## RTC读写相关指令

1. 设置外部时钟源

IOCTL\_SET\_CLK\_SOURCE

使用示例：

设置外部时钟1（0x11）

uint8\_t val8 = 0x11;

ret = ioctl(fd, IOCTL\_SET\_CLK\_SOURCE, &val8);

1. 时钟参数秒设置使能

IOCTL\_SET\_SECOND\_EN

使用示例：

使能秒设置（驱动内部执行0x55→0xAA序列）

ret = ioctl(fd, IOCTL\_SET\_SECOND\_EN);

1. 设置秒值

IOCTL\_SET\_SECOND\_VALUE

使用示例：

设置秒值（例如：1620000000 对应2021-05-03 00:00:00）

uint64\_t val64 = 1620000000ULL;

ret = ioctl(fd, IOCTL\_SET\_SECOND\_VALUE, &val64);

1. 读取当前时钟源

IOCTL\_GET\_CLK\_SOURCE

使用示例：

ret = ioctl(fd, IOCTL\_GET\_CLK\_SOURCE, &val8);

switch (val8) {

case 0x11: printf("(External clock 1)\n"); break;

case 0x22: printf("(External clock 2)\n"); break;

case 0x33: printf("(Internal reference clock)\n"); break;}

1. 读取外部时钟状态

IOCTL\_GET\_EXT\_CLK\_STATE

使用示例：

ret = ioctl(fd, IOCTL\_GET\_EXT\_CLK\_STATE, &val8);

uint8\_t clk1 = (val8 >> 4) & 0x0F;

uint8\_t clk2 = val8 & 0x0F;

printf("Clock 1 state: %s\n", (clk1 == 0x05) ? "Locked" : "Unlocked");

printf("Clock 2 state: %s\n", (clk2 == 0x05) ? "Locked" : "Unlocked");

1. 读取当前秒值

IOCTL\_GET\_SECOND\_VALUE

使用示例：

ret = ioctl(fd, IOCTL\_GET\_SECOND\_VALUE, &val64);

print\_hex64(val64, "Current second value");

1. 读取当前微秒值

IOCTL\_GET\_MICRO\_VALUE

使用示例：

ret = ioctl(fd, IOCTL\_GET\_MICRO\_VALUE, &val64);

print\_hex64(val64, "Current microsecond value");

1. 读取调试用秒值锁存

IOCTL\_GET\_DEBUG\_SECOND

使用示例：

ret = ioctl(fd, IOCTL\_GET\_DEBUG\_SECOND, &val64);

if (ret < 0) {

perror("IOCTL\_GET\_DEBUG\_SECOND failed");

} else {

print\_hex64(val64, "Debug second value (latched)");

}

1. 读取调试用微秒值锁存

IOCTL\_GET\_DEBUG\_MICRO

使用示例：

ret = ioctl(fd, IOCTL\_GET\_DEBUG\_MICRO, &val64);

if (ret < 0) {

perror("IOCTL\_GET\_DEBUG\_MICRO failed");

} else {

print\_hex64(val64, "Debug microsecond value (latched)");

}

1. 读取RTC计时器

IOCTL\_GET\_RTC\_TIMER

使用示例：

ret = ioctl(fd, IOCTL\_GET\_RTC\_TIMER, &val64);

if (ret < 0) {

perror("IOCTL\_GET\_RTC\_TIMER failed");

} else {

print\_hex64(val64, "RTC timer value");

}

## 中断读写相关指令

1. 设置PPS触发模式

IOCTL\_SET\_PPS\_TRIGGER

使用示例：

设置PPS触发模式为上升沿（0x55）

val8 = 0x55;

ret = ioctl(fd, IOCTL\_SET\_PPS\_TRIGGER, &val8);

1. 设置PPS触发模式

IOCTL\_SET\_PPS\_TRIGGER

使用示例：

设置PPS触发模式为上升沿（0x55）

val8 = 0x55;

ret = ioctl(fd, IOCTL\_SET\_PPS\_TRIGGER, &val8);

1. 读取PPS1状态

IOCTL\_GET\_PPS1\_STATE

使用示例：

ret = ioctl(fd, IOCTL\_GET\_PPS1\_STATE, &val8);

(val8 == 0x55) "Normal" :

(val8 == 0xAA) "Cleared" : "Error"

1. 读取PPS2状态

IOCTL\_GET\_PPS2\_STATE

使用示例：

ret = ioctl(fd, IOCTL\_GET\_PPS1\_STATE, &val8);

(val8 == 0x55) "Normal" :

(val8 == 0xAA) "Cleared" : "Error"

1. 设置比较值

IOCTL\_SET\_COMPARE\_VALUE

使用示例：

设置比较值为100ms

val64 = 100000ULL;

ret = ioctl(fd, IOCTL\_SET\_COMPARE\_VALUE, &val64);

1. 开启(0xaa)或关闭(0x55)比较中断

IOCTL\_SET\_COMPARE\_MODE

使用示例：

val8 = 0xAA;//启动

ret = ioctl(fd, IOCTL\_SET\_COMPARE\_MODE, &val8);

1. 读取当前中断状态(0x11 pps中断 ，0x22比较中断 ，0x33 两者都有)

IOCTL\_GET\_IRQ\_STATUS

使用示例：

ret = ioctl(fd, IOCTL\_GET\_IRQ\_STATUS, &irq\_status);

if (ret < 0) {

perror("IOCTL\_GET\_IRQ\_STATUS failed");

} else {

printf("当前中断状态: 0x%02X\n", irq\_status);

}

1. 开启或关闭中断屏蔽

IOCTL\_SET\_IRQ\_MASK

使用示例：

// 1. 屏蔽PPS秒脉冲中断（设置bit 2）

val8 = 0x04; // 0b0100

ret = ioctl(fd, IOCTL\_SET\_IRQ\_MASK, &val8);

if (ret < 0) {

perror("IOCTL\_SET\_IRQ\_MASK failed");

} else {

printf("已屏蔽PPS秒脉冲中断 (0x04)\n");

}

// 2. 屏蔽比较中断（设置bit 3）

val8 = 0x08; // 0b1000

ret = ioctl(fd, IOCTL\_SET\_IRQ\_MASK, &val8);

if (ret < 0) {

perror("IOCTL\_SET\_IRQ\_MASK failed");

} else {

printf("已屏蔽比较中断 (0x08)\n");

}

// 3. 同时屏蔽PPS秒脉冲中断和比较中断（设置bit 2和bit 3）

val8 = 0x0C; // 0b1100

ret = ioctl(fd, IOCTL\_SET\_IRQ\_MASK, &val8);

if (ret < 0) {

perror("IOCTL\_SET\_IRQ\_MASK failed");

} else {

printf("已同时屏蔽PPS秒脉冲中断和比较中断 (0x0C)\n");

}

// 4. 不屏蔽任何中断（清除bit 2和bit 3）

val8 = 0x00;

ret = ioctl(fd, IOCTL\_SET\_IRQ\_MASK, &val8);

if (ret < 0) {

perror("IOCTL\_SET\_IRQ\_MASK failed");

} else {

printf("已清除所有中断屏蔽 (0x00)\n");

}

1. 清除中断

IOCTL\_CLEAR\_IRQ

使用示例：

ret = ioctl(fd, IOCTL\_CLEAR\_IRQ, &irq\_status);

if (ret < 0) {

perror("IOCTL\_CLEAR\_IRQ failed");

} else {

printf("中断已清除\n");}

1. 获取当前输出pps源(0x11 pps外部源1 ，0x22外部源2 ，0x33 内部源)

IOCTL\_GET\_PPS\_STATE

使用示例：

uint8\_t pps\_state;

ret = ioctl(fd, IOCTL\_GET\_PPS\_STATE, &pps\_state);

if (ret < 0) {

perror("IOCTL\_GET\_PPS\_STATE failed");

} else {

printf("PPS state: 0x%02X\n", pps\_state);}

1. 读取两个pps秒脉冲之间比较中断的计数值

IOCTL\_COMPARE\_CNT

使用示例：

uint8\_t count;

ret = ioctl(fd, IOCTL\_COMPARE\_CNT, &count);

if (ret < 0) {

perror("IOCTL\_COMPARE\_CNT failed");

} else {

printf(" - (第 %d 次)\n",count);}