投递

银行：

name	deadline	plan	content
中国工商银行	2023-06-20	广东省省行、深圳分行科技英才	广东省省行
中国建设银行	2023-06-25		广东省省行、江西省省行

建设银行实习时间：7月中下旬-8月中下旬，一个月，1200元/月

初筛，面试，

招商银行：18402809049， @Neversay1v1r1

嵌入式：

name	data_cv	detail
深圳佰维科技有限公司	2023-6-25	成都，第一志愿，嵌入式软件实习，第二硬件开发实习
云鲸智能	2023-6-20	东莞，电气实习
TCL	2023-6-25	泛智屏BU研发中心，智能硬件部，惠州
华为	2023-6-25	嵌入式实习，成都
经纬恒润	2023-6-26	软件开发实习（嵌入式）、硬件实习生

面试

程序的局部变量存在于栈中，全局变量存在于静态区，动态申请数据存在于堆中。

关键字 const ： 1）只读 2）使用 const 能产生更紧凑的代码 3）使编译器很自然地保护那些不希望被改变的参数，防止其被无意的代码修改。

BIN base 2

OCT base 8

DEC base 10

HEX base16

STM32定时器

1. 定时器时钟使能，首先，需要使能所需要的定时器时钟，这需要通过RCC（reset and clock control）寄存器来配置，如果使用TIM2定时器，需要使能TIM2的时钟；
2. 定时器配置：配置定时器的基本参数，例如计时器的工作模式，时钟分频系数，计数器的自动重载等，这些配置可以通过定时器的相关寄存器进行配置；
3. 定时器中断设置（可选）：如果需要使用定时器中断，可以配置中断源和中断优先级，并使能定时器中断；
4. 启动定时器：配置完成后，启动定时器开始计数，定时器可以通过软件触发或外部触发来启动计数；
5. 定时器中断处理（可选）：然后配置了定时器中断，当计数器达到设定的值时，会触发中断，在中断服务程序中可以执行相应的操作，例如更新计时器的值，处理中断标志等。

AD采样：

1. ADC配置，首先需要配置ADC模块的相关寄存器，包括时钟使能，采样时间，采样通道等，可以通过寄存器操作或者使用STM32提供的库函数进行配置；
2. GPIO配置：为了使用ADC，需要将相应的GPIO引脚配置为模拟输入模式，并使能相应的ADC通道，可以通过配置GPIO寄存器来完成；
3. 触发方式设置：可以择触发ADC转换的方式，例如软件触发或者外部触发。如果选择外部触发，还要配置相应的触发源和机性；
4. ADC转换：在进行ADC转化之前，可以设置ADC分辨率，对其方式，连续转换模式等，然后可以通过启动转换命令或触发源来开始ADC转换；
5. 中断或轮询获取结果：可以选择使用中断或者轮询的方式来获取ADC转换结果，若使用中断方式，可以在转换完成后触发中断，并在中断服务程序中读取转换结果；如果使用轮询方式，着需要在转换完成后主动查询并读取结果；
6. 结果处理：获取到ADC转换结果之后，根据需要进行相应的处理，例如数据转换，单位换算等

Linux嵌入式和单片机嵌入式的区别

1. 复杂性：Linux嵌入式是基于Linux内核搭建的，具有完整的操作系统功能，包括多任务处理，文件系统，网络协议等，相比之下，单片机嵌入式通常使用裸机或者实时操作系统（RTOS）具有更简单的系统结构和功能；
2. 处理能力：Linux嵌入式系统通常在较强大的处理器上运行，如ARM,X86等，据备较高的处理能力和存储内存，而单片机嵌入式系统使用单片机芯片，其处理能力和存储内存有限；
3. 开发环境：
4. 系统定制性：
5. 成本和功耗：

   总的来说，Linux嵌入式系统适用于需要复杂功能，较高性能和较大内存需求的应用，如智能手机，平板电脑，网络设备等，而单片机嵌入式适用于资源有限功耗要求低，对实时性要求较高的应用，如传感器，家电，小型电子设备等

C语言的堆和栈的区别

C语言程序经过编译连接后形成的二进制映像文件由堆，栈，数据段（只读数据段，已经初始化读写数据段，未初始化数据段）和代码段组成

1. 存储内容不同；
2. 管理方式：栈由系统自动分配，而堆需要程序员自己申请（malloc，realloc，callloc）并指明大小，且由程序员释放；
3. 空间大小不同，栈的空间一般较小，而堆的空间较大且灵活；
4. 是否产生碎片：栈不产生碎片，而堆采用的是链表的存储方式，会产生碎片；
5. 扩展方式不同：栈向低地址扩展，是一块连续的内存区域，而堆向高地址扩展，是不连续的内存区域；

6. 分配方式不同：栈可以静态分配和动态分配，而堆只能动态分配；
7. 分配效率不同：栈由系统自动分配，速度快，不受程序员控制，而堆由new分配的内存，一般速度比较慢，而且容易产生内村碎片，不过使用方便。

存储器

ROM：只读存储器，掉电不丢失数据，CPU 只能从中读取数据而不能修改，访问速度慢，我们的 .c .h文件中的代码，全局变量，const 定义的常量数据都存储在 ROM 中，嵌入式中的 ROM 通常由 flash 构成，大小以字节为单位，为了多存储数据它们的容量需求较大。

RAM：随机访问存储器，可以快速读取和写入，掉电后数据自动丢失，程序中需要被改变或者更新的变量都存储在 RAM 中，CPU 可以修改他们的值，嵌入式应用中 RAM 制作工艺较为复杂，成本较高，对 RAM 的容量需求不是很大

EPROM：可擦可编程只读存储器

EEPROM：电子可擦除可编程只读存储器

易失性数据存储器（volatile memory），也称可变存储器或者 RAM ，特点是掉电后不能保存数据，而且无法恢复。在计算机和数字系统中用来存储程序，数据和中间结果。

SRAM ：静态随机存储器，只要保持通电，里面保存的数据就可以恒常保持，即 SRAM 不需要刷新电路，就能保存他内部储存的数据

DRAM ：动态随机存储器，里面存储的数据需要周期性地更新，每个一段时间，都需要刷新充电一次，否则数据就会消失。速度比 SRAM 慢，但是比 ROM 快，价格上比 SRAM 要便宜很多，计算机内存一般使用 DRAM ，又分为 DDR RAM,

SRAM 具有较高的性能，缺点是集成度较低，功耗较 DRAM 要大，SRAM 结构复杂，成本较高。DRAM 拥有更高的密度，成本低，但访问速度比 SRAM 慢

非易失性数据存储器（non-volatile memory），掉电后，所存储的数据不会消失

ROM

flash：可以对称为块的存储单元块进行擦写和再编程，允许在操作中被多次擦写的存储器。

分为 NOR flash 和 NAND flash

- 结构不同：NOR flash 采用并行连接的存储单元结构，每个存储单元都是独立的访问线，可以同时读取和写入；NAND flash 采用串行的存储单元结构，多个存储单元被结合成一个页page ， 多个页再组合成一个块block，从而形成一个存储数组。 NAND flash 的读取和写入是以页为单位进行的，
- 工作原理不同：NOR flash 通过直接访问存储单元的地址来完成读写操作，可以快速的随机读取，适合需要快速响应和执行代码的应用，然而 NOR flash 的写入速度慢，且擦除需要对整个块进行，导致读写和擦除的效率低。而 NAND flash 采用更复杂的读取和写入算法，随机读取性能较差，写入速度较快，且擦除操作可以针对单个块进行。
- 应用领域不同：NOR flash 适用于需要快速响应和执行代码的应用，常被用于嵌入式系统，微控制器。NAND flash 使用于需要大容量存储的应用，如 SSD，存储卡，USD 闪存驱动器。

  总的来说，NOR flash 和 NAND 存在结构，工作原理，应用领域方面的不同。NOR flash 具有快速读取速度和随机访问能力，使用于需要快速响应和执行代码的应用。而NAND flash 具有较高的存储密度，适用于大容量存储的应用。

  3D XPoint

  HHD 机械硬盘

  光盘和软盘

UFS：通用闪存存储，采用串行数据传输技术，只有两个数据通道，但是速率超过 eMMC ,工作模式为全双工，同一天通道允许同时读和写，而且读写可以同时进行，传输效率提高。无论是数据传输技术，还说工作模式，UFS 都优于 eMMC。

eMMC：嵌入式多媒体存储卡，并行数据传输技术，主控和存储单元之间拥有八个数据通道，它们可以同步工作，工作模式是半双工，每个通道允许读写传输，但是同一时间只能读或写

嵌入式存储，消费级存储，工业级存储，·

单片机基础Microcontroller unit

按照其存储器类型可分为 无片内 ROM 和 带片内 ROM 型，无片内 ROM 型的芯片，必须接 EPROM 才能应用；带片内 ROM 型的芯片又分为片内 EPROM 型

按照用途可分为通用性和专用型

根据 数据总线的宽度 和 一次可处理的数据字节长度 可分为8，16，32为 MCU

国内 MCU 应用市场最广泛的是 消费电子领域，其次是 工业领域 和 汽车电子领域。消费电子包括家用电器，电视，游戏机等，工业领域包括智能家居，自动化，医疗应用，其次领域包括汽车新能源和安全控制系统

单片机的运行主要使用了三个部件：CPU，RAM 和 ROM，其中 CPU负责算术运算和逻辑运算，单片机运行时 CPU会复制 ROM 中所需的部分程序和数据到 ROM 中运行，执行程序的功能，运行结束断电后，ROM 中的数据全部清空，等待下次使用。

外扩 ROM：存储图片，视频等数据，外扩 ROM 如 SD 卡，通过一定的通信方式和单片机相连，可以读取调用外部存储器的数据。

外扩 RAM：因为存储在 ROM 中的数据总是要复制在 RAM 中运行，所以 RAM 也会不够用，导致溢出，通过外扩 RAM 来辅助运行，

单片机基本功能：

1. IO口
2. Timer定时器：
3. 外部中断
4. 通讯接口：一般包括 UART，I2C ，485，CAN，SPI 接口等
5. 看门狗定时器：为 MCU 因为意外故障而导致死机提供了一种自我恢复能力。

CPU：中央处理器，从内存或缓存中取出指令，放入指令寄存器，对指令译码分解成一系列的微操作，然后发出各种控制命令，执行微操作系列，从而完成系统指令的执行

内存：CPU 不能字节调用存储在硬盘上的系统，程序和应用，必须首先将硬盘中的有关内容存储在内存中，这样才能被 CPU 读取运行。因此，内存作为硬盘和 CPU 的“中转站”，对电脑运行速度有较大影响

虚拟内存： 当运行数据超出物理内存容纳限度时，部分数据就会溢出，这时系统会把硬盘上的部分空间模拟成内存-虚拟内存，并将暂时不运行的程序或者不使用的数据存放在这部分空间，等待需要的时候方便及时调用。

硬盘：长时间存储数据或程序，就需要硬盘。

CPU 内部由 ALU（算术逻辑单元）， CU （控制器），寄存器，中断系统组成，其本质是一个集成电路，外部通过总线与控制总线，数据总线，地址总线相连，对数据和程序进行相关操作。

冯·诺伊曼体系：数据和指令都存储在同一存储区域，取指令与取数据利用同一数据总线，

哈佛体系：程序存储器和数据存储器分开，且各自都有自己的总线，取指令和取数据可以在同一周期运行，速度快

DSP 适用于数字信号处理，例如 FFT, 数字滤波算法，加密算法和复杂控制算法等，具有强大的数据处理能力和高运行速度

通讯相关

全双工：发送数据的同时也可以接收数据，两者同时进行，一条数据线用作发送数据，另一条用作接收数据，是双向的；

半双工：一段时间内，只有一种动作发生，发或者收，半双工只有一条数据线，既要用作发送数据，又要用来接收数据，是单向的。

同步：同步通信效率高，对时序的要求也高，同步传输往往通过特定的时钟线路协调时序，同步传输的单位是帧（bit），而异步传输的单位是字符，相邻字符之间的间隔可以任意长。同步通信多用于多点对多点，而异步通信只适用于点对点

串行：一个字节的数据，排成一行一位一位地依次发送给接收设备

并行：一个字节的数据，排成一列一次性发送给设备，速度快，吞吐量大，消耗更多的 IO 资源

串行通信主要用在速度要求不高，有一定距离地传输场景，如 UART, I2C 通信，并行通信多用于传输速率要求高，吞吐量大的场景，如数字视频接口

- UART：异步串行收发器，信号线包括 RX 和 TX 两条，分别用作发送数据和接收数据，基本数据格式为 起始位+数据（七位或八位）+校验位（可选）+停止位 ，波特率就是 UART 通信数据传输的速率，也就是每秒传输的数据位数，一般选择9600，15200，38400等，波特率越大，传输速度越快，但是稳定的传输距离越短，抗干扰能力越差。优点：通信距离远，缺点是速度慢，
- I2C：使用两线在主控制器和从机之间进行数据通信，一条 SCL（串行时钟线），一条 SDA（串行数据线），都需要接上拉电阻，空闲时候，两条线都是高电平，支持两种模型，标准模式和快速模式，标准模式下最高速率100K bit/s，快速模式最快达到400K bit/s，使用ACK 应答信号来确认每帧数据成功传输，可以在总线上挂接多个设备，每个设备都可以作为主机（或者从机），每个设备都具有唯一地址，通过地址来识别和访问，使用软件通过 IO 口就可以实现
- 485：异步半双工通信，多点通信，允许多个发送器连接到同一总线上，使用A, B 两根信号线，逻辑”1“表示两线之间2-6V 的电压差，逻辑”0“表示-2到-6V 的电压差，传输速率高，抗干扰能力强，传输距离远
- CAN:
- SPI: 一种高速，全双工，同步的数据总线，使用单独的数据线和一个单独的时钟信号来保证发送端和接收端的完美同步，时钟是一个震荡信号，SPI 以主从方式工作，通常有一个主设备和一个或多个从设备，一般需要4根线进行通信，CS/SS（片选信号线，选择需要通信的从设备） SCK（串行时钟），SDO（主出从入信号线），SDI（主入从出信号线），全双工串行通信，高速数据传传输速率，灵活的数据传输，不限于8位，简单的硬件结构，从设备不需要唯一地址（与 I2C 不同），从机使用主机时钟，不需要精密的时钟（与 UART 不同），不需要收发器（与 CAN 不同）

STM32相关

ARM Cortex三个系列， A（应用）系列高性能，高时钟频率，深流水线，广泛应用于智能手机，平板电脑，智能家居家电；R（实时） 系统响应快，高时钟频率，应用于汽车电子；M（微控制器）功耗低，应用于嵌入式系统，物联网，可穿戴设备等。

STM32 的 IO 模式具有四种，输入，输出，复用和模拟

输入设置只有上下拉电阻的配置，共三种配置：上拉输入，上拉输出，浮空输入（既不上拉，也不下拉）

输出主要有两种：推挽输出和开漏输出，开漏输出一般搭配上拉电阻

推挽（push-pull）输出

推挽输出能够输出高电平和低电平，且电平都是确定的，都具有驱动能力，一般不需要上下拉电阻，常用于 UART 和 SPI 等单向通信接口（通信线路上信号只往一个方向走）

开漏输出（open-drain）

开漏输出的高电平没有驱动能力，实际是个高阻态，一般需要接上拉电阻完成对外驱动，可以是单片机内部的也可以是外部的，开漏输出常用于 I2C 等双向通信接口（通信线路上信号可以往两个方向走）。以 I2C 为例，总线接上拉电阻，总线上的设备默认都是没有输出的，总线处于高电平状态，一旦由任意一个设备输出低电平，整个总线被拉低，其他设备都可以读到这个低电平。

51单片机相关

51单片机的中断处理流程

面试题

1. 程序的局部变量存在于栈区，全局变量位于静态区，动态申请变量位于堆区
2. const 的含义：
   - 只读
   - 使用关键字 const 可以产生更加紧凑的代码
   - 使编译器很自然的保护那些不希望被改变的参数，防止其被无意的代码修改
3. int const *p；const 修饰指针p，表示指针p的值不能改变，而p的地址是可以改变的

   int const p;　const 修饰的是p（地址）是常量，不能改变，但是p的值可以改变

   int const *p　const; 表示只读，其地址以及地址中的值都不可改变

   指针常量：int * const　p = &a；指针是常量

   常量指针：const　int *p = &a；常量的指针

   指向常量的指针常量：const int * const p = &a；

   指针常量，即指针是一个常量；常量指针，即常量的指针
4. #define 与 const 的区别
   - const 定义常量是有数据类型的，而宏定义定义的常量却没有，const 可以有不同的作用域
   - 编译器会对 const 常量进行安全检查，而对宏定义常量只进行字符替换，没有类型的安全检查，并且在字符替换中可能产生错误
5. 内存溢出：指程序在申请内存时，没有足够的内存空间供其使用，比如申请了一个integer整型数据，但给他存了一个long 才能存下的数据，就会导致内存溢出
6. 内存泄漏： 指你向系统申请分配内存进行使用，可以使用完了以后没有归还，结果那块内存就不能再访问，系统也不能再次将他分配给需要的程序。

   按照发生的方式来分，内存泄漏分为：常发性内存泄漏，偶发性内存泄漏，一次性内存泄漏，隐式内存泄漏
7. 什么是预编译，何时需要预编译？、

8. 进程与线程的区别

   进程是资源分配的最小单位

线程是程序执行的最小单位，也是处理器调度的基本单位

进程有独立的地址空间，而线程是共享进程中的数据，使用相同的地址空间

线程之间需要通信，同一进程下的线程共享全局变量，静态变量等数据，而进程之间通信需要以通信的方式进行

每个独立的进程有一个程序运行的入口，顺序执行序列和程序入口，但是线程不能独立执行，必须依存在应用程序中，由应用程序提供多个线程执行控制

9. 线程与线程的优缺点

   线程执行开销小，但是不利于资源的管理和保护，进程执行开销大，但是可以很好的进行资源管理和保护

10. 嵌入式系统四个阶段：无操作系统阶段，简单操作系统阶段，实时操作系统阶段，面向 Internet 阶段
11. 嵌入式系统的组成：硬件层，中间层，系统软件层，应用软件层
12. 如何应用一个已经定义过的全局变量：应用头文件，或者使用 extern 关键字
13. 在不同的 C 文件中以 static 形式来声明同名全局变量
14. 嵌入式系统硬件层的组成：

    硬件层：嵌入式微处理器，存储器，通用设备接口和 I/O 接口

    嵌入式核心模块=微处理器，电源电路，时钟电路，存储器

15. 时钟周期：也称震荡周期，定义为时钟频率的倒数（单片机外接晶振的倒数，如12Mhz 的晶振，时钟周期就是 1/12 us），他是单片机中最基本，最小的时间单位

    状态周期，他说时钟周期的两倍

    机器周期：单片机的基本操作周期，在一个操作周期内，单片机完成一项基本操作，如取指令，由12个时钟周期组成

    指令周期：指 CPU 执行一条指令所需要的时间，一般一个指令周期含有1-4个及其周期

16. 51单片机的时钟引脚是 XTAL1(19脚), XTAL2(18脚)，两种方式，片内时钟震荡方式，外部时钟方式，将 XTAL1接地，XTAL2接外部时钟信号
17. STC89C51 的编程控制引脚：RST单片机复位引脚，PSEN（29），ALE (30), EA (31)
18. STC89C51 的 P0 口（39-32）：双向8位 IO 口，每个可独立控制，为高阻态，不能正常输出高电平，使用时需要接上拉电阻

    P1（1-8），P2（21-28）， P3（10-17）：准双向八位 IO 口，每个口可以单独控制，内带上拉电阻

19. C 语言中，关键字 static 的作用：
    - 在函数体，一个被声明为静态的变量在这一函数被调用过程中维持其值不变
    - 在模块内，一个被声明为静态的变量可以被模块内所用函数访问，但不能被模块外其他函数访问，他是一个本地的全局变量
    - 在模块内，一个被声明为静态的函数只可被这一模块内的其他函数调用，也就是，最高函数被限制在声明他的模块的本地范围内使用
20. static 全局变量与普通的全局变量的区别，static 函数与普通函数的区别

    全局变量的声明之前加上 static 就成了静态的全局变量

    全局变量和静态全局变量都是静态存储方式，区别在于，全局变量的作用域是整个源文件，当一个源文件由多个源文件组成时，全局变量在所有源文件中都是有效的，而静态全局变量这限制了其作用域，即只在定义该变量的源文件中有效

    同理， static 函数与普通函数作用域不同，只在当前源文件中使用的函数应该声明为内部函数（static），在当前源文件中声明和定义。而对于可在当前源文件之外使用的函数，应该在头文件中声明，用使用这个函数的源文件需要包括该头文件

21. 扇区是对硬盘而言，是物理层的。块和簇是相对文件系统而言的，是逻辑层的，磁盘控制器其作用除了读取数据，控制磁头等作用外，还可以用来映射两层的关系。

簇/块是操作系统读写文件的基本单位，在 Windows 下 NTFS 等文件系统中叫簇，而在 Linux 下的文件系统叫块；

扇区是磁盘读写的基本单位，扇区是磁盘的最小的物理存储单元，是磁头从磁盘中读取数据的最小单位（一般512B），即磁头每次从磁盘读取数据，都是一个扇区一个扇区读取的。但由于操作系统无法对数目众多的扇区进行寻址，所以就将相邻的扇区组合在一起形成一个簇，然后再对簇进行管理（每个簇包含2，4，8，16，32或64个扇区）；

块的大小在硬盘格式化时被指定，一般有1K，2K，4K，如果设置为4K，那么磁盘读取8个扇区之后，才将数据块传给操作系统。

文件系统是操作系统的一部分，所以文件系统操作文件的最小单位是块和簇；

操作系统经常与内存和硬盘这两种存储设备进行通信，与硬盘通信，就虚拟出了块/簇这个基本单位，而与内存通信，就虚拟出了页这个基本单位。

22. I2C通信启动信号：SCL为高电平时，SDA由高电平跳变为低电平。结束信号：SCL为高电平时，SDA由低电平跳变为高电平。所有数据都是以八位字节传送的，没法送一个字节，就在脉冲 9 期间释放数据线，由接收方反馈一个应答信号。应答信号为低电平时，规定为有效应答位（ACK），表示接收器已经成功接收到该字节；应答信号为高电平时，规定为非应答信号（NACK），表示接收器未成功接收该字节。
23. 处理器访问内存时，CPU 核，cache，MMU 如何协同工作

    MMU 负责将虚拟地址转化成物理地址，CPU 通过地址来访问内存中的单元，如果 CPU 没有 MMU 或者 MMU 没启动，那 CPU 发出的地址，此时必须是物理地址，将直接传到 CPU 芯片的外部地址引脚上，直接被物理内存接收。

    如果 CPU 启动了 MMU ，CPU 内核发出的虚拟地址将被 MMU 截获，同时 MMU 将该地址翻译为物理地址发送到CPU 芯片的外部地址引脚上，也就是将虚拟地址映射到物理地址中。
24. 同步串口与异步串口的区别：

    同步串行通信方式的特点：以数据块为单位传输信息，在一个数据块内，字符与字符无间隔，发送方与接收方的时钟严格同步。

    异步串行通信方式的特点：以字符为单位传输信息，相邻两个字符之间的间隔是任意，发送方与接收方的时钟只要相近即可。

    同步串行通信的数据格式：2个同步字符作为一个数据块的起始标志 + n个连续传输的数据 + 2个字节循环冗余校验码（CRC）

    异步串行通信的数据格式：起始位+数据位+奇偶校验位+停止位（1~2位）

    用途：同步串口应用于通信网中，多点对多点的通信，有大批量数据需要传输；异步串口，应用于工业和实际应用中，适用于点对点通信，短距离，速率不高的情况下。
25. ARM 外围接口丰富，标准化和通用性做的很好，功耗较低，主要优势在于控制方面，适合用于一些消费电子领域。

    DSP 主要用来计算的，比如 加密解密，FFT， 调制解调等，优势是强大的数据处理能力和较高的运行速度；

    FPGA：现场可编程门阵列，是专用集成电路（ASIC）中集成度最高的一种，使用 VHDL 或 verilogHDL 来编程，灵活性强，能够进行编程，除错，再编程和重复操作，因此可以充分地进行开发和验证
26. 一个程序从开始运行到结束地完整过程（四个过程）
    1. 编译预处理：处理伪命令，头文件包含，宏定义，条件编译
    2. 编译：把预处理之后的文件进行语法分析，生成汇编代码
    3. 汇编：把汇编文件生成机器代码（二进制代码）
    4. 链接：去指定路径下找库函数（头文件包含的是声明，具体实现封装在库中）
27. 大小端的区别以及各自的优点

    大端模式：指数据的低位保存在内存的高地址中，而数据的高位保存在低地址中；

    存储数据时先存最高有效位，ARM 采用该方式

小端模式：指数据的低位保存在内存的低地址中，而数据的高位保存在高地址中。

存储数据是先存最低有效位，Intel AMD 采用这种方式

28. Bootlader的作用：

    简单的说，Bootloader 就是操作系统运行之前的一段小程序，通过这段小程序，可以初始化硬件设备，从而将系统的软硬件环境带到一个合适的状态，以便最终调用操作系统做准备，对于 Bootloader 的启动过程又分为两个阶段

    1. stage 1 全部由汇编编写，工作是（1）初始化硬件设备（2）为加载 stage 2 准备 RAM 空间（3）拷贝 stage 2 到RAM 空间 （4）设置好堆栈段为 stage 2 的 C 语言环境做准备（5）跳转到 stage 2 代码的入口点
    2. stage 2 全部由 C 语言编写，工作是（1）初始化本阶段要使用到的硬件设备（2）检测系统内存映射（3）将内核映像和根文件系统从 flash 上读到 RAM 空间种（3）为内核设置启动参数 （4）调用内核
29. 为什么需要 Bootloader，每种不同的 CPU 体系都有不同的 Bootloader ，除了依赖于 CPU 的结构体系外，还依赖于具体的嵌入式板级设备的配置，比如板卡的硬件地址分配。对于两块不同的开发板，即使它们基于同一种 CPU 构建的，但是他们的硬件资源或配置不一样，它们上面运行的 Bootloader 也会不一样。
30. 为了使开发板成功运行 Linux 内核，需要完成以下三个方面的开发工作

    bootloader 启动转载

    kernel 内核

    rootfs 根文件目录

31. Linux 内核的组成：
    1. 内存管理：
    2. 进程管理：
    3. 进程间通信：
    4. 虚拟文件系统：
    5. 网络接口：
32. boot0 接0 ，从用户 flash 启动；boot0 接1，boot1 接0，从系统存储器启动；boot0 接1，boot1接1，从内嵌 SRAM 启动。
33. 程序在进入 main 函数之前执行流程
    1. 主栈指针 MSP 指向地址 0x0800 0000处
    2. 单片机硬件产生复位信号
    3. MSP 跳转到复位向量地址 0x0000 0004处
    4. 跳转到复位程序入口处，初始化向量表
    5. 复位系统寄存器，将各个寄存器的值设置为默认值
    6. 设置各个系统的时钟频率，设置堆栈
    7. 进入 main 函数
34.
35.
36.
37.
38.
39.
40.
41.
42.