```
Entrance:       j Start
                nop
EXCINTHandler:  mfc0 $k0, $12           # $k0 <- CP0.$cause
                andi $k1, $k0, 0xc      # $k1 = EXcCode (cause[3:2])
                addi $s1, $zero, 0x4    # 0x0100, syscall
                addi $s2, $zero, 0x8    # 0x1000, UnInstr
                addi $s3, $zero, 0xC    # 0x1100, OV
                beq $k1, $s1, Handle_SYSCALL
                beq $k1, $s2, Handle_UnInstr
                beq $k1, $s3, Handle_OV
Handle_INT:     sll $v1, $v1, 0x1
                ori $v1, $v1, 0x1       # 循环右移，每次在最低位补 1
                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 个数据一组，所以用 6 位 mask (4 + 2，地址最低两位恒为 2'b00)
                addi $v0, $v0, 0x1      # increase $v0 for SYSCALL
                bne $v0, $at, Disp      # if $v0 = 0xFFFFFFFF, reset it to 0x5 (This step is useless in this program, since $v0 [0~32])
                addi $v0, $zero, 0x5
Disp:           addi $s1, $zero, 0x8    # 5'b01000, SW[4:3]=2'b01 && SW[0]=1
                addi $s2, $zero, 0x10   # 5'b10000, SW[4:3]=2'b10 && SW[0]=1
                addi $s3, $zero, 0x18   # 5'b11000, SW[4:3]=2'b11 && SW[0]=0
                lw $s5, 0x0($a2)
                andi $s5, $s5, 0x18     # 0x18 = 5'b11000, mask to get SW[4:3]
                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     # SW[4:3]=2'b01 (&& SW[0]=0), 0x00000000 -> 0x11111111 -> ... -> 0xFFFFFFFF
                beq $s5, $s2, SW_10     # SW[4:3]=2'b10 (&& SW[0]=0), show cycle accumulation of $v0
                beq $s5, $s3, SW_11     # SW[4:3]=2'b11 (&& SW[0]=0), show pictures
SW_00:          bne $v1, $at, L3        # if ($v1 = 0xFFFFFFFF)
                sll $v1, $v1, 0x1       #       $v1 <<= 0x1 // $v1 = 0xFFFFFFFE
L3:             sw  $v1, 0x0($a1)       # else
                j Disp_done             #       // show $t0 on SSeg7
SW_01:          lw $k0, 0x20($fp)       # 显示预置数字
                sw $k0, 0x0($a1)
                j Disp_done
SW_10:          sw $v0, 0x0($a1)        # 显示 $v0 (累加)
                j Disp_done
SW_11:          lw $k0, 0x60($fp)       # show PictureSet1
                sw $k0, 0x0($a1)
Disp_done:      lw $s1, 0x0($a2)        # $s1 = {counte$0_out, counte$1_out, counte$2_out, led_out[0x12:0x0], SW}
                sll $s1, $s1, 0x2
                sw $s1, 0x0($a2)        # Align SW[0x15:0x0] with LED && choose counter0
                addi $s2, $zero, 0x7fff # reset counter0 init_value
                sw $s2, 0x0($a3)
                nop                     # 128 nop, to ensure that counter0 has reset.
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                nop
                eret
Handle_SYSCALL: addi $s7, $zero, 0x20
                sll $s7, $s7, 0x2
                add $v0, $zero, $zero       # set $v0 to 0, it will be reused in SYSCALL loop
                add $k1, $zero, $zero       # use k1 as a tmp_cnt
                lui $s6, 0x10               # use $s6 as the tmp_cnt's threshold
Show_PicSet2:   addi $k1, $k1, 0x1
                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
                lw $k0, 0xA0($v0)           # PicSet2 baseAddr 0xA0($v0 == 0，使用的是RAM的地址)
                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(0xFFFFFFF7)放到Seg7里面
                addi $v0, $v0, 0x4
                bne $v0, $s7, Show_PicSet2  # $s7 = 0x80，用其进行计数，总共0x80 / 4要循环32次
                add $v0, $zero, $zero       # reset $v0 to 0
                add $s7, $zero, $zero
SYSCALL_done:   j Handle_EPCp4
Handle_UnInstr: j Handle_EPCp4              # 对于出现异常的指令一律不执行，跳过之
Handle_OV:      nop
Handle_EPCp4:   mfc0 $26, $14
                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PCPlus4，而我在实现的时候是用ID_EX_REG的PCPlus4，要减8
                mtc0 $26, $14
                eret
                nop
                nop
Start:          add $a0, $zero, $zero     # $a0 0x0000_0000 RAM
                lui $a1, 0xE000           # $a1 0xE000_0000 SSeg7
                lui $a2, 0xF000           # $a2 0xF000_0000 Switch/LED (SPIO)
                ori $a3, $a2, 0x4         # $a3 0xF000_0004 CounterX 都是在准备地址
                lui $at, 0xFFFF
                ori $at, $at, 0xFFFF      # $at = 0xFFFFFFFF
                addi $t9, $zero, 0x20     # 32(DEM) $t9 = 0x20
                add $v1, $at, $zero
                sll $v1, $v1, 0x1         # $v1 = 0xFFFFFFFE
                addi $t0, $zero, 0xE    # 关中断 设置Status位为0b1110,被设置为0的位是被屏蔽的Excp
                mtc0 $t0, $13
                lui  $t0, 0x7FFF
                ori  $t0, $t0, 0xFFFF   # $t0 = 0x7FFFFFFF
                addi $t1, $zero, 0x2
                add  $t0, $t0, $t1      # overflow here
                break
                addi $t0, $zero, 0x2AB  # ...10101010_11, {GPIOf0[13:0], LED, counter_set}
                addi $t1, $zero, 0x7fff # counter0 init val 0x00080000
                sw $t0, 0x0($a2)        # choose Ctrl_Reg, also set init_val of LED
                sw $zero, 0x0($a3)      # write Ctrl_Reg, counter0 WorkMode = 2'b00
                lw $t3, 0x0($a2)        # $t3 = {counter0_out, counter1_out, counter2_out, led_out[12:0], SW}
                sll $t3, $t3, 0x2       # Align SW[15:0] with LED && choose counter0 (srl makes $t3[1:0] = 2'b00)
                sw  $t3, 0x0($a2)
                sw  $t1, 0x0($a3)       # write counter0 init value (== 0x00080000)
                addi $t0, $zero, 0xF    # 开中断
                mtc0 $t0, $13
Loop:           lw  $t0, 0x0($a2)       # $t0 = {counter0_out, counter1_out, counter2_out, led_out[12:0], SW}
                sll $t0, $t0, 0x2       # Align SW[15:0] with LED
                sw $t0, 0x0($a2)
                bne $v0, $t9, Loop
                SYSCALL
                add $v0, $zero, $zero   # reset cnt $v0
                j Loop
                nop
                nop
```