HY-225: Οργάνωση Υπολογιστών
Άνοιξη 2012

Τμ. Επ. Υπολογιστών
© Πανεπιστήμιο Κρήτης

Άσκηση 4:
Γλώσσα Μηχανής, Format Εντολών

Παράδοση: Τετάρτη 7 Μαρτίου 2012 (βδ. 4.2) σε χαρτί, στο μάθημα (από βδ. 2.2)

4.1 Περίληψη Θεμάτων Γλώσσας Μηχανής του MIPS:

Εσωτερικά, ο υπολογιστής λειτουργεί μόνο με δυαδικές τιμές και σύμβολα. Έτσι, γιά να μπορέσει να εκτελεστεί ένα πρόγραμμα Assembly πρέπει αυτό να μεταφραστεί σε *Γλώσσα Μηχανής*, δηλ. σε δυαδικά σύμβολα. Τη μετάφραση αυτή κάνει ένα πρόγραμμα, ο *Assembler*.

Όλες οι εντολές του MIPS, όπως και των άλλων υπολογιστών στυλ RISC, έχουν σταθερό μέγεθος 32 bits. Μέσα τους περιέχουν από 2 έως 6 πεδία (fields). Υπάρχουν 3 διαφορετικές μορφές που οι εντολές μπορεί να έχουν. Η μορφή (format) της κάθε εντολής καθορίζεται από το πρώτο πεδίο της εντολής, που ονομάζεται **`op`**, έχει μέγεθος 6 bits, και είναι πάντα στην ίδια θέση (MS bits) γιά όλες τις εντολές. Φυσικά, με τα 6 bits που έχει, το πεδίο αυτό καθορίζει, εκτός από το format της εντολής, και ένα μέρος ή και ολόκληρο το **`opcode`** της εντολής, δηλαδή το τι πράξη ή ενέργεια πρέπει να εκτελέσει αυτή η εντολή. Οι τρείς μορφές εντολών είναι:

R-format:
: Αυτό φαίνεται στη σελίδα 137 του βιβλίου (Α' τόμος Ελληνικής μετάφρασης από 4η Αμ. έκδοση), και αποτελείται, μετά το πεδίο **`op`**, από τρία πεδία των 5 bits καθένα που επιλέγουν έναν από τους 32 καταχωρητές καθένα, ένα πεδίο των 5 bits που δεν θα το χρησιμοποιήσουμε εμείς στο υποσύνολο των εντολών που θα μελετήσουμε, και ένα πεδίο μεγέθους 6 bits που αποτελεί επέκταση του **`op`** και λέγεται **`funct`** (function code).

I-format:
: Αυτό φαίνεται στη σελίδα 138 του βιβλίου (Α' τόμος, 4η εκδ.), και αποτελείται, μετά το πεδίο **`op`**, από δύο πεδία των 5 bits καθένα που επιλέγουν έναν από τους 32 καταχωρητές καθένα, και ένα πεδίο μεγέθους 16 bits που λέγεται **`imm`** (immediate, ή offset, ή constant, ή address) και που περιέχει μία "άμεση" σταθερή ποσότητα.

J-format:
: Αυτό φαίνεται στη σελίδα 171 του βιβλίου (Α' τόμος, 4η εκδ.), και περίεχει, μετά το πεδίο **`op`**, ένα πεδίο "**`target`**" μεγέθους 26 bits που προσδιορίζει μεγάλο μέρος μιας πλήρους διεύθυνσης μνήμης.

Οι εντολές του MIPS, το format τους, και τα opcodes τους φαίνονται και στις σελίδες Α-49 έως Α-70 του παραρτήματος Α του βιβλίου, από παλαιότερη Αμερικανική έκδοση, που υπάρχει στο αρχείο: **`~hy225/spim/documentation/HP_AppA.pdf`** Ο τρόπος που το format και το opcode καθορίζονται από τα πεδία `op` και `funct` με μια κωδικοποίηση μεταβλητού μεγέθους αποτελεί αντικείμενο της άσκησης 5.2.

Άσκηση 4.2: Γλώσσα Μηχανής και Κώδικες Μεταβλητού Μήκους

Σ' αυτή την άσκηση θα μελετήσετε τον τρόπο που το format και το opcode στον MIPS καθορίζονται από τα πεδία `op` και `funct` με μια κωδικοποίηση μεταβλητού μεγέθους. Γιά να μην παιδευόμαστε όμως με μεγάλο πλήθος εντολών, θα χρησιμοποιήσουμε ένα

δικό μας, φανταστικό format εντολών, ενός φανταστικού υπολογιστή, του MIPS_8. Ολες οι εντολές του MIPS_8 έχουν μέγεθος 8 bits. Ο MIPS_8 έχει μόνο 8 καταχωρητές, τους $0, $1, ..., $7, και οι σταθερές του ποσότητες (immediates) μπορούν να είναι μόνο οι 32 μη-αρνητικοί ακέραιοι 0, 1, 2, ..., 31. Οι εντολές του MIPS_8 έχουν μόνον έναν τελεστέο --είτε καταχωρητή είτε σταθερή ποσότητα immediate-- και έχουν μόνο δύο format, τα εξής:

R-format:

- **`op`** (3 MS bits): πρώτο μέρος του opcode,
- **`R`** (3 επόμενα bits): ο καταχωρητής - τελεστέος,
- **`funct`** (2 LS bits): δεύτερο μέρος του opcode.

I-format:

- **`op`** (3 MS bits): ο (μοναδικός) opcode,
- **`Imm`** (5 LS bits): η σταθερά - τελεστέος.

4(a): Εστω ότι "ξοδεύουμε" και τους οκτώ (8) διαθέσιμους συνδυασμούς του πεδίου op γιά 8 εντολές I-format, τις εντολές **`ki, li, mi, ni, pi, qi, ri, si`**. Σ' αυτή την περίπτωση, μπορούμε να έχουμε καμία εντολή R-format; Γιατί όχι; Έστω πως επιμέναμε να έχουμε την εντολή R-format "**`LL`**" με `op=001` και `funct=10`. Τότε, η εντολή "**`LL`** `$5`" με ποιάν άλλη εντολή I-format (και με τι τελεστέο) θα ήταν ίδια κι απαράλλακτη, με συνέπεια να μη μπορούμε να τις έχουμε και τις δύο στον MIPS_8; Αποδείξτε εν συντομία αλλά με "μαθηματική" ακρίβεια και σαφήνεια ότι το ίδιο θα ίσχυε γιά οιαδήποτε άλλη δυνατή εντολή R-format.

4(b): Εστω τώρα ότι "ξοδεύουμε" μόνο τους 7 από τους 8 διαθέσιμους συνδυασμούς του πεδίου op --τους 000, 001, 010, 011, 100, 101, και 110-- γιά 7 εντολές I-format --τις **`ki, li, mi, ni, pi, qi, ri`**. Σ' αυτή την περίπτωση, πόσες εντολές R-format μπορούμε να έχουμε; Τι κώδικες **`op`** και **`funct`** θα έχει καθεμιά τους; Γιατί δεν μπορούμε να έχουμε περισσότερες από τόσες εντολές R-format;

4(c): Γιά την περίπτωση 4(b), σχεδιάστε ένα συνδυαστικό κύκλωμα, χρησιμοποιώντας πύλες AND, OR, NOT (μόνο τέτοιες, και όχι έτοιμους αποκωδικοποιητές), που να δέχεται σαν είσοδο μιαν εντολή (8 bits) και να παράγει σαν εξόδους:

- ένα σήμα I που ανάβει όταν και μόνον όταν η εντολή είναι I-format,
- ένα σήμα R που ανάβει όταν και μόνον όταν η εντολή είναι R-format,
- 7 σήματα, ένα γιά καθε εντολή I-format, που να ανάβει όταν και μόνον όταν βλέπει τη συγκεκριμένη εντολή, και
- ένα σήμα γιά καθε εντολή R-format, που να ανάβει όταν και μόνον όταν βλέπει τη συγκεκριμένη εντολή.

4(d): Ανάλογη ερώτηση με την 4(b), αλλά έστω ότι τώρα έχουμε μόνο 6 εντολές I-format. Πόσες εντολές R-format μπορούμε να έχουμε; Γράψτε τα opcodes όλων των εντολών, και των δύο format. Προσπαθήστε να επιλέξτε τα opcodes έτσι ώστε να απλοποιείται η αποκωδικοποίηση των σημάτων I και R (κατ' αναλογία προς την ερώτηση 4(c)).
