Άσκηση δεύτερη του Εργαστηρίου

Στην δεύτερη αυτή άσκηση δίδονται η λειτουργία κάποιων εντολών σε γλώσσα C για τον C compiler της εταιρείας CCS και η επεξήγηση των αρχείων που ενσωματώνονται στα προγράμματα στα οποία καθορίζονται ο τρόπος λειτουργίας του PIC καθώς και η συχνότητα λειτουργίας του μικροελεγκτή.

Γράφοντας ένα πρόγραμμα σε γλώσσα C ο χρήστης έχει την αίσθηση της ανοικοδόμησης ενός σπιτιού. Συνδυάζοντας τις εντολές είναι σαν να συνδυάζει το τσιμέντο με την άμμο και φτιάχνει τις συναρτήσεις ή λειτουργίες (functions) αντίστοιχα με τα τούβλα. Κατόπιν συνδυάζοντας τις συναρτήσεις δημιουργείται το πρόγραμμα ή αντίστοιχα τοποθετώντας τα τούβλα χτίζονται οι τοίχοι και το σπίτι.

Κάθε πρόγραμμα σε γλώσσα C θα πρέπει να έχει τουλάχιστον μία λειτουργία με το όνομα main() η οποία είναι η κύρια ρουτίνα του προγράμματος και είναι το σημείο εκκίνησης της εκτέλεσης του προγράμματος. Όλες οι συναρτήσεις μπορούν να σχετίζονται με το κυρίως πρόγραμμα άμεσα ή έμμεσα. Παρόλο που κάποιες συναρτήσεις μπορούν από μόνες του να είναι πλήρη προγράμματα μπορούν κάποιες μεταβλητές να μοιράζονται με άλλες συναρτήσεις ή με το κυρίως πρόγραμμα (main). Πολλές φορές η main ρουτίνα είναι η μόνη στο πρόγραμμα και μπορεί να περιέχει μερικές δηλώσεις και αρχικοποιήσεις. Στην πιο απλή περίπτωση ένα πρόγραμμα θα έχει την παρακάτω μορφή.

```
void main ()
        {
        while (1)
        }
```

Το πρόγραμμα αυτό μπορεί να μεταφραστεί χωρίς κανένα πρόβλημα αλλά δεν κάνει τίποτε απλά βάζει το μικροελεγκτή σε ένα ατέρμων βρόχο.

Σε όλα τα προγράμματα του εργαστηρίου θα πρέπει στην πρώτη γραμμή να δηλωθούν στον compiler μια σειρά από στοιχεία που αφορούν τον μικροελεγκτή για τον οποίο γράφεται το πρόγραμμα μιας και κάθε μικροελεγκτής περιέχει και διαφορετικό υλικό κομμάτι (hardware) παράδειγμα δεν περιέχουν όλοι οι μικροελεγκτές πόρτα USB.

Το αρχείο αυτό έχει το όνομα main.h είναι δηλαδή ένα header file και όπως θα φανεί αργότερα στην C υπάρχουν πολλά τέτοια αρχεία επικεφαλίδας που επιτελούν διάφορες εργασίες και μπορούν να έχουν την έννοια της βιβλιοθήκης. Θα πρέπει να τονιστεί ότι ένα από τα πλεονεκτήματα της γλώσσας C είναι η ύπαρξη αυτών των βιβλιοθηκών.

Στο header αρχείο main.h καθορίζεται αρχικά ο τύπος του μικροελεγκτή που θα χρησιμοποιηθεί και περιέχεται στην αναπτυξιακή πλακέτα που είναι ο PIC 18F4550 το υλικό κομμάτι του οποίου εξετάζεται στο μάθημα των μικρουπολογιστών Ι . Για τον καθορισμό των καταχωρητών του PIC μέσα στο αρχείο 18F4550.h υπάρχουν μια σειρά από ισοδυναμίες που αντιστοιχίζουν

τα ονόματα των καταχωρητών με τις πραγματικές τους θέσεις στην μνήμη καθώς επίσης και ο ορισμός διαφόρων συναρτήσεων που υπάρχουν στον compiler αυτό και η σωστή σύνταξη τους σε σχόλια. Επίσης στο αρχείο αυτό (main.h) περιέχεται ο ορισμός των ασφαλειών που καθορίζουν τον τρόπο λειτουργίας του PIC. Μερικές από τις ασφάλειες αυτές υπάρχουν σε όλους τους μικροελεγκτές της σειράς και κάποιες αφορούν μόνο τον συγκεκριμένο μικροελεγκτή. Οι πιο πολύ χρησιμοποιούμενες ασφάλειες είναι αυτές που καθορίζουν τον τρόπο λειτουργίας του ταλαντωτή (RC,XT,HS,HSPLL,INTRC) . Στην αναπτυξιακή πλακέτα για την λειτουργία του μικροελεγκτή χρησιμοποιείται κρύσταλλος υψηλής συχνότητας με ενεργοποιημένο τον εσωτερικό πολλαπλασιαστή συχνότητας PLL και η επιλογή είναι HSPLL. Μία άλλη ασφάλεια που είναι σημαντική είναι αυτή που καθορίζει την λειτουργία του WDT (Watch Dog Timer) που αφορά την λειτουργία ενός μετρητή επιτήρησης του συστήματος και στην εφαρμογή αυτή είναι απενεργοποιημένη. Μια άλλη ασφάλεια που υπάρχει σε όλους τους μικροελεγκτές της σειράς είναι η PUT (Power Up Timer) και αφορά το ξεκίνημα του μικροελεγκτή στην συγκεκριμένη εφαρμογή είναι απενεργοποιημένη. Αυτό που πρέπει να γνωρίζει ο χρήστης είναι ότι δεν πρέπει να αλλάξει τις ασφάλειες αυτές διότι από αυτές εξαρτάται η σωστή λειτουργία του boot-loader. Εκτός από τις ασφάλειες στο main.h αρχείο ορίζεται η συχνότητα λειτουργίας του μικροελεγκτή που είναι 48MHz από την οποία καθορίζονται οι συναρτήσεις χρονοκαθυστέρησης (Delays). Επίσης στην main.h ορίζεται η διεύθυνση αρχής του προγράμματος και οι διευθύνσεις διακλάδωσης των διακοπών (interrupts) που μετακινούνται λόγω της ύπαρξης του προγράμματος του boot-loader στις διευθύνσεις 0x800, 0x808 και 0x818 αντίστοιχα με τις κανονικές διευθύνσεις 0x000, 0x008, 0x018.

Το main.h header αρχείο ενσωματώνεται στο πρόγραμμα με την εντολή

include <main.h> η εντολή #include είναι εντολή στον preprocessor του C compiler και δηλώνει ακριβώς την ενσωμάτωση του αρχείου main.h στο πρόγραμμα του χρήστη που ακολουθεί. Όλες οι εντολές προς τον preprocessor ξεκινούν με το σύμβολο της δίεσης # και δεν χρειάζονται το Ελληνικό ερωτηματικό στο τέλος τους.

Ένα άλλο κομμάτι που αποτελεί αναπόσπαστο μέρος των προγραμμάτων σε C είναι η δήλωση των μεταβλητών η δήλωση των σταθερών, και η δήλωση των συναρτήσεων variable definition , constants declaration , function declaration.

Η δήλωση των μεταβλητών όταν γίνεται στην αρχή πριν από την συνάρτηση main και έξω από κάθε άλλη συνάρτηση πρόκειται για μια γενική μεταβλητή (global variable) αυτού του τύπου μεταβλητές είναι διαθέσιμες οπουδήποτε στο πρόγραμμα και δεσμεύουν ανάλογα με τον τύπο τους χώρο της μνήμης δεδομένων (Ram). Όταν οι μεταβλητές ορίζονται μέσα σε συναρτήσεις οι μεταβλητές αυτές είναι διαθέσιμες τοπικά μέσα στην συνάρτηση που δηλώθηκαν και δεν είναι διαθέσιμες στο υπόλοιπο πρόγραμμα. Οι μεταβλητές αυτές ονομάζονται τοπικές (local) και ελευθερώνουν τις θέσεις μνήμης μετά την χρήση τους. Οι διάφοροι τύποι μεταβλητών που είναι διαθέσιμες στο CCS

C Compiler καθώς και το μέγεθος τους στην μνήμη δίδονται παρακάτω σε συσχετισμό με τις μεταβλητές που υπάρχουν στην standard C.

Basic Types

Type-Specifier	Size	Range Unsigned	Signed	Digits
int1	1 bit number	0 to 1	N/A	1/2
int8	8 bit number	0 to 255	-128 to 127	2-3
int16	16 bit number	0 to 65535	-32768 to 32767	4-5
int32	32 bit number	0 to 4294967295	-2147483648 to 2147483647	9-10
int48	48 bit number	0 to 281474976710655	-140737488355328 to 140737488355327	14-15
int64	64 bit number	N/A	-9223372036854775808 to 9223372036854775807	18-19
float32	32 bit float	-1.5×10^{45} to 3.4×10^{38}		7-8

C Standard Type	Default Type
short	int1
char	unsigned int8
int	int8
long	int16
long long	int32
float	float32
double	N/A

Οι τελεστές (Operators)

Οι τελεστές είναι σύμβολα που χρησιμοποιούνται στην γλώσσα C και παριστούν διεργασίες μεταξύ διαφόρων τύπων δεδομένων. Στο CCS C compiler δίδονται στον παρακάτω πίνακα με επεξήγηση της λειτουργίας τους.

Operators

+	Addition Operator
+=	Addition assignment operator, x+=y, is the same as x=x+y
&=	Bitwise and assignment operator, x&=y, is the same as x=x&y
&	Address operator
&	Bitwise and operator

^=	Bitwise exclusive or assignment operator, x^=y, is the same as x=x^y
^	Bitwise exclusive or operator
|=	Bitwise inclusive or assignment operator, x|=y, is the same as x=x|y
|	Bitwise inclusive or operator
?:	Conditional Expression operator
- -	Decrement
/=	Division assignment operator, x/=y, is the same as x=x/y
/	Division operator
==	Equality
>	Greater than operator
>=	Greater than or equal to operator
++	Increment
*	Indirection operator
!=	Inequality
<<=	Left shift assignment operator, x<<=y, is the same as x=x<<y
<	Less than operator
<<	Left Shift operator
<=	Less than or equal to operator
&&	Logical AND operator
!	Logical negation operator
||	Logical OR operator
%=	Modules assignment operator x%=y, is the same as x=x%y
%	Modules operator
*=	Multiplication assignment operator, x*=y, is the same as x=x*y
*	Multiplication operator
~	One's complement operator
>>=	Right shift assignment, x>>=y, is the same as x=x>>y
>>	Right shift operator
->	Structure Pointer operation
-=	Subtraction assignment operator, x-=y, is the same as x=x-y
-	Subtraction operator
sizeof	Determines size in bytes of operand

Ο ορισμός μιας σταθεράς μπορεί να γίνει την εντολή constant
Παράδειγμα const float PI =3.14; , ή μπορεί να γραφεί με την εντολή define π.χ. #define clock PIN_B5 .
Οι δήλωση των συναρτήσεων γίνεται πριν από την main και γράφεται ο ορισμός της συνάρτησης με το ελληνικό ερωτηματικό στο τέλος ;
Παράδειγμα

void init (void); // Function declaration

Ιδιαίτερη προσοχή πρέπει να δίδεται στα ονόματα των μεταβλητών και συναρτήσεων δεν πρέπει να θυμίζουν εντολές της C όπως οι λέξεις που υπάρχουν στον παρακάτω πίνακα.

auto	default	for	int32	return	union
break	defined	goto	register	short	unsigned
bit	do	if	return	signed	void
byte	double	inline	short	sizeof	volatile
case	else	int	signed	static	while
char	enum	int1	sizeof	struct	
const	extern	int8	long	switch	
continue	float	int16	register	typedef	

Πίνακας δεσμευμένων λέξεων.

Οι έτοιμες συναρτήσεις για είσοδο και έξοδο από τις πόρτες του μικροελεγκτή.
Συναρτήσεις εξόδου.
output_x(value); Στην εντολή αυτή το X μπορεί να είναι μία οποιαδήποτε πόρτα όπως a,b,c,d,e. Η τιμή (value) μπορεί είναι ένας αριθμός μεταξύ 0,255. παράδειγμα output_b(0xff); Η εντολή βγάζει στην πόρτα B τον αριθμό 0xff.
output_bit(pin,value); Στην εντολή το pin είναι κάποιο από τα pins κάποιας πόρτας και value μπορεί να είναι 0 ή 1. Παράδειγμα
output_bit(PIN_B5,1); Με την εντολή αυτή τίθεται το pin 5 από την πόρτα B σε λογικό1.
output_float(pin); Με την εντολή αυτή τίθεται το pin σε κατάσταση εισόδου με υψηλή αντίσταση. Παράδειγμα
output_float(PIN_B0); Με την εντολή αυτή το pin 0 της πόρτας B τίθεται σε κατάσταση υψηλής αντίστασης.
output_high(pin); Με την εντολή αυτή κάποιο pin από κάποια πόρτα τίθεται σε λογικό 1 (5V). Παράδειγμα
output_high(PIN_B2); Με την εντολή αυτή το pin 2 της πόρτας B τίθεται σε λογικό 1.
output_low(pin); Με την εντολή αυτή κάποιο pin από κάποια πόρτα τίθεται σε λογικό 0 (0V). Παράδειγμα
output_low(PIN_B3); Με την εντολή αυτή το pin 3 της πόρτας B τίθεται σε λογικό 0.
output_toggle(pin); Με την εντολή αυτή κάποιο pin από κάποια πόρτα αλλάζει κατάσταση, (αντιστρέφεται) αν ήταν σε λογικό 0 τίθεται σε λογικό 1 και το αντίστροφο. Παράδειγμα

output_toggle(PIN_B3); Με την εντολή αυτή το pin 3 της πόρτας B αλλάζει λογική κατάσταση (αντιστρέφεται).

Συναρτήσεις εισόδου.

input(pin); Με την εντολή αυτή κάποιο pin από κάποια πόρτα ελέγχεται και αν είναι σε λογικό 1 επιστέφει την τιμή 1 (true) ενώ αν είναι σε λογικό 0 επιστρέφει 0 (false). Παράδειγμα

```
if  (input(PIN_A0) ){
        output_high(PIN_B0)
                        }
    else  {
        output_low(PIN_B0)
          }
```

Στο παράδειγμα αυτό ελέγχεται το pin 0 από την πόρτα A και είναι σε λογικό 1 τότε θέτει το pin 0 της πόρτας B σε λογικό 1 διαφορετικά θέτει το pin 0 της πόρτας B σε λογικό 0.

input_change_x(); Στην εντολή αυτή ελέγχεται η πόρτα x όπου x = a,b,c,d,e και αν έχει αλλάξει η είσοδος της πόρτας σε σχέση με την προηγούμενη φορά που χρησιμοποιήθηκε η εντολή αυτή τότε επιστέφει λογικό 1 ενώ αν είναι δεν έχει αλλάξει η κατάσταση της πόρτας επιστέφει λογικό 0.

Παράδειγμα

check_port = input_change_a(); Αν την προηγούμενη φορά είχε η πόρτα τον αριθμό 4 και μετά την εκτέλεση της εντολής εξακολουθεί να έχει την ίδια τιμή τότε η μεταβλητή check_port παίρνει τιμή 0 διαφορετικά θα έχει τιμή 1. Η εντολή αυτή είναι πολύ χρήσιμη στις περιπτώσεις που θέλουμε να ελέγξουμε αλλαγές σε κάποια πόρτα χωρίς να διαβάζουμε διαδοχικά την πόρτα.

input_state(pin); Στην εντολή αυτή ελέγχετε ένα pin από κάποια πόρτα και αν αυτό είναι σε λογικό 0 επιστρέφει λογικό μηδέν ενώ αν το pin αυτό είναι σε λογικό 1 επιστρέφει λογικό 1. Η διαφορά αυτής της εντολής από την εντολή input(pin) είναι ότι για τον έλεγχο δεν αλλάζει την φορά του pin σε είσοδο.

Παράδειγμα

input_state(PIN_A1); Αν το pin 1 της πόρτας A είναι σε λογικό 0 επιστρέφει 0 διαφορετικά επιστέφει 1.

input_x(); Με την εντολή αυτή διαβάζεται το περιεχόμενο της πόρτας x και η τιμή που επιστρέφει είναι το περιεχόμενο της πόρτας x.

Παράδειγμα

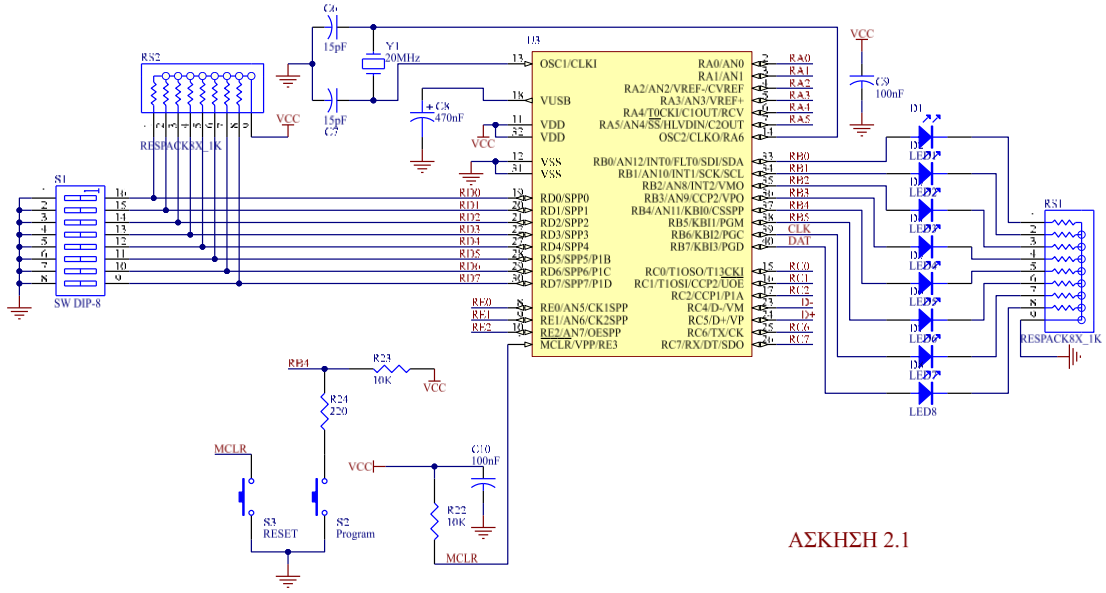
eisodos = input_a(); Η εντολή αυτή επιστρέφει στην μεταβλητή eisodos την τιμή εισόδου της πόρτας A. Η εντολή αυτή μπορεί να αντικαταστήσει την εντολή eisodos = PORTA; όπου θα πρέπει να οριστεί η διεύθυνση της πόρτας A με την εντολή #byte PORTA = 0xf80

Άσκηση

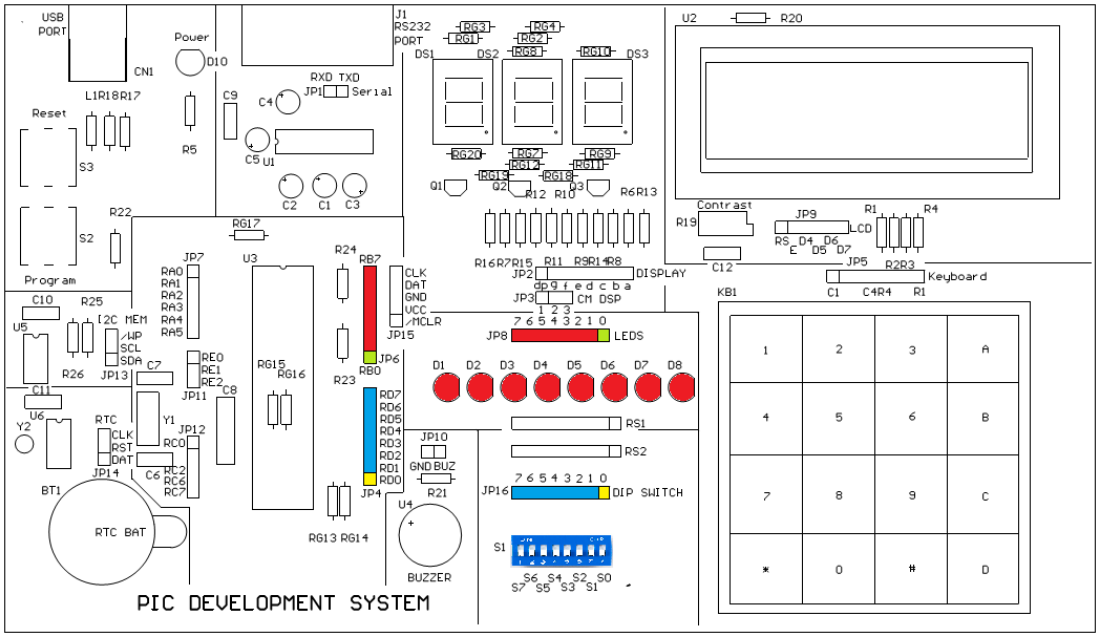
Στην παρακάτω άσκηση συνδέονται η πόρτα B στα Leds της πλακέτας και στην πόρτα D τα Dip Switches.

Ο σκοπός της άσκησης είναι η ανάγνωση από την πόρτα D Dip switches και ο καθορισμός του τρόπου που θα αναβοσβήνουν τα Leds που συνδέονται στην πόρτα B. Στην άσκηση αυτή γίνεται χρήση υποπρογραμμάτων και πινάκων.

Το σχηματικό της εφαρμογής.



Οι συνδέσεις με τις καλωδιοταινίες της πλακέτας του εργαστηρίου



Το πρόγραμμα της εφαρμογής

```
#include <main.h>

#use standard_io ( A )   // Standard είσοδοι και έξοδοι
#use standard_io ( B)    // Standard είσοδοι και // έξοδοι
#use standard_io ( C )   // Standard είσοδοι και // έξοδοι

#byte PORTA        =0xF80  ορισμός των θυρών με την θέση τους στην μνήμη
#byte PORTB        =0xF81
#byte PORTC        =0xF82
#byte PORTD        =0xF83
#byte PORTE        =0xF84
                              // Variables Definition
int8 pattern = 0;
int8 leds[8] = {1,2,4,8,16,32,64,128}; //Ορισμός πίνακα με 8 τιμές

// Δήλωση συναρτήσεων

void pattern1(void);
void pattern2(void);
void pattern3(void);

void main()
{
     set_tris_b(0x00);     // Καθορισμός της πόρτας B ως έξοδος
     set_tris_d(0xff);     //  Καθορισμός της πόρτας D ως είσοδος
     while (1){             //  Εκτέλεσε συνεχώς ότι υπάρχει από το άγκιστρο και
                           //   κάτω
     pattern = PORTD;
     if (pattern == 0x00) {
                            pattern1(); // κλήση υπορουτίνας
                            }
     if (pattern == 0x0f) {
                            pattern2();// κλήση υπορουτίνας
                            }
     if (pattern == 0xff) {
                           pattern3();// κλήση υπορουτίνας
                           }
          }
}
// Ορισμός συναρτήσεων
void pattern1(void){                  // Υπορουτίνα του pattern1
                        int8 i;
                        for (i=0;i<8;i++){
                                          PORTB= leds[i];
```

```
                                    delay_ms(500);
                                   }
                    }
void pattern2(void){                // Υπορουτίνα του pattern2
                    int8 i;
                    for (i=8;i>0;i--){
                                   PORTB= leds[i-1];
                                   delay_ms(500);
                                   }
                    }

void pattern3(void){                // Υπορουτίνα του pattern3
                    int8 i;
                    for (i=0;i<4;i++){
                                    PORTB= leds[2*i];
                                    delay_ms(500);
                                    }
                   }
```

Ασκήσεις που θα πρέπει να ετοιμαστούν για το δεύτερο εργαστήριο

1. Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για τον μικροελεγκτή PIC18F4550 που να λειτουργεί τον μικροελεγκτή ως ελεγκτή πλυντηρίου. Το σύστημα θα λειτουργεί ως εξής.

α. Αρχικά ελέγχει το μπουτόν start που συνδέεται στο RD7 και αν αυτό πατηθεί (Τεθεί σε λογικό 0) τότε το πρόγραμμα προχωρά στο επόμενο στάδιο διαφορετικά περιμένει εδώ.

β. Κατόπιν θα πρέπει να ελεγχθεί η πόρτα του κάδου που συνδέεται στο RD0 και θα πρέπει να είναι κλειστή δηλαδή σε λογικό1.

γ. Μετά από χρόνο 2 sec ανοίγει η βαλβίδα του νερού που συνδέεται στο RB1 θέτοντας το RB1 σε λογικό 1.

δ. Κατόπιν ελέγχεται το RD1 συνεχώς και όταν αυτό τεθεί σε λογικό 0 (γέμισε ο κάδος με νερό) τότε κλείνει την βαλβίδα του νερού θέτοντας το RB1 σε λογικό 0.

ε. Μετά από χρόνο 2 sec ανάβει η αντίσταση θέρμανσης του νερού θέτοντας το RB2 σε λογικό 1.

ζ. Μετά από 3 sec αρχίσει η κίνηση του κάδου αριστερά δεξιά θέτοντας εναλλάξ το RB3 και RB4 σε λογικό 1 με ενδιάμεσους χρόνους 3 sec (όταν ενεργοποιείται η κίνηση προς αριστερά απενεργοποιείται η κίνηση προς τα δεξιά και το ανάποδο) η κίνηση αυτή γίνεται για 10 φορές.

η. Κατόπιν σβήνει η αντίσταση θέρμανσης (RB2 σε λογικό 0)

θ. Κατόπιν και μετά από 3 sec ενεργοποιείται η αντλία εξόδου του νερού θέτοντας το RB5 σε λογικό 1 για χρόνο 5 sec και κατόπιν σε λογικό 0.

ι. Το πρόγραμμα επιστρέφει στην αρχή και περιμένει το πάτημα του κουπιού start.

2, Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για τον μικροελεγκτή PIC18F4550 που να διαβάζει από την πόρτα D ένα αριθμό από τα 4 περισσότερο σημαντικά bits και να αναβοσβήνει τα Leds που είναι συνδεδεμένα στην πόρτα B σε κινητή τελεία (ένα led κάθε φορά από αριστερά Led 0 προς τα δεξιά Led7 και επιστροφή πίσω με τον ίδιο τρόπο) τόσες φορές όσες ο αριθμός που έχει διαβάσει από την πόρτα D.

3. Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για τον μικροελεγκτή PIC18F4550 που να διαβάζει από την πόρτα B ένα αριθμό και να ελέγχει αν ο αριθμός έχει άρτιο αριθμό από 1 να θέτει το bit RD0 σε λογικό 1 και το RD1 σε λογικό 0, ενώ αν έχει περιττό αριθμό από 1 να θέτει το bit RD1 σε λογικό 1 και το RD0 σε λογικό 0. Στην πόρτα B συνδέονται τα διακοπτάκια (dip-switches) και στην πόρτα D τα leds.

Ερωτήσεις που πρέπει να απαντηθούν.

1. Πόσα bytes απαιτούνται για τον δυαδικό αριθμό 10110110101
   Α. 10
   Β. 1
   Γ. 2
   Δ. 1.3
   Ε. κανένα από τα παραπάνω.
2. Ένα δεκαεξαδικό ψηφίο με πιο από τους παρακάτω τρόπους μπορεί να περιγραφεί
   Α. Με δύο δεκαδικά ψηφία.
   Β. Με ένα byte.
   Γ. Με ένα Nibble.
   Δ. Με ένα οκταδικό ψηφίο.
   Ε. Με τρία δυαδικά ψηφία.
3. Ο δυαδικός αριθμός 0011 είναι στο δεκαεξαδικό ο αριθμός 0x03. Ποιος από τους παρακάτω είναι ο αριθμός 00110011.
   Α. 0x303
   Β. 0x33
   Γ. 0x06
   Δ. 0x36
   Ε. 0x3F

4. Ποιο είναι το λάθος στο παρακάτω πρόγραμμα

```
Const int a=10;
int b;
Int c;
c= b>a;
```

A. Το όνομα της μεταβλητής A δεν είναι με κεφαλαίο γράμμα.
B. Στην πρώτη εντολή υπάρχει το semicolon σύμβολο.
Γ. Το σύμβολο > δεν μπορεί να εφαρμοστεί στην δήλωση μετά το =
Δ. Όλες οι εντολές είναι λάθος.
Ε. Δεν υπάρχει κανένα λάθος.

5. Στο παρακάτω πρόγραμμα υπάρχουν λάθη. Να διορθωθούν

```
int a=0;
int b=2;
byte PORTB = 0x81
float c;
for(c=1;c<=10;c++){
a= b+c;
b=a;
PORTB = c;
}
```

6.Συμπληρώστε και διορθώστε το παρακάτω πρόγραμμα έτσι ώστε να είναι σωστό

```
include <main.h>
byte PORTB = 0x81

void main(void){

        while (s>1) (
                for(i=1;i<10;i+1){
                PORTB=i;
                }
        }
```