**作业一**

**教材第16-18页**

**1.选择题 （写在教材上）**

**2.填空题 （写在教材上）**

**3.简答题 （写在作业本上）**

**（1）简述嵌入式系统的定义和特点。**

**答：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，应用系统对功能，可靠性，成本，体积，功耗有严格要求的专用计算机系统。**

**嵌入式系统的特点：专用性强，实时性好，可裁剪性好，可靠性高和功耗低。**

**（3）简述嵌入式系统的组成。**

**答：嵌入式系统是由硬件和软件两大部分组成，硬件是整个系统的物理基础，它提供软件的运行平台和通信接口；软件用于控制系统的运行。**

**嵌入式系统的硬件可以分为3部分，即微处理器，外围电路和外部设备。**

**嵌入式系统的软件结构可分为4个层次，即板级支持包，实时操作系统，应用程序接口和应用程序。**

**（5）简述嵌入式微处理器的特点。**

**答：1.功耗低，是嵌入式系统设计的首要目标。**

**2.集成丰富的外围设备接口，可以提高产品集成度，缩小体积，提高可靠性。**

**3.对实时多任务有很强的支持能力。**

**作业二**

**教材第72-75页**

**1.选择题 （写在教材上）**

**2.填空题 （写在教材上）**

**3.简答题 （写在作业本上）**

**（5）分析程序状态寄存器各位的功能描述，并说明C,Z,N,V在什么情况下进行置1和清零。**

**答：（1）CPSR中的[31:27]为条件标志位，具体含义如下：**

**N：符号标志位。当用两个补码表示的带符号数进行运算时，N=1表示运算的结果为负数；N=0表示运算的结果为正数或零。**

**Z：结果是否为0的标志。Z=1，表示运算结果为0；Z=0，表示运算结果为非0。**

**C：进位或借位标志位。加法运算结果产生了借位时C=1，减法运算产生了借位时C=1，否则为0；包含移位操作的非加/减运算指令，C为移出值的最后一位；其他运算指令，C的值通常不变。**

**V：溢出标志位。对于加减法运算指令，V=1表示符号位溢出，其它指令的影响V位。**

**（2）CPSR中的[7:0]为控制位，具体含义如下：**

**I：IRQ中断使能位；**

**F：FIQ中断使能位；**

**T：处理器运行状态控制位；**

**M[4:0]：运行模式位。**

**（3）其它标志位。**

**（8）分析下列每条语句的功能，并说明程序所实现的功能。**

**CMP R0， #0**

**MOVEQ R1， #0**

**MOVGT R1， #1**

**解：CMP R0， #0 ;** **将R0的值减去0，并根据结果设置CPSR的标志位**

**MOVEQ R1， #0 ; 若R0等于0，Z=1，则将立即数0装入到R1**

**MOVGT R1， #1 ; 若R0大于0，Z=0，N=V，则将立即数1装入到R1**

**程序实现的功能是判断R0的值与0的关系，将结果装入R1，若R0=0则R1=0，若R0>0则R1=1.用C语言描述如下。**

**设R0对应变量a，R1对应变量b，**

**if(a==0)**

**b=0;**

**else if(a>0)**

**b=1;**

**4.程序题（写在作业本上）**

**对程序各条指令进行注释，最后分析整个程序的功能。**

**AREA Example1,CODE,READONLY ; 声明代码段Example1**

**ENTRY ; 标识程序入口**

**CODE32 ; 声明32位ARM指令**

**START ; 标号/标签**

**MOV R0, #0 ; 设置参数，R0=0**

**MOV R1, #10 ; 设置参数，R1=10**

**LOOP ; 标号/标签**

**BL ADD\_SUB ; 调用子程序ADD\_SUB**

**B LOOP ; 跳转到LOOP**

**ADD\_SUB ; 标号/标签**

**ADDS R0, R0, R1 ; R0 = R0 + R1，并根据结果设置CPSR的标志位**

**MOV PC, LR ; 子程序返回，PC=LR**

**END ; 声明文件结束**

**整个程序的功能：程序实现了一个死循环，每次循环R0都增加10。**

**作业三**

**教材第110-113页**

**1.选择题 （写在教材上）**

**2.填空题 （写在教材上）**

**3.简答题 （写在作业本上）**

**（1）简述GCC编译4个过程的主要功能。**

**答：1）预编译过程的主要功能是读取源文件，并对头文件预编译语句和一些特殊符号进行分析和处理。**

**2）编译的主要功能包括检查代码语法和将预编译后的文件转换成汇编语言。**

**3）汇编的主要功能是将汇编语言代码变成目标代码（机器代码）。**

**4）连接的主要功能是连接目标代码，并生成可执行文件。**

**（5）简述Make工具和Makefile的基本结构。**

**答：Make又叫工程管理器，即管理较多的工程文件。它最主要的功能是通过Makefile文件来描述源程序之间的相互依赖关系，并自动完成维护编译工作。Make工具能够根据文件的时间戳自动发现更新过的文件，可以减少编译工作。**

**Makefile基本结构：Makefile可定义文件依赖关系，它由若干规则组成，格式如下。**

**target:dependency**

**<tab键>command**

**（6）简述在Linux操作系统下，用C语言实现文件操作可以采用哪两种方法。**

**答：在Linux操作系统下，用C语言实现文件操作可以采用两种方法，一种是通过C语言库函数调用来实现，另一种是通过Linux系统调用来实现。前者独立于具体操作系统，即在任何操作系统下，使用C语言库函数操作文件按的方法都相同，后者则依赖于Linux操作系统。**

**4.编程题 （写在作业本上）**

**（4）编写一个程序，将系统时间以“year-month-day hour:minute:second”格式保存在time.txt文件中。**

**#include<stdio.h>**

**#include<string.h>**

**#include<fcntl.h>**

**#include<time.h>**

**#define MAX 40**

**int main()**

**{**

**int fd,n,ret;**

**char writebuf[MAX];**

**struct tm \*t;**

**time\_t lt;**

**lt=time(NULL);**

**t=localtime(&lt); //将日历时间转化为本地时间**

**sprintf(writebuf,"%d-%d-%d %d:%d:%d\n",t->tm\_year+1900,t->tm\_mon+1,t->tm\_mday,t->tm\_hour,t->tm\_min,t->tm\_sec); //按指定格式保存时间**

**/\*打开文件，如果文件不存在，则会创建文件\*/**

**fd = open("time.txt", O\_RDWR | O\_CREAT);**

**/\*向文件写入字符串\*/**

**ret = write(fd, writebuf, strlen(writebuf));**

**if (ret < 0){**

**perror("Write Error!"); return 1;**

**}**

**else**

**{**

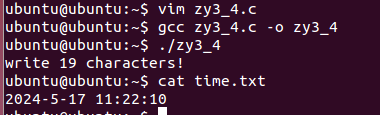
**printf("write %d characters!\n", ret);**

**}**

**/\*关闭时，会自动保存文件\*/**

**close(fd);**

**}**



**作业四**

**教材第144-146页**

**1.选择题 （写在教材上）**

**2.填空题 （写在教材上）**

**3. 简答题（写在作业本上）**

**（1）Linux内核支持多种处理器，如果目标机使用的是ARM处理器核，使用的交叉编译工具链是arm-linux-，内核移植时要指定处理器的类型以及使用的交叉编译工具链，简述具体操作方法。**

**答：具体操作如下。**

**打开内核顶层目录下的Makefile文件，在文件中找到如下内容。**

**ARCH？=$(SUBARCH)**

**CROSS\_COMPILE?=$(CONFIG\_CROSS\_COMPILE:“%”=%)**

**将找到的以上代码修改为如下内容。**

**ARCH？=arm**

**CROSS\_COMPILE?=arm-none-linux-gnueabi-**

**其中，ARCH是CPU架构变量；CROSS\_COMPILE是交叉编译工具链变量。修改完成后，保存文件退出。**

**（2）BootLoader的结构分两部分，简述各部分功能。**

**BootLoader第一部分主要包含依赖于CPU的体系结构硬件初始化的代码，通常都用汇编语言来实现。这个阶段的功能有：基本的硬件设备初始化（屏蔽所有的中断、关闭处理器内部指令/数据Cache等）；为第二阶段准备RAM空间；复制BootLoader的第二阶段代码到RAM；设置堆栈；跳转到第二阶段的C程序入口点。**

**第二阶段通常用C语言完成，以便实现更复杂的功能，也使程序有更好的可读性和可移植性。这个阶段的任务有：初始化本阶段要使用到的硬件设备；检测系统内存映射；将内核映像和根文件系统映像从Flash读到RAM；为内核设置启动参数；调用内核。**

**作业五**

**教材第193-195页**

**1.选择题 （1-15小题，写在教材上）**

**2.填空题 （1-10小题，写在教材上）**

**3.简答题 （写在作业本上）**

**（1）简述驱动程序的功能。**

**答：作为Linux内核的重要组成部分，设备驱动程序主要完成以下的功能：**

**(1) 对设备初始化和释放。**

**(2) 把数据从内核传送到硬件和从硬件读取数据。**

**(3) 读取应用程序传送给设备文件的数据和回送应用程序请求的数据。**

**(4) 检测错误和处理中断。**

**（2）简述驱动程序的组成。**

**答：Linux 设备驱动程序可以分为两个主要组成部分：**

**（1）对子程序进行自动配置和初始化，检测驱动的硬件设备是否正常，能否正常工作。**

**（2）设备服务子程序和中断服务子程序，这两者分别是驱动程序的上下两部分。驱动上部分即设备服务子程序的执行是系统调用的结果，并且伴随着用户态向核心态的演变，在此过程中还可以调用与进程运行环境有关的函数，比如 sleep()函数。驱动程序的下半部分即中断服务子程序。**

**（3）简述驱动程序和应用程序的区别。**

**答：第一，应用程序一般有一个main函数，并从头到尾执行一个任务；驱动程序没有main函数，它在加载时，通过调用module\_init宏，完成驱动设备的初始化和注册工作之后便停止工作，并等待被应用程序调用。**

**第二，应用程序可以和GLIBC库连接，因此可以包含标准的头文件；驱动程序不能使用标准的C库，因此不能调用所有的C库函数，比如输出函数不能使用printf，只能用内核的printk，包含的头文件只能是内核的头文件，比如Linux/module.h。**

**第三，驱动程序运行在内核空间（又称内核态）比应用程序执行的优先级要高很多。应用程序则运行在最低级别的用户空间（又称用户态），在这一级别禁止对硬件的直接访问和对内存的未授权访问。**

**4.编程题 （写在作业本上）**

**（1）假设有某个LED设备的驱动程序demo.c，要编译成目标机上的驱动程序模块demo.ko，已知移植好的Linux内核源代码通过了内核编译，保存在**

**/home/linux/workdir/linux-3.14.fs4412目录下。假设该设备的主设备号为249，次设备号为0，设备文件名为/dev/demo。请完成**

**1）编写Makefile文件；**

**2）写出加载驱动程序的命令；**

**3）写出创建设备文件的命令。**

**答：1）Makefile文件内容如下：**

**obj-m:=demo.o**

**ARCH=arm**

**CROSS\_COMPILE=arm-linux-**

**KERNELDIR:=** **/home/linux/workdir/linux-3.14.fs4412**

**PWD:=$(shell pwd)**

**ALL:**

**$(MAKE) -C $(KERNELDIR) M=$(PWD) modules**

**clean:**

**$(MAKE) -C $(KERNELDIR) M=$(PWD) clean**

**2）加载驱动程序的命令：insmod demo.ko**

**3）创建设备文件的命令：mknod /dev/demo c 249 0**

**作业六**

**教材第219-220页**

**1.选择题 （写在教材上）**

**2.填空题 （写在教材上）**

**3.简答题 （写在作业本上）**

**（1）什么是嵌入式数据库？**

**答：嵌入式数据库一般是指运行在嵌入式系统上，且不启动服务端的轻型数据库，它被应用程序启动，并伴随应用程序的退出而终止。**

**（2）嵌入式数据库和传统数据库从应用层面比较有什么特点？**

**答：从应用层面比较，嵌入式数据库有以下特点：小内核，高性能，成本低，可裁剪和嵌入性。**

**4.编程题 （写在作业本上）**

**（1）编写一个C语言程序实现：创建数据库company.db，在该数据库中建立personnel表，personnel表的字段信息如表1所示，插入表2所示的3条记录，然后查询并在屏幕上显示所有记录。**

表1 personnel表的字段信息

|  |  |  |
| --- | --- | --- |
| 字段 | 类型 | 说明 |
| id | integer | 职员工号，为主键 |
| name | varchar（20） | 职员名称 |
| salary | integer | 职员薪水 |

表2 待插入的记录

|  |  |  |
| --- | --- | --- |
| id | name | salary |
| 001 | Mike | 3248 |
| 002 | Bill | 4789 |
| 003 | Tom | 3899 |

**解：**

**#include<stdio.h>**

**#include<sqlite3.h>**

**int main()**

**{**

**sqlite3 \*db=NULL;**

**int rc, i=0, nrow,ncol;**

**char \*Errormsg, \*\*Result;**

**rc=sqlite3\_open("company.db",&db);**

**if(rc){**

**fprintf(stderr,"can't open database:%s\n",sqlite3\_errmsg(db));**

**sqlite3\_close(db);**

**return 1; }**

**else**

**printf("open database successly!\n");**

**char \*sql="create table personnel(id integer primary key,name varchar(20),salary integer)";**

**sqlite3\_exec(db,sql,0,0,&Errormsg);**

**sql="insert into personnel values(001,'Mike',3248)";**

**sqlite3\_exec(db,sql,0,0,&Errormsg);**

**sql="insert into personnel values(002,'Bill',4789)";**

**sqlite3\_exec(db,sql,0,0,&Errormsg);**

**sql="insert into personnel values(003,'Tom',3899)";**

**sqlite3\_exec(db,sql,0,0,&Errormsg);**

**sql="select \* from personnel";**

**sqlite3\_get\_table(db,sql,&Result,&nrow,&ncol,&Errormsg);**

**printf("row=%d column=%d\n",nrow,ncol);**

**printf("the result is:\n");**

**for(i=0;i<(nrow+1)\*ncol;i++)**

**{**

**printf("%20s",Result[i]);**

**if((i+1)%ncol==0)printf("\n");**

**}**

**sqlite3\_free(Errormsg);**

**sqlite3\_free\_table(Result);**

**sqlite3\_close(db);**

**return 0;**

**}**

