在u-boot log中有如下信息：

```
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
```

The log is from board/pegmatite/setup.c

```c
/*
 * Construct the boot command for the specified configuration.
 */
static int mmc_get_bootcmd_for_part(int dev_num, int part, disk_partition_t *pinfo, char *cmd, int len, block_dev_desc_t * dev_desc, int boot_recover) {
    /* If the device uses SSP partitioning then it is using CRAM */
    if (dev_desc->part_type == PART_TYPE_SSP) {
        int partition;
        ulong part_size;
        ulong part_sizeB;
        disk_partition_t pinfoB;
        const uint32_t main_addr = 0x100000;
        uint32_t cramfs_addr;
        const char format2[] = "setenv cramfsaddr %#x;"
                               "mmc dev %d;"
                               "mmc read %#x %#lx %#lx;"
                               "cramfsload %#x /main.img;"
                               "source %#x;"
                               "mmc read %#x %#lx %#lx;"
                               "cramfsload %#x /main.img;"
                               "source %#x;"
                               "loop.l 0xd0000000 1";

        if (strcmp((const char *)pinfo->name, "Kernel") != 0) /* The kernel partition is on partition 2 for both EXT2 and CRAM */
            return 1;

        /* Search for recovery partition */
        for (partition = 1; partition < 35; partition++) {
            if (get_partition_info(dev_desc, partition, &pinfoB) == 0) {
                if (strcmp((const char *)pinfoB.name, "RecoveryKernel") == 0 && pinfoB.data_len != 0) {
                    break;
                }
            }
        }

        /* If there is no recovery partition then just make it the same as the primary */
        if (partition == 35) {
            memcpy(&pinfoB, pinfo, sizeof(disk_partition_t));
        }

        /* Boot from the recovery partition */
        if (boot_recover) {
            memcpy(pinfo, &pinfoB, sizeof(*pinfo));
        }

        /* Load the kernel CRAM partition right in front of U-Boot allowing space for the malloc area */
        cramfs_addr = ((gd->start_addr_sp - max(pinfo->data_len, pinfoB.data_len)) >> 20) << 20;
        part_size = (pinfo->data_len + (dev_desc->blksz - 1)) / dev_desc->blksz;
        part_sizeB = (pinfoB.data_len + (dev_desc->blksz - 1)) / dev_desc->blksz;

        sprintf(cmd, format2, cramfs_addr,
                dev_num,
                cramfs_addr, pinfo->start, part_size,
                main_addr,
                main_addr,
                cramfs_addr, pinfoB.start, part_sizeB,
                main_addr,
                main_addr);
    }
    else {


                       ①
        const uint32_t uImage_addr = 0x400000;

           ②
        const uint32_t dtb_addr    = 0xf00000;
        enum board_type board = get_board_type( );
        struct board_info_s board_info;
        char set_eth_addr[50]; /* to hold "setenv ethaddr <eth_address>" */
        char serial_num[50]; /* to hold serial_num=<serial_number> */
        int mmcblk_num = 0;
        const char format[] = "%smmc dev %d;"

           ③
                              "ext2load mmc %d:%d %#x /boot/uImage;"
                              "ext2load mmc %d:%d %#x /boot/%s.dtb;"
                              "setenv bootargs ¥bootargs root=/dev/mmcblk%dp2 uio_pdrv_genirq.of_id=generic-uio rootwait%s;"
                              "bootm %#x - %0#x";

        if (part != 2) /* The kernel partition is on partition 2 for EXT2 */
            return 1;

        set_eth_addr[0] = 0;
        serial_num[0] = 0;

        if (0 == i2c_read_board_info(&board_info)) {
                int i;
                char *c;
                char separator = ':';

                strcpy(set_eth_addr, "setenv ethaddr ");
                c = set_eth_addr + strlen(set_eth_addr);
                for (i=0; i<6; i++, c += 3) {
                        if (5 == i) separator = ';';
                        sprintf(c, "%02x%c", (unsigned char)board_info.eth_mac[i], separator);
                }

                if (board_info.serial_num[0]) {
                        sprintf(serial_num, " serial_num=%s", board_info.serial_n
um);
                }
        }

        if (dev_num != board_configs[board].onboard_dev)
        {
            if ((dev_num < board_configs[board].dc_dev) || (-1 == board_configs[board].onboard_dev))
                mmcblk_num = 0;
            else
                mmcblk_num = 1;
        }

        sprintf(cmd, format, set_eth_addr, dev_num,
                dev_num, part, uImage_addr,
                dev_num, part, dtb_addr, getenv("DTB"),
                mmcblk_num,
                                serial_num,
                uImage_addr, dtb_addr);
    }

    return 0;
}
```

①

SD card启动走的是该分支

②

uImage和dtb的载入address都是hardcode的。感觉好像有点不太好。

③

需要填写待定参数的boot command。

board_late_init()

|

|

```
      \|/

setup_bootcmd()
```