

Σκοπός του μαθήματος.

Το μάθημα των ενσωματωμένων συστημάτων II είναι συνέχεια του μαθήματος ενσωματωμένα συστήματα I και εξετάζει τους μικροελεγκτές περισσότερο από την πλευρά του προγράμματος. Η συγγραφή των προγραμμάτων γίνεται σε γλώσσα C αντί της assembly που γίνεται στο μάθημα των ενσωματωμένων συστημάτων I. Η γλώσσα C προσφέρει πολλά πλεονεκτήματα όπως εύκολη εποπτεία, εύκολη μεταφορά σε άλλο μικροελεγκτή, μικρό μέγεθος λίστας του προγράμματος, πληθώρα βιβλιοθηκών, εύκολη απομνημόνευση εντολών. Τα μειονεκτήματα της γλώσσας είναι το μεγαλύτερο μέγεθος στην μνήμη του προγράμματος και δεδομένων, μικρότερη ταχύτητα σε σχέση με την assembly.

Προαπαιτούμενα.

Απαιτούνται γνώσεις του hardware του μικροελεγκτή που είναι γνωστές από το μάθημα Ενσωματωμένων Συστημάτων I και γνώσεις στον προγραμματισμό με την γλώσσα C από τα μαθήματα Προγραμματισμός I και II.

Προγράμματα και Εργαλεία .

Στο μάθημα θα χρησιμοποιηθεί ο C compiler της εταιρείας CCS Version 5.01. Η τρέχουσα έκδοση είναι η 5.092. Η επιλογή του C Compiler έγινε με γνώμονα την πληθώρα των βιβλιοθηκών που διαθέτει και την ευκολία στην λειτουργία του είτε μέσω του δικού του editor, είτε μέσω του MPLAB στο οποίο μπορεί εύκολα να ενσωματωθεί.

Όπως τα περισσότερα προγράμματα εφαρμογών λειτουργεί με Projects δηλαδή όλα τα απαραίτητα αρχεία της εφαρμογής θα πρέπει να βρίσκονται σε ένα φάκελο όπου δημιουργείται το project. Αυτό είναι πολύ σημαντικό διότι διαφορετικά δημιουργούνται προβλήματα δυσλειτουργίας.

Παρακάτω περιγράφονται δύο διαφορετικοί τρόποι δημιουργίας Project. Ο πρώτος χρησιμοποιώντας σαν editor το MPLAB και καλώντας τον C compiler μέσα από το περιβάλλον αυτό ενώ ο δεύτερος χρησιμοποιεί τον editor της ίδιας εταιρείας με τον C compiler. Κάθε τρόπος έχει πλεονεκτήματα και μειονεκτήματα.

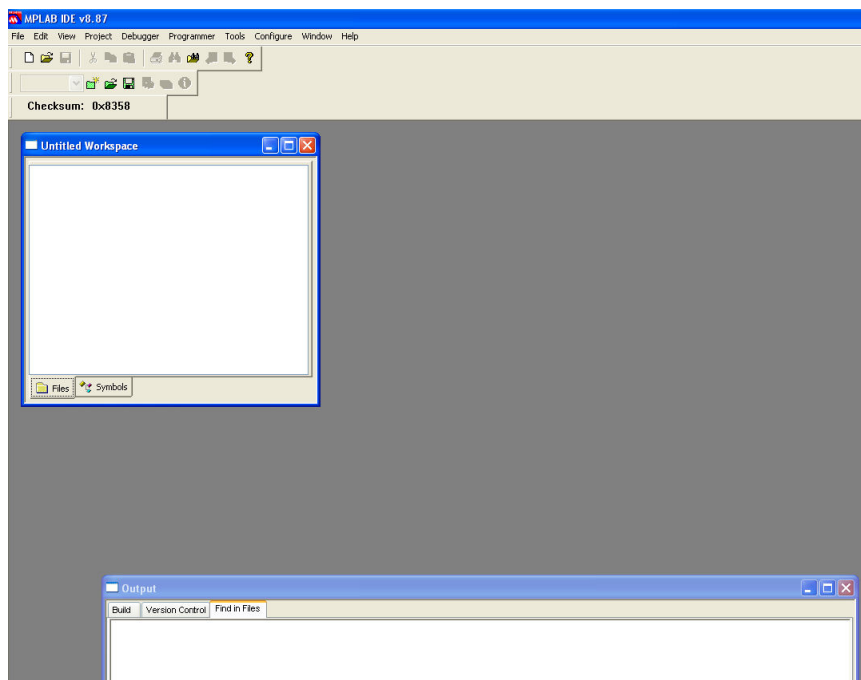
Δημιουργία νέου Project μέσα από το περιβάλλον εργασίας του MPLAB.

Η δημιουργία νέου Project μπορεί να γίνει

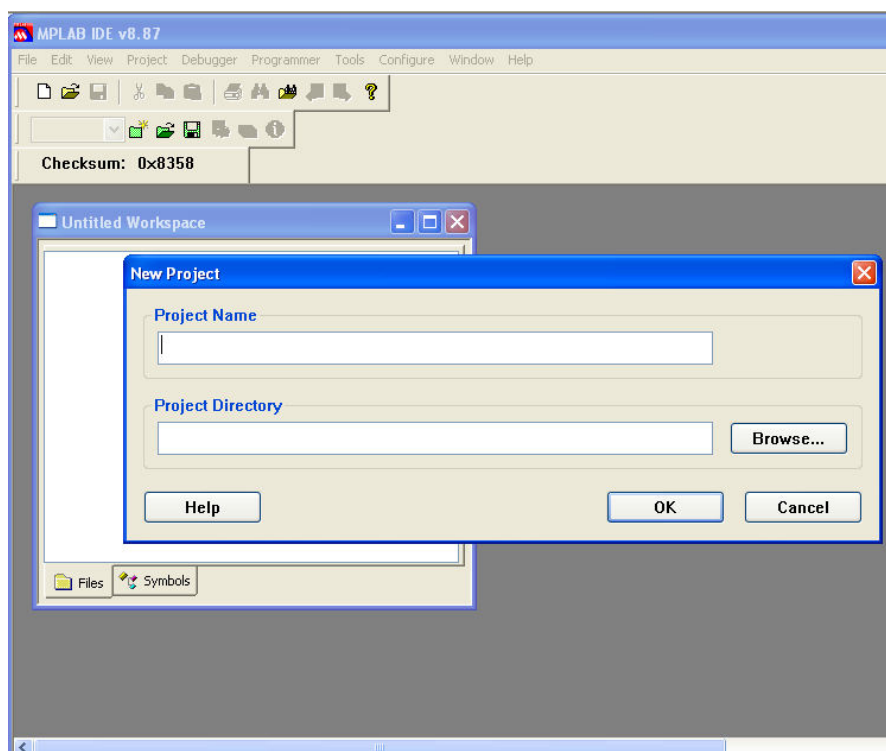
Ανοίγοντας το πρόγραμμα MPLAB (διπλό κλίκ στο εικονάκι στην επιφάνεια εργασίας).



Εργαστήριο στα Ενσωματωμένα Συστήματα – Άσκηση 1^η

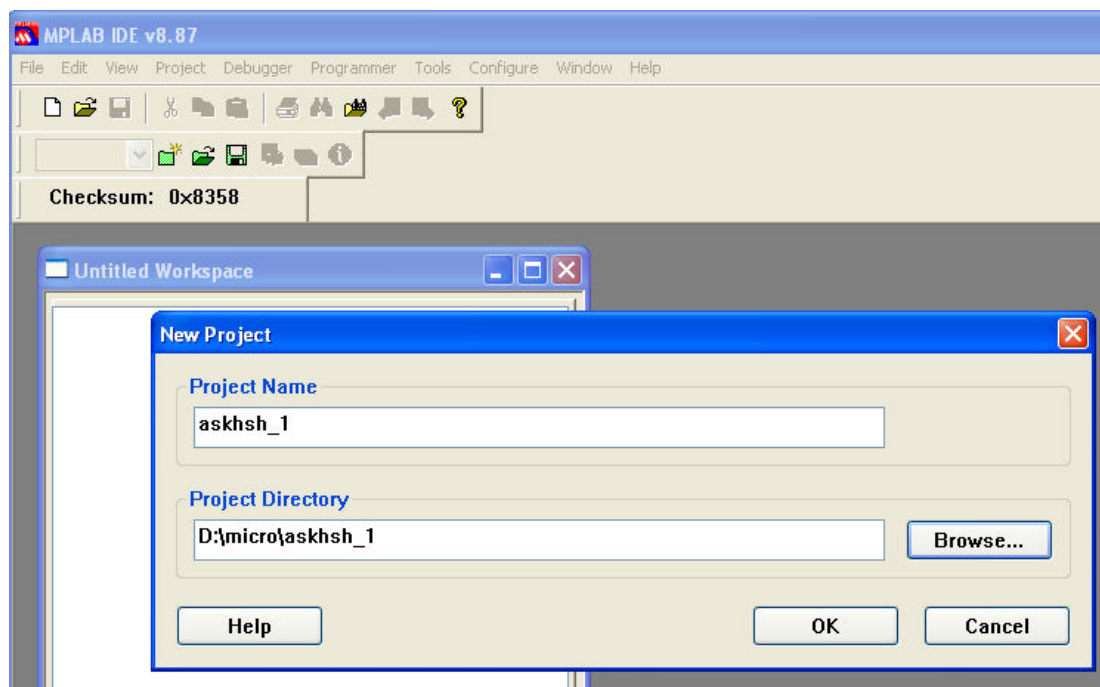


Από το παράθυρο Project επιλέγετε **New** και εμφανίζεται το παρακάτω παράθυρο

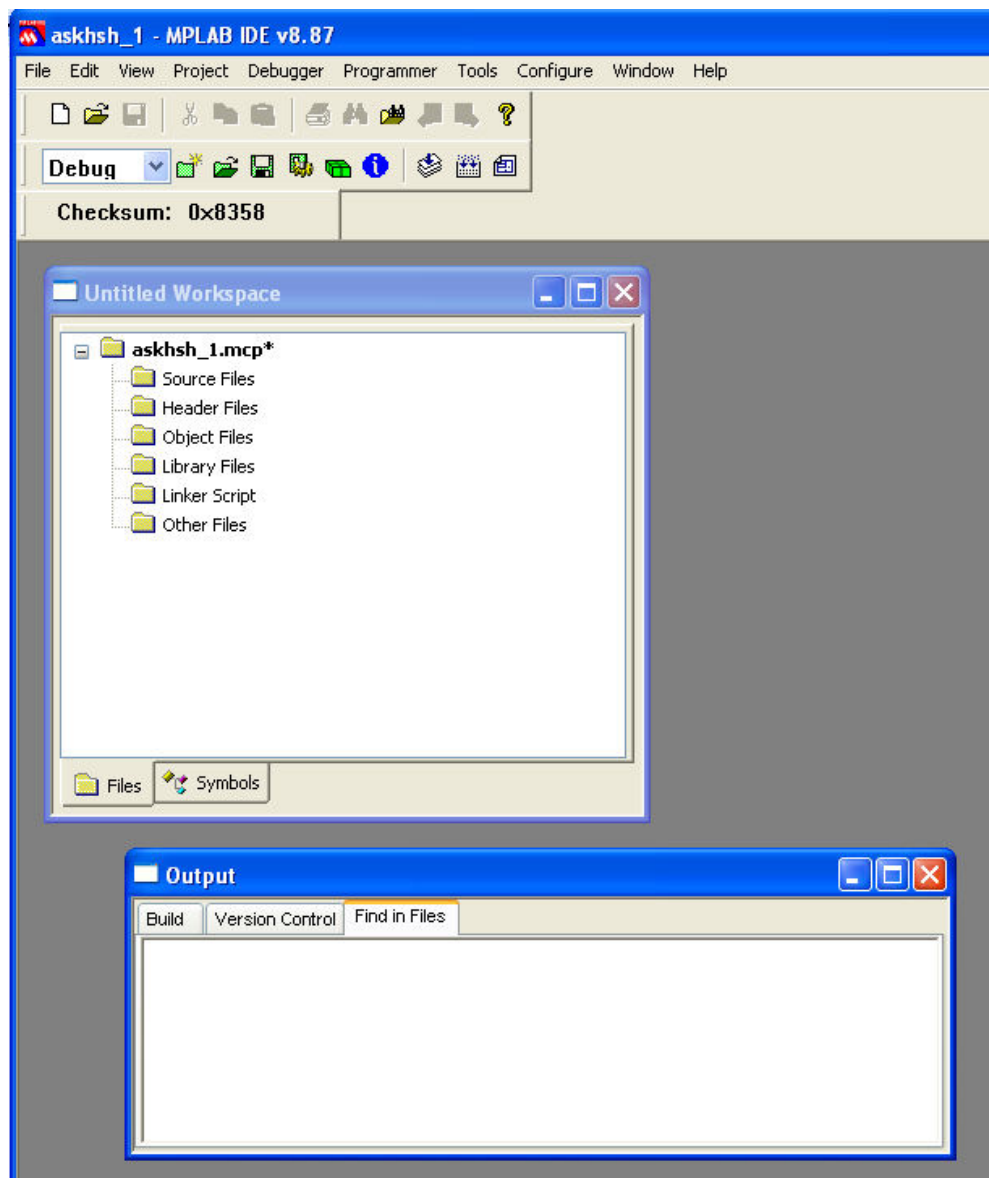


Στο πεδίο **Project Name** δίδετε ένα όνομα με λατινικούς χαρακτήρες και νούμερα αποφεύγοντας τα σημεία στίξης και σύμβολα, καθώς και τους Ελληνικούς χαρακτήρες.

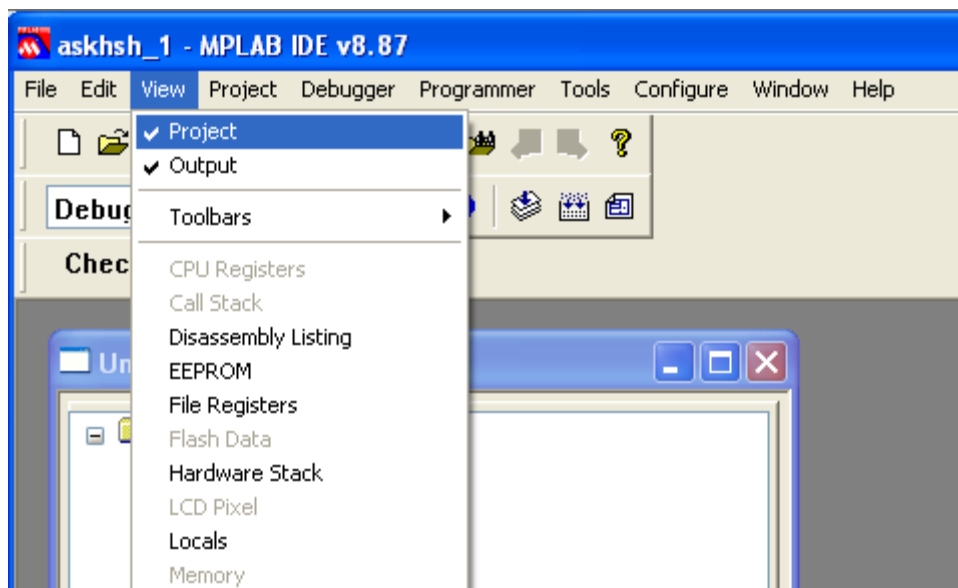
Στο πεδίο **Project Directory** από την επιλογή **Browse** δρομολογείτε την θέση όπου θα δημιουργηθεί το Project. Φροντίζετε να μην απαιτείται path μεγαλύτερο από 62 χαρακτήρες. Αποφεύγεται να δημιουργείτε το Project στην επιφάνεια εργασίας. Δημιουργήστε ένα νέο φάκελο στον τοπικό δίσκο όπου θα φυλάσσονται όλα τα Project του μαθήματος και μέσα στο φάκελο δημιουργήστε ένα υποφάκελο με όνομα Askhsh_1.



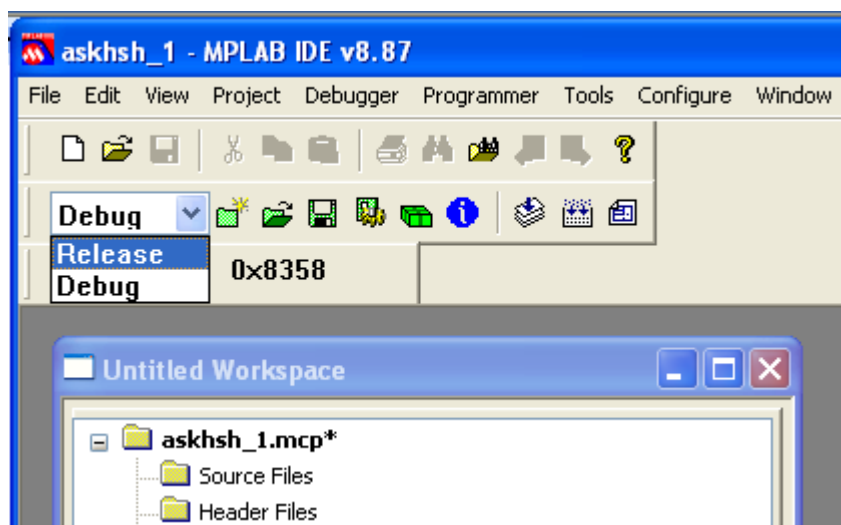
Κατόπιν επιλέγετε το **ok** και εμφανίζεται το παρακάτω παράθυρο



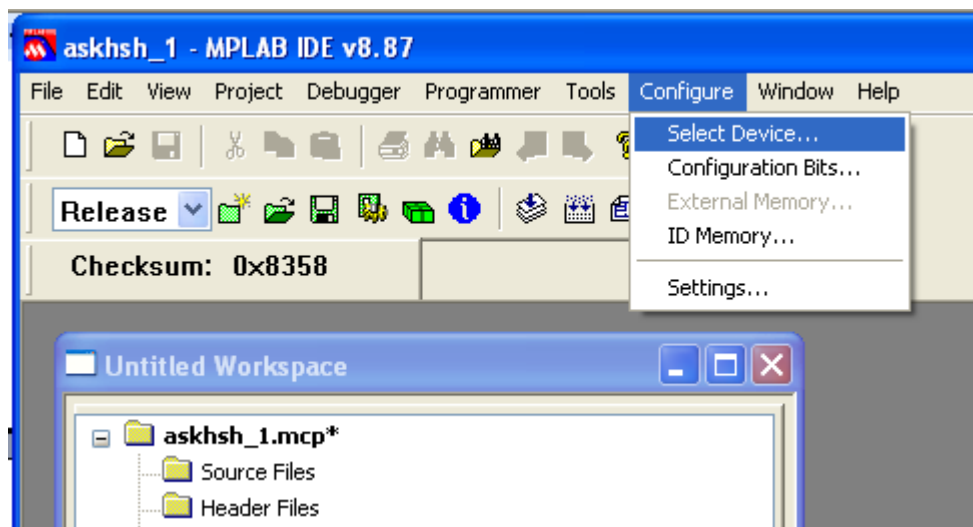
Στο παράθυρο αυτό φαίνονται όλοι οι φάκελοι του Project καθώς και το παράθυρο της εξόδου, που δημιουργούνται αυτόματα από το πρόγραμμα. Στην περίπτωση που δεν φαίνεται κάποιο από τα παραπάνω παράθυρα επιλέγετε από μενού **View** και από εκεί τα **Project** και **Output**. Όπως φαίνεται στο παρακάτω παράθυρο.



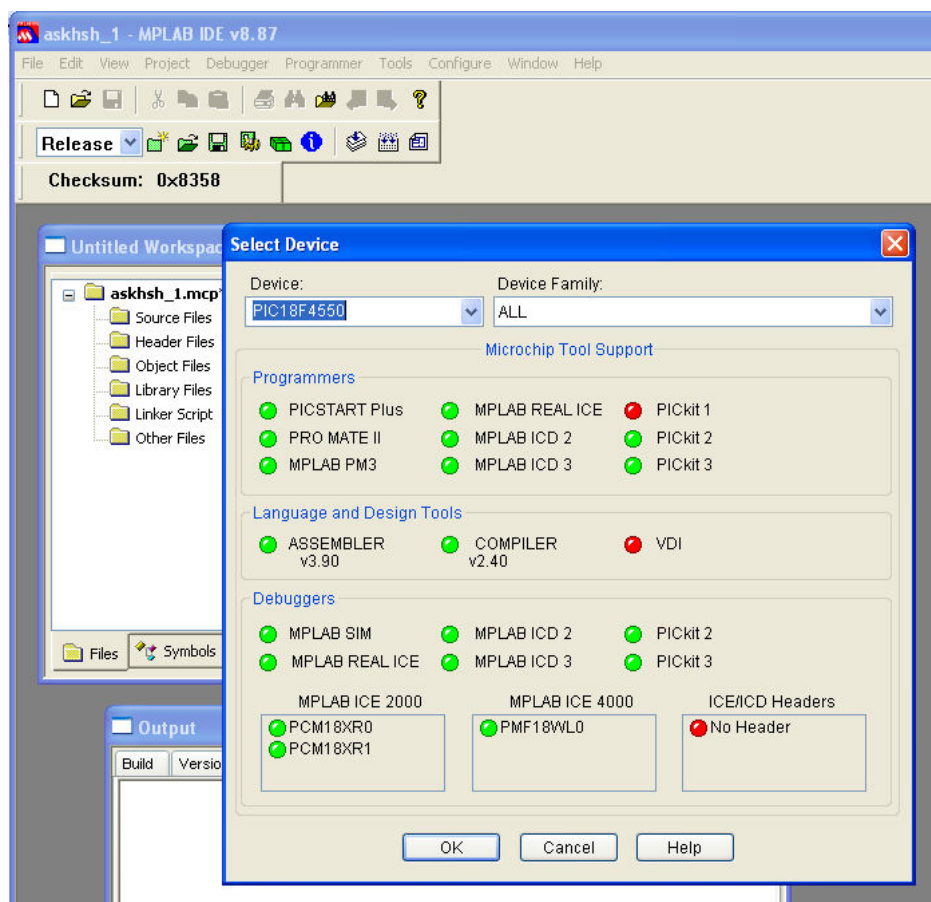
Από το μενού **Debug** και **Release** επιλέγετε το **Release**.



Κατόπιν από το κεντρικό μενού επιλέγετε **Configure** και κατόπιν **Select Device**.

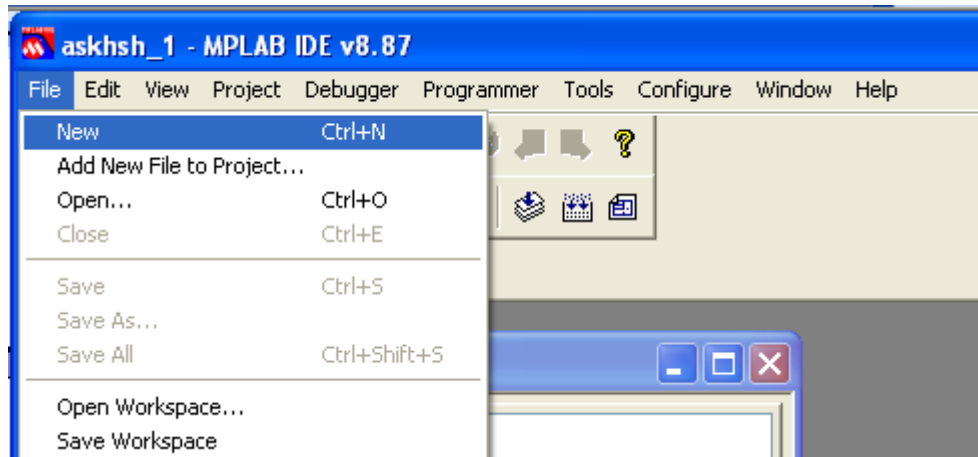


Από το μενού που εμφανίζεται επιλέγετε στο πεδίο **Device** τον **PIC18F4550**
Και κατόπιν **ok**.

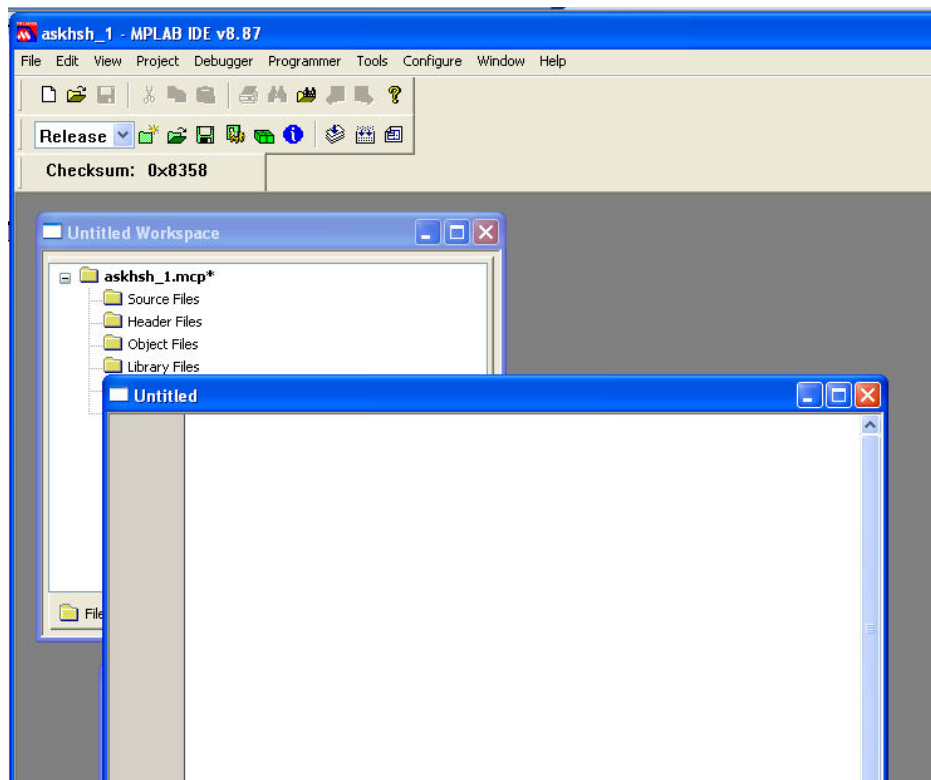


Μπορείτε να μετακινήσετε τα παράθυρα του Project και του Output κάνοντας κλικ και σύροντας το επιθυμητό παράθυρο με το αριστερό κουμπί του mouse

στην επιθυμητή θέση μέσα στο περιβάλλον του MPLAB. Για την δημιουργία ενός νέου πηγαίου κώδικα για την εφαρμογή επιλέγετε από το κεντρικό μενού **File** και κατόπιν **New**.



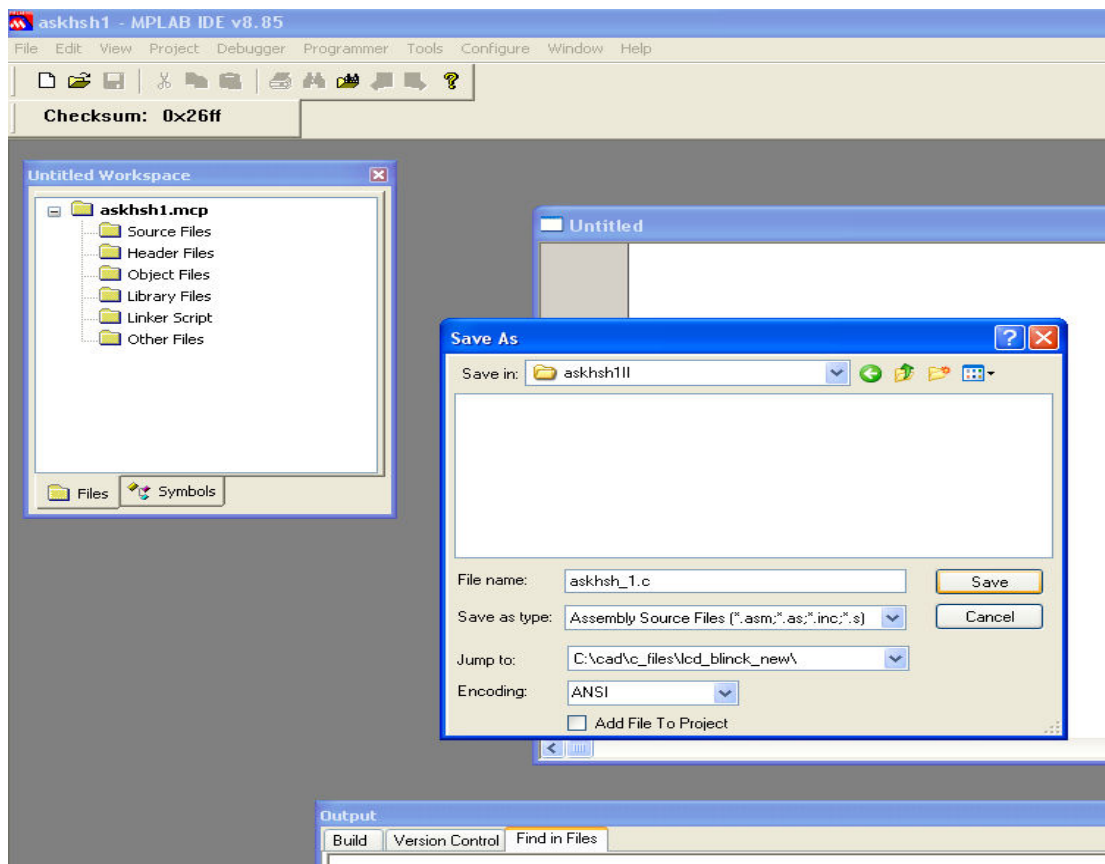
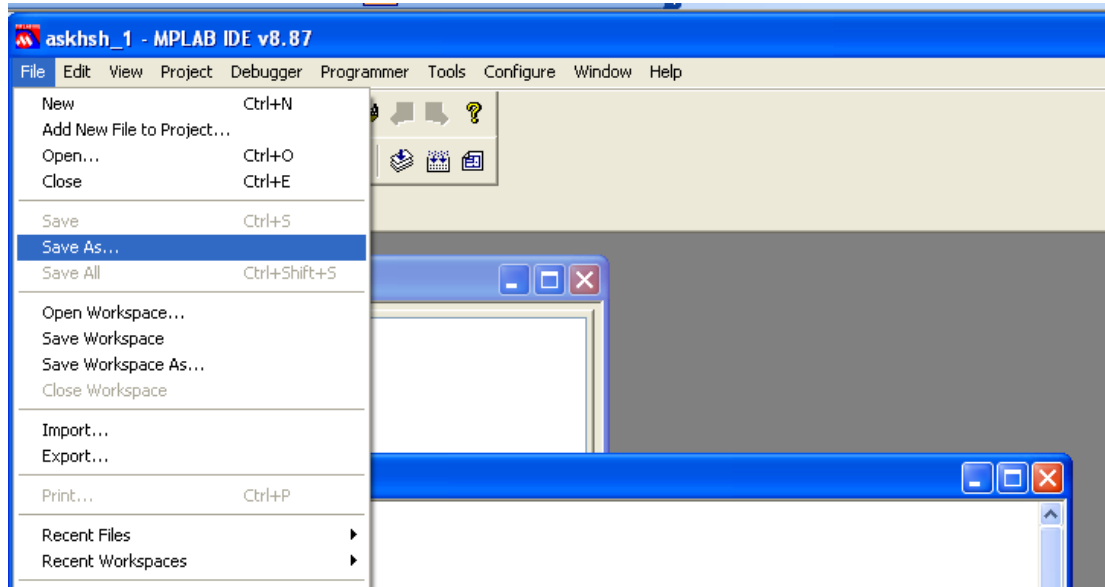
Εμφανίζεται ένα νέο παράθυρο με τίτλο untitled δηλαδή χωρίς όνομα και μέσα σε αυτό θα γραφεί το πρόγραμμα σε γλώσσα C.



Για να σώσετε το αρχείο αυτό επιλέγετε από το κεντρικό μενού **File** και κατόπιν **Save As** και στο νέο παράθυρο καθοδηγείτε το πρόγραμμα να σώσει το αρχείο στον ίδιο φάκελο όπου δημιουργήσατε το Project. Αυτό είναι πολύ

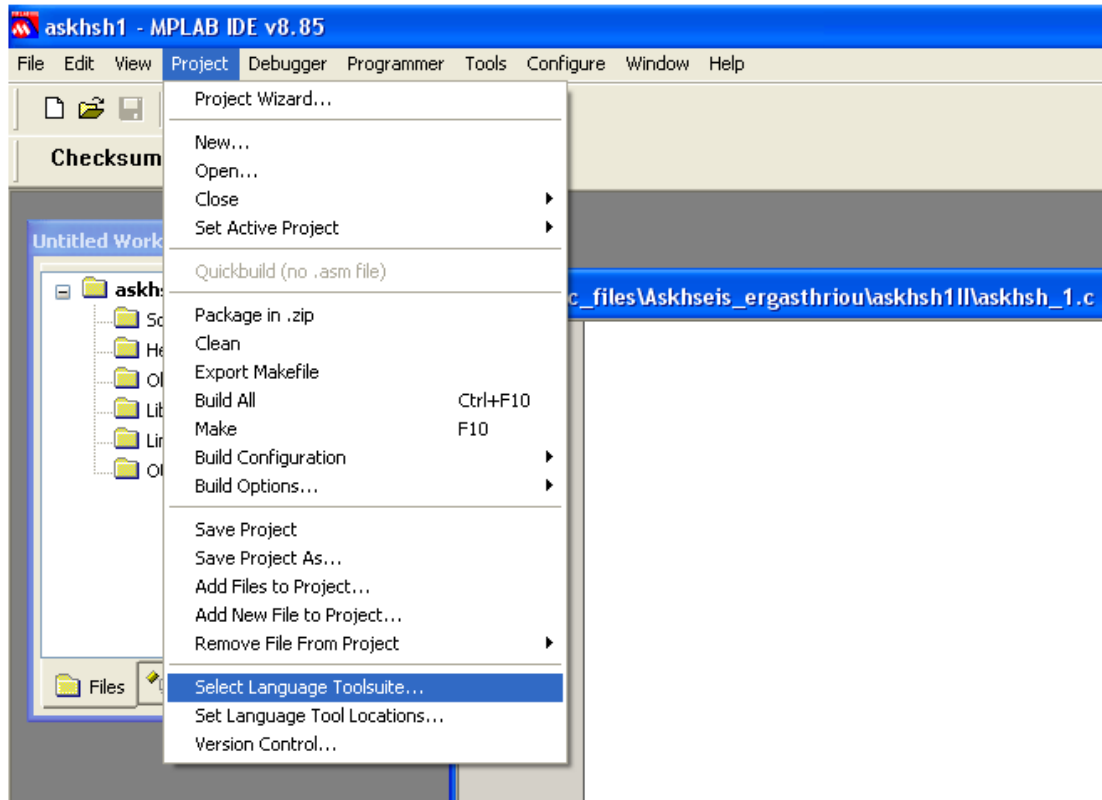
Εργαστήριο στα Ενσωματωμένα Συστήματα – Άσκηση 1^η

σημαντικό και χρήζει της προσοχής σας. Επίσης στο πεδίο **File Name** θα πρέπει να δώσετε ένα όνομα που θα έχει κατάληξη **c** (στο παράδειγμα askhsh_1.c)



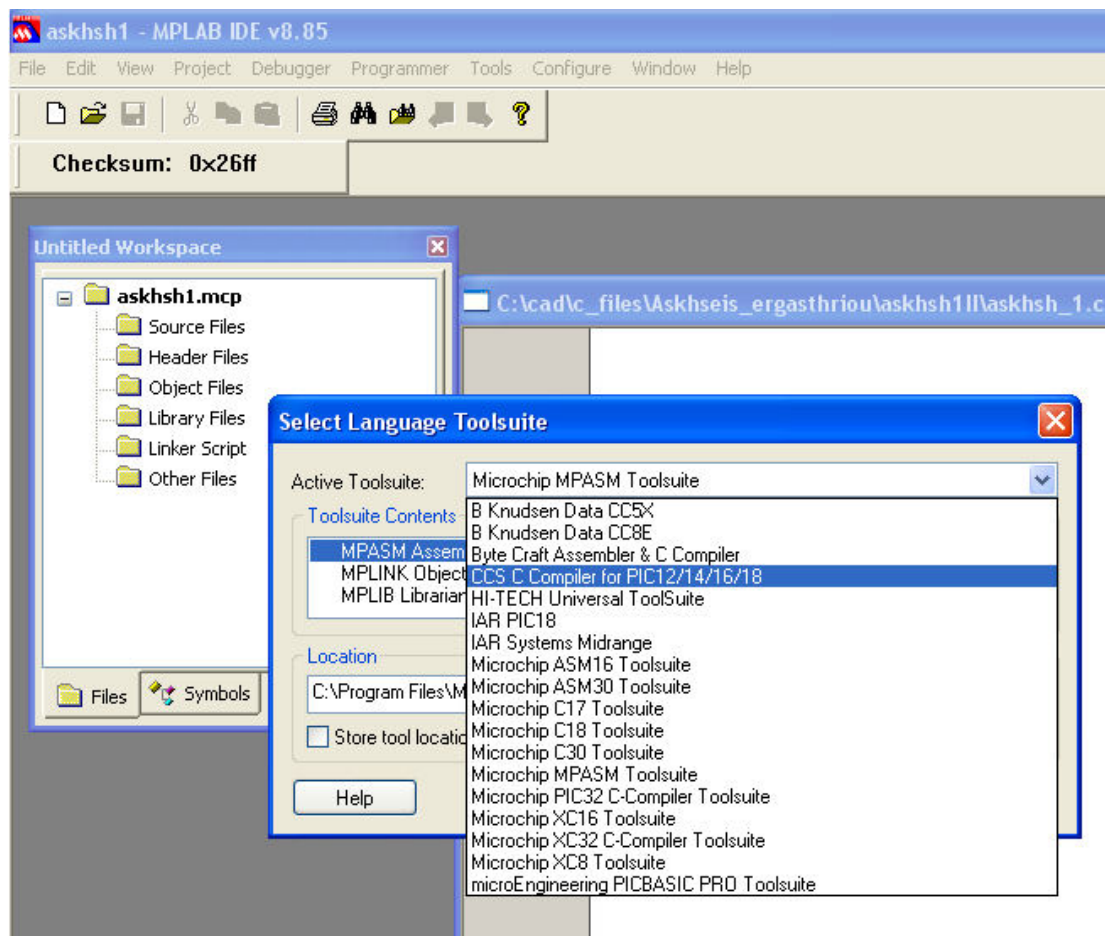
Κατόπιν επιλέγετε **Save** και η εφαρμογή σας έχει σωθεί στον ίδιο φάκελο με το Project.

Από την επιλογή **Project** επιλέγετε **Select Language Toolsuite**

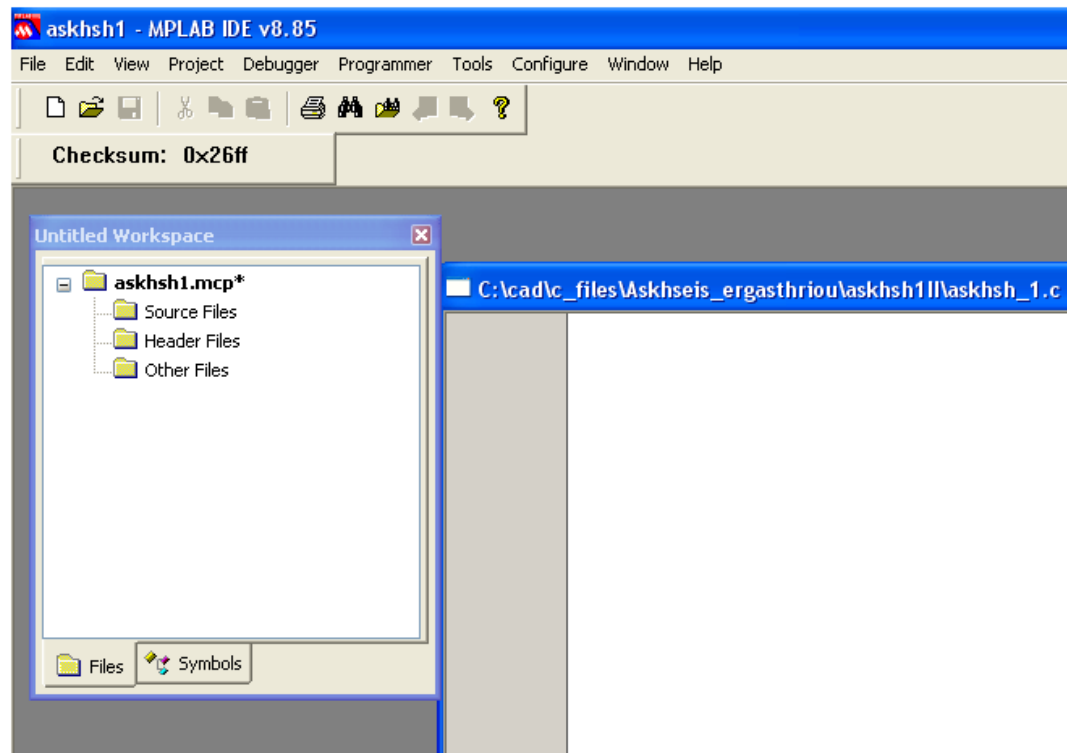


Από το παράθυρο που ανοίγει επιλέγετε από το πεδίο **Active Toolsuite** την επιλογή **CCS C Compiler for PIC 12/14/16/18**

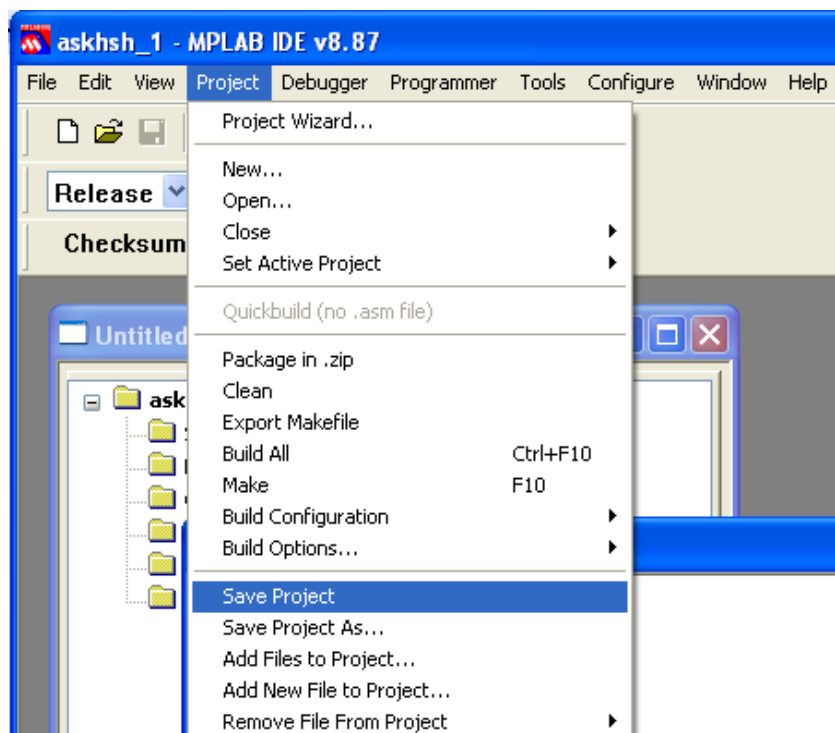
Εργαστήριο στα Ενσωματωμένα Συστήματα – Άσκηση 1^η



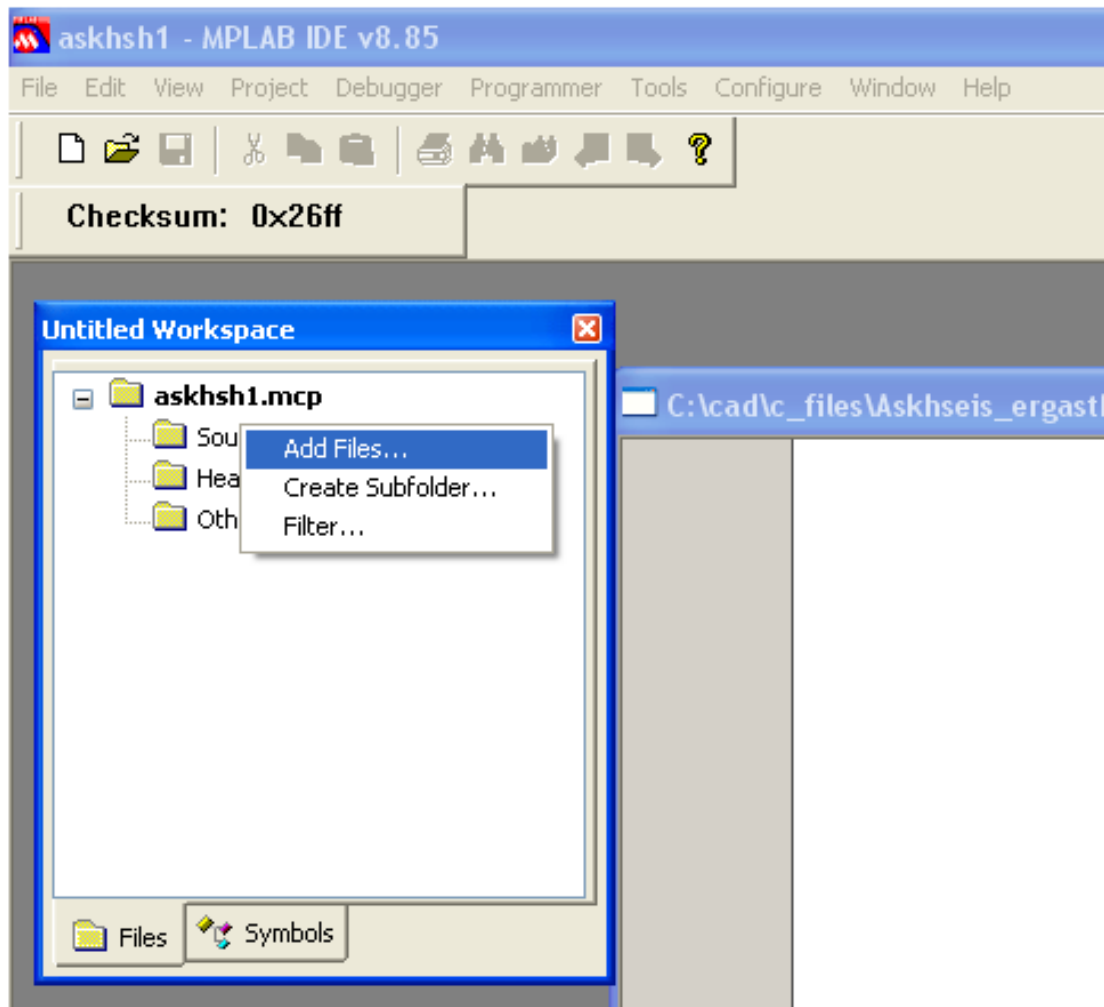
και κατόπιν επιλέγετε **ok**.



Όπως θα παρατηρήσετε στο όνομα του Project υπάρχει ένα αστεράκι αυτό σημαίνει ότι θα πρέπει να ξανασώσετε όλο το project και αυτό μπορεί να γίνει από το κεντρικό μενού από την επιλογή **Project** , **Save Project**.

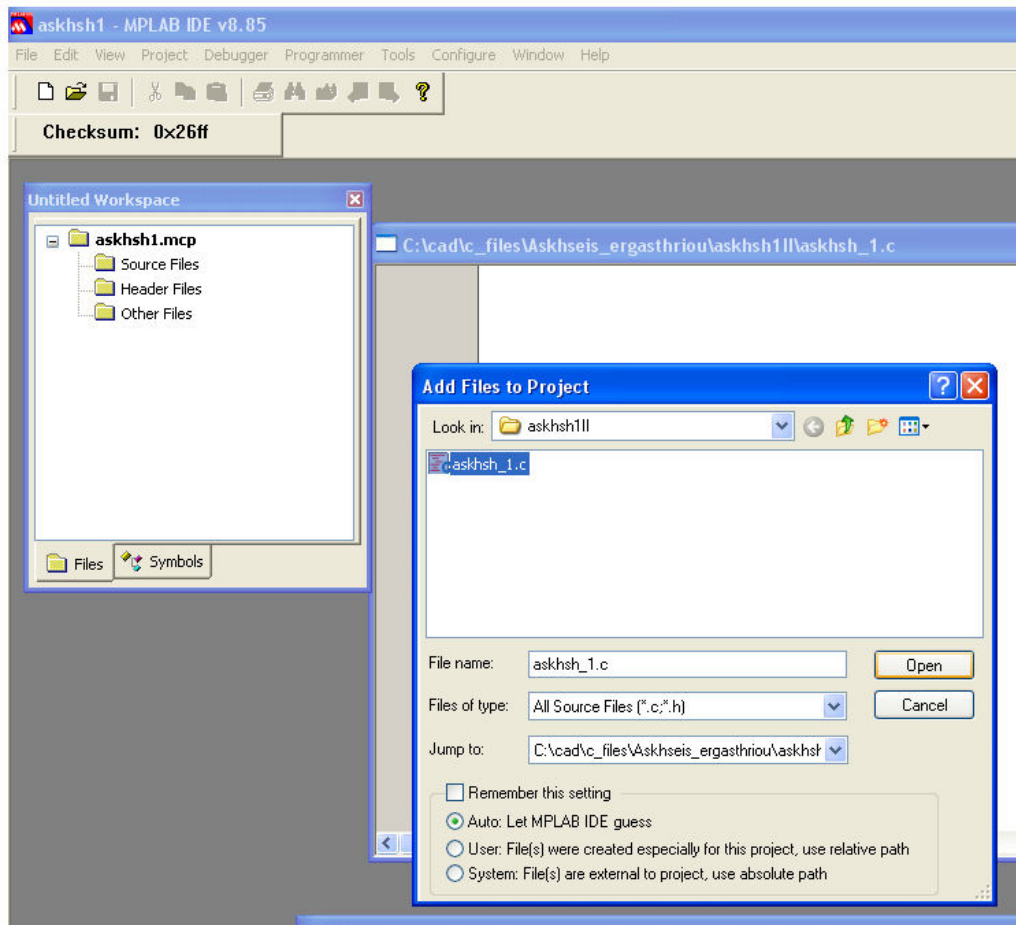


Για να προσθέσετε το πηγαίο αρχείο στο Project στο παράθυρο του project με το δεξί κουμπί του mouse επιλέγετε το φάκελο **Source Files** και κατόπιν με αριστερό επιλέγετε **Add Files**.

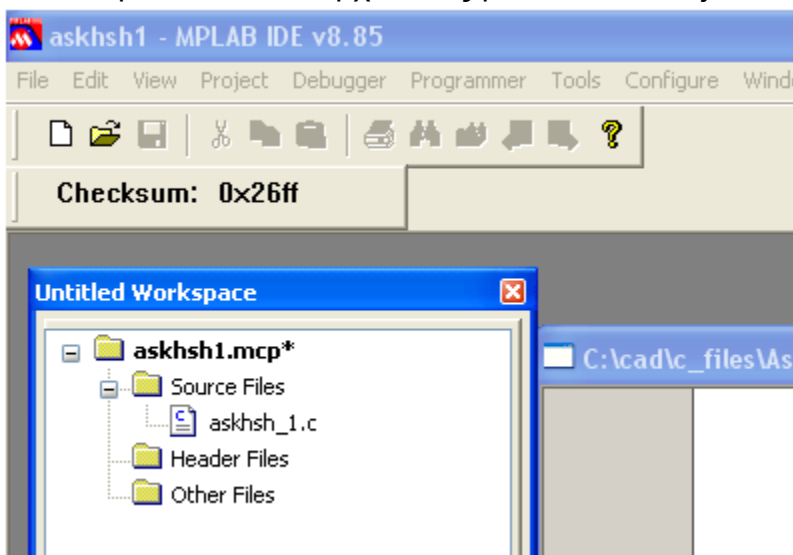


Κατόπιν καθοδηγείτε στο φάκελο του Project όπου υπάρχει το αρχείο με κατάληξη **c** και αφού το επιλέξετε πατάτε **open**.

Εργαστήριο στα Ενσωματωμένα Συστήματα – Άσκηση 1^η

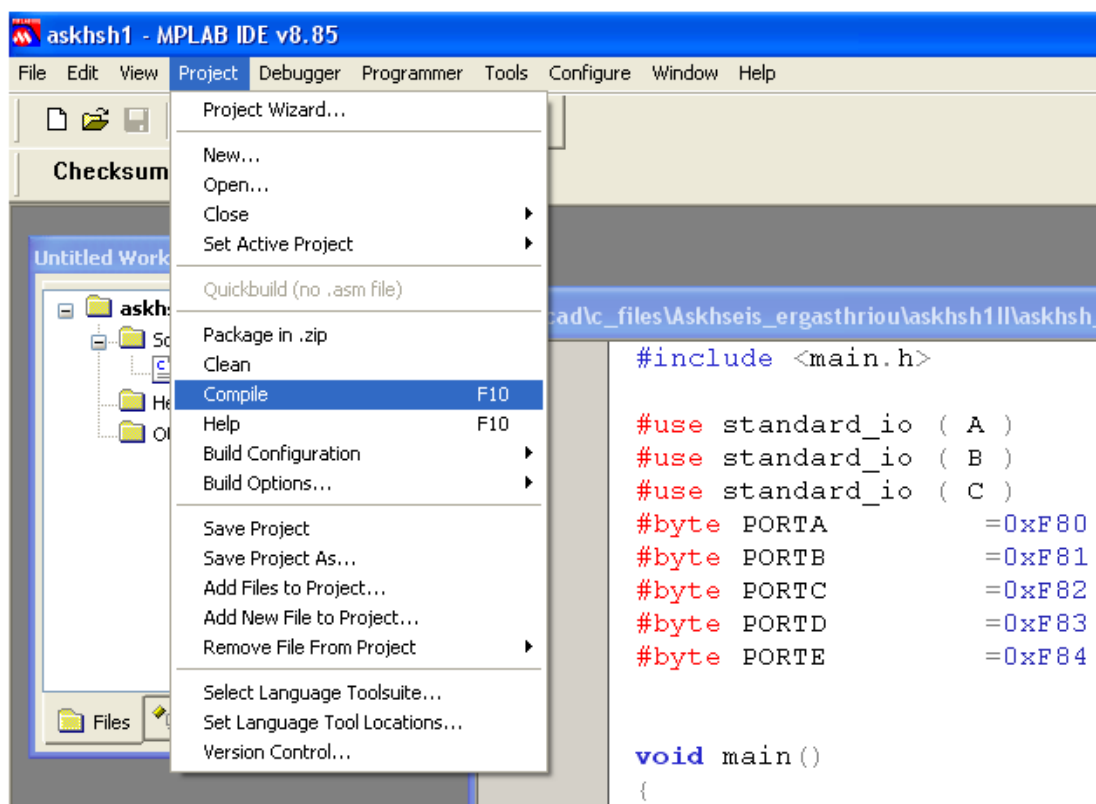


Με τον τρόπο αυτό το αρχείο σας μπαίνει στο Project.



Μέσα στο παράθυρο που έχει στο όνομα του κατάληξη **c** μπορείτε να γράψετε την εφαρμογή σας. Στο παράδειγμα askhsh_1.c .

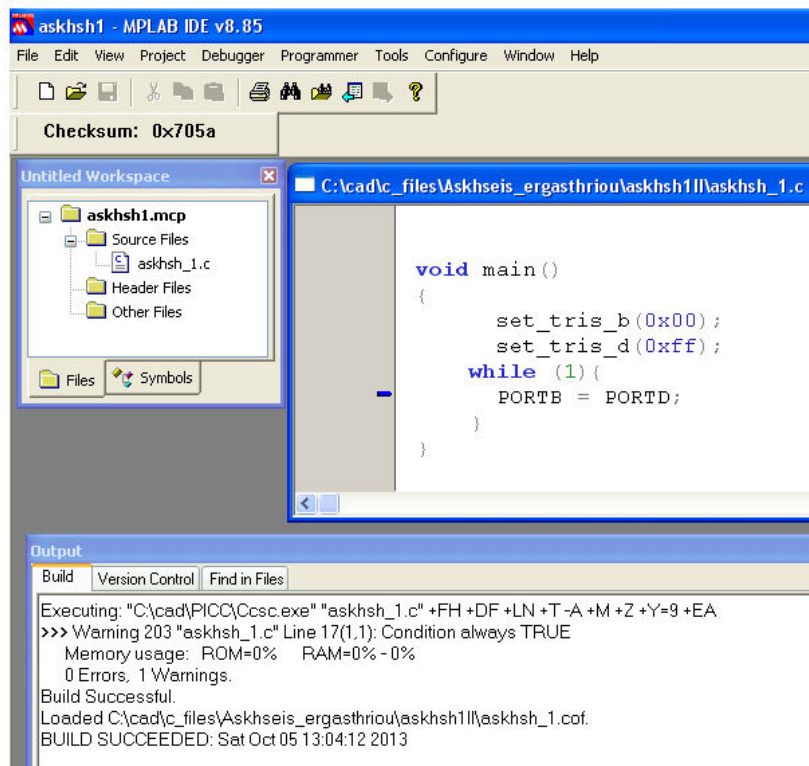
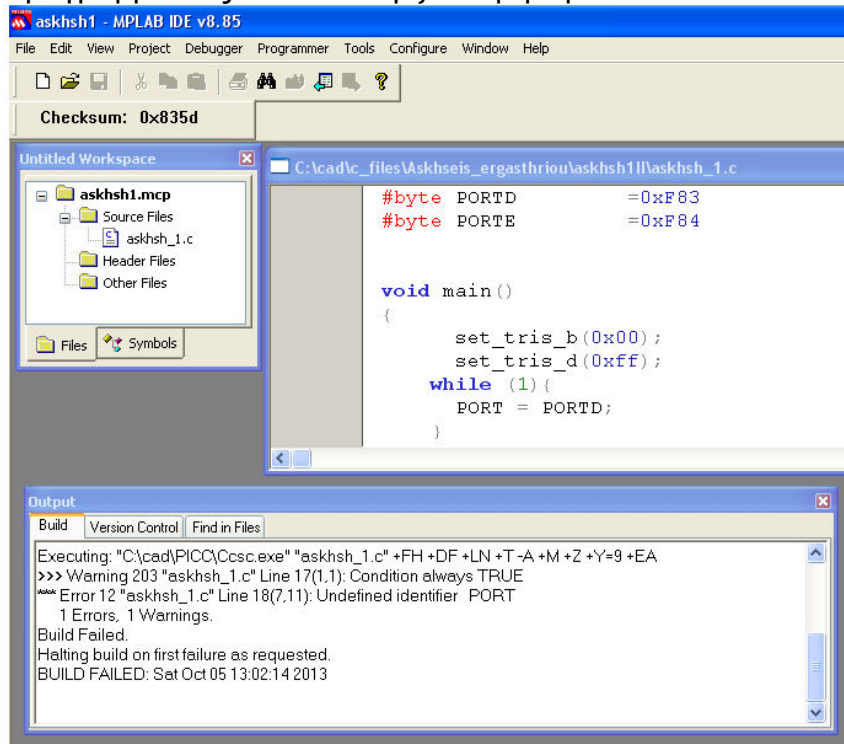
Αφού εγγραφεί όλο το πρόγραμμα θα πρέπει να σωθεί ξανά και για να γίνει μετάφρασή του από γλώσσα C σε γλώσσα μηχανής επιλέγετε από το κεντρικό μενού **Project** και κατόπιν **compile** ή **Build all**. Με τον τρόπο αυτό το MPLAB μεταφράζει το πηγαίο C κώδικα σε γλώσσα μηχανής. Ο compiler ο assembler και ο linker είναι τα προγράμματα που μετέχουν στην λειτουργία αυτή. Ο compiler μεταφράζει τον κώδικα σας που είναι σε γλώσσα C και έχει κατάληξη C σε assembly γλώσσα και κατόπιν ο assembler σε γλώσσα μηχανής. Ο linker συνδέει τυχών εξωτερικές βιβλιοθήκες ή εξωτερικά υποπρογράμματα σε ένα hex αρχείο. Από τα αρχεία που δημιουργούνται και έχουν το ίδιο όνομα χρειάζεται το αρχείο με κατάληξη **hex**. Το αρχείο **hex** χρειάζεται ο προγραμματιστής στην συγκεκριμένη περίπτωση ο boot-loader για να περάσει τον κώδικα στο μικροελεγκτή. Το αρχείο με κατάληξη **lst** είναι βοηθητικό αρχείο και χρησιμεύει για τον έλεγχο του προγράμματος που έχετε γράψει για συντακτικά λάθη και όχι για λάθη λογικής. Το αρχείο με κατάληξη **err** περιέχει πληροφορίες για τυχών λάθη που υπάρχουν στο πρόγραμμα σας και εμφανίζονται σαν πληροφορίες στο τέλος του **output** παραθύρου.



Αν υπάρχουν λάθη μετά την μετάφραση στο παράθυρο **Output** θα υπάρξει το μήνυμα **Build Failed**. Αυτό σημαίνει ότι θα πρέπει να διορθώσετε τα λάθη

Εργαστήριο στα Ενσωματωμένα Συστήματα – Άσκηση 1^η

και να ξανακάνετε compile μέχρι να μην υπάρχει κανένα λάθος στο πρόγραμμά σας και να υπάρξει το μήνυμα **Build Succeeded**



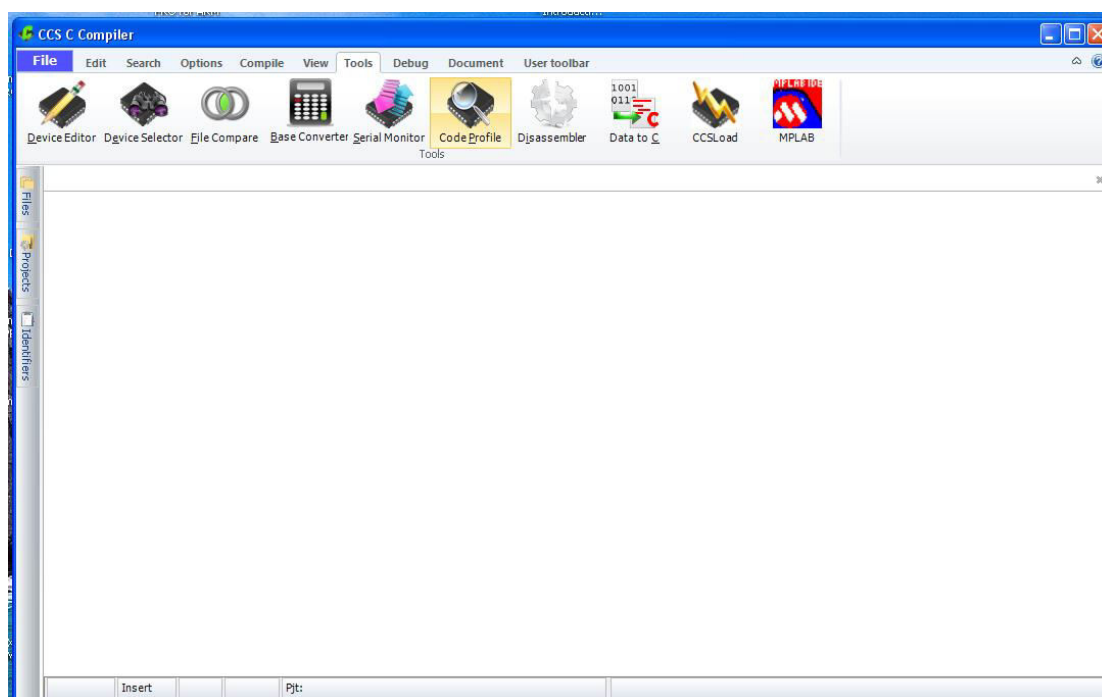
Για την εύρεση των λαθών πηγαίνετε στο παράθυρο Output και πατάτε διπλό κλικ με το αριστερό κουμπί του mouse πάνω στο μήνυμα του λάθους. Το πρόγραμμα σας κατευθύνει άμεσα στο σημείο του λάθους μέσα στο πρόγραμμά σας. Είναι πολύ σημαντικό να ξεκινάτε την διόρθωση από το πρώτο λάθος που εμφανίζεται στην λίστα του output παράθυρου και να πηγαίνετε προς τα κάτω.

Δημιουργία νέου Project μέσα από το περιβάλλον εργασίας του C compiler CCS.

Για να ανοίξει το πρόγραμμα κάνετε διπλό κλικ με το αριστερό κουμπί του mouse



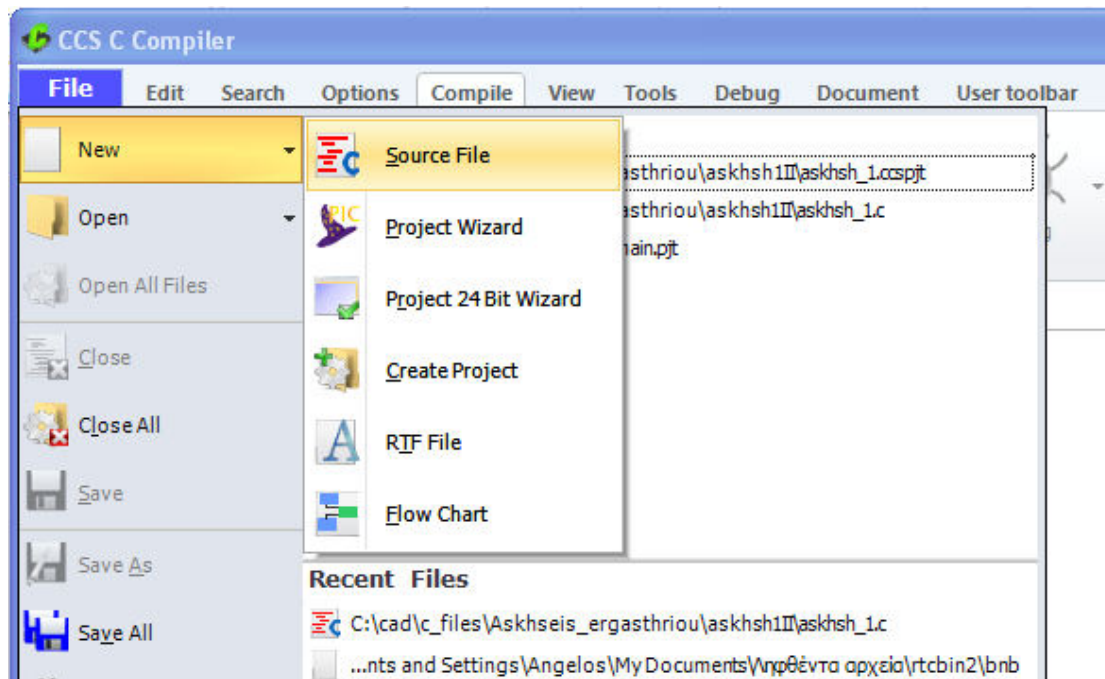
στο εικονάκι που βρίσκεται στην επιφάνεια εργασίας. Εμφανίζεται η παρακάτω εικόνα



Στο περιβάλλον αυτό υπάρχουν διάφορες επιλογές με πολλά βοηθητικά προγράμματα ενσωματωμένα.

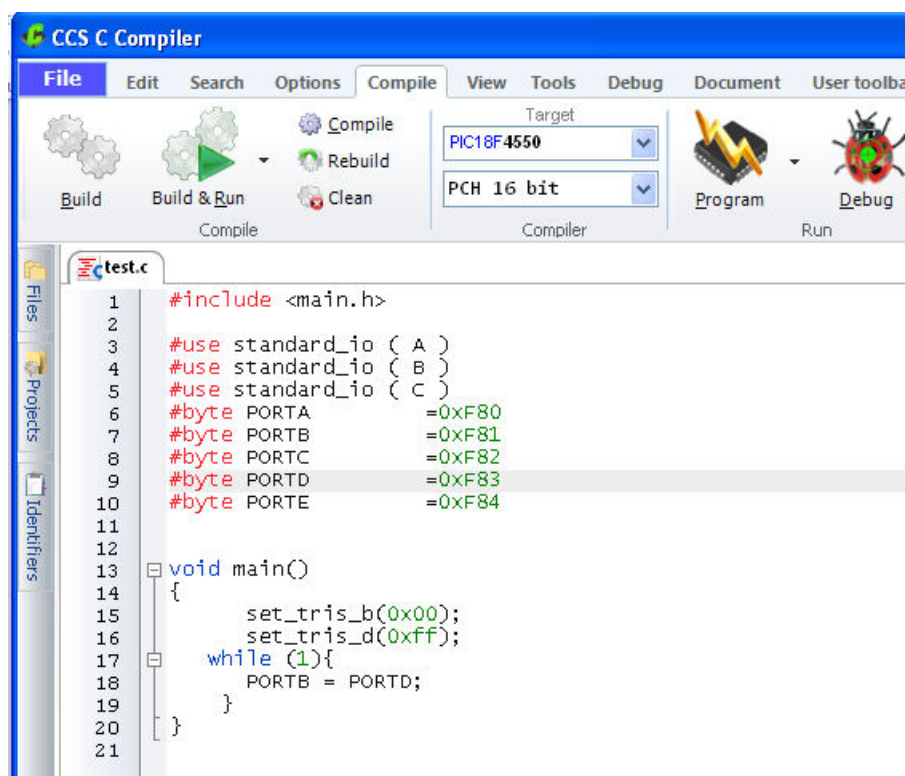
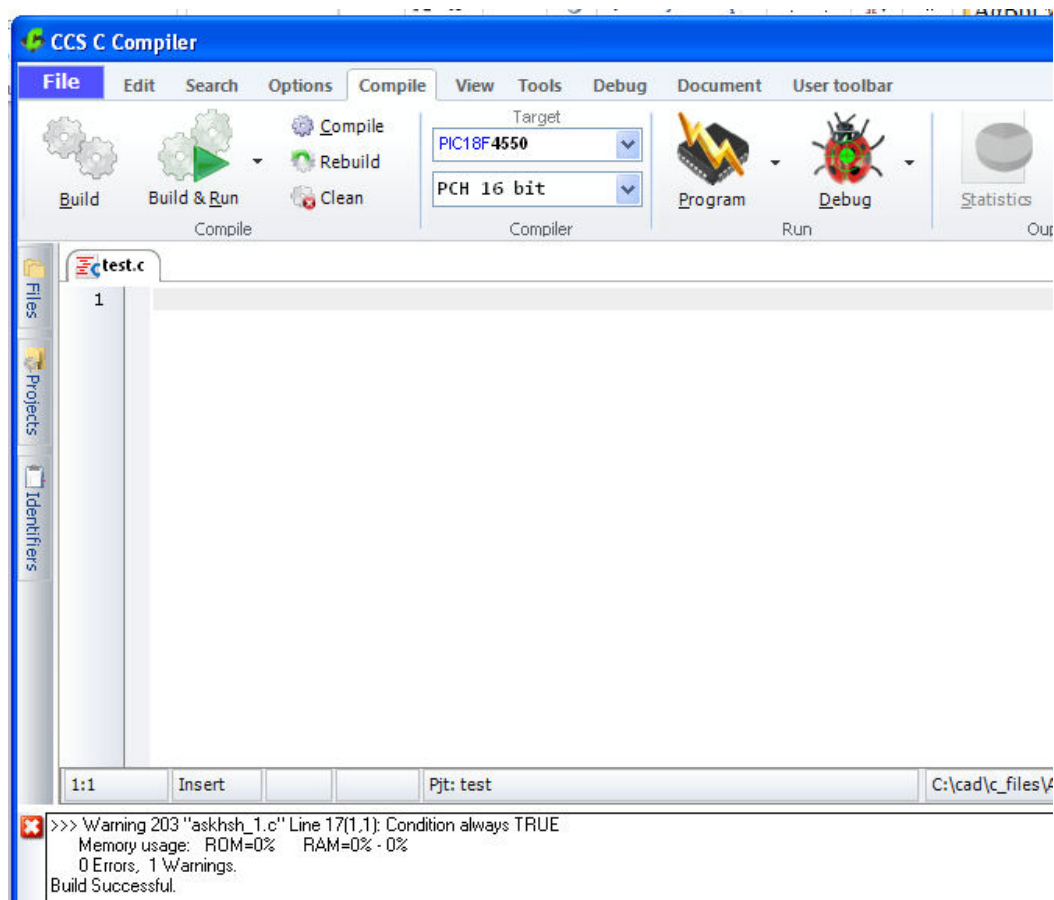
Για την δημιουργία του Project υπάρχουν δύο επιλογές είτε χρησιμοποιώντας τον αυτόματο τρόπο μέσω του **Project Wizard** ή με τον απλό τρόπο επιλέγοντας **Create Project**. Η δεύτερη περίπτωση προϋποθέτει την ύπαρξη πηγαίου αρχείου. Επειδή με τον αυτόματο τρόπο δημιουργούνται νέα includes αρχεία με τα setting των ασφαλειών που δεν βολεύουν στην

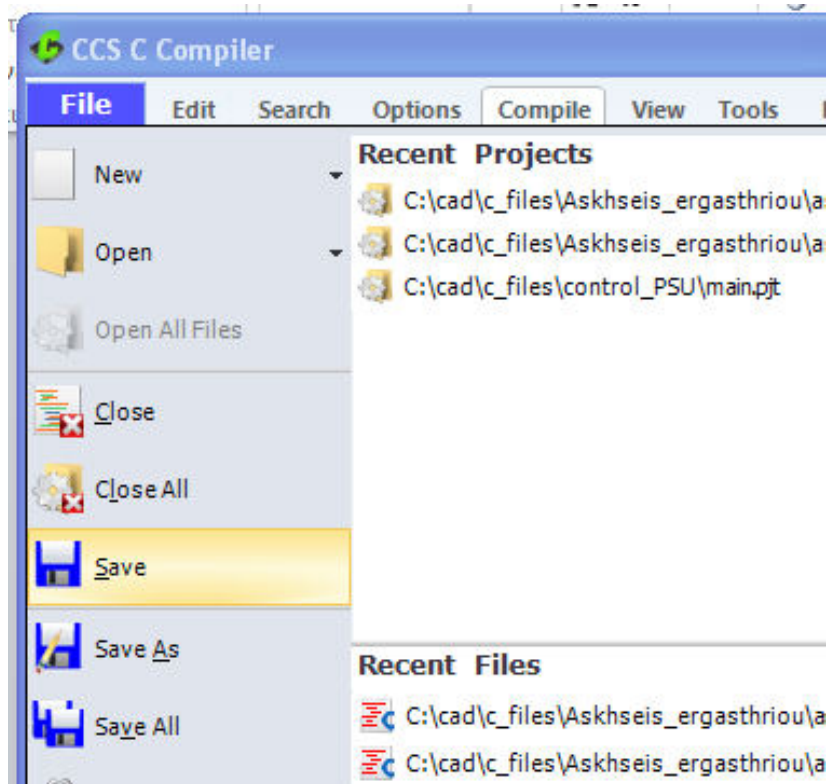
περίπτωση που κάποιος χρησιμοποιεί boot-loader όπως στην περίπτωση μας για τον λόγο αυτό επιλέγετε το απλό τρόπο δημιουργίας του Project. Για την δημιουργία του πηγαίου κώδικα σε C επιλέγετε από το επάνω tool bar **file** , **new** , **source file**



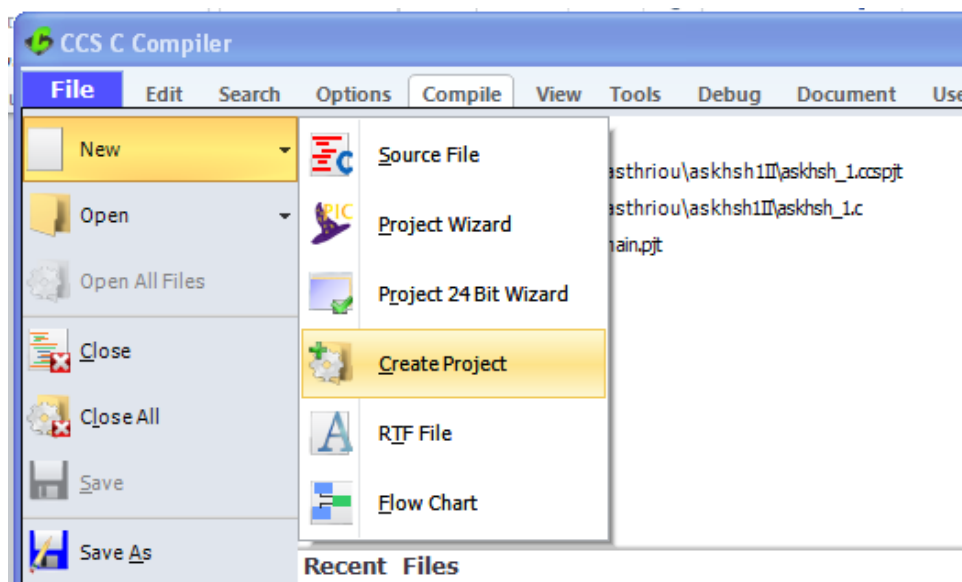
και αμέσως ανοίγει ένα νέο παράθυρο όπου θα πρέπει να οριστεί η θέση όπου θα αποθηκευτεί ο πηγαίος κώδικας. Είναι σημαντικό να είναι στον ίδιο φάκελο με το Project. Στο παράθυρο που ανοίγει γράφεται το πρόγραμμα σε γλώσσα C και αφού ολοκληρωθεί η εγγραφή σώζεται στο φάκελο του Project από την επιλογή **File Save**.

Εργαστήριο στα Ενσωματωμένα Συστήματα – Άσκηση 1^η

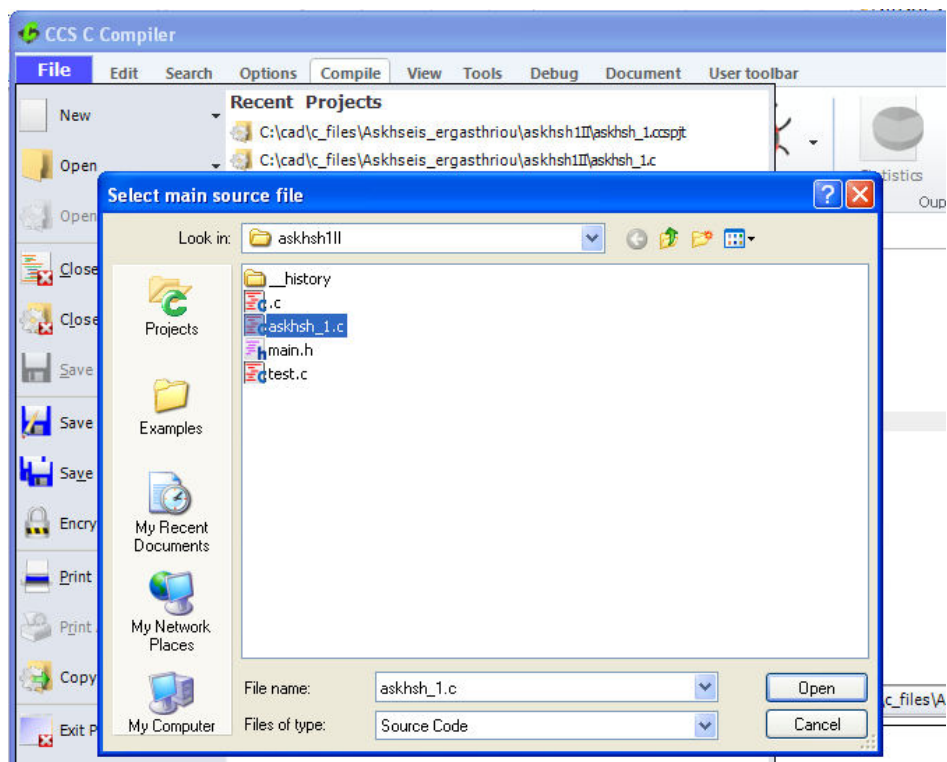




Για την δημιουργία του Project επιλέγετε από το επάνω tool bar την επιλογή **File, New, Create Project**.

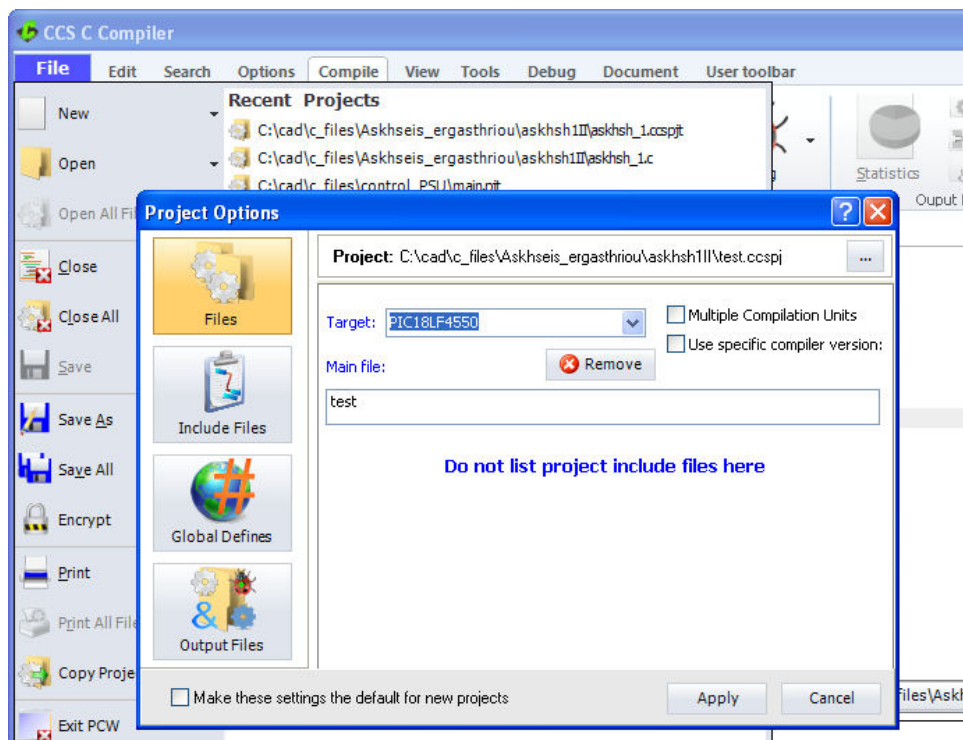


Το πρόγραμμα ζητά το πηγαίο αρχείο C που έχει μόλις προηγουμένως δημιουργηθεί.

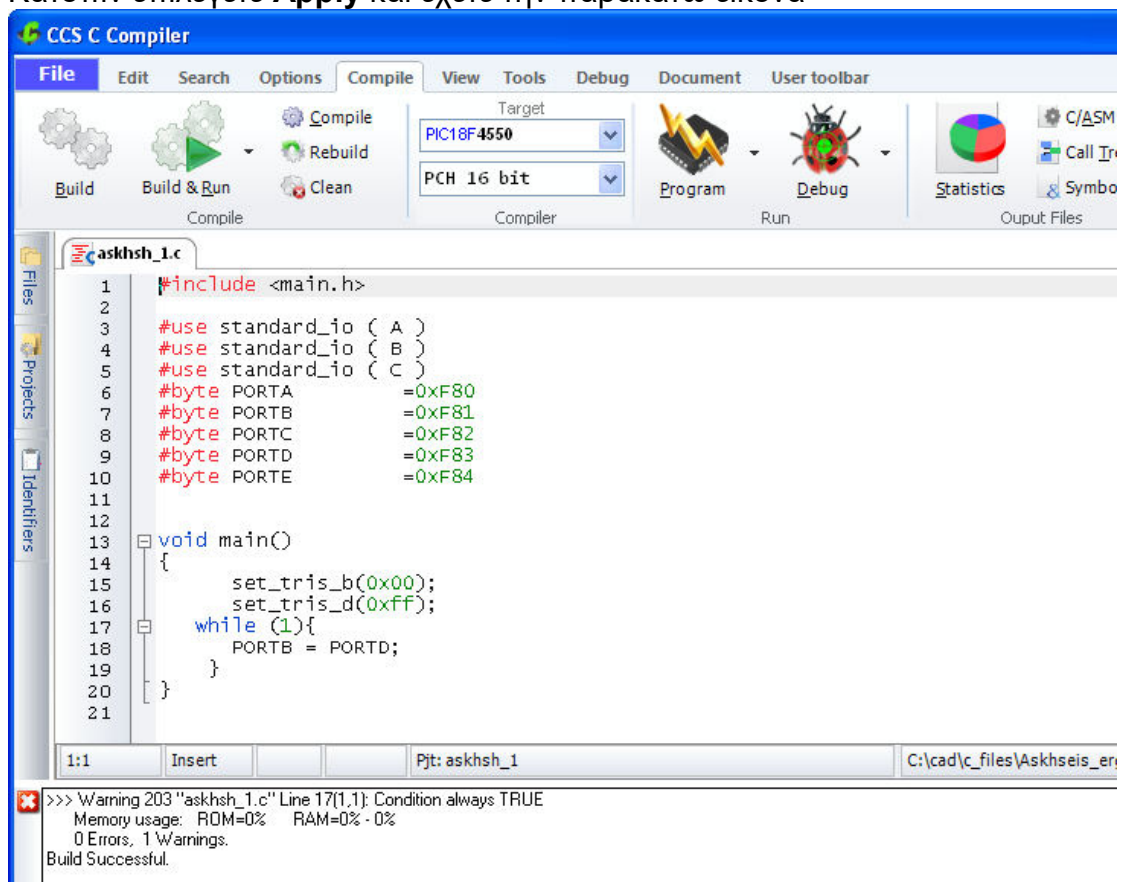


Επιλέγετε το αρχείο και κατόπιν επιλέγετε το **open**.
Από το επόμενο παράθυρο και από το πεδίο **Target** επιλέγετε το τύπο του μικροελεγκτή που θα χρησιμοποιηθεί στην εφαρμογή, για την πλακέτα του εργαστηρίου **PIC18F4550**.

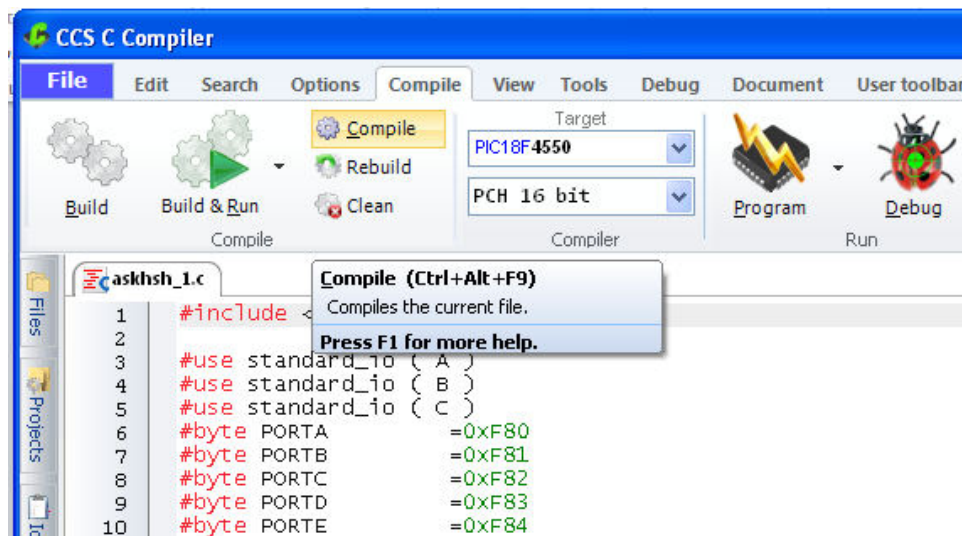
Εργαστήριο στα Ενσωματωμένα Συστήματα – Άσκηση 1^η



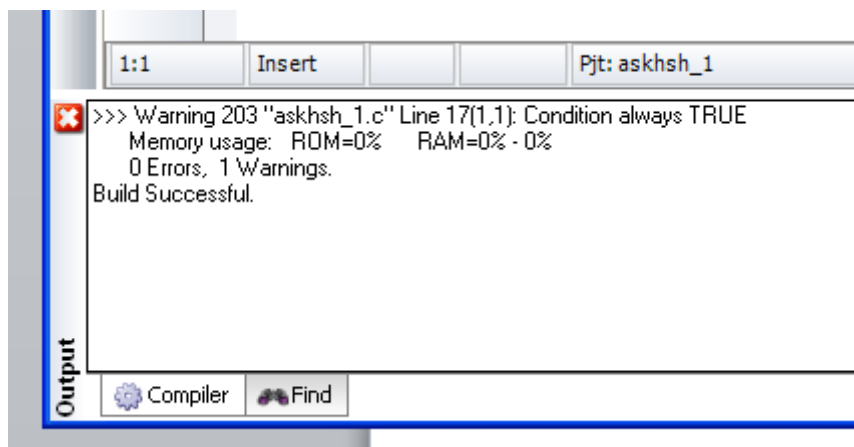
Κατόπιν επιλέγετε **Apply** και έχετε την παρακάτω εικόνα



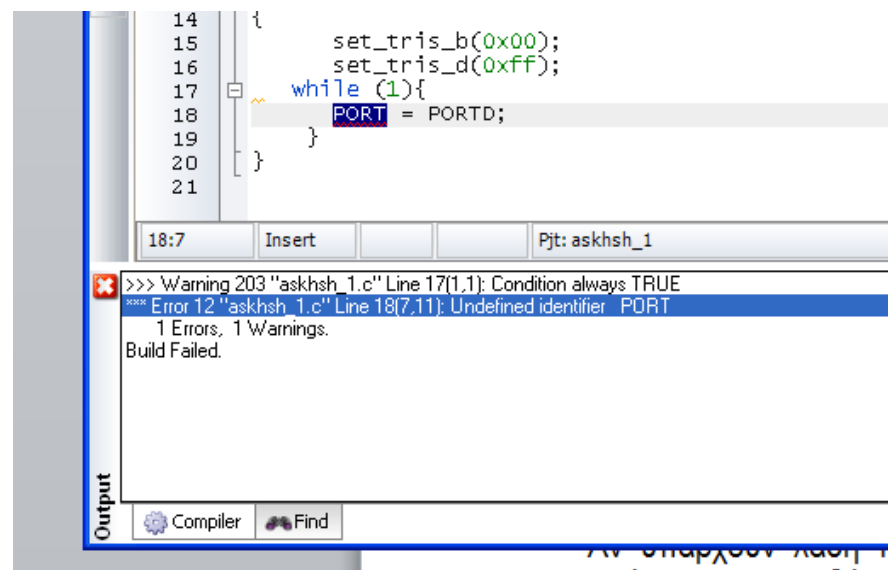
Για να μεταφραστεί το πρόγραμμα σε γλώσσα C επιλέγετε **Compile**



Στο κάτω παράθυρο Output εμφανίζονται διάφορες πληροφορίες σχετικά με την μετάφραση όπως αν υπάρχουν λάθη, το χώρο ποσοστιαία που καταλαμβάνει το πρόγραμμα στην μνήμη προγράμματος και δεδομένων.

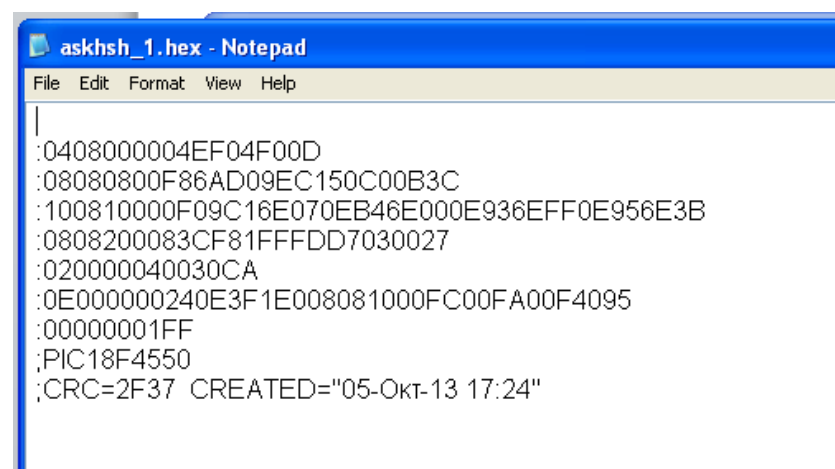


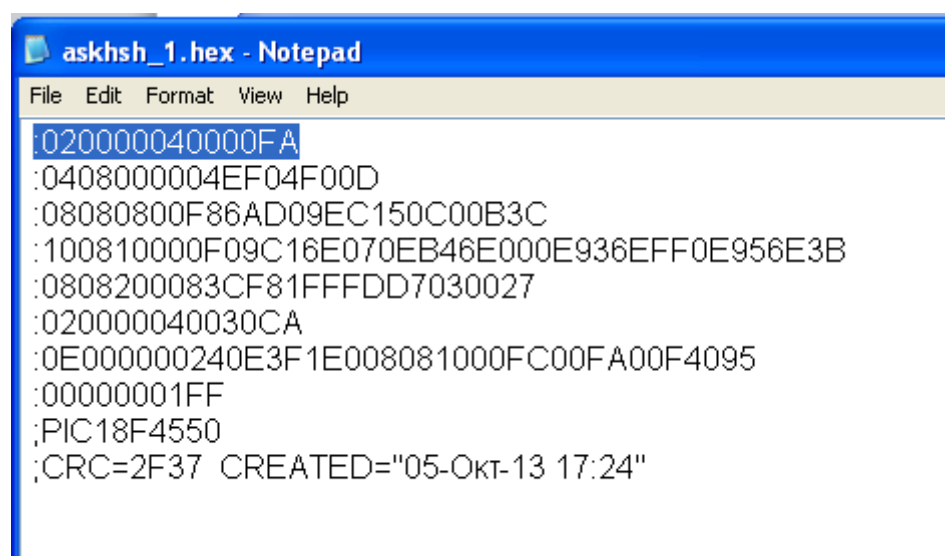
Αν υπάρχουν λάθη τότε κάνοντας διπλό κλικ πάνω στο μήνυμα λάθους το πρόγραμμα κατευθύνει τον χρήστη άμεσα στη θέση του λάθους στο κώδικα.



Στο σημείο αυτό θα πρέπει να τονιστεί ότι σε κάθε πρόγραμμα που θα δημιουργείτε στον φάκελο του Project θα πρέπει να συμπεριλαμβάνεται το αρχείο **main.h**. Στο αρχείο αυτό έχουν οριστεί όλες οι ασφάλειες λειτουργίας του PIC και δεν πρέπει να αλλαχτούν διότι μπορεί να δημιουργηθεί δυσλειτουργία στον Boot-Loader. Επίσης ένα άλλο πρόβλημα που δημιουργείται από τον συγκεκριμένο C Compiler είναι στην λειτουργία του με τον Boot-Loader λείπει από το Hex file η πρώτη γραμμή. Η γραμμή αυτή θα πρέπει να προστεθεί ανοίγοντας το Hex file με κάποιον editor όπως το σημειωματάριο (Note pad). Η γραμμή που λείπει θα πρέπει να προστεθεί στην αρχή (επάνω μέρος) του Hex file. Όπως φαίνεται στις παρακάτω εικόνες. Η γραμμή που πρέπει να προστεθεί είναι η ίδια σε όλα τα προγράμματα και είναι η

:020000040000FA





```
askhsh_1.hex - Notepad
File Edit Format View Help
:0200000040000FA
:0408000004EF04F00D
:08080800F86AD09EC150C00B3C
:100810000F09C16E070EB46E000E936EFF0E956E3B
:0808200083CF81FFFD7030027
:0200000040030CA
:0E000000240E3F1E008081000FC00FA00F4095
:00000001FF
;PIC18F4550
;CRC=2F37 CREATED="05-Οκτ-13 17:24"
```

Αφού προστεθεί η γραμμή αυτή σώζετε το Hex αρχείο και είναι έτοιμο να εγγραφεί στην μνήμη του μικροελεγκτή μέσω του προγράμματος του Boot-Loader όπως περιγράφεται παρακάτω.

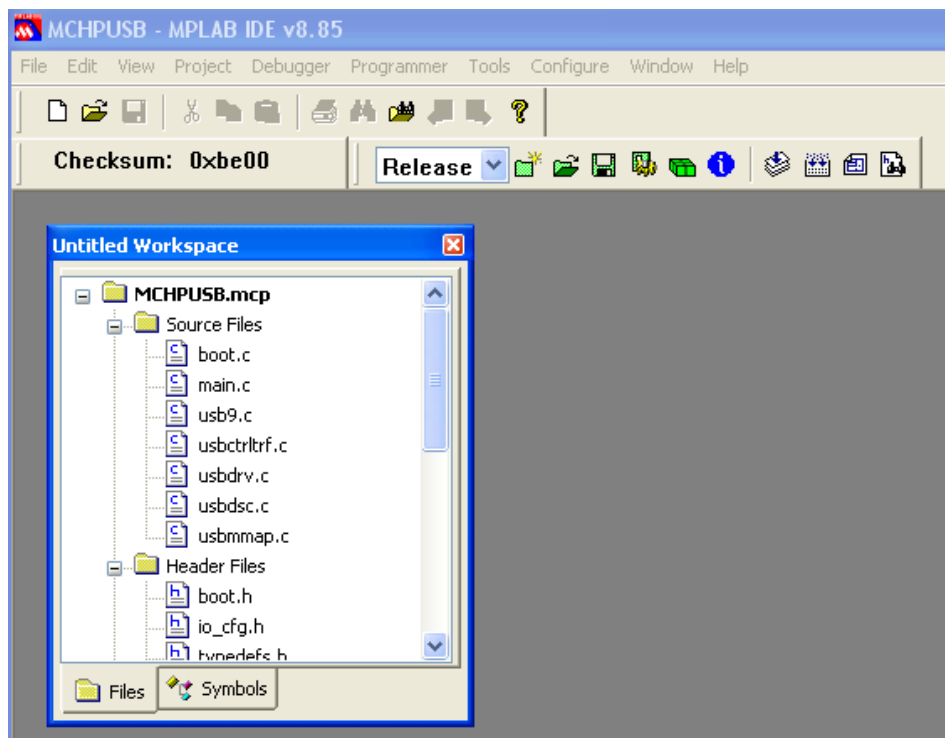
Προγραμματισμός του PIC18F4550 μέσω του Boot-Loader.

Για να μπορεί να προγραμματίζεται ο PIC της νέας πλακέτας μέσω της θύρας USB χωρίς την παρουσία κυκλώματος προγραμματιστή απαιτούνται δύο προγράμματα. Το πρόγραμμα που τρέχει αρχικά στον μικροελεγκτή και το πρόγραμμα που θα τρέχει στον Η/Υ. Για την εγγραφή του προγράμματος του μικροελεγκτή **για την πρώτη φορά** απαιτείται ένα προγραμματιστής για την οικογένεια των PIC και ειδικά για τον μικροελεγκτή PIC 18F4550. Για τον σκοπό αυτό μπορούν να χρησιμοποιηθούν η παλαιά αναπτυξιακή πλακέτα σαν προγραμματιστής ή κάποιος άλλος προγραμματιστής π.χ. PicKit2, PicKit3, PicKit4, PicStart, ICD2, ICD3, ICD4 κ.τ.λ.

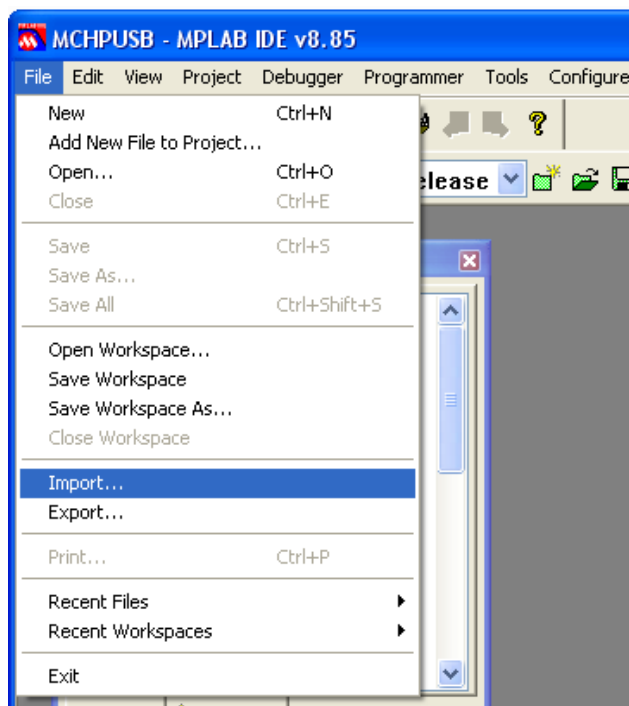
Στο εργαστήριο χρησιμοποιούμε το PicKit3 για την εγγραφή του προγράμματος εκκίνησης (boot-loader) στο μικροελεγκτή. **Θα πρέπει να τονιστεί ότι η διαδικασία αυτή έχει ήδη γίνει για τις πλακέτες του εργαστηρίου και δεν χρειάζεται να ξαναγίνει.**

Ο προγραμματισμός γίνεται ως εξής:

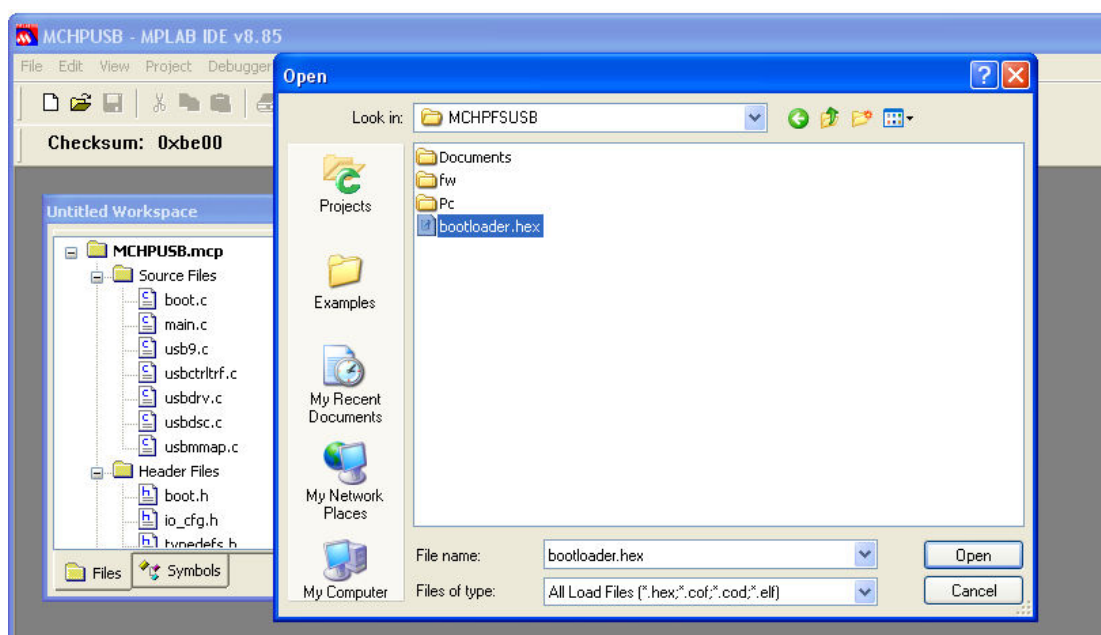
Θα πρέπει αρχικά να φορτωθεί στο MPLAB το project του boot-loader που δίδεται μαζί με όλα τα απαραίτητα αρχεία.



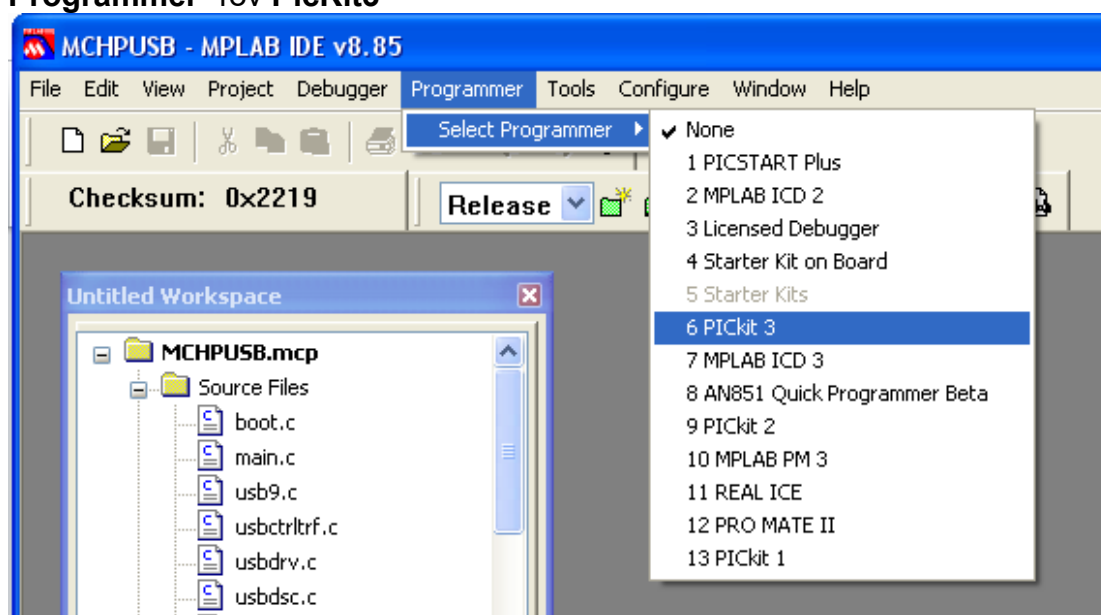
Κατόπιν από την επιλογή **Files** του κεντρικού μενού επιλέγετε **import**



και κατευθύνετε στην θέση όπου υπάρχει το **hex file** του **boot-loader**. Αφού



φορτωθεί επιλέγετε από το κεντρικό μενού **Programmer**, **Select Programmer** τον **PicKit3**



και κατόπιν από την ίδια επιλογή του κεντρικού μενού (**Programmer**) επιλέγετε **Erase Part** και κατόπιν **Program**. Με την διαδικασία αυτή εγγράφετε στον PIC 18F4550 το πρόγραμμα για την επικοινωνία του PIC με τον Η/Υ μέσω USB.

Η διαδικασία που ακολουθεί δεν απαιτείται στο εργαστήριο διότι το πρόγραμμα επικοινωνίας υπάρχει ήδη εγκατεστημένο.

Για την εγγραφή του προγράμματος στο Η/Υ ακολουθείται η παρακάτω διαδικασία.

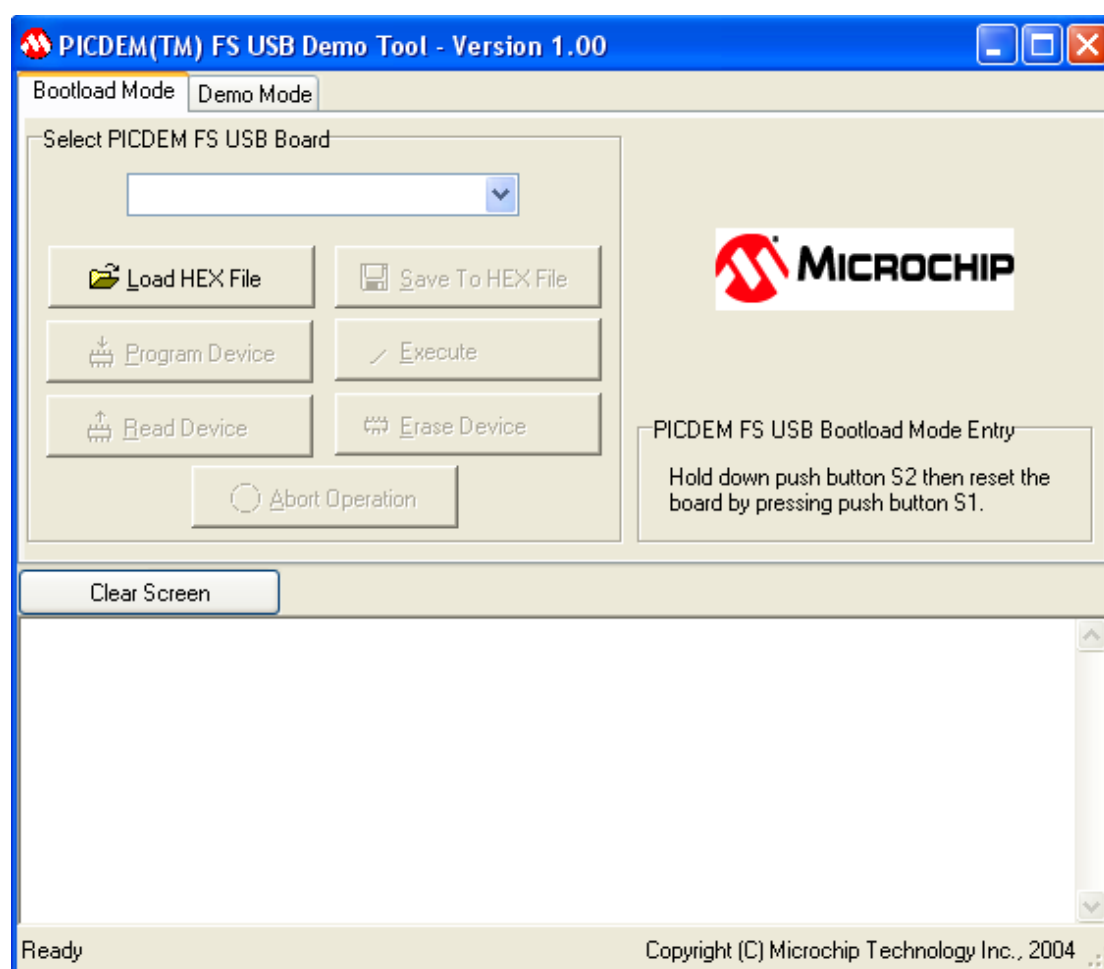
Αντιγράφετε από τον φάκελο MCHPFSUSB/PC/Pdfsusb το αρχείο PDFSUSB.exe στο επιθυμητό φάκελο. Το πρόγραμμα αυτό δεν έχει ανάγκη εγκατάστασης. Κατόπιν Συνδέεται την αναπτυξιακή πλακέτα σε μία USB θύρα του Η/Υ. Την πρώτη φορά που θα συνδεθεί και για κάθε Η/Υ ζητείται να εγκατασταθεί ο κατάλληλος driver. Ο driver βρίσκεται στον φάκελο \\MCHPFSUSB\\Pc\\MCHPUSB Driver\\Release και είναι εκεί όπου θα πρέπει να κατευθύνεται το πρόγραμμα για την λήψη του driver.

Λειτουργία του προγράμματος του Boot-Loader για τον προγραμματισμό της αναπτυξιακής πλακέτας με το πρόγραμμά σας.

Για να φορτώσετε την δική σας εφαρμογή στον PIC χρησιμοποιείται την εφαρμογή PDFSUSB.exe που είχατε αντιγράψει σε κάποιο φάκελο όπως

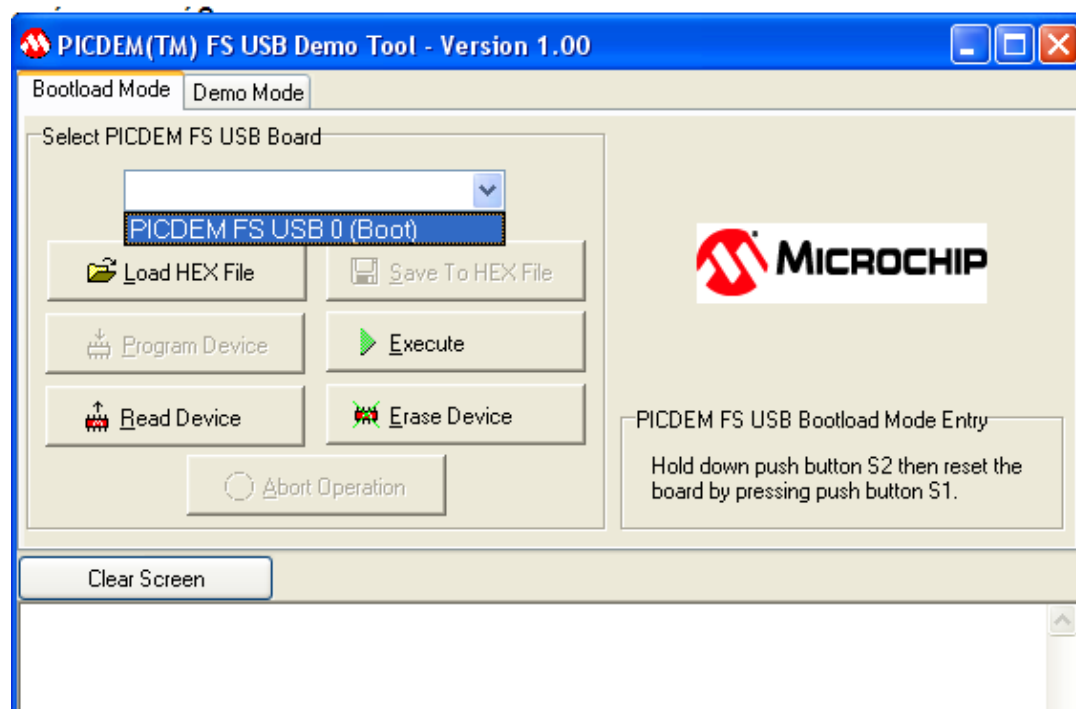


περιγράφηκε παραπάνω. Με διπλό κλίκ στο εικονίδιο της εφαρμογής θα εμφανιστεί το παρακάτω παράθυρο

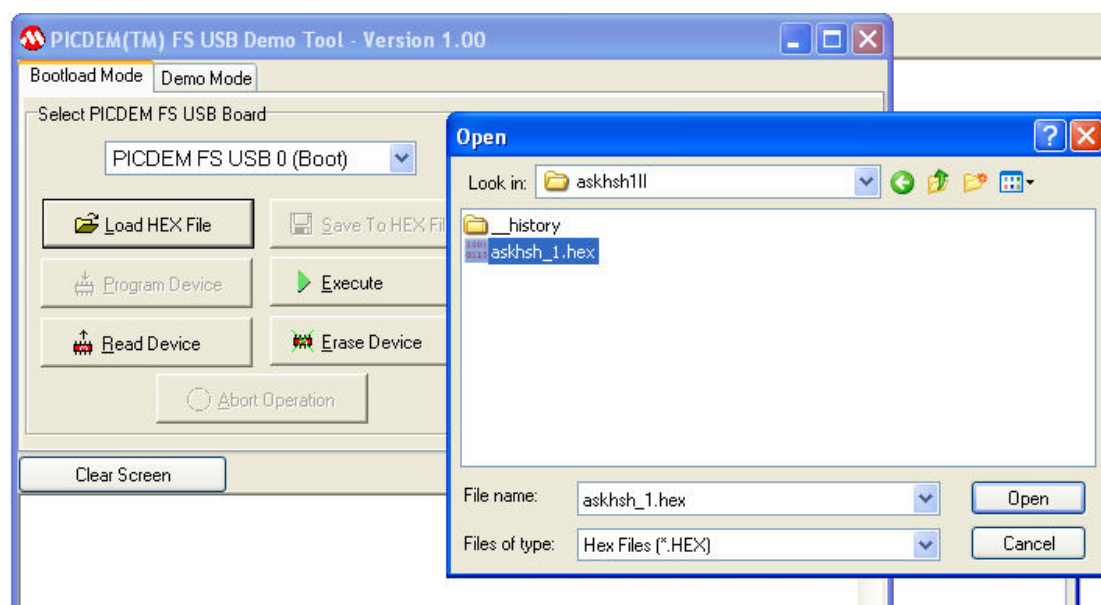


Για να γίνει η σύνδεση της αναπτυξιακής πλακέτας μέσω USB με τον Η/Υ πατάτε τα δύο μπουτόνς που υπάρχουν στην αναπτυξιακή πλακέτα ταυτόχρονα και κατόπιν αφήνετε πρώτα το επάνω μπουτόν (Reset) (κίτρινο) και κατόπιν το κάτω (Program) (κόκκινο). Με τον τρόπο αυτό το πρόγραμμα εκκίνησης (Boot-Loader) ενεργοποιείται για να υλοποιηθεί η διασύνδεση.

Από το πρόγραμμα στο Η/Υ επιλέγετε από το πεδίο **Select PICDEM FS USBBoard** την επιλογή **PICDEM FS USB 0(Boot)**



Κατόπιν από το πεδίο **Load HEX File** επιλέγετε το αρχείο με κατάληξη **HEX** της εφαρμογής σας που δημιουργήθηκε από τον Compiler όπως περιεγράφηκε παραπάνω.



Κατόπιν επιλέγετε **Erase Device** και μετά **Program Device** και τέλος **Execute** για να αποδεσμευτεί η πλακέτα από την σύνδεσή της με τον Η/Υ. Με την λειτουργία Execute παύει να τρέχει ο Boot-Loader και ξεκινά να τρέχει η εφαρμογή που είχε φορτωθεί όπως περιεγράφηκε στο προηγούμενο βήμα. Η διαδικασία μέσω του Boot-Loader έχει το πλεονέκτημα ότι μπορεί να φορτώνετε τις εφαρμογές σας στο PIC της πλακέτας χωρίς την παρουσία κάποιου προγραμματιστή αλλά έχει και κάποια μειονεκτήματα. Τα βασικά μειονεκτήματα είναι ότι θα πρέπει να θέσετε στο πρόγραμμά σας ως αρχική διεύθυνση την διεύθυνση 0x800 και όχι την 0x00 και τα interrupts τις διευθύνσεις 0x808 και 0x818 ενώ οι κανονικές διευθύνσεις είναι 0x08 και 0x18. Επιπλέον δεν μπορείτε να αλλάζετε όλες τις σημαίες για τις διάφορες λειτουργίες του PIC. Αυτό έχει ως αποτέλεσμα δεν μπορείτε να αλλάξετε την ταχύτητα λειτουργίας της πλακέτας (48MHz) όπως και πολλές άλλες λειτουργίες του PIC. Επίσης δεσμεύονται οι διευθύνσεις 0x000 – 0x800 της μνήμης του προγράμματος όπου γράφεται το πρόγραμμα του Boot-Loader.

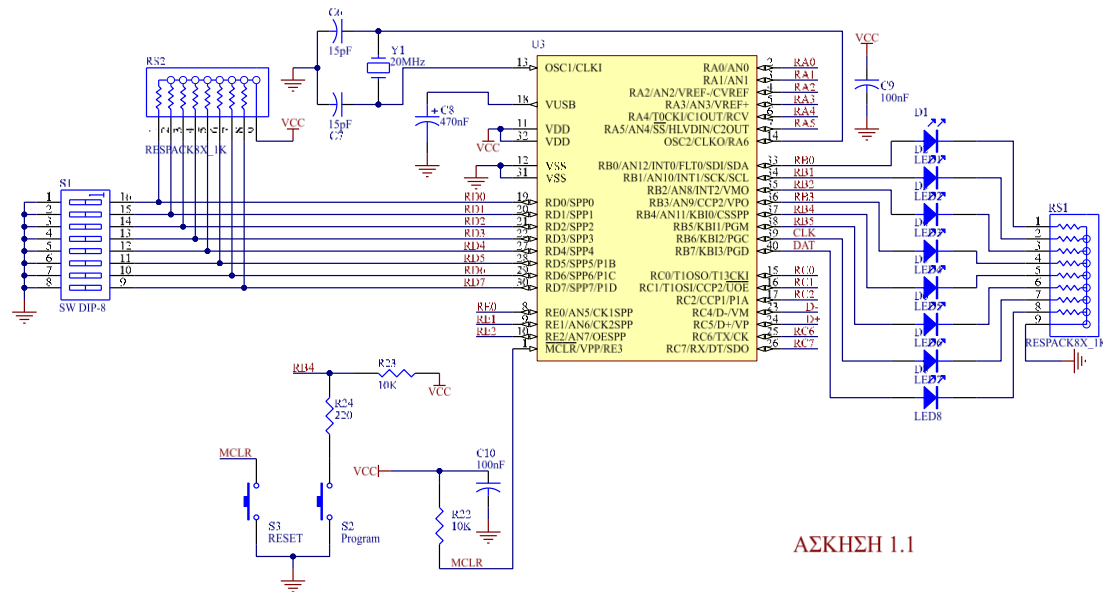
Άσκηση πρώτη του Εργαστηρίου

Στην πρώτη αυτή άσκηση δίδονται όλα τα απαραίτητα εργαλεία για την σύνδεση της νέας πλακέτας του εργαστηρίου καθώς και ο τρόπος λειτουργίας της .

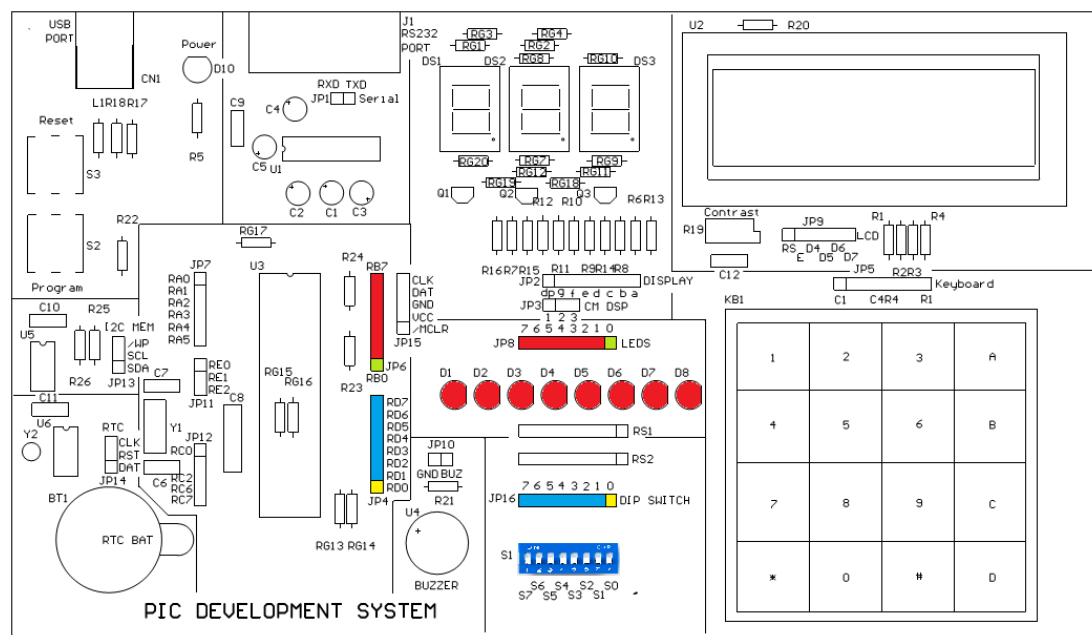
Στην άσκηση αρχικά συνδέονται η πόρτα B στα Leds της πλακέτας και την πόρτα D στους διακόπτες (Dip Switches).

Ο σκοπός της άσκησης είναι η ανάγνωση από την πόρτα D (Dip switches) και η εγγραφή των δεδομένων αυτών στην πόρτα B. (Leds).

Το σχηματικό της εφαρμογής



Οι συνδέσεις στην πλακέτα του εργαστηρίου



Το πρόγραμμα της εφαρμογής Άσκηση 1_1

```
#include <main.h>           // Περιέλαβε στο πρόγραμμα το εξωτερικό
                             //πρόγραμμα main.h

#use standard_io ( A )
#use standard_io ( B )
#use standard_io ( C )
#byte PORTA      =0xF80 //Διεύθυνση της πόρτας A
#byte PORTB      =0xF81 //Διεύθυνση της πόρτας B
#byte PORTC      =0xF82 //Διεύθυνση της πόρτας C
#byte PORTD      =0xF83 //Διεύθυνση της πόρτας D
#byte PORTE      =0xF84 //Διεύθυνση της πόρτας E

void main()              // Κύριο πρόγραμμα
{
    set_tris_b(0x00);    // Αρχικοποίηση της πόρτας B ως έξοδος
    set_tris_d(0xff);    // Αρχικοποίηση της πόρτας D ως είσοδος
    while (1){           // Εκτέλεσε για πάντα τις εντολές που περιέχονται
        PORTB = PORTD;    // Θέσε στην πόρτα B ότι έχει η πόρτα D
    }
}
```

Στο πρόγραμμα όπως και σε κάθε πρόγραμμα του εργαστηρίου στην πρώτη γραμμή θα υπάρχει η εντολή ενσωμάτωσης στο πρόγραμμά σας του αρχείου main.h που όπως προαναφέρθηκε μέσα στο αρχείο αυτό ορίζονται όλες οι ασφάλειες λειτουργίας του PIC.(τύπος του ταλαντωτή, η λειτουργία του power up timer, η λειτουργία του WDT όπως και πολλές άλλες ασφάλειες που δεν είναι στα ενδιαφέροντα του πρώτου προγράμματος να αναφερθούν).

Οι επόμενες τρεις γραμμές ορίζουν την λειτουργία των θυρών A,B,C ως πόρτες εισόδου εξόδου και αυτό γιατί ο συγκεκριμένος PIC όπως και οι περισσότεροι της σειράς PIC18Fxxx έχουν στα ίδια πόδια πολλαπλές λειτουργίες πέραν της εισόδου εξόδου. Στις επόμενες 5 γραμμές ορίζονται οι διευθύνσεις των θυρών A,..E στην μνήμη δεδομένων του PIC. Αυτό συμβαίνει επειδή στον CCS C compiler δεν ορίζονται οι διευθύνσεις αυτές.

Στο παρακάτω μέρος όπου είναι το κύριο πρόγραμμα (main) όπως και σε κάθε πρόγραμμα σε C. Στην συνάρτηση αυτή ορίζονται οι λειτουργίες που θα εκτελεί ο μικροελεγκτής. Όπως σε κάθε πρόγραμμα η πρώτη λειτουργία του προγράμματος θα πρέπει να είναι ο προγραμματισμός της λειτουργίας των περιφερειακών συσκευών στην προκειμένη περίπτωση των θυρών B, D. Για τον προγραμματισμό της λειτουργίας των θυρών ο κατασκευαστής του Compiler έχει δημιουργήσει κατάλληλες συναρτήσεις (Functions) που κάνουν το έργο των προγραμματιστών πιο εύκολο. Η κλήση των ρουτινών αυτών γίνεται στην αρχή του προγράμματος και καθορίζουν την λειτουργία των θυρών. Όταν απαιτείται ένα bit της θύρας να είναι είσοδος τότε το αντίστοιχο

bit του καταχωρητή TRIS τίθεται σε λογικό 1 ενώ όταν απαιτείται να είναι έξοδος τότε το αντίστοιχο bit του καταχωρητή TRIS τίθεται σε λογικό 0. Στην παραπάνω άσκηση απαιτείται η πόρτα D να είναι είσοδος οπότε όλα τα bits τίθενται σε λογικό 1 οπότε προκύπτει ο αριθμός $b11111111 = 0xff$. Ενώ η πόρτα B απαιτείται να είναι έξοδος οπότε όλα τα bit τίθενται σε λογικό 0 $b00000000 = 0x00$.

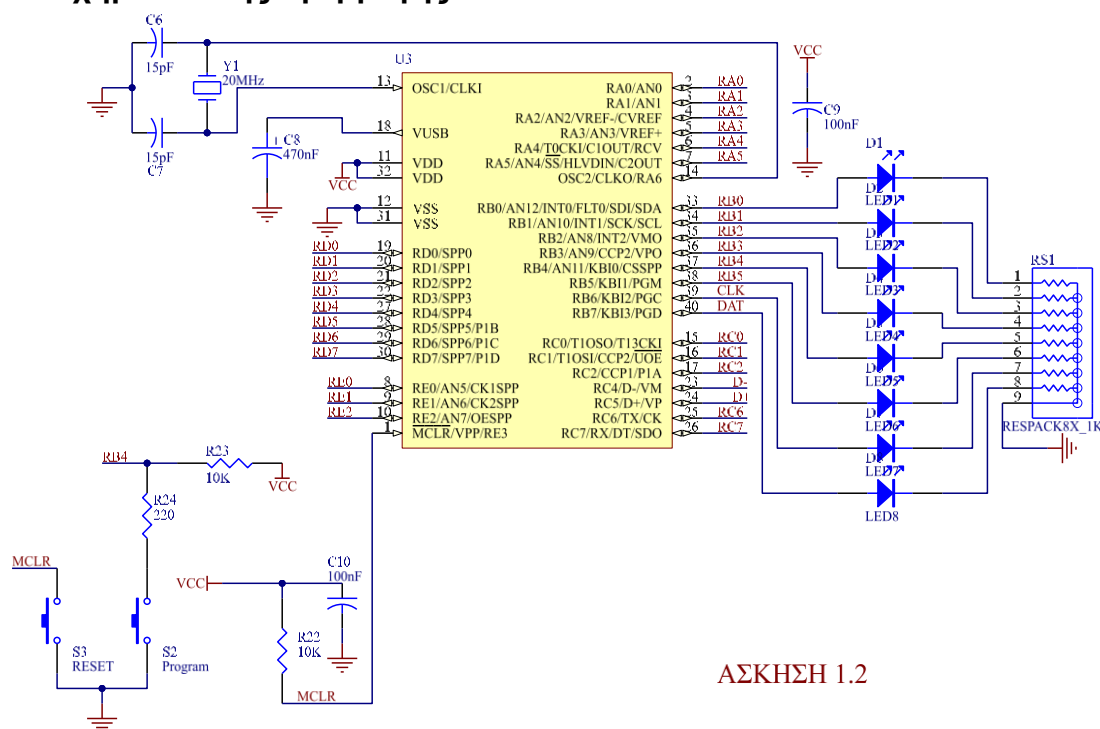
```
set_tris_b(0x00);
set_tris_d(0xff);
```

Κατόπιν το πρόγραμμα μπαίνει σε ένα ατέρμον βρόχο με την εντολή while (1) δηλαδή πάντα αληθείς άρα το πρόγραμμα που περιέχεται στις αγκύλες θα εκτελείται συνεχώς. Η λειτουργία που υπάρχει μέσα στις αγκύλες θέτουν στην πόρτα B (έξοδος) ότι διαβάζεται από την πόρτα D (είσοδος). PORTB = PORTD;

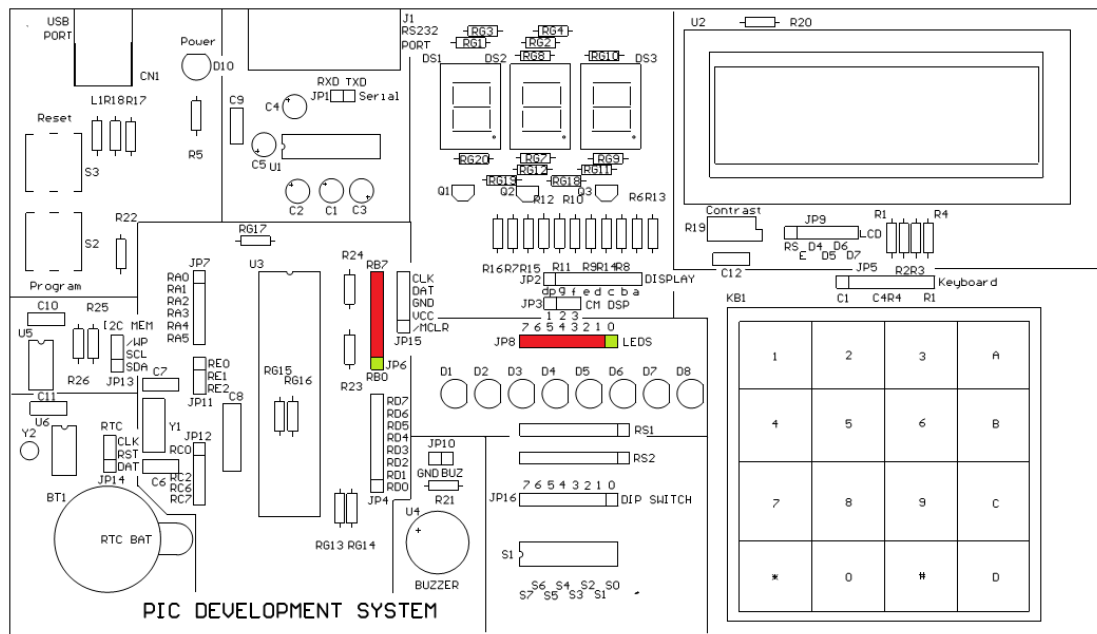
Άσκηση δεύτερη του πρώτου εργαστηρίου

Στην άσκηση αυτή τα Leds που συνδέονται μέσω της καλωδιωταινίας στην πόρτα B του PIC αναβοσβήνουν με ρυθμό περίπου 1Hz.

Το σχηματικό της εφαρμογής



Οι συνδέσεις στην πλακέτα του εργαστηρίου



Το πρόγραμμα της εφαρμογής Άσκηση 1_2

```
#include <main.h>
```

```
#use standard_io ( A )
#use standard_io ( B )
#use standard_io ( C )
#byte PORTA      =0xF80
#byte PORTB      =0xF81
#byte PORTC      =0xF82
#byte PORTD      =0xF83
#byte PORTE      =0xF84
```

```
void main()
{
    set_tris_b(0x00);
    while (1){
        PORTB = ~ PORTB;
        Delay_ms(500);
    }
}
```

Στην άσκηση αυτή αυτό που διαφοροποιείται είναι το πρόγραμμα μέσα στον ατέρμων βρόχο όπου στην άσκηση αυτή όλα τα bits της πόρτας B αλλάζουν κατάσταση (σύμβολο αντιστροφής ~) σε κάθε κύκλο μαζί χρησιμοποιείται μία

ακόμη έτοιμη συνάρτηση του C Compiler για την χρονοκαθυστέρηση των 500 msec.

Βιβλιογραφία

1. CCS C Compiler Reference Manuals
http://www.ccsinfo.com/downloads/ccs_c_manual.pdf
2. Mplab Quick start Microchip site www.microchip.com
3. Tutorial by Example by Peter H. Anderson
4. PIC C An introduction to Programming the Microchip PIC on C by Nigel Gardner.
5. Interfacing PIC Microcontrollers Embedded Design by Interactive Simulation by Martin Bates
6. Designing Embedded Systems with PIC Microcontrollers by Tim Wilmshurst
7. From Assembly to C using the PIC18xx2 by Robert B. Reese
8. Microcontrollers in Practice by M. Mitescu I. Susnea
9. PIC programming laboratory by James Gribbley
10. Microcontroller based applied digital control by_Dogan Ibrahim
11. Learn Hardware Firmware and Software Design_O.G.Popa
12. Advanced PIC Microcontroller Projects in C by Dogan Ibrahim
13. Embedded C Programming & Microchip Pic by Richard Barnett, Larry o Cull, Sarah Cox
14. Interfacing PIC Microcontrollers to Peripheral Devices by Bohdan Borowik

Ασκήσεις και ερωτήσεις που θα πρέπει να ετοιμαστούν για το πρώτο εργαστήριο.

Ασκήσεις

α. Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για τον μικροελεγκτή PIC18F4550 που να διαβάζει από την πόρτα D ένα αριθμό να προσθέτει σε αυτόν τον αριθμό των γραμμάτων του πατρώνυμου κατόπιν να ελέγχει τον αριθμό των bits που είναι σε λογικό «1» και αν ο αριθμός είναι άρτιος τότε να θέτει το λιγότερο σημαντικό bit της πόρτας B σε λογικό «1» διαφορετικά να το θέτει σε λογικό «0». Το πρόγραμμα εκτελείτε συνεχώς.

β, Να γραφεί πρόγραμμα σε γλώσσα προγραμματισμού C για τον μικροελεγκτή PIC18F4550 που να διαβάζει από την πόρτα B ένα αριθμό και ανάλογα με την κατάσταση των bits να κάνει την πόρτα D είσοδο ή έξοδο (Αν ένα από τα bits της πόρτα B είναι σε λογικό «1» τότε το αντίστοιχο bit της πόρτας D να γίνεται είσοδος ενώ αν το bit της πόρτας B είναι σε λογικό «0» το αντίστοιχο bit της πόρτας D να γίνεται έξοδος. Κατόπιν να θέτει στην πόρτα D τον αριθμό των γραμμάτων του επώνυμου αφού τον πολλαπλασιάσει με το 6. Το πρόγραμμα εκτελείτε συνεχώς.

γ. Συνδέστε στην πόρτα D τα διακοπτάκια της πλακέτας και στην πόρτα B τα Leds. Κατόπιν τρέξτε το παρακάτω πρόγραμμα και στα πρώτα 5 sec αλλάξτε την κατάσταση των διακοπών σε κάποια επιθυμητή θέση και κατόπιν μέσα στα επόμενα 5 sec σε κάποια άλλη θέση τους διακόπτες. Σημειώστε τι παρατηρείτε στα leds στα επόμενα 5 sec και εξηγήστε την λειτουργία αυτή.

```
#include <main.h>
#use standard_io(A)
#use standard_io(B)
#use standard_io(C)
#byte PORTA=0xF80
#byte PORTB=0xF81
#byte PORTC=0xF82
#byte PORTD=0xF83
#byte PORTE=0xF84
```

```
void main()
{
    set_tris_b(0x00);
    set_tris_d(0xff);
    int a;

    while(1) {
        PORTB = PORTD;
```

```
delay_ms(5000);
set_tris_b(0xFF);
a = PORTB;
delay_ms(5000);
set_tris_b(0x00);
PORTB = a;
delay_ms(5000);
    }
}
```

Ερωτήσεις.

Οι ερωτήσεις είναι προαιρετικές και για όσους θέλουν θα πρέπει να απαντηθούν μέσω email στην διεύθυνση angmicro2@gmail.com Δηλώνοντας το Όνομα το επώνυμο και το τμήμα του φοιτητή-τριας .

A. Ποιες από τις παρακάτω εκφράσεις είναι έγκυρες ?

1. 5;
2. 1+2;
3. {1;2;3;4}
4. Όλες οι εκφράσεις είναι έγκυρες στην στάνταρτ C.
5. Καμία από τις εκφράσεις δεν είναι έγκυρη

B. Πόσες φορές θα εκτελεστεί ότι υπάρχει μέσα στα άγκιστρα ?

```
for (i=1; i>=10; i=i+1)
{
    }
}
```

1. 11
2. 9
3. 1
4. 0
5. Επ' άπειρο.

Γ. Πόσες φορές θα εκτελεστεί ότι υπάρχει μέσα στα άγκιστρα ?

```
for (i=1; i<=10; i=i+3)
{
    }
}
```

1. 0
2. 2
3. 3
4. 4
5. Επ' άπειρο.

Δ. Ποιος ο σωστός τρόπος δήλωσης σχολίων στην standard C ?

1. int x // holds current x position;
2. process(data); /* removes symbols +-* / in the data */

3. `// temporary>> // int dummy;`
4. Όλοι οι τρόποι είναι έγκυροι
5. Κανένας τρόπος δεν είναι έγκυρος.

Ε Τι θα κάνει ο compiler με την παρακάτω έκφραση ?
`for (1;2;3)`

1. Θα παράξει ένα λάθος
2. Θα παράξει μία προειδοποίηση
3. Και τα δύο 1,2
4. Κανένα από τα δύο 1,2