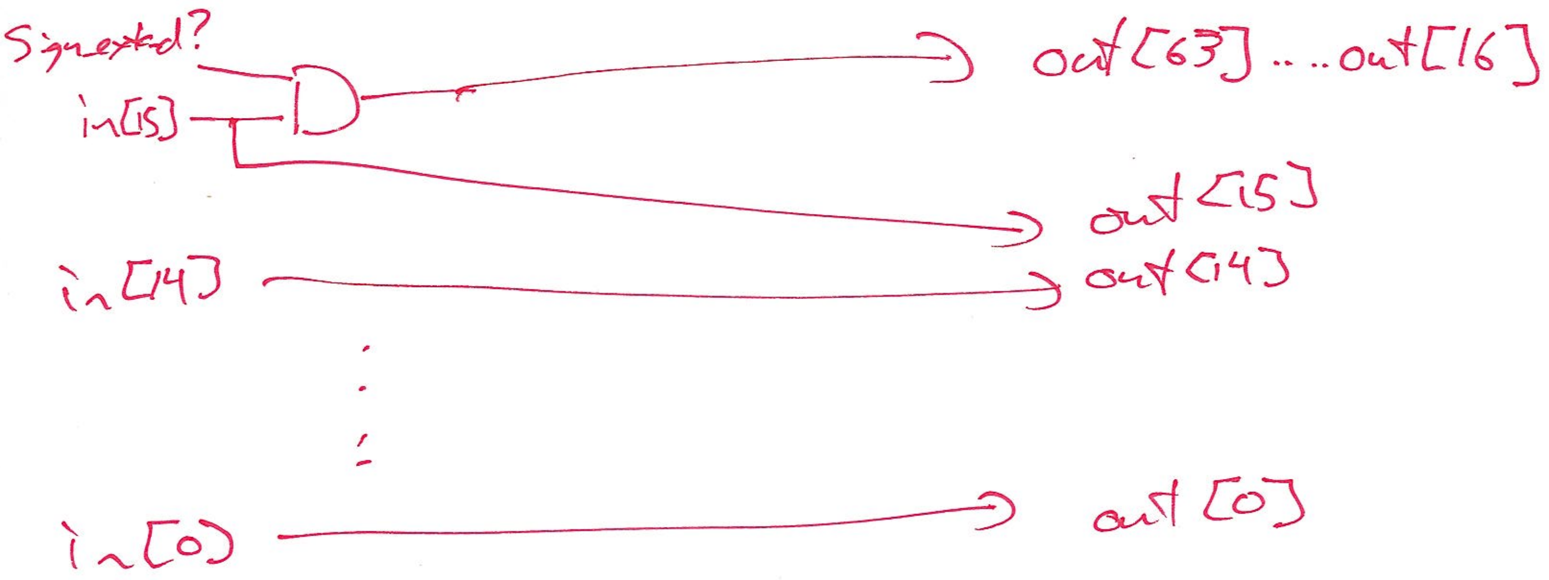
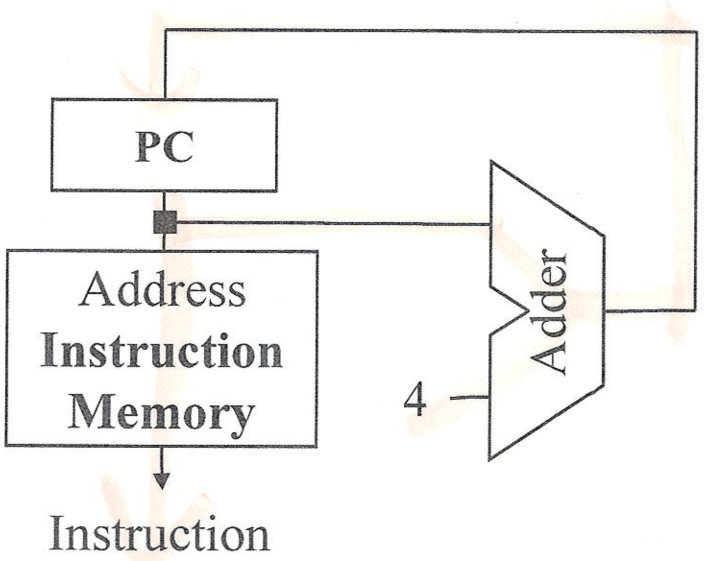
Review Problem 21

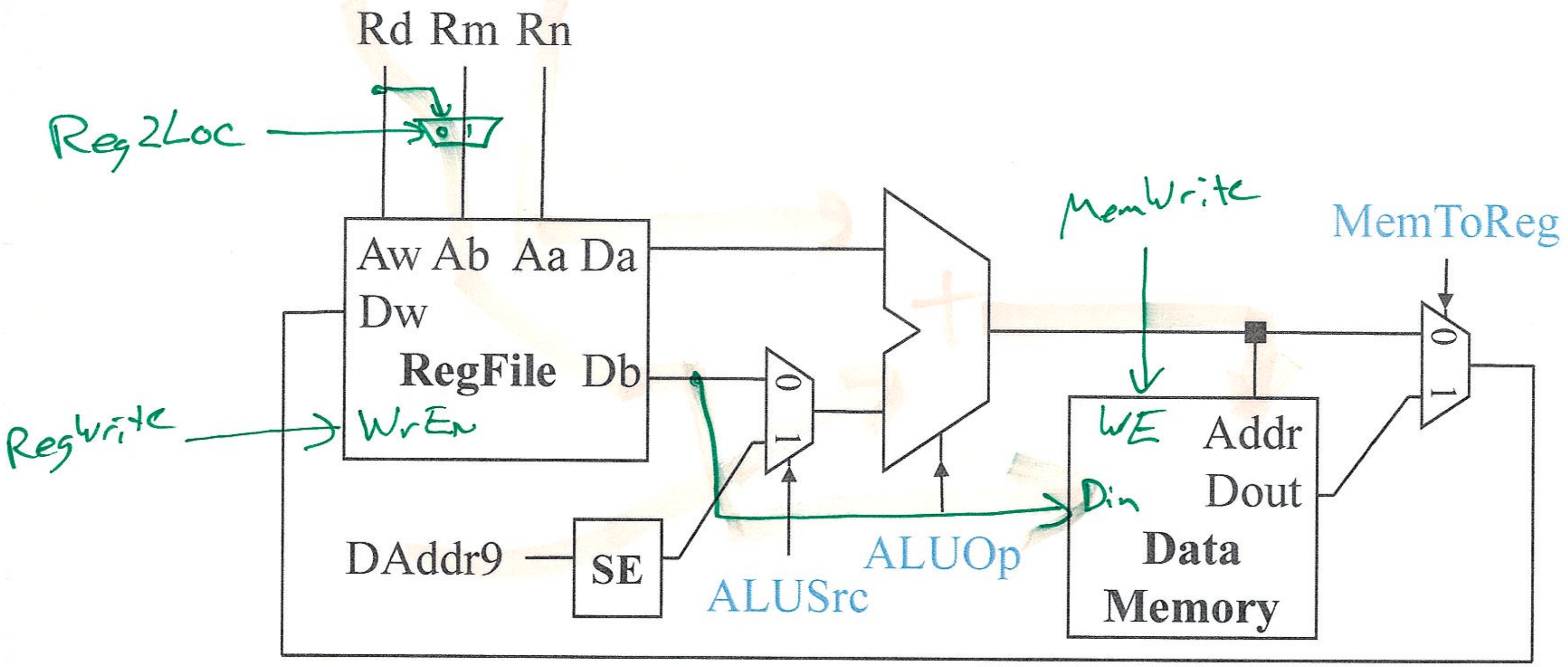
- Immediate vals for some instructions are sign-extended, while others are not. Build a 16bit to 64bit sign-extend unit that can handle both.



Datapath + Store

Addr = Reg[Rn] + SE(DAddr9);
Mem[Addr] = Reg[Rd];



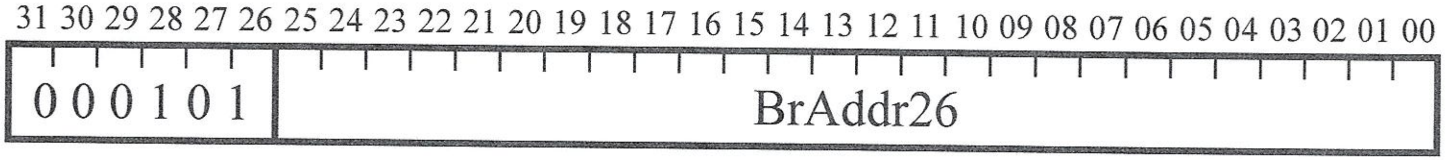


Branch RTL

Branch Instruction: `B BrAddr26`

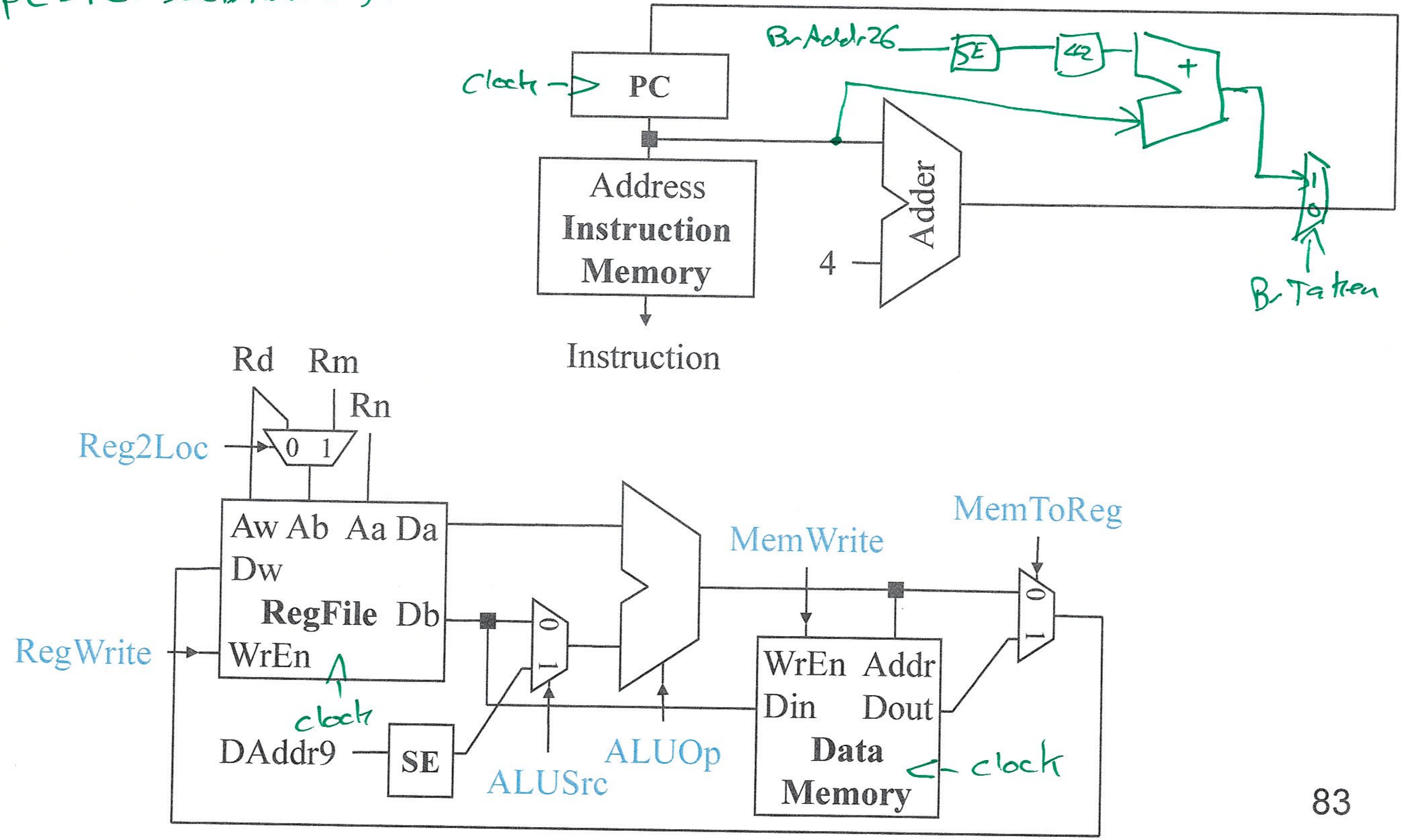
Instr = MEM[PC];

PC = PC + SignExtend(BrAddr26) << 2;



Datapath + Branch

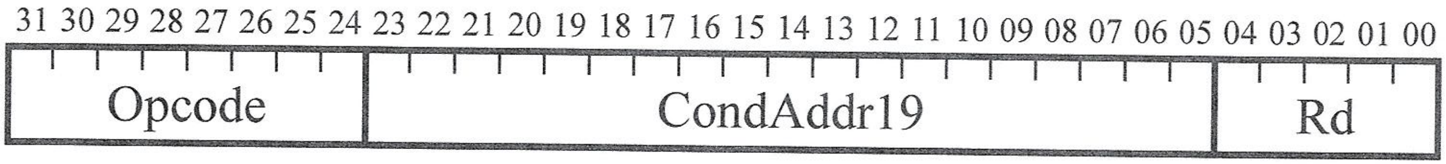
Instr = MEM[PC];
PC = PC + SE(BrAddr26)<<2



Conditional Branch RTL

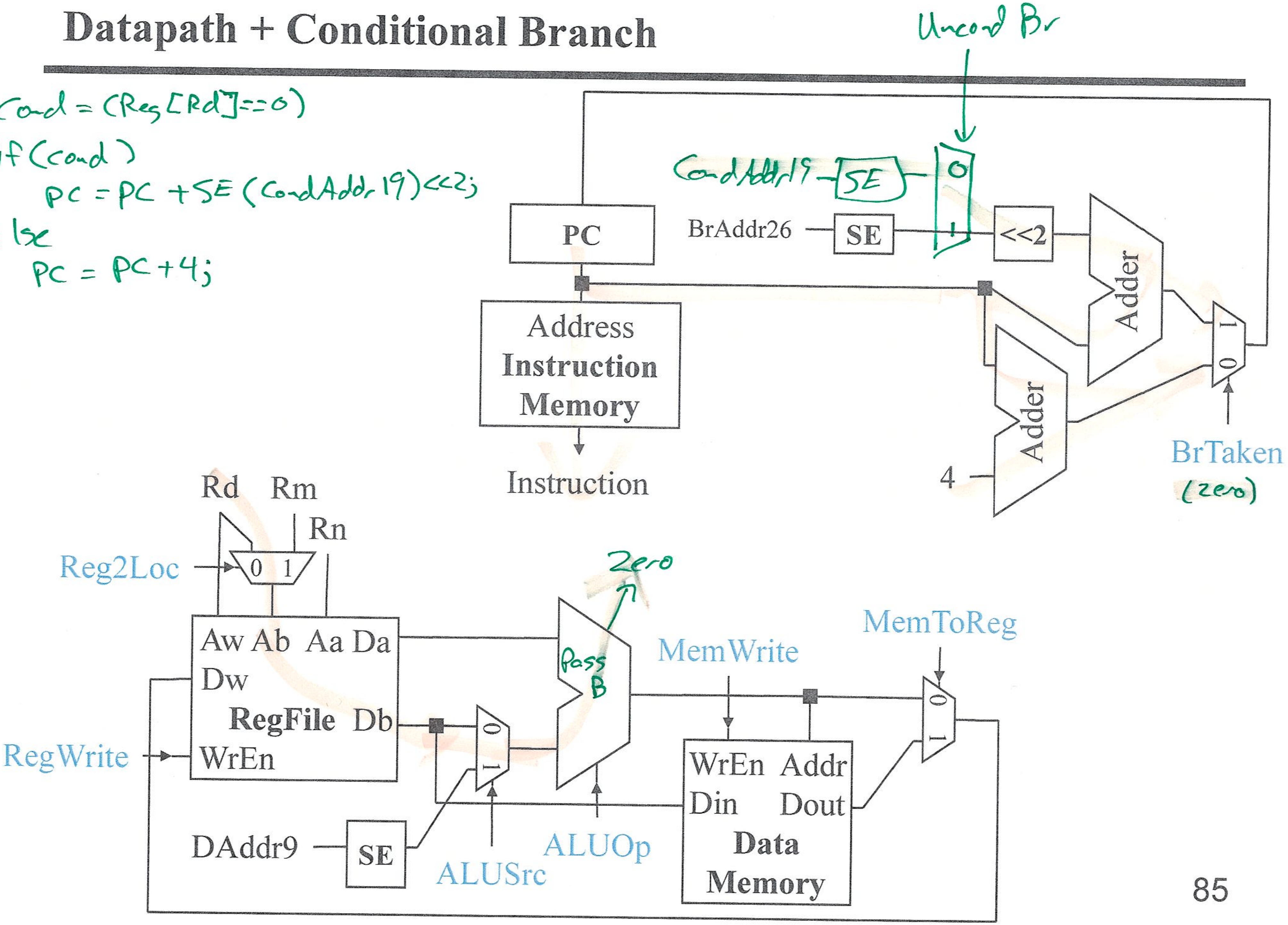
Conditional Branch Instruction: `CBZ Rd, CondAddr19`

```
Instr = Mem[PC];
Cond = (Reg[Rd] == 0);
 if (cond)
    PC = PC + Sign Extend ( CondAddr19) << 2;
else
    PC = PC+4;
```



Datapath + Conditional Branch

```
Cond = (Reg[Rd]==0)
if (cond)
    PC = PC + SE(CondAddr19)<<2;
else
    PC = PC+4;
```

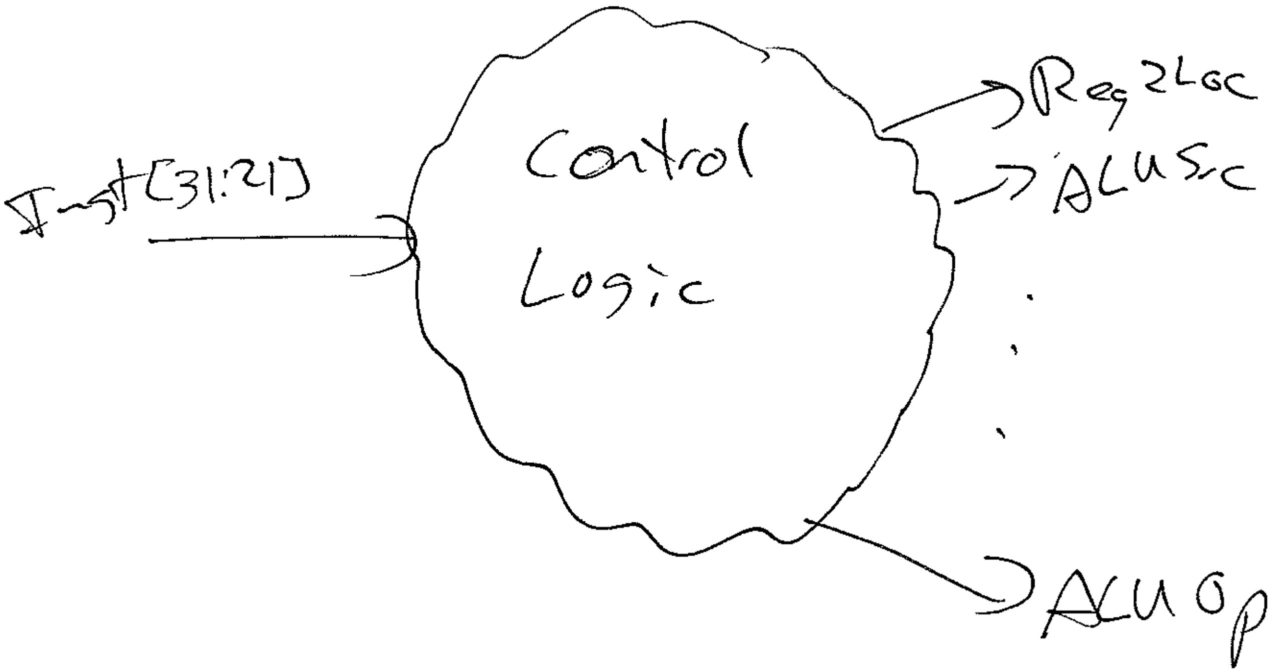


Control

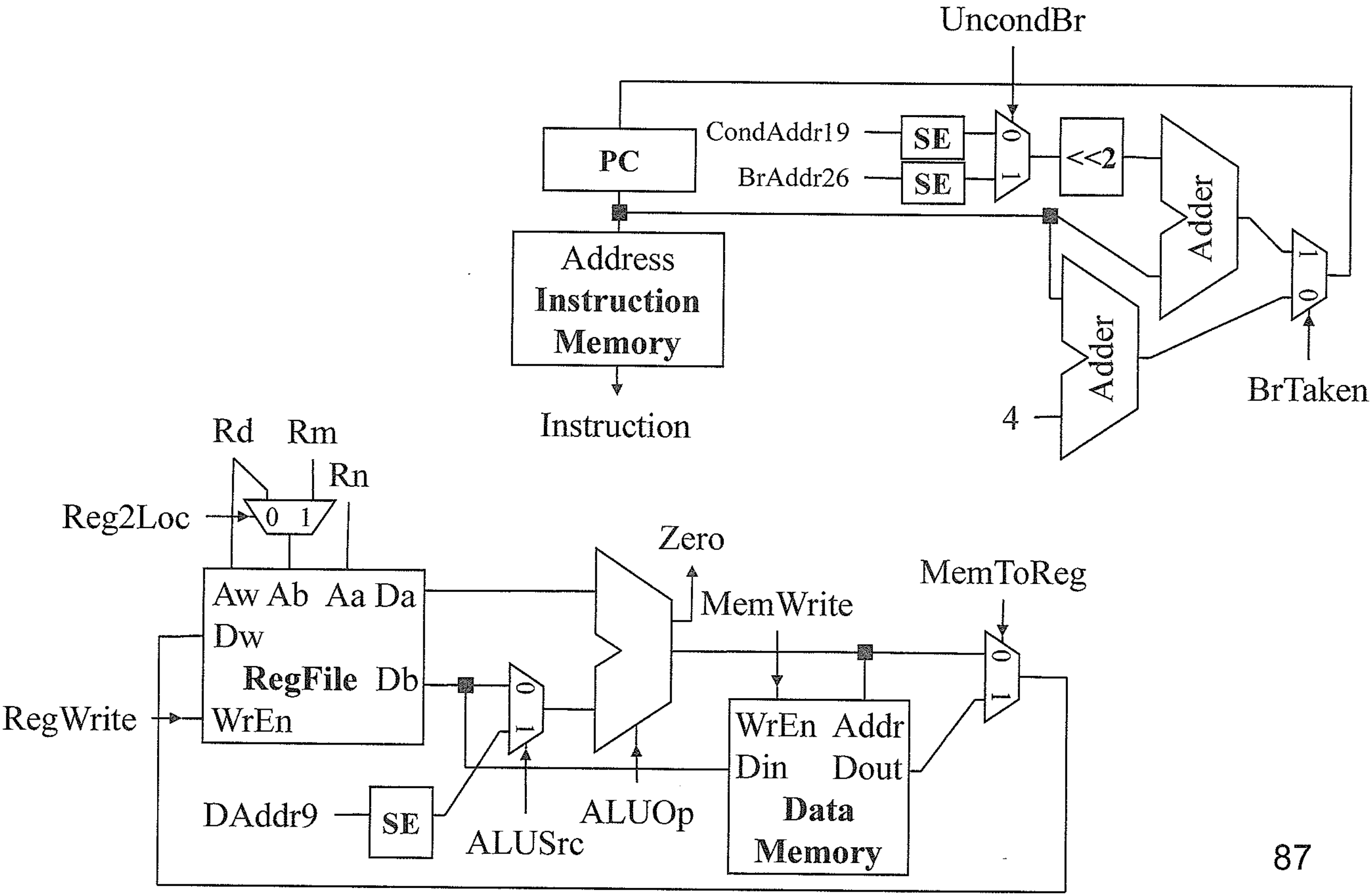
Identify control points for pieces of datapath

- Instruction Fetch Unit
- ALU
- Memories
- Datapath muxes
- Etc.

Use RTL for determine per-instruction control assignments


Inst[31:21]
Control Logic
Reg2Loc
ALUSrc
ALUOp

Complete Datapath



UncondBr
PC
CondAddr19
BrAddr26
SE
SE
0
1
<<2
Adder
Adder
1
0
4
BrTaken
Address
Instruction
Memory
Instruction
Rd
Rm
Rn
Reg2Loc
0
1
Zero
MemWrite
MemToReg
Aw Ab Aa Da
Dw
RegFile Db
WrEn
RegWrite
0
1
0
1
WrEn Addr
Din Dout
Data
Memory
DAddr9
SE
ALUSrc
ALUOp