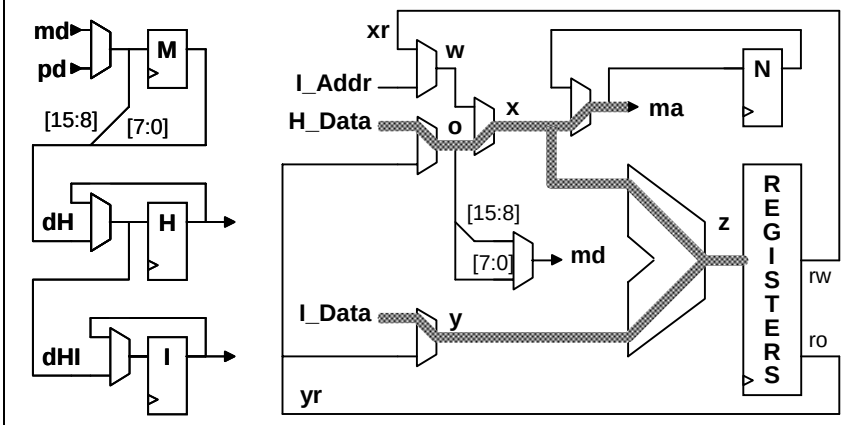
Powerup:

(BOOT2, RESET2)

0 => ma

0 + 0 => PC

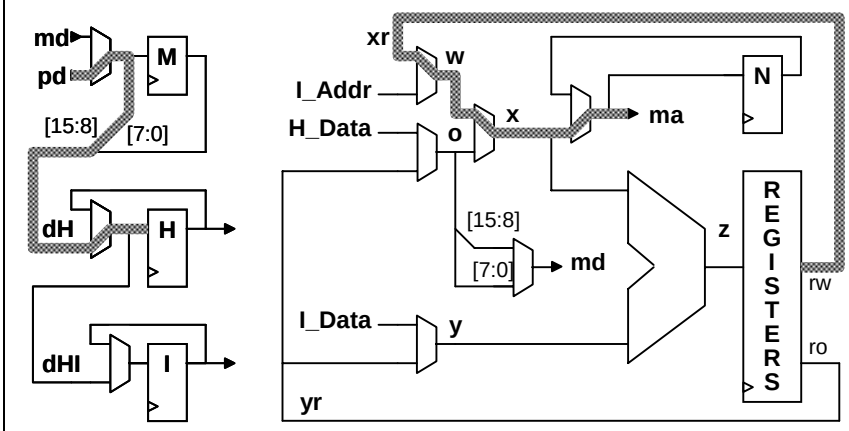


Boot read:

(BOOT2)

PC => ma

pd => H[15:8]



Boot write:

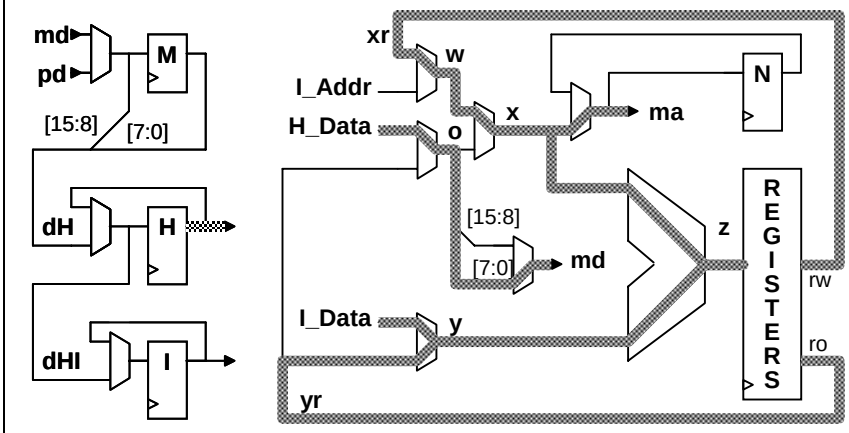
(BOOT2, RISE)

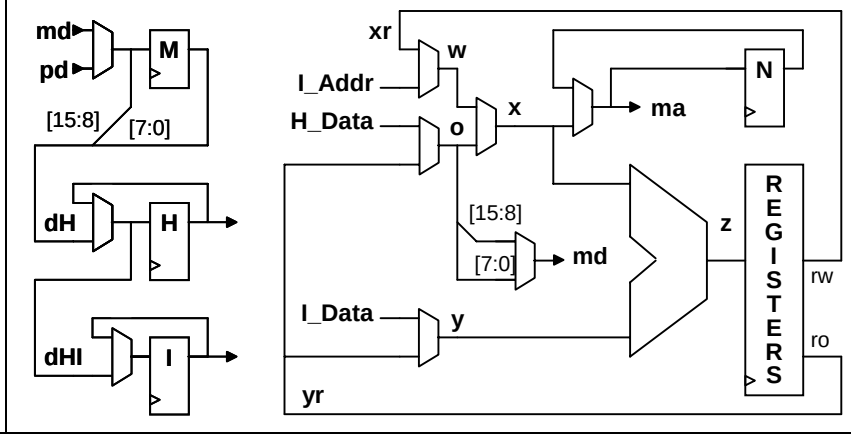
PC => ma

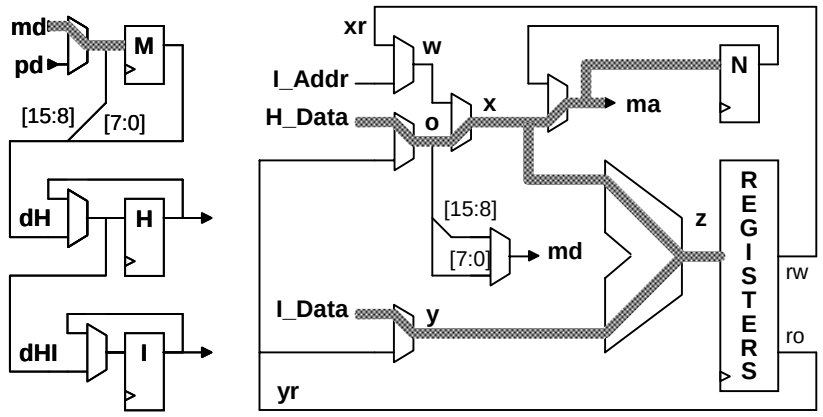
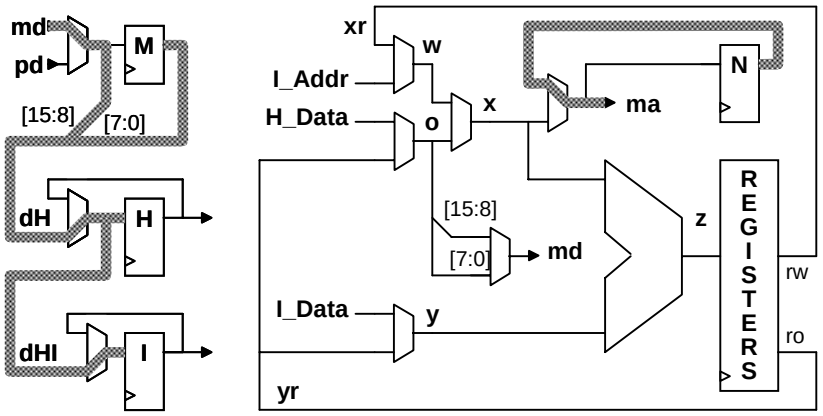
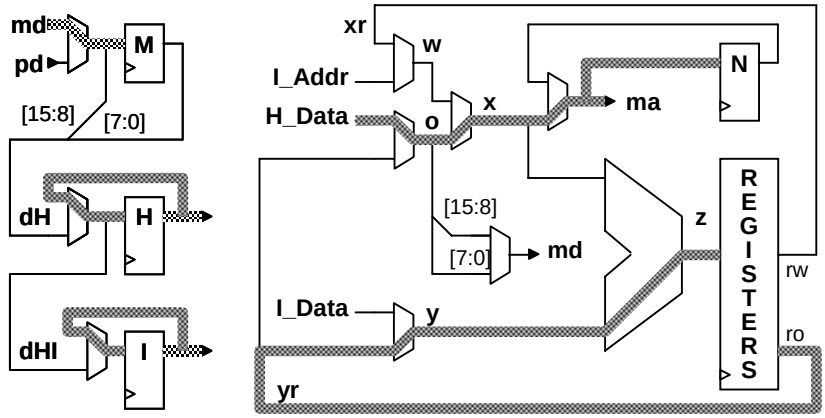
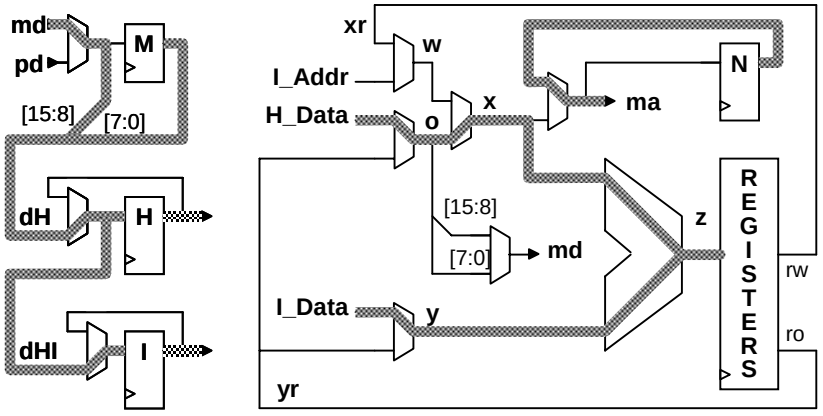
H[15:8] => md

PC + 1 => PC

write





Restart low: (RESET2) 0 => ma, N 0 + 2 => PC md => M	
Restart high: (RESET2, HIGH) N => ma md.M => H, l done	
Call low: (call_op) H => ma, N PC => RA md => M	
Call high: (call_op, HIGH) N => ma H + 2 => PC md.M => H, l done	

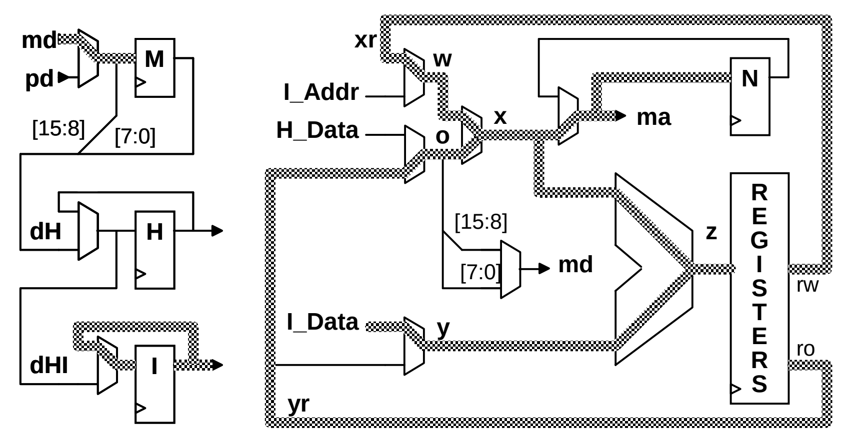
ALU low:

(alu_ops)

PC/RA => ma, N

PC/RA + 2 => PC

md => M



ALU high:

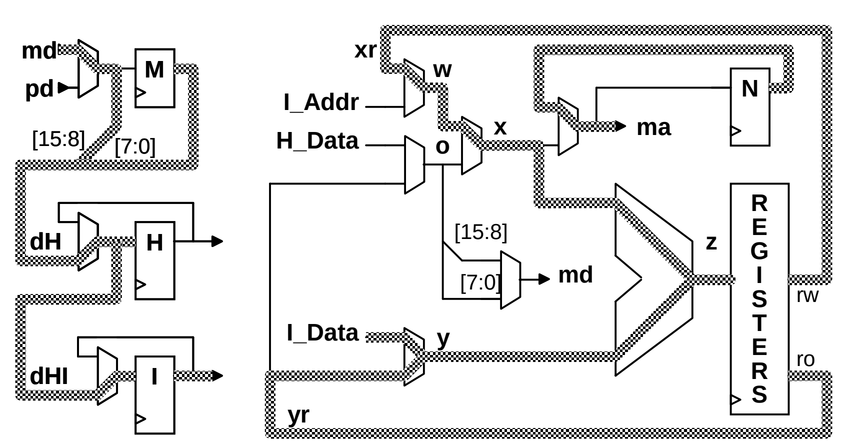
(alu_ops, HIGH)

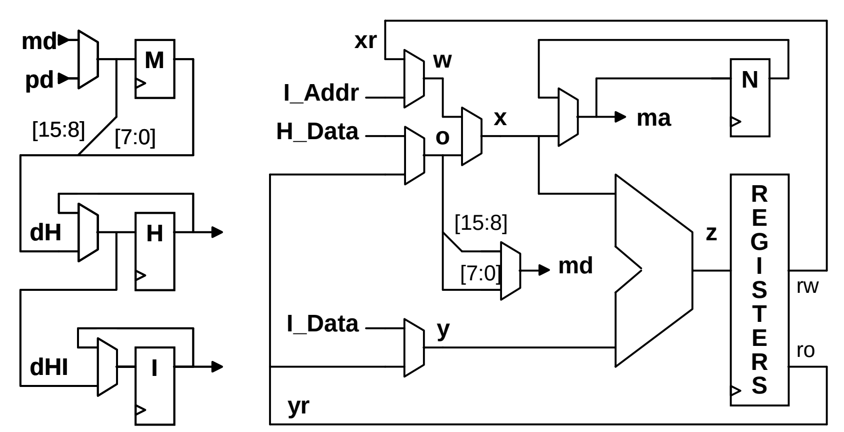
N => ma

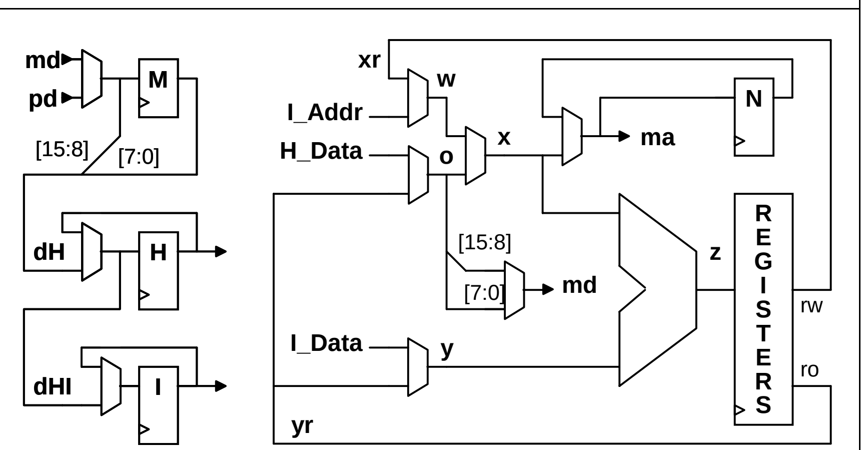
if (Ok)
  Rx op Ry/short/dist => Rx

md.M => H, I

done







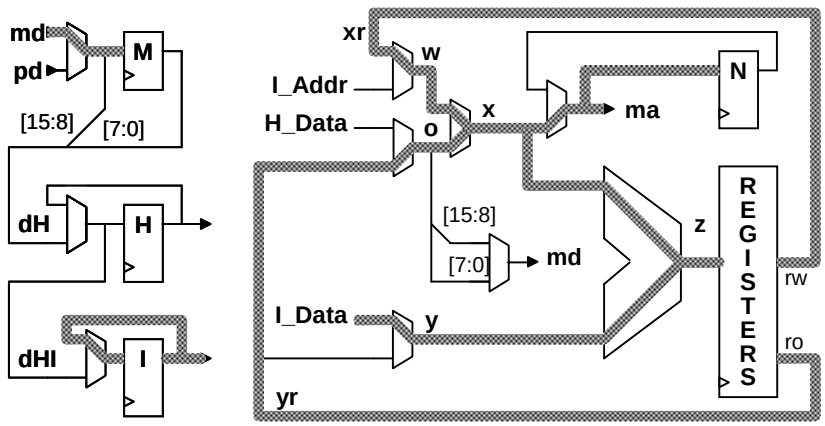
Write low fetch:

(mem_ops, Op[r] == write)

PC/RA => ma, N

PC/RA + 2 => PC

md => M



Write high fetch:

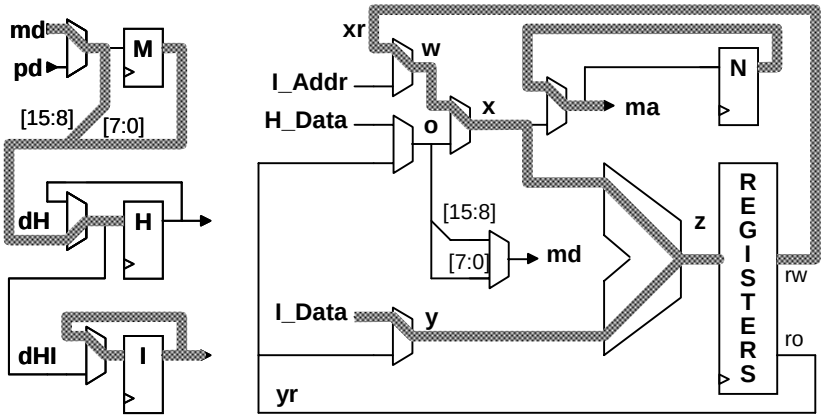
(mem_ops, Op[r] == write, HIGH)

N => ma

if (Ok && Op[xxx] != FL)
  Rx +/- stride => Rx

md.M => H

next



Write low data:

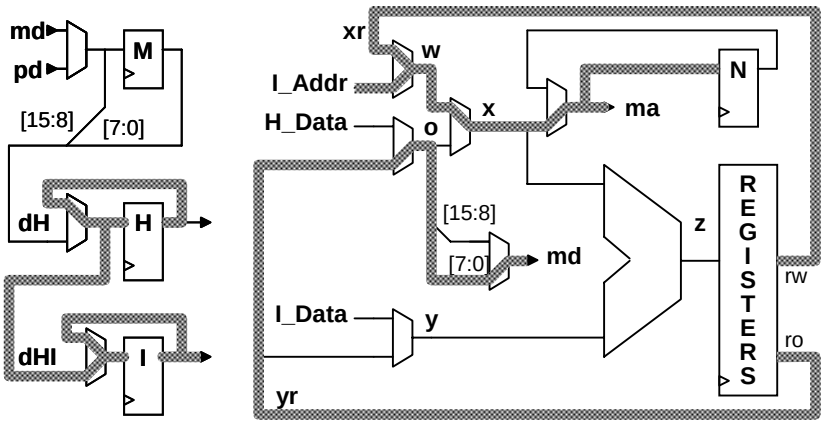
(mem_ops, Op[r] == write, SECOND)

Rx/direct => ma, N

Ry[7:0] => md, P

if (Ok) write

if (Op[b] == byte)
  H => H, I
  done



Write high data:

(mem_ops, Op[r] == write, SECOND, HIGH)

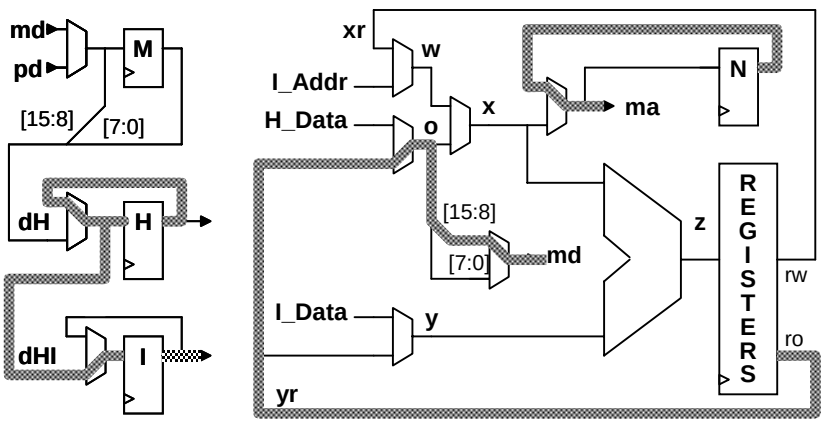
N => ma

Ry[15:8] => md

if (Ok) write

H => H, I

done



Read low data:

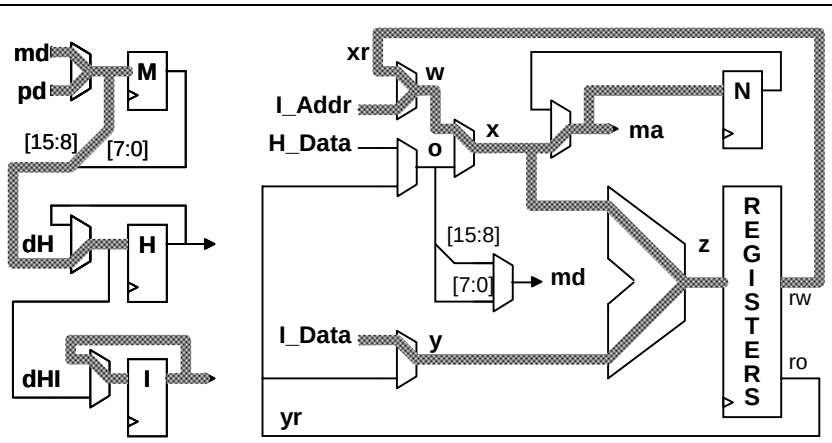
(mem_ops, Op[r] == read)

Rx/direct => ma, N

md => H

if (Ok && Op[xxx] != FL)
  Rx +/- stride => Rx

if (Op[b] == byte)
  pd/md => H[15:8]
  next

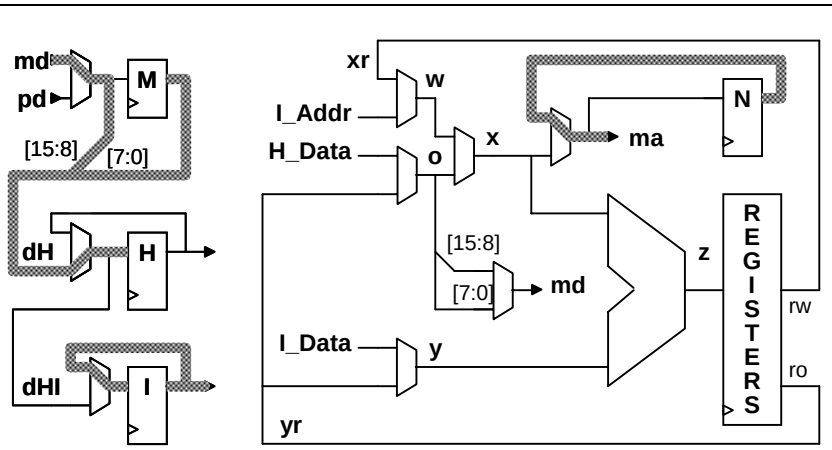


Read high data:

(mem_ops, Op[r] == read, HIGH)

md.M => H

next



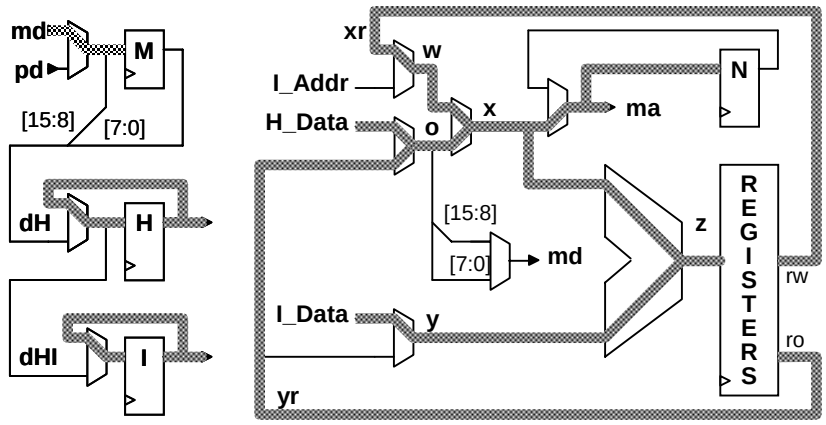
Read low fetch:

(mem_ops, Op[r] == read, SECOND)

if (Ok && Op[yyy] == PC/RA)
  H => ma, N
  H + 2 => PC

else
  PC/RA => ma, N
  PC/RA + 2 => PC

md => M



Read high fetch:

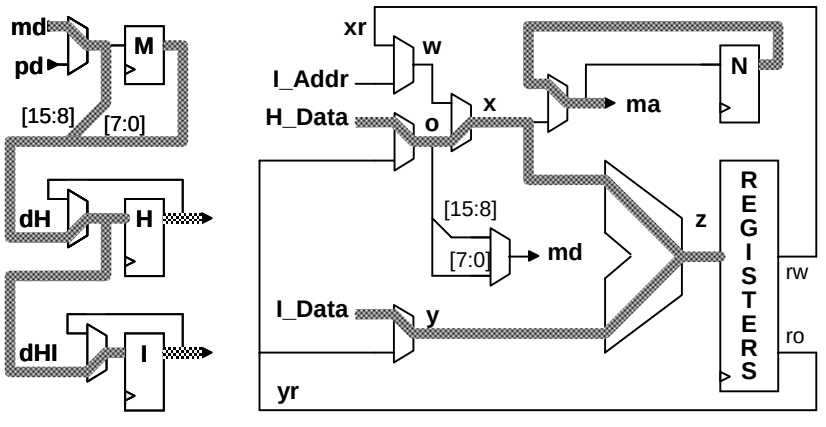
(mem_ops, Op[r] == read, SECOND, HIGH)

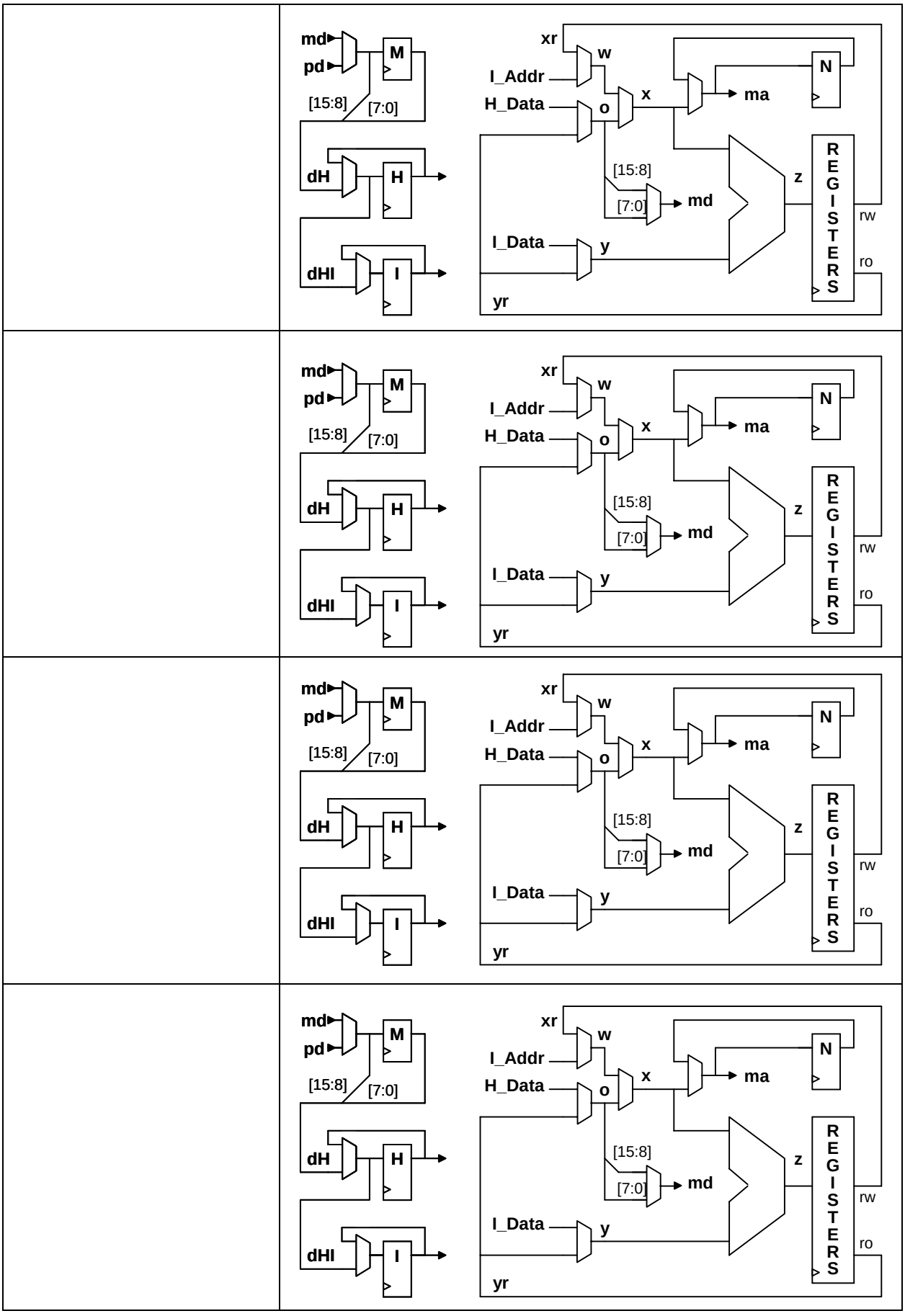
N => ma

if (Ok && Op[yyy] != PC/RA)
  H + 0 => Ry

md.M => H, I

done




md
pd
M
[15:8]
[7:0]
dH
H
dHl
I
xr
w
I_Addr
H_Data
o
x
N
ma
[15:8]
[7:0]
md
z
REGISTERS
rw
ro
I_Data
y
yr