ΕΡΓΑΣΤΗΡΙΟ ΨΗΦΙΑΚΗΣ ΕΠΕΞΕΡΓΑΣΙΑΣ ΣΗΜΑΤΩΝ

## Εργαστήριο 3

## Αιτήσεις διακοπής

|  |
| --- |
| Ομάδα 17 12-10-2018 |
| Ασημακόπουλος Κωνσταντίνος 1046966 |
| Λουκαρέας Παύλος 1046970 |

**Περίληψη**

Σε αυτή την εργαστηριακή άσκηση εξοικειωθήκαμε με τον μηχανισμό των διακοπών, καθώς και στη χρήση μηχανισμών της ΚΜΕ όπως οι χρονομετρητές. Πέραν των παραπάνω, εξοικειωθήκαμε με ορισμένα περιφερειακά που βρίσκονται πάνω στο DSK, τα οποία χειριστήκαμε με κατάλληλο τρόπο στον κώδικα. Οι διακοπές είναι πολύ σημαντικές καθώς επιτρέπουν στην ΚΜΕ να εξυπηρετήσει σπάνια ή εξαιρετικής σημασίας γεγονότα. Αυτά μπορεί να προέρχονται είτε από εσωτερικές είτε από εξωτερικές λειτουργικές μονάδες

**Άσκηση 3.1**

Στο ερώτημα αυτό μας ζητείται να φορτωθούν κάποια συγκεκριμένα interrupts σε κάποιες συγκεκριμένες θέσεις του IML και να ορίσουμε τις θέσεις INT10 και INT15 του IMH με όποιες διακοπές θέλουμε. Το πρόγραμμα είναι το παρακάτω:

IMH\_pointer .set 0x019c0000

IML\_pointer .set 0x019c0004

.def entry

.text

entry: MVKL .S1 IML\_pointer, A3

MVKH .S1 IML\_pointer, A3

MVKL .S1 IMH\_pointer, A2

MVKH .S1 IMH\_pointer, A2

MVK .S1 0X3DCD , A0

MVK .S1 0X3041 , A1

STW .D1 A1 , \*A3++[1]

STW .D1 A0 , \*A3

IDLE

.end

**Άσκηση 3.2**

Σε αυτό το ερώτημα μας ζητείται να δημιουργήσουμε ένα αρχείο διανυσμάτων διακοπών. Όταν συμβεί μια αίτηση διακοπής θα πρέπει να παραπέμψουμε την ΚΜΕ σε ένα συγκεκριμένο σημείο του κύριου προγράμματος. Για να δηλώσουμε το σημείο αυτό χρησιμοποιούμε το αρχείο vectors.asm όπου εκεί θα πρέπει μέσα σε οκτώ εντολές να κάνουμε την διακλάδωση εκεί που επιθυμούμε.

.title "vectors.asm"

.ref entry

.ref INT4\_R

.sect "vectors"

rst: MVKL .S2 entry, B0

MVKH .S2 entry, B0

B .S2 B0

nop

nop

nop

nop

nop

ΝΜΙ\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

INT2\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

INT3\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

TINT0\_ISR: MVKL .S2 INT4\_R, B0

MVKH .S2 INT4\_R, B0

B .S2 B0

nop

nop

nop

nop

nop

TINT1\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

SDINT\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

GPINT4\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

GPINT5\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

GPINT6\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

GPINT7\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

EDMAINT\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

EMUDTDMA\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

EMURTDXRX\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

EMURTDXTX\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

XINT0\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

RINT0\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

XINT1\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

RINT1\_ISR: nop

nop

nop

nop

nop

nop

nop

nop

**Άσκηση 3.3**

Σε αυτή την άσκηση μας ζητείται να γράψουμε ένα πρόγραμμα που να ενεργοποιεί το INT4. Για να χρησιμοποιήσουμε το INT4 πρέπει να είναι ενεργοποιημένα το ΝΜΙ και το GIE flag. Χρησιμοποιώντας την εντολή MVC μπορούμε να μεταφέρουμε τιμές σε αυτούς τους καταχωρητές ελέγχου. Ο κώδικας που ενεργοποιεί το ΙΝΤ4 είναι ο παρακάτω:

.def entry

.text

entry: MVKL .S2 0X13, B0 ;First bit of NMI and INT4

MVC .S2 B0, IER

MVKL .S2 1, B1 ;First bit of GIE

MVC .S2 B1, CSR

NOP

IDLE

.end

**Άσκηση 3.4**

Σε αυτό το ερώτημα μας ζητήθηκε να επαναφέρουμε όλες τις εκκρεμείς διακοπές της ΚΜΕ. Η λειτουργία αυτή μπορεί να γίνει μέσω του καταχωρητή ICR ο οποίος στον επόμενο κύκλο μετά από μια διακοπή αλλάζει τα bits του IFR. Η ρουτίνα μας είναι η παρακάτω:

.def entry

.text

entry: MVKL .S2 0xFFFF, B2

MVKH .S2 0xFFFF, B2

MVC .S2 B2 , ICR ;Set 1 for every interrupt

IDLE

.end

**Άσκηση 3.5**

Σε αυτό το ερώτημα μας ζητείται να γράψουμε ένα πρόγραμμα που να προγραμματίζει το χρονόμετρο 0 σύμφωνα με τις ιδιότητες που μας δίνονται. Το πρόγραμμα υλοποιήθηκε ως εξής:

.def entry

.def INT4\_R

.text

IMH\_pointer .set 0x019c0000

IML\_pointer .set 0x019c0004

clock\_period .set 0x03567E00 ;56 million cycles for 1sec

timer0\_cntr .set 0x01940000 ;Timer0 control register address

timer0\_per .set 0x01940004 ;Timer0 period address

timer0\_init .set 0x2C1 ;Timer0 initialization values

entry: MVKL .S1 IML\_pointer, A3

MVKH .S1 IML\_pointer, A3

MVK .S1 0X1 , A1

STH .D1 A1 , \*A3

MVKL .S2 0X13 , B0 ;First bit of NMI and INT4

MVC .S2 B0 , IER

MVKL .S2 1 , B1 ;First bit of GIE

MVC .S2 B1 , CSR

NOP

MVKL timer0\_cntr ,B1 ;Timer0 to B1

MVKH timer0\_cntr ,B1

MVKL timer0\_per ,B2 ;Timer0 period to B2

MVKH timer0\_per ,B2

MVKL 0x90080000 ,B10 ;LED to B10

MVKH 0x90080000 ,B10

MVKL clock\_period ,B3 ;Clock period to B3

MVKH clock\_period ,B3

MVK 0x1 ,B11 ;XOR to make B11 = 1

MV B11 ,B12 ;B12 = 1

STB B12 ,\*B10

ZERO B0

SET B0, 8, 9, B0 ;set CLKSRC & C/P field of control Timer0

STW B3, \*B2 ;save period to period address

MVK timer0\_init,B0

STW B0,\*B1 ;save timer0 init values to timer control ;register

loop: B loop

nop 5

INT4\_R: XOR B12,B11,B12

STB B12,\*B10

B IRP

nop 5

Η βασική λογική του κώδικα είναι ότι αφού έχει φορτώσει τις διευθύνσεις που απαιτούνται, ενεργοποιούμε το ΙΝΤ4 σύμφωνα με τον κώδικα του ερωτήματος 3 και ορίζουμε την περίοδο του χρονομέτρου σύμφωνα με την εκφώνηση. Έπειτα, αφού φορτώσουμε την περίοδο του ρολογιού, αρχικοποιούμε το χρονόμετρο και μπαίνουμε σε μια άπειρη επανάληψη περιμένοντας την αίτηση διακοπής. Μόλις το χρονόμετρο φτάσει την περίοδο που έχουμε ορίσει γίνεται ένα αίτημα διακοπής ΤΙΝΤ0 και οι εντολές στο αρχείο vectors.asm που έχουμε γράψει στο προηγούμενο ερώτημα θα οδηγήσουν στο να επιστέψει στο αντίστοιχο label (INT4\_R) και την επαναφορά στο loop, μετά από αυτό συμβαίνει το ίδιο αφού έχει αντιστραφεί η κατάσταση του LED.