



ΠΟΛΥΤΕΧΝΕΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΗΜΜΥ

ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΕΠΕΞΕΡΓΑΣΤΩΝ & ΥΛΙΚΟΥ  
ΕΡΓΑΣΤΗΡΙΑΚΕΣ ΑΣΚΗΣΕΙΣ ΓΙΑ ΤΟ ΜΑΘΗΜΑ:  
ΗΡΥ 203 - ΠΡΟΧΩΡΗΜΕΝΗ ΛΟΓΙΚΗ ΣΧΕΔΙΑΣΗ

ΕΑΡΙΝΟ ΕΞΑΜΗΝΟ 2021

## Εργαστήριο 2

ΕΞΟΙΚΕΙΩΣΗ ΜΕ ΤΗ ΓΛΩΣΣΑ ΠΕΡΙΓΡΑΦΗΣ ΥΛΙΚΟΥ VHDL ΚΑΙ  
ΤΗΝ ΙΕΡΑΡΧΙΚΗ ΣΧΕΔΙΑΣΗ

ΕΚΠΟΝΗΣΗ: Καθ. Α. Δόλλας, Δρ. Ε. Σωτηριάδης

ΣΥΝΕΡΓΑΤΕΣ: Δρ. Ευριπίδης Σωτηριάδης, Μ. Κιμιωνής,

ΕΚΔΟΣΗ : 12.0  
Χανιά

## Σκοπός

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

## 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
C <sub>in</sub>	είσοδος	1
S	έξοδος	4
C <sub>3</sub>	έξοδος	1

Πίνακας 1: Είσοδοι - έξοδοι κυκλώματος

Εξισώσεις που θα υλοποιηθούν (οι εξισώσεις του CLA)

$$P_i = A_i \oplus B_i$$

$$G_i = A_i \cdot B_i$$

$$C_0 = G_0 + P_0 \cdot C_{in}$$

$$C_1 = G_1 + P_1 \cdot G_0 + P_1 \cdot P_0 \cdot C_{in}$$

$$C_2 = G_2 + P_2 \cdot G_1 + P_2 \cdot P_1 \cdot G_0 + P_2 \cdot P_1 \cdot P_0 \cdot C_{in}$$

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

$$S_0 = A_0 \oplus B_0 \oplus C_{in}$$

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

**Το κύκλωμα λειτουργεί ως εξής:**

Τα A,B είναι οι αριθμητικές εισόδους του κυκλώματος και το C<sub>in</sub> είναι το κρατούμενο εισόδου. Ο αθροιστής είναι τύπου 4-bit Carry Look Ahead (CLA). Το αποτέλεσμα της πράξης φαίνεται στο S και το κρατούμενο εξόδου στο C<sub>3</sub>.

### 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.

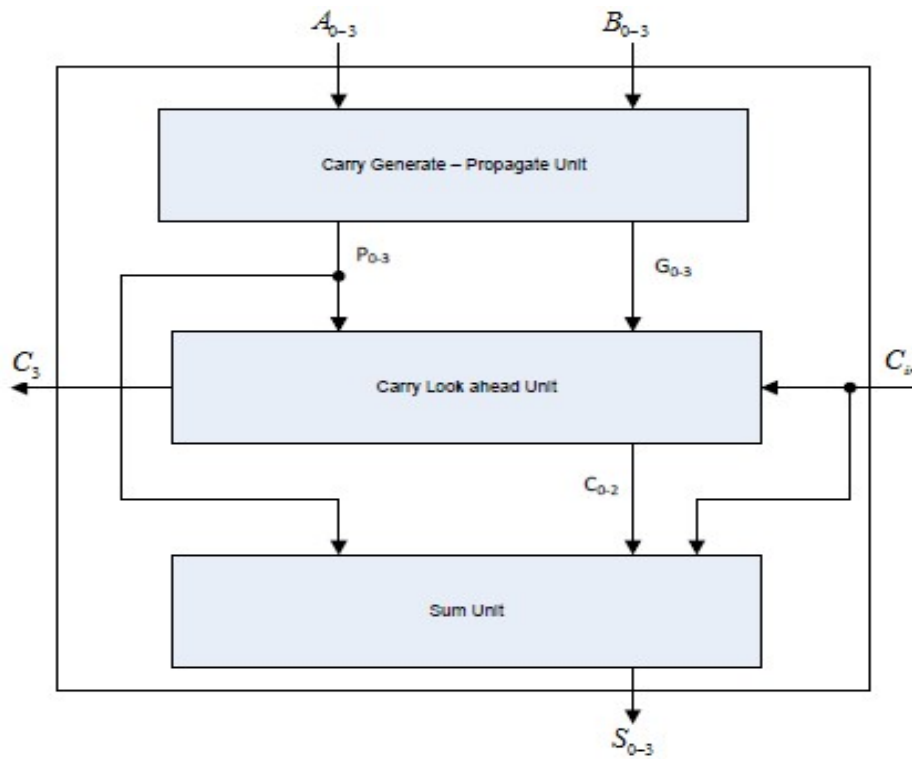
Συγκεκριμένα:

**1) Carry Generate/Propagate Unit:** υπολογίζει τα τέσσερα (4) σήματα propagate και τα τέσσερα (4) σήματα generate. Ουσιαστικά κάθε μονάδα που παίρνει δύο σήματα εισόδου και δημιουργεί το G και P είναι ένας ημιαθροιστής (HalfAdder - HA)

**2) Carry Look Ahead Unit:** υπολογίζει τα τρία (3) σήματα Carry Look Ahead και το CarryOut.

**3) Sum Unit:** 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 κατάρτιση (partitioning) του κυκλώματος φαίνεται στο Σχήμα 1. Προφανώς η κατάρτιση 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 module που 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 carry look ahead. Θα χρειαστεί να δημιουργήσετε instances.
- (3) Δημιουργήστε topLevel στο οποίο θα συνδέσετε όλα τα σήματα που θέλετε να «δείτε» στην προσομοίωση. Φτιάξτε το τελικό testbench, και συνδέσετε το με το topLevel.

Παραδοτέα:

Ένα αρχείο .zip (ή ανάλογο) με τα παρακάτω αρχεία

- 1) Κυκλώματα και block diagrams σε ηλεκτρονική μορφή (όχι φωτογραφίες χειρόγραφων) σε μορφή αρχείου .pdf
- 2) Κώδικες VHDL (όχι ολόκληρο το project της Xilinx) σε μορφή αρχείου .vhd
- 3) Screenshots από τα simulations και σχόλια που εξηγούν γιατί το simulation είναι επαρκές.

Κυκλώματα και Block diagrams :20%
Κώδικες :50%
Προσομοίωση : 30%

### **ΠΡΟΣΟΧΗ!**

**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. Ως αντιγραφή νοείται και μέρος της αναφοράς, π.χ. σχήματα.**

**ΚΑΛΗ ΕΠΙΤΥΧΙΑ!☺**