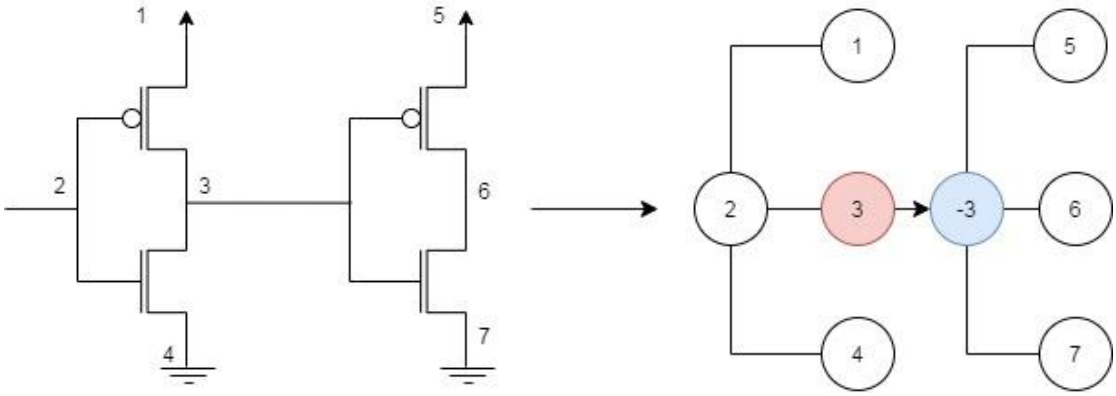
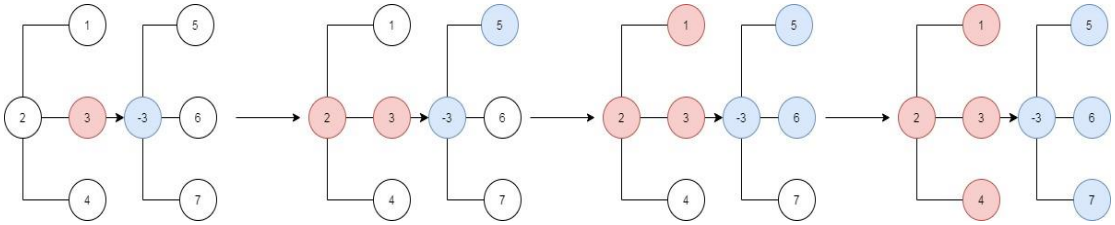
ΑΝΑΦΟΡΑ 3ου set

Κουρκουλος Αγγελος ΑΜ:2017030111

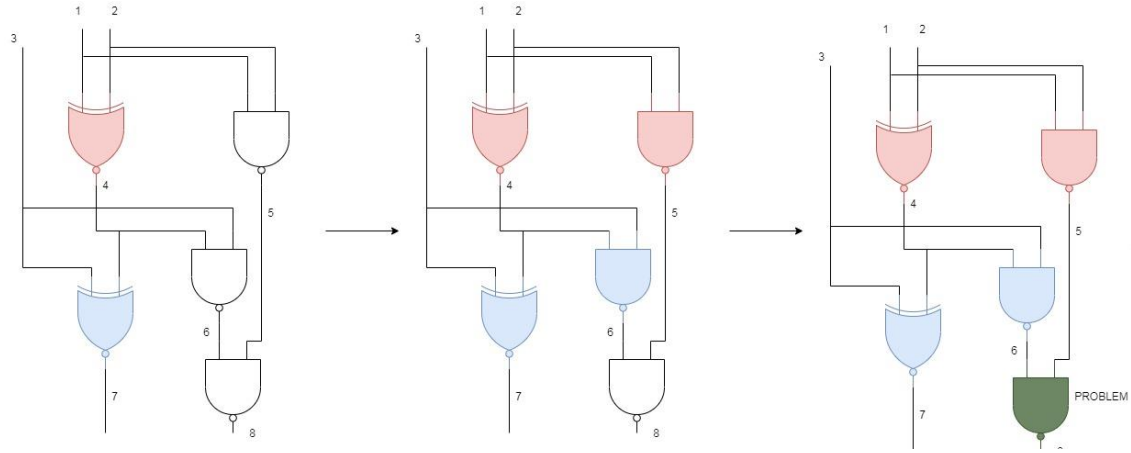
Σε αυτή την άσκηση σκοπός ήταν να φτιάξουμε ένα αρχείο που να περιγραφεί ένα 8bito RCA στον οποίο έπρεπε να βρούμε τα κατάλληλα σημεία δηλαδη τους κομβους όπου αν τον σπάσουμε σε ένα από αυτούς θα έχουμε 2 ξεχωριστά κυκλώματα. Για την επίλυση του προβλήματος ακολουθήθηκε η παρακάτω ακολουθία βημάτων. Αρχικά δημιουργήθηκε ο RCA 8bit στο αρχείο εισόδου file2 το οποίο είχε την αντίστοιχη μορφή του αρχείου εισόδου της εργαστηριακής άσκηση 3 καθώς χρησιμοποιήθηκε και το αρχείο βιβλιοθήκης πυλών της προηγούμενης άσκησης. Έπειτα μέσω των συναρτήσεων initialize() και initializeLib() που είχαν υλοποιηθεί κατασκευαστήκαν τα παραπάνω αρχεία και αποθηκευτικά σε στατικους πίνακες ικανοποιητικού μεγεθους .Στη συνέχεια δημιουργήθηκε το επίπεδο netlist καθώς και τα inputs / outputs του κυκλώματος χρησιμοποιώντας τη συνάρτηση createFinalNet(). Από αυτό το σημείο και έπειτα ακολουθούν νέες συναρτήσεις που δημιουργήθηκαν αποκλειστικά για τη λύσει του προβλήματος μας . Το πρώτο πράγμα που χρειάστηκε είναι η συνάρτηση findPossibkeBreaks() η οποία παίρνει ως είσοδο το επίπεδο netlist και βρίσκει όλους τους κόμβους οι οποίοι είναι πιθανά καλά σημεία για να διαχωρίσουμε εκεί το κύκλωμα. Αυτο γίνεται επιλέγοντας τους κόμβους που είναι gate για κάποια transistor αλλά ταυτόχρονα είναι source η drain για κάποια αλλά. Έπειτα για κάθε ένα από αυτά τα σημεία δημιουργείτε ένας γραφος από τη συνάρτηση createGraph όπου κάθε γειτονικος κόμβος του κυκλώματος γίνεται γειτονικος κόμβος του γράφου Εκτός από τον εξεταζόμενο για σημείο διαχωρισμού κόμβο ο οποίος δημιουργεί ένα αντίγραφο του κατασκευάζει ακμή με αυτόν και γειτνιάζει με τους κόμβους που τον βλέπουν ως source η drain ενώ αφήνει τους γειτονικούς κόμβους που τον θεωρούν ως gate να γειτνιάζουν με το αντίγραφο. Επίσης ο αρχικός κόμβος χρωματιζεται ροζ ενώ το αντίγραφο βάφεται μπλε. Ακολουθεί παράδειγμα για το πώς θα εκτελούνταν ο αλγόριθμος με 2 πύλες NOT και κόμβος διάσπασης ο κόμβος 3 :



Μετά τη δημιουργία του γράφου χρησιμοποιούμε τη συνάρτηση coloreGraph ()η οποία παίρνει τον αρχικό γράφο και χρωματίζει κάθε κόμβου με βάση τον γειτονικό του χωρίς να λαμβάνει υπόψη την σύνδεση μεταξύ κόμβου διάσπασης και του αντιγράφου του. Σε περίπτωση που το υποκυκλωμα από τη μεριά του ροζ γράφου βαφτεί εντελώς ροζ ενώ το υποκυκλωμα από την μεριά του μπλε γράφου βαφτεί εντελώς μπλε τότε ξέρουμε ότι αυτός ο κόμβος είναι ένας αρκετά καλός κόμβος για να κόψουμε το κύκλωμα. Από την άλλη στην περίπτωση που τα χρώματα μπερδευτούν δηλαδή ένας ροζ κόμβος προσπαθήσει να αλλάξει το χρώμα ενός μπλε κόμβου σε ροζ η το αντίθετο τότε ξέρουμε ότι η υπόθεση μας για τον κόμβο διάσπασης είναι λάθος οπότε απορρίπτουν αυτό τον κόμβο. Ακολουθεί το παράδειγμα για το πως γεμίζουν οι κόμβοι με χρώμα :



Το παραπάνω παράδειγμα είναι μια σωστή υπόθεση για τον κόμβο διάσπασης μιας πύλης NOT . Ακολουθεί ένα παράδειγμα για μια λάθος υπόθεσης ενός full adder με κόμβο διάσπασης τον κόμβο 4:



Μόλις τελειώσουμε την διαδικασία για τους πιθανούς κόμβους διάσπασης και καταλήξουμε για το ποιοι κόμβοι είναι αυτοί που διασπανε επιτυχώς το κύκλωμα έρχεται η στιγμή να αποφασίσουμε για το ποιος από τους όλους είναι ο βέλτιστο . Για κάθε έναν λοιπόν βρίσκουμε ποιος έχει τη μικρότερη διαφορά από όλους τους κόμβους που βρίσκονται πριν και μετά από τον κόμβο διάσπασης δηλαδή ποιος είναι αυτός που βρίσκεται κεντρικότερο στο συνολικό κύκλωμα. Όταν βρούμε ποιος είναι ο καλύτερος κόμβος περνούμε ένα πίνακα με όλους τους ροζ και ένα με όλους τους μπλε κόμβους του γράφου του και τους ταξινομούμε . Έπειτα τους δίνουμε σαν ορισμα στην συνάρτηση που υλοποιήθηκε separateTheInpFileTo2() η οποία φτιάχνει 2 αρχεία όπου αποθηκευει την πληροφορία για τα netlist,τα inputs,τα outputs και τα rails για τα 2 υποκυκλωματα. Μόλις δημιουργηθούν τα 2 αρχεία τρέχουμε ένα παρόμοιο αλγόριθμο με το εργαστήριο 2 για να κάνουμε simulation για κάθε κύκλωμα χρησιμοποιώντας tesr_vectors που δημιουργούμε με το χέρι μέσα στο πρόγραμμα και επαληθεύουν τη λειτουργία τους.

Το αρχείο εισόδου που αντιπροσωπεύει τον 8bit RCA για αυτή την άσκηση είναι το :

```
MyLib.LIB

##RAILS

## INPUTS
1,2,3,9,10,16,17,23,24,30,31,37,38,44,45,51,
## OUTPUTS
7,14,21,28,35,42,49,56,57
## NETLIST
G1,XOR_2 ,IN,1,2,OUT,4
G2,NAND_2,IN,1,2,OUT,5
G3, XOR_2,  IN,3,4, OUT,7
G4,NAND_2,IN,3,4,OUT,6
G5,NAND_2,IN,6,5,OUT,8
G6,XOR_2 ,IN,9,10,OUT,11
G7,NAND_2,IN,9,10,OUT,12
G8, XOR_2,  IN,8,11, OUT,14
G9,NAND_2,IN,8,11,OUT,13
G10,NAND_2,IN,13,12,OUT,15
G11,XOR_2 ,IN,16,17,OUT,18
G12,NAND_2,IN,16,17,OUT,19
G13, XOR_2,  IN,15,18, OUT,21
G14,NAND_2,IN,15,18,OUT,20
G15,NAND_2,IN,20,19,OUT,22
G16,XOR_2 ,IN,23,24,OUT,25
G17,NAND_2,IN,23,24,OUT,26
G18, XOR_2,  IN,22,25, OUT,28
G19,NAND_2,IN,22,25,OUT,27
G20,NAND_2,IN,27,26,OUT,29
G21,XOR_2 ,IN,30,31,OUT,32
G22,NAND_2,IN,30,31,OUT,33
G23, XOR_2,  IN,29,32, OUT,35
G24,NAND_2,IN,29,32,OUT,34
G25,NAND_2,IN,34,33,OUT,36
G26,XOR_2 ,IN,37,38,OUT,39
G27,NAND_2,IN,37,38,OUT,40
G28, XOR_2,  IN,36,39, OUT,42
G29,NAND_2,IN,36,39,OUT,41
G30,NAND_2,IN,41,40,OUT,43
G31,XOR_2 ,IN,44,45,OUT,46
G32,NAND_2,IN,44,45,OUT,47
G33, XOR_2,  IN,43,46, OUT,49
G34,NAND_2,IN,43,46,OUT,48
G35,NAND_2,IN,48,47,OUT,50
G36,XOR_2 ,IN,51,52,OUT,53
G37,NAND_2,IN,51,52,OUT,54
G38, XOR_2,  IN,50,53, OUT,56
G39,NAND_2,IN,50,53,OUT,55
G40,NAND_2,IN,55,54,OUT,57
## TESTBENCH
## TEST_IN
52 ; 51 ; 45 ; 44 ; 38 ; 37 ; 31 ; 30 ; 24 ; 23 ; 17 ;16 ; 10 ; 9 ; 2 ; 1 ; 3
## TEST_OUT
7 ; 14 ; 21 ; 28 ; 35 ; 42 ; 49 ; 56 ; 57
## TEST_VECTORS
0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0 ; 0
## SIMULATE
## TEST_VECTORS
1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1
## SIMULATE
## TEST_VECTORS
1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 1 ; 0 ; 0
## SIMULATE
## TEST_VECTORS
1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1 ; 1
## SIMULATE
## END_TEST
## END SIMULATION
```

Τα test_In,test_out,test_Vector δεν έχουν καμια σημασία απλά χρειάστηκαν για να δοκιμαστεί στην αρχή ότι το συνολικό κύκλωμα λειτουργεί σωστά κάτι που δεν φαίνεται στο παραδοτέο .

Η δομη του αλγορίθμου που υλοποιήθηκε είναι :

```
define the arrays we need to store the information from file

initialization(…)                    //it stores the information from file in the appropriate array

define the arrays we need to store the information from Library

initializeLib(…)                     //it stores the information from Library in the appropriate array

define the arrays we need to store the converted information we will create from Library and file arrays



while(flag==1){  //// loop we need in case that the gates are not with the corect order in the starting netlist

      flag=0;

        int c=0;

        while(netlist[c]!= 0){  ////Loop for every gate

              if(logicgate[c]==NOT){  //// for every case of Lgate call the func createFinalNet to add the gate to final
                    createFinalNet(…);                                                                    //// netlist
              }
```

```
            else if(logicgate[c]==NOR_2){
                    createFinalNet(…)   ;
             }
             else if(logicgate[c]==NAND_2){
                    createFinalNet(…)   ;
             }
             else if(logicgate[c]==XOR_2){
                    createFinalNet(…)   ;
             }
             else if(logicgate[c]==NMOS || PMOS){ ///add the transistor to final netlist
                    createFinalNetAddMos(…)     ;
             }
        }
}

define the arrays we need to store the information from Graph

findPossibleBreak() ; ////finds the the nodes which are possible to be a good point to break the circuit

for(every possibleBreak){
    Graph gr=createGraph(possibleBreak);
    int validBreakCheck=colorGraph(gr)
    if(validBreakCheck==1){ //if coloures are not mixed
        validBreak[i]= possibleBreak
        i++;
    }
}

while(accuracyofvalidBreak[i]!=0){ // finds the best break point
     if(accuracyofvalidBreak[i]<min){
         min=accuracyofvalidBreak[i];
         bestbreak=validBreak[i];
     }
     i++;
    }

separateTheInpFileTo2(bestBreak)  //creates the 2 files

for(every File){
    initialization(File);
    manual_test_vector={..};
    while (testvector[i] != 0){     //Loop for every testVector
          while(check==1){     //Loop that run until nothing changed so we done
               check=0;
               j=0;
                    while(transistor[j]!=0){  // Loop for every transistor

                      if(transistor[j]==PMOS && newNode[netlist[j][0]]==0 ){
                      …
                      check=1;
                       }
                      else if(transistor[j]==PMOS && newNode[netlist[j][0]]==1 ){
                     …
                     check=1;
                     }
                      else if(transistor[j]==NMOS && newNode[netlist[j][0]]==0 ){
```

```
                    ...
                    check=1;
                    }
                    else if(transistor[j]==NMOS && newNode[netlist[j][0]]==1 ){
                    ...
                    check=1;
                    }
                   }
          }
    i++;
    }
}
```

Οι κόμβο ο που βρέθηκαν ότι είναι ικανοι για τη διάσπαση του κυκλώματος στο κυκλομα του RCA οι οποιοι είναι το Cin για κάθε fullAdder καθώς και ο καλύτερος κόμβος διάσπασης είναι:

```
the break at the node 28 is valid
 the break at the node 58 is valid
 the break at the node 88 is valid
 the break at the node 118 is valid
 the break at the node 148 is valid
 the break at the node 178 is valid
 the break at the node 208 is valid
 the best break is at node 118
```

Τα 2 αρχεία που δημιουργήθηκαν λόγο της διάσπασης του κυκλώματος από την συνάρτηση separateTheInpFileTo2() είναι:

NewFile1:

```
##RAILS
VCC 1 ; 7 ; 13 ; 19 ; 25 ; 31 ; 37 ; 43 ; 49 ; 55 ; 61 ; 67 ; 73 ; 79 ; 85 ; 91 ; 97 ; 103 ; 109 ; 115 ;
GND 6 ; 12 ; 18 ; 24 ; 30 ; 36 ; 42 ; 48 ; 54 ; 60 ; 66 ; 72 ; 78 ; 84 ; 90 ; 96 ; 102 ; 108 ; 114 ; 120
## INPUTS
2 ; 3 ; 14 ; 32 ; 33 ; 62 ; 63 ; 92 ; 93
## OUTPUTS
17 ; 47 ; 77 ; 107 ; 118
## NETLIST
U0 PMOS 2 1 4
U1 NMOS 2 4 6
U2 NMOS 3 4 5
U3 PMOS 3 5 2
U4 PMOS 2 7 10
U5 PMOS 3 7 10
U6 NMOS 2 10 11
U7 NMOS 3 11 12
U8 PMOS 14 13 16
U9 NMOS 14 16 18
U10 NMOS 5 16 17
U11 PMOS 5 17 14
U12 PMOS 14 19 22
U13 PMOS 5 19 22
U14 NMOS 14 22 23
U15 NMOS 5 23 24
U16 PMOS 22 25 28
U17 PMOS 10 25 28
U18 NMOS 22 28 29
U19 NMOS 10 29 30
U20 PMOS 32 31 34
U21 NMOS 32 34 36
U22 NMOS 33 34 35
U23 PMOS 33 35 32
U24 PMOS 32 37 40
U25 PMOS 33 37 40
U26 NMOS 32 40 41
U27 NMOS 33 41 42
U28 PMOS 28 43 46
U29 NMOS 28 46 48
U30 NMOS 35 46 47
U31 PMOS 35 47 28
U32 PMOS 28 49 52
U33 PMOS 35 49 52
U34 NMOS 28 52 53
U35 NMOS 35 53 54
U36 PMOS 52 55 58
U37 PMOS 40 55 58
U38 NMOS 52 58 59
U39 NMOS 40 59 60
U40 PMOS 62 61 64
U41 NMOS 62 64 66
U42 NMOS 63 64 65
U43 PMOS 63 65 62
U44 PMOS 62 67 70
U45 PMOS 63 67 70
U46 NMOS 62 70 71
```

```
U46 NMOS 62 70 71
U47 NMOS 63 71 72
U48 PMOS 58 73 76
U49 NMOS 58 76 78
U50 NMOS 65 76 77
U51 PMOS 65 77 58
U52 PMOS 58 79 82
U53 PMOS 65 79 82
U54 NMOS 58 82 83
U55 NMOS 65 83 84
U56 PMOS 82 85 88
U57 PMOS 70 85 88
U58 NMOS 82 88 89
U59 NMOS 70 89 90
U60 PMOS 92 91 94
U61 NMOS 92 94 96
U62 NMOS 93 94 95
U63 PMOS 93 95 92
U64 PMOS 92 97 100
U65 PMOS 93 97 100
U66 NMOS 92 100 101
U67 NMOS 93 101 102
U68 PMOS 88 103 106
U69 NMOS 88 106 108
U70 NMOS 95 106 107
U71 PMOS 95 107 88
U72 PMOS 88 109 112
U73 PMOS 95 109 112
U74 NMOS 88 112 113
U75 NMOS 95 113 114
U76 PMOS 112 115 118
U77 PMOS 100 115 118
U78 NMOS 112 118 119
U79 NMOS 100 119 120
## END SIMULATION
```

NewFile2:

```
VCC 121 ; 127 ; 133 ; 139 ; 145 ; 151 ; 157 ; 163 ; 169 ; 175 ; 181 ; 187 ; 193 ; 199 ; 205 ; 211 ; 217 ; 223 ; 229 ; 235
GND 126 ; 132 ; 138 ; 144 ; 150 ; 156 ; 162 ; 168 ; 174 ; 180 ; 186 ; 192 ; 198 ; 204 ; 210 ; 216 ; 222 ; 228 ; 234 ; 240
## INPUTS
118 ; 122 ; 123 ; 152 ; 153 ; 182 ; 183 ; 212 ; 213
## OUTPUTS
137 ; 167 ; 197 ; 227 ; 238
## NETLIST
U0 PMOS 122 121 124
U1 NMOS 122 124 126
U2 NMOS 123 124 125
U3 PMOS 123 125 122
U4 PMOS 122 127 130
U5 PMOS 123 127 130
U6 NMOS 122 130 131
U7 NMOS 123 131 132
U8 PMOS 118 133 136
U9 NMOS 118 136 138
U10 NMOS 125 136 137
U11 PMOS 125 137 118
U12 PMOS 118 139 142
U13 PMOS 125 139 142
U14 NMOS 118 142 143
U15 NMOS 125 143 144
U16 PMOS 142 145 148
U17 PMOS 130 145 148
U18 NMOS 142 148 149
U19 NMOS 130 149 150
U20 PMOS 152 151 154
U21 NMOS 152 154 156
U22 NMOS 153 154 155
U23 PMOS 153 155 152
U24 PMOS 152 157 160
U25 PMOS 153 157 160
U26 NMOS 152 160 161
U27 NMOS 153 161 162
U28 PMOS 148 163 166
U29 NMOS 148 166 168
U30 NMOS 155 166 167
U31 PMOS 155 167 148
U32 PMOS 148 169 172
U33 PMOS 155 169 172
U34 NMOS 148 172 173
U35 NMOS 155 173 174
U36 PMOS 172 175 178
U37 PMOS 160 175 178
U38 NMOS 172 178 179
U39 NMOS 160 179 180
U40 PMOS 182 181 184
U41 NMOS 182 184 186
U42 NMOS 183 184 185
U43 PMOS 183 185 182
U44 PMOS 182 187 190
U45 PMOS 183 187 190
U46 NMOS 182 190 191
U46 NMOS 182 190 191
U47 NMOS 183 191 192
U48 PMOS 178 193 196
U49 NMOS 178 196 198
U50 NMOS 185 196 197
U51 PMOS 185 197 178
U52 PMOS 178 199 202
U53 PMOS 185 199 202
U54 NMOS 178 202 203
U55 NMOS 185 203 204
U56 PMOS 202 205 208
U57 PMOS 190 205 208
U58 NMOS 202 208 209
U59 NMOS 190 209 210
U60 PMOS 212 211 214
U61 NMOS 212 214 216
U62 NMOS 213 214 215
U63 PMOS 213 215 212
U64 PMOS 212 217 220
U65 PMOS 213 217 220
U66 NMOS 212 220 221
U67 NMOS 213 221 222
U68 PMOS 208 223 226
U69 NMOS 208 226 228
U70 NMOS 215 226 227
U71 PMOS 215 227 208
U72 PMOS 208 229 232
U73 PMOS 215 229 232
U74 NMOS 208 232 233
U75 NMOS 215 233 234
U76 PMOS 232 235 238
U77 PMOS 220 235 238
U78 NMOS 232 238 239
U79 NMOS 220 239 240
## END SIMULATION
```

Το 2 αρχεία συνδέονται μέσο του κόμβου 118 που είναι και ο καλύτερος κόμβος διάσπασης και μπορούμε να το δούμε στα παραπάνω αρχεία.

Διαβάζοντας τα παραπάνω 2 αρχεία και τρέχοντας τον αλγόριθμο για τα παρακάτω test_vector χρησιμοποιώντας για test_in τα inputs και test_out τα output του κάθε αρχείου προκύπτουν τα αποτελέσματα:

Manual_Test_vectors={{0,0,0,0,0,0,0,0,0 } , {1,0,1,0,1,0,1,0,1} , {0,0,1,0,1,0,1,0,1 } , {0,1,0,0,0,1,1,1,1 },{1,1,1,1,1,1,1,1,1 } }

Η σειρα των inp είναι : inp1,inp2,Cin,Inp3,inp4,inp5,inp6,inp7,inp8 και αντιστοιχίζονται 1-1 με τα Manual_Test_vectors.

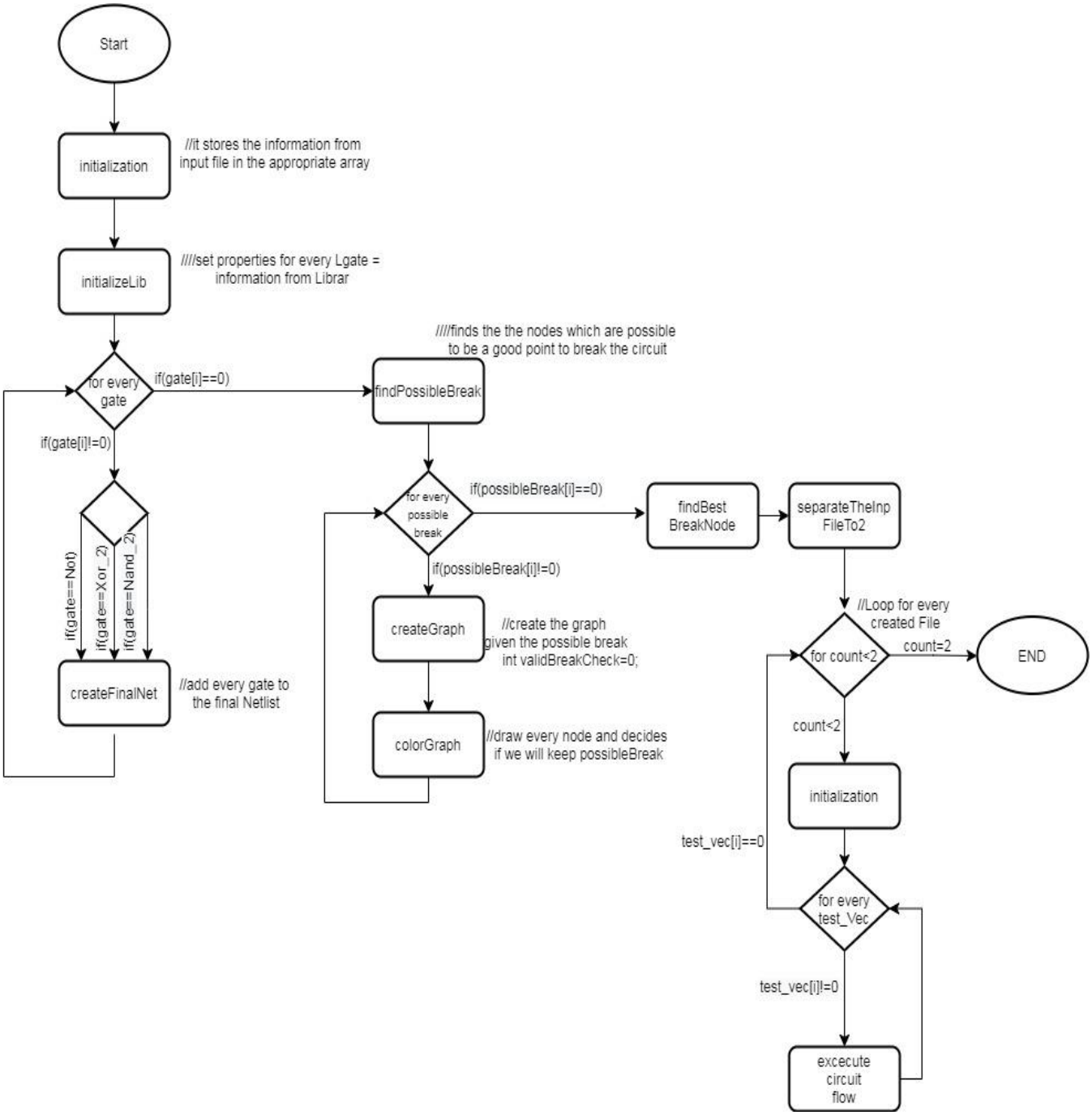
Τα αποτελέσματα που παίρνουμε είναι :

```
For the 0 TEST_VECTORS For File 1 the output 17 is : 0
For File 1 the output 47 is : 0
For File 1 the output 77 is : 0
For File 1 the output 107 is : 0
For File 1 the output 118 is : 0
For the 1 TEST_VECTORS For File 1 the output 17 is : 0
For File 1 the output 47 is : 0
For File 1 the output 77 is : 0
For File 1 the output 107 is : 0
For File 1 the output 118 is : 1
For the 2 TEST_VECTORS For File 1 the output 17 is : 1
For File 1 the output 47 is : 1
For File 1 the output 77 is : 1
For File 1 the output 107 is : 1
For File 1 the output 118 is : 0
For the 3 TEST_VECTORS For File 1 the output 17 is : 1
For File 1 the output 47 is : 1
For File 1 the output 77 is : 0
For File 1 the output 107 is : 1
For File 1 the output 118 is : 1
For the 4 TEST_VECTORS For File 1 the output 17 is : 1
For File 1 the output 47 is : 1
For File 1 the output 77 is : 1
For File 1 the output 107 is : 1
For File 1 the output 118 is : 1
```

```
For the 0 TEST_VECTORS For File 2 the output 137 is : 0
For File 2 the output 167 is : 0
For File 2 the output 197 is : 0
For File 2 the output 227 is : 0
For File 2 the output 238 is : 0
For the 1 TEST_VECTORS For File 2 the output 137 is : 0
For File 2 the output 167 is : 0
For File 2 the output 197 is : 0
For File 2 the output 227 is : 0
For File 2 the output 238 is : 1
For the 2 TEST_VECTORS For File 2 the output 137 is : 1
For File 2 the output 167 is : 1
For File 2 the output 197 is : 1
For File 2 the output 227 is : 1
For File 2 the output 238 is : 0
For the 3 TEST_VECTORS For File 2 the output 137 is : 1
For File 2 the output 167 is : 1
For File 2 the output 197 is : 0
For File 2 the output 227 is : 1
For File 2 the output 238 is : 1
For the 4 TEST_VECTORS For File 2 the output 137 is : 1
For File 2 the output 167 is : 1
For File 2 the output 197 is : 1
For File 2 the output 227 is : 1
For File 2 the output 238 is : 1
```

Βλέπουμε ότι τα αποτελέσματα είναι σωστά αφού το κάθε output που εμφανίζεται στην έξοδο είναι η έξοδος του κάθε full adder του υπόκύκλώμάτός καθώς και το Cin που είναι η τελευταία έξοδος. Η εγκυρότητα των αποτελεσμάτων για τα ενδεικτικά test_Vector μας υποδηλώνει την ορθότητα του αλγορίθμου.

Ακλουθεί ενδεικτικό flowchart του κυκλώματος :


Start
initialization
//it stores the information from input file in the appropriate array
initializeLib
////set properties for every Lgate = information from Librar
for every gate
if(gate[i]==0)
if(gate[i]!=0)
if(gate==Not)
if(gate==Xor_2)
if(gate==Nand_2)
createFinalNet
//add every gate to the final Netlist
////finds the the nodes which are possible to be a good point to break the circuit
findPossibleBreak
for every possible break
if(possibleBreak[i]==0)
if(possibleBreak[i]!=0)
createGraph
//create the graph given the possible break int validBreakCheck=0;
colorGraph
//draw every node and decides if we will keep possibleBreak
findBest BreakNode
separateTheInp FileTo2
//Loop for every created File
for count<2
count=2
END
count<2
initialization
test_vec[i]==0
for every test_Vec
test_vec[i]!=0
excecute circuit flow