

题型及分值：

一、填空（每空 1 分，共 20 分）

二、选择（每题 1 分，共 30 分）

三、简答（每题 3 分，共 30 分）

四、综合（每题 10 分，共 20 分）

内容不限于以下知识点，还包括课后实验操作中完成的内容。

第一章 嵌入式基本概念

嵌入式系统是以应用为中心，以计算机技术为基础（另包含微电子技术、控制技术和通讯技术），并且软硬件可剪裁，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。

“嵌入性”、“专用性”与“计算机系统”是嵌入式系统的三个基本要素。

常见的嵌入式操作系统：

人们日常所广泛使用的嵌入式设备有哪些？列举 3 个以上手机、空调、冰箱，而车载 GPS 系统，智能家电，机器人也是属于嵌入式设备。

冰箱

嵌入式系统的三大基本特征是嵌入性、专用性、计算机系统。

嵌入式系统是以应用为中心，以计算机技术为基础的专用计算机系统。

嵌入式系统是以应用为中心，以计算机技术为基础的专用计算机系统。

BSP 是指板级支持包。

嵌入式系统进行开发时，通常采用的方式是交叉编译。

常见的嵌入式操作系统有哪些？请列举出三个。

Linux, VxWorks, Windows CE

下面（ ）不属于嵌入式系统的三大基本特征？B

A、专用性 B、体积微小 C、嵌入性 D、计算机系统

嵌入式系统的出现，可追溯到（B）

A、20 世纪 50 年代 B、20 世纪 60 年代

C、20 世纪 70 年代 D、20 世纪 80 年代

嵌入式片上系统的英文简称是 D

A、MPU B、MCP C、DSP D、SoC

在嵌入式操作系统与硬件之间，新加入的硬件抽象层，其英文简称为（B）

A、HSC B、HAL C、DSP D、BSP

下面不属于嵌入式操作系统的是（B）

A、VxWorks

B、Windows

C、Android

D、uC/OS

通常所说的 32 位微处理器是指(C)

A、地址总线的宽度为 32 位

B、处理的数据长度只能为 32 位

C、CPU 字长为 32 位

D、通用寄存器数目为 32 个

下面(B)特性不符合嵌入式操作系统特点。

A、实时性

B、不可定制

C、微型化

D、易移植

下面哪点不是嵌入式操作系统的特点?(C)

A、内核精简

B、专用性强

C、功能强大

D、高实时性

嵌入式应用通常考虑进行运行速度优化和代码尺寸优化,这是由嵌入式系统的(C)

A、专用性特点决定的

B、计算机系统特点决定的

C、资源受限特点决定的

D、功耗约束特点决定的

嵌入式应用通常需要考虑低功耗设计,即俗称的“省电”,这是由嵌入式系统的(D)

A、专用性特点决定的

B、计算机系统特点决定的

C、资源受限特点决定的

D、功耗约束特点决定的

比较嵌入式系统与通用计算机的区别。

嵌入式系统与通用计算机系统有着完全不同的技术要求和技术发展方向。通用计算机系统的技术要求是高速、海量的数值计算,其技术发展方向是总线速度的无限提升、存储容量的无限扩大;而嵌入式计算机系统的技术要求则是智能化控制,技术发展方向是与对象系统密切相关的潜在性能、控制能力与控制的可靠性不断提高。

1. 简述嵌入式系统的概念、组成及特点。

答:嵌入式系统是以应用为中心,以计算机技术为基础,采用可裁减软硬件,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。一般由嵌入式微处理器、外围硬件设备、嵌入式操作系统以及用户的应用程序等四个部分组成。其特点有:

1) 嵌入式系统通常是面向特定应用的

2) 嵌入式系统是将先进的计算机技术、半导体技术和电子技术与各个行业的具体应用相结合后的产物

3) 嵌入式系统的硬件和软件都必须高效率地设计,量体裁衣、去除冗余

4) 嵌入式系统和具体应用有机地结合在一起,它的升级换代也是和具体产品同步进行

5) 为了提高执行速度和系统可靠性,嵌入式系统中的软件一般都固化在存储器芯片或单片机本身中

6) 嵌入式系统本身不具有自主开发能力。

第二章 嵌入式的硬件体系结构

通常微处理器在处理一条指令要经过三个步骤：取指（从存储器装载一条指令）、译码（识别将要被执行的指令）、执行（处理指令并将结果写回寄存器）。

嵌入式微处理器的架构分为__RISC__和 CISC 两种类型。

Reduced Complex

ARM 核有两个指令集，分别是： ARM 指令集和__Thumb 指令集__。

ARM 支持两个指令集， ARM 核因运行的指令集不同，分别有两个状态_ARM 状态_和 _-__Thumb 状态__。状态寄存器 CPSR 的__T_位反映了处理器运行不同指令的当前状态。

ARM 处理器有两种总线架构，冯. 诺依曼和__哈佛__。

编译链接代码时，有两种存储代码和数据的字节顺序，一种是_小端格式_，另一种是__大端格式_。

ARM 字数据存储格式有：__大端格式__和小端格式。

嵌入式微处理器 ARM7 采用了典型的三级流水线，具体指哪三级__取指__、__译码__、__执行__。

对于 ARM7 三级流水线，当一条指令被译码时，上一条指令正被（ D ）

A、取指 B、译码 C、PC 值递增 D、执行

ARM 处理器的工作模式有（ C ）

A、5 种 B、6 种 C、7 种 D、8 种

哈佛结构和冯诺依曼结构的区别是（ A ）。

A、指令和数据分开存储 B、不需要程序计数器
C、统一编址 D、单一数据总线

将高速缓存分为指令缓存和数据缓存的体系结构是（ B ）

A、冯·诺依曼结构 B、哈佛结构 C、RISC D、CISC

指令和数据共享同一总线的体系结构是（ A ）

A、冯·诺依曼结构 B、哈佛结构 C、RISC D、CISC

每条指令都采用标准字长、执行时间短，便于指令的流水线优化的微处理器属于（ C ）

A、冯·诺依曼结构 B、哈佛结构 C、RISC 架构 D、CISC 架构

下面关于哈佛结构描述正确的是（ A ）

A、程序存储空间与数据存储空间分离 B、存储空间与 I/O 空间分离
C、程序存储空间与数据存储空间合并 D、存储空间与 I/O 空间合并

在指令系统的各种寻址方式中，获取操作数最快的方式是（ B ）。

A、直接寻址 B、立即寻址 C、寄存器寻址 D、间接寻址

以下哪个指令能够完成从 ARM 指令集跳转到 Thumb 指令集：（ A ）。

A、BX 指令 B、B 指令 C、BL 指令 D、SWI 指令

同 CISC 相比，下面哪一项不属于 RISC 处理器的特征（ D ）。

A、采用固定长度的指令格式，指令规整、简单、基本寻址方式有 2~3 种
B、减少指令数和寻址方式，使控制部件简化，加快执行速度
C、数据处理指令只对寄存器进行操作，只有加载/存储指令可以访问存储器，以提高指令的执行效率，同时简化处理器的设计
D、RISC 处理器都采用哈佛结构

在 ARM 体系结构中，_____寄存器作为连接寄存器，当进入子程序时或者处理器响应异常的时候，用来保存 PC 的返回值；_____寄存器作为处理器的程序计数器指针。（ C ）

A、R0, R14 B、R13, R15 C、R14, R15 D、R14, R0

以下叙述中，不符合 RISC 指令系统特点的（ B ）。

A、指令长度固定，指令种类少 B、寻址方式种类丰富，指令功能尽量增强
C、设置大量通用寄存器，访问存储器指令简单 D、选取使用频率较高的一些简单指令

写出下列缩写的英文全称和中文含义？

ARM: Advanced RISC Machine: 先进精简指令集设计。

RTOS: real time operation system, 实时操作系统

SOC: System on Chip, 片上系统

MMU: memory management unit, 内存管理单元

流水技术：将一重复的时序过程分解为若干子过程，每个子过程都可有效地在其专用功能段上与其它子过程同时执行，这种技术称为流水技术。

哈佛结构：数据和指令分开存储，PC 指针仅指向程序存储器而不指向数据存储器。

简述冯·诺依曼结构和哈佛结构的区别。指令和数据分开存储

简述 RISC 指令集的特点。结构优先选取使用频率最高的简单指令，避免复杂指令；将指令长度固定，指令格式和寻址方式种类减少；以控制逻辑为主。

简述 CISC 指令集的缺点。指令集的各种指令中，其使用频率却相差悬殊，大约有 20% 的指令

被反复使用，占整个程序代码的 80%。而余下的 80% 的指令却不经常使用，在程序设计中只占 20%。

第三章 嵌入式 Linux 操作系统

嵌入式 LINUX 的内核有五个组成部分，它们是进程调度、__内存管理__、__文件管理__、__设备控制__和__网络功能__。

__mv__ 命令可以移动文件和目录，还可以为文件和目录重新命名。

VI 的工作模式有哪三种：__命令行模式__、__插入模式__、__底行模式__。

VI 从编辑状态进入命令行模式的方式是：键入__ESC__按键。

VI 从命令行模式进入编辑状态的方式是：键入__i__按键。

VI 从命令行模式进入底行模式的方式是：键入__:___按键。

VI 在底行模式中，如果要对刚刚编辑的文件存盘退出，则应键入__wq__命令。

VI 在底行模式中，如果要对刚刚编辑的文件不保存，且退出，则应键入__q!__命令。

Linux 有两种工作界面：字符界面和__图形界面__。

__rm__ 命令可删除文件或目录，其主要差别就是是否使用递归开关-r 或-R。

删除文件命令为：__rm__。

文件权限读、写、执行的三种标志符号依次是：__r w x__。

在使用 ls 命令时，为了查看当前目录下的文件属性，应使用参数__-l__。

Linux 文件权限一共 10 位长度，分成四段，第一段表示的内容是__文件类型__。

第一段表示文件类型

第二段表示用户对这文件的权限

第三段表示用户所属的组对这文件的权限

第四段表示其它用户对这文件的权限

Linux 文件系统的文件都按其作用分门别类地放在相关的目录中，对于外部设备文件，一般应将其放在__dev__目录中。

关于 Linux 下面说法正确的是（ C ）

- A、路径名以“\”符号分割
- B、命令和路径名大小写不敏感
- C、文件系统是从“/”开始的统一的目录空间
- D、文件系统中有诸如 C:、D:之类的驱动器盘符

嵌入式操作系统来解决代码体积与嵌入式应用多样性的问题一般是（ A ）

- A、使用可定制操作系统
- B、将操作系统分布在多个处理器上运行
- C、增大嵌入式设备的存储容量
- D、使用压缩软件对操作系统进行压缩

操作系统定制的目的是（ D ）

- A、让操作系统网络通信速度更快 B、让操作系统操作界面符合用户习惯
C、让操作系统所占的存储空间尽量小 D、让操作系统能在指定的处理器上运行

在 vi 编辑器中的命令模式下，键入（ B ）可在光标当前所在行下添加一新行。

- A、 a B、 o C、 I D、 A

在 vi 编辑器中的命令模式下，键入（ C ）可进入编辑状态。

- A、 a B、 o C、 I D、 A

在 vi 编辑器中的命令模式下，键入（ B ）可删除光标所在行。

- A、 a B、 d C、 I D、 A

如果要将文件名 file1 修改为 file2，下列命令（ B ）可以实现。

- A、 cp file1 file2 B、 mv file1 file2
C、 ls file1 >file2 D、 ll file1 >file2

要给文件 file 加上其他人可执行属性的命令是：（ C ）

- A、 chmod a+x B、 chown a+x C、 chmod o+x D、 chown o+x

改变文件所有者的命令为（ C ）？

- A. chmod B. touch C. chown D. cat

不存盘退出 vi 的指令是（ B ）。

- A、 q B、 q! C、 w D、 wq

下面哪种不属于 VI 三种工作模式之一（ D ）。

- A、 命令行模式 B、 插入模式
C、 底行模式 D、 工作模式

在 vi 编辑器的命令模式中，删除一行的命令是：（ B ）

- A、 yy B、 dd C、 pp D、 xx

执行命令 ls -l 时，某行显示如下：

```
-rw-r--r-- 1 puhb puhb 656 Jun 17 20:55 hello.c
```

用户 puhb 对该文件具有什么权限？如何设置相应权限？

读写，~~chmod 相关信息~~一是chmod+(r/w/x) hello.c, 二是chmod+8进制数 hello.c

创建目录 dir1，并将 hello.c 文件复制到该目录中；改变 hello.c 文件的权限，设置该文件为用户拥有读、写、执行，用户组有读、写，其他用户只有读权限，写出相应命令。

```
mkdir dir1 cp hello.c dir1/ chmod 764 mkdir dir1  
cp hello.c dir1  
cd dir1  
chmod 764 hello.c
```

第四章 嵌入式程序开发基础

设置断点

Linux 下调试 c 程序时, 在提示符 (gdb) 下输入 b 表示 break。

gcc 的编译可分为 预处理、编译、汇编 和链接等四个阶段。

gcc 命令编译源程序时, 使用 -o 选项指定输出文件的文件名。

为了利用 GDB 调试 C/C++ 程序, 在用 GCC 编译源程序时, 需要利用选项 -g。

GNU 工具集中, 编译、链接工具是 GCC。

执行当前目录下的 myprog 可执行文件, 应在命令行中输入的命令为: ./myprog。

与其他编程语言一样, shell 脚本也允许将数值存放到变量中。shell 脚本文件的变量共有 3 种, 即 环境变量、用户自定义变量 和系统变量。

GNU 工具集中, 执行 make 命令时, 如果不加任何参数, 则该命令会自动执行 makefile 文件中的命令。

makefile 文件中, 其命令行前面的空位是通过键入 TAB (按键) 生成的。

如果已经执行过 make 命令, 现在对 makefile 中的相关内容进行了修改, 但没有修改 C 源程序文件, 现在再次执行 make, 是否会按最新修改的 makefile 来执行? 为什么? 应如何操作才能达到预期结果? 请写出相应命令。

~~不会。若只修改了 makefile 文件而没有修改源程序文件则不运行 makefile 文件; 若修改了源程序文件则只进行修改部分的编译。~~ make -w file

位操作: 欲将 R0 (32 位) 的第 5 位、13 位置为 0, 其它位保持不变, 请写出相应的 C 语句。

~~R0 &= 0xFFFFFFFF~~ R0&=~(0x1<<5);
R0&=~(0x1<<13);

为了利用 GDB 调试 C/C++ 程序, 在编译时需要把调试信息加载到可执行文件中, 则用 GCC 编译源程序时, 需要利用选项 (A)。

A、-g B、-E C、-Wall D、-O2

下列说法中, 不正确的是 (A)。

- A、Shell 程序编写完后还需要用 gcc 编译器编译
- B、可以通过将 shell 程序作为 sh 命令的输入来执行 shell 程序
- C、shell 程序中定义的函数不能有参数
- D、Linux 是免费使用和自由传播的类 UNIX 操作系统, 但它并不是没有版权

不会按最新修改的 makefile 来执行, 因为再次执行 make 时, make 会自动检查 makefile 中相关文件 (c 源程序文件) 的时间戳, 而现在只是 makefile 中的相关内容修改了, 源程序时间戳没有更新, 所以 make 命令不被执行。
make clean
make

执行 shell 文件前, 应先给文件加相应权限, 请写出相应命令。

~~chmod 777 shell~~ chmod (r/w/x) xx.sh chmod 8进制数 xx.sh

编写 shell 程序实现计算 1+2+3+……+100 的和, 并输出所有能被 13 整除的数。

```
#!/bin/sh
s=0
for ((i=1;i<=101;i++))
do
    ((s=s+i))
    If((s%13==0))
```

```
        then
            echo &s
        fi
done
echo sum is $s
```

阅读下面的 shell 程序，写出执行结果。

```
#!/bin/sh
for name in Tom Jack Harry
do
    echo "$name is my friend"
done
Tom is my friend
Jack is my friend
Harry is my friend
```

阅读下面的 shell 程序，写出执行结果。

```
#!/bin/sh

myvar="Hello, world"

echo 1= $myvar

echo 2= "$myvar"

echo 3= '$myvar'

echo 4= \ $myvar

echo 5= \' $myvar \'

echo 6= " $myvar "

echo 7= ''

echo 8= \" $myvar \"
```

```
1= Hello, world
2= Hello, world
3=$myvar
4=$myvar
5=' Hello, world'
6=' Hello, world'
7=" $myvar"
8=" Hello, world"
```

注意前面数字

第五章 嵌入式开发环境建立

在 Linux 中，查看本机的 IP 地址的命令为__`ifconfig`__。

嵌入式开发一般采用__交叉编译__方式，其中宿主机一般是指_有指令操作系统的 PC 机_。

以下叙述中正确的是（ C ）。

- A、宿主机与目标机之间只需要建立逻辑连接即可
- B、在嵌入式系统中，调试器与被调试程序一般位于同一台机器上
- C、在嵌入式系统开发中，通常采用的是交叉编译器
- D、宿主机与目标机之间的通信方式只有串口和并口两种

NFS 是（ C ）系统。

- A、文件
- B、磁盘
- C、网络文件
- D、操作

嵌入式系统的开发通常是在交叉开发环境实现的，交叉开发环境是指（ A ）。

- A、在宿主机上开发，在目标机上运行
- B、在目标机上开发，在宿主机上运行
- C、在宿主机上开发，在宿主机上运行
- D、在目标机上开发，在目标机上运行

要配置 NFS 服务器，在服务器端主要配置（ C ）文件。

- A、/etc/rc.d/rc.inet1
- B、/etc/rc.d/rc.M
- C、/etc/exports
- D、/etc/rc.d/rc.S

什么是交叉编译？为什么要采用交叉编译？

在一种计算机环境中运行的编译程序，能编译出在另外一种环境下运行的代码，。这个编译过程就叫交叉编译。简单地说，就是在一个平台上生成另一个平台上的可执行代码。

大多数目标机没有编译代码的功能，所以需要在有代码编译功能的机器上编译出能在目标机上运行的代码。

第六章 文件处理与进程通信

主要内容：

文件操作、进程管理、进程通信、管道、共享内存

习题：

为了获得当前进程的 ID 号，可使用函数__getpid()__。

（ B ）不是进程和程序的区别。

- A、程序是一组有序的静态指令，进程是一次程序的执行过程
- B、程序只能在前台运行，而进程可以在前台或后台运行
- C、程序可以长期保存，进程是暂时的
- D、程序没有状态，而进程是有状态的

下列进程通信方式中能够实现不同计算机间进程通信的是（ C ）。

- A、管道
- B、消息队列
- C、套接字
- D、共享内存

Linux 系统调用的文件操作主要有哪些函数？其基本用法？

open 打开或创建文件；

close 关闭一个打开的文件，并释放文件描述符；

read 从打开的文件读取数据；

write 向打开的文件写入数据；

lseek 将文件指针定位到相应位置，返回值为最终的偏移量。

Linux 系统中的缓冲区文件与非缓冲区文件有何区别？

缓冲区文件在读写时先将数据写入缓冲区，缓冲区满后再写入磁盘，减少磁盘的读取次数，增加磁盘寿命。

何为文件描述符？该值一般取何值？如果该值为 0，则代表何义？

Linux 的内核利用文件描述符来访问文件。文件描述符是非负整数，它是一个索引值，并指向内核中每个进程打开文件的记录表。当打开一个现存文件或新建一个文件时，内核会向进程返回一个文件描述符。当读写文件时，也需要使用文件描述符来指定待读写的文件。

0 代表标准输入。

什么是进程描述符？怎样获得进程描述符？

进程描述符是一个进程的所有信息，是一个定义的名为 task_struct 的结构体。

什么是僵尸进程？什么是孤儿进程？

僵尸进程：一个进程使用 fork 创建子进程，如果子进程退出，而父进程并没有调用 wait 或 waitpid 获取子进程的状态信息，那么子进程的进程描述符仍然保存在系统中。这种进程称之为僵尸进程。

孤儿进程：一个父进程退出，而它的一个或多个子进程还在运行，那么那些子进程将成为孤儿进程。孤儿进程将被 init 进程(进程号为 1)所收养，并由 init 进程对它们完成状态收集工作

什么是进程？程序中如何产生子进程？怎样区别子进程和父进程？

Linux 中每个正在运行的程序都被称为进程。

一个进程使用 fork 创建子进程

~~使用 getpid() 函数~~，返回值为 0 的是子进程，非 0 的是父进程。

判断fork()函数返回情况

简述嵌入式系统中进程间通信主要采用的形式。

管道、共享内存、FIFO、消息队列、信号

(1) 管道 (Pipe) 及命名管道 (named pipe)：管道可用于具有亲缘关系进程间的通信；命名管道，除具有管道所具有的功能外，它还允许无亲缘关系进程间的通信。

(2) 信号 (Signal)：信号是在软件层次上对中断机制的一种模拟，它是比较复杂的通信方式，用于通知进程有某事件发生，一个进程收到一个信号与处理器收到一个中断请求效果上可以说是一样的。

(3) 消息队列 (Message Queue)：消息队列是消息的链接表，包括 Posix 消息队列 SystemV 消息队列。它克服了前两种通信方式中信息量有限的缺点，具有写权限的进程可以按照一定的规则向消息队列中添加新消息；对消息队列有读权限的进程则可以从消息队列中读取消息。

4) 共享内存 (Shared memory)：可以说这是最有用的进程间通信方式。它使得多个进程可以

访问同一块内存空间，不同进程可以及时看到对方进程中对共享内存中数据的更新。这种通信方式需要依靠某种同步机制，如互斥锁和信号量等。

(5) 信号量 (Semaphore)：主要作为进程之间以及同一进程的不同线程之间的同步和互斥手段。

(6) 套接字 (Socket)：这是一种更为一般的进程间通信机制，它可用于网络中不同机器间的进程间通信，应用非常广泛

system() 函数及 exec 函数族均可在进程中开始其它进程，但二者在执行过程上有区别。简述其区别。

system() 函数可以在进程中开始某一进程，并使用系统函数库来创建新进程。system() 会调用 fork() 产生子进程，由子进程来调用 /bin/sh -c string 来执行参数 string 字符串所代表的命令，此命令执行完后随即返回原调用的进程。

当进程调用一种 exec 函数时，该进程执行的程序完全替换为新程序，而新程序从其 main 函数开始执行。调用 exec 并不创建新进程，所以前后的进程 ID 并不变化，只是将当前进程重新初始化了数据段、代码段和堆栈段，在执行完之后，原调用进程的内容除了进程号外，其他全部被新的进程替换了。

```
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>
int main()
{
    int ret;

    /*调用 execlp()函数，这里相当于调用了"ps -l"命令*/
    ret = execlp("ps", "ps", "-l", NULL);

    printf("ps OK %d",ret);

    return 0;
}
```

以上程序运行的结果。

显示当前文件目录下的所有文件的详细信息。

阅读下面的程序，写出程序运行的结果

```
/* fork.c */
#include <sys/types.h>
#include <unistd.h>
#include <stdio.h>
#include <stdlib.h>

int main(void)
```

```

{
    pid_t result;
    /*调用 fork()函数*/
    result = fork();
    /*通过 result 的值来判断 fork()函数的返回情况，首先进行出错处理*/
    if(result == -1)
    {
        printf("Fork error\n");
    }
    else if (result == 0) /*返回值为 0 代表子进程*/
    {
        printf("The returned value is %d\n
                In child process!!\nMy PID is %d\n",result,getpid()); }
    else /*返回值大于 0 代表父进程*/
    {
        printf("The returned value is %d\n
                In father process!!\nMy PID is %d\n",result,getpid()); }
    return result;}

```

程序运行的结果：

The returned value is 76	The returned value is: 0
In father process!!	In child process!!
My PID is 75	My PID is 76

阅读程序，写出程序的执行结果。假定父子进程均可在一个时间片内执行完，且操作系统每次均先调度子进程。

```

int main()
{
    pid_t result;
    int var=10;
    printf("before fork, var=%d\n", var);
    result=fork();
    if(result<0)
        printf("fork fail\n");
    else if(result>0)
    {
        var++;
        printf("This is parent!\n");
    }
    else
    {
        var--;
        printf("This is child!\n");
    }
    printf("after fork var=%d\n", var);
}

```

```
        return 0;
    }
```

Before fork, var=10

This is child!

After fork var = 9

This is parent!

After fork var=11

如果要实现父子进程的双向通信，可以创建两个管道来进行。

2. 以下程序在父进程和子进程之间创建了一个管道，然后建立它们之间的通信，实现父进程向子进程写数据的功能。说明标号所在行代码的功能。

第七章 嵌入式 Linux 网络开发应用

在嵌入式 Linux 网络编程应用中，端口号的取值范围为 0~65535。

下列关于网络编程的叙述中，错误的是(C)。

- A、一个完整的网络程序应该包含两个独立的程序，它们分别运行在客户端和服务端
- B、相同条件下 UDP 发送数据的速度要比 TCP 快
- C、当使用 UDP 编程时，如果函数 sendto()成功返回，表示系统发出的数据被通讯的对方准确接收到了
- D、端口号是 16bit 的地址码，端口号和 IP 地址构成一个套接字 (socket)

简单说明 TCP 和 UDP 协议有什么区别？

TCP 提供的是面向连接、可靠的字节流服务。双方在交换数据之前必须先建立一个 TCP 连接，能保证数据从一端发送到另一端。

UDP 不提供可靠性，他只是发送但不保证送到，但是传输速度很快。

socket 类型有哪三类，并进行简单描述。

字节流 Socket，基于 TCP，提供可靠字节流传输；

数据报 Socket，基于 UDP，提供不可靠的报文传输；

原始套接字，基于 IP，允许用户直接对 IP 操作。

TCP 协议编程中，server 端程序的步骤是什么？

Socket() -> bind() -> listen() -> accept() -> recv() -> send() -> recv() -> close()

创建套接字，绑定 ip 地址和端口，监听，建立连接，接受数据，写数据，关闭

TCP 协议编程中，client 端程序的步骤是什么？

Socket() -> connect() -> send() -> recv() -> close()

创建套接字，建立连接，发送数据，接受数据，关闭/断开连接

当服务器与客户端建立连接后，如果在客户端和服务端中都调用了 recv() 函数，通信如何进行？为什么？

recv() 一直等待，直到协议把数据读取出来。但是因为两边都在等待，所以通信会阻塞。

第八章 嵌入式驱动程序设计

嵌入式 linux 系统中，设备文件分为字符设备、块设备和网络接口设备三种。

加载 Linux 驱动程序使用命令 insmod。

卸载 Linux 驱动程序使用命令 rmmmod。

完全把系统软件和硬件部分隔离开来的是硬件抽象层，从而大大提高了系统的可移植性。

(A) 完全把系统软件和硬件部分隔离开来，从而大大提高了系统的可移植性。

A、硬件抽象层 B、驱动映射层 C、硬件交互层 D、中间层

在关于设备驱动的描述中，下面哪项是错误的 (D)。

- A、操作系统通过各种驱动程序来驾驭硬件设备
- B、操作系统为硬件提供统一的操作方式
- C、操作系统最基本的组成部分是硬件驱动程序
- D、常见的驱动程序作为内核模块动态加载，比如声卡驱动、网卡驱动、CPU、PCI 总线等

每个设备文件都对应有两个设备号，而标识该设备的种类，也标识了该设备所使用的驱动程序的类别号是指(A)

A、主设备号 B、次设备号 C、块设备号 D、字符设备号

写出下列英文缩写的中文含义。

RTOS IPC GPIO

RTOS: 实时操作系统

IPC: 进程间通信

GPIO: 通用 I/O 接口

设备驱动程序的作用是什么？

驱动程序是内核的一部分，是操作系统内核和机器硬件之间的接口，它由一组函数和一些私有数据组成，是连接应用程序与具体硬件的桥梁。

驱动程序负责将应用程序如读、写等操作正确无误地传递给相关的硬件，并使硬件能够做出正确反应的代码。驱动程序像一个黑盒子，它隐藏了硬件的工作细节，应用程序只需要通过一组标准化的接口实现对硬件的操作。

字符设备驱动程序开发的流程主要是什么？

- ① 编写设备驱动程序及 makefile 文件
- ② 执行 make 命令，生成驱动程序文件 (*.ko)
- ③ 进入 root，执行 mknod 设置设备进入点（在/dev 目录中）
- ④ 执行 insmod 命令加载驱动程序（lsmod 可查看）
- ⑤ 编写用户验证程序，并编译生成可执行程序
- ⑥ 执行用户验证程序，查看运行结果
- ⑦ 运行 dmesg 命令查看内核中驱动程序的输出
- ⑧ 执行 rmmmod 命令卸载驱动，rm 命令删除设备进入点

字符设备驱动程序的主体框架？根据程序写出相应的注释。（第 8 章 PPT 41 页）

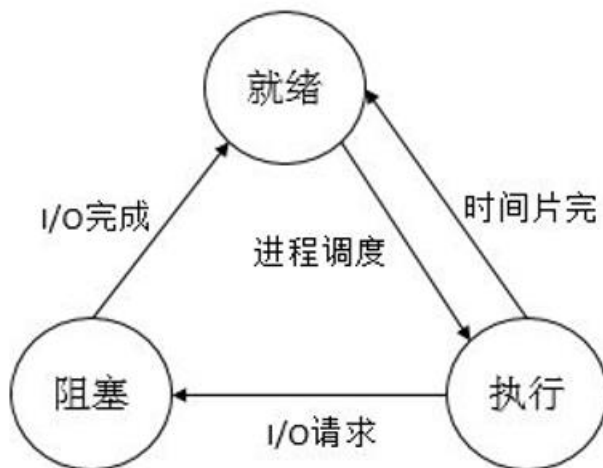
```
struct file_operations { // <linux/fs.h>
    struct module *owner;
    loff_t (*llseek) (struct file *, loff_t, int);
    ssize_t (*read) (struct file *, char __user *, size_t, loff_t *);
    ssize_t (*write) (struct file *, const char __user *, size_t, loff_t *);
    int (*readdir) (struct file *, void *, filldir_t);
    unsigned int (*poll) (struct file *, struct poll_table_struct *);
    long (*unlocked_ioctl) (struct file *, unsigned int, unsigned long);
    long (*compat_ioctl) (struct file *, unsigned int, unsigned long);
    int (*mmap) (struct file *, struct vm_area_struct *);
    int (*open) (struct inode *, struct file *);
    int (*release) (struct inode *, struct file *);
    int (*fsync) (struct file *, int datasync);
    int (*aio_fsync) (struct kiocb *, int datasync);
    int (*fasync) (int, struct file *, int);
    int (*lock) (struct file *, int, struct file_lock *);
    .....};
```

13:27:56

嵌入式系统开发与应用-题库

一、选题题

- 1、下列关于网络编程的叙述中，错误的是（ C ）。
- A、 一个完整的网络程序应该包含两个独立的程序，它们分别运行在客户端和服务端
 - B、 相同条件下 UDP 发送数据的速度要比 TCP 快
 - C、 当使用 UDP 编程时，如果函数 `sendto()` 成功返回，表示系统发出的数据被通讯的对方准确接收到了
 - D、 端口号是 16bit 的正整数，端口号和 IP 地址构成一个套接字（socket）
- 2、以下关于进程的三种状态之间的转换，正确的是（ C ）
- A、 就绪态的进程可能会因为等待 I/O 操作而转换为阻塞态
 - B、 阻塞态的进程，当其等待的 I/O 操作完成后，马上转为执行态
 - C、 执行态的进程既可以转为就绪态，也可能转为阻塞态
 - D、 进程可以在三种状态间任意转换



3、

下面关于哈佛结构描述正确的是（ A ）

- A、 程序存储空间与数据存储空间分离
- B、 存储空间与 IO 空间分离
- C、 程序存储空间与数据存储空间合并
- D、 存储空间与 IO 空间合并

4、通常情况下，用户的应用程序是运行在（ A ）

- A、 用户空间
- B、 系统空间
- C、 内核空间
- D、 内存空间

5、表达式 $a \mid = (0x1 \ll 20)$ 的功能是（ A ）。注：数据最低位为第 0 位

- A、 将 a 的第 20 位置为 1，其它位不变
- B、 将 a 的第 20 位置为 1，其它位全部清 0
- C、 将 a 的第 21 位置为 1，其它位不变
- D、 将 a 的第 21 位置为 1，其它位全部清 0

6、BootLoader 首先完成系统硬件的初始化，然后把操作系统内核从 flash 区拷贝到 ram 区，并跳转到内核的入口，将系统控制权交给操作系统。以下不属于 BootLoader 的是（ A ）

- A、 BusyBox
- B、 LILO
- C、 GRUB
- D、 U-Boot

7、在嵌入式操作系统与硬件之间，新加入的硬件抽象层，其英文简称为(B)

- A、 HSC
- B、 HAL
- C、 BSP
- D、 DSP

8、以下叙述中，不符合 RISC 指令系统特点的是(B)。

- A、 指令长度固定，指令种类少
- B、 寻址方式种类丰富，指令功能尽量增强
- C、 设置大量通用寄存器，访问存储器指令简单
- D、 选取使用频率较高的一些简单指令

9、shell 脚本文件要执行，应(C)

- A、 将之编译生成可执行文件
- B、 将之文本编译、链接后生成可执行程序
- C、 用 chmod 命令将该文本文件赋予可执行权限即可
- D、 文本文件不能执行

10、不存盘退出 vi 的指令是(B)。

- A、 q
- B、 q!
- C、 w
- D、 wq

11、主要用于 NOR 型 Flash 存储器的文件系统是(D)。

- A、 ext3

- B、 ntfs
- C、 YAFFS
- D、 JFFS

12、欲将一个 32 位的寄存器 R0 的第 20~23 位置为 1，其它位保持不变，则以下表达式正确的是（ A ）

- A、 $R0 \mid= (0x0f \ll 20)$
- B、 $R0 \&= (0x0f \ll 20)$
- C、 $R0 \mid= (0x0f \ll 19)$
- D、 $R0 \&= (0x0f \ll 19)$

13、下面（B）不属于嵌入式系统的三大基本特征？

- A、 专用性
- B、 体积微小
- C、 嵌入性
- D、 计算机系统

14、shell 脚本的第一句必须指定解释的 Shell，通常固定为（ C ）

- A、 # /bin/bash
- B、 #! bin/bash
- C、 #! /bin/bash
- D、 #! bash

15、关于程序和进程，以下说法错误的是（ C ）。

- A、 程序是一个静态的指令集合，并不占有系统的运行资源
- B、 进程是一个动态的概念，是正在运行的程序
- C、 进程就是程序的另一种叫法

D、 一个程序可以产生多个进程

16、（ A ）完全把系统软件和硬件部分隔离开来，从而大大提高了系统的可移植性。

A、 硬件抽象层

B、 驱动映射层

C、 硬件交互层

D、 中间层

17、表达式 $a \&= (\sim 0x1)$ 的功能是（ A ）

A、 将第 0 位清 0，其它位保持不变

B、 将第 0 位置为 1，其它位不变

C、 将第 0 位清 0，其它位置为 1

D、 将第 0 位置为 1，其它位清 0

18、在 shell 脚本中，要使用自定义变量，应采用的格式为（ C ）

A、 变量名

B、 \$(变量名)

C、 \$变量名

D、 (\$变量名)

19、在 vi 编辑器的命令模式中，删除一行的命令是：（ B ）

A、 yy

B、 dd

C、 pp

D、 xx

20、同 CISC 相比，下面哪一项不属于 RISC 处理器的特征（ D ）

- A、 采用固定长度的指令格式，指令规整、简单
- B、 减少指令数和寻址方式，使控制部件简化，加快执行速度
- C、 数据处理指令只对寄存器进行操作，只有加载/存储指令可以访问存储器，以提高指令的执行效率，同时简化处理器的设计
- D、 RISC 处理器都采用哈佛结构

21、以下关于 `printf()` 函数，错误的是（ B ）

- A、 `printf()` 函数的用法与 `printf()` 函数类似
- B、 `printf()` 函数执行时，也是直接向屏幕输出指定信息
- C、 `printf()` 函数中，可以指定输出时执行的级别
- D、 `printf()` 函数是将输出信息输出在内核中的指定缓冲区

22、Linux 启动时，最先运行的模块是（ B ）。

- A、 系统内核
- B、 BootLoader
- C、 系统初始化程序
- D、 文件系统

23、以下说法错误的是（ B ）

- A、 `make` 命令在执行时，会自动检查时间戳
- B、 `make` 命令检查时间戳时，发现有源程序文件更新了，则自动编译所有的程序
- C、 `make` 命令执行时，发现有源程序文件更新了，则只编译修改的文件，并重新生成最终的程序
- D、 `make` 命令执行时，如果仅修改了 `makefile` 文件，而源程序文件没有修改，则 `make` 文件不被执行

24、通过 VMware 来安装 Linux 操作系统，以下说法正确的是（ B ）

- A、 因为安装新的操作系统会破坏以前系统中的文件，因此在安装前应对文件进行备份
- B、 VMware 作为虚拟机，只相当于运行在 Windows 中的一个应用程序，不会破坏原有的系统和文件
- C、 VMware 中，因为虚拟机的硬件较低，所以只能安装纯命令方式的 Linux 操作系统
- D、 在 VMware 中启动 Linux 后，原系统不再可用

25、通常，对嵌入式系统的开发环境（如编译系统），是位于（ A ）

- A、 宿主机中
- B、 目标机中
- C、 宿主机和目标机中都有
- D、 取决于哪边安装了 windows 系统

26、以下说法正确的是（ A ）

- A、 头文件一般是文本文件，而库文件是编译后的二进制文件
- B、 静态链接库用的更多，因为这种方式程序更小，运行速度更快
- C、 在#include 命令中，用尖括号< >指定的文件，其搜索的范围更大
- D、 在#include 命令中，用双引号" "指定的文件，其搜索的范围仅在系统指定的目录中

27、下面的存储器，不能直接在芯片内执行程序的是（ D ）

- A、 SRAM
- B、 DRAM
- C、 NOR Flash
- D、 NAND Flash

28、fopen、fread、fwrite...等函数,与 open、read、write...等函数的区别是(B)

- A、 前者是系统调用函数,后者是库函数调用
- B、 前者是库函数调用,后者是系统调用函数
- C、 前者是非缓冲区操作方式,后者是缓冲区操作方式
- D、 前者更高级,后者更低级

29、内核驱动模块与应用程序相比,以下说法错误的是(C)

- A、 内核驱动程序运行于内核中
- B、 内核驱动程序可以完成对硬件的操作
- C、 内核驱动程序也是由 C 语言编写,从 main() 函数开始执行
- D、 加载内核驱动程序是通过命令 insmod 来进行

30、在 vi 编辑器中的命令模式下,键入以下(C)命令不能进入编辑状态。

- A、 a
- B、 i
- C、 d
- D、 o

31、嵌入式应用通常需要考虑低功耗设计,即俗称的“省电”,这是由嵌入式系统的(D)

- A、 专用性特点决定的
- B、 计算机系统特点决定的
- C、 资源受限特点决定的
- D、 功耗约束特点决定的

32、下面不属于嵌入式操作系统的是(B)

- A、 VxWorks

- B、 Windows
- C、 Android
- D、 uC/OS

33、以下哪个指令能够完成从 ARM 指令集跳转到 Thumb 指令集：（ A ）

- A、 BX 指令
- B、 B 指令
- C、 BL 指令
- D、 SWI 指令

34、关于 fork() 函数，以下说法错误的是（ D ）

- A、 fork() 函数的功能是创建调用进程的副本
- B、 调用 fork() 函数，会有两次返回值
- C、 由 fork() 函数创建的新进程被称为子进程
- D、 子进程从头开始执行

35、关于不同的进程间通信方式，以下说法错误的是（ B ）

- A、 管道只能用于具有亲缘关系的进程间通信
- B、 管道只能实现进程间的单向通信
- C、 共享内存方式可以实现多个进程间通过访问同一内存空间实现进程间通信
- D、 套接字（Socket）通信方式既可用于网络、也可用于本机中的不同进程通信

36、关于 Linux 下面说法正确的是（ C ）

- A、 路径名以 “\” 符号分割
- B、 命令和路径名大小写不敏感
- C、 文件系统是从 “/” 开始的统一的目录空间

D、 文件系统中诸如 C:、D:之类的驱动器盘符

37、嵌入式片上系统的英文简称是(D)

A、 MPU

B、 MCP

C、 DSP

D、 SoC

38、在进行嵌入式开发时，需要将宿主机与目标机连接起来。通常，连接的方式不包括以下的(B)

A、 串口

B、 并口

C、 网线

D、 JTAG

39、将高速缓存分为指令缓存和数据缓存的体系结构是(B)

A、 冯·诺依曼结构

B、 哈佛结构

C、 RISC

D、 CISC

40、哈佛结构和冯诺依曼结构的区别是(A)。

A、 指令和数据分开存储

B、 不需要程序计数器

C、 统一编址

D、 单一数据总线

41、对于开发能在嵌入式系统中运行的 C 语言程序，以下说法正确的是（ B ）

- A、 使用的编译程序仍为 gcc，编译过程在宿主机中完成
- B、 使用的编译程序为 arm-linux-gcc，编译过程在宿主机中完成
- C、 使用的编译程序仍为 gcc，编译过程在目标机中完成
- D、 使用的编译程序为 arm-linux-gcc，编译过程在目标机中完成

42、下面（B）特性不符合嵌入式操作系统特点。

- A、 实时性
- B、 不可定制
- C、 微型化
- D、 易移植

43、以下不属于 Linux 系统组成的四个主要部分的是（ B ）

- A、 内核
- B、 外设
- C、 Shell
- D、 文件系统

44、在 shell 脚本文件中，用户自定义变量时，以下说法正确的是（ B ）

- A、 变量在使用前需明确说明其类型，再使用
- B、 定义变量可通过“=”来直接对变量赋值来定义
- C、 定义变量时，“=”的两边可加空格，以使程序更清楚
- D、 自定义变量名不能与系统变量或环境变量同名

45、通常所说的 32 位微处理器是指（ C ）

- A、 地址总线的宽度为 32 位

- B、 处理的数据长度只能为 32 位
- C、 CPU 字长为 32 位
- D、 通用寄存器数目为 32 个

46、嵌入式系统的出现，可追溯到 (B)

- A、 20 世纪 50 年代
- B、 20 世纪 60 年代
- C、 20 世纪 70 年代
- D、 20 世纪 80 年代

47、每个设备文件都对应有两个设备号，而标识该设备的种类，也标识了该设备所使用的驱动程序的类别号是指 (A)

- A、 主设备号
- B、 次设备号
- C、 块设备号
- D、 字符设备号

48、在 gdb 中，想要查看变量的值的命令是 (C)

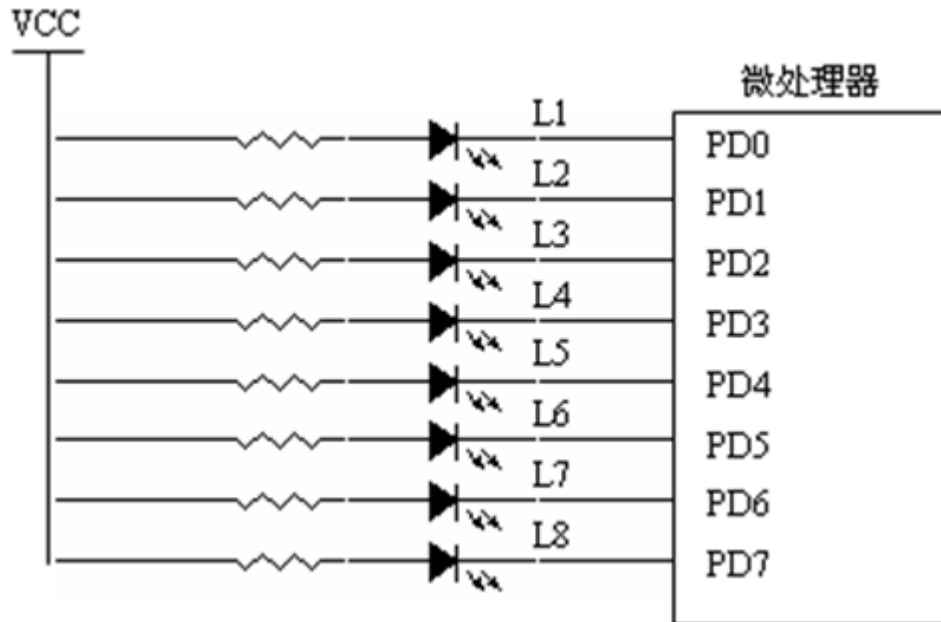
- A、 info
- B、 list
- C、 print
- D、 see

49、ARM7 处理器中，采用了 (A) 级流水线。

- A、 3
- B、 4
- C、 5

D、 6

50、下图中，欲要点亮 L2 灯，则端口 PD 中输出的数据应为（ B ）



- A、 FEH
- B、 FDH
- C、 01H
- D、 02H

51、以下关于通过共享内存通信的说法中，正确的是（ A ）

- A、 进程间通信的共享内存，是位于操作系统的内核中
- B、 创建的共享内存的大小，被固定为 4096 字节
- C、 对所创建的共享内存，进程都对其具有读写操作权限，
- D、 `shmdt()` 函数的功能是将共享内存从系统内核中注销

52、文件描述符的值如果为 0，表示该文件为（ B ）

- A、 打开错误

- B、 标准输入
- C、 标准输出
- D、 标准错误

53、语句 `currpos = lseek(fd, 0, SEEK_END);`的功能是 (B)

- A、 返回文件的当前指针位置
- B、 返回文件的大小
- C、 指向文件的开头位置
- D、 指向文件的当前位置

54、下列说法中，不正确的是 (A)。

- A、 shell 程序编写完后还需要用 gcc 编译器编译
- B、 可以通过将 shell 程序作为 sh 命令的输入来执行 shell 程序
- C、 shell 脚本是文本文件
- D、 Linux 是免费使用和自由传播的类 UNIX 操作系统，但它并不是没有版权

55、下面哪种不属于 vi 三种工作模式之一 (D)。

- A、 命令行模式
- B、 插入模式（编辑模式）
- C、 底行模式
- D、 工作模式

56、宿主机与目标机之间进行文件的传递，不能采用以下 (C) 方式。

- A、 通过串口
- B、 NFS（网络文件系统）
- C、 ping

D、 FTP（文件传输协议）

57、关于 system() 函数与 exec() 函数，以下说法错误的是（ C ）

- A、 system() 函数与 exec() 函数都可以在当前进程中调用进程
- B、 system() 函数调用其它进程，执行完成后会返回到原进程继续执行后面的语句
- C、 exec() 函数调用其它进程，执行完成后会返回到原进程继续执行后面的语句
- D、 exec() 函数调用其它进程时，会将自身原有内容全部覆盖

58、read（）函数的返回值如果小于参数中指定的字节数，说明（ C ）

- A、 本次读取文件不成功
- B、 本次读取文件错误
- C、 本次读取已到文件末尾
- D、 缓冲区不小不足

59、在 makefile 文件中，使用“%”用于表示（ D ）

- A、 百分比
- B、 一个字符
- C、 多个字符
- D、 一个或多个字符

60、在 ARM 体系结构中，_____寄存器作为连接寄存器，当进入子程序时或者处理器响应异常的时候，用来保存 PC 的返回值；_____寄存器作为处理器的程序计数器指针。（ C ）

- A、 R0，R14
- B、 R13，R15
- C、 R14，R15

D、 R14, R0

61、无名管道与命名管道的主要区别是（ D ）

- A、 前者是通过系统内核来进行进程间通信的
- B、 命名管道是通过在磁盘上创建文件来实现不同进程间通信
- C、 有名管道可以实现双向通信，而无名管道则只能单向通信
- D、 前者只能在具有亲缘关系的进程间通信，而后者则无此限制

62、对于 ARM7 三级流水线，当一条指令被译码时，上一条指令正被（ D ）。

- A、 取指
- B、 译码
- C、 PC 值递增
- D、 执行

63、如果从网上下载有“ubuntu-14.04.6-desktop-i386.iso”的安装镜像文件，则安装后该 ubuntu 系统为（ C ）的操作系统。

- A、 纯命令 32 位
- B、 纯命令 64 位
- C、 图形版 32 位
- D、 图形版 64 位

64、网络文件系统是（ C ）。

- A、 tmpfs
- B、 ntfs
- C、 NFS
- D、 vfat

65、wait()函数的功能是 (B)

- A、 使当前进程进入睡眠状态
- B、 等待子进程运行完毕
- C、 等待父进程运行完毕
- D、 等待进程运行完毕

66、为了利用 GDB 调试 C/C++程序，在编译时需要把调试信息加载到可执行文件中，
则用 GCC 编译源程序时，需要利用选项 (A)。

- A、 -g
- B、 -E
- C、 -Wall
- D、 -O2

67、进程号是一个 (B)

- A、 正数
- B、 非负正整数
- C、 int 型的数
- D、 整数

68、以下关于客户端程序和服务器程序，正确的说法是 (A)

- A、 网络通信时，要先有服务器程序启动，等待客户端程序运行并向服务器端口发起连接
- B、 有时不一定有服务器程序，比如当访问通过浏览器打开网站时
- C、 使用 QQ 进行即时通信时，不需要服务器
- D、 网络通信时，应先启动客户端程序，以向服务器发出连接请求

69、ARM 处理器的工作模式有 (C)

- A、 5 种
- B、 6 种
- C、 7 种
- D、 8 种

70、下面哪点不是嵌入式操作系统的特点？（ C ）

- A、 内核精简
- B、 专用性强
- C、 功能强大
- D、 高实时性

71、以下关于 fifo 管道的说法中，错误的是（ C ）。

- A、 fifo 管道即是有名管道
- B、 创建了 fifo 管道后，仍然使用 read()、write() 等系统调用对管道进行读写操作
- C、 创建了 fifo 管道后，当进程结束后，则该管道自动被系统注销
- D、 fifo 管道可以用于无亲缘关系的进程间通信

72、指令和数据共享同一总线的体系结构是（ A ）

- A、 冯·诺依曼结构
- B、 哈佛结构
- C、 RISC 架构
- D、 CISC 架构

73、缓冲区操作文件方式与非缓冲区操作文件方式的区别，以下说法错误的是（ C ）

- A、 缓冲区操作文件方式在读写文件时，会先在缓冲区中进行

- B、系统调用函数 `open()`、`read()`、`write()` 等函数，都是非缓冲区调用函数
- C、缓冲区操作文件方式中，进入缓冲区的内容会马上写入到文件中
- D、非缓冲区操作文件方式会频繁地切换到内核空间以读写文件

74、通过 `write()` 函数向文件中写入数据后，（ B ）

- A、程序末尾必须用 `close()` 函数关闭文件，这样相应数据才能写入文件中
- B、程序末尾可以不用 `close()` 函数关闭文件，因为 `write()` 函数是非缓冲区函数
- C、程序末尾必须用 `close()` 函数关闭文件，因为打开的文件必须关闭
- D、程序末尾可以不用 `close()` 函数关闭文件，因为缓冲区中的数据会自动写入文件

75、Linux 的内核利用文件描述符来访问文件。文件描述符是一个（ C ）

- A、整数
- B、整型数
- C、大于或等于 0 的整数
- D、任意整数

76、在 shell 脚本中，以下说法正确的是（ C ）

- A、单引号 `' '` 与双引号 `" "` 的功能相同
- B、单引号 `' '` 与双引号 `" "` 的功能不相同，前者只能用于字符
- C、单引号 `' '` 中的字符串，如果包含保留字符，则保留字符失效，按原样输出
- D、双引号 `" "` 中的字符串，如果包含保留字符，则保留字符失效，按原样输出

77、表达式 `a &= 0x00ff` 的功能是（ C ）

- A、将 `a` 的低 8 位全部置为 1，高位置为 0
- B、将 `a` 的低 8 位全部置为 1，高位保持不变
- C、将 `a` 的低 8 位保持不变，高位清为 0

D、 将 a 的低 16 位保持不变，高位清为 0

78、微机系统总线所使用的频率，被称为（ C ）

A、 主频

B、 倍频

C、 外频

D、 分频

79、嵌入式应用通常考虑进行运行速度优化和代码尺寸优化，这是由嵌入式系统的（ C ）。

A、 专用性特点决定的

B、 计算机系统特点决定的

C、 资源受限特点决定的

D、 功耗约束特点决定的

80、关于 open（）函数的三个参数，以下说法错误的是（ B ）

A、 第一个参数为要打开的文件的名字（可包含路径）

B、 第二个参数中，所有选项均可同时使用，使用时各参数用“|”分隔

C、 第三个参数为 8 进制数，用于指示被打开的文件的权限

D、 第三个参数只有在新建文件时才需要

81、关于管道，以下说法错误的是（ C ）

A、 管道是半双工的，数据只能单向流动；

B、 管道只能用于具有亲缘关系的进程之间的通信；

C、 管道作为一种独立的文件系统，其实质是通过磁盘文件来实现进程间通信

D、 管道中，写入的数据每次都添加在管道缓冲区的末尾

82、在 vi 编辑器中，在命令行状态复制和粘贴的命令是（ C ）

- A、 c 和 v
- B、 ctrl+c 和 ctrl+v
- C、 y 和 p
- D、 c 和 p

83、在关于设备驱动的描述中，下面哪项是错误的（ ~~B~~ ）。

- A、 操作系统通过各种驱动程序来驾驭硬件设备
- B、 操作系统为硬件提供统一的操作方式
- C、 硬件驱动程序是操作系统基本的组成部分
- D、 常见的驱动程序作为内核模块动态加载，比如声卡驱动、网卡驱动、CPU、PCI 总线等

84、ARM 处理器中的字是（ C ）位的。

- A、 8
- B、 16
- C、 32
- D、 64

85、shell 脚本是一个（ C ）

- A、 文本文件
- B、 可执行文件
- C、 可执行的文本文件
- D、 二进制文件

二、填空

86、gcc 编译时，只生成目标文件而不生成可执行文件的参数选项是 -__c__。

87、GNU 工具集中，执行 make 命令时，如果不加任何参数，则该命令会自动执行_makefile_文件中的命令。

88、在 Linux 中，所有外部设备的镜像文件一般是存放在根目录下的（ dev ）目录中。（注：请注意大小写）

```
89、 1 #include <stdio.h>
      2 #include <unistd.h>
      3 #include <sys/types.h>
      4 int main()
      5 {
      6     pid_t  pid;
      7     pid = fork( );
      8     printf("PID = %d \n", pid);
      9     return 0;
     10 }
```

设以上程序的输出为：

PID=3014

PID=0

则可知生成的子进程的进程号为__3014__。

90、在 vi 编辑器中，从编辑状态返回到命令行状态，应键入（ **ESC** ）键（写按键的大写英文字母）。

91、为了利用 gdb 调试 C/C++ 程序，在用 gcc 编译源程序时，需要利用选项 - **g**。

92、gcc 命令编译源程序时，使用 - **o** 选项指定输出文件的文件名。

93、如果在程序中，所包含的头文件没有在系统指定的目录中，此时需要在 gcc 编译时用 - **I** 参数来指明头文件所在目录。
注意大写

94、在 socket 网络编程模型中，端口号的取值范围为 **0~65535**。

95、为了实现父子进程通过管道进行通信，则在父进程中应调用 **fork()** 函数创建一个子进程；该子进程创建应在创建管道之 **后**（填前、后）。

96、makefile 中的隐式规则，又被称为 **默认** 规则。通过该方式，告诉 make 使用默认的方式来完成编译任务。

97、Linux 下调试 C 程序时，在提示符（gdb）下输入 b 表示 **break**。
设置断点

98、ARM 字数据存储有两种存储数据的字节顺序，一种是大端格式，另一种是 **小端格式**。

99、在 Socket 编程模型中，传输控制协议的简称是 **TCP**。它是一种 **可靠**（填可靠或不可靠）的传输协议。

100、read（）函数的返回值为 **实际读到的字节数**。

101、阅读下面的 shell 程序，写出执行结果。

1=__ Hello, world __

2=__ \$myvar __

3=__ \$myvar __

```
#!/bin/sh
```

```
myvar="Hello, world"
```

```
echo 1= "$myvar"
```

```
echo 2= '$myvar'
```

```
echo 3= \ $myvar
```

102、

```
4 int main()
```

```
5 {
```

```
6     pid_t    pid;
```

```
7     pid = fork( );
```

```
8
```

```
9         if(pid==0) {
```

```
10             printf("PID =%d\n",getpid()); }
```

```
11         else {
```

```
12             printf("PID =%d\n",getpid()); }
```

```
13             printf("pid = %d \n", pid);
```

```
14             return 0;
```

```
15 }
```

设以上程序的一组输出为：

PID=2001 父进程自己本身的PID

pid=2002 返回的子进程的PID

则可以推出：

- (1) 该输出为_父_进程的输出。（填父、子）因为第一行不为0
- (2) 另一组输出中的第一行输出为_PID=2002_。子进程自己本身的PID
- (3) 另一组输出中的第二行输出为__pid=0__。子进程没有子进程了，所以为0

103、嵌入式微处理器的架构分为__RISC__和 CISC 两种类型。（英文字母应大写）

104、在 gdb 中，设置了断点后，对程序进行单步跟踪，如果想要跟踪进入函数的内部执行，则应使用命令_step_；而只想把函数调用作为一条语句执行，则应使用命令_next_。

105、编写 shell 程序实现计算 $1+2+3+\dots+100$ 的和，并输出所有能被 13 整除的数。

上面有

106、在 makefile 中可以使用变量。变量声明时应先通过_=_符号来赋初值；使用时，在变量名前应加__\$__符号。

107、使用 `pipe(fd);` 语句创建了管道之后，则二元数组中的 `fd[0]` 固定用于对管道的__读__操作，而 `fd[1]` 固定用于对管道的__写__操作。

108、__shell__是 Linux 系统下的命令解释器，也是使用 Linux 系统的主要环境，它提供了用户与内核进行交互操作的一种接口。

109、在 Linux 中，如果想要临时切换到管理员权限（超级用户），应使用的命令是（ sudo ）

110、如果从网上下载有“ubuntu-14.04.6-server-i386.iso”的安装镜像文件，则安装后该 ubuntu 系统为__32__位的操作系统。

111、对管道的读操作，是通过__read()__系统调用函数来实现的；对管道的写操作，是通过__write()__系统调用函数来实现的；（写函数名称）

112、设在程序中已经对管道的文件描述符进行了如下定义，int fd[2];则在程序中创建管道的语句是__pipe(fd)__;

113、如果要获取当前进程的进程号，则应调用的函数为__getpid()__。

114、在 Linux 中，如当前已经是管理员权限（超级用户），如果想要立即关闭计算机电源，则在命令行中输入的命令是（ shutdown -P 0 ）

115、shell 脚本文件的变量共有 3 种，即系统变量、__环境变量__和__用户自定义变量__。

116、ARM 核有两个指令集，分别是： ARM 指令集和____THUMB____指令集。（英文全部用大写字母）

117、lseek（）函数中，如果要设置文件指针的偏移方式为从当前位置处开始，则该参数为__SEEK_CUR__。

118、通常在设备驱动程序中，对字符设备的读操作使用__read()__函数来进行；对字符设备写操作使用__write()__函数来进行。

119、卸载 Linux 驱动程序使用命令_rmmod__。

120、module_init()函数是模块_加载_函数，当通过__insmod_命令加载内核模块时，该函数会自动被内核执行，完成模块的相关初始化工作。

121、在 Socket 编程模型中，用户数据报协议的简称是__UDP__。它是一种__不可靠_（填可靠或不可靠）的传输协议。

122、从 gdb 中退出的命令是__quit__。

123、加载 Linux 驱动程序使用命令__insmod__。

124、管道中，写入的数据每次都添加在管道缓冲区的__末尾__；而每次读出数据时，都是从缓冲区的__头部__读。

125、嵌入式 Linux 系统中，设备文件分为__字符__设备、__块__设备和__网络接口__设备三种。

126、makefile 文件中，其命令行前面的空位是通过键入__TAB_键（按键名称写大写英文字母）生成的。

127、嵌入式微处理器 ARM7 采用了典型的三级流水线，即一条指令的执行可分为取指、__译码__和执行三个步骤。

128、要查看系统的搜索路径，应使用的环境变量是_PATH__。

129、执行当前目录下的 myprog 可执行文件，应在命令行中输入的命令为：__./myprog__。

130、用于统计程序执行时间的命令（程序）是__time__。

131、通用 I / O 接口的英文简称为__GPIO__。

132、在使用 gcc 进行编译程序时，如果没有使用-o 参数指定输出文件，则系统默认的输出文件是__a.out__。

133、在 ARM 系统的大端格式中，如果存储的数据为 0x11223344，则其在内存中由高位到低位存放的为 0x__44332211__。（请注意，按 16 进制格式书写，前缀 0x 不用再写）

134、进程间通信时，不同的进程间使用一个公共的缓冲区来进行信息交换，该缓冲区位于__内核__中。

135、Linux 的文件系统只有一个文件树，整个文件系统是以一个（ / ）为起点的。（写中文名称或符号）

136、在操作系统中，IPC 是指__进程间通信__。

Interprocess Communication

137、socket 的编程模型中，公共端口的地址值应小于__1024__。

138、如果有两个文件：myprog.c、proc1.c，其中 proc1.c 中有 myprog.c 中所使用的函数。现欲将之编译生成可执行程序 myproc，则相应的编译命令应为__gcc myprog.c proc1.c -o myproc__。

gcc myprog.c proc1.c -o myproc

139、open 函数的返回值为打开的文件的__文件描述符__。

140、孤儿进程与僵尸进程，对系统危害更大的是__僵尸进程__。

三、操作练习

144、

 [demo.rar](#)

 [实验 12 Linux 驱动程序结构实验.doc](#)

145、请完成附件内容后上传为附件。

 [6.1 文件处理.docx](#)

146、

 [实验 11 Linux 系统网络编程.doc](#)


147、

 [6.2-3 Linux 进程控制及进程通信.doc](#)

148、

 [vi、gcc、gdb 课后练习.docx](#)

149、请在 Linux 中完成附件中的操作。

 [makefile 及 shell 练习.docx](#)

150、创建目录 dir1、dir2，并将 hello.c 文件分别复制到两个目录中，并用 ls -l 命令显示。将包含以上命令及执行结果的命令窗口抓图如下：

```
Mkdir dir1
```

```
Mkdir dir2
```

```
Cp hello.c /dir1/
```

```
Cp hello.c /dir1/
```

```
Cd dir1
```

```
Ls -l
```

151、在 dir2 目录中，用数字格式改变 hello.c 文件的权限，设置该文件为用户拥有读、写、执行，用户组有读、写，其他用户只有读权限，写出相应命令，并用 ls 命令验证。

```
Chomd 763
```