RISC-V ALUControl Cheat Sheet

Standard ALUControl Codes

ALUControl	Operation	Description
0000	AND	Bitwise AND
0001	OR	Bitwise OR
0010	ADD	Addition
0110	SUB	Subtraction
0111	SLT	Set if A < B (signed)
1000	XOR	Bitwise XOR
1001	SLL	Shift Left Logical
1010	SRL	Shift Right Logical
1011	SRA	Shift Right Arithmetic
1100	SLTU	Set if A < B (unsigned)

Instruction Mapping to ALUControl

Instruction	Type	ALUOp	funct3	funct7[5]	ALUControl
add	R	10	000	0	0010
sub	R	10	000	1	0110
addi	I	10	000	-	0010
and	R	10	111	0	0000
andi	I	10	111	-	0000
or	R	10	110	0	0001
ori	I	10	110	-	0001
xor	R	10	100	0	1000
xori	I	10	100	-	1000
sll	R	10	001	0	1001
slli	I	10	001	0	1001
srl	R	10	101	0	1010
srli	I	10	101	0	1010
sra	R	10	101	1	1011
srai	I	10	101	1	1011
slt	R	10	010	0	0111
slti	I	10	010	-	0111
sltu	R	10	011	0	1100
sltiu	I	10	011	-	1100

Branch Instructions

Instruction	ALUOp	funct3	ALUControl	Comparison Used
beq	01	000	0110	A == B (A - B == 0)
bne	01	001	0110	A B
blt	01	100	0111	A < B (signed)
bge	01	101	0111	A B (signed)
bltu	01	110	1100	A < B (unsigned)
bgeu	01	111	1100	A B (unsigned)

Tips for Memorization

- ALUControl is **internal to your CPU design**, not part of the RISC-V ISA.
- ADD (‘0010‘) is used by default for load/store and ‘addi‘.
- SUB (‘0110‘) is reused for branches and ‘sub‘.
- Use SLT (‘0111‘) for signed comparisons, SLTU (‘1100‘) for unsigned.
- Right shifts: SRL = ‘1010‘, SRA = ‘1011‘; Left shift = ‘1001‘.