实验 3 报告

学号：2016K8009929029
姓名：张丽玮

一、实验任务（10%）

用 MIPS 汇编写一个软件编程电子表，运行在 soc_lite 上，需要点用一些外设，从而在板子上实现一个 12/24 的时钟功能，最好能够包括闹钟功能等。可以考虑一下精确计时，采用中断、轮询等方法。

二、实验设计（30%）

总的来说实现几个方面：

（1）与外设的连接

（2）时分秒的累加进位操作

（3）复位操作

（4）时分秒设置功能

（5）利用中断尽量提高计时精确性

1、与外设的连接

阅读 ucas_cde.h，里面包含了外设的对应地址，这一次主要用到的是这两个宏定义：

```
#define NUM_ADDR        0xbfaff010
#define SWITCH_ADDR     0xbfaff020
#define BTN_KEY_ADDR    0xbfaff024
```

NUM_ADDR 是用来显示数值的，而 BIN_KEY_ADDR 是按键按下时的传入信号地址。

将数值传入 NUM_ADDR 地址，也可以从 NUM_ADDR 读出现在的时间，而从 BIN_KEY_ADDR 读取是否有按键按下。

2、时分秒的累加进位操作

从 seconds 开始每次加一，一旦到了要进位处就置零，改写高位（min 和 hour）。主要要考虑一个 16 进制时分秒每个各占 2 位，分为低位和高位处理，以及在 12 小时制和 24 小时制下，hour 的进位标准不同。

基本步骤大致如下（Min 和 hour 相似）

```
second_1:
    .set noreorder
    li    k1, NUM_ADDR
    lw    k0, 0(k1)
    andi  t0, k0, 0x000f  #sec lo
    andi  t1, k0, 0x00f0  #sec hi
    srl   t1, t1, 0x4

    li    t6, 0x6
    li    t7, 0xa
    addiu t0, t0, 0x1
    beq   t0, t7, seclo
    addiu k0, k0, 0x1     #delay slot, seclo+1
    j     ret_to_ra
    nop

seclo:
    .set noreorder
    addi  k0, k0, -10     #set sec_lo to zero
    addiu t1, t1, 0x1     #sec_hi_temp+1
    beq   t1, t6, sechi
    addiu k0, k0, 0x10    #delay slot, sechi+1
    j     ret_to_ra
    nop

sechi:
    .set noreorder
    addi  k0, k0, -96     #set sec_hi to zero
    j     ret_to_ra
    nop
```

3、复位操作

实际上就是获取复位按键是否按下，如果按下就后从 23 (12 进制下为 11)时 59 分 55 秒开始计时。

否则正常进行计时。这里也采用了轮询的方式，来判断按键是否按下。这里注意用 cp0 的寄存器实现，count 是自带时钟，status 和 cause 用来判断中断，compare 是人为设置，表示多少时间片 count==compare 的时候进入时钟中断。

```
c_to_normal:
    li      t2, KBT_SET
    li      t1, BTN_KEY_ADDR
    lw      t1, 0(t1)
    beq     t1, t2, c_to_normal
    nop
    li      k0,   CYCLE
    mtc0    k0,   CP0_COMPARE
    nop
    mtc0    zero, CP0_CAUSE
    nop
    mfc0    ra,   CP0_EPC
    li      k0,   NORMAL_MODE_S
    j       handle_finish
    nop
```

4、时分秒设置功能

实际上还是一个读取按键的操作，根据按键情况，写入 NUM_ADDR 里显示出来。这里也是轮询来判断是否设置了时分秒，从而可以达到及时输出的效果。

```
read:
    li      t1, BTN_KEY_ADDR
    lw      k0, 0(t1)
    beq     k0, zero, read      # if equal to 0, no button is touched
    nop
    li      t2, KBT_HOUR        # load the value of hour KBT
    beq     k0, t2, h_1_jal     # if equal, then go to hour+1
    nop
    li      t2, KBT_MIN         # load the value of minite KBT
    beq     k0, t2, m_1_jal     # if equal, then go to minite+1
    nop
    li      t2, KBT_SEC         # load the value of second KBT
    beq     k0, t2, s_1_jal     # if equal, then go to second+1
    nop
    li      t2, KBT_SET         # load the kbt value of SET
    beq     k0, t2, c_to_normal # if equal again, go to wait set KBT disappear
    nop
    j       d1                  # Otherwise, keep reading
    nop
```

5、时钟中断

时钟中断实际上就是 compare==count 的时候进入一个 Interrupt 操作，

```
time_irq:
    # load the address of num confreg
    li      k1,   NUM_ADDR
    lw      k0,   0(k1)
    jal     convert
    nop
    nop
    sw      k0,   0(k1)
    # Reset
    li      k0,   CYCLE
    mtc0    k0,   CP0_COMPARE
    nop
    mtc0    zero, CP0_CAUSE
    nop
    mfc0    ra,   CP0_EPC
    li      k0,   NORMAL_MODE_S
    j       clk_return
    nop
    nop
```

实际上就是从 NUM_ADDR 里取数，进行校准的一个过程。校准之后再将寄存器重置，准备下一个时钟中断。

三、实验过程（60%）

（一）实验流水账

10 月 29 日 读任务书。

10 月 30 日 阅读相关资料。

10 月 31 日 做操作系统时钟中断相关，顺便思考体系结构实验的时钟中断。

11 月 1 日 还在写操作系统和人工智能，草草看了几眼具体的仿真实现过程。

11 月 2 日 开始动手写代码，写了外设连接部分。

11 月 3-4 日 基本完成代码，但是无法上板测试。

11 月 5 日 测试通过。

（二）错误记录

具体描述实验过程中的错误，环境问题、仿真阶段、上板阶段的都可以记录。

1、错误 1

（1）错误现象

显示结果为 func2-3 的测试结果

（2）分析定位过程

定位具体生成操作有什么错误。

（3）错误原因

并没有实际读懂任务书，obj 中的文件并非自己 start.s 生成的。

（4）修正效果

正常编译，不再显示 func2-3 结果

（5）归纳总结（可选）

仔细搞清楚修改 start 文件如何影响 bit 文件生成，多读几遍并理解任务书。

1、错误 2

（1）错误现象

显示结果一直为 0.

（2）分析定位过程

先分析逻辑是否有错，并未将结果传入 NUM_ADDR 中还是传入不及时还是模式不对。

（3）错误原因

按键连接地址出错。

（4）修正效果

正常输出结果。

（5）归纳总结（可选）

自习阅读.h 文件并且搞清楚那些宏定义的意义。

（这次没有图，也没有详细的论据。原因是我自己连不上板子，然后对于仿真调试没有理解清楚，只能勉强请同学帮我看一下 bit 跑的情况的样子……基本只能一句句筛查 debug）

四、实验总结（可选）

为什么会这样呢，明明原本是好的，这是为什么呢？

上一次 vivado 炸了之后，这一次连不上板子，之前连不上倒是没事，基本仿真 PASS 了上板问题不大，但是这一次不行啊啊啊啊啊啊啊。

我最大的敌人可能是我自己的电脑。

（顺带一提我操作系统早就连不上了然后因为 TLB 例外验收出事了 orz）