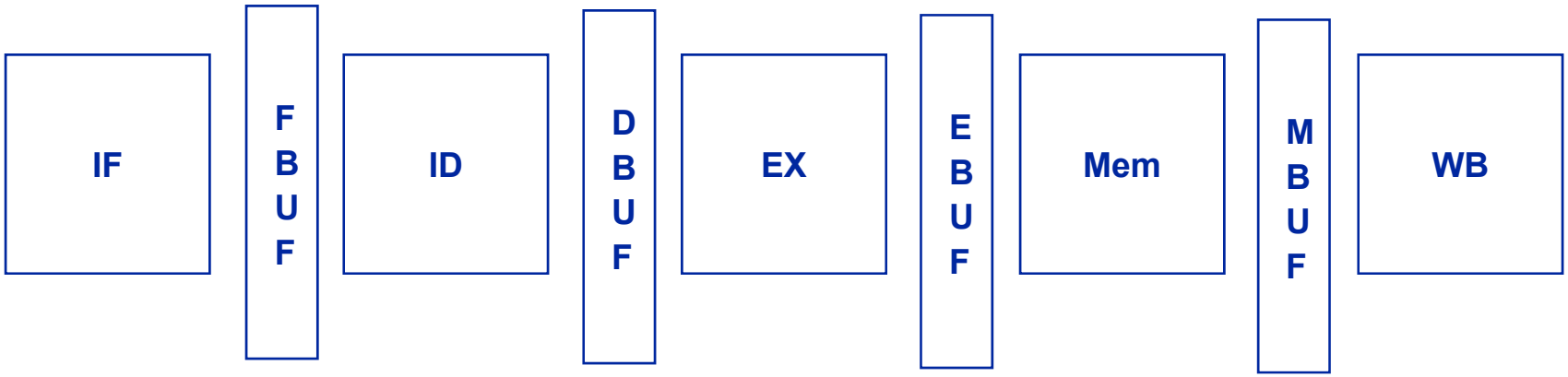
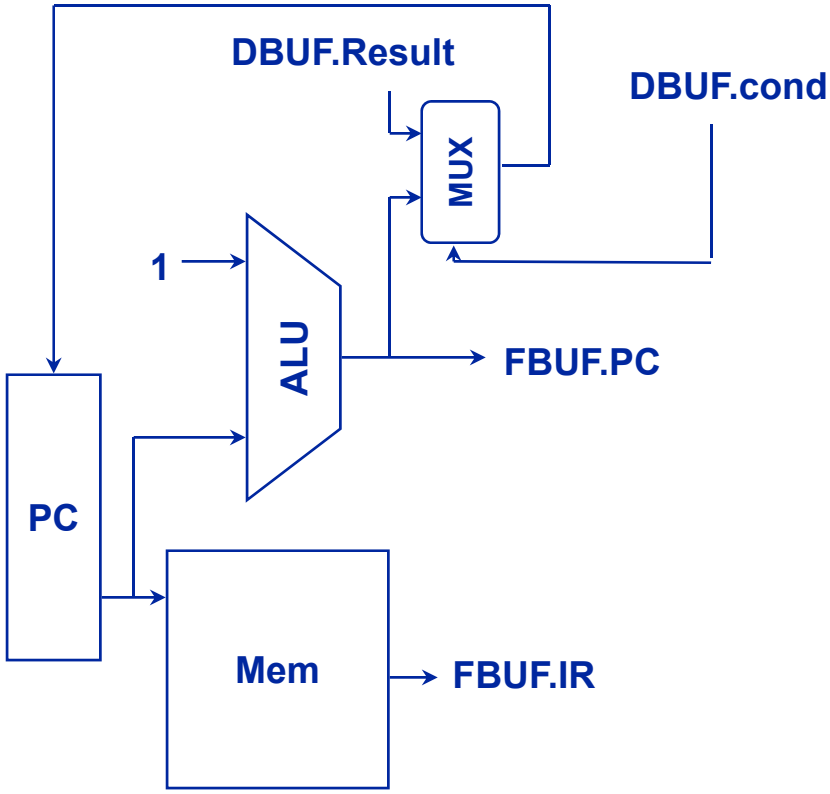
Pipeline top level



How to Execute an Instruction

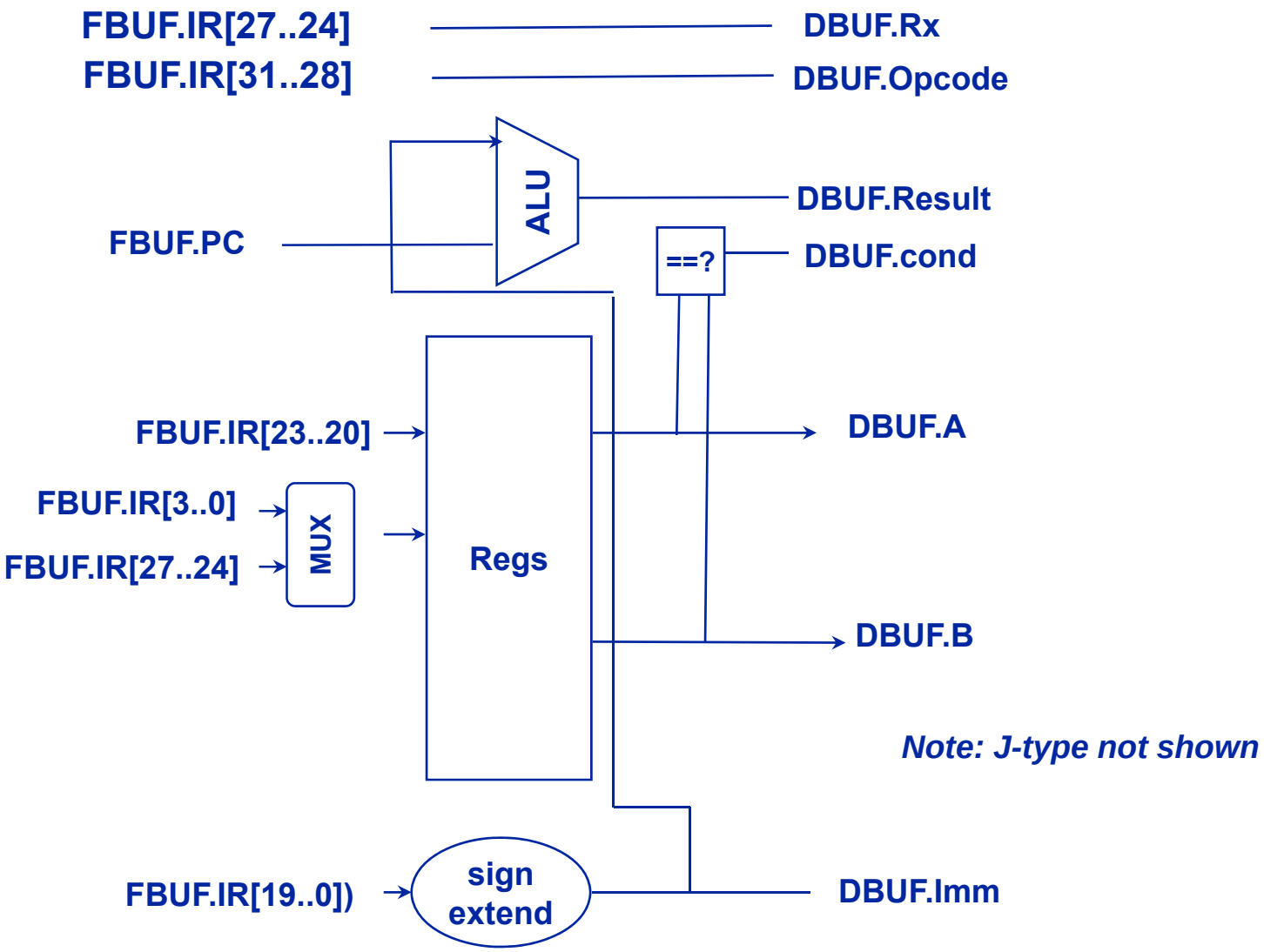
- Instruction fetch ("IF")
  - FBUF.IR = Mem[PC]
  - NPC = PC + 1
  - FBUF.PC = NPC
  - if (DBUF.cond) PC = DBUF.Result else PC = NPC



How to Execute an Instruction

- Instruction decode/Register fetch ("ID")
  - DBUF.opcode = FBUF.IR[31..28]
  - DBUF.A = Regs[IR[23..20]]
  - if (DBUF.opcode == 0100) // A "sw"
    - DUF.B = Regs[IR[27..24]]
  - Else
    - DUF.B = Regs[IR[3..0]]
  - DBUF.Imm = sign-extend(FBUF.IR[19..0])
  - DBUF.Rx = FBUF.IR[27..24]
  - DBUF.PC = FBUF.PC
  - DBUF.Result = FBUF.PC + Imm
  - DBUF.Cond = (A == B)
  - If (DBUF.opcode == 0110) // J-type
    - Regs[DBUF.IR[3..0]] = FBUF.PC
    - DBUF.Result = DBUF.A
    - DBUF.Cond = 1

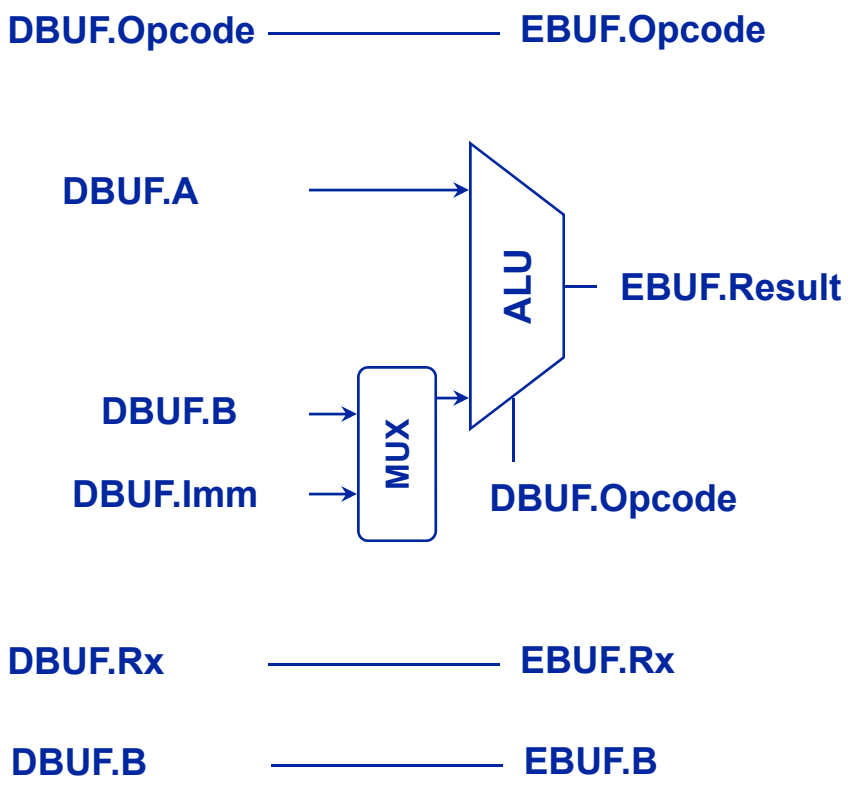
Instruction decode



Executing an Instruction (cont.)

- Execute ("EX")
  - EBUF.Rx = DBUF.Rx
  - EBUF.B = DBUF.B
  - EBUF.Opcode = DBUF.Opcode
  - I-type (EBUF.Opcode == lw, sw, addi, nandi):
    - EBUF.Result = DBUF.A *op* DBUF.Imm
  - I-type (EBUF.Opcode == BEQ)
    - Do nothing (done in ID) // for BEQ
  - R-type:
    - EBUF.Result = DBUF.A *op* DBUF.B

Execute stage



Executing an Instruction (cont.)

- Memory Access/Branch completion ("MEM")
  - MAR = EBUF.Result
  - Din = EBUF.B
  - if (EBUF.opcode = 0100) WtMem=1 // sw
  - else WtMem = 0 // lw
  - MBUF.Rx = EBUF.Rx
  - if (EBUF.opcode = 0011) MBUF.Result = Dout
  - else MBUF.Result = EBUF.Result
- Write back ("WB")
  - If (R-type or I-type and != sw, beq)
    - Regs[MBUF.Rx] = MBUF.Result