第 2 章　嵌入式硬件基础

芯片：半导体材料制成的电路器件

引脚：半导体器件和外部连接的部分，排布在芯片两侧或四周

封装：将半导体器件以特定方式、特定方式、特定材料包装起来，提高可靠性、便于使用

（各种封装的名字和特点）

双列直插式封装 DIP

塑料方形扁平式封装 PQFP

薄形小外形封装 TSOP

塑料引线芯片载体封装 PLCC

插针网格阵列封装 PGA

球栅阵列封装 BGA

上拉电阻的功能：用于将引脚信号钳位在高电平，或者用来在驱动能力不足时提供电流

下拉电阻的功能：用于将信号钳位在低电平，或者用于吸收电流。

退耦电容：用于退耦，基于滤波作用来消除电路回波噪声，或者基于储能来快速提供能量补充

旁路电容：在电路中起到滤波、抗干扰的作用

第 3 章　嵌入式处理器

处理器包含执行单元 EU 和控制单元 CU

嵌入式系统中处理器架构：冯·诺依曼架构和哈佛架构

【简答】哈佛结构的特点是什么？

采用独立的物理存储器来分别存储程序指令与数据，并通过两套独立的总线进行独立传输由于程序存储器和数据存储器独立编址且采用独立的总线，该结构可以提供更大的存储器带宽，实现更为高效的指令流水机制，也是的数据的移动和交换更加方便。哈佛结构的这一优点，使其非常适合于设计运算量大、运算速度和数据吞吐量要求高的数字信号处理器。

【简答】嵌入式处理器类型、分类有哪些？

MCU

MPU

DSP

CPLD FPGA

SoC

指令格式：

80x86 指令格式

ARM 指令格式

MIPS32 指令格式

嵌入式处理器**类型**：

嵌入式微控制器 MCU

MCU 又称单片机，通常是 8 位

嵌入式微处理器 MPU

MPU 较 MCU 处理能力强，多位 16、32 乃至 64 位

嵌入式微处理器 VS 通用微处理器　差异：嵌入式处理器可能采用不同的架构，且集成大量 I/O 模块和存储器，构建嵌入式系统时不再需要外接这些模块，功耗较低

数字信号处理器 DSP

用于做数字信号处理（数学运算）

DSP 共性特点：

计算核体系优化

哈佛结构

突出的数学计算性能

集成度高

基于总线的可扩展性

较低的功耗等

CPLD，FPGA

可编程器件 PLD （逻辑可变：处理器的硬件逻辑是可变的

主要代表：CPLD FPGA

片上系统 SoC

处理器内核和各种外围芯片期间集成到一个芯片中，提高效率、性能，降低功耗；

可分为通用和专用两类

众核计算单元

典型嵌入式处理器体系

ARM 架构

MIPS 架构

PowerPC 架构　特点：一开始就注重高性能

ARM 架构：

ARM 指令集有两种（双指令集）：32 位 ARM 指令集、16 位 Thumb 指令集

MIPS 架构：

是一种简单的流水型、高度可扩展 32 位，64 位 RISC 架构

MIPS 架构倾向基于软件方式来避免流水线中的数据相关问题，把复杂的问题交给编译器处理

性能评价指标：

频率

处理速度　处理速度的评价指标：MFLOPS、TFLOPS、DMIPS、MHz

字长

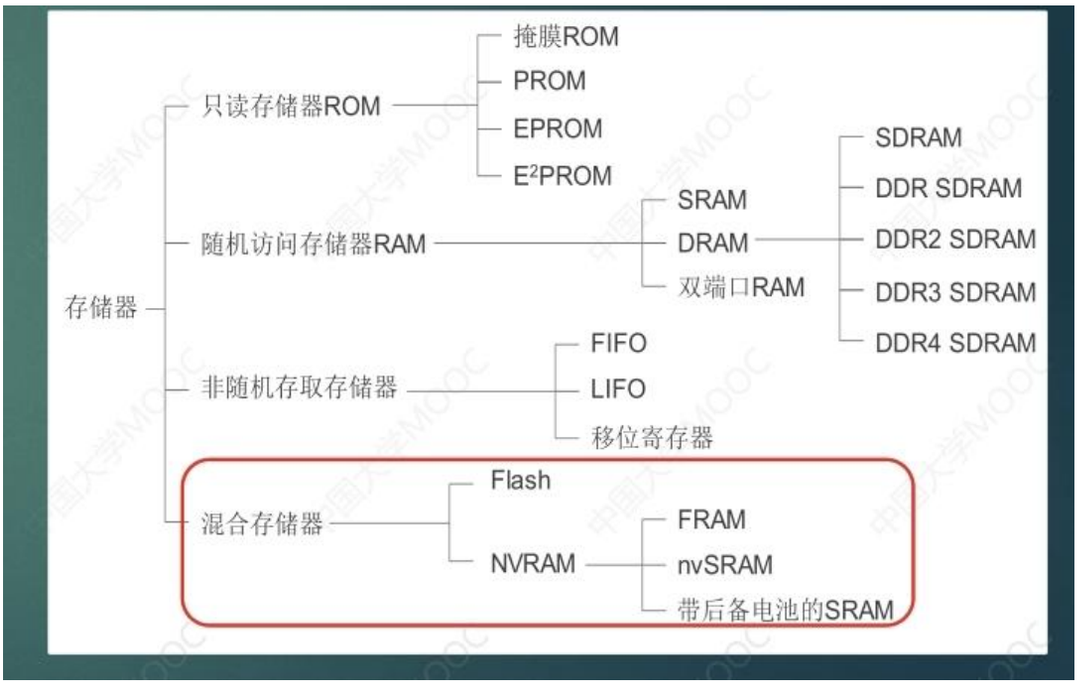
功耗
温度范围
无故障工作时间　无故障工作时间是器件、系统在失效或故障前可正常工作时间长度。
平均无故障工作时间 MTTF
平均故障间隔时间 MTBF：产品在操作使用或测试期间的故障之间能正常运行的平均连续时间

第四章　嵌入式存储技术

通用计算机采用了 Cache、主存储器、外部存储器组成的三级存储体系
读取周期：两次读取操作之间的最小时间间隔就是一个读取周期 t_{RC}
写周期：两次写操作之间的最小时间间隔就是写周期 t_{WC}

存储器结构模型：
基本结构：　存储体+I/O 接口电路+信号线
存储器技术指标：
只读性，易失性，位容量，访问速度，访问时间，功耗，可靠性
存储器分类：
随机访问存储器 RAM：SRAM，DRAM，双端口 RAM
主存采用的存储方案：SRAM DRAM
【重点】SRAM 和 DRAM 比较：
SRAM 比 DRAM 快；工作时，SRAM 比 DRAM 耗电多；DRAM 的存储密度大于 SRAM，存储容量大；DRAM 需要周期性刷新
DPRAM：两个处理器同时访问同一个存储器，进行数据共享
只读存储器 ROM
掩膜 ROM，PROM，EPROM，E^2PROM
PROM（一次性可编程）
混合存储器　同时具有 RAM 的快速读写访问特性和 ROM 的非易失特性
Flash：
NOR Flash
NAND Flash　（由于其高密度，多用于大量数据的存储）
NOR 型和 NAND 型接口方式不同：NOR 属于 SRAM 型接口，NAND 属于 I/O 接口
平时用的 USB（SSD 盘、U 盘）的闪存都是 NAND 型 Flash
·需要的容量不大，但是需要速度很快，用 NOR 型
·需要存储大量数据，而不是程序，用 NAND 型
Flash 设备一次能擦除一个扇区，而不是逐个字节擦除

NVRAM：非易失性随机访问存储器
·FRAM 铁电存储器
·BBSRAM 有后备电池供电的 SRAM，既保持了 RAM 的随机、快速访问特性，同时通过后备电源解决了系统掉电后 SRAM 的供电问题
存储器测试与验证
可读写存储器的测试：数据总线测试，地址总线测试，存储器件测试
只读存储器测试：校验和方式，CRC 校验方式



混合存储器包括 Flash 和 NVRAM（FRAM，nvSRAM，BBSRAM）

第 5 章　**最小系统外围电路设计**

嵌入式系统硬件是以处理器与存储器为核心

最小嵌入式系统硬件包括了嵌入式处理器、片上/片外存储器以及电源供电、复位、时钟等外围辅助电路。

1.电源电路

2.复位电路

复位：嵌入式系统上电之后，首先要对必要的寄存器、I/O 接口等资源的值和状态进行初始化

复位电路包括：

上电复位电路

阻容式复位电路（RC 复位）　开关接在电容的两端

手动复位电路　抖动问题

专用复位电路　一个专用的器件

看门狗复位电路：具有系统逻辑监控功能和故障时自动复位能力的电路

系统正常：周期性喂狗，向看门狗输出信号，看门狗收到信号就会清零，不会溢出，系统正常运行

系统出错：不喂狗，计数器计数，看门狗溢出，产生复位信号，系统复位

看门狗电路提高了系统的可靠性

看门狗电路的主要作用：在软件代码跑飞时进行复位

软件复位：通过软件设置一个特殊功能寄存器的位完成控制器的复位，复位结果如同硬件复位一样

3.　时钟电路

时钟的主要目的：同步各个器件的操作

时钟来自石英振荡器

电路抖动与消抖

毛刺信号产生原因：开关引脚的断开和接通，电磁干扰

抖动信号的特征：随机出现且能量极小
消除抖动：
硬件消抖（解决信号抖动的根本方式）
软件消抖

第 6 章 接口、总线与网络扩展

6.1 通用 I/O 与串行总线

（同步）SPI 串行外设接口 4 根线 全双工、一主多从式通信

两条数据线 MISO MOSI 用于收发数据，时钟线用于同步，片选线

SPI 的主机完成的工作：负责通讯过程中片选、时钟的发送，操纵了整个数据传输的所有细节

I^2C 二线结构，由 SDA（串行数据线）和 SCL（时钟线）组成

多主的主从式通信总线，2 线制

SPI 和 I^2C 区别：

SPI 没有地址，通过片选信号来选择芯片

I^2C 先给出地址，才知道和哪一个设备通讯

I^2C 发生总线竞争问题，需要总线仲裁机制

总线仲裁功能：确定现在谁可以通过总线发送数据

当两个设备同时向总线发送数据，发生总线竞争，默认是地址小的设备优先发送

I^2C 地址为 7 位，可以扩展到 10 位

（异步）UART 通用异步收发器

起始位：通知其他设备数据即将到来

停止位：告诉设备已经传送了一个字节的数据

USART 通用同步/异步收发器，在 URAT 基础上增加了同步机制，既可以异步也可以同步

RS-232 RS-485

RS-485 比 RS-232 抗干扰能力强的原因：RS-485 采用差分传输

GPIO 通用 IO

目的：让接口不是专用的，根据需要变成某一种类型的接口，增加芯片灵活运用的场合，增强 IO 接口的通用型

6.2 典型工业总线

（异步）CAN 总线 控制区域网络

CAN 协议体系包括物理层、数据链路层、应用层

表 6-8 通用通信总线参数比较

总线类型	最少线数	通信类型	多主支持	数据速率	节点数量	最大线缆长度(m)
SPI	3 线	同步	不支持	>1Mbps	<10	<3
I²C	2 线	同步	支持	<4Mbps	<10	<3
CAN	1 或 2 或差分 2 线	异步	支持	20Kbps~1Mbps	128 或更多	10000
LIN	1 线	异步	不支持	<20Kbps	16	40
UART	2 线	异步	不支持	3Kbps~4Mbps	2	15

第 7 章　嵌入式软件结构与基础软件

嵌入式软件结构分两种：基于裸机的嵌入式软件，基于嵌入式操作系统的软件结构

基于裸机的嵌入式软件（硬件上直接部署用户软件） 不采用 EOS

基于嵌入式操作系统的软件结构（通过嵌入式操作系统部署用户软件）：经典架构，衍生的架构

基础软件组件：

1、 ROM Monitor 监控程序　　启动代码：引导+初始化代码

2、 Boot Loader

一个 Boot Loader 的运行过程：

单阶段运行模式

多阶段模式

3、 BSP 板级支持包——一个系统级的功能软件包

BSP 作用：具有对硬件、外设初始化功能，支持 EOS 能够更好地运行于硬件平台、便于 EOS 移植

4、 嵌入式虚拟机

虚拟机软件主要分为两种类型：I 型虚拟机和 II 型虚拟机

I 型虚拟机：可以直接安装、运行于硬件平台的虚拟机，无需宿主操作系统

XEN

可以直接对硬件设备访问，效率更高

II 型虚拟机　VXware

嵌入式系统 BSP vs 通用系统的 BIOS

1） BIOS 是驻留在板上 ROM 或 Flash 中的一段可执行程序；BSP 是一个软件包，不能独立编译、运行。

2） BIOS 对于通用计算机具有一定的通用性；BSP 的形式取决于嵌入式操作系统、内涵取决于嵌入式硬件

3） BIOS 执行开机自检、初始化基本环境，加载 Boot Loader，之后将常驻程序库驻留在内存特定位置；BSP 中的组件可实现这些功能，且为操作系统提供了设备驱动

4） BIOS 是二进制代码，通常只能由设备提供商修改和升级；BSP 具有开放的源代码，允许设计人员编程修改，并允许在 BSP 中添加与本系统无关的驱动或程序

嵌入式软件的基本设计方法（一般了解）

嵌入式软件开发：

通用计算机软件开发中，开发平台和运行平台相同，完成开发即可运行；

嵌入式软件开发，大部分开发平台和运行平台不同

需要模拟环境、交叉编译环境，开发的软件需要经过处理才能在目标平台运行

嵌入式软件开发的形式：

基于裸机的软件开发 （无操作系统）

基于嵌入式操作系统的软件开发 （EOS 作为嵌入式硬件和软件的中间层）

第 8 章 嵌入式操作系统及其服务机制

协作与占先多任务管理

协作式多任务 OS

占先式多任务 OS （抢占式）

8.1 嵌入式操作系统的架构与模型

微内核、超微内核架构

核心功能与可选组件进行分离

EOS 采用微内核结构

嵌入式操作系统使用的内核一般是微内核的模型，可定制的模型

任务间通信：任务间相互传递信息，也就是进程间交换数据

交换的数据：控制信息、大批数据

8.2 服务机制与特性

内核：非抢先式内核、抢先式内核

任务：需要完成工作的一个特定部分

- 任务状态：反映任务当前在系统中所处的情形，由内核维护

  基本状态：运行、就绪、阻塞、睡眠、挂起

- 任务优先级：表示任务紧急程度的重要属性

  注：最低优先级的中断程序也要比最高优先级的任务优先级高

  通常情况下，优先级的值越小，表示优先级越高

  静态优先级、动态优先级

- 任务上下文及其切换：任务间切换要对上下文进行保护

任务控制块 TCB：进行任务管理的数据结构

任务调度器：任务调度由任务调度器完成

以特定的调度策略从就绪队列中选取要执行的任务，把它实际加载到处理机上运行

常用的调度策略：基于优先级的抢占式调度（排在前面），时间片轮转调度

任务优先级：表示任务紧急程度的重要属性

共享资源、临界资源

优先级翻转：

因为低优先级任务持有高优先级任务所需的临界资源，从而导致高优先级任务被延迟执行

解决优先级翻转问题的办法：优先级继承协议，优先级天花板协议

优先级继承协议：思想：占有独占资源的低优先级的任务的优先级 提高到被阻塞的高优先级任务的优先级

优先级天花板协议：

硬实时任务必须完成的时间：截止期
如果到截止期还没完成叫做逾期
硬实时：截止期必须被满足。任何截止期超越都会导致系统错误
软实时：软截止期允许被错过。一个任务的截止期被错过时，不会引发严重后果

任务间通信方式：
共享内存区域
消息队列
消息邮箱
管道

多任务管理是内核的基本机制
任务调度是内核进行多任务管理的核心功能之一，调度过程会切换任务上下文
抢先式调度是实时操作系统的重要特性
基于优先级的抢占式任务调度是保障实时性的基本要素

第 9 章　嵌入式软件组件
9.1 嵌入式图形库
μC/GUI
Nano-X Window
MiniGUI
QT
9.2 嵌入式文件系统
嵌入式系统的文件存储介质主要为 Flash
JFFS/JFFS2
Yaffs/Yaff2
UBIFS
只读文件系统 Romfs Cramfs AXFS
RAM 文件系统

9.3 嵌入式数据库
SQLite
Berkeley DB
eXtremeDB

第 10 章 嵌入式应用软件设计方法
10.1 嵌入式软件典型结构
轮转结构　不允许优先级
带有中断的轮转结构（前后台软件结构）　中断程序有优先级次序，所有任务代码在同一优先级上
　前台：中断
　后台：轮转结构
函数队列调度结构　中断程序有优先级次序，任务代码也有优先级次序

解决无法区分事件优先级的问题

基于嵌入式操作系统的软件结构（实时操作系统结构）　中断程序有优先级次序，任务代码也有优先级次序

（轮转结构、前后台软件结构、函数队列调度结构都是基于裸机的）

10.2 中断与数据共享问题

中断知识

禁止中断：系统做某些处理的时候不应该响应中断

为了解决共享数据等问题，提高可靠性，应该屏蔽掉中断

不可屏蔽中断 NMI：有些中断不应该被屏蔽

常见问题：

CPU 是如何探测到中断请求的？

每一条指令执行结束，CPU 都要查询是否有中断请求发生

注意：缺页中断响应时间，是在指令执行过程中响应

中断向量表放在什么地方？

有些处理器中，中断向量表总在一个地方，如 80186 的中断向量表总在 0x00000 处

有些处理器中，IVT 位置不固定，允许用户设置并允许用户程序通过某种方法获取

中断向量表是连续放置的

不会被中断的机器指令具有原子性

微处理器启动时中断是被禁止的还是启动的？

禁止的，在初始化中断向量后打开

·可以用 C 语言编写中断程序

关键字 interrupt 声明的程序是中断程序

用汇编语言写的中断程序将比 C 语言写的程序执行速度更快

中断程序设计原则

规则 1：中断程序不能调用任何可能会阻塞自己的 RTOS 服务　（RTOS：实时操作系统）

规则 2：确保 RTOS 调用返回中断程序

规则 3：避免执行分配、释放内存的操作

不允许在中断服务处理程序中使用信号量，否则中断程序可能被阻塞，导致整个系统错误

共享数据问题

解决共享数据问题：

采用开、关中断方式：在访问共享数据的时候不允许中断，关中断后微处理器将不会响应中断直至重新开中断

采用有条件开、关中断方式：引入计数器 count

（教材 453）

中断延迟：系统响应一个中断花费的时间

影响中断延迟的因素：

1. 中断（或所有中断）被禁止的最长时间
2. 任一优先级更高的中断的中断程序执行时间
3. 微处理器停止当前任务、保存必要信息以及执行中断中指令所需的时间

4. 从中断程序保存上下文到完成一次响应需要的时间
追求目标：中断响应延迟时间尽可能短
应该让所有中断程序尽可能短
如果高优先级中断程序的执行时间很长，那么低优先级中断程序可能很长时间不能执行，饿死现象

禁止中断影响中断延迟的时间（第一个影响因素）
为了解决共享数据问题，有时候禁止终端是必要的
禁止中断时间越短，系统的响应性能就越好

禁止中断的时间随着处理器的不同会不同

一些软件开发方法（基本思想）
实时系统基本设计
实时系统不仅关注系统功能，而且关注时间正确性（这些任务是否能及时的速度被执行）
硬实时系统：要求严格的时间期限
软实时系统：要求良好的时间期限

任务数量规划：
嵌入式系统设计的一个首要任务是将系统的工作划分位 RTOS 中的任务
多少任务为宜？任务少，任务粒度大；任务多，任务粒度小
任务越多，越容易控制不同部分的响应时间，如划分优先级；
任务越多，模块化程度越高，结构越清晰；
任务越多，任务的功能越单一，易于优化、维护和管理。
但是任务并不是越多越好，任务太多可能导致切换逻辑出现问题，任务间切换的代价也会高
需要合理规划任务数量和任务的粒度

为任务设定优先级：为需要快速响应的任务赋予更高的优先级
封装任务：

推荐的任务结构：任务运行于无限循环，等待 RTOS 的信号
优点：仅在等待一些不会及时发生的事件时阻塞，如读写 IO

考虑取消时间片：减少调度开销。倾向于使用优先级调度方式
任务切换带来额外开销，同一优先级别采用先来先服务的方式
考虑限制 RTOS 的使用：对 RTOS 裁剪定制，缩小代码体积，节省内存空间

信号量的封装：
在许多不同的模块中使用同一个信号量，不但会引起混乱还会导致错误
比较好的方法是：将信号量和其所保护的数据封装到一个模块中，以防错误的发生

节省存储空间：嵌入式系统存储空间有限

10.3 嵌入式软件设计机制
代码的可重入性
一个可以被多个任务同时调用或并发使用而不会出现逻辑错误的代码
判断一个函数是否可重入的规则：
1. 一个可重入的函数一般用原子的方法使用变量，除非这些变量存储在调用这个函数的堆栈中或这些变量是任务的私有变量
2. 一个可重入函数一般不调用其他不可重入的函数
3. 一个可重入函数一般不以非原子的方法使用硬件

软件看门狗方法
多任务系统中引入看门狗任务、心跳任务
该任务模拟看门狗电路，在系统运行时对其他重要任务和系统的运行状况进行监测
当发现系统中出现某种异常时，对任务或系统进行恢复处理
“与逻辑”的看门狗软件结构
“或逻辑”的看门狗软件结构
实现方式：通用实现方式、VxWorks 中的

建议任务在系统启动时就规划好，避免频繁地创建、取消任务

使用状态机刻画软件逻辑
状态机可以清晰地描述系统不同状态之间的转换关系。缺点在于，随着状态间的关系增加，状态空间呈现爆炸趋势
实时 UML 建模

10.4 软件工程方法
模型化嵌入式软件开发：
嵌入式软件日益复杂，传统方法难以满足研制需求
V 型开发过程　Y 型开发过程
模型驱动的软件开发 MDD：采用工具建模　“基于模型，正确构造”

第 11 章 调试、测试与仿真方法
开发是在宿主机开发，调试在目标机调试
11.1 嵌入式软件调试
基于宿主机的调试（在宿主机上进行调试）
指令集模拟器 ISS：利用软件来模拟目标嵌入式硬件指令系统，可以识别并解释执行目标处理器的指令
设备模拟器：模拟外围设备
将 ISS 和设备模拟器合起来，就能够在虚拟主机上虚拟出嵌入式目标平台
固定虚拟平台 FVP（虚拟机）

ROM Monitor 软件调试
ROM 仿真器　对单片机简单的机器调试的方式
ROM 仿真器是使用 RAM 器件和附加电路来仿真 ROM 的硬件设备
ICE 在线仿真器

JTAG 调试
11.2 嵌入式软件测试
七个实际测试阶段
常见的软件测试方法主要包括源码级的白盒测试及功能级的黑盒测试

【习题】
第一章
集成电路技术促进了嵌入式系统的诞生。
嵌入式系统设计中的资源配置基本要求是：根据需求量身定制
裸机软件结构：应用软件直接部署在硬件之上，其需要完成硬件配置和驱动
嵌入式系统的快速发展主要得益于：
　嵌入式操作系统、大规模集成电路技术、网络通信技术、高级编程语言技术
嵌入式系统的开发通常采用宿主机（开发主机）+目标机（嵌入式系统）的方式，目标机不具备作为开发主机的能力，且需要采用交叉编译、远程调试等技术

【第二章】
上拉电阻的作用：将引脚钳位在高电平
开放收集器 OC 的特点：不能输出高电平
对器件进行封装的意义在于：增强电气性能，增强散热性能，增强机械性能，增强化学稳定性
TTL、CMOS、RS-232 接口电路的电平特性：三类电平可以互相转换，前两者是正逻辑电平，后者是负逻辑电平
嵌入式系统的典型设计形式有：
　基于嵌入式处理器专门设计、基于单板计算机设计、基于通用计算机设计
上拉电阻的阻值越大，其上拉能力越强，产生的功耗越低
为了解决芯片引脚的信号浮动问题，可以在引脚连接一个上拉电阻或下拉电阻

【第三章】
嵌入式处理器中，每一个“计算核”都由执行单元和控制单元构成
MCU 适合用于交通灯控制器
DSP 适合于设计高保真数字耳机
哈佛体系结构，通过为数据和指令采用独立的存储器与独立的数据总线 提高数据与指令的访问速度
ARM Cortex-A 面向应用的配置，手机、平板
ARM Cortex-R 面向嵌入式的配置，用于有实时特征的高端嵌入式系统
ARM Cortex-M 面向微处理器配置，用于可穿戴、物联网等深度嵌入式系统
为了提升降低处理器复杂度并提升性能，可在处理器片内使用哈佛结构，片外采用冯诺依曼类型的接口
DSP 适合于数字信号处理，内部才用了多硬件乘法器单元

【第四章 存储系统】

NOR 型 Flash 的接口和存储结构使其具有随机访问特性，适合存放可执行代码并在片内执行，支持片内执行
NAND 型 Flash 不支持片内执行

处理器对存储器的访问都遵守先地址有效，再数据访问的逻辑次序

Flash 器件，扇区是最小擦除单位

嵌入式系统中，目前通常采用（半导体）技术的存储介质
嵌入式存储子系统以半导体存储器为主体，形式多样

整流电路：交流电变直流电

影响嵌入式处理器运行时功耗的因素：
电压越高功耗越大
频率越高功耗越大
工作态期间的数量越多功耗越大

消除电路中毛刺信号（或抖动）的方法有：
使用防抖器件
优化电路设计
引入滤波算法

【第六章】
1.SPI 通信接口中，主设备的 MOSI 引脚与从设备的（MOSI）引脚连接
2. 集成电路总线 I2C 中，多设备同时发送数据时会通过（总线仲裁）机制选定一个可以使用总线的主设备。
3. 差分传输是一种可靠性高的信号传输技术，在两根线上都传输同一信号。两条线路上同时传输的信号（幅值相等，极性相反）
4. RS-422 更适合于高可靠的远距离通信
5. GPIO 是通用接口，设计人员可以通过（寄存器组）配置接口的功能特性
6. SPI 接口支持一主多从的通信模式，通信之前，主设备通过（片选）线选定将要与之通信的从设备。
7. 集成电路总线 I2C 采用了寻址机制，所有外围器件都具有一个（7）位或（10）位的从器件专用地址码
8. 对于没有提供专门的 SPI 接口的嵌入式微控制器，可以用其（GPIO）引脚来模拟实现 SPI 的数据线、时钟线和片选线
9. RS-485 标准主要定义了（物理层的电气信号）特性，可以在 UART 规范的基础上进一步实现差分、多站的数据通信。

【第七章】
采用（虚拟机或嵌入式虚拟机）技术，可以在同一台嵌入式硬件设备上同时隔离地运行多个操作系统
嵌入式软件开发过程中，通常都要用到（交叉编译）技术，将宿主机上编写的代码转换为目标机上的代码