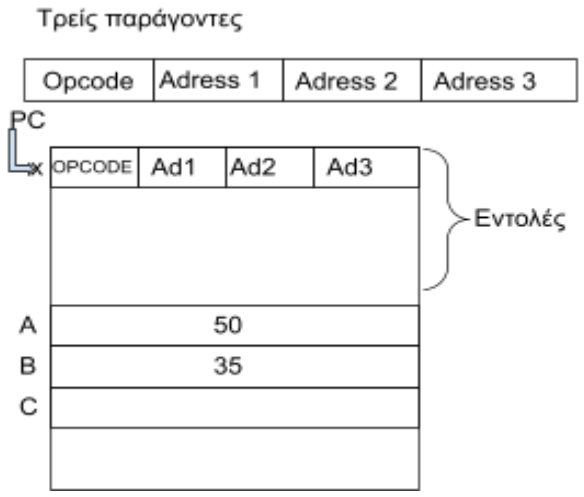
Αρχιτεκτονική Διάλεξη 7

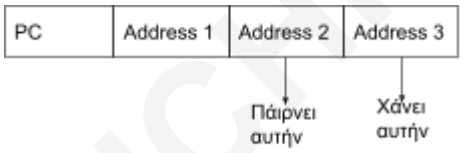


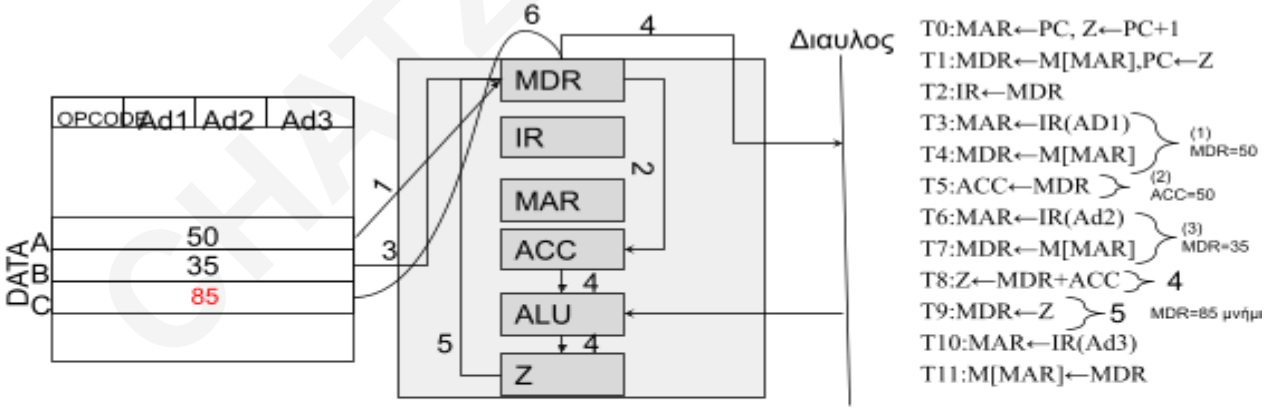
Γίνεται η επόμενη εντολή που δείχνει ο PC να είναι 4 θέσεις μνήμης πιο κάτω 4 και παραπάνω.
Δηλαδή να είναι
PC←PC+4(5,6,...κτλπ)

Σε πολλά συστήματα στο 3ο βήμα της ανάκλησης συμβαίνει ο MDR να φορτώνεται στο IR. Το μέγεθος δλδ του MDR=IR δηλαδή μια ολόκληρη λέξη.

T2:IR ← MDR(OPCODE)

Αρα δεν θα εχουμε το πρόβλημα που είχαμε στην διαλεξη 6 να χάνονται διευθύνσεις τύπου :





Άσκηση:

Οι εντολές ενός συστήματος εχουν γενικά την μορφή :

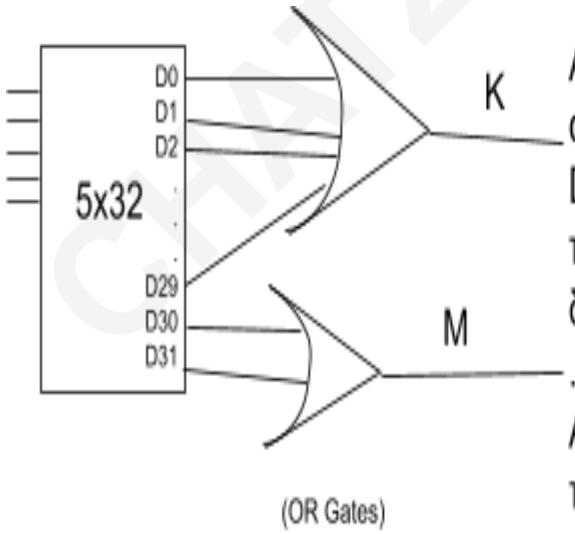
OPCODE	Address 1	Address 2	Address 3	
5 bit	5 bit	5 bit	5 bit	20 bit

Να εξετάσετε αν το σύστημα μπορεί να υποστηρίξει: 30 εντολές 3 παραγόντων
63 εντολές 2 παραγόντων
31 εντολές 1 παραγόντων
32 εντολές 0 παραγόντων

Λύση:

Ξεκινάμε με εντολές 3 παραγόντων.

OPCODE	XXXXX	YYYYY	ZZZZZ	
00000	XXXXX	YYYYY	ZZZZZ	1η εντολή 3 παραγοντων
00001	XXXXX	YYYYY	ZZZZZ	2η εντολή 3 παραγοντων
00010	XXXXX	YYYYY	ZZZZZ	3η εντολή 3 παραγοντων
……..	……..	……..	……..	
11101	XXXXX	YYYYY	ZZZZZ	30η εντολή 3 παραγοντων



Αν τα 5 πρώτα bit σχηματίσουν τους αριθμούς 0-29, τότε μια απο τις εξόδους του DEC θα είναι 1 τότε K=1 αρα έχουμε εντολή τριών παραγόντων, και οι παράγοντες θα δινονται απο τα πεδία X,Y,Z(5 bit το καθένα)

.

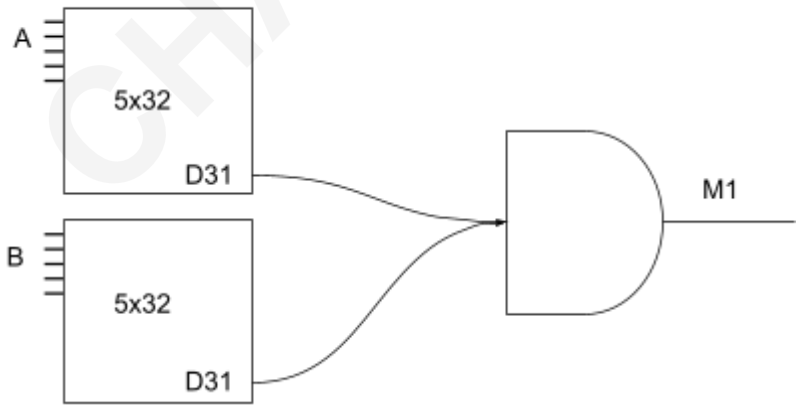
Αν K=0 και M=1, ή εντολή δεν είναι τριών παραγόντων

ΆΡΑ ΓΙΑ ΕΝΤΟΛΈΣ 3 ΠΑΡΑΓΌΝΤΩΝ (30) ΕΊΜΑΣΤΕ ΚΑΛΆ

Για 63 εντολές 2 παραγόντων ?

Από το αριστερό πεδίο 5 Bit A έχουν δεσμευτεί 30 από τους 32 συνδυασμούς για εντολές τριών παραγόντων. Περισσεύουν 2 οι οποίοι σε συνδυασμό με τους 32 συνδυασμούς του πεδίου X προσφέρουν δυνατότητα για το πολύ 2x32 = 64 εντολές δύο παραγόντων (Y,Z)

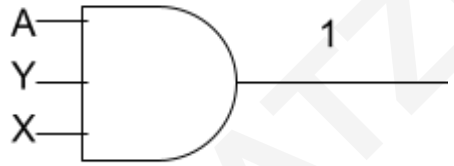
OPCODE	XXXXX	YYYYY	ZZZZZ	
11110	00000	YYYYY	ZZZZZ	1η εντολή 2 παραγοντων
11110	00001	YYYYY	ZZZZZ	2η εντολή 2 παραγοντων
11110	00010	YYYYY	ZZZZZ	3η εντολή 2 παραγοντων
……..	……..	……..	……..	
11110	11111	YYYYY	ZZZZZ	32η εντολή 2 παραγοντων
11111	00000	YYYYY	ZZZZZ	33η εντολή 2 παραγοντων
11111	00001	YYYYY	ZZZZZ	34η εντολή 2 παραγοντων
……..	…….	……..	……..	
11111	11110	YYYYY	ZZZZZ	63η εντολή 2 παραγοντων



Απο το πεδίο X περισσέυει ο συνδιασμος IIIII=31 ο οποίος με τους 32 συνδιασμους του Y μπορεί να μου δώσει το πολύ 32 εντολές ενος παράγοντα Z.

Αρα και με 2 εντολές είμαι εντάξει

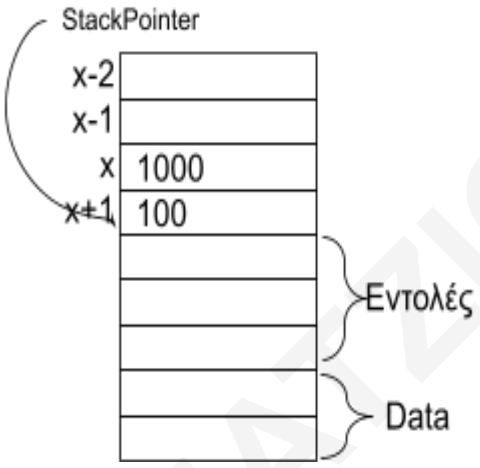
OPCODE	XXXXX	YYYYY	ZZZZZ	
11110	11111	00000	ZZZZZ	1η εντολή 1 παραγοντων
11111	11111	00001	ZZZZZ	2η εντολή 1 παραγοντων
11111	11111	00010	ZZZZZ	3η εντολή 1 παραγοντων
……..	……..	……..	……..	
11111	11111	11110	ZZZZZ	31η εντολή 1 παραγοντων
11111	11111	11111	00000	1η εντολή 0 παραγοντων
11111	11111	11111	00001	2η εντολή 0 παραγοντων
……..	…….	……..	……..	
11111	11111	11111	11111	32η εντολή 0 παραγοντων



Αμα βγει ενα ειναι μηδεν παραγοντες

1 Εντολή 1
2 Εντολή 2
3 Εντολή 3
4 x=a method(a,b)
5 Εντολή 5
20 a method( )
27 return

Για Να κλιθεί η υπουρουτινα πρέπει να ανακληθεί η εντολή 4, ο PC (Ο οποίος θα δείχνει 5) να αποθηκευτεί έτσι ώστε να γνωρίζουμε που θα επιστρέψουμε όταν εκτελεστεί η return. Η αποθήκευση γίνεται στη στοίβα



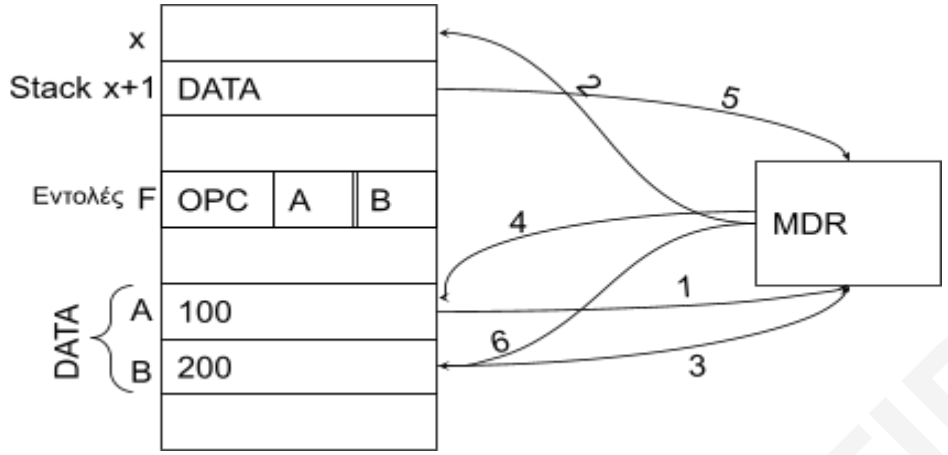
StackPointer(SP): Δείχνει την τελευταια θέση που εχει χρησιμοποιηθει.

Για να γραψουμε στην στοιβα
1)SP=SP-1
2)Η τιμή τους SP → MAR
3) τα data θα πανε μεσω MDR

Έστω η τελευταία γεμάτη x+1 εχει τιμή 100
Ο SP=x+1 ο SP μειωνεται κατα 1 και δειχνει την επομενη ελευθερη (την X). Ο MAR λαμβάνει την τιμή X
Ο MDR στέλνει το 1000 στην θέση που δειχνει ο MAR

Άσκηση

Εναλλαγη των τιμών που βρισκονται στις θέσεις A,B χρησιμοποιουνται ως ενδιάμεση μνήμη την κορυφή της στοίβας.



T0: MAR←PC,Z ← PC+1	MAR←F, Z ← F+1	
T1: MDR←M[MAR], PC←Z	MDR← |OPC|A|B, PC ← F+1	
T2:IR←MDR		
T3:MAR←IR[Address 1]	MAR←A	} 1
T4:MDR←M[MAR]	MDR←100	
T5: Z←SP-1	Z←X	} 2
T6:SP←Z, MAR←Z	SP←X, MAR←X	
T7:M[MAR]←MDR	M[X]←100	
T8:MAR←IR(Address 2)	MAR←B	} 3
T9:MDR←M[MAR]	MDR←M[B]=200	
T10 MAR←IR(Address 1)	MAR←A	
T11 M[MAR]←MDR	M[A]←200 (το 200 παει τελική θέση	
T12 MAR←SR, Z←SP+1	MAR←X,Z←X+1 ο SP εχει διαυλο	
T13 SP←Z, MDR←M[MAR]	SP←X+1 μεσω εσωτερικου διαυλου ταυτοχρονα ο MDR πάιρνει τα δεδομένα MDR=100	
T14 MAR← IR[Address 2]		
T15 M[MAR]←Mdr		