u-boot在启动kernel时会传递boot params给kernel

for example:

bootcmd: mmc dev 1;ext2load mmc 1:2 0x400000 /boot/uImage;ext2load mmc 1:2 0xf00000 /boot/mv6220-toc.dtb;setenv bootargs $bootargs root=/dev/mmcblk1p2 uio_pdrv_genirq.of_id=generic-uio rootwait;bootm 0x400000 - 0xf00000

kernel对这些params的处理一般如下

memblock=debug

```
static int __init early_memblock(char *p)
{
        if (p && strstr(p, "debug"))
                memblock_debug = 1;
        return 0;
}
early_param("memblock", early_memblock);
```

bootmem_debug

```
static int bootmem_debug;

static int __init bootmem_debug_setup(char *buf)
{
        bootmem_debug = 1;
        return 0;
}
early_param("bootmem_debug", bootmem_debug_setup);
```

earlyprintk

```
static int __init setup_early_printk(char *buf)
{
        if (!early_console) {
                early_console = &early_console_dev;
                register_console(&early_console_dev);
        }
        return 0;
}

early_param("earlyprintk", setup_early_printk);
```

```
/*
 * Only for really core code.  See moduleparam.h for the normal way.
 *
 * Force the alignment so the compiler doesn't space elements of the
 * obs_kernel_param "array" too far apart in .init.setup.
 */
#define __setup_param(str, unique_id, fn, early)                        \
        static const char __setup_str_##unique_id[] __initconst \
                __aligned(1) = str; \
        static struct obs_kernel_param __setup_##unique_id      \
                __used __section(.init.setup)                   \
                __attribute__((aligned((sizeof(long)))))        \
                = { __setup_str_##unique_id, fn, early }

#define __setup(str, fn)                                        \
        __setup_param(str, fn, fn, 0)

/* NOTE: fn is as per module_param, not __setup!  Emits warning if fn
 * returns non-zero. */
#define early_param(str, fn)                                    \
        __setup_param(str, fn, fn, 1)
```

每个early_param macro就是为了create a struct obs_kernel_param，并把该structure放入“.init.setup” section中。

```
struct obs_kernel_param {
        const char *str;
        int (*setup_func)(char *);
        int early;
};
```

这样所有的early_param macro定义的kernel params构成了一个struct obs_kernel_param array。

该数组在vmlinux.lds中如下表示

```
__setup_start = .; *(.init.setup) __setup_end = .
```

数组的head pointer is __setup_start,tail pointer is __setup_end。

```
void __init parse_early_options(char *cmdline)
{
        parse_args("early options", cmdline, NULL, 0, 0, 0, do_early_param);
}
```

parse_args()会吧cmdline string分割成一个个"param"，然后调用callback do_early_param()处理。

```
static int __init do_early_param(char *param, char *val, const char *unused)
{
        const struct obs_kernel_param *p;

        for (p = __setup_start; p < __setup_end; p++) {
                if ((p->early && parameq(param, p->str)) ||
                    (strcmp(param, "console") == 0 &&
                     strcmp(p->str, "earlycon") == 0)
                ) {
                        if (p->setup_func(val) != 0)
                                pr_warn("Malformed early option '%s'\n", param);
                }
        }
        /* We accept everything at this stage. */
        return 0;
}
```

把cmdline上的某个param与数组中的param name比较，match则调用handler。

Question: When kernel handle early params ?

Answer: start_kernel()

|

|

\|/

setup_arch() / arch/arm/kernel/setup.c

|

|

\|/

parse_early_param()

处理kernel params是非常早的，在memory paging完备以前就处理了。