Computer design

Example: Multiply step machine

P.J. Drongowski
SandSoftwareSound.net

Copyright © 1987-2013 Paul J. Drongowski

Multiply step machine

- Load / store, register to register ISA
- Register indirect addressing
- Perform integer multiply
  - Shift and add algorithm
  - Condition code (StepFlag) to sense bit
- Branch to direct address

Instruction format

31 – 29	28 – 26	25 – 23	22 – 0
Opcode	SRC	DST	IMMEDIATE

Instruction set

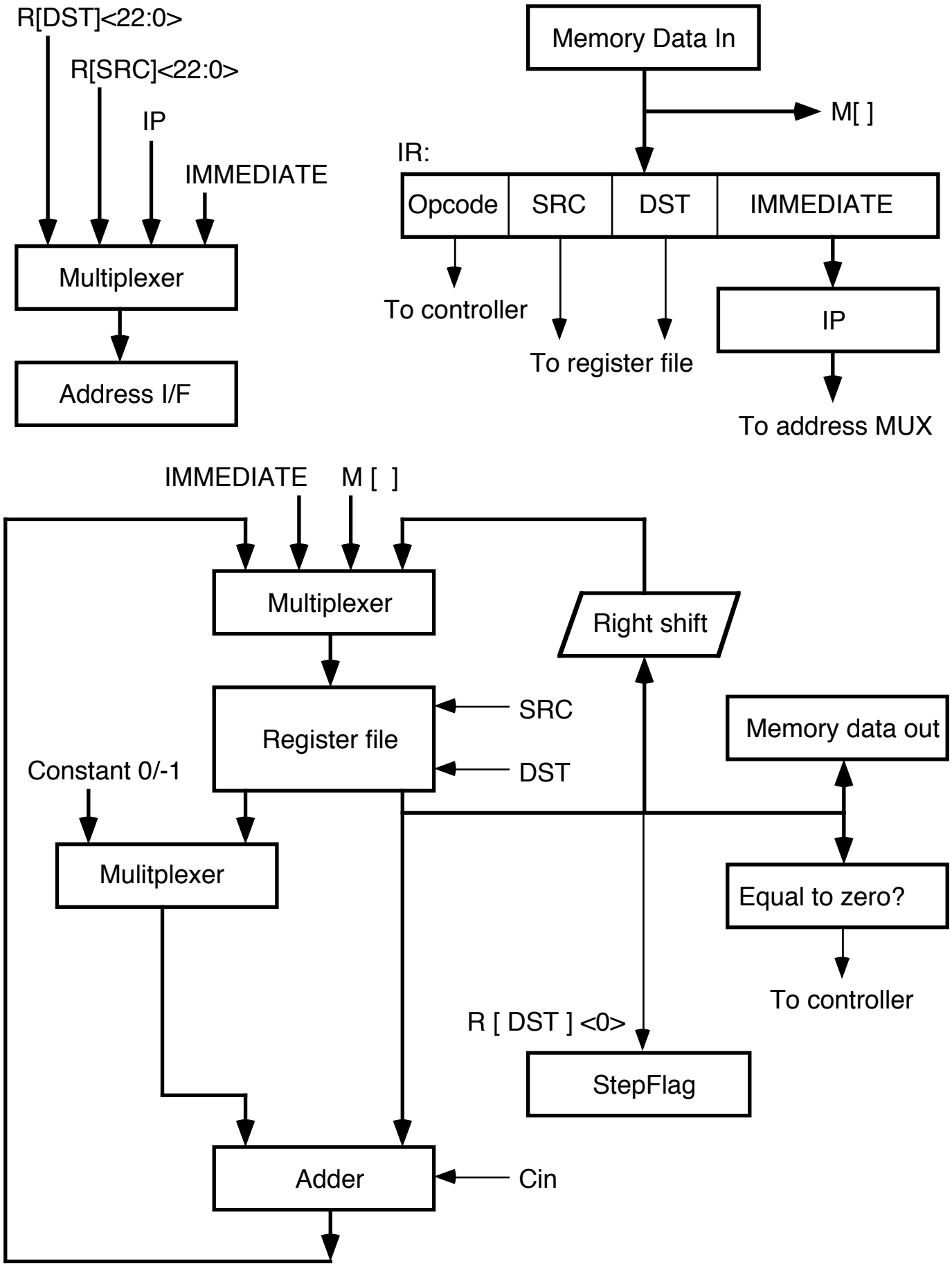
Opcode	Name	Behavior
0	Load	R[DST]←M[R[SRC]]
1	Store	M[R[DST]]←R[SRC]
2	LoadImm	R[DST]←IMMEDIATE
3	MulStep	**if** (StepFlag=1) **then** R[DST]←R[DST]+R[SRC]
4	Shift	StepFlag←R[DST]<0>; R[DST]←R[DST]÷2
5	Decr	R[DST]←R[DST]-1
6	BrZero	**if** (R[DST]= 0) **then** IP←IMMEDIATE
7	Branch	IP←IMMEDIATE

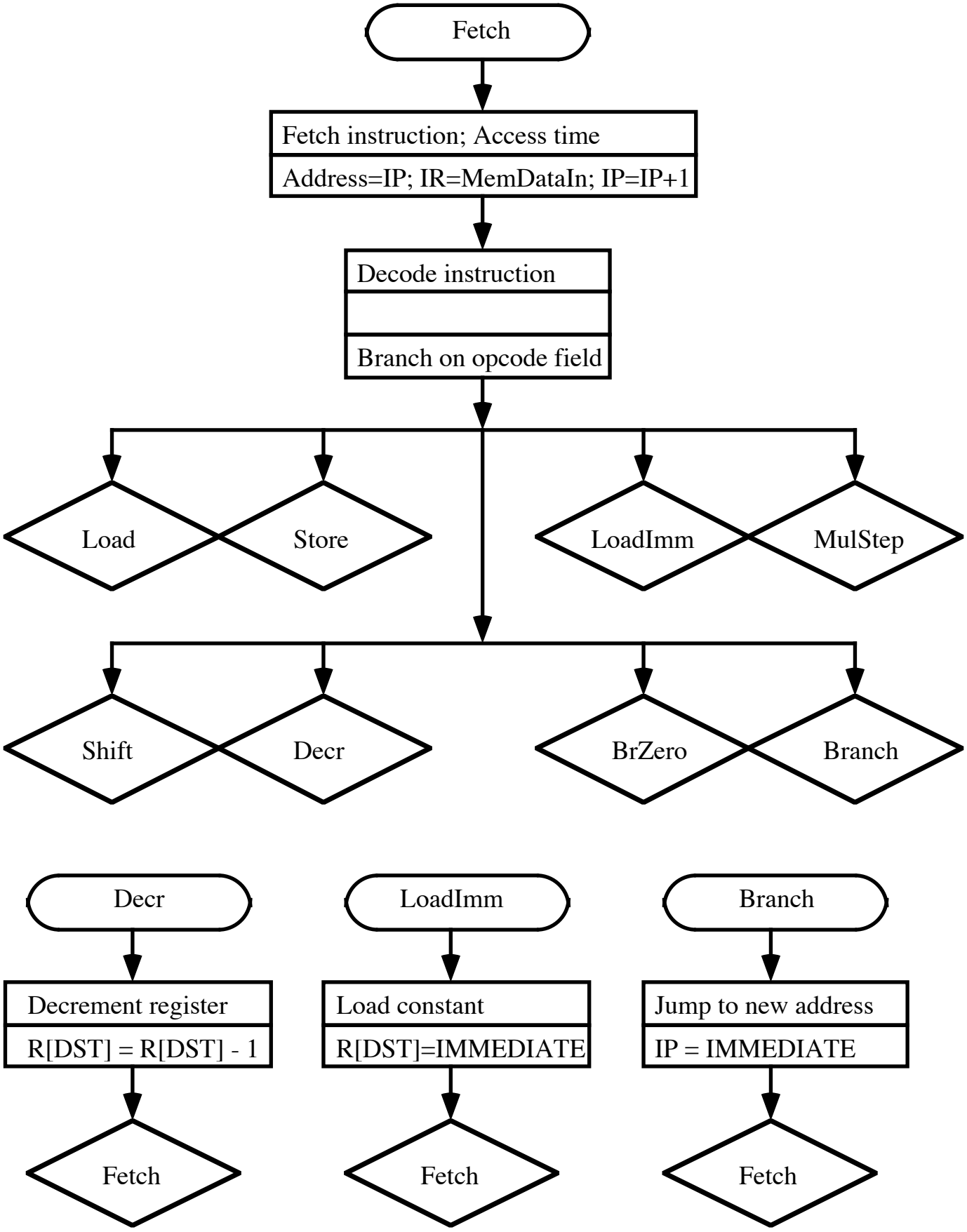
Observations

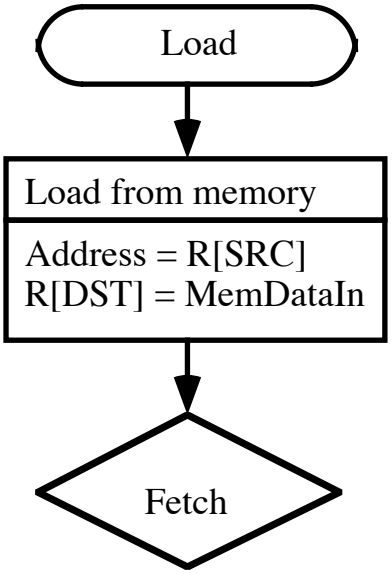
- Initial mapping
  - Allocate register (IR) to hold instruction
  - Make instruction pointer (IP) a counter
  - Need three memory interface ports
    - Memory address (output)
    - Memory data out
    - Memory data in
  - Send IMMEDIATE field direct to address output port
  - Read registers during fetch and begin computation
  - Control adder carry-in from microcode
  - Shift via wires

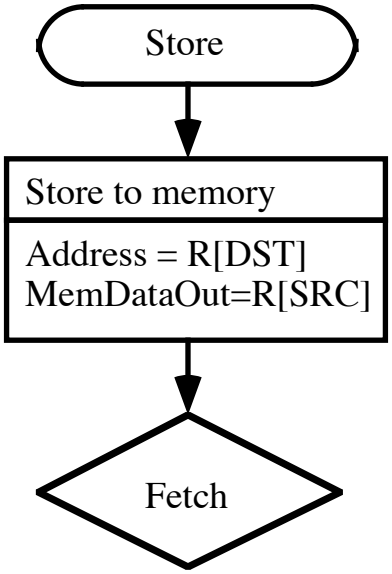
- Controller
  - Microcode is nearly an extravagance
  - Build three state sequential machine controller
    - State 0 - Fetch
    - State 1 - Execute
    - State 2 - Write result
  - Map opcode field directly to register operations
    - IP = IMMEDIATE if:
      - Opcode = 6 & R[DST] = 0
      - Opcode = 7
    - IP = IP + 1 if Opcode $\in$ {0, 1, 2, 3, 4, 5}
  - First steps toward pipelining

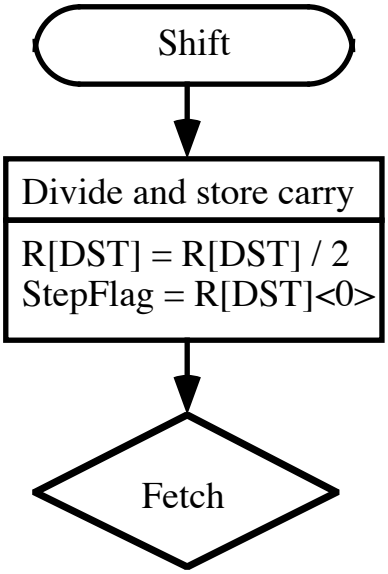
- Pipelining
  - Allow delayed branch
  - Need register forwarding
  - Add pipeline registers for IP and IR

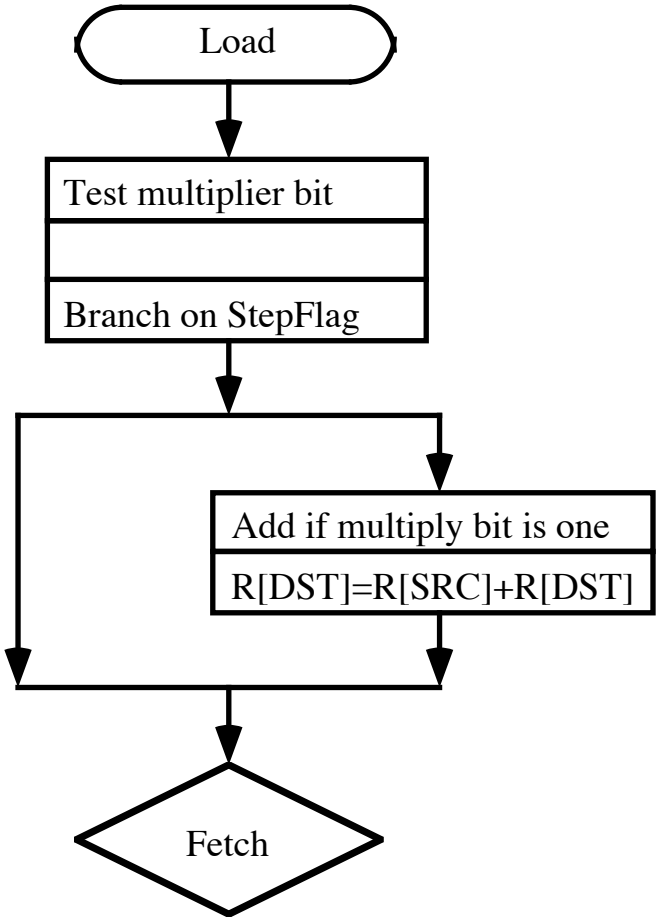

R[DST]<22:0>
R[SRC]<22:0>
IP
IMMEDIATE
Multiplexer
Address I/F
Memory Data In
M[ ]
IR:
Opcode
SRC
DST
IMMEDIATE
To controller
To register file
IP
To address MUX
IMMEDIATE
M [ ]
Multiplexer
Right shift
Register file
SRC
DST
Memory data out
Constant 0/-1
Mulitplexer
Equal to zero?
To controller
R [DST] <0>
StepFlag
Adder
Cin

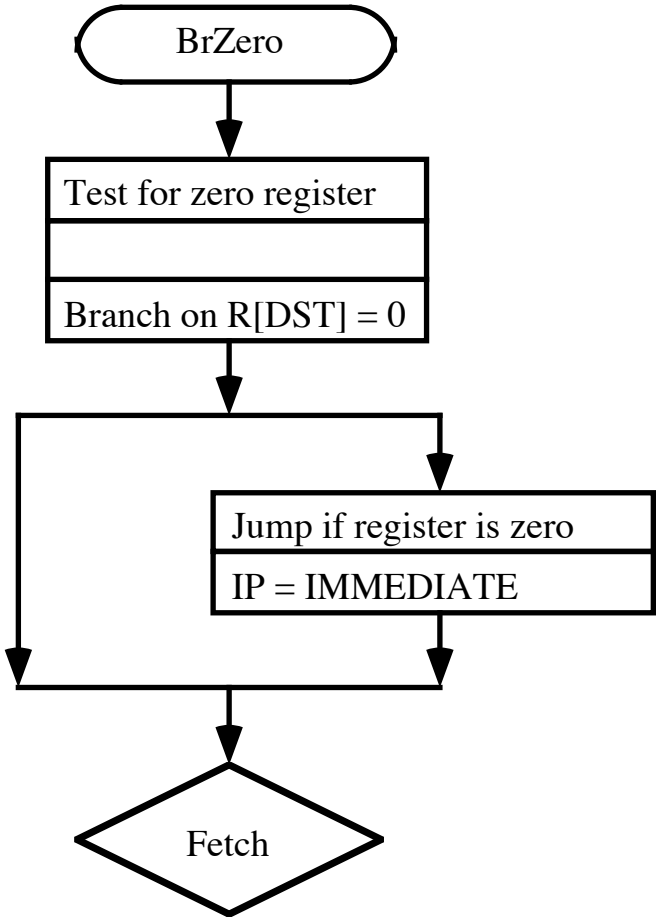

Fetch
Fetch instruction; Access time
Address=IP; IR=MemDataIn; IP=IP+1
Decode instruction
Branch on opcode field
Load
Store
LoadImm
MulStep
Shift
Decr
BrZero
Branch
Decr
Decrement register
R[DST] = R[DST] - 1
Fetch
LoadImm
Load constant
R[DST]=IMMEDIATE
Fetch
Branch
Jump to new address
IP = IMMEDIATE
Fetch


Load
Load from memory
Address = R[SRC]
R[DST] = MemDataIn
Fetch


Store
Store to memory
Address = R[DST]
MemDataOut=R[SRC]
Fetch


Shift
Divide and store carry
R[DST] = R[DST] / 2
StepFlag = R[DST]<0>
Fetch

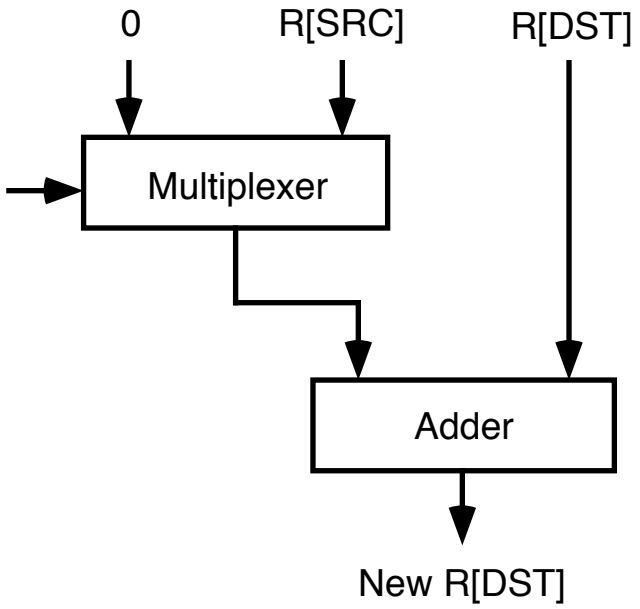

Load
Test multiplier bit
Branch on StepFlag
Add if multiply bit is one
R[DST]=R[SRC]+R[DST]
Fetch


BrZero
Test for zero register
Branch on R[DST] = 0
Jump if register is zero
IP = IMMEDIATE
Fetch

Eliminate branches

- Step toward pipelining
- Execute all instructions in three phases/stages
- Branch replacement
  - Conditional transfer
  - Select result based on opcode and condition

```
if (StepFlag == 1)
```




```
if (R[DST] == 0)
```

- Perform increment as intrinsic operation
- Conditionally transfer IMMEDIATE to IP
- See "Observations."