ΠΑΝΕΠΙΣΤΗΜΙΟ
ΔΥΤΙΚΗΣ ΜΑΚΕΔΟΝΙΑΣ
ΤΜΗΜΑ ΜΗΧΑΝΙΚΩΝ
ΠΛΗΡΟΦΟΡΙΚΗΣ & ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

Εισαγωγή στην Επιστήμη των Υπολογιστών
Διδάσκων: Παντελής Αγγελίδης
7ο εργαστηριακό μάθημα

Αντικείμενο: **Λογικές εντολές AND, OR, XOR, NOT**

Χρησιμοποιούμενες εντολές:

AND κάνε λογικό AND μεταξύ των bit του AC και του R (bitwise) και βάλε το αποτέλεσμα στο AC, ενημέρωσε το Z

OR κάνε λογικό OR μεταξύ των bit του AC και του R (bitwise) και βάλε το αποτέλεσμα στο AC, ενημέρωσε το Z

XOR κάνε λογικό XOR μεταξύ των bit του AC και του R (bitwise) και βάλε το αποτέλεσμα στο AC, ενημέρωσε το Z

NOT κάνε λογικό NOT στα bit του AC (bitwise) και βάλε το αποτέλεσμα στο AC, ενημέρωσε το Z

Relatively Simple CPU Simulator
http://media.pearsoncmg.com/aw/aw_carpinel_compsys_1/rscpu/web.html

1) AND

Η εντολή 'AND' εκτελεί λογικό 'ΚΑΙ' του περιεχομένου του συσσωρευτή με το περιεχόμενο του καταχωρητή R. Επηρεάζει τον καταχωρητή κατάστασης Z (flag). Χρησιμοποιείται κυρίως όταν θέλουμε να μηδενίσουμε οποιοδήποτε δυαδικό ψηφίο μιας θέσης μνήμης, χωρίς να αλλάξει η τιμή των υπολοίπων. Για να το πετύχουμε αυτό εκτελούμε τη λογική πράξη 'ΚΑΙ' ανάμεσα στο περιεχόμενο της θέσης μνήμης και μιας μάσκας που έχει '0' στις θέσεις των δυαδικών ψηφίων που πρέπει να μηδενίσουμε και '1' στη θέση των υπολοίπων

Ας υποθέσουμε ότι η θέση μνήμης 48 έχει το παρακάτω περιεχόμενο

	B_7	B_6	B_5	B_4	B_3	B_2	B_1	B_0	
48:	**1**	**1**	**1**	**1**	**0**	**1**	**0**	**1**	**->$(F5)_{16}$**

Και ότι θέλουμε να μηδενίσουμε τα δυαδικά ψηφία B_7, B_6, B_0, χωρίς να αλλάξουμε τις τιμές των υπολοίπων.
Γράψτε κώδικα που να υλοποιεί το παραπάνω και το αποτέλεσμα να αποθηκεύεται στη θέση 56 και θα είναι ο αριθμός $(00110100)_2$ ή $(34)_{16}$.
Το πρόγραμμα θα περιλαμβάνει ένα λογικό 'ΚΑΙ' μεταξύ του αριθμού στη θέση 48 με έναν αριθμό που θα έχει όλα τα ψηφία του '1' εκτός από τα B_7, B_6, B_0 που θα είναι '0', δηλαδή με τον $(00111110)_2$ ή $(3E)_{16}$.

Το πρόγραμμα θα κάνει την εξής πράξη:

	11110101		$(F5)_{16}$
AND	00111110	AND	$(3E)_{16}$
	00110100		$(34)_{16}$

2) OR

Η εντολή 'OR' εκτελεί λογικό 'Η' του περιεχομένου του συσσωρευτή με το περιεχόμενο του καταχωρητή R. Επηρεάζει τον καταχωρητή κατάστασης Z (flag). Με την εντολή αυτή μπορούμε να δώσουμε την τιμή '1' σε οποιοδήποτε δυαδικό ψηφίο μιας θέσης μνήμης χωρίς να επηρεασθεί η τιμή των υπολοίπων. Η 'μάσκα' θα πρέπει να έχει '1' στις θέσεις που πρέπει να γίνουν '1' και '0' στις υπόλοιπες.

Γράψτε κώδικα που να κάνει τα B_3, B_6, B_7 του αριθμού 00010010, '1'. Για να γίνει αυτό κάνουμε λογικό 'OR' μεταξύ του αριθμού και του 11001000 που έχει τα B_3, B_6, B_7: '1'

	00010010		$(12)_{16}$
OR	11001000	OR	$(C8)_{16}$
	11011010		$(DA)_{16}$

3) XOR

Η εντολή 'XOR' εκτελεί λογικό ΑΠΟΚΛΕΙΣΤΙΚΟ Η' του περιεχομένου του συσσωρευτή με το περιεχόμενο του καταχωρητή R. Επηρεάζει τον καταχωρητή κατάστασης Z (flag).

Με την εντολή αυτή μπορούμε να αντικαταστήσουμε οποιοδήποτε δυαδικό ψηφίο μιας θέσης μνήμης με το συμπλήρωμά του, χωρίς να επηρεασθεί η τιμή των υπολοίπων. Η 'μάσκα' θα πρέπει να έχει '1' στις θέσεις που θα αντικατασταθούν και '0' στις υπόλοιπες.

Να γραφεί κώδικας που βρίσκει το συμπλήρωμα του δυαδικού αριθμού $(01110101)_2$ ή $(75)_{16}$ κάνοντας 'ΑΠΟΚΛΕΙΣΤΙΚΟ Η' μεταξύ αυτού και του αριθμού (11111111)2 ή (FF)16.

	01110101		$(75)_{16}$
XOR	11111111	XOR	$(FF)_{16}$
	10001010		$(8A)_{16}$

4) NOT

Η εντολή 'NOT' εκτελεί λογικό NOT στα bit του AC (bitwise) και βάζει το αποτέλεσμα και πάλι στο AC.

Γράψτε κώδικα που να βρίσκει το συμπλήρωμα του αριθμού $(00010010)_2$