浙江大学

本科实验报告

仪器存储器接口设计

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 6 月 3 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 6 月 3 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：仪器存储器接口设计　实验类型：　同组学生姓名：

1. 在仪器系统设计过程中，如何估算所属的程序存储器和数据存储器空间大小？

在仪器系统设计过程中，估算程序存储器和数据存储器空间大小是一个关键步骤，需要综合考虑多个因素。以《电子系统设计与实践》课程中的”记忆示波器系统设计”实验为例，该实验要求设计一个能够对模拟信号进行采样、存储和回放的系统，具有实时显示、回放显示、测量等多种工作模式。

实验系统主要技术指标：使用 6264 芯片（8KB SRAM）作为波形数据存储器；ADC 采样频率为 2kHz；波形存储时长为 4 秒；支持多种工作模式（实时显示、回放显示、测量模式）；信号发生器功能可输出正弦波、三角波、方波、锯齿波。

我们可以基于此实验，来分析存储器空间估算的具体方法：

(1) **程序存储器估算：**

- **代码复杂性：**估算应用程序的代码量，包括操作系统、驱动程序、应用逻辑以及任何第三方库。对于记忆示波器系统，需要包括 ADC 驱动、DAC 驱动、按键处理、模式切换、波形生成算法、数码管显示驱动等功能模块。
- **编译器和优化：**不同的编译器和优化级别会影响最终生成的可执行代码大小。在项目初期，可以基于类似项目的经验或通过初步的代码编译进行估计。
- **固件更新需求：**如果系统支持在线固件更新，需要预留额外的空间用于存储新固件，或者采用双 Bank 存储设计。
- **引导代码：**如果系统采用引导加载程序（Bootloader），需要为其预留独立的空间。
- **错误处理和日志：**考虑是否需要在程序存储器中存储错误代码、日志信息或诊断程序。
- **冗余和备份：**对于关键系统，可能需要存储程序的多个版本以提供冗余或回滚功能。

(2) **数据存储器估算：**

具体而言，

波形数据存储需求计算：采样点数计算为：采样频率 × 存储时长 = 2kHz × 4s = 8000 个采样点。ADC 采用 8 位，每个采样点占用 1 字节。因此所需存储容量为：8000 点 × 1 字节/点 = 8000 字节　7.8KB。6264 芯片容量为 8KB（8192 字节），能够满足 7.8KB 的存储需求，设计合理。

其他数据存储需求：

- **临时数据：**估算程序运行时所需的 RAM 空间，包括变量、堆栈、队列、缓冲区等。对于记忆示波器系统，需要考虑 ADC 采样缓冲区、DAC 输出缓冲区、按键状态变量、模式标志位、数码管显示缓冲区等。这部分数据在断电后会丢失。
- **配置参数：**系统运行所需的配置参数、校准数据等，如 ADC 校准参数、DAC 校准参数、各种工作模式的配置信息，这些通常需要非易失性存储器来保存，如 EEPROM 或 Flash 的一部分。
- **波形发生器数据：**正弦波、三角波、方波、锯齿波的波形数据表，可以预先计算并存储在程序存储器或数据存储器中。
- **测量结果缓存：**频率测量、幅值测量的结果需要临时存储，以供数码管显示使用。
- **通信缓冲区：**如果系统支持与上位机通信，需要预留通信接口的缓冲区空间。

存储器容量余量设计：

虽然理论计算显示 6264 的 8KB 容量刚好满足 4 秒波形存储需求，但在实际设计中需要考虑数据对齐（某些处理器要求数据按特定边界对齐，可能造成存储空间的浪费）、缓冲区管理（实现环形缓冲区或双缓冲机制时需要额外的存储空间）、扩展功能（如果后续需要增加存储时长或提高采样精度，当前设计可能需要升级）等因素。

(3) **综合考虑：**

- **微控制器选择：**微控制器自带的内部存储器（RAM 和 Flash）容量是首要考虑因素。如果内部存储器不足，则需要考虑外部存储器。对于记忆示波器系统，由于需要大容量的波形数据存储，选择了外部 6264 SRAM 芯片。
- **成本和空间：**存储器的尺寸和成本是重要的设计因素。通常会选择能满足应用要求的最小存储器容量的微控制器以降低成本。6264 芯片在满足性能要求的同时具有较好的成本效益。
- **易失性与非易失性：**根据数据在断电后是否需要保留来选择易失性存储器（如 RAM）或非易失性存储器（如 ROM、EEPROM、Flash、FRAM）。记忆示波器的波形数据在断电后可以丢失，因此选择了 SRAM。
- **读写速度和耐久性：**高速应用可能需要 SRAM，而频繁擦写的数据需要考虑存储器的擦写寿命。SRAM 的高速读写特性满足了 2kHz 采样频率的实时性要求。
- **未来扩展性：**考虑到未来功能增加或代码升级的可能性，在估算时应预留一定的余量，以避免后期存储空间不足导致更换微控制器。如果需要扩展到更长的存储时间或更高的采样率，可能需要考虑更大容量的存储器。

2. 在仪器系统设计中，一旦发现所选择的存储器空间不够，要如何进行升级？

(1) **优化代码和数据：**审查并优化现有代码，移除不必要的函数、变量和冗余代码，使用更高效的算法和数据结构来减少代码大小；调整编译器的优化等级，以生成更紧凑的代码；对存储的数据进行压缩，以减少所需的存储空间；重新评估系统功能，移除不必要或次要的功能，以降低对存储空间的需求。

(2) **更换更大容量的内部存储器微控制器：**如果当前微控制器存在更大容量的兼容型号（引脚兼容、功能兼容），这是最直接且通常成本效益较高的方式。然而，这可能需要重新验证硬件设计和软件移植。

(3) **增加外部存储器：**

- **外部串行存储器：**对于较小的应用系统，当微控制器没有外部地址总线但需要额外数据存储时，外部串行存储器件（如串行 EEPROM、串行 Flash）是最佳选择，它们通常是非易失性的。EEPROM 接口一般以 I^2C 总线接口居多。
- **外部并行存储器：**对于需要高速数据存储的应用，通常选择并行 SRAM。一些设计也会将 Flash 器件用作程序存储器，并保留一个扇区作为数据存储区，以降低成本和空间。
- **混合使用：**内部存储器可以存储不常改变的代码，而外部存储器可以用于存储更新频繁的代码和数据。
- **引导存储器：**如果微控制器没有内部程序存储器，并行的非易失性存储器件通常是正确的选择。对于高速应用，可以使用外部非易失性串行存储器件来引导微控制器，并将主代码存储在内部或外部高速 SRAM 中。

(4) **采用更高密度的存储器芯片：**选择相同类型但存储密度更高的存储器芯片，例如将 1MB 的 Flash 替换为 2MB 的 Flash。这通常需要检查新芯片的引脚兼容性、电气特性和时序要求。

(5) **使用更高效的存储器类型：**Flash 存储器（闪存）是一种不挥发性内存，能够长时间保持数据，其存储特性相当于硬盘，是各类仪器系统的存储介质基础。NOR Flash 适用于存储少量代码，NAND Flash 适用于高存储密度数据。Flash 相比 ROM 更易于在线升级固件。FRAM（铁电存储器）同时拥有 RAM 和非易失性存储器的特性，具有更高的读写次数和更快的读写速度，在可多次编程应用中比 EEPROM 性能更优越。

(6) **重新评估系统架构：**在极端情况下，可能需要重新设计系统架构，例如引入外部处理器或存储管理单元来扩展存储能力。

3. 在 EEPROM 操作过程中，发现读出的数据与刚写入的数据不一致，要如何处理？

(1) **写操作后等待时间不足（快速操作错误）：**即快速连续地进行读写操作，可能导致写入数据出错。虽然时序波形和硬件连接看起来正常，但读出的数据与写入的不同。主要原因可能在于，EEPROM 内部写入操作需要一定的时间来完成对存储单元的实际编程。如果在写入操作尚未完成时就进行下一次读写操作，可能会导致数据损坏或读取到旧数据。因此，解决方法是在每次访问 EEPROM 前或每次写操作后，强制等待一段大于 EEPROM 数据手册中规定的 T_{WRL}（写周期时间）或 T_{HDDL}（保持数据时间）的时间。例如，对于 I^2C 接口的 EEPROM，在 STOP 条件后，总线需要保持空闲一段时间，以便内部写入操作完成。

(2) **连续写操作错误：**连续对 EEPROM 进行写操作时，可能会出现写入失败的情况。解决措施类似快速操作错误，在每次写操作完成后，需要强制等待一个规定的时间，确保前一个写操作已完成。

(3) **页写（Page Write）操作的边界问题：**使用页写功能时，写入数据没有写入到期望地址，同时会将期望地址附近数据破坏。这是因为 EEPROM 内部有一个页大小的写缓冲 RAM，页写操作会

选中一页并将内容映射到缓冲 RAM，数据写入从低端地址开始修改，超过页范围后会回到 00 地址。页写操作会将数据写入一个特定的” 页” 内。如果写入的数据跨越了页的边界，超出的数据可能会被写入到当前页的起始地址，导致数据覆盖或错位。因此，当需要进行页写或连续多字节写入时，首先需要对数据长度和起始写入地址进行判断。如果可能存在 EEPROM 地址自加出错现象（即跨页写入），需要将需要操作的数据分为两部分：先将前面一部分数据通过连续多字节写操作写入到当前页的结束地址处，然后再循环进行页写操作或多字节连续写操作，处理下一页的数据。

(4) **读操作引起的失误（越界现象）**：读出数据与 EEPROM 实际地址对应数据不符，原因是在连续多字节读取时，如果读取长度超过了当前页的边界，可能会导致地址自加出错。解决措施是当需要读取连续多字节时，首先需要对数据长度与起始读取地址进行判断。如果可能存在 EEPROM 地址自加出错现象（越界现象），则需要做相应的出错处理，或者返回一个出错标志。

(5) **电源稳定性问题**：检查 EEPROM 的供电是否稳定，是否有电压跌落或纹波过大。不稳定的电源可能导致写入操作失败。确保退耦电容正确连接并具有足够的容量。

(6) **I/O 引脚电平问题**：检查微控制器与 EEPROM 之间的 SCL 和 SDA 引脚电平是否正确，是否有干扰或信号完整性问题。确保上拉电阻值正确，且没有其他器件干扰总线。

(7) **EEPROM 寿命失效**：EEPROM 的存储单元在频繁擦写后可能会失效，一旦超过额定寿命，该存储单元将不可靠。EEPROM 的存储单元有擦写次数限制（通常在十万到一百万次之间）。在仪器系统设计过程中，必须将 EEPROM 的寿命考虑其中。如果预计某个存储单元的使用寿命将超过额定寿命，必须在接近额定寿命（并保留余量）的情况下，更改 EEPROM 的存储单元地址，将数据写入到新的、未磨损的地址。

(8) **初始化错误**：确保 EEPROM 在每次上电或复位后都进行了正确的初始化，包括器件地址、工作模式等。

(9) **读写时序不匹配**：仔细核对微控制器生成的读写时序是否完全符合 EEPROM 数据手册的要求，包括起始条件、停止条件、数据传输、ACK/NACK 信号等。

4. 以汽车行驶里程表存储为例，分析因 EEPROM 寿命失效而更改存储单元地址的具体方法。

汽车行驶里程表是一个典型的需要频繁更新且数据不能丢失的应用场景。里程数据会随着汽车的行驶而不断增加，并且需要掉电保存。EEPROM 由于其非易失性、按字节擦写（或页擦写）的特点，常被用于存储这类重要参数。

EEPROM 寿命失效问题：EEPROM 的每个存储单元都有其固定的擦写寿命，通常在 10^5 到 10^6 次之间。对于汽车里程表，里程数据会持续累加，这意味着存储里程数据的 EEPROM 单元会不断被擦写。如果总是写入同一个地址，这个地址很快就会达到擦写寿命，导致数据损坏或丢失。

磨损均衡（Wear Leveling）：为了解决 EEPROM 寿命失效问题，通常采用” 磨损均衡”（Wear Leveling）技术，即通过算法将数据均匀地写入到 EEPROM 的不同存储单元，从而延长整个 EEPROM 的使用寿命。以下是一种常见的方法：

(1) **划分存储区域**：将 EEPROM 的存储空间逻辑上划分为多个数据块或扇区。例如，对于一个 24C08（1KB）的 EEPROM，可以将其划分为多个大小相同的页或块。

(2) **引入指针或索引**：设置一个” 当前写入地址指针” 或” 有效数据块索引”，这个指针或索引本身也存储在 EEPROM 中，但更新频率远低于里程数据。

(3) **循环写入**：在系统首次启动或 EEPROM 首次使用时，将所有里程数据块初始化为无效状态，并将当前写入地址指针指向第一个数据块。每当里程数据需要更新时（例如，每增加 1 公里或 10 公里，或定时更新），系统不会直接擦写上一个里程数据所在的地址。相反，系统会读取当前写入地址指针，将新的里程数据写入到下一个可用的、未磨损的数据块中。写入完成后，更新当前写入地址指针，使其指向下一个循环写入的地址。如果到达 EEPROM 的末尾，则循环回到起始地址。在写入新数据前，可以选择性地将上一个存储的里程数据标记为无效（例如，写入一个特定的无效标记）。

(4) **数据读取**：在读取里程数据时，系统会遍历 EEPROM 的所有数据块，找到最近一次写入的有效里程数据。可以通过时间戳、版本号或特定标记来判断哪个数据是最新有效的数据。或者，根据当前写入地址指针，找到最后一次写入的地址，并读取该地址的里程数据。为了增加可靠性，可以连续存储几个相同的里程值，读取时进行多数表决。

(5) **容错机制（可选）**：在每个里程数据块中添加校验和或 CRC（循环冗余校验）码，以检测数据在写入或存储过程中是否发生错误。可以同时存储多份里程数据副本，如果一份数据损坏，可以从其他副本恢复。软件可以记录每个存储块的擦写次数，当某个块的擦写次数接近其寿命限制时，触发预警或提前将其标记为不可用。

比如说，假设 EEPROM 有 100 个地址，我们使用一个指针 `write_address` 来指示下一次写入的地址。初始状态为 `write_address = 0`；第一次里程更新时，将里程数据写入地址 0，然后 `write_address` 递增为 1；第二次里程更新时，将里程数据写入地址 1，然后 `write_address` 递增为 2... 第 100 次里程更新时，将里程数据写入地址 99，然后 `write_address` 递增为 0（循环）；读取里程时，遍历地址，找到有效且最新的里程数据。例如，可以存储” 里程值 + 序列号”，读取时找到最大的序列号对应的里程值。

通过这种磨损均衡的方法，每次更新里程时，数据都会写入不同的 EEPROM 地址，从而将擦写操作分散到整个 EEPROM 芯片上，大大延长了 EEPROM 的整体使用寿命，确保汽车行驶里程表的可靠性。

5. 在计算机系统中，分析硬盘和内存分别属于什么存储器类型？

(1) **内存（RAM - Random Access Memory）：**

内存主要属于随机存取存储器（RAM），是与 CPU 直接交换数据的内部存储器，也称为主存（内存）。其主要特点包括：随机存取（读写数据所需时间与数据位置无关）、易失性（当电源关闭时，RAM 中的数据会丢失，因此通常作为程序临时数据存储媒介）、高速（现代 RAM 的写入和读取速度非常快，存取延迟微不足道，几乎是所有访问设备中写入和读取速度最快的）、需要刷新（动态 RAM 需要定期刷新以保持数据）、对静电敏感（静电可能导致数据流失或电路损坏）。

内存主要分为静态随机存储器（SRAM）和动态随机存储器（DRAM）。其主要功能是存放操作系统、各种应用程序以及程序运行时所需的临时数据，CPU 直接从内存中读取指令和数据进行处理。在计算机系统中，内存是主存储器，位于 CPU 和硬盘之间，是 CPU 能直接访问的存储器。

(2) **硬盘（HDD - Hard Disk Drive / SSD - Solid State Drive）：**

硬盘属于非易失性存储器（Non-Volatile Memory），也称为辅助存储器或外部存储器。它与 Flash 存储器（闪存）的特性类似，即在没有电流供应的条件下也能够长久地保持数据。其主要特点包括：非易失性（断电后数据不会丢失）、大容量（相比内存，硬盘通常提供更大的存储容量，能够存储大量的操作系统、应用程序和用户数据）、速度相对较慢（相对于内存，硬盘的读写速度较慢，特别是传统机械硬盘涉及机械运动。固态硬盘虽然速度远快于 HDD，但仍慢于 RAM）、成本较低（单位存储容量的成本低于内存）。

硬盘主要分为机械硬盘（HDD，基于磁性存储，通过旋转盘片和读写磁头进行数据存取）和固态硬盘（SSD，基于闪存技术，通过电子方式存储数据，没有机械部件，因此速度更快、更耐用、更安静）。硬盘主要用于长期存储操作系统、各种应用程序、文档、图片、视频等用户数据。在计算机系统中，硬盘是辅助存储器，通常作为永久存储设备，CPU 不能直接访问，需要先将数据加载到内存中才能被 CPU 处理。

总结一下，内存（RAM）是计算机的” 工作区”，速度快但容量相对小且易失，用于 CPU 实时处理数据。硬盘是计算机的” 仓库”，容量大、非易失但速度相对慢，用于长期存储程序和数据。它们共同协作，构成计算机的存储体系，实现数据的快速处理和持久保存。

6. 比较采用软件定时器和实时时钟芯片进行当前时间设置与运行的区别。

(1) **软件定时器（Software Timer）：**

软件定时器是通过微控制器内部的硬件定时器（Timer）结合软件代码实现的。它利用硬件定时器产生周期性中断，在中断服务程序中对计数器变量进行增减操作，从而实现计时功能。

当前时间设置与运行：初始时间值通常在程序启动时通过编程设置，或者从非易失性存储器（如 EEPROM）中读取上次保存的时间。硬件定时器以固定的频率（例如 1ms）产生中断，在中断服务程序中，一个计数器变量会不断累加。通过累加的次数可以计算出经过的时间（秒、分、小时等）。

特点：精度受微控制器主频、晶振精度以及中断响应时间的影响，容易受到程序运行的干扰，例如长时间的中断禁用、复杂的计算任务等，可能导致计时不准确或” 漂移”。软件定时器本身不额外消耗太多硬件功耗，但其运行依赖于微控制器的正常工作，如果微控制器进入低功耗模式或断电，计时会停止。相对简单，只需要配置微控制器内部定时器并编写中断服务程序。无需额外硬件成本，仅占用微控制器资源。无法在掉电后保持时间信息，需要额外的非易失性存储器（如 EEPROM）来保存时间，并在下次上电时重新加载。

(2) **实时时钟芯片（RTC - Real-Time Clock）：**

实时时钟芯片是一种独立的集成电路，专门用于提供精确的时间和日期信息。它通常包含日历、时间寄存器，并需要外接 32.768kHz 晶体、匹配电容和备用电源（如纽扣电池）。

当前时间设置与运行：通过微控制器与 RTC 芯片进行通信（通常是 I^2C 或 SPI 接口），将初始设置的时间和日期数据写入 RTC 内部的寄存器。RTC 芯片内部的振荡器（通常由 32.768kHz 晶体驱动）独立运行，精确地进行计时。微控制器可以随时读取 RTC 芯片的寄存器来获取当前的实时时间和日期。

特点：具有较高的计时精度，通常在外部 32.768kHz 晶体的作用下，可保持在每天误差小于 0.50 秒的水平（对于时钟模块）。其精度与温度有很大关系，晶体的频率会受温度影响。匹配电容在

RTC 外围器件中起到修正晶体与 RTC 之间匹配问题的重要作用。通常具有极低的功耗，即使在主电源断开的情况下，也能依靠备用电池（如纽扣电池）长时间（几年甚至更久）维持计时。需要额外的硬件（RTC 芯片、晶体、电池、匹配电容）和相应的通信协议（如 I^2C）来读写数据。增加了额外的硬件成本。能够独立于主电源工作，即使系统断电也能保持时间信息，无需在每次上电时重新设置。通常提供到 2099 年内的日历功能，并以 BCD 码形式存储时间数据。

特性	软件定时器	实时时钟芯片 (RTC)
精度	较低，易受程序干扰，依赖主晶振	较高，独立晶振，更稳定
功耗	依赖微控制器工作，无法在低功耗或掉电下计时	极低，可由备用电池独立供电，掉电保持
成本	无额外硬件成本	增加硬件成本（芯片、晶体、电池等）
复杂性	软件实现简单，硬件配置简单	硬件连接和通信协议相对复杂
掉电保持	无法保持，需额外存储器保存	独立供电，掉电可保持时间
应用场景	对时间精度要求不高、不需要掉电保持的计时功能	对时间精度要求高、需要掉电保持、需要日历功能的场合，如数据记录仪、仪器仪表、智能家居等

选择软件定时器还是 RTC 芯片取决于具体的应用需求。对于对时间精度要求不高、不需要掉电保持的简单计时功能，软件定时器可能更经济高效。而对于需要精确时间、掉电保持、日历功能等复杂时间管理的应用，RTC 芯片是更优的选择。