



AIC8800D
Wi-Fi6/BT5.0 SoC
休眠唤醒调适手册

Revision: 1.0

2022/06/01

历史更新记录

时间	修改内容	修订人	版本
2022/06/01	初版	Aiden	1.0

Confidential AICSEMI INC

全志 Allwinner 移植方式(SDIO)

dts 部分

需要确认 bt_wake 和 bt_host_wake 是否设定在 dts 当中

如 A133 为例，可在

android/longan/device/config/chips/al33/configs/b4/linux-5.4/board.dts

中看到该设定

```

222 &addr_mgt {
223     compatible = "allwinner,sunxi-addr_mgt";
224     type_addr_wifi = <0x0>;
225     type_addr_bt = <0x0>;
226     type_addr_eth = <0x0>;
227     status = "okay";
228 };
229
230 &bt1pm {
231     compatible = "allwinner,sunxi-bt1pm";
232     uart_index = <0x1>;
233     bt_wake = <gpio PL 4 GPIO_ACTIVE_HIGH>;
234     bt_hostwake = <gpio PL 3 GPIO_ACTIVE_HIGH>;
235     wakeup-source;
236     status = "okay";
237 };
238
239 &sdio2 {
240     non-removable;
241     bus-width = <8>;
242     mmc-ddr-1_8v;
243     mmc-hs200-1_8v;
244     mmc-hs400-1_8v;
245     no-sdio;

```

aic8800_bt1pm

将定义好的 bt_wake 和 bt_host_wake 设定在 aic8800_bt1pm 中

```

00765: GFP_KERNEL);
00766: if (!bsi)
00767:     return -ENOMEM;
00768:
00769: bsi->host_wake = of_get_named_gpio_flags(np, "bt_hostwake", 0, &config);
00770: if (!gpio_is_valid(bsi->host_wake)) {
00771:     BT_ERR("get gpio bt_hostwake failed\n");
00772:     ret = -EINVAL;
00773:     goto err0;
00774: }
00775:
00776: /* set host_wake_assert */
00777: bsi->host_wake_assert = (config == OF_GPIO_ACTIVE_LOW) ? 0 : 1;
00778: BT_DBG("bt_hostwake gpio=%d assert=%d\n", bsi->host_wake, bsi->host_wake_assert);
00779:
00780: if (assert_level != -1) {
00781:     bsi->host_wake_assert = (assert_level & 0x02) > 0;
00782:     BT_DBG("override host_wake assert to %d", bsi->host_wake_assert);
00783: }
00784:
00785: ret = devm_gpio_request(dev, bsi->host_wake, "bt_hostwake");
00786: if (ret < 0) {
00787:     BT_ERR("can't request bt_hostwake gpio %d\n",
00788:           bsi->host_wake);
00789:     goto err0;
00790: }
00791:
00823: }
00824: bsi->ext_wake = of_get_named_gpio_flags(np, "bt_wake", 0, &config);
00825: if (!gpio_is_valid(bsi->ext_wake)) {
00826:     BT_ERR("get gpio bt_wake failed\n");
00827:     ret = -EINVAL;
00828:     goto err2;
00829: }
00830:
00831: ret = devm_gpio_request(dev, bsi->ext_wake, "bt_wake");
00832: if (ret < 0) {
00833:     BT_ERR("can't request bt_wake gpio %d\n",
00834:           bsi->ext_wake);
00835:     goto err2;
00836: }
00837:
00838: /* set ext_wake_assert */
00839: bsi->ext_wake_assert = (config == OF_GPIO_ACTIVE_LOW) ? 0 : 1;
00840: BT_DBG("bt_wake gpio=%d assert=%d\n", bsi->ext_wake, bsi->ext_wake_assert);
00841:
00842: if (assert_level != -1) {
00843:     bsi->ext_wake_assert = (assert_level & 0x01) > 0;
00844:     BT_DBG("override ext_wake assert to %d", bsi->ext_wake_assert);

```

aic8800_bsp

Makefile 中将以下两个 CONFIG 改成 y，并将在 aic_bsp_driver.h 中将

AICBT_LPM_ENABLE_DEFAULT 改成 1

```

Makefile
00001: CONFIG_SDIO_SUPPORT := y
00002: CONFIG_SDIO_PWRCTRL := y
00003: CONFIG_AIC_FW_PATH = "/vendor/etc/firmware"
00004: export CONFIG_AIC_FW_PATH
00005: ccflags-y += -DCONFIG_AIC_FW_PATH="\$(CONFIG_AIC_FW_PATH)\\"
00006:
00007: MODULE_NAME := aic8800_bsp
00008: ifeq ($(CONFIG_SDIO_SUPPORT), y)
00009: ccflags-y += -DAICWF_SDIO_SUPPORT
00010: ccflags-$(CONFIG_SDIO_PWRCTRL) += -DCONFIG_SDIO_PWRCTRL
00011: endif
00012:
00013: CONFIG_GPIO_WAKEUP = y
00014:
00015: ccflags-$(CONFIG_GPIO_WAKEUP) += -DCONFIG_GPIO_WAKEUP
00016:
00017:
00018: obj-m := $(MODULE_NAME).o
00019: $(MODULE_NAME)-y := \
00020:     aic_bsp_main.o \
00021:     aic_bsp_driver.o \
00022:     aic8800.o \
00023:     aic8800_txrxif.o \
00024:     md5.o
00025:
00026:
00027: # Platform support list
00028: CONFIG_PLATFORM_ROCKCHIP ?= n
00029: CONFIG_PLATFORM_ALLWINNER ?= y
00030: CONFIG_PLATFORM_AMLOGIC ?= n
00031: CONFIG_PLATFORM_UBUNTU ?= n
00032:

```

```

Aic_bsp_driver.h
00336: CHIP_REV_U02 = 3,
00337: CHIP_REV_U03 = 7,
00338: CHIP_REV_U04 = 7,
00339: };
00340: ///aic bt tx pwr lvl :lsb->msb: first byte, min pwr lvl; second byte, max pwr
00341: ///pwr lvl:(20(min), 30, 40, 50, 60(max)
00342: #define AICBT_TXPWR_LVL 0x00006020
00343:
00344: #define AICBSP_HWINFO_DEFAULT (-1)
00345: #define AICBSP_CPMODE_DEFAULT AICBSP_CPMODE_WORK
00346: #define AICBSP_FWLOG_EN_DEFAULT 0
00347:
00348: #define AICBT_BTMODE_DEFAULT AICBT_BTMODE_BT_ONLY_SW
00349: #define AICBT_BTPORT_DEFAULT AICBT_BTPORT_UART
00350: #define AICBT_UART_BAUD_DEFAULT AICBT_UART_BAUD_1_5M
00351: #define AICBT_UART_FC_DEFAULT AICBT_UART_FLOWCTRL_ENABLE
00352: #define AICBT_LPM_ENABLE_DEFAULT 1
00353: #define AICBT_TXPWR_LVL_DEFAULT AICBT_TXPWR_LVL
00354:
00355: #define FEATURE_SDIO_CLOCK 60000000 // 0: default, other: target c
00356: #define FEATURE_SDIO_PHASE 2 // 0: default, 2: 180度
00357:

```

aic8800_fdrv

在 Makefile 中将以下两个 CONFIG 改成 y，平台选择 CONFIG_PLATFORM_ALLWINNER

```

Makefile
00036: # Support of P2P Debugfs for enabling/disabling NOA and Op
00037: CONFIG_RWNX_P2P_DEBUGFS := n
00038:
00039: # } // WAITING FOR KCONFIG
00040:
00041:
00042: # Enable A-MSDU support (need FW support)
00043: ## Select this if FW is compiled with AMSDU support
00044: CONFIG_RWNX_SPLIT_TX_BUF ?= n
00045: ## Select this TO send AMSDU
00046: CONFIG_RWNX_AMSDUS_TX ?= n
00047:
00048: # Enable BFMER support (need FW support)
00049: CONFIG_RWNX_BFMER ?= n
00050:
00051: CONFIG_SDIO_SUPPORT = y
00052: CONFIG_USB_SUPPORT = n
00053: CONFIG_RX_REORDER ?= y
00054: CONFIG_ARP_OFFLOAD = y
00055: CONFIG_RADAR_OR_IR_DETECT = n
00056: CONFIG_DOWNLOAD_FW = y
00057: CONFIG_RFTST = y
00058: CONFIG_USB_BT = y
00059: CONFIG_USB_5G ?= y
00060: CONFIG_SDIO_PWRCTRL ?= y
00061: CONFIG_CREATE_TRACE_POINTS = n
00062: CONFIG_TXRX_THREAD_PRIO = n
00063: # CONFIG_COEX = n for BT ONLY, CONFIG_COEX = y for combo an
00064: CONFIG_COEX = y
00065: CONFIG_GPIO_WAKEUP = y
00066: CONFIG_SET_VENDOR_EXTENSION IE = n
00067: CONFIG_SUPPORT_REALTIME_CHANGE_MAC = y
00068: CONFIG_WPA3_FOR_OLD_KERNEL ?= n
00069:

```

```

11:
12: ccflags-$(CONFIG_RX_REORDER) += -DAICWF_RX_REORDER
13: ccflags-$(CONFIG_ARP_OFFLOAD) += -DAICWF_ARP_OFFLOAD
14: ccflags-$(CONFIG_RADAR_DETECT) += -DRADAR_OR_IR_DETECT
15: ccflags-$(CONFIG_DOWNLOAD_FW) += -DCONFIG_DOWNLOAD_FW
16:
17: # Platform support list
18: CONFIG_PLATFORM_ROCKCHIP ?= n
19: CONFIG_PLATFORM_ALLWINNER ?= y
20: CONFIG_PLATFORM_AMLOGIC ?= n
21: CONFIG_PLATFORM_UBUNTU ?= n
22: CONFIG_PLATFORM_INGENIC_T20 ?= n
23:
24: ccflags-y += -DAIC_TRACE_INCLUDE_PATH=$(src)
25:

```

libbt-vendor

```

BLUETOOTH_UART_DEVICE_PORT = "/dev/ttyS0"
FW_PATCHFILE_LOCATION = "/vendor/firmware/"
LPM_IDLE_TIMEOUT_MULTIPLE = 5
SCO_USE_I2S_INTERFACE = TRUE
BTVND_DBG = TRUE
BTHW_DBG = TRUE
VNDUSERIAL_DBG = TRUE
UPIO_DBG = FALSE
PROC_BTWRITE_TIMER_TIMEOUT_MS = 0
BT_WAKE_VIA_PROC = TRUE
BT_WAKE_VIA_PROC_NOTIFY_DEASSERT = TRUE

```

以上设定完后即可开启休眠唤醒的功能

瑞芯微 Rockchip 移植方式(SDIO)

aic8800_bt1pm

RK 平台默认已经有 bwrite 等接口，不需要使用 aic8800_bt1pm 处理蓝牙休眠唤醒。

aic8800_bsp

Makefile 中将以下两个 CONFIG 改成 y，在 aic_bsp_driver.h 中将 AICBT_LPM_ENABLE_DEFAULT 改成 1

```

Makefile
00001: CONFIG_SDIO_SUPPORT := y
00002: CONFIG_SDIO_PWRCTRL := y
00003: CONFIG_AIC_FW_PATH = "/vendor/etc/firmware"
00004: export CONFIG_AIC_FW_PATH
00005: ccflags-y += -DCONFIG_AIC_FW_PATH="$(CONFIG_AIC_FW_PATH)\\"
00006:
00007: MODULE_NAME := aic8800_bsp
00008: ifeq ($(CONFIG_SDIO_SUPPORT), y)
00009: ccflags-y += -DAICWF_SDIO_SUPPORT
00010: ccflags-y += -DCONFIG_SDIO_PWRCTRL
00011: endif
00012:
00013: CONFIG_GPIO_WAKEUP = y
00014:
00015: ccflags-y += -DCONFIG_GPIO_WAKEUP
00016:
00017:
00018: obj-m := $(MODULE_NAME).o
00019: $(MODULE_NAME)-y := \
00020:     aic_bsp_main.o \
00021:     aic_bsp_driver.o \
00022:     aicdio.o \
00023:     md5.o
00024:
00025:
00026:
00027: # Platform support list
00028: CONFIG_PLATFORM_ROCKCHIP ?= y
00029: CONFIG_PLATFORM_ALLWINNER ?= n
00030: CONFIG_PLATFORM_AMLOGIC ?= n
00031: CONFIG_PLATFORM_UBUNTU ?= n
00032:
00033: ifeq ($(CONFIG_PLATFORM_ROCKCHIP), y)
00034: ccflags-y += -DCONFIG_PLATFORM_ROCKCHIP

```

```

aic_bsp_driver.h
00336: CHIP_REV_U02 = 3,
00337: CHIP_REV_U03 = 7,
00338: CHIP_REV_U04 = 7,
00339: };
00340: /// aic bt tx pwr lvl :lsb->msb: first byte, min pwr lvl; second byte, max pwr
00341: /// pwr lvl:(min), 30, 40, 50, 60(max)
00342: #define AICBT_TXPWR_LVL 0x00006020
00343:
00344: #define AICBSP_HWINFO_DEFAULT (-1)
00345: #define AICBSP_CPMODE_DEFAULT AICBSP_CPMODE_WORK
00346: #define AICBSP_FWLOG_EN_DEFAULT 0
00347:
00348: #define AICBT_BTMODE_DEFAULT AICBT_BTMODE_BT_ONLY_SW
00349: #define AICBT_BTPORT_DEFAULT AICBT_BTPORT_UART
00350: #define AICBT_UART_BAUD_DEFAULT AICBT_UART_BAUD_1_5M
00351: #define AICBT_UART_FC_DEFAULT AICBT_UART_FLOWCTRL_ENABLE
00352: #define AICBT_LPM_ENABLE_DEFAULT 1
00353: #define AICBT_TXPWR_LVL_DEFAULT AICBT_TXPWR_LVL
00354:
00355: #define FEATURE_SDIO_CLOCK 60000000 // 0: default, other: target c
00356: #define FEATURE_SDIO_PHASE 2 // 0: default, 2: 180度
00357:

```

aic8800_fdrv

在 Makefile 中将以下两个 CONFIG 改成 y，平台选择 CONFIG_PLATFORM_ROCKCHIP


```

Makelife
00199: endif
00200:
00201: ifeq $(CONFIG_RWNX_BCMCM), y)
00202: ccflags-y += -DNX_TXQ_CNT=5
00203: else
00204: ccflags-y += -DNX_TXQ_CNT=4
00205: endif
00206:
00207: # For old kernel (<=3.19)
00208: ifeq $(shell test $(VERSION) -lt 4 -a "$(CONFIG_VENDOR_RWNX)
ccflags-y += -DCONFIG_VENDOR_RWNX_VHT_N80
00210: endif
00211:
00212: ccflags-$(CONFIG_RX_REORDER) += -DAICWF_RX_REORDER
00213: ccflags-$(CONFIG_ARP_OFFLOAD) += -DAICWF_ARP_OFFLOAD
00214: ccflags-$(CONFIG_RADAR_DETECT) += -DRADAR_OR_IR_DETECT
00215: ccflags-$(CONFIG_DOWNLOAD_FW) += -DCONFIG_DOWNLOAD_FW
00216:
00217: # Platform support list
00218: CONFIG_PLATFORM_ROCKCHIP ?= y
00219: CONFIG_PLATFORM_ADLWINNER ?= n
00220: CONFIG_PLATFORM_AMLOGIC ?= n
00221: CONFIG_PLATFORM_UBUNTU ?= n
00222: CONFIG_PLATFORM_INGENIC_T20 ?= n
00223:
00224: ccflags-y += -DAIC_TRACE_INCLUDE_PATH=$(src)
00225:
00226: ifeq $(CONFIG_PLATFORM_INGENIC_T20), y)

```

libbt-vendor

在 vnd_generic.txt 中将以下两个参数打开

```
BLUETOOTH_UART_DEVICE_PORT = "/dev/ttyS0"
FW_PATCHFILE_LOCATION = "/vendor/firmware/"
LPM_IDLE_TIMEOUT_MULTIPLE = 5
SCO_USE_I2S_INTERFACE = TRUE
BTVND_DBG = TRUE
BTHW_DBG = TRUE
VNDUSERIAL_DBG = TRUE
UPIO_DBG = FALSE
PROC_BTWRITE_TIMER_TIMEOUT_MS = 0
BT_WAKE_VIA_PROC = TRUE
BT_WAKE_VIA_PROC_NOTIFY_DEASSERT = TRUE
```

在 upio.c 中修改 init_rfkill()的代码

```
static int init_rfkil()
{
    #if 1//For RK
        char path[64];
        char buf[16];
        int fd, sz, id;
```

```
int rfkill_id;

if (is_rfkill_disabled())
    return -1;

for (id = 0; ; id++)
{
    snprintf(path, sizeof(path), "/sys/class/rfkill/rfkill%d/type", id);
    fd = open(path, O_RDONLY);
    if (fd < 0)
    {
        ALOGE("init_rfkill : open(%s) failed: %s (%d)\n", \
            path, strerror(errno), errno);
        return -1;
    }

    sz = read(fd, &buf, sizeof(buf));
    close(fd);

    if (sz >= 9 && memcmp(buf, "bluetooth", 9) == 0)
    {
        rfkill_id = id;
        break;
    }
}

asprintf(&rfkill_state_path, "/sys/class/rfkill/rfkill%d/state", rfkill_id);
return 0;

#endif

#if 0
char path[64];
char buf[16];
int fd, sz, id;
const char *basepath = "/sys/devices/platform/aic-bt/rfkill";

DIR *d;
struct dirent *de;
if (!(d = opendir(basepath)))
    goto fail;

while ((de = readdir(d))) {
```

```
if (strstr(de->d_name, "rfkill")) {
    snprintf(path, sizeof(path), "%s/%s/type", basepath, de->d_name);
    fd = open(path, O_RDONLY);
    if (fd < 0)
        continue;

    sz = read(fd, &buf, sizeof(buf));
    close(fd);

    if (sz >= 9 && memcmp(buf, "bluetooth", 9) == 0) {
        ALOGD("%s: rfkill path %s/%s", __func__, basepath, de->d_name);
        asprintf(&rfkill_state_path, "%s/%s/state", basepath, de->d_name);
        closedir(d);
        return 0;
    }
}
}
closedir(d);

fail:
    ALOGE("%s: No rfkill control node found", __func__);
    return -1;
#endif
}
```

以上设定完后即可开启休眠唤醒的功能

Q&A

Q:成功休眠下去时，ic 功耗大约为多少

A:大约在 7-8mA

Q:发现平台休眠下去时 ic 功耗降不下来

A:确认 ic 的 bt_wake 脚是否为低，并且透过 ic 打印确认是否能敲回车

Q:平台休眠后 ping 不通

A:确认 wifi_host_wake 脚是否拉高，SDIO 的 D1 是否拉低，若以上两个电平符合，可用镊子点一下 wifi_host_wake 是否可唤醒主控，若不可唤醒，请检查是否相关的 CONFIG 没打开