Αρχιτεκτονική Η/Υ - (ΜΗΥΠ 323)

Φθινοπωρινό Εξάμηνο 2013

Τμήμα Ηλεκτρολόγων Μηχανικών και

Μηχανικών Υπολογιστών και Πληροφορικής

Τεχνολογικό Πανεπιστήμιο Κύπρου

1ο Millstone Report 5ου εξαμήνου 2013

Ημερ. Παράδοσης : 15/10/2013

Διδάσκων : Δρ. Βάσος Σωτηρίου

Ονόματα Φοιτητών : Μιχαήλ Παναγιώτου

Βαρνάβας Ξυδάς

Abstract:

Σε αυτή την αναφορά, αρχικά, θα αναλύσουμε το πώς λειτουργεί η πρώτη φάση του προγράμματος που υλοποιήσαμε με βάση τις προδιαγραφές και τις οδηγίες που δόθηκαν από τον διδάσκων του μαθήματος. Ακολούθως θα εξηγήσουμε το πώς μοιράσαμε το φόρτο εργασίας έτσι ώστε να έχουμε καλύτερη και ταχύτερη υλοποίηση εντός του χρονικού πλαισίου παράδοσης. Στο τέλος, θα αναφερθούμε σε δυσκολίες που συναντήσαμε κατά την υλοποίηση του προγράμματος και τρόπους που επινοηθήκαμε για να λύσουμε αυτά τα προβλήματα.

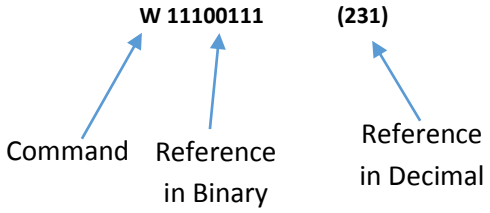
Introduction:

Για σκοπούς ευκολότερης ανάλυσης θα σπάσουμε το πρόγραμμα μας σε 6 κύρια μέρη:

1. *Data File Generator*
2. *Binary to Decimal Function*
3. *Data File Parser*
4. *Direct Mapped Function*
5. *Fully Associative Function*
6. *N-Ways Associative Function*

Data File Generator:

Example of data that has been generated:



Υλοποιήσαμε την γεννήτρια αναφορών μέσα στην main() συνάρτηση. Η γεννήτρια παράγει ένα αριθμό αναφορών ίσο με την μέγιστη διεύθυνση που μπορεί να δεχθεί η κύρια μνήμη (π.χ Κύρια μνήμη έχει διευθυνσιοποίηση 0 – 63 => η μέγιστη διεύθυνση είναι η 63) επί ένα αριθμό που ορίζεται από τον χρήστη (NUM_MUL).

```
for (i = 0; i <(NUM_MUL*ram_size); i++)
```

Για να μπορέσει η γεννήτρια να επιλέξει την εντολή (Write/Read/Modify/Flush) με κάποιο τυχαίο τρόπο χρησιμοποιήσαμε την συνάρτηση Rand() που παρέχει η βιβλιοθήκη stdlib.h. Ουσιαστικά η εντολή rand() παράγει ένα αριθμό μεταξύ του 0 και του 3 και τον αναθέτει στην μεταβλητή random_num2. Χρησιμοποιούμε το περιεχόμενο αυτής της μεταβλητής ως index σε ένα array από χαρακτήρες (`char entoli[4]={'R','W','M','F'};`) και έτσι πετύχαμε να επιλέγουμε εντελώς τυχαία την εντολή που θα χρησιμοποιηθεί κάθε φορά.

Με παρόμοιο τρόπο παράγουμε (randomly) και την διεύθυνση αναφοράς και την αναθέτουμε στη μεταβλητή random_num.

Με την βοήθεια της συνάρτησης dec2bin που υλοποιήσαμε μετατρέπουμε τον τυχαίο δεκαδικό αριθμό που παράγουμε σε δυαδικό.

Decimal to Binary Function:

Η συνάρτηση αυτή που υλοποιήσαμε δέχεται σαν παραμέτρους ένα δεκαδικό αριθμό και ένα πίνακα ακεραίων. Στην ουσία αυτό που κάνει είναι διαίρεση με το 2 και κρατά το υπόλοιπο μέσα στον πίνακα με σειρά (MSB to LSB). Έτσι μπορούμε να μετατρέψουμε ένα δεκαδικό αριθμό σε δυαδικό.

Data File Parser:

Σε αυτό το σημείο έχουμε καταφέρει με επιτυχία να δημιουργήσουμε ένα αρχείο με τυχαίες αναφορές σε διεύθυνσης και τυχαία εντολή σε κάθε αναφορά. Άρα μπορούμε τώρα να περάσουμε στην επόμενη φάση του προγράμματος μας η οποία είναι να διαβαστεί το αρχείο δεδομένων που έχει παραχθεί. Το πρόγραμμα πρέπει να είναι ικανό να αναγνωρίζει το κάθε πεδίο κάθε αναφοράς και να το χειρίζεστε αναλόγως.

Με την εντολή fscanf_s διαβάζουμε την κάθε γραμμή του αρχείου χαρακτήρα προς χαρακτήρα και φυλάσσουμε τον χαρακτήρα που δείχνει ο δείκτης μέσα σε μία μεταβλητή.

```
fscanf_s(input,"%c",&μεταβλητή,1);
```

Αναλόγως τώρα με τον τύπο κρυφής μνήμης που θα επιλεγεί θα γίνει και ανάλογος χειρισμός των μεταβλητών που γέμισαν από τον parser.

Ένα παράδειγμα εξόδου που μπορεί να προκύψει είναι:

Address: 11100111 | Command: Write | Tag: 11100 | Index: 1 | block Offset: 11

Direct Mapped Function:

Στην περίπτωση που ο χρήστης επιλέξει κρυφή μνήμη άμεσης απεικόνισης υπολογίζουμε το Tag, index, block offset(αν υπάρχει) με βάση τις στατικές παραμέτρους που έδωσε ο χρήστης. Όπως εξηγήσαμε πιο πάνω την binary διεύθυνση αναφοράς την κάνουμε parse και την φυλάσσουμε σε προσωρινά σε ένα array. Οι μεταβλητές index, tag, block offset λειτουργούν σαν όρια και έτσι με την βοήθεια ενός for loop επιτυγχάνουμε να σπάσουμε και να τυπώσουμε την διεύθυνση αναφοράς σε ένα αρχείο εξόδου με την εξής μορφή:

Address: 01000101 | Command: Modify | Tag: 0100 | Index: 01 | block Offset: 01

Fully Associative Function:

Και στην *Fully Associative* κρυφή μνήμη υπολογίζουμε και τυπώνουμε τα Tag και block offset αλλά όχι το Index διότι ουσιαστικά έχουμε 0 bits για index άρα όλα τα bits που είχαμε για index στην direct mapped θα πάνε και θα προστεθούνε στο Tag.

N-Ways Associative Function:

Παρόμοιος τρόπος σκέψεις και εδώ. Το μόνο επιπρόσθετο στοιχείο που προκύπτει είναι ότι θα πρέπει να υπολογιστεί και η οδός με βάση την στατική παράμετρο που θα δώσει ο χρήστης. Σε αυτό το millstone δεν υλοποιούμε την πλήρη λειτουργία μιας κρυφής μνήμης, απλά την διευθυνσιοποίηση. Άρα χρησιμοποιούμε την συνάρτηση rand() για να υπολογίσουμε τυχαία πια θα είναι η οδός της συγκεκριμένης αναφοράς.

Division of Labor:

Μιχάλης Παναγιώτου:

a) **Υλοποίηση Data File Generator**
b) **Υλοποίηση Συνάρτησης Direct Mapped**
c) **Υλοποίηση Συνάρτησης N-Ways Associative**

Βαρνάβας Ξυδάς:

a) **Υλοποίηση Parser**
b) **Υλοποίηση Συνάρτησης Full Associative**
c) **Υλοποίηση Συνάρτησης Bin2Dec**

Στην γραφή του κώδικα χρησιμοποιήσαμε τεχνική pair programming. Δηλαδή, ο ένας έγραφε κώδικα και ο άλλος έκανε debugging ταυτόχρονα. Έτσι με αυτό τον τρόπο καταφέραμε να περιορίσουμε τα λάθη μας στο τελικό κώδικα στο ελάχιστο και είχαμε και μεγαλύτερη απόδοση σαν ομάδα.

Challenges and Difficulties during the Implementation:

1. Συναντήσαμε κάποιες δυσκολίες στο άνοιγμα και δημιουργία αρχείων λόγω του ότι χρησημοποιούσαμε την `fopen` η οποία είναι κάπως παλεμένη μέθοδος. Λύσαμε αυτό το πρόβλημα με τη χρήση της `fopen_s` και ενός ελεκτή σφαλμάτων `errorCode` τύπου `errno_t`.
2. Άλλο σημαντικό bug που εντοπίσαμε είναι στην χρήση της συνάρτησης rand(). Αν τρέχαμε το πρόγραμμα 2 φορές συνεχόμενα παρατηρούσαμε ότι οι τυχαίοι αριθμοί που παράγονταν ήταν πάντα ίδιοι και με την ίδια σειρά. Λύσαμε αυτό το πρόγραμμα με την βοήθεια της `srand((int)time(NULL));` η οποία επηρεάζει την συνάρτηση rand() και οι τυχαίοι αριθμοί παράγονται με βάση ένα ρολόι.
3. Τρέξαμε το πρόγραμμα μας με μια πληθώρα από διαφορετικές παραμέτρους και εντοπίσαμε κάποια πιθανά λάθη που μπορούν να παρουσιαστούν λόγο λανθασμένων στατικών παράμετρών (π.χ size of Cache = -2). Επιλύσαμε τα προβλήματα αυτά με την εισαγωγή ελέγχων στην αρχή την κύριας συνάρτησης (main()).