a

MIPS instructions

op1, op2 are registers, op3 is register or constant
cont[op1] means contents of op1

move op1, op2	cont[op1] = cont[op2]
add op1, op2, op3	cont[op1] = cont[op2] + cont[op3]
sub op1, op2, op3	cont[op1] = cont[op2] - cont[op3]
mul op1, op2, op3	cont[op1] = cont[op2] * cont[op3]
div op1, op2, op3	cont[op1] = cont[op2] / cont[op3]
rem op1, op2, op3	cont[op1] = cont[op2] % cont[op3]
not op1, op2	cont[op1] = not cont[op2] (bitwise)
and op1, op2, op3	cont[op1] = cont[op2] and cont[op3] (bitwise)
or op1, op2, op3	cont[op1] = cont[op2] or cont[op3] (bitwise)
nand op1, op2, op3	cont[op1] = cont[op2] nand cont[op3] (bitwise)
nor op1, op2, op3	cont[op1] = cont[op2] nor cont[op3] (bitwise)
xor op1, op2, op3	cont[op1] = cont[op2] xor cont[op3] (bitwise)
sll op1, op2, AMT	cont[op1] = cont[op2] shift left logical by AMT bits
srl op1, op2, AMT	cont[op1] = cont[op2] shift right logical by AMT bits
sra op1, op2, AMT	cont[op1] = cont[op2] shift right arithmetic by AMT bits
rol op1, op2, AMT	cont[op1] = cont[op2] rotate left by AMT bits
ror op1, op2, AMT	cont[op1] = cont[op2] rotate right by AMT bits

b label	goto label
j label	goto label
beq op1, op2, label	if (cont[op1]==cont[op2]) goto label
bne op1, op2, label	if (cont[op1]!=cont[op2]) goto label
bgt op1, op2, label	if (cont[op1]>cont[op2]) goto label
bge op1, op2, label	if (cont[op1]>=cont[op2]) goto label
blt op1, op2, label	if (cont[op1]<cont[op2]) goto label
ble op1, op2, label	if (cont[op1]<=cont[op2]) goto label
beqz op1, label	if (cont[op1]==0) goto label
bnez op1, label	if (cont[op1]!=0) goto label
bgtz op1, label	if (cont[op1]>0) goto label
bgez op1, label	if (cont[op1]>=0) goto label
bltz op1, label	if (cont[op1]<0) goto label
blez op1, label	if (cont[op1]<=0) goto label

la R, label	cont[R] = address of label
li R, constant	cont[R] = constant
lw R, ??	cont[R] = M[ADDR]
lb R, ??	cont[R] = m[ADDR], sign-extended
lbu R, ??	cont[R] = m[ADDR], zero-extended
sw R, ??	M[ADDR] = cont[R]
sb R, ??	m[ADDR] = low 8-bits of cont[R]

if ?? is a label, ADDR = address of label
if ?? is (R), ADDR = cont[R]
if ?? is constant(R), ADDR = cont[R] + constant
if ?? is label(R), ADDR = cont[R] + address of label

mtc0 op1, op2	contents of coprocessor 0 register op1 = contents of MIPS register op2
mfc0 op1, op2	contents of MIPS register op1 = contents of coprocessor 0 register op2

Syscall Usage

print an int	$v0=1, $a0=int to be printed
print a string	$v0=4, $a0=address of string to be printed
print a char	$v0=11 $a0=ascii # of character to print
read an int	$v0=5, input int appears in $v0
exit	$v0=10

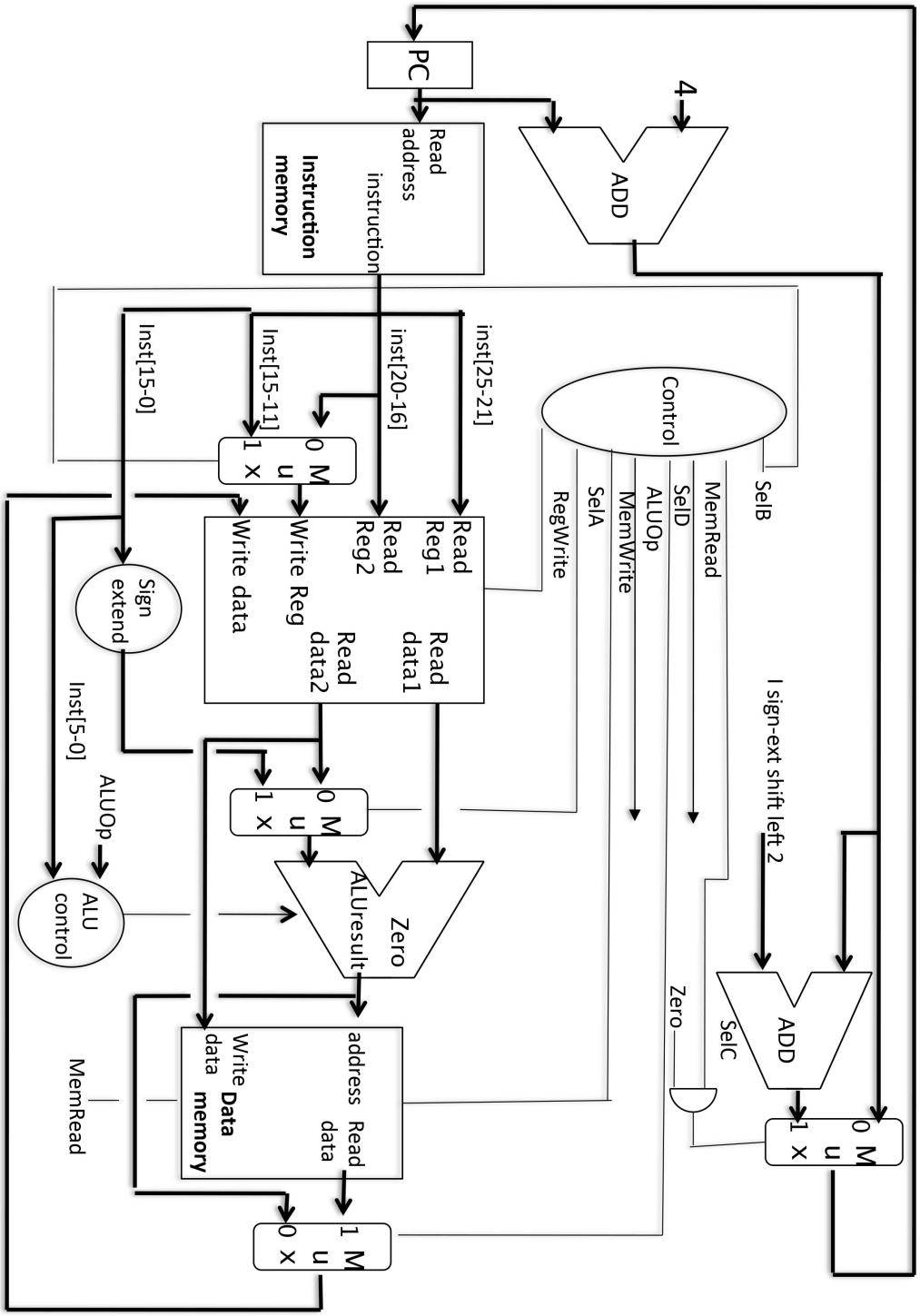
MIPS Register Names

$0	
$1	
$2,$3	$v0,$v1
$4 - $7	$a0 - $a3
$8 - $15	$t0 - $t7
$16 - $23	$s0 - $s7
$24 - $25	$t8 - $t9
$26 - $27	$k0 - $k1
$28	$gp
$29	$sp
$30	$s8
$31	$ra

a

```
0000 00ss ssst tttt dddd d000 0010 0000       add rd,rs,rt
0000 00ss ssst tttt dddd d000 0010 0010       sub rd,rs,rt
0000 00ss ssst tttt 0000 0000 0001 1000       mult rs,rt
0000 00ss ssst tttt 0000 0000 0001 1010       div rs,rt

0000 00ss ssst tttt dddd d000 0010 0001       addu rd,rs,rt
0000 00ss ssst tttt dddd d000 0010 0011       subu rd,rs,rt
0000 00ss ssst tttt 0000 0000 0001 1001       multu rs,rt
0000 00ss ssst tttt 0000 0000 0001 1011       divu rs,rt

0000 0000 0000 0000 dddd d000 0001 0000       mfhi rd
0000 00ss sss0 0000 0000 0000 0001 0001       mthi rs
0000 0000 0000 0000 dddd d000 0001 0010       mflo rd
0000 00ss sss0 0000 0000 0000 0001 0011       mtlo rs

0000 00ss ssst tttt dddd d000 0010 0100       and rd,rs,rt
0000 00ss ssst tttt dddd d000 0010 0111       nor rd,rs,rt
0000 00ss ssst tttt dddd d000 0010 0101       or rd,rs,rt
0000 00ss ssst tttt dddd d000 0010 0110       xor rd,rs,rt

0000 00ss ssst tttt dddd d000 0000 0100       sllv rd,rt,rs
0000 00ss ssst tttt dddd d000 0000 0110       srlv rd,rt,rs
0000 00ss ssst tttt dddd d000 0000 0111       srav rd,rt,rs

0010 00ss ssst tttt iiii iiii iiii iiii       addi rt,rs,I
0010 01ss ssst tttt iiii iiii iiii iiii       addiu rt,rs,I
0011 00ss ssst tttt iiii iiii iiii iiii       andi rt,rs,I
0011 1100 000t tttt iiii iiii iiii iiii       lui rt,I
0011 01ss ssst tttt iiii iiii iiii iii        ori rt,rs,I
0011 10ss ssst tttt iiii iiii iiii iiii       xori rt,rs,I

0000 0000 000t tttt dddd diii ii00 0000       sll rd,rt,I
0000 0000 000t tttt dddd diii ii00 0010       srl rd,rt,I
0000 0000 000t tttt dddd diii ii00 0011       sra rd,rt,I

1000 11bb bbbt tttt iiii iiii iiii iiii       lw rt,I(rb)
1000 00bb bbbt tttt iiii iiii iiii iiii       lb rt,I(rb)
1001 00bb bbbt tttt iiii iiii iiii iiii       lbu rt,I(rb)
1010 11bb bbbt tttt iiii iiii iiii iiii       sw rt,I(rb)
1010 00bb bbbt tttt iiii iiii iiii iiii       sb rt,I(rb)

0000 01ss sss0 0000 iiii iiii iiii iiii       bltz rs,I
0000 01ss sss0 0001 iiii iiii iiii iiii       bgez rs,I
0001 10ss sss0 0000 iiii iiii iiii iiii       blez rs,I
0001 11ss sss0 0000 iiii iiii iiii iiii       bgtz rs,I

0001 00ss ssst tttt iiii iiii iiii iiii       beq rs,rt,I
0001 01ss ssst tttt iiii iiii iiii iiii       bne rs,rt,I

0000 00ss ssst tttt dddd d000 0010 1010       slt rd,rs,rt
0010 10ss ssst tttt iiii iiii iiii iiii       slti rt,rs,I

0000 10ii iiii iiii iiii iiii iiii iiii       j I
0000 00ss sss0 0000 0000 0000 0000 1000       jr rs
0000 11ii iiii iiii iiii iiii iiii iiii       jal I
0000 00ss sss0 0000 dddd d000 0000 1001       jalr rd,rs

0000 0000 0000 0000 0000 0000 0000 1100       syscall
```

B



ALU OP

Function	ALU Op 1	ALU Op 0
Forde Add	0	0
Force Sub	0	1
Do Func	1	0