基于 AM335x SDK6.0 去掉 EEPROM 验证 ID 功能

以 startkit 为例，默认执行 startkit 配置。

I. Uboot

Board.c 文件

- S_init 函数，

屏蔽 EEPROM 读写部分以及 header 校验部分

```
#if 0                                                   //EEPROM
        /* Check if baseboard eeprom is available */
        if (i2c_probe(CONFIG_SYS_I2C_EEPROM_ADDR)) {
                puts("Could not probe the EEPROM; something fundamentally "
                        "wrong on the I2C bus.\n");
        }

        /* read the eeprom using i2c */
        if (i2c_read(CONFIG_SYS_I2C_EEPROM_ADDR, 0, 2, (uchar *)&header,
                                                        sizeof(header))) {
                puts("Could not read the EEPROM; something fundamentally"
                        " wrong on the I2C bus.\n");
        }

        if (header.magic != 0xEE3355AA) {
                /*
                 * read the eeprom using i2c again,
                 * but use only a 1 byte address
                 */
                if (i2c_read(CONFIG_SYS_I2C_EEPROM_ADDR, 0, 1,
                                (uchar *)&header, sizeof(header))) {
                        puts("Could not read the EEPROM; something "
                                "fundamentally wrong on the I2C bus.\n");
                        hang();
                }

                if (header.magic != 0xEE3355AA) {
                        printf("Incorrect magic number (0x%x) in EEPROM\n",
                                        header.magic);
                        hang();
                }
        }
#endif

        enable_board_pin_mux(&header);
```

修改 enable_board_pin_mux 函数。因为执行完上面步骤后，函数输入的行参为 null 或者默认值。函数定义在 board/ti/am335x/mux.c 中

```
enable_board_pin_mux(&header);
```

只保留以下 starterkit 相关部分，其他全部删除或者屏蔽掉。

```
        /* Starter Kit EVM */
                configure_module_pin_mux(i2c1_pin_mux);
                configure_module_pin_mux(gpio0_7_pin_mux);
                configure_module_pin_mux(rgmii1_pin_mux);
                configure_module_pin_mux(mmc0_pin_mux_sk_evm);
```

接下来去掉判断部分，只保留对 VTT 电源操作的代码（如果客户板子没有用 GPIO 控制 VTT 电源，这部分代码也可以相应删除）

```
        enable_board_pin_mux(&header);
//      if (!strncmp("A335X_SK", header.name, HDR_NAME_LEN)) {    //EEPROM
                /*
                 * EVM SK 1.2A and later use gpio0_7 to enable DDR3.
                 * This is safe enough to do on older revs.
                 */
                gpio_request(GPIO_DDR_VTT_EN, "ddr_vtt_en");
                gpio_direction_output(GPIO_DDR_VTT_EN, 1);
//      }                                                           //EEPRO
M
```

接下来对 DDR 初始化的代码只保留对 starterkit 上 DDR3 的初始化代码，其他全部屏蔽或者删除（如果客户板子采用的 DDR3 型号与 Starterkit 上不一致，请更改 DDR3 参数）

```
//      if (!strncmp("A335X_SK", header.name, HDR_NAME_LEN))           //EEPRO
M
                config_ddr(303, MT41J128MJT125_IOCTRL_VALUE, &ddr3_data,
                                &ddr3_cmd_ctrl_data, &ddr3_emif_reg_data);
#if 0
        else if  (!strncmp("A335BNLT", header.name, 8))
                config_ddr(400, MT41K256M16HA125E_IOCTRL_VALUE,
                                &ddr3_beagleblack_data,
                                &ddr3_beagleblack_cmd_ctrl_data,
                                &ddr3_beagleblack_emif_reg_data);
        else if (!strncmp("A33515BB", header.name, 8) &&
                                strncmp("1.5", header.version, 3) <= 0)
                config_ddr(303, MT41J512M8RH125_IOCTRL_VALUE, &ddr3_evm_data,
                                &ddr3_evm_cmd_ctrl_data, &ddr3_evm_emif_reg_data);
        else
                config_ddr(266, MT47H128M16RT25E_IOCTRL_VALUE, &ddr2_data,
                                &ddr2_cmd_ctrl_data, &ddr2_emif_reg_data);
#endif
#endif
}
```

II. Kernel

board-am335xevm.c

- am335x_evm_setup 函数

  需要做以下修改，

  如果不需要 CPSW，也可以将 am33xx_cpsw_macidfillup 删掉。

```
static void am335x_evm_setup(struct memory_accessor *mem_acc, void *context)
{
        int ret;
        char tmp[10];

        /* 1st get the MAC address from EEPROM */
        ret = mem_acc->read(mem_acc, (char *)&am335x_mac_addr,
                EEPROM_MAC_ADDRESS_OFFSET, sizeof(am335x_mac_addr));
/*
        if (ret != sizeof(am335x_mac_addr)) {
                pr_warning("AM335X: EVM Config read fail: %d\n", ret);
                return;
        }
*/                                      // EEPROM

        /* Fillup global mac id */
        am33xx_cpsw_macidfillup(&am335x_mac_addr[0][0],
                                &am335x_mac_addr[1][0]);

        /* get board specific data */
/*                                      //EEPROM
        ret = mem_acc->read(mem_acc, (char *)&config, 0, sizeof(config));
        if (ret != sizeof(config)) {
                pr_err("AM335X EVM config read fail, read %d bytes\n", ret);
                pr_err("This likely means that there either is no/or a failed
EEPROM\n");
                goto out;
        }
*/
```

去掉以下 EEPROM 检验代码

```
#if 0                                                   //EEPROM
        if (config.header != AM335X_EEPROM_HEADER) {
                pr_err("AM335X: wrong header 0x%x, expected 0x%x\n",
                        config.header, AM335X_EEPROM_HEADER);
                goto out;
        }

        if (strncmp("A335", config.name, 4)) {
                pr_err("Board %s\ndoesn't look like an AM335x board\n",
                        config.name);
                goto out;
        }

        snprintf(tmp, sizeof(config.name) + 1, "%s", config.name);
        pr_info("Board name: %s\n", tmp);
        snprintf(tmp, sizeof(config.version) + 1, "%s", config.version);
        pr_info("Board version: %s\n", tmp);
#endif

#if 0                                                   //EEPROM
        if (!strncmp("A335BONE", config.name, 8)) {
                daughter_brd_detected = false;
                if(!strncmp("00A1", config.version, 4) ||
                   !strncmp("00A2", config.version, 4))
                        setup_beaglebone_old();
                else
                        setup_beaglebone();
        } else if (!strncmp("A335BNLT", config.name, 8)) {
                setup_beagleboneblack();
        } else if (!strncmp("A335X_SK", config.name, 8)) {
                daughter_brd_detected = false;
#endif
```

仅保留对 starterkit 初始化代码，

```
                setup_starterkit();
#if 0
        } else {
                /* only 6 characters of options string used for now */
                snprintf(tmp, 7, "%s", config.opt);
                pr_info("SKU: %s\n", tmp);

                if (!strncmp("SKU#01", config.opt, 6))
                        setup_general_purpose_evm();
                else if (!strncmp("SKU#02", config.opt, 6))
                        setup_ind_auto_motor_ctrl_evm();
                else
                        goto out;
        }
#endif
```

至此，EEPROM 相关效验代码已经去掉，程序将默认执行 starterkit EVM 的初始化。

III. 增加 d_can 驱动

这里再多提一点，在 kernel 里如何在 starterkit 基础上增加 CAN 的初始化，因为 starterkit 本身不支持 CAN 接口。

在 board-am335xevm.c 下

首先在结构体 evm_sk_dev_cfg 增加，

```
/* EVM - Starter Kit */
static struct evm_dev_cfg evm_sk_dev_cfg[] = {
        {am335x_rtc_init, DEV_ON_BASEBOARD, PROFILE_ALL},
        {mmc1_wl12xx_init,      DEV_ON_BASEBOARD, PROFILE_ALL},
        {mmc0_init,     DEV_ON_BASEBOARD, PROFILE_ALL},
        {d_can_init,    DEV_ON_BASEBOARD, PROFILE_ALL},
        {rgmii1_init,   DEV_ON_BASEBOARD, PROFILE_ALL},
        {rgmii2_init,   DEV_ON_BASEBOARD, PROFILE_ALL},
        {lcdc_init,     DEV_ON_BASEBOARD, PROFILE_ALL},
        {enable_ecap2,      DEV_ON_BASEBOARD, PROFILE_ALL},
        {mfd_tscadc_init,       DEV_ON_BASEBOARD, PROFILE_ALL},
        {gpio_keys_init,  DEV_ON_BASEBOARD, PROFILE_ALL},
        {gpio_led_init,  DEV_ON_BASEBOARD, PROFILE_ALL},
        {lis331dlh_init, DEV_ON_BASEBOARD, PROFILE_ALL},
        {mcasp1_init,   DEV_ON_BASEBOARD, PROFILE_ALL},
        {uart1_wl12xx_init, DEV_ON_BASEBOARD, PROFILE_ALL},
        {wl12xx_init,       DEV_ON_BASEBOARD, PROFILE_ALL},
        {gpio_ddr_vtt_enb_init, DEV_ON_BASEBOARD, PROFILE_ALL},
        {sgx_init,      DEV_ON_BASEBOARD, PROFILE_ALL},
        {NULL, 0, 0},
};
```

其次在 d_can_init 函数中增加“case EVM_SK:”并且去掉 if(profile == PROFILE_1)，如下，

```
static void d_can_init(int evm_id, int profile)
{
        switch (evm_id) {
        case IND_AUT_MTR_EVM:
                if ((profile == PROFILE_0) || (profile == PROFILE_1)) {
                        setup_pin_mux(d_can_ia_pin_mux);
                        /* Instance Zero */
                        am33xx_d_can_init(0);
                }
                break;
        case EVM_SK:
        case GEN_PURP_EVM:
        case GEN_PURP_DDR3_EVM:
                setup_pin_mux(d_can_gp_pin_mux);
                /* Instance One */
                am33xx_d_can_init(1);
                break;
        default:
                break;
        }
}
```