ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

19/05/2021

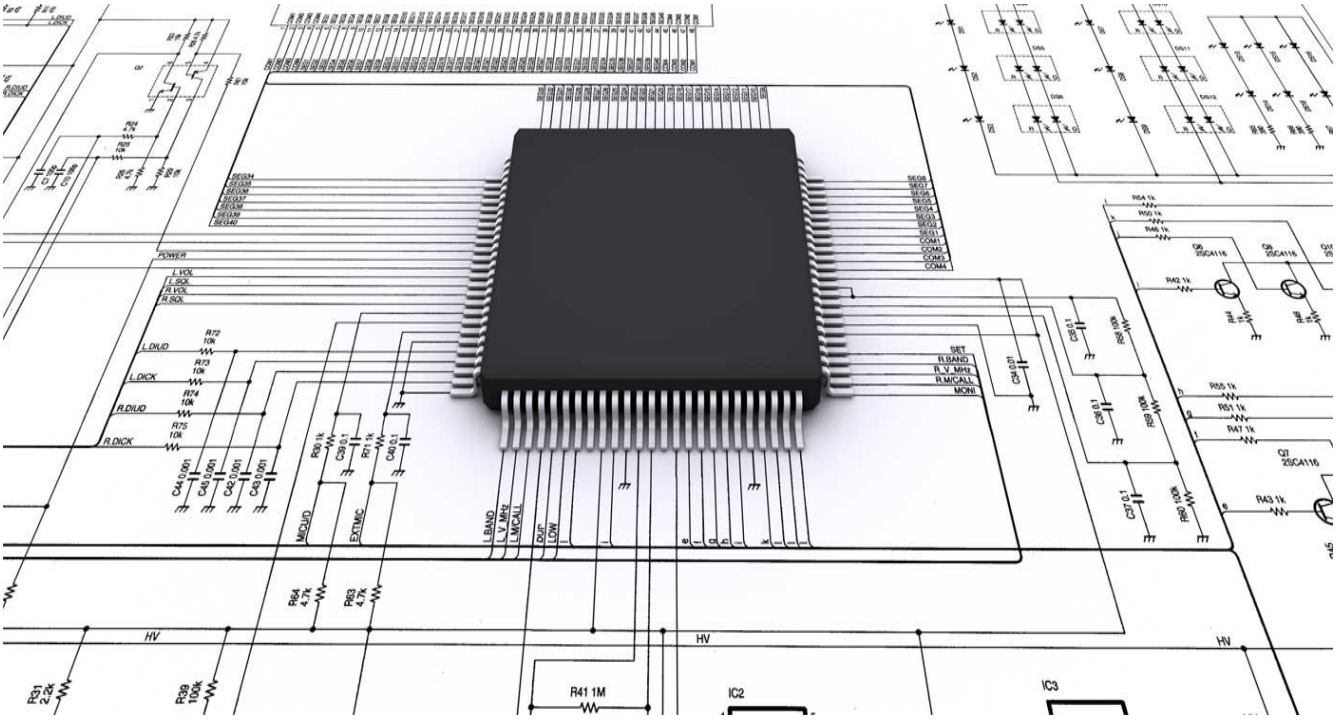
ΗΡΥ608/419-ΑΝΑΠΤΥΞΗ ΕΡΓΑΛΕΙΩΝ CAD ΓΙΑ ΣΧΕΔΙΑΣΗ ΟΛΟΚΛΗΡΩΜΕΝΩΝ ΚΥΚΛΩΜΑΤΩΝ

ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ 2021

ΑΣΚΗΣΗ 5
ΒΑΣΙΚΕΣ ΛΕΙΤΟΥΡΓΙΕΣ ΓΙΑ FORMAL VERIFICATION ΕΞΑΓΩΓΗ ΛΟΓΙΚΩΝ ΔΟΜΩΝ ΠΥΛΩΝ ΑΠΟ ΣΥΝΔΕΣΜΟΛΟΓΙΑ ΤΡΑΝΖΙΣΤΟΡ

ΑΝΑΦΟΡΑ

Μυλωνάκης
Εμμανουήλ
2015030079



ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Υλοποίηση - Περιγραφή.

Για την πέμπτη εργαστηριακή άσκηση, υλοποιήθηκε μία από τις βασικές λειτουργίες της μεθόδου Formal Verification. Διαβάζοντας ένα επίπεδο netlist από CMOS transistor, αναζητούσαμε ύπο-δομές ισοδύναμες με λογικές πύλες (NOT, NOR και NAND) ώστε να εξαχθεί το αντίστοιχο netlist πυλών.

Η κύρια δομή δεδομένων που χρησιμοποιήθηκε για την διεκπεραίωση της άσκησης αυτής ήταν ένας πίνακας DAG[#CMOS][6] τύπου ακέραιος, όπου οι γραμμές του δημιουργούνται δυναμικά με βάση το πλήθος των transistor του netlist και έχει σταθερό αριθμό 6 στηλών. Κρατάει τις εξής πληροφορίες από το netlist:

- **1η Στήλη:** Χαρακτήρας 'P' ή 'N' για το αν πρόκειται για PMOS ή NMOS transistor αντίστοιχα.
- **2η Στήλη:** Τον αριθμό του κόμβου-πύλης για το transistor. Το πρώτο pin που αναγράφεται στο netlist
- **3η Στήλη:** Τον αριθμό του κόμβου-[Source/Drain] για το transistor. Το δεύτερο pin που αναγράφεται στο netlist
- **4η Στήλη:** Τον αριθμό του κόμβου-[Source/Drain] για το transistor. Το τρίτο pin που αναγράφεται στο netlist
- **5η Στήλη:** Την τιμή ενός offset που εξυπηρετεί στην λειτουργικότητα του αλγορίθμου. Αρχικοποιείται με την τιμή -1.
- **6η Στήλη:** Τον αριθμό UX που αναγράφεται στο netlist. Χρησιμοποιείται μόνο κατά την εκτύπωση αποτελεσμάτων στο αρχείο εξόδου.

Η βασική ιδέα του αλγορίθμου είναι για κάθε ένα transistor του netlist να βρεθεί σε ποια από τις πύλες NOT, NOR ή NAND συμμετέχει. Έχουμε μία μεταβλητή offset η οποία αρχικοποιείται στην τιμή 1 και μόλις για το ένα αυτό transistor βρεθούν όλα τα υπόλοιπα που συμμετέχουν στην ίδια πύλη, αποθηκεύουμε στην παραπάνω δομή την τιμή του offset για τα transistor αυτά και το αυξάνουμε το offset κατά 1. Έτσι, δεν ελέγχουμε στην συνέχεια του αλγορίθμου transistors το οποία ήδη ανήκουν σε κάποια πύλη και κατά μία έννοια τα έχουμε ομαδοποιημένα (δεν πραγματοποιείται κάποιου είδους ταξινόμηση με βάση το offset) διότι με το να αυξάνεται μονίμως το offset έχουμε την πληροφορία ότι το transistor ανήκει στον $1^{ο}$ υπογράφο, $2^{ο}$ υπογράφο, $3^{ο}$ υπογράφο κοκ. Τέλος, λόγω της αρχικοποίησης με -1 στην $5^{η}$ στήλη του παραπάνω πίνακα, αν η τιμή αυτή δεν ενημερώθει από το offset τότε θα γνωρίζουμε ποιο ή ποια transistor δεν συμμετέχουν σε κάποια από τις πύλες.

ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ
ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ ΚΑΙ ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

Ψευδοκώδικας

```
func main:
start
offset = 1;
for i=0 in DAG[#CMOS][5]          //Για κάθε transistor του netlist

   if: DAG[i][5η στήλη] != -1     //Αν το τρανζιστορ ανήκει σε κάποια πύλη
      continue;                  //Επόμενη επανάληψη. Δεν ελέγχουμε για αυτό.

   if: is_in_gate_NOT(i, offset)   //Αν το τρανζιστορ ανήκει στην πύλη NOT.
      offset++;                  //Αύξησε offest κατα ένα.
      continue;                  //Επόμενη επανάληψη.

   if: is_in_gate_NOR(i, offset)   //Αν το τρανζιστορ ανήκει στην πύλη NOR.
      offset++;                  //Αύξησε offest κατα ένα.
      continue;                  //Επόμενη επανάληψη.

   if: is_in_gate_NAND(i, offset)   //Αν το τρανζιστορ ανήκει στην πύλη NAND.
      offset++;                  //Αύξησε offest κατα ένα.
      continue;                  //Επόμενη επανάληψη.
end


func boolean is_in_gate_NOT(i, offset):
start
   if: DAG[i][1η στήλη] == 'P'       //Αν είναι PMOS
      Check below conditions:
      - Find NMOS in series connection with common gate. Otherwise return false.;
      - Source of PMOS is VCC node. Otherwise return false.;
      - Source of NMOS is GND node. Otherwise return false.;
   else if: DAG[i][1η στήλη] == 'N'  //Αν είναι NMOS
      Check below conditions:
      - Find PMOS in series connection with common gate. Otherwise return false.;
      - Source of PMOS is VCC node. Otherwise return false.;
      - Source of NMOS is GND node. Otherwise return false.;

   Input = Node-Gate;
   Output= Node-Drain;
   Update DAG with offset.
   return true;
end


Σημειογραφία [X/Y]
X: Αφορά την συνάρτηση is_in_gate_NOR
Y: Αφορά την συνάρτηση is_in_gate_NAND

func boolean is_in_gate_[NOR/NAND](i, offset):
start
   if: DAG[i][1η στήλη] == ['N'/'P']       //Αν είναι PMOS
      Check below conditions:
      - Find [NMOS/PMOS] transistors in parallel connection. Otherwise return false.;
      - Find [PMOS/NMOS] transistor in series conection with the previous [NMOS/PMOS]. Otherwise return false.;
      - Keep tracking in series connected [PMOS/NMOS]. Otherwise return false.;
      - Is #PMOS == #NMOS. Otherwise return false;
      - Source of every [NMOS/PMOS] is [GND/VCC] node. Otherwise return false.;
      - Source of last [PMOS/NMOS] in series is [VCC/GND] node. Otherwise return false.;

      Inputs = Gates of NMOS or Gates of PMOS;
      Output = [Drain of NMOS / Drain of PMOS];
      Update DAG with offset.
      return true;

   return false;
end
```

Ο αλγόριθμος αυτός έχει το πλεονέκτημα ότι δεν χρειάζεται να διαβάζουμε το αρχείο βιβλιοθήκης και να κάνουμε συγκρίσεις για ισοδύναμους υπογράφους. Ο παραδοτέος κώδικας υποστηρίζει μόνο τις τρείς πύλες της εργαστηριακής άσκησης αλλά ιδέα του μπορεί να υποστηρίξει αρχείο με πολλαπλές πύλες και πολλαπλές εισόδους. Πιο συγκεκριμένα, εάν για κάποιο transistor του netlist θέλουμε να ελέγξουμε αν ανήκει στην πύλη NOR, πρώτα απ'ολα πρέπει να είναι τύπου NMOS, διαφορετικά όπως φαίνεται και στον ψευδοκώδικα επιστρέφεται false. Στην πύλη NOR τα NMOS τρανζίστορ συμμετέχουν σε παράλληλη συνδεσμολογία με τους ακροδέκτες πηγής στην γείωση και τους ακροδέκτες εκροής προς στην έξοδο της. Άρα αναζητώντας και βρίσκοντας όλα τα παράλληλα NMOS τρανζιστορ έπειτα είμαστε σε θέση να βρούμε όλα τα εν σειρά PMOS με το τελευταίο να μας οδηγεί σε κόμβο τάσης. Με άλλα λόγια, εγγυημένα θα ξεκινήσουμε από τον κόμβο γείωση βρίσκοντας όλα τα παράλληλα NMOS και θα καταλήξουμε στον κόμβο τάσης ακολουθώντας τα εν σειρά PMOS. Το πλήθος τον ΝΜΟΣ και PMOS πρέπει να είναι ίσο και μας δίνει επίσης την πληροφορία για το αν έχουμε πύλη 2 ή 3, κοκ εισόδων. Αντρίστροφη λογική για την πύλη NAND.

Παραδοτέα.

Στα παραδοτέα αρχεία, το FA.txt αφορά έναν Full Adder και είναι το αρχείο εισόδου μας. Παράγονται δύο αρχεία εξόδου, το Gate_Netlist.txt που αφορά netlist πυλών και το Subgraphs.txt που φαίνονται όλοι οι υπογράφοι του κυκλώματος σε flat επίπεδο.