Σχεδιασμός Συστημάτων VLSI

Εργαστήριο 4

Λουδάρος Ιωάννης (1067400)

Μπορείτε να δείτε την τελευταία έκδοση του Project εδώ ή σκανάροντας τον κωδικό QR που βρίσκεται στην επικεφαλίδα.

Περιγραφή Αναφοράς

Παρακάτω παραθέτω τις απαντήσεις μου στην "4η Εργαστηριακή Άσκηση" του μαθήματος "Σχεδιασμός Συστημάτων VLSI" καθώς και σχόλια τα οποία προέκυψαν κατά την εκπόνηση της.

Περιεχόμενα

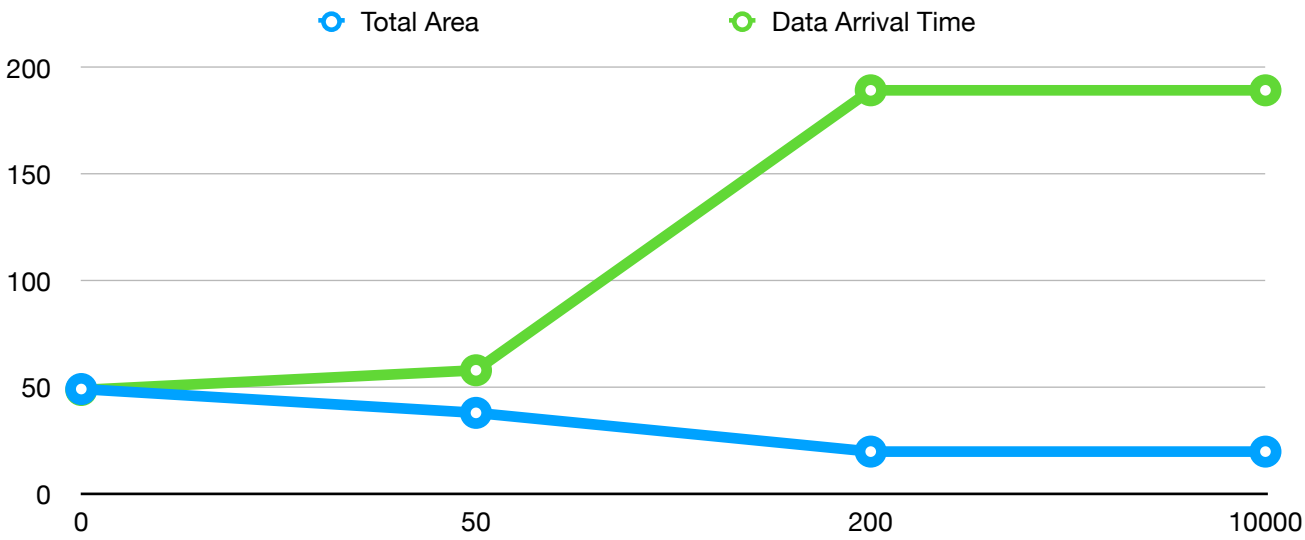
Απαντήσεις

1. Άσκηση 1

1.1. Πειραματισμός με την Μέγιστη Καθυστέρηση

Έγιναν 4 δοκιμές με χρόνους 0, 50, 200, 10000. Σε όλες τις περιπτώσεις χρησιμοποιήθηκε "`compile_ultra`". Μπορείτε να βρείτε τα αντίστοιχα scripts στον κατάλογο "`my_scripts`", με όνομα "`ex_1_1_(καθυστέρηση).tcl`".

Παρατηρούμε τις εξής διαφορές μεταξύ των διαφορετικών χρόνων:



	0	50	200	10000
Total Area	49.2221	38.0246	19.8288	19.8288
Data Arrival Time	48.77	58.03	189.53	189.53

1.2. Διαφορετικοί τρόποι compilation

Μπορείτε να βρείτε τα αντίστοιχα scripts στον κατάλογο "`my_scripts`", με όνομα "`ex_1_2_200_(compile command).tcl`".

Τα αποτελέσματα μας φαίνονται παρακάτω:

	compile	compile_ultra
Total Area	25.8941	19.8288

Μας γίνεται ξεκάθαρη η διαφορά μεταξύ "`compile`" και "`compile_ultra`".

Συγκρίνοντας τα netlist που παράγονται, βλέπουμε επίσης, ότι το "`compile_ultra`" έχει κάνει τον σχεδιασμό μας μη ιεραρχικό, για να κάνει όλες τις δυνατές απλοποιήσεις.

2. Άσκηση 2

2.1. RCA με καταχωρητές

Μπορείτε να βρείτε τον σχεδιασμό μας εδώ :

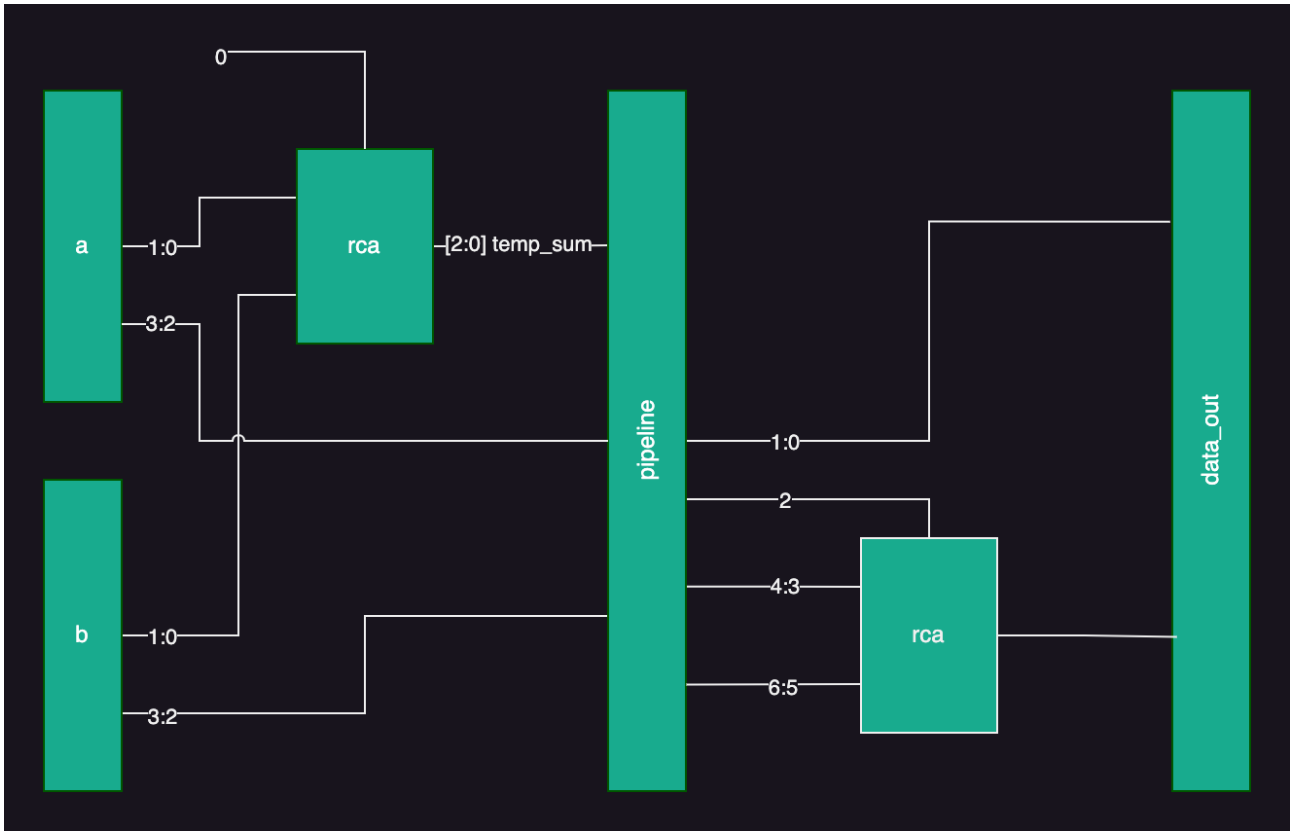
rca_w_regs.v

Σύμφωνα με το `timing_report`, το data arrival time είναι `43.73`. Άρα η μέγιστη συχνότητα είναι:
$\frac{1}{43.73} \approx 0.0228$

Μπορείτε να βρείτε τα αντίστοιχα scripts στον κατάλογο “`my_scripts`”, με όνομα “`ex_2_1_(compile command).tcl`”.

2.2. RCA με Pipeline

Η λογική μας για την παραγωγή του σχεδιασμού φαίνεται στο παρακάτω σχηματικό:

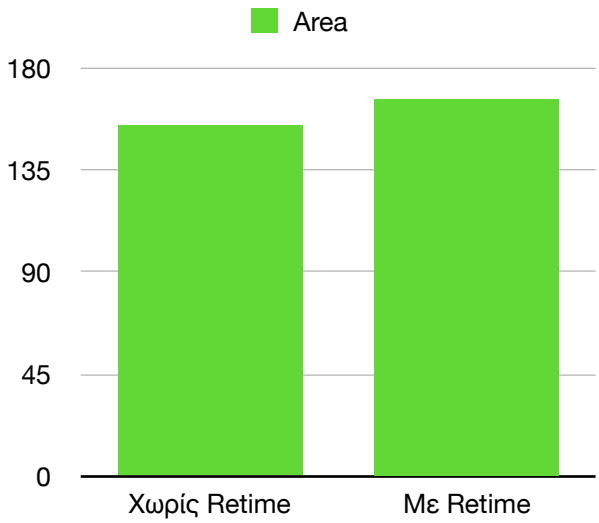


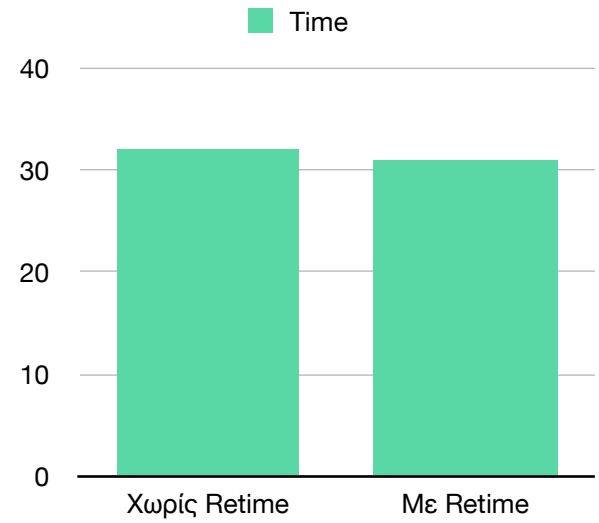
Μπορείτε να βρείτε τα αντίστοιχα scripts στον κατάλογο “`my_scripts`”, με όνομα “`ex_2_2_(retime option).tcl`”.

Μπορείτε να βρείτε τον σχεδιασμό μας εδώ :

rca_pipelined.v

Ανάλυση area-report, με και χωρίς retime.





	Χωρίς Retime	Με Retime
Total Area	154.198079	166.095359
Data Arrival Time	31.85	30.80

3. Άσκηση 3

3.1. Σύνθεση του Accumulator

Επειδή το presto μόνο που δεν με έφτισε για τον δικό μου accumulator, χρησιμοποίησα τον δικό σας.

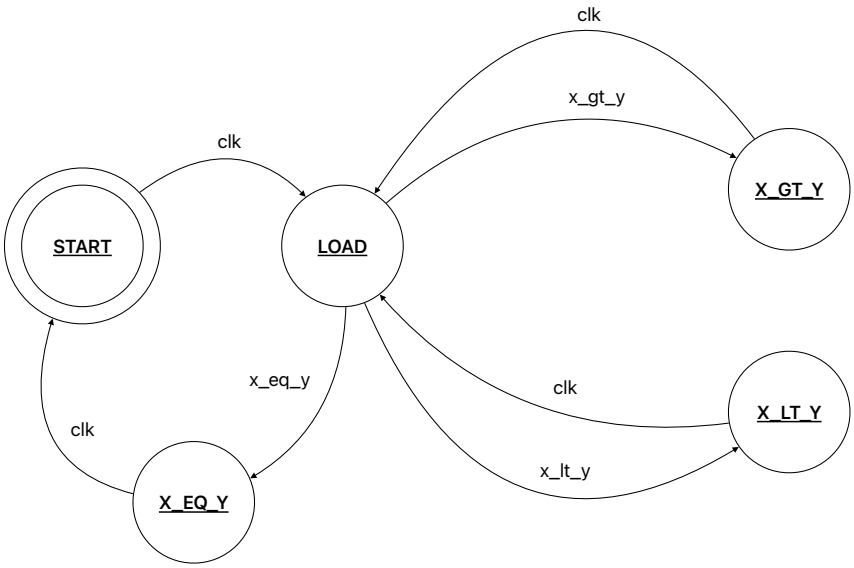
	Accumulator
Total Area	214.850880
Data Arrival Time	32.76

3.2. Σύνθεση του Vending Machine

	Vending Machine
Total Area	37.324800
Data Arrival Time	34.07

4. Άσκηση 4

Παραθέτω το FSM που χρησιμοποιήθηκε για την δημιουργία του κυκλώματος.



START

x_sel : 1
y_sel : 1
x_keep : 1
y_keep : 1
x_ld : 0
y_ld : 0
data_en : 0

LOAD

x_ld : 1
y_ld : 1

X_GT_Y

x_sel : 0
y_sel : 0
x_keep : 0
y_keep : 1
x_ld : 0
y_ld : 0

X_EQ_Y

data_en : 1
x_ld : 0
y_ld : 0

X_LT_Y

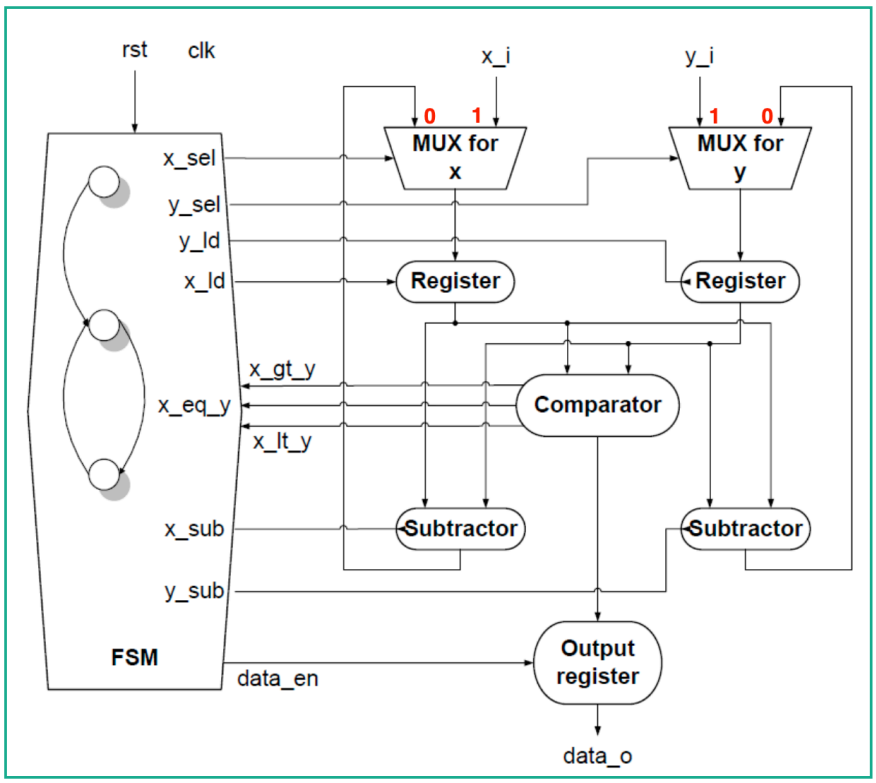
x_sel : 0
y_sel : 0
x_keep : 1
y_keep : 0
x_ld : 0
y_ld : 0

Μπορείτε να βρείτε τις περιγραφές των δύο modules, πατώντας τα κουμπιά παρακάτω:

gcd_dataflow.v

gcd_fsm.v

Να σημειωθεί ότι έχει γίνει η παραδοχή ότι το κύκλωμα δεν γίνεται να πάρει σε μόνο μια είσοδο από τις x και y για τιμή το 0. Αν θέλαμε να προστεθεί αυτή η ικανότητα, φτάνει ένα δέντρο από πύλες and που θα εξετάζει αν όλα τα ψηφία του x και του y είναι μηδέν, και να μας ενημερώνει κατευθείαν ότι υπάρχει error.



Η μόνη προσθήκη που χρειαζόμαστε στο υπάρχον σχήμα είναι να προσθέσουμε άλλον έναν πολυπλέκτη εκατέρωθεν ώστε να γίνεται να κρατήσουμε την τιμή των μεταβλητών πριν γίνει η αφαίρεση. Η έξοδος των καινούργιων πολυπλεκτών οδηγεί την είσοδο μηδέν των υπαρχόντων. Ελέγχονται από το σήμα keep_y και keep_x αντίστοιχα.

Εξομοίωση Κυκλώματος

Το κύκλωμα ελέγχεται ενδελεχώς για όλες τις δυνατές εισόδους από το “`gcd_tb`”, το οποίο εξάγει τα αποτελέσματα για όλες τις εισόδους σε μορφή κυματομορφής (`results_gcd/gcd.vcd`), αλλά και σε απλή μορφή πράξης-αποτελέσματος (`results_gcd/gcd_results.txt`). Η δεύτερη μορφή ελέγχεται μέσω του “`src/4/test_gcd_results`”, το οποίο τελικά μας επιβεβαιώνει αν το κύκλωμα μας λειτουργεί σωστά ή όχι.

```
vlsi2_2023_7@dagobah:~/lab4

[vlsi2_2023_7@dagobah lab4]$ make ex4_sim
vcs -full64  -q -debug_access+all -timescale=10ns/1ns -v2005 gcd_tb -o zitoumeno4
Doing common elaboration

1 module and 0 UDP read.
make[1]: Entering directory '/home/vlsi2_2023_7/lab4/csrc'
make[1]: Leaving directory '/home/vlsi2_2023_7/lab4/csrc'
make[1]: Entering directory '/home/vlsi2_2023_7/lab4/csrc'
../zitoumeno4 up to date
make[1]: Leaving directory '/home/vlsi2_2023_7/lab4/csrc'
./zitoumeno4
Chronologic VCS simulator copyright 1991-2022
Contains Synopsys proprietary information.
Compiler version T-2022.06-SP1-1_Full64; Runtime version T-2022.06-SP1-1_Full64;  May 23 04:13 2023
$finish called from file "src/4/gcd_tb.v", line 41.
$finish at simulation time            266860560
           V C S   S i m u l a t i o n   R e p o r t
Time: 266860560 ns
CPU Time:      4.700 seconds;       Data structure size:   0.0Mb
Tue May 23 04:14:00 2023
python3 src/4/test_gcd_results.py ./results_gcd/gcd_results.txt
✅ The test was succesful!. Μπράβο μωρή φιλενάδα <3
❗Open results_gcd/gcd.vcd for the waveforms.
[vlsi2_2023_7@dagobah lab4]$
```

Σύνθεση Κυκλώματος

Το κύκλωμα μας είναι συνθέσιμο και δουλεύει χωρίς κανένα πρόβλημα σε ρολόι 130.

	Vending Machine
Total Area	309.329275
Data Arrival Time	113.50