in arch/arm/include/asm/cache.h

```
#define L1_CACHE_SHIFT          CONFIG_ARM_L1_CACHE_SHIFT
#define L1_CACHE_BYTES          (1 << L1_CACHE_SHIFT)

/*
 * Memory returned by kmalloc() may be used for DMA, so we must make
 * sure that all such allocations are cache aligned. Otherwise,
 * unrelated code may cause parts of the buffer to be read into the
 * cache before the transfer is done, causing old data to be seen by
 * the CPU.
 */
#define ARCH_DMA_MINALIGN       L1_CACHE_BYTES

/*
 * With EABI on ARMv5 and above we must have 64-bit aligned slab pointers.
 */
#if defined(CONFIG_AEABI) && (__LINUX_ARM_ARCH__ >= 5)
#define ARCH_SLAB_MINALIGN 8
#endif

#define __read_mostly __attribute__((__section__(".data..read_mostly")))
```

in config-3.18.7-yocto-standard

```
CONFIG_ARM_L1_CACHE_SHIFT_6=y
CONFIG_ARM_L1_CACHE_SHIFT=6
```

即ARMv7 Cortext A53的L1 cache line size = 2 ^6 = 32 bytes

ARCH_DMA_MINALIGN 用于

```
#ifdef CONFIG_HAS_DMA
static inline int dma_get_cache_alignment(void)
{
#ifdef ARCH_DMA_MINALIGN
        return ARCH_DMA_MINALIGN;
#endif
        return 1;
}
#endif
```

从comment看，kmalloc()分配的space必须对齐在L1 cache line上，因为kmalloc()返回的memory

可能用于DMA传输。

The __read_mostly modifier make the variable be put into ".data..read_mostly" section.

in arch/arm/kernel/vmlinux.lds.S

```
        .data : AT(__data_loc) {
                _data = .;              /* address in memory */
                _sdata = .;

                /*
                 * first, the init task union, aligned
                 * to an 8192 byte boundary.
                 */
                INIT_TASK_DATA(THREAD_SIZE)

#ifdef CONFIG_XIP_KERNEL
                . = ALIGN(PAGE_SIZE);
                __init_begin = .;
                INIT_DATA
                ARM_EXIT_KEEP(EXIT_DATA)
                . = ALIGN(PAGE_SIZE);
                __init_end = .;
#endif

                NOSAVE_DATA
                CACHELINE_ALIGNED_DATA(L1_CACHE_BYTES)
                READ_MOSTLY_DATA(L1_CACHE_BYTES)

                /*
                 * and the usual data section
                 */
                DATA_DATA
                CONSTRUCTORS

                _edata = .;
        }
```

in include/asm-generic/vmlinux.lds.h

```
#define READ_MOSTLY_DATA(align)                                         \
        . = ALIGN(align);                                               \
        *(.data..read_mostly)                                           \
        . = ALIGN(align);
```

in vmlinux.lds

```
    . = ALIGN((1 << 6)); *(.data..read_mostly) . = ALIGN((1 << 6));
```

对齐在L1 cache line上！

而".data..read_mostly" section则是放在.data segment。

kernel在载入时好像并没有对它作特殊处理！