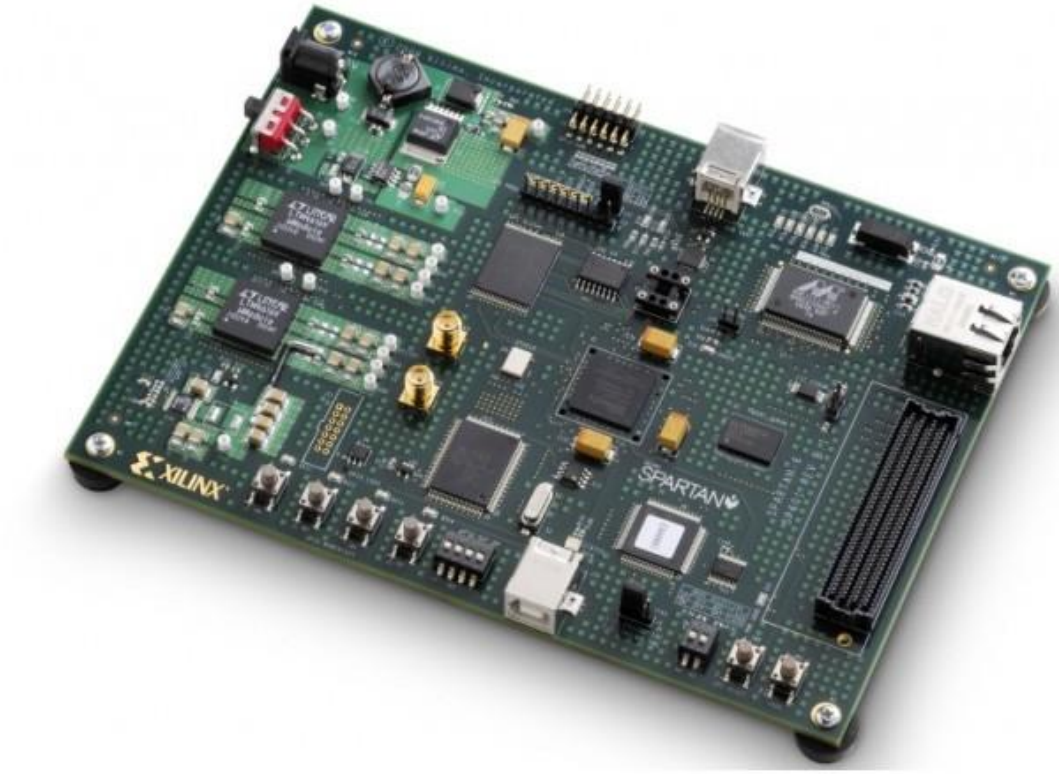
CAD for Digital H/W (E-CAD)

Αναφορά Εργασίας



Λαγός Δημήτρης 4480

Μαυροφόρος Νίκος 4770

Περιγραφή Modules

Σκοπός της φετινής εργασίας εξαμήνου είναι η σχεδίαση ενός Συστήματος Εισαγωγής & Απεικόνισης Χαρακτήρων. Συγκεκριμένα, εισάγουμε δεδομένα (1,2,3,4,f,←,↑,→,↓) από το πληκτρολόγιο, τα οποία αποκωδικοποιούνται στο fpga και εμφανίζονται στην οθόνη. Με το πάτημα του πλήκτρου f, τα δεδομένα που εμφανίζονται στην οθόνη αναβοσβήνουν. Στο κύκλωμα που θα συνθέσουμε χρησιμοποιούμε ένα κοινό ρολόι 25Mhz, το οποίο παράγεται από την διαίρεση του ρολόι του fpga (100Mhz) με 4 χρησιμοποιώντας counter, στο **module pix_clk.**

Θέλουμε να αντλήσουμε τα scancodes από το πάτημα του κάθε χαρακτήρα. Για να γίνει αυτό, δειγματοληπτούμε τις τιμές του ps2data στο falling edge του ps2clk (του ρολογιού του πληκτρολογίου). Για να ανιχνεύσουμε το falling edge του ps2clk, αντί να το χρησιμοποιήσουμε σαν δεύτερο ρολόι, που είναι ανεπιθύμητη πρακτική, θα δειγματληπτήσουμε και το ps2clk, αναζητώντας πότε οι τελευταίες 8 τιμές του γίνονται h'F0. Τότε θα έχουμε falling edge. Όταν έχουμε ένα σωστό πακέτο, τροφοδοτούμε την έξοδο scancode με τα αντίστοιχα δεδομένα του ps2data. Ταυτόχρονα, στέλνουμε και ένα σήμα flag το οποίο υποδεικνύει ότι εισάγαμε έναν νέο χαρακτήρα. Η λογική αυτή υλοποιείται στο **module kbd_protocol**.

Στην συνέχεια, αποκωδικοποιούμε τα scancodes στο **module kbd_decoder**. Αποκωδικοποιούμε τα scancodes που λαμβάνουμε, μόνο αν έχουμε ενεργοποιημένη την είσοδο flag από το προηγούμενο module.Αν έχουμε εισάγει χαρακτήρα (1,2,3,4), τότε τροφοδούμε το bus starting_address με την αρχική διεύθυνση της μνήμης rom που ξεκινάει η αναπαράσταση του χαρακτήρα. Η σχετική διευθυνσιοδότηση γίνεται σε επόμενο module. Ταυτόχρονα, ενεργοποιούμε το σήμα char_enable για να υποδηλώσουμε ότι έχουμε δεχτεί χαρακτήρα. Το σήμα αυτό μηδενίζεται μόνο όταν έχουμε reset. Αν έχουμε εισάγει χρώμα (R,G,B), τότε τροφοδοτούμε τα σήματα r,g,b με 1, ώστε να τα αποκωδικοποιήσουμε σε επόμενο module. Τα σήματα αυτά μηδενίζονται σε κάθε νέο κύκλο του ρολογιού μας.

Το **module vgasync** υλοποιεί την λογική επικοινωνίας με την οθόνη. Χρησιμοποιώντας τους counters pixel_counter και line_counter και τις παραμέτρους που δόθηκαν, μπορούμε να παράξουμε τα σήματα οριζόντιου και κάθετου συγχρονισμού με την οθόνη h_sync και v_sync. Δημιουργούμε το σήμα display_area, το οποίο υποδηλώνει ότι έχουμε μπει στην περιοχή εμφάνισης χαρακτήρων στην μέση της οθόνης. Τροφοδοτούμε και το bus line το οποίο μετράει την απόσταση από το TOP_BORDER του display_area. Το bus αυτό θα αξιοποιηθεί σε επόμενο Module για την σχετική διευθυνσιοδότηση της rom. Επιπλέον, διαμορφώνει κατάλληλα τα όρια του display_area ανάλογα με το πάτημα των κουμπιών αριστερά και δεξιά.

Στο **module iterator** επιτυγχάνεται η σχετική διευθυνσιοδότηση της μνήμης.Η σχετική διεύθυνση δημιουργείται προσθέτοντας τα σήματα εισόδου starting_address από το kbd_decoder και του line από το vgasync(που είναι το offset). Η έξοδος address_output διευθυνσιοδοτεί την μνήμη, και αλλάζει τιμή σε κάθε νέα γραμμή του display_area.

Στο **module char_rom** υπάρχει η αναπαράσταση των ψηφίων 1-4 σαν μνήμη 64x8 bit.Σαν είσοδος επίτρεψης χρησιμοποιούμε το char_enable.

Το **module shifter** παίρνει σαν είσοδο την 8bit αρτηρία δεδομένων της μνήμης και στην έξοδο του βγάζει τα περιεχόμενα της σειριακά , MSB first. Το shift των ψηφίων γίνεται σε καθέ θετική ακμή του pix_clk.

Στο **module coloriser** γίνεται ο χρωματισμός των pixel που θέλουμε να εμφανίσουμε στην οθόνη σε συγχρονισμό με το ρολόι. Λαμβάνουμε σαν είσοδο την έξοδο του shifter καθώς και τα σήματα Rin,Gin,Bin από το kbd_decoder. Αρχικά, υπολογίζουμε την τιμή του χρώματος που πρέπει να έχουν τα pixel που θα εμφανιστούν. Αν ενεργοποιηθεί κάποιο από τα σήματα Rin,Gin,Bin, αυτό προστίθεται ή αφαιρείται στον τόνο του χρώματος.Τέλος, τροφοδοτούμε την οθόνη με τις τιμές που υπολογίσαμε, μόνο αν βρισκόμαστε στο display_area και αν το pixel του shifter που αντιστοιχεί στην αναπαράσταση του ψηφίου είναι 1. Διαφορετικά, στέλνουμε 0 στα {R,G,B}.

Στο **module bonus** υλοποιούμε την λειτουργία flash.Όταν πατάμε το κουμπί f, έρχεται μια θετική ακμή στο αντίστοιχο **flag ενεργοποιώντας ή απενεργοποιώντας την δυαδική κατάσταση f_state**. Το flash υλοποιείται με έναν counter που αυξάνει τιμή με κάθε pulse του vsync. Αυτό γίνεται γιατί το vsync είναι αρκετά αργό, ώστε να μην μας επιβάλλει να φτιάξουμε νέο ρολόι. Έτσι, έχουμε μια μέτρηση vsyncs μέχρι το 127. Αν το f_state είναι ενεργοποιημένο, τότε για τα 64 πρώτα vsync pulses έχουμε ενφάνιση χαρακτήρα και για τα επόμενα 64 στέλνουμε στα r,g,b 0. Έτσι δημιουργείται η αίσθηση του flash.

Περιγραφή Λειτουργίας

Αρχικά δημιουργούμε το ρολόι μας με συχνότητα λειτουργίας 25MHz. Αντλούμε τα scancodes από την επικοινωνία με το πληκτρολόγιο. Σε κάθε παλμό του ρολογιού μας, έχουμε σήμα για το αν έχουμε νέο scancode (δηλαδή, αν έχουμε πατήσει και αφήσει κάποιο πλήκτρο) καθώς και το scancode αυτό. Αν έχουμε καινούριο scancode, αυτό αποκωδικοποιείται, στέλνοντας τα σήματα αυτά στον **iterator** και στον **colorizer.**

Σε κάθε παλμό του ρολογιού, ελέγχουμε αν έχουμε νέο scancode, και αν έχουμε τροφοδοτούμε κατάλληλα τα σήματα εξόδου r,g,b,char_enable,starting_address.

Παράλληλα, δημιουργούμε τον συγχρονισμό με την οθόνη: Σε κάθε παλμό του ρολογιού, αυξάνεται ο pixel_counter μέχρι να φτάσει στην τιμή 800, όπου μηδενίζεται. Σε κάθε μηδενισμό του pixel_counter, αυξάνεται ο line_counter, μέχρι να φτάσει την τιμή 449 όπου και μηδενίζεται. Ανάλογα με την τιμή των σημάτων αυτών, στέλνουμε τα hsync και vsync στην οθόνη, συνυπολογίζοντας πάντα και τα porches. Όταν τα σήματα αυτά φτάσουν στην μέση της οθόνης ενεργοποιούμε το σήμα display_area, σήμα που παραμένει ενεργό όσο θα είμαστε στα 16x8 pixel της μέσης της οθόνης, όπου και θα εμφανιστούν και οι χαρακτήρες μας. Ταυτόχρονα, η αρτηρία line παίρνει τιμή την κάθετη απόσταση από το top_border που ξεκινάει το display_area.

Συνεπώς, όσο βρισκόμαστε στην ίδια γραμμή του display_area, η τιμή του line θα παραμένει η ίδια, τροφοδοτώντας τον iterator που τελικά θα παράξει την διεύθυνση της

μνήμης μαζί με το starting address ,έτσι ώστε να βγουν τα δεδομένα από την μνήμη σειριακά, σε κάθε χτύπο ρολογιού.

Με κάθε αλλαγή γραμμής και ενεργοποίησης του display_area, θα αλλάζει και η διευθυνσιοδότηση της μνήμης, έτσι ώστε να εμφανιστούν όλοι οι χαρακτήρες, ξεκινώντας από πάνω αριστερά μέχρι να φτάσουμε κάτω δεξία.

Τέλος, τα δεδομένα της μνήμης πάνε στον colorizer, όπου και αποφασίζεται το χρώμα που θα έχουν, ανάλογα με την τιμή που υπάρχει στην αρτηρία δεδομένων της μνήμης, και τον συνδιασμό χρωμάτων που επιλέξαμε.