嵌入式系统期末复习

考试题型:

一、 填空（10 空/10 分）

二 、选择 （10 题/10 分）

三 、简答题 (5 题/30 分)

四 、读指令回答问题（15 空/30 分）

①写出指令寻址方式 ② 写出完成下列操作的指令或指令完成的操作（6 题）

③根据指令操作，在相应的寄存器填写相应的内容（14 空）

五 、编程题（2 题/20 分）

第一章

1 、嵌入式系统的定义：以应用为中心、以计算机技术为基础，软硬件可裁剪，适应应用系统对功能、可靠性、成本、体积、功耗严格要求的专用计算机系统。

2 、嵌入式系统的组成:

(1)硬件层: ①嵌入式微处理器②存储器③通用设备接口和 I/O 接口

(2)中间层

(3)软件层①操作系统（驱动程序）②应用系统

第二章

1、 嵌入式微处理器体系结构

(1)冯·诺依曼体系结构和哈佛系统结构

冯·诺依曼体系结构也称普林斯顿结构，是一种将程序指令存储器和数据存储器合并在一起的存储器结构。在冯·诺依曼体系结构的计算机中，程序和数据共用一个存储空间，程序指令存储地址和数据存储地址指向同一个存储器的不同物理位置，采用统一的地址及数据总线，程序指令和数据宽度相同。PC 只负责提供程序执行所需要的指令或数据，而不决定程序流程，要控制程序流程，则必须修改指令。

采用冯·诺依曼体系结构的微处理器有: Intel 公司的 8086 系列 CPU、ARM 公司的 ARM 系列微处理器、MIPS 公司的 MIPS 系列微处理器。

哈佛体系结构是一种将程序指令存储和数据存储分开的体系结构，是一种并行的体系结构，主要特点是将程序和数据存储在不同的存储空间中，即程序存储器和数据存储器是两个相互独立的存储器，每个存储器独立寻址独立访问，有较高的执行效率。

采用哈佛体系结构的微处理器有: 所有的 DSP 处理器、ATMEL 公司的 AVR 系列和 ARM 公司的 ARM9、ARM10 和 ARM11 系列微处理器等。

(2) CISC 与 RISC

CISC 架构的设计目的是要用最少的机器语言指令来完成所需的计算任务。

```
MUL  addrA ,addrB
```

RISC 架构要用软件在指定各个操作步骤。

```
MOV A,addrA;  MOV B,addrB;

MUL A,B;            STR addrA
```

类别	CISC	RISC
指令系统	指令数量很多	较少，通常少于100
执行时间	有些指令执行时间很长，如整块的存储器内容拷贝；或将多个寄存器的内容拷贝到存贮器	没有较长执行时间的指令
编码长度	编码长度可变，1-15字节	编码长度固定，通常为4个字节
寻址方式	寻址方式多样	简单寻址

操作	可以对存储器和寄存器进行算术和逻辑操作	只能对寄存器对行算术和逻辑操作， 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，生成高效的 目标代码程序

(3) 信息存储的字节顺序

①小端字节顺序存储法：低字节数据存放在内存低地址处，高字节数据存放在内存高地址处。

②大端字节顺序存储法：高字节数据存放在内存低地址处，低字节数据存放在内存高地址处

2 、嵌入式操作系统的分类：

嵌入式操作系统可以从系统的类型、响应时间和软件结构进行分类。

Ⅰ、 按系统的类型分类

① 商用系统：功能强大，性能稳定，应用范围相对较广，相应辅助软件工具齐全；价格通常较贵

典型代表：风河公司（WindRiver)的 VxWorks、微软公司的 Windowns CE、Palm 公司的 Palm OS 等。

② 专用系统：一些专业厂家为本公司产品特制的嵌入式操作系统。

③开源系统：成本低、开源、性能优良、资源丰富、技术支持强等优点

典型代表：μC/OS 和各类嵌入式 Linux 系统

Ⅱ、按响应时间分类

(1) 实时操作系统(RTOS)：对响应时间严格要求

① 硬实时系统：响应时间超过规定时限，将导致灾难性后果的实时系统。

② 软实时系统：响应时间超过规定时限，但后果可以忍受的实时系统。

根据响应时间在微秒、毫秒和秒级的不同，可分为强实时、准实时和弱实时三种。

(2) 非实时操作系统：在响应时间上没有严格要求，

如：分时操作系统

Ⅲ 、按软件结构分类

(1) 单体结构（Monolithic Structure）

(2) 分层结构（Layered Structure）

(3) 微内核结构（Microkernel Model）

3 、嵌入式系统的设计方法

嵌入式系统的设计一般有 5 个阶段：①需求分析。②体系结构设计。③硬件/软件设计。 （填空题）

④系统集成。⑤系统测试

★系统需求分析

确定设计任务和设计目标，并提炼出设计规格说明书，作为正式设计指导和验收的标准

★体系结构设计

描述系统如何实现所述的功能和非功能要求（处理器的选定、操作系统的选定、开发环境的选定）

★硬件/软件设计

基于体系结构，对系统的软件、硬件进行详细设计

★系统集成

把系统的软件、硬件和执行装置集成在一起进行调试，发现并改革单元设计过程中的错误。

★系统测试

对设计好的系统进行测试，看其是否满足规格说明书中给定的要求。

4 嵌入式系统的开发模式：一般采用宿主机/目标机的开发模式

第三章

1、 什么是 ARM?

ARM（ Advanced RISC Machines）既可以认为是一个公司的名字，它是一家微处理器行业的知名企业，1990 年

成立于英国剑桥；也可以认为是一类微处理器的通称；也可以认为是一种技术的名字。

2 、ARM 命名原则

| ARMv | n | variants | x (variants) |

ARM 体系的 7 个基本版本：v1～v7

分成 4 个部分：

ARMv—固定字符，即 ARM version

n 指令集版本号，n=1～7。

variants —扩充(用字母表示)。

x (vatiants) —排除 x 之后指定的扩充。

例：ARMv5TxM： 表示 ARM 指令集版本 5，支持 T 扩充，不支持 M 扩充。

扩展：T—支持 Thumb 指令　　D—支持片上调试　　M—支持快速乘法　I—支持嵌入式跟踪调试

E—支持增强型 DSP 指令　　J—支持 Java 程序　　F—具备向量浮点单元 VFP

3 、ARM 体系架构

◆ 目前使用的 ARM 系列处理器中，除了 ARM7 采用冯·诺依曼体系结构，ARM9 以上的版本都采用哈佛结构。

◆ 32 位嵌入式 RISC 处理器

◆ 有大端存储格式和小端小端存储格式之分，缺省为小端存储格式。

◆ ARM 处理器支持以下数据类型：

▲ 字（Word）：字的长度为 32 位，而在 8 位/16 位处理器体系结构中，字的长度一般为 16 位，请注意区分。

▲ 半字(Half-Word)：半字的长度为 16。

▲ 字节（Byte）：各种处理器体系结构中，字节的长度均为 8 位。

4、 ARM 状态各模式下的寄存器

寄存器类别	寄存器在汇编中的名称	各模式下实际访问的寄存器						
		用户	系统	管理	中止	未定义	中断	快中断
通用寄存器和程序计数器	R0	R0						
	R1	R1						
	R2	R2						
	R3	R3						
	R4	R4						
	R5	R5						
	R6	R6						
	R7	R7						
	R8	R8						R8_fiq
	R9	R9						R9_fiq
	R10	R10						R10_fiq
	R11	R11						R11_fiq
	R12	R12						R12_fiq
	R13(SP)	R13		R13_svc	R13_abt	R13_und	R13_irq	R13_fiq
	R14(LR)	R14		R14_svc	R14_abt	R14_und	R14_irq	R14_fiq
	R15(PC)	R15						
状态寄存器	R16(CPSR)	CPSR						
	SPSR	无		SPSR_svc	SPSR_abt	SPSR_und	SPSR_irq	SPSR_fiq

所有的 37 个寄存器，分成两大类：

① 31 个通用 32 位寄存器；

② 6 个状态寄存器。

5 、处理器工作状态

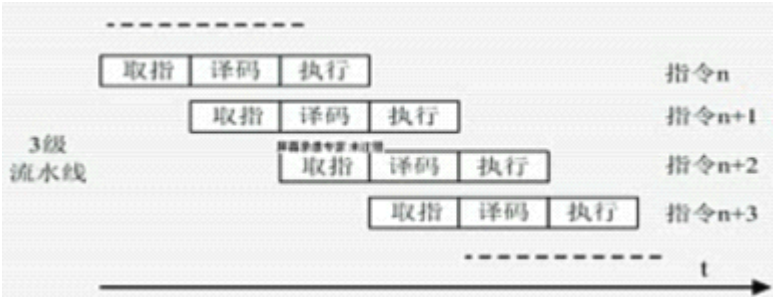
ARM 状态：32 位，这种状态下执行的是字方式的 ARM 指令；

Thumb 状态：16 位，这种状态下执行半字方式的 ARM 指令；

用 BX Rn 指令来进行两种状态的切换。

注意：两个状态之间的切换并不影响处理器模式或寄存器内容。

6、 ARM 的三级流水线（一个时间完成三步操作）



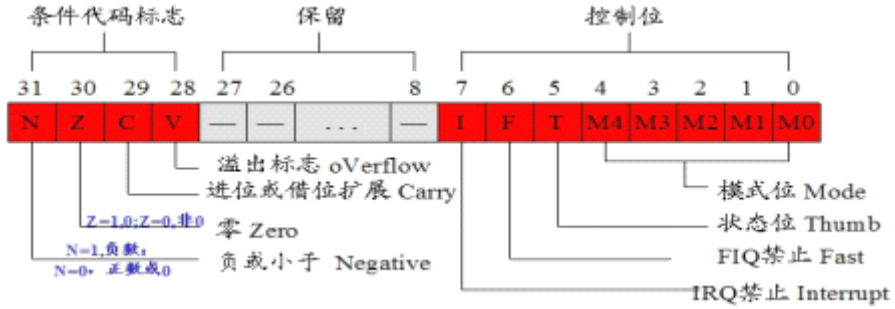
ARM 状态:　　PC 值=当前程序执行位置+4+4

Thumb 状态:　PC 值=当前程序执行位置+2+2

7 、程序状态寄存器（6 个 32 位寄存器）

ARM 体系包含一个当前程序状态寄存器（CPSR）和 5 个备份的程序状态寄存器（SPSR）

CPSR 可在任何运行模式下被访问，它包括条件码标志位、中断静止位、当前处理器模式标志位等



8、 运行模式位的具体含义 M【4:0】　　【背】

M[4:0]	模式	可见的ARM状态寄存器
10000	用户模式	R0~R14,PC,CPSR
10001	FIQ模式	R0~R7,R8_fiq~R14_fiq,PC,CPSR,SPSR_fiq
10010	IRQ模式	R0~R12,R13_irq,R14_irq,PC,CPSR,SPSR_irq
10011	SVR模式	R0~R12,R13_svc,R14_svc,PC,CPSR,SPSR_svc
10111	中止模式	R0~R12,R13_abt,R14_abt,PC,CPSR,SPSR_abt
11011	未定义模式	R0~R12,R13_und,R14_und,PC,CPSR,SPSR_und
11111	系统模式	R0~R14,PC,CPSR

9、 ARM 的处理模式

处理器运行模式	说明	备注
用户模式（USR）	正常程序执行模式	不能直接切换到其它模式
系统模式（SYS）	运行操作系统的特权任务	与用户模式类似，但具有可以直接切换到其它模式等特权
快速中断模式（FIQ）	支持高速数据传输及通道处理	FIQ异常响应时进入此模式
外部中断模式（IRQ）	用于通用中断处理	IRQ异常响应时进入此模式
管理模式（SVC）	操作系统保护模式	系统复位和软件中断响应时进入此模式
数据访问终止模式（ABT）	用于支持虚拟内存和/或内存保护	在ARM7TDMI没有支用处
未定义指令中止模式（UND）	支持硬件协处理器的软件仿真	未定义指令异常响应时进入此模式

USR 模式之外的其他模式也称为特权模式，可以完全访问系统资源，可自由改变模式，其中 FIQ、IRQ、SVC、ABT 和 UND5 种模式也被称为异常模式。

10 、异常

当正常的程序执行流程发生暂时的停止时，称之为异常。

在处理异常之前，处理器当前状态必须保留，以便在异常处理程序完成后，原来的程序能够重新执行。

(1)异常类型

表 2-3　ARM 体系结构所支持的异常

异常类型	具体含义
复位	当处理器的复位电平有效时，产生复位异常，程序跳转到复位异常处理程序处执行。
未定义指令	当 ARM 处理器或协处理器遇到不能处理的指令时，产生未定义指令异常。可使用该异常机制进行软件仿真。
软件中断	该异常由执行 SWI 指令产生，可用于用户模式下的程序调用特权操作指令。可使用该异常机制实现系统功能调用。
指令预取中止	若处理器预取指令的地址不存在，或该地址不允许当前指令访问，存储器会向处理器发出中止信号，但当预取的指令被执行时，才会产生指令预取中止异常。
数据中止	若处理器数据访问指令的地址不存在，或该地址不允许当前指令访问时，产生数据中止异常。
IRQ（外部中断请求）	当处理器的外部中断请求引脚有效，且 CPSR 中的 I 位为 0 时，产生 IRQ 异常。系统的外设可通过该异常请求中断服务。
FIQ（快速中断请求）	当处理器的快速中断请求引脚有效，且 CPSR 中的 F 位为 0 时，产生 FIQ 异常。

(2)对异常的响应

①将下一条指令的地址保存到相应连接寄存器 LR，以便程序在异常处理返回时能从正确的位置重新开始执行。

②讲 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。

(3)对异常的返回

①将连接寄存器 LR 的值减去相应的偏移量后送到 PC 中。

②将 SPSR 复制回 CPSR 中。

③若在进入异常处理时设置了中断禁止位，要在此清除。

异常	地址	进入模式
复位	0x0000 0000	管理模式
未定义指令	0x0000 0004	未定义模式
软件中断	0x0000 0008	管理模式
预取指令中止	0x0000 000c	中止模式
数据中止	0x0000 0010	中止模式
保留	0x0000 0014	保留
IRQ	0x0000 0018	IRQ
FIQ	0x0000 001c	FIQ

异常向量表

异常类型	优先级
复位	1（最高优先级）
数据中止	2
FIQ	3
IRQ	4
预取指令中止	5
未定义指令	6
SWI	7（最低优先级）

优先级

11、 ARM 指令的寻址方式

① 寻址方式分类——立即寻址

```
MOV R0,#0xFF000  ;将立即数 0xFF000 装入 R0 寄存器
```

② 寻址方式分类——寄存器寻址

```
SUB  R0,R1,R2    ;将 R1 的值减去 R2 的值，结果保存到 R0
```

③ 寻址方式分类——寄存器间接寻址

```
LDR R1,[R2] ;将 R2 指向的存储单元的数据读出  保存在 R1 中
```

④ 寻址方式分类——寄存器移位寻址

```
MOV  R0,R2,LSL #3 ;R2 的值左移 3 位，结果放入 R0,  ;即是 R0=R2×8
ANDS R1,R1,R2,LSL R3  ;R2 的值左移 R3 位，然后和      ;R1 相 “与”操作，结果放入 R1
```

⑤ 寻址方式分类——基址变址寻址

```
LDR R2,[R3,#0x0C]       ;读取(R3)+0x0C 地址上的存储单元的内容，放入 R2
```

⑥ 寻址方式分类——多寄存器寻址

```
LDMIA R1!,{R2-R7,R12}   ;将 R1 指向的单元中的数据;读出到 R2～R7、R12 中
STMIA R0!,{R2-R7,R12}   ;将寄存器 R2～R7、R12 的值;保存到 R0 指向的存储;
```

⑦ 寻址方式分类——相对寻址

```
BL   SUBR1    ;调用到 SUBR1 子程序
BEQ LOOP      ;条件跳转到 LOOP 标号处
     ...
LOOP      MOV     R6,#1
     ...
SUBR1 ...
```

~~⑧ 寻址方式分类——堆栈寻址~~

12 、ARM 指令集

(1)数据处理指令

①数据传送指令;

②算术运算指令;

③逻辑运算指令;

④比较指令。

助记符	说明	操作
MOV{cond}{S} Rd,operand2	数据传送	Rd ← operand2
MVN{cond}{S} Rd,operand2	数据非传送	Rd ← (~operand2)

助记符	说明	操作
ADD{cond}{S} Rd, Rn, opera	加法运算指令	Rd ← Rn+operand2
SUB{cond}{S} Rd, Rn, operand2	减法运算指令	Rd ← Rn-operand2
RSB{cond}{S} Rd, Rn, operand2	逆向减法指令	Rd ← operand2-Rn
ADC{cond}{S} Rd, Rn, operand2	带进位加法	Rd ← Rn+operand2+Carry
SBC{cond}{S} Rd, Rn, operand2	带进位减法指令	Rd ← Rn-operand2-(NOT)Carry
RSC{cond}{S} Rd, Rn, operand2	带进位逆向减法指令	Rd ← operand2-Rn-(NOT)Carry

MUL{cond}{S} Rd,Rm,Rs	32位乘法指令	Rd ← Rm*Rs (Rd≠Rm)
MLA{cond}{S} Rd,Rm,Rs,Rn	32位乘加指令	Rd ← Rm*Rs+Rn(Rd≠Rm)
UMULL{cond}{S}RdLo,RdHi,Rm,Rs	64位无符号乘法指令	(RdLo,RdHi) ← Rm*Rs
UMLAL{cond}{S} RdLo,RdHi,Rm,Rs	64位无符号乘加指令	(RdLo,RdHi)←Rm*Rs+(RdLo,RdHi)
SMULL{cond}{S} RdLo,RdHi,Rm,Rs	64位有符号乘法指令	(RdLo,RdHi) ← Rm*Rs
SMLAL{cond}{S} RdLo,RdHi,Rm,Rs	64位有符号乘加指令	(RdLo,RdHi)←Rm*Rs+(RdLo,RdHi)

助记符	说明	操作
CMP{cond} Rn, operand2	比较指令	标志N、Z、C、V←Rn-operand2
CMN{cond} Rn, operand2	负数比较指令	标志N、Z、C、V←Rn+operand2
TST{cond} Rn, operand2	位测试指令	标志N、Z、C、V←Rn & operand2
TEQ{cond} Rn, operand2	相等测试指令	标志N、Z、C、V←Rn ^ operand2

助记符	说明	操作
AND{cond}{S} Rd, Rn, operand2	逻辑与操作指令	Rd ← Rn & operand2
ORR{cond}{S} Rd,Rn,operand2	逻辑或操作指令	Rd ← Rn | operand2
EOR{cond}{S} Rd, Rn, operand2	逻辑异或操作指令	Rd ← Rn ^ operand2
BIC{cond}{S} Rd, Rn, operand2	位清除指令	Rd ← Rn & (~operand2)

(2)存储器访问指令（加载/存储指令）

①单个有符号/无符号字、半字、字节操作的加载/存储指令（LDR/STR）

LDR—从内存中读取单一字、 （有符号或无符号）半字或字节数据存入寄存器中

LDR R2,[R5] ;将R5指向地址的字数据存入R2

STR—将寄存器中的单一字、半字或字节数据保存到内存中。

STR R1,[R0,#0x04];将R1的数据存储到地址为R0+0x04的内存单元当中

②一次加载/存储多个寄存器内容的批量加载/存储指令（LDM/STM）

LDM{cond}<模式> Rn{!},reglist{^} ;reglist←[Rn...]，Rn回写等

STM{cond}<模式> Rn{!},reglist{^};[Rn...]←reglist,Rn回写等

▲cond：指令执行的条件;

▲模式：控制地址的增长方式，一共有8种模式;

▲！： 表示在操作结束后，将最后的地址写回Rn中;

▲Reglist:表示寄存器列表，可以包含多个寄存器，它们使用“,”隔开，如{R1,R2,R6-R9}，寄存器由小到大排列;

③寄存器与存储器内容交换的指令（SWP）

助记符	说明	操作
SWP{cond}{B} Rd,Rm,[Rn]	寄存器和存储器字（字节）数据交换	Rd ← [Rn],[Rn] ← Rm (Rn≠Rd或Rm)

B为可选后缀，若有B，则交换字节，否则交换32位字。

(3)跳转指令

助记符	说明	操作
B{cond} label	跳转指令	PC←label
BL{cond} label	带链接的跳转指令	LR←PC-4，PC←label
BX{cond} Rm	带状态切换的跳转指令	PC←label，切换处理器状态

(4)程序状态寄存器（PSR）处理指令

助记符	说明	操作
SWI{cond} immed_24	软中断指令	产生软中断，处理器进入管理模式
MRS{cond} Rd,psr	读状态寄存器指令	Rd←psr，psr为CPSR或SPSR
MSR{cond} psr_fields, Rd/#immed_8r	写状态寄存器指令	psr_fields←Rd/#immed_8r，psr为CPSR或SPSR

(6)伪指令

常用伪指令有AREA伪指令、CODE32（CODE16）伪指令、ENTRY指令、ADR伪指令、ADRL伪指令、LDR伪指令、NOP伪指令、END伪指令。

第四章

1、 μC/OS-II的主要特点（选择题）

①公开源代码。②可移植性好。③可固化。④可裁剪。⑤抢占式内核。

⑥多任务。⑦可确定。⑧任务栈。⑨系统服务。⑩稳定性与可靠性。

SEP3203 25根地址线，32根数据线

补充例题

1、编写 1500-1-2-3- ··· -50 的汇编程序

①、
```
AREA EXAMPLE,CODE,READONLY
    CODE 32
    ENTRY
    MOV R0, #50
    LDR MOV R1, =1500
    MOV R3, #0
START
    SUBS R2, R1, R0
    SUB R0, R0, #1
    CMP R3, R0
    BNE START
    END
```

②
```
AREA EXAMPLE,CODE,READONLY
    CODE 32
    ENTRY
   MOV R0, #0
   LDR R1, =1500
   MOV R3, #50
LOOP
  SUBS,R2, R1, R0
  ADD R0, R0, #1
  CMP R3, R0
  BNE LOOP
 END
```

③
```
AREA EXAMPLE,CODE,READONLY
    CODE 32
    ENTRY
  LDR R0, =1500
  MOV R1, #50
  MOV R2, #1
LOOP
  ADD R2, R2, #1
  CMP R1, R2
  BNE LOOP
  SUB R0, R0, R2
  END
```

2 参考 CPSR 寄存器中各标志的含义，使处理器处于系统模式

```
AREA EXAMPLE,CODE,READONLY
    CODE 32
    ENTRY
MRS R0, CPSR
ORR R0, R0,0X1F
MSR CPSR_C,R0
END
```

3、将寄存器中起始地址 0X10 处的四个数据移动到地址 0X20 处

```
  AREA EXAMPLE,CODE,READONLY
     ENTRY
  LDR R0, =0X10
  LDR R5, =0X20
START
  LDMIA R0!, {R1-R4}
  STMIA R5!,{R1-R4}
  END
```