ΚΕΦΑΛΑΙΟ 10

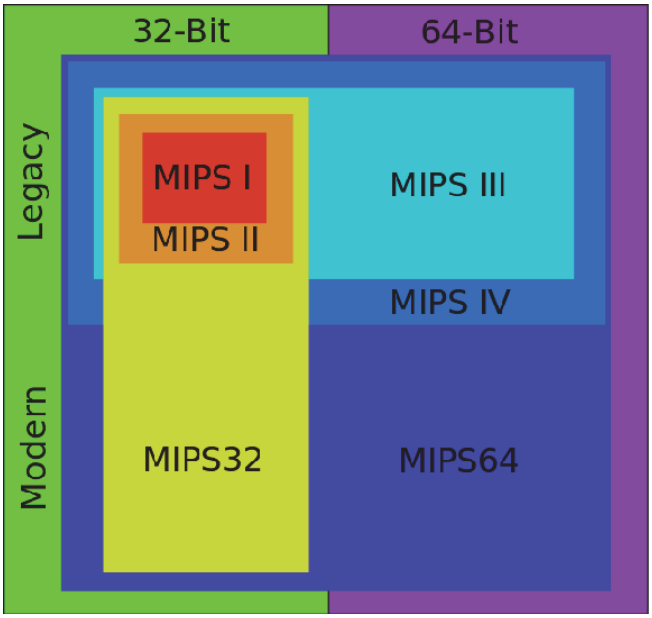
ΤΕΧΝΙΚΕΣ ΑΥΞΗΣΗΣ ΤΗΣ ΑΠΟΔΟΣΗΣ ΤΩΝ ΕΠΕΞΕΡΓΑΣΤΩΝ

10.1 Αρχιτεκτονική MIPS

1. Τι γνωρίζετε για την αρχιτεκτονική επεξεργαστών MIPS.

Τα μαθήματα σχετικά με την Αρχιτεκτονική των Υπολογιστών σε πολλά πανεπιστήμια βασίζονται στην αρχιτεκτονική MIPS.
Με τον όρο MIPS (*Microprocessor without Interlocked Pipeline Stages*) εννοούμε μία αρχιτεκτονική συνόλου εντολών γλώσσας μηχανής που αναπτύχθηκε από την εταιρεία MIPS Technologies και ανήκει στην κατηγορία επεξεργαστών με αρχιτεκτονική RISC (Reduced Instruction Set Computer). Οι πρώτες αρχιτεκτονικές MIPS ήταν των 32 bit, ενώ οι επόμενες ήταν των 32/64 bit.
Υπάρχουν πολλές εκδόσεις του συνόλου εντολών της αρχιτεκτονικής MIPS στις οποίες περιλαμβάνονται οι MIPS I, MIPS II, MIPS III, MIPS IV, MIPS V, MIPS32 (32 bit), and MIPS64 (64 bit). Η MIPS32/64 διαφέρει κυρίως από τις MIPS I–V διότι προδιαγράφει τον privileged kernel mode system control coprocessor επί πλέον από την αρχιτεκτονική MIPS σε user mode.
Αρχικά ο MIPS σχεδιάστηκε σαν επεξεργαστής γενικού σκοπού. Χρησιμοποιήθηκε αρχικά σε workstation και server. Από το 2017 χρησιμοποιείται σε μικρά gateways και routers.

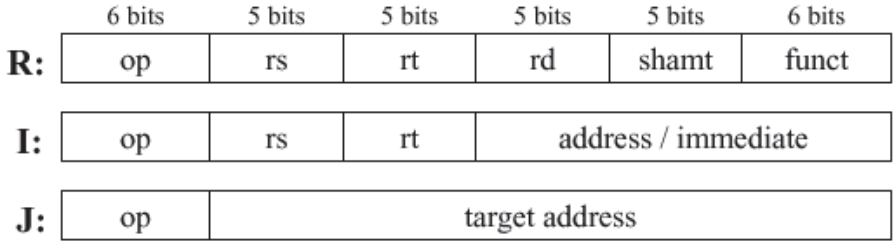
2. Μελετήστε το διάγραμμα των αρχιτεκτονικών MIPS που δίδεται στην συνέχεια.



3. Μελετήστε το υποσύνολο εντολών του MIPS που περιγράφεται στην συνέχεια.

Class	Instruction	Usage		Meaning	op	fn
Copy	Load upper immediate	`lui`	`rt, imm`	rt ← (imm, 0X0000)	15	
Arithmetic	Add	`add`	`rd,rs,rt`	rd ← (rs) + (rt)	0	32
	Subtract	`sub`	`rd,rs,rt`	rd ← (rs) − (rt)	0	34
	Set less than	`slt`	`rd,rs,rt`	rd ← if (rs) < (rd) then 1 else 0	0	42
	Add immediate	`addi`	`rt,rs,imm`	rt ← (rs) + imm	8	
	Set less than immediate	`slti`	`rd,rs,imm`	rd ← if (rs) < imm then 1 else 0	10	
Logic	AND	`and`	`rd,rs,rt`	rd ← (rs) ∧ (rt)	0	36
	OR	`or`	`rd,rs,rt`	rd ← (rs) ∨ (rt)	0	37
	XOR	`xor`	`rd,rs,rt`	rd ← (rs) ⊕ (rt)	0	38
	NOR	`nor`	`rd,rs,rt`	rd ← ((rs) ∨ (rt))′	0	39
	AND immediate	`andi`	`rt,rs,imm`	rt ← (rs) ∧ imm	12	
	OR immediate	`ori`	`rt,rs,imm`	rt ← (rs) ∨ imm	13	
	XOR immediate	`xori`	`rt,rs,imm`	rt ← (rs) ⊕ imm	14	
Memory access	Load word	`lw`	`rt,imm(rs)`	rt ← mem[(rs) + imm]	35	
	Store word	`sw`	`rt,imm(rs)`	mem[(rs) + imm] ← (rt)	43	
Control transfer	Jump	`j`	`L`	goto L	2	
	Jump register	`jr`	`rs`	goto (rs)	0	8
	Branch less than 0	`bltz`	`rs,L`	if (rs) < 0 then goto L	1	
	Branch equal	`beq`	`rs,rt,L`	if (rs) = (rt) then goto L	4	
	Branch not equal	`bne`	`rs,rt,L`	if (rs) ≠ (rt) then goto L	5	

4. Μελετήστε το format των εντολών της αρχιτεκτονικής MIPS που περιγράφεται στην συνέχεια.



op: basic operation of the instruction (opcode)
rs: first source operand register
rt: second source operand register
rd: destination operand register
shamt: shift amount
funct: selects the specific variant of the opcode (function code)
address: offset for load/store instructions (+/-2^{15})
immediate: constants for immediate instructions

5. Μελετήστε την λειτουργία που εκτελεί η εντολή του επεξεργαστή MIPS

add $8, $16, $17

και την κωδικοποίησή της σε γλώσσα μηχανής που περιγράφονται στην συνέχεια.

Η εντολή add $8, $16, $17 προσθέτει στο περιεχόμενο του καταχωρητή $16, το περιεχόμενο του καταχωρητή $17 και το αποτέλεσμα το τοποθετεί στον καταχωρητή $8.
Η μετάφραση αυτής της εντολής γίνεται σύμφωνα με το format R των εντολών MIPS. Ο κωδικός της εντολής είναι op=0→0b000000, και η λειτουργία fn=32→0b1000000.

R	000000	10000	10001	01000	00000	100000

ή 0x02114020 (στο δεκαεξαδικό).

6. Μελετήστε την λειτουργία της εντολής του επεξεργαστή μMIPS

   or $8, $16, $17

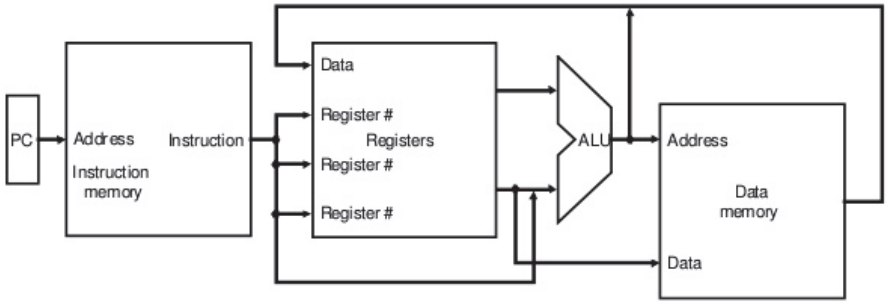
   και την κωδικοποίησή της σε γλώσσα μηχανής που περιγράφονται στην συνέχεια.

   Η εντολή αυτή κάνει λογικό OR ψηφίο με ψηφίο μεταξύ των περιεχομένων των καταχωρητών $16 και $17 και το αποτέλεσμα τοποθετείται στον καταχωρητή $8.
   Η μετάφραση της εντολής αυτής γίνεται σύμφωνα με το format R δίδεται στην συνέχεια. Ο κωδικός της εντολής είναι op=0→0b000000, και η λειτουργία fn=37→0b100101
   .

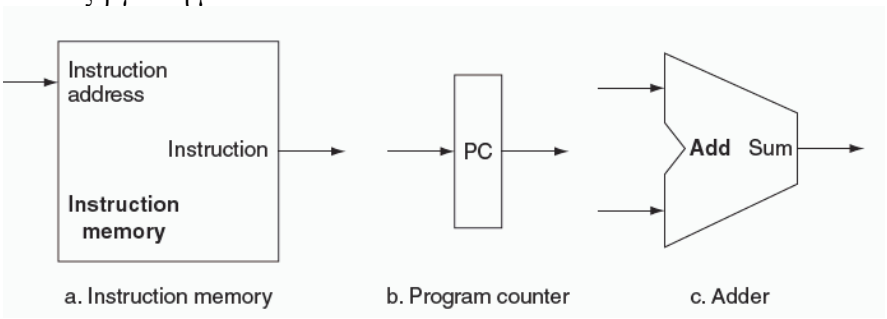
R	000000	10000	10001	01000	00000	100101

   ή 0x02114025 (στο δεκαεξαδικό)

7. Μελετήστε το βασικό διάγραμμα βαθμίδων της υλοποίησης του επεξεργαστή MIPS που δίδεται στην συνέχεια.

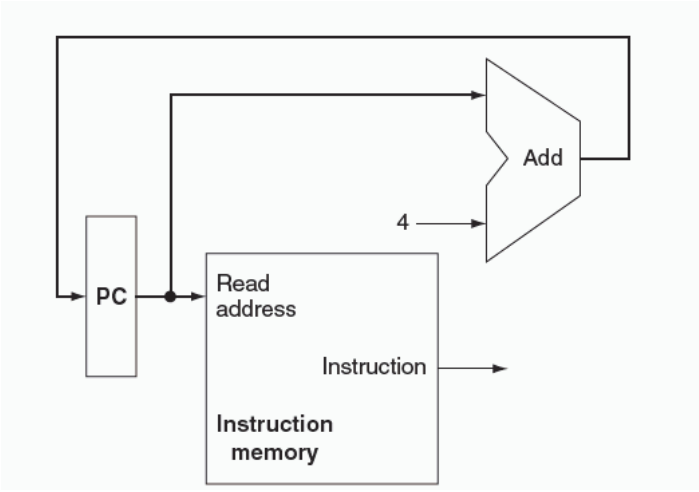


8. Μελετήστε τα κυκλώματα που δίδονται στην συνέχεια. Τα κυκλώματα αυτά θα χρησιμοποιηθούν για την σύνθεση της υπομονάδος ανάκλησης εντολών του επεξεργαστή μMIPS.



Στην *μνήμη εντολών* (*Instruction Memory*) τοποθετούνται τα προς εκτέλεση προγράμματα.
Ο *program counter* (PC) είναι ένας καταχωρητής των 32 bit στον οποίο τα δεδομένα εισόδου γράφονται στο τέλος κάθε κύκλου ωρολογίου. Χρησιμοποιείται για να δίνει διευθύνσεις στην μνήμη εντολών. Η είσοδος ωρολογίου δεν φαίνεται στο σχήμα.
Ο *προσθετής* (adder) είναι δυαδικός προσθετής των 32 bit.

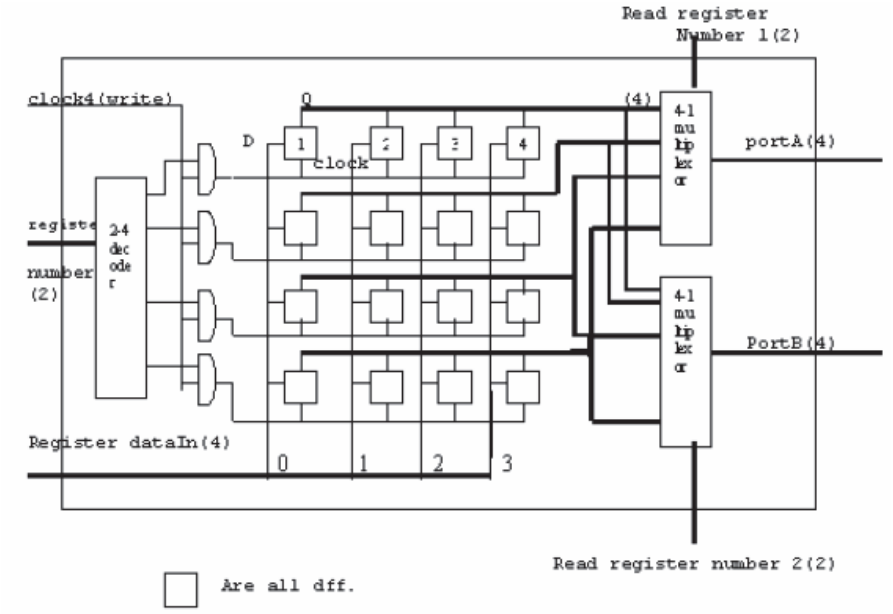
9. Μελετήστε το τμήμα του επεξεργαστή μMIPS με το οποίο γίνεται η ανάκληση εντολών από την μνήμη εντολών και αύξηση του Program Counter (PC) κατά 4 για να δείχνει την επόμενη εντολή που θα εκτελεσθεί για εντολές που δεν περιλαμβάνουν διακλάδωση.



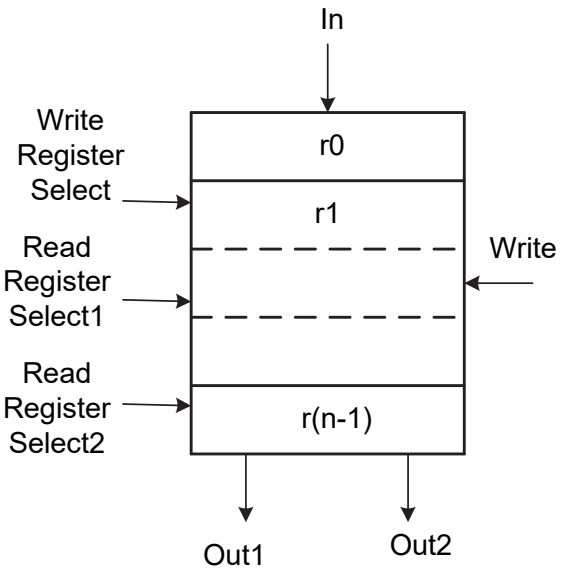
10. Τι γνωρίζετε για το register file.

    Το *register file* είναι διατάξεις από καταχωρητές που χρησιμοποιούνται εσωτερικά στις μονάδες επεξεργασίας δεδομένων (datapath) των επεξεργαστών. Στους σύγχρονους επεξεργαστές υπάρχουν register file με πολλές πόρτες ανάγνωσης δεδομένων.

11. Μελετήστε την δομή του register file 4x4 με δύο πόρτες εξόδου που δίδεται στην συνέχεια.

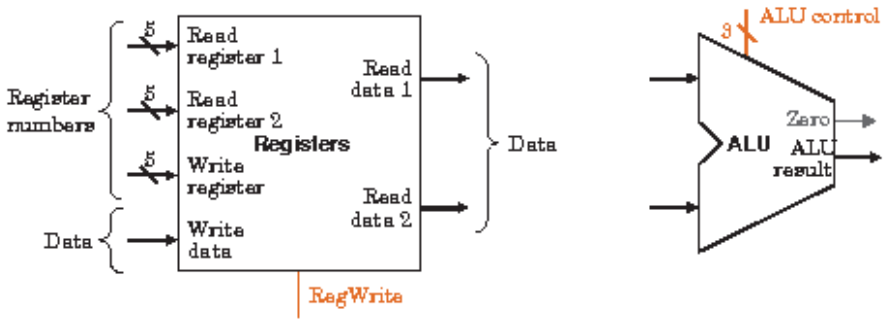


12. Μελετήστε το λογικό σύμβολο ενός register file με *n* καταχωρητές r0, r1,…, r(n-1) και δύο πόρτες ανάγνωσης δεδομένων που δίδεται στην συνέχεια.



Το *register file* (*registers*) αποτελείται από n (=2^k) καταχωρητές με μήκος m bit. Από το register file μπορεί να γίνει ταυτόχρονη ανάγνωση δύο καταχωρητών στις εξόδους Out 1 και Out 2. Το περιεχόμενο καταχωρητής θα βγει στις εξόδους Out 1 προσδιορίζεται από την τιμή των εισόδων Read register Select 1, ενώ το ποιος καταχωρητής θα βγει στις εξόδους Out 2 προσδιορίζεται από την τιμή των εισόδων Read register Select 2. Για να εγγραφούν δεδομένα σε έναν από τους καταχωρητές αυτά τοποθετούνται στην είσοδο In, στις εισόδους Write register Select τοποθετείται ο αριθμός του καταχωρητή που πρέπει να εγγραφούν και ενεργοποιείται η είσοδος Write.

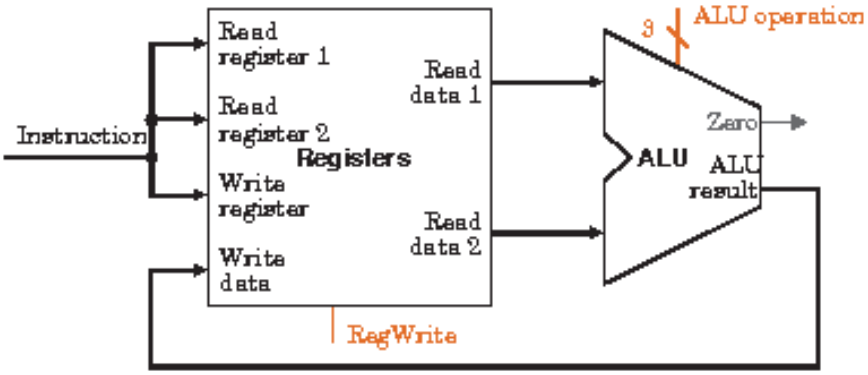
13. Μελετήστε το register file και την ALU που χρησιμοποιούνται στην αρχιτεκτονική MIPS που δίδεται στην συνέχεια.



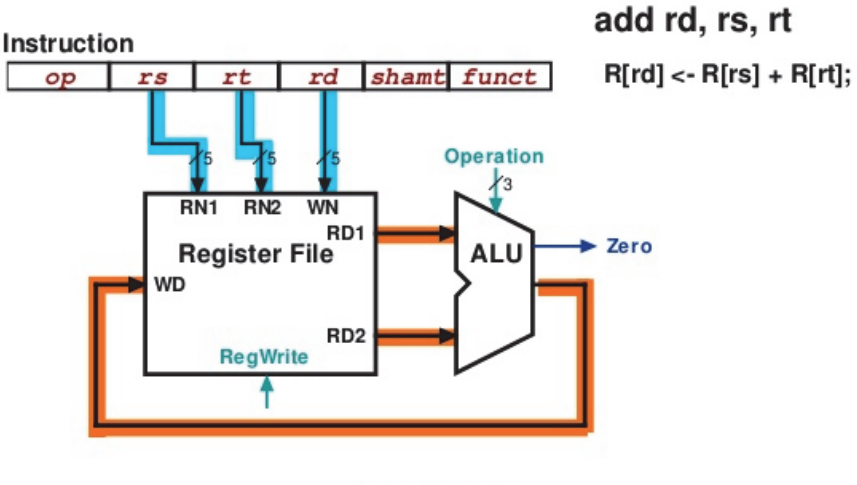
Το *register file* (*registers*) του MIPS αποτελείται από 32 καταχωρητές με μήκος 32 bit. Από το register file μπορεί να γίνει ταυτόχρονη ανάγνωση δύο καταχωρητών στις εξόδους Read data 1 και Read data 2. Το περιεχόμενο καταχωρητής θα βγει στις εξόδους Read data 1 προσδιορίζεται από την τιμή των εισόδων Read register 1, ενώ το ποιος καταχωρητής θα βγει στις εξόδους Read data 2 προσδιορίζεται από την τιμή των εισόδων Read register 2. Για να εγγραφούν δεδομένα σε έναν από τους καταχωρητές αυτά τοποθετούνται στην είσοδο write data, στις εισόδους Write register τοποθετείται ο αριθμός του καταχωρητή που πρέπει να εγγραφούν και ενεργοποιείται η είσοδος RegWrite.

Η *αριθμητική λογική μονάδα* (ALU) του MIPS κάνει αριθμητικές και λογικές πράξεις των 32 bit. Οι είσοδοι ALU *operation* προσδιορίζουν ποια αριθμητική ή λογική πράξη θα εκτελέσει η ALU. Η *έξοδος Zero* της αριθμητικής λογικής μονάδας γίνεται 1 όταν η έξοδος ALU result έχει την τιμή 0x00000000.

14. Μελετήστε την διασύνδεση του register file με την ALU στον MIPS που περιγράφεται ώστε να είναι δυνατή η εκτέλεση των εντολών του τύπου
add rd, rs, rt.

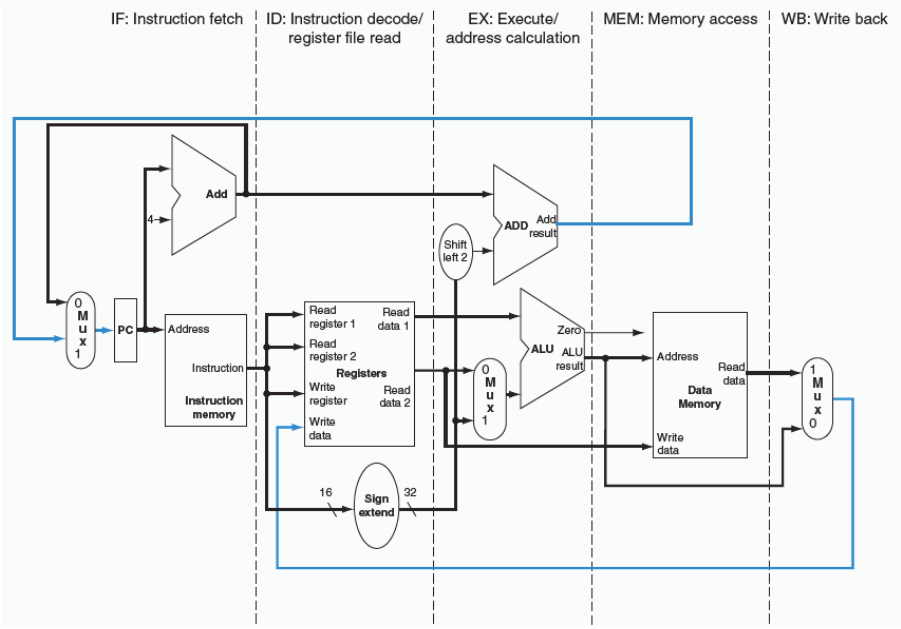


15. Μελετήστε την εκτέλεση της εντολής
add rd, rs, rt
στην αρχιτεκτονική MIPS.

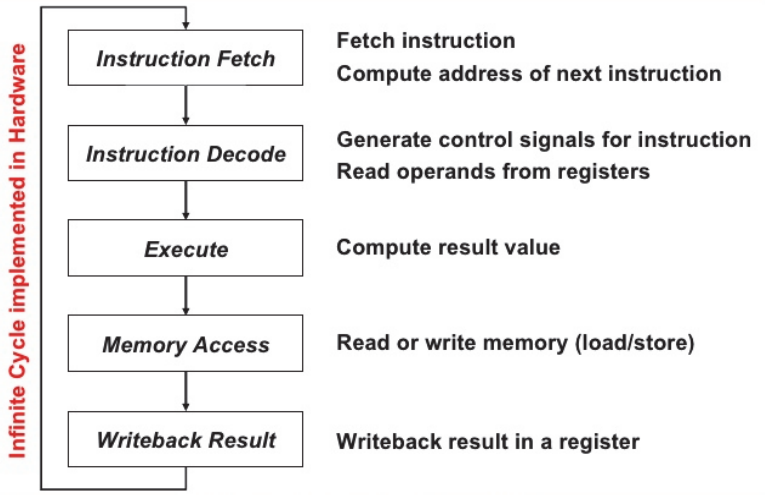


10.2 Αρχιτεκτονική Pipeline

16. Μελετήστε την αρχιτεκτονική του επεξεργαστή MIPS που δίδεται στην συνέχεια και τις λειτουργίες που εκτελεί η κάθε βαθμίδα..



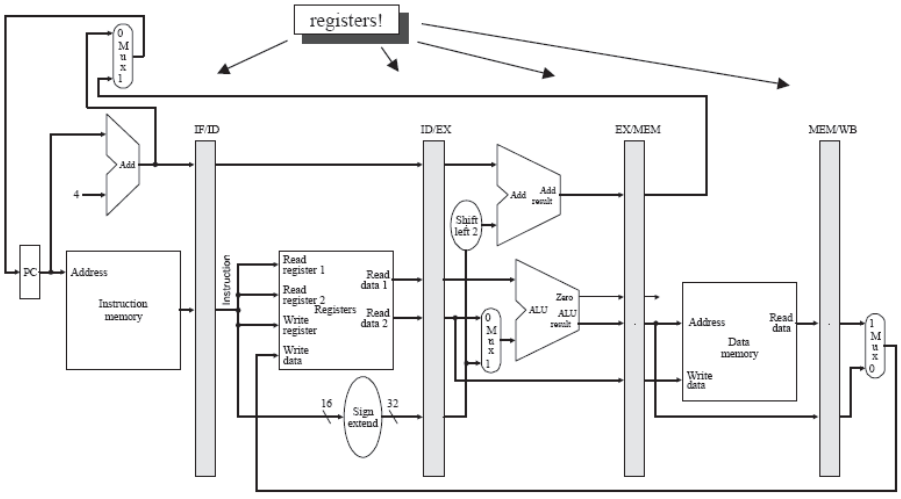
17. Μελετήστε τον κύκλο εντολής του MIPS που περιγράφεται στην συνέχεια.



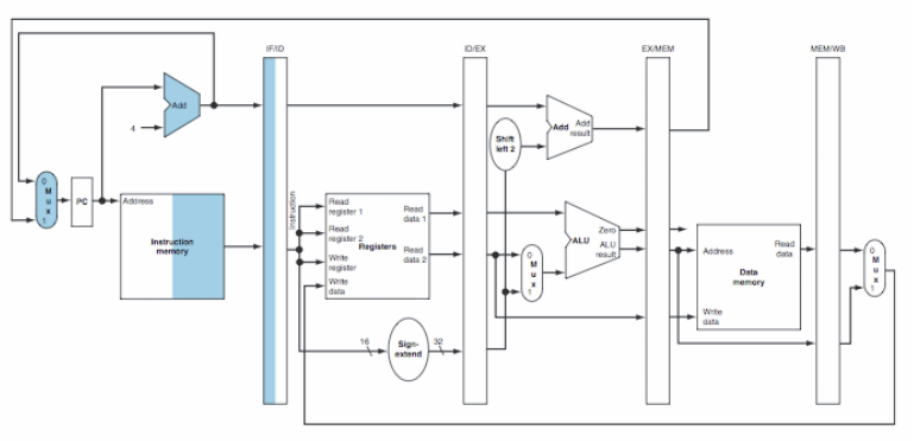
18. Τι γνωρίζετε για την τεχνική pipeline execution των εντολών σε γλώσσα μηχανής.

    Το *pipelining* είναι τεχνική αλληλοεπικάλυψης της εκτέλεσης των εντολών γλώσσας μηχανής σε έναν επεξεργαστή, ώστε να περιορισθεί ο χρόνος εκτέλεσης ενός συνόλου εντολών. Για την *υλοποίηση του pipelining* η μονάδα επεξεργασίας δεδομένων διαιρείται σε βαθμίδες και τοποθετούνται pipeline latches (registers) μεταξύ των βαθμίδων.
    Στην αρχή κάθε κύκλου ωρολογίου γίνεται εγγραφή των εξόδων των βαθμίδων στα pipeline latches των οποίων οι έξοδοι παραμένουν σταθερές κατά το υπόλοιπο του κύκλου για να χρησιμοποιηθούν σαν είσοδοι από την επόμενη βαθμίδα.

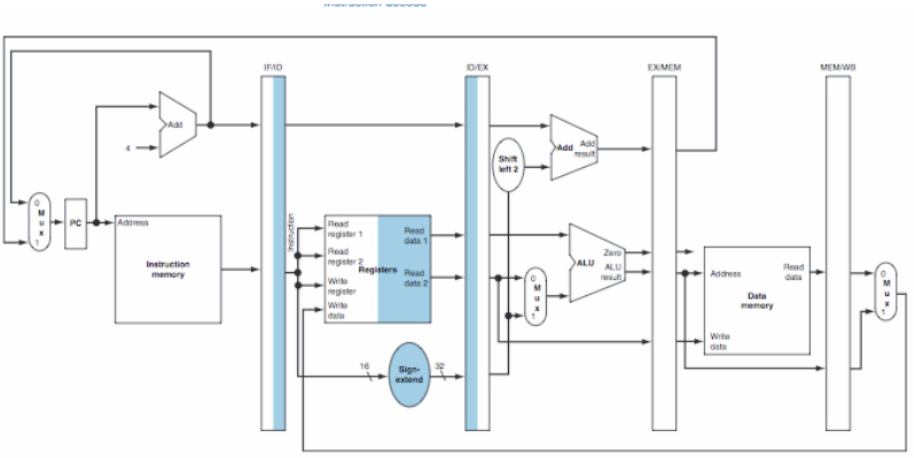
19. Μελετήστε την αρχιτεκτονική MIPS η οποία έχει τροποποιηθεί ώστε να έχει την δυνατότητα για pipeline execution.



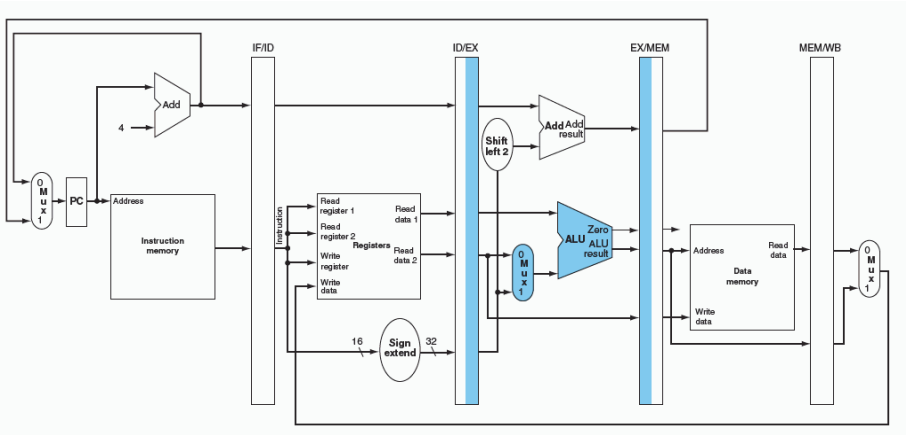
20. Μελετήστε την ανάκληση εντολής (IF) σε αρχιτεκτονική MIPS η οποία έχει τροποποιηθεί ώστε να έχει την δυνατότητα για pipeline execution.



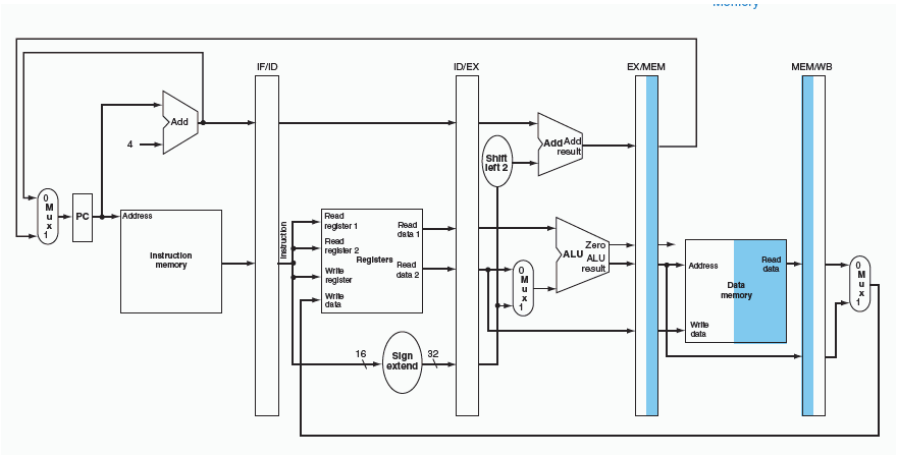
21. Μελετήστε την ανάγνωση του register file (RR ή ID) στην αρχιτεκτονική MIPS η οποία έχει τροποποιηθεί ώστε να έχει την δυνατότητα για pipeline execution.



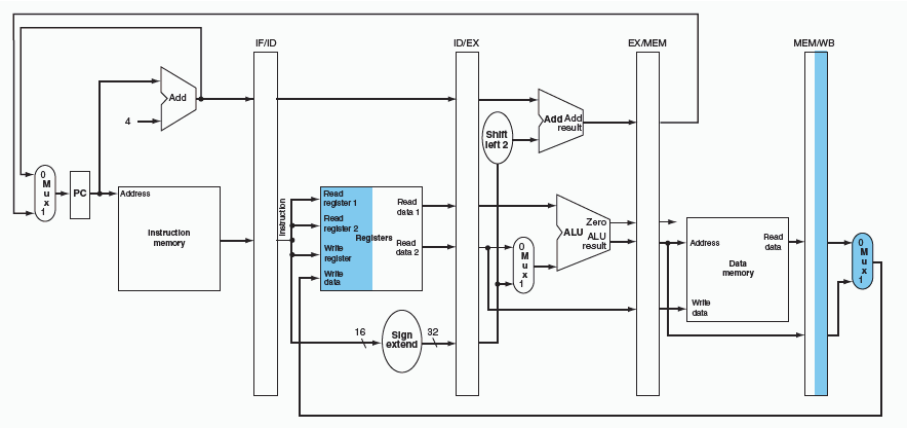
22. Μελετήστε την εκτέλεση της εντολής (EX) στην αρχιτεκτονική MIPS η οποία έχει τροποποιηθεί ώστε να έχει την δυνατότητα για pipeline execution.



23. Μελετήστε την προσπέλαση μνήμης (MEM) στην αρχιτεκτονική MIPS η οποία έχει τροποποιηθεί ώστε να έχει την δυνατότητα για pipeline execution.

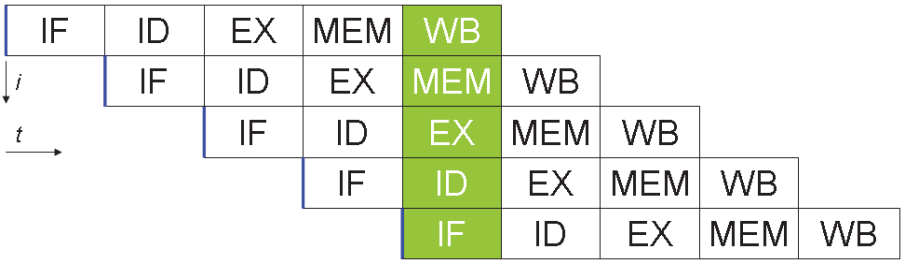


24. Μελετήστε την εγγραφή του αποτελέσματος (WB) στην αρχιτεκτονική MIPS η οποία έχει τροποποιηθεί ώστε να έχει την δυνατότητα για pipeline execution.



25. Μελετήστε την εκτέλεση διαδοχικών εντολών γλώσσας μηχανής σε αρχιτεκτονική MIPS χωρίς pipeline.

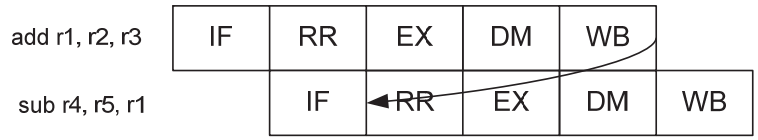
26. Μελετήστε την εκτέλεση τμήματος εντολών γλώσσας μηχανής σε επεξεργαστή MIPS με δυνατότητα pipelined execution.



27. Τι γνωρίζετε για τα RAW data hazard σε υπολογιστές με αρχιτεκτονική για pipeline execution.

    Τα *read after write* (RAW) *data hazard* αναφέρονται σε μία κατάσταση κατά την εκτέλεση ενός προγράμματος όπου μία εντολή πρόκειται να χρησιμοποιήσει ένα αποτέλεσμα το οποίο δεν έχει υπολογιστεί ή ανακτηθεί από προηγούμενη εντολή. Αυτό μπορεί να προκύψει όταν η εντολή εκτελείται μετά από κάποια εντολή και έχει εκτελεστεί μόνο ένα τμήμα της προηγούμενης εντολής στο pipeline.

28. Μελετήστε το RAW data hazard κατά την εκτέλεση των εντολών
    add **r1**, r2, r3
    sub r4, r5, **r1**
    σε επεξεργαστή MIPS με αρχιτεκτονική pipeline που δίδεται στην συνέχεια.



29. Μελετήστε την εκτέλεση ενός τμήματος προγράμματος στο οποίο υπάρχει RAW data hazard σε επεξεργαστή MIPS με αρχιτεκτονική pipeline που δίδεται στην συνέχεια στο οποίο γίνεται stall στην εκτέλεση των εντολών.

	0	1	2	3	4	5	6	7	8	9	10	11
ADD $r0, $r1, $r2	IF	ID	EX	MEM	WB							
SUB $r4, $r0, $r3		IF	*	*	*	ID	EX	MEM	WB			
AND $r5, $r0, $r6			IF	*	*	*	ID	EX	MEM	WB		
OR $r7, $r0, $r8				IF	*	*	*	ID	EX	MEM	WB	
XOR $r9, $r0, $r10					IF	*	*	*	ID	EX	MEM	WB

30. Τι γνωρίζετε για την τεχνική data forwarding.

    Data forwarding (ή *operand forwarding*) είναι μία τεχνική βελτιστοποίησης στις CPU με αρχιτεκτονική pipeline ώστε να περιοριστεί η μείωση στην απόδοση η οποία οφείλεται στα pipeline stalls. Τα data hazard μπορούν να προκαλέσουν pipeline stall όταν η εντολή που εκτελείται πρέπει να περιμένει τα αποτελέσματα από κάποια προηγούμενη εντολή η οποία δεν έχει ακόμα τελειώσει.