UNIVERSITY OF
PATRAS
ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΑΤΡΩΝ

Πολυτεχνική Σχολή

Τμήμα Μηχανικών Η/Υ & Πληροφορικής

Διπλωματική Εργασία

Τίτλος

Γιαννάκης Εμμανουήλ Δημήτριος

Α.Μ. : 1067491

Επιβλέπων

Νικολός Δημήτριος, Καθηγητής

Μέλη Εξεταστικής Επιτροπής

Βέργος Χαρίδημος, Καθηγητής

Ευχαριστίες

Περίληψη

Abstract

ΠΕΡΙΕΧΟΜΕΝΑ

1. ΕΙΣΑΓΩΓΗ

1.1 Ο επεξεργαστής

Οι επεξεργαστές αποτελούν τον πυρήνα των υπολογιστικών συστημάτων, εκτελούν ποικίλες εργασίες με μεγάλη ταχύτητα και αποτελεσματικότητα. Αναλαμβάνουν την επεξεργασία δεδομένων, την εκτέλεση εντολών και τον συντονισμό των λειτουργιών του υπολογιστικού συστήματος. Οι εξελίξεις στην τεχνολογία των επεξεργαστών έχουν οδηγήσει σε συνεχή αύξηση της ταχύτητας, της απόδοσης και της ενεργειακής αποδοτικότητας. Οι επεξεργαστές είναι κατασκευασμένοι με βάση διάφορες αρχιτεκτονικές.

1.2 Αρχιτεκτονική συνόλου εντολών

Η αρχιτεκτονική συνόλου εντολών (Instruction Set Architecture - ISA) είναι μέρος του abstract μοντέλου ενός υπολογιστή που καθορίζει τον τρόπο με τον οποίο η CPU ελέγχεται από το λογισμικό. Η ISA αποτελεί τη σύνδεση μεταξύ του υλικού και του λογισμικού, καθορίζοντας το τι είναι ικανός να κάνει ο επεξεργαστής, αλλά και τον τρόπο με τον οποίο γίνεται αυτό. Μπορεί να θεωρηθεί ως το εγχειρίδιο του προγραμματιστή, επειδή είναι το τμήμα της μηχανής που είναι ορατό στον προγραμματιστή. Η ISA ορίζει τους υποστηριζόμενους τύπους δεδομένων, τους καταχωρητές, τον τρόπο με τον οποίο το υλικό διαχειρίζεται την μνήμη, ποιες εντολές μπορεί να εκτελέσει ένας επεξεργαστής. Τα αποτελέσματα της εκτέλεσης των εντολών μπορούν να είναι αριθμητικά αποτελέσματα, αλλαγή της ροής του προγράμματος με βάση μια συνθήκη ή κατάσταση, αλλαγή της κατάστασης που βρίσκεται ο επεξεργαστής. Οι κατηγορίες για τις αρχιτεκτονικές συνόλου εντολών είναι οι εξής:

1. Reduced Instruction Set Computer (RISC)
   Το πλήθος των εντολών είναι περιορισμένο και οι εντολές αυτές εκτελούν βασικές λειτουργίες. Κάθε εντολή εκτελεί μια απλή λειτουργία γενικού σκοπού. Υπάρχουν ξεχωριστές εντολές για την επεξεργασία δεδομένων και την προσπέλαση της μνήμης.

2. Complex Instruction Set Computer (CISC)
   Το πλήθος των εντολών είναι μεγάλο και οι εντολές αυτές εκτελούν λειτουργίες ειδικού σκοπού. Κάθε εντολή εκτελεί μια εξειδικευμένη λειτουργία.

1.3 Αρχιτεκτονική RISC-V

Η αρχιτεκτονική RISC-V ανήκει στην αρχιτεκτονική RISC και είναι η πρώτη open-source αρχιτεκτονική συνόλου εντολών. Δημιουργήθηκε το 2010 στο Πανεπιστήμιο της Καλιφόρνια, Berkeley από τους Krste Asanović, Andrew Waterman, David Patterson και άλλους. Από τη δημιουργία της έως και σήμερα πολλά άτομα έχουν συμβάλει στην ανάπτυξή της [1]. Για τον σχεδιασμό της RISC-V δόθηκε έμφαση στην απλότητα και στην επεκτασιμότητα. Όσον αφορά την απλότητα, η RISC-V στη βάση της, ορίζεται ως μια μικρή αρχιτεκτονική συνόλου εντολών που αφορά ακέραιους αριθμούς και οι εντολές αυτές πρέπει να είναι παρούσες σε όλες τις υλοποιήσεις. Κάθε σύνολο εντολών για ακέραιους αριθμούς χαρακτηρίζεται από το μήκος των καταχωρητών και το μήκος των διευθύνσεων. Οι δύο βασικές παραλλαγές της RISC-V ISA είναι οι RV32I και RV64I οι οποίες αναφέρονται σε μήκος διεύθυνσης 32-bit και 64-bit αντίστοιχα. Όσον αφορά την επεκτασιμότητα, υπάρχουν επεκτάσεις (“extensions”) του συνόλου εντολών, όπως για παράδειγμα η «F» η οποία αφορά πράξεις με αριθμούς κινητής υποδιαστολής. Οι επεκτάσεις χωρίζονται σε standard και non-standard. Οι standard θεωρούνται εκείνες που είναι γενικά χρήσιμες και είναι σχεδιασμένες έτσι ώστε να διαχωρίζονται με σαφήνεια από τις υπόλοιπες standard. Οι non-standard θεωρούνται εκείνες που είναι υψηλά εξιδεικευμένες

και ίσως να μην διαχωρίζονται με σαφήνεια από τις υπόλοιπες standard και non-standard [2].

RV32I/RV32E/RV64I/RV128I Base Integer Instruction Set

Το βασικό σύνολο εντολών για 32, 64, 128 bit αντίστοιχα. Περιλαμβάνει εντολές για αριθμητικές και λογικές πράξεις ακέραιων αριθμών, εντολές για ανάγνωση και προσπέλαση μνήμης.

"M" extension Multiplication - Division

Το σύνολο αυτό περιλαμβάνει εντολές για πράξεις πολλαπλασιασμού και διαίρεσης. Γίνεται διαχωρισμός από το Base Integer Instruction Set για να απλουστευθούν οι low-end υλοποιήσεις που δεν προβλέπουν τέτοιες πράξεις.

"A" extension Atomic Instructions

Το σύνολο αυτό περιλαμβάνει εντολές για ανάγνωση, τροποποίηση, εγγραφή της μνήμης για τον συγχρονισμό των πυρήνων ενός RISC-V επεξεργαστή που μοιράζονται την ίδια μνήμη.

"F" extension Single-Precision Floating-Point

Το σύνολο αυτό περιλαμβάνει εντολές για πράξεις με αριθμούς κινητής υποδιαστολής. Η επέκταση αυτή είναι συμβατή με το αριθμητικό πρότυπο IEEE 754-2008. Γίνεται διαχωρισμός από το Base Integer Instruction Set για να απλουστευθούν οι low-end υλοποιήσεις που δεν προβλέπουν τέτοιες πράξεις.

"D" extension Double-Precision Floating-Point

Το σύνολο αυτό περιλαμβάνει εντολές για πράξεις με αριθμούς κινητής υποδιαστολής διπλής ακρίβειας. Βασίζεται στην επέκταση «F».

"Q" extension Quad-Precision Floating-Point

Το σύνολο αυτό περιλαμβάνει εντολές για πράξεις με αριθμούς κινητής υποδιαστολής τετραπλής ακρίβειας. Βασίζεται στις επεκτάσεις «F» και «D».

"C" extension Compressed Instructions

Το σύνολο αυτό περιλαμβάνει συμπιεσμένες εντολές κωδικοποιημένες στα 16-bit οι οποίες αφορούν βασικές λειτουργίες. Αυτό έχει ως αποτέλεσμα τη μείωση του μεγέθους του κώδικα.

2. ΣΥΝΟΛΟ ΕΝΤΟΛΩΝ RV32I

Το σύνολο RV32I είναι το βασικό σύνολο εντολών της αρχιτεκτονικής RISC-V. Παρακάτω θα αναλυθούν όλες οι εντολές του συνόλου, καθώς και τύποι δυαδικής κωδικοποίησης αυτών των εντολών.

2.1 Τύποι κωδικοποίησης εντολών

Στο σύνολο εντολών RV32I εμφανίζονται έξι διαφορετικοί τύποι κωδικοποίησης και καθένας προορίζεται για διαφορετική ομάδα εντολών. Πιο αναλυτικά:

R-type:

31 25	24 20	19 15	14 12	11 7	6 0
funct7	rs2	rs1	funct3	rd	opcode

Σε αυτό τον τύπο κωδικοποίησης αξιοποιούνται τρία πεδία καταχωρητών rs1, rs2 και rd. Παρατηρούνται επίσης και τα πεδία funct7 και funct3. Η ομάδα εντολών στην οποία συναντάται αυτός ο τύπος είναι οι εντολές πράξεων μεταξύ δυο καταχωρητών (Register – Register).

I-type:

31 20	19 15	14 12	11 7	6 0
imm[11:0]	rs1	funct3	rd	opcode

Σε αυτό τον τύπο κωδικοποίησης αξιοποιούνται δύο πεδία καταχωρητών rs1 και rd. Παρατηρούνται επίσης το πεδίο άμεσου δεδομένου imm και το

funct3. Η ομάδα εντολών στην οποία συναντάται αυτός ο τύπος είναι οι εντολές πράξεων μεταξύ ενός καταχωρητή κι ενός άμεσου δεδομένου (Register - Immediate) καθώς και από τις εντολές ανάγνωσης της μνήμης δεδομένων (Load).

S-type:

31 25	24 20	19 15	14 12	11 7	6 0
imm[11:5]	rs2	rs1	funct3	imm[4:0]	opcode

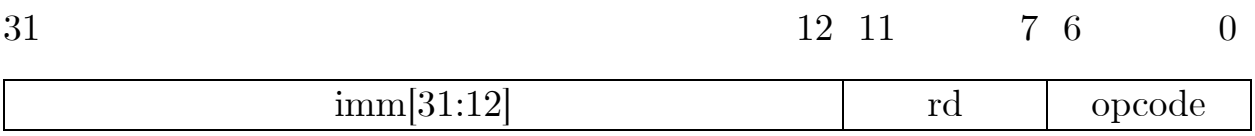
Σε αυτό τον τύπο κωδικοποίησης αξιοποιούνται δύο πεδία καταχωρητών rs1 και rs2. Παρατηρούνται επίσης δύο πεδία για το άμεσο δεδομένο imm και το funct3. Η ομάδα εντολών στην οποία συναντάται αυτός ο τύπος είναι οι εντολές εγγραφής στην μνήμη δεδομένων (Store).

B-type:

31 25	24 20	19 15	14 12	11 7	6 0
imm[12|10:5]	rs2	rs1	funct3	imm[4:1|11]	opcode

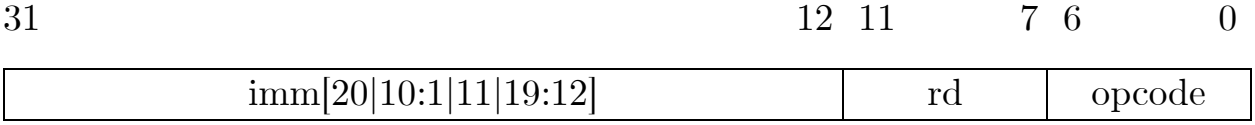
Σε αυτό τον τύπο κωδικοποίησης αξιοποιούνται δύο πεδία καταχωρητών rs1 και rs2. Παρατηρούνται επίσης δύο πεδία για το άμεσο δεδομένο imm και το funct3. Η ομάδα εντολών στην οποία συναντάται αυτός ο τύπος είναι οι εντολές διακλάδωσης υπό συνθήκη (Branch).

U-type:



Σε αυτό τον τύπο κωδικοποίησης αξιοποιείται ένα πεδίο για τον καταχωρητή rd. Παρατηρείται επίσης ένα πεδίο για το άμεσο δεδομένο imm. Η ομάδα εντολών στην οποία συναντάται αυτός ο τύπος είναι οι εντολές LUI και AUIPC.

J-type:



Σε αυτό τον τύπο κωδικοποίησης αξιοποιείται ένα πεδίο για τον καταχωρητή rd. Παρατηρείται επίσης ένα πεδίο για το άμεσο δεδομένο imm. Η ομάδα εντολών στην οποία συναντάται αυτός ο τύπος είναι οι εντολές άλματος (Unconditional Jumps).

Παρατηρείται πως σε καθεμιά κατηγορία υπάρχει διαφορετικός αριθμός πεδίων που έχουν διάφορες χρήσεις. Όμως, τα κοινά πεδία μεταξύ των εντολών είναι στη ίδια θέση, αυτό το χαρακτηριστικό προσφέρει απλότητα στον σχεδιασμό και κάνει την παραγωγή των δυαδικών κωδικοποιήσεων των εντολών πιο εύκολη [3]. Παρακάτω ακολουθεί πίνακας που επεξηγεί κάθε ξεχωριστό πεδίο από τους παραπάνω τύπους κωδικοποίησης.

Όνομα	Μέγεθος	Χρήση	Θέση
rs1	5 bit	Ο πρώτος καταχωρητής που θα γίνει ανάγνωσή του από το αρχείο καταχωρητών	19 – 15 bits
rs2	5 bit	Ο δεύτερος καταχωρητής που θα γίνει ανάγνωσή του από το αρχείο καταχωρητών	24 – 20 bits
rd	5 bit	Ο καταχωρητής στον οποίο αποθηκεύεται το αποτέλεσμα	11 – 7 bits
opcode	7 bit	Ο μοναδικός κωδικός εκτέλεσης για κάθε ομάδα εντολών	0 – 6 bits
funct3	3 bit	Εξειδικεύει περισσότερο τη λειτουργία της εντολής	14 – 12 bits
funct7	7 bit	Εξειδικεύει περισσότερο τη λειτουργία της εντολής	31 – 25 bits
imm	Ανάλογα με τον τύπο κωδικοποίησης	Άμεσο δεδομένο που χρησιμοποιείται στην εκτέλεση της εντολής	Ανάλογα με τον τύπο κωδικοποίησης
shamt*	5 bit	Πλήθος θέσεων ολίσθησης στις εντολές ολίσθησης με άμεσο δεδομένο.	24 – 20 bits

Πίνακας 1: Πεδία της κωδικοποίησης των εντολών του συνόλου RV32I

* Το πεδίο αυτό δεν εμφανίζεται στα σχήματα των κωδικοποιήσεων που παρουσιάστηκαν παραπάνω. Όμως, είναι παρών στις εντολές SLLI, SRLI, SRAI οι οποίες θα αναλυθούν σε επόμενη υποενότητα της Ενότητας 2.

2.2 Εντολές αριθμητικών και λογικών πράξεων

2.2.1 Εντολές για πράξεις μεταξύ δυο καταχωρητών

Η κατηγορία αυτών των εντολών περιλαμβάνει όλες τις εντολές οι οποίες εκτελούν μια μαθηματική ή λογική πράξη μεταξύ των δεδομένων που περιέχουν δυο καταχωρητές (Register - Register). Ακολουθεί πίνακας με αυτές τις εντολές:

Διευκρίνηση: όπου εμφανίζεται ο συμβολισμός c(x0), συμβολίζει το περιεχόμενο του καταχωρητή x0. Το ίδιο ισχύει και για του υπόλοιπους καταχωρητές.

Εντολή	Πράξη	Λειτουργία
ADD x0, x1, x2	x0 ‹– x1 + x2	Αριθμητική πρόσθεση των c(x1), c(x2). Αποθήκευση του αποτελέσματος στον x0.
SUB x0, x1, x2	x0 ‹– x1 - x2	Αριθμητική αφαίρεση των c(x1), c(x2). Αποθήκευση του αποτελέσματος στον x0.
SLL x0, x1, x2	x0 ‹– x1 << x2[4:0]	Λογική ολίσθηση του c(x1) κατά X θέσεις αριστερά. Η τιμή X ισούται με τα 5 λιγότερο σημαντικά bit της τιμής c(x2). Αποθήκευση του αποτελέσματος στον καταχωρητή x0.
SLT x0, x1, x2	(x1 < x2) ? x0 ‹– 1 : 0	Το c(x0) παίρνει την τιμή 1 αν c(x1) < c(x2), αλλιώς παίρνει την τιμή 0.
SLTU x0, x1, x2	(x1 < x2) ? x0 ‹– 1 : 0	Ίδια λειτουργία με SLT. Οι αριθμοί θεωρούνται μη προσημασμένοι.
XOR x0, x1, x2	x0 ‹– x1 ^ x2	Λογική πράξη XOR ανά bit μεταξύ των c(x1), c(x2).

SRL x0, x1, x2	x0 ‹– x1 >> x2[4:0]	Λογική ολίσθηση του c(x1) κατά X θέσεις δεξιά. Η τιμή X ισούται με τα 5 λιγότερο σημαντικά bit της τιμής c(x2). Αποθήκευση του αποτελέσματος στον x0.
SRA x0, x1, x2	x0 ‹– x1 >>> x2[4:0]	Αριθμητική ολίσθηση του c(x1) κατά X θέσεις δεξιά. Η τιμή X ορίζεται ως τα 5 λιγότερο σημαντικά bit της τιμής c(x2). Αποθήκευση του αποτελέσματος στον x0.
OR x0, x1, x2	x0 ‹– x1 | x2	Λογική πράξη OR ανά bit μεταξύ των c(x1), c(x2).
AND x0, x1, x2	x0 ‹– x1 & x2	Λογική πράξη AND ανά bit μεταξύ των c(x1), c(x2).

Πίνακας 2: Εντολές Register - Register

2.2.2 Εντολές για πράξεις μεταξύ καταχωρητή – άμεσου δεδομένου

Η κατηγορία αυτών των εντολών περιλαμβάνει όλες τις εντολές οι οποίες εκτελούν μια μαθηματική ή λογική πράξη μεταξύ του περιεχομένου ενός καταχωρητή και ενός άμεσου δεδομένου (Register - Immediate). Ακολουθεί πίνακας με αυτές τις εντολές:

Διευκρίνηση: όπου εμφανίζεται ο συμβολισμός c(x0), συμβολίζει το περιεχόμενο του καταχωρητή x0. Το ίδιο ισχύει και για του υπόλοιπους καταχωρητές.

Εντολή	Πράξη	Λειτουργία
ADDI x0, x1, imm	x0 ‹– x1 + x2	Αριθμητική πρόσθεση των c(x1), imm. Αποθήκευση του αποτελέσματος στον x0.
SLTI x0, x1, imm	(x1 < imm) ? x0 ‹– 1 : 0	Το c(x0) παίρνει την τιμή 1 αν c(x1) < imm, αλλιώς παίρνει την τιμή 0.
SLTIU x0, x1, imm	(x1 < imm) ? x0 ‹– 1 : 0	Ίδια λειτουργία με SLTI. Οι αριθμοί θεωρούνται μη προσημασμένοι.
XORI x0, x1, imm	x0 ‹– x1 ^ imm	Λογική πράξη XOR ανά bit μεταξύ των c(x1), imm.
ORI x0, x1, imm	x0 ‹– x1 ^ imm	Λογική πράξη OR ανά bit μεταξύ των c(x1), imm.
ANDI x0, x1, imm	x0 ‹– x1 ^ imm	Λογική πράξη AND ανά bit μεταξύ των c(x1), imm.
SLLI x0, x1, imm	x0 ‹– x1 << imm[4:0]	Λογική ολίσθηση του c(x1) κατά X θέσεις αριστερά. Η τιμή X ισούται με τα 5 λιγότερο σημαντικά bit του imm. Αποθήκευση του αποτελέσματος στον καταχωρητή x0.
SRLI x0, x1, imm	x0 ‹– x1 >> imm[4:0]	Λογική ολίσθηση του c(x1) κατά X θέσεις δεξιά. Η τιμή X ισούται με τα 5 λιγότερο σημαντικά bit του imm. Αποθήκευση του αποτελέσματος στον καταχωρητή x0.

SRAI x0, x1, imm	x0 ‹– x1 >>> imm[4:0]	Αριθμητική ολίσθηση του c(x1) κατά X θέσεις δεξιά. Η τιμή X ορίζεται ως τα 5 λιγότερο σημαντικά bit της τιμής imm. Αποθήκευση του αποτελέσματος στον x0.

6. ΒΙΒΛΙΟΓΡΑΦΙΑ

[1] Harris, S., & Harris, D. (2021). *Digital design and computer architecture: RISC-V Edition*. Morgan Kaufmann.

[2] "The RISC-V Instruction Set Manual, Volume I: User-Level ISA, Document Version 2.2", Editors Andrew Waterman and Krste Asanovi´c, RISC-V Foundation, May 2017.

[3] Patterson, D. A., & Hennessy, J. L. (n.d.). *Computer Organization and Design RISC-V Edition: The Hardware Software Interface*. Morgan Kaufmann.