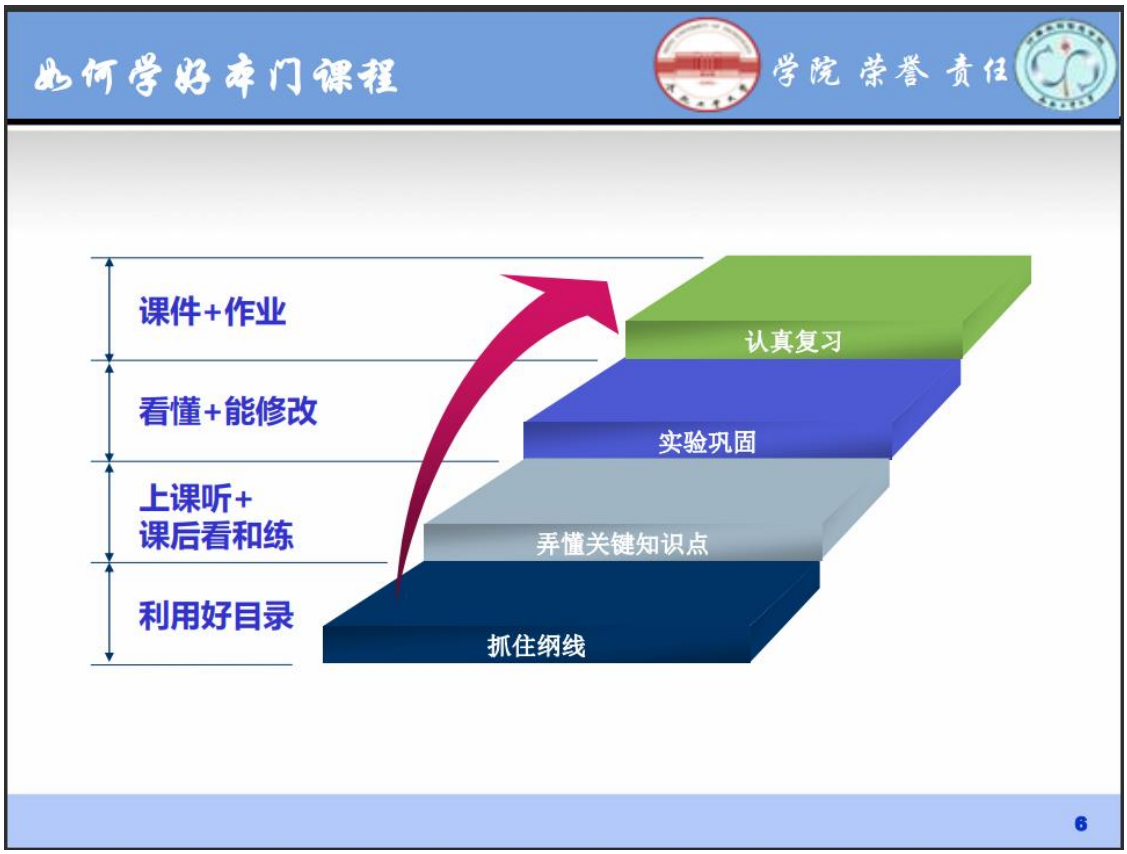
写在前面

题型：单选 10 填空 10 判断 10 简答 5 分析设计题 2

如何备考？

所有考点仅包括在 **ppt** 中。



每一章需要掌握的**重点**在哪里？

每一章 ppt 的**最后一页 ppt** 都有**本章要点，本章要点+概要梳理** 对照 **ppt** 复习。

概要梳理☆☆☆

概要梳理

学院 荣誉 责任

第一章：嵌入式系统的特点，嵌入式系统的组成，嵌入式系统的分类

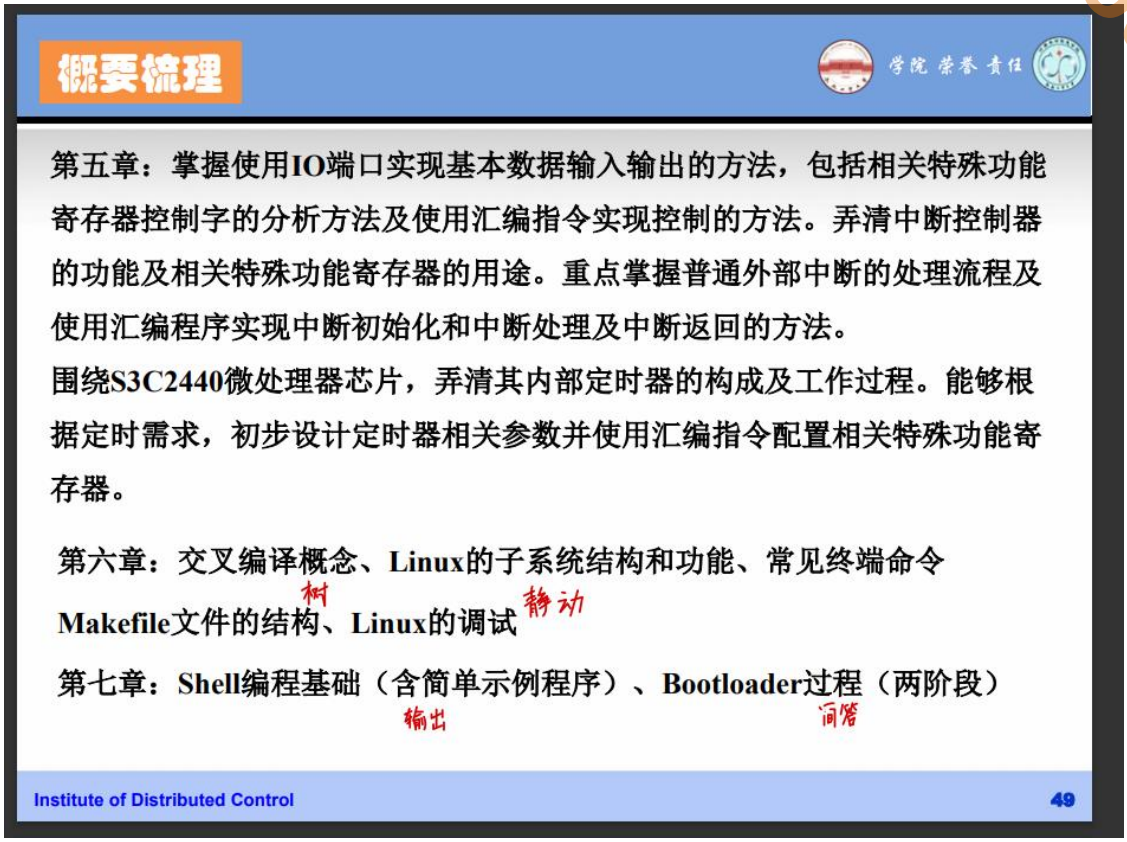
第二章，计算机系统两种体系结构和两种指令系统的区别；ARM内核命名规则的含义，ARM微处理器七种运行模式的特点，ARM微处理器两种工作状态的区别和切换方法，CPSR寄存器控制字分析，ARM体系结构中的两种存储格式，ARM处理器MMU的地址转换过程及虚拟地址到物理地址的转换方法、ARM的七种异常类型，异常向量、异常向量地址、异常向量表的含义，ARM状态下异常处理过程。

概要梳理

学院 荣誉 责任

第三章：8种寻址方式的特点，熟悉常见arm汇编指令的格式和功能；能够看懂基本的arm汇编程序，能够结合第五章学习的各种接口，编写简单的接口操作汇编程序。

第四章：熟悉嵌入式存储器的分类，及不同类型存储器的特点和使用场合。弄清嵌入式存储器系统的构成及其存储空间的分布和特点。掌握基本的存储器芯片与嵌入式微处理器芯片的连接方法。



最后一节课录课笔记

个人感觉还是概要梳理讲的清楚些，老师再过每章节时，感觉就像把概要梳理讲了一遍，所以下面如果有些记录不清晰的，就看概要梳理上是怎么说的，个人觉得以概要梳理+每章节 ppt 最后一页为主

第一章

都是概念，学起来最轻松

嵌入式系统的概念

嵌入式系统的特征(指的是三要素？)

嵌入式系统的区别

嵌入式系统的特点

嵌入式系统的分类

第二章

介绍了 ARM 微处理器

其特点相对于通用计算机来说。嵌入式系统的微处理器和通用计算机的微处理器从体系结构角度 从指令集角度 从流水线角度的不同要知道，知道微处理器属于什么类型。

出卷子宗旨：死记硬背的只有必须要记的才要死记硬背，类似这样的肯定不考

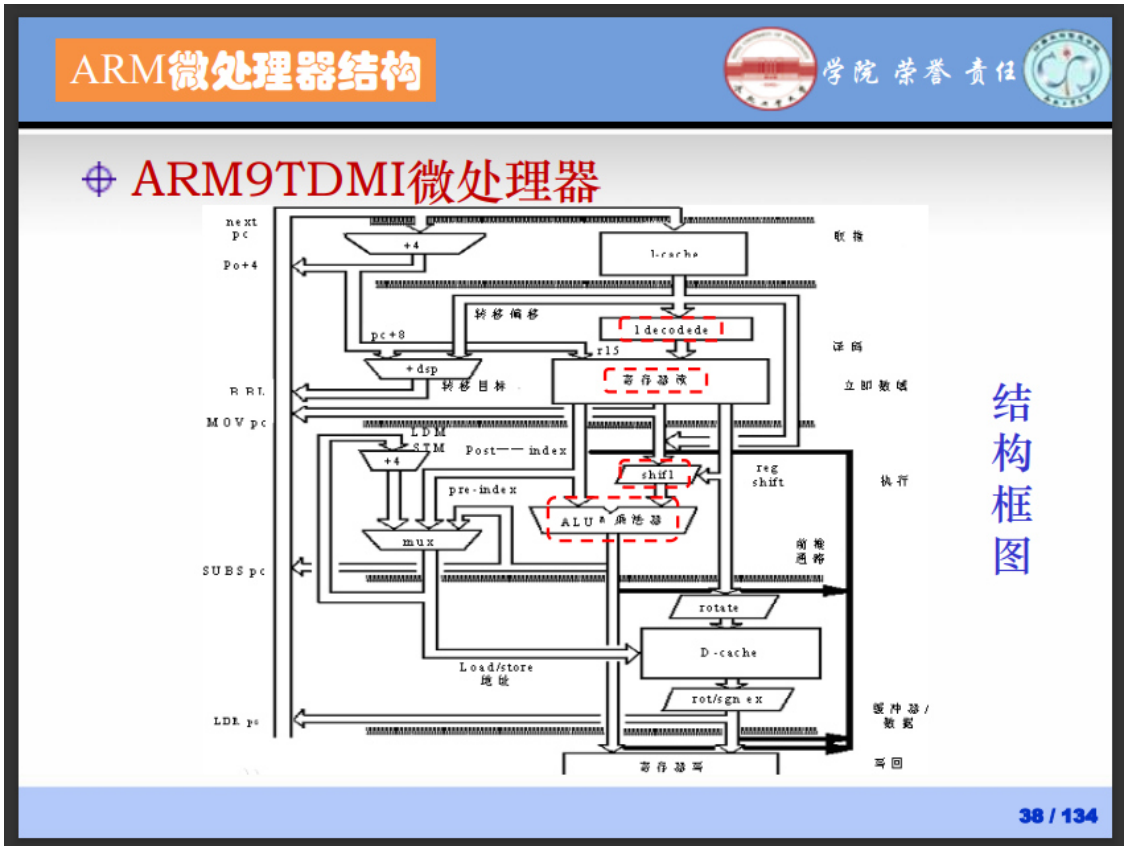
ARM处理器系列	特　点	应用场合
ARM7	提供Thumb16位压缩指令集和Embedded ICE软件调试方式，适用于更大规模的SoC设计。	应用于多媒体和嵌入式设备，包括Internet设备、网络和调制解调器设备，以及移动电话、PDA等无线设备。
ARM9	兼容ARM7系列；提供更加灵活的设计。	主要应用于发动机管理、仪器仪表、安全系统和机顶盒等领域。
ARM9E	含有DSP指令集；在ARM7处理器内核的基础上使用了Jazelle增强技术。	应用于下一代无线设备、数字消费品、成像设备、工业控制、存储设备和网络设备等领域。
ARM10E	使用向量浮点(VFP)单元VFP10提供高性能的浮点解决方案，从而极大提高了处理器的整型和浮点运算性能。	主要应用于视频游戏机和高性能打印机等场合。
Xscale	提供全性能、高性价比、低功耗的解决方案，支持16位Thumb指令并集成数字信号处理（DSP）指令。	主要应用于手提式通讯和消费电子类设备。

但比如，嵌入式系统是什么，这样的该背的还是要背的。

不同之处还有 ARM 芯片不仅包括内核，还包括其他东西、

ARM 微处理器内部 例如到底有什么样的总线，有哪些东西，如何连接。这都不是我们的重点。

比如类似于这样的，不会去考察



2.4 节 运行模式及其特点要搞清楚 基于模式 有两种状态 ARM 和 Thumb 状态要清楚

还有寄存器..

虚拟地址到物理地址的转换

七种异常类型对应着五种异常模式，异常处理的过程，异常所对应的返回指令，每个异常对应的返回的位置和

第三章

寻址方式 结合着寻址方式 常见的指令 对外围设备控制的时候给的指令要知道是干嘛的

第四章

4.3 连接 尤其是地址线的连接 特殊情况 8 位对 8 位... 顺次连接 怎么错位连接的

认识嵌入式存储器的类型 特点 以及一些连接的方法

第五章

所有寄存器 每一位的定义会给我们 因此不需要背任何一个寄存器的结构。

SMR 特殊寄存器内部的分析 给了寻址需求分析 能会分析 而不是第一位是什么 第二位是什么 那不需要，都会给我们

我们需要做的是分析每一位到底是 0 还是 1

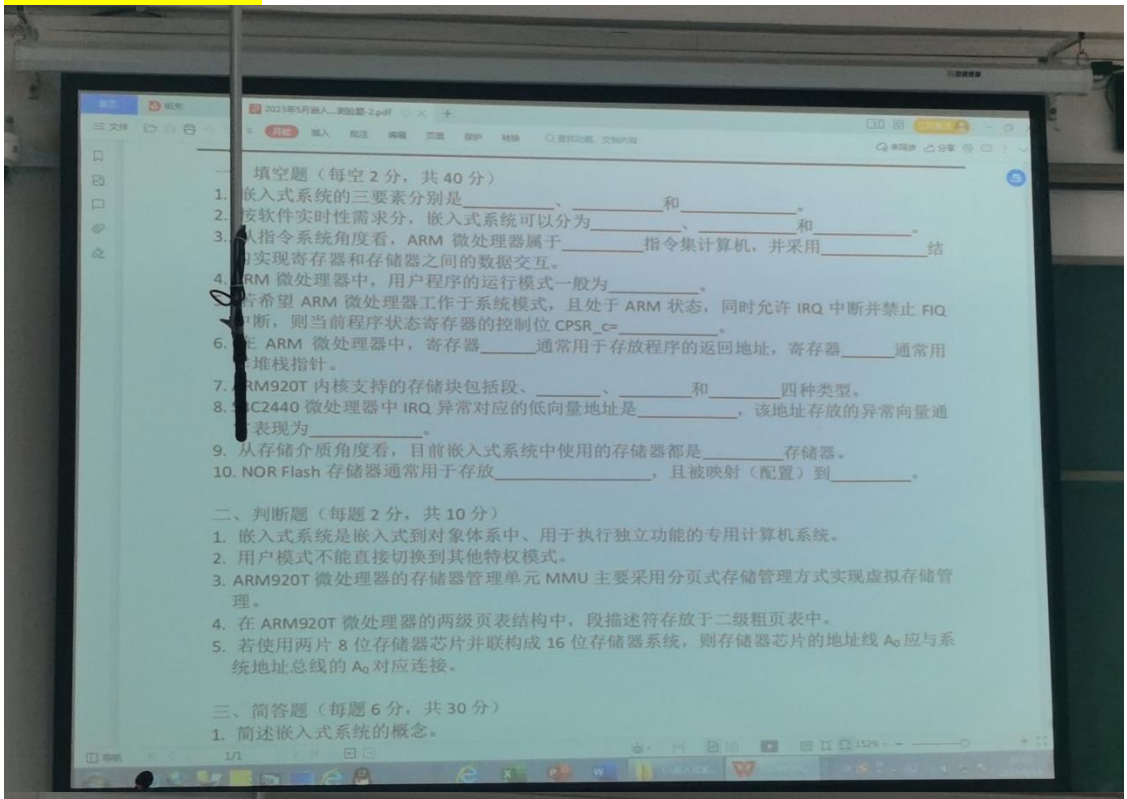
每一小节相应的例子是最好的复习回顾，包括分析，完全搞懂例子是复习外设最大的帮助

第六章

不会做太多要求，不会出现在大题里，会出现在小题里。不会让你写 makefile 或者 shell。。

期末考了 bootloader 两种启动方式。。

课堂测试卷子-21 级



一、填空题（每空 2 分，共 40 分）

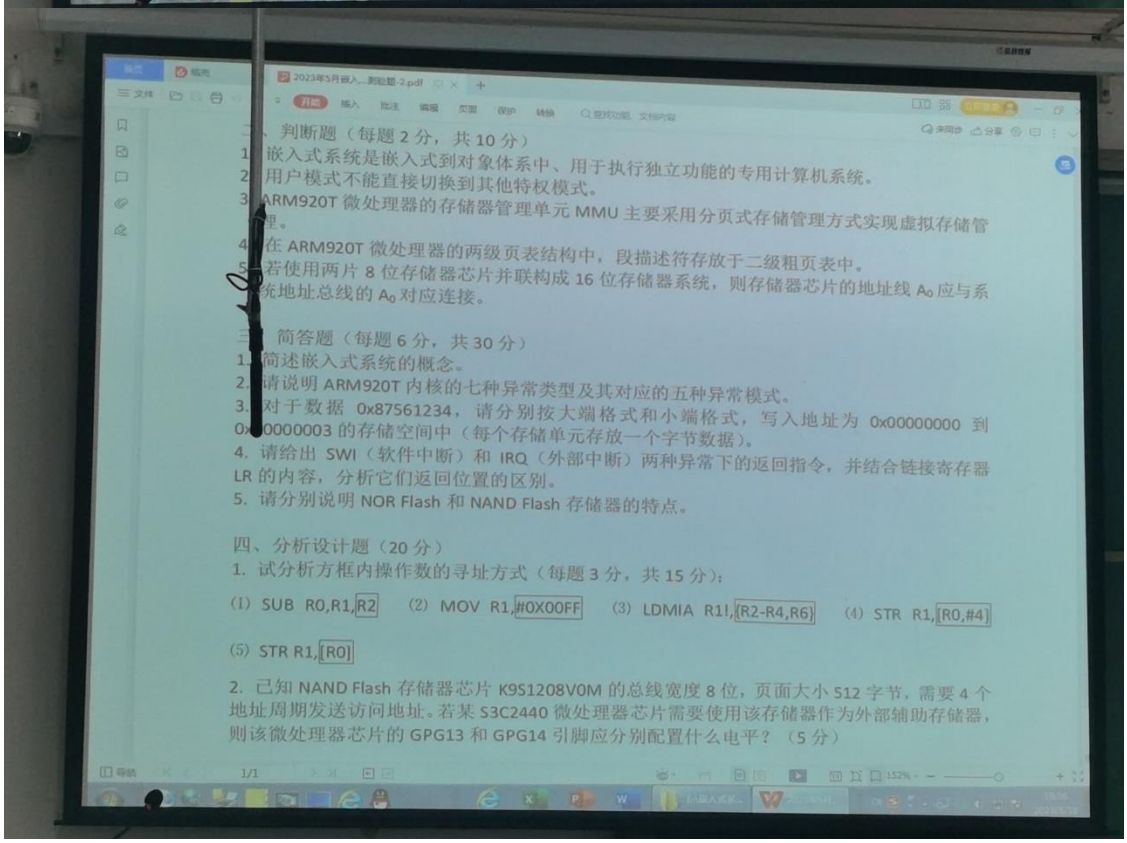
1. 嵌入式系统的三要素分别是_____、_____和_____。
2. 按软件实时性需求分，嵌入式系统可以分为_____、_____和_____。
3. 从指令系统角度看，ARM 微处理器属于_____指令集计算机，并采用_____结构实现寄存器和存储器之间的数据交互。
4. ARM 微处理器中，用户程序的运行模式一般为_____。
5. 若希望 ARM 微处理器工作于系统模式，且处于 ARM 状态，同时允许 IRQ 中断并禁止 FIQ 中断，则当前程序状态寄存器的控制位 CPSR_c=_____。
6. 在 ARM 微处理器中，寄存器_____通常用于存放程序的返回地址，寄存器_____通常用于堆栈指针。
7. ARM920T 内核支持的存储块包括段、_____、_____和_____四种类型。
8. S3C2440 微处理器中 IRQ 异常对应的低向量地址是_____，该地址存放的异常向量通常表现为_____。
9. 从存储介质角度看，目前嵌入式系统中使用的存储器都是_____存储器。
10. NOR Flash 存储器通常用于存放_____，且被映射（配置）到_____。

二、判断题（每题 2 分，共 10 分）

1. 嵌入式系统是嵌入式到对象体系中、用于执行独立功能的专用计算机系统。
2. 用户模式不能直接切换到其他特权模式。
3. ARM920T 微处理器的存储器管理单元 MMU 主要采用分页式存储管理方式实现虚拟存储管理。
4. 在 ARM920T 微处理器的两级页表结构中，段描述符存放于二级粗页表中。
5. 若使用两片 8 位存储器芯片并联构成 16 位存储器系统，则存储器芯片的地址线 A_0 应与系统地址总线的 A_0 对应连接。

三、简答题（每题 6 分，共 30 分）

1. 简述嵌入式系统的概念。



二、判断题（每题 2 分，共 10 分）

1. 嵌入式系统是嵌入式到对象体系中、用于执行独立功能的专用计算机系统。
2. 用户模式不能直接切换到其他特权模式。
3. ARM920T 微处理器的存储器管理单元 MMU 主要采用分页式存储管理方式实现虚拟存储管理。
4. 在 ARM920T 微处理器的两级页表结构中，段描述符存放于二级粗页表中。
5. 若使用两片 8 位存储器芯片并联构成 16 位存储器系统，则存储器芯片的地址线 A_0 应与系统地址总线的 A_0 对应连接。

三、简答题（每题 6 分，共 30 分）

1. 简述嵌入式系统的概念。
2. 请说明 ARM920T 内核的七种异常类型及其对应的五种异常模式。
3. 对于数据 0x87561234，请分别按大端格式和小端格式，写入地址为 0x00000000 到 0x00000003 的存储空间中（每个存储单元存放一个字节数据）。
4. 请给出 SWI（软件中断）和 IRQ（外部中断）两种异常下的返回指令，并结合链接寄存器 LR 的内容，分析它们返回位置的区别。
5. 请分别说明 NOR Flash 和 NAND Flash 存储器的特点。

四、分析设计题（20 分）

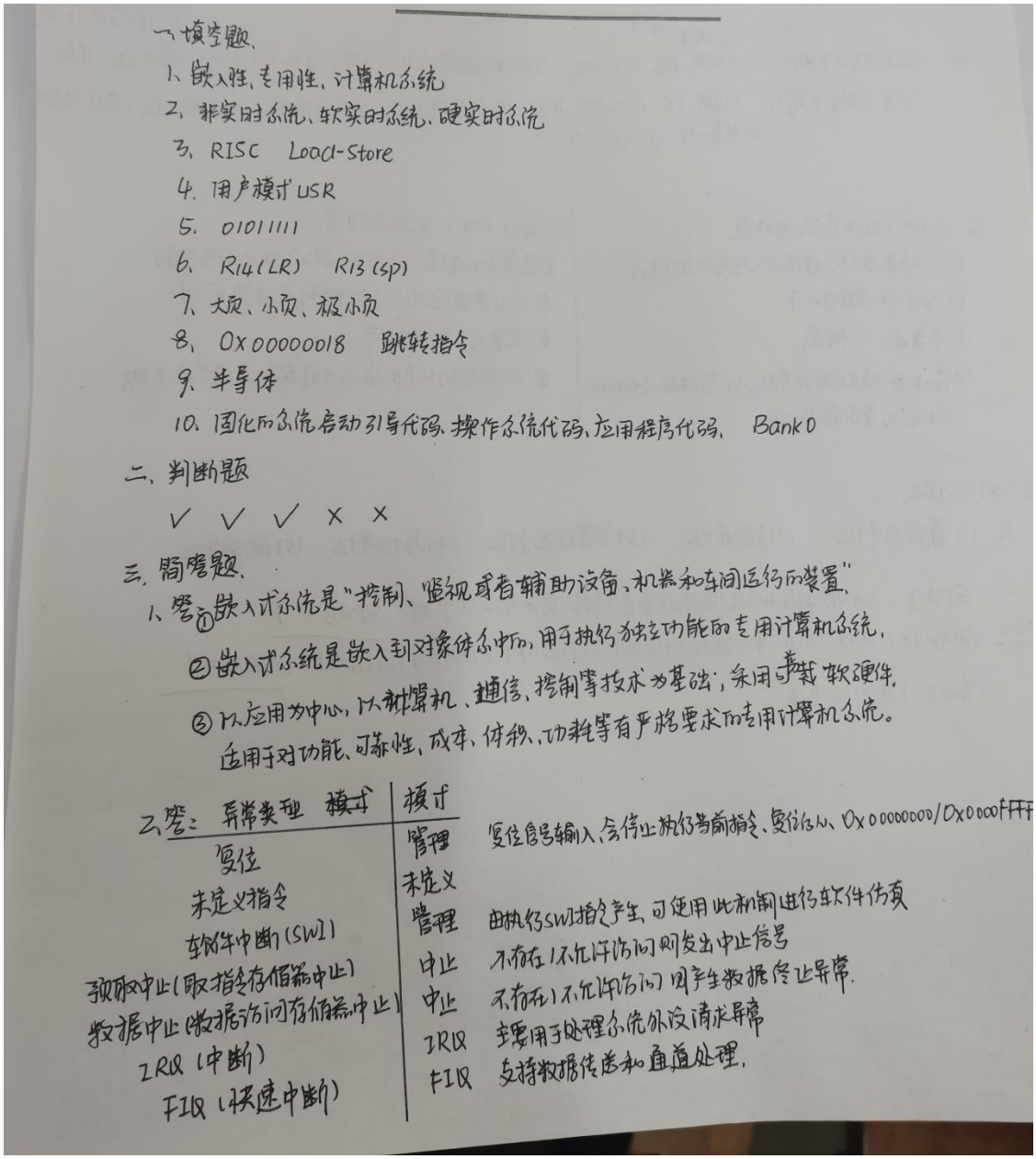
1. 试分析方框内操作数的寻址方式（每题 3 分，共 15 分）：

(1) SUB R0,R1,[R2]　(2) MOV R1,[#0X00FF]　(3) LDMIA R1!,{R2-R4,R6}　(4) STR R1,[R0,#4]

(5) STR R1,[R0]

2. 已知 NAND Flash 存储器芯片 K9S1208V0M 的总线宽度 8 位，页面大小 512 字节，需要 4 个地址周期发送访问地址。若某 S3C2440 微处理器芯片需要使用该存储器作为外部辅助存储器，则该微处理器芯片的 GPG13 和 GPG14 引脚应分别配置什么电平？（5 分）

21 级答案-45 班



一、填空题

1、嵌入性、专用性、计算机系统

2、非实时系统、软实时系统、硬实时系统

3、RISC Load-Store

4、用户模式 USR

5、01011111

6、R14(LR) R13(sp)

7、大页、小页、极小页

8、0x00000018 跳转指令

9、半导体

10、固化的系统启动引导代码、操作系统代码、应用程序代码、 Bank0

二、判断题

✓ ✓ ✓ × ×

三、简答题

1、答：①嵌入式系统是"控制、监视或者辅助设备、机器和车间运行的装置"

②嵌入式系统是嵌入到对象体系中的，用于执行独立功能的专用计算机系统。

③以应用为中心，以计算机、通信、控制等技术为基础，采用可裁剪软硬件，适用于对功能、可靠性、成本、体积、功耗等有严格要求的专用计算机系统。

2、答：

异常类型	模式	
复位	管理	复位信号输入、会停止执行当前指令、复位后从 0x00000000/0xFFFF0000 处取指令
未定义指令	未定义	
软件中断(SWI)	管理	由执行SWI指令产生，可使用此机制进行软件仿真
预取中止(取指令存储器中止)	中止	不存在/不允许访问则发出中止信号
数据中止(数据访问存储器中止)	中止	不存在/不允许访问则产生数据停止异常
IRQ(中断)	IRQ	主要用于处理系统外设请求异常
FIQ(快速中断)	FIQ	支持数据传送和通道处理

3. 大端格式

地址	数据
0x00000000	87
0x00000001	56
0x00000002	12
0x00000003	34

小端格式

地址	数据
0x00000000	34
0x00000001	12
0x00000002	56
0x00000003	87

4. SWI(软件中断)：(ARM下) LR状态：PC-4 LR
MOVS PC, R14_svc. 它的返回值是将LR寄存器的值加上4作为返回地址，重新返回被打断的指令

IRQ(外部中断)：SUBS PC, R14_irq, #4，它的返回值是将LR寄存器的值作为返回地址，到下条指令
LR状态：PC'-4(=PC) LR

5. Nor Flash存储器特点：
① 访问速度快，程序可以在其内直接运行
② CPU可以直接访问
③ 容量小、价格高
④ 常作为内部或者外部ROM，存放Boot-loader程序，引导系统启动.

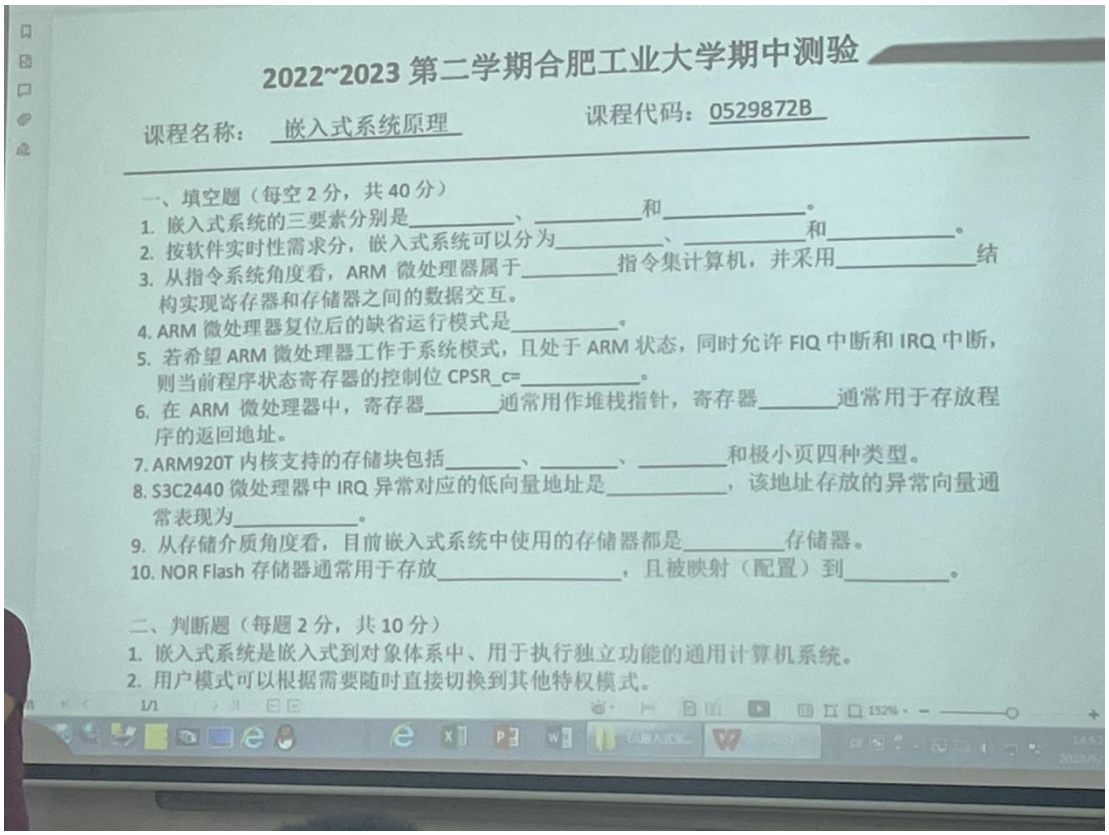
Nand Flash存储器特点：
① 速度相对慢，程序必须放入RAM才能执行
② CPU需要通过专门的控制器才能访问
③ 容量大、价格经济
④ 需要借助内部RAM才能实现引导系统启动.

四、分析设计题.

1、(1)寄存器寻址 (2)立即寻址 (3)多寄存器寻址 (4)基址寻址 (5)间接寻址.

2、GPG13：NAND Flash存储器容量选择位：页=512字节时. GPG13=1
NCON=0 GPG14：NAND Flash存储器地址周期选择位 4个地址周期(NCON=0) GPG14=1
添上：均为高电平.

计科 123+物理网



2022~2023 第二学期合肥工业大学期中测验

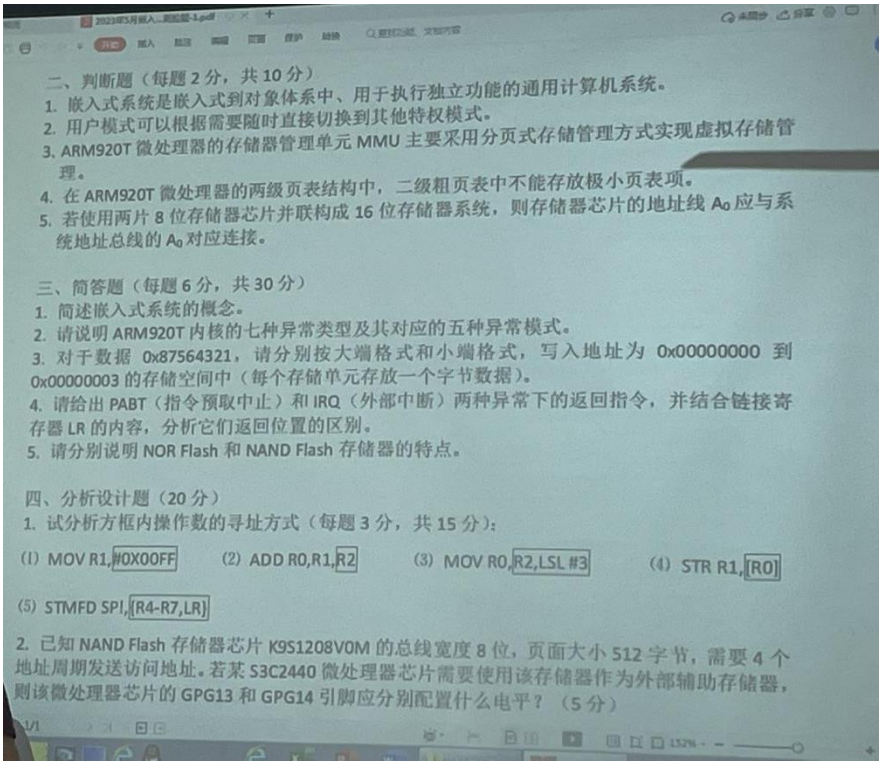
课程名称：嵌入式系统原理　　课程代码：0529872B

一、填空题（每空 2 分，共 40 分）

1. 嵌入式系统的三要素分别是_____、_____和_____。
2. 按软件实时性需求分，嵌入式系统可以分为_____、_____和_____。
3. 从指令系统角度看，ARM 微处理器属于_____指令集计算机，并采用_____结构实现寄存器和存储器之间的数据交互。
4. ARM 微处理器复位后的缺省运行模式是_____。
5. 若希望 ARM 微处理器工作于系统模式，且处于 ARM 状态，同时允许 FIQ 中断和 IRQ 中断，则当前程序状态寄存器的控制位 CPSR_c=_____。
6. 在 ARM 微处理器中，寄存器_____通常用作堆栈指针，寄存器_____通常用于存放程序的返回地址。
7. ARM920T 内核支持的存储块包括_____、_____、_____和极小页四种类型。
8. S3C2440 微处理器中 IRQ 异常对应的低向量地址是_____，该地址存放的异常向量通常表现为_____。
9. 从存储介质角度看，目前嵌入式系统中使用的存储器都是_____存储器。
10. NOR Flash 存储器通常用于存放_____，且被映射（配置）到_____。

二、判断题（每题 2 分，共 10 分）

1. 嵌入式系统是嵌入式到对象体系中、用于执行独立功能的通用计算机系统。
2. 用户模式可以根据需要随时直接切换到其他特权模式。



二、判断题（每题 2 分，共 10 分）

1. 嵌入式系统是嵌入式到对象体系中、用于执行独立功能的通用计算机系统。
2. 用户模式可以根据需要随时直接切换到其他特权模式。
3. ARM920T 微处理器的存储器管理单元 MMU 主要采用分页式存储管理方式实现虚拟存储管理。
4. 在 ARM920T 微处理器的两级页表结构中，二级粗页表中不能存放极小页表项。
5. 若使用两片 8 位存储器芯片并联构成 16 位存储器系统，则存储器芯片的地址线 A_0 应与系统地址总线的 A_0 对应连接。

三、简答题（每题 6 分，共 30 分）

1. 简述嵌入式系统的概念。
2. 请说明 ARM920T 内核的七种异常类型及其对应的五种异常模式。
3. 对于数据 0x87564321，请分别按大端格式和小端格式，写入地址为 0x00000000 到 0x00000003 的存储空间中（每个存储单元存放一个字节数据）。
4. 请给出 PABT（指令预取中止）和 IRQ（外部中断）两种异常下的返回指令，并结合链接寄存器 LR 的内容，分析它们返回位置的区别。
5. 请分别说明 NOR Flash 和 NAND Flash 存储器的特点。

四、分析设计题（20 分）

1. 试分析方框内操作数的寻址方式（每题 3 分，共 15 分）：

(1) MOV R1,#0X00FF　(2) ADD R0,R1,R2　(3) MOV R0,R2,LSL #3　(4) STR R1,[R0]

(5) STMFD SP!,{R4-R7,LR}

2. 已知 NAND Flash 存储器芯片 K9S1208V0M 的总线宽度 8 位，页面大小 512 字节，需要 4 个地址周期发送访问地址。若某 S3C2440 微处理器芯片需要使用该存储器作为外部辅助存储器，则该微处理器芯片的 GPG13 和 GPG14 引脚应分别配置什么电平？（5 分）

计算机与信息学院期中测验

专业 _____ 班级 _____ 学号 2021218151 姓名 王晓雪婷 成绩 _____

日期 _____ 教师 _____ 共 _____ 页 第 _____ 页

一、填空题

1、嵌入性、专用性、计算机系统

2、非实时系统、软定时系统、硬定时系统

3、RISC、Load-Store

4、管理模式

5、0001 1111

6、R13、R14

7、段、大页、小页

8、0x0000 0018，一条跳转指令

9、半导体

10、固化的系统启动引导代码、操作系统代码、应用程序代码、Bank0

二、判断题

1、X 2、X 3、√ 4、√ 5、√

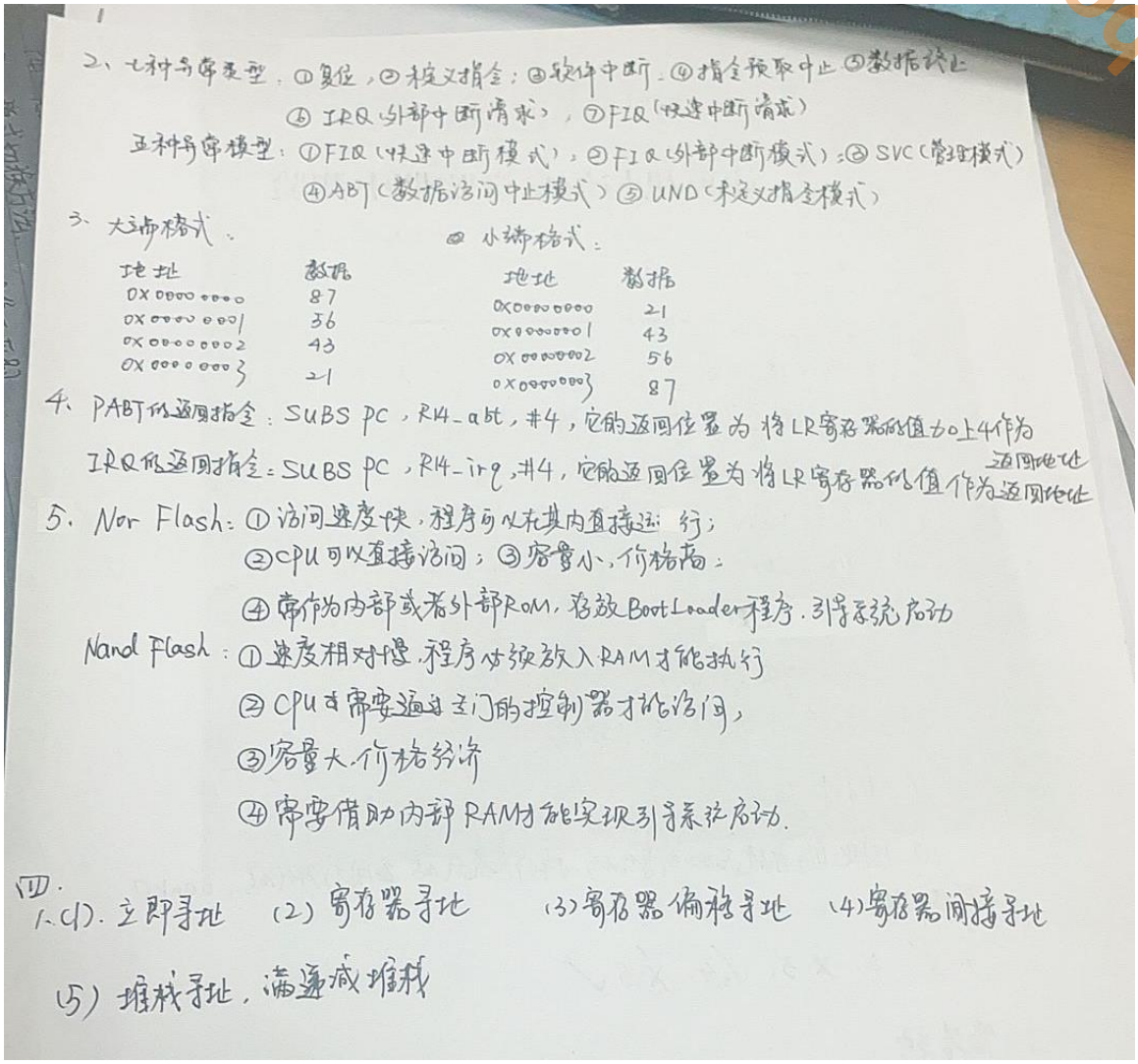
三、简答题

1、①嵌入式系统是"控制、监视或者辅助设备、机器和车间运行的装置"。——IEEE

②嵌入式系统是嵌入到对象体系中的、用于执行独立功能的专用计算机系统。

③以应用为中心，以计算机、通信、控制等技术为基础，采用可剪裁软硬硬件，适用于对功能、可靠性、成本、体积、功耗等有严格要求的专用计算机系统。

判断第五题是错的



2、都是高电平

期末考了 nor flash 引导 rom ； 看门狗的用途；大题：寻址方式、定时器

第一章 绪论

1、嵌入式系统的特点（与桌面应用系统的区别）

①嵌入式系统中运行的任务是**专用而确定**的；桌面通用系统需要支持大量、需求多样的应用程序；②嵌入式系统往往对**实时性**提出较高的要求；③嵌入式系统对**可靠性**要求高；④嵌入式系统**有功耗约束**；⑤嵌入式系统**内核小，可用资源少**；⑥嵌入式系统的**开发需要专用工具和特殊方法**。

2、嵌入式系统的结构

一般由**嵌入式处理器、外围硬件设备、嵌入式操作系统**（可选）和**用户的应用软件系统**四部分组成。

3、分类

按照**嵌入式微处理器字长**：4 位、8 位、16 位、32 位、64 位系统；

按照**软件实时性**：非实时系统、软实时系统、硬实时系统（嵌入式实时系统可分为强实时性：*μs~ms* 级，一般实时：*ms~s* 级，弱实时型：*s* 级以上）；

按**系统的复杂程度**：小型系统、中型系统、复杂系统；

按**系统形态**：芯片级系统、板级系统、设备级系统。

第二章 ARM 微处理器与编程模式

1、冯氏结构与哈佛结构区别

冯氏结构：程序和数据存放在**同一存储器**的不同地址；**顺序**执行指令；

取指令（**或数据**）→分析指令→执行指令。

哈佛结构：程序和数据存放在**不同存储器**中；**并行**执行指令；

取指令（**和数据**）→分析指令→执行指令。

2、两种指令系统

类别	CISC	RISC
指令系统	指令数量多	较少
执行时间	有些指令执行时间很长	较短
编码长度	可变，1-15 字节	固定，通常为 4 字节
寻址方式	寻址方式多样	简单寻址
操作	可对存储器和寄存器进行算术和逻辑操作	只能对寄存器进行算术和逻辑操作，Load/Store 体系结构
编译	难以用优化编译器生成高效的目标代码程序	采用优化编译技术，生成高效目标代码程序

❗：冯氏和哈佛结构指的是计算机的存储器结构，与指令系统没有关系！

3、命名规则

ARM 体系结构版本的命名规则：ARM Vn | variants | ×(variants)

Vn：对应指令集版本号。variants 与×(variants)分别代表支持和不支持的变种类型。

ARM 微处理器（内核）命名规则：**ARM{x}{y}{z}{T}{D}{M}{I}{E}{J}{F}{-S}**

x：family number；y：内存系统（2/4/6）；z：内存大小（0/2/6）；

T：支持 Thumb 指令集；D：支持片上调试；J：Java 加速器；S 可综合，提供 VHDL 或 verilog 语言设计文件。

4、运行模式

USR 用户模式、FIQ 快速中断模式、IRQ 外部中断模式、SVC 管理模式、ABT 数据访问中止模式、UND 未定义指令格式、SYS 系统模式。

1 + 6：用户模式 + 特权模式。2 + 5：用户、系统模式 + 异常模式。

特权模式可以访问所有系统资源，可以任意进行处理器模式切换。**用户模式**不能访问某些被保护的系统资源，不能直接进行处理器模式切换，需要产生异常处理，在异常处理中进行处理器模式切换。**用户模式与系统模式**不能由异常进入，使用完全相同的寄存器组。

模式切换可以通过程序切换，也可以当特定异常出现时进入相应模式。启动时的模式切换过程为：SVC（复位后缺省模式）→多种特权模式变换（主要完成各模式堆栈设置）→用户模式的运行模式。

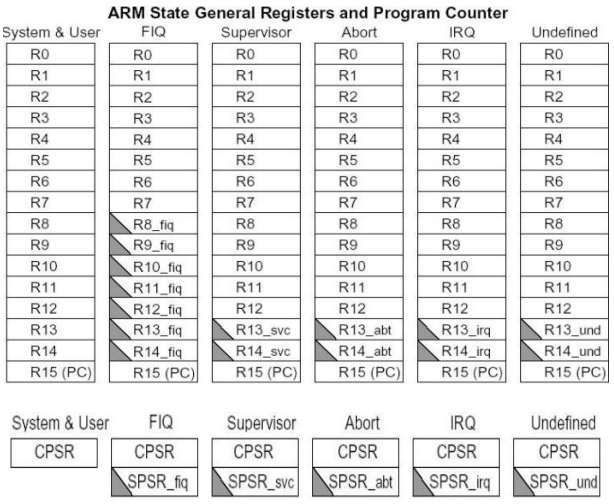
5、工作状态

ARM 状态（RISC）执行 32 位字对齐的 ARM 指令；Thumb 执行 16 位半字对齐的 Thumb 指令。

：两种状态之间的切换不影响处理器的模式或寄存器的内容。

	ARM	Thumb
协处理器、信号量指令	有	无
乘加/64 位乘指令	有	无
访问 CPSR/SPSR 指令	有	无
跳转指令	三种跳转指令都可以指定条件	除 B 指令外都是无条件跳转，跳转范围小
数据处理指令	三个操作数	两个操作数，访问寄存器 R8-R15 受限
单寄存器加载/存储指令	无限制	只能访问寄存器 R0-R7
批量寄存器加载/存储指令	无限制	只能访问 R0-R7 的子集

模式切换：操作数寄存器 Rm 状态位（位 0）为 0 时，执行 BX 指令可以切换到 ARM 状态，为 1 时切换到 Thumb 状态；Thumb 状态下发生异常（先切换到 ARM 再切换到 Thumb），异常处理返回时自动切换到 Thumb 状态；处理器进行异常处理时若 PC 指针放在异常链接寄存器 LR 中，并从异常向量地址开始执行程序，处理器会切换到 ARM 状态。



6、寄存器组织

31 个 32 位通用寄存器。**未分组寄存器**：R0-R7，都可以用；**分组寄存器**中 R8-R12 每个

寄存器对应两个不同的物理寄存器 usr 和 fiq，R13-R14 每个对应 6 个，其中一个 USR 与 SYS 共用，R13 也叫堆栈指针，R14 叫子程序链接寄存器；**程序计数器** PC。

6 个 32 位状态寄存器。**当前程序状态寄存器 CPSR**：可以在任何运行模式下被访问。**备份程序状态寄存器 SPSR**，USR 和 SYS 下无效。

状态寄存器 31-28 条件码标志，27-8 保留不用，7-0 控制位，其中 4-0 这 5 位反映运行模式。0b10000 USR，0b10001 FIR，0b10010 IRQ，0b10011 SVC，0b10111 ABT，0b11011 UND，0b11111SYS。

控制位中 I 和 F 是中断禁止位，决定中断能否发生，为 1 表示对应中断禁止；T 为工作状态为，反应处理器运行状态，1 在 Thumb，0 在 ARM，其由外部引脚决定，程序不得更改。

7、存储格式

大端：高字节存在低地址，低字节存在高地址；

小段：高字节存在高地址，低字节存在低地址。

存储器管理单元 MMU：将虚拟地址转换成物理地址，是以**存储块**为单位进行的。包括分段式、分页式和段页式存储管理。分页式把虚拟存储空间分成一个个固定大小的**页**，查询放在主存中的**页表**实现虚拟地址到物理地址的转换，由于查询页表代价很大，因此采用**快表技术**（TLB）提高地址转换速率。

基于 TLB 的地址转换过程：

①微处理器访问主存时，首先根据虚拟地址在 TLB 中查找对应的转换条目，如果存在则直接转换；②如果不存在则继续在主存的页表中查找，并把查找结果添加到 TLB 中，称为 TLB 更新；③在添加时，如果 TLB 已满，可根据一定的淘汰算法进行替换。

四种存储块：段，1MB（一级页表中段 1MB 对齐）；大页，64KB；小页，4KB；极小页，1KB。

地址转换过程：

①根据给定的虚拟地址查找一级页表获得一个条目；②若该条目为段描述符则返回物理地址，转换结束（单步搜索）；③若该条目为粗页表目录项，则继续利用虚拟地址查找二级粗页表，获得大页描述符或小页描述符，返回物理地址，转换结束（两步搜索）；④若该条目为细页表目录项，则继续利用虚拟地址查找二级细页表，获得极小页描述符，返回物理地址，转换结束（两步搜索）；⑤其他情况出错。

8、异常处理

异常处理过程：

当异常发生时，系统执行完当前指令后，需要保存处理器的当前状态(执行现场)，然后将跳转到相应的异常处理程序处执行。当异常处理程序执行完成后，程序应恢复所保存的执行现场，从而返回到发生异常时的下一条指令继续执行。处理器允许多个异常同时发生，它们将会按固定的优先级进行处理。

七种异常类型：复位（SVC）、未定义指令（UND）、软件中断（SWI）（SVC）、指令欲取中止（ABT）、数据访问中止（ABT）、IRQ 中断、FIQ 中断。

异常向量：异常程序的入口地址。**异常向量地址**：异常出现后，强制从异常类型对应的固定地址开始执行程序。**异常向量表**是由七个异常向量及其异常处理函数跳转关系组成的表，每个异常向量占 4 个字节，表现为一条跳转指令。

ARM 下的异常处理：

①将下一条指令的地址存入相应链接寄存器 LR，以便程序在处理异常返回时能从正确的位置重新开始执行；②将 CPSR 复制到相应的 SPSR 中；③根据异常类型，强制设置 CPSR 的运行模式位；④强制 PC 从相关的异常向量地址取下一条指令执行，从而跳转到相应的异常处理程序处。同时设置中断禁止位。

异常类型	异常返回位置	需设置的 PC 值	对应的返回指令
复位	无返回	无	无
未定义指令	未定义指令后第一条指令地址	LR	MOVS PC, LR
软中断 SWI	SWI 指令后第一条指令地址	LR	MOVS PC, LR
指令欲取中止	本欲取中止指令地址	LR - 4	SUBS PC, LR, #4
IRQ	断点后第一条指令地址	LR - 4	SUBS PC, LR, #4
FIQ	断点后第一条指令地址	LR - 4	SUBS PC, LR, #4
数据欲取中止	本数据中止指令地址	LR - 8	SUBS PC, LR, #8

第三章 ARM 指令集与汇编程序设计

1、寻址方式

（1）寄存器寻址，操作数在寄存器中。　　MOV R1，R2；R2→R1

（2）立即寻址，操作数是数据本身。　　MOV R0，#0xFF000；0xFF000→R0

（3）寄存器偏移寻址，操作数先移位再使用。 MOV R0，R2，LSL #3；R2×8→R2

LSL：逻辑左移，低位补 0；LSR 逻辑右移，高位补 0；ASR 算术右移，高位不变；

ROR：循环右移；RRX；带进位循环右移，C→R[31]，R[0]→C。

（4）寄存器间接寻址，操作数的地址在寄存器中。 LDR R0，[R2]；[R2]→R0

（5）基址寻址，基址寄存器中的内容与指令给出的偏移量相加形成有效地址。基址寄存器不能为 R15（联系 LDR、STR、LDM、STM 等指令）

LDR R2，[R3，#8]；[R3 + 8]→R2

LDR R2，[R3]，#2；[R3]→R2，R2 + 2→R2

（6）多寄存器寻址，一次可以不传送超过 16 个寄存器。

LDM**IA** R1，{R2-R4，R6}；[R1]→R2，[R1 + 4]→R3，…，[R1 + 12]→R6

STM**IA** R1，{R2-R4，R6}；[R2]→R1，[R3]→R1 + 4，…，[R6]→R1 + 12

LDM**IA** R1!，{R2-R4，R6}；[R1]→R2，…，[R1 + 12]→R6，[R1 + 16]→R1

IA 表示每次执行完加载/存储指令后 R1 按**字**长度增加。

（7）堆栈寻址，用专门的寄存器（堆栈指针 SP）指向一块存储区域，SP 指向的存储单元为栈顶。根据**入栈**时 SP 的变化以及 SP 指向的内容可分为：**满递增（FA）、慢递减（FD），空递增（EA）、空递减（ED）**。注意与多寄存器寻址中的 IA、IB、DA、DB 区分！

（8）相对寻址，基址寻址的一种变通。PC 提供的基准地址加上指令中操作数表示的偏移量得到有效地址 EA。可用于子程序的调用和返回。

2、ARM 指令集

属于 RISC，只能处理寄存器中的数据，对存储器访问要通过专门的加载/存储指令完成。

（1）指令格式：

<操作码>{执行条件，如 EQ}{S：是否影响 CPSR} <目标寄存器 Rd>，<第一个操作数寄存器 Rd>{第二个操作数}

EQ：Z=1，相等；NE，Z=0，不相等；

无符号数：CS≥，HI＞，LS≤，CC＜；

有符号数：GE≥，GT＞，LE≤，LT＜；

（2）跳转指令

使用跳转指令可以实现前后 32MB 的跳转，直接向 PC 写入跳转地址值可以实现 4GB 空间的任意跳转。三种跳转指令为（每种跳转指令后都可以加条件码）：

跳转指令 B label；label→PC

带返回的跳转指令 BL label；PC - 4→LR，label→PC

带状态切换的跳转指令 BX Rm；Rm→PC，切换处理器状态，Rm[0]为 0 为 ARM。

还有一种是带返回和状态切换的跳转指令 BLX Rm；PC - 4→LR，Rm→PC，切换处理器状态。

（3）数据处理指令（数据传送指令、算式/逻辑运算指令、比较指令）

只能对寄存器操作（MOV R0，#0x00ff ×），可选择使用 S 来影响标志位。比较指令不用 S 也会影响状态标志。

数据传送 MOV、MVN；比较 CMP（-）、CMN（+）、TST（&）、TEQ（=）；

算术运算 ADD、ADC、SUB、SBC、RSB（逆向减）、RSC（带进位逆向减）；

逻辑运算 AND、ORR、EOR、BIC（清除 1 对应的位置）；

乘法指令 MUL（32 位）、MLA（32 位乘加）、（U/S)(MUL/MLA)L（64 位无/有符号数乘/乘加指令）。

（4）存储器访问指令（单寄存器加载/存储指令、多寄存器加载/存储指令、寄存器和存储器交换指令）

使用**单寄存器加载指令**（LDR/STR）加载**数据**到 **PC** 寄存器，可实现程序的**跳转**功能。

顺序是：操作码 + 条件 + B/H + T

LDR Rd，add；存储器中 32 位字数据→寄存器，读；加 **B 表示字节加载**，同时高 24 位清零，**H 表示无符号半字数据**，高 16 位清零，若 B/H 前有 S 表示带符号，高位补符号位（后边数据的第一位），**T 表示用户模式**加载，USR 下无效。

STR Rd，add；寄存器→存储器，写。与 LDR 相比，加 B 或者 H 的话高位不变。

LDRSB R1，[R0，R3]；将 R0+R3 地址上的字节读到 R1，高 24 位用符号位拓展

LDRSH R1，[R9]；将 R9 地址上半字数据读到 R1，高 16 位用符号位拓展

LDRH R1，[R2]，#2；将 R2 地址半字数据读到 R1，高 16 位用符号位拓展，R2 自加 2

STRH R1，[R0，#2]!；将 R1 数据低 2 字节数据保存到 R0 + 2 地址中，然后 R0 自加 2

多寄存器加载/存储指令：LDM/STM{cond}<模式>　Rn!，reglist{^}，允许一条指令传送 **16 个寄存器**的任何子集，主要用途是**现场保护、数据复制、参数传递**等。Rn 不能为 R15，！表示最后的地址要写回更新 Rn，**reglist 中必须从小到大**排列，^表示不允许在 USR/SYS 下使

用。

八种模式：IA、IB、DA、DB（前/后地址加/减 4）、FA、FD、EA、ED。**数据复制**操作要先设置好**数据指针和目标指针**，然后使用**多寄存器寻址指令** LDMIA/STMIA（后缀是前四个）进行读取和存储；**堆栈操作**要先设置**堆栈指针**，然后使用**堆栈寻址指令** STMFD/LDMFD（后缀是后四个）实现堆栈操作。堆栈必须成对匹配使用。

LDMIA R0!，{R3 - R9}；加载 R0 地址上的多字数据到 R3 - R9 中，R0 值更新

STMIA R1!，{R3 - R9}；将 R3 - R9 的数据存储到 R1 指向的地址上，R1 值更新

STMFD SP!，{R0 - R7，**LR**}；现场保存，将大括号中的入栈

LDMFD SP!，{R0 - R7，**PC**}；恢复现场，异常处理返回

注意 LDMIA 和 STMIA 的顺序正好和 LDR 和 STR 相反。

寄存器与存储器交换指令 SWP{sond}{B} Rd，Rm，[Rn]；内存单元→Rd，Rm→内存单元。有 B 交换字节数据，否则是 32 位字数据，Rn 是要进行数据交换的存储器地址，**Rn 不能与 Rd 和 Rm 相同**。

SWP R1，R1，[R0]；将 R1 的内容与 R0 指向的存储单元的内容进行交换

SWPB R1，R2，[R0]；将 R0 指向的存储单元的内容读取一字节数据到 R1 中，高 24 位清零，R2 的低八位数据写入该存储单元

（5）杂项指令

MRS{cond} Rd，psr；读状态寄存器指令，CPSR/SPSR→Rd，**Rd 不允许为 R15**

MSR{cond} psr_fields，#inmmed_8r/RD；写状态寄存器指令

MSR CPSR_c，#0xD3；CPSR 低八位为 0xD3，切换到管理模式

MSR CPSR_cxsf，R3；CPSR=R3

（6）伪指令

不属于 ARM 指令集，为了编程方便而定义，等效为一条或多条 ARM 指令。常用的为：ADR、ADRL、LDR、EQU、NOP。

ADR/ADRL{cond} reg，expr；将基于 PC 或者寄存器相对偏移的**地址值**读取到目标寄存器中。

LDR{cond} reg，**=**[expr | label_expr]；加载 32 位立即数或者地址值到指定寄存器中。指令位置到程序标号的偏移量必须小于 4KB，与 ARM 的 LDR 相比多了一个=。

name EQU expr{,type}；将一个数值或者寄存器名赋给制定的符号名。expr 是基于寄存器的地址值、程序中的标号、32 位地址常量或 32 位**常量**，expr 为 32 位常量是可以使用 type 指示 expr 数据的类型，取值为 CODE32、CODE16 和 DATA。

abcd EQU 2 / label + 16 / 0x1c，CODE32；符号 abcd 的值为 2 / label +16 / 绝对地址 0x1c 且此处为 ARM 指令。

NOP；空操作，编译时被替换为类似 MOV R0，R0 这样的无用指令。

第四章 嵌入式存储器系统

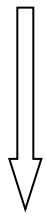
1、嵌入式存储器分类

按照用途：主存储器（位于微处理器芯片内部，存储速度快，容量有限）、外部存储器（位于主板外，存取速度慢，容量大，成本低）；

按存储介质：半导体存储器（采用大规模集成电路技术，使用最广泛）、光盘存储器（利用介质材料的光效应）、磁表面存储器；

按存取方式分类：随机存取存储器 RAM、顺序存取存储器 SAM、直接存取存储器 DAM、只读存储器 ROM、闪速存储器 Flash Memory（非易失性，包括 Nor Flash 和 Nand Flash）。

2、结构



内部寄存器（存储数据，加速指令的执行速度）

芯片内的高速缓存 Cache（存储将要执行的指令和数据，提高指令的执行速度）

RAM 等主存储器（存储将要执行的指令和数据）

ROM、Nor Flash 等主存储器（存储固化的系统程序，运行初始化程序等）

外部存储器（存储大量的数据）

3、存储器系统空间分布（以 S3C2440 为例）

支持**编程设定**大小端模式。可寻址外部存储空间为 1GB，其被分为 8 个 Bank，每个 128MB。Bank0 的数据位宽只能为 16 或 32 位（引脚电平决定），其他的还可以为 8 位（编程设定）；前 6 个可以外接 ROM、SRAM，后两个可以多接一个 SDRAM。Bank6 和 7 的容量可以**编程设定**，但两者必须相等。要知道 1GB 的存储空间是怎么分配的。

4、**Nor Flash 和 NAND Flash**

Nor Flash：读写速度快，程序可以在其内直接运行；CPU 可以直接访问；容量小，价格高；存固件，常作为内部或外部 ROM 存放 Boot-Loader 程序引导系统启动；通常配置到 Bank0。

NAND FLash：速度相对慢，程序必须放入 RAM 才能执行；CPU 需要通过专门的控制器才能访问；容量大，价格经济；需要借助内部 RAM 才能实现引导系统启动；支持自动启动引导。

使用 NAND Flash 为引导 ROM 的启动流程：首先将 **4kB** 的 BootLoader 程序加载到 NAND Flash 中，系统上电后，Nand Flash 控制器会自动的把 NAND Flash 的前 4K 数据搬移到 4K 内部 RAM 中，并把 0x00000000 设置为内部 RAM 的起始地址，CPU 从内部 RAM 的 0x00000000 位置开始启动。主程序会放到 SDRAM 中，在系统启动后开始执行。

5、外部存储器芯片连接

本质上是**数据总线、地址总线和控制总线**的连接。特别是地址总线：

存储器芯片的地址引脚	S3C2440 的地址引脚（8 位数据总线）	S3C2440 的地址引脚（16 位数据总线）	S3C2440 的地址引脚（32 位数据总线）
A0	A0	A1	A2
A1	A1	A2	A3
…	…	…	…

第五章 常用外围设备与接口

1、通用输入输出端口 GPIO（以 S3C2440 为例）

130 个 GPIO 引脚，分为 9 组：GPA~GPJ。**没有 GPI！**每个端口具有多种功能，具体哪一种可在主程序运行之前编程设置对应的**控制寄存器**。

GPA：23 位**输出**端口； GPB：11 位输入/输出端口； GPC：16 位输入/输出端口；

GPD：16 位输入/输出端口；GPE：16 位输入/输出端口； GPF：8 位输入/输出端口；

GPG：16 位输入/输出端口；GPH：11 位输入/输出端口； GPC：13 位输入/输出端口。

一些功能关键字的含义：

nFRE、nFWE 表示读写；nGCS1~5 表示片选；ALE 和 CLE 表示锁存。

要求：编程实现使用 GPIO 端口点亮发光二极管。

2、中断系统

中断的作用：并行处理、实时处理、故障处理。借助**中断控制器**，接收并管理 60 个中断请求信号。有 32 个一级中断源（实际 45 个），15 个二级中断源。

中断控制器的功能：外部中断请求信号管理、中断模式设定、中断请求信号标记、中断屏蔽设定、中断优先级管理、中断服务标记。通过 **16 个特殊功能寄存器** SFR 实现中断管理功能，与外部中断相关的 SFR 有 9 个，与一级中断源相关的有 5 个，与二级中断源相关的有 2 个。

外部中断控制寄存器（EXTINI0~2）：用于**设定**外部中断请求信号的触发方式；

外部中断滤波寄存器：用于**设定** 8 个外部中断请求信号（EINT16~23）的滤波器的时钟信号来源和滤波宽度，因为为了保证微处理器能有效检测到外部中断请求信号，信号的有效逻辑电平至少要保持 40ns；

外部中断屏蔽寄存器：用于**设定** 20 个外部中断请求信号（EINT4~23）是否允许中断；

外部中断挂起寄存器：用于**标记** 20 个外部中断请求信号（EINT4~23）是否触发，供中断服务程序读取和判断。

源挂起寄存器：用于**标记** 32 个一级中断源的中断请求信号是否触发，通常需要人工**清除置位**以保证能收到同一中断源的下一次中断请求；

中断模式寄存器：用于**设定** 32 个一级中断源的中断模式，同一时刻仅有一个中断源能在 FIQ 下处理，应将最紧迫的中断源设置为 FIQ。**1 表示按 FIQ** 处理；

中断屏蔽寄存器：用于**设定** 32 个一级中断源是否被允许中断，1 不服务；

中断控制器借助**优先级裁决器（仲裁判决器）**实现对 32 个一级中断源的优先级判决，一级裁决器有 6 个，二级裁决器有 1 个。REQ0 总是最高优先级，REQ5 总是最低优先级，而 REQ1~4 的优先级类型有两种，静态优先级优先级固定，设定好后不能更改，动态优先级优先级循环，处理后自动降为最低。优先级寄存器用于**设定** 32 个一级中断源在 **IRQ 模式**下的优先级顺序。

中断挂起寄存器用于**标记** 32 个一级中断源的中断请求是否即将或者正在被微处理器服务。**位于优先级逻辑之后**，同一时刻又有 1 位被置 1，表示其在所有已触发的中断中优先级最高且未被屏蔽，向微处理器发出 IRQ 信号，仅对 IRQ 模式的中断有效。通常需要通过写入数据来人工**清除置位**，写 1 表示清除，0 不变。

中断偏移寄存器用于标记 INTPND 中置“1”位对应中断源的偏移量，值代表即将或者正在被微处理器服务的**中断源号**。仅对 IRQ 模式的中断有效。

二级中断源相关的 SFR 包括次级源挂起寄存器和中断次级屏蔽寄存器。

外部中断处理流程：

外部中断 EINT0-EINT3 发生后，源挂起寄存器 SRCPND 相应位置 1；EINT4-EINT23 发生后外部中断挂起寄存器 EINTPEND 相应位置 1。如果没有被外部中断屏蔽寄存器 EINTMASK 屏蔽，那么 SPCPND 相应位 EINT4-7 或 EINT8-23 置 1。如果中断源没有被中断屏蔽寄存器 INTMSK 屏蔽，那么会根据中断模式寄存器 INTMOD 判断中断模式，如果是 FIQ，那么进入 FIQ 模式；如果是 IRQ，这时会经过优先级寄存器 PRIORITY 选出一个优先级高的中断源并把中断挂起寄存器 INTPND 相应位置 1，然后进入 IRQ 让 CPU 处理。

3、时钟与定时部件

（1）时钟控制模块

三种内部时钟信号：内核时钟 FCLK、总线时钟 HCLK 和 I/O 接口时钟 PCLK。

FCLK = FOUT = 2m×Fin /（P × 2^S）.Fin 为输入时钟源的频率，m、p、s 计算得到。

时钟分频控制 CLKDIVN[2:1]：

00：HCLK = FCLK / 1　01：HCLK = FCLK / 2

10：HCLK = FCLK / 4 当 CAMDIVN[9] = 0；HCLK = FCLK / 8 当 CAMDIVN[9] = 1

11：HCLK = FCLK / 3 当 CAMDIVN[9] = 0；HCLK = FCLK / 6 当 CAMDIVN[9] = 1

CLKDIVN[0]:

0：PCLK = HCLK / 1　1：PCLK = HCLK / 2

MPLLCON 中 MDIV 为[19:12]，PDIV 为[9:4]，SDIV 为[1:0]，m/p/s 在其基础上加 8/2/0。

要求：能根据寄存器中的值计算时钟频率。

（2）定时器

5 个 16 位递减定时器，功能：定时、计数、**脉宽调制（PWM）**。每个定时器都有一个可以生成 5 种不同分频信号（1/2、1/4、1/8、1/16 和外部时钟 TCLK）的时钟分频器，定时器 0 和 1 共用一个 8 位预分频器，2、3、4 共用另外一个。

实际执行减 1 操作的是 TCNTn，**不允许程序对其直接访问**，而应该先把计数初值写入 TCNTBn，每次定时开始时，**值会被复制到 TCNTn 中**。TCNTn 减到 0 时，可根据 TCON 的“自动重载”标志决定是否赋值 TCNTBn 到 TCNTn 中以开启下次定时操作。可以通过程序读取 TCNTOn 的值间接获取 TCNTn 的动态计数值。

内部设有**双缓冲功能**，在当前定时操作中如果设定新的定时器值，仅当当前定时操作执行完毕后才有效，非立即有效。

工作流程：①设初值；②Start，开始计数直到 TCNTn = TCMPn，此时 TOUTn 反转；③TCNTn 继续减 1，等于 0 时再次反转并触发中断；④若 TCON 寄存器中设置为自动重载，则 TCNTBn 的值给 TCNTN，TCMPBn 的值给 TCMPn，重复过程①~④。

要求：编程实现使用定时器实现不同占空比下蜂鸣器的发声。

（3）看门狗定时器 WDT

一种用于当噪声或系统错误引起故障时恢复控制器操作的定时器。本质上是一个 16 位内部定时器，计数器值为 0 时通过触发中断服务激活 **128 个 PCLK 时钟周期的内部复位信号**。

WTDAT 相当于 TCNTBn，WTCNT 相当于 TCNTn，但是 WTCNT 必须要被设定一个初值。

第六章 嵌入式 Linux 编程基础

1、嵌入式层次划分

应用系统（应用程序的程序集）、**文件系统**（文件存放在磁盘等存储设备上的组织方法）、**SHELL**（用户与内核交互的一种接口）、**内核**（运行程序和管理硬件设备的核心）。

2、交叉编译

在一种平台上编译出能在另一种平台（体系结构不同）上运行的程序，用来编译这种程序的编译器叫做交叉编译器。

3、文件系统

Windows 与 Linux 文件系统的区别：

①目录结构不同。Windows 有多个树根，Linux 只有一个；②文件属性不同。Windows 文件系统只负责文件存储，Linux 文件系统管理所有软硬件资源。

Linux 有三种文件类型：**普通文件**（又分文本文件、二进制文件）、**目录文件**（存储一组相关文件位置、大小等与文件有关的信息）、**设备文件**。

“.”代表该目录自己，“..”代表父目录。对于根目录，“.”和“..”都代表自己。

4、Linux 内核结构

五个主要子系统：

进程调度：负责控制进程访问 CPU，Linux 采用基于优先级的进程调度方法。程序是一组指令的有序集合，是静态的，而进程是具有独立功能的程序的一次运行过程，是动态的。Linux 提供了**抢占式**的多任务模式，由调度程序决定什么时候停止一个进程的运行，以便其他进程能够得到执行机会，这个强制的挂起动作称为抢占。**进程间通信**：在不同进程之间传播或交换信息，通过在内核中开辟一块缓冲区，不同进程通过缓冲区实现数据交换。**内存管理**：使多个进程安全地共享内存。**虚拟文件系统**：为文件系统提供了一个通用的接口，提供了内核中的一个抽象功能。**网络接口**，在设计上遵循模拟协议本身的分层体系结构。

5、终端命令

[root@localhost/home]# ：root 为用户名，localhost 为计算机名，home 为当前工作目录，# 表示用户角色（超级用户是#，其他用户是$）。

增加用户 useradd：useradd sjp，增加名字为 sjp 的用户；切换用户 su：su -root，切换到 root 用户，并将其环境变量同时代入；关机 shutdown：shutdown now，立即关机；

拷贝 cp： cp /home/test /tmp，将/home 目录下的 test 文件 copy 到/temp 目录下

cp -r /home/lky /tmp/，将/home 目录下的 lky 目录 copy 到/tmp 目录下

移动或更名 mv、删除 rm（-r 表示目录）、创建目录 mkdir、改变工作目录 cd、查看当前路径 pwd、查看目录 ls、打包压缩 tar、解压缩 unzip、改变访问权限 chmod。

6、编译调试

Makefile 文件的结构：

```
目标：依赖项列表
（Tab 缩进）命令
```

目标：欲生成的目标文件；依赖项：生成目标需要的文件。是一种**树形结构**。

调试：

分为**静态调试和动态调试**。静态调试在程序编译阶段查错并修正错误，主要为语法错误；动态调试在程序运行阶段查错并修正错误，主要是算法错误、输入错误。

第七章 SHELL 编程与 BootLoader

1、SHELL 编程基础

要求：能根据程序或一行代码写输出

变量赋值：变量名=变量值，**等号**两端不能留空格，如果**字符串**两边有空格必须加引号。

引用：$变量名或者${变量名}，变量名一个字符用方式一，多于一个字符用建议用方式二。

echo arg：在屏幕上显示由 arg 制定的字符串；

read ABC：从键盘输入内容为变量 ABC 赋值；

双引号内的字符除了$、倒引号‘和\仍保留其功能外其余字符视为普通字符，$表示变量替换，倒引号表示命令替换，\为转义字符；

单引号括起来的字符不论是什么都视为普通字符；

倒引号括起来的字符串被解释为命令行，以它的标准输出结果取代整个倒引号部分。

until 循环条件为假时执行循环命令。函数调用中如果函数内部需要使用传递给函数的参数一般用$0、$1、…、$n 以及$#、$*、$@等特殊变量，$0 是执行脚本的文件名，$1 是传递给函数的第一个参数，$#是传递给函数的参数个数，$*和$@为传递给函数的所有参数。

echo $[10+1]　　输出 11

echo “$[2+3],$HOME”　输出 5,/root

echo $[2<<3],$[8>>1]　　输出 16,4

echo $[2>3],$[3>2]　输出 0,1

echo “$HOME, that is your directory.” 输出/root, that is your directory.

echo ‘$HOME, that is your directory.’ 输出$HOME, that is your directory.

```
for foo in bar fud 43
    do
        echo $foo
    done
exit 0
```

```
输出：
    bar
    fud
    43
```

2、BootLoader 两阶段过程

要求：能简答解释 BootLoader 分成哪两部分以及作用

运行模式包括下载模式（用于调试和版本修改）以及启动加载模式（自主模式，是 BootLoader 的正常工作模式）。

BootLoader 通常分为 **stage1** 和 **stage2** 两大部分，采用**分级载入机**制。

stage1 是依赖于 CPU 体系结构的代码，例如设备初始化代码等，通常用汇编语言来实现，以达到短小精悍的目的；

stage2 通常用 C 语言来实现，可以实现更复杂的功能，而且代码会具有更好的可读性和可移植性。