Συνδυαστικά Κυκλώματα

Ο σχεδιασμός συνδυαστικών κυκλωμάτων αρχίζει με μια προφορική διατύπωση του προβλήματος και τελειώνει με ένα λογικό διάγραμμα του κυκλώματος ή με κάποιες συναρτήσεις Boole από τις οποίες μπορεί να βγει το λογικό διάγραμμα. Η διαδικασία περιλαμβάνει τα εξής βήματα:

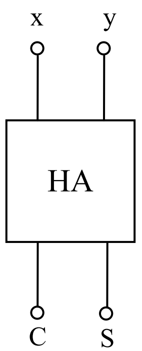
- Καθορίζεται το πρόβλημα.
- Καθορίζεται ο αριθμός των διαθέσιμων μεταβλητών εισόδου και των απαιτούμενων μεταβλητών εξόδου.
- Διαλέγουμε συμβολικά ονόματα για τις μεταβλητές εισόδου και εξόδου.
- Βρίσκουμε τον πίνακα αληθείας που καθορίζει τις απαιτούμενες σχέσεις μεταξύ εισόδων και εξόδων.
- Απλοποιούμε τη συνάρτηση Boole για κάθε έξοδο.
- Σχεδιάζουμε το λογικό διάγραμμα.

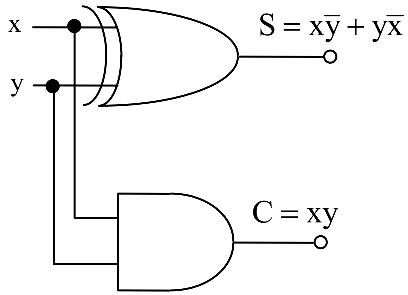
Ημιαθροιστής (Half-Adder)

x	y	C	S
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

$$S = \overline{x}y + x\overline{y}$$

$$C = xy$$





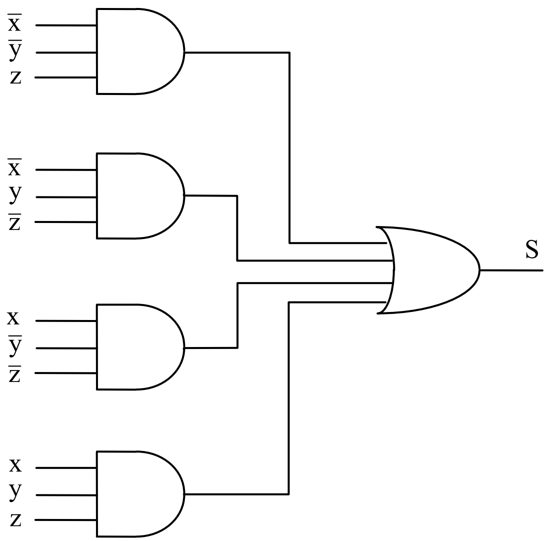
-

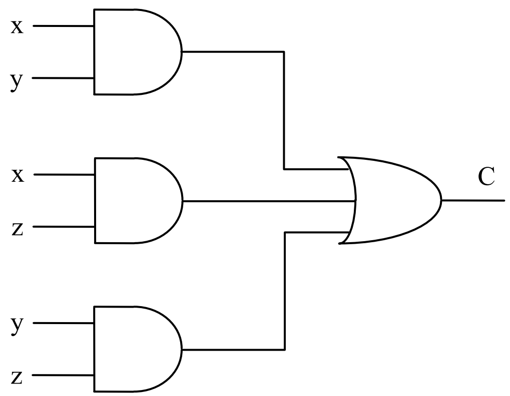
Πλήρης Αθροιστής (Full-Adder)

x	y	z	C	S
0	0	0	0	0
0	0	1	0	1
0	1	0	0	1
0	1	1	1	0
1	0	0	0	1
1	0	1	1	0
1	1	0	1	0
1	1	1	1	1

$$S = \bar{x}\,\bar{y}z + \bar{x}y\bar{z} + x\bar{y}\,\bar{z} + xyz$$

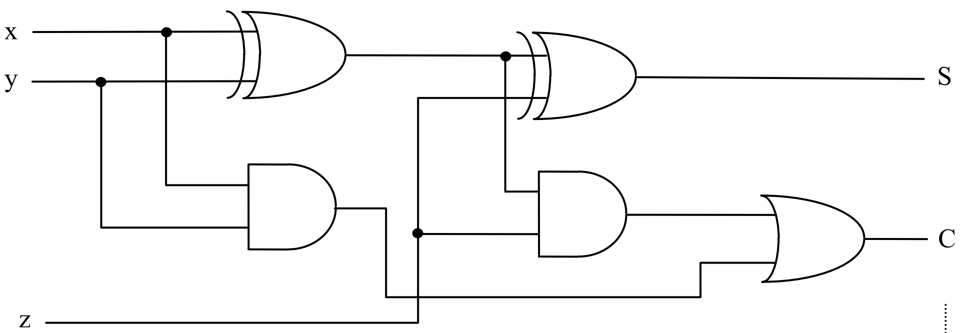
$$C = xy + xz + yz$$

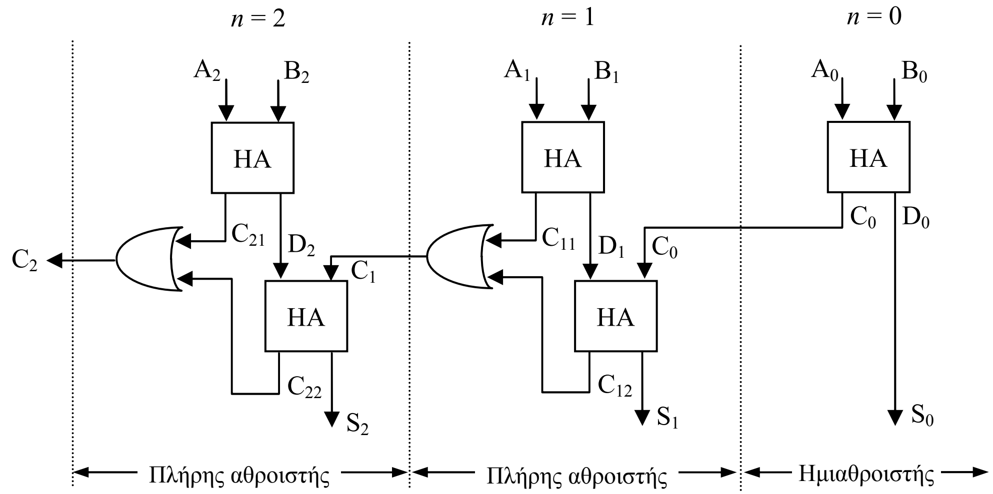


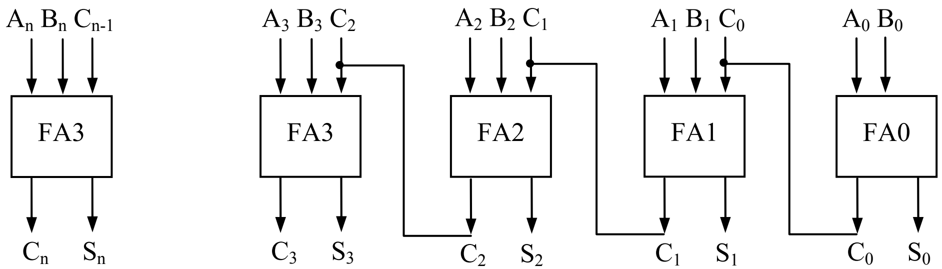


$$S = z \oplus (x \oplus y) = \bar{z}(x\bar{y} + \bar{x}y) + z(x\bar{y} + \bar{x}y) = \bar{z}(x\bar{y} + \bar{x}y) + z(xy + \bar{x}\,\bar{y}) = x\bar{y}\,\bar{z} + \bar{x}y\bar{z} + xyz + \bar{x}\,\bar{y}z$$

και η έξοδος του κρατουμένου είναι: $C = z(x\bar{y} + \bar{x}y) + xy = x\bar{y}z + \bar{x}yz + xy = z(x \oplus y) + xy$







-

Ημιαφαιρέτης (Half-Subtractor)

x	y	B	D
0	0	0	0
0	1	1	1
1	0	0	1
1	1	0	0

Το κρατούμενο εξόδου B είναι 0 υπό τον όρο ότι $x \geq y$. Είναι 1 για $x = 0$ και $y = 1$. Το D είναι το αποτέλεσμα της πράξης $2B + x - y$.

$$D = \overline{x}y + x\overline{y}$$

$$B = \overline{x}y$$

Είναι ενδιαφέρον να σημειωθεί ότι η συνάρτηση του D είναι ακριβώς η ίδια όπως η λογική συνάρτηση του S του ημιαθροιστή.

Πλήρης Αφαιρέτης (Full Subtractor)

x	y	z	B	D
0	0	0	0	0
0	0	1	1	1
0	1	0	1	1
0	1	1	1	0
1	0	0	0	1
1	0	1	0	0
1	1	0	0	0
1	1	1	1	1

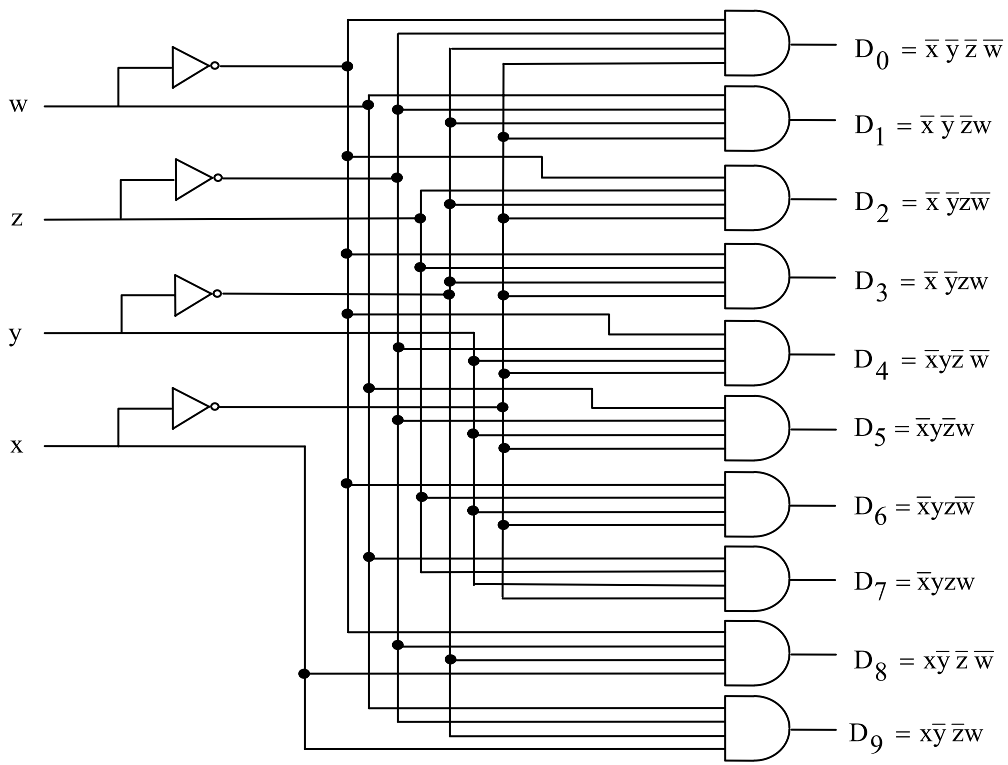
$$D = \overline{x}\,\overline{y}z + \overline{x}y\overline{z} + x\overline{y}\,\overline{z} + xyz$$

$$B = \overline{x}y + \overline{x}z + yz$$

Σημαντικό: Ο πλήρης αφαιρέτης μπορεί να χρησιμοποιηθεί ως συγκριτής δύο αριθμών. Εάν το κρατούμενο z της τελευταίας βαθμίδας (δηλ. της βαθμίδας υψηλότερης σημαντικότητας) είναι 0 τότε ο μειωτέος x είναι μεγαλύτερος από τον αφαιρετέο y. Αν το κρατούμενο z της τελευταίας βαθμίδας είναι 1 τότε ο μειωτέος x είναι μικρότερος από τον αφαιρετέο y.

Αποκωδικοποιητές (Decoders)

Ένας *αποκωδικοποιητής* (decoder) είναι ένα συνδυαστικό κύκλωμα που μετατρέπει τη δυαδική πληροφορία *n* γραμμών εισόδου σε έως 2^n μοναδικές γραμμές εξόδου. Οι αποκωδικοποιητές που θα μας απασχολήσουν ονομάζονται αποκωδικοποιητές από *n* σε *m* γραμμές, όπου $m \leq 2^n$. Σκοπός τους είναι να παράγουν τους 2^n (ή λιγότερους) ελαχιστόρους των *n* μεταβλητών εισόδου.



-

Μετατροπή από δυαδικό σε δεκαδικό αριθμό

Είσοδοι				Έξοδοι									
x	y	z	w	D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	D_8	D_9
0	0	0	0	1	0	0	0	0	0	0	0	0	0
0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	1	0	0	0	1	0	0	0	0	0	0	0
0	0	1	1	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	1	0	0	0	0	0
0	1	0	1	0	0	0	0	0	1	0	0	0	0
0	1	1	0	0	0	0	0	0	0	1	0	0	0
0	1	1	1	0	0	0	0	0	0	0	1	0	0
1	0	0	1	0	0	0	0	0	0	0	0	1	0
1	0	0	1	0	0	0	0	0	0	0	0	0	1

Ένας αποκωδικοποιητής μας δίνει τους 2^n ελαχιστόρους των *n* μεταβλητών εισόδου. Αφού κάθε συνάρτηση Boole μπορεί να εκφραστεί σε κανονική μορφή αθροίσματος ελαχιστόρων, **μπορεί κανείς να χρησιμοποιήσει έναν αποκωδικοποιητή για να παράγει τους ελαχιστόρους και μία εξωτερική πύλη OR για να σχηματίσει το άθροισμα.**

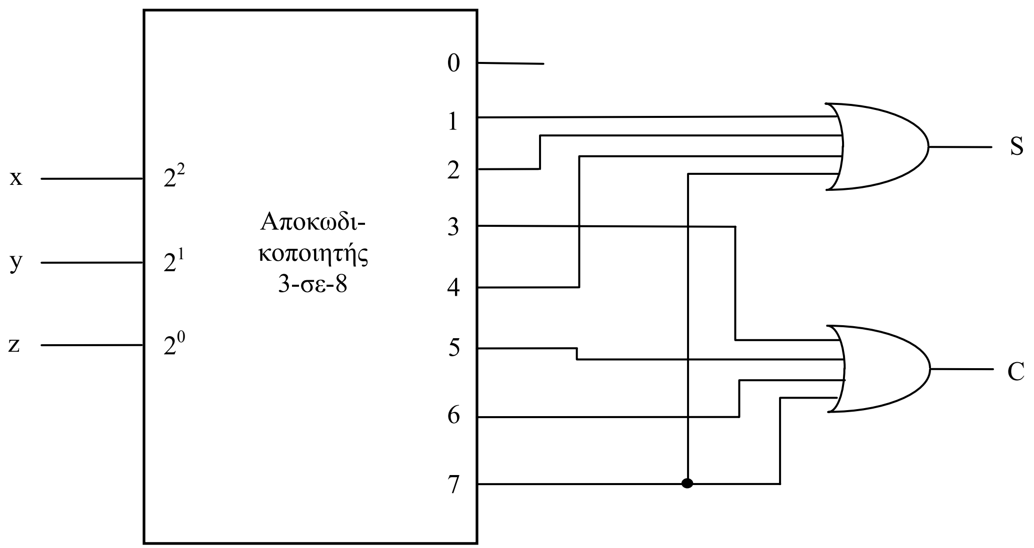
Παράδειγμα:

Να υλοποιηθεί ένα κύκλωμα πλήρη αθροιστή με έναν αποκωδικοποιητή και δύο πύλες OR.

Λύση: Από τον πίνακα αληθείας του πλήρη αθροιστή, παίρνουμε τις συναρτήσεις αυτού του συνδυαστικού κυκλώματος σαν άθροισμα ελαχιστόρων

$$S(x, y, z) = \Sigma(1, 2, 4, 7)$$

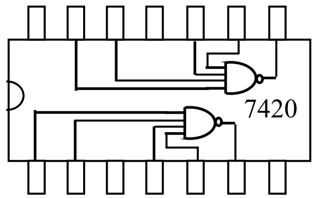
$$C(x, y, z) = \Sigma(3, 5, 6, 7)$$



-

Παράδειγμα:

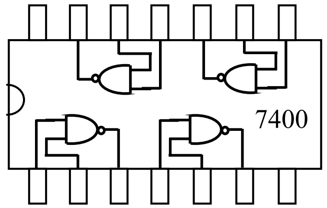
Σε ένα συμβούλιο αξιολόγησης μίας προμήθειας υπάρχουν 3 κριτές. Οι ψήφοι τους έχουν διαφορετικά βάρη:
Του κριτή A έχει βάρος 3
Του κριτή B έχει βάρος 2
Του κριτή Γ έχει βάρος 1



Καθένας από τους κριτές έχει έναν διακόπτη που τον κλείνει για να ψηφίσει ΝΑΙ και τον ανοίγει για να ψηφίσει ΟΧΙ. Σε περίπτωση ισοψηφίας το αποτέλεσμα είναι ΟΧΙ.

1) Να δώσετε τον πίνακα αληθείας της συνάρτησης που υλοποιεί το αποτέλεσμα της ψηφοφορίας.

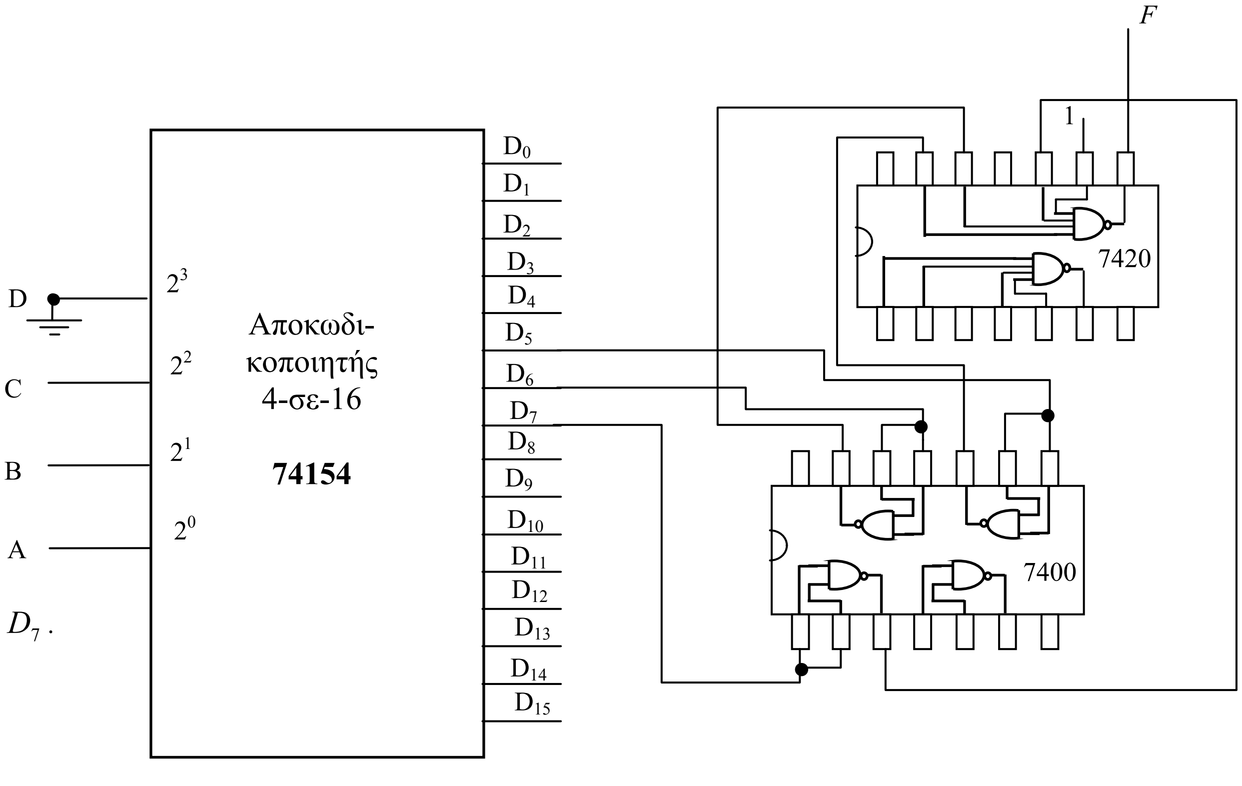
2) Από τον παραπάνω πίνακα αληθείας και χωρίς να απλοποιήσετε τη συνάρτηση που προκύπτει να σχεδιάσετε το κύκλωμα που δείχνει αν το αποτέλεσμα της ψηφοφορίας είναι ΝΑΙ ή ΟΧΙ.



Έχετε στη διάθεσή σας τα ολοκληρωμένα 74154 (αποκωδικοποιητής 4-σε-16), 7420 και 7400 (των οποίων η χωροδιάταξη παρουσιάζεται παραπλεύρως).

Λύση

3	2	1	
A	B	C	F
0	0	0	0
0	0	1	0
0	1	0	0
0	1	1	0
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1



$$F = A\overline{B}C + AB\overline{C} + ABC = D_5 + D_6 + D_7 \,.$$

Κωδικοποιητές (Coders)

Ένας κωδικοποιητής είναι ένα ψηφιακό κύκλωμα που εκτελεί την αντίστροφη λειτουργία από ό,τι ο αποκωδικοποιητής. Ένας αποκωδικοποιητής έχει 2^n (ή λιγότερες) γραμμές εισόδου και *n* γραμμές εξόδου. Οι γραμμές εξόδου παράγουν το δυαδικό κώδικα που αντιστοιχεί στις μεταβλητές εισόδου.

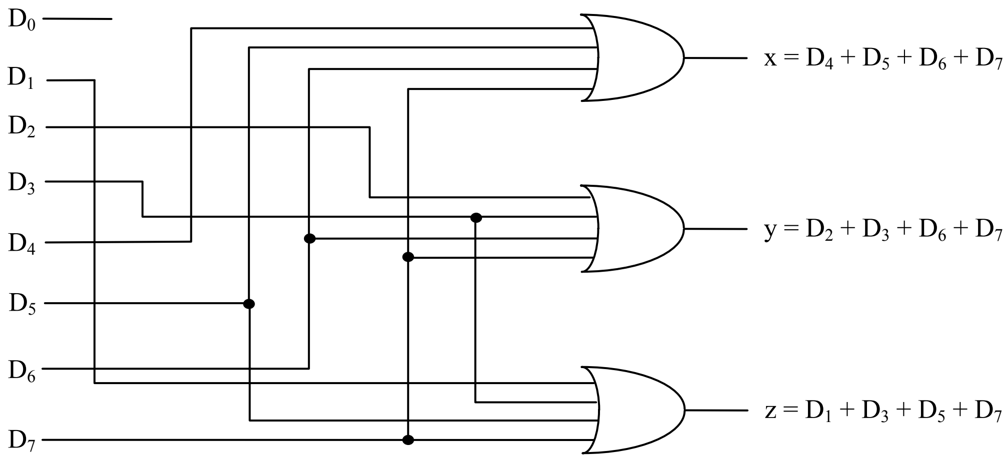
Είσοδοι								Έξοδοι		
D_0	D_1	D_2	D_3	D_4	D_5	D_6	D_7	x	y	z
1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	1
0	0	1	0	0	0	0	0	0	1	0
0	0	0	1	0	0	0	0	0	1	1
0	0	0	0	1	0	0	0	1	0	0
0	0	0	0	0	1	0	0	1	0	1
0	0	0	0	0	0	1	0	1	1	0
0	0	0	0	0	0	0	1	1	1	1

Ο κωδικοποιητής μπορεί να υλοποιηθεί με πύλες OR, των οποίων οι είσοδοι καθορίζονται κατευθείαν από τον πίνακα αληθείας. Η έξοδος z είναι ίση με 1, αν το οκταδικό ψηφίο εισόδου είναι 1 ή 3 ή 5 ή 7. Η έξοδος y είναι 1 για τα οκταδικά ψηφία 2, 3, 6, ή 7 και η έξοδος x είναι 1 για τα ψηφία 4, 5, 6, ή 7. Αυτές οι συνθήκες μπορούν να εκφραστούν από τις επόμενες συναρτήσεις Boole εξόδου

$$z = D_1 + D_3 + D_5 + D_7$$

$$y = D_2 + D_3 + D_6 + D_7$$

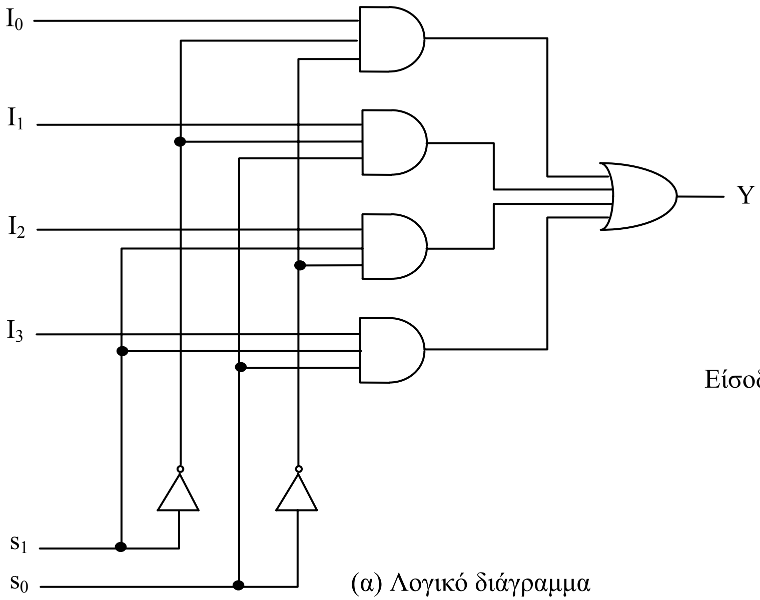
$$x = D_4 + D_5 + D_6 + D_7$$



-

Πολυπλέκτες (Multiplexers)

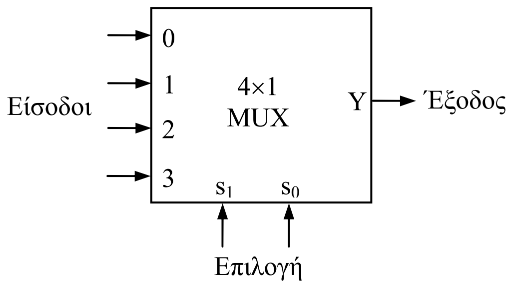
Πολυπλεξία (multiplexing) σημαίνει τη μεταβίβαση ενός μεγάλου αριθμού πληροφοριών μέσα από ένα μικρότερο αριθμό καναλιών ή γραμμών. Ένας ψηφιακός *πολυπλέκτης* είναι ένα συνδυαστικό κύκλωμα που επιλέγει δυαδικές πληροφορίες ανάμεσα σε πολλές γραμμές εισόδου και τις κατευθύνει στη μία μοναδική γραμμή εξόδου. Η επιλογή της μίας συγκεκριμένης γραμμής εισόδου γίνεται μέσω μερικών γραμμών επιλογής. Κανονικά, υπάρχουν 2^n γραμμές εισόδου και n γραμμές επιλογής που οι συνδυασμοί των bit τους καθορίζουν ποιά είσοδος επιλέγεται.



(α) Λογικό διάγραμμα

s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

(β) Πίνακας αληθείας

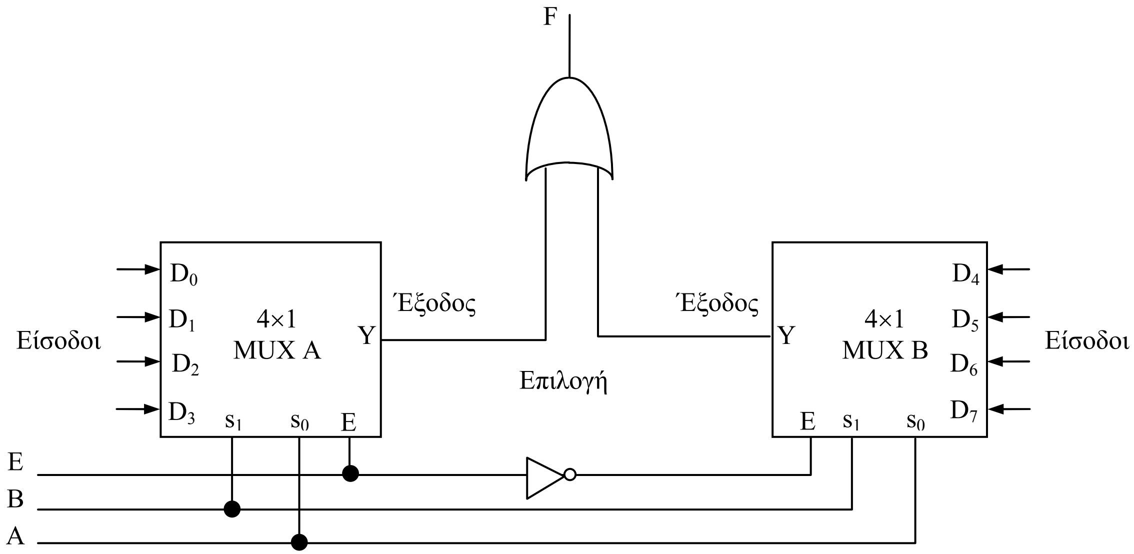


(γ) Σχηματικό διάγραμμα

Η συνάρτηση εξόδου γράφεται ως:

$$Y = \overline{s}_1\ \overline{s}_0\ I_0 + \overline{s}_1\ s_0\ I_1 + s_1\ \overline{s}_0\ I_2 + s_1\ s_0\ I_3$$

Οι πύλες AND και οι αντιστροφείς στον πολυπλέκτη μοιάζουν με το κύκλωμα του αποκωδικοποιητή και όντως αποκωδικοποιούν τις γραμμές επιλογής. Γενικά, ένας πολυπλέκτης από 2^n - σε - 1 γραμμή κατασκευάζεται από έναν αποκωδικοποιητή από *n* - σε - 2^n γραμμές, προσθέτοντας σ' αυτόν 2^n γραμμές εισόδου - μία σε καθε πύλη AND. Οι έξοδοι των πυλών AND εφαρμόζονται σε μία μοναδική πύλη OR για να δώσουν τη μία γραμμή εξόδου. Σαν μέγεθος ενός πολυπλέκτη συνήθως δίνεται ο αριθμός 2^n των γραμμών εισόδου και η μία, γραμμή εξόδου. Αυτό συνεπάγεται ότι περιέχει επίσης *n* γραμμές επιλογής. Ένας πολυπλέκτης (multiplexer) συχνά ονομάζεται συντετμημένα "MUX".



-

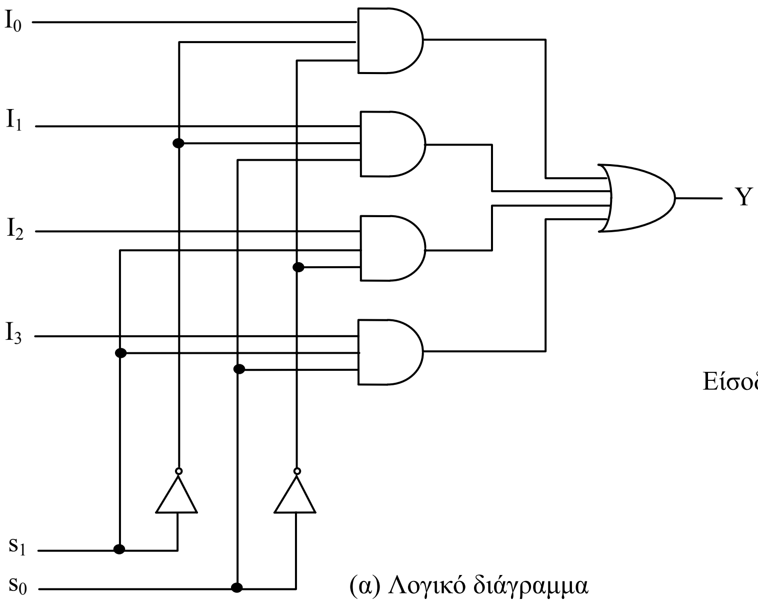
Παράδειγμα: Να υλοποιηθεί με τη βοήθεια πολυπλέκτη η συνάρτηση Boole τριών μεταβλητών:

$$F = \sum (1, 3, 5, 6)$$

(όπου οι αριθμοί στην παρένθεση αντιστοιχούν στο δεκαδικό ισοδύναμο του σχετικού ελαχιστόρου).

Λύση:

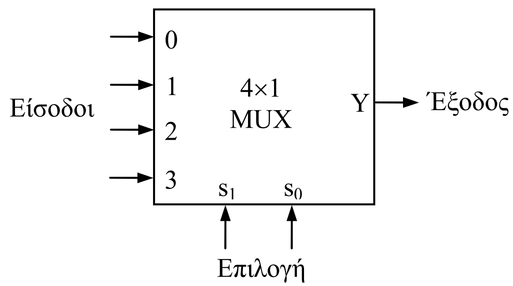
Όπως είδαμε, ένας αποκωδικοποιητής μπορεί να χρησιμοποιηθεί για την υλοποίηση μιας συνάρτησης Boole αν μεταχειριστούμε μια εξωτερική πύλη OR. Μια ματιά σε έναν πολυπλέκτη δείχνει ότι πρόκειται ουσιαστικά για έναν αποκωδικοποιητή με την πύλη OR ήδη διαθέσιμη. Εκείνους από τους ελαχιστόρους του αποκωδικοποιητή που χρειαζόμαστε μπορούμε να τους διαλέξουμε με τις γραμμές εισόδου. Τους ελαχιστόρους της συνάρτησης που θέλουμε να υλοποιήσουμε, τους επιλέγουμε κάνοντας τις αντίστοιχες εισόδους ίσες με 1, ενώ τους υπόλοιπους τους αδρανοποιούμε κάνοντας τις αντίστοιχές τους εισόδους 0. Αυτό οδηγεί σε μια μέθοδο υλοποίησης κάθε συνάρτησης Boole *n* μεταβλητών με έναν πολυπλέκτη 2^{n-1}-σε-1 γραμμών.



(α) Λογικό διάγραμμα

s_1	s_0	Y
0	0	I_0
0	1	I_1
1	0	I_2
1	1	I_3

(β) Πίνακας αληθείας

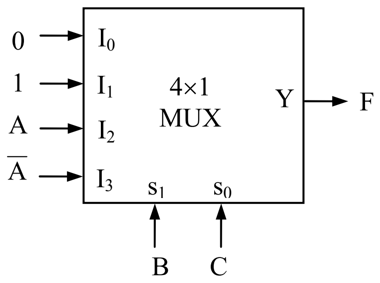


(γ) Σχηματικό διάγραμμα

Η γενική διαδικασία για τη σύνθεση του κυκλώματος υλοποίησης μιας δοσμένης συνάρτησης Boole *n* μεταβλητών με χρήση πολυπλέκτη 2^{n-1}-σε-1 γραμμών έχει ως εξής:

- Πρώτα εκφράζουμε τη συνάρτηση ως άθροισμα ελαχιστόρων. Έστω ότι η σειρά των μεταβλητών είναι ABCD…, όπου A είναι η αριστερότερη μεταβλητή και BCD… είναι οι υπόλοιπες *n* - 1 μεταβλητές.
- Συνδέουμε τις *n* - 1 μεταβλητές στις γραμμές επιλογής του πολυπλέκτη, συνδέοντας το B στην περισσότερο σημαντική γραμμή επιλογής, το C στην επόμενη, κ.ο.κ.
- Ας θεωρήσουμε τώρα τη μεταβλητή A. Αφού αυτή η μεταβλητή είναι στην υψηλότερη θέση σημαντικότητας, θα εμφανίζεται με το συμπλήρωμά της στους ελαχιστόρους 0 ως $(2^n/2)$ - 1, που είναι οι πρώτοι μισοί. Οι υπόλοιποι μισοί θα έχουν ασυμπλήρωτη τη μεταβλητή τους A. Για μια συνάρτηση λ.χ. τριών μεταβλητών A, B, C έχουμε οκτώ ελαχιστόρους. Η μεταβλητή A εμφανίζεται συμπληρωμένη στους 0 ως 3 και ασυμπλήρωτη στους 4 ως 7.
- Καταγράφουμε τις εισόδους του πολυπλέκτη και κάτω από αυτές όλους τους ελαχιστόρους σε δύο σειρές. Στην πρώτη σειρά γράφουμε όλους εκείνους, όπου το A είναι συμπληρωμένο, και στη δεύτερη αυτούς με το A ασυμπλήρωτο.
- Μετά σημειώνουμε με έναν κύκλο τους ελαχιστόρους που περιέχονται στη δοσμένη συνάρτηση και κοιτάμε τις στήλες μία-μία:
- Εάν οι δύο ελαχιστόροι σε μία στήλη δεν είναι μέσα σε κύκλο, βάζουμε 0 στην αντίστοιχη είσοδο του πολυπλέκτη.
- Εάν οι δύο ελαχιστόροι είναι μέσα σε κύκλο, βάζουμε 1 στην αντίστοιχη είσοδο του πολυπλέκτη.
- Εάν ο κάτω ελαχιστόρος έχει σημειωθεί, αλλά όχι και ο επάνω, τότε βάζουμε A στην αντίστοιχη είσοδο του πολυπλέκτη.
- Εάν είναι σημειωμένος ο επάνω ελαχιστόρος, αλλά όχι και ο κάτω, τότε τροφοδοτούμε το $\overline{A}$ στην αντίστοιχη είσοδο του πολυπλέκτη.

-



(α) Υλοποίηση με πολυπλέκτη

Ελαχι-στόρος	A	B	C	F
0	0	0	0	0
1	0	0	1	1
2	0	1	0	0
3	0	1	1	1
4	1	0	0	0
5	1	0	1	1
6	1	1	0	1
7	1	1	1	0

(β) Πίνακας αληθείας

	I_0	I_1	I_2	I_3
$\overline{A}$	0	1	2	3
A	4	5	6	7
	0	1	A	$\overline{A}$

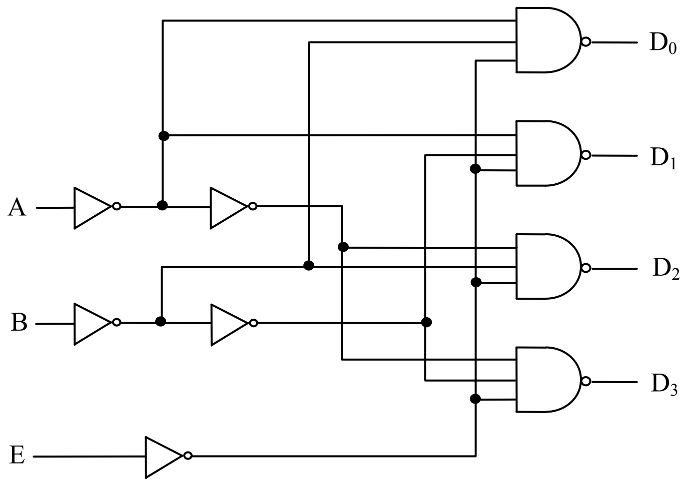
(γ) Πίνακας υλοποίησης

Υλοποίηση της F = $\sum(1,3,5,6)$ με πολυπλέκτη.

Όταν BC = 00, η έξοδος F = 0 αφού I_0 = 0. Επομένως, και οι δύο ελαχιστόροι $\overline{\mathbf{A}}\,\overline{\mathbf{B}}\,\overline{\mathbf{C}}$ και $\mathbf{A}\overline{\mathbf{B}}\,\overline{\mathbf{C}}$ δίνουν έξοδο 0, αφού η έξοδος είναι 0 όταν BC = 00, ανεξάρτητα από την τιμή του A. Όταν BC = 01 η έξοδος F = 1, αφού I_1 = 1. Επομένως, και οι δύο ελαχιστόροι $\overline{\mathbf{A}}\,\overline{\mathbf{B}}\mathbf{C}$ και $A\overline{B}C$ παράγουν έξοδο 1, αφού η έξοδος είναι 1 όταν BC = 01, ανεξάρτητα από την τιμή του A. Όταν BC = 10, επιλέγεται η είσοδος I_2. Αφού το A συνδέεται σ’ αυτή την είσοδο, η έξοδος θα είναι ίση με 1 μόνο για τον ελαχιστόρο $\mathbf{AB}\overline{\mathbf{C}}$ αλλά όχι για τον $\overline{\mathbf{A}}\mathbf{B}\overline{\mathbf{C}}$, διότι όταν $\overline{\mathbf{A}}$ = 1, τότε A = 0, και αφού I_2 = 0, έχουμε ότι F = 0. Τελικά, όταν BC = 11, επιλέγεται η είσοδος I_3. Αφού σ’ αυτή την είσοδο συνδέεται το $\overline{\mathbf{A}}$, η έξοδος θα είναι ίση με 1 μόνο για τον ελαχιστόρο $\overline{\mathbf{A}}\mathbf{BC}$, αλλά όχι και για τον ABC. Όλα αυτά συνοψίζονται στο Σχ. 5.14β που είναι ο πίνακας αληθείας της συνάρτησης που θέλουμε να υλοποιήσουμε.

Αποπολυπλέκτες (Demultiplexers)

Ο αποπολυπλέκτης είναι ένα συνδυαστικό κύκλωμα που δέχεται πληροφορίες από μία απλή γραμμή και τις μεταβιβάζει σε μία από τις 2^n δυνατές γραμμές εξόδου. Η επιλογή μιας συγκεκριμένης γραμμής εξόδου γίνεται ανάλογα με τις τιμές των *n* γραμμών επιλογής. Ο αποκωδικοποιητής του σχήματος μπορεί να λειτουργήσει και ως αποπολυπλέκτης, εάν θεωρήσουμε τη γραμμή Ε σαν είσοδο δεδομένων (πληροφοριών) και τις γραμμές Α και Β σαν γραμμές επιλογής.

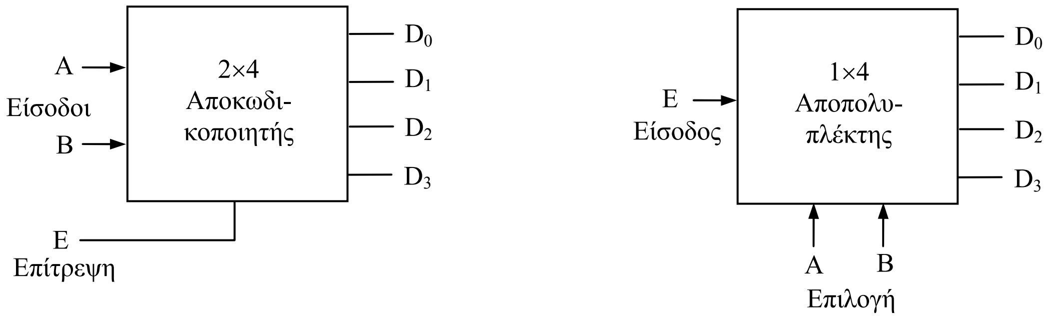


(α) Λογικό διάγραμμα

E	A	B	D_0	D_1	D_2	D_3
1	X	X	1	1	1	1
0	0	0	0	1	1	1
0	0	1	1	0	1	1
0	1	0	1	1	0	1
0	1	1	1	1	1	0

(β) Πίνακας αληθείας

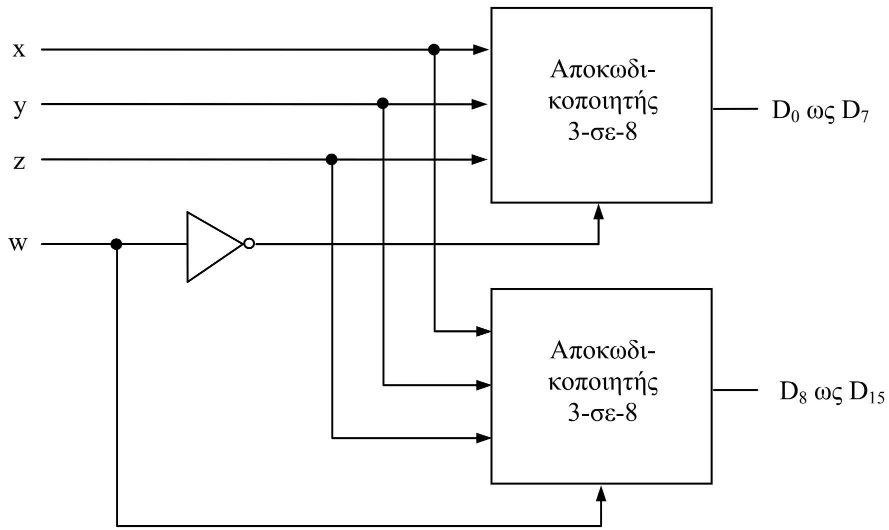
Αποκωδικοποιητής από 2 - σε - 4 γραμμές, με είσοδο επίτρεψης (Ε).



(α) Αποκωδικοποιητής με επίτρεψη

(β) Αποπολυπλέκτης

Σχηματικά διαγράμματα για το προηγούμενο κύκλωμα.

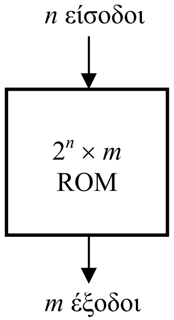


Ένας αποκωδικοποιητής 4-σε-16, υλοποιημένος από δύο αποκωδικοποιητές 3-σε-8.

Μνήμη ROM

Η μνήμη-ανάγνωσης-μόνο (Read Only Memory – ROM) είναι μια διάταξη που περιλαμβάνει τον αποκωδικοποιητή και τις πύλες OR μέσα στο ίδιο chip. Οι συνδέσεις μεταξύ των εξόδων του αποκωδικοποιητή και των εισόδων των πυλών OR μπορούν να καθοριστούν με “προγραμματισμό” της ROM. Η ROM χρησιμοποιείται πολύ συχνά για την υλοποίηση περίπλοκων συνδυαστικών κυκλωμάτων με ένα μόνο chip, καταργώντας έτσι την ανάγκη για όλα τα καλώδια διασύνδεσης.

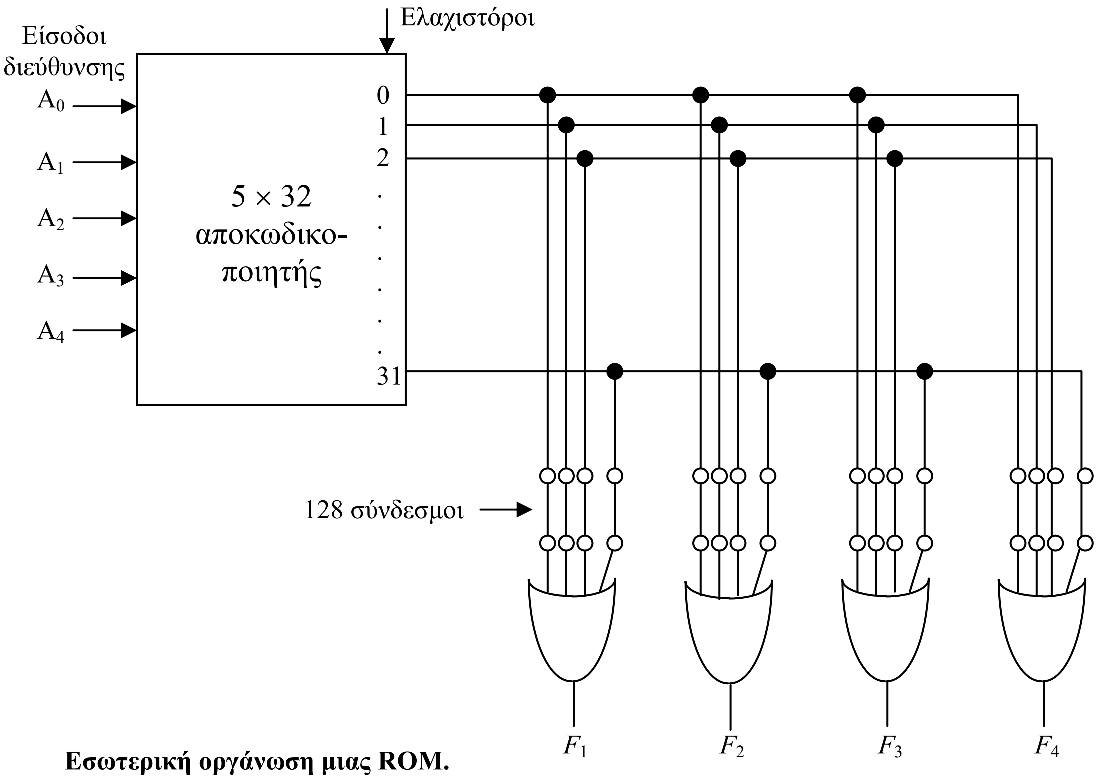
Η ROM είναι βασικά μια συσκευή μνήμης στην οποία αποθηκεύεται ένα σταθερό σύνολο δυαδικών πληροφοριών οι οποίες καθορίζονται από το χρήστη και μετά ενσωματώνονται στο chip ώστε να σχηματιστούν οι απαραίτητες διασυνδέσεις. **Οι ROM κατασκευάζονται με ειδικούς εσωτερικούς συνδέσμους τους οποίους μπορούμε αργότερα να κάψουμε ή να σπάσουμε. Μόλις οριστικοποιηθεί ένα τέτοιο σύνολο διασυνδέσεων για μια ROM, αυτό παραμένει σταθερό ακόμα και όταν “κοπεί το ρεύμα”, δηλαδή διακοπεί η τροφοδοσία.**



Σχηματικό διάγραμμα ROM.

Η ROM έχει n γραμμές εισόδου και m γραμμές εξόδου. Κάθε συνδυασμός των bits εισόδου λέγεται “διεύθυνση” (address). Κάθε συνδυασμός των bits εξόδου λέγεται “λέξη” (word). Ο αριθμός των bits κάθε λέξης ισούται με τον αριθμό των γραμμών εξόδου, δηλαδή m. Μία διεύθυνση είναι βασικά ένας δυαδικός αριθμός που δείχνει έναν από τους ελαχιστόρους των n μεταβλητών. Ο αριθμός των δυνατών διαφορετικών διευθύνσεων για n μεταβλητές εισόδου είναι 2^n. Μία λέξη εξόδου μπορεί να επιλεγεί με μία μοναδική διεύθυνση και αφού υπάρχουν 2^n διαφορετικές διευθύνσεις σε μία ROM, υπάρχουν 2^n διαφορετικές λέξεις αποθηκευμένες στο chip. Η λέξη που βγαίνει στις γραμμές εξόδου κάθε δοσμένη στιγμή εξαρτάται από τη διεύθυνση που εφαρμόζεται στις γραμμές εισόδου. Μια ROM χαρακτηρίζεται από τον αριθμό των λέξεων 2^n και από τον αριθμό των bits σε κάθε λέξη, m.

Έστω μία ROM 32×8. Η μονάδα αποτελείται από 32 λέξεις των 8 bits η καθεμία. Αυτό σημαίνει ότι υπάρχουν 8 γραμμές εξόδου και ότι υπάρχουν 32 διαφορετικές λέξεις αποθηκευμένες σ' αυτή το ROM, η καθεμία από τις οποίες μπορεί να εμφανιστεί στις γραμμές εξόδου. Η συγκεκριμένη λέξη, που θα εμφανιστεί στις γραμμές εξόδου, καθορίζεται από τις 5 γραμμές εισόδου. Υπάρχουν μόνο 5 είσοδοι σε μία ROM 32×8, διότι $2^5=32$ και με 5 μεταβλητές μπορούμε να ορίσουμε 32 διευθύνσεις ή ελαχιστόρους.

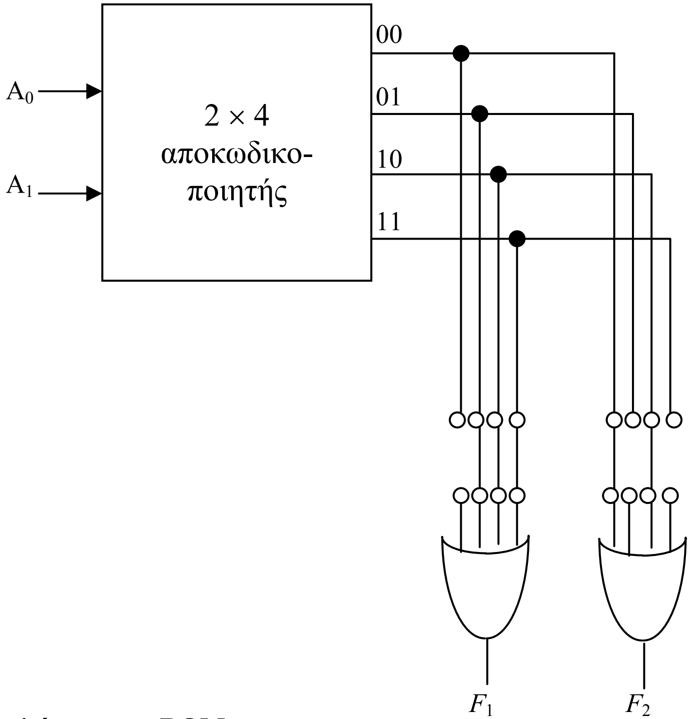


Εσωτερική οργάνωση μιας ROM.

Ο αριθμός των λέξεων σε μία ROM καθορίζεται από το γεγονός ότι χρειάζονται n γραμμές εισόδου για να ξεχωρίσουν 2^n λέξεις. Οι ROM περιγράφονται συχνά με το συνολικό αριθμό bits που περιέχουν, τα οποία είναι $2^n \times m$. Για παράδειγμα, μία ROM των 2048 bits μπορεί να είναι οργανωμένη σε 512 λέξεις των 4 bits η καθεμία. Αυτό σημαίνει ότι το chip αυτό έχει 4 γραμμές εξόδου και 9 γραμμές εισόδου για να ορίσει 2^9=512 λέξεις. Ο συνολικός αριθμός των bits που είναι αποθηκευμένα στη μονάδα είναι 512×4=2048.

Εσωτερικά, η ROM είναι ένα συνδυαστικό κύκλωμα με πύλες AND συνδεδεμένες σαν αποκωδικοποιητής και με τόσες πύλες OR, όσες και οι έξοδοι της ROM. Το Σχ. 5.19 δείχνει την εσωτερική κατασκευή μιας ROM 32×4. Οι 5 μεταβλητές εισόδου αποκωδικοποιούνται σε 32 γραμμές με τη βοήθεια 32 πυλών AND και 5 αντιστροφέων. Κάθε μία από τις 32 διευθύνσεις επιλέγει μία και μόνο έξοδο του αποκωδικοποιητή. Οι 32 έξοδοι του αποκωδικοποιητή συνδέονται με “συνδέσμους” σε κάθε πύλη OR.

Παράδειγμα



Πίνακας αληθείας

A_1	A_0	F_1	F_2
0	0	0	1
0	1	1	0
1	0	1	1
1	1	1	0

Υλοποίηση συνδυαστικού κυκλώματος με ROM.

$$F_1\left(A_1, A_0\right) = \sum(1,2,3)$$

$$F_2\left(A_1, A_0\right) = \sum(0,2)$$

<u>Προσοχή: Όταν ένα συνδυαστικό κύκλωμα υλοποιείται με ROM, οι συναρτήσεις πρέπει να εκφράζονται ως άθροισμα ελαχιστόρων.</u>

Η ROM που θα υλοποιήσει τις παραπάνω συναρτήσεις πρέπει να έχει 2 εισόδους και 2 εξόδους, άρα το μέγεθός της θα είναι 4×2. Αφού γνωρίζουμε τους ελαχιστόρους των συναρτήσεων μπορούμε εύκολα να προσδιορίσουμε ποιοι από τους 8 συνδέσμους πρέπει να κοπούν και ποιοι να παραμείνουν.

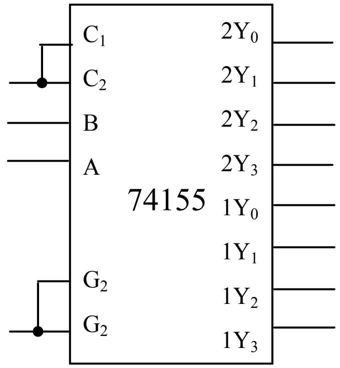
Γενική διαδικασία υλοποίησης οποιουδήποτε συνδυαστικού κυκλώματος με ROM:

- Πρώτα βρίσκουμε το μέγεθος της απαιτούμενης ROM από τον αριθμό των εισόδων και εξόδων.
- Μετά παίρνουμε τον πίνακα αληθείας της ROM ο οποίος χρησιμοποιείται κατευθείαν για τον προγραμματισμό, χωρίς άλλη αλλαγή ή απλοποίηση. Στην πράξη, όταν σχεδιάζει κανείς ένα κύκλωμα με τη βοήθεια της ROM, δεν χρειάζεται να σχεδιάσει τις εσωτερικές συνδεσμολογίες. Το μόνο που χρειάζεται να κάνει είναι να καθορίσει το συγκεκριμένο τύπο ROM (π.χ. τον κωδικό της) και να δώσει τον πίνακα αληθείας της ο οποίος περιέχει όλες τις πληροφορίες προγραμματισμού της ROM, χωρίς να χρειάζεται εσωτερικό λογικό διάγραμμα.

Δυστυχώς, τα δεδομένα που αποθηκεύονται σε μια ROM εισάγονται κατά τη διαδικασία κατασκευής και δεν μπορούν να μεταβληθούν στη συνέχεια. Ένας πολύ βολικός τύπος μνημών ROM είναι οι “σβηνόμενες προγραμματιζόμενες ROM” (Erasable Programmable Read-Only Memory, EPROM) το περιεχόμενο των οποίων μπορεί εύκολα να προγραμματιστεί και αποθηκευθεί και μπορεί επίσης να αλλάξει εάν χρειαστεί.

Παράδειγμα

Να σχεδιαστούν οι ακόλουθες λογικές συναρτήσεις χρησιμοποιώντας μόνο ένα ολοκληρωμένο κύκλωμα 74155 και ένα ολοκληρωμένο 7400:



$$F_1 = \overline{x}\cdot\overline{z}$$

$$F_2 = x\cdot\overline{y} + x\cdot\overline{y}\cdot\overline{z}$$

$$F_3 = x\cdot y\cdot\overline{z} + \overline{x}\cdot\overline{y}\cdot z$$

Το 74155 μπορεί να λειτουργήσει σαν διπλός αποκωδικοποιητής 2-σε-4 ή σαν απλός αποκωδικοποιητής 3-σε-8, οπότε οι είσοδοι C_1 και C_2, καθώς επίσης και οι είσοδοι επίτρεψης G_1 και G_2 πρέπει να είναι βραχυκυκλωμένες μεταξύ τους όπως φαίνεται στο σχήμα. Ο πίνακας αλήθειας του σχήματος περιγράφει τη λειτουργία του 74155. Επειδή εσωτερικά το ολοκληρωμένο αυτό χρησιμοποιεί πύλες NAND, η έξοδος που επιλέγεται γίνεται 0, ενώ οι υπόλοιπες μένουν στο 1. Λάβετε υπ’ όψη σας αυτή την ιδιοτυπία όταν κάνετε τη σχεδίασή σας.

Είσοδοι				**Έξοδοι**							
G	C	B	A	$2Y_0$	$2Y_1$	$2Y_2$	$2Y_3$	$1Y_0$	$1Y_1$	$1Y_2$	$1Y_3$
1	X	X	X	1	1	1	1	1	1	1	1
0	0	0	0	0	1	1	1	1	1	1	1
0	0	0	1	1	0	1	1	1	1	1	1
0	0	1	0	1	1	0	1	1	1	1	1
0	0	1	1	1	1	1	0	1	1	1	1
0	1	0	0	1	1	1	1	0	1	1	1
0	1	0	1	1	1	1	1	1	0	1	1
0	1	1	0	1	1	1	1	1	1	0	1
0	1	1	1	1	1	1	1	1	1	1	0

Λύση

Γράφουμε τις 3 συναρτήσεις σαν άθροισμα γινομένων (ελαχιστόρων). Θα έχουμε:

$$F_1 = \bar{x}\,\bar{z} = \bar{x}\,\bar{z}\left(y + \bar{y}\right) = \bar{x}y\bar{z} + \bar{x}\,\bar{y}\,\bar{z}\,.$$

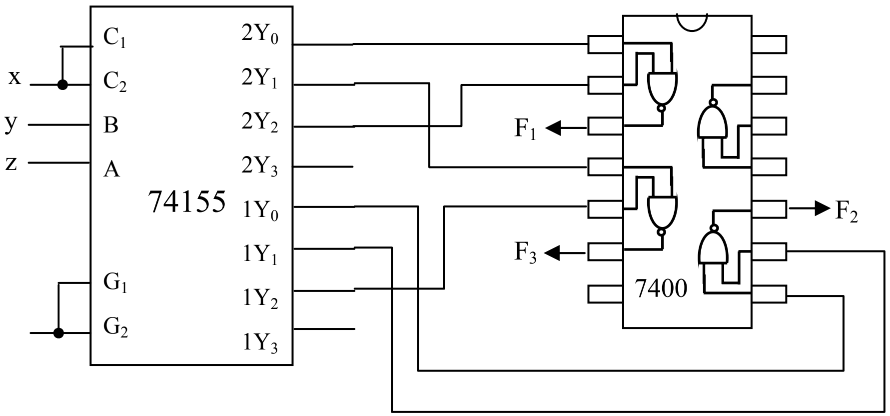
$$F_2 = x\bar{y} + x\bar{y}\,\bar{z} = x\bar{y}\left(z + \bar{z}\right) + x\bar{y}\,\bar{z} = x\bar{y}z + x\bar{y}\,\bar{z}\,.$$

$$F_3 = xy\bar{z} + \bar{x}\,\bar{y}\,z\,.$$

Η F_1 αληθεύει στα $2Y_2$ και $2Y_{0.}$

Η F_2 αληθεύει στα $1Y_1$ και $1Y_{0.}$

Η F_3 αληθεύει στα $1Y_2$ και $2Y_{1.}$ Άρα τελικά το λογικό κύκλωμα θα είναι:



-

Παράδειγμα

Ένας αποκωδικοποιητής σε 7 τμήματα είναι ένα συνδυαστικό κύκλωμα (το ολοκληρωμένο 7447) που μετατρέπει την πληροφορία 4 γραμμών εισόδου που χρειάζονται για τη δυαδική αναπαράσταση ενός μονοψήφιου δεκαδικού αριθμού σε 7 γραμμές εξόδου οι οποίες εμφανίζουν τον αντίστοιχο δεκαδικό αριθμό. Ο πίνακας αλήθειας αυτού του αποκωδικοποιητή φαίνεται στο ακόλουθο σχήμα. Αν κάποια έξοδος (a-g) είναι 0 τότε το αντίστοιχο τμήμα του ενδείκτη 7 τμημάτων είναι σβηστό, ενώ αν είναι 1 ανάβει.

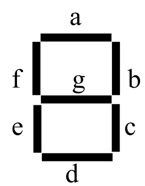
Α) Να δοθεί η συνάρτηση που παρουσιάζει την έξοδο a.

Β) Να υλοποιηθεί με τη βοήθεια αποκωδικοποιητή 4-σε-16 και ό,τι πύλες θέλετε.

Γ) Για την απεικόνιση του αριθμού 2 ποιος συνδυασμός πρέπει να συμβεί στην είσοδο του αποκωδικοποιητή και ποιες έξοδοι είναι ενεργοποιημένες;

Δ) Να υλοποιηθεί η συνάρτηση της εξόδου a με τη βοήθεια πολυπλέκτη.

D	C	B	A	a	b	c	d	e	f	g
0	0	0	0	1	1	1	1	1	1	0
0	0	0	1	0	1	1	0	0	0	0
0	0	1	0	1	1	0	1	1	0	1
0	0	1	1	1	1	1	1	0	0	1
0	1	0	0	0	1	1	0	0	1	1
0	1	0	1	1	0	1	1	0	1	1
0	1	1	0	0	0	1	1	1	1	1
0	1	1	1	1	1	1	0	0	0	0
1	0	0	0	1	1	1	1	1	1	1
1	0	0	1	1	1	1	0	0	1	1
1	0	1	0	0	0	0	1	1	0	1
1	0	1	1	0	0	1	1	0	0	1
1	1	0	0	0	1	0	0	0	1	1
1	1	0	1	1	0	0	1	0	1	1
1	1	1	0	0	0	0	1	1	1	1
1	1	1	1	0	0	0	0	0	0	0

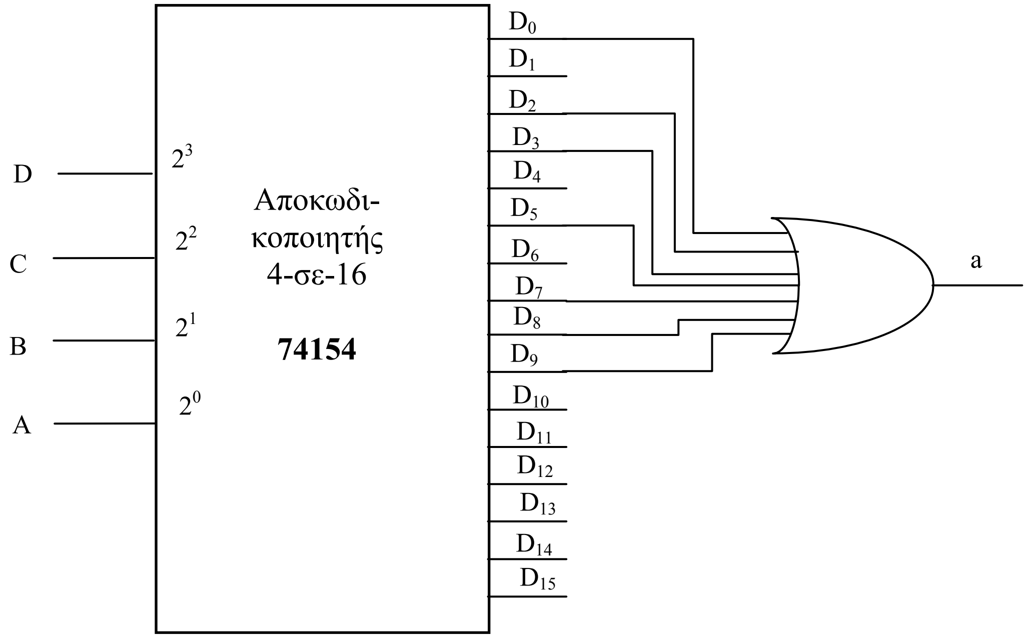


Λύση

A) Από τον πίνακα αληθείας της εξόδου a και λαμβάνοντας υπόψη μόνο τις 10 πρώτες γραμμές (συνδυασμούς) εισόδων θα έχουμε:

$$a = \overline{A}\,\overline{B}\,\overline{C}\,\overline{D} + \overline{A}\,B\,\overline{C}\,\overline{D} + AB\overline{C}\,\overline{D} + A\overline{B}C\overline{D} + ABC\overline{D} + \overline{A}\,\overline{B}\,\overline{C}D + A\overline{B}\,\overline{C}D\,.$$

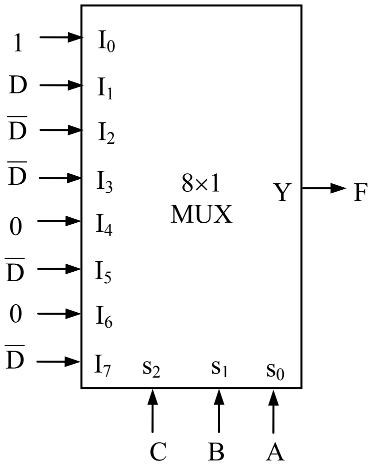
B) Ως γνωστόν, ο αποκωδικοποιητής υλοποιεί σε κάθε μία έξοδό του έναν ελαχιστόρο των γραμμών εισόδου. Κάθε δε συνάρτηση Boole γράφεται ως άθροισμα των ελαχιστόρων της. Άρα, οδηγώντας σε μια πύλη OR τους ελαχιστόρους της συνάρτησης a υλοποιούμε το λογικό κύκλωμά της.



Γ) Για την απεικόνιση του αριθμού 2 πρέπει να είναι ενεργοποιημένες στην έξοδο οι γραμμές: a, b, g, e, d. Ο συνδυασμός που υλοποιεί την ενεργοποίηση αυτών των γραμμών είναι ο συνδυασμός που αντιστοιχεί στον ελαχιστόρο D_2.

-

Δ) Σύμφωνα με τη διαδικασία υλοποίησης μιας συνάρτησης με πολυπλέκτη που παρουσιάστηκε στο Παράδειγμα 5.4 θα χρησιμοποιήσουμε πολυπλέκτη 2^3x1=8x1 δεδομένου ότι η προς υλοποίηση συνάρτηση είναι συνάρτηση 4 μεταβλητών.



(α) Υλοποίηση με πολυπλέκτη

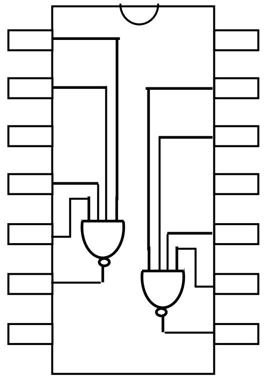
	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
$\overline{D}$	0	1	2	3	4	5	6	7
D	8	9	10	11	12	13	14	15
	1	D	$\overline{D}$	$\overline{D}$	0	$\overline{D}$	0	$\overline{D}$

(β) Πίνακας υλοποίησης

Παράδειγμα

x	y	z	w	F
0	0	0	0	1
0	0	0	1	0
0	0	1	0	1
0	0	1	1	0
0	1	0	0	1
0	1	0	1	1
0	1	1	0	1
0	1	1	1	1
1	0	0	0	0
1	0	0	1	0
1	0	1	0	0
1	0	1	1	0
1	1	0	0	0
1	1	0	1	1
1	1	1	0	0
1	1	1	1	1

Δίνεται ο πίνακας αλήθειας που φαίνεται αριστερά.



Α) Να δοθεί η συνάρτηση F σαν άθροισμα γινομένων και σαν γινόμενο αθροισμάτων.

Β) Να σχεδιάσετε το λογικό διάγραμμα της απλοποιημένης συνάρτησης με χρήση ενός 7420 του οποίου η διάταξη των πυλών παρουσιάζεται παραπλεύρως, καθώς και ενός 7400 του οποίου η περιγραφή έχει προηγηθεί.

Γ) Να σχεδιαστεί η συνάρτηση με χρήση πολυπλέκτη.

Λύση

Α) Με βάση τον πίνακα αληθείας γράφουμε τη συνάρτηση ως άθροισμα ελαχιστόρων και ως γινόμενο μεγιστόρων.

$$F = \overline{x}\,\overline{y}\,\overline{z}\,\overline{w} + \overline{x}\,\overline{y}\,z\,\overline{w} + \overline{x}\,y\,\overline{z}\,\overline{w} + \overline{x}\,y\,\overline{z}\,w + \overline{x}\,y\,z\,\overline{w} + \overline{x}\,y\,z\,w + x\,y\,\overline{z}\,w + x\,y\,z\,w\,.$$

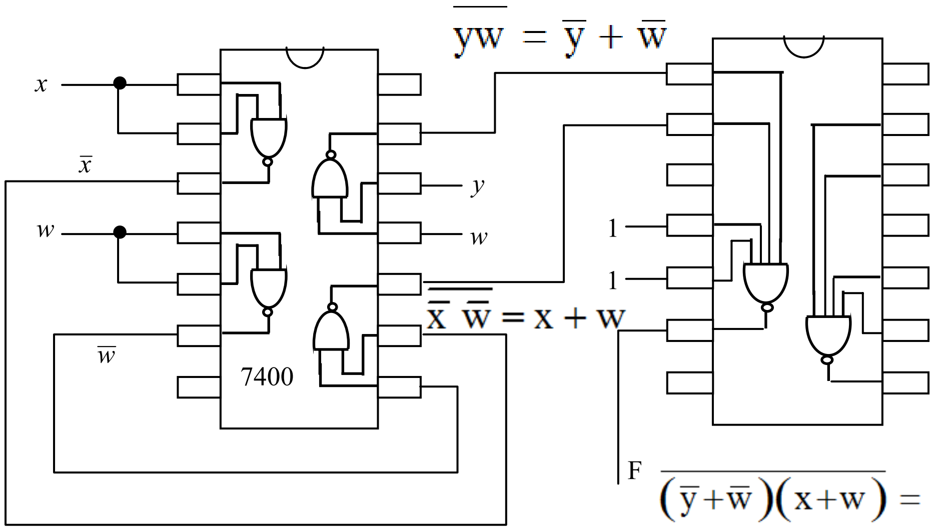
$$F = \left(x+y+z+\overline{w}\right)\left(x+y+\overline{z}+\overline{w}\right)\left(\overline{x}+y+z+w\right)\left(\overline{x}+y+z+\overline{w}\right)\left(\overline{x}+y+\overline{z}+w\right)\left(\overline{x}+y+\overline{z}+\overline{w}\right)$$
$$\left(\overline{x}+\overline{y}+z+w\right)\left(\overline{x}+\overline{y}+\overline{z}+w\right).$$

-

B) Απλοποιούμε τη δοσμένη συνάρτηση με τη βοήθεια του χάρτη Karnaugh.

xy \ zw	00	01	11	10
00	1			1
01	1	1	1	1
11		1	1	
10				

Η απλοποιημένη συνάρτηση είναι: $F = yw + \overline{x}\ \overline{w}$.

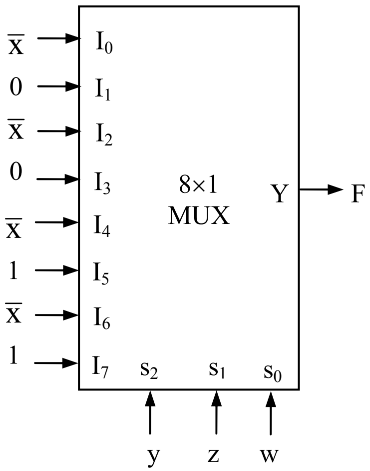


$$\overline{(\overline{y}+\overline{w})(x+w)} =$$

$$\left(\overline{\overline{y}+\overline{w}}\right) + \left(\overline{x+w}\right) =$$

$$yw + \overline{x}\ \overline{w} = F$$

Γ) Θα χρησιμοποιήσουμε πολυπλέκτη 2^3x1=8x1 δεδομένου ότι η προς υλοποίηση συνάρτηση είναι συνάρτηση 4 μεταβλητών.



(α) Υλοποίηση με πολυπλέκτη

	I_0	I_1	I_2	I_3	I_4	I_5	I_6	I_7
$\overline{x}$	0	1	2	3	4	5	6	7
x	8	9	10	11	12	13	14	15
	$\overline{x}$	0	$\overline{x}$	0	$\overline{x}$	1	$\overline{x}$	1

(β) Πίνακας υλοποίησης

Παράδειγμα

Σχεδιάστε ένα συνδυαστικό κύκλωμα που συγκρίνει 2 αριθμούς 4-bit και αποφασίζει αν οι αριθμοί είναι ίσοι μεταξύ τους, ή είναι μικρότερος ή μεγαλύτερος ο ένας σε σχέση με τον άλλον.

Λύση

Έστω οι 2 αριθμοί 4-bit, A και B. Ο συγκριτής θα έχει 3 δυαδικές εξόδους X, Y και Z που θα παριστάνουν τα αποτελέσματα της σύγκρισης A=B, A>B και A<B, αντίστοιχα. Ανάλογα με το σχετικό μέγεθος των αριθμών, η αντίστοιχη έξοδος X, Y ή Z θα δείχνει λογικό 1. Για παράδειγμα, οι δύο αριθμοί A και B που τους παριστάνουμε ως ($A_3A_2A_1A_0$) και ($B_3B_2B_1B_0$), θα είναι ίσοι (δηλαδή X=1) εάν, A_3=B_3, A_2=B_2, A_1=B_1, A_0=B_0. Για να προσδιορίσουμε αν ο A είναι μεγαλύτερος ή μικρότερος από τον B, εξετάζουμε το σχετικό μέγεθος των ζευγών bit αντίστοιχης βαρύτητας, ξεκινώντας από το πλέον σημαντικό bit. Η σύγκριση συνεχίζεται διαδοχικά στο αμέσως επόμενο ζεύγος bit χαμηλότερης σημαντικότητας εφόσον το ζεύγος bit της σημαντικότητας που εξετάσαμε είναι ίσα μεταξύ τους. Η σύγκριση συνεχίζεται μέχρι να βρεθεί ένα ζεύγος άνισων μεταξύ τους bit. Στο ζεύγος αυτό, εάν A_i=1 και B_i=0, τότε A>B, ενώ εάν A_i=0 και B_i=1, τότε A<B.
Για παράδειγμα, ο έλεγχος της κατάστασης A>B μέσω της συνάρτησης Y, δίνει 1 μόνο όταν A_i=1 και B_i=0 με A_j=B_j για j>i, δηλαδή ελέγχει από αριστερά προς τα δεξιά ένα προς ένα τα bits. Αντίστοιχα γίνεται ο έλεγχος της κατάστασης A<B, μέσω της συνάρτησης Z, η οποία δίνει 1 μόνο όταν A_i=0 και B_i=1 με A_j=B_j για j>i.
Οι εκφράσεις Boole που υλοποιούν τις συναρτήσεις X, Y και Z δίνονται από τις παρακάτω εξισώσεις:

$$X = x_3 \cdot x_2 \cdot x_1 \cdot x_0 \qquad \text{όπου} \qquad x_i = A_i \cdot B_i + \overline{A}_i \cdot \overline{B}_i$$

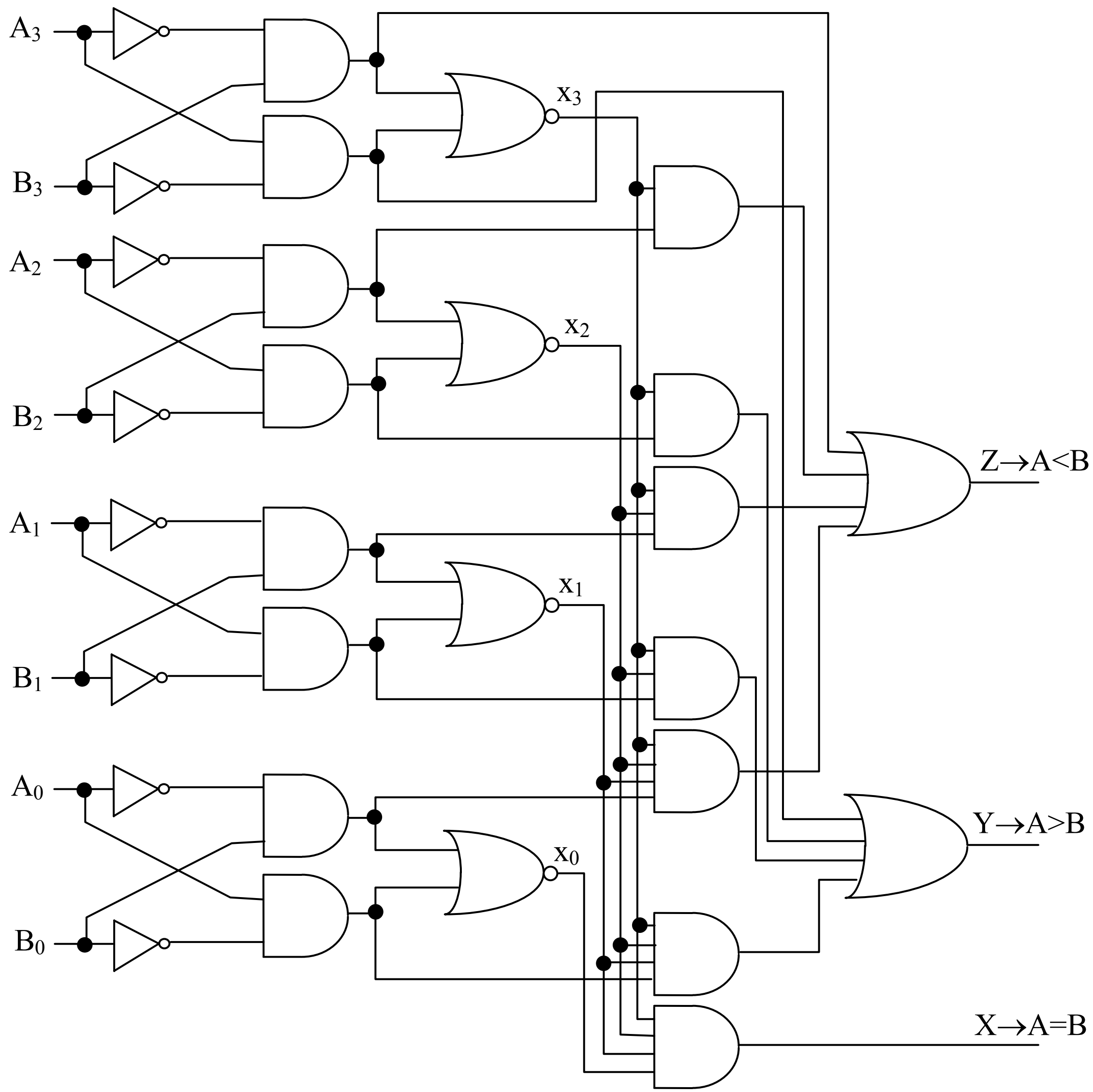
$$Y = A_3 \cdot \overline{B}_3 + x_3 \cdot A_2 \cdot \overline{B}_2 + x_3 \cdot x_2 \cdot A_1 \cdot \overline{B}_1 + x_3 \cdot x_2 \cdot x_1 \cdot A_0 \cdot \overline{B}_0$$

$$Z = \overline{A}_3 \cdot B_3 + x_3 \cdot \overline{A}_2 \cdot B_2 + x_3 \cdot x_2 \cdot \overline{A}_1 \cdot B_1 + x_3 \cdot x_2 \cdot x_1 \cdot \overline{A}_0 \cdot B_0$$

Ας εξετάσουμε την εξίσωση για το X. Η μεταβλητή x_3 θα είναι 1 μόνο όταν η A_3 και η B_3 είναι ίσες.

Ομοίως, οι συνθήκες για να είναι ίσες με 1 οι μεταβλητές x_2, x_1, και x_0, είναι να είναι ίσες οι A_2 και η B_2, οι A_1 και η B_1, και οι A_0 και η B_0, αντίστοιχα. Πολλαπλασιάζοντας μέσω πύλης AND τις x_3, x_2, x_1, και x_0, διασφαλίζουμε ότι η X θα είναι 1 όταν και οι τέσσερις μεταβλητές x_3, x_2, x_1, και x_0 είναι 1. Άρα X=1, σημαίνει ότι A=B.

Παρόμοια, μπορούμε να δούμε ότι οι εξισώσεις για τις συναρτήσεις Y και Z, ελέγχουν τις περιπτώσεις A>B και A<B αντίστοιχα. Στο Σχ. 5.21 φαίνεται το λογικό διάγραμμα του συγκριτή.



Συγκριτής δύο αριθμών 4-bit.

-

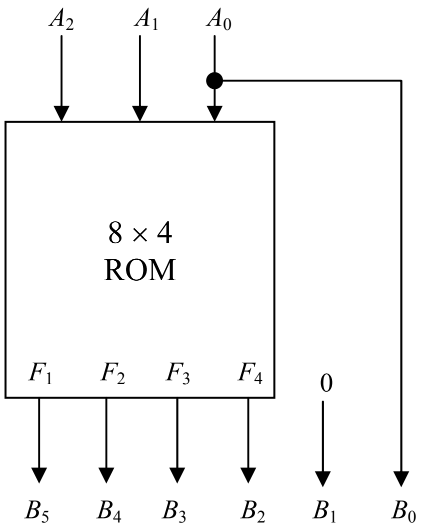
Παράδειγμα

Σχεδιάστε ένα συνδυαστικό κύκλωμα με ROM. Το κύκλωμα πρέπει να δέχεται έναν αριθμό των 3 bits και να παράγει ένα δυαδικό αριθμό στην έξοδο, ίσο με το τετράγωνο του αριθμού εισόδου.

Λύση

Στο παρακάτω σχήμα δίνεται ο πίνακας αληθείας του συνδυαστικού κυκλώματος.

Είσοδοι			Έξοδοι						Δεκαδικός
A_2	A_1	A_0	B_5	B_4	B_3	B_2	B_1	B_0	
0	0	0	0	0	0	0	0	0	0
0	0	1	0	0	0	0	0	1	1
0	1	0	0	0	0	1	0	0	4
0	1	1	0	0	1	0	0	1	9
1	0	0	0	1	0	0	0	0	16
1	0	1	0	1	1	0	0	1	25
1	1	0	1	0	0	1	0	0	36
1	1	1	1	1	0	0	0	1	49



Χρειαζόμαστε 3 εισόδους και 6 εξόδους. Παρατηρούμε ότι η έξοδος B_0 είναι πάντα ίση με την είσοδο A_0, κι έτσι δεν χρειάζεται ROM για να παράγουμε την B_0. Επίσης, η έξοδος B_1 είναι πάντα 0. Άρα τελικά η ROM χρειάζεται να παράγει μόνο τις 4 εξόδους. Έτσι, η ROM που χρειαζόμαστε πρέπει να έχει 3 εισόδους και 4 εξόδους. Οι 3 είσοδοι ορίζουν 8 λέξεις και άρα το μέγεθος της ROM πρέπει να είναι 8×4.

A_2	A_1	A_0	F_1	F_2	F_3	F_4
0	0	0	0	0	0	0
0	0	1	0	0	0	0
0	1	0	0	0	0	1
0	1	1	0	0	1	0
1	0	0	0	1	0	0
1	0	1	0	1	1	0
1	1	0	1	0	0	1
1	1	1	1	1	0	0