Αναφορά Δεύτερης εργαστηριακής άσκησης

HPY 419 - HPY 608 - Ανάπτυξη Εργαλείων CAD για Σχεδίαση Ολοκληρωμένων Κυκλωμάτων 2022- 2023

ΑΙΚΑΤΕΡΙΝΗ ΤΣΙΜΠΙΡΔΩΝΗ:2018030013

Εισαγωγή:

Σκοπός αυτού του προγράμματος είναι να φτιαχτεί το netlist ενός συστήματος που αποτελείται από components. Πιο συγκεκριμένα στο πρόγραμμα αυτό φτιάχνεται ένα κύκλωμα ενός πλήρους αθροιστή N-bit.Το πρόγραμμα ζητά από το χρήστη να εισάγει τον αριθμό των bit που θα χρησιμοποιηθούν στο πλήρες κύκλωμα αθροιστή, επικυρώνει την είσοδο του χρήστη και, στη συνέχεια, προχωρά στη δημιουργία μιας netlist για το κύκλωμα.

Το πρόγραμμα ορίζει τρεις δομές: Component, netlist και netlist_equals. Η δομή Component αποθηκεύει τα χαρακτηριστικά κάθε στοιχείου, συμπεριλαμβανομένου του μοναδικού αναγνωριστικού(**id**), του τύπου του στοιχείου(**type**), των θυρών εισόδου(**inputs**), των θυρών εξόδου(**outputs**) και έναν pointer που δείχνει ποιο είναι το επόμενο στοιχείο που ακολουθει(***next**).
Η δομή του netlist περιέχει μια σειρά στοιχείων(**components**) και τον συνολικο αριθμό των στοιχείων στη λίστα δικτύου(**num_components**).
Τέλος, η δομή netlist_equals χρησιμοποιείται για να εκφράσει τον τελεστή '=' όπως μας ζητήθηκε στην εκφώνηση, στην ουσία παίρνει ένα input και το αντιστοιχεί σε ένα output .

Οι δομές αυτές χρησιμοποιούν πίνακες οι οποίοι είναι δύο διαστάσεων ώστε να μπορεί να αποθηκεύει περισσότερες από μία τιμές για κάθε πεδίο. Για παράδειγμα η δομή για τα inputs μοιάζει έτσι **char inputs[MAX_INPUTS][MAX_CHARACTERS]**; .Έχει τη δυνατότητα να αποθηκεύει τόσες εισόδους όσες μας λέει το Max_inputs και επειδή οι είσοδοι αποθηκεύονται ως χαρακτήρες στη δεύτερη διάσταση βάζουμε πόσοι χαρακτήρες μπορούν να αποθηκευτούν για κάθε είσοδο. Άρα όταν εμείς θα βάλουμε είσοδο A0 ,B0,Cin , το A0 θα αποθηκευτεί ως πρώτο στοιχείο του πίνακα inputs το B0 ως δεύτερο και το Cin ως τρίτο. Αντίστοιχα συμβαίνει με τους πίνακες output, ώστε το πρόγραμμα να κατασκευάζει netlist για στοιχεία που έχουν παραπάνω από μία είσοδο,έξοδο.

Functions:

add_component: Αυτή η μέθοδος κατασκευάζει και προσθέτει ένα νέο στοιχείο (**component)** στο netlist.Παίρνει τα ακόλουθα ορίσματα **netlist** ένας δείκτης στη δομή δεδομένων netlist ,**head** ένας δείκτης στην κεφαλή της συνδεδεμένης λίστας στοιχείων,**type** ένας πίνακας χαρακτήρων που αντιπροσωπεύει τον τύπο του νέου στοιχείου,**inputs** ένας πίνακας χαρακτήρων 2D που αντιπροσωπεύει τις θύρες εισόδου του νέου στοιχείου,**output**: ένας πίνακας χαρακτήρων 2D που αντιπροσωπεύει τις θύρες εξόδου του νέου στοιχείου
Η συνάρτηση αρχικά εκχωρεί μνήμη για το νέο στοιχείο χρησιμοποιώντας τη συνάρτηση malloc. Στη συνέχεια, ορίζει το αναγνωριστικό για το νέο στοιχείο ως τον τρέχοντα αριθμό στοιχείων στη λίστα δικτύου. Αντιγράφει τον τύπο, τις εισόδους, την έξοδο στα αντίστοιχα πεδία στο new_component χρησιμοποιώντας τη συνάρτηση strcpy.

Το new_component προστίθεται στην αρχή της συνδεδεμένης λίστας ορίζοντας το επόμενο πεδίο του στην τρέχουσα κεφαλή και ενημερώνοντας την κεφαλή ώστε να δείχνει στο

new_component, αυτό το κάνει ώστε να γνωρίζει με ποια σειρά είναι συνδεδεμένο κάθε στοιχείο. Στη συνέχεια, καλείται η συνάρτηση addComponentToTheNetlist για να προσθέσει το νέο στοιχείο στη λίστα δικτύου.
addComponentToTheNetlist:
Η συνάρτηση addComponentToTheNetlist είναι υπεύθυνη για την προσθήκη ενός στοιχείου σε μια λίστα δικτύου. Έχει δύο ορίσματα: έναν δείκτη στο στοιχείο που θα προστεθεί (Component* c) και έναν δείκτη στη λίστα δικτύου όπου θα προστεθεί το στοιχείο (netlist* netlist).

Η συνάρτηση ορίζει ένα δείκτη i ίσο με τον τρέχοντα αριθμό στοιχείων στη λίστα δικτύου (netlist->num_components). Στη συνέχεια, προσθέτει το στοιχείο στη λίστα διαδικτύου αντιγράφοντας το στον πίνακα στοιχείων της netlist χρησιμοποιώντας τον δείκτη i (netlist->components[i] = *c). Τέλος, αυξάνει τον αριθμό των στοιχείων στη λίστα δικτύου κατά 1 (netlist->num_components++).

Αυτή η συνάρτηση καλείται από τη συνάρτηση add_component η οποία δημιουργεί ένα νέο στοιχείο και το προσθέτει στη συνδεδεμένη λίστα στοιχείων και στη λίστα δικτύου χρησιμοποιώντας το addComponentToTheNetlist.

Προσθέτοντας το στοιχείο στη λίστα δικτύου, περιλαμβάνεται πλέον στη λίστα δικτύου και οι είσοδοι και οι έξοδοι του μπορούν να συνδεθούν με άλλα στοιχεία σε αυτη.
CreateNetlistForFullAdder:

Η συνάρτηση CreateNetlistForFullAdder δημιουργεί μια netlist για ένα πλήρες κύκλωμα αθροιστή με **N** bit. Η συνάρτηση παίρνει τρία ορίσματα: **N** που είναι ο αριθμός των πλήρων αθροιστών, **fp** που είναι δείκτης προς το αρχείο όπου θα γραφτεί η netlist και **netlist_equals_arr** που είναι ένας πίνακας δομών για τις ισοδυναμίες('=')netlist. Η συνάρτηση αρχικοποιεί μια νέα δομή netlist και ορίζει τον αριθμό των στοιχείων σε 0. Επίσης, αρχικοποιεί μια νέα κεφαλή δείκτη(*head) Component σε NULL.

Στη συνέχεια, η συνάρτηση εισάγει έναν βρόχο από το 0 έως το N - 1 και για κάθε επανάληψη του βρόχου, δημιουργεί τις απαραίτητες εισόδους, εξόδους για κάθε πλήρη αθροιστή και τα προσθέτει στη λίστα δικτύου χρησιμοποιώντας τη συνάρτηση add_component.Επίσης χρησιμοποιεί και κάποια ορίσματα(input_carry,output_carry) τα οποία έχουν οριστεί στην αρχή ώστε να διαχειριστεί σωστά το κρατούμενο διοτι το κρατούμενο του προηγούμενου αθροιστή πρέπει να είναι μία είσοδος του επόμενου. Αφού προσθέτει τα διάφορα component μετά χρησιμοποιούμε την δομή **netlist_equals_arr** , όπου είναι η δομή για τις ισοδυναμίες ώστε να αποθηκεύσει σωστά ποιες θα είναι οι έξοδοι του συστήματος (πχ S5=U5_S) .
Τέλος, η συνάρτηση γράφει τη netlist στο αρχείο που δείχνει το **fp** χρησιμοποιώντας τη συνάρτηση fprintf.
Η netlist είναι γραμμένη με μία συγκεκριμένη μορφή που αναφέρεται στην εκφώνηση, με κάθε γραμμή να περιέχει ένα στιγμιότυπο στοιχείου και τις εισοδους του και το U<number> μπροστα.

Συνοπτικά, η συνάρτηση CreateNetlistForFullAdder δημιουργεί μια netlist για ένα πλήρες κύκλωμα αθροιστή με N bit, δημιουργεί τα απαραίτητα στοιχεία για κάθε πλήρη αθροιστή, τα προσθέτει στη λίστα δικτύου και εγγράφει τη λίστα δικτύου σε ένα αρχείο.

Σημειώσεις:

Το πρόγραμμα περιέχει όπως αναφέρετε τις παραπάνω εσωτερικές δομές δηλαδή τους πίνακες για τα inputs and outputs περιέχει τη δομή για τα component και τη δομή για netlist όπως και τη δομή για την ισοδυναμία. Αυτές είναι οι δομές που αντιλήφθηκα ότι χρειάζονται βέβαια στα επόμενα project μπορεί να χρειάζονται παραπάνω δομές ή να χρειαστούν επεκταθούν οι υπάρχουσες.

Το πρόγραμμα μπορεί να χρησιμοποιηθεί για τη δημιουργία μιας netlist και για άλλα στοιχεία. Το πρόγραμμα ορίζει μια δομή Component που μπορεί να κρατήσει πληροφορίες για οποιονδήποτε τύπο ψηφιακής λογικής συνιστώσας, όπως πύλες AND, πύλες OR, πύλες XOR και ούτω καθεξής.

Για να δημιουργήσετε μια netlist για διαφορετικό τύπο στοιχείου, θα χρειαστεί να φτιαχτεί μία άλλη συνάρτηση όπως η **CreateNetlistForFullAdder** η οποία θα κάνει κάτι διαφορετικό.

Αποτελέσματα:

Η έξοδος είναι ένα αρχείο το οποίο περιέχει το Netlist με το format που ζητήθηκε
U<number> <component type> <list of inputs>
…..
S<number>=UXXX_S
COUT=UXXX_C

Π.χ για N=7

```
1   U0 Full_adder A0,B0,cin
2   U1 Full_adder A1,B1,U0_C
3   U2 Full_adder A2,B2,U1_C
4   U3 Full_adder A3,B3,U2_C
5   U4 Full_adder A4,B4,U3_C
6   U5 Full_adder A5,B5,U4_C
7   U6 Full_adder A6,B6,U5_C
8   S0=U0_S
9   S1=U1_S
10  S2=U2_S
11  S3=U3_S
12  S4=U4_S
13  S5=U5_S
14  S6=U6_S
15  Cout=U6_C
16
```

Αυτό το αποτέλεσμα όπου είναι και το αναμενόμενο βλέπουμε ότι ο πρώτος FULL ADDER παίρνει εισόδους A0 B0 Cin ο δεύτερος παίρνει A1 B1 και το κρατούμενο το προηγούμενου

U0_C και συνεχίζει αυτή συστοιχία στο τέλος βγάζει τις εξόδους με ‘=” και ο τελευταίος βγάζει και το Cout που είναι το carry out του συστήματος πού είναι το τελευταίο carry out του τελευταίου full adder δηλαδή του U6.

Σε περίπτωση που ο χρήστης δώσει παραπάνω bit από όσα θέλουμε τότε βγάζει μήνυμα invalid input και τερματίζει.