[概论 1](#_Toc26458)

[硬件基础知识 4](#_Toc7867)

[ARM架构 5](#_Toc30990)

[接口技术 12](#_Toc16549)

[人机界面 13](#_Toc7774)

[功率驱动 14](#_Toc32551)

[软件设计及操作系统 15](#_Toc26788)

[实时操作系统 18](#_Toc23443)

[软件设计 20](#_Toc15295)

[旁路攻击SCA及防御 21](#_Toc23236)

[工控系统信息安全 21](#_Toc16330)

[TrustZone 22](#_Toc2149)

[芯片操作系统与嵌入式软件安全 24](#_Toc19904)

[试卷 25](#_Toc19866)

[一、改错题(共5小题，每小题3分，共15分) 25](#_Toc5629)

[二、简答题（简明扼要） 26](#_Toc32238)

[三、论述题(共2小题，每题10分，共20分) 33](#_Toc30713)

[四、设计题 36](#_Toc21680)

可靠性：

ARM体系架构、接口技术、软件设计及代码优化、实时操作系统

安全性：

芯片操作系统与嵌入式软件安全

CM3\_M9900 、旁路攻击及防御、工业控制系统信息安全、 TrustZone技术

# 概论

1.嵌入式系统安全 = 可靠 + 安全

## 组成

由于嵌入式系统存储空间有限，要求软件代码紧凑、可靠，对实时性有严格要求。

嵌入式操作系统不是必需的。

嵌入式系统由嵌入式硬件与嵌入式软件组成

嵌入式硬件以芯片、模板、组件、控制器形式嵌于设备内部

嵌入式软件包括嵌入式操作系统(或调度器)和各种应用软件，一般固化在ROM或闪存中

软件与外部硬件和设备联系紧密，具有灵活的实用性由于嵌入式系统存储空间有限，要求

软件代码紧凑、可靠，对实时性有严格要求

# 硬件基础知识

1. 振荡器

振荡源：内部（不精准）、外部（RC/陶瓷/晶体/EC及差异、配置）

外接振荡元件一般选用晶体振荡器（替代品：陶瓷振荡器），或用价廉

的RC振荡器，也可用外部时钟源，作为振荡元件。也有的单片机将振荡元件也集成在芯片内部，叫内部振荡器（精度不高，1-5%的误差）

时钟占空比1:1

工作频率与功耗控制（节能模式）

指令周期：

1. 复位 ppt 5.1接口技术-通信

内/外部复位、电平及复位时长要求、特殊功能寄存器赋值与可靠性关系

外部复位电平及复位时长要求：H≥80%\*Vcc L≤12%\*Vcc，复位时长以时钟个数计

部分寄存器不受任何复位影响，其状态在上电时不确定，在其他复位时不变；

部分寄存器复位时会变为确定初值；

部分寄存器不受WDT复位的影响；

复位后要根据具体情况进行赋值操作。

1. 端口

* **通用端口电平要求**：H ≥70%\*Vcc L ≤ 30%\*Vcc
* 复位电平要求：H= ≥80%\*Vcc L= ≤ 12%\*Vcc
* **信号上升时间**：10%\*Vcc上升到90%\*Vcc所需时间
* **信号下降时间**：90%\*Vcc下降到10%\*Vcc所需时间

好信号：上升/下降时间 ≤ 信号周期\*4%

1. 功耗控制

功耗控制 P=K\* f²

分总线和部件控制时钟（节能、规避干扰）

1. 死机的本质：

硬件不会死机，只是硬件出错，没有按照正常的软件顺序执行，造成设备的运行结果不是预期的状态

1. 推挽输出与OC（OD）输出（ OC（OD）输出优缺点）

推挽输出功耗大，外部有短路容易坏，适用于驱动力大、信号快的，快充快放。

板与板之间用OC输入，有电阻存在，限制了电流，即使短路也不容易短路。

OC（OD）开路保证可靠性

# ARM架构

1. cortex系列
   1. Cortex-A: 高性能, 丰富的功能
   2. Cortex-R: 高可靠性, 高实时应用
   3. Cortex-M：低功耗, 代替微控制器（单片机）
2. M3架构：

采用Armv7-m架构，包括所有的16位Thumb指令集和基本的32位Thumb-2指令集架构，但不能执行ARM指令集。*Thumb-2在Thumb指令集架构(ISA)上进行了大量的改进，它与Thumb相比，具有更高的代码密度并提供16/32位指令的更高性能*

1. 特权模式和cpu模式

**用户模式(usr)**：ARM处理器正常执行程序时的处理。

**非用户模式（特权模式）**：

◆ 快速中断模式(fiq)：用于高速数据传输或通道处理。

◆ 外部中断模式(irq)：用于通用的中断处理。

◆ 管理模式(svc)：操作系统使用的保护模式。

◆ 指令/数据访问终止模式(abt)：当数据或指令预取终止时进入该模式，可用于虚拟存储及存储保护。

◆ 系统模式(sys)：运行具有特权的操作系统任务时的模式。

◆ 未定义指令中止模式(und)：当未定义的指令执行时进入该模式，可用于支持硬件协处理器的软件仿真

1. 中断和异常处理机制

中断响应的一般过程

(1)在每条指令结束后, 系统都自动检测中断请求信号, 如果有中断请求，且CPU处于开中断状态下, 则响应中断。

(2)保护现场, 在保护现场前, 一般要关中断, 以防止现场被破坏。保护现场是用堆栈指令将原程序中用到的寄存器推入堆栈。

(3)中断服务, 即为相应的中断源服务。

(4)恢复现场, 用堆栈指令将保护在堆栈中的数据弹出来, 在恢复现场前要关中断, 以防止现场被破坏。在恢复现场后应及时开中断。

(5)返回, 此时CPU将压入到堆栈的断点地址弹回到程序计数器, 从而使CPU继续执行刚才被中断的程序。

ARM中断步骤：

（1）保存现场。保存当前的PC值到R14，保存当前的程序运行状态到SPSR。

（2）模式切换。根据发生的中断类型，进入IRQ 模式或FIQ模式。

（3）获取中断服务程序的地址。PC指针跳到异常向量表所保存的IRQ或FIQ地址处，IRQ或FIQ的异常向量地址处一般保存的是中断服务子程序的地址，PC指针跳入到中断服务子程序，进行中断处理。

（4）多个中断请求处理。在ARM系统中，可以存在多个中断请求源，比如串口中断、AD中断、外部中断、定时器中断及DMA中断等，所以可能出现多个中断源同时请求中断的情况。为了更好地区分各个中断源，通常为这些中断定义不同的优先级别，并为每一个中断设置一个中断标志位。当发生中断时，通过判断中断优先级以及访问中断标志位的状态来识别哪一个中断发生了， 进而调用相应的函数进行中断处理。

（5）中断返回，恢复现场。当完成中断服务程序后，将SPSR中保存的程序运行状态恢复到CPSR中，R14中保存的被中断程序的地址恢复到PC中，继续执行被中断的程序。

应用程序中的异常处理：

1) 在异常向量表中的特定位置放置跳转指令，跳转到异常处理程序

2) 当ARM处理器发生异常时，程序计数器PC会被强制设置为对应

的异常向量，从而跳转到异常处理程序

3) 当异常处理完成以后，返回到主程序继续执行

1. 合法的操作模式转换机制

工作模式可通过软件控制改变，也可通过外部中断或异常处理改变

1. Thumb与ARM指令集区别

* 完成相同的操作，Thumb指令通常需要更多的指令。ARM指令集更为适合对系统运行时间要求苛刻的应用场合
* Thumb指令集没有包含进行异常处理时需要的一些指令，因此在异常中断的低级处理时，还是需要使用ARM指令

百度百科：

1.分支指令:跳转的范围小,除B指令外,都是无条件跳转.

2.数据处理指令:Thumb指令只有2个操作数,而ARM指令是3个操作数.

3.单寄存器加载存储指令:Thumb指令只能访问R0~R7.

4.多寄存器加载存储指令:Thumb指令只能访问R0~R7的子集.

5.Thumb特有指令:PUSH和POP作用于R13.

1. 安全处理器sc000、sc100、sc300特点

<https://www.cnblogs.com/sky-heaven/p/5485365.html>

SC000

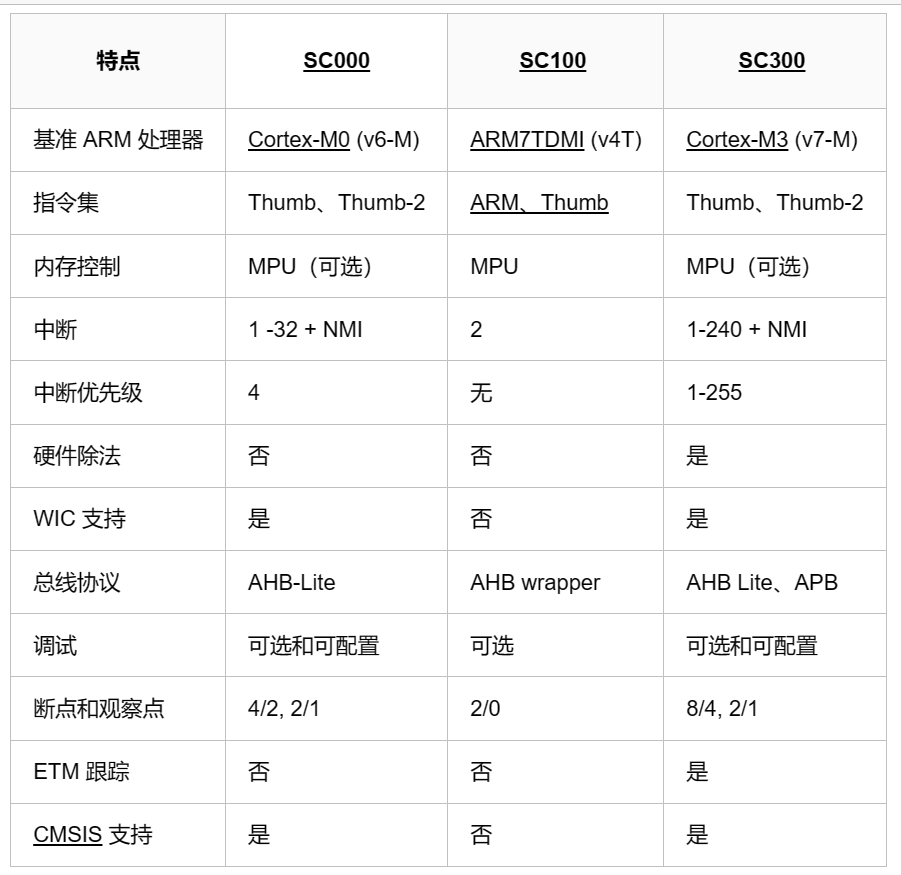
* 只占用 8/16 位的空间但却拥有 32 位的性能
* 具有卓越的能效表现
* 超低功耗
* 容易配置
* 向上兼容二进制的 SC300 处理器
* 基于流行的 Cortex-M0 处理器

SC100

* ARM 推出的第一个 SecurCore 处理器
* 入门级安全解决方案
* 基于 ARM7TDMI™

SC300

* 高性能
* 高功效
* 容易配置
* 适用于对安全性和性能要求较高的高端应用
* 基于 Cortex™-M3



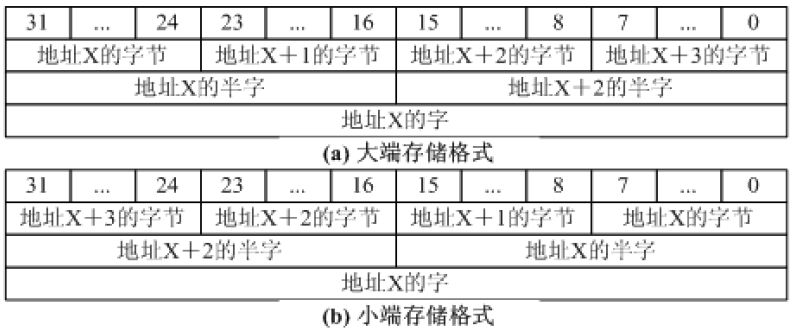
1. 大小端存储

* 大端格式（big-endian）：字的地址对应的是该字中最高有效字节所对应的地址；半字的地址对应的是该半字中最高有效字节所对应的地址。

32位字数据的最高字节存储在低字节地址中，而其最低字节则存储在高字节地址中。

◆ 小端格式（low-endian）：字的地址对应的是该字中最低有效字节所对应的地址；半字的地址对应的是该半字中最低有效字节所对应的地址。

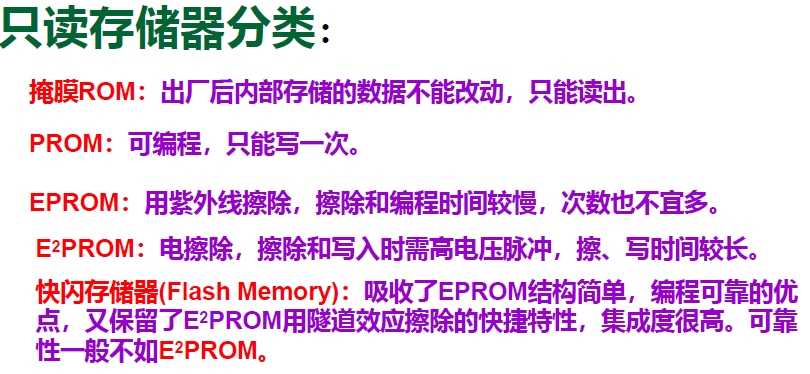
32位字数据的最高字节存储在高字节地址中，而其最低字节则存储在低字节地址中（ARM默认）



1. 主辅存种类
2. 主存储器：速度较快，容量较小，价格较高，用于存储当前计算机运行所需要的程序和数据，可与CPU直接交换信息，习惯上称为主存， 又称内存(内部存储器)。







ROM:

优点：电路简单，断电后数据不丢失，具有非易失性。

缺点：只适用于存储固定数据的场合

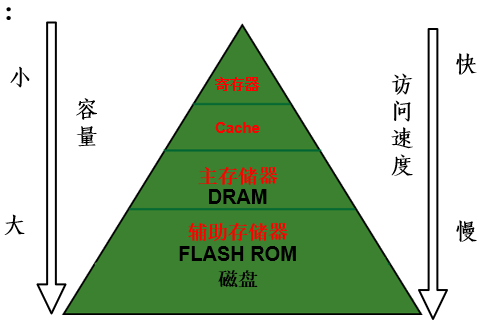
RAM:

优点：读、写方便，使用灵活。

缺点：一旦停电所存储的数据将随之丢失（易失性）

⑵ 辅存储器：速度较慢，容量较大，价格较低，用于存放计算机当前暂时不用的程序、数据或需要永久保持的信息,辅存又称外存(外部存储器)或海量存储器。外存要配备专门的设备才能完成对外存的读写。通常将外存归入到计算机外设类。软盘、硬盘、光盘、磁光盘MO、U盘

1. 存储层次结构



1. 时钟管理

时钟控制逻辑可以产生系统所需要的时钟信号，包括提供给CPU的FCLK，提供给AHB总线设备的HCLK 和提供给APB总线设备的PCLK

S3C2410 有2 个锁相环(PLLs)

◆ 一个提供FCLK、HCLK和PCLK

◆ 另一个提供USB时钟(48MHz)

1. 电源管理

1.空闲模式(IDLE Mode)

FCLK被关断，主要用于CPU core节电。可以任何通过外部中断唤醒。

2 .正常模式(NORMAL Mode)

耗电最大、可以通过关闭具体控制器的时钟来节电。

3 .低速模式(SLOW Mode)

在此模式下可以没有内部PLL，耗电情况依赖于外部时钟的频率。

4 .休眠/掉电模式(POWER-OFF Mode)

除了处理器唤醒逻辑单元外，处理器不损耗任何电量。可以通过EINT[15:0]

或RTC 报警唤醒系统

1. 定时技术

（1）时钟控制：系统为每个定时器设置有： 预分频器、分频器。

（2）定时器组成（5 部分）：

减法计数器(TNTn)、初值寄存器(TCNTBn)、比较寄存器(TCMPBn)、观察寄存器(TCNTOn)、控制逻辑等部分构成。

死区的概念：是一小段时间间隔，在这个时间间隔内，禁止两个开关同时处于开启状态。

死区发生器是在大功率设备PWM控制中常采用的一种技术，该功能用于在一个开关设备的断开和另一个开关设备的闭合之间插入一个时间间隔，这个时间间隔使得两个开关设备不可能同时被打开，即使是很短的一段时间也是如此

定时器0有死区发生器，用于大功率设备控制

RTC(Real time clock 实时时钟)

通常采用RTC 来提供可靠的系统时间，包括时、分、秒和年、月、日等；而且要求在系统处于关机状态下它也能够正常工作（通常在系统电源关闭后，由后备电池供电），它的外围也不需要太多的辅助电路，典型的就是只需要一个高精度的32.768KHz晶振和电阻电容

1. 中断

独立向量：每个中断享有独立的入口地址；

分组向量：若干中断按组共享组入口地址；

统一向量：所有中断共享一个入口地址。

计算机在执行某一程序的过程中，由于计算机系统内、外的某种原因，而必须中止原程序的执行，转去执行相应的处理程序，待处理结束之后，再返回继续执行被中止的原程序的过程。

普通中断：

快速异常中断：FIQ异常中断为快速异常中断，它比IRQ异常中断优先级高

1. DMA

如果一个定时器被配置为DMA模式，该定时器不会产生中断请求了

源：

目的：

块

# 接口技术

1. UART

异步串行通信接口也称为异步接收发送器

1. 通信方式：

●单工通信：数据仅能从设备A到设备B进行单一方向的传输。

●半双工通信(Half duplex)：数据可以从设备A到设备B进行

传输，也可以从设备B到设备A进行传输，但不能在同一时刻

进行双向传输。

●全双工通信(Full duplex)：数据可以在同一时刻从设备A传

输到设备B，或从设备B传输到设备A，即可以同时双向传输

1. 串行异步协议

1个起始位（”0”）、5-8个数据位、1个奇偶校验位（不一定需要）和1-2个停止位（”1”）

波特率（每秒传送的位数）、发送\接收缓存数量、中断种类

1. RS232

969年，EIA颁布的RS-232C标准规定：DTE和DCE之间最大传输距离为15m，传输数据速率不能高于20Kbps

1. RS485

半双工差分双绞线传输，有极性，最大的通信距离约为1219M，最大传输速率为10Mbps。传输速率与传输距离成反比，在100Kbps的传输速

率下，才可以达到最大的通信距离

RS485应用注意事项：极性、收发切换时间、最远端端匹配电阻（各110欧）、T型线长度（<3.4M）、线路保护

1. 串行通信协议应用注意事项
2. 接收与死机（超时处理、重新初始化、接收缓冲区防溢出）
3. 停止位（又叫数据保护时间）位数与可靠通信（有效通信速率、保障接收方的数据处理时间）
4. 波特率精度（ ± 2.5%、外置振荡器、配置选择与通信精度）与可靠通信
5. 通信协议、帧格式、收发切换时间及校验
6. 帧间隔（字符帧、通信协议帧）、帧长度
7. 波特率与通信距离
8. I2C

I2C总线最主要的优点是其简单性和有效性总线的长度可高达25英尺(约7.6m)，并且能够以100Kbps的最大传输速率支持40个组件。支持多主控。每个设备有唯一地址

I2C总线是由数据线SDA和时钟SCL构成，两根信号线必须OC并接上拉电阻。同步总线。

启动信号由主控器产生；在SCL信号为高时，SDA 产生一个由高变低的电平变化，产生启动信号。

仲裁：是一个在有多个主机同时尝试控制总线但只允许其中一个控制总线并使报文不被破坏的过程

1. I2C应用注意事项

1、ACK与死机（标准为永久等待，实际要加超时判错）

2、帧间隔与正常通信（保护时间）

3、不是所有芯片都支持广播地址

4、7位与10位地址

5、SCL、SDA 一定要开路输出（注意配置端口）

6、时钟速率与通信距离、EMC

7、结束与复位

1. 方向切换时间
2. SPI 串行外设接口总线

主机方式最高速率：5Mb/s

SPI系统可直接与各个厂家生产的多种标准外围器件直接接口, 它只需4条

线: 串行时钟线(SCK)、主机输入/ 从机输出数据线MISO、主机输出/从机

输入数据线MOSI 和低电平有效的从机选择线CS( SS)

具体时序由从设备决定、适用于大数据量（FLASH显示高速通信

1. 1-WIRE 493

数据

# 人机界面

1. 按键 ppt500
2. LED
3. 蜂鸣器

分类：

1. 按其驱动方式的原理分为：有源蜂鸣器（也叫自激式蜂鸣器）和无源蜂鸣器（也叫他激式蜂鸣器）
2. 按构造方式的不同，可分为：电磁式蜂鸣器和压电式蜂鸣器

压电式结构简单耐用但音调单一音色差、电压高、电流小。而电磁式由于音色好、电压低、电流大，所以多用于语音、音乐等设备。

1. 按封装的不同，可分为：插件蜂鸣器和贴片式蜂鸣器；
2. 按电流的不同，可分为：直流蜂鸣器和交流蜂鸣器

蜂鸣器驱动电路一般都包含以下几个部分：一个三极管、一个蜂鸣器、一个续流二极管和一个电源滤波电容。

1．蜂鸣器

其主要参数是外形尺寸、发声方向、工作电压、工作频率、工作电流、驱动方式（直流/方波）等。

2．续流二极管

蜂鸣器本质上是一个感性元件，其电流不能瞬变，因此必须有一个续流二极管提供续流。否则，在蜂鸣器两端会产生几十伏的尖峰电压，可能损坏驱动三极管。

3．滤波电容

滤波电容的作用是滤波，滤除蜂鸣器电流对其它部分的影响，也可改善电源的交流阻抗，如果可能，最好是再并联一个220uF的电解电容。

4．三极管，三极管起开关作用。

保护？

电磁干扰？

# 功率驱动

1. 功率晶体管

可靠性高

1. 场效应管

速度快

1. 晶闸管

晶体闸流管的简称，又可称做可控硅整流器

1. 电磁继电器

动作频率： ≤5次/秒、驱动及保护（保护二极管、灭弧）

1. 固态继电器（输入、隔离、输出、保护、类型）

* 输入电路：输入电路可分为直流输入电路，交流输入电路和交直流输入电路三种。有些输入控制电路还具有与TTL/CMOS兼容，正负逻辑控制和反相等功能，可以方便的与TTL,MOS逻辑电路连接。
* 隔离（耦合）：光电耦合和变压器耦合两种。
* 输出电路：大功率晶体三极管、单向可控硅、双向可控硅、功率场效应管、绝缘栅型双极晶体管（IGBT）。固态继电器的输出电路也可分为直流输出电路，交流输出电路和交直流输出电路等形式。

类型：

按负载类型，可分为直流固态继电器和交流固态继电器。交流固态继电器又可分为单相交流固态继电器和三相交流固态继电器。交流固态继电器，按导通与关断的时机，可分为随机型交流固态继电器和过零型交流固态继电器。

1. 步进电机 ppt543



失步原因：电机负载能力差、高速运转中电压不稳，造成输出扭矩下降，最终导至失步。转速设置过高，力矩不够。外部阻力过大、速度上升或下降过快。

注意事项：缓升缓降、齿轮间隙、皮带张紧。

1. BLDC马达

# 软件设计及操作系统

1. 嵌入式软件设计的演变 551
2. 顺序设计

顺序调用任务，

1. 状态机程序设计
2. 简易任务调度器







1. 操作系统
2. 任务切换（栈指针切换/堆栈搬移、公栈/私栈）
3. 嵌入式软件架构与层次

嵌入式系统分为４层，硬件层、驱动层、系统层和应用层。

1、硬件层处理器、存储器、通用接口（例通信）和I/O接口、扩展接口及电源等；

2、驱动层包含硬件抽象层HAL、板级支持包BSP和设备驱动程序；

3、系统层RTOS（或调度器）、文件系统、GUI、网络系统及通用组件组成；

4、应用层基于实时系统开发的应用程序组成。

分层便于在不同的CPU、不同OS间移植，节省开发时间，提高软件可靠性

1. 实时多任务操作系统与分时多任务操作系统

分时系统：软件的执行在时间上的要求并不严格，时间上的错误一般不会造成灾难性的后果。

实时系统：虽然事件可能在无法预知的时刻到达，但是软件上必须在事件发生时能够在严格的时限内作出响应（系统响应时间），即使是在尖峰负荷下，也应如此，系统时间响应的超时就意味着致命的失败。另外，实时操作系统的重要特点是具有系统的可确定性，即系统能对运行情况的最好和最坏等的情况能做出精确的估计。

1. 实时操作系统中的重要概念

* 时钟节拍

是特定的周期性中断。这个中断可以看作是系统心脏的脉动。周期取决于不同应用，一般在10ms到200ms之间。时钟的节拍式中断使得内核可以将任务延时若干个整数时钟节拍，以及当任务等待事件发生时，提供等待超时的依据。时钟节拍率越快，系统的额外开销就越大。

* 系统响应时间(System response time )

系统发出处理要求到系统给出应答信号的时间。

◼ 任务切换时间(Context-switching time)

任务之间切换而使用的时间。

◼ 中断延迟(Interrupt latency )

硬件接收到中断信号到操作系统作出响应，并转入中断服务程序的时间

1. 嵌入式Linux

种类？587

启动600

存储与文件系统602

写均衡605

7．RTOS

优点：

1、实时；

2、可以多人同时开发。因为每一个任务都互相独立了；

3、有些程序结构变简单了。

缺点（更多资源、成本、时效、能耗）

1、需要更多RAM/ROM

2、多一些程序用来各任务间同步；

3、多一些程序用来各任务之间数据交换；

4、由于OS需占用CPU时间，应用程序运行变慢。

什么时候用（任务耗时、复杂、资源充足、低功耗与RTOS：

1.当多个任务都很耗时，而这些任务可以并发调用，为了提高任务

之间的调度并发性，应该考虑使用OS；

2.当业务逻辑过于复杂，而通过自设计的调度器来调度时，使得设

计不能简单，相较于os往往趋于复杂，不易维护，为了使系统设计

更加简单可靠，可以考虑使用os；

3.当系统资源充足，开发团队熟悉OS的设计方法，使用OS更加节省

开发时间；

4.对于低功耗设备，使用前后台系统比使用os方法更易控制系统的

功耗，因为os系统方式，为了使任务调度及时，往往定时器中断调

度更频繁，cpu的唤醒频率更频繁，前后台系统功耗处理上要优于os

方式。

# 实时操作系统

1. 概念

◼软实时系统：各个任务运行得越快越好，并不要求限定某一任务必

须在多长时间内完成。

◼硬实时系统：各任务不仅要执行无误而且要做到准

1. 代码临界段

临界区，指处理时不可分割的代码。一旦这部分代码开始执行，则不允许任何中断打入。为确保临界段代码的执行，在进入临界段之前要关中断，而临界段代码执行完以后要立即开中断

数据撕裂（例如大于CPU位宽的数据、大于CPU位宽的定时器访问）：比如写一组数据到FLASH，写一半掉电了。下次再上电Flash数据就错了。再比如一个大于CPU字宽的变量，非中断程序读，中断程序写，若在非中断程序没完全读完时产生中断并写数据。

临界段代码影响中断响应时间（因为要关中断）

1. 资源

任何为任务所占用的实体都可称为资源。

资源可以是硬件设备，也可以是软件、存储空间

可以被一个以上任务使用的资源叫做共享资源。为了防止数据被破

坏，每个任务在与共享资源打交道时，必须独占该资源。这叫做互

斥

1. 任务619

* 休眠态：该任务驻留在内存中，但并不被多任务内核所调度。

◼就绪态：任务准备运行，但优先级比运行任务低，暂时不能运行。

◼运行态：任务是指该任务掌握了CPU的控制权，正在运行中。

◼挂起态：也可以叫做等待事件态。指该任务在等待某一事件的发生，

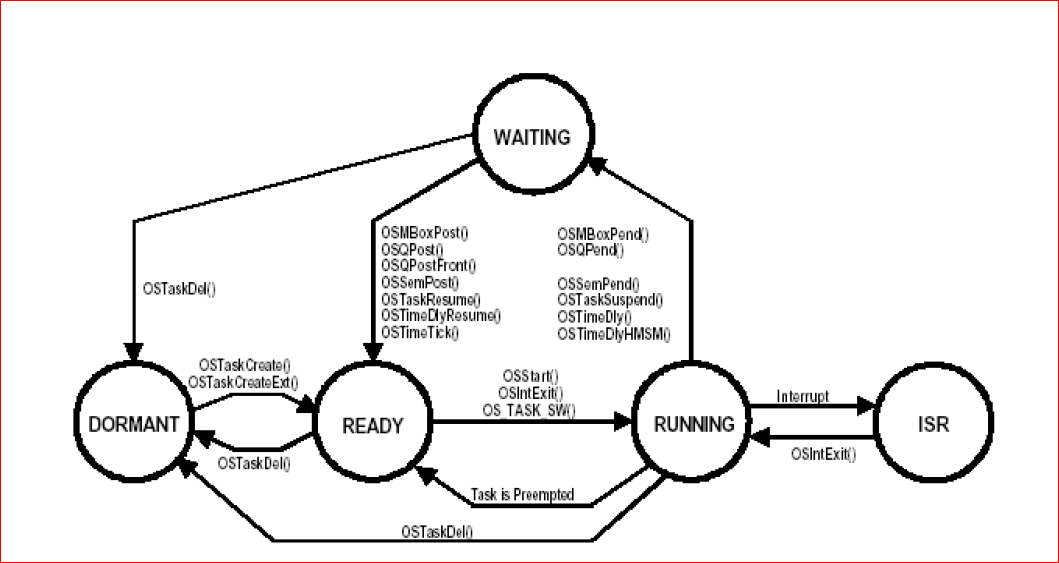
（例如等待某外设的I/O操作，等待某共享资源由暂不能使用变成

能使用状态，等待定时脉冲的到来或等待超时信号的到来以结束目

前的等待，等等）。

◼被中断态：发生中断时，CPU提供相应的中断服务，正在运行的任

务暂不能运行，就进入了被中断状态。



1. 调度 624
2. 中断时间 636

中断响应时间= 中断延迟+ 保存CPU内部寄存器的时间

中断恢复时间= 判定是否有优先级更高的任务进入了就绪态

的时间+ 恢复那个优先级更高任务的CPU内部寄存器的时间+

执行中断返回指令的时间

1. 存储器需求 644
2. 实时内核优缺点 646

# 软件设计

1. 代码优化731

平衡

利用硬件中断/DMA

小变量类型/寄存器变量：范围越小速度越快，占用内存越少

结构体和联合

局部/全局变量

算法优化：(查表、求余、平方、公共表达式、数学方法，移位乘除)

宏代码

内嵌汇编

循环语句

switch:跳转表和比较链/树、嵌套

自增/减和复合赋值、优化赋值

减少浮点运算

函数优化（减少参数/返回值、原型定义、本地函数、尽量避免使用标准库））

1. 代码可靠性746

程序存储器（复位区分/恢复/延时、 WDT、参数/执行检查、陷阱与错误捕获、 SLEEP 、变频、自检、校正）

RAM（数据单元、备份与空间距离、内存清理）、

不挥发存储器EEPROM/FLASH（数据单元、多编码备份、密文存储 、防拔、寿命与页面管理）

输入（周期性配置/输入/滤波、寿命管理）、

输出(周期性刷新内容/配置、寿命管理)）

# 旁路攻击SCA及防御

1. 密码攻击756

数学攻击：

实体攻击：

实现攻击：

1. 主动攻击((逃避检查、减少密码运算轮数、非法指令、 破坏RNG)(降/升时钟、异常电压/时钟/复位、光/射线/温度)
2. 被动攻击(提取泄漏物理量(温度/声波/能量/时间/电磁波/光)分析出密钥)
3. 旁路攻击 780
4. 旁路攻击分类(时间/能耗/声音/光/电磁波)
5. 旁路攻击分析技术(故障分析/侵入分析/时间分析/简单功耗分析/差分功耗分析/电磁辐射分析/高阶差分分析/汉明差分分析/模板分析)
6. 防御立足点(消除泄漏/增加难度(增加噪声/减小有效信息量)
7. 电路级抗攻击(互补电路/电路级Masking/预充电/随机化)、安全芯片
8. 软件防御技术811：

数据冗余(数据单元/备份)

控制冗余(前序/入口检查)

执行冗余(随机顺序/等价实现/随机延时功耗)

特定算法

软件陷阱

SLEEP躲避

软件防御技术关键是代码分块的粒度粗细和等价实现的数量

# 工控系统信息安全

1. 工控系统安全介绍817：

工业控制系统(ICS)是综合集成了计算机、网络、现代通信，微电子以及自动化技术，是对多种控制系统的总称，包括：

◼ 可编程逻辑控制器(PLC)；

◼ 监控和数据采集(SCADA)系统；

◼ 分布式控制系统（DCS)；

1. 特点821

* 实时性要求高，强调实时I/O能力；
* 可用性要求高，系统一旦上线，不能接受重新启动之类的响应，中断停机必须有计划（例检修）；
* 工控硬件要求寿命长、可靠性高，防电磁干扰，防爆，防尘等要求非常严格；
* 特有的工业控制协议通讯协议，不同厂商控制设备采用不同通信协议，很多协议不公开；
* 工控系统上线生产后，一般不会调整；
* 工控系统要求封闭性比较强。

1. 功能层次

(ERP(商务)、MES(制造管理)、SCADA/DCS、PLC、I/O）



1. 威胁

震网病毒、工控系统安全事件;“两化”融合,系统漏洞已知与无法修复、无法杀毒、非工控应用软件带来未知风险、移动介质和笔记本随意接入、工业无线网络边界不可见,网络安全风险急速增加)

1. 工控系统的脆弱性841

由于兼容性的问题，系统补丁和杀毒等安全措施不到位；设计主要考虑可用性/实时性，对安全性考虑不足，缺乏安全机制；工控系统用户信息安全意识薄弱，缺少完善管理)

1. 安全理念853

白名单：

层次化、边缘化、透明化

1. 安全策略

纵向分层、横向分域进行安全隔离防护，可用性监控、行为监控、指令监控)

1. 国际上有两种不同的工控系统信息安全解决方案868:

主动隔离式解决方案、被动检测式解决方案

安全不是一个结果，而是一个过程。

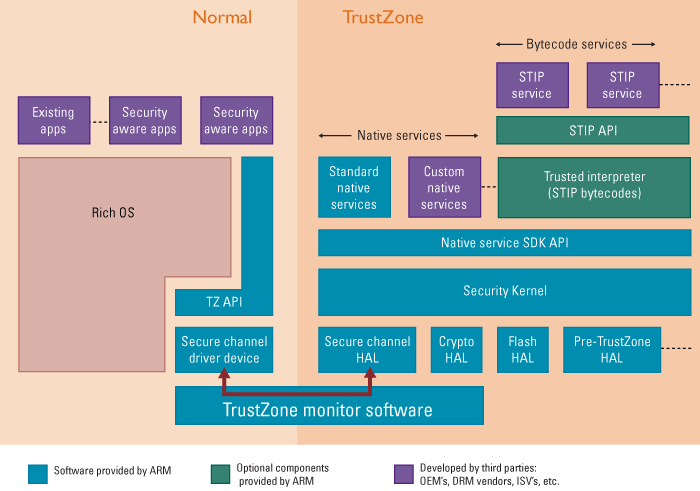
# TrustZone

1. 一套系统级安全解决方案，它是在尽量不影响原有处理器设计的情况下提高了系统的安全性 888

从ARMv6架构的处理器开始就已经集成了TrustZone技术，主要应用于Cortex-A 和 Cortex-M 系列处理器

1. 硬件架构（监控模式 893

它将CPU内核隔离成安全和普通两个区域，即单个的物理处理器包含了两个虚拟处理器核：安全处理器核和普通处理器核



1. 普通环境想要进入监控模式是严格被控制的，仅能通过以下的方式：中断、外部中断或直接调用SMC（Secure Monitor Call）指令。

安全环境进入监控模式则更加灵活些，可以直接通过写程序状态寄存器（Current Program Status Register，CPSR），另外也可通过异常机制切换到普通环境。

1. 为了实现安全性的要求，TrustZone形成了TOS和ROS同时运行在CPU上的两套操作系统。TrustZone启动时，安全引导程序从SoC的Rom中运行，进入TEE执行环境初始化，并启动TOS，逐级检查TOS各阶段的关键代码以保证TOS完整性；随后运行REE的引导程序，并启动ROS，至此完成整个系统的安全引导过程。
2. TrustZone３种方式的完整性安全策略：
3. 首先，它会先从片内执行引导程序完成系统安全状态的配置才启动操作系统，只有通过安全验证的模块才允许被加载；其次，在系统运行期间，由TrustZone技术提供的安全代码区会处理普通代码区的安全请求，在处理之前把安全请求保存在共享内存中，当安全检测通过后请求会被处理；最后，一组受限的、可信的进程可以在远离ROS的私有空间内安全地执行。
4. TrustZone架构存在问题

TrustZone镜像未加密存储

物理指针使用频繁

缺少严格的IO输入输出控制与检测

缺少ASLR, DEP机制

1. TrustZone采取的安全机制不完善

在安全功能实现中加入部分验证

部分厂商实现时使用位掩码屏蔽不再使用的功能

# 芯片操作系统与嵌入式软件安全

1. Global Platform的主要技术规范，该系列规范定义的安全域概念和功能，嵌入式系统不同角色的安全责任。GP的主要安全通道协议。

GlobalPlatform（GP）是跨行业的国际标准组织，致力于开发、制定并发布安全芯片的技术标准，以促进多应用产业环境的管理及其安全、可互操作的业务部署。

GP的工作重心主要集中在安全单元（SE）、可信执行环境（TEE）和移动消息（Mobile Messaging）等领域，其成熟的技术规范是建立端到端可信业务解决方案的工具，并服务于产业环境的多个成员，支持多种商业模式。

GP是全球基于安全芯片的安全基础设施统一的标准的制定者。

1. 如何统筹考虑密码算法在安全SoC中实现中的安全性、成本、功耗和性能。
2. 嵌入式安全中的错误注入攻击，功耗攻击和时间攻击的概念及主要防范方法。

错误注入：

嵌入式软件应能抵抗重复的错误数据的注入。

嵌入式软件应能阻止通过对重复错误注入得到的响应的分析导致的信息泄露。

这个安全目标要求能够有效的检测到这样的攻击并能对这样的尝试进行计数。

1. TEE的概念和特点，不同安全级别的技术要求。

TEE是Trusted Execution Environment的缩写，意为可信执行环境，在目前常见的手机系统构架中，TEE和RichOS（Android、iOS）一起运行，功能为隔离Rich OS及其应用程序对硬件和软件安全资源的访问，从而保证安全。

1. 嵌入式软件的安全审计的概念和主要功能。

嵌入式软件应提供方法来记录安全相关的事件，以便帮助管理者发现潜在的攻击和由于嵌入式软件的安全特性的错误配置使之陷入易被攻击的状态。

功能：

审计功能给嵌入式软件管理者提供一个机会回顾之前的操作信息以确定嵌入式软件是否受到一系列的攻击；

这些数据提供潜在系统攻击的早期的指示以便准备适当的响应；

这些记录数据可能包括芯片和操作系统的详细的历史信息，还有一些操作信息如开启和关闭环境传感器，checksum 校验失败，装载和删除应用等。

# 试卷

## 一、改错题(共5小题，每小题3分，共15分)

1.实时操作系统内核的性能可以以每秒钟能做多少次任务切换来评价。

错。做任务切换所需要的时间取决于CPU有多少寄存器要入栈。实时内核的性能不应该以每秒钟能做多少次任务切换来评价。

2.在ARM中复位属于异常。

对。

3.在大端存储模式下存储32位数0x1234567到2000-2003四字节单元中，2000H单元的数值为0x12.

错。应为0x01H。

4. I2C总线是异步通信总线。

错。同步。

5.商业操作系统几乎都是不可剥夺型内核。

错。商业软件几乎没有不可剥夺型内核。

1. ARM TrustZone技术是系统范围的安全方法

对

2.旁道攻击是一种针对密码系统实现上的物理攻击方式。但它既没有系统的攻击方式，也没有系统的解决方法。

对

3.在嵌入式系统设计中嵌入式操作系统是必需的。

错，不是必需的。

4. SPI总线对于不同的串行接口外围芯片，它们的时钟时序是不同的。

对

5. COS命令APDU包含一个必备的命令头。其中关于命令是否密文传输由INS说明。

错误，INS是指令类型，由CLA说明。

1.在小端存储模式下存储32位数1234567H到2000:200四字节单元中，200单元的数值为0x70.

错，67H  
2.uC0S-11操作系统是不可剥夺型内核。

错，商业cos几乎没有不可剥夺的内核  
3.SPI总线是异步通信总线。

错，同步通信  
4.YAFFS文件系统是为NOR FLASH量身定做的。

错，专为NAND Flash设计的  
5.C0S命令APDU包含一个必备的命令头。其中关于命令是否密文传输由P2说明。

错，CLA

1. 使用 C语言标准库可以加快开发进度,但在嵌入式系统开发中要尽量避免使用标准库。

对

2． 将数字信号转换成模拟信号的电路是ADC转换器。

模拟转数字

1. Thumb指令集是复杂指令集。

精简

1. Nand Flash比Nor Flash成本高。

成本低

1. ARM TrustZone 技术是系统范围的安全方法

对

## 二、简答题（简明扼要）

1.什么是嵌入式系统?嵌入式系统由哪几个部分组成?

答：

（1）嵌入式系统是以应用为中心，以计算机技术为基础，并且软/硬件可裁剪，可满足应用系统对功能、可靠性、成本、体积和功耗有严格要求的专用计算机系统。

（2）组成：微处理器、外围支撑硬件、嵌入式实时操作系统（或调度器）、用户应用软件。

2简述NAND Flash和NOR Flash的特点。

答：NOR和NAND是现在市场上两种主要的非易失闪存技术。

NAND FLASH：以页为单位进行读和编程操作，以块为单位进行擦除操作。数据、地址采用同一总线。实现串行读取。随机读取速度慢且不能按字节随

机编程。芯片尺寸小、引脚少，是位成本最低的固态存储器。

比较在p121



NOR Flash 的读取速度比NAND Flash 稍快一些， NAND Flash 的擦除和写

入速度比NOR Fl ash 快很多。

► NOR Flash 芯片在写入操作时，需要先进行擦除操作。NAND Flash 的擦除

单元更小，因此相应的擦除电路更少。

► NOR Flash 带有S RAM 接口，有足够的地址引脚来寻址，可以像其他SRAM

存储器那样与微处理器连接； NAND Flash 器件使用复杂的1/0 口来串行地

存取数据，各个产品或厂商的方法还各不相同，因此，与微处理器的接口复杂。

► NAND Flash 读和写操作都采用5 12 字节的块，这一点类似硬盘管理操作，很自然地，基于NAND Flash 的存储器就可以取代硬盘或其他块设备。

► NAND Fla s h 的单元尺寸几乎是NOR Flash 器件的一半，即NAND Flash 结

构可以在给定的尺寸内提供更高的存储容最，也就相应地降低了价格。

► NAND Fl a sh 中每个块的最大擦写次数是一百万次，而NOR Flash 的擦写次

数是十万次。

► 所有Fl ash 器件都受位交换现象的困扰。在某些情况下， NAND Flash 发生

的次数要比NOR Flas h 多。

5.简述WDT看门狗的原理和作用。

答：

看门狗：是一种电路，具有监视并恢复程序正常运行的功能。它是一定时器电路。

**看门狗的工作原理**：设一系统程序完整运行一周期的时间是Tp，看门狗的定时周期为Ti，要求Ti>Tp。在程序运行一周期后，修改定时器的计数值，只要程序正常运行，定时器就不会溢出。若由于干扰等原因使系统不能在Tp时刻修改定时器的计数值，定时器将在Ti 时刻溢出，引发系统复位，使系统得以重新运行，从而起到监控作用。

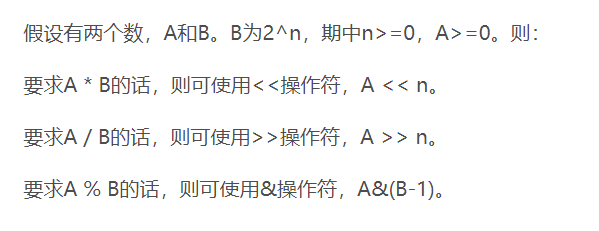
WDT的基本工作原理如下：在整个系统运行以后就启动了看门狗的计数器，此时看门狗就开始自动计时，如果到达了一定的时间还不去给它清零，看门狗计数器就会溢出从而引起看门狗中断，造成系统的复位。

看门狗的作用就是防止程序无限制的运行，造成死循环。它可以用在接收和发送数据时对接受和发送超时的处理，起到保护数据，保护电路的作用。

6.很多嵌入式芯片没有除法指令，系统是如何实现除法和取模运算的?

答：

用位操作（移位和+/-）代替除法；取模运算用 & 代替取模（P632）

  
7.要提高嵌入式系统可靠性，对未使用的引脚该如何配置?（问的老师）

答:芯片手册有说明的按手册处理；

未说明的，输出脚悬空处理；GPIO可设置为输出低悬空处理；输入脚有内部上拉或下拉的可配置下拉或上拉，否则必须通过电阻或直接接地或电源。

8.设计COS时，文件头和文件体为什么不能存放在存储器的相同页面?

答：

文件头不变，文件体可变，分页防止撕裂。

这样在修改文件体的时候文件头是不动的，可以达到文件头不变的目的。

9.试举例说明什么是时间攻击。

答：

1996年美国科学家Paul首先发现针对密码芯片的时间攻击法。该方法通过对密码芯片运算过程中执行时间信息的采集，结合密码算法的内部实现，证实了算法指令和执行时间存在相关性，从而推测出密钥信息。在Paul的基础上，Dhem等人将时间攻击方法成功地应用于破解RSA密码算法。

某个函数负责比较用户输入的密码和存放在系统内密码是否相同，如果该函数是从第一位开始比较，发现不同就立即返回，那么通过计算返回的速度就知道了大概是哪一位开始不同的，这样就实现了电影中经常出现的按位破解密码的场景。密码破解复杂度成千上万倍甚至百万千万倍的下降。

10.Y是无符号整型变量，请从代码大小、执行时间、完成功能等角度分析比较if(Y>1)和if(Y>=2)的异同。  
答：

（1）代码大小: Y>1的代码长度多余Y>=2， 因为Y>1要先判断大于，再判断不等于，而Y>=2只要直接判断大于等于即可。

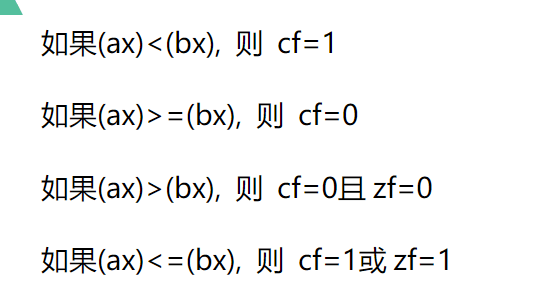
（2）执行时间: Y>1用时更长，因为Y>=2可以直接通过psw位来进行比

较。

（3）完成功能：一样

总的来说，总过程就是：先进行减法，然后先判断是否“>=0”，若是，psw=1；若不是还要判断是否“=”0

一般>=和<直接判PSW中的C标志就可以了；若判断>则先判断C标志，还要判断Z标志。<=等价于>



1.简述嵌入式系统的特点。

答：

面向特定应用、软/硬件可裁剪、低功耗、低成本、体积受限、实时性、较长的生命周期、本身不具备自主开发能力，需特定开发工具（仿真机、开发器）。

4. 电磁兼容性包含的两个要素是什么?

答：

①能在一定的干扰环境下工作； 有很多量化的指标，抗雷击。。。 ②不产生不可容忍的干扰。

5.软实时系统和硬实时系统的区别是什么?

答：

软实时系统：各个任务运行得越快越好，并不要求限定某一任务必须在多长时间内完成。

硬实时系统：各任务不仅要执行无误而且要做到准时

6.为什么在设计串口异步通信电路时必须使用外部晶体振荡器?

答：

计算机的整个工作是在时钟信号的驱动下，按照严格的时序有规律地一个节拍一个节拍地执行各种操作。单片机内部设计有定时电路，只需外接振荡元件即可工作。

内部振荡器的精度不高，会存在1-5%的误差

9.评估实时操作系统实时性能的三个重要时间是什么? （有问题）

答：（中断响应时间、中断处理时间、中断恢复时间）

系统响应时间：系统发出处理要求到系统给出应答信号的时间。

任务切换时间：任务之间切换而使用的时间。

中断延迟：硬件接收到中断信号到操作系统作出响应，并转入中断服务程序的时间。

10.提高输入和输出信号可靠性的软件方法是什么?

答：（6.4 PPT24）

输入：

周期性重复配置，不同内容放在不同block

周期性输入与滤波（例如程序判断、算术平均、加权平均、去极值算术平均、低通/高通/带通滤波）

输入部件寿命管理（例如按键次数）

时间限制（超时处理）与延时（绝对时间与相对时间）

输出：

内存可靠性高于端口，利用端口数据备份周期性刷新内容与配置（注意：内容在前）

输出部件寿命管理（例如LED、马达、喇叭）

输出与反馈

时间限制（超时处理）与延时（绝对时间与相对时间）

1.从应用的角度可将嵌入式微处理器分成哪几类?

答：

嵌入式微处理器 （Embedded Microprocessor Unit, EMPU）、嵌入式微控制器（Micro Controller Unit, MCU）、嵌入式DSP处理器（Digital Signal Processor, DSP）、嵌入式片上系统（System on Chip，SoC）。

3.比较CISC与RISC指令系统之间的主要差异。

答：

指令系统：RISC设计者把主要精力放在那些经常使用的指令上，尽量使它们具有简单高效的特色。对不常用的功能，常通过组合指令来实现。而CISC的指令系统比较丰富，有专用指令来完成特定的功能。  
存储器操作：RISC对存储器操作有限制，使控制简单化；而CISC机器的存储器操作指令多，操作直接。  
 程序：RISC汇编语言程序一般需要较大的内存空间，实现特殊功能时程序复杂，不易设计；而CISC汇编语言程序编程相对简单，科学计算及复杂操作的程序设计相对容易，效率较高。  
 CPU：由于RISC CPU包含较少的单元电路，因而面积小、功耗低；而CISC CPU包含丰富的电路单元，因而功能强、面积大、功耗大。  
 设计周期：RISC微处理器结构简单，布局紧凑，设计周期短，且易于采用最新技术；CISC微处理器结构复杂，设计周期长。  
 易用性：RISC微处理器结构简单，指令规整，性能容易把握，易学易用；CISC微处理器结构复杂，功能强大，实现特殊功能容易。  
 应用范围：RISC更适用于嵌入式系统；而CISC则更适合于通用计算机。

4.嵌入式系统硬件通常包含有哪些电路模块

答：

微处理器、定时器、中断控制器、I/O设备、存储器、传感器、接口等。

外围硬件设备：以嵌入式微处理器为中心，包括电源、时钟、复位、输入输出及驱动、存储、等其他电路模块 。

6.嵌入式操作系统内核的主要工作是什么?

答：

内核的工作主要在任务管理(Task Management)、任务调度(Task Scheduling)、进程间通信(IPC)。

多任务系统中，内核负责管理任务，或者说为任务分配CPU时间，并且负责任务之间的通讯。

内核的基本服务是任务切换。之所以使用实时内核可以大大简化应用系统的设计，是因为实时内核允许将应用分成若干个任务，由实时内核来管理它们。

调度（Scheduler、dispatcher）也是内核的主要职责之一，就是要决定该轮到哪个任务运行了。多数实时内核是基于优先级调度法的。每个任务根据其重要程度的不同被赋予一定的优先级。

10.请说明嵌入式芯片引脚配置为推挽输出和0D开温输出的差别及对可靠性的影响。

答：

0C/OD不需要输出高电平，需要外部上拉电阻，可以通过控制外部上拉电阻电流来控制，电路不容易损坏，通常用于电路板之间的连接，而推挽输出通常用于板内。推挽输出的高低电平都有信号驱动，适合比较高的时钟频率（上课讲的）。

1. 嵌入式文件系统的写均衡是指什么。

JFFS中使用多级链表管理需要回收的脏块，并且使用系统生成伪随机变量决定要回收的块

（区块擦写均衡）对每次写入请求，都合理安排写入的区块，使每个区块的擦写次数同步增长

均衡写入存储介质，保证nand flash上各块的擦写次数均衡，由于Nand Flash由于有擦除次数的限制，从而通过写均衡提高nand flash的使用寿命。

1. 可重入型函数如何编写

可重入型函数或者只使用局部变量，即变量保存在CPU寄存器中或堆栈中。如果使用全局变量，则要对全局变量予以保护。

1. 提高循环语句效率的方法

* 在多重循环中,应将最长的循环放在最内层,最短的循环放在最外层.这样可以减少CPU跨切循环的次数；

◼充分分解小的循环；

◼公共表达式放在循环外；

◼使用指针，少使用数组；

◼判断条件用0；(例如自减延时函数)；

◼while与do while：do…while编译后的代码的长度短于while；

◼WHILE(1)与FOR(;;): FOR(;;)编译后的代码的长度短于WHILE(1);

◼循环展开；

◼相关循环放到一个循环里。

1. 数据单元有哪两部分组成，为何要使用数据单元？

数据+校验码

使用数据单元是用来保证数据的完整性。为防止硬件攻击行为修改存储体上的静态数据和程序代码, 可以在存储数据(及程序代码)的时候增加一定长度的冗余数据,也就是校验码组成数据单元，保障安全。

1. 什么是能量攻击（功耗攻击）

一种利用密码设备运行时的功耗泄露信息来推测密码系统所进行的操作和秘密信息的攻击方法

1. 死机的本质

硬件不会死机，只是硬件出错，没有按照正常的软件顺序执行，造成设备的运行结果不是预期的状态

## 三、论述题(共2小题，每题10分，共20分)

1.什么是旁路攻击?抵御旁路攻击的软件方法有哪些?

旁路攻击 SCA（side channel attack），是指加密系统能够以各种形式泄漏信息，比如温度、声波、能量耗散、执行时间、电磁泄漏和光信号等，通过提取加密电路中的泄漏物理量，分析泄漏物理量变化规律来从而分析出密钥。

抵御旁路攻击的软件方法：数据冗余(数据单元/备份)；控制冗余(前序/入

口检查)；执行冗余(随机顺序/等价实现/随机延时功耗)；特定算法；软件陷阱；

SLEEP躲避。

数据冗余(数据单元/备份)：为防止硬件攻击行为修改存储体上的静态数据和程序代码, 可以在存储数据(及程序代码)的时候增加一定长度的冗余数据, 用来保证数据的完整性。一种简单的方法就是在每组数据的后面增加循环冗余码。

控制冗余(前序/入口检查)：控制冗余的主要思想就是在包含有关键代码的函数中添加多重控制, 以最大程度上保证程序执行是处于完全安全、可信的环境之中。

执行冗余(随机顺序/等价实现/随机延时功耗)：对密码算法中独立的(逻辑上没有继承关系、时间上没有先后关系)代码段按随机顺序执行；随机延时和随机功耗插入；未用资源的随机介入；SLEEP的介入；平衡内部数据的汉明重量；做内部数据比特组的分隔；数据单元的等价实现、同步及随机访问。

软件防御技术关键是代码分块的粒度粗细和等价实现的数量

2.请说明COS设计中完工操作的作用和实现方法。（PPT没有，要改写）

COS：chip OS，芯片操作系统。一般指 智能卡、USB- KEY 等安全产品的操作系统

SOC：system on chip ，片上系统

完工操作作用：

为了保证安全，维护COS开发者的利益，防止芯片厂商仿制、冒充产品。所以COS有一个架构，实际上交给芯片厂商的COS是不完整的，必须从芯片厂商那里拿回芯片后，重新对芯片进行下载、配置才能得到完整的功能，此时因为芯片厂商拿到的COS不完整，故而足够安全。

实现方法：

在交给芯片厂商时，COS被放在ROM空间里，但相关的函数命令的入口却被放在EEPROM中，并且其中的跳转表为空。待厂商处理完毕，芯片拿回后，再通过命令将下载的数据完整地填充在EEPROM跳转表中，如此就可以通过查询跳转来实现芯片正常功能。

把COS放在ROM空间里，但是COS里有很多模块，比如说COS有一条一条的命令，每一个命令有一个入口，可以把每个命令的入口存放到COS的FLASH或者EEPROM里。交给厂家的时候，EEPROM里是空的，所以就没有办法让COS完成功能，。当我们实际使用的时候，利用ROM里的一条命令，将下载的数据存放到EEPROM的跳转表里。当这个表填完后，若想实现某一个功能，需要查找跳转表，才能访问ROM空间的实际位置。

完工操作，就是将COS的一些信息完善化，填充好， 让这个卡片变成一个真正能用的设备。它是一条特殊指令，在跳转码里去改。

1.什么是实现攻击?抵御实现攻击的软件方法有哪些?

实现攻击：加密算法的实现电路经常会泄漏一些有用信息，这些信息能够被用来分析加密算法的敏感数据或密钥，这即是实现攻击。实现攻击可分为主动式攻击（失效分析攻击）和被动式攻击（旁路攻击）

抵御实现攻击的软件方法：

（1）数据冗余；

（2）控制冗余；

（3）执行冗余；

（4）特定算法；

（5）软件陷阱；

1. SLEEP躲避

2.工业控制系统有何特点?实现工控系统安全理念有哪些?国际上两种不同的工控系统信息安全解决方案是什么?

工业控制系统特点：

（1）实时性要求高，强调实时I/O能力；

（2）可用性要求高，系统一旦上线，不能接受重新启动之类的响应，中断停机必须有计划（例检修）；

（3）工控硬件要求寿命长、可靠性高，防电磁干扰，防爆，防尘等要求非常严格；

（4）特有的工业控制协议通讯协议，不同厂商控制设备采用不同通信协议，很多协议不公开；

（5）工控系统上线生产后，一般不会调整；

（6）工控系统要求封闭性比较强

实现工控系统安全理念：白名单、层次化、边缘化、透明化

国际上两种不同的工控系统信息安全解决方案：

主动隔离式解决方案：即相同功能和安全要求的设备放在同一区域内，区域间通信靠专有管道执行，通过对管道的管理来阻挡非法通信，保护网络区域及其中的设备。其典型代表是加拿大 Byres Security 公司推出的 Tofino 工控系统信息安全解决方案。

被动检测式解决方案：被动检测式解决方案延续了IT系统的网络安全防护策略。由于IT系统具有结构、程序、通信多变的特点，所以除了身份认证、数据加密等技术以外，多采用病毒查杀、入侵检测等黑名单匹配的方式确定非法身份，通过多层次的部署来加强网络信息安全。其典型代表是美国Industrial De-fender公司的工控系统信息安全解决方案。

1. 阐述串口通信在应用时的注意事项

1、接收与死机（超时处理、重新初始化、接收缓冲区防溢出）

2、停止位（又叫数据保护时间）位数与可靠通信（有效通信速率、保障接收方的数据处理时间）

3、波特率精度（ ± 2.5%、外置振荡器、配置选择与通信精度）与可靠通信

4、通信协议、帧格式、收发切换时间及校验

5、帧间隔（字符帧、通信协议帧）

6、波特率与通信距离

## 四、设计题

用C语言设计一个去极值平均滤波算法的函数。

说明： a)接口自己定义；

b)请注意代码质量；

c)添加注释。

### 去极值平均滤波

答：假设数据在数组uchar Data[18]中，选择18是因为去掉最大、最小还有16个数，可以避免做除法。代码如下：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

函数名: .uchar AVR (void)

描 述: 去极值平均滤波算法函数，数据在数组uchar Data[18]中

输入值: 无

输出值: 无

返回值: 去极值平均值

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

.uchar AVR (void)// 去极值平均滤波算法函数

{

uchar \*p; //使用指针比数组快

uchar Temp ; //定义缓冲区，可以避免频繁进行数组或指针运算，

uchar Min = 0xff; //数值滤波最小值,初始化=0xff

uchar Max = 0; //数值滤波最大值,初始化=0x00

uint Sum = 0; //累加和,初始化=0x0000

for (p=&Data; p<(&Data+18); p++) {//滤波计算,数据数组Data[18]去极值平均

Temp = \*p; //取数据

Sum += Temp; //计算数值累加和

if (Temp < Min ){ //比较，求最小值

Min = Temp;

}

else {

if (Temp >= Max ) { //比较，求最大值

Max = Temp;

}

}

}

Sum -= Max + Min; //累加和去除最大和最小值

return( Sum >> 4); //返回16次的平均值

}

### 滑动滤波

滑动滤波参考：

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  
函数名: .uchar AVR (void)  
描  述: 滑动平均滤波法，数据存放在uchar Data[16]，Data数组当作一个队列，Data[0]是队首，先出。  
输入值: 数组的首地址指针Data  
输出值: 无  
返回值: 滑动平均值  
\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  
#define N 16  
.uchar  AVR (uchar \*data)// 滑动平均滤波函数  
{  
uchar  \*q; //循环求和的指针  
uchar  \*New\_value=&data; //新的值,使用指针更快  
uchar i=0; //计数器  
uint  Sum = 0; //累加和,初始化=0x0000  
for(;;){  
\*New\_value= get\_ad();  //获取新值  
i+=1; //计数器加1  
\*New\_value++; //指针后移，指向Data的下一个位置  
if(i>=N){ //如果队列满了，就重置  
p=&data;  
i=0;  
}  
  
for (q=&data; q<(&data+18); q++){ //滤波计算,数据数组Data[18]去极值平均  
Sum  += \*q;   //计算数值累加和  
}  
return( Sum >> 4); //返回滑动滤波的16次的平均值  
}

<https://blog.csdn.net/weixin_42019584/article/details/80142292>

### 去极值滑动平均滤波算法

#define size 6000//数组大小

#define N 12//滑动平均滤波计算平均值时所取的点数

/\*上面两句在使用下面这个函数的时候加到程序的开头\*/

void Smooth(float data[])

{

uchar Temp ; //定义缓冲区，可以避免频繁进行数组或指针运算，

uchar Min = 0xff; //数值滤波最小值,初始化=0xff

uchar Max = 0; //数值滤波最大值,初始化=0x00

Sum1=0;

for(int j=0;j<size;j++)

{

if(j<N/2)

{

for(int k=0;k<N;k++)

{

Sum+=data[j+k];

}

data[j]=Sum/N;

}

else

if(j<size -N/2)

{

for(int k=0;k<N/2;k++)

{

Sum+=(data[j+k]+data[j-k]);

}

data[j]=Sum1/N;

}

else

{

for(int k=0;k<size-j;k++)

{

Sum1+=data[j+k];

}

for(int k=0;k<(N-size+j);k++)

{

Sum1+=data[j-k];

}

data[j]=Sum1/N;

}

}

}