9ο Εργαστήριο Αρχιτεκτονικής Η/Υ: Είσοδος-έξοδος και εξαιρέσεις (διακοπές)

Α. Ευθυμίου

Παραδοτέο: Τρίτη 17 Μάη 2016, 23:00

Ο σκοπός αυτής της άσκησης είναι η εμβάθυνση της κατανόησης λειτουργίας του συστήματος εισόδου-εξόδου ενός υπολογιστή και των διακοπών (interrupts).

Οι απαντήσεις στις ερωτήσεις που θα δείτε στο κείμενο δεν χρειάζεται να δοθούν. Είναι για να βοηθήσουν τη δική σας κατανόηση.

Θα πρέπει να έχετε μελετήσει τα μαθήματα για το σύστημα εισόδου-εξόδου και τις εξαιρέσεις-διακοπές που αντιστοιχούν στις ενότητες 6.1-6.6, 4.9, και 5.4 (υλοποίηση προστασίας με εικονική μνήμη μέχρι το τέλος του τμήματος 5.4) του βιβλίου.

Μή ξεχάσετε να επιστρέψετε τα παραδοτέα που αναφέρονται στο τέλος του κειμένου για να πάρετε βαθμό γι'αυτή την εργαστηριακή άσκηση!

1 Προετοιμασία

Για να ξεκινήσετε θα χρειαστείτε όλα τα αρχεία του εργαστηρίου δίνοντας τις εξής εντολές:

```
git remote add lab09_starter https://github.com/UoI-CSE-MYY402/lab09_starter.git
git fetch lab09_starter
git merge lab09_starter/master -m "Fetched lab09 starter files"
```

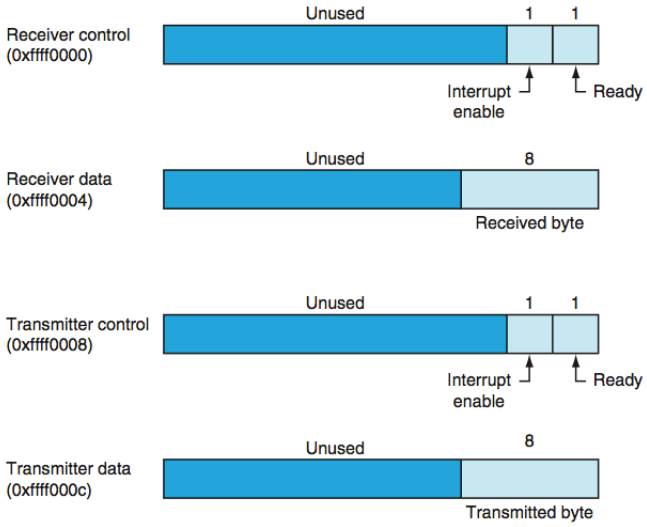
Θα χρησιμοποιήσετε τον MARS και το "Keyboard and Display MMIO Simulator" που βρίσκεται κάτω από το μενού Tools. Το εργαλείο αυτό προσομοιώνει ένα τερματικό: είσοδο από το πληκτρολόγιο, στο κάτω μέρος του παραθύρου, και έξοδο προς οθόνη, στο πάνω μέρος. Μπορείτε να σκεφτείτε το πληκτρολόγιο ως δέκτη που δέχεται είσοδο προς τον υπολογιστή όταν ο χρήστης πατήσει ένα πλήκτρο και την οθόνη ως εκπομπό που εκπέμπει έξοδο από τον υπολογιστή (χαρακτήρες) και την εμφανίζει στην οθόνη. Τα δύο τμήματα λειτουργούν ξεχωριστά και ανεξάρτητα: οι χαρακτήρες που πληκτρολογούνται δεν εμφανίζονται (αντηχούνται - echo) αυτόματα στην οθόνη. Για να γίνει η αντήχηση, θα πρέπει να υπάρχει ένα κατάλληλο πρόγραμμα που να την κάνει. Σε αυτή την άσκηση θα εξετάσετε μερικούς διαφορετικούς τρόπους υλοποίησης της αντήχησης.

2 Περιόδευση - Polling

Ο απλούστερος τρόπος χειρισμού εισόδου-εξόδου από υπολογιστή είναι η περιόδευση (polling). Ένα περιφεριακό διαθέτει δύο καταχωρητές προσπελάσιμους μέσω memory-mapped I/O: έναν καταχωρητή κατάστασης/ελέγχου (Status/Control) και έναν καταχωρητή δεδομένων είτε εισόδου, είτε εξόδου ανάλογα με το είδος του περιφεριακού. Ο επεξεργαστής παρακολουθεί την κατάσταση του περιφερειακού και, όταν αυτό είναι έτοιμο, διαβάζει ή γράφει τα δεδομένα.

Για το Keyboard and Display MMIO Simulator (για συντομία MMIO) του MARS, οι διευθύνσεις στις οποίες αντιστοιχούν οι καταχωρητές και τα πεδία τους φαίνονται στο σχήμα 1. Τα ζεύγη των καταχωρητών είναι ξεχωριστά για το πληκτρολόγιο (Receiver - δέκτης) και για την οθόνη (Transmitter - εκπομπος).

Το πεδίο (ενός bit) ready του καταχωρητή ελέγχου δέκτη (Receiver control) είναι μόνο για ανάγνωση και οι αλλαγές σε αυτό αγνοούνται. Το ready αλλάζει από 0 σε 1 όταν πληκτρολογείται ένας χαρακτήρας και από 1 σε 0 όταν ο χαρακτήρας αυτός διαβάζεται μέσω του καταχωρητή receiver data. Το πεδίο (ενός bit) Interrupt enable καθορίζει αν το πληκτρολόγιο θα προκαλεί διακοπές (τιμή 1) ή όχι. Αυτό το bit μπορεί να γραφτεί ή να διαβαστεί από ένα πρόγραμμα και αρχικά είναι 0. Αν το Interrupt enable έχει την τιμή 1, όταν πληκτρολογείται ένας χαρακτήρας (το ready bit γίνεται 1), προκαλείται διακοπή-εξαίρεση στον επεξεργαστή.



Σχήμα 1: Καταχωρητές ελέγχου και δεδομένων.

Ο καταχωρητής Receiver data περιέχει τον κωδικό ASCII του χαρακτήρα που έχει πληκτρολογηθεί όταν το ready bit του Receiver Control είναι 1. Αν το ready bit είναι 0, η τιμή αυτού του καταχωρητή είναι απροσδιόριστη (undefined, unspecified): δεν μπορούμε να υποθέσουμε κάτι γι'αυτήν. Η συσκευή μπορεί να καταλάβει πότε διαβάζεται αυτός ο καταχωρητής και τότε αλλάζει το ready bit σε 0. Γι'αυτό η «μνήμη» που αντιστοιχεί σε περιφερειακά λέγεται ότι έχει «παρενέργειες» (side-effects), σε αντίθεση με την κοινή μνήμη όπου δεν συμβαίνουν τέτοια φαινόμενα.

Για την «οθόνη»-πομπό, ο καταχωρητής ελέγχου (Transmitter control) έχει την ίδια δομή με αυτή του δέκτη. Το Ready bit δείχνει πότε είναι έτοιμος ο πομπός για να δεχθεί τον επόμενο χαρακτήρα, γιατί χρειάζεται κάποιος χρόνος μέχρι το περιφερειακό (οθόνη) να εμφανίσει τον χαρακτήρα και μέχρι να τελειώσει δεν μπορεί να δεχθεί τον επόμενο. Στο MMIO αυτή η καθυστέρηση καθορίζεται από τις επιλογές κάτω από τον χώρο εμφάνισης χαρακτήρων (check-box DAD κλπ). Υπάρχουν διάφορες επιλογές που θα εξετάσουμε αργότερα αλλά επειδή δεν υπάρχει η έννοια του χρόνου στον Mars, η καθυστέρηση ορίζεται ως αριθμός εντολών που εκτελούνται και όχι ως κύκλοι ρολογιού ή δευτερόλεπτα.

Περισσότερες λεπτομέρειες για τους καταχωρητές αυτούς μπορείτε να βρείτε στο Help του εργαλείου και στην ενότητα B.8 του συγγράμματος. Το σύγγραμμα αναφέρεται σε έναν άλλο προσομοιωτή, που λέγεται SPIM, αλλά ο MARS είναι συμβατός με τον SPIM.

Με τη μέθοδο polling ο επεξεργαστής παρατηρεί συνεχώς το ready bit του καταχωρητή ελέγχου και όταν το περιφερειακό είναι έτοιμο, διαβάζει ή γράφει τον αντίστοιχο καταχωρητή δεδομένων. Ο σκελετός κώδικα για ανάγνωση ενός χαρακτήρα από το πληκτρολόγιο είναι ο εξής:

```
    lui  $t0,0xffff #ffff0000
waitloop:
    lw   $t1, 0($t0)      # receiver control
    andi $t1, $t1,   0x0001
    beq  $t1, $zero, waitloop
    # Receiver is ready: get the char
    lw   $v0, 4($t0)      # receiver data
```

Φορτώστε στον MARS το αρχείο singleEcho.asm που βρίσκεται στο lab09_starter. Το πρόγραμμα που περιέχει διαβάζει έναν χαρακτήρα από το πληκτρολόγιο, με την υπορουτίνα getc και μετά τον εμφανίζει στην οθόνη με την υπορουτίνα putc. Παρατηρήστε τον κώδικα για να τον καταλάβετε και μετά

ξεκινήστε το MMIO και συνδέστε το εργαλείο στον Mars πατώντας το κουμπί "Connect to MIPS". Στο κύριο παράθυρο του MARS επιλέξτε το 0xFFFF0000 (MMIO) για να το βλέπετε στο data segment. Παρατηρήστε τις 4 πρώτες λέξεις στην παραπάνω διεύθυνση που είναι οι 4 καταχωρητές που αναφέρθηκαν παραπάνω.

Υπάρχει κάποιο ready bit που είναι ήδη 1; Σε τί αντιστοιχεί, την είσοδο από το πληκτρολόγιο ή την έξοδο;

Εκτελέστε τον κώδικα εντολή προς εντολή και παρατηρήστε πως ο επεξεργαστής περιμένει περιοδεύοντας (polling) το ready bit του δέκτη (receiver control register). Κάντε κλικ στο κάτω τμήμα του MMIO και πατήστε ένα πλήκτρο. Παρατηρείστε ότι το Ready bit του Receiver γίνεται 1 και στον καταχωρητή δεδομένων εμφανίζεται ο κωδικός ASCII του χαρακτήρα. Συνεχίζοντας την εκτέλεση, πάντα εντολή προς εντολή, ο επεξεργαστής βγαίνει από την επανάληψη αναμονής και διαβάζει τον χαρακτήρα. Παρατηρείστε ότι το Ready bit γίνεται 0 μετά την ανάγνωση.

Τώρα συνεχίζοντας την εκτέλεση θα φτάσετε στην υπορουτίνα που εμφανίζει τον χαρακτήρα στην «οθόνη», το επάνω τμήμα του MMIO. Σε αυτή την περίπτωση το ready bit είναι ήδη 1, επομένως δεν θα γίνουν επαναλήψεις. Παρατηρείστε όμως την αλλαγή τιμής στο ready bit μετά την αποθήκευση του χαρακτήρα στον καταχωρητή δεδομένων. Θα πρέπει να γίνει 0 και αν υπήρχαν ακόμη αρκετές εντολές για να εκτελεστούν, θα έμενε 0 μέχρι τις επόμενες 500 εντολές (αν δεν έχετε αλλάξει το DAD και τα υπόλοιπα στα δεξιά του).

3 Παραδοτέο 1: Double echo

Τώρα είναι η σειρά σας να αλλάξετε κάπως τον προηγούμενο κώδικα και να εμφανίζετε 2 φόρες συνεχόμενα κάθε χαρακτήρα που πληκτρολογεί ο χρήστης. Στη main σας θα έχετε μια επανάληψη που δεν τελειώνει ποτέ και θα διαβάζετε ένα χαρακτήρα με την getc και θα τον εμφανίζετε δύο φορές με την putc.

Σας δίνεται ο αρχικός σκελετός στο αρχείο double_echo.asm, ένα από τα αρχεία που πήρατε από το GitHub.

Όταν το τελειώσετε, βάλτε το run speed slider του κύριου παραθύρου του MARS στη μέγιστη ταχύτητα και την καθυστέρηση (Delay length) του MMIO στις 100 εντολές, και τρέξτε το πατώντας το F5 (Run->Go). Συχνά πρέπει να μεγαλώσετε το παράθυρο του MMIO για να μπορείτε να βάλετε ό,τι τιμές θέλετε στο πεδίο αυτό. Αφήστε το check box DAD (display after delay) άδειο, γιατί φαίνεται να υπάρχουν προβλήματα όταν είναι επιλεγμένο. Πληκτρολογείστε μερικούς χαρακτήρες και δείτε ότι εμφανίζονται 2 ίδιοι στο επάνω τμήμα του MMIO με κάποια μικρή καθυστέρηση.

Προσοχή μερικές φορές το MMIO κολλάει και θέλει reset. Σπάνια κολλάει και ο ίδιος ο MARS και χρειάζεται να τερματιστεί και να ξαναξεκινήσει.

Δοκιμάστε ξανά με χαμηλότερη ταχύτητα από τον slider του κύριου παραθύρου του MARS (π.χ. 10 εντολές το δευτερόλεπτο) για να βλέπετε σε ποιά εντολή βρίσκεται κάθε στιγμή. Τώρα θα φαίνεται καθαρότερα η καθυστέρηση στην εμφάνιση του 2ου χαρακτήρα.

Σε αυτές τις χαμηλότερες ταχύτητες εκτέλεσης, δοκιμάστε να πληκτρολογήσετε σχετικά γρήγορα 2-3 χαρακτήρες. Ποιός χαρακτήρας εμφανίζεται; Γιατί συμβαίνει αυτό;

4 Είδοδος-έξοδος με διακοπές

Από τα προηγούμενα σίγουρα θα έχετε συμπεράνει ότι η περιόδευση είναι σπατάλη χρόνου για τον επεξεργαστή. Η εναλακτική λύση είναι είσοδος-έξοδος κατευθυνόμενη από διακοπές (interrupt-driven I/O). Όταν το περιφερειακό είναι έτοιμο, διακόπτει τον επεξεργαστή και τότε τρέχει η υπορουτίνα που διαβάζει ή γράφει τα δεδομένα από/προς το περιφερειακό.

Γνωρίζουμε όμως ότι οι διακοπές (εξαιρέσεις) στον MIPS τρέχουν κώδικα του λειτουργικού συστήματος (ΛΣ) από τη διεύθυνση 0x80000180. Επιπλέον τα προγράμματα των απλών χρηστών κανονικά δεν επιτρέπεται να προσπελαύνουν το τμήμα της μνήμης που αντιστοιχεί σε περιφερειακά (memory mapped

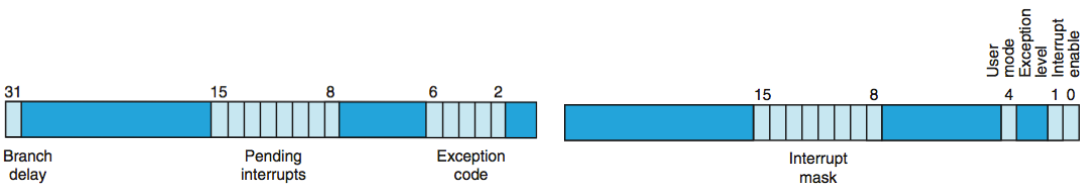
I/O). Έτσι στα επόμενα προγράμματα θα υπάρχει ένα μικρό μέρος κώδικα χρήστη και ο χειρισμός των περιφερειακών θα γίνεται από κώδικα του ΛΣ.

Στους επεξεργαστές MIPS ένα τμήμα της κεντρικής μονάδας επεξεργασίας, που ονομάζεται συνεπεξεργαστής (coprocessor) 0, κρατά όλη την πληροφορία που χρειάζεται το λογισμικό για να χειριστεί εξαιρέσεις. Για τους σκοπούς της άσκησης θα χρησιμοποιηθεί ένα μέρος της πληροφορίας (προσοχή οι καταχωρητές στην Ελληνική έκδοση του βιβλίου ονομάζονται ανάποδα στα σχήματα B.7.1-2):

- Ο καταχωρητής EPC, exception PC (αριθμός καταχωρητή 14), κρατά την διεύθυνση της εντολής που εκτελούσε ο επεξεργστής όταν έγινε η εξαίρεση. Αν η εξαίρεση έγινε από εξωτερική διακοπή, όπως το MMIO, η εντολή που βρίσκεται στη διεύθυνση του EPC **δεν** έχει εκτελεστεί. Αυτός ο καταχωρητής χρησιμοποιείται ώς διεύθυνση επιστροφής όταν τελειώσει το ΛΣ με τον χειρισμό της διακοπής.
- Ο καταχωρητής αιτίας διακοπής, Cause (αριθμός καταχωρητή 13), κρατά πληροφορίες για το είδος εξαίρεσης που συνέβει και για αυτές που εκκρεμούν. Το σχήμα 2α΄ δείχνει τα πεδία του καταχωρητή. Το πεδίο Exception code είναι ένας αριθμός που κωδικοποιεί την αιτία της διακοπής. Η τιμή 0 σημαίνει εξωτερική διακοπή υλικού και χρησιμοποιείται από το MMIO. Η τιμή 13 σημαίνει εξαίρεση παγίδας (trap), που θα αναφερθεί παρακάτω. Κάθε bit του πεδίου Pending interrupts, χρησιμοποιείται για να δείξει ποιές εξαιρέσεις εκκρεμούν. Το bit 8, όταν έχει την τιμή 1, σημαίνει ότι το πληκτρολόγιο-δέκτης του MMIO έχει προκαλέσει διακοπή. Το bit 9 χρησιμοποιείται από την οθόνη-πομπό του MMIO.

  Αν και τα διάφορα πεδία του καταχωρητή παίρνουν αυτόματα τιμές με το που γίνει μια εξαίρεση, όταν τελειώσει ο χειρισμός της εξαίρεσης από το ΛΣ, τα πεδία πρέπει να «καθαριστούν» (clear-reset) από το λογισμικό, σε τιμές που δείχνουν ότι δεν υπάρχει πλεόν λόγος εξαίρεσης (γενικά το 0).
- Ο καταχωρητής κατάστασης συστήματος, Status (αριθμός καταχωρητή 12), σχήμα 2β΄, κρατά πληροφορίες για την τρέχουσα κατάσταση του επεξεργαστή. Το πεδίο Interrupt enable, στο bit 0, δηλώνει αν ο επεξεργαστής δέχεται εξαιρέσεις. Αν η τιμή του είναι 0, ο επεξεργαστής δεν διακόπτεται. Αυτό γίνεται για σύντομα χρονικά διαστήματα όταν τρέχει κώδικας που αν διακοπεί, θα υπάρξει σοβαρό πρόβλημα. Το πεδίο Exception level (EXL), στο bit 1, δείχνει αν ο επεξεργαστής έχει διακοπεί. Όταν γίνει μια εξαίρεση, παίρνει την τιμή 1 και απαγορεύονται-αγνοούνται άλλες εξαιρέσεις μέχρι να γίνει ξανά 0. Το πεδίο Interrupt mask μπορεί να απαγορεύσει συγκεκριμένα είδη εξαιρέσεων: κάθε bit αντιστοιχεί σε κάποια εξαίρεση. Το πεδίο αυτό δεν χρησιμοποιείται στην άσκηση. Όπως και στον Cause, το πεδίο EXL παίρνει αυτόματα την τιμή 1 όταν γίνει μια διακοπή, αλλά πρέπει να καθαριστεί στο 0 από το λογισμικό στο τέλος της εκτέλεσης της υπορουτίνας χειρισμού εξαιρέσεων.

Φορτώστε στον MARS το αρχείο intEcho.asm που πήρατε από το GitHub. Παρατηρείστε ότι η πρώτη εντολή της main, όπου βρίσκεται ο κώδικας χρήστη, είναι μια άγνωστη εντολή, `teqi $zero, 0`. Η εντολή αυτή είναι παρόμοια με την syscall που έχετε ήδη δει. Αν ο καταχωρητής πηγής (`$zero`) είναι ίσος με τη



(α΄) Καταχωρητής αιτίας, Cause (β΄) Καταχωρητής κατάστασης, Status

Σχήμα 2: Καταχωρητές εξαιρέσεων

σταθερά, προκαλείται μια εξαίρεση που λέγεται παγίδα (trap). Έτσι τρέχει ο χειριστής εξαιρέσεων του ΛΣ στη διεύθυνση 0x80000180. Με αυτό τον τρόπο ζητάμε από το ΛΣ να κάνει κάτι που απαγορεύεται σε έναν απλό χρήστη να το κάνει απευθείας, όπως ακριβώς με τις εντολές syscall. Ο υπόλοιπος κώδικας χρήστη, μετά την teqi, είναι μια επανάληψη δίχως τέλος, γιατί θέλουμε να παρατηρήσουμε τις διακοπές που θα προκαλούνται κάθε φορά που θα δίνεται ένας χαρακτήρας στο πληκτρολόγιο του MMIO.

Διαβάστε τον κώδικα χειρισμού εξαιρέσων (exception handler) που βρίσκεται μετά την οδηγία ktext 0x80000180. Αυτή είναι μια ειδική οδηγία του assembler που δηλώνει ότι το παρακάτω είναι κώδικας του πυρήνα του ΛΣ και πρέπει να τοποθετηθεί στη συγκεκριμένη διεύθυνση. Επομένως ο υπόλοιπος κώδικας του αρχείου intEcho.asm είναι κώδικας ΛΣ.

Στην αρχή απαγορεύονται οι εξαιρέσεις (καθαρισμός του interrupt enable του Status) και αποθηκεύονται στη στοίβα όλοι οι καταχωρητές που χρησιμοποιούνται από τον χειριστή εξαιρέσεων. Παρατηρείστε ότι ακόμη και καταχωρητές $t αποθηκεύονται, γιατί μια διακοπή μπορεί να συμβεί οποιαδήποτε στιγμή, και μετά την εξυπηρέτησή της, το πρόγραμμα που είχε διακοπεί μπορεί να συνεχίσει την εκτέλεσή του. Επομένως θα πρέπει να έχει όλες τις προηγούμενες τιμές καταχωρητών κατά τη συνέχιση της εκτέλεσης.

Μετά εξετάζεται ο λόγος για τον οποίο έγινε η εξαίρεση. Αυτός κωδικοποιείται στο πεδίο Exception Code του καταχωρητή Cause (13) που βρίσκεται στον συνεπεξεργαστή 0. Επειδή δεν μπορούν να γίνουν απευθείας πράξεις με καταχωρητές του συνεπεξεργαστή 0, το πρόγραμμα έχει εντολές mfc0 (move from corpocessor 0) και mtc0 (move to coprocessor 0) που μεταφέρουν τιμές μεταξύ ενός καταχωρητή του MIPS και ενός του συνεπεξεργαστή. Ενας πλήρης χειριστής εξαιρέσεων θα εξέταζε και θα χειριζόταν όλα τα δυνατά είδη εξαιρέσεων. Το συγκεκριμένο πρόγραμμα εξετάζει μόνο δύο περιπτώσεις: αν είναι εξαίρεση-παγίδα (αυτή που προκαλεί στην αρχή το πρόγραμμα χρήστη) ή αν είναι εξωτερική διακοπή (από το MMIO).

Στην πρώτη περίπτωση καλείται η υπορουτίνα enable_term_int, που θέτει τον καταχωρητή ελέγχου του receiver (πληκτρολόγιο) ώστε να προκαλεί διακοπές κάθε φορά που δίνεται ένας χαρακτήρας από το πληκτρολόγιο. Πίσω στον χειριστή διακοπών, αλλάζει τη διεύθυνση της εντολής που θα εκτελεστεί μετά την επιστροφή από την εξαίρεση (EPC, 14) ώστε να μην είναι πάλι η teqi, αλλά η επόμενη εντολή. Αυτό συμβαίνει γιατί, στη γενική περίπτωση, η εντολή που διακόπηκε ξαναεκτελείται, έτσι η αρχική τιμή του EPC είναι η διεύθυνση της teqi.

Αν η εξαίρεση προκλήθηκε από εξωτερική διακοπή, ο κώδικας ελέγχει το ready bit του καταχωρητή ελέγχου του receiver και αν είναι 1, διαβάζει την τιμή του χαρακτήρα που πληκτρολογήθηκε. Μετά, αν και ο trasnmitter είναι ελεύθερος (το αντίστοιχο ready bit είναι 1), γράφει το χαρακτήρα στον καταχωρητή δεδομένων του, ώστε να αντηχήσει στην οθόνη. Αν ο transmitter δεν είναι διαθέσιμος, δεν γίνεται αντήχηση και ο χαρακτήρας απορρίπτεται (discard).

Αφού καταλάβετε τον κώδικα, ξεκινήστε το MMIO και συνδέστε το με τον MIPS. Απενεργοποιείστε το DAD και δώστε μια μικρή τιμή στο Delay length. Εκτελέστε εντολή προς εντολή το πρόγραμμα ώστε να δείτε την εξαίρεση που προκαλεί η teqi και το χειρισμό της: τις αλλαγές στους καταχωρητές του συνεπεξεργαστή και στους καταχωρητές του MMIO βάζοντας την κατάλληλη επιλογή στο data segment (από τη διεύθυνση 0xffff0000). Όταν η εκτέλεση επιστρέψει στην επανάληψη στον κώδικα χρήστη, βάλτε ένα breakpoint στην αρχή του χειριστή εξαιρέσεων και αφήστε την εκτέλεση να προχωρήσει μόνη της (F5). Δώστε ένα χαρακτήρα στο MMIO και δείτε ότι αμέσως η εκτέλεση πηγαίνει στον χειριστή εξαιρέσεων και ενεργοποιείται το breakpoint. Εκτελέστε ξανά εντολή προς εντολή για να δείτε τι συμβαίνει τώρα που η εξαίρεση προέρχεται από το MMIO.

Τέλος, βγάλτε το breakpoint, αυξήστε κάπως το Delay length και πληκτρολογώντας γρήγορα προσπαθήστε να δείτε αν μπορείτε να προκαλέσετε την αποτυχία αντήχησης κάποιου χαρακτήρα, επειδή ο transmitter δεν είναι έτοιμος όταν έρθει ένας χαρακτήρας από το πληκτρολόγιο.

5 Παραδοτέο 2: buffered input-output

Τώρα που είδατε πως λειτουργεί η είσοδος-έξοδος κατευθυνόμενη από διακοπές, μπορείτε να τη κάνετε να δουλεύει με καλύτερο τρόπο. Συγκεκριμένα θέλουμε να αποφύγουμε την αποτυχία αντήχησης που

μπορεί να συμβεί αν, μερικές φορές, ο Transmitter αργήσει λίγο παραπάνω να αποκριθεί. Για το σκοπό αυτό προσθέτουμε έναν χώρο προσωρινής αποθήκευσης (buffer) που είναι οργανωμένος ως κυκλική ουρά με χώρο αποθήκευσης 16 χαρακτήρων. Όταν ένας χαρακτήρας πληκτρολογείται, ο χειριστής εξαιρέσεων τον διαβάζει και τον βάζει στο τέλος της κυκλικής ουράς. Αν ο transmitter είναι έτοιμος, θα εμφανίζει τον επόμενο χαρακτήρα από την αρχή της ουράς. Αν δεν είναι έτοιμος εκείνη τη στιγμή, θα προκληθεί διακοπή όταν ετοιμαστεί και ο χειριστής εξαιρέσεων θα εμφανίσει τον επόμενο χαρακτήρα.

Ένα σημαντικό μέρος του κώδικα έχει ήδη υλοποιηθεί στο αρχείο intBuffer.asm. Λείπει η υπορουτίνα putChar που βρίσκει τον χαρακτήρα στην αρχή της ουράς και τον εμφανίζει στην «οθόνη» του MMIO, γράφοντάς τον στον καταχωρητή δεδομένων του transmitter του MMIO. Για το χειρισμό της κυκλικής ουράς, αντιγράψτε και μετατρέψτε τον αντίστοιχο κώδικα της getChar.

Τώρα για να καταφέρετε να το κάνετε να αποτύχει στην αντήχηση χαρακτήρων με την ταχύτητα εκτέλεσης του MARS στο μέγιστο, θα πρέπει να αυξήσετε το Delay length σε λίγες χιλιάδες και να πληκτρολογήσετε γρήγορα πάνω από 16 χαρακτήρες!

6 Παραδοτέα

Τα παραδοτέα της άσκησης είναι τα αρχεία `double_echo.asm` και `intBuffer.asm`. Οι απαντήσεις στις ερωτήσεις του κειμένου δεν χρειάζεται να δοθούν. Είναι για να βοηθήσουν τη δική σας κατανόηση.

Η παράδοση θα γίνει μέσω GitHub, όπως πάντα.