Εθνικό Μετσόβιο Πολυτεχνείο
Σχολή Ηλεκτρολόγων Μηχανικών και Μηχανικών Υπολογιστών

Μ Α Θ Η Μ Α
ΕΡΓΑΣΤΗΡΙΟ ΜΙΚΡΟΫΠΟΛΟΓΙΣΤΩΝ (ΡΟΗ Υ)

Ε Ρ Γ Α Σ Ι Α
6Η ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ

Σ Π Ο Υ Δ Α Σ Τ Ε Σ
ΚΑΡΑΜ ΚΩΝΣΤΑΝΤΙΝΟΣ
ΚΟΛΙΟΣ ΑΠΟΣΤΟΛΟΣ

2022–23

Οι πλήρεις κώδικες όλων των ασκήσεων περιέχονται στο zipped αρχείο.

Ζήτημα 6.1 *(Lab6_1.c)*

Οι συναρτήσεις που υλοποιήθηκαν φαίνονται παρακάτω:

```
/* KEYPAD FUNCTIONS */

unsigned char ram[2];       // memory
unsigned char keypad[2];    // keypad[0] contains rows 1, 2
                            // keypad[1] contains rows 3, 4
unsigned char x, y;         // x is 1st digit and y is 2nd digit (in
ASCII))

unsigned char scan_row(int i) {
    unsigned char a =0b00000001;
    i = i - 1;
    a = (a << i);
    a = ~a;
    PCA9555_0_write(REG_OUTPUT_1, a);   // set 0 (inverted logic) to the
i-th row
    uint8_t inp = PCA9555_0_read(REG_INPUT_1);  // read input to check if
column is tapped on the i-th row
    return ~(inp | 0x0F);   // keep 4 MSB which are for the column
(~inverted logic)
}

// swap lo with hi bits function
unsigned char swap_nibbles(unsigned char x) {
    return ((x & 0x0F) << 4 | (x & 0xF0) >> 4);
```

```
}

void scan_keypad() {
    unsigned char i;

    i = scan_row(1);   // 4 LSB -> *, 0, #, D
    keypad[0] = swap_nibbles(i);    // from MSB to LSB

    i = scan_row(2);   // 4 MSB -> 7, 8, 9, C
    keypad[0] += i;

    i = scan_row(3);   // 4 LSB -> 4, 5, 6, B
    keypad[1] = swap_nibbles(i);    // from MSB to LSB

    i = scan_row(4);   // 4 MSB -> 1, 2, 3, A
    keypad[1] += i;
}

int scan_keypad_rising_edge() {

    scan_keypad();

    unsigned char tmp_keypad[2];    // save 1st call state at tmp_keypad[]
    tmp_keypad[0] = keypad[0];
    tmp_keypad[1] = keypad[1];

    _delay_ms(10);

    scan_keypad();

    keypad[0] &= tmp_keypad[0];     // compare states before and after 10
ms
    keypad[1] &= tmp_keypad[1];

    tmp_keypad[0] = ram[0];
    tmp_keypad[1] = ram[1];

    ram[0] = keypad[0];     // save state after comparison at ram memory
    ram[1] = keypad[1];

    keypad[0] &= ~tmp_keypad[0];
    keypad[1] &= ~tmp_keypad[1];

    return (keypad[0] || keypad[1]);    // returns true if key is pressed
}
```

```
unsigned char keypad_to_ascii() {
    if (keypad[0] & 0x01)
        return '*';

    if (keypad[0] & 0x02)
        return '0';

    if (keypad[0] & 0x04)
        return '#';

    if (keypad[0] & 0x08)
        return 'D';

    if (keypad[0] & 0x10)
        return '7';

    if (keypad[0] & 0x20)
        return '8';

    if (keypad[0] & 0x40)
        return '9';

    if (keypad[0] & 0x80)
        return 'C';

    if (keypad[1] & 0x01)
        return '4';

    if (keypad[1] & 0x02)
        return '5';

    if (keypad[1] & 0x04)
        return '6';

    if (keypad[1] & 0x08)
        return 'B';

    if (keypad[1] & 0x10)
        return '1';

    if (keypad[1] & 0x20)
        return '2';

    if (keypad[1] & 0x40)
```

```
        return '3';

    if (keypad[1] & 0x80)
        return 'A';

    return 0;   // error
}
```

Και ο κώδικας που χρησιμοποιήθηκε στην main είναι:

```
int main ()
{
    twi_init();
    DDRB = 0x00;
    DDRD = 0xFF;
    PCA9555_0_write(REG_CONFIGURATION_0, 0x00);
    PCA9555_0_write(REG_CONFIGURATION_1, 0xF0);

    LCD_init();

    while(1)
    {
        ram[0] = 0;     // clear memory
        ram[1] = 0;

        while (1) {
            if (scan_keypad_rising_edge()) {
                x = keypad_to_ascii();
                LCD_init();
                LCD_data(x);
                break;
            }
        }
    }
}
```

Θα εξηγήσουμε εν συντομία την λειτουργία του προγράμματος. Η συνάρτηση scan_row δέχεται ως όρισμα τον αριθμό της γραμμής που θέλουμε να ελέγξουμε και αφού διαβάσουμε την είσοδο απομονώνουμε τα τέσσερα MSB τα οποία αντιστοιχούν στις τέσσερις στήλες. Η συνάρτηση scan_keypad αποθηκεύει στην αντίστοιχη θέση (όπως αναγράφεται στα σχόλια) του πίνακα keypad[] το κουμπί που πατήθηκε. Με την συνάρτηση scan_keypad_rising_edge, καλούμε την συνάρτηση scan_keypad δύο φορές με διαφορά 10 msec, ώστε να εξαλείψουμε το φαινόμενο του σπινθηρισμού. Η συνάρτηση αυτή επιστρέφει true εάν έχει πατηθεί κάποιο κουμπί και σώζει ως ολοκληρωμένη πλέον ενέργεια το κουμπί

που πατήθηκε στην αντίστοιχη θέση του πίνακα keypad[]. Τέλος, η συνάρτηση keypad_to_ascii κάνει την αντιστοίχιση των χαρακτήρων με την θέση που βρίσκονται στον πίνακα keypad[].

Ζήτημα 6.2 *(Lab6_2.c)*

Οι συναρτήσεις είναι ίδιες με αυτές του *Ζητήματος 6.1*, και παραθέτουμε τον κώδικα της main που έχει αλλάξει:

```
int main ()
{
    twi_init();
    DDRB = 0xFF;
    DDRD = 0xFF;
    PCA9555_0_write(REG_CONFIGURATION_0, 0x00);
    PCA9555_0_write(REG_CONFIGURATION_1, 0xF0);

    while(1)
    {

    check:

        ram[0] = 0;     // clear memory
        ram[1] = 0;

        while (1) {
            // scan 1st digit
            if (scan_keypad_rising_edge()) {
                x = keypad_to_ascii();
                break;
            }
        }
        // 1st digit not equal
        if (x != '6')
            goto wrong;

        // scan 2nd digit
        while (1) {
            if (scan_keypad_rising_edge()) {
                y = keypad_to_ascii();
                break;
            }
        }
```

```
        // 2nd digit not equal
        if (y != '0')
            goto wrong;

        // correct password
        PORTB = 0xFF;
        _delay_ms(4000);
        PORTB = 0x00;
        _delay_ms(1000);
        goto check;

    wrong:
        for (int j=0; j<10; j++) {
            PORTB = 0xFF;
            _delay_ms(250);
            PORTB = 0x00;
            _delay_ms(250);
        }
    }
}
```

Θα εξηγήσουμε συντομία την λειτουργία του προγράμματος. Οι συναρτήσεις είναι ίδιες με αυτές του *Ζητήματος 6.1* και δεν θα ξαναβρεθούμε σε αυτές. Το πρόγραμμα αφού καθαρίσει την μνήμη ram του μικροελεγκτή, περιμένει να διαβάσει το πρώτο ψηφίο που θα πατηθεί. Εάν αυτό είναι το 6 τότε με τον ίδιο τρόπο διαβάζει και το δεύτερο ψηφίο το οποίο εάν ισούται με το 0, σημαίνει ότι κωδικός είναι σωστός και τα LED ανάβουν για 4 δευτερόλεπτα και περιμένουμε επιπλέον 1 δευτερόλεπτο (η εκφώνηση έλεγε πως μετά το πάτημα δύο ψηφίων το πρόγραμμα δεν θα πρέπει να δέχεται κανέναν αριθμό για 5 δευτερόλεπτα) μέχρι να ξεκινήσει πάλι το πρόγραμμα. Αντίθετα εάν κάποιο από τα δύο ψηφία είναι λάθος, τότε τα LED αναβοσβήνουν για 5 δευτερόλεπτα και το πρόγραμμα ξεκινάει από την αρχή, αφού είναι συνεχούς λειτουργίας.

Το διάγραμμα ροής φαίνεται παρακάτω: