include

```
# define swab16 __swab16
# define swab32 __swab32
# define swab64 __swab64
# define swahw32 __swahw32
# define swahb32 __swahb32
# define swab16p __swab16p
# define swab32p __swab32p
# define swab64p __swab64p
# define swahw32p __swahw32p
# define swahb32p __swahb32p
# define swab16s __swab16s
# define swab32s __swab32s
# define swab64s __swab64s
# define swahw32s __swahw32s
# define swahb32s __swahb32s
```

__swab16 - return a byteswapped 16-bit value
__swab32 - return a byteswapped 32-bit value
__swab64 - return a byteswapped 64-bit value

__swahw32 - return a word-swapped 32-bit value
__swahw32(0x12340000) is 0x00001234

__swahb32 - return a high and low byte-swapped 32-bit value
__swahb32(0x12345678) is 0x34127856

__swab16p - return a byteswapped 16-bit value from a pointer

静态判断当前kernel工作在哪种byte order之下，通过

> __LITTLE_ENDIAN_BITFIELD
> __BIG_ENDIAN_BITFIELD

for example:

in drivers/w1/w1.h

```
struct w1_reg_num
{
#if defined(__LITTLE_ENDIAN_BITFIELD)
    __u64   family:8,
        id:48,
        crc:8;
#elif defined(__BIG_ENDIAN_BITFIELD)
    __u64   crc:8,
        id:48,
        family:8;
#else
#error "Please fix <asm/byteorder.h>"
#endif
};
```

Generic Byte-reordering support

in include/linux/byteorder/generic.h

```
#define cpu_to_le64 __cpu_to_le64
#define le64_to_cpu __le64_to_cpu
#define cpu_to_le32 __cpu_to_le32
#define le32_to_cpu __le32_to_cpu
#define cpu_to_le16 __cpu_to_le16
#define le16_to_cpu __le16_to_cpu
#define cpu_to_be64 __cpu_to_be64
#define be64_to_cpu __be64_to_cpu
#define cpu_to_be32 __cpu_to_be32
#define be32_to_cpu __be32_to_cpu
#define cpu_to_be16 __cpu_to_be16
#define be16_to_cpu __be16_to_cpu
#define cpu_to_le64p __cpu_to_le64p
#define le64_to_cpup __le64_to_cpup
#define cpu_to_le32p __cpu_to_le32p
#define le32_to_cpup __le32_to_cpup
#define cpu_to_le16p __cpu_to_le16p
#define le16_to_cpup __le16_to_cpup
#define cpu_to_be64p __cpu_to_be64p
#define be64_to_cpup __be64_to_cpup
#define cpu_to_be32p __cpu_to_be32p
#define be32_to_cpup __be32_to_cpup
#define cpu_to_be16p __cpu_to_be16p
#define be16_to_cpup __be16_to_cpup
#define cpu_to_le64s __cpu_to_le64s
#define le64_to_cpus __le64_to_cpus
#define cpu_to_le32s __cpu_to_le32s
#define le32_to_cpus __le32_to_cpus
#define cpu_to_le16s __cpu_to_le16s
#define le16_to_cpus __le16_to_cpus
#define cpu_to_be64s __cpu_to_be64s
#define be64_to_cpus __be64_to_cpus
#define cpu_to_be32s __cpu_to_be32s
#define be32_to_cpus __be32_to_cpus
#define cpu_to_be16s __cpu_to_be16s
#define be16_to_cpus __be16_to_cpus
```