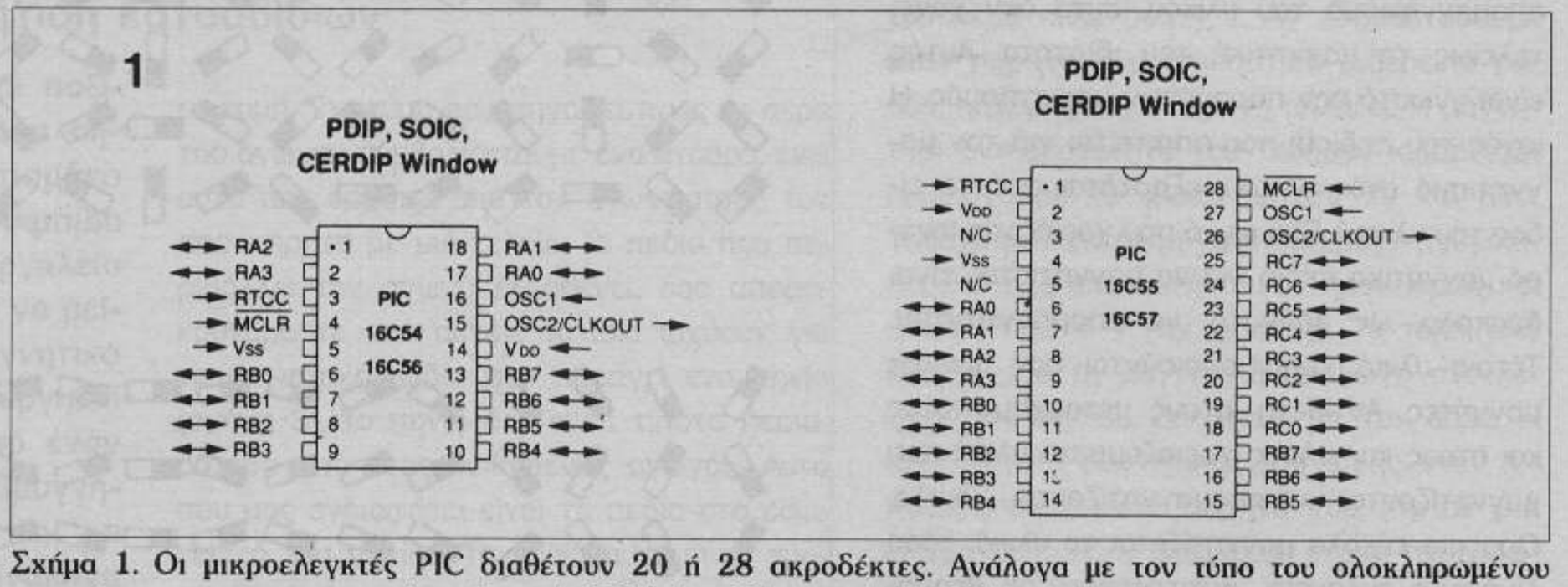
Εκπαιδευτική σειρά

Προγραμματισμός του PIC

Μέρος 1ο.

Εισαγωγή στην αρχιτεκτονική και στον προγραμματισμό

Οι μικροελεγκτές (μαζί με τα κατάλληλα προγράμματα) μπορούν να αντικαταστήσουν πολύπλοκα ηλεκτρονικά κυκλώματα. Οι μικροελεγκτές χρησιμοποιούνται ευρέως γιά τον έλεγχο μηχανημάτων και οργάνων μετρήσεων. Οι χρονικά εξαρτημένες λειτουργίες, που υλοποιούνται δύσκολα με απλές πύλες, κατασκευάζονται πανεύκολα με τη βοήθεια των μικροελεγκτών. Σε αυτή την εκπαιδευτική σειρά του ΕΛΕΚΤΟΡ θα ασχοληθούμε με τους μικροελεγκτές της οικογένειας PIC-16C5x. Θα αναφερθούμε στην αρχιτεκτονική και κυρίως, στον προγραμματισμό των μικροελεγκτών.



Σχήμα 1. Οι μικροελεγκτές PIC διαθέτουν 20 ή 28 ακροδέκτες. Ανάλογα με τον τύπο του ολοκληρωμένου μπορούν να προγραμματιστούν μιά φορά (OPT) ή πολλές φορές (EPROM με παραθυράκι).

Η οικογένεια ολοκληρωμένων PIC-16C5x κατασκευάζεται από την εταιρεία Arizona Microchip Technology. Πρόκειται γιά μικροελεγκτές τύπου CMOS με ενσωματωμένη μνήμη προγράμματος δεδομένων. Σα μνήμη προγράμματος χρησιμοποιείται μιά EPROM (ή PROM) με μήκος λέξης 12 ψηφίων (αντί των συνηθισμένων μνημών, με μήκος λέξης 8 ψηφίων). Το μέγεθος της μνήμης προγράμματος και δεδομένων των διάφορων μικροελεγκτών της οικογένειας 16C5x, φαίνεται στον πίνακα 1. Οι μικροελεγκτές χρησιμοποιούν στατική μνήμη. Αυτό σημαίνει, ότι η συχνότητα χρονισμού μπορεί να μηδενιστεί (0 Hz), χωρίς να χάσουμε τα δεδομένα. Η διάταξη της μνήμης προγράμματος με λέξεις των 12 ψηφίων επιτρέπει την ύπαρξη εντολών 3 byte, που εκτελούνται σ' έναν κύκλο μηχανής. Ο μικροελεγκτής καταλαβαίνει 33 απλές εντολές. Ολες οι εντολές, εκτός από τις εντολές διακλάδωσης, εκτελούνται ταχύτατα στη διάρκεια ενός κύκλου μηχανής. Από αυτό, η εταιρεία Microchip χαρακτηρίζει το μικροελεγκτή σαν RISC (Reduced Instruction Set Computer), υπολογιστής με περιορισμένο πλήθος εντολών). Αυτό μπορεί να δυσκολεύει λίγο στον προγραμματισμό, αλλά η εκτέλεση των εντολών γίνεται ταχύτατα. Οι μικροελεγκτές διαφέρουν, εκτός από το μέγεθος της μνήμης, στο πλήθος των ακροδεκτών εισόδου / εξόδου (I/O) και στη συχνότητα χρονισμού. Το είδος του ταλαντωτή, φαίνεται από τα δυό τελευταία γράμματα στην ονομασία του ολοκληρωμένου (XT, RC, HS, LP, βλέπε πίνακα 2). Οι μικροελεγκτές διατίθενται σε διάφορα μεγέθη ολοκληρωμένων (σχήμα 1), με ή χωρίς παράθυρο. Τα ολοκληρωμένα χωρίς παράθυρο είναι φτηνότερα, αλλά μπορούν να προγραμματιστούν μόνο μιά φορά (έκδοση OTP, One Time Programmable). Το πρόγραμμα στα ολοκληρωμένα που έχουν παράθυρο μπορεί να σβηστεί με τη βοήθεια ενός λαμπτήρα υπεριώδους ακτινοβολίας (UV). Γιά να προστατευτεί το πρόγραμμα του ολοκληρωμένου από αντιγραφή, αρκεί να γράψουμε ένα συγκεκριμένο ψηφίο (flag).

Η αρχιτεκτονική του PIC.

Οι μικροελεγκτές έχουν αρχιτεκτονική Harvard, με ξεχωριστούς διαύλους γιά τη μνήμη προγράμματος και τη μνήμη δεδομένων. Καθώς ο μικροελεγκτής εκτελεί μιά εντολή, διαβάζει την επόμενη από τη μνήμη προγράμματος. Ετσι, επιταχύνεται η εκτέλεση του προγράμματος. Αυτή η διαδικασία (γνωστή με την ονομασία Look ahead) δεν μπορεί να εφαρμοστεί με τις εντολές διακλάδωσης. Γιαυτό το λόγο, ο χρόνος εκτέλεσης των εντολών διακλάδωσης διαρκεί δύο κύκλους μηχανής. Μ' αυτό το θέμα θα ασχοληθούμε αργότερα, όταν θα περιγράφουμε τις υπόλοιπες εντολές. Η δομή του μικροελεγκτή φαίνεται στο σχήμα 2. Στην πάνω αριστερή γωνία διακρίνονται η μνήμη προγράμματος με τον απαριθμητή εντολών PC, ο καταχωρητής και ο αποκωδικοποιητής εντολών. Από κάτω φαίνεται η μονάδα αριθμητικής επεξεργασίας (ALU, Arithmetic Logic Unit) με τον καταχωρητή εργα-

Πίνακας 1. Οι μικροελεγκτές της οικογένειας PIC-16C5x

Μικροελεγκτής	EPROM	RAM (*)	Ακροδ.I/O (**)
PIC 16C54	512 x 12	32 x 8	13
PIC 16C55	512 x 12	32 x 8	21
PIC 16C56	1 K x 12	32 x 8	13
PIC 16C57	2 K x 12	80 x 8	21

(*) Συμπερ. των SFR
(**) Συμπερ. των ακροδεκτών TRCC

Πίνακας 2. Διαχωρισμός ανάλογα με τη συχνότητα χρονισμού

Ονομασία	Min.	Max.	Παρατηρήσεις
-LP	DC	40 KHz	Χαμηλής ισχύος
-RC	DC	4 MHz	Ταλαντωτής RC
-XT	DC	4 MHz	Κρυσταλλικός ή κεραμικός ταλ.
-HS	DC	20 MHz*	Υψηλής ταχύτητας

* Εμπορικές και βιομηχανικές εφαρμογές.
Γιά την αυτοκινητοβιομηχανία 16 MHz

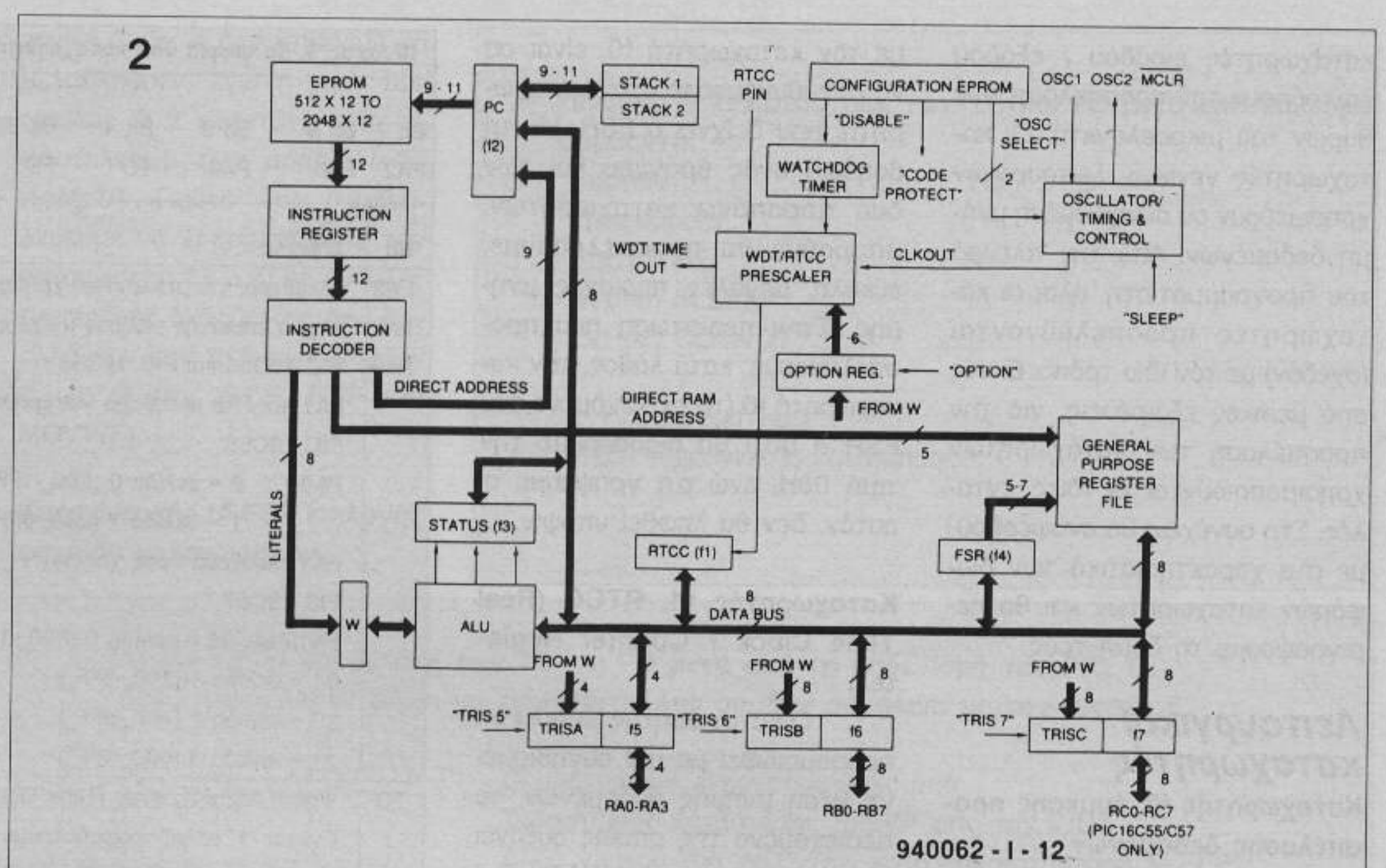
Πίνακας 3. Επίδραση των εντολών στα ψηφία του διαύλου διευθύνσεων

Ψηφίο του καταχ. PC	GOTO	Εντολή CALL	Μέσω του PC
A10	Bit 6 του f3	Bit 6 του f3	Bit 6 του f3
A9	Bit 5 του f3	Bit 5 του f3	Bit 5 του f3
A8	Από την εντολή	Πάντα 0	Πάντα 0
A7-A0	Από την εντολή	Από την εντολή	Από τη βαθμ. ALU

σίας (W register). Στη μέση, δεξιά, βρίσκεται η μνήμη δεδομένων (General Purpose Register File). Κάτω απ' αυτήν φαίνεται η μονάδα χρονισμού, με τον απαριθμητή / χρονιστή, τη μονάδα επιτήρησης (Watchdog) και τους καταχωρητές εισόδου / εξόδου.

Η μνήμη προγράμματος

Η μνήμη προγράμματος χωρίζεται σε σελίδες. Σε κάθε σελίδα μπορούν να αποθηκευτούν 512 εντολές. Οι μικροελεγκτές PIC 16C54 και PIC 16C55 έχουν μόνο μιά σελίδα. Ο PIC 16C56 έχει δύο, που διαχωρίζονται με τη βοήθεια της γραμμής διεύθυνσης A9. Τέλος, ο PIC 16C57 έχει τέσσερις σελίδες, που προσπελαύνονται μέσω των γραμμών διευθύνσεων A9 και A10. Στο σχήμα 3 φαίνεται η διάταξη της μνήμης προγράμματος. Ανάλογα με τον τύπο του μικροελεγκτή, η μνήμη προγράμματος προσπελαύνεται από τις 9 ως 11 γραμμές διευθύνσεων, που παρέχονται από τον απαριθμητή προγράμματος (PC, Program Counter). Προσοχή χρειάζεται όταν καλούμε υποπρογράμματα, εκτελούμε εντολές διακλάδωσης ή εντολές που μεταβάλλουν το περιεχόμενο του PC. Τα προβλήματα δημιουργούνται, καθώς το μήκος της λέξης δεδομένων είναι μόνο 8 ψηφία και οι τελεστές των εντολών δεν υπερβαίνουν τα 9 ψηφία. Στον πίνακα 3 φαίνεται ο τρόπος που σχηματίζεται η διεύθυνση προσπέλασης της μνήμης προγράμματος (δηλ. το περιεχόμενο του PC), όταν εκτελούνται οι παραπάνω εντολές. Οπως βλέπετε, αν εξαιρέσουμε την εντολή GOTO, οι άλλες δυό εντολές δεν μπορούν να προσπελάσουν το πρώτο μισό της σελίδας στην μνήμη προγράμματος (A8 = 0). Τα παραπάνω μας ενδιαφέρουν, κυρίως, όταν γράφουμε προγράμματα γιά τους μικροελεγκτές PIC 16C56 και PIC 16C57. Πριν εκτελέσουμε κάποια από τις παραπάνω εντολές, θα πρέπει να έχουμε καθορισεί σωστά την τιμή των ψηφίων 5 και 6 του καταχωρητή f3. Αυτά τα ψηφία δεν αλλάζουν τιμή, όταν, κατά την εκτέλεση του προγράμματος, ο απαριθμητής προγράμματος αλλάξει σελίδα μνήμης. Πριν χρησιμοποιηθούν, η τιμή τους πρέπει να καθορίζεται από το χρήστη.



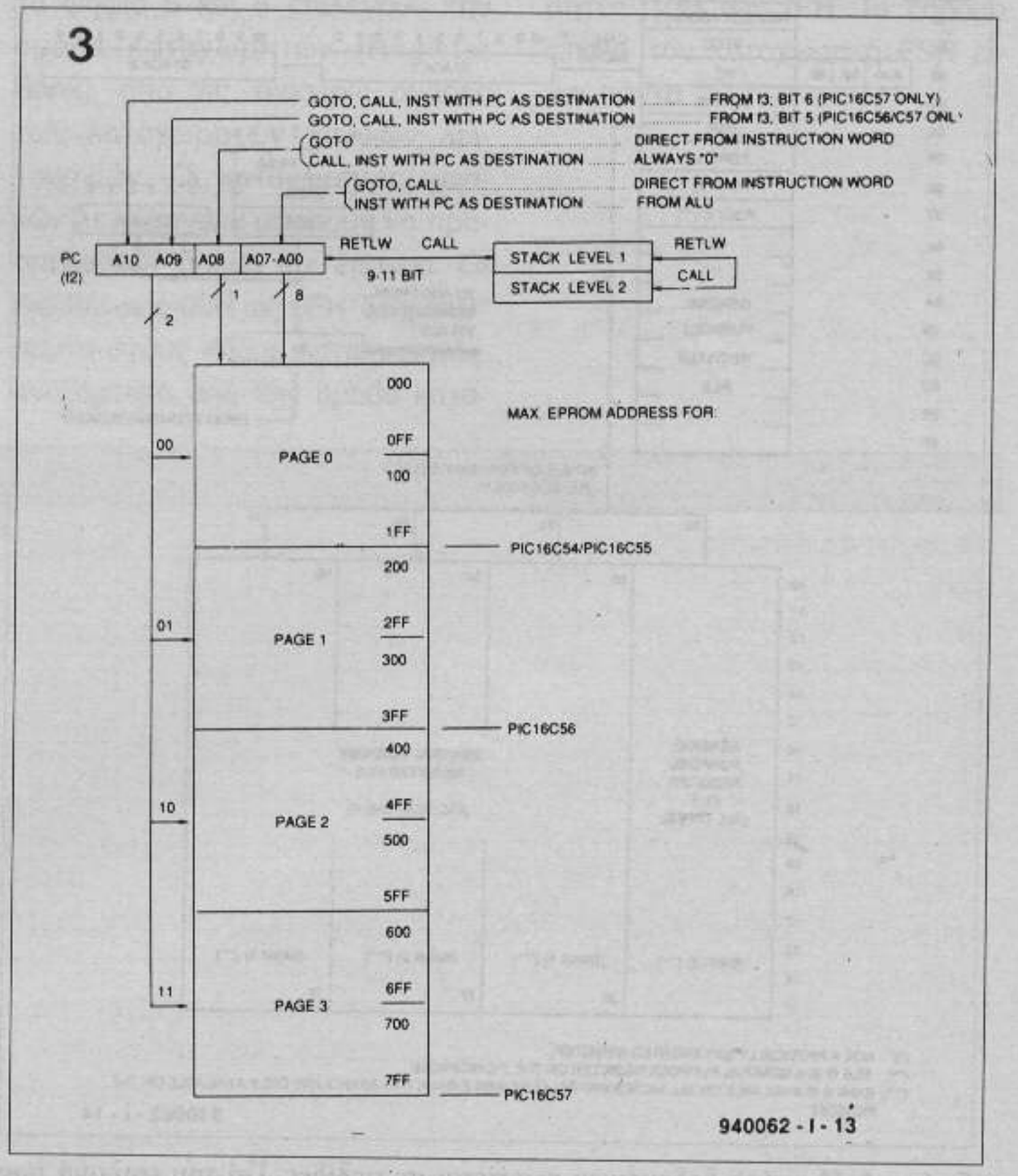
Σχήμα 2. Το δομικό διάγραμμα του PIC16C5x. Στο σχήμα φαίνονται οι διάφορες βαθμίδες και το εύρος των διαύλων του μικροελεγκτή.

Μνήμη δεδομένων, αρχείο καταχωρητών.

Η εταιρεία Microchip έχει δώσει στην εσωτερική μνήμη δεδομένων του μικροελεγκτή PIC την ονομασία αρχείο δεδομένων (Register File). Με τον όρο καταχωρητής εννοούμε μιά θέση μνήμης, που μπορεί να προσπελαθεί απευθείας από την ALU. Στην περίπτωση του μικροελεγκτή PIC, αυτό ισχύει γιά όλες τις θέσης της μνήμης δεδομένων. Το σύνολο των καταχωρητών θα μπορούσαμε να το χαρακτηρίσουμε σαν αρχείο. Κάτω από αυτές τις συνθήκες, ο όρος "αρχείο καταχωρητών" δικαιολογείται απόλυτα. Η μορφή του αρχείου καταχωρητών φαίνεται στο σχήμα 4. Η μνήμη χωρίζεται στους λειτουργικούς καταχωρητές (Operational Register), στους καταχωρητές εισόδου / εξόδου (θύρες) και στους καταχωρητές γενικών (General) και ειδικών λειτουργιών (Special Purpose Register). Οι λειτουργικοί καταχωρητές είναι υπεύθυνοι γιά τη λειτουργία του μικροελεγκτή. Τα αποτελέσματα των διαφόρων πράξεων, που εκτελούνται από το πρόγραμμα, αποθηκεύονται σ' αυτούς τους καταχωρητές. Οι

Πίνακας 4. Το μέγεθος των καταχωρητών PC και σωρού

Μικροελεγκτής	Μέγεθος PC και σωρού
PIC 16C54	9 bit (A8)
PIC 16C55	9 bit (A8)
PIC 16C56	10 bit (A8,A9)
PIC 16C57	11 bit (A8,A9,A10)



Σχήμα 3. Η διάταξη της μνήμης προγράμματος.

καταχωρητές εισόδου / εξόδου επιτρέπουν την προσπέλαση των θυρών του μικροελεγκτή. Οι καταχωρητές γενικών λειτουργιών χρησιμεύουν σα συνηθισμένη μνήμη δεδομένων. Από την πλευρά του προγραμματιστή, όλοι οι καταχωρητές προσπελαύνονται (σχεδόν) με τον ίδιο τρόπο. Εκτός από μερικές εξαιρέσεις, γιά την προσπέλαση των καταχωρητών χρησιμοποιούνται οι ίδιες εντολές. Στη συνέχεια θα αναφερθούμε στα χαρακτηριστικά των διαφόρων καταχωρητών και θα περιγράψουμε τη δομή τους.

Λειτουργικοί καταχωρητές

Καταχωρητής f0, έμμεσης προσπέλασης δεδομένων

Με τη βοήθεια του καταχωρητή f0 μπορούμε να προσπελάσουμε έμμεσα το αρχείο καταχωρητών. Η προσπέλαση γίνεται με τη βοήθεια του καταχωρητή FSR. Στον FSR αποθηκεύουμε τη διεύθυνση του καταχωρητή που θέλουμε να προσπελάσουμε. Κάθε φορά που γράφουμε ή διαβάζουμε τον καταχωρητή f0, είναι σα να απευθυνόμαστε στον καταχωρητή, που δείχνει ο FSR. Με τη βοήθεια ενός βρόγχου και των δυό παραπάνω καταχωρητών, μπορούμε να προσπελάσουμε, εύκολα, μεγάλες περιοχές μνήμης. Στην περίπτωση που προσπελάσουμε, κατά λάθος, τον καταχωρητή f0 (το περιεχόμενο του FSR = 00_H) θα διαβάσουμε την τιμή 00H, ενώ ότι γράψουμε σ' αυτόν, δεν θα ληφθεί υπόψιν.

Καταχωρητής f1, RTCC (Real Time Clock / Counter Register)

Ο καταχωρητής μπορεί να προσομοιωθεί με μιά συνηθισμένη θέση μνήμης δεδομένων, το περιεχόμενο της οποίας αυξάνει με τον ρυθμό που επιβάλλει ένα σήμα χρονισμού. Το σήμα χρονισμού μπορεί να ληφθεί από μιά εξωτερική πηγή, μέσω του ακροδέκτη RTCC, ή από τον ταλαντωτή του μικροελεγκτή. Ενας ενσωματωμένος διαιρέτης μπορεί να μειώσει τη συχνότητα του σήματος χρονισμού. Με τον καταχωρητή f1 θα ασχοληθούμε αργότερα, όταν αναφερθούμε στο χρονιστή επιτήρησης (Watchdog Timer).

Πίνακας 5. Τα ψηφία του καταχωρητή κατάστασης λειτουργίας f3.

Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0	Καταχωρητής
PA2	PA1	PA0	TO	PD	Z	DC	C	f 3

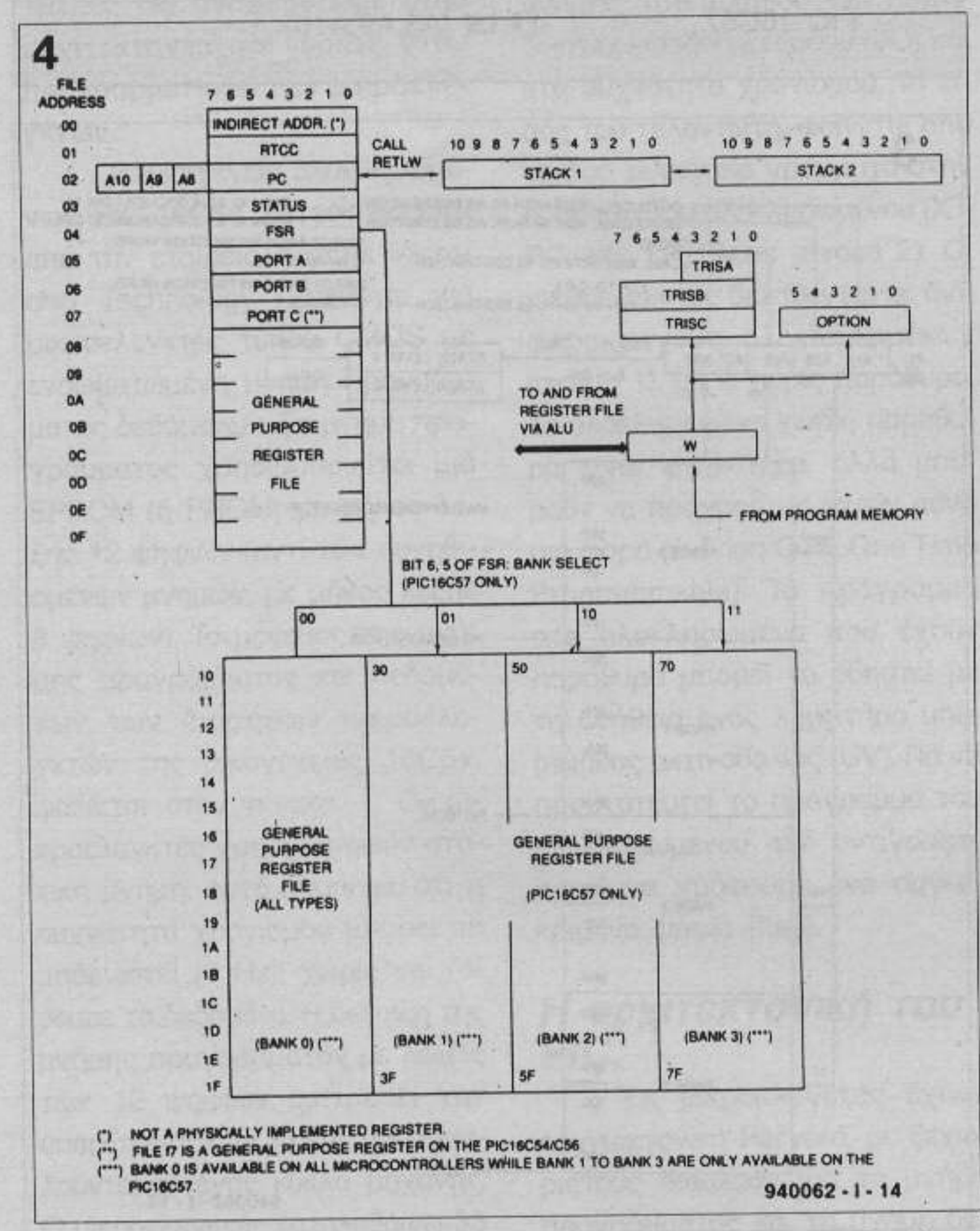
Bit	*Σημασία*
PA2	Ελεύθερο, γιά μελλοντική χρήση (*)
PA1	Ψηφία επιλογής σελίδας μνήμης
PA0	PIC 16C54 και PIC 16C55 PA1 και PA0 ελεύθερα γιά χρήση (*) PIC 16C56 PA0: 0 = σελίδα 0 (000_H-$1FF_H$) 1 = σελίδα 1 (200_H-$3FF_H$) PA1 ελεύθερο προς χρήση (*) PIC 16C57 PA1/PA0: 00 = σελίδα 0 (000_H-$1FF_H$) 01 = σελίδα 1 (200_H-$3FF_H$) 10 = σελίδα 2 (400_H-$5FF_H$) 11 = σελίδα 3 (600_H-$7FF_H$)
TO	Ψηφίο λήξης χρόνου (Time Out) Γίνεται "1" μόλις τροφοδοτούμε το μικροελεγκτή ή με τις εντολές CLRWRT και SLEEP. Μόλις υπερχειλίσει ο απαριθμητής επιτήρησης (WDT), το ψηφίο γίνεται "0". Το ψηφίο δεν μπορεί να ενεργοποιηθεί ή να απενεργοποιηθεί από καμιά άλλη εντολή.
PD	Ψηφίο διακοπής λειτουργίας (Power Down) Το ψηφίο γίνεται "1" μόλις τροφοδοτήσουμε το μικροελεγκτή ή εκτελέσουμε την εντολή CLRWRT. Η εντολή SLEEP το μηδενίζει. Το ψηφίο δεν μπορεί να ενεργοποιηθεί ή να απενεργοποιηθεί από καμιά άλλη εντολή.
Z	Ψηφίο μηδενικού αποτελέσματος (Zero) Γίνεται "1" όταν το αποτέλεσμα μιάς αριθμητικής ή λογικής πράξης είναι 00_H.
DC	Ψηφιακό υπόλοιπο/υπόλοιπο (Digital Carry/Borrow) Το υπόλοιπο του τρίτου ψηφίου, κατά την εκτέλεση των εντολών ADDWF και SUBWF.
C	Υπόλοιπο (Carry/Borrow) Το υπόλοιπο του έβδομου ψηφίου, κατά την εκτέλεση των εντολών ADDWF και SUBWF. Χρησιμοποιείται από τις εντολές RRF και RLF.

(*) Γιά να εξασφαλίσετε συμβατότητα με τους διάφορους μικροελεγκτές της οικογένειας PIC, δεν πρέπει να χρησιμοποιήσετε αυτά τα ψηφία στο πρόγραμμα.

Καταχωρητής f2, PC (απαριθμητής προγράμματος)

Ο απαριθμητής προγράμματος σχηματίζει τη διεύθυνση προσπέλασης της μνήμης προγράμματος. Ανάλογα με το μικροελεγκτή, στα 8 ψηφία του καταχωρητή προστίθεται ένα ως τρία ψηφία (A8...A10, βλέπε πίνακα 4). Μετά από την εκτέλεση κάθε εντολής φόρτωσης καταχωρητή ή αριθμητικής / λογικής πράξης, το περιεχόμενο του καταχωρητή PC αυξάνεται κατά 1. Ο καταχωρητής PC φορτώνεται ξεχωριστά όταν εκτελείται μιά από τις παρακάτω εντολές:

GOTO: Με την εντολή GOTO επεμβαίνουμε απευθείας στα ψηφία A0 ως A8 του καταχωρητή PC. Τα ψηφία A9 και A10 φορτώνονται με τα ψηφία 5 και 6 του καταχωρητή κατάστασης λειτουργίας (f3). Πριν εκτελέσουμε την εντολή GOTO στους μικροελεγκτές PIC 16C56 και PIC 16C57, πρέπει να φροντίσουμε, ώστε ο PC να "δείχνει" στη διεύθυνση που θέλουμε να προσπε-



Σχήμα 4. Η μνήμη δεδομένων χωρίζεται σε σελίδες. Γιά την επιλογή των θέσεων μνήμης χρησιμοποιείται ο καταχωρητής FSR.

λάσουμε (ψηφία 5 και 6 του f3 και A8...A0). CALL: Η εντολή κλήσης υποπρογράμματος διαφέρει από την εντολή διακλάδωσης GOTO, στο ότι το ψηφίο A8 έχει πάντα την τιμή "0". Αυτό σημαίνει, ότι με την εντολή CALL μπορούμε να προσπελάσουμε μιά περιορισμένη περιοχή της μνήμης προγράμματος. Το ίδιο ισχύει γιά τις εντολές, που αλλάζουν το περιεχόμενο του καταχωρητή PC. Το ψηφίο A8 έχει την τιμή "0", ενώ τα A9 και A10 φορτώνονται από τον καταχωρητή κατάστασης λειτουργίας.

Καταχωρητής κατάστασης λειτουργίας f3.

Η τιμή των ψηφίων 0 ως 2 (C, DC και Z) του καταχωρητής κατάστασης λειτουργίας **(Status Word Register)** εξαρτάται από το αποτέλεσμα των αριθμητικών / λογικών πράξεων. Οι πράξεις αυτές εκτελούνται από τη βαθμίδα ALU. Τα ψηφία 3 (PD-Flag) και 4 (TO-Flag) δείχνουν την αιτία που οδήγησε στον (τελευταίο) μηδενισμό του μικροελεγκτή. Με τη βοήθεια των ψηφίων 5 (PA0) και 6 (PA1) επιλέγουμε την επιθυμητή σελίδα μνήμης. Η τιμή τους αντιγράφεται στα ψηφία A9 και A10, κατά την εκτέλεση των εντολών που χρησιμοποιούν τον καταχωρητή PC (CALL, GOTO). Οταν η μνήμη RAM του μικροελεγκτή δεν επαρκεί, μπορούμε να χρησιμοποιήσουμε τον καταχωρητή f3 γιά να γράψουμε δεδομένα. Σ' αυτήν την περίπτωση, παρατηρούμε ότι τα ψηφία 3 και 4 δεν μπορούν να μεταβληθούν, ενώ η τιμή των ψηφίων 0...2 εξαρτάται από το αποτέλεσμα των αριθμητικών πράξεων. Γιαυτό, σας συμβουλεύουμε να χρησιμοποιείτε τον καταχωρητή f3 μαζί με εντολές μεταφοράς δεδομένων, που δεν αλλάζουν την τιμή των ψηφίων 0 ως 2 (πχ. BCF, BSF και MOVWF).

Καταχωρητής f4 FSR επιλογής αρχείου καταχωρητών.

Πίνακας 6. Η κατάσταση των TO και PD μετά από διάφορα συμβάντα:

Συμβάν	TO	PD
Τροφοδοσία	1	1
Λήξη WDT	0	x
Εντολή SLEEP	1	0
Εντολή CLRWDT	1	1

x = αδιάφορη τιμή
WDT = χρονιστής επιτήρησης

Πίνακας 7. Η κατάσταση των TO και PD μετά από το μηδενισμό του PIC. Οι τιμές παραμένουν μέχρι να εμφανιστεί ένα συμβάν σύμφωνα με τον πίνακα 6.

TO	PD	Ο μηδενισμός ενεργοποιήθηκε από:
0	0	Τερματισμός κατάστασης SLEEP από τον WDT
0	1	Λήξη χρόνου του WDT (όχι κατά την κατάσταση SLEEP)
1	0	Τερματισμός κατάστασης SLEEP από εξωτερικό RESET
1	1	Εφαρμογή τροφοδοσίας στο ολοκληρωμένο
X	X	Ανυπαρξία παλμών στον ακροδέκτη εξωτερ. μηδενισμού.

Η τιμή των ψηφίων παραμένει όπως έχει.

Η λειτουργία του καταχωρητή FSR (File Select Register) διαφέρει ανάμεσα στον PIC-16C57 και στους υπόλοιπους μικροελεγκτές της οικογένειας PIC-16C5x. Γιά τους PIC-16C54/C55/C56 ισχύουν τα εξής: τα ψηφία 0...4 καθορίζουν τη διεύθυνση του καταχωρητή που θέλουμε να προσπελάσουμε έμμεσα, μέσω του f0. Τα ψηφία 5...7 έχουν πάντα τιμή "1". Αν επιθυμούμε, μπορούμε να χρησιμοποιήσουμε τον καταχωρητή γιά αποθήκευση δεδομένων (5 ψηφία). Στην περίπτωση του PIC 16C567, τα ψηφία 5 και 6 επιλέγουν την ομάδα καταχωρητών (Register Bank), από τις τέσσερις ομάδες των καταχωρητών γενικών λειτουργιών. Οι καταχωρητές γενικών λειτουργιών μπορούν να προσπελαθούν άμεσα και έμμεσα. Οι διευθύνσεις 00H ως 0FH "δείχνουν" πάντα στους ίδιους καταχωρητές, ανεξάρτητα από την ομάδα καταχωρητών που έχουμε επιλέξει. Αλλάζοντας ομάδα καταχωρητών, ουσιαστικά αλλάζουν οι καταχωρητές 10H ως 1FH. Το έβδομο ψηφίο του καταχωρητή FSR είναι πάντα "1".

Βασικά κυκλώματα

Προγραμματισμός του PIC

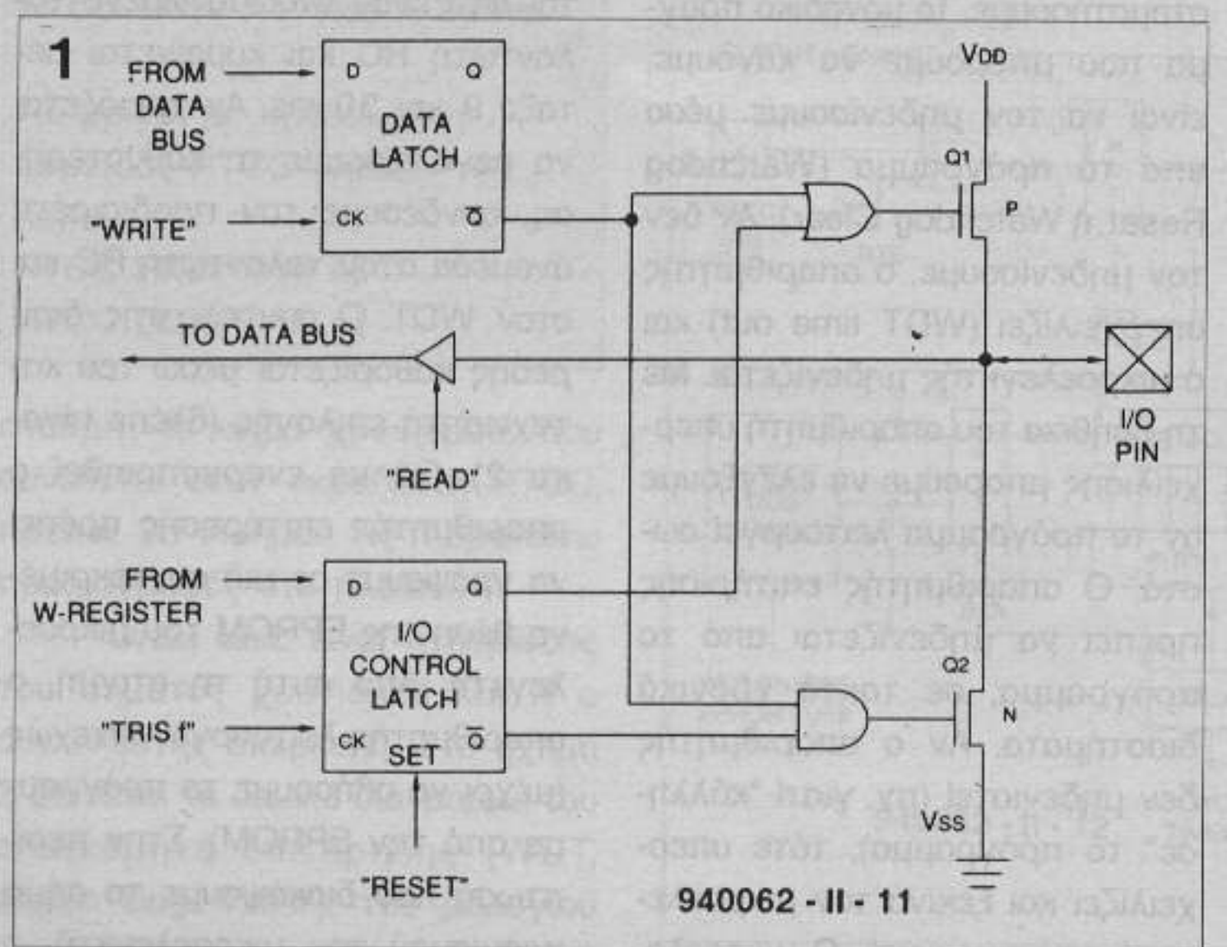
Μέρος 2. Οι κυριότεροι καταχωρητές

Καταχωρητής σωρού (Stack)

Οι μικροελεγκτές της σειράς PIC 16C5x διαθέτουν δυό καταχωρητές σωρού. Η διαχείριση του σωρού γίνεται απευθείας από το ηλεκτρονικό κύκλωμα του μικροελεγκτή. Η εντολή κλήσης υποπρογράμματος CALL αποθηκεύει στον πρώτο καταχωρητή σωρού την τιμή του απαριθμητή εντολών, αυξημένη κατά 1. Ταυτόχρονα, το αρχικό περιεχόμενο του πρώτου καταχωρητή σωρού αποθηκεύεται στο δεύτερο καταχωρητή σωρού. Οι πληροφορίες που υπήρχαν στον δεύτερο καταχωρητή σωρού χάνονται. Από αυτό φαίνεται, ότι δεν μπορούμε να καλούμε ταυτόχρονα περισσότερα από δυό υποπρογράμματα. Με την κλήση του τρίτου υποπρογράμματος, χάνεται η διεύθυνση επιστροφής του πρώτου. Η εντολή RETLW φορτώνει το περιεχόμενο του πρώτου καταχωρητή σωρού στον απαριθμητή εντολών PC. Το περιεχόμενο του δεύτερου καταχωρητή σωρού φορτώνεται στον πρώτο. Ο δεύτερος καταχωρητής σωρού παραμένει όπως έχει.

Καταχωρητές εισόδου/εξόδου (I/O)

Οι καταχωρητές I/O (θύρες) συνδέουν το ηλεκτρονικό κύκλωμα (ακροδέκτες εισόδου / εξόδου) με το λογισμικό. Οι θύρες I/O μπορούν να εγγραφούν και να αναγνωστούν όπως οι υπόλοιποι καταχωρητές. Με την εντολή ανάγνωσης διαβάζουμε τη λογική στάθμη που εφαρμόζεται στους ακροδέκτες. Κάθε θύρα του ολοκληρωμένου αντιστοιχεί σε ένα ψηφίο στους καταχωρητές I/O. Αυτό μπορεί να γίνει τόσο στις θύρες που είναι προγραμματισμένες σαν είσοδοι, όσο και στις θύρες που λειτουργούν σαν έξοδοι.



Σχήμα 1. Η εσωτερική συνδεσμολογία ενός ακροδέκτη I/O

Μετά το μηδενισμό του μικροελεγκτή, οι θύρες I/O συμπεριφέρονται σαν είσοδοι (=υψηλή αντίσταση). Γιά να τις προγραμματίσουμε σαν εξόδους, χρησιμοποιούμε την εντολή TRIS. Οταν το ψηφίο ενός καταχωρητή TRIS είναι "0", ο αντίστοιχος ακροδέκτης του μικροελεγτκή λειτουργεί σαν έξοδος. Μόλις προγραμματιστεί μιά θύρα σαν έξοδος, εμφανίζεται σ' αυτή η λογική τάση που καθορίζεται από το αντίστοιχο ψηφίο του καταχωρητή I/O. Γιαυτό, σας προτείνουμε, πριν μετατρέψετε μιά είσοδο σε έξοδο, να καθορίσετε από πριν τη λογική στάθμη εξόδου από τον καταχωρητή I/O. Ο προκαθορισμός της τάσης εξόδου διευκολύνει τη σύνδεση του μικροελεγκτή σε εξωτερικές συσκευές. Στο σχήμα 1 φαίνεται το δομικό διάγραμμα μιάς θύρας I/O. Ο καταχωρητής f5 αντιστοιχεί στη θύρα A. Από τον καταχωρητή χρησιμοποιούνται μόνο τα 4 λιγότερο σημαντικά ψηφία (RA0...RA3). Τα ψηφία 4 ως 7 δεν υποστηρίζονται ακόμα από τον μικροελεγκτή. Οταν διαβάζουμε τον καταχωρητή, τα ψηφία 4 ως 7 έχουν την τιμή "0". Γιά να είναι τα προγράμματα που γράφετε συμβατά με τους μελλοντικούς μικροελεγκτές της οικογένειας PIC, η τιμή των παραπάνω ψηφίων πρέπει να θεωρηθεί τυχαία. Ο καταχωρητής f6 αντιστοιχεί στη θύρα B και ο f7 στη θύρα C. Οι μικροελεγκτές PIC 16C54 και PIC 16C56 δεν διαθέτουν θύρα C. Σ' αυτήν την περίπτωση, ο καταχωρητής f7 μπορεί να χρησιμοποιηθεί σαν κοινός καταχωρητής δεδομένων.

Πίνακας 1. Αντιστοιχία διευθύνσεων και καταχωρητών

PIC16C54/C55/C56

$f08_H$ - f1FH : Καταχωρητές γενικών καθηκόντων

PIC16C57

f08H - f0FH : Καταχωρητές γενικών καθηκόντων

f10H - f1FH : Καταχωρητές γενικών καθ. της ομάδας μνήμης 0

f20H - f2FH : Ιδιοι με τους καταχωρητές f00H - f0FH

f30H - f3FH : Καταχωρητές γενικών καθ. της ομάδας μνήμης 1

f40H - f4FH : Ιδιοι με τους καταχωρητές f00H - f0FH

f50H - f5FH : Καταχωρητές γενικών καθ. της ομάδας μνήμης 2

f60H - f6FH : Ιδιοι με τους καταχωρητές f00H - f0FH

f70H - f7FH : Καταχωρητές γενικών καθ. της ομάδας μνήμης 3

Καταχωρητής	FSR		Διευθύνσεις
	ψηφίο 6	ψηφίο 5	
f10H - f1FH	0	0	10H - 1FH
f30H - f3FH	0	1	10H - 1FH
f50H - f5FH	1	0	10H - 1FH
f70H - f7FH	1	1	10H - 1FH

Καταχωρητές γενικών καθηκόντων

Οι διευθύνσεις των καταχωρητών γενικών καθηκόντων (λειτουργιών) φαίνονται στον πίνακα 1. Η επιλογή της επιθυμητής ομάδας καταχωρητών γίνεται με τη βοήθεια των ψηφίων 5 και 6 του καταχωρητή FSR. Η διεύθυνση προσπέλασης, που ακολουθεί μιά εντολή ανάγνωσης ή εγγραφής, έχει μήκος 5 ψηφίων. Με πέντε ψηφία μπορούμε να προσπελάσουμε μέχρι 1FH θέσεις μνήμης. Γιά να απευθυνθούμε σε μεγαλύτερη περιοχή της μνήμης μεταχειριζόμαστε τα ψηφία 5 και 6 του FSR. Ο τρόπος σχηματισμού της διεύθυνσης, φαίνεται στον πίνακα 1. Αν, γιά παράδειγμα, θέλουμε να απευθυνθούμε στον καταχωρητή $f5D_H$, το ψηφίο 6 του FSR πρέπει να έχει την τιμή "1", το 5 την τιμή "0" και η διεύθυνση που συνοδεύει την εντολή, να είναι $1D_H$.

Καταχωρητές ειδικών λειτουργιών

Καταχωρητής εργασίας W

Ο καταχωρητής εργασίας μπορεί να συγκριθεί με το συσσωρευτή, που υπάρχει σε πολλούς μικροελεγκτές και μικροεπεξεργαστές. Υπάρχει όμως μιά διαφορά: ο καταχωρητής W δεν αποτελεί το μοναδικό χώρο αποθήκευσης των αποτελεσμάτων της βαθμίδας ALU, όπως συμβαίνει σε πολλούς άλλους μικροελεγτές.

Καταχωρητής TRIS

Ο καταχωρητής TRIS καθορίζει τη συμπεριφορά των ακροδεκτών εισόδου / εξόδου. Σε κάθε ακροδέκτη αντιστοιχεί ένα

ψηφίο, σ' ένα καταχωρητή TRIS. Αν το ψηφίο έχει τιμή "1", η αντίστοιχη θύρα λειτουργεί σαν είσοδος. Αν το ψηφίο είναι "0", η θύρα συμπεριφέρεται σαν έξοδος. Η επιθυμητή τιμή γράφεται στους καταχωρητές με την εντολή TRIS. Οταν μηδενιστεί ο μικροελεγκτής, τα ψηφία των καταχωρητών TRIS λαμβάνουν την τιμή 1. Μ' αυτόν τον τρόπο, οι θύρες συμπεριφέρονται σαν είσοδοι.

Καταχωρητής επιλογής (Option Register)

Με τη βοήθεια του καταχωρητή επιλογής μπορούμε να καθορίσουμε, αν ο προδιαιρέτης θα τοποθετηθεί στην είσοδο του απαριθμητή επιτήρησης (WDT), ή στην είσοδο του ρολογιού / απαριθμητή (RTCC). Με τον ίδιο καταχωρητή μπορούμε να επιλέξουμε το συντελεστή διαίρεσης, την πηγή του σήματος χρονισμού του RTCC και το ενεργό μέτωπο του παλμού. Ο καταχωρητής επιλογής μπορεί να προσπελαθεί μόνο με την εντολή OPTION. Το μέγεθος του καταχωρητή είναι 6 ψηφία. Η σημασία κάθε ψηφίου φαίνεται στον πίνακα 2. Μετά το μηδενισμό του μικροελεγκτή, ολα τα ψηφία έχουν την τιμή "1".

WDT,RTCCκαι προδιαιρέτης

Απαριθμητής επιτήρησης WDT

Από τη στιγμή που θα ενεργοποιηθεί ο απαριθμητής επιτήρησης, είναι αδύνατο να τον σταματήσουμε. Το μοναδικό πράγμα που μπορούμε να κάνουμε, είναι να τον μηδενίσουμε μέσα από το πρόγραμμα (Watchdog Reset ή Watchdog Clear). Αν δεν τον μηδενίσουμε, ο απαριθμητής υπερχειλίζει (WDT time out) και ο μικροελεγκτής μηδενίζεται. Με τη βοήθεια του απαριθμητή υπερχείλισης μπορούμε να ελέγξουμε αν το πρόγραμμα λειτουργεί σωστά. Ο απαριθμητής επιτήρισης πρέπει να μηδενίζεται από το πρόγραμμα, σε τακτά χρονικά διαστήματα. Αν ο απαριθμητής δεν μηδενιστεί (πχ. γιατί "κόλλησε" το πρόγραμμα), τότε υπερχειλίζει και ξεκινά τον μικροελεγκτή από την αρχή. Ο μικροελεγκτής ανακτά τις αρχικές παραμέτρους λειτουργίας και το πρόγραμμα εκτελείται από την αρχή. Αυτή η μέθοδος χρησιμοποιείται σε εφαρμογές, όπου μιά τυχόν δυσλειτουργία του προγράμματος θα προκαλούσε ανεπιθύμητες (ως επικίνδυνες) καταστάσεις. Ο μηδενισμός του απαριθμητή επιτήρησης πρέπει να εκτελείται σε διάφορα σημεία του κυρίως προγράμματος. Η εντολή μηδενισμού δεν πρέπει να τοποθετηθεί ποτέ μέσα σ' ένα υποπρόγραμμα εξυπηρέτησης διακοπής. Η διακοπή μπορεί να εκτελείται κανονικά, αλλά το κυρίως πρόγραμμα να έχει "κολλήσει". Ο χρόνος που απαιτείται, μέχρι να υπερχειλίσει ο απαριθμητής επιτήρησης, εξαρτάται από τον τύπο του μικροελεγκτή PIC. Ο χρόνος καθορίζεται από έναν ενσωματωμένο ταλαντωτή RC και κυμαίνεται μεταξύ 9 και 30 ms. Αν χρειάζεται να μεγαλώσουμε τη καθυστέρηση, συνδέουμε τον προδιαιρέτη ανάμεσα στον ταλαντωτή RC και στον WDT. Ο συντελεστής διαίρεσης καθορίζεται μέσω του καταχωρητή επιλογής (βλέπε πίνακα 2). Γιά να ενεργοποιηθεί ο απαριθμητής επιτήρησης πρέπει να γράψουμε σε μιά συγκεκριμένη θέση της EPROM του μικροελεγκτή. Από αυτή τη στιγμή, ο απαριθμητής λειτουργεί συνεχώς (μέχρι να σβήσουμε το πρόγραμμα από την EPROM). Στην περίπτωση που διακόψουμε το σήμα χρονισμού του μικροελεγκτή, ο WDT θα συνεχίσει να μετρά, λόγω του ενσωματωμένου ταλαντωτή RC. Ο WDT και ο τυχόν προδιαιρέτης μηδενίζονται με τις εντολές CLRWDT και SLEEP.

Ο προδιαιρέτης

Ο μικροελεγκτής PIC διαθέτει ενσωματωμένο έναν προδιαιρέτη 8 ψηφίων. Με τη βοήθεια του ψηφίου PSA στον καταχωρητή επιλογής, μπορούμε να συνδέσουμε τον προδιαιρέτη στον απαριθμητή επιτήρησης ή στο RTCC. Ο συντελεστής διαίρεσης καθορίζεται με τα ψηφία PS0...PS2 του καταχωρητή επιλογής. Οταν ο προδιαιρέτης είναι συνδεδεμένος στον RTCC, μηδενίζεται κάθε φορά που γράφουμε στον καταχωρητή RTCC. Αντίθετα, αν συνδεθεί στον απαριθμητή επιτήρησης, μηδενίζεται με τις εντολές CLRWDT και SLEEP. Το σημείο που θα συνδεθεί ο προδιαιρέτης καθορίζεται μέσω του λογισμικού και μπορεί να αλλάξει ανά πάσα στιγμή. Γιά να εξασφαλιστεί ότι, κατά την ανάθεση του προδιαιρέτη στον WDT ή στον RTCC, δεν θα μηδενιστεί τυχαία ο μικροελεγκτής, η εταιρία Microchip προτείνει την παρακάτω διαδικασία:

Πίνακας 2. Η σημασία των bit στον καταχωρητή επιλογής

bit 5	bit 4	bit 3	bit 2	bit 1	bit 0
RTS	RTE	PSA	PS2	PS1	PS0

Συντελεστής διαίρεσης

PS2	PS1	PS0	RTCC	WDT
0	0	0	1:2	1:1
0	0	1	1:4	1:2
0	1	0	1:8	1:4
0	1	1	1:16	1:8
1	0	0	1:32	1:16
1	0	1	1:64	1:32
1	1	0	1:128	1:64
1	1	1	1:256	1:128

- PSA Ανάθεση του προδιαιρέτη
0 : στον RTCC
1 : στον WDT
- RTE Ενεργό μέτωπο παλμού στον ακροδέκτη RTCC
0 : LOW (0) -> HIGH (1)
1 : HIGH (1) -> LOW (0)
- RTS Χρονισμός του RTCC από:
0 : "Χρονιστή εκτέλεσης εντολών" (=Fosc/4)
1 : Από παλμούς στον ακροδέκτη RTCC

Ανάθεση του προδιαιρέτη, από τον RTCC, στον WDT.

```
1. MOVLW xx0x0xxxb   ;επιλογή εσωτερικού χρονισμού και
                     ;νέου συντελεστή διαίρεσης
2. OPTION            Αν ο συντελ/τής
                     ;διαίρεσης είναι 000 ή 001, διαλέξτε
                     ;αρχικά κάποιο μεγαλύτερο.
3. CLRF  1           ;Μηδενισμός του RTCC και του
                     ;προδιαιρ.
4. MOVLW xxxx1xxxb   ;Ανάθεση του προδιαιρέτη στον WDT,
5. OPTION            ;χωρίς να αλλάξουμε το συντελεστή
                     ;διαίρεσης.
6. CLRWDT            ;Μηδενισμός του WDT και του προδιαιρ.
7. MOVLW xxxx1xxxb   ;Επιλογή του συντελεστή διαίρεσης
8. OPTION
```

Ανάθεση του προδιαιρέτη, από τον WDT, στο RTCC.

```
1. CLRWDT   ;Μηδενισμός του WDT και του προδιαιρ.
2. MOVLW    xxxx0xxxb ;Ανάθεση του προδιαιρέτη στον
            RTCC και
3. OPTION   ;ταυτόχρονο καθορισμό του συντελεστή
            ;διαίρεσης
```

Τα βήματα 1 και 2 εκτελούνται όταν ο χρονισμός του RTCC γίνεται από εξωτερική πηγή. Τα βήματα 7 και 8 χρειάζονται μόνο, όταν ο νέος συντελεστής διαρέσης είναι 000 ή 001.

Ο RTCC

Ο RTCC του μικροελεγκτή PIC είναι ένας απαριθμητής 8 ψηφίων. Το περιεχόμενο του απαριθμητή μπορεί να αναγνωστεί και να εγγραφεί σα μιά κοινή θέση μνήμης. Αν ο απαριθμητής μετρήσει ως την τιμή $0FF_H$, επιστρέφει ξανά στο 00_H. Ο χρονισμός του RTCC γίνεται με δύο τρόπους: εσωτερικά, μέσω του σήματος χρονισμού του μικροελεγκτή (Fosc/4), ή εξωτερικά, από τον ακροδέκτη RTCC. Αν επιλέξουμε την εξωτερική πηγή χρονισμού, μπορούμε να καθορίσουμε το (ενεργό) μέτωπο του παλμού, με το οποίο θα σκανδαλίζεται ο απαριθμητής. Ο προδιαιρέτης μπορεί να συνδεθεί μεταξύ της πηγής χρονισμού και του απαριθμητή, επιτρέποντας τη μέτρηση περισσότερων παλμών. Οπως αναφέραμε προηγουμένως, έχουμε τη δυνατότητα να συνδέσουμε τον προδιαιρέτη στον WDT ή στον RTCC, κατά τη διάρκεια εκτέλεσης του προγράμματος. Οταν χρησιμοποιείτε τον RTCC, πρέπει να γνωρίζετε ότι οι παλμοί χρονισμού καθυστερούν δυό κύκλους μηχανής, εξαιτίας της βαθμίδας συγχρονισμού. Αυτό σημαίνει, ότι αν, γιά παράδειγμα, γράψουμε στον απαριθμητή RTCC, οι καινούργιοι παλμοί που θα μετρήσει η βαθμίδα, θα εμφανιστούν στο πρόγραμμα μετά από δυό κύκλους μηχανής. Οταν χρησιμοποιούμε τον προδιαιρέτη, ο συχρονισμός γίνεται στην έξοδό του. Οταν ο απαριθμητής χρονίζεται εσωτερικά, δεν επηρεάζεται από το σήμα που συνδέεται στον ακροδέκτη RTCC. Γιά να λειτουργεί σωστά ο μικροελεγκτής, ο ακροδέκτης RTCC πρέπει να είναι πάντα συνδεδεμένος σε κάποια λογική

Γιά χρήση χωρίς προδιαιρέτη:
RTCC HIGH $\geq$ 2 tosc + 20 ns
RTCC LOW $\geq$ 2 tosc + 20 ns

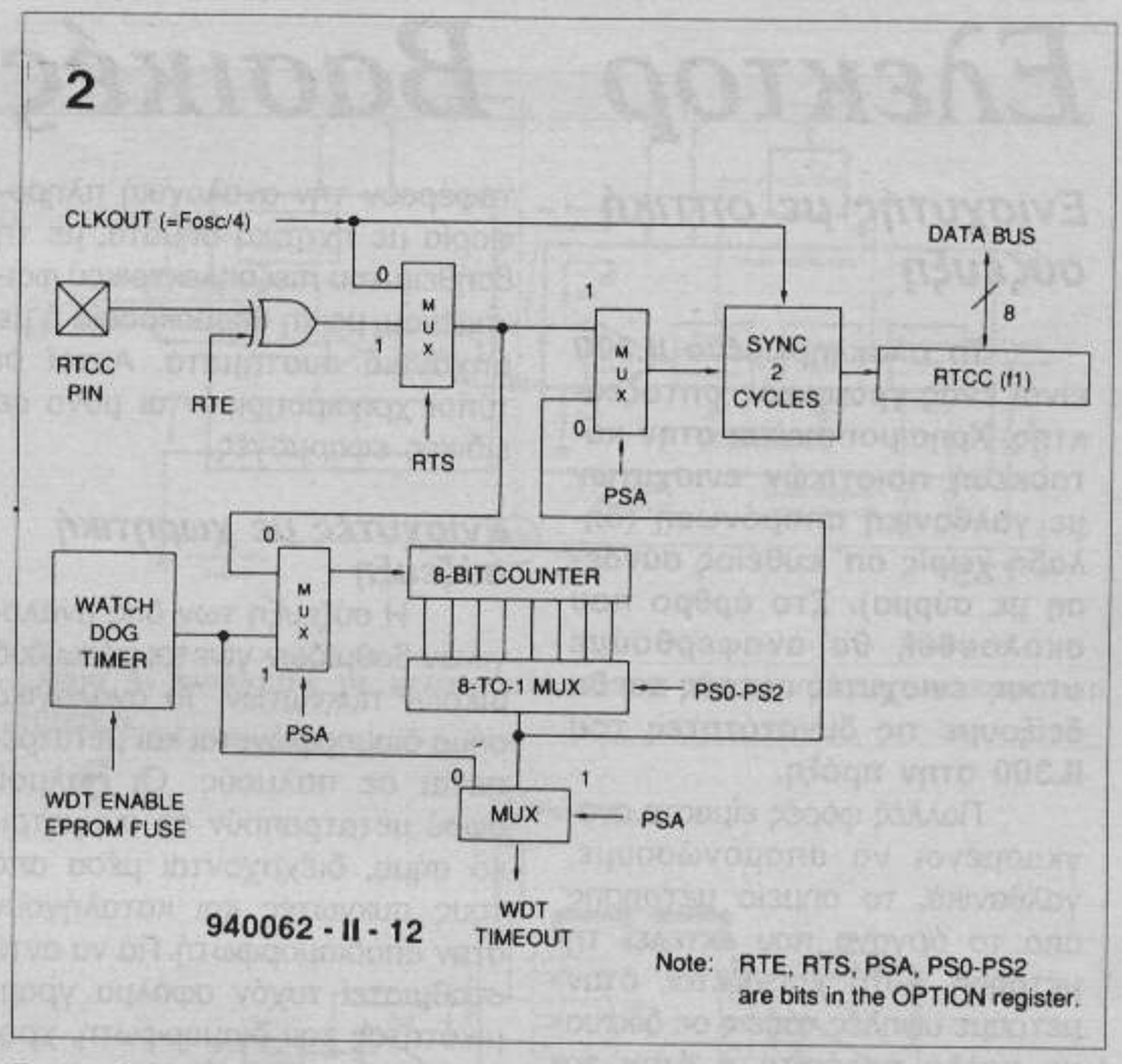
Γιά χρήση με προδιαιρέτη:
Περίοδος RTCC (4tosc + 40 ns)/N
RTCC HIGH $\geq$ 10 ns
RTCC LOW $\geq$ 10 ns

στάθμη. Το σήμα χρονισμού, που συνδέεται στον ακροδέκτη RTCC, πρέπει να πληρεί τις παραπάνω προδιαγραφές στο πλαίσιο.

Οπου tosc είναι η περίοδος του σήματος χρονισμού και N ο συντελεστής διαίρεσης. Στο σχήμα 2 φαίνεται το δομικό διάγραμμα του απαριθμητή επιτήρησης (WDT, Watch Dog Timer), του ρολογιού πραγματικού χρόνου / απαριθμητή (RTCC, Real Time Clock / Counter) και του προδιαιρέτη. Οι παραπάνω βαθμίδες ελέγχονται από τον καταχωρητή επιλογής.

Λανθάνουσα λειτουργία (SLEEP)

Η λειτουργία του μικροελεγκτή διακόπτεται μόλις εκτελεστεί η εντολή SLEEP. Η εντολή SLEEP μηδενίζει τον απαριθμητή επιτήρησης (στην περίπτωση που είναι ενεργοποιημένος), δίνει την τιμή 0 στο ψηφίο PD του καταχωρητή κατάστασης λειτουργίας f3, ενεργοποιεί το ψηφίο TO και σταματά τον ταλαντωτή χρονισμού. Γιά να μειώσετε την κατανάλωση του ολοκληρωμένου, όλα τα σήματα που συνδέονται στις εισόδους (συμπεριλαμβανομένης και της εισόδου RTCC), πρέπει να έχουν σταθερή λογική στάθμη. Μόλις διακοπεί η λειτουργία του ταλαντωτή χρονισμού, ο RTCC σταματά να απαριθμεί παλμούς. Ο μικροελεγκτής επαναφέρεται στη ζωή μόλις υπερχειλίσει ο απαριθμητής επιτήρησης ή αν εφαρμοστεί στιγμιαία λογικό "0" στη είσοδο $\overline{\text{MCLR}}$. Τότε, ο PIC ξεκινά τη διαδικασία μηδενισμού. Μόλις δεχθεί το σήμα μηδενισμού, περιμένει λίγο χρόνο και κατόπιν εκτελεί το πρόγραμμα από την αρχή. Με τη βοήθεια του ψηφίου PD, στον καταχωρητή κατάστασης λειτουργίας, μπορούμε να διαπιστώσουμε την αιτία που προκάλεσε το μηδενισμό του μικροελεγκτή. Ετσι, μπορούμε να



Σχήμα 2. Η εσωτερική συνδεσμολογία του WDT, RTCC και του προδιαιρέτη.

διακρίνουμε αν ο μηδενισμός προκλήθηκε εξαιτίας της εφαρμογής της τροφοδοσίας, ή από κάποιο άλλο λόγο. Το ψηφίο TO δείχνει αν η κατάσταση SLEEP αναιρέθηκε από τον απαριθμητή επιτήρησης ή από εξωτερικό σήμα που εφαρμόστηκε στον ακροδέκτη MCLR. Περισσότερες πληροφορίες μπορείτε να βρείτε στους πίνακες, που συνοδεύουν το πρώτο άρθρο της σειράς.

Ο μηδενισμός του PIC

Ο μικροελεγκτής ξεκινά τη διαδικασία μηδενισμού μόλις εφαρμόσουμε στιγμιαία έναν παλμό "0" στην είσοδο $\overline{\text{MCLR}}$, ή όταν υπερχειλίσει ο απαριθμητής επιτήρησης. Από τη στιγμή εφαρμογής του σήματος μηδενισμού, μέχρι να αρχίσει η εκτέλεση του προγράμματος από την αρχή, περνούν 9...30 ms. Ο χρόνος αυτός εξαρτάται από τον ενσωματωμένο απαριθμητή OST (Oscillator Startup Timer). Στην περίπτωση εξωτερικού σήματος μηδενισμού, ο χρόνος OST ξεκινά να μετρά από τη στιγμή που εμφανιστεί το ανερχόμενο μέτωπο του παλμού στην είσοδο $\overline{\text{MCLR}}$. Οσο χρόνο η τάση στην είσοδο $\overline{\text{MCLR}}$ είναι χαμηλή ("0"), ο μικροελεγκτής παραμένει στην κατάσταση μηδενισμού. Αν συνδέσουμε την είσοδο $\overline{\text{MCLR}}$ με την τάση τροφοδοσίας, ο χρόνος OST ξεκινά αμέσως μόλις τροφοδοτήσουμε το μικροελεγκτή. Η κατάσταση του μικροελεγκτή μετά τον μηδενισμό έχει ως εξής:

- Ο ταλαντωτής λειτουργεί, ή μόλις ξεκινά.
- Οι θύρες I/O συμπεριφέρονται σαν είσοδοι (οι καταχωρητές TRIS έχουν την τιμή 0FFh).
- Ο απαριθμητής προγράμματος δείχνει στη διεύθυνση:
  α) γιά τους PIC 16C54 και 16C55 : 01FFh
  β) PIC 16C56 : 03FFh
  γ) PIC 16C57 : 07FFh - Ο καταχωρητής επιλογής έχει την τιμή xx111111b.
- Ο απαριθμητής επιτήρησης και ο προδιαιρέτης έχουν μηδενιστεί.
- Τα τρία σημαντικότερα ψηφία του καταχωρητή κατάστασης λειτουργίας έχουν την τιμή 000h.
- Στους μικροελεγκτές με ταλαντωτή RC, το σήμα CLKOUT στην έξοδο OSC2, έχει χαμηλή λογική στάθμη.

Προγραμματισμός του PIC

Μέρος 3ο. Οι εντολές του μικροελεγκτή PIC 16C5x

Στα προηγούμενα άρθρα αναφερθήκαμε στην κατασκευαστική δομή του μικροελεγκτή PIC. Στη συνεχεια θα παρουσιάσουμε τις εντολές της οικογένειας PIC 16C5x.

Σε γενικές γραμμές, οι εντολές χωρίζονται σε δυό μέρη: στην καθαυτό εντολή (Operation Code), που δείχνει πιά πράξη πρόκειται να εκτελεστεί και στον τελεστή (συνήθως μιά θέση μνήμης ή ο κάποιος καταχωρητής) που λαμβάνει μέρος στην πράξη. Στους περισσότερους μικροελεγκτές και μικροεπεξεργαστές καθένα από τα μέρη της εντολής αποτελείται από ένα ή περισσότερα Byte. Αντίθετα, στους μικροελεγκτές της σειράς PIC 16C5x, η εντολή και ο τελεστής αποτελούνται από 12 ψηφία (bit). Αυτή η σχεδίαση έχει πλεονεκτήματα αλλά και μειονεκτήματα. Αρχικά, περιορίζεται το μέγιστο πλήθος των εντολών (στην περίπτωσή μας σε 33). Από την άλλη πλευρά, επιταχύνεται η επεξαργασία των εντολών, καθώς κάθε εντολή μπορεί να αναγνωστεί από τον μικροελεγκτή μέσα σ' ένα κύκλο μηχανής. Οι εντολές του μικροελεγκτή PIC 16C5x χωρίζονται σε τρείς κατηγορίες:

- εντολές προσπέλασης αρχείου καταχωρητών κατά Byte
- εντολές προσπέλασης αρχείου καταχωρητών κατά Bit
- εντολές ελέγχου και ανάθεσης αρχικών τιμών

Οι εντολές προσπέλασης κατά Byte διαφέρουν, σε σχέση με άλλους μικροελεγκτές. Συνήθως, το αποτέλεσμα μιάς πράξης, γιά παράδειγμα της πρόσθεσης δυό αριθμών, αποθηκεύεται στο συσσωρευτή (καταχωρητής εργασίας W στην περίπτωση του PIC). Αντίθετα, στους μικροελεγκτές 16C5x, ο προγραμματιστής μπορεί να επιλέξει σε ποιόν καταχωρητή θα καταλήξει το αποτέλεσμα. Ο αποδέκτης μπορεί να είναι ο καταχωρητής εργασίας (W) ή ο καταχωρητής που έλαβε μέρος στην πράξη (f). Αυτό καθορίζεται από το ψηφίο κατεύθυνσης d. Η εταιρία Microchip σφήνει στον προγραμματιστή τη δυνατότητα να ορίσει την αντιστοιχία του ψηφίου d. Ετσι, στην αρχή του προγράμματος πρέπει να καθορίσετε την τιμή του d ως εξής:

```
; Definition of the Destination
; W equ 0H ; Destination = W
; F equ 1H ; Destination = f
```

Στον πίνακα 1 φαίνονται περιληπτικά οι εντολές του μικροελεγκτή, ενώ στον πίνακα 2 δίνονται οι εντολές του μεταφραστή (Assembler). Στη συνέχεια, θα παρουσιάσουμε αναλυτικά τις εντολές, με τη βοήθεια ενός παραδείγματος.

Παράδειγμα: αναλάμπον LED.

Το μέγιστο ρεύμα που μπορούν να παρέχουν οι έξοδοι του PIC 16C5x είναι 20 mA. Παράλληλα, πρέπει να εξασφαλίσουμε ότι το ρεύμα τροφοδοσίας δεν θα υπερβεί τα 50 mA και το ρεύμα ως προς τη γείωση τα 150 mA. Γιά τις ανάγκες του παραδείγματος (πρόγραμμα LED SMPL.ASM, κύκλωμα σχήματος 1) συνδέουμε το LED, μέσω μιάς αντίστασης 330 Ω, στη θύρα RA0. Το άλλο άκρο του LED οδηγείται στη θετική τάση τροφοδοσίας +5 V. Οι υπόλοιπες θύρες του μικροελεγκτή συνδέονται στην τάση +5 V, μέσω των αντιστάσεων πρόσδεσης 10 K (Pull Up). Το πλήκτρο συνδέεται ανάμεσα στη θύρα RA1 και στη γείωση. Το κύκλωμα συναρμολογείται στην πειραματική πλακέτα γιά τον μικροελεγκτή PIC, που δημοσιεύσαμε στο διπλό τεύχος του ΕΛΕΚΤΟΡ (7-8/94). Μόλις πιέσουμε το πλήκτρο S1, το LED αρχίζει να αναβοσβήνει. Αυτή η λειτουργία εκτελείται μέσα από το πρόγραμμα. Γιά να μεταφράσουμε το πρόγραμμα σε γλώσσα μηχανής πληκτρολογούμε στο PC την εντολή MPALC LED SMPL.ASM και πιέζουμε το πλήκτρο εισαγωγής. Στον προγραμματισμό πρέπει να καθορίσουμε ότι ο μικροελεγκτής PIC θα χρονίζεται από κρυσταλλικό ταλαντωτή (XT fuse). Γιά να κατανοήσουμε τη λειτουργία του προγράμματος θα περιγράψουμε κάθε γραμμή αναλυτικά. Οι εντολές των γραμμών 1 ως 18 απευθύνονται στο μεταφραστή. Αυτές καθορίζουν τις αρχικές συνθήκες λειτουργίας του μεταφραστή. Η σημασία κάθε εντολής επεξηγείται από τα σχόλια. Ακολουθεί ο καθορισμός της διεύθυνσης έναρξης του προγράμματος (Reset Vector). Η διεύθυνση αυτή διαφέρει από μικροελεγκτή σε μικροελεγκτή. Γιαυτό το σκοπό καθορίζουμε στη γραμμή 19 (σταθερά controler) τον τύπο του μικροελεγκτή (54 = PIC 16C54). Με τη βοήθεια της εντο-

Data Directives		
data	<expr>	Create a 12bit data value or character string
zero	<mem units>	Initialze with zero <mem units> of program space
set	<label>...<expr>	Define assembler variable
res	<mem units>,	Reserve words of program space
equ	<label>...<expr>	Define an assembler constant
include	"<file_name>"	include a file into the assembly source flow

940062-III-T4

Listing Directives		
list	<option>	Set various list control options
page		Force a page eject
title	"title text"	Define an new title for the listing header
subtitl	"subtitle text"	Define a new subtitle for the listing header

940062-III-T5

Control Directives		
if	<expr>	Start of a conditional assembly block
else		Start an alternate confitional assembly block
endif		Terminate a conditional assembly block
org	<label>...<addr>	Set absolute address for following code block
end		Terminate assembly code block

940062-III-T6

Macro Directives		
macro	<label>...[<arg>[,<arg>]...]	Begin a macro body definition
endm		Terminate macro body definition
local	<label>[,<label>]...	Define assembler labels as local to a macro
exitm		Exit macro

940062-III-T7

Byte - oriented file register operations			
Mnemonic		Operation	Status aff.
ADDWF	f,d	Add W and f	C, DC, Z
ANDWF	f,d	AND W and f	Z
CLRF	f	Clear f	Z
CLRW	-	Clear W	Z
COMF	f,d	Complement f	Z
DECF	f,d	Decrement f	Z
DECFSZ	f,d	Decrement f, Skip if zero	-
INCF	f,d	Increment f	Z
INCFSZ	f,d	Increment f, Skip if zero	-
IORWF	f,d	Inclusive OR W and f	Z
MOVF	f,d	Move f	Z
MOVWF	f	Wove W to f	-
NOP	-	No operation	-
RLF	f,d	Rotate left through Carry	C
RRF	f,d	Rotate right through Carray	C
SUBWF	f,d	Subtract W from f	C, DC, Z
SWAPF	f,d	Swap halves f	-
XORWF	f,d	Exclusive OR W and f	Z

940062-III-T1

Bit-oriented file register operations			
Mnemonic		Operation	Status aff.
BCF	f,b	Clear bit (b) in file (f)	-
BSF	f,b	Set bit (b) in file (f)	-
BTFSC	f,b	Test bit (b) in file (f): Skip if clear	-
BTFSS	f,b	Test bit (b) in file (f): Skip if set	-

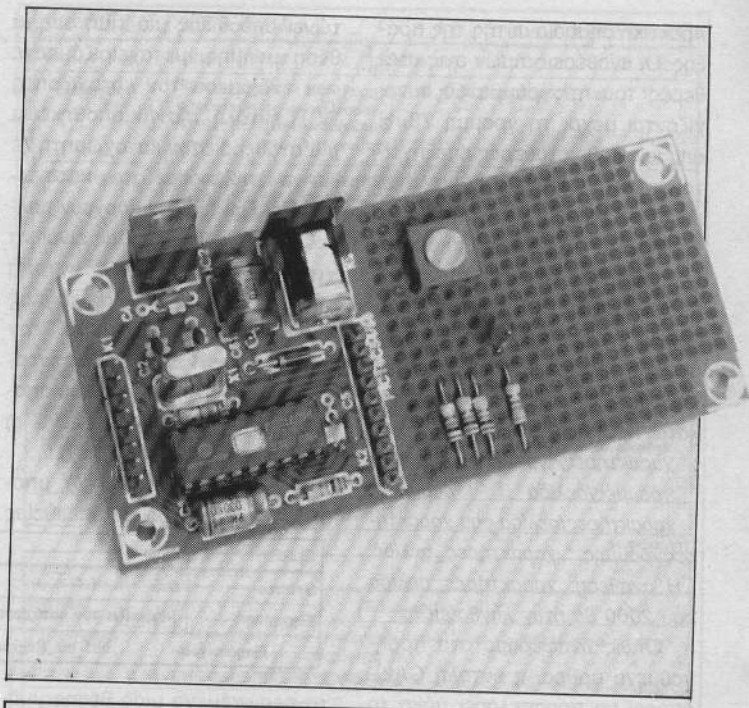
940062-III-T2

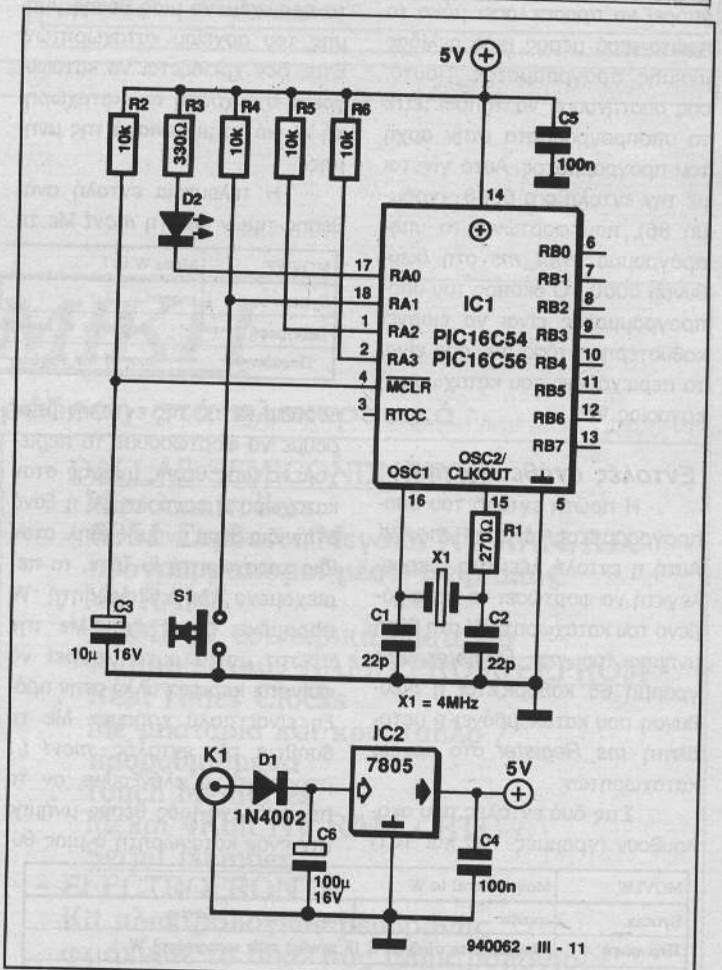
Literal and control operations			
Mnemonic		Operation	Status aff.
ANDLW	k	AND literal (k) and W	Z
CALL	k	Call subroutine (see detail description)	-
CLRWDT	-	Clear Watchdogtimer	TO, PD
GOTO	k	G to address (see detail description)	-
IORLW	k	incl. OR literal (k) and W	Z
MOVLW	k	Move literal (k) to W	-
OPTION	-	Load OPTION-register with W	-
RETLW	k	Return, place literal (k) in W	-
SLEEP	-	Go into standby mode	TO, PD
TRIS	f	Load W in TRIS-register f	-
XORLW	k	Exclusive OR literal (k) and W	Z

940062-III-T3

```
        title    "Sample flashing LED"  ; Program title
        subtitl  "Declarations"         ; Program subtitle

        ; Listing Options for MPALC assembler
        list     c=136                  ; Columns per line
        list     n=80                   ; Lines per page
        list     r=HEX ; [HEX,DEC,OCT]  ; HEX as default
        list     l=ON  ; [ON / OFF]     ; Listing ON
        list     x=ON  ; [ON / OFF]     ; Macro expansion on
        list     t=ON  ; [ON / OFF]     ; Truncate listing lines
        list     e=0   ; [0,1,2,3]      ; 0 = Report all Messages
                                        ; 1 = Report Warnings, fatals, criticals
                                        ; 2 = Report fatals, criticals
                                        ; 3 = Report criticals

        page                            ; New Page

        ; select controller
controller      equ     56              ; 54: PIC 16C54, 55: PIC 16C55 ...

        if controller == 54             ; PIC 16C54
            list    p=16C54             ;
            Adr_Reset equ 01FFH         ;
            endif
        if controller == 55             ; PIC 16C55
            list    p=16C55             ;
            Adr_Reset equ 01FFH         ;
            endif
        if controller == 56             ; PIC 16C56
            list    p=16C56             ;
            Adr_Reset equ 03FFH         ;
            endif
        if controller == 57             ; PIC 16C57
            list    p=16C57             ;
            Adr_Reset equ 07FFH         ;
            endif

                                                        940062-B-T43-1
        ; Main declarations

        ; Destination of Byte-Oriented File Register Operations
W       equ     00H                     ; Destination is W-Register
F       equ     01H                     ; Destination is F-Register

        ; Directions of I/O - Ports
Input   equ     01H                     ; Input
Output  equ     00H                     ; Output

; ------------------------------------------------------------------------------
; Register file declarations

        ; Operational Register File
INDIRECT        equ     00H             ; Indirect Data addressing
RTCC            equ     01H             ; Real Time Clock/Counter Register
PC              equ     02H             ; Program counter
STATUS          equ     03H             ; Status Word Register
FSR             equ     04H             ; File Select Register

        ; I/O Registers (Ports)
Port_A          equ     05H             ;
Port_B          equ     06H             ;
Port_C          equ     07H             ; PIC 16C55/C57 only

        ; General Purpose Registers, user defined
us_Register     equ     009H            ;
ms_Register     equ     00AH            ;

; ------------------------------------------------------------------------------
; Hardware declarations

LED_B           equ     00H             ; LED connected to RA0
LED_P           equ     Port_A          ;
LED_off_time    equ     000H            ; LED off for 256 ms
LED_on_time     equ     000H            ; LED on for 256 ms

                                                        940062-B-T43-2
KEY_B           equ     01H             ; KEY connected to RA1
KEY_P           equ     Port_A          ;


        subtitl "Subroutines"
        page
; ==============================================================================
; Subroutines
; ==============================================================================

        org     0000H                   ; Subroutines only in first half of
                                        ; page
; ------------------------------------------------------------------------------
; Subroutine Wait_ms
;
; Parameter:    ms in W-Register
; Return value:  000H
;
; Delay time is 1 ms * (W Register) at Fosc = 4Mhz
; Note: W Register = 000H <=> 256 ms
;

Wait_ms   movwf ms_Register             ; store ms
          movlw 0FEH                    ; correct call time (incl one movlw
          movwf us_Register             ; before calling) for first loop

W_Loop_ms movlw 0F9H                    ; One loop 249 * W_Loop_us + 4 cycles
          addwf us_Register,F           ;

W_Loop_us nop                           ; one loop = 4 cycles
          decfsz us_Register,F          ; us_register - 1, skip if zero
            goto W_Loop_us              ;

          decfsz ms_Register,F          ; ms_register - 1, skip if zero
            goto W_Loop_ms              ;

          nop                           ; correct time of last ms loop

          retlw 000H                    ;
;
; ------------------------------------------------------------------------------

        subtitl "Main program"
        page
; ==============================================================================
; Main Program
; ==============================================================================
;
Main_Start                              ; Starting after RESET
          movlw 00EH                    ; Set RA0, which is connected to LED,
          tris  Port_A                  ; as output
          goto  LED_off                 ; Initially: LED off

Loop      btfsc KEY_P, KEY_B            ; Key pressed?  (RA1 = 0?)
            goto Loop                   ; No, wait until key pressed

          bcf   LED_P, LED_B            ; LED on
          movlw LED_on_time             ; wait
          call  Wait_ms                 ;

LED_off   bsf   LED_P, LED_B            ; LED off
          movlw LED_off_time            ; wait
          call  Wait_ms                 ;

          goto  Loop
;
; ------------------------------------------------------------------------------
          org   Adr_Reset               ; define reset vector
Reset     goto  Main_Start              ; Start at Main_Start after RESET
; ==============================================================================
          end
```

940062-B-T43-4





λής ελέγχου *if...endif* αναθέτουμε στη σταθερά *Adr_Reset* την πρώτη διεύθυνση της μνήμης, που θα εκτελέσει ο μικροελεγκτής μετά από το μηδενισμό. Αυτή η διεύθυνση χρησιμοποιείται από το μεταφραστή (γραμμή 144) γιά να υπολογίσει τη θέση του προγράμματος στη μνήμη. Η ανάθεση του τύπου του μικροελεγκτή στη σταθερά p (List p=16Cxx) χρειάζεται γιά τη λειτουργία του μεταφραστή. Οι υπεύθυνοι της εταιρίας δεν γνώριζαν ποιά είναι η

πρακτική σημασία αυτής της πράξης. Οι αναθέσεις τιμών στις σταθερές του προγράμματος συνεχίζονται μέχρι τη γραμμή 78. Η σημασία κάθε σταθεράς επεξηγείται από τα σχόλια. Γιά να είναι εύκολα κατανόητο το πρόγραμμα, πρέπει να δώσετε ιδιαίτερη σημασία στην ανάθεση των σταθερών τιμών. Οι σχέσεις που έχουν οι τιμές μεταξύ τους πρέπει να φαίνονται εύκολα. Ενα καλό παράδειγμα φαίνεται στο παρακάτω πρόγραμμα:

χαρακτήρες/ γραμμής equ 80
γραμμές/σελίδα equ 25
*χαρακτήρες/σελίδα equ χαρακτήρεςγραμμής * χαρακτήρες_σελίδα*
Η ανάθεση: χαρακτήρες_σελίδα equ 2000 θα ήταν λανθασμένη.

Οπως αναφέραμε στα προηγούμενα άρθρα, η εντολή CALL μπορεί να προσπελάσει μόνο το πρώτο μισό μέρος μιάς σελίδας μνήμης προγράμματος. Γιαυτό, σας συστήνουμε να τοποθετείτε τα υποπρογράμματα στην αρχή του προγράμματος. Αυτό γίνεται με την εντολή org 0000_H (γραμμή 86), που φορτώνει το υποπρόγραμμα *Wait_ms* στη διεύθυνση 0000_H. Ο σκοπός του υποπρογράμματος είναι να εισάγει καθυστέρηση τόσα ms, όσο είναι το περιεχόμενο του καταχωρητή εργασίας W.

Εντολές ανάθεσης τιμών

Η πρώτη εντολή του υποπρογράμματος είναι η *movwf*. Αυτή η εντολή λέει στο μικροελεγκτή να φορτώσει το περιεχόμενο του καταχωρητή W στη θέση μνήμης του *ms_Register*. Στη γραμμή 65 καθορίζεται η διεύθυνση που καταλαμβάνει η μεταβλητή *ms_Register* στο αρχείο καταχωρητών.

Στις δυό εντολές που ακολουθούν (γραμμές 100 και 101) αποθηκεύεται στη θέση μνήμης *us_Register* η τιμή $0FE_H$. Καθώς δεν υπάρχει εντολή που να φορτώνει απευθείας μιά τιμή σε μιά θέση μνήμης, μεταχειριζόμαστε σαν ενδιάμεσο τον καταχωρητή W. Η εντολή *movlw* αποθηκεύει μιά σταθερά στον καταχωρητή W. Γιά να αναθέσουμε στον καταχωρητή W την τιμή 0 υπάρχουν πολλοί τρόποι. Μιά μέθοδος είναι να χρησιμοποιήσουμε την εντολή movlw 000_H. Ενας άλλος τρόπος είναι να μεταχειριστούμε την εντολή *clrw*. Η εντολή *clrw* ενεργοποιεί το ψηφίο μηδενικού αποτελέσματος (Zero Flag), ενώ η *movlw* το αφήνει όπως έχει.

Με την εντολή *clrf* μπορούμε να μηδενίσουμε απευθείας το περιεχόμενο μιάς θέσης μνήμης του αρχείου καταχωρητών. Ετσι, δεν χρειάζεται να καταφύγουμε στη χρήση του καταχωρητή W γιά το μηδενισμό της μνήμης.

Η τελευταία εντολή ανάθεσης τιμών είναι η *movf*. Με τη βοήθεια αυτής της εντολής μπορούμε να φορτώσουμε το περιεχόμενο μιάς θέσης μνήμης στον καταχωρητή εργασίας W ή ξανά στην ίδια θέση μνήμης (δηλ. στον ίδιο καταχωρητή f). Τότε, το περιεχόμενο του καταχωρητή W παραμένει όπως έχει. Με την πρώτη ματιά αυτό μπορεί να φαίνεται περιττό, αλλά στην πράξη είναι πολύ χρήσιμο. Με τη βοήθεια της εντολής *movf f,1* μπορούμε να ελέγξουμε, αν το περιεχόμενο μιάς θέσης μνήμης (πχ ενός καταχωρητή ή μιάς θύρας I/O) έχει τιμή 0. Οπως ήδη προαναφέραμε, το υποπρόγραμμα περιμένει γιά τόσα ms, όσο είναι το περιεχόμενο του καταχωρητή W. Το υποπρόγραμμα έχει δυό βρόγχους, που ο ένας καλύπτει τον άλλο. Η εκτέλεση του εσωτερικού βρόγχου (γραμμές 103 ως 108) διαρκεί ακριβώς 1 ms (συχνότητα χρονισμού του μικροελεγκτή = 4 MHz). Ο βρόγχος αυτός καλείται τόσες φορές, όση είναι η τιμή του καταχωρητή W (γραμμές 110/111). Το πλήθος των επαναλήψεων του εσωτερικού βρόγχου δεν αντιγράφεται στον καταχωρητή us, αλλά προστίθεται στο περιεχόμενό του. Αυτό έχει σαν αποτέλεσμα να συνυπολογίζεται η χρονική καθυστέρηση κατά την κλήση του υποπρογράμματος (γραμμές 100/101), στο χρόνο εκτέλεσης του πρώτου εσωτερικού βρόγχου. Ο εσωτερικός βρόγχος τελειώνει όταν ο καταχωρητής us πάρει την τιμή 00_H.

MOVLW	Move Literal to W		
Syntax	movlw k	Statusbits	Κανένα
Περιγραφή	Φορτώνει τη σταθερά K (8 ψηφία) στον καταχωρητή W		
Παράδειγμα:	movlw 0FEH		

MOVF	Move f		
Syntax	movf f,d	Statusbits	Z
Περιγραφή	d = 0 : φορτώνει το περιεχόμενο του καταχωρητή W στον f d = 1 : φορτώνει το περιεχόμενο του καταχωρητή f ξανά στον f		
Παράδειγμα:	movf Port_B, W		

940062-III-T12

CLRW	Clear W Register		
Syntax	clrw	Statusbits	Z
Περιγραφή	Μηδενίζει τον καταχωρητή W και ενεργοποιεί το Flag Z		
Παράδειγμα:	clrw		

940062-III-T10

CLRF	Clear f		
Syntax	clrf f	Statusbits	Z
Περιγραφή	Μηδενίζει τον καταχωρητή f		
Παράδειγμα:	clrf ms_Register		

940062-III-T11

MOVWF	Move W to f		
Syntax	movwf f	Statusbits	Κανένα
Περιγραφή	Φορτώνει το περιεχόμενο του καταχωρητή W στον f		
Παράδειγμα:	movwf ms_Register		

940062-III-T8

Αριθμητικές εντολές

Οι μικροελεγκτές της σειράς PIC 16C5x αναγνωρίζουν τις μαθηματικές εντολές *ADDWF, SUBWF, INCF* και *DECF*. Ενα παράδειγμα της εντολής *ADDWF* φαίνεται στη γραμμή 104. Στη γραμμή 104 προστίθεται ο αριθμός $0F9_H$, που βρίσκεται αποθηκευμένος στον καταχωρητή W (γραμμή 103) στο περιεχόμενο του καταχωρητή us. Το αποτέλεσμα μιάς αριθμητικής εντολής μπορεί να αποθηκευτεί στον καταχωρητή W ή στον καταχωρητή f, που λαμβάνει μέρος στην πράξη. Μ' αυτό τον τρόπο μπορούμε να προσθέσουμε την ίδια τιμή σε πολλούς καταχωρητές *f*, χωρίς να επαναλαμβάνουμε τις εντολές ανάθεσης τιμών (όπως συμβαίνει σε άλλους μικροελεγκτές). Οι εντολές πρόσθεσης και αφαίρεσης δεν λαμβάνουν υπόψιν το υπόλοιπο της πράξης. Αν προσθέσουμε αριθμούς με περισσότερα από 8 ψηφία (bit), πρέπει να υπολογίσουμε ξεχωριστά το υπόλοιπο. Στο πρόγραμμα 2 φαίνεται το παράδειγμα πρόσθεσης δυό αριθμών, που αποτελούνται από 32 ψηφία ο καθένας. Οι αριθμοί είναι ο Z (Z HH, Z HL, Z LH, Z LL) και ο N (N HH, N HL, N LH, N LL). Η εντολή *SUBWF* εκτελείται σύμφωνα με τη μέθοδο του συμπληρώματος ως προς 2. Αυτό έχει σαν αποτέλεσμα να ενεργοποιείται το ψηφίο του υπολοίπου (Carry flag) όταν δεν υπάρχει υπόλοιπο. Οι εντολές *INCF* και *DECF* αυξάνουν ή μειώνουν κατά 1 το περιεχόμενο μιάς θέσης μνήμης τού αρχείου καταχωρητών. Το αποτέλεσμα εγγράφεται ξανά στο αρχείο καταχωρητών ή στον καταχωρητή W. Αν το περιεχόμενο του καταχωρητή f γίνει μηδέν, ενεργοποιείται το ψηφίο μηδενικού αποτελέσματος Z.

Στο επόμενο τεύχος θα παρουσιάσουμε το τέταρτο και τελευταίο άρθρο της εκπαιδευτικής σειράς προγραμματισμού του PIC. Στο διπλό τεύχος του Ιουλίου / Αυγούστου βρίσκεται το σχέδιο της πειραματικής πλακέτας, που βλέπετε στη φωτογραφία του άρθρου. Τα προγράμματα της σειράς, καθώς και άλλα παραδείγματα βρίσκονται στη δισκέτα 946196-1.

ADDWF	Add W to f		
Syntax	addwf f,d	Statusbits	C,DC,Z
Περιγραφή	Προσθέτει το περιεχόμενο του καταχωρητή W στον καταχωρητή f και αποθηκεύει το αποτέλεσμα αν d = 0 : στον καταχωρητή W αν d = 1 : στον καταχωρητή f		
Παράδειγμα:	addwf us_Register, F		

940062-III-T13

```
; Add two 32-Bit numbers N and Z. The result is in N.
;   If N+Z > 2**32 the CARRY is set
;
; N_LL, Z_LL: Bit  0.. 7
; N_LH, Z_LH: Bit  8..15
; N_HL, Z_HL: Bit 16..23
; N_HH, Z_HH: Bit 24..31
;
ADD_LL    movf    Z_LL, W   ; Load W with Z_LL
          addwf   N_LL, F   ; ADD N_LL and W, store at N_LL
          btfsc 3,0         ; Skip if Carry = 0
            goto  CARRY_LH  ;

          movf    Z_LH, W   ; Load W with Z_LH
ADD_LH    addwf   N_LH, F   ; ADD N_LH and W, store at N_LH
          btfsc 3,0         ; Skip if Carry = 0
            goto  CARRY_HL  ;

          movf    Z_HL, W   ; Load W with Z_HL
ADD_HL    addwf   N_HL, F   ; ADD N_HL and W, store at N_HL
          btfsc 3,0         ; Skip if Carry = 0
            goto  CARRY_HH  ;

          movf    Z_HH, W   ; Load W with Z_HH
ADD_HH    addwf   N_HH, F   ; ADD N_HH and W, store at N_HH
          retlw   000H      ; Return, load W with 000H

; Routine for the Ripple-Carry

CARRY_LH  incfsz  Z_LH, W   ; Load W with Z_LH + 1, Skip if W = 0
            goto  ADD_LH    ;
CARRY_HL  incfsz  Z_HL, W   ; Load W with Z_HL + 1, Skip if W = 0
            goto  ADD_HL    ;
CARRY_HH  incfsz  Z_HH, W   ; Load W with Z_HH + 1, Skip if W = 0
            goto  ADD_HH    ;
          setc              ; Overrun => Carry = 1
          retlw   000H      ; Return, load W with 000H
```

940062-III-T14

SUBWF	Subtract W from f		
Syntax	subwf f,d	Statusbits	C,DC,Z
Περιγραφή	Αφαιρεί από τον καταχωρητή f το περιεχόμενο του W και αποθηκεύει το αποτέλεσμα: αν d = 0 : στον καταχωρητή W αν d = 1 : στον καταχωρητή f		
Παρατήρηση	Η αφαίρεση γίνεται σύμφωνα με τη μέθοδο του συμπληρώματος ως προς 2. Αυτό σημαίνει, ότι το ψηφίο υπολοίπου (Carry) ενεργοποιείται όταν η πράξη δεν έχει υπόλοιπο. f > W => C = 1, Z = 0 f = W => C = 1, Z = 1 f < W => C = 0, Z = 0		
Παράδειγμα:	subwf us_Register, F		

940062-III-T15

INCF	Increment f		
Syntax	incf f,d	Statusbits	Z
Περιγραφή	Το περιεχόμενο του f αυξάνεται κατά 1 και το αποτέλεσμα αποθηκ. αν d = 0 : στον καταχωρητή W αν d = 1 : στον καταχωρητή f		
Παράδειγμα:	incf us_Register, F		

940062-III-T16

DECF	Decrement f		
Syntax	decf f,d	Statusbits	Z
Περιγραφή	Το περιεχόμενο του f μειώνεται κατά 1 και το αποτέλεσμα αποθηκ αν d = 0 : στον καταχωρητή W αν d = 1 : στον καταχωρητή f		
Παράδειγμα:	decf us_Register, F		

940062-III-T17

Προγραμματισμός του PIC

Μέρος 4. Διακλαδώσεις και λογικές εντολές

Στο τέταρτο και τελευταίο μέρος της εκπαιδευτικής σειράς προγραμματισμού θα αναφερθούμε στις υπόλοιπες εντολές του μικροελεγκτή PIC 16C5x

Συνεχίζουμε την ανάλυση του προγράμματος που δημοσιεύσαμε στο προηγούμενο άρθρο. Συνήθως, η επεξεργασία των εντολών ξεκινά από τις κατώτερες διευθύνσεις της μνήμης και καταλήγει στις υψηλότερες. Γιά τις τυχόν διαφοροποιήσεις στη σειρά εκτέλεσης του προγράμματος είναι υπεύθυνες οι εντολές ελέγχου ροής. Η απλούστερη εντολή του μικροελεγκτή είναι η εντολή NOP (γραμμή 106). Αυτή αυξάνει το περιεχόμενο του απαριθμητή προγράμματος κατά 1. Οι εντολές διακλάδωσης χωρίζονται σ' αυτές που εκτελούνται μόλις εκπληρωθεί μιά συνθήκη και σ' αυτές που εκτελούνται σε κάθε περίπτωση. Στην τελευταία κατηγορία ανήκουν οι εντολές GOTO (γραμμή 108) και CALL (γραμμή 135).

Εντολές διακλάδωσης χωρίς συνθήκη

Πρέπει να παρατηρήσουμε ότι η εντολή GOTO συνοδεύεται μόνο από τα 8 κατώτερα ψηφία (bit) της διεύθυνσης διακλάδωσης. Τα υπόλοιπα ψηφία καθορίζονται μέσω του καταχωρητή κατάστασης λειτουργίας (status reg.). Στον PIC 16C56, ο καταχωρητής καθορίζεται απο το ψηφίο 9, ενώ στον PIC 16C57 απο τα ψηφία 9 και 10 της διεύθυνσης διακλάδωσης. Πριν εκτελεστεί η εντολή πρέπει να ορίσουμε τα αντίστοιχα ψηφία του καταχωρητή κατάστασης λειτουργίας. Μετά από το μηδενισμό του μικροελεγκτή, τα ψηφία έχουν την τιμή 0. Ετσι, εκτελώντας την εντολή GOTO στη διεύθυνση μηδενισμού (1FFH στην περίπτωση των PIC 16C54 και 55, $3FF_H$ γιά τον PIC 16C56 και $7FF_H$ γιά τον PIC 16C57), αρχίζει η επεξεργασία από την αρχή του προγράμματος (γραμμή 145). Στον κλασικό προγραμματισμό, τα υποπρογράμματα αντικαθιστούν επαναλαμβανόμενα κομμάτια του κώδικα. Στο σύγχρονο προγραμματισμό τα υποπρογράμματα βοηθούν στην καλύτερη δόμηση και κατανόηση του κώδικα. Το τέλος του υποπρογράμματος δηλώνεται με την εντολή RETLW. Το όγδοο ψηφίο της διεύθυνσης κλήσης του υποπρογράμματος πρέπει να έχει πάντα (!) την τιμή 0. Η εντολή CALL συνοδεύεται από τα ψηφία 0 ως 7 της διεύθυνσης έναρξης του υποπρογράμματος. Τα υπόλοιπα ψηφία (ψηφίο 9 στον PIC 16C56 και 9, 10 στον PIC 16C57) δηλώνονται μέσα από τα ψηφία PA0 και PA1 του καταχωρητή κατάστασης λειτουργίας (όπως στην εντολή GOTO). Η διεύθυνση κλήσης του υποπρογράμματος πρέπει να βρίσκεται μέσα στο πρώτο μισό της σελίδας μνήμης προγράμματος. Αυτή η αδυναμία του μικροελεγκτή μπορεί να παρακαμφθεί, βάζοντας σαν πρώτη εντολή του υποπρογράμματος την εντολή GOTO. Η εντολή GOTO θα "δείχνει" στο πρώτο μισό της σελίδας μνήμης προγράμματος, όπου θα συνεχίζεται το υποπρόγραμμα. Μόλις εκτελέσουμε την εντολή CALL, το περιεχόμενο του απαριθμητή προγράμματος (PC) αποθηκεύεται στη μνήμη σωρού. Καθώς υπάρχουν μόνο δυό θέσεις στη μνήμη σωρού, δεν μπορούμε να καλέσουμε ταυτόχρονα περισσότερα από δυό υποπρογράμματα. Στη μνήμη σωρού αποθηκεύεται ολόκληρο το περιεχόμενο του απαριθμητή προγράμματος. Αυτό σημαίνει, ότι το υποπρόγραμμα μπορεί να κληθεί από οποιοδήποτε σημείο του προγράμματος. Η εντολή RETLW δηλώνει το τέλος του υποπρογράμματος. Μόλις η εντολή εκτελεστεί από τον μικροελεγκτή, ο απαριθμητής προγράμματος φορτώνεται με το περιεχόμενο της μνήμης σωρού. Η εκτέλεση του προγράμματος συνεχίζεται με την επόμενη εντολή, που ακολουθεί την CALL. Η σταθερή τιμή που συνοδεύει την εντολή RETLW αποθηκεύεται

NOP	No OPeration		
Syntax	nop	Status bits	none
Description	no operation, program counter incremented by 1		
Example	*nop*		

GOTO	Unconditional Branch		
Syntax	goto <addr>	Status bits	none
Description	Direct branch to an address of which the 9 lower bits (0 to 8) come from <addr>; bit 9 of the PIC 16C56 and bits 9 and 10 of the PIC 16C57 are loaded from the PA <2:0> bits in the status register.		
Example	*goto Main_Start*		

CALL	Subroutine Call		
Syntax	call <addr>	Status bits	none
Description	Value (Program Counter + 1) placed onto stack. Branch to address <addr> in the lower half of the page (bit 8 = 0 !) defined by the status register (A9 = PA0, A10 = PA1 for PIC16C54 and 16C57 only).		
Example	*call Wait_ms*		

RETLW	Return literal to W		
Syntax	retlw k	Staus bits	none
Description	The W register is loaded with the 8-bit literal k. The program counter is loaded from the top of stack register 1. The content of stack register 2 is moved to stack register 1		
Example	*retlw 000H*		

BTFSC	Bit Test, skip if clear		
Syntax	btfsc f, b	Staus bits	none
Description	Skip next instruction if bit b in register f is 0.		
Example	*btfsc 3,0 ;jump if CARRY (C) = 0*		

IORLW	Inclusive OR literal with W		
Syntax	iorlw k	Status bits	Z
Description	The contents of the W register are OR'ed with the 8-bit literal k. The result is placed in the W register.		
Example	*iorlw 046H*		

XORLW	Exclusive OR literal and W		
Syntax	xorlw k	Status bits	Z
Description	The contents of the W register are XOR'ed with the 8-bit literal k. The result is placed in the W register.		
Example	*xorlw 046H*		

COMF	Complement f		
Syntax	comf f, d	Status bits	Z
Description	The contents of register f are complemented. d = 0: result stored into W register d = 1 result stored into register f		
Example	*movlw 01010111B ;* *movwf us_Register ; us_Register = 01010111B* *comf us_Register, 1 ; us_Register = 10101000B*		

RRF	Rotate Right f through Carry		
Syntax	rrf f, d	Status bits	C
Description	The contents of register f are rotated one bit to the right through the Carry Flag. d = 0: result stored into W register d = 1: result stored into register f		
Example	*movlw 10011010B ; load 10011010 into register RReg* *movwf RReg ; RReg = 10011010* *bsf 3,0 ; C = 1* *rrf RReg, 1 ; => RReg = 11001101; C = 0*		

στον καταχωρητή W. Μ' αυτόν τον τρόπο μπορούμε, γιά παράδειγμα, να ελέγξουμε από ποιό σημείο επέστρεψε το υποπρόγραμμα.

Εντολές διακλάδωσης υπό συνθήκη

Στους μικροελεγκτές PIC 16C5X, οι εντολές διακλάδωσης υπό συνθήκη έχουν μιά ιδιαιτερότητα: η διεύθυνση διακλάδωσης είναι πάντα σταθερή. Οταν εκληρώνεται η συνθήκη, παρακάμπτεται η επόμενη εντολή. Γιά να διακλαδώσετε το πρόγραμμα σε μιά άλλη διεύθυνση, μπορείτε να τοποθετήσετε, μετά την εντολή διακλάδωσης υπό συνθήκη, την εντολή GOTO. Ετσι, όταν δεν (!) θα πληρούται η συνθήκη, το πρόγραμμα θα συνεχίζει εκεί που "δείχνει" η εντολή GOTO (γραμμές 130 και 131). Υπάρχουν τέσσερεις εντολές διακλάδωσης υπό συνθήκη. Οι εντολές BTFSC f,b και BTFSS f,b ελέγχουν την τιμή του ψηφίου "b" στον καταχωρητή "f". Οι εντολές DECFSZ f,d και INCFSZ f,d αυξάνουν (αντίστοιχα μειώνουν) κατά 1 το περιεχόμενο του καταχωρητή "f". Οταν το περιεχόμενο του καταχωρητή γίνει 0 παρακάμπτουν την επόμενη εντολή. Η καινούργια τιμή αποθηκεύεται ξανά στον ίδιο καταχωρητή (d=1) ή στον καταχωρητή εργασίας W (d=0). Στην τελευταία περίπτωση, το περιεχόμενο του καταχωρητή "f" παραμένει το ίδιο.

Παράδειγμα: σύγκριση με μιά σταθερή τιμή

Σε πολλές εφαρμογές χρειάζεται να συγκρίνουμε κάποιο καταχωρητή (θέση μνήμης) με μιά σταθερή τιμή. Το αντίστοιχο πρόγραμμα φαίνεται στον πίνακα 1. Γιά να συγκρίνουμε δυό καταχωρητές χρησιμοποιούμε, στη θέση της εντολής movlw, την εντολή movf.

Λογικές εντολές

Οι μικροελεγκτές PIC 16C5X αναγνωρίζουν επτά λογικές εντολές. Η εντολή XORWF χρησιμοποιήθηκε ήδη στο προηγούμενο πρόγραμμα. Δεν υπάρχουν πολλά πράγματα που πρέπει να πούμε γιαυτές τις εντολές. Η λογική πράξη εκτελείται ανάμεσα στους δυό καταχωρητές, ψηφίο προς ψηφίο (ψηφίο 0 με το ψηφίο 0, ψηφίο 1 με το ψηφίο 1 κλπ.). Οπως συμβαίνει και με την εντολή AND, το αποτέλεσμα μπορεί να αποθηκευτεί ξανά στον καταχωρητή "f" του αρχείου καταχωρητών ή στον καταχωρητή W. Οταν το αποτέλεσμα μιάς λογικής πράξης είναι 0, ενεργοποιείται το ψηφίο μηδενικού αποτελέσματος Z (Z flag). Οι εντολές ολίσθησης συγκαταλέγονται συνήθως, στις λογικές εντολές. Οι μικροελεγκτές PIC αναγνωρίζουν δυό εντολές ολίσθησης: δεξιόστροφη (Rotate Right) και αριστερόστροφη (Rotate Left). Η ολίσθηση πραγματοποιείται μέσω του ψηφίου υπολοίπου (Carry bit). Δεξιόστροφη ονομάζεται η ολίσθηση όταν το σημαντικότερο ψηφίο της λέξης ολισθαίνει προς το λιγότερο σημαντικό ψηφίο. Μετά την εκτέλεση της εντολής, στη θέση του σημαντικότερου ψηφίου βρίσκεται το ψηφίο του υπολοίπου και στη θέση του υπολοίπου το λιγότερο σημαντικό ψηφίο της λέξης. Ακριβώς αντίθετα λειτουργεί η αριστερόστροφη ολίσθηση. Οπως συμβαίνει μ' όλες τις εντολές που επεξεργάζονται λέξεις των 8 ψηφίων (Byte), το αποτέλεσμα μπορεί να αποθηκευτεί ξανά στο αρχείο καταχωρητών ή στον καταχωρητή W.

Εντολές επεξεργασίας ψηφίων

Η σειρά μικροελεγκτών PIC 16C5X διαθέτει τέσσερεις εντολές επεξεργασίας ψηφίων. Οι δυό από αυτές (BTFSC και BTFSS) ελέγχουν την τιμή ενός ψηφίου, από το αρχείο καταχωρητών και αλλάζουν ανάλογα τη ροή του προγράμματος. Σ' αυτές τις δυό εντολές αναφερθήκαμε προηγουμένως, όταν εξηγούσαμε τις εντολές διακλάδωσης υπό συνθήκη. Οι άλλες δυό εντολές αλλάζουν την τιμή ενός συγκεκριμένου ψηφίου μέσα στη μνήμη του αρχείου καταχωρητών. Η πρώτη παράμετρος της εντολής δείχνει τη διεύθυνση του καταχωρητή μέσα στη μνήμη και η άλλη τη θέση του ψηφίου στον καταχωρητή. Η τιμή 0 αντιστοιχεί στο λιγότερο σημαντικό ψηφίο (LSB) του καταχωρητή. Στη γραμμή 133 του

RLF	Rotate left f through Carry		
Syntax	rlf f, d	Status bits	C
Description	The contents of register f are rotated one bit to the left through the Carry Flag. d = 0: result stored into W register d = 1: result stored into register f		
Example	*movlw 10011010B ; load 10011010 into register RReg* *movwf RReg ; RReg = 10011010* *bsf 3,0 ; C = 1* *rlf RReg, 1 ; => RReg = 00110101; C = 0*		

BSF	Bit Set f		
Syntax	bsf f, b	Status bits	none
Description	Bit b in register f is set to 1		
Example	*bsf Port_A, 000H ; set bit 0 of Port_A*		

BTFSS	Bit test, skip if Set		
Syntax	btfss f, b	Status bits	none
Description	Skip next instruction if bit b in register f is 1.		
Example	*btfss 3,2 ;jump if zero (Z = 1)*		

INCFSZ	Increment f, skip if 0		
Syntax	incfsz f, d	Status bits	none
Description	The contents of register f are incremented. d = 0: result stored into W register d = 1: result stored into register Skip next instruction if the result is 0.		
Example	*incfsz us_Register, F*		

DECFSZ	Decrement f, skip if 0		
Syntax	decfsz f, d	Staus bits	none
Description	The contents of register f are decremented. d = 0: result stored into W register d = 1: result stored back into register f Skip next instruction if the result is 0.		
Example	*decfsz us_Register, F*		

ANDWF	AND W with f		
Syntax	andwf f, d	Status bits	Z
Description	AND the W register with register f. d = 0: result stored into W register d = 1: result stored into register f.		
Example	*andwf us_Register, W*		

IORWF	Inclusive OR W with f		
Syntax	iorwf f, d	Status bits	Z
Description	Inclusive OR the W register with register f. d = 0: result stored into W register d = 1: result stored into register f		
Example	*iorwf us_Register, W*		

XORWF	Exclusive OR W with f		
Syntax	xorwf f, d	Status bits	Z
Description	Exclusive OR the contents of the W register with register f. d = 0: result stored into W register d = 1: result stored into register f		
Example	*xorwf us_Register, W*		

ANDLW	AND literal and W		
Syntax	andlw k	Status bits	Z
Description	The contents of the W register are ANDed with the 8-bit literal k. The result is placed in the W register.		
Example	*andl 046H*		

προγράμματος βρίσκεται η εντολή BCF. Η εντολή αυτή μηδενίζει το ψηφίο που ελέγχει τη θύρα RA0, με αποτέλεσμα να φωτίσει το LED. Η εντολή BSF στη γραμμή 137 θέτει το ψηφίο ελέγχου της θύρας RA0 = 1 και το LED σβήνει. Κατά τη χρήση των εντολών BSF και BCF πρέπει να προσέξετε τα παρακάτω: γιά να αλλάξει ο μικροελεγκτής την τιμή κάποιου ψηφίου, διαβάζει ολόκληρη τη λέξη (8 bit), αλλάζει την τιμή του ψηφίου και γράφει τη λέξη ξανά στη μνήμη. Αλλάζοτας την τιμή ενός ψηφίου σε μιά θύρα I/O, ο μικροελεγκτής θα αποθηκεύσει στο μανδαλωτή (Latch) την τιμή όλων των ψηφίων της θύρας. Αν, κατόπιν, αλλάξουμε μιά γραμμή της θύρας, από είσοδο σε έξοδο, θα εμφανιστεί στο συγκεκριμένο ακροδέκτη του μικροελεγκτή η λογική στάθμη που αναγνώστηκε όταν ήταν ακόμα είσοδος. Γιαυτό, πριν αλλάξουμε μιά γραμμή, από είσοδο σε έξοδο, πρέπει να φροντίσουμε, ώστε το αντίστοιχο ψηφίο του μανδαλωτή να έχει την επιθυμητή τιμή.

Διασυνδέσεις με εξωτερικές συσκευές

Θύρες εισόδου / εξόδου (I/O)

Ας ξαναγυρίσουμε στο παράδειγμα του προγράμματος. Το κυρίως τμήμα του προγράμματος πρέπει να ξεκινά καθορίζοντας τις αρχικές συνθήκες των θυρών I/O (γραμμές 126 και 127). Οι μικροελεγκτές PIC διαθέτουν τις θύρες I/O και τον καταχωρητή επιλογής (OPTION). Γιά τον έλεγχο των θυρών και του καταχωρητή υπαρχουν δυό ειδικές εντολές. Η εντολή TRIS καθορίζει αν μιά θύρα θα συμπεριφέρεται σαν είσοδος ή σαν έξοδος, ενώ η εντολή OPTION αλλάζει το περιεχόμενο του καταχωρητή επιλογής. Οι διάφοροι παράμετροι I/O μπορούν να αλλάξουν κατά την εκτέλεση του προγράμματος. Γιά να αυξηθεί η αξιοπιστία της εφαρμογής, η εταιρία Microchip προτείνει τον επανακαθορισμό των παράμέτρων I/O σε τακτά χρονικά διαστήματα. Η πρόταση της Microchip είναι, ο επανακαθορισμός να γίνεται κάθε φορά πρωτού προσπελάσουμε τη θύρα.

Καταχωρητής επιλογής (Option)

Η εντολή OPTION αποθηκεύει το περιεχόμενο του καταχωρητή W στον καταχωρητή επιλογής. Στον καταχωρητή επιλογής και στη σημασία των ψηφίων του αναφερθήκαμε στο δεύτερο άρθρο της σειράς.

Λοιπές εντολές

Εκτός από τις εντολές που είδαμε στο πρόγραμμα, υπάρχουν ακόμα άλλες τρεις. Η εντολή SWAPF είναι δύσκολο να καταταχθεί σε μιά συγκεκριμένη κατηγορία. Η εντολή αντιμεταθέτει τα 4 λιγότερο σημαντικά ψηφία της λέξης (ψηφίο 0 ως 3, Low nibble) με τα ψηφία 4 ως 7 (high nibble). Η εντολή χρησιμοποιείται σε πράξεις με αριθμούς BCD. Ο απαριθμητής επιτήρησης (WDT) των μικροελεγκτών PIC ενεργοποιείται κατά τον προγραμματισμό του ολοκληρωμένου. Γιά τον μηδενισμό του υπάρχει η εντολή CLRWDT. Η εντολή αυτή μηδενίζει τον προδιαιρέτη, στην περίπτωση που τον συνδέσουμε με τον απαριθμητή επιτήρησης. Ο απαριθμητής επιτήρησης μηδενίζεται, επίσης, με την εντολή SLEEP. Μόλις εκτελεστεί η εντολή, ο μικροελεγκτής οδηγείται στην κατάσταση ηρεμίας και μειώνεται η κατανάλωσή του. Γιά να επανέλθει στην κανονική κατάσταση λειτουργίας, πρέπει να υπερχειλήσει ο απαριθμητής επιτήρησης ή να μηδενιστεί εξωτερικά ο μικροελεγκτής.

Το τέλος του προγράμματος

Γιά να δηλώσουμε το τέλος του προγράμματος πρέπει να καθορίσουμε την τιμή του δείκτη RESET. Αυτό γίνεται μέσω της εντολής ORG (γραμμή 144) του μεταφραστή Assembler. Το μέγεθος της μνήμης εξαρτάται από τον τύπο του μικροελεγκτή. Γιαυτό το σκοπό, η διεύθυνση μηδενισμού καθορίζεται με τη βοήθεια των εντολών IF-ENDIF, που βρίσκονται στην αρχή του προγράμματος.

BCF	Bit Clear f		
Syntax	bsf f, b	Status bits	none
Description	Reset bit b in register f to 0.		
Example	*bcf Port_A, 000H*	*; reset bit 0 of Port_A*	

TRIS	Load TRIS register (tristate port)		
Syntax	tris f	Status bits	none
Description	TRIS register f is loaded with the contents of the W register. Valid values for f are: 05H for Port A, 06H for Port b, and 07H for Port C. A set bit (1) defines a port pin as an input; a reset bit (0) defines a port pin as an output		
Example	*movlw 00EH* *tris Port_A*	*; set RA0 to output* *;*	

OPTION	Load Option Register		
Syntax	option	Status bits	none
Description	The contents of the W register is loaded into the OPTION register.		
Example	*movlw 00100110B* *option*	*; Transition on RTTC pin* *; Increment on low to high* *; transition on RTTC pin* *; Prescaler to RTTC* *; Prescaler value = 1 : 128* *; Load contents of W register* *; into OPTION register*	

SWAPF	Swap f		
Syntax	swapf f, d	Status bits	none
Description	The upper and lower ninbbles of of register f are exchanged. d = 0: result stored into W register d = 1 result stored into register f		
Example	*movlw 10011010B* *movwf RReg* *swapf RReg, 1*	*;* *; RReg = 10011010B* *; RReg = 10101001B*	

CLRWDT	Clear Watchdog Timer		
Syntax	clrwdt	Status bits	TO, PD
Description	Reset the watchdog timer. Also reset the prescaler of the WDT. Set PD and TO.		
Example	*clrwdt*		

SLEEP	Put into sleep mode		
Syntax	sleep	Status bits	TO, PD
Description	The power-down status bit (Pd) is cleared. Time-out status bit (TO) is set. Watchdog Timer and its prescaler are cleared.		
Example	*sleep*		

```
          ; Compare Register 'us_Register' with 046H

Comp_Value equ 046H

COMPARE   movlw Comp_Value      ; Load W with 046H
          xorwf us_Register, W  ; XOR W with us_Register
          btfsc 3,2             ; W = 0?
            goto EQUAL          ;   Yes: us_Register =  046H
NOT_EQUAL                       ;   No:  us_Register <> 046H
          movlw Comp_Value      ;
          subwf us_Register, W  ; us_Register - W
          btfsc 3,0             ; W > 0 (Carry = 1?)
            goto GREATER        ;    Yes: us_Register > 046H
          goto LESS             ;    No:  us_Register < 046H
LESS      .....                 ; Routine for us_Register < 046H
GREATER   .....                 ; Routine for us_Register > 046H
EQUAL     .....                 ; Routine for us_Register = 046H
```