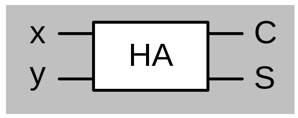
Αριθμητικά κυκλώματα

Ημιαθροιστής (Half Adder)

Ο ημιαθροιστής είναι ένα κύκλωμα το οποίο προσθέτει δύο δυαδικά ψηφία (bits) και δίνει ως αποτέλεσμα το άθροισμά τους και το κρατούμενο. Με βάση αυτή την περιγραφή, ο ημιαθροιστής έχει δύο εισόδους, έστω ***x*** και ***y***, που δέχονται τα δύο bits που προστίθενται και δύο εξόδους, μία για το άθροισμα ***S*** (sum) και μία για το κρατούμενο ***C*** (carry).



Λαμβάνοντας υπόψη ότι κατά την πρόσθεση δύο δυαδικών ψηφίων ισχύει:

	Κρατούμενο	*Άθροισμα*	
	C	***S***	
0 + 0	0	0	(= 0_{10})
0 + 1 ή 1 + 0	0	1	(= 1_{10})
1 + 1	1	0	(= 2_{10})

ο πίνακας αλήθειας του ημιαθροιστή είναι ο ακόλουθος:

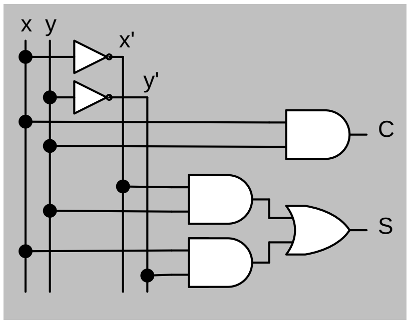
x	**y**	**C**	**S**
0	0	0	0
0	1	0	1
1	0	0	1
1	1	1	0

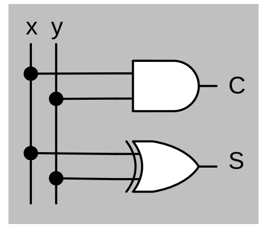
Οι λογικές συναρτήσεις των εξόδων του κυκλώματος που προκύπτουν από τον πίνακα αλήθειας είναι οι ακόλουθες:

$$C = xy$$

$$S = x'y + xy' = x \oplus y$$

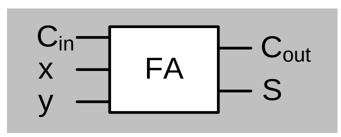
Στα ακόλουθα λογικά κυκλώματα φαίνονται δύο υλοποιήσεις του ημιαθροιστή, η πρώτη με βασικές πύλες AND, OR και NOT και η δεύτερη με τη χρήση της παράγωγης πύλης XOR.





Πλήρης αθροιστής (Full Adder)

Ο πλήρης αθροιστής είναι ένα κύκλωμα που προσθέτει δύο δυαδικά ψηφία, καθώς και κρατούμενο εισόδου που έχει προκύψει από προηγούμενη άθροιση και δίνει ως αποτέλεσμα το άθροισμα και το κρατούμενο εξόδου. Με βάση αυτή την περιγραφή, ο πλήρης αθροιστής έχει τρεις εισόδους, έστω ***x***, ***y*** και $\boldsymbol{C_{in}}$, που δέχονται τα δύο bits που προστίθενται και το κρατούμενο εισόδου και δύο εξόδους, μία για το άθροισμα ***S*** και μία για το κρατούμενο εξόδου $\boldsymbol{C_{out}}$.



Λαμβάνοντας υπόψη ότι:

	Κρατούμενο	Άθροισμα	
	C	**S**	
0 + 0	0	0	$(= 0_{10})$
0 + 1 ή 1 + 0	0	1	$(= 1_{10})$
1 + 1	1	0	$(= 2_{10})$
1 + 1 +1	1	1	$(= 3_{10})$

ο πίνακας αλήθειας του πλήρους αθροιστή είναι ο ακόλουθος:

C_{in}	x	y	C_{out}	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

$$\mathbf{C_{out} = \Sigma(m_3, m_5, m_6, m_7) = C'_{in}xy + C_{in}x'y + C_{in}xy' + C_{in}xy}$$

$$\mathbf{S = \Sigma(m_1, m_2, m_4, m_7) = C'_{in}x'y + C'_{in}xy' + C_{in}x'y' + C_{in}xy}$$

Απλοποίηση λογικών συναρτήσεων εξόδων:

C_{in} \ xy	00	01	11	10
0	0	0	1	0
1	0	1	1	1

C_{in} \ xy	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$C_{out} = C_{in}x + C_{in}y + xy$$

$$S = C'_{in}x'y + C'_{in}xy' + C_{in}x'y' + C_{in}xy$$

Όπως βλέπουμε, η λογική συνάρτηση του αθροίσματος ***S*** δεν απλοποιείται σε επίπεδο βασικών πυλών. Χρησιμοποιώντας όμως παράγωγες πύλες, η συνάρτηση για το άθροισμα μπορεί να εκφραστεί ως εξής:

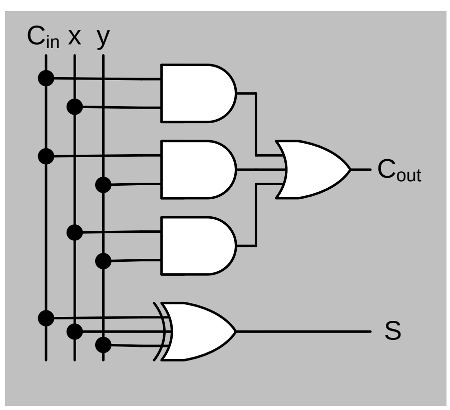
$$S = C'_{in}x'y + C'_{in}xy' + C_{in}x'y' + C_{in}xy = C'_{in}(x'y + xy') + C_{in}(x'y' + xy) =$$

$$= C'_{in}(x \oplus y) + C_{in}(x \oplus y)' = C'_{in}F + C_{in}F'$$

όπου $F = x \oplus y$. Επομένως,

$$S = C_{in} \oplus F = C_{in} \oplus (x \oplus y) = C_{in} \oplus x \oplus y$$

Το λογικό κύκλωμα του πλήρους αθροιστή είναι το ακόλουθο:



<u>Παράδειγμα 5</u>.   Να υλοποιηθεί ένας πλήρης αθροιστής με δύο ημιαθροιστές και μία πύλη OR.

Η λογική συνάρτηση του αθροίσματος του πλήρους αθροιστή μπορεί να εκφραστεί ως ακολούθως:

$$S = C_{in} \oplus x \oplus y = C_{in} \oplus (x \oplus y) = C_{in} \oplus S_1$$

όπου: $S1 = (x \oplus y)$

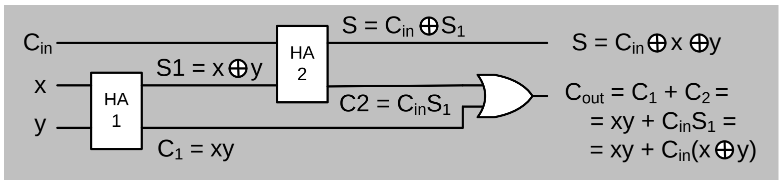
Η λογική συνάρτηση του κρατούμενου εξόδου του πλήρους αθροιστή μπορεί να εκφραστεί ως ακολούθως:

$$C_{out} = C'_{in}xy + C_{in}x'y + C_{in}xy' + C_{in}xy = (C'_{in} + C_{in})xy + C_{in}(x'y + xy') = xy + C_{in}(x \oplus y) =$$

$$= C_1 + C_{in}(x \oplus y) = C_1 + C_{in}S_1 = C_1 + C_2$$

όπου: $C_1 = xy$ ***και*** $C_2 = C_{in}S_1$

Επομένως το λογικό κύκλωμα του πλήρους αθροιστή μπορεί να υλοποιηθεί ως ακολούθως:



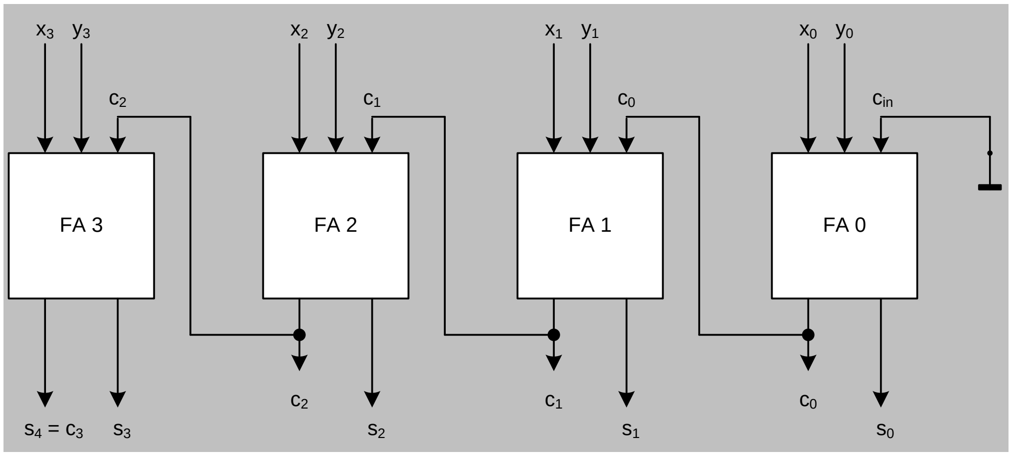
<u>Παράδειγμα 6.</u> ***Να υλοποιηθεί ένας παράλληλος δυαδικός αθροιστής των 4 bits με τέσσερις πλήρεις αθροιστές (4 bit ripple carry adder).***

Όπως έχουμε ήδη εξετάσει, η πρόσθεση δύο δυαδικών αριθμών Χ και Υ, των 4 bits ο καθένας, ακολουθεί τον κλασικό αλγόριθμο:

	c_3	c_2	c_1	c_0	
X =		x_3	x_2	x_1	x_0
+ Y =		y_3	y_2	y_1	y_0
Άθροισμα **S** =	$s_4=c_3$	s_3	s_2	s_1	s_0
Επιμέρους Κρατούμενα		c_3	c_2	c_1	c_0

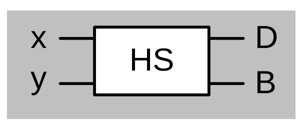
Όπως βλέπουμε, για κάθε τάξη ***v*** *προστίθενται στους συντελεστές* x_v *και* y_v *της τάξης το κρατούμενο εξόδου* c_{v-1} *της προηγούμενης τάξης και παράγονται το άθροισμα* s_v *και το κρατούμενο εξόδου* c_v *της τάξης. Να σημειωθεί ότι η είσοδος του κρατούμενου εισόδου* C_{in} *της τάξης των μονάδων (FA 0), πρέπει να πάρει τιμή λογικό "0".*

Σύμφωνα με τα παραπάνω, το ζητούμενο λογικό κύκλωμα είναι το ακόλουθο:



Ημιαφαιρέτης (Half Substractor)

Ο ημιαφαιρέτης είναι ένα κύκλωμα που αφαιρεί δύο δυαδικά ψηφία (bits) και δίνει ως αποτέλεσμα τη διαφορά τους και το δανεικό ψηφίο, το οποίο ισούται με μονάδα αν απαιτείται δανεισμός ψηφίου για να γίνει η αφαίρεση μεταξύ των δύο bits. Με βάση αυτή την περιγραφή, ο ημιαφαιρέτης έχει δύο εισόδους που δέχονται τα δύο bits που αφαιρούνται, δηλαδή μία για τον μειωτέο και μία για τον αφαιρετέο, έστω ***x*** και ***y*** αντίστοιχα, και δύο εξόδους, μία για τη διαφορά ***D*** (difference) και μία για το δανεικό ***B*** (borrow).



Λαμβάνοντας υπόψη ότι κατά την αφαίρεση δύο δυαδικών ψηφίων ισχύει:

	Δανεικό **B**	Διαφορά **D**
0 - 0	0	0
1 - 1	0	0
1 - 0	0	1
0 - 1	1	1

ο πίνακας αλήθειας του ημιαφαιρέτη είναι ο ακόλουθος:

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

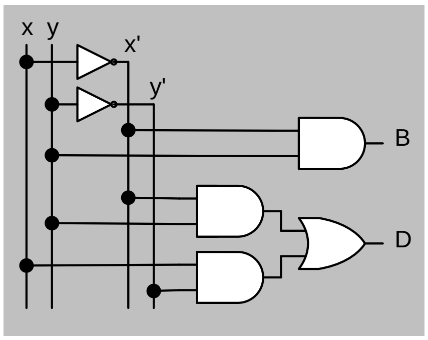
Οι λογικές συναρτήσεις των εξόδων του κυκλώματος που προκύπτουν από τον πίνακα αλήθειας είναι οι ακόλουθες:

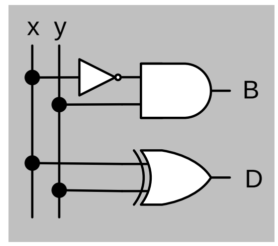
B = x'y

D = x'y + xy' = x ⊕ y

Παρατηρούμε ότι η λογική συνάρτηση για τη διαφορά ***D*** είναι η ίδια με τη λογική συνάρτηση για το άθροισμα ***S*** του ημιαθροιστή.

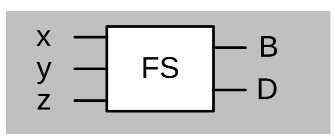
Στα ακόλουθα λογικά κυκλώματα φαίνονται δύο υλοποιήσεις του ημιαφαιρέτη, η πρώτη με βασικές πύλες AND, OR και NOT και η δεύτερη με τη χρήση της παράγωγης πύλης XOR.





Πλήρης αφαιρέτης (Full Substractor)

Ο πλήρης αφαιρέτης είναι ένα κύκλωμα που αφαιρεί δύο δυαδικά ψηφία λαμβάνοντας υπόψη ότι μπορεί στην αμέσως προηγούμενη αφαίρεση να είχε γίνει δανεισμός μιας μονάδας. Αυτό σημαίνει ότι ο πλήρης αφαιρέτης έχει τρεις εισόδους, έστω x για τον μειωτέο, y για τον αφαιρετέο και z για τυχόν δανεικό από προηγούμενη αφαίρεση, καθώς και δύο εξόδους, έστω D για τη διαφορά και B για το τυχόν νέο δανεικό που θα προκύψει.



Για να κάνουμε την αφαίρεση, προσθέτουμε στον αφαιρετέο (y) το κρατούμενο από προηγούμενη αφαίρεση (z) και το άθροισμά τους το αφαιρούμε από τον μειωτέο (x), δηλαδή: x – (y + z)

Λαμβάνοντας υπόψη ότι:

	Δανεικό	Διαφορά
	B	**D**
0 - 0	0	0
1 - 1	0	0
1 - 0	0	1
0 - 1	1	1
0 - 10	1	0
1 - 10	1	1

ο πίνακας αλήθειας του πλήρους αφαιρέτη είναι ο ακόλουθος:

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

$$\mathbf{B = \Sigma(m_1, m_2, m_3, m_7) =}$$
$$\mathbf{= x'y'z + x'yz' + x'yz + xyz}$$

$$\mathbf{D = \Sigma(m_1, m_2, m_4, m_7) =}$$
$$\mathbf{= x'y'z + x'yz' + xy'z' + xyz}$$

Απλοποίηση λογικών συναρτήσεων εξόδων:

x \ yz	00	01	11	10
0	0	1	1	1
1	0	0	1	0

x \ yz	00	01	11	10
0	0	1	0	1
1	1	0	1	0

$$\mathbf{B = x'y + x'z + yz}$$

$$\mathbf{D = x'y'z + x'yz' + xy'z' + xyz}$$

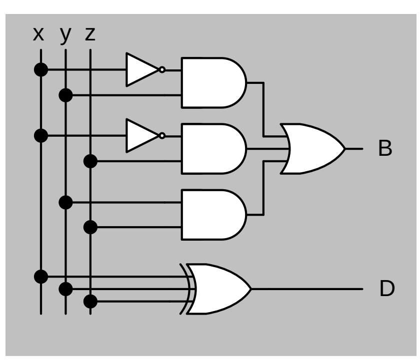
Όπως βλέπουμε, η λογική συνάρτηση της διαφοράς ***D*** δεν απλοποιείται σε επίπεδο βασικών πυλών. Χρησιμοποιώντας όμως παράγωγες πύλες, η συνάρτηση για τη διαφορά μπορεί να εκφραστεί ως εξής:

$$\mathbf{D = x'y'z + x'yz' + xy'z' + xyz = x'(y'z + yz') + x(y'z' + yz) =}$$

$$\mathbf{= x'(y \oplus z) + x(y \oplus z)' =} \text{ [} \textit{θέτοντας} \ (y \oplus z) = F \text{] } \mathbf{= x'F + xF' = x \oplus F =}$$

$$\mathbf{= x \oplus (y \oplus z) = x \oplus y \oplus z}$$

Το λογικό κύκλωμα του πλήρους αφαιρέτη είναι το ακόλουθο:



Παράδειγμα 7. ***Να δείξετε πώς μπορούμε να μετατρέψουμε έναν πλήρη αθροιστή σε πλήρη αθροιστή-αφαιρέτη.***

Η αφαίρεση δύο δυαδικών αριθμών μπορεί να γίνει με πρόσθεση στο μειωτέο, του συμπληρώματος ως προς 2 του αφαιρετέου.

Το συμπλήρωμα ως προς 2 ενός δυαδικού αριθμού ισούται με το συμπλήρωμά του ως προς 1, συν 1 ($\Sigma_{(B)} = 1 + \Sigma_{(B-1)}$).

Το συμπλήρωμα ως προς 1 ενός δυαδικού αριθμού προκύπτει με την απλή αντικατάσταση των 1 με 0 και των 0 με 1.

*Επομένως η αφαίρεση **X – Y** μπορεί να γίνει ως εξής:*

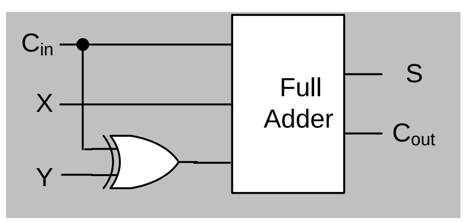
$$X - Y = X + \Sigma_{(B)}(Y) + 1 = X + Y' + 1$$

Ο Πίνακας αλήθειας της λογικής πράξης XOR είναι ο ακόλουθος:

X	Y	$Z = X \oplus Y$
0	0	0
0	1	1
1	0	1
1	1	0

*Βλέπουμε ότι όταν **X = 0**, **Z = Y** και όταν **X = 1, Z = X'**.*

*Επομένως ένας πλήρης αθροιστής μπορεί να μετατραπεί σε πλήρη αθροιστή - αφαιρέτη που υλοποιεί είτε την πρόσθεση **X + Y,** είτε την αφαίρεση **X – Y,** ως ακολούθως:*



*Στο κύκλωμα αυτό όταν $C_{in} = 0$ η έξοδος της πύλης XOR είναι **Y**, ενώ όταν $C_{in} = 1$ η έξοδος της πύλης XOR είναι **Y'**.*

*Άρα η έξοδος **S** του κυκλώματος, ανάλογα με την τιμή που δίνουμε στο C_{in}, είναι:*

για $C_{in} = 0$, $S = X + Y + 0 = X + Y$

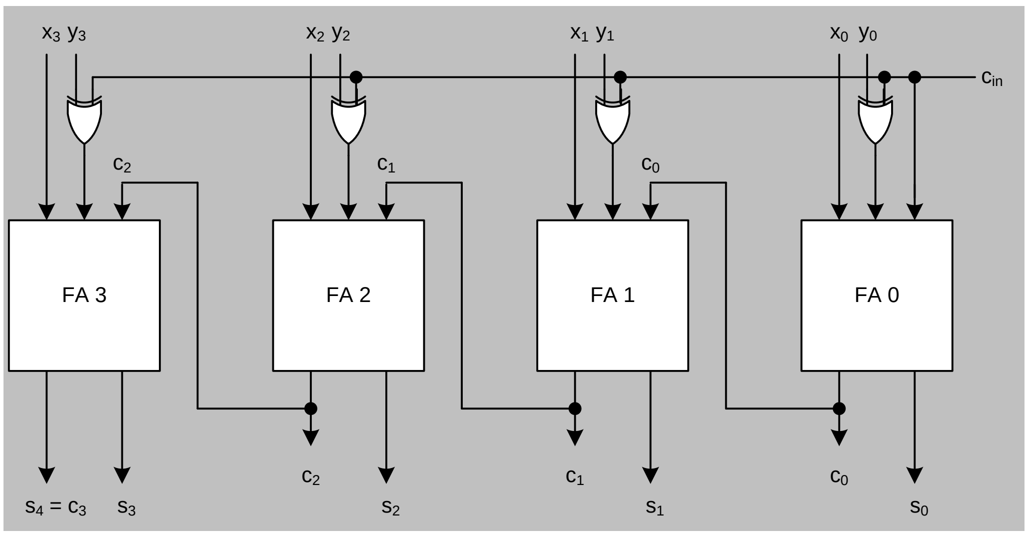
για $C_{in} = 1$, $S = X + Y' + 1 = X - Y$

Παράδειγμα 8. ***Να υλοποιηθεί ένας παράλληλος αθροιστής – αφαιρέτης των 4 bits με τέσσερις πλήρεις αθροιστές.***

Έχουμε ήδη εξετάσει στο Παράδειγμα 6 πώς μπορούμε να υλοποιήσουμε έναν παράλληλο αθροιστή των 4 bits με τέσσερις πλήρεις αθροιστές.

Επίσης εξετάσαμε στο Παράδειγμα 7 πώς μπορούμε να μετατρέψουμε έναν πλήρη αθροιστή σε πλήρη αθροιστή - αφαιρέτη με την προσθήκη μιας πύλης XOR.

Επομένως μπορούμε να υλοποιήσουμε έναν παράλληλο αθροιστή – αφαιρέτη των 4 bits με τέσσερις πλήρεις αθροιστές χρησιμοποιώντας τέσσερις πύλες XOR, μία για κάθε πλήρη αθροιστή. Η έξοδος κάθε μίας από αυτές θα οδηγεί την είσοδο $\boldsymbol{y_i}$*, για κάθε έναν αθροιστή αντίστοιχα και θα δέχεται ως εισόδους το κρατούμενο εισόδου* $\boldsymbol{C_{in}}$ *και το αντίστοιχο για κάθε τάξη ψηφίο* $\boldsymbol{y_i}$*.*



Όταν το κρατούμενο εισόδου $\boldsymbol{C_{in}}$ *έχει τιμή μηδέν (*$\boldsymbol{C_{in} = 0}$*), κάθε μία πύλη XOR θα δίνει έξοδο* $\boldsymbol{y_i}$*, ενώ όταν* $\boldsymbol{C_{in}}$ *έχει τιμή ένα (*$\boldsymbol{C_{in} = 1}$*), κάθε πύλης XOR δίνει έξοδο το συμπλήρωμα ως προς 2 του* $\boldsymbol{y_i}$*.*

Εάν θέσουμε $\boldsymbol{C_{in} = 0}$*, το κύκλωμα λειτουργεί ως παράλληλος αθροιστή των 4 bits και εκτελεί την πρόσθεση* $\boldsymbol{x_3x_2x_1x_0 + y_3y_2y_1y_0 = s_4s_3s_2s_1s_0}$*.*

Εάν θέσουμε $\boldsymbol{C_{in} = 1}$*, το κύκλωμα εκτελεί την πράξη:* $\boldsymbol{x_3x_2x_1x_0 + \Sigma_{(1)}(y_3y_2y_1y_0) + 1} = \boldsymbol{(s_4)}s_3s_2s_1s_0$*, όπου* $\boldsymbol{(s_4)}$ *είναι ψηφίο υπερχείλισης και μπορεί να παραλειφθεί. Άρα, για* $\boldsymbol{C_{in} = 1}$*, το κύκλωμα λειτουργεί ως παράλληλος αφαιρέτης των 4 bits και εκτελεί την αφαίρεση* $\boldsymbol{x_3x_2x_1x_0 - y_3y_2y_1y_0 = s_3s_2s_1s_0}$ *με τη χρήση των συμπληρωμάτων.*

Παράδειγμα 9. ***Να σχεδιαστεί λογικό κύκλωμα που συγκρίνει δύο διψήφιους δυαδικούς αριθμούς*** $\boldsymbol{X = x_1x_0}$ ***και*** $\boldsymbol{Y = y_1y_0}$ ***και αναγνωρίζει τη συνθήκη*** $\boldsymbol{X = Y}$***.***

Από την περιγραφή του κυκλώματος προκύπτει ότι πρέπει να έχει τέσσερις εισόδους, μία για κάθε ψηφίο των αριθμών X και Y, δηλαδή $\boldsymbol{x_1, x_0}$ *και* $\boldsymbol{y_1, y_0}$ *και μία έξοδο, έστω* ***F****, η οποία θα παίρνει τιμή 1 όταν ικανοποιείται η συνθήκη* ***X = Y*** *και τιμή 0 σε όλες τις άλλες περιπτώσεις.*

Για να ικανοποιείται η συνθήκη ***X = Y****, θα πρέπει να ισχύει ταυτόχρονα* $\boldsymbol{x_1 = y_1}$ *και* $\boldsymbol{x_0 = y_0}$*.*

Πίνακας αλήθειας:

X_1	X_0	Y_1	Y_0	F
0	0	0	0	1
0	0	0	1	0
0	0	1	0	0
0	0	1	1	0
0	1	0	0	0
0	1	0	1	1
0	1	1	0	0
0	1	1	1	0
1	0	0	0	0
1	0	0	1	0
1	0	1	0	1
1	0	1	1	0
1	1	0	0	0
1	1	0	1	0
1	1	1	0	0
1	1	1	1	1

Λογική συνάρτηση εξόδου:

X_1X_0 \ Y_1Y_0	00	01	11	10
00	1	0	0	0
01	0	1	0	0
11	0	0	1	0
10	0	0	0	1

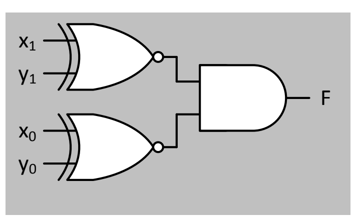
$$\boldsymbol{F = \Sigma(m_0, m_5, m_{10}, m_{15}) =}$$

$$\boldsymbol{= x'_1x'_0y'_1y'_0 + x'_1x_0y'_1y_0 + x_1x_0y_1y_0 + x_1x'_0y_1y'_0 =}$$

$$\boldsymbol{= x'_1y'_1x'_0y'_0 + x'_1y'_1x_0y_0 + x_1y_1x_0y_0 + x_1y_1x'_0y'_0 =}$$

$$\boldsymbol{= (x'_1y'_1 + x_1y_1)\,(x'_0y'_0 + x_0y_0) = (x_1 \oplus y_1)'(x_0 \oplus y_0)'}$$

Λογικό κύκλωμα:



<u>Σημείωση:</u> Χρησιμοποιώντας την ιδιότητα της λογικής πράξης XNOR να δίνει έξοδο "1" μόνο όταν οι δύο είσοδοί της είναι ίσες, μπορούμε να προσδιορίσουμε απ' ευθείας τη λογική συνάρτηση F, χωρίς να ακολουθήσουμε την κλασική μέθοδο σχεδίασης. Με βάση αυτή την ιδιότητα της λογικής πράξης XNOR και το γεγονός ότι για να ικανοποιείται η συνθήκη ***X = Y****, θα πρέπει να ισχύει ταυτόχρονα* $\boldsymbol{x_1 = y_1}$ ***<u>ΚΑΙ</u>*** $\boldsymbol{x_0 = y_0}$*, η λογική συνάρτηση της εξόδου θα είναι:* $\boldsymbol{F = (x_1 \oplus y_1)'}$ ***AND*** $\boldsymbol{(x_0 \oplus y_0)'}$*.*