CS2100 Reference Sheet

Core Instruction Set

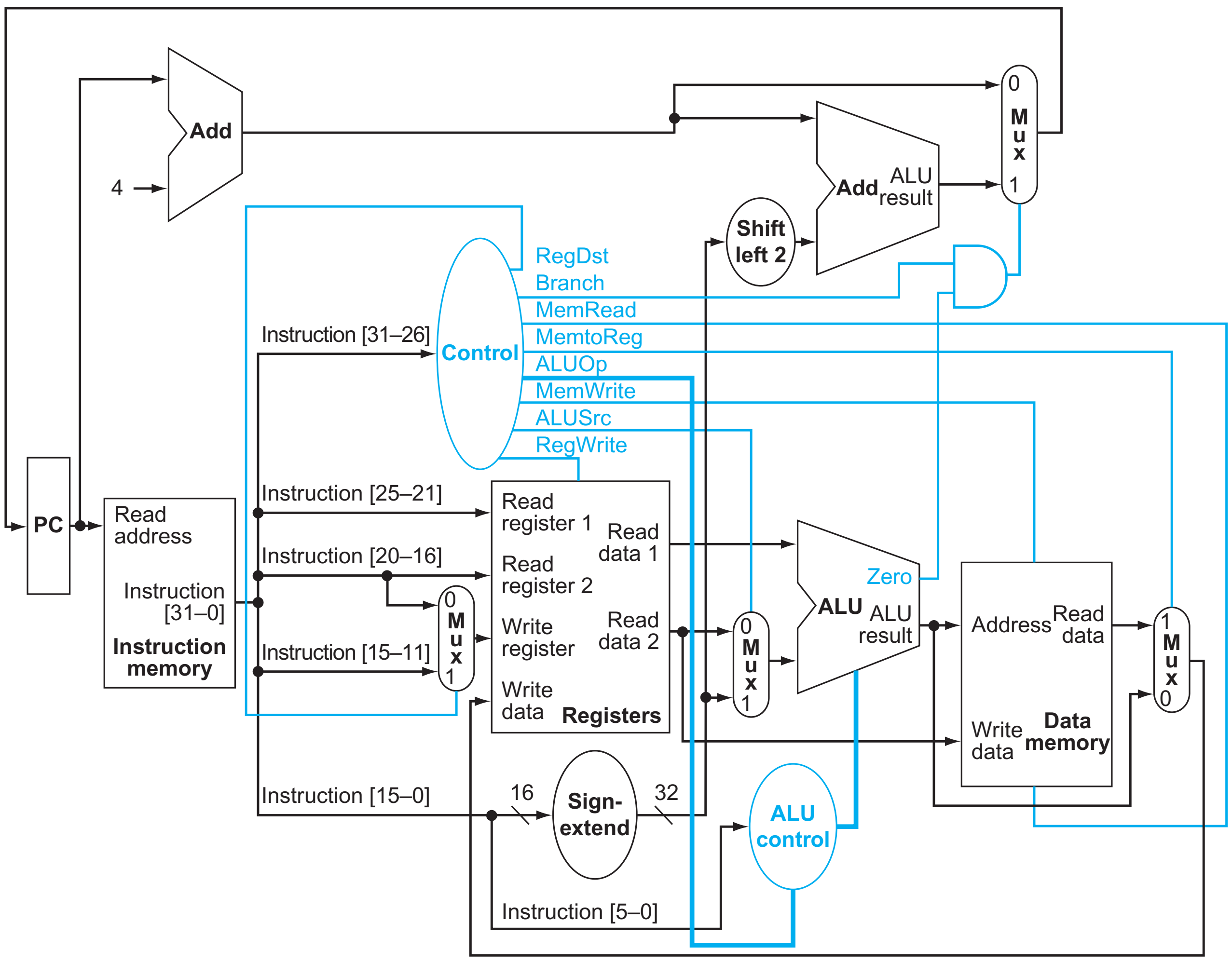
Name	Mnemonic	Operands	Fmt	Opcode/Funct	Operation
Add	`add`	`rd, rs, rt`	R	0/0x20	R[rd] = R[rs] + R[rt] [a]
Add Imm.	`addi`	`rt, rs, imm`	I	0x08	R[rt] = R[rs] + SignExtImm [ab]
Subtract	`sub`	`rd, rs, rt`	R	0/0x22	R[rd] = R[rs] - R[rt] [a]
And	`and`	`rd, rs, rt`	R	0/0x24	R[rd] = R[rs] & R[rt]
And Imm.	`andi`	`rt, rs, imm`	I	0x0c	R[rt] = R[rs] & ZeroExtImm [c]
Or	`or`	`rd, rs, rt`	R	0/0x25	R[rd] = R[rs] | R[rt]
Or Imm.	`ori`	`rt, rs, imm`	I	0x0d	R[rt] = R[rs] | ZeroExtImm [c]
Exclusive-Or	`xor`	`rd, rs, rt`	R	0/0x26	R[rd] = R[rs] ^ R[rt]
Exclusive-Or Imm.	`xori`	`rt, rs, imm`	I	0x0e	R[rt] = R[rs] ^ ZeroExtImm [c]
Nor	`nor`	`rd, rs, rt`	R	0/0x27	R[rd] = ~(R[rs] | R[rt])
Shift Left Logical	`sll`	`rd, rt, shamt`	R	0/0x00	R[rd] = R[rt] << shamt
Shift Right Logical	`srl`	`rd, rt, shamt`	R	0/0x02	R[rd] = R[rt] >>> shamt
Set Less Than	`slt`	`rd, rs, rt`	R	0/0x2a	R[rd] = (R[rs] < R[rt]) ? 1 : 0
Set Less Than Imm.	`slti`	`rt, rs, imm`	I	0x0a	R[rt] = (R[rs] < SignExtImm) ? 1 : 0 [b]
Load Upper Imm.	`lui`	`rt, imm`	I	0x0f	R[rt] = {imm, 16'b0}
Load Word	`lw`	`rt, imm(rs)`	I	0x23	R[rt] = M[R[rs] + SignExtImm] [b]
Store Word	`sw`	`rt, imm(rs)`	I	0x2b	M[R[rs] + SignExtImm] = R[rt] [b]
Branch on Equal	`beq`	`rs, rt, label`	I	0x04	if(R[rs] == R[rt]) PC = PC + 4 + BranchAddr [d]
Branch on Not Equal	`bne`	`rs, rt, label`	I	0x05	if(R[rs] != R[rt]) PC = PC + 4 + BranchAddr [d]
Jump	`j`	`target`	J	0x02	PC = JumpAddr [e]
Jump And Link	`jal`	`target`	J	0x03	R[31] = PC + 8; PC = JumpAddr [e]
Jump Register	`jr`	`rs`	R	0/0x08	PC = R[rs]

Register Name, Number, Use

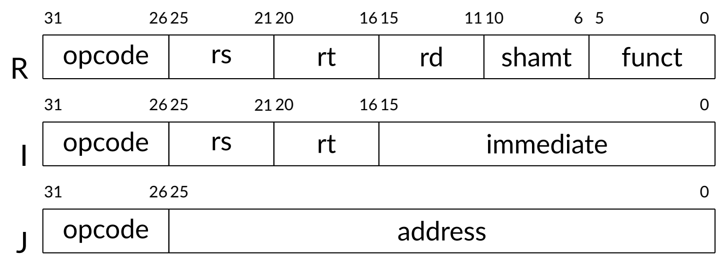
Name	Num	Use
`$zero`	0	The constant value 0
`$at`	1	Assembler temporary
`$v0-$v1`	2 – 3	Values for function results and expression evaluation
`$a0-$a3`	4 – 7	Arguments
`$t0-$t7`	8 – 15	Temporaries
`$s0-$s7`	16 – 23	Saved temporaries
`$t8-$t9`	24 – 25	Temporaries
`$k0-$k1`	26 – 27	Reserved for os kernel
`$gp`	28	Global pointer
`$sp`	29	Stack pointer
`$fp`	30	Frame pointer
`$ra`	31	Return address

a. May cause overflow exception
b. SignExtImm = {16{imm[15]}, imm}
c. ZeroExtImm = {16{1b'0}, imm}
d. BranchAddr = {14{imm[15]}, imm, 2'b0}
e. JumpAddr = {PC+4[31:28], addr, 2'b0}
f. Operands considered unsigned numbers (vs. 2's comp.)

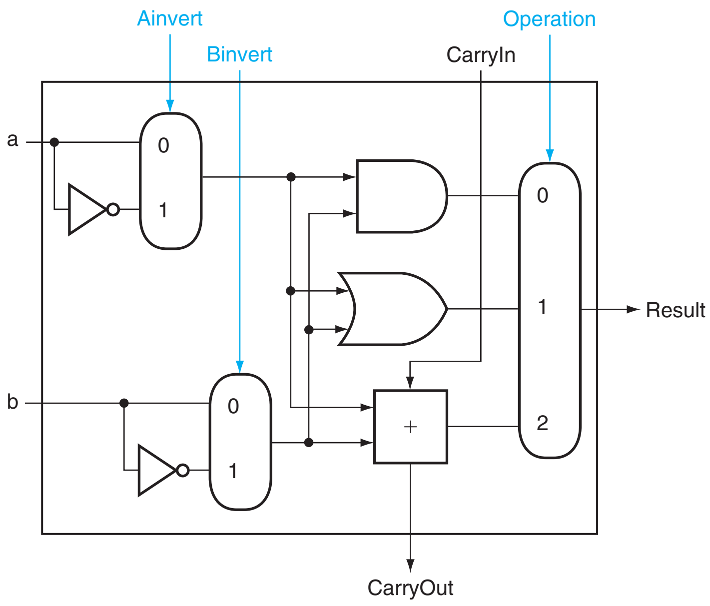
MIPS Data Path with Control Unit



Basic Instruction Formats



1-bit ALU



ALUControl	Ainvert	Binvert	Operation	Action
0000	0	0	00	and
0001	0	0	01	or
0010	0	0	10	add
0110	0	1	10	sub
0111	0	1	11	slt
1100	1	1	00	nor

MIPS Control Signals

	RegDst	ALUSrc	MemToReg	RegWrite	MemRead	MemWrite	Branch	ALUop
R-format	1	0	0	1	0	0	0	10
lw	0	1	1	1	1	0	0	00
sw	x	1	x	0	0	1	0	00
beq	x	0	x	0	0	0	1	01

MIPS ALUControl

Instruction	Opcode	ALUOp	Funct	ALU Action	ALUControl
`lw`	LW	00	xxxxxx	add	0010
`sw`	SW	00	xxxxxx	add	0010
`beq`	Branch equal	01	xxxxxx	subtract	0110
`add`	R-type	10	100000	add	0010
`sub`	R-type	10	100010	subtract	0110
`and`	R-type	10	100100	and	0000
`or`	R-type	10	100101	or	0001
`slt`	R-type	10	101010	set on less than	0111

4-bit Number Systems

Value	Sign & Mag	1s Comp.	2s Comp.
+7	0111	0111	0111
+6	0110	0110	0110
+5	0101	0101	0101
+4	0100	0100	0100
+3	0011	0011	0011
+2	0010	0010	0010
+1	0001	0001	0001
+0	0000	0000	0000
-0	1000	1111	-
-1	1001	1110	1111
-2	1010	1101	1110
-3	1011	1100	1101
-4	1100	1011	1100
-5	1101	1010	1011
-6	1110	1001	1010
-7	1111	1000	1001
-8	-	-	1000

Nibble

HEX	DEC	BIN	1s Comp.	2s Comp.
0	0	0000	+0	+0
1	1	0001	+1	+1
2	2	0010	+2	+2
3	3	0011	+3	+3
4	4	0100	+4	+4
5	5	0101	+5	+5
6	6	0110	+6	+6
7	7	0111	+7	+7
8	8	1000	-7	-8
9	9	1001	-6	-7
A	10	1010	-5	-6
B	11	1011	-4	-5
C	12	1100	-3	-4
D	13	1101	-2	-3
E	14	1110	-1	-2
F	15	1111	-0	-1

ASCII

BIN	DEC	HEX	ASCII	DEC	HEX	ASCII
000000	0	0	NUL	64	40	@
000001	1	1	SOH	65	41	A
000010	2	2	STX	66	42	B
000011	3	3	ETX	67	43	C
000100	4	4	EOT	68	44	D
000101	5	5	ENQ	69	45	E
000110	6	6	ACK	70	46	F
000111	7	7	BEL	71	47	G
001000	8	8	BS	72	48	H
001001	9	9	HT	73	49	I
001010	10	a	LF	74	4a	J
001011	11	b	VT	75	4b	K
001100	12	c	FF	76	4c	L
001101	13	d	CR	77	4d	M
001110	14	e	SO	78	4e	N
001111	15	f	SI	79	4f	O
010000	16	10	DLE	80	50	P
010001	17	11	DC1	81	51	Q
010010	18	12	DC2	82	52	R
010011	19	13	DC3	83	53	S
010100	20	14	DC4	84	54	T
010101	21	15	NAK	85	55	U
010110	22	16	SYN	86	56	V
010111	23	17	ETB	87	57	W
011000	24	18	CAN	88	58	X
011001	25	19	EM	89	59	Y
011010	26	1a	SUB	90	5a	Z
011011	27	1b	ESC	91	5b	[
011100	28	1c	FS	92	5c	\
011101	29	1d	GS	93	5d	]
011110	30	1e	RS	94	5e	^
011111	31	1f	US	95	5f	_
100000	32	20	Space	96	60	‘
100001	33	21	!	97	61	a
100010	34	22	"	98	62	b
100011	35	23	#	99	63	c
100100	36	24	$	100	64	d
100101	37	25	%	101	65	e
100110	38	26	&	102	66	f
100111	39	27	’	103	67	g
101000	40	28	(	104	68	h
101001	41	29	)	105	69	i
101010	42	2a	*	106	6a	j
101011	43	2b	+	107	6b	k
101100	44	2c	,	108	6c	l
101101	45	2d	-	109	6d	m
101110	46	2e	.	110	6e	n
101111	47	2f	/	111	6f	o
110000	48	30	0	112	70	p
110001	49	31	1	113	71	q
110010	50	32	2	114	72	r
110011	51	33	3	115	73	s
110100	52	34	4	116	74	t
110101	53	35	5	117	75	u
110110	54	36	6	118	76	v
110111	55	37	7	119	77	w
111000	56	38	8	120	78	x
111001	57	39	9	121	79	y
111010	58	3a	:	122	7a	z
111011	59	3b	;	123	7b	{
111100	60	3c	<	124	7c	|
111101	61	3d	=	125	7d	}
111110	62	3e	>	126	7e	~
111111	63	3f	?	127	7f	DEL

IEEE 754 Floating Point Standard

$$(-1)^S \times M \times 2^{E-B}$$

Single Precision Format (Bias = 127)

31	30 – 23	22 – 0
S	Exponent	Mantissa

Double Precision Format (Bias = 1023)

63	62 – 52	51 – 32
S	Exponent	Mantissa[51:32]

31 – 0
Mantissa[31:0]

K-Map

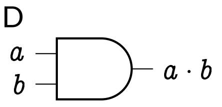
AB \ CD	00	01	11	10
00	m_0	m_1	m_3	m_2
01	m_4	m_5	m_7	m_6
11	m_{12}	m_{13}	m_{15}	m_{14}
10	m_8	m_9	m_{11}	m_{10}

Laws of Boolean Algebra

Name	AND	OR
Identity Law	$x \cdot 1 = x$	$x + 0 = x$
Invert/Complement Law	$x \cdot x' = 0$	$x + x' = 1$
Commutative Law	$x \cdot y = y \cdot x$	$x + y = y + x$
Associative Law	$x \cdot (y \cdot z) = (x \cdot y) \cdot z$	$x + (y + z) = (x + y) + z$
Distributive Law	$x \cdot (y + z) = x \cdot y + x \cdot z$	$x + y \cdot z = (x + y) \cdot (x + z)$
Idempotency	$x \cdot x = x$	$x + x = x$
Zero/One Element	$x \cdot 0 = 0$	$x + 1 = 1$
Involution	$(x')' = x$	
Absorption 1	$x + x \cdot y = x$	$x \cdot (x + y) = x$
Absorption 2	$x + x' \cdot y = x + y$	$x \cdot (x' + y) = x \cdot y$
DeMorgan's Law	$(x \cdot y)' = x' + y'$	$(x + y)' = x' \cdot y'$
Consensus	$x \cdot y + x' \cdot z + y \cdot z = x \cdot y + x' \cdot z$	$(x+y) \cdot (x'+z) \cdot (y+z) = (x+y) \cdot (x'+z)$

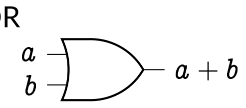
Truth Tables

AND



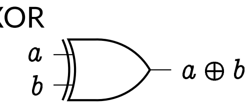
a	b	$a \cdot b$
0	0	0
0	1	0
1	0	0
1	1	1

OR



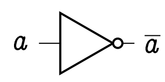
a	b	$a + b$
0	0	0
0	1	1
1	0	1
1	1	1

XOR



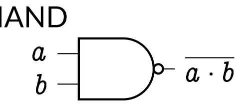
a	b	$a \oplus b$
0	0	0
0	1	1
1	0	1
1	1	0

NOT



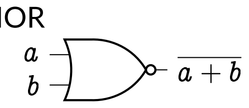
a	$\overline{a}$
0	1
1	0

NAND



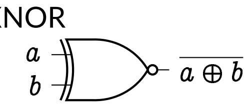
a	b	$\overline{a \cdot b}$
0	0	1
0	1	1
1	0	1
1	1	0

NOR



a	b	$\overline{a + b}$
0	0	1
0	1	0
1	0	0
1	1	0

XNOR



a	b	$\overline{a \oplus b}$
0	0	1
0	1	0
1	0	0
1	1	1