4η Εργαστηριακή Άσκηση-ce430

Αξελός Χρήστος | ΑΕΜ 1814 | caxelos@inf.uth.gr|11/01/16

ΣΤΟΧΟΙ

- Εξικοίωση στην οδήγηση εξόδων στο LCD, χρήση της πλακέτας Spartan 3E

ΠΡΟΒΛΗΜΑΤΑ ΣΤΗΝ ΥΛΟΠΟΙΗΣΗ

- Χρησιμοποίησα την 64-bit distributed ram της πλακέτας αντί για την πάνω αριστερά μνήμη bram
- Το κύκλωμα δεν έτρεχε στην πλακέτα. Οι κυματομορφές θεώρησα ότι είναι σωστές αλλά δεν βρήκα το λόγο που δεν έτρεχαν στην πλακετα

1. ΜΕΡΟΣ 1: Υλοποήση ΜΠΚ Εντολών

ΥΛΟΠΟΙΗΣΗ

Γίνεται στο module instruction_fsm, όπου χρησιμοποιηώ 6 καταστάσεις. Οι μεταβολή του counter του fsm "wcounter"(waitCounter) γίνεται στο αρχείο init.v:

A) πάρε τα 4 upper bits(και αναμονή 40ns)

B) κάνε το LCD_E = 1(και αναμονή 240ns)

C) κάνε το LCD_E = 0(και αναμονή 1μs)

D) πάρε τα 4 lower bits(και αναμονή 60μs)

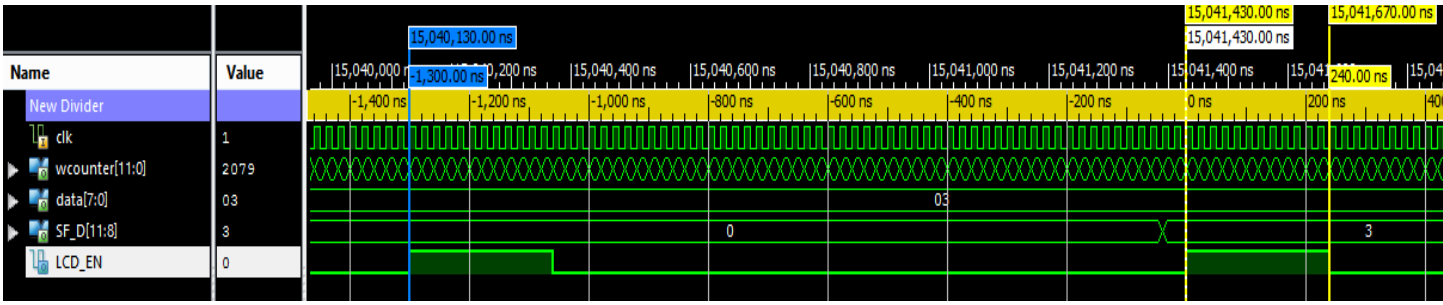
E) κάνε το LCD_E = 1(και αναμονή 240ns)

F) κάνε το LCD_E = 0(και αναμονή 40μs)

- Επειδή το 4ο bit του data[7:0] δεν μεταβάλλεται, για να μην έχω πρόβλημα με latches δημιούργησα τα regs: sixth, seventh, seventh και threeToZero[3:0], ώστε το 4ο bit να γίνει ανάθεση ως wire

ΕΠΑΛΗΘΕΥΣΗ

Σχήμα 1: Θέλουμε να στείλουμε την 1η εντολή 0x03, όπου ο μπλε κέρσορας δείχνει την αρχή των upper bits ενώ για t = 15,041,430ns στέλνονται τα upper bits



ΜΕΡΟΣ 2: Υλοποίηση Κεντρικής ΜΠΚ Ελέγχου

ΥΛΟΠΟΙΗΣΗ

Για να οργανώσω το κεντρικό fsm, το χωρίζω σε 2 υποfsm. Ένα πριν γίνει η διαδικασία της αρχικοποίησης και διαμόρφωσης, όπου απαιτούνται 12 συνολικά καταστάσεις, και ένα για την αποστολή των δεδομένων, όπου χρησιμοποιούνται 3 καταστάσεις:

- Όταν έχει εκτυπωθεί όλο το μύνημα που θέλουμε και δεν υπάρχουν άλλες αλλαγές
- Έχει γεμίσει το 4-bit buffer για αποστολή δεδομένων για αποστολή
- Σε οποιαδήποτε άλλη περίπτωση, όταν δηλαδή γεμίζει το buffer

Επειδή στα δεδομένα δεν αλλάζουν τα upper bits αλλά μένουν σταθερά στην τιμή data[7:4]=4'b0100, δεν χρειάζομαι κατάσταση όπως στα lower bits που μεταβάλλονται. Χρησιμοποιώ στο module init έναν 21-bit μετρητή, ο οποίος στέλνει τις παρακάτω εντολές σε ένα μεγάλο fsm τις χρονικές στιγμές(σε κύκλους 50MHz):

A) 21'd750_000: 0x03

B) 21'd955000: 0x03

C) 21'd960001: 0x03

D) 21'd962002: 0x02

E) 21'd964003: 0x28

F) 21'd964004: 0x06

G)21'd964005: 0x0C

H)21'd964006: 0x01

I)21'd1046007: 0x80

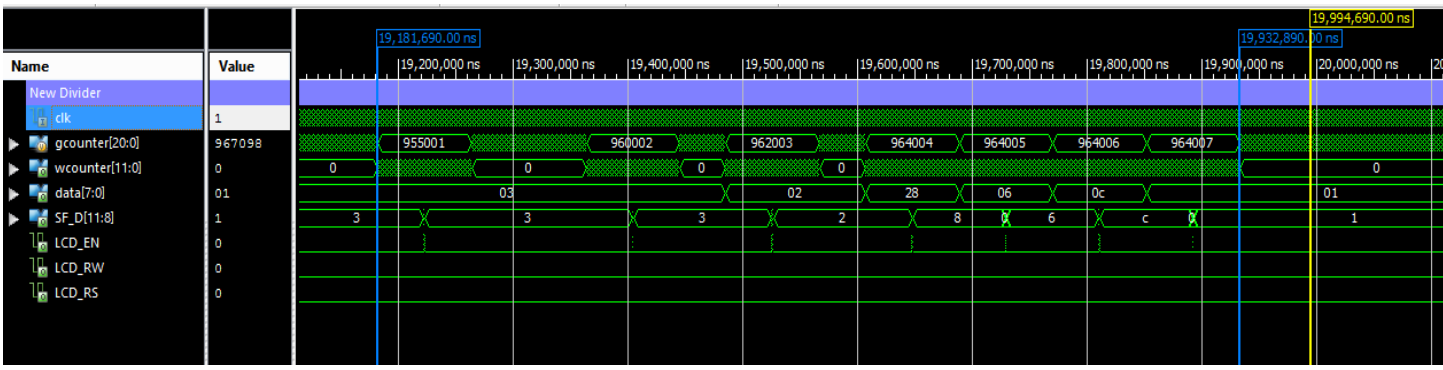
Από εδώ και κάτω ενεργοποιείται απο σήμα LCD_RS και στέλνονται δεδομένα

- Για την ανάκτηση δεδομένων από την μνήμη, χρησιμοποιώ το σήμα CONFIG_EN = LCD_RS, το οποίο όσο είναι μηδέν "εμποδίζει" την μνήμη να διαβάζει. Για την επιλογή της διεύθυνσης της μνήμης, χρησιμοποιώ τα 6LSB του 21-bit μετρητή gcounter[5:0]. Το module της μνήμης 8bitbram καλείται από το module init
- Τα υπόλοιπα στοιχεία που ίσως είναι δυσνόειτα τα εξηγώ στον κώδικα
- Παρακάτω φαίνονται οι κυματομορφές αποστολής εντολής(εικόνα 1) και δεδομένων (εικόνα 2)

Σχήμα2: Αποστολές των εντολών, με αρχή την 1η 0x03(1ος μπλε κέρσορας) και τέλος την αποστολή της εντολής clear 0x01(2ος μπλε κέρσορας).

Ο gcounter(global counter) είναι ο μετρητής του κεντρικού fsm, ενώ ο wcounter(wait counter) του fsm εντολών

Το LCD_RS παραμένει μηδέν, αφού στέλνουμε εντολές. Στο σχήμα αυτό δεν φαίνονται καλά τα upper bits του SF_D[11:8] και οι ενεργοποιήσεις καθώς είναι έκανα αρκετο zoom out, αλλά φαίνονται καθαρά στο σχήμα του μέρους 1



Σχήμα3: Αποστολή της τελευταίας εντολής 0x80(ανάθεση διεύθυνσης στην ddram=0) και δεκαεξαδικών δεδομένων 0x40 = 'A', 0x41 = 'B', 0x42 = 'C' , κτλ.

- *Ο κίτρινος κέρσορας δείχνει την αποστολή της τελευταίας εντολής*
- *Ο 1ος μπλε την ενεργοποίηση σήματος για αποστολή δεδομένων αντί εντολών*
- *Ο 2ος μπλε την αποστολή της 0x40, ο 3ος 0x41 , ο 4ος 0x42, κ.τ.λ.*