

U-BOOT FOR MagicARM2410

BY thw01 08.02.13

工作环境:

RedHat Linux AS 4

交叉编译器:

Eldk 4.1

U-BOOT 版本:

1.3.1

目标板:

周立功: MagicARM2410 实验箱

感谢openmoko网站及wmmwang的无私奉献!!!

注意文中的红色部分, 为添加或修改或值得注意部分!!!

前奏:

1、建立阳初2410的配置:

(1) 拷贝board/sbc2410x到board/my2410;

(2) 拷贝include/configs/sbc2410x.h到include/configs/my2410.h

2、修改文件:

(1) include/configs/my2410.h:

修改:

```
# define CFG_PROMPT "SBC2410X#"
```

为:

```
# define CFG_PROMPT "my2410#"
```

(2) Makefile:

```
sbc2410x_config : unconfig
```

```
$(MKCONFIG) $(@:_config=) arm arm920t sbc2410x NULL s3c24x0
```

```
my2410_config : unconfig
```

```
$(MKCONFIG) $(@:_config=) arm arm920t my2410 NULL s3c24x0
```

一、使 U-BOOT 能从 NAND FLASH 启动起来

添加的代码主要来自于 VIVI。

1、在 cpu/arm920t/start.s 中修改如下:

```
#include <config.h>
```

```
#include <version.h>
```

```
//my add 1
```

```

#include <s3c2410.h>
//my add end
.....
bl cpu_init_crit
#endif

//my modify 1
#ifdef CONFIG_AT91RM9200
#if defined(CONFIG_AT91RM9200) || defined(CONFIG_S3C2410)
//my modify end

#ifndef CONFIG_SKIP_RELOCATE_UBOOT

//my add 1
#ifndef CONFIG_S3C2410_NAND_BOOT
//my add end

relocate:                /* relocate U-Boot to RAM */
    adr r0, _start        /* r0 ← current position of code */
    ldr r1, _TEXT_BASE    /* test if we run from flash or RAM */
    .....
    stmia r1!, {r3-r10}   /* copy to target address [r1] */
    cmp r0, r2            /* until source end address [r2] */
    ble copy_loop
//my add 1
#else /* NAND_BOOT */
relocate:
copy_myself:
/* mov r10, lr */

@ reset NAND
mov r1, #S3C2410_NAND_BASE
ldr r2, =0xf842 @ initial value enable tacls=3, rph0=6, rph1=0
str r2, [r1, #oNFCNF]
ldr r2, [r1, #oNFCNF]
bic r2, r2, #0x800 @ enable chip
str r2, [r1, #oNFCNF]
mov r2, #0xff @ RESET command
strb r2, [r1, #oNFCMD]
mov r3, #0 @ wait
1: add r3, r3, #0x1
cmp r3, #0xa
blt 1b
2: ldr r2, [r1, #oNFSTAT] @ wait ready

```

```

tst r2, #0x1
beq 2b
ldr r2, [r1, #oNFCNF]
orr r2, r2, #0x800 @ disable chip
str r2, [r1, #oNFCNF]

ldr r0, _TEXT_BASE /* upper 128 KiB: relocated uboot */
sub r0, r0, #CFG_MALLOC_LEN /* malloc area */
sub r0, r0, #CFG_GBL_DATA_SIZE /* bdfinfo */
#ifdef CONFIG_USE_IRQ
sub r0, r0, #(CONFIG_STACKSIZE_IRQ+CONFIG_STACKSIZE_FIQ)
#endif
sub sp, r0, #12 /* leave 3 words for abort-stack */

@ copy u-boot to RAM
ldr r0, _TEXT_BASE
mov r1, #0x0
mov r2, #CFG_UBOOT_SIZE
bl nand_read_ll

tst r0, #0x0
beq ok_nand_read
#ifdef CONFIG_DEBUG_LL
bad_nand_read:
ldr r0, STR_FAIL
ldr r1, SerBase
bl PrintWord
1: b 1b @ infinite loop
#endif

ok_nand_read:
#ifdef CONFIG_DEBUG_LL
ldr r0, STR_OK
ldr r1, SerBase
bl PrintWord
#endif

@ verify
mov r0, #0
@ldr r1, =0x33f00000
ldr r1, _TEXT_BASE
mov r2, #0x400 @ 4 bytes * 1024 = 4K-bytes
go_next:
ldr r3, [r0], #4

```

```

ldr r4, [r1], #4
teq r3, r4
bne notmatch
subs    r2, r2, #4
beq done_nand_read
bne go_next
notmatch:
#ifdef CONFIG_DEBUG_LL
sub r0, r0, #4
ldr r1, SerBase
bl PrintHexWord
ldr r0, STR_FAIL
ldr r1, SerBase
bl PrintWord
#endif
1:  b  1b
done_nand_read:
#endif /* NAND_BOOT */
//my add end
#endif /* CONFIG_SKIP_RELOCATE_UBOOT */
#endif
    /* Set up the stack                */
.....

```

2、 修改 include/configs/my2410.h

```

#ifndef __CONFIG_H
#define __CONFIG_H

//my add 1
#define CONFIG_S3C2410_NAND_BOOT    1
#define CONFIG_S3C2410_NAND_SKIP_BAD    1

#define CFG_UBOOT_SIZE                0x40000
//my add end
/*
 * If we are developing, we might want to start armboot from ram
 * so we MUST NOT initialize critical regs like mem-timing ...
 */
//#undef CONFIG_SKIP_LOWLEVEL_INIT /* undef for developing */
.....
/*
 * Hardware drivers
 */
//my modfiy 1
/*#define CONFIG_DRIVER_CS8900  1*/ /* we have a CS8900 on-board */

```

```

/*#define CS8900_BASE      0x19000300*/
/*#define CS8900_BUS16    1 *//* the Linux driver does accesses as shorts */

#define CONFIG_DRIVER_DM9000 1
#define CONFIG_DM9000_BASE 0x18000300
#define DM9000_IO    CONFIG_DM9000_BASE
#define DM9000_DATA  (CONFIG_DM9000_BASE + 4)
#define CONFIG_DM9000_USE_16BIT 1
//my modfiy end

```

3、 在 board/my2410/中**添加** nand_read.c 文件:

```

/*
 * nand_read.c: Simple NAND read functions for booting from NAND
 *
 * This is used by cpu/arm920/start.S assembler code,
 * and the board-specific linker script must make sure this
 * file is linked within the first 4kB of NAND flash.
 *
 * Taken from GPLv2 licensed vivi bootloader,
 * Copyright (C) 2002 MIZI Research, Inc.
 *
 * Author: Hwang, Chideok <hwang@mizi.com>
 * Date  : $Date: 2004/02/04 10:37:37 $
 *
 * u-boot integration and bad-block skipping (C) 2006 by OpenMoko, Inc.
 * Author: Harald Welte <laforge@openmoko.org>
 */

#include <common.h>

#ifdef CONFIG_S3C2410_NAND_BOOT

#define __REGb(x)    (*(volatile unsigned char *) (x))
#define __REGi(x)    (*(volatile unsigned int *) (x))
#define NF_BASE      0x4e000000
#define NFCONF        __REGi(NF_BASE + 0x0)
#define NFCMD          __REGb(NF_BASE + 0x4)
#define NFADDR         __REGb(NF_BASE + 0x8)
#define NFDATA         __REGb(NF_BASE + 0xc)
#define NFSTAT         __REGb(NF_BASE + 0x10)

#define BUSY 1
inline void wait_idle(void)
{
    int i;

```

```

    while (!(NFSTAT & BUSY))
        for (i=0; i<10; i++);
}

#define NAND_SECTOR_SIZE    512
#define NAND_BLOCK_MASK    (NAND_SECTOR_SIZE - 1)
#define NAND_PAGE_SIZE     0x4000

/* low level nand read function */
int nand_read_ll(unsigned char *buf, unsigned long start_addr, int size)
{
    int i, j;

    if ((start_addr & NAND_BLOCK_MASK) || (size & NAND_BLOCK_MASK))
        return -1; /* invalid alignment */

    /* chip Enable */
    NFCONF &= ~0x800;
    for (i=0; i<10; i++);

    for (i=start_addr; i < (start_addr + size);) {
#ifdef CONFIG_S3C2410_NAND_SKIP_BAD
        if (start_addr % NAND_PAGE_SIZE == 0) {
            unsigned char data;
            NFCMD = 0x50;
            NFADDR = 517&0xf;
            NFADDR = (i >> 9) & 0xff;
            NFADDR = (i >> 17) & 0xff;
            NFADDR = (i >> 25) & 0xff;
            wait_idle();
            data = (NFDATA & 0xff);
            if (data != 0xff) {
                /* Bad block */
                i += NAND_PAGE_SIZE;
                size += NAND_PAGE_SIZE;
                continue;
            }
        }
    }
#endif
    /* READ0 */
    NFCMD = 0;

    /* Write Address */

```

```

        NFADDR = i & 0xff;
        NFADDR = (i >> 9) & 0xff;
        NFADDR = (i >> 17) & 0xff;
        NFADDR = (i >> 25) & 0xff;

        wait_idle();

        for (j=0; j < NAND_SECTOR_SIZE; j++, i++) {
            *buf = (NFDATA & 0xff);
            buf++;
        }
    }

    /* chip Disable */
    NFCONF |= 0x800;    /* chip disable */

    return 0;
}

#endif /* CONFIG_S3C2410_NAND_BOOT */

```

4、 修改 include/s3c2410.h

```

#define S3C2410_ECCSIZE      512
#define S3C2410_ECCBYTES    3

//my modify 1
/*typedef enum {
    S3C24X0_UART0,
    S3C24X0_UART1,
    S3C24X0_UART2
} S3C24X0_UARTS_NR;*/
//my modify end

/* S3C2410 device base addresses */
#define S3C24X0_MEMCTL_BASE    0x48000000
.....
#define S3C2410_ADC_BASE      0x58000000
#define S3C24X0_SPI_BASE     0x59000000
#define S3C2410_SDI_BASE     0x5A000000

//my add 1
#define oNFCONF                0x00
#define oNFCMD                 0x04
#define oNFADDR                0x08
#define oNFDATA                0x0C

```

```

#define oNFSTAT      0x10
#define oNFECCEC    0x14

#ifdef __ASSEMBLER__
//my add end

/* include common stuff */
#include <s3c24x0.h>

//my add 1
typedef enum {
    S3C24X0_UART0,
    S3C24X0_UART1,
    S3C24X0_UART2
} S3C24X0_UARTS_NR;
//my add end

static inline S3C24X0_MEMCTL * S3C24X0_GetBase_MEMCTL(void)
{
    .....
    {
        return (S3C2410_SDI * const)S3C2410_SDI_BASE;
    }
//my add 1
#endif
//my add end

/* ISR */
#define pISR_RESET    (*(unsigned *) (_ISR_STARTADDRESS+0x0))

```

5、 修改 board/my2410/lowlevel_init.S

```

#define B2_Tacp      0x0
#define B2_PMC      0x0

#define B3_Tacs      0x0 /* 0clk */
#define B3_Tcos      0x3 /* 4clk */
#define B3_Tacc      0x7 /* 14clk */
#define B3_Tcoh      0x1 /* 1clk */
#define B3_Tah       0x3 /* 0clk */
#define B3_Tacp      0x3
#define B3_PMC       0x0 /* normal */

#define B4_Tacs      0x0
#define B4_Tcos      0x0

```

6、 修改 board/my2410/sbc2410x.c

```

/* some delay between MPLL and UPLL */
delay (8000);

/* set up the I/O ports *///modify
gpio->GPACON = 0x007FFFFFFF;
gpio->GPBCON = 0x00044555;
gpio->GPBUP = 0x000007FF;
gpio->GPCCON = 0xAAAAAAAA;
gpio->GPCUP = 0x0000FFFF;
.....
gpio->GPEUP = 0x0000FFFF;
gpio->GPFCON = 0x000055AA;
gpio->GPFUP = 0x000000FF;
/*gpio->GPGCON = 0xFF95FFBA;
gpio->GPGUP = 0x0000FFFF;*/
//modify
gpio->GPGCON = 0xFF95FF9A;
gpio->GPGUP = 0x0000FFFB;
//modify
gpio->GPHCON = 0x002AFAAA;
gpio->GPHUP = 0x000007FF;

//my add 1 DM9000 reset ( 2410 GPG2 <--> DM9000 RST)
gpio->GPGDAT = gpio->GPGDAT | ( 1 << 2 );
delay (1);
gpio->GPGDAT = gpio->GPGDAT & ( ~ ( 1 << 2 ) );
delay (1);

gpio->EXTINT0=0x22222222;
gpio->EXTINT1=0x22222222;
gpio->EXTINT2=0x22222222;

```

7、修改 board/my2410/u-boot.lds

```

.....
.text      :
{
    cpu/arm920t/start.o    (.text)
    cpu/arm920t/s3c24x0/nand_read.o (.text)
    *(.text)
}
.....

```

8、修改 drivers/net/dm9000x.c, 因不使用 DM9000 的 MII 接口, 所以去掉:

```

.....
/* Activate DM9000 */
DM9000_iow(DM9000_RCR, RCR_DIS_LONG | RCR_DIS_CRC | RCR_RXEN); /* RX enable */

```

```

DM9000_iow(DM9000_IMR, IMR_PAR);    /* Enable TX/RX interrupt mask */

#if 0
    i = 0;
    while (!(phy_read(1) & 0x20)) { /* autonegation complete bit */
        udelay(1000);
    .....
        break;
    }
    printf("mode\n");
#endif
    return 0;
}

```

9、 修改 `cpu/arm920t/s3c24x0/Makefile`:

```
LIB    = $(obj)lib$(SOC).a
```

```
COBJS = i2c.o interrupts.o serial.o speed.o \
        usb.o usb_ohci.o nand_read.o # nand.o
```

```
SRCS  := $(SOBJS:.o=.S) $(COBJS:.o=.c)
```

```
OBJS  := $(addprefix $(obj),$(SOBJS) $(COBJS))
```

OK, 这边就编译一下, 就可以烧到 NAND FLASH 运行了。

二、 为 U-BOOT 添加 NAND FLASH 驱动，能够使 CONFIG_CMD_NAND 的命令奏效

添加的代码主要来自于 U-BOOT 中其他的板子（board/mp1/vcma9）

1、添加 sbc2410x.h 到 drivers/mtd/nand_legacy/, board/my2410/中，内容如下：

```
/*
 * (C) Copyright 2002, 2003
 * David Mueller, ELSOFT AG, d.mueller@elsoft.ch
 *
 * See file CREDITS for list of people who contributed to this
 * project.
 *
 * This program is free software; you can redistribute it and/or
 * modify it under the terms of the GNU General Public License as
 * published by the Free Software Foundation; either version 2 of
 * the License, or (at your option) any later version.
 *
 * This program is distributed in the hope that it will be useful,
 * but WITHOUT ANY WARRANTY; without even the implied warranty of
 * MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the
 * GNU General Public License for more details.
 *
 * You should have received a copy of the GNU General Public License
 * along with this program; if not, write to the Free Software
 * Foundation, Inc., 59 Temple Place, Suite 330, Boston,
 * MA 02111-1307 USA
 */
/*****
 * Global routines used for VCMA9
 *****/

#include <s3c2410.h>

#if defined(CONFIG_CMD_NAND)
/*typedef enum {
    NFCE_LOW,
    NFCE_HIGH
} NFCE_STATE;*/

static inline void NF_Conf(u16 conf)
```

```

{
    S3C2410_NAND * const nand = S3C2410_GetBase_NAND();

    nand->NFCONF = conf;
}

static inline void NF_Cmd(u8 cmd)
{
    S3C2410_NAND * const nand = S3C2410_GetBase_NAND();

    nand->NFCMD = cmd;
}

static inline void NF_CmdW(u8 cmd)
{
    NF_Cmd(cmd);
    udelay(1);
}

static inline void NF_Addr(u8 addr)
{
    S3C2410_NAND * const nand = S3C2410_GetBase_NAND();

    nand->NFADDR = addr;
}

//static inline void NF_SetCE(NFCE_STATE s)
static inline void NF_SetCE(int s)
{
    S3C2410_NAND * const nand = S3C2410_GetBase_NAND();

    switch (s) {
        case NFCE_LOW:
            nand->NFCONF &= ~(1<<11);
            break;

        case NFCE_HIGH:
            nand->NFCONF |= (1<<11);
            break;
    }
}

static inline void NF_WaitRB(void)
{

```

```

S3C2410_NAND * const nand = S3C2410_GetBase_NAND();

while (!(nand->NFSTAT & (1<<0)));
}

static inline void NF_Write(u8 data)
{
    S3C2410_NAND * const nand = S3C2410_GetBase_NAND();

    nand->NFDATA = data;
}

static inline u8 NF_Read(void)
{
    S3C2410_NAND * const nand = S3C2410_GetBase_NAND();

    return(nand->NFDATA);
}

static inline void NF_Init_ECC(void)
{
    S3C2410_NAND * const nand = S3C2410_GetBase_NAND();

    nand->NFCONF |= (1<<12);
}

static inline u32 NF_Read_ECC(void)
{
    S3C2410_NAND * const nand = S3C2410_GetBase_NAND();

    return(nand->NFECC);
}

```

2、修改 include/configs/my2410.h 如下:

```

.....
#define CONFIG_CMD_ELF
#define CONFIG_CMD_PING

//my add 2
#define CONFIG_CMD_NAND
//my add end
#define CONFIG_BOOTDELAY 3
#define CONFIG_BOOTARGS "console=ttySAC0 root=/dev/nfs
nfsroot=192.168.0.1:/friendly-arm/rootfs_netserver
ip=192.168.0.69:192.168.0.1:192.168.0.1:255.255.255.0:debian:eth0:off"

```

```

.....
 * NAND flash settings
 */
//my add 2
#if defined(CONFIG_CMD_NAND)
#define CFG_NAND_LEGACY 1
#define NFCE_LOW 0
#define NFCE_HIGH 1
#define CFG_NAND_BASE 0x4E00000
//my add end
#define CFG_MAX_NAND_DEVICE 1 /* Max number of NAND devices */
#define SECTORSIZE 512

```

3、修改 include/linux/mtd/nand_ids.h 如下:

加入自己的 Nand Flash 芯片型号, 对如下结构体赋值进行修改:

```

static struct nand_flash_dev nand_flash_ids[] = {
    .....
    {"Samsung K9F1208U0B", NAND_MFR_SAMSUNG, 0x76, 26, 0, 4, 0x4000, 0},
    .....
}

```

4、修改 drivers/mtd/nand_legacy/nand_legacy.c 如下:

```

.....
#include <asm/io.h>
#include <watchdog.h>

//my modify 2
#if defined(CONFIG_CMD_NAND) && defined(CFG_NAND_LEGACY)
#include "sbc2410x.h"
//#endif

//#if defined(CONFIG_CMD_NAND) && defined(CFG_NAND_LEGACY)
//my modify end
#include <linux/mtd/nand_legacy.h>
#include <linux/mtd/nand_ids.h>
.....
#endif
    oob_config.badblock_pos = 5;

//my modify 2
/*for (i=0; i<CFG_MAX_NAND_DEVICE; i++) {
    if (nand_dev_desc[i].ChipID == NAND_ChipID_UNKNOWN) {
        nand = &nand_dev_desc[i];
        break;
    }
}*/

```

```
nand = &nand_dev_desc[0];  
//my modify end
```

```
if (!nand)  
    return (0);
```

5、修改 board/my2410/sbc2410x.c

```
#include <s3c2410.h>  
  
#if defined(CONFIG_CMD_NAND)  
//my modify  
//#include <linux/mtd/nand.h>  
#include "sbc2410x.h"  
//my modify end  
#endif
```

OK, 编译烧写 U-BOOT 到 NAND FLASH, 启动就可以使用 U-BOOT 的读写命令了。

三、 使用 saveenv 时使得环境变量保存在 NAND FLASH 中不丢失

1、因为定义了 CFG_NAND_LEGACY，因此主要修改 saveenv 中对 nand 的读写函数为 nand_legacy 的读写函数，修改 common/env_nand.c 如下：

```
.....
#error CONFIG_INFERN0 not supported yet
#endif

//my add 3
int nand_legacy_erase(struct nand_chip* nand, size_t ofs, size_t len, int clean);
//my add end

int nand_legacy_rw (struct nand_chip* nand, int cmd,
                    size_t start, size_t len,
                    size_t * retlen, u_char * buf);

//my add 3
extern struct nand_chip nand_dev_desc[CFG_MAX_NAND_DEVICE];
//my add end

/* info for NAND chips, defined in drivers/nand/nand.c */
//my modify 3
//extern nand_info_t nand_info[];
nand_info_t nand_info[CFG_MAX_NAND_DEVICE];
//my modify end

/* references to names in env_common.c */
extern uchar default_environment[];
.....
int ret = 0;

puts ("Erasing Nand...");
//my modify 3
//if (nand_erase(&nand_info[0], CFG_ENV_OFFSET, CFG_ENV_SIZE))
if (nand_legacy_erase(nand_dev_desc+0, CFG_ENV_OFFSET, CFG_ENV_SIZE, 0))
//my modify end
    return 1;

puts ("Writing to Nand... ");
total = CFG_ENV_SIZE;
//my modify 3
//ret = nand_write(&nand_info[0], CFG_ENV_OFFSET, &total, (u_char*)env_ptr);
nand_legacy_rw(nand_dev_desc+0, 0x00 | 0x02, CFG_ENV_OFFSET, CFG_ENV_SIZE, &total,
(u_char*)env_ptr);
//my modify end
if (ret || total != CFG_ENV_SIZE)
    return 1;

.....
```

```

int ret;

total = CFG_ENV_SIZE;
//my modify 3
//ret = nand_read(&nand_info[0], CFG_ENV_OFFSET, &total, (u_char*)env_ptr);
ret = nand_legacy_rw(nand_dev_desc+0, 0x01 | 0x02, CFG_ENV_OFFSET, CFG_ENV_SIZE,
&total, (u_char*)env_ptr);
//my modify end
if (ret || total != CFG_ENV_SIZE)
    return use_default();

```

2、修改 include/configs/my2410.h:

```

.....
* NAND flash settings
*/
#if defined(CONFIG_CMD_NAND)

//my add 3
#define CFG_ENV_IS_IN_NAND    1
#define CFG_ENV_SIZE        0x10000 /* Total Size of Environment Sector */
#define CFG_ENV_OFFSET      0x20000
//my add end

#define CFG_NAND_LEGACY    1
#define NFCE_LOW    0
#define NFCE_HIGH    1

```

OK, 编译烧写 U-BOOT 到 NAND FLASH, 就可以使用 SAVEENV 保存环境变量到 NAND FLASH 不丢失了。

在启动后如果有 CRC 错误信息, 解决方法是: 使用 saveenv 命令保存环境变量, 重新启动就不会有 CRC 错误信息了。

附录 (U-BOOT 启动):

启动后:

U-Boot 1.3.1 (Feb 13 2008 - 15:52:31)

DRAM: 64 MB

Flash: 1 MB

NAND: 64 MB

In: serial

Out: serial

Err: serial

Hit any key to stop autoboot: 0

my2410 #

HELP 后:

```
?      - alias for 'help'
askenv - get environment variables from stdin
autoscr - run script from memory
base    - print or set address offset
bdfinfo - print Board Info structure
boot    - boot default, i.e., run 'bootcmd'
bootd   - boot default, i.e., run 'bootcmd'
bootelf - Boot from an ELF image in memory
bootm   - boot application image from memory
bootp   - boot image via network using BootP/TFTP protocol
bootvx  - Boot vxWorks from an ELF image
cmp     - memory compare
coninfo - print console devices and information
cp      - memory copy
crc32   - checksum calculation
date    - get/set/reset date & time
dcache  - enable or disable data cache
dhcp    - invoke DHCP client to obtain IP/boot params
echo    - echo args to console
erase   - erase FLASH memory
exit    - exit script
flinfo  - print FLASH memory information
go      - start application at address 'addr'
help    - print online help
icache  - enable or disable instruction cache
iminfo  - print header information for application image
imls    - list all images found in flash
itest   - return true/false on integer compare
loadb   - load binary file over serial
loads   - load S-Record file over serial line
loady   - load binary file over serial line (ymodem mode)
```

loop - infinite loop on address range
md - memory display
mm - memory modify (auto-incrementing)
mtest - simple RAM test
mw - memory write (fill)
nand - legacy NAND sub-system
nboot - boot from NAND device
nfs - boot image via network using NFS protocol
nm - memory modify (constant address)
ping - send ICMP ECHO_REQUEST to network host
printenv- print environment variables
protect - enable or disable FLASH write protection
rarpboot- boot image via network using RARP/TFTP protocol
reset - Perform RESET of the CPU
run - run commands in an environment variable
saveenv - save environment variables to persistent storage
setenv - set environment variables
sleep - delay execution for some time
test - minimal test like /bin/sh
tftpboot- boot image via network using TFTP protocol
version - print monitor version

saveenv 后:

Saving Environment to NAND...

Erasing Nand...Writing to Nand... done