# U-BOOT-2009 移植配置步骤

----2018 年 9 月

有关于 uboot 的启动流程和步骤，网上有很多资料，这里不再重复，仅记录下 uboot 移植到 freescale mx28 芯片的步骤和要点仅供参考。

1、下载 uboot-imx-rel_imx_2.6.31_10.07.11.tar，地址 http://git.freescale.com/git/cgit.cgi。里面有支持 imx28 的板级支持包，如没有这个包，则需要复制相似的包文件进行修改


2、修改 uboot/board/freescale/mx28_evk/lowlevel_init.S

```
#include <version.h>

//add  config reg as val
.macro REG reg, val
        ldr r2, =\reg
        ldr r3, =\val
        str r3, [r2]
.endm


/* Set up the platform, once the cpu has been initialized */
.globl lowlevel_init
lowlevel_init:
/*config pin AURT0_CTS and RTS as DUART.TX and TX */
/*config pin PWM0 and PWM1 as i2c_sclk and I2C_SDA*/

        REG 0x80018168,0x000000F0
        REG 0x80018164,0x000000A0
        REG 0x80018178,0x0000000F
        REG 0x80018174,0x00000005

        /* All SDRAM settings are done by sdram_prep */
        mov pc, lr
```

3、修改 uboot/board/freescale/mx28_evk/mx28_evk.c

```
//add gpmi pins
static struct pin_desc gpmi_pins_desc[] = {
        { PINID_GPMI_D00, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_D01, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_D02, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_D03, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_D04, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_D05, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_D06, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_D07, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_RDN, PIN_FUN1, PAD_8MA, PAD_1V8, 1 },
        { PINID_GPMI_WRN, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_ALE, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_CLE, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_RDY0, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_RDY1, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_CE0N, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_CE1N, PIN_FUN1, PAD_4MA, PAD_3V3, 0 },
        { PINID_GPMI_RESETN, PIN_FUN1, PAD_4MA, PAD_3V3, 0 }
};
```

```
};
static struct pin_group gpmi_pins = {
        .pins           = gpmi_pins_desc,
        .nr_pins        = ARRAY_SIZE(gpmi_pins_desc)
};
```

```
*/
int board_init(void)
{
        /* Will change it for MX28 EVK later */
        gd->bd->bi_arch_number = MACH_TYPE_MX28EVK;
        /* Adress of boot parameters */
        gd->bd->bi_boot_params = PHYS_SDRAM_1 + 0x100;

//add nand_gpmi
#ifdef CONFIG_NAND_GPMI
        setup_gpmi_nand();
#endif

#ifdef CONFIG_HW_WATCHDOG
    pin_set_type(PINID_GPMI_RDY1, PIN_GPIO);
    pin_gpio_direction(PINID_GPMI_RDY1, 1);
    pin_gpio_set(PINID_GPMI_RDY1, 0);
#endif


        return 0;
}
```

```
            break;
        case 1:
#if 0       //nandflash pins conflict

            /* Set up MMC pins */
            pin_set_group(&mmc1_pins);

            /* Power on the card slot 1 */
            pin_set_type(PINID_PWM4, PIN_GPIO);
            pin_gpio_direction(PINID_PWM4, 1);
            pin_gpio_set(PINID_PWM4, 0);

            /* Wait 10 ms for card ramping up */
            udelay(10000);

            /* Set up SD1 WP pin */
            pin_set_type(PINID_SSP1_GPIO_WP, PIN_GPIO);
            pin_gpio_direction(PINID_SSP1_GPIO_WP, 0);
#endif
            break;
        default:
```

```
void enet_board_init(void)
{
        /* Set up ENET pins */
        pin_set_group(&enet_pins);

//modify phy pins  PINID_ENET0_RX_CLK------>PINID_LCD_D16

        /* Power on the external phy
        pin_set_type(PINID_SSP1_DATA3, PIN_GPIO);
        pin_gpio_direction(PINID_SSP1_DATA3, 1);
        pin_gpio_set(PINID_SSP1_DATA3, 0);
*/
        /* Reset the external phy */
        pin_set_type(PINID_LCD_D16, PIN_GPIO);
        pin_gpio_direction(PINID_LCD_D16, 1);
        pin_gpio_set(PINID_LCD_D16, 0);
        udelay(200);
        pin_gpio_set(PINID_LCD_D16, 1);
}
```

```
// add nand gpmi  and watchdog
#ifdef CONFIG_NAND_GPMI
void setup_gpmi_nand()
{
        /* Set up GPMI pins */
        pin_set_group(&gpmi_pins);
}


 #ifdef CONFIG_HW_WATCHDOG
    int bValue = 0;
    void hw_watchdog_reset(){
        if(bValue==1) {
            bValue=0;
            pin_gpio_set(PINID_GPMI_RDY1, 1);
        }else{
            bValue=1;
            pin_gpio_set(PINID_GPMI_RDY1, 0);
        }
    }
 #endif
#endif
```

4、修改 uboot/cpu/arm926ejs/mx28/generic.c

```
//add gpmi clk
static inline void __enable_gpmi_clk(void)
{
        /* Clear bypass bit*/
        REG_SET(REGS_CLKCTRL_BASE, HW_CLKCTRL_CLKSEQ,
                BM_CLKCTRL_CLKSEQ_BYPASS_GPMI);
        /* Set gpmi clock to ref_gpmi/12 */
        REG_WR(REGS_CLKCTRL_BASE, HW_CLKCTRL_GPMI,
              REG_RD(REGS_CLKCTRL_BASE, HW_CLKCTRL_GPMI) &
              (~(BM_CLKCTRL_GPMI_DIV)) &
              (~(BM_CLKCTRL_GPMI_CLKGATE)) |
              1);
}
static u32 mx28_get_gpmiclk(void)
{
        const u32 xtal = 24, ref = 480;
        u32 clkfrac, clkseq, clkctrl;
        u32 frac, div;
        u32 gpmiclk;
        /* Enable gpmi clock */
        __enable_gpmi_clk();

        clkfrac = REG_RD(REGS_CLKCTRL_BASE, HW_CLKCTRL_FRAC1);
```

```
        clkfrac = REG_RD(REGS_CLKCTRL_BASE, HW_CLKCTRL_FRAC1);
        clkseq = REG_RD(REGS_CLKCTRL_BASE, HW_CLKCTRL_CLKSEQ);
        clkctrl = REG_RD(REGS_CLKCTRL_BASE, HW_CLKCTRL_GPMI);

        if (clkseq & BM_CLKCTRL_CLKSEQ_BYPASS_GPMI) {
                /* xtal path */
                div = (clkctrl & BM_CLKCTRL_GPMI_DIV) >>
                        BP_CLKCTRL_GPMI_DIV;
                gpmiclk = xtal / div;
        } else {
                /* ref path */
                frac = (clkfrac & BM_CLKCTRL_FRAC1_GPMIFRAC) >>
                        BP_CLKCTRL_FRAC1_GPMIFRAC;
                div = (clkctrl & BM_CLKCTRL_GPMI_DIV) >>
                        BP_CLKCTRL_GPMI_DIV;
                gpmiclk =  (ref * 18 / frac) / div;
        }

        return gpmiclk;
}
```

```
u32 mxc_get_clock(enum mxc_clock clk)
{
        switch (clk) {
        case MXC_ARM_CLK:
                return mx28_get_pclk() * 1000000;
//add
        case MXC_GPMI_CLK:
                return mx28_get_gpmiclk() * 1000000;
        case MXC_AHB_CLK:
```

5、修改 uboot/cpu/arm926ejs/mx28/pinctrl.c

```
}
//add
void watchdog_feed(void)
{
        static int value = 1;

        pin_set_type(PINID_GPMI_RDY1, PIN_GPIO);
        pin_gpio_direction(PINID_GPMI_RDY1, 1);
        if(value == 0)
                value = 1;
        else
                value = 0;

        pin_gpio_set(PINID_GPMI_RDY1, value);

}
```

6、修改 uboot/cpu/arm926ejs/mx28/serial.c

```
void serial_puts(const char *s)
{
        while (*s)
        {
        //add watchdog feed
                watchdog_feed();
                serial_putc(*s++);
                watchdog_feed();
        }
}
```

```
}

/* Receive character */
int serial_getc(void)
{
//add  count tick
//watchdog feed
        int tick = 1000;
        /* Wait while TX FIFO is empty */
        while (REG_RD(REGS_UARTDBG_BASE, HW_UARTDBGFR) & BM_UARTDBGFR_RXFE)
        {
                if((tick--) == 0)
                {
                        watchdog_feed();
                        tick = 1000;
                }
        }
        watchdog_feed();

        /* Read data byte */
        return REG_RD(REGS_UARTDBG_BASE, HW_UARTDBGDR) & 0xff;
```

7、修改 uboot/include/configs/mx28_evk.h

```
//add
#define CONFIG_HW_WATCHDOG  //define watchdog
#define BOARD_LATE_INIT         1

/*
 * Boot Linux
 */
#define CONFIG_CMDLINE_TAG
#define CONFIG_SETUP_MEMORY_TAGS
#define CONFIG_BOOTDELAY    3           //modify
#define CONFIG_BOOTFILE     "uImage"
//#define CONFIG_BOOTARGS       "console=ttyAM0,115200n8 "
#define CONFIG_BOOTCOMMAND  "run nand_boot"
#define CONFIG_LOADADDR     0x42000000
#define CONFIG_SYS_LOAD_ADDR    CONFIG_LOADADDR

#define UBOOT_IMAGE_SIZE        0x80000 //add uboot size

//modify MTD partition
#define MTDIDS_DEFAULT  "nand0=nandflash0"
#define MTDPARTS_DEFAULT    "mtdparts=nandflash0:12m(boot),"\
```

```
//modify MTD partition
#define MTDIDS_DEFAULT  "nand0=nandflash0"
#define MTDPARTS_DEFAULT    "mtdparts=nandflash0:12m(boot),"\
                                        "512k(env),"\
                                        "512k(reserve),"\
                                        "2m(null)"\
                                        "512k(reserve),"\
                                        "64m(rootfs),"\
                                        "-(opt)"

//add MTD config
#define CONFIG_MTD_DEVICE       1
#define CONFIG_MTD_PARTITIONS   1
#define CONFIG_CMD_MTDPARTS     1
#define CONFIG_CMD_UBIFS
#define CONFIG_CMD_UBI
#define CONFIG_LZO              1
#define CONFIG_RBTREE           1
```

```
//modify environment
#define CONFIG_EXTRA_ENV_SETTINGS \
        "kernel=uImage\0"
\
        "kernelsize=0x400000\0"
\
        "rootfs=rootfs.ubifs\0"
\
        "showbitmap=0\0"
\
        "kerneladdr="       "0x00200000\0"
\
        "kerneladdr2="      "0x00780000\0"
\
        "gatewayip="        "192.168.1.1\0"
\
    "ds2460_mac=00:04:00:00:00:00\0"
\
    "nfsroot=/nfsroot/rootfs\0"
\
    "bootargs=gpmi=g console=ttyAM0,115200n8 ubi.mtd=5 root=ubi0:rootfs root
fstype=ubifs ro\0"                                          \
```

```
    "bootargs=gpmi=g console=ttyAM0,115200n8 ubi.mtd=5 root=ubi0:rootfs root
fstype=ubifs ro\0"                                                  \
    "bootargs_nand=setenv bootargs gpmi=g console=ttyAM0,115200n8 ubi.mtd=5
root=ubi0:rootfs rootfstype=ubifs ro\0"                             \
    "upuboot=tftp $(loadaddr) $(serverip):imx28_ivt_uboot.sb;nand erase 0x0
0x100000; nand write $(loadaddr) 0x0 0x100000\0"    \
    "upkernel="         "tftp $(loadaddr) $(serverip):$(kernel);"
                    \
                        "nand erase  $(kerneladdr) $(kernelsize);"
                    \
                        "nand write $(loadaddr) $(kerneladdr) $(filesize
);\0"                   \
        "uprootfs="         "mtdparts default;"
                    \
                            "nand erase rootfs;"
                    \
                            "ubi part rootfs;"                  \
                            "ubi create  rootfs;"
                    \
                            "tftp $(loadaddr) $(rootfs);"
                    \
                            "ubi write $(loadaddr) rootfs $(filesize)\0"
                    \
```

```
                        "ubi write $(loadaddr) rootfs $(filesize)\0"
                    \
    "tftp_boot=tftp $(loadaddr) $(serverip):uImage; bootm;\0"
                    \
    "nand_boot=nand read.jffs2 $(loadaddr) $(kernelladdr) $(kernelsize);"
                    \
     "bootm $(loadaddr);"                   \
     "nand read.jffs2 $(loadaddr) $(kerneladdr2) $(kernelsize);"     \
      "bootm $(loadaddr)\0"                 \
```

```
/*
 * FEC Driver
 */
#define CONFIG_MXC_FEC
#define CONFIG_GET_FEC_MAC_ADDR_FROM_IIM    //add
#define CONFIG_FEC0_IOBASE      REGS_ENET_BASE
#define CONFIG_FEC0_PHY_ADDR        5       //modify
#define CONFIG_NET_MULTI
#define CONFIG_ETH_PRIME
#define CONFIG_RMII
#define CONFIG_CMD_MII
#define CONFIG_CMD_DHCP
#define CONFIG_CMD_PING
```

```
//add GPMI NAND CONFIG
#define CONFIG_CMD_NAND
#define CONFIG_NAND_GPMI
#define CONFIG_GPMI_NFC_SWAP_BLOCK_MARK
#define CONFIG_GPMI_NFC_V1
#define CONFIG_GPMI_REG_BASE    GPMI_BASE_ADDR
#define CONFIG_BCH_REG_BASE BCH_BASE_ADDR
#define NAND_MAX_CHIPS      8
#define CONFIG_SYS_NAND_BASE        0x40000000
#define CONFIG_SYS_MAX_NAND_DEVICE  1

//add APBH DMA CONFIG
#define CONFIG_APBH_DMA
#define CONFIG_APBH_DMA_V1
#define CONFIG_MXS_DMA_REG_BASE ABPHDMA_BASE_ADDR

/*
 * Environments on NAND
```

```
/*
 * Environments on NAND
 */
#define CONFIG_CMD_ENV
#define CONFIG_ENV_OVERWRITE

#define CONFIG_ENV_IS_IN_NAND   1
#define CONFIG_ENV_OFFSET       0xc00000 /* nand env offset 12M  */
#define CONFIG_ENV_SIZE         (128*1024) /* 128 KB */
#define CONFIG_ENV_SECT_SIZE    (128*1024)

/* The global boot mode has been detected by Boot ROM and a boot mode
 * is stored at address of 0x0001a7f0.
 */
#define GLOBAL_BOOT_MODE_ADDR 0x0001a7f0
#define BOOT_MODE_SD0 0x9
#define BOOT_MODE_SD1 0xa

#define CONFIG_HW_WATCHDOG //add watchdog

#endif /* __MX28_EVK_H */
-- INSERT --                                                    215,42
```

8、修改 uboot/include/asm-arm/

```
#ifndef __ASSEMBLER__
enum mxc_clock {
    MXC_ARM_CLK = 0,
    MXC_AHB_CLK,
    MXC_IPG_CLK,
    MXC_GPMI_CLK,   //add
};
```

9、修改 uboot/drivers/mtd/nand/nand_base.c

```
    realpage = (int)(from >> chip->page_shift);
    page = realpage & chip->pagemask;

    col = (int)(from & (mtd->writesize - 1));

    buf = ops->datbuf;
    oob = ops->oobbuf;

    while(1) {
        watchdog_feed(); //add
        bytes = min(mtd->writesize - col, readlen);
        aligned = (bytes == mtd->writesize);
```

增加看门狗喂狗操作，可以搜索关键词定位进行修改

```
        steps = 0;
    } else
        pos = eccsize;

    chip->cmdfunc(mtd, NAND_CMD_SEQIN, pos, page);
    for (i = 0; i < steps; i++) {
        watchdog_feed();        //add
        if (sndcmd) {
            if (mtd->writesize <= 512) {
                uint32_t fill = 0xFFFFFFFF;

                len = eccsize;
                while (len > 0) {
                    int num = min_t(int, len, 4);
                    chip->write_buf(mtd, (uint8_t *)&fill,
```

```
    /* Shift to get page */
    realpage = (int)(from >> chip->page_shift);
    page = realpage & chip->pagemask;

    while(1) {
        watchdog_feed();    //add
        sndcmd = chip->ecc.read_oob(mtd, chip, page, sndcmd);
```

```
        memset(chip->oob_poi, 0xff, mtd->oobsize);

    while(1) {
        int bytes = mtd->writesize;
        int cached = writelen > bytes && page != blockmask;
        uint8_t *wbuf = buf;

        watchdog_feed();// add

        /* Partial page write ? */
```

在 static struct nand_flash_dev *nand_get_flash_type(struct mtd_info *mtd, struct nand_chip *chip, int busw, int *maf_id)中修改

```
        ffs(mtd->erasesize) - 1;
//modify
    if(chip->chipsize &0xffffffff)
        chip->chip_shift = ffs(chip->chipsize) - 1;
    else
        chip->chip_shift = ff((unsigned)(chip->chipsize >> 32)) + 31;
    /* Set the bad block position */
    chip->badblockpos = mtd->writesize > 512 ?
```

10、修改 uboot/drivers/mtd/nand/nand_device_info.c

```
//add drivers S34ML01G2
{
.end_of_table              = false,
.manufacturer_code         = 0x01,
.device_code               = 0xf1,
.cell_technology           = NAND_DEVICE_CELL_TECH_SLC,
.chip_size_in_bytes        = 128LL*SZ_1M,
.block_size_in_pages       = 64,
.page_total_size_in_bytes  = 2*SZ_1K + 64,
.ecc_strength_in_bits      = 4,
.ecc_size_in_bytes         = 512,
.data_setup_in_ns          = 20,
.data_hold_in_ns           = 10,
.address_setup_in_ns       = 20,
.gpmi_sample_delay_in_ns   = 6,
.tREA_in_ns                = -1,
.tRLOH_in_ns               = -1,
.tRHOH_in_ns               = -1,
"S34ML01G2",
},
                                                          43,9
```

```
    },
//modify  drivers MX30LF1G08
    {
    .end_of_table              = false,
    .manufacturer_code         = 0xc2,
    .device_code               = 0xf1,
    .cell_technology           = NAND_DEVICE_CELL_TECH_SLC,
    .chip_size_in_bytes        = 128LL*SZ_1M,
    .block_size_in_pages       = 64,
    .page_total_size_in_bytes  = 2*SZ_1K + 64,
    .ecc_strength_in_bits      = 4,
    .ecc_size_in_bytes         = 512,
    .data_setup_in_ns          = 20,
    .data_hold_in_ns           = 10,
    .address_setup_in_ns       = 20,
    .gpmi_sample_delay_in_ns   = 6,
    .tREA_in_ns                = -1,
    .tRLOH_in_ns               = -1,
    .tRHOH_in_ns               = -1,
    "MX30LF1G08",
    },
//modify drivers MX30LF2
```

```
//modify drivers MX30LF2
    {
    .end_of_table              = false,
    .manufacturer_code         = 0xc2,
    .device_code               = 0xda,
    .cell_technology           = NAND_DEVICE_CELL_TECH_SLC,
    .chip_size_in_bytes        = 256LL*SZ_1M,
    .block_size_in_pages       = 64,
    .page_total_size_in_bytes  = 2*SZ_1K + 64,
    .ecc_strength_in_bits      = 4,
    .ecc_size_in_bytes         = 512,
    .data_setup_in_ns          = 20,
    .data_hold_in_ns           = 10,
    .address_setup_in_ns       = 20,
    .gpmi_sample_delay_in_ns   = 6,
    .tREA_in_ns                = -1,
    .tRLOH_in_ns               = -1,
    .tRHOH_in_ns               = -1,
    "MX30LF2",
    },
    {
```

```
//add
static struct nand_device_info *nand_device_info_fn_spansion(const uint8_t
id[]){
    //struct nand_device_info  *table;

    printf("nand_device_info_fn_spansion called.\n");

    /* Check for an SLC device. */
    if (ID_GET_CELL_TYPE_CODE(id) == ID_CELL_TYPE_CODE_SLC) {
        printf("nand slc\n");
        /* Type 2 */
        return nand_device_info_search(nand_device_info_table_type_2,
                        ID_GET_MFR_CODE(id), ID_GET_DEVICE_CODE(id));
    }
}
```

```
//add
static struct nand_device_info *nand_device_info_fn_mx(const uint8_t id[])
{
    //struct nand_device_info  *table;

    printf("nand_device_info_fn_mx called.\n");
    /* Check for an SLC device. */

    if (ID_GET_CELL_TYPE_CODE(id) == ID_CELL_TYPE_CODE_SLC) {

        printf("nand slc\n");
        /* Type 2 */
        return nand_device_info_search(nand_device_info_table_type_2,
                ID_GET_MFR_CODE(id), ID_GET_DEVICE_CODE(id));
    }
}
```

```
static struct nand_device_mfr_info  nand_device_mfr_directory[] =
{
    //add
    {
    .id = NAND_MFR_SPANSION,
    .fn = nand_device_info_fn_spansion,
    },

    {
    .id = NAND_MFR_TOSHIBA,
    .fn = nand_device_info_fn_toshiba,
    },
    {
    .id = NAND_MFR_SAMSUNG,
    .fn = nand_device_info_fn_samsung,
    },
    //add
    {
    .id = NAND_MFR_MX30LF1G08,
    .fn = nand_device_info_fn_mx,
    },
    {
```

11、修改 uboot/drivers/mtd/nand/nand_ids.c

```
    /* 16 Gigabit */
    {"NAND 2GiB 1,8V 8-bit",    0xA5, 0, 2048, 0, LP_OPTIONS},
    {"NAND 2GiB 3,3V 8-bit",    0xD5, 0, 2048, 0, LP_OPTIONS},
    {"NAND 2GiB 1,8V 16-bit",   0xB5, 0, 2048, 0, LP_OPTIONS16},
    {"NAND 2GiB 3,3V 16-bit",   0xC5, 0, 2048, 0, LP_OPTIONS16},
//add
    {"NAND 2GiB 3,3V 8-bit",        0x48, 0, 2048, 0, LP_OPTIONS},

     /* 32 Gigabit ,only use 2G due to the linux mtd limitation*/
    {"NAND 4GiB 3,3V 8-bit",        0xD7, 0, 2048, 0, LP_OPTIONS},
```

```
/*
*   Manufacturer ID list
*/
struct nand_manufacturers nand_manuf_ids[] = {
//add
    {NAND_MFR_SPANSION, "Spansion"},
    {NAND_MFR_TOSHIBA, "Toshiba"},
    {NAND_MFR_SAMSUNG, "Samsung"},
    {NAND_MFR_FUJITSU, "Fujitsu"},
```

12、修改 uboot/drivers/mtd/nand/nand_util.c

```
    }

    for (;
         erase.addr < opts->offset + erase_length;
         erase.addr += meminfo->erasesize) {

        watchdog_feed();
//      WATCHDOG_RESET ();

        if (!opts->scrub && bbtest) {
            int ret = meminfo->block_isbad(meminfo, erase.addr
```

```
    if (!opts->quiet)
        printf("\n");

    if (nand_block_bad_old) {
        struct nand_chip *priv_nand = meminfo->priv;

        priv_nand->block_bad = nand_block_bad_old;
//      priv_nand->scan_bbt(meminfo);
    }

    return 0;
```

```
 */
int nand_write_skip_bad(nand_info_t *nand, loff_t offset, size_t *
            u_char *buffer)
{
    int rval;
//modify
    size_t left_to_write = (*length+2047)/(2048)*(2048);
    size_t len_incl_bad;
    u_char *p_buffer = buffer;
```

把所有的 WATCHDOG_RESET()替换成 watchdog_feed(),搜索 watchdog_reset 进行查找替换

```
        watchdog_feed();

//      WATCHDOG_RESET ();
```

13、修改 uboot/include/linux/mtd/nand.h

```
 * NAND Flash Manufacturer ID Codes
 */
//add SPANSION
#define NAND_MFR_SPANSION    0X01
#define NAND_MFR_TOSHIBA     0x98
#define NAND_MFR_SAMSUNG     0xec
#define NAND_MFR_FUJITSU     0x04
#define NAND_MFR_NATIONAL    0x8f
#define NAND_MFR_RENESAS     0x07
#define NAND_MFR_STMICRO     0x20
#define NAND_MFR_HYNIX       0xad
#define NAND_MFR_MICRON      0x2c
#define NAND_MFR_AMD         0x01
#define NAND_MFR_SANDISK     0x45
#define NAND_MFR_INTEL       0x89
//add MX30LF1G08
#define NAND_MFR_MX30LF1G08 0xc2
```

14、编写脚本 vim build,再 chmod 777 build

```
#!/bin/bash

export ARCH=arm
export CROSS_COMPILE=/usr/local/arm-fsl-linux-gnueabi/bin/arm-fsl-linux-gnue
abi-
make distclean
make mx28_evk_config
make

```

15、./build 执行

16、出现错误

Undefined reference to “board_nand_init”

发现是缺少 board_nand_init(),mx28 没有这个函数，取了个巧，从周立功官方源码找到了定义的这个函数在 gpmi_nfc_mil.c 里面。添加相关 gpmi 管理 nandflash 的代码：gpmi_nfc_mil.c 、 gpmi_nfc_bch.h 、 gpmi_nfc_gpmi.h 、 gpmi_nfc_hal.c 、 gpmi_nfc_mil.c.orig 到 uboot/drivers/mtd/nand 里面

```
djh@ubuntu:~/linux/uboot-2.6.31$ cd drivers/mtd/nand/
djh@ubuntu:~/linux/uboot-2.6.31/drivers/mtd/nand$ ls
atmel_nand.c      gpmi_nfc_mil.c       nand_base.o          nand_ids.o
bfin_nand.c       gpmi_nfc_mil.c.orig  nand_bbt.c           nand.o
davinci_nand.c    gpmi_nfc_mil.o       nand_bbt.o           nand_plat.c
diskonchip.c      kirkwood_nand.c      nand.c               nand_util.c
fsl_elbc_nand.c   libnand.a            nand_device_info.c   nand_util.o
fsl_upm.c         Makefile             nand_device_info.h   ndfc.c
gpmi_nfc_bch.h    mpc5121_nfc.c        nand_device_info.o   nomadik.c
gpmi_nfc_gpmi.h   mx31_nand.c          nand_ecc.c           omap_gpmc.c
gpmi_nfc_hal.c    mxc_nand.c           nand_ecc.o           s3c2410_nand.c
gpmi_nfc_hal.o    nand_base.c          nand_ids.c           s3c64xx.c
djh@ubuntu:~/linux/uboot-2.6.31/drivers/mtd/nand$ vim Makefile
```

修改 Makefile

```
COBJS-$(CONFIG_NAND_ATMEL) += atmel_nand.o
COBJS-$(CONFIG_DRIVER_NAND_BFIN) += bfin_nand.o
COBJS-$(CONFIG_NAND_DAVINCI) += davinci_nand.o
COBJS-$(CONFIG_NAND_FSL_ELBC) += fsl_elbc_nand.o
COBJS-$(CONFIG_NAND_FSL_UPM) += fsl_upm.o
COBJS-$(CONFIG_NAND_KIRKWOOD) += kirkwood_nand.o
COBJS-$(CONFIG_NAND_MPC5121_NFC) += mpc5121_nfc.o
COBJS-$(CONFIG_NAND_NDFC) += ndfc.o
COBJS-$(CONFIG_NAND_NOMADIK) += nomadik.o
COBJS-$(CONFIG_NAND_S3C2410) += s3c2410_nand.o
COBJS-$(CONFIG_NAND_S3C64XX) += s3c64xx.o
COBJS-$(CONFIG_NAND_OMAP_GPMC) += omap_gpmc.o
COBJS-$(CONFIG_NAND_PLAT) += nand_plat.o
COBJS-$(CONFIG_MX31_NAND) += mx31_nand.o
COBJS-$(CONFIG_MXC_NAND) += mxc_nand.o nand_device_info.o
COBJS-$(CONFIG_NAND_GPMI) += gpmi_nfc_hal.o gpmi_nfc_mil.o nand_device_info.
o
endif
                                                          52,1          62%
```

具体的代码太多了，不一一截图，统一把源码放到附录里

17、再添加 dma 操作函数到 drivers/dma/里。.h 文件添加

```
djh@ubuntu:~/linux/bootloader/u-boot-2.6.08/drivers$ cd dma/
djh@ubuntu:~/linux/bootloader/u-boot-2.6.08/drivers/dma$ ls
apbh_dma.c  fsl_dma.c  Makefile      MCD_tasks.c
apbh_dma.o  libdma.a   MCD_dmaApi.c  MCD_tasksInit.c
djh@ubuntu:~/linux/bootloader/u-boot-2.6.08/drivers/dma$
```

```
djh@ubuntu:~/linux/uboot-2.6.31$ cd include/asm-arm/
djh@ubuntu:~/linux/uboot-2.6.31/include/asm-arm$ ls
apbh_dma.h          arch-mx23       arch-pxa        dma-mapping.h  proc-armv
arch                arch-mx25       arch-s3c24x0    errno.h        processor.h
arch-arm720t        arch-mx27       arch-s3c44b0    fec.h          ptrace.h
arch-arm925t        arch-mx28       arch-s3c4510b   global_data.h  setup.h
arch-arm926ejs      arch-mx31       arch-s3c64xx    hardware.h     sizes.h
arch-at91           arch-mx35       arch-sa1100     imx_iim.h      string.h
arch-at91rm9200     arch-mx50       atomic.h        io.h           system.h
arch-davinci        arch-mx51       bitops.h        mach-types.h   types.h
arch-imx            arch-mx53       byteorder.h     macro.h        u-boot-arm.h
arch-ixp            arch-nomadik    cache-cp15.h    memory.h       u-boot.h
arch-kirkwood       arch-omap       cache.h         mmu.h          unaligned.h
arch-ks8695         arch-omap24xx   clock.h         posix_types.h
arch-lpc2292        arch-omap3      config.h        proc
djh@ubuntu:~/linux/uboot-2.6.31/include/asm-arm$
```

修改 makefile

```
#

include $(TOPDIR)/config.mk

LIB := $(obj)libdma.a

COBJS-$(CONFIG_FSLDMAFEC) += MCD_tasksInit.o MCD_dmaApi.o MCD_tasks.o
COBJS-$(CONFIG_FSL_DMA) += fsl_dma.o
COBJS-$(CONFIG_APBH_DMA) += apbh_dma.o

COBJS   := $(COBJS-y)
SRCS    := $(COBJS:.o=.c)
OBJS    := $(addprefix $(obj),$(COBJS))

```

## 18、再次执行./build

```
../lib/gcc/arm-fsl-linux-gnueabi/4.4.4/armv5te -lgcc -Map u-boot.map -o u-boot
/usr/local/arm-fsl-linux-gnueabi/bin/arm-fsl-linux-gnueabi-objcopy -O srec u-boo
t u-boot.srec
/usr/local/arm-fsl-linux-gnueabi/bin/arm-fsl-linux-gnueabi-objcopy --gap-fill=0x
ff -O binary u-boot u-boot.bin
djh@ubuntu:~/linux/uboot-2.6.31$ ls
```

```
djh@ubuntu:~/linux/uboot-2.6.31$ ls
api                          drivers         lib_mips     onenand_ipl
board                        examples        lib_nios     post
build                        fs              lib_nios2    README
CHANGELOG                    include         lib_ppc      rules.mk
CHANGELOG-before-U-Boot-1.1.5 lib_arm        lib_sh       System.map
common                       lib_avr32       lib_sparc    tags
config.mk                    lib_blackfin    MAINTAINERS  tools
COPYING                      libfdt          MAKEALL      u-boot
cpu                          lib_generic     Makefile     u-boot.bin
CREDITS                      lib_i386        mkconfig     u-boot.lds
disk                         lib_m68k        nand_spl     u-boot.map
doc                          lib_microblaze  net          u-boot.srec
djh@ubuntu:~/linux/uboot-2.6.31$
```

已经生成 uboot 的所有文件了，在这里使用 u-boot 文件，飞思卡尔芯片的 uboot 和其他芯片不一样，只能用 SD 卡以及烧写工具 mfgtool。具体参考《EasyARM-iMX28xx Linux 开发指南 20150901 V1.03》