ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΗΜΜΥ

ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ & ΥΛΙΚΟΥ

ΕΡΓΑΣΤΗΡΙΑΚΕΣ ΑΣΚΗΣΕΙΣ ΓΙΑ ΤΟ ΜΑΘΗΜΑ:

ΗΡΥ 203 - ΠΡΟΧΩΡΗΜΕΝΗ ΛΟΓΙΚΗ ΣΧΕΔΙΑΣΗ

ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ 2020

Εργαστήριο 2

ΕΞΟΙΚΕΙΩΣΗ ΜΕ ΤΗ ΓΛΩΣΣΑ ΠΕΡΙΓΡΑΦΗΣ ΥΛΙΚΟΥ VHDL ΚΑΙ ΤΗΝ ΙΕΡΑΡΧΙΚΗ ΣΧΕΔΙΑΣΗ

ΕΚΠΟΝΗΣΗ: Καθ. Α. Δόλλας, Δρ. Κ. Παπαδημητρίου, Δρ. Ε. Σωτηριάδης, Μ. Κιμιωνής

ΣΥΝΕΡΓΑΤΕΣ: Δρ. Ε.Σωτηριάδης, Μ.Κιμιωνής, Ι.Πολογιώργη

ΕΚΔΟΣΗ : 11

Χανιά

Σκοπός

Είναι η περαιτέρω εξοικείωση με τη γλώσσα VHDLκαι την ιεραρχική σχεδίαση με πολλαπλά αρχεία. Θα σχεδιάσετε και θα υλοποιήσετε έναν fulladder 4-bit τύπου CarryLookAhead (CLA), με ιεραρχική σχεδίαση.

Προετοιμασία

Πριν την προσέλευση σας στο εργαστήριο πρέπει:
1)να έχετε υλοποιήσει σε γλώσσαVHDL το παρακάτω κύκλωμα, περιλαμβανομένου και του testbench, και
2) να έχετε επαληθεύσει επαρκώς την λειτουργία του κυκλώματος καλύπτοντας τις χαρακτηριστικές περιπτώσεις. Επίσης θα δείξετε συνοπτικά τη διαδικασία σχεδίασης του κυκλώματος.

ΠΡΟΣΟΧΗ: Ήδη από αυτό το απλό κύκλωμα, πλήρης επαλήθευση θα σήμαινε 512 εισόδους (4 για A, 4 για B και 1 για Cin). Αυτό πρακτικά είναι ανέφικτο για νέους σχεδιαστές κυκλωμάτων, στον χρόνο που θέλουμε να διαθέσετε για το εργαστήριο. Επομένως, επαληθεύσετε πλήρως τα Modules για G και P, τα Modules για S, και επαρκώς τα υπόλοιπα, προτιμώντας τις χαρακτηριστικές περιπτώσεις, όπως δηλ. το A να είναι όλο «1», το B όλο «0» και το Cin=1, κλπ.

Εξισώσεις που θα υλοποιηθούν (οι εξισώσεις του CLA)

$$P_i = A_i \oplus B_i$$
$$G_i = A_i \cdot B_i$$

$$C_0 = G_0 + P_0 \cdot C_{in}$$
$$C_1 = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_{in}$$
$$C_2 = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_{in}$$
$$C_3 = G_3 + P_3 \cdot G_2 + P_3 \cdot P_2 \cdot G_1 + P_3 \cdot P_2 \cdot P_1 \cdot G_0 + P_3 \cdot P_2 \cdot P_1 \cdot P_0 \cdot C_{in}$$

$$S_0 = A_0 \oplus B_0 \oplus C_{in}$$
$$S_i = A_i \oplus B_i \oplus C_{i-1}, \text{for } i > 0$$

Ζητούμενα

Να σχεδιάσετε και να υλοποιήσετε ένα κύκλωμα που έχει εισόδους και εξόδους όπως στον Πίνακα 1.

Όνομα	in/out	Πλάτος σε bit
A	είσοδος	4
B	είσοδος	4
Cin	είσοδος	1
S	έξοδος	4
C_3	έξοδος	1

Πίνακας 1: Είσοδοι – έξοδοι κυκλώματος

Το κύκλωμα λειτουργεί ως εξής:

Τα A,B είναι οι αριθμητικές είσοδοι του κυκλώματος και το Cin είναι το κρατούμενο εισόδου. Ο αθροιστής είναι τύπου 4-bitCarryLookAhead (CLA). Το αποτέλεσμα της πράξης φαίνεται στο S και το κρατούμενο εξόδου στο C_3.

Παρατηρήσεις/Σημειώσεις

Για την ιεραρχική σχεδίαση σε VHDL,χρειάζεται να κάνουμε ανάλυση του κυκλώματος από πάνω προς τα κάτω(topdown) και υλοποίηση από κάτω προς τα πάνω(bottomup). Συνεπώς, για τη συγκεκριμένη άσκηση η ανάλυση γίνεται ως εξής: ο adder των τεσσάρων bit αποτελείται από τρεις διαφορετικές μονάδες που συνδέονται κατάλληλα μεταξύ τους. Αυτές φαίνονται στο Σχήμα 1.

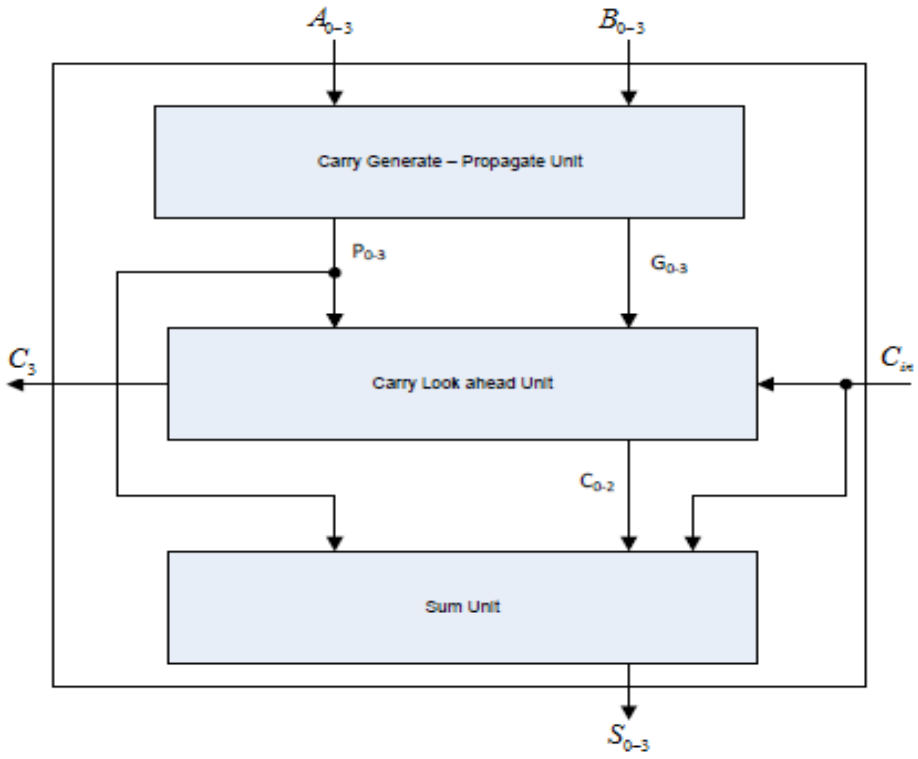
Συγκεκριμένα:
1)CarryGenerate/PropagateUnit: υπολογίζει τα τέσσερα(4) σήματα propagate και τα τέσσερα(4) σήματα generate. Ουσιαστικά κάθε μονάδα που παίρνει δύο σήματα εισόδου και δημιουργεί το Gκαι Ρείναι ένας ημιαθροιστής (HalfAdder – HA)

2)CarryLookAheadUnit: υπολογίζει τα τρία (3) σήματα CarryLookAhead και το CarryOut.

3)SumUnit: υπολογίζει τα τέσσερα(4) σήματα του αθροίσματος(sum).

Στο στάδιο της υλοποίησης υλοποιούμε κάθε μονάδα ξεχωριστά, και μετά ενώνουμε τις τρεις(3) μονάδες μεταξύ τους.

Συνολικά πρέπει να υλοποιήσετε τέσσερα(4) διαφορετικά modules σε τέσσερα(4) διαφορετικά αρχεία.



Σχήμα1:4-bit Carry Look Ahead Adder block diagram

Παρατηρούμε ότι σε σχέση με την θεωρία, η αποσύνθεση (decomposition) του κυκλώματος φαίνεται στις εξισώσεις του CLA, ενώ η κατάτμηση (partitioning) του κυκλώματος φαίνεται στο Σχήμα 1. Προφανώς η κατάτμηση είναι μία λογική προσέγγιση επί της αποσύνθεσης, αλλά θα μπορούσαν να υπάρχουν και άλλες προσεγγίσεις.

Παρατηρήσεις/Σημειώσεις

(1) Επαληθεύσετε τη λειτουργία κάθε υποκυκλώματος αφού το σχεδιάσετε ξεχωριστά, με χρήση ξεχωριστού testbench.
(2) Η σύνδεση των modules γίνεται ιεραρχικά, δηλ. έχουμε modules που συνδέονται μεταξύ τους μέσα σε ένα άλλο moduleπου βρίσκεται σε ένα παραπάνω επίπεδο. Αυτό με τη σειρά του μπορεί να συνδέεται με άλλα modules, μέσω ενός module που βρίσκεται σε ένα ακόμη παραπάνω επίπεδο κ.ο.κ. Σκεφτείτε πως θα συνδέσετε τις μονάδες για τη δημιουργία

του carrylookahead. Θα χρειαστεί να δημιουργήσετε instances.
(3) Δημιουργήσετε topLevel στο οποίο θα συνδέσετε όλα τα σήματα που θέλετε να «δείτε» στην προσομοίωση. Φτιάξετε το τελικό testbench, και συνδέσετε το με το topLevel.

Παραδοτέα:

Πηγαίος κώδικας VHDL με επαρκή σχόλια, κυματομορφές προσομοίωσης, παρουσίαση κυκλώματος, επίσης να δείξετε τη διαδικασία λύσης, τι χρησιμοποιήσατε από τη θεωρία και πως.

Βαθμολογία:

Διεξαγωγή εργαστηρίου	Προετοιμασία:30%
	Προσομοίωση: 70%

ΠΡΟΣΟΧΗ!

1) Η προεργασία να είναι σε ηλεκτρονική μορφή, σε αρχείο pdf.

2) Η έλλειψη προετοιμασίας οδηγεί σε απόρριψη.

3)Η διαπίστωση αντιγραφής σε οποιοδήποτε σκέλος της άσκησης οδηγεί στην απόρριψη από το σύνολο των εργαστηριακών ασκήσεων. Αυτό γίνεται οποιαδήποτε στιγμή στη διάρκεια του εξαμήνου.

ΚΑΛΗ ΕΠΙΤΥΧΙΑ!☺