

3^η Άσκηση: Σχεδίαση επεξεργαστή pipeline

Κύρκος Κωνσταντίνος 2022030112

Χαράλαμπος Μυλωνάκης 2022030133

1. Σκοπός της Άσκησης

Σκοπός της 3ης εργαστηριακής άσκησης είναι η σχεδίαση ενός επεξεργαστή pipeline. Σαν βάση για την σχεδίαση χρησιμοποιήθηκε ο multi-cycle επεξεργαστής που σχεδιάστηκε στην προηγούμενη εργασία και προστέθηκαν σε αυτόν καταχωρητές για τα ενδιαμέσως στάδια καθώς και τα σήματα ελέγχου (control). Επιπλέον σχεδιάστηκαν τα κατάλληλα units για την αντιμετώπιση των datahazards με forwarding και με stalls.

2. Περιγραφή της Σχεδίασης

Το κύριο χαρακτηριστικό του pipeline επεξεργαστή είναι ότι εκτελεί πολλές εντολές ταυτόχρονα σε έναν κύκλο του ρολογιού, σε αντίθεση με τους single και multi cycle επεξεργαστές όπου για να ξεκινήσει μια νέα εντολή πρέπει πρώτα να έχει ολοκληρωθεί η τρέχουσα. Η αρχή της λειτουργίας του pipeline επεξεργαστή μοιάζει πολύ με αυτή του multi-cycle ως προς το ότι βασίζεται στον διαχωρισμό της εκτέλεσης σε στάδια τα οποία είναι όμοια με τα στάδια του multi-cycle:

IF stage / DEC stage / EXEC stage / MEM stage / WB stage

Για να γίνει το παραπάνω, τοποθετήθηκαν καταχωρητές μεταξύ των 5 σταδίων που αποθηκεύουν όλες τις εξόδους που παράγονται από το κάθε στάδιο και περνάνε σαν εισόδους στο επόμενο. Ακόμη, τοποθετήθηκαν καταχωρητές και για τα control σήματα ώστε να μη χάνεται η πληροφορία που χρειάζονται οι εντολές για να λειτουργήσουν όταν περνάνε από το ένα στάδιο στο επόμενο. Όλοι οι απαραίτητοι καταχωρητές έχουν ομαδοποιηθεί στα εξής modules (για λόγους ιεραρχικής σχεδίασης):

IF DEC / DEC EX / EX MEM / MEM WB

Για την αντιμετώπιση των datahazards δημιουργήθηκαν το Forwarding Unit και το Hazard Detection Unit. Στην περίπτωση του Forwarding Unit, hazard μπορεί να προκύψει όταν η τιμή ενός καταχωρητή του Register File χρειαστεί σε μια επόμενη εντολή (αφού οι εντολές εκτελούνται ταυτόχρονα, δε θα έχουν προλάβει να ολοκληρωθούν οι 5 κύκλοι εκτέλεσης). Αφού πραγματοποιηθούν (σύμφωνα με την θεωρία) οι απαραίτητοι έλεγχοι και επιβεβαιωθεί ότι υπάρχει hazard, δίνουμε την κατάλληλη εισόδο στον Register File χωρίς να περιμένουμε αυτή να βγει από το WB stage. Στην περίπτωση του Hazard Detection Unit, επιτυγχάνεται η αντιμετώπιση των datahazards μέσω stalls. Σε αυτή την περίπτωση, όταν ένας καταχωρητής παίρνει τιμή από την μνήμη, πρέπει αναγκαστικά να περάσει και από τα 5 στάδια εκτέλεσης, οπότε αμα χρησιμοποιείται από μια επόμενη εντολή, αυτή πρέπει να περιμένει. Μόλις ανιχνευθεί hazard, παγώνουν τα IF stage και DEC stage απενεργοποιώντας τα write enable των κατάλληλων καταχωρητών και εισάγοντας μηδενικά στην βαθμίδα σημάτων ελέγχου του DEC EX.

Παρακάτω φαίνεται το block diagram της σχεδίασης.

