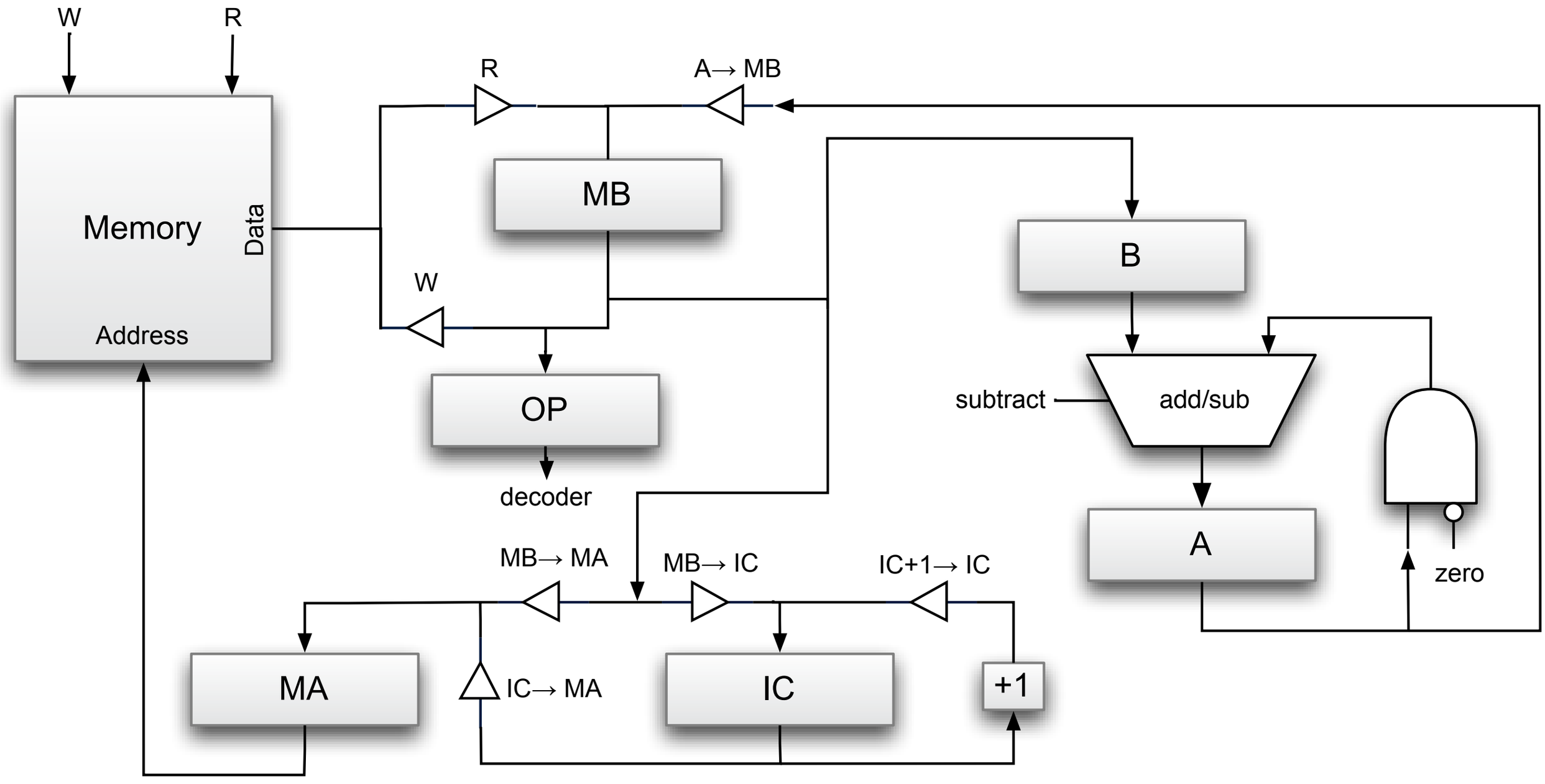
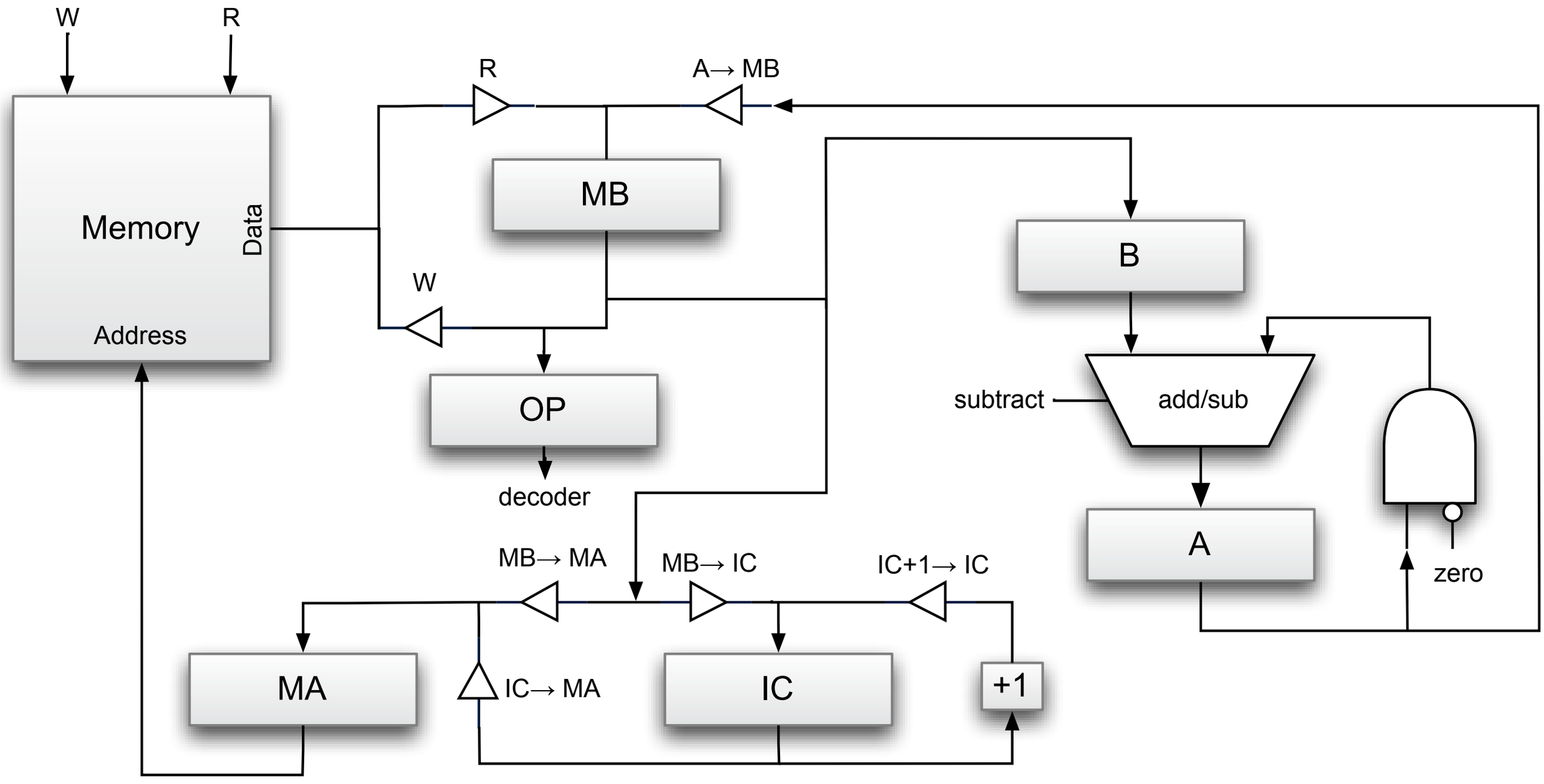
∑niac datapath

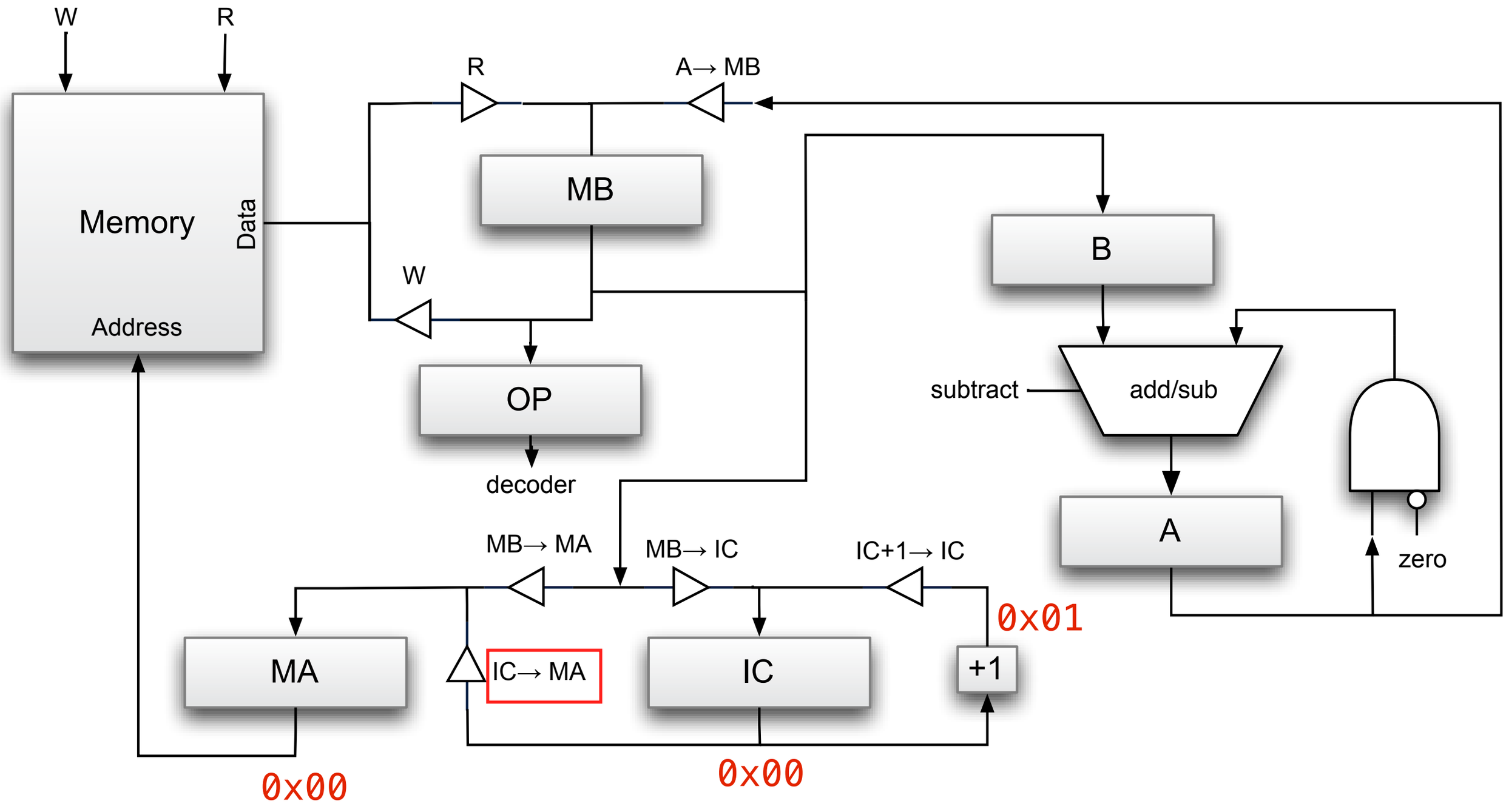


∑niac datapath



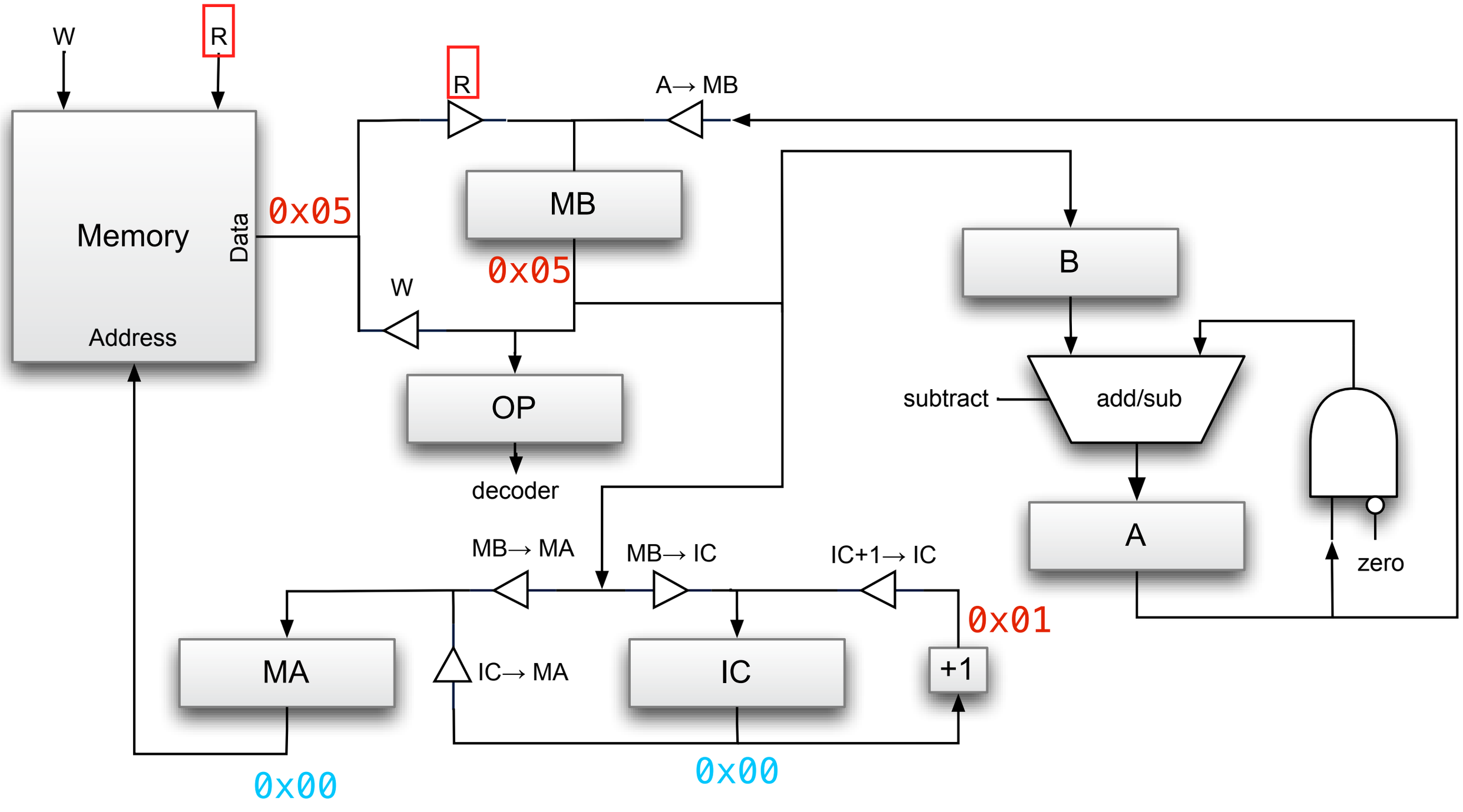

```
ld A, C3
```

∑niac datapath



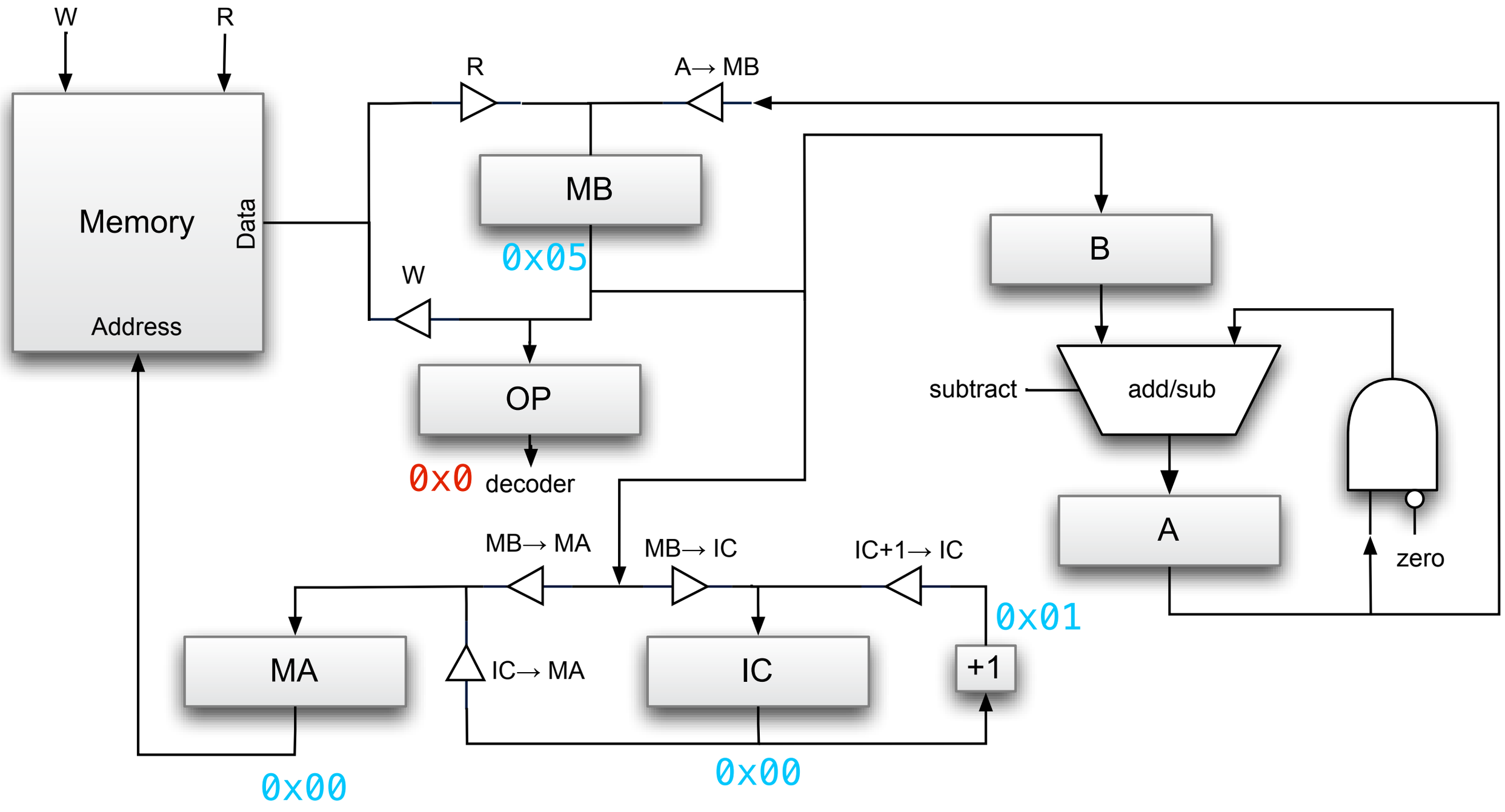
`ld A, C3` fetching instruction

∑niac datapath



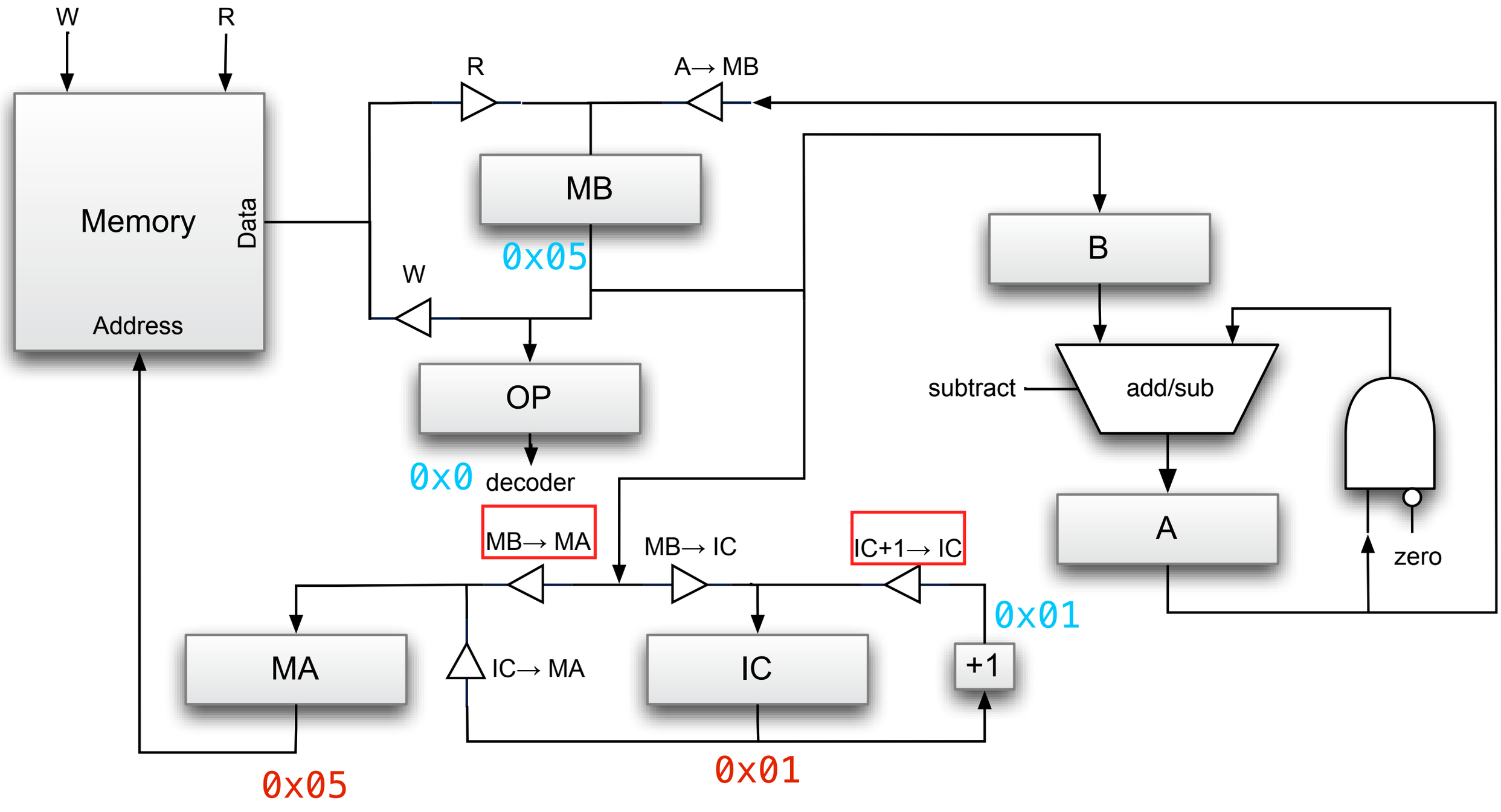
`ld A, C3` fetching instruction

∑niac datapath



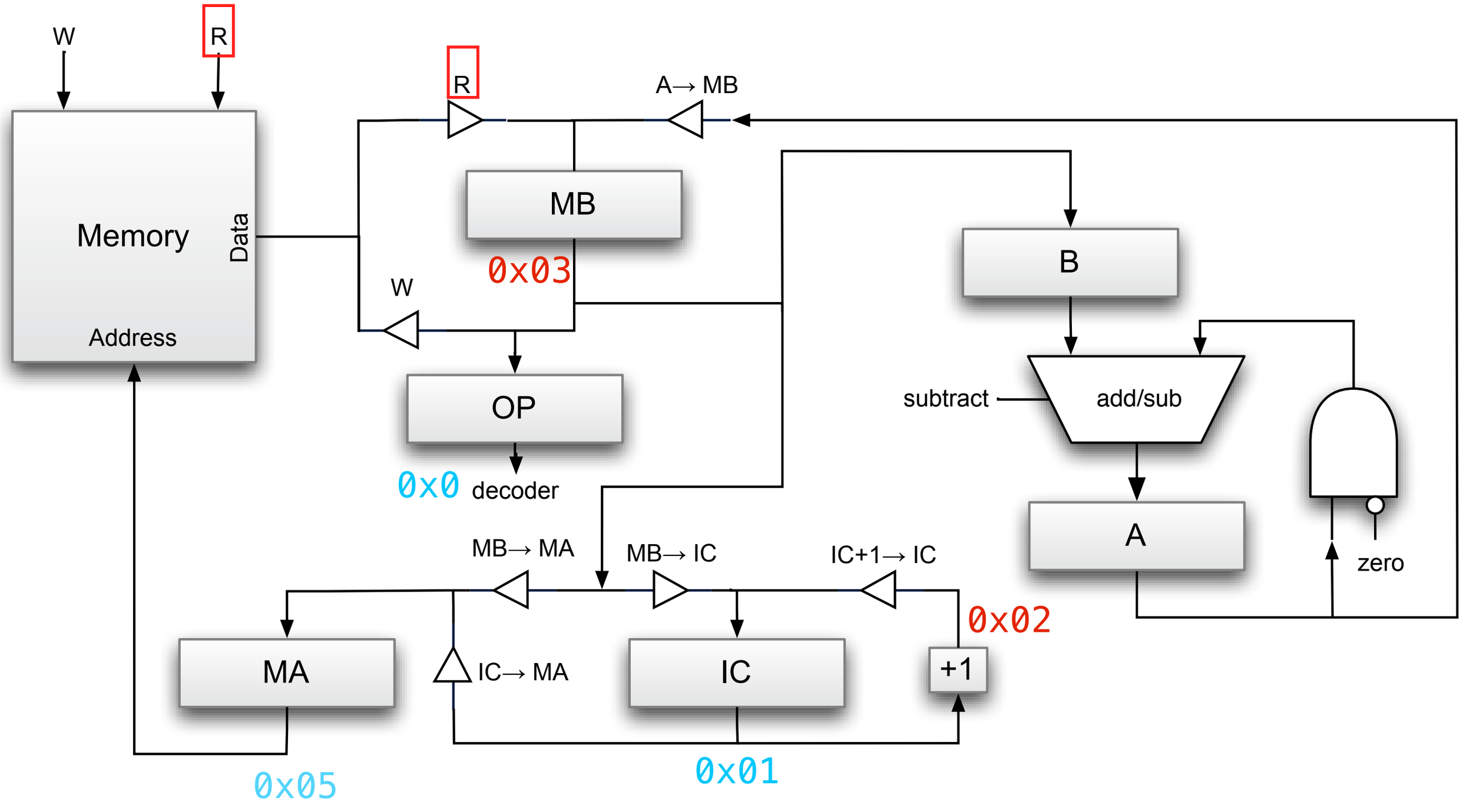
`ld A, C3` decoding instruction

∑niac datapath



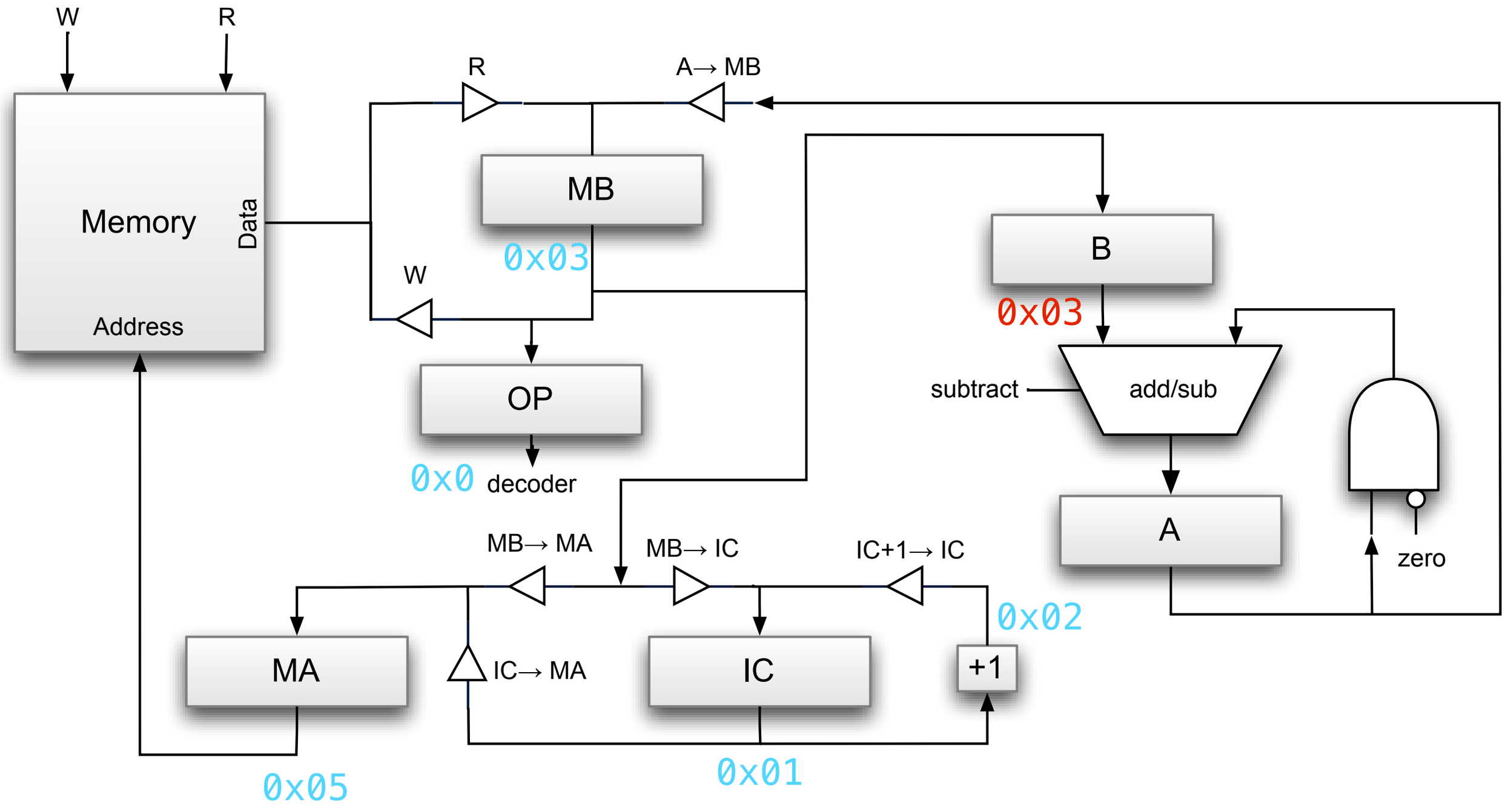
`ld A, C3` load C3 into MA and increment IC

∑niac datapath



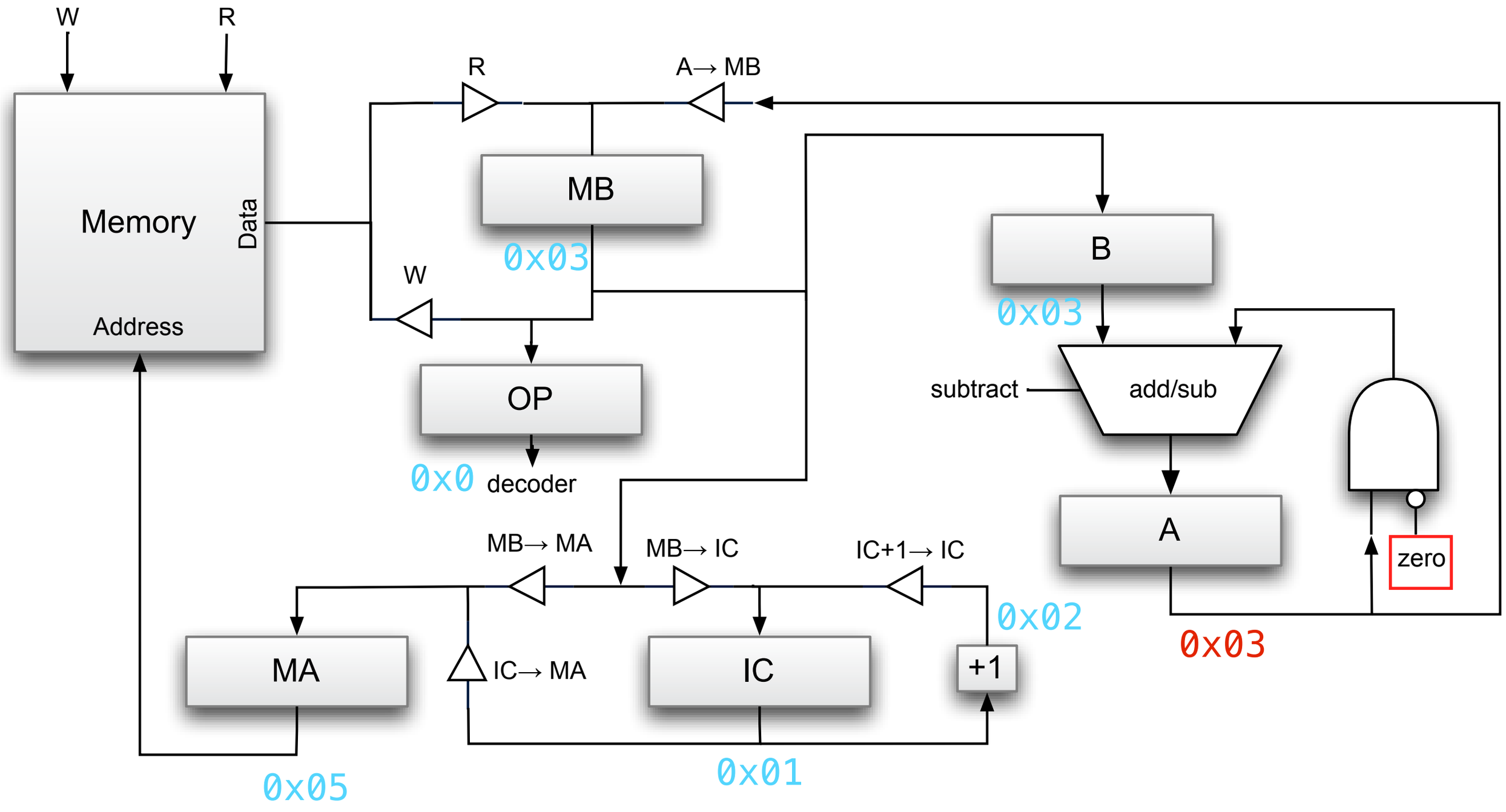
`ld A, C3` fetch Mem[C3]

∑niac datapath



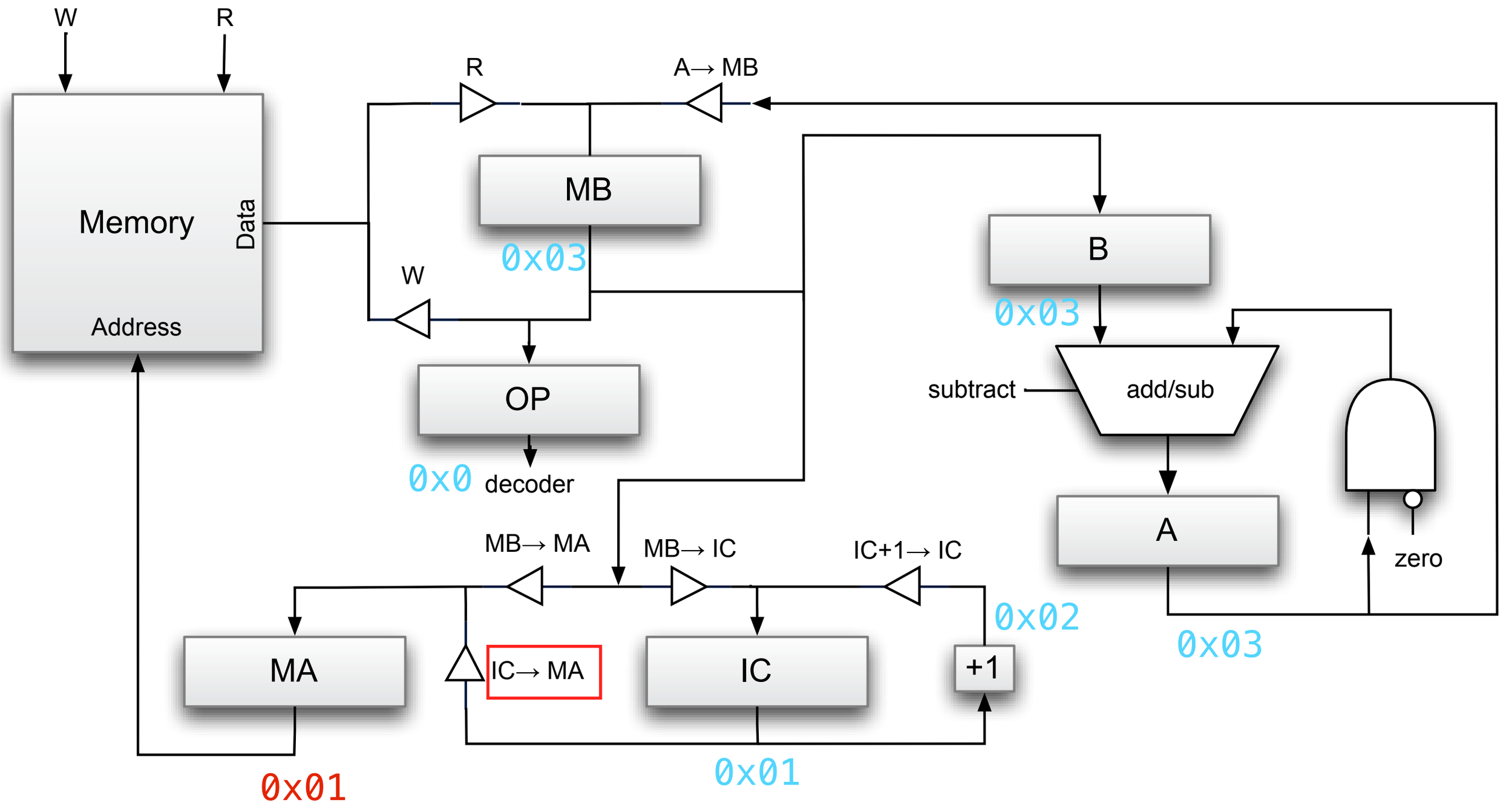
`ld A, C3` load Mem[C3] into B

∑niac datapath



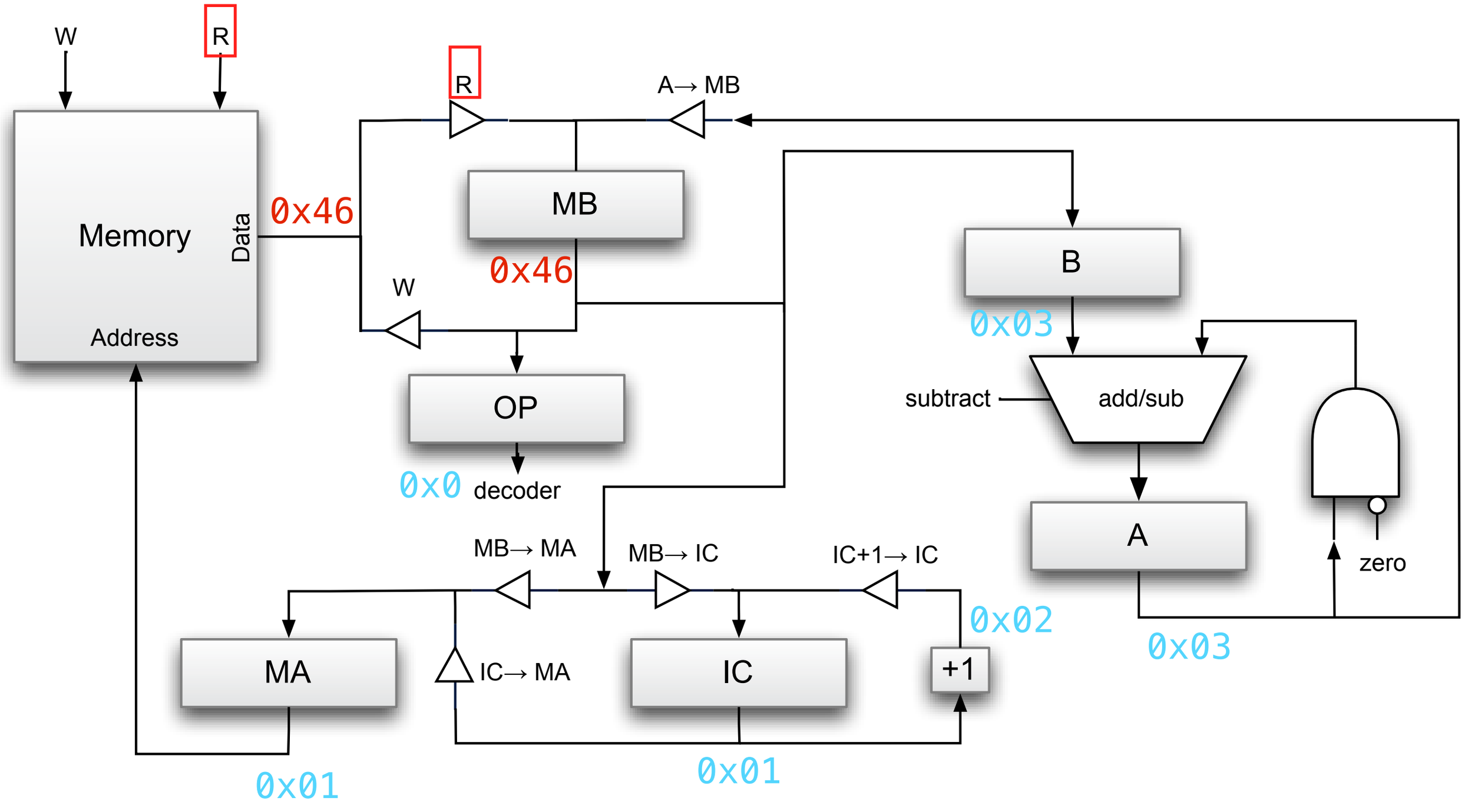
`ld A, C3` load Mem[C3] into A

∑niac datapath



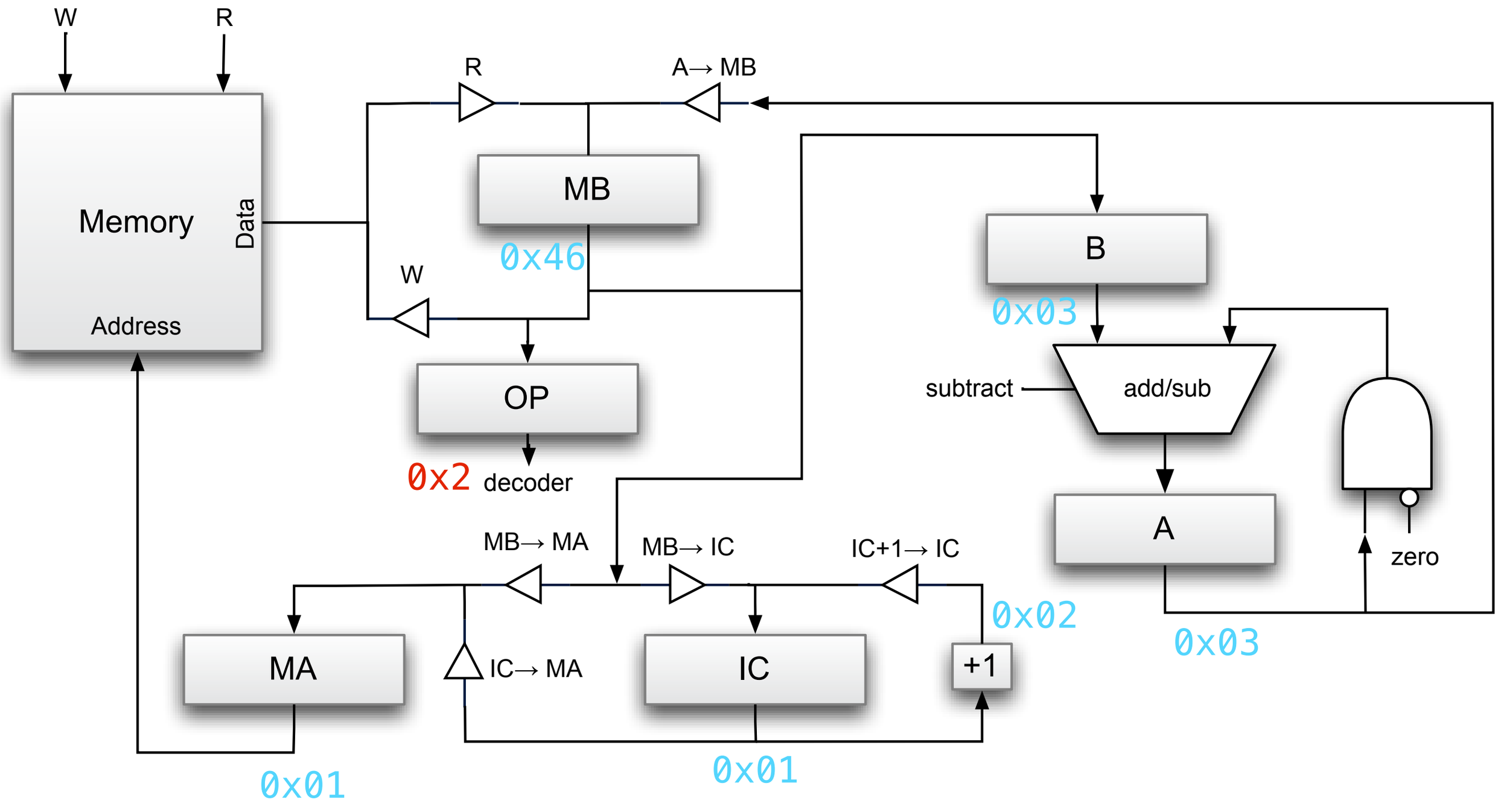
`add A, C10` load addr. of next instruction into MA

∑niac datapath



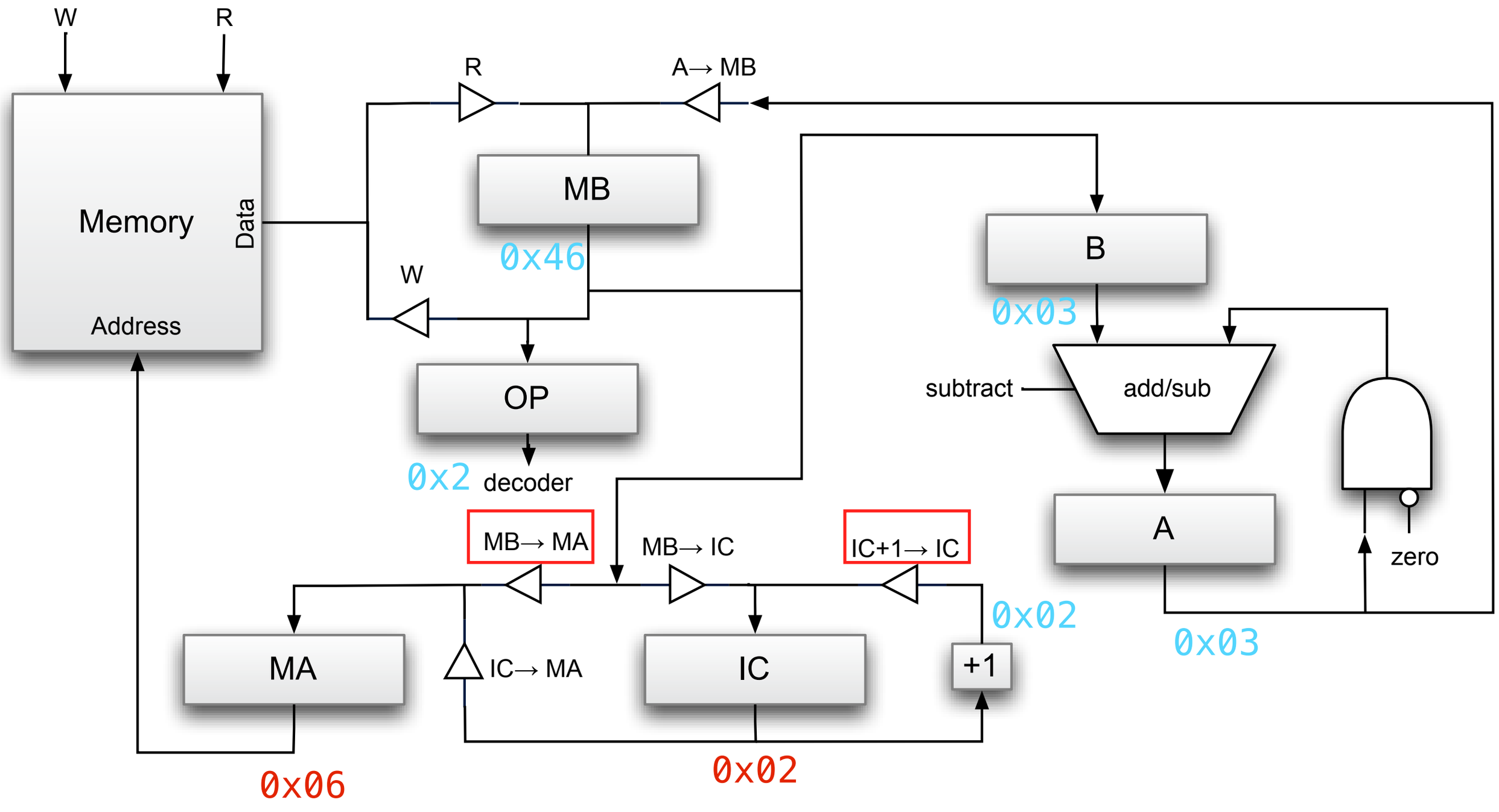
`add A, C10` fetch instruction

∑niac datapath



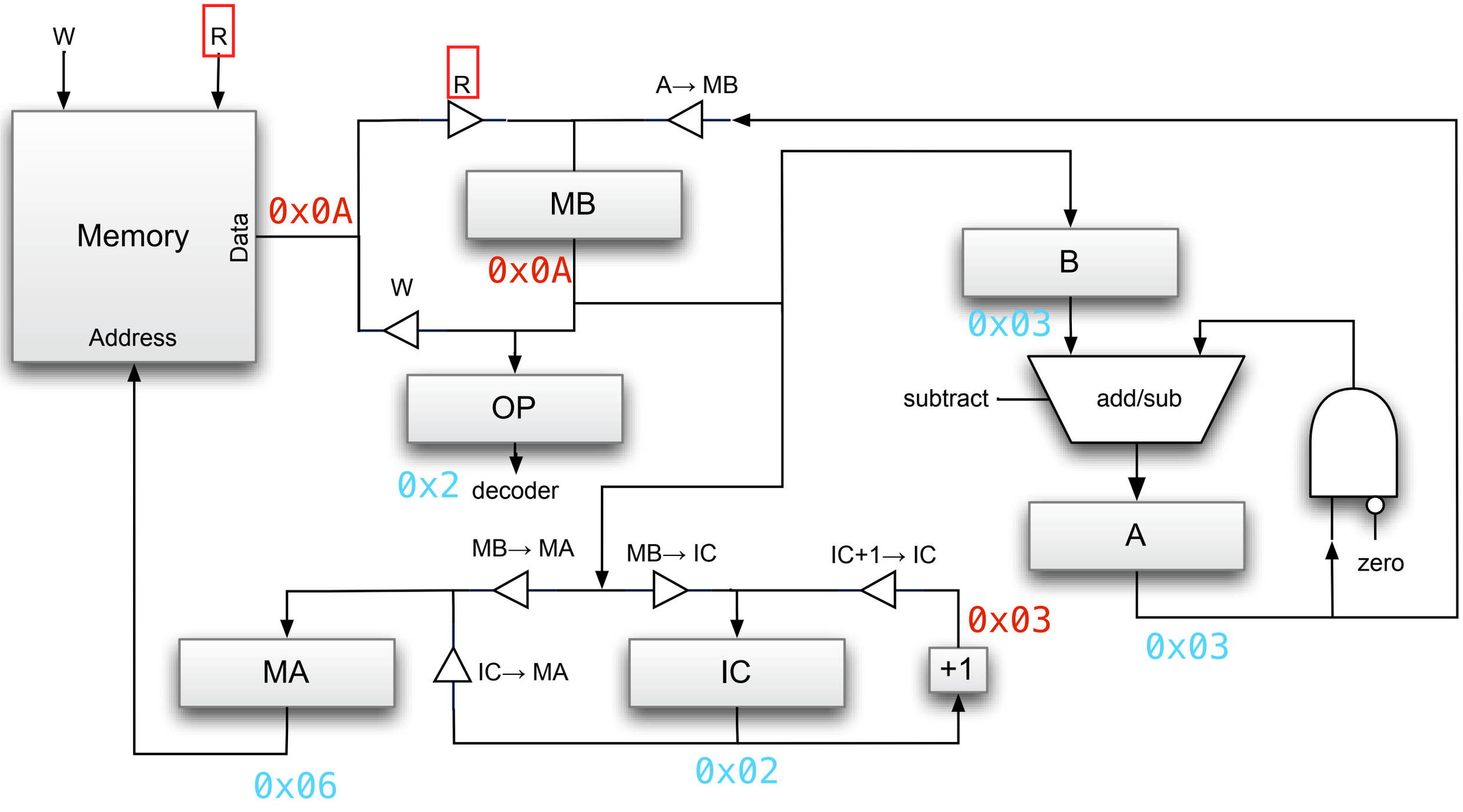
`add A, C10` decode instruction

∑niac datapath



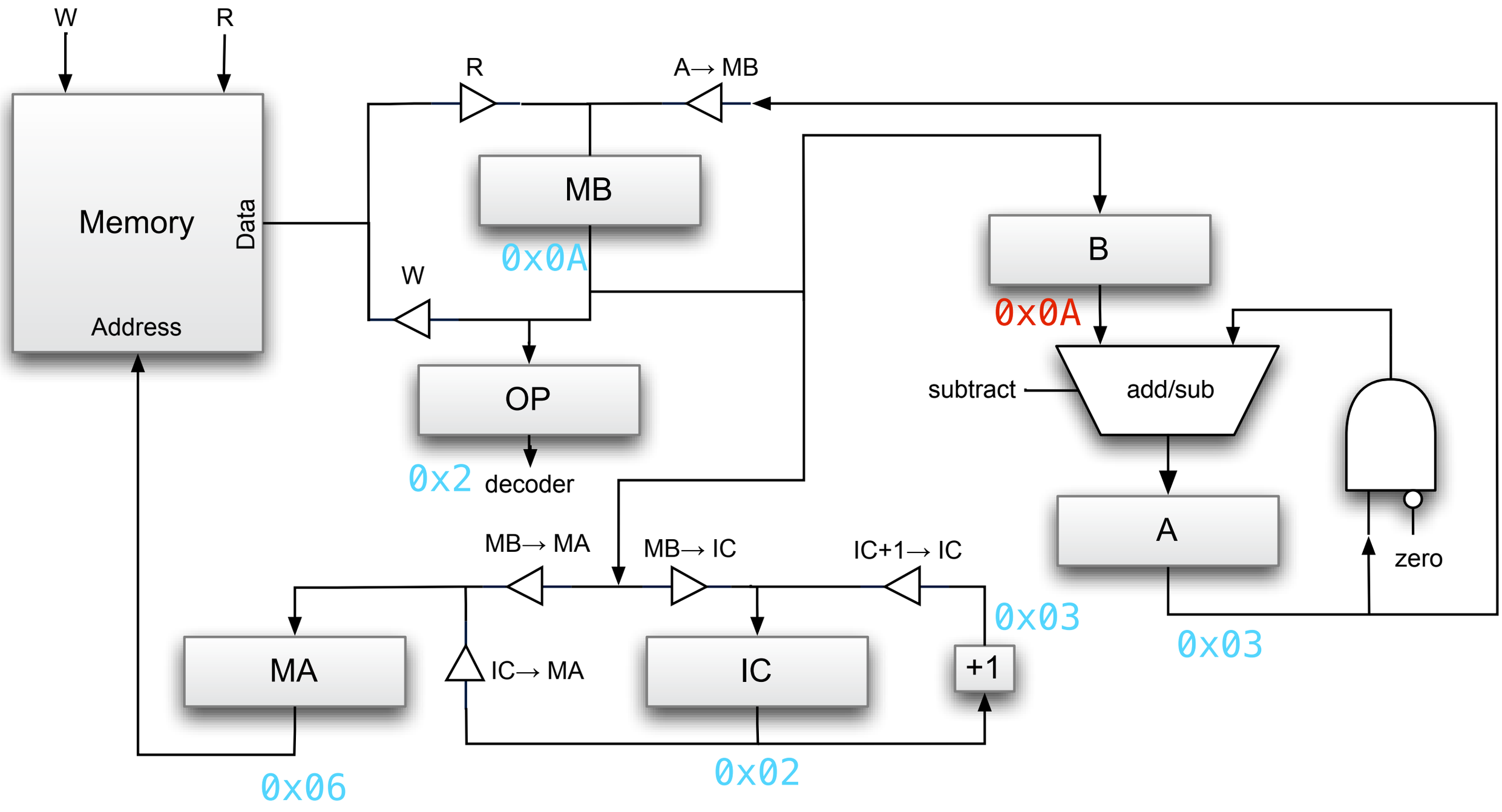
`add A, C10` load C10 into MA and increment IC

∑niac datapath



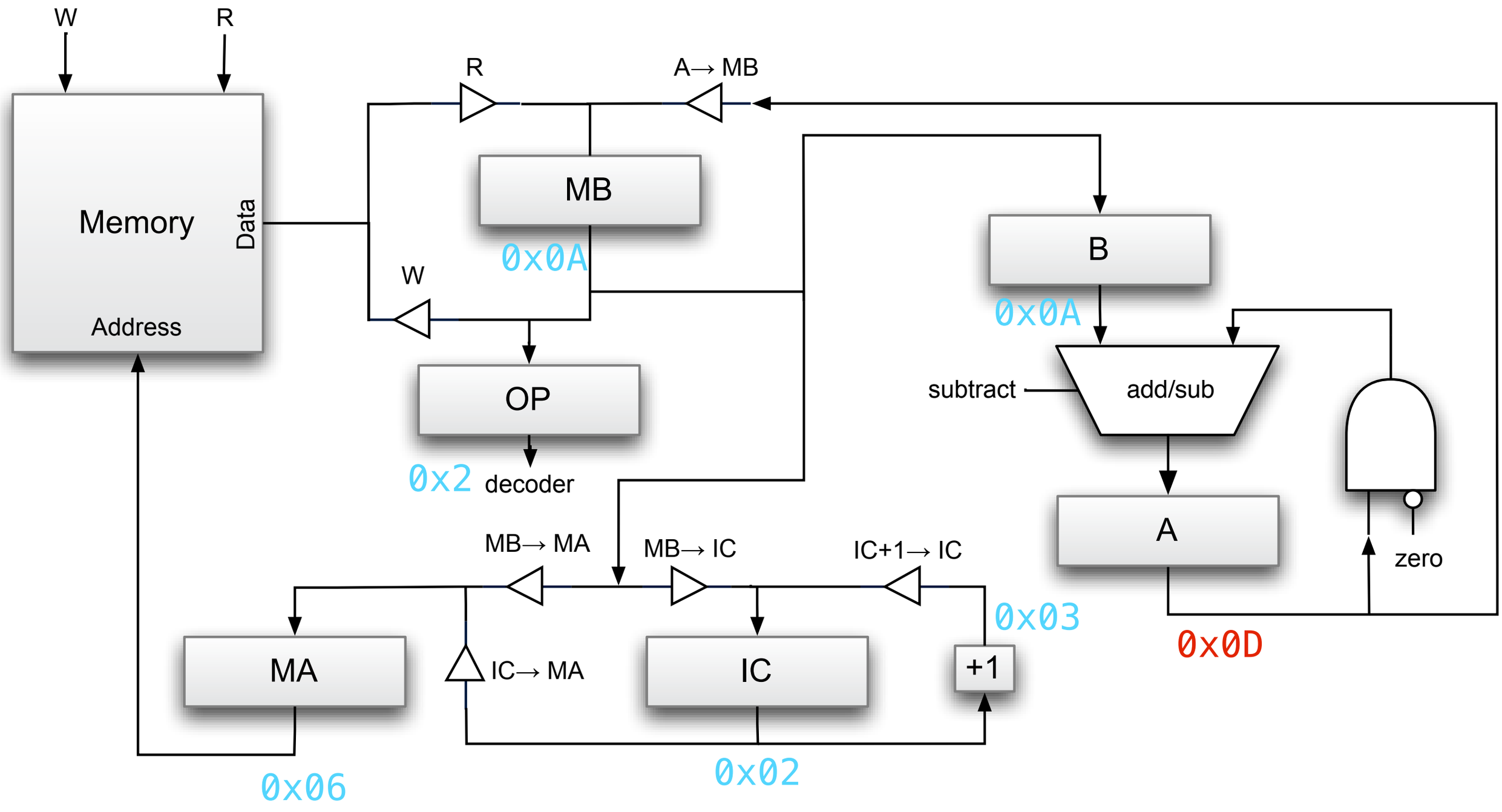
`add A, C10` fetch Mem[C10]

∑niac datapath



`add A, C10` load Mem[C10] into B

∑niac datapath




```
add A, C10  A = A + B
```

	000 ld	001 sub	010 add	011 st	100 b	101 bneg	111 halt
t0	Mem ⟹ MB: R, ld_mb						
t1	MB ⟹ OP: ld_op						
t2	Inc IC: IC+1>IC, ld_ic				MB>IC, ld_ic	A7=0: IC + 1> IC A7=1: MB>IC ld_ic	halt
t3	MB>MA, ld_ma, set E				IC>MA, ld_ma		

E

t0	Mem ⟹ MB: R, ld_mb			A>MB, ld_mb
t1	MB ⟹ B: ld_b			MB ⟹ Mem: W
t2	zero, ld_a	sub, ld_a	ld_a	-
t3	IC>MA, ld_ma, reset E			