ULNAv2-B Instruction Set, by Alphabetical Order

Instruction	Instruction Class	Instruction Number	RTN (PC ← PC + 1, unless otherwise specified)	Updates Cond Codes
add.	A	1	R[W] ← R[A] + R[B]	Y
addi.	B	9	R[W] ← R[A] + SignExtend(Imm5)	Y
and.	A	2	R[W] ← R[A] AND R[B]	Y
andi.	C	18	R[W] ← R[W] AND OneExtend(Imm8)	Y
ash	B	8	R[W] ← ArithmeticShiftRight(R[A], SignExtend(Imm5))	
b	D	24	PC ← PC + SignExtend(Imm11)	
b.eq	D	26	PC ← PC + SignExtend(Imm11) if Z == 1	
b.ge	D	27	PC ← PC + SignExtend(Imm11) if N == V	
b.gt	D	25	PC ← PC + SignExtend(Imm11) if Z == 0 AND N == V	
b.le	D	30	PC ← PC + SignExtend(Imm11) if NOT (Z == 0 AND N == V)	
b.lt	D	28	PC ← PC + SignExtend(Imm11) if NOT (N == V)	
b.ne	D	29	PC ← PC + SignExtend(Imm11) if Z == 0	
bl	C	23	R[W] ← PC + 1; PC ← PC + SignExtend(Imm8)	
br	A	4	PC ← R[A]	
brl	B	13	R[W] ← PC + 1; PC ← R[A]	
cmp.	A	5	Update Cond Codes by performing R[A] - R[B]	Y
cmpi.	C	21	Update Cond Codes by performing R[W] - SignExtend(Imm8)	Y
halt	D	31	MemWord[0xFFFF] ← SignExtend(imm11); PC ← PC	
ldw	A	7	R[W] ← MemWord[R[A] + R[B]]	
ldwi	B	15	R[W] ← MemWord[R[A] + SignExtend(Imm5)]	
lsh	B	10	R[W] ← LogicalShiftRight(R[A], SignExtend(Imm5))	
movi	C	20	R[W] ← ZeroExtend(Imm8)	
movis	C	22	R[W] ← R[W][7:0] OR LogicalShiftLeft(Imm8, 8)	
negi.	B	12	R[W] ← NOT R[A] + SignExtend(Imm5)	Y
or.	A	0	R[W] ← R[A] OR R[B]	Y
ori.	C	16	R[W] ← R[W] OR ZeroExtend(Imm8)	Y
rot	B	14	R[W] ← RollRight(R[A], SignExtend(Imm5))	
stw	A	3	MemWord[R[A] + R[B]] ← R[W]	
stwi	B	11	MemWord[R[A] + SignExtend(Imm5)] ← R[W]	
undef - 06	A	6	don't care	
undef - 17	C	17	don't care	
undef - 19	C	19	don't care	

ULNAv2-B Instruction Set, by Instruction Number Order

Instruction Number	Instruction
0	or.
1	add.
2	and.
3	stw
4	br
5	cmp.
6	undef - 06
7	ldw
8	ash
9	addi.
10	lsh
11	stwi
12	negi.
13	brl
14	rot
15	ldwi
16	ori.
17	undef - 17
18	andi.
19	undef - 19
20	movi
21	cmpi.
22	movis
23	bl
24	b
25	b.gt
26	b.eq
27	b.ge
28	b.lt
29	b.ne
30	b.le
31	halt

1