ΕΙΣΑΓΩΓΗ ΣΤΗΝ ΕΠΙΣΤΗΜΗ ΤΩΝ ΥΠΟΛΟΓΙΣΤΩΝ

http://eclass.aueb.gr/courses/INF511/

Χειρισμός Δεδομένων (ΚΕΦΑΛΑΙΟ 2)

Αλκμήνη Σγουρίτσα

Κοδριγκτώνος 12, 2ος όροφος

E-mail: alkmini@aueb.gr

ΚΕΦΑΛΑΙΟ 2: Χειρισμός Δεδομένων

- Εισαγωγή στην Αρχιτεκτονική Υπολογιστών
  - Κεντρική Μονάδα Επεξεργασίας, Κυρίως Μνήμη, Κρυφή Μνήμη
- Είδη εντολών στη Γλώσσα Μηχανής
- Παράδειγμα: μια απλή Γλώσσα Μηχανής
  - Είδη και δομή εντολών
- Εκτέλεση προγράμματος

Αρχιτεκτονική Υπολογιστών

- Κεντρική μονάδα επεξεργασίας (CPU) ή επεξεργαστής: πακεταρισμένη σε μικρά επίπεδα τετράγωνα chip (5cm x5cm)

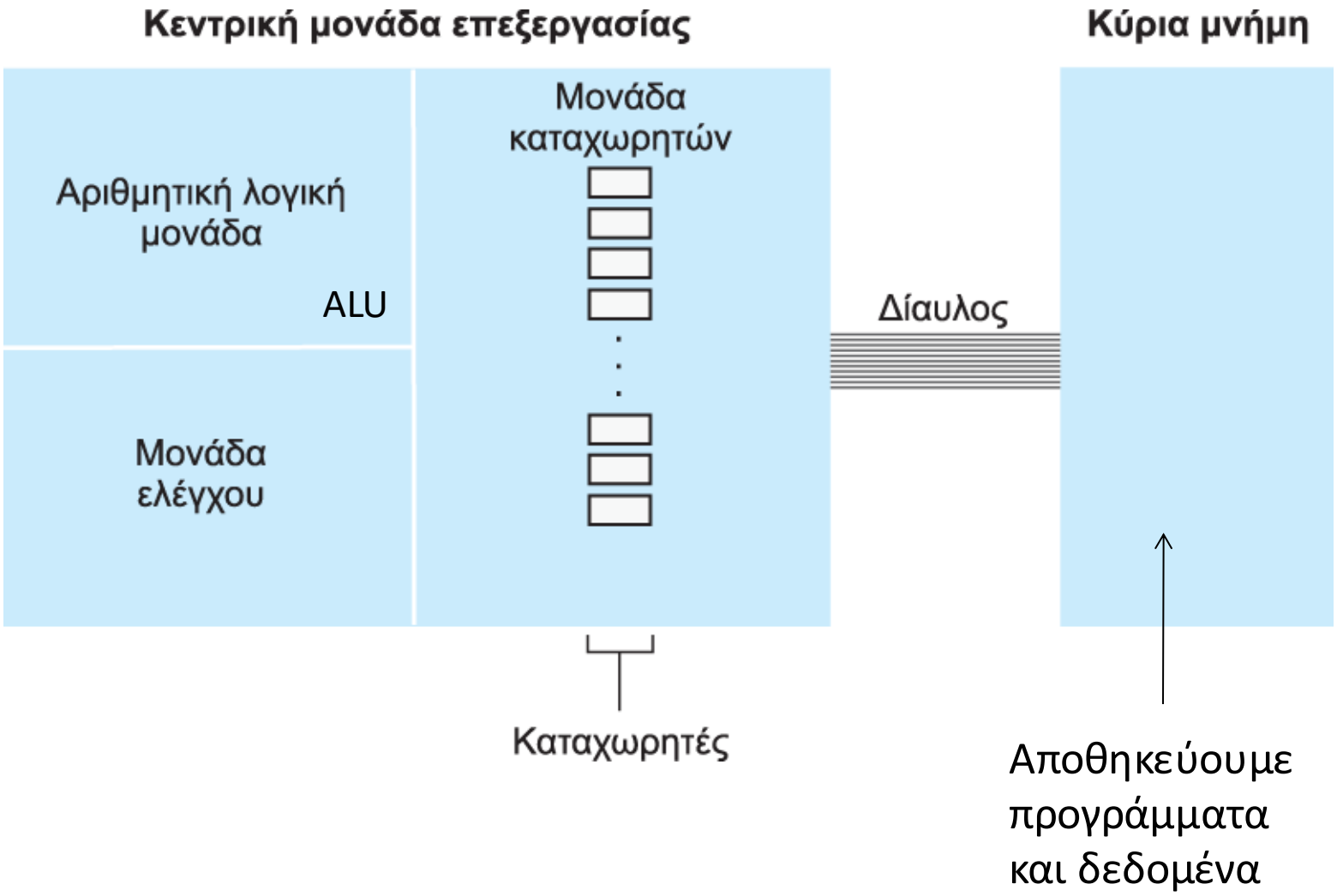
- Η **Κεντρική μονάδα επεξεργασίας** περιέχει:

- 1. Αριθμητική Λογική Μονάδα (Arithmetic Logical Unit, ALU): κυκλώματα για εκτέλεση πράξεων (πρόσθεση, αφαίρεση κλπ)

- 2. Μονάδα Ελέγχου (Control Unit): ελέγχει/συντονίζει τις ενέργειες της CPU

- 3. Καταχωρητές (Registers): για προσωρινή αποθήκευση δεδομένων στην CPU
  - Καταχωρητές Γενικής Χρήσης (General-purpose) και Ειδικής Χρήσης (Special-purpose)

Αρχιτεκτονική Υπολογιστών



Αρχιτεκτονική Υπολογιστών

- **Τα δεδομένα και τα προγράμματα** βρίσκονται στην μνήμη

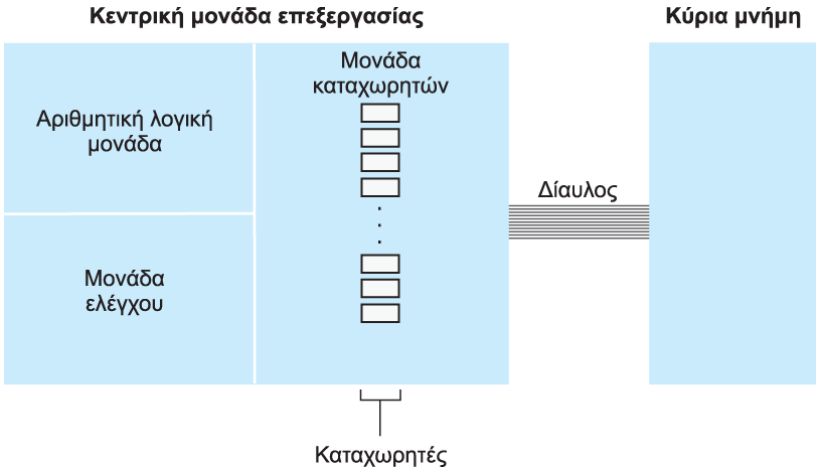
- Για να εκτελεστεί μια πράξη με δεδομένα αποθηκευμένα στην κύρια μνήμη, η Μονάδα Ελέγχου πρέπει να τα μεταφέρει από τη μνήμη στους Καταχωρητές Γενικής Χρήσης

- **Δίαυλος** (bus): συνδέει την CPU με την κύρια μνήμη.

- ALU: κάνει αριθμητικές πράξεις στα δεδομένα των καταχωρητών
  - Μεταφέρει το αποτέλεσμα σε έναν άλλον καταχωρητή Γενικής Χρήσης

Παράδειγμα: Πρόσθεση τιμών που είναι αποθηκευμένες στη μνήμη

Υλοποίηση της εντολής υψηλού επιπέδου x = y + z



Βήμα 1. Πάρε από τη μνήμη την πρώτη από τις τιμές που πρόκειται να προστεθούν και τοποθέτησέ τη σε έναν καταχωρητή.

Βήμα 2. Πάρε από τη μνήμη την άλλη τιμή και τοποθέτησέ τη σε έναν άλλον καταχωρητή.

Βήμα 3. Ενεργοποίησε το κύκλωμα της πρόσθεσης με τους καταχωρητές που χρησιμοποιήθηκαν στα Βήματα 1 και 2 ως εισόδους και έναν τρίτο καταχωρητή για αποθήκευση του αποτελέσματος.

Βήμα 4. Αποθήκευσε το αποτέλεσμα στη μνήμη.

Βήμα 5. Τέλος.

3 είδη αποθηκευτικού χώρου

- Κρυφή Μνήμη (cache memory)
- Κύρια Μνήμη (main memory)
- Σκληρός δίσκος (HD)

- Χρόνος ανάκτησης δεδομένων: HD > main memory > cache

- Σε μέγεθος: cache < main memory < HD

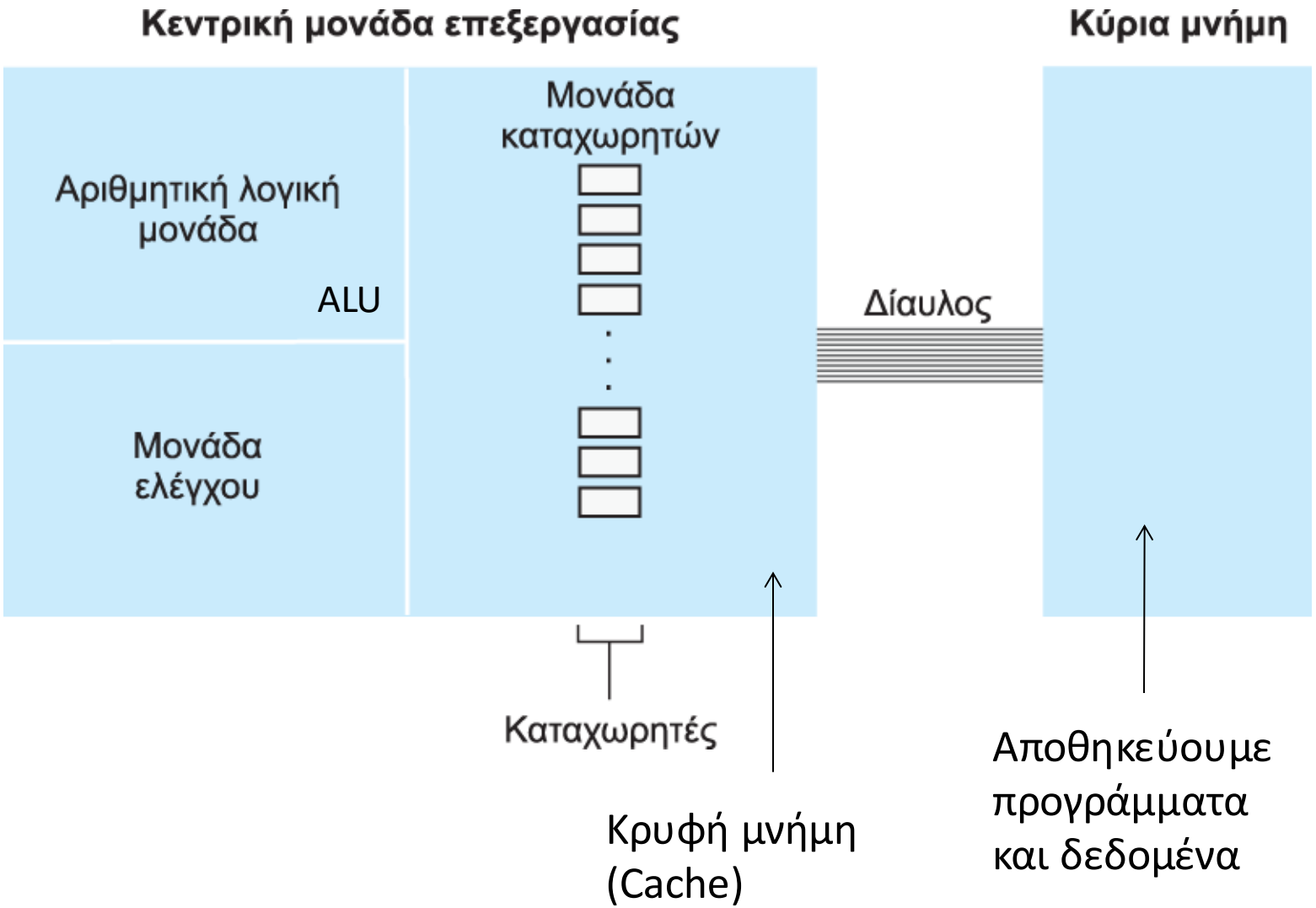
Κρυφή μνήμη (Cache memory)

- «Κρυφή» ή λανθάνουσα μνήμη (cache memory): μικρή μνήμη που βρίσκεται μέσα στην CPU
- Μέγεθος cache: κυμαίνεται από δεκάδες ή εκατοντάδες KB σε δεκάδες MB
- Cache: πολύ υψηλής ταχύτητας (στην πρόσβαση)

- Ο υπολογιστής κρατάει στην cache ένα **αντίγραφο ενός πολύ μικρού τμήματος της κύριας μνήμης** (εντολές και δεδομένα)
  - Αυτές τις εντολές/δεδομένα που χρησιμοποιούνται πιο συχνά από την CPU

- Αντί να γίνεται επικοινωνία Καταχωρητών - Κύριας Μνήμης, γίνεται επικοινωνία Καταχωρητών - Cache
- Αποτέλεσμα: πολύ πιο γρήγορη εκτέλεση εντολών

Αρχιτεκτονική Υπολογιστών



Αρχιτεκτονική Υπολογιστών

- Το **τι αποθηκεύουμε στην cache** είναι σημαντικό ώστε η ΚΜΕ να βρίσκει αυτά που ζητάει εκεί με μεγάλη πιθανότητα (δηλ. σε μεγάλο ποσοστό των αναζητήσεων)

- Αυτή η πιθανότητα λέγεται **λόγος ευστοχίας cache (cache hit ratio)**
- Cache hit ratio = (Αριθμός αιτημάτων ΚΜΕ προς την cache που βρίσκουν αυτό που ζητάνε) / (Συνολικός αριθμός αιτημάτων ΚΜΕ)

- Παράδειγμα: Υποθέστε ότι η CPU ζητά να φέρει από τη μνήμη 200 εντολές προγράμματος και 300 αριθμούς για αριθμητικές πράξεις
- Πρώτα τα αναζητά ένα ένα στην cache
- Αν βρει 150 εντολές και 250 αριθμούς στην cache, έχουμε cache hit ratio 400/500 = 80%
- Ό,τι δεν βρίσκει στην cache, το ζητάει και το λαμβάνει από την Κύρια μνήμη (…αλλά με μεγαλύτερη καθυστέρηση!)

Έννοια του αποθηκευμένου προγράμματος

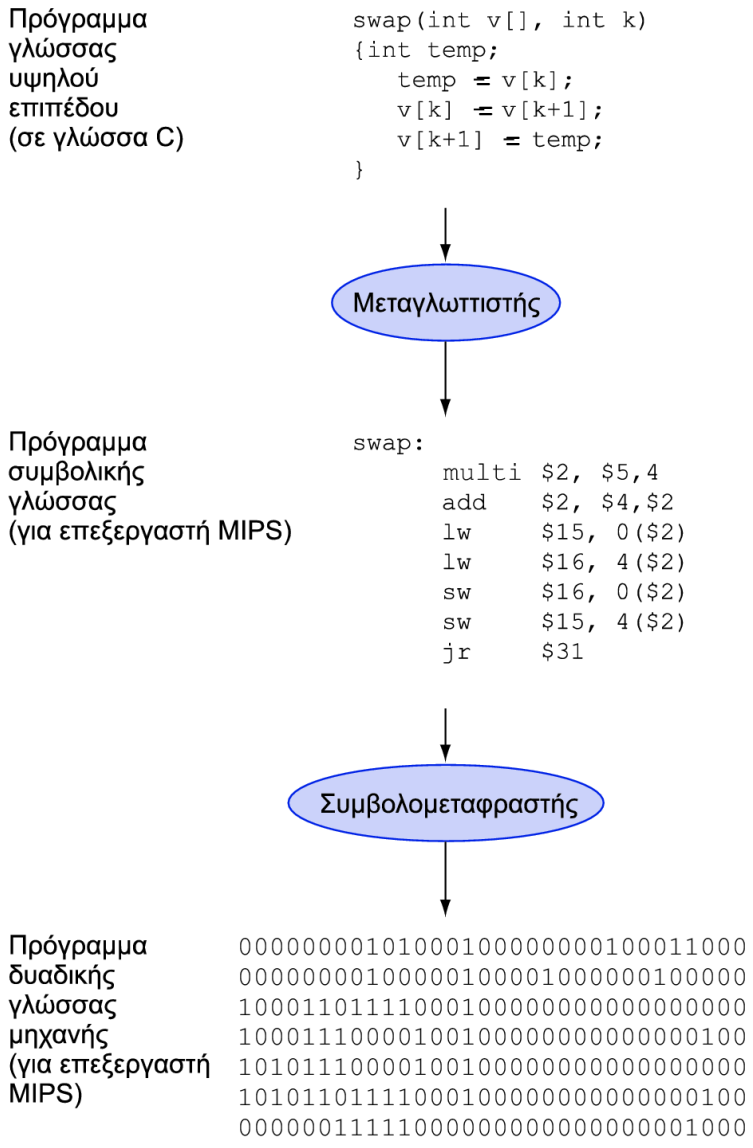
- Πώς γνωρίζει η ΚΜΕ τι εντολές/βήματα θα εκτελέσει;
- Οι πρώτοι υπολογιστές: το πρόγραμμα ήταν μέρος της μηχανής – τα βήματα ήταν ενσωματωμένα στη μονάδα ελέγχου → χαμηλή ευελιξία
- **Έννοια του αποθηκευμένου προγράμματος (Stored-program concept)**
  - Το πρόγραμμα αποτελείται από εντολές
  - Οι εντολές είναι και αυτές δεδομένα
  - Το πρόγραμμα αποθηκεύεται στην κύρια μνήμη
  - … μαζί με τα δεδομένα (π.χ. αριθμούς) που χρησιμοποιεί
  - Η CPU ανακτά σειριακά μια-μια τις εντολές του προγράμματος από τη μνήμη, τις αποκωδικοποιεί (καταλαβαίνει) και τις εκτελεί
- Τρομερή βελτίωση των υπολογιστών!

Από τη Γλώσσα Υψηλού Επιπέδου στη Γλώσσα Μηχανής

- Γλώσσα υψηλού επιπέδου
  - Κοντά στην ανθρώπινη λογική
  - Πρόγραμμα εύκολο στη γραφή και την ανάγνωση
  - Ανεξάρτητη του υπολογιστή στον οποίο αναπτύσσεται
- Συμβολική γλώσσα (assembly language)
  - Αναπαράσταση εντολών μηχανής με κείμενο
- Γλώσσα μηχανής
  - Κωδικοποιημένες εντολές σε σειρές από bit

Δείτε αυτό το βίντεο

https://www.youtube.com/watch?v=1OukpDfsuXE



Γλώσσα μηχανής: ορισμοί

- **Εντολή μηχανής:** εντολή κωδικοποιημένη ως ακολουθία bit (που αναγνωρίζεται από τη CPU)

- **Γλώσσα μηχανής:** το σύνολο όλων των εντολών, μαζί με ένα σύστημα κωδικοποίησης που κάνει τις εντολές αναγνωρίσιμες από τον υπολογιστή

Ομάδες εντολών γλώσσας μηχανής (1)

- Ομάδα μεταφοράς δεδομένων (data transfer): μεταφέρει δεδομένα μεταξύ CPU και κύριας μνήμης.
  - LOAD: αντιγράφει δεδομένα από την μνήμη σε έναν καταχωρητή γενικής χρήσης
  - STORE: αντιγράφει δεδομένα από έναν καταχωρητή γενικής χρήσης στη μνήμη
  - Εντολές I/O: για επικοινωνία με μέρη του υπολογιστή εκτός CPU και κύριας μνήμης (π.χ. σκληρός δίσκος, οθόνη)

Ομάδες εντολών γλώσσας μηχανής (2)

- Αριθμητική και λογική ομάδα (arithmetic / logical): για αριθμητικές πράξεις
  - Αριθμητικές πράξεις (+ , - , x , /)
  - Λογικές πράξεις: AND, OR, XOR
  - SHIFT / ROTATE: Ολίσθηση περιεχομένου καταχωρητή (επόμενη διαφάνεια)

- Ομάδα εντολών (control): κατευθύνει την εκτέλεση του προγράμματος
  - Εντολή άλματος JUMP (BRANCH): μετάβαση σε κάποια θέση μνήμης για εκτέλεση της εντολής που είναι αποθηκευμένη εκεί
    - όπως η εντολή goto στη Basic
    - Unconditional (χωρίς συνθήκη) JUMP
    - Υπό συνθήκη JUMP (conditional JUMP): άλμα, αν μια συνθήκη ικανοποιείται

Παράδειγμα: Διαίρεση τιμών αποθηκευμένων στη μνήμη

Υλοποίηση της εντολής υψηλού επιπέδου x = y ÷ z

Βήμα 1. ΦΟΡΤΩΣΕ (LOAD) σε έναν καταχωρητή την μία τιμή από τη μνήμη.

Βήμα 2. ΦΟΡΤΩΣΕ (LOAD) σε έναν διαφορετικό καταχωρητή την άλλη τιμή από τη μνήμη.

Βήμα 3. Αν η δεύτερη τιμή είναι μηδέν, ΠΗΓΑΙΝΕ (JUMP) στο Βήμα 6.

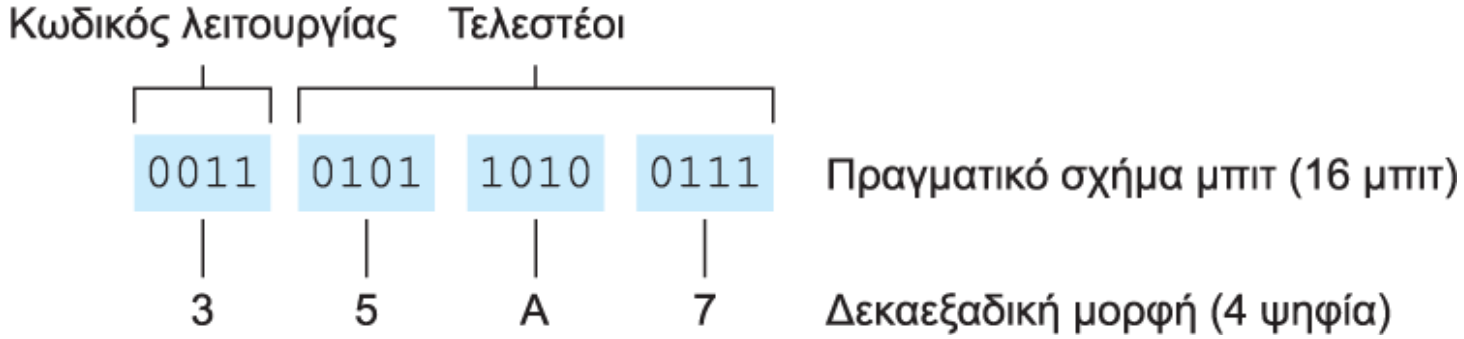
Βήμα 4. Διαίρεσε το περιεχόμενο του πρώτου καταχωρητή με αυτό του δεύτερου καταχωρητή και τοποθέτησε το αποτέλεσμα σε έναν τρίτο καταχωρητή.

Βήμα 5. ΑΠΟΘΗΚΕΥΣΕ (STORE) τα περιεχόμενα του τρίτου καταχωρητή στη μνήμη.

Βήμα 6. ΤΕΛΟΣ (STOP).

Ποια βήματα ανήκουν στην
- Ομάδα μεταφοράς δεδομένων: Βήματα 1,2,5
- Αριθμητική και λογική ομάδα: Βήμα 4
- Ομάδα ελέγχου: Βήματα 3,6

Τα μέρη μιας εντολής μηχανής



- Αποτελείται από 16 bit (2 bytes)
- **Πεδίο κωδικού λειτουργίας (op-code):** ακολουθία bit που προσδιορίζει ποια λειτουργία κάνει η εντολή
  - Π.χ. LOAD, STORE, JUMP, XOR, ADD,… (μια λειτουργία ανά εντολή)
- **Πεδίο τελεστέων (operand):** ακολουθία bit που παρέχει λεπτομέρειες σχετικά με την εντολή
  - Το πλήθος τελεστέων διαφέρει ανάλογα με το κωδικό λειτουργίας
  - Π.χ. **Στην LOAD**, τελεστές είναι: η θέση μνήμης από όπου θα πάρουμε τα δεδομένα και ο καταχωρητής όπου θα τα αντιγράψουμε
  - **Στην ADD**, τελεστές είναι: οι καταχωρητές των 2 όρων του αθροίσματος και ο καταχωρητής όπου θα καταγραφεί το άθροισμα

Επανάληψη: Δεκαεξαδικός συμβολισμός

Σχήμα μπιτ	Δεκαεξαδική αναπαράσταση
0000	0
0001	1
0010	2
0011	3
0100	4
0101	5
0110	6
0111	7
1000	8
1001	9
1010	A
1011	B
1100	C
1101	D
1110	E
1111	F

Παράρτημα Γ: Μια Απλή Γλώσσα Μηχανής (1)

Κωδ. Λειτ.	Τελεστέος	Περιγραφή
1	**R X Y**	Φορτώνει (LOAD) τον καταχωρητή R με την ακολουθία bit που υπάρχει στο κελί μνήμης με διεύθυνση XY.
2	**R X Y**	Φορτώνει (LOAD) τον καταχωρητή R με την ακολουθία bit XY.
3	**R X Y**	Αποθηκεύει (STORE) την ακολουθία bit που περιέχει ο καταχωρητής R στο κελί μνήμης με διεύθυνση XY.
4	**0 R S**	Μετακινεί (MOVE) την ακολουθία bit που περιέχει ο καταχωρητής R στον καταχωρητή S.

Παράρτημα Γ: Μια Απλή Γλώσσα Μηχανής (2)

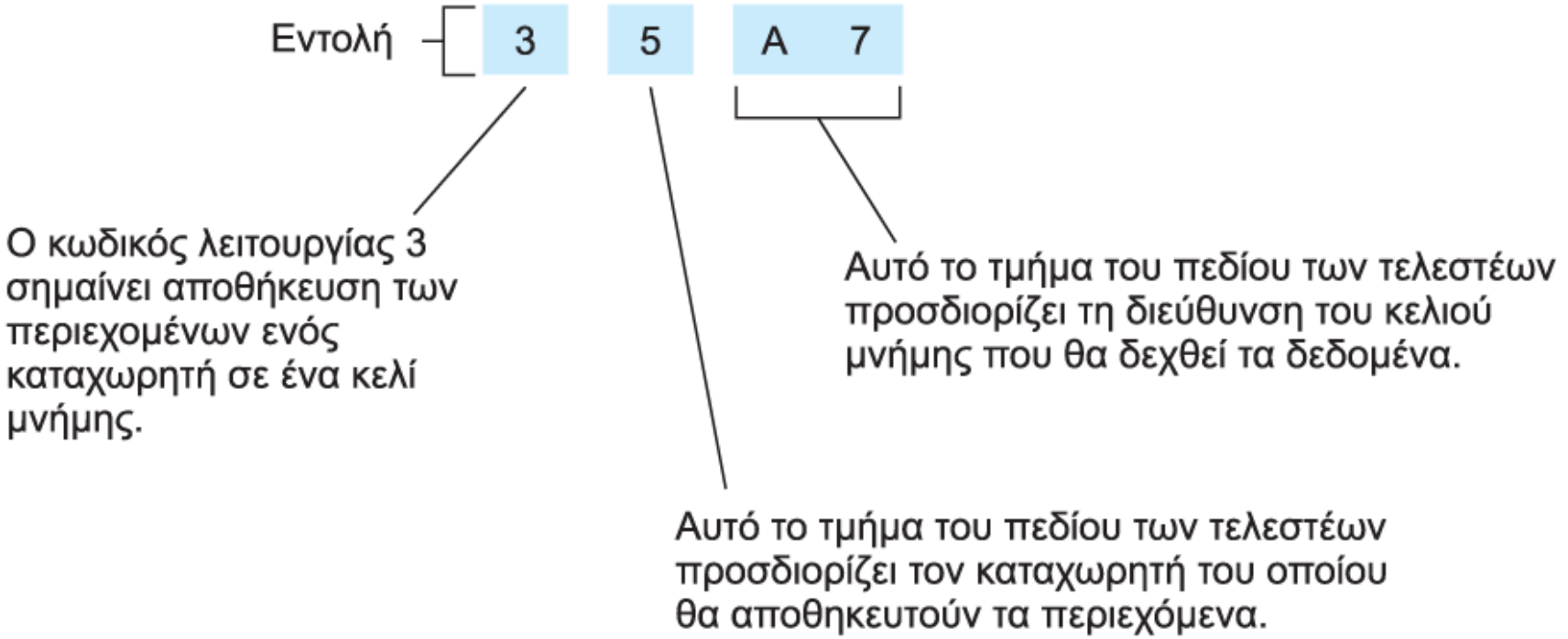
Κωδ Λειτ.	Τελεστέος	Περιγραφή
5	**R S T**	Προσθέτει (ADD) τις ακολουθίες bit των καταχωρητών S και T (σε μορφή συμπληρώματος ως προς δύο), και τοποθετεί το αποτέλεσμα στον καταχωρητή R.
6	**R S T**	Προσθέτει (ADD) τις ακολουθίες bit των καταχωρητών S και T (σε μορφή κινητής υποδιαστολής), και τοποθετεί το αποτέλεσμα στον καταχωρητή R.
7	**R S T**	Εκτελεί την πράξη OR στις ακολουθίες bit των καταχωρητών S και T και τοποθετεί το αποτέλεσμα στον καταχωρητή R.
8	**R S T**	Εκτελεί την πράξη AND στις ακολουθίες bit των καταχωρητών S και T και τοποθετεί το αποτέλεσμα στον καταχωρητή R.

Παράρτημα Γ: Μια Απλή Γλώσσα Μηχανής (3)

Κωδ Λειτ.	Τελεστέος	Περιγραφή
9	**R S T**	Εκτελεί την πράξη XOR στις ακολουθίες bit των καταχωρητών S και T και τοποθετεί το αποτέλεσμα στον καταχωρητή R.
A	**R 0 X**	Περιστρέφει (rotate) την ακολουθία bit του καταχωρητή R ένα bit προς τα δεξιά X φορές. Το bit του άκρου χαμηλής τάξης (LSB) τοποθετείται στο άκρο υψηλής τάξης (MSB) (κυκλική ολίσθηση)
B	**R X Y**	Μεταπηδά (JUMP) στην εντολή που βρίσκεται στο κελί μνήμης με διεύθυνση XY, **αν** η ακολουθία bit του R είναι ίδια με την ακολουθία bit του καταχωρητή 0. Αλλιώς συνεχίζεται η κανονική ροή της εκτέλεσης.
C	**0 0 0**	Τερματίζει (HALT) την εκτέλεση.

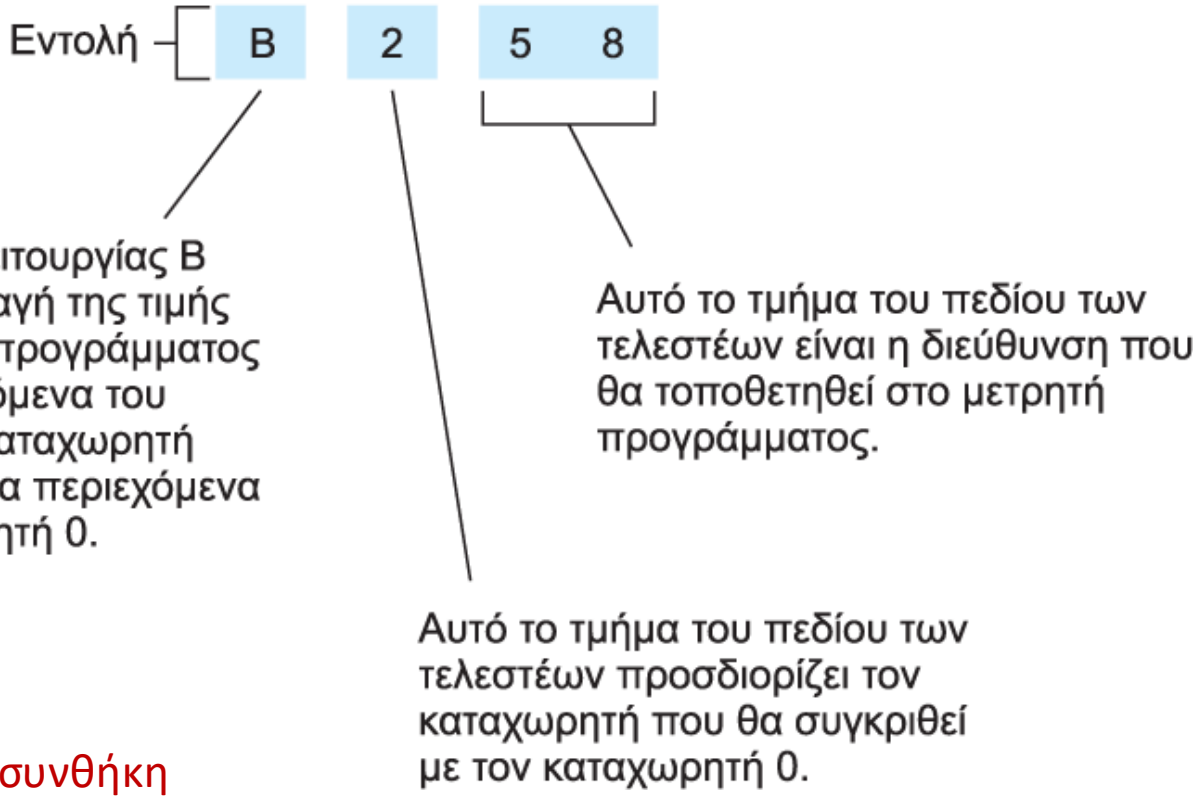
Αποκωδικοποίηση της εντολής 3 5 A 7

3	**R X Y**	Αποθηκεύει (STORE) την ακολουθία bit που περιέχει ο καταχωρητής R στο κελί μνήμης με διεύθυνση XY.



Αποκωδικοποίηση μιας εντολής JUMP

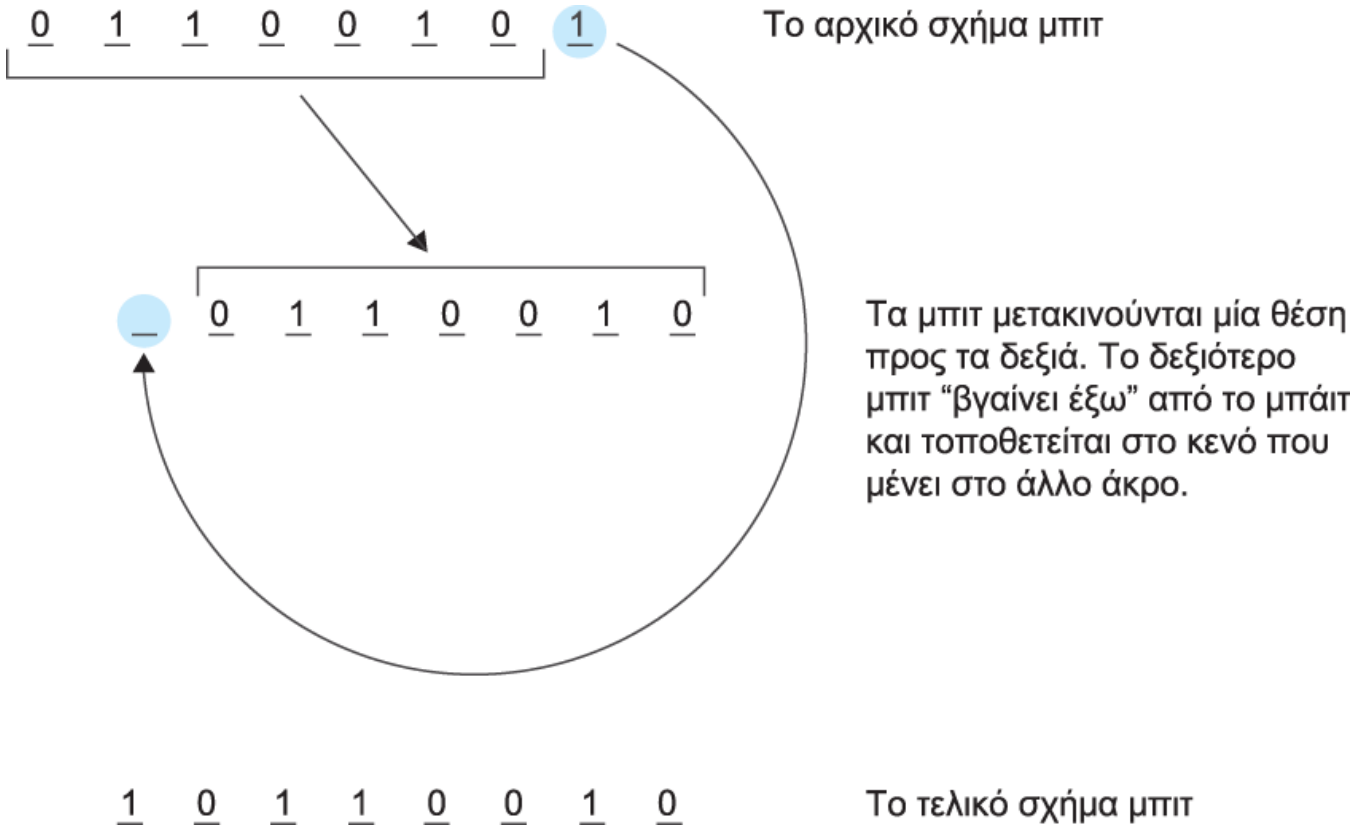
B R X Y Μεταπηδά (JUMP) στην εντολή που βρίσκεται στο κελί μνήμης με διεύθυνση XY, **αν** η ακολουθία bit του R είναι ίδια με την ακολουθία bit του καταχωρητή 0. Αλλιώς συνεχίζεται η κανονική ροή της εκτέλεσης.



- JUMP: άλμα υπό συνθήκη
- Αν [Κατ.0] ≠ [Κατ.2], η εντολή τερματίζεται, πηγαίνουμε στην επόμενη
- Αν [Κατ.0] = [Κατ.2], η τιμή 5 8 εγγράφεται στον Program Counter
- Τι σημαίνει η εντολή B 0 5 8 ? JUMP χωρίς συνθήκη (πάντα [Κατ.0] = [Κατ.0])

Παράδειγμα Κυκλικής περιστροφής προς τα δεξιά

A R 0 X	Περιστρέφει (rotate) την ακολουθία bit του καταχωρητή R ένα bit προς τα δεξιά X φορές. Το bit του άκρου χαμηλής τάξης (LSB) τοποθετείται στο άκρο υψηλής τάξης (MSB) (κυκλική ολίσθηση)



Παράδειγμα Εντολής AND

8 R S T	Εκτελεί την πράξη AND στις ακολουθίες bit των καταχωρητών S και T και τοποθετεί το αποτέλεσμα στον καταχωρητή R.

```
        10011100 (S)
AND     10101101 (T)
-------------------
        10001100 (R)
```

Χρησιμότητα του AND

- Από τα 8 bits ενός κελιού, πώς μπορούμε να απομονώσουμε το 2ο από το άκρο χαμηλής τάξης bit;

Εκτελούμε την πράξη AND με το 00000010

```
        10011100
AND     00000010
-------------------
        00000000
```

```
        10011110
AND     00000010
-------------------
        00000010
```

- Αν το 2ο από το άκρο χαμηλής τάξης bit ήταν 1 και θέλουμε να το αλλάξουμε σε 0;

Εκτελούμε την πράξη AND με το 11111101

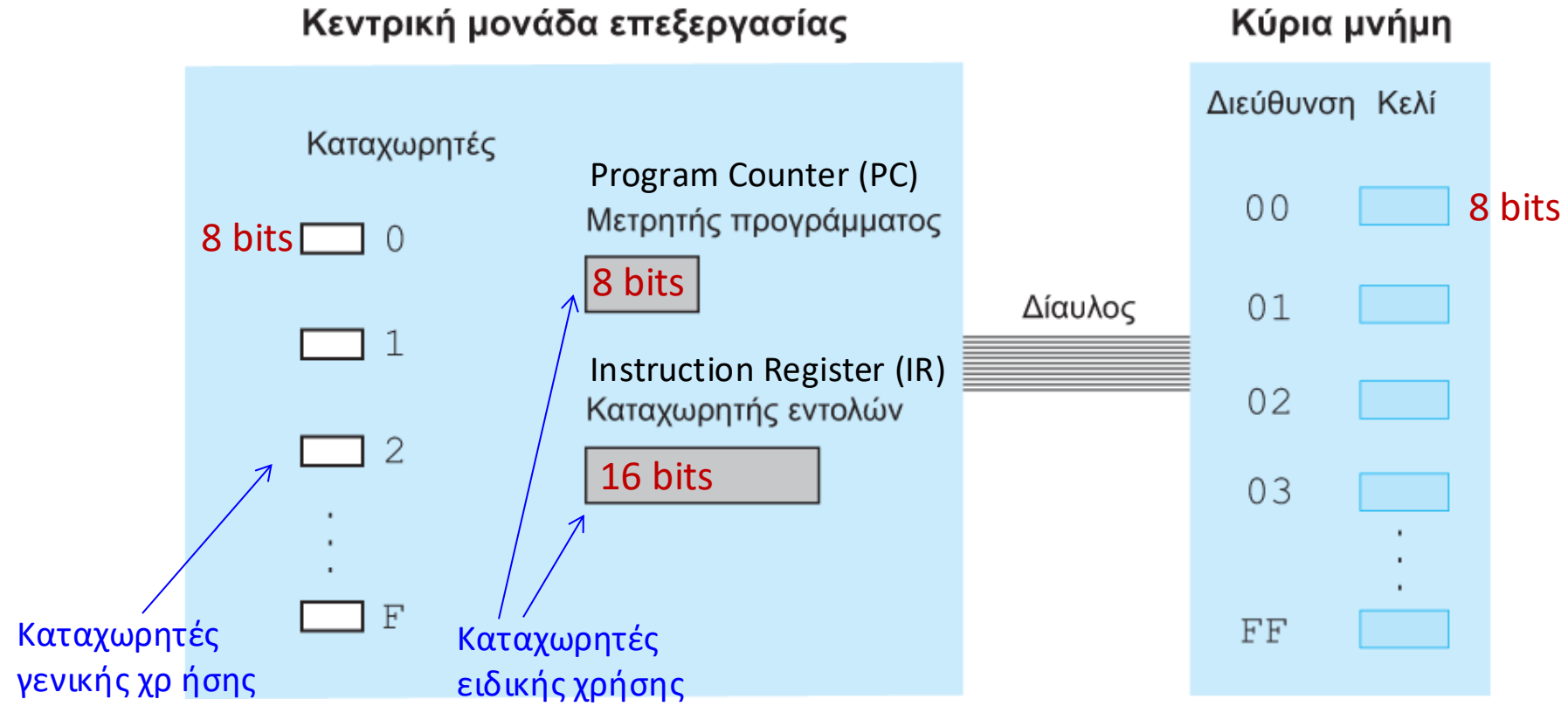
```
        10011110
AND     11111101
-------------------
        10011100
```

- Αν το 2ο από το άκρο χαμηλής τάξης bit ήταν 0 και θέλουμε να το αλλάξουμε σε 1;

Εκτελούμε την πράξη OR με το 00000010

Η Αρχιτεκτονική του Υπολογιστή

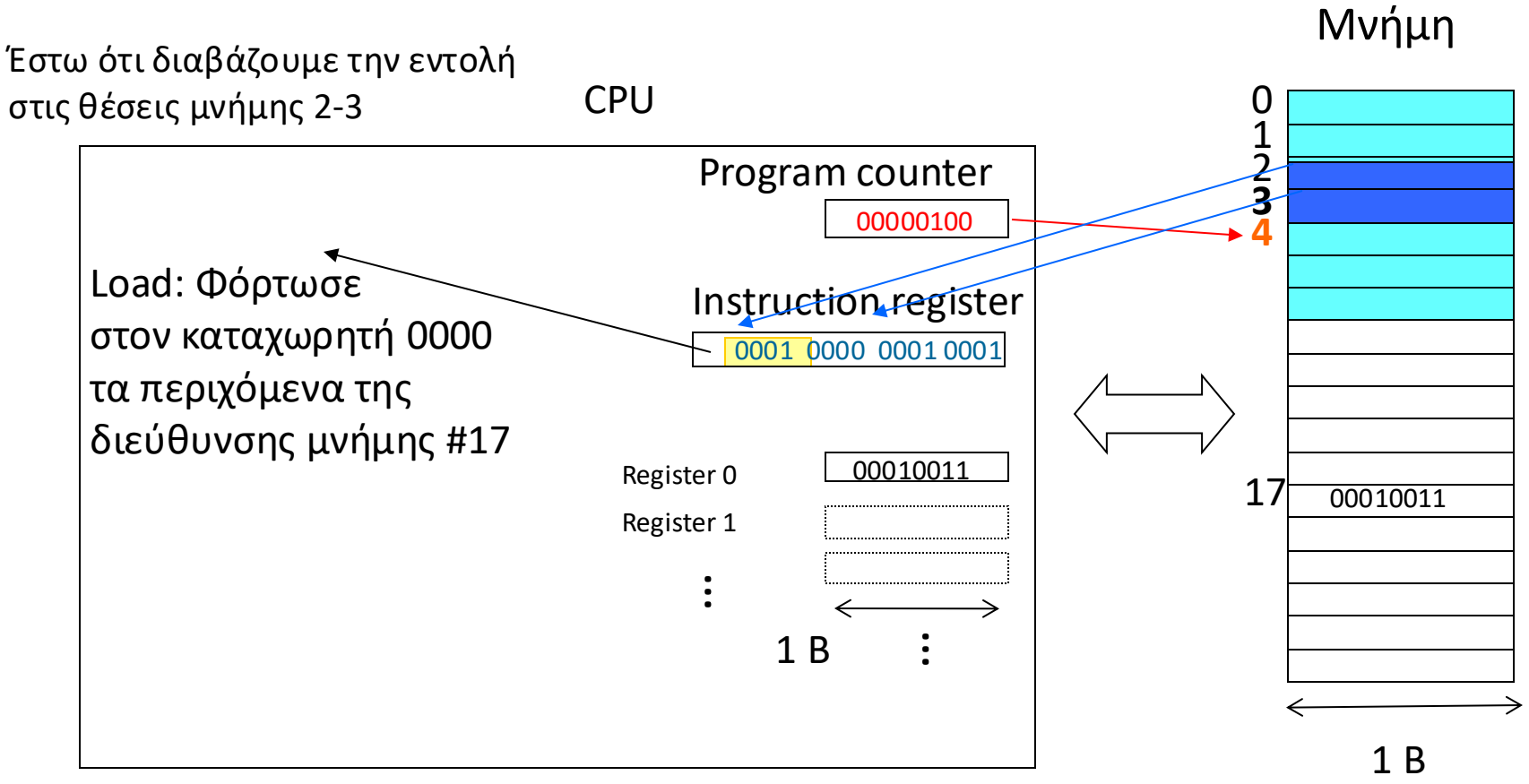
16 καταχωρητές γενικής χρήσης 0, ... 15 (4 bits για κάθε διεύθυνσης καταχωρητή)
256 κυψελίδες κύριας μνήμης: 0,...255 (8 bits για κάθε διεύθυνσης μνήμης)
8 bits (1 byte) δεδομένων ανά κυψελίδα



Μετρητής προγράμματος (Program Counter, PC) : περιέχει την διεύθυνση μνήμης όπου βρίσκεται η επόμενη εντολή προς εκτέλεση, από αυτήν που εκτελείται τώρα

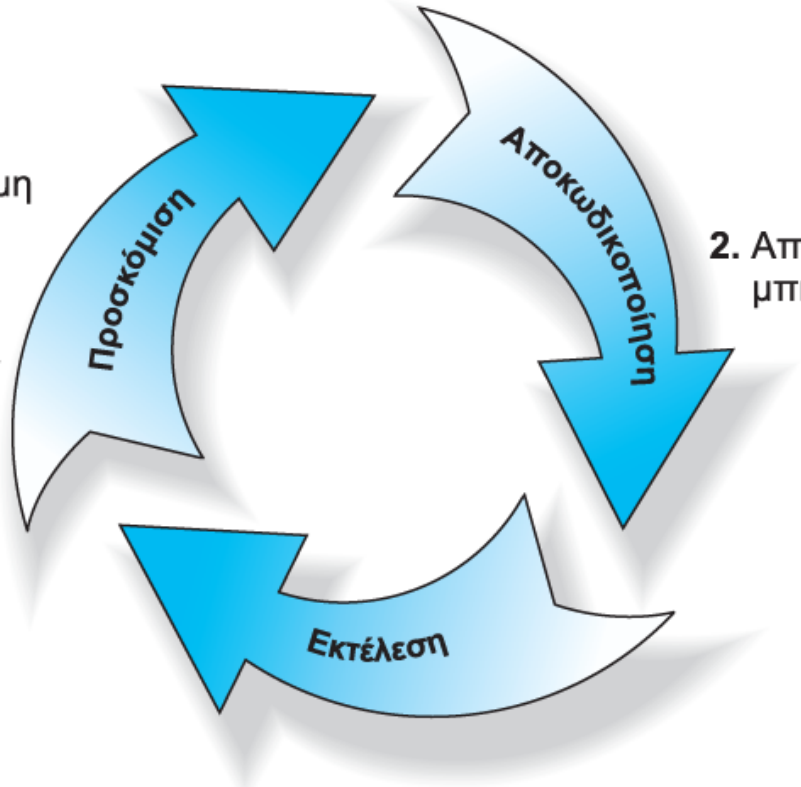
Καταχωρητής εντολών (Instruction Register, IR): δείχνει την τρέχουσα εντολή

Ένα στιγμιότυπο εκτέλεσης



Εντολές = 2 bytes
Κάθε εντολή αποθηκεύεται σε 2 διαδοχικές θέσεις μνήμης
δεδομένα = 1 byte

Ο κύκλος εκτέλεσης μιας εντολής



1. Πάρε την επόμενη εντολή από τη μνήμη (όπως ορίζεται από το μετρητή προγράμματος), και κατόπιν αύξησε τον καταχωρητή προγράμματος.

2. Αποκωδικοποίησε το σχήμα μπιτ στον καταχωρητή εντολών.

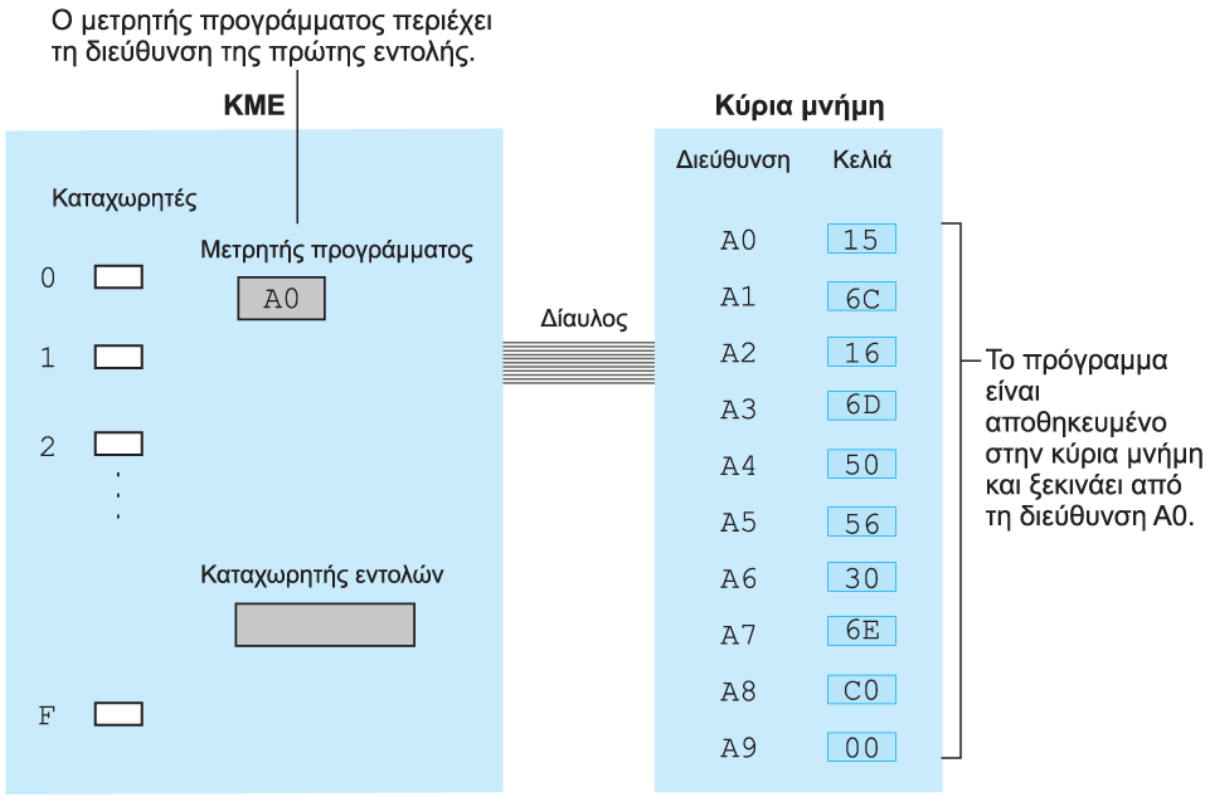
3. Εκτέλεσε την ενέργεια που ζητάει η εντολή στον καταχωρητή εντολών.

Λάθος στην μετάφραση:
→ «..κατόπιν αύξησε τον Μετρητή Προγράμματος»

Παράδειγμα προγράμματος πρόσθεσης σε γλώσσα μηχανής

	Κωδικοποιημένες Εντολές	Ερμηνεία
Βήμα 1. Πάρε από τη μνήμη μία από τις τιμές που πρόκειται να προστεθούν και τοποθέτησέ τη σε έναν καταχωρητή.	**1 5 6 C**	Φόρτωσε στον καταχωρητη 5 την ακολουθία bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6C
Βήμα 2. Πάρε από τη μνήμη την άλλη τιμή και τοποθέτησέ τη σε έναν άλλον καταχωρητή.	**1 6 6 D**	Φόρτωσε στον καταχωρητη 6 την ακολουθία bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6D
Βήμα 3. Ενεργοποίησε το κύκλωμα της πρόσθεσης με τους καταχωρητές που χρησιμοποιήθηκαν στα Βήματα 1 και 2 ως εισόδους και ένα διαφορετικό καταχωρητή για την αποθήκευση του αποτελέσματος.	**5 0 5 6**	Πρόσθεσε τα περιεχόμενα των καταχωρητών 5 και 6 (σε μορφή συμπληρώματος ως προς δύο), και αποθήκευσε το αποτέλεσμα στον καταχωρητή 0
Βήμα 4. Αποθήκευσε το αποτέλεσμα στη μνήμη.	**3 0 6 E**	Αποθήκευσε τα περιεχόμενα του καταχωρητή 0 στο κελί μνήμης με διεύθυνση 6E
Βήμα 5. Τέλος	**C 0 0 0**	Τέλος (HALT)

Παράδειγμα εκτέλεσης προγράμματος



156C Φόρτωσε στον καταχωρητη 5 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6C.

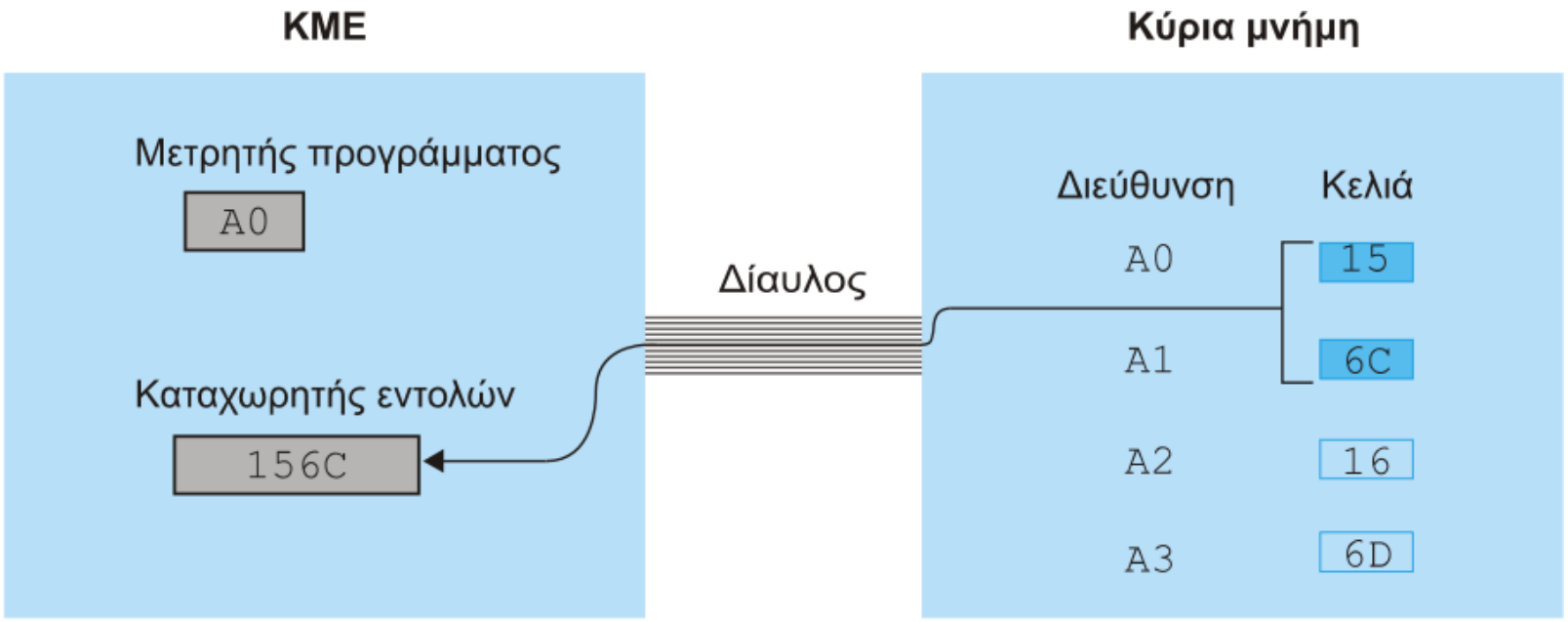
166D Φόρτωσε στον καταχωρητη 6 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6D.

5056 Πρόσθεσε τα περιεχόμενα των καταχωρητών 5 και 6 σε μορφή συμπληρώματος ως προς δύο, και αποθήκευσε το αποτέλεσμα στον καταχωρητή 0.

306E Αποθήκευσε τα περιεχόμενα του καταχωρητή 0 στο κελί μνήμης με διεύθυνση 6E.

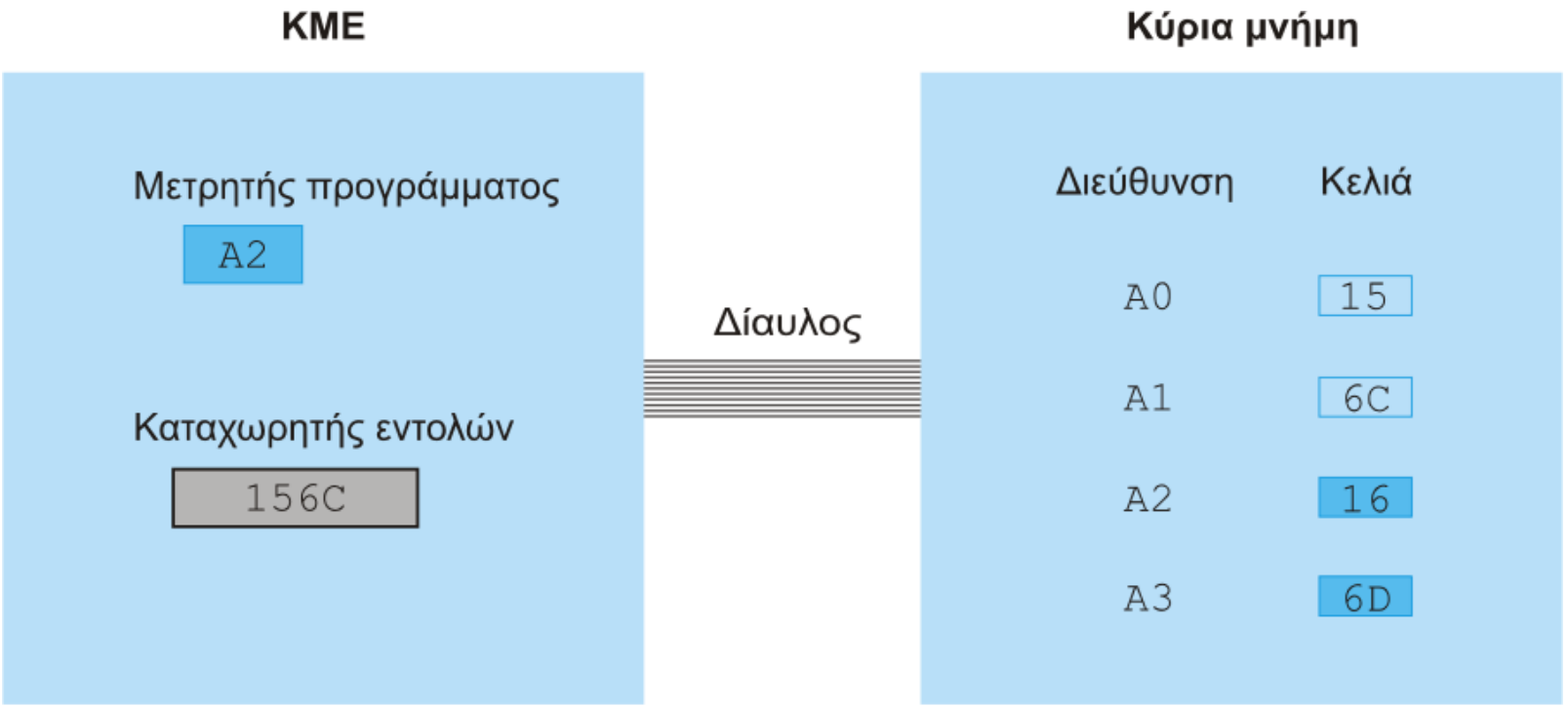
C000 Τέλος (HALT).

Βήμα προσκόμισης (Fetch) του κύκλου μηχανής (α)



α. Στην αρχή του βήματος προσκόμισης, προσκομίζεται από τη μνήμη η εντολή που ξεκινάει στη διεύθυνση A0 και τοποθετείται στον καταχωρητή εντολών.

Βήμα προσκόμισης (Fetch) του κύκλου μηχανής (β)

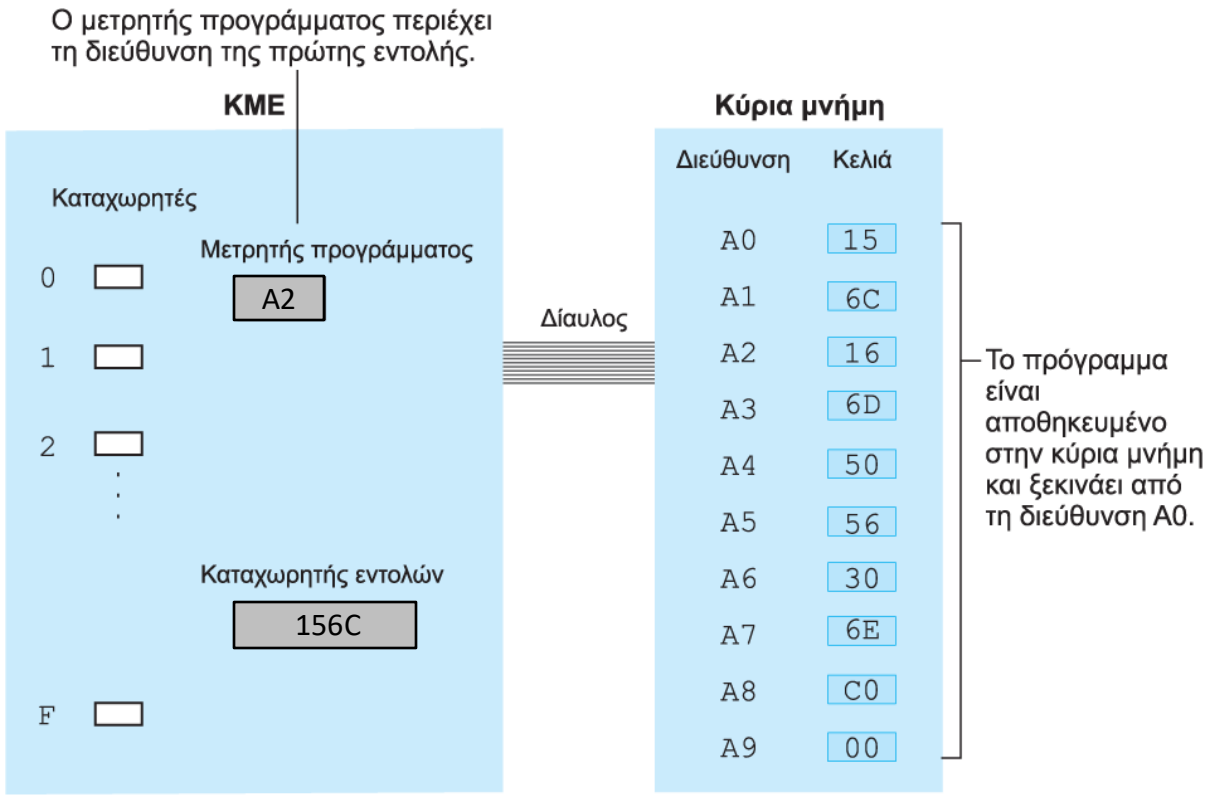


β. Κατόπιν ο μετρητής προγράμματος αυξάνεται έτσι ώστε να δείχνει στην επόμενη εντολή.

Βήμα αποκωδικοποίησης και εκτέλεσης του κύκλου μηχανής

- Κατόπιν, γίνεται ανάλυση της εντολής 1 5 6 C που βρίσκεται στον καταχωρητή εντολών
- LOAD register 5 με το περιεχόμενο της διεύθυνσης μνήμης 6 C
- Στέλνεται σήμα στην διεύθυνση μνήμης 6 C
- Η διεύθυνση 6 C στέλνει τα περιεχόμενά της και αυτά φορτώνονται στον καταχωρητή γενικής χρήσης 5
- Νέος κύκλος
  - CPU στέλνει σήμα στην διεύθυνση μνήμης A2 που υπάρχει στον Program Counter
  - Εντολή 1 6 6 D έρχεται στην CPU από την διεύθυνση μνήμης A2 και γράφεται στον Instruction counter
  - Program Counter = A4
  - Εκτελείται η εντολή 1 6 6 D
- Κ.ο.κ

Παράδειγμα εκτέλεσης προγράμματος



156C Φόρτωσε στον καταχωρητη 5 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6C.

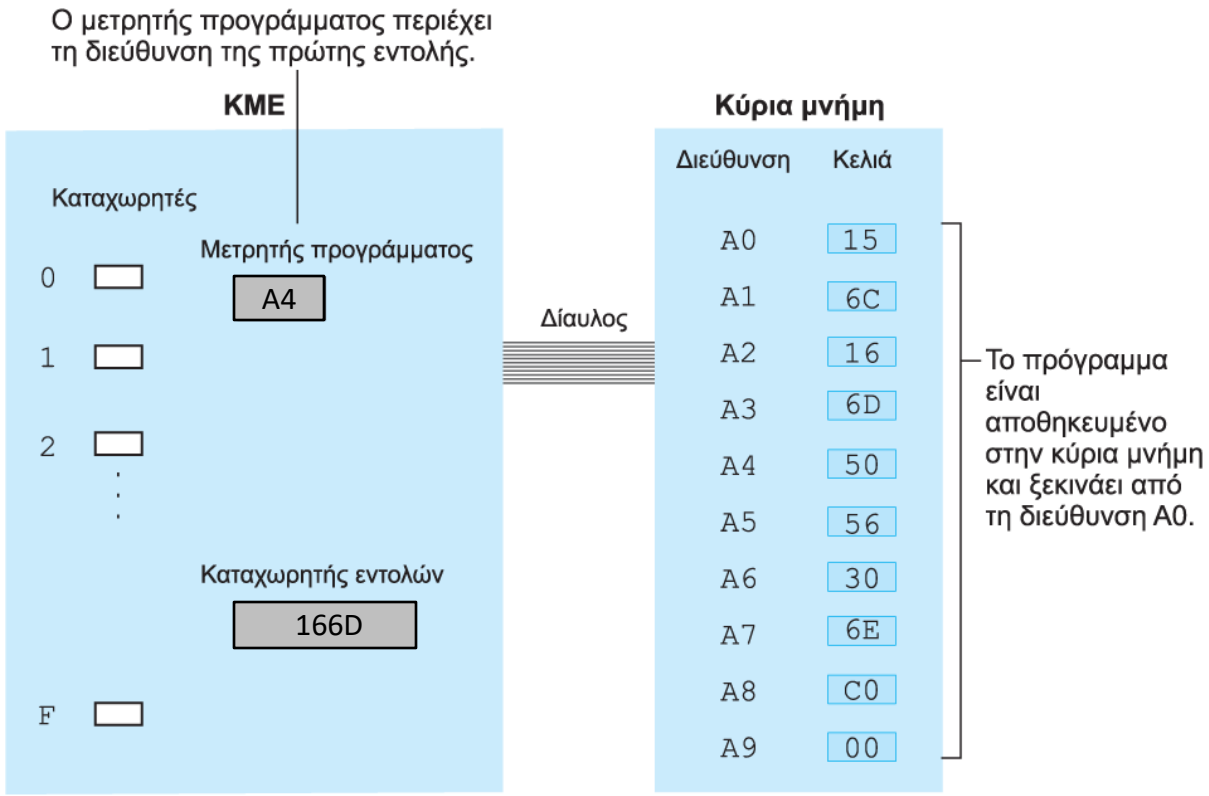
166D Φόρτωσε στον καταχωρητη 6 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6D.

5056 Πρόσθεσε τα περιεχόμενα των καταχωρητών 5 και 6 σε μορφή συμπληρώματος ως προς δύο, και αποθήκευσε το αποτέλεσμα στον καταχωρητή 0.

306E Αποθήκευσε τα περιεχόμενα του καταχωρητή 0 στο κελί μνήμης με διεύθυνση 6E.

C000 Τέλος (HALT).

Παράδειγμα εκτέλεσης προγράμματος



156C Φόρτωσε στον καταχωρητη 5 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6C.

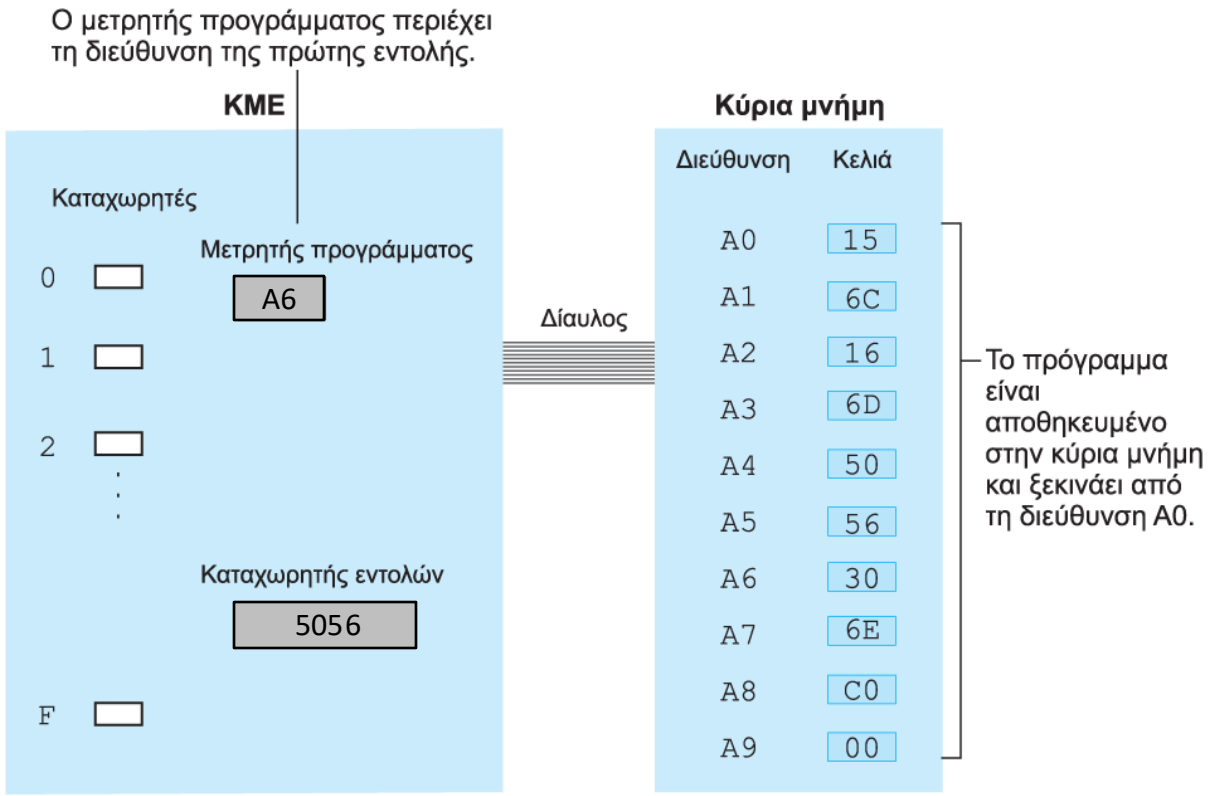
166D Φόρτωσε στον καταχωρητη 6 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6D.

5056 Πρόσθεσε τα περιεχόμενα των καταχωρητών 5 και 6 σε μορφή συμπληρώματος ως προς δύο, και αποθήκευσε το αποτέλεσμα στον καταχωρητή 0.

306E Αποθήκευσε τα περιεχόμενα του καταχωρητή 0 στο κελί μνήμης με διεύθυνση 6E.

C000 Τέλος (HALT).

Παράδειγμα εκτέλεσης προγράμματος



156C Φόρτωσε στον καταχωρητη 5 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6C.

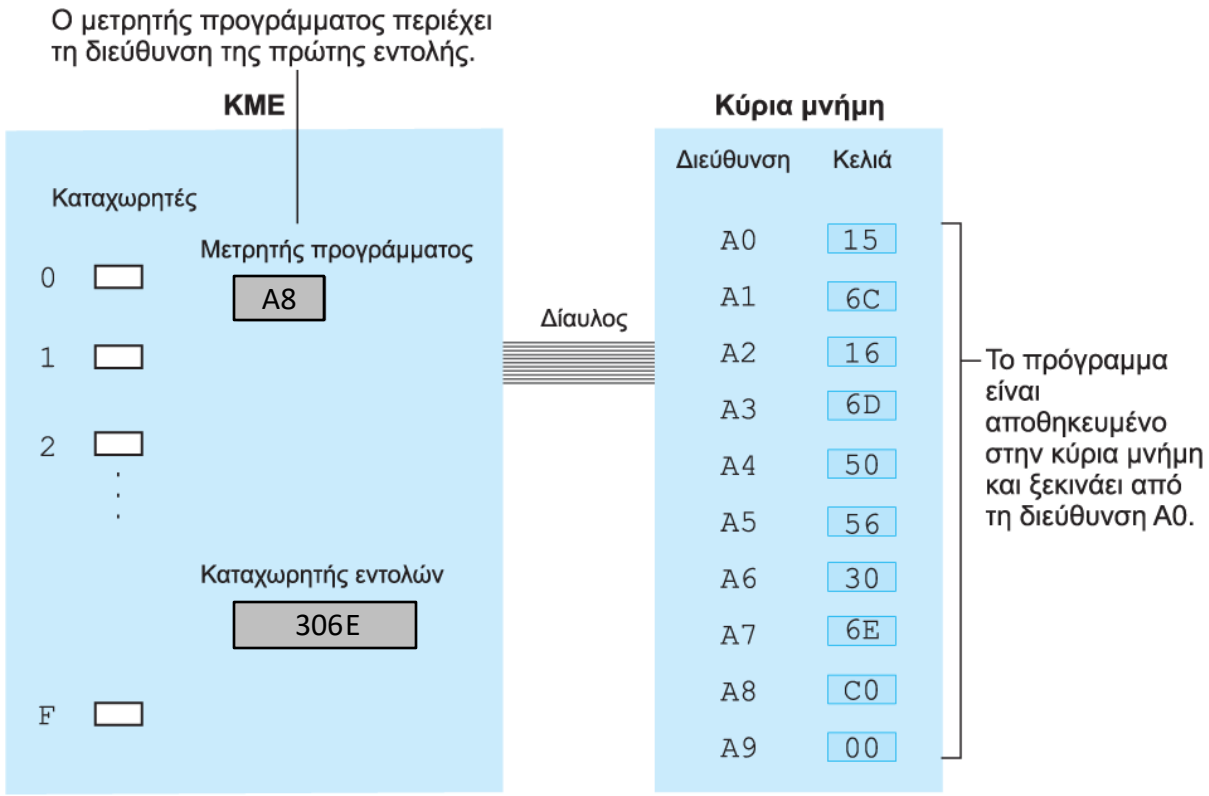
166D Φόρτωσε στον καταχωρητη 6 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6D.

5056 Πρόσθεσε τα περιεχόμενα των καταχωρητών 5 και 6 σε μορφή συμπληρώματος ως προς δύο, και αποθήκευσε το αποτέλεσμα στον καταχωρητή 0.

306E Αποθήκευσε τα περιεχόμενα του καταχωρητή 0 στο κελί μνήμης με διεύθυνση 6E.

C000 Τέλος (HALT).

Παράδειγμα εκτέλεσης προγράμματος



156C Φόρτωσε στον καταχωρητη 5 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6C.

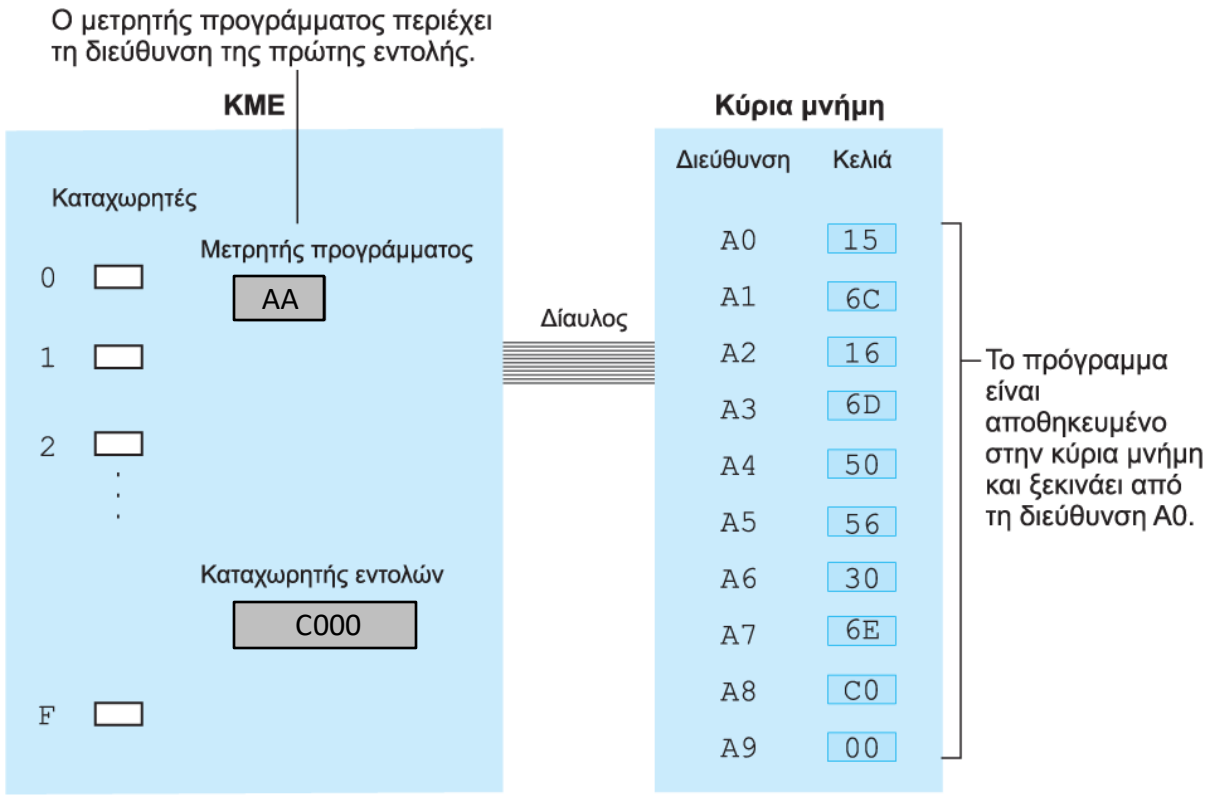
166D Φόρτωσε στον καταχωρητη 6 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6D.

5056 Πρόσθεσε τα περιεχόμενα των καταχωρητών 5 και 6 σε μορφή συμπληρώματος ως προς δύο, και αποθήκευσε το αποτέλεσμα στον καταχωρητή 0.

306E Αποθήκευσε τα περιεχόμενα του καταχωρητή 0 στο κελί μνήμης με διεύθυνση 6E.

C000 Τέλος (HALT).

Παράδειγμα εκτέλεσης προγράμματος



156C Φόρτωσε στον καταχωρητη 5 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6C.

166D Φόρτωσε στον καταχωρητη 6 το σχήμα bit που βρίσκεται στο κελί μνήμης με διεύθυνση 6D.

5056 Πρόσθεσε τα περιεχόμενα των καταχωρητών 5 και 6 σε μορφή συμπληρώματος ως προς δύο, και αποθήκευσε το αποτέλεσμα στον καταχωρητή 0.

306E Αποθήκευσε τα περιεχόμενα του καταχωρητή 0 στο κελί μνήμης με διεύθυνση 6E.

C000 Τέλος (HALT).

Ερώτηση για το σπίτι

Υποθέστε ότι ο μετρητής προγράμματος έχει την τιμή 00 και ότι η κύρια μνήμη είναι όπως φαίνεται παρακάτω. Να εκτελέσετε το πρόγραμμα και να δώσετε τις τιμές που παίρνει ο μετρητής προγράμματος και ο καταχωρητής εντολών με χρονολογική σειρά. Εξηγήστε τι κάνει αυτό το πρόγραμμα.

Σημείωση: όλα οι αριθμοί είναι στο δεκαεξαδικό σύστημα.

Κύρια μνήμη	
Διεύθυνση	Περιεχόμενο
00	13
01	14
02	24
03	0A
04	55
05	34
06	22
07	80

Κύρια μνήμη	
Διεύθυνση	Περιεχόμενο
08	81
09	52
0A	20
0B	00
0C	B1
0D	10
0E	25
0F	00

Κύρια μνήμη	
Διεύθυνση	Περιεχόμενο
10	35
11	14
12	C0
13	00
14	22

Ερώτηση για το σπίτι (Απάντηση)

Χρόνος	Μετρητής Προγράμματος	Καταχωρητής εντολών
0	00	-
1	00	1314
2	02	1314
3	02	1314
4	02	240A
5	04	240A
6	04	240A
7	04	5534
8	06	5534
9	06	5534
10	06	2280
11	08	2280
12	08	2280
13	08	8152

Καταχωρητής	Περιεχόμενο
3	22
4	0A
5	2C
2	80

Ερώτηση για το σπίτι (Απάντηση)

Χρόνος	Μετρητής Προγράμματος	Καταχωρητής εντολών
14	0A	8152
15	0A	8152
16	0A	2000
17	0C	2000
18	0C	2000
19	0C	B110
20	0E	B110
21	0E	B110
22	10	B110
23	10	3514
24	12	3514
25	12	3514
26	12	C000
27	14	C000

Καταχωρητής	Περιεχόμενο
1	00
0	00

Έλεγχος ότι κατ0=κατ1

Αποθήκευση του 2C στην κύρια μνήμη στη διεύθυνση 14

Τέλος προγράμματος

Προγράμματα και Δεδομένα

- Στην κύρια μνήμη αποθηκεύονται και προγράμματα και δεδομένα.
- Ερώτηση: Πώς γνωρίζει η μηχανή ποιες πληροφορίες αφορούν προγράμματα και ποιες δεδομένα;
  - Η απάντηση είναι ότι δεν γνωρίζει!
  - Το ποιο πρόγραμμα θα εκτελεστεί ρυθμίζεται στο μετρητή προγράμματος.
  - Αν στο μετρητή προγράμματος αποθηκευτεί μια διεύθυνση δεδομένων, η μηχανή θα τα εκτελούσε σαν εντολές προγράμματος.
- Ερώτηση: Αυτό μήπως σημαίνει ότι η αποθήκευση προγραμμάτων και δεδομένων μαζί στη μνήμη είναι κακή πρακτική;
  - Το αντίθετο! Έχει αποδειχτεί πολύ χρήσιμο, π.χ. ένα πρόγραμμα μπορεί να τροποποιεί κάποιο άλλο πρόγραμμα.

Τέλος Κεφαλαίου 2

- Είδαμε τον τρόπο που η μηχανή χειρίζεται το δεδομένα και τις πληροφορίες που είναι αποθηκευμένες και τον τρόπο εκτέλεσης ενός προγράμματος.

- Στο επόμενο κεφάλαιο θα ασχοληθούμε με το λειτουργικό σύστημα: συντονισμός διεργασιών και χρονοπρογραμματισμός.