Εργαστήριο 2

Χειμερινό Εξάμηνο 2015-2016

Στόχοι του εργαστηρίου

- Εντολές προσπέλασης της μνήμης
- Χρήση πινάκων

Χρήση μνήμης

Τα δεδομένα αποθηκεύονται σε ξεχωριστό τμήμα μνήμης το οποίο χρησιμοποιείται αποκλειστικά για αυτό το σκοπό. Γι' αυτό τον λόγο η δήλωση του πίνακα, που θα χρησιμοποιήσετε, πρέπει να γίνει στο τμήμα **.data.** Ο τρόπος δήλωσης του πίνακα και η σύνταξη των εντολών load (lw) και store (sw), οι οποίες χρησιμοποιούνται για την μεταφορά δεδομένων μεταξύ μνήμη και καταχωρητή, εξηγούνται παρακάτω.

Δήλωση πίνακα

```
.data                # στο τμήμα .data πρέπει να δηλωθεί ο πίνακας

Name_array: .space N     # Name_array: δώστε ότι όνομα
                         # θέλετε στον πίνακα
                         # .space N:  δηλώνει ότι θέλουμε να
                         # κρατήσουμε χώρο ίσο με N bytes για
                         # τον πίνακα Name_array
```

Εντολή load (lw)

Η εντολή αυτή αποθηκεύει δεδομένα σε καταχωρητή, τα οποία έχει πάρει από συγκεκριμένη διεύθυνση της μνήμης.

Σύνταξη: **lw Rt, Address(Rs)**	Σημασία**: Rt=Memory[Address+Rs]**

Προσοχή: όπου Rt, Rs είναι καταχωρητές και όπου Address είναι το όνομα του πίνακα (label) που δώσατε στο τμήμα .data.
Με βάση την παραπάνω δήλωση (στο τμήμα .data) θα έπρεπε να γράψουμε:
lw Rt, Name_array(Rs)

Από την σύνταξη της εντολής αυτής γίνεται κατανοητό ότι ζητάμε τα δεδομένα της μνήμης στην διεύθυνση [Name_array+Rs] και τα οποία θα αποθηκευτούν στον καταχωρητή Rt.

Εντολή store (sw)

Η εντολή αυτή αποθηκεύει δεδομένα από έναν καταχωρητή σε συγκεκριμένη διεύθυνση στη μνήμη.

Σύνταξη**: sw Rt, Address(Rs)**	Σημασία: **Memory[Address+Rs]=Rt**

Ο τρόπος λειτουργίας είναι παρόμοιος με την εντολή **lw(load word).** Η λειτουργία που υλοποιεί αυτή η εντολή είναι: Αποθήκευσε τα περιεχόμενα του καταχωρητή Rt στην μνήμη και συγκεκριμένα στην διεύθυνση **[Address+Rs].**

Η διάφορα με την πρώτη εντολή είναι ότι η sw αποθηκεύει δεδομένα στην μνήμη ενώ η lw διαβάζει τα δεδομένα της μνήμης.

Ευθυγράμμιση

Όπως γνωρίζετε από την θεωρία ο MIPS είναι ένας 32bit επεξεργαστής. Αυτό συνεπάγεται ότι όλοι οι καταχωρητές του έχουν μέγεθος 32 bits ή αλλιώς 4 bytes. Ορίζουμε σαν **word** τα δεδομένα μεγέθους 4 bytes τα οποία χωράνε σε έναν register.

Όταν ορίζουμε έναν πίνακα στην μνήμη με την εντολή **.space** καθορίζουμε το μέγεθος του ίσο με έναν αριθμό από **bytes** όπως παρουσιάστηκε πριν. Εφόσον όμως τα στοιχεία που αποθηκεύονται σε αυτόν θα είναι μεγέθους 4 bytes το καθένα, πρέπει να καθορίσετε το μέγεθος του πίνακα να είναι πολλαπλάσιο του 4. Έτσι το πρώτο στοιχείο του πίνακα βρίσκεται στα bytes 0 έως 3. Αν θέλετε να το κάνετε load θα πρέπει να συντάξετε την εντολή *lw Rt, Address(Rs)* με τον Rs να έχει περιεχόμενο 0. Αν τώρα θέλετε το δεύτερο στοιχείο του πίνακα αυτό βρίσκεται στα bytes 4 έως 7. Για να το κάνετε load θα πρέπει να γράψετε την ίδια εντολή αλλά ο Rs θα πρέπει να έχει περιεχόμενο τον αριθμό 4.

Σημείωση: Όταν ορίζετε έναν πίνακα πρέπει να δηλώνετε και τι μέγεθος θα έχουν τα δεδομένα που θα αποθηκεύονται σε αυτόν. Αυτό γίνεται με την εντολή **.align n** όπου **n** έναν ακέραιος που θα δώσετε εσείς. Η εντολή αυτή σημαίνει ότι τα στοιχεία του πίνακα έχουν μέγεθος $\mathbf{2^n}$. Έτσι για έναν πίνακα 10 θέσεων με λέξεις 4 bytes πρέπει να συντάξετε την εντολή:

```
.align 2          #μέγεθος στοιχείου 4 bytes
label: .space 40  #μέγεθος πίνακα 40 bytes = 10 στοιχεία
```

Ένα παράδειγμα χρήσης μνήμης

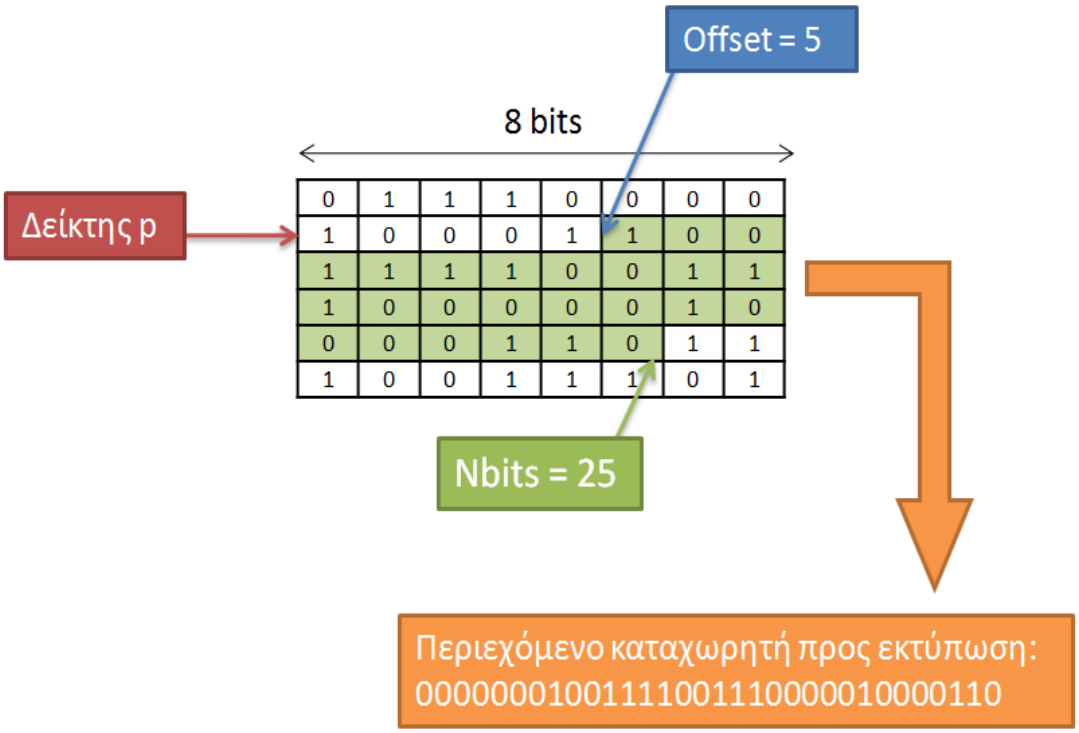
```
.data
.align 2                    # λέξεις 4 bytes
vector: .space 24           # πίνακας 6 θέσεων


.text

# …


li $t1,4                    # ένας τρόπος πρόσβασης στο 2ο στοιχείο
lw $t0,vector($t1)          # του πίνακα


# …


la $t2,vector                       # ακόμα ένας τρόπος για πρόσβαση στο 2ο
lw $t3,4($t2)               # στοιχείο του πίνακα


# …


la $t2,vector                       # ακόμα ένας τρόπος για πρόσβαση στο 2ο
addi $t2,$t2,4              # στοιχείο του πίνακα
lw $t0,0($t2)
```

Για το επόμενο εργαστήριο έχετε να υλοποιήσετε τις παρακάτω εργαστηριακές ασκήσεις. Την ώρα του εργαστηρίου θα εξετασθείτε προφορικά πάνω στον κώδικα που θα παραδώσετε.

Άσκηση 1. Bitstream Processing (8 μονάδες)

Να υλοποιηθεί ένα πρόγραμμα MIPS assembly *bits_read(char *p, int offset, int nbits)* $0 \le nbits \le 32$, $0 \le offset \le 7$ το οποίο θα μεταφέρει *nbits* ξεκινώντας από το bit *offset* της θέσης μνήμης **p* και θα τα τοποθετεί σε έναν καταχωρητή. Στο τέλος θα εκτυπώνει το περιεχόμενο του καταχωρητή στην οθόνη.

Παράδειγμα:



Να χρησιμοποιήσετε την ακολουθία του παρακάτω παραδείγματος στην υλοποίηση του προγράμματός σας:

```
.data
array:    .byte 0x70, 0x8C, 0xF3, 0x82, 0x1B, 0x9D, 0x49, 0x80, 0x50
```

Θα διαβάζετε τον δείκτη **p*, το *offset* και το *nbits* από το πληκτρολόγιο. Για λόγους απλότητας, μπορείτε να θεωρήσετε ότι ο δείκτης είναι μία ακέραια θετική τιμή που θα προσθέσετε στην διεύθυνση του array. Για παράδειγμα, με δείκτη p = 4 αναφερόμαστε στο byte 0x1B του συγκεκριμένου παραδείγματος.

Η παραπάνω ακολουθία είναι απλώς ένα παράδειγμα. Το πρόγραμμα σας θα πρέπει να δουλεύει σωστά για **οποιονδήποτε** πίνακα bytes. Θα πρέπει επίσης να δώσετε ιδιαίτερη προσοχή σε ακραίες περιπτώσεις όπως *nbits <= 8*.

Θεωρητική Άσκηση (2 μονάδες)

Η παρακάτω άσκηση είναι θεωρητική και πρέπει να παραδοθεί γραμμένη στο χαρτί κατά την διάρκεια της εξέτασης του εργαστηρίου. Κατά την διάρκεια της εξέτασης είναι πιθανόν να σας κάνουμε ερωτήσεις επιπλέον αλλά σχετικές με τις θεωρητικές ασκήσεις για να διαπιστώσουμε εάν καταλάβατε την λύση που δώσατε.

Κωδικοποίηση Εντολών. Η επόμενη ερώτηση αναφέρεται στην κωδικοποίηση συνόλου εντολών.

1) Θεωρείστε την περίπτωση ενός επεξεργαστή με 64 καταχωρητές στον οποίο το μήκος όλων των εντολών είναι 15 bits. Έχουμε τις παρακάτω κωδικοποιήσεις εντολών:

   a) 7 εντολές με διευθυνσιοδότηση 2 καταχωρητών, και 60 εντολές με διευθυνσιοδότηση 1 καταχωρητή. Πόσες εντολές χωρίς διευθυνσιοδότηση καταχωρητών μπορούμε να έχουμε;

   b) 7 εντολές με διευθυνσιοδότηση 2 καταχωρητών και 10 εντολές χωρίς διευθυνσιοδότηση καταχωρητών. Πόσες εντολές του 1 καταχωρητή μπορούμε να έχουμε;

Μην απαντάτε με ένα απλό ναι ή όχι; εξηγήστε λεπτομερώς την απάντησή σας και δώστε τα διαφορετικά formats των εντολών καθώς και το μήκος του κάθε πεδίου για κάθε διαφορετικό format.

Θα πρέπει να στέλνετε με email τις λύσεις των εργαστηριακών ασκήσεων σας στους διδάσκοντες στο uth_ece232lab@gmail.com.

Το email σας θα πρέπει να περιέχει ως attachment **ένα zip file** με τον κώδικα σας.

Κάθε διαφορετική άσκηση στην εκφώνηση θα βρίσκεται και σε διαφορετικό asm file. **Το όνομα των asm files θα ΠΡΕΠΕΙ να αρχίζει με το ΑΕΜ σας.**
Για παράδειγμα, το lab1.zip θα περιέχει 2 asm files, ένα για κάθε μία από τις ασκήσεις του lab1, με ονόματα 9999_lab1a.asm, 9999_lab1b.asm για τον φοιτητή με ΑΕΜ 9999.

Το email σας θα έχει Subject: CE232, lab N (N ο αριθμός του lab, N=1, 2, …). Το email σας θα έχει body: το όνομα σας και το ΑΕΜ σας.

Θα πρέπει να στέλνετε το email σας πριν βγείτε από την εξέταση του εργαστηρίου.