**ΕΡΓΑΣΙΑ ΕΞΑΜΗΝΟΥ ΕΤΟΣ 2022 – 2023**

**ΣΙΔΕΡΗΣ ΣΤΥΛΙΑΝΟΣ**

**ΕΤΟΣ ΣΠΟΥΔΩΝ 40**

**ΑΜ: 1075217**

Σκοπός της εργασίας του εξαμήνου είναι να σχεδιαστεί ένα σύστημα εισαγωγής και ανίχνευσης συγκεκριμένων χαρακτήρων από το πληκτρολόγιο, απεικόνιση των χαρακτήρων στο κέντρο μια οθόνης και παράλληλα κίνηση του ρομποτικού βραχίονα του εργαστηρίου σε κατάλληλες θέσεις μέσο σέρβοκινητήρα. Παρακάτω θα γίνει μια περιγραφή των modules που σχεδιάστηκαν που αποτελούν τα κομμάτια του τελικού top module και σε κάθε ένα θα επισημάνουμε σημεία σημαντικού ενδιαφέροντος .Η περιγραφή θα γίνει με την ροή που έχει η πληροφορία και πως αυτή αποκωδικοποιείτε και παράγονται σήματα οπτικοποίησης και παραγωγής PWM για κίνηση ρομποτικού βραχίονα.

Στα scan code που μας ενδιαφέρουν με βάση την εκφώνηση της άσκησης έχουμε προσθέσει 5 ακόμα, τα τελευταία στον πίνακα .

Πίνακας 1

|  |  |  |
| --- | --- | --- |
| Πλήκτρο | Scan code | Χρήση |
| F | 2B | Βραχίονας πλήρες αναδιπλούμενος |
| Q | 15 | Βραχίονας επεκταμένος κατά 1/4 |
| H | 33 | Βραχίονας επεκταμένος κατά 1/2 |
| X | 22 | Βραχίονας σε πλήρη επέκταση |
| R | 2D | Κόκκινο χρώμα απεικόνιση οθόνη |
| G | 34 | Πράσινο χρώμα απεικόνιση οθόνη |
| B | 32 | Μπλε χρώμα απεικόνιση οθόνη |
| O | 44 | Άσπρο χρώμα απεικόνιση στην οθόνη |
| D | 23 | Αναβοσβήνει ο χαρακτήρας στην οθόνη |

**Module kbd\_protocol**

Το module αυτό χρησιμοποιείτε για να λάβει τα scan codes που εισάγουμε από το πληκτρολόγιο .Το πληκτρολόγιο είναι κατασκευασμένο με τρόπο ώστε καθώς μας δίνει σειριακά τα bit των scan code ,μας δίνει και έναν clock ανάλογο της ταχύτητας ροής των εξερχόμενων δεδομένων έτσι ώστε να έχουμε τον απόλυτο έλεγχο της ροής αυτής.

Έχοντας ρολόι λειτουργείας τα 100MHz στο FPGA μας και ένα αργό ρολόι του πληκτρολογίου, θα κάνουμε δειγματοληψία τού αργού με το γρήγορο. Καθώς ανιχνεύουμε τον χαρακτήρα 0xF0 τότε ένα σήμα ενεργοποιείτε και τα επόμενα 11 bit είναι το scan code του χαρακτήρα πού πατήσαμε περικλειόμενο από τα bit start ,stop και parity.Όταν η ισοτιμία μας είναι σωστή ένα flag και μαζί με το scan code αποτελούν την έξοδο του module.

**Module scancode\_decoder**

Το λαμβανόμενο scan codeπρέπει νααποκωδικοποιηθεί σε σήματα που εμείς μπορούμε να χειριστούμε καλύτερα και σε όσο δυνατών λιγότερα χωρίς πλεονασμούς. Αυτό το module παίρνει σαν είσοδο το scan code και το flag από **module kbd\_protocol** και αν το scan code είναι των χαρακτήρων F,Q,H,X τότε στην έξοδο μας δίνει μια διεύθυνση(start\_adress\_out) και ένα σήμα char\_enable.Η χρήση της διεύθυνσης δίνετε στο **module correct-line** που θα εξηγηθεί παρακάτω.

Αν όμως το scan code είναι των χαρακτήρων R,G,B,O,D τότε θα βγάλουμε στην έξοδο τιμές που θα βοηθήσουν το module χρωματισμού που θα δούμε παρακάτω.Tο char\_enable σε αυτή την περίπτωση δεν ενεργοποιείτε.

Αν το scan code δεν ανήκει σε κανέναν από τούς χαρακτήρες του πίνακα 1 δεν γίνετε κάποια αλλαγή στον αποκωδικοποιητή και δεν ενεργοποιείτε το char\_enable.Έτσι καμία αλλαγή δεν γίνετε στο υπόλοιπο κύκλωμα .

**Module vgasync**

Σε αυτό το module γίνεται ο συγχρονισμός της οθόνη. Χρειαστήκαμε έναν διαιρετή συχνότητας για να κατεβάσουμε το ρολόι στην συχνότητα **pixel clock 25MHz** . Έχουμε δύο counter ένας για γραμμές και ένας για ποιο pixel στην γραμμή. Έτσι με αυτό τον τρόπο με τον συνδυασμό αυτόν των δύο ξέρουμε ανά πάσα στιγμή σε ποιο pixel της οθόνης βρίσκεται η σάρωση. Οριοθετήσαμε τα όρια των περιοχών με βάση τις δοθέντες τιμές . Με απλό υπολογισμό οριοθετήσαμε το νοητό τετράγωνο(**display\_area**) που θα εμφανίζεται ο χαρακτήρας. Υπολογίσαμε τις χρονικές στιγμές που τα σήματα **hsync, vsync** θα είναι στο λογικό 1 ή 0.

Η τελευταία λειτουργεία αυτού του module είναι να υπολογίσει σε ποια γραμμή του display\_area είμαστε σε κάθε στιγμή. Κάνοντας μια αφαίρεση της γραμμής(counter γραμμών) που είμαστε μείον το πάνω άκρο του **display\_area** παίρνουμε:

1. Έναν αρνητικό αριθμό αν cnt\_line < πάνω σύνορο **display\_area.**
2. Έναν θετικό αριθμό μικρότερο 16.
3. Έναν θετικό αριθμό μεγαλύτερο 16.

Όμως μόνο στην περίπτωση 2 ενεργοποιείτε το σήμα **display\_area** έτσι δεν μας απασχολεί το αποτέλεσμα στις περιπτώσεις 1 και 3.η τιμή της γραμμής που είμαστε στο display area βγαίνει στην έξοδο line.

Τις εξόδους display\_area και lineθα τις χρησιμοποιήσουμε σε συνεργασία με το start\_adress\_out που παίρνουμε σαν έξοδο από το προηγούμενο module( scancode\_decoder **)** για να δοθεί στο επόμενο module που εξηγείται αμέσως μετά.

**Module correct\_line**

Τώρα αφού εξηγήσαμε τα σήματα start\_adress\_out (από scan\_decoder)και display\_area και cnt\_line (από vgasync)μπορούμε να περιγράψουμε αυτό το module.Με αρχική διεύθυνση το start\_adress\_out και με offset το line περνούμε τα περιεχόμενα της ανάλογης θέσης μνήμης που θα περιγράψουμε παρακάτω.Πχ αν έχουμε scan code του χαρακτήρα Q το start\_adress\_out είναι το 010000 και προσθέτουμε σε αυτό το line(start\_adress\_out + line).

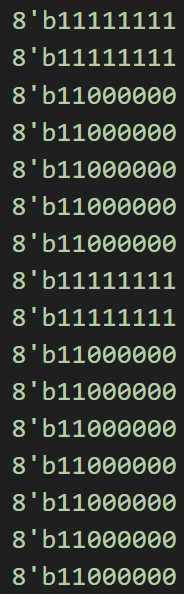
Όπως είναι λογικό πρέπει να είναι και ενεργό το σήμα display\_area αλλιώς η έξοδος του module θα είναι 0.

**Module myRom**

Εδώ έχουμε μια μνήμη. Έχουμε αποθηκεύσει στην μνήμη την παράσταση των χαρακτήρων.

Όπου θα θέλουμε μαύρο χρώμα στην οθόνη υπάρχει στην μνήμη 0 και όπου άσπρο χρώμα.

Στην παρακάτω φωτογραφία δίνετε ένα παράδειγμα για το γράμμα F.



Όταν ανιχνεύσουμε κάποιο scan code που ανήκει στους χαρακτήρες του πίνακα 1 τότε παίρνουμε μια έξοδο char\_enable από το module scancode\_decoder όπως είχαμε περιγράψει παραπάνω. Αυτό το σήμα αν είναι στο λογικό 1 τότε η μνήμη δίνει έξοδο που αναλογεί στον χαρακτήρα που πατήσαμε , αλλιώς δίνει έξοδο 000000 που παρακάτω αποκωδικοποιείτε σε μαύρο χρώμα.

**Module piso**

Σχεδιάσαμε καταχωρητή parallel input serial output.Περνούμε από την μνήμη τα δεδομένα και σειριακά όπως απαιτείται τα δίνουμε στο παρακάτω module που με βάσει τα μπιτ που λαμβάνει από παρασκευάζει τα διανύσματα R ,G, B .

Αν δεν είμαστε στο area display τότε δίνει σειριακά μηδενικά(αποτέλεσμα μαύρο χρώμα).