ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 6

Σκοπός της εργαστηριακής άσκησης είναι μια πρώτη γνωριμία σας με τις διαδικασίες του rapid system prototyping. Θα περιγράψουμε ένα κύκλωμα σε HDL, θα πάρουμε την ισοδύναμη υλοποίησή του σε CLBs μέσω σύνθεσης και τέλος θα φτιάξουμε ένα πρωτότυπο αυτού του κυκλώματος στις αναπτυξιακές μας πλακέτες.

Πριν από οτιδήποτε άλλο, πρέπει να διαβάσετε (ακόμη κι αν το έχετε κάνει ήδη πρέπει να το ξανακάνετε) τα manuals :

(α) xsa-3S-manual-v1_1.pdf

(β) xst-manual-v4_0.pdf

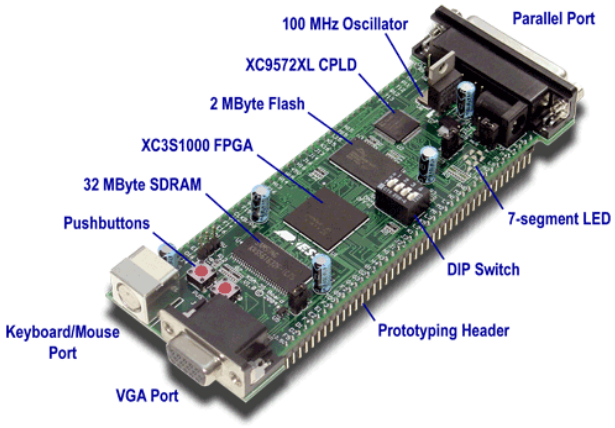
και να έχετε πάντοτε σε χρήση τα Excel αρχεία :

(α) XSA-3S-pins.xls

(β) XSA+XST4-pins.xls

Α. Σύντομη περιγραφή των αναπτυξιακών πλακετών XSA-3S και XSTend v.4.0

Για την πρωτοτυποποίηση των σχεδιασμών μας θα χρησιμοποιήσουμε ένα FPGA χωρητικότητας 1.000.000 ισοδύναμων πυλών, της σειράς SPARTAN 3 της εταιρείας Xilinx. Το FPGA αυτό για εργαστηρικούς λόγους διατίθεται και στη μορφή της αναπτυξιακής πλακέτας XSA-3S που έχει κατασκευάσει η εταιρεία XESS και που απεικονίζεται στην παρακάτω εικόνα :



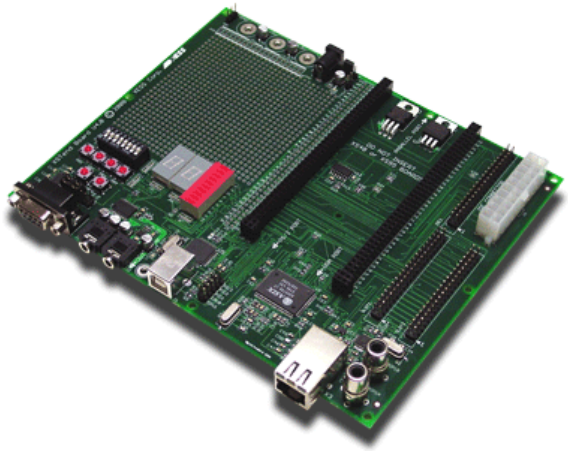
Στην πλακέτα αυτή προσφέρεται εκτός από το FPGA μια σειρά από interfaces διασύνδεσης (VGA, PS/2, Parallel Port) μαζί με τους απαραίτητους υποσχεδιασμούς τους καθώς και μια σειρά από σχεδιασμούς ελέγχου και παρατήρησης (dip switches, push buttons, 7 segment leds). Επίσης παρέχονται μνήμες :

(α) SDRAM, για bulk αποθήκευση ενδιάμεσων αποτελεσμάτων,

(β) Flash, για αποθήκευση κώδικα προγραμματισμού του ολοκληρωμένου ή για reconfiguration

και τέλος ένας προγραμματιζόμενος χρονιστής και ένα CPLD για τον έλεγχο των πηγών προγραμματισμού του FPGA και διάφορων άλλων λειτουργιών. Για όλες τις εργαστηριακές μας ασκήσεις καθώς και για την εξαμηνιαία εργασία σας, θα συνδέσουμε την πλακέτα αυτή με ένα προσωπικό υπολογιστή μέσω της παράλληλης θύρας και θα προγραμματίζουμε το FPGA με κώδικα που θα παράγεται στο PC και θα μεταβιβάζεται στο FPGA μέσω αυτής της σύνδεσης.

Παρότι η πλακέτα αυτή είναι standalone (χρειάζεται μόνο τροφοδοσία για να λειτουργήσει) μπορούμε να επεκτείνουμε επιπλέον τις δυνατότητές της προσκολλώντας την σε μια πλακέτα XSTend v.4.0 (XST4), η οποία μας δίνει επιπλέον interfaces (serial, audio USB, Ethernet, video), πάρα πολλούς ακροδέκτες παρατήρησης (headers) και σχεδιασμούς ελέγχου και παρατήρησης, χώρο για ανάπτυξη των δικών μας κυκλωμάτων και τη δυνατότητα τροφοδοσίας από ATX τροφοδοτικό. Η πλακέτα XST4 φαίνεται στην παρακάτω φωτογραφία:

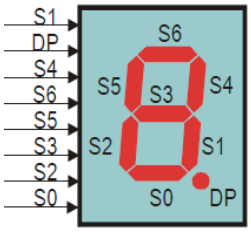


Στο εργαστήριο υπάρχει πάντα ένα ελεγμένο έτοιμο τέτοιο ζεύγος πλακετών προς χρήση, μαζί με το απαραίτητο τροφοδοτικό και το καλώδιο προγραμματισμού. Σε καμμία περίπτωση δε θα πρέπει να προσπαθήσετε να διαχωρίσετε τις δύο αυτές πλακέτες, να αλλάξετε σχήμα τροφοδοσίας ή να αλλάξετε τρόπο προγραμαμτισμού του FPGA.

Στην παρούσα εργαστηριακή άσκηση, θα χρησιμοποιήσουμε μόνο την πλακέτα XSA και θα χρησιμοποιούμε τη πλακέτα XST απλά για τροφοδοσία του συστήματός μας. Σκοπός του 1ου μέρους της άσκησης είναι να περιγράψουμε μια πύλη XOR 4 εισόδων, να τροφοδοτήσουμε τις εισόδους της από τα dip switches (υπενθυμίζεται : της πλακέτας XSA) και να βλέπουμε την έξοδο σαν αριθμό (0 ή 1) στο 7 segment.

Β. HDL περιγραφή του σχεδιασμού

Το κύκλωμα που θα χρειστεί να περιγράψουμε έχει 4 εισόδους (a, b, c, d που θα συνδεθούν στα dip switches) και 8 εξόδους (s[7:0] και dp που θα οδηγούν τα 8 led του 7 segment).



Θεωρούμε ότι η έξοδος s[i] οδηγεί το λαμπάκι Si της παραπάνω εικόνας και ότι η έξοδος dp το DP του 7 segment. Άρα για να βλέπουμε το 0, θα πρέπει να είναι s[0]=s[1]=s[2]=s[4]=s[5]=s[6] = 1 και s[3]=0, ενώ για να βλέπουμε το 1, θα πρέπει να είναι s[1]=s[4]=1 και όλα τα υπόλοιπα 0. Σε κάθε περίπτωση υποθέστε ότι dp=1.

(αρχείο 6/mlx.v)

```
module mlx (a, b, c, d, s, dp);
  input a, b, c, d;
  output [6:0] s;
  output       dp;
  wire o;

  assign o = a^b^c^d;
  assign s[6:0] = o ? 7'b0010010 : 7'b1110111;
  assign dp = 1;
endmodule
```

Γ. Περιγραφή των διανυσμάτων εξομοίωσης

Για την εξομοίωση του κυκλώματός μας, απλά βάζουμε όλες τις πιθανές τιμές στις εισόδους του :

(αρχείο 6/testmlx.v)

```
module testmlx ();
  reg a, b, c, d;
  wire [6:0] s;
  wire       dp;

  mlx i0 (a, b, c, d, s, dp);
  initial {a,b,c,d} <= 0;
```

```
always #100 {a,b,c,d} <= {a,b,c,d} + 1;
endmodule
```

Δ. UCF

Για τη σύνθεση του κυκλώματός μας, θα χρησιμοποιήσουμε το ISE της εταιρείας Xilinx (συγκεκριμένα την έκδοση WebPack, που είναι περιορισμένη ως προς τις σειρές FPGAs που υποστηρίζει, αλλά διατίθεται δωρεάν). Το εργαλείο αυτό είναι στην πράξη μια ολόκληρη πλατφόρμα εργαλείων (γραφικός editor + HDL editor + simulator + synthesis + place & route + bitgen), οπότε θα επικεντρωθούμε μόνο στα σημεία ενδιαφέροντος. Ενδεχόμενα να χρειαστεί να καταφύγετε στην online βοήθεια αρκετές φορές για να εντοπίσετε συγκεκριμένες ενέργειες του εργαλείου.

Για να έχουμε την ικανότητα υλοποίησης του κυκλώματός μας στην πλακέτα XSA θα πρέπει πέρα από τη περιγραφή του σχεδιασμού μας, να υποδείξουμε στο εργαλείο σύνθεσης, σε ποιους ακροδέκτες του FPGA θα συνδεθούν τα σήματα του σχεδιασμού μας. Αυτό είναι απαραίτητο, καθώς στην πλακέτα XSA μόνο συγκεκριμένα pins του FPGA συνδέονται με συγκεκριμένους άλλους σχεδιασμούς. Αυτή η διασύνδεση περιγράφεται αναλυτικά τόσο στο εγχειρίδιο της XSA όσο και στο αντίστοιχο Excel αρχείο.

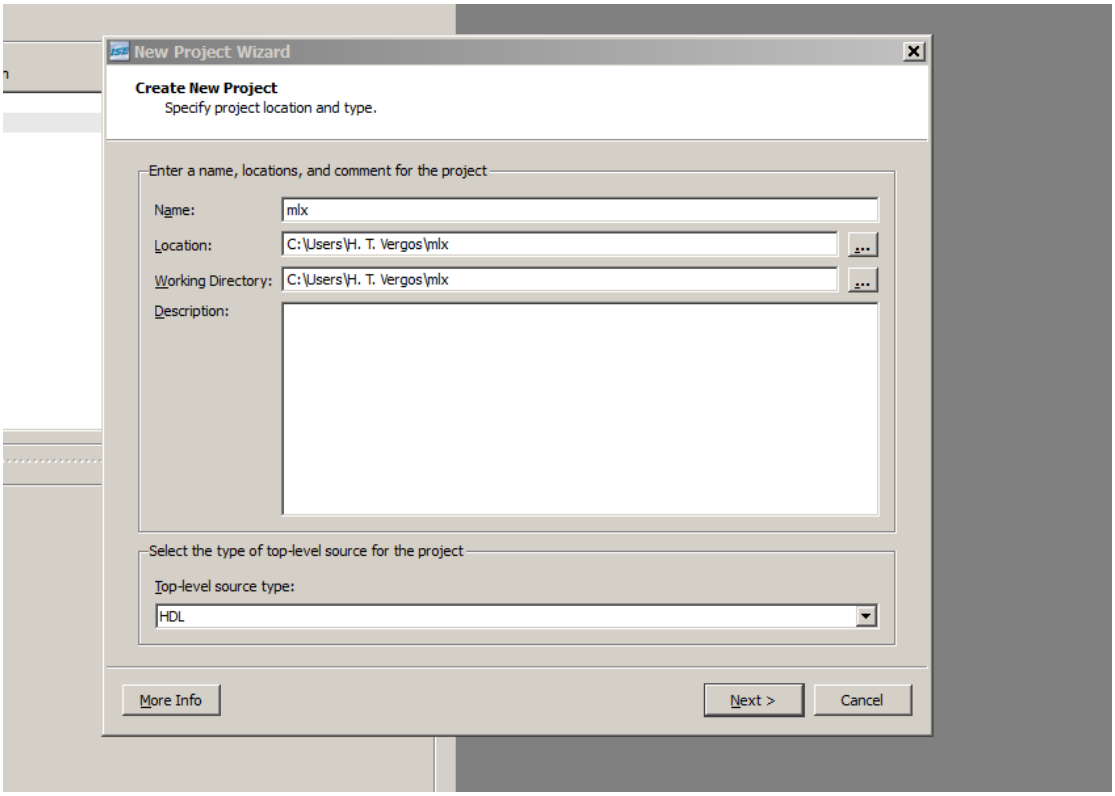
A	B	C	D	E	F	G	H	
Connections Between the FPGA, CPLD and other Components on the XSA-3								
Net Name	**FPGA Pin (U1)**	**CPLD Pin (U2)**	**Parallel Port**	**LEDs**	**Switch / Buttons**	**SDRAM (U4)**	**Flash (U3)**	**O**
FPGA-CCLK	T15	52						
FPGA-DIN-D0	M11	1		LED-C (S1)			DQ0 (29)	
FPGA-D1	N11	2		LED-DP			DQ1 (31)	
FPGA-D2	P10	4		LED-B (S4)			DQ2 (33)	
FPGA-D3	R10	5		LED-A (S6)			DQ3 (35)	
FPGA-D4	T7	7		LED-F (S5)			DQ4 (38)	
FPGA-D5	R7	9		LED-G (S3)			DQ5 (40)	
FPGA-D6	N6	10		LED-E (S2)			DQ6 (42)	
FPGA-D7	M6	11		LED-D (S0)			DQ7 (44)	
FPGA-DONE	R14	6						

Για παράδειγμα, στο απόσπασμα του Excel για την πλακέτα XSA της παραπάνω φωτογραφίας βρίσκουμε ότι το S0 led του 7 segment δυνδέεται με το ποδαράκι M6 του FPGA και συνεπώς πρέπει να ζητήσουμε από το εργαλείο της σύνθεσης να φτιάξει έτσι το κύκλωμά μας ώστε να συνδέσει το σήμα s[0] στο ποδαράκι M6. Όλοι αυτοί οι περιορισμοί που ζητάμε από το εργαλείο της σύνθεσης, αποτελούν ένα αρχείο περιορισμών (user constraints file) το οποίο θα πρέπει να επισυνάψουμε με το σχεδιασμό μας. Για το παράδειγμά μας, αυτό το αρχείο έχει ως εξής **(αρχείο 6/mlx.ucf)**

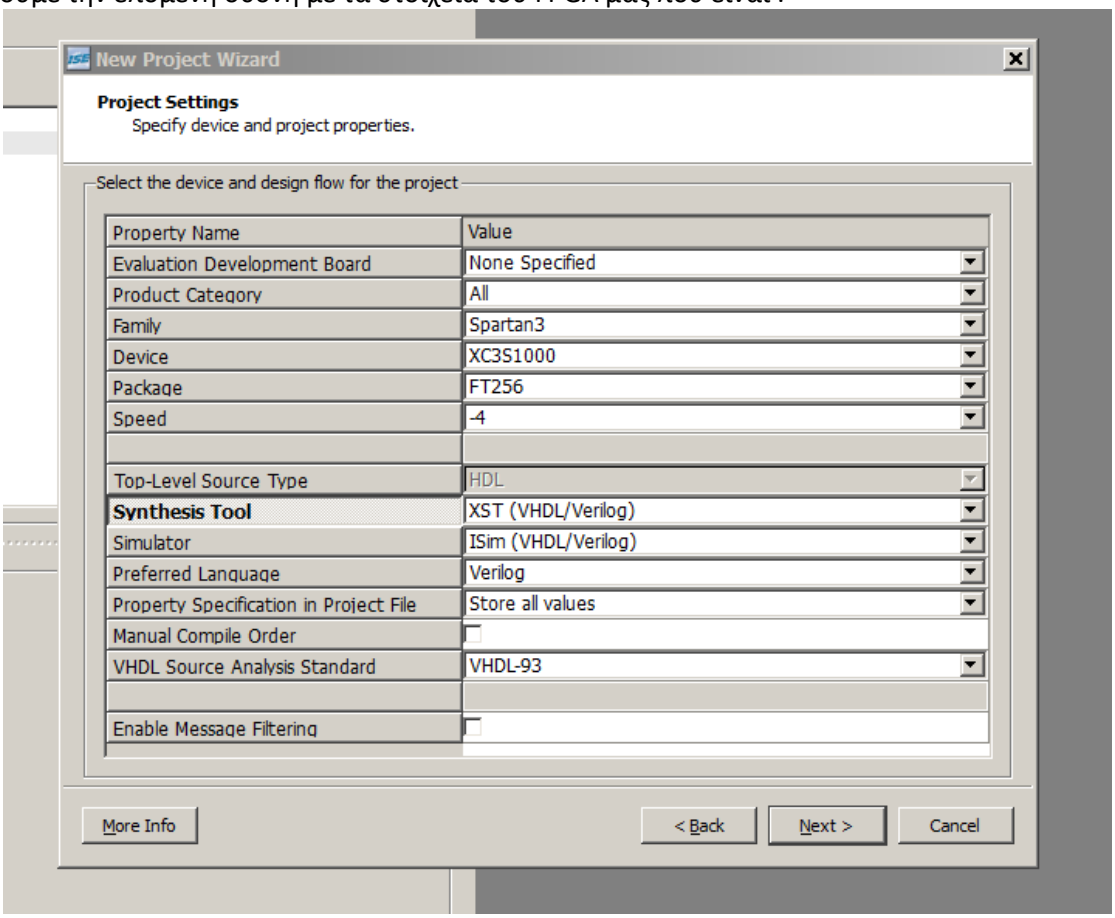
```
net a loc=k4;
net b loc=k3;
net c loc=k2;
net d loc=j4;
net s<6> loc=r10;
net s<5> loc=t7;
net s<4> loc=p10;
net s<3> loc=r7;
net s<2> loc=n6;
net s<1> loc=m11;
net s<0> loc=m6;
net dp loc=n11;
```

Ε. Σύνθεση του κυκλώματός μας και επισκόπηση των αποτελεσμάτων

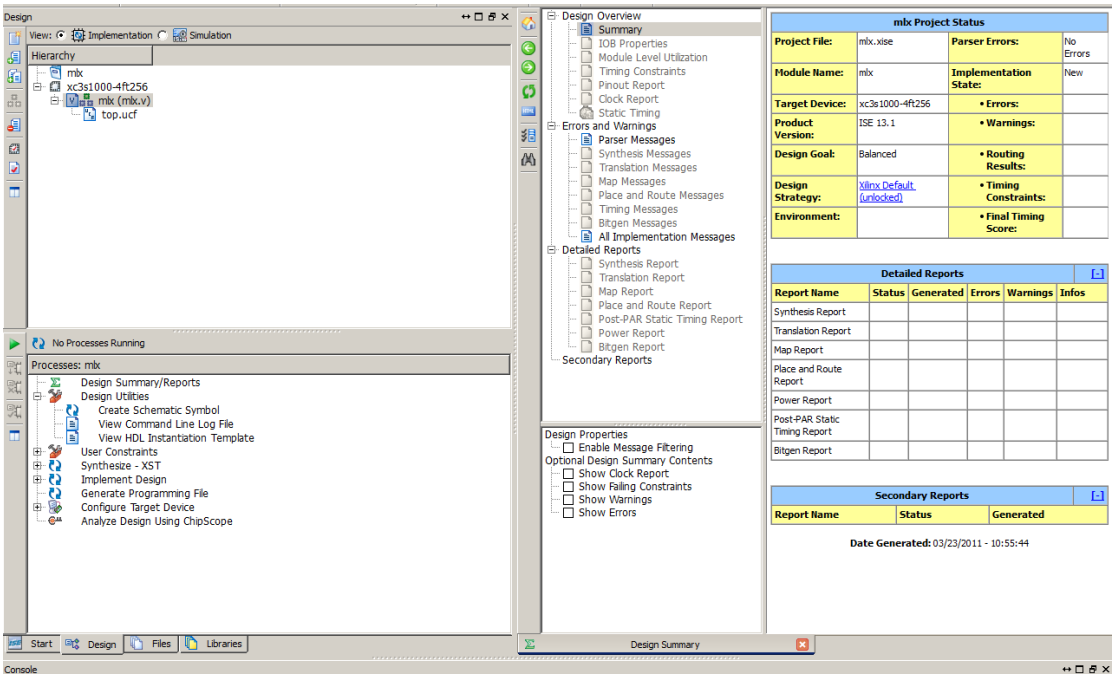
Ξεκινάμε το ISE Project Navigator που είναι ένα gui για όλο το φάσμα των εργαλείων του ISE. Επιλέγουμε New Project, δίνουμε ένα όνομα (π.χ. mlx) και έναν κατάλογο στον οποίο θα δημιουργηθεί ένα ολόκληρο folder με το παραπάνω όνομα για αυτό το project. Στο κάτω μέρος αφήνουμε τη default επιλογή ότι δηλαδή το κορυφαίο σχεδιαστικό μας κομμάτι θα είναι περιγεγραμμένο σε HDL (θυμηθείτε ότι το ISE έχει και γραφικό editor και πολλά άλλα).



Συμπληρώνουμε την επόμενη οθόνη με τα στοιχεία του FPGA μας που είναι :

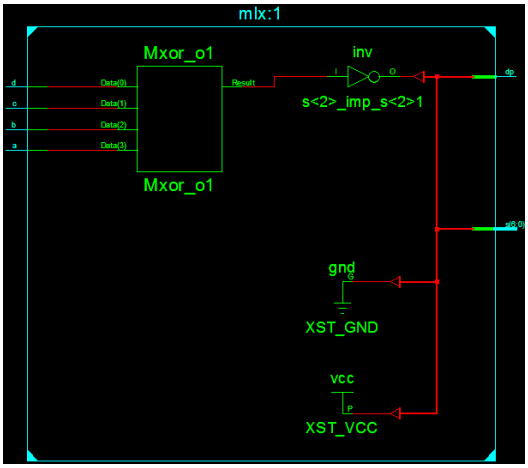


Λέμε δηλαδή στο εργαλείο ότι θα προγραμματίσουμε ένα XC3S1000 FPGA της Spartan 3 σειράς ταχύτητας 4 και σε tiny package 256 pins. Του λέμε επίσης ότι θα πρέπει να χρησιμοποιήσει το δικό του εργαλείο σύνθεσης και παρότι αφήνουμε το default εξομοιωτή, μπορούμε να του πούμε να διασυνδεθεί με την ήδη εγκατεστημένη έκδοση του ModelSim που τυχόν διαθέτουμε. Αφού αποδεχθούμε και την επόμενη διαλογική εικόνα, μεταφερόμαστε στο project navigator. Στη συνέχεια θα πρέπει να προσθέσουμε τα αρχεία μας στο project που μόλις δημιουργήσαμε μέσω του Project -> Add Source. Προσθέτουμε λοιπόν τόσο το mlx.v όσο και το mlx.ucf (όχι το testbench, αφού δεν περιγράφει τίποτε για τον σχεδιασμό μας) και έτσι καταλήγουμε στην εξής οθόνη :

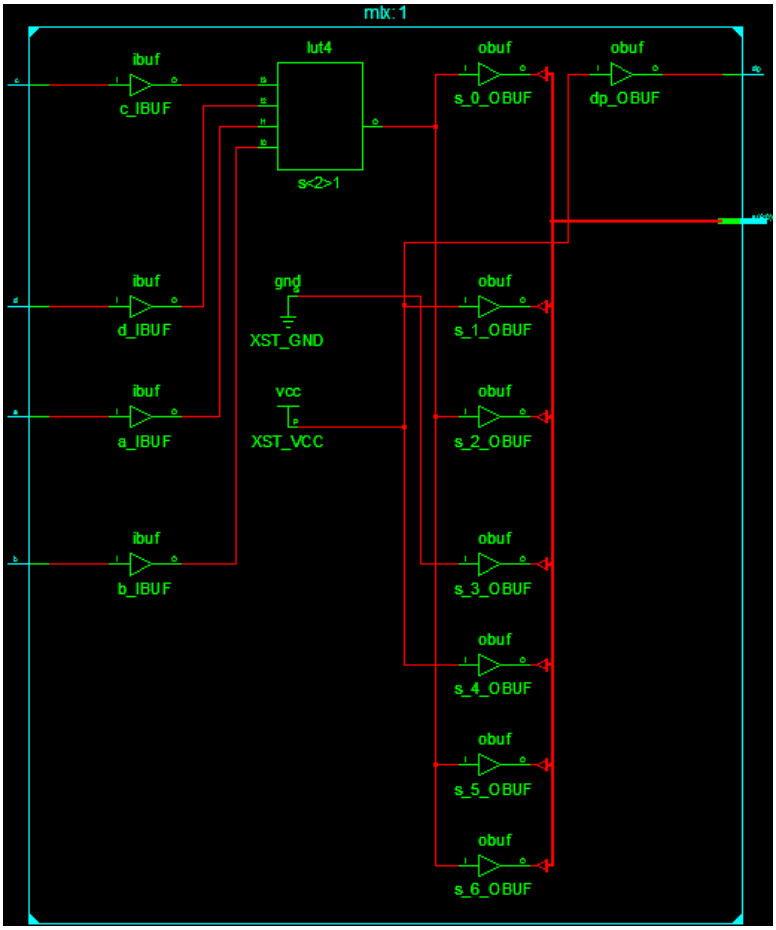


όπου πάνω αριστερά είναι η ιεραρχία του σχεδιασμού μας, κάτω αριστερά το τι μπορούμε να τρέξουμε ανάλογα με το τι θα επιλέξουμε από την ιεραρχία και δεξιά τα διάφορα reports που έχουν προκύψει.

Επιλέγουμε Process -> Implement Top module, και αυτόματα θα κληθούν τα εργαλεία μετάφρασης, σύνθεσης, απεικόνισης σε CLBs και Place & Route. Ας καλέσουμε κάποια από τα διαφορετικά εργαλεία για να δούμε τι ακριβώς μετατροπές έχουν γίνει στον σχεδιασμό μας. Επιλέγουμε Tools -> Schematic Viewer -> RTL και αφού αποδεχθούμε τις αρχικές επιλογές θα δούμε ένα σχηματικό παράθυρο του top level module. Χτυπώντας πάνω στο σύμβολο, θα δούμε την υλοποίησή του :

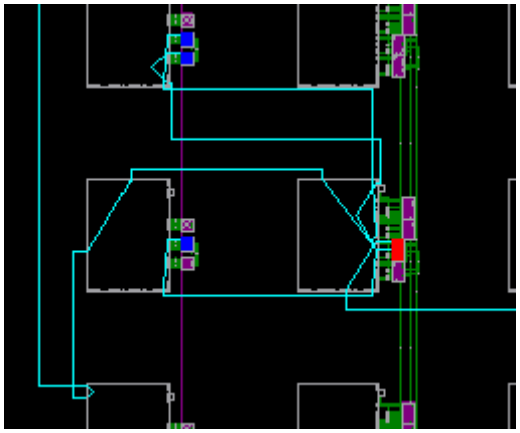


Κλείστε το παράθυρο του σχηματικού αυτού και επιλέξτε Tools -> Schematic Viewer -> Technology, αποδεχθείτε τις αρχικές επιλογές και πατήστε πάλι πάνω στο top level module του σχηματικού. Θα πάρετε έτσι το σχηματικό για τη συγκεκριμένη τεχνολογία που θα μιάζει με αυτό της εικόνας :

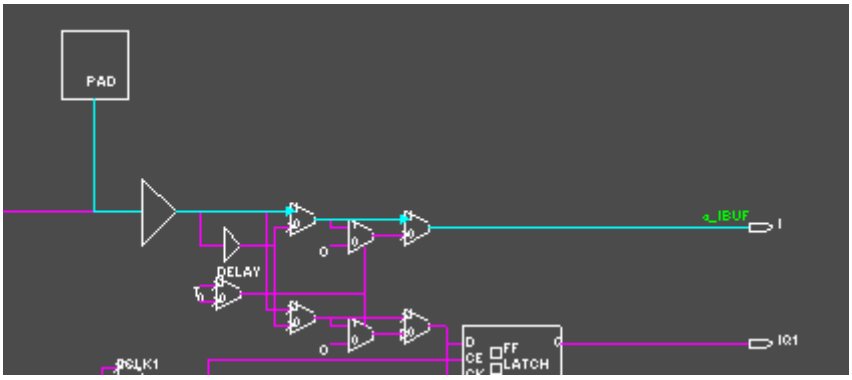


Δηλαδή, ο σχεδιασμός σας θα υλοποιηθεί με κάποιους input buffers, κάποιους output buffers και όλη σας η λογική με ένα LookUpTable 4 μεταβλητών. Ας δούμε στη συνέχεια, πως πράγματι αυτά μεταφράστηκαν στο πραγματικό κόσμο, εντός δηλαδή του FPGA που θέλαμε. Επιλέξτε Tools -> FPGA Editor -> Post Place & Route. Το εργαλείο που μας παρουσιάζεται, μας δείχνει την πραγματική υλοποίηση του κυκλώματός μας εντός του FPGA. Αφού μεγιστοποιήσετε το Array 1 υποπαράθυρο, από την πάνω πλευρά κάντε ορατά περισσότερα resources του FPGA κάνοντας τις εξής επιλογές :

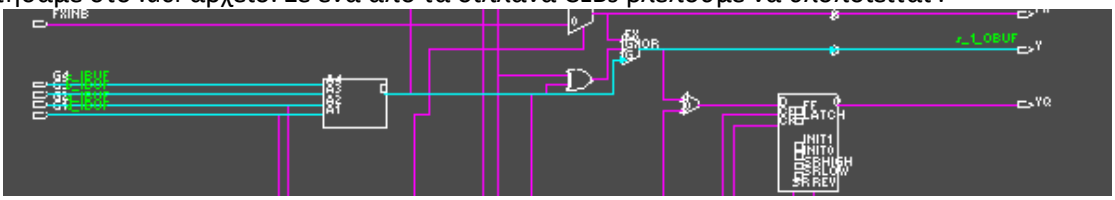
στα visibility options. Κάντε zoom in και προσπαθείστε να εντοπίσετε resources που χρησιμοποιούνται όπως αυτά της παρακάτω εικόνας στην αριστερή πλευρά του FPGA (τα πλέον αριστερά είναι IOB, ενώ το κόκκινο μέρος ενός CLB)



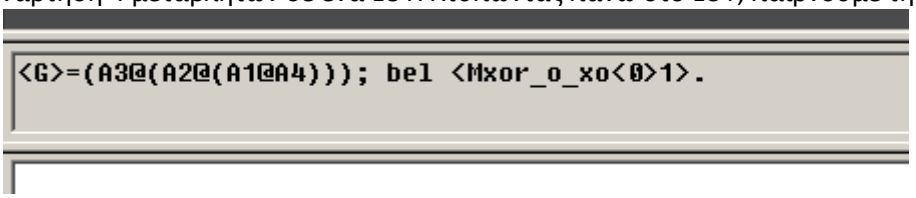
Χτυπώντας σε κάποιο από τα μπλε χρησιμοποιούμενα resources θα δείτε πως ακριβώς χρησιμοποιείται. Για παράδειγμα στην παρακάτω εικόνα :



φαίνεται ο σχεδιασμός που υλοποιείται εντός του ακροδέκτη k4. Υλοποιεί ένα pin εισόδου για το σήμα a όπως ακριβώς ζητήσαμε στο .ucf αρχείο. Σε ένα από τα διπλανά CLBs βλέπουμε να υλοποιείται :



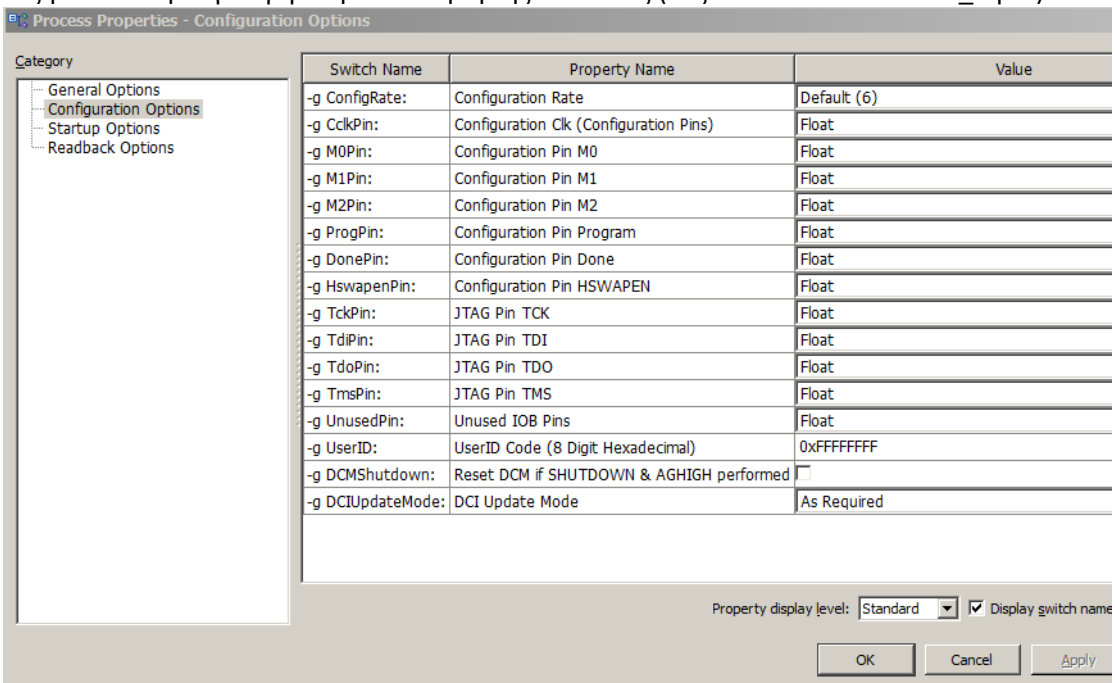
μια συνδυαστική συνάρτηση 4 μεταβλητών σε ένα LUT. Χτυπώντας πάνω στο LUT, παίρνουμε τη συνάρτησή του :



Αφού κλείσετε όλα τα εργαλεία τα οποία είδαμε ως τώρα, ας πάμε να προσδιορίσουμε τη συχνότητα λειτουργίας του σχεδιασμού μας. Θα χρησιμοποιήσουμε το Tools -> Timing Analyzer σε Post Place & Route mode. Στο παράθυρο που ανοίγει επεκτείνετε το Data Sheet report και επιλέξτε Pad to Pad. Εμφανίζονται τα χειρότερα μονοπάτια καθυστέρησης από κάθε είσοδο σε κάθε έξοδο (στο σχεδιασμό μας μόνο τα s[1] και s[4] αλλάζουν τιμές). Από εκεί προκύπτει ότι η χειρότερη καθυστέρηση του σχεδιασμού μας είναι 13.070ns.

ΣΤ. Υλοποίηση του κυκλώματός μας

Αφού ο σχεδιασμός μας έχει υλοποιηθεί, το επόμενο βήμα είναι η δημιουργία του αρχείου προγραμματισμού. Αφού κλείσετε όσα εργαλεία είναι ακόμη ανοιχτά, επιλέξτε στην ιεραρχία το mlx.v και στο κάτω αριστερά παράθυρο, κάντε δεξί κλικ στο Generate Programming File. Στο νέο παράθυρο που ανοίγει επιλέξτε Configuration Options και επιλέξτε να απομακρυνθούν όλες οι pull up αντιστάσεις που βάζει το εργαλείο στα configuration pins μιας και αυτό ζητείται στη συγκεκριμένη υλοποίηση της πλακέτας (δες το xsa-3S-manual-v1_1.pdf).



Αφού λοιπόν επιλέξουμε τις επιλογές που φαίνονται στην εικόνα, τις αποδεχόμαστε και κάνουμε διπλό κλικ στο Generate Programming file, οπότε και θα δημιουργηθεί ένα αρχείο mlx.bit στον κατάλογο του σχεδιασμού μας, που είναι το αρχείο προγραμματισμού. Ήρθε η ώρα να προγραμματίσουμε το FPGA μας.

Για την επικοινωνία της πλακέτας μας με το FPGA, θα χρησιμοποιήσουμε τα εργαλεία της XESS. Παρότι υπάρχουν αρκετά εμείς θα χρησιμοποιήσουμε μόνο το GXSLOAD. Δώστε λοιπόν τροφοδοσία στις πλακέτες σας και τρέξτε το GXSLOAD. Επιλέξτε Board Type XSA3S1000, Port επικοινωνίας το LPT1 και σύρτε το αρχείο mlx.bit εντός του χώρου FPGA/CPLD, για να δείξετε που θέλετε να αποθηκευτεί το αρχείο σας (με το ίδιο εργαλείο μπορούμε να προγραμματίσουμε και τις μνήμες της XSA). Πατείστε το Load και όταν ολοκληρωθεί η διαδικασία, έχετε πλέον ένα λειτουργικό πρωτότυπο !!! Χαρείτε το !!! Αλλάξτε κάποιο από τα dip switches (με τη βοήθεια της μύτης ενός μολυβιού) και δείτε πως αλλάζει και η ένδειξη στα leds. (Υπόδειξη: το λογικό 1 στα dip switches είναι μάλλον ανάποδα από ότι θα περιμένατε).

Ζ. Ζητούμενα

(1) Περιγράψτε ένα κύκλωμα το οποίο θα διαβάζει έναν δυαδικό αριθμό από τα dip switches και θα τον απεικονίζει στα led bars. Παραδοτέα : ο κώδικας περιγραφής, η μέγιστη συχνότητα λειτουργίας του κυκλώματός σας και το αρχείο προγραμματισμού.

(2) Περιγράψτε ένα κύκλωμα το οποίο θα πολλαπλασιάζει 2 αριθμούς, ο καθένας των 2 δυαδικών ψηφίων και θα απεικονίζει το αποτέλεσμά τους στο 7 segment. Εξετάστε 2 πιθανότητες : (α) οι αριθμοί να είναι μη προσημασμένοι και (β) οι αριθμοί να είναι σε παράσταση συμπληρώματος του 2. Χρησιμοποιείστε το DP του 7 segment για να υποδείξετε το πρόσημο του αποτελέσματος. Παραδοτέα για κάθε υποπερίπτωση : ο κώδικας περιγραφής, η μέγιστη συχνότητα λειτουργίας και το αρχείο προγραμματισμού.