```
1
    Entrance:
                     j Start
                     nop
3
    EXCINTHandler:
                     mfc0 $k0, $12
                                            # $k0 <- CP0.$cause
                                           # $k1 = EXcCode (cause[3:2])
                     andi $k1, $k0, 0xc
4
5
                                            # 0 \times 0100, syscall
                     addi $s1, $zero, 0x4
6
                     addi $s2, $zero, 0x8
                                             \# 0x1000, UnInstr
                     addi $s3, $zero, 0xC
7
                                             # 0x1100, OV
8
                     beq $k1, $s1, Handle_SYSCALL
                     beq $k1, $s2, Handle_UnInstr
beq $k1, $s3, Handle_OV
9
10
                     sll $v1, $v1, 0x1
ori $v1, $v1, 0x1
11
    Handle INT:
                                             #循环右移,每次在最低位补 1
12
13
                     addi fp, fp, 0x4
                     andi $fp, $fp, 0x003F # 更新 $fp, 因为预置数字和预置图像都是 16
14
                     个数据一组, 所以用 6 位 mask (4 + 2, 地址最低两位恒为 2'b00)
15
                     addi $v0, $v0, 0x1
                                             # increase $v0 for SYSCALL
                                             16
                     bne $v0, $at, Disp
                     step is useless in this program, since $v0 [0~32])
17
                     addi $v0, $zero, 0x5
18
                     addi $s1, $zero, 0x8
                                             # 5'b01000, SW[4:3]=2'b01 && SW[0]=1
     Disp:
                     addi $s2, $zero, 0x10
19
                                           # 5'b10000, SW[4:3]=2'b10 && SW[0]=1
20
                     addi $s3, $zero, 0x18
                                             # 5'b11000, SW[4:3]=2'b11 && SW[0]=0
21
                     lw $s5, 0x0 ($a2)
22
                     andi $s5, $s5, 0x18
                                             \# 0x18 = 5'b11000, mask to get SW[4:3]
23
                     beq $s5, $zero, SW 00
                                             # SW[4:3]=2'b00 (&& SW[0]=0), dot/line of
                     SSeg7 shift in loop.
                     beq $s5, $s1, SW 01
24
                                             \# SW[4:3]=2'b01 (&& SW[0]=0), 0x00000000 ->
                     0x11111111 -> ... -> 0xFFFFFFF
25
                     beq $s5, $s2, SW 10
                                             # SW[4:3]=2'b10 (&& SW[0]=0), show cycle
                     accumulation of $v0
26
                     beq $s5, $s3, SW 11
                                             # SW[4:3]=2'b11 (&& SW[0]=0), show pictures
27
     SW 00:
                     bne $v1, $at, L3
                                             # if ($v1 = 0xFFFFFFFF)
                                             #
28
                     sll $v1, $v1, 0x1
                                                     v1 \ll 0x1 // v1 = 0xfffffff
                                             # else
29
                     sw $v1, 0x0($a1)
    L3:
30
                     j Disp done
                                             #
                                                      // show $t0 on SSeg7
    SW 01:
                                             # 显示预置数字
31
                     lw $k0, 0x20($fp)
32
                     sw $k0, 0x0($a1)
33
                     j Disp_done
34
    SW 10:
                     sw $v0, 0x0 ($a1)
                                             # 显示 $v0 (累加)
35
                     j Disp_done
36
    SW 11:
                     lw $k0, 0x60 ($fp)
                                             # show PictureSet1
                     sw $k0, 0x0($a1)
37
    Disp done:
38
                     lw $s1, 0x0 ($a2)
                                             # $s1 = {counte$0 out, counte$1 out,
    counte$2 out, led out[0x12:0x0], SW}
39
                     sll $s1, $s1, 0x2
40
                     sw $s1, 0x0 ($a2)
                                             # Align SW[0x15:0x0] with LED && choose
                     counter0
41
                     addi $s2, $zero, 0x7fff # reset counter0 init value
42
                     sw $s2, 0x0 ($a3)
43
                                             # 128 nop, to ensure that counter0 has reset.
                     nop
44
                     nop
45
                     nop
46
                     nop
47
                     nop
48
                     nop
49
                     nop
50
                     nop
51
                     nop
52
                     nop
53
                     nop
54
                     nop
55
                     nop
56
                     nop
57
                     nop
58
                     nop
59
                     nop
60
                     nop
61
                     nop
62
                     nop
63
                     nop
64
                     nop
65
                     nop
66
                     nop
```

67	nop
68 69	nop nop
70	nop
71	nop
72	nop
73 74	nop
75	nop nop
76	nop
77	nop
78 79	nop
80	nop nop
81	nop
82	nop
83 84	nop
85	nop nop
86	nop
87	nop
88 89	nop
90	nop nop
91	nop
92	nop
93 94	nop
95	nop nop
96	nop
97	nop
98 99	nop
100	nop nop
101	nop
102	nop
103 104	nop nop
105	nop
106	nop
107	nop
108 109	nop nop
110	nop
111	nop
112 113	nop
114	nop nop
115	nop
116	nop
117 118	nop
119	nop nop
120	nop
121	nop
122 123	nop
124	nop nop
125	nop
126	nop
127 128	nop
129	nop nop
130	nop
131	nop
132 133	nop
134	nop nop
135	nop
136	nop
137 138	nop
139	nop nop
-	

```
140
141
                     nop
142
                     nop
143
                     nop
144
                     nop
145
                     nop
146
                     nop
147
                     nop
148
                     nop
149
                     nop
150
                     nop
151
                     nop
                     nop
153
                     nop
154
                     nop
155
                     nop
156
                     nop
157
                     nop
158
                     nop
159
                     nop
160
                     nop
161
                     nop
162
                     nop
163
                     nop
164
                     nop
165
                     nop
166
                     nop
167
                     nop
168
                     nop
169
                     nop
170
                     nop
171
                     eret
172
     Handle SYSCALL: addi $s7, $zero, 0x20
173
                     sll $s7, $s7, 0x2
                     add $v0, $zero, $zero
174
                                              # set $v0 to 0, it will be reused in
                     SYSCALL loop
                     add $k1, $zero, $zero
175
                                                # use k1 as a tmp cnt
176
                     lui $s6, 0x10
                                                # use $s6 as the tmp cnt's threshold
177
     Show_PicSet2:
                     addi $k1, $k1, 0x1
178
                     bne $k1, $s6, Show_PicSet2 #
                     用$k1进行计数,直到0x0010_0000时,才可以改变内存
                     add $k1, $zero, $zero # reset $k1 = 0, 计数用完,重新赋值为0
179
180
                     lw $k0, 0xA0($v0)
                                                # PicSet2 baseAddr 0xA0 ($v0 ==
                     0,使用的是RAM的地址)
181
                     sw $k0, 0x0($a1)
                                                # a1 ==
                     0xE000 0000, 将0x0000 00A0(对应要除以4, 也就是coe文件中的0x0000 0028)
                     处的值(0xffffffff7)放到Seg7里面
                     addi $v0, $v0, 0x4
182
                     bne $v0, $s7, Show PicSet2 # $s7 = 0x80, 用其进行计数, 总共0x80 /
183
                     4要循环32次
                     add $v0, $zero, $zero
184
                                               # reset $v0 to 0
185
                     add $s7, $zero, $zero
                     j Handle_EPCp4
186
     SYSCALL done:
                                               # 对于出现异常的指令一律不执行,跳过之
     Handle_UnInstr: j Handle_EPCp4
187
188
     Handle OV:
                     nop
189
     Handle EPCp4:
                     mfc0 $26, $14
190
                     addi $26, $26, 0x4
                     返回EPC+4处,说明Ov产生时,存入EPC的值必须是本条指令的PC地址,而不是PC
                     Plus4,而我在实现的时候是用ID EX REG的PCPlus4,要减8
191
                     mtc0 $26, $14
192
                     eret
193
                     nop
194
                     nop
195
                     add $a0, $zero, $zero
                                            # $a0 0x0000 0000 RAM
     Start:
196
                     lui $a1, 0xE000
                                            # $a1 0xE000_0000 SSeg7
                                           # $a2 0xF000_0000 Switch/LED (SPIO)
197
                     lui $a2, 0xF000
                                           # $a3 0xF000_0004 CounterX
198
                     ori $a3, $a2, 0x4
                     都是在准备地址
199
                     lui $at, OxFFFF
200
                     ori $at, $at, OxFFFF
                                            # $at = 0xFFFFFFF
201
                     addi $t9, $zero, 0x20
                                            # 32 (DEM) $t9 = 0x20
                     add $v1, $at, $zero
202
                     sll $v1, $v1, 0x1
                                            # $v1 = 0xFFFFFFE
203
```

```
# 关中断 设置Status位为0b1110,
204
                      addi $t0, $zero, 0xE
                      被设置为0的位是被屏蔽的Excp
205
                      mtc0 $t0, $13
206
                      lui $t0, 0x7FFF
207
                      ori $t0, $t0, OxFFFF
                                               # $t0 = 0x7FFFFFF
208
                      addi $t1, $zero, 0x2
                                              # overflow here
209
                      add $t0, $t0, $t1
210
                      break
                      addi $t0, $zero, 0x2AB # ...10101010 11, {GPIOf0[13:0], LED,
211
                      counter_set}
212
                      addi \$t\overline{1}, \$zero, 0x7fff # counter0 init val 0x00080000
213
                      sw $t0, 0x0($a2)
                                               # choose Ctrl_Reg, also set init_val of LED
214
                      sw \$zero, 0x0(\$a3)
                                               # write Ctrl_Reg, counter0 WorkMode = 2'b00
215
                      lw $t3, 0x0 ($a2)
                                               # $t3 = {counter0 out, counter1 out,
                      counter2_out, led_out[12:0], SW}
216
                      sll $t3, $t3, 0x2
                                               # Align SW[15:0] with LED && choose counter0
                      (srl makes $t3[1:0] = 2'b00)
217
                      sw $t3, 0x0($a2)
                      sw $t1, 0x0($a3)
218
                                               # write counter0 init value (== 0 \times 00080000)
219
                      addi $t0, $zero, 0xF
                                              # 开中断
220
                      mtc0 $t0, $13
221
                      lw $t0, 0x0($a2)
                                               # $t0 = {counter0 out, counter1 out,
     Loop:
      counter2 out, led out[12:0], SW}
222
                      sll $t0, $t0, 0x2
                                               # Align SW[15:0] with LED
223
                      sw $t0, 0x0($a2)
                      bne $v0, $t9, Loop
224
225
                      SYSCALL
226
                      add $v0, $zero, $zero # reset cnt $v0
227
                      j Loop
228
                      nop
229
                      nop
```