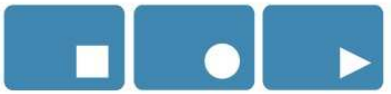
ΥΠΟΥΡΓΕΙΟ ΕΘΝΙΚΗΣ ΠΑΙΔΕΙΑΣ ΚΑΙ ΘΡΗΣΚΕΥΜΑΤΩΝ
ΕΙΔΙΚΗ ΥΠΗΡΕΣΙΑ ΔΙΑΧΕΙΡΙΣΗΣ ΕΠΕΑΕΚ
ΕΥΡΩΠΑΪΚΗ ΕΝΩΣΗ
ΣΥΓΧΡΗΜΑΤΟΔΟΤΗΣΗ
ΕΥΡΩΠΑΪΚΟ ΚΟΙΝΩΝΙΚΟ ΤΑΜΕΙΟ
ΕΥΡΩΠΑΪΚΟ ΤΑΜΕΙΟ ΠΕΡΙΦΕΡΕΙΑΚΗΣ ΑΝΑΠΤΥΞΗΣ



ΠΑΙΔΕΙΑ ΜΠΡΟΣΤΑ
2ο Επιχειρησιακό Πρόγραμμα
Εκπαίδευσης και Αρχικής
Επαγγελματικής Κατάρτισης

Πρόγραμμα Αναμόρφωσης Προπτυχιακών Σπουδών

Τμήματος Μαθηματικών

Πανεπιστημιο Πατρων

Μάθημα: Εισαγωγή στην Επιστήμη των Υπολογιστών

6η Εβδομαδα - Δεττερα 9 Νοεμβριοτ 2005

Δ. **ΤΜΗΜΑ ΕΛΕΓΧΟΤ** (Control), **ΚΑΤΑΧΩΡΗΤΕΣ** (Registers), **ΔΙΑΤΛΟΙ** (Busses), **ΚΤΚΛΟΙ ΜΗΧΑΝΗΣ** (Machine Cycles) και **ΓΛΩΣΣΑ ΜΗΧΑΝΗΣ** (Machine Language) **ΤΜΗΜΑ ΑΠΟΜΝΗΜΟΝΕΤΣΗΣ** (Memory)

1. Τμήμα Ελέγχου

Το τμήμα ελέγχου αποτελεί τον «**νου**» του Η.Τ. που κατευθύνει το όλο σύστημα για την ορθή εκτέλεση των εντολών που του δίδονται. Έτσι, στο τμήμα ελέγχου αποκωδικοποιούνται οι διάφορες εντολές (η μία μετά την άλλη διαδοχικά βάσει της σειράς που προβλέπεται) και δημιουργούνται τα κατάλληλα σήματα ελέγχου για τη μεταφορά των δεδομένων και τον απαιτούμενο συγχρονισμό στην εκτέλεση των εντολών κ.λ.π. Φυσικά οι εντολές είναι γραμμένες σε γλώσσα Μηχανής (την μόνη που καταλαβαίνει ο Η.Τ.).

2. Βασικοί καταχωρητές και δίαυλοι

Θεμελιώδη ρόλο στην όλη λειτουργία του τμήματος ελέγχου έχουν οι δύο καταχωρητές **CIR**(Current Instruction Register) και **PC/IC** (Program/Instruction Counter) του τμήματος, στους οποίους αποθηκεύονται **η προς εκτέλεση εντολή** και **η θέση μνήμης όπου έχει αποθηκευτεί η επόμενη προς εκτέλεση εντολή** (βλέπε το κλασσικό διάγραμμα ενός ψηφιακού υπολογιστή του Neumann, στο σχήμα 1). Βέβαια, οι δυο παραπάνω καταχωρητές μαζί με μερικούς άλλους (όπως π.χ. ο MAR και ο Data/Buffer Register - DR) και με τη βοήθεια **των τριών διαύλων επικοινωνίας** (Busses) που ενυπάρχουν σε κάθε Η.Υ. (Ο δίαυλος μεταφοράς των δεδομένων - **Data Bus**, ο δίαυλος μεταφοράς της διεύθυνσης μνήμης - **Address bus** και ο δίαυλος μεταφοράς των σημάτων ελέγχου - **Control bus**), επιτυγχάνεται η αριστοποίηση της επικοινωνίας των ζωτικών τμημάτων των Η.Υ. και η βέλτιστη απόδοση της λειτουργίας του. Τέλος, οι βασικές λειτουργίες ενός Η.Υ. εκτελούνται σε καθορισμένα χρονικά διαστήματα που προσδιορίζονται από **τους παλμούς του ρολογιού της μηχανής**.

Έτσι, η χρονική διάρκεια ενός βασικού κύκλου μηχανής αντιστοιχεί σε ένα σταθερό πλήθος παλμών, ενώ το πλήθος των βασικών λειτουργιών μιάς εντολής εξαρτάται απ΄ αυτή την ίδια την εντολή.

3. Κύκλοι μηχανής

Από το διάγραμμα της διαδοχής των λειτουργιών (βλέπε σχ. 16) είναι σαφές ότι μετά την **προκαταρκτική εργασία** - της τοποθέτησης των εντολών του προγράμματος και των δεδομένων του στην μνήμη, οπότε ο καταχωρητής **IC** (Απαριθμητής εντολών ή μετρητής προγράμματος) περιέχει τη θέση μνήμης της πρώτης προς εκτέλεση εντολής. **Για την επεξεργασία μιας εντολής** ακολουθούν **δύο υποσύνολα βασικών λειτουργιών**, που ονομάζονται **κύκλοι μηχανής** (Machine Cycles). Ο πρώτος κύκλος, που ονομάζεται **κύκλος ανάκλησης** (fetch cycle), ανακαλεί από την μνήμη μια εντολή στην μονάδα ελέγχου, όπου εξετάζεται για να καθοριστεί το είδος της και ο τρόπος επεξεργασίας της. Ο δεύτερος κύκλος, που ονομάζεται **κύκλος εκτέλεσης** (execution cycle), καθορίζει την εκτέλεση των βασικών λειτουργιών της εντολής.

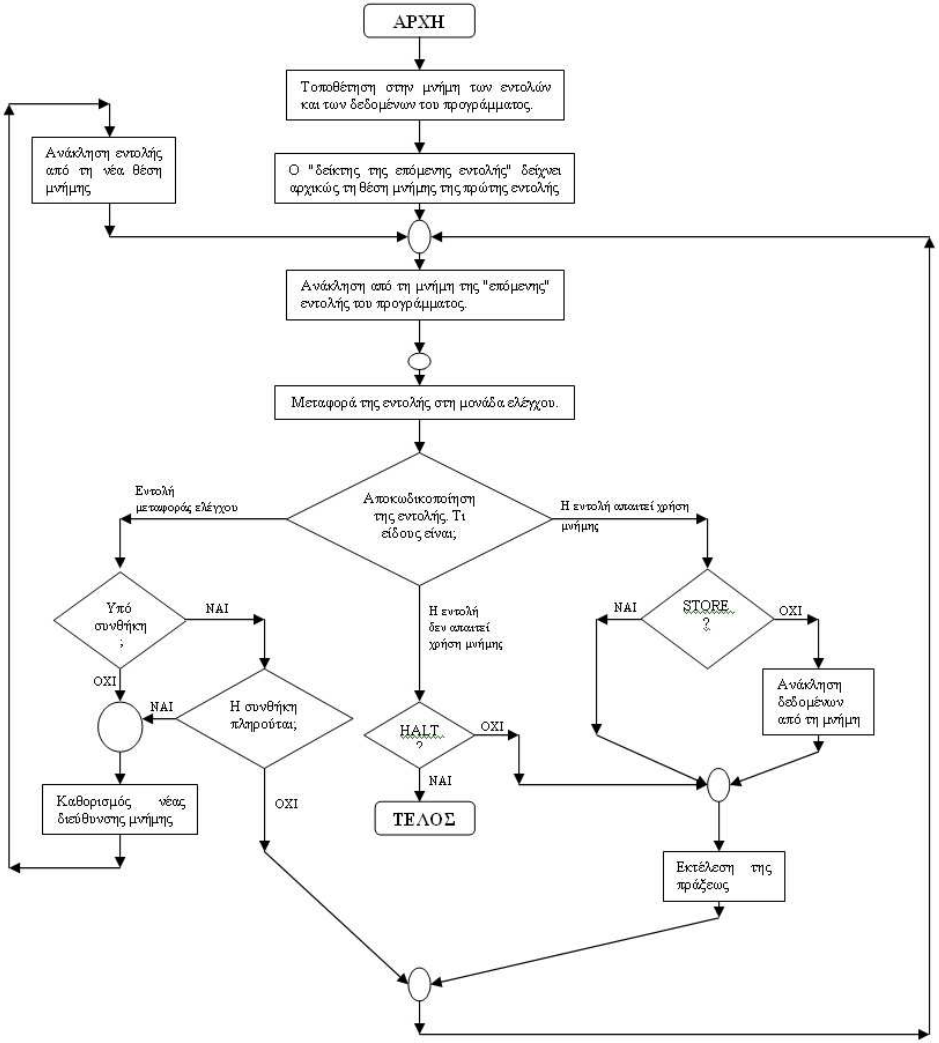
Προφανώς **ο κύκλος ανάκλησης είναι ο ίδιος** για όλες τις εντολές, ο δε **κύκλος εκτέλεσης διαφέρει από εντολή σε εντολή**. Π.χ. η εντολή της πρόσθεσης απαιτεί έναν κύκλο εκτέλεσης, ενώ η εντολή του πολλαπλασιασμού απαιτεί περισσότερους. Το σχήμα 16, που δίδει το διάγραμμα του τρόπου με το οποίο επιτυγχάνεται η διαδοχική εκτέλεση των εντολών ενός προγράμματος (σε γλώσσα μηχανής) είναι εύγλωττο. Π.χ. είναι σαφές ότι **το σύνολο των εντολών** που εκτελεί μια μηχανή, χωρίζεται **σε 3 είδη**, τα:

1. Εντολές **που απαιτούν** χρήση μνήμης, όπως π.χ. **ADD 15**

2. Εντολές **που δεν απαιτούν** χρήση μνήμης, όπως π.χ. **HALT**

3. Εντολές μεταφοράς ελέγχου (αλλαγής ροής προγράμματος), όπως π.χ. **JUMP 20**.

Σχήμα 16. Τρόπος εκτέλεσης διαδοχικών εντολών προγράμματος σε Γ.Μ.



Εξάλλου στο σχήμα 16 είναι σαφείς και ξέχωροι οι δύο κύκλοι της μηχανής· ο κύκλος της ανάκλησης της εντολής και ο κύκλος της εκτέλεσης (ανάλογα με την εντολή που εκτελείται). Τέλος, για την σαφήνεια της όλης διαδικασίας θα παραθέσουμε παρακάτω λεπτομερειακά τις βασικές λειτουργίες που λαμβάνουν χώρα σε κάθε έναν από αυτούς τους δύο κύκλους, για την περίπτωση της εκτέλεσης της συγκεκριμένης εντολής (με χρήση μνήμης):

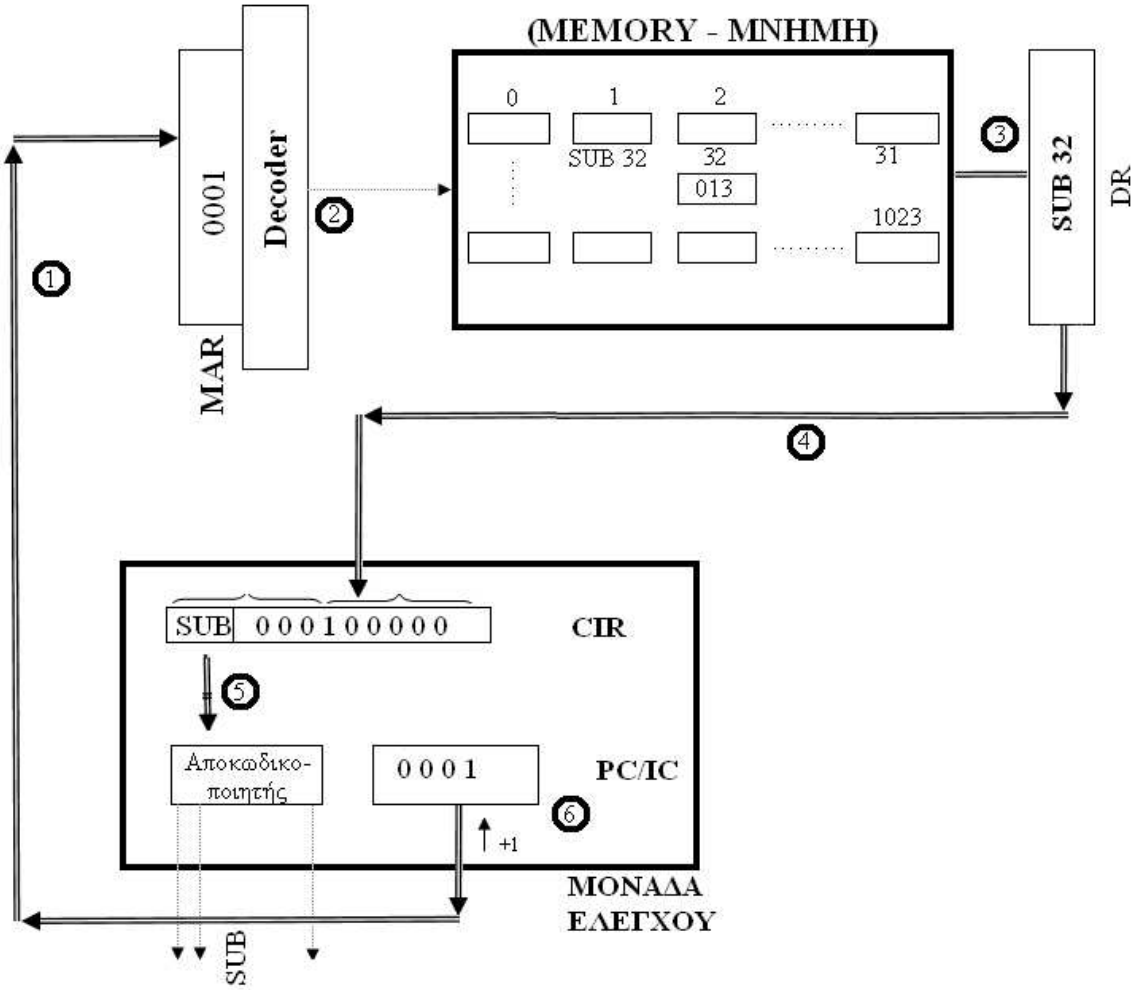
SUB 32

σε μηχανή με MAR των 10 bits (άρα με μνήμη 1024 θέσεων) από τη διεύθυνση 0 (=0000000000) έως την 1023 (=1111111111).

Φυσικά για την εκτέλεσή της, που υποτίθεται ότι είναι αποθηκευμένη στη θέση 1 της μνήμης του Η.Υ., θα πρέπει ο καταχωρητής IC να έχει τιμή 1, οπότε στο παρακάτω σχήμα 17, απεικονίζονται **όλες οι εργασίες που λαμβάνουν χώρα για την ανάκληση της εντολής** (αριθμούνται διαδοχικά), που είναι:

1°: Απόστολή της διεύθυνσης από τον PC στον MAR.
2°: Ο Decoder αποκωδικοποιεί τη θέση μνήμης 1.
3°: Μεταφορά του περιεχομένου της θέσης μνήμης 1 στον DR.
4°: Μεταβίβαση του περιεχομένου του DR στον CIR για αποκωδικοποίηση.
5°: Αποκωδικοποίηση της εντολής και ετοιμασία της διαδικασίας για SUB.
6°: Το περιεχόμενο του καταχωρητή PC αυξάνεται κατά 1.

Σχήμα 17. Διάγραμμα κύκλου ανάκλησης εντολής



Στη συνέχεια ξεκινά ο κύκλος της εκτέλεσης, οπότε λαμβάνουν χώρα οι παρακάτω λειτουργίες (εντολή με χρήση μνήμης), όπως αριθμούνται, στο σχήμα 18:

1°: Από τον CIR αποστέλλεται η διεύθυνση μνήμης που εμπλέκεται στον MAR.

2°: Ο Decoder ενεργοποιεί τη θέση μνήμης 32.

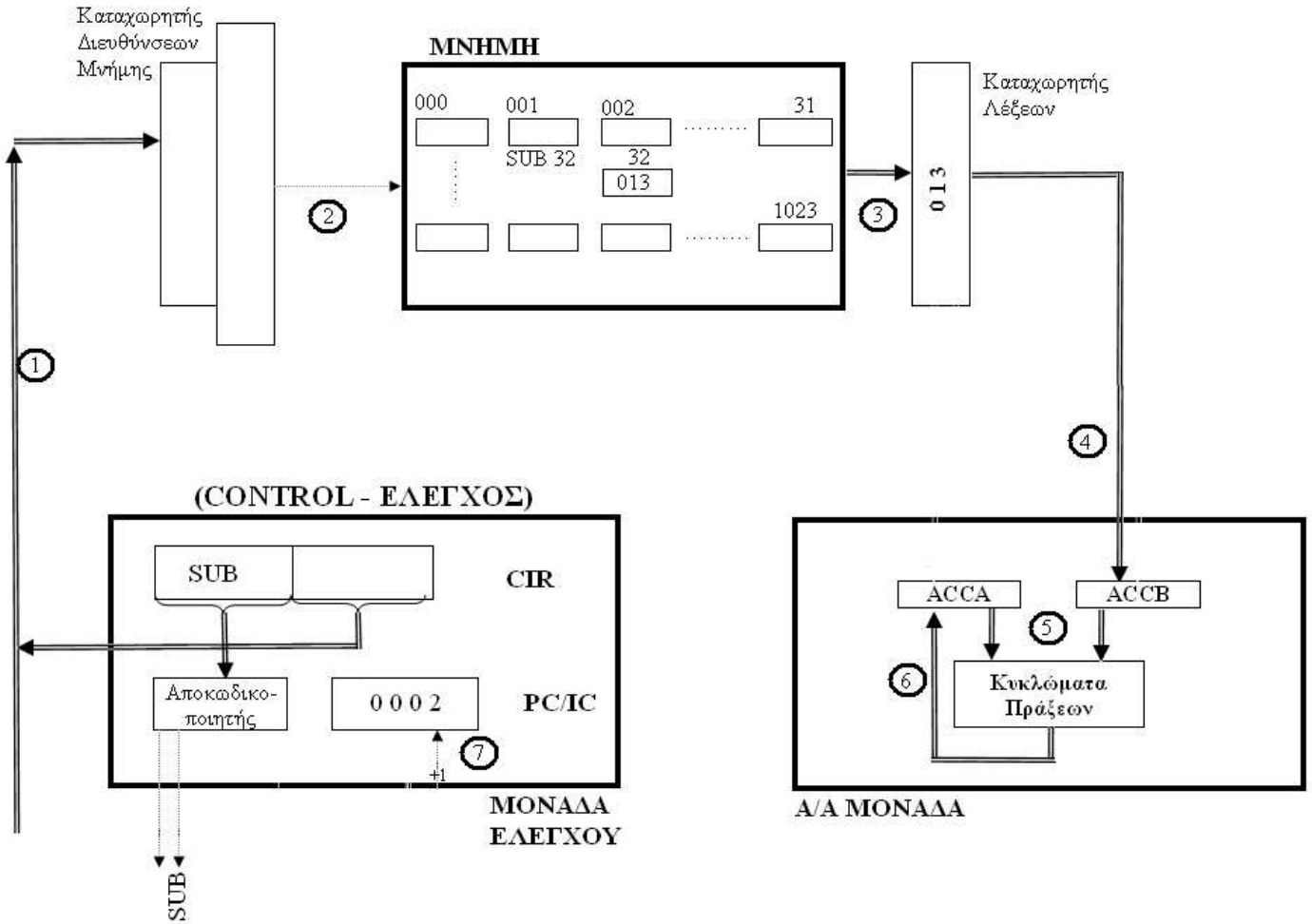
3°: Το περιεχόμενο της θέσης μνήμης 32 αποστέλλεται στον DR.

4°: Το περιεχόμενο του DR αποστέλλεται στην ALU για επεξεργασία.

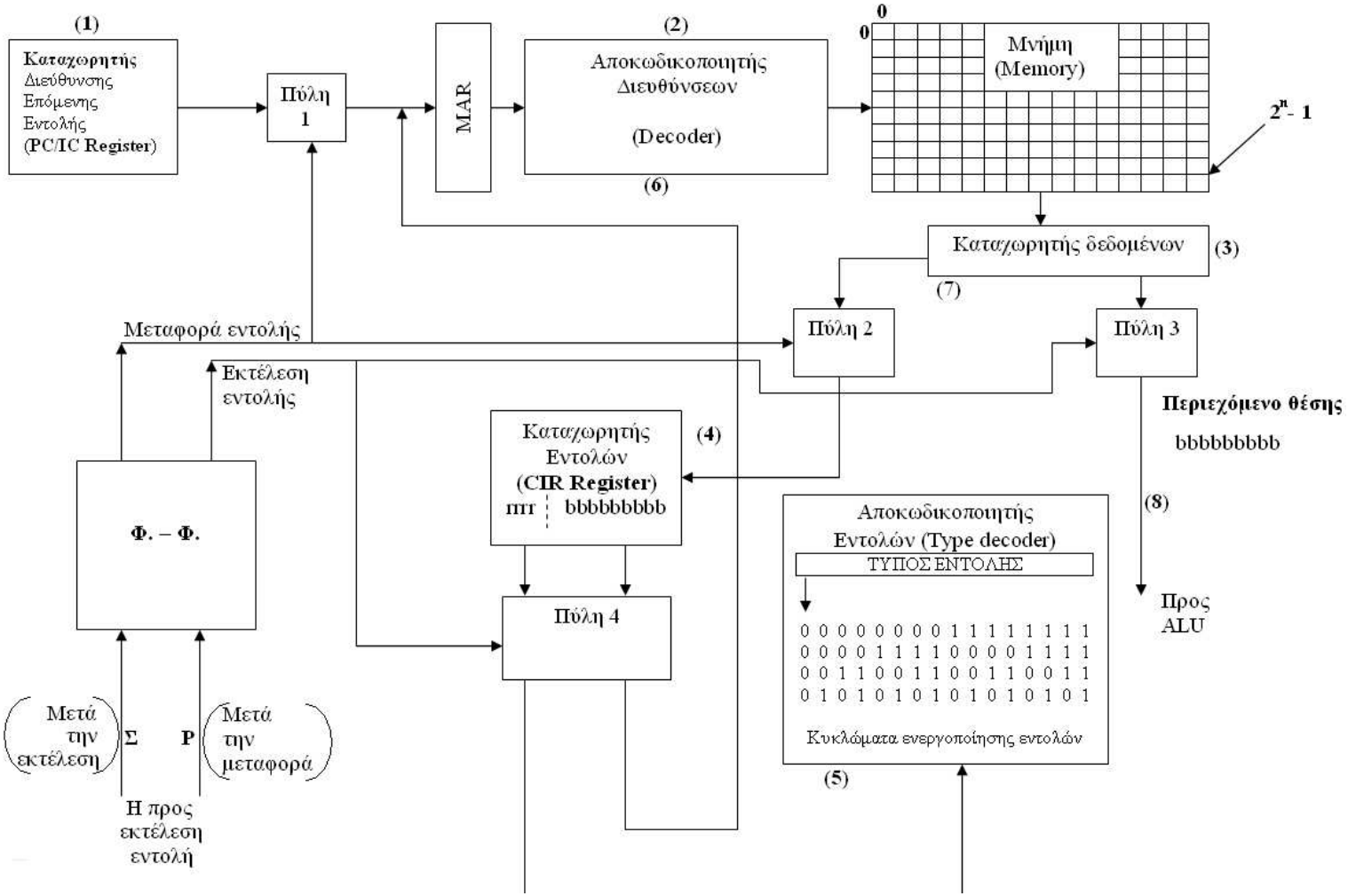
5°: Εκτελείται η πράξη της αφαίρεσης.

6°: Η σχηματιζόμενη διαφορά αποθηκεύεται στον ACCA.

Σχήμα 18. Διάγραμμα κύκλου εκτέλεσης της εντολής



Σχήμα 19. Ανάκληση και Εκτέλεση εντολών γλώσσας μηχανής



Τέλος, το σύνθετο διάγραμμα των λειτουργιών με τη χρήση Flip-Flop για την ενεργοποίηση των 2 κύκλων αποδίδεται από το παρακάτω σχήμα 19, ένας σύντομος σχολιασμός του οποίου ακολουθεί.
Κατ' αρχή, το Φλιπ-Φλοπ (Φ.-Φ.) είναι στην κατάσταση Σ (set) και συνεπώς γίνεται υλοποίηση της μεταφοράς της **προς εκτέλεση εντολής** (της πρώτης), με ενεργοποίηση των πυλών 1 και 2 από τη μνήμη στο καταχωρητή (CIR).
Μόλις ο προηγούμενος κύκλος της ανάκλησης ολοκληρωθεί, ένας παλμός του ρολογιού του Η.Υ. προκαλεί αλλαγή της κατάστασης του Φλιπ-Φλοπ, που τίθεται στην κατάσταση P (Reset) με αποτέλεσμα να ενεργοποιηθεί ο κύκλος της εκτέλεσης της εντολής, δίδοντας εντολή στην πύλη 4 να αφήσει τα δύο μέρη του καταχωρητή εντολών (rrrr :bbbbbbbbb) να περάσουν και το μεν πρώτο (rrrr) να μεταφερθεί στο αποκωδικοποιητή εντολών, ενώ συγχρόνως το δεύτερο τμήμα (bbb bbb bbb) μεταφέρεται στον MAR, όπου με τη βοήθεια του decoder προσδιορίζεται η θέση μνήμης που το περιεχόμενό της μεταφέρεται στον καταχωρητή δεδομένων (Data register), όπου δια μέσου της πύλης 3 μεταφέρεται στην ALU για να λάβει μέρος στην λειτουργία της εντολής.

Τέλος, κατά τη διάρκεια της εκτέλεσης, η διεύθυνση του καταγραφέα PC αυξάνεται κατά 1, έτσι ώστε όταν ο κύκλος της μεταφοράς αρχίσει να υπάρχει στον PC η διεύθυνση της επόμενης για εκτέλεση εντολής.

Παράδειγμα:

Ας υποθέσουμε το παρακάτω πρόγραμμα σε γλώσσα μηχανής, που είναι αποθηκευμένο στις αναγραφόμενες θέσεις μνήμης (σε παρένθεση οι εντολές συμβολικά).

Θέση Μνήμης	Εντολή Μηχανής	Η εντολή συμβολικά
1	0001 0000 010000	(LDA A 16)
2	0010 0000 010011	(ADD A 19)
3	0011 0000 010000	(STA A 16)

και ας παρακολουθήσουμε την πορεία της εκτέλεσής των στο σχήμα 19.

Φυσικά, στην αρχή, στο βήμα 1, τίθεται η τιμή 1 στον PC (θέση μνήμης της πρώτης προς εκτέλεση εντολής). Το Φ.-Φ. τίθεται στην κατάσταση Σ, πράγμα που ενεργοποιεί την μεταφορά της διεύθυνσης 1 διά της πύλης 1 στον MAR και στον αποκωδικοποιητή (βήμα 2) που υποκινεί τη μνήμη να θέσει το περιέχομενο της θέσης 1 στον καταχωρητή δεδομένων (βήμα 3) που μεταφέρει το περιεχόμενό του διά της πύλης 2 στον καταχωρητή εντολών (βήμα 4), που περιέχει τώρα την εντολή «**0001 0000 010000**». Στην συνέχεια το Φ.-Φ. ενεργοποιεί την κατάσταση Ρ για την εκτέλεση της εντολής, οπότε οι πύλες 3 και 4 αφήνουν τη διέλευση του «0001» στον αποκωδικοποιητή εντολών (βήμα 5) και του περιεχομένου «**0000010000**» στον MAR, οπότε ενεργοποιείται ο αποκωδικοποιητής (βήμα 6) και υποκινεί την μνήμη να θέσει το περιεχόμενο της θέσης 16 (≡0000010000) στον καταχωρητή δεδομένων (βήμα 7), που στη συνέχεια προωθείται στην ALU και εκτελείται η εντολή με αποτέλεσμα ο συσσωρευτής Α να περιέχει το περιεχόμενο της θέσης μνήμης 16.

Το Φ.-Φ.τίθεται σε κατάσταση Σ και με την ίδια διαδικασία μεταφέρεται η εντολή που βρίσκεται στη θέση 2, στον καταχωρητή εντολών, οπότε ξεκινάει η εκτέλεσή της γιατί το Φ.-Φ. τίθεται σε κατλάσταση Ρ, οπότε αθροίζεται το περιεχόμενο του συσσωρευτή Α με το περιεχόμενο της θέσης μνήμης 19, ενώ παράλληλα αυξάνεται το περιεχόμενο του PC κατά 1. Νέα αλλαγή του Φ.-Φ. προκαλεί την μεταφορά της εντολής 3 στον καταχωρητή εντολών, οπότε νέα αλλαγή του Φ.-Φ. προκαλεί την εκτέλεση της εντολής που είναι αποθήκευση του περιεχομένου του συσσωρευτή στη θέση μνήμης 16.

Δηλαδή, το παραπάνω πρόγραμμα **πρόσθεσε** τα περιεχόμενα των θέσεων μνήμης 16 και 19 **και έθεσε** το άθροισμά τους στη θέση μνήμης 16, ενώ το αρχικό περιεχόμενό της μνήμης σβήνεται.

Μια τελική παρατήρηση από την προηγούμενη διαδικασία. **Όταν γίνεται προσπέλαση της μνήμης με διεύθυνση που προέρχεται από τον PC** (καταχωρητή διεύθυνσης της επόμενης προς εκτέλεση εντολής) **η λέξη μνήμης** που εξάγεται στον καταχωρητη δεδομένων (DR) αντιστοιχεί σε εντολή της μηχανής και **οδεύει στον καταχωρητή εντολών (CIR)**. Ενώ όταν γίνεται **προσπέλαση μνήμης από τον καταχωρητή των εντολών** (κατά την εκτέλεση) **τότε** το περιεχόμενο της λέξης μνήμης **οδεύει στην ALU**, γιατί είναι δεδομένο (data).

4. Γλώσσα Μηχανής

α. Εισαγωγή

Το οποιοδήποτε **υπολογιστικό σύστημα**, που είναι αφιερωμένο για την εξυπηρέτηση ενός ειδικού έργου (όπως π.χ. οι κρατήσεις θέσεων για τις αεροπορικές εταιρείες, ή η παρακολούθηση ασθενών σε μονάδες εντατικής θεραπείας) **είναι σχεδιασμένο ως μια ολότητα**, που αποτελείται από διάφορα «**στρώματα**» (layers) - επίπεδα.

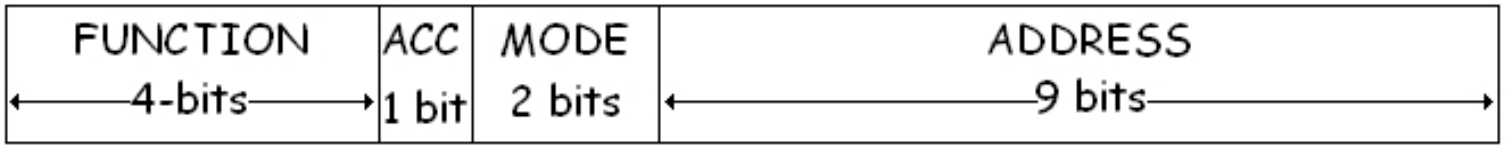
Το βασικό επίπεδο είναι οι διάφορες **λογικές πύλες** (ή μάλλον, **η μία η NAND**) και τα **διάφορα κυκλώματα**, με τα οποία επιτυγχάνονται η επεξεργασία και η αποθήκευση των πληροφοριών. Αυτό το επίπεδο είναι εκείνο που ονομάζουμε HARDWARE του συστήματος (H/W). **Μόνο του** το H/W **αδυνατεί** να κάνει οποιαδήποτε εργασία, όπως και κάθε άλλη μη προγραμματιζόμενη μηχανή.

Εάν όμως πλαισιωθεί το H/W **με το κατάλληλο σύνολο προγραμμάτων** (συλλογής συνόλων κατάλληλων εντολών, του SOFTWARE - λογισμικού), τότε **το σύστημα H/W - S/W**, αξιοποιώντας την τεράστια ταχύτητα που διαθέτει, **καθίσταται το τεχνολογικό θαύμα της εποχής μας**. Σ΄ αυτό λοιπόν το επίπεδο και στην ειδική μορφή του **προγραμματισμού σε γλώσσα μηχανής** θα στρέψουμε τώρα το ενδιαφέρον μας, γιατί είναι **πολύ στενά συνδεδεμένη** με το H/W του υπολογιστή. Έτσι, λοιπόν, θα υποθέσουμε ότι έχουμε έναν υποθετικό, **εκπαιδευτικό Η.Υ.**, τον «Miracle» με τα ακόλουθα βασικά χαρακτηριστικά.

β. Βασικά χαρακτηριστικά του Miracle

Η μηχανή αποτελείται **από μιά CPU με 2 16-bit συσσωρευτές** (accumulators), τον **A** και **B**, **ένα 9-bit μετρητή εντολών** (Program/Instruction counter - PC/IC) και **μία μνήμη με μέγεθος 512 λέξεων των 16 bits** η κάθε μία.

Η CPU έχει ένα σύνολο 16 εντολών που εκτελεί (instruction repertory), με την εξής δομή (4 εντολές είναι χωρίς διεύθυνση μνήμης - addresses):



Τέλος, οι 16 εντολές που μπορεί να εκτελέσει ο MIRACLE διακρίνονται **σε εντολές με διεύθυνση** (address) μνήμης, σε **εντολές χωρίς διεύθυνση**, που εμπλέκουν μόνο κάποιο συσσωρευτή και σε **εντολές αλλαγής της διαδοχικής ροής του προγράμματος** (μεταφοράς ελέγχου). Πιο συγκεκριμένα έχουμε τις εντολές που αντιπροσωπεύουν:

γ. Το ρεπερτόριο των εντολών της γλώσσας του Miracle

(i) Λειτουργίες με διεύθυνση (Κάθε μία εμπλέκει κάποιον συσσωρευτή)

	CODE	Συμβολισμός	ΛΕΙΤΟΥΡΓΙΑ (FUNCTION, τα 4 αρχικά bits)	ΠΑΡΑΔΕΙΓΜΑ
	0000	LDA	Βάλε το περιεχόμενο της θέσης μνήμης που ακολουθεί στον συσσωρευτή που αναγράφεται	(π.χ. LDA A 2)
	0001	STA	Αποθήκευσε το περιεχόμενο του συσσωρευτή που αναγράφεται στη θέση μνήμης που ακολουθεί	(π.χ. STA B 48)
	0010	ADD	Πρόσθεσε στον συσσωρευτή που αναγράφεται το περιεχόμενο της θέσης μνήμης που ακολουθεί	(π.χ. ADD A 49)
	0011	SUB	Αφαίρεσε από τον συσσωρευτή που αναγράφεται το περιεχόμενο της θέσης μνήμης που αναγράφεται	(π.χ. SUB B 25)
	0100	ORA	Εκτέλεσε την πράξη OR μεταξύ του συσσωρευτή που αναγράφεται και της θέσης μνήμης που ακολουθεί	(π.χ. ORA A 15)
	0101	AND	Όπως προηγούμενα για την πράξη της AND	(π.χ. AND A 17)
Εντολές Μεταφοράς Ελέγχου	0110	JMP	Πήγαινε και εκτέλεσε την εντολή που βρίσκεται στη θέση μνήμης που ακολουθεί (Πήδημα χωρίς όρους)	(π.χ. JMP 13)
	0111	BZE	Πήγαινε στην εντολή της θέσης μνήμης που ακολουθεί όταν το περιεχόμενο του συσσωρευτή **είναι μηδέν**	(π.χ. BZE A 15)
	1000	BNZ	Όπως προηγούμενα, αλλά με περιεχόμενο συσσωρευτή **διάφορο του μηδενός** (**N**on **Z**ero)	(π.χ. BNZ B 16)
	1001	BMI	Όπως προηγούμενα, αλλά με περιεχόμενο συσσωρευτή **αρνητικό** (**MI**nus)	(π.χ. BMI B 25)
	1010	BPL	Όπως προηγούμενα, αλλά με περιεχόμενο συσσωρευτή **θετικό** (**PL**us)	(π.χ. BPL A 14)

(ii) Λειτουργίες χωρίς διεύθυνση

1011	LRS	Μετατόπισε το περιεχόμενο του συσσωρευτή που αναγράφεται μια θέση δεξιά· (π.χ. **LRS A**)
1100	NEG	Αντιστροφή του προσήμου του συσσωρευτή που αναγράφεται· (π.χ. **NEG B**)
1101	INA	Εισαγωγή ενός αριθμού από τη μονάδα εισόδου στον αναφερόμενο συσσωρευτή· (π.χ. **INA A**) Προσοχή, **τα δεδομένα εισάγονται στον Η.Υ. δια μέσου συσσωρευτή** και όχι άμεσα στη μνήμη του.
1110	OUT	Εκτύπωση του περιεχομένου του αναφερόμενου συσσωρευτή στον εκτυπωτή· π.χ. **OUT B**
1111	HLT	Τέλος του προγράμματος

(iii) Καθορισμός των συσσωρευτών

Το 5ο bit καθορίζει τον Συσσωρευτή που εμπλέκεται στη λειτουργία της εντολής. Εάν είναι **0** εμπλέκεται ο **A**, εάν είναι **1** εμπλέκεται ο **B**.

(iv) Τρόπος πρόσβασης στη μνήμη

Το πεδίο mode (6ο και 7ο bit) καθορίζει τον τρόπο υλοποίησης της λειτουργίας. Έτσι, αν είναι:

00	**Άμεση πρόσβαση** LDA A # 1 (Immediate mode)	Ο αναφερόμενος αριθμός είναι ο operand, δηλαδή ο αριθμός που άμεσα εισάγεται στον αναφερόμενο συσσωρευτή.
01	**Ευθεία πρόσβαση** LDA A 35 (Direct mode)	Ο αναφερόμενος αριθμός 35 είναι η διεύθυνση μνήμης που βρίσκεται ο operand.
10	Index by accumulator A STA B 16A	Ο αναφερόμενος αριθμός προστιθέμενος στο περιεχόμενο του Acc. A δίδει τη διεύθυνση μνήμης όπου θα αποθηκευτεί το περιεχόμενο του B.
11	Index by accumulator B SUB A 25B	Ο αναφερόμενος αριθμός (25) προστιθέμενος στο περιεχόμενο του B δίδει τη διεύθυνση μνήμης το περιεχόμενο της οποίας θα αφαιρεθεί από τον συσσωρευτή A.

(v) Διεύθυνση μνήμης

Τέλος, τα **τελευταία 9 bits** (Address) δίδουν **τη δυαδική διεύθυνση** της μνήμης, της ισοδύναμης με τον αριθμό που αναγράφεται και πρέπει να είναι μεταξύ του 0 και 511 (για τον Miracle).

δ. Παραδείγματα χρήσης της συμβολικής γλώσσας

Η εσωτερική μορφή της εντολής **ADD A 49**, π.χ., όπως αποθηκεύεται στη μνήμη είναι: **0010 0 01 000110001**, ενώ η εσωτερική μορφή: **1000001000001111** εντολής παριστά:

$$1000 \equiv BNZ,\ 0 \equiv ACCA,\ 01 \equiv \text{ευθεία πρόσβαση},\ 000001111 \equiv 15,$$

δηλαδή, την

BNZ A 15

Στη συνέχεια ακολουθούν συγκεκριμένα παραδείγματα **με στόχους**: **την εύρεση του Μ.Κ.Δ. δύο αριθμών**, **τον μηδενισμό ορισμένων θέσεων μνήμης**, **την ταξινόμηση αριθμών κατά τάξη μεγέθους** καθώς και δύο ασκήσεις για εξάσκηση.

(i) Εύρεση του Μ.Κ.Δ. δύο αριθμών

```
0   INA A       ↑ Διάβασε τον πρώτο αριθμό
1   STA A 50    ↑ Τοποθέτησέ τον στη θέση 50
2   INA A       ↑ Διάβασε τον δεύτερο αριθμό
3   STA A 51    ↑ Τοποθέτησέ τον στη θέση 51
4   LDA A 50    ↑ Βρες το A - B
5   SUB A 51
6   BZE A 13    ↑ Έλεγξε: if A=B πήγαινε στην 13
7   BMI A 10    ↑ Έλεγξε: if A<B πήγαινε στην 10
8   STA A 50    ↑ Τοποθέτησέ τον A-B στον A
9   JMP 4       ↑ Πήδα στην 4, για επανάληψη
10  NEG A       ↑ Τοποθέτησέ το B-A στον Acc A
11  STA A 51    ↑ Τοποθέτησέ το B-A στον B
12  JMP 4       ↑ Πήδα στην 4, για επανάληψη
13  LDA A 50    ↑ Βάλε στον Acc A το περιεχόμενο της 50
14  OUT A       ↑ Βγάλε έξω τον Μ.Κ.Δ.
15  HLT         ↑ Σταμάτα
```

Σχόλια:
Το προγραμμα βρίσκει τον Μ.Κ.Δ. των αριθμών Α και Β με τον αλγόριθμο του Ευκλείδη, και με τη βοήθεια των εξής δύο βημάτων:
1ο. Εάν οι δύο αριθμοί είναι ίσοι, τότε ο Μ.Κ.Δ. είναι η κοινή τους τιμή.
2ο. Άλλως, αφαίρεσε από τον μεγαλύτερο τον μικρότερο και επανέλαβε το βήμα 1.
Π.χ., εάν A=20 και B=8
20 8, 12 8, 4 8, 4 4
→ ο Μ.Κ.Δ.=**4**.

(ii) Μηδενισμός ορισμένων θέσεων μνήμης (από θέση 19 έως θέση 69)

```
0   LDA A #0     ↑ Μηδένισε τον Acc A
1   LDA B #50    ↑ Τοποθέτησέ στον Acc B το 50
2   STA A 19B    ↑ Μηδένισε την 19 + B
3   SUB B #1     ↑ Αφαίρεσε 1 από Acc B
4   BNZ B 2      ↑ Βρες την επόμενη θέση μηδενισμού
5   HLT          ↑ Σταμάτα
```

Σχόλιο:
Το προγραμμα μηδενίζει τις θέσεις μνήμης 19, 20, 21,...,69.

(iii) Ταξινόμηση αριθμών

Το παρακάτω πρόγραμμα διαβάζει ένα σύνολο από Ν αριθμούς (που ακολουθούν), τους ταξινομεί κατά τάξη μεγέθους και τέλος τους εκτυπώνει. Ο Ν αποθηκεύεται στη θέση 49, ενώ στην 48 υπάρχει ένα «flag»· τέλος, οι αριθμοί αποθηκεύονται από τη θέση 50 και πέρα.

```
0   INA A
1   STA A 49
2   BZE A 47
3   INA B
4   STA B 49A
5   SUB A #1
6   JMP 2
7   LDA A #0
8   STA A 48
9   LDA A #2
10  LDA B 49A
11  SUB B 48A
12  BPL B 20
13  STA B 48
14  LDA B 49A
15  STA B 47
16  LDA B 48A
17  STA B 49A
18  LDA B 47
19  STA B 48A
20  SUB A 49    ↑ SEE IF ARRAY EXHAUSTED
21  BZE A 25
22  ADD A 49
23  ADD A #1
24  JMP 10
25  LDA B 48
26  BNZ A 7
27  LDA A 49    ↑ OUTPUT RESULTS
28  LDA B 49A
29  OUT B
30  SUB A #1
31  BNZ A 28
32  HLT         ↑ STOP
```

Ασκήσεις:

1) Να γραφεί ένα πρόγραμμα Miracle, που να βρίσκει το άθροισμα:

$$1 + 2 + 3 + \cdots + 50.$$

2) Να γραφεί ένα πρόγραμμα Miracle, που να βρίσκει το 7! (Τόσο χωράει μια θέση μνήμης του !...$2^{16} - 1$).

3) Τι θα εκτυπωθεί με το ακόλουθο πρόγραμμα:

```
0 LDA B #3
1 ADD B #5
2 SUB B #2
3 OUT B
4 HLT
```

4) Ομοίως για το πρόγραμμα:

```
0 LDA A 35
1 LDA B #17
2 OUT A
3 OUT B
4 HLT
```

$17 + 63$

$35 + 39$.

5) Ομοίως για το πρόγραμμα:

```
0 LDA A 5
1 STA A 2
2 LDA A #3
3 OUT A
4 HLT
5 + 2100
```

6) Ποιο πρέπει να είναι το input του προγράμματος που ακολουθεί για να εκτυπωθεί το 102;

```
0 INA A
1 STA A 50
2 ADD A 50
3 OUT A
4 HLT
```

7) Να γραφεί ένα πρόγραμμα Miracle, που να διαβάζει 100 αριθμούς και να εκτυπώνει τον μεγαλύτερό τους.

Εργαστήριο 11ο (Ημερομηνία του Πάσχα):

Ο Gauss επινόησε **τον ακόλουθο αλγόριθμο** για τον προσδιορισμό της ημερομηνίας του Πάσχα(P) σε ημέρες Απριλίου (ή Μαΐου) για το οποιοδήποτε έτος (E):

$$P = D + F + 3,$$

όπου:

$$D = [19A + 16]_{30} \qquad \text{και} \qquad F = [2B + 4C + 6D]_7,$$

με:

$$A = [E]_{19}, \qquad B = [E]_{19} \qquad \text{και} \qquad C = [E]_7.$$

Να γραφεί ένα υποπρόγραμμα συνάρτηση:

`PASHA(E)`

που να υλοποιεί τον παρακάτω αλγόριθμο του Gauss. Στη συνέχεια να γραφεί ένα κύριο πρόγραμμα που να υπολογίζει τις ημερομηνίες του Πάσχα για τα έτη 1997 μέχρι και 2050 και να εκτυπώνει με κατάλληλη επικεφαλίδα και την διάκριση του μήνα (π.χ. Απρίλιος ή Μάιος).
(**Υπόδειξη:** Να γίνει χρήση της συνάρτησης `MOD(I,J)`.)

Εργαστήριο 12ο:(Αριθμοί πρώτοι πρώτοι προς αλλήλους σε συμβολική γλώσσα)

Με τη βοήθεια του MIRACLE (βλέπε 5η εβδομάδα) που θα υποθέσουμε ότι έχει εφοδιασθεί με 2 επιπλέον εντολές τις "**MULA A N1**" και "**DIVA A N2**" για τον πολλαπλασιασμό/διαίρεση του περιεχομένου του συσσωρευτή A διά του περιεχομένου της μνήμης στη διεύθυνση N1, N2 και τοποθέτηση του γινομένου/πηλίκου στον συσσωρευτή, να γραφεί ένα πρόγραμμα στη γλώσσα (συμβολική) του MIRACLE που να διαβάζει 2 φυσικούς αριθμούς, και να ελέγχει εάν οι αριθμοί είναι πρώτος προς αλλήλους ή όχι. Τέλος, να εκτυπώνει τους 2 αυτούς αριθμούς μαζί με την ένδειξη 1 εάν είναι πρώτοι προς αλλήλους, ή την ένδειξη 0 εάν είναι σύνθετοι.

Προεραιτική Άσκηση:
(α) Να δημιουργήσετε **τον αλγόριθμο σχηματισμού του γινομένου** 2 πολυωνύμων βαθμών M και N.
(β) Όμοια, για τον σχηματισμό **του γινομένου 2 πινάκων τετραγωνικών**, τάξεων K.