本屏开了 32 个缓冲单元），LED 屏的显示子程序始终从这个缓冲区中取显示的数据来显示，而我们的流动程序只要将显示缓冲区的数据左移或右移就可以了。而我们待显示的数据也是每隔一段时间将时钟和温度数据通过转换程序变成显示数据送入显示缓冲区的。

看过我的电子钟的视频录像的朋友一定会问了，待显示的数据很长，你的显示缓冲区只有 32 个字节，怎么显示呢？在此，我们又开了一个 RAM 备用显示缓冲区，用来存放一个整屏没有显示完全的数据部份，我们用一个移动程序让备用显示缓冲区的数据移入显示缓冲区，当所有要显示的数据完成之后我们再读取新的时间和温度信息并转换成显示数据送入备用显示缓冲区。我们的数据显示问题就解决了。在此顺便说一下，现在有许多的爱好都朋友在学单片机时总是认为 RAM 没有什么太大的用处，其实正好相反。RAM 在越高级的程序和系统中的用处就越大，数据的堆栈，缓冲区，信息处理等都对 RAM 有一定的要求，因为我们平日的程序是小规模的，所以感觉 128BIT 的 RAM 已经是太多了。我认为只有当你可以很好的利用单片机的全部资源（如 RAM，FALSH，定时/计数器，串口，内置比较器等）才算是真正的入门单片机，我也正和大家一样朝这个方向努力。

显示的问题解决了，下面看看怎么进行数据读取、转换并送入显示缓冲区的吧。我所说的只是一种思路，实际的程序还得大家自己考虑。其实学习最重要的不是学会某一样事物，而从事物中学会一种好的学习方法，用这种方法来应对更多新的知识和技术。一个一个的学是跟不上技术的发展速度的。拿来看看资料就会用了，才是我们的目的，不是吗？

我们从时钟芯片和温度传感器中得到的只是十六进制或 BCD 码的数据，首先我们就得将其变为显示的数据类型，也就是每一列的字节数据。这是一组数据，用这一组数据字节组成一个要显示的值。这里我们用查表法将事先设定好的每一个数据的值逐一取出送入备用显示缓冲区。这一过程要在上一次循环显示完全结束后才可以进行，不然会显示乱码的。这样我们就得到了一个流程：读取时间、温度数据-----将其转换为一组表值数据-----将全部数据一次性送入备用显示缓冲区------当显示数据完全流出了备用显示缓冲区后---读取时间、温度数据-（循环）。时间显示的流动我们用定时器来完成，这样不会再占用 CPU 时间让显示流畅且稳定性高。而 1~9，“月”，“日”，这样的数据表是分为不同的表的，我们在每一个表的最后加一个结束标志数据，当查表程序发现了标志数据后就开始查下一个数据表了。而一次整体循环的最后也应有一个标志位，告诉程序已经完成了一次显示应该重新读时间、温度并送入备用显示缓冲区了。其中，“月”，“日”，“：”等是固定的显示，到位显示就行了，不需要根据时间、温度数据改变。

设计时应注意的问题：

显示部分最应该注意合理的利用 RAM 空间，如果是用 89S52 的 RAM 为 256BIT。显示缓冲区加上备用缓冲区的数据不要超过 RAM 的范围。尽量选择高一点的晶振，提高读取时间、温度数据的时间，以免导致 LED 显示断帧。因为大部分数据处理都是在流动中断程序中，所以要做好程序的堆栈。适当的改变显示流动

速度使人眼看着更舒服的同时要兼顾时间、温度处理的速度。

时钟芯片要注意的地方很多，它输出的数据不是十六进制数而是有一定规律的 BCD 码数据，通信协议和内部 RAM 的结构和相关资料中是明确的介绍。涓流供电时要注意是不是在内部的控制寄存器中启动了涓流供电。时钟在程序写好后一定要有时钟启动命令和数据写入允许命令。目前我还没有写键盘时间调整的程序，只是在程序开始时写入一个当前的时间。为了不让每一次复位单片机都重新写入初始化时间而导致时钟不准，我们采用了 DS1302 内部的 RAM 寄存器作一个初始化标志。当第一次烧入程序时，程序开始除了写入时间数据时还在第 1 个 RAM 字节上输入一个数据（任意值），而在程序的最最开始写入一个读第 1 个 RAM 的命令，如果这个数据和我们最先设定的数据相同就表示我们不是第一次初始化的时间写入，如果数据为 0 或其它的值就说明这是首次烧入，并调用写入初始时间程序。其中的技巧众多，开动你的脑筋看看有没有更好的方法。

温度是单总线的 DS18B20 数字温度传感器，直接数字信号输出，用现成的程序就可以读出数据，这没什么可说的。值得注意的是单总线的上拉电阻和复杂的通信协议对时序的要求。18B20 内部的 EEPROM 可以用来扩展温度报警和数据、标志的掉电保存，这些由大家奇思妙想吧。

一般的制作思路和注意事项已经粗略的讲述了，余下的就靠大家在实际制作中研究、探索了。如果在文章中有什么错误或是对我有的文章有什么意见和建议也欢迎与本人联系。

最终祝大家学习进步！