Linux Device Drivers - memory allocation

Jernej Vičič

Overview

Introduction

- *kmalloc* and *kfree*,
- allocation and freeing of memory,
- there are different ways,
- we have no problems with segmentation, paging,
- core has *unified memory management interface*.

kmalloc

- this function is fast,
- can block,
- does not clean the memory (previous content is still there),
- contiguous in physical memory,

kmalloc - flags

```
#include <linux/slab.h>
void *kmalloc(size_t size, int flags);
```

- prototip,
- *size* – block size,
- *flags* – flags,
- *GFP_KERNEL* – most used,
  - *__get_free_pages* – is called,
  - calling function is executing a system call on behalf of a process.
- current process cannot sleep,
- use *GFP_ATOMIC*.

kmalloc - flags

- sometimes kmalloc is called out of the process context:
  - *GFP_ATOMIC* – interrupt handlers, other code outside the context of the process, never sleeps,
  - *GFP_KERNEL* – normal allocation, can sleep,
  - *GFP_USER* – used for allocation of user pages, can sleep,
  - *GFP_HIGHUSER* – same as USER, but allocates *high memory*
  - *GFP_NOFS* – system calls to the file system cannot be performed during allocation.
  - *GFP_NOIO* – system calls to the I/O cannot be performed during allocation.

kmalloc – memory zones

- memory zones,
- depends on architecture,
- kernel distinguishes min 3 zones:
  - DMA-capable memory, space that can be accessed by peripheral devices,
  - normal memory,
  - high memory, the mechanism for accessing large pieces of memory, we can not directly address from the kernel,