实验名称：_____仿真器配置_____姓名：_____严旭铧_____学号：___3220101731

专业：电气工程及其自动化
姓名：严旭铧
学号：3220101731
日期：2024.3.14
地点：紫金港东三 406

浙江大学实验报告

课程名称：___微机原理与应用综合实验____指导老师：__ 胡斯登___ _成绩：_____
实验名称：_____仿真器配置_____实验类型：_微机实验_____同组学生姓名：褚玘铖

实验 1　仿真器配置

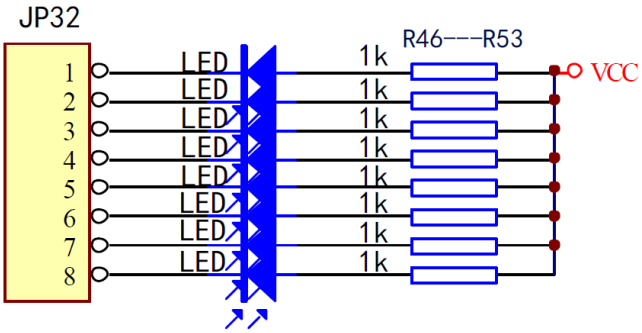
一、 仿真器配置

1. 安装仿真器驱动程序，此时要注意禁止数字证书。
   在 Win11 中，开始菜单→设置→**系统→恢复→高级启动**→立即重新启动→疑难解答→高级选项→启动设置→重启→按 7 或 F7 禁用驱动程序强制签名。
   ※注：每次计算机重启后，要正常连接仿真器硬件，似乎都要进行上述流程禁止数字证书。
2. 仿真器有两条线，一条 A-A 线用于和 PC 传输信息，一条 A-B 线用于供电。必须注意，接线时断电。
3. 打开伟福后，不勾“使用伟福软件模拟器”，仿真器型号为 SP51-AT89S51。如果反复出现该框，说明连接不好，检查接口，或者是否忘记了禁用签名，或者是仿真器盒子的开关没有拨上去。正常应该是呈现下面的状态：

4. 在伟福中新建文件和项目时的流程：
   (1) 先在需要存放这些程序的目录新建文件夹，命名一般是英文/数字/下划线。
   (2) 打开 VW，新建文件→保存文件→找到目标目录→命名为 name.ASM→同目录下新建项目，命名为 name（自动补全为.PRJ）→添加模块文件 name.ASM→**不要添加包含文件**。

二、 按键控制 LED 灯

1. LED 灯亮灭控制原理
   LED 为发光二极管，正向电位差足够大才能正向导通。如图所示，阴极接 I/O 口，阳极接上拉电阻后接到 VCC。阳极侧始终保持高电平输入（逻辑 1）。显然，要正向导通发光，需要 I/O 口电位够低,因此需要输出低电平（逻辑 0）。
   因此，要实现 LED 亮灭的各种效果，重点是，在 I/O 口输出怎样的“01”序列。



2. 按照课件上的代码进行尝试，按钮按下则接通 VCC，P0 的 8 位分别对应了 8 个按键，按下的按键对应的那一位的值就是 0。全黑时，查看 P0 的值为 FF（1111 1111）；全亮时，查看 P0 的值为 00。几次验证后，发现 P0 输入的按键 K1~K8 对应 P0 储存的字节的低位到高位，对应控制 D00~D07 灯。可以作出对应表：
   根据该表可以更方便地实现诸如从左到右流动等效果。

表 1 P0 口各位与 LED 的对应关系

P0 各位	8（高）	7	6	5	4	3	2	1（低）
LED	D07	D06	D05	D04	D03	D02	D01	D00

例如，P0 输入为 5B（0101 1011）时，从上到下看依次为（xxox xoxo x 为暗，o 为亮）。

三、 实现 LED 灯的“滚动”，闪烁和呼吸

1. 滚动的实现（以三个灯亮滚动为例）

(1) 动起来：

从灯的状态抽象出来分析，滚动一次，前后的变化无非是(xxxx xooo)→(oxxx oooo)or(xxoo ooox)。以灯从低位到高位滚动为例，用逻辑状态来表示，(1110 0000)→(1100 0001)→……→(1110 0000)。

指令集中有一个天然适配的指令：**累加器移位/循环指令**。低位到高位滚动，逻辑状态上看是向左移，不需要考虑进位，用 **RL A**。反之用 RR 即可。

(2) 能看见：

同时，因为单片机指令执行很快，以 89S51 为例，晶振为 12MHz，一个机器周期为 1us，执行一条指令至少需要 1us，至多需要 4us。人眼能分辨的最小周期大概在 1/10~1/60s 之间，要使得被看见，需要增加延迟时间，让滚动的指令延迟之后再执行。

那么需要追加一个延迟用的 Delay 函数。最简易的视线思路就是利用 **DJNZ R0,$**指令，对寄存器中的值进行自减判断，非零跳转自身。如果放一个很大的数例如 0FFH，那么要减 255 次才能跳出自减，这个时候消耗的时间就会比较多。再根据乘法原理，嵌套两三层循环后，可以得到比较明显的延时。例如，我套三层：10H*40H*80H=20000H=131072D，一条指令至少 1us，DJNE 是双周期指令，则该 delay 延迟至少 0.13*2=0.26s,每次滚动至少间隔 0.26s，实际可能更长。

(3) 滚动主体：

也就是初值的设定。这个很简单，为 A 赋一个合适的初值就行。需要什么样的排列，就先写二进制下 8 位 0 和 1，再转 16 进制，在开始循环前赋到 A 就行。

※注：在写入大于 9FH 的值时，写两位，例如 MOV A,#AF，会报错，需要写三位即#0AF。

2. 闪烁的实现

闪烁无非是亮暗交替。把 0 送到 P1，隔一段时间再把 1 送到 P1，如此往复循环即可。间隔的时间与滚动中**“能看见”**部分涉及到的延迟相关，需要让人眼能分辨出来，写一个延迟的函数就可以实现。亮转暗再转亮，如果是灯全部亮的话，可以用 **CPL A**，按位取反，就全部变暗了

如果亮暗前后的时间是等间隔的，那就是均匀闪烁，这在呼吸中是分析的基础。

3. 呼吸的实现

(1) 呼吸的过程

从亮到暗再到亮，在模拟电路中好做，因为可以比较平滑地调节电压，但是在单片机数字电路下，输出都是高低电平的固定值，因此，呼吸效果必然是某种叠加在人眼产生的效应。显然，闪烁是基础。分析一下闪烁的过程，它分亮和暗两部分，指令几乎是瞬间的，看做两个点，但是两个点之间的间隔是可以通过延时来调节的。因此，对一次或者十次闪烁分析，如果亮的时间长，暗的时间短，我们可以看到一定的节奏的闪烁。当闪烁的频率很快的时候，我们不会看到明显的“持续的暗”，因为马上被后面的亮起来掩盖了。通过视觉暂留，亮和暗之间的界限被模糊了，这就是我们进行呼吸的基础。

(2) 呼吸变化速率的调节

引入占空比的概念来分析，对于 LED 来说：

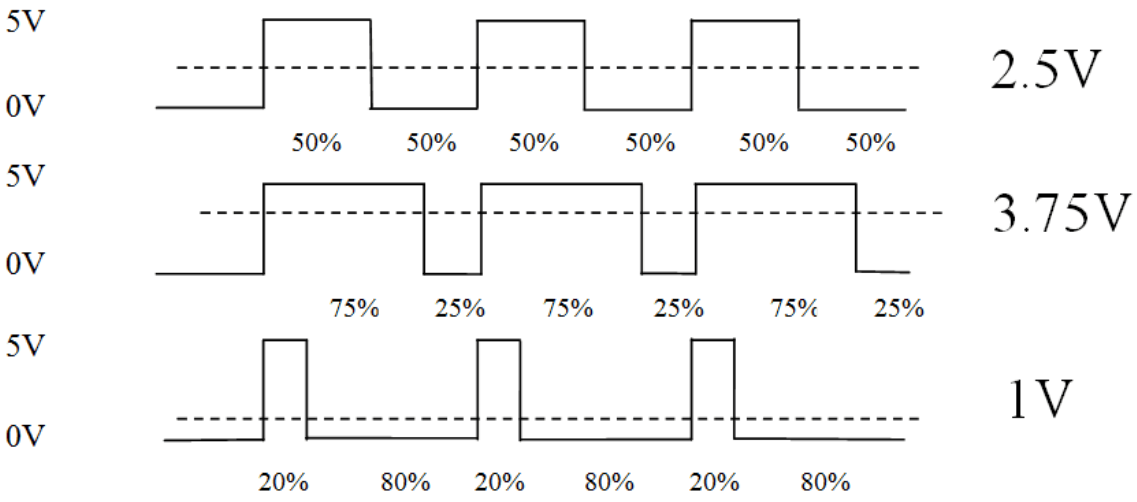
"亮"时间：LED 接通电流，发光。

"暗"时间：LED 关断电流，不发光。

占空比 ＝"亮"时间 /("亮"时间 ＋"暗"时间)

可以用一个寄存器存放控制“暗”时间的变量，慢慢 DEC 和 INC，改变其占空比，整体观感就是变暗和变亮。然后搞个 Delay，控制闪烁就行。

(3) 一些散发联想：PWM 调光，然后让它更直观



这应该就是个 PWM 调光的过程。上课时候讲的我感觉不是很直观。后面联想到之前手机 PWM 调光的事情，手机 OLED 屏幕用手机摄像头调一下快门速度就可以直观地看到频闪，比较明显地能够看到“占空比”。于是我又调了一下参数，用手机分别拍了一下，把快门时间拉到 1/100s 以下就比较明显了，但这个频率人眼看不出来。

手机相机专业模式下，快门拉快，录屏再逐帧截图。左图明显暗，右图明显更亮，占空比也很明显，左图小，右图大，很直观地可以展示出来。