# ΗΥ 232
## Οργάνωση και Σχεδίαση Υπολογιστών

## Διάλεξη 10
# Διοχέτευση (Pipeline)

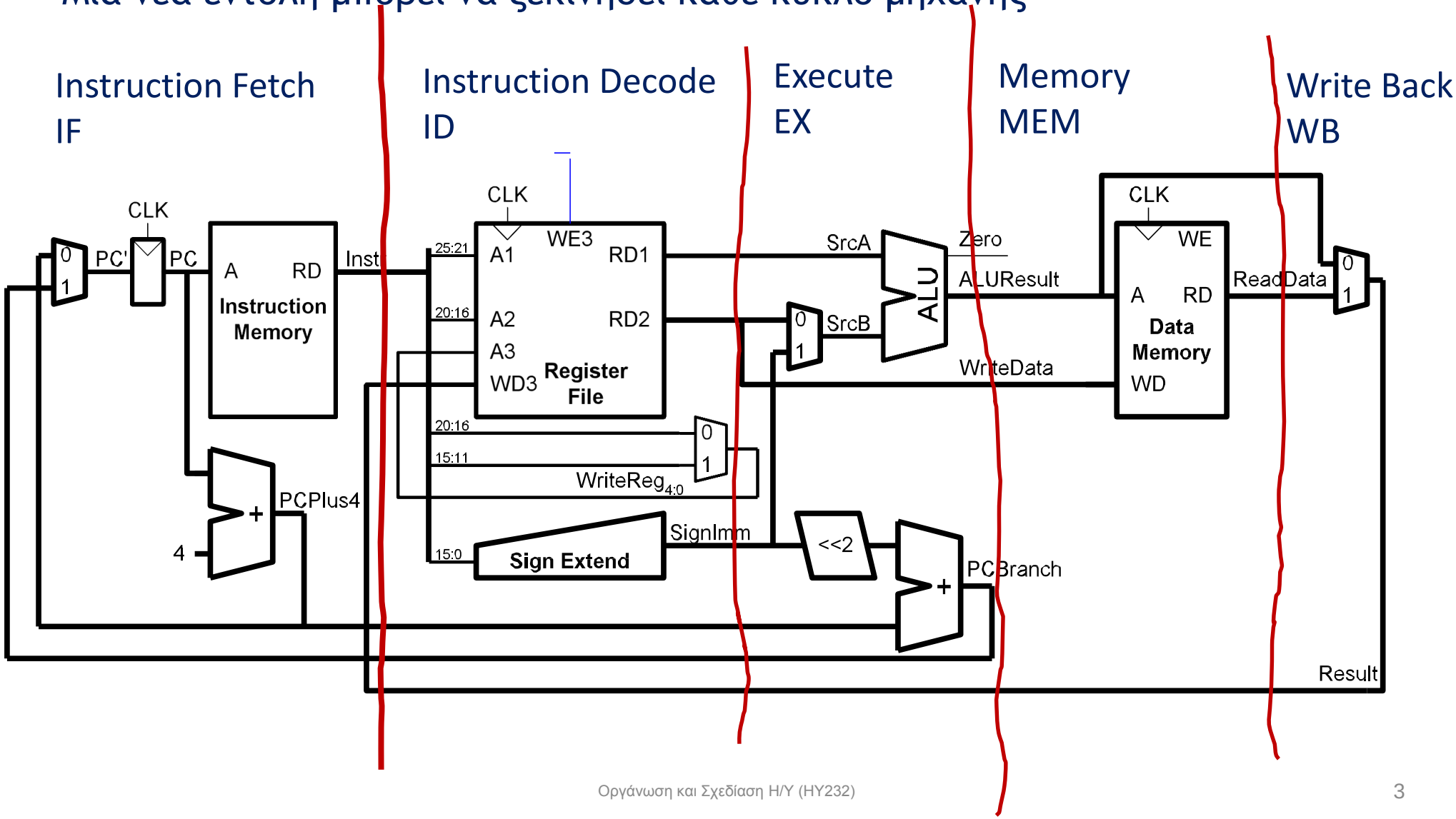
Νίκος Μπέλλας
Τμήμα Μηχανικών Η/Υ, Τηλεπικοινωνιών και Δικτύων

# Θέματα Απόδοσης

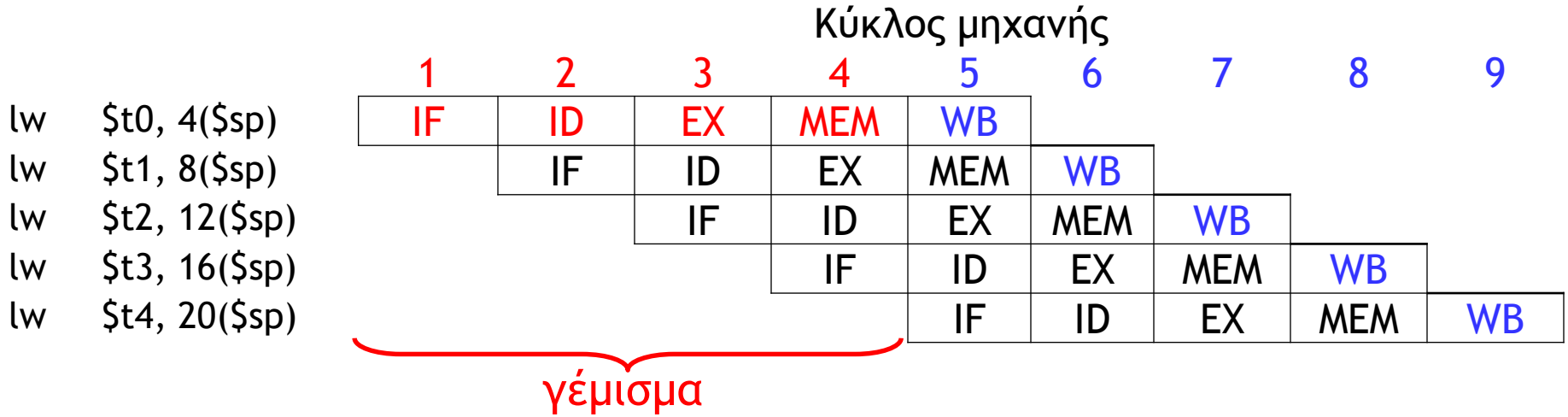
- Αν και απλή, η υλοποίηση ενός κύκλου ρολογιού είναι πολύ αργή
  - Μεγάλη περίοδος ρολογιού
  - Ο χρόνος εκτέλεσης είναι ο ίδιος για όλες τις εντολές και ίσος με τον χρόνο εκτέλεσης της πιο αργής εντολής
- Αυτό παραβιάζει μια βασική σχεδιαστική αρχή
  - Βελτιστοποίησε την εκτέλεση της πιο συχνής εντολής
- Η χρήση της διοχέτευσης (pipeline) μειώνει δραστικά την περίοδο ρολογιού

# Pipeline 5 κύκλων

Έστω ότι στον επεξεργαστή ενός κύκλου η περίοδος ρολογιού είναι Tc1 = 800ps
Στον νέο επεξεργαστή με pipeline μπορεί να είναι στην καλύτερη περίπτωση 800/5=160ps.
Ας θεωρήσουμε μια πιο ρεαλιστική προσέγγιση Tc2 = 200ps
Μια νέα εντολή μπορεί να ξεκινήσει κάθε κύκλο μηχανής



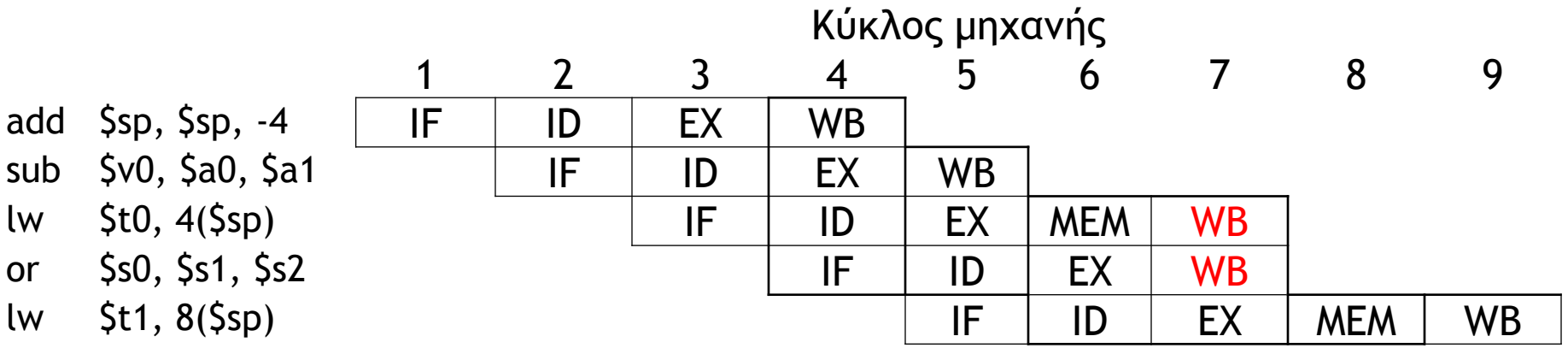
# Γιατί Διοχέτευση;




- Χρόνος εκτέλεσης σε ένα ιδανικό pipeline (CPI=1):
  - Χρόνος να γεμίσει το pipeline με εντολές + ένας κύκλος ανά εντολή
  - Για N εντολές χρειαζόμαστε 4+N κύκλους μηχανής.
  - Χρόνος pipeline CPU = (4+N)*200ps = 200.8ns για N=1000
- Χρόνος εκτέλεσης σε έναν επεξεργαστή ενός κύκλου
  - Χρόνος single-cycle CPU = N*800ps = 800ns
- Επιτάχυνση = 800/200.8 = 4

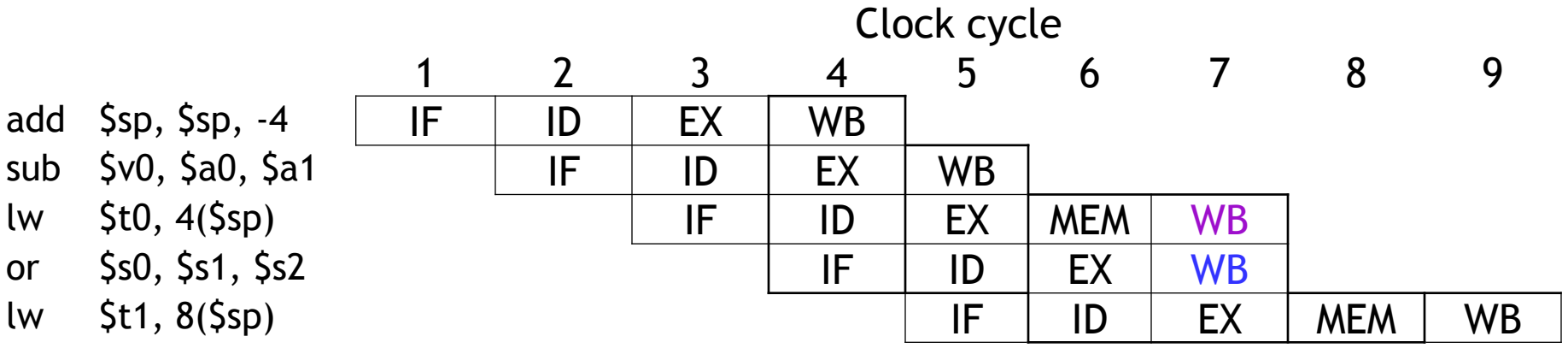
# Pipelining για τις υπόλοιπες εντολές

- Η εντολή `lw` χρησιμοποιεί και τα 5 στάδια του pipeline
- Εντολές με R-format δεν χρειάζονται το στάδιο μνήμης MEM, μόνο τα 4 στάδια IF, ID, EX, και WB
- Τι θα συμβεί εάν βάλουμε στο pipeline την `lw` με εντολές R-format;



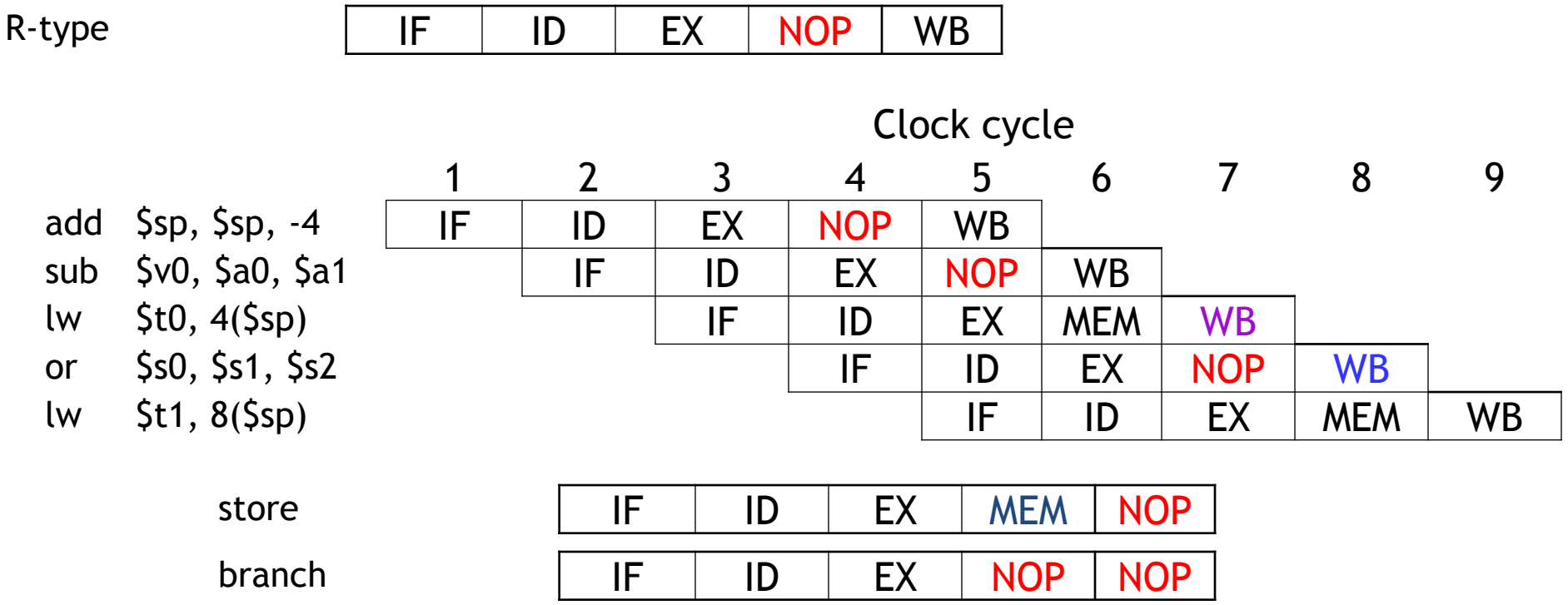
# Δομικοί κίνδυνοι (structural hazards)

- Κάθε δομικό στοιχείο του επεξεργαστή μπορεί να χρησιμοποιηθεί μόνο από μία εντολή κάθε κύκλο μηχανής
- Πρόβλημα:
  - Η εντολή `lw` χρησιμοποιεί το Write Port του Register File στο 5ο στάδιο της
  - Η εντολή `or` χρησιμοποιεί το Write Port του Register File στο 4ο στάδιο της



# Λύση : χρήση εντολών NOP

- Κανόνες:
  - Όλες οι εντολές παίρνουν 5 κύκλους να εκτελεσθούν
  - Όλες οι εντολές περνάνε από τα ίδια 5 στάδια με την ίδια σειρά.
    - Ακόμη και εάν κάποια στάδια δεν κάνουν καμία επεξεργασία στην εντολή (NOP)



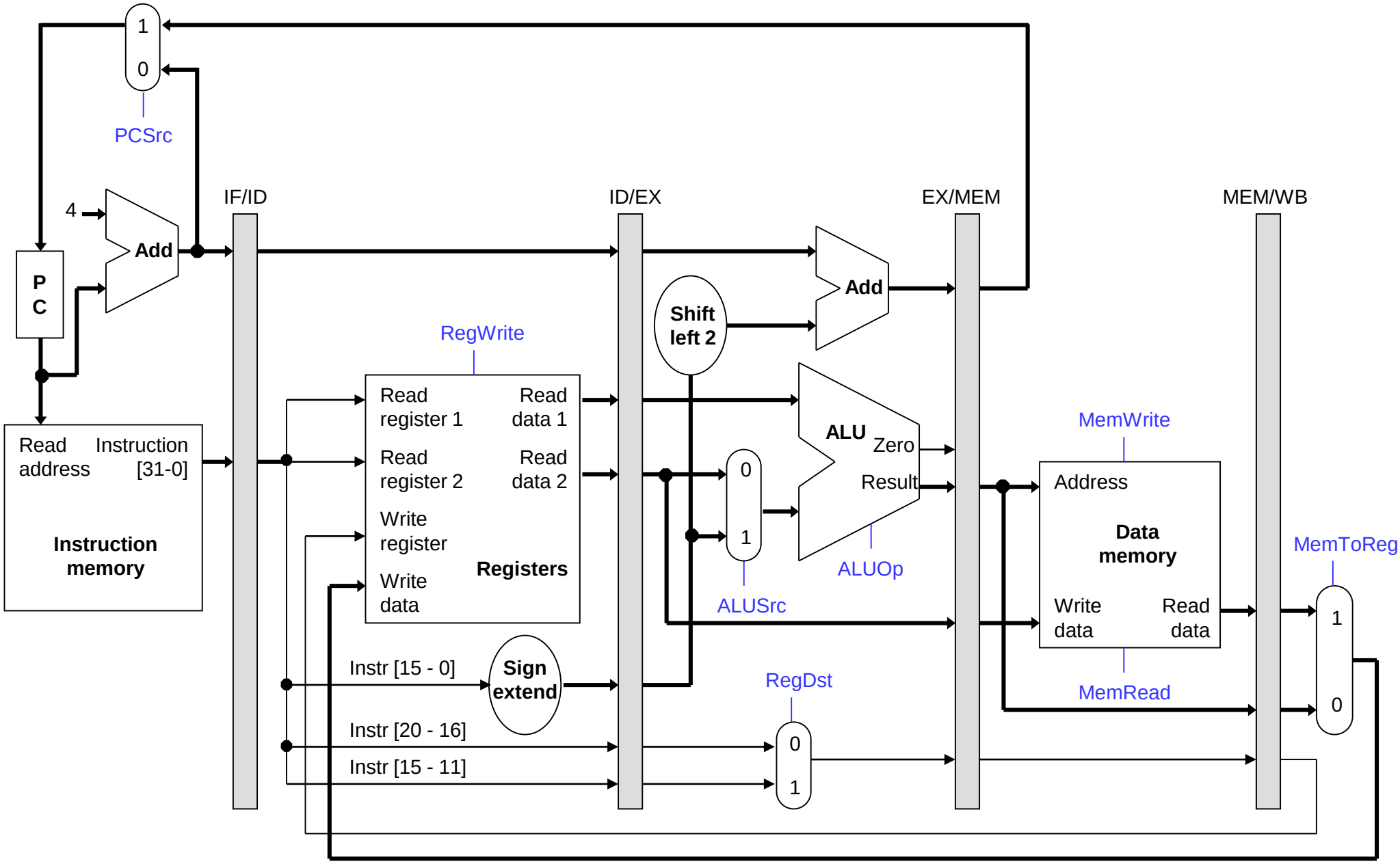
# Καταχωρητές Διοχέτευσης

- Για να υλοποιήσουμε ένα σύστημα διοχέτευσης, θα χρησιμοποιήσουμε ενδιάμεσους καταχωρητές στο datapath.
- Όλα τα ενδιάμεσα αποτελέσματα και σήματα ελέγχου πρέπει να σώζονται στο τέλος κάθε κύκλου στον αντίστοιχο καταχωρητή διοχέτευσης.
- Ονόματα καταχωρητών διοχέτευσης.

IF/ID ID/EX EX/MEM MEM/WB

- Το τελευταίο στάδιο (Write Back, WB) δεν χρειάζεται καταχωρητές διοχέτευσης γιατί η εντολή τερματίζεται.
- Είναι οι καταχωρητές διοχέτευσης ορατοί στον προγραμματιστή;

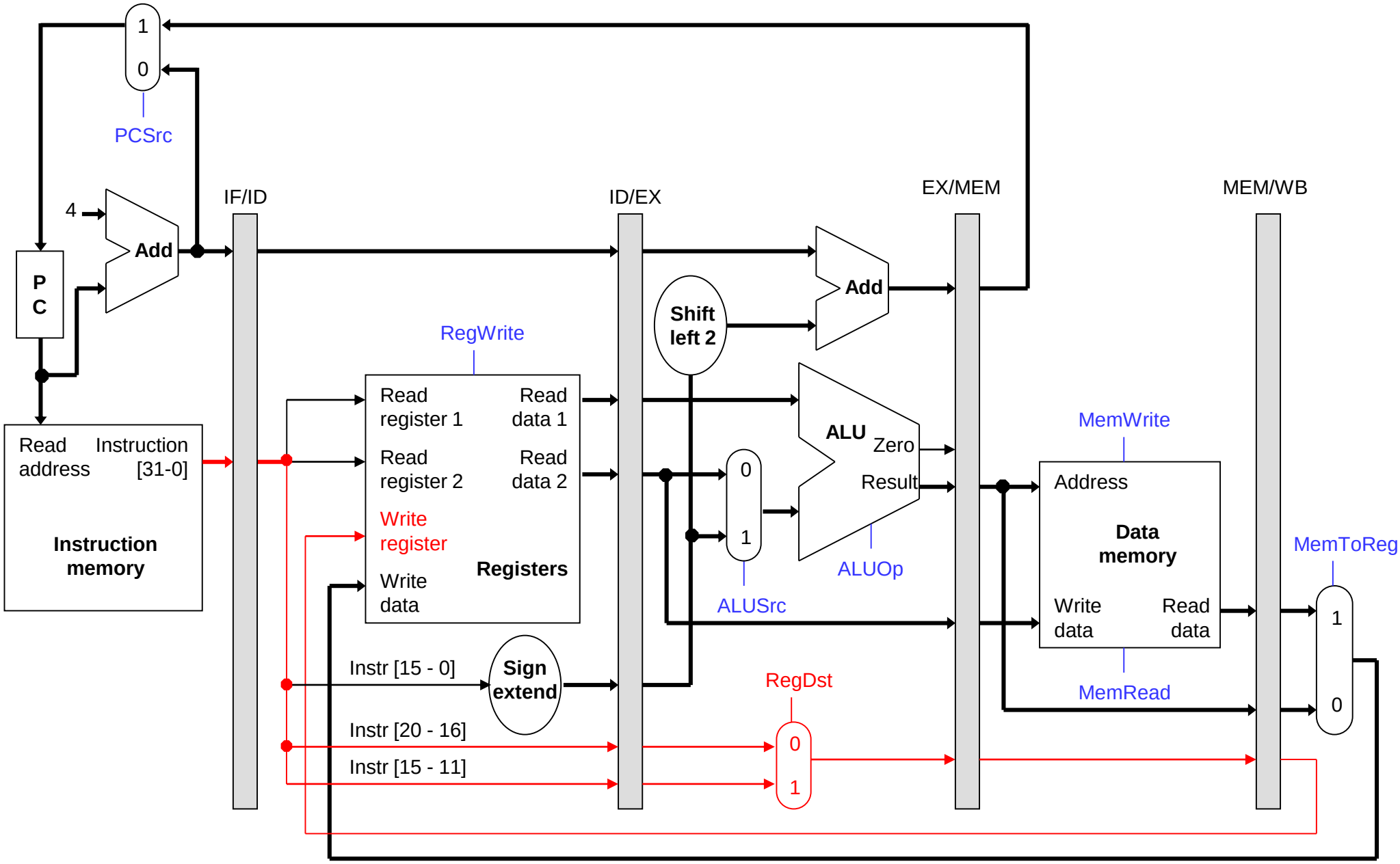
# Διοχέτευση Δεδομένων



# Προώθηση δεδομένων

- Οι τιμές των δεδομένων που χρειάζονται σε επόμενα στάδια της διοχέτευσης πρέπει να μεταφερθούν μέσω των καταχωρητών διοχέτευσης
- Ο καταχωρητής προορισμού `rd` είναι ένα χαρακτηριστικό παράδειγμα.
  - Το πεδίο `rd` της εντολής ανακτάται στο πρώτο στάδιο (Instruction Fetch, IF). Αλλά χρησιμοποιείται μόνο μετά από 5 κύκλους στο στάδιο Write Back, WB.
  - Συνεπώς, το πεδίο `rd` πρέπει να μεταφερθεί από όλους τους καταχωρητές διοχέτευσης μέχρι το 5ο στάδιο.
- Δεν μπορούμε να το κρατήσουμε σε ένα απλό καταχωρητή εντολής (instruction register) γιατί σε κάθε κύκλο μηχανής μια νέα εντολή εισέρχεται στον επεξεργαστή από την μνήμη εντολών.

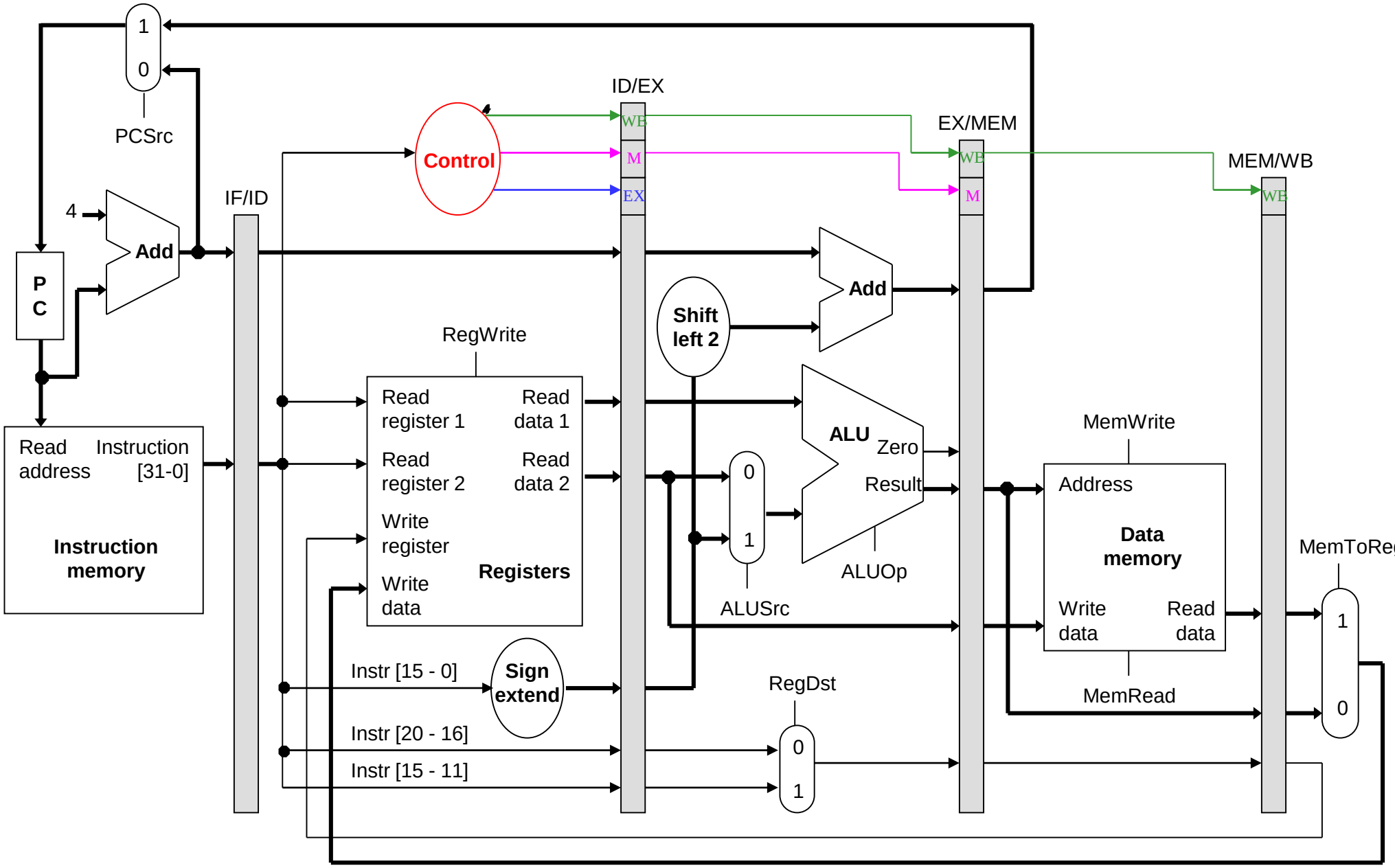
# Καταχωρητής Προορισμού



# Σήματα Ελέγχου

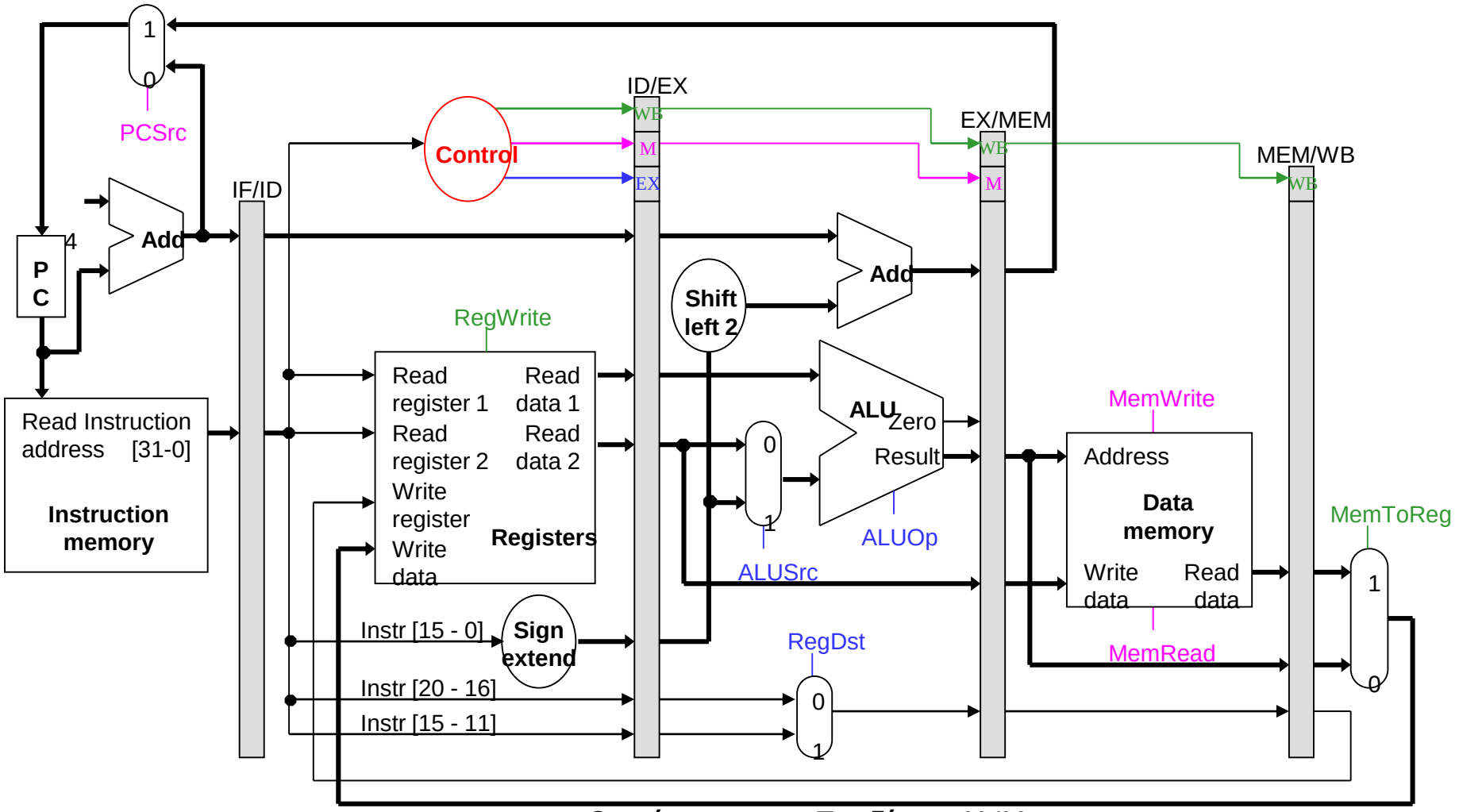
- Η δημιουργία των σημάτων ελέγχου γίνεται ακριβώς όπως στην περίπτωση του επεξεργαστή ενός κύκλου.
  - Μετά το Instruction Fetch η εντολή αποκωδικοποιείται από την μονάδα ελέγχου και δημιουργούνται τα σήματα ελέγχου.
- Όπως και πριν, κάποια σήματα ελέγχου θα χρειαστούν στα τελευταία στάδια του pipeline (πχ 4ο ή 5ο)
- Όπως ακριβώς και τα δεδομένα, έτσι και αυτά τα σήματα ελέγχου θα πρέπει να προωθηθούν μέσω των καταχωρητών διοχέτευσης στα επόμενα στάδια μέχρι να χρησιμοποιηθούν.

# Διοχέτευση Σημάτων Ελέγχου και Δεδομένων



# Διοχέτευση Σημάτων Ελέγχου και Δεδομένων

| Stage | Control signals needed | | |
|---|---|---|---|
| EX | ALUSrc | ALUOp | RegDst |
| MEM | MemRead | MemWrite | PCSrc |
| WB | RegWrite | MemToReg | |



# Μερικές Παρατηρήσεις

- Συνήθως, όταν θέλουμε να γράψουμε σε έναν καταχωρητή (και μόνον τότε) θέτουμε το αντίστοιχο WriteEnable=1.
  - Ο PC γράφεται σε κάθε κύκλο μηχανής επειδή πάντα πρέπει να δείχνουμε στην επόμενη εντολή και συνεπως πάντα WriteEnable=1.
  - Το ίδιο και για όλους τους καταχωρητές διοχέτευσης: WriteEnable=1

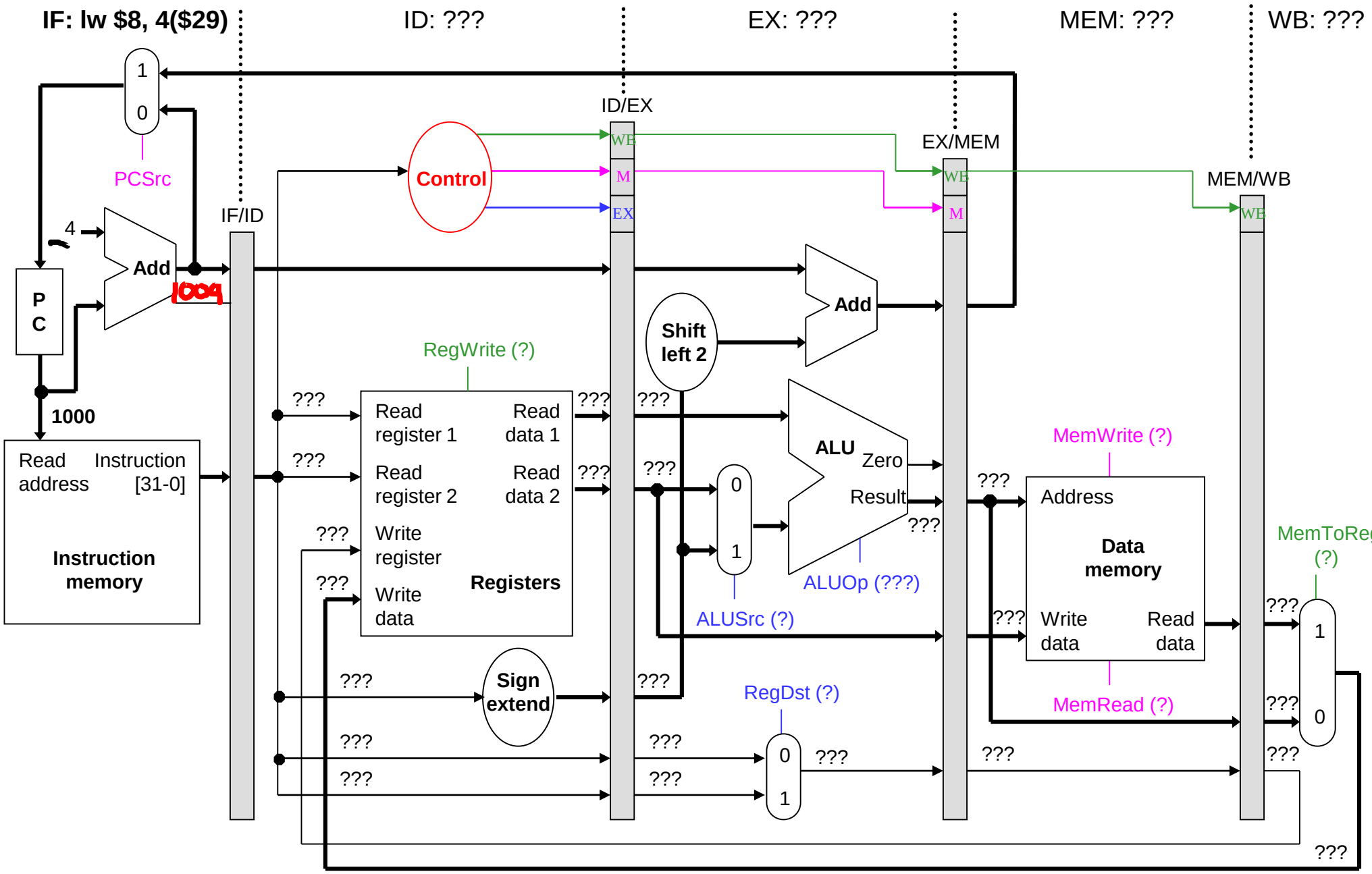
# Παράδειγμα εκτέλεσης εντολών

Διευθύνσεις στο δεκαδικό σύστημα

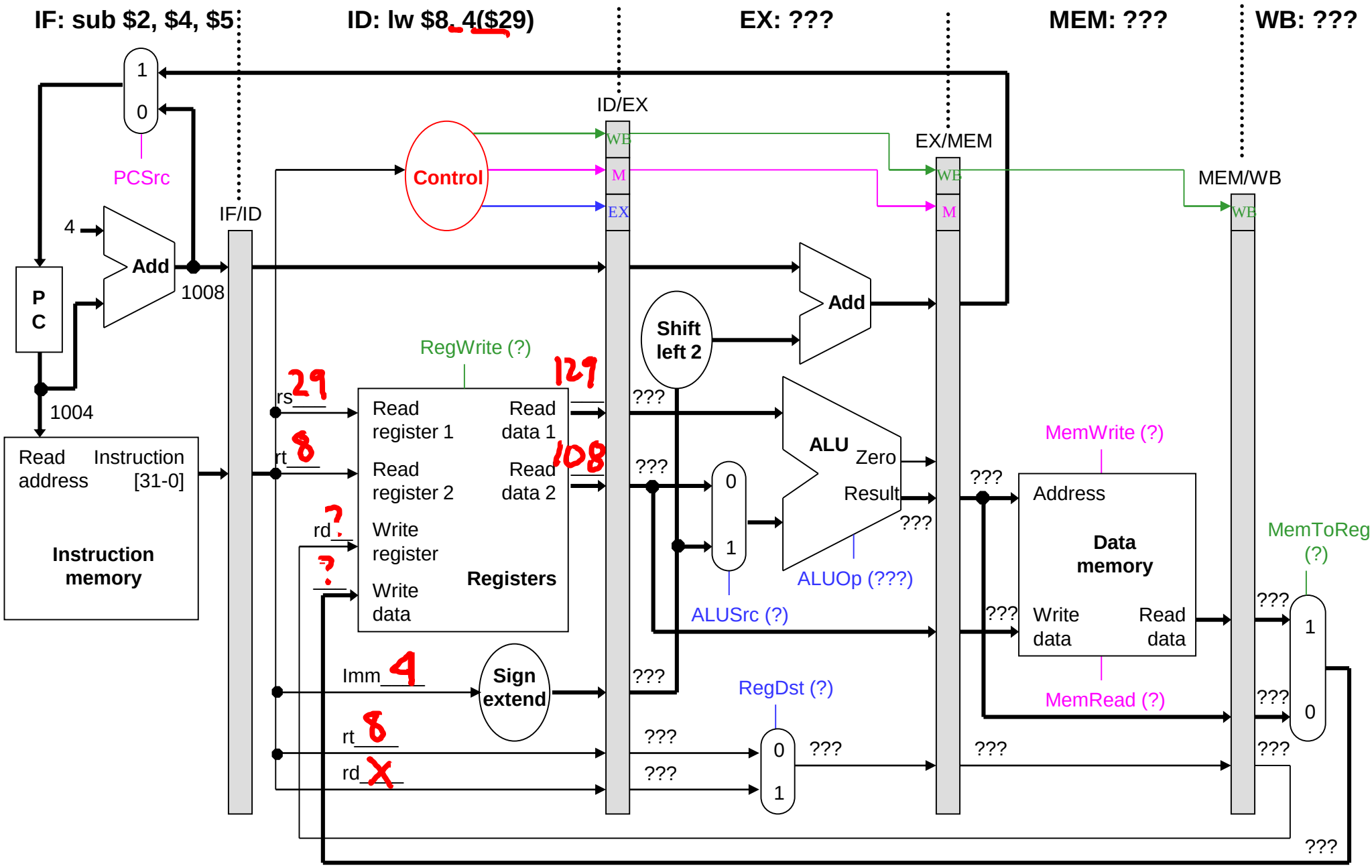
```
1000: lw   $8,  4($29)
1004: sub  $2,  $4, $5
1008: and  $9,  $10, $11
1012: or   $16, $17, $18
1016: add  $13, $14, $0
```

- Υποθέτουμε ότι:
  - Η αρχική τιμή κάθε καταχωρητή `$N` είναι N+100. Για παράδειγμα, ο `$8 ($t0)` περιέχει την τιμή d'108.
  - Αρχικά, όλες οι θέσεις μνήμης περιέχουν την τιμή 99.
  - Το X στο διάγραμμα είναι Don't Care.
  - Τα ερωτηματικά ??? σημαίνει ότι η τιμή του σήματος είναι άγνωστη
  - Παρατηρείστε ότι οι εντολές είναι ανεξάρτητες; δεν υπάρχουν εξαρτήσεις μεταξύ καταχωρητών εξόδου μιας εντολής και εισόδου των επόμενων.
    - Περισσότερα για αυτό στα επόμενα μαθήματα

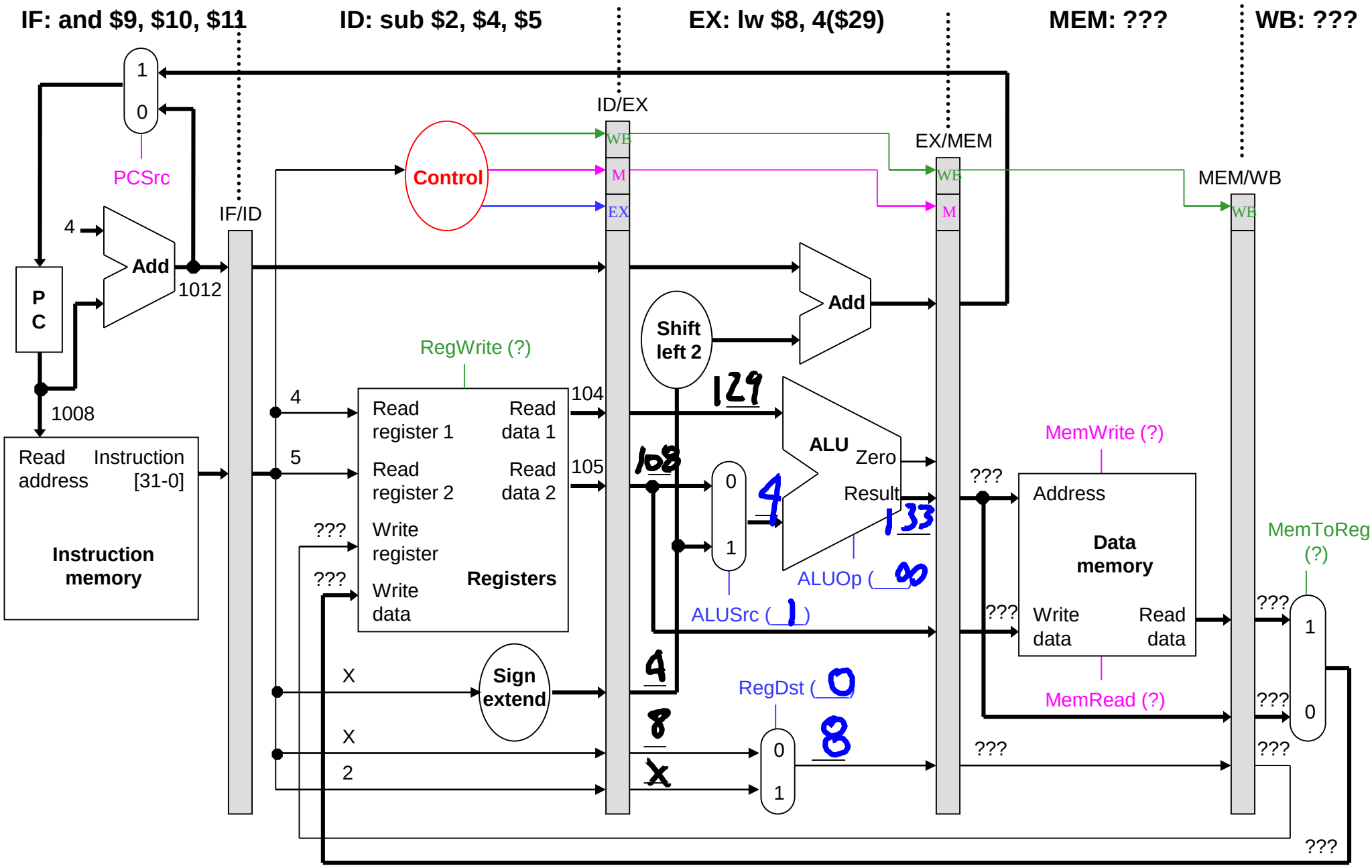
# Κύκλος 1 (γέμισμα)



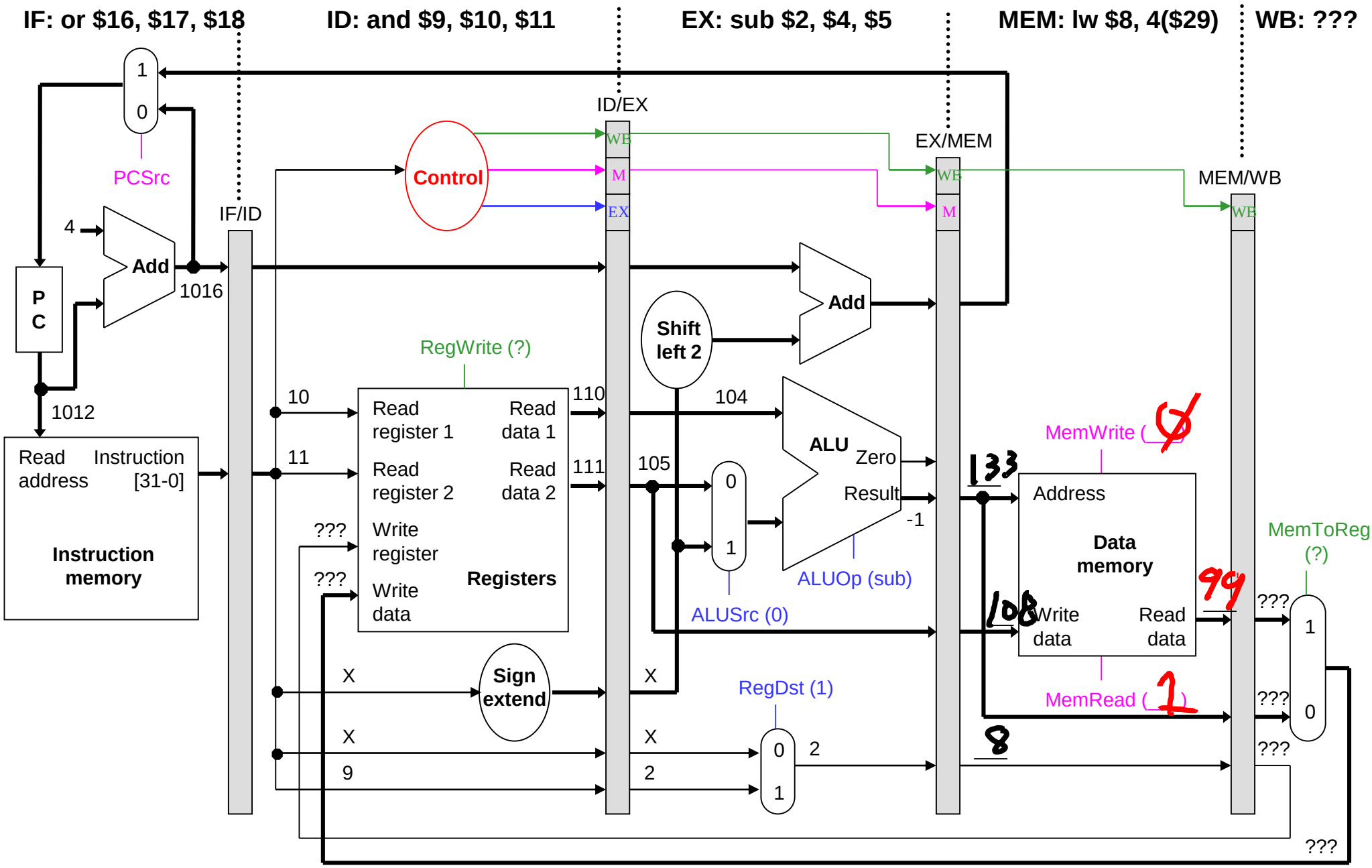
# Κύκλος 2 (γέμισμα)



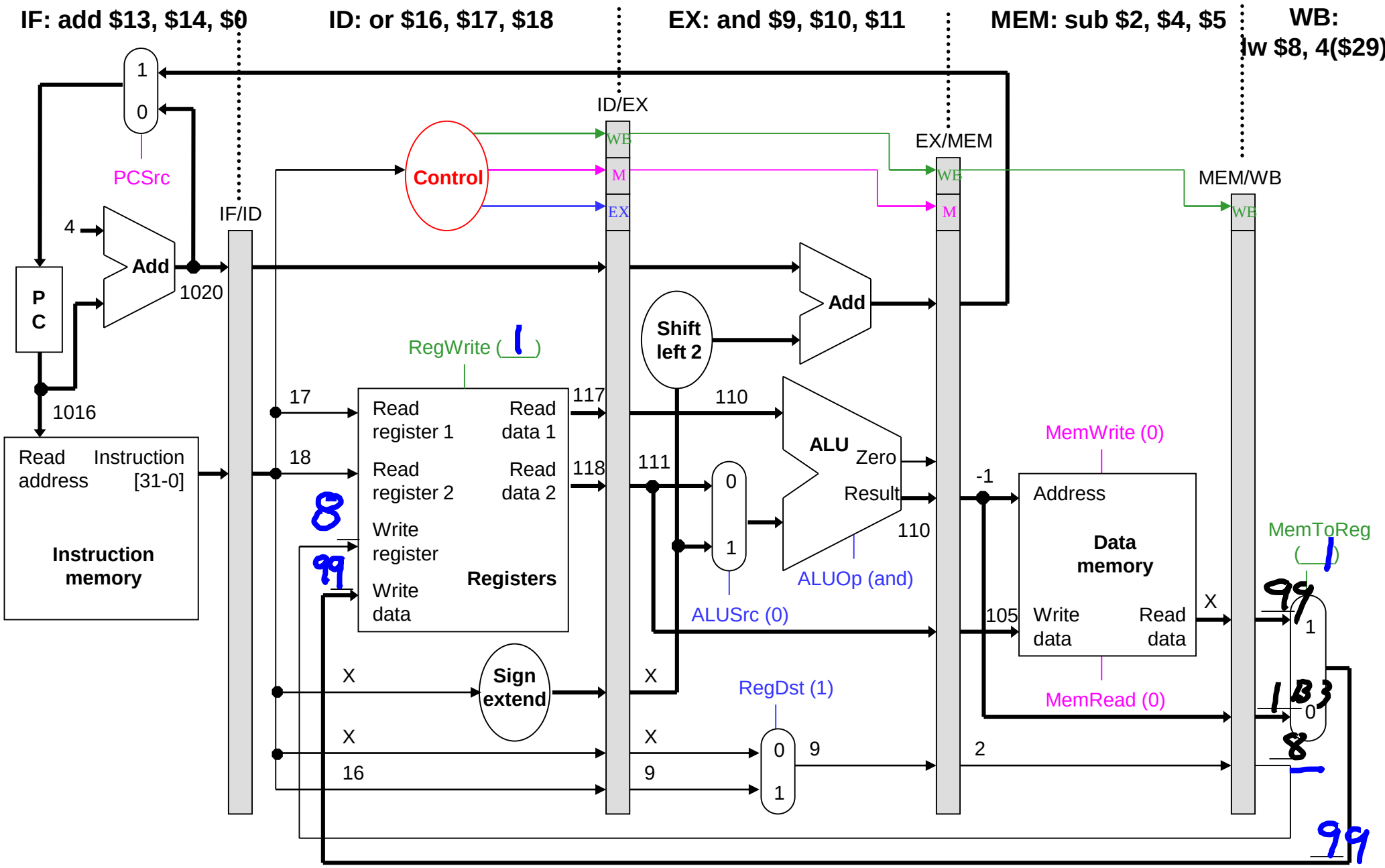
# Κύκλος 3 (γέμισμα)



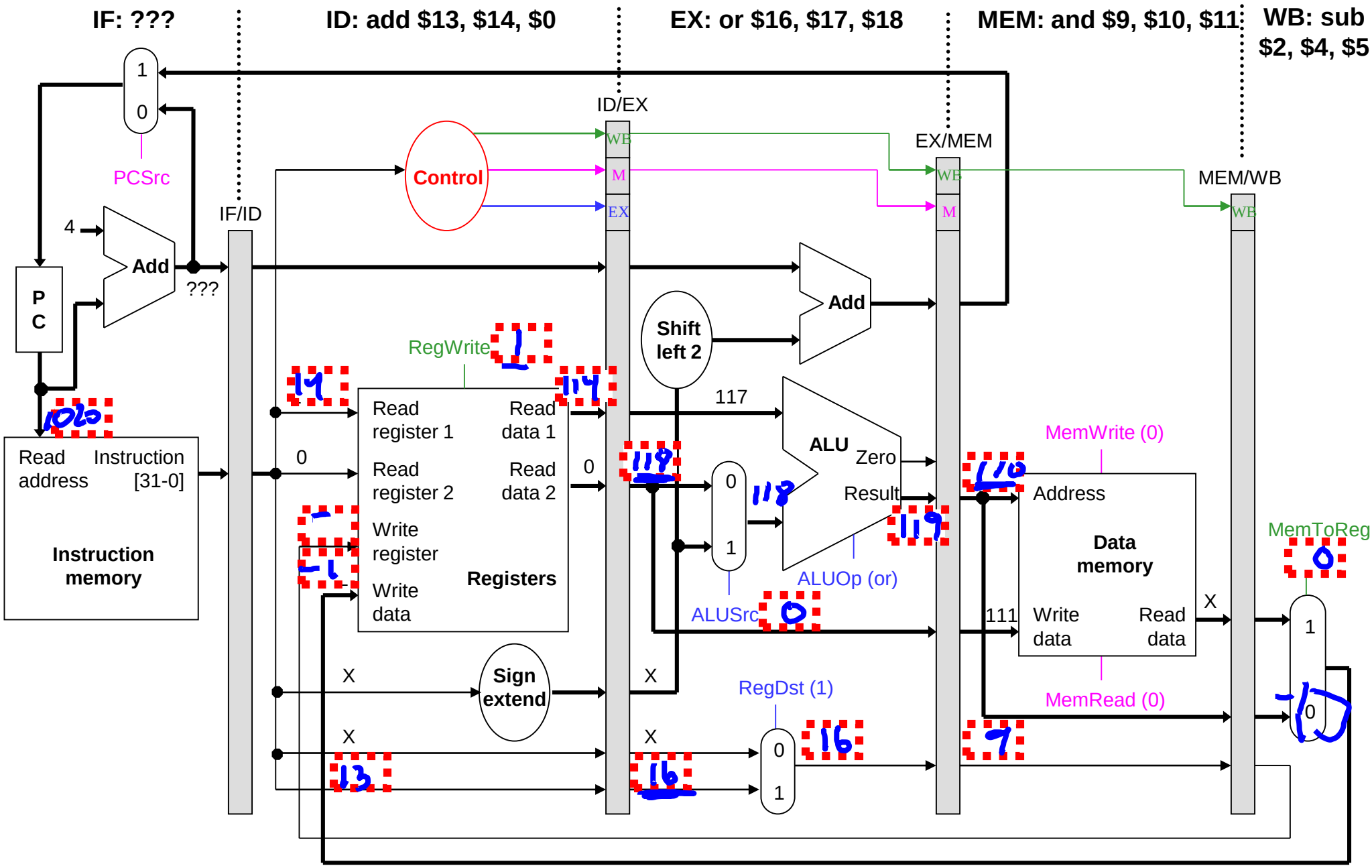
# Κύκλος 4 (γέμισμα)



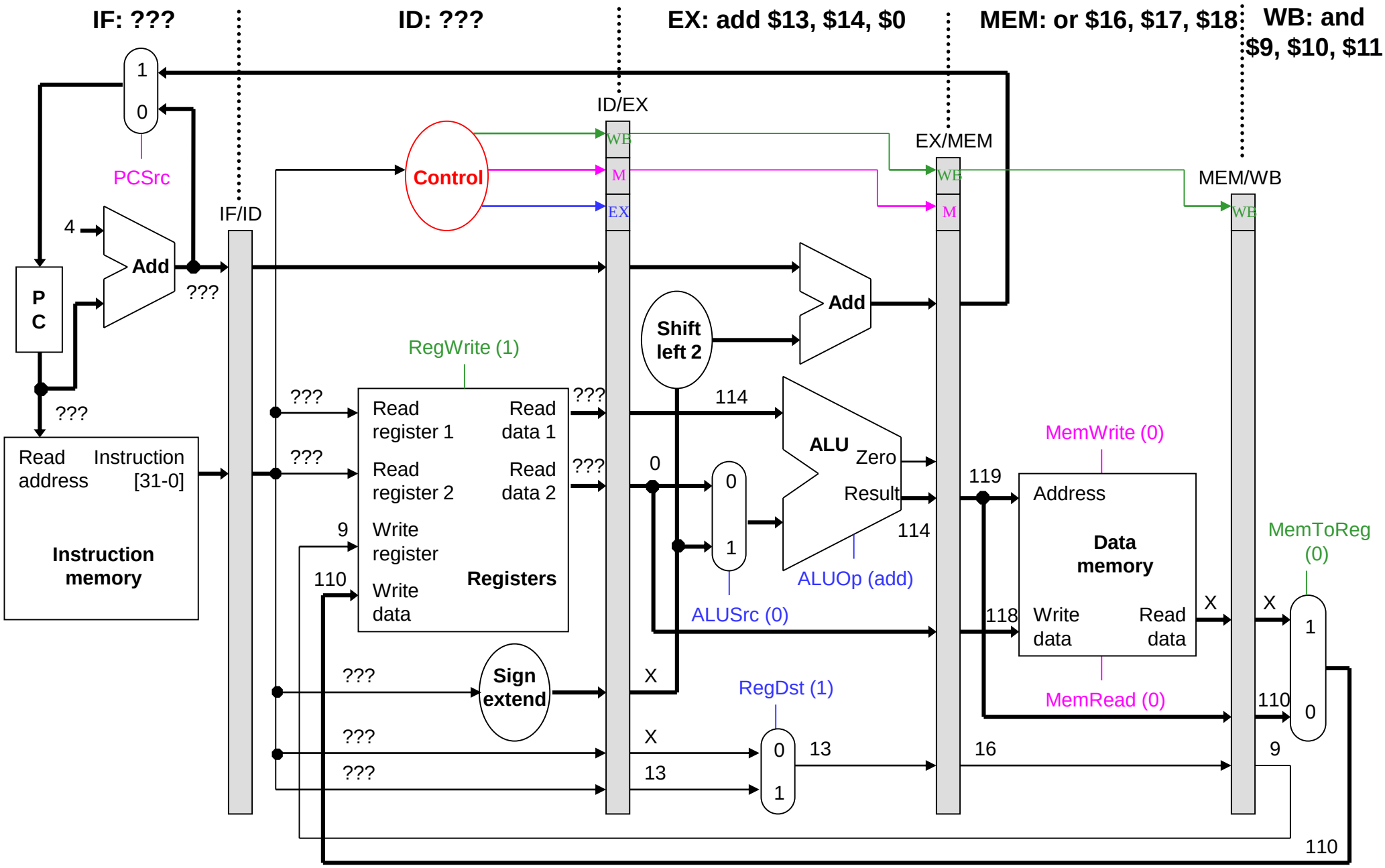
# Κύκλος 5


IF: add $13, $14, $0
ID: or $16, $17, $18
EX: and $9, $10, $11
MEM: sub $2, $4, $5
WB: lw $8, 4($29)
ID/EX
EX/MEM
MEM/WB
IF/ID
PCSrc
Control
WB
M
EX
Add
4
1020
P C
1016
Shift left 2
RegWrite (1)
Read address
Instruction [31-0]
Instruction memory
Read register 1
Read register 2
Write register
Write data
Read data 1
Read data 2
Registers
17
18
117
118
110
111
ALU
Zero
Result
ALUOp (and)
ALUSrc (0)
-1
105
MemWrite (0)
Address
Data memory
Write data
Read data
MemRead (0)
MemToReg (1)
X
Sign extend
16
RegDst (1)
9
2
8
99
183

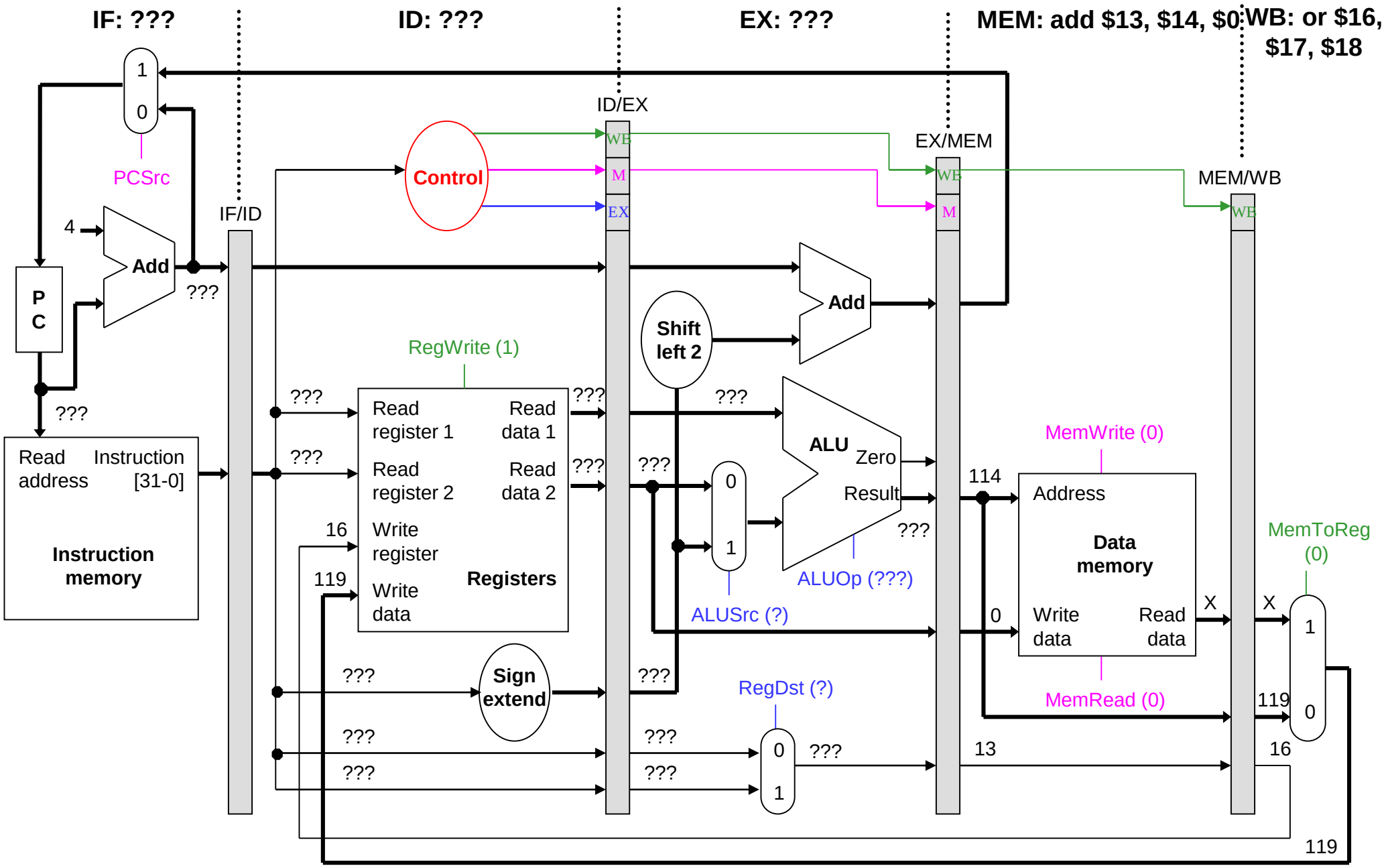
# Κύκλος 6 (άδειασμα)



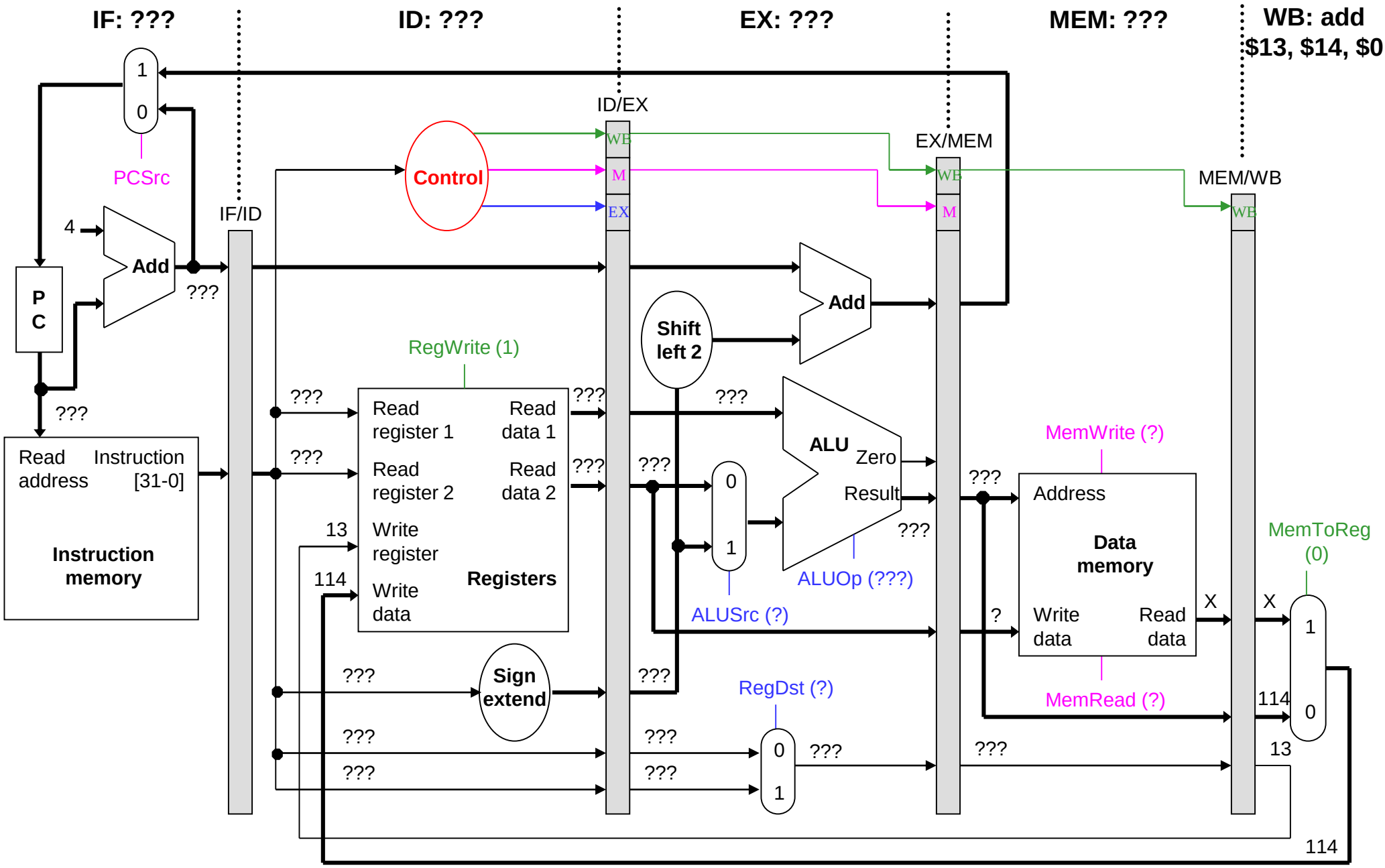
# Κύκλος 7 (άδειασμα)



# Κύκλος 8 (άδειασμα)



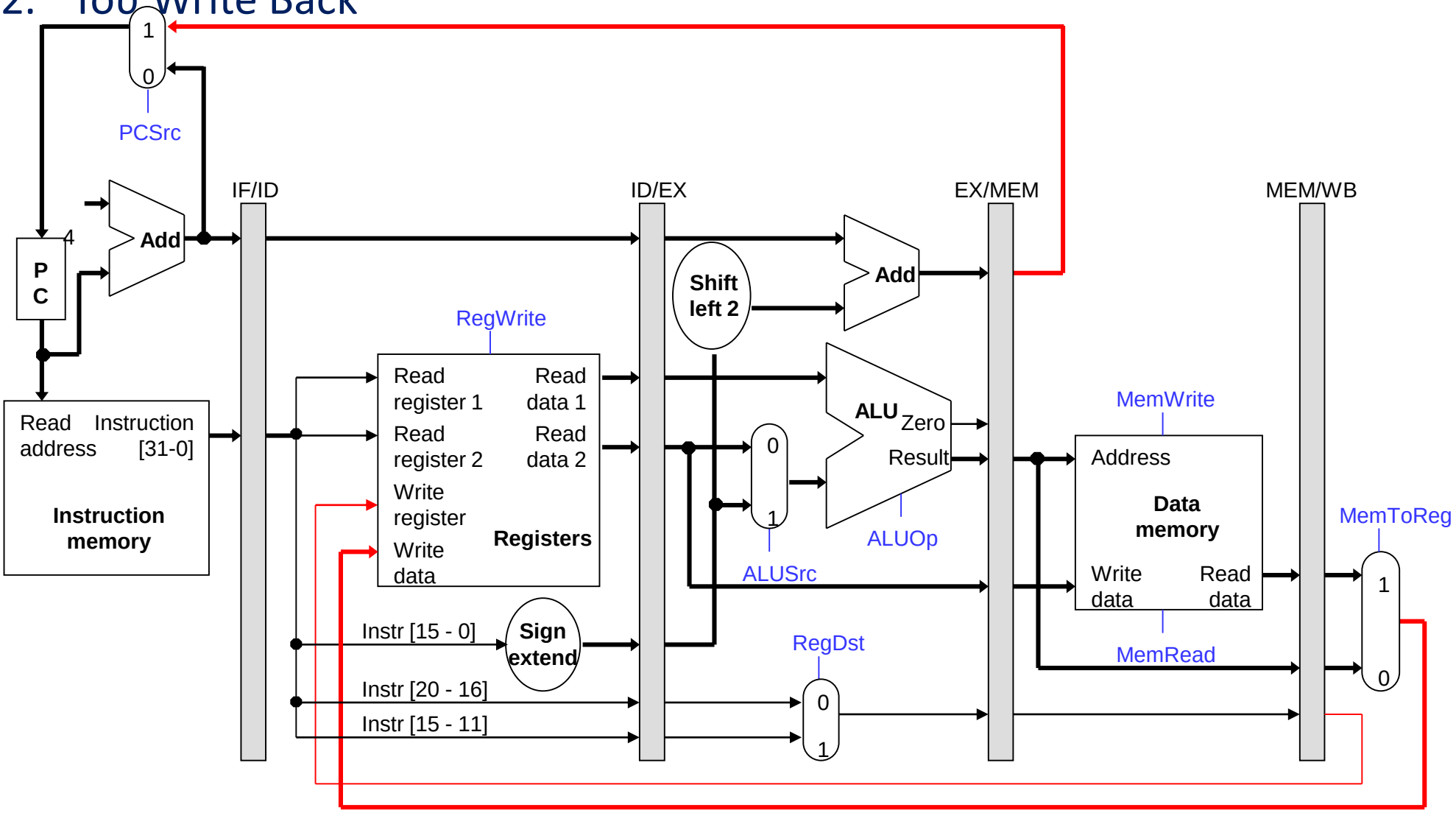
# Κύκλος 9 (άδειασμα)



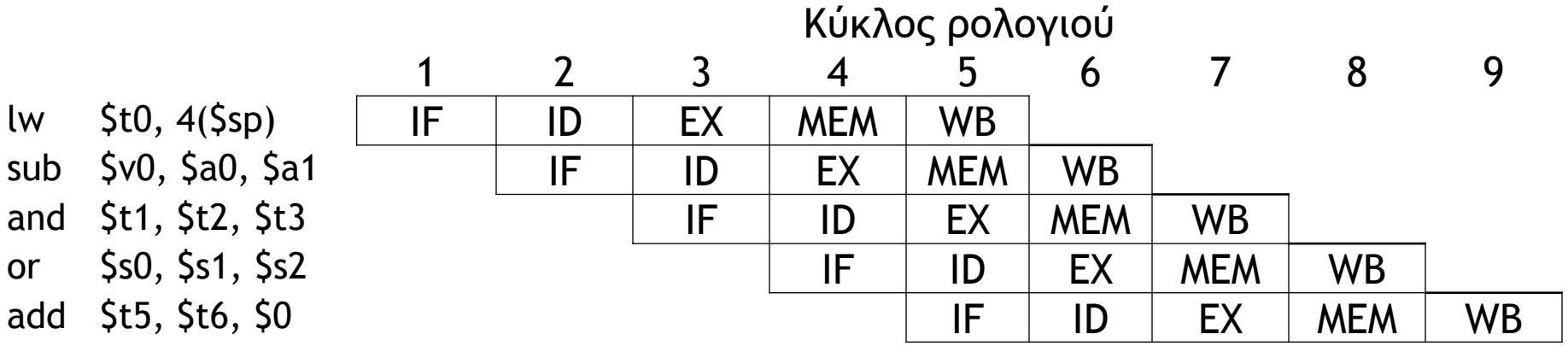
# Προσοχή

Τα πάντα κινούνται από αριστερά στα δεξιά εκτός:

1. Της απόφασης διακλάδωσης
2. Του Write Back



# Πιο συμβολικά




Στον κύκλο 5 ο επεξεργαστής έχει γεμίσει: υπάρχουν τόσες εντολές εν πτήση όσα και τα στάδια της διοχέτευσης

Από εδώ και πέρα θα εξετάσουμε τα προβλήματα που μπορεί να δημιουργήσει η μέθοδος της διοχέτευσης και πως αυτά επιλύονται.