**ΕΡΓΑΣΤΗΡΙΑΚΗ ΑΣΚΗΣΗ 6 (ΜΙΚΡΟΠΡΟΓΡΑΜΜΑΤΙΣΜΟΣ 1)**

**05.04.2024**

Μέλη ομάδας :

Τσάλα Ζαφειρία 1084963 ([up1084963@ac.upatras.gr](mailto:up1084963@ac.upatras.gr))

Φουσκαρής Αλέξιος-Ιωσήφ 1100747 ([up1100747@ac.upatras.gr](mailto:up1100747@ac.upatras.gr))

Ομάδα Α4

**Ερώτημα Α.**

**LDA $K : Φόρτωσε τον Accumulator με το περιεχόμενο της διεύθυνσης Κ (της κύριας μνήμης)**

1. PC + 1 -> PC , MAR // To K θα πάει στον MDR

2. MDR + 0 -> Χ // To K στον accumulator

3. Χ + 0 -> MAR // To Κ στον MAR, άρα το περιεχόμενο της Κ στον MDR

4. MDR + 0 -> ACC // O MDR (δηλ. το περιεχ. της Κ) στον ACC

5. PC + 1 -> PC, MAR // O MAR θα δείξει στο opcode της επόμ. εντολής

6. NEXT(PC) // LOAD\_DIR\_SEQ~ για να πάμε στο μικροπρόγραμμα της

Στην παραπάνω μακροεντολή LDA $K μπορούμε να παραλείψουμε τις μικροεντολές MDR + 0 -> Χ και Χ + 0 -> MAR και να τις αντικαταστήσουμε από την ενιαία μικροεντολή MDR+0 -> MAR.

Επομένως ο ψευδοκώδικας γίνεται :

1. PC + 1 -> PC , MAR // To K θα πάει στον MDR

2. MDR + 0 -> MAR // To Κ στον MAR, άρα το περιεχόμενο της Κ στον MDR

3. MDR + 0 -> ACC // O MDR (δηλ. το περιεχ. της Κ) στον ACC

4. PC + 1 -> PC, MAR // O MAR θα δείξει στο opcode της επόμ. εντολής

5. NEXT(PC) // LOAD\_DIR\_SEQ~ για να πάμε στο μικροπρόγραμμα της

**ADD $K : Πρόσθεσε στον Accumulator το περιεχόμενο της διεύθυνσης Κ**

1. PC + 1 -> PC , MAR

2. MDR + 0 -> X

3. X + 0 -> NOP, MAR

4. MDR + ACC -> ACC

5. PC + 1 ->PC, MAR

6. NEXT(PC)

Στην παραπάνω μακροεντολή ADD $K μπορούμε να παραλείψουμε τις μικροεντολές MDR + 0 -> Χ και Χ + 0 -> NOP,MAR και να τις αντικαταστήσουμε από την ενιαία μικροεντολή MDR+0 -> NOP,MAR.

Επομένως ο ψευδοκώδικας γίνεται :

1. PC + 1 -> PC , MAR

2. MDR + 0 -> NOP, MAR

3. MDR + ACC -> ACC

4. PC + 1 ->PC, MAR

5. NEXT(PC)

**STA $K : Αποθήκευσε το περιεχόμενο του Accumulator στη θέση μνήμης με διεύθυνση Κ**

1. PC + 1 -> PC , MAR

2. MDR + 0 -> X

3. X + 0 ->NOP,MAR

4. ACC + 0 -> NOP, MWE~

5. PC + 1 -> PC, MAR

6. NEXT(PC)

Στην παραπάνω μακροεντολή STA $K μπορούμε να παραλείψουμε τις μικροεντολές MDR + 0 -> Χ και Χ + 0 -> NOP,MAR και να τις αντικαταστήσουμε από την ενιαία μικροεντολή MDR+0 -> NOP,MAR (όπως και παραπάνω)

Επομένως ο ψευδοκώδικας γίνεται:

1. PC + 1 -> PC , MAR

2. MDR + 0 -> NOP, MAR

3. ACC + 0 -> NOP, MWE~

4. PC + 1 -> PC, MAR

5. NEXT(PC)

Προφανώς και συμπεραίνουμε από τα παραπάνω πως ο βοηθητικός καταχωρητής Χ είναι αχρείαστος γενικά.

**Περιεχόμενα Mapper και περιεχόμενα κύριας (main) μνήμης:**

|  |  |
| --- | --- |
| MAPPER | MAIN |
| m00 02 // εντολή LDA  m01 08 // εντολή ADD  m02 0e // εντολή STA | m00 00 // opcode εντολής LDA  m01 08 // έντελο εντολής LDA  m02 01 // opcode εντολής ADD  m03 09 // έντελο εντολής ADD  m04 02 // opcode εντολής STA  m05 0a // έντελο εντολής STA  m06 f0  m07 ff  m08 03 // περιοχή δεδομένων  m09 02  m0a 01 |

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **BOOTSTRAP** | BRA | BIN | CON | I | I | I | APORT | BPORT | DDATA | SH~ | SELB | MWE~ | MARCLK | MSTATUS | LDS~ | PCE~ | CARRYE~ | MDE~ | DDATAE~ | **ADDRESS** |
|  | (4:0) | (2:0) | (2:0) | (2:0) | (5:3) | (8:6) | (3:0) | (3:0) | (1:0) |  |  |  |  |  |  |  |  |  |  |  |
| SW+0->PC,MAR | 00000 | 000 | 000 | 111 | 000 | 011 | 0000 | 0001 | 00 | 0 | 1 | 1 | 1 | 0 | 1 | 0 | 1 | 1 | 1 | m00 |
| NEXT(PC) | xxxxx | 000 | xxx | xxx | xxx | 001 | xxxx | xxxx | xx | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | m01 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **LDA $K** | BRA | BIN | CON | I | I | I | APORT | BPORT | DDATA | SH~ | SELB | MWE~ | MARCLK | MSTATUS | LDS~ | PCE~ | CARRYE~ | MDE~ | DDATAE~ | **ADDRESS** |
|  | (4:0) | (2:0) | (2:0) | (2:0) | (5:3) | (8:6) | (3:0) | (3:0) | (1:0) |  |  |  |  |  |  |  |  |  |  |  |
| PC+1->PC,MAR | xxxxx | 000 | xxx | 101 | 000 | 011 | 0001 | 0001 | 01 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | m02 |
| MDR+0->ACC | xxxxx | 000 | xxx | 111 | 000 | 011 | xxxx | 0010 | xx | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | m03 |
| ACC+0->NOP,MAR | xxxxx | 000 | xxx | 100 | 000 | 001 | 0010 | xxxx | xx | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | m04 |
| MDR+0->ACC | 00000 | 000 | 000 | 111 | 000 | 011 | 0000 | 0000 | 00 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | m05 |
| PC+1->PC,MAR | xxxxx | 000 | xxx | 101 | 000 | 011 | 0001 | 0001 | 01 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | m06 |
| NEXT(PC) | xxxxx | 000 | xxx | xxx | xxx | 001 | xxxx | xxxx | xx | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | m07 |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **ADD $K** | BRA | BIN | CON | I | I | I | APORT | BPORT | DDATA | SH~ | SELB | MWE~ | MARCLK | MSTATUS | LDS~ | PCE~ | CARRYE~ | MDE~ | DDATAE~ | **ADDRESS** |
|  | (4:0) | (2:0) | (2:0) | (2:0) | (5:3) | (8:6) | (3:0) | (3:0) | (1:0) |  |  |  |  |  |  |  |  |  |  |  |
| PC+1->PC,MAR | xxxxx | 000 | xxx | 101 | 000 | 011 | 0001 | 0001 | 01 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | m08 |
| MDR+0->X | 00000 | 000 | 000 | 111 | 000 | 011 | 0000 | 0010 | xx | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | m09 |
| X+0->NOP,MAR | 00000 | 000 | 000 | 100 | 000 | 001 | 0010 | 0000 | xx | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | m0a |
| MDR+ACC->ACC | 00000 | 000 | 000 | 101 | 000 | 011 | 0000 | 0000 | 00 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | m0b |
| PC+1->PC,MAR | xxxxx | 000 | xxx | 101 | 000 | 011 | 0001 | 0001 | 01 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | m0c |
| NEXT(PC) | xxxxx | 000 | xxx | xxx | xxx | 001 | xxxx | xxxx | x | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | m0d |
|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| **STA $K** | BRA | BIN | CON | I | I | I | APORT | BPORT | DDATA | SH~ | SELB | MWE~ | MARCLK | MSTATUS | LDS~ | PCE~ | CARRYE~ | MDE~ | DDATAE~ | **ADDRESS** |
|  | (4:0) | (2:0) | (2:0) | (2:0) | (5:3) | (8:6) | (3:0) | (3:0) | (1:0) |  |  |  |  |  |  |  |  |  |  |  |
| PC+1->PC,MAR | xxxxx | 000 | xxx | 101 | 000 | 011 | 0001 | 0001 | 01 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | m0e |
| MDR+0->X | 00000 | 000 | 000 | 111 | 000 | 011 | 0000 | 0010 | 00 | 0 | 1 | 1 | 0 | 0 | 1 | 1 | 1 | 0 | 1 | m0f |
| X+0->NOP,MAR | 00000 | 000 | 000 | 100 | 000 | 001 | 0010 | 0000 | 00 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 1 | m10 |
| ACC+0->NOP,MWE | 00000 | 000 | 000 | 100 | 000 | 001 | 0000 | 0000 | 00 | 0 | 1 | 0 | 0 | 0 | 1 | 1 | 1 | 1 | 1 | m11 |
| PC+1->PC,MAR | xxxxx | 000 | xxx | 101 | 000 | 011 | 0001 | 0001 | 01 | 0 | 1 | 1 | 1 | 0 | 1 | 1 | 1 | 1 | 0 | m12 |
| NEXT(PC) | xxxxx | 000 | xxx | xxx | xxx | 001 | xxxx | xxxx | xx | 0 | 0 | 1 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | m13 |

**Καταχωρητές που χρησιμοποιούμε**Accumulator: (00)16 --> (0000)2

Program Counter: (01)16 --> (0001)2

Βοηθ. Καταχωρητής Χ: (02)16 --> (0010)2

|  |  |  |
| --- | --- | --- |
| **Mapper** | | |
| Κώδικας εντολής | Opcode/Θέση | Περιεχόμενα |
| LDA $K | 00000000 | 00000010 |
| ADD $K | 00000001 | 00001000 |
| STA $K | 00000010 | 00001110 |

|  |  |  |
| --- | --- | --- |
| **Main Memory** | | |
| Κώδικας εντολής | Θέση | Περιεχόμενο |
| LDA $08 | 00000000 | 00000000 |
| 00000001 | 00001000 |
| ADD $09 | 00000010 | 00000001 |
| 00000011 | 00001001 |
| STA $oa | 00000100 | 00000010 |
| 00000101 | 00001010 |
|  | 00000110 | 11110000 |
|  | 00000111 | 11111111 |
|  | 00001000 | 00000011 |
|  | 00001001 | 00000010 |
|  | 00001010 | 00000001 |

|  |
| --- |
| Αποθηκεύουμε στην θέση 0a  το άθροισμα του περιεχομένου των θέσεων 08 και 09 |

**Ερώτημα Β.**

Οι εντολές που θα χρησιμοποιήσουμε είναι:

LDA A

ADD B

STA C

**Ερώτημα Γ.**

Για να εκτελέσουμε την πράξη 10 φορές , θα χρειαστούμε 30 εντολές συνολικά () . Θα χρησιμοποιήσουμε το sequence εντολών που έχουμε παρουσιάσει στο ερώτημα Β

**Ερώτημα Δ.**

Θα μπορούσαμε να ξεπεράσουμε το παραπάνω πρόβλημα με χρήση loop μέσα στο πρόγραμμα. . Το πρόβλημα που προκύπτει από την παραπάνω υλοποίηση(στο ερώτημα Γ) είναι ότι αυξάνεται η μνήμη που καταλαμβάνει το πρόγραμμά μας καθώς και ο χρόνος εκτέλεσής του. Επίσης η συντήρηση του είναι ακόμα πιο δύσκολη και κουραστική.