SYDNEY CADY

2. 1, 2, 3, 4, 5, 6, 7, 9, 10, 12, 18, 21, 22, 26, 29, 38, 47

2.1

f = g + (h-5)

```
subi  f, h, 5
add   f, f, g
```

2.2

```
add  f, g, h
add  f, i, f
```

} f = g+h+i

2.3

B[8] = A[i-j]

f, g, h, i, j → \$s0, \$s1, \$s2, \$s3, \$s4
A, B → \$s6, \$s7

```
sub  $t0, $s3, $s4
add  $t0, $s6, $t0
lw   $t1, 16($t0)
sw   $t1, 32($s7)
```

2.4

B[g] = A[f] + A[1+f]

2.5

memimize

```
add  $t0, $s6, $s0
add  $t1, $s7, $s1
lw   $s0, 0($t0)
lw   $t0, 4($t0)
add  $t0, $t0, $s0
sw   $t0, 0($t1)
```

2.6

Add.	Data	
24	2	0
36	4	1
32	3	2
36	6	3
40	1	4

2.6.1

Tmp = Array[0]
tmp2 = Array[1]
array[0] = array[4]
array[1] = temp
array[4] = array[3]
array[3] = temp2

?

2.6.2

```
lw  $t0, 0($s6)
lw  $t1, 4($s6)
lw  $t2, 16($s6)
sw  $t2, 0($s6)
sw  $t0, 4($s6)
lw  $t0, 12($s6)
sw  $t0, 16($s6)
sw  $t1, 12($s6)
```

2.7

Lil-Endian Address	Data	Big Endian Address	Data
12	ab	12	12
8	cd	8	ef
4	ef	4	cd
0	12	0	ab

2.9

B[8] = A[i] + A[j]

?

```
sll  $t0, $s1, 2        # t0 = 4g
add  $t0, $t0, $s7      # t0 = Addr[B[g]]
lw   $t0, 0($t0)        # t0 = B[g]
addi $t0, $t0, 1        # t0 = B[g] + 1
sll  $t0, $t0, 2        # t0 = 4(B[g]+1)
lw   $s0, 0($t0)        # f = A[B[g]+1]
```

2.10

```
# t0 = A[1]
# t1 = A[0]
# $t1 = A[1]
# $t0 = & A[1]
# f = t1 + t0
```

$\boxed{2\,[A]}$

2.12 $\$s0 = 0x8000\,0000$ $\$s1$ 0xD000 0000

2.12.1

$\$t0 = \$s0 + \$s1 = 5000\,0000$

2.12.2

overflow

2.12.3

A B C D
10 11 12 13

$s1 - s0 = $ ~~8~~
0x5000 0000

2.12.4

no

2.12.5

$t0 = (\$s0 + s1) + \$s0) = D0000000$

2.12.6

yes overflow

2.18

MIP → 128 $\log_2 128 = 7$

2.18.1

R → rs, rt, rd = 7 bits
opcode = 8
leaving 5

2.18.2

op=8
rs, rt = 7
IMM/10 for address

12.58, 3

more reg → so immidate data could be reduce it can have higher reg lookup plus instr. count requres neure bits to be stored.

↑ instr → would need ↑ reg

2.21

not $t1, $t2 ⟶ Nor $t1, $t2, $t2

2.22 ?

A = C[0] << 4

```
lw   $t3, 0($s1)
sll  $t1, $t3, 4
```

2.26

2.26.1

$t1 = 10
$s2 = 0 } $s2 = 20

2.26.2

```
addi  $t2, $t2, -1
beq   $t2, $0, loop
```

2.26.3

5×N ?

2.29

$\$t1 = i$

$\$s2 =$ resut

$\$s0 =$ Memarray

```
for (i=0; i<100; i++)
{
    result += MemArray[s0];
    s0 = s0 + 4;
}
```

2.38

```
lbu   $t0, 0($t1)
sw    $t0, 0($t2)
```

$t1 = 0x1000 0000
$t2 = 0x1000 0010

@ 0x1000 0000 = 0x 11223344

@ $t2 = 0x 0000 0011 ?

2.47

70% artmetic
10% load/store
20% branch

2.47.1

arith: 2 cycles
load/store: 6 cycles
branch: 3 cycles

$$\frac{(70\times2)+(6\times10)+(3\times20)}{100} = 2.6$$

2.47.2

25% improvement performance, load/store 4, branch 0

?

$$\frac{1.25(70\times2)+60+60}{100} = 1.07$$

2.47.3

50% ↑ perf, load/store i, branch ∅

!

$$\frac{0.5(70.2) + 60 + 60}{100} = 0.14$$