ΕΘΝΙΚΟ ΜΕΤΣΟΒΙΟ ΠΟΛΥΤΕΧΝΕΙΟ

ΣΗΜΜΥ

Ψηφιακά Συστήματα VLSI

Ακαδημαϊκό έτος 2020-21

2η Εργαστηριακή Άσκηση

Αιμιλία Σιόκουρου – 03117703

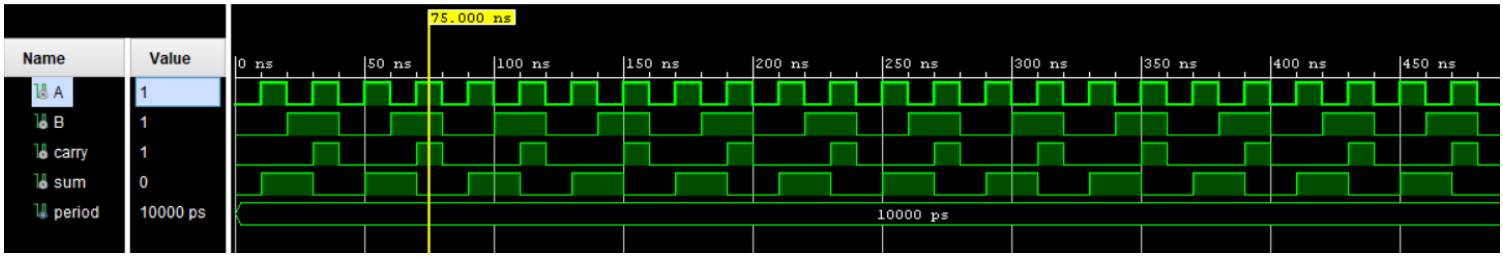
Νάταλυ Πεγειώτη – 03117707

**Όλοι οι κώδικες της VHDL παρατίθενται σε ξεχωριστά αρχεία*

1) *Να υλοποιήσετε συνδιαστικό Ημιαθροιστή (Half Adder – HA) με περιγραφή ροής δεδομένων (Dataflow)*

Ο **κώδικας** της Dataflow αρχιτεκτονικής καθώς και το αντίστοιχο **Testbench** παρατίθενται σε αρχεία VHDL στον φάκελο “1”.

Η αντίστοιχη προσομοίωση (**simulation**) φαίνεται πιο κάτω:

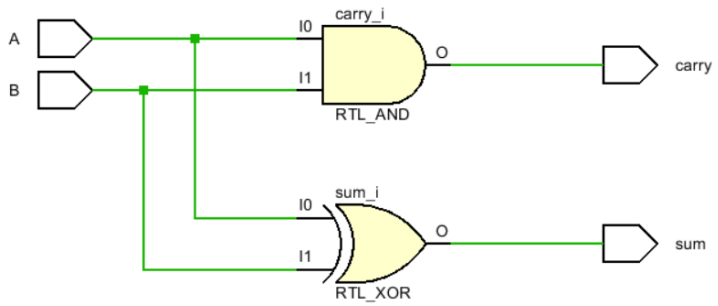


Ακολουθεί πίνακας αληθείας ενός ημιαθροιστή:

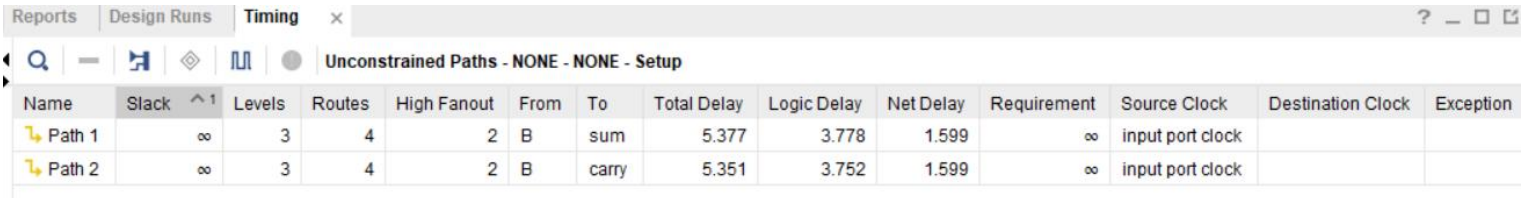
Truth Table			
Input		Output	
A	B	Sum	Carry
0	0	0	0
0	1	1	0
1	0	1	0
1	1	0	1

Παρατηρούμε ότι η προσομοίωσή μας είναι ορθή, ανάλογη του πιο πάνω πίνακα.

Το **RTL schematic** του ημιαθροιστή παρουσιάζεται παρακάτω:



Critical Path



Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception
Path 1	∞	3	4	2	B	sum	5.377	3.778	1.599	∞	input port clock		
Path 2	∞	3	4	2	B	carry	5.351	3.752	1.599	∞	input port clock		

Κρίσιμο είναι το μονοπάτι με την μεγαλύτερη συνολικη καθυστέρηση, δηλαδή max{Total Delay}, όπου στην περίπτωση αυτή είναι το Path 1.

Κατανάλωση πόρων του FPGA

Πιο κάτω παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

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

Graph | **Table**

Resource	Estimation	Available	Utilization %
LUT	1	17600	0.01
IO	4	100	4.00

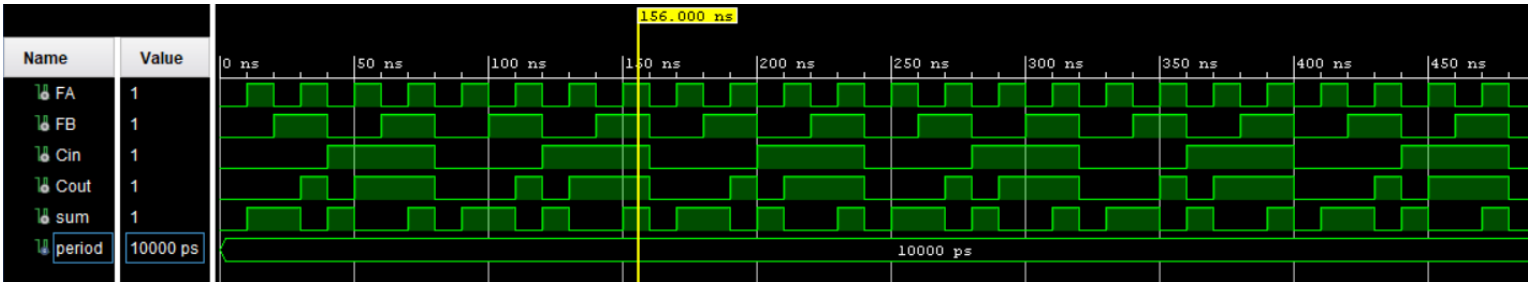
Πρόκειται για έναν LUT (Lookup Table) και τέσσερις (4) I/O.

2) Να υλοποιήσετε Πλήρη Αθροιστή (Full Adder – FA) με τους ακόλουθους τρόπους:

α) με περιγραφή δομής (Structural) και βασιζόμενοι στην δομική μονάδα του Ζητήματος 1, καθώς και χρήση επιπλέον λογικής που θεωρείτε απαραίτητη. Να σχεδιαστεί συνδυαστικό και ακολουθιακό κύκλωμα

Οι **κώδικες** του ερωτήματος και τα αντίστοιχα **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο "2a".

Η προσομοίωση (**simulation**) του **συνδυαστικού** κυκλώματος φαίνεται πιο κάτω:

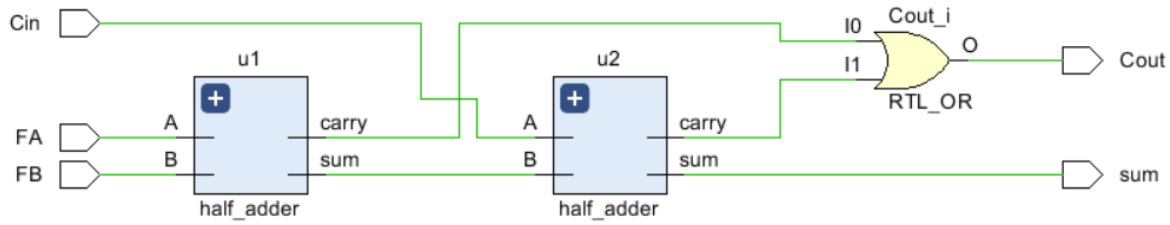


Ακολουθεί πίνακας αληθείας ενός πλήρη αθροιστή:

Input			Output	
A	**B**	**C_{in}**	**Sum**	**Carry**
0	0	0	0	0
0	0	1	1	0
0	1	0	1	0
0	1	1	0	1
1	0	0	1	0
1	0	1	0	1
1	1	0	0	1
1	1	1	1	1

Παρατηρούμε ότι η προσομοίωσή μας είναι ορθή, ανάλογη του πιο πάνω πίνακα.

Το **RTL schematic** του πλήρη αθροιστή παρουσιάζεται παρακάτω:



Critical Path

Reports | Design Runs | Timing

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock	Destination Clock	Exception
Path 1	∞	3	4	2	FA	Cout	5.377	3.778	1.599	∞	input port clock		
Path 2	∞	3	4	2	Cin	sum	5.351	3.752	1.599	∞	input port clock		

Το Path 1 είναι το κρίσιμο μονοπάτι.

Κατανάλωση πόρων του FPGA

Πιο κάτω παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

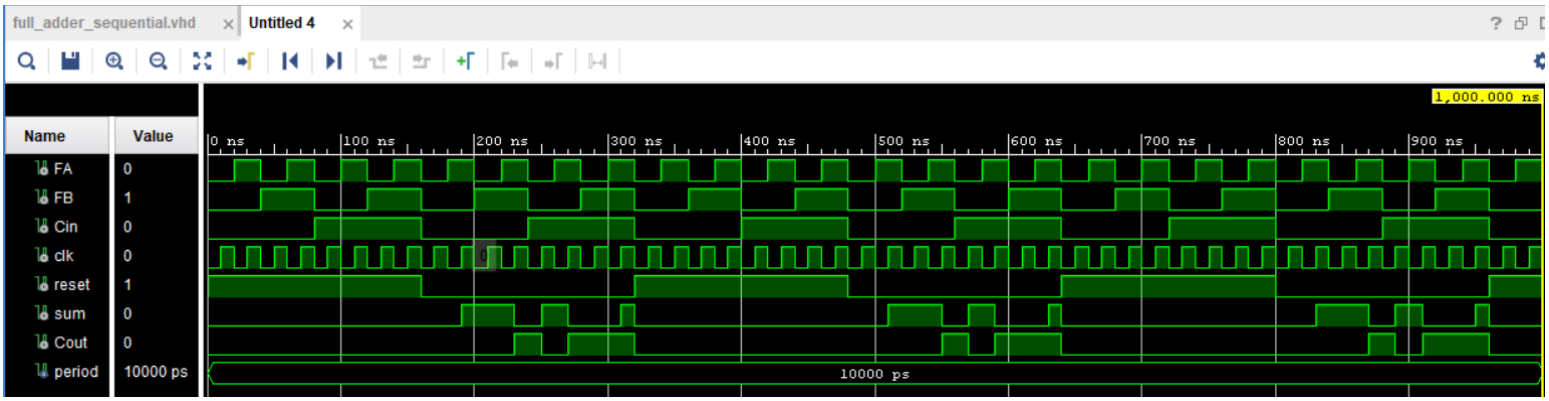
Utilization — Post-Synthesis | **Post-Implementation**

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	1	17600	0.01
IO	5	100	5.00

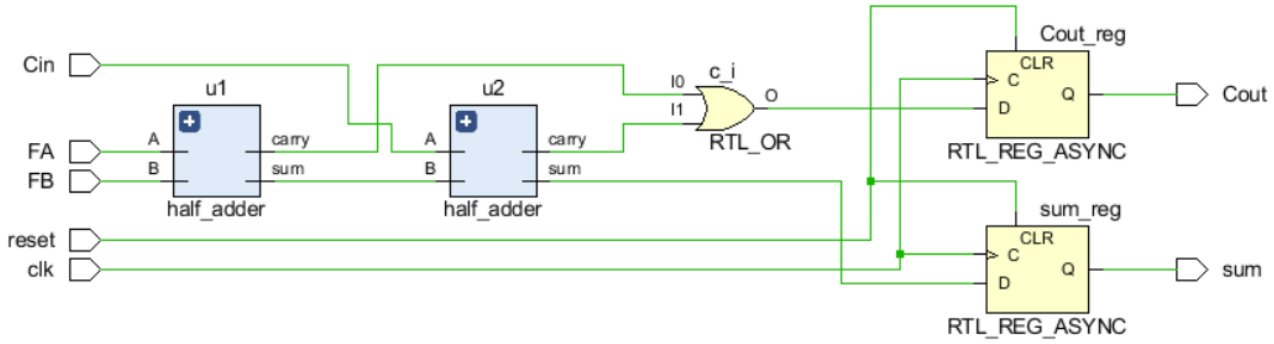
Πρόκειται για έναν (1) LUT (Lookup Table) και πέντε (5) I/O.

Η προσομοίωση (**simulation**) του **ακολουθιακού** κυκλώματος φαίνεται πιο κάτω:



Παρατηρούμε την προσθήκη ρολογιού και εισόδου μηδενισμού (reset) η οποία ενεργοποιείται για είσοδο ‘1’. Τα αποτελέσματα είναι ορθά όπως και πριν.

Το **RTL schematic** του πλήρη αθροιστή παρουσιάζεται παρακάτω:



Critical Path

Reports | Design Runs | Timing

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 1	∞	2	2	1	Cout_reg/C	Cout	4.076	3.276	0.800	∞	
Path 2	∞	2	2	1	sum_reg/C	sum	4.076	3.276	0.800	∞	
Path 3	∞	2	3	2	FA	Cout_reg/D	1.932	1.132	0.800	∞	input port clock
Path 4	∞	2	3	2	Cin	sum_reg/D	1.906	1.106	0.800	∞	input port clock
Path 5	∞	1	2	2	reset	Cout_reg/CLR	1.782	0.982	0.800	∞	input port clock
Path 6	∞	1	2	2	reset	sum_reg/CLR	1.782	0.982	0.800	∞	input port clock

Παρατηρούμε την ύπαρξη δύο κρίσιμων μονοπατιών με την ίδια μέγιστη συνολική καθυστέρηση, τα οποία είναι τα Path 1 και Path 2.

Κατανάλωση πόρων του FPGA

Πιο κάτω παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

Utilization — Post-Synthesis | Post-Implementation

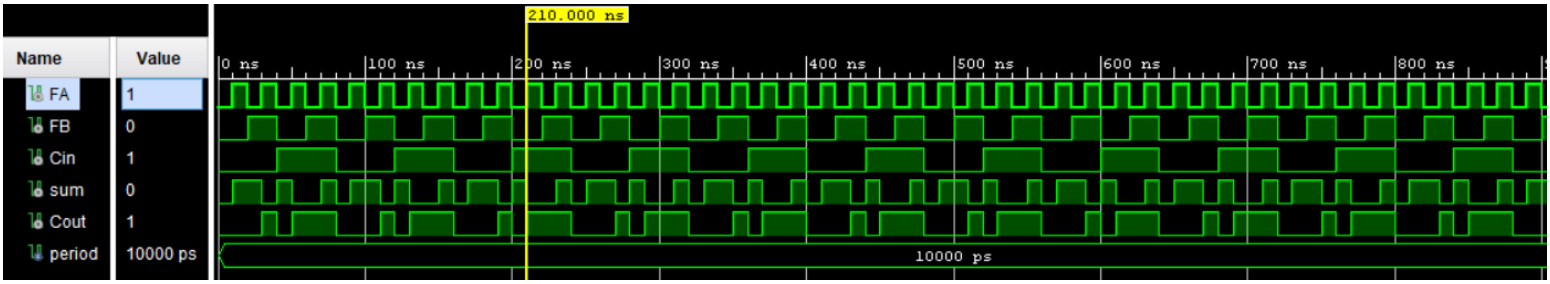
Graph | Table

Resource	Utilization	Available	Utilization %
LUT	1	17600	0.01
FF	2	35200	0.01
IO	7	100	7.00
BUFG	1	32	3.13

b) με περιγραφή συμπεριφοράς (Behavioral) και χρησιμοποιώντας τον τελεστή '+' της VHDL. Να σχεδιαστεί συνδυαστικό και ακολουθιακό κύκλωμα.

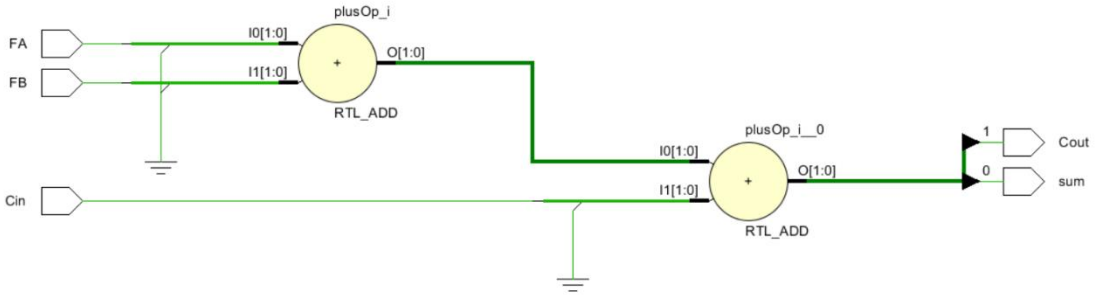
Οι **κώδικες** του ερωτήματος και τα αντίστοιχα **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο “2b”.

Η προσομοίωση (**simulation**) του **συνδυαστικού (Behavioral)** κυκλώματος φαίνεται πιο κάτω:



Τα αποτελέσματα είναι και πάλι αντίστοιχα με τον πίνακα αληθείας του πλήρη αθροιστή που παρατέθηκε προηγουμένως.

Το **RTL schematic** του πλήρη αθροιστή παρουσιάζεται παρακάτω:



Παρατηρούμε ότι το κύκλωμα είναι διαφορετικό από το αντίστοιχο συνδιαστικό κύκλωμα του ερωτήματος 2a) αφού σε αυτή την περίπτωση δεν χρησιμοποιήθηκαν “Half adders” για τη σύνθεση του κυκλώματος, αλλά ο τελεστής “+”.

Critical Path

Reports | Design Runs | **Timing**

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 1	∞	3	4	2	Cin	Cout	5.377	3.778	1.599	∞	input port clock
Path 2	∞	3	4	2	Cin	sum	5.351	3.752	1.599	∞	input port clock

Το Path 1 είναι το κρίσιμο μονοπάτι.

Κατανάλωση πόρων του FPGA

Δεξιά παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

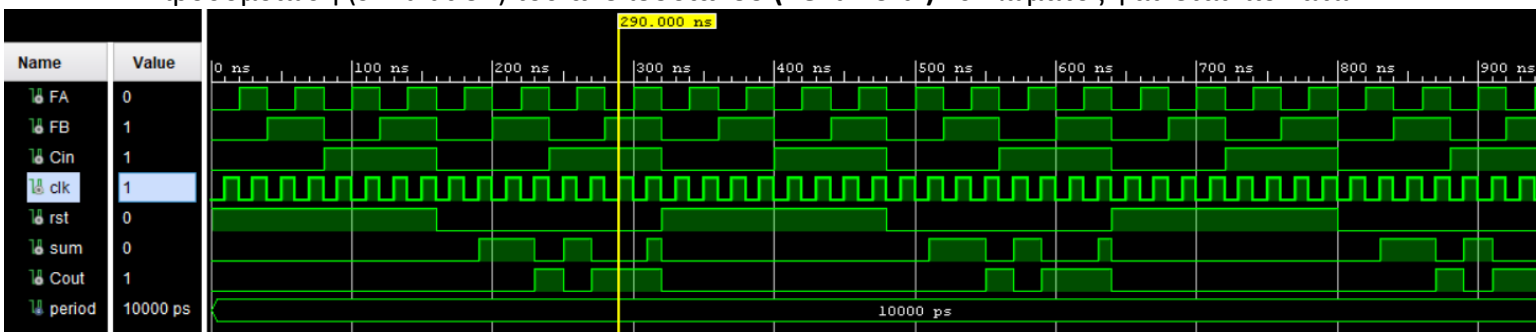
Γίνεται χρήση των ίδιων πόρων με το αντίστοιχο συνδιαστικό κύκλωμα του ερωτήματος 2a).

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

Graph | **Table**

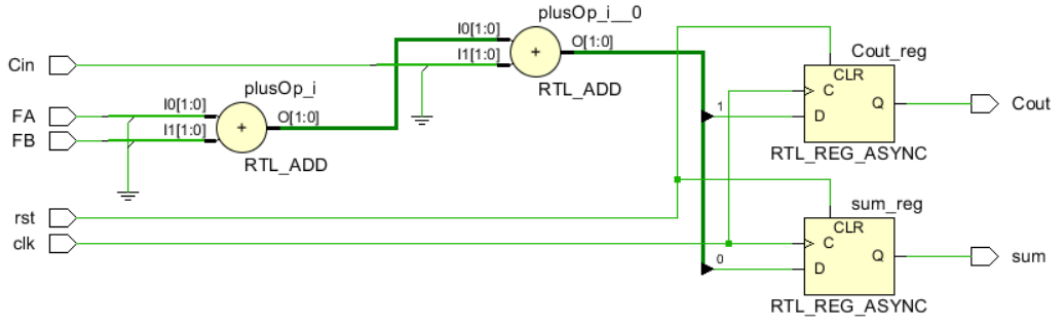
Resource	Utilization	Available	Utilization %
LUT	1	17600	0.01
IO	5	100	5.00

Η προσομοίωση (**simulation**) του **ακολουθιακού (Behavioral)** κυκλώματος φαίνεται πιο κάτω:



Παρατηρούμε εκ νέου τα ορθά αποτελέσματα που προκύπτουν, καθώς και την ύπαρξη ρολογιού και εισόδου μηδενισμού (reset) που ενεργοποιείται στην τιμή ‘1’.

Το **RTL schematic** του πλήρη αθροιστή παρουσιάζεται παρακάτω:



Παρατηρείται η προσθήκη 2 D-FF για την λειτουργία των νέων εισόδων clk και reset.

Critical Path

Reports | Design Runs | Timing

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 1	∞	2	2	1	Cout_reg/C	Cout	4.076	3.276	0.800	∞	
Path 2	∞	2	2	1	sum_reg/C	sum	4.076	3.276	0.800	∞	
Path 3	∞	2	3	2	Cin	Cout_reg/D	1.932	1.132	0.800	∞	input port clock
Path 4	∞	2	3	2	Cin	sum_reg/D	1.906	1.106	0.800	∞	input port clock
Path 5	∞	1	2	2	rst	Cout_reg/CLR	1.782	0.982	0.800	∞	input port clock
Path 6	∞	1	2	2	rst	sum_reg/CLR	1.782	0.982	0.800	∞	input port clock

Παρατηρούμε την ύπαρξη των ίδιων δύο κρίσιμων μονοπατιών, Path 1 και Path 2, τα οποία προέκυψαν και στο αντίστοιχο κύκλωμα της Structural περιγραφής.

Κατανάλωση πόρων του FPGA

Δεξιά παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

Γίνεται χρήση των ίδιων πόρων με το αντίστοιχο ακολουθιακό κύκλωμα του ερωτήματος 2a).

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

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	1	17600	0.01
FF	2	35200	0.01
IO	7	100	7.00
BUFG	1	32	3.13

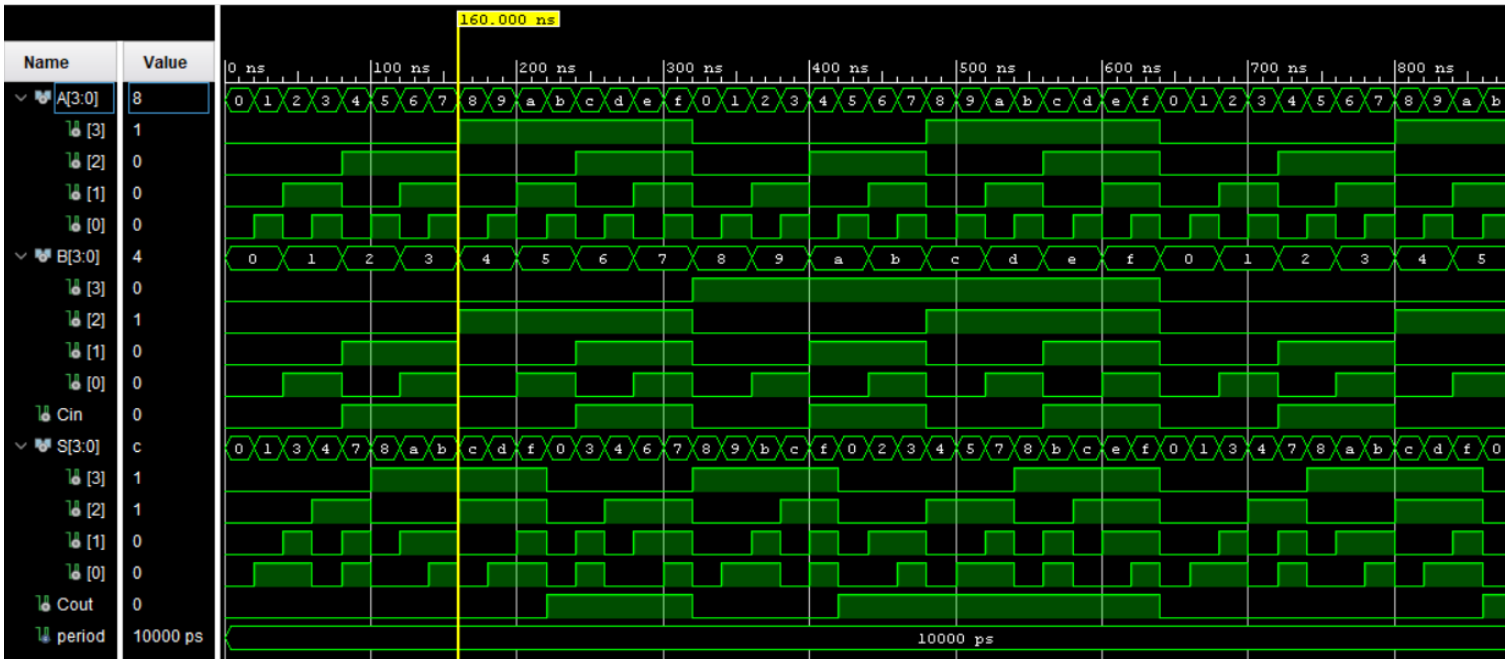
3) Να υλοποιήσετε Παράλληλο Αθροιστή Διάδοσης Κρατουμένου (Ripple-Carry Adder - RCA) των 4 bits (4-bit RCA) με περιγραφή δομής (Structural). Συγκεκριμένα:

α) βασιζόμενοι στην δομική μονάδα του συνδυαστικού FA του Ζητήματος 2.a ή 2.b, να σχεδιάσετε έναν συνδυαστικό 4-bit RCA.

Η ακόλουθη υλοποίηση έγινε με βάση την δομική μονάδα του συνδιαστικού FA του ερωτήματος 2α).

Οι **κώδικες** του ερωτήματος και τα αντίστοιχα **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο "3α".

Η προσομοίωση (**simulation**) του **συνδυαστικού 4-bit-RCA** κυκλώματος φαίνεται πιο κάτω:

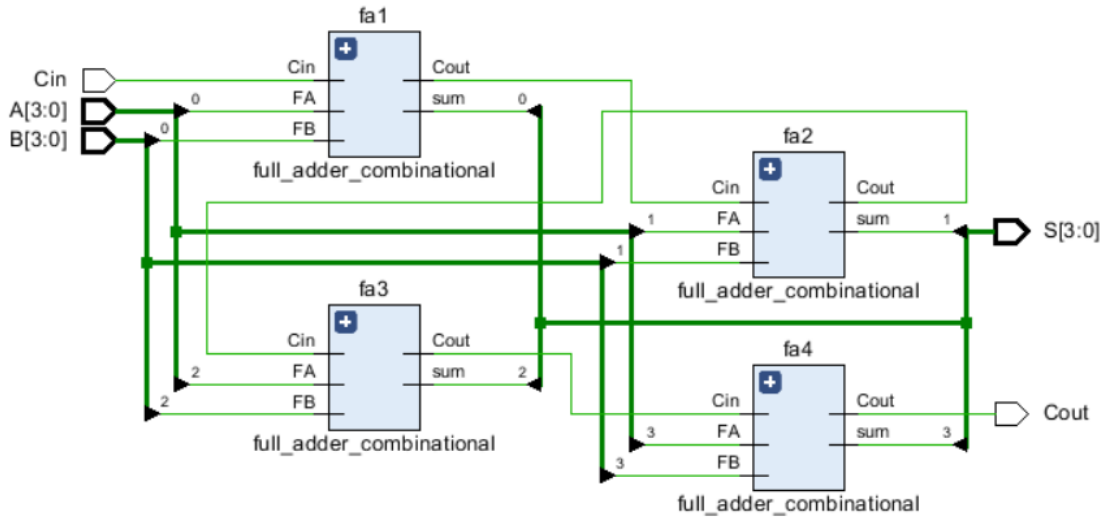


Για την απόδειξη της ορθότητας της υλοποίησης μπορούμε να δούμε μερικά από τα αποτελέσματα της προσομοίωσης:

1. Για είσοδο A = $(1)_{16}$ = $(0001)_2$, B = $(0)_{16}$ = $(0000)_2$ και Cin = 0 παίρνουμε τις εξόδους S = $(1)_{16}$ = $(0001)_2$ και Cout = 0 → Όρθο
2. Για είσοδο A = $(3)_{16}$ = $(0011)_2$, B = $(1)_{16}$ = $(0001)_2$ και Cin = 0 παίρνουμε τις εξόδους S = $(4)_{16}$ = $(0100)_2$ και Cout = 0 → Όρθο
3. Για είσοδο A = $(6)_{16}$ = $(0110)_2$, B = $(3)_{16}$ = $(0011)_2$ και Cin = 1 παίρνουμε τις εξόδους S = $(a)_{16}$ = $(1010)_2$ και Cout = 0 → Όρθο
4. Για είσοδο A = $(8)_{16}$ = $(1000)_2$, B = $(4)_{16}$ = $(0100)_2$ και Cin = 0 παίρνουμε τις εξόδους S = $(c)_{16}$ = $(1100)_2$ και Cout = 0 → Όρθο
5. Για είσοδο A = $(b)_{16}$ = $(1011)_2$, B = $(5)_{16}$ = $(0101)_2$ και Cin = 0 παίρνουμε τις εξόδους S = $(0)_{16}$ = $(0000)_2$ και Cout = 1 → Όρθο

Επομένως η υλοποίηση μας είναι ορθή.

Το **RTL schematic** του παράλληλου αθροιστή διάδοσης κρατουμένου παρουσιάζεται παρακάτω:



Το κύκλωμα προκύπτει από τον συνδιασμό τεσσάρων (4) πλήρων αθροιστών, όπως υλοποιήθηκαν στο ερώτημα 2a).

Critical Path

Reports | Design Runs | **Timing**

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 1	∞	4	5	3	A[1]	S[2]	5.970	3.904	2.066	∞	input port clock
Path 2	∞	4	5	3	A[1]	S[3]	5.970	3.904	2.066	∞	input port clock
Path 3	∞	4	5	3	A[1]	Cout	5.964	3.898	2.066	∞	input port clock
Path 4	∞	3	4	3	Cin	S[0]	5.351	3.752	1.599	∞	input port clock
Path 5	∞	3	4	3	B[0]	S[1]	5.351	3.752	1.599	∞	input port clock

Παρατηρούμε την ύπαρξη δύο κρίσιμων μονοπατιών με την ίδια μέγιστη συνολική καθυστέρηση, τα οποία είναι τα Path 1 και Path 2.

Κατανάλωση πόρων του FPGA

Πιο κάτω παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

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

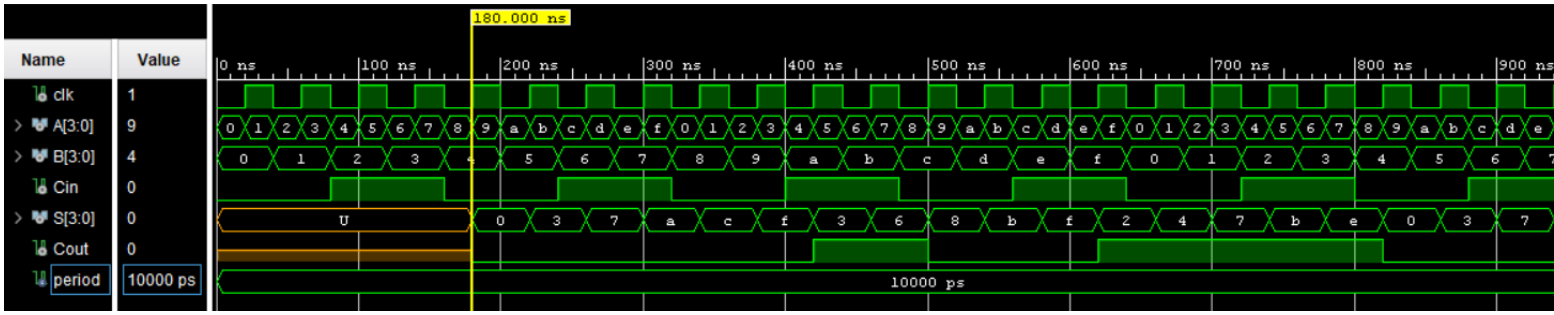
Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	4	17600	0.02
IO	14	100	14.00

b) βασιζόμενοι στην δομική μονάδα του ακολουθιακού FA του Ζητήματος 2.a ή 2.b, να σχεδιάσετε έναν ακολουθιακό 4-bit RCA με χρήση της τεχνικής Pipeline. Το κύκλωμα θα πρέπει να τροφοδοτείται με ένα διαφορετικό ζεύγος εισόδων σε κάθε κύκλο ρολογιού και να δίνει αντίστοιχα ορθό αποτέλεσμα σε κάθε κύκλο ρολογιού έπειτα από κάποια αρχική καθυστέρηση $T_{latency}$. Να κάνετε χρήση επιπλέον λογικής που θεωρείτε απαραίτητη.

Η ακόλουθη υλοποίηση έγινε με βάση την δομική μονάδα του συνδιαστικού FA του ερωτήματος 2a).
Οι **κώδικες** του ερωτήματος και τα αντίστοιχα **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο "3b".

Η προσομοίωση (**simulation**) του **ακολουθιακού 4-bit-RCA** κυκλώματος φαίνεται πιο κάτω:

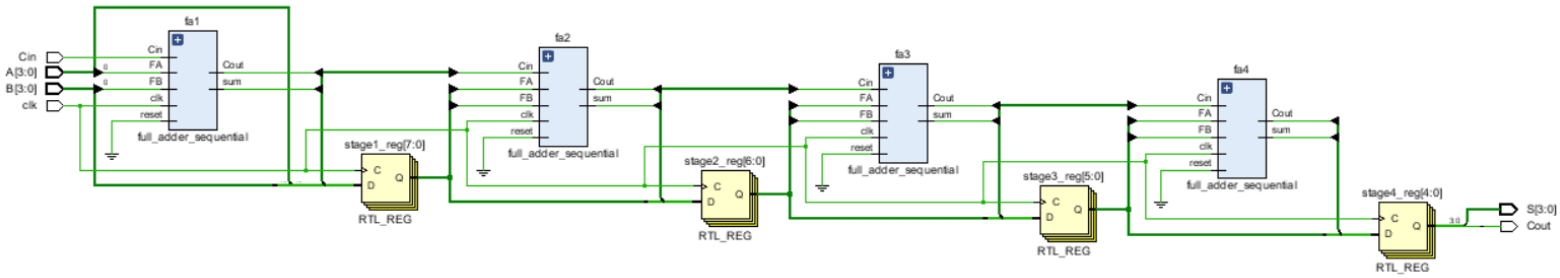


Για την απόδειξη της ορθότητας της υλοποίησης μπορούμε να δούμε μερικά από τα αποτελέσματα της προσομοίωσης:

1. Για είσοδο A = $(0)_{10}$ = $(0)_{16}$ = $(0000)_2$, B = $(0)_{10}$ = $(0)_{16}$ = $(0000)_2$ και Cin = 0 παίρνουμε, μετά από 4 κύκλους και στη θετική ακμή του ρολογιού, τις εξόδους S = $(0)_{10}$ = $(0)_{16}$ = $(0000)_2$ και Cout = 0 → Όρθο $(0)_{10}$
2. Για είσοδο A = $(2)_{10}$ = $(2)_{16}$ = $(0010)_2$, B = $(1)_{10}$ = $(1)_{16}$ = $(0001)_2$ και Cin = 0 παίρνουμε, μετά από 4 κύκλους και στη θετική ακμή του ρολογιού, τις εξόδους S = $(12)_{10}$ = $(3)_{16}$ = $(0011)_2$ και Cout = 0 → Όρθο $(3)_{10}$
3. Για είσοδο A = $(4)_{10}$ = $(4)_{16}$ = $(0100)_2$, B = $(2)_{10}$ = $(2)_{16}$ = $(0010)_2$ και Cin = 1 παίρνουμε, μετά από 4 κύκλους και στη θετική ακμή του ρολογιού, τις εξόδους S = $(7)_{10}$ = $(7)_{16}$ = $(0111)_2$ και Cout = 0 → Όρθο $(7)_{10}$
4. Για είσοδο A = $(6)_{10}$ = $(6)_{16}$ = $(0110)_2$, B = $(3)_{10}$ = $(3)_{16}$ = $(0011)_2$ και Cin = 1 παίρνουμε, μετά από 4 κύκλους και στη θετική ακμή του ρολογιού, τις εξόδους S = $(10)_{10}$ = $(a)_{16}$ = $(1010)_2$ και Cout = 0 → Όρθο $(10)_{10}$
5. Για είσοδο A = $(12)_{10}$ = $(c)_{16}$ = $(1100)_2$, B = $(6)_{10}$ = $(6)_{16}$ = $(0110)_2$ και Cin = 1 παίρνουμε, μετά από 4 κύκλους και στη θετική ακμή του ρολογιού, τις εξόδους S = $(3)_{10}$ = $(3)_{16}$ = $(0011)_2$ και Cout = 1 → Όρθο $(13)_{10}$

Παρατηρούμε ότι σε κάθε κύκλο ρολογιού η είσοδος τροφοδοτείται με ένα νέο ζεύγος εισόδων, του οποίου το ορθό αποτέλεσμα προκύπτει έπειτα από 4 κύκλους ρολογιού. Με εξαίρεση τους 4 πρώτους κύκλους ($T_{latency}$), σε κάθε επόμενο κύκλο προκύπτει η ορθή έξοδος της αντίστοιχης εισόδου (4 κύκλοι προηγουμένως).
Επομένως η υλοποίηση μας είναι ορθή.

Το **RTL schematic** του παράλληλου αθροιστή διάδοσης κρατουμένου παρουσιάζεται παρακάτω:



Το κύκλωμα προκύπτει από τον συνδιασμό τεσσάρων (4) πλήρων αθροιστών, όπως υλοποιήθηκαν στο ερώτημα 2a), καθώς επίσης και τεσσάρων (4) D flip-flop τα οποία προκαλούν την επιθυμητή καθυστέρηση.

Critical Path

Reports | Design Runs | **Timing**

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 1	∞	2	2	1	stage4_reg[4]/C	Cout	4.076	3.276	0.800	∞	
Path 2	∞	2	2	1	stage4_reg[3]/C	S[3]	4.076	3.276	0.800	∞	
Path 3	∞	2	2	1	stage4_reg[0]/C	S[0]	4.058	3.258	0.800	∞	
Path 4	∞	2	2	1	stage4_reg[1]/C	S[1]	4.058	3.258	0.800	∞	
Path 5	∞	2	2	1	stage4_reg[2]/C	S[2]	4.058	3.258	0.800	∞	
Path 6	∞	2	3	2	Cin	stage3_reg[0]_srl4/D	2.239	1.106	1.133	∞	input port clock
Path 7	∞	2	3	2	Cin	fa1/Cout_reg/D	1.906	1.106	0.800	∞	input port clock
Path 8	∞	2	2	2	stage1_reg[2]/C	stage3_reg[1]_srl3/D	1.836	0.751	1.085	∞	
Path 9	∞	2	2	2	stage2_reg[3]/C	stage3_reg[2]_srl2/D	1.836	0.751	1.085	∞	
Path 10	∞	1	2	1	A[1]	stage1_reg[2]/D	1.782	0.982	0.800	∞	input port clock

Παρατηρούμε την ύπαρξη δύο κρίσιμων μονοπατιών με την ίδια μέγιστη συνολική καθυστέρηση, τα οποία είναι τα Path 1 και Path 2.

Κατανάλωση πόρων του FPGA

Πιο κάτω παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

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

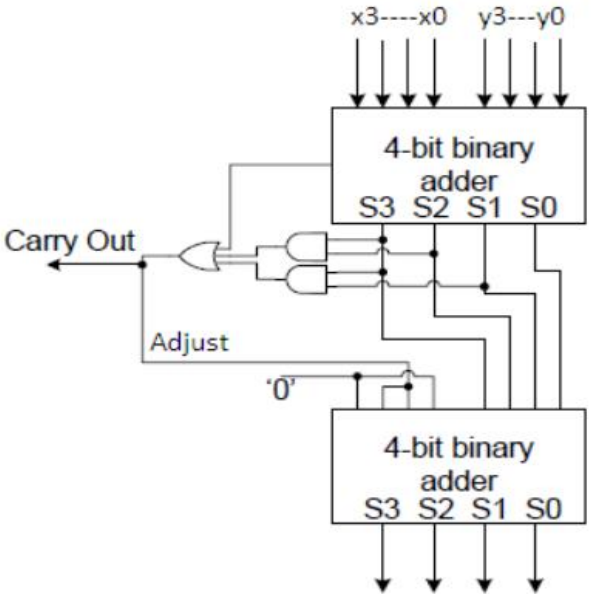
Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	8	17600	0.05
LUTRAM	3	6000	0.05
FF	22	35200	0.06
IO	15	100	15.00
BUFG	1	32	3.13

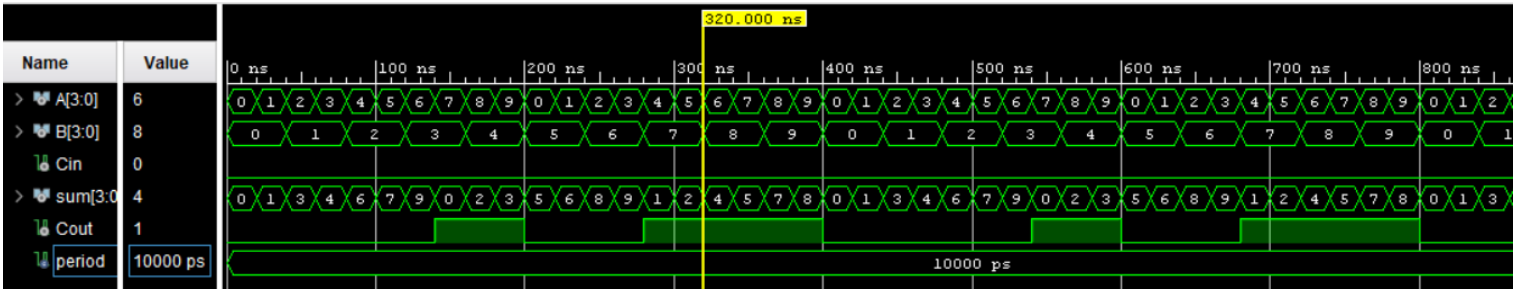
4) Να υλοποιήσετε έναν συνδυαστικό BCD Πλήρη Αθροιστή (BCD Full Adder - BCD FA) με περιγραφή δομής (Structural). Να χρησιμοποιήσετε τη δομική μονάδα που υλοποιήθηκε στο Ζήτημα 3.α, οποιαδήποτε συνδυαστική δομική μονάδα από τα προηγούμενα ερωτήματα, καθώς και επιπλέον λογική που θεωρείτε απαραίτητη.

Οι **κώδικες** του ερωτήματος και τα αντίστοιχα **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο "4".

Ακολουθήσαμε το παρακάτω κύκλωμα το οποίο μελετήσαμε στην λογική σχεδίαση:



Η προσομοίωση (**simulation**) του **συνδιαστικού BCD** κυκλώματος φαίνεται πιο κάτω:



Στον BCD κώδικα ο μέγιστος δεκαδικός αριθμός του οποίου μπορεί να γίνει αναπαράσταση είναι το 9. Επομένως μπορούμε να αθροίσουμε δύο ψηφία (0..9), και το μέγιστο άθροισμα που μπορεί να προκύψει είναι το 18 (9+9). Το αποτέλεσμα προκύπτει σε μια έξοδο sum 4bit (BCD) και σε μια έξοδο Cout (κρατούμενο).

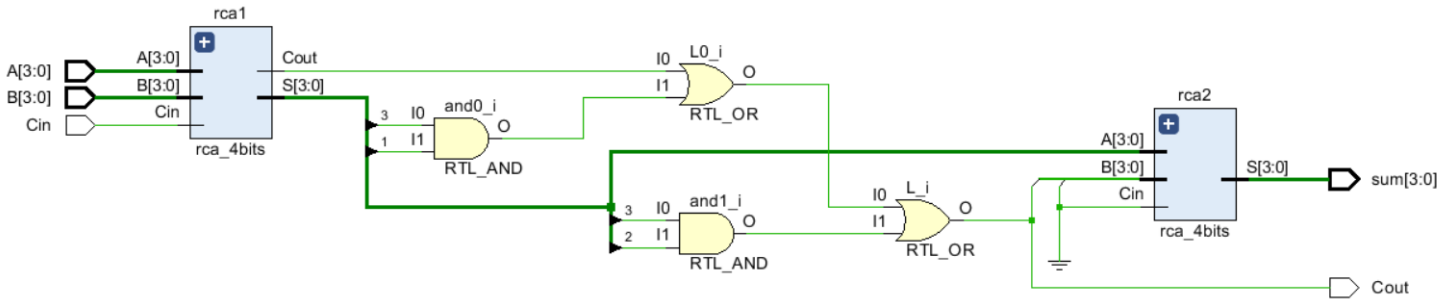
Για την απόδειξη της ορθότητας της υλοποίησης μπορούμε να δούμε μερικά από τα αποτελέσματα της προσομοίωσης (το Cin τέθηκε ίσο με 0 σε κάθε περίπτωση):

1. Για είσοδο A = $(0)_{10}$ = $(0000)_{BCD}$ = $(0000)_2$ και B = $(0)_{10}$ = $(0000)_{BCD}$ = $(0000)_2$ παίρνουμε τις εξόδους S = $(0)_{10}$ = $(0000)_{BCD}$ = $(0000)_2$ και Cout = 0 → Όρθο $(0)_{10}$
2. Για είσοδο A = $(2)_{10}$ = $(0010)_{BCD}$ = $(0010)_2$ και B = $(1)_{10}$ = $(0001)_{BCD}$ = $(0001)_2$ παίρνουμε τις εξόδους S = $(3)_{10}$ = $(0011)_{BCD}$ = $(0011)_2$ και Cout = 0 → Όρθο $(3)_{10}$

3. Για είσοδο A = $(4)_{10}$ = $(0100)_{BCD}$ = $(0100)_2$ και B = $(2)_{10}$ = $(0010)_{BCD}$ = $(0010)_2$ παίρνουμε τις εξόδους S = $(6)_{10}$ = $(0110)_{BCD}$ = $(0110)_2$ και Cout = 0 → Όρθο $(6)_{10}$
4. Για είσοδο A = $(7)_{10}$ = $(0111)_{BCD}$ = $(0111)_2$ και B = $(3)_{10}$ = $(0011)_{BCD}$ = $(0011)_2$ παίρνουμε τις εξόδους S = $(0)_{10}$ = $(0000)_{BCD}$ = $(0000)_2$ και Cout = 1 → Όρθο $(10)_{10}$
5. Για είσοδο A = $(9)_{10}$ = $(1001)_{BCD}$ = $(1001)_2$ και B = $(4)_{10}$ = $(0100)_{BCD}$ = $(0110)_2$ παίρνουμε τις εξόδους S = $(3)_{10}$ = $(0011)_{BCD}$ = $(0011)_2$ και Cout = 1 → Όρθο $(13)_{10}$

Προκύπτουν τα ορθά αποτελέσματα, επομένως έχουμε ορθή υλοποίηση.

Το **RTL schematic** του παράλληλου αθροιστή διάδοσης κρατουμένου παρουσιάζεται παρακάτω:



Το κύκλωμα προκύπτει από τον συνδιασμό δύο (2) RCA, όπως υλοποιήθηκαν στο ερώτημα 3a) και κάποιων επιπλέων πυλών.

Critical Path

Reports | Design Runs | **Timing**

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 1	∞	4	5	4	B[0]	Cout	5.948	3.876	2.072	∞	input port clock
Path 2	∞	4	5	4	B[0]	sum[1]	5.948	3.876	2.072	∞	input port clock
Path 3	∞	4	5	4	B[0]	sum[2]	5.948	3.876	2.072	∞	input port clock
Path 4	∞	4	5	4	B[0]	sum[3]	5.948	3.876	2.072	∞	input port clock
Path 5	∞	3	4	3	Cin	sum[0]	5.351	3.752	1.599	∞	input port clock

Παρατηρούμε την ύπαρξη τεσσάρων κρίσιμων μονοπατιών με την ίδια μέγιστη συνολική καθυστέρηση, τα οποία είναι τα Path 1, Path 2, Path 3 και Path 4.

Κατανάλωση πόρων του FPGA

Πιο κάτω παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

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

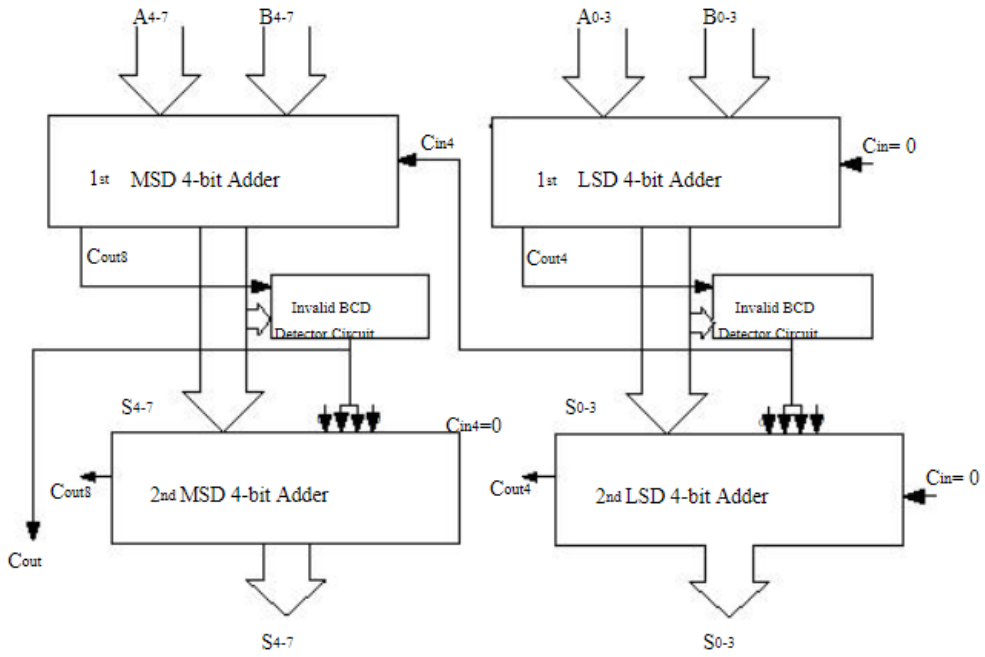
Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	6	17600	0.03
IO	14	100	14.00

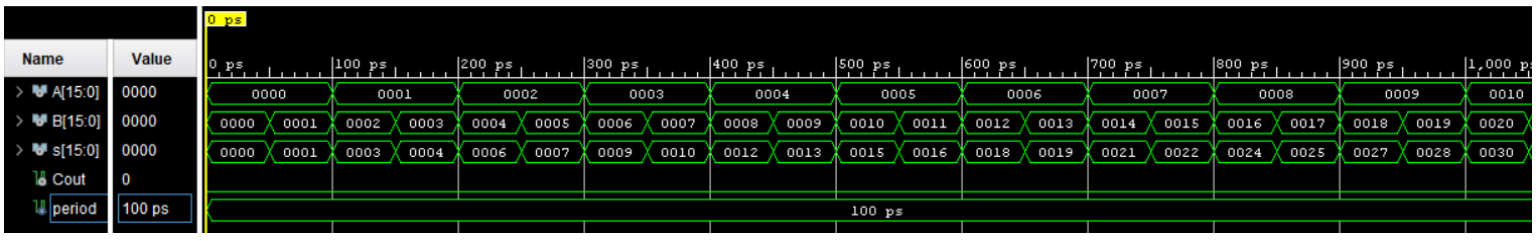
5) ***Να υλοποιήσετε έναν Παράλληλο BCD Αθροιστή των 4 ψηφίων (4-BCD Parallel Adder - 4-BCD PA) με περιγραφή δομής (Structural), βασιζόμενοι στην δομική μονάδα του Ζητήματος 4.***

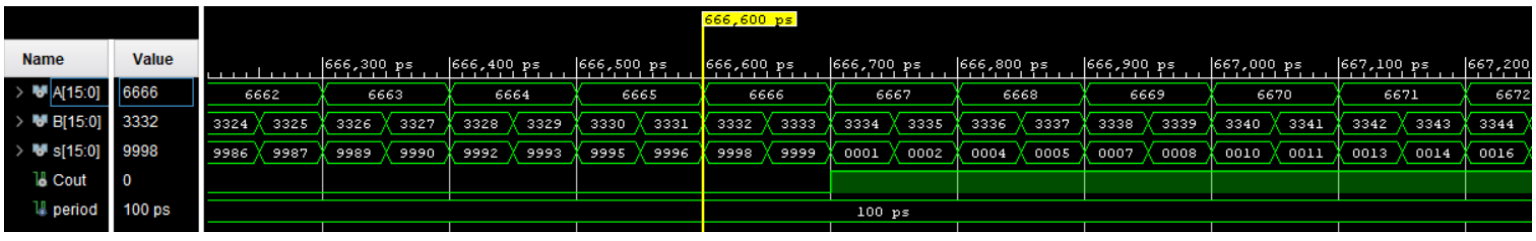
Οι **κώδικες** του ερωτήματος και τα αντίστοιχα **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο "5".

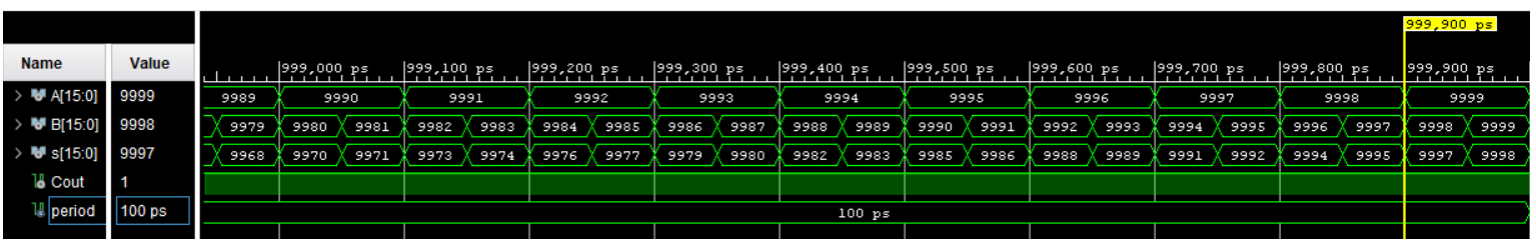
Ακολουθήσαμε την παρακάτω λογική η οποία αθροίζει 2 BCD ψηφία και στην συνέχεια το εφαρμόσαμε στις ανάγκες μας για να αθροίσουμε 4 ψηφία.



Η προσομοίωση (**simulation**) του **συνδιαστικού** κυκλώματος **BCD 4 ψηφίων** φαίνεται πιο κάτω:







Σε αυτή την περίπτωση μπορούμε να αθροίσουμε αριθμούς από το $(0)_{10}$ = $(0000)_{BCD}$ μέχρι το $(9999)_{10}$ = $(1001\ 1001\ 1001\ 1001)_{BCD}$. Γίνεται λοιπόν αντιληπτό ότι το μέγιστο άθροισμα που μπορεί να προκύψει είναι το 19998 (9999+9999).

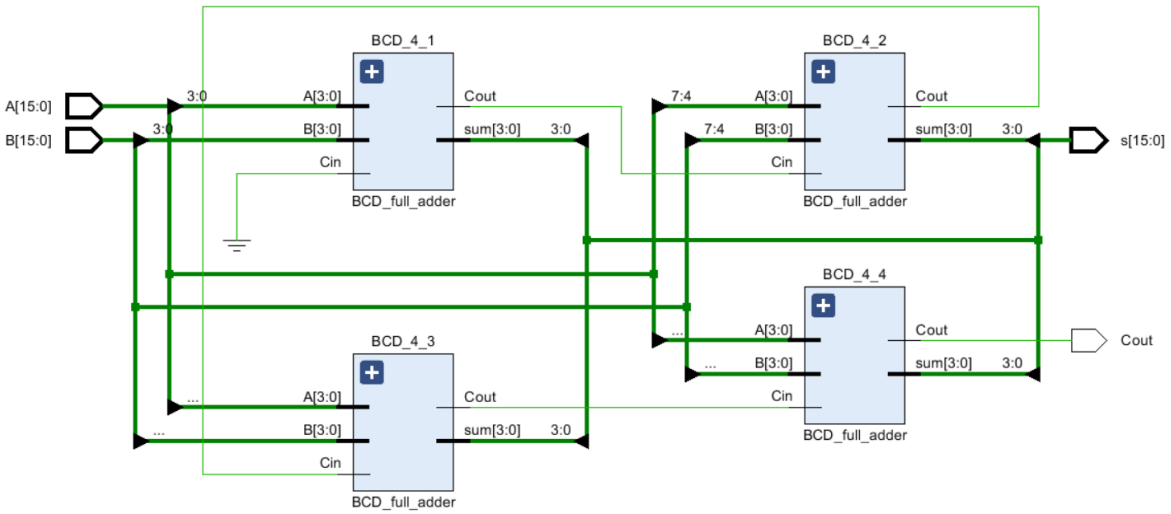
Οι περιπτώσεις είναι πάρα πολλές και γι' αυτό παραθέτουμε αποτελέσματα από την αρχή, το μέσο και το τέλος της προσομοίωσης.

Για την απόδειξη της ορθότητας της υλοποίησης μπορούμε να δούμε μερικά από τα αποτελέσματα της προσομοίωσης:

1. Για είσοδο A = $(0)_{10}$ = $(0000)_{BCD}$ = (0000) και B = $(0)_{10}$ = $(0000)_{BCD}$ = (0000) παίρνουμε τις εξόδους S = $(0)_{10}$ = $(0000)_{BCD}$ = (0000) και Cout = 0 → Όρθο $(0)_{10}$
2. Για είσοδο A = $(5)_{10}$ = $(0101)_{BCD}$ = (0005) και B = $(11)_{10}$ = $(0001\ 0001)_{BCD}$ = (0011) παίρνουμε τις εξόδους S = $(16)_{10}$ = $(1010\ 0110)_{BCD}$ = (0016) και Cout = 0 → Όρθο $(16)_{10}$
3. Για είσοδο A = $(6666)_{10}$ = $(0110\ 0110\ 0110\ 0110)_{BCD}$ = (6666) και B = $(3332)_{10}$ = $(0011\ 0011\ 0011\ 0010)_{BCD}$ = (3332) παίρνουμε τις εξόδους S = $(9998)_{10}$ = $(1001\ 1001\ 1001\ 1000)_{BCD}$ = (9998) και Cout = 0 → Όρθο $(9998)_{10}$
4. Για είσοδο A = $(6668)_{10}$ = $(0110\ 0110\ 0110\ 1000)_{BCD}$ = (6668) και B = $(3336)_{10}$ = $(0011\ 0011\ 0011\ 0110)_{BCD}$ = (3336) παίρνουμε τις εξόδους S = $(4)_{10}$ = $(0100)_{BCD}$ = (0004) και Cout = 1 → Όρθο $(10004)_{10}$
5. Για είσοδο A = $(9999)_{10}$ = $(1001\ 1001\ 1001\ 1001)_{BCD}$ = (9999) και B = $(9998)_{10}$ = $(1001\ 1001\ 1001\ 1001)_{BCD}$ = (9998) παίρνουμε τις εξόδους S = $(9997)_{10}$ = $(1001\ 1001\ 1001\ 0111)_{BCD}$ = (9997) και Cout = 1 → Όρθο $(19997)_{10}$

Προκύπτουν τα ορθά αποτελέσματα, επομένως έχουμε ορθή υλοποίηση.

Το **RTL schematic** του παράλληλου αθροιστή διάδοσης κρατουμένου παρουσιάζεται παρακάτω:



Το κύκλωμα προκύπτει από τον συνδυασμό τεσσάρων (4) BCD πλήρων αθροιστών, όπως υλοποιήθηκαν στο ερώτημα 4).

Critical Path

Reports | Design Runs | **Timing** ×

Unconstrained Paths - NONE - NONE - Setup

Name	Slack ^1	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 1	∞	10	11	7	A[1]	Cout	9.545	4.648	4.897	∞	input port clock
Path 2	∞	10	11	7	A[1]	s[13]	9.545	4.648	4.897	∞	input port clock
Path 3	∞	10	11	7	A[1]	s[14]	9.545	4.648	4.897	∞	input port clock
Path 4	∞	10	11	7	A[1]	s[15]	9.545	4.648	4.897	∞	input port clock
Path 5	∞	9	10	7	A[1]	s[10]	8.948	4.524	4.424	∞	input port clock
Path 6	∞	9	10	7	A[1]	s[12]	8.948	4.524	4.424	∞	input port clock
Path 7	∞	9	10	7	A[1]	s[9]	8.948	4.524	4.424	∞	input port clock
Path 8	∞	8	9	7	A[1]	s[11]	8.347	4.400	3.947	∞	input port clock
Path 9	∞	7	8	7	A[1]	s[5]	7.763	4.276	3.487	∞	input port clock
Path 10	∞	7	8	7	A[1]	s[6]	7.763	4.276	3.487	∞	input port clock

Παρατηρούμε την ύπαρξη τεσσάρων (4) κρίσιμων μονοπατιών με την ίδια μέγιστη συνολική καθυστέρηση, τα οποία είναι τα Path 1, Path 2, Path 3 και Path 4.

Κατανάλωση πόρων του FPGA

Πιο κάτω παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

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

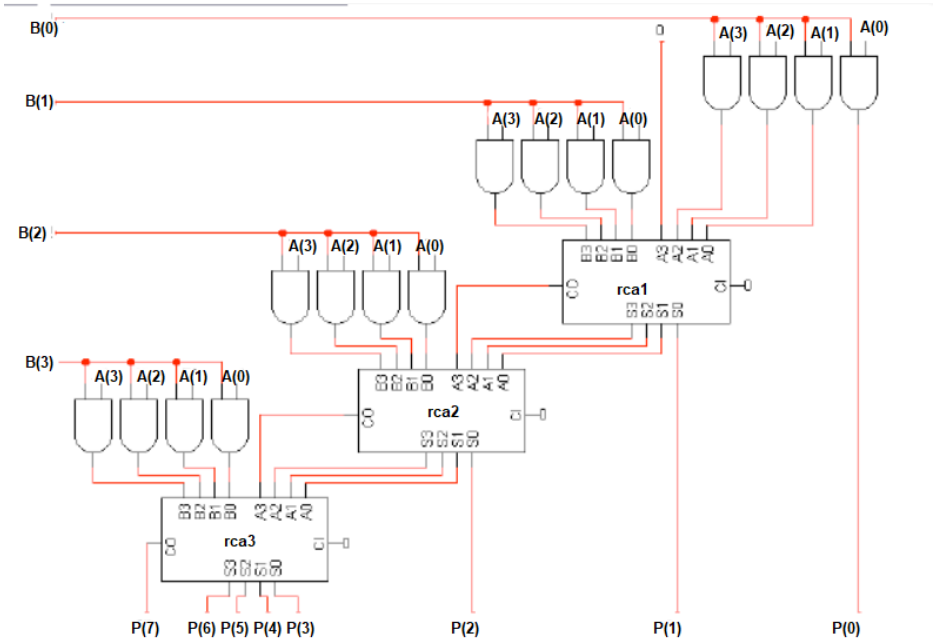
Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	24	17600	0.14
IO	49	100	49.00

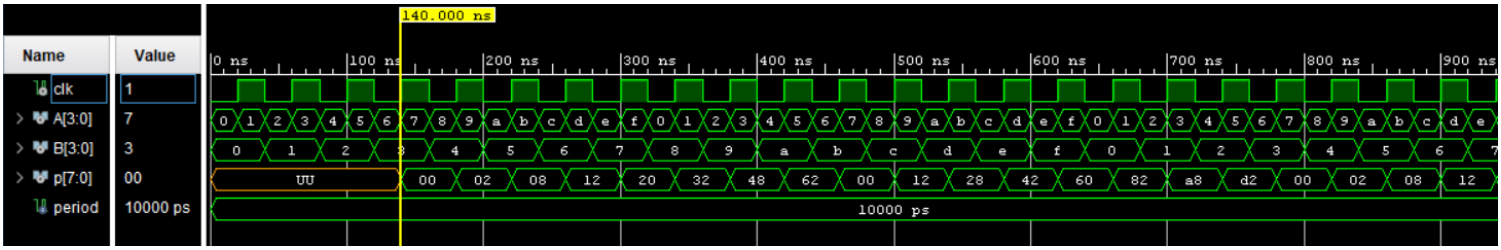
6) ***Να υλοποιήσετε ένα συστολικό (είδος pipeline) Πολλαπλασιαστή Διάδοσης Κρατουμένων των 4 bits κάνοντας χρήση ακολουθιακών FA που σχεδιάσατε στο Ζήτημα 2.a ή 2.b. Το κύκλωμα θα πρέπει να τροφοδοτείται με ένα διαφορετικό ζεύγος εισόδων σε κάθε κύκλο ρολογιού και να δίνει αντίστοιχα ορθό αποτέλεσμα σε κάθε κύκλο ρολογιού έπειτα από κάποια αρχική καθυστέρηση*** $T_{latency}$.

Οι **κώδικες** του ερωτήματος και τα αντίστοιχα **testbench** βρίσκονται σε αρχεία VHDL στον φάκελο "6".

Η λογική που ακολουθήσαμε είναι η ακόλουθη:



Η προσομοίωση (**simulation**) του κυκλώματος **πολλαπλασιαστή** φαίνεται πιο κάτω:

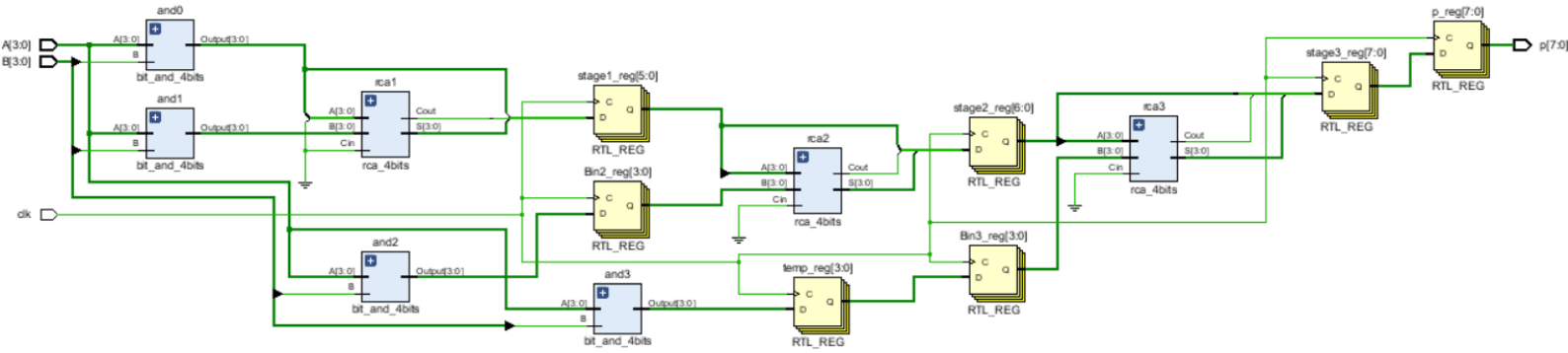


Για την απόδειξη της ορθότητας της υλοποίησης μπορούμε να δούμε μερικά από τα αποτελέσματα της προσομοίωσης:

1. Για είσοδο A = $(0)_{10}$ = $(0)_{16}$ και B = $(0)_{10}$ = $(0)_{16}$ παίρνουμε, μετά από 3 κύκλους και στη θετική ακμή του ρολογιού, την έξοδο p = $(0)_{10}$ = $(0)_{16}$ → Όρθο
2. Για είσοδο A = $(2)_{10}$ = $(2)_{16}$ και B = $(1)_{10}$ = $(1)_{16}$ παίρνουμε, μετά από 3 κύκλους και στη θετική ακμή του ρολογιού, την έξοδο p = $(2)_{10}$ = $(2)_{16}$ → Όρθο
3. Για είσοδο A = $(6)_{10}$ = $(6)_{16}$ και B = $(3)_{10}$ = $(3)_{16}$ παίρνουμε, μετά από 3 κύκλους και στη θετική ακμή του ρολογιού, την έξοδο p = $(18)_{10}$ = $(12)_{16}$ → Όρθο
4. Για είσοδο A = $(8)_{10}$ = $(8)_{16}$ και B = $(4)_{10}$ = $(4)_{16}$ παίρνουμε, μετά από 3 κύκλους και στη θετική ακμή του ρολογιού, την έξοδο p = $(32)_{10}$ = $(20)_{16}$ → Όρθο
5. Για είσοδο A = $(10)_{10}$ = $(a)_{16}$ και B = $(5)_{10}$ = $(5)_{16}$ παίρνουμε, μετά από 3 κύκλους και στη θετική ακμή του ρολογιού, την έξοδο p = $(50)_{10}$ = $(32)_{16}$ → Όρθο

Παρατηρούμε ότι σε κάθε κύκλο ρολογιού η είσοδος τροφοδοτείται με ένα νέο ζεύγος εισόδων, του οποίου το ορθό αποτέλεσμα προκύπτει έπειτα από 3 κύκλους ρολογιού. Με εξαίρεση τους 3 πρώτους κύκλους ($T_{latency}$), σε κάθε επόμενο κύκλο προκύπτει η ορθή έξοδος της αντίστοιχης εισόδου (3 κύκλοι προηγουμένως).
Επομένως η υλοποίηση μας είναι ορθή.

Το **RTL schematic** του πολλαπλασιαστή διάδοσης κρατουμένων παρουσιάζεται παρακάτω:



Critical Path

Reports | Design Runs | Timing

Unconstrained Paths - NONE - NONE - Setup

Name	Slack	Levels	Routes	High Fanout	From	To	Total Delay	Logic Delay	Net Delay	Requirement	Source Clock
Path 1	∞	2	2	1	p_reg[3]/C	p[3]	4.076	3.276	0.800	∞	
Path 2	∞	2	2	1	p_reg[4]/C	p[4]	4.076	3.276	0.800	∞	
Path 3	∞	2	2	1	p_reg[5]/C	p[5]	4.076	3.276	0.800	∞	
Path 4	∞	2	2	1	p_reg[6]/C	p[6]	4.076	3.276	0.800	∞	
Path 5	∞	2	2	1	p_reg[7]/C	p[7]	4.076	3.276	0.800	∞	
Path 6	∞	2	2	1	p_reg[0]/C	p[0]	4.058	3.258	0.800	∞	
Path 7	∞	2	2	1	p_reg[1]/C	p[1]	4.058	3.258	0.800	∞	
Path 8	∞	2	2	1	p_reg[2]/C	p[2]	4.058	3.258	0.800	∞	
Path 9	∞	2	3	4	B[2]	Bin2_reg[0]/R	2.706	1.106	1.600	∞	input port clock
Path 10	∞	2	3	4	B[2]	Bin2_reg[1]/R	2.706	1.106	1.600	∞	input port clock

Παρατηρούμε την ύπαρξη πέντε (5) κρίσιμων μονοπατιών με την ίδια μέγιστη συνολική καθυστέρηση, τα οποία είναι τα Path 1, Path 2, Path 3, Path 4 και Path 5.

Κατανάλωση πόρων του FPGA
Δεξιά παρατηρούμε τους πόρους του FPGA που χρησιμοποιούνται.

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

Graph | **Table**

Resource	Utilization	Available	Utilization %
LUT	18	17600	0.10
LUTRAM	3	6000	0.05
FF	33	35200	0.09
IO	17	100	17.00
BUFG	1	32	3.13

Ερωτήματα θεωρίας:

I. ***Ποια είναι η διαφορά ενός ακολουθιακού κυκλώματος που έχει σύγχρονα σήματα ελέγχου σε σχέση με το ίδιο ακολουθιακό κύκλωμα που έχει ασύγχρονα σήματα ελέγχου (π.χ reset, enable, load, etc.); Πως διαφοροποιείται η λειτουργία τους;***

Στα σύγχρονα ακολουθιακά κυκλώματα , η κατάσταση του κυκλώματος αλλάζει μόνο σε διακριτούς χρόνους, σε απόκριση ενός σήματος ρολογιού. Στα ασύγχρονα ακολουθιακά κυκλώματα, η κατάσταση του κυκλώματος μπορεί να αλλάξει ανά πάσα στιγμή ως απόκριση στην αλλαγή εισόδων.

Στα σύγχρονα κυκλώματα λόγω της καθυστέρησης διάδοσης του σήματος ρολογιού που υπάρχει για να πραγματοποιηθούν όλα τα στοιχεία του κυκλώματος, υπάρχει καθυστέρηση στην ταχύτητα λειτουργίας. Αντιθέτως, στα ασύγχρονα σήματα δεδομένου ότι δεν υπάρχει καθυστέρηση σήματος ρολογιού, τα αποτελέσματα προκύπτουν άμεσα.

II. ***Για ποιο λόγο πραγματοποιείται η σύνθεση του κυκλώματος αφού ο σχεδιαστής έχει υλοποιήσει το κύκλωμα του με VHDL και έχει ελέγξει τη ορθή λειτουργία του με χρήση testbench; Τι παίρνει ως είσοδο το στάδιο της σύνθεσης και τι παράγει ως έξοδο;***

Ο λόγος που ο σχεδιαστής παρόλο που έχει επαληθεύσει την σωστή λειτουργία του κυκλώματος που έχει περιγράψει μέσω του test bench πραγματοποιεί σύνθεση είναι διότι, εκεί μπορεί να ελέγξει το resource utilization δηλαδή την κατανάλωση πόρων του FPGA και να δει αν μπορεί να βελτιστοποιήσει το πρόγραμμα του για να μειώσει τους πόρους που χρησιμοποιούνται στο FPGA.

Η είσοδος στο στάδιο της σύνθεσης είναι πηγαίος κώδικας RTL(Register Transfer Level) και timing constraints(χρονικοί περιορισμοί), ενώ στην έξοδο παράγεται χαρτογραφημένο στο FPGA το μεταγλωττισμένο VHDL και συγκεκριμένα σε επίπεδο πυλών.

III. ***Να επισημάνετε και να δικαιολογήσετε τις διαφορές που υπάρχουν μεταξύ ενός testbench για συνδυαστικό κύκλωμα και ενός testbench για ακολουθιακό κύκλωμα.***

Ένα συνδυαστικό κύκλωμα ορίζεται ως ανεξάρτητα χρονικά κύκλωμα το οποίο δεν εξαρτάται από προηγούμενες εισόδους για την δημιουργία οποιασδήποτε εξόδου, ενώ ένα ακολουθιακό κυκλώμα εξαρτάται από κύκλους ρολογιού και από τις υπάρχουσες καθώς και τις προηγούμενες εισόδους, για την δημιουργία της εξόδου.

Άρα η διαφορά μεταξύ ενός test bench για ένα ακολουθιακό και σε ένα συνδυαστικό κύκλωμα είναι ότι σε ένα ακολουθιακό κύκλωμα είναι απαραίτητο να ορίσουμε ένα ρολόι πέρα από τις υπόλοιπες εισόδους.

IV. ***Να σχολιάσετε και να αιτιολογήστε τις διαφορές που παρατηρείτε στο κρίσιμο μονοπάτι και στην κατανάλωση πόρων του FPGA για τα κυκλώματα που υλοποιήθηκαν στο ζητούμενο 3.***

Όσον αφορά το κρίσιμο μονοπάτι, παρατηρούμε ότι και στις δύο περιπτώσεις προκύπτουν 2 κρίσιμα μονοπάτια, δηλαδή μονοπάτια με τη μέγιστη καθυστέρηση. Παρατηρούμε όμως ότι η μέγιστη καθυστέρηση στο ζητούμενο 3a) είναι αρκετά μεγαλύτερη από ότι στο ζητούμενο 3b), γεγονός που είναι λογικό αφού στην pipeline υλοποίηση, μετά από κάποια αρχική καθυστέρηση, σε κάθε κύκλο ρολογιού χρειάζεται να πραγματοποιηθεί μόνο η πρόσθεση των 2 τελευταίων bits (τελευταίο bit κάθε εισόδου), αφού οι προηγούμενες

προσθέσεις έχουν ήδη πραγματοποιηθεί προηγουμένως. Συμπεραίνουμε λοιπόν ότι με την pipeline υλοποίηση η συνολική καθυστέρηση που προκύπτει είναι μικρότερη.

Σχετικά με την κατανάλωση πόρων μπορούμε να παρατηρήσουμε τεράστια αύξηση κατανάλωσης στην περίπτωση του pipeline. Ενώ στην περίπτωση 3a) χρησιμοποιούνται μόνο 4 LUTs και 14 IO, στην περίπτωση 3b παρατηρούμε τη χρήση 8 LUTs, 3 LUTRAMs, 22 FF, 15 IO και 1 BUFG.

Μπορούμε λοιπόν να συμπεράνουμε ότι έχουμε να επιλέξουμε ανάμεσα στην ταχύτητα και την κατανάλωση πόρων. Στην περίπτωση του pipeline, επιτυγχάνουμε μικρότερη καθυστέρηση αλλά μεγαλύτερη κατανάλωση πόρων, αντίθετα με την άλλη περίπτωση όπου η κατανάλωση είναι αρκετά μικρή αλλά η υλοποίηση υστερεί στην ταχύτητα.