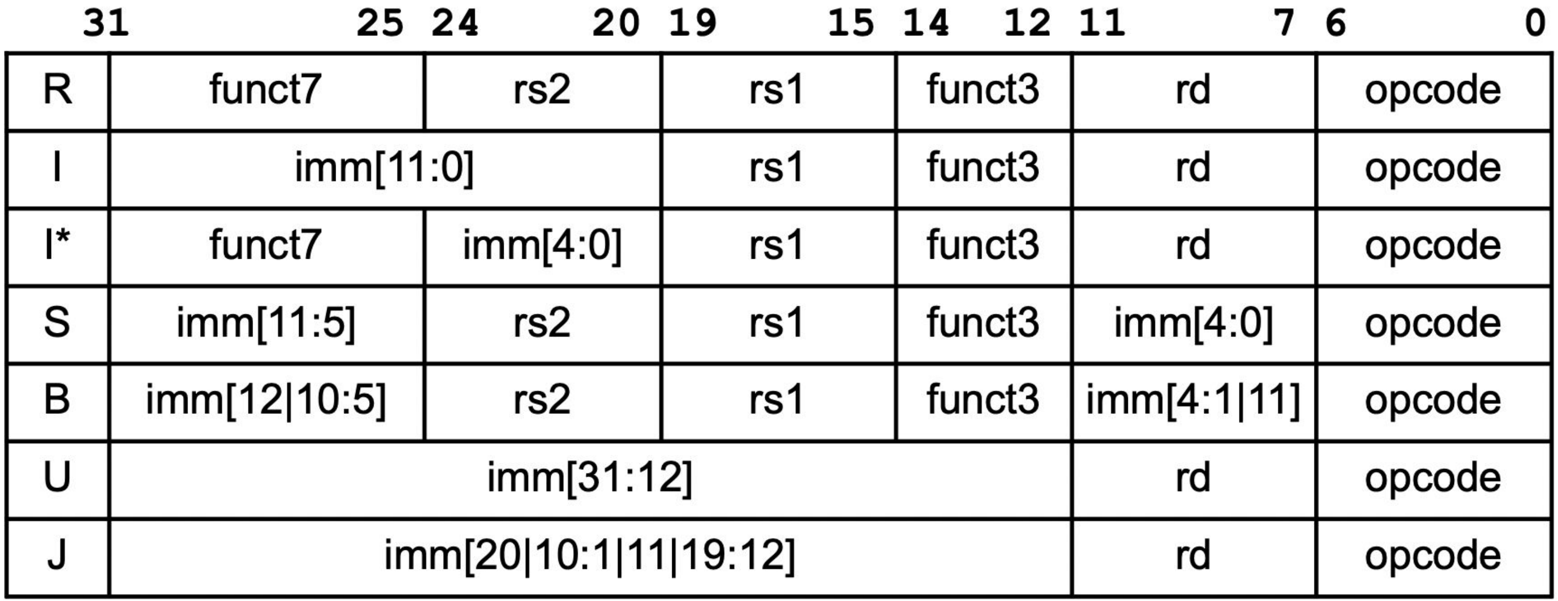
Lab 6

CS61C

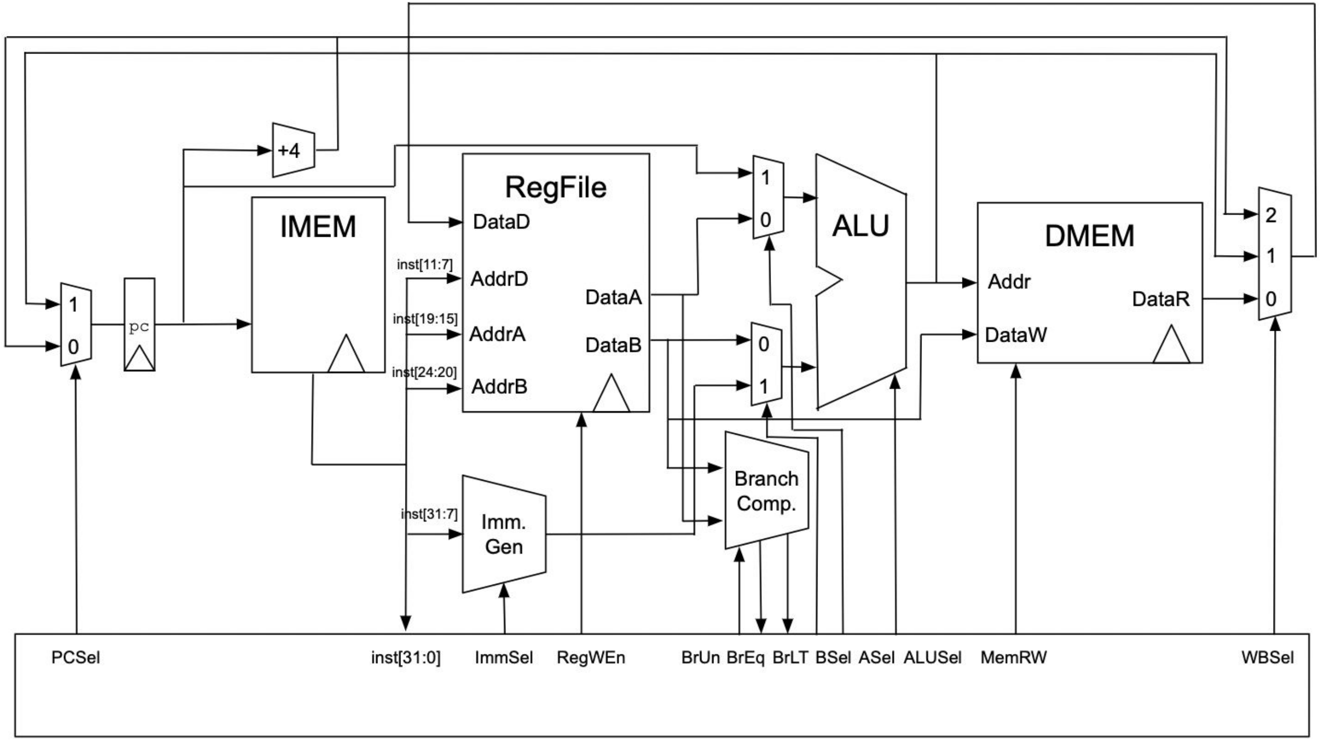
Immediates



	31 25	24 20	19 15	14 12	11 7	6 0
R	funct7	rs2	rs1	funct3	rd	opcode
I	imm[11:0]		rs1	funct3	rd	opcode
I*	funct7	imm[4:0]	rs1	funct3	rd	opcode
S	imm[11:5]	rs2	rs1	funct3	imm[4:0]	opcode
B	imm[12|10:5]	rs2	rs1	funct3	imm[4:1|11]	opcode
U	imm[31:12]				rd	opcode
J	imm[20|10:1|11|19:12]				rd	opcode

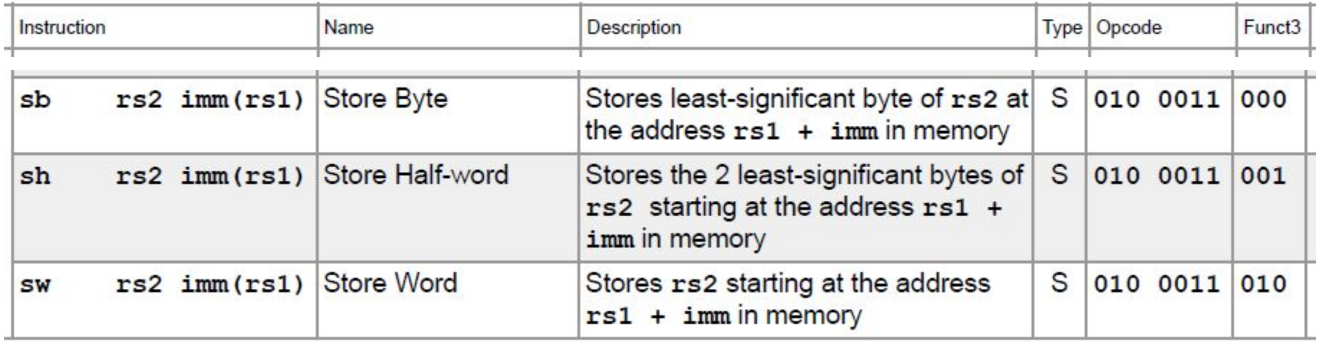
Immediates are sign-extended to 32 bits, except in I* type instructions and **`sltiu`**.

Control Signals



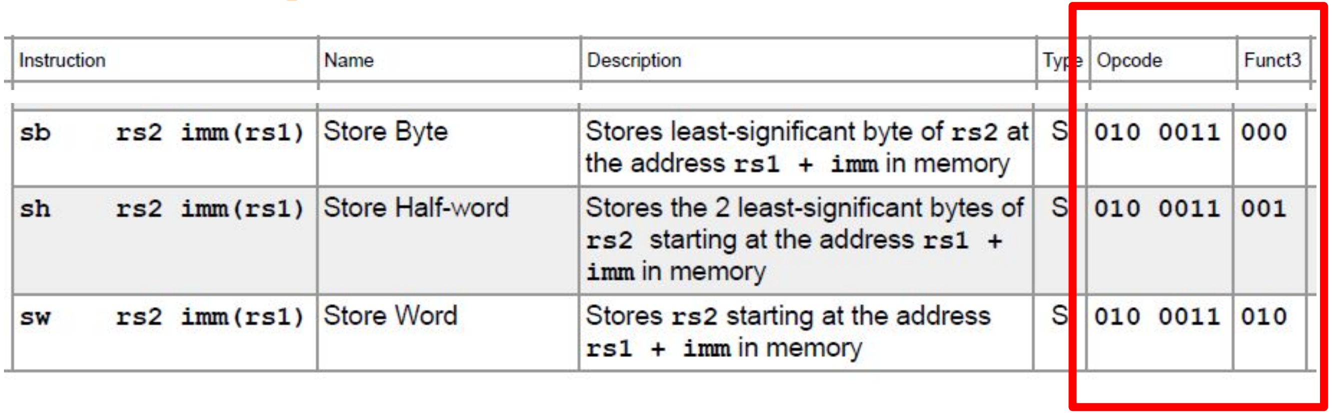
MemRW Example

MemRW Value	Occurs when
0	Reading from memory
1	Writing to memory
Don’t Care	Can never be don’t care!

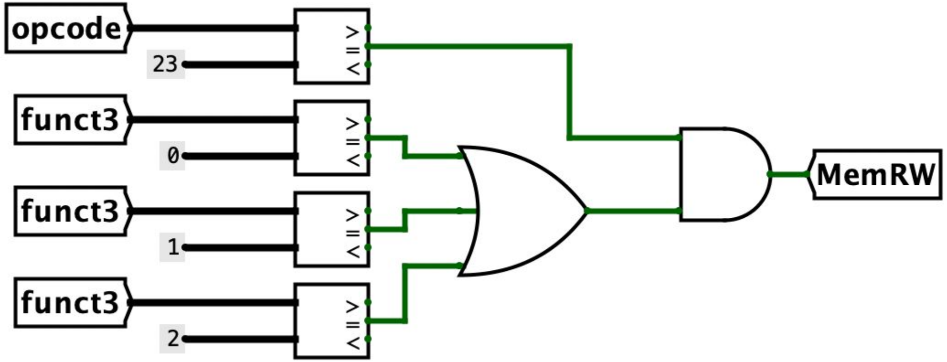


Instruction	Name	Description	Type	Opcode	Funct3
`sb rs2 imm(rs1)`	Store Byte	Stores least-significant byte of `rs2` at the address `rs1 + imm` in memory	S	`010 0011`	`000`
`sh rs2 imm(rs1)`	Store Half-word	Stores the 2 least-significant bytes of `rs2` starting at the address `rs1 + imm` in memory	S	`010 0011`	`001`
`sw rs2 imm(rs1)`	Store Word	Stores `rs2` starting at the address `rs1 + imm` in memory	S	`010 0011`	`010`

MemRW Example

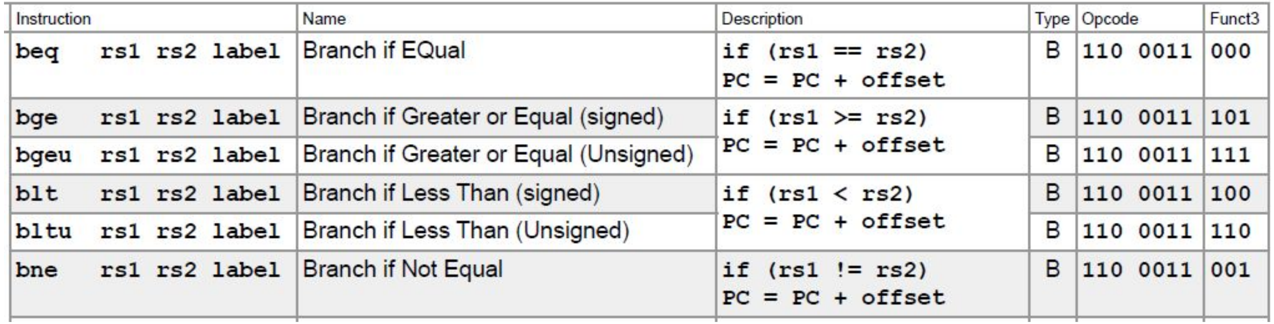


Instruction	Name	Description	Type	Opcode	Funct3
sb rs2 imm(rs1)	Store Byte	Stores least-significant byte of rs2 at the address rs1 + imm in memory	S	010 0011	000
sh rs2 imm(rs1)	Store Half-word	Stores the 2 least-significant bytes of rs2 starting at the address rs1 + imm in memory	S	010 0011	001
sw rs2 imm(rs1)	Store Word	Stores rs2 starting at the address rs1 + imm in memory	S	010 0011	010



BrUn

BrUn Value	Occurs when
0	Signed branch comparison
1	Unsigned branch comparison
Don’t Care	beq, bne, non-branch instruction



Instruction	Name	Description	Type	Opcode	Funct3

`beq rs1 rs2 label`	Branch if EQual	`if (rs1 == rs2)`
		`PC = PC + offset`
			B	110 0011	000

`bge rs1 rs2 label`	Branch if Greater or Equal (signed)	`if (rs1 >= rs2)`
		`PC = PC + offset`
			B	110 0011	101

`bgeu rs1 rs2 label`	Branch if Greater or Equal (Unsigned)		B	110 0011	111

`blt rs1 rs2 label`	Branch if Less Than (signed)	`if (rs1 < rs2)`
		`PC = PC + offset`
			B	110 0011	100

`bltu rs1 rs2 label`	Branch if Less Than (Unsigned)		B	110 0011	110

`bne rs1 rs2 label`	Branch if Not Equal	`if (rs1 != rs2)`
		`PC = PC + offset`
			B	110 0011	001

Critical Path

- The longest path in the circuit
- Limits the maximum clock speed
- Can be measured from:
  - register to register
  - register to output (since the output typically also goes to a register)
  - input to register (since the input typically also comes from a register)
- clock period >= clk-to-q + max CL delay + setup time
  - Maximum cumulative delay of all combinational logic through a path (max CL delay)...
  - coming from the output of a register (clk-to-q)...
  - and ending at the input of another register (setup time)
  - Clock period must be *long enough* so that the longest delay (clk-to-q + max CL delay) is accounted for while satisfying the register timing requirement (setup time)

- When calculating delays, you don’t pass through a register!