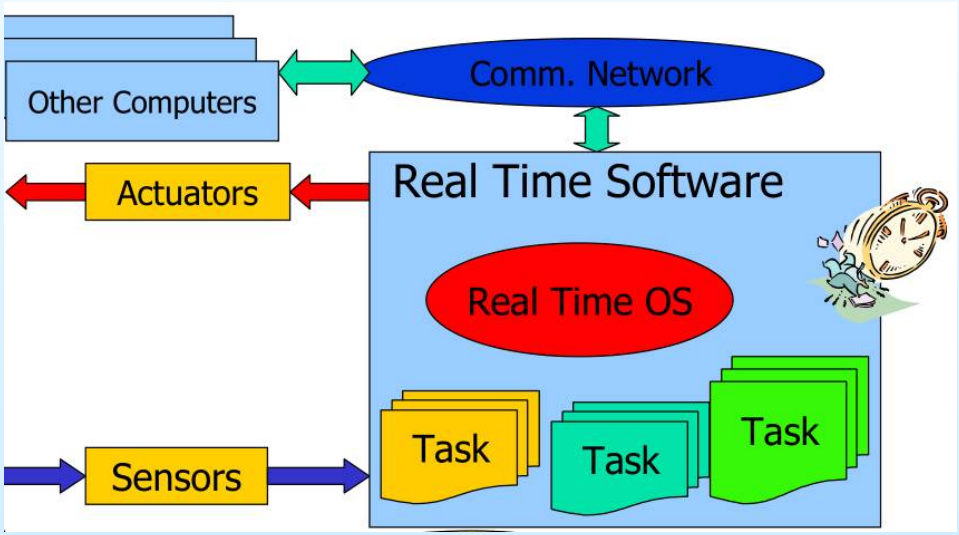
面向机器人工程专业的

《嵌入式实时操作系统实践》课程建设

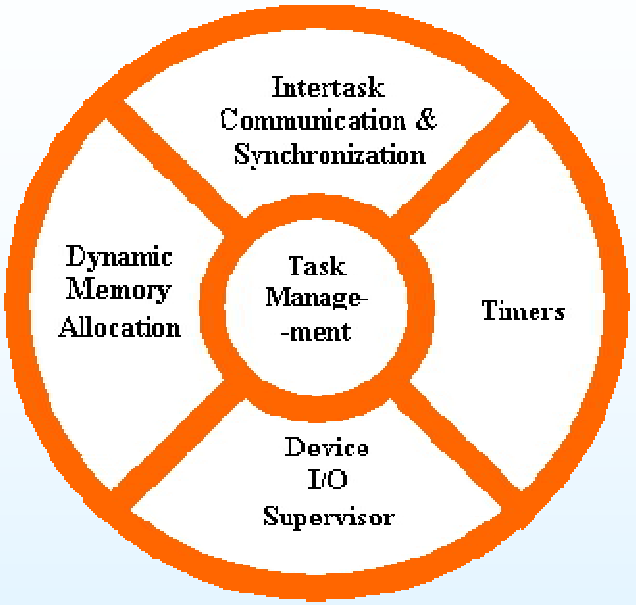


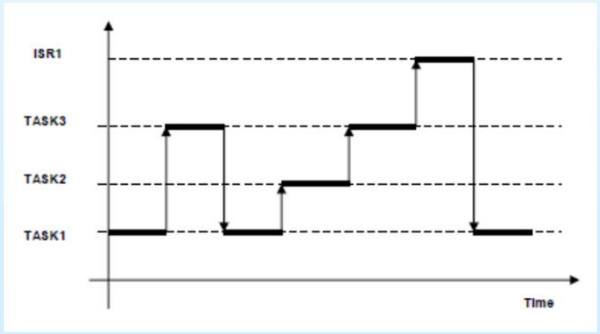
东南大学
马旭东
xdma@seu.edu.cn

2019年11月22日杭州

报告提纲

- 机器人工程专业与软件基础
- 课程组织与RTOS基础理论
- IA32裸机多任务管理实验
- 虚拟Linux OS与任务调度
- SylixOS 系统定制与应用开发
- 课程综合与成绩评估





➔围绕现代RTOS技术展开教学与编程训练
为后续课程提供软件开发和实时多任务运行平台概念

1.机器人工程专业与软件基础—背景故事

➔东南大学2014年两件事：智能机器人专业方向和新专业申报。

以经典控制理论为基础的传统自动化(Basic Automation)已经不再具有挑战性，自动化工程师的主要任务已经为机械、电气、能源、化工等行业工程师所替代，可靠便捷的PLC和数字仪表等技术日益成熟…

初心：自动化的窘境--缺少行业依托，缺少（软硬件）实体化

➔缺少对未来的挑战。立足点：开源技术的发展

申报建设：机器人工程（智能机器人）新专业是在办学过程中，“以市场需求为导向，以学生发展为根本”育人意识的体现，也是机器人工程教学科研团队十多年产学研合作科研和教学成果的产物，适应了国家经济建设和创新人才培养的需求。（事实：从事系统与控制器技术的一群教师2001年开始研究互联网机器人，转而开发工业机器人控制器…..)

2016年3月教育部新增审批国内第一个机器人工程专业----东南大学。由此拉开了国内机器人工程专业的申报建设热潮：2017年备案25所，2018年60所，2019年101所….,新专业建设风起云涌，泡沫四溅（？）背景却是信息化革命的继续…人工智能热潮 vs依托于工业化基础--智能制造工程….

1.机器人工程专业与软件基础—挑战与资源

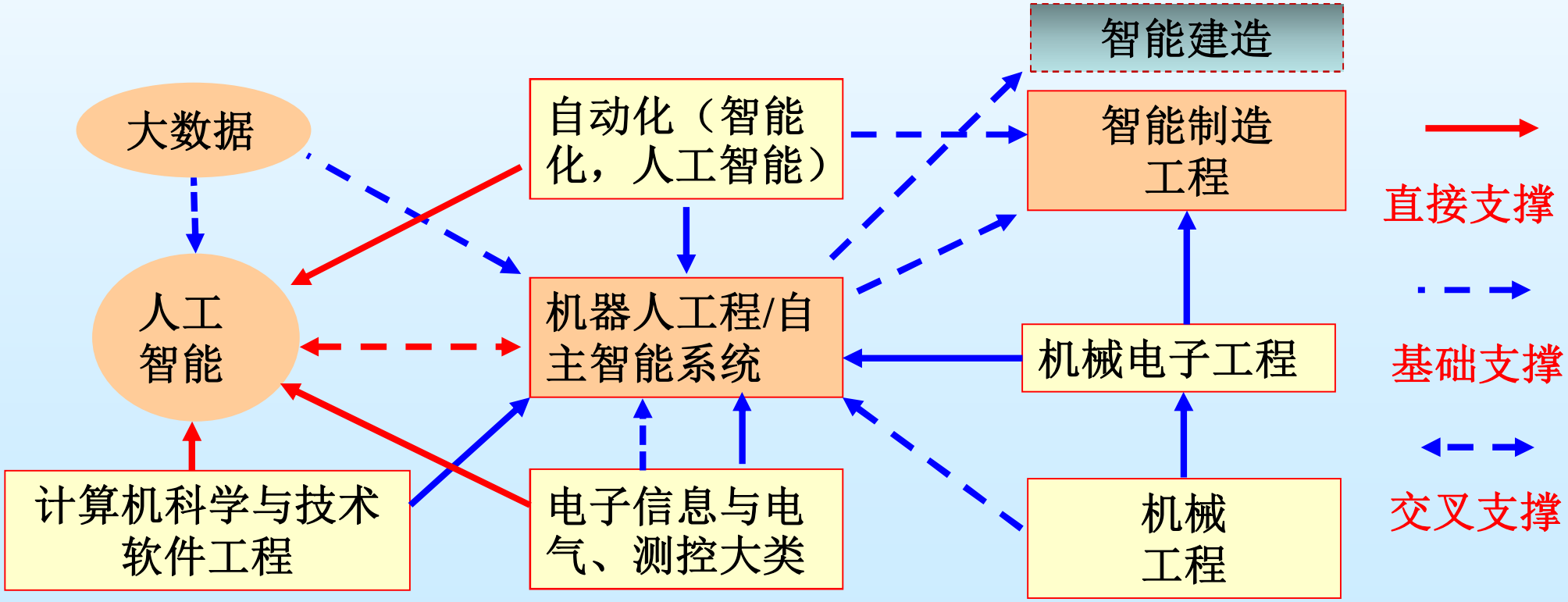
国内：过去一年新的热潮与新专业的挑战 （专业如何合理定位？）

机器人工程 080803T vs.人工智能 080717T (智能科学与技术080907T）

vs.智能制造工程 080213T （机械电子工程080204）

vs.数据科学与大数据技术 080910T

学科交叉典型的新工科专业



1.机器人工程专业与软件基础—ECE转换

自动化-传统机器人→机器人控制器→工业应用（智能制造）

→智能感知、交互与作业 （核心团队）

（控制理论与控制工程）→网络化机器人（系统与控制理论）

→多机器人协作与编队（系统与控制理论）

软件、算法 →自主智能设备（高级应用）

数据结构、系统结构、软件与运行平台新技术

自动化→模式识别与智能系统

→机器视觉

→人工智能

→机器学习

→导航制导与控制

→UAV，AUV，UWV…

→检测技术与自动化装置

→智能交通系统

→智能检测技术

→系统工程→模型、数据….

机器人工程（BRE）

嵌入式计算为基础

(数字化、智能化、多学科新技术)

机械工程

计算机科学与技术

光机电仪器

电子信息与电气

1.机器人工程专业与软件基础—RTOS课程定位

工程、机电、软件基础

sense

think

act

FYS机器智能与机器人

工程设计导论—智能机器人模块

第一学年

认知

数据结构与算法

精密机械设计基础

信息通信网络概论

第二学年

专业基础

嵌入式实时操作系统实践·

ROS软件基础?

数字系统课程设计（SoC）

(1)

第三学年

电机驱动与运动控制

机器人技术基础

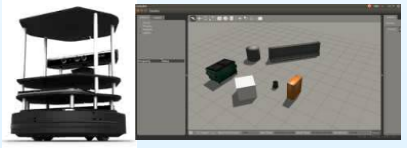
机器人视觉/图像处理

(2)

RTOS+

工业机器人系统

机器人软件工程



(3)

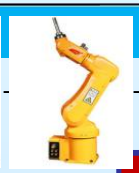
应用提高

计算机控制系统

人工智能

感知与人机交互

智能机器人系统综合课程设计



(1)

第四学年

(2)

机器人动力学与控制

多机器人系统建模与分析

模式识别与机器学习

设计

毕业设计

(3)

竞赛(智能车,RoboCup,电设等)

SRTP

科研实践（实习）

课外实践

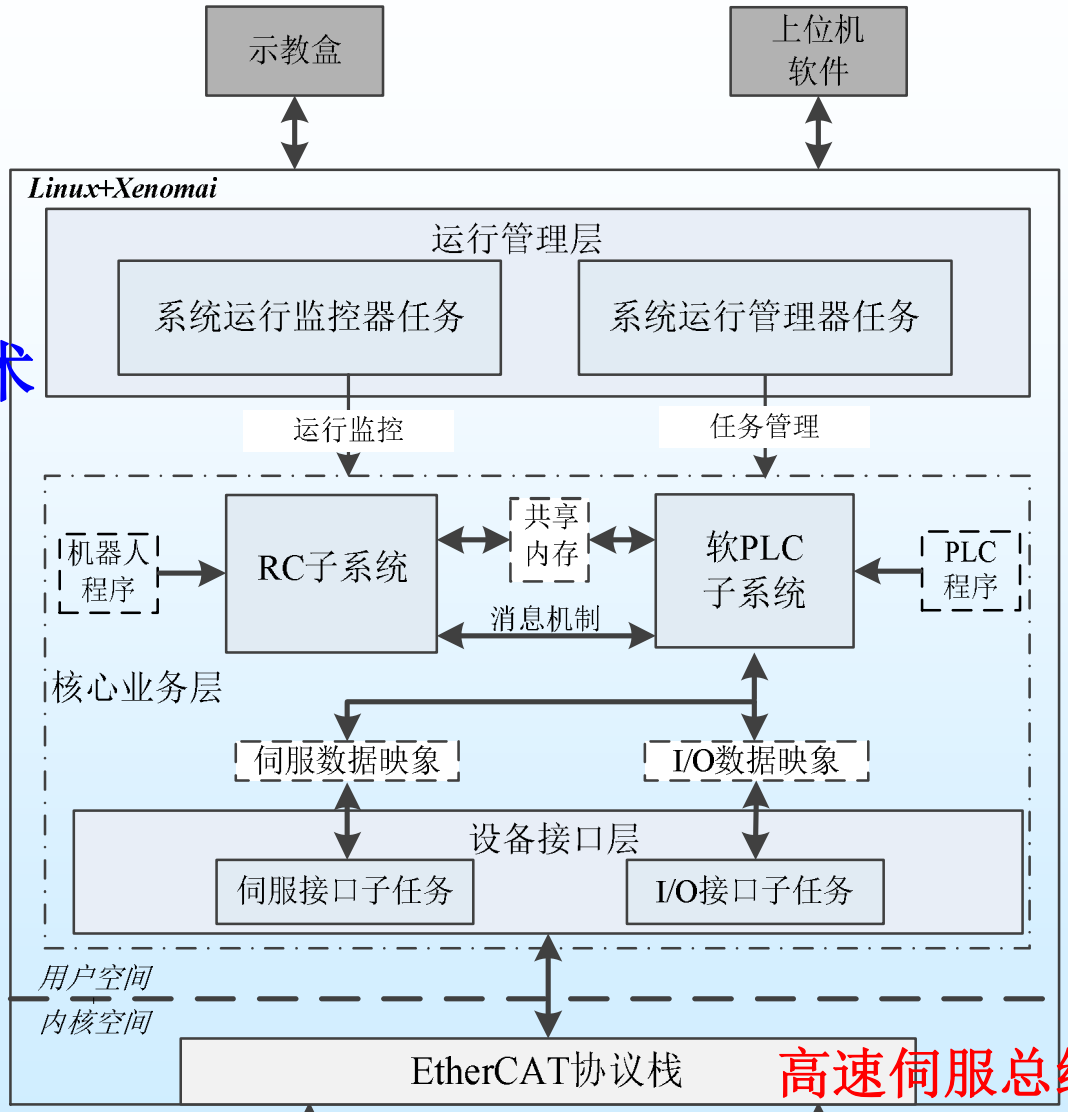
PBL

1.机器人工程专业与软件基础—专业需求

—工业机器人控制器软件—多轴同步、高性能规划计算、精密测量....

RTOS
（VxWorks)
(RT-Linux)

基础核心技术



多核、多处理器系统

Real-time
Multi-tasking

高速伺服总线

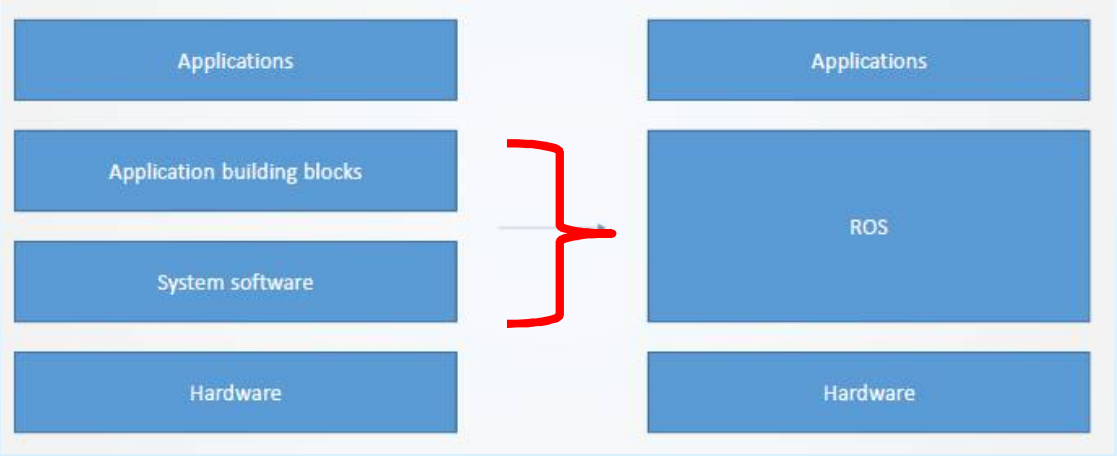
运行管理与控制器

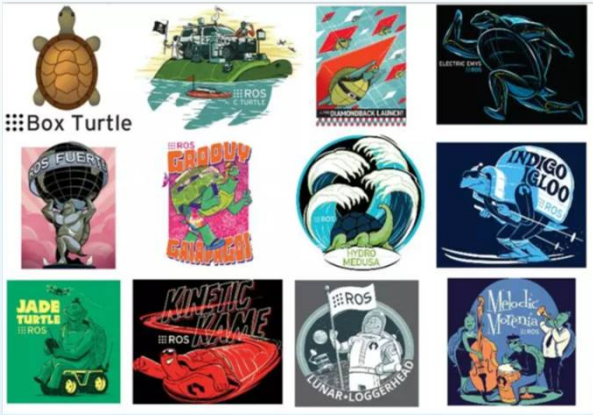
嵌入式系统

1.机器人工程专业与软件基础—机器人应用软件平台

ROS (Robot Operating System, 机器人操作系统) 提供一系列程序库和工具以帮助软件开发者创建机器人应用软件。它提供了硬件抽象、设备驱动、函数库、可视化工具、消息传递和软件包管理等诸多功能。**ROS遵守BSD开源许可协议。**（wiki）

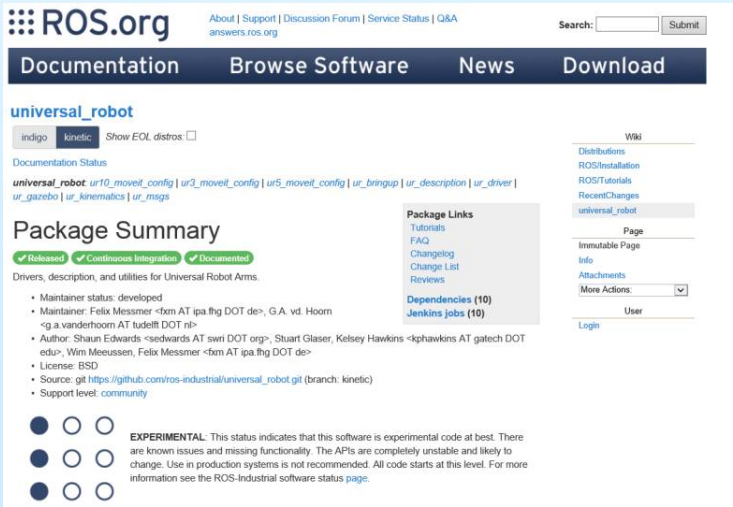
OS平台：Ubuntu (Linux)/MS-Windows





ROS 2.0 设计目标

- 支持涉及不可靠网络的多机器人系统
- 缩小原型机与产品间的差距
- 支持嵌入式微处理器
- 支持实时控制（精细作业） **Real-time 特性**
- 跨系统平台支持



2. 课程组织与RTOS基础理论—概念和元素

OS操作系统： 提供人机接口与控制所有程序执行的（底层）程序➔为程序执行提供环境　（对象：只有硬件和应用编程基础的大二学生）

◆**OS目标**➔执行用户程序并帮助用户易于解决相关问题;方便使用计算机系统;使计算机硬件更高效地工作

◆**OS功能**

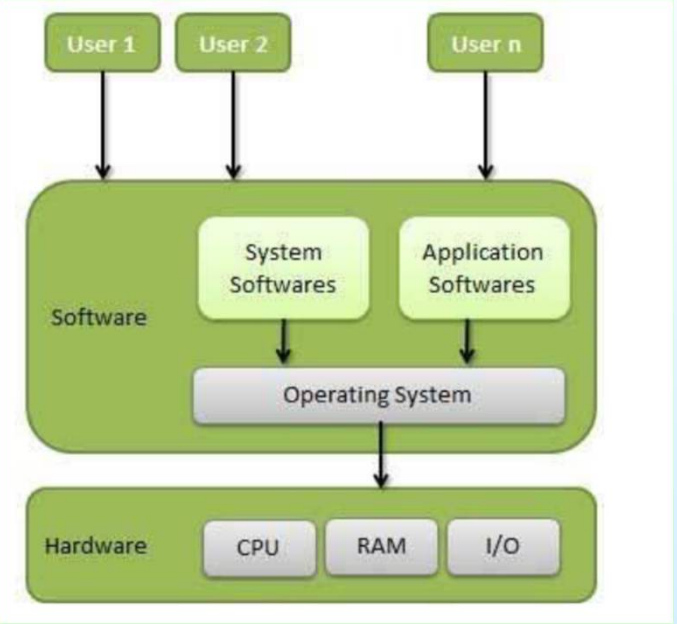
◆**OS本质**

（优势与问题 RTOS/RTS）

进阶学习

- 多任务、多线程、并发过程抽-引入调度
- 硬件层抽象（设备驱动
- 文件系统
- 通信

存储管理
处理器管理
设备管理
文件管理
安全管理
系统性能控制
作业统计
帮助检测错误
软件用户协作

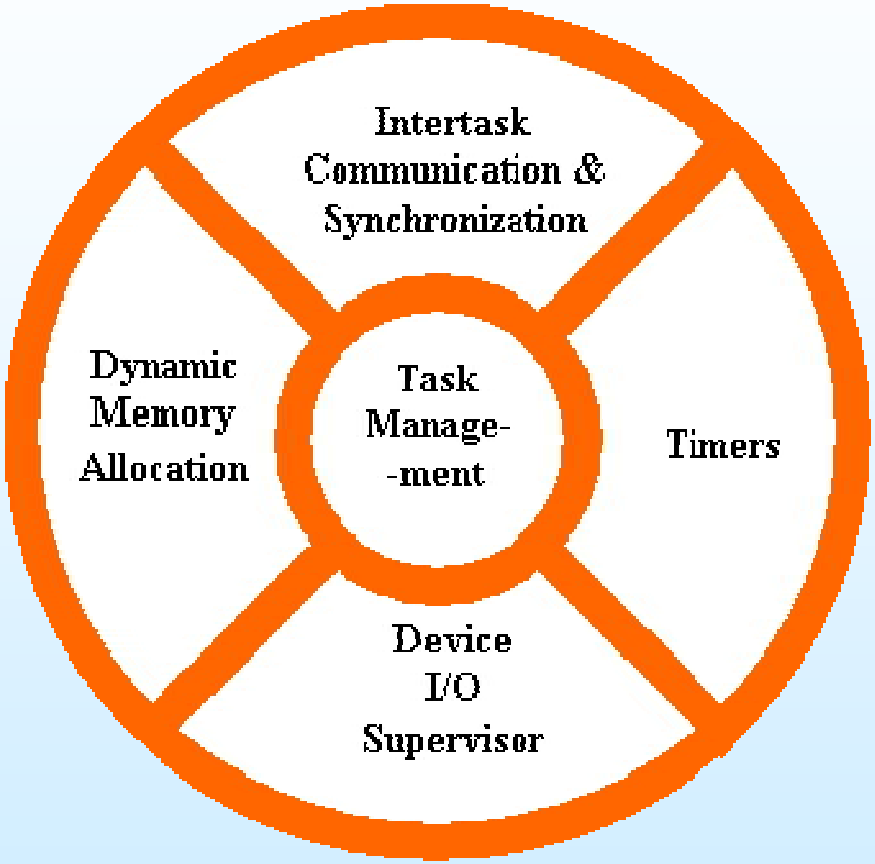


➔重点关心:并发和实时问题（Concurrency, real-time issues）
(协调、安全是RTS的核心，降低不确定性，提高软件质量）

2. 课程组织与RTOS基础理论——概念和元素

- Introduction
- Structure of RTOS
- Components of RTOS
- RTOS Kernel
- Tasks
- Memory
- Timers
- I/O
- IPCs
- Device Drivers
- Expectations
- Examples

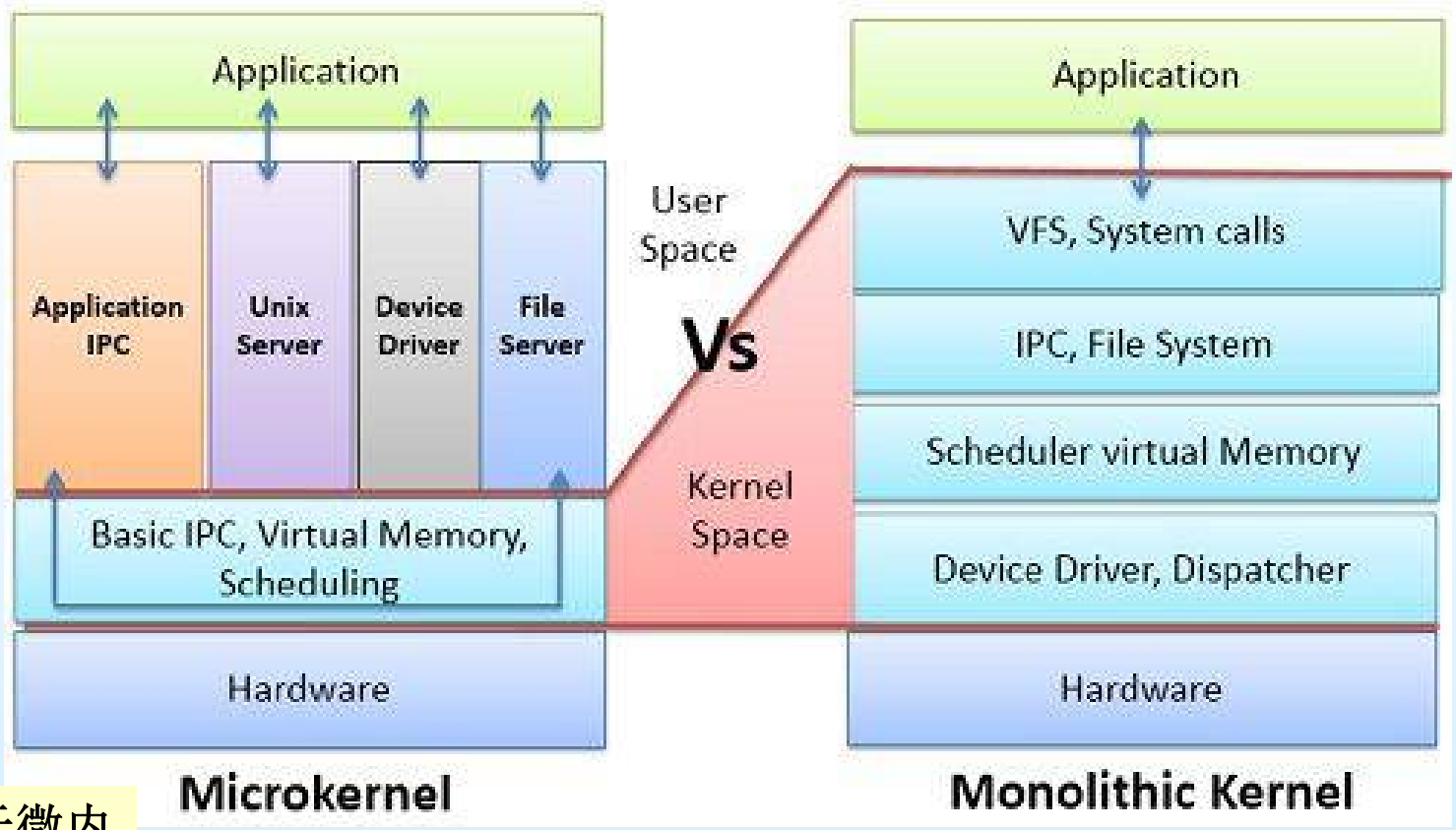
RTOS 内核（KERNEL）组成



➔微内核vs.完整（宏）内核（包括各类文件、驱动、通信协议栈等）--简单理解

2. 课程组织与RTOS基础理论—概念和元素

- Introduction
- Structure of RTOS
- Components of RTOS
- RTOS Kernel
- Tasks
- Memory
- Timers
- I/O
- IPCs
- Device Drivers
- Expectations
- Examples



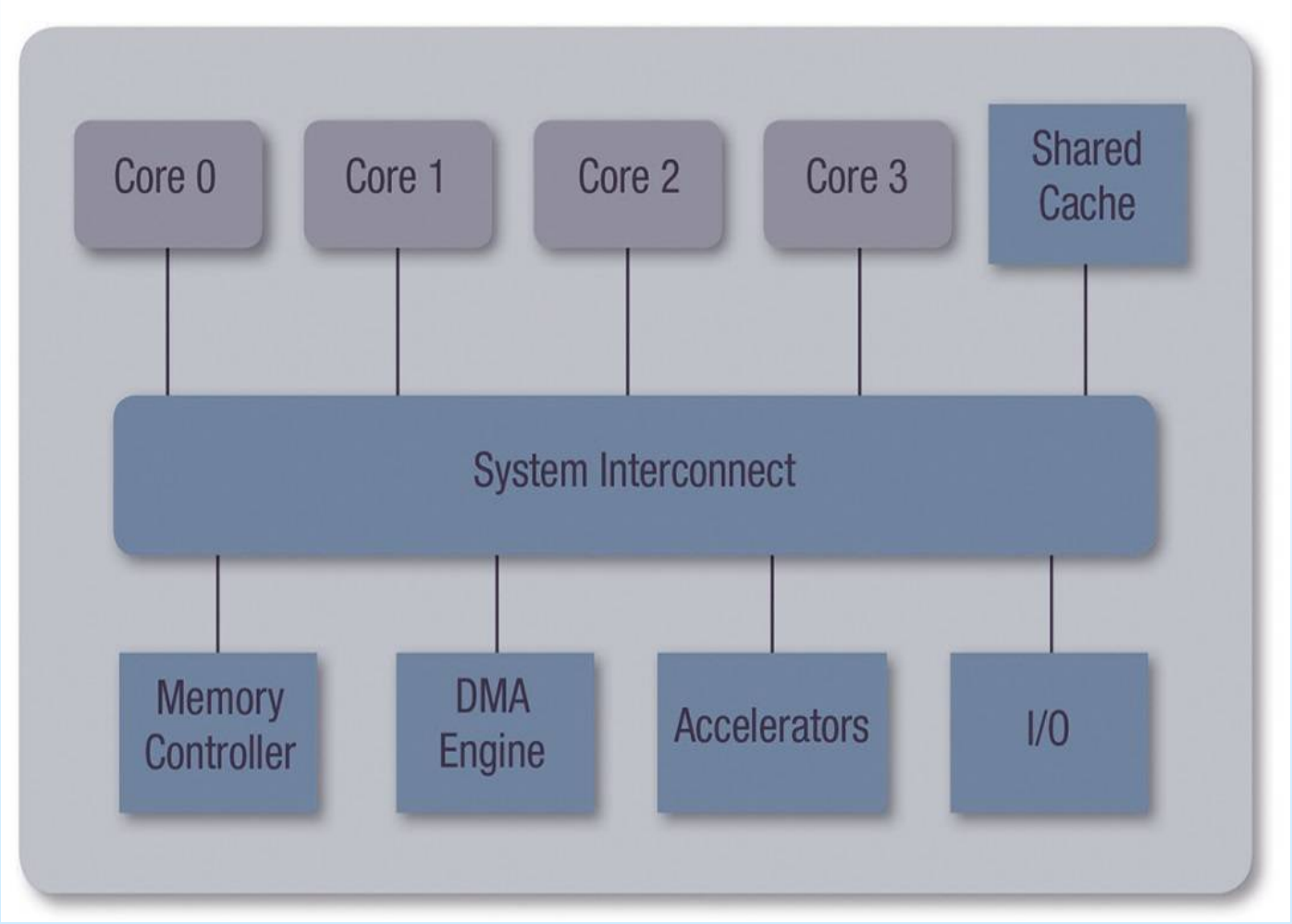
鸿蒙系统（HarmonyOS）:基于微内核的全场景分布式OS，可按需扩展，实现更广泛的系统安全，主要用于物联网，特点是低时延，甚至可到毫秒级乃至亚毫秒级。 三层架构:内核，基础服务，程序框架 (思政)

微内核vs.完整（宏）内核

（不同的用户/内核空间）

2. 课程组织与RTOS基础理论—目标：多核系统

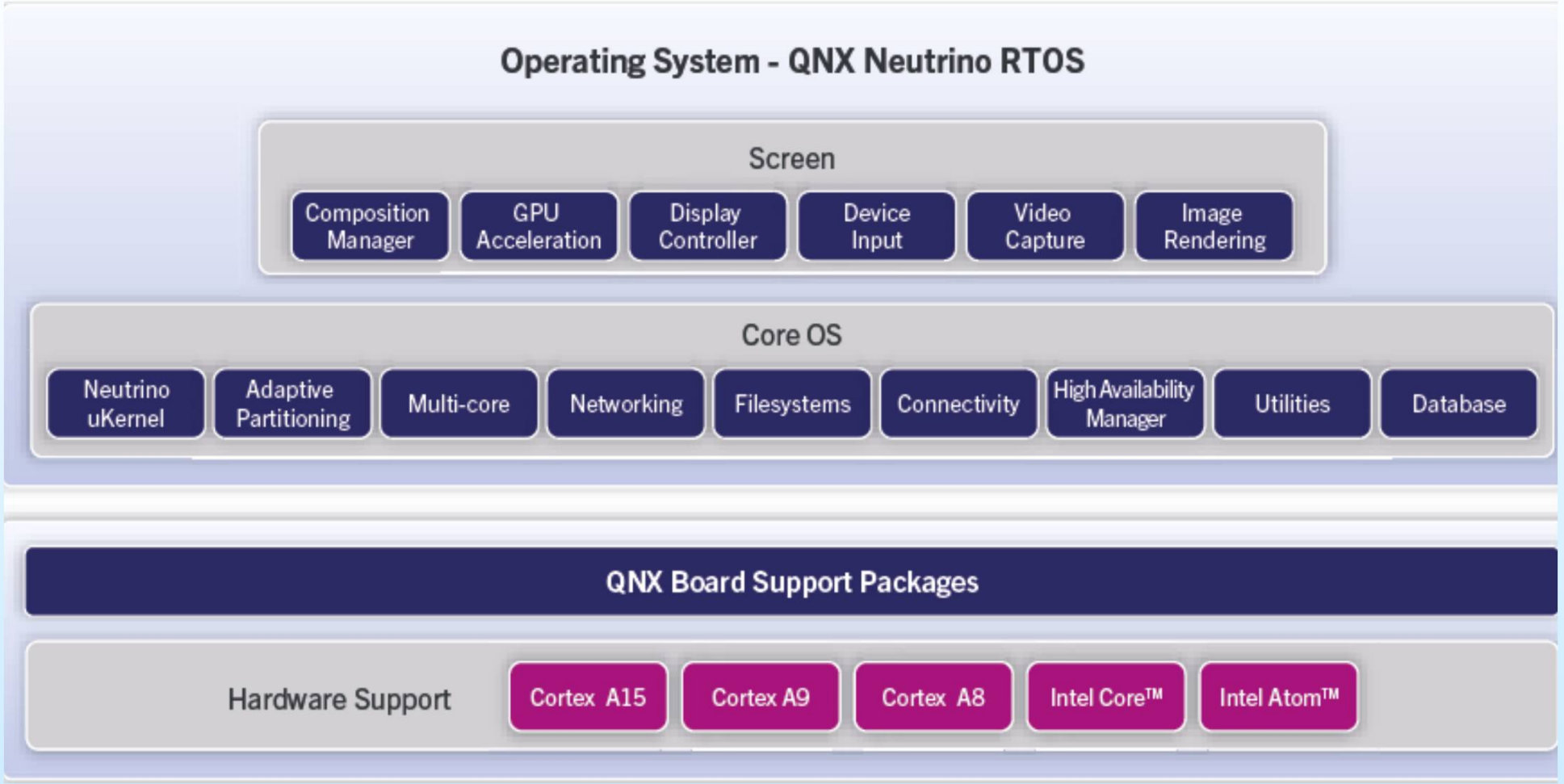
MultiCore Processor与管理—硬件结构



➔面临着计算机结构与组织-微机系统与接口(X86)课程教学改革压力

2. 课程组织与RTOS基础理论—目标：多核系统

MultiCore Processor与管理—架构(QNX)



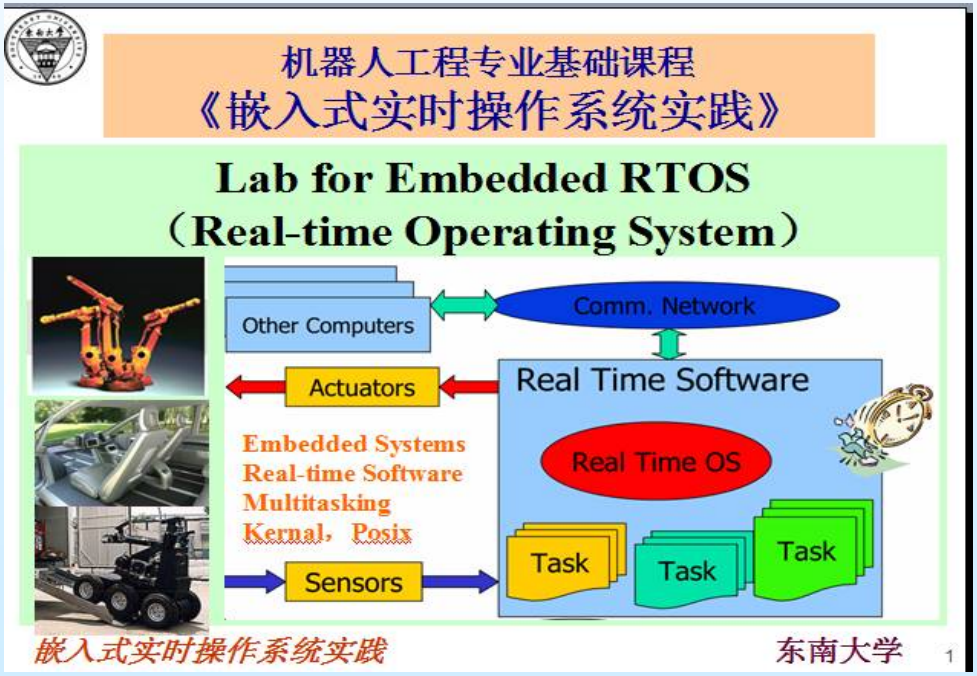
→多核处理器三种运行模式AMP，SMP，BMP(Bound Multiprocessing)

2. 课程组织与RTOS基础理论—思路与架构（效率）

经典课程组织—软件，处理器，实验平台—效率、成本….

➔VxWorks, μCOS II,Free-RTOS（RT-Linux）

本课程➔低成本，基于开源技术的实时操作系统技术（Linux RT扩展： Xenomai；SylixOS（多平台支持 SMP 调度的国产RTOS）



先修课程：程序设计，微机系统与接口➔拓展

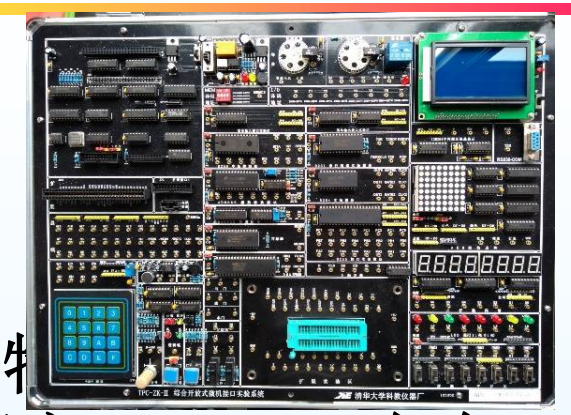
主流性 ➔ 开源化

- RTOS基础理论
- IA32多任务软件编程
- Vmlinux-RTOS内核编程(RMS/优先级调度）（X86-双核）
- SylixOS构建与应用编程（Base，IPC）四核
- 基于RTOS-EtheCAT的PMSM电机伺服控制演示系统 （QT-GUI）CortexA9/X86

2. 课程组织与RTOS基础理论—教材与参考书

- 刘旭明 嵌入式实时操作系统原理与最佳实践，机械工业出版社，2014　　（STM32）
- （美）William Stallings，陈向群等译，操作系统-精髓与设计原理（第8版），电子工业出版社2017 （经典教材）
- (日）川合秀实，30天自制操作系统；人民邮电出版社，2012.9
- Xiaocong Fan，实时嵌入式系统设计原则与工程实践，清华大学出版社，2017.1　（《实时系统与控制》参考书）
- https://code-time.com/default.html
- https://www.freertos.org/Documentation/RTOS_book.html
- 《嵌入式实时操作系统实践》实验指导书（v0.9）自编

3.IA32裸机多任务管理实验—硬件基础编程



1. 了解掌握IA-32微处理器汇编语言程序指令汇编、连接、调试与运行基本概念；

2.掌握汇编程序nasm使用方法；

3.掌握IA-32运行调度程序基本框架，理解内核（特权）调度程序（标号sched到again之间程序）对应用程序（PL=3）任务（TaskA和TaskB(TaskC…)）的调度管理（策略：RR/RMS）；理解基于硬件（复杂开销）的任务切换实现方法；

4.掌握IA-32环境下基于定时中断（=基于时间，RMS）的任务调度、切换与通信基本原理，根据运行情况，进行修改分析（1）时间片长度的影响；（2）构思任务状态指示器（建议：利用逻辑或DAC输出-示波器观察）；3）中断如何结合进入分时调度内核。

5. 集中撰写实验报告，整理总结上述实验内容，形成基于硬件机制的IA32多任务调度机制和程序架构及开发过程

3.IA32裸机多任务管理实验—硬件基础编程

Ring3:应用程序
(可调用ring0任务)

调度切换控制 跳转指向任务TSS或指令描述符
自动切换上下文环境参数,硬件保护
→JMP, CALL, INT n,
选择子:偏移地址

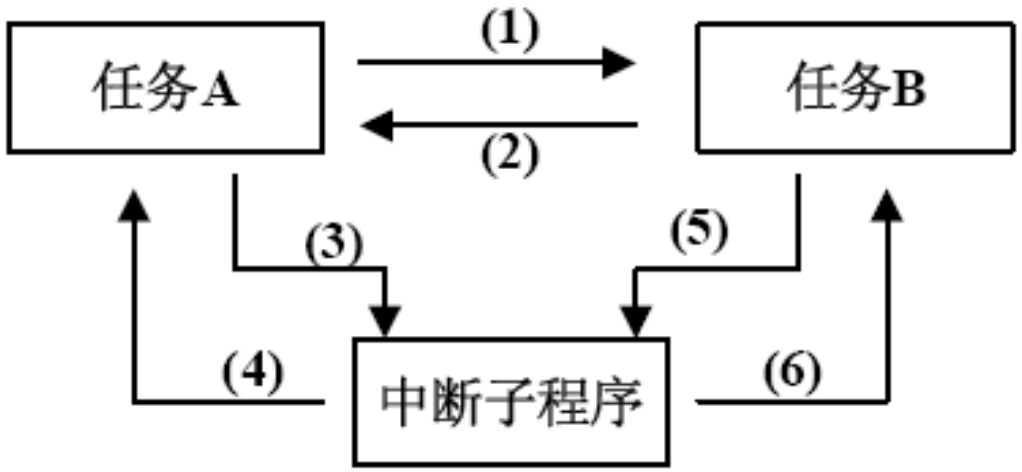
Ring 0（内核））

TaskA
LED 点阵从右往左扫描

TaskB
七段数码管动态显示00-99

中断任务:按键中断一次,执行流水灯任务;字符输出任务(不合理,但显示可行)

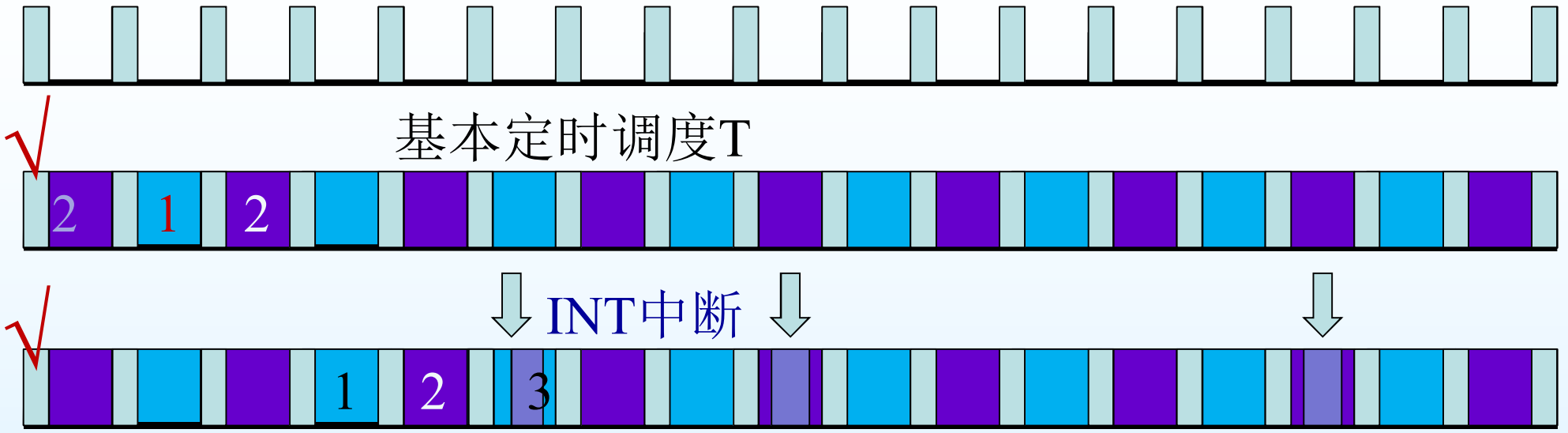
系统8253: 0.7uS-54.6msINT 08任务:按时间切换调度Tasks



IA-32多任务调度切换模型

任务A，B可逻辑切换→增加任务C,D,E,....如何RR调度，优先级调度?—OS作用与规范

3.IA32裸机多任务管理实验—硬件基础编程



IA32定时/优先调度实验模型

T= 1 ms（PC INT 0:默认55.4ms，需要修改8253初值）

T0 ：1 ms一次，切换任务，例行任务

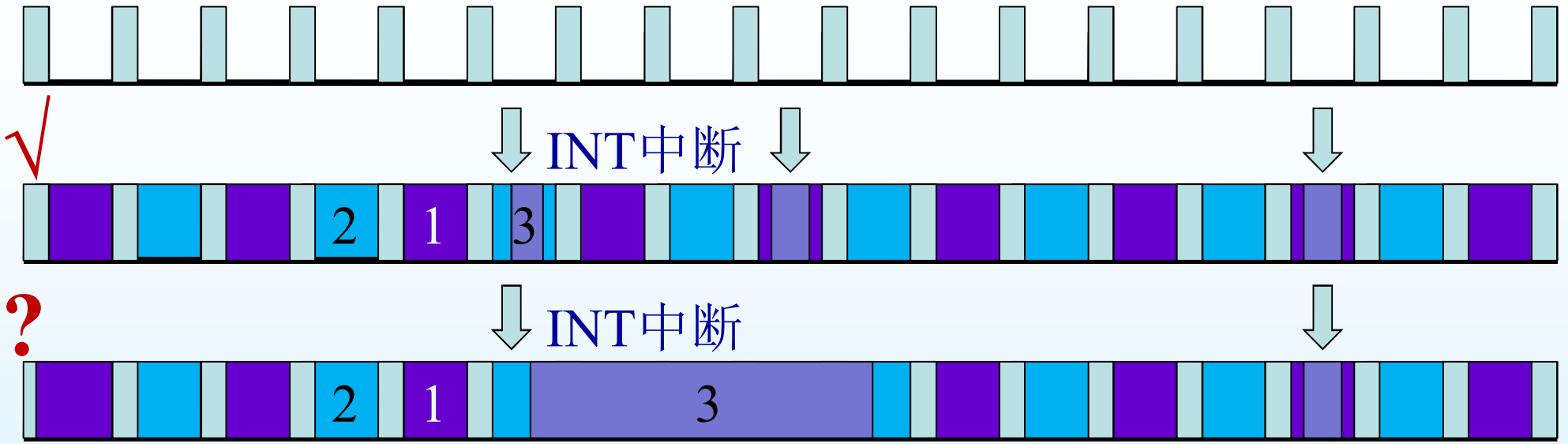
实际：按任务队列表（优先）调度，硬件TSS保护

T1(A) :连续执行，点阵列移动（时间控制）

T2(B) :连续执行，显示T1移动次数00-99

T3(C) :定时或外部中断触发，5 s /随机

3.IA32裸机多任务管理实验—硬件基础编程



IA32定时/优先调度实验模型

T= 1 ms（PC INT 0:默认55.4ms，需要修改8253初值）

T0 ：1 ms一次，切换任务，例行任务

实际：按任务队列表（优先）调度，硬件TSS保护

T1(A) ：连续执行，点阵列移动（时间控制）

T2(B) ：连续执行，显示T 1 移动次数 0 0 — 9 9

T3(C) ：定时或外部中断触发， 5 s / 随机

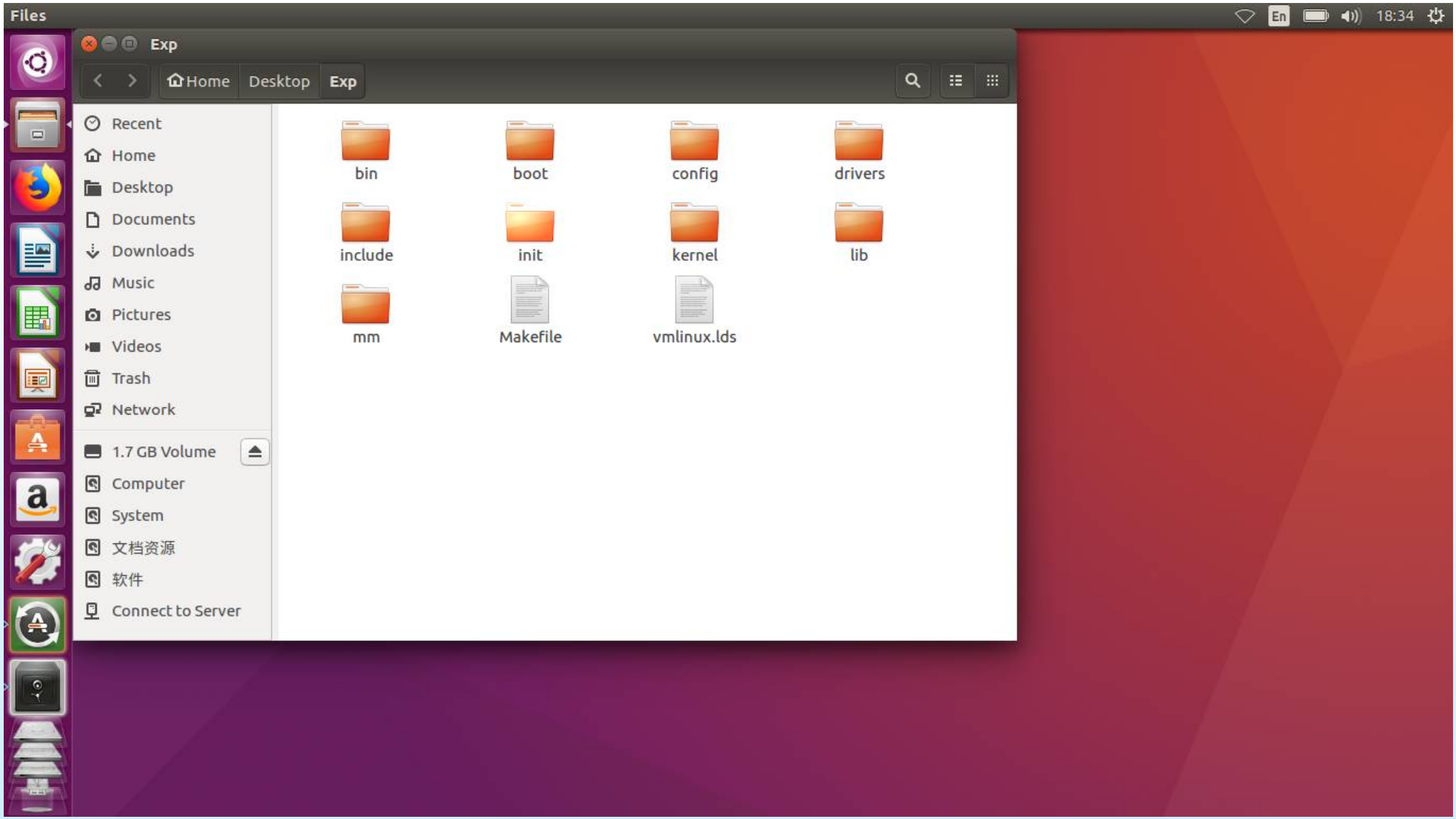
4. 虚拟Linux OS与任务调度—实验2：内核编程环境

1、开发环境配置与源程序分析

➔构建**Ubuntu(linux)**操作系统（盘或虚拟机，此任务为实验前准备工作），认识基本操作，打开微内核（存于目录**EXP**中），分析工程/项目模块结构，分析各模块源程序内容，从代码实现角度认识理解**RTOS**；➔构思编写完整的原型**RTOS （X86）**

实验软件平台：Linux，采用32位ubuntu14.04系统进行开发，使用Windows的同学可以在系统上安装VMware、boch等虚拟机。
编程开发语言：采用C语言和汇编语言编程，以C语言为主。
编程开发工具：C语言和汇编语言编译器使用gcc，链接器使用ld。构建工具为GNU make（均为Ubuntu系统中自带）
运行环境：开发的RTOS微内核程序运行使用qemu虚拟机（基于X86处理器）--体量小，安装方便，在ubuntu的shell中输入以下命令完成qemu的安装：$ sudo apt-get install qemu 或
$ sudo ln -s /usr/bin/qemu-system-i386 /usr/bin/qemu

4. 虚拟Linux OS与任务调度—实验2：内核编程环境



GUI下文件目录EXP ➔显示子目录文件

4. 虚拟Linux OS与任务调度—实验2：内核编程资源

经典vwlinux程序 (开源早期版本）

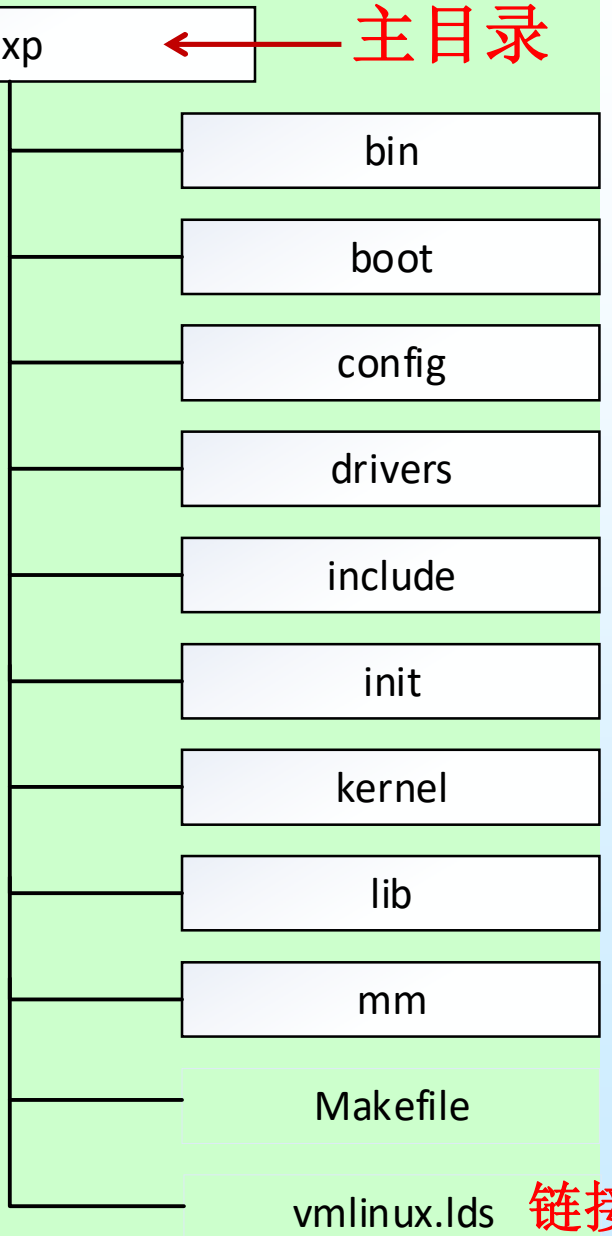
基于时间片的分时操作系统微内核

➔认识基本操作，打开微内核（存于目录EXP中），分析工程/项目模块结构，分析各模块源程序内容，从代码实现角度认识理解RTOS；

优先级：非强占，优先的任务得到更多的执行机会

框架程序项目(project)组成结构

➔Makefile文件是编译辅助工具软件make的参数配置文件(可自动选择编译链接）



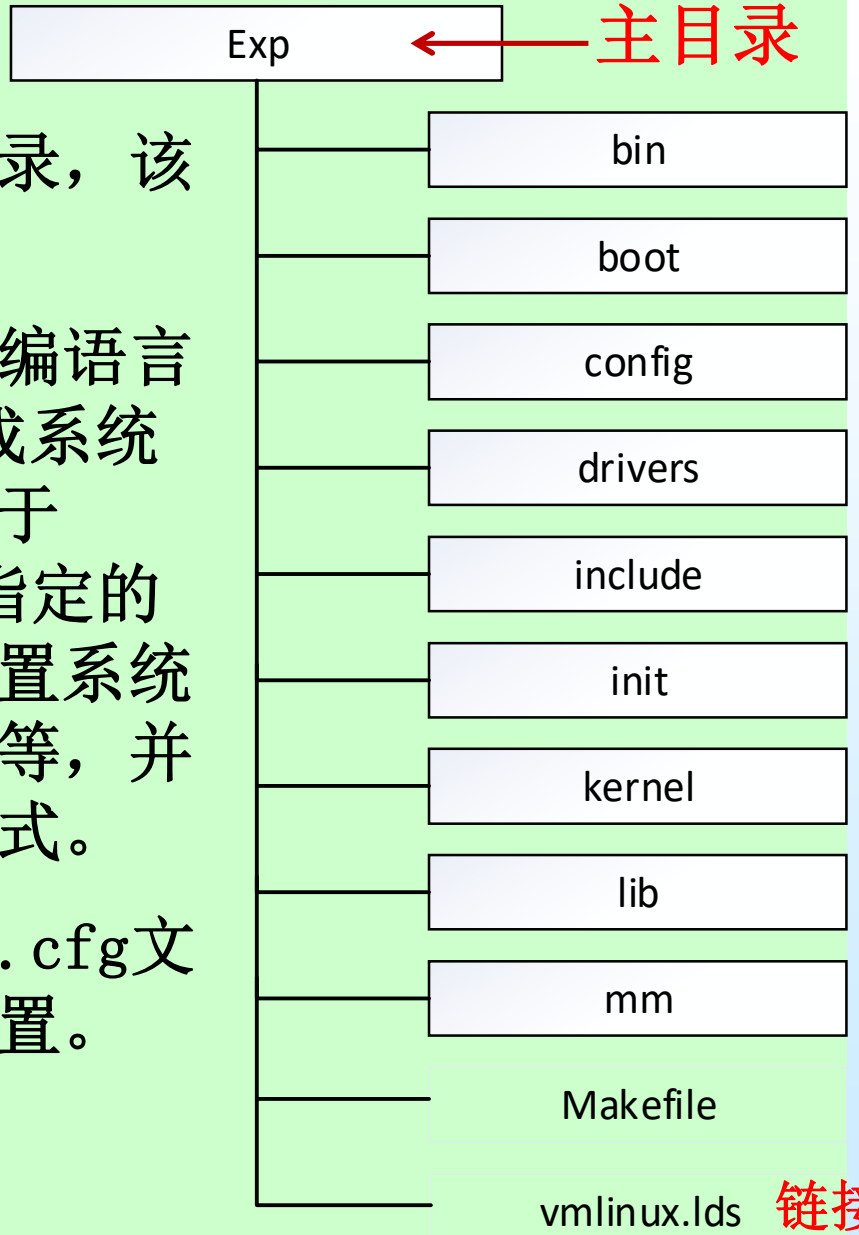
4. 虚拟Linux OS与任务调度—实验2：内核编程资源

分时操作系统微内核程序

bin-辅助工具目录：辅助工具的配置目录，该目录包含了制作.iso文件的必要工具。

Boot--引导启动目录：包含一个AT&T汇编语言文件和一个C语言文件，该目录主要完成系统的启动引导任务。具体实现的任务是基于Grub2，采用multiboot2标准，将一些指定的硬件信息读入内存的指定位置，同时设置系统工作所需的临时GDT，IDT，堆栈，页表等，并使X86计算机从实模式进入32bit保护模式。

config--grub配置目录：包含一个grub.cfg文件，该文件是对grub开机引导目录的设置。



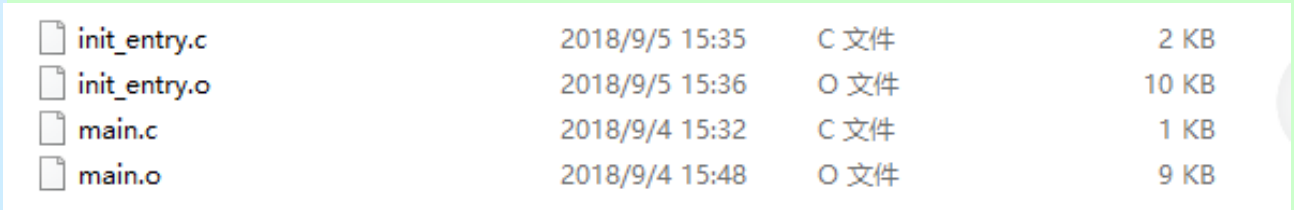
4. 虚拟Linux OS与任务调度—实验2：内核编程资源

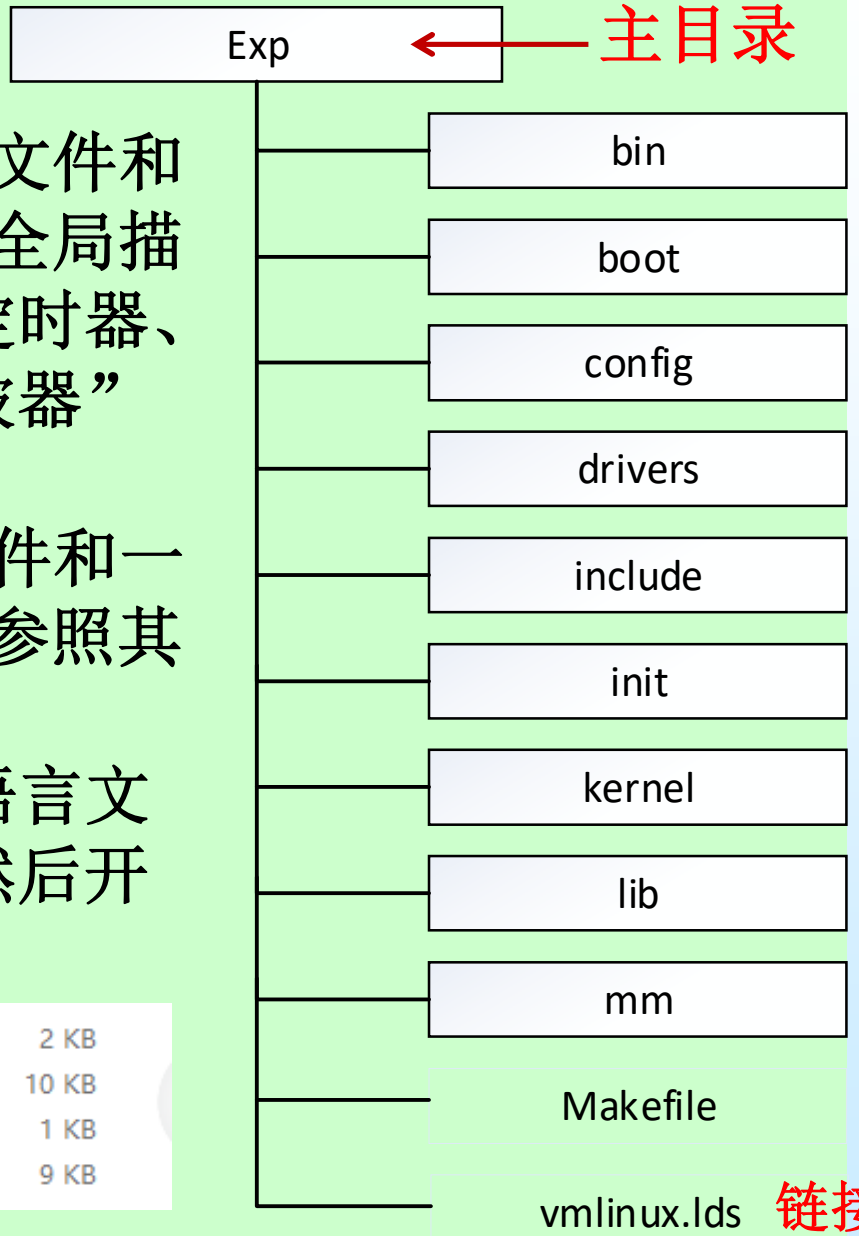
分时操作系统微内核程序

drivers--驱动程序目录：包含8个C语言文件和2个AT&T汇编语言文件，该目录主要是对全局描述符表GDT，中断描述符表IDT、键盘、定时器、显卡的图像模式以及任务切换显示“示波器”等的初始化设置。

Include--头文件主目录：包含20个.h文件和一个sys文件夹，各头文件的作用及功能请参照其他各目录的作用与描述。

Init--内核初始化模块目录：包含2个C语言文件，用于执行内核所有的初始化工作，然后开始设定实验任务的执行与切换。



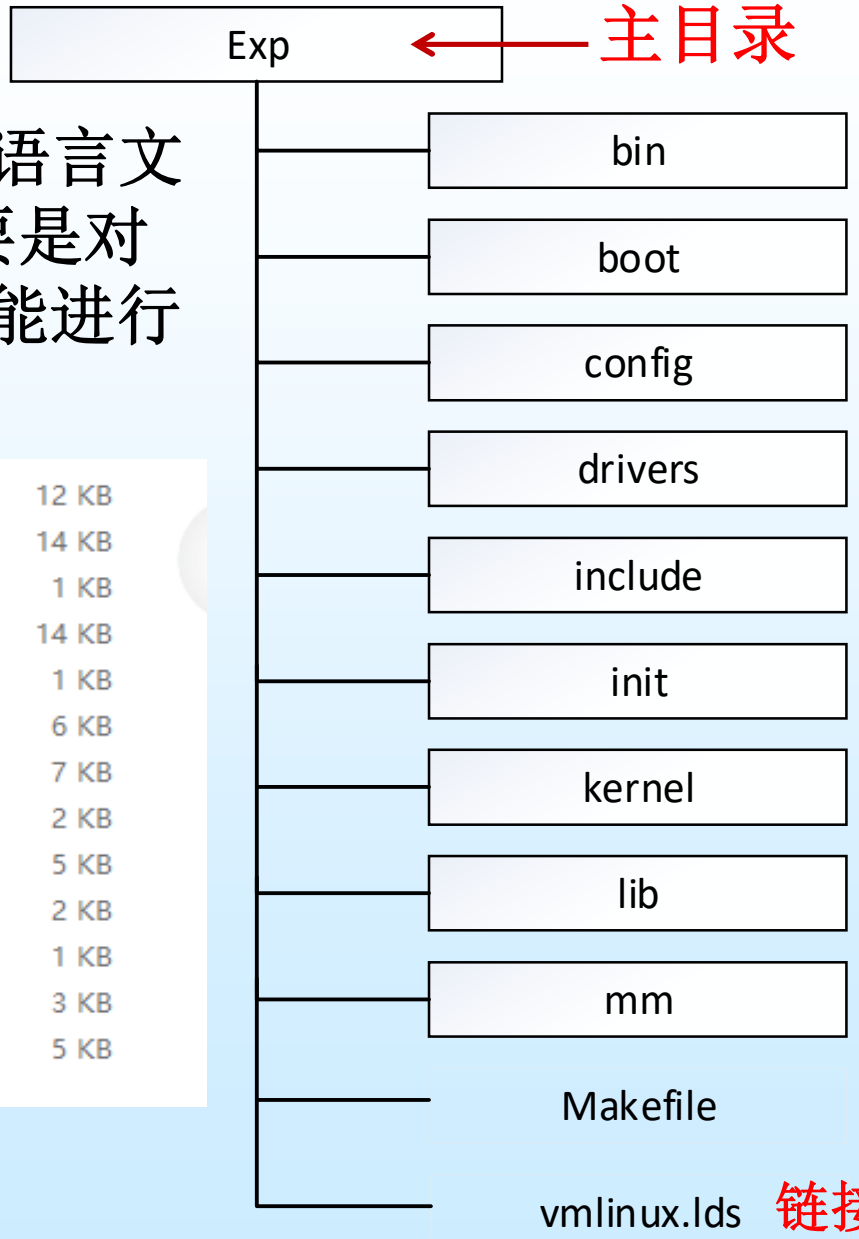


4. 虚拟Linux OS与任务调度—实验2：内核编程资源

分时操作系统微内核程序

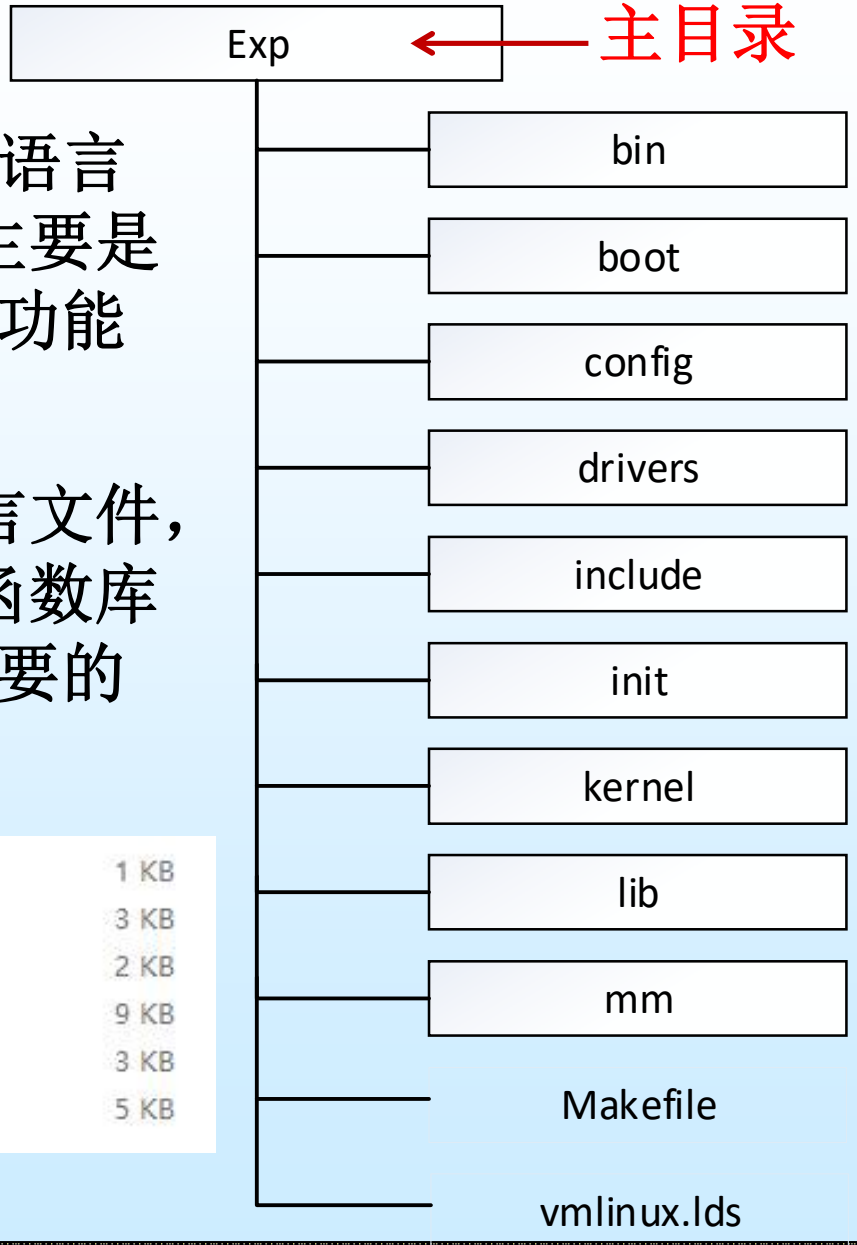
Kernel--内核程序目录：共包含了3个C语言文件和3个AT&T汇编语言文件，该目录主要是对任务创建、任务调度、字符串打印等功能进行的设置。

.sched.c.swp	2018/9/5 14:54	SWP 文件	12 KB
head_32.o	2018/9/4 15:48	O 文件	14 KB
head_32.S	2015/9/24 23:41	S 文件	1 KB
kernel.o	2018/9/4 15:48	O 文件	14 KB
kernel.S	2018/8/14 22:57	S 文件	1 KB
printk.c	2018/8/29 11:11	C 文件	6 KB
printk.o	2018/9/4 15:48	O 文件	7 KB
sched.c	2018/8/29 9:57	C 文件	2 KB
sched.o	2018/9/4 15:48	O 文件	5 KB
switch_to.o	2018/9/4 15:48	O 文件	2 KB
switch_to.S	2018/8/15 16:56	S 文件	1 KB
task.c	2018/9/4 13:05	C 文件	3 KB
task.o	2018/9/4 15:48	O 文件	5 KB



4. 虚拟Linux OS与任务调度—实验2：内核编程资源

分时操作系统微内核程序



Kernel--内核程序目录：共包含了3个C语言文件和3个AT&T汇编语言文件，该目录主要是对任务创建、任务调度、字符串打印等功能进行的设置。

Lib--内核库函数目录：包含了3个C语言文件，由于操作系统级的程序不能使用标准C函数库以及其它的一些函数库，此处对一些必要的功能函数进行了实现。

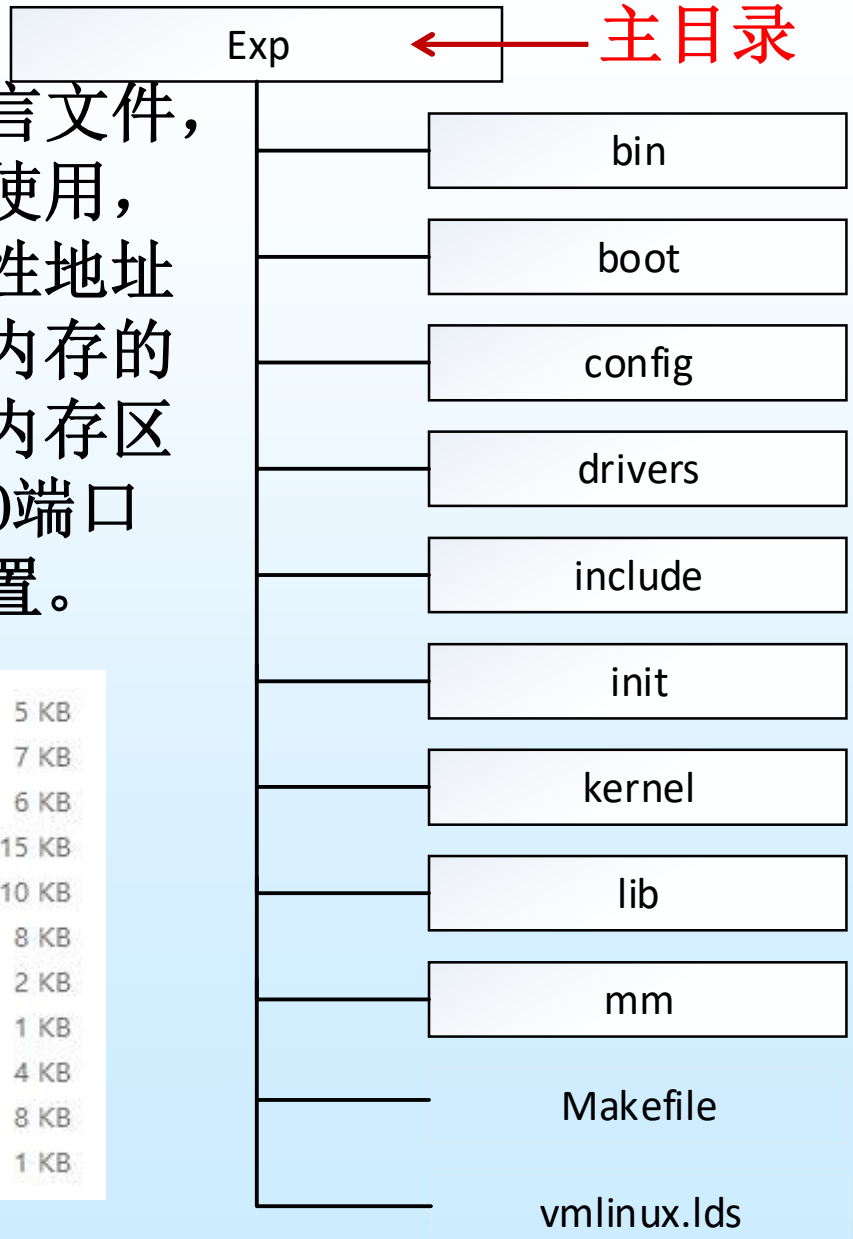
common	2018/8/29 11:11	C 文件	1 KB
common.o	2018/9/4 15:48	O 文件	3 KB
console	2018/8/29 11:12	C 文件	2 KB
console.o	2018/9/4 15:48	O 文件	9 KB
misc	2018/8/29 11:13	C 文件	3 KB
misc.o	2018/9/4 15:48	O 文件	5 KB

4. 虚拟Linux OS与任务调度—实验2：内核编程资源

分时操作系统微内核程序

mm--内存管理程序目录：包含了3个C语言文件，该目录主要用于管理程序对主内存区的使用，实现了进程逻辑地址到线性地址以及线性地址到主内存区物理内存地址的映射，通过内存的分页管理机制，在进程虚拟内存页与主内存区建立了对应关系。同时，它也提供了I/O端口到内存的映射关系和完成了内存堆的设置。

heap	2018/8/13 16:54	C 文件	5 KB
heap.o	2018/9/4 15:48	O 文件	7 KB
iomapandvmm	2018/8/29 11:14	C 文件	6 KB
iomapandvmm.o	2018/9/4 15:48	O 文件	15 KB
ioremap.o	2018/8/16 15:34	O 文件	10 KB
memory.o	2018/8/16 15:34	O 文件	8 KB
page_alloc.o	2018/8/16 15:34	O 文件	2 KB
pm.o	2018/8/13 12:10	O 文件	1 KB
pmm	2018/8/29 11:13	C 文件	4 KB
pmm.o	2018/9/4 15:48	O 文件	8 KB
vm.o	2018/8/13 12:11	O 文件	1 KB



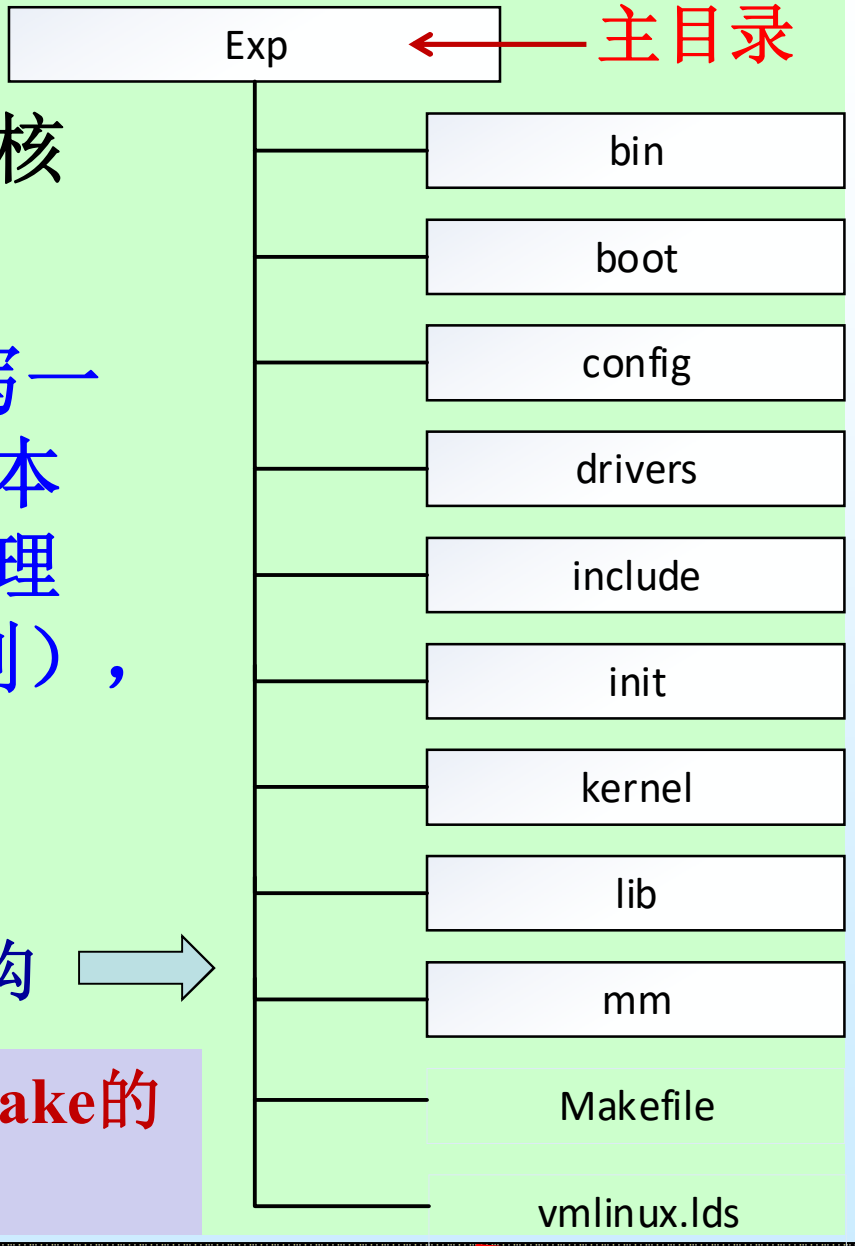
4. 虚拟Linux OS与任务调度—实验2：内核编程实验

经典**vwlinux**程序 (早期版本）

基于时间片的分时操作系统微内核

实验➔基于IA32处理器（虚拟机qemu），用SUBLIME或Vi修改编写一RTOS微内核原型源程序，实现基本的基于时间片的分时任务调度管理（类似于Linux系统调度运行机制），提供最简单的任务协调同步机制

框架程序项目(project)组成结构



➔**Makefile**文件是编译辅助工具软件**make**的参数配置文件(可自动选择编译链接）

4. 虚拟Linux OS与任务调度—实验2：基本编程

2.基于时间片的OS微内核任务的创建与调度（原理）

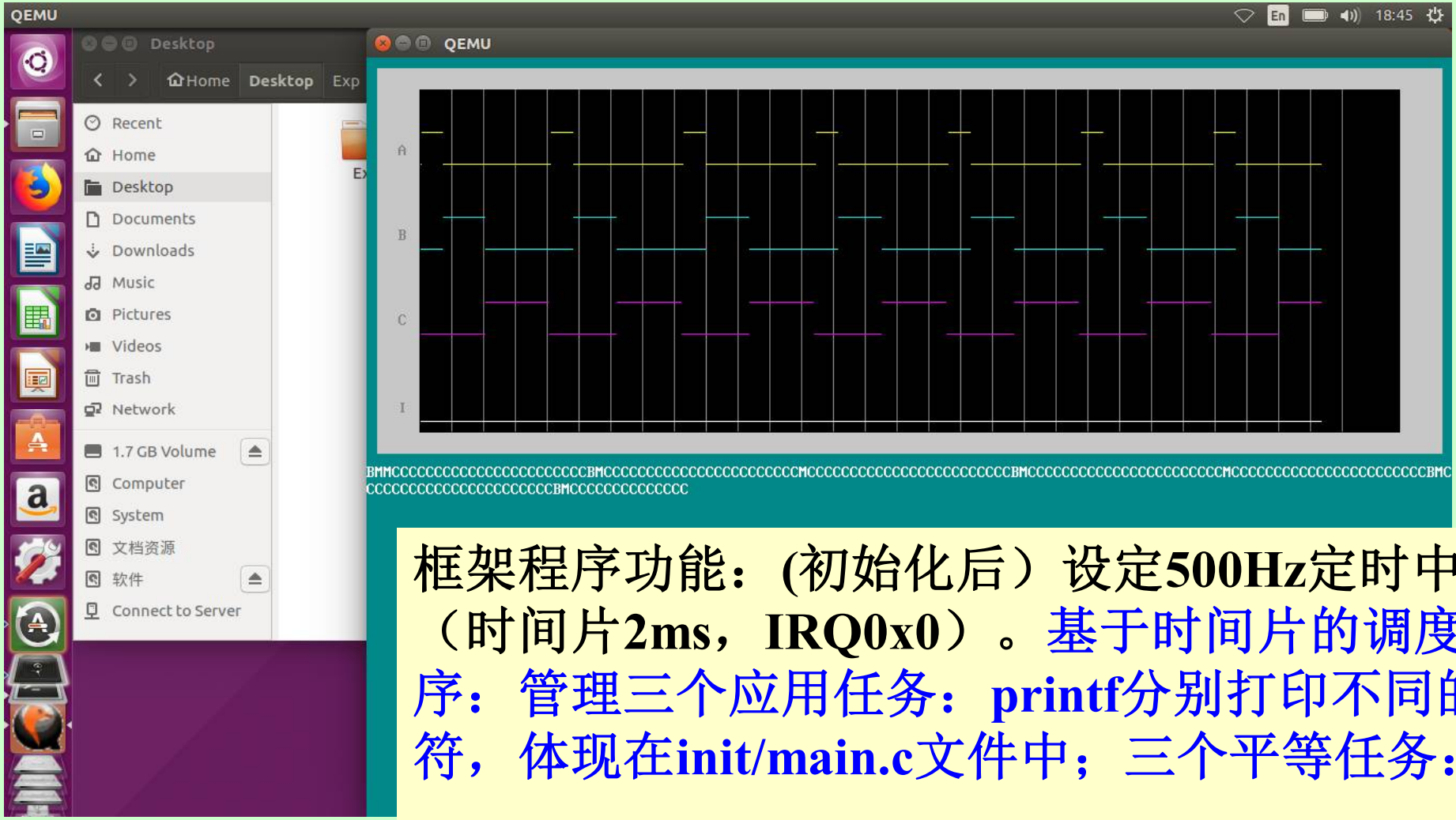
实验框架程序：任务=内核线程。内核线程作为内核态的一个逻辑执行流，拥有私有的栈空间，但是除了私有栈之外，不拥有其他资源，所有的内核线程拥有相同的页表，共享所有的全局数据。

说明：IA32CPU提供的任务切换机制（MS-Windows）使用TSS段进行切换（如实验一）→繁琐复杂且效率低，一般操作系统不完全采用硬件切换机制。本实验所阐述的任务切换只发生在内核态，不涉及特权级转移过程，可以完全由硬件实现。

任务切换TCB/PCB→保存当前任务的现场，恢复下一个要执行的任务的现场，所以需要一个数据结构task_struct来保存这些现场信息。这个数据结构一般会放置任务相关的一些信息并且以链表的形式组织起来，这个结构被称为PCB（Process Control Block）或者TCB（Task Control Block）。

4. 虚拟Linux OS与任务调度—实验2：基本编程

基于时间片调度（程序设计与运行） （time-slot）



框架程序功能：(初始化后）设定500Hz定时中断（时间片2ms，IRQ0x0）。基于时间片的调度程序：管理三个应用任务：printf分别打印不同的字符，体现在init/main.c文件中；三个平等任务：

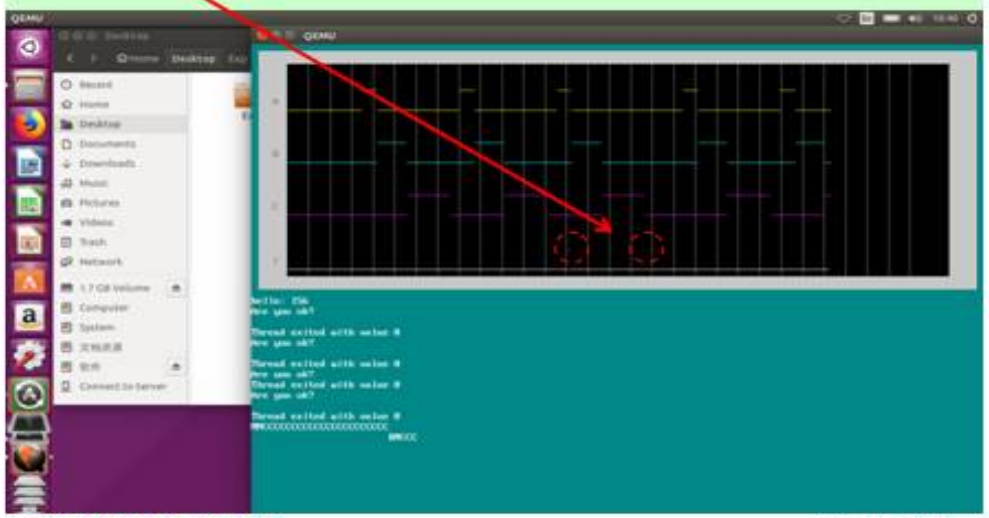
不同优先任务运行（实例A:1，B:2,C:3）

4. 虚拟Linux OS与任务调度—实验2：基本编程

基于时间片调度队列（程序设计与运行） （time-slot）

Vm-Linux调度实验(RTOS内核)

三个任务A/B/C ： 周期20 40 60
键盘中断任务I:



实时系统与控制 东南大学 42

Vm-Linux调度实验(RTOS内核)

```
void schedule()
{       if (current) {
                //printk("i am here%d\n",current->count);
                if(current->count>0)
                        current->count = current->count-1;
                else{
                        current->count = current->priority;
                        change_task_to(current->next);
                }
        scope_draw(current->pid);
        }
}
```

实时系统与控制 东南大学 45

4. 虚拟Linux OS与任务调度—实验2：基本编程

基于时间片调度（程序设计与运行）--任务队列

中断任务：硬件支持键盘中断（IRQ0x01），随机键入，插入队列。

任务状态指示由模拟的图形“示波器”完成，显示为4个通道：从上往下分别是任务A(1)，任务B(2)，任务C(3)和键盘中断的运行状态，高电平为运行状态，低电平为挂起状态。

➔基于时间的任务执行与切换原理

实验任务：

- 重点阅读理解kernel/task.c文件以及调用的相关函数定义，创建和管理多个（5~10个）优先级相同或不同的任务，实验时观察屏幕上方示波器显示的波形，分析原因。
- 重点阅读理解drivers/keyboard.c和drivers/idt.c文件以及调用的相关函数定义，修改键盘中断任务，体会中断响应的实时性。

4. 虚拟Linux OS与任务调度—实验2：基本编程

独立程序流

任务A

任务B

```
int task1(void *arg)
{
        while (1) {

                printk("B");
                wait(100);
        }
        return 0;
}
int task2(void *arg)
{
        while (1)
        {
                printk("M");
                wait(100);
        }
        return 0;
}
```

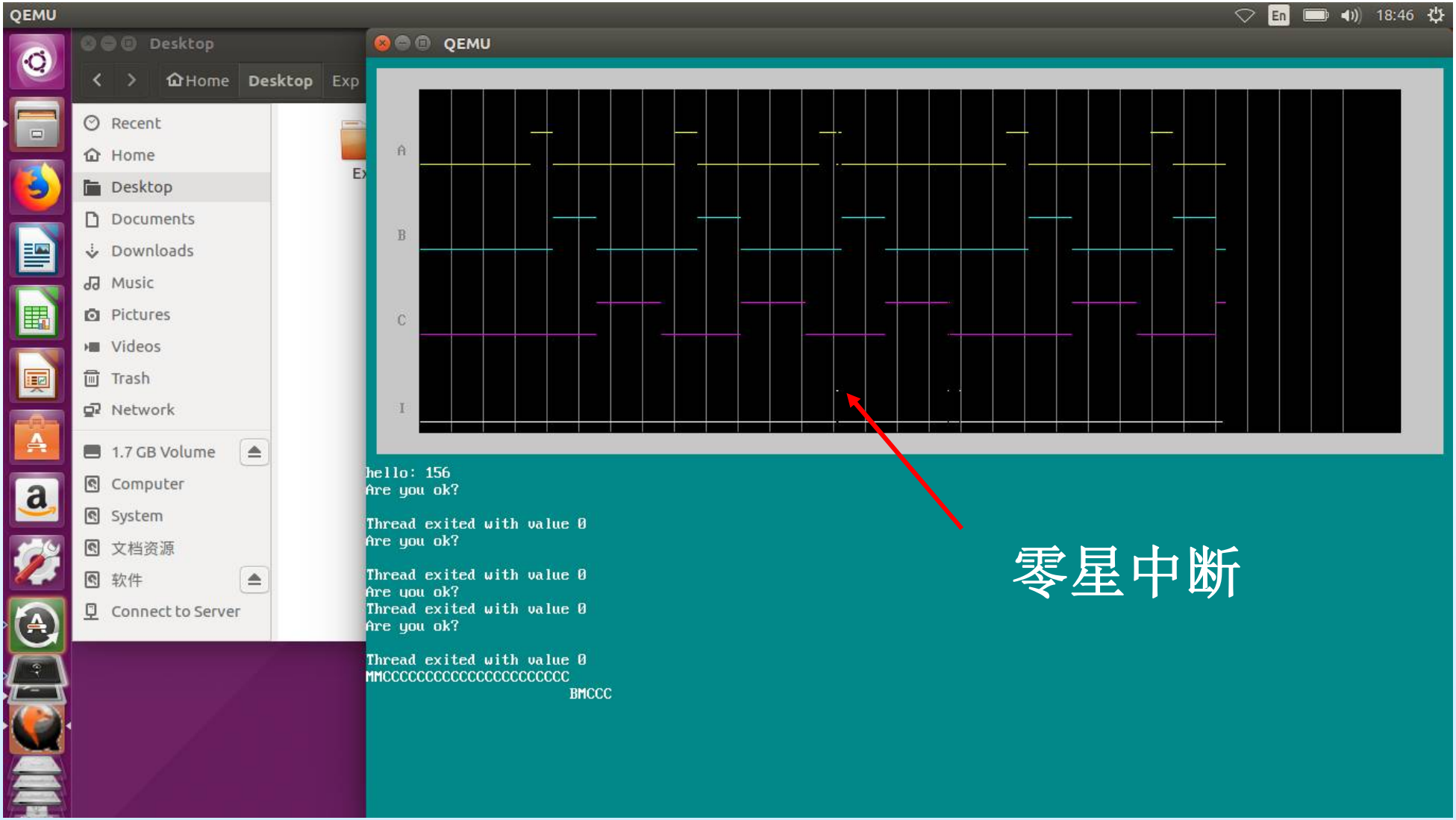
```
int task3(void *arg)
{

        while (1)
        {
                printk("C");
                wait(100);
        }
        return 0;
}
```

任务C

按键中断任务

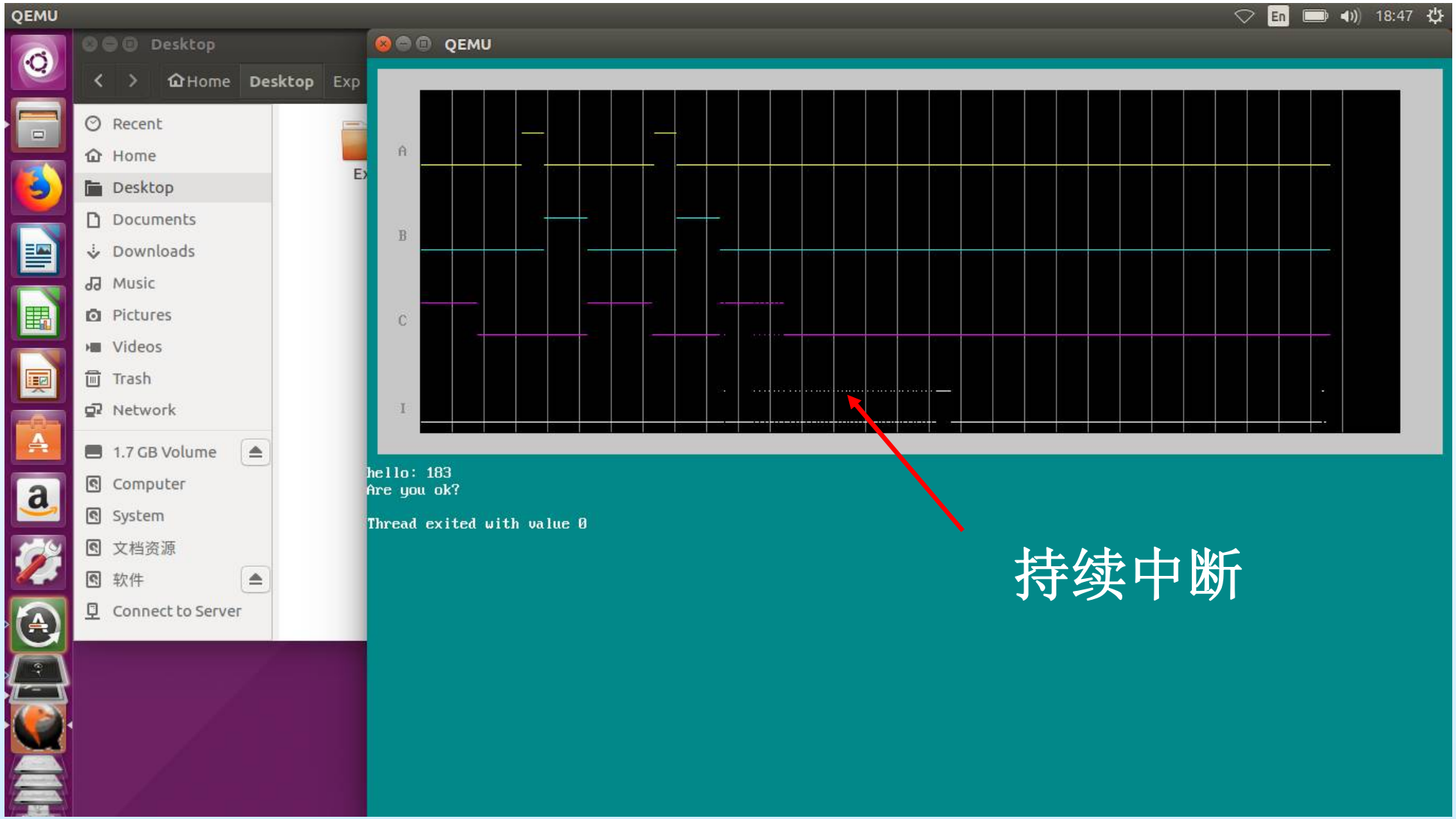
```
int rttask()
{
        printk("\nAre you ok?\n");
        return 0;
}
```

4. 虚拟Linux OS与任务调度—实验2：基本编程



实际显示效果

4. 虚拟Linux OS与任务调度—实验2：基本编程



实际显示效果

4. 虚拟Linux OS与任务调度—实验2：提高编程

3、构思设计：各种RTOS微内核参数与结构修改方案

（具有优先级和实时特性—补丁patch）

基于IA32处理器，分别探索在基于时间片的分时任务调度管理机制之上的优先级（任务获得的执行时间长短）及实时调度机制（Linux+实时补丁）的实现方案，用SUBLIME或Vi修改编写RTOS微内核原型源程序，实现编程（类似于RT-Liunx、Xenomai等系统），观察结果（此为验收检查内容）。

优先：高优先级的任务具有抢占能力或获得更多的执行时间

实时：获得确定的响应延时(概念）

5. SylixOS系统定制与应用开发—系统与应用

实验3内容

1、理解认识商业化RTOS典型功能

2、构建RealEvo-IDE集成开发环境

3、 RealEvo-IDE的编程修改及运行方法；

（面向X86系统：虚拟系统和运行）

4. 定制创建（指定参数、裁剪构造）1-3例典型基于X86的目标RTOS基本（Base）系统，编译链接，在PC本机上虚拟运行，输入系统提供的相关操作命令，记录现象，分析结果，形成X86/PC环境下操作系统开发过程的基本概念；

5. 分析理解SylixOS操作系统源程序各组成模块原理和功能，设计修改其中某段程序内容，编译链接，在PC本机上虚拟运行，记录现象，分析结果是否与设计相符，进一步加深对RTOS技术和编写开发RTOS的认识理解。

5. SylixOS 系统定制与应用开发—系统与应用

SylixOS（国产：北京翼辉信息）是支持 **SMP** 调度的原创大型硬实时操作系统 ,其诞生可以摆脱国内一些关键性设备对国外嵌 入式操作系统的依赖，为国内的嵌入式信息技术行业提供一个全新的选择。**SylixOS** 目前是以公开源代码项目的形式存在。

1. 内核自主化率达到 **100%**（依据工信部评估报告）， 拥有完全自主可控的技术能力，满足国产化需求；

2. 开源**OS**，可靠性、安全性更容易验证；

3. 支持对称多处理器（**SMP**）平台，并且具有实时进程及动态加载机制，满足多部门分布式软件开发需求，支持各部门应用软件在操作系统上的集成；

4.处理器跨平台支持 支持 ARM、MIPS、PowerPC、x86、SPARC、DSP、RISC-V、C-SKY 等架构处理器，支持主流国产通用处理器，如飞腾全系列、龙芯全系列、中天微**CK810** 、兆芯全系列等，便于用户在升级硬件平台的时候，进行应用程序的移植，减少移植的工作量；

5. SylixOS 系统定制与应用开发—系统与应用

参数特性（课程思政）

5.SylixOS 产品成熟，编程简便，系统架构简洁，配合专用的集成开发环境 **RealEvo-IDE** 及硬件模拟器 **RealEvo-Simulator**，便于系统开发与调试，加快软件研发速度，缩短产品研制周期；

6.针对不同的处理器提供优化的驱动程序，提高系统整体性能；

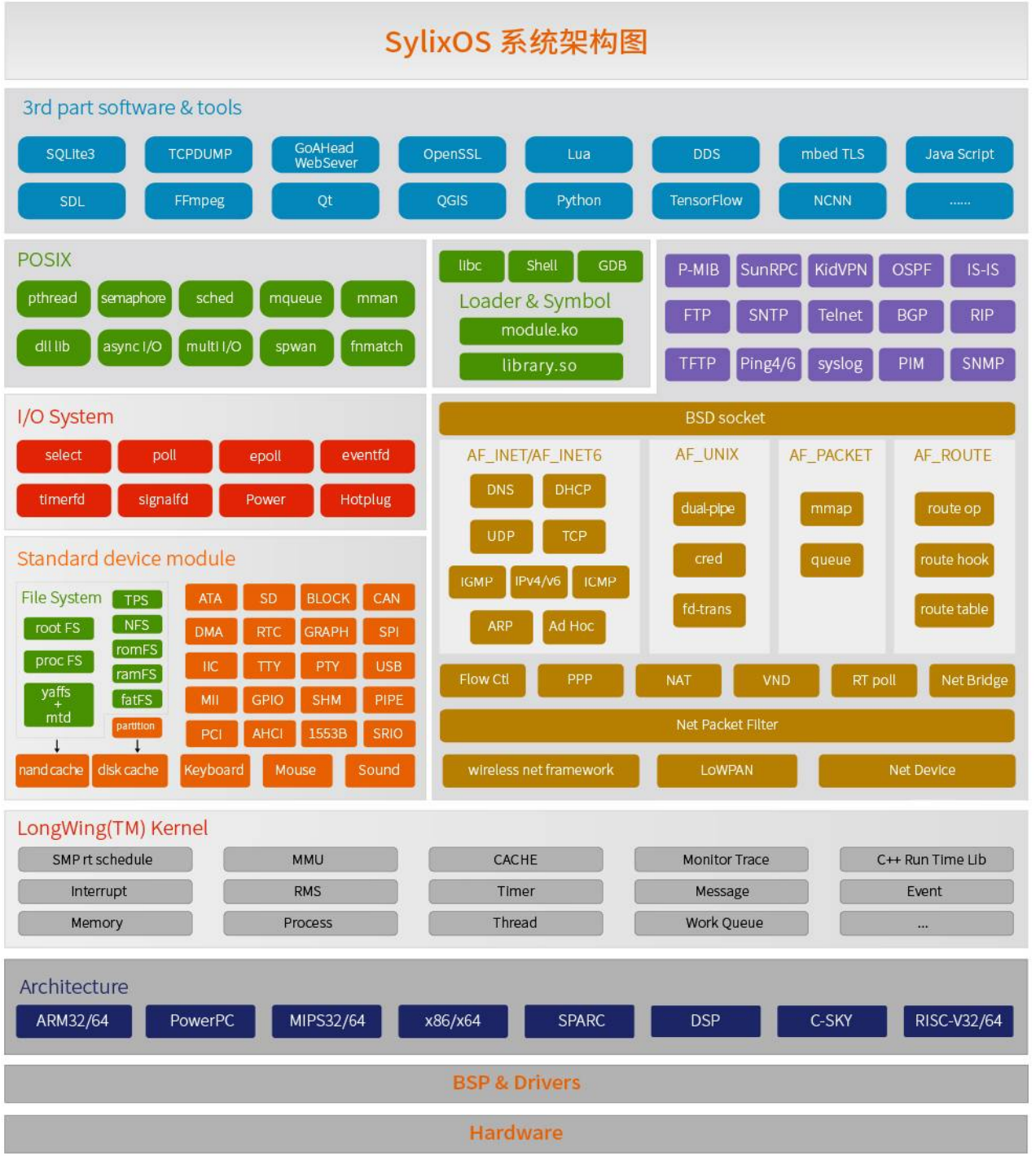
7.硬实时内核，调度算法先进高效，性能强劲；

8.SylixOS 应用编程接口符合 **GJB7714-2012**《军用嵌入式实时操作系统应用编程接口》， 符合 **IEEE**、**ISO**、**IEC** 相关操作系统编程接口规范，用户已有应用程序可方便的迁移到 **SylixOS** 上。

9.SylixOS 支持国家标准可信计算。

➔学生评判了解，与开源代码（系统比较）

5. SylixOS 系统定制与应用开发—系统与应用



SylixOS系统架构

5. SylixOS 系统定制与应用开发—系统与应用

参数特性（学习）

硬实时操作系统，具有如下功能特点：

1. 兼容 IEEE 1003（ISO/IEC 9945）操作系统接口规范；
2. 兼容 POSIX 1003.1b（ISO/IEC 9945-1）实时编程标准；
3. 支持国军标 GJB7714-2012 操作系统接口规范；
4. 优秀的实时性能（任务调度与切换算法时间复杂度为 O(1)）
5. 支持无限多任务；抢占式调度支持 256 个优先级；
6. 支持虚拟进程；支持优先级继承，防止优先级翻转；
7. 极其稳定的内核，很多基于 SylixOS 开发的产品都需要 7x24 小时不间断运行；
8. 支持紧耦合同构多处理器（SMP），ARM Cortex-A9 SMPCore、Intel/AMD 全系列 、龙芯全系列、飞腾 全系列、Freescale i.MX6 系列 、Xilinx Zynq-7000 、Zynq UltraScale+ MPSoC 系列多核处理器；

5. SylixOS 系统定制与应用开发—系统与应用

参数特性（学习）

硬实时操作系统，具有如下功能特点：

9. 根据项目需求可以支持1~2秒启动；12. 支持标准 I/O、多路 I/O 复用与异步 I/O 接口；

10. 支持多种新兴异步事件同步化接口，例如：signalfd、timerfd、eventfd 等；

11. 支持众多标准文件系统：TPSFS（掉电安全）、FAT、YAFFS、ROOTFS、PROCFS、NFS、ROMFS 等；

12. 支持文件记录锁，可支持数据库；

13. 支持内存管理单元（MMU）；

14. 支持第三方 GUI 图形库，Qt、Microwindows、μC/GUI 等；

15. 支持动态装载应用程序、动态链接库以及内核模块；

16. 支持标准 TCP/IPv4/IPv6 双网络协议栈，提供标准socket 操作接口

5. SylixOS 系统定制与应用开发—系统与应用

参数特性（学习）

硬实时操作系统，具有如下功能特点：

17. 支持 AF_UNIX, AF_PACKET, AF_INET, AF_INET6 协议域；

18. 内部集成众多网络工具，例如：FTP、TFTP、NAT、PING、TELNET、NFS 等；

19. 内部集成 Shell 接口、支持环境变量（兼容常用 Linux Shell 操作）；

20. 支持众多标准设备抽象，如：TTY、BLOCK、DMA、ATA、SATA、GRAPH、RTC、PIPE 等；

21. 支持多种工业设备或总线模型，如：CAN、I2C、SPI、SDIO 、PCI/PCIE、1553B、USB 等；

22. 提供高速定时器设备接口，可提供高于主时钟频率的定时服务；支持热插拔设备；支持设备功耗管理；

23. 提供内核行为跟踪器，方便进行应用性能与故障分析；

5. SylixOS 系统定制与应用开发—系统与应用

参数特性（学习）

网络功能

参数特性（学习）

网络功能

1. 支持 10M/100M/1G/10G 以太网；
2. 支持 wireless net framework；
3. 支持 Mesh 网络和 MAODV 自组网协议；
4. 支持主流的 WiFi 和 3G/4G 模块、网卡冗余、虚拟网卡、网桥；
5. 支持 IPv4/IPv6 双网络协议栈，提供标准的 socket 接口；
6. 支持 AF_UNIX、AF_PACKET、AF_INET、AF_INET6、AF_ROUTE 协议域
7. 支持众多网络工具，例如：FTP、TFTP、NAT、PING、TELNET、NFS、PPP、KidVPN、VLAN ；
8. 支持主流工业实时以太网，例如：EtherCAT；
9. 支持丰富的网络中间件，例如： SNTP、libxemail、libcurl、GoAhead-WebServer、DHCP-Server、ACE、TAO、OpenDDS、LCM、pcap、Tcpdump、NcFTP Client、SNTP Server、noPoll、Boa 等；
10. 支持内置规则防火墙、外挂网络防火墙。

参数特性（学习）

质量与安全

QoS 服务质量　**QOS**（Quality of Service服务质量）用于在有限的网络带宽资源下，为不同服务提供不同质量的数据传输保障。

SylixOS 支持传输服务优先级（Priority）和可靠数据传输（Don't drop）两种 QoS 模型：

网络安全

内建网络安全模块：

1：网络登陆黑白名单管理系统： 一旦检测到有针对用户登陆方面的攻击发生，立即会隔离发起攻击的可疑机器（加入黑名单），同时经过一个可设置的冷却时间（也可设置为永久被隔离）后才再次允许对方访问 SylixOS 设备。此模块的保护功能针对 SylixOS 内部所有的内建网络服务模块均有效。

2：网络数据包过滤器：

外挂网络安全模块：

网络安全防火墙

5. SylixOS 系统定制与应用开发—系统与应用

参数特性（学习）

图形显示 （支持第三方 GUI 图形库，Qt、Microwindows、μC/GUI）

1. 支持多屏显示、OpenGL、VNC 远程显示；
2. 支持 Qt、Microwindows、μC/GUI、MiniGUI 等图形用户界面（GUI），支持 Qwt 等第三方 Qt 控件库；
3. 支持触摸屏、键盘、鼠标，支持输入设备热插拔。

（RealEvo-QtSylixOS 软件，方便用户在 Qt Creater 上开发调试应用界面）

5. SylixOS 系统定制与应用开发—基本编程应用

（实验4：RTOS宏内核与应用）

实验4 内容（定制创建应用系统实例）

1）线程创建与管理 掌握线程的概念，掌握线程创建、退出函数的使用方法

2）线程调度 优先级调度/RR（Round-Robin）调度/ POSIX线程调度

SylixOS内核支持256个优先级：0~255。优先级0为最高优先级，优先级255为最低优先级。在线程创建时确定优先级，并允许程序运行中动态修改。但是对于内核而言，从就绪队列中选择一个线程调度时优先级是确定的，也就是说内核不会动态计算每个线程的优先级，因此这种调度策略属于静态调度策略。相对于动态调度策略调度时需要根据某个目标动态确定线程优先级并调度。静态调度策略效率高于动态调度策略。理论上可以支持1024个优先级，但事实上，256个已经足够使用。

对于实时系统，响应速度是最重要的，因此，作为POSIX基本定义的实时扩展，POSIX 1003.1b定义的调度策略都是基于优先级的。其他评价指标，如公平、吞吐量等则是次要的。 POSIX标准规定优先级高的线程优先级数字大，SylixOS中优先级与之相反

POSIX定义结构体sched_param来表示调度相关参数, POSIX定义了以下函数用来动态地选择线程的调度策略（SCHED_FIFO和SCHED_RR）

实验4 内容（定制创建应用系统实例）

3） POSIX通信互斥型通信：

- 共享资源需要独占访问，可以使用信号量、互斥量进行互斥型通信；
- 通知型通信：上述的A线程通知B线程，可以用信号量、事件集、条件变量进行通知型通信；
- 消息型通信：某线程或中断服务程序只负责采集数据，但不直接加工数据，而是将数据传递给另一个线程进行数据加工，可以使用消息队列进行消息型通信。

➔了解线程中信号量、互斥锁和条件变量的概念和作用
掌握信号量、互斥锁和条件变量的相关函数及其使用方法

线程间通信手段	用途
二进制型信号量	互斥型通信、通知型通信
记数型信号量	通知型通信
互斥型信号量	互斥型通信
事件集	通知型通信
条件变量	通知型通信
消息队列	消息型通信

线程间通信手段

5. SylixOS 系统定制与应用开发—应用设计演示

实验5：基于RTOS的应用演示软件设计

（面向实验箱系统：Cortex A9处理器/X86控制器真实系统和运行）

1. 设计编写1-2例基于（X86）SylixOS RTOS的原理性应用（App）程序，在PC本机上虚拟运行应用程序，并记录现象，分析结果，形成所定制开发的RTOS环境开发过程的基本概念；

2.定制创建（指定参数、裁剪构造）1-3例面向实验箱系统的典型基于Cortex A9的目标RTOS基本（Base）系统，编译链接，上载到实验箱目标机，并运行系统，输入系统提供的相关操作命令，记录现象，分析结果，形成嵌入式系统环境下操作系统开发过程的基本概念；

3. 分析理解SylixOS操作系统源程序各组成模块原理和功能，设计修改其中某段程序内容，编译链接，上载到实验箱目标机，并运行实验箱程序，记录现象，分析结果是否与设计相符，进一步加深对嵌入式RTOS技术和编写开发RTOS的认识理解

5. SylixOS 系统定制与应用开发—应用设计演示

（面向实验箱系统：Cortex A9处理器/X86控制器真实系统和运行）

SylixOS-EVB-i.MX6Q 验证平台硬件配置

序号	接口名称	其他说明
1	电路板电源开关	
2	电源输入接口	DC-12V
3	调试串口	DB9 母头
4	MiniUSB	i.MX6Q 处理器的USB OTG接口，可用于系统更新
5	RJ45	i.MX6Q 处理器原生 千兆以太网接口
6	HDMI	i.MX6Q 处理器 原生数字高清接口
7	USB HOST	USB 主接口
8	应用串口	DB9 公头，可以进行串口应用编程
9	音频接口	实现音频的输入输出（左边：麦克风、右边：耳机）
10	Mini-PCIE	PCIEx1接口，
11	CAN	分别是CAN1 CAN2
12	RTC电池	CR2032
13	摄像头接口	连接摄像头模块
14	SATA接口	SATA2.0接口，可以连接硬盘
15	SATA电源接口	向SATA硬盘供电接口（使用电源转接线）
16	SPI扩展模块接口	
17	I2C扩展模块接口	
18	UART扩展模块接口	
19	SD卡	
20	JTAG接口	可以用于JTAG调试
21	i.MX6Q核心板	4核处理器
22	LCD显示	LVDS接口
23	键盘	键盘输入
24	数码管	4位8段数码管
25	阵列按键	4x4阵列按键
26	GPIO按键	用来进行GPIO实验
27	复位按钮	

Cortex A9 四核处理器

IA32 Atone 双核处理器

5. SylixOS 系统定制与应用开发—应用设计演示

嵌入式系统实验箱参数

核心板

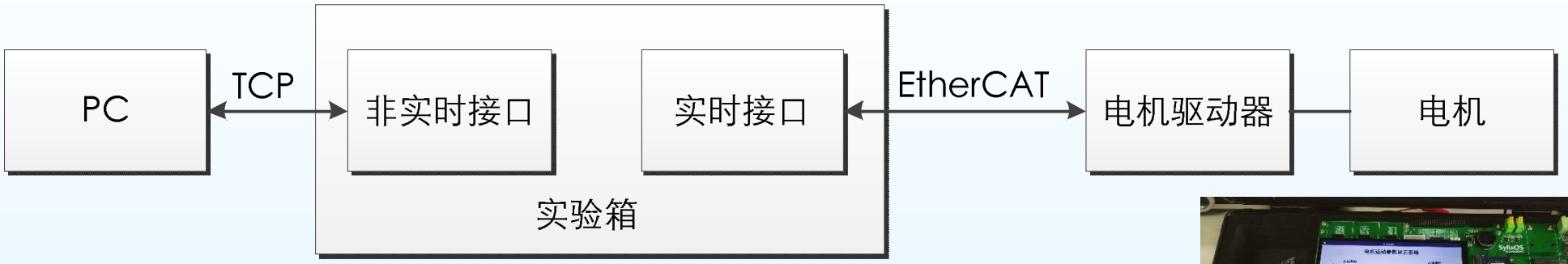
i.MX6Q工业级4核处理器；
1G Byte DDR3内存；
2M Byte SPI Nor Flash，系统默认从SPI Flash启动；
4G Byte eMMC。

底板

1个电源开关；
1个DC-12V电源接口；
3路RS232接口，其中1路为DB9母头做Debug接口，另外2路为DB9 M
1路USB OTG接口；1路千兆以太网接口；
1路HDMI、1路LCD接口、2路LVDS接口，其中1路LVDS接LCD液晶屏；
5路USB HOST接口，其中1路连接键盘；
1对音频输入输出接口；1路mini-PCIE接口；1个摄像头接口；
1个RTC芯片ISL1208；1路SATA接口，配合IDE大4P电源接口；
3个TTL电平扩展模块接口，分别有UART、SPI、I2C；1路SD卡接口；
7路GPIO按键，8路GPIO控制的LED；
zlg7290芯片扩展4位数码管和4x4阵列按键；1路PWM接口。

5. SylixOS 系统定制与应用开发—应用设计演示

（面向实验箱系统：Cortex A9处理器/X86控制器真实系统和运行）



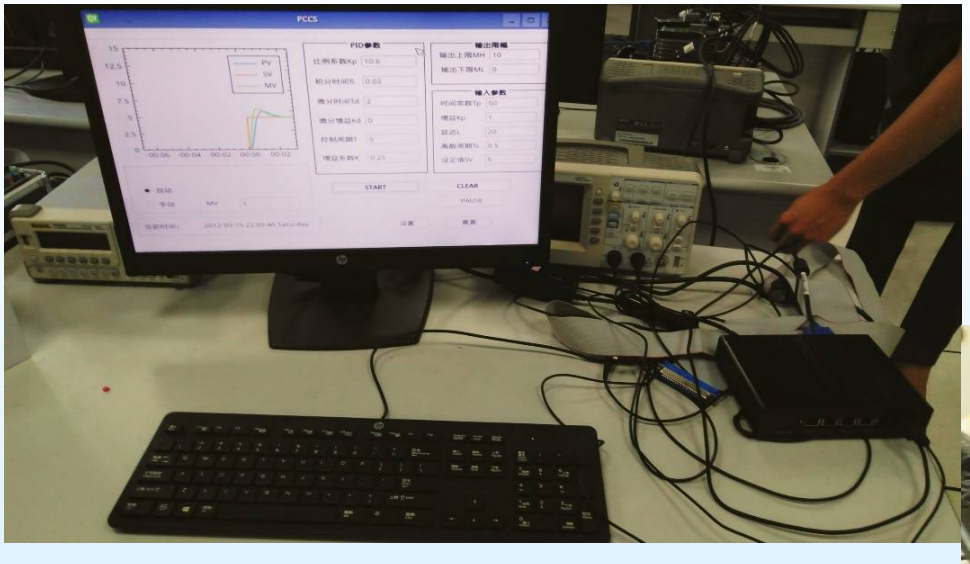
（X86双核工控机）

嵌入式系统：GUI触摸屏
X86工控机：普通键盘显示器

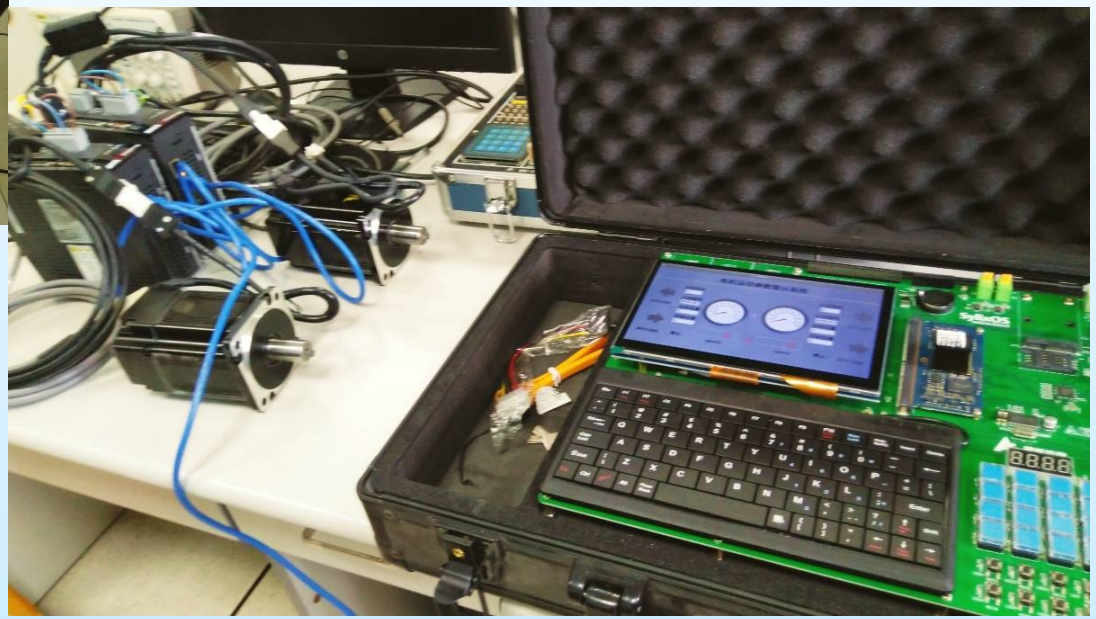
利用按键或GUI触摸屏（QT库）设计实时多任务系统

5. SylixOS 系统定制与应用开发—应用设计演示

（面向实验箱系统：Cortex A9处理器/X86控制器真实系统和运行）



基于SylixOS嵌入式系统：
闭环控制（EtherCAT运动对象—交流伺服电机）



基于SylixOS--X86控制器：
过程回路闭环控制（控制、虚拟对象、MMI任务）

RT多任务扩展：多电机同步控制

5. SylixOS 系统定制与应用开发—应用设计演示

实验内容：

4. 设计编写1-2例基于（Cortex A9）SylixOS RTOS的原理性应用（App）程序，上载到实验箱目标机，并运行设计编写的实验箱应用程序，运行，并记录现象，分析结果，形成所定制开发的RTOS环境开发过程的基本概念；

5、分析基于POSIX标准的线程和任务间通信机制和示例应用程序编写方法，了解中断机制及操作方法。

6、（选做）设计编写引入POSIX和中断技术源程序，分别加入到4中应用程序（APP）中，观察分析运行结果；

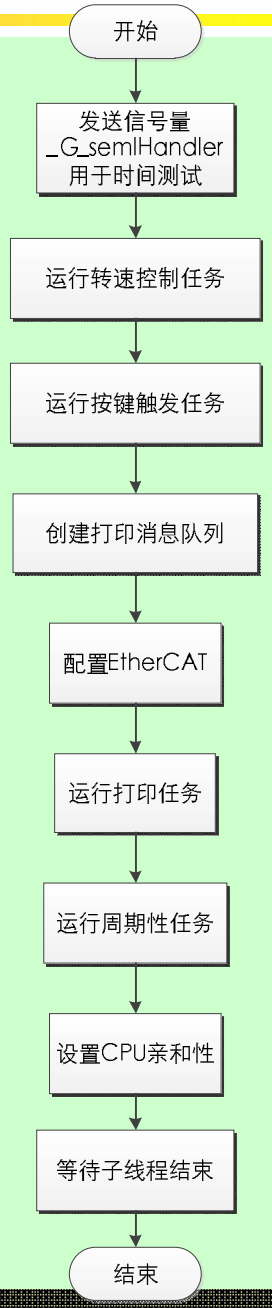
7、（观察分析）实时多任务应用系统开发（高速伺服通信总线下运动控制系统）

8、集中撰写实验报告，整理总结上述实验内容，形成面向嵌入式系统的宏内核之RTOS组成及开发过程。

5. SylixOS 系统定制与应用开发—应用设计演示

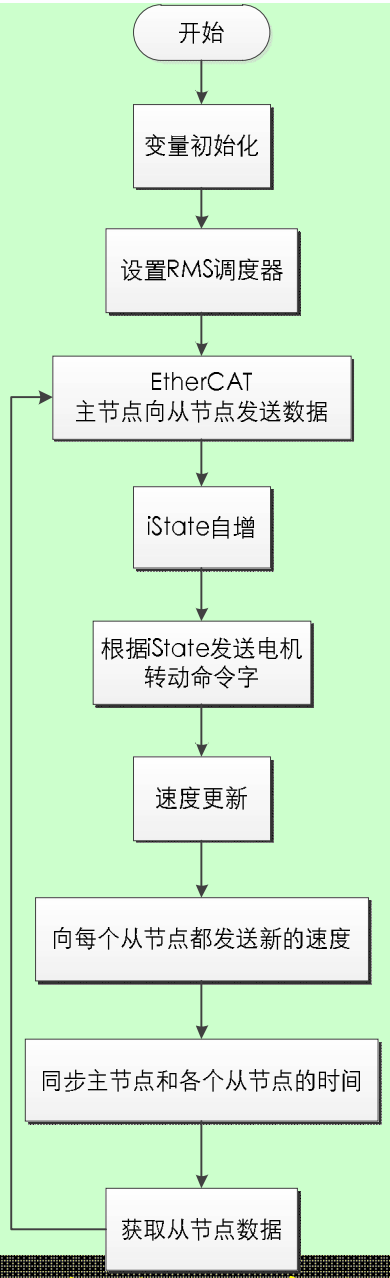
单电机控制示例

主程序流程图



DMS单调速率调度

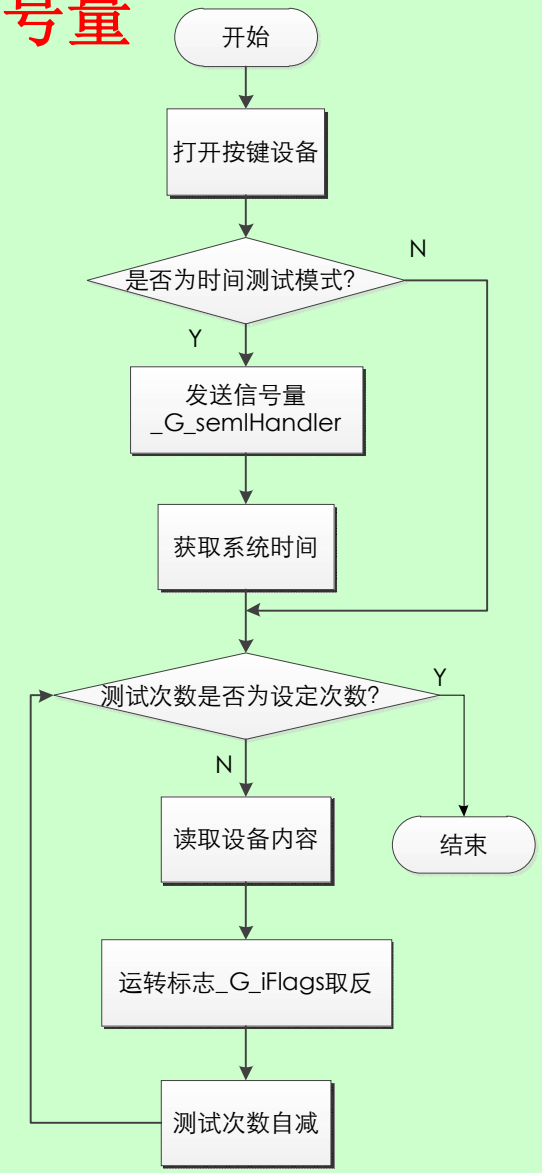
周期性任务流程图



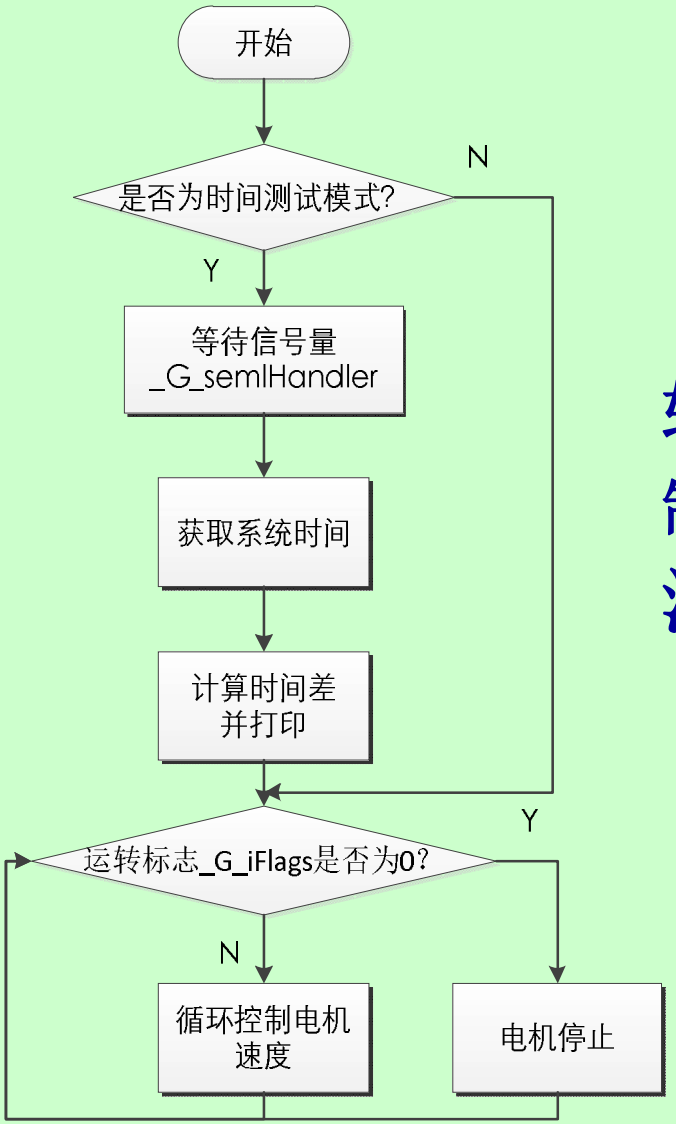
5. SylixOS 系统定制与应用开发—应用设计演示

任务间通信：信号量

按键按键触发任务流程图



转速控制任务流程图



6. 课程综合与成绩评估--设计与实验报告

《嵌入式实时操作系统实践》课程包括两份实验与设计报告，及实验分工、课程总体体会（篇），要求顺序介绍所完成的实验过程和内容，重点介绍实验中要求完成的针对需要所作的源代码修改内容和结果分析，尤其是对实时多任务内核调度和通信及应用源程序架构的创新设计总结。

报告一：针对所完成的**IA32**多任务运行管理程序设计和多任务与**RTOS**内核基础编程相关内容；

报告二：针对所完成的**SylixOS RTOS**定制与应用编程开发相关内容。并对课程全部内容进行总结应在课程结束后一周内提交，并归还相关资料**U**盘。

每组**1-2**人，自由组合

6. 课程综合与成绩评估--设计与实验报告

报告一 IA32多任务调度与RTOS微内核设计开发

1、基础编程实验1 IA32多任务运行管理程序设计

1）开发环境配置与源程序架构简单分析

2）IA32多任务运行调度和任务间通信的基本原理

3）任务指示器和指示结果

4）内核分时优先级调度修改方案

2. 基础编程实验2 多任务微内核OS构建编程开发

1）Vmlinux微内核程序开发工作原理；

2）微内核程序的理解，工程/项目模块结构分析，主要模块源程序分析说明；

3）基于时间片的分时操作系统微内核程序分时调度与通信原型源程序的实现分析；

4）具有优先级和实时特性的RTOS微内核修改方案

6. 课程综合与成绩评估--设计与实验报告

报告二 SylixOS RTOS宏内核定制与应用编程开发

1)SylixOS内核源程序原理架构分析和基本开发方法（不同对象环境）

2)X86系统目标RTOS基本系统构建过程与应用编程原理；模拟运行结果分析

3）嵌入式处理器实验箱目标RTOS基本系统构建与应用（Cortex-A9）基本原理

4）SylixOS操作系统源程序各组成模块原理和功能分析；程序内容设计修改及运行，现象记录分析，总结对嵌入式RTOS技术和编写开发RTOS的认识理解。

5）1-2例基于（Cortex A9）SylixOS RTOS的原理性应用（App）程序设计编写，运行记录，结果分析；总结所定制开发的RTOS环境开发过程；

6）基于POSIX标准的线程和任务间通信机制和示例应用程序编写方法、中断机制及操作方法总结分析。

7）选做的内容： 引入POSIX和中断技术源程序设计编写，分别加入到4中应用程序（APP）中，运行结果观察分析

8）演示系统总结；

报告三 实验课程总结

简单说明课程实验分工、课程体会收获与建议。

6. 课程综合与成绩评估—评估与总结

成绩评估：平时10%

验收40%　　（实验进展检查）

报告50%　　（合作、总结、分析）

课程总结：两轮教学实践，课程不断完善（指导书等资源）

校企（产教）互动（融合）良好

学生总体反映:

1.编程量不足（学时不够)

2.基本概念理论课讲授不到位（感性、理性衔接不理想 –学时不够）

3.实物电机偏少（多电机控制系统演示）

谢 谢！

请大家批评指正