# Αναφορά 1ης Εργασίας

Μπέτσιος Θωμάς

Α.Μ. 2013030104

**Στόχος**

Στόχος της 1ης εργασίας είναι η επανάληψη της γλώσσας VHDL και η εξοικείωση στην χρήση της, καθώς και η επανάληψη και καλύτερη κατανόηση του instruction set architecture (Καθώς η εργασία ακολουθεί ενα instruction set βασισμένο σε αυτό του επεξεργαστή MIPS), και πως γίνεται το fetching και το decoding του συνόλου των εντολών αυτών. Ο κύριος στόχος είναι η δημιουργία και κατανόηση ενός ολοκληρωμένου επεξεργαστή ενός κύκλου υλοποιώντας το data & το control path.

**Περιγραφή**

Η εργασία χωρίζεται σε τρείς φάσεις.

**1η φάση:**

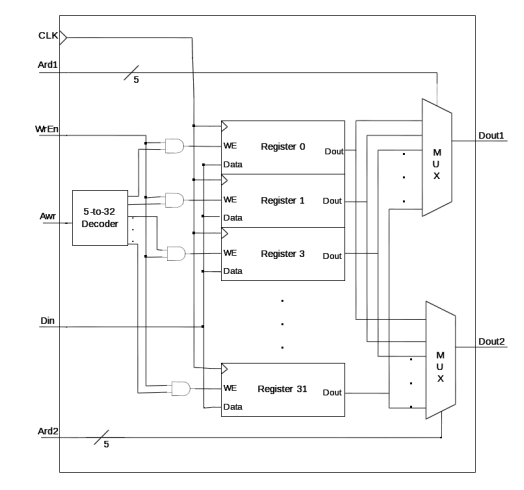
Στην 1η φάση μας ζητάται να υλοποιήσουμε μια μονάδα αριθμητικών και λογικών πράξεων (ALU) και ένα αρχείο καταχωρητών.

i) Αρχικά δημιουργήσα την ΑLU με τις εισόδους και τις εξόδους που μας ζητούνταν από την εκφώνηση. Στην συνέχεια, χρησιμοποιώντας behavioral τρόπο σχεδίασης, υλοποιήσα μία σειρά από λογικές και αριθμητικές πράξεις (πρόσθεση,αφαίρεση, λογικό AND, ολισθήσεις κλπ), όπως ζητούνταν από τον αντίστοιχο πίνακα της εκφώνησης. Τέλος εκανα προσομοίωση της λειτουργίας για μερικές πιθανές εισόδους, έτσι ώστε να επιβεβαιώσω την ορθότητα του κυκλώματός μας.

Παρατηρούμε πώς:

* η έξοδοι εμφανίζεται με καθυστέρηση σε σχέση με τις εισόδους, καθώς έκανα χρήση του after της vhdl.
* όταν το αποτέλεσμα μιας λογικής ή αριθμητικής πράξης είναι 0 τότε το σήμα zero γίνεται ‘1’.
* όταν στην πρόσθεση ή στην αφαίρεση έχουμε υπερχείληση ή κρατούμενο εξόδου, οι αντίστοιχες μεταβλητές γίνονται ‘1’.

ii) Έπειτα υλοποιήσα με behavioral τρόπο σχεδίασης έναν καταχωρητή (Register), έναν Πολλυπλέκτη (Mux) και έναν Αποκωδικοποιητή (Decoder). Στην συνέχεια χρησιμοποιώντας στιγμιότυπα απο τα παραπάνω υποκυκλώματα δημιουργησα το Register File και συνδεσα στο port map τις εισόδους και τις εξόδους ωστε το τελικό κύκλωμα να υλοποιεί την λειτουργικότητα που θέλουμε. Τέλος δημιουργησα ενα testbench για να ελέγχξουμε την ορθή λειτουργία του κυκλώματος.



Πιο αναλυτικά για το Register File:

Στο Top Level (Register File) δημιουργησα 32 στιγμιότυπα απο ίδιους Registers (32 bits) καθώς και 2 Mux (32 to 1 bits) και 1 Decoder ( 5 to 32 bits), με σκοπό να τους συνδέσω όπως μας ζητείται στο σχεδιάγραμμα της άσκησης.

Δε ξεχάσα να προσθέσω καθυστέρηση 10 nsec στις εξόδους των υποκυκλωμάτων για να προσομοιώσουμε τις καθυστερήσεις ενος πραγματικού κυκλώματος, το οποίο σημαίνει ότι η έξοδος των καταχωρητών αλλα και των άλλων κυκλωμάτων θα αλλάζει τιμή 10 nsec μετά από την είσοδο.

Παρατηρούμε οτι:

* όταν το WE είναι ‘0’ τα δεδομένα δεν γράφονται στους καταχωρητές και επομένως δε περνάνε στην έξοδο
* έχοντας το WE = ‘1’ επιβεβαιώνω πως η έξοδος γίνεται η επιθυμητή όταν αλλάζει η είσοδος.
* Ο καταχωρητής R0 έχει πάντα την τιμή μηδέν.

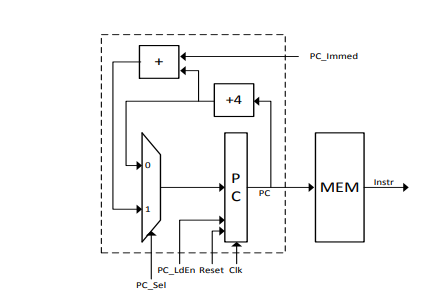
**2η φάση:**

Σκοπός της 2ης φάσης της ασκησης είναι η σχεδίαση των βασικών βαθμίδων του dataPath ενός απλού επεξεργαστή ενός κύκλου. Τις βαθμίδες αυτές αργότερα θα τις ενώσω σε ένα ανώτερο επίπεδο που θα υλοποιεί όλες τις απαραίτητες λειτουργίες του επεξεργαστή όσον αφορά το data path.

1. Αρχικα, μελέτησα την κωδικοποίηση εντολών του CHARIS και ομαδοποίησα τις λειτουργίες για να μπορώ να υλοποιήσω την αποκωδικοποίηση των εντολών αυτών.

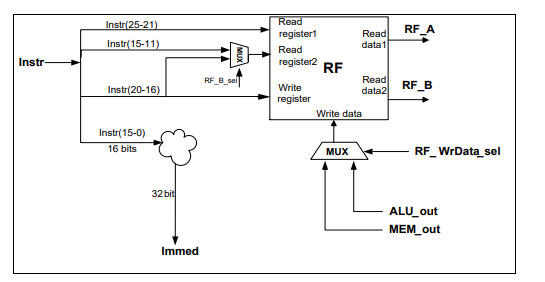
ii) Σε αυτό το μέρος, έφτιαξα μια μνήμη RAM 2048 θέσεων των 32 bits, με σήματα εισόδου και εξόδου που μας δίνονται από την εκφώνηση.

iii) Στο τρίτο μέρος της άσκησης, σχεδίασα την βαθμίδα ανάκλησης εντολών (IFSTAGE). Αυτή η βαθμίδα, αφορά την σωστή λειτουργία του Program Counter (PC) register ο οποίος διευθυνσιοδοτεί το πρόγραμμα, αυξάνοντας την τιμή του κατά 4 όταν το πρόγραμμα έχει κανονική ροή, ενώ στην περίπτωση branch και jump εντολών αυξάνεται κατά 4 συν μία κατάλληλη τιμή PC\_immed, με σκοπό να πάει το πρόγραμμα στην ορθή θέση.



iv) Στη συνέχεια, υλοποίησα την βαθμίδα αποκωδικοποίησης εντολών(DECSTAGE),στην οποία χρησιμοποιώ το register file που είχα υλοποιήσει στην 1η φάση της εργασίας. Το αρχείο καταχωρητών παίρνει από τη RAM μια 32-bit ποσότητα, η οποία στη συνέχεια αποκωδικοποιέιται και ανάλογα με την εντολή τα δεδομένα γράφονται σε 2 καταχωρητές ή σε έναν καταχωρητή και απομένουν τα bits που αποτελούν το immediate.

* Το 16-bit immediate υπόκειται σε sign extend μέσω μίας μονάδας extender αφού θέλουμε να είναι 32-bit, που μπορεί να υλοποιεί (zero-filling, sign-extend ή zero-filling στα 16 LSB οταν έχουμε LUI εντολή).
* Επίσης προσθεσα έναν πολυπλέκτη, ο οποίος επιλέγει αν τα δεδομένα που θα γραφτούν στον καταχωρητή προορισμού θα προέρχονται από την μνήμη ή από την ALU.



v) Σε αυτό το μέρος (EXSTAGE), προσθέσα την ALU. Προσθεσα επιπρόσθετα έναν πολυπλέκτη ο οποίος ρυθμίζει την δεύτερη είσοδος της ALU αν θα είναι ένας δεύτερος καταχωρητής η το Immediate. Σαν έξοδο απο την ALU μπορούμε να πάρουμε είτε μία τιμή από μια πράξη είτε μία διεύθυνση. Η έξοδος αυτή εξαρτάται από τον τύπο της εντολής που θα εκτελέσει η ALU αφού σε εντολές όπως το sw(store word) χρειαζόμαστε διευθύνσεις μνήμης.

vi) Στο τελευταίο μέρος της 2ης φάσης, υλοποιήσα την βαθμίδα πρόσβασης μνήμης (ΜΕΜSTAGE), η οποία χρησιμοποιείται για το γράψιμο δεδομένων από τη θέση 0x400 και έπειτα, και κατά την εκτέλεση των εντολών load και store. Πρόσεξα λοιπόν, όταν εισάγουμε μία διεύθυνση από την ALU, να προσθέτω την αρχική θέση αυτή για να τοποθετούνται τα δεδομένα στην σωστή θέση μνήμης. Επίσης υλοποιεί μιαλειτουργικότητα σε περίπτωση που έχουμε εντολές sb/lb.

**3η φάση:**

Στην 3η φάση, υλοποίησα το DATAPATH ενώνοντας τα παραπάνω modules καθώς το CONTROL και τον επεξεργαστή ενός κύκλου (PROC\_SC).

Αρχικά για το DATAPATH ένωσα τις 4 παραπάνω βαθμίδες χρησιμοποιώντας signals ωστε να πετύχω την λειτουργικότητα που ζητάται. Παρατήρησα στο testing του Datapath, οτι ενώ τα test των προηγούμενων βαθμίδων βγάζουν σωστά αποτελέσματα, στο συγκεκριμένο η έξοδος της ALU και κατ’επέκταση και η διεύθυνση για τα δεδομένα στο DECSTAGE έχει την τιμή Χ. Έλεγξα αρκετές φορές τα σήματα και νομίζω οτι έχω σωστή συνδεσμολογία, αλλα δεν μπόρεσα να βρώ τι φταίει.

Έπειτα στο CONTROL, υλοποίησα μια fsm με καταστάσεις όλες τις πιθανές εντολές, η οποία ανάλογα με την κατάσταση παράγει και τα αντίστοιχα δεδομένα ελέγχου που στην συνέχεια θα τα συνδέσουμε με το DATAPATH και την μνήμη RAM. Στο testing του CONTROL παρατήρησα οτι οι τιμές παράγονται σωστά, αλλα έχω ένα θέμα με τον συγχρονισμό, γιατι αλλάζει κατάσταση και στον αρνητικό χτύπο ρολογιού ενώ χρησιμοποιώ το rising\_edge(Clk) της vhdl.

Τέλος συνδεσα το DATAPATH το CONTROL και την RAM σε ένα module με όνομα PROC\_SC. Έκανα τις κατάλληλες συνδέσεις στο port map ωστε να βγαίνει η σωστή υλοποίηση.