Name	Action	Opcode bitfields						8bit opcode (after SRL)	Hex opcode (after SRL)	Decimal opcode	offset	Opcode abbreviation	Additional Opcode bits	Sub offset	Opcode	Special register case #
Divide	HI=rs%rt; LO=rs/rt	000000	rs	rt		0	11010	00000000	00	00	24	DIV	1A	0	DIV rs,rt	2
Divide Unsigned	HI=rs%rt; LO=rs/rt	000000	rs	rt		0	11011	00000000	00	00	25	DIVU	1B	1	DIVU rs,rt	2
Move To HI	HI=rs	000000	rs			0	10001	00000000	00	00	28	MTHI	11	2	MTHI rs	3
Move To LO	LO=rs	000000	rs			0	10011	00000000	00	00	29	MTLO	13	3	MTLO rs	3
Multiply	HI,LO=rs*rt	000000	rs	rt		0	11000	00000000	00	00	30	MULT	18	4	MULT rs,rt	2
Multiply Unsigned	HI,LO=rs*rt	000000	rs	rt		0	11001	00000000	00	00	31	MULTU	19	5	MULTU rs,rt	2
Jump Register	pc=rs	000000	rs			0	1000	00000000	00	00	44	JR	08	6	JR rs	3
System Call	epc=pc; pc=0x3c	000000	00000000000000000000				1100	00000000	00	00	47	SYSCALL	0C	7	SYSCALL	10
Breakpoint	epc=pc; pc=0x3c	000000	code				1101	00000000	00	00	40	BREAK	0D	8	BREAK	10
Add	rd=rs+rt	000000	rs	rt	rd	0	100000	00000000	00	00	0	ADD	20	9	ADD rd,rs,rt	0
Add Unsigned	rd=rs+rt	000000	rs	rt	rd	0	100001	00000000	00	00	3	ADDU	21	10	ADDU rd,rs,rt	0
And	rd=rs&rt	000000	rs	rt	rd	0	100100	00000000	00	00	4	AND	24	11	AND rd,rs,rt	0
Nor	rd=~(rs|rt)	000000	rs	rt	rd	0	100111	00000000	00	00	7	NOR	27	12	NOR rd,rs,rt	0
Or	rd=rs|rt	000000	rs	rt	rd	0	100101	00000000	00	00	8	OR	25	13	OR rd,rs,rt	0
Set On Less Than	rd=rs<rt	000000	rs	rt	rd	0	101010	00000000	00	00	10	SLT	2A	14	SLT rd,rs,rt	0
Set On Less Than Unsigned	rd=rs<rt	000000	rs	rt	rd	0	101011	00000000	00	00	13	SLTU	2B	15	SLTU rd,rs,rt	0
Subtract	rd=rs-rt	000000	rs	rt	rd	0	100010	00000000	00	00	14	SUB	22	16	SUB rd,rs,rt	0
Subtract Unsigned	rd=rs-rt	000000	rs	rt	rd	0	100011	00000000	00	00	15	SUBU	23	17	SUBU rd,rs,rt	0
Exclusive Or	rd=rs^rt	000000	rs	rt	rd	0	100110	00000000	00	00	16	XOR	26	18	XOR rd,rs,rt	0
Shift Left Logical	rd=rt<<sa	000000	rs	rt	rd	sa	0	00000000	00	00	18	SLL	00	19	SLL rd,rt,sa	1
Shift Left Logical Variable	rd=rt<<rs	000000	rs	rt	rd	0	100	00000000	00	00	19	SLLV	04	20	SLLV rd,rt,rs	0
Shift Right Arithmetic	rd=rt>>sa	000000	0	rt	rd	sa	11	00000000	00	00	20	SRA	03	21	SRA rd,rt,sa	1
Shift Right Arithmetic Variable	rd=rt>>rs	000000	rs	rt	rd	0	111	00000000	00	00	21	SRAV	07	22	SRAV rd,rt,rs	0
Shift Right Logical	rd=rt>>sa	000000	rs	rt	rd	sa	10	00000000	00	00	22	SRL	02	23	SRL rd,rt,sa	1
Shift Right Logical Variable	rd=rt>>rs	000000	rs	rt	rd	0	110	00000000	00	00	23	SRLV	06	24	SRLV rd,rt,rs	0
Move From HI	rd=HI	000000	0		rd	0	10000	00000000	00	00	26	MFHI	10	25	MFHI rd	4
Move From LO	rd=LO	000000	0		rd	0	10010	00000000	00	00	27	MFLO	12	26	MFLO rd	4
Jump And Link Register	rd=pc; pc=rs	000000	rs	0	rd	0	1001	00000000	00	00	43	JALR	09	27	JALR rs	3
Branch On >= 0	if(rs>=0) pc+=offset*4	000001	rs	1	offset			00000001	01	01	33	BGEZ	01	0	BGEZ rs,offset	6
Branch On >= 0 And Link	r31=pc; if(rs>=0) pc+=offset*4	000001	rs	10001	offset			00000001	01	01	34	BGEZAL	11	1	BGEZAL rs,offset	6
Branch On	if(rs<0) pc+=offset*4	000001	rs	0	offset			00000001	01	01	37	BLTZ	00	2	BLTZ rs,offset	6
Branch On	r31=pc; if(rs<0) pc+=offset*4	000001	rs	10000	offset			00000001	01	01	38	BLTZAL	10	3	BLTZAL rs,offset	6
Move From Coprocessor	rt=CPR[0,rd]	010000	0	rt	rd	0		00010000	10	16	45	MFC0	00	0	MFC0 rt,rd	7
Move To Coprocessor	CPR[0,rd]=rt	010000	100	rt	rd	0		00010000	10	16	46	MTC0	04	1	MTC0 rt,rd	7

Jump	pc=pc_upper|(target<<2)	000010	target			00000010	02	02	41	J		0	J target	8
Jump And Link	r31=pc; pc=target<<2	000011	target			00000011	03	03	42	JAL		1	JAL target	8
Branch On Equal	if(rs==rt) pc+=offset*4	000100	rs	rt	offset	00000100	04	04	32	BEQ		2	BEQ rs,rt,offset	9
Branch On Not Equal	if(rs!=rt) pc+=offset*4	000101	rs	rt	offset	00000101	05	05	39	BNE		3	BNE rs,rt,offset	9
Branch On	if(rs<=0) pc+=offset*4	000110	rs	0	offset	00000110	06	06	36	BLEZ		4	BLEZ rs,offset	6
Branch On > 0	if(rs>0) pc+=offset*4	000111	rs	0	offset	00000111	07	07	35	BGTZ		5	BGTZ rs,offset	6
Add Immediate	rt=rs+imm	001000	rs	rt	imm	00001000	08	08	1	ADDI		6	ADDI rt,rs,imm	9
Add Immediate Unsigned	rt=rs+imm	001001	rs	rt	imm	00001001	09	09	2	ADDIU		7	ADDIU rt,rs,imm	9
Set On Less Than Immediate	rt=rs<imm	001010	rs	rt	imm	00001010	0A	10	11	SLTI		8	SLTI rt,rs,imm	9
Set On	rt=rs<imm	001011	rs	rt	imm	00001011	0B	11	12	SLTIU		9	SLTIU rt,rs,imm	9
And Immediate	rt=rs&imm	001100	rs	rt	imm	00001100	0C	12	5	ANDI		10	ANDI rt,rs,imm	9
Or Immediate	rt=rs|imm	001101	rs	rt	imm	00001101	0D	13	9	ORI		11	ORI rt,rs,imm	9
Exclusive Or Immediate	rt=rs^imm	001110	rs	rt	imm	00001110	0E	14	17	XORI		12	XORI rt,rs,imm	9
Load Upper Immediate	rt=imm<<16	001111	rs	rt	imm	00001111	0F	15	6	LUI		13	LUI rt,imm	11
Load Byte	rt=*(char*)(offset+rs)	100000	rs	rt	offset	00100000	20	32	48	LB		14	LB rt,offset(rs)	12
Load Halfword	rt=*(short*)(offset+rs)	100001	rs	rt	offset	00100001	21	33	50	LH		15	LH rt,offset(rs)	12
Load Word	rt=*(int*)(offset+rs)	100011	rs	rt	offset	00100011	23	35	52	LW		16	LW rt,offset(rs)	12
Load Byte Unsigned	rt=*(Uchar*)(offset+rs)	100100	rs	rt	offset	00100100	24	36	49	LBU		17	LBU rt,offset(rs)	12
Load Halfword Unsigned	rt=*(Ushort*)(offset+rs)	100101	rs	rt	offset	00100101	25	37	51	LBU		18	LBU rt,offset(rs)	12
Store Byte	*(char*)(offset+rs)=rt	101000	rs	rt	offset	00101000	28	40	53	SB		19	SB rt,offset(rs)	12
Store Halfword	*(short*)(offset+rs)=rt	101001	rs	rt	offset	00101001	29	41	54	SH		20	SH rt,offset(rs)	12
Store Word	*(int*)(offset+rs)=rt	101011	rs	rt	offset	00101011	2B	43	55	SW		21	SW rt,offset(rs)	12

This spreadsheet was based on the table found here : https://opencores.org/project,plasma,opcodes but contains many additions to make it more useful for keeping track of instruction formats