ΕΡΓΑΣΤΗΡΙΟ 6

ΟΡΓΑΝΩΣΗ ΥΠΟΛΟΓΙΣΤΩΝ
ΗΡΥ312

Επιμέλεια :

Ξανθάκης Μανόλης ΑΜ :2013030101
Λογοθέτης Φραγκούλης ΑΜ: 2013030016

Προσθήκη εξαιρέσεων και κρυφής μνήμης

Εισαγωγή και περιγραφή της άσκησης

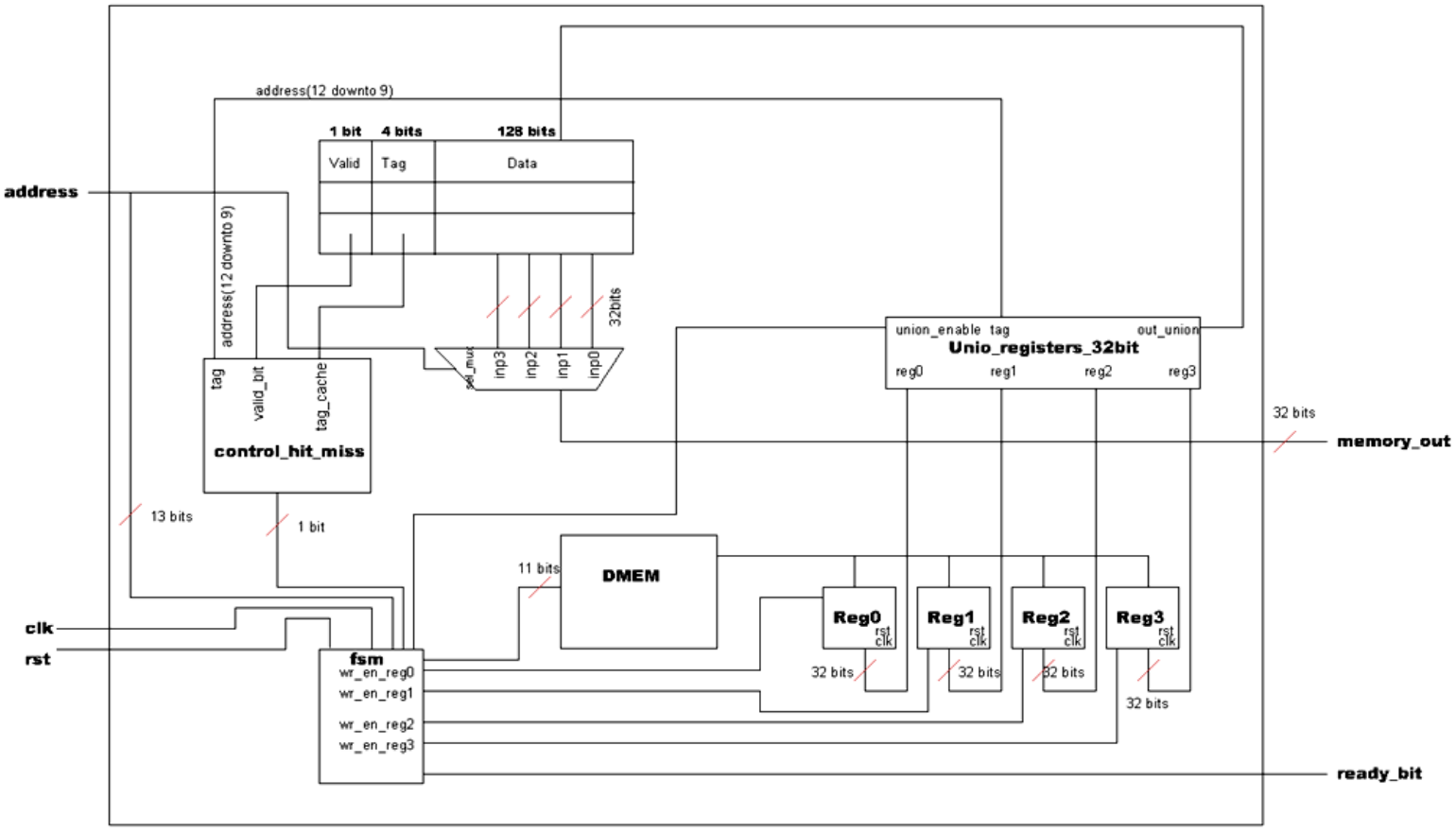
Σκοπός του εργαστηρίου ήταν η σχεδίαση ενός συστήματος μνήμης με κυρίως και κρυφή μνήμη . Πιο συγκεκριμένα η κρυφή μνήμη (cache) που δημιουργήσαμε είναι άμεσης απεικόνισης(direct mapped: κάθε θέση της μνήμης απεικονίζεται απευθείας σε ακριβώς μία θέση στην κρυφή μνήμη). Η μορφή της cache ειναι η παρακάτω :

Lines	1 bit	;;; bits	128 bits
0	Valid	Tag	Data
…			
31			

Valid bit: Σήμα που δείχνει αν μια καταχώρηση περιέχει έγκυρη διεύθυνση.
Tag : Περιέχει τις πληροφορίες της διεύθυνσης που απαιτούνται για να προσδιοριστεί αν το σχετικό μπλοκ αντιστοιχεί στη λέξη που ζητήθηκε(συγκρίνεται με tag του address).

Τα βήματα που ακολουθήθηκαν :

Βήμα 1: Σχεδιασμός Datapath



Η κρυφή μνήμη περιέχει 32 λέξεις και η διεύθυνση είναι 13 bit ακλουθώντας το format του πίνακα 1. Η ετικέτα στην κρυφή μνήμη*(tag_cache)* συγκρίνεται με το υψηλότερο τμήμα της διεύθυνσης

(address(12 downto 9)) για να προσδιοριστεί αν η καταχώριση στην κρυφή μνήμη αντιστοιχεί στην διεύθυνση που ζητήθηκε. Επειδή η κρυφή μνήμη έχει 2^5 λέξεις και κάθε set αποτελείται από 1 block χρησιμοποιούνται 4 bit για το tag όπως φαίνεται παρακάτω:

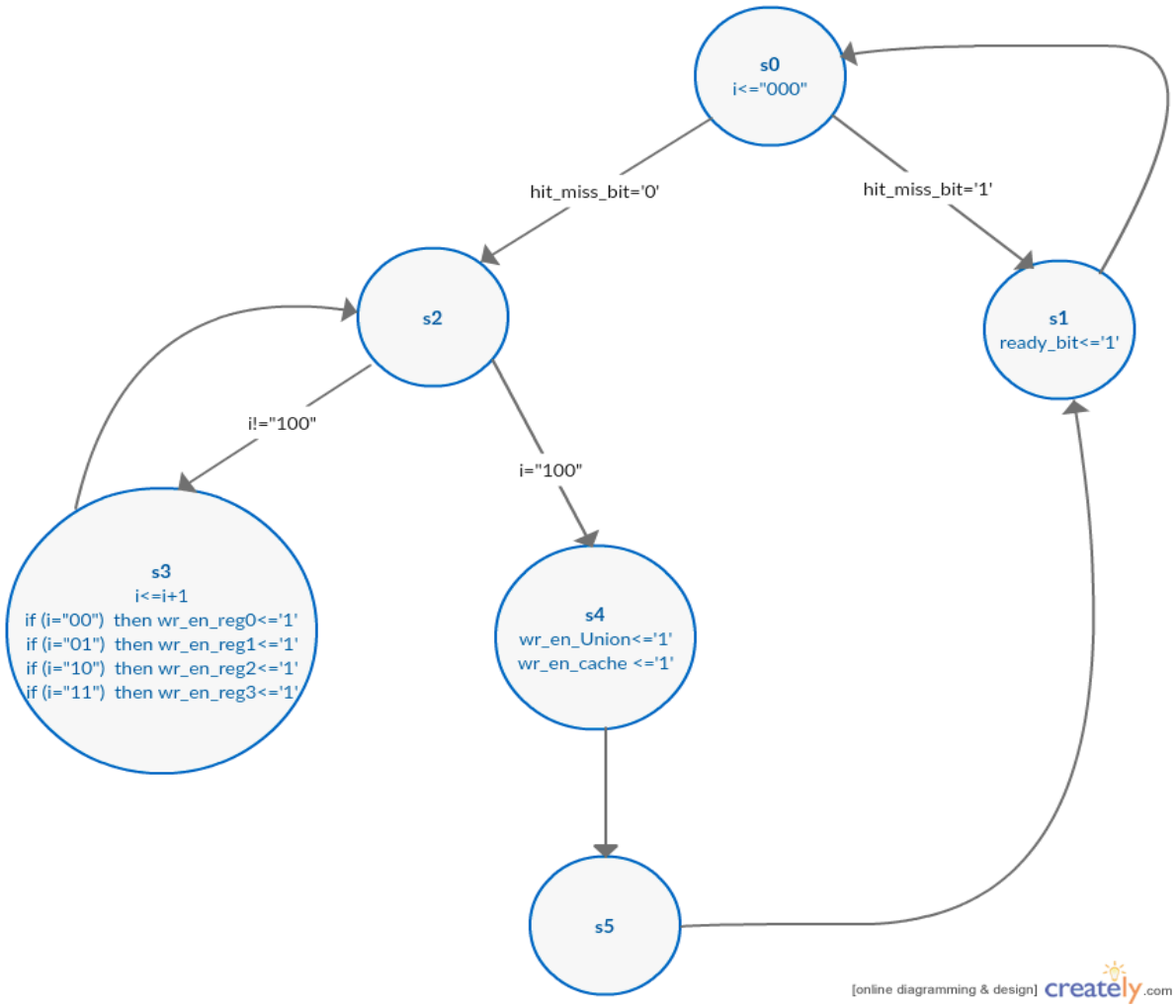
Πίνακας 1

tag	index	Word offset	Byte offset
4 bits	5bits	2bits	2bits

Αν το tag_cache και το address(12 downto 9) *(tag)* είναι ίσα και το valid bit ενεργοποιημένο τότε η αίτηση έχει ευστοχία στην cache ,αλλιώς έχει αστοχία.

Η κυρίως μνήμη έχει δεκαεξαπλάσιο μέγεθος από την κρυφή μνήμη δηλαδή 128*16=2048 bits δεδομένων μνήμης .

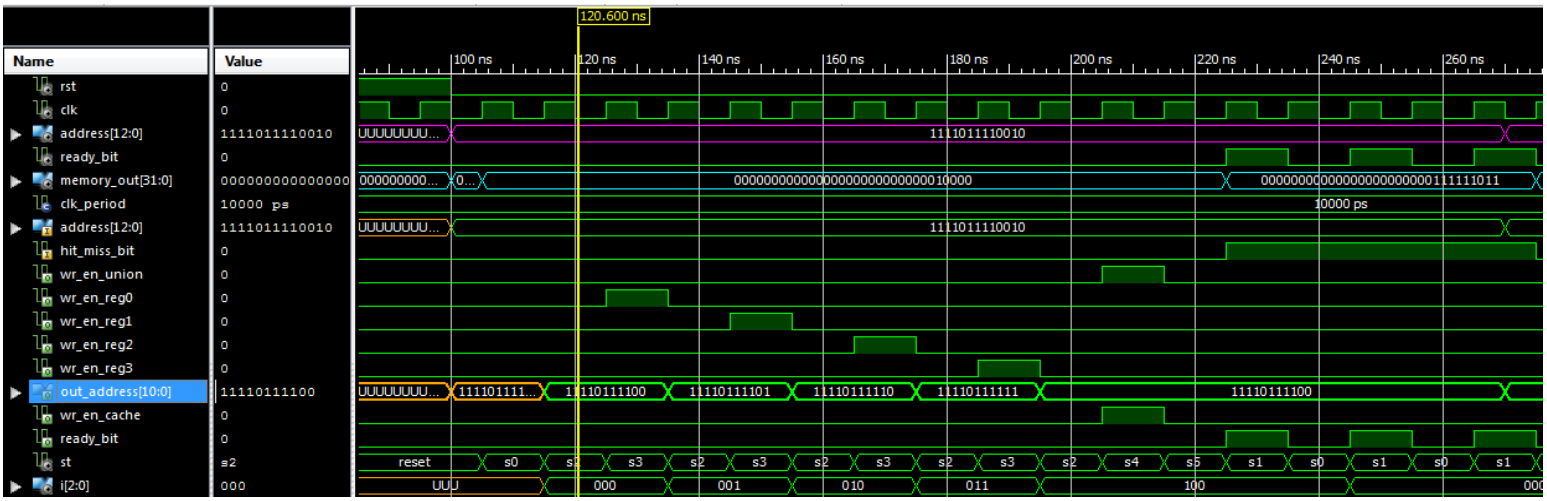
Βήμα 2: Σχεδιασμός FSM



Για τον χειρισμό των περιπτώσεων αστοχίας και ευστοχίας όπως αυτές αναφέρθηκαν παραπάνω δημιουργήσαμε μια μηχανή πεπερασμένων καταστάσεων(fsm). Η μνήμη cache αρχικοποιείται μέσω του αρχείου cache.data και η κύρια μνήμη μέσω του αρχείου dmem.data.

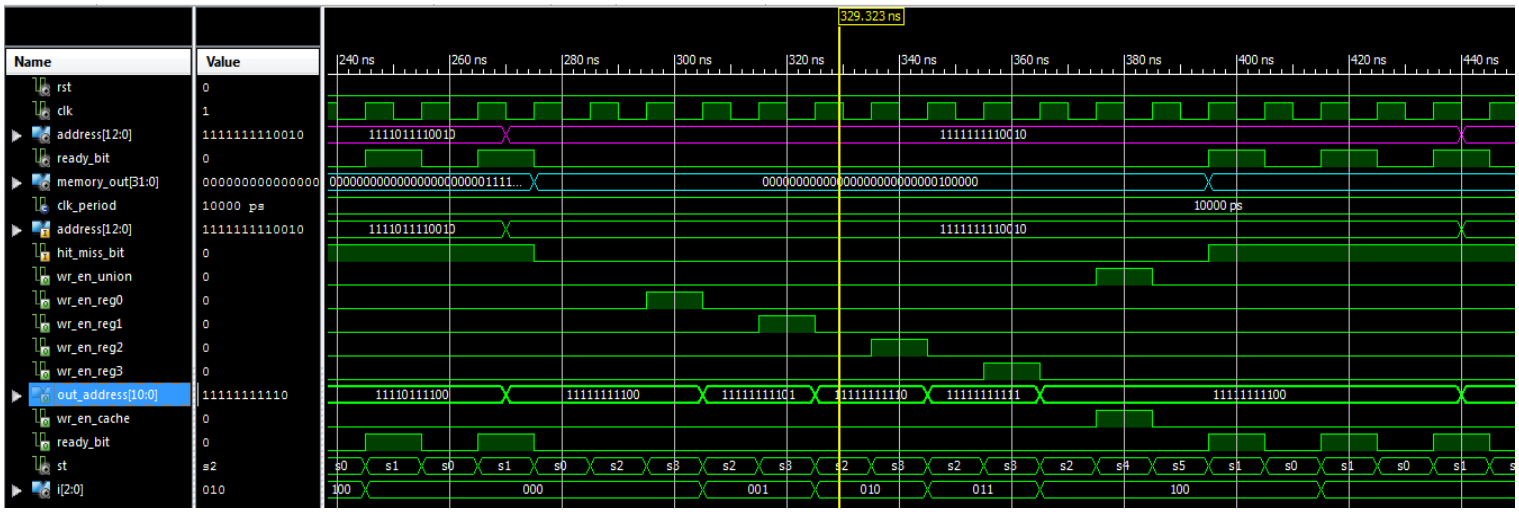
Βήμα 3 : Επιβεβαίωση ορθής λειτουργίας με Simulations

address<="1111011110010"
tag=1111
index=01111
wo=00
bo=10



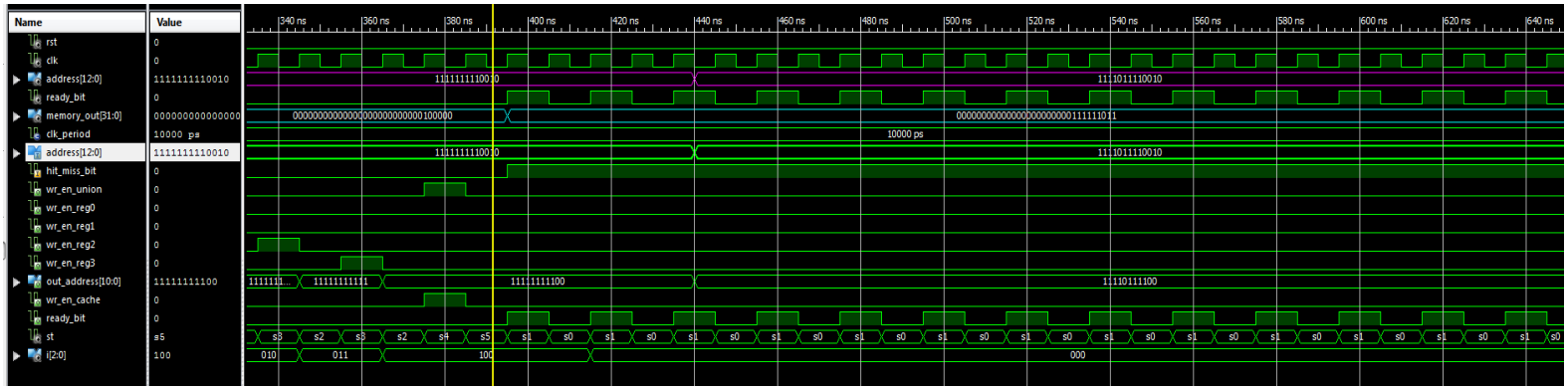
✓ Παρατηρούμε ότι δίνοντας την διεύθυνση "1111011110010" ως είσοδο αρχικά έχουμε αστοχία (hit_miss_bit='0') .Έτσι στέλνεται η διεύθυνση στην κυρίως μνήμη η οποία επιστρέφει τα δεδομένα στην cache.Αυτό επαληθεύεται και στο simulation καθώς δίνοντας ξανά την ίδια διεύθυνση έχουμε ευστοχία.

address<="1111111110010"
tag=1111
index=11111
wo=00
bo=10



✓ Το ίδιο συμβαίνει και εδώ , δηλαδή δίνοντας μια διεύθυνση η οποία δεν υπάρχει στην κρυφή μνήμη ,γίνεται miss και έπειτα δίνοντας την ίδια διεύθυνση έχουμε hit.

address<="1111011110010"



✓ Τέλος δίνοντας ως είσοδο την αρχική διεύθυνση παρατηρούμε ότι έχουμε hit.

<u>Σημείωση</u>

Σε κάθε περίπτωση διαλέγονται οι σωστές λέξεις από τον πολυπλέκτη και περνάνε ως έξοδος(memory_out) . Ο έλεγχος του πολυπλέκτη γίνεται μέσω του word offset, ο οποίος επιλέγει την ζητούμενη λέξη από τις 4.