ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 7

Σκοπός της εργαστηριακής άσκησης είναι να φτιάξουμε ένα ακολουθιακό κύκλωμα χρησιμοποιώντας και τις 2 αναπτυξιακές μας πλακέτες ελαφρά μεγαλύτερης δυσκολίας από αυτό της προηγούμενης άσκησης. Αντικείμενο είναι αρχικά να φτιάξουμε έναν μετρητή δευτερολέπτων ο οποίος θα απεικονίζει τη μέτρησή του στα δύο 7 segment του XST board και κατόπιν να τον τροποποιήσετε ώστε να παρέχει επιπλέον δυνατότητες.

Α. Συχνότητα 1Hz

Η πλακέτα XSA έχει ενσωματωμένο έναν χρονιστή συχνότητας 100MHz. Συνεπώς για να πάρουμε την επιθυμητή συχνότητα του 1Hz θα πρέπει να διαιρέσουμε τη βασική με 10^8. Προφανώς αυτό μπορούμε να το πετύχουμε ανιχνεύοντας πότε ένας μετρητής με $\lceil \log_2 10^8 \rceil = 27$ δυαδικά ψηφία φθάνει σε μία συγκεκριμένη τιμή μέτρησης, αλλά κάτι τέτοιο θα απαιτούσε τη σύγκριση 27 ψηφίων, πράγμα που ενδεχόμενα να μας έριχνε σημαντικά τη μέγιστη συχνότητα λειτουργίας του σχεδιασμού μας και συνεπώς θα χάλαγε την ακρίβεια του ρολογιού μας. Συνεπώς θα φτιάξουμε μικρότερους μετρητές που η εξάντληση της ακολουθίας μέτρησης του ενός θα είναι η επίτρεψη μέτρησης του άλλου. Επειδή είναι $10^8 = 25^4 * 2^8$ θα φτιάξουμε ένα μετρητή mod25 και έναν 8 bit και θα χρησιμοποιήσουμε 4 από το πρώτο είδος και 1 από το δεύτερο σε μια αλυσίδα επίτρεψης.

Εδώ υπάρχει ένα κρυφό και ταυτόχρονα καίριο σημείο. Γιατί να φτιάξουμε μετρητές με είσοδο επίτρεψης και όχι απλούς μετρητές παίρνοντας τη πιο σημαντική έξοδο του ενός ως ρολόι του επόμενου ? Η απάντηση σε αυτό το ερώτημα έχει να κάνει με τους φυσικούς πόρους που διαθέτει το FPGA. Όπως εξηγήθηκε στο μάθημα, το FPGA διαθέτει ειδικούς buffers για σήματα ρολογιού, οι οποίοι όμως είναι περιορισμένοι. Αν συνεπώς χρησιμοποιήσουμε μόνο ένα ρολόι, θα χρησιμοποιηθεί γι' αυτό ο BUFGP (global buffer) με αποτέλεσμα να έχουμε το μικρότερο δυνατό skew ανάμεσα στο ρολόι που λαμβάνουν όλα τα flip flops του σχεδιασμού μας. Έτσι λοιπόν, ο κώδικας για τη συχνότητα 1 Hz είναι ο ακόλουθος :

(αρχείο 7/cnt25.v)

```
module cnt25 (reset, clk, enable, clkdiv25);
input reset, clk, enable;
output clkdiv25;
reg [5:0] cnt;

assign clkdiv25 = (cnt==5'd24);
always @(posedge reset or posedge clk)
  if (reset) cnt <= 0;
   else if (enable)
        if (clkdiv25) cnt <= 0;
          else cnt <= cnt + 1;
endmodule

module cnt8b (reset, clk, enable, clkdiv256);
input reset, clk, enable;
output clkdiv256;
reg [7:0] cnt;

assign clkdiv256 = (cnt==8'd255);
always @(posedge reset or posedge clk)
  if (reset) cnt <= 0;
   else if (enable) cnt <= cnt + 1;
endmodule


module OneHertz(reset, clk, en_nxt);
input clk, reset;
output en_nxt;
wire clk1Hz;
wire first, second, third, fourth;

cnt25 i0 (reset, clk, 1'b1, first);
cnt25 i1 (reset, clk, first, second);
```

```
cnt25 i2 (reset, clk, first & second, third);
cnt25 i3 (reset, clk, first & second & third, fourth);
cnt8b i4 (reset, clk, first & second & third & fourth, clk1Hz);
assign en_nxt = first & second & third & fourth & clk1Hz;
endmodule
```

Μπορούμε να επιβεβαιώσουμε τη λειτουργία του κυκλώματος παραγωγής της συχνότητας 1Hz, εξομοιώνοντάς το με τον ακόλουθο κώδικα :

```
module testOneHertz();
 reg reset, clk;
 wire en_nxt;

 OneHertz CUT (reset, clk, en_nxt);
 initial begin reset=0; clk = 0; # 10 reset = 1; # 9 reset = 0; end
 always #1 clk=~clk;
endmodule
```

Β. Μετρητής δευτερολέπτων

Για να μετρήσουμε τα δευτερόλεπτα, θα μπορούσαμε απλά να χρησιμοποιήσουμε έναν δυαδικό μετρητή 6 δυαδικών ψηφίων με αναδίπλωση μετά την τιμή 59. Ωστόσο για την απεικόνιση αυτής της λύσης στα δύο 7-segments θα έπρεπε μετά να υπολογίζουμε το πηλίκο και το υπόλοιπο της μέτρησης ως προς 10. Μια εναλλακτική πρόταση είναι να χρησιμοποιήσουμε 2 εναλλακτικούς μετρητές έναν για τις μονάδες των δευτερολέπτων και έναν για τις δεκάδες. Ο πρώτος είναι ένας δεκαδικός μετρητής, ενώ ο δεύτερος ένας μετρητής modulo 6. Ο κώδικας γι' αυτή τη δεύτερη λύση, είναι ο ακόλουθος (προσέξτε ότι εξακολουθούμε να χρησιμοποιούμε παντού τη λογική του ενός ρολογιού) :

(αρχείο 7/secondcounter.v)

```
module singleseconds (reset, clk, enable, ss, nxt);
 input reset, clk, enable;
 output [3:0] ss;
 output      nxt;
 reg    [3:0] ss;

 assign nxt= (ss == 4'd9);
 always @(posedge clk or posedge reset)
  if (reset) ss <= 4'd0;
    else if (enable)
         if (nxt) ss <= 0;
          else ss <= ss + 1;
endmodule

module tenthsofseconds (reset, clk, enable, ts);
 input reset, clk, enable;
 output [2:0] ts;
 wire        again;
 reg    [2:0] ts;

 assign again = (ts == 3'd5);
 always @(posedge clk or posedge reset)
  if (reset) ts <= 4'd0;
    else if (enable)
         if (again) ts <= 0;
          else ts <= ts + 1;
endmodule

module secondcounter (reset, clk, enable, ts, ss);
 input reset, clk, enable;
 output [2:0] ts;
```

```
  output [3:0] ss;
  wire       ent;

  singleseconds   i0 (reset, clk, enable, ss, ent);
  tenthsofseconds i1 (reset, clk, enable & ent, ts);
endmodule
```

του οποίου μπορείτε να επαληθεύσετε την ορθή λειτουργία με τον ακόλουθο κώδικα εξομοίωσης :

```
module test();
  reg reset, clk;
  wire [2:0] ts;
  wire [3:0] ss;

  secondcounter CUT (reset, clk, 1'b1, ts, ss);

  initial begin reset =0; clk = 0; #10 reset = 1; #9 reset = 0; end
  always #4 clk = ~clk;
endmodule
```

Γ. Απεικόνιση στα Leds

Χρειαζόμαστε τέλος ένα συνδυαστικό κύκλωμα που θα παίρνει στην είσοδο τη δυαδική μέτρηση και θα βγάζει στην έξοδο εξόδους για τα 7 segments. Ένα τέτοιο κύκλωμα είναι το εξής :

(αρχείο 7/bin_2_7.v)

```
module bin_2_7 (x, s);
  input  [3:0] x;
  output [6:0] s;

  assign s = (x == 4'd0 ) ? 7'b1111110 :  (x == 4'd1 ) ? 7'b0110000 :
         (x == 4'd2 ) ? 7'b1101101 : (x == 4'd3 ) ? 7'b1111001 :
         (x == 4'd4 ) ? 7'b0110011 : (x == 4'd5 ) ? 7'b1011011 :
         (x == 4'd6 ) ? 7'b1011111 : (x == 4'd7 ) ? 7'b1110010 :
         (x == 4'd8 ) ? 7'b1111111 : (x == 4'd9 ) ? 7'b1111011 :
         (x == 4'd10) ? 7'b1110111 : (x == 4'd11) ? 7'b0011111 :
         (x == 4'd12) ? 7'b1001110 : (x == 4'd13) ? 7'b0111101 :
         (x == 4'd14) ? 7'b1001111 : 7'b1000111 ;
endmodule
```

Δ. Top Level Module

Το τελευταίο module που χρειαζόμαστε είναι αυτό που ενώνει όλους τους υπόλοιπους σχεδιασμούς και είναι το κορυφαίο στην ιεραρχία μας. Αυτό θα περιέχει τους διαιρέτες συχνότητας, τον μετρητή δευτερολέπτων και 2 αντίγραφα του κυκλώματος απεικόνισης. Το κύκλωμα αυτό λαμβάνει μόνο reset και ρολόι και βγάζει τις εξόδους των δύο 7 segments. Ο κώδικας συνεπώς που χρειάζεται **(αρχείο 7/tops.v)** είναι :

```
module tops (reset, clk, left, right);
  input reset, clk;
  output [6:0] left, right;
  wire [2:0] ts;
  wire [3:0] ss;

  OneHertz      i0 (reset, clk, en_nxt);
  secondcounter i1 (reset, clk, en_nxt, ts, ss);
  bin_2_7       lt ({1'b0,ts}, left);
  bin_2_7       rt (ss, right);
endmodule
```

και όλο το κύκλωμά μας μπορεί να ελεγχθεί με το ακόλουθο testbench :

```
module test();
  reg reset, clk;
```

```
wire [6:0] left, right;

tops CUT (reset, clk, left, right);

initial
  begin
     reset = 0; clk =0;
   #2 reset = 1;
   #2 reset = 0;
  end

 always #5 clk = ~clk;
endmodule
```

Πλέον έχουμε ολοκληρώσει το σχεδιασμό μας και μπορούμε να προχωρήσουμε στην υλοποίησή του.

Ε. Αρχείο φυσικών παραμέτρων

Στη συνέχεια μπορούμε να προχωρήσουμε στον ορισμό των pins που θα μας χρειαστούν. Προσοχή χρειάζεται στο ότι πρέπει να χρησιμοποιήσουμε το Excel αρχείο του XST+XSA. Για το reset θα επιλέξουμε το 1ο dip switch της XSA πλακέτας και για ρολόι το pin t9 που οδηγείται από το CPLD (CLKA). Έτσι το αρχείο μας **(αρχείο 7/tops.ucf)** διαμορφώνεται στο εξής :

```
net clk loc=t9;
net reset loc=k4;
net left<6> loc=h14;
net left<5> loc=m4;
net left<4> loc=p1;
net left<3> loc=n3;
net left<2> loc=m15;
net left<1> loc=h13;
net left<0> loc=g16;
net right<6> loc=e2;
net right<5> loc=e1;
net right<4> loc=f3;
net right<3> loc=f2;
net right<2> loc=g4;
net right<1> loc=g3;
net right<0> loc=g1;
```

Ζ. Ζητούμενα

(1) Εφαρμόστε όλη τη ροή σύνθεσης και υλοποίησης του κυκλώματος όπως αυτή αναπτύχθηκε στην εργαστηριακή άσκηση 7 ώστε να καταλήξετε σε ένα λειτουργούν πρωτότυπο. Φωνάξτε τον επιβλέποντα, για να του το δείξετε.

(2) Επαυξήστε τις δυνατότητες του σχεδιασμού σας με τα ακόλουθα :

- Λειτουργία stop-watch : Η αλλαγή κάποιου διακόπτη παγώνει την ένδειξη δευτερολέπτων, αλλά το ρολόι σας συνεχίζει να μετράει κανονικά. Με την επαναφορά του διακόπτη, η ένδειξη συνεχίζει κανονικά από την πραγματική τιμή των δευτερολέπτων.
- Μέτρηση δεκάτων : Πέρα από δευτερόλεπτα το ρολόι σας είναι ικανό να μετρά και δέκατα του δευτερολέπτου. Η απεικόνιση γίνεται στα led bars δίπλα από τα 7 segment, όπου ακολουθείται η κωδικοποίηση θερμομέτρου, δηλαδή ο αριθμός των αναμένων leds αυξάνει προς κάποια διεύθυνση για κάθε δέκατο του δευτερολέπτου που περνά.