ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Προηγμένη Σχεδίαση Ψηφιακών Συστημάτων
Ψηφιακά Ολοκληρωμένα Κυκλώματα ΙΙ

Ενότητα 3

Σχεδίαση και Υλοποίηση σε FPGA ενός Επεξεργαστή RISC (MIPS R2000)

Καθ. Αντώνης Πασχάλης

2012

Κίνητρα Υλοποίησης Επεξεργαστών σε FPGA

- Άνοιγμα της ψαλίδας του κόστους μίας υλοποίησης σε ASIC σε σχέση με μία υλοποίηση σε FPGA
- Ασφυκτικοί χρονικοί περιορισμοί υλοποίησης, ώστε να ικανοποιείται το "time to market"
- Ραγδαία αύξηση των ενσωματωμένων συστημάτων, που υλοποιούνται σε FPGA και βασίζονται σε πυρήνες επεξεργαστών (processor cores)
- Hard processor cores:
  - Xilinx Virtex II Pro (PowerPC based),
  - Altera Excalibur
- Soft processor cores (HDL versions):
  - Xilinx MicroBlaze
  - Altera Nios

Κίνητρα Υλοποίησης Soft Processor Cores σε FPGA

- Οι hard processor cores πλεονεκτούν σε μέγεθος, απόδοση και κατανάλωση ισχύος και παρέχουν μία σχετικά φθηνή λύση, αλλά …
- Υπάρχει περιορισμός στον αριθμό των επεξεργαστών, που μπορούν να χρησιμοποιηθούν, ώστε να μην καλύπτονται οι συγκεκριμένες ανάγκες μίας εφαρμογής
- Δεν προσαρμόζονται στις χαμηλότερες απαιτήσεις για απόδοση, που μπορεί να έχει μία εφαρμογή, παρέχοντας δυνατότητα αλλαγής της πολυπλοκότητάς τους
- Εμφανίζουν προβλήματα στο rooting που προέρχονται από τη συγκεκριμένη θέση τοποθέτησης των επεξεργαστών
- Μείωση του yield (των ορθά παραγόμενων FPGAs)
- Μείωση της πελατειακής βάσης για το συγκεκριμένο FPGA

Σχεδιαστικές Επιλογές για Υλοποίηση Soft Processor Cores σε FPGA

- **Ιδιαίτερη μελέτη των κρίσιμων καθυστερήσεων διάδοσης, που εμφανίζονται κατά τη διασύνδεση των Block RAMs και των ενσωματωμένων multipliers με τις υπόλοιπες λειτουργικές μονάδες του datapath**
- **Στην RTL υλοποίηση προσεκτική μελέτη της καθυστέρησης διάδοσης από καταχωρητή σε καταχωρητή και αύξηση των ενδιάμεσων προσωρινών καταχωρητών, όπου απαιτείται**
- **Βέλτιστη εκμετάλλευση των εισόδων των LUTs**
- **Ιδιαίτερη μελέτη των σημάτων που έχουν μεγάλα fan-outs**
- **Οι multipliers πρέπει να έχουν καταχωρητές στην είσοδό τους και την έξοδό τους, όπως οι Block RAMs**
- **Οι multipliers μπορεί να είναι και pipelined**

Σχεδιαστικές Επιλογές για Υλοποίηση Soft Processor Cores σε FPGA

- **Αποφεύγουμε την υλοποίηση πολύπλοκης μονάδας ελέγχου**
  - **Εάν το μέγεθος της Μνήμης Εντολών, που υλοποιείται με Block RAMs, το επιτρέπει, χρησιμοποιούμε κάποια ψηφία, επιπλέον των 32 ψηφίων της εντολής, σαν σήματα ελέγχου**
- **Χρησιμοποιούμε αθροιστές και αφαιρέτες, όπου είναι δυνατό, γιατί είναι μία φθηνή και αποδοτική λύση για FPGA**
  - **Στη σύγκριση χρησιμοποιούμε αφαιρέτη αντί για συγκριτή**
- **Χρησιμοποιούμε τους πολυπλέκτες με φειδώ, γιατί είναι μία ακριβή και μη αποδοτική λύση για FPGA**
  - **Στην ολίσθηση για παράδειγμα μπορούμε να χρησιμοποιήσουμε υπάρχοντες multipliers, αντί ολισθητή**
- **Χρησιμοποιούμε τα ξεχωριστά read και write ports των διαθέσιμών dual-port Block RAMs, όπου απαιτείται**
  - **Υλοποιούμε το αρχείο καταχωρητών χρησιμοποιώντας μία dual-port Block RAM 32x32, όπου τα 2 data output ports αντιστοιχούν στα 2 read ports**

Φάσεις Εκτέλεσης μίας Εντολής

- **Φάση 1: Instruction Fetch (IF)**
- **Φάση 2: Instruction Decode and Register File Read (ID)**
- **Φάση 3: Instruction Execution or Address Calculation (EX)**
- **Φάση 4: Data Memory Access (MEM)**
- **Φάση 5: Write Back to Register File (WB)**

Φάση 1: Instruction Fetch (IF)

- **Διάβασμα της εντολής από τη μνήμη εντολών IM SRAM. Αποθήκευση στον καταχωρητή εντολών IR**
- **Υπολογισμός της διεύθυνσης της αμέσως επόμενης εντολής (PC+4) με τη χρήση του αυξητή κατά 4 INC. Αποθήκευση στον καταχωρητή N.**
- **Αποθήκευση στον καταχωρητή P των 4 msb της τρέχουσας τιμής του PC.**

Φάση 2: Instruction Decode and Register File Read (ID)

- **Αποκωδικοποίηση της εντολής στη μονάδα ελέγχου Control.**
- **Υπολογισμός της διεύθυνσης για ψευδο-άμεση διευθυνσιοδότηση με τη χρήση του κυκλώματος PSD. Αποθήκευση στον καταχωρητή D.**
- **Ανάκληση καταχωρητών από το αρχείο καταχωρητών Reg. File. Αποθήκευση στους καταχωρητές A και B.**
- **Επέκταση πρόσημου ή μηδενός του πεδίου immediate με τη χρήση του κυκλώματος EXT S/Z. Αποθήκευση στον καταχωρητή I.**

Φάση 3: Instruction Execution or Address Calculation (EX)

- **Επιλογή από τους τελεστέους, που είναι αποθηκευμένοι στους καταχωρητές A, B και I, για την εκτέλεση πράξης με τη χρήση του πολυπλέκτη A-L mux.**
- **Εκτέλεση αριθμητικών πράξεων (πρόσθεση, αφαίρεση) προσημασμένων και μη προσημασμένων (προσοχή), λογικών πράξεων και ολισθήσεων στη μονάδα A-L. Αποθήκευση του αποτελέσματος στον καταχωρητή ALU out.**
- **Εκτέλεση προσημασμένου πολλαπλασιασμού. Αποθήκευση του αποτελέσματος στους καταχωρητές Hi και Lo.**
- **Ανίχνευση μηδενός στην έξοδο της ALU. Ενημέρωση της σημαίας Z που αποθηκεύεται στο D-FF Z.**
- **Ενημέρωση της σημαίας Ne που αποθηκεύεται στο D-FF Ne.**
- **Ανίχνευση υπερχείλισης στην έξοδο της ALU. Ενημέρωση της σημαίας OV που αποθηκεύεται στο D-FF OV.**
  - **Άλλες σημαίες, εάν απαιτείται, υλοποιούνται αντίστοιχα.**

Φάση 3: Instruction Execution or Address Calculation (EX)

συνέχεια

- **Υπολογισμός διεύθυνσης της μνήμης δεδομένων DM SRAM για διευθυνσιοδότηση με μετατόπιση. Αποθήκευση της διεύθυνσης στον καταχωρητή ALU out.**
- **Αποθήκευση στον καταχωρητή MDR in του τελεστέου, που πρόκειται να εγγραφεί στη μνήμη δεδομένων και είναι αποθηκευμένος στον καταχωρητή B.**
- **Υπολογισμός διεύθυνσης της μνήμης εντολών IM SRAM για PC-σχετική διευθυνσιοδότηση στη μονάδα ADD. Αποθήκευση της διεύθυνσης στον καταχωρητή M.**
- **Μεταφορά δεδομένων στους καταχωρητές Hi ή Lo.**

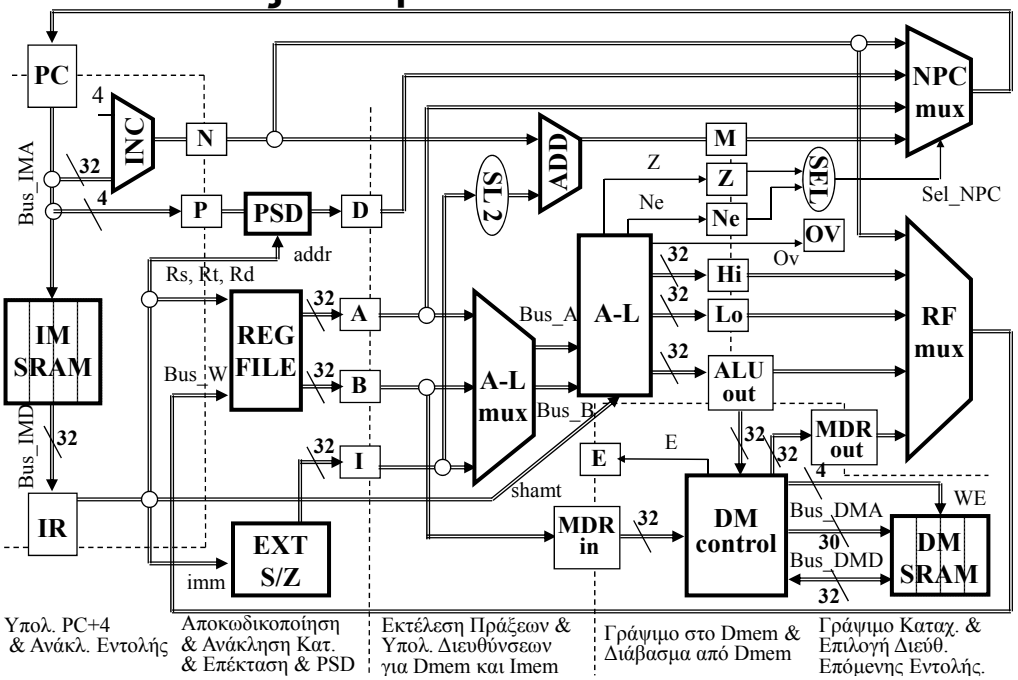
Φάση 4: Data Memory Access (MEM)

- **Ανίχνευση λάθους ευθυγράμμισης στη διεύθυνση της μνήμης δεδομένων, που είναι αποθηκευμένη στον καταχωρητή ALU out, με τη χρήση της μονάδας DM control. Ενημέρωση της σημαίας E που αποθηκεύεται στο D flip-flop E.**
- **Ενεργοποίηση των κατάλληλων σημάτων γραψίματος WE, ανάλογα με το μέγεθος του τελεστέου, με τη χρήση της μονάδας DM control.**
- **Κατάλληλη αποθήκευση (ανάλογα με το μέγεθος του), του τελεστέου που διαβάζεται από τη μνήμη δεδομένων, στον καταχωρητή MDR out με τη χρήση της μονάδας DM control.**
- **Προσπέλαση της μνήμης δεδομένων DM SRAM μέσω της αρτηρίας διευθύνσεων Bus_DMA και της αμφίδρομης αρτηρίας δεδομένων Bus_DMD.**

Φάση 5: Write Back to Register File (WB)

- **Επιλογή δεδομένων για εγγραφή στο αρχείο καταχωρητών, που είναι αποθηκευμένα στους καταχωρητές ALU out, Hi, Lo, N και MDR out, με τη χρήση του πολυπλέκτη RF mux.**
- **Επιλογή της διεύθυνσης της επόμενης εντολής με τη χρήση του πολυπλέκτη NPC mux και του κυκλώματος επιλογής SEL. Υποστηρίζονται οι ακόλουθες περιπτώσεις:**
  - **PC = N = PC+4 για ακολουθιακή εκτέλεση των εντολών. Η επιλογή μπορεί να γίνει το ταχύτερο στη Φάση 2.**
  - **PC = M = PC+4+4μ για εντολές διακλάδωσης με συνθήκη (συνδυασμοί τιμών για τις σημαίες Z και Ne). PC = N = PC+4 για εντολές διακλάδωσης με συνθήκη (συνδυασμοί τιμών για τις σημαίες Z και Ne). Η επιλογή μπορεί να γίνει το ταχύτερο στη Φάση 4.**
  - **PC = A για εντολές μεταπήδησης με διευθυνσιοδότηση καταχωρητή. Η επιλογή μπορεί να γίνει το ταχύτερο στη Φάση 3.**
  - **PC = D για εντολές μεταπήδησης με ψευδο-άμεση διευθυνσιοδότηση. Η επιλογή μπορεί να γίνει το ταχύτερο στη Φάση 3.**

Δίοδος Δεδομένων Πολλών Κύκλων



Load Byte (Signed) (Τύπου I)

- **LB rt,immediate(rs) op=0x20**
  rt <- memory (base + offset)

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU

ALU : Effective address calculation

Load Byte (Signed) (Τύπου I)

- **LB rt,immediate(rs) op=0x20**

MEM	E = 0 (χωρίς έλεγχο ευθυγράμμισης)	DM CONTROL
	WE = 0000	DM CONTROL
	Bus_DMA = ALU_out	DM CONTROL
	Bus_DMD = DM[Bus_DMA]	DM SRAM
	MDR_out = Sign_ext[Bus_DMD[0/1/2/3]]*	DM CONTROL
WB	Bus_W = MDR_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

*** Επιλέγεται ένα από τα 4 bytes (0/1/2/3) του Bus_DMD ανάλογα με την τιμή των 2 lsb του ALU_out (00/01/10/11)**

Load Byte Unsigned (Τύπου I)

- **LBU rt,immediate(rs) op=0x24**
  rt <- memory [base + offset]

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU

Load Byte Unsigned (Τύπου I)

- **LBU rt,immediate(rs) op=0x24**

MEM	E = 0 (χωρίς έλεγχο ευθυγράμμισης)	DM CONTROL
	WE = 0000	DM CONTROL
	Bus_DMA = ALU_out	DM CONTROL
	Bus_DMD = DM[Bus_DMA]	DM SRAM
	MDR_out = Zero_ext[Bus_DMD[0/1/2/3]]*	DM CONTROL
WB	Bus_W = MDR_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

*** Επιλέγεται ένα από τα 4 bytes (0/1/2/3) του Bus_DMD ανάλογα με την τιμή των 2 lsb του ALU_out (00/01/10/11)**

Load Halfword (Signed) (Τύπου I)

- **LH rt,immediate(rs) op=0x21**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU

Load Halfword (Signed) (Τύπου I)

- **LH rt,immediate(rs) op=0x21**

MEM	E = DM_error (έλεγχος ευθυγράμμισης)	DM CONTROL
	WE = 0000	DM CONTROL
	Bus_DMA = ALU_out	DM CONTROL
	Bus_DMD = DM[Bus_DMA]	DM SRAM
	MDR_out = Sign_ext[Bus_DMD[0&1/2&3]]*	DM CONTROL
WB	Bus_W = MDR_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

*** Επιλέγονται 2 από τα 4 bytes (0&1/2&3) του Bus_DMD ανάλογα με την τιμή των 2 lsb του ALU_out (00/10)**

Restriction

Load Halfword Unsigned (Τύπου I)

- **LHU rt,immediate(rs) op=0x25**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU

Load Halfword Unsigned (Τύπου I)

- **LHU rt,immediate(rs) op=0x25**

MEM	E = DM_error (έλεγχος ευθυγράμμισης)	DM CONTROL
	WE = 0000	DM CONTROL
	Bus_DMA = ALU_out	DM CONTROL
	Bus_DMD = DM[Bus_DMA]	DM SRAM
	MDR_out = Zero_ext[Bus_DMD[0&1/2&3]]*	DM CONTROL
WB	Bus_W = MDR_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

*** Επιλέγονται 2 από τα 4 bytes (0&1/2&3) του Bus_DMD ανάλογα με την τιμή των 2 lsb του ALU_out (00/10)**

Restriction

Load Word (Τύπου I)

- **LW rt,immediate(rs) op=0x23**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU

Load Word (Τύπου I)

- **LW rt,immediate(rs) op=0x23**

MEM	E = DM_error (έλεγχος ευθυγράμμισης)	DM CONTROL
	WE = 0000	DM CONTROL
	Bus_DMA = ALU_out	DM CONTROL
	Bus_DMD = DM[Bus_DMA]	DM SRAM
	MDR_out = Bus_DMD[0&1&2&3]*	DM CONTROL
WB	Bus_W = MDR_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

*** Επιλέγονται και τα 4 bytes (0&1&2&3) του Bus_DMD όταν η τιμή των 2 lsb του ALU_out είναι 00**

Restriction

Store (Least Significant) Byte (Τύπου I)

- **SB rt,immediate(rs) op=0x28**
  memory [base + offset] <- rt

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	MDR_in = B	
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU

Store (Least Significant) Byte (Τύπου I)

- **SB rt,immediate(rs) op=0x28**

MEM	E = 0 (χωρίς έλεγχο ευθυγράμμισης)	DM CONTROL
	WE = 0001/0010/0100/1000	DM CONTROL
	Bus_DMA = ALU_out	DM CONTROL
	Bus_DMD[0/1/2/3]* = MDR_in (LSB)	DM CONTROL
	DM[Bus_DMA] = Bus_DMD[0/1/2/3]*	DM SRAM
WB**	PC = N	SEL - NPC MUX

*** Αποκτά τιμή το ένα από τα 4 bytes (0/1/2/3) του Bus_DMD ανάλογα με την τιμή των 2 lsb του ALU_out (00/01/10/11)**

**** Οι φάσεις MEM και WB εκτελούνται στον ίδιο κύκλο**

Store (Least Significant) Halfword (Τύπου I)

- **SH rt,immediate(rs) op=0x29**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	MDR_in = B	
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU

Store (Least Significant) Halfword (Τύπου I)

- **SH rt,immediate(rs) op=0x29**

MEM	E = DM_error (έλεγχος ευθυγράμμισης)	DM CONTROL
	WE = 0011/1100	DM CONTROL
	Bus_DMA = ALU_out	DM CONTROL
	Bus_DMD[0&1/2&3]* = MDR_in (LSH)	DM CONTROL
	DM[Bus_DMA] = Bus_DMD[0&1/2&3]*	DM SRAM
WB**	PC = N	SEL - NPC MUX

*** Αποκτούν τιμή τα 2 από τα 4 bytes (0&1/2&3) του Bus_DMD ανάλογα με την τιμή των 2 lsb του ALU_out (00/10)**

Restriction

**** Οι φάσεις MEM και WB εκτελούνται στον ίδιο κύκλο**

Store Word (Τύπου I)

- **SW rt,immediate(rs) op=0x2B**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	MDR_in = B	
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU

Store Word (Τύπου I)

- **SW rt,immediate(rs) op=0x2B**

MEM	E = DM_error (έλεγχος ευθυγράμμισης)	DM CONTROL
	WE = 1111	DM CONTROL
	Bus_DMA = ALU_out	DM CONTROL
	Bus_DMD[0&1&2&3]* = MDR_in	DM CONTROL
	DM[Bus_DMA] = Bus_DMD[0&1&2&3]*	DM SRAM
WB**	PC = N	SEL - NPC MUX

*** Αποκτούν τιμή και τα 4 bytes (0&1&2&3) του Bus_DMD όταν η τιμή των 2 lsb του ALU_out είναι 00**

Restriction

**** Οι φάσεις MEM και WB εκτελούνται στον ίδιο κύκλο**

Add Immediate (Signed) (Τύπου I)

- **ADDI rt, rs, immediate op=0x08**

  rt <- rs + immediate

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Add Immediate Unsigned (Τύπου I)

- **ADDIU rt, rs, immediate op=0x09**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για μη προσ/μένους	A-L ALU (optional)
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

AND Immediate (Unsigned) (Τύπου I)

- **`ANDI rt, rs, immediate op=0x0C`**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Zero_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A AND Bus_B	A-L ALU, A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

OR Immediate (Unsigned) (Τύπου I)

- **`ORI rt, rs, immediate op=0x0D`**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Zero_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A OR Bus_B	A-L ALU, A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

XOR Immediate (Unsigned) (Τύπου I)

- **XORI rt, rs, immediate op=0x0E**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Zero_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	ALU_out = Bus_F = Bus_A XOR Bus_B	A-L ALU, A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Load Upper Immediate (Τύπου I)

- **LUI rt, immediate (rs = 0) op=0x0F**

rt <- immediate || 0^{16}

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	I = Sign_ext*[IR[imm]]	EXT S/Z
EX	Bus_B = I	A-L MUX
	Bus_S = Bus_B SLL by shamt = 16 (internally)	A-L SHIFTER
	ALU_out = Bus_S	A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

*** Δεν έχει σημασία αφού ακολουθεί ολίσθηση στο EX**

Move From Hi (Τύπου R)

- **MFHI rd op=0x00 funct=0x10**

rd <- hi

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
WB	Bus_W = Hi	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Move From Lo (Τύπου R)

- **MFLO rd op=0x00 funct=0x12**

rd <- lo

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
WB	Bus_W = Lo	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Move To Hi (Τύπου R)

- **MTHI rs op=0x00 funct=0x11**

hi <- rs

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
EX	Hi = A	
WB*	PC = N	SEL - NPC MUX

*** Οι φάσεις EX και WB εκτελούνται στον ίδιο κύκλο**

Move To Lo (Τύπου R)

- **MTLO rs op=0x00 funct=0x13**

lo <- rs

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
EX	Lo = A	
WB*	PC = N	SEL - NPC MUX

*** Οι φάσεις EX και WB εκτελούνται στον ίδιο κύκλο**

Add (Signed) (Τύπου R)

- **ADD rd, rs, rt op=0x00 funct=0x20**

```
rd <- rs + rt
```

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Add Unsigned (Τύπου R)

- **ADDU rd, rs, rt op=0x00 funct=0x21**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A + Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για μη προσ/μένους	A-L ALU (optional)
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Subtract (Signed) (Τύπου R)

- **SUB rd, rs, rt op=0x00 funct=0x22**

  rd <- rs - rt

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Subtract Unsigned (Τύπου R)

- **SUBU rd, rs, rt op=0x00 funct=0x23**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για μη προσ/μένους	A-L ALU (optional)
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Multiply (Signed) (Τύπου R)

- **MULT rs, rt op=0x00 funct=0x18**

[Hi:Lo] <- rs x rt

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX*	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	Hi&Lo = Bus_A x Bus_B (signed)	A-L MULTIPLIER
WB**	PC = N	SEL - NPC MUX

***Ο αριθμός των κύκλων για τη φάση EX εξαρτάται από την υλοποίηση του πολλαπλασιαστή (ακολουθιακός, array)**

**** Η φάση WB εκτελείται στον τελευταίο κύκλο της φάσης EX**

AND (Τύπου R)

- **AND rd, rs, rt op=0x00 funct=0x24**

  rd <- rs *AND* rt

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A AND Bus_B	A-L ALU, A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

OR (Τύπου R)

- **OR rd, rs, rt op=0x00 funct=0x25**

  rd <- rs *OR* rt

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A OR Bus_B	A-L ALU, A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

XOR (Τύπου R)

- **XOR rd, rs, rt op=0x00 funct=0x26**

  rd <- rs *XOR* rt

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A XOR Bus_B	A-L ALU, A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

NOR (Τύπου R)

- **NOR rd, rs, rt op=0x00 funct=0x27**

  rd <- rs *NOR* rt

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A NOR Bus_B	A-L ALU, A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Shift Left Logical (Τύπου R)

- **SLL rd, rt, shamt op=0x00 funct=0x00**

rd <- rt << shamt (inserting zeroes in empty bits)

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	B = RF[IR[rt]]	REG FILE
EX	Bus_B = B	A-L MUX
	Bus_S = Bus_B SLL by IR[shamt]	A-L SHIFTER
	ALU_out = Bus_S	A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

No Operation (Τύπου R)

- **NOP ≡ SLL $zero, $zero, 0**

Shift Right Logical (Τύπου R)

- **SRL rd, rt, shamt op=0x00 funct=0x02**

rd <- rt >> shamt (inserting zeroes in empty bits)

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	B = RF[IR[rt]]	REG FILE
EX	Bus_B = B	A-L MUX
	Bus_S = Bus_B SRL by IR[shamt]	A-L SHIFTER
	ALU_out = Bus_S	A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Shift Right Arithmetic (Τύπου R)

◆ **SRA rd, rt, shamt op=0x00 funct=0x03**

rd <- rt >> shamt (duplicating bit31 in empty bits)

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	B = RF[IR[rt]]	REG FILE
EX	Bus_B = B	A-L MUX
	Bus_S = Bus_B SRA by IR[shamt]	A-L SHIFTER
	ALU_out = Bus_S	A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Shift Left Logical Variable (Τύπου R)

◆ **SLLV rd, rt, rs op=0x00 funct=0x04**

rd <- rt << rs (inserting zeroes in empty bits)

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	Bus_S = Bus_B SLL by Bus_A	A-L SHIFTER
	ALU_out = Bus_S	A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Shift Right Logical Variable (Τύπου R)

- **SRLV rd, rt, rs op=0x00 funct=0x06**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	Bus_S = Bus_B SRL by Bus_A	A-L SHIFTER
	ALU_out = Bus_S	A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Shift Right Arithmetic Variable (Τύπου R)

- **SRAV rd, rt, rs op=0x00 funct=0x07**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	Bus_S = Bus_B SRA by Bus_A	A-L SHIFTER
	ALU_out = Bus_S	A-L MUX out
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Set Less Than (Signed) Immediate (Τύπου I)

- **SLTI rt, rs, immediate op=0x0A**

rt <- (rs < immediate) set rt=1 if rs < immediate

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	Bus_F = Bus_A - Bus_B (signed) ALU_out = 0..0&msb[Bus_F]	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

msb=1 if A<B

Set Less Than Unsigned Immediate (Τύπου I)

- **SLTIU rt, rs, immediate op=0x0B**

rt <- (rs < immediate) set rt=1 if rs < immediate

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = I	A-L MUX
	Bus_F = Bus_A - Bus_B (unsigned) ALU_out = 0..0&msb[Bus_F]	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για μη προσ/μένους	A-L ALU (optional)
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rt]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Set Less Than (Signed) (Τύπου R)

- **`SLT rd, rs, rt  op=0x00 funct=0x2A`**

  `rd <- (rs < rt) set rd=1 if rs < rt`

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	Bus_F = Bus_A - Bus_B (signed) ALU_out = 0..0&msb[Bus_F]	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για προσημασμένους	A-L ALU
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Set Less Than Unsigned (Τύπου R)

- **`SLTU rd, rs, rt  op=0x00 funct=0x2B`**

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	Bus_F = Bus_A - Bus_B (unsigned) ALU_out = 0..0&msb[Bus_F]	A-L ALU, A-L MUX out
	Έλεγχος υπερχείλισης για μη προσ/σμένους	A-L ALU (optional)
WB	Bus_W = ALU_out	RF MUX
	RF[IR[rd]] = Bus_W	REG FILE
	PC = N	SEL - NPC MUX

Branch on Equal (Τύπου I)

- **BEQ rs, rt, label op=0x04**

If rs = rt BRANCH (16-bit offset || 0^2)= 18 bits, ±128 Kbytes

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B IF ALU_out = 0 THEN Z = 1 ELSE Z = 0	A-L ALU, NOR TREE
	M = N + (I SLL by 2)	SL 2 - ADD
WB	IF Z = 1 THEN PC = M ELSE PC = N	SEL - NPC MUX

ALU: Evaluates condition

Branch on Not Equal (Τύπου I)

- **BNE rs, rt, label op=0x05**

If rs /= rt BRANCH (16-bit offset || 0^2)= 18 bits, ±128 Kbytes

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]]	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B IF ALU_out = 0 THEN Z = 1 ELSE Z = 0	A-L ALU, NOR TREE
	M = N + (I SLL by 2)	SL 2 - ADD
WB	IF Z = 0 THEN PC = M ELSE PC = N	SEL - NPC MUX

Branch on Less than Equal Zero (Τύπου I)

- **BLEZ rs, label op=0x06**

If rs <= 0 BRANCH (16-bit offset || 0^2)= 18 bits, ±128 Kbytes

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]], rt = $zero	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B IF ALU_out = 0 THEN Z = 1 ELSE Z = 0 Ne = msb[ALU_out]	A-L ALU, NOR TREE
	M = N + (I SLL by 2)	SL 2 - ADD
WB	IF Z = 1 or Ne = 1 THEN PC = M ELSE PC = N	SEL - NPC MUX

msb=1 if A<B

Branch on Greater than Zero (Τύπου I)

- **BGTZ rs, label op=0x07**

If rs > 0 BRANCH (16-bit offset || 0^2)= 18 bits, ±128 Kbytes

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]], rt = $zero	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B IF ALU_out = 0 THEN Z = 1 ELSE Z = 0 Ne = msb[ALU_out]	A-L ALU, NOR TREE
	M = N + (I SLL by 2)	SL 2 - ADD
WB	IF Z = 0 and Ne = 0 THEN PC = M ELSE PC = N	SEL - NPC MUX

Branch on Less than Zero (Τύπου I)

- **BLTZ rs, label op=0x01 rt=0x00**

If rs < 0 BRANCH (16-bit offset || 0^2)= 18 bits, ±128 Kbytes

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]], rt = $zero (internally)	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B Ne = msb[ALU_out]	A-L ALU
	M = N + (I SLL by 2)	SL 2 - ADD
WB	IF Ne = 1 THEN PC = M ELSE PC = N	SEL - NPC MUX

msb=1 if A<B

Branch on Greater than Equal Zero (Τύπου I)

- **BGEZ rs, label op=0x01 rt=0x01**

If rs >= 0 BRANCH (16-bit offset || 0^2)= 18 bits, ±128 Kbytes

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]], rt = $zero (internally)	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B Ne = msb[ALU_out]	A-L ALU
	M = N + (I SLL by 2)	SL 2 - ADD
WB	IF Ne = 0 THEN PC = M ELSE PC = N	SEL - NPC MUX

Branch on Less Than Zero And Link (Τύπου I)

- **BLTZAL rs, label op=0x01 rt=0x10**

If rs < 0 BRANCH (16-bit offset || 0^2)= 18 bits, ±128 Kbytes
Place return address on R31

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]], rt = $zero (internally)	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B Ne = msb[ALU_out]	A-L ALU
	M = N + (I SLL by 2)	SL 2 - ADD
WB	Bus_W = N	RF MUX
	RF[IR[rd]] = Bus_W, rd = 31 (internally)	REG FILE
	IF Ne = 1 THEN PC = M ELSE PC = N	SEL - NPC MUX

Branch on Greater Than Equal Zero And Link

- **BGEZAL rs, label op=0x01 rt=0x11**

If rs >= 0 BRANCH (16-bit offset || 0^2)= 18 bits, ±128 Kbytes
Place return address on R31

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
	B = RF[IR[rt]], rt = $zero (internally)	REG FILE
	I = Sign_ext[IR[imm]]	EXT S/Z
EX	Bus_A = A	A-L MUX
	Bus_B = B	A-L MUX
	ALU_out = Bus_F = Bus_A - Bus_B Ne = msb[ALU_out]	A-L ALU
	M = N + (I SLL by 2)	SL 2 - ADD
WB	Bus_W = N	RF MUX
	RF[IR[rd]] = Bus_W, rd = 31 (internally)	REG FILE
	IF Ne = 0 THEN PC = M ELSE PC = N	SEL - NPC MUX

Jump Register (Τύπου R)

- **JR rs** **op=0x00 funct=0x08**

PC <- rs

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
WB	PC = A	SEL - NPC MUX

Jump And Link Register (Τύπου R)

- **JALR {rd,} rs** **op=0x00 funct=0x09**

optional

PC <- rs,
Place return address on R31

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
ID	A = RF[IR[rs]]	REG FILE
WB	PC = A	SEL - NPC MUX
	Bus_W = N	RF MUX
	RF[IR[rd]] = Bus_W, by default rd = 31*	REG FILE

Εάν δεν ορίζεται διαφορετικά στο πρόγραμμα, ο assembler θέτει σαν default τιμή το 31

Jump (Τύπου J)

- **J target op=0x02**

PC region branch (NOT PC relative)
Jump to (26-bit target || 0^2)= 28 bits
(Jumps target first 256 Mbytes memory region)

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
	P = PC (4 msb)	
ID	D = P&IR[addr]]&00	PSD
WB	PC = D	SEL - NPC MUX

Jump And Link (Τύπου J)

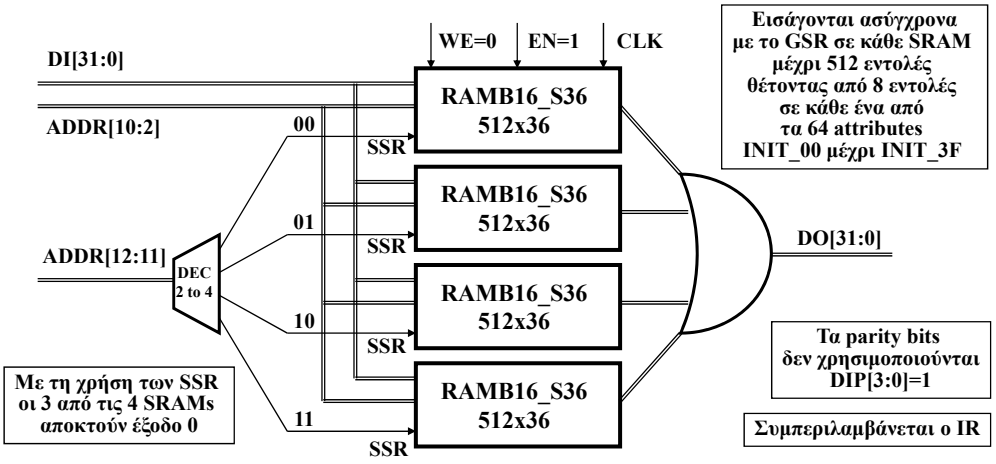
- **JAL target op=0x03**

PC region branch (NOT PC relative)
Jump to (26-bit target || 0^2)= 28 bits, 256 Mbytes region
Place return address on R31

Φάσεις	Μικρο-λειτουργίες	Μονάδες
IF	IR = IM[PC]	IM SRAM
	N = PC+4	INC
	P = PC (4 msb)	
ID	D = P&IR[addr]]&00	PSD
WB	PC = D	SEL - NPC MUX
	Bus_W = N	RF MUX
	RF[IR[rd]] = Bus_W, rd = 31 (internally)	REG FILE

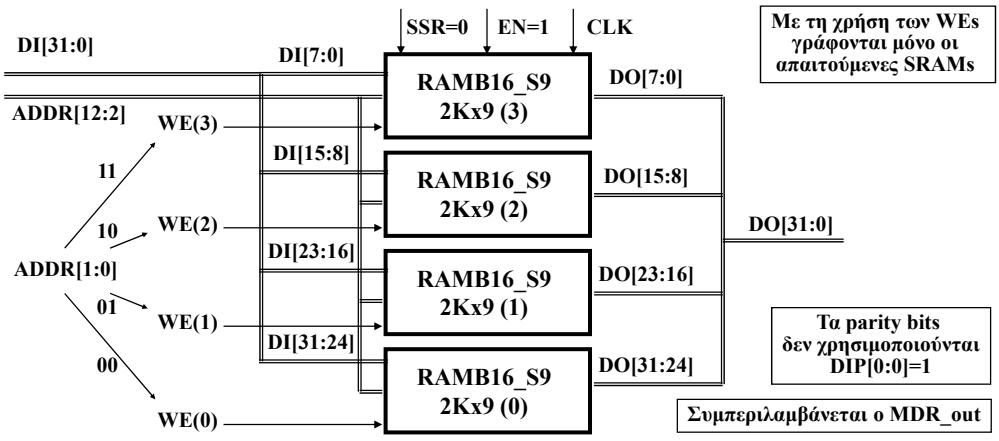
Υλοποίηση της Μνήμης Εντολών (2Kx36)

- **Η υλοποίηση θα γίνει με 4 single-port Block RAMs και υποθέτοντας ότι θα χρησιμοποιηθεί ένα Spartan-3 FPGA**
  - **Το XC3S400 διαθέτει 2x8=16 Block RAMs των 18Kbits**
  - **Η υλοποίηση θα γίνει σύμφωνα με το XAPP463**

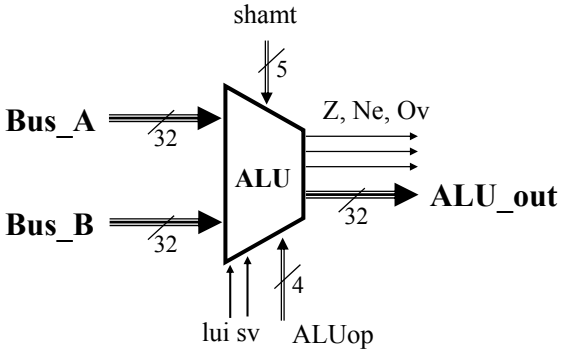


Υλοποίηση της Μνήμης Δεδομένων (2Kx36)

- **Η υλοποίηση θα γίνει με 4 single-port Block RAMs και υποθέτοντας ότι θα χρησιμοποιηθεί ένα Spartan-3 FPGA**
  - **Το XC3S400 διαθέτει 2x8=16 Block RAMs των 18Kbits**
  - **Η υλοποίηση θα γίνει σύμφωνα με το XAPP463**

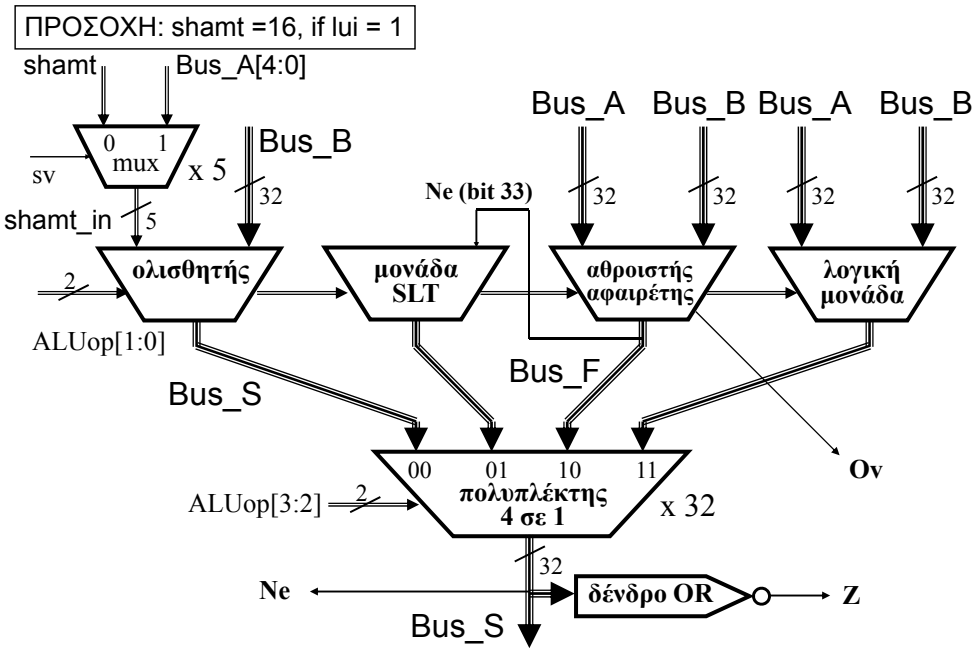


Αριθμητική και Λογική Μονάδα (ALU)



Bus_A, Bus_B, ALU_out : οι δύο είσοδοι και η έξοδος της ALU μεγέθους 32 ψηφίων
shamt : η αρτηρία μεγέθους 5 ψηφίων που προσδιορίζει το σταθερό αριθμό ολισθήσεων (shamt = 16, όταν lui = 1)
ALUop : το σήμα ελέγχου μεγέθους 4 ψηφίων που επιλέγει την πράξη
sv : το σήμα ελέγχου που μας πληροφορεί ότι ο αριθμός των ολισθήσεων είναι μεταβλητός
Z, Ne, Ov : τα σήματα κατάστασης που μας πληροφορούν ότι η έξοδος της ALU (ALU_out) είναι μηδέν, αρνητική ή υπάρχει υπερχείλιση

Αριθμητική και Λογική Μονάδα (ALU)



Σήματα Ελέγχου της ALU για Εντολές Τύπου R

εντολή	πεδίο funct		ALUop	sv
ADD	32	1000 00	10 00	X
ADDU	33	1000 01	10 01	X
SUB	34	1000 00	10 10	X
SUBU	35	1000 11	10 11	X
AND	36	1001 00	11 00	X
OR	37	1001 01	11 01	X
XOR	38	1001 10	11 10	X
NOR	39	1001 11	11 11	X
SLL	0	0000 00	00 00	0
SRL	2	0000 10	00 10	0
SRA	3	0000 11	00 11	0
SLLV	4	0001 00	00 00	1
SRLV	6	0001 10	00 10	1
SRAV	7	0001 11	00 11	1
SLT	42	1010 10	01 10	X
SLTU	43	1010 11	01 11	X

ALUop[1] = sub/add'
ALUop[0] = unsign/sign'

ALUop[1:0] = 00 AND
ALUop[1:0] = 01 OR
ALUop[1:0] = 10 XOR
ALUop[1:0] = 11 NOR

ALUop[1:0] = 00 SLL
ALUop[1:0] = 10 SRL
ALUop[1:0] = 11 SRA

Σήματα Ελέγχου της ALU για Εντολές Τύπου I

εντολή	πεδίο opcode		πεδίο rt	ALUop
ADDI	8	001000		10 00
ADDIU	9	001001		10 01
ANDI	12	001100		11 00
ORI	13	001101		11 01
XORI	14	001110		11 10
LUI	15	001111		00 00
LW	35	100011		10 00
SW	43	101011		10 00
BEQ	4	000100		10 10
BNE	5	000101		10 10
SLTI	10	001010		01 10
SLTIU	11	001011		01 11
BLEZ	6	000110		10 10
BGTZ	7	000111		10 10
BLTZ	1	000001	00000	10 10
BGEZ	1	000001	00001	10 10
BLTZAL	1	000001	10000	10 10
BGEZAL	1	000001	10001	10 10