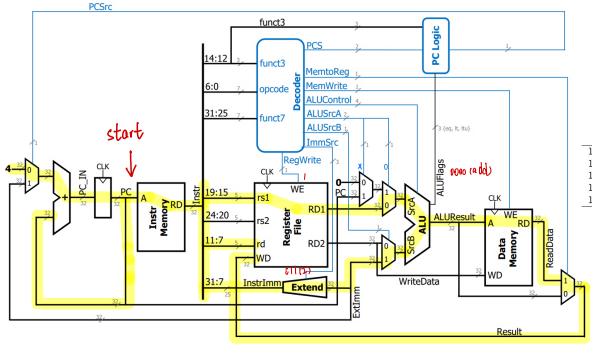
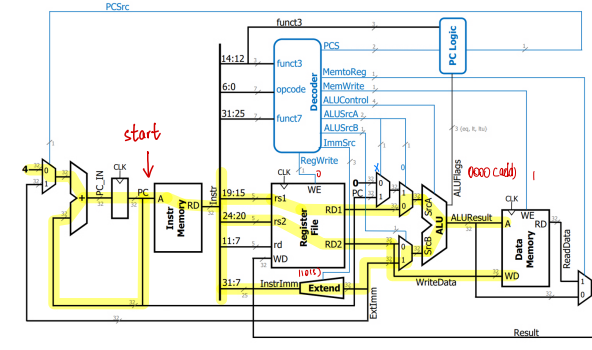
1. Datapath visualization.

lw (I type)



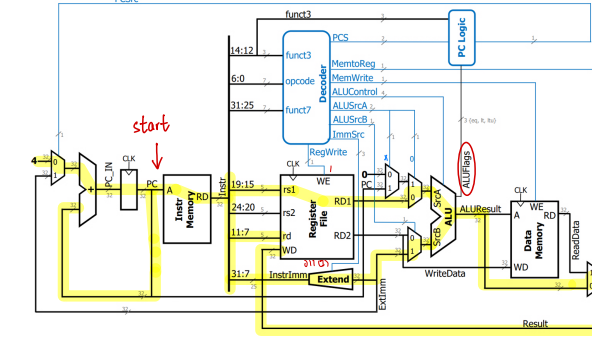
lb	Load Byte	I	0000011	0x0		rd = M[rs1+imm][0:7]
lh	Load Half	I	0000011	0x1		rd = M[rs1+imm][0:15]
lw	Load Word	I	0000011	0x2		rd = M[rs1+imm][0:31]
lbu	Load Byte (U)	I	0000011	0x4		rd = M[rs1+imm][0:7]
lhu	Load Half (U)	I	0000011	0x5		rd = M[rs1+imm][0:15]

sw (S-type)



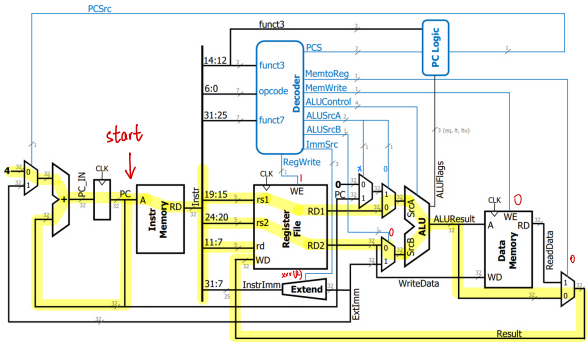
sb	Store Byte	S	0100011	0x0		M[rs1+imm][0:7] = rs2[0:7]
sh	Store Half	S	0100011	0x1		M[rs1+imm][0:15] = rs2[0:15]
sw	Store Word	S	0100011	0x2		M[rs1+imm][0:31] = rs2[0:31]

DP Imm (I)



addi	ADD (Immediate)	I	0010011	0x0		rd = rs1 + imm
xori	XOR (Immediate)	I	0010011	0x4		rd = rs1 ^ imm
ori	OR (Immediate)	I	0010011	0x6		rd = rs1 | imm
andi	AND (Immediate)	I	0010011	0x7		rd = rs1 & imm
slli	Shift Left Logical Imm	I	0010011	0x1	imm[11:5]=0x00	rd = rs1 << imm[0:4]
srli	Shift Right Logical Imm	I	0010011	0x5	imm[11:5]=0x00	rd = rs1 >> imm[0:4]
srai	Shift Right Arith Imm	I	0010011	0x5	imm[11:5]=0x20	rd = rs1 >> imm[0:4]
slti	Set Less Than Imm	I	0010011	0x2		rd = (rs1 < imm)?1:0
sltiu	Set Less Than Imm (U)	I	0010011	0x3		rd = (rs1 < imm)?1:0

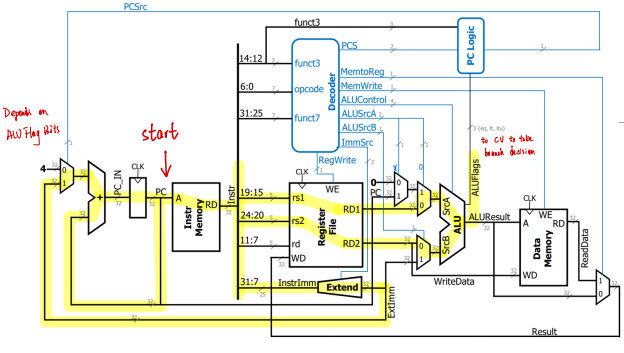
DP Reg.



RV - Instruction

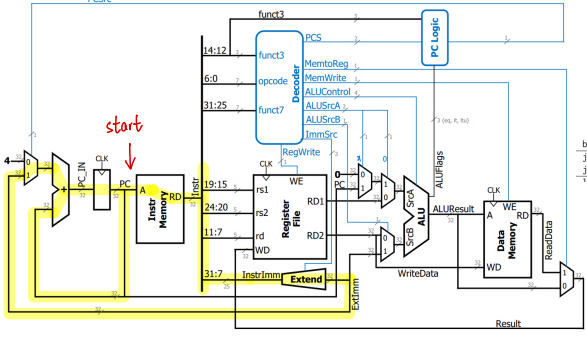
Inst	Name	FMT	Opcode	funct3	funct7	Description (C)
add	ADD	R	0110011	0x0	0x00	rd = rs1 + rs2
sub	SUB	R	0110011	0x0	0x20	rd = rs1 - rs2
xor	XOR	R	0110011	0x4	0x00	rd = rs1 ^ rs2
or	OR	R	0110011	0x6	0x00	rd = rs1 | rs2
and	AND	R	0110011	0x7	0x00	rd = rs1 & rs2
sll	Shift Left Logical	R	0110011	0x1	0x00	rd = rs1 << rs2
srl	Shift Right Logical	R	0110011	0x5	0x00	rd = rs1 >> rs2
sra	Shift Right Arith	R	0110011	0x5	0x20	rd = rs1 >> rs2
slt	Set Less Than	R	0110011	0x2	0x00	rd = (rs1 < rs2)?1:0
sltu	Set Less Than (U)	R	0110011	0x3	0x00	rd = (rs1 < rs2)?1:0

Branch



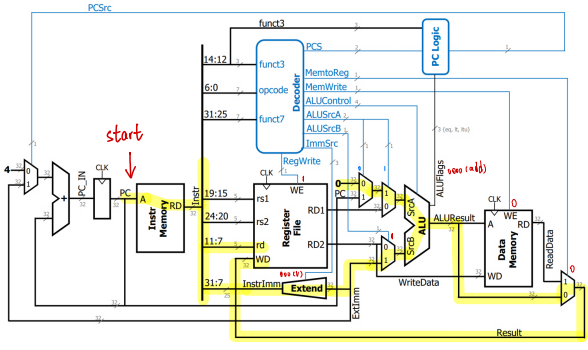
beq	Branch ==	B	1100011	0x0		if(rs1 == rs2) PC += imm
bne	Branch !=	B	1100011	0x1		if(rs1 != rs2) PC += imm
blt	Branch <	B	1100011	0x4		if(rs1 < rs2) PC += imm
bge	Branch ≥	B	1100011	0x5		if(rs1 >= rs2) PC += imm
bltu	Branch < (U)	B	1100011	0x6		if(rs1 < rs2) PC += imm
bgeu	Branch ≥ (U)	B	1100011	0x7		if(rs1 >= rs2) PC += imm

jal (without linking) -> PC += Imm.
↳ J type



bgeu	Branch ≥ (U)	B	1100011	0x7		if(rs1 >= rs2) PC += imm
jal	Jump And Link	J	1101111			rd = PC+4; PC += imm
jalr	Jump And Link Reg	I	1100111	0x0		rd = PC+4; PC = rs1 + imm

lui → U type



auipc → U type

jalr	Jump And Link Reg	I	1100111	0x0		rd = PC+4; PC = rs1 + imm
lui	Load Upper Imm	U	0110111			rd = imm << 12
auipc	Add Upper Imm to PC	U	0010111			rd = PC + (imm << 12)