浙江大学

本科实验报告

仪器人机交互接口设计

课程名称：仪器系统设计

姓名：

学院：生物医学工程与仪器科学学院

专业：生物医学工程

学号：

指导老师：周泓

2025 年 5 月 29 日

浙江大学实验报告

专业：生物医学工程
姓名：
学号：
日期：2025 年 5 月 29 日
地点：

课程名称：仪器系统设计　　指导老师：周泓　　成绩：
实验名称：仪器人机交互接口设计　　实验类型：　　同组学生姓名：

1. 如何利用最少的按键，完成闹钟的当前时间与闹钟时间的设置

为了利用最少的按键完成闹钟的当前时间与闹钟时间的设置，我经过深入分析认为最少可以用 4 个按键来实现全部功能。我采用多义键（多功能键）的设计思路，根据我的理解，一个完整的命令通常不是由一次按键操作完成，而是需要按两次以上的键才能完成，且这几个键的操作要遵守一定的顺序，称为按键序列。在我的设计中，每个按键在不同的操作模式或按键序列下可以具有不同的功能。

(1) 理论最少按键数分析

从功能需求分析：

- 需要进入/退出设置模式（需要 1 个操作）
- 需要在当前时间和闹钟时间设置间切换（需要 1 个操作）
- 需要调整数值的增加/减少（需要 2 个操作：增、减）
- 需要在时间段间移动（小时 $\rightarrow$ 分钟，需要 1 个操作）
- 需要开关闹钟（需要 1 个操作）

通过多义键设计，我可以将某些功能合并到同一个键：

- ”设置”键可以承担进入/退出设置模式的功能
- ”模式”键可以在设置模式下切换时间类型，在非设置模式下切换闹钟开关
- ”确定”键可以实现时间段移动功能，并与模式键结合实现闹钟开关
- ”增加/减少”键独立负责数值调整

因此，理论上最少需要 4 个按键即可实现全部功能。

(2) 4 按键配置方案

- **设置键 (SET)**：设计为双义键。
  - 在正常模式下短按：进入设置模式。
  - 在设置模式下短按：退出设置模式。

- **模式键 (MODE)**：设计为多义键。
  - 在正常模式下短按：开启/关闭闹钟功能。
  - 在设置模式下短按：在” 设置当前时间” 和” 设置闹钟时间” 之间切换。
- **增加键 (INC/UP)**：设计为双义键。
  - 在设置模式下短按：增加当前时间段数值（小时或分钟）。
  - 在设置模式下长按：快速连续增加数值（连击功能）。
- **确定/减少键 (OK/DEC)**：设计为多义键。
  - 在设置模式下短按：减少当前时间段数值（小时或分钟）。
  - 在设置模式下长按：快速连续减少数值（连击功能）。
  - 在设置小时完成后双击：移动到分钟设置。
  - 在设置分钟完成后双击：完成当前时间/闹钟时间设置。

(3) 4 按键操作流程

- **设置当前时间：**
  - 按下”SET” 键进入设置模式（系统默认进入当前时间设置）。
  - 如果显示界面显示的是闹钟设置，按下”MODE” 键切换到当前时间设置。
  - 使用”INC”/”OK/DEC” 键调整小时，调整完成后双击”OK/DEC” 键移动到分钟设置。
  - 使用”INC”/”OK/DEC” 键调整分钟，调整完成后双击”OK/DEC” 键完成设置。
  - 按下”SET” 键退出设置模式。
- **设置闹钟时间：**
  - 按下”SET” 键进入设置模式。
  - 按下”MODE” 键切换到闹钟时间设置（显示界面会有”ALM” 等标识）。
  - 使用”INC”/”OK/DEC” 键调整闹钟小时，调整完成后双击”OK/DEC” 键移动到分钟设置。
  - 使用”INC”/”OK/DEC” 键调整闹钟分钟，调整完成后双击”OK/DEC” 键完成设置。
  - 按下”SET” 键退出设置模式。
- **开关闹钟：**
  - 在正常显示模式下，按下”MODE” 键即可切换闹钟的开启/关闭状态（显示屏会显示相应的闹钟图标）。
- **快速调整（连击功能）：**
  - 当需要大幅调整时间时，可以长按”INC” 或”OK/DEC” 键实现连续快速增减。

通过这种设计方式，仅用 4 个按键（SET、MODE、INC、OK/DEC），通过多义键和按键序列的巧妙组合，即可实现闹钟的当前时间与闹钟时间的设置功能。这是在保证功能完整性的前提下能够实现的最少按键数量。

2. 自行设计一个空调遥控器，完成按键的复合功能定义

在设计空调遥控器时，我认为按键的复合功能定义可以极大地节省按键数量并提高用户体验。我设计的按键列表与复合功能定义：

- **电源键 (Power)**：我设计为单义键。
  - 短按：开启/关闭空调。
- **模式键 (Mode)**：我设计为单义键。
  - 短按：切换空调模式（制冷、制热、除湿、送风、自动）。
- **温度调节键 (↑ / ↓)**：我设计为双义键。
  - 短按：每按一次温度升高/降低 1 度。
  - 长按（连击）：温度持续升高/降低，直到松开按键。
- **风速键 (Fan Speed)**：我设计为双义键。
  - 短按：切换风速档位（自动、低、中、高）。
  - 长按：进入/退出静音模式。
- **摆风键 (Swing)**：我设计为双义键。
  - 短按：开启/关闭上下摆风。
  - 长按：开启/关闭左右摆风。
- **定时键 (Timer)**：我设计为多义键。
  - 短按：进入定时设置模式。
  - 再次短按：在定时开、定时关之间切换。
  - 在定时设置模式下，我配合温度调节键实现定时时长调整。
  - 长按：取消定时。
- **睡眠键 (Sleep)**：我设计为双义键。
  - 短按：开启/关闭睡眠模式（自动调节温度和风速）。
  - 长按：开启/关闭节能模式。
- **健康键 (Health)**：我设计为双义键。
  - 短按：开启/关闭健康除菌功能。
  - 长按：开启/关闭自清洁功能。

设计思路：

- 常用功能单按：将最常用的功能（如开关、模式、温度调节、风速）设置为按键的短按功能，方便用户快速操作。

- 次常用功能长按/组合：次常用或进阶功能设置为长按或与其他按键组合的功能，例如长按风速键进入静音模式。
- 模式切换：利用” 定时” 键的多次短按来切换定时开和定时关，减少按键数量。
- 连击：让温度调节键利用连击功能实现快速增减，提升用户体验。
- 直观图标：遥控器上的按键应配合直观的图标，帮助用户理解复合功能。

通过如上设计，可以在有限的按键数量下，实现空调遥控器的所有常用功能，并提供一些便捷的复合操作，提升用户的使用便利性。

3. 简述科学计算器的键盘设计方案

科学计算器的键盘设计方案需要考虑到其复杂的运算功能和大量的符号输入，因此我经过调研结合过往课程 (电设) 经验 +CASIO 用户体验后选择采用矩阵式键盘结合多义键的设计。

(1) 键盘结构：

- 非编码矩阵式键盘或编码矩阵式键盘：根据我的调研分析，由于科学计算器按键数量较多，我认为采用矩阵式键盘以节省 I/O 口线是最合理的选择。
  - 非编码矩阵式键盘：我了解到它通过行线和列线的交叉连接来识别按键，需要软件扫描法或线反转法来识别键值。扫描法逐行扫描查询，线反转法则通过 I/O 口线方向反转来快速获取键值。
  - 编码矩阵式键盘：我发现它内部设有键盘编码器，被按下键的键值由编码器直接给出，同时具有防抖和解决连击的功能，处理速度快。
- 标准数字键和运算符键：在我的设计中，0-9 数字键、小数点、加、减、乘、除、等于等基础运算符应置于易于操作的区域。
- 功能键区：我专门设置函数运算（sin, cos, tan, log, ln, e^x, x^y 等）、括号、常数（π, e）、记忆功能（M+, M-, MR, MC）、方向键、清零（AC）和删除（DEL）等按键。

(2) 多义键设计：

- Shift/2nd Function 键：根据我的研究，这是科学计算器最核心的多义键设计。我通过一个”Shift” 或”2nd Function” 键，可以将每个按键赋予两种或多种功能。
  - 在我的设计中，数字键上方可以对应常用的科学常数或单位转换。
  - 运算键上方可以对应反函数（asin, acos, atan 等）。
  - 其他功能键上方可以对应更高级的运算或模式切换。
- Alpha 键（编程计算器）：对于可编程的科学计算器，我可能还会加入一个”Alpha” 键，用于输入字母字符，以便于变量定义或程序编写。
- 模式键 (Mode)：我用它在不同运算模式之间切换，例如角度单位（DEG/RAD/GRAD）、浮点显示模式（Normal/Sci/Eng）、复数模式等。

(3) 键盘去抖：

- 根据我的了解，由于按键抖动可能导致计算机将一次按键操作误判为多次操作，因此必须采用去抖动方法。

- 软件去抖：我认为这是最常用的方法。首次检测到按键按下信息后，延时 10～20ms，再次判断按键状态，确认按键是否按下，并在按键释放后才认为完成一次操作。
- 硬件去抖：我了解到可以采用 RS 触发器等逻辑电路实现，但通常只用于按键数目较少的场合。科学计算器因按键多，我更倾向于采用软件去抖。

(4) 键值识别与分析：

- 识键：我需要确定是否有键按下。
- 译键：我需要识别是哪一个键被按下并确定相应的键值。
- 键值分析：我根据键值找出相应处理程序的入口并执行。单义键根据键值直接跳转，双义键和多义键需要结合模式键或按键序列来确定最终功能。

我设计的软件流程：

(1) 扫描键盘：我让系统持续扫描键盘，检测是否有键按下。

(2) 去抖动：检测到按键信号后，我设置延时 10-20ms 进行二次确认，消除抖动。

(3) 识别键值：我通过扫描法或线反转法（非编码矩阵）或编码器直接获取键值。

(4) 键值分析：

- 我判断是否存在”Shift” 或其他模式键被按下。
- 根据当前按键和模式键的状态，我查询对应的功能表，确定最终的操作。
- 如果是数字或运算符号，我将其加入输入缓冲区。
- 如果是命令键（如等于号），我执行相应的运算程序。

(5) 等待释放：为了保证单次键入，在处理完按键后，我让程序等待按键释放才进行下一次扫描。

(6) 显示更新：我将运算结果或输入内容显示在 LCD 屏幕上。

通过我的设计，科学计算器能够在有限的物理按键数量下，实现丰富而复杂的运算功能，同时保证操作的准确性和用户体验。

4. 用实例说明，非编码矩阵式键盘扫描法和线反转法进行键值识别的异同点

非编码矩阵式键盘是按键数量较多系统中常用的键盘类型，它通过行线和列线交叉连接来识别按键。我了解到键值识别常用的方法是扫描法和线反转法。

实例：4×4 非编码矩阵式键盘

假设我有一个 4×4 的非编码矩阵式键盘，连接到单片机的 P1 口。P1.0-P1.3 作为列线（输入），P1.4-P1.7 作为行线（输出）。

	Column 0 (P1.0)	Column 1 (P1.1)	Column 2 (P1.2)	Column 3 (P1.3)
Row 0 (P1.4)	Key 0	Key 1	Key 2	Key 3
Row 1 (P1.5)	Key 4	Key 5	Key 6	Key 7
Row 2 (P1.6)	Key 8	Key 9	Key 10	Key 11
Row 3 (P1.7)	Key 12	Key 13	Key 14	Key 15

表 1: 4×4 矩阵式键盘布局

(1) 扫描法 (Scanning Method)

- 原理：根据我的理解，扫描法通过逐行输出低电平（或高电平），然后读取列线状态来判断是否有键按下以及是哪个键被按下。它相当于在行线上进行” 扫描”。
- 步骤：
  (1) 判断是否有键闭合：我将所有行线（P1.4-P1.7）设置为输出低电平，然后读取所有列线（P1.0-P1.3）的状态。如果所有列线都为高电平，则无键闭合；否则有键闭合。
  (2) 消除键抖动影响：检测到有键闭合后，我让软件延时 10～20ms，再次检测是否有键闭合，以确认按键的有效性。
  (3) 确定闭合键的键值：
    – 我从第一行开始，依次将一根行线设置为低电平，其余行线设置为高电平。
    – 例如，我将 P1.4（Row 0）设置为低电平，P1.5-P1.7 设置为高电平。
    – 我读取所有列线（P1.0-P1.3）的状态。如果某一列线变为低电平，则说明该行与该列的交叉点上的键被按下。
    – 例如，如果 P1.0 变为低电平，则 Key 0 被按下。
    – 我重复此过程，直到所有行被扫描完毕或找到被按下的键。
  (4) 计算键号：我通过公式键号 = 行号 × 列数 + 列号计算闭合键的键号。例如，当 PC 口输出扫描字 FBH 时（PC2 为 0），若检测到 PA0 为 0，则闭合键的键值为 2×8+0=10H。
  (5) 等待键释放：为了保证键每闭合一次，处理器仅做一次处理，我让程序等待闭合键释放后才能转去执行相应的键处理程序。
- 优点：我认为实现简单，不需要额外的硬件支持。
- 缺点：我发现扫描速度相对较慢，尤其当被按下键位于键盘的最后一行时，需要多次扫描才能获得键值。

(2) 线反转法 (Line Reversal Method)

- 原理：根据资料，线反转法通过两次读取 I/O 口线状态，并反转 I/O 口线的传输方向来快速定位被按下的键。
- 步骤：
  (1) 第一次读取：

  - 将行线（P1.4-P1.7）设置为输出线并输出低电平（例如 0000B），将列线（P1.0-P1.3）设置为输入线。
  - 读取列线（P1.0-P1.3）的状态。如果无键按下，则输入线为 1111B；若有键按下，则对应的列线变为低电平。我将此数据存入内存单元 N。
  - 例如，如果 Key 1 被按下，P1.1 变为低电平，则读取的列线状态为 1101B。

(2) 反转 I/O 口线方向并第二次读取：

  - 将行线（P1.4-P1.7）设置为输入线，将列线（P1.0-P1.3）设置为输出线并输出低电平（例如 0000B）。
  - 读取行线（P1.4-P1.7）的状态。如果 Key 1 被按下，P1.4（Row 0）变为低电平，则读取的行线状态为 0111B。我将此数据存入内存单元 N+1。

(3) 拼接特征码：将 N+1 单元的数和 N 单元的数据拼接起来，得到按下键的特征码。例如，对于 Key 1，特征码为 0111_1101B (7DH)。

(4) 查表获取顺序码：由于特征码的离散性很大，不便程序处理，我通常会建立一个键特征码与顺序码的转换表，通过查表获取按键的顺序码，以便于后续处理。

- 优点：识别键值的速度较快，只需经过两个步骤即可。
- 缺点：我认为必须借助可编程的通用接口芯片或单片机 I/O 口支持线反转功能。

特性	扫描法	线反转法
相同点	都用于非编码矩阵式键盘的键值识别。	都用于非编码矩阵式键盘的键值识别。
	都需要进行去抖动处理。	都需要进行去抖动处理。
不同点	原理：逐行（或列）扫描，根据输入线电平判断。	原理：两次读写，反转 I/O 方向，拼接特征码。
	速度：相对较慢，取决于被按下键的位置。	速度：较快，通常只需两步。
	硬件：对 I/O 口要求不高，可直接连接。	硬件：要求 I/O 口支持方向反转，或借助可编程接口芯片。
	编程：逻辑相对简单，但循环次数可能较多。	编程：逻辑稍复杂，需查表转换。
	应用：多数采用扫描法识别按键。	应用：当速度要求高时考虑使用。

表 2: 扫描法与线反转法异同点比较

5. 结合实例，给出判别一个按键是否有效的依据

根据学习和实验经验，判别一个按键是否有效的依据，主要是为了解决按键抖动和防止误操作的问题。我了解到按键从最初按下到可靠接触要经过数毫秒的抖动时间，松开时也存在同样问题，抖动时间一般在 5～10ms 之间。抖动可能导致计算机将一次按键操作误判为多次操作。

(1) 首次检测到按键状态变化：

- 当按键从未按下状态（例如高电平）变为按下状态（例如低电平）时，我认为这标志着一次潜在的按键操作的开始。

(2) 延时去抖：

- 在首次检测到按键按下信号后，不立即确认，而是软件延时一段” 去抖时间”（通常为 10～20ms）。设置这个时间应大于按键的机械抖动时间，以确保按键接触稳定。

(3) 再次检测按键状态并确认：

- 延时结束后，再次检测按键的状态。如果此时按键仍保持在按下状态（例如仍为低电平），则我认为按键被有效按下并予以确认。如果状态恢复到未按下，则我认为这是一次无效的抖动，不进行处理。

(4) 等待按键释放：

- 在按键被确认有效按下并执行了相应的处理程序后，让程序持续监测按键的状态，直到按键完全释放（例如从低电平变为高电平）。只有当按键完全释放后，我才认为本次按键操作结束，并准备好接收下一次按键输入。这是为了防止” 连击” 问题，即一次按键操作被计算机响应多次。

实例说明：
假设我有一个连接到单片机 P1.0 口的独立式按键，正常状态下 P1.0 为高电平，按下时为低电平。我设计的有效按键判断流程（软件去抖）如图 1所示：

- 第一次检测：如果用户只是轻微触碰，或按键产生了抖动，P1.0 可能会短暂变为低电平。
- 延时：我设置的 10ms 延时是关键。如果这仅仅是抖动，那么在 10ms 内，按键会恢复到高电平。
- 第二次检测：如果 10ms 后 P1.0 仍然是低电平，我认为这说明按键确实被稳定地按下了，而不是短暂的抖动。此时我才认为按键是有效的。
- 等待释放：即使按键被确认有效，在处理完对应的功能后，我让程序不会立即返回，而是会一直等待直到 P1.0 恢复到高电平（按键松开），才能开始下一次按键检测，从而避免了连击现象。

通过设计的这种软件去抖动和等待释放的机制，可以有效判别按键的有效性，防止由于机械抖动和快速按压导致的误操作或多次响应。

6. 结合实例，给出按键输入异常的处理方法

(1) 按键抖动 (Key Bounce)：物理按键在按下或松开时，触点会发生多次瞬间的闭合与断开，导致电平信号产生抖动。

- 异常表现：一次按键操作被识别为多次按键，例如按一下”1” 却输入了”111”。
- 我的处理方法：
  - 软件去抖：我认为这是最常用且高效的方法。当检测到按键电平变化时，我不立即响应，而是延时 10-20ms（此时间大于按键抖动时间），再次检测电平是否稳定。如果稳定，则我确认按键有效。

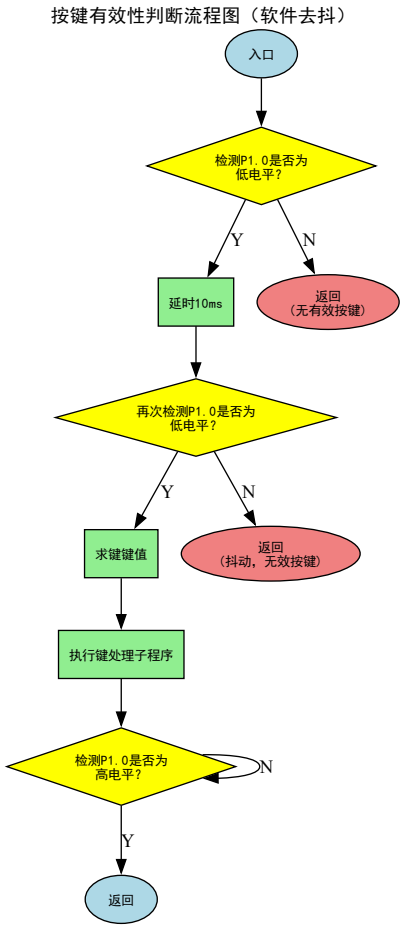


图 1: 按键有效性判断流程图（软件去抖）

  - 硬件去抖：我可以使用施密特触发器、RS 触发器或 RC 滤波电路等硬件电路来消除抖动。适用于按键数量较少的情况，但会增加硬件成本和复杂性。
- 实例：在独立式按键检测程序中，我首次检测到按键按下后，延时 10ms 再读一次按键状态进行确认。

(2) 连击 (Key Repetition / Sticky Key)：用户按住按键不放，如果系统处理不当，可能只响应一次或在长时间内不响应。但我发现在某些特定场景下，连击也可能被合理利用。

- 异常表现：
  - 用户按住按键不放，但系统只响应一次，无法实现连续输入（如游戏中的移动）。
  - 用户希望连击，但系统没有提供连击功能，导致操作不便（如调整音量、亮度）。
- 我的处理方法：
  - 单次键入（防连击）：在按键有效按下并处理后，我让程序持续检测按键状态，直到按键完全释放才允许下一次按键的检测和处理。这是我认为最常见的防连击策略。
  - 允许连击：在需要连击功能的场景（如音量调节、参数递增/递减），在首次按下并确认后，我让程序启动一个定时器。如果按键持续按下超过一定阈值时间（如 200-500ms），则我让其进入连击模式，以一定间隔重复发送键值，直到按键松开。
- 实例：在空调遥控器中，我可以将温度调节键设置为长按后实现温度的连续增加或减少。

(3) 串键 (Key Chattering / Ghosting)：当键盘的按键密度较高时，可能因操作不慎使双键或多键同时动作。

- 异常表现：用户只希望按下一个键，但系统识别出两个或多个键同时按下，导致错误功能或乱码。
- 我的处理方法：
  - " 两键同时按下" 保护：我认为最简单的方法是当只有一个键按下时才读取键盘信息，否则不进行处理。或者当第一个按键未松开时，按第二个按键不产生键值。
  - "N 键同时按下" 技术：我选择不理会所有被按下的键，直到只剩下一个键按下时为止；或者将按键信息存入内部缓冲存储器，再进一步处理。
  - "N 键锁定" 技术：我只处理一个键，任何其他按下又松开的键不产生键值，通常第一个被按下或最后一个松开的键产生键码。我认为这种方法最简单也最常用。
- 实例：在一个输入 PIN 码的设备上，如果同时按下多个数字键，我设计系统拒绝输入，直到只有一个键被识别。

(4) 死键/卡键 (Stuck Key)：按键被物理卡住或电路短路，导致按键信号一直保持按下状态。

- 异常表现：系统持续响应同一个键的功能，可能导致无限循环、参数不断变化、或界面锁定。
- 我的处理方法：
  - 超时检测：我在按键处理流程中加入超时机制。如果一个按键被检测到持续按下超过预设的极长时间（例如几秒），我让系统认为该键卡死，并发出警告或进入某种保护模式（如忽略该键的所有后续输入）。
  - 看门狗定时器：在嵌入式系统中，我可以让看门狗定时器监测程序是否" 跑飞" 或陷入死循环。如果按键卡死导致程序进入无限循环，我设置看门狗定时器强制系统复位。

  - 用户干预/错误提示：我提供明确的错误提示，指导用户检查按键或重新启动设备。
- 实例：在一个仪器控制面板上，如果” 开始” 键被卡死，系统会不断尝试启动程序。通过我设计的超时检测，程序可以识别到” 开始” 键异常，暂停操作并提示用户” 按键故障”。

通过上述各种异常处理方法，可以大大提高仪器系统人机交互的稳定性和可靠性，避免因按键问题导致的用户操作困扰和系统故障。

7. 分析比较采用段码式 LED 与点阵式 LED 显示图案的不同

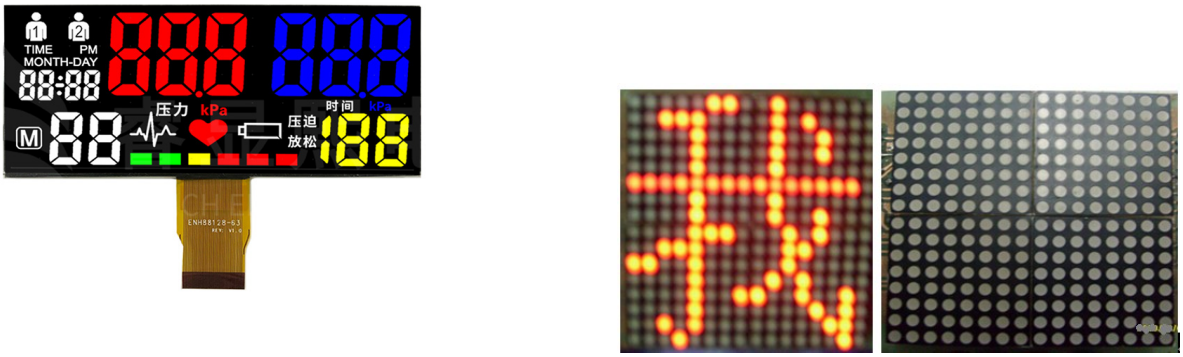
(1) 段码式 LED 显示器 (Segment LED Display)

- 工作原理：由数个 LED 组成一个阵列，并封装于一个标准尺寸的管壳内。最常见的是由 7 个 LED（加上一个小数点为 8 个 LED）构成的” 日” 字形 7 段数码显示器。每个 LED 对应一个段，通过控制不同段的亮灭来显示字符。
- 显示图案特点：
  - 字符类型受限：主要用于显示数字（0-9）和少数简单的英文字母（如 A, B, C, D, E, F, H, P, U 等）以及一些基本符号（如小数点、负号）。
  - 字型固定：字符的形状是固定的，由 LED 段的排列决定，无法自由定制字形。因此，显示的数字和符号比较简单，显示更多种类且字型逼真的字符则比较困难。
  - 不适合复杂图形：无法显示复杂的汉字、图形或动画效果，因为它不具备像素级别的独立控制能力。
  - 显示效果：亮度高，显示稳定可靠。
  - 驱动方式：可以采用静态显示或动态显示。
    * 静态显示：相应段的 LED 恒定导通或截止，亮度高，控制程序简单，功耗大。
    * 动态显示：当显示位数较多时，采用分时轮流工作，利用视觉暂留效应实现多位显示，节省 I/O 口线。
- 典型应用：数字时钟、计算器、仪器仪表上的数值显示、简单的状态指示等。

(2) 点阵式 LED 显示器 (Dot Matrix LED Display)

- 工作原理：以点阵式进行显示，由多个发光二极管按照矩阵形式排列，例如 5×7、7×9 点阵等，最常用的是 5×7 点阵，由 35 个发光二极管组成 5 列 ×7 行的矩阵。每个发光二极管都是一个独立的像素点，通过控制每个点的亮灭来形成图案。
- 显示图案特点：
  - 字符类型丰富：可以显示几乎所有英文字母、数字、符号，以及复杂图形和汉字。字型可以根据需要自由定制。
  - 字型逼真可调：由于是像素级控制，显示的符号比较逼真，更易识别。可以通过改变点阵中点的亮灭组合来显示不同的字模（如字母 A 的点阵图）。
  - 支持复杂图形和动画：多个点阵式 LED 显示器可以组成大屏幕显示屏，用来显示汉字、图形、表格，并且能产生各种动画效果。

  - 接口复杂：相对于段码式，点阵式 LED 显示器的接口电路及控制程序比较复杂。
  - 驱动方式：常采用动态扫描方式显示，以节省 I/O 口线。
- 典型应用：LED 广告屏、滚动字幕、电子显示屏、车站和机场的信息显示牌、汉字显示设备、游戏机屏幕等。



(a) 段码式 LED 显示器　　(b) 点阵式 LED 显示器

图 2: 段码式 LED 与点阵式 LED 显示器对比

特性	段码式 LED 显示器	点阵式 LED 显示器
结构	7 段、8 段、12 段等固定段的组合	5×7、7×9 等矩阵排列的 LED 点
显示内容	数字、少量简单字母和符号	任意字符（中英文、数字）、复杂图形、动画
字型	固定、简单	可定制、逼真
灵活性	低	高
接口复杂性	简单	复杂
应用场景	强调数值显示，简单状态指示	强调信息量、视觉效果、复杂图形和动态显示

表 3: 段码式 LED 与点阵式 LED 显示器比较

总而言之，段码式 LED 显示器适用于仅需显示数字或少数简单字符的场合，其优点是结构简单、成本低廉。而点阵式 LED 显示器则提供了更高的灵活性和更丰富的显示能力，可以显示复杂的字符、图形和动画，适用于需要大量信息展示和更好视觉效果的场景，但其接口电路和控制程序相对复杂。

8. 查阅资料，描述室外 LED 显示屏的设置与控制方式

室外 LED 显示屏（LED Display, LED Screen）又称电子显示屏或飘字屏幕，通过红色、蓝色、白色、绿色 LED 灯的亮灭来显示文字、图片、动画、视频，内容可以根据不同场合的需要做出不同的调节。

图 3: 室外 LED 显示屏

(1) 物理设置（硬件构成）

传统的 LED 显示屏通常由显示模块、控制系统及电源系统组成。

- 显示模块 (LED Display Modules)：
  - LED 单元板：室外 LED 显示屏的基本组成单元。每一块单元板由若干个 LED 灯珠组成，按照点阵排列（如 P10，表示像素点间距为 10mm）。单元板的像素点间距（P 值）决定了显示屏的清晰度。室外屏通常采用较大的像素间距，如 P4、P5、P6、P8、P10 等，以适应远距离观看的需求。
  - 箱体/模组：多个 LED 单元板组装成一个箱体或模组，这些箱体具有防尘、防水（达到 IP65 或更高等级）、防潮、防腐蚀、散热等特性，以适应恶劣的室外环境。箱体之间可拼接，形成更大尺寸的显示屏。
  - 防护等级：鉴于室外环境的严酷性，显示屏的防护等级至关重要，需要具备良好的防晒、防雨、防雷、防风、防震能力。
- 电源系统：
  - 开关电源：为 LED 单元板提供稳定的直流供电。由于 LED 灯珠数量庞大，对电源的稳定性和功率要求很高。
  - 配电箱：包含断路器、漏电保护器、防雷器等，确保供电安全和稳定。
- 控制系统：
  - 发送卡 (Sending Card)：通常安装在控制计算机内部，负责将计算机的显示数据转换为 LED 显示屏可识别的信号格式。

  - 接收卡 (Receiving Card)：安装在每个 LED 箱体内部或附近，接收发送卡传来的信号，并将其分配给对应的 LED 单元板，控制 LED 灯珠的亮灭。
  - 视频处理器 (Video Processor)：对于播放视频或高分辨率图像的显示屏，视频处理器是必需的。它负责对输入信号（如 HDMI, DVI, VGA 等）进行处理、缩放、拼接，使其适应显示屏的实际分辨率和显示效果。
  - 光纤传输设备：当控制室与显示屏距离较远时，为保证信号传输的稳定性和带宽，常采用光纤进行数据传输。
- 支撑结构：钢结构框架或立柱，用于固定和支撑显示屏，并抵抗风力等外部载荷。

(2) 控制方式

室外 LED 显示屏的控制方式多样，根据应用需求和数据传输模式可分为以下几种：

- 同步控制系统：
  - 原理：这种控制方式类似于计算机显示器，显示屏实时同步显示计算机屏幕上的内容。发送卡将计算机屏幕信号实时传输到接收卡，接收卡再控制 LED 显示。
  - 特点：
    * 实时性：实时同步显示，无延时。
    * 内容丰富：可以播放视频、图片、动画、文本等各种多媒体内容。
    * 操作方便：通过计算机软件直接控制，所见即所得。
    * 硬件要求：需要一台高性能的计算机（主控电脑）和相应的发送卡、接收卡、视频处理器。
  - 应用：大型广告屏、体育场馆的直播屏幕、舞台背景屏等。
- 异步控制系统：
  - 原理：显示内容预先通过计算机编辑好，然后通过串口（RS232/RS485）、网口（RJ45）、U 盘、Wi-Fi 等方式发送到 LED 显示屏内置的控制器中，由控制器独立存储和播放。
  - 特点：
    * 脱机运行：内容传输完成后，计算机可以断开，显示屏仍可独立运行。
    * 成本较低：不需要高性能的计算机实时连接。
    * 内容限制：主要用于播放文本、图片、简单动画等，不适合实时视频流。
    * 传输方式：支持多种传输方式，如 U 盘更新、GPRS/4G 无线传输等，方便远程管理。
  - 应用：门店广告牌、信息发布屏、公交车报站屏等小型或信息更新不频繁的场合。
- 集群控制系统：
  - 原理：通过网络（有线或无线，如 3G/4G/5G）实现多块 LED 显示屏的集中管理和控制。在一个中心控制室可以对分布在不同地点的多块显示屏进行内容更新、状态监控等操作。
  - 特点：
    * 集中管理：大大提高管理效率，降低维护成本。

    - 远程控制：无需现场操作，方便快捷。
    - 内容分发：能够批量分发内容到多块屏幕。
  - 应用：连锁店广告网络、城市信息发布系统、交通诱导屏等。

- 手机 APP 控制：
  - 原理：通过 Wi-Fi 或蓝牙连接，用户可以使用手机 APP 直接编辑和发送文本、图片到小型 LED 显示屏。
  - 特点：操作便捷，适用于个人或小型商业用途。
  - 应用：车载 LED 屏、小型商店招牌等。

室外 LED 显示屏的设置与控制方式，是集成了电子、光学、计算机、通信等多个领域技术于一体的复杂系统工程，其目的在于在户外环境中实现高效、可靠、多样的信息传播。

9. 比较采用 MCU 内置式段码式 LCD 显示接口与外置式段码式 LCD 显示接口的异同点

段码式 LCD 显示器是常见的一种字符显示屏，其驱动方式可以采用 MCU（微控制器）内置的驱动接口，也可以采用外置的专用驱动芯片。两者在实现方式、成本、设计灵活性等方面存在异同。

(1) MCU 内置式段码式 LCD 显示接口

- 工作原理：根据我的了解，一些 MCU（如 MSP430X4xx 系列单片机）内部集成了段码式 LCD 驱动模块。这些 MCU 能够直接产生 LCD 所需的交流驱动信号和扫描时序，通过其 I/O 口线连接到 LCD 的段选端和公共电极（COM）端。
- 特点：
  - 优点：
    - 降低成本：我认为省去了外部专用 LCD 驱动芯片，从而降低了整个系统的硬件成本和 PCB 面积。
    - 简化设计：我发现无需额外驱动芯片的选型和布线，简化了硬件设计。
    - 功耗优化：我观察到内置驱动器通常能更好地与 MCU 的电源管理集成，实现更低的系统功耗。
    - 提高集成度：减少外部元件数量，我认为能提高系统的集成度和可靠性。
  - 缺点：
    - MCU 选择受限：我必须选用带有 LCD 驱动功能的 MCU，这限制了 MCU 的选择范围，可能无法满足其他特定的性能或外设需求。
    - 驱动能力限制：我发现内置驱动器的段/COM 线数量、驱动能力（最大段数和驱动电流）通常是固定的，可能无法驱动大型或多位数的 LCD 显示屏。
    - 灵活性较低：我认为内置驱动器的参数和驱动模式相对固定，可配置的灵活性不如外置专用芯片。
    - 占用 MCU 资源：即使是内置驱动，我发现也需要 MCU 的相应 I/O 口和内部定时器/模块资源来支持 LCD 驱动功能。
- 典型应用：我认为它适用于低成本、低功耗、显示位数和段数不多的应用，如电子手表、计算器、小型仪表、水电气表等。

(2) 外置式段码式 LCD 显示接口

- 工作原理：根据我的学习，通过 MCU 与一个独立的专用 LCD 驱动芯片（如合泰 HT1621 系列、NXP PCF8566、PCF85132 等）连接，由专用驱动芯片负责产生 LCD 所需的交流驱动信号和扫描时序。MCU 通过 I/O 口线或 I2C/SPI 等总线与驱动芯片通信，发送显示数据和控制命令。
- 特点：
  - 优点：
    - MCU 选择灵活：我可以根据项目需求自由选择 MCU，不受 LCD 驱动功能的限制。
    - 驱动能力强：我发现外置驱动芯片通常具有更强大的驱动能力，支持更多的段/COM 线，可以驱动更大、更复杂的段码式 LCD。
    - 设计灵活性高：我了解到驱动芯片通常提供丰富的配置选项，可以灵活调整驱动电压、频率、偏置比等参数，以适应不同 LCD 的要求。
    - 减轻 MCU 负担：LCD 驱动的时序生成和复杂控制由专用芯片完成，我认为能减轻 MCU 的软件负担和实时性要求。
  - 缺点：
    - 增加成本：我认为引入了额外的硬件成本（驱动芯片本身）和 PCB 面积。
    - 增加设计复杂性：我需要选型合适的驱动芯片，并进行额外的布线和软件驱动开发（通过 I/O 或总线接口通信）。
    - 功耗可能更高：我发现额外的芯片可能会增加整体系统的功耗。
- 典型应用：我认为它适用于显示位数和段数较多、对 LCD 驱动性能要求较高、或 MCU 本身不带 LCD 驱动模块的应用，如中高端仪表、医疗设备、工控设备等。

特性	**MCU 内置式段码式 LCD 显示接口**	**外置式段码式 LCD 显示接口**
驱动器位置	MCU 内部集成	独立专用芯片
硬件成本	低	相对高
PCB 面积	小	相对大
MCU 选择	受限（需带 LCD 驱动功能）	灵活（可根据其他需求自由选择）
驱动能力	有限（段/COM 线数量、电流）	强（支持更多段、位数）
设计灵活性	较低（参数固定）	较高（可配置参数多）
开发难度	软件配置相对简单（MCU 内部寄存器）	软件开发相对复杂（需通信协议）
系统功耗	通常较低	可能稍高（额外芯片功耗）
集成度	高	较低（外部元件多）

表 4: MCU 内置式与外置式段码式 LCD 显示接口比较

- MCU 内置式接口我认为适用于成本敏感、功耗要求低、显示内容相对简单且位数不多的场合，其优点是集成度高、简化设计。

- 外置式接口我认为适用于需要驱动更大型或更复杂 LCD、对 MCU 选型有特殊要求，或需要更高驱动灵活性的场合，其优点是驱动能力强、MCU 选择灵活。

在实际应用中，我会根据项目的具体需求（成本、功耗、显示规模、MCU 资源等）来选择合适的段码式 LCD 显示接口方案。

10. 试安装一个字模软件，获取仿宋体”仪器系统设计”的显示字库

安装 PCtoLCD2002 软件，选择” 字符模式”。然后在字体选项中选择” 仿宋”，字宽字高等比放大，得到如图 4所示结果。

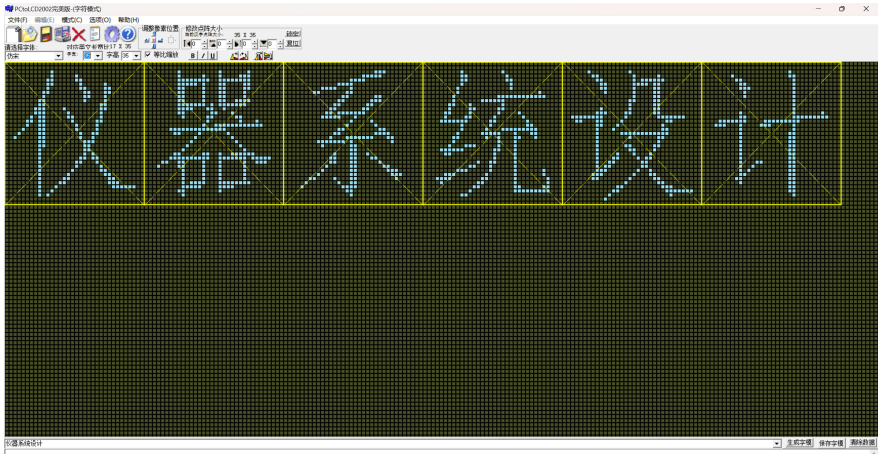


图 4: PCtoLCD2002 软件获取仿宋体” 仪器系统设计” 字库

11. 设计一款采用彩色 LCD 显示的玩具球，其显示界面颜色可根据环境温度和光照条件自动调节，简述初步设计方案

根据我的分析和设计思路，设计一款采用彩色 LCD 显示的玩具球，其显示界面颜色可根据环境温度和光照条件自动调节，需要结合传感器技术、微控制器、彩色 LCD 驱动以及软件算法。

我的初步设计方案：

(1) 核心硬件选型：

- 微控制器 (MCU)：我选择带有彩色 LCD 驱动接口的 MCU，例如带有 TFT 控制器（如 ARM9 或更高平台）的 MCU，可以直接挂接 TFT 屏。如果我选择低端 MCU，则需要外加 TFT 控制模块。我需要考虑 MCU 的运算能力（处理传感器数据和图像数据）、存储空间（存储字体、图片和算法）、以及 I/O 口数量。
- 彩色 LCD 显示屏：我选择适合玩具球尺寸的彩色 LCD 屏幕，如 2.8 英寸、3.5 英寸等。真彩色（24 位色，1677 万多色）的 LCD 能提供丰富的颜色表现力。分辨率越高，显示越清晰。TFT 屏通常不带控制器，需要 MCU 或外部控制器驱动。
- 温度传感器：我选用高精度、响应快的温度传感器，如 DS18B20（数字温度传感器，单总线接口）或 NTC 热敏电阻（需 ADC 转换）。

- 光照传感器：我选用光敏电阻（需 ADC 转换）或数字环境光传感器（如 BH1750，I2C 接口）。
- 电源模块：我设计提供 MCU、LCD 和传感器所需的稳定电源，考虑电池供电（如锂电池）和充电管理。
- 外壳：我设计透明或半透明的球形外壳，能够清晰地看到 LCD 显示，并能保护内部电子元件。

(2) 传感器数据采集与处理：

- 温度数据：我让 MCU 通过相应的接口（如单总线或 ADC）周期性地读取温度传感器数据。
- 光照数据：我让 MCU 通过相应的接口（如 ADC 或 I2C）周期性地读取光照传感器数据。
- 数据平滑/滤波：我对采集到的传感器数据进行简单的滤波处理（如滑动平均），以消除环境噪声和传感器抖动，保证数据稳定性。

(3) 颜色调节算法设计：

- 定义颜色范围：我根据温度和光照变化，预设一系列颜色方案或颜色渐变规则。
  - 温度-颜色映射：
    - 低温（如低于 10°C）：我让它显示冷色调，如蓝色、青色。
    - 中温（如 10°C-25°C）：我让它显示中性色调，如绿色、黄色。
    - 高温（如高于 25°C）：我让它显示暖色调，如红色、橙色。
    - 我可以设计一个温度-RGB 值渐变函数，使颜色平滑过渡。
  - 光照-颜色映射：
    - 强光照（白天）：我让它显示高对比度、高亮度的颜色，如白色背景、黑色文字，或鲜艳的颜色以确保可见性。
    - 弱光照（夜晚/昏暗环境）：我让它显示低亮度、柔和的颜色，如深色背景、浅色文字，避免刺眼，例如黑色：RGB (0,0,0)，白色：RGB (255,255,255)。
    - 我也可以根据光照强度调整 LCD 的背光亮度，以节省功耗并在不同光照下保持视觉舒适度。
- 颜色叠加/混合：我可以将温度和光照的颜色调节算法结合起来，例如：
  - 最终颜色 = (温度颜色 × 温度权重) + (光照颜色 × 光照权重)
  - 或者我根据环境亮度选择不同的温度-颜色方案。
- 显示内容：除了颜色变化，我让 LCD 还可以显示时间、动画表情、简单的文字提示（如” 冷”、” 热”）、或根据温度光照条件变化的图案。

(4) LCD 驱动与显示更新：

- LCD 初始化：我按照所选 LCD 模组的规格书，初始化 LCD 控制器，包括接口模式、时序、分辨率等。
- 图形库/显示驱动：我让 MCU 集成 LCD 的图形库或显示驱动程序，以便在 LCD 上绘制像素、线条、文本、图片等。
- 周期性更新：我让 MCU 周期性地执行：

a. 读取传感器数据。

b. 应用颜色调节算法计算新的显示颜色。

c. 更新 LCD 的背景色、文本颜色或图形颜色。

d. 刷新 LCD 显示。

(5) 我设计的软件流程如图 5所示：

通过以上方案，可以实现一个能够根据环境温度和光照条件自动调节显示界面颜色的彩色 LCD 玩具球。

12. 查阅资料，深入理解触摸屏坐标校准的必要性和方法

根据我查阅的资料和学习，触摸屏是一种新型的输入设备（苹果最初的创新），它是目前最简单、方便、自然的一种人机交互方式。使用者不必接受专业训练，仅需以手指触摸计算机显示屏上的图符或文字就能实现对主机操作，大大简化了计算机的操作模式。

(1) 触摸屏坐标校准的必要性

根据我的理解，触摸屏的坐标校准是指建立触摸点在触摸屏上的物理坐标与显示屏上像素坐标之间的映射关系。我认为它之所以必要，主要有以下几个原因：

- 物理误差：
  - 制造公差：我了解到触摸屏在生产过程中，电阻膜或电容膜的均匀性、电极的直线度、层间间距等都可能存在微小的制造公差，导致触摸屏的电阻分布或电容分布不完全线性均匀。
  - 安装误差：我观察到触摸屏在安装到显示设备上时，很难做到与显示屏的像素阵列完全对齐，可能存在平移、旋转、缩放等误差。
  - 接触电阻：我发现电阻式触摸屏的上下两层导电层在接触瞬间会产生接触电阻，这会影响测量电压的精度。
  - 外部干扰：我了解到电磁干扰、温度变化、湿度等环境因素都可能对触摸屏的电压测量或电容感应产生影响，导致测量误差。
- 非线性与畸变：
  - 电阻屏的非线性：我发现尤其在屏幕边缘，电阻式触摸屏的测量结果可能出现非线性畸变，导致边缘触摸点的精度下降。
  - 电容屏的边沿效应：我了解到电容屏在边缘区域的电场分布可能不够均匀，导致触控不准确。
- 用户习惯：
  - 我观察到不同用户在触摸屏幕时，手指的接触面积、按压力度、触摸角度可能存在差异，尤其是在电阻屏上，这些差异可能会影响触摸点的位置识别。
- 不同分辨率显示屏：
  - 我认为如果触摸屏被用于不同分辨率的显示屏，则需要进行校准以适应新的显示尺寸和像素密度。

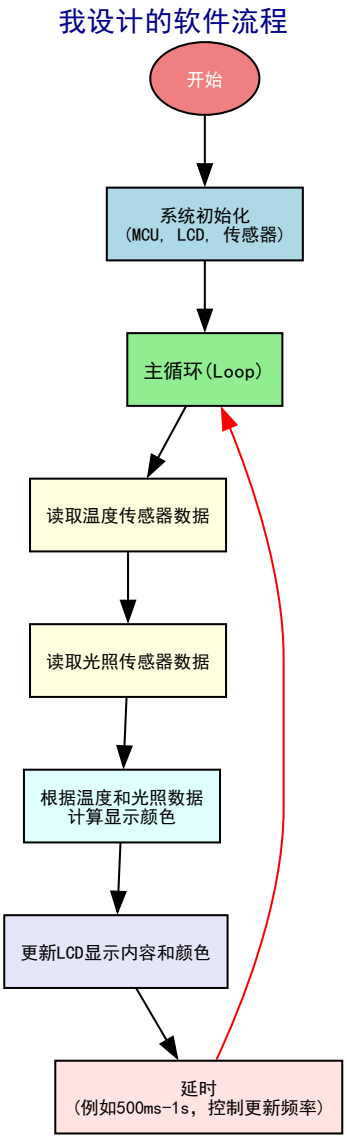


图 5: 彩色 LCD 玩具球软件流程图

- 系统升级或更换：
  - 我了解到当设备的固件升级、操作系统更换或触摸屏/显示屏部件被更换时，原有的校准数据可能不再适用，需要重新校准。

综上，触摸屏在物理上是一套独立的坐标定位系统，每次触摸的数据通过校准转为屏幕上的坐标。如果不进行校准，触摸点在显示屏上的位置可能会与用户的实际触摸位置不符，导致” 点不准” 的问题，严重影响用户体验和设备的可用性。

(2) 触摸屏坐标校准的方法

根据我的学习，触摸屏校准通常通过软件实现，其核心思想是建立一个从触摸屏原始坐标（ADC 值）到显示屏像素坐标的转换模型（通常是线性或非线性变换）。我了解到最常用的方法是” 多点校准法” 或” 三点/四点校准法”。

(1) 在显示屏上依次显示几个已知的校准点（通常是屏幕的角落或中心点）。

(2) 提示用户依次触摸这些校准点。

(3) 让系统记录用户触摸这些点时触摸屏返回的原始坐标（例如，对于电阻屏，是 X 和 Y 方向的 ADC 值）。

(4) 根据这些” 已知显示点” 和” 实际触摸原始点” 的数据对，计算出转换矩阵或转换参数。

(5) 让后续所有触摸屏返回的原始坐标，都将通过这个转换矩阵（或参数）映射到显示屏的像素坐标上。

我了解的常见校准方法则包括：

(1) 三点校准法：

- 原理：我了解到假设触摸屏的映射关系是线性变换（即只考虑平移、旋转、缩放，不考虑畸变），只需要三个非共线的点即可确定一个二维平面上的线性变换矩阵。
- 步骤：
  - 我在屏幕上依次显示三个校准点（例如，左上角、右上角、左下角）。
  - 用户依次触摸这三个点,我让系统记录下对应的三组原始触摸坐标 $(x_1, y_1), (x_2, y_2), (x_3, y_3)$。
  - 我已知这三个触摸点对应的显示屏目标坐标 $(X_1, Y_1), (X_2, Y_2), (X_3, Y_3)$。
  - 我通过解线性方程组或矩阵运算，计算出一个 3×3 的转换矩阵，该矩阵包含缩放、旋转、平移的参数。
  - $X = A \times x + B \times y + C$
  - $Y = D \times x + E \times y + F$
  - 其中 A, B, C, D, E, F 是我需要通过校准计算出的参数。
- 优点：我认为简单高效，计算量小，适用于对精度要求不高且线性度较好的触摸屏。
- 缺点：我发现无法补偿触摸屏的非线性畸变，在屏幕边缘可能存在较大误差。

(2) 四点校准法：

- 原理：我了解到类似于三点校准，但通过四个点的校准可以更好地拟合非线性畸变，提供更高的精度。通常选择屏幕的四个角点。
- 步骤：我发现与三点校准类似，但记录四组数据对。可以通过最小二乘法或其他算法来计算转换参数，以更好地拟合这四个点。
- 优点：我认为比三点校准精度更高，尤其是在边缘区域。
- 缺点：增加了计算量，但对于现代 MCU 来说我认为不是问题。

(3) 多点校准法（更高级）：

- 原理：我了解到在屏幕上显示更多校准点（如 5 点、9 点甚至更多），通过更复杂的算法（如多项式拟合、Bézier 曲线拟合等）来建立映射关系，以补偿更严重的非线性畸变。
- 步骤：我记录更多的数据对，并使用更复杂的数学模型进行拟合。
- 优点：我认为精度最高，能有效补偿各种非线性畸变。
- 缺点：我发现计算量最大，实现最复杂。

我认为触摸屏坐标校准的必要性源于物理制造公差、安装误差、环境因素和非线性特性等导致的触摸点与显示点不一致的问题。通过三点、四点或多点校准等方法，可以建立触摸屏原始坐标与显示屏像素坐标之间的精确映射关系，从而确保用户触摸的准确性，提供良好的用户交互体验。我认为校准是任何触摸屏设备正常工作的关键步骤。