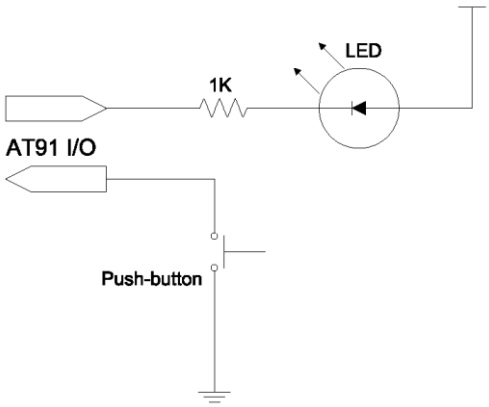
Εισαγωγική Άσκηση

Εργαστηριακή Άσκηση 1

i. Σκοπός

Σκοπός αυτής της εργαστηριακής άσκησης είναι η εξοικείωσή σας με τη μονάδα εισόδου / εξόδου του AT91, τη μονάδα διαχείρισης διακοπών (interrupts) και τη χρήση του μετρητή του συστήματος (Timer / Counter). Στο τέλος αυτής της άσκησης θα είστε σε θέση να προγραμματίζετε το AT91 ώστε να μπορεί να δεχθεί είσοδο από κάποια μονάδα και να διαβιβάσει δεδομένα στην ίδια ή κάποια άλλη, να διαχειρίζεστε σήματα διακοπών και να χρησιμοποιείτε το μετρητή του συστήματος ώστε να προγραμματίσετε χρονικές ακολουθίες.

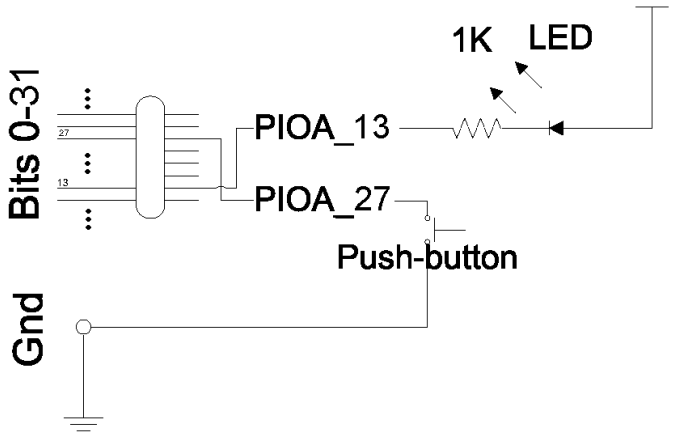


Το υποκύκλωμα της πλακέτας το οποίο πρέπει να χρησιμοποιήσετε, φαίνεται σε απλοποιημένη μορφή στην παραπάνω εικόνα και αποτελείται από ένα διακόπτη push button, ένα LED και μία αντίσταση 1K. Στην άσκηση καλείστε να προγραμματίσετε το AT91 έτσι ώστε να ελέγχει (μέσω της υπομονάδας εξόδου) το εικονιζόμενο LED. Το LED θα πρέπει είτε να αναβοσβήνει με συχνότητα 1 δευτερολέπτου είτε να παραμένει στην ίδια κατάσταση (σβηστό ή αναμμένο). Κάθε διαδοχικό πάτημα του διακόπτη σηματοδοτεί την εναλλαγή μεταξύ αυτών των δύο καταστάσεων.

Από το σχηματικό της προηγούμενης εικόνας προκύπτει πως κάθε πάτημα του διακόπτη προκαλεί ένα χαμηλό δυναμικό στη γραμμή εισόδου προς το AT91. Θα προγραμματίσουμε συνεπώς έτσι το AT91 ώστε χαμηλό δυναμικό σε αυτή τη γραμμή να προκαλεί μια διακοπή. Η ρουτίνα εξυπηρέτησης διακοπών θα περιέχει τον απαραίτητο κώδικα ώστε να ελέγχει ποια περιφερειακή συσκευή προκάλεσε τη διακοπή. Αν η διακοπή έχει προκληθεί από το πάτημα του διακόπτη και το LED δεν είναι στην κατάσταση ¨**ΑΝΑΒΟΣΒΗΝΕΙ**¨, τότε θα σηματοδοτηθεί η έναρξη λειτουργίας της μονάδας μέτρησης (Timer) για τη μέτρηση χρονικού διαστήματος ίσου με 1 δευτερόλεπτο, διαφορετικά αν το LED είναι στην κατάσταση ¨**ΑΝΑΒΟΣΒΗΝΕΙ**¨ θα σηματοδοτηθεί η λήξη λειτουργίας της μονάδας μέτρησης. Πέρα από το πάτημα του διακόπτη, θα προγραμματίσουμε το AT91 ώστε διακοπή να μπορεί να προκληθεί και από την εξάντληση της επιθυμητής μέτρησης από το μετρητή του συστήματος. Αν προκληθεί συνεπώς διακοπή από την ολοκλήρωση της μέτρησης του ενός δευτερολέπτου, η υπομονάδα εξόδου που ελέγχει τον ακροδέκτη στον οποίο είναι συνδεδεμένο το LED θα προγραμματιστεί έτσι ώστε να αντιστραφεί η κατάσταση του LED (από ενεργό να γίνει ανενεργό ή το αντίστροφο). Επιπλέον, θα προγραμματιστεί η μονάδα μέτρησης, ώστε να μετρήσει εκ νέου χρονικό διάστημα ίσο με 1 δευτερόλεπτο.

ii. Κύκλωμα

Η πλήρης συνδεσμολογία που θα πρέπει να χρησιμοποιήσετε παρουσιάζεται στο επόμενο διάγραμμα. Οι επιβλέποντες του εργαστηρίου θα σας εξηγήσουν σε ποια σημεία της πλακέτας θα βρείτε τα LEDs και τα Push-buttons καθώς και τις θέσεις των ακροδεκτών 13 (PIOA_13) και 27 (PIOA_27) της μονάδας εισόδου – εξόδου (PIOA) στις οποίες αυτά αντιστοιχούν.



Όπως παρατηρείτε, για τη διασύνδεση του κυκλώματος που αναπτύξατε με το AT91 έχουν χρησιμοποιηθεί μόνο οι ακροδέκτες 13 και 27 της μονάδας εισόδου / εξόδου. Οι παράμετροι λειτουργίας των ακροδεκτών της μονάδας εισόδου / εξόδου καθορίζονται μέσω

του προγραμματισμού των καταχωρητών της μονάδας αυτής. Αναλυτικός οδηγός για τον προγραμματισμό των καταχωρητών αυτής της μονάδας παρατίθεται στο Παράστημα Β.

Ο ακροδέκτης 27, που συνδέεται με τον διακόπτη, πρέπει να ρυθμιστεί σε λειτουργία εισόδου. Όσο ο διακόπτης δεν είναι πατημένος, η γραμμή δεν οδηγείται από εξωτερική πηγή. Για να μην μένει σε απροσδιόριστη κατάσταση, είναι απαραίτητο να ενεργοποιηθεί η εσωτερική pull up αντίσταση, ώστε το δυναμικό εισόδου της γραμμής να διατηρείται υψηλό. Όταν πατηθεί ο διακόπτης, η γραμμή 27 θα οδηγηθεί με χαμηλό δυναμικό (η εσωτερική αντίσταση που είναι της τάξης των 100K, θα αρχίσει να διαρέεται από ρεύμα πολύ χαμηλής έντασης). Η αλλαγή της κατάστασης της γραμμής 27 θα ενεργοποιήσει μια διακοπή. Αναλυτικός οδηγός για τον προγραμματισμό της μονάδας διαχείρησης διακοπών παρατίθεται στο Παράρτημα Γ. Τέλος, θα πρέπει να λάβετε υπ' όψιν σας ότι ο ελεγκτής εισόδου / εξόδου θα αντιληφθεί τόσο το πάτημα, όσο και την απελευθέρωση του διακόπτη (και στις δυο περιπτώσεις υπάρχει αλλαγή κατάστασης δυναμικού εισόδου), οπότε πρέπει να προβλεφθεί το γεγονός πως θα δημιουργηθούν 2 σήματα διακοπών.

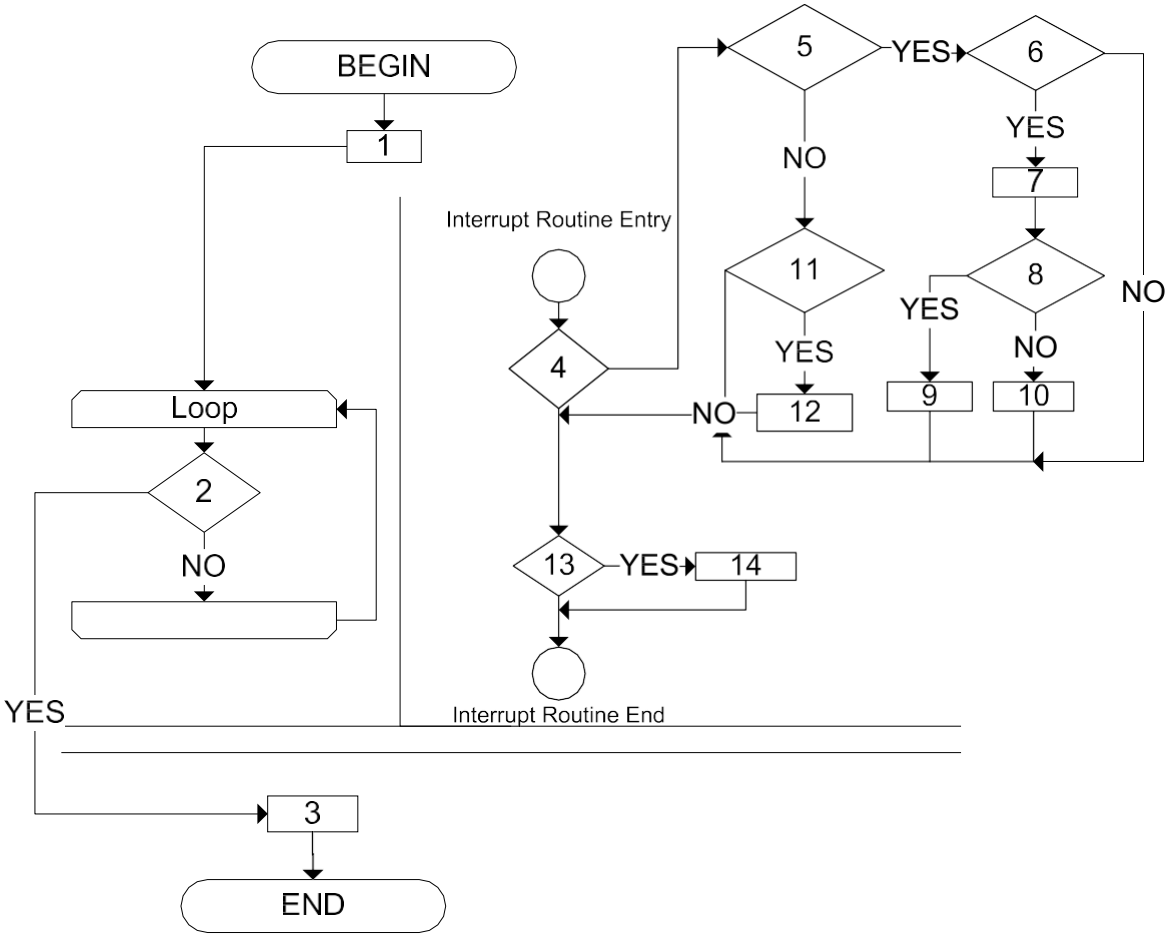
Ο ακροδέκτης 13 πρέπει να ρυθμιστεί σε λειτουργία εξόδου. Χαμηλό δυναμικό σε αυτόν τον ακροδέκτη συνεπάγεται σβήσιμο του LED. Υψηλό δυναμικό αντίθετα συνεπάγεται διαφορά τάσης στα άκρα της φωτοδιόδου μεγαλύτερη των 0.7V και συνεπώς ενεργοποίηση της πηγής φωτός.

Για τη μέτρηση του διαστήματος μεταξύ των 2 καταστάσεων του LED στην κατάσταση ¨**ΑΝΑΒΟΣΒΗΝΕΙ**¨, θα χρησιμοποιήσουμε τη μονάδα μέτρησης του AT91. Αναλυτικός οδηγός για τον προγραμματισμό αυτής της μονάδας παρατίθεται στο Παράστημα Δ.

iii. Υλοποίηση

Ο ζητούμενος προγραμματισμός του AT91 για την επίλυση της άσκησης ακολουθεί το παρακάτω flowchart. Στο flowchart έχουμε αριθμήσει με $1, 2, \ldots, 14$ τα διαφορετικά τμήματα του κώδικα που παρατίθενται με την ίδια αρίθμηση παρακάτω. Για παράδειγμα, στο σημείο 2 γίνεται ένας ατέρμονος βρόχος μέχρι να πατηθεί το πλήκτρο e, οπότε και τερματίζει το πρόγραμμα. Ο κώδικας που αντιστοιχεί στο 2 του flowchart είναι αυτός που παρατίθεται στο υποκεφάλαιο «2. Κεντρικός βρόχος».

Το αρχείο "header.h" το οποίο παρατίθεται στο παράρτημα Α, περιέχει τις δηλώσεις των δομών που χρησιμοποιούνται στον υπόλοιπο κώδικα (συμβατές με τη σημειολογία των παραρτημάτων Β, Γ και Δ) και κάποιες εντολές αρχικοποίησης. Θα πρέπει να περιλαμβάνετε το αρχείο "header.h" στον κώδικά σας για τις επόμενες ασκήσεις, χωρίς να το μεταβάλλετε.



Διάγραμμα ροής προγράμματος

1. Αρχικοποίηση συστήματος

Αρχικοποιεί το πρόγραμμα, τις μεταβλητές του συστήματος και επιτρέπει την πρόσβαση στα περιφερειακά.

```
#include <sys/types.h>
#include <sys/stat.h>
#include <fcntl.h>
#include <sys/ioctl.h>
#include <unistd.h>
#include <sys/mman.h>
#include <stdio.h>
#include <stdlib.h>

#include <header.h>

#define PIOA_ID         2
#define TC0_ID          17

#define BUT_IDLE        0
```

```
#define BUT_PRESSED     1
#define BUT_RELEASED    2

#define LED_IDLE        0
#define LED_FLASHING    1

void FIQ_handler(void);

PIO* pioa = NULL;
AIC* aic = NULL;
TC*  tc = NULL;

unsigned int button_state = BUT_IDLE;
unsigned int led_state = LED_IDLE;

int main( int argc, const char* argv[] ){
unsigned int gen;


STARTUP;                        //ΑΡΧΙΚΟΠΟΙΗΣΗ ΣΥΣΤΗΜΑΤΟΣ
tc->Channel_0.RC = 8192;        // ΠΕΡΙΟΔΟΣ 1 ΔΕΥΤΕΡΟΛΕΠΤΟ
tc->Channel_0.CMR = 0x2084; // SLOW CLOCK , WAVEFORM , DISABLE CLK ON RC COMPARE
tc->Channel_0.IDR = 0xFF;       // ΑΠΕΝΕΡΓΟΠΟΙΗΣΗ ΟΛΩΝ ΤΩΝ ΔΙΑΚΟΠΩΝ
tc->Channel_0.IER = 0x10;       // ΕΝΕΡΓΟΠΟΙΗΣΗ ΜΟΝΟ ΤΟΥ RC COMPARE

aic->FFER  = (1<<PIOA_ID) | (1<<TC0_ID); //ΟΙ ΔΙΑΚΟΠΕΣ 2,17 ΕΙΝΑΙ ΤΥΠΟΥ FIQ
aic->IECR  = (1<<PIOA_ID) | (1<<TC0_ID); //ΕΝΕΡΓΟΠΟΙΗΣΗ ΔΙΑΚΟΠΩΝ:PIOA &
TC0
pioa->PUER  = 0x8000000;                //ΕΝΕΡΓΟΠΟΙΗΣΗ ΣΤΗ ΓΡΑΜΜΗ 27:PULL-UP
pioa->ODR   = 0x8000000;                //ΓΡΑΜΜΗ 27: ΛΕΙΤΟΥΡΓΙΑ ΕΙΣΟΔΟΥ
pioa->SODR  = 0x2000;                   //ΓΡΑΜΜΗ 13 : ΔΥΝΑΜΙΚΟ ΕΞΟΔΟΥ HIGH
pioa->OER   = 0x2000;                   //ΓΡΑΜΜΗ 13: ΛΕΙΤΟΥΡΓΙΑ ΕΞΟΔΟΥ

gen         =pioa->ISR;                 //PIOA: ΕΚΚΑΘΑΡΙΣΗ ΑΠΟ ΤΥΧΟΝ ΔΙΑΚΟΠΕΣ
pioa->PER   = 0x8002000;                //ΓΡΑΜΜΕΣ 27 , 13 : ΓΕΝΙΚΟΥ ΣΚΟΠΟΥ
gen         = tc->Channel_0.SR;     // TC0 : ΕΚΚΑΘΑΡΙΣΗ ΑΠΟ ΤΥΧΟΝ ΔΙΑ ΚΟΠΕΣ
aic->ICCR  = (1<<PIOA_ID)|(1<<TC0_ID); //AIC: ΕΚΚΑΘΑΡΙΣΗ ΑΠΟ ΤΥΧΟΝ ΔΙΑΚΟΠΕΣ
pioa->IER  = 0x8000000;  // ΕΝΕΡΓΟΠΟΙΗΣΗ ΔΙΑΚΟΠΩΝ ΣΤΗ ΓΡΑΜΜΗ 27
```

2. Κεντρικός βρόχος

Εκτελείται μέχρι να πατηθεί το πλήκτρο 'e' + enter. Εδώ μπορεί να προστεθεί ο κώδικας επεξεργασίας των δεδομένων. Θεωρείται ως νήμα χαμηλής προτεραιότητας, διότι μπορεί να διακοπεί από τη ρουτίνα εξυπηρέτησης διακοπών.

```
while( (tmp = getchar()) != 'e')
{

}
```

3. Τερματισμός συστήματος

Επιστρέφει τις δεσμευμένες περιοχές στο σύστημα και απενεργοποιεί τις διακοπές.

```
aic->IDCR = (1<<PIOA_ID) | (1<<TC0_ID); // ΔΙΑΚΟΠΗ ΤΩΝ AIC interrupts
tc->Channel_0.CCR = 0x02;               // ΑΠΕΝΕΡΓΟΠΟΙΗΣΗ ΤΟΥ Timer
CLEANUP;
return 0;
}
```

Έλεγχος διακόπτη

4. Έλεγχος μονάδας εισόδου/εξόδου. Εξετάζει αν η μονάδα εισόδου/εξόδου προκάλεσε τη διακοπή, ή περιμένει να εξυπηρετηθεί.

5. Έλεγχος κατάστασης διακόπτη. Εξετάζει αν ο διακόπτης πατήθηκε ή αφέθηκε.

6. Έλεγχος σημειωμένης κατάστασης διακόπτη. Ελέγχει αν η κατάσταση στην οποία είχε σημειωθεί πως βρισκόταν ο διακόπτης, πριν αρχίσει να εξυπηρετείται η διακοπή, είναι η ¨ΜΗ ΠΑΤΗΜΕΝΟΣ¨.

7. Αλλαγή τρέχουσας κατάστασης. Σημειώνει πως η κατάσταση του διακόπτη είναι ¨ΠΑΤΗΜΕΝΟΣ¨.

8. Έλεγχος κατάστασης LED Ελέγχει αν το LED έχει σημειωθεί πως είναι σε κατάσταση ¨ΔΕΝ ΑΝΑΒΟΣΒΗΝΕΙ¨.

9. Ενεργοποίηση Μετρητή Ξεκινά τη μέτρηση του 1 δευτερολέπτου και σημειώνει πως η κατάσταση του LED είναι ¨ΑΝΑΒΟΣΒΗΝΕΙ¨.

```
void FIQ_handler(void)
{
unsigned int data_in = 0;
unsigned int fiq = 0;
unsigned int data_out;

fiq = aic->IPR; //ΕΝΤΟΠΙΣΜΟΣ ΠΕΡΙΦΕΡΕΙΑ ΚΟΥ ΠΟΥ ΠΡΟΚΑΛΕΣΕ ΤΗ ΔΙΑΚΟΠΗ

if( fiq & (1<<PIOA_ID) ){   //ΕΛΕΓΧΟΣ ΓΙΑ PIOA
    data_in = pioa->ISR;        // ΕΚΚΑΘΑΡΙΣΗ ΤΗΣ ΠΗΓΗΣ ΤΗΣ ΔΙΑΚΟΠΗΣ
    aic->ICCR = (1<<PIOA_ID);   //ΕΚΚΑΘΑΡΙΣΗ ΤΗΣ ΔΙΑΚΟΠΗΣ ΑΠΟ AIC
    data_in = pioa->PDSR;       // ΑΝΑΓΝΩΣΗ ΤΙΜΩΝ ΕΙΣΟΔΟΥ
    if(data_in & 0x8000000){        //ΔΙΑΚΟΠΤΗΣ ΠΑΤΗΜΕΝΟΣ;
        if(button_state == BUT_IDLE){
            button_state = BUT_PRESSED;
            if( led_state == LED_IDLE ){    //ΑΝ ΔΕΝ ΑΝΑΒΟΣΒΗΝΕΙ
                tc->Channel_0.CCR = 0x05;   // ΕΝΑΡΞΗ ΜΕΤΡΗΤΗ
                led_state = LED_FLASHING;
            }
```

Έλεγχος διακόπτη

10. Απενεργοποίηση Μετρητή. Απενεργοποιεί τον μετρητή και σημειώνει πως το LED είναι σε κατάσταση ¨ΔΕΝ ΑΝΑΒΟΣΒΗΝΕΙ¨.

11. Έλεγχος σημειωμένης κατάστασης διακόπτη. Ελέγχει αν η κατάσταση στην οποία είχε σημειωθεί πως βρισκόταν ο διακόπτης, πριν αρχίσει να εξυπηρετείται η διακοπή, είναι η ¨ΠΑΤΗΜΕΝΟΣ¨.

12. Αλλαγή τρέχουσας κατάστασης. Σημειώνει πως η κατάσταση του διακόπτη είναι ¨ΜΗ ΠΑΤΗΜΕΝΟΣ¨.

```
            else{
                tc->Channel_0.CCR = 0x02;   //ΔΙΑΚΟΠΗ ΜΕΤΡΗΤΗ
                led_state = LED_IDLE;
            }
        }
    }else{
        if(button_state == BUT_PRESSED)
            button_state = BUT_IDLE;
    }
}
```

Έλεγχος Μετρητή

13. Έλεγχος Μετρητή. Εξετάζει αν η Μονάδα Μετρητή προκάλεσε τη διακοπή, ή περιμένει να εξυπηρετηθεί.

14. Επανέναρξη Μέτρησης. Επαναπρογραμματίζει τη Μονάδα Μετρητή για να ξεκινήσει πάλι η μέτρηση του 1 δευτερολέπτου.

```
if( fiq & (1<<TC0_ID) ){
    data_out =tc->Channel_0.SR;// ΕΚΚΑΘΑΡΙΣΗ ΤΗΣ ΠΗΓΗΣ ΤΗΣ ΔΙΑΚΟΠΗΣ
    aic->ICCR = (1<<TC0_ID);    //ΕΚΚΑΘΑΡΙΣΗ ΔΙΑΚΟΠΗΣ ΚΑΙ ΑΠΟ AIC
    data_out = pioa->ODSR; // ΑΝΑΓΝΩΣΗ ΤΙΜΩΝ ΕΞΟΔΟΥ
    pioa->SODR = data_out & 0x2000;
    pioa->CODR = data_out & 0x2000;
    tc->Channel_0.CCR = 0x05;
   }
}
```

Παρατηρήστε πως στην αρχή του προγράμματος εκτελείται κλήση της μακροεντολής STARTUP. Η μακροεντολή STARTUP περιέχει τον κώδικα που είναι απαραίτητος για την αρχικοποίηση του συστήματος. Μια από τις διαδικασίες που εκτελούνται είναι και η ενημέρωση του λειτουργικού συστήματος για τη θέση μνήμης όπου έχει τοποθετηθεί η ρουτίνα εξυπηρέτησης διακοπών. Η ρουτίνα εξυπηρέτησης διακοπών πρέπει να έχει πάν- τα το όνομα FIQ_handler, διότι η μακροεντολή STARTUP κάνει χρήση του συγκεκριμένου ονόματος για να την δηλώσει στο λειτουργικό σύστημα. Μόλις ενεργοποιηθεί κάποια από τις διακοπές που έχουν οριστεί ως ενεργές κατά την αρχικοποίηση των περιφερειακών συσκευών, ο επεξεργαστής θα διακόψει τη ροή εκτέλεσης του προγράμματος και θα μεταβεί στην ρουτίνα εξυπηρέτησης διακοπών. Στη ρουτίνα εξυπηρέτησης πρέπει να γίνει έλεγχος για να εντοπιστεί το περιφερειακό που προκάλεσε τη διακοπή. Αν έχει προκληθεί από τη μονάδα εισόδου/εξόδου, θα γίνει έλεγχος της κατάστασης του διακόπτη και του LED, ώστε να υπολογιστεί η νέα κατάσταση στην οποία θα εισέλθει το LED. Αν έχει προκληθεί από τη μονάδα μετρητή, θα γίνει έλεγχος της τρέχουσας κατάστασης του LED (αναμμένο/σ- βηστό), θα γίνει αντιστροφή της κατάστασης του LED και θα επαναπρογραμματιστεί η μονάδα μετρητή.

Τέλος, παρατηρείστε πως το πρόγραμμα ολοκληρώνεται με την κλήση της μακροεν- τολής CLEANUP. Η μακροεντολή CLEANUP περιέχει τον απαραίτητο κώδικα για την αποδέσμευση των πηγών του συστήματος που είχαν δεσμευτεί με την κλήση της μακροεν- τολής STARTUP.

iv. Αποσφαλμάτωση

Η αποσφαλμάτωση του κυκλώματος και ο έλεγχος ορθής λειτουργίας, θα σας βοηθή- σει στο να εντοπίσετε σημεία του κυκλώματός σας που δεν εμφανίζουν την αναμενόμενη

συμπεριφορά (π.χ. βραχυκυκλωμένα, ανοιχτά κυκλώματα λόγω κακής επαφής καλωδίων κλπ). Η μεθοδολογία που μπορείτε να ακολουθήσετε είναι η ρύθμιση των γραμμών εισόδου / εξόδου με τη βοήθεια εργαλείων της κονσόλας του λειτουργικού συστήματος, η εγγραφή τιμών στους καταχωρητές που ελέγχουν την έξοδο ώστε να διαπιστωθεί αν το LED αναβοσβήνει και η ανάγνωση τιμών από τους καταχωρητές εισόδου, ώστε να διαπιστωθεί η αλλαγή του δυναμικού εισόδου κατά το πάτημα του διακόπτη.

Τα εργαλεία που θα χρησιμοποιήσετε είναι τα **mw** για την εγγραφή τιμών στις επιθυμητές θέσεις μνήμης και **md** για την ανάγνωση τιμών από τις επιθυμητές θέσεις μνήμης.

Υπενθύμιση

Η εντολή md 0 <phy_addr> <word_count> χρησιμοποιείται για να εμφανίσει τα περιεχόμενα της μνήμης. Ξεκινά από τη διεύθυνση phy_addr και τυπώνει τις πρώτες <word_count> διαδοχικές λέξεις. Για παράδειγμα, η εντολή

md 0 0xFFFFF400 4

θα τυπώσει τις τιμές των 4 πρώτων καταχωρητών της μονάδας εισόδου / εξόδου. Η εντολή mw 0 <phy_addr> <value> χρησιμοποιείται για να αποθηκεύσει την τιμή <value> στη θέση μνήμης <phy_addr>, εκτελώντας προσπέλαση λέξης (4 bytes). Για παράδειγμα, η εντολή

mw 0 0xFFFFF400 0x10

εκτελεί την εγγραφή της τιμής 16 στον καταχωρητή PIO_PER.
Επειδή η ανάγνωση και η εγγραφή γίνονται με προσπέλαση λέξης, οι διευθύνσεις μνήμης που εισάγονται ως παράμετροι στις 2 αυτές εντολές πρέπει να είναι ακέραια πολλαπλάσια του 4.

Για παράδειγμα, για να ρυθμίσετε την γραμμή 13 σε λειτουργία εξόδου, εκτελέστε τις επόμενες εντολές :

Εντολή	Λειτουργία Γραμμής	Καταχωρητής
mw 0 0xFFFFF400 0x2000	Γενικού Σκοπού	PIO_PER
mw 0 0xFFFFF410 0x2000	Έξοδος	PIO_OER
mw 0 0xFFFFF434 0x2000	Χαμηλό δυναμικό (Led1 ON)	PIO_CODR
mw 0 0xFFFFF430 0x2000	Υψηλό δυναμικό (Led1 OFF)	PIO_SODR

Η εγγραφή στον καταχωρητή PIO_SODR θα ενεργοποιήσει το LED, ενώ η εγγραφή στον PIO_CODR θα απενεργοποιήσει το LED. Με τις επόμενες εντολές γίνεται ο έλεγχος της γραμμής 27, στην οποίο είναι συνδεδεμένος ο διακόπτης :

Εντολή	**Λειτουργία Γραμμής**	**Καταχωρητής**
mw 0 0xFFFFF400 0x8000000	Γενικού Σκοπού	PIO_PER
mw 0 0xFFFFF414 0x8000000	Είσοδος	PIO_ODR
mw 0 0xFFFFF464 0x8000000	Ενεργοποίηση Pull up	PIO_PUER
md 0 0xFFFFF43C 1	Ανάγνωση δυναμικού εισόδου	PIO_PDSR

Αν ο διακόπτης είναι πατημένος, η τιμή του bit 27 της λέξης που θα εμφανίσει η εντολή md θα είναι 0 (χαμηλό δυναμικό), ενώ αν δεν είναι πατημένος, η τιμή του bit 27 της λέξης που θα εμφανιστεί θα είναι 1 (λόγω της pull up).

v. Εναλλακτική υλοποίηση

Ο χειρισμός των Led αντί να γίνεται μόνο από τα push-buttons που διαθέτει η αναπτυξιακή πλακέτα θα γίνεται και από το πληκτρολόγιο. Για παράδειγμα, θέλουμε η κατάσταση του Led1 να ελέγχεται και από του πλήκτρο «b».

Για το σκοπό αυτό ο κώδικας του κεντρικού βρόγχου που παρουσιάζεται στην υποενότητα [iii.2] θα μετατραπεί σε:

```
while( (tmp = getchar()) != 'e')
{
  if (tmp == 'b') {
   key1_state = BUT_PRESSED;
   FIQ_handler();
  }
}
```

Και στον χειρισμό του FIQ_handler θα προστεθεί ο παρακάτω κώδικας ως επέκταση της εξωτερικής δομής επιλογής ελέγχου διακοπής από τη PIOA:

```
else{
   if(key1_state == BUT_PRESSED){
     key1_state = BUT_IDLE;
     if(led_state == LED_IDLE){ // AN  ΔΕΝ ΑΝΑΒΟΣΒΗΝΕΙ
        tc->Channel_0.CCR = 0x05; // ΕΝΑΡΞΗ  ΜΕΤΡΗΤΗ
        led_state = LED_FLASHING;
     }
    else{
       tc->Channel_0.CCR = 0x02;  // ΔΙΑΚΟΠΗ  ΜΕΤΡΗΤΗ
       led_state = LED_IDLE;
    }
  }
 }
```

v. Επέκταση

Τροποποιήστε τον κώδικα που παρουσιάζεται στην υποενότητα [**ii]** ώστε να αναβοσβήνουν και τα υπόλοιπα LEDs της αναπτυξιακής πλακέτας:

α) ένα led κάθε φορά, παρέχοντας διαφορετικό κώδικα για κάθε περίπτωση

β) όλα τα led μαζί και κάθε led ελεγχόμενο από διαφορετικό push-button και keyboard-key

Επισήμανση: το Led0 που είναι και το led επισήμανσης τροφοδοσίας τίθεται από το σύστημα κατά το power-up στην κατάσταση «ΑΝΑΜΜΕΝΟ» και θα πρέπει να σβήνει κατά την φάση της αρχικοποίησης του κώδικά σας.

Στον παρακάτω πίνακα δίνεται η αντιστοίχιση των led και των push-buttons της αναπτυξιακής πλακέτας με αριθμούς ακροδεκτών της μονάδας εισόδου-εξόδου ξεκινώντας από τον ακροδέκτη 0.

Στοιχείο	**Θέση ακροδέκτη**
Led0 (power-up led)	PIOA_23
Led1	PIOA_13
Led2	PIOA_14
Push-button3	PIOA_27
Push-button4	PIOA_26
Push-button5	PIOA_25
Push-button6	PIOA_24

Το Led0 ελέγχεται μέσω διακόπτη (τρανζιστορ τύπου N) στον οποιο καταλήγει ο ακροδέκτης PIOA_23 και συνεπώς **ανάβει και σβήνει για τις αντίθετες τιμές** στον ακροδέκτη του από ότι τα Led1 και Led2.

Τα Push-buttons 1 και 2 αντιστοιχούν στην εκκίνηση των λειτουργιών reset και wake-up του συστήματος, ενώ το Push-button 0 δεν ορίζεται.