### 1.与其他32位嵌入式微处理器相比 为什么arm得到了广泛的应用?

答:一是因为 [ARM公司](https://baike.baidu.com/item/ARM%E5%85%AC%E5%8F%B8)是一家既不生产芯片（fabless）也不销售芯片（chipless）的公司，它通过出售芯片技术授权，建立起新型的微处理器设计、生产和销售商业模式。更重要的是，这种商业模式取得极大的成功，随着无线通信的快速发展,超过85%的无线通信设备采用了arm技术，尤其是在智能手机市场．Arm能耗低，适合移动端应用,移动端兴起的带动arm ip核的出售.

二是ARM的商业模式是开放的，任何厂商都可以购买授权,采用ARM技术IP核的微处理器遍及各类电子产品：汽车、消费电子、成像、工业控制、海量存储、网络、安保和无线等市场, ARM将其技术授权给世界上许多著名的半导体、软件和OEM厂商，每个厂商得到的都是一套独一无二的ARM相关技术及服务,具有强大的第三方生态系统的支持，降低了开发成本，更有竞争力．利用这种合伙关系，ARM很快成为许多全球性RISC标准的缔造者。

三是：ARM不断创新，抓住机遇，推出Cortex系列后，赶上平板时代，ARM的性能功耗足以满足平板电脑的需求

### 2.你认为嵌入式微处理器发展趋势是什么, 为什么？

答: 一是针对专用领域的加速,新型应用如物联网IoT,对数字信号处理的加速. 在运动控制视频图像采集、传送和显示与监测方面起到很重要的作用．

二是高集成度、低功耗、小型化,控制机器人的运动.

三是低功耗的网络应用, 使得更多物体能远程控制．

四是提供方便、快捷的人机界面，生活物件如台灯闹钟可以和人互动．

### 3.如何对微处理器进行简单的评价？

**1.响应时间**：也称执行时间，是指一个任务从开始到完成的时间

**2.字长** 字长是微型计算机能够直接处理的二进制数据的位数。字长越长，能表示数值的有效位 数越多，在同样的运算速度下，精度也就越高。

**3.主频** 微处理器内部有一个按某一频率产生的时钟脉冲信号，称为主时钟信号。主时钟信号的频率称为微机的主频，是用于协调微机操作的时钟信号。主频决定了计算机的处理速度， 频率越高，处理速度越快.

**4.运算速度** 运算速度是指计算机每秒钟运算的次数。现在普遍采用的方法是根据指令使用的频度和每一种指令的执行时间来计算得出平均速度，以此来衡量计算机的运算速度

### 4.查找最新的arm处理器的型号介绍与本章相关内容的异同。

在 2020 年的 TechDay 上，ARM 直接发布了两款架构。其中一款是顺理成章、按部就班的 Cortex-A78，另一款则显示了 ARM 在业务模型和设计理念\_上的新变化，这款被称为 Cortex-Xl 的全新架构将成为 ARM 在 2020 年的旗舰产品.

**相同：**Cortex-A78依旧采用的是 ARM v8.2 指令集, 保持 ISA 的相关兼容性

**不同：**

1.在高通骁龙 865 中使用的 Cortex-A77 已成为高端 SoC 的首选,Cortex-A78 能耗比更加出色,对比Cortex-A77限定条件下可以性能提升20%或功耗降低50%,

2.新的分支预测器(由[CPU](https://baike.baidu.com/item/CPU/120556)来判断程序分支的进行方向，能够加快运算速度)可以在每个周期处理两个分支，此前的 Cortex-A77 虽然在后端引入了辅助分支预测器，但是其前端的分支预测器依旧只能处理一个分支。在分支预测上做出了加强指令占用更少的资源和电源，但是会输出同样的性能。

3. 寄存器部分，包括寄存器重命名模块和寄存器文件也针对效率进行了优化，一部分减少了尺寸。尤其是寄存器部分已经重新设计了它们能容纳的条目密度，并且在相同的空间内可以打包更多的数据，从而使设计人员能够在不降低其功能或者性能的情况下减少结构的整体尺寸。

4.计算单元,其中改进一个ALU,优化了乘法运算. 增加了一个读写单元, 读取带宽提升了50%，写入带宽翻倍.

5. A78的面积也缩小了5％，这样可以使采用4个A78核心的集群面积减少15％，这样可以为其他组件腾出空间，并且降低成本。

Cortex-X1 有30%的性能提升幅度.基于Cortex-A78进一步加强而来,拥有更大的缓存.前端为了性能扩大了部分端口和资源. 每时钟周期指令解码数量从4提升到5.每时钟周期指令发射数量从6提升到8，提升33%；

只有13级流水的X1无法达到更高的主频，绝对性能还是劣势。生态方面也还不足。

ARM Cortex-M最新的处理器是Cortex-M55，Arm Cortex-M55处理器是具有AI功能的Cortex-M处理器，并且是第一个采用Arm Helium矢量处理技术的功能，带来增强的节能信号处理和机器学习（ML）性能。采用M55处理器的芯片还在验证中，需要等一段时间才会有芯片出来。目前我们能买到的最新的是Cortex-M33，M23处理器，是基于V8M架构设计的，主要功能是在原来M7和M0+基础上增加了安全功能和浮点运算能力，

### 5 微控制器的基本组件有哪些 这些基本组件的重要作用是什么？

1.微处理器: 控制整个微控制器的工作

2.系统管理：地址映射与存储空间配置，存储数据格式 

3.时钟与电源管理：产生独立的时钟源，不同的操作模式实现低功耗控制 ,产生系统时钟，注意不是外部晶振的 频率，它对系统性能有重要影响. 注 意时钟灵活性，实现芯片中的各种接口模块的不同速率要求.

4.定时器：通过对时基信号(对时钟源进行分频)计数，以实现在一定时间下对特定事件的处理。为了方便周期性处理，定时器计数值还可以自动重装. 定时器溢出时系统复位.

5.通用输入输出口：同一引脚可以配置成单独的输入或输出功能、双向的输入输出功能，可以配置成三态或者推挽输入输出

6. 中断控制器：对处理器中断输入的扩充，以支持对更多中断源输入的响应。中断控制器通常输出给处理器以各类中断请求信号与中断编号(指明 中断服务程序地址或中断向量)

7.直接存储器访问：总线上的独立主设备，能够在设置源起始地址、目的起始地址、传输长度等以后，无需处理器干预，自动进行数据传输操作.

### 6 简单描述微控制器的选择方法。

MCU 选型：  
1    制定所需硬件接口列表, 第一种接口是通信接口，包括 USB、IIC、SPI 和UART等外设接口，第二种接口是数字输入和输出、ADC输入或PWM输出接口等等。这两种接口类型将指出MCU需要的引脚数  
2   检查软件架构：    需要频率多少，时间多少。对所需的计算能力在数量级上有大致概念，以确定微处理器类型及其频率  
3   选择架构    8位还是16位32位？不要忘了可能的未来要求和功能扩展.  
4     确定存储器需求：  确保不会出现程序空间不足，或者说可变空间肯定是最优先考虑的。记得给 功能扩展和后续版本留些空间  
5   寻找微控制器 ，和FAE谈谈应用和要求, 另一个是从熟悉的芯片供应商网站搜索  
6    检查成本和功耗:如果应用将通过电池供电并属于移动类型，那么确保具有低功耗是头等大事。 如果应用需要高度专业化或者高端处理器，那么其单价可能非常重要  
7    检查 部件供货情况: 备货周期是多少？是否多家分销商都保有库存？如果不希望陷入订单困境需要等待三个月来履行。另外，还有就是MCU新旧程度以及在产品生命周期内的供货，那么需要寻找制造商保证.  
8    选择开发套件: 应当明确有哪些开发套件可用。如果没有可用的开发套件，那么很有可能所选部件不是最佳的选择。大多数开发套件的价格在 100 美元以内，除非它设计为用于多种MCU开发  
9   调查编译器 和工具: 从各类编译器和调试工具、集成开发环境、示例代码中选择合适的工具  
10    开始实验: 选择高风险部件，构建测试电路与接口，在开发套件上工作。以发现一些未 预料的问题，早期实验将能确保正确的选择，而且在需要变动时，影响最小

### 7 考虑学科的相关知识 并结合本人的情况 你的学习需要加强哪些内容？

1. 学习高速缓存cache 的具体实现,了解cpu与内存的数据传输,优化微处理器性能.

2 学习中断的具体实现,了解中断对用户对系统的影响,做到扎实基础,熟悉中断到恢复的完整过程

3 熟练掌握寄存器的使用, 学习如何利用寄存器,不同状态下寄存器的类型,在有限的寄存器下如何实现多种复杂功能,并且达到高效率.

4.学习处理器的寻址方式.

### 8 预习s3c2410a的arm920t微处理器有几种工作模式？

7种工作模式

除了用户模式外，其他模式可以自由地访问系统资源和改变模式，称为特权模式, FIQ，IRQ、超级用户、异常中止和未定义这5种模式称为异常模式，分别对应特 定的异常出现时进入的模式每种特权模式都有某些附加的通用或状态寄存器，减少模式切换时对现场的保护

9月22日作业

### 1.ARM处理器离开异常时的地址需要进行校正，试问这是在处理器设计中必须的吗？

是必须的, 需要恢复被打断程序运行时寄存器数据,恢复程序运行时状态*CPSR*,通过进入异常时保存的返回地址，返回到被打断程序继续执行

### 2.预习：S3C2410A的ARM920T处理器核的Load-Store指令

9月29日作业

### LDR指令与LDR伪指令有何异同？

LDR指令用于从存储器将一个32位的字加载到目的寄存器Rd中

LDR伪指令用于加载32b的立即数或一个地址值到指定寄存器,

同: 都是加载到Rd中.

异: LDR伪指令的参数多一个”=” ,并且可以被编译器替换成一条合适的指令

可以把一个地址写到某寄存器中, ldr伪指令和mov是比较相似的。只不过mov指令限制了立即数的长度为8位，也就是不能超过512。而ldr伪指令没有这个限制。如果使用ldr伪指令时，后面跟的立即数没有超过8位，那么在实际汇编的时候该ldr伪指令是被转换为 mov指令的。

### 2.ARM技术原理与应用(第2版)-第四章第3、4题(附：结果截屏)[除了题目中提示的ADS工具，也可以另行使用编程实践中拟选择的工具] 

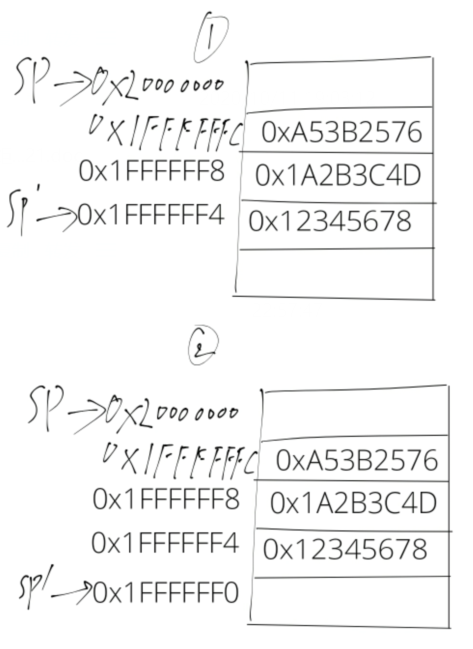
第3题:

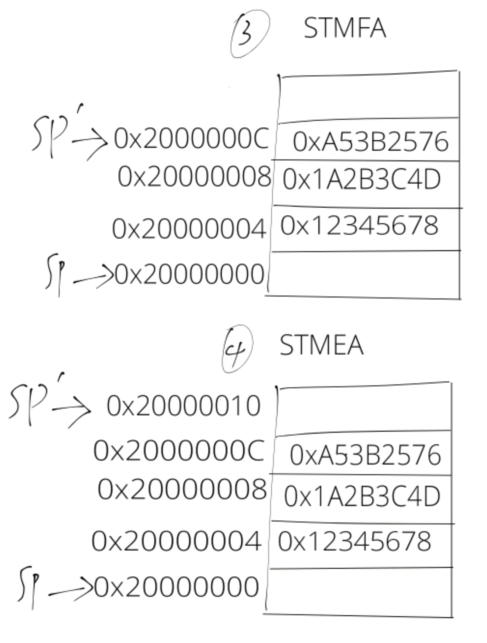
1. 立即数寻址
2. 立即数寻址
3. 寄存器寻址, R0+R2
4. 寄存器及移位常数 R0+R2\*4
5. 寄存器及移位常数 R0-R2\*4
6. 寄存器移位寻址,前索引偏移
7. 寄存器移位寻址,后索引偏移
8. 寄存器移位寻址,后索引偏移,LDMIA多寄存器寻址
9. 寄存器移位寻址,后索引偏移STMIA多寄存器寻址
10. 寄存器寻址 用寄存器Rn的值减去operand2 ，结果保存到Rd中，Rd = Rn - operand2
11. 立即数寻址
12. 寄存器间接寻址,R5是寄存器间接寻址,R1是寄存器寻址
13. 堆栈寻址
14. 相对寻址

第4题:

1. 画出

①STMFD ② STMED





2.写出对应的出栈指令

①LDMFD

②LDMED

③LDMFA

④LDMEA

3. STMDB R0!,{R1-R3} 和 STMFD SP!,{R1-R3}

STMDB R0!,{R1-R3} 最后R0改变.

{!}为可选后缀，若选用该后缀，则当数据加载与存储完成后，将最后的地址写入基址寄存器也就是R0寄存器.

STMFD SP!,{R1-R3}最后SP(R13)改变.

STMFD SP!,{R1-R3} 满递减入栈,

(4)

