Διευκρίνιση

Το συγκεκριμένο αρχείο έχει ανακτηθεί από τον ιστότοπο: http://teachers.cm.ihu.gr/kalomiros/Real_time/Ergastirio/Nea_fylla_ergou/

4. ΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ ΤΟΥ ΜΙΚΡΟΕΛΕΓΚΤΗ PIC16F877

4.1 Θεωρητική Εισαγωγή

Οι PIC είναι ολοκληρωμένα κυκλώματα που ανήκουν στην κατηγορία των μικροελεγκτών. Πήραν το όνομα τους από τα αρχικά των λέξεων Peripheral Interface Controller (Ελεγκτής Περιφερειακής Διεπαφής). Περικλείουν όλα τα απαραίτητα στοιχεία για την κατασκευή ενός ψηφιακού προγραμματιζόμενου συστήματος: κεντρική μονάδα επεξεργασίας, θύρες εισόδου-εξόδου για επικοινωνία με περιφεριακά συστήματα και μνήμη. Ορισμένοι μικροελεγκτές διαθέτουν επίσης μετατροπείς A/D και D/A, καθώς και τυπικά κανάλια επικοινωνίας, όπως UART ή I^2C. Έτσι, μπορούν να παίξουν τον ρόλο της κεντρικής μονάδας ελέγχου ενός συστήματος. Εξωτερικά μοιάζουν με απλά ψηφιακά ολοκληρωμένα, όμως μέσα τους κρύβουν ένα μικρό υπολογιστή.

4.1.1 Διαφορές μικροελεγκτή - μικροεπεξεργαστή

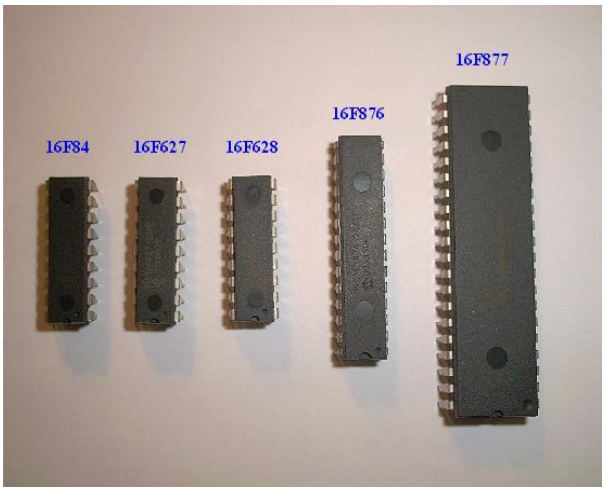
Ο μικροελεγκτής είναι ένα μικρό αυτόνομο υπολογιστικό σύστημα, προγραμματισμένο να εκτελεί μια συγκεκριμένη λογική ακολουθία εντολών, οι οποίες έχουν καταχωρηθεί στην προγραμματιζόμενη μόνιμη μνήμη του. Κάθε φορά που επανεκινείται ο μικροελεγκτής εκτελείται η ίδια λογική. Θα δέχεται τα δεδομένα, θα τα επεξεργάζεται και με βάση τα αποτελέσματα της επεξεργασίας, θα ελέγχει το περιβάλλον του. Πρόκειται, δηλαδή για σύστημα ειδικού σκοπού, αφιερωμένο στον έλεγχο και την εξυπηρέτηση ενός συγκεκριμένου αυτοματισμού.

Αντίθετα, ένας μικροεπεξεργαστής μετά την εκκίνηση του δεν είναι από μόνος του σε θέση να εκτελέσει μια λογική ακολουθία. Αν και μπορεί να συνδεθεί με μνήμες RAM και ROM, αυτές αποτελούν ξεχωριστές μονάδες, που συνήθως δεν ολοκληρώνονται μέσα στον ίδιο τον μικροεπεξεργαστή.

Οι μικρελεγκτές αναφέρονται συχνά ως υπολογιστές σε προγραμματιζόμενο τσιπ (Computer on a programmable chip).

4.1.2 Χαρακτηριστικά του μικροελεγκτή PIC16F877

Ο PIC16F877 είναι ένας ισχυρός και εύκολος στον προγραμματισμό μικροελεγκτής τεχνολογίας CMOS της εταιρίας Microchip. Με τον όρο CMOS εννοούμε την τεχνολογία που στηρίζεται σε συμπληρωματικά τρανζίστορ MOSFET. Είναι τεχνολογία χαμηλής ισχύος και υψηλής ταχύτητας. Ο PIC16F877 βασίζεται στην αρχιτεκτονική RISC και αποτελεί ένα πακέτο 40-ακροδεκτών. Είναι συμβατός με προγενέστερους μικροελεγκτές της ίδιας οικογένειας.



Σχήμα 4-1: Η οικογένεια τον μικροελεγκτών PIC 16fXXX

Ο μικροελεγκτής PIC16F877 διαθέτει μνήμη δεδομένων EEPROM των 256 bytes. Επίσης, διαθέτει ενσωματωμένο χρονιστή επιτήρησης (watch-dog timer) και μία σύγχρονη σειριακή θύρα. Μπορεί να παράγει έξοδο παλμοσειράς με διαμορφούμενο εύρος (PWM). Αυτή μπορεί να χρησιμοποιηθεί για τον έλεγχο διαφόρων βιομηχανικών εφαρμογών (π.χ βαλβίδες ή σερβοκινητήρες). Έχει πέντε (3) θύρες εισόδου/εξόδου (θύρες B, C, D) των οχτώ (8) δυαδικών ψηφίων (bits) και άλλες δύο, A και B των έξι (RA0-RA5) και τριών ακροδεκτών (RE0-RE2) αντίστοιχα. Αυτές μπορούν να χρησιμοποιηθούν ή σαν απλές ψηφιακές θύρες, ή σαν θύρες των υπόλοιπων περιφερειακών υποσυστημάτων που διαθέτει on-chip. Για παράδειγμα, κάποιοι ακροδέκτες της θύρας A μπορούν να χρησιμοποιηθούν ως αναλογικές είσοδοι του μετατροπέα αναλογικού σήματος σε ψηφιακό (A/D). Ορισμένοι ακροδέκτες της θύρας C αποτελούν επίσης ακροδέκτες για τη σειριακή επικοινωνία (UART). Ακόμη, διαθέτει τρεις (3) μετρητές χρόνου που του δίνουν μεγάλες δυνατότητες σε εφαρμογές, όπου οι πολλαπλές μετρήσεις χρόνου είναι απαραίτητες.

Ο μικροελεγκτής αυτός είναι κατάλληλος για την παραγωγή παλμών ελεγχόμενης διάρκειας. Διαθέτει δύο χρονιστές/μετρητές (timer/counter) [TMR0,2] των 8-bits, καθώς

και έναν με 16-bits [TMR1]. Επίσης, διαθέτει έναν μετατροπέα Αναλογικού σήματος σε Ψηφιακό (Analog to Digital converter), με οκτώ αναλογικά κανάλια εισόδου και με ανάλυση 10-bits.

Μια ακόμη δυνατότητα του μικροελεγκτή, είναι η σειριακή επικοινωνία. Μάλιστα, διαθέτει δύο περιφερειακά, ένα για ασύγχρονη ή σύγχρονη επικοινωνία του τύπου USART (Universal Synchronous Asynchronous Receiver Transmitter), και ένα για σύγχρονη μόνο επικοινωνία, το οποίο ονομάζεται SSP (Synchronous Serial Port)-Σύγχρονη Σειριακή θύρα. Παρακάτω δίνονται μερικά επιπρόσθετα χαρακτηριστικά του PIC16F877.

Ο παρακάτω πίνακας 4.1 συνοψίζει τα παραπάνω χαρακτηριστικά.

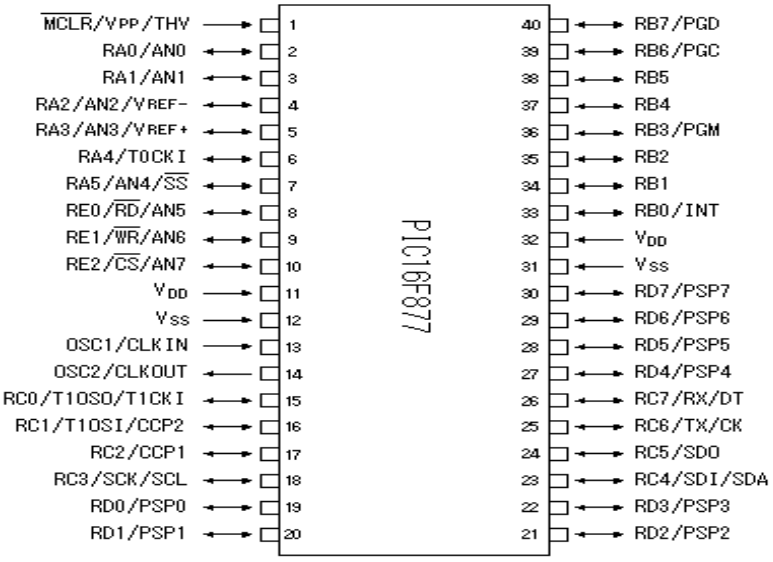
Χαρακτηριστικό	**PIC16F877**
Μέγιστη συχνότητα λειτουργίας (MHz)	20
Μνήμη προγράμματος Flash (14-bit λέξεις)	8K
Μνήμη δεδομένων (bytes)	368
EEPROM Μνήμη δεδομένων(bytes)	256
Πόρτες Εισόδου/Εξόδου	RA0-5 (6 ακροδέκτες) RB0-7 (8 ακροδέκτες) RC0-7 (8 ακροδέκτες) RD0-7 (8 ακροδέκτες) RE0-2 (3 ακροδέκτες)
Χρονιστές	3
Σειριακή επικοινωνία	USART
Παράλληλη επικοινωνία	PSP (Parallel Slave Port)
10-bit Αναλογική/Ψηφιακή μετατροπή	8 κανάλια
Σύνολο εντολών	35
Ακροδέκτες	40

Πίνακας 4.1: Χαρακτηριστικά του μικροελεγκτή PIC16F877

Όλα αυτά τα χαρακτηριστικά καθιστούν το κύκλωμα αυτό ιδανικό για υψηλού επιπέδου εφαρμογές σε βιομηχανικές συσκευές και καταναλωτικές εφαρμογές. Τέτοιες χρήσεις είναι σε συστήματα ελέγχου, συστήματα συλλογής δεδομένων, συναγερμούς, αναγνώστες καρτών (card readers) και πολλά άλλα.

4.1.3 Παρουσίαση του μικροελεγκτή PIC16F877

Ο μικροελεγκτής PIC16F877 αποτελείται συνολικά από 40 pins (είκοσι σε κάθε μεριά του). Το διάγραμμα ακροδεκτών του PIC16F877 φαίνεται στο σχήμα 4.2



Σχήμα 4.2 Ο μικροελεγκτής PIC16F877 και οι ακροδέκτες του

4.1.4 Δομή του PIC

Η δομή του PIC μπορεί να χωριστεί σε δυο μέρη, τον πυρήνα (core) και τις περιφερειακές μονάδες του (peripheral units). Ο πυρήνας του μικροελεγκτή αποτελείται από τα στοιχεία εκείνα, τα οποία είναι απολύτως απαραίτητα για τη λειτουργία του. Οι περιφερειακές μονάδες είναι ενσωματωμένες στο μικροελεγκτή και είναι αυτές που τον κάνουν να διαφέρει από έναν μικροεπεξεργαστή.

Στον πυρήνα του PIC16F877 ανήκουν οι εξής μονάδες:

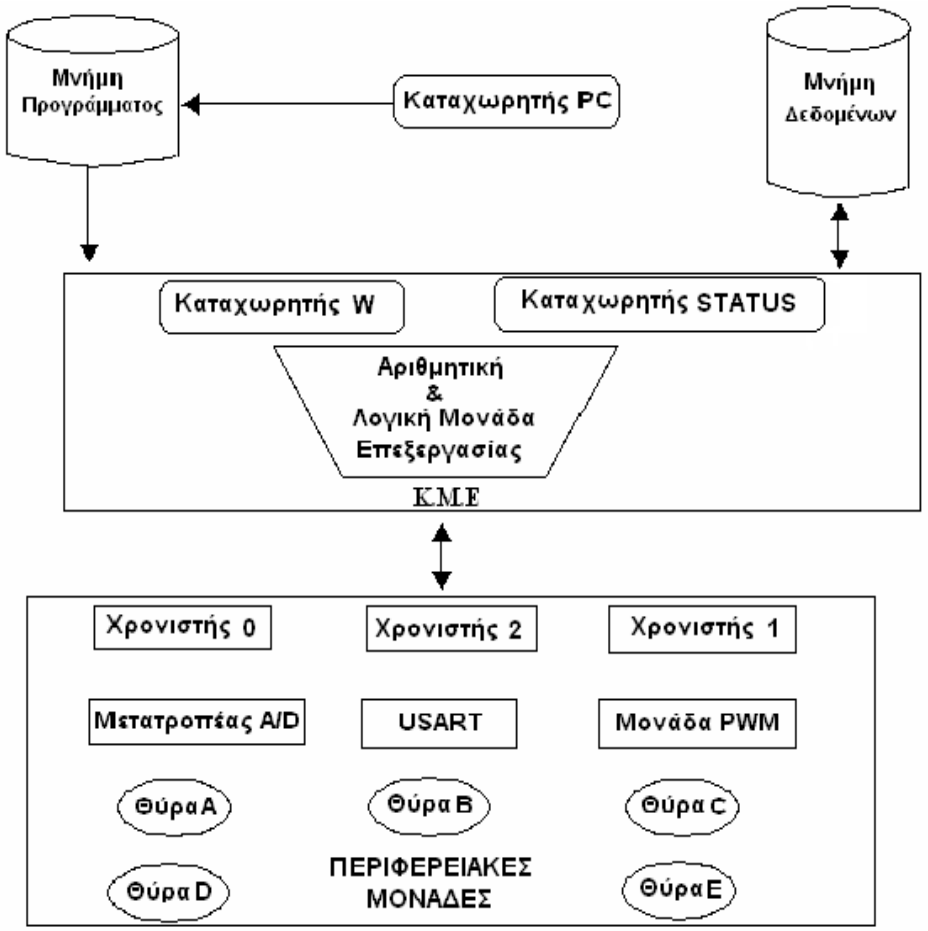
- Κεντρική μονάδα επεξεργασίας
- Μνήμη
- Λειτουργίες διακοπών

Στις περιφερειακές μονάδες ανήκουν:

- Οι θύρες εισόδου/εξόδου γενικής χρήσης
- Οι μετρητές χρόνου (τρεις μονάδες)
- Η μονάδα διαμόρφωσης πλάτους (PWM)
- Οι θύρες σειριακής επικοινωνίας (τρεις)
- Η θύρα παράλληλης επικοινωνίας
- Οι συγκριτές
- Η μονάδα παραγωγής τάσης αναφοράς
- Ο μετατροπέας αναλογικού σήματος σε ψηφιακό

4.1.5 Κεντρική μονάδα επεξεργασίας (CPU)

Στη μνήμη του PIC υπάρχει πλήρης διαχωρισμός μεταξύ της μνήμης εντολών και μνήμης προγράμματος. Η κεντρική μονάδα επεξεργασίας, η γνωστή και ως CPU (Central processing unit), εκτελεί τις εντολές του προγράμματος που έχουμε αποθηκεύσει αφού τις καλέσει από τη μνήμη προγράμματος. Από τη μνήμη αυτή οι εντολές προσκομίζονται με τη σειρά, αποκωδικοποιούνται και εκτελούνται. Εδώ, πρέπει να σημειώσουμε ότι ο PIC αναγνωρίζει τριάντα πέντε (35) εντολές προγραμματισμού στις οποίες θα αναφερθούμε αναλυτικότερα παρακάτω.



Σχήμα 4.3 Βασική δομή του PIC

Η κεντρική μονάδα επεξεργασίας μπορεί να θεωρηθεί και ως μια Αριθμητική και Λογική Μονάδα (Arithmetic Logic Unit), η οποία εκτελεί αριθμητικές πράξεις (πρόσθεση και η αφαίρεση) καθώς και λογικές πράξεις (AND, OR, XOR, κτλ). Η ALU δέχεται και επιστρέφει δεδομένα από και προς τους διάφορους καταχωρητές. Επίσης, ένας αριθμός καταχωρητών ειδικής χρήσης είναι υπεύθυνος για τον έλεγχο και τη λειτουργία της. Η μονάδα επεξεργάζεται δεδομένα μήκους οκτώ δυαδικών ψηφίων (8-bits).

Για την εύρεση επόμενης εντολής που πρέπει να προσκομισθεί για να εκτελεσθεί, η κεντρική μονάδα επεξεργασίας χρησιμοποιεί τον μετρητή προγράμματος (program counter). Κάθε φορά που ο μικροεπεξεργαστής φέρνει μια εντολή από την μνήμη, το περιεχόμενο του μετρητή προγράμματος αυξάνεται κατά ένα. Έτσι λοιπόν ο καταχωρητής αυτός περιέχει πάντα την διεύθυνση της επόμενης προς εκτέλεση εντολής.

Ο μετρητής προγράμματος του PIC έχει μήκος 13 bits. Άρα, μπορούν να αναπαρασταθούν 2^{13} αριθμοί, δηλαδή από 0 έως 8191. Αυτοί οι αριθμοί αντιπροσωπεύουν τις αντίστοιχες διευθύνσεις στη μνήμη προγράμματος. Συνεπώς ένας PIC μπορεί να έχει μέχρι 8 KB μνήμης προγράμματος.

4.1.6 Ο καταχωρητής w.

Σημαντικό κομμάτι του πυρήνα του PIC είναι ο καταχωρητής w ή αλλιώς ο καταχωρητής εργασίας. Ο καταχωρητής αυτός μπορεί να θεωρηθεί ως συσσωρευτής ή καταχωρητής προσωρινής αποθήκευσης. Στο καταχωρητή w δεν μπορεί να γίνει προσπέλαση άμεσα, αλλά μετακινούνται τα περιεχόμενά του σε άλλους καταχωρητές, στους οποίους η πρόσβαση είναι άμεση. Κάθε αριθμητική πράξη που επιτελείται στο PIC, χρησιμοποιεί τον καταχωρητή w. Παραδείγματος χάρη αν θέλουμε να προσθέσουμε τα περιεχόμενα δύο καταχωρητών, πρέπει να μεταφέρουμε το περιεχόμενο του πρώτου καταχωρητή στον w και στη συνέχεια να το προσθέσουμε με το περιεχόμενο του δεύτερου καταχωρητή.

Οι ελεγκτές PIC διαθέτουν αρκετά ισχυρή αρχιτεκτονική από την άποψη ότι το αποτέλεσμα μιας αριθμητικής πράξης μπορεί να αποθηκευτεί ή στον καταχωρητή "w", ή στον καταχωρητή προέλευσης των δεδομένων. Αποθηκεύοντας το αποτέλεσμα στον καταχωρητή προέλευσης εξαλείφεται ουσιαστικά η ανάγκη χρήσης πρόσθετων εντολών για την αποθήκευση αυτή.

4.1.7 Ο καταχωρητής STATUS

Ο καταχωρητής STATUS (Καταχωρητής Κατάστασης) αποτελεί τον βασικό καταχωρητή που χρησιμοποιείται για τον έλεγχο της εκτέλεσης του προγράμματος. Ο καταχωρητής αυτός χωρίζεται σε τρία τμήματα.

Το πρώτο τμήμα περιέχει τις σημαίες (Flags) ή bits κατάστασης της εκτέλεσης (τις "Z", "dc" και "C"). Τα τρία αυτά bits απεικονίζουν τη κατάσταση της εκτέλεσης του προγράμματος. Το bit "Z", ή η σημαία του μηδενός (Zero Flag), τίθεται σε λογικό "1" όταν το αποτέλεσμα κάποιας πράξης γίνει ίσο με το μηδέν (add, sub, clear, πράξεις λογικής επεξεργασίας). Η σημαία κρατουμένου (Carry Flag) "C", τίθεται σε λογικό "1" όταν το αποτέλεσμα κάποιας πράξης γίνει μεγαλύτερο από 255 (0x0FF), και χρησιμοποιείται για να δηλώσει ότι πρέπει να ενημερωθούν και τα υψηλότερης τάξης bytes που είναι σχετικά με το αποτέλεσμα.

Η σημαία δεκαδικού κρατουμένου (Digit Carry Flag) "dc", τίθεται σε λογικό "1" όταν τα τέσσερα λιγότερο σημαντικά bits (nibble) του αποτελέσματος μιας αριθμητικής πράξης, δώσουν αριθμό μεγαλύτερο από 15.

Αυτά τα bits, που αντιπροσωπεύουν οι σημαίες κατάστασης, μπορούν να διαβαστούν και να εγγραφούν, καθώς και να ενημερώνεται η κατάστασή τους, ανάλογα με την εκτέλεση της κάθε εντολής.

Τα δύο bits, «*RP0»* και «*RP1»* (b5, b6) χρησιμοποιούνται αποκλειστικά για τη προσπέλαση των υψηλότερων σελίδων της μνήμης. Είναι δυνατή τόσο η ανάγνωση όσο και η εγγραφή σε αυτά.

7	**6**	**5**	**4**	**3**	**2**	**1**	**0**
0	RP1	RP0	-	-	Z	DC	C

Πίνακας 4.3: Ο καταχωρητής STATUS και τα bits που τον αποτελούν

Στο σχήμα 4.3 παρουσιάζεται η κεντρική μονάδα επεξεργασίας μαζί με τα στοιχεία του PIC που συνδέονται άμεσα με αυτήν.

4.1.8 Μνήμη

Στη σχεδίαση μικροεπεξεργαστών και μικροελεγκτών έχουν επικρατήσει δύο αρχιτεκτονικές. Στην πρώτη χρησιμοποιείται μία μνήμη τόσο για την αποθήκευση του προγράμματος όσο και για την αποθήκευση των δεδομένων (αρχιτεκτονική Von-Neumann). Στη δεύτερη χρησιμοποιούνται δύο ξεχωριστές μνήμες. Η μία χρησιμοποιείται για την αποθήκευση του προγράμματος και λέγεται μνήμη προγράμματος ενώ η άλλη για την αποθήκευση των δεδομένων και λέγεται μνήμη δεδομένων (αρχιτεκτονική Harvard).

Ο μικροελεγκτής PIC ακολουθεί την αρχιτεκτονική Harvard. Στην αρχιτεκτονική αυτή εντολές και δεδομένα κινούνται σε ξεχωριστούς διαδρόμους (διαύλους), με αποτέλεσμα αυτό να μπορεί να γίνει όχι μόνον με πολύ μεγαλύτερη ταχύτητα αλλά ακόμη και την ίδια χρονική στιγμή. Αντίθετα, στην πρώτη αρχιτεκτονική εντολές και δεδομένα μοιράζονται τον ίδιο διάδρομο, με αποτέλεσμα να ελαττώνεται η ταχύτητα μεταφοράς τους. Επιπλέον, το πλεονέκτημα της δεύτερης αρχιτεκτονικής να χρησιμοποιεί ξεχωριστούς χώρους μνήμης για την αποθήκευση των δεδομένων και του προγράμματος, δίνει τη δυνατότητα χρησιμοποίησης μνημών με διαφορετικό μήκος λέξης. Έτσι, στην περίπτωση του PIC, η μνήμη προγράμματος έχει μήκος λέξης δεκατεσσάρων (14) δυαδικών ψηφίων (bits) αντί των οκτώ (8) της μνήμης των δεδομένων, με σκοπό όλες οι εντολές να κωδικοποιούνται σε μία λέξη.

Το μέγεθος της μνήμης προγράμματος κυμαίνεται από 2 ως 8 KBytes και συνήθως είναι τύπου Flash. Η συγκεκριμένη τεχνολογία επιτρέπει όχι μόνον την εγγραφή αλλά και το σβήσιμο της μνήμης να γίνεται με ηλεκτρικό τρόπο. Αυτό σημαίνει ότι ο

προγραμματισμός του μικροελεγκτή γίνεται εύκολα ακόμη κι όταν αυτός βρίσκεται συνδεδεμένος στο κύκλωμα της εκάστοτε εφαρμογής.

Το μέγεθος της μνήμης δεδομένων αποτελείται από τρία τμήματα, με μέγεθος 128 Bytes το καθένα, δηλαδή 384 Bytes συνολικά. Το κάθε τμήμα αποτελείται τόσο από καταχωρητές γενικού όσο και ειδικού σκοπού. Μερικοί από τους καταχωρητές ειδικού σκοπού χρησιμοποιούνται για τον έλεγχο του πυρήνα του PIC ενώ άλλοι για τον έλεγχο των περιφερειακών του.

	File Address		File Address		File Address		File Address
Indirect addr.(*)	00h	Indirect addr.(*)	80h	Indirect addr.(*)	100h	Indirect addr.(*)	180h
TMR0	01h	OPTION_REG	81h	TMR0	101h	OPTION_REG	181h
PCL	02h	PCL	82h	PCL	102h	PCL	182h
STATUS	03h	STATUS	83h	STATUS	103h	STATUS	183h
FSR	04h	FSR	84h	FSR	104h	FSR	184h
PORTA	05h	TRISA	85h		105h		185h
PORTB	06h	TRISB	86h	PORTB	106h	TRISB	186h
PORTC	07h	TRISC	87h		107h		187h
PORTD(1)	08h	TRISD(1)	88h		108h		188h
PORTE(1)	09h	TRISE(1)	89h		109h		189h
PCLATH	0Ah	PCLATH	8Ah	PCLATH	10Ah	PCLATH	18Ah
INTCON	0Bh	INTCON	8Bh	INTCON	10Bh	INTCON	18Bh
PIR1	0Ch	PIE1	8Ch	EEDATA	10Ch	EECON1	18Ch
PIR2	0Dh	PIE2	8Dh	EEADR	10Dh	EECON2	18Dh
TMR1L	0Eh	PCON	8Eh	EEDATH	10Eh	Reserved(2)	18Eh
TMR1H	0Fh		8Fh	EEADRH	10Fh	Reserved(2)	18Fh
T1CON	10h		90h		110h		190h
TMR2	11h	SSPCON2	91h		111h		191h
T2CON	12h	PR2	92h		112h		192h
SSPBUF	13h	SSPADD	93h		113h		193h
SSPCON	14h	SSPSTAT	94h		114h		194h
CCPR1L	15h		95h		115h		195h
CCPR1H	16h		96h		116h		196h
CCP1CON	17h		97h	General Purpose Register 16 Bytes	117h	General Purpose Register 16 Bytes	197h
RCSTA	18h	TXSTA	98h		118h		198h
TXREG	19h	SPBRG	99h		119h		199h
RCREG	1Ah		9Ah		11Ah		19Ah
CCPR2L	1Bh		9Bh		11Bh		19Bh
CCPR2H	1Ch		9Ch		11Ch		19Ch
CCP2CON	1Dh		9Dh		11Dh		19Dh
ADRESH	1Eh	ADRESL	9Eh		11Eh		19Eh
ADCON0	1Fh	ADCON1	9Fh		11Fh		19Fh
	20h		A0h		120h		1A0h
General Purpose Register 96 Bytes		General Purpose Register 80 Bytes		General Purpose Register 80 Bytes		General Purpose Register 80 Bytes	
	7Fh	accesses 70h-7Fh	EFh F0h	accesses 70h-7Fh	16Fh 170h	accesses 70h - 7Fh	1EFh 1F0h
Bank 0		Bank 1	FFh	Bank 2	17Fh	Bank 3	1FFh

Πίνακας 4.4: Χάρτης μνήμης μικροελεγκτή PIC16F877

Οι καταχωρητές είναι ένα από τα βασικότερα στοιχεία της αρχιτεκτονικής ενός μικροελεγκτή. Η ευκολία και οι δυνατότητες προγραμματισμού του μικροελεγκτή έχουν άμεση σχέση με το πλήθος, το είδος και τις δυνατότητες των καταχωρητών του. Κάθε εντολή ενός προγράμματος χρησιμοποιεί έναν τουλάχιστον καταχωρητή.

Υπάρχουν δύο ομάδες καταχωρητών. Η πρώτη ομάδα, που βρίσκεται στις χαμηλότερες διευθύνσεις, περιέχει καταχωρητές ειδικών λειτουργιών (**special function registers**), όπως αυτών του ελέγχου των περιφερειακών που βρίσκονται ενσωματωμένα στον μικροελεγκτή. Η δεύτερη ομάδα περιέχει καταχωρητές γενικής χρήσης και αναφέρεται ως αρχείο καταχωρητών γενικού σκοπού (**general purpose register file** – 368 προσπελάσιμοι καταχωρητές).

Κατά την εκκίνηση, ο μικροελεγκτής βλέπει εξ' ορισμού την σελίδα μνήμης μηδέν. Εάν χρειαστεί να προσπελάσουμε τους καταχωρητές που βρίσκονται στη σελίδα μνήμης "1" θα πρέπει να το ορίσουμε με την ακόλουθη εντολή:

bsf STATUS RP0,

η οποία θέτει σε λογικό '1' το bit RP0 του καταχωρητή STATUS. Για να προσπελάσουμε έπειτα τους καταχωρητές της σελίδας μνήμης μηδέν, χρησιμοποιούμε την εντολή

bcf STATUS RP0 (Αναφορά στις εντολές θα γίνει σε επόμενο κεφάλαιο).

4.1.9 Εντολές

Οι μικροελεγκτές PIC ακολουθούν την αρχιτεκτονική RISC και έχουν συνολικά 35 εντολές μήκους μιας λέξης 14 bits. Έτσι σε αντίθεση με τους μικροελεγκτές αρχιτεκτονικής CISC, ο PIC εκτελεί την κάθε εντολή σε ένα κύκλο μηχανής, με αποτέλεσμα τη σημαντική βελτίωση της ταχύτητας επεξεργασίας. Στο σημείο αυτό ας τονίσουμε ότι μοναδική εξαίρεση αποτελούν οι εντολές διακλάδωσης, οι οποίες εκτελούνται σε δύο κύκλους μηχανής.

4.1.10 Θύρες εισόδου / εξόδου

Ο μικροελεγκτής έχει πέντε θύρες εισόδου / εξόδου, των οκτώ (8) δυαδικών ψηφίων (bits). Αυτές μπορούν να χρησιμοποιηθούν είτε σαν απλές θύρες είτε σαν θύρες των υπόλοιπων περιφερικών που διαθέτει. Η τέταρτη και πέμπτη θύρα μπορούν να χρησιμοποιηθούν και για παράλληλη επικοινωνία. Στην κάθε θύρα αντιστοιχεί ένας καταχωρητής ελέγχου (TRIS) εισόδου / εξόδου και ένας καταχωρητής δεδομένων (PORT). Για παράδειγμα, στη θύρα Β αντιστοιχούν οι καταχωρητές TRISB, PORTB. Τόσο ο καταχωρητής TRIS όσο και ο καταχωρητής δεδομένων PORT της θύρας, εμφανίζονται ως τυπικοί καταχωρητές μέσα στο χώρο διευθύνσεων. Η εγγραφή δεδομένων στη θύρα εξόδου μπορεί να γίνει οποιαδήποτε στιγμή, αλλά πρώτα πρέπει να

έχουμε ορίσει την θύρα ως έξοδο. Αυτό γίνεται μηδενίζοντας το αντίστοιχο bit ελέγχου του καταχωρητή TRIS (θέτοντας 1 κάποιο bit του καταχωρητή TRIS, ορίζουμε τον αντίστοιχο ακροδέκτη ως είσοδο, ενώ θέτοντας 0 το bit, ορίζουμε τον ακροδέκτη ως έξοδο).

4.1.11 Οι εντολές της γλώσσας Assembly του PIC

Ο πίνακας 4.4 παραθέτει τις 35 εντολές που αποτελούν τη γλώσσα Assembly των μικροελεγκτών PIC. Κάθε εντολή αποτελείται από το μνημονικό μέρος του κωδικού εντολής (opcode) και το όρισμα (καταχωρητή f ή σταθερό αριθμό k) πάνω στον οποίο ενεργεί η εντολή. Το σύμβολο d στις εντολές του πίνακα 4.4 σημαίνει "destination" (προορισμός) και είναι 0 αν ο προορισμός του αποτελέσματος είναι ο καταχωρητής εργασίας w, ενώ είναι 1 αν ο προορισμός του αποτελέσματος είναι ο ίδιος ο καταχωρητής f. Στην τελευταία στήλη φαίνονται οι σημαίες που επηρεάζονται κατά την εκτέλεση των εντολών.

Οι εντολές των ελεγκτών PIC μπορούν να χωριστούν σε τέσσερις κατηγορίες.

1. Αριθμητικές εντολές
2. Ελέγχου εκτέλεσης
3. Ελέγχου του μικροεπεξεργαστή
4. Χειρισμού των bit των καταχωρητών

Η πρώτη κατηγορία αποτελείται από τις «αριθμητικές» εντολές, που περιλαμβάνουν τη πράξη της πρόσθεσης και της αφαίρεσης ανάμεσα στα περιεχόμενα καταχωρητών, καθώς και πράξεις αύξησης ή μείωσης των τιμών τους και πράξεις σε επίπεδο bit.

Στην επόμενη κατηγορία εντολών, ανήκουν οι εντολές «ελέγχου εκτέλεσης». Αυτές, τις αποτελούν οι εντολές άλματος (goto), κλήσης υπορουτίνας (call) και οι εντολές επιστροφής (return) από κάποια υπορουτίνα, καθώς επίσης και οι εντολές διακλάδωσης υπό όρους.

Στην επόμενη κατηγορία ανήκουν οι εντολές «ελέγχου του μικροεπεξεργαστή». Οι εντολές αυτές επηρεάζουν βασικά τη λειτουργία του επεξεργαστή και τα κυκλώματα που συσχετίζονται με αυτόν.

Η τελευταία κατηγορία αποτελείται από τις εντολές «χειρισμού των bit των καταχωρητών» (τοποθέτηση ή μηδενισμός bit). Με τις εντολές αυτές ελέγχουμε άμεσα, κάθε ένα ξεχωριστό bit των καταχωρητών. Η πιο προφανής χρήση των εντολών αυτών είναι ο άμεσος έλεγχος επιμέρους κυκλωμάτων και ακροδεκτών του μικροελεγκτή.

Μνημονικό / Τελεστής		Λειτουργία	Σημαία - Flag
Εντολές χειρισμού ψηφιολέξεων - Byte oriented file register operations			
ADDWF	f, d	Πρόσθεσε το W και το f	C, DC, Z
ANDWF	f, d	Κάνε την λογική πράξη AND ανάμεσα στο W και το f	Z
CLRF	f	Μηδένισε το f	Z
CLRW	-	Μηδένισε το W	Z
COMF	f, d	Φτιάξε το συμπλήρωμα του f και αποθήκευσέ το στο d	Z
DECF	f, d	Μείωσε την τιμή του f	Z
DECFSZ	f, d	Μείωσε την τιμή του f, παρέκαμψε την επόμενη εντολή αν ο f γίνει 0	
INCF	f, d	Αύξησε την τιμή του f	Z
INCFSZ	f, d	Αύξησε την τιμή του f, παρέκαμψε την επόμενη εντολή αν ο f γίνει 0	
IORWF	f, d	Κάνε την λογική πράξη IOR ανάμεσα στο W και το f	Z
MOVF	f, d	Μετέφερε το περιεχόμενο του f	Z
MOVWF	f	Μετέφερε το περιεχόμενο του W στο f	
NOP	-	Εντολή δίχως λειτουργία (απλή χρονική καθυστέρηση ενός κύκλου μηχανής)	
RLF	f, d	Μετέφερε προς τα αριστερά το περιεχόμενο του f μέσο του ψηφίου Carry	C
RRF	f, d	Μετέφερε προς τα δεξιά το περιεχόμενο του f μέσο του ψηφίου Carry	C
SUBWF	f, d	Αφαίρεσε το W από το f	C, DC, Z
SWAPF	f, d	Αντιμετάθεσε τα δύο μισά της ψηφιολέξης (byte) στο f	
XORWF	f, d	Λογική πράξη XOR ανάμεσα στο W και το f	Z

Εντολές χειρισμού ψηφίων - Bit oriented file register operations			
BCF	f, b	Μηδένισε το ψηφίο b του καταχωρητή f	
BSF	f, b	Κάνε λογικό 1 το ψηφίο b του καταχωρητή f	
BTFSC	f, b	Εξέτασε το ψηφίο b του καταχωρητή f, παρέκαμψε την επόμενη εντολή αν είναι 0	
BTFSS	f, b	Εξέτασε το ψηφίο b του καταχωρητή f, παρέκαμψε την επόμενη εντολή αν είναι 1	
Εντολές πράξεων με σταθερούς αριθμούς. Εντολές ελέγχου προγράμματος			
ADDLW	k	Πρόσθεσε τον σταθερό αριθμό k με το W	C, DC, Z
ANDLW	k	Κάνε την λογική πράξη AND ανάμεσα στο k και το W	Z
CALL	k	Κάλεσε την υπορουτίνα στη διεύθυνση k	
CLRWDT	-	Μηδένισε τον επιτηρητή Watchdog Timer	$\overline{TO}$, $\overline{PD}$
GOTO	k	Πήγαινε και εκτέλεσε την εντολή που υπάρχει στην διεύθυνση k	
IORLW	k	Κάνε την λογική πράξη IOR ανάμεσα στο k και το W	Z
MOVLW	k	Μετέφερε το περιεχόμενο του k στο W	
RETFIE	-	Επέστρεψε στην διεύθυνση που ήσουν πριν συμβεί η διακοπή (interrupt)	
RETLW	k	Επέστρεψε από υπορουτίνα και φόρτωσε τον σταθερό αριθμό k στο W	
RETURN	-	Επέστρεψε από υπορουτίνα	
SLEEP	-	Ενεργοποίησε την λειτουργία χαμηλής κατανάλωσης (Sleep - κατανάλωση 2μA)	$\overline{TO}$, $\overline{PD}$
SUBLW	k	Αφαίρεσε το περιεχόμενο του W από το σταθερό αριθμό k	C, DC, Z
XORLW	k	Κάνε την λογική πράξη XOR ανάμεσα στο k και το W	Z

Πίνακας 4.4: Οι εντολές του μικροελεγκτή PIC

f είναι ο καταχωρητής στον οποίο ενεργεί η εντολή, ενώ το d (0 ή 1) δηλώνει τον προορισμό του αποτελέσματος (w ή f). Στην τελευταία στήλη βρίσκονται οι σημαίες που επηρεάζονται από την εντολή.

Οι PIC υποστηρίζουν τρία είδη διευθυνσιοδότησης: την άμεση, την έμμεση και την απευθείας. Με την απευθείας διευθυνσιοδότηση, μεταφέρουμε τα δεδομένα από ένα καταχωρητή σε μια θέση μνήμης. Με την άμεση διευθυνσιοδότηση μεταφέρουμε αριθμητικά δεδομένα (literals) στον καταχωρητή εργασίας W. Τέλος, στην έμμεση διευθυνσιοδότηση, προσπελαύνουμε ένα καταχωρητή μέσω ενός βοηθητικού, ο οποίος περιέχει την διεύθυνση του καταχωρητή που θέλουμε να εγγράψουμε ή να διαβάσουμε.

4.1.12 Υπορουτίνες

Ένα δομημένο πρόγραμμα χρησιμοποιεί υπορουτίνες, εφόσον υπάρχει ομάδα εντολών που επαναλαμβάνεται συχνά. Η έννοια της υπορουτίνας στην γλώσσα assembly είναι ίδια με αυτή των γλωσσών υψηλού επιπέδου. Το όνομα της υπορουτίνας δηλώνεται με μία ετικέτα στην πρώτη γραμμή της ρουτίνας. Κάθε υπορουτίνα τελειώνει με την εντολή return, με την οποία επιστρέφει στο κυρίως πρόγραμμα από την οποία καλείται.

CALL ΟΝΟΜΑ Ή ΔΙΕΥΘΥΝΣΗ: Η εντολή αυτή εκτελεί ως επόμενη εντολή την πρώτη εντολή της υπορουτίνας η οποία βρίσκεται στη θέση που δείχνουν το όνομα ή η διεύθυνση και ο Program Counter (PC) φορτώνεται με την διεύθυνση της υπορουτίνας.

Σε σχέση με την εντολή GOTO, η παρούσα εντολή κάνει μια επιπλέον ενέργεια. Αποθηκεύει τη διεύθυνση της αμέσως επόμενης εντολής, την PC+1, σε μια ειδική μνήμη που λέγεται σωρός. Έτσι, όταν ο PIC τελειώσει την εκτέλεση της υπορουτίνας, θα μπορέσει να επιστρέψει μετά την CALL στην επόμενη εντολή. Αυτή η εντολή εκτελείται σε δύο κύκλους εντολής.

RETURN: Μετά την εκτέλεση της, μια υπορουτίνα πρέπει να επιστρέψει στην επόμενη εντολή της CALL και να συνεχίσει ομαλά την εκτέλεση του κυρίως προγράμματος. Στην προηγούμενη εντολή είδαμε ότι για να γίνει αυτό, χρειάζεται η διεύθυνση να έχει αποθηκευτεί στον σωρό, έτσι ώστε να είναι δυνατή η ανάκτηση της, με την κατάλληλη εντολή.

4.2 Εργαστηριακό Μέρος

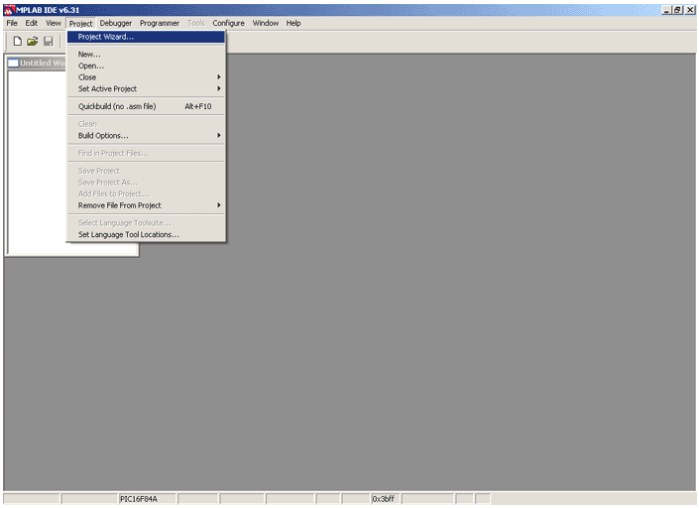
4.2.1 Εισαγωγή στον εξομοιωτή MPLAB

Η εργαστηριακή άσκηση που ακολουθεί έχει σκοπό να εξοικειώσει τους σπουδαστές με τις βασικές εντολές προγραμματισμού των μικροελεγκτών PIC και με το περιβάλλον συμβολομετάφρασης και προσομοίωσης του κώδικα σε γλώσσα Assembly. Θα χρησιμοποιήσουμε το περιβάλλον MPLAB, που αποτελεί ελεύθερο λογισμικό της εταιρίας Microchip.

Μπορείτε να καταφορτώσετε το λογισμικό MPLAB v.8 (legacy) από το site της εταιρίας Microchip (www.microchip.gr).

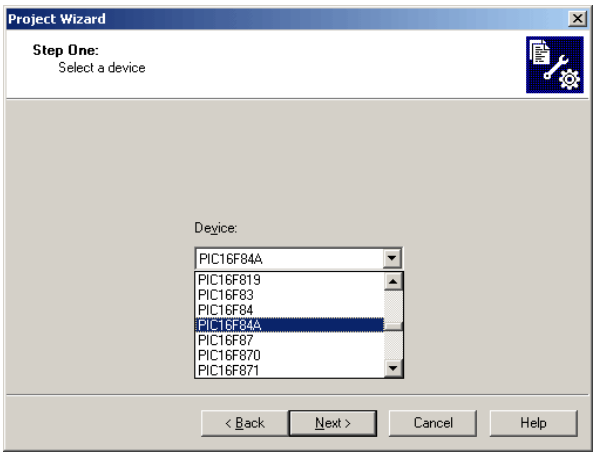
Βήμα 1ο : Δημιουργία Σχεδίου Εργασίας

Ξεκινείστε το λογισμικό MPLAB, ρυθμίζουμε τις λεπτομέρειες του παραθύρου έργου επιλέγοντας από το μενού PROJECT είτε την επιλογή NEW είτε την επιλογή PROJECT WIZARD. Στην δεύτερη περίπτωση θα εμφανιστεί ένα παράθυρο όπως αυτό στην εικόνα 4.1



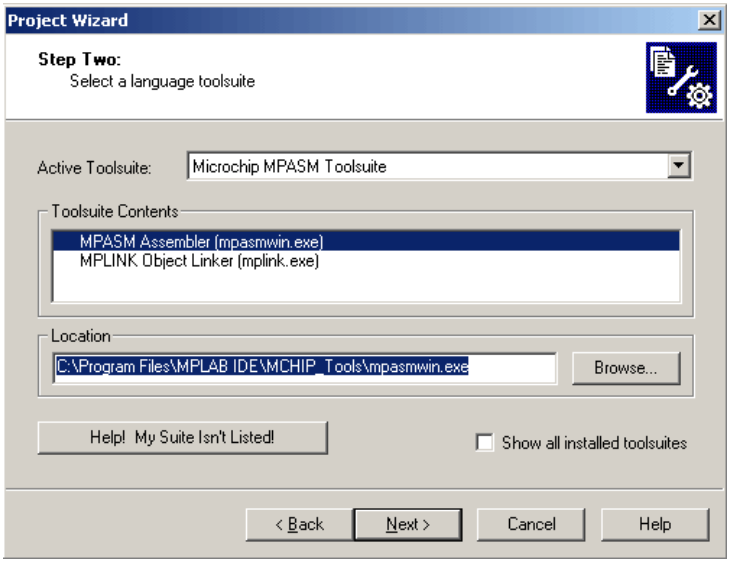
Εικόνα 4.1: Το παράθυρο έναρξης project στο MPLAB

Στη συνέχεια θα πρέπει να επιλέξουμε τον τύπο του μικροελεγκτή που θα χρησιμοποιήσουμε όπως φαίνεται στην εικόνα 4.2



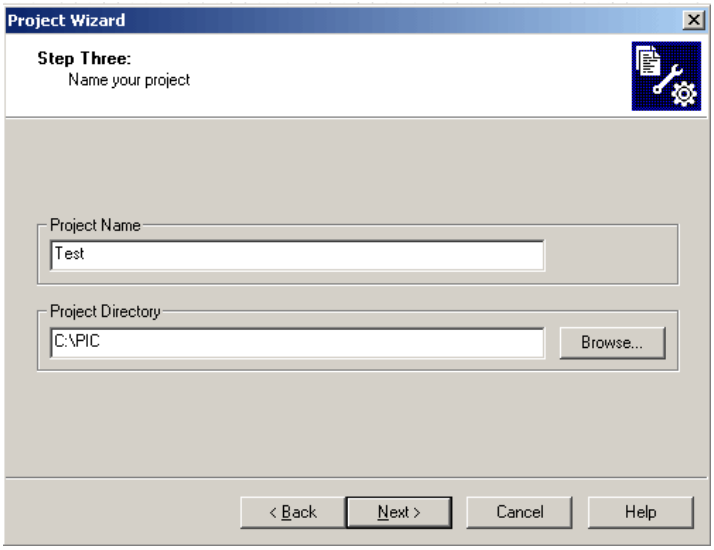
Εικόνα 4.2: Το παράθυρο επιλογής μικροελεγκτή

Με το επόμενο βήμα, καθορίζουμε την γλώσσα προγραμματισμού που θα χρησιμοποιήσουμε που στην περίπτωσή μας είναι η Συμβολική γλώσσα Assembly για τους μικροελεγκτές PIC και για το λόγο αυτό επιλέγουμε να φορτωθεί ο συμβολομεταφραστής MPASMWIN.EXE με καθορισμένη διαδρομή αναζήτησης που οδηγεί στο φάκελο που έχει εγκατασταθεί το λογισμικό MPLAB.



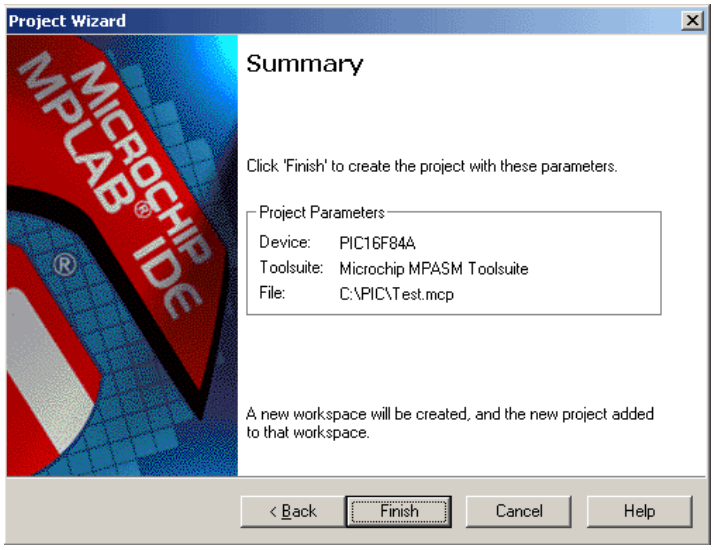
Εικόνα 4.3: Το παράθυρο επιλογής της γλώσσας προγραμματισμού

Τέλος, δίνουμε όνομα στο έργο που θα δημιουργήσουμε καθώς και τον φάκελο στο οποίο το έργο θα αποθηκευτεί σύμφωνα με το παράθυρο της εικόνας 4.4.

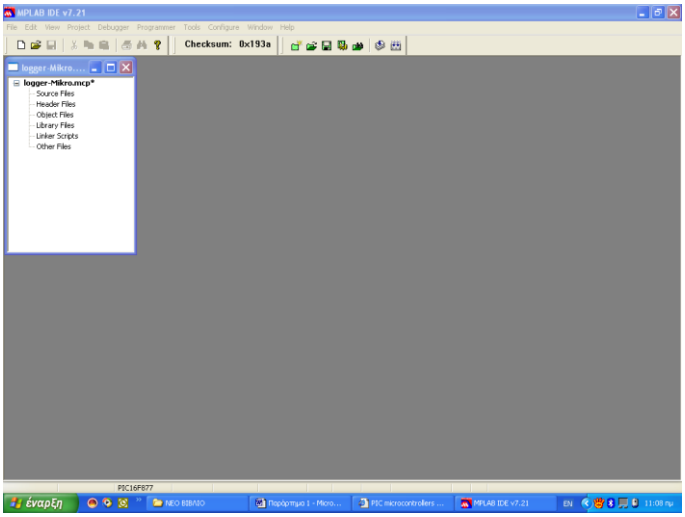


Εικόνα 4.4: Το παράθυρο απόδοσης ονόματος στο έργο.

Στον παραπάνω φάκελο αυτό, θα αποθηκεύουμε και όλα τα αρχεία με τους κώδικες assembly που θα δημιουργήσουμε στην συνέχεια. Η διαδικασία αυτή ολοκληρώνεται με την διαδοχική επιλογή των πλήκτρων NEXT και FINISH οπότε εμφανίζεται ένα παράθυρο με την περίληψη όλων των ρυθμίσεων που δώσαμε έως τώρα.



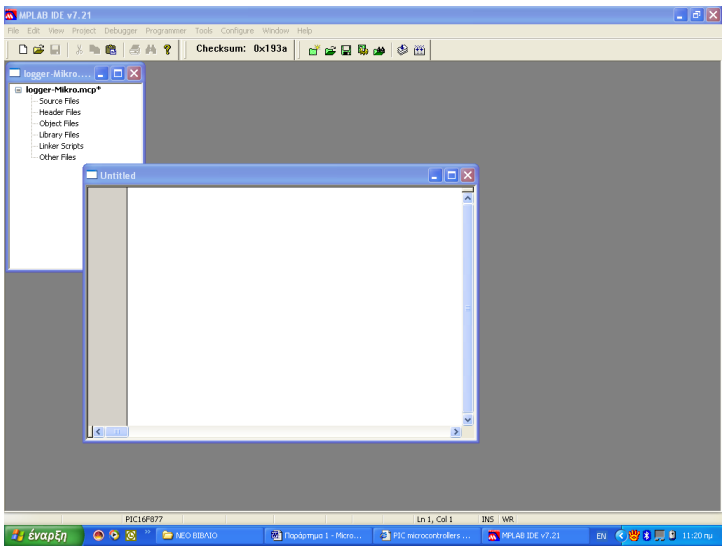
Εικόνα 4.5: Το παράθυρο περίληψης των ρυθμίσεων του έργου



Εικόνα 4.6: Το παράθυρο του συντάκτη (editor) για την συγγραφή κώδικα

Βήμα 2ο Δημιουργία .ASM Αρχείου

Μετά την ολοκλήρωση της δημιουργίας του έργου, εμφανίζεται ένα παράθυρο όμοιο με εκείνο της εικόνας 4.6. Για να γράψουμε τον κώδικα assembly πρέπει να δημιουργήσουμε ένα αρχείο με την βοήθεια του συντάκτη (editor) του MPLAB. Αυτό γίνεται από το μενού FILE και την επιλογή NEW, οπότε ανοίγει ένα λευκό παράθυρο του συντάκτη στο οποίο γράφουμε τις εντολές του κώδικα assembly που μας ενδιαφέρει.



Εικόνα 4.7: Το παράθυρο έργου (project)

Στο ίδιο μενού διακρίνουμε τις γνωστές επιλογές των WINDOWS για την δημιουργία νέου αρχείου (NEW) , για την φόρτωση ήδη αποθηκευμένου αρχείου

(OPEN) και για την αποθήκευση αρχείου (SAVE). Επίσης στο ίδιο μενού προβλέπεται και η δημιουργία, φόρτωση και αποθήκευση σε αρχείο του χώρου εργασίας που έχουμε δημιουργήσει (Workspace, αρχεία με προέκταση .mcw) που θα περιλαμβάνει όλες τις πληροφορίες τόσο για το έργο (project με προέκταση .mcp)) όσο και για τα αρχεία assembly (με προέκταση .asm)που συνδέονται με το έργο.

Επιλέγουμε File-New. Στον Editor που ανοίγει πληκτρολογούμε το αρχείο σε γλώσσα Assembly που περιγράφει την εφαρμογή μας. Το παρακάτω πρόγραμμα μεταφέρει τιμές στον καταχωρητή εργασίας και σε θέσεις μνήμης και εκτελεί απλές αριθμητικές πράξεις.

```
#include "p16F877.inc"
        ;Initialization
        Org 0                   ;Start from program memory address 0
        movlw b'00001010'       ;Load in w decimal value '10'
        movwf 22h               ;Trasfer w value to memory 22h
        movlw b'00000101'       ;Load in w decimal '5'
        movwf 23h               ;Trasfer w value to memory 23h

        ;main
        movf 22h,w              ;Transfer content of memory 22h to w
        addwf 23h,w             ;Add w with memory 23h
        movwf 24h               ;Transfer result to memory 24h
        end
```

Μετά την συγγραφή του κώδικα και έχοντας επιλεγμένο το παράθυρο του συντάκτη (η γραμμή τίτλου του παραθύρου να είναι έντονη) από την επιλογή Αποθήκευση (Save) αποθηκεύουμε το αρχείο του κώδικα assembly με την προέκταση .asm. στον ίδιο φάκελο που έχει αποθηκευτεί και το αρχείο έργου (project).

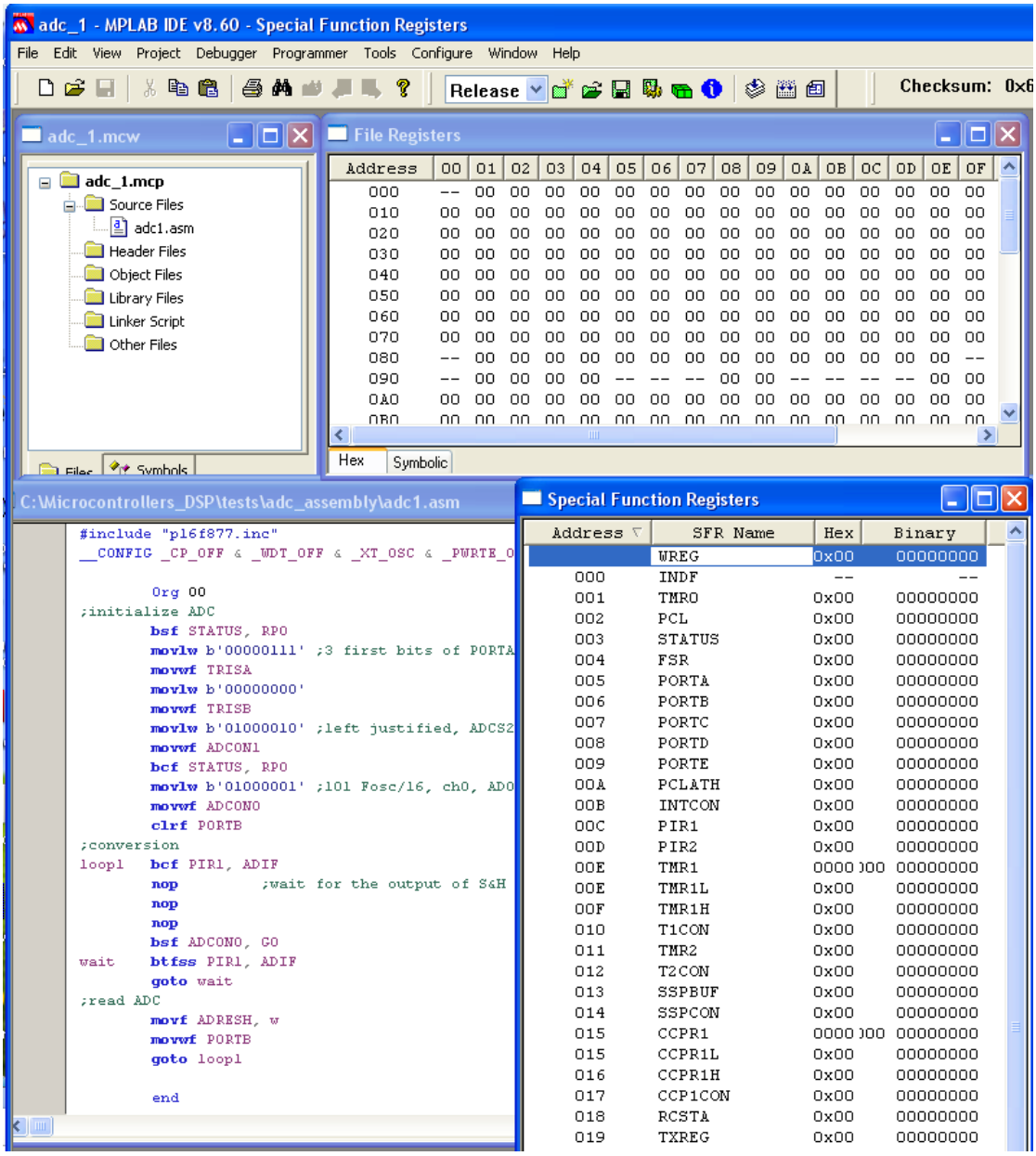
Το επόμενο βήμα είναι να ενημερώσουμε το αρχείο έργου με τα αρχεία κώδικα που θα περιλαμβάνει. Αυτό γίνεται κάνοντας διαδοχικά δεξί κλίκ πάνω στην επιλογή Source Files του παραθύρου project και στην συνέχεια επιλέγοντας την επιλογή Add Files. Στο παράθυρο που εμφανίζεται επιλέγουμε το ήδη αποθηκευμένο αρχείο κώδικα (.asm) που έχουμε από πριν δημιουργήσει σύμφωνα με το παραπάνω βήμα.

Βήμα 3ο Δημιουργία δεκαεξαδικού αρχείου (.hex)

Επιλέξτε Project-Build all. Θα προκύψουν πιθανά λάθη. Όταν διορθωθούν τα λάθη θα δημιουργηθεί το τελικό αρχείο προγραμματισμού.

Βήμα 4ο. Προσομοίωση

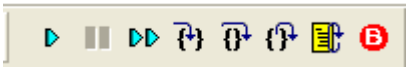
Από την επιλογή View, ανοίξτε τα παράθυρα File Registers και Special Function Registers, όπως φαίνεται στην παρακάτω οθόνη.



Εικόνα 4.8 Παράθυρα για την προσομοίωση του κώδικα

Το παράθυρο File Registers εμφανίζει τα περιεχόμενα της μνήμης RAM του μικροελεγκτή. Το επιλέγουμε πατώντας το πλήκτρο RAM στη γραμμή εργαλείων. Το παράθυρο SFR εμφανίζει τις τιμές των βασικών καταχωρητών ειδικού σκοπού. Κάθε φορά που αλλάζει τιμή ένας καταχωρητής ειδικού σκοπού το περιεχόμενό του εμφανίζεται με κόκκινο.

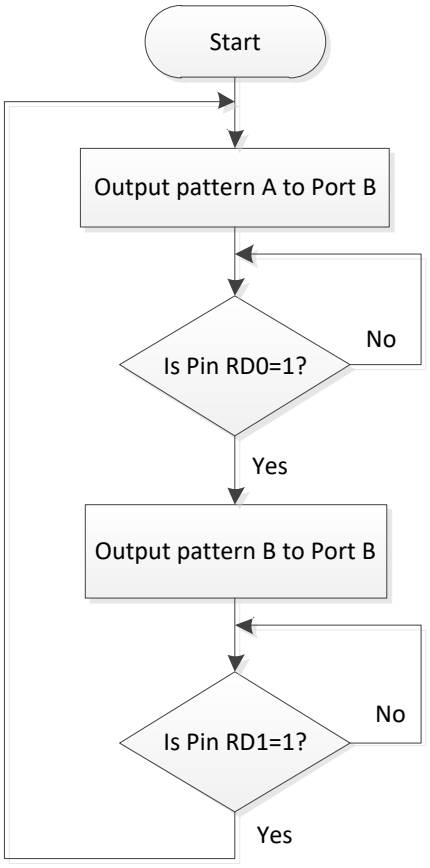
Επιλέξτε Debugger-Select tool-MPLAB SIM. Στη συνέχεια επιλέγοντας Debugger-Step into, η εκτέλεση του κώδικα προσομοιώνεται εντολή προς εντολή. Το ίδιο αποτέλεσμα,



πετυχαίνουμε με τα εργαλεία προσομοίωσης. Κάθε φορά μπορούμε να μεταβούμε στην εκτέλεση της επόμενης εντολής πατώντας το εικονίδιο 
Παρατηρείστε πως αλλάζουν τιμές οι καταχωρητές και οι θέσεις μνήμης.

4.2.2 Προγραμματισμός των θυρών PIO του PIC16F877.

Το παρακάτω πρόγραμμα επιδεικνύει τον τρόπο προγραμματισμού των θυρών Parallel I/O (PIO) του PICια είσοδο και για έξοδο. Το διάγραμμα ροής του κώδικα είναι το εξής:



Σχήμα 4.4 Το διάγραμμα ροής της εφαρμογής εισόδου/εξόδου

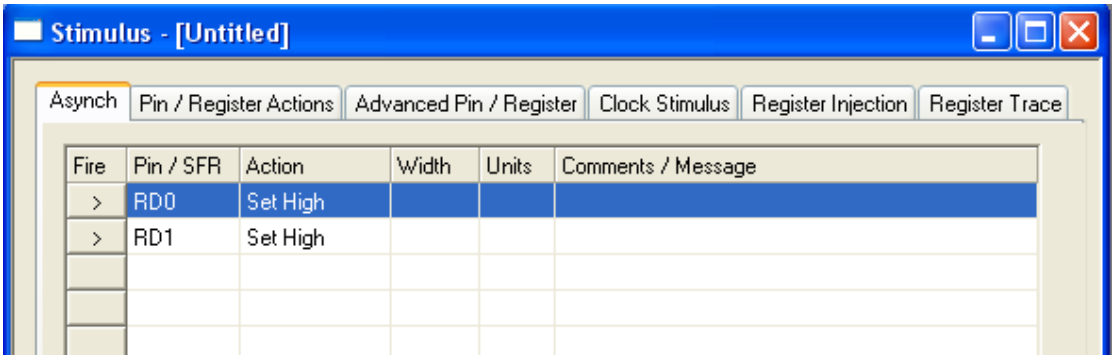
Δημιουργείστε σχέδιο εργασίας με όνομα erg42.mcp

Πληκτρολογήστε τον κώδικα και αποθηκεύστε τον με όνομα erg42.asm. Προσομοιώστε την λειτουργία του.

```
#include "p16F877.inc"
        ;Initialization
        Org 0                   ;Start from program memory address 0
        bsf STATUS, RP0         ;Change to bank1
        movlw b'00000000'       ;Make all pins of portB Outputs
        movwf TRISB
        movlw b'00011111'       ;Make 5 pins of portD inputs
        movwf TRISD
        bcf STATUS, RP0         ;Return to bank0

main    movlw b'01010101'       ;Output to portB
        movwf PORTB
        btfss PORTD, 0          ;Wait until pin0 of portD receives 1
        goto $-1
        movlw b'10101010'       ;If pin0 of portD is zero change output combination to portB
        movwf PORTB
        btfss PORTD, 1          ;Wait until pin1 of portD receives 1
        goto $-1
        goto main
        end
```

Χρησιμοποιείστε την δυνατότητα Debugger-Stimulus-New workbook για να δώσετε εικονικά τιμές στους ακροδέκτες της διάταξης.

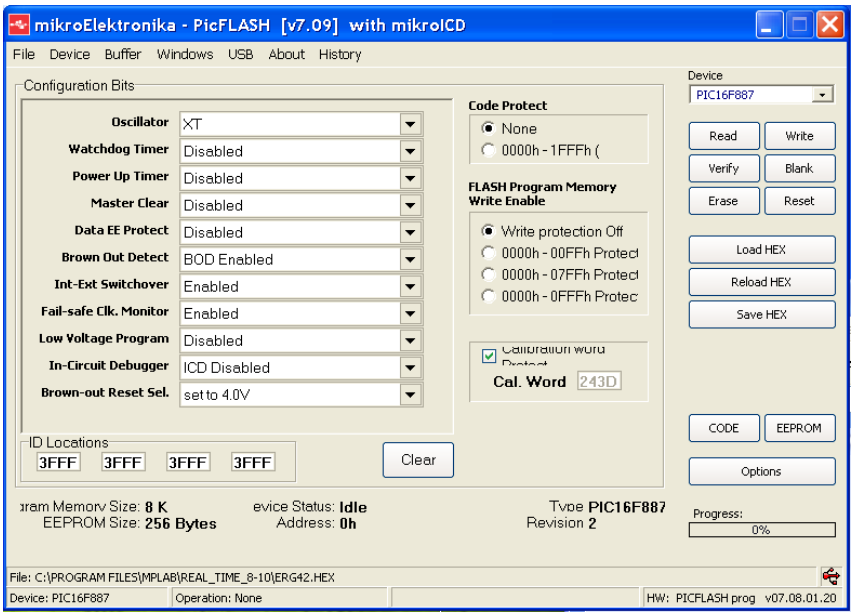


Με κλικ πάνω στις κενές στήλες αντιστοιχούμε ακροδέκτη θύρας και ορίζουμε την ενέργεια που επιτελείται (Action), όταν πατήσουμε στο πλήκτρο Fire.

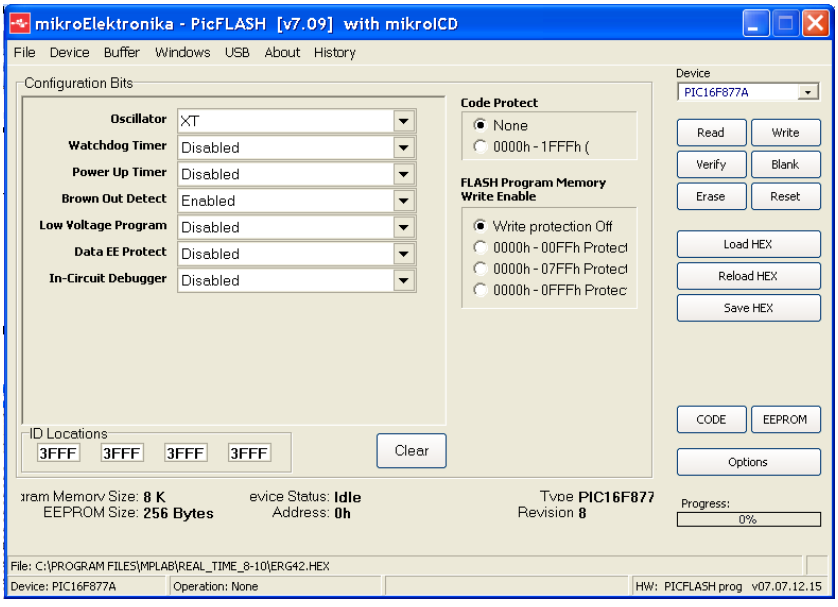
4.2.3 Προγραμματισμός της διάταξης

Δημιουργείστε το δεκαεξαδικό αρχείο για τον προγραμματισμό της διάταξης, με την επιλογή Project-Build all.
Συνδέστε τον υπολογιστή με την αναπτυξιακή πλακέτα EasyPIC5. Η σύνδεση γίνεται μέσω θύρας USB.

Ανοίξτε στην οθόνη τον προγραμματιστή PICflash. Καλέστε το αρχείο .hex που μόλις δημιουργήσατε. Αν ο μικροελεγκτής που φέρει η αναπτυξιακή πλακέτα είναι ο PIC16F887, κάνετε τις παρακάτω επιλογές των ψηφίων διαμόρφωσης (Configutation Bits), όπως φαίνεται στον πίνακα.



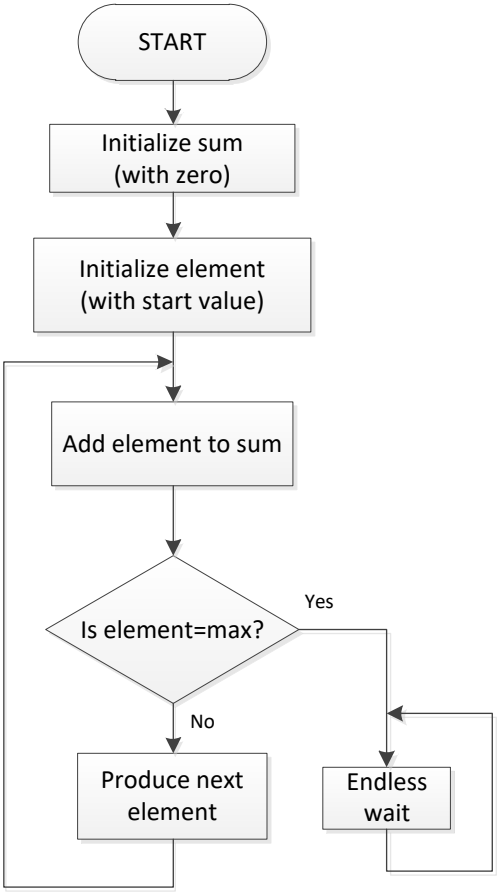
Προγραμματίστε την διάταξη επιλέγοντας Write.
Αν η πλακέτα φέρει τον μικροελεγκτή PIC16F877A κάνετε τις ρυθμίσεις των bits διαμόρφωσης όπως παρακάτω:



Α. Παρατηρείστε την κατάσταση των LEDs της θύρας Β.
Β. Πιέστε τον διακόπτη εισόδου RD0. Τι παρατηρείτε;
Γ. Πιέστε τον διακόπτη εισόδου RD1. Επιβεβαιώστε την καλή λειτουργία του προγράμματος που γράψατε.

ΥΠΟΧΡΕΩΤΙΚΗ ΕΡΓΑΣΙΑ

Να μελετήσετε το παρακάτω διάγραμμα ροής. Το πρόγραμμα προσθέτει όλους τους ακεραίους που βρίσκονται στο διάστημα 1 έως 10 και βρίσκει το άθροισμα (sum).



;This program adds up all integers in interval [start, max]
;For example from 1 to 10 [1, 10]

```
;The sum is stored in memory location 21h
;8.10.2012 by John Kalomiros

#include "p16f877.inc"

element     equ 20h   ;define address to store current added element
sum         equ 21h   ;define address for sum
start       equ d'1'  ;start int value (decimal)
max         equ d'10' ;max int value (decimal)

        Org 00
;Initialize memory
        movlw 00
```

```
        movwf sum       ;initialize sum
        movlw start
        movwf element   ;load first element
;main code starts here
main    movf element,w
        addwf sum,1     ;add current element to sum
        movf element, w
        sublw max       ;compare element with stop value
        btfsc STATUS, Z ;if element exceeds max value, then stop
loop    goto loop       ;endless loop - wait here
        incf element    ;otherwise produce next element
        goto main       ;continue loop
        end
```

Αφού κατανοήσετε τον κώδικα να τον προσομοιώσετε στο MPLAB.
Να μεταφέρετε τα παράθυρα προσομοίωσης στην εργασία σας.
Στη συνέχεια να τροποποιήσετε τον κώδικα ώστε στο τέλος της διαδικασίας το άθροισμα sum να εμφανίζετε στη θύρα B.