



Τμήμα Μηχανικών Η/Υ και Πληροφορικής
Πανεπιστήμιο Πατρών
Πολυτεχνική Σχολή

Τομέας Υλικού και Αρχιτεκτονικής των Υπολογιστών

Διδάσκουσα: Μαριλένα Δούναβη

Ακαδημαϊκό Έτος: 2024 – 2025

Ημ/νία Παράδοσης: 13/03/2025

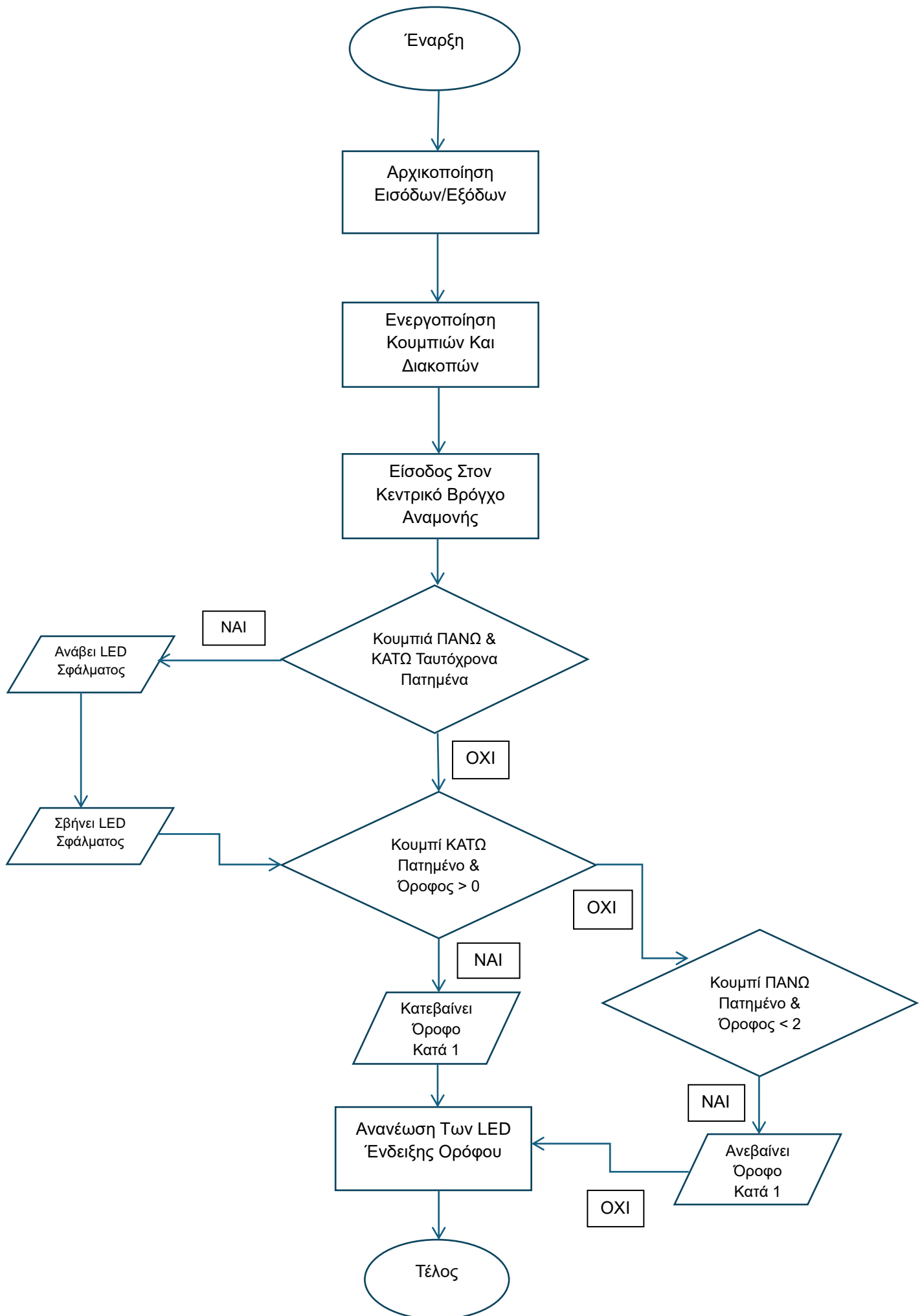
Εργαστήριο Προηγμένων Μικροεπεξεργαστών 1^η Εργαστηριακή Άσκηση Ανελκυστήρας

Εαρινό Εξάμηνο 2025

Στοιχεία Φοιτητών (ΤΜΗΜΑ Β3):

Ονοματεπώνυμο:	Σωκράτης Μαντές	Ευστάθιος Καλογερόπουλος
A.M.:	1093421	1082930
E – mail:	up1093421@ac.upatras.gr	up1082930@ac.upatras.gr
Εξάμηνο:	8	8

1.1 Διάγραμμα Ροής



1.2 Επεξήγηση Κώδικα

Αρχικά, ορίζουμε την συχνότητα λειτουργίας clock frequency του μικροελεγκτή σε 16.000.000 Hz =16 MHz ως έναν θετικό ακέραιο αριθμό χωρίς πρόσημο και μήκους 32 bit, ώστε να μην υπάρχει πιθανότητα υπερχείλισης κατά την εκτέλεση. Αφού εισαγάγουμε και τις απαραίτητες βιβλιοθήκες, ορίζουμε τις παγκόσμιες μεταβλητές level, x. Στην μεταβλητή level αποθηκεύεται ο τρέχοντας όροφος στον οποίο βρίσκεται ο χρήστης και την αρχικοποιούμε στο 0 (βρίσκεται στο ισόγειο). Επίσης, η μεταβλητή x λειτουργεί ως λογικό flag που μας δείχνει αν θα συνεχίσουμε να τρέχουμε τον βρόγχο της main() ή όχι. Όσο βρίσκεται στο 0 η main() μπαίνει σε κατάσταση αναμονής και η διαδικασία συνεχίζεται μέσα από τις διακοπές ISR.

Στην συνάρτηση main() ξεκινάμε ορίζοντας σε ποια από τα pins του PORTD θα συνδέσουμε τα LEDs. Θα αξιοποιήσουμε 3 pins, καθώς υπάρχουν 3 καταστάσεις στις οποίες ανάβει το LED. Η πρώτη όταν προκύπτει σφάλμα όταν πατάμε και τα 2 κουμπιά ταυτόχρονα, η δεύτερη όταν ο χρήστης βρίσκεται στον 1^ο όροφο και η τρίτη όταν ο χρήστης βρίσκεται στον 2^ο όροφο. Χρησιμοποιώντας το DIR θέτουμε το bit για κάθε pin στο 1, ώστε τα pins να λειτουργούν ως έξοδοι. Ακόμη, με την χρήση του OUT βάζουμε τα pins στο λογικό 1 δηλαδή σε υψηλή στάθμη, ώστε τα LEDs να μην είναι αναμμένα στην αρχική κατάσταση. Στη συνέχεια μέσω του ειδικού καταχωρητή ρύθμισης Pin Control Register ρυθμίζουμε τα PIN5 και PIN6 της θύρας PORTF ώστε να λειτουργούν ως εισοδοί όταν ο χρήστης πατάει το κουμπί του ανελκυστήρα για να πάει προς τα πάνω ή προς τα κάτω. Τα PIN5, PIN6 έχουν ενεργοποιημένη pull up αντίσταση και προκαλούν διακοπή κάθε φορά που η κατάσταση τους αλλάζει από 0 σε 1 και από 1 σε 0. Τέλος, ενεργοποιούμε το παγκόσμιο flag I γιατί χωρίς αυτό η συνάρτηση διακοπής ISR δεν θα κληθεί ποτέ ακόμα και αν πατηθεί το κουμπί του ανελκυστήρα.

Ορίζουμε την συνάρτηση διακοπής ISR η οποία θα ενεργοποιείται αυτόματα όταν υπάρξει αλλαγή σε οποιοδήποτε από τα PIN5, PIN6 του PORTF που έχουν ρυθμιστεί για διακοπή. Εκτελούμε έναν bitwise έλεγχο για να δούμε ποιο από τα δύο pins προκάλεσε την διακοπή, χρησιμοποιώντας τον 8-bit καταχωρητή σημαίας διακοπών INTFLAGS που ανήκει στην θύρα PORTF. Με το λογικό AND (PORTF.INTFLAGS & 0b01000000) != 0 ελέγχουμε αν είναι True η συνθήκη ότι το 6^ο bit του καταχωρητή είναι 1, δηλαδή αν το PIN6 ενεργοποίησε τον διακόπτη και με το λογικό AND (PORTF.INTFLAGS & 0b00100000) != 0 ελέγχουμε αν είναι True η συνθήκη ότι το 5^ο bit του καταχωρητή είναι 1, δηλαδή αν το PIN5 ενεργοποίησε τον διακόπτη. Οπότε, ελέγχουμε αν και τα δύο bits είναι ενεργά δηλαδή ότι και τα δύο κουμπιά πατήθηκαν ταυτόχρονα και αν ισχύει τότε βρισκόμαστε στην περίπτωση σφάλματος του ανελκυστήρα. Σε αυτή την περίπτωση ο καταχωρητής PORTD.OUT ελέγχει τι σήμα στέλνουμε στα pins του PORTD και με το |= θέτει όλα τα bits στο 1 εκτός από το bit0 (PIN0) που το μηδενίζει ώστε να ανάψει το LED σφάλματος. Αμέσως μετά με το |= θέτει το bit 0 σε 1 και το LED σφάλματος ξανασβήνει δημιουργώντας έναν πολύ σύντομο παλμό. Όμοια εκτελούμε έναν δεύτερο ταυτόχρονο έλεγχο για το αν το PIN6 ενεργοποίησε τον διακόπτη ((PORTF.INTFLAGS & 0b01000000) != 0) και αν δεν βρισκόμαστε ήδη στο ισόγειο (όροφος 0). Αν και οι δύο συνθήκες είναι True τότε η μεταβλητή level μειώνεται κατά 1 δηλαδή ο χρήστης κατεβαίνει όροφο. Αλλιώς εκτελούμε εκτελούμε έναν τρίτο ταυτόχρονο έλεγχο για το αν το PIN5 ενεργοποίησε τον διακόπτη ((PORTF.INTFLAGS & 0b00100000) != 0) και αν δεν βρισκόμαστε ήδη στον τελευταίο όροφο (όροφος 2). Αν και οι δύο συνθήκες είναι True τότε η μεταβλητή level αυξάνεται κατά 1 δηλαδή ο χρήστης ανεβαίνει όροφο. Αφού εκτελεστούν αυτές οι ενέργειες χρησιμοποιούμε ξανά το καταχωρητή PORTD.OUT για να ελέγξουμε τι σήμα στέλνουμε στα pins του PORTD και με το |= τα bits 0,1,2 γίνονται 1, ώστε να σβήσουν όλα τα LEDs. Ουσιαστικά γίνεται reset όλων των LEDs και μετά πραγματοποιείται ο τελικός έλεγχος προκειμένου να δούμε σε ποιο όροφο βρίσκεται ο χρήστης αφού έχει πατήσει κάποιο κουμπί του ανελκυστήρα. Αν η μεταβλητή level είναι ίση με 1 τότε βρίσκεται στον 1^ο όροφο, άρα πρέπει να ανάψει το LED του πρώτου ορόφου. Χρησιμοποιούμε έναν clear register για το PORTD (PORTD.OUTCLR = 0b00000010;) ο οποίος μηδενίζει επιλεκτικά όσα bits του δώσουμε, οπότε θέτει το PIN1 = 0 (αφού του έχουμε ορίσει να μηδενίσει το bit 1) να ανάψει το LED του πρώτου ορόφου. Αλλιώς, εκτελούμε τον ίδιο έλεγχο για να δούμε αν βρίσκεται στον 2ο όροφο. Τότε ο clear register θα μηδενίσει τα bit 1 και 2 και θα θέσει τα PIN1 = 0 και PIN2 = 0, άρα θα ανάψουν και τα δύο LEDs, γεγονός που δείχνει ότι βρισκόμαστε στον δεύτερο όροφο. Στην συγκεκριμένη περίπτωση ορίζουμε να ανάψουν και τα δύο LEDs προκειμένου να ξεχωρίζουμε πιο εύκολα τις δύο διαφορετικές καταστάσεις. Πριν τελειώσει η διακοπή είναι πολύ σημαντικό να μηδενίσουμε τα interrupt flags αλλιώς η διακοπή θα συνεχίσει να επαναλαμβάνεται, καθώς το σύστημα θα θεωρεί ότι το γεγονός συνεχίζει να ισχύει. Οπότε δημιουργούμε μια τοπική μεταβλητή y τύπου int και της δίνουμε την τρέχουσα τιμή του PORTF.INTFLAGS, δηλαδή κρατάμε ποια pins προκάλεσαν την διακοπή. Και μετά δίνουμε στο PORTF.INTFLAGS την ίδια του την τιμή, οπότε πρακτικά γράφουμε 1 στα bit που θέλουμε να καθαρίσουμε ώστε να γίνουν 0.

1.3 Κώδικας – Σχόλια

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

int level = 0; // Αρχικός όροφος του ανελκυστήρα
int x = 0;     // Flag λογικής για έλεγχο του κύριου βρόχου

// Διακοπή για την ανίχνευση πατήματος κουμπιών
ISR(PORTF_PORT_vect) {
    // Και τα δύο κουμπιά (Up και Down) πατήθηκαν ταυτόχρονα
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && (PORTF.INTFLAGS & 0b00100000) != 0) {
        PORTD.OUT &= 0b11111110; // Ανάβει το LED σφάλματος (PIN0)
        _delay_ms(del);
        PORTD.OUT |= 0b00000001; // Σβήνει το LED σφάλματος (PIN0)
        _delay_ms(del);
    }

    // Αν πατηθεί το κουμπί κάτω (PIN6) και δεν είμαστε στον 0ο όροφο
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && level > 0) {
        level--; // Κατεβαίνουμε όροφο
    }
    // Αν πατηθεί το κουμπί πάνω (PIN5) και δεν είμαστε στον 2ο όροφο
    else if ((PORTF.INTFLAGS & 0b00100000) != 0 && level < 2) {
        level++; // Ανεβαίνουμε όροφο
    }

    // Ενημέρωση LEDs απευθείας μέσα στην ISR
    PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs
    if (level == 1) {
        PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
    }
    else if (level == 2) {
        PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
    }

    // Καθαρισμός των interrupt flags
    int y = PORTF.INTFLAGS;
    PORTF.INTFLAGS = y;
}

int main() {
    // Ορισμός των LED pins ως έξοδοι
    PORTD.DIR |= 0b00000111; // PIN0 (Error), PIN1, PIN2 ως έξοδοι
    PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs αρχικά

    // Ενεργοποίηση pull-ups και ρύθμιση διακοπών στα PIN5 και PIN6
    PORTF.PIN5CTRL = PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
    PORTF.PIN6CTRL = PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;

    sei(); // Ενεργοποίηση παγκόσμιων διακοπών

    // Κύριος βρόχος: περιμένει διακοπή
    while (x == 0) {
        // Δεν κάνει τίποτα
    }

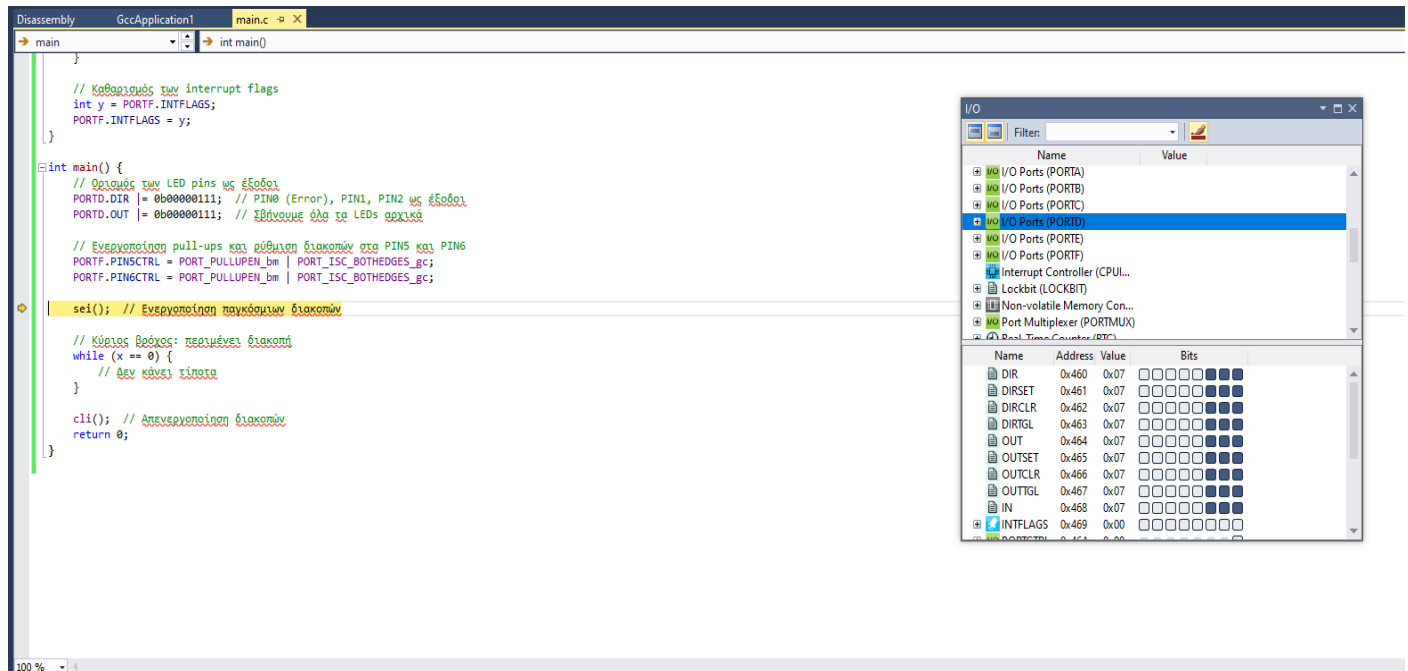
    cli(); // Απενεργοποίηση διακοπών
    return 0;
}
```

1.4 Παράδειγμα Υλοποίησης

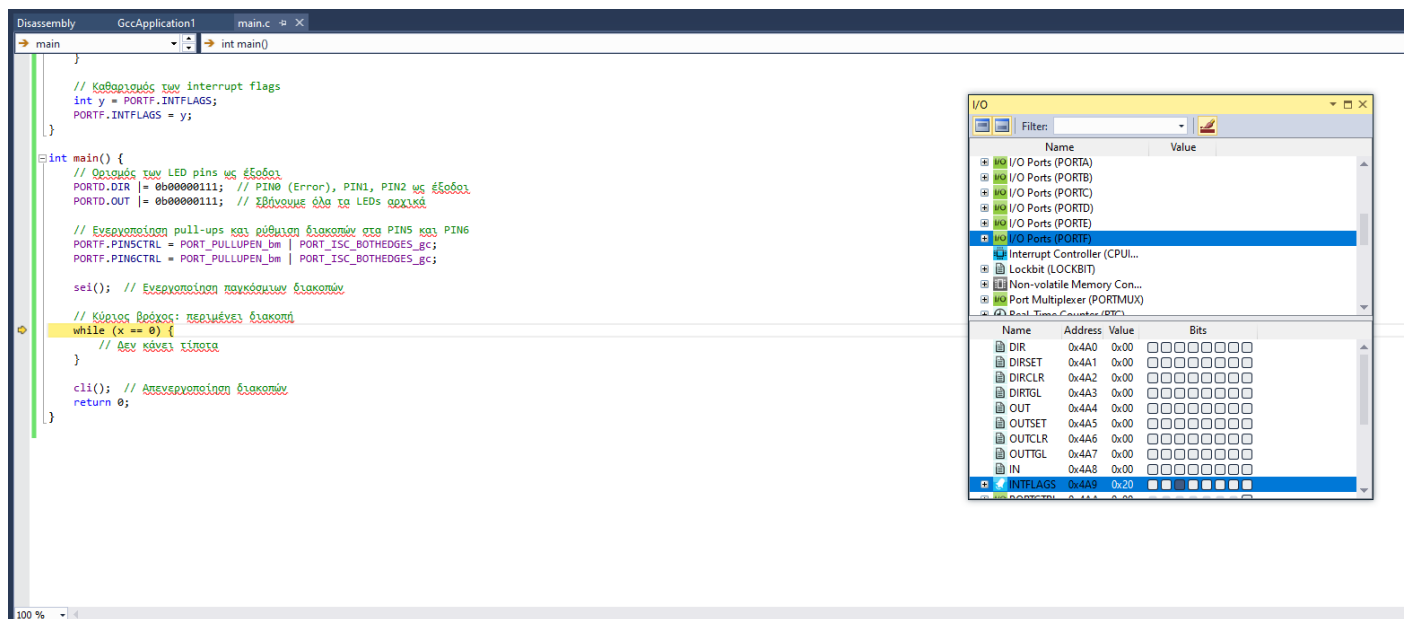
Παρακάτω υλοποιείται το σενάριο :

Αρχική Κατάσταση → Πατάω Κουμπι ΠΑΝΩ → Πατάω Κουμπι ΠΑΝΩ → Πατάω Κουμπι ΚΑΤΩ → Πατάω Κουμπι ΠΑΝΩ Και ΚΑΤΩ Ταυτόχρονα.

Στην αρχική κατάσταση όλα τα LEDs είναι σβησμένα.



Ο χρήστης πατάει το κουμπι ΠΑΝΩ, δηλαδή ενεργοποιείται το PIN5 του καταχωρητή PORTF.INTFLAGS.



Πραγματοποιείται η διακοπή ISR, οι πρώτες 2 συνθήκες ελέγχου δεν ικανοποιούνται, οπότε εκτελείται η συνθήκη όπου πατάμε το κουμπί ΠΑΝΩ και δεν βρισκόμαστε σε όροφο μεγαλύτερο του 2.

DisassemblyGccApplication1main.c

#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

int level = 0; // Αρχικός όροφος του ανελκυστήρα
int x = 0; // Flag λογικής για έλεγχο του κλειστού θρόνου

// Διακοπή (ISR) για την ανίχνευση πατώματος κουμπιών
ISR(PORTF_vect) {
 // Και τα δύο κουμπιά (Up και Down) πατήθηκαν ταυτόχρονα
 if ((PORTF.INTFLAGS & 0b01000000) != 0 && (PORTF.INTFLAGS & 0b00100000) != 0) {
 PORTD.OUT |= 0b11111110; // Ανάβει το LED σφάλματος (PIN0)
 _delay_ms(200);
 PORTD.OUT |= 0b00000001; // Σβήνει το LED σφάλματος (PIN0)
 }

 // Αν πατηθεί το κουμπί κάτω (PIN6) και δεν είμαστε στον 0ο όροφο
 if ((PORTF.INTFLAGS & 0b01000000) != 0 && level > 0) {
 level--; // Κατεβαίνουμε όροφο
 }

 // Αν πατηθεί το κουμπί πάνω (PIN5) και δεν είμαστε στον 2ο όροφο
 else if ((PORTF.INTFLAGS & 0b00100000) != 0 && level < 2) {
 level++; // Ανεβαίνουμε όροφο
 }

 // Ενημέρωση LEDs απευθείας μέσα στην ISR
 PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs
 if (level == 1) {
 PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
 }
 else if (level == 2) {
 PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
 }

 // Καθαρισμός των interrupt flags
 int y = PORTF.INTFLAGS;
 PORTF.INTFLAGS = y;
}

I/O

Filter:

NameValue

I/O Ports (PORTA)

0x00

I/O Ports (PORTB)

0x00

I/O Ports (PORTC)

0x00

I/O Ports (PORTD)

0x00

I/O Ports (PORTE)

0x00

I/O Ports (PORTF)

0x00

Interrupt Controller (CPU...

0x00

Lockbit (LOCKBIT)

0x00

Non-volatile Memory Con...

0x00

Port Multiplexer (PORTMUX)

0x00

Real-Time Counter (RTC)

0x00

NameAddressValueBits

DIR

0x4A0

0x00

00000000

DIRSET

0x4A1

0x00

00000000

DIRCLR

0x4A2

0x00

00000000

DIRTGL

0x4A3

0x00

00000000

OUT

0x4A4

0x00

00000000

OUTSET

0x4A5

0x00

00000000

OUTCLR

0x4A6

0x00

00000000

OUTTGL

0x4A7

0x00

00000000

IN

0x4A8

0x00

00000000

INTFLAGS

0x4A9

0x20

00000001

DisassemblyGccApplication1main.c

main.c

C:\Users\User\Documents\Atmel Studio\7.0\GccApplication1\GccApplication1\main.c

#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

int level = 0; // Αρχικός όροφος του ανελκυστήρα
int x = 0; // Flag λογικής για έλεγχο του κλειστού θρόνου

// Διακοπή (ISR) για την ανίχνευση πατώματος κουμπιών
ISR(PORTF_vect) {
 // Και τα δύο κουμπιά (Up και Down) πατήθηκαν ταυτόχρονα
 if ((PORTF.INTFLAGS & 0b01000000) != 0 && (PORTF.INTFLAGS & 0b00100000) != 0) {
 PORTD.OUT |= 0b11111110; // Ανάβει το LED σφάλματος (PIN0)
 _delay_ms(200);
 PORTD.OUT |= 0b00000001; // Σβήνει το LED σφάλματος (PIN0)
 }

 // Αν πατηθεί το κουμπί κάτω (PIN6) και δεν είμαστε στον 0ο όροφο
 if ((PORTF.INTFLAGS & 0b01000000) != 0 && level > 0) {
 level--; // Κατεβαίνουμε όροφο
 }

 // Αν πατηθεί το κουμπί πάνω (PIN5) και δεν είμαστε στον 2ο όροφο
 else if ((PORTF.INTFLAGS & 0b00100000) != 0 && level < 2) {
 level++; // Ανεβαίνουμε όροφο
 }

 // Ενημέρωση LEDs απευθείας μέσα στην ISR
 PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs
 if (level == 1) {
 PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
 }
 else if (level == 2) {
 PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
 }

 // Καθαρισμός των interrupt flags
 int y = PORTF.INTFLAGS;
 PORTF.INTFLAGS = y;
}

I/O

Filter:

NameValue

I/O Ports (PORTA)

0x07

I/O Ports (PORTB)

0x07

I/O Ports (PORTC)

0x07

I/O Ports (PORTD)

0x07

I/O Ports (PORTE)

0x07

I/O Ports (PORTF)

0x07

Interrupt Controller (CPU...

0x07

Lockbit (LOCKBIT)

0x07

Non-volatile Memory Con...

0x07

Port Multiplexer (PORTMUX)

0x07

Real-Time Counter (RTC)

0x07

NameAddressValueBits

DIR

0x460

0x07

00000000

DIRSET

0x461

0x07

00000000

DIRCLR

0x462

0x07

00000000

DIRTGL

0x463

0x07

00000000

OUT

0x464

0x07

00000000

OUTSET

0x465

0x07

00000000

OUTCLR

0x466

0x07

00000000

OUTTGL

0x467

0x07

00000000

IN

0x468

0x07

00000000

INTFLAGS

0x469

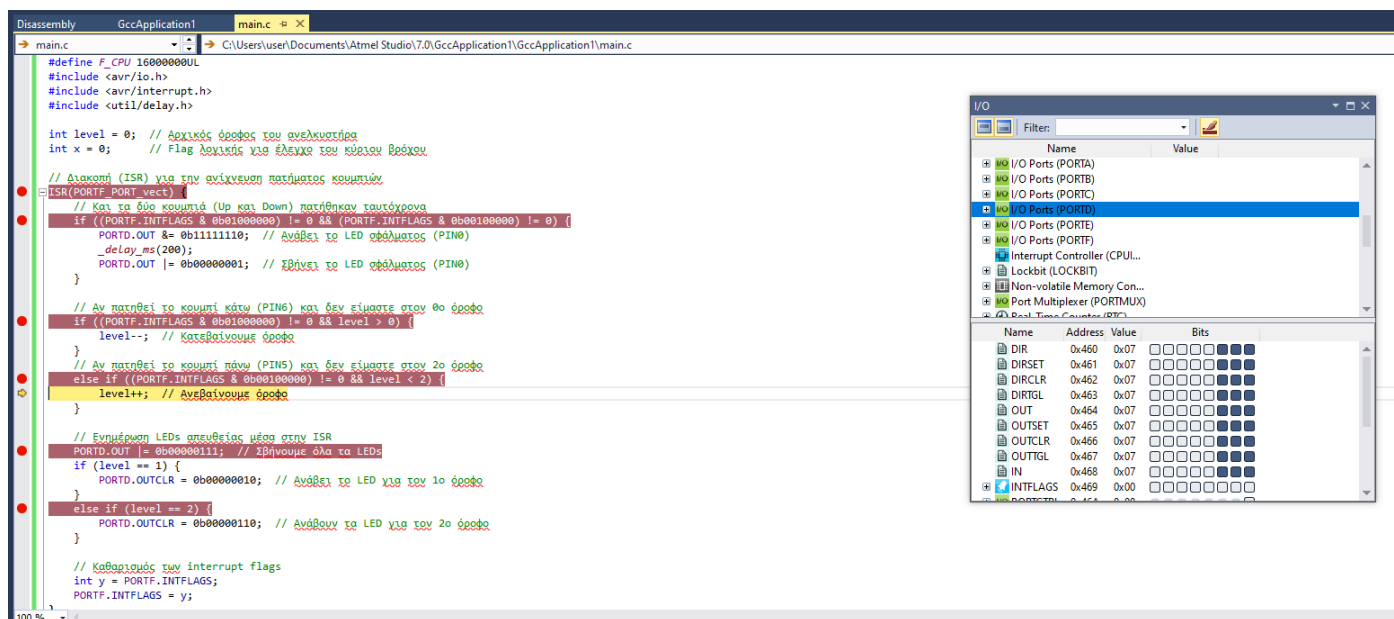
0x00

00000000

Ο όροφος στον οποίο βρισκόμαστε αυξάνεται κατά 1 και το LED που αντιστοιχεί στο PIN1 ανάβει εφόσον εκτελεστεί η συνθήκη ελέγχου if (level == 1) {

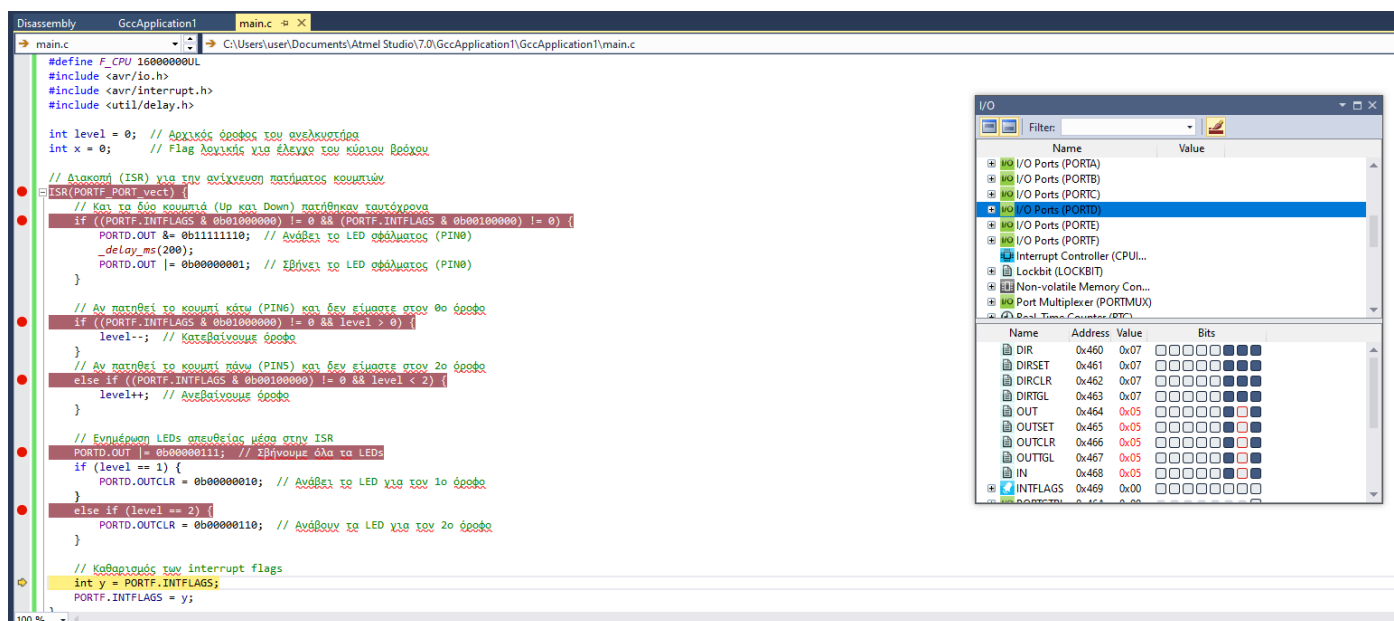
```
PORTD.OUTCLR = 0b00000010;
```

επιβεβαιώνοντας πως βρισκόμαστε στον πρώτο όροφο. Στην συνέχεια μηδενίζονται τα interrupt flags πριν επιστρέψουμε στον βρόγχο αναμονής της main().



The screenshot shows the Atmel Studio IDE with the main.c file open. The code defines a CPU frequency of 16000000UL and includes headers for AVR I/O, interrupts, and delays. It sets up a variable 'level' to 0 and a flag 'x' to 0. An interrupt service routine (ISR) is defined for PORTF, which checks for button presses on PIN0 and PIN5. If PIN0 is pressed, it toggles the LED on PIN0 and increments 'level'. If PIN5 is pressed, it toggles the LED on PIN5 and increments 'level'. The main function calls the ISR and then checks if 'level' is 1 or 2. If 'level' is 1, it clears the LED on PIN1 (PORTD.OUTCLR = 0b00000010). If 'level' is 2, it clears the LED on PIN2 (PORTD.OUTCLR = 0b00000110). Finally, it clears the interrupt flags (PORTF.INTFLAGS = 0).

The I/O register window on the right shows the status of various I/O registers. The 'INTFLAGS' register is highlighted, showing its address (0x469) and value (0x00). The 'OUT' register is also highlighted, showing its address (0x464) and value (0x00).



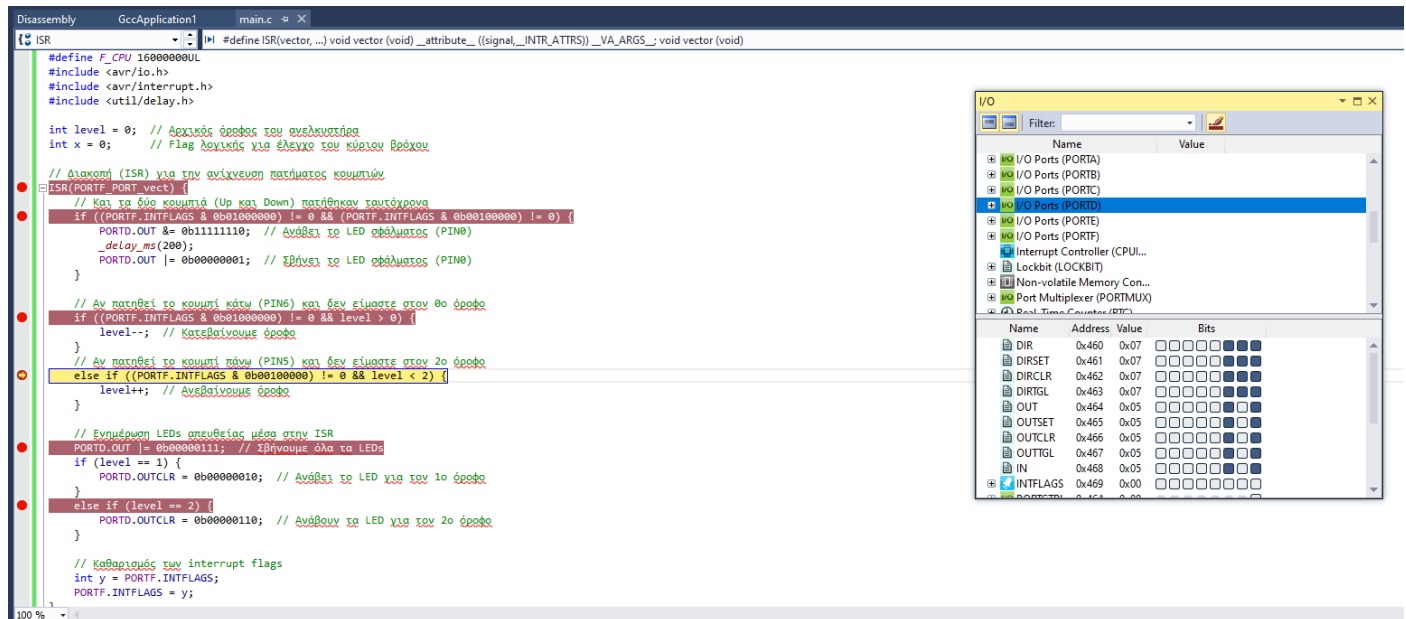
The screenshot shows the Atmel Studio IDE with the main.c file open. The code is the same as in the previous screenshot. The I/O register window on the right shows the status of various I/O registers. The 'INTFLAGS' register is highlighted, showing its address (0x469) and value (0x00). The 'OUT' register is also highlighted, showing its address (0x464) and value (0x00). The 'OUTCLR' register is highlighted, showing its address (0x466) and value (0x05).

Ο χρήστης πατάει ξανά το κουμπί ΠΑΝΩ, δηλαδή ενεργοποιείται πάλι το PIN5 του καταχωρητή PORTF.INTFLAGS, οπότε εκτελείται πάλι η συνθήκη όπου πατάμε το κουμπί ΠΑΝΩ και δεν βρισκόμαστε σε όροφο μεγαλύτερο του 2. Ο όροφος στον οποίο βρισκόμαστε αυξάνεται κατά 1 και τα LED που αντιστοιχούν στο PIN1 και στο PIN2 ανάβουν εφόσον εκτελεστεί η συνθήκη else if (level == 2) {

```
PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
```

```
}
```

επιβεβαιώνοντας πως βρισκόμαστε στον δεύτερο όροφο. Στην συνέχεια μηδενίζονται τα interrupt flags πριν επιστρέψουμε στον βρόγχο αναμονής της main().



```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

int level = 0; // Αρχικός όροφος του ανελκυστήρα
int x = 0; // Flag λογικής για έλεγχο του κύριου βρόχου

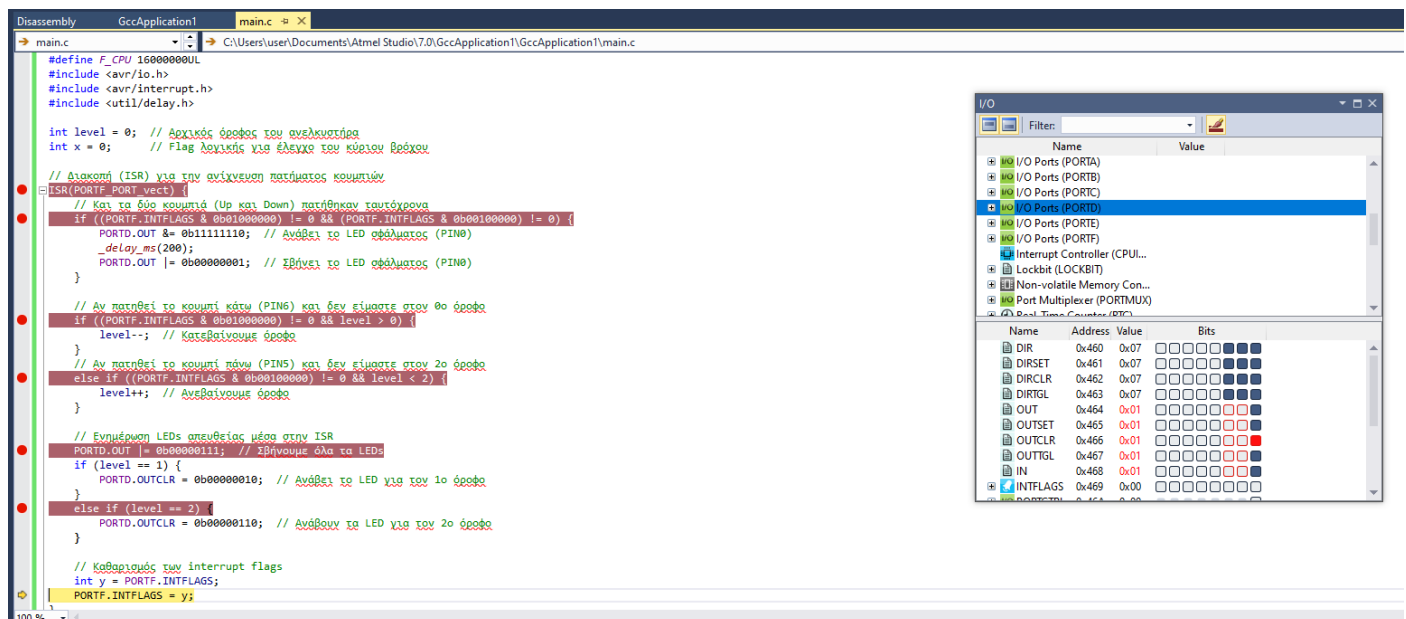
// Διακοπή (ISR) για την ανίχνευση πατήματος κουμπιών
ISR(PORTF_PORT_vect) {
    // Και τα δύο κουμπιά (Up και Down) πατήθηκαν ταυτόχρονα
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && (PORTF.INTFLAGS & 0b00100000) != 0) {
        PORTD.OUT |= 0b11111110; // Ανάβει το LED σφάλματος (PIN0)
        _delay_ms(200);
        PORTD.OUT |= 0b00000001; // Σβήνει το LED σφάλματος (PIN0)
    }

    // Αν πατηθεί το κουμπί κάτω (PIN6) και δεν είμαστε στον 0ο όροφο
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && level > 0) {
        level--; // Κατεβαίνουμε όροφο
    }

    // Αν πατηθεί το κουμπί πάνω (PIN5) και δεν είμαστε στον 2ο όροφο
    else if ((PORTF.INTFLAGS & 0b00100000) != 0 && level < 2) {
        level++; // Ανεβαίνουμε όροφο
    }

    // Ενημέρωση LEDs απευθείας μέσα στην ISR
    PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs
    if (level == 1) {
        PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
    }
    else if (level == 2) {
        PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
    }

    // Καθαρισμός των interrupt flags
    int y = PORTF.INTFLAGS;
    PORTF.INTFLAGS = y;
}
```



```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

int level = 0; // Αρχικός όροφος του ανελκυστήρα
int x = 0; // Flag λογικής για έλεγχο του κύριου βρόχου

// Διακοπή (ISR) για την ανίχνευση πατήματος κουμπιών
ISR(PORTF_PORT_vect) {
    // Και τα δύο κουμπιά (Up και Down) πατήθηκαν ταυτόχρονα
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && (PORTF.INTFLAGS & 0b00100000) != 0) {
        PORTD.OUT |= 0b11111110; // Ανάβει το LED σφάλματος (PIN0)
        _delay_ms(200);
        PORTD.OUT |= 0b00000001; // Σβήνει το LED σφάλματος (PIN0)
    }

    // Αν πατηθεί το κουμπί κάτω (PIN6) και δεν είμαστε στον 0ο όροφο
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && level > 0) {
        level--; // Κατεβαίνουμε όροφο
    }

    // Αν πατηθεί το κουμπί πάνω (PIN5) και δεν είμαστε στον 2ο όροφο
    else if ((PORTF.INTFLAGS & 0b00100000) != 0 && level < 2) {
        level++; // Ανεβαίνουμε όροφο
    }

    // Ενημέρωση LEDs απευθείας μέσα στην ISR
    PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs
    if (level == 1) {
        PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
    }
    else if (level == 2) {
        PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
    }

    // Καθαρισμός των interrupt flags
    int y = PORTF.INTFLAGS;
    PORTF.INTFLAGS = y;
}
```

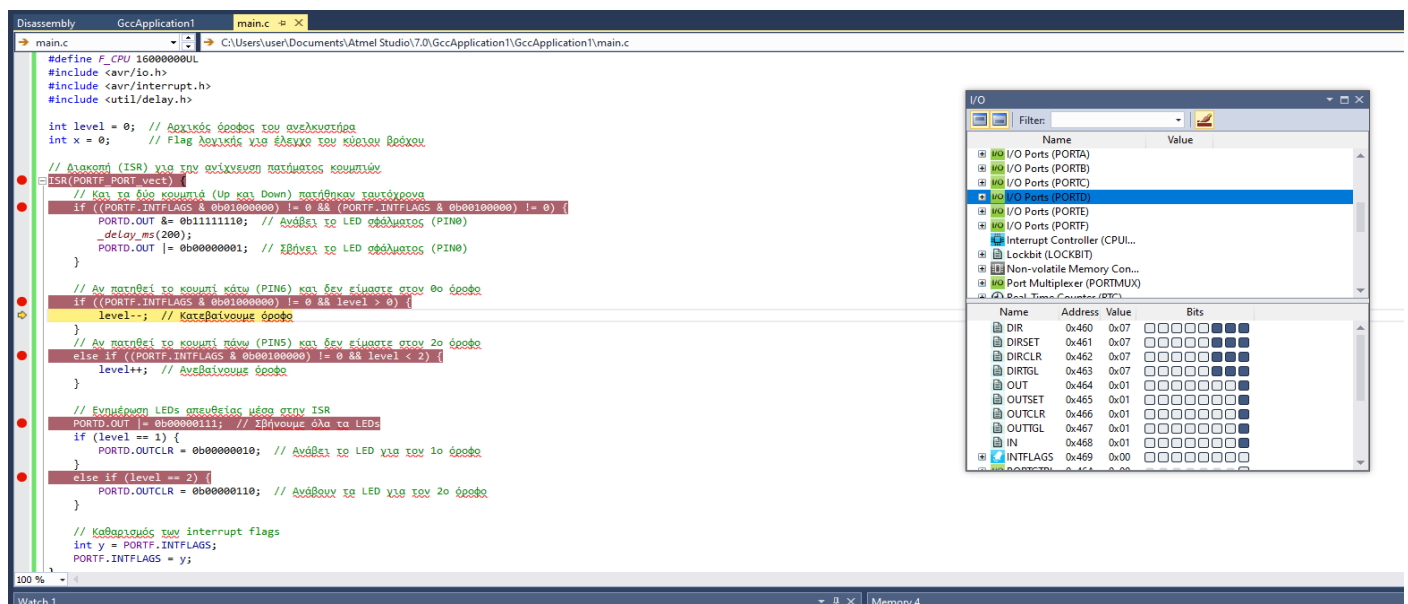
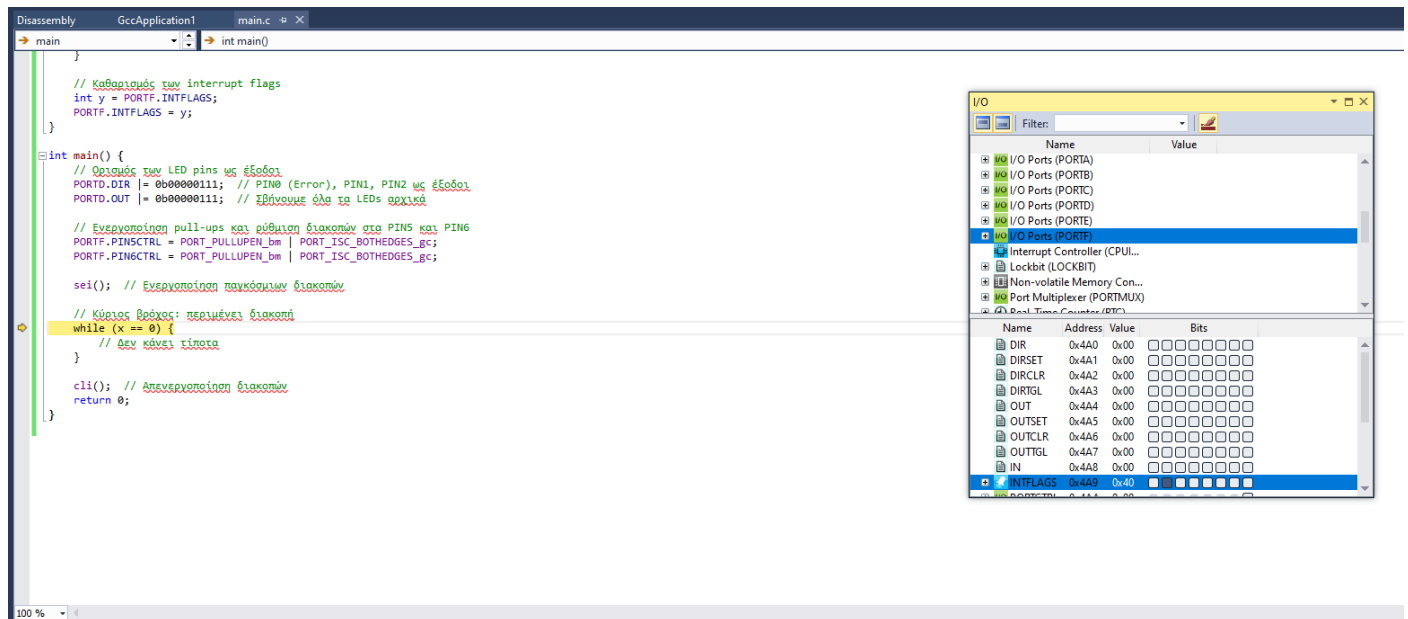

Ο χρήστης πατάει το κουμπί ΚΑΤΩ, δηλαδή ενεργοποιείται το PIN6 του καταχωρητή PORTF.INTFLAGS, οπότε εκτελείται η συνθήκη όπου πατάμε το κουμπί ΚΑΤΩ και δεν βρισκόμαστε στον όροφο 0. Ο όροφος στον οποίο βρισκόμαστε μειώνεται κατά 1 και το LED που αντιστοιχεί στο PIN1 ανάβει εφόσον εκτελεστεί η συνθήκη

```
if (level == 1) {
```

```
PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
```

```
}
```

επιβεβαιώνοντας πως βρισκόμαστε στον πρώτο όροφο. Στην συνέχεια μηδενίζονται τα interrupt flags πριν επιστρέψουμε στον βρόγχο αναμονής της main().





Disassembly GccApplication1 main.c

```
main.c
C:\Users\User\Documents\Atmel Studio\7.0\GccApplication1\GccApplication1\main.c

// Αν πατηθεί το κουμπί κάτω (PIN6) και δεν είμαστε στον 0ο όροφο
if ((PORTF.INTFLAGS & 0b01000000) != 0 && level > 0) {
    level--; // Κατεβαίνουμε όροφο
}
// Αν πατηθεί το κουμπί πάνω (PIN5) και δεν είμαστε στον 2ο όροφο
else if ((PORTF.INTFLAGS & 0b01000000) != 0 && level < 2) {
    level++; // Ανεβαίνουμε όροφο
}

// Ενημέρωση LEDs απευθείας μέσα στην ISR
PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs
if (level == 1) {
    PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
}
else if (level == 2) {
    PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
}

// Καθαρισμός των interrupt flags
int y = PORTF.INTFLAGS;
PORTF.INTFLAGS = y;
}

int main() {
    // Ορισμός των LED pins ως έξοδοι
    PORTD.DIR |= 0b00000111; // PIN0 (Error), PIN1, PIN2 ως έξοδοι
    PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs αρχικά

    // Ενεργοποίηση pull-ups και ρύθμιση διακοπών στα PIN5 και PIN6
    PORTF.PINCTRL = PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
    PORTF.PINCTRL = PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;

    sei(); // Ενεργοποίηση παγκόσμιων διακοπών

    // Κύριος βρόχος: περιμένει διακοπή
    while (x == 0) {
        // Δεν κάνει τίποτα
    }
}
```

I/O

Name	Address	Value	Bits
DIR	0x460	0x07	00000111
DIRSET	0x461	0x07	00000111
DIRCLR	0x462	0x07	00000111
DIRTGL	0x463	0x07	00000111
OUT	0x464	0x05	00000101
OUTSET	0x465	0x05	00000101
OUTCLR	0x466	0x05	00000101
OUTTGL	0x467	0x05	00000101
IN	0x468	0x05	00000101
INTFLAGS	0x469	0x00	00000000

Ο χρήστης πατάει τα κουμπιά ΠΑΝΩ και ΚΑΤΩ ταυτόχρονα, δηλαδή ενεργοποιούνται ταυτόχρονα το PIN5 και το PIN6 του καταχωρητή PORTF.INTFLAGS.

Disassembly GccApplication1 main.c

```
main.c
C:\Users\User\Documents\Atmel Studio\7.0\GccApplication1\GccApplication1\main.c

// Καθαρισμός των interrupt flags
int y = PORTF.INTFLAGS;
PORTF.INTFLAGS = y;
}

int main() {
    // Ορισμός των LED pins ως έξοδοι
    PORTD.DIR |= 0b00000111; // PIN0 (Error), PIN1, PIN2 ως έξοδοι
    PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs αρχικά

    // Ενεργοποίηση pull-ups και ρύθμιση διακοπών στα PIN5 και PIN6
    PORTF.PINCTRL = PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;
    PORTF.PINCTRL = PORT_PULLUPEN_bm | PORT_ISC_BOTHEDGES_gc;

    sei(); // Ενεργοποίηση παγκόσμιων διακοπών

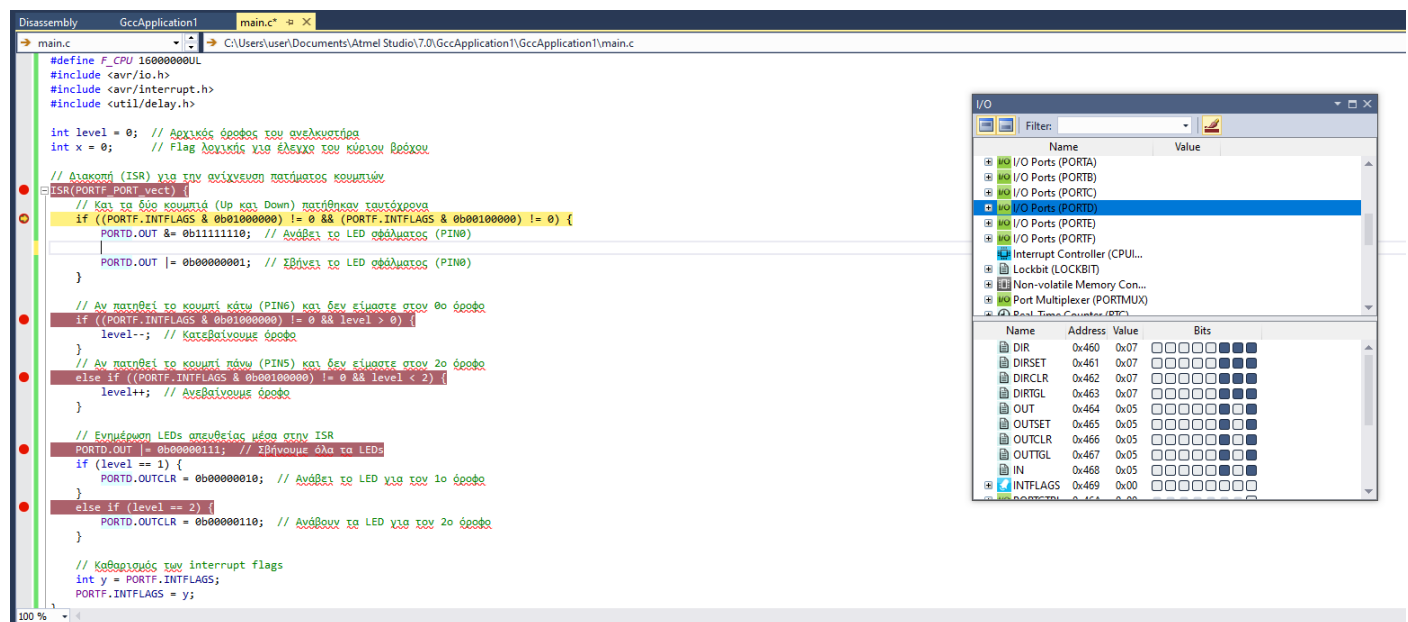
    // Κύριος βρόχος: περιμένει διακοπή
    while (x == 0) {
        // Δεν κάνει τίποτα
    }

    cli(); // Απενεργοποίηση διακοπών
    return 0;
}
```

I/O

Name	Address	Value	Bits
DIR	0x4A0	0x00	00000000
DIRSET	0x4A1	0x00	00000000
DIRCLR	0x4A2	0x00	00000000
DIRTGL	0x4A3	0x00	00000000
OUT	0x4A4	0x00	00000000
OUTSET	0x4A5	0x00	00000000
OUTCLR	0x4A6	0x00	00000000
OUTTGL	0x4A7	0x00	00000000
IN	0x4A8	0x00	00000000
INTFLAGS	0x4A9	0x60	01100000

Τώρα εκτελείται η αρχική ταυτόχρονη συνθήκη και το LED που αντιστοιχεί στο PIN0 ανάβει, εφόσον πρέπει να ανάψει το LED σφάλματος. Δεν εκτελούνται καθόλου οι ακόλουθες 2 συνθήκες και κατευθείαν μηδενίζονται τα interrupt flags πριν επιστρέψουμε στον βρόγχο αναμονής της main().



```
#define F_CPU 16000000L
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

int level = 0; // Αρχικός όροφος του ανελκυστήρα
int x = 0; // Flag λογικής για έλεγχο του κύριου βρόχου

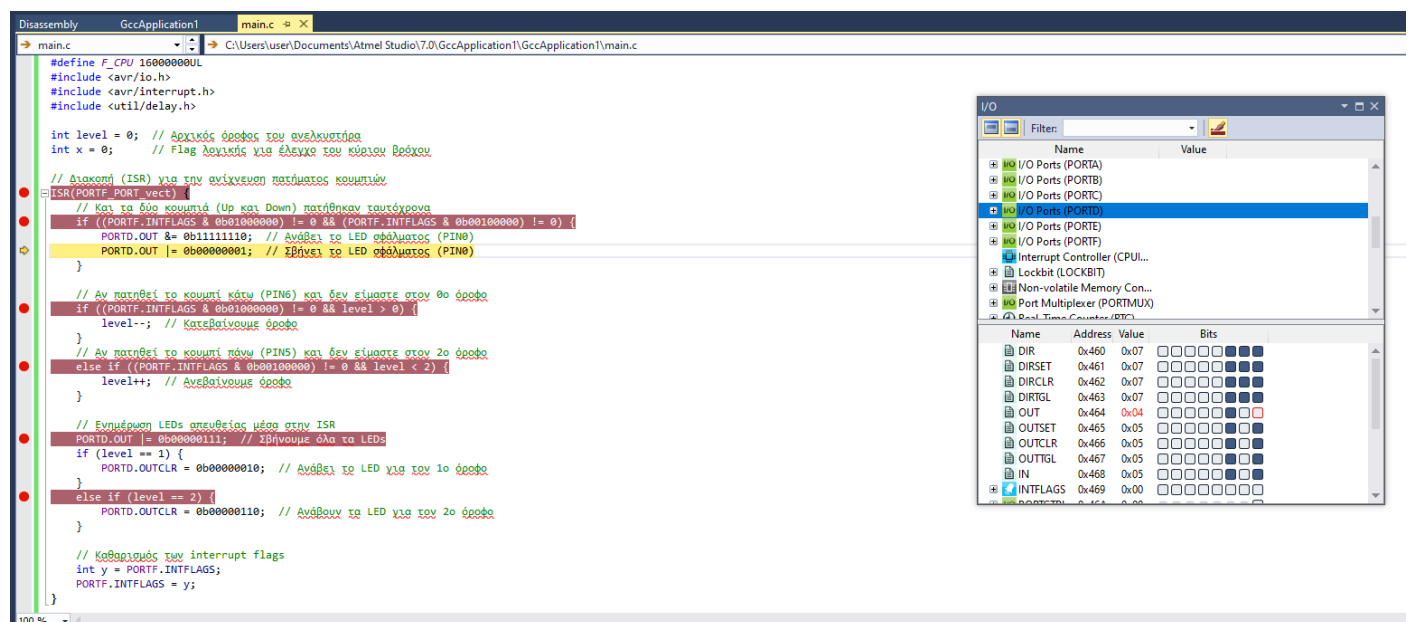
// Διακοπή (ISR) για την ανίχνευση πατώματος κουμπιών
ISR(PORTF_vect) {
    // Και τα δύο κουμπιά (Up και Down) πατήθηκαν ταυτόχρονα
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && (PORTF.INTFLAGS & 0b00100000) != 0) {
        PORTD.OUT &= 0b11111110; // Ανάβει το LED σφάλματος (PIN0)
        PORTD.OUT |= 0b00000001; // Σβήνει το LED σφάλματος (PIN0)
    }

    // Αν πατηθεί το κουμπί κάτω (PIN6) και δεν είμαστε στον 0ο όροφο
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && level > 0) {
        level--; // Κατεβαίνουμε όροφο
    }

    // Αν πατηθεί το κουμπί πάνω (PIN5) και δεν είμαστε στον 2ο όροφο
    else if ((PORTF.INTFLAGS & 0b00100000) != 0 && level < 2) {
        level++; // Ανεβαίνουμε όροφο
    }

    // Ενημέρωση LEDs απευθείας μέσα στην ISR
    PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs
    if (level == 1) {
        PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
    }
    else if (level == 2) {
        PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
    }

    // Καθαρισμός των interrupt flags
    int y = PORTF.INTFLAGS;
    PORTF.INTFLAGS = y;
}
```



```
#define F_CPU 16000000L
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

int level = 0; // Αρχικός όροφος του ανελκυστήρα
int x = 0; // Flag λογικής για έλεγχο του κύριου βρόχου

// Διακοπή (ISR) για την ανίχνευση πατώματος κουμπιών
ISR(PORTF_vect) {
    // Και τα δύο κουμπιά (Up και Down) πατήθηκαν ταυτόχρονα
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && (PORTF.INTFLAGS & 0b00100000) != 0) {
        PORTD.OUT &= 0b11111110; // Ανάβει το LED σφάλματος (PIN0)
        PORTD.OUT |= 0b00000001; // Σβήνει το LED σφάλματος (PIN0)
    }

    // Αν πατηθεί το κουμπί κάτω (PIN6) και δεν είμαστε στον 0ο όροφο
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && level > 0) {
        level--; // Κατεβαίνουμε όροφο
    }

    // Αν πατηθεί το κουμπί πάνω (PIN5) και δεν είμαστε στον 2ο όροφο
    else if ((PORTF.INTFLAGS & 0b00100000) != 0 && level < 2) {
        level++; // Ανεβαίνουμε όροφο
    }

    // Ενημέρωση LEDs απευθείας μέσα στην ISR
    PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs
    if (level == 1) {
        PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
    }
    else if (level == 2) {
        PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
    }

    // Καθαρισμός των interrupt flags
    int y = PORTF.INTFLAGS;
    PORTF.INTFLAGS = y;
}
```



DisassemblyGccApplication1main.c

main.c

C:\Users\user\Documents\Atmel Studio\7.0\GccApplication1\GccApplication1\main.c

```
#define F_CPU 16000000UL
#include <avr/io.h>
#include <avr/interrupt.h>
#include <util/delay.h>

int level = 0; // Αρχικός όροφος του ανελευστήρα
int x = 0; // Flag λογικής για έλεγχο του κύριου βρόχου

// Διακοπή (ISR) για την ανίχνευση πατώματος κουμπιών
ISR(PORTF_vect) {
    // Και τα δύο κουμπιά (Up και Down) πατήθηκαν ταυτόχρονα
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && (PORTF.INTFLAGS & 0b00100000) != 0) {
        PORTD.OUT |= 0b11111110; // Ανάβει το LED σφάλματος (PIN0)
        PORTD.OUT |= 0b00000001; // Σβήνει το LED σφάλματος (PIN0)
    }

    // Αν πατηθεί το κουμπί κάτω (PIN6) και δεν είμαστε στον 0ο όροφο
    if ((PORTF.INTFLAGS & 0b01000000) != 0 && level > 0) {
        level--; // Κατεβαίνουμε όροφο
    }

    // Αν πατηθεί το κουμπί πάνω (PIN5) και δεν είμαστε στον 2ο όροφο
    else if ((PORTF.INTFLAGS & 0b00100000) != 0 && level < 2) {
        level++; // Ανεβαίνουμε όροφο
    }

    // Ενημέρωση LEDs απευθείας μέσα στην ISR
    PORTD.OUT |= 0b00000111; // Σβήνουμε όλα τα LEDs
    if (level == 1) {
        PORTD.OUTCLR = 0b00000010; // Ανάβει το LED για τον 1ο όροφο
    }
    else if (level == 2) {
        PORTD.OUTCLR = 0b00000110; // Ανάβουν τα LED για τον 2ο όροφο
    }

    // Καθαρισμός των interrupt flags
    int y = PORTF.INTFLAGS;
    PORTF.INTFLAGS = y;
}
```

I/O

Name	Value
I/O Ports (PORTA)	
I/O Ports (PORTB)	
I/O Ports (PORTC)	
I/O Ports (PORTD)	
I/O Ports (PORTE)	
I/O Ports (PORTF)	
Interrupt Controller (CPU...	
Lockbit (LOCKBIT)	
Non-volatile Memory Con...	
Port Multiplexer (PORTMUX)	
Real Time Counter (RTC)	

Name	Address	Value	Bits
DIR	0x460	0x07	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
DIRSET	0x461	0x07	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
DIRCLR	0x462	0x07	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
DIRTGL	0x463	0x07	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
OUT	0x464	0x07	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
OUTSET	0x465	0x07	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
OUTCLR	0x466	0x07	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
OUTTGL	0x467	0x07	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
IN	0x468	0x07	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>
INTFLAGS	0x469	0x00	<input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/>

100 %