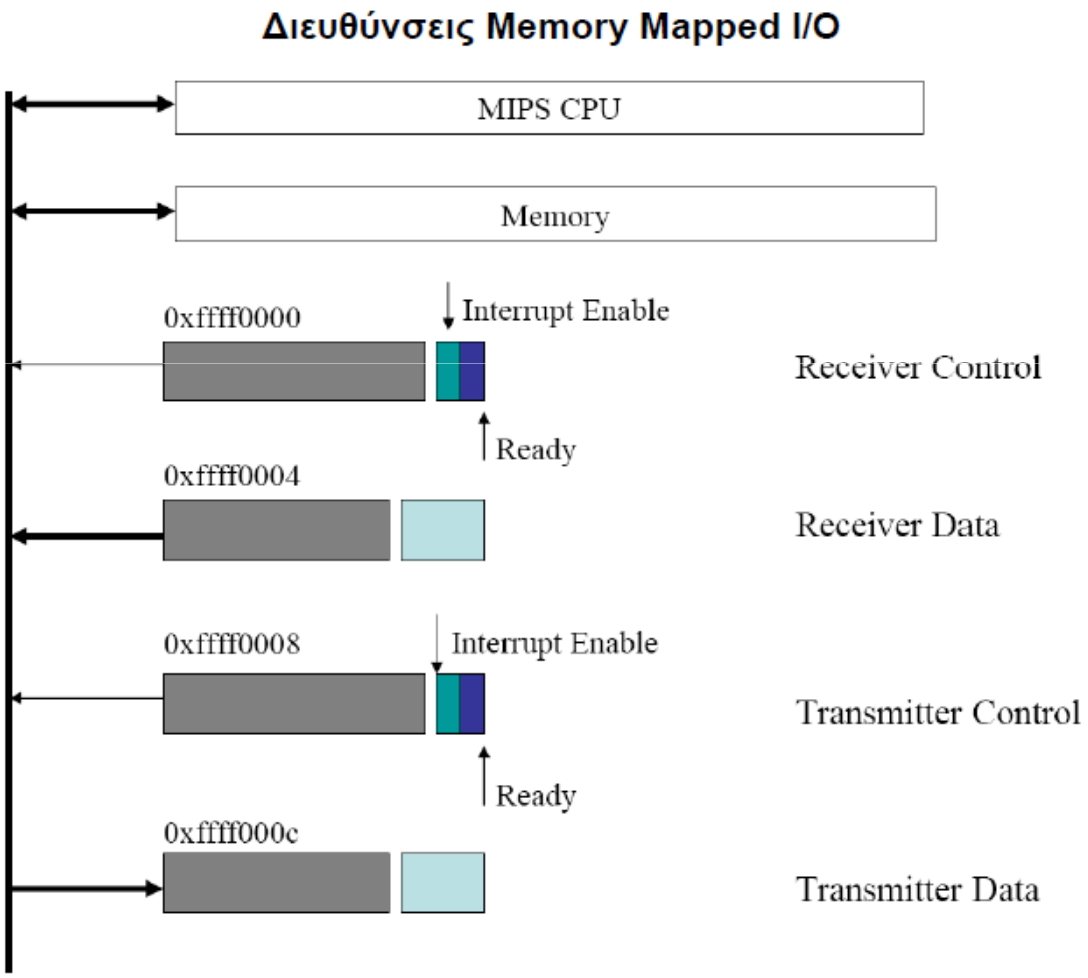
Φροντιστήριο
8/12/2014

Επικοινωνία CPU – I/O Devices (1)

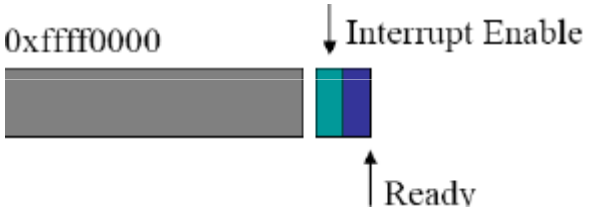
- Απλές I/O Devices
  - Πληκτρολόγιο (Receiver)
  - Κονσόλα SPIM
- 2 ειδικές θέσεις στην μνήμη (Memory Mapped I/O)
  - Data
  - Control

Memory Mapped I/O



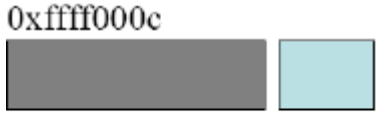
Control Register

- 32 bits width
  - Χρήσιμα: 2 least significant bits
- Ready bit
  - Είναι ενεργοποιημένο μόνο αν η συσκευή είναι έτοιμη να δεχτεί ή να διαβάσει έναν νέο χαρακτήρα
- Interrupt bit
  - Ενεργοποίηση των Interrupts για την συγκεκριμένη συσκευή



Data Register

- 32 bits width
  - Χρήσιμα: 8 least significant bits
- 8 bits
  - Ο χαρακτήρας είτε που διαβάστηκε από το πληκτρολόγιο είτε που θα εμφανιστεί στην κονσόλα



Memory Mapped I/O Addresses

- Receiver → Πληκτρολόγιο
- Transmitter → Κονσόλα

Όνομα Καταχωρητή	Διεύθυνση
Receiver Control	0xffff0000
Receiver Data	0xffff0004
Transmitter Control	0xffff0008
Transmitter Data	0xffff000c

Τεχνική Polling

- Π.χ. Να γραφτεί ένας χαρακτήρας στην κονσόλα
  - Διαβάζω τα περιεχόμενα της θέσης μνήμης 0xffff0008
  - Αν το Ready bit είναι 0 συνέχισε να διαβάζεις την παραπάνω θέση μνήμης
  - Αν το Ready bit είναι 1 αποθήκευσε τον χαρακτήρα (8 bits) στη θέση μνήμης 0xffff000c.
- Q1: Πώς κάνω ανάγνωση ενός χαρακτήρα από το πληκτρολόγιο?

1ο Μέρος εργαστηρίου (1/2)

- Το πρόγραμμα σας θα διαβάζει μία συμβολοσειρά θα την μετατρέπει σε ΚΕΦΑΛΑΙΑ και θα την εκτυπώνει στην κονσόλα.
- Οι λέξεις χωρίζονται μόνο με ΚΕΝΑ!
- Μπορείτε να χρησιμοποιήσετε τις συναρτήσεις που μετατρέπουν μία συμβολοσειρά σε ΚΕΦΑΛΑΙΑ από το 3ο Εργαστήριο.

1ο Μέρος εργαστηρίου (2/2)

- 2 συναρτήσεις
  - Print_string
    - Εμφανίζει στην κονσόλα μία ολόκληρη συμβολοσειρά που είναι αποθηκευμένη στην μνήμη.
  - Read_string
    - Διαβάζει μία ολόκληρη συμβολοσειρά και θα την αποθηκεύει στην μνήμη.
- 2 συναρτήσεις
  - Write_ch
    - Εμφανίζει έναν χαρακτήρα στην κονσόλα
  - Read_ch
    - Διαβάζει έναν χαρακτήρα από το πληκτρολόγιο
- **ΟΧΙ SYSCALL ΓΙΑ ΤΟ I/O**

1ο Μέρος εργαστηρίου: Flowchart

1) Εκτύπωσε ένα μήνυμα: “Give the string:” (Χρήση συνάρτησης Print_string)
2) Διάβασε μία συμβολοσειρά (Χρήση συνάρτησης Read_string)
3) Άλλαξε τα γράμματα σε ΚΕΦΑΛΑΙΑ
4) Εκτύπωση της νέας συμβολοσειράς (Χρήση συνάρτησης Print_string)

Συναρτήσεις Print_string και write_ch

- Print_string (String Address)
  1) Διάβασε έναν έναν τους χαρακτήρες του string μέχρι τον **'\0'**
  2) Για κάθε χαρακτήρα που διαβάζεις κάλεσε την συνάρτηση write_ch.
- Write_ch (Character)
  1) Έλεγξε αν η κονσόλα είναι έτοιμη να δεχτεί νέο χαρακτήρα (βλέπε διαφάνεια 7)
  2) Αν είναι έτοιμη, αποθήκευσε τον Character στην διεύθυνση transmitter data.

Συναρτήσεις Read_string και read_ch

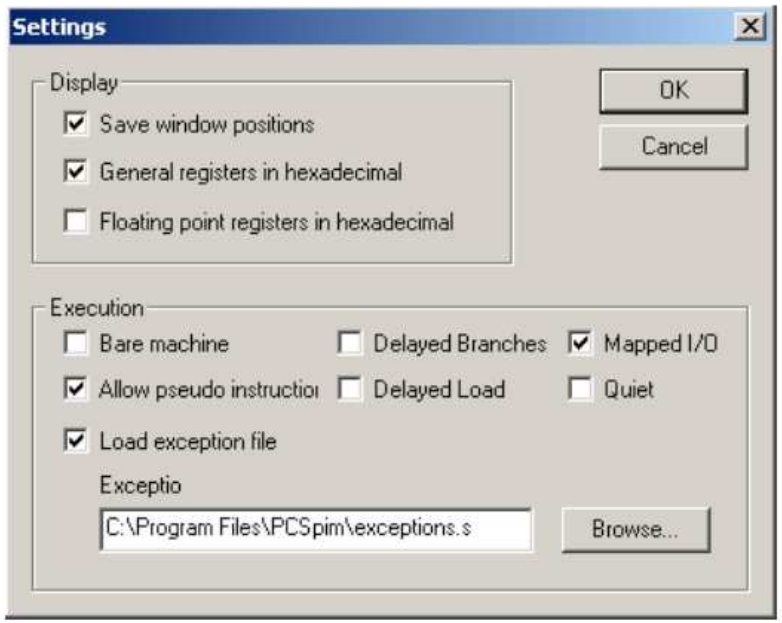
- Read_string (Address to save string)
  1) Βάλε έναν καταχωρητή να δείχνει στην πρώτη διεύθυνση που θα αποθηκευθεί το string.
  2) Διάβασε έναν χαρακτήρα με την χρήση της συνάρτησης read_ch.
  3) Αποθήκευσε τον χαρακτήρα που σου επέστρεψε η read_ch στην μνήμη στη θέση που δείχνει ο καταχωρητής του βήματος 1.
  4) **Κάλεσε την συνάρτηση write_ch με όρισμα τον παραπάνω χαρακτήρα**
  5) Αν ο χαρακτήρας που μόλις αποθήκευσες είναι ο '\0' τότε το string τελείωσε και επέστρεψε.
  6) Διαφορετικά, συνέχισε στον βήμα 2.
- Read_ch
  1) Έλεγξε αν το πληκτρολόγιο διάβασε νέο χαρακτήρα
  2) Αν διάβασε, διάβασε την θέση μνήμης Receiver Data και επέστρεψε τον χαρακτήρα

Παρατηρήσεις

Παρατήρηση 1: Πριν τρέξετε τον κώδικά σας πρέπει να ενεργοποιήσετε τα «memory mapped IO» στον SPIM από το μενού Simulator->Settings και να ενεργοποιήσετε την επιλογή «Mapped I/O» εάν δεν είναι ήδη ενεργοποιημένη.

Παρατήρηση 2: Κατά την υλοποίηση του κομματιού εισόδου (read_ch, κλπ) για λόγους debugging μπορείτε να χρησιμοποιήσετε syscall για την εκτύπωση διαγνωστικών μηνυμάτων.

Μενού ρυθμίσεων SPIM για ενεργοποίηση Memory Mapped I/O και καθορισμό αρχείου interrupt handler.



Επικοινωνία CPU – I/O Devices (2)

- Interrupts/ Exceptions
  - I/O Device στέλνει “interrupt” σήμα στον CPU
  - CPU σταματάει “άμεσα” την εκτέλεση του προγράμματος και ξεκινάει τον interrupt handler
- Interrupt Handler
  - Κώδικας που εξυπηρετεί τα interrupts
  - Οι συμβάσεις κλήσεις υπορουτίνας δεν ισχύουν
  - 2 Καταχωρητές
    - $k1, $k2

Interrupts

- Ενεργοποίηση
  - $12 Συνεπεξεργαστή
    - Διαβάζω την τιμή του $12 από τον συνεπεξεργαστή (mfc0 $t0, $12)
    - Bit 11 ➔ Διακοπές πληκτρολογίου
    - Bit 0 ➔ Ενεργοποίηση διακοπών για τον επεξεργαστή
    - Αποθήκευση νέας τιμής στον $12. (mtc0 $t0, $12).
  - Control registers
    - Bit 1 ➔ Interrupt enable

2ο Μέρος Εργαστηρίου

- Ένα απλό μενού επιλογών που θα επικοινωνεί με το πληκτρολόγιο μέσω Interrupts

Flowchart προγράμματος (1/2)

1) Δεσμεύστε 2 ποσότητες των 4 Bytes, cflag και cdata.
2) Αρχικοποιήστε την cflag περιοχή με την τιμή 0.
   - Η cflag θέση μνήμης χρησιμοποιείται για την "επικοινωνία" μεταξύ του επεξεργαστή και Interrupt Handler.
     - cflag = 0 ➔ κανένας νέος χαρακτήρας
     - cflag = 1 ➔ ένας νέος χαρακτήρας στην θέση cdata
3) Ενεργοποιήστε τα interrupts του πληκτρολογίου (Διαφάνεια 15)
4) Εμφανίστε ένα απλό μενού επιλογών όπως στο εργαστήριο 3 **με την χρήση syscall**.
   - Επιλογή 1
   - Επιλογή 2
   - Έξοδος

Flowchart προγράμματος (2/2)

5) Στην συνέχεια το πρόγραμμα σας μπαίνει σε ένα loop στο οποίο θα ελέγχει τη θέση μνήμης cflag.
6) Αν η τιμή είναι 0, ξαναδιάβασε τη θέση μνήμης cflag.
7) Διαφορετικά αν είναι 1, διάβασε την περιοχή cdata.
   - Αν η τιμή του cdata είναι 1 ή 2 (ανάλογα το τι πληκτρολόγησε ο χρήστης) θα εμφανίζεται στην κονσόλα με syscall το μήνυμα «Ενεργοποιήθηκε η επιλογή X» (όπου X είναι 1 ή 2) και συνεχίστε την εκτέλεση του κώδικα.
   - Αν η τιμή είναι του cdata είναι space, με syscall θα τερματίζεται το πρόγραμμα.
7) **ΜΗΔΕΝΙΖΩ ΤΗΝ ΘΕΣΗ ΜΝΗΜΗΣ CFLAG**
8) Ο κώδικας επιστρέφει στο loop ανάγνωσης της θέσης μνήμης cflag.

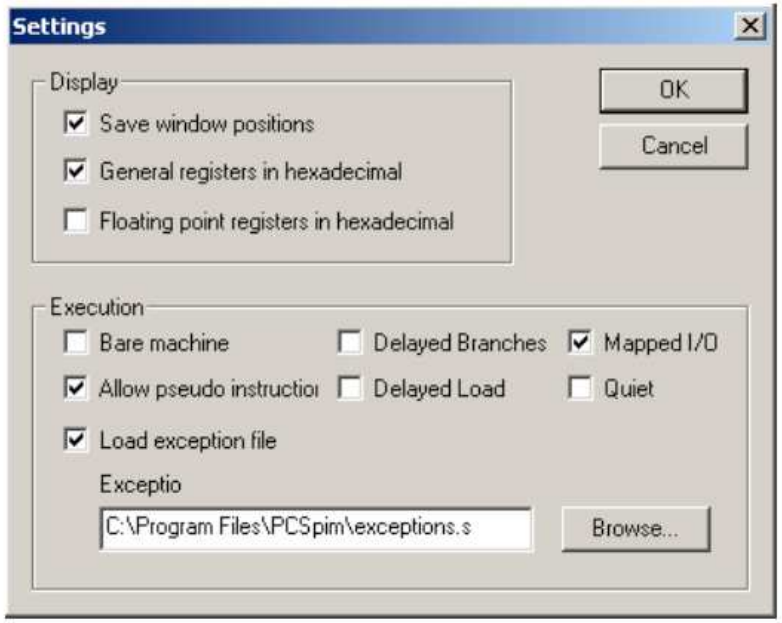
Αλλαγές Exception File (1/2)

- Αρχείο exceptions.s
  - Σημείο εγκατάστασης του SPIM, π.χ. C:\Program Files\PCSpim
  - Δημιουργία αντιγράφου και αλλαγή ονόματος
  - Εισαγωγή του κώδικα Interrupt στο σωστό σημείο
    - *#Interrupt-specific code goes here*
  - Αλλαγές στον SPIM

Αλλαγές Exception File(2/2)

- Simulator → Settings → Browse
  - Επιλογή ως αρχείο exception το νέο αρχείο που έχετε δημιουργήσει

Μενού ρυθμίσεων SPiM για ενεργοποίηση Memory Mapped I/O και καθορισμό αρχείου interrupt handler.



Κώδικας Exception File

- Εφόσον έχουμε ενεργοποιήσει τα σωστά Interrupts, ο Interrupt Handler θα εκτελείται κάθε φορά που έχουμε ένα νέο Interrupt, δηλαδή στο συγκεκριμένο εργαστήριο κάθε φορά που έχουμε ένα νέο χαρακτήρα.
- **Κώδικας exception file**
  - Ανάγνωση της διεύθυνσης Receiver Data (Διαφάνεια 3)
    - Κάθε φορά που έρχεται ένα νέο Interrupt από το πληκτρολόγιο τα δεδομένα θα μπαίνουν αμέσως στη θέση μνήμης Receiver Data
  - Αποθήκευση του χαρακτήρα που διαβάστηκε στην θέση μνήμης cdata.
  - Αποθήκευση της τιμής 1 στην θέση μνήμης cflag.