CY16173　八木橋晃一　　プロジェクト演習２__中間課題

問１

・tactsw_opent()：input.c の open()が実行されたとき
・tactsw_read()：input.c の read()が実行されたとき
・tactsw_release()：input.c の close()が実行されたとき
・tactsw_intr()：スイッチが押されたとき（割り込みハンドラが起きたとき）
・tactsw_setup()：tactsw_init()内の ret = tactsw_setup(major);が実行されたとき
・tactsw_init()：アルマジロでコマンド # insmod tactsw.ko を実行したとき
・tactsw_exit()：アルマジロでコマンド # rmmod tactsw.ko を実行したとき

問２

ソースコード

```
printk("tactsw_read:BEFORE_wait_event¥n");
printk("tactsw_read:before_mlen=%d¥n",tactsw_info.mlen);
ret = wait_event_interruptible(tactsw_info.wq, (tactsw_info.mlen != 0) );
printk("tactsw_read:after_mlen=%d¥n",tactsw_info.mlen);
printk("tactsw_read:AFTER_wait_event¥n");
```

どのように調査し、確認したか

tactsw.c の関数 tactsw_read()内において、上記のソースコードのように printk を挿入し、どこで待ちに入るかを確認した。

動作説明

「tactsw_read:BEFORE_wait_event」、「tactsw_read:mlen= 0 」が出力され待ちの状態に入った。その後、スイッチを押すと待ちの状態が解除され、「tactsw_read:AFTER_wait_event」が出力された後に再び「tactsw_read:BEFORE_wait_event」、「tactsw_read: before_mlen= 0 」が出力され待ちの状態に入った。このことから、tactsw_read()内の wait_event_interruptible(tactsw_info.wq, (tactsw_info.mlen != 0))にて待ちが実現されている。また、出力結果から mlen の値が 0 の時に待ちに入ることがわっかた。

問３

ソースコード

tactsw_read()内

```
printk("tactsw_read:before_mlen=%d¥n",tactsw_info.mlen);
ret = wait_event_interruptible(tactsw_info.wq, (tactsw_info.mlen != 0) );
printk("tactsw_read:after_mlen=%d¥n",tactsw_info.mlen);
```

tactsw_intr()内

```
printk("tactsw_intr:BEFORE_wake_up_interruptible¥n");
wake_up_interruptible(&(tactsw_info.wq));

printk("tactsw_intr:IRQ_HANDLED_END¥n");
        return IRQ_HANDLED;
```

どのように調査し、確認したか

各関数の呼び出される前と呼び出された後、各関数の始まりと終わり、条件分岐の始まりに printk を挿入し、実行して出力結果を確認した。また、上記のソースコードのように printk を挿入して、待ちに入る前と後の mlen の値を比較した。

次に tactsw_intr（）内の「tactsw_info.mlen = mlen+1;」をコメント化して実行した。

その後、「tactsw_info.mlen = mlen+1;」のコメント化を解除し、tactsw_read()内の「tactsw_info.mlen -= read_size;」をコメント化して実行した。

最後に、「tactsw_info.mlen -= read_size;」のコメント化を解除し、tactsw_intr()内の「wake_up_interruptible(&(tactsw_info.wq));」をコメント化して実行した。

動作説明

input.c を実行されたあとに、tactsw_open()が実行され、tactsw_read()内の「ret = wait_event_interruptible(tactsw_info.wq, (tactsw_info.mlen != 0));」にて待ちに入る。待ちに入る前の mlen の値は 0 であった。その後スイッチを押す（キー入力を 2 回する）と tactsw_intr()が呼ばれ「tactsw_intr:BEFORE_wake_up_interruptible」、「tactsw_intr:IRQ_HANDLED_END」が出力された後に「tactsw_read:after_mlen=2」が出力された。このことから、「wake_up_interruptible(&(tactsw_info.wq));」でプロセスが起こされ、プロセスが再会されるのは tactsw_intr() の処理が終わったあとの、「ret = wait_event_interruptible(tactsw_info.wq, (tactsw_info.mlen != 0));」からである。

・「tactsw_info.mlen = mlen+1;」をコメント化した場合
あらかじめ何もキー入力がないときは tactsw_intr（）の処理が行われるだけになった。また、mlen の値はキー入力を行っても変化しなかった。

・「tactsw_info.mlen -= read_size;」をコメント化した場合
あらかじめ３回押し（６回キー入力し）、待ちに入るまで input.c を実行した。その結果終始 mlen の値は 6 のままであった。次に 1 回押し(2 回キー入力し) input.c を実行すると mlen の値は終始８であった。

・「wake_up_interruptible(&(tactsw_info.wq));」をコメント化した場合
なにもキー入力がない状態で input.c を実行し、待ちに入ってからキー入力をした。その結果、「rwait_event_interruptible(tactsw_info.wq, (tactsw_info.mlen != 0))」が実行されたあと、tactsw_intr () が実行されたが、「if (irq == gpio_to_irq(gpio))」、「if (val==0)」、「if (mlen < MSGLEN)」の順に入ったあと、「return IRQ_HANDLED；」まで実行され、「if (irq == gpio_to_irq(gpio))」、「if (mlen < MSGLEN)」の順に入り「return IRQ_HANDLED；」まで実行された。このループから抜けることができなくなり、プロセスが起きなくなったことにより動作を再開しなくなった。

問４
ソースコード
```
printk("tactsw_read:BEFORE_wait_event¥n");
printk("tactsw_read:before_mlen=%d¥n",tactsw_info.mlen);
ret = wait_event_interruptible(tactsw_info.wq, (tactsw_info.mlen != 0) );
printk("tactsw_read:AFTER_wait_event¥n");
```

どのように調査したか
あらかじめ６回押し(12 回キー入力し)、待ちに入るまで input.c を実行した。

どのように確認したか
各関数の呼び出される前と呼び出された後、各関数の始まりと終わり、条件分岐の始まりに printk を挿入し、実行して出力結果を確認した。

動作説明
input.c を実行するとまず open()が実行され、その後 tactsw_open()が実行される。次に read()が実行され、tactsw_read()の実行が開始される。「tactsw_read:BEFORE_wait_event」、

「tactsw_read: before_mlen=12」、「tactsw_read:AFTER_wait_event」と出力されたことから、今回（あらかじめキー入力があった場合）は待ちに入っていないことが確認された。その後、tactsw_read()の処理が終わり、write()が実行された後に再び read()が実行される。この過程を通して mlen の値が減少していくことが確認された。また、３回目の input.c の実行の際に mlen の値が 0 となり、待ちに入ることが確認された。