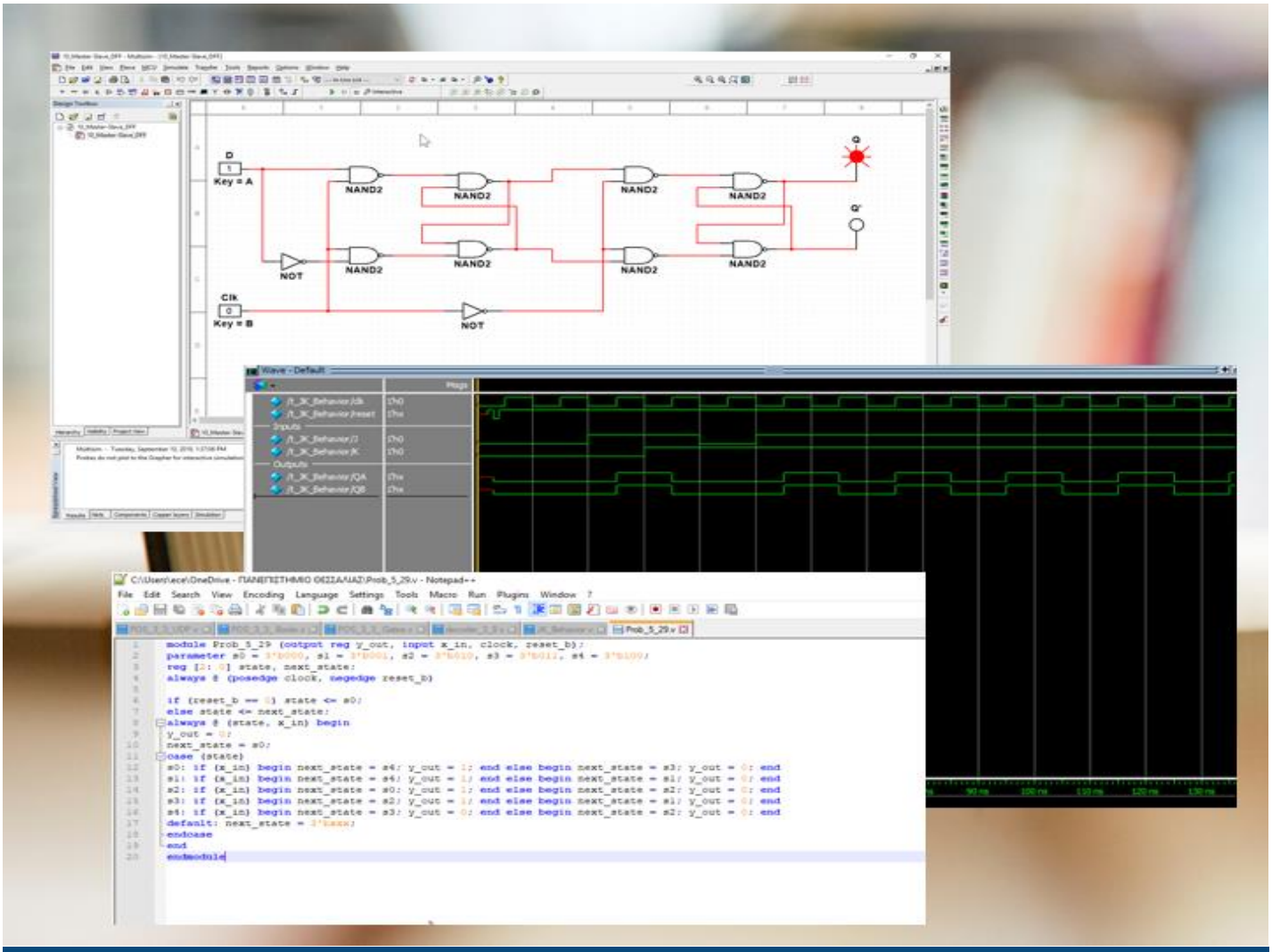
ΠΑΝΕΠΙΣΤΗΜΙΟ ΘΕΣΣΑΛΙΑΣ

ΤΜΗΜΑ ΗΛΕΚΤΡΟΛΟΓΩΝ ΜΗΧΑΝΙΚΩΝ & ΜΗΧΑΝΙΚΩΝ ΥΠΟΛΟΓΙΣΤΩΝ



ECE119 Ψηφιακή Σχεδίαση

Εργαστηριακές ασκήσεις, Multisim - Verilog

Lab 5: Binary Conversion and Adders

Καραμπερόπουλος Δημήτρης

Σεπτέμβριος 2026

Lab 5

Required Tools and Technology 1
Lab 5: Binary Conversion and Adders 2
Learning Objectives 2
Expected Deliverables 2
5.1 Theory and Background 3
5.2 Simulate: Building a Half-Adder Circuit 6
5.3 Simulate: Building a Full-Adder Circuit 8
5.4 Conclusion 10
5.5 Exercise: HDL – Verilog, Haff adder 12
5.6 Exercise: HDL – Verilog, Full adder 12
5.7 Exercise: HDL – Verilog, Ripple Carry 4-bit, Add-Sub 13
5.8 Exercise: HDL - Verilog - Test Circuit 14

Required Tools and Technology

Software: NI Multisim 14.0 or newer	✓ **Install Multisim:** http://www.ni.com/gate/gb/GB_ACADEMICEVALMULTISIM/US ✓ **View Help:** http://www.ni.com/multisim/technical-resources/
MultisimLive	✓ https://www.multisim.com/

Lab 5: Binary Conversion and Adders

In the first lab, you explored simple logic circuits with two binary inputs, using truth tables to describe their behavior.

As digital circuits become more complex, the number of possible input combinations increases rapidly, making truth-table-based design less practical.

In this lab, you will explore two fundamental concepts:

Binary and BCD Representation

How numbers are represented in digital systems and how binary values can be converted into Binary-Coded Decimal (BCD).

Binary Adders

How combinational logic circuits perform binary addition.

Learning Objectives

In this lab, students will:

1. Construct half and full adders with logic gates and create truth tables from them
2. Confirm the truth table for a full adder.
3. Design and simulate a 4-bit adder-subtractor in Verilog, and interpret its results for signed and unsigned arithmetic using the carry-out and overflow flags.

Expected Deliverables

In this lab, you will submit the following:

- Answers to the theory and conclusion questions.
- Completed truth tables.
- Multisim circuit files.
- Verilog HDL files for the Half Adder, Full Adder, and 4-bit Adder-Subtractor.
- Verilog testbench for the 4-bit Adder-Subtractor.
- Completed simulation results table (S, C, V).
- Answers to the simulation evaluation questions.

5.1 Theory and Background

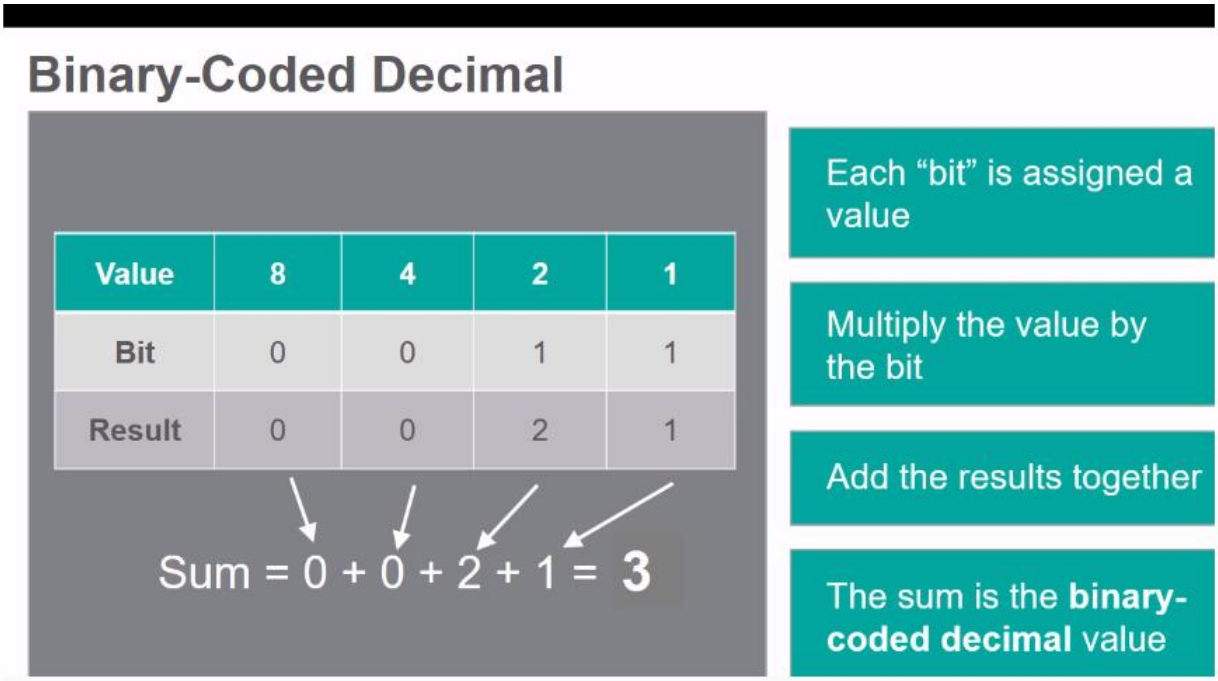


Figure 5-1 Video. View the video here: https://youtu.be/YYGAPRraclY

Video Summary

- Binary numbers can be converted into BCD representation to facilitate decimal digit representation and display in digital systems.
- Half adders can be represented in Multisim by a single component or by creating them using an AND and an XOR gate
- Half adders have two inputs and two outputs
- Full adders can be represented in Multisim by a single component or two AND gates, two XOR gates and one OR gate.
- Full adders have three inputs and two outputs

Binary-Coded Decimals

Let's look at the following example of a 4-bit binary number (four binary numbers code to one decimal number):

0101

The number furthest to the right is given the value of 1 (similar to the ones column in regular addition). The position immediately to its left is given the value of 2 (similar to a

tens column). The value to the left of this is 4 (similar to the hundreds column). The pattern you will notice is that the values assigned to the number on the left increase by twice the value of the number before it.

In our example, from left to right, the BCD value for the four-bit binary code above, adding up the values is:

0 + 4 + 0 +1 = 5

Conversions that result in a single digit number (0-9) are already in BCD format. In order to output the decimal value of a binary number greater than 9 in circuitry, a Binary-to-BCD converter such as a Shift-Add-3 algorithm must be used.

Adders

Half-Adders:

- A half-adder does binary addition on two inputs (A and B).
- The two outputs are labeled sum (S) and carry (C).
- Half adders can be built with:
  - an XOR gate and an AND gate (shown on the left).
  - a component in Multisim (shown on the right).

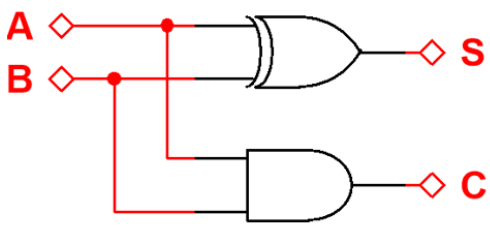


Figure 5-2 Half Adder from Gates

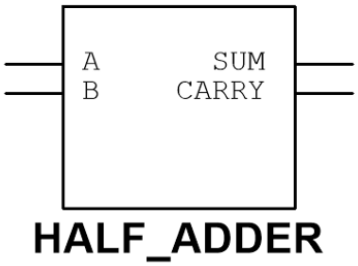


Figure 5-3 Half Adder Multisim

Full-Adders:

- A full-adder does binary addition on three inputs: A, B, and C_{in}.
- Full adders usually work in a cascade fashion where they are used to add binary numbers with an increasing number of bits.
- The two outputs are sum (S) and carry (C_{out}).
- You will notice that full adders can also use logic gates or a component.

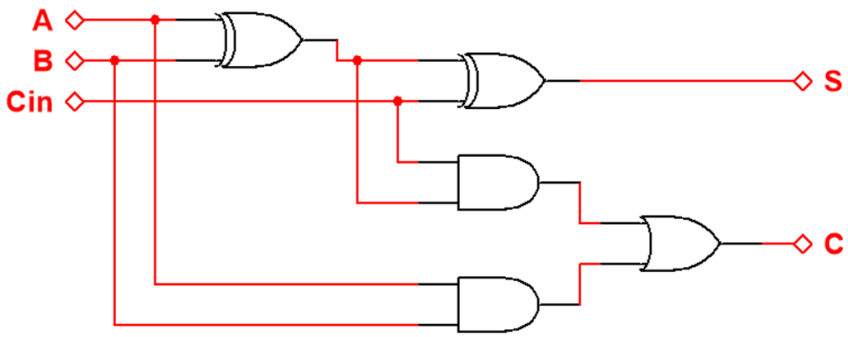


Figure 5-4 Full Adder from Gates

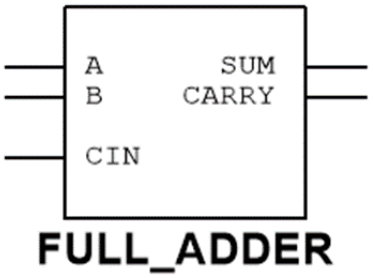


Figure 5-5 Full Adder Multisim

5-1 What are the similarities and differences between half adders and full adders? Select the right answers (multiple choices)

a) Τόσο ο half adder όσο και ο full adder είναι συνδυαστικά κυκλώματα. Και οι δύο προσθέτουν δύο ψηφία ενός bit.

b) Τόσο ο half adder όσο και ο full adder είναι συνδυαστικά κυκλώματα. Ωστόσο, ο πρώτος προσθέτει δύο ψηφία ενός bit, ενώ ο δεύτερος προσθέτει τρία ψηφία.

c) Ο half adder έχει δύο τιμές εισόδου a και b που αντιπροσωπεύουν τα bits δεδομένων. Ο full adder έχει μια επιπλέον είσοδο, το κρατούμενο από την κατώτερη βαθμίδα (Cin).

d) Τόσο ο half adder όσο και ο full adder έχουν δύο τιμές εισόδου a και b που αντιπροσωπεύουν τα bits δεδομένων.

e) Ο half adder έχει δύο εξόδους. Η μία είναι το άθροισμα της εισόδου (S) και η άλλη είναι το κρατούμενο του αθροίσματος (C). Ο full adder έχει επίσης δύο εξόδους (S; Cout).

f) Ο half adder έχει δύο εξόδους. Η μία είναι το άθροισμα της εισόδου (S) και η άλλη είναι το κρατούμενο του αθροίσματος (C). Ο full adder έχει τρεις εξόδους (S; C1, C2).

g) Στην περίπτωση του half adder, το κρατούμενο από την κατώτερη βαθμίδα προστίθεται. Στην περίπτωση του full adder, το προηγούμενο κρατούμενο δεν αθροίζεται.

h) Στην περίπτωση του half adder, το κρατούμενο από την κατώτερη βαθμίδα δεν προστίθεται. Στην περίπτωση του full adder, το προηγούμενο κρατούμενο μεταφέρεται και αθροίζεται.

i) Ο half adder αποτελείται από μία πύλη XOR και μία πύλη AND. Ο full adder αποτελείται πρακτικά από 2 half adders συνδυασμένους. Αποτελείται από δύο πύλες XOR, δύο πύλες AND και μία πύλη OR.

5.2 Simulate: Building a Half-Adder Circuit

Half-Adder Circuits

Half-adder circuits can be built using a combination of logic gates.

- Launch Multisim.
- Open a new circuit.
  - Select **File>>New**.
  - In the menu that appears, select **Blank** and click **Create**.
- Connect the following circuit:
  - Place an **XOR** gate and an **AND** gate from the **Misc Digital** group.
  - Place two **INTERACTIVE_DIGITAL_CONSTANTs** from the **Sources** group.
  - Place two **PROBE_DIG_REDs** from the **Indicators** group.
  - Wire them as shown:

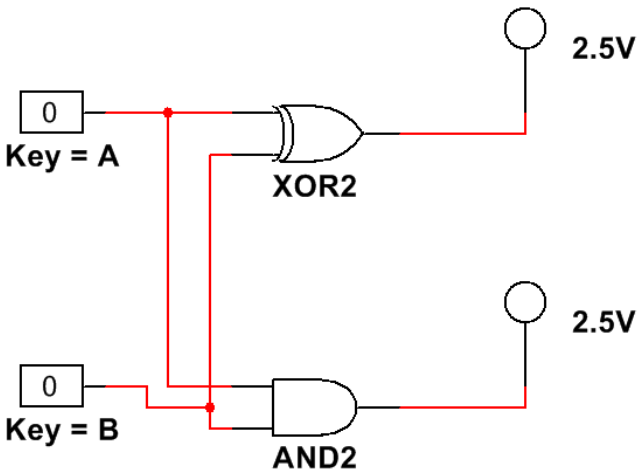


Figure 5-6 A Half Adder Circuit

- Multisim:	Όνομα αρχείου “**5_half_adder.ms14**”.
	Προσθήκη στο zip file με όνομα ”Lab05_ονοματεπώνυμο_AM.zip”
ή	
- MultisimLive:	Όνομα αρχείου “**5_half_adder.png**”.
Schematic image	Προσθήκη στο zip file με όνομα ”Lab05_ονοματεπώνυμο_AM.zip”

- Click the **Run** button to begin simulating the circuit.



Figure 5-7 Run Button

- Using the **A** and **B** keys, vary the inputs into the circuit.

5-2 Fill out the truth table below.

A	B	XOR (SUM)	AND (CARRY)
0	0		
0	1		
1	0		
1	1		

- Stop the simulation by clicking the **Stop** button.



Figure 5-8 Stop Button

Can you see how the XOR and AND gates represent the sum and carry of the numbers A and B added together?

- 0 + 0 = 0
- 1 + 0 or 0 + 1 = 1, with no carry.
- 1 + 1 = 2, but 2 is not a binary number. In binary, 2 is represented as 10. The 1 is the carry and the 0 is the sum.

5.3 Simulate: Building a Full-Adder Circuit

Full-Adder Circuits

Connect the following circuit:

- Place two **XOR** gates, two **AND** gates and one **OR** gate from the **Misc Digital** group.
- Wire them as shown:

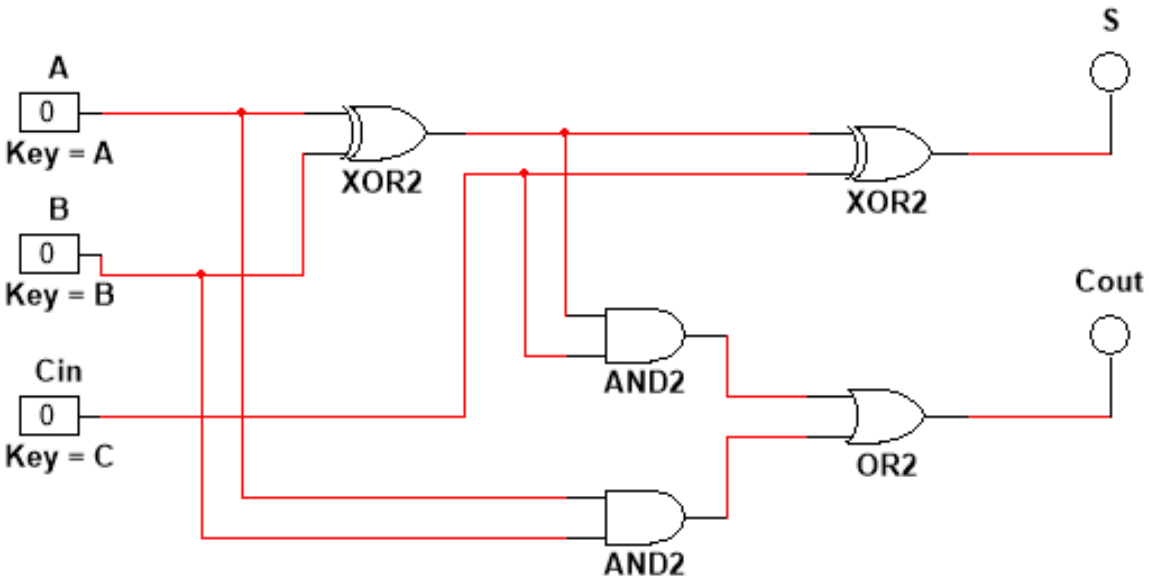


Figure 5-9 A Full Adder Circuit

- Multisim:	Όνομα αρχείου “**5_full_adder.ms14**”.
	Προσθήκη στο zip file με όνομα ”Lab05_ονοματεπώνυμο_AM.zip”
ή	
- MultisimLive:	Όνομα αρχείου “**5_full_adder.png**”.
Schematic image	Προσθήκη στο zip file με όνομα ”Lab05_ονοματεπώνυμο_AM.zip”

- Click the **Run** button to begin simulating the circuit.
- Using the **A, B** and **C** keys, vary the inputs into the circuit.

5-3 Fill out the truth table below.

A	B	C_{in}	SUM	C_{out}
0	0	0		
0	0	1		
0	1	0		
0	1	1		
1	0	0		
1	0	1		
1	1	0		
1	1	1		

Again, can you see how the truth table represents the sum and carry of the numbers A, B, and C_{IN} added together?

- 0 + 0 + 0 = 0
- 0 + 0 + 1 = 1, with no carry.
- 0 + 1 + 1 = 2, but 2 is not a binary number. In binary, 2 is represented as 10. The 1 is the carry and the 0 is the sum.
- 1 + 1 + 1 = 3. In binary, 3 is represented as 11. The first 1 is the carry and the second 1 is the sum.

5.4 Conclusion

5-4 Under what conditions would you use a half adder? A full adder?
Select the right answers (multiple choices)

a) Half adder: Μπορεί να προσθέσει δύο αριθμούς του ενός bit, λαμβάνοντας υπόψη την ύπαρξη προηγούμενου κρατουμένου

b) Half adder: Μπορεί να προσθέσει δύο αριθμούς του ενός bit, χωρίς όμως να λαμβάνει υπόψη την ύπαρξη προηγούμενου κρατουμένου.

c) Full adder: Μπορεί να προσθέσει δύο αριθμούς του ενός bit, λαμβάνοντας υπόψη την ύπαρξη προηγούμενου κρατουμένου.

d) Full adder: Μπορεί να προσθέσει δύο αριθμούς του ενός bit, χωρίς όμως να λαμβάνει υπόψη την ύπαρξη προηγούμενου κρατουμένου.

e) Half adder: Προσθέτει δύο αριθμούς του ενός bit

f) Half adder: Προσθέτει τρεις αριθμούς του ενός bit

g) Full adder: Προσθέτει δύο αριθμούς του ενός bit

h) Full adder: Προσθέτει τρεις αριθμούς του ενός bit

5-5 When converting binary numbers to BCD:

A. Numbers are given specific values depending on their position in the sequence
B. The value of the number on the left increases by twice the value of the number on the right
C. Single digit numbers are already in BCD format
D. All of the above

5-6 Half-adders do binary addition on two inputs using two logic gates. What is the correct combination of these logic gates?

A. XOR and OR
B. XOR and AND
C. XNOR and AND
D. NOT and OR

5-7 The outputs of a half adder are:

A. Carry and C_{in}
B. Sum and Carry
C. Sum and C_{in}

D. Carry and C_{out}

5-8 Which is a feature of full-adders?

A. They have two inputs
B. They have three outputs
C. They work in a cascading fashion
D. All of the above

5-9 When connecting full-adders together in Multisim, the Carry of one is connect to _____ input of the next.

A. A
B. B
C. C_{in}
D. None of the above

5.5 Exercise: HDL – Verilog, Half adder

Ο Ημιαθροιστής (Half Adder – H/A) είναι ένα λογικό κύκλωμα το οποίο στην είσοδό του δέχεται 2 δυαδικά ψηφία A και B και στην έξοδό του δίνει το άθροισμα τους S και το κρατούμενο C της πρόσθεσής τους.

Γράψτε την περιγραφή HDL σε επίπεδο πυλών του παραπάνω κυκλώματος.

Ονομάστε το module: "half_adder".

- Verilog:	Όνομα αρχείου "**half_adder.v** ". Προσθήκη στο zip file με όνομα "Lab05_ονοματεπώνυμο_AM.zip"

5.6 Exercise: HDL – Verilog, Full adder

Κατόπιν, σκεφτείτε πως μπορείτε να υλοποιήσετε έναν Πλήρη Αθροιστή (Full Adder – Π/A) ο οποίος σχηματίζει το αριθμητικό άθροισμα 3 bits εισόδου. Δηλαδή, έχει 3 εισόδους και 2 εξόδους. Δύο από τα bits εισόδου είναι τα 2 bits τα οποία προσθέτουμε. Η τρίτη είσοδος αναπαριστά το κρατούμενο της αμέσως προηγούμενης λιγότερο σημαντικής θέσης. Συμβολίστε με S το άθροισμα και C το κρατούμενο.

Έχετε ήδη γράψει τον πίνακα αληθείας και σχεδιάσατε την τελική του μορφή χρησιμοποιώντας 2 H/A και μια πύλη OR στο Multisim.

Γράψτε τώρα την περιγραφή HDL σε επίπεδο πυλών του παραπάνω κυκλώματος.

Θα πρέπει να χρησιμοποιήσετε 2 στιγμιότυπα από τον ημιαθροιστή της άσκησης 5.5

Θα πρέπει να δώσετε την οδηγία (directive): **`include "half_adder.v"**
Θα πρέπει να την γράψετε ακριβώς πριν από το module (έξω από αυτό).
Με αυτόν τον τρόπο συμπεριλαμβάνετε το περιεχόμενο ενός άλλου αρχείου, στο σημείο όπου βρίσκετε η οδηγία `include.

Ονομάστε το module: "full_adder".

- Verilog:	Όνομα αρχείου "**full_adder.v**". Προσθήκη στο zip file με όνομα "Lab05_ονοματεπώνυμο_AM.zip"

5.7 Exercise: HDL – Verilog, Ripple Carry 4-bit Adder-Subtractor

Γράψτε την περιγραφή HDL σε επίπεδο πυλών ενός 4-bit δυαδικού αθροιστή-αφαιρέτη (Adder-Subtractor), σύμφωνα με το σχήμα 4.13 του βιβλίου Morris Mano.

Το κύκλωμα θα πρέπει να εκτελεί πρόσθεση και αφαίρεση δύο δυαδικών αριθμών 4 bits (A και B), οι οποίοι μπορούν να ερμηνευθούν είτε ως μη προσημασμένοι (unsigned) είτε ως προσημασμένοι (signed, two's complement).

Η λειτουργία καθορίζεται από την είσοδο ελέγχου M:

M=0: Πρόσθεση (A+B)
M=1: Αφαίρεση (A-B)

Το κύκλωμα θα διαθέτει τις ακόλουθες εξόδους:

S[3:0]: Αποτέλεσμα της πράξης (4 bits).
C: Τελικό κρατούμενο (Carry-out).
V: Ένδειξη υπερχείλισης (Signed Overflow).

Για την υλοποίηση:

Χρησιμοποιήστε 4 στιγμιότυπα του Full Adder της άσκησης 5.6.

Υλοποιήστε επίσης την έξοδο υπερχείλισης V χρησιμοποιώντας τα κατάλληλα κρατούμενα των Full Adders, σύμφωνα με τη σχέση V = C_3 XOR C_4 που παρουσιάστηκε στο μάθημα.

Χρησιμοποιήστε την οδηγία **`include "full_adder.v"** πριν από τη δήλωση του module.

Διευκρίνιση:

Δεν χρειάζεται να δημιουργήσετε διαφορετικό κύκλωμα για signed και unsigned αριθμούς. Το ίδιο κύκλωμα εκτελεί την ίδια δυαδική πράξη. Εκείνο που αλλάζει είναι η ερμηνεία των εισόδων και των εξόδων.

Ονομάστε το module: “add_sub_4_bit”.

- Verilog:	Όνομα αρχείου “**add_sub_4_bit.v**”. Προσθήκη στο zip file με όνομα ”Lab05_ονοματεπώνυμο_ΑΜ.zip”

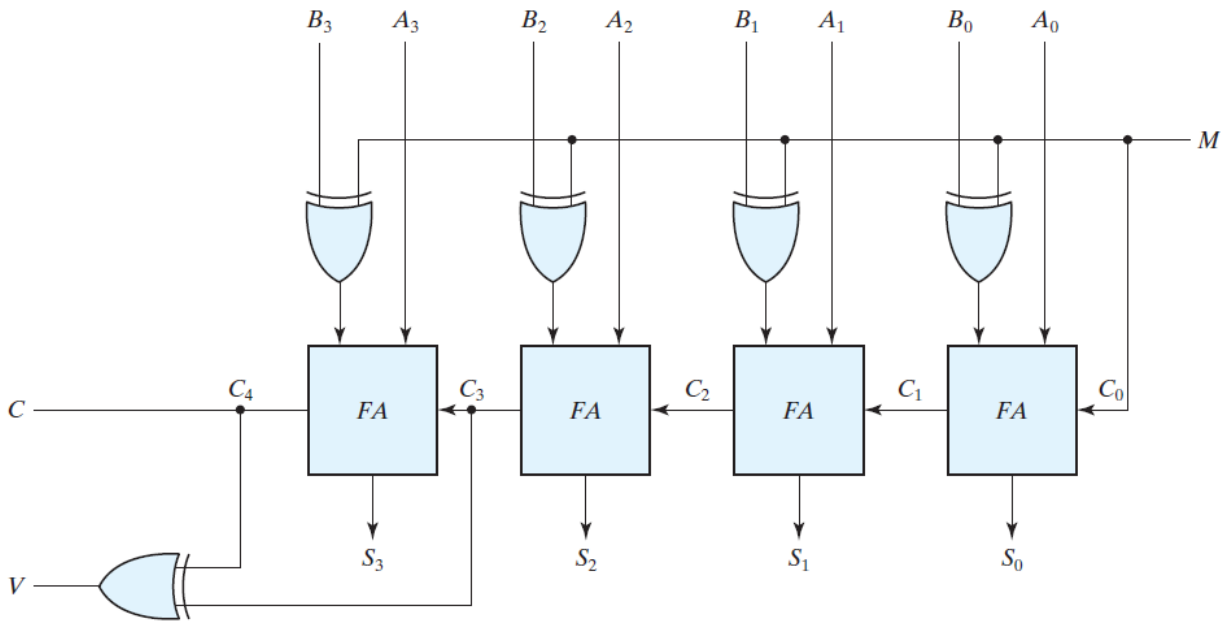


FIGURE 4.13
Four-bit adder–subtractor (with overflow detection)

5.8 Exercise: HDL - Verilog - Test Circuit

Δημιουργήστε μια υπομονάδα διέγερσης (testbench) για την προσομοίωση και τον έλεγχο του κυκλώματος της άσκησης 5.7 (`add_sub_4_bit`).

Το testbench θα πρέπει να εφαρμόζει τους παρακάτω συνδυασμούς εισόδων σε διαφορετικές χρονικές στιγμές.

Χρόνος	A	B	M	Πράξη
10 ns	0011	0010	0	Unsigned: 3+2
20 ns	1111	0001	0	Unsigned: 15+1
30 ns	0101	0011	1	Unsigned: 5−3
40 ns	0011	0101	1	Unsigned: 3−5
50 ns	0011	0101	1	Signed: 3−5
60 ns	0111	0011	0	Signed: 7+3
70 ns	1001	1101	0	Signed: (−7)+(−3)

Ζητούμενα:

1. Χρησιμοποιήστε την εντολή **$monitor** για την παρακολούθηση των εισόδων και των εξόδων (A, B, M, S, C, V).
2. Χρησιμοποιήστε τις εντολές **$dumpfile** και **$dumpvars** για τη δημιουργία αρχείου VCD.
3. Ανοίξτε το αρχείο VCD στο VS Code και παρατηρήστε τις κυματομορφές.
4. Καταγράψτε τις εξόδους S, C και V για κάθε περίπτωση.
5. Ερμηνεύστε τα αποτελέσματα σύμφωνα με τους κανόνες signed και unsigned αριθμητικής που παρουσιάστηκαν στο μάθημα.

Χρησιμοποιήστε την οδηγία **`include "add_sub_4_bit.v"** πριν από τη δήλωση του module.

Ονομάστε το module: "t_add_sub_4_bit".

- Verilog: Όνομα αρχείου " **t_add_sub_4_bit.v** ".
Προσθήκη στο zip file με όνομα "Lab05_ονοματεπώνυμο_ΑΜ.zip"

Αξιολόγηση αποτελεσμάτων προσομοίωσης

5-10 Με βάση τα αποτελέσματα της προσομοίωσης, συμπληρώστε τον παρακάτω πίνακα, καταγράφοντας τις εξόδους S, C και V για κάθε περίπτωση.

Στην τελευταία στήλη εξηγήστε το αποτέλεσμα της πράξης, λαμβάνοντας υπόψη αν οι αριθμοί ερμηνεύονται ως signed ή unsigned.

Χρόνος	Πράξη	S [3:0]	C	V	Ερμηνεία αποτελέσματος
10 ns	Unsigned: 3+2				
20 ns	Unsigned: 15+1				
30 ns	Unsigned: 5−3				
40 ns	Unsigned: 3−5				
50 ns	Signed: 3−5				
60 ns	Signed: 7+3				
70 ns	Signed: (−7)+(−3)				

Ερωτήσεις αξιολόγησης

5-11 Στην unsigned πρόσθεση δύο αριθμών 4 bits, πότε χρειάζεται να χρησιμοποιήσουμε και το C για το πλήρες αποτέλεσμα;

A. Όταν C=0, επειδή υπάρχει υπερχείλιση.
B. Όταν C=1, επειδή το αποτέλεσμα απαιτεί 5 bits.
C. Όταν V=1, επειδή το αποτέλεσμα είναι αρνητικό.
D. Πάντα, ανεξάρτητα από την τιμή του C.

5-12 Στην unsigned αφαίρεση, τι σημαίνει C=0;

A. Το αποτέλεσμα είναι μηδέν.
B. Υπάρχει signed overflow.
C. Έχει γίνει δανεισμός (borrow), επειδή A μικρότερο από B.
D. Η αφαίρεση ολοκληρώθηκε χωρίς δανεισμό.

5-13 Τι σημαίνει V=1 σε μια signed πράξη με αριθμούς 4 bits;

A. Το αποτέλεσμα είναι αρνητικό.
B. Υπάρχει τελικό κρατούμενο (carry-out).
C. Το αποτέλεσμα είναι σωστό, αλλά χρειάζεται συμπλήρωμα ως προς 2.
D. Το μαθηματικό αποτέλεσμα δεν μπορεί να αναπαρασταθεί σε 4 signed bits.

5-14 Γιατί οι δοκιμές στα 40 ns και 50 ns δίνουν τις ίδιες εξόδους S, C και V, παρότι ερμηνεύονται ως unsigned και signed αντίστοιχα;

A. Επειδή το κύκλωμα εκτελεί τις ίδιες δυαδικές πράξεις, ανεξάρτητα από την ερμηνεία των αριθμών.
B. Επειδή το C είναι πάντα ίδιο με το V.
C. Επειδή το κύκλωμα μετατρέπει αυτόματα τους unsigned αριθμούς σε signed.
D. Επειδή η αφαίρεση δεν επηρεάζεται ποτέ από τις τιμές των εισόδων.