HY220: Εργαστήριο Ψηφιακών Κυκλωμάτων

Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης
Χειμερινό Εξάμηνο 2020

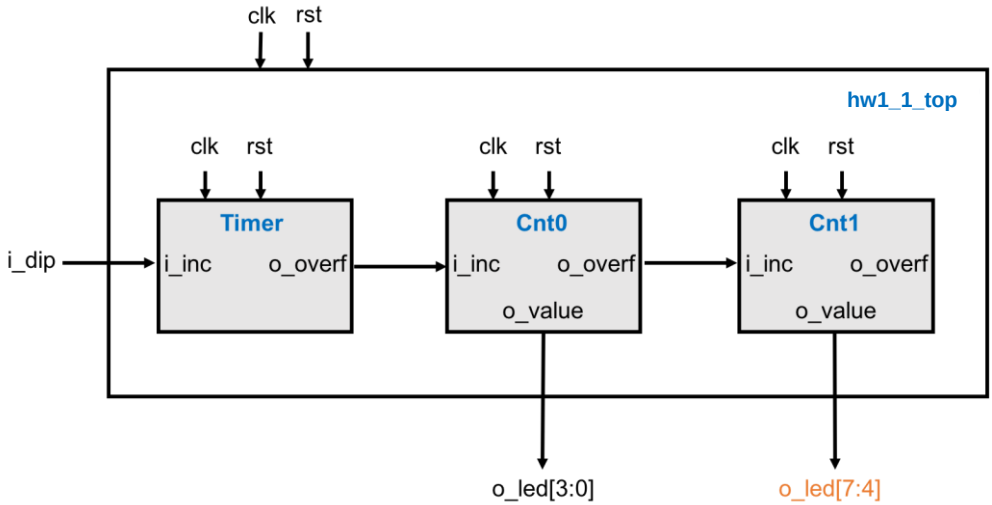
1η Σειρά Ασκήσεων

Ημερομηνία Παράδοσης: Παρασκευή 04/12/2020 23:59

Άσκηση 1.1

Ο σκοπός της *Άσκησης 1.1* είναι να εξοικειωθείτε με προσομοίωση Verilog/SystemVerilog και το περιβάλλον Xilinx Vivado.

Σας δίνεται έτοιμος κώδικας SystemVerilog και testbench που εμφανίζει δύο μετρητές 4-bit στην έξοδο o_led (στην πλακέτα του εργαστηρίου θα ήταν 8 λαμπάκια led). Θα πρέπει να ακολουθήσετε την ροή του εργαλείου Xilinx Vivado και τα βήματα που χρειάζεται για να κάνετε προσομοίωση και να δείτε το να δουλεύει. Ο κώδικας SystemVerilog που σας δίνεται υλοποιεί το παρακάτω σχέδιο:



Το σχέδιο έχει την εξής ιεραρχία:

- **hw1_1_top** *(hw1_1_top.sv)*: περιέχει 3 instances (Timer, Cnt0, Cnt1) του μπλοκ **counter**, τα συνδέει μεταξύ τους και συνδέει τις εισόδους και τις εξόδους όπως φαίνεται στο σχήμα.
- **counter** *(counter.sv)*: είναι ένα απλό μπλοκ που υλοποιεί ένα παραμετρικό μετρητή N bits τον οποίο εμφανίζει στην έξοδο *o_value*. Το μπλοκ έχει την είσοδο *i_increase* και σε κάθε κύκλο που είναι ενεργοποιημένη η είσοδος αυξάνει την τιμή του μετρητή κατά ένα. Όταν ο μετρητής κάνει wrap-round τότε γεννάει στην έξοδο *o_overflow* ένα παλμό ενός κύκλου ρολογιού. Το μπλοκ έχει επίσης την παράμετρο MAX που καθορίζει (at compile time) σε ποια τιμή θα κάνει wrap-around ο μετρητής.
- **hw1_1_tb** *(hw1_1_tb.sv)*: ένα απλό testbench *για προσομοίωση* που δημιουργεί το ρολόι και το reset.

Στο σχέδιο ο μετρητής Timer παίρνει την είσοδο *i_increase* από το σήμα *i_dip* (στην πλακέτα του εργαστηρίου θα είχε συνδεθεί σε ένα διακόπτη - DIP switch πάνω στην πλακέτα). Όταν ο

διακόπτης είναι ενεργοποιημένος τότε θα αυξάνεται η εσωτερική τιμή του Timer. Το ρολόι του κυκλώματος είναι 100MHz (περίοδος ρολογιού 10ns) οπότε στον κώδικα ο Timer έχει τις κατάλληλες παραμέτρους ώστε να κάνει wrap-around μία φορά ανά 10 κύκλους και άρα γεννάει το σήμα *o_overflow* μία φορά ανά 100ns. Με τη σύνδεση που σας δίνεται, ο μετρητής Cnt0 (4-bits) αυξάνει την τιμή του μία φορά ανά 10 κύκλους (100 ns) και όταν αυτός με τη σειρά του κάνει overflow (κάθε 160 κύκλους - 1600ns) τότε θα αυξάνει ο μετρητής Cnt1. Η έξοδος *o_value* του μετρητή Cnt0 είναι συνδεδεμένη σε 4 εξωτερικά σήματα (στην πλακέτα θα ήταν 4 λαμπάκια led έτσι ώστε να βλέπετε την τιμή του - στο δυαδικό).

Τι πρέπει να κάνετε για την Άσκηση 1.1:

1. Χρησιμοποιείστε το testbench που σας δίνεται για να τρέξετε προσομοίωση με τον simulator του Vivado. Βάλτε στις κυματομορφές (waveforms) τα όλα σήματα εισόδου και εξόδου από τα 3 counter instances (Timer, Cnt0, Cnt1) και παρατηρείστε τα σήματα *o_overflow*. Για την προσομοίωση η MAX τιμή του Timer είναι ορισμένη στο 10 έτσι ώστε η προσομοίωση να τελειώνει γρηγορότερα. Σημαδέψτε με marker τη στιγμή που ο Cnt1 βγάζει στην έξοδό του *o_value* την τιμή 5. **Κρατήστε «στιγμιότυπο οθόνης» (screenshot) για να το παραδώσετε (hw1_1_waves1 σε μορφή png/jpg/pdf)** .

2. Συνδέστε την έξοδο του μετρητή Cnt1 στα υπόλοιπα λαμπάκια (o_led[7:4]) έτσι ώστε να βλέπετε και τους 2 μετρητές στην έξοδο. Αλλάξτε την παράμετρο MAX των Cnt0 και Cnt1 έτσι ώστε ο μετρητές να κάνουν wrap-around μετά από 4 αυξήσεις. Ακολουθείστε πάλι τα βήματα της προσομοίωσης και **κρατήστε «στιγμιότυπο οθόνης» (screenshot) για να το παραδώσετε (hw1_1_waves2 σε μορφή png/jpg/pdf)** όπου θα φαίνονται όλες οι πόρτες (είσοδοι και έξοδοι) του block **hw1_1_top** για τους πρώτους 200 κύκλους ρολογιού της προσομοίωσης μετά την ολοκλήρωση του reset.

Άσκηση 1.2

Για την *Άσκηση 1.2* θα πρέπει να σχεδιάσετε σε SystemVerilog έναν παραμετρικό χρονομετρητή (hw1_2_timer.sv) που μετράει προς τα κάτω με τις παρακάτω προδιαγραφές:

- Λειτουργία με ρολόι 25MHz (clk25) και ασύγχρονο active-low reset (arst_n)
- Πόρτα εισόδου N-bits για την φόρτωση μέγιστης τιμή του μετρητή (i_value)
- Πόρτα εισόδου 1-bit για την σηματοδότηση φόρτωσης (i_load)
- Πόρτα εξόδου 1-bit για να σηματοδοτήσει την ολοκλήρωση μέτρησης (o_done)

Ο χρονομετρητής αυτός μετράει προς τα κάτω όταν η τιμή του δεν είναι μηδέν. Κάθε φορά που έρχεται το σήμα i_load τότε φορτώνει εσωτερικά την τιμή i_value. Όσο ο μετρητής είναι μηδέν το σήμα o_done είναι ενεργοποιημένο (λογική τιμή 1). Η έξοδος o_done θα πρέπει να βγαίνει Κ κύκλους (το Κ είναι σταθερά) μετά από την στιγμή που ο μετρητής γίνεται μηδέν. Σαν Κ βάλτε το πηλίκο της ακέραιας διαίρεσης (floor) του τελευταίου ψηφίου του ΑΜ με το 2 (πχ αν το ΑΜ σας είναι 103**7** τότε **7**/2=3.5 άρα Κ=3).

Παραδώστε τον κώδικα του hw1_2_timer.sv μαζί με ένα κατάλληλο testbench (hw1_2_tb.sv) που να δοκιμάζετε τουλάχιστον 3 διαφορετικά σενάρια φόρτωσης σε διαφορετικές χρονικές στιγμές. **Επίσης σχεδιάστε και παραδώστε ένα σχήμα με το κύκλωμα του hw1_2_timer (hw1_2_timer_circuit σε μορφή png/jpg/pdf)** που υλοποιήσατε δείχνοντας τα βασικά ψηφιακά στοιχεία όπως πύλες, καταχωρητές, πολυπλέκτες κτλ. (όπως δείχνουμε στις διαλέξεις).

Τι πρέπει να παραδώσετε:

Μέχρι τη ημερομηνία της προθεσμίας πρέπει να παραδώστε σε ένα zip/tar.gz αρχείο (e.g. hw1.zip ή hw1.tar.gz) τα παρακάτω:

1. Το στιγμιότυπο **hw1_1_waves1**
2. Το στιγμιότυπο **hw1_1_waves2**
3. Τον SystemVerilog κώδικα **hw1_2_timer.sv**
4. Το SystemVerilog testbench **hw1_2_tb.sv**
5. Το σχήμα του **hw1_2_timer_circuit**

Στείλτε το **hw1.zip** σας με e-mail στο hy220@csd.uoc.gr με τίτλο: *"HW1 – Ονοματεπώνυμο – ΑΜ"*.

Οι κώδικες θα ελέγχονται για αντιγραφές με ειδικό λογισμικό!