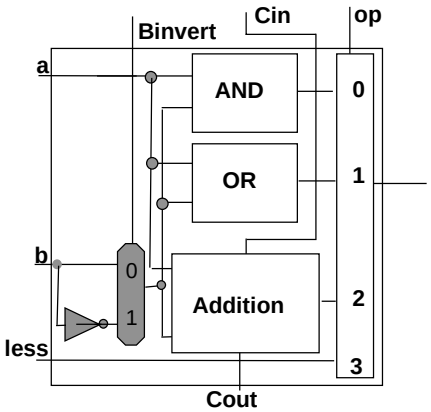
Completing the ALU Design

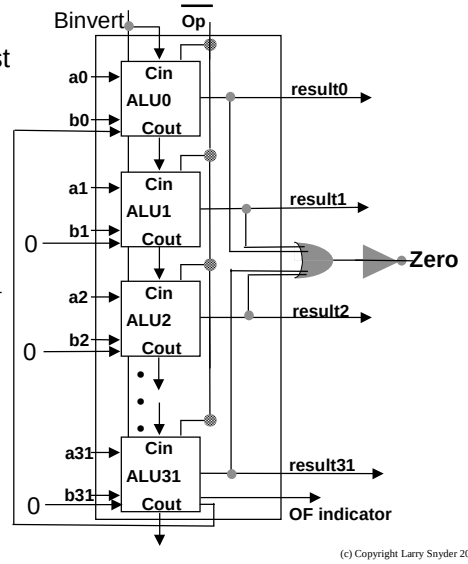
Adding functionality to an ALU involves incorporating new logic into the existing design cleanly

32-Bit ALU

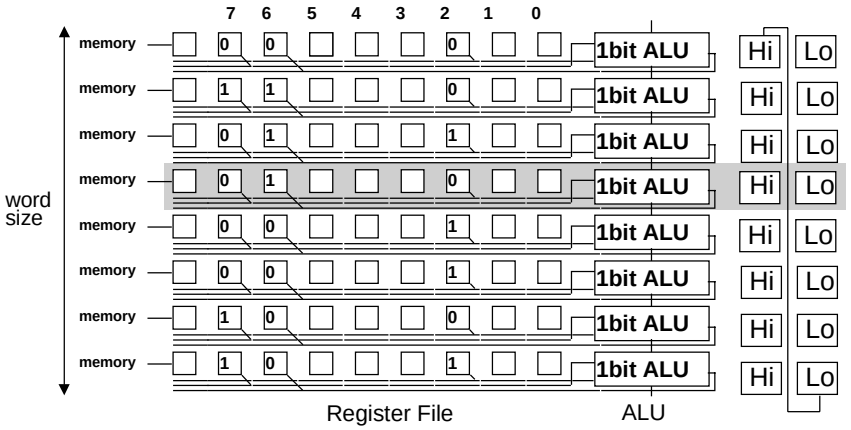
Basic design:

add, sub, and, or, slt and zero test



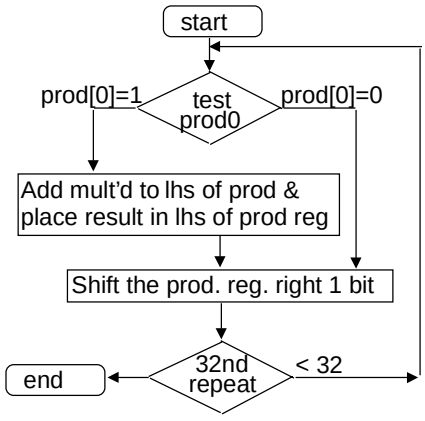


Placement In ALU



Final Algorithm

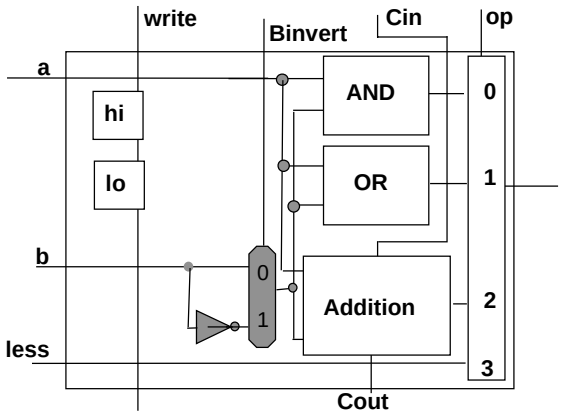
- Place multiplier in low side of product register



Mult'd	Product
0110	0000 0101
0110	0110 0101
0110	0011 0010
0110	0011 0010
0110	0001 1001
0110	0111 1001
0110	0011 1100
0110	0011 1100
0110	0001 1110

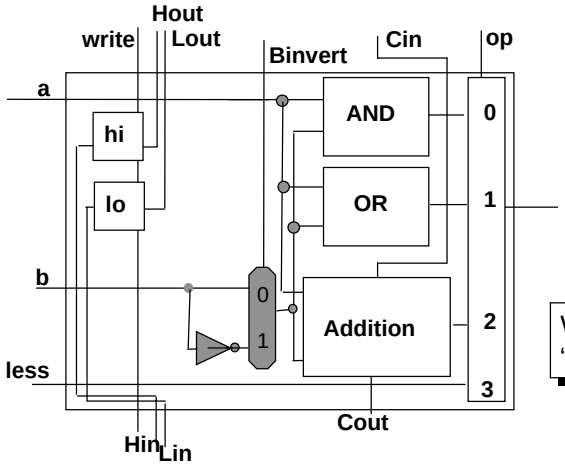
Add Multiply/Divide Registers

- Single bit for hi and lo



Construct a coherent register

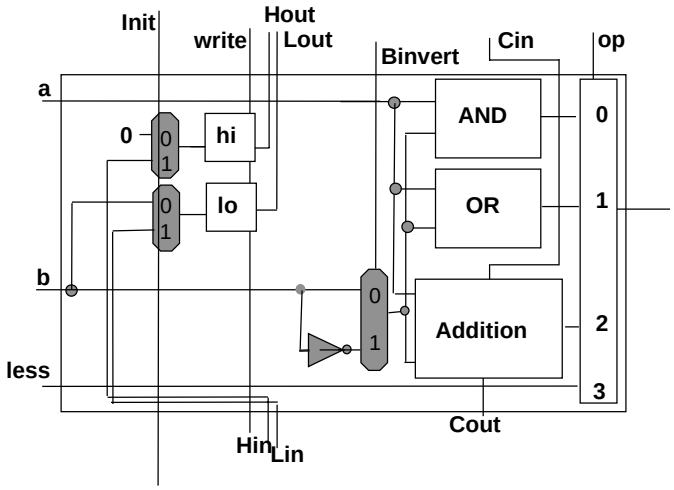
- Chain hi/lo together



What are the "end" inputs?

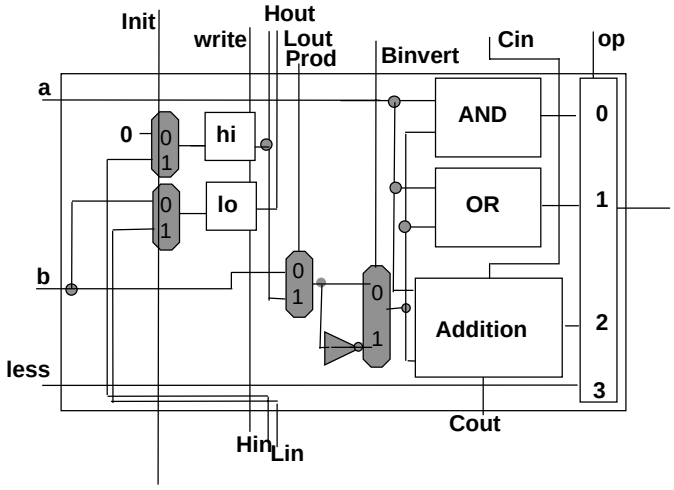
Revise Design For Initialization

- Set lo to b, set hi to zero



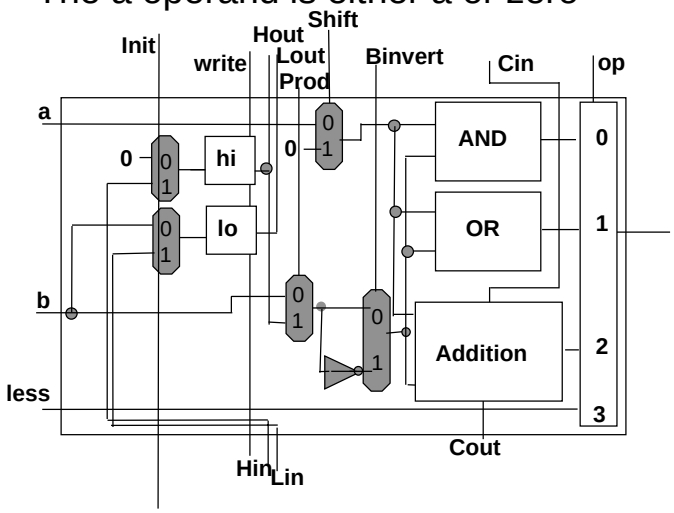
Set-up Addition Inputs

- Hi becomes b operand

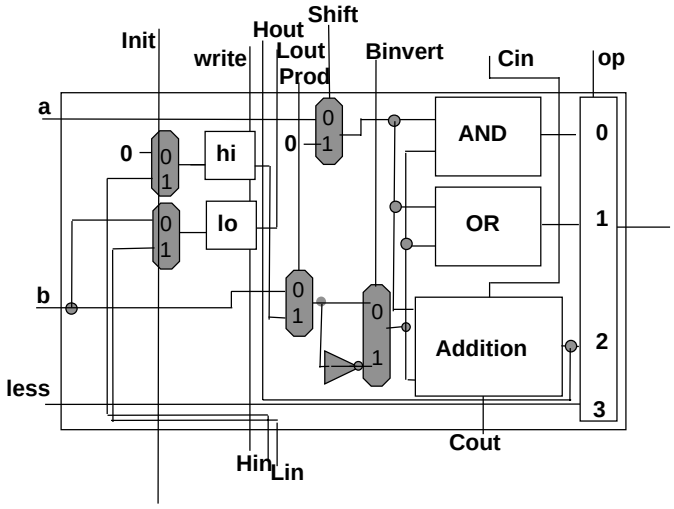


Set-up for Just Shift

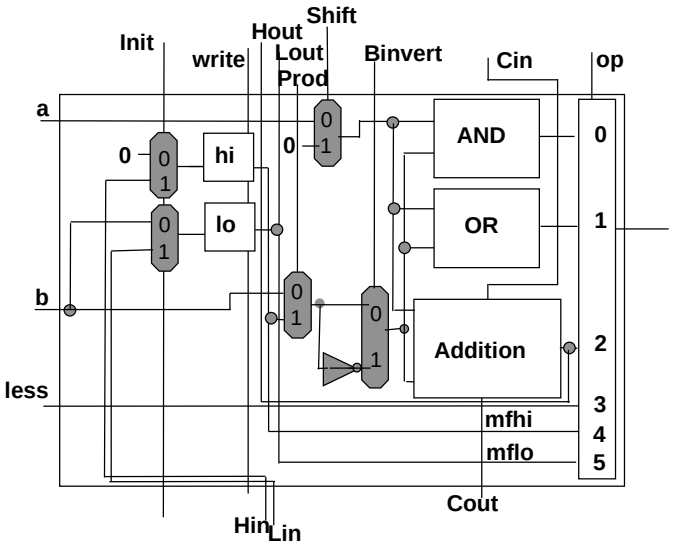
- The a operand is either a or zero



Capture Output



Move from Hi/Lo



Multiply

- The multiply operation does not reference the op mux
- `Mult` fires-up the control logic to perform the multiply
  - It first strobes init and write to initialize the register
  - It sets prod to 1
  - It iteratively sets Shift and write 32 times