



[www.enjoylinux.cn](http://www.enjoylinux.cn)

# LINUX

## 内核开发-2



版权声明：本课件及其印刷物、视频的版权归成都国嵌信息技术有限公司所有，并保留所有权力：任何单位或个人未经成都国嵌信息技术有限公司书面授权，不得使用该课件及其印刷物、视频从事商业、教学活动。已经取得书面授权的，应在授权范围内使用，并注明“来源：国嵌”。违反上述声明者，我们将追究其法律责任。

# Contents



Linux内存管理

Linux进程地址空间

Linux内核地址空间

Linux内核链表

Linux内核定时器

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# Contents



Linux内存管理

Linux进程地址空间

Linux内核地址空间

Linux内核链表

Linux内核定时器

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 内存管理子系统



**内存**是Linux内核所管理的最重要的资源之一，**内存管理子系统**是操作系统中最重要的部分之一。对于立志从事内核开发的工程师来说，熟悉Linux的内存管理系统非常重要。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 地址类型



✓物理地址

✓线性地址（虚拟地址）

✓逻辑地址

他们之间的关系？

嵌入式Linux技术咨询QQ号：550491596  
嵌入式Linux学习交流QQ群：65212116





# 物理地址



物理地址是指出现在**CPU地址总线**上的寻址物理内存的**地址信号**，是地址变换的最终结果。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 逻辑地址



[www.enjoylinux.cn](http://www.enjoylinux.cn)

程序代码经过编译后  
在汇编程序中使用的地址。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 线性地址



线性地址又名**虚拟地址**，在32位CPU架构下，可以表示4G的地址空间，用16进制表示就是0x00000000到0xffffffff。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116





# 地址转换



**CPU**要将一个**逻辑地址**转换为**物理地址**，需要两步：首先**CPU**利用**段式内存管理单元**，将**逻辑地址**转换成**线性地址**，再利用**页式内存管理单元**，把**线性地址**最终转换为**物理地址**。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 段式、页式管理

✓什么是段式管理?

✓什么是页式管理?



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 段式管理(16位CPU)



[www.enjoylinux.cn](http://www.enjoylinux.cn)

16位CPU内部拥有20位的地址线，它的寻址范围就是2的20次方，也就是1M的内存空间。但是16位CPU用于存放地址的寄存器（IP，SP.....）只有16位，因此只能访问65536个存储单元，64K。

嵌入式Linux技术咨询QQ号：550491596  
嵌入式Linux学习交流QQ群：65212116



# 段式管理(16位CPU)



[www.enjoylinux.cn](http://www.enjoylinux.cn)

为了能够访问1M的内存空间，CPU就采用了内存分段的管理模式，并在CPU内部加入了段寄存器。16位CPU把1M内存空间分为若干个逻辑段，每个逻辑段的要求如下：

- 1、逻辑段的起始地址(段地址)必须是16的倍数，即最后4个二进制位必须全为0。
- 2、逻辑段的最大容量为64K(why?)

嵌入式Linux技术咨询QQ号：550491596  
嵌入式Linux学习交流QQ群：65212116



# 段式管理(16位CPU)



## 物理地址的形成方式:

由于段地址必须是16的倍数，所以值的一般形式为XXXX0H，即前16位二进制位是变化的，后四位是固定的0，鉴于段地址的这种特性，可以只保存前16位二进制位来保存整个段基地址，所以每次使用时要用段寄存器左移补4个0（乘以16）来得到实际的段地址。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116





# 段式管理(16位CPU)



在确定了某个存储单元所属的段后，只是知道了该存储单元所属的范围（**段地址** -> **段地址+65536**），如果想确定该内存单元的具体位置，还必须知道**该单元在段内的偏移**。有了**段地址**和**偏移量**，就可以唯一的确定内存单元在存储器中的具体位置。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 段式管理(16位CPU)



[www.enjoylinux.cn](http://www.enjoylinux.cn)

逻辑地址=段基地址+段内偏移量

由逻辑地址得到物理地址的公式为:

$PA = \text{段寄存器的值} * 16 + \text{逻辑地址}$

为什么要乘16?

嵌入式Linux技术咨询QQ号: 550491596

嵌入式Linux学习交流QQ群: 65212116



# 段式管理(16位CPU)



段寄存器是为了对内存进行分段管理而增加的，  
16位CPU有四个段寄存器，程序可同时访问四个  
不同含义的段。

- 1) **CS + IP** : 用于代码段的访问，**CS** 指向存放程序的段基址，**IP**指向下条要执行的指令在**CS**段的偏移量，用这两个寄存器就可以得到一个内存物理地址，该地址存放着一条要执行的指令。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 段式管理(16位CPU)



2) **SS + SP** : 用于堆栈段的访问, **SS** 指向堆栈段的基地址, **SP** 指向栈顶, 可以通过**SS**和**SP**两个寄存器直接访问栈顶单元的内存物理位置。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 段式管理(16位CPU)



- 3) **DS + BX** : 用于数据段的访问。DS中的值左移四位得到数据段起始地址，再加上BX中的偏移量，得到一个存储单元的物理地址。
- 4) **ES + BX** : 用于附加段的访问。ES中的值左移四位得到附加段起始地址，再加上BX中的偏移量，得到一个存储单元的物理地址。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116





# 段式管理(32位CPU)



32位pc的内存管理仍然采用“分段”的管理模式，逻辑地址同样由段地址和偏移量两部分组成，32位pc的内存管理和16位pc的内存管理有相同之处也有不同之处，因为32位pc采用了两种不同的工作方式：**实模式和保护模式**。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 段式管理(32位CPU)



## 1) 实模式

在实模式下，**32位CPU**的内存管理与**16位CPU**是一致的。

## 2) 保护模式

段基地址长达**32位**，每个段的最大容量可达**4G**，段寄存器的值是段地址的“选择器”(Selector)，用该“选择器”从内存中得到一个**32位**的段地址，存储单元的物理地址就是该段地址加上段内偏移量，这与

**32位CPU**的物理地址计算方式完全不同。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 段式管理(32位CPU)



32位CPU内有6个段寄存器，其值在不同的模式下具有不同的含义：

1、在实模式下：

段寄存器的值\*16就是段地址

2、在保护模式下：

段寄存器的值是一个选择器，间接指出一个32位的段地址

嵌入式Linux技术咨询QQ号：550491596

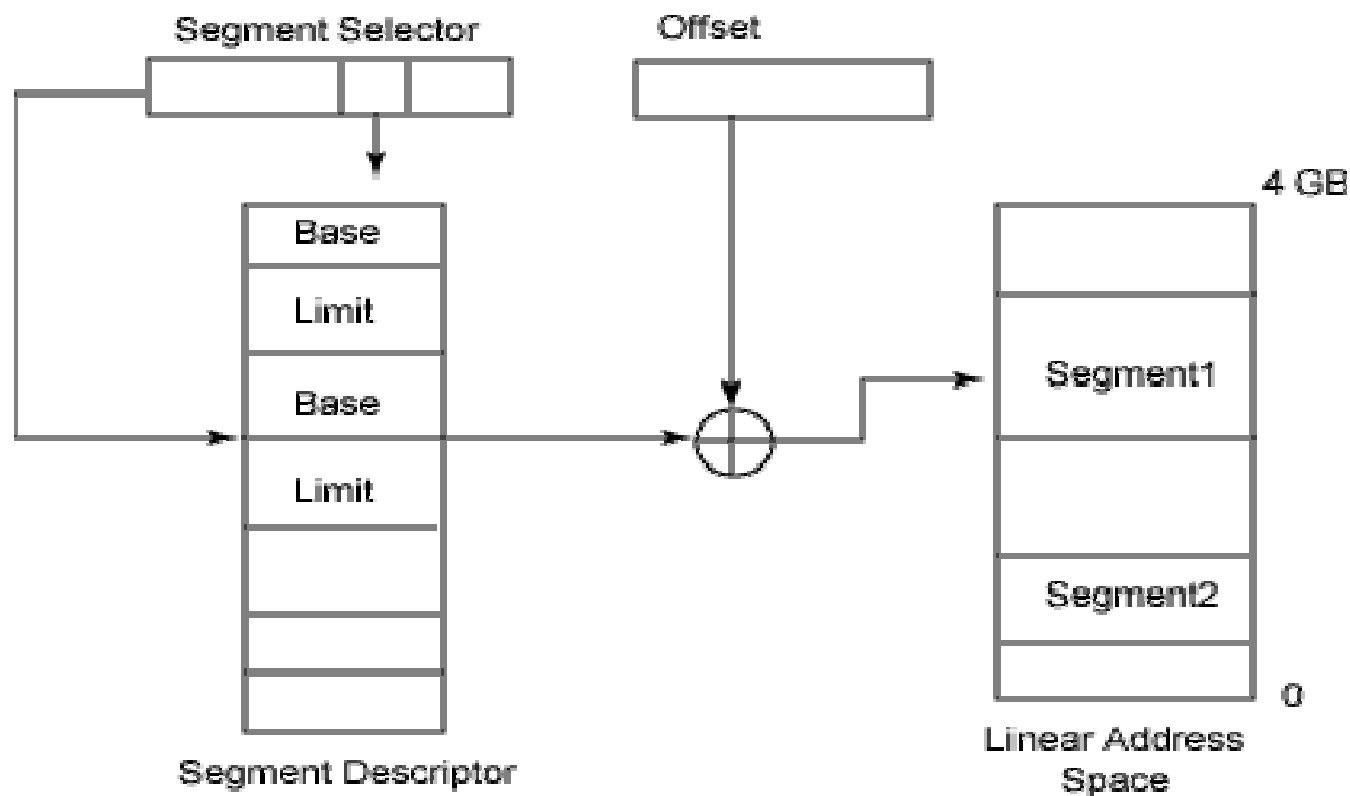
嵌入式Linux学习交流QQ群：65212116



# 段式管理(32位CPU)



[www.enjoylinux.cn](http://www.enjoylinux.cn)



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 分页管理



[www.enjoylinux.cn](http://www.enjoylinux.cn)

从管理和效率的角度出发，**线性地址**被分为固定长度的组，称为**页** (page)，例如**32位**的机器，线性地址最大可为**4G**，如果用**4KB**为一个页来划分，这样整个线性地址就被划分为**2的20次方**个页。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116





# 分页管理



[www.enjoylinux.cn](http://www.enjoylinux.cn)

另一类“页”，称之为**物理页**，或者是**页框**、**页帧**。分页单元把所有的**物理内存**也划分为固定长度的管理单位，它的长度一般与线性地址页是相同的。

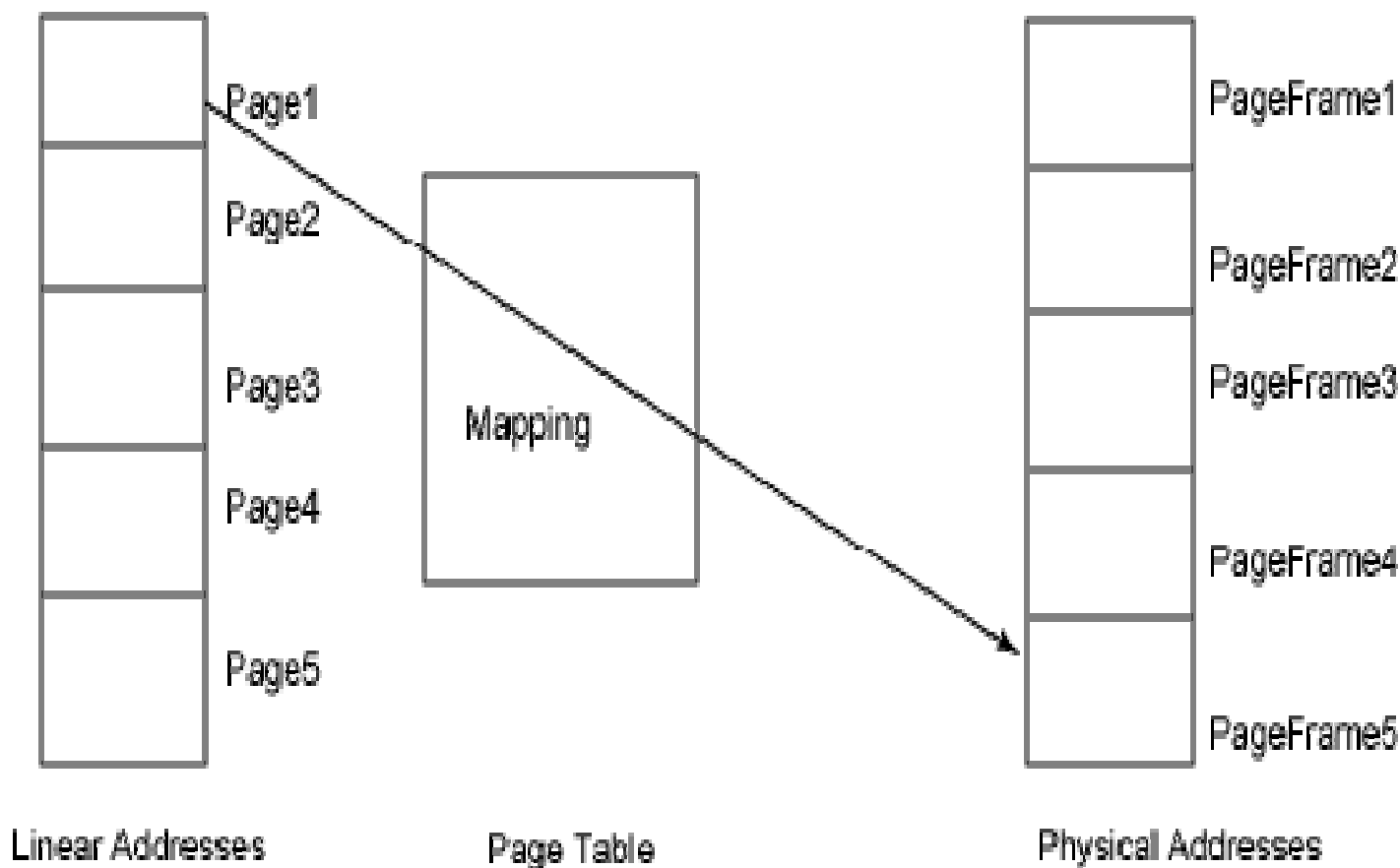
嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 分页管理



[www.enjoylinux.cn](http://www.enjoylinux.cn)



嵌入式Linux技术咨询QQ号: 550491596

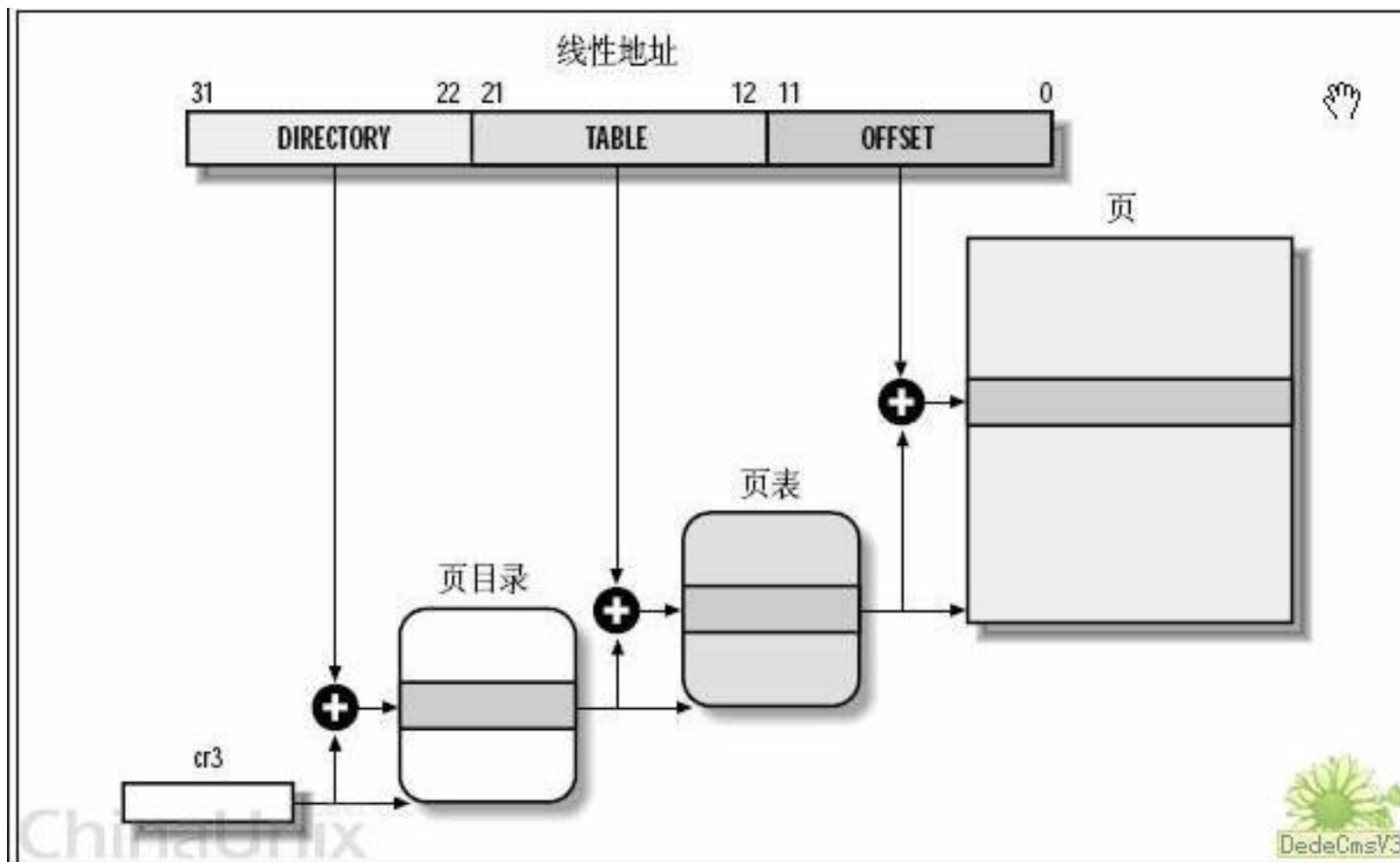
嵌入式Linux学习交流QQ群: 65212116



# 分页管理



[www.enjoylinux.cn](http://www.enjoylinux.cn)



嵌入式Linux技术咨询QQ号: 550491596

嵌入式Linux学习交流QQ群: 65212116

# 分页管理



- 1、分页单元中，页目录的地址放在**CPU的cr3寄存器**中，是进行地址转换的开始点。
- 2、每一个进程，都有其独立的虚拟地址空间，运行一个进程，首先需要将它的页目录地址放到**cr3寄存器**中，将其他进程的保存下来。
- 3、每一个**32位的线性地址**被划分为三部份：**页目录索引(10位)**：**页表索引(10位)**：**偏移(12位)**

**嵌入式Linux技术咨询QQ号: 550491596**  
**嵌入式Linux学习交流QQ群: 65212116**



# 分页管理



依据以下步骤进行地址转换：

- 1、装入进程的页目录地址（操作系统在调度进程时，把这个地址装入CR3）
- 2、根据线性地址前十位，在页目录中，找到对应的索引项，页目录中的项是一个页表的地址
- 3、根据线性地址的中间十位，在页表中找到页的起始地址
- 4、将页的起始地址与线性地址的最后12位相加，得到物理地址

嵌入式Linux技术咨询QQ号：550491596  
嵌入式Linux学习交流QQ群：65212116





# 分页管理



这样的二级模式是否能够覆盖**4G**的物理地址空间？为什么？（通过计算得出结论）

页目录共有： **$2^{10}$** 项，也就是说有这么多个页表；每个目表对应了： **$2^{10}$** 页；  
每个页中可寻址： **$2^{12}$** 个字节。

$$2^{32} = 4\text{GB}$$

嵌入式Linux技术咨询QQ号：550491596  
嵌入式Linux学习交流QQ群：65212116



# Linux内存管理



[www.enjoylinux.cn](http://www.enjoylinux.cn)

Linux内核的设计并没有全部采用Intel所提供的段机制，仅仅是**有限度地**使用了分段机制。这不仅简化了Linux内核的设计，而且为把Linux移植到其他平台创造了条件，因为很多RISC处理器并不支持段机制。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# Linux内存管理



## 所有段的基地址均为0

由此可以得出，每个段的逻辑地址空间范围为0-4GB。因为每个段的基地址为0，因此，**逻辑地址与线性地址保持一致**（即逻辑地址的偏移量字段的值与线性地址的值总是相同的），在Linux中所提到的**逻辑地址和线性地址（虚拟地址）**，可以认为是**一致的**。看来，Linux巧妙地把段机制给绕过去了，而**完全利用了分页机制**。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# Linux页式管理



[www.enjoylinux.cn](http://www.enjoylinux.cn)

前面介绍了i386的二级页管理架构，不过有些CPU(Alpha 64位)使用三级，甚至四级架构，**Linux 2.6.29**内核为每种CPU提供统一的界面，采用了**四级页管理架构**，来兼容二级、三级、四级管理架构的CPU。

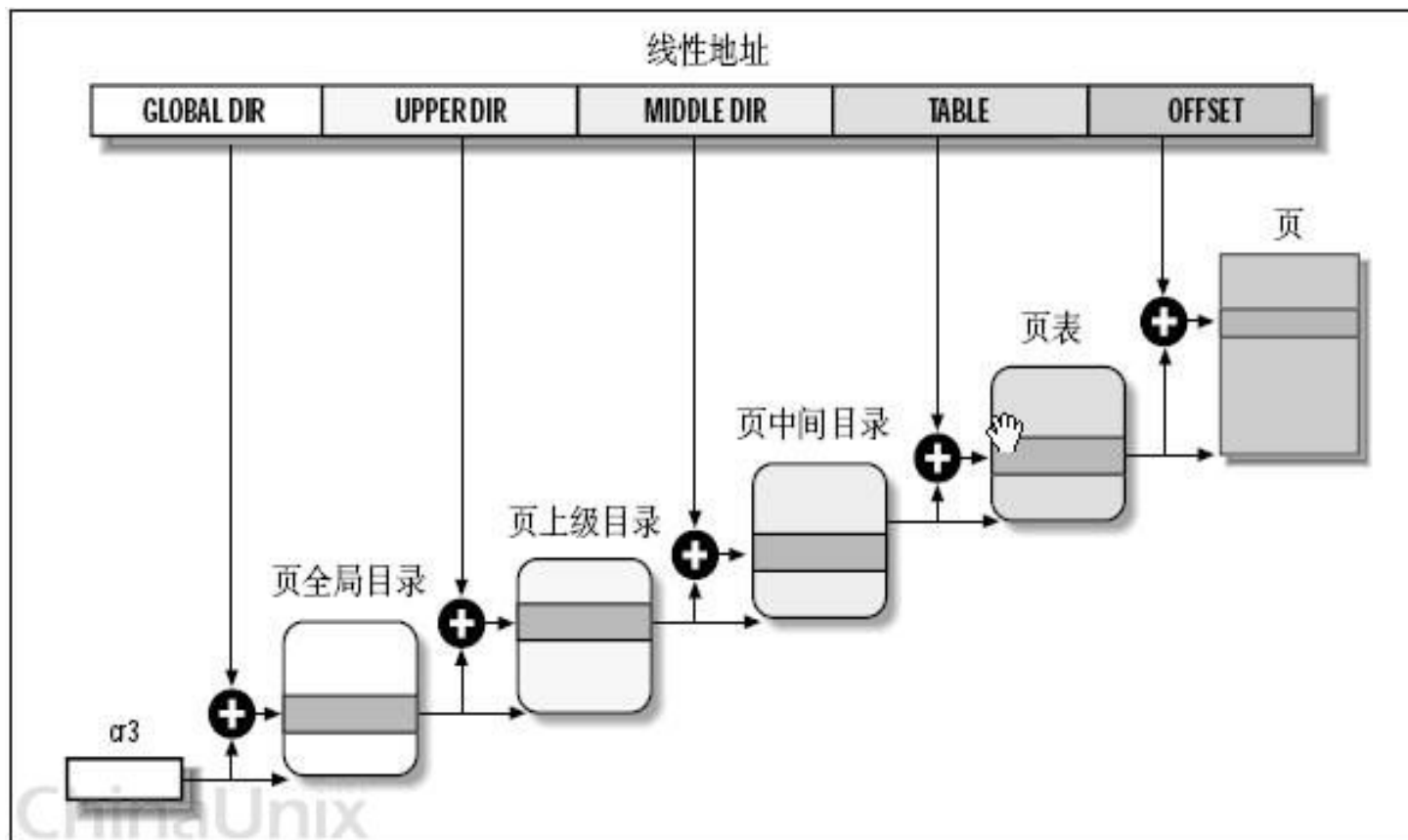
嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# Linux页式管理



[www.enjoylinux.cn](http://www.enjoylinux.cn)



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# Linux页式管理



[www.enjoylinux.cn](http://www.enjoylinux.cn)

这四级分别为:

1. **页全局目录 (Page Global Directory):** 即 pgd, 是多级页表的抽象最高层。
2. **页上级目录 (Page Upper Directory):** 即 pud。
3. **页中间目录 (Page Middle Directory):** 即 pmd, 是页表的中间层。
4. **页表 (Page Table Entry):** 即 pte。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116





# Contents



Linux内存管理

Linux进程地址空间

Linux内核地址空间

Linux内核链表

Linux内核定时器

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 虚拟内存



[www.enjoylinux.cn](http://www.enjoylinux.cn)

Linux操作系统采用虚拟内存管理技术，使得每个进程都有独立的进程地址空间，该空间是大小为**3G**，用户看到和接触的都是虚拟地址，无法看到实际的物理地址。利用这种虚拟地址不但能起到保护操作系统的作用，而且更重要的是用户程序可使用比实际物理内存更大的地址空间。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 虚拟内存



[www.enjoylinux.cn](http://www.enjoylinux.cn)

**Linux**将**4G**的虚拟地址空间划分为两个部分——**用户空间**与**内核空间**。用户空间从**0**到**0xbfffffff**，内核空间从**3G**到**4G**。用户进程**通常情况下**只能访问用户空间的虚拟地址，不能访问内核空间。例外情况是用户进程通过**系统调用**访问内核空间。

嵌入式Linux技术咨询QQ号: 550491596

嵌入式Linux学习交流QQ群: 65212116

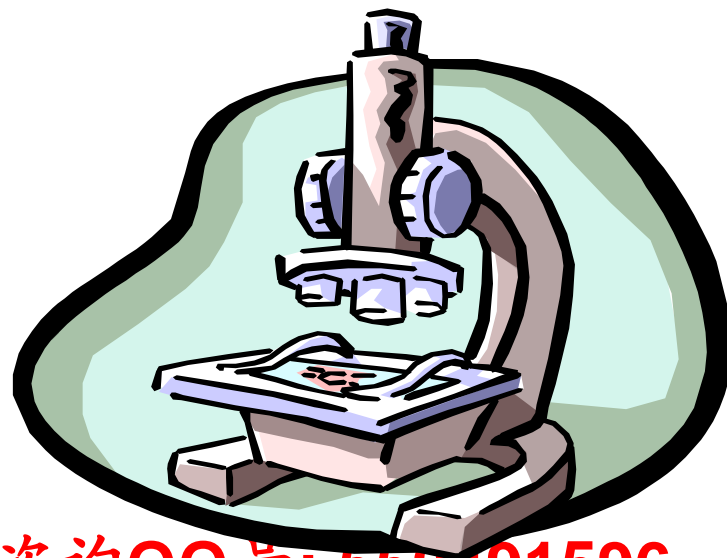


# 进程空间



[www.enjoylinux.cn](http://www.enjoylinux.cn)

用户空间对应进程，所以每当进程切换，用户空间就会跟着变化。



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 进程空间



每个进程的用户空间都是完全独立、互不相干的。把同一个程序同时运行**10次**（为了能同时运行，让它们在返回前睡眠**100秒**），会看到**10个**进程使用的线性地址一模一样。

**( cat /proc/<pid>/maps )**

**嵌入式Linux技术咨询QQ号: 550491596**  
**嵌入式Linux学习交流QQ群: 65212116**



# 进程空间



创建进程**fork()**、程序载入**execve()**、动态内存分配**malloc()**等进程相关操作都需要分配内存给进程。这时进程申请和获得的不是物理地址，仅仅是虚拟地址。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116





# 进程空间



[www.enjoylinux.cn](http://www.enjoylinux.cn)

实际的物理内存只有当进程真的去访问新获取的虚拟地址时，才会由“请页机制”产生“缺页”异常，从而进入分配实际页框的程序。该异常是虚拟内存机制赖以存在的基本保证——它会告诉内核去为进程分配物理页，并建立对应的页表，这之后虚拟地址才实实在在地映射到了物理地址上。

嵌入式Linux技术咨询QQ号: 550491596

嵌入式Linux学习交流QQ群: 65212116



# 内核内存分配



在应用程序中，常使用**malloc**函数进行动态内存分配，而在Linux内核中，通常使用**kmalloc**来动态分配内存。

**kmalloc** 原型是：

✓ **#include <linux/slab.h>**

✓ **void \*kmalloc(size\_t size, int flags)**

参数：

**size**:要分配的内存大小。

**flags**:分配标志，它控制 **kmalloc** 的行为。

嵌入式Linux技术咨询QQ号：550491596

嵌入式Linux学习交流QQ群：65212116



# 分配标志



[www.enjoylinux.cn](http://www.enjoylinux.cn)

最常用的标志是**GFP\_KERNEL**，它的意思是该内存分配是由运行在内核态的进程调用的。也就是说，调用它的函数属于某个进程的，当空闲内存太少时，**kmalloc**函数会使当前进程进入睡眠，等待空闲页的出现。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 分配标志



[www.enjoylinux.cn](http://www.enjoylinux.cn)

如果**kmalloc**是在进程上下文之外调用，比如在中断处理，任务队列处理和内核定时器处理中。这些情况属于中断上下文，不能进入睡眠，这时应该使用优先权**GFP\_ATOMIC**。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 分配标志



## ✓ GFP\_ATOMIC

用来在进程上下文之外的代码（包括中断处理）中分配内存，从不睡眠。

## ✓ GFP\_KERNEL

进程上下文中的分配。可能睡眠。（16M-896M）

## ✓ \_\_GFP\_DMA

这个标志要求分配能够 DMA 的内存区（物理地址在16M以下的页帧）

## ✓ \_\_GFP\_HIGHMEM

这个标志表示分配的内存位于高端内存。（896M以上）

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 按页分配



[www.enjoylinux.cn](http://www.enjoylinux.cn)

如果模块需要分配大块的内存，那使用面向页的分配技术会更好

✓ **get\_zeroed\_page**(unsigned int flags)

返回指向新页面的指针,并将页面清零。

✓ **\_\_get\_free\_page**(unsigned int flags)

和get\_free\_page类似，但不清零页面。

✓ **\_\_get\_free\_pages**(unsigned int flags,unsigned int order )

分配若干个连续的页面，返回指向该内存区域的指针，但也不清零这段内存区域。

嵌入式Linux技术咨询QQ号: 550491596

嵌入式Linux学习交流QQ群: 65212116





# 释放



[www.enjoylinux.cn](http://www.enjoylinux.cn)

当程序用完这些页, 可以使用下列函数之一来释放它们:

✓ void **free\_page**(unsigned long addr)

✓ void **free\_pages**(unsigned long addr, unsigned long order)

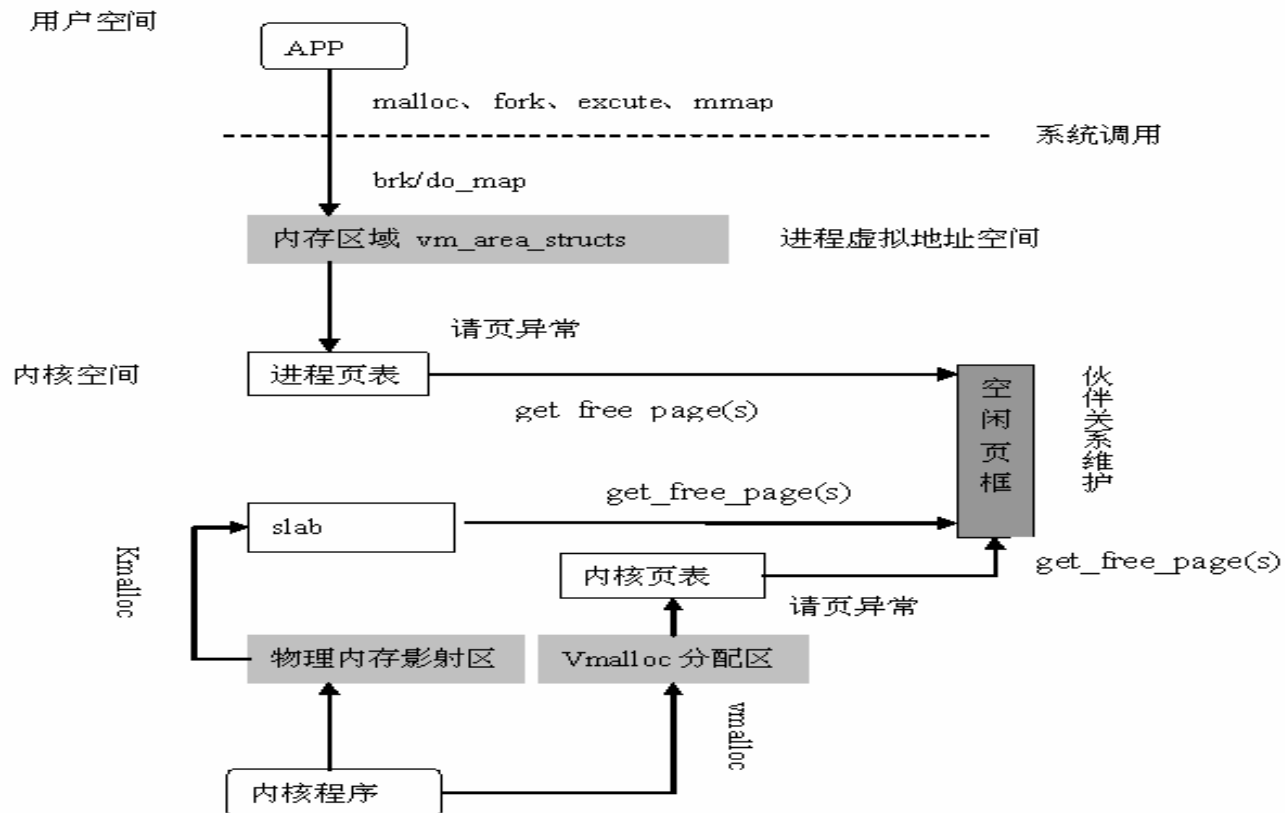
**\*\*如果释放的和先前分配数目不等的页面, 会导致系统错误\*\***

**嵌入式Linux技术咨询QQ号: 550491596**

**嵌入式Linux学习交流QQ群: 65212116**



# 内存使用



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 实验



[www.enjoylinux.cn](http://www.enjoylinux.cn)

## 内核内存分配

### 编写内核模块

- 1、在模块中使用**kmalloc**分配内存，并使用分配的内存。
- 2、在模块中使用按页分配分配内存，并使用分配的内存。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# Contents



Linux内存管理

Linux进程地址空间

Linux内核地址空间

Linux内核链表

Linux内核定时器

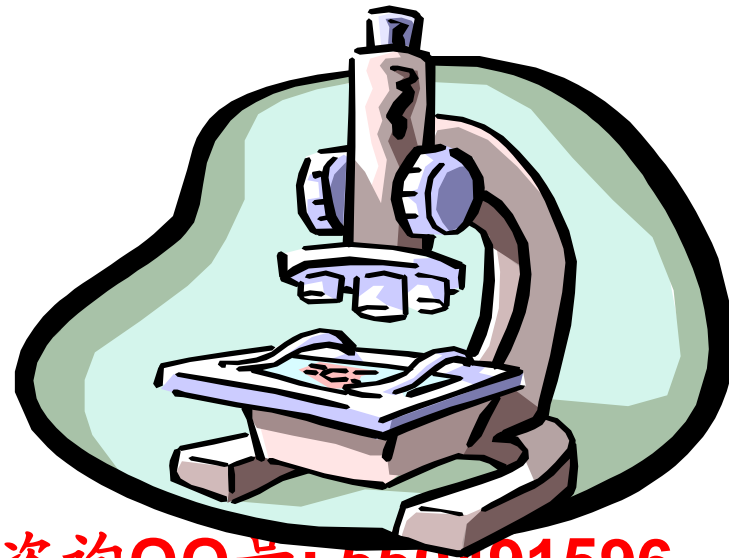
嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 内核空间



内核空间是由内核负责映射，它并不会跟着进程改变，是固定的。



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116

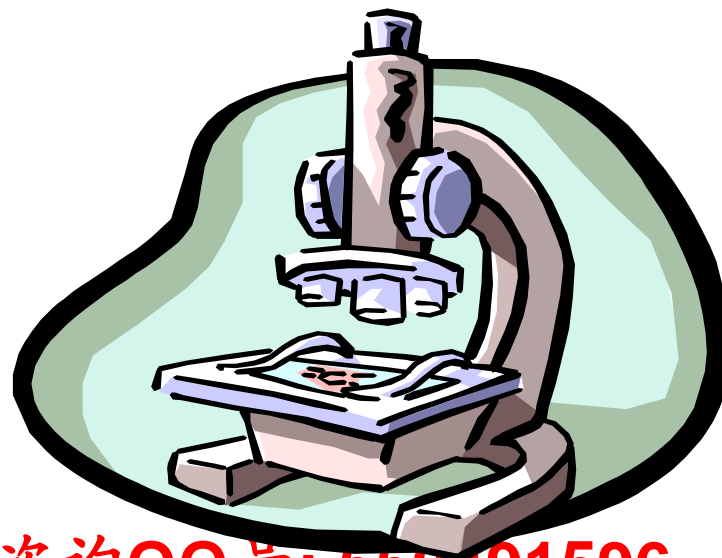


# 高端内存



[www.enjoylinux.cn](http://www.enjoylinux.cn)

物理内存896MB以上的部分称之为高端内存。



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116





# 内核空间分布



[www.enjoylinux.cn](http://www.enjoylinux.cn)

3G				4G		
直接映射区	8M	动态映射区	8K	KMAP区	固定映射区	4K
896M(max)	120M(min)		4M	4M		

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 内核空间分布



## 直接内存映射区 ( Direct Memory Region )

从**3G**开始，**最大896M**的线性地址区间，我们称作**直接内存映射区**，这是因为该区域的线性地址和物理地址之间存在线性转换关系

**线性地址 = 3G + 物理地址**

例：

物理地址区间**0x100000-0x200000**映射到线性空间就是**3G+0x100000-3G+0x200000**。

**嵌入式Linux技术咨询QQ号：550491596**

**嵌入式Linux学习交流QQ群：65212116**



# 内核空间分布



[www.enjoylinux.cn](http://www.enjoylinux.cn)

## 动态内存映射区 ( Vmalloc Region )

该区域的地址由内核函数**vmalloc**来进行分配，其特点是线性空间连续，但对应的物理空间不一定连续。**vmalloc**分配的线性地址所对应的物理页可能处于低端内存，也可能处于高端内存。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 内核空间分布



[www.enjoylinux.cn](http://www.enjoylinux.cn)

## 永久内存映射区 ( PKMap Region )

对于896MB以上的高端内存，可使用该区域来访问，访问方法：

1. 使用`alloc_page(__GFP_HIGHMEM)`分配高端内存页
2. 使用`kmap`函数将分配到的高端内存映射到该区域

嵌入式Linux技术咨询QQ号：550491596  
嵌入式Linux学习交流QQ群：65212116



# 内核空间分布



## 固定映射区 ( Fixing Mapping Region )

**PKMap**区上面，有**4M**的线性空间，被称作固定映射区，它和**4G**顶端只有**4K**的隔离带。固定映射区中每个地址项都服务于特定的用途，如**ACPI\_BASE**等。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# Contents



Linux内存管理

Linux进程地址空间

Linux内核地址空间

Linux内核链表

Linux内核定时器

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116





# 简介



[www.enjoylinux.cn](http://www.enjoylinux.cn)

链表是一种常用的数据结构，它通过**指针**将一系列**数据节点**连接成一条**数据链**。相对于数组，链表具有更好的动态性，**建立链表时无需预先知道数据总量，可以随机分配空间，可以高效地在链表中的任意位置实时插入或删除数据**。链表的开销主要是访问的顺序性和组织链的空间损失。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 简介



[www.enjoylinux.cn](http://www.enjoylinux.cn)

通常链表数据结构**至少**包含**两个域**：**数据域**和**指针域**，数据域用于存储数据，指针域用于建立与下一个节点的联系。按照指针域的组织以及各个节点之间的联系形式，链表又可以分为**单链表**、**双链表**、**循环链表**等多种类型。

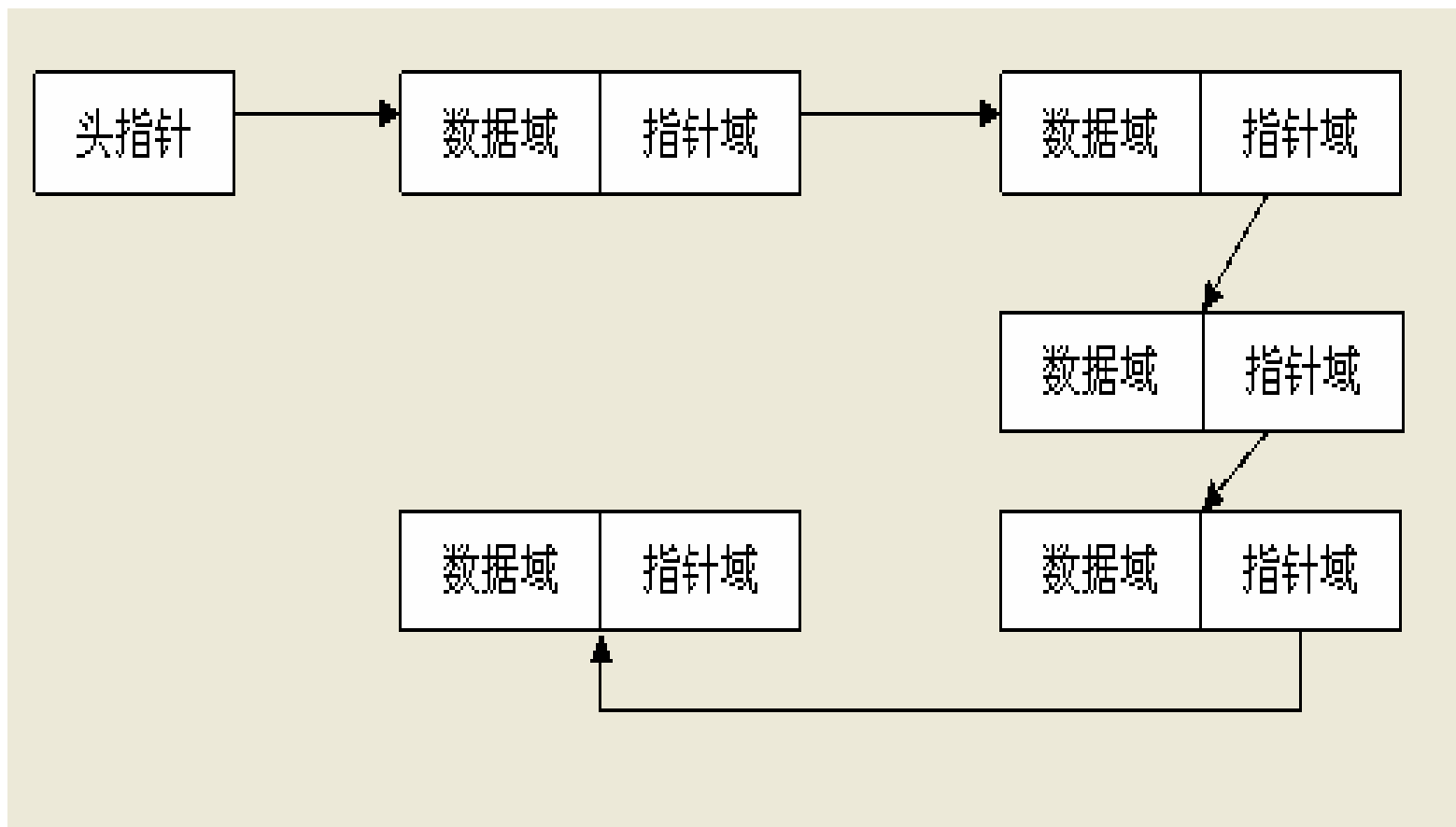
嵌入式Linux技术咨询QQ号：550491596  
嵌入式Linux学习交流QQ群：65212116





[www.enjoylinux.cn](http://www.enjoylinux.cn)

# 单向链表



嵌入式Linux技术咨询QQ号: 550491596

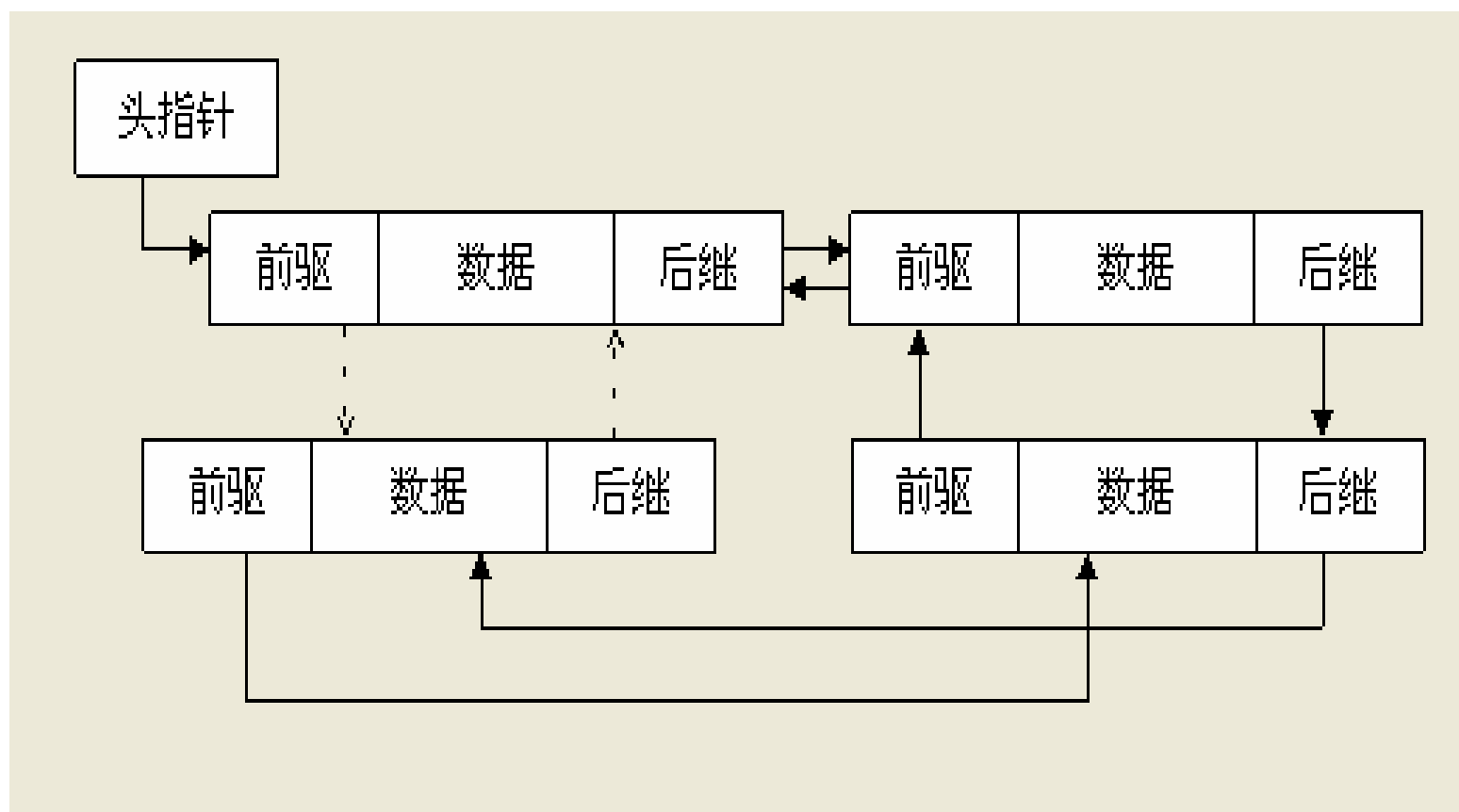
嵌入式Linux学习交流QQ群: 65212116



# 双向链表



[www.enjoylinux.cn](http://www.enjoylinux.cn)



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 内核链表



在Linux内核中使用了大量的链表结构来组织数据。这些链表大多采用了[include/linux/list.h]中实现的一套精彩的链表数据结构。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 内核链表



链表数据结构的定义:

```
struct list_head
{
    struct list_head *next, *prev;
};
```

**list\_head**结构包含两个指向**list\_head**结构的指针**prev**和**next**，由此可见，**内核的链表**具备双链表功能，实际上，通常它都组织成**双向循环链表**。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116

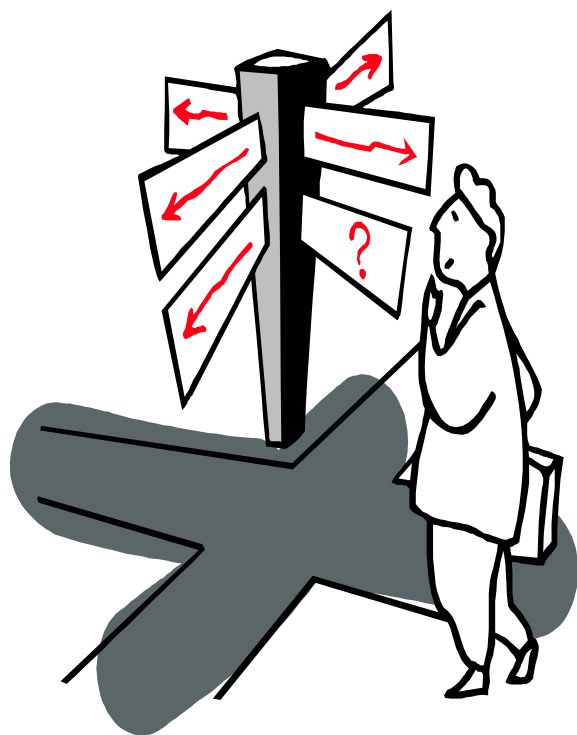




# 传统链表PK内核链表



[www.enjoylinux.cn](http://www.enjoylinux.cn)



## 图示区别!

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 链表操作



Linux内核中提供的链表操作主要有:

✓ 初始化链表头

**INIT\_LIST\_HEAD**(list\_head \*head)

✓ 插入节点

**list\_add**(struct list\_head \*new, struct list\_head \*head)

**list\_add\_tail**(struct list\_head \*new, struct list\_head \*head)

嵌入式Linux技术咨询QQ号: 550491596

嵌入式Linux学习交流QQ群: 65212116



# 链表操作



[www.enjoylinux.cn](http://www.enjoylinux.cn)

✓ 删除节点

**list\_del**(struct list\_head \*entry)

✓ 提取数据结构

**list\_entry**(ptr, type, member)

已知数据结构中的节点指针ptr，找出数据结构，例：

**list\_entry(aup, struct autofs, list)**

嵌入式Linux技术咨询QQ号：550491596

嵌入式Linux学习交流QQ群：65212116



# 链表操作



## ✓ 遍历

**list\_for\_each**(`struct list_head *pos, struct list_head *head`)

例:

```
struct list_head *entry;  
struct list_head cs46xx_devs; //链表头
```

```
list_for_each(entry, &cs46xx_devs)  
{  
    card = list_entry(entry, struct cs_card, list);  
    if (card->dev_midi == minor)  
        break;  
}
```

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 链表操作



[www.enjoylinux.cn](http://www.enjoylinux.cn)

## 实例分析



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 实验



[www.enjoylinux.cn](http://www.enjoylinux.cn)

## 内核链表操作

### 编写内核模块

- 1、完成结点的插入
- 2、完成链表的遍历
- 3、完成结点的删除

( 遍历时访问结点中的数据 )

嵌入式Linux技术咨询QQ号: 550491596

嵌入式Linux学习交流QQ群: 65212116





# Contents



Linux内存管理

Linux进程地址空间

Linux内核地址空间

Linux内核链表

Linux内核定时器

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 度量时间差



[www.enjoylinux.cn](http://www.enjoylinux.cn)

**时钟中断**由系统的定时硬件以周期性的时间间隔产生，这个间隔（即频率）由内核根据**HZ**来确定，**HZ**是一个与体系结构无关的常数，可配置（**50-1200**），在**X86**平台，默认值为**1000**。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 度量时间差



[www.enjoylinux.cn](http://www.enjoylinux.cn)

每当时钟中断发生时，全局变量 **jiffies**(unsigned long)就加1，因此 **jiffies**记录了自linux启动后时钟中断发生的次数。驱动程序常利用**jiffies**来计算不同事件间的时间间隔。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 延迟执行



[www.enjoylinux.cn](http://www.enjoylinux.cn)

如果对延迟的精度要求不高，最简单的实现方法如下--忙等待：

```
✓ unsigned long j=jiffies + jit_delay*HZ;
```

```
✓ while (jiffies<j)
```

```
{
```

```
    /* do nothing */
```

```
}
```

嵌入式Linux技术咨询QQ号：550491596

嵌入式Linux学习交流QQ群：65212116



# 内核定时器



定时器用于控制某个函数(定时器处理函数)在未来的某个特定时间执行。内核定时器注册的处理函数只执行一次--不是循环执行的。

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 内核定时器



内核定时器被组织成双向链表，并使用**struct timer\_list**结构描述。

```
struct timer_list {  
    struct list_head entry /*内核使用*/;  
    unsigned long expires; /*超时的jiffies值*/  
    void (*function)(unsigned long); /*超时处理函数*/  
    unsigned long data; /*超时处理函数参数*/  
    struct tvec_base *base; /*内核使用*/  
};
```

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116





# 内核定时器



操作定时器的有如下函数:

- ✓ **void init\_timer(struct timer\_list \*timer);**  
初始化定时器队列结构。
- ✓ **void add\_timer(struct timer\_list \* timer);**  
启动定时器。
- ✓ **int del\_timer(struct timer\_list \*timer);**  
在定时器超时前将它删除。当定时器超时后，系统会自动地将它删除。

**嵌入式Linux技术咨询QQ号: 550491596**  
**嵌入式Linux学习交流QQ群: 65212116**



# 内核定时器



[www.enjoylinux.cn](http://www.enjoylinux.cn)

## 实例分析



嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116



# 实验



## 内核定时器

### 编写内核模块实现定时器

嵌入式Linux技术咨询QQ号: 550491596  
嵌入式Linux学习交流QQ群: 65212116

