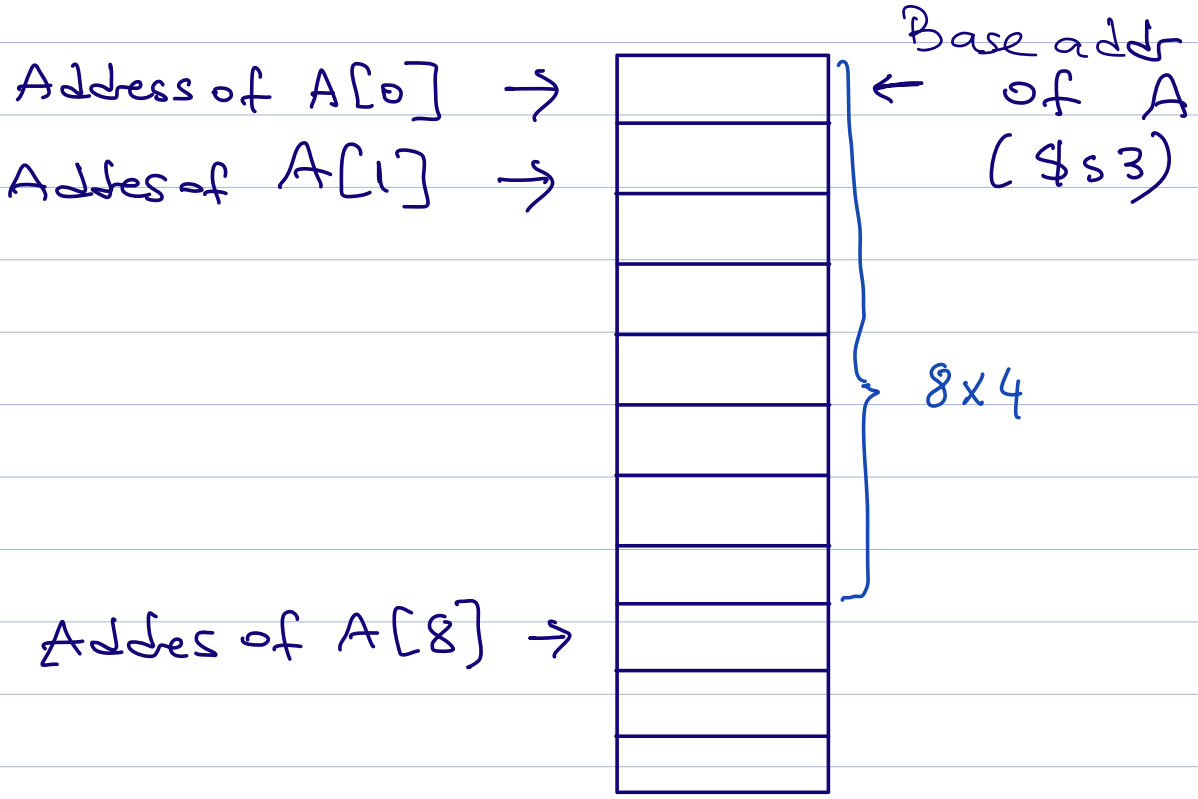
Instructions (Slide 22)



Addes of A[8] = Base addr of A + offset

= $s3 + 8 x 4

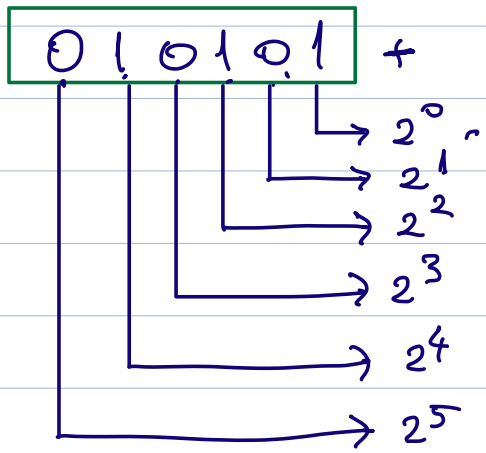
= $s3 + 32

A[8] Addr

Addr of A
Addr of A[0]

Addr → $s3+32

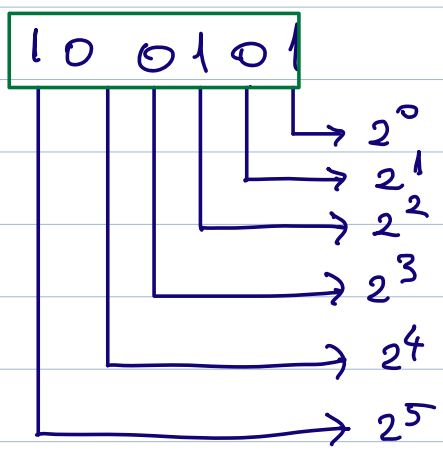
Unsigned Binary Integers (slide 27)



$$010101 = 0\times2^5 + 1\times2^4 + 0\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0$$

$$= 16 + 4 + 1 = 21$$

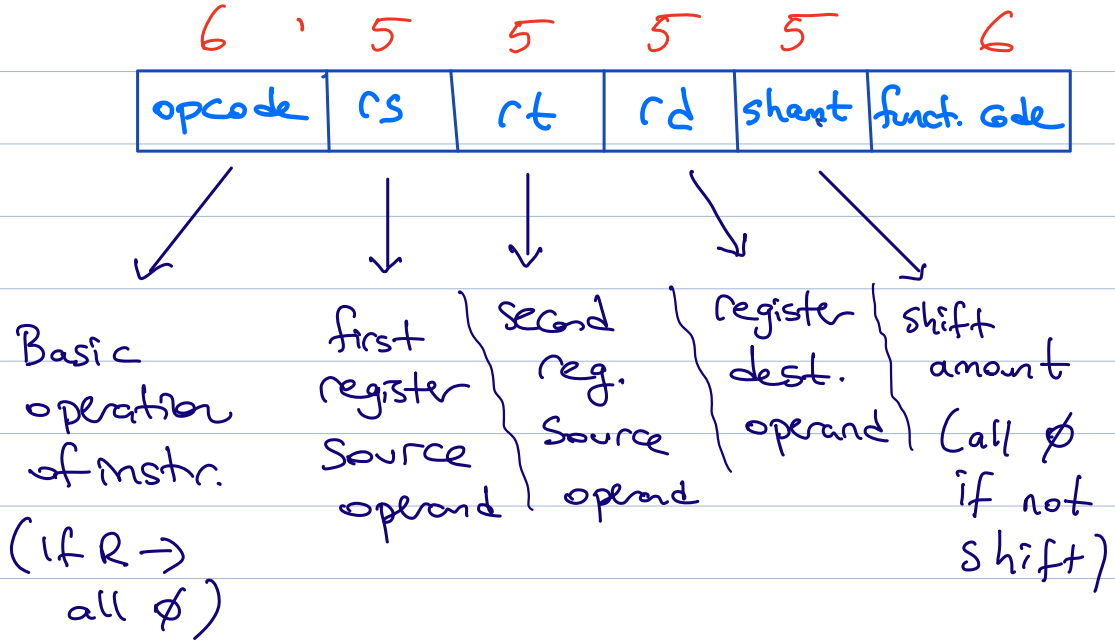
2's Complement Signed Integers



$$100101 = (-1)\times2^5 + 0\times2^4 + 0\times2^3 + 1\times2^2 + 0\times2^1 + 1\times2^0$$

$$= -32 + 4 + 1 = -27$$

R-Format



`add $s1,$s2,$s3` Select specific variant of the operation

Name	Register Number	Usage	Preserve on call?
$zero	0	constant 0	n.a.
$at	1	reserved for assembler	n.a.
$v0 - $v1	2-3	returned values	no
$a0 - $a3	4-7	arguments	yes
$t0 - $t7	8-15	temporaries	no
$s0 - $s7	16-23	saved values	yes
$t8 - $t9	24-25	temporaries	no
$gp	28	global pointer	yes
$sp	29	stack pointer	yes
$fp	30	frame pointer	yes
$ra	31	return addr	yes

I-format

rs : base register (lw & sw)

rt : destination register for lw

load / store any word within a region of $\pm 2^{15}$ bytes of base register.

lw $S2, 20($S3)
sw $S4, 40($S5)

$S2 ← Memory[20 + $S3]
Memory[40 + $S5] ← $S4

R-type opcode = ∅
I-type

lw : op = 35

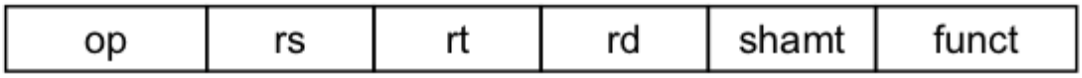
sw : op: 43

Shift Operation

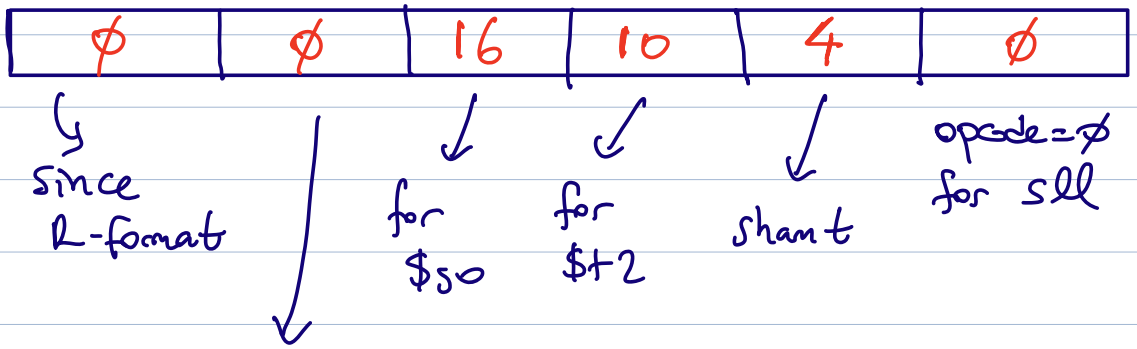
sll & srl ⇒ R-format

opcode = ∅

function code → sll = ∅
function code → srl = 2



Example: sll $t2, $s0, 4



and : R-format (function code = 36)

or : R-format (function code = 37)

nor : R-format (function code = 39)

* MIPS does not have NOT operation (1 → 0 & 0 → 1)

a NOR b = NOT (a OR b)

a NOR 0 = NOT (a OR 0)

= NOT (a)

$zero register

Example: (Page 84 - textbook)

Assume that $t1 has base address of array A and $s2 corresponds to h. The assignment give below:

```
A[300] = h + A[300];
```

is compiled into:

```
lw   $t0, 1200($t1)   (I)
add  $t0, $s2, $t0    (R)
sw   $t0  1200($t1)   (I)
```

What is the machine language code?

6 5 5 16

op	rs	rt	rd	shamt	funct.
			Constant / Add		
35	9	8	1200		
0	18	8	8	0	32
43	9	8	1200		

(lw) opcode = 35

base register 9 ($t1) dest reg: 8 ($t0)

Offset to select A[300] (300 x 4 = 1200)

sw opcode = 43

base register 9 ($t1)

add opcode = 0 funct code = 32

$s2, $t0

18 8

For Constants

→ andi $s1, $2, 100 ⇒ $s1 = $s2 & 100

12	18	17	100

(I-type)

opcode = 12

→ and $s1, $s2, $s3 ⇒ $s1 = $s2 & $s3

0	18	19	17	0	36

(R-type)

opcode = 0 funct code = 36

ori ⇒ opcode = 13

or ⇒ opcode = 0 funct code = 37

BEQ & BNE instructions

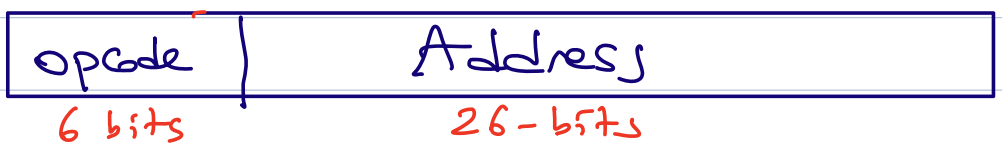
beq $s1, $s2, Label (If $s1 == $s2) goto Label

bne $s1, $s2, Label (If $s1 ≠ $s2) goto Label

beq & bne = I-type

↙ op=4 ↘ op=5

J-type



Alternate Code for Slide 44 (beq in place of bne)

```
      beq  $s3, $s4, If_Part
      sub  $s0, $s1, $s2
      j    Exit
If_Part: add  $s0, $s1, $s2
Exit:
```

Slide 46

while (save[i] == k) i += 1;

i → $s3 Base Addr → $s6

k → $s5

Store "save[i]" in a register

Address of save[i] = Address of array save + offset

Address of save[i] = $s6 + 4 × i

lw instr $t0 ← Memory [$s6 + 4×i]

If no mult ⇒ sll by 2 bits

SLT Instruction

slt $s1, $s2, $s3 → R-format

If ($s2 < $s3) then $s1 = 1

otherwise $s1 = 0

I-format

slti $s1, $s2, constant

if ($s2 < constant) then $s1 = 1
otherwise $s1 = 0

BLT Instruction

blt $s1, $s2, label → pseudo instruction

if ($s1 < $s2) go to label

Pseudo instructions: Assembly language instructions that do not have a direct machine language equivalent.

During assembly → assembler translates each pseudo instruction into one or more machine language instructions.

How to perform blt ?

```
slt  $to, $s1, $s2
bne  $to, $zero, L
```

(If $s1 < $s2, $to = 1)

$to : either $\emptyset$ or $\perp$.

$s1 = $s2 or $s1 > $s2

$s1 < $s2