프로젝트 1 검증 문서 2020203090 한용옥

프로그램 실행 방법

1. 모든 `.cpp`, `.h` 파일을 한 경로에 넣는다
2. `main.cpp` 를 연다
3. 전역 변수 `file_path` 에 인코딩할 `.asm` 파일의 경로를 적는다
4. 전역 변수 `out_path` 에 인코딩된 파일의 경로를 적는다
5. `main.cpp` 를 실행한다

프로그램 모드 설정

`main.cpp` 의 전역 변수 `debug` 가
`true` 면 콘솔에 토큰화된 명령어, 라벨 테이블, 각 명령어, PC, 인코딩된 명령어가 출력, `.asm` 을 읽어 `.bin` 으로 인코딩
`false` 면 아무것도 뜨지 않은 채 `.asm` 을 읽어 `.bin` 으로 인코딩

`.asm` 설정

`.asm` 입력 형식 오류는 가정하지 않음

1. 라벨은 공백없이 `:` 로 마무리
2. 모든 토큰은 공백으로 분리
3. `#` 뒤는 모두 무시됨(주석이므로)

테스트한 파일 및 인코딩 결과

test.asm

```
fib:
    addi $sp, $sp, -8        # 스택 공간 확보
    sw   $ra, 4($sp)         # 복귀 주소 저장
    sw   $a0, 0($sp)         # n 저장

    slti $t0, $a0, 2         # if (n < 2)
    bne  $t0, $zero, base    # return n

    addi $a0, $a0, -1        # a0 = n - 1
    jal  fib                 # fib(n-1) 호출
    lw   $a0, 0($sp)         # a0 = n (복원)
    add  $t1, $v0, $zero     # t1 = fib(n-1)

    addi $a0, $a0, -2        # a0 = n - 2
    jal  fib                 # fib(n-2) 호출
    add  $v0, $v0, $t1       # fib(n) = fib(n-1) + fib(n-2)

    j    end                 # 종료로 점프

base:
    add  $v0, $a0, $zero     # v0 = n (0 or 1일 때)

end:
    lw   $a0, 0($sp)         # a0 복원
    lw   $ra, 4($sp)         # ra 복원
    addi $sp, $sp, 8         # 스택 해제
    jr   $ra                 # 복귀
```

test.bin (= 인코딩 결과)

```
00100011101111011111111111111000
10101111101111110000000000000100
10101111101001000000000000000000
00101000100010000000000000000010
00010101000000000000000000001001
00100000100001001111111111111111
00001100000000000000000000000000
10001111101001000000000000000000
00000000010000000100100000100000
00100000100001001111111111111110
00001100000000000000000000000000
00000000010010010001000000100000
00001000000000000000000000001110
00000000100000000001000000100000
10001111101001000000000000000000
10001111101111110000000000000100
00100011101111010000000000001000
00000011111000000000000000001000
```

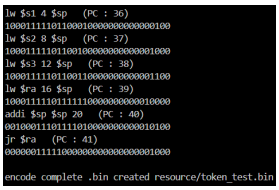
`debug==true` 일 때의 콘솔 출력

```
parsed instruction
sll $t1 $a1 2
add $t1 $a0 $t1
lw $t0 0 $t1
lw $t2 4 $t1
sw $t2 0 $t1
sw $t0 4 $t1
jr $ra
addi $sp $sp -20
sw $ra 16 $sp
sw $s3 12 $sp
sw $s2 8 $sp
sw $s1 4 $sp
sw $s0 0 $sp
add $s2 $a0 $zero
add $s3 $a1 $zero
add $s0 $zero $zero
slt $t0 $s0 $s3
beq $t0 $zero exit1
addi $s1 $s0 -1
slti $t0 $s1 0
bne $t0 $zero exit2
sll $t1 $s1 2
add $t2 $s2 $t1
lw $t3 0 $t2
lw $t4 4 $t2
slt $t0 $t4 $t3
beq $t0 $zero exit2
add $a0 $s2 $zero
add $a1 $s1 $zero
jal swap
addi $s1 $s1 -1
j for2tst
addi $s0 $s0 1
j for1tst
lw $s0 0 $sp
lw $s1 4 $sp
lw $s2 8 $sp
lw $s3 12 $sp
lw $ra 16 $sp
addi $sp $sp 20
jr $ra
```

```
.asm label_table
label PC <- table format
exit2 32
exit1 34
for2tst 19
for1tst 16
sort 7
swap 0

ins to bin, PC is increased before encoded
sll $t1 $a1 2   (PC : 1)
00000000000001010100100010000000
add $t1 $a0 $t1   (PC : 2)
00000000100010010100100000100000
lw $t0 0 $t1   (PC : 3)
10001101001010000000000000000000
lw $t2 4 $t1   (PC : 4)
10001101001010100000000000000100
sw $t2 0 $t1   (PC : 5)
10101101001010100000000000000000
sw $t0 4 $t1   (PC : 6)
10101101001010000000000000000100
jr $ra   (PC : 7)
00000011111000000000000000001000
addi $sp $sp -20   (PC : 8)
00100011101111011111111111101100
sw $ra 16 $sp   (PC : 9)
10101111101111110000000000010000
sw $s3 12 $sp   (PC : 10)
10101111101100110000000000001100
sw $s2 8 $sp   (PC : 11)
10101111101100100000000000001000
sw $s1 4 $sp   (PC : 12)
10101111101100010000000000000100
sw $s0 0 $sp   (PC : 13)
10101111101100000000000000000000
add $s2 $a0 $zero   (PC : 14)
00000000100000001001000000100000
add $s3 $a1 $zero   (PC : 15)
00000000101000001001100000100000
add $s0 $zero $zero   (PC : 16)
00000000000000001000000000100000
```

```
slt $t0 $s0 $s3   (PC : 17)
00000010000100110100000000101010
beq $t0 $zero exit1   (PC : 18)
00010001000000000000000000010001
addi $s1 $s0 -1   (PC : 19)
00100010000100011111111111111111
slti $t0 $s1 0   (PC : 20)
00101010001010000000000000000000
bne $t0 $zero exit2   (PC : 21)
00010101000000000000000000001100
sll $t1 $s1 2   (PC : 22)
00000000000010001010010001000000
add $t2 $s2 $t1   (PC : 23)
00000010010010010101000000100000
lw $t3 0 $t2   (PC : 24)
10001101010010110000000000000000
lw $t4 4 $t2   (PC : 25)
10001101010011000000000000000100
slt $t0 $t4 $t3   (PC : 26)
00000001100010110100000000101010
beq $t0 $zero exit2   (PC : 27)
00010001000000000000000000000110
add $a0 $s2 $zero   (PC : 28)
00000010010000000010000000100000
add $a1 $s1 $zero   (PC : 29)
00000010001000000010100000100000
jal swap   (PC : 30)
00001100000000000000000000000000
addi $s1 $s1 -1   (PC : 31)
00100010001100011111111111111111
j for2tst   (PC : 32)
00001000000000000000000000010011
addi $s0 $s0 1   (PC : 33)
00100010000100000000000000000001
j for1tst   (PC : 34)
00001000000000000000000000010000
lw $s0 0 $sp   (PC : 35)
10001111101100000000000000000000
```


lw $s1 4 $sp   (PC : 36)
10001111101100010000000000000100
lw $s2 8 $sp   (PC : 37)
10001111101100100000000000001000
lw $s3 12 $sp   (PC : 38)
10001111101100110000000000001100
lw $ra 16 $sp   (PC : 39)
10001111101111110000000000010000
addi $sp $sp 20   (PC : 40)
00100011101111010000000000010100
jr $ra   (PC : 41)
00000011111000000000000000001000
encode complete .bin created resource/token_test.bin