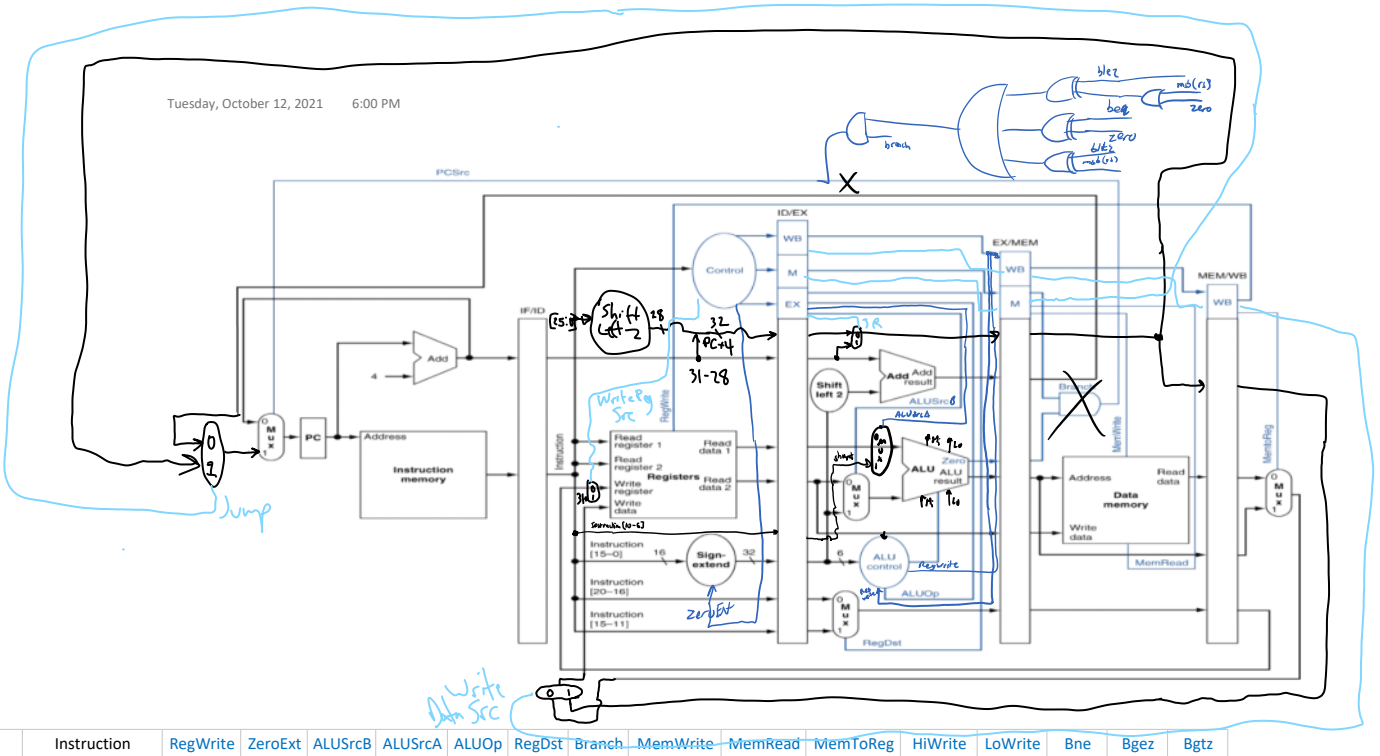
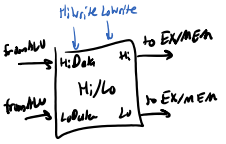
Tuesday, October 12, 2021 6:00 PM



Instruction Type	Instruction	RegWrite	ZeroExt	ALUSrcB	ALUSrcA	ALUOp	RegDst	Branch	MemWrite	MemRead	MemToReg	HiWrite	LoWrite	Bne	Bgez	Bgtz
Arithmetic, R-type	add rd, rs, rt	1	X	0	0	add	1	0	0	X	1	0	0	X	X	X
	addu rd, rs, rt	1	X	0	0	add	1	0	0	X	1	0	0	X	X	X
	sub rd, rs, rt	1	X	0	0	sub	1	0	0	X	1	0	0	X	X	X
	mul rd, rs, rt	1	X	0	0	mult	1	0	0	X	1	0	0	X	X	X
Arithmetic, I-type	addi rt, rs, imm	1	0	1	0	add	0	0	0	X	1	0	0	X	X	X
	addiu rt, rs, imm	1	1	1	0	add	0	0	0	X	1	0	0	X	X	X
Arithmetic, hi-lo	mult rs, rt	0	X	0	0	mult	X	0	0	X	X	1	1	X	X	X
	multu rs, rt	0	X	0	0	multu	X	0	0	X	X	1	1	X	X	X
	madd rs, rt	0	X	0	0	madd	X	0	0	X	X	1	1	X	X	X
	msub rs, rt	0	X	0	0	msub	X	0	0	X	X	1	1	X	X	X
Data, I-type	lw rt, imm(rs)	1	0	1	0	add	0	0	0	1	0	0	0	X	X	X
	sw rt, imm(rs)	0	0	1	0	add	X	0	1	X	X	0	0	X	X	X
	lh rt, imm(rs)	1	0	1	0	add	0	0	0	2	0	0	0	X	X	X
	sh rt, imm(rs)	0	0	1	0	add	X	0	2	X	X	0	0	X	X	X
	lb rt, imm(rs)	1	0	1	0	add	0	0	0	3	0	0	0	X	X	X
	sb rt, imm(rs)	0	0	1	0	add	X	0	3	X	X	0	0	X	X	X
	lui rt, imm	1	X	1	0	sll16	0	0	0	X	1	0	0	X	X	X
Branch, I-type	beq rs, rt, imm	0	0	0	0	sub	X	1	0	X	X	0	0	0	X	X
	bgez rs, imm	0	0	0	0	X	X	1	0	X	X	0	0	X	1	X
	bne rs, rt, imm	0	0	0	0	sub	X	1	0	X	X	0	0	1	X	X
	bgtz rs, imm	0	0	0	0	add	X	1	0	X	X	0	0	X	X	1
	blez rs, imm	0	0	0	0	add	X	1	0	X	X	0	0	X	X	0
	bltz rs, imm	0	0	0	0	X	X	1	0	X	X	0	0	X	0	X
Jump, J-type	j addr													X	X	X
	jal addr													X	X	X
Jump, R-type	jr rs													X	X	X
Logical, R-type	and rd, rs, rt	1	X	0	0	and	1	0	0	X	1	0	0	X	X	X
	or rd, rs, rt	1	X	0	0	or	1	0	0	X	1	0	0	X	X	X
	nor rd, rs, rt	1	X	0	0	nor	1	0	0	X	1	0	0	X	X	X
	xor rd, rs, rt	1	X	0	0	xor	1	0	0	X	1	0	0	X	X	X
	seh rd, rt	1	X	0	0	seh	1	0	0	X	1	0	0	X	X	X
	seb rd, rt	1	X	0	0	seb	1	0	0	X	1	0	0	X	X	X
	slt, rd, rs, rt	1	X	0	0	slt	1	0	0	X	1	0	0	X	X	X
	sltu rd, rs, rt	1	X	0	0	sltu	1	0	0	X	1	0	0	X	X	X
	movn rd, rs, rt	X	X	X	0	passA	1	0	0	X	1	0	0	X	X	X
	movz rd, rs, rt	X	X	X	0	passA	1	0	0	X	1	0	0	X	X	X
Shifts/Rotates, R-type	sll rd, rt, shamt	1	X	0	1	sll	1	0	0	X	1	0	0	X	X	X
	srl rd, rt, shamt	1	X	0	1	srl	1	0	0	X	1	0	0	X	X	X
	sllv rd, rt, rs	1	X	0	0	sll	1	0	0	X	1	0	0	X	X	X
	srlv rd, rt, rs	1	X	0	0	srl	1	0	0	X	1	0	0	X	X	X
	sra rd, rt, shamt	1	X	0	1	sra	1	0	0	X	1	0	0	X	X	X
	srav rd, rt, rs	1	X	0	0	sra	1	0	0	X	1	0	0	X	X	X
	rotr rd, rt, shamt	1	X	0	1	rotr	1	0	0	X	1	0	0	X	X	X



	rotrv rd, rt, rs	1	X	0	0	rotr	1	0	0	X	1	0	0	X	X	X
Logical, I-type	andi rt, rs, imm	1	0	1	0	and	0	0	0	X	1	0	0	X	X	X
	ori rt, rs, imm	1	0	1	0	or	0	0	0	X	1	0	0	X	X	X
	xori rt, rs, imm	1	0	1	0	xor	0	0	0	X	1	0	0	X	X	X
	slti rt, rs, imm	1	0	1	0	slt	0	0	0	X	1	0	0	X	X	X
	sltiu rt, rs, imm	1	1	1	0	sltu	0	0	0	X	1	0	0	X	X	X
Hi/Lo	mthi rs	0	X	X	0	mthi	X	0	0	X	X	1	0	X	X	X
	mtlo rs	0	X	X	0	mtlo	X	0	0	X	X	0	1	X	X	X
	mfhi rd	1	X	X	X	mfhi	1	0	0	X	1	0	0	X	X	X
	mflo rd	1	X	X	X	mflo	1	0	0	X	1	0	0	X	X	X
Instruction Type	Instruction	RegWrite	ZeroExt	ALUSrcB	ALUSrcA	ALUOp	RegDst	Branch	MemWrite	MemRead	MemToReg	HiWrite	LoWrite	Bne	Bgez	Bgtz

ALUControl	Operation	Info
5'b00000	AND	A & B
5'b00001	OR	A | B
5'b00010	ADD	A + B (no overflow detection)
5'b00011	SLL	B << A
5'b00100	SRL	B >> A
5'b00101	MULT	A * B (32 msb -> hi, 32 lsb -> lo)
5'b00110	SUB	A - B (no overflow detection)
5'b00111	SLT	If A < B then 1 else 0 (signed)
5'b01000	MADD	{hi, lo} + A * B (32 msb -> hi, 32 lsb -> lo)
5'b01001	MSUB	{hi, lo} - A * B (32 msb -> hi, 32 lsb -> lo)
5'b01010	SLL16	B << 16
5'b01011	MTHI	A -> hi
5'b01100	NOR	~(A | B)
5'b01101	XOR	A ^ B
5'b01110	MTLO	A -> lo
5'b01111	MFHI	hi
5'b10000	MFLO	lo
5'b10001	ADD16B	A[15:0] + B[15:0]
5'b10010	ADD8B	A[7:0] + B[7:0]
5'b10011	PASSA	A
5'b10100	SLTU	If A < B then 1 else 0 (unsigned)
5'b10101	ROTR	{B[A-1:0], B[32:A]}
5'b10110	SRA	B >>> A
5'b10111	SEH	{{16{B[15]}}, B[15:0]}
5'b11000	SEB	{{24{B[7]}}, B[7:0]}