ΕΡΓΑΣΤΗΡΙΟ ΣΥΓΧΡΟΝΑ ΘΕΜΑΤΑ ΑΡΧΙΤΕΚΤΟΝΙΚΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

Δούρου Βασιλική Ευαγγελία- Α.Μ.:1072633- Εργαστηριακό τμήμα: Β

Πεσκελίδης Παύλος- Α.Μ.:1072483- Εργαστηριακό τμήμα: Β

**Άσκηση 5:**

Οι καταχωρητές που θα χρησιμοποιηθούν στην άσκηση, σύμφωνα με την ενημέρωση, είναι ο Accumulator:0001, ο Program Counter:0011, ο βοηθητικός καταχωρητής X:0101 και ο βοηθητικός καταχωρητής Y:1011.

Οι ψευδοκώδικες για τις ζητούμενες μακροεντολές είναι οι ακόλουθοι:

Για την LOAD R,$K:

PC+1->PC,MAR

MDR+0->X

PC+1->PC,MAR

MDR+0->NOP,MAR

X+0->NOP

MDR+0->RAMF,MAR

PC+1->PC,MAR

NEXT(PC)

Αρχικά, αυξάνουμε κατά ένα τον PC για να διευθυνσιοδοτήσουμε την κύρια μνήμη για να πάρουμε το πρώτο έντελο, δηλαδή τον καταχωρητή R. Έπειτα, περνάμε το περιεχόμενο του MDR στον βοηθητικό καταχωρητή Χ. Στη συνέχεια, αυξάνουμε πάλι τον μετρητή προγράμματος για να πάρουμε το δεύτερο έντελο και διευθυνσιοδοτούμε την κύρια μνήμη για να πάρουμε το περιεχόμενο της θέσης μνήμης με διεύθυνση Κ(δηλαδή το έντελο). Μετά, προσθέτουμε το Χ με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R. Έπειτα, περνάμε το περιεχόμενο του MDR, δηλαδή το περιεχόμενο της διεύθυνσης Κ, στο register file στη θέση που διευθυνσιοδοτείται από το port B και επειδή SELB=0 (όπως θα φανεί στις 40-αδες), θα είναι η θέση που θα δείχνει ο feedback register, δηλαδή ο καταχωρητής R. Τέλος, αυξάνουμε κατά ένα τον PC και διευθυνσιοδοτούμε την κύρια μνήμη με την νέα του τιμή για να αρχίσει την εκτέλεση της επόμενης εντολής και αρχίζει η διερμήνευση της επόμενης μακροεντολής, διαβάζοντας το opcode της.

Για την STORE R,$K:

PC+1->PC,MAR

MDR+0->X

PC+1->PC,MAR

MDR+0->NOP,MAR

X+0->NOP

Bport+0->NOP,MWE~

PC+1->PC,MAR

NEXT(PC)

Όπως και προηγουμένως, αρχικά, αυξάνουμε κατά ένα τον PC για να διευθυνσιοδοτήσουμε την κύρια μνήμη για να πάρουμε το πρώτο έντελο, δηλαδή τον καταχωρητή R. Έπειτα, περνάμε το περιεχόμενο του MDR στον βοηθητικό καταχωρητή Χ. Στη συνέχεια, αυξάνουμε πάλι τον μετρητή προγράμματος για να πάρουμε το δεύτερο έντελο και διευθυνσιοδοτούμε την κύρια μνήμη για να πάρουμε το περιεχόμενο της θέσης μνήμης με διεύθυνση Κ(δηλαδή το έντελο). Μετά, προσθέτουμε το Χ με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R. Στη συνέχεια, κάνουμε enable το σήμα MWE~ για να εγγραφεί στην κύρια μνήμη το περιεχόμενο του port B, δηλαδή το περιεχόμενο του καταχωρητή R, αφού το SELB=0 (όπως θα φανεί στις 40-αδες). Τέλος, αυξάνουμε κατά ένα τον PC και διευθυνσιοδοτούμε την κύρια μνήμη με την νέα του τιμή για να αρχίσει την εκτέλεση της επόμενης εντολής και αρχίζει η διερμήνευση της επόμενης μακροεντολής, διαβάζοντας το opcode της.

Για την ADD R1,$K:

PC+1->PC,MAR

MDR+0->X

PC+1->PC,MAR

MDR+0->NOP,MAR

X+0->NOP

Bport+0->Q,QREG

X+0->NOP

MDR+Q->Bport

PC+1->PC,MAR

NEXT(PC)

Αρχικά, αυξάνουμε κατά ένα τον PC για να διευθυνσιοδοτήσουμε την κύρια μνήμη για να πάρουμε το πρώτο έντελο, δηλαδή τον καταχωρητή R1. Έπειτα, περνάμε το περιεχόμενο του MDR στον βοηθητικό καταχωρητή Χ. Στη συνέχεια, αυξάνουμε πάλι τον μετρητή προγράμματος για να πάρουμε το δεύτερο έντελο και διευθυνσιοδοτούμε την κύρια μνήμη για να πάρουμε το περιεχόμενο της θέσης μνήμης με διεύθυνση Κ(δηλαδή το έντελο). Μετά, προσθέτουμε το Χ με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R1. Έπειτα, περνάμε το περιεχόμενο του port B, δηλαδή του καταχωρητή R1 αφού SELB=0 (όπως φαίνεται στις 40-αδες), στον καταχωρητή Q για να μπορέσουμε να προσθέσουμε τα δύο περιεχόμενα που θέλουμε. Έπειτα, προσθέτουμε πάλι το Χ με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R1 και προσθέτουμε τα περιεχόμενα του MDR, δηλαδή τα περιεχόμενα της θέσης μνήμης με διεύθυνση Κ, με τα περιεχόμενα του Q και αποθηκεύουμε το αποτέλεσμα στο port B, δηλαδή στον καταχωρητή R1, αφού θα είναι SELB=0 (όπως φαίνεται στις 40-αδες) και στον feedback register είναι από την προηγούμενη εντολή ο καταχωρητής R1. Τέλος, αυξάνουμε κατά ένα τον PC και διευθυνσιοδοτούμε την κύρια μνήμη με την νέα του τιμή για να αρχίσει την εκτέλεση της επόμενης εντολής και αρχίζει η διερμήνευση της επόμενης μακροεντολής, διαβάζοντας το opcode της.

Για την SUB R1,$K:

PC+1->PC,MAR

MDR+0->X

PC+1->PC,MAR

MDR+0->NOP,MAR

X+0->NOP

Bport+0->Q,QREG

X+0->NOP

Q-MDR->Bport

PC+1->PC,MAR

NEXT(PC)

Παρόμοια με την προηγούμενη εντολή, αρχικά, αυξάνουμε κατά ένα τον PC για να διευθυνσιοδοτήσουμε την κύρια μνήμη για να πάρουμε το πρώτο έντελο, δηλαδή τον καταχωρητή R1. Έπειτα, περνάμε το περιεχόμενο του MDR στον βοηθητικό καταχωρητή Χ. Στη συνέχεια, αυξάνουμε πάλι τον μετρητή προγράμματος για να πάρουμε το δεύτερο έντελο και διευθυνσιοδοτούμε την κύρια μνήμη για να πάρουμε το περιεχόμενο της θέσης μνήμης με διεύθυνση Κ(δηλαδή το έντελο). Μετά, προσθέτουμε το Χ με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R1. Έπειτα, περνάμε το περιεχόμενο του port B, δηλαδή του καταχωρητή R1 αφού SELB=0 (όπως φαίνεται στις 40-αδες), στον καταχωρητή Q για να μπορέσουμε να προσθέσουμε τα δύο περιεχόμενα που θέλουμε. Έπειτα, προσθέτουμε πάλι το Χ με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R1 και αφαιρούμε τα περιεχόμενα του MDR, δηλαδή τα περιεχόμενα της θέσης μνήμης με διεύθυνση Κ, από τα περιεχόμενα του Q και αποθηκεύουμε το αποτέλεσμα στο port B, δηλαδή στον καταχωρητή R1, αφού θα είναι SELB=0 (όπως φαίνεται στις 40-αδες) και στον feedback register είναι από την προηγούμενη εντολή ο καταχωρητής R1. Τέλος, αυξάνουμε κατά ένα τον PC και διευθυνσιοδοτούμε την κύρια μνήμη με την νέα του τιμή για να αρχίσει την εκτέλεση της επόμενης εντολής και αρχίζει η διερμήνευση της επόμενης μακροεντολής, διαβάζοντας το opcode της.

Για την ADD R1,R2:

PC+1->PC,MAR

MDR+0->X

PC+1->PC,MAR

MDR+0->Y

X+0->NOP

Bport+0->Q,QREG

Q+0->X

Y+0->NOP

Bport+X->Bport

PC+1->PC,MAR

NEXT(PC)

Σχόλιο: Καθώς δεν διευκρινίστηκε από την εκφώνηση αν μετά την πρόσθεση των περιεχομένων των δύο καταχωρητών τα αποτελέσματα θα αποθηκευτούν στον R1 ή στον R2, εμείς επιλέξαμε να τα αποθηκεύσουμε στον R2.

Αρχικά, αυξάνουμε κατά ένα τον PC για να διευθυνσιοδοτήσουμε την κύρια μνήμη για να πάρουμε το πρώτο έντελο, δηλαδή τον καταχωρητή R1. Έπειτα, περνάμε το περιεχόμενο του MDR στον βοηθητικό καταχωρητή Χ. Στη συνέχεια, αυξάνουμε πάλι τον μετρητή προγράμματος για να πάρουμε το δεύτερο έντελο, δηλαδή τον καταχωρητή R2. Έπειτα, περνάμε το περιεχόμενο του MDR στον βοηθητικό καταχωρητή Y. Μετά, προσθέτουμε το Χ με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R1. Έπειτα, περνάμε το περιεχόμενο του port B, δηλαδή του καταχωρητή R1 αφού SELB=0 (όπως φαίνεται στις 40-αδες), στον καταχωρητή Q και, στη συνέχεια, αποθηκεύουμε τα περιεχόμενα του Q στον Χ. Ακόμη, προσθέτουμε το Y με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R2 και προσθέτουμε τα περιεχόμενα του port B, δηλαδή του καταχωρητή R2, αφού SELB=0 (όπως φαίνεται στις 40-αδες), με τα περιεχόμενα του Χ, δηλαδή του καταχωρητή R1, και αποθηκεύουμε τα αποτελέσματα στον καταχωρητή R2. Τέλος, αυξάνουμε κατά ένα τον PC και διευθυνσιοδοτούμε την κύρια μνήμη με την νέα του τιμή για να αρχίσει την εκτέλεση της επόμενης εντολής και αρχίζει η διερμήνευση της επόμενης μακροεντολής, διαβάζοντας το opcode της.

Για την SHL R:

PC+1->PC,MAR

MDR+0->NOP

Bport+0->Bport,SRAMU

PC+1->PC,MAR

NEXT(PC)

Αρχικά, αυξάνουμε κατά ένα τον PC για να διευθυνσιοδοτήσουμε την κύρια μνήμη για να πάρουμε το έντελο, δηλαδή τον καταχωρητή R. Στη συνέχεια, προσθέτουμε τον MDR με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R και κάνουμε ολίσθηση στα περιεχόμενα του port B, δηλαδή του καταχωρητή R, αφού SELB=0 (όπως θα φανεί στις 40-αδες), και γράφουμε το SRAMU για να δηλώσουμε ότι θα κάνουμε απλή αριστερή ολίσθηση (θα θέσουμε I[8:6]:111 και SH~:0 όπως θα φανεί και στις 40-αδες). Τέλος, αυξάνουμε κατά ένα τον PC και διευθυνσιοδοτούμε την κύρια μνήμη με την νέα του τιμή για να αρχίσει την εκτέλεση της επόμενης εντολής και αρχίζει η διερμήνευση της επόμενης μακροεντολής, διαβάζοντας το opcode της.

Για την DEC R:

PC+1->PC,MAR

MDR+0->NOP

Bport+0->Q,QREG

X+0->NOP

Q-1->Bport

PC+1->PC,MAR

NEXT(PC)

Αρχικά, αυξάνουμε κατά ένα τον PC για να διευθυνσιοδοτήσουμε την κύρια μνήμη για να πάρουμε το έντελο, δηλαδή τον καταχωρητή R. Στη συνέχεια, προσθέτουμε τον MDR με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R και περνάμε τα περιεχόμενα του port B, δηλαδή του καταχωρητή R, αφού SELB=0 (όπως θα φανεί στις 40-αδες), στον καταχωρητή Q. Μετά, προσθέτουμε το Χ με το 0, για να μπορέσουμε να έχουμε στον feedback register τον καταχωρητή R και, έπειτα, αφαιρούμε από το Q, δηλαδή το περιεχόμενο του R, το 1 και αποθηκεύουμε το αποτέλεσμα στο port B, δηλαδή στον καταχωρητή R, αφού SELB=0 (όπως θα φανεί στις 40-αδες). Τέλος, αυξάνουμε κατά ένα τον PC και διευθυνσιοδοτούμε την κύρια μνήμη με την νέα του τιμή για να αρχίσει την εκτέλεση της επόμενης εντολής και αρχίζει η διερμήνευση της επόμενης μακροεντολής, διαβάζοντας το opcode της.

Για την HALT:

PC+0->PC,MAR

Για αυτή την μακροεντολή απλά προσθέτουμε το μηδέν στον PC για να μείνει ίδια η τιμή του και να μην μπορεί να συνεχιστεί το πρόγραμμα.

Επίσης, ο ψευδοκώδικας του Bootstrap είναι ο ακόλουθος:

1. Switches + 0 -> PC, MAR

2. NEXT(PC)

Εδώ, αρχικοποιείται το PC ανάλογα με τις τιμές των dip switches και στη συνέχεια, αρχίζει η διερμήνευση της επόμενης μακροεντολής, διαβάζοντας το opcode της.

Οι 40-αδες για τις παραπάνω εντολές είναι οι ακόλουθες:















Το ζητούμενο μακροπρόγραμμα είναι το ακόλουθο, όπου R1:02 και R2:07:

LOAD 02,$20

ADD 02,$21

SUB 02,$22

DEC 02

SHL 02

SHL 02

LOAD 07,$20

ADD 07,$21

DEC 07

SHL 07

ADD 02,07

STORE 07,$23

HALT

Θα θεωρήσουμε για κάθε μακροεντολή ένα μοναδικό opcode. Έτσι, θα έχουμε LOAD R,$K:00H, STORE R,$K:01H, ADD R1,$K:02H, SUB R1,$K:03H, ADD R1,R2:04H, SHL R:05H, DEC R:06H και HALT:07H.

Στη συνέχεια, θα θεωρήσουμε ότι το μικροπρόγραμμα για κάθε μακροεντολή ξεκινάει από την ακόλουθη διεύθυνση της μικρομνήμης:

LOAD R,$K:02H

STORE R,$K:0AH

ADD R1,$K:12H

SUB R1,$K:1CH

ADD R1,R2:26H

SHL R:31H

DEC R:36H

HALT:3DH

Τα περιεχόμενα του Mapper φαίνονται στον ακόλουθο πίνακα:

|  |  |  |
| --- | --- | --- |
| **Mapper** | | |
| Κώδικας εντολής | Opcode/Θέση | Περιεχόμενα |
| LOAD R,$K | 00000000 | 00000010 |
| STORE R,$K | 00000001 | 00001010 |
| ADD R1,$K | 00000010 | 00010010 |
| SUB R1,$K | 00000011 | 00011100 |
| ADD R1,R2 | 00000100 | 00100110 |
| SHL R | 00000101 | 00110001 |
| DEC R | 00000110 | 00110110 |
| HALT | 00000111 | 00111101 |

Τα περιεχόμενα της κύριας μνήμης, αφού δεν δόθηκαν ούτε οι θέσεις W,X,Y,Z ούτε τα περιεχόμενα τους, είναι τα ακόλουθα:

|  |  |  |
| --- | --- | --- |
| **Main Memory** | | |
| Κώδικας εντολής | Θέση | Περιεχόμενο |
| LOAD 02,$20 |  |  |
| 00000000 | 00000000 |
| 00000001 | 00000010 |
| 00000010 | 00100000 |
| ADD 02,$21 | 00000011 | 00000010 |
| 00000100 | 00000010 |
| 00000101 | 00100001 |
| SUB 02,$22 | 00000110 | 00000011 |
| 00000111 | 00000010 |
| 00001000 | 00100010 |
| DEC 02 | 00001001 | 00000110 |
| 00001010 | 00000010 |
| SHL 02 | 00001011 | 00000101 |
| 00001100 | 00000010 |
| SHL 02 | 00001101 | 00000101 |
| 00001110 | 00000010 |
| LOAD 07,$20 | 00001111 | 00000000 |
| 00010000 | 00000111 |
| 00010001 | 00100000 |
| ADD 07,$21 | 00010010 | 00000010 |
| 00010011 | 00000111 |
| 00010100 | 00100001 |
| DEC 07 | 00010101 | 00000110 |
| 00010110 | 00000111 |
| SHL 07 | 00010111 | 00000101 |
| 00011000 | 00000111 |
| ADD 02,07 | 00011001 | 00000100 |
| 00011010 | 00000010 |
| 00011011 | 00000111 |
| STORE 07,$23 | 00011100 | 00000001 |
| 00011101 | 00000111 |
| 00011110 | 00100011 |
| HALT | 00011111 | 00000111 |
| Περιοχή Δεδομένων:  X: | 00100000 | 00000100 |
| Y: | 00100001 | 00000011 |
| Z: | 00100010 | 00000010 |

Το αποτέλεσμα στην κύρια μνήμη, στη θέση μνήμης W:

m23 1C