Ομάδα Α10:
Γεώργιος Παγώνης : 03117030
Δημήτριος - Σταμάτιος Μπούρας : 03117072
8ο εξάμηνο - ΨΗΦΙΑΚΑ ΣΥΣΤΗΜΑΤΑ VLSI
ΣΗΜΜΥ

3η σειρά ασκήσεων

Όλα τα κυκλώματα περιέχονται σε ξεχωριστά αρχεία vhd στο zip που αποστείλαμε. Παρακάτω για κάθε ερώτημα της άσκησης ξεχωριστά θα αναφέρουμε ποιο testbench αντιστοιχεί σε ποιο ερώτημα καθώς και τα υπόλοιπα αρχεία που απαιτούνται να βρίσκονται στο ίδιο vivado project με αυτό , για να εξασφαλιστεί η ορθή λειτουργία του.
Μια γρήγορη λύση αποτελεί και η προσθήκη όλων των vhd αρχείων του zip σε ένα vivado project και η εκτέλεση όλων των κυκλωμάτων μέσα σε αυτό το project θέτοντας ως “top” κάθε φορά το κατάλληλο.

1)

Τα σήματα εισόδου και τα modules έχουν την εξής λειτουργία το καθένα :

rst : αρχικοποιεί την ram , μηδενίζει όλους τους καταχωρητές , μηδενίζει μετρητή στο control unit .
x : 8bit αριθμός εισόδου
valid_in : μας υποδηλώνει πως η είσοδος x είναι ορθή και πρέπει να ληφθεί υπόψη για υπολογισμό εξόδου

Control_unit :

Αυτό το module αποτελείται από έναν μετρητή ο οποίος είναι υπεύθυνος κάθε κύκλο να δίνει σωστή διεύθυνση πρόσβασης στην rom και ram .
Επίσης έχει υλοποιηθεί έτσι ώστε να γράφει στην ram μόνο όταν έρθει ορθή τιμή x (valid_in =1) προσέχοντας να έχουν περάσει τουλάχιστον 8 κύκλοι από την προηγούμενη είσοδο .Αλλιώς η είσοδος αυτή αγνοείται.
Όταν λάβει σε σωστό χρόνο την νέα τιμή x επιτρέπει μέσω των σημάτων we , en την αποθήκευση της στην ram και μέσω του σήματος mac_init αρχίζει την λειτουργία του mac_module.

Mac_module :

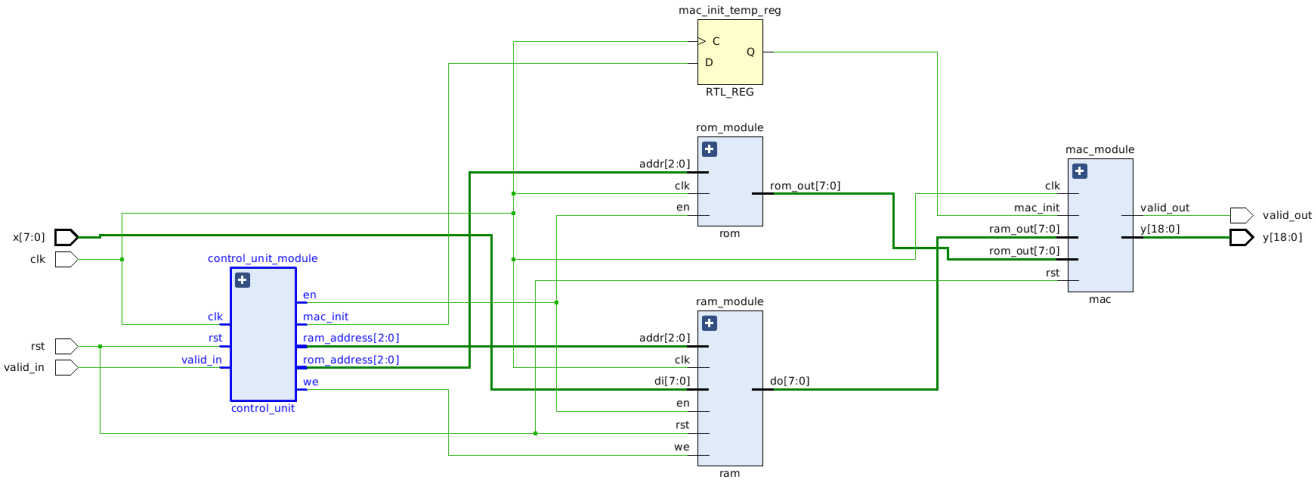
Το module αυτό λαμβάνει κάθε κύκλο 2 τιμές , τις πολλαπλασιάζει , τις προσθέτει στο ήδη υπάρχον άθροισμα και αφού υπολογιστεί ολόκληρο το άθροισμα το εμφανίζει και ενημερώνει το valid_out. Σαν έξοδο το αποτέλεσμα εμφανίζεται στο 19bit σήμα y (16 bit από τους πολλαπλασιασμούς και 3 bit από τις αθροίσεις).

Ram and Rom :

Χρησιμοποιήθηκαν ακριβώς όπως μας δόθηκαν , με μόνη αλλαγή η ram να πραγματοποιεί swift right τα δεδομένα της κάθε φορά που είναι να γραφτεί νέα τιμή.

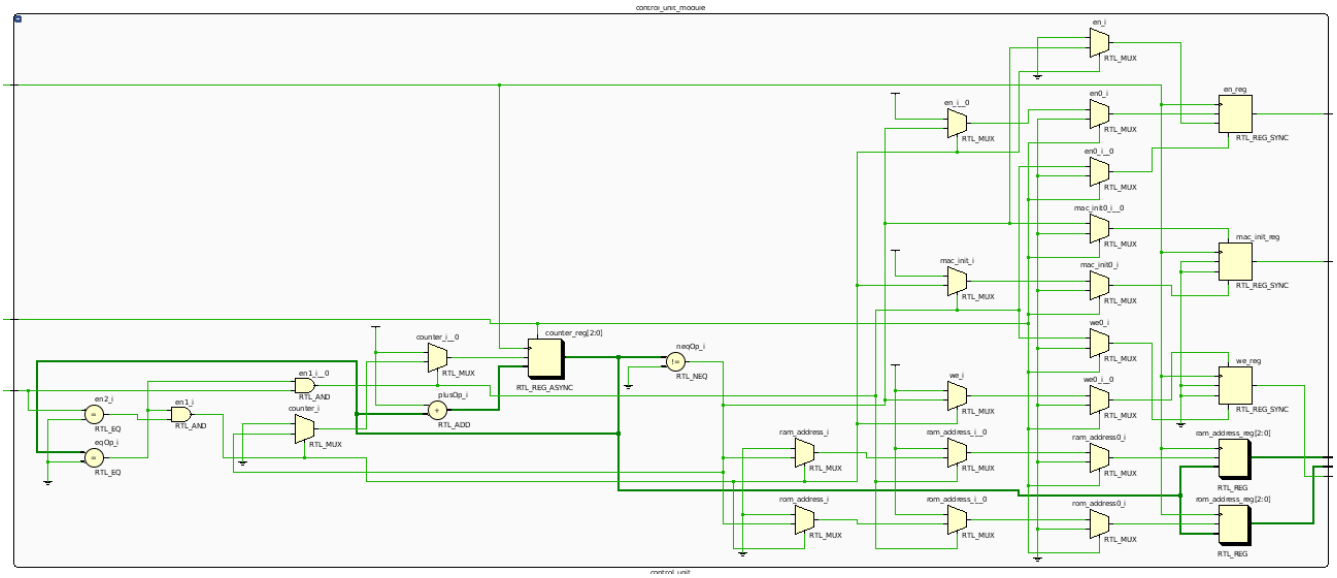
Testbench : fir_tb
Λοιπά απαραίτητα αρχεία :fir , mac , control_unit, ram ,rom

RTL schematic

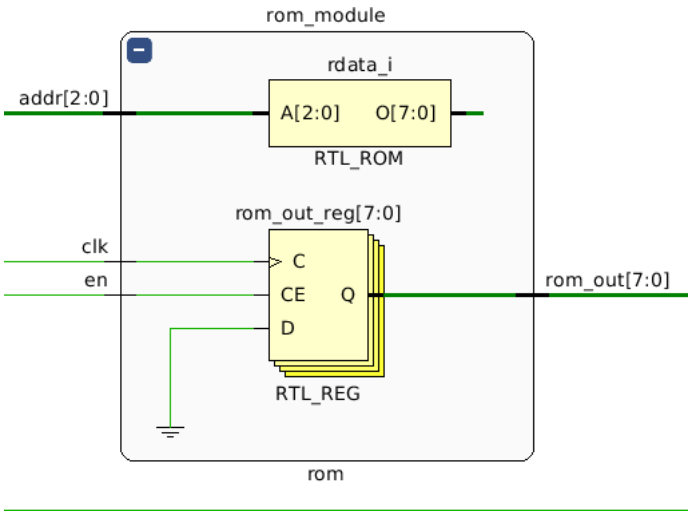


Τα παραπάνω modules εσωτερικά έχουν την δομή που φαίνεται παρακάτω :

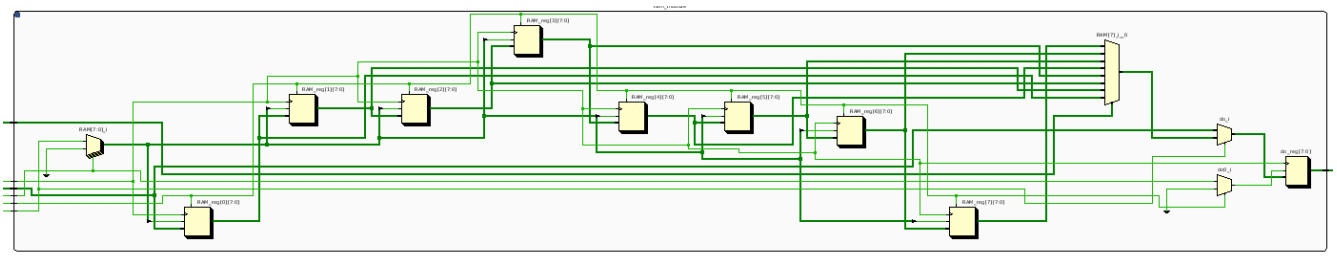
control_unit_module :



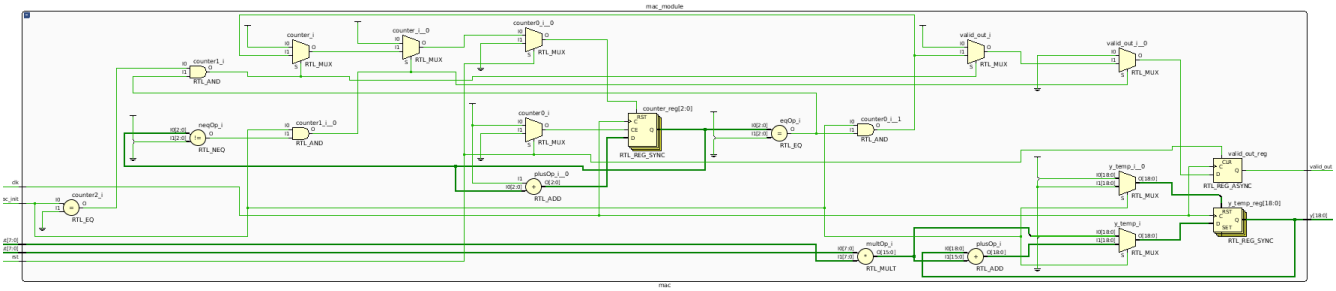
rom_module:



ram_module:



mac_module:



Critical Path and Time Delay:

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement
Path 1	∞	10	10	19	rom_module/rom_out_reg[1]/C	mac_module/y...p_reg[17]/D	5.613	2.700	2.913	∞
Path 2	∞	10	10	19	rom_module/rom_out_reg[1]/C	mac_module/y...p_reg[18]/D	5.521	2.608	2.913	∞
Path 3	∞	10	10	19	rom_module/rom_out_reg[1]/C	mac_module/y...p_reg[16]/D	5.500	2.587	2.913	∞
Path 4	∞	9	9	19	rom_module/rom_out_reg[1]/C	mac_module/y...p_reg[13]/D	5.499	2.586	2.913	∞
Path 5	∞	9	9	19	rom_module/rom_out_reg[1]/C	mac_module/y...p_reg[15]/D	5.480	2.567	2.913	∞
Path 6	∞	9	9	19	rom_module/rom_out_reg[1]/C	mac_module/y...p_reg[14]/D	5.407	2.494	2.913	∞
Path 7	∞	9	9	19	rom_module/rom_out_reg[1]/C	mac_module/y...p_reg[12]/D	5.386	2.473	2.913	∞
Path 8	∞	7	7	8	ram_module/do_reg[3]/C	mac_module/y_temp_reg[9]/D	5.287	2.545	2.742	∞
Path 9	∞	7	7	8	ram_module/do_reg[3]/C	mac_module/y...p_reg[11]/D	5.268	2.526	2.742	∞
Path 10	∞	7	7	8	ram_module/do_reg[3]/C	mac_module/y...p_reg[10]/D	5.195	2.453	2.742	∞

Critical path : path 1

Time Delay : 5.613

Resource Utilization:

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	88	17600	0.50
FF	109	35200	0.31
IO	31	100	31.00
BUFG	1	32	3.13

2)

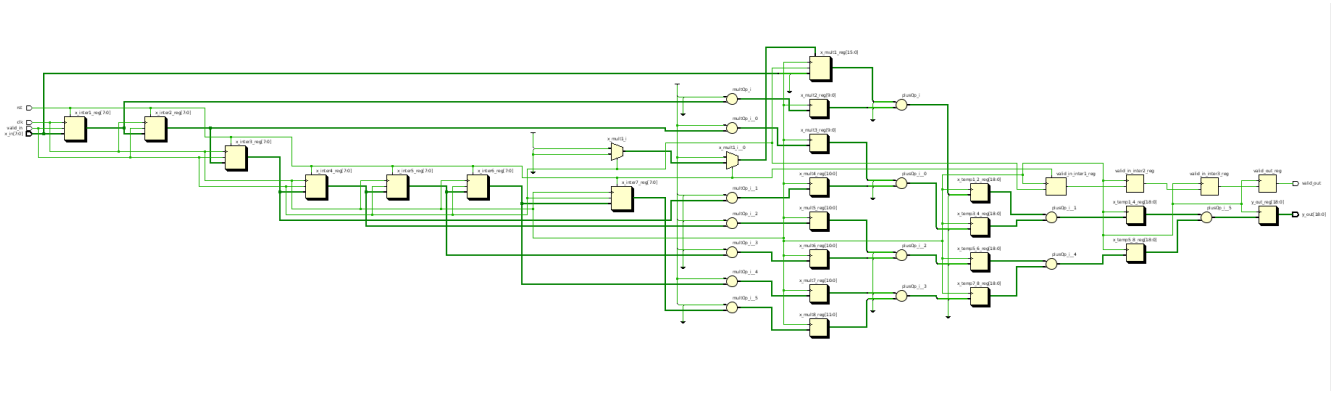
Σημείωση :

Η υλοποίηση μας βασίστηκε στο σχήμα που δόθηκε στην εκφώνηση με μια μικρή βελτίωση .
Αντί οι προσθέσεις να γίνονται σειριακά η μία μετά την άλλη και μετά από 7 κύκλους να έχουμε το ορθό αποτέλεσμα , γίνονται σε δομή πυραμίδας δηλαδή 4 προσθέσεις ταυτόχρονα σε ζευγάρια τους

αρχικούς αριθμούς μετά τα 4 αποτελέσματα και αυτά μεταξύ τους σε ζευγάρια . Τα 2 νέα αποτελέσματα μεταξύ τους και αυτό που προκύπτει είναι το ζητούμενο σε μόλις 3 κύκλους, χωρίς κόστος υλικού .

Καταχωρητές έχουν προστεθεί μετά από κάθε πρόσθεση και πολλαπλασιασμό για μείωση του critical path.
Testbench : filter_pipeline_tb
Λοιπά απαραίτητα αρχεία :filter_pipeline

RTL schematic



Critical Path and Time Delay:

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement
Path 1	∞	2	2	1	y_out_reg[0]/C	y_out[0]	4.076	3.276	0.800	∞
Path 2	∞	2	2	1	y_out_reg[10]/C	y_out[10]	4.076	3.276	0.800	∞
Path 3	∞	2	2	1	y_out_reg[11]/C	y_out[11]	4.076	3.276	0.800	∞
Path 4	∞	2	2	1	y_out_reg[12]/C	y_out[12]	4.076	3.276	0.800	∞
Path 5	∞	2	2	1	y_out_reg[13]/C	y_out[13]	4.076	3.276	0.800	∞
Path 6	∞	2	2	1	y_out_reg[14]/C	y_out[14]	4.076	3.276	0.800	∞
Path 7	∞	2	2	1	y_out_reg[15]/C	y_out[15]	4.076	3.276	0.800	∞
Path 8	∞	2	2	1	y_out_reg[16]/C	y_out[16]	4.076	3.276	0.800	∞
Path 9	∞	2	2	1	y_out_reg[17]/C	y_out[17]	4.076	3.276	0.800	∞
Path 10	∞	2	2	1	y_out_reg[18]/C	y_out[18]	4.076	3.276	0.800	∞

Critical path : path 1 - 10

Time Delay : 4.076

Resource Utilization:

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	39	17600	0.22
LUTRAM	1	6000	0.02
FF	77	35200	0.22
DSP	7	80	8.75
IO	31	100	31.00
BUFG	1	32	3.13

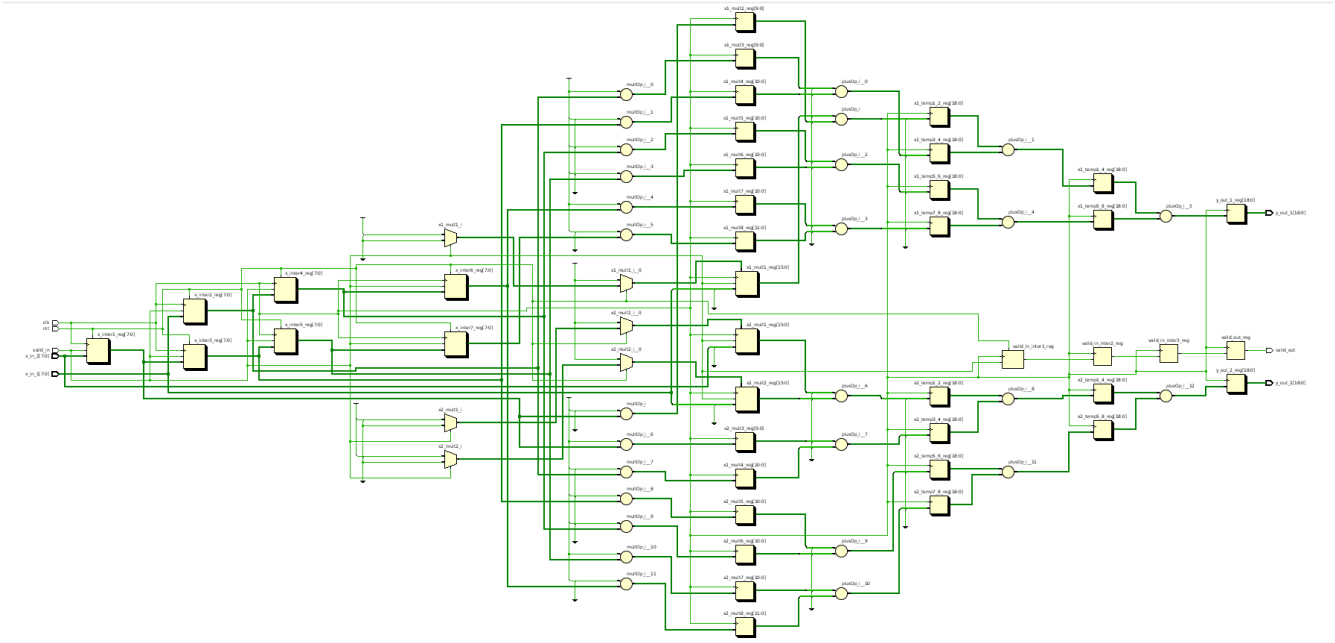
3)

Σημείωση :

Ίδια υλοποίηση με άσκηση 2 με μόνη διαφορά πως λαμβάνουμε 2 τιμές σε κάθε κύκλο , άρα το swift right γίνεται 2 καταχωρητές δεξιά . Από κει και έπειτα ο υπολογισμός των 2 αποτελεσμάτων γίνεται ανεξάρτητα το ένα από το άλλο.

Testbench : filter_parallel_tb
Λοιπά απαραίτητα αρχεία :filter_parallel

RTL schematic



Critical Path and Time Delay:

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement
Path 1	∞	2	2	1	y_out_1_reg[0]/C	y_out_1[0]	4.076	3.276	0.800	∞
Path 2	∞	2	2	1	y_out_1_reg[10]/C	y_out_1[10]	4.076	3.276	0.800	∞
Path 3	∞	2	2	1	y_out_1_reg[11]/C	y_out_1[11]	4.076	3.276	0.800	∞
Path 4	∞	2	2	1	y_out_1_reg[12]/C	y_out_1[12]	4.076	3.276	0.800	∞
Path 5	∞	2	2	1	y_out_1_reg[13]/C	y_out_1[13]	4.076	3.276	0.800	∞
Path 6	∞	2	2	1	y_out_1_reg[14]/C	y_out_1[14]	4.076	3.276	0.800	∞
Path 7	∞	2	2	1	y_out_1_reg[1]/C	y_out_1[1]	4.076	3.276	0.800	∞
Path 8	∞	2	2	1	y_out_1_reg[2]/C	y_out_1[2]	4.076	3.276	0.800	∞
Path 9	∞	2	2	1	y_out_1_reg[3]/C	y_out_1[3]	4.076	3.276	0.800	∞
Path 10	∞	2	2	1	y_out_1_reg[4]/C	y_out_1[4]	4.076	3.276	0.800	∞

Critical path : path 1 - 10

Time Delay : 4.076

Resource Utilization:

Utilization Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	175	17600	0.99
LUTRAM	1	6000	0.02
FF	283	35200	0.80
DSP	4	80	5.00
IO	58	100	58.00
BUFG	1	32	3.13

Παρατηρήσεις :

Στις ασκήσεις 2 και 3 το critical path είναι το ίδιο και αποτελείται από την πράξη του πολλαπλασιασμού που πρέπει να πραγματοποιηθεί .

Ενώ στην άσκηση 1 είναι λίγο μεγαλύτερο γιατί πρέπει να προηγηθεί πρόσβαση στην rom .