微机原理与嵌入式系统期末试卷

一、填空题（20 分）

1. 冯 · 诺依曼从逻辑结构上将计算机划分为_____、_____、_____、输入设备和输出设备五大部件。

2. 在字长 $n = 8$ 的计算机中，考虑补码表示法，一个有符号整数 1000 0000b 表示的十进制真值是_____，一个无符号整数 1111 1111b 表示的十进制真值是_____。

3. _____、_____和控制相关冲突会导致指令流水线出现断流。

4. 常见存储器类型有_____、_____和光存储器等。

5. 主存块到 Cache 块的地址映射包括_____、_____、_____三种地址映射方案。

6. Cortex-M3 处理器进入异常处理子程序之前需要把_____、_____、LR、R0-R3、R12 等寄存器的值保护起来。

7. 常见的 I/O 端口编址方式有_____、_____两种。

8. 一个 8 位有符号数“1000 0001 b”经符号扩展转换至 16 位有符号数，结果为_____。

9. 总线操作周期一般可以分为：请求及仲裁、_____、_____及结束等阶段。

10. Cortex-M3/M4 中断向量表中保存的是_____。

二、单项选择题（10 分）

1. 以下属于串行接口的是（　　）

   A) I^2C

   B) PCI

   C) AHB

   D) APB

2. Cortex-M3 内部的 NVIC 支持（　　）个外部中断的管理。

   A) 127

B) 128

C) 240

D) 255

3. Cortex-M3 采用的是以下哪种堆栈 (　)

A) Full Ascending Stack

B) Full Descending Stack

C) Empty Ascending Stack

D) Empty Descending Stack

4. Cortex-M3 处理器中寄存器 R14 代表了 (　)

A) 通用寄存器

B) 链接寄存器

C) 程序计数器

D) 程序状态寄存器

5. 以下关于 Cortex-M3 描述正确的有 (　)

A) 存储器访问指令只能支持对齐数据传输

B) 特定的存储器区域可以支持位段（Bit-Band，也称作位带）操作

C) 存放在 SRAM（0x2000 0000-0x3FFF FFFF）区的代码是不允许执行的

D) Cortex-M3 不会改变代码的执行顺序，因而不需要存储器屏障指令

6. 以下关于 T32 指令集寻址方式描述不正确的是 (　)

A) 前变址寻址时，当前操作数地址由基址与偏移加和形成

B) 后变址寻址时，当前操作数地址即基址寄存器中的数值

C) 寄存器间接寻址时，把基址寄存器的值与地址偏移量相加得到操作数地址

D) 立即数寻址时，操作数本身包含在指令中

7. 以下关于微型计算机存储器系统描述不正确的有 (　)

A) 内存即主存储器

B) 缓存的访问速度快于内存

C) 闪存属于 ROM

D) 内存属于 ROM

8. 以下关于微机和嵌入式系统说法**不**正确的是 (　　)

A) 嵌入式系统的系统内核小

B) 嵌入式系统的专用性强

C) 嵌入式系统针对多任务执行优化

D) 在通用任务处理方面微机比嵌入式系统有优势

9. 以下关于指令周期说法正确的是 (　　)

A) 指令周期就是 T 周期（时钟周期）

B) 总线周期就是 T 周期（时钟周期）

C) 时钟频率越高，时钟周期就越长

D) CPU 周期也称为总线周期

10. 以下关于 AHB 说法正确的是 (　　)

A) AHB Lite 支持多主控设备

B) AHB 的突发传输不仅支持单个数据的传输，也支持数据块的传输

C) AHB 的突发传输在传输开始前必须指定传输数据的长度

D) AHB 所传输的数据本身和数据的地址需要在同一个时钟周期推送到总线上

三、判断题（10 分，打✓或✗）

1. Cortex-M3 处理器内核采用了冯 · 诺依曼结构的三级流水线。(　　)

2. Cortex-M3 处理器工作在线程模式下时，代码一定是非特权的。(　　)

3. Cortex-M3 中某些特殊寄存器只有特权访问等级才能访问。(　　)

4. 在单主控设备、多从控设备的情形下是**不**需要总线仲裁的。(　　)

5. AHB 所传输的数据及其地址必须在同一个时钟周期推送到总线上。(　　)

6. 冯 · 诺依曼结构和哈佛结构是两种不同的 ISA。(　　)

7. Cortex-M3 系列处理器支持 Thumb-2 指令集。(　　)

8. Cortex-M3 中 MOV 指令可将数据从一个寄存器送到存储器的特定位置。(　　)

9. Cortex-M3 中异常的类型号越小，该异常所对应的优先级就越高。(　　)

10. 存储器字扩展时，各芯片不同位数据线分别与数据总线 $D_7 \sim D_0$ 位并联。(　　)

四、简答题（24 分）

1. 作为一种性能指标，MIPS 能否客观反映计算机的运行速度？为什么？

2. 简述机器指令、微程序、微指令的联系。

3. 简要对比 SRAM 和 DRAM。

4. 为什么 AMBA 总线中没有定义电气特性和机械特性？

5. 接口电路的输入需要用缓冲器，而输出需要用锁存器。为什么？

6. 简述 Crotex-M 处理器的中断优先级分组机制。

五、程序设计题（16 分，每小题 8 分）

1. 阅读以下代码并回答问题（提示：汇编指令可参考附录 1）。

```
        MOV R0, #76
        MOV R1, #-1
        CMP R0, R1
        MOVHI R2, #100    ①
        MOVLS R2, #50     ②
LOOP B LOOP
```

(1) 这段代码的功能是？

(2) 执行这段代码后 R2 寄存器里完整的内容是什么（注意 R2 位数，写出 16 进制数值）？

(3) 如果代码①和②分别改为 MOVGT R2,#100 和 MOVLE R2,#50。那么执行完这段代码后 R2 寄存器里完整的内容是什么（写出 16 进制数值）？

2. 汇编程序中调用 C 函数实现求两个整数相加的和（汇编程序和 C 函数要有完整代码）。

六、综合设计题（20 分）

在一个 STM32 点亮 LED 的应用中，部分程序代码如附录 2 所示。请回答以下问题。

1. 简述 GPIO_Configuration 函数对 I/O 接口配置的步骤。

2. 分析该程序，LED 分别连接在哪些 I/O 引脚上？请画出一个简明硬件连接图；根据此连接图，当引脚输出高电平时，是点亮还是熄灭 LED？

3. 分析循环点亮 LED 代码，补充相应的注释 (1) 和 (2)。

4. 可以用定时器实现精确循环（点亮）功能。已知系统时钟为 72MHz，采用定时器 TIM2 产生周期为 500ms 的定时时间间隔（控制 LED 的亮灭）。请简述定时器配置的主要步骤，并补充完整定时函数的空余部分 (3) 和 (4)。

附录 1：ARM 汇编指令相关信息

CMP 指令

寄存器内数值与立即数比较的汇编语法为：“CMP[cond] [q]<Rd>,<Rn>”，该指令将更新标志位 APSR.N、APSR.Z、APSR.C、APSR.V。CMP 指令类似于 SUB 指令（如“SUB R2,R1”指令将 R2-R1 的结果存入 R2），只不过 CMP 指令仅更新标识位，但不写目的寄存器。

B 指令

跳转指令，其典型汇编语法为“B[cond] <label>”。

T32 指令集部分条件码的含义

条件码	助记符	含义	标志位取值
0000	EQ	等于	Z == 1
0010	CS	产生了进位	C == 1
0100	MI	负数	N == 1
0110	VS	溢出	V == 1
1000	HI	无符号比较大于	C == 1且Z == 0
1001	LS	无符号比较小于等于	C == 0或Z == 1
1010	GE	有符号比较大于等于	N == V
1011	LT	有符号比较小于	N! = V
1100	GT	有符号比较大于	Z == 0且N == V
1101	LE	有符号比较小于等于	Z == 1或N! = V

附录 2：综合设计题程序代码

```
void GPIO_Configuration(void)
{
        GPIO_InitTypeDef GPIO_InitStructure;
        GPIO_InitStructure.GPIO_Pin=GPIO_Pin_2|GPIO_Pin_3;
        GPIO_InitStructure.GPIO_Speed= GPIO_Speed_50MHz;
        GPIO_InitStructure.GPIO_Mode=GPIO_Mode_Out_PP;
        GPIO_Init (GPIOA, &GPIO_InitStructure);
}
//...
while(1)
{
        /*循环点亮LED*/
        GPIO_WriteBit(GPIOA,GPIO_Pin_2,(BitAction)0x01);          //(1)
        Delay(0xFFFFF);
        GPIO_WriteBit(GPIOA,GPIO_Pin_2,(BitAction)0x00);          //(2)
        Delay(0xFFFFF);
        GPIO_WriteBit(GPIOA,GPIO_Pin_3,(BitAction)0x01);
        Delay(0xFFFFF);
        GPIO_WriteBit(GPIOA,GPIO_Pin_3,(BitAction)0x00);
        Delay(0xFFFFF);
}
//...
void TIM2_Delay500MS( )
{
        TIM_TimeBaseInitTypeDef   TIM_TimeBaseStructure;
        RCC_APB1PeriphClockCmd (RCC_APB1Periph_TIM2, ENABLE);
        TIM_TimeBaseStructure.TIM_Prescaler=     (3)      ;
        TIM_TimeBaseStructure.TIM_Period=        (4)      ;
        TIM_TimeBaseStructure.TIM_CounterMode= TIM_CounterMode_Up;
        TIM_TimeBaseInit(TIM2,&TIM_TimeBaseStructure);
        TIM_ClearFlag(TIM2,TIM_FLAG_Update);
        TIM_Cmd(TIM2,ENABLE);
        while (TIM_GetFlagStatus (TIM2,TIM_FLAG_Update) == RESET);
}
```